CSE 484/M584: Computer Security (and Privacy)

Spring 2025

David Kohlbrenner
dkohlbre@cs

UW Instruction Team: David Kohlbrenner, Yoshi Kohno, Franziska Roesner, Nirvan Tyagi. Thanks to Dan Boneh, Dieter Gollmann, Dan Halperin, John Manferdelli, John Mitchell, Vitaly Shmatikov, Bennet Yee, and many others for sample slides and materials

Admin

- HW1 due Wednesday
- Lab 2 (Cryptolab) next Wednesday
  - Note there was an update to the short answer CTR question (there is an oracle you can interact with now!)
  - Start now if you haven't!
- Lab 1a/b Exploits
  - Check partner handin status ASAP!
  - We will file CSSC cases shortly

How Can a Cipher Be Attacked?

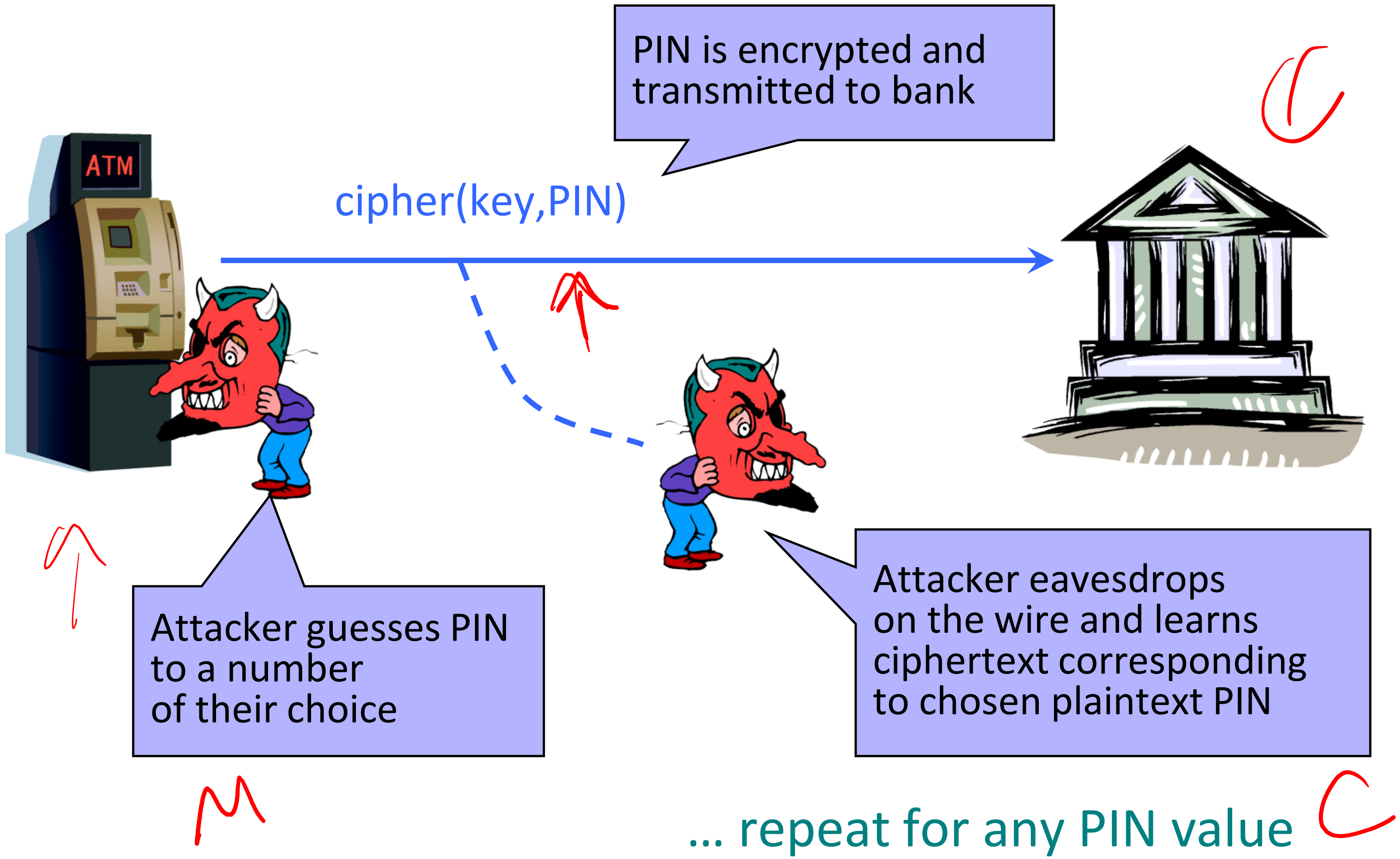
- Attackers knows ciphertext and encryption algorithm
  - What else does the attacker know? Depends on the application in which the cipher is used!
- Ciphertext-only attack
- KPA: Known-plaintext attack (stronger)
  - Knows some plaintext-ciphertext pairs
- CPA: Chosen-plaintext attack (even stronger)
  - Can obtain ciphertext for any plaintext of his choice

