

Public-Key Cryptography

Public-Key Cryptography

Lecture 7

Public-Key Encryption

Public-Key Cryptography

Lecture 7

Public-Key Encryption

Diffie-Hellman Key-Exchange, El Gamal Encryption

PKE scheme

PKE scheme

- SKE:
 - Syntax
 - KeyGen outputs $K \leftarrow \mathcal{K}$
 - Enc: $\mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$
 - Dec: $\mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$
 - Correctness
 - $\forall K \in \text{Range}(\text{KeyGen}),$
 $\text{Dec}(\text{Enc}(m, K), K) = m$
 - Security (IND-CPA)

Shared/Symmetric-Key
Encryption
(a.k.a. private-key
encryption)

PKE scheme

- SKE:
 - Syntax
 - KeyGen outputs $K \leftarrow \mathcal{K}$
 - Enc: $\mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$
 - Dec: $\mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$
 - Correctness
 - $\forall K \in \text{Range}(\text{KeyGen}),$
 $\text{Dec}(\text{Enc}(m, K), K) = m$
 - Security (IND-CPA)

Shared/Symmetric-Key
Encryption
(a.k.a. private-key
encryption)

PKE scheme

- SKE:

- Syntax

- KeyGen outputs

$$K \leftarrow \mathcal{K}$$

- Enc: $\mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$

- Dec: $\mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$

- Correctness

- $\forall K \in \text{Range}(\text{KeyGen}),$
Dec(Enc(m,K), K) = m

- Security (IND-CPA)

- PKE

Shared/Symmetric-Key
Encryption
(a.k.a. private-key
encryption)

PKE scheme

• SKE:

• Syntax

• KeyGen outputs

$$K \leftarrow \mathcal{K}$$

• Enc: $\mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$

• Dec: $\mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$

• Correctness

• $\forall K \in \text{Range}(\text{KeyGen}),$
Dec(Enc(m,K), K) = m

• Security (IND-CPA)

• PKE

a.k.a. asymmetric-key encryption

Shared/Symmetric-Key
Encryption
(a.k.a. private-key
encryption)

PKE scheme

- SKE:

- Syntax

- KeyGen outputs

$$K \leftarrow \mathcal{K}$$

- Enc: $\mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$

- Dec: $\mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$

- Correctness

- $\forall K \in \text{Range}(\text{KeyGen}),$
Dec(Enc(m,K), K) = m

- Security (IND-CPA)

- PKE

a.k.a. asymmetric-key encryption

- Syntax

PKE scheme

Shared/Symmetric-Key
Encryption
(a.k.a. private-key
encryption)

- SKE:

- Syntax

- KeyGen outputs

$$K \leftarrow \mathcal{K}$$

- Enc: $\mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$

- Dec: $\mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$

- Correctness

- $\forall K \in \text{Range}(\text{KeyGen}),$
 $\text{Dec}(\text{Enc}(m, K), K) = m$

- Security (IND-CPA)

- PKE

a.k.a. asymmetric-key encryption

- Syntax

- KeyGen outputs

$$(PK, SK) \leftarrow \mathcal{PK} \times \mathcal{SK}$$

Shared/Symmetric-Key
Encryption
(a.k.a. private-key
encryption)

PKE scheme

- SKE:

- Syntax

- KeyGen outputs

$$K \leftarrow \mathcal{K}$$

- Enc: $\mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$

- Dec: $\mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$

- Correctness

- $\forall K \in \text{Range}(\text{KeyGen}),$
 $\text{Dec}(\text{Enc}(m, K), K) = m$

- Security (IND-CPA)

- PKE

a.k.a. asymmetric-key encryption

- Syntax

- KeyGen outputs

$$(PK, SK) \leftarrow \mathcal{PK} \times \mathcal{SK}$$

- Enc: $\mathcal{M} \times \mathcal{PK} \rightarrow \mathcal{C}$

Shared/Symmetric-Key
Encryption
(a.k.a. private-key
encryption)

PKE scheme

- SKE:

- Syntax

- KeyGen outputs

$$K \leftarrow \mathcal{K}$$

- Enc: $\mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$

- Dec: $\mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$

- Correctness

- $\forall K \in \text{Range}(\text{KeyGen}),$
 $\text{Dec}(\text{Enc}(m, K), K) = m$

- Security (IND-CPA)

- PKE

a.k.a. asymmetric-key encryption

- Syntax

- KeyGen outputs

$$(PK, SK) \leftarrow \mathcal{PK} \times \mathcal{SK}$$

- Enc: $\mathcal{M} \times \mathcal{PK} \rightarrow \mathcal{C}$

- Dec: $\mathcal{C} \times \mathcal{SK} \rightarrow \mathcal{M}$

Shared/Symmetric-Key
Encryption
(a.k.a. private-key
encryption)

PKE scheme

- SKE:

- Syntax

- KeyGen outputs

$$K \leftarrow \mathcal{K}$$

- Enc: $\mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$

- Dec: $\mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$

- Correctness

- $\forall K \in \text{Range}(\text{KeyGen}),$
 $\text{Dec}(\text{Enc}(m, K), K) = m$

- Security (IND-CPA)

- PKE

a.k.a. asymmetric-key encryption

- Syntax

- KeyGen outputs

$$(PK, SK) \leftarrow \mathcal{PK} \times \mathcal{SK}$$

- Enc: $\mathcal{M} \times \mathcal{PK} \rightarrow \mathcal{C}$

- Dec: $\mathcal{C} \times \mathcal{SK} \rightarrow \mathcal{M}$

- Correctness

Shared/Symmetric-Key
Encryption
(a.k.a. private-key
encryption)

PKE scheme

- SKE:

- Syntax

- KeyGen outputs

$$K \leftarrow \mathcal{K}$$

- Enc: $\mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$

- Dec: $\mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$

- Correctness

- $\forall K \in \text{Range}(\text{KeyGen}),$
 $\text{Dec}(\text{Enc}(m, K), K) = m$

- Security (IND-CPA)

- PKE

a.k.a. asymmetric-key encryption

- Syntax

- KeyGen outputs

$$(PK, SK) \leftarrow \mathcal{PK} \times \mathcal{SK}$$

- Enc: $\mathcal{M} \times \mathcal{PK} \rightarrow \mathcal{C}$

- Dec: $\mathcal{C} \times \mathcal{SK} \rightarrow \mathcal{M}$

- Correctness

- $\forall (PK, SK) \in \text{Range}(\text{KeyGen}),$
 $\text{Dec}(\text{Enc}(m, PK), SK) = m$

Shared/Symmetric-Key
Encryption
(a.k.a. private-key
encryption)

PKE scheme

• SKE:

• Syntax

• KeyGen outputs

$$K \leftarrow \mathcal{K}$$

• Enc: $\mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}$

• Dec: $\mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$

• Correctness

• $\forall K \in \text{Range}(\text{KeyGen}),$ $\text{Dec}(\text{Enc}(m, K), K) = m$

• Security (IND-CPA)

• PKE

a.k.a. asymmetric-key encryption

• Syntax

• KeyGen outputs

$$(PK, SK) \leftarrow \mathcal{PK} \times \mathcal{SK}$$

• Enc: $\mathcal{M} \times \mathcal{PK} \rightarrow \mathcal{C}$

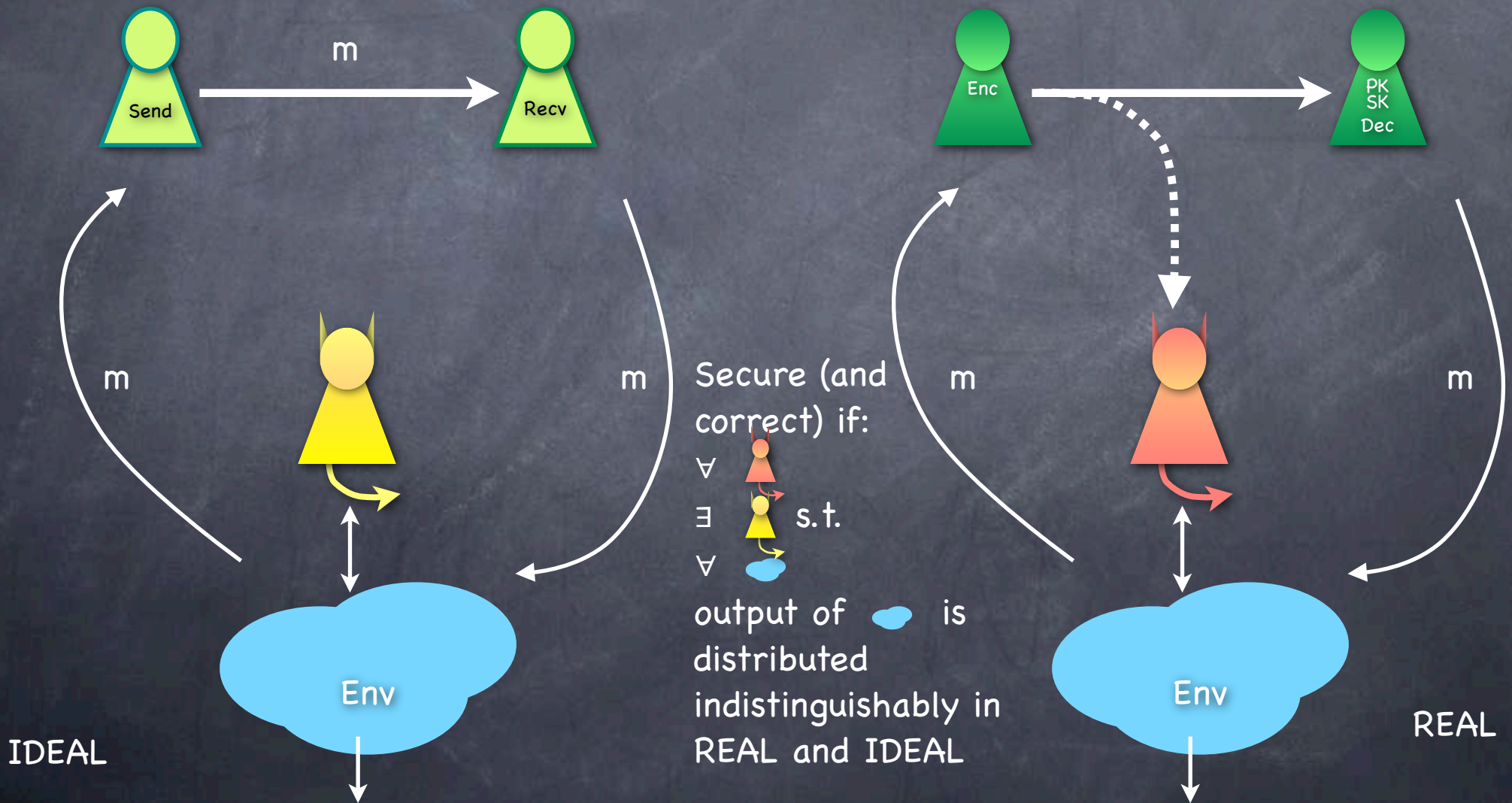
• Dec: $\mathcal{C} \times \mathcal{SK} \rightarrow \mathcal{M}$

• Correctness

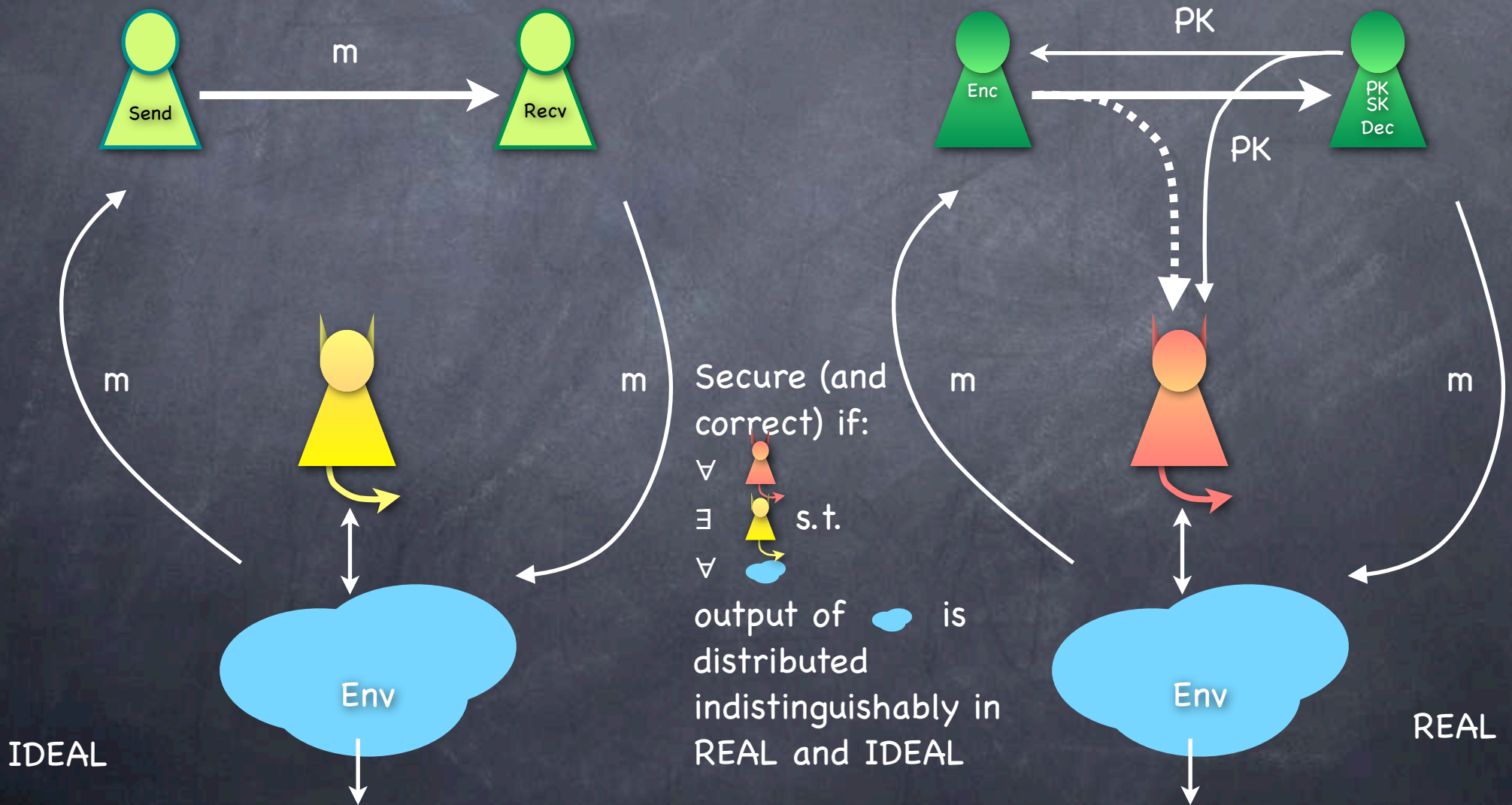
• $\forall (PK, SK) \in \text{Range}(\text{KeyGen}),$ $\text{Dec}(\text{Enc}(m, PK), SK) = m$

