Online Cryptography Course

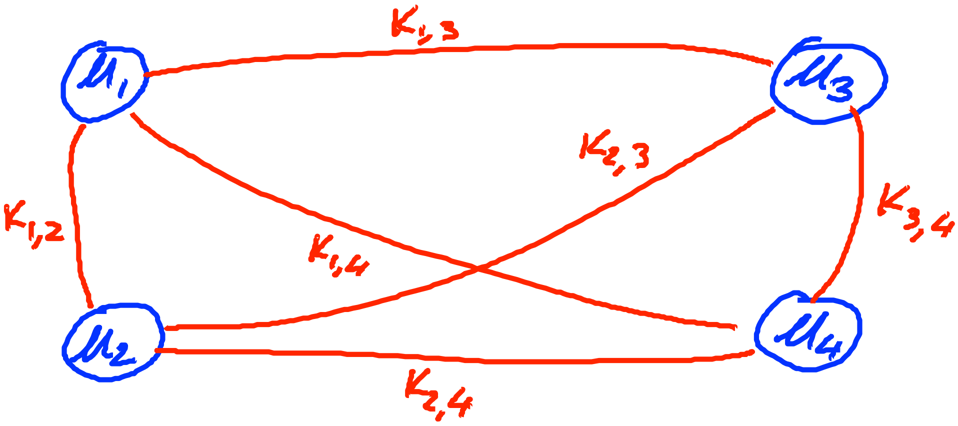
Dan Boneh

Basic key exchange

Trusted 3rd parties

Key management

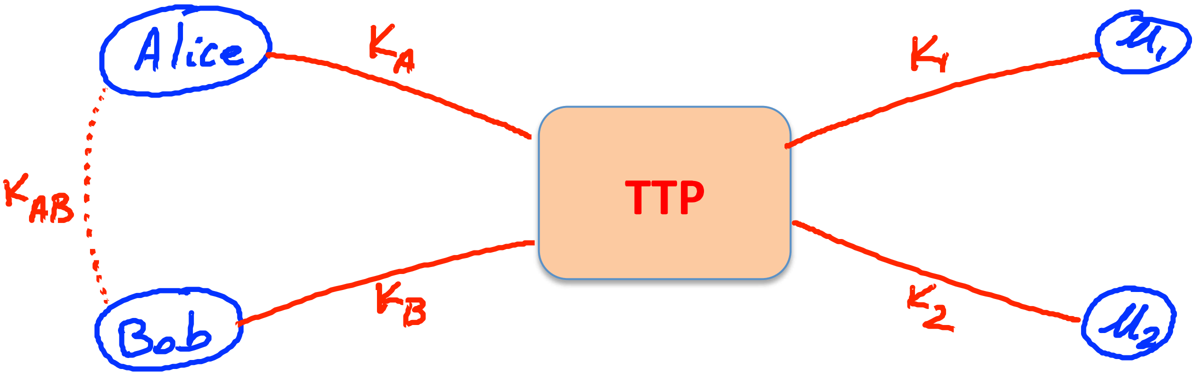
Problem: n users. Storing mutual secret keys is difficult



Total: O(n) keys per user

A better solution

Online Trusted 3rd Party (TTP)

