Section 2

Defining FHE
Public Key Encryption

\[ \text{PKE}(\text{Gen}, \text{Enc}, \text{Dec}) \]
\[ \text{Gen}: () \rightarrow (\text{pk}, \text{sk}) \]
\[ \text{Enc}: (\text{pk}, m) \rightarrow c \]
\[ \text{Dec}: (\text{sk}, c) \rightarrow m \]
Correctness of PKE

For every $(sk, pk) \leftarrow \text{Gen}()$ and $m \leftarrow [M]$, $r \leftarrow [R]$:

$$\text{Dec}(sk, \text{Enc}(pk, m; r)) = m$$
Chosen Plaintext Attack (CPA) security

- Ciphertext Indistinguishability under Chosen Plaintext Attack
- Experiment:

  \[
  \text{INDCPA}_{\text{game}}(b : \{0, 1\})
  \]
  \[
  (sk, pk) \leftarrow \text{Gen}()
  \]
  \[
  A(pk) \rightarrow (m_0, m_1)
  \]
  \[
  b' \leftarrow A(\text{Enc}(pk, m_b))
  \]
  \[
  \text{return } b' : \{0, 1\}
  \]
Chosen Plaintext Attack (CPA) security

- Ciphertext Indistinguishability under Chosen Plaintext Attack
- Experiment:

\[
\text{INDCPA}_{\text{game}}(b:\{0,1\})
\]
\[
\text{(sk, pk) } \leftarrow \text{Gen()}
\]
\[
A(pk) \rightarrow (m_0, m_1)
\]
\[
b' \leftarrow A(\text{Enc}(pk, m_b))
\]
\[
\text{return } b':\{0,1\}
\]

Definition

\[
\text{Adv}(A) = |\Pr(\text{Game}(0)=1) - \Pr(\text{Game}(1)=1)|
\]

Definition

An encryption scheme \((\text{Gen}, \text{Enc}, \text{Dec})\) is \textbf{IND-CPA} secure if any polynomial time \(A\) has advantage \(\text{Adv}(A) \sim 0\)
Significance of CPA security

- Adversary can choose messages $m_0$, $m_1$
  - No assumption about input distribution
  - Adversary may have partial information about messages
  - Adversary may influence the choice of messages

- Ciphertext $c = \text{Enc}(pk, m_b)$ is computed honestly
  - Adversary cannot tamper with ciphertexts

- Adversary models a passive attacker
Definition of CCA security

**Definition**

An encryption scheme \((\text{Gen}, \text{Enc}, \text{Dec})\) is **IND-CCA** secure if any polynomial time \(A\) has advantage \(\text{Adv}(A) \approx 0\) in the following game.

\[
\text{Game}(b:\{0,1\})
\]

\[
\begin{align*}
(\text{sk}, \text{pk}) & \leftarrow \text{Gen}() \\
A[D](\text{pk}) & \rightarrow (m_0, m_1) \\
c & \leftarrow \text{Enc}(\text{pk}, m_b) \\
b' & \leftarrow A[D'][c] \\
\text{return} & \quad b':\{0,1\}
\end{align*}
\]

- \(A[D]\) is an adversary with oracle access to
  \(D(x) = \text{Dec}(\text{sk}, x)\)
- \(A[D']\) uses a modified oracle (next slide)
IND-CCA1 vs IND-CCA2

There are two variants of CCA security, depending on the type of oracle given to the adversary after receiving the challenge ciphertext:

- **IND-CCA1** security: No decryption oracle after receiving the challenge
  
  \[ D'(x) = \text{Nil} \]

- **IND-CCA2** security: decrypt any ciphertext, except the challenge \( c \)
  
  \[ D'(x) = \begin{cases} 
  \text{Nil} & \text{if } (x \neq c) \\
  \text{Dec}(sk, x) & \text{else} 
  \end{cases} \]
Significance of CCA security

- Goal: model active attacks, where adversary can tamper with ciphertexts
- Standard notion for regular encryption schemes
- IND-CCA2 theoretically equivalent to non-malleable encryption
  - Any attempt to modify a ciphertext should be detected
Significance of CCA security

- Goal: model active attacks, where adversary can tamper with ciphertexts
- Standard notion for regular encryption schemes
- IND-CCA2 theoretically equivalent to non-malleable encryption
  - Any attempt to modify a ciphertext should be detected
- Seems incompatible with homomorphic encryption
  - Ability to modify ciphertexts can be a useful feature
  - Homomorphic encryption is perfectly malleable
- We will not consider CCA security
Homomorphic Encryption: first attempt

- Assume $f: M \rightarrow M$
  
  $$f(\text{Enc}(pk,m)) = \text{Enc}(pk,f(m))$$

  $$\text{Eval}(pk,f,\text{Enc}(pk,m)) = \text{Enc}(pk,f(m))$$
Homomorphic Encryption: second attempt

\[ \text{Dec}(sk, \text{Eval}(pk, f, \text{Enc}(pk, m))) = f(m) \]
Multi-input functions

