Announcements

• Final Exam May 10, 2024, 9-11AM, DRLB A2
• Homework:
  • Fine to collaborate, but write up your own solutions
Recap of last lecture
Formal Definition: Collision-Resistant Hash Functions

A compressing family of functions $\mathcal{H} = \{ h : \{0,1\}^m \rightarrow \{0,1\}^n \}$ (where $m > n$) for which it is computationally hard to find collisions.

**Def:** $\mathcal{H}$ is collision-resistant if for every PPT algorithm $A$, there is a negligible function $\mu$ s.t.

$$\Pr_{h \leftarrow \mathcal{H}} \left[ A(1^n, h) = (x, y) : x \neq y, h(x) = h(y) \right] = \mu(n)$$
Generic attack on C.R. functions

Let \( H: M \rightarrow \{0,1\}^n \) be a hash function \((|M| >> 2^n)\)

Generic alg. to find a collision \textbf{in time} \( O(2^{n/2}) \) hashes

Algorithm:
1. Choose \( 2^{n/2} \) random messages in \( M \): \( m_1, ..., m_{2^{n/2}} \) (distinct w.h.p)
2. For \( i = 1, ..., 2^{n/2} \) compute \( t_i = H(m_i) \in \{0,1\}^n \)
3. Look for a collision \( (t_i = t_j) \). If not found, got back to step 1.

How well will this work?
The birthday paradox

Let \( r_1, \ldots, r_n \in \{1, \ldots, B\} \) be IID integers.

**Thm:** When \( n \approx \sqrt{B} \) then \( \Pr[r_i = r_j | \exists i \neq j] \geq \frac{1}{2} \)

**Proof:** (for uniform indep. \( r_1, \ldots, r_n \))

\[
\Pr[\exists i \neq j: r_i = r_j] = 1 - \Pr[\forall i \neq j: r_i \neq r_j] = 1 - \left( \frac{B-1}{B} \right) \left( \frac{B-2}{B} \right) \cdots \left( \frac{B-n+1}{B} \right) = \\
= 1 - \frac{1}{n} \prod_{i=1}^{n-1} \left( 1 - \frac{i}{B} \right) \\
\geq 1 - \frac{1}{n} e^{-\frac{n^2}{2B}} = 1 - \left( 1 - e^{-\frac{n^2}{2B}} \right) \\
\geq 1 - e^{-0.72} = 0.53 > \frac{1}{2}
\]
Merkle-Dåmgard

Given \( h: T \times X \rightarrow T \) (compression function)

we obtain \( H: X^{\leq L} \rightarrow T \). \( H_i \) - chaining variables

PB: padding block

If no space for PB add another block

\[ \text{1000...0} \| \text{msg len} \]
HMAC

\[
\text{k} \oplus \text{ipad} \quad \text{m}[0] \quad \text{m}[1] \quad \text{m}[2] \quad \text{PB}
\]

\[
\text{h} \quad \text{h} \quad \text{h} \quad \text{h} \quad \text{h}
\]

\[
\text{IV} \quad (\text{fixed})
\]

\[
\text{k} \oplus \text{opad}
\]

\[
\text{h} \quad \text{h}
\]

\[
\text{tag}
\]

\[
\text{K}_1 \quad \text{K}_2
\]
Today

• Encryption schemes with confidentiality and integrity
• Authenticated Encryption
  • IND-CPA + Ciphertext integrity
  • IND-CCA
Story so far

Confidentiality: semantic security against a CPA attack
• Encryption secure against eavesdropping only

Integrity:
• Existential unforgeability under a chosen message attack
• CBC-MAC, HMAC, PMAC, CW-MAC

This module: encryption secure against tampering
• Ensuring both confidentiality and integrity
Sample tampering attacks

TCP/IP: (highly abstracted)

source machine

TCP/IP stack

destination machine

dest = 80 | data

data

WWW
port = 80

Bob
port = 25
Sample tampering attacks

IPsec: (highly abstracted)

TCP/IP stack

WWW
port = 80

data

stuff

k

Bob
port = 25

packets encrypted using key k
Reading someone else’s data

Note: attacker obtains decryption of any ciphertext beginning with “dest=25”

Easy to do for CBC with rand. IV (only IV is changed)
Encryption is done with CBC with a random IV.

What should IV’ be?

- IV’ = IV ⊕ (...25...)
- IV’ = IV ⊕ (...80...)
- IV’ = IV ⊕ (...80...) ⊕ (...25...)
- It can’t be done

\[ m[0] = D(k, c[0]) \oplus IV = \text{“dest=80...”} \]
The lesson

CPA security cannot guarantee secrecy under active attacks.