M_0, C_0
M_1, C_1

M_0, M_1, M_2
$C_0\ C_1\ C_2$

Chosen Plaintext Attack

CPA



… repeat for any PIN value C

How Can a Cipher Be Attacked?

- Attackers knows ciphertext and encryption algorithm
  - What else does the attacker know? Depends on the application in which the cipher is used!
- Ciphertext-only attack
- KPA: Known-plaintext attack (stronger)
  - Knows some plaintext-ciphertext pairs
- CPA: Chosen-plaintext attack (even stronger)
  - Can obtain ciphertext for any plaintext of his choice
- CCA: Chosen-ciphertext attack (very strong)
  - Can decrypt any ciphertext <u>except</u> the target

C_x

$C_0 \; C_1$

$C_0 \; C_1 \; C_2$

$M_0 \; M_1 \; M_2$

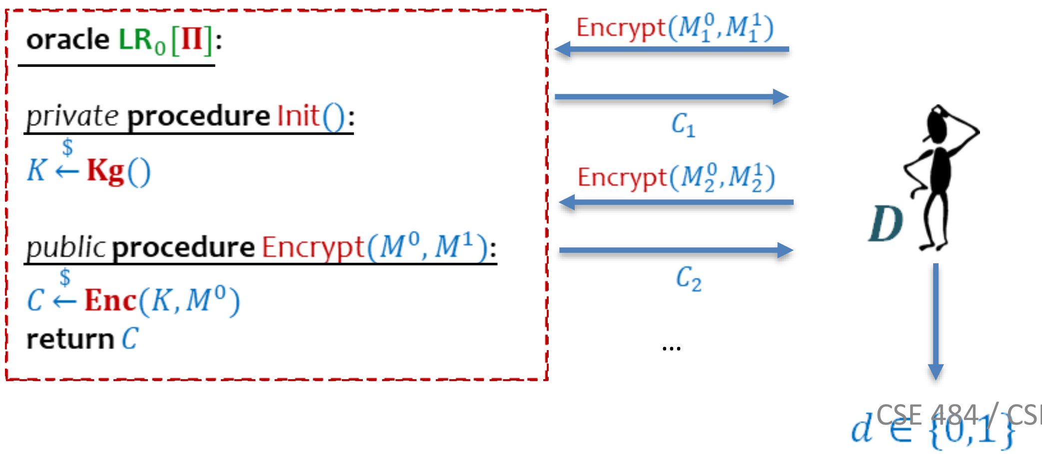
Very Informal Intuition

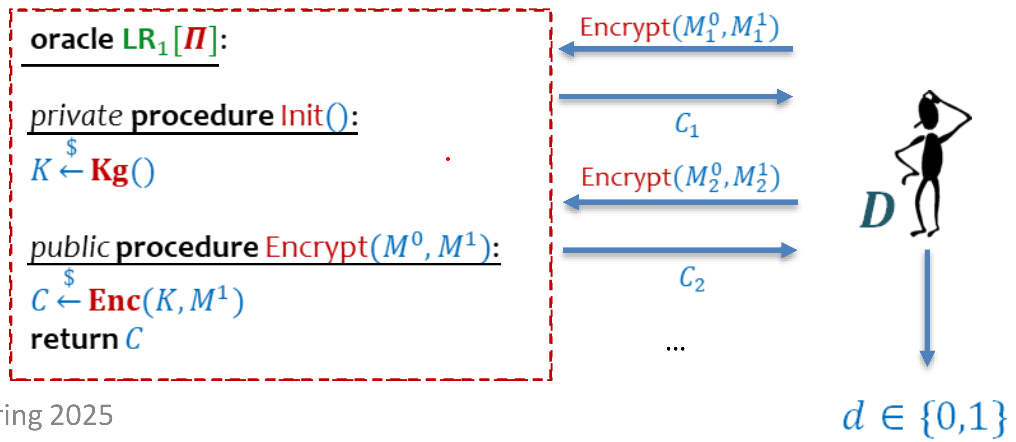
Minimum security requirement for a modern encryption scheme