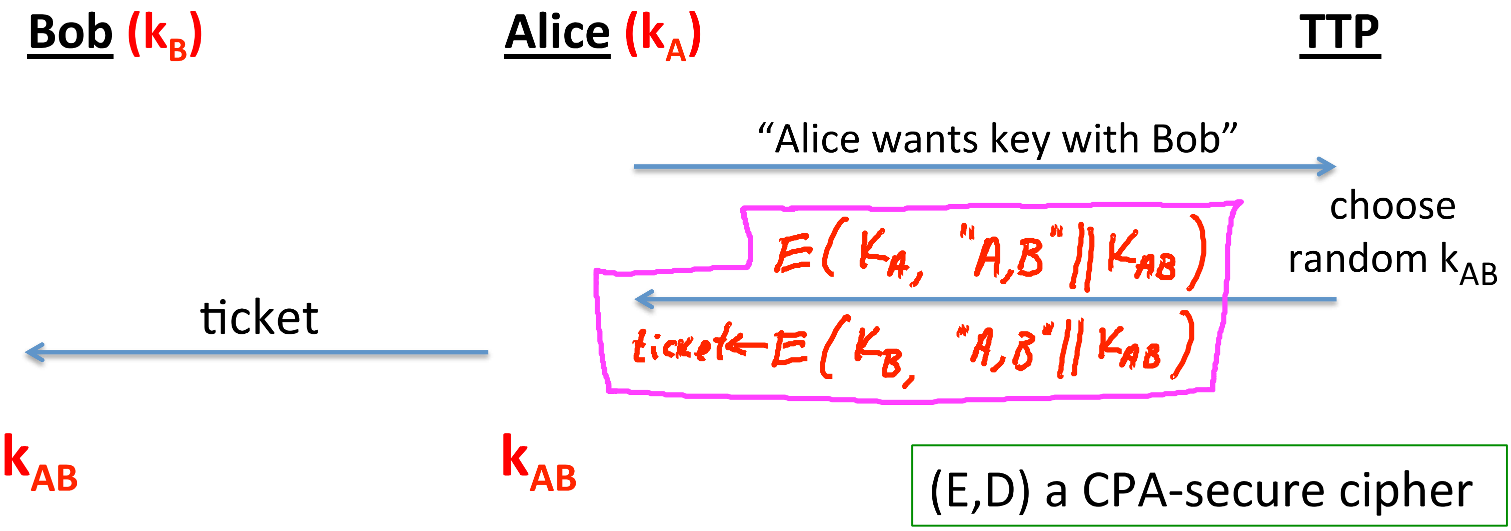


Every user only remembers one key.

Dan Boneh

Generating keys: a toy protocol

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



Bob (k_B)

Alice (k_A)

TTP

"Alice wants key with Bob"

choose random k_{AB}

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

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

ticket

k_{AB}

k_{AB}

(E,D) a CPA-secure cipher

Generating keys: a toy protocol

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

Eavesdropper sees: $E(k_A,$ "A, B" ll $k_{AB})$; $E(k_B,$ "A, B" ll $k_{AB})$

(E,D) is CPA-secure ⇒
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)

End of Segment

Online Cryptography Course Dan Boneh

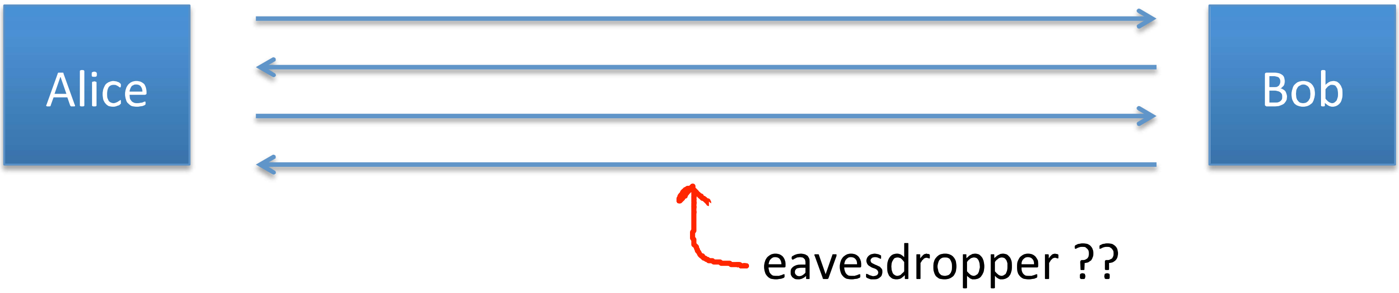
Basic key exchange

Merkle Puzzles

Key exchange without an online TTP?

Goal: Alice and Bob want shared key, unknown to eavesdropper

- For now: security against eavesdropping only (no tampering)



eavesdropper ??

Can this be done using generic symmetric crypto?

Merkle Puzzles (1974)

Answer: yes, but very inefficient

Main tool: puzzles

- Problems that can be solved with some effort
- Example: $E(k,m)$ a symmetric cipher with $k \in \{0,1\}^{128}$
  - **puzzle(P) = E(P, "message")** where $P = 0^{96} \| b_1 \ldots b_{32}$
  - Goal: find P by trying all 2^{32} possibilities

Merkle puzzles

Alice: prepare 2^{32} puzzles

- For $i=1, \ldots, 2^{32}$ choose random $\mathbf{P_i} \in \{0,1\}^{32}$ and $\mathbf{x_i}, \mathbf{k_i} \in \{0,1\}^{128}$

  set puzzle$_i$ ⟵ E($0^{96} \| \mathbf{P_i}$, **"Puzzle # x_i" ll k_i**)
- Send puzzle$_1$, … , puzzle$_{2^{32}}$ to Bob

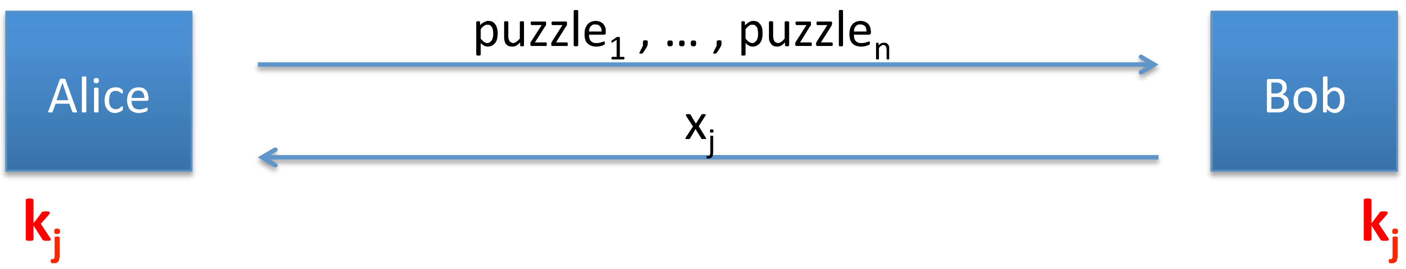
Bob: choose a random puzzle$_j$ and solve it. Obtain (x_j, k_j) .

