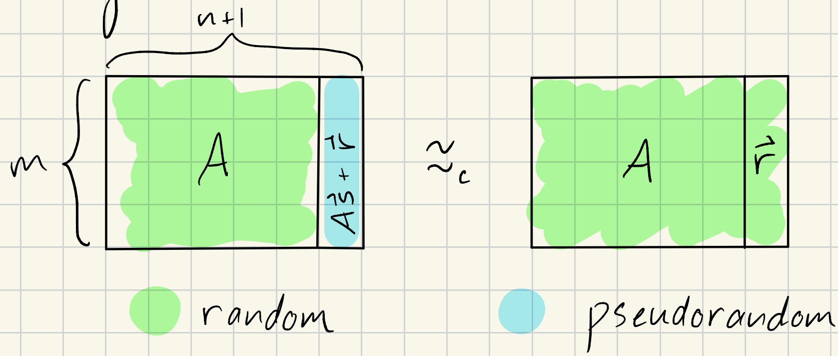
Fully Homomorphic Encryption

- What is FHE?
- "Leveled" FHE
  (low depth)
- Full FHE

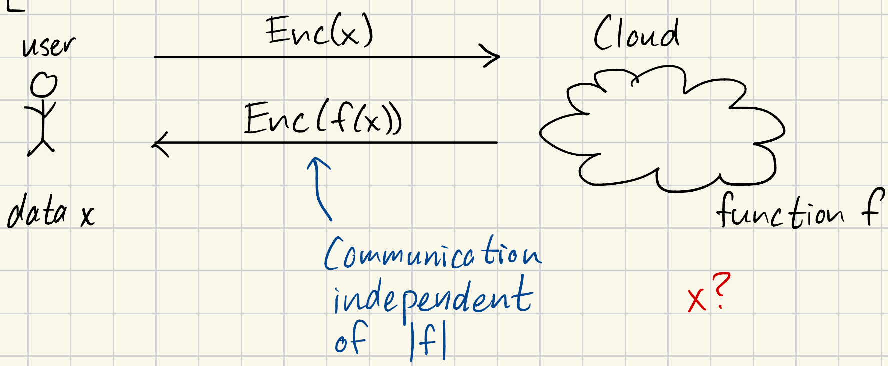
LWE(n, m, q, χ_B):

$$\left\{ (A, A\vec{s} + \vec{e}) : \begin{array}{l} A \xleftarrow{\$} \mathbb{Z}_q^{m\times n} \\ \vec{s} \xleftarrow{\$} \mathbb{Z}_q^{n} \\ \vec{e} \xleftarrow{\$} \chi_B^{m} \end{array} \right\} \approx \text{Uniform}\left[\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^{m} \right]$$

Visually



FHE



Examples:
- PIR: $f(i) = DB[i]$
- Private ML inference
- Outsourcing
- Search for E2EE cloud storage

History

- 1976: public-key crypto (Diffie-Helman)
- 1978: Rivest, Adelman, Dertouzous define FHE

  time passes...
- 2009: Craig Gentry (Stanford PhD student) gives first construction
  - new assumption (non-standard)
  - beautiful idea: bootstrapping
- 2011: FHE from LWE (Brakerski, Vaikuntanathan)
- 2013: Gentry, Sahai, Waters "3^{rd} gen FHE" ← today
  - simple
  - also from LWE

Syntax

KeyGen$(1^\lambda) \to sk$

Enc$(sk, \mu \in \{0,1\}) \to c$

Dec$(sk, c) \to \mu$

Eval$(f, c_1, \ldots c_\ell) \to c$

$\wedge$ $\oplus$ circuit — circuit inputs — output

public-key variants exist for simplicity, we consider secret-key

Properties

1. Correctness: $\forall f : \{0,1\}^\ell \to \{0,1\}$, $\mu_1, \ldots, \mu_\ell \in \{0,1\}$

   $sk \leftarrow$ KeyGen(1^λ)* w.p. 1

   $$\text{Dec}(sk, \text{Eval}(f, \text{Enc}(sk, \mu_1), \ldots, \text{Enc}(sk, \mu_\ell))) = f(\mu_1, \ldots, \mu_\ell)$$

   *let $y \leftarrow A(x, r)$ be a randomized alg. Then, "$\forall y \leftarrow A(x)$" denotes "$\forall r$, w/ $y \leftarrow A(x, r)$"

