# CSE467: Computer Security

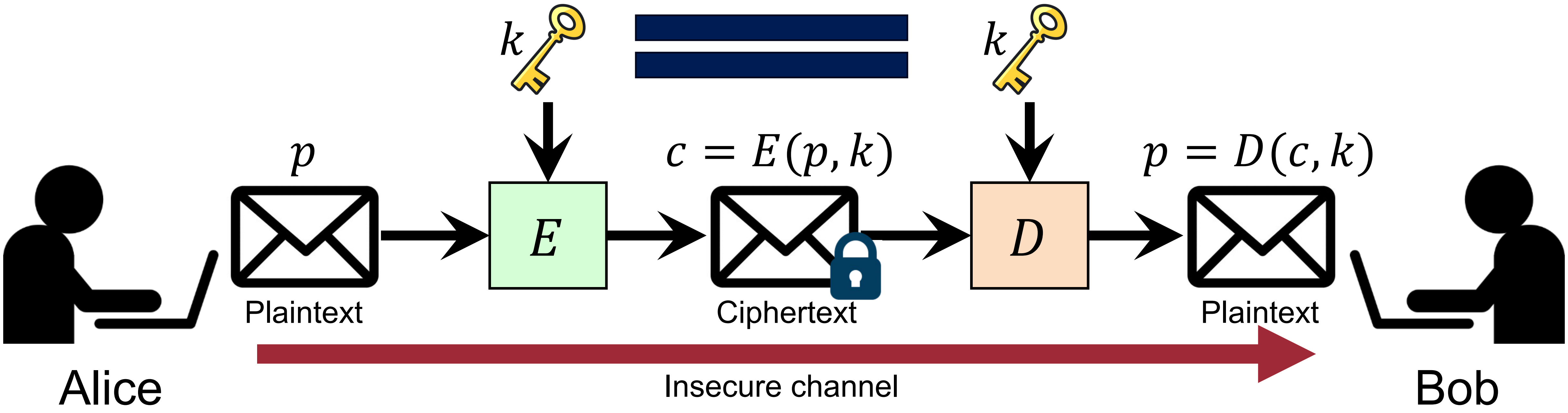
## 5-2. Asymmetric-key Encryption

Seongil Wi


Department of Computer Science and Engineering

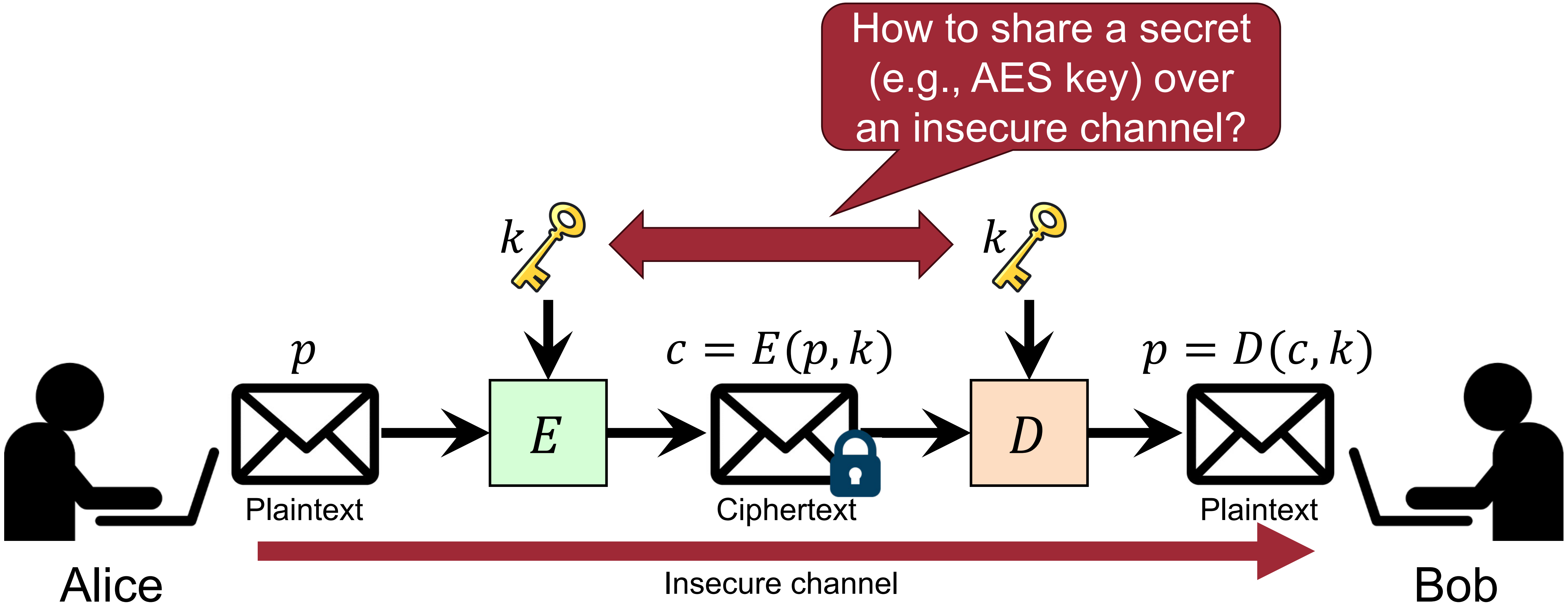
# Recap: Symmetric-key Encryption

- **Symmetric**: the encryption and decryption keys *are the same*



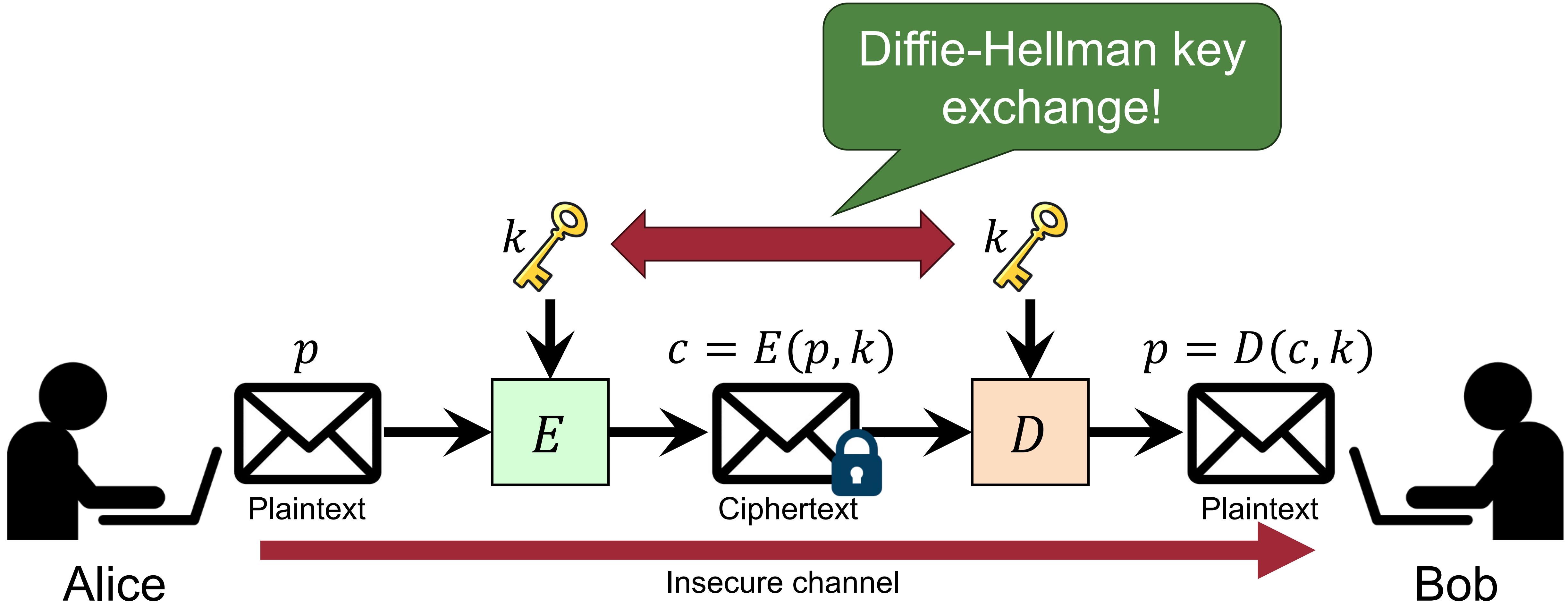
# Recap: Symmetric-key Encryption

- **Symmetric**: the encryption and decryption keys *are the same*



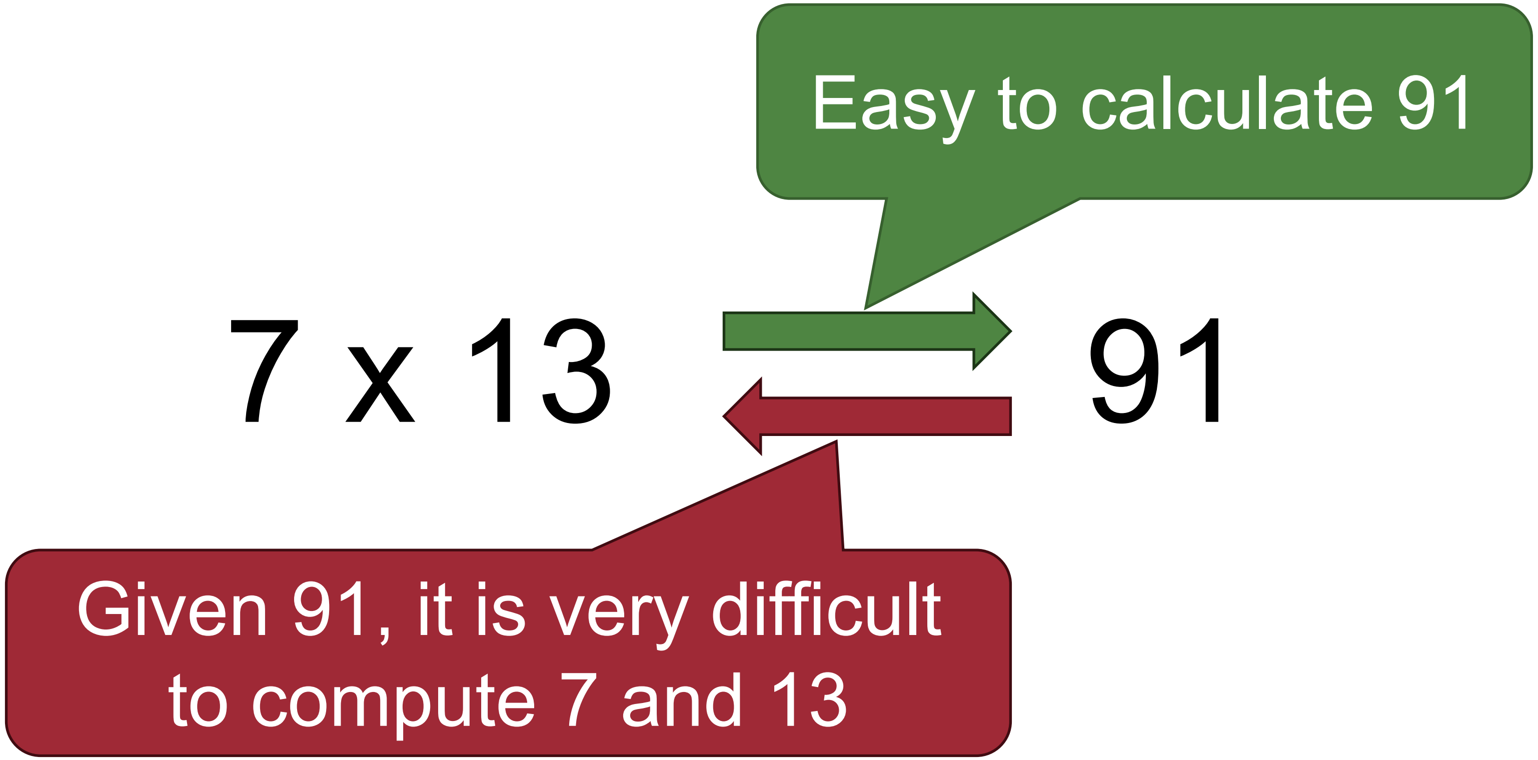
# Motivation of the Diffie-Hellman Key Exchange

- **Symmetric**: the encryption and decryption keys *are the same*



# Core Idea: One-way Function

- Easy in one direction, but hard in the reverse direction
  - $f$ is easy to compute, but $f^{-1}$ is difficult to compute




***Integer Factorization Problem***

# Core Idea: One-way Function

- Easy in one direction, but hard in the reverse direction
  - $f$ is easy to compute, but $f^{-1}$ is difficult to compute

Given $\underline{g, p, x}$, it is easy to calculate $y$

$$g^x \bmod p \longrightarrow y$$

# Core Idea: One-way Function

- Easy in one direction, but hard in the reverse direction
  - $f$ is easy to compute, but $f^{-1}$ is difficult to compute

$$g = 3$$
$$p = 5$$
$$x = 2$$

$$g^x \bmod p \longrightarrow y = ?$$

# Core Idea: One-way Function

- Easy in one direction, but hard in the reverse direction
  - $f$ is easy to compute, but $f^{-1}$ is difficult to compute

$g = 3$
$p = 5$
$x = 2$

Given $\underline{g, p, x}$, it is easy to calculate $y$

$$g^x \bmod p \longrightarrow y = 4$$

# Core Idea: One-way Function

- Easy in one direction, but hard in the reverse direction
  - $f$ is easy to compute, but $f^{-1}$ is difficult to compute

Given $\underline{g, p, x}$, it is easy to calculate $y$

$$g^x \bmod p \rightleftarrows y$$

Given $\underline{g, p, y}$, it is very difficult to compute $x$

# Core Idea: One-way Function



- Easy in one direction, but hard in the reverse direction
  - $f$ is easy to compute, but $f^{-1}$ is difficult to compute

$$g = 3$$
$$p = 5$$
$$x = ?$$

$$g^x \bmod p \longleftarrow y = 4$$

Given $g, p, y$, it is very difficult to compute $x$

***Discrete Logarithm Problem***

# Core Idea: One-way Function

- Easy in one direction, but hard in the reverse direction
  - $f$ is easy to compute, but $f^{-1}$ is difficult to compute

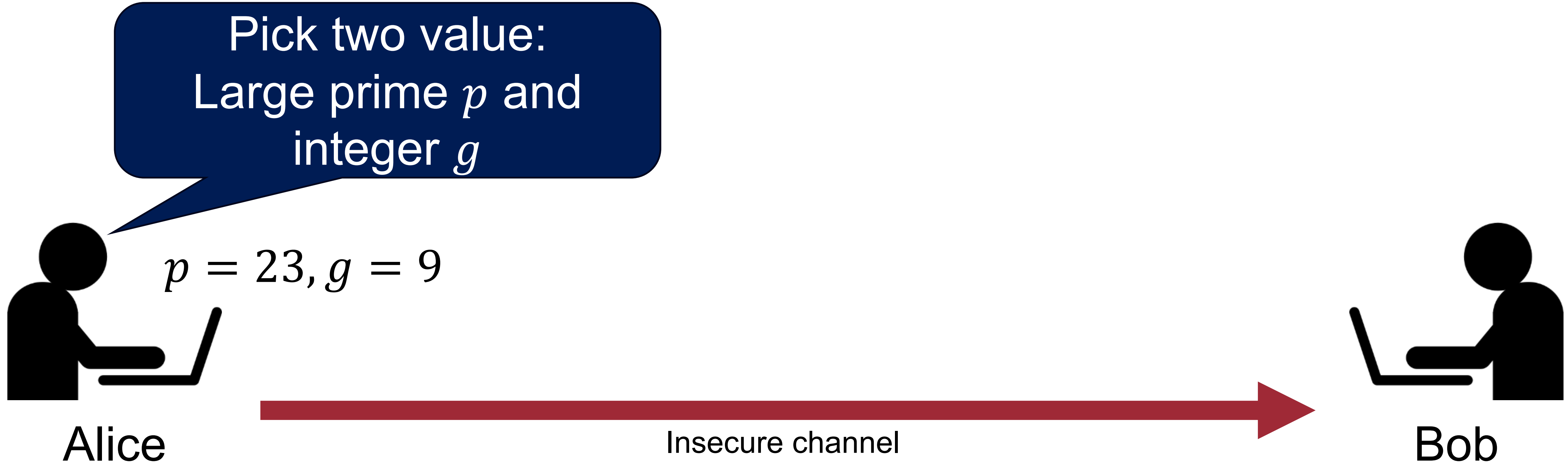
$$g = 3$$
$$p = 5$$
$$x = ?$$

$$g^x \bmod p \longleftarrow y = 4$$

There is no efficient algorithm known for computing discrete logarithms in general

