

- Today:
- Shamir Secret Sharing
 - Diffie-Hellman Key Exchange

Shamir Secret Sharing

- Suppose Alice has a secret s (eg., a key)
- Suppose she wants to protect s , as follows:
 - Say, she has n friends A_1, \dots, A_n , and she wants to "share" her secret among these friends s.t.
 - * any t or more friends can reconstruct s
 - * any set of $< t$ friends learn nothing about s .

Alice will give A_i a "share" a_i s.t.

- $\forall I \subseteq [n] \quad |I| \geq t$ given $\{a_i\}_{i \in I}$ one can easily find s .
- $\forall I \subseteq [n] \quad |I| < t$, $\{a_i\}_{i \in I}$ are independent of s .

Easy cases:

$$\underline{t=1}: \quad A_i = a$$

$t=n$: Choose at random a_1, \dots, a_n
 s.t. $S = a_1 \oplus \dots \oplus a_n$

(This can be done by choosing a_1, \dots, a_{n-1} at random, and
 setting $a_n = S \oplus (a_1 \oplus \dots \oplus a_{n-1})$)

What about $1 < t < n$?

Shamir's scheme ("How to Share a Secret", 1979)

Suppose $S \in \text{GF}[P]$ for some prime P .

To secret share S :

- Let $a_0 = S$
- Choose at random $a_1, \dots, a_{t-1} \leftarrow \text{GF}(P)$
- Let $f(x) = \sum_{i=0}^{t-1} a_i x^i$
- Let $a_i = (i, \underbrace{f(i)}_{y_i}) \quad \forall i \in [n]$.

Note: Evaluation is easy

Secret reconstruction via interpolation

Given: $(x_i, y_i) \quad 1 \leq i \leq t \quad (w \log)$

$$f(x) = \sum_{i=1}^t f_i(x) \cdot y_i$$

$$\text{where } f_i(x) = \begin{cases} 1 & \text{at } x = x_i \\ 0 & \text{for } x \in \{x_j\}_{j \in [t] - \{i\}} \end{cases}$$

Specifically,

$$f_i(x) = \frac{\prod_{j \neq i} (x - x_j)}{\prod_{j \neq i} (x_i - x_j)}$$

To reconstruct the secret s , simply compute

$$s = f(0) = \sum_{i=1}^t y_i \frac{\prod_{j \neq i} (-x_j)}{\prod_{j \neq i} (x_i - x_j)}$$

Theorem: Shamir's secret sharing scheme

is information theoretically secure, i.e.,

an adv. with $< t$ shares has no info about s .

□ A degree $t-1$ poly

Pf: Fix any s .
 For a randomly chosen deg $t-1$ curve f
 st. $f(0) = s$, for any non-zero x_1, \dots, x_d , $d < t$,
 it holds that $f(x_1), \dots, f(x_d)$ are randomly
 distributed (independent of s).

Diffie-Hellman Key Exchange

Q: How can Alice & Bob establish a shared secret in
 the presence of an eavesdropper?
 (Eve is passive - only listens).

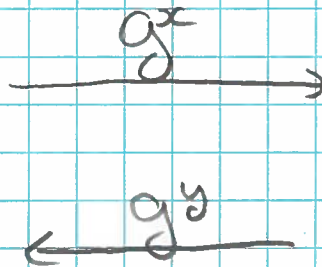
Precursor to public key cryptography.

= Let G be a cyclic group with generator g
 (i.e. $G = \{g^0, g^1, g^2, \dots, g^{|G|-1}\}$)

G & g are fixed & public.

A

Alice chooses a random
 secret $x \leftarrow \{0, 1, \dots, |G|-1\}$,
 and computes and sends
 g^x



B

Bob similarly chooses
 secret $y \leftarrow \{0, 1, \dots, |G|-1\}$
 and computes
 and sends g^y

Alice computes $K = (g^y)^x$

Bob computes $K = (g^x)^y$

Eve learns only g^x & g^y

Computational Diffie Hellman Assumption (CDH)

Given g^x, g^y it is hard to compute g^{xy}

(i.e. there is only a negl chance of succeeding).

CDH \Rightarrow Eve does not learn K , except w. negl prob.

Q: Can Alice & Bob use K as a shared secret to encrypt and/or to MAC later traffic?

Eve may learn a lot of information about K (such as 200 msb's).

Decisional Diffie-Hellman Assumption (DDH)

Given g^x, g^y it is hard to distinguish g^{xy} from g^u , where u is random in $\{0, 1, \dots, |G|-1\}$, w.p. $> \frac{1}{2} + \text{negl}$

Thm: DDH \Rightarrow DH Key exchange is secure
i.e., Eve cannot distinguish between K
and a fresh random key.

PS: Follows immediately from the DDH Assumption.

\Rightarrow Assuming DDH, we can use K to encrypt or MAC
later.

* Don't use the same K for both!

A MAC can leak enough information to break the
encryption, but not enough to allow forgery,
and vice versa.

* Use K to derive 2 fresh keys: One for MAC
& one for encryption (using PRG).