- Security against chosen-plaintext attack (CPA)
  - Ciphertext leaks no information about the plaintext
  - Even if the attacker correctly guesses the plaintext, they cannot verify their guess
  - Every ciphertext is unique, encrypting same message twice produces completely different ciphertexts
    - Implication: encryption must be randomized or stateful

The Shape of the Formal Approach

- INDistinguishability under Chosen Plaintext Attack ("IND-CPA")
- Formalized cryptographic game
  - Adversary submits pairs of plaintexts (M_0, M_1)
  - Gets back ONE of the ciphertexts (C_b)
  - Adversary must **guess** which ciphertext this is (C_0 or C_1)
  - If they can do better than 50/50, they win





Very Informal Intuition

Minimum security requirement for a modern encryption scheme

- Security against chosen-plaintext attack (CPA)
  - Ciphertext leaks no information about the plaintext
  - Even if the attacker correctly guesses the plaintext, they cannot verify their guess
  - Every ciphertext is unique, encrypting same message twice produces completely different ciphertexts
    - Implication: encryption must be randomized or stateful
- Security against chosen-ciphertext attack (CCA)
  - Integrity protection – it is not possible to change the plaintext by modifying the ciphertext

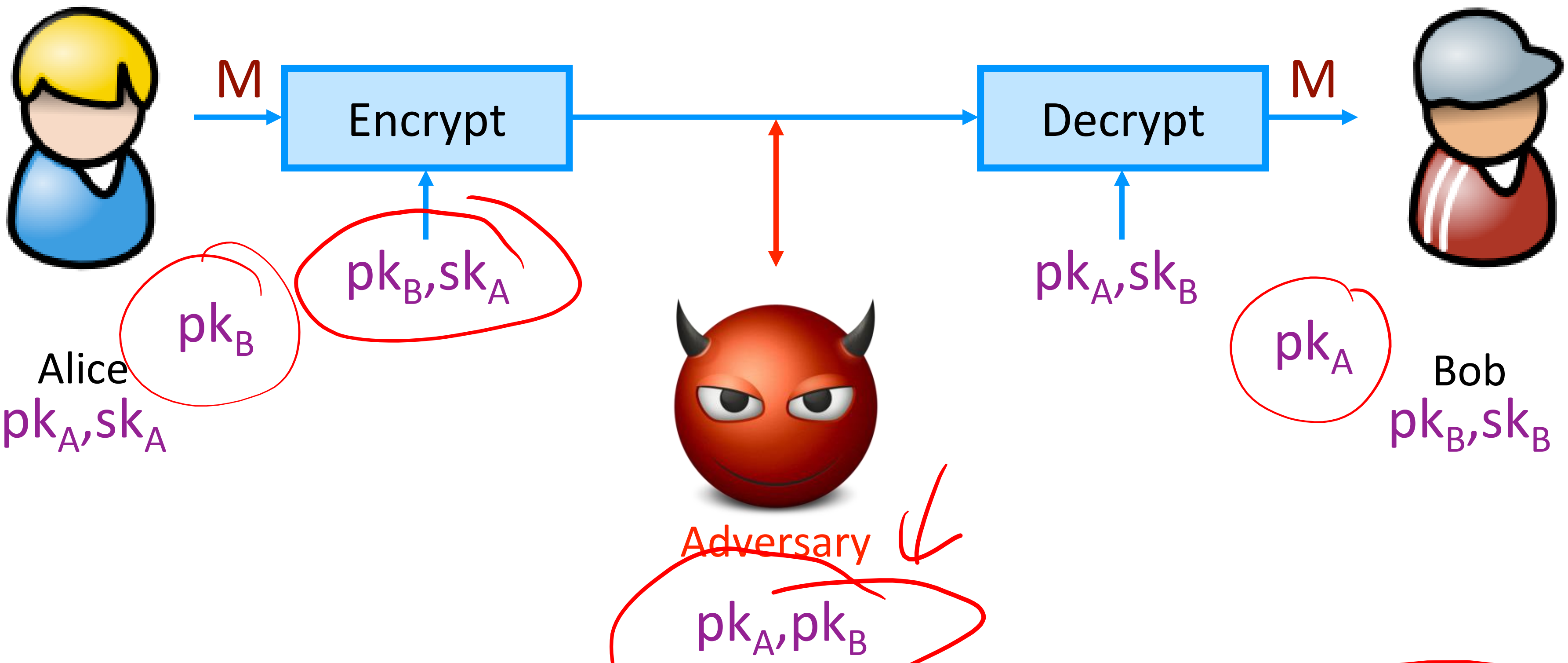
Flavors of Cryptography

- Symmetric cryptography
  - Both communicating parties have access to a shared random string K, called the key.

