VIII. CCA Security and message authentication

- Security against chosen ciphertext attacks (CCA security) considered the right notion of security for encryption schemes
- Strengthens CPA security
- Show how to achieve it for private-key encryption schemes
- Need no additional assumptions
- Use message authentication codes (MACs)
- MACs can be constructed from PRFs, hence from one-way functions

The CCA indistinguishability game

CCA indistinguishability game $\mathrm{PrivK}^{\mathrm{cca}}_{A,\Pi}(n)$

1. $k \leftarrow \mathrm{Gen}(1^n)$
2. A on input 1^n has access to encryption algorithm $\mathrm{Enc}_k(\cdot)$ and to decryption algorithm $\mathrm{Dec}_k(\cdot)$. A outputs 2 messages $m_0, m_1 \in \{0,1\}^*$ of equal length.
3. $b \leftarrow \{0,1\}$, $c \leftarrow \mathrm{Enc}_k(m_b)$. c is given to A.
4. $b' \leftarrow A(1^n, c)$, here A has access to encryption algorithm $\mathrm{Enc}_k(\cdot)$ and to decryption algorithm $\mathrm{Dec}_k(\cdot)$, but query $\mathrm{Dec}_k(c)$ is forbidden.
5. Output of experiment is 1, if $b = b'$. Otherwise output is 0.

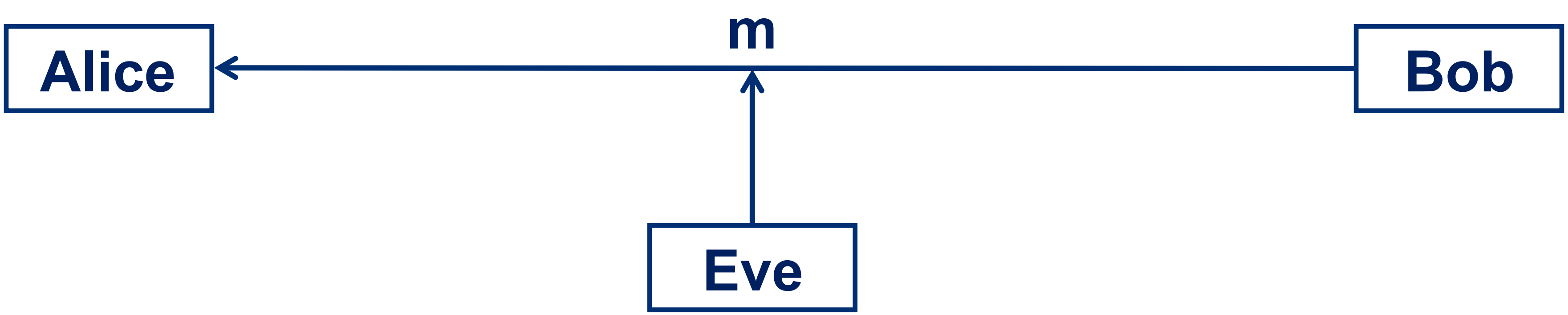
CCA-security

Definition 8.1 $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$ has indistinguishable encryptions under chosen ciphertext attacks (is cca-secure) if for every probabilistic polynomial time algorithm A there is a negligible function $\mu : \mathbb{N} \to \mathbb{R}^+$ such that

$$\Pr\left[\text{PrivK}^{\text{cca}}_{A,\Pi}(n) = 1\right] \le 1/2 + \mu(n).$$

Observation cpa-security does not imply cca-security.