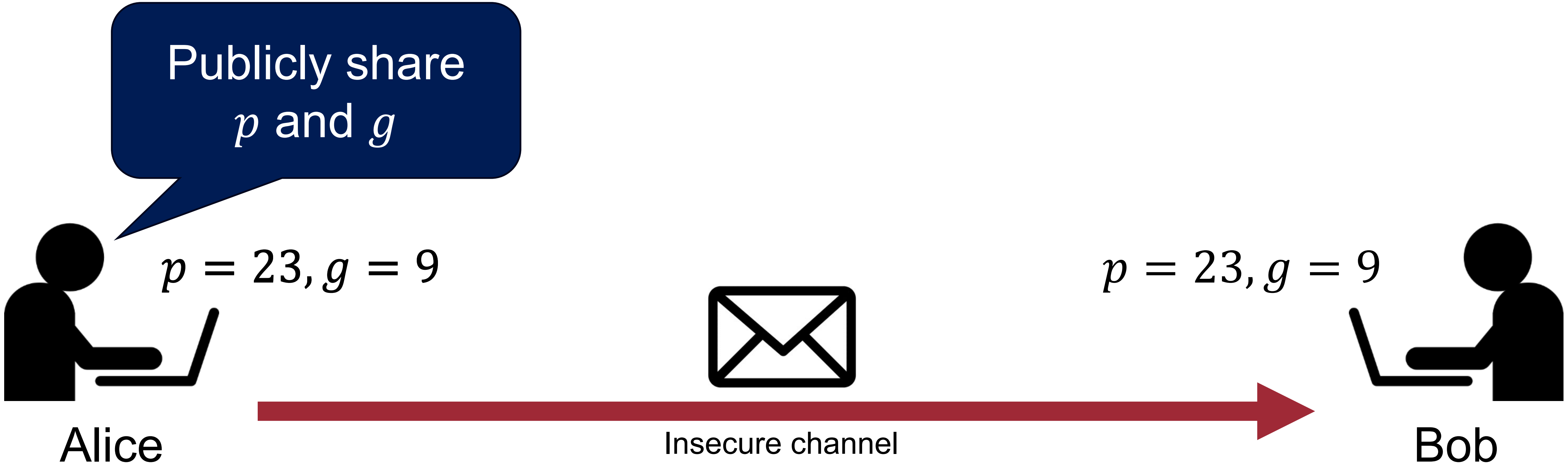
# Diffie-Hellman Key Exchange (1)

$$g^x \, mod \, p \rightleftarrows y$$



# Diffie-Hellman Key Exchange (2)

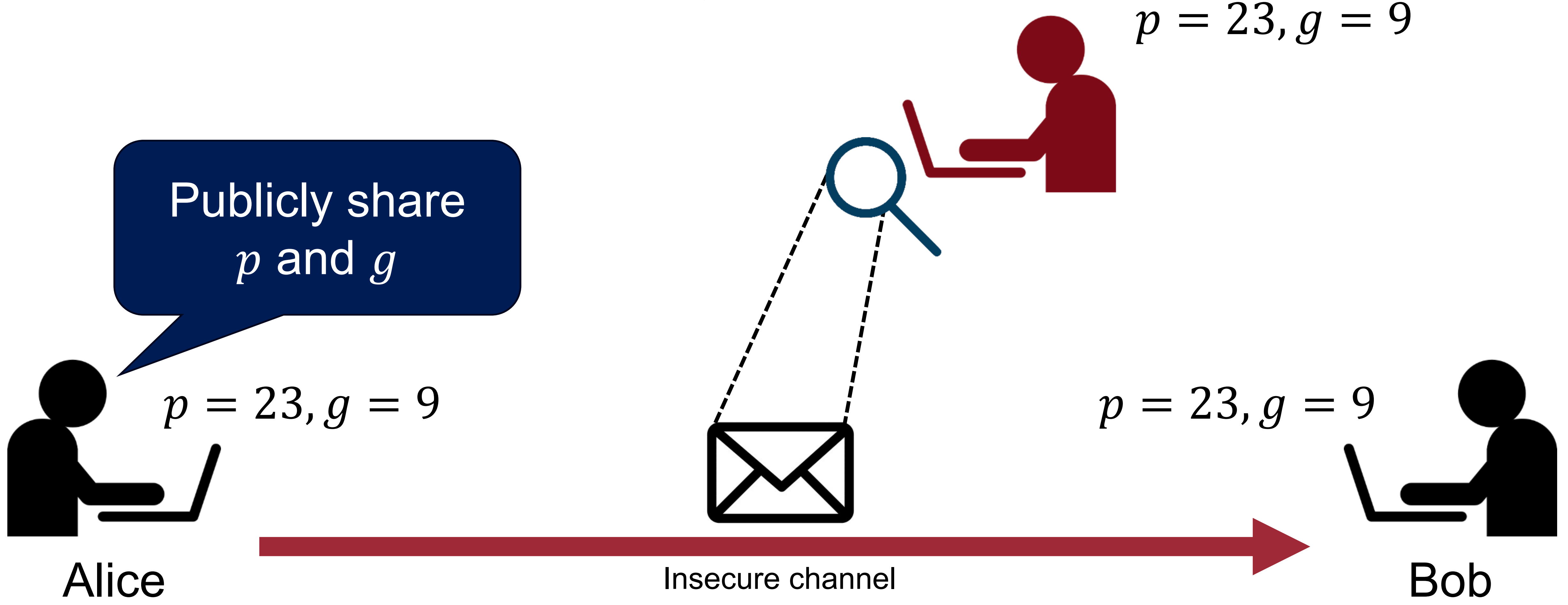
$$g^x \bmod p \rightleftarrows y$$



# Diffie-Hellman Key Exchange (2)

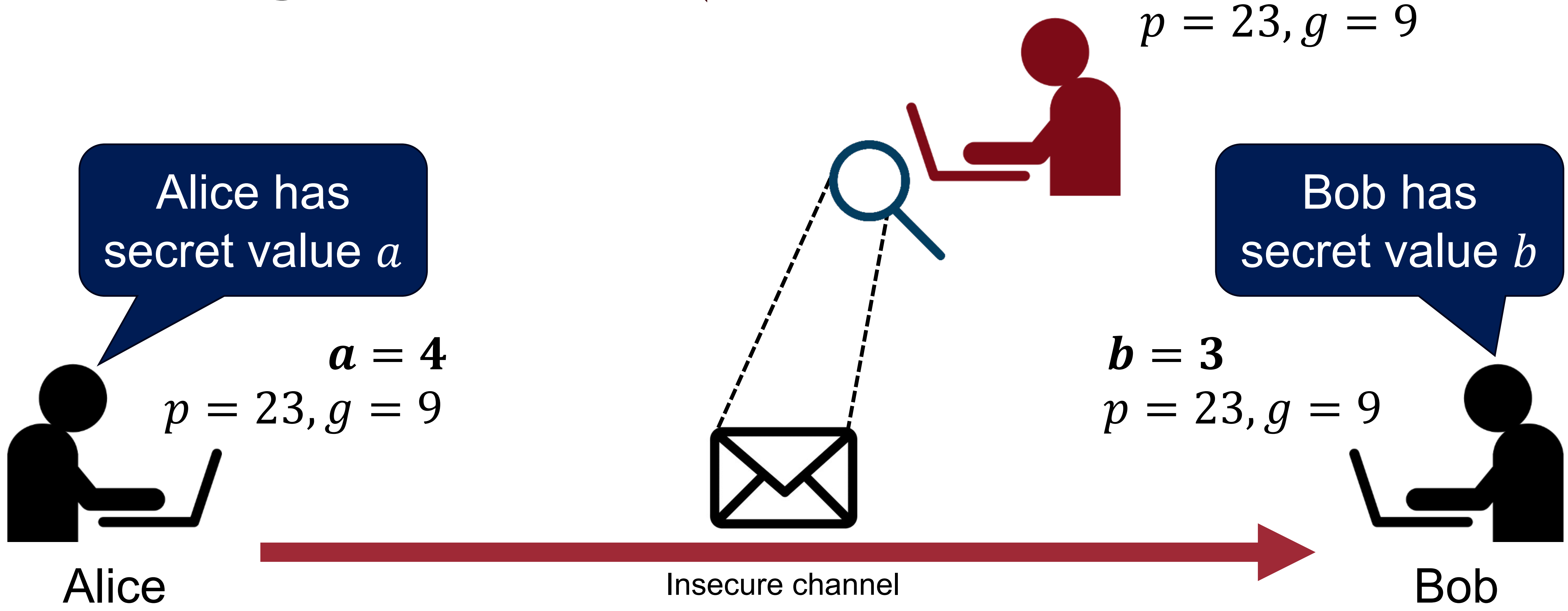


$$g^x \bmod p \rightleftarrows y$$



# Diffie-Hellman Key Exchange (3)

$$g^x \bmod p \rightleftarrows y$$

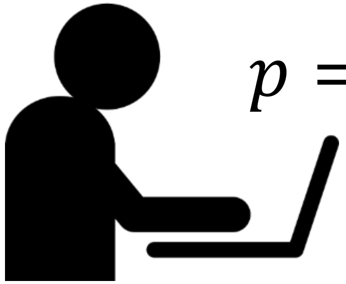


# Diffie-Hellman Key Exchange (4)

$$g^x \, mod \, p \rightleftarrows y$$

$p = 23, g = 9$

Send
$A = g^a \, mod \, p$
to Bob



$\boldsymbol{a = 4}$
$p = 23, g = 9$
$A = (g^a \, mod \, p) = 6$

$\boldsymbol{b = 3}$
$p = 23, g = 9$
$A = (g^a \, mod \, p) = 6$

Alice

Insecure channel

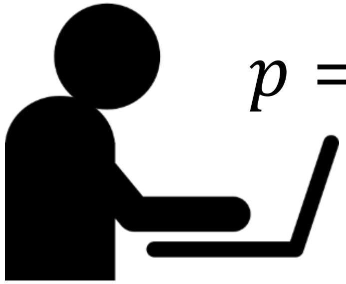
Bob

# Diffie-Hellman Key Exchange (4)

$$g^x \bmod p \rightleftarrows y$$

$p = 23, g = 9$

$\mathrm{A} = (g^a \bmod p) = 6$



$\boldsymbol{a = 4}$

$p = 23, g = 9$

$A = (g^a \bmod p) = 6$

$\boldsymbol{b = 3}$

$p = 23, g = 9$

$A = (g^a \bmod p) = 6$

Alice

Insecure channel

Bob

# Diffie-Hellman Key Exchange (4)

$$g^x \bmod p \quad \longleftarrow \quad y$$

$p = 23, g = 9$

$\mathrm{A} = (g^a \bmod p) = 6$

Given $\underline{g, p, y}$, it is very difficult to compute $a$

$a = 4$

$p = 23, g = 9$

$\mathrm{A} = (g^a \bmod p) = 6$

$p = 23, g = 9$

$\mathrm{A} = (g^a \bmod p) = 6$

Alice

Insecure cha...

Bob

***Discrete Logarithm Problem***

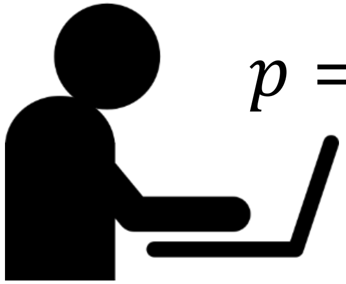
# Diffie-Hellman Key Exchange (4)

$$g^x \bmod p \rightleftarrows y$$

$p = 23, g = 9$

$A = (g^a \bmod p) = 6$

Send
$B = g^b \bmod p$
to Alice



$\boldsymbol{a = 4}$

$p = 23, g = 9$

$A = (g^a \bmod p) = 6$

$B = (g^b \bmod p) = 16$

Alice

$\boldsymbol{b = 3}$

$p = 23, g = 9$

$A = (g^a \bmod p) = 6$

$B = (g^b \bmod p) = 16$

Bob

Insecure channel

# Diffie-Hellman Key Exchange (4)



$$g^x \bmod p \rightleftarrows y$$

$p = 23, g = 9$

$A = (g^a \bmod p) = 6$

$B = (g^b \bmod p) = 16$

$\boldsymbol{a = 4}$

$p = 23, g = 9$

$A = (g^a \bmod p) = 6$

$B = (g^b \bmod p) = 16$

$\boldsymbol{b = 3}$

$p = 23, g = 9$

$A = (g^a \bmod p) = 6$

$B = (g^b \bmod p) = 16$

Alice

Insecure channel

Bob

# Diffie-Hellman Key Exchange (5)

**Symmetric key:**

$$K = g^{ab} \; mod \; p$$

$p = 23, g = 9$

$A = (g^a \; mod \; p) = 6$

$B = (g^b \; mod \; p) = 16$

$\boldsymbol{a = 4}$

$p = 23, g = 9$

$A = (g^a \; mod \; p) = 6$

$B = (g^b \; mod \; p) = 16$

$\boldsymbol{b = 3}$

$p = 23, g = 9$

$A = (g^a \; mod \; p) = 6$

$B = (g^b \; mod \; p) = 16$

Alice

Insecure channel

Bob

# Diffie-Hellman Key Exchange (5)

**Symmetric key:**

$K = g^{ab} \bmod p$

$p = 23, g = 9$
$A = (g^a \bmod p) = 6$
$B = (g^b \bmod p) = 16$

$K = (B^a \bmod p) = (g^{ab} \bmod p)$
$= (16^4 \bmod 23) = 9$

**Theorem:**

$((X \bmod p)^k \bmod p) = (X^k \bmod p)$

$a = 4$

$p = 23, g = 9$
$A = (g^a \bmod p) = 6$
$B = (g^b \bmod p) = 16$

$b = 3$

$p = 23, g = 9$
$A = (g^a \bmod p) = 6$
$B = (g^b \bmod p) = 16$

Alice

Insecure channel

Bob

# Diffie-Hellman Key Exchange (5)

**Symmetric key:**

$$K = g^{ab} \ mod \ p$$

$p = 23, g = 9$

$A = (g^{a} \ mod \ p) = 6$

$B = (g^{b} \ mod \ p) = 16$

$K = (B^{a} \ mod \ p) = (g^{ab} mod \ p)$
$= (16^{4} \ mod \ 23) = 9$

$K = (A^{b} \ mod \ p) = (g^{ab} mod \ p)$
$= (6^{3} \ mod \ 23) = 9$

$\boldsymbol{a = 4}$

$p = 23, g = 9$

$A = (g^{a} \ mod \ p) = 6$

$B = (g^{b} \ mod \ p) = 16$

$\boldsymbol{b = 3}$

$p = 23, g = 9$

$A = (g^{a} \ mod \ p) = 6$

$B = (g^{b} \ mod \ p) = 16$

Alice

Insecure channel

Bob

# Security of the Diffie-Hellman Key Exchange



**Symmetric key:**

$$K = g^{ab} \ mod \ p$$

$p = 23, g = 9$

$A = (g^a \ mod \ p) = 6$

$B = (g^b \ mod \ p) = 16$

The attacker cannot efficiently compute $(g^{ab} \ mod \ p)$ ***without knowing*** $a$ ***and*** $b$

# Why should $p$ be Prime?

**Symmetric key:**

$$K = g^{ab} \ mod \ p$$

$g = 2$

Too simple key pattern that can be inferred

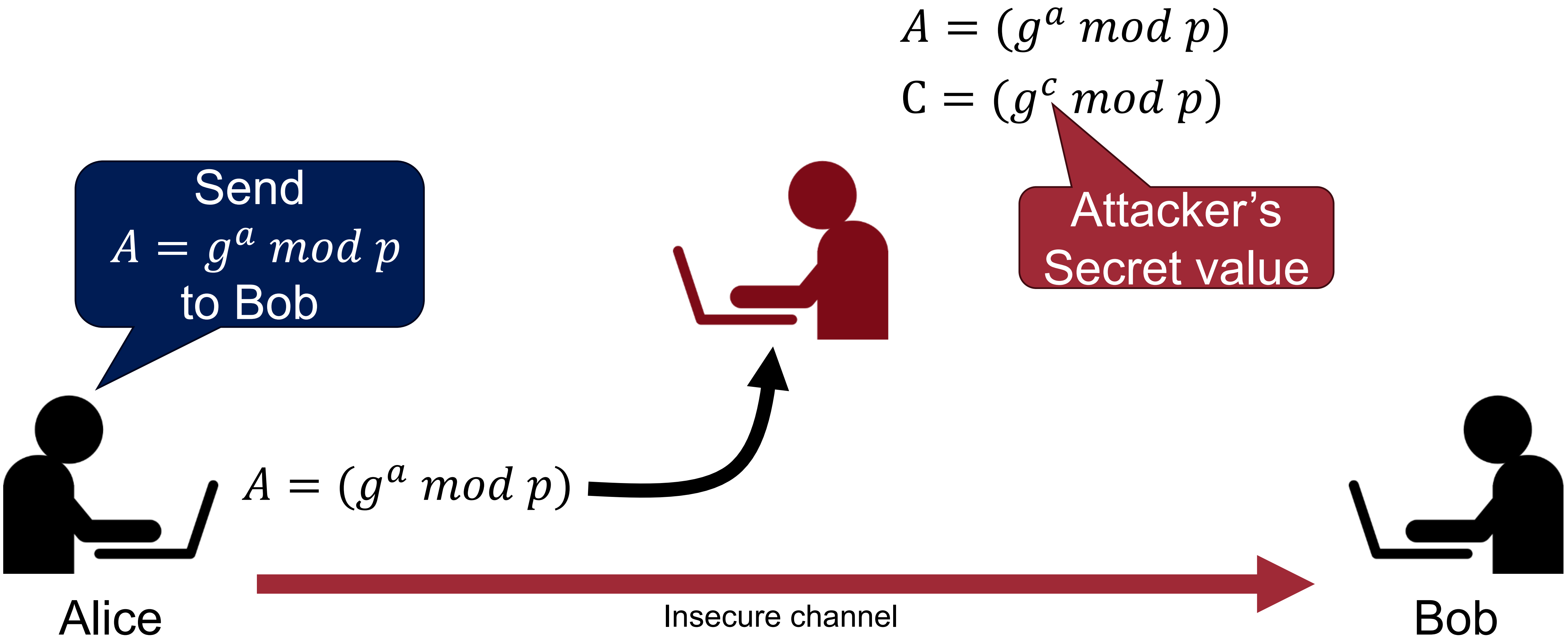
$p = 11$

- $2^0 \ mod \ 11 = 1$
- $2^1 \ mod \ 11 = 2$
- $2^2 \ mod \ 11 = 4$
- $2^3 \ mod \ 11 = 8$
- $2^4 \ mod \ 11 = 5$
- $2^5 \ mod \ 11 = 10$
- $2^6 \ mod \ 11 = 9$
- $2^7 \ mod \ 11 = 7$
- $2^8 \ mod \ 11 = 3$
- $2^9 \ mod \ 11 = 6$
- $2^{10} \ mod \ 11 = 1$

