CIS 5560

Cryptography
Lecture 14

Announcements

- MT grades will be released tomorrow/Thursday
- HW5 released tomorrow.
- Project will be released April 6.
-

Recap of last lecture

Authenticated enc. ⇒ CCA security

Thm: Let (E,D) be a cipher that provides AE.

Then (E,D) is CCA secure !

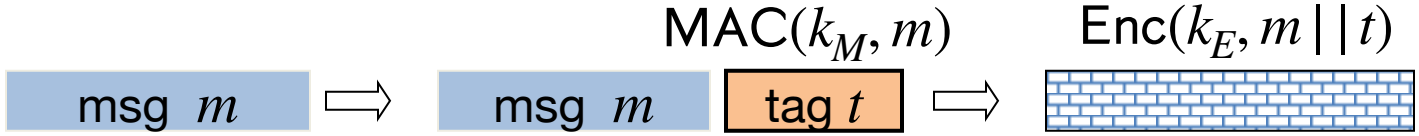
In particular, for any q-query eff. A there exist eff. B_1, B_2 s.t.

$$Adv_{CCA}[A,E] \le 2q \cdot Adv_{CI}[B_1,E] + Adv_{CPA}[B_2,E]$$

Combining MAC and ENC (CCA)

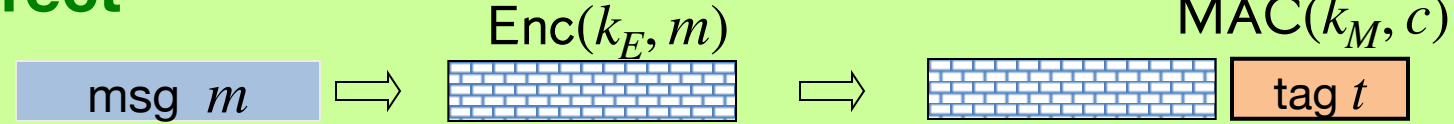
Encryption key k_E. MAC key = k_M

Option 1: (SSL)

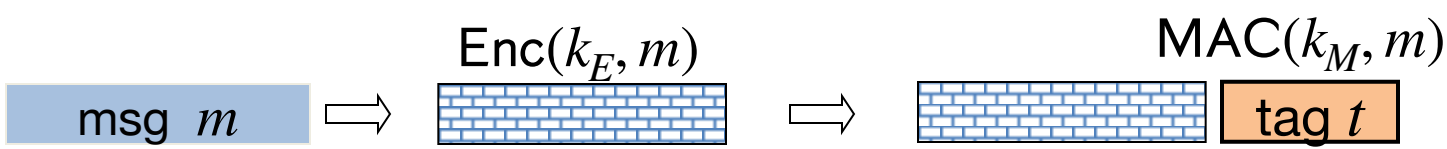


always correct

Option 2: (IPsec)



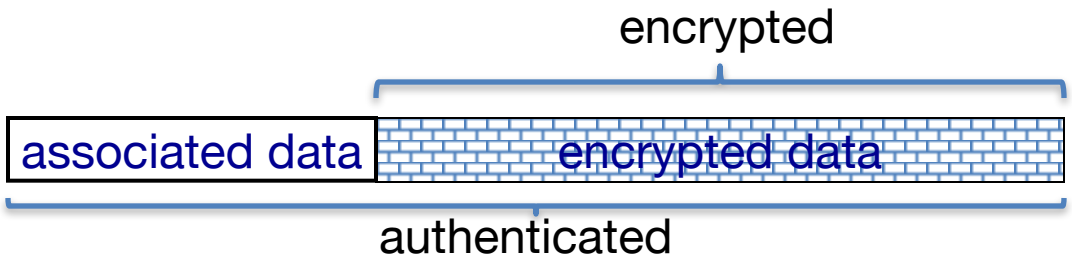
Option 3: (SSH)



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.



Universal Hashes

- We have seen MACs from PRFs and from CRHFs
- However, the fastest kind of MAC, and the one used in AES-GCM, takes a third, different, approach.
- It constructs MACs from *universal hash functions (UHF).*
- A UHF is similar to a CRHF, except that it relies on a key:

  for *all* adversaries A, the following probability is negligible:

$$Pr_{k \leftarrow \mathcal{K}}[H(k,m) = H(k,m') \mid (m,m') \leftarrow A] = \mathsf{negl}()$$

Simplest UHF

- The simplest UHF is defined based on polynomials modulo a prime as follows.
- Let p be a large prime.
- Our message space will be $\mathbb{Z}_p^{\ell+1}$ for some ℓ
- The hash function is defined as follows:

$$H(k, m = (m_0, \ldots, m_\ell)) \\ := m_0 \cdot k^0 + m_1 \cdot k^1 + m_2 \cdot k^2 + \cdots + m_\ell \cdot k^\ell$$

Today’s Lecture

- Number Theory refresher
  - Arithmetic modulo primes
  - Fermat's Little Theorem
  - Quadratic residuosity
  - Discrete Logarithms
  - Arithmetic modulo composites
  - Euler's Theorem
  - Factoring

Notation

From here on:

- N denotes a positive integer.
- p denote a prime.