Message authentication



1. Did Bob send message m, or was it Eve?
2. Did Eve modify the message m, that was sent by Bob?

Message authentication codes

Definition 8.2 A message authentication code (MAC) is a triple $\mathrm{M} = (\text{Gen}, \text{Mac}, \text{Vrfy})$ of ppts, where

1. $\text{Gen}(1^n)$ outputs a key $k \in \{0,1\}^{\geq n}$.
2. Mac takes as input a key k and a message $m \in \{0,1\}^*$, and outputs a tag t, $t \leftarrow \text{Mac}_k(m)$.
3. Vrfy takes as input a key k, a message $m \in \{0,1\}^*$, and a tag t. It outputs a bit b, $b = 1$ means valid, $b = 0$ means invalid. Vrfy assumed to be determinitic, $b := \text{Vrfy}_k(m, t)$.

For every key k and message m: $\text{Vrfy}_k(m, \text{Mac}_k(m)) = 1$.

Message authentication codes

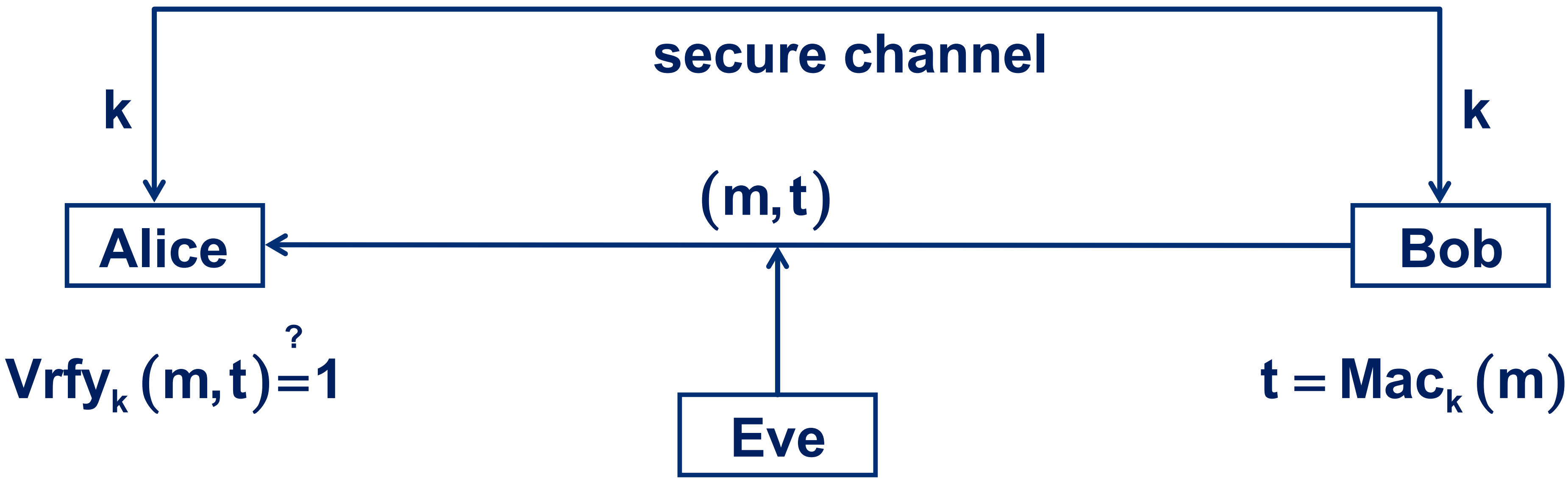
Definition 8.2 A message authentication code (MAC) is a triple $\mathrm{M} = (\text{Gen}, \text{Mac}, \text{Vrfy})$ of ppts, where

1. $\text{Gen}(1^n)$ outputs a key $k \in \{0,1\}^{\geq n}$.
2. Mac takes as input a key k and a message $m \in \{0,1\}^*$, and outputs a tag t, $t \leftarrow \text{Mac}_k(m)$.
3. Vrfy takes as input a key k, a message $m \in \{0,1\}^*$, and a tag t. It outputs a bit b, $b = 1$ means valid, $b = 0$ means invalid. Vrfy assumed to be determinitic, $b := \text{Vrfy}_k(m, t)$.

For every key k and message m: $\text{Vrfy}_k(m, \text{Mac}_k(m)) = 1$.