• Security (IND-CPA, PKE version)

SIM-CPA (PKE Version)



SIM-CPA (PKE Version)



IND-CPA (PKE version)

- Experiment picks a random bit b . It also runs KeyGen to get a key (PK, SK) . Adv given PK



- Adv sends two messages m_0, m_1 to the experiment
- Expt returns $Enc(m_b, K)$ to the adversary
- Adversary returns a guess b'
- Experiment outputs 1 iff $b' = b$

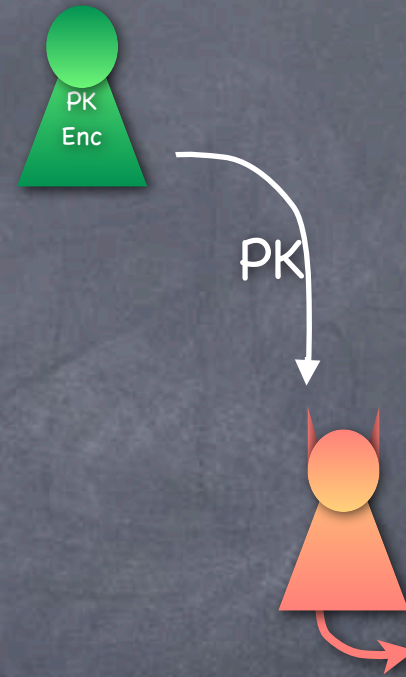


- **IND-CPA secure** if for all PPT adversaries $\Pr[b' = b] - 1/2 \leq \nu(k)$

$b \leftarrow \{0,1\}$

IND-CPA (PKE version)

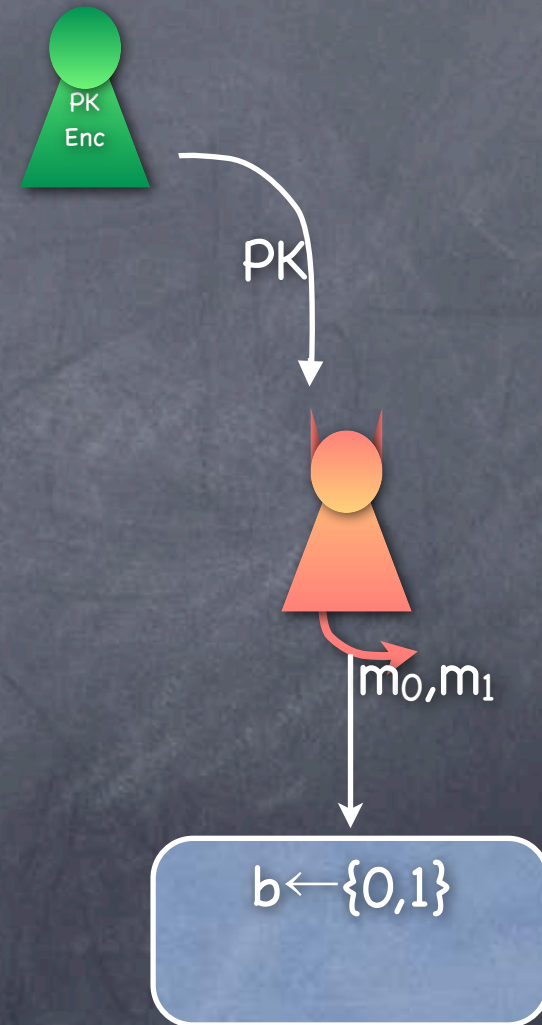
- Experiment picks a random bit b . It also runs KeyGen to get a key (PK, SK) . Adv given PK



- Adv sends two messages m_0, m_1 to the experiment
- Expt returns $\text{Enc}(m_b, K)$ to the adversary
- Adversary returns a guess b'
- Experiment outputs 1 iff $b' = b$
- **IND-CPA secure** if for all PPT adversaries $\Pr[b' = b] - 1/2 \leq \nu(k)$

IND-CPA (PKE version)

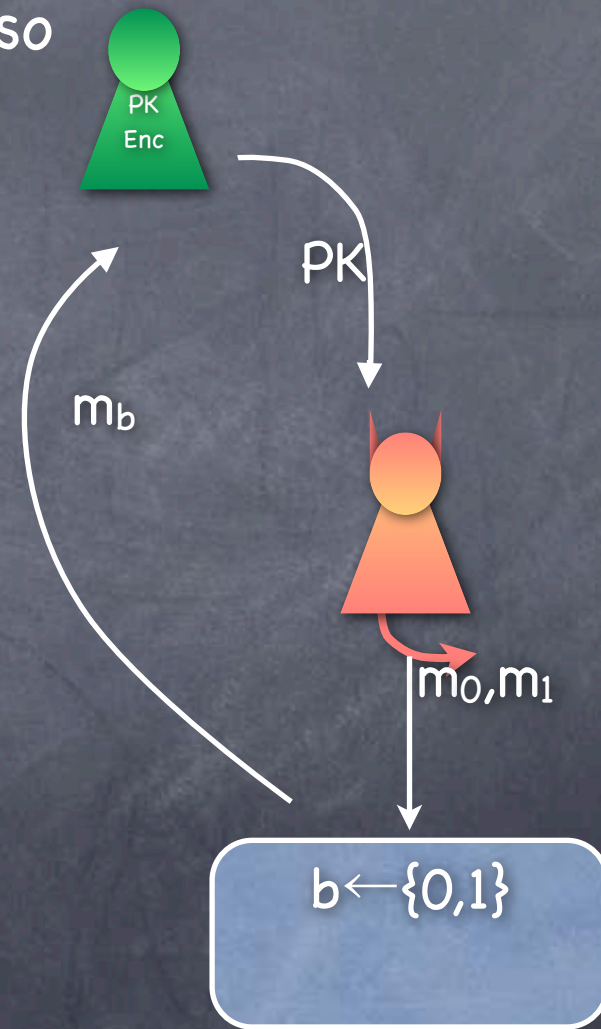
- Experiment picks a random bit b . It also runs KeyGen to get a key (PK, SK) . Adv given PK



- Adv sends two messages m_0, m_1 to the experiment
- Expt returns $\text{Enc}(m_b, K)$ to the adversary
- Adversary returns a guess b'
- Experiment outputs 1 iff $b' = b$
- **IND-CPA secure** if for all PPT adversaries $\Pr[b' = b] - 1/2 \leq \nu(k)$

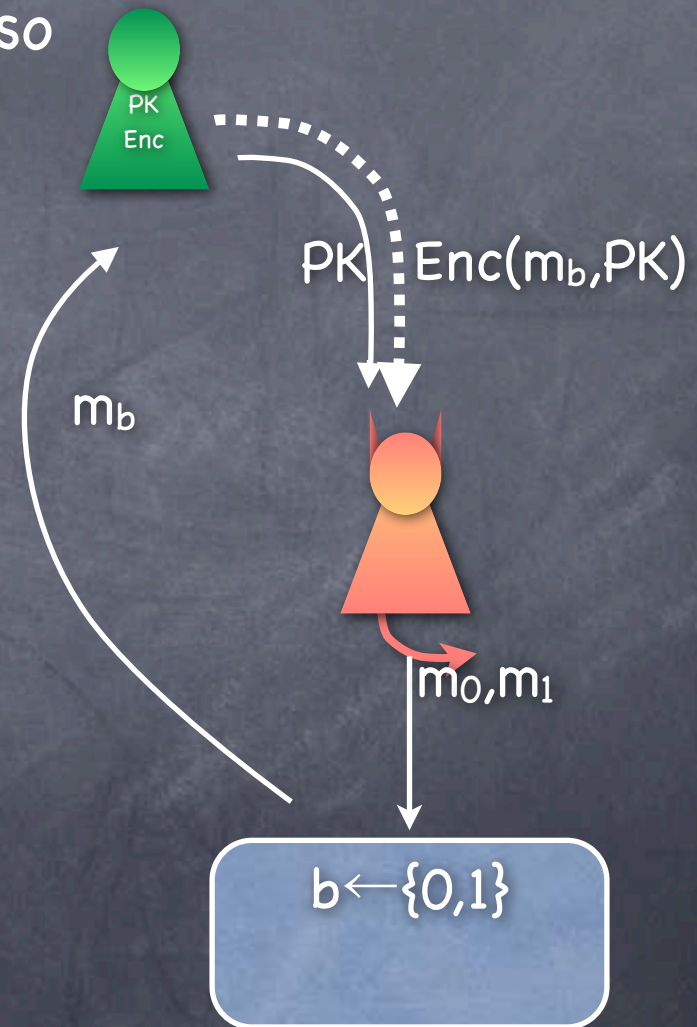
IND-CPA (PKE version)

- Experiment picks a random bit b . It also runs KeyGen to get a key (PK, SK) . Adv given PK
 - Adv sends two messages m_0, m_1 to the experiment
 - Expt returns $Enc(m_b, K)$ to the adversary
 - Adversary returns a guess b'
 - Experiment outputs 1 iff $b' = b$
- **IND-CPA secure** if for all PPT adversaries $\Pr[b' = b] - 1/2 \leq \nu(k)$



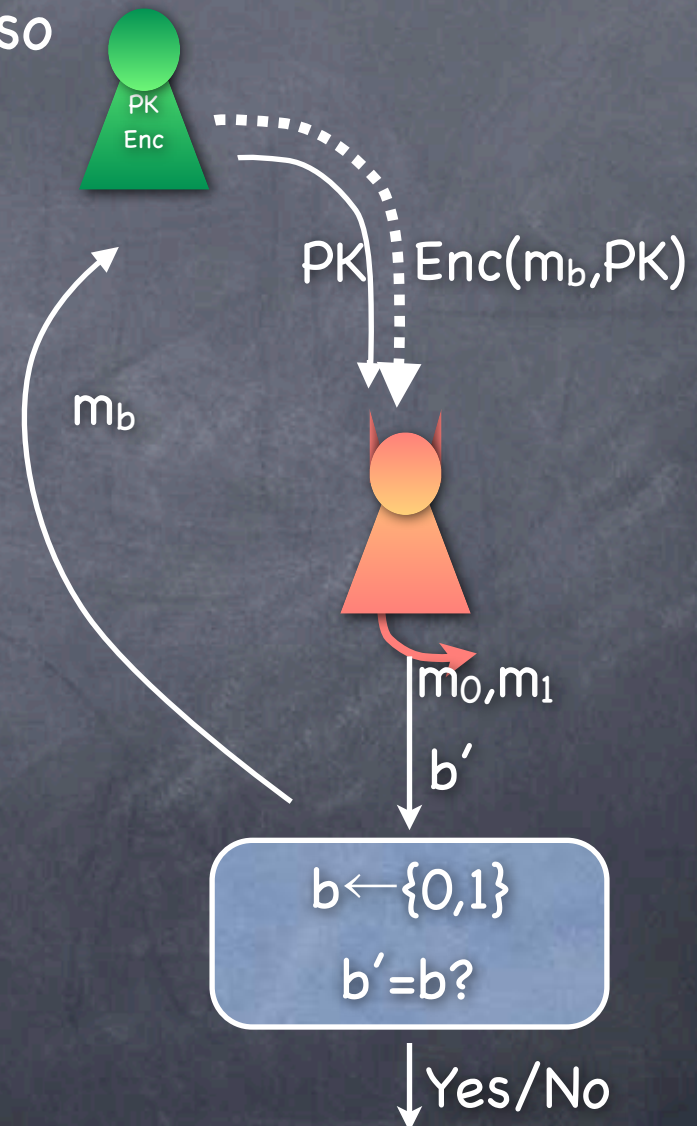
IND-CPA (PKE version)

- Experiment picks a random bit b . It also runs KeyGen to get a key (PK, SK) . Adv given PK
 - Adv sends two messages m_0, m_1 to the experiment
 - Expt returns $Enc(m_b, K)$ to the adversary
 - Adversary returns a guess b'
 - Experiment outputs 1 iff $b' = b$
- **IND-CPA secure** if for all PPT adversaries $\Pr[b' = b] - 1/2 \leq \nu(k)$