Notation: $\mathbb{Z}_N = \{0,1,\ldots,N-1\}$

Can do addition and multiplication modulo N

Greatest common divisor

Def: For all $x, y \in \mathbb{Z}$, $\gcd(x, y)$ is the greatest common divisor of x, y

Example: $\gcd(12,18) = 6$

Fact: for all $x, y \in \mathbb{Z}$, there exist $a, b \in \mathbb{Z}$ such that

$a \cdot x + b \cdot y = \gcd(x, y)$

a, b can be found efficiently using the extended Euclid algorithm

If $\gcd(x, y) = 1$, we say that x and y are **relatively prime**

Modular inversion

Over the rationals, inverse of 2 is ½ . What about $\mathbb{Z}_N$?

Def: The **inverse** of $x \in \mathbb{Z}_N$ is an element $y \in \mathbb{Z}_N$ s.t.

$$x \cdot y = 1 \mod N$$

y is denoted x^{-1}.

Example: let N be an odd integer. What is the inverse of $2 \mod N$?

Modular inversion

Which elements have an inverse in $\mathbb{Z}_N$?

Lemma: $x \in \mathbb{Z}_N$ has an inverse if and only if $\gcd(x, N) = 1$

Proof:

$$\gcd(x, N) = 1 \implies \exists a, b : a \cdot x + b \cdot N = 1$$
$$\implies a \cdot x = 1 \mod N$$

$$\gcd(x, N) \neq 1 \implies \forall a : \gcd(a \cdot x, N) > 1 \Rightarrow a \cdot x \neq 1$$

Invertible elements

Def: $\mathbb{Z}_N^*$ = set of invertible elements in $\mathbb{Z}_N$

$$= \{x \in \mathbb{Z}_N : \gcd(x, N) = 1\}$$

Examples:

1. for prime p, $\mathbb{Z}_p^* := \{0, \ldots, p-1\}$
2. $\mathbb{Z}_{12}^* := \{1,5,7,11\}$

For $x \in \mathbb{Z}_N$, we can find x^{-1} using extended Euclid algorithm.

Solving modular linear equations

Solve: $a \cdot x + b = 0$, where $a, x, b \in \mathbb{Z}_N$

Solution: $x = -b \cdot a^{-1} \mod N$

Find a^{-1} using extended Euclid algorithm.

Run time: $O(\log^2 N)$

Fermat's theorem (1640)

Thm: Let p be a prime. Then,

$$\forall x \in \mathbb{Z}_p^* : x^{p-1} = 1 \mod p$$

Example: $p = 5$. Then $3^4 = 81 = 1 \mod 5$

How can we use this to compute inverses?

$$x \in \mathbb{Z}_p^* \Rightarrow x \cdot x^{p-2} = 1 \Rightarrow x^{-1} = x^{p-2}$$

(less efficient than Euclid)

Application: generating random primes

Suppose we want to generate a large random prime

say, prime p of length 1024 bits (i.e. $p \approx 2^{1024}$)

Step 1: sample $p \in [2^{1024}, 2^{1025} - 1]$

Step 2: test if $2^{p-1} = 1 \mod p$

If so, output p and stop. If not, goto step 1 .

Simple algorithm (not the best).

$\Pr[p \notin \text{PRIMES} \mid \text{test passes}] < 2^{-60}$

The structure of $\mathbb{Z}_p^*$

Thm (Euler): $\mathbb{Z}_p^*$ is a **cyclic group**, that is

$$\exists g \in \mathbb{Z}_p^* \text{ such that } \{1, g, g^2, g^3, \ldots, g^{p-2}\} = \mathbb{Z}_p^*$$

g is called a **generator** of $\mathbb{Z}_p^*$

Example: $p = 7$. $\{1, 3, 3^2, 3^3, 3^4, 3^5\} = \{1, 3, 2, 6, 4, 5\} = \mathbb{Z}_7^*$

Not every element is a generator: $\{1, 2, 2^2, 2^3, 2^4, 2^5\} = \{1, 2, 4\}$

Order

For $g \in \mathbb{Z}_p^*$ the set $\{1, g, g^2, g^3, \ldots\}$ is called

the **group generated by** g, denoted $\langle g \rangle$

Def: the **order** of $g \in \mathbb{Z}_p^*$ is the size of $\langle g \rangle$

$$\text{ord}_p(g) \;=\; |\langle g \rangle| \;=\; \textbf{smallest } a > 0 \textbf{ s.t. } g^a = 1 \mod p$$

