Efficient Fully Homomorphic Encryption from (Standard) LWE

Brakerski and Vaikuntanathan, FOCS 2011

Main contributions

- A scheme based on the standard learning with errors (LWE)
  - standard LWE as opposed to ring-LWE
- Security relies on (worst-case, classical) hardness of standard, well studied problems on arbitrary lattices.
  - Gentry: based on (worst-case, quantum) hardness of relatively untested ideal lattices problems.
- No squashing, thereby removing the (average-case) sparse subset-sum assumption, which is a very strong assumption.

Learning with errors (LWE) problem

- A vector $\mathbf{s} \in \mathbb{Z}_q^n$ satisfies a polynomial number of equations with errors: $\langle \mathbf{a}_i, \mathbf{s} \rangle \approx b_i$, or more precisely, $b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i$ where $\mathbf{a}_i \in_{\text{ur}} \mathbb{Z}_q^n$ and e_i is a samll random error, $1 \le i \le \text{poly}(n)$.

  LWE: Given $\{\mathbf{a}_i,\ b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i\}_{i=1}^{\text{poly}(n)}$, find $\mathbf{s}$.

- Decision LWE: distinguish between the two distributions

$$\{\mathbf{a}_i,\ \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i\}_{i=1}^{\text{poly}(n)} \quad \text{and} \quad \{\mathbf{a}_i,\ u_i\}_{i=1}^{\text{poly}(n)}$$

  where $\mathbf{a}_i \in_{\text{ur}} \mathbb{Z}_q^n$, $u_i \in_{\text{ur}} \mathbb{Z}_q$, and the noise/error $e_i \in \mathbb{Z}_q$, sampled according to some distribution, is much smaller than q.

- Worst-case SVP $\le$ average-case DLWE

Secret-key encryption based on LWE

- Since $\{\mathbf{a},\ \langle \mathbf{a},\mathbf{s}\rangle + e\}$ is almost uniformly random, so is $\{\mathbf{a},\ \langle \mathbf{a},\mathbf{s}\rangle + 2e\}$, provided q is odd. ($2^{-1} \bmod q$ exists; thus, as e ranges over $\mathbb{Z}_q$, $2e$ also ranges over $\mathbb{Z}_q$.)

- To encrypt a bit $\mu \in \{0,1\}$ using secret key $\mathbf{s} \in \mathbb{Z}_q^n$, we choose a random $\mathbf{a} \in \mathbb{Z}_q^n$ and a noise $e \ll q$ and encrypt μ as

$$c := \left(\mathbf{a},\ w = \langle \mathbf{a},\mathbf{s}\rangle + 2e + \mu\right) \quad \in \mathbb{Z}_q^n \times \mathbb{Z}_q$$

- To decrypt $c = (\mathbf{a},\ w)$, we compute

$$x := \underbrace{\underbrace{\left(w - \langle \mathbf{a},\mathbf{s}\rangle\right) \bmod q}_{=\ 2e+\mu,\ \text{since } e \ll q} \bmod 2}_{=\ \mu \bmod 2\ =\ \mu}$$

Convert it to a public-key encryption scheme

- Use **s** as the secret key and use a sequence $\left\{\mathbf{a}_i,\, b_i = \langle \mathbf{a}_i, \mathbf{s}\rangle + 2e_i\right\}_{i=1}^{m}$ as the public key.

- To encrypt a bit $\mu \in \{0,1\}$ using public key $\left\{\mathbf{a}_i,\, b_i\right\}_{i=1}^{m}$, we choose a random vector $(r_1,\ \ldots,\ r_m) \in \{0,\ 1\}^m$ and encrypt μ as

$$c \ := \ \left(\sum r_i \mathbf{a}_i,\ \sum r_i b_i + \mu\right) \ = \ \left(\mathbf{a},\ w = \langle \mathbf{a}, \mathbf{s}\rangle + 2e + \mu\right)$$

  where $\mathbf{a} = \sum r_i \mathbf{a}_i$ and $e = \sum r_i e_i$.