Only use one of two modes:

- If message needs integrity but no confidentiality:
  use a **MAC**

- If message needs both integrity and confidentiality:
  use **authenticated encryption** modes (this module)
Goals

An **authenticated encryption** system (Gen, Enc, Dec) is a cipher where

As usual: $\text{Enc} : \mathcal{K} \times \mathcal{M} \rightarrow \mathcal{C}$

but $\text{Dec} : \mathcal{K} \times \mathcal{C} \rightarrow \mathcal{M} \cup \{\bot\}$

**Security:** the system must provide

- IND-CPA, and
- **ciphertext integrity:** attacker cannot create new ciphertexts that decrypt properly
Ciphertext integrity

Let \((\text{Gen, Enc, Dec})\) be a cipher with message space \(\mathcal{M}\).

Def: \((\text{Gen, Enc, Dec})\) has **ciphertext integrity** if for all PPT \(A\):

\[
\text{Adv}_{\text{CI}}[A] = \Pr[b = 1] = \text{negl}(\lambda)
\]
Authenticated encryption

Def: \((G, E, D)\) provides **authenticated encryption** \((AE)\) if it

1. is IND-CPA secure, and
2. has ciphertext integrity

Bad example: CBC with rand. IV does not provide AE

- \(D(k, \cdot)\) never outputs \(\bot\), hence adv. easily wins CI game
Implication 1: authenticity

Attacker cannot fool Bob into thinking a message was sent from Alice

\[ c_i = Enc(k, m_i) \]

\( c \notin \{c_1, \ldots, c_q\} \)

\[ \text{if } \text{Dec}(k, c) \neq \perp \text{ Bob knows message is from someone who knows } k \]

(but message could be a replay)
Implication 2

Authenticated encryption

↓

Security against chosen ciphertext attacks
Chosen ciphertext attacks
Example chosen ciphertext attacks

Adversary $A$ has ciphertext $c$ that it wants to decrypt

- Often, $A$ can fool server into decrypting other ciphertexts (not $c$)

- Often, adversary can learn partial information about plaintext

TCP/IP packet
Chosen ciphertext security

**Adversary’s power:** both CPA and CCA
- Can obtain the encryption of arbitrary messages of his choice
- Can decrypt any ciphertext of his choice, other than challenge
  (conservative modeling of real life)

**Adversary’s goal:**

Learn partial information about challenge plaintext
Chosen ciphertext security: definition

Let \((\text{Gen}, \text{Enc}, \text{Dec})\) be a cipher with message space \(\mathcal{M}\)

Challenger

- \(k \leftarrow \text{Gen}(1^\lambda)\)
- \(b \leftarrow \{0,1\}\)

for \(i \in \{1,\ldots,q\}\):

1. **CPA query:**
   - \(m_{i,0}, m_{i,1} \in \mathcal{M} : |m_{i,0}| = |m_{i,1}|\)
   - \(c_i \leftarrow \text{Enc}(k, m_{i,b})\)

2. **CCA query:**
   - \(c_j \in \mathcal{C} : c_j \notin \{c_1, \ldots, c_i\}\)
   - \(m_j \leftarrow \text{D}(k, c_j) : m_j \in \mathcal{M} \cup \{\perp\}\)

Adversary

- \(b' \in \{0,1\}\)
Chosen ciphertext security: definition

E is CCA secure if for all “efficient” A: \( \Pr[b = b'] = 1/2 + \mu(\lambda) \)

Question: Is CBC with rand. IV CCA-secure?
Thm: Let (E,D) be a cipher that provides AE. Then (E,D) is CCA secure!

In particular, for any q-query eff. A there exist eff. B_1, B_2 s.t.

\[
\text{Adv}_{\text{CCA}}[A,E] \leq 2q \cdot \text{Adv}_{\text{CI}}[B_1,E] + \text{Adv}_{\text{CPA}}[B_2,E]
\]
Proof by pictures

Chal. $k \leftarrow K$

CPA query: $m_{i,0}, m_{i,1}$

Adv. $c_i = E(k, m_{i,0})$

CCA query: $c_j$

$D(k, c_j)$

$\approx$

Chal. $k \leftarrow K$

CPA query: $m_{i,0}, m_{i,1}$

Adv. $c_i = E(k, m_{i,0})$

CCA query: $c_j$

$\perp$

Chal. $k \leftarrow K$

CPA query: $m_{i,0}, m_{i,1}$

Adv. $c_i = E(k, m_{i,1})$

CCA query: $c_j$

$D(k, c_j)$

$\approx$

Chal. $k \leftarrow K$

CPA query: $m_{i,0}, m_{i,1}$

Adv. $c_i = E(k, m_{i,1})$

CCA query: $c_j$

$\perp$
So what?

