Public-Key Encryption

CS/ECE 407

Today's objectives

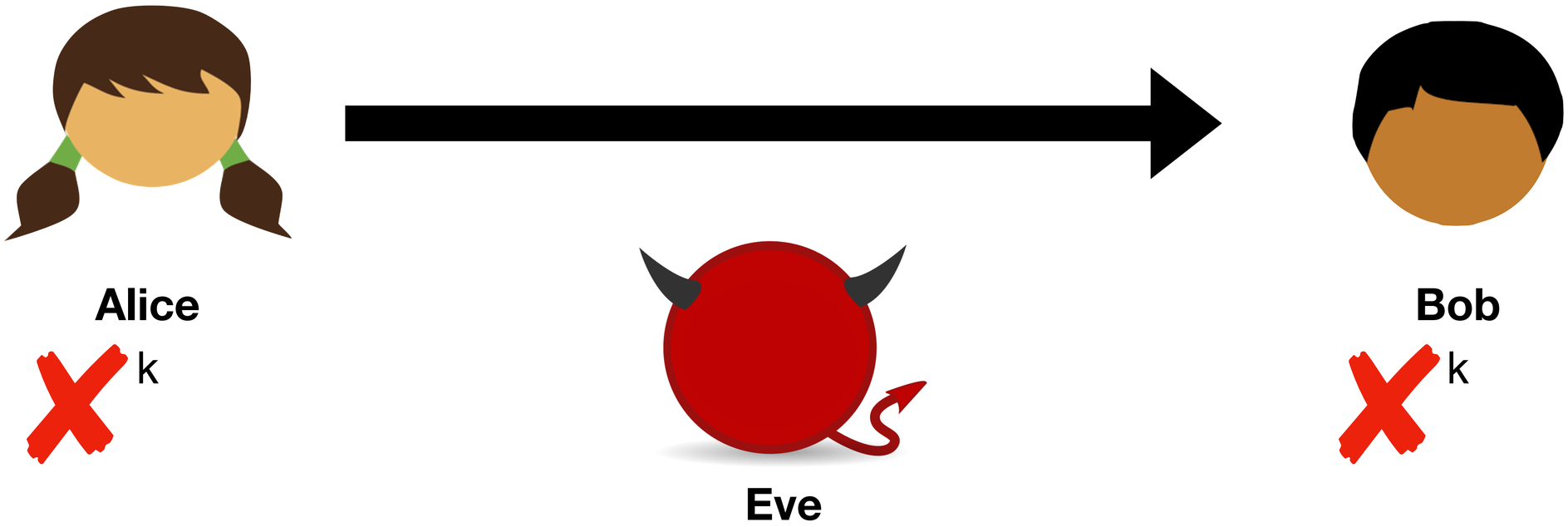
Recall the discrete logarithm/DDH assumption

Recall Diffie Hellman Key Exchange

Define public-key encryption scheme

Define CPA security for public-key schemes

Construct a CPA-secure public-key scheme



Public Key Cryptography:

Can Alice and Bob securely communicate, even if they are speaking for the very first time?

Definition: Group

A **group** is a set S with some operation ★ s.t.

$$x \bigstar (y \bigstar z) = (x \bigstar y) \bigstar z$$

And where there exists some element $\underline{\mathbf{1}}$

$$x \bigstar \underline{\mathbf{1}} = \underline{\mathbf{1}} \bigstar x = x$$

And where for every x there exists some x^{-1} s.t.

$$x \bigstar x^{-1} = \underline{\mathbf{1}}$$

Definition: Group

A **group** is a set S with some operation $\cdot$ s.t.

$$x \cdot (y \cdot z) = (x \cdot y) \cdot z$$

And where there exists some element $\underline{\mathbf{1}}$

$$x \cdot \underline{\mathbf{1}} = \underline{\mathbf{1}} \cdot x = x$$

And where for every x there exists some x^{-1} s.t.

$$x \cdot x^{-1} = \underline{\mathbf{1}}$$

Definition: Group

A **group** is a set S with some operation $\cdot$ s.t.

$$x \cdot (y \cdot z) = (x \cdot y) \cdot z$$

And where there exists some element $\underline{\mathbf{1}}$

$$x \cdot \underline{\mathbf{1}} = \underline{\mathbf{1}} \cdot x = x$$

And where for every x there exists some x^{-1} s.t.