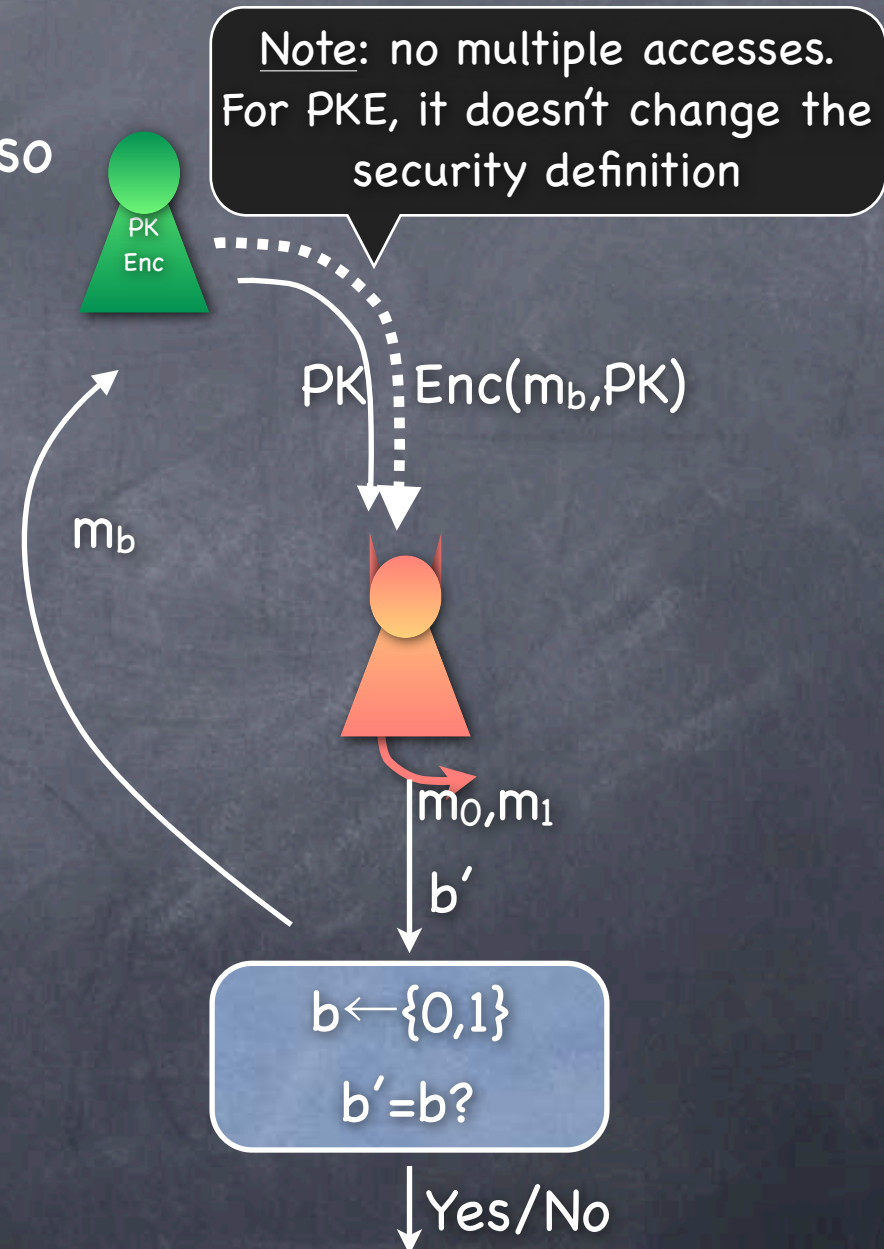
IND-CPA (PKE version)

- Experiment picks a random bit b . It also runs KeyGen to get a key (PK, SK) . Adv given PK
 - Adv sends two messages m_0, m_1 to the experiment
 - Expt returns $Enc(m_b, K)$ to the adversary
 - Adversary returns a guess b'
 - Experiment outputs 1 iff $b' = b$
- **IND-CPA secure** if for all PPT adversaries $\Pr[b' = b] - 1/2 \leq \nu(k)$



IND-CPA (PKE version)

- Experiment picks a random bit b . It also runs KeyGen to get a key (PK, SK) . Adv given PK
 - Adv sends two messages m_0, m_1 to the experiment
 - Expt returns $Enc(m_b, K)$ to the adversary
 - Adversary returns a guess b'
 - Experiment outputs 1 iff $b' = b$
- **IND-CPA secure** if for all PPT adversaries $\Pr[b' = b] - 1/2 \leq \nu(k)$



Perfect Secrecy?

Perfect Secrecy?

- No perfectly secret and correct PKE (even for one-time encryption)

Perfect Secrecy?

- No perfectly secret and correct PKE (even for one-time encryption)
 - Public-key and ciphertext (the total shared information between Alice and Bob at the end) should together have entire information about the message

Perfect Secrecy?

- No perfectly secret and correct PKE (even for one-time encryption)
 - Public-key and ciphertext (the total shared information between Alice and Bob at the end) should together have entire information about the message
 - Intuition: If Eve thinks Bob could decrypt it as two messages based on different SKs, Alice should be concerned too

Perfect Secrecy?

- No perfectly secret and correct PKE (even for one-time encryption)
 - Public-key and ciphertext (the total shared information between Alice and Bob at the end) should together have entire information about the message
 - Intuition: If Eve thinks Bob could decrypt it as two messages based on different SKs, Alice should be concerned too
 - i.e., Alice conveys same information to Bob and Eve

Perfect Secrecy?

- No perfectly secret and correct PKE (even for one-time encryption)
 - Public-key and ciphertext (the total shared information between Alice and Bob at the end) should together have entire information about the message
 - Intuition: If Eve thinks Bob could decrypt it as two messages based on different SKs, Alice should be concerned too
 - i.e., Alice conveys same information to Bob and Eve
 - [Exercise]

Perfect Secrecy?

- No perfectly secret and correct PKE (even for one-time encryption)
 - Public-key and ciphertext (the total shared information between Alice and Bob at the end) should together have entire information about the message
 - Intuition: If Eve thinks Bob could decrypt it as two messages based on different SKs, Alice should be concerned too
 - i.e., Alice conveys same information to Bob and Eve
 - [Exercise]
- PKE only with computational security

Perfect Secrecy?

- No perfectly secret and correct PKE (even for one-time encryption)
 - Public-key and ciphertext (the total shared information between Alice and Bob at the end) should together have entire information about the message
 - Intuition: If Eve thinks Bob could decrypt it as two messages based on different SKs, Alice should be concerned too
 - i.e., Alice conveys same information to Bob and Eve
 - [Exercise]
- PKE only with computational security

Unless
assumptions of
imperfect
eavesdropping

Diffie-Hellman Key-exchange

Diffie-Hellman Key-exchange

- A candidate for how Alice and Bob could generate a shared key, which is “hidden” from Eve

Diffie-Hellman Key-exchange

- A candidate for how Alice and Bob could generate a shared key, which is “hidden” from Eve



Diffie-Hellman Key-exchange

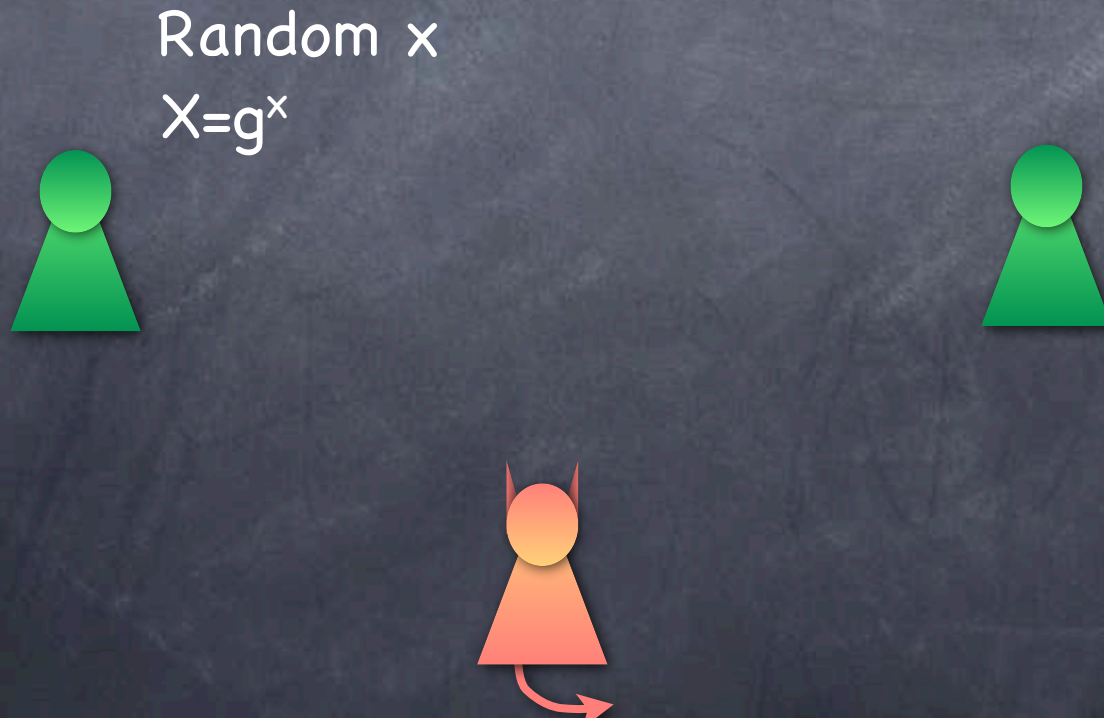
- A candidate for how Alice and Bob could generate a shared key, which is “hidden” from Eve

Random x



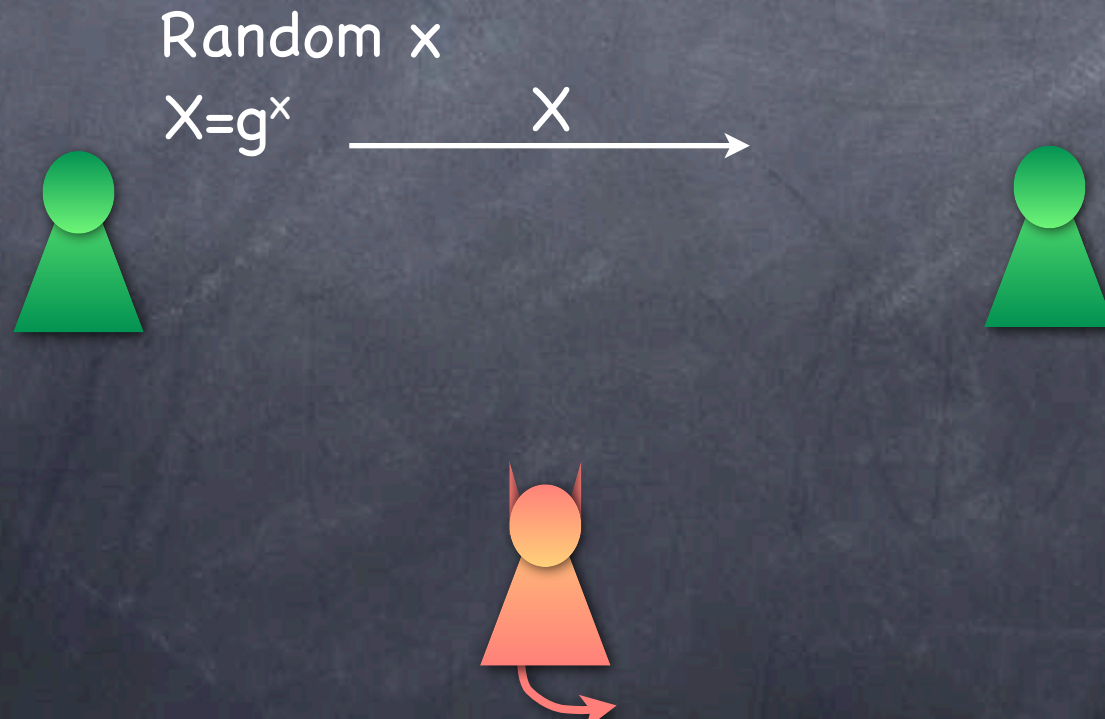
Diffie-Hellman Key-exchange

- A candidate for how Alice and Bob could generate a shared key, which is “hidden” from Eve



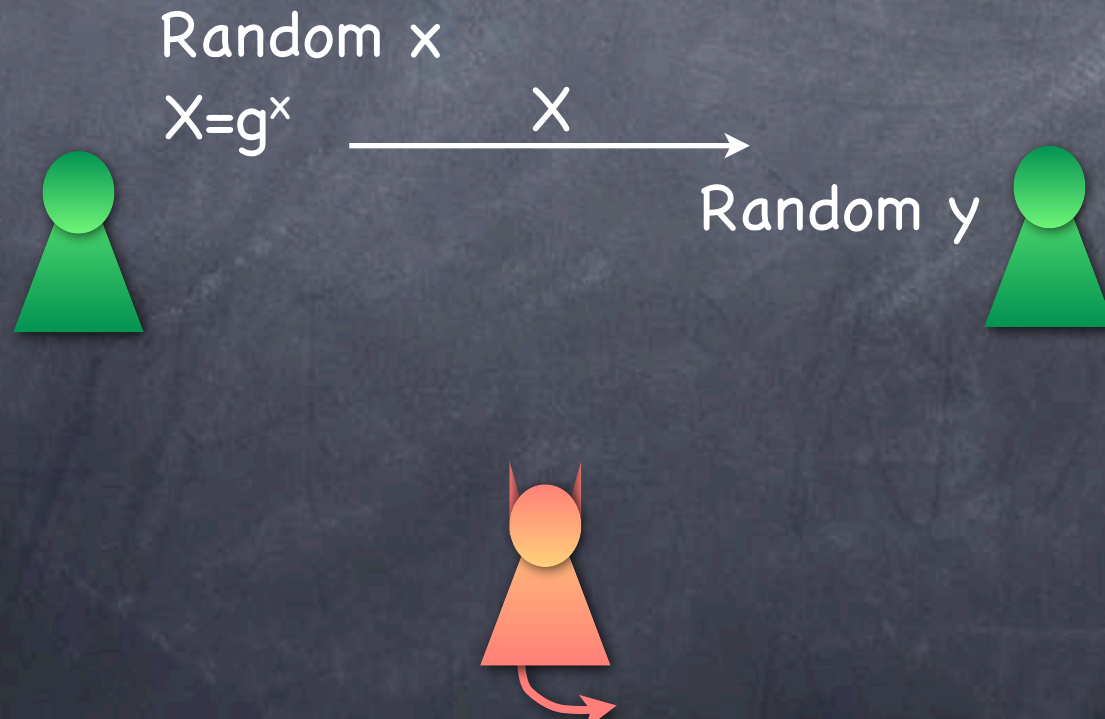
Diffie-Hellman Key-exchange

- A candidate for how Alice and Bob could generate a shared key, which is "hidden" from Eve