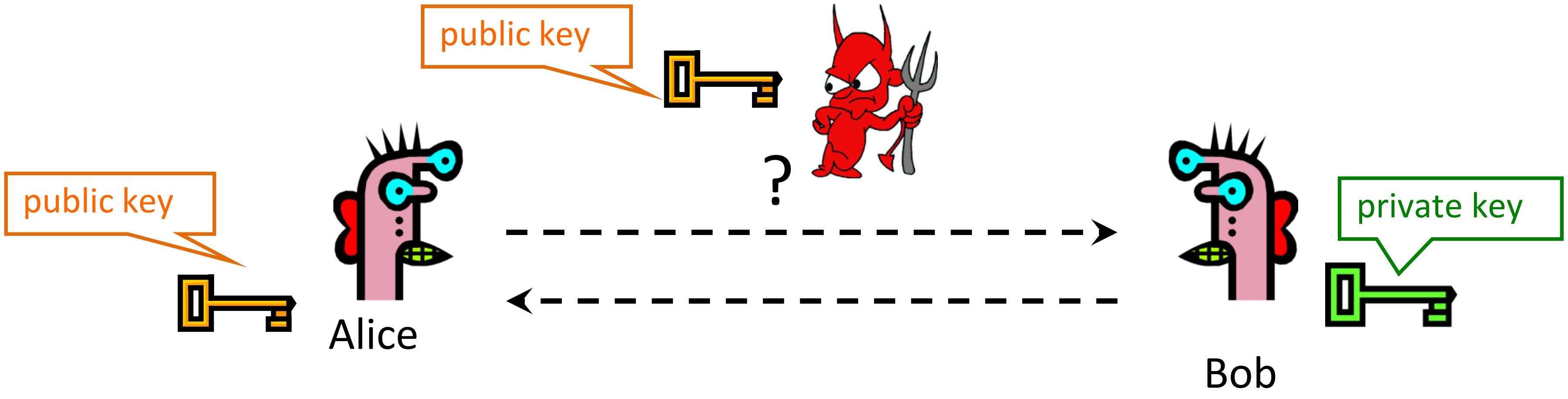
- Asymmetric cryptography
  - Each party creates a public key pk and a secret key sk.

Asymmetric Setting for Encryption

Each party creates a public key pk and a secret key sk



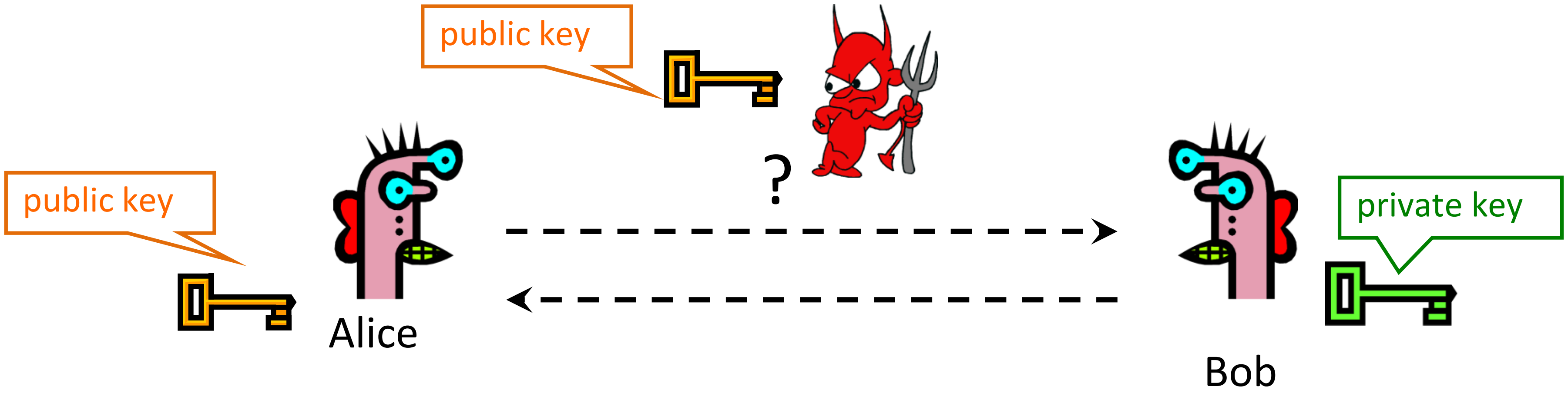
Public Key Crypto: Basic Problem



Given: Everybody knows Bob's public key
Only Bob knows the corresponding private key

Goals: 1. Alice wants to send a secret message to Bob
2. Bob wants to authenticate a message

Public Key Crypto: Basic Problem



Ignore for now: How do we know it's REALLY Bob's??

