Lecture 9

Discrete logarithm problem (DLP):

Given safe prime p ($2q+1 = p$ for prime q) where p is a large prime ($|p| \geq 2048$ bits), it is infeasible to compute x such that $y = g^x \bmod p$ given $\langle g, y, p, q \rangle$

Mod p:

* every element of Z^*_p or $[1 \ldots p-1]$ generates a set of integers.
* every generated set will be of one of four sizes (in general, every divisor of $p-1$)
  * 1
  * $p-1$
  * 2
  * q ($= \frac{p-1}{2}$)

* Every generator of q integers is in the set G_q.

* Every member of G_q generates a set of q integers.

* How large should p be for DLP to be hard

* NIST $|p| \geq 2048$
  - ↳ DLP takes 2^{112}

* For safe primes: $|q| = 2047$
  - ↳ can use smaller q with a non-safe prime
    - ↳ not covered
    - ↳ $|q| \geq 224$
      - ↑ Birthday Paradox Alg.

* Elliptic curves (over a finite field)
  - ↳ form a group of points
  - ↳ point doubling instead of multiplication
  - ↳ DLP-style hard problem.
  - ↳ faster and smaller than G_q.
  - ↳ anything you do in G_q, you can do over elliptic curves (EC)

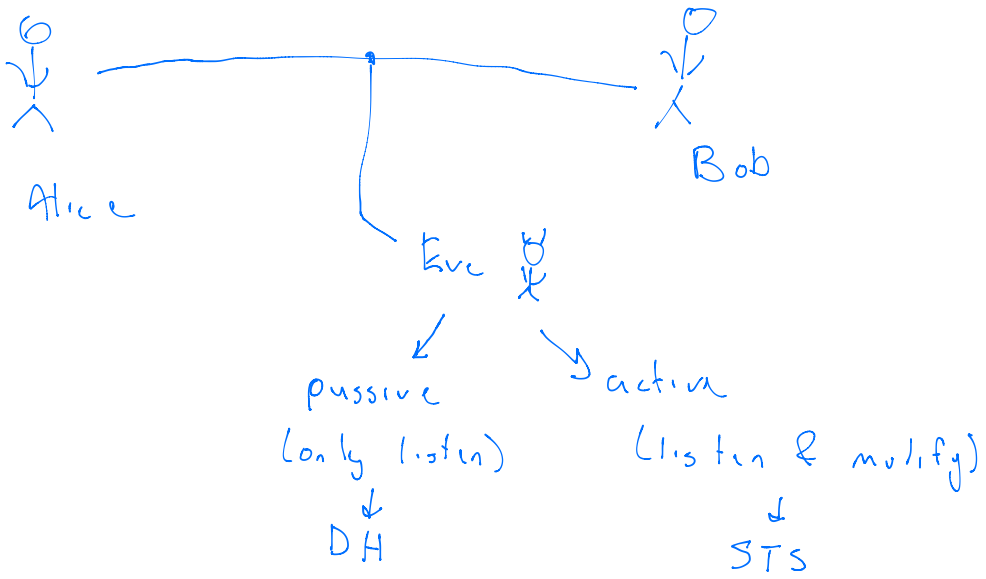
DHE → ECDHE

DSA → ECDSA

↑

this class

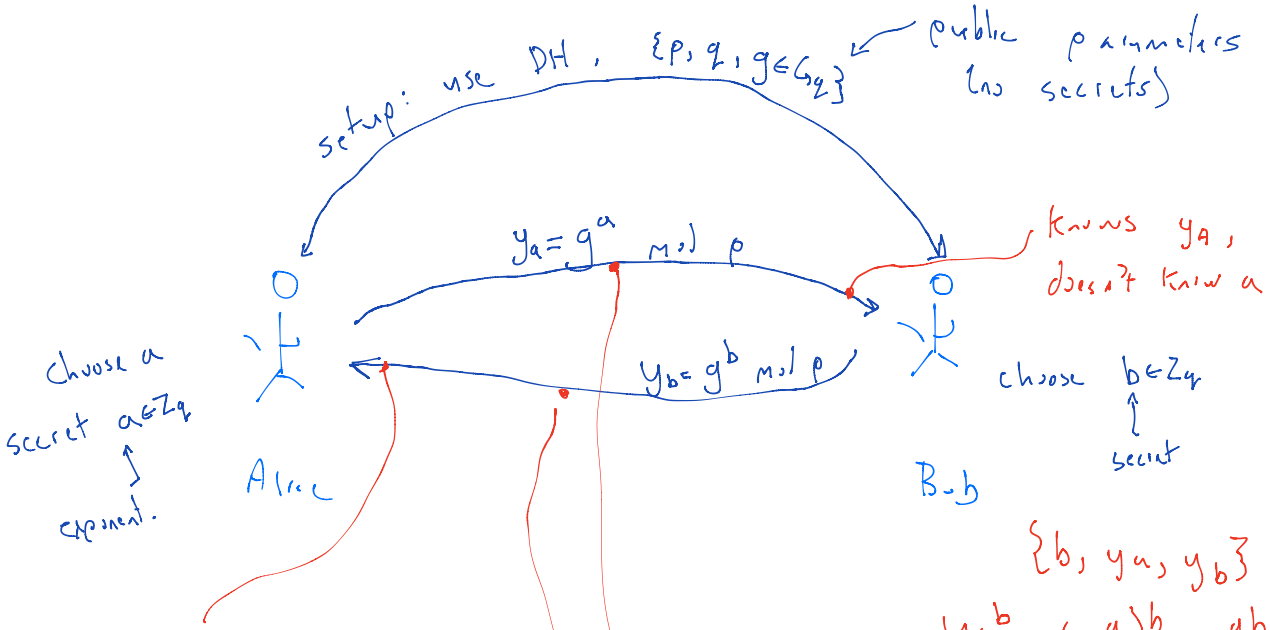
Key Exchange



Diffie-Hellman Key Exchange (DH / DHE)

* Assumptions.