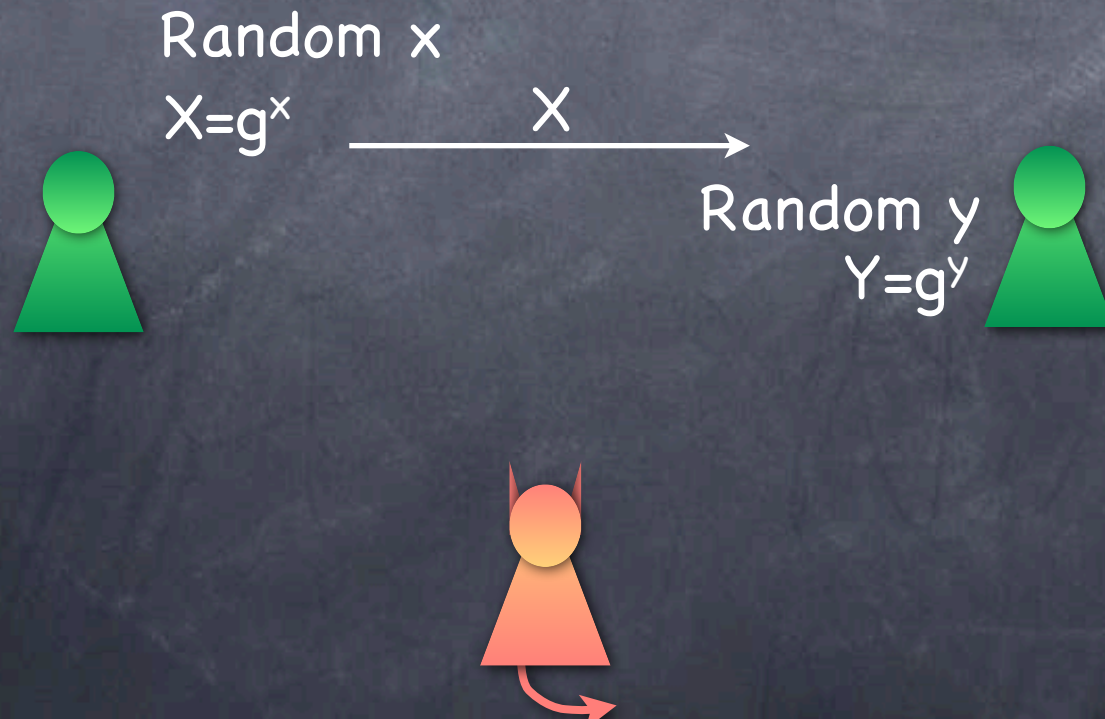
Diffie-Hellman Key-exchange

- A candidate for how Alice and Bob could generate a shared key, which is "hidden" from Eve



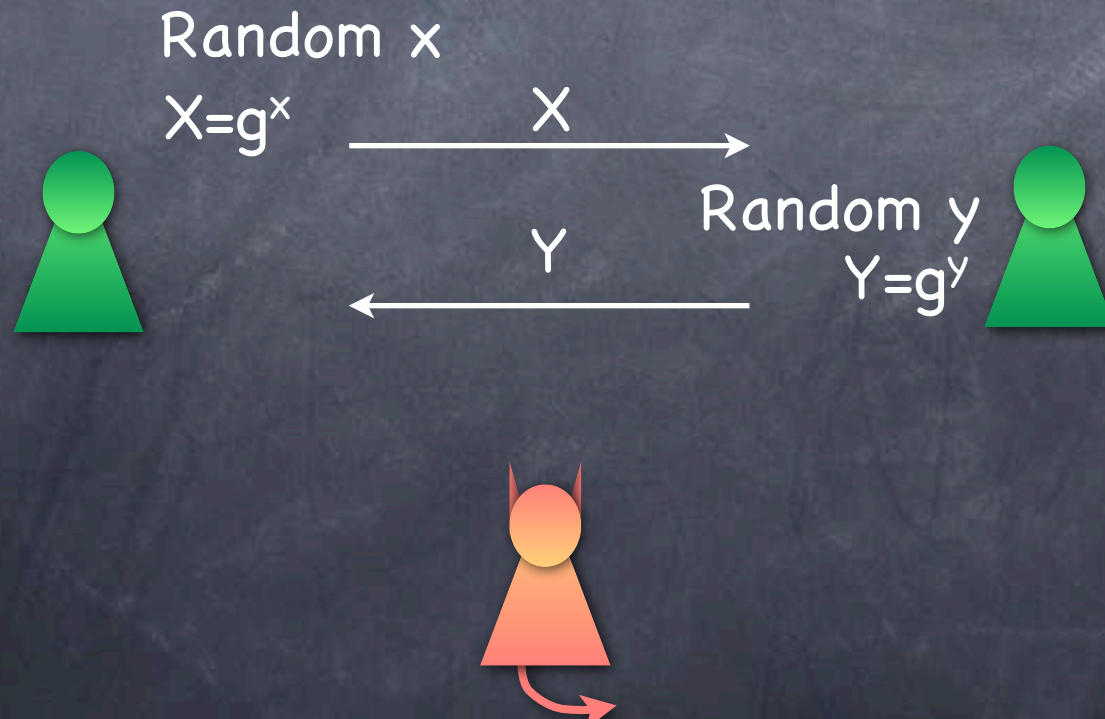
Diffie-Hellman Key-exchange

- A candidate for how Alice and Bob could generate a shared key, which is "hidden" from Eve



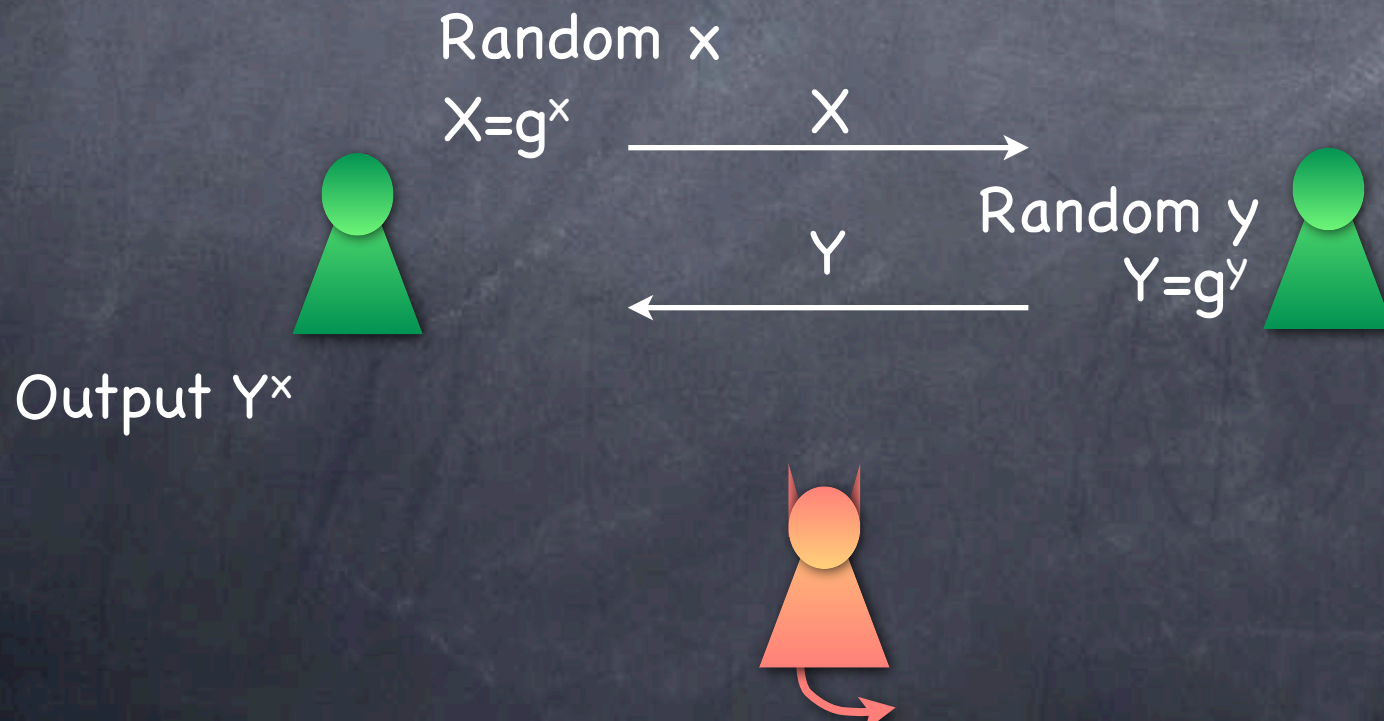
Diffie-Hellman Key-exchange

- A candidate for how Alice and Bob could generate a shared key, which is "hidden" from Eve



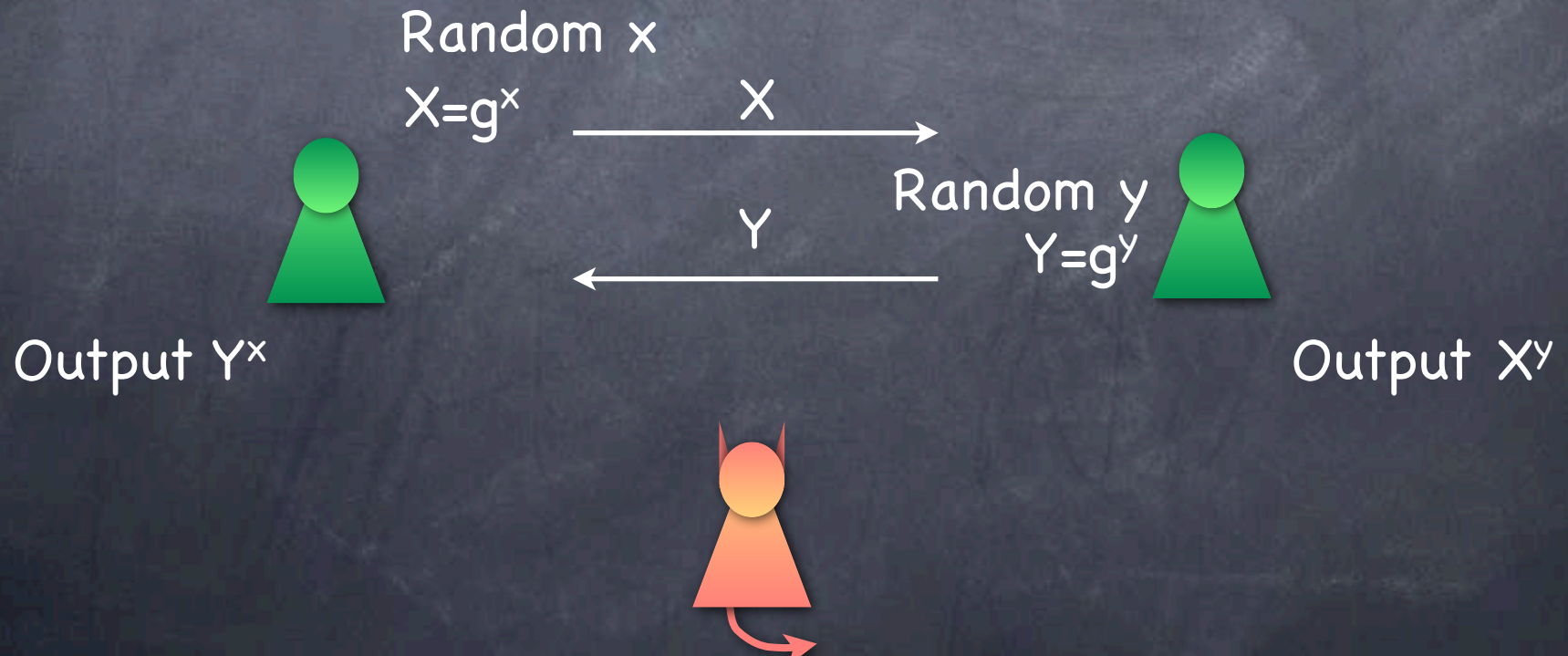
Diffie-Hellman Key-exchange

- A candidate for how Alice and Bob could generate a shared key, which is "hidden" from Eve



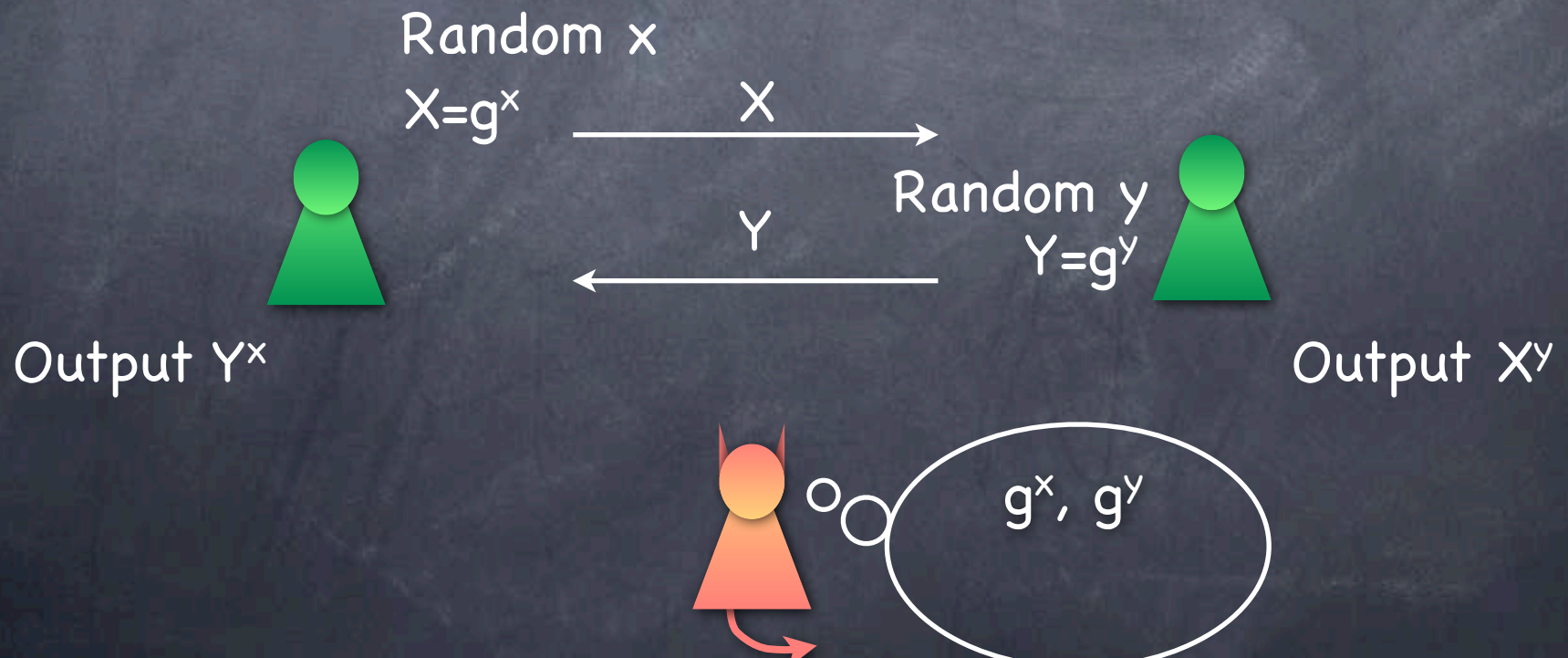
Diffie-Hellman Key-exchange

- A candidate for how Alice and Bob could generate a shared key, which is "hidden" from Eve