- Send x_j to Alice

Alice: lookup puzzle with number x_j . Use k_j as shared secret

Dan Boneh

In a figure



Alice

puzzle_1 , … , puzzle_n

x_j

Bob

k_j

k_j

Alice's work: O(n) (prepare n puzzles)

Bob's work: O(n) (solve one puzzle)

Eavesdropper's work: $O(n^2)$ (e.g. 2^{64} time)

Impossibility Result

Can we achieve a better gap using a general symmetric cipher?

Answer: unknown

But: roughly speaking,

quadratic gap is best possible if we treat cipher as a black box oracle [IR'89, BM'09]

End of Segment

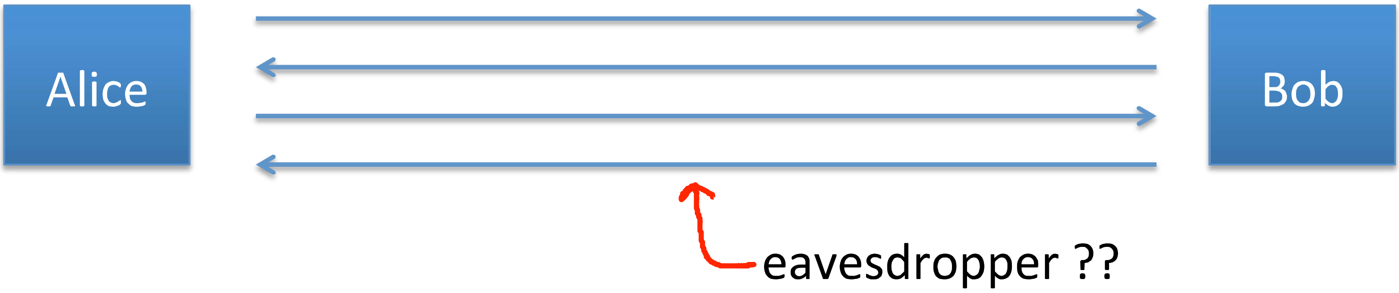
Basic key exchange

The Diffie-Hellman protocol

Key exchange without an online TTP?

Goal: Alice and Bob want shared secret, unknown to eavesdropper

- For now: security against eavesdropping only (no tampering)



Can this be done with an exponential gap?

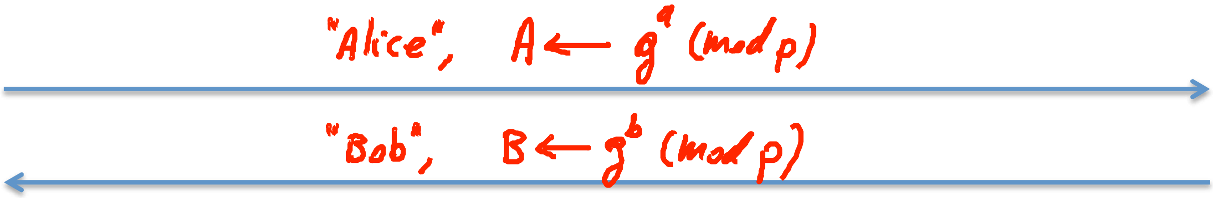
The Diffie-Hellman protocol (informally)

Fix a large prime p (e.g. 600 digits)

Fix an integer g in {1, …, p}

Alice **Bob**

choose random **a** in {1,…,p-1} choose random **b** in {1,…,p-1}



"Alice", $A \leftarrow g^a \pmod p$

"Bob", $B \leftarrow g^b \pmod p$

$$\mathbf{B^a} \pmod p \;=\; (g^b)^a \;=\; \mathbf{k_{AB} = g^{ab}} \pmod p \;=\; (g^a)^b \;=\; \mathbf{A^b} \pmod p$$

Dan Boneh

Security (much more on this later)

Eavesdropper sees: p, g, $A=g^a \pmod p$, and $B=g^b \pmod p$

Can she compute $g^{ab} \pmod p$??

More generally: define $DH_g(g^a, g^b) = g^{ab} \pmod p$

How hard is the DH function mod p?