- Note: m must be much smaller than q to ensure $e \ll q$.

Is it additively homomorphic?

- Given ciphertexts of m and m', //plaintexts: $m,\ m' \in \{0,1\}$//

$$c_m = (\mathbf{a}, w) = \left(\mathbf{a},\ \langle \mathbf{a}, \mathbf{s} \rangle + 2e + m\right)$$

$$c_{m'} = (\mathbf{a}', w') = \left(\mathbf{a}',\ \langle \mathbf{a}', \mathbf{s} \rangle + 2e' + m'\right)$$

can we compute a ciphertext $c_{m+m'}$ of $m + m'$?

- Adding up c_m and $c_{m'}$ yields

$$c_m + c_{m'} = (\mathbf{a} + \mathbf{a}', w + w') = \left(\mathbf{a} + \mathbf{a}',\ \langle \mathbf{a} + \mathbf{a}', \mathbf{s} \rangle + 2(e + e') + m + m'\right)$$

- It is a ciphertext of $m + m'$. So, simply let $c_{m+m'} := c_m + c_{m'}$.
- The scheme is additively homomorphic.

Is it multiplicatively homomorphic?

- Given ciphertexts of m and m',

$$c_m = (\mathbf{a}, w) = (\mathbf{a}, \ \langle \mathbf{a}, \mathbf{s} \rangle + 2e + m) \qquad \in \mathbb{Z}_q^n \times \mathbb{Z}_q$$

$$c_{m'} = (\mathbf{a}', w') = (\mathbf{a}', \ \langle \mathbf{a}', \mathbf{s} \rangle + 2e' + m') \qquad \in \mathbb{Z}_q^n \times \mathbb{Z}_q$$

  we wish to compute a ciphertext $c_{mm'}$ of $m \cdot m'$.

- Cannot simply multiply c_m and $c_{m'}$. Why?

- Ciphertexts $(\mathbf{a}, w)$, $(\mathbf{a}', w')$ give "approximations" of m, m':

$$m \ \approx \ w - \langle \mathbf{a}, \mathbf{s} \rangle \ = \ w - \sum \mathbf{a}[i] \cdot \mathbf{s}[i] \quad \text{where } \mathbf{a} = (\mathbf{a}[1], \ldots, \mathbf{a}[n])$$

$$m' \ \approx \ w' - \langle \mathbf{a}', \mathbf{s} \rangle \ = \ w' - \sum \mathbf{a}'[i] \cdot \mathbf{s}[i]$$

- Our goal is to obtain $m \cdot m' \approx \overline{w} - \langle \overline{\mathbf{a}}, \mathbf{s} \rangle$ for some $(\overline{\mathbf{a}}, \overline{w})$.

Re-linearization

- $m \cdot m' \approx \left(w - \sum \mathbf{a}[i] \cdot \mathbf{s}[i]\right) \cdot \left(w' - \sum \mathbf{a}'[i] \cdot \mathbf{s}[i]\right)$

$$= h_0 + \sum_{i=1}^{n} h_i \cdot \mathbf{s}[i] + \sum_{1 \le i \le j \le n} h_{i,j} \cdot \underbrace{\mathbf{s}[i] \cdot \mathbf{s}[j]}_{\text{quadratic}}$$

$$= \sum_{0 \le i \le j \le n} h_{i,j} \cdot \underbrace{\mathbf{s}[i] \cdot \mathbf{s}[j]}_{\text{quadratic}} \qquad \text{//here we let } \mathbf{s}[0] = 1\text{//}$$

- To linearize the quadratic terms, take another key $\mathbf{t} \in \mathbb{Z}_q^n$ and encode/approximate $\mathbf{s}[i] \cdot \mathbf{s}[j]$ as:

$$\mathbf{s}[i] \cdot \mathbf{s}[j] \approx b_{i,j} - \langle \mathbf{a}_{i,j}, \mathbf{t} \rangle \qquad //b_{i,j} = \langle \mathbf{a}_{i,j}, \mathbf{t} \rangle + 2e_{i,j} + \mathbf{s}[i] \cdot \mathbf{s}[j]//$$

- Now, substitute this into the above equation of $m \cdot m'$.