If Mac with $k \leftarrow \text{Gen}(1^n)$ defined only for $m \in \{0,1\}^{l(n)}$, $l : \mathbb{N} \to \mathbb{N}$ a polynomial, then M is called fixed-length MAC for messages of length $l(n)$.

Message authentication codes



The forging game

Message authentication game $\text{Mac-forge}_{A,M}(n)$

1. $k \leftarrow \text{Gen}(1^n)$.
2. A is given 1^n and oracle access to $\text{Mac}_k(\cdot)$. It outputs pair (m, t). $\mathcal{Q} := $ set of queries made by A to $\text{Mac}_k(\cdot)$.
3. Output of experiment is 1, if and only if (1) $\text{Vrfy}_k(m, t) = 1$, and (2) $m \notin \mathcal{Q}$.

Definition 8.3 $M = (\text{Gen}, \text{Mac}, \text{Vrfy})$ is called existentially unforgeable under an adaptive chosen-message attack, or secure, if for every probabilistic polynomial time adversary A there is a negligible function $\mu : \mathbb{N} \to \mathbb{R}^+$ such that

$$\Pr\left[\text{Mac-forge}_{A,M}(n) = 1\right] \leq \mu(n).$$

Construction of message authentication codes

proceeds in 2 steps

1. construct fixed-length MACs
2. design general technique to go from fixed length MACs to arbitrary MACs

1. step uses pseudorandom functions

2. step uses various techniques, e.g. hash functions (discussed in Cryptograpic Protocols)

Keyed functions

$$F: \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$$
$$(k,x) \mapsto F(k,x)$$

called keyed function. Write $F(k,x) = F_k(x)$.

- F called length-preserving, if F is only defined for $(x,k) \in \{0,1\}^* \times \{0,1\}^*$ with $|x| = |k|$ and if for all (x,k) $|F_k(x)| = |k| = |x|$.

- F called efficient, if there is a polynomial time algorithm A with $A(k,x) = F_k(x)$ for all $x,k \in \{0,1\}^*$.

- F called permutation, if for every $n \in \mathbb{N}$ and $k \in \{0,1\}^n$ $F_k : \{0,1\}^n \to \{0,1\}^n$ is bijective.

Pseudorandom function (PRF)

Definition 3.4 (restated) Let $F: \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be a keyed, efficient and length-preserving function. F is called a pseudorandom function, if for all ppt distinguishers D there is a negligible function μ such that for all $n \in \mathbb{N}$

$$\left| \Pr\left[D^{F_k(\cdot)}\left(1^n\right) = 1 \right] - \Pr\left[D^{f(\cdot)}\left(1^n\right) = 1 \right] \right| \leq \mu(n),$$

where $k \leftarrow \{0,1\}^n, f \leftarrow \mathrm{Func}_n$.

$$\mathrm{Func}_n := \left\{ f: \{0,1\}^n \to \{0,1\}^n \right\}$$

PRFs and MACs

Construction 8.4 Let $F:\{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^*$ be a keyed, efficient, and length-preserving function. Define MAC $\mathrm{M}_F = (\mathrm{Gen}_F, \mathrm{Mac}_F, \mathrm{Vrfy}_F)$ as follows:

Gen_F : on input $1^n : k \leftarrow \{0,1\}^n$.

Mac_F : on input $k, m \in \{0,1\}^n$, output $t := F_k(m)$.

Vrfy_F : on input k, m, t output 1, if and only if $t = F_k(m)$.

MAC $\mathrm{M}_F = (\mathrm{Gen}_F, \mathrm{Mac}_F, \mathrm{Vrfy}_F)$ is a fixed-length MAC for messages of length n.

PRFs and MACs

Construction 8.4 Let $F: \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be a keyed, efficient, and length-preserving function. Define MAC $\mathrm{M}_F = (\mathrm{Gen}_F, \mathrm{Mac}_F, \mathrm{Vrfy}_F)$ as follows:

Gen_F : on input $1^n : k \leftarrow \{0,1\}^n$.