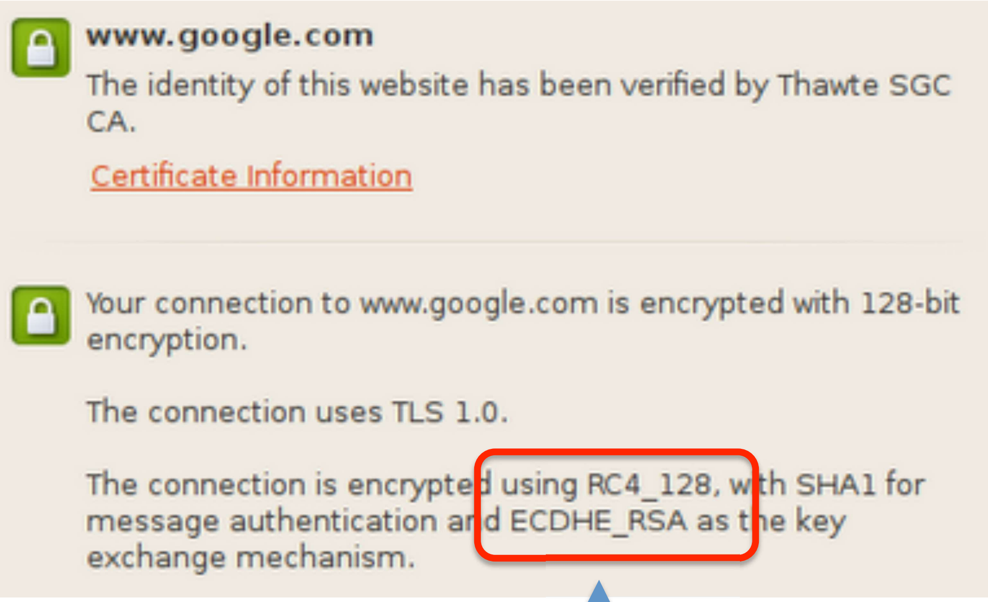
How hard is the DH function mod p?

Suppose prime p is n bits long.

Best known algorithm (GNFS): run time exp($\tilde{O}(\sqrt[3]{n})$)

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

As a result: slow transition away from (mod p) to elliptic curves



www.google.com

The identity of this website has been verified by Thawte SGC CA.

Certificate Information

Your connection to www.google.com is encrypted with 128-bit encryption.

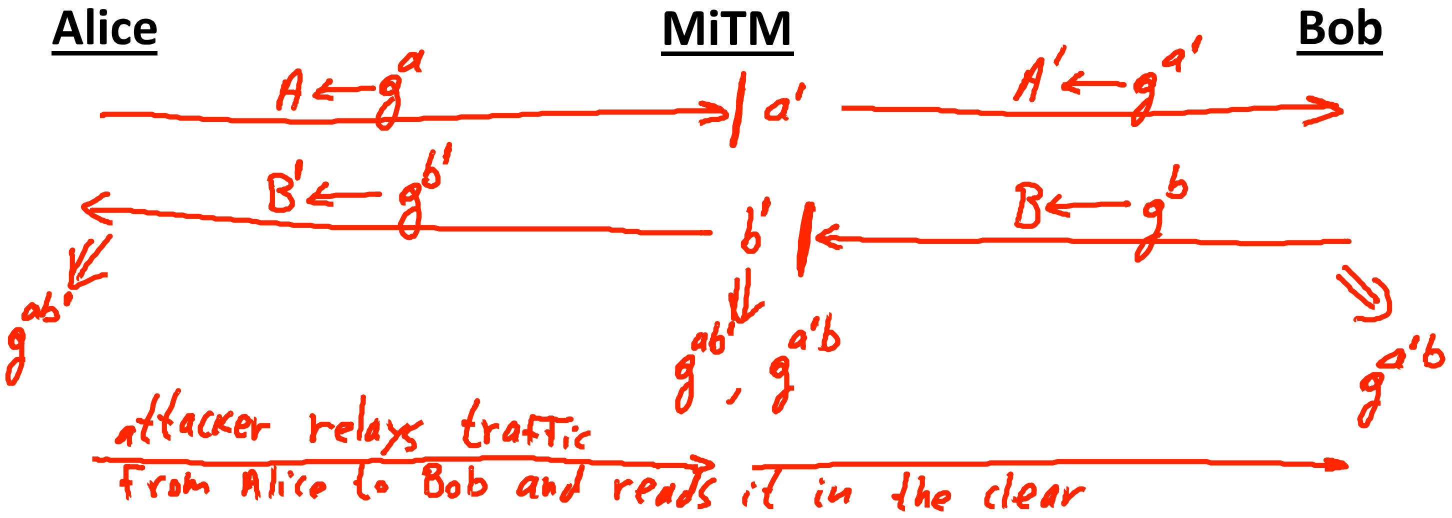
The connection uses TLS 1.0.

The connection is encrypted using RC4_128, with SHA1 for message authentication and ECDHE_RSA as the key exchange mechanism.