$$x \cdot x^{-1} = \underline{\mathbf{1}}$$

The number of elements in a group is called its **order**

Definition: Cyclic Group

A group G is **cyclic** if there exists an element g s.t. for every element h in G, there exists a number k s.t. $h = g^k$

$$g^k = \underbrace{g \cdot g \cdot g \ldots \cdot g}_{\text{k times}}$$

Decisional Diffie-Hellman Assumption

Let Gen be an algorithm that defines a family of cyclic groups and their generators

We say the DDH problem is hard for that family if the following ensembles are indistinguishable:

$$\left\{ (\mathbb{G}, g, g^x, g^y, g^z) \;\middle|\; \begin{array}{l} (\mathbb{G}, g) \leftarrow \mathrm{Gen}(\lambda) \\ x \leftarrow \{0,\ldots |\mathbb{G}| - 1\} \\ y \leftarrow \{0,\ldots |\mathbb{G}| - 1\} \\ z \leftarrow \{0,\ldots |\mathbb{G}| - 1\} \end{array} \right\} \approx \left\{ (\mathbb{G}, g, g^x, g^y, g^{xy}) \;\middle|\; \begin{array}{l} (\mathbb{G}, g) \leftarrow \mathrm{Gen}(\lambda) \\ x \leftarrow \{0,\ldots |\mathbb{G}| - 1\} \\ y \leftarrow \{0,\ldots |\mathbb{G}| - 1\} \end{array} \right\}$$

Diffie Hellman Key Exchange

Under DDH, g^{xy} appears to be a uniform group element

Public Key Encryption

A public-key encryption scheme is a triple `(KeyGen, Enc, Dec)`

`KeyGen()` outputs a **key pair** `(pk, sk)`

$$\texttt{Enc(pk, m)} \rightarrow \texttt{c}$$

$$\texttt{Dec(sk, c)} \rightarrow \texttt{m}$$

Correctness:

Public Key Encryption

A public-key encryption scheme is a triple $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$

$\mathsf{KeyGen}()$ outputs a **key pair** $(\mathsf{pk}, \mathsf{sk})$

$$\mathsf{Enc}(\mathsf{pk}, \mathsf{m}) \rightarrow \mathsf{c}$$

$$\mathsf{Dec}(\mathsf{sk}, \mathsf{c}) \rightarrow \mathsf{m}$$

Correctness: For any $(\mathsf{pk},\mathsf{sk}) \leftarrow \mathsf{Gen}()$ pairs and for all m

$$\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, \mathsf{m})) = \mathsf{m}$$

Symmetric-key (KeyGen, Enc, Dec) has **CPA security** if:

```
k ← K
encrypt(m0, m1):
  ct ← Enc(k, m0)
  return ct
```

≈

```
k ← K
encrypt(m0, m1):
  ct ← Enc(k, m1)
  return ct
```

Public key encryption scheme
(Gen, Enc, Dec) has **CPA security** if:

```
(pk, sk) ← KeyGen()

key(): return pk
encrypt(m0, m1):
  ct ← Enc(pk, m0)
  return ct
```

≈

```
(pk, sk) ← KeyGen()

key(): return pk
encrypt(m0, m1):
  ct ← Enc(pk, m1)
  return ct
```

ElGamal Public Key Encryption

Let g be the generator of some cyclic group G of order q

```
KeyGen():
  sk ← Z_q
  pk = g^sk
  return (pk, sk)
```

```
Enc(pk, m ∈ G):
  b ← Z_q
  s = pk^b
  c_1 = g^b
  c_2 = m·s
  return (c_1, c_2)
```

```
Dec(sk, (c_1, c_2)):
  s = c_1^sk
  return c_2·s^-1
```

Public key encryption scheme (Gen, Enc, Dec) has **CPA security** if:

```
(pk, sk) ← KeyGen()

key():
  return pk

encrypt(m0, m1):
  ct ← Enc(pk, m0)
  return ct
```

≈

```
(pk, sk) ← KeyGen()

key():
  return pk

encrypt(m0, m1):
  ct ← Enc(pk, m1)
  return ct
```

