

Advanced Tools from Modern Cryptography

Lecture 16

Encryption & Homomorphic Encryption

Public-Key Encryption

a.k.a. asymmetric-key encryption

- Syntax

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

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

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

- Correctness

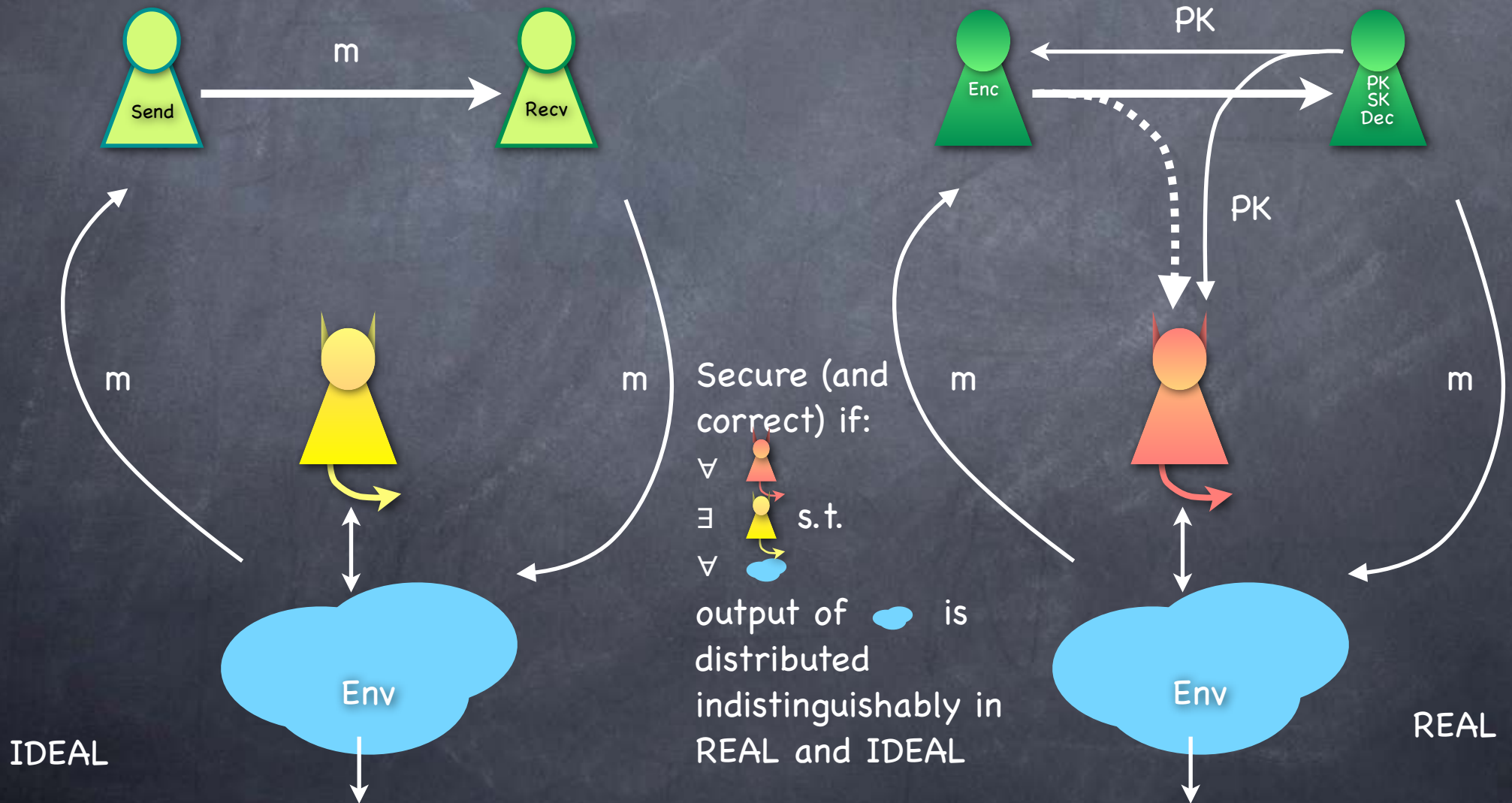
- $\forall (PK, SK) \in \text{Range}(\text{KeyGen}), \text{Dec}(\text{Enc}(m, PK), SK) = m$