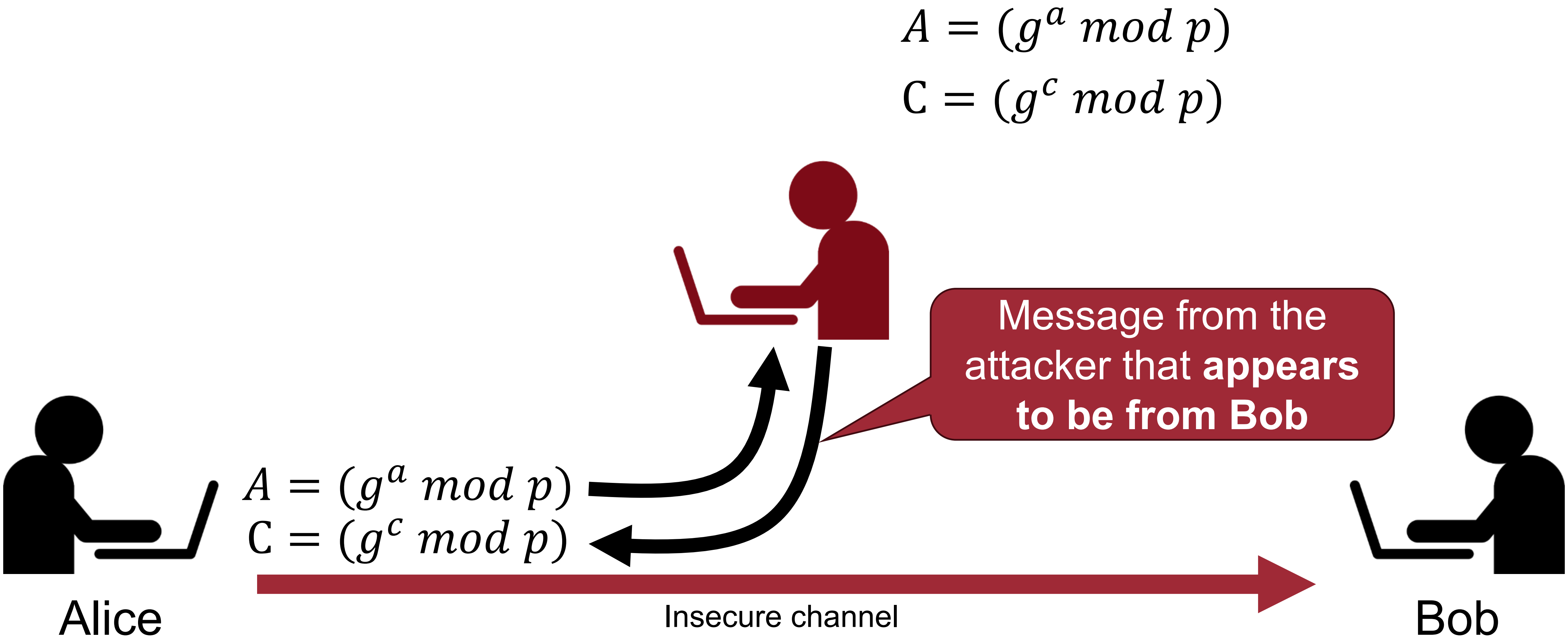
$p = 12$

- $2^0 \ mod \ 12 = 1$
- $2^1 \ mod \ 12 = 2$
- $2^2 \ mod \ 12 = 4$
- $2^3 \ mod \ 12 = 8$
- $2^4 \ mod \ 12 = 4$
- $2^5 \ mod \ 12 = 8$
- $2^6 \ mod \ 12 = 4$
- $2^7 \ mod \ 12 = 8$
- $2^8 \ mod \ 12 = 4$
- $2^9 \ mod \ 12 = 8$
- $2^{10} \ mod \ 12 = 4$

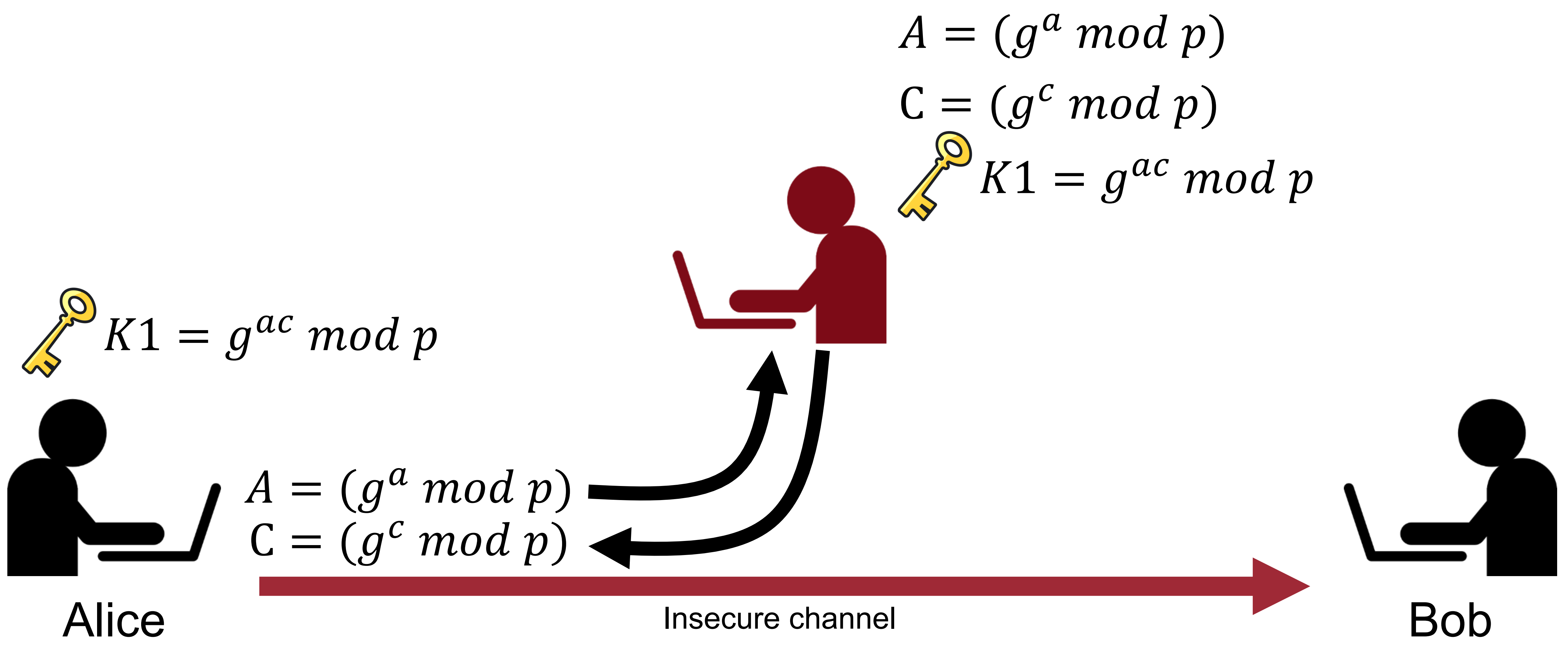
# Man-in-the-Middle Attack in DH Key Exchange



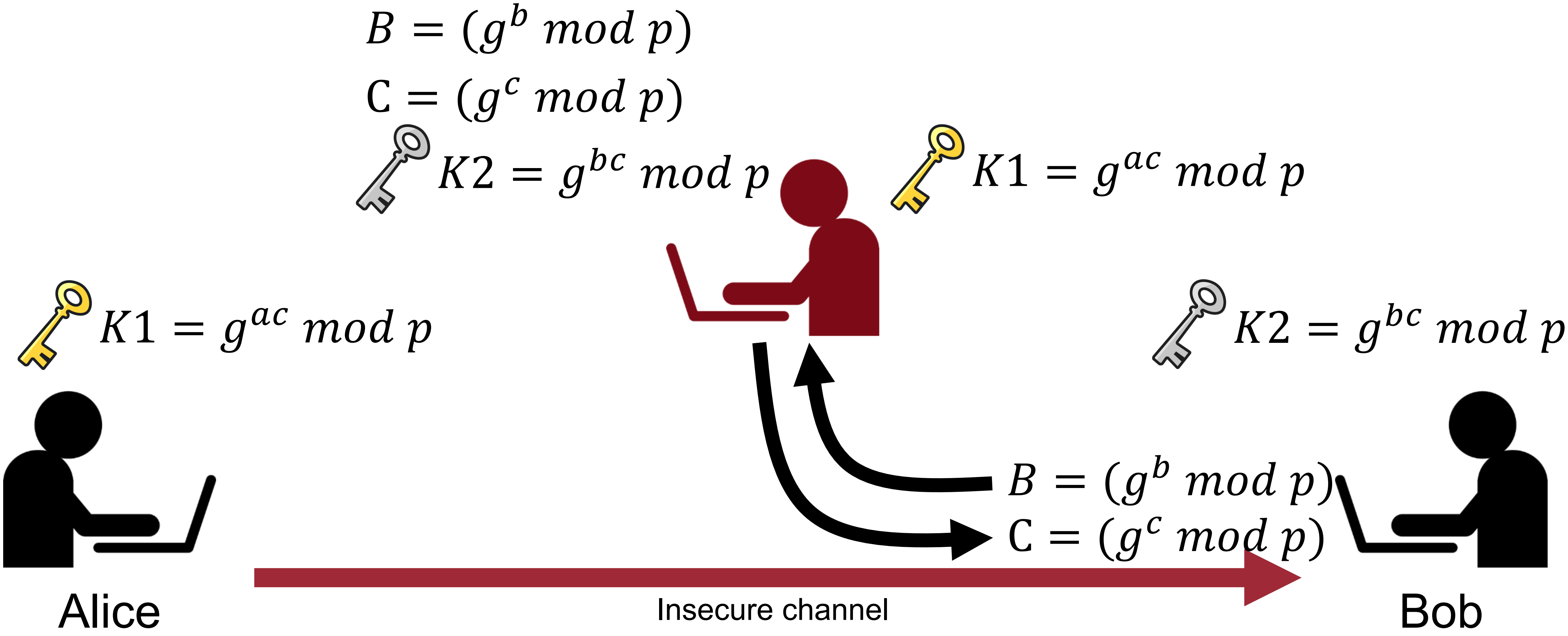
# Man-in-the-Middle Attack in DH Key Exchange



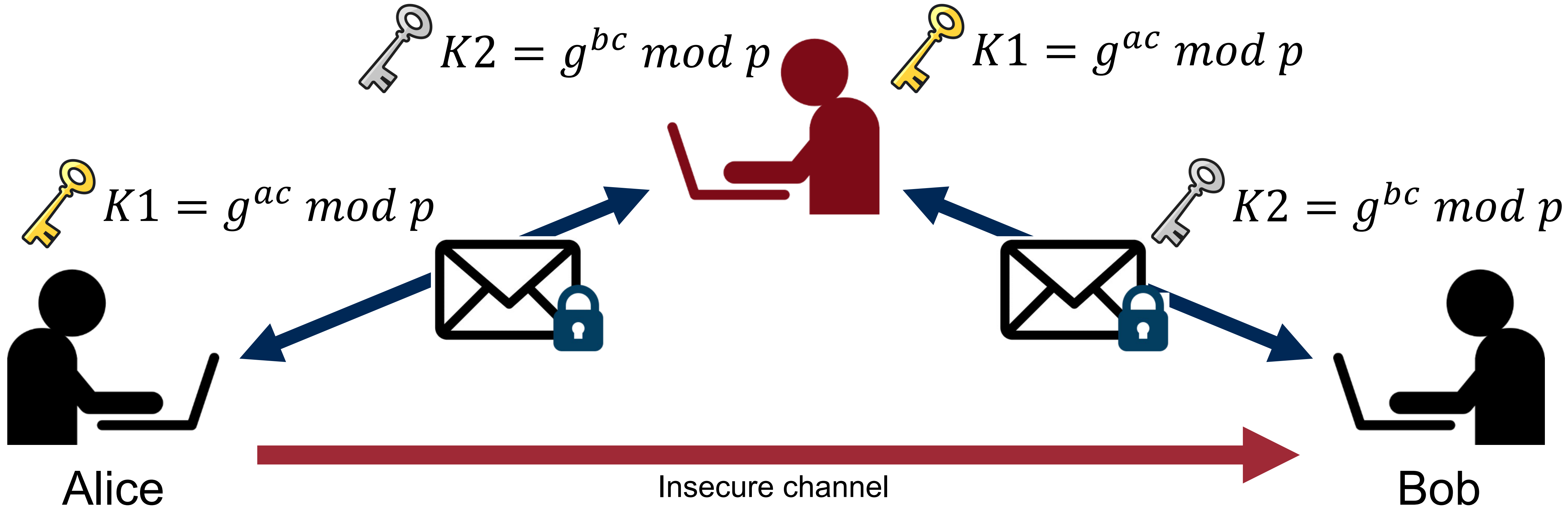
# Man-in-the-Middle Attack in DH Key Exchange



# Man-in-the-Middle Attack in DH Key Exchange



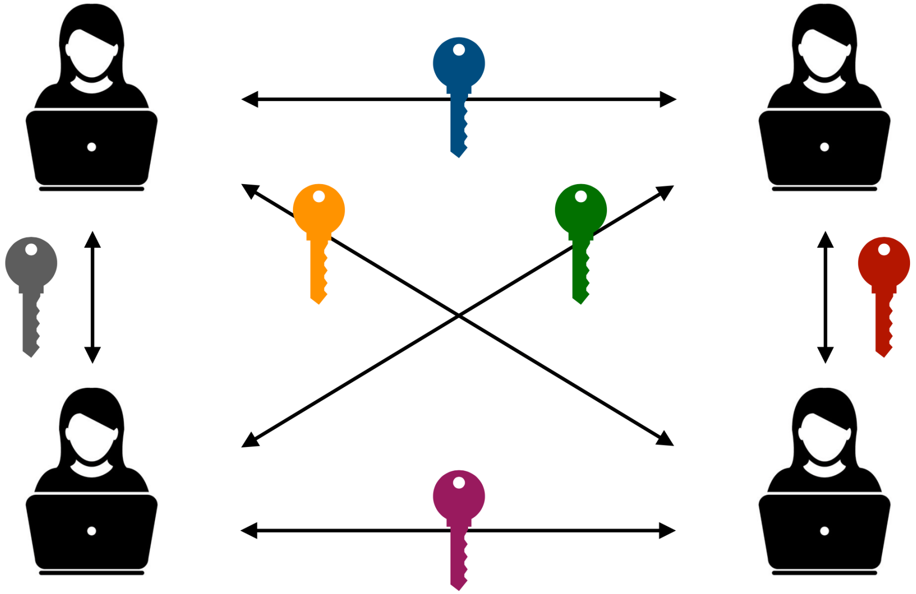
# Man-in-the-Middle Attack in DH Key Exchange



# Symmetric-key Cryptography

- Recap: the same key shared between two parties
- What happens if there are many users?
  - $n$ users: $\binom{n}{2} = n(n-1)/2$
  - Example: 100 users → 4,950 keys
- Key distribution and maintenance problem

How to solve this issue?