- $$m \cdot m' \approx \sum_{0 \le i \le j \le n} h_{i,j} \cdot \underbrace{\mathbf{s}[i] \cdot \mathbf{s}[j]}_{\text{quadratic}}$$

$$\approx \sum h_{i,j} \cdot \left(b_{i,j} - \left\langle \mathbf{a}_{i,j}, \mathbf{t} \right\rangle \right)$$

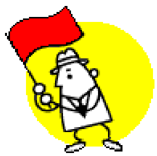
$$= \left(\sum h_{i,j} \cdot b_{i,j} \right) - \left\langle \sum h_{i,j} \mathbf{a}_{i,j}, \mathbf{t} \right\rangle$$

$$= \bar{w} - \left\langle \bar{\mathbf{a}}, \mathbf{t} \right\rangle$$

- Let $c_{m \cdot m'} := \left(\bar{\mathbf{a}}, \bar{w} \right);$ we have a ciphertext of $m \cdot m'$ under key $\mathbf{t}$. Thus, from the ciphertexts of $m,\ m'$ under key $\mathbf{s},$ we can compute a ciphertext of $m \cdot m'$ under another key $\mathbf{t}$.

- In the above re-linearization argument, we had

$$m \cdot m' \approx \sum h_{i,j} \cdot \mathbf{s}[i] \cdot \mathbf{s}[j]$$



$$\approx \sum h_{i,j} \cdot \left(b_{i,j} - \langle \mathbf{a}_{i,j}, \mathbf{t} \rangle \right)$$

where "$\approx$" means "differs by a small $2e \ll q$."

- Unfortunately, the last $\approx$ does not necessarily hold, for even though $\mathbf{s}[i] \cdot \mathbf{s}[j] \approx b_{i,j} - \langle \mathbf{a}_{i,j}, \mathbf{t} \rangle$, it may happen that

$$h_{i,j} \cdot \mathbf{s}[i] \cdot \mathbf{s}[j] \not\approx h_{i,j} \cdot \left(b_{i,j} - \langle \mathbf{a}_{i,j}, \mathbf{t} \rangle \right)$$

unless $h_{i,j}$ is extremely small.

- In binary, $h_{i,j} = \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,j,\tau} \cdot 2^{\tau}$, where $h_{i,j,\tau} \in \{0,1\}$.
- Thus, $m \cdot m' \approx \sum_{0 \le i \le j \le n} h_{i,j} \cdot \mathbf{s}[i] \cdot \mathbf{s}[j]$

$$\approx \sum_{0 \le i \le j \le n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,j,\tau} \cdot \underbrace{2^{\tau} \cdot \mathbf{s}[i] \cdot \mathbf{s}[j]}_{\approx b_{i,j,\tau} - \langle \mathbf{a}_{i,j,\tau}, \mathbf{t} \rangle}$$

$$\approx \sum_{0 \le i \le j \le n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,j,\tau} \left(b_{i,j,\tau} - \langle \mathbf{a}_{i,j,\tau}, \mathbf{t} \rangle \right)$$

- In the above, by $\underbrace{2^{\tau}\cdot\mathbf{s}[i]\cdot\mathbf{s}[j]}_{\approx b_{i,t,\tau}-\langle \mathbf{a}_{i,j,\tau},\mathbf{t}\rangle}$, we meant to obtain