DDH  **ElGamal is CPA secure**

Public key encryption scheme (Gen, Enc, Dec) has **one-time secrecy** if:

```
(pk, sk) ← KeyGen()
count ← 0

key():
  return pk

eavesdrop(m0, m1):
  if count > 0:
    return error
  count ← count + 1
  ct ← Enc(pk, m0)
  return ct
```

≈

```
(pk, sk) ← KeyGen()
count ← 0

key():
  return pk

eavesdrop(m0, m1):
  if count > 0:
    return error
  count ← count + 1
  ct ← Enc(pk, m1)
  return ct
```

Public key encryption scheme (Simpler, weaker than one-time secrecy)

$$\left\{ (\mathrm{pk}, c) \;\middle|\; \begin{array}{l} (\mathrm{pk}, \mathrm{sk}) \leftarrow \mathrm{KeyGen}(1^\lambda) \\ c \leftarrow \mathrm{Enc}(\mathrm{pk}, m_0) \end{array} \right\} \approx \left\{ (\mathrm{pk}, c) \;\middle|\; \begin{array}{l} (\mathrm{pk}, \mathrm{sk}) \leftarrow \mathrm{KeyGen}(1^\lambda) \\ c \leftarrow \mathrm{Enc}(\mathrm{pk}, m_1) \end{array} \right\}$$

$$\left\{ (\mathrm{pk}, c) \;\middle|\; \begin{array}{l} (\mathrm{pk}, \mathrm{sk}) \leftarrow \mathrm{KeyGen}(1^\lambda) \\ c \leftarrow \mathrm{Enc}(\mathrm{pk}, m_0) \end{array} \right\}$$

$$\left\{ (\mathrm{pk}, c) \;\middle|\; \begin{array}{l} (\mathrm{pk}, \mathrm{sk}) \leftarrow \mathrm{KeyGen}(1^\lambda) \\ c \leftarrow \mathrm{Enc}(\mathrm{pk}, m_0) \end{array} \right\}$$

$$\equiv$$

$$\left\{ (\mathrm{pk}, (c_1, c_2)) \;\middle|\; \begin{array}{l} \mathrm{sk}, b \leftarrow \{0,\ldots,q-1\} \\ \mathrm{pk} = g^{\mathrm{sk}} \\ c_1 = g^b \\ \mathrm{mask} = g^{b \cdot \mathrm{sk}} \\ c_2 = m_0 \cdot \mathrm{mask} \end{array} \right\}$$

$$\left\{ (\mathrm{pk}, c) \;\middle|\; \begin{array}{l} (\mathrm{pk}, \mathrm{sk}) \leftarrow \mathrm{KeyGen}(1^\lambda) \\ c \leftarrow \mathrm{Enc}(\mathrm{pk}, m_0) \end{array} \right\}$$

$$\equiv$$

$$\left\{ (\mathrm{pk}, (c_1, c_2)) \;\middle|\; \begin{array}{l} \mathrm{sk}, b \leftarrow \{0,\ldots,q-1\} \\ \mathrm{pk} = g^{\mathrm{sk}} \\ c_1 = g^b \\ \mathrm{mask} = g^{b \cdot \mathrm{sk}} \\ c_2 = m_0 \cdot \mathrm{mask} \end{array} \right\} \approx \left\{ (\mathrm{pk}, (c_1, c_2)) \;\middle|\; \begin{array}{l} \mathrm{sk}, b, r \leftarrow \{0,\ldots,q-1\} \\ \mathrm{pk} = g^{\mathrm{sk}} \\ c_1 = g^b \\ \mathrm{mask} \leftarrow g^r \\ c_2 = m_0 \cdot \mathrm{mask} \end{array} \right\}$$

DDH

$$\left\{ (\mathrm{pk}, c) \;\middle|\; \begin{array}{l} (\mathrm{pk}, \mathrm{sk}) \leftarrow \mathrm{KeyGen}(1^\lambda) \\ c \leftarrow \mathrm{Enc}(\mathrm{pk}, m_0) \end{array} \right\}$$

$$\left\{ (\mathrm{pk}, (c_1, c_2)) \;\middle|\; \begin{array}{l} \mathrm{sk}, b, r \leftarrow \{0,\ldots,q-1\} \\ \mathrm{pk} = g^{\mathrm{sk}} \\ c_1 = g^b \\ c_2 \leftarrow g^r \end{array} \right\}$$