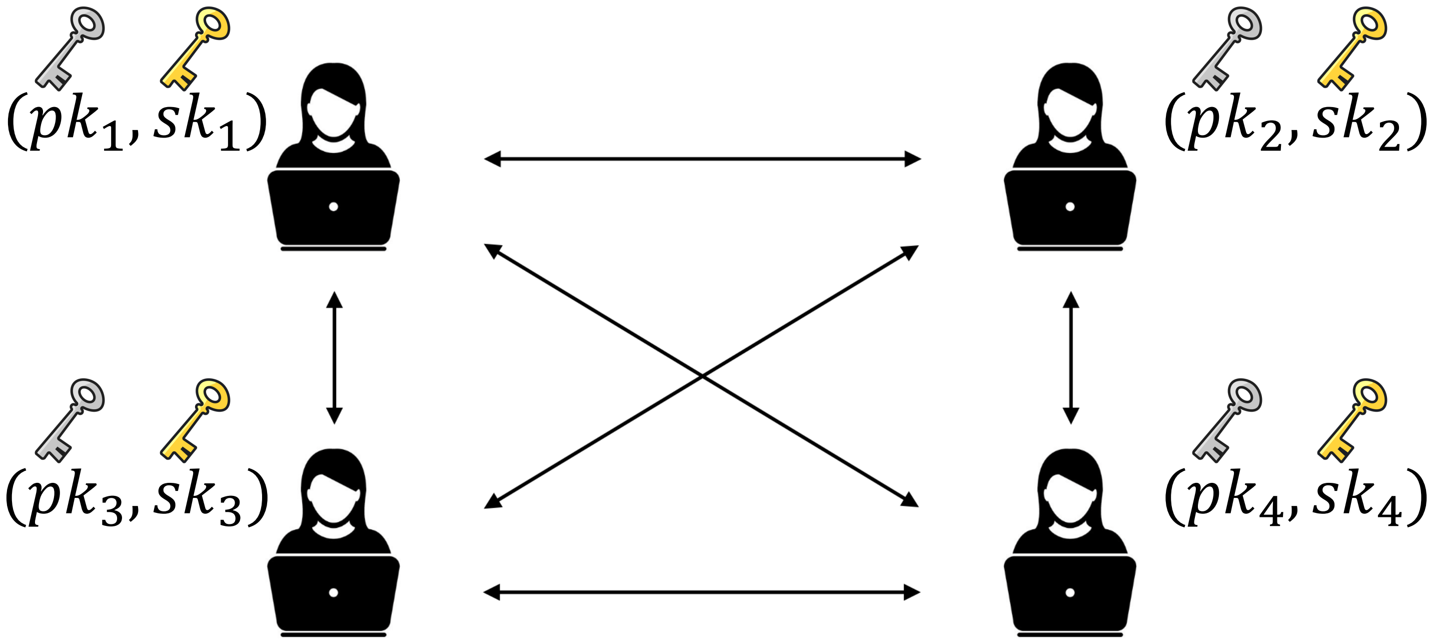


# Asymmetric-key Cryptography

- Each party has two distinct keys: public key and private key
  - Also known as public-key algorithm
- Invented in 1976 by Diffie and Hellman (ACM Turing Award 2015)

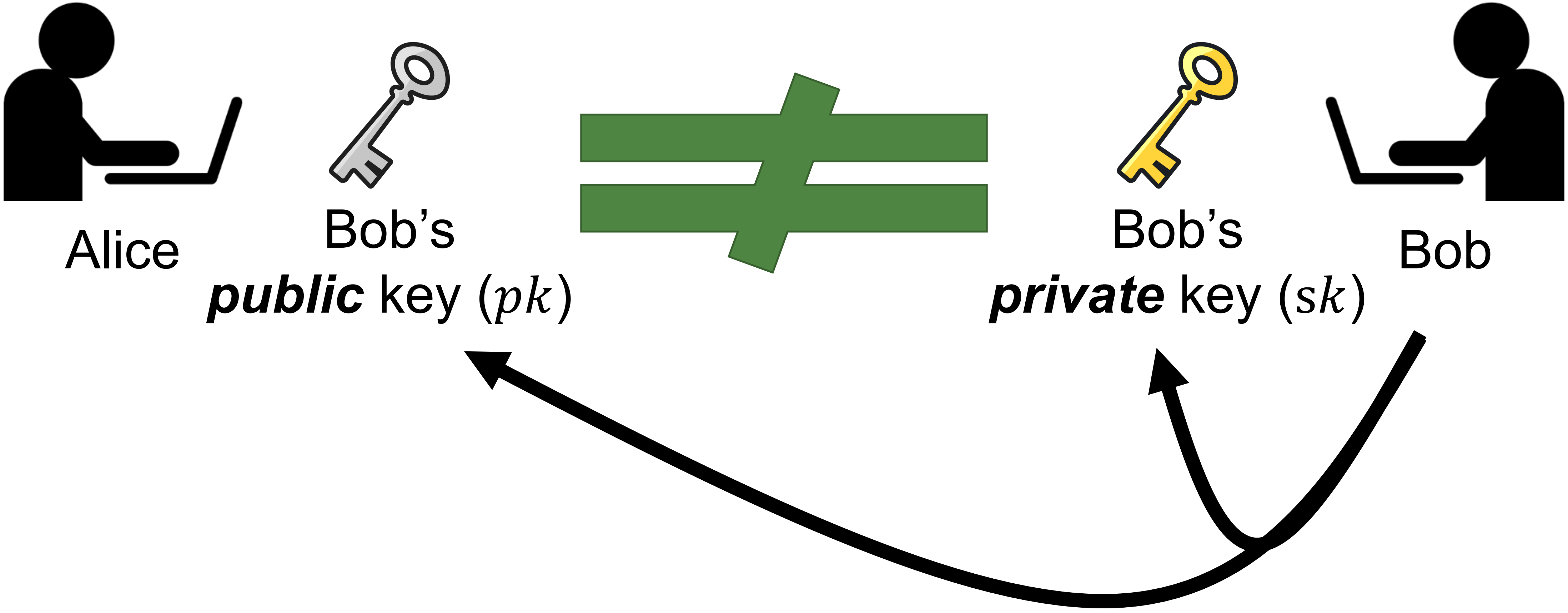
# Asymmetric-key Cryptography

- $pk$: public key, widely disseminated, used for encryption
- $sk$: private key kept secretly, used for decryption
- $n$ users: $2n$ keys



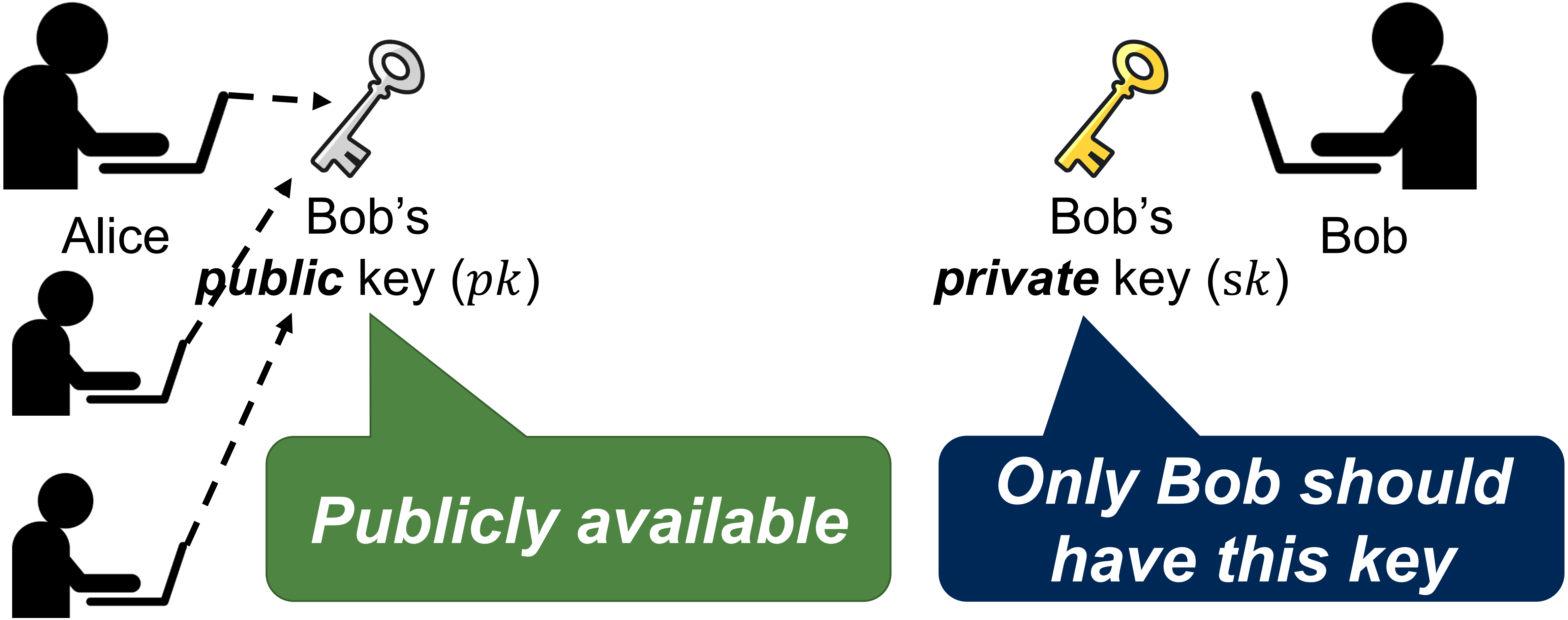
# Asymmetric-key Cryptography

- $pk$: public key, widely disseminated, used for encryption
- $sk$: private key kept secretly, used for decryption



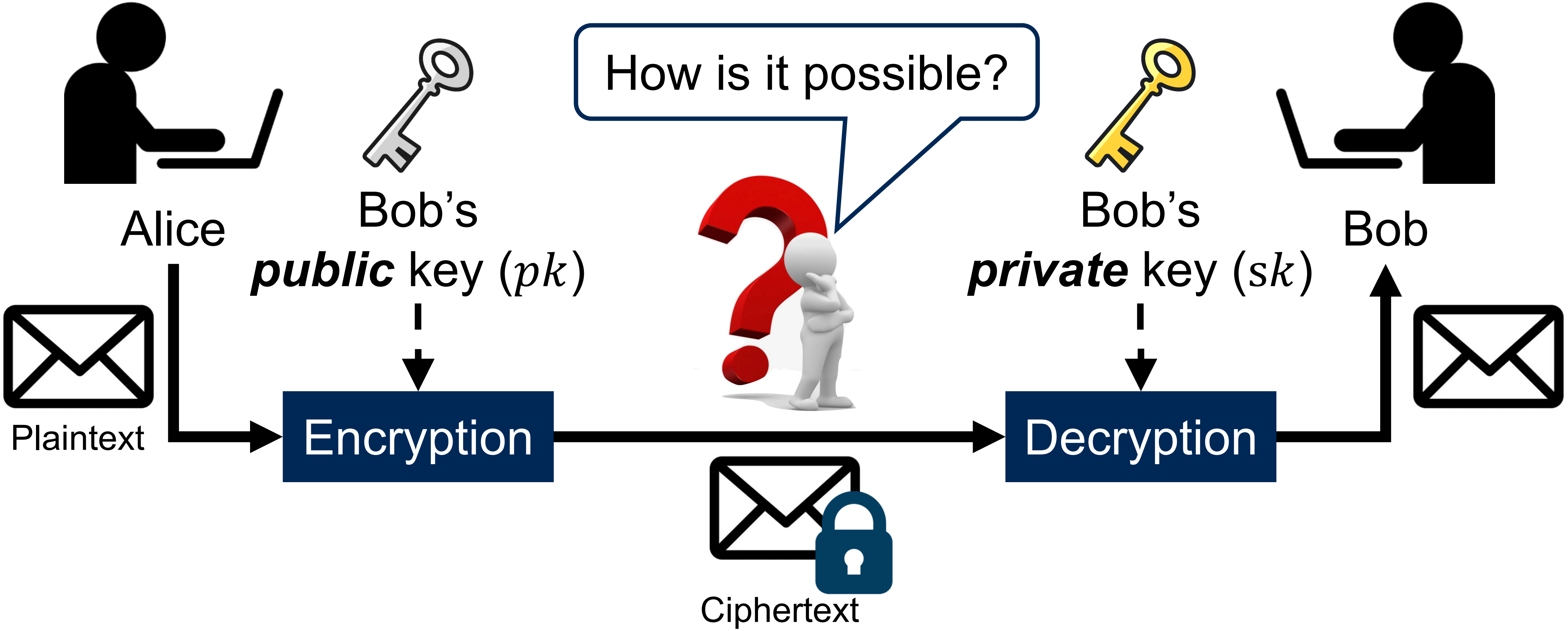
# Asymmetric-key Cryptography

- $pk$: public key, widely disseminated, used for encryption
- $sk$: private key kept secretly, used for decryption



# Asymmetric-key Cryptography

- $pk$: public key, widely disseminated, used for encryption
- $sk$: private key kept secretly, used for decryption



# RSA Cryptosystem



- Invented by Rivest, Shamir, and Adleman (MIT) in 1977
  - ACM Turing award in 2002

- Rely on the practical difficulty of factoring the product of two large prime numbers
  - Security based on *Integer Factorization Problem*

