# Day 5: Key Exchange & Public-key Encryption

Today:

$\boxed{0}$ Power is back! (and our website too!)

$\boxed{1}$ Key Exchange

  $\boxed{a}$ Diffie-Hellman Key Exchange

$\boxed{2}$ Public-Key Encryption

$\boxed{3}$ Merkle Puzzles?

# [1] Key Exchange

History:

Ralph Merkle was a college student.

Merkle took a security class w/ a project

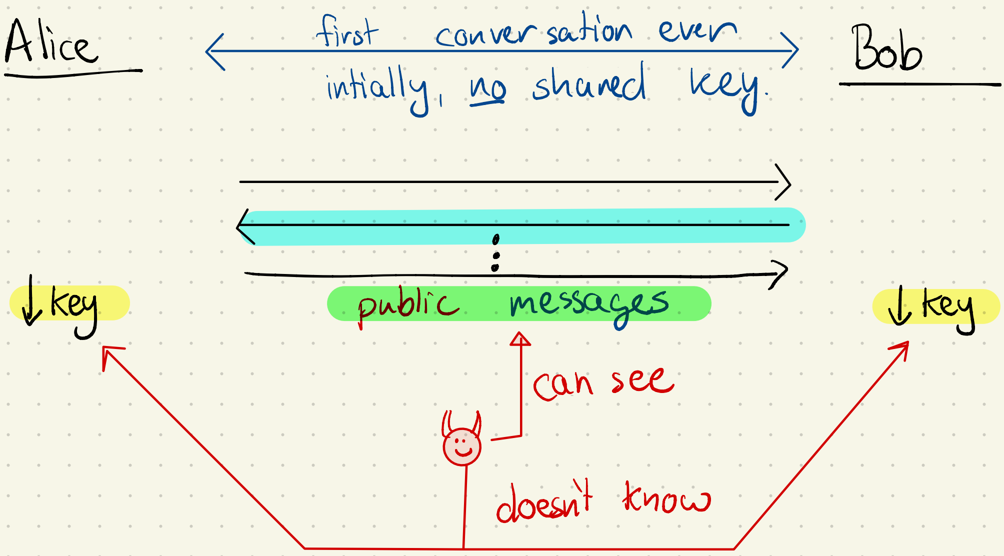
Merkle submitted a project proposal. Intro:

Project Proposal

Topic: Establishing secure communications between seperate secure sites over insecure communication lines.

Assumptions: No prior arrangements have been made between the two sites, and it is assumed that any information known at either site is known to the enemy. The sites, however, are now secure, and any new information will not be divulged.

A revolutionary problem!



Unfortunately, the professor didn't think so...

Project 2 looks more reasonable, maybe because your description Project 1 is muddled terribly. Talk to me about these today.

C.S. 244
FALL 1974

Ralph Merkle

Project Proposal

Topic: Establishing secure communications between seperate

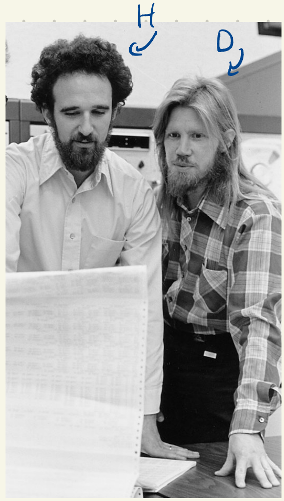
Just kidding, Berkeley is a great place!

- The challenges of being a student at Berkeley...

- Across the bay, Marty Hellman and Whit Diffie were thinking about the very same problem.
- And they had an even better solution...

1976

"New Directions in Cryptography"




I. Introduction

WE STAND TODAY on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals. In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels and supply the equivalent of a written signature. At the same time, theoretical developments in information theory and computer science show promise of providing provably secure cryptosystems, changing this ancient art into a science.

Applications
- digital commerce
- remote access

Problem: key generation without secure channels

Key observation:

- In some cyclic groups:
  - Exponentiation is easy: $G, x \to G^x$
  - Logarithms seem hard: $G, G^x \to x$
- Why is exponentiation easy?
  → Recursive algorithm:
  → $G^0 = 1$
  → $G^x = (G^2)^{x/2}$ (even $x$)
  → $G^x = G \cdot (G^2)^{\frac{x-1}{2}}$ (odd $x$)

  each step cuts $x$ in half.

  → for $x \approx 2^{256} \to$ only $\sim 256$ steps
- Why is computing logarithms hard?
  → We don't know how to do much better than guessing... $H = G^x$

  $G^1 \stackrel{?}{=} H$, $G^2 \stackrel{?}{=} H$, $G^3 \stackrel{?}{=} H$, ...

  → for $x \approx 2^{256} \to \sim 2^{255}$ steps.

→ So, for secret $\underbrace{x}_{\text{secret key}}$, $\underbrace{G^x}_{\text{public key}}$ doesn't reveal $x$.