  1. Eve is passive. (fix w/ STS using signatures)
  2. Alice and Bob know the are talking to eachother (fix w/ PKI)

* Goal is for Alice and Bob to compute a number (or key) that only they know and Eve doesn't learn even though she listens to the whole thing.



knows y_b, doesn't know b.

$\{a, y_a, y_b\}$

↳ $y_b^a = (g^b)^a = g^{ab}$

$\{b, y_a, y_b\}$

$y_a^b = (g^a)^b = g^{ab}$

Adv knows y_a, does not know a

Adv knows y_b, doesn't know b.

↳ $\{y_a, y_b\}$

↳ y_a ↳ y_b

no known way to compute g^{ab}

↳ computational Diffie-Hellman problem (CDH)

↳ break DLP ⇒ break CDH

break CDH ⇏ break DLP

Generalization

Consider $y = g^{x} \mod p$

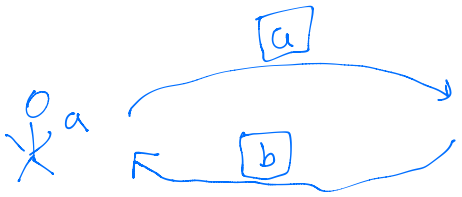
$\cong H(x)$ (kind of like a hash function)

$g^{x} \bmod p \triangleq \boxed{x}$

Rules (malleability / homomorphism)

① $\boxed{x} \cdot \boxed{y} = \boxed{x+y}$ $= g^{x} g^{y} = g^{x+y}$

② $\boxed{x}^{y} = \boxed{x \cdot y}$ $= (g^{x})^{y} = g^{xy}$



$\boxed{b}^{a} = \boxed{ba} = \boxed{ab}$

should select.

$\boxed{a}^{b} = \boxed{ab}$

should select

Real World Consideration

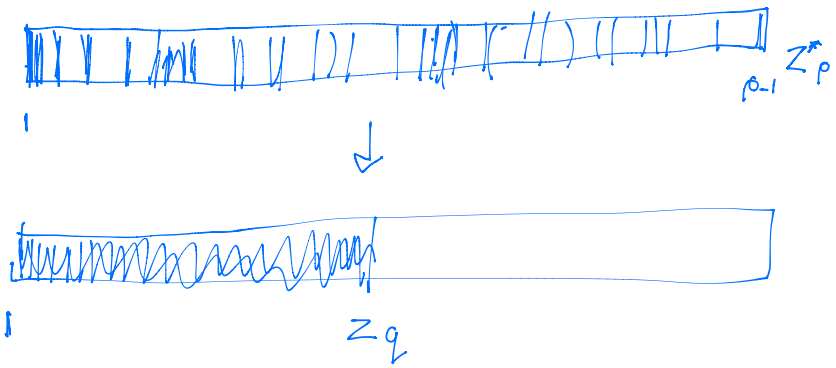
* Alice and Bob agree on $g^{ab} \bmod p$
  - $g \in G_q$
  - $g^a \in G_q$
  - $g^b \in G_q$.
  - $g^{ab} \in G_q$.