Elliptic curve Diffie-Hellman

Insecure against man-in-the-middle

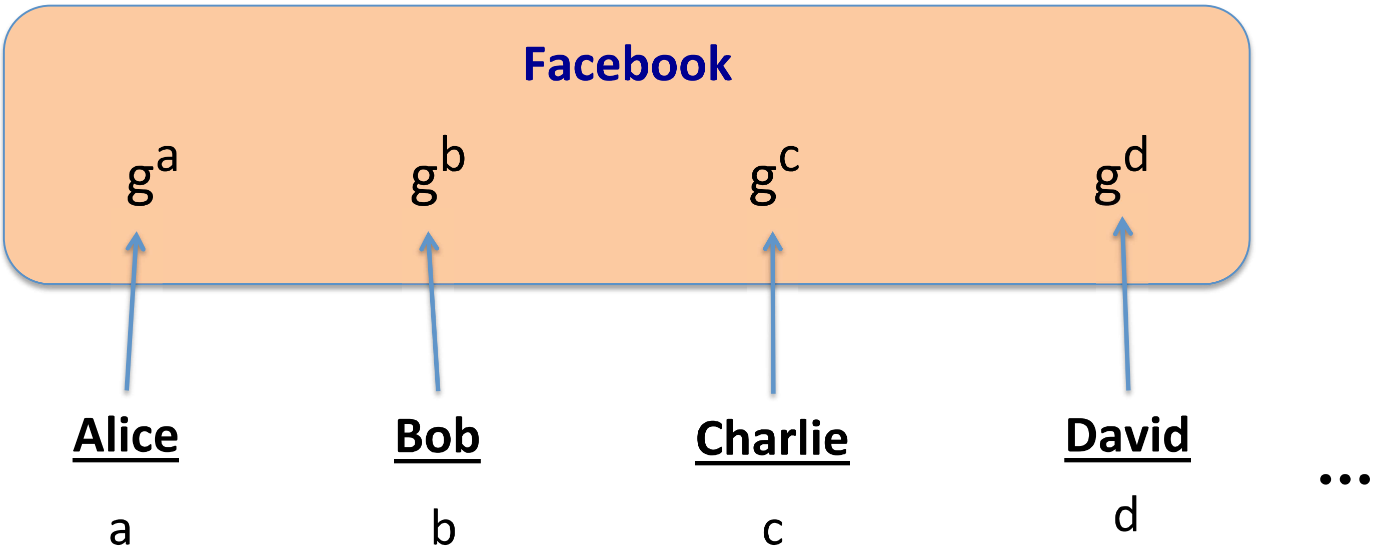
As described, the protocol is insecure against **active** attacks