Given: Everybody knows Bob's public key
Only Bob knows the corresponding private key

Goals: 1. Alice wants to send a secret message to Bob
2. Bob wants to authenticate a message

Applications of Public Key Crypto

- Encryption for confidentiality
  - Anyone can encrypt a message
    - With symmetric crypto, must know secret key to encrypt
  - Only someone who knows private key can decrypt
  - Key management is simpler (or at least different)
    - Secret is stored only at one site: good for open environments
- Digital signatures for integrity
  - Can "sign" a message with your private key

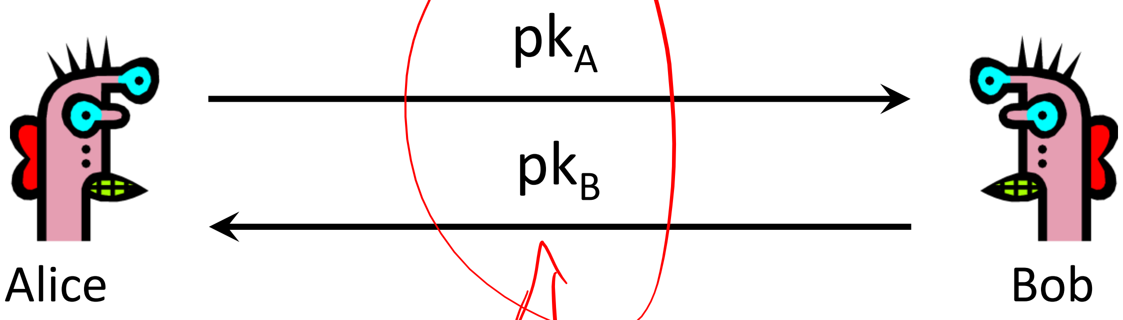
Applications of Public Key Crypto

- Encryption for confidentiality
  - Anyone can encrypt a message
    - With symmetric crypto, must know secret key to encrypt
  - Only someone who knows private key can decrypt
  - Key management is simpler (or at least different)
    - Secret is stored only at one site: good for open environments
- Digital signatures for integrity
  - Can "sign" a message with your private key
- Session key establishment / "Key exchange"
  - Exchange messages to create a secret session key
  - Then switch to symmetric cryptography (why?)

K

Key Exchange

sk_A
pk_A



sk_B
pk_B

Compute shared secret $k = \text{KEx}(sk_A, pk_B)$

Compute shared secret $k = \text{KEx}(sk_B, pk_A)$

Groups

- Group: A set G of elements and an operation $\oplus$ such that:
  - Associative: $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
  - Identity: $a \oplus I = a$
  - Inverse: $a \oplus a^{-1} = I$

Notation: $a^2 = a \oplus a$, $a^3 = a \oplus a \oplus a$, …

Groups

- Group: A set G of elements and an operation $\oplus$ such that:
  - Associative: $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
  - Identity: $a \oplus I = a$
  - Inverse: $a \oplus a^{-1} = I$
  - Order: Number of elements in group
  - Optional useful property: Cyclic: $\{g, g^2, g^3, \ldots, g^{order}\} = G$ for "generator" g

Notation: $a^2 = a \oplus a$, $a^3 = a \oplus a \oplus a$, …

$g, g \oplus g, g \oplus g \oplus g$

Groups

Notation: $a^2 = a \oplus a$, $a^3 = a \oplus a \oplus a$, …

- Group: A set G of elements and an operation $\oplus$ such that:
  - Associative: $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
  - Identity: $a \oplus I = a$
  - Inverse: $a \oplus a^{-1} = I$
  - Order: Number of elements in group
  - Optional useful property: Cyclic: $\{g, g^2, g^3, \ldots, g^{order}\} = G$ for "generator" g

n-1

- Example Group 1: Additive Group of Integers Modulo n (Z_n or Z/nZ)
  - Special case: $n = p$ where p is a prime (Z_p)
  - $G = \{0, 1, \ldots, p-1\}$
  - $\oplus = +$ mod p