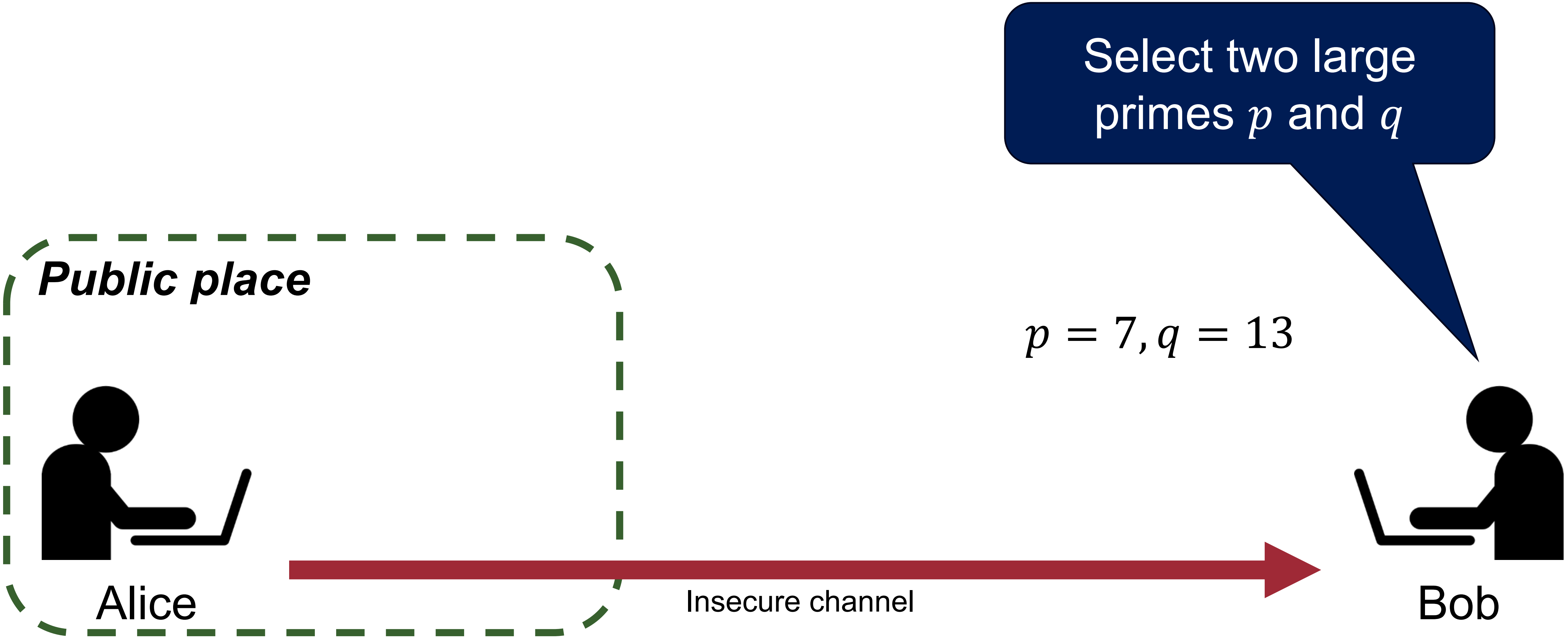
# Integer Factorization Problem

Given large prime $p$ and $q$,
it is easy to calculate $n$

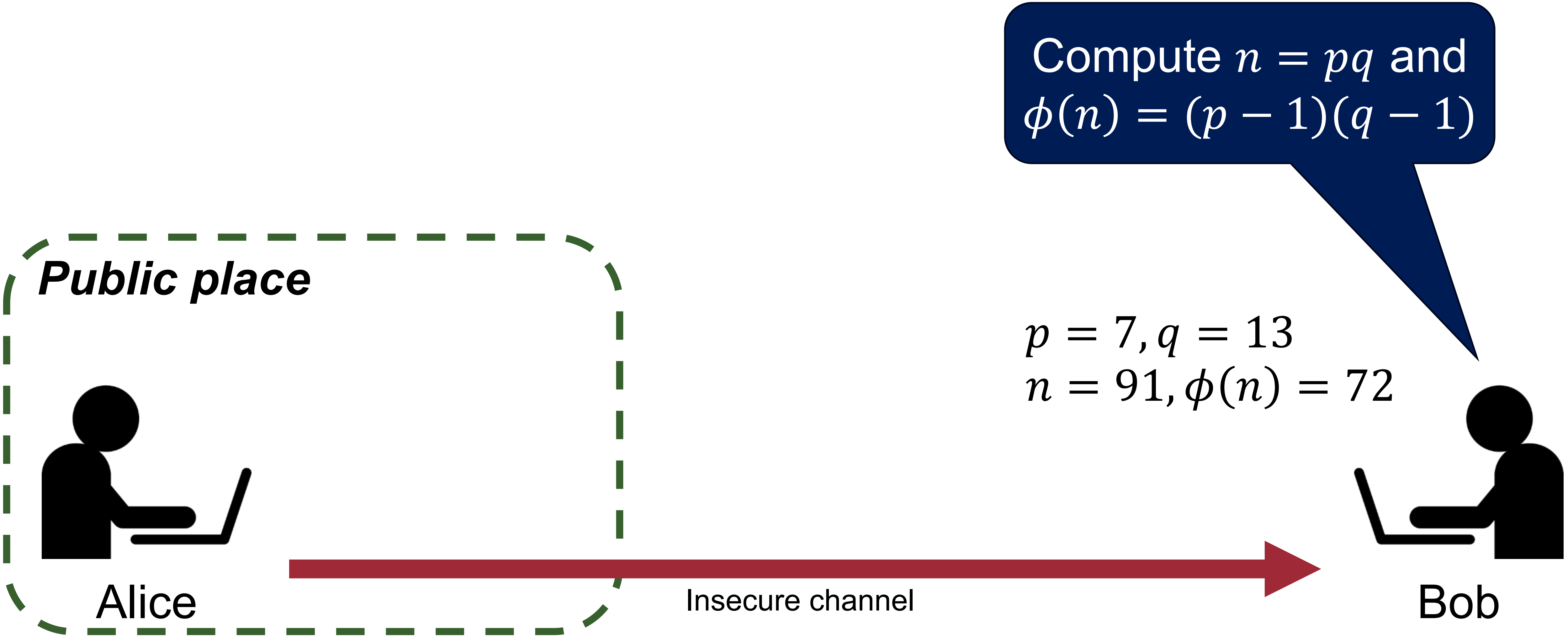
$$p \times q \rightleftarrows n$$

Given $\underline{n}$, it is very difficult
to compute $x$

# RSA Algorithm (1): Key Generation



# RSA Algorithm (1): Key Generation



# RSA Algorithm (1): Key Generation

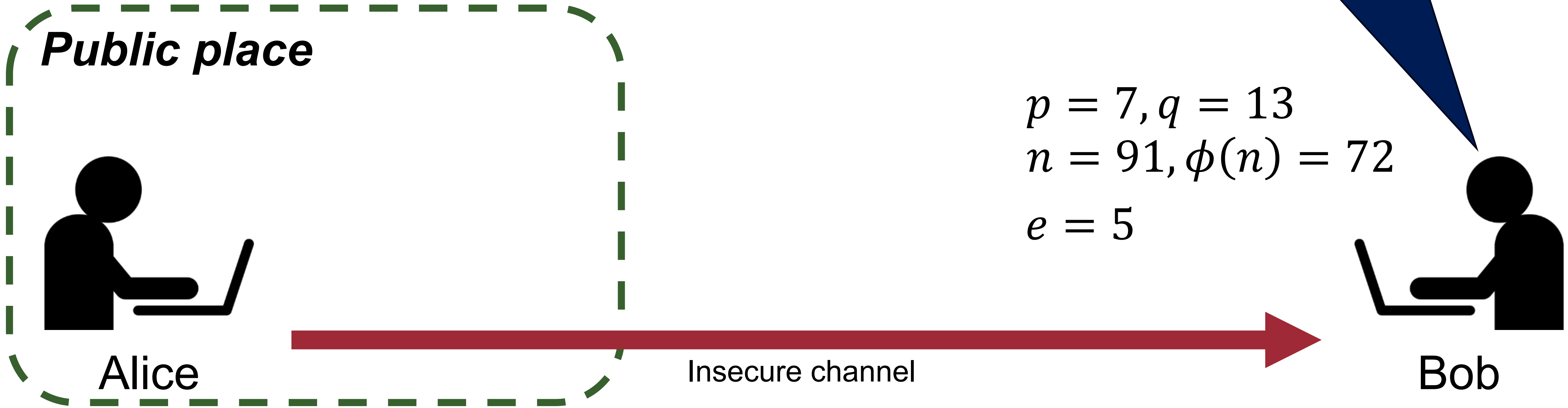


Choose $e$ s.t.

- $1 < e < \phi(n)$ **and**
- $\gcd(\phi(n), e) = 1$

$p = 7, q = 13$
$n = 91, \phi(n) = 72$

$e = 5$



# RSA Algorithm (1): Key Generation

How to find $d$?
→ Extended Euclidean Algorithm!

Choose $d$ s.t.

- $1 < d < \phi(n)$ **and**
- $(ed \bmod \phi(n)) = 1$

***Public place***

$p = 7, q = 13$
$n = 91, \phi(n) = 72$

$e = 5$
$d = 29$

Alice

Insecure channel

Bob

# Euclidean Algorithm



**Goal:** Finding Greatest Common Divisor (GCD)

**Fact 1**: $\gcd(a, 0) = a$

**Fact 2**: $\gcd(a, b) = \gcd(b, r)$, where $r$ is the remainder of dividing $a$ by $b$ $(a > b)$

***Example***

$\gcd(72, 5)$

$p = 7, q = 13$
$n = 91, \phi(n) = 72$

$e = 5$
$d = 29$

Choose $e$ s.t.

- $1 < e < \phi(n)$ **and**
- $\gcd(\phi(n), e) = 1$

# Euclidean Algorithm



**Goal:** Finding Greatest Common Divisor (GCD)

**Fact 1**: $\gcd(a, 0) = a$
**Fact 2**: $\gcd(a, b) = \gcd(b, r)$, where $r$ is the remainder of dividing $a$ by $b$ $(a > b)$

***Example***

$\gcd(72, 5) \qquad 72 = (5 * 14) + 2$

# Euclidean Algorithm



**Goal:** Finding Greatest Common Divisor (GCD)

**Fact 1**: $\gcd(a, 0) = a$
**Fact 2**: $\gcd(a, b) = \gcd(b, r)$, where $r$ is the remainder of dividing $a$ by $b$ $(a > b)$

***Example***

$\gcd(72, 5)$ $\quad 72 = (5 * 14) + 2$

$\gcd(5, 2)$ $\quad 5 = (2 * 2) + 1$

# Euclidean Algorithm



**Goal:** Finding Greatest Common Divisor (GCD)

**Fact 1**: $\gcd(a, 0) = a$
**Fact 2**: $\gcd(a, b) = \gcd(b, r)$, where $r$ is the remainder of dividing $a$ by $b$ $(a > b)$

***Example***

$\gcd(72, 5) \qquad 72 = (5 * 14) + 2$

$\gcd(5, 2) \qquad 5 = (2 * 2) + 1$

$\gcd(2, 1) \qquad 2 = (2 * 1) + 0$

# Euclidean Algorithm



**Goal:** Finding Greatest Common Divisor (GCD)

**Fact 1**: $\gcd(a, 0) = a$
**Fact 2**: $\gcd(a, b) = \gcd(b, r)$, where $r$ is the remainder of dividing $a$ by $b$ ($a > b$)

***Example***

$\gcd(72, 5)$ $\quad 72 = (5 * 14) + 2$

$\gcd(5, 2)$ $\quad 5 = (2 * 2) + 1$

$\gcd(2, 1)$ $\quad 2 = (2 * 1) + 0$

$\gcd(1, 0)$

# Euclidean Algorithm



**Goal:** Finding Greatest Common Divisor (GCD)

**Fact 1**: $\gcd(a, 0) = a$
**Fact 2**: $\gcd(a, b) = \gcd(b, r)$, where $r$ is the remainder of dividing $a$ by $b$ $(a > b)$

***Example***

$\gcd(72, 5)$ $\quad 72 = (5 * 14) + 2$

$\gcd(5, 2)$ $\quad 5 = (2 * 2) + 1$

$\gcd(2, 1)$ $\quad 2 = (2 * 1) + 0$

$\gcd(1, 0) = 1$

# Extended Euclidean Algorithm



- **Goal:** Computing integers $x$ and $y$ s.t.

$$ax + by = \gcd(a, b)$$

Choose $e$ s.t.
- $1 < e < \phi(n)$ **and**
- $\gcd(\phi(n), e) = 1$

Choose $d$ s.t.
- $1 < d < \phi(n)$ **and**
- $(ed \bmod \phi(n)) = 1$

$p = 7, q = 13$
$n = 91, \phi(n) = 72$

$e = 5$
$d = 29$

# Extended Euclidean Algorithm



- **Goal:** Computing integers $x$ and $y$ s.t.

$$a x + b y = \gcd(a, b)$$

$$e d + \phi(n)(-k) = \gcd(\phi(n), e) = 1$$

Choose $e$ s.t.
- $1 < e < \phi(n)$ **and**
- $\gcd(\phi(n), e) = 1$

Choose $d$ s.t.
- $1 < d < \phi(n)$ **and**
- $(ed \bmod \phi(n)) = 1$

$p = 7, q = 13$
$n = 91, \phi(n) = 72$

$e = 5$
$d = 29$

# Extended Euclidean Algorithm



- **Goal:** Computing integers $x$ and $y$ s.t.

$$ax + by = \gcd(a, b)$$

$$ed + \phi(n)(-k) = \gcd(\phi(n), e) = 1$$

$$(e = 5, \phi(n) = 72)$$

We can find the value $d$! ☺

# Extended Euclidean Algorithm

- **Goal:** Computing integers $x$ and $y$ s.t.

$$ax + by = \gcd(a, b)$$

$$ed + \phi(n)(-k) = \gcd(\phi(n), e) = 1$$

$$(e = 5, \phi(n) = 72)$$

***Example***

$\gcd(72, 5)$ $\quad 72 = (5 * 14) + 2$

$\gcd(5, 2)$ $\quad 5 = (2 * 2) + 1 \Rightarrow 5 - (2 * 2) = 1$

$\gcd(2, 1)$ $\quad 2 = (2 * 1) + 0$

$\gcd(1, 0) = 1$

$x = 1$
$y = -2$

# Extended Euclidean Algorithm



- **Goal:** Computing integers $x$ and $y$ s.t.

$$ax + by = \gcd(a, b)$$

$$ed + \phi(n)(-k) = \gcd(\phi(n), e) = 1$$

$$(e = 5, \phi(n) = 72)$$

***Example***

$$2 = 72 - (5 * 14)$$

$\gcd(72, 5)$ $\quad 72 = (5 * 14) + 2$

$\gcd(5, 2)$ $\quad 5 = (2 * 2) + 1 \Rightarrow 5 - (2 * 2) = 1$

$\gcd(2, 1)$ $\quad 2 = (2 * 1) + 0$

$\gcd(1, 0) = 1$

# Extended Euclidean Algorithm



- **Goal:** Computing integers $x$ and $y$ s.t.

$$ax + by = \gcd(a, b)$$

$$ed + \phi(n)(-k) = \gcd(\phi(n), e) = 1$$

$$(e = 5, \phi(n) = 72)$$

***Example***

$\gcd(72, 5)$ $\quad 72 = (5 * 14) + 2 \Rightarrow 5 - ((72 - 5 * 14) * 2) = 1$

$\gcd(5, 2)$ $\quad 5 = (2 * 2) + 1 \Rightarrow 5 - (2 * 2) = 1$

$\gcd(2, 1)$ $\quad 2 = (2 * 1) + 0$

$\gcd(1, 0) = 1$

# Extended Euclidean Algorithm

- **Goal:** Computing integers $x$ and $y$ s.t.

$$ax + by = \gcd(a, b)$$

$$ed + \phi(n)(-k) = \gcd(\phi(n), e) = 1$$

$$(e = 5, \phi(n) = 72)$$

***Example***

$\gcd(72, 5)$ $\quad 72 = (5 * 14) + 2 \Rightarrow 5 * 29 + 72(-2) = 1$

$\gcd(5, 2)$ $\quad 5 = (2 * 2) + 1 \Rightarrow 5 - (2 * 2) = 1$

$\gcd(2, 1)$ $\quad 2 = (2 * 1) + 0$

$\gcd(1, 0) = 1$

$x = \mathrm{d} = 29$

$\mathrm{y} = -\mathrm{k} = -2$

# Logic Flow



$$
\begin{aligned}
&r_1 \leftarrow a; \quad r_2 \leftarrow b; && \text{(Initialization)}\\
&\text{while } (r_2 > 0)\\
&\{\\
&\quad q \leftarrow r_1 / r_2;\\
&\quad r \leftarrow r_1 - q \times r_2;\\
&\quad r_1 \leftarrow r_2; \; r_2 \leftarrow r;\\
&\}\\
&\gcd(a, b) \leftarrow r_1
\end{aligned}
$$

Euclidean Algorithm

$$
\begin{aligned}
&r_1 \leftarrow a; \quad r_2 \leftarrow b;\\
&s_1 \leftarrow 1; \quad s_2 \leftarrow 0; && \text{(Initialization)}\\
&t_1 \leftarrow 0; \quad t_2 \leftarrow 1;\\
&\text{while } (r_2 > 0)\\
&\{\\
&\quad q \leftarrow r_1 / r_2;\\
&\quad r \leftarrow r_1 - q \times r_2;\\
&\quad r_1 \leftarrow r_2; \; r_2 \leftarrow r; && \text{(Updating } r\text{'s)}\\
&\quad s \leftarrow s_1 - q \times s_2;\\
&\quad s_1 \leftarrow s_2; \; s_2 \leftarrow s; && \text{(Updating } s\text{'s)}\\
&\quad t \leftarrow t_1 - q \times t_2;\\
&\quad t_1 \leftarrow t_2; \; t_2 \leftarrow t; && \text{(Updating } t\text{'s)}\\
&\}\\
&\gcd(a, b) \leftarrow r_1; \; s \leftarrow s_1; \; t \leftarrow t_1
\end{aligned}
$$

Extended Euclidean Algorithm

# Exercise

- Given $a = 161$ and $b = 28$, find $\gcd(a, b)$ and the values of $x$ and $y$ such that $ax + by = \gcd(a, b)$

# RSA Algorithm (1): Key Generation

How to find $d$?
→ Extended Euclidean Algorithm!