Alice **MiTM** **Bob**

attacker relays traffic from Alice to Bob and reads it in the clear

Another look at DH



Facebook

g^a g^b g^c g^d

Alice **Bob** **Charlie** **David** …

a b c d

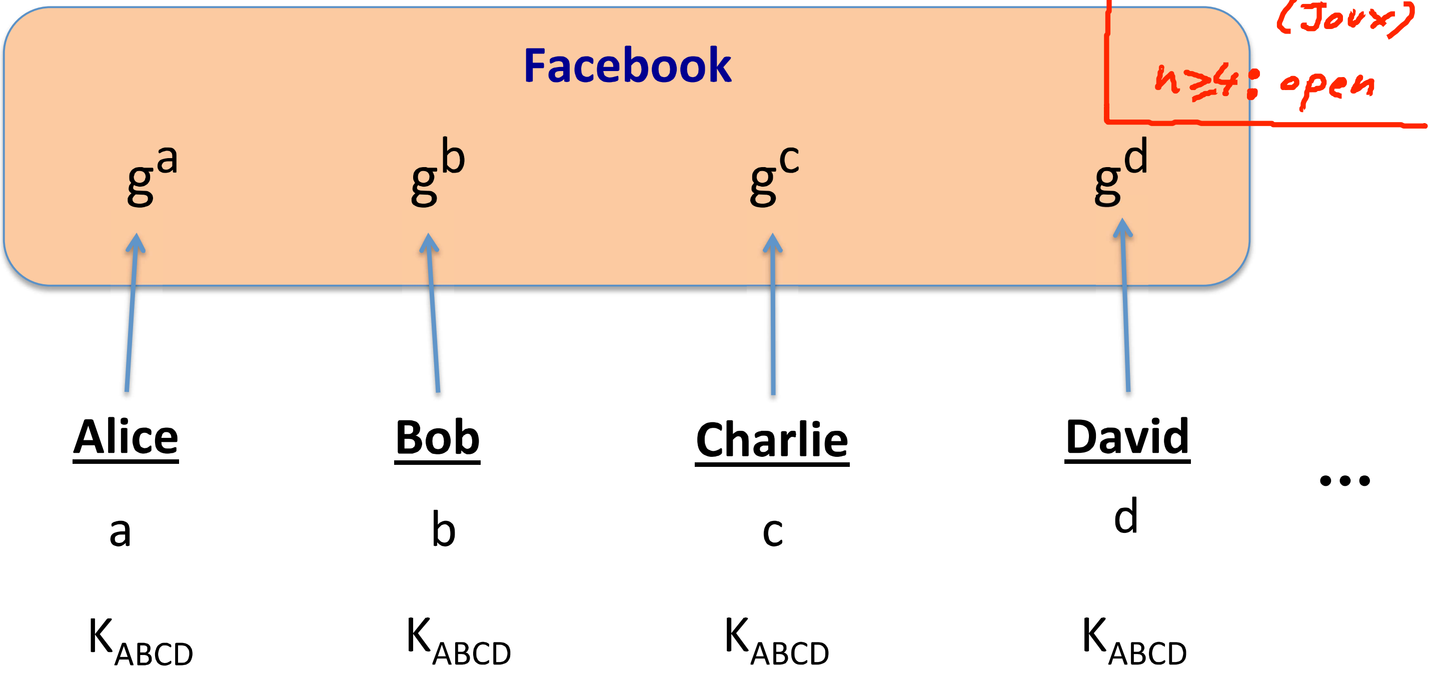
$K_{AC}=g^{ac}$ $K_{AC}=g^{ac}$

An open problem

n=2 : DH

n=3 : known (Joux)

n≥4 : open



Facebook

g^a g^b g^c g^d

Alice **Bob** **Charlie** **David** …

a b c d

K_{ABCD} K_{ABCD} K_{ABCD} K_{ABCD}

End of Segment

Online Cryptography Course Dan Boneh

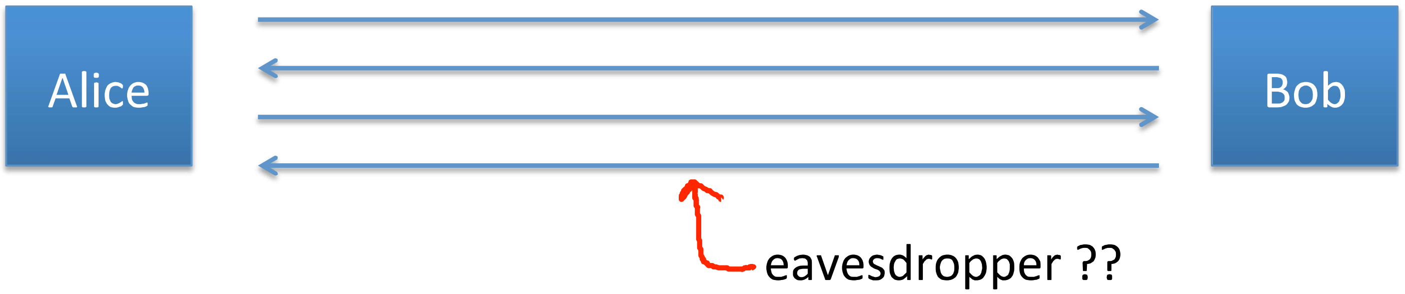
Basic key exchange

Public-key encryption

Establishing a shared secret

Goal: Alice and Bob want shared secret, unknown to eavesdropper

- For now: security against eavesdropping only (no tampering)



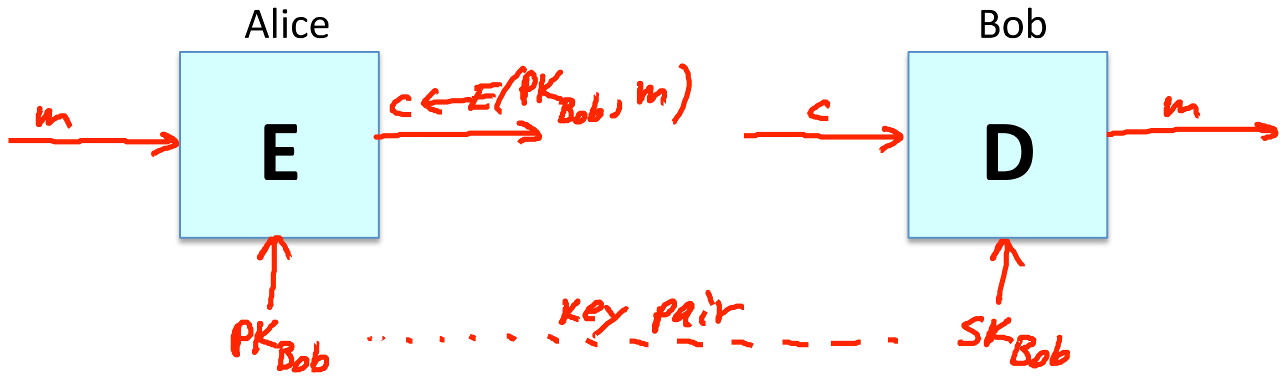
Alice

Bob

eavesdropper ??