- Many inputs are encrypted independently

\[
c_1 \leftarrow \text{Enc} (pk , m_1) \\
... \\
c_k \leftarrow \text{Enc} (pk , m_k)
\]
Multi-input functions

- Many inputs are encrypted independently
  
  \[ \begin{align*}
  c_1 & \leftarrow \text{Enc}(pk, m_1) \\
  \ldots \\
  c_k & \leftarrow \text{Enc}(pk, m_k)
  \end{align*} \]

- \( k \)-ary function \( f: (m_1, \ldots, m_k) \rightarrow m \)
  
  \[ \begin{align*}
  \text{Eval}(pk, f, c_1, \ldots, c_k) \\
  & = \text{Enc}(pk, f(m_1, \ldots, m_k)) \ ?
  \end{align*} \]

  \[ \begin{align*}
  \text{Dec}(sk, \text{Eval}(pk, f, c_1, \ldots, c_k)) \\
  & = f(m_1, \ldots, m_k)
  \end{align*} \]
Multi-key Homomorphic encryption

- Assume multiple users: $P_1, P_2, \ldots$
- Each user has a key (pair): $P_i : (pk_i, sk_i)$
- Data is encrypted and sent to different users

$$c_1 \leftarrow \text{Enc}(pk_1, m_1)$$
$$\ldots$$
$$c_t \leftarrow \text{Enc}(pk_t, m_t)$$

- Users pool data together to perform a joint computation on $c_1, \ldots, c_t$
Multi-key Homomorphic encryption

- Assume multiple users: $P_1, P_2, ...$
- Each user has a key (pair): $P_i : (pk_i, sk_i)$
- Data is encrypted and sent to different users
  
  $c_1 \leftarrow \text{Enc}(pk_1, m_1)$
  
  $\ldots$
  
  $c_t \leftarrow \text{Enc}(pk_t, m_t)$
  
- Users pool data together to perform a joint computation on $c_1, \ldots, c_t$

- Final result is an encryption of $f(m_1, \ldots, m_t)$ under what key?
  
  $\text{Eval}(\ldots, f, c_1, \ldots, c_t)$
  
  $\sim \text{Enc}(\ldots, f(m_1, \ldots, m_t))$
FHE is a useful and challenging problem already in the single key setting.
Restricting Homomorphic Encryption

- FHE is a useful and challenging problem already in the single key setting
- In order to approach the problem we will further restrict it by parametrizing by a set of allowed computations/functions $\text{Func} = \{ f: \ldots \}$ where each $f: (M, \ldots, M) \to M$ may take a different number of arguments
Restricting Homomorphic Encryption

- FHE is a useful and challenging problem already in the single key setting.

- In order to approach the problem we will further restrict it by parametrizing by a set of allowed computations/functions $\text{Func} = \{f: \ldots \}$ where each $f: (M, \ldots, M) \rightarrow M$ may take a different number of arguments.

- More generally, one may consider functions $f: (M_1, \ldots, M_k) \rightarrow M$ taking inputs from different sets (types), e.g., $\text{ifThenElse}: (\text{Bool}, \text{Int}, \text{Int}) \rightarrow \text{Int}$.
Examples and Function Composition

- \((M, +, 0)\): abelian group, e.g., “fixed size” integers (modulo \(N\))
- Addition: \(f(x_1, \ldots x_t) = x_1 + \ldots + x_t\)
- Scalar multiplication: \(g_a(x) = a \cdot x\)
- Linear combinations: \(h(x_1, \ldots x_t) = \sum_i 2^{i-1} x_i\)
Examples and Function Composition

- $(M, +, 0)$: abelian group, e.g., “fixed size” integers (modulo $N$)
- Addition: $f(x_1, \ldots, x_t) = x_1 + \ldots + x_t$
- Scalar multiplication: $g_a(x) = a \cdot x$
- Linear combinations: $h(x_1, \ldots, x_t) = \sum_i 2^{i-1} x_i$

1-hop, n-hop, multi-hop: can functions $f$ be composed?

$$h(x_1, \ldots, x_t) = f(g_1(x_1), \ldots, g_{2^t-1}(x_t))$$
Correctness of Function Composition

Let $x, y, z \in M$ be messages and $f, g : M \rightarrow M$ two functions such that $y = f(x)$ and $z = g(y) = (g \circ f)(x)$.