Choose $d$ s.t.
- $1 < d < \phi(n)$ **and**
- $(ed \bmod \phi(n)) = 1$

***Public place***

$p = 7, q = 13$
$n = 91, \phi(n) = 72$

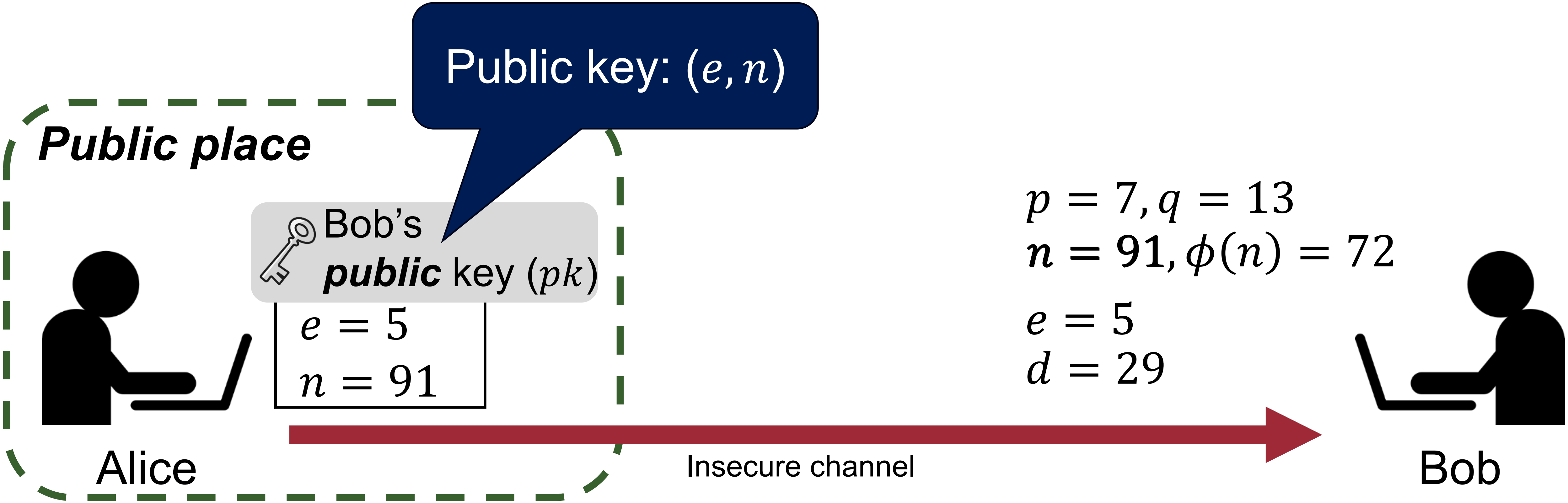
$e = 5$
$d = 29$

Alice

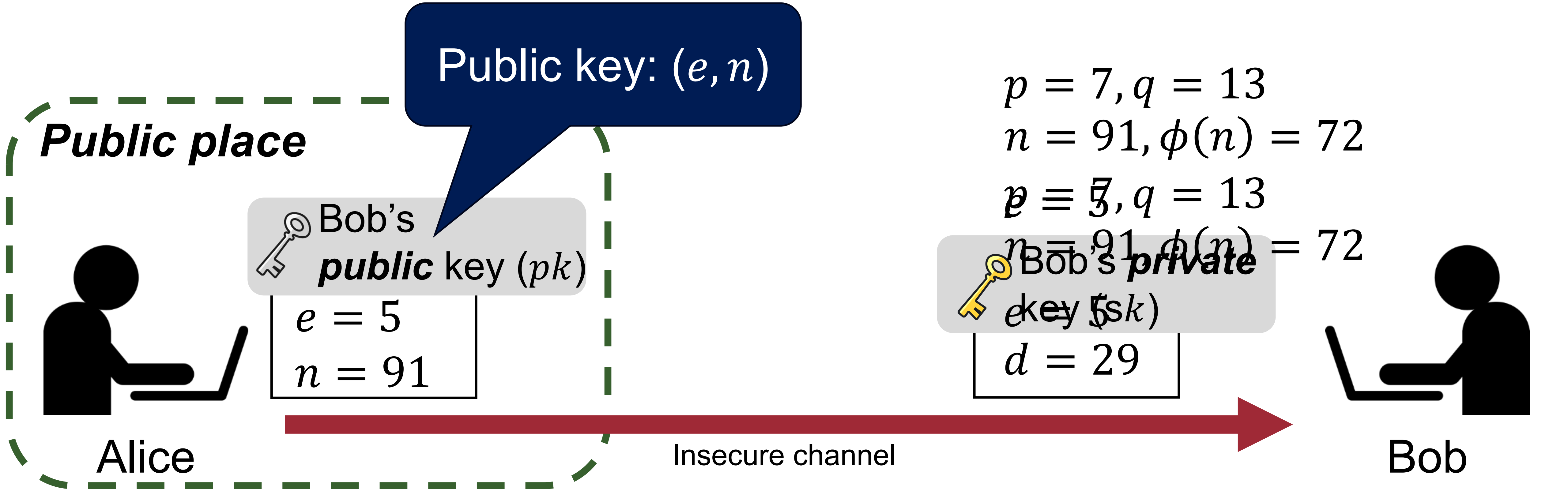
Insecure channel

Bob

# RSA Algorithm (1): Key Generation

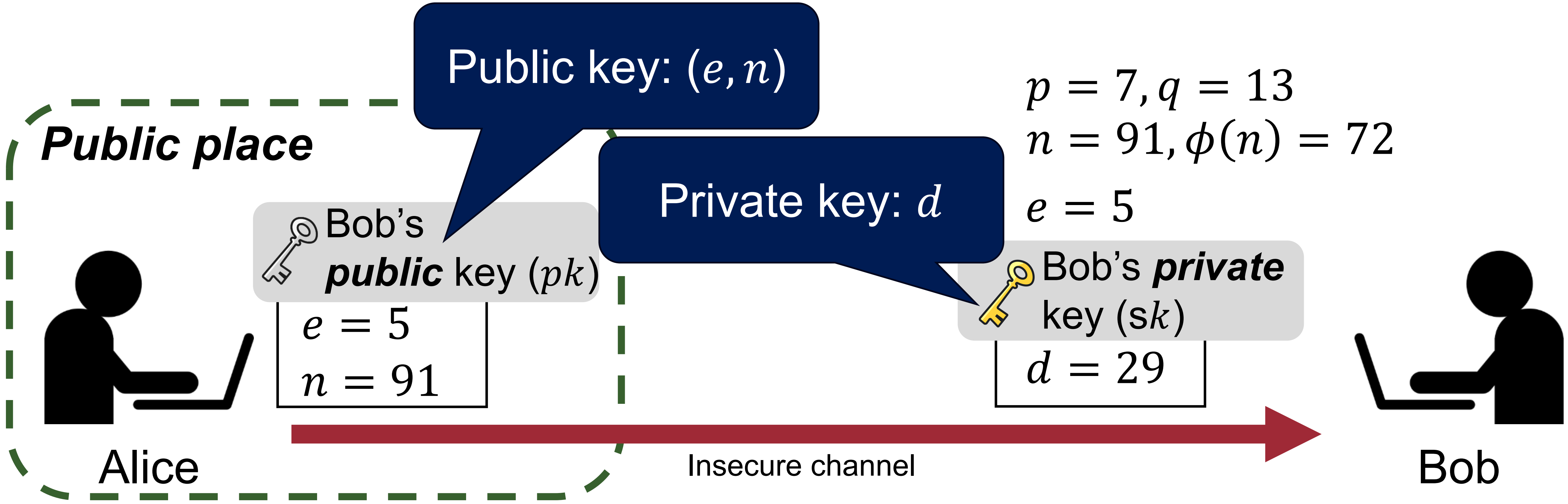


# RSA Algorithm (1): Key Generation



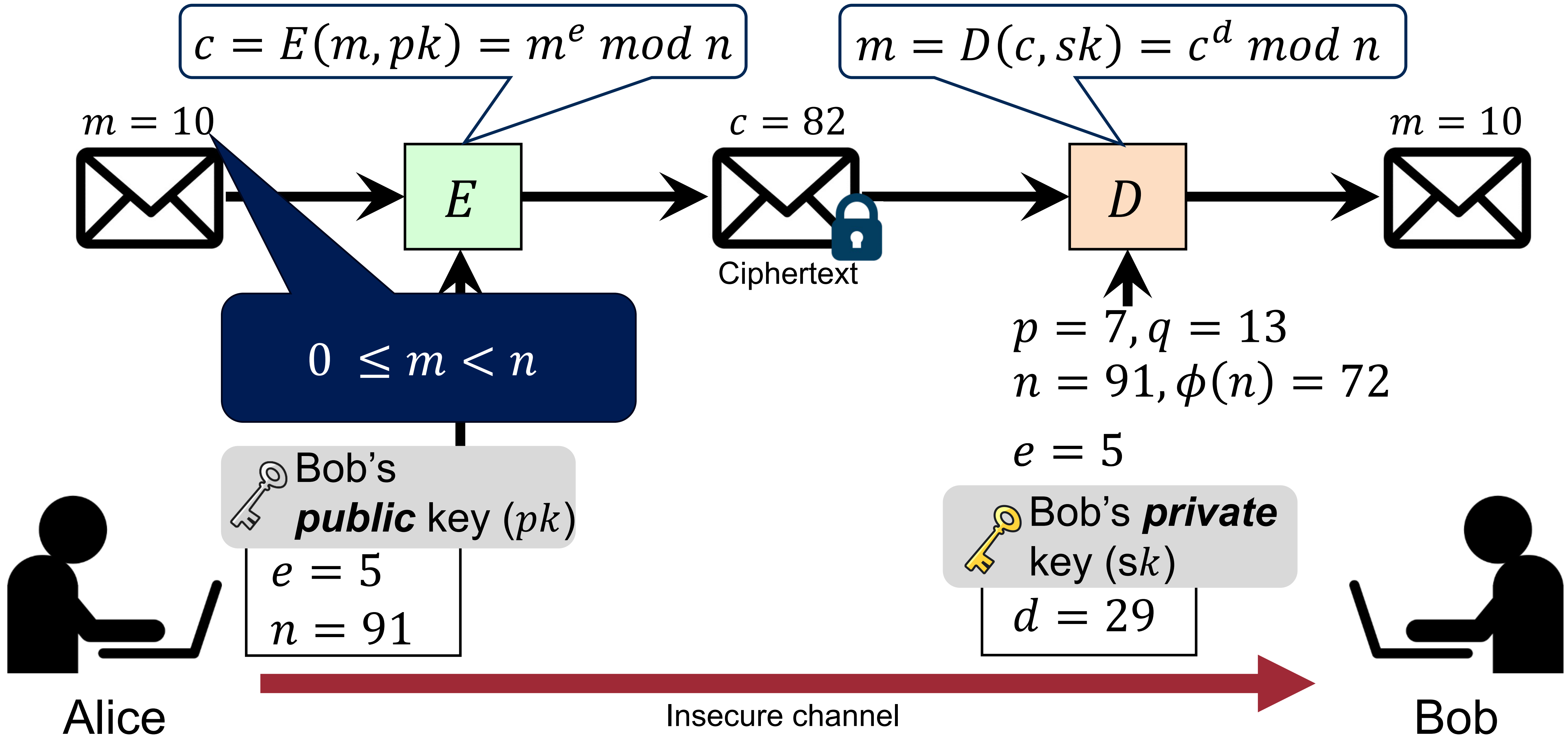
$p = 7, q = 13$
$n = 91, \phi(n) = 72$

# RSA Algorithm (1): Key Generation

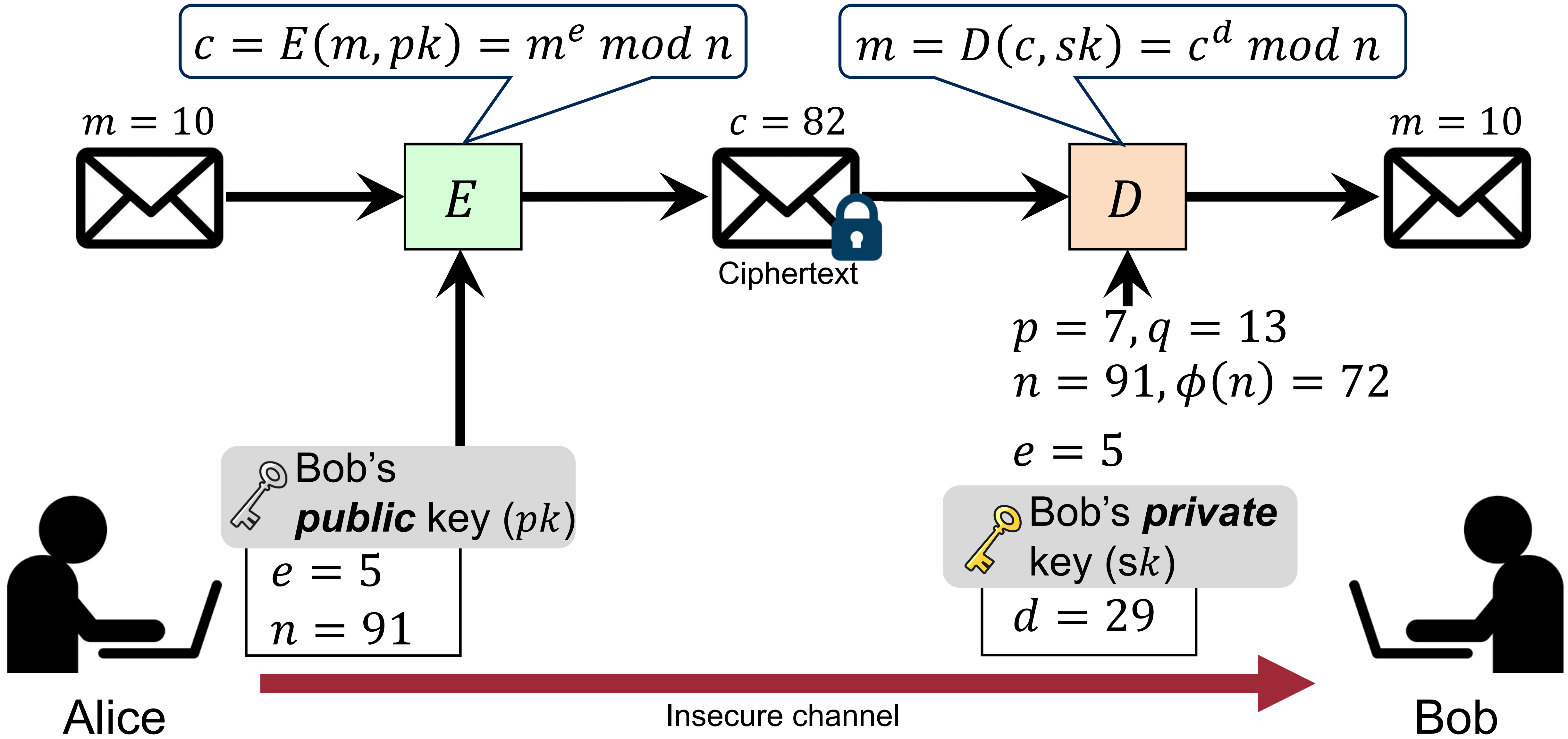


# RSA Algorithm (2): Encryption and Decryption





# RSA Algorithm (2): Encryption and Decryption



# Correctness of the RSA Algorithm

$c = E(m, pk) = m^e \bmod n$

$m = D(c, sk) = c^d \bmod n$

**Correctness**: $m = (m^e \bmod n)^d \bmod n$
$= m^{ed} \bmod n$

**Theorem:**

$((X \bmod p)^k \bmod p) = (X^k \bmod p)$

# Correctness of the RSA Algorithm



$\boxed{c = E(m, pk) = m^e \bmod n}$ $\boxed{m = D(c, sk) = c^d \bmod n}$

**Theorem:**

$((X \bmod p)^k \bmod p) = (X^k \bmod p)$

**Correctness**: $m = (m^e \bmod n)^d \bmod n$

$= m^{ed} \bmod n$

$= m^{1+k\cdot\phi(n)} \bmod n$

We choose $d$ s.t.