- Security

- Against Chosen-Plaintext Attack: IND-CPA security

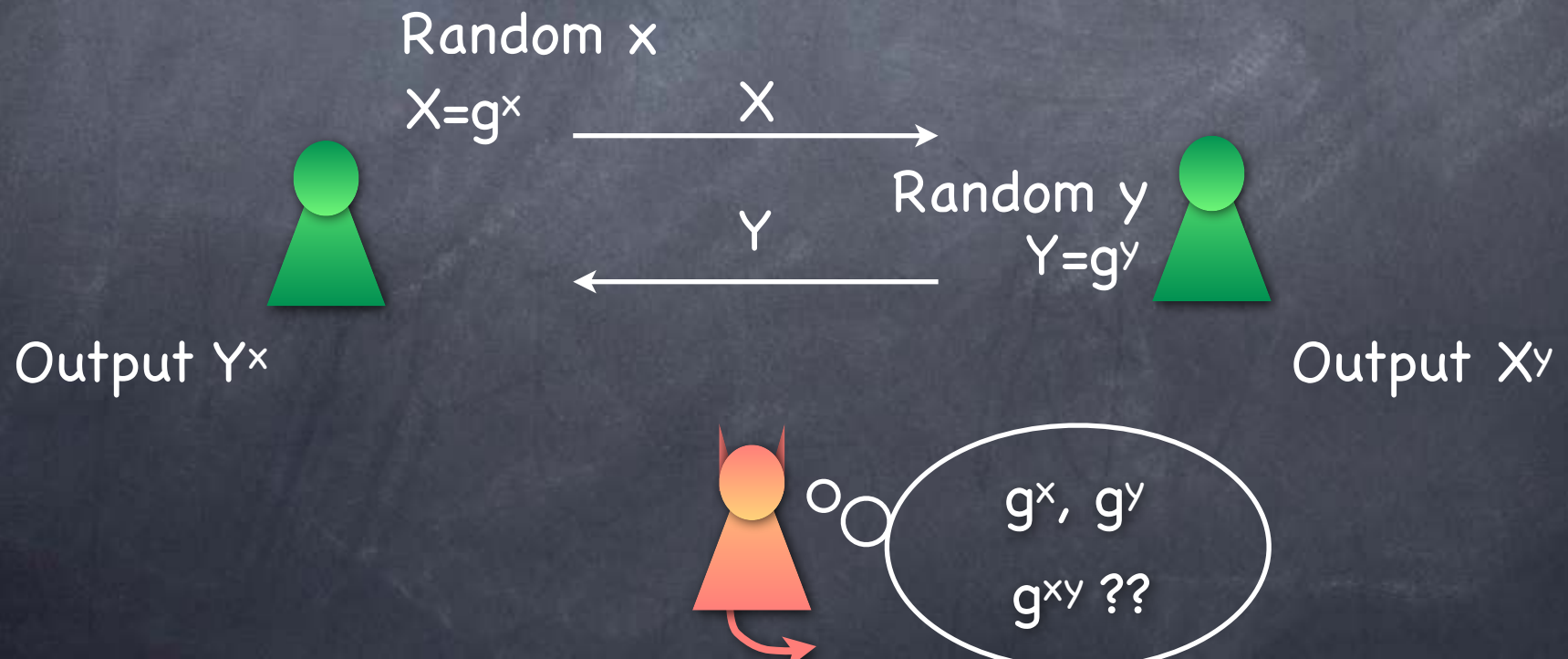
- (Stronger notions of security exist: e.g., IND-CCA security)