* We want to use g^{ab} as an AES key:
  * Is it big enough? Yes → 2048 bits.
  * Is it random? It is a random element of G_q if a and b are random.
  * However the elements of G_q are not every possible 2048-bit value.
  * Use an extractor

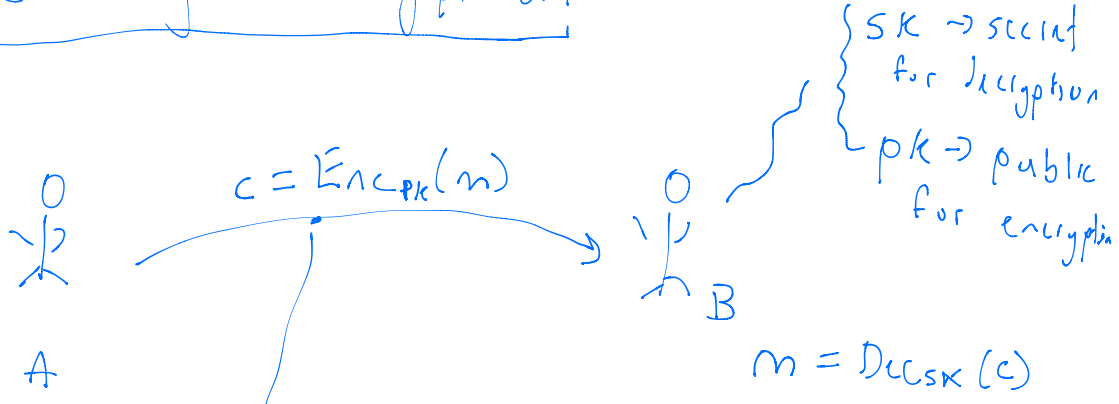
$$K = Ext(g^{ab})$$

AES/MAC keys (128+128 = 256)

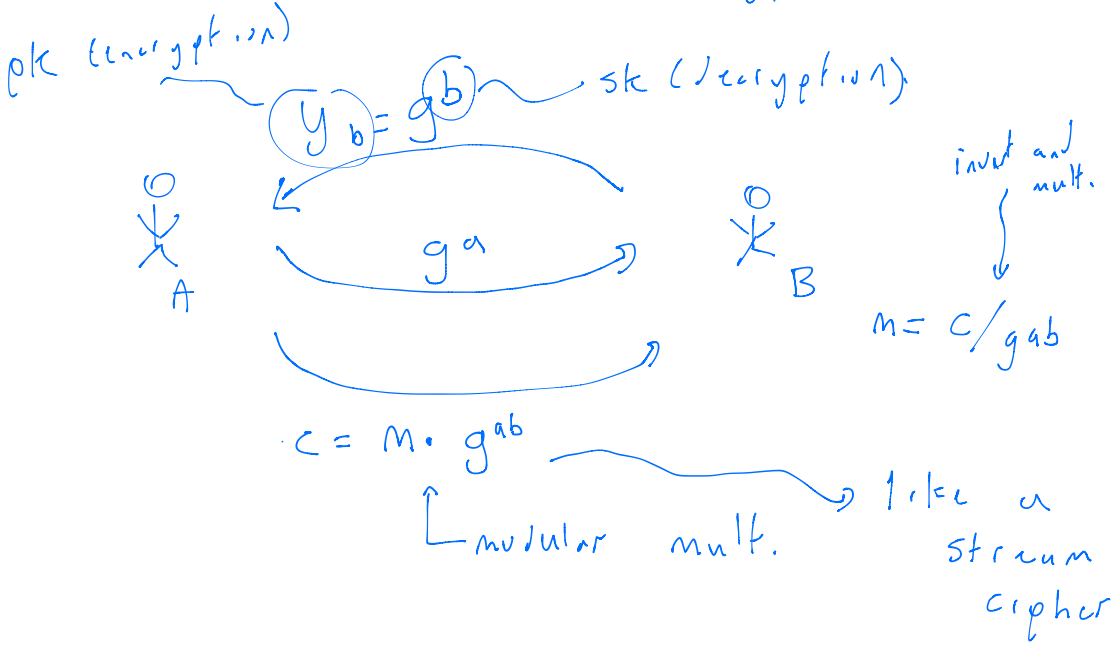
$H_\infty(g^{ab}) \approx |q|$



Public key Encryption



Idea: Turn DH into encryption.



