CS489/689
Privacy, Cryptography, Network and Data Security

Homomorphic encryption, MPC, and PSI

Winter 2023, Tuesday/Thursday 8:30-9:50am

Computing on Ciphertexts

Consider the following:

Two ciphertexts use the same key, $c_1 = E_K(\mathbf{a})$, $c_2 = E_K(\mathbf{b})$

Let **f()** be a function that operates over plaintext **a** and **b**

Computing on Ciphertexts

Consider the following:

Two ciphertexts use the same key, $c_1 = E_K(\mathbf{a})$, $c_2 = E_K(\mathbf{b})$

Let **f()** be a function that operates over plaintext **a** and **b**

Goal: the existence of a function **g()** such that

$$g(c, d) = E_A(f(a, b))$$

Computing on Ciphertexts

Consider the following:

Two ciphertexts use the same key, $c_1 = E_K(\mathbf{a})$, $c_2 = E_K(\mathbf{b})$

Let **f()** be a function that operates over plaintext **a** and **b**

Goal: the existence of a function **g()** such that

$$g(c_1, c_2) = E_K(f(a, b))$$

g() is a **homomorphic function** on the ciphertexts c, d, …

Homomorphic Encryption in the World

- Used as a tool in many business scenarios:
  - https://www.ibm.com/security/services/homomorphic-encryption
  - https://www.statcan.gc.ca/en/data-science/network/homomorphic-encryption
  - https://www.microsoft.com/en-us/research/project/microsoft-seal/

Partial versus Fully Homomorphic Encryption

The function on the plaintexts is:

…either multiplication or addition **but not both.**

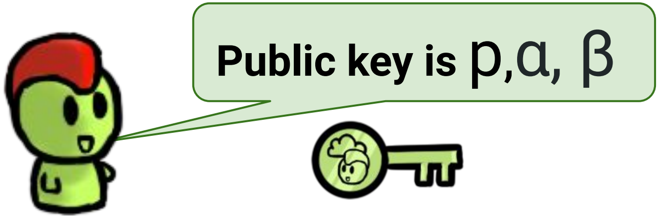
Partial HE

…either multiplication or addition **both…or even xor**

Fully HE

Recall ElGamal Public Key Cryptosystem

- Let p be a prime such that the DLP in $(\mathbf{Z}_p^*, \cdot)$ is infeasible
- Let $\alpha \in \mathbf{Z}_p^*$ be a primitive element
- Let $\mathcal{P} = \mathbf{Z}_p^*$, $\mathcal{C} = \mathbf{Z}_p^* \times \mathbf{Z}_p^*$ and…
- $\mathcal{K} = \{(p, \alpha, a, \beta) : \beta \equiv \alpha^a \pmod{p}\}$
- For a secret random number k in $\mathbf{Z}_{p-1}$ define:
  - $e_K(x,k) = (y_1, y_2)$, where $y_1 = \alpha^k \bmod p$ and $y_2 = x\beta^k \bmod p$
- For y_1, y_2 in $\mathbf{Z}_p^*$, define $d_K(y_1, y_2) = y_2(y_1^{\,a})^{-1} \bmod p$



Consider Multiplicative HE

$f(a, b) = a \cdot b$

Private key: a, **public key:** α^a

Instead of k, choose r and s

Consider Multiplicative HE

$f(a, b) = a \cdot b$

Private key: a, **public key:** α^a

Instead of k, choose r and s

$c_1 = g^r$, $c_2 = \mathbf{a}\, g^{ra}$;
$d_1 = g^s$, $d_2 = \mathbf{b}\, g^{sa}$

Consider Multiplicative HE

$f(a, b) = a \cdot b$

Private key: a, **public key:** α^a

Instead of k, choose r and s

$c_1 = g^r$, $c_2 = \mathbf{a}\, g^{ra}$;

$d_1 = g^s$, $d_2 = \mathbf{b}\, g^{sa}$

g(c, d):

- $e_1 = c_1 \cdot d_1 = g^r g^s = g^{r+s}$
- $e_2 = c_2 \cdot d_2 = ab\, g^{rx} g^{sx} = ab\, g^{x(r+s)}$

Consider Additive HE

- Multiplicative HE: Idea: encrypt a, b as g^a and g^b, respectively
- $g(E_A(g^a), E_A(g^b)) = E_A(g^{a+b})$
- Need to break discrete logarithm of g^{a+b}
  - Only works for small a, b

Note of Caution!