$\left(\mathbf{a}_{i,j,\tau},\ b_{i,j,\tau}\right)\in \mathbb{Z}_q^n\times\mathbb{Z}_q$ and $e_{i,j,\tau}\ll q$ such that

$$b_{i,j,\tau} = \left\langle \mathbf{a}_{i,j,\tau},\mathbf{t}\right\rangle + 2e_{i,j,\tau} + 2^{\tau}\cdot\mathbf{s}[i]\cdot\mathbf{s}[j]$$

$$\Rightarrow\ 2^{\tau}\cdot\mathbf{s}[i]\cdot\mathbf{s}[j]\ \approx\ b_{i,j,\tau} - \left\langle \mathbf{a}_{i,j,\tau},\mathbf{t}\right\rangle.$$

Summary: multiplicative homomorphism

- Given ciphertexts of m and m' under key $\mathbf{s}$,

$$c_m = (\mathbf{a}, w) \quad \Rightarrow \quad m \approx w - \langle \mathbf{a}, \mathbf{s} \rangle$$

$$c_{m'} = (\mathbf{a}', w') \quad \Rightarrow \quad m' \approx w' - \langle \mathbf{a}', \mathbf{s} \rangle$$

  we wish to compute a ciphertext $c_{mm'}$ of $m \cdot m'$.

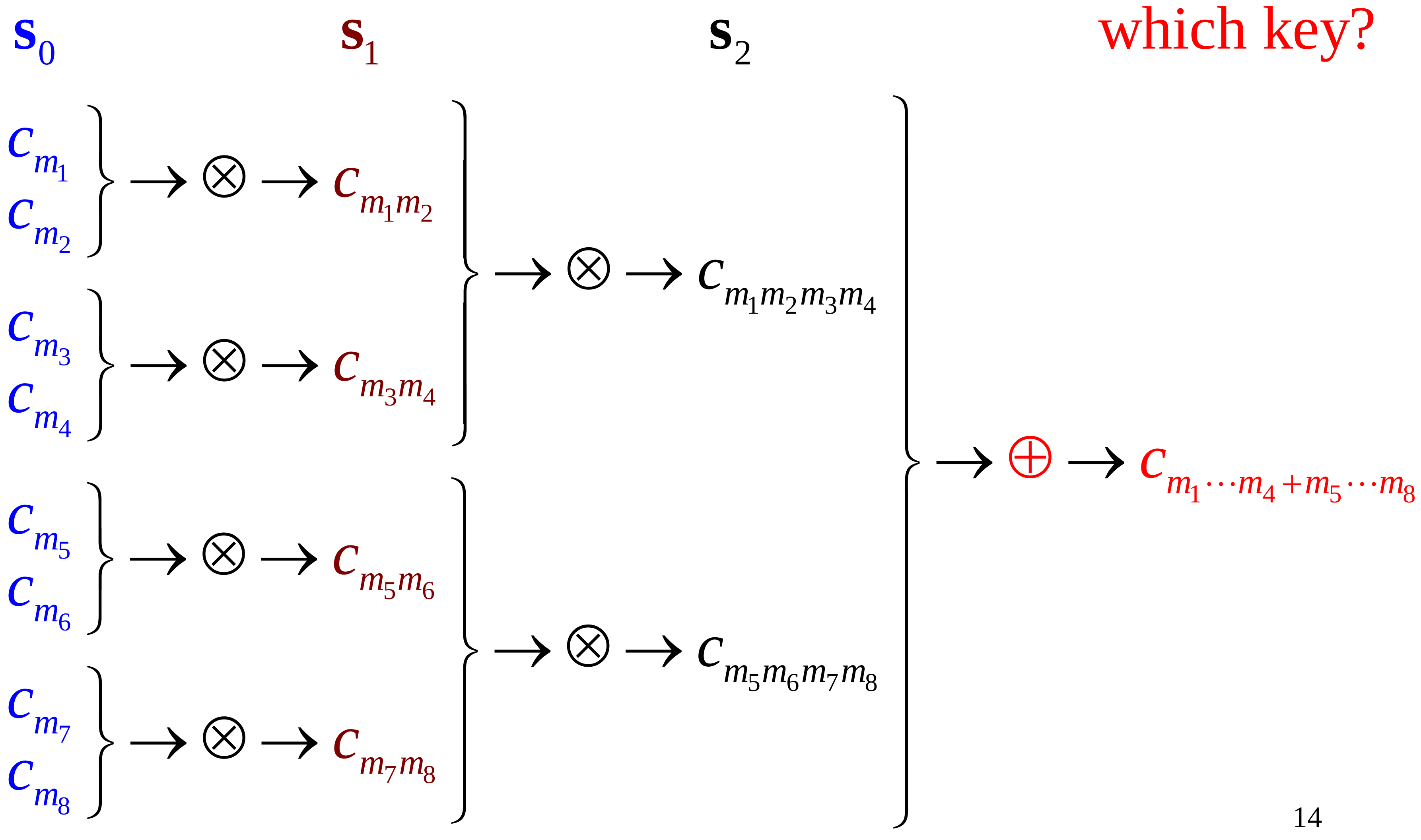
- We obtained $m \cdot m' \approx \sum_{0 \le i \le j \le n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,j,\tau} \left(b_{i,t,\tau} - \langle \mathbf{a}_{i,j,\tau}, \mathbf{t} \rangle \right)$