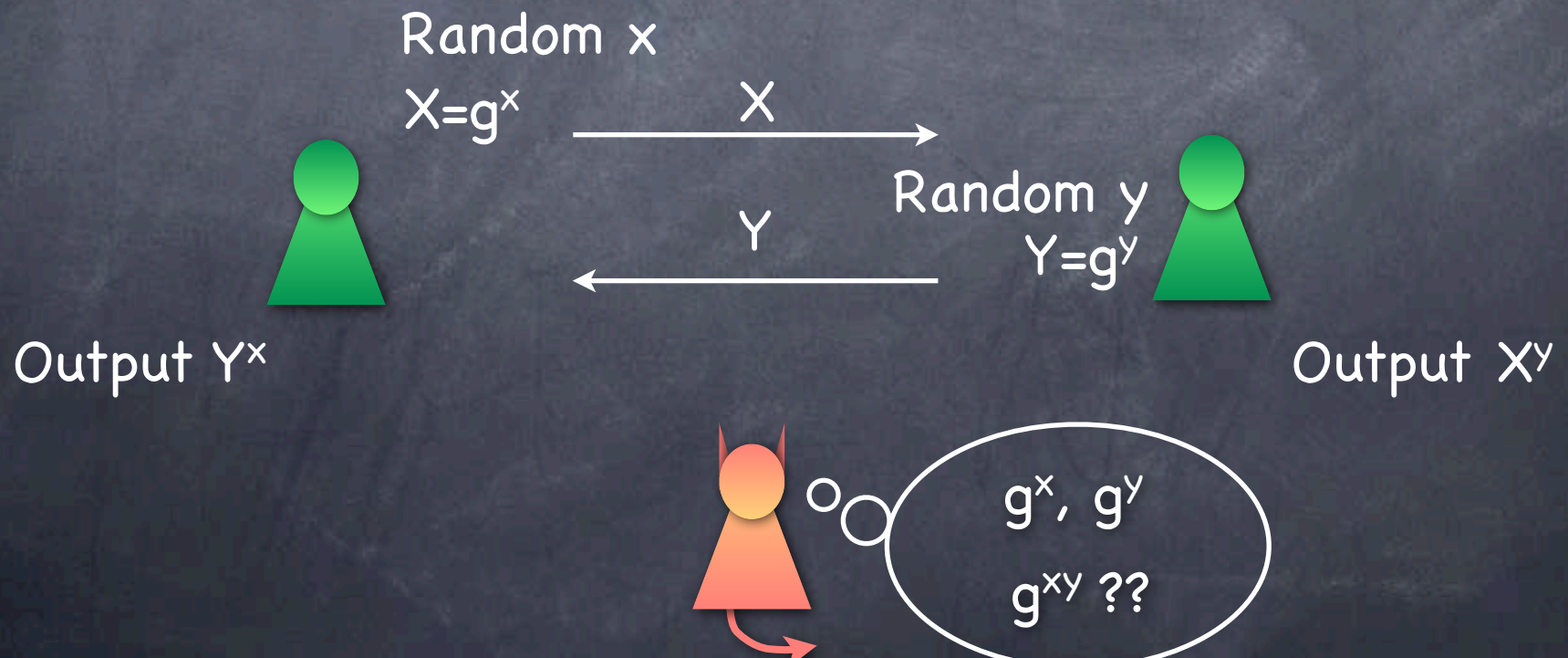
Diffie-Hellman Key-exchange

- A candidate for how Alice and Bob could generate a shared key, which is "hidden" from Eve



Diffie-Hellman Key-exchange

- A candidate for how Alice and Bob could generate a shared key, which is "hidden" from Eve



Why DH-Key-exchange
could be secure

Why DH-Key-exchange could be secure

- Given g^x, g^y for random x, y , g^{xy} should be "hidden"

Why DH-Key-exchange could be secure

- Given g^x, g^y for random x, y , g^{xy} should be "hidden"
 - i.e., could still be used as a pseudorandom element

Why DH-Key-exchange could be secure

- Given g^x, g^y for random x, y , g^{xy} should be "hidden"
 - i.e., could still be used as a pseudorandom element
 - i.e., $(g^x, g^y, g^{xy}) \approx (g^x, g^y, R)$

Why DH-Key-exchange could be secure

- Given g^x, g^y for random x, y , g^{xy} should be "hidden"
 - i.e., could still be used as a pseudorandom element
 - i.e., $(g^x, g^y, g^{xy}) \approx (g^x, g^y, R)$
- Is that reasonable to expect?

Why DH-Key-exchange could be secure

- Given g^x, g^y for random x, y , g^{xy} should be "hidden"
 - i.e., could still be used as a pseudorandom element
 - i.e., $(g^x, g^y, g^{xy}) \approx (g^x, g^y, R)$
- Is that reasonable to expect?
 - Depends on the "group"

Groups, by examples

Groups, by examples

- A set G (for us finite, unless otherwise specified) and a "group operation" $*$ that is associative, has an identity, is invertible, and (for us) commutative

Groups, by examples

- A set G (for us finite, unless otherwise specified) and a "group operation" $*$ that is associative, has an identity, is invertible, and (for us) commutative
- Examples: $\mathbb{Z} = (\text{integers}, +)$ (this is an infinite group),
 $\mathbb{Z}_N = (\text{integers modulo } N, + \text{ mod } N)$,
 $G^n = (\text{Cartesian product of a group } G, \text{ coordinate-wise operation})$

Groups, by examples

- A set G (for us finite, unless otherwise specified) and a "group operation" $*$ that is associative, has an identity, is invertible, and (for us) commutative
- Examples: $\mathbb{Z} = (\text{integers}, +)$ (this is an infinite group),
 $\mathbb{Z}_N = (\text{integers modulo } N, + \text{ mod } N)$,
 $G^n = (\text{Cartesian product of a group } G, \text{ coordinate-wise operation})$
- Order of a group G : $|G| = \text{number of elements in } G$

Groups, by examples

- A set G (for us finite, unless otherwise specified) and a "group operation" $*$ that is associative, has an identity, is invertible, and (for us) commutative
- Examples: $\mathbb{Z} = (\text{integers}, +)$ (this is an infinite group),
 $\mathbb{Z}_N = (\text{integers modulo } N, + \text{ mod } N)$,
 $G^n = (\text{Cartesian product of a group } G, \text{ coordinate-wise operation})$
- Order of a group G : $|G| = \text{number of elements in } G$
- For any $a \in G$, $a^{|G|} = a * a * \dots * a$ ($|G|$ times) = identity

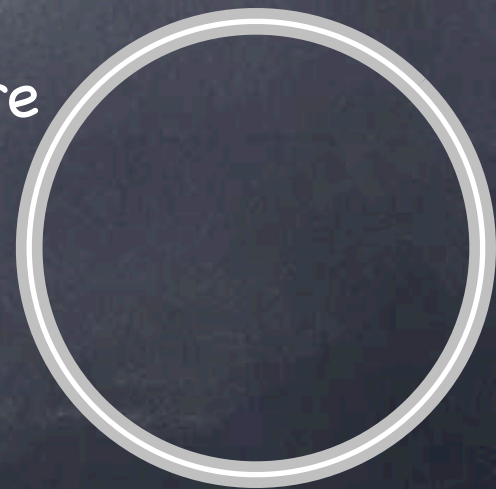
Groups, by examples

- A set G (for us finite, unless otherwise specified) and a "group operation" $*$ that is associative, has an identity, is invertible, and (for us) commutative
- Examples: $\mathbb{Z} = (\text{integers}, +)$ (this is an infinite group),
 $\mathbb{Z}_N = (\text{integers modulo } N, + \text{ mod } N)$,
 $G^n = (\text{Cartesian product of a group } G, \text{ coordinate-wise operation})$
- Order of a group G : $|G| = \text{number of elements in } G$
- For any $a \in G$, $a^{|G|} = a * a * \dots * a$ ($|G|$ times) = identity



Groups, by examples

- A set G (for us finite, unless otherwise specified) and a "group operation" $*$ that is associative, has an identity, is invertible, and (for us) commutative
- Examples: $\mathbb{Z} = (\text{integers}, +)$ (this is an infinite group),
 $\mathbb{Z}_N = (\text{integers modulo } N, + \text{ mod } N)$,
 $G^n = (\text{Cartesian product of a group } G, \text{ coordinate-wise operation})$
- Order of a group G : $|G| = \text{number of elements in } G$
- For any $a \in G$, $a^{|G|} = a * a * \dots * a$ ($|G|$ times) = identity
- Finite **Cyclic group** (in multiplicative notation): there is one element g such that $G = \{g^0, g^1, g^2, \dots, g^{|G|-1}\}$



Groups, by examples

- A set G (for us finite, unless otherwise specified) and a "group operation" $*$ that is associative, has an identity, is invertible, and (for us) commutative
- Examples: $\mathbb{Z} = (\text{integers}, +)$ (this is an infinite group),
 $\mathbb{Z}_N = (\text{integers modulo } N, + \text{ mod } N)$,
 $G^n = (\text{Cartesian product of a group } G, \text{ coordinate-wise operation})$
- Order of a group G : $|G| = \text{number of elements in } G$
- For any $a \in G$, $a^{|G|} = a * a * \dots * a$ ($|G|$ times) = identity
- Finite **Cyclic group** (in multiplicative notation): there is one element g such that $G = \{g^0, g^1, g^2, \dots, g^{|G|-1}\}$



Groups, by examples

- A set G (for us finite, unless otherwise specified) and a "group operation" $*$ that is associative, has an identity, is invertible, and (for us) commutative
- Examples: $\mathbb{Z} = (\text{integers}, +)$ (this is an infinite group),
 $\mathbb{Z}_N = (\text{integers modulo } N, + \text{ mod } N)$,
 $G^n = (\text{Cartesian product of a group } G, \text{ coordinate-wise operation})$
- Order of a group G : $|G| = \text{number of elements in } G$
- For any $a \in G$, $a^{|G|} = a * a * \dots * a$ ($|G|$ times) = identity
- Finite **Cyclic group** (in multiplicative notation): there is one element g such that $G = \{g^0, g^1, g^2, \dots, g^{|G|-1}\}$
 - Prototype: \mathbb{Z}_N (additive group), with $g=1$



Groups, by examples

- A set G (for us finite, unless otherwise specified) and a "group operation" $*$ that is associative, has an identity, is invertible, and (for us) commutative
- Examples: $\mathbb{Z} = (\text{integers}, +)$ (this is an infinite group),
 $\mathbb{Z}_N = (\text{integers modulo } N, + \text{ mod } N)$,
 $G^n = (\text{Cartesian product of a group } G, \text{ coordinate-wise operation})$
- Order of a group G : $|G| = \text{number of elements in } G$
- For any $a \in G$, $a^{|G|} = a * a * \dots * a$ ($|G|$ times) = identity
- Finite **Cyclic group** (in multiplicative notation): there is one element g such that $G = \{g^0, g^1, g^2, \dots, g^{|G|-1}\}$
 - Prototype: \mathbb{Z}_N (additive group), with $g=1$
 - or any g s.t. $\text{gcd}(g, N) = 1$



Groups, by examples



Groups, by examples



- \mathbb{Z}_N^* = (generators of \mathbb{Z}_N , multiplication mod N)

Groups, by examples



- \mathbb{Z}_N^* = (generators of \mathbb{Z}_N , multiplication mod N)
 - Numbers in $\{0, \dots, N-1\}$ which have a multiplicative inverse mod N

Groups, by examples



- \mathbb{Z}_N^* = (generators of \mathbb{Z}_N , multiplication mod N)
 - Numbers in $\{0, \dots, N-1\}$ which have a multiplicative inverse mod N
 - If N is prime, \mathbb{Z}_N^* is a cyclic group, of order $N-1$

Groups, by examples



- \mathbb{Z}_N^* = (generators of \mathbb{Z}_N , multiplication mod N)
 - Numbers in $\{0, \dots, N-1\}$ which have a multiplicative inverse mod N
 - If N is prime, \mathbb{Z}_N^* is a cyclic group, of order N-1
 - e.g. $\mathbb{Z}_5^* = \{1, 2, 3, 4\}$ is generated by 2 (as 1, 2, 4, 3), and by 3 (as 1, 3, 4, 2)