Assume $(Gen, Enc, Dec, Eval)$ can evaluate $f$ and $g$ correctly:

- $Dec(sk, Eval(pk, f, Enc(pk, x))) = f(x)$
- $Dec(sk, Eval(pk, g, Enc(pk, y))) = g(y)$

Does it follow that $ct_X \leftarrow Enc(pk, x)$ $ct_Y \leftarrow Eval(pk, f, ct_X)$ $ct_Z \leftarrow Eval(pk, g, ct_Y)$ $Dec(sk, ct_Z)$?
Correctness of Function Composition

Let $x, y, z \in M$ be messages and $f, g : M \rightarrow M$ two functions such that $y = f(x)$ and $z = g(y) = (g \circ f)(x)$.

Assume $(Gen, Enc, Dec, Eval)$ can evaluate $f$ and $g$ correctly:

1. $Dec(sk, Eval(pk, f, Enc(pk, x))) = f(x)$
2. $Dec(sk, Eval(pk, g, Enc(pk, y))) = g(y)$

Question

Does it follow that

$$ctX \leftarrow Enc(pk, x)$$
$$ctY \leftarrow Eval(pk, f, ctX)$$
$$ctZ \leftarrow Eval(pk, g, ctY)$$
$$Dec(sk, ctZ) \overset{?}{=} z$$
Formalizing Restricted Composition

- Restrict scheme to a set $F$ of strongly typed functions:
  \[ f : M_1 \times \ldots \times M_k \rightarrow M_0 \]

- $\text{Enc}, \text{Dec}, \text{Eval}$ are given type information
Formalizing Restricted Composition

- Restrict scheme to a set $\mathcal{F}$ of strongly typed functions:
  $$f : M_1 \times \ldots M_k \rightarrow M_0$$

- $\text{Enc, Dec, Eval}$ are given type information

- We can use types to bound computation depth:
  - Start from $f : M \rightarrow M$
  - Define $M_i = M$ for $i = 1, \ldots, n$
  - Define $f_i : M_i \rightarrow M_{i+1}$, where $f_i(x) = f(x)$

- $\mathcal{F} = \{f\}$ allows arbitrary composition

- $\mathcal{F} = \{f_0\}$: no composition

- $\mathcal{F} = \{f_0, f_1, \ldots, f_n\}$: bounded depth composition
(Multi-hop) Correctness Game

- State: (initially empty) list $L$ of message-ciphertext pairs

\[
\text{CorrectFHEgame}() = (sk, pk) \leftarrow \text{Gen}()
L \leftarrow []
A[E,F](pk)
(m, c) \leftarrow \text{last}(L)
\text{return } (\text{Dec}(sk, c) \neq m)
\]

\[
E(m) = c \leftarrow \text{Enc}(pk, m)
L \leftarrow L;(m, c)
\text{return } c
\]

\[
F(f, I) = (ms, cs) \leftarrow \text{unzip } L[I]
m \leftarrow f(ms)
c \leftarrow \text{Eval}(pk,f,cs)
L \leftarrow L;(m, c)
\text{return } c
\]
Terminology

Reading papers, you will find references to

- Fully Homomorphic Encryption
- Somewhat Homomorphic Encryption
- Leveled Fully Homomorphic Encryption
- etc.
Terminology

Reading papers, you will find references to

- Fully Homomorphic Encryption
- Somewhat Homomorphic Encryption
- Leveled Fully Homomorphic Encryption
- etc.

We will use FHE as a catchall term

- Definition is parametrized by a set of functions $\mathcal{F}$
- Functions in $\mathcal{F}$ can be composed only if their types match
- $\mathcal{F}$ is closed under composition
- Can use “phantom” types to limit composition

We will rarely define $\mathcal{F}$ formally, but it is a useful exercise
Security of Homomorphic Encryption

\[
\text{INDCPA}\text{game}(b:\{0,1\}) \\
(sk,pk) \leftarrow \text{Gen}() \\
A(pk) \rightarrow (m_0,m_1) \\
\text{return } A(\text{Enc}(pk,m_b)) : \{0,1\}
\]

Remark

*The IND-CPA security definition depends only on Gen and Enc, but not on Dec (or Eval).*

Question

*Can the IND-CPA security definition be applied as it is to FHE schemes \((\text{Gen}, \text{Enc}, \text{Dec}, \text{Eval})\)?
A trivial FHE scheme

Consider the following FHE scheme:

- Let \((\text{Gen}, \text{Enc}, \text{Dec})\) be IND-CPA secure
- Define \(\text{TrivialFHE} = (\text{Gen}, \text{Enc}', \text{Dec}', \text{Eval})\)

\[
\begin{align*}
\text{Enc}'(pk, m) &= (\text{Enc}(pk, m), []) \\
\text{Dec}'(sk, (ct, [])) &= \text{Dec}(sk, ct) \\
\text{Dec}'(sk, (ct, [f; fs])) &= f(\text{Dec}'(sk, (ct, fs))) \\
\text{Eval}(pk, f, (ct, [fs])) &= (ct, [f; fs])
\end{align*}
\]

**Question**

- Is \(\text{TrivialFHE}\) a correct FHE scheme?
- Is \(\text{TrivialFHE}\) a secure FHE scheme?
- What makes the above scheme “trivial”?
Compactness

- The TrivialFHE scheme is both correct and secure.
- The problem with TrivialFHE is that it is not efficient:
  - Computation is performed by Dec, not Eval!