Elgamal Encryption

(1) Key Generation.

- Public parameters $\langle p, q, g \rangle$
- secret key: $x \in \mathbb{Z}_q$
- public key: $y = g^x \bmod p$

(2) Encrypt.

$$Enc_y(m, r) = \langle g^r, m \cdot y^r \rangle$$

$r \in_R \mathbb{Z}_q$ (random factor)

$$= \langle c_1, c_2 \rangle$$

③ Decrypt

$$Dec_x(c) = c_2 \cdot c_1^{-x}$$
$$= (m \cdot y^r)(g^r)^{-x}$$
$$= (m g^{xr})(g^{-xr})$$
$$= m$$

Elgamal Generalization

KeyGen:

x → secret key.

$\boxed{x}$ → public key.

Encryption

$$Enc_{\boxed{x}}(m) = \langle \boxed{r}, m \boxed{x}^r \rangle$$
$$= \langle \boxed{r}, m \boxed{xr} \rangle$$

Decryption.

$$Dec_x(c) = m = m \boxed{xr} / \boxed{r}^x = m \boxed{xr} / \boxed{xr}$$
$$= m$$

Real World Considerations

* ModExp is reasonably fast (milliseconds) but slower than AES
* Elgamal is limited to 2048 bit messages.
* m must be in G_q to encrypt but not all messages are in G_q.
  - ↳ Use a mapping.

$a \longrightarrow a' = a^2 \bmod p$

↳ $[1, q]$ ↑ always in G_q.

$a' \longrightarrow a = a'^{2^{-1}} \bmod p$

↓

2 answers but only one in $[1, q]$

*Note this only works for safe primes.

Security of Elgamal

① Infeasible to compute x given $y = g^x \bmod p$ where x is the secret key and y is the public key

↳ DLP

② Elgamal is CPA-secure under an assumption called DDH

↓

decisional DH

In the setting of integers mod p, we have 3 basic (standard) infeasibility assumptions:

(1) DLP: infeasible to compute x given g^x

(2) CDH: infeasible to compute g^{xy} given $\langle g, g^x, g^y \rangle$

(3) DDH: infeasible to distinguish

$\langle g, g^x, g^y, Z \rangle$ where $Z \in_R G_q$

or $\langle g, g^x, g^y, g^{xy} \rangle$

DLP $>$ CDH $>$ DDH

↑ hardest ↑ easiest but still hard

Elgamal is CPA-secure.

Adversary's view: $\langle g, y, c \rangle$ (public key: $y = g^x$)

$= \langle g, y, g^r, m \cdot y^r \rangle$

$= \langle g, g^x, g^r, m \cdot g^{xr} \rangle$ (3)

indistinguishable under DDH → $\langle g, g^x, g^r, m \cdot z \rangle$ (2) (z ~ random number)

perfectly indistinguishable b/c it is a OTP → $\langle g, g^x, g^r, z \rangle$ (1)

(1) leaks nothing about m because m isn't there.

Argument:
* (1) leaks no info. about m
* (2) is the same as (1) so it leaks no info.
* (3) is indistinguishable from (2) so it leaks no info
  ↳ CPA secure

Elgamal is not CCA Secure

* Elgamal is malleable / homomorphic

① $Enc(m, r) \cdot a$

↳ $\langle c_1, c_2 \cdot a \rangle$

↳ $\langle g^r, (my^r)a \rangle$

↳ $\langle g^r, (ma)y^r \rangle$

↳ $Enc(ma, \ )$

② $Enc(m, r) \cdot Enc(m', r')$

↳ $\langle c_1, c_2 \rangle \cdot \langle c_1', c_2' \rangle$

$= \langle c_1 \cdot c_1', c_2 \cdot c_2' \rangle$

$= \langle g^r g^{r'}, m \cdot y^r \cdot m' \cdot y^{r'} \rangle$

$= \langle g^{r+r'}, (m \cdot m') y^{r+r'} \rangle$

$= Enc(m \cdot m', r + r')$

Pro: compute on encrypted data.

Con: not CCA-secure.

Elgamal is not CCA-secure

① Choose m_0 and m_1 as challenge messages.

② Receive $c_b = Enc(m_b, r_b)$

③ Ask for decryption of

$$c_b' = c_b \cdot Enc(1, \tilde{r})$$

$$= Enc(m_b, r)$$

(rerandomization)

r: new randomness

$r = r_b + \tilde{r}$

r_b: oracle's randomness in c_b

④ Get m_b