Authenticated encryption:

- ensures confidentiality against an active adversary that can decrypt some ciphertexts

Limitations:

- does not prevent replay attacks
- does not account for side channels (timing)
Constructions of AE
... but first, some history

Authenticated Encryption (AE): introduced in 2000 [KY’00, BN’00]

Crypto APIs before then:
• Provide API for CPA-secure encryption (e.g. CBC with rand. IV)
• Provide API for MAC (e.g. HMAC)

Every project had to combine the two itself without a well defined goal
• Not all combinations provide AE ...
Combining MAC and ENC  (CCA)

Encryption key  \( k_E \).  
MAC key = \( k_M \)

Option 1: (SSL)  
\[ \text{msg } m \mapsto \text{msg } m \tag{tag } t \mapsto \text{Enc}(k_E, m || t) \]

always correct

Option 2: (IPsec)  
\[ \text{msg } m \mapsto \text{Enc}(k_E, m) \mapsto \text{MAC}(k_M, c) \tag{tag } t \]

Option 3: (SSH)  
\[ \text{msg } m \mapsto \text{Enc}(k_E, m) \mapsto \text{MAC}(k_M, m) \tag{tag } t \]
A.E. Theorems

Let \((E,D)\) be CPA secure cipher and \((S,V)\) secure MAC. Then:

1. **Encrypt-then-MAC**: always provides A.E.

2. **MAC-then-encrypt**: may be insecure against CCA attacks
   
   However: when \((E,D)\) is rand-CTR mode or rand-CBC
   M-then-E provides A.E.
Security of Encrypt-then-MAC
Standards (at a high level)

- **GCM**: CTR mode encryption then CW-MAC
  (accelerated via Intel’s PCLMULQDQ instruction)
- **CCM**: CBC-MAC then CTR mode encryption (802.11i)
- **EAX**: CTR mode encryption then CMAC

All support AEAD: (auth. enc. with associated data). All are nonce-based.
CBC paddings attacks
Recap

Authenticated encryption: CPA security + ciphertext integrity

- Confidentiality in presence of active adversary
- Prevents chosen-ciphertext attacks

Limitation: cannot help bad implementations ... (this segment)

Authenticated encryption modes:

- Standards: GCM, CCM, EAX
- General construction: encrypt-then-MAC
The TLS record protocol (CBC encryption)

Decryption: \( \text{dec}(k_{b \rightarrow s}, \text{record}, \text{ctr}_{b \rightarrow s}) : \)

step 1: CBC decrypt record using \( k_{\text{enc}} \)

step 2: check pad format: abort if invalid

step 3: check tag on \( \text{[} ++\text{ctr}_{b \rightarrow s} \text{ ll header ll data} \text{]} \)
abort if invalid

Two types of error:
- padding error
- MAC error
Padding oracle

Suppose attacker can differentiate the two errors (pad error, MAC error):

⇒ Padding oracle:

attacker submits ciphertext and learns if last bytes of plaintext are a valid pad

Nice example of a chosen ciphertext attack
Using a padding oracle \((\text{CBC encryption})\)

Attacker has ciphertext \(c = (c[0], c[1], c[2])\) and it wants \(m[1]\)
Using a padding oracle (CBC encryption)

step 1: let $g$ be a guess for the last byte of $m[1]$

iv ↓  c[0]  ↓  c[1]  ↓  g  ↓  ⊕ 0x01
|D(k,·) | D(k,·) | = last-byte ⊕ g ⊕ 0x01

if last-byte = $g$: valid pad
otherwise: invalid pad
Using a padding oracle  
(CBC encryption)

Attack: submit \((\text{IV, } c'[0], \ c[1])\) to padding oracle

\[\Rightarrow \text{attacker learns if last-byte } = g\]

Repeat with \(g = 0, 1, \ldots, 255\) to learn last byte of \(m[1]\)

Then use a \((02, 02)\) pad to learn the next byte and so on ...
Lesson

1. Encrypt-then-MAC would completely avoid this problem:
   
   MAC is checked first and ciphertext discarded if invalid

2. MAC-then-CBC provides A.E., but padding oracle destroys it
Will this attack work if TLS used counter mode instead of CBC? (i.e. use MAC-then-CTR)

- Yes, padding oracles affect all encryption schemes
- It depends on what block cipher is used
- No, counter mode need not use padding