**Definition**

A FHE scheme is **compact** if the size of ciphertext $ct = \text{Eval}(pk,f,\text{Enc}(pk,m))$ is independent of $\text{Size}(f)$.

- Weaker forms of compactness:
  - Ciphertext size may grow logarithmic with $\text{Size}(f)$
  - Ciphertext size may depend on $\text{Depth}(f)$


\[ f_0(x, y) = x + y \]
\[ f_1(x, y) = y + x \]

Game[A](b: \{0, 1\})
\[
(\text{sk}, \text{pk}) \leftarrow \text{Gen}()
\]
\[
\text{ctX} \leftarrow \text{Enc}(\text{pk}, x)
\]
\[
\text{ctY} \leftarrow \text{Enc}(\text{pk}, y)
\]
\[
\text{ct} \leftarrow \text{Eval}(\text{pk}, f_b, \text{ctX}, \text{ctY})
\]
\[
\text{return } A(\text{ct})
\]

**Question**

*Assume \((\text{Gen}, \text{Enc}, \text{Dec}, \text{Eval})\) is a secure FHE scheme. Can an efficient adversary \(A\) recover the bit \(b = \text{Game}[A](b)\)?
Passive Attacks to FHE

Game[A](b: \{0, 1\})

(\text{sk}, \text{pk}) \leftarrow \text{Gen}()

\text{State} \leftarrow []

b' \leftarrow A[E,D,F](\text{pk})

\text{return } b'

Adversary has access to three stateful oracles:

- Encryption oracle: \text{E}(m_0, m_1)
- Function Evaluation oracle: \text{F}(f_0, f_1, I)
- Decryption oracle: \text{D}(i)
- Joint State: List of message-message-ciphertext triplets
  \((m_0, m_1, ct)\)
Passive Attack (oracles)

\[
E(m_0, m_1) = \text{ct} \leftarrow \text{Enc}(pk, m_b)
\]
\[
\text{State} \leftarrow (\text{State};(m_0, m_1, \text{ct}))
\]
\[
\text{return ct}
\]

\[
F(f_0, f_1, I) = (ms_0, ms_1, cts) \leftarrow \text{unzip State}[I]
\]
\[
\text{ct} \leftarrow \text{Eval}(pk, f_b, cts)
\]
\[
m_0 \leftarrow f_0(ms_0)
\]
\[
m_1 \leftarrow f_1(ms_1)
\]
\[
\text{State} \leftarrow \text{State};(m_0, m_1, \text{ct})
\]
\[
\text{return ct}
\]

\[
D(i): (m_0, m_1, \text{ct}) \leftarrow \text{State}[i]
\]
\[
\text{if } (m_0 \equiv m_1)
\]
\[
\text{then return Dec}(sk, \text{ct})
\]
\[
\text{else return Nil}
\]
The game we just described guarantees function privacy.
A similar definition without function privacy can be obtained by requiring $f_0 \equiv f_1$ in the function evaluation queries.

\[
\begin{align*}
F'(f, I): (ms_0, ms_1, cts) & \leftarrow \texttt{unzip} \ State[I] \\
ct & \leftarrow \texttt{Eval}(pk, f, cts) \\
m_0 &= f(ms_0) \\
m_1 &= f(ms_1) \\
\text{State} & \leftarrow (\text{State}; (m_0, m_1, ct)) \\
\text{return } ct
\end{align*}
\]
Example: Circuit Privacy

- Assume messages are single bits \( m: \{0, 1\} \)
- Let \( \text{FHE} = (\text{Gen, Enc, Dec, Eval}) \) a function private FHE scheme supporting \( \text{NAND}(x, y) = \text{not } (x \&\& y) \)
- \( \text{Eval}C(pk, C, \ldots) \): evaluates boolean circuit \( C: \{0, 1\}^n \rightarrow \{0, 1\} \) one gate at a time using \( \text{Eval}(pk, \text{NAND}, \ldots) \)
- Let \( C_0, C_1 : \text{NAND} \) circuits with the same number of inputs and NAND gates
- \( (sk, ps) \leftarrow \text{Gen}() \)
- Let \( xs_0, xs_1 \) be input bits such that \( C_0(xs_0) = C_1(xs_1) \)

Question

Are the following two distributions indistinguishable?

\[
(pk, \text{Eval}C(pk, C_0, \text{Enc}(pk, xs_0)))
\]
\[
(pk, \text{Eval}C(pk, C_1, \text{Enc}(pk, xs_1)))
\]