Z_n

p mod p

Groups

- Group: A set G of elements and an operation $\oplus$ such that:
  - Associative: $(a \oplus b) \oplus c = a \oplus (b \oplus c)$
  - Identity: $a \oplus I = a$
  - Inverse: $a \oplus a^{-1} = I$
  - Order: Number of elements in group
  - Optional useful property: Cyclic: $\{g, g^2, g^3, \ldots, g^{order}\} = G$ for "generator" g
- Example Group 1: Additive Group of Integers Modulo n (Z_n or Z/nZ)
  - Special case: $n = p$ where p is a prime (Z_p)
  - $G = \{0, 1, \ldots, p\text{-}1\}$
  - $\oplus = +$ mod p
- Example Group 2: Multiplicative Group of Integers Modulo n (Z_n*or (Z/nZ)*)
  - Special case: $n = p$ where p is a prime
  - $G = \{1, 2, \ldots, p\text{-}1\}$
  - $\oplus = *$ mod p

Notation: $a^2 = a \oplus a$, $a^3 = a \oplus a \oplus a$, …

Z_p^*

Common Groups Under Modular Arithmetic

- Additive Group of Integers Modulo prime p (Z_p)
  - Example: p=11
  - Can we find a generator?

$+ \bmod 11$

$g = 10$

$10 \bmod 11 = 10$

$g^2 \quad (10+10) \bmod 11 = 9$

$g^3 \quad (10+10+10) \bmod 11 = 8$

$g^4 \quad 7$

g^5

Common Groups Under Modular Arithmetic

- Additive Group of Integers Modulo prime p (Z_p)
  - Example: $p=11$
  - Can we find a generator?
  - ALL non-identity elements are generators for prime-order groups!

Common Groups Under Modular Arithmetic

G = { 1 ... p-1 }

- Multiplicative Group of Integers Modulo prime p (Z_p*)
  - Example: p=11
  - Can we find a generator?

a ⊕ b =

a * b mod p

g ??

g¹

g²

g³ ...

2, 4, 8, 16,

2 4 8, 5, 10, 9, 7, 3, 6

7, 5, 2, 3, 10, 4, 6, 8, 1

7 49

7, 10

Common Groups Under Modular Arithmetic

- Multiplicative Group of Integers Modulo prime p (Z_p*)
  - Example: p=11
  - Can we find a generator?

gradescope!

Hardness Assumptions Over Groups

Hardness Assumptions Over Groups

- Discrete Logarithm (DL) problem over G for random generator g:
  - Pick random $x \leftarrow \{1, 2, \ldots, \text{order}\}$
  - Compute $X = g^x$
  - Problem: Given g and X, compute x

$X = g^x$

$X = 9$

$g = 7$

$g^x = 9 \bmod 11$

Hardness Assumptions Over Groups

- Discrete Logarithm (DL) problem over G for random generator g:
  - Pick random $x \leftarrow \{1, 2, \ldots, \text{order}\}$
  - Compute $X = g^x$
  - Problem: Given g and X, compute x

- Computational Diffie-Hellman (CDH) problem:
  - Pick random $x, y \leftarrow \{1, 2, \ldots, \text{order}\}$
  - Compute $X = g^x$ and $Y = g^y$
  - Problem: Given g, X, and Y, compute g^{xy}

x Y g

Key Generation

$\mathbb{Z}_p^*$

- Public info on group G: order p and generator g

Alice

Pick secret key sk ← {1, 2, …, p}

Set public key pk ← g^{sk}

Diffie-Hellman Protocol

- Alice and Bob never met and share no secrets
- <u>Public</u> info on group G: order p and generator g

$sk_A \leftarrow x$
$pk_A \leftarrow g^x$

Alice

Bob

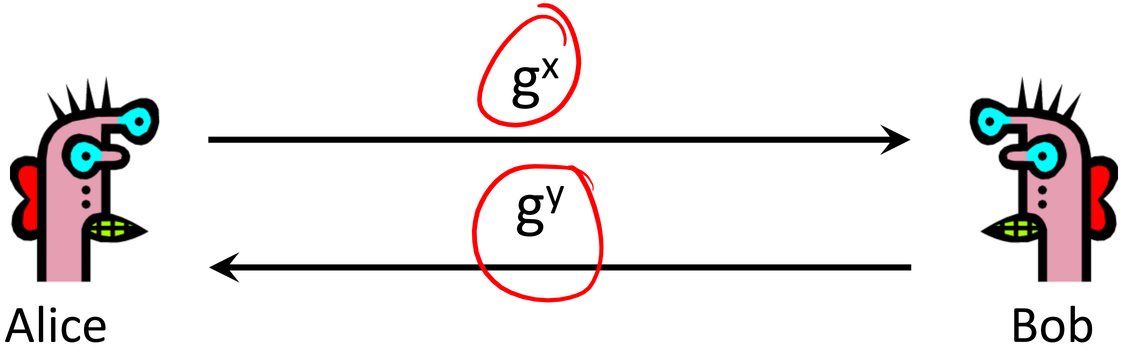
$sk_B \leftarrow y$
$pk_B \leftarrow g^y$