$$= \underbrace{\sum_{0 \le i \le j \le n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,j,\tau} \cdot b_{i,t,\tau}}_{w_{mm'}} - \Big\langle \underbrace{\sum_{0 \le i \le j \le n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,j,\tau} \cdot \mathbf{a}_{i,j,\tau}}_{\mathbf{a}_{mm'}}, \ \mathbf{t} \Big\rangle$$

- This suggests: $c_{mm'} = \left(\mathbf{a}_{mm'}, \ w_{mm'} \right)$ under another key $\mathbf{t}$.

It is somewhat homomorphic

- Use a sequence of keys: $\mathbf{s}_0, \mathbf{s}_1, \ldots$



- We will use a sequence of keys $\mathbf{s}_0,\ \mathbf{s}_1,\ \ldots,\ \mathbf{s}_L$.
- Key $\mathbf{s}_{\ell-1}$ is "encrypted" under key $\mathbf{s}_\ell$ in the sense that

$$\underbrace{2^\tau \cdot \mathbf{s}_{\ell-1}[i] \cdot \mathbf{s}_{\ell-1}[j]}_{\approx\, b_{\ell,\mathrm{i},,t,\tau} - \langle \mathbf{a}_{\ell,\mathrm{i},,t,\tau},\, \mathbf{s}_\ell \rangle}$$

where $\mathbf{a}_{\ell,i,j,\tau} \in \mathbb{Z}_q^n$, $e_{\ell,i,j,\tau} \ll q$, and

$$b_{\ell,i,j,\tau} = \langle \mathbf{a}_{\ell,i,j,\tau},\ \mathbf{s}_\ell \rangle + 2e_{\ell,i,j,\tau} + 2^\tau \cdot \mathbf{s}[i] \cdot \mathbf{s}[j].$$

- In key generation, we will generate $\mathbf{s}_0,\ \mathbf{s}_1,\ \ldots$ and $\mathbf{a}_{\ell,i,j,\tau},\ e_{\ell,i,j,\tau}$, and compute $b_{\ell,i,j,\tau}$.

The scheme allows L levels of multiplications

- The error in the ciphertext grows with each multiplication (and addition, but the latter is relatively small).
- Analysis shows that the scheme allows up to $L = \varepsilon \log n$ levels of multiplications for any arbitrary constant $\varepsilon < 1$.
  - This corresponds to degree $D = n^{\varepsilon}$ polynomials.
- Beyond that, the error may become too large (close to q) and detroy the ciphertext.
- Use bootstrapping to refresh the ciphertext!

Is it bootstrappable?