# 1a Diffie-Hellman Key Exchange

Let $\mathbb{G}$ be a cyclic group of order $q$ (prime)
with generator $G$ (known to all).

| **Alice** | | **Bob** |
|---|---|---|
| $a \leftarrow \text{random}(\mathbb{Z}_q)$ | | $b \leftarrow \text{random}(\mathbb{Z}_q)$ |
| $A \leftarrow G^a$ | | $B \leftarrow G^b$ |
| | $\xrightarrow{\quad A \quad}$ | |
| | $\xleftarrow{\quad B \quad}$ | |
| $\downarrow$ key $= B^a$ | | $\downarrow$ key$' = A^b$ |

Notice: $\text{key} = B^a = (G^b)^a = (G^a)^b = A^b = \text{key}'$

Also notice: if logarithms were easy:

- adversary sees: $A = G^a$ $\quad$ $B = G^b$
- computes: $a$ $\quad$ $b$
- knows: $G^{ab} = \text{key}$

Note: hard logarithms are <u>necessary</u> for secure DH, but <u>insufficient</u>. See "the CDH assumption".

Synthesizing, DH key Exchange has $\underline{2}$ algs:

- KeyGen ():

  $sk \leftarrow \text{random}(\mathbb{Z}_q)$ ⟵ keep "secret key"

  $pk \leftarrow G^{sk}$ ⟵ send "public key"

- DeriveShared (sk, pk'):

  $k \leftarrow (pk')^{sk}$ ⟵ use "shared key"

## $\boxed{2}$ Public-Key Encryption

→ Closely related to key exchange (equivalent)

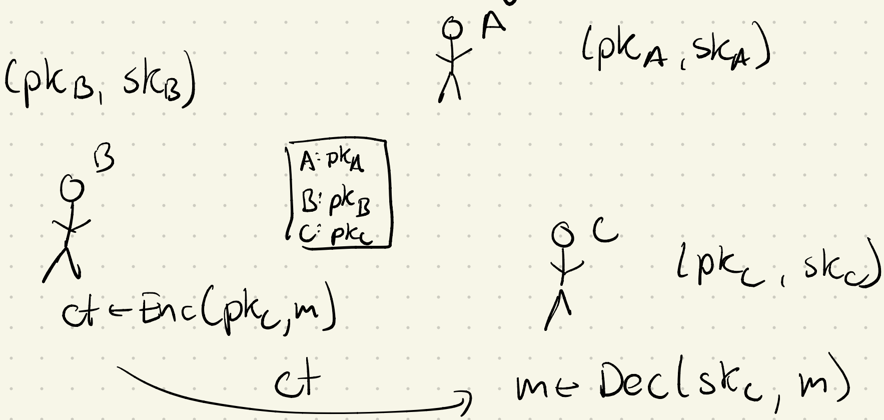
→ Idea:

  → everyone generates (pk, sk) pair

  → publishes pk

  → Enc (pk, msg) → ct

  → Dec (sk, ct) → msg



# DH-based Public Key Encryption

KeyGen():
- $sk \leftarrow \text{random}(\mathbb{Z}_q)$
- $pk \leftarrow G^{sk}$
- output $(sk, pk)$

Enc(pk, m): one-time DH sk for this msg
- $a \leftarrow \text{random}(\mathbb{Z}_q)$
- $k \leftarrow H(pk^a)$ DH output. Must be hashed so we can use stream cipher.
- $c \leftarrow \text{StreamCipher.Enc}(k, m)$
- output $(G^a, c)$

Dec(sk, ct): one-time DH public-key.
- $(A, c) \leftarrow ct$ DH output
- $k \leftarrow H(A^{sk})$
- $m \leftarrow \text{StreamCipher.Dec}(k, c)$
- output m.

# 3 Merkle puzzles:

Fix $N$: a "domain size"

$n$: a "number of keys"

| Alice | | Bob |
|---|---|---|
| chooses $n$ random values from $\{0, 1, \ldots, N-1\}$ (a) | | same. (b) |
| computes a list of the hashes of each value (A) | | same (B) |
| | list of hashes $\longrightarrow$ | |
| | $\longleftarrow$ list of hashes | |
| output the smallest value in a whose hash is in Bob's list (or fail if there is none) | | same |

$\downarrow$

Cool facts:

- When $n = \sqrt{N}$, the probability of failure is $\approx \frac{1}{e} \approx 37\%$.
- By setting $n = k\sqrt{N}$, the probability of failure is exponentially small in $k$.

→ Generalization of the "birthday paradox": there are 365 possible birthdays, but a room of 23 people has a double-birthday w/ >50% probability!

# 4 Problem Overview

Required:

1. DH key exchange:
   (key generation, derive shared key)
2. DH-based public-key encryption:
   (encryption, decryption)

Bonus:

3. Merkle puzzles
   (key gen, derive shared key)
   (success probability estimation)
   either order
4. Attacking Diffie-Hellman