Mac_F : on input $k, m \in \{0,1\}^n$, output $t := F_k(m)$.

Vrfy_F : on input k,m,t output 1, if and only if $t = F_k(m)$.

Theorem 8.5 If F is a pseudorandom function, then Construction 8.4 is secure MAC.

From forgers to distinguishers

D on input 1^n and oracle access to $f:\{0,1\}^n \to \{0,1\}^n$

1. Simulate $A(1^n)$. When A queries for a tag of $m' \in \{0,1\}^n$, answer with $t = f(m')$.
2. When A outputs a pair (m,t), do
   - Query $f(m)$ and obtain $\hat{t}$.
   - If $t = \hat{t}$ and A never queried m in Step 1, output 1, otherwise output 0.

Arbitrary length MACs

Construction 8.6 $\mathrm{M}' = (\mathrm{Gen}', \mathrm{Mac}', \mathrm{Vrfy}')$ fixed-length MAC with message length n. MAC $\mathrm{M} = (\mathrm{Gen}, \mathrm{Mac}, \mathrm{Vrfy})$ defined as:

Gen: same as Gen'.

Mac: on input $k \in \{0,1\}^n$, $m \in \{0,1\}^l$, $l < 2^{n/4}$, parse m as $m_1 \cdots m_d, m_i \in \{0,1\}^{n/4}$. $r \leftarrow \{0,1\}^{n/4}$. For $i = 1, \ldots, d$ compute $t_i \leftarrow \mathrm{Mac}'_k(r \| l \| i \| m_i)$. Output $t := (r, t_1, \ldots, t_d)$.

Vrfy: on input k,m,t output 1, if and only if $\mathrm{Vrfy}'(r \| l \| i \| m_i, t_i) = 1$ for $i = 1, \cdots, d$.

Arbitrary length MACs

Construction 8.6 $\mathcal{M}' = (\text{Gen}', \text{Mac}', \text{Vrfy}')$ fixed-length MAC with message length n. MAC $\mathcal{M} = (\text{Gen}, \text{Mac}, \text{Vrfy})$ defined as:

Gen: same as Gen′.

Mac: on input $k \in \{0,1\}^n$, $m \in \{0,1\}^l$, $l < 2^{n/4}$, parse m as $m_1 \cdots m_d, m_i \in \{0,1\}^{n/4}$. $r \leftarrow \{0,1\}^{n/4}$. For $i = 1, \ldots, d$ compute $t_i \leftarrow \text{Mac}'_k(r \| l \| i \| m_i)$. Output $t := (r, t_1, \ldots, t_d)$.

Vrfy: on input k,m,t output 1, if and only if . $\text{Vrfy}'(r \| l \| i \| m_i, t_i) = 1$ for $i = 1, \cdots, d$.

Theorem 8.7 If $\mathcal{M}'$ is a secure MAC, then $\mathcal{M}$ is a secure MAC.

Combining encryption & authentication

a) encrypt-and-authenticate

– $c \leftarrow \mathrm{Enc}_{k_1}(m), t \leftarrow \mathrm{Mac}_{k_2}(m)$

output (c, t)

b) authenticate-then-encrypt

– $t \leftarrow \mathrm{Mac}_{k_2}(m), c \leftarrow \mathrm{Enc}_{k_1}(m \,||\, t)$

output c

c) encrypt-then-authenticate

– $c \leftarrow \mathrm{Enc}_{k_1}(m), t \leftarrow \mathrm{Mac}_{k_2}(c)$

output (c, t)

a) and b) not secure, c) provably secure for MACs with unique tags.

Unique tags

Definition 8.8 A MAC $\mathrm{M} = (\text{Gen}, \text{Mac}, \text{Vrfy})$ has unique tags if for every key k and every message m there is a unique t such that $\text{Vrfy}_k(m, \text{Mac}_k(m)) = 1$.