Examples: $\text{ord}_7(3) = 6$; $\text{ord}_7(2) = 3$; $\text{ord}_7(1) = 1$

Thm (Lagrange): $\forall g \in \mathbb{Z}_p^*$: $\text{ord}_p(g)$ divides $p - 1$

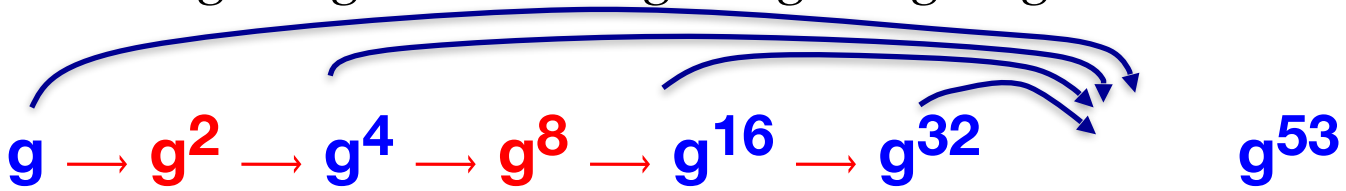
Exponentiation

Finite cyclic group $\mathbb{G}$ (for example $\mathbb{G} = \mathbb{Z}_p^*$)

Goal: given $g \in \mathbb{G}$ and x, compute g^x.
How: **Repeated Squaring Algorithm.**

Example: suppose $x = 53 = (110101)_2 = 32+16+4+1$

Then: $g^{53} = g^{32+16+4+1} = g^{32} \cdot g^{16} \cdot g^4 \cdot g^1$



The Discrete Logarithm (DL) problem

Let $\mathbb{G}$ be a finite cyclic group and g a generator of $\mathbb{G}$

Def: We say that **DLOG is hard in** $\mathbb{G}$ if for all efficient algorithms A:

$$\Pr_{h \leftarrow G, x \leftarrow \mathbb{Z}_q} [A(\mathbb{G}, q, h, h^x) = x] < \mathsf{negl}(q)$$

Example candidates:

(1) $\mathbb{Z}_p^*$ for large p,

(2) Elliptic curve groups mod p

How difficult is DL?

- Baby Step-Giant Step algorithm: time and space $O(\sqrt{p})$.
- Pohlig-Hellman algorithm: time $O(\sqrt{q})$ where q is the largest prime factor of the order of group (e.g. $p-1$ for $\mathbb{Z}_p^*$).
  That is, *there are DL-easy primes.*

Computing DL in $\mathbb{Z}_p^*$ (n-bit prime p)

Best known algorithm (GNFS): run time ~ $2^{O(\sqrt[3]{n})}$

cipher key size	modulus size	EC group size
80 bits	1024 bits	160 bits
128 bits	3072 bits	256 bits
256 bits (AES)	**15360** bits	512 bits

As a result: we have largely moved from $\mathbb{Z}_p^*$ to elliptic curves

One-way Permutation (Family)

$$F\big(p, g, x\big) = (p, g, g^x \bmod \mathrm{p})$$

$$\mathscr{F}_n = \{F_{n,p,g}\} \text{ where } F_{n,p,g}(x) = (p, g, g^x \bmod \mathrm{p})$$

Theorem: Under the discrete log assumption, F is a one-way permutation (resp. $\mathscr{F}_n$ is a one-way permutation family).

An application: collision resistance

Choose a group G where DL is hard (e.g. $(Z_p)^*$ for large p)