SIM-CPA



Diffie-Hellman Key-exchange

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

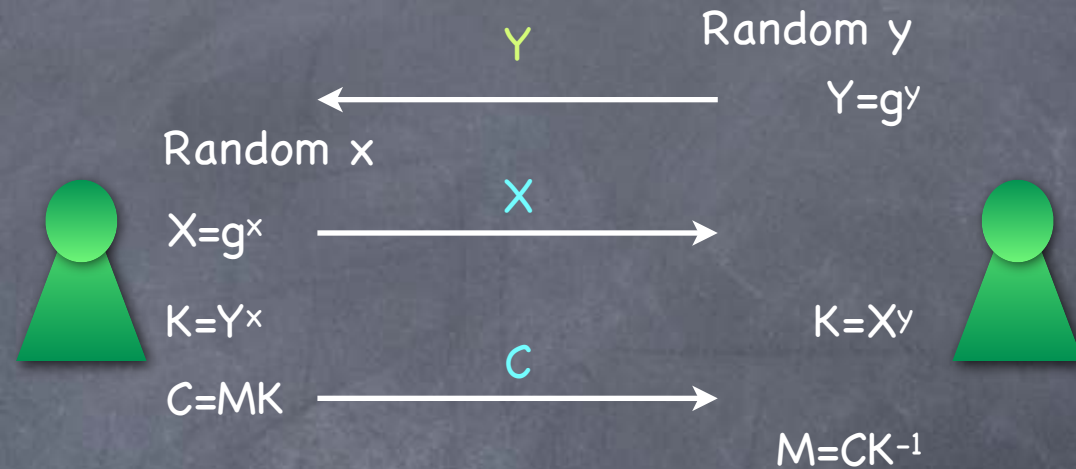


Why DH-Key-exchange could be secure

- 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?
- Decisional DH Assumption: A family of cyclic groups, with
$$\{(g^x, g^y, g^{xy})\}_{(G,g) \leftarrow \text{GroupGen}; x,y \leftarrow [|G|]} \approx \{(g^x, g^y, g^r)\}_{(G,g) \leftarrow \text{GroupGen}; x,y,r \leftarrow [|G|]}$$
where (G,g) s.t. G is generated by g (and typically $|G|$ prime, so that operations in exponent are in a field)

El Gamal Encryption

- Based on DH key-exchange
- Bob's "message" in the key-exchange is his PK
- Alice's message in the key-exchange and the message masked with this key 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

Homomorphic Encryption

- **Group Homomorphism:** Two groups G and G' are homomorphic if there exists a function (homomorphism) $f:G \rightarrow G'$ such that for all $x, y \in G$, $f(x) +_{G'} f(y) = f(x +_G y)$
- Homomorphic Encryption: A CPA secure (public-key) encryption s.t. $\text{Dec}(C) +_M \text{Dec}(D) = \text{Dec}(C +_C D)$ for ciphertexts C, D
 - i.e. $\text{Enc}(x) +_C \text{Enc}(y)$ is like $\text{Enc}(x +_M y)$
 - Interesting when $+_C$ doesn't require the decryption key
- e.g. El Gamal: $(g^{x_1}, m_1 Y^{x_1}) \times (g^{x_2}, m_2 Y^{x_2}) = (g^{x_3}, m_1 m_2 Y^{x_3})$

Rerandomization

- Often (but not always) another property is required of a homomorphic encryption scheme
- Unlinkability
 - For any two ciphertexts $c_x = \text{Enc}(x)$ and $c_y = \text{Enc}(y)$, $\text{Add}(c_x, c_y)$ should be identically distributed as $\text{Enc}(x +_M y)$.
Add is a randomized operation
 - Alternately, a ReRand operation s.t. for all valid ciphertexts c_x , $\text{ReRand}(c_x)$ is identically distributed as $\text{Enc}(x)$
 - Then, we can let $\text{Add}(c_x, c_y) = \text{ReRand}(c_x +_c c_y)$ where $+_c$ may be deterministic
 - Rerandomization useful even without homomorphism