Observation If algorithm Mac is deterministic then MAC $\mathrm{M} = (\text{Gen}, \text{Mac}, \text{Vrfy})$ has unique tags.

Encrypt-then-authenticate

$\Pi = (\mathrm{Gen}_E, \mathrm{Enc}, \mathrm{Dec})$ private-key encryption scheme,

$\mathrm{M} = (\mathrm{Gen}_M, \mathrm{Mac}, \mathrm{Vrfy})$ MAC.

Construction 8.9 $\Pi' = (\mathrm{Gen}', \mathrm{Enc}', \mathrm{Dec}')$ defined as:

$\mathrm{Gen}(1^n)$: $\quad k_1 \leftarrow \mathrm{Gen}_E(1^n), k_2 \leftarrow \mathrm{Gen}_M(1^n)$, return $k = (k_1, k_2)$.

$\mathrm{Enc}'_k(m)$: $\quad c' \leftarrow \mathrm{Enc}_{k_1}(m), t \leftarrow \mathrm{Mac}_{k_2}(c')$, return $c = (c', t)$.

$\mathrm{Dec}'_k(c)$ $\quad c = (c', t)$, if $\mathrm{Vrfy}_{k_2}(c', t) = 1$, output $\mathrm{Dec}_{k_1}(c')$.

else output $\perp$.

Theorem 8.10 If M is a secure MAC with unique tags and if Π is a CPA-secure private-key encryption scheme, then Π' is a CCA-secure private-key encryption scheme.

Encrypt-then-authenticate

$q :=$ number of queries of A to decryption oracle $\mathrm{Dec}_k(\cdot)$

valid query A queries $\mathrm{Dec}_k(\cdot)$ with some (c',t), where $\mathrm{Vrfy}_{k_2}(c',t) = 1$

new query A queries $\mathrm{Dec}_k(\cdot)$ with (c',t), where (c',t) was not obtained from $\mathrm{Enc}_k(\cdot)$

VQ := there is a query from A to $\mathrm{Dec}_k(\cdot)$ with some (c',t), where (c',t) was not obtained from $\mathrm{Enc}_k(\cdot)$ and $\mathrm{Vrfy}_{k_2}(c',t) = 1$

$(\mathrm{VQ} = j)$:= A's first valid query is the j-th new query

Encrypt-then-authenticate

$$\Pr\left[\mathrm{PrivK}^{\mathrm{cca}}_{A,\Pi'}(n)=1\right]\le\Pr\left[\mathrm{VQ}\right]+\Pr\left[\mathrm{PrivK}^{\mathrm{cca}}_{A,\Pi'}(n)=1\wedge\neg\mathrm{VQ}\right]$$

Claim 8.11 $\Pr\left[\mathrm{VQ}\right]$ is negligible.

Claim 8.12 $\Pr\left[\mathrm{PrivK}^{\mathrm{cca}}_{A,\Pi'}(n)=1\wedge\neg\mathrm{VQ}\right]-\frac{1}{2}$ is negligible.

Forger A_M

A_M on input 1^n and oracle access to $Mac_{k_2}(\cdot)$

1. $k_1 \leftarrow Gen_E, i \leftarrow \{1,\ldots,q\}$. Simulate A, where implicitly $k = (k_1, k_2)$.
2. Whenever A queries $Enc_k(\cdot)$ on message m', do

   $c' \leftarrow Enc_{k_1}(m')$, query $Mac_{k_2}(c')$ to get t, return (c', t).
3. Whenever A queries $Dec_k(\cdot)$ on ciphertext (c', t), do
   a) If (c', t) was a response to a previous encryption query for message m', answer with m'.
   b) if this is the i-th new query, then set $out := (c', t)$ and answer with $\perp$.
   c) otherwise answer with $\perp$.
4. When A returns (m_1, m_2) do

   $b \leftarrow \{0,1\}$, encrypt m_b as in 2.
5. Output out.