- The scheme is somewhat homomorphic, capable of evaluating polynomials of degree $\leq D = n^{\varepsilon} < n$. $(\varepsilon < 1.)$
- For bootstrapping, the scheme must be able to evaluate the decryption circuit homomorphically.
- Ciphertext: $(\mathbf{a},\ w) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$.
- Decryption: $w - \langle \mathbf{a}, \mathbf{s} \rangle \bmod q \bmod 2,$ which is equivalent to evaluating a polynomial of degree $\geq \max(n,\ \log q) > D.$
- The decryption complexity is too big for bootstrapping!

- All prior SHE schemes encounter the same problem: short of evaluating the decryption circuit.
- Gentry, followed by all others, handled the situation by resorting to squashing, which required a very strong sparse subset-sum assumption.
- This paper proposes a non-squashing technique to make the decryption circuit evaluable, thereby removing the undesired sparse subset-sum assumption.
- The proposed technique, called dimension-modulus reduction, is to reduce the dimension n and modulus q of the ciphertext, making $\max(n, \log q)$ smaller.

Dimension-modulus reduction

- Basic idea: given a ciphertext with parameter $(n, \log q)$, convert it to a ciphertext with parameter $(k, \log p)$ which are much smaller than $(n, \log q)$.
  - Convert $(\mathbf{a}, w) \in \mathbb{Z}_q^n \times \mathbb{Z}_q \Rightarrow (\mathbf{a}', w') \in \mathbb{Z}_p^k \times \mathbb{Z}_p$.
- Typically, $k =$ security parameter, $p = \text{poly}(k)$,

$$n = k^c \text{ with } c > 1, \text{ and } q = 2^{n^\varepsilon}.$$

- Suppose it can evaluate polys of degree $D = n^\varepsilon = k^{c-\varepsilon}$.
- Choose c to be large enough so that this is sufficient to evaluate the $(k, \log p)$ decryption circuit.

Dimension reduction $(n \to k)$ (q remains the same)

- Given a ciphertext $\left(\mathbf{a}, w = \langle \mathbf{a}, \mathbf{s} \rangle + 2e + \mu\right) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ under a secret key $\mathbf{s} \in \mathbb{Z}_q^n$, we want to convert it to a ciphertext $\left(\mathbf{a}', w' = \langle \mathbf{a}', \mathbf{t} \rangle + 2e' + \mu\right) \in \mathbb{Z}_q^k \times \mathbb{Z}_q$ under a key $\mathbf{t} \in \mathbb{Z}_q^k$.
- The technique is similar to that of re-linearization:
  - We have $\mu \approx w - \langle \mathbf{a}, \mathbf{s} \rangle = \sum_{0 \le i \le n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \cdot 2^{\tau} \cdot \mathbf{s}[i]$. $//\mathbf{s}[0] = 1//$
  - Encode $2^{\tau} \cdot \mathbf{s}[i] \approx b_{i,\tau} - \langle \mathbf{a}_{i,\tau}, \mathbf{t} \rangle$. //Note: $\mathbf{a}_{i,\tau}, \mathbf{t} \in \mathbb{Z}_q^k$//
  - Then, $\mu \approx \sum_{0 \le i \le n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \left(b_{i,\tau} - \langle \mathbf{a}_{i,\tau}, \mathbf{t} \rangle\right) = w' - \langle \mathbf{a}', \mathbf{t} \rangle$.
  - New ciphertext: $(\mathbf{a}', w') \in \mathbb{Z}_q^k \times \mathbb{Z}_q$.

Dimension-modulus reduction: $(n,q) \rightarrow (k,p)$

- Want to convert a ciphertext $(\mathbf{a}, w) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ under key $\mathbf{s} \in \mathbb{Z}_q^n$ to a ciphertext $(\hat{\mathbf{a}}, \hat{w}) \in \mathbb{Z}_p^k \times \mathbb{Z}_p$ under key $\mathbf{t} \in \mathbb{Z}_p^k$.
- We have $\mu \approx w - \langle \mathbf{a}, \mathbf{s} \rangle = \sum_{0 \le i \le n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \cdot 2^\tau \cdot \mathbf{s}[i] \bmod q$.
- Rather than encoding $2^\tau \cdot \mathbf{s}[i] \in \mathbb{Z}_q$ as in the last slide, we encode $\left\lfloor \frac{p}{q} \cdot 2^\tau \cdot \mathbf{s}[i] \right\rceil \in \mathbb{Z}_p$ under key $\mathbf{t} \in \mathbb{Z}_p^k$.
  - This is to scale down $2^\tau \cdot \mathbf{s}[i]$ from $\mathbb{Z}_q$ to $\mathbb{Z}_p$.