Diffie-Hellman Protocol

- Alice and Bob never met and share no secrets
- <u>Public</u> info on group G: order p and generator g

Adv

$sk_A \leftarrow x$
$pk_A \leftarrow g^x$



$sk_B \leftarrow y$
$pk_B \leftarrow g^y$

Diffie-Hellman Protocol

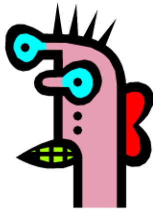
- Alice and Bob never met and share no secrets
- <u>Public</u> info on group G: order p and generator g

$sk_A \leftarrow x$
$pk_A \leftarrow g^x$


Alice

$\xrightarrow{g^x}$

$\xleftarrow{g^y}$


Bob

$sk_B \leftarrow y$
$pk_B \leftarrow g^y$

Compute shared secret $k=(g^y)^x=g^{xy}$

$Kex(P_B, S_A)$

Compute shared secret $k=(g^x)^y=g^{xy}$

Diffie-Hellman Protocol

- Alice and Bob never met and share no secrets
- Public info on group G: order p and generator g

$sk_A \leftarrow x$
$pk_A \leftarrow g^x$

Alice

$$\xrightarrow{g^x}$$

$$\xleftarrow{g^y}$$

Bob

$sk_B \leftarrow y$
$pk_B \leftarrow g^y$

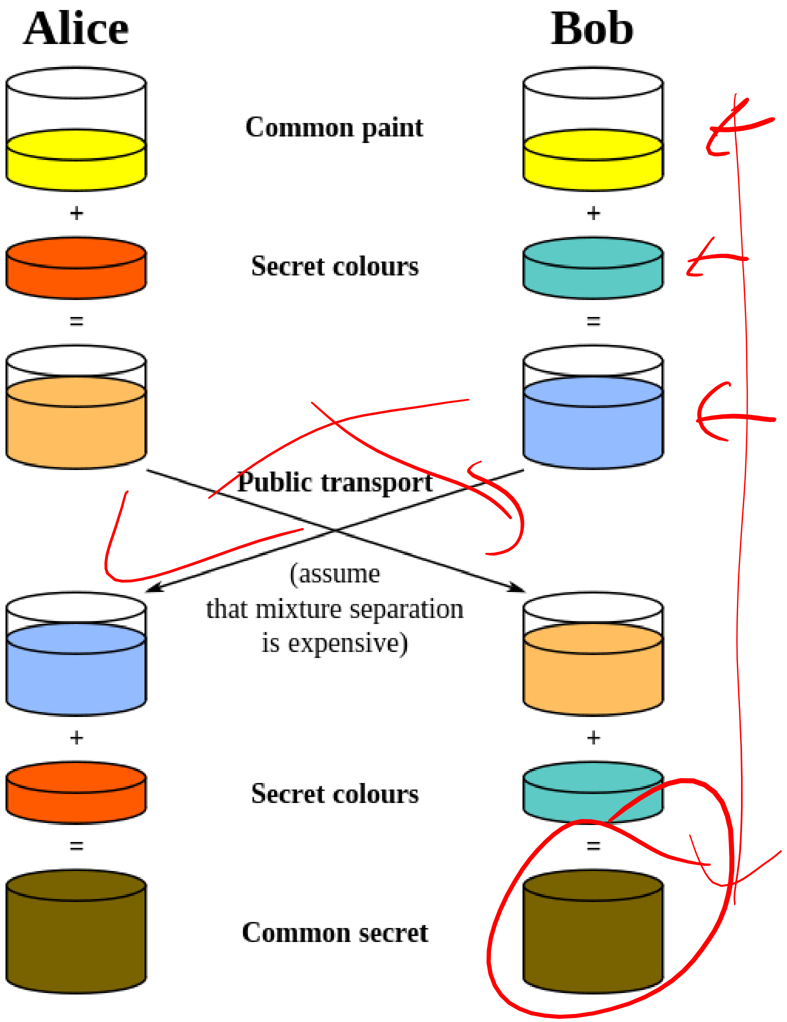
Compute shared secret $k=(g^y)^x=g^{xy}$

Compute symmetric key $K= H(g^{xy})$

Compute shared secret $k=(g^x)^y=g^{xy}$

Compute symmetric key $K= H(g^{xy})$

Diffie-Hellman: Conceptually



Common paint: p and g

Secret colors: x and y

Send over public transport:

g^x

g^y

Common secret: g^{xy}

[from Wikipedia]

Why is Diffie-Hellman Secure?