$(ed \bmod \phi(n)) = 1$

# Correctness of the RSA Algorithm



$c = E(m, pk) = m^e \bmod n$

$m = D(c, sk) = c^d \bmod n$

**Theorem:**

$((X \bmod p)^k \bmod p) = (X^k \bmod p)$

**Correctness**: $m = (m^e \bmod n)^d \bmod n$

$= m^{ed} \bmod n$

$= m^{1+k\cdot\phi(n)} \bmod n$

$= m \cdot \left(m^{\phi(n)}\right)^k \bmod n$

$= m \bmod n$

$= m$

We choose $d$ s.t. $(ed \bmod \phi(n)) = 1$

**Euler's Theorem:**

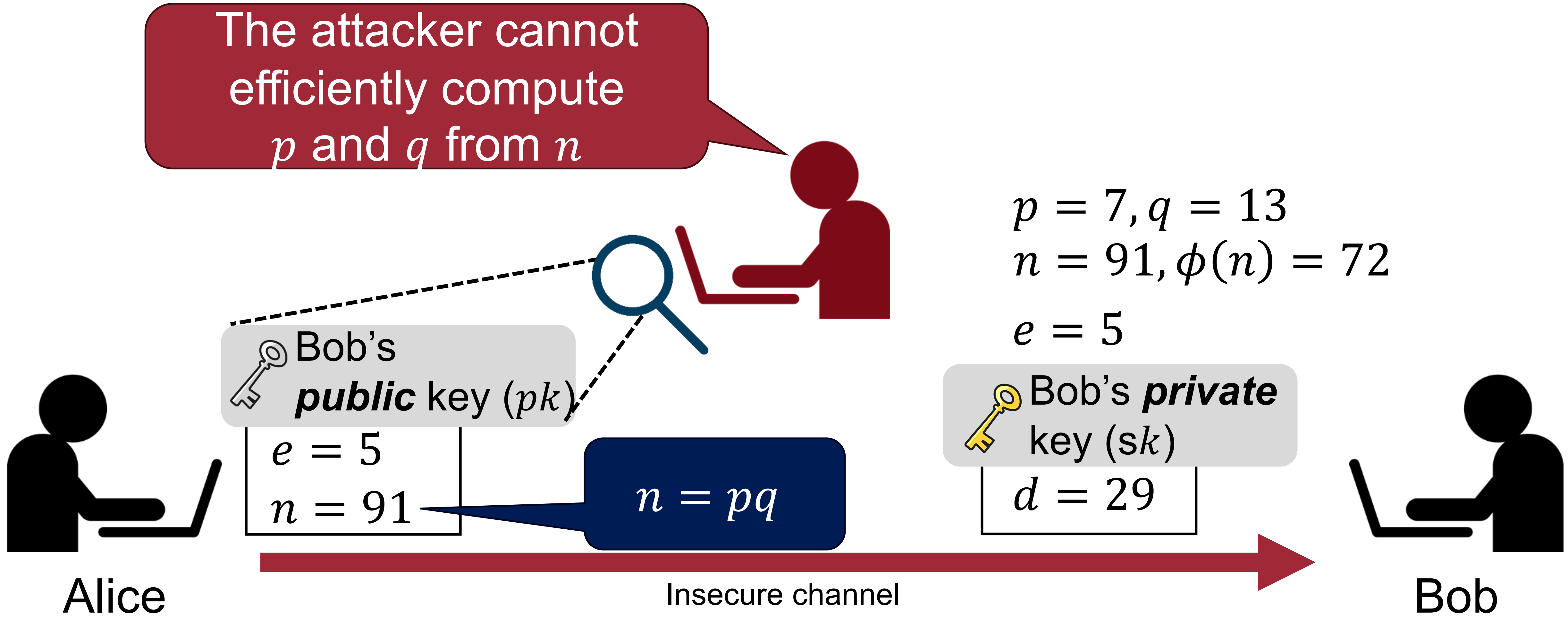
$\left(X^{\phi(n)} \bmod n\right) = 1$ where $\gcd(X, n) = 1$