- To encode $\lfloor p/q \cdot 2^\tau \cdot \mathbf{s}[i] \rceil \in \mathbb{Z}_p$ under key $\mathbf{t} \in \mathbb{Z}_p^k$:
  - Randomly choose $\mathbf{a}_{i,\tau} \in \mathbb{Z}_p^k$ and $e_{i,\tau} \ll p$, and let

$$b_{i,\tau} = \langle \mathbf{a}_{i,\tau}, \mathbf{t} \rangle + e + \lfloor p/q \cdot (2^\tau \cdot \mathbf{s}[i]) \rceil \mod p$$

  - This gives $2^\tau \cdot \mathbf{s}[i] \approx \frac{q}{p} \cdot (b_{i,\tau} - \langle \mathbf{a}_{i,\tau}, \mathbf{t} \rangle) \mod p$

- Thus, $\mu \approx \sum_{0 \le i \le n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \cdot 2^\tau \cdot \mathbf{s}[i] \mod q$ (from last slide)

$$\approx \sum_{0 \le i \le n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \left(\frac{q}{p} \cdot (b_{i,\tau} - \langle \mathbf{a}_{i,\tau}, \mathbf{t} \rangle) \right) \mod p \underbrace{\mod q}_{\text{not needed}}$$

- $$\mu \approx \sum_{0\le i\le n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \left(\frac{q}{p} \cdot \left(b_{i,\tau} - \langle \mathbf{a}_{i,\tau}, \mathbf{t} \rangle \right) \right) \bmod p$$
$$= w' - \langle \mathbf{a}', \mathbf{t} \rangle$$

- This suggests:

$$w' = \sum_{0\le i\le n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \left(\frac{q}{p} \cdot \left(b_{i,\tau} \right) \right) \bmod p \qquad \in \mathbb{Z}_p$$

$$\mathbf{a}' = \sum_{0\le i\le n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \left(\frac{q}{p} \cdot \left(\mathbf{a}_{i,\tau} \right) \right) \bmod p \qquad \in \mathbb{Z}_p^k$$

The New FHE Scheme

based on the idea of re-linearization
and dimension-modulus reduction
without squashing

Parameters

- Security parameter κ.
- Dimensions n and k.
- Odd moduli q and p.
- Noise distributions χ over $\mathbb{Z}_q$ and $\hat{\chi}$ over $\mathbb{Z}_p$.
- Long: n, q, χ. Short: k, p, $\hat{\chi}$.
- L: maximum depth of circuits that can be evaluated.
- m: used in key generation.
- Example: $k=\kappa$, $n=k^4$, $q\approx 2^{\sqrt{n}}$, $p=(n^2\log q)\cdot\text{poly}(k)$, $m=O(n\log q)$, $L=1/3\cdot\log n$, χ is n-bounded, and $\hat{\chi}$ is k-bounded.

Bounded distributions

- A distribution ensemble $\{\chi_\kappa\}_{\kappa\in\mathbb{N}}$, over the integers, is called B-bounded if

$$\Pr\left[|x| > B : x \leftarrow_{\mathrm{R}} \chi_\kappa\right] \leq 2^{-\tilde{\Omega}(\kappa)}.$$

(The probability that $|x| > B$ is negligible.)

- Recall that our χ and $\hat{\chi}$ will be n- and k-bounded, respectively.