- Paillier's
- Simplified DGHV
- …

Some mathematical details have been omitted from the remainder of the lecture

Paillier's Encryption Scheme

- Let p, q be two large primes; N = pq

Paillier's Encryption Scheme

- Let p, q be two large primes; $N = pq$
- Ciphertexts are mod N^2

Paillier's Encryption Scheme

- Let p, q be two large primes; $N = pq$
- Ciphertexts are mod N^2
- Choose r; plaintext m (mod p) is encrypted as $g^m r^N \pmod{N^2}$

Paillier's Encryption Scheme

- Let p, q be two large primes; $N = pq$
- Ciphertexts are mod N^2
- Choose r; plaintext m (mod p) is encrypted as $g^m r^N \pmod{N^2}$
- If factorization of N is known, breaking the discrete logarithm is efficient

Paillier's Encryption Scheme

- Let p, q be two large primes; N = pq
- Ciphertexts are mod N^2
- Choose r; plaintext m (mod p) is encrypted as $g^m\, r^N \pmod{N^2}$
- If factorization of N is known, breaking the DL is efficient

⇒ Efficient additive HE for large numbers

$$D(E(m_1,r_1) \cdot E(m_2,r_2) \bmod n^2) = m_1 + m_2 \bmod n$$

Product to addition

Paillier's Encryption Scheme

- Let p, q be two large primes; N = pq
- Ciphertexts are mod N^2
- Choose r; plaintext m (mod p) is encrypted as $g^m r^N \pmod{N^2}$
- If factorization of N is known, breaking the DL is efficient

⇒ Efficient additive HE fo

$$D(E(m_1, r_1) \cdot E(m_2, r_2) \bmod n^2) = m_1 + m_2 \bmod n$$

$$D(E(m_1, r_1) \cdot g^{m_2} \bmod n^2) = m_1 + m_2 \bmod n$$

Fully HE

- Need to encrypt message m in the base, then both operations work
- Many schemes now, usually abbreviated by the first letters of the last names of the authors
- Different security assumptions (not factoring or discrete log)
  - Lattice problems: Learning with errors, …

Examples:

- First construction by Gentry in 2009
- E.g. FV, BGV, or DGHV (not used in practice)

Consider Simplified DGHV (not used in practice)

- $m \in \{0, 1\}$
- Protocols is over the integers
- Secret key: prime p

Consider Simplified DGHV (not used in practice)

- $m \in \{0, 1\}$
- Protocols is over the integers
- Secret key: prime p
- **Encryption**
  - Choose q, r; $r < p$
  - $c = qp + 2r + m$

Consider Simplified DGHV (not used in practice)

- $m \in \{0, 1\}$
- Protocols is over the integers
- Secret key: prime p
- **Encryption**
  - Choose q, r; $r < p$
  - $c = qp + 2r + m$
- **Decryption**
  - $m = c \bmod 2 \oplus (\lfloor c/p \rfloor \bmod 2)$

Computing with Simplified DGHV

- Ciphertexts
  - $c_1 = q_1 p + 2r_1 + a$
  - $c_2 = q_2 p + 2r_2 + b$

Computing with Simplified DGHV

- Ciphertexts
  - $c_1 = q_1 p + 2r_1 + a$
  - $c_2 = q_2 p + 2r_2 + b$
- Addition
  - $c_1 + c_2 = (q_1+q_2)p + 2(r_1+r_2) + a + b$

Computing with Simplified DGHV

- Ciphertexts
  - $c_1 = q_1 p + 2r_1 + a$
  - $c_2 = q_2 p + 2r_2 + b$
- Addition
  - $c_1 + c_2 = (q_1+q_2)p + 2(r_1+r_2) + a + b$
- Multiplication
  - $c_1 \cdot c_2 = ??? = q'p + 2r' + ab$
  - $r' = 2r_1 r_2 + r_1 b + r_2 a$
    - Note the increase in length!!