Groups, by examples



- \mathbb{Z}_N^* = (generators of \mathbb{Z}_N , multiplication mod N)
 - Numbers in $\{0, \dots, N-1\}$ which have a multiplicative inverse mod N
 - If N is prime, \mathbb{Z}_N^* is a cyclic group, of order $N-1$
 - e.g. $\mathbb{Z}_5^* = \{1, 2, 3, 4\}$ is generated by 2 (as 1, 2, 4, 3), and by 3 (as 1, 3, 4, 2)
 - (Also cyclic for certain other values of N)

Discrete Log Assumption

Discrete Log Assumption

- **Discrete Log** (w.r.t g) in a (multiplicative) cyclic group G generated by g : $DL_g(X) :=$ unique x such that $X = g^x$ ($x \in \{0, 1, \dots, |G|-1\}$)

Discrete Log Assumption

- **Discrete Log** (w.r.t g) in a (multiplicative) cyclic group G generated by g : $DL_g(X) := \text{unique } x \text{ such that } X = g^x$ ($x \in \{0, 1, \dots, |G|-1\}$)
- In a (computationally efficient) group, given standard representation of g and x , can efficiently find the standard representation of $X=g^x$ (**How?**)

Discrete Log Assumption

Repeated
squaring

- **Discrete Log** (w.r.t g) in a (multiplicative) cyclic group G generated by g : $DL_g(X) := \text{unique } x \text{ such that } X = g^x$ ($x \in \{0, 1, \dots, |G|-1\}$)
- In a (computationally efficient) group, given standard representation of g and x , can efficiently find the standard representation of $X=g^x$ (**How?**)

Discrete Log Assumption

Repeated
squaring

- **Discrete Log** (w.r.t g) in a (multiplicative) cyclic group G generated by g : $DL_g(X) := \text{unique } x \text{ such that } X = g^x$ ($x \in \{0, 1, \dots, |G|-1\}$)
- In a (computationally efficient) group, given standard representation of g and x , can efficiently find the standard representation of $X=g^x$ (**How?**)
 - But given X and g , **may not be easy** to find x (depending on G)

Discrete Log Assumption

Repeated
squaring

- **Discrete Log** (w.r.t g) in a (multiplicative) cyclic group G generated by g : $DL_g(X) := \text{unique } x \text{ such that } X = g^x$ ($x \in \{0, 1, \dots, |G|-1\}$)
- In a (computationally efficient) group, given standard representation of g and x , can efficiently find the standard representation of $X=g^x$ (**How?**)
 - But given X and g , **may not be easy** to find x (depending on G)
 - **DLA**: Every PPT Adv has negligible success probability in the DL Expt: $(G, g) \leftarrow \text{GroupGen}; X \leftarrow G; \text{Adv}(G, g, X) \rightarrow z; g^z = X?$

Discrete Log Assumption

Repeated
squaring

- **Discrete Log** (w.r.t g) in a (multiplicative) cyclic group G generated by g : $DL_g(X) := \text{unique } x \text{ such that } X = g^x$ ($x \in \{0, 1, \dots, |G|-1\}$)
- In a (computationally efficient) group, given standard representation of g and x , can efficiently find the standard representation of $X=g^x$ (**How?**)
 - But given X and g , **may not be easy** to find x (depending on G)
 - **DLA**: Every PPT Adv has negligible success probability in the DL Expt: $(G, g) \leftarrow \text{GroupGen}; X \leftarrow G; \text{Adv}(G, g, X) \rightarrow z; g^z = X?$

OWF collection:
 $\text{Raise}(x; G, g)$
 $= (g^x; G, g)$

Discrete Log Assumption

Repeated squaring

- **Discrete Log** (w.r.t g) in a (multiplicative) cyclic group G generated by g : $DL_g(X) := \text{unique } x \text{ such that } X = g^x$ ($x \in \{0, 1, \dots, |G|-1\}$)
- In a (computationally efficient) group, given standard representation of g and x , can efficiently find the standard representation of $X=g^x$ (**How?**)
 - But given X and g , **may not be easy** to find x (depending on G)
 - **DLA**: Every PPT Adv has negligible success probability in the DL Expt: $(G, g) \leftarrow \text{GroupGen}; X \leftarrow G; \text{Adv}(G, g, X) \rightarrow z; g^z = X?$
- If DLA broken, then Diffie-Hellman key-exchange broken

OWF collection:
 $\text{Raise}(x; G, g)$
 $= (g^x; G, g)$

Discrete Log Assumption

Repeated squaring

- **Discrete Log** (w.r.t g) in a (multiplicative) cyclic group G generated by g : $DL_g(X) := \text{unique } x \text{ such that } X = g^x$ ($x \in \{0, 1, \dots, |G|-1\}$)
- In a (computationally efficient) group, given standard representation of g and x , can efficiently find the standard representation of $X=g^x$ (**How?**)
 - But given X and g , **may not be easy** to find x (depending on G)
 - **DLA**: Every PPT Adv has negligible success probability in the DL Expt: $(G, g) \leftarrow \text{GroupGen}; X \leftarrow G; \text{Adv}(G, g, X) \rightarrow z; g^z = X?$
- If DLA broken, then Diffie-Hellman key-exchange broken
 - Eve gets x, y from g^x, g^y (sometimes) and can compute g^{xy} herself

OWF collection:
 $\text{Raise}(x; G, g)$
 $= (g^x; G, g)$

Discrete Log Assumption

Repeated squaring

- **Discrete Log** (w.r.t g) in a (multiplicative) cyclic group G generated by g : $DL_g(X) := \text{unique } x \text{ such that } X = g^x$ ($x \in \{0, 1, \dots, |G|-1\}$)
- In a (computationally efficient) group, given standard representation of g and x , can efficiently find the standard representation of $X=g^x$ (**How?**)
 - But given X and g , **may not be easy** to find x (depending on G)
 - **DLA**: Every PPT Adv has negligible success probability in the DL Expt: $(G, g) \leftarrow \text{GroupGen}; X \leftarrow G; \text{Adv}(G, g, X) \rightarrow z; g^z = X?$
- If DLA broken, then Diffie-Hellman key-exchange broken
 - Eve gets x, y from g^x, g^y (sometimes) and can compute g^{xy} herself
 - A "key-recovery" attack

OWF collection:
 $\text{Raise}(x; G, g)$
 $= (g^x; G, g)$

Discrete Log Assumption

Repeated squaring

- **Discrete Log** (w.r.t g) in a (multiplicative) cyclic group G generated by g : $DL_g(X) := \text{unique } x \text{ such that } X = g^x$ ($x \in \{0, 1, \dots, |G|-1\}$)
- In a (computationally efficient) group, given standard representation of g and x , can efficiently find the standard representation of $X=g^x$ (**How?**)
 - But given X and g , **may not be easy** to find x (depending on G)
 - **DLA**: Every PPT Adv has negligible success probability in the DL Expt: $(G, g) \leftarrow \text{GroupGen}; X \leftarrow G; \text{Adv}(G, g, X) \rightarrow z; g^z = X?$
- If DLA broken, then Diffie-Hellman key-exchange broken
 - Eve gets x, y from g^x, g^y (sometimes) and can compute g^{xy} herself
 - A "key-recovery" attack
 - Note: could break pseudorandomness without breaking DLA too

OWF collection:
 $\text{Raise}(x; G, g)$
 $= (g^x; G, g)$

Decisional Diffie-Hellman (DDH) Assumption

Decisional Diffie-Hellman (DDH) Assumption

- $\{(g^x, g^y, g^{xy})\}_{(G,g) \leftarrow \text{GroupGen}; x,y \leftarrow [|G|]} \approx \{(g^x, g^y, g^r)\}_{(G,g) \leftarrow \text{GroupGen}; x,y,r \leftarrow [|G|]}$

Decisional Diffie-Hellman (DDH) Assumption

- $\{(g^x, g^y, g^{xy})\}_{(G,g) \leftarrow \text{GroupGen}; x,y \leftarrow [|G|]} \approx \{(g^x, g^y, g^r)\}_{(G,g) \leftarrow \text{GroupGen}; x,y,r \leftarrow [|G|]}$
- At least as strong as DLA

Decisional Diffie-Hellman (DDH) Assumption

- $\{(g^x, g^y, g^{xy})\}_{(G,g) \leftarrow \text{GroupGen}; x,y \leftarrow [|G|]} \approx \{(g^x, g^y, g^r)\}_{(G,g) \leftarrow \text{GroupGen}; x,y,r \leftarrow [|G|]}$
- At least as strong as DLA
 - If DDH assumption holds, then DLA holds [Why?]

Decisional Diffie-Hellman (DDH) Assumption

- $\{(g^x, g^y, g^{xy})\}_{(G,g) \leftarrow \text{GroupGen}; x,y \leftarrow [|G|]} \approx \{(g^x, g^y, g^r)\}_{(G,g) \leftarrow \text{GroupGen}; x,y,r \leftarrow [|G|]}$
- At least as strong as DLA
 - If DDH assumption holds, then DLA holds [Why?]
- But possible that DLA holds and DDH assumption doesn't

Decisional Diffie-Hellman (DDH) Assumption

- $\{(g^x, g^y, g^{xy})\}_{(G,g) \leftarrow \text{GroupGen}; x,y \leftarrow [|G|]} \approx \{(g^x, g^y, g^r)\}_{(G,g) \leftarrow \text{GroupGen}; x,y,r \leftarrow [|G|]}$
- At least as strong as DLA
 - If DDH assumption holds, then DLA holds [Why?]
- But possible that DLA holds and DDH assumption doesn't
 - e.g.: DLA is widely assumed to hold in \mathbb{Z}_p^* (p prime), but DDH assumption doesn't hold there!

A Candidate DDH Group



A Candidate DDH Group

- Consider \mathbb{QR}_p^* : subgroup of Quadratic Residues (“even power” elements) of \mathbb{Z}_p^*



A Candidate DDH Group

- Consider \mathbb{QR}_p^* : subgroup of Quadratic Residues ("even power" elements) of \mathbb{Z}_p^*



A Candidate DDH Group

- Consider \mathbb{QR}_p^* : subgroup of Quadratic Residues (“even power” elements) of \mathbb{Z}_p^*
- Easy to check if an element is a QR or not: check if raising to $|G|/2$ gives 1 (identity element)



A Candidate DDH Group

- Consider \mathbb{QR}_p^* : subgroup of Quadratic Residues ("even power" elements) of \mathbb{Z}_p^*
- Easy to check if an element is a QR or not: check if raising to $|G|/2$ gives 1 (identity element)
- DDH does not hold in \mathbb{Z}_p^*** : g^{xy} is a QR w/ prob. $3/4$; g^z is QR only w/ prob. $1/2$.



A Candidate DDH Group

- Consider \mathbb{QR}_p^* : subgroup of Quadratic Residues (“even power” elements) of \mathbb{Z}_p^*
- Easy to check if an element is a QR or not: check if raising to $|G|/2$ gives 1 (identity element)
- **DDH does not hold in \mathbb{Z}_p^*** : g^{xy} is a QR w/ prob. $3/4$; g^z is QR only w/ prob. $1/2$.
- How about in \mathbb{QR}_p^* ?



A Candidate DDH Group

- Consider \mathbb{QR}_p^* : subgroup of Quadratic Residues (“even power” elements) of \mathbb{Z}_p^*
- Easy to check if an element is a QR or not: check if raising to $|G|/2$ gives 1 (identity element)
- **DDH does not hold in \mathbb{Z}_p^*** : g^{xy} is a QR w/ prob. $3/4$; g^z is QR only w/ prob. $1/2$.
- How about in \mathbb{QR}_p^* ?
 - Could check if cubic residue in \mathbb{Z}_p^* !



A Candidate DDH Group

- Consider \mathbb{QR}_p^* : subgroup of Quadratic Residues (“even power” elements) of \mathbb{Z}_p^*
- Easy to check if an element is a QR or not: check if raising to $|G|/2$ gives 1 (identity element)
- DDH does not hold in \mathbb{Z}_p^*** : g^{xy} is a QR w/ prob. $3/4$; g^z is QR only w/ prob. $1/2$.
- How about in \mathbb{QR}_p^* ?
 - Could check if cubic residue in \mathbb{Z}_p^* !
 - But if $(p-1)$ is not divisible by 3, all elements in \mathbb{Z}_p^* are cubic residues!



A Candidate DDH Group

- Consider \mathbb{QR}_p^* : subgroup of Quadratic Residues (“even power” elements) of \mathbb{Z}_p^*
- Easy to check if an element is a QR or not: check if raising to $|G|/2$ gives 1 (identity element)
- **DDH does not hold in \mathbb{Z}_p^*** : g^{xy} is a QR w/ prob. $3/4$; g^z is QR only w/ prob. $1/2$.
- How about in \mathbb{QR}_p^* ?
 - Could check if cubic residue in \mathbb{Z}_p^* !
 - But if $(p-1)$ is not divisible by 3, all elements in \mathbb{Z}_p^* are cubic residues!
 - “Safe” if $(p-1)/2$ is also prime: p called a **safe-prime**



A Candidate DDH Group

- Consider \mathbb{QR}_p^* : subgroup of Quadratic Residues ("even power" elements) of \mathbb{Z}_p^*
- Easy to check if an element is a QR or not: check if raising to $|G|/2$ gives 1 (identity element)
- DDH does not hold in \mathbb{Z}_p^* : g^{xy} is a QR w/ prob. $3/4$; g^z is QR only w/ prob. $1/2$.
- How about in \mathbb{QR}_p^* ?



