Section 3

Bootstrapping
Bootstrapping

- For simplicity: fix message space to \{0, 1\}
- HE=(Gen, Enc, Dec, Eval)
  - Homomorphic functions: Func = \{ \text{nand} \}
  - Supports only bounded computations: Depth(C) < D
Bootstrapping

- For simplicity: fix message space to \{0, 1\}

- HE=(Gen, Enc, Dec, Eval)
  - Homomorphic functions: Func = \{ nand \}
  - Supports only bounded computations: Depth(C) < D

**Question**

*Can we use HE to build a FHE scheme supporting arbitrary circuits/functions?*

- The process of building FHE from HE is called “bootstrapping”
Decryption as a boolean function

- Everything is a sequence of bits
  - Secret key $sk: \{0, 1\}^k$
  - Ciphertext $ct: \{0, 1\}^l$
- $Dec(sk, ct): \{0, 1\}$
Decryption as a boolean function

- Everything is a sequence of bits
  - Secret key $sk: \{0, 1\}^k$
  - Ciphertext $ct: \{0, 1\}^l$

- $Dec(sk, ct): \{0, 1\}$

- Usually we think of $Dec$ as a function
  - described by secret key $sk$
  - mapping ciphertext $ct$ to message bit $Dec(sk, ct): \{0, 1\}$
Decryption as a boolean function

- Everything is a sequence of bits
  - Secret key $sk: \{0, 1\}^k$
  - Ciphertext $ct: \{0, 1\}^l$

- $Dec(sk, ct): \{0, 1\}$

- Usually we think of $Dec$ as a function
  - described by secret key $sk$
  - mapping ciphertext $ct$ to message bit $Dec(sk, ct): \{0, 1\}$

- But we can also think of $Dec$ as a function
  - described by ciphertext $ct$
  - mapping secret key $sk$ to message bit $Dec(sk, ct): \{0, 1\}$
Homomorphic Decryption

- Fix a ciphertext $c$
- Define $f_c : sk \mapsto \text{Dec}(sk, c)$
- Assume $\text{Size}(f_c) < S$, $\text{Depth}(f_c) < D$
- Let $bk[1..k] = \text{Enc}(pk, sk[1..k])$

**Question**

What is the result of the following computation?

$$\text{Eval}_{C}(pk, f_c, bk[1..k])$$
Proxy Re-encryption

- Primary key: \((pk, sk)\)
- Secondary key: \((pk1, sk1)\)
- Re-encryption key: \(rk = Enc(pk1, sk[1..k])\)
- Input ciphertext \(c = Enc(pk, m)\)
- Decryption function \(f_c(sk) = Dec(sk, c)\)

**Question**

*What is the result of the following computation?*

\(EvalC(pk1, f_c, rk)\)
Decrypt and compute (unary)

- Homomorphic Encryption ($\text{Gen, Enc, Dec, Eval}$)
- Assume $\text{Func} = \{ f_c \mid c: \text{CipherText} \}$ where
  \[ f_c(sk) = \text{not} \ (\text{Dec}(sk, c)) \]
Homomorphic Encryption \((\text{Gen}, \text{Enc}, \text{Dec}, \text{Eval})\)

Assume \(\text{Func} = \{ f_c \mid c: \text{CipherText} \}\) where

\[ f_c(s_k) = \neg (\text{Dec}(s_k, c)) \]

\((p_k, s_k) \leftarrow \text{Gen}()\
\text{ek} = \text{Enc}(p_k, s_k)\
\text{c} = \text{Enc}(p_k, m)\

**Question**

*What is the result of the following computation?*

\(\text{EvalC}(p_k, f_c, \text{ek})\)
Homomorphic Encryption \((\text{Gen}, \text{Enc}, \text{Dec}, \text{Eval})\)

Assume \(\text{Func} = \{ f_{c,c'} \mid c,c': \text{CipherText} \}\) where

\[
f_{c,c'}(sk) = \text{Dec}(sk,c) \text{ nand } \text{Dec}(sk,c')
\]
Decrypt and compute (binary)

- Homomorphic Encryption (Gen, Enc, Dec, Eval)
- Assume Func = \{ f_{c,c'} | c, c' : CipherText \} where
  \[ f_{c,c'}(sk) = Dec(sk, c) \text{ nand } Dec(sk, c') \]

  \[ (pk, sk) \leftarrow Gen() \]
  \[ ek \leftarrow Enc(pk, sk) \]
  \[ c \leftarrow Enc(pk, m) \]
  \[ c' \leftarrow Enc(pk, m') \]