Bootstrapping…in Fully HE Schemes

- $r > p/2 \Rightarrow$ decryption fails

Bootstrapping…in Fully HE Schemes

- $r > p/2 \Rightarrow$ decryption fails
- Multiplication quickly increases noise (doubles the length)

Bootstrapping...in Fully HE Schemes

- $r > p/2 \Rightarrow$ decryption fails
- Multiplication quickly increases noise (doubles the length)
- Bootstrapping is a procedure that reduces the noise to its initial length
  - Bootstrapping is slow in most fully HE schemes
    - DGHV does not have bootstrapping
  - When using fully HE, it is therefore important to reduce the number of subsequent multiplications

Practical Used FHE

- FV, BGV, BFV, CKKS
  - Lattice-based encryption schemes
  - Encrypt vectors (usually as polynomials)
- TFHE
  - Fully HE over the Torus
  - Usually encrypts bits
  - Very fast bootstrapping (frequently performed)

Try it…

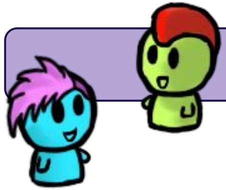
- Download Microsoft's SEAL library
  - https://www.microsoft.com/en-us/research/project/microsoft-seal/
- Create a key
- Encrypt two 8 bit numbers bit-wise using batch encoding (allows rotation)
- Perform comparison, for each position: If prefix is equal and bits are different, output 1 if bit of first number is 1; else output 0
- Decrypt result

Try it…on your own.

- Download Microsoft's SEAL library
  - https://www.microsoft.com/en-us/research/project/microsoft-seal/
- Create a key
- Encrypt two 8 bit numbers bit-wise using batch encoding (allows rotation)
- Perform comparison, for each position: If prefix is equal and bits are different, output 1 if bit of first number is 1; else output 0
- Decrypt result

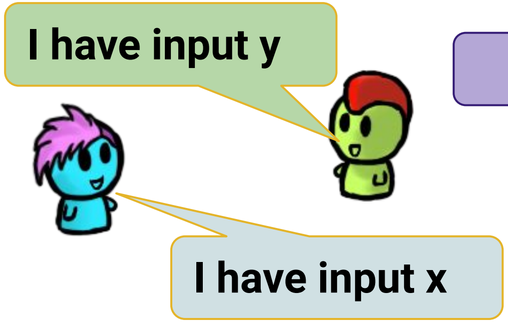
Multi-Party Computation (MPC)

What is Multi-Party Computation?



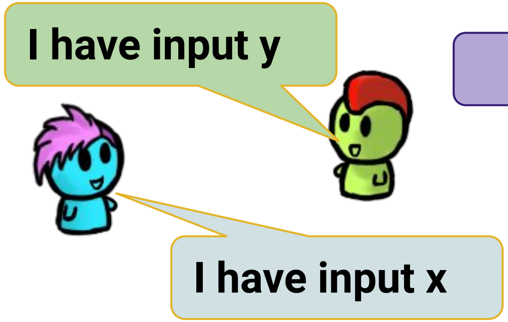
1) **At least two parties**

What is Multi-Party Computation?



1) **At least two parties**

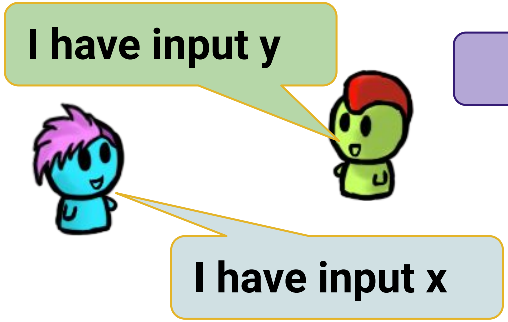
What is Multi-Party Computation?