DDH Candidate:

\mathbb{QR}_p^*

where P is a safe-prime

- Could check if cubic residue in \mathbb{Z}_p^* !
 - But if $(P-1)$ is not divisible by 3, all elements in \mathbb{Z}_p^* are cubic residues!
- "Safe" if $(P-1)/2$ is also prime: P called a **safe-prime**

El Gamal Encryption

El Gamal Encryption

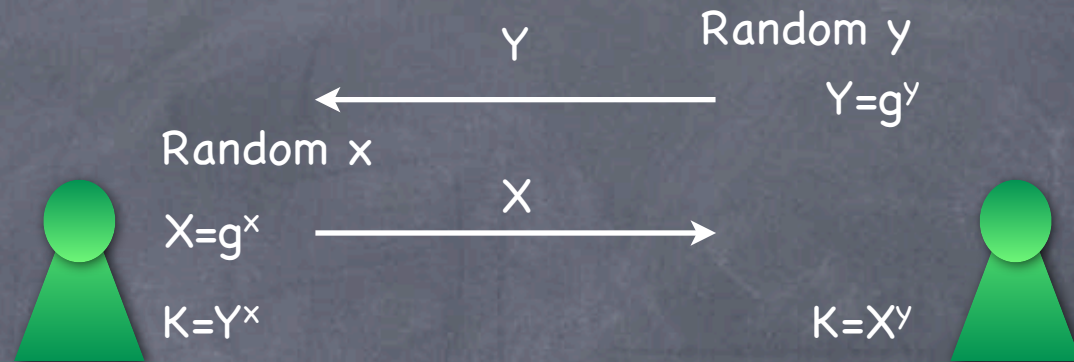
- Based on DH key-exchange

El Gamal Encryption

- Based on DH key-exchange
 - Alice, Bob generate a key using DH key-exchange

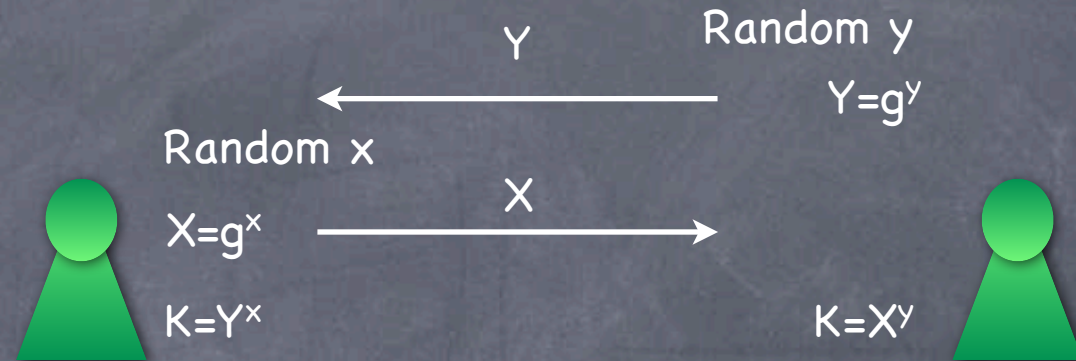
El Gamal Encryption

- Based on DH key-exchange
- Alice, Bob generate a key using DH key-exchange



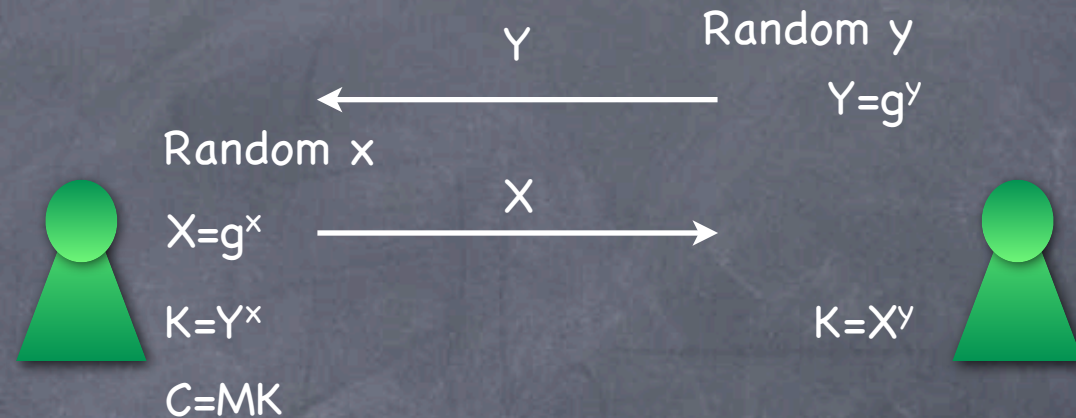
El Gamal Encryption

- Based on DH key-exchange
 - Alice, Bob generate a key using DH key-exchange
 - Then use it as a one-time pad



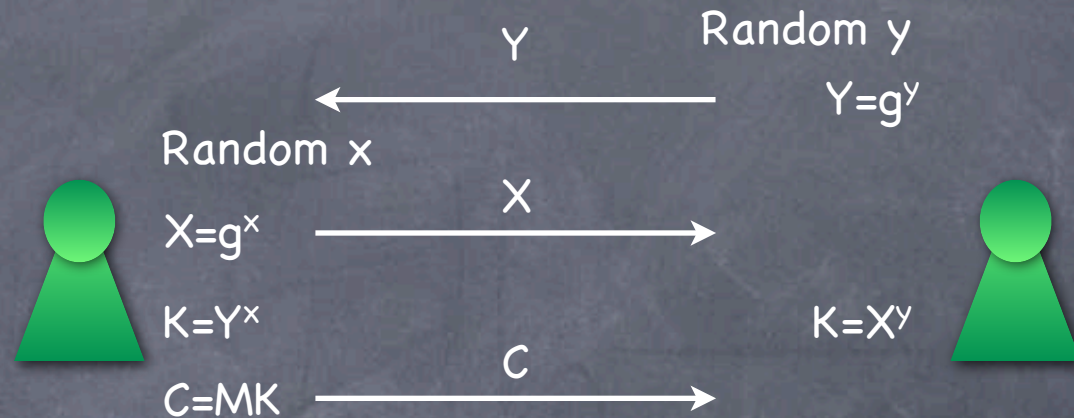
El Gamal Encryption

- Based on DH key-exchange
 - Alice, Bob generate a key using DH key-exchange
 - Then use it as a one-time pad



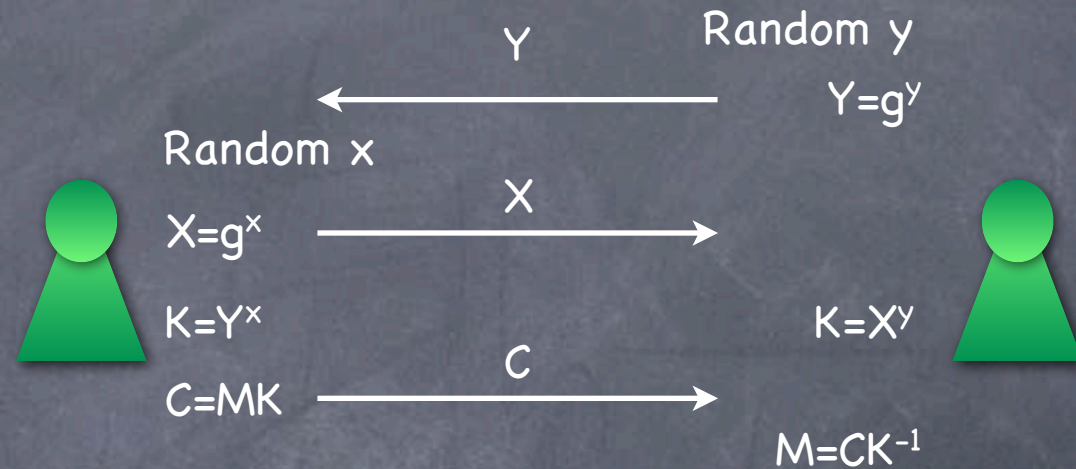
El Gamal Encryption

- Based on DH key-exchange
 - Alice, Bob generate a key using DH key-exchange
 - Then use it as a one-time pad



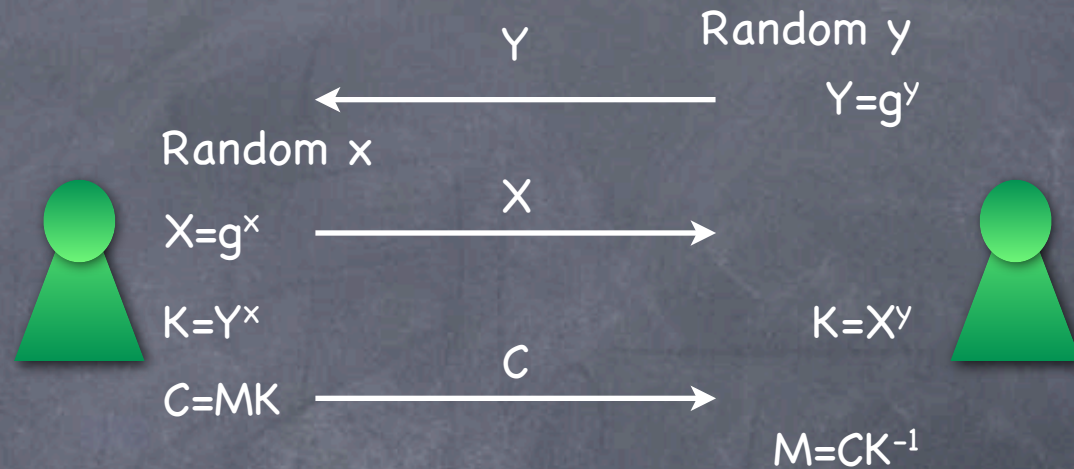
El Gamal Encryption

- Based on DH key-exchange
 - Alice, Bob generate a key using DH key-exchange
 - Then use it as a one-time pad



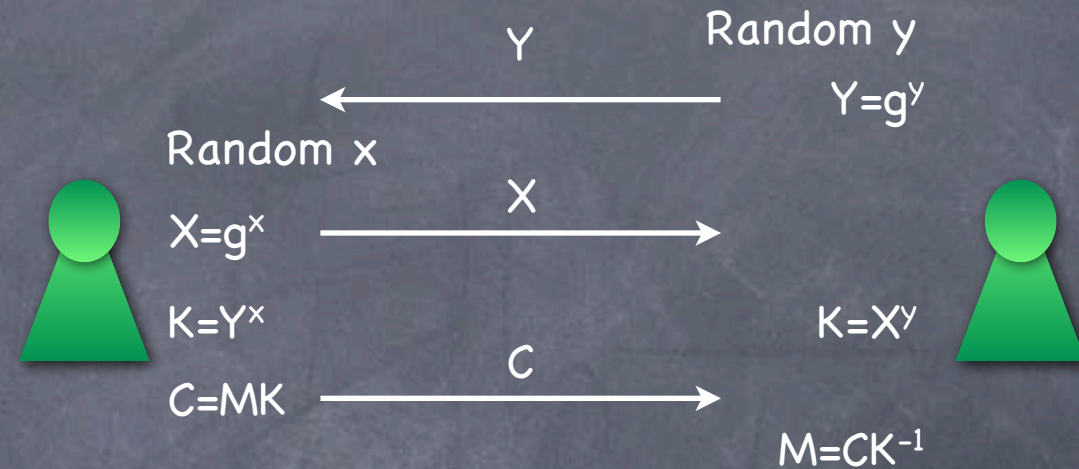
El Gamal Encryption

- Based on DH key-exchange
 - Alice, Bob generate a key using DH key-exchange
 - Then use it as a one-time pad
- Bob's "message" in the key-exchange is his PK



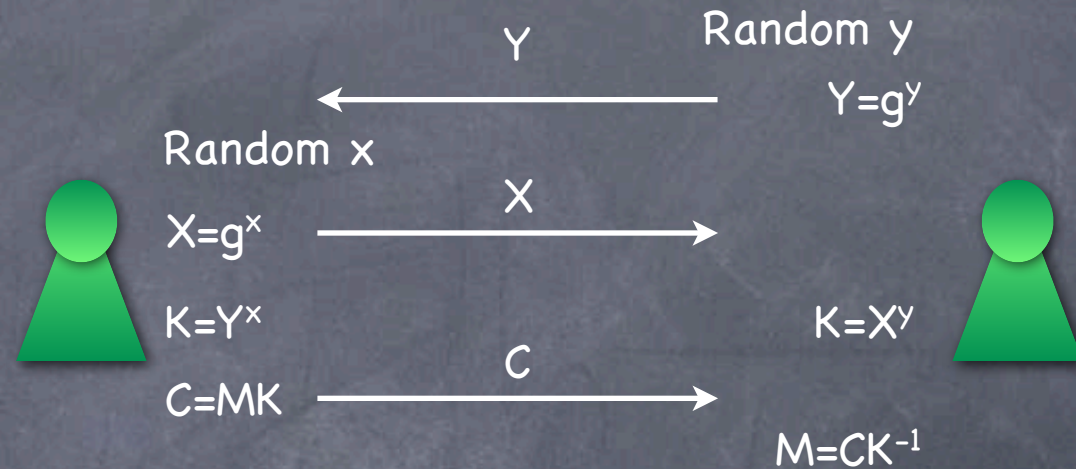
El Gamal Encryption

- Based on DH key-exchange
 - Alice, Bob generate a key using DH key-exchange
 - Then use it as a one-time pad
- Bob's "message" in the key-exchange is his PK
- Alice's message in the key-exchange and the ciphertext of the one-time pad together form a single ciphertext



El Gamal Encryption

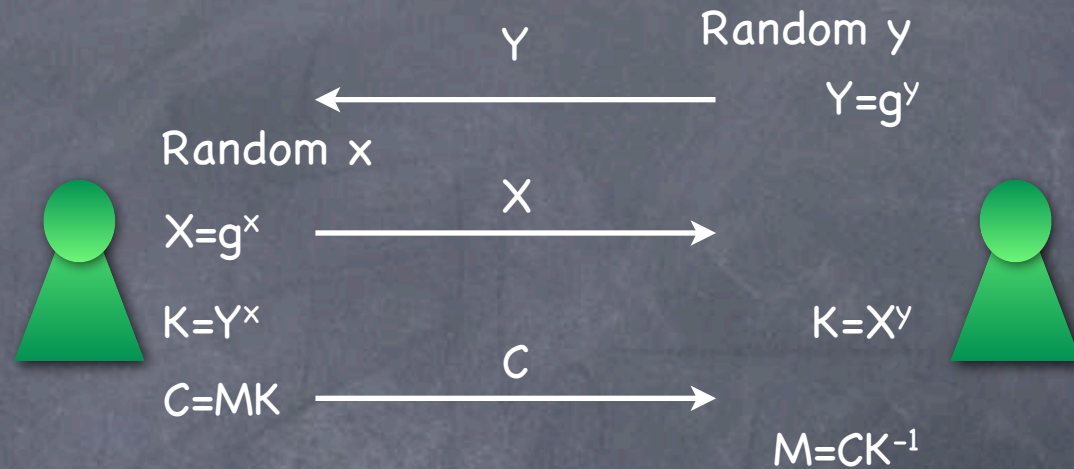
- Based on DH key-exchange
 - Alice, Bob generate a key using DH key-exchange
 - Then use it as a one-time pad
- Bob's "message" in the key-exchange is his PK
- Alice's message in the key-exchange and the ciphertext of the one-time pad together form a single ciphertext