**Question**

*What is the result of the following computation?*

\[ EvalC(pk, f_{c,c'}, ek) \]
Bootstrapping

- Given (1-hop) `(Gen, Enc, Dec, Eval)` supporting functions
  \[ f_{c,c'}(sk) = Dec(sk,c) \text{ nand } Dec(sk,c') \]
Bootstrapping

- Given (1-hop) \( (\text{Gen}, \text{Enc}, \text{Dec}, \text{Eval}) \) supporting functions
  \[
f_{c,c'}(sk) = \text{Dec}(sk, c) \text{ nand } \text{Dec}(sk, c')
  \]

- Define (multi-hop) FHE scheme with \( \text{Func} = \{ \text{nand} \} \)

  \[
  \text{Gen}'() = (sk, pk) \leftarrow \text{Gen}()
  \]
  \[
  ek \leftarrow \text{Enc}(pk, sk)
  \]
  \[
  \text{return} (sk, (pk, ek))
  \]
  
  \[
  \text{Enc}'((pk, ek), m) = \text{Enc}(pk, m)
  \]
  
  \[
  \text{Eval}'((pk, ek), \text{nand}, c, c')
  = \text{EvalC}(pk, f_{c,c'}, ek)
  \]
Let \((\text{Gen}', \text{Enc}', \text{Dec}, \text{Eval}')\) be the new encryption scheme

**Theorem**

If \(\text{Dec}(sk, c) = m\) and \(\text{Dec}(sk, c') = m'\), then

\[
\text{Dec}(sk, \text{Eval}'((pk, ek), \text{nand}, c, c')) = m \text{nand} m'
\]
Correctness

Let \((\text{Gen}', \text{Enc}', \text{Dec}', \text{Eval}')\) be the new encryption scheme

**Theorem**

If \(\text{Dec}(sk, c) = m\) and \(\text{Dec}(sk, c') = m'\), then

\[
\text{Dec}(sk, \text{Eval}'((\text{pk}, \text{ek}), \text{nand}, c, c')) = m \text{ nand } m'
\]

Strong correctness property:

\[
\begin{align*}
\text{Dec}(sk, \text{Eval}'((\text{pk}, \text{ek}), \text{nand}, c, c')) \\
= \text{Dec}(sk, c) \text{ nand } \text{Dec}(sk, c')
\end{align*}
\]

for any ciphertexts \(c, c'\)!
Assume FHE = \((\text{Gen}, \text{Enc}, \text{Dec}, \text{Eval})\) is IND-CPA secure

Build new scheme FHE':

\[
\begin{align*}
\text{Gen}'() &= (sk, pk) \leftarrow \text{Gen}() \\
ek &\leftarrow \text{Enc}(pk, sk) \\
\text{return} &\quad (sk, (pk, ek)) \\
\text{Enc}'((pk, ek), m) &= \text{Enc}(pk, m)
\end{align*}
\]

Is FHE' IND-CPA secure?
Goal: build a FHE supporting NAND circuits of depth up to $L$, for any given $L$

Key generation procedure takes $L$ as input:
Goal: build a FHE supporting NAND circuits of depth up to L, for any given L

Key generation procedure takes L as input:

\[
\text{Gen}'(L) = \\
\text{for } (i = 0..L) \\
\quad \text{(sk}[i], \text{pk}[i]) \leftarrow \text{Gen}() \\
\text{for } (i = 1..L) \\
\quad \text{ek}[i] = \text{Enc}(\text{pk}[i], \text{sk}[i-1]) \\
\quad \text{sk}' = \text{sk}[0..L] \\
\quad \text{pk}' = \text{pk}[0..L], \text{ek}[1..L] \\
\text{return } (\text{sk}', \text{pk}')
\]

\[
\text{Enc}'(\text{pk}', m) = \text{Enc}(\text{pk}[0], m)
\]
FHE Today

State of the art
We can build leveled FHE from standard LWE assumption
- Built using bootstrapping
- Inefficient, but better than nothing

Open problem
Build (non-leveled) FHE from standard LWE
- In practice, one can apply bootstrapping with\
  \[ ek = \text{Enc}(pk, sk) \]
  - Much smaller key than leveled FHE
  - No known attacks to circular security
  - Still, it is not known how to prove security