1) At least two parties

2) Both Alice and Bob know a function f

What is Multi-Party Computation?

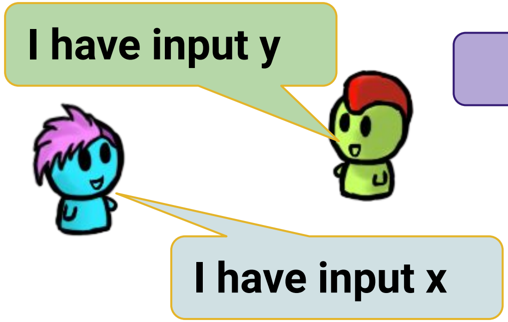


1) At least two parties

2) Both Alice and Bob know a function f

Goal: learn f(x, y) but not reveal anything else about x or y

What is Multi-Party Computation?



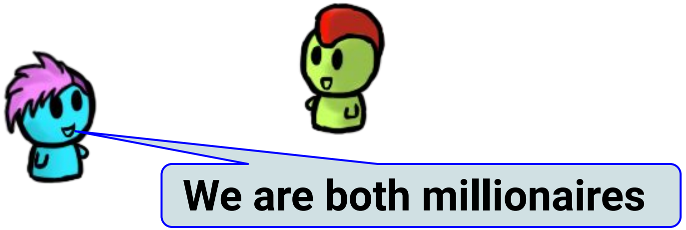
1) At least two parties

2) Both Alice and Bob know a function f

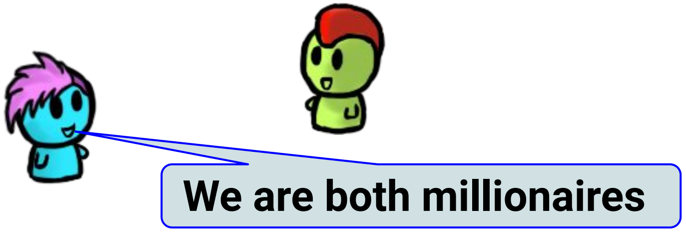
Goal: learn f(x, y) but not reveal anything else about x or y

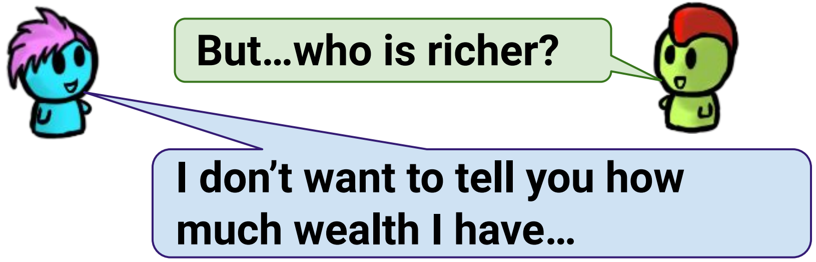
Critical: Secret inputs, public outputs (to at least one party)

Toy Example, Basically "Millionaire's Problem"



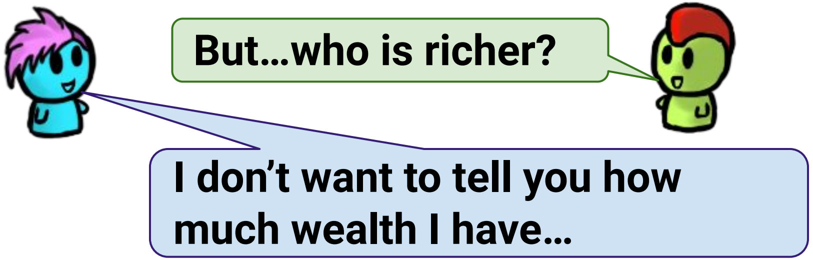
Toy Example, “The Millionaire's Problem”





Toy Example, “The Millionaire's Problem”