Encrypt-then-authenticate

query $c = (c', t)$ from A to $\mathrm{Dec}_k(\cdot)$ new if c not obtained by querying $\mathrm{Enc}_k(\cdot)$

query $c = (c', t)$ from A to $\mathrm{Dec}_k(\cdot)$ valid if $\mathrm{Verfy}_k(c', t) = 1$

VQ := there is a new and valid query from A to $\mathrm{Dec}_k(\cdot)$ (c', t), where (c', t) was not obtained from $\mathrm{Enc}_k(\cdot)$

$(\mathrm{VQ} = j)$:= A's first valid query is the j-th new query

Encrypt-then-authenticate - notation

query $c = (c', t)$ from A to $Dec_k(\cdot)$ new if c not obtained by querying $Enc_k(\cdot)$

query $c = (c', t)$ from A to $Dec_k(\cdot)$ valid if $Verfy_k(c', t) = 1$

VQ := there is a new and valid query from A to $Dec_k(\cdot)$ (c', t), where (c', t) was not obtained from $Enc_k(\cdot)$

$(VQ = j)$:= A's first valid query is the j-th new query

$(\widetilde{VQ} = j)$:= A's first valid query in simulated attack is the j-th new query in simulated attack

Observation $\Pr[VQ = j] = \Pr[\widetilde{VQ} = j]$

Forger A_E

A_E on input 1^n and oracle access to $\mathrm{Enc}_{k_1}(\cdot)$

1. $k_2 \leftarrow \mathrm{Gen}_M$. Simulate A, where implicitly $k = (k_1, k_2)$.
2. Whenever A queries $\mathrm{Enc}_k(\cdot)$ on message m', do
   query $\mathrm{Enc}_{k_1}(m')$ to get c', $t \leftarrow \mathrm{Mac}_{k_2}(c')$, return (c', t) .
3. Whenever A queries $\mathrm{Dec}_k(\cdot)$ on ciphertext (c', t), do
   a) If (c', t) was a response to a previous encryption query for message m', answer with m'.
   c) otherwise return $\perp$
4. When A returns (m_0, m_1), return (m_0, m_1) as challenge.
5. After receiving challenge ciphertext c', compute $t \leftarrow \mathrm{Mac}_{k_2}(c')$ and return $c = (c', t)$ to A.
6. Continue to simulate A.
7. Output the same bit b that A outputs.

Encrypt-then-authenticate

$\left(\widetilde{\mathsf{VQ}} = \mathsf{j}\right)$:= A's first valid query in simulated attack is the j-th new query in simulated attack

Observation $\Pr\left[\mathsf{VQ} = \mathsf{j}\right] = \Pr\left[\widetilde{\mathsf{VQ}} = \mathsf{j}\right]$

Summary

- goals and techniques of cryptography
- confidentiality and encryption schemes
- principles of modern cryptography – Kerckhoff's principle
- foundations of cryptography approach
- perfect secrecy and its characterizations
- indistinguishables encryptions and eavesdropping attacks
- pseudorandom generators and encryption schemes with indistinguishable encryptions against eavesdroppers
- multiple encryptions
- chosen plaintext attacks

Summary

- pseudorandom functions and cpa-secure encryption schemes
- block ciphers as pseudorandom permutations
- Feistel ciphers and DES
- SPNs and AES
- one-way functions and hardcore predicates
- from one-way functions to PRGs
- from PRGs to PRFs
- extension to public-key cryptography
- eavesdrooping and chosen plaintext attacks for public-key cryptography

Summary

- security for multiple encryptions
- trapdoor permutations and hardcore predicates
- from trapdoor permutations to public-key encryption
- hybrid encryption
- cca-security
- message authentication codes
- MACs from PRFs
- encrypt-then-authenticate paradigm
- encrypt-then-authenticate and cca-secure private key encryption