This segment: a different approach

Public key encryption



Alice

Bob

E

D

PK: public key, SK: secret key

Public key encryption

Def: a public-key encryption system is a triple of algs. (G, E, D)

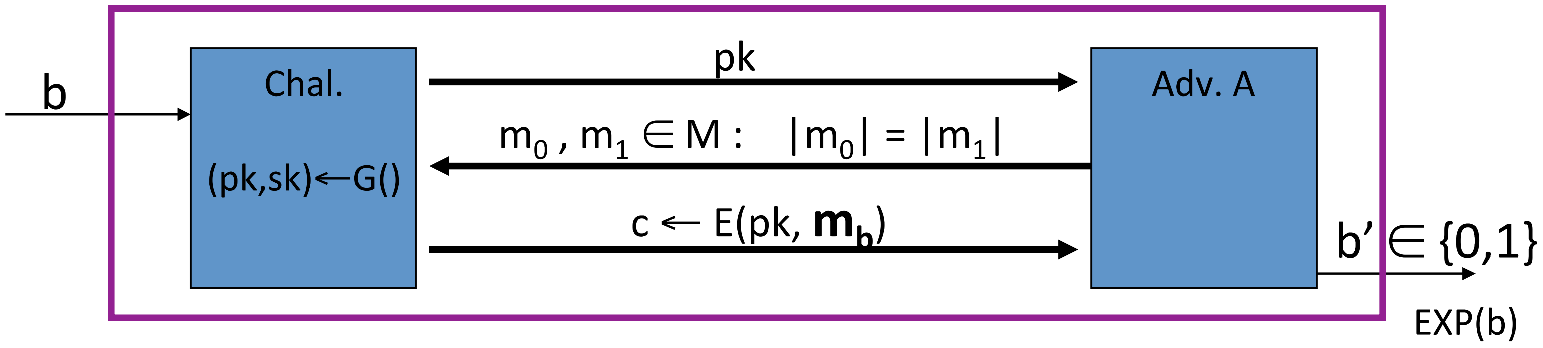
- G(): randomized alg. outputs a key pair (pk, sk)
- E(pk, m): randomized alg. that takes $m \in M$ and outputs $c \in C$
- D(sk,c): det. alg. that takes $c \in C$ and outputs $m \in M$ or $\perp$

Consistency: $\forall$(pk, sk) output by G :

$$\forall m \in M: \quad D(sk,\ E(pk, m)\ ) = m$$

Semantic Security

For b=0,1 define experiments EXP(0) and EXP(1) as:



b

Chal.

(pk,sk)$\leftarrow$G()

pk

m_0 , $m_1 \in$ M : $|m_0| = |m_1|$

c $\leftarrow$ E(pk, $\mathbf{m_b}$)

Adv. A

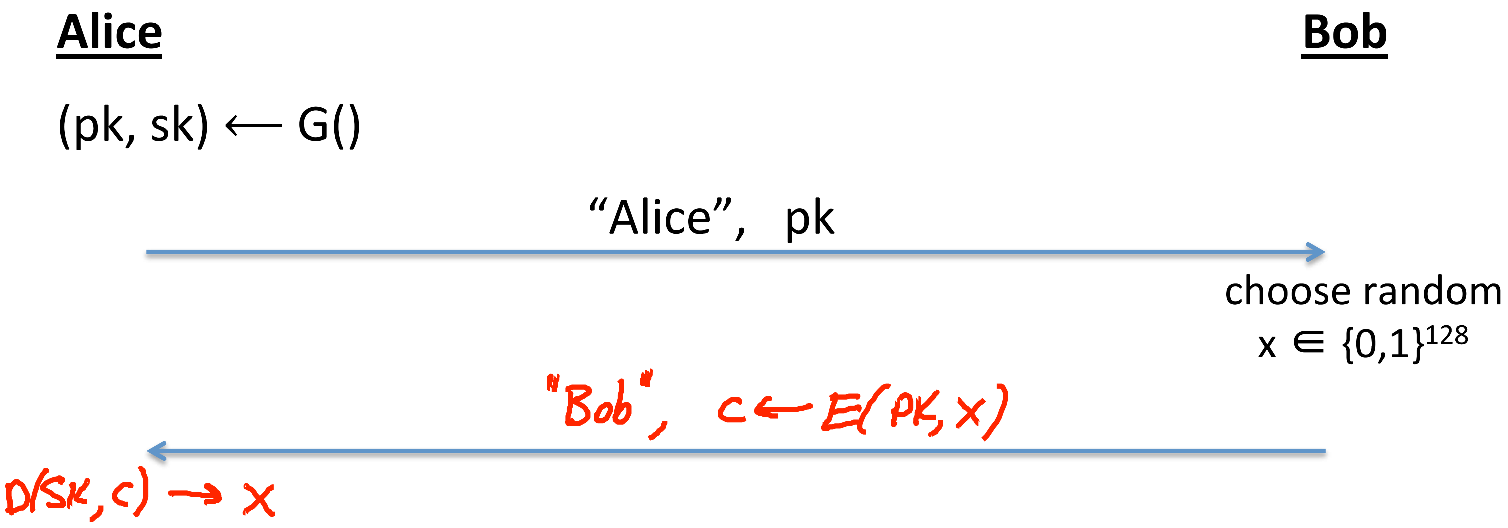
b' $\in$ {0,1}

EXP(b)