2. Semantic Security:

   $$\{\text{Enc}(sk, 0)\} \approx \{\text{Enc}(sk, 1)\}$$

3. Compactness: $\forall f, \mu_i, sk, c_i \leftarrow Enc(sk, \mu_i)$,

$$|Eval(f, c_1, \ldots, c_\ell)| = \underbrace{poly(\lambda)}_{\text{independent of } |f|}$$

Without compactness, any encryption gives FHE:

$$Eval(f, \vec{c}) \to (f, \vec{c})$$
$$Dec(sk, (f, \vec{c})) \to f(Dec(sk, c_1), \ldots, Dec(sk, c_\ell))$$

Eigenvalue Strawman

sk is a vector $\vec{s} \in \mathbb{Z}_q^n$

$Enc(\vec{s}, \mu) \to$ matrix $C \in \mathbb{Z}_q^{n \times n}$ s.t. $\overbrace{C\vec{s} = \mu\vec{s}}^{\mu \text{ is an eigenvalue of } C \text{ w/ eigenvector } \vec{s}}$

$Dec(\vec{s}, C) \to$ compute $C\vec{s}$ $(= \mu\vec{s})$, find μ

Homomorphic?

$C_1 \leftarrow Enc(\vec{s}, \mu_1)$, $C_2 \leftarrow Enc(\vec{s}, \mu_2)$

addition:

$$C_+ = C_1 + C_2$$
$$C_+ s = (C_1 + C_2)\vec{s} = C_1\vec{s} + C_2\vec{s} = \mu_1\vec{s} + \mu_2\vec{s} = \overbrace{(\mu_1 + \mu_2)}^{\checkmark}\vec{s}$$

multiplication:

$$C_\times = C_1 \cdot C_2$$
$$C_\times \vec{s} = C_1 C_2 \vec{s} = C_1 \mu_2 \vec{s} = \mu_2 C_1 \vec{s} = \overbrace{\mu_2 \mu_1}^{\checkmark}\vec{s}$$

Wow! Full $(+, \times)$ homomorphism

Insecure:

- Finding eigenvectors/eigenvalues is easy
- e.g. with Gaussian elimination $\leftarrow$ solve $C\vec{s} = 0$ or $(C - I)\vec{s} = 0$

Idea: make Gaussian elimination hard using noise

2nd try: sk Regev Encryption

KeyGen$(1^\lambda) \to \vec{s}$: $\tilde{s} \xleftarrow{\$} \mathbb{Z}_q^{n-1}$, $\vec{s} \leftarrow \begin{pmatrix} \tilde{s} \\ -1 \end{pmatrix} \in \mathbb{Z}_q^n$

Enc$(\vec{s}, \mu)$: $A \xleftarrow{\$} \mathbb{Z}_q^{n \times (n-1)}$, $\vec{e} \xleftarrow{\$} \chi_B^n$

output $C \leftarrow \underbrace{(A, A\tilde{s} + \vec{e})}_{\text{concatenation}} + \mu I_n$ ← $n \times n$ identity matrix

≈ random by LWE

Dec$(\vec{s}, C)$: output $\begin{cases} 0 & \|C\vec{s}\|_\infty \text{ is small} \\ 1 & \text{o.w.} \end{cases}$

Correctness:

$$C\vec{s} = (A, A\tilde{s} + \vec{e}) \begin{pmatrix} \tilde{s} \\ -1 \end{pmatrix} + \mu I_n \vec{s}$$

$$= A\tilde{s} - A\tilde{s} - \vec{e} + \mu\vec{s}$$

$$= \mu\vec{s} - \underbrace{\vec{e}}_{\substack{\text{noise} \\ (\|\vec{e}\|_\infty \le B)}} \begin{cases} \text{small}: & \mu = 0 \\ \text{large}: & \mu = 1 \end{cases}$$

$C\vec{s} = \mu\vec{s} + \text{noise}$ (apx eigenvector)

Additive Homomorphism is pretty good

Eval$(\text{"+"}, C_1, C_2) \to C_1 + C_2$:

$$(C_1 + C_2)\vec{s} = C_1\vec{s} + C_2\vec{s}$$

$$= \mu_1\vec{s} - \vec{e}_1 + \mu_2\vec{s} - \vec{e}_2$$

$$= (\mu_1 + \mu_2)\vec{s} - \underbrace{(\vec{e}_1 + \vec{e}_2)}_{\|\vec{e}_1 + \vec{e}_2\|_\infty \le 2B}$$

\# homomorphisms: $O\left(\frac{q}{B}\right)$

Multiplicative Homomorphism is bad:

Eval$(\text{"}\times\text{"}, C_1, C_2) \to C_1 C_2$:

$$C_1 C_2 \vec{s} = C_1(\mu_2\vec{s} - \vec{e}_2)$$

$$= \mu_2(C_1\vec{s}) - C_1\vec{e}_2$$

$$= \mu_2(\mu_1\vec{s} - \vec{e}_1) - C_1\vec{e}_2$$

$$= \underbrace{\mu_1\mu_2\vec{s}}_{\substack{\text{encryption of} \\ \mu_1\mu_2 \checkmark}} - \underbrace{\mu_2\vec{e}_1}_{\substack{\text{small noise} \\ (\text{since } \|e_1\|_\infty \le B \\ \text{and } \mu_2 \in \{0,1\})}} - \underbrace{C_1\vec{e}_2}_{\text{BIG since } C \text{ is} \approx \text{random}}$$

Q: Can we force ct matrix C to have small $\|C\|_\infty$

Idea: binary representation

Binary Decomposition: $\hat{\cdot}$

- $\hat{\cdot}$ for $\mathbb{Z}_q$:

  for $x \in \mathbb{Z}_q$, $\hat{x} = (\underbrace{x_0, \ldots, x_{\log q - 1}}_{\text{bits of } x}) \in \{0,1\}^{\log q}$

  s.t. $x = \sum_{i=0} x_i 2^i$

  fact: inverse of $\hat{x}$ is linear

  $x = \hat{x}^T \cdot \underbrace{(1, 2, \ldots, 2^{\log q - 1})}_{\vec{g}}$

- $\hat{\cdot}$ for $\mathbb{Z}_q^n$

  for $\vec{x} \in \mathbb{Z}_q^n$, $\hat{\vec{x}} = (x_{0,0} \ldots x_{0,\log q - 1}, \ldots, x_{n,0}, \ldots, x_{n,\log q - 1}) \in \mathbb{Z}_q^{n \log q}$

$\vec{x} = \hat{\vec{x}} \cdot G$ is linear with

$$G = \begin{pmatrix} 1 & & & \\ 2 & & & \\ \vdots & & & \\ 2^{\log q - 1} & & & \\ & 1 & & \\ & 2 & & \\ & \vdots & & \\ & 2^{\log q - 1} & & \\ & & \ddots & \\ & & & 1 \\ & & & 2 \\ & & & \vdots \\ & & & 2^{\log q - 1} \end{pmatrix} \in \mathbb{Z}_q^{n \log q \times n}$$

$= \vec{g} \otimes I_n$ (if you like tensor notation)

- $\hat{\cdot}$ for $\mathbb{Z}_q^{n \times n}$

$$\underbrace{C = \begin{pmatrix} - \vec{c}_1 - \\ \vdots \\ - \vec{c}_n - \end{pmatrix}}_{n \times n} \Rightarrow \hat{C} = \underbrace{\begin{pmatrix} - \hat{\vec{c}}_1 - \\ \vdots \\ - \hat{\vec{c}}_n - \end{pmatrix}}_{n \times n \log q}$$

again, $C = \hat{C} \cdot G$ is linear

same as before

Now back to FHE...

3rd Try (GSW)

KeyGen(1^λ): $\vec{s} = \begin{pmatrix} \tilde{s} \\ -1 \end{pmatrix} \in \mathbb{Z}_q^n$

Enc$(\vec{s}, \mu) \to A \xleftarrow{\$} \mathbb{Z}_q^{m \times (n-1)}$ $(m = n \log q)$

$\vec{e} \xleftarrow{\$} \chi_B^m$

$C \leftarrow (A, A\tilde{s} + \vec{e}) + \mu G \in \mathbb{Z}_q^{m \times n \log q}$

Output $\hat{C} \in \mathbb{Z}_q^{m \times m}$

Fact: $\|\hat{C}\| \le 1$

Dec$(\vec{s}, \hat{C})$:

Compute $\hat{C} G \vec{s} = C\vec{s}$

$$= (A, A\tilde{s} + \vec{e}) \begin{pmatrix} \tilde{s} \\ -1 \end{pmatrix} + \mu G \vec{s}$$

$$= \mu G \vec{s} - \vec{e}$$

if first element is small, output $\mu = 0$, else $\mu = 1$

$$(\mu G \vec{s})_1 = \left(\mu (1, 0, \ldots) \begin{pmatrix} s_1 \\ \vdots \\ s_{n-1} \\ -1 \end{pmatrix} \right)_1$$

$$= \mu s_1$$

and $|\mu s_1| \approx \Theta(q)$ w/ high probability

Checking $\times$ homomorphism:

Eval$(\text{"}\times\text{"}, \hat{C}_1, \hat{C}_2) \to \hat{C}_1 \hat{C}_2$

Dec$(\vec{s}, \hat{C}_1, \hat{C}_2)$:

$$\hat{C}_1 \hat{C}_2 G \vec{s} = \hat{C}_1 C_2 \vec{s}$$

$$= \hat{C}_1 (\mu_2 G \vec{s} + \vec{e}_2)$$

$$= \mu_2 \hat{C}_1 G \vec{s} + \hat{C}_1 \vec{e}_2$$

$$= \mu_2 C_1 \vec{s} + \hat{C}_1 \vec{e}_2$$

$$= \mu_2 (\mu_1 G \vec{s} - \vec{e}_1) + \hat{C}_1 \vec{e}_2$$

$$= \underbrace{\mu_1 \mu_2 G \vec{s}}_{\checkmark} - \underbrace{\mu_2 \vec{e}_1}_{\text{small}} + \underbrace{\hat{C}_1 \vec{e}_2}_{\text{small since } \|\hat{C}\|_\infty \le 1}$$

Could also think about $(+)$

but let's just do NAND(x,y) = NOT(AND(x,y))

$\to$ It's universal:

$\to \neg(x) = \text{NAND}(x, x)$

$\to x \wedge y = \neg\text{NAND}(x, y)$

$\to x \vee y = \text{NAND}(\neg x, \neg y)$

$\mathrm{Eval}(\mathrm{NAND}, C_1, C_2) \to I_m - \hat{C}_1 \hat{C}_2$

proof omitted

Where are we?

→ FHE scheme based on Regev

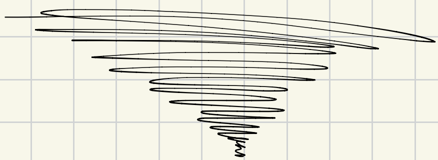
→ Noise grows (slowly)

→ Bounded-depth circuits ("leveled" FHE)

A brilliant way to reset noise: <u>bootstrapping</u>

Fact: decryption is a fn: $f_c(\cdot) = \mathrm{Dec}(\cdot, c)$
(with $f_c(\vec{s}) = \mu$)

Idea: eval f in FHE!



$$\begin{aligned} &\mathrm{Eval}(f_c, \mathrm{Enc}(\vec{s}, \vec{s})) \\ &= \mathrm{Enc}(\vec{s}, f_c(\vec{s})) && \text{(correctness)} \\ &= \mathrm{Enc}(\vec{s}, \mathrm{Dec}(\vec{s}, c)) && \text{(def}^{\text{n}}\ f_c) \\ &= \mathrm{Enc}(\vec{s}, \mu) && \text{(correctness)} \end{aligned}$$

So:

→ we started w/ C (encryption of μ)

→ ended w/ an encryption of μ

But, a noise analysis shows progress:

$$\underbrace{\mathrm{Eval}(\underbrace{f_c}_{\text{has some fixed depth}}, \underbrace{\mathrm{Enc}(\vec{s}, \vec{s})}_{\text{low noise (a fresh ct)}})}_{\text{has larger but fixed noise level that DOES NOT depend on c's noise}}$$

Caveats

1. Requires that (depth of f_c < depth limit)
   (or, Dec fails (in FHE))
2. Enc($\vec{s}$, $\vec{s}$) is made public
   - This is not part of SemSec / CPA / CCA
   - It's a new assumption - "circular security"
     ↳ reasonable
3. Eval(f_c, ·) is very expensive
   - so FHE expensive

   (but note, lattice crypto can be quite competitive w/ D-log for signatures, PKE...)

Recap

- FHE history
- FHE definition
- Eigen-encryption
- Eigen-encryption w/ noise
- Eigen-encryption w/ noise + binary decomp
  → leveled FHE
- Bootstrapping
  → leveled FHE → FHE

Today

Many kinds of FHE:
- GSW w/ $\vec{\mu} \in \{0, 1, \ldots, k\}^n$, $k \ll q$
  (vectorized operations, not just bits)
- CKKS: FHE for apx computations
  → ML apps
- t-FHE:
  → don't bootstrap w/ Dec(·, c); use a lookup table

Not there yet but lots of progress & investment:
Google, DARPA, Zama, Intel, Galois...