- Alice and Bob never met and share no secrets
- Public info on group G: order p and generator g

$sk_A \leftarrow x$
$pk_A \leftarrow g^x$

Alice

$g^x \longrightarrow$

$\longleftarrow g^y$

Bob

$sk_B \leftarrow y$
$pk_B \leftarrow g^y$

Compute shared secret $k=(g^y)^x=g^{xy}$

Compute symmetric key $K= H(g^{xy})$

Compute shared secret $k=(g^x)^y=g^{xy}$

Compute symmetric key $K= H(g^{xy})$

Why is Diffie-Hellman Secure?

- Alice and Bob never met and share no secrets
- Public info on group G: order p and generator g

Exactly the CDH problem!

$sk_A \leftarrow x$
$pk_A \leftarrow g^x$


Alice

$\xrightarrow{g^x}$

$\xleftarrow{g^y}$


Bob

$sk_B \leftarrow y$
$pk_B \leftarrow g^y$

Compute shared secret $k=(g^y)^x=g^{xy}$

Compute symmetric key $K= H(g^{xy})$

Compute shared secret $k=(g^x)^y=g^{xy}$

Compute symmetric key $K= H(g^{xy})$

Hardness Assumptions Over Groups

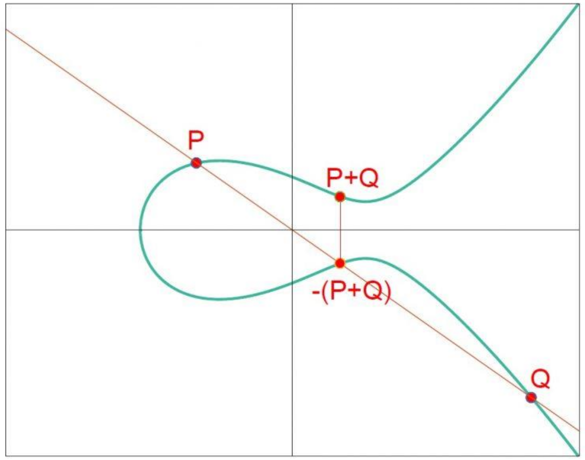
- Discrete Logarithm (DL) problem over G for generator g:
  - Pick random x ← {1, 2, …, order}
  - Compute $X = g^x$
  - Problem: Given g and X, compute x

- Computational Diffie-Hellman (CDH) problem:
  - Pick random x, y ← {1, 2, …, order}
  - Compute $X = g^x$ and $Y = g^y$
  - Problem: Given g, X, and Y, compute g^{xy}

- Caveat: Assumption doesn't hold or holds differently for different groups!
  - For ~128 bits of security:
  - Z_p: Not secure! Discrete log just corresponds to modular division!
  - Z_p*: 2048-4096 bit prime SAFE p = 2q+1 for prime q, use generator for subgroup of size q

Hardness Assumptions Over Groups

- Discrete Logarithm (DL) problem over G for generator g:
  - Pick random x ← {1, 2, …, order}
  - Compute $X = g^x$
  - Problem: Given g and X, compute x



- Computational Diffie-Hellman (CDH) problem:
  - Pick random x, y ← {1, 2, …, order}
  - Compute $X = g^x$ and $Y = g^y$
  - Problem: Given g, X, and Y, compute g^{xy}

- Caveat: Assumption doesn't hold or holds differently for different groups!
  - For ~128 bits of security:
  - Z_p: Not secure! Discrete log just corresponds to modular division!
  - Z_p*: 2048-4096 bit prime SAFE p = 2q+1 for prime q, use generator for subgroup of size q
  - Elliptic curves: (x,y) coordinates in Z_p for 256 bit prime p

Person-in-the-Middle Attacks

- Diffie-Hellman protocol (by itself) does not provide integrity (against <u>active</u> attackers)

$sk_A \leftarrow x$
$pk_A \leftarrow g^x$

$sk_B \leftarrow y$
$pk_B \leftarrow g^y$

Stepping Back: Asymmetric Crypto

- We've just seen session key establishment
  - Can then use shared key for symmetric crypto
- Next: public key encryption
  - For confidentiality
- Then: digital signatures
  - For integrity