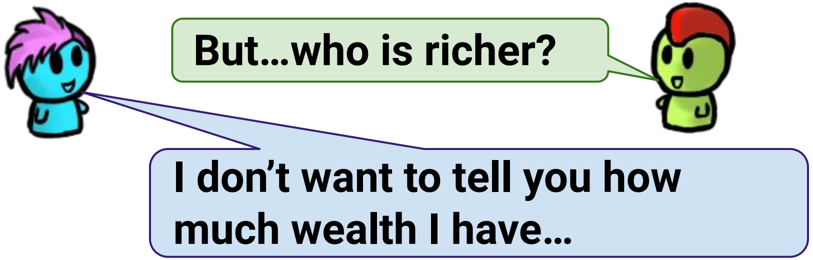
Act.



Q: how can Bob and Alice determine who is richer?

Toy Example, “The Millionaire's Problem”

Act.



Q: how can Bob and Alice determine who is richer?

A: A multi-party computation to compute the f: x < y

Fun Facts:

- “Yao’s millionaires’ problem” (Andrew C. Yao, Turing Award 2000)

Exponential Solution

Let E_a be Alice's public key. Alice has i billions, Bob has j millions, such that $1 < i, j < 10$.

1. Bob picks a random N-bit integer, and computes privately the value of $E_a(x)$; call the result k.

2. Bob sends Alice the number $k - j + 1$

3. Alice computes privately the values of $y_u = D_a(k - j + u)$ for $u = 1, 2, \ldots, 10$.

Exponential Solution Con't

4. Alice generates a random prime p of N/2 bits, and computes the values $\mathbf{z_u = y_u \ (mod\ p)}$ for all u; if all z_u differ by at least 2 in the mod p sense, stop; otherwise generates another random prime and repeat the process until all z_u differ by at least 2; let p, z_u denote this final set of numbers;

5. Alice sends the prime p and the following 10 numbers to B: $z_1, z_2, \ldots, z_i$ followed by $z_i + 1, z_{i+1} + 1, \ldots, z_{10} + 1$; the above numbers should be interpreted in the mod p sense.

Exponential Solution More Con’t

6. Bob looks at the j-th number (not counting p) sent from Alice, and decides that $i \geq j$ if it is equal to x mod p, and $i < j$ otherwise.

7. Bob tells Alice what the conclusion is.

Why Exponential Solution Works?

Q: Can anyone identify a reason it would fail?

Why Exponential Solution Works?

Q: Can anyone identify a reason it would fail?

Q: What does Alice know?

Q: What does Bob know?

Why Exponential Solution Works?

Q: Can anyone identify a reason it would fail?

Q: What does Alice know?

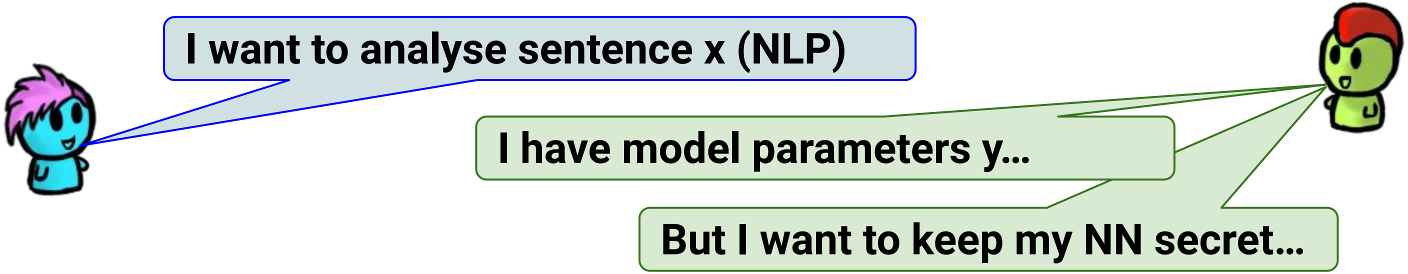
Q: What does Bob know?

Short A: Other than lies…no.