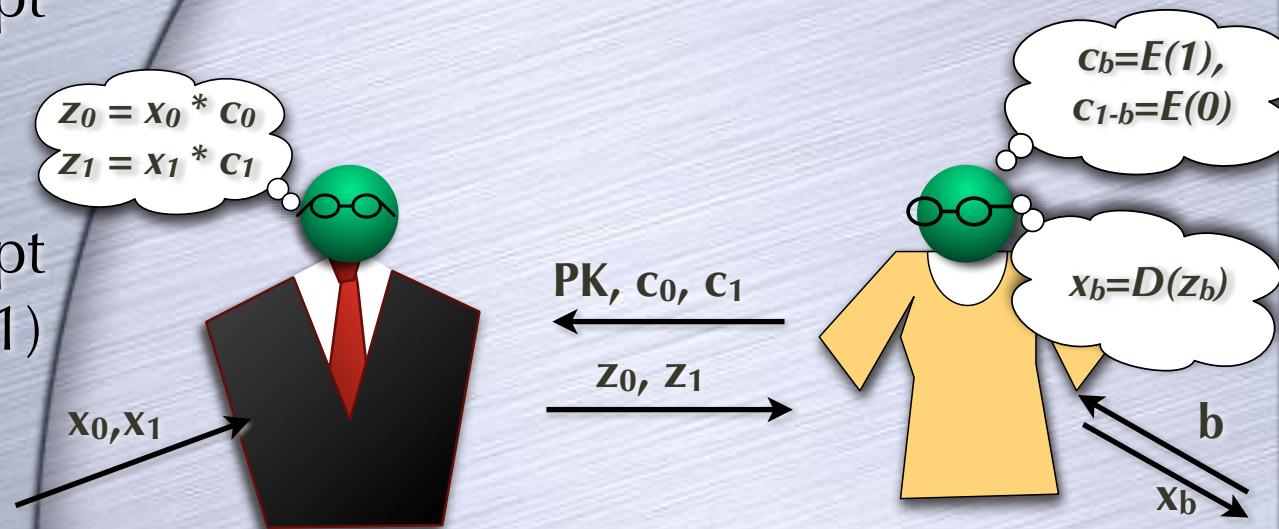
Unlinkable Homomorphic Encryption



- Considers only passive corruption
- Functionality gives “handles” to messages posted; accepts requests for posting fresh messages, or derived messages
- Unlinkability: Above, receiver gets only the message $m_1 + m_2$ in IDEAL. Even if A & Recv collude, can't tell if it is a fresh message or derived from other messages

An OT Protocol (for passive corruption)

- Using an (unlinkable) rerandomizable encryption scheme
 - Receiver picks (PK, SK) . Sends PK and $E(0), E(1)$ in suitable order
 - Sender “multiplies” c_i with x_i :
 $1 * c := \text{ReRand}(c)$, $0 * c := E(0)$
- Simulation for passive-corrupt receiver: set $z_b = E(x_b)$ and $z_{1-b} = E(0)$
- Simulation for passive-corrupt sender: set c_0, c_1 to be say $E(1)$



Homomorphic Encryption for MPC

- Recall GMW (passive-secure): each input was secret-shared among the parties, and computed on shares (using OTs for \times gates)
- Alternate approach: each wire value is kept encrypted, publicly, and the key is kept shared
 - All parties encrypt their inputs and publish
 - Evaluate each wire **using homomorphism** (coming up)
 - Finally decrypt the output wire value using **threshold decryption**
 - Threshold decryption: KeyGen protocol so that PK is public and SK shared; Decryption protocol that lets the parties decrypt a ciphertext keeping their SK shares private

Threshold El Gamal (Passive Security)

- Goal: n parties to generate a PK for El Gamal, so that SK is shared amongst them. Can decrypt messages only if all n parties come together. Will require security against **passive corruption**.
- Distributed Key-Generation:
 - $(G,g) \leftarrow \text{Groupgen}$ by Party₁ (DDH should hold for Party₁ too)
 - Each Party _{i} picks random exponent y_i and publishes $Y_i = g^{y_i}$
 - All parties compute $Y = \prod_i Y_i$. Public-key = (G,g,Y)
 - Secret-key = (G,g,y) , where $y := \sum_i y_i$ (secret). Note: $Y = g^y$
- Encryption as in El Gamal
- Distributed Decryption: Given ciphertext (X,C) , each party publishes $K_i^{-1} = X^{-y_i}$. All parties compute $K^{-1} = \prod_i K_i^{-1}$ and $M = CK^{-1}$

Homomorphic Encryption for MPC

- Passive-securely computing using homomorphism
 - Notation: Encrypted values shown as $[m]$ etc.
 - Operations available: $[x]+[y] = [x+y]$, and $a*[x] = [ax]$
 - e.g., in $GF(2)$, $0*[x] = \text{Enc}(0)$, $1*[x] = \text{ReRand}([x])$
- Addition directly, without communication
- Multiplication: All parties have $[x]$ and $[y]$. Need $[xy]$.
 - Each party P_i picks a_i, b_i and publishes $[a_i], [b_i], [a_i y], [b_i x]$
 - All compute $[x+a], [y+b], [ay], [bx]$ where $a = \sum_i a_i$ and $b = \sum_i b_i$
 - Each P_i publishes $[a_i b] = \sum_j a_i * [b_j]$, and all compute $[ab]$
 - Threshold decrypt $(x+a), (y+b)$. Compute $[z]$ where $z=(x+a)(y+b)$.
 - All compute $[xy] = [z] - [ay] - [bx] - [ab]$