Def: E =(G,E,D) is sem. secure (a.k.a IND-CPA) if for all efficient A:

$$\text{Adv}_{\text{SS}}[A,E] = \left| \Pr[\text{EXP}(0)=1] - \Pr[\text{EXP}(1)=1] \right| < \text{negligible}$$

Establishing a shared secret



Alice

Bob

$(pk, sk) \longleftarrow G()$

"Alice", pk

choose random $x \in \{0,1\}^{128}$

"Bob", $c \leftarrow E(pk, x)$

$D(sk, c) \rightarrow x$

x: shared secret

Security (eavesdropping)

Adversary sees **pk, E(pk, x)** and wants **x** $\in$ M

Semantic security ⇒

adversary cannot distinguish

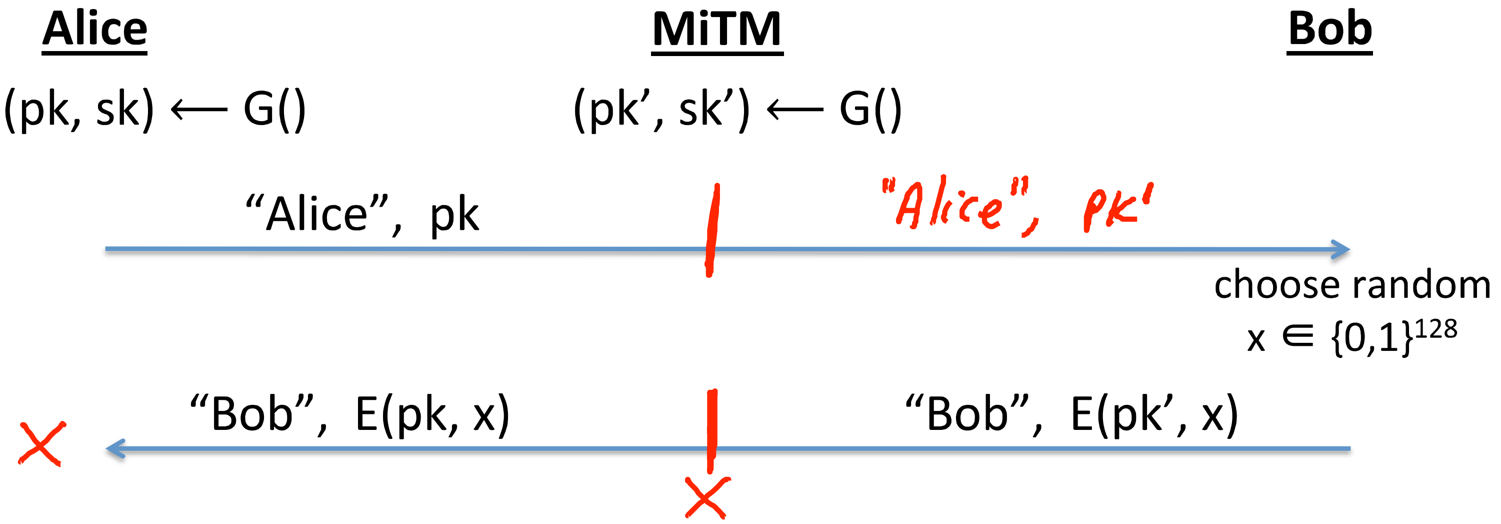
$\{$ pk, E(pk, x), x $\}$ from $\{$ pk, E(pk, x), rand $\in$ M $\}$

⇒ can derive session key from x.

Note: protocol is vulnerable to man-in-the-middle

Insecure against man in the middle

As described, the protocol is insecure against **active** attacks



Alice

MiTM

Bob

(pk, sk) ⟵ G()

(pk', sk') ⟵ G()

"Alice", pk

"Alice", pk'

choose random $x \in \{0,1\}^{128}$

"Bob", E(pk, x)

"Bob", E(pk', x)

Public key encryption: constructions

Constructions generally rely on hard problems from number theory and algebra

Next module:

- Brief detour to catch up on the relevant background

Further readings

- Merkle Puzzles are Optimal,
  B. Barak, M. Mahmoody-Ghidary, Crypto '09

- On formal models of key exchange (sections 7-9)
  V. Shoup, 1999

End of Segment