KeyGen: $PK = (G, g, Y)$, $SK = (G, g, y)$

El Gamal Encryption

- Based on DH key-exchange
 - Alice, Bob generate a key using DH key-exchange
 - Then use it as a one-time pad
- Bob's "message" in the key-exchange is his PK
- Alice's message in the key-exchange and the ciphertext of the one-time pad together form a single ciphertext

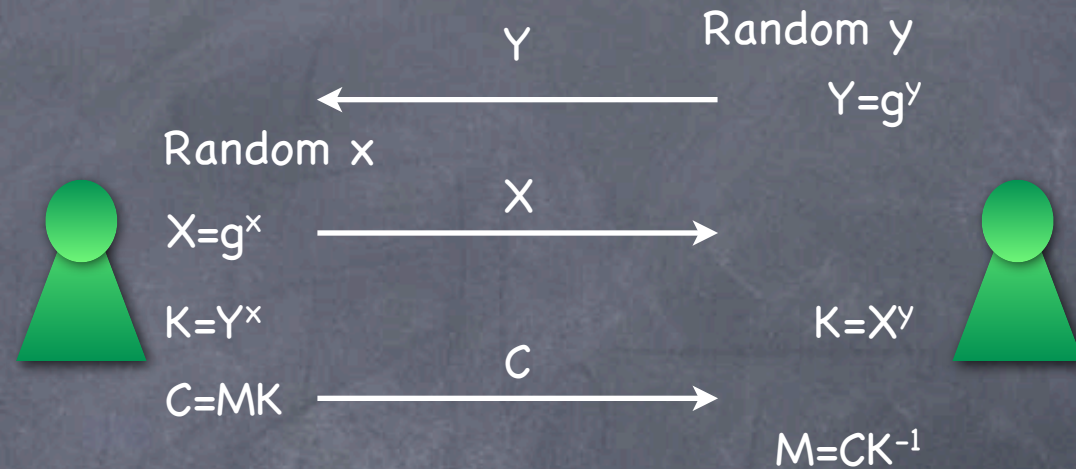


KeyGen: $PK = (G, g, Y)$, $SK = (G, g, y)$

$Enc_{(G, g, Y)}(M) = (X = g^x, C = MY^x)$

El Gamal Encryption

- Based on DH key-exchange
 - Alice, Bob generate a key using DH key-exchange
 - Then use it as a one-time pad
- Bob's "message" in the key-exchange is his PK
- Alice's message in the key-exchange and the ciphertext of the one-time pad together form a single ciphertext



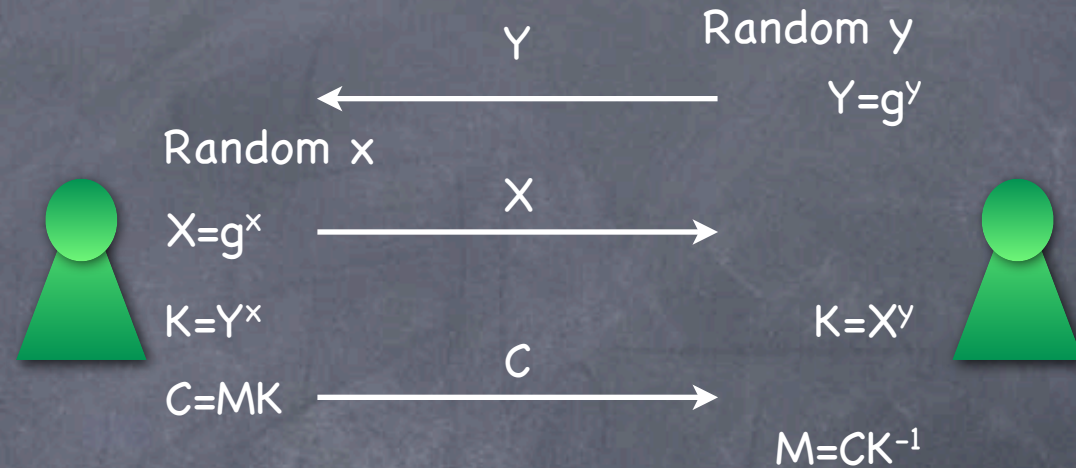
KeyGen: $PK = (G, g, Y)$, $SK = (G, g, y)$

$Enc_{(G, g, Y)}(M) = (X = g^x, C = MY^x)$

$Dec_{(G, g, y)}(X, C) = CX^{-y}$

El Gamal Encryption

- Based on DH key-exchange
 - Alice, Bob generate a key using DH key-exchange
 - Then use it as a one-time pad
- Bob's "message" in the key-exchange is his PK
- Alice's message in the key-exchange and the ciphertext of the one-time pad together form a single ciphertext



$$\text{KeyGen: } PK = (G, g, Y), SK = (G, g, y)$$

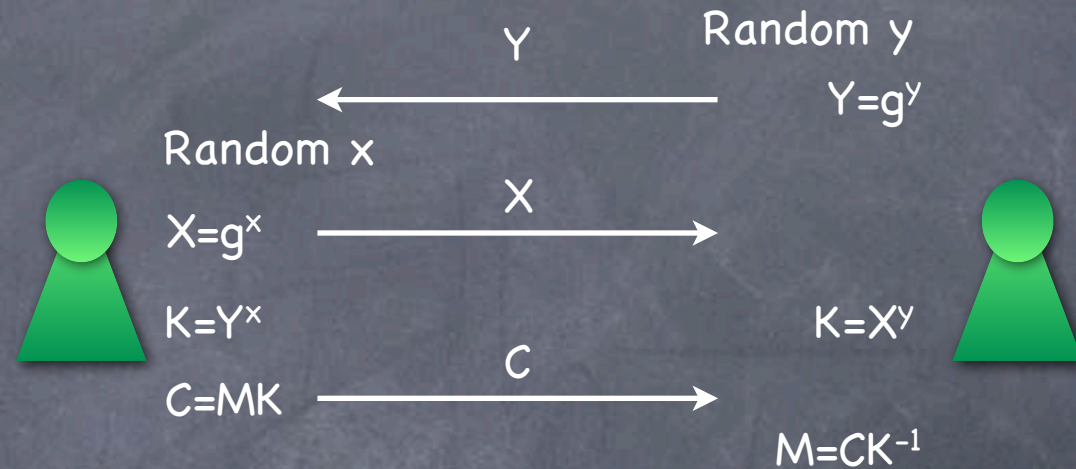
$$\text{Enc}_{(G, g, Y)}(M) = (X = g^x, C = MY^x)$$

$$\text{Dec}_{(G, g, y)}(X, C) = CX^{-y}$$

- KeyGen uses GroupGen to get (G, g)

El Gamal Encryption

- Based on DH key-exchange
 - Alice, Bob generate a key using DH key-exchange
 - Then use it as a one-time pad
- Bob's "message" in the key-exchange is his PK
- Alice's message in the key-exchange and the ciphertext of the one-time pad together form a single ciphertext



KeyGen: $PK = (G, g, Y)$, $SK = (G, g, y)$

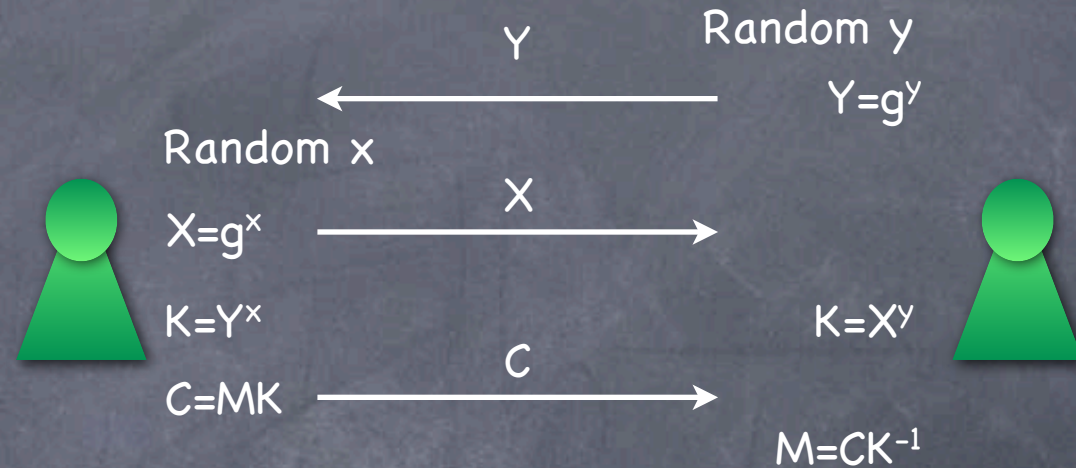
$Enc_{(G, g, Y)}(M) = (X = g^x, C = MY^x)$

$Dec_{(G, g, y)}(X, C) = CX^{-y}$

- KeyGen uses GroupGen to get (G, g)
- x, y uniform from $[|G|]$

El Gamal Encryption

- Based on DH key-exchange
 - Alice, Bob generate a key using DH key-exchange
 - Then use it as a one-time pad
- Bob's "message" in the key-exchange is his PK
- Alice's message in the key-exchange and the ciphertext of the one-time pad together form a single ciphertext



KeyGen: $PK = (G, g, Y)$, $SK = (G, g, y)$

$Enc_{(G, g, Y)}(M) = (X = g^x, C = MY^x)$

$Dec_{(G, g, y)}(X, C) = CX^{-y}$

- KeyGen uses GroupGen to get (G, g)
- x, y uniform from $[|G|]$
- Message encoded into group element, and decoded

Security of El Gamal

Security of El Gamal

- El Gamal IND-CPA secure if DDH holds (for the collection of groups used)

Security of El Gamal

- El Gamal IND-CPA secure if DDH holds (for the collection of groups used)
 - Construct a DDH adversary A^* given an IND-CPA adversary A

Security of El Gamal

- El Gamal IND-CPA secure if DDH holds (for the collection of groups used)
 - Construct a DDH adversary A^* given an IND-CPA adversary A
 - $A^*(G, g; g^x, g^y, g^z)$ (where $(G, g) \leftarrow \text{GroupGen}$, x, y random and $z = xy$ or random) plays the IND-CPA experiment with A :

Security of El Gamal

- El Gamal IND-CPA secure if DDH holds (for the collection of groups used)
 - Construct a DDH adversary A^* given an IND-CPA adversary A
 - $A^*(G, g; g^x, g^y, g^z)$ (where $(G, g) \leftarrow \text{GroupGen}$, x, y random and $z = xy$ or random) plays the IND-CPA experiment with A :
 - But sets $PK = (G, g, g^y)$ and $\text{Enc}(M_b) = (g^x, M_b g^z)$

Security of El Gamal

- El Gamal IND-CPA secure if DDH holds (for the collection of groups used)
 - Construct a DDH adversary A^* given an IND-CPA adversary A
 - $A^*(G, g; g^x, g^y, g^z)$ (where $(G, g) \leftarrow \text{GroupGen}$, x, y random and $z = xy$ or random) plays the IND-CPA experiment with A :
 - But sets $PK = (G, g, g^y)$ and $\text{Enc}(M_b) = (g^x, M_b g^z)$
 - Outputs 1 if experiment outputs 1 (i.e. if $b = b'$)

Security of El Gamal

- El Gamal IND-CPA secure if DDH holds (for the collection of groups used)
 - Construct a DDH adversary A^* given an IND-CPA adversary A
 - $A^*(G, g; g^x, g^y, g^z)$ (where $(G, g) \leftarrow \text{GroupGen}$, x, y random and $z = xy$ or random) plays the IND-CPA experiment with A :
 - But sets $PK = (G, g, g^y)$ and $\text{Enc}(M_b) = (g^x, M_b g^z)$
 - Outputs 1 if experiment outputs 1 (i.e. if $b = b'$)
 - When $z = \text{random}$, A^* outputs 1 with probability = $1/2$

Security of El Gamal

- El Gamal IND-CPA secure if DDH holds (for the collection of groups used)
 - Construct a DDH adversary A^* given an IND-CPA adversary A
 - $A^*(G, g; g^x, g^y, g^z)$ (where $(G, g) \leftarrow \text{GroupGen}$, x, y random and $z=xy$ or random) plays the IND-CPA experiment with A :
 - But sets $PK=(G, g, g^y)$ and $\text{Enc}(M_b)=(g^x, M_b g^z)$
 - Outputs 1 if experiment outputs 1 (i.e. if $b=b'$)
 - When $z=\text{random}$, A^* outputs 1 with probability = $1/2$
 - When $z=xy$, exactly IND-CPA experiment: A^* outputs 1 with probability = $1/2 + \text{advantage of } A$.

Today

Today

- Public Key Encryption
 - CPA security
 - Diffie-Hellman Key Exchange & El Gamal Encryption
 - DDH Assumption
 - Candidate group: $\mathbb{Q}\mathbb{R}_p^*$ where p is a "safe prime"

Today

- Public Key Encryption
 - CPA security
 - Diffie-Hellman Key Exchange & El Gamal Encryption
 - DDH Assumption
 - Candidate group: $\mathbb{Q}\mathbb{R}_p^*$ where P is a "safe prime"
- Next: Building CPA secure PKE, more generally.
CCA security for PKE.