$\equiv$

one-time pad $\equiv$

$$\left\{ (\mathrm{pk}, (c_1, c_2)) \;\middle|\; \begin{array}{l} \mathrm{sk}, b \leftarrow \{0,\ldots,q-1\} \\ \mathrm{pk} = g^{\mathrm{sk}} \\ c_1 = g^b \\ \mathrm{mask} = g^{b \cdot \mathrm{sk}} \\ c_2 = m_0 \cdot \mathrm{mask} \end{array} \right\}$$

$\approx$

DDH

$$\left\{ (\mathrm{pk}, (c_1, c_2)) \;\middle|\; \begin{array}{l} \mathrm{sk}, b, r \leftarrow \{0,\ldots,q-1\} \\ \mathrm{pk} = g^{\mathrm{sk}} \\ c_1 = g^b \\ \mathrm{mask} \leftarrow g^r \\ c_2 = m_0 \cdot \mathrm{mask} \end{array} \right\}$$

$$\left\{ (\mathrm{pk}, c) \;\middle|\; \begin{array}{l} (\mathrm{pk}, \mathrm{sk}) \leftarrow \mathrm{KeyGen}(1^\lambda) \\ c \leftarrow \mathrm{Enc}(\mathrm{pk}, m_0) \end{array} \right\} \approx \left\{ (\mathrm{pk}, c) \;\middle|\; \begin{array}{l} (\mathrm{pk}, \mathrm{sk}) \leftarrow \mathrm{KeyGen}(1^\lambda) \\ c \leftarrow \mathrm{Enc}(\mathrm{pk}, m_0) \end{array} \right\}$$

Public key encryption scheme (Gen, Enc, Dec) has **CPA security** if:

```
(pk, sk) ← KeyGen()

key():
  return pk

encrypt(m0, m1):
  ct ← Enc(pk, m0)
  return ct
```

≈

```
(pk, sk) ← KeyGen()

key():
  return pk

encrypt(m0, m1):
  ct ← Enc(pk, m1)
  return ct
```

DDH  **ElGamal is CPA secure**

Public key encryption scheme (Gen, Enc, Dec) has **one-time secrecy** if:

```
(pk, sk) ← KeyGen()
count ← 0

key():
  return pk

eavesdrop(m0, m1):
  if count > 0:
    return error
  count ← count + 1
  ct ← Enc(pk, m0)
  return ct
```

```
(pk, sk) ← KeyGen()
count ← 0

key():
  return pk

eavesdrop(m0, m1):
  if count > 0:
    return error
  count ← count + 1
  ct ← Enc(pk, m1)
  return ct
```

Public key encryption scheme
(Gen, Enc, Dec) has **CPA security** if:

```
(pk, sk) ← KeyGen()

key(): return pk
eavesdrop(m0, m1):
  ct ← Enc(pk, m0)
  return ct
```

≈

```
(pk, sk) ← KeyGen()

key(): return pk
eavesdrop(m0, m1):
  ct ← Enc(pk, m1)
  return ct
```

For public-key schemes,

one-time secrecy $\Longrightarrow$ CPA security

This is **not true** for symmetric-key schemes!!

Hybrid-#bound

```
pk ← key()
count ← 0

key(): return pk

eavesdrop(m0, m1):
  count ← count + 1
  if count < bound:
    return Enc(pk, m1)
  else if count = bound:
    return OTS(m0, m1)
  else:
    return Enc(pk, m0)
```

◇ **OTS.Left/OTS.Right**

Observations

$$\textbf{Hybrid-\#1} \diamond \textbf{OTS.Left} \equiv \textbf{CPA.Left}$$

For adversary issuing q queries:

$$\textbf{Hybrid-\#q} \diamond \textbf{OTS.Right} \equiv \textbf{CPA.Right}$$

$$\textbf{Hybrid-\#q} \diamond \textbf{OTS.Right} \equiv \textbf{Hybrid-\#(q+1)} \diamond \textbf{OTS.Left}$$

Because encryption scheme is OTS-secure

$$\textbf{Hybrid-\#q} \diamond \textbf{OTS.Left} \approx \textbf{Hybrid-\#q} \diamond \textbf{OTS.Right}$$

Today's objectives

Recall the discrete logarithm/DDH assumption

Recall Diffie Hellman Key Exchange

Define public-key encryption scheme

Define CPA security for public-key schemes

Construct a CPA-secure public-key scheme