“Real-World” Example

I want to analyse sentence x (NLP)

“Real-World” Example



“Real-World” Example

I want to analyse sentence x (NLP)

I have model parameters y…

But I want to keep my NN secret…

Require: A function f over public parameters, but secret architecture

Goal: A MPC for f(x, y) such that only Alice learns the analysis of her sentence and Alice does not learn the NN

“Types” of MPC: Participant Set

Two-party

Multi-Party

MPC Server Model

- Assume n >> 3 clients with an input
  - Example collect statistics about emoji usage in texting

MPC Server Model

- Assume n >> 3 clients with an input
  - Example collect statistics about emoji usage in texting
- Dedicate 3 (or 2) parties as computation nodes (servers)

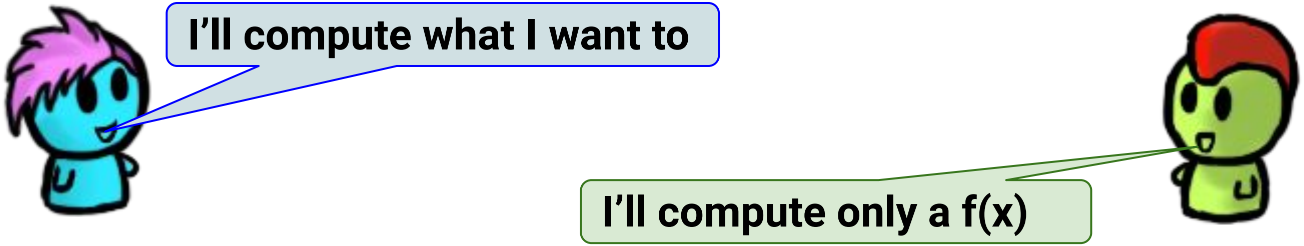
MPC Server Model

- Assume n >> 3 clients with an input
    - Example collect statistics about emoji usage in texting
- Dedicate 3 (or 2) parties as computation nodes (servers)
- The clients send “encrypted” versions of their inputs

MPC Server Model

- Assume n >> 3 clients with an input
  - Example collect statistics about emoji usage in texting
- Dedicate 3 (or 2) parties as computation nodes (servers)
- The clients send “encrypted” versions of their inputs
- The servers perform multi-party computation
  - Decrypt input
  - Compute f

“Types” of MPC: Functionality



Generic

Specific

Generic / Specific Functions, in more words

- Specific functions:

A multi-party computation protocol that can only be used for **a specific function f**

- Generic functions:

A multi-party computation protocol that can be used for **“any” function f**

“Types” of MPC: Security

I’m beyond curious

Active

I’m just curious

Passive

Passive Security

- Passive security
  (also called security against **semi-honest adversaries**)

  Each party **follows the protocol** but keeps a record of all messages and after the protocol is over, **tries to infer additional information** about the other parties’ inputs

Active Security

- Active security

  (also called security against **malicious adversaries**)

  Each party **may arbitrarily deviate from the protocol**. Either the protocol computes f or the protocol is aborted.

Relation of Passive and Active Security

- Passive security is a **prerequisite** for active security

Q: What does a prerequisite here mean?

Relation of Passive and Active Security

- Passive security is a **prerequisite** for active security
  - A protocol can be secure against passive adversaries but not active ones
  - A protocol secure against active adversaries is also secure against passive adversaries

Relation of Passive and Active Security

- Passive security is a **prerequisite** for active security
  - A protocol can be secure against passive adversaries but not active ones
  - A protocol secure against active adversaries is also secure against passive adversaries
- Any protocol secure against passive adversaries can be turned into a protocol secure actives adversaries

Q: Suggestions?

Known as Goldreich's compiler (Oded Goldreich, Knuth Prize 2017)

Relation of Passive and Active Security

- Passive security is a **prerequisite** for active security
  - A protocol can be secure against passive adversaries but not active ones
  - A protocol secure against active adversaries is also secure against passive adversaries
- Any protocol secure against passive adversaries can be turned into a protocol secure actives adversaries
  - Adding additional protocol steps proving the correct computation of each message

Known as Goldreich's compiler (Oded Goldreich, Knuth Prize 2017)