Let q = |G| be a prime. Choose generators g, h of G

For $x,y \in \{1,\ldots,q\}$ define $\boxed{H(x,y) = g^x \cdot h^y}$ in G

Lemma: finding collision for H(.,.) is as hard as computing $\text{Dlog}_g(h)$

Proof: Suppose we are given a collision $H(x_0,y_0) = H(x_1,y_1)$

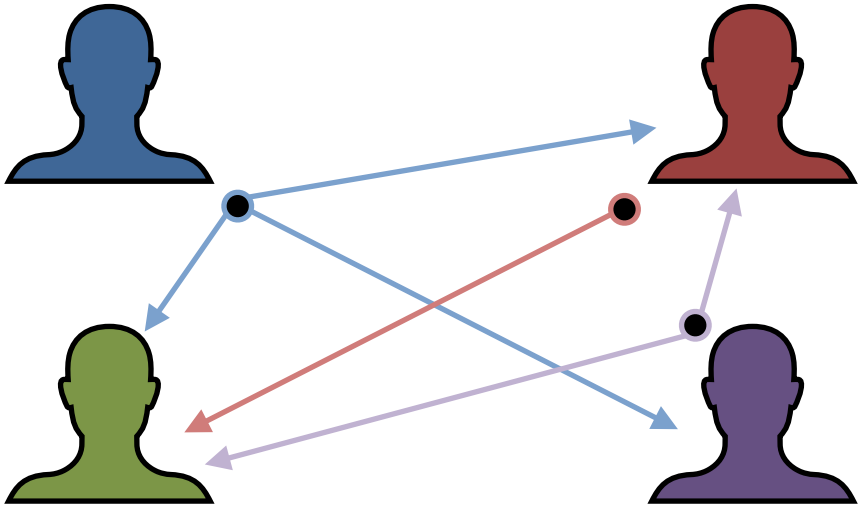
$\neq 0$

then $g^{x_0} \cdot h^{y_0} = g^{x_1} \cdot h^{y_1} \Rightarrow g^{x_0 - x_1} = h^{y_1 - y_0} \Rightarrow h = g^{(x_0 - x_1)/(y_1 - y_0)}$

Key management

Goal: n users want to communicate with each other securely.

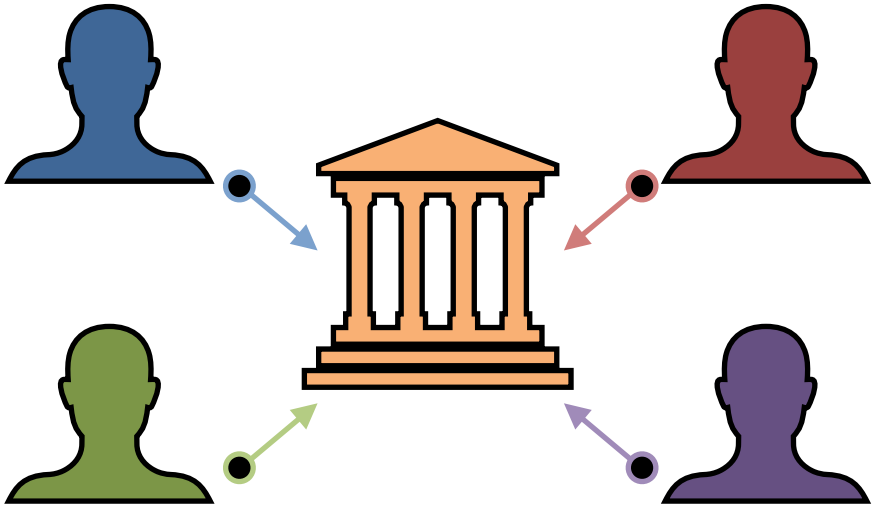
Problem: Storing mutual secret keys is onerous!



Total: $O(n)$ keys per user

Solution 1: Online Trusted Party

All parties have a *single key* with an online Trusted 3rd Party 🏛



They communicate securely with TTP, which produces pairwise keys.

Generating keys: a toy protocol

Alice wants a shared key with Bob.

Bob (k_B)

Alice (k_A)

TTP

"Alice wants key with Bob"

Sample k_{AB}

$\text{Enc}(k_A, (\text{"A||B"}, k_{AB})$

$\text{ticket} = \text{Enc}(k_B, (\text{"A||B"}, k_{AB})$

ticket

k_{AB}

k_{AB}

(Enc, Dec) a CPA-secure cipher

Generating keys: a toy protocol

Alice wants a shared key with Bob. Eavesdropping security only.

Eavesdropper sees: $\text{Enc}(k_A, (\text{"A||B"}, k_{AB}))$; $\text{ticket} = \text{Enc}(k_B, (\text{"A||B"}, k_{AB}))$

(Enc, Dec) is CPA-secure $\Rightarrow$ eavesdropper learns nothing about k_{AB}

Note: TTP needed for every key exchange, knows all session keys.

(basis of Kerberos system)

Toy protocol: insecure against active attacks

Example: insecure against replay attacks

Attacker records session between Alice and merchant Bob

- For example a book order

Attacker replays session to Bob

- Bob thinks Alice is ordering another copy of book

Key question

Can we generate shared keys without an **online** trusted 3rd party?

Answer: yes!

Starting point of public-key cryptography:

- Merkle (1974), Diffie-Hellman (1976), RSA (1977)
- More recently: ID-based enc. (BF 2001), Functional enc. (BSW 2011)

The Diffie—Hellman protocol (informally)

Fix a DL-hard group $\mathbb{G}$ of prime order p. Fix generator g of $\mathbb{G}$

Alice

choose random a in $\{1,\ldots,p-1\}$

Bob

choose random b in $\{1,\ldots,p-1\}$

"Alice", $A := g^a$

"Bob", $B := g^b$

$$B^a = g^{ba} = k_{AB} = g^{ab} = g^{ab} = A^b$$