Also, refer to *Fermat's little theorem* ☺

# Security of the RSA Algorithm

$$c = E(m, pk) = m^e \bmod n$$

$$m = D(c, sk) = c^d \bmod n$$

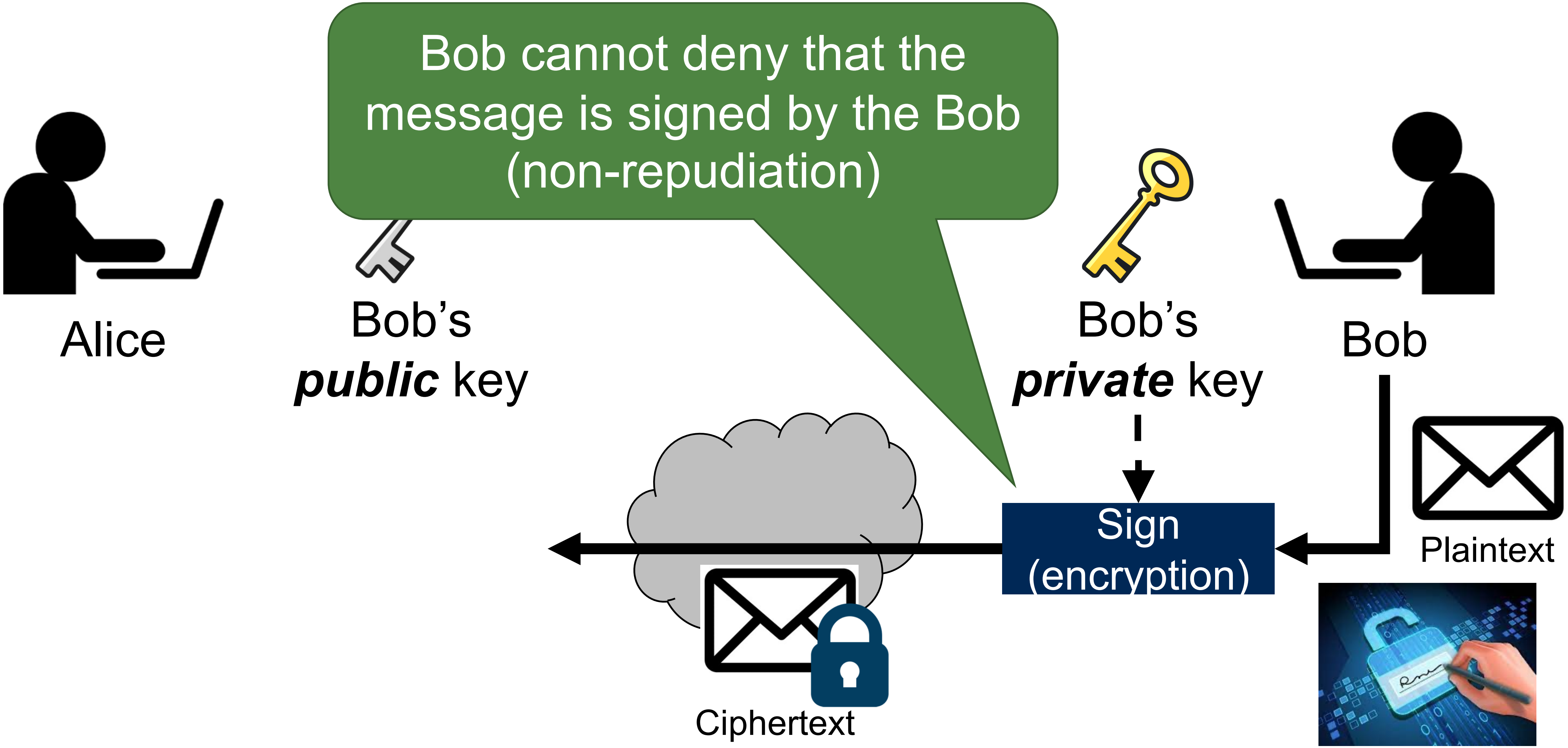


# Comparison with Symmetric-Key Cryptography

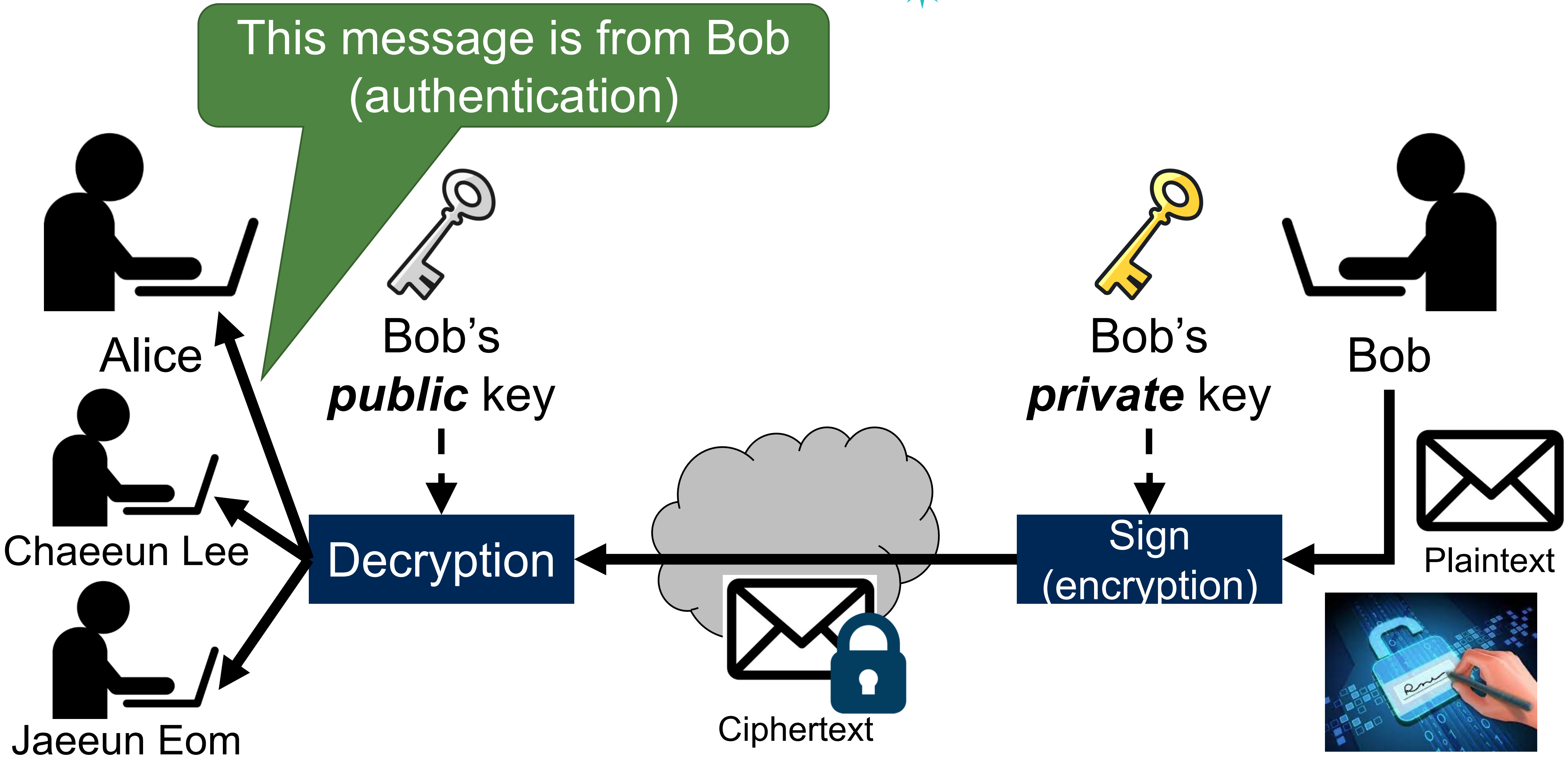


- Pros
  - No need to share a secret
  - More applications: Digital sign

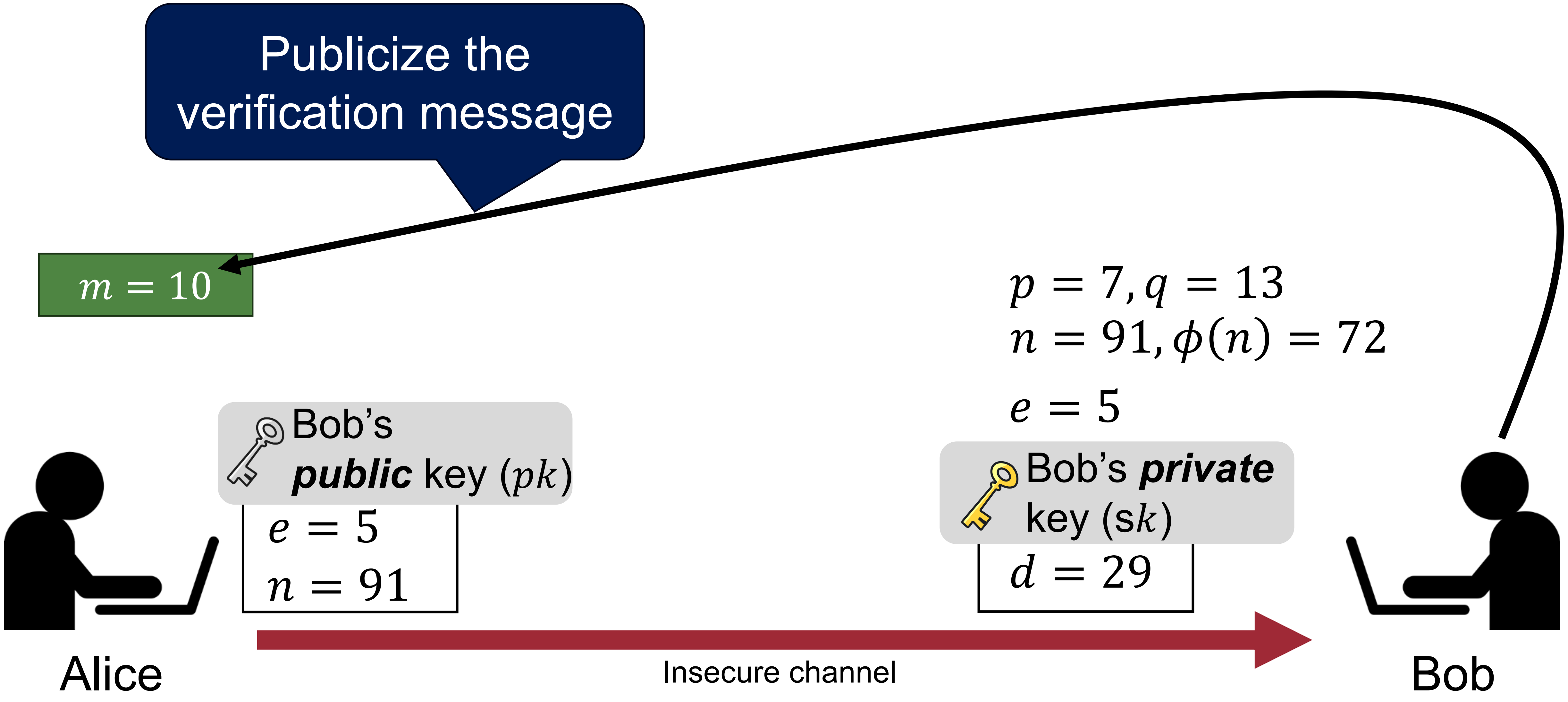
# Digital Signature



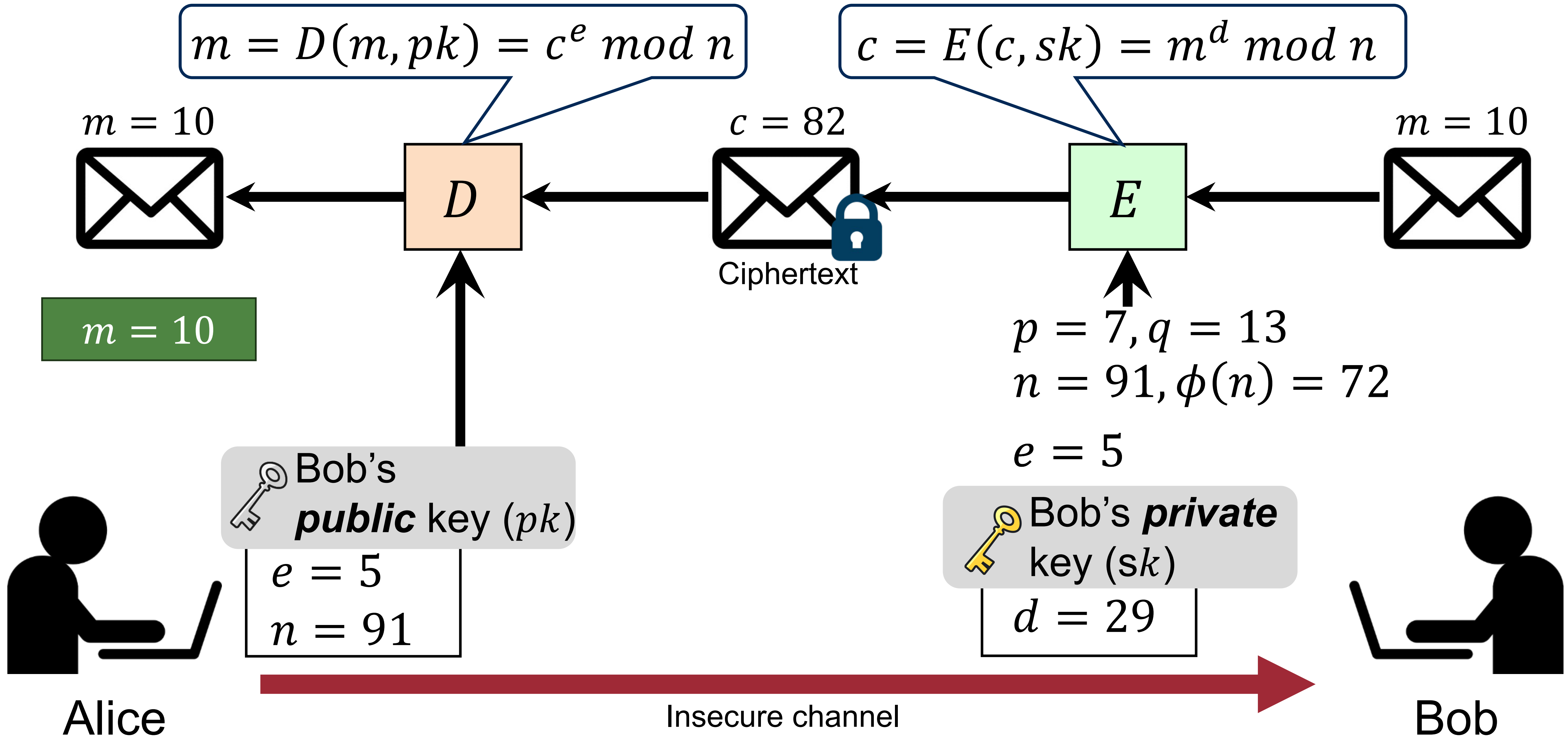
# Digital Signature



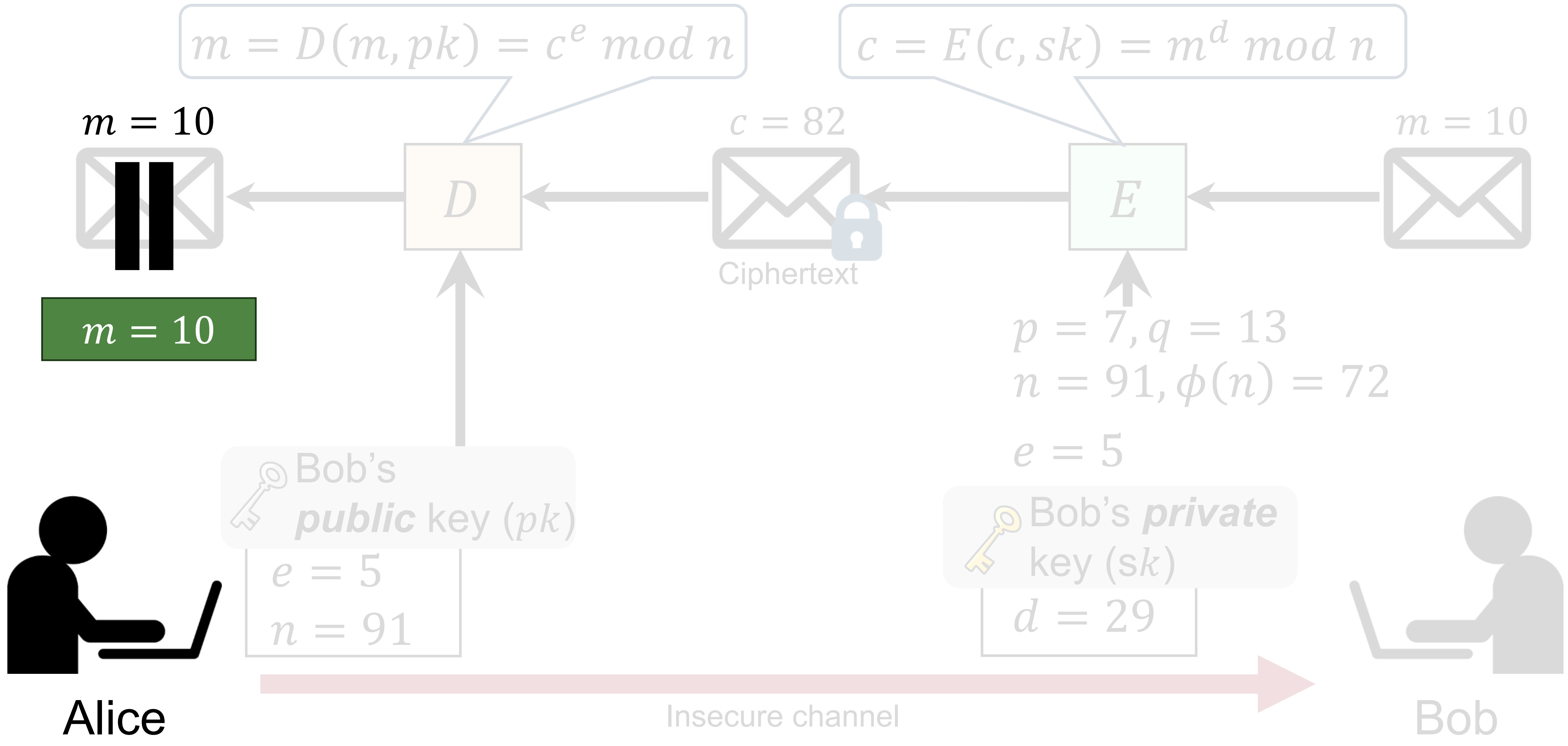
# Digital Signature in Detail (1)



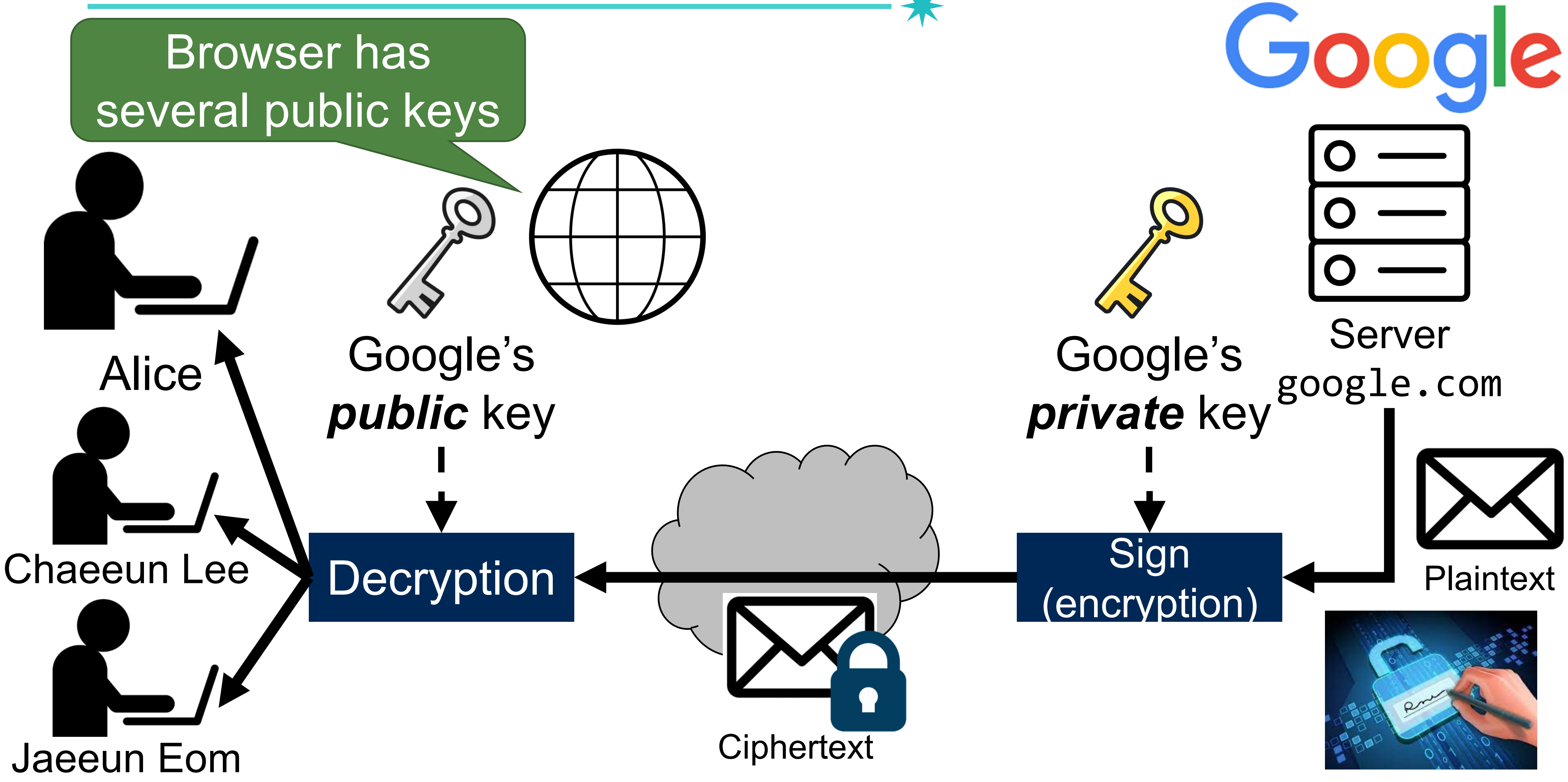
# Digital Signature in Detail (2)



# Digital Signature in Detail (3)

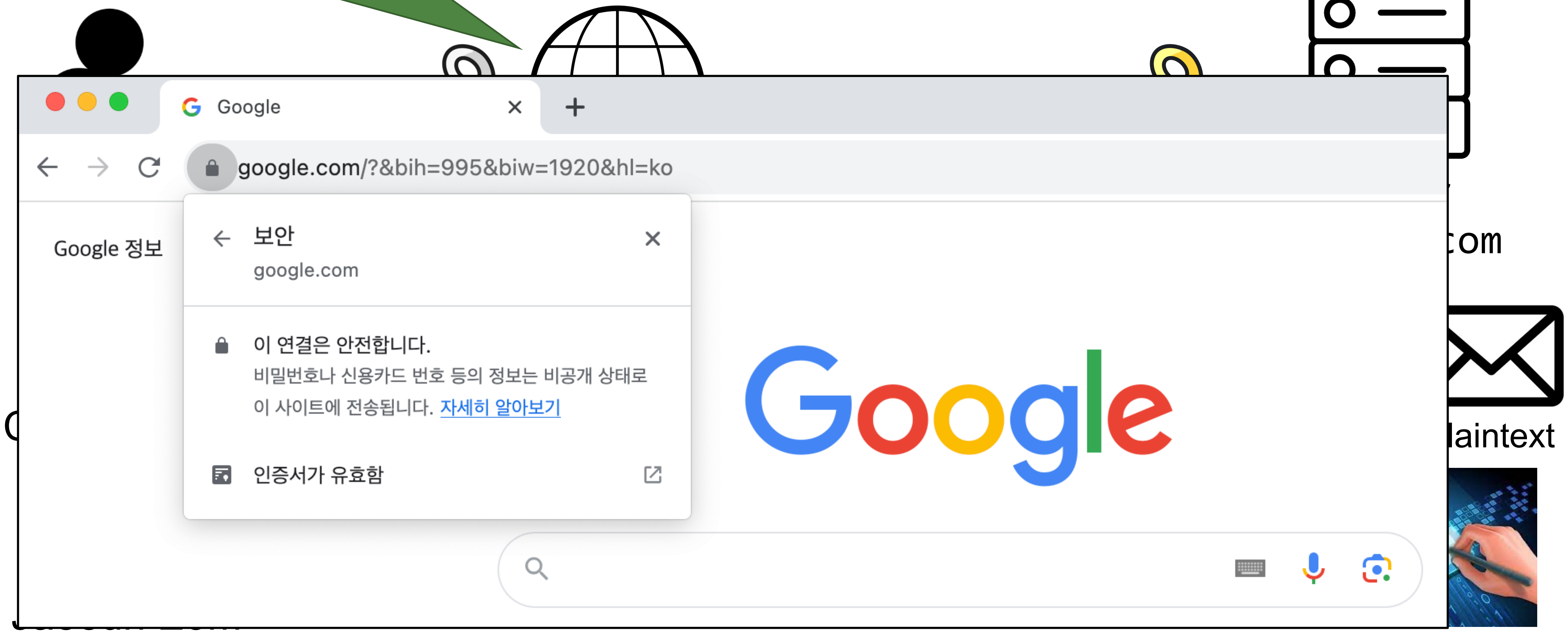


# Application of Digital Signature in HTTPs



# Application of Digital Signature in HTTPs

Browser has several public keys

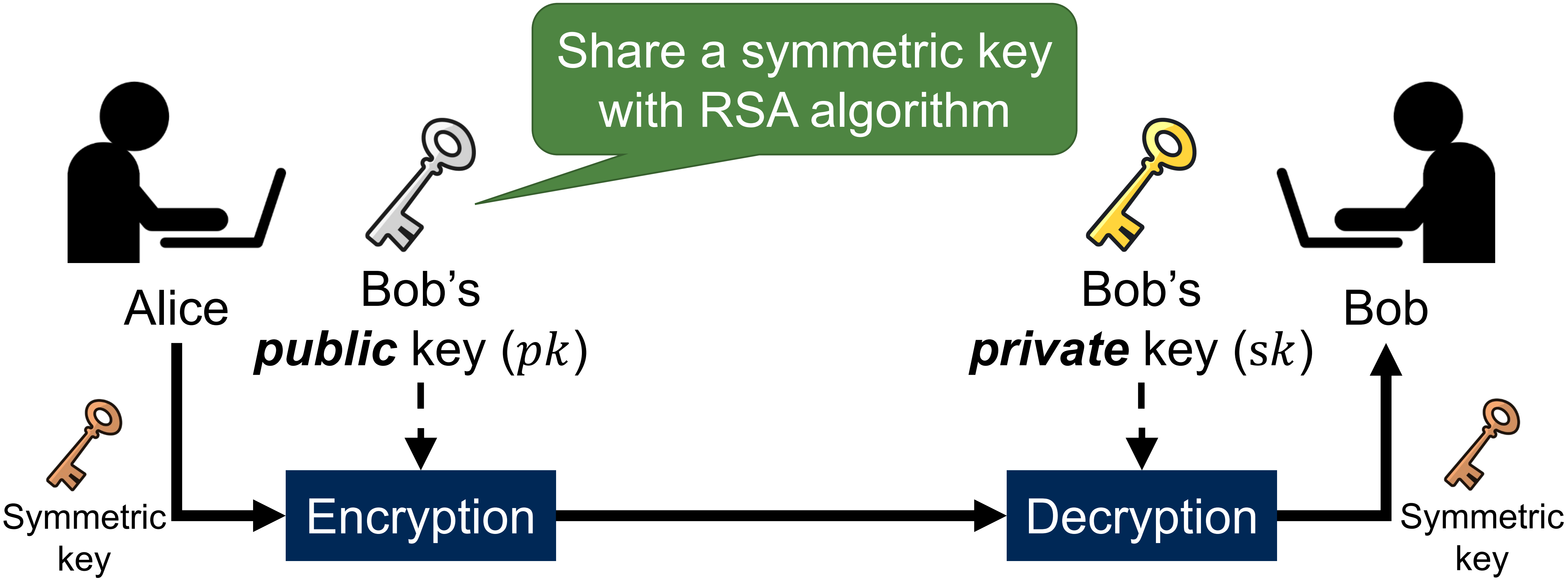


# Comparison with Symmetric-Key Cryptography



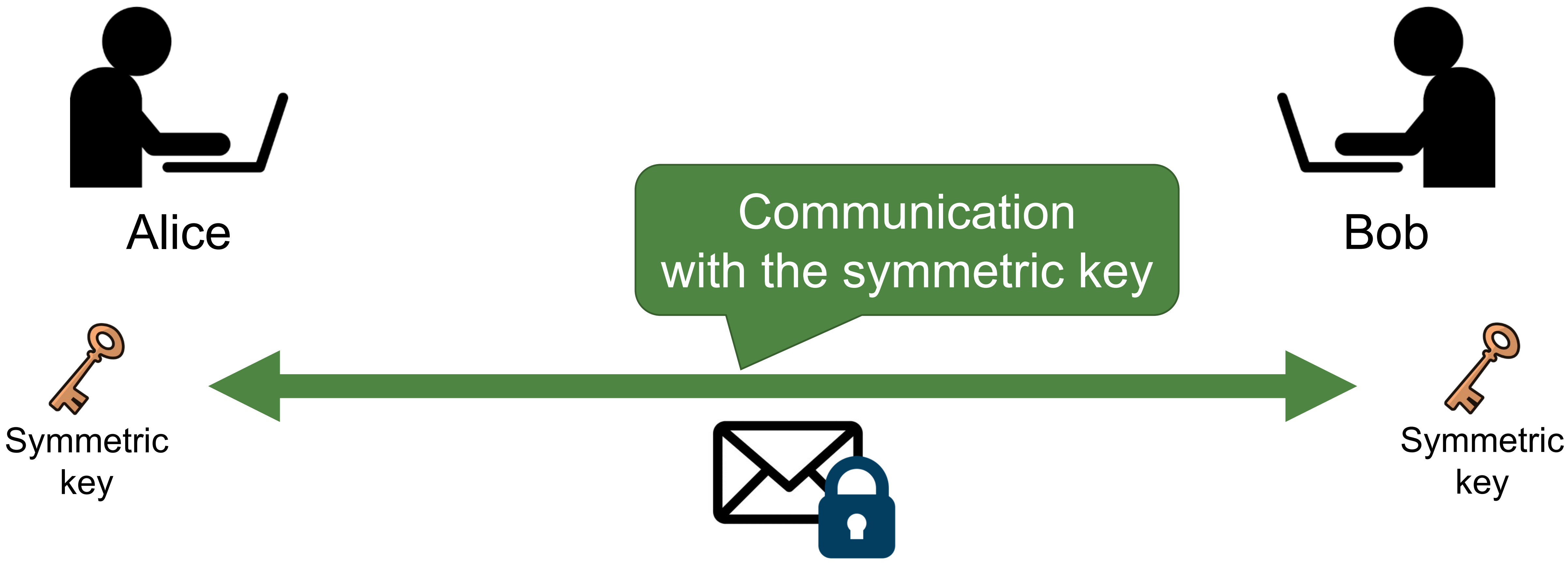
- Pros
  - No need to share a secret
  - Enable multiple senders to communicate privately with a single receiver
  - More applications: Digital sign
- Cons
  - Slower in general: due to the larger key
    - Roughly 2-3 orders of magnitude slower

# In Practice: Combination of Two Schemes



# In Practice: Combination of Two Schemes





# Summary

- Public-key revolution: solve key distribution and maintenance problem
  - Diffie-Hellman key exchange
  - Public-key encryption
  - Digital signature

- (Next lecture) Public key infrastructure, hash, MAC, and homomorphic encryption

# Question?