Key generation SH.Keygen(1^κ)

- Generate $L+1$ keys $\mathbf{s}_0,\ \mathbf{s}_1,\ \ldots,\ \mathbf{s}_L \leftarrow_{\mathrm{R}} \mathbb{Z}_q^n$.
- For $1 \le \ell \le L,\ 0 \le i \le j \le n,\ 0 \le \tau \le \lfloor \log q \rfloor$,
  - $\mathbf{a}_{\ell,i,j,\tau} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^n$ and $e_{\ell,i,j,\tau} \leftarrow_{\mathrm{R}} \chi$
  - $b_{\ell,i,j,\tau} := \langle \mathbf{a}_{\ell,i,j,\tau},\ \mathbf{s}_\ell \rangle + 2e_{\ell,i,j,\tau} + 2^\tau \cdot \mathbf{s}[i] \cdot \mathbf{s}[j]$
  - $\psi_{\ell,i,j,\tau} := \left(\mathbf{a}_{\ell,i,j,\tau},\ b_{\ell,i,j,\tau}\right)$.

- $\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{m\times n}$, $\mathbf{e} \leftarrow_{\mathrm{R}} \chi^m$, $\mathbf{b} := \mathbf{A}\mathbf{s}_0 + 2\mathbf{e}$.
- Output of key generation:
  - Secret key $sk = \mathbf{s}_L$.
  - Public key $pk = (\mathbf{A},\ \mathbf{b})$.
  - Evaluation key $evk = \mathbf{\Psi} = \{\psi_{\ell,i,j,\tau}\} = \{(\mathbf{a}_{\ell,i,j,\tau},\ b_{\ell,i,j,\tau})\}$.

Encryption key

embedded in

$$\mathbf{b} := \mathbf{A}\mathbf{s}_0 + 2\mathbf{e}$$

Decryption key

$$\mathbf{s}_0 \qquad \mathbf{s}_1 \qquad \cdots \qquad \mathbf{s}_{L-1} \qquad \mathbf{s}_L$$

$$\overline{\mathbf{s}_0} \qquad \overline{\mathbf{s}_1} \qquad \cdots \qquad \overline{\mathbf{s}_{L-1}}$$

Evaluation key

encrypted as $\underbrace{2^{\tau} \cdot \mathbf{s}_{\ell-1}[i] \cdot \mathbf{s}_{\ell-1}[j]}_{\approx\, b_{\ell,\mathrm{i},,t,\tau} - \langle \mathbf{a}_{\ell,\mathrm{i},,t,\tau},\, \mathbf{s}_{\ell} \rangle}$

Encryption $\mathrm{SH.Enc}_{pk}(\mu)$

- Recall $pk = (\mathbf{A},\ \mathbf{b})$.
- To encrypt a message $\mu \in \{0,1\}$:
  - Sample a vector of m bits, $\mathbf{r} \leftarrow_{\mathrm{R}} \{0,1\}^m$.
  - $\mathbf{v} := \mathbf{A}^T \mathbf{r}$.
  - $w := \mathbf{b}^T \mathbf{r} + \mu$.
  - Ciphertext $c := ((\mathbf{v},\ w),\ 0)$.
- 0 here indicates level 0 or fresh ciphertext.
- In general, ciphertexts are of the form $((\mathbf{v},\ w),\ \ell)$.

Decryption $\mathrm{SH.Dec}_{sk}(c)$

- Recall $sk = \mathbf{s}_L$.
- To decrypt a ciphertext $c =: \big((\mathbf{v},\ w),\ L\big)$:
  - $\mu \ := \ \big(w - \langle \mathbf{v},\ \mathbf{s}_L \rangle\big) \bmod q \bmod 2.$
- Note: the ciphertext is an output of SH.Eval.

Homomorphic evaluation SH.Eval$_{evk}(f, c_1, \ldots, c_t)$

- Boolean function $f: \{0,1\}^t \to \{0,1\}$:
  - represented by a circuit with layers of "+" and "$\times$" gates;
  - each layer is either all "+" gates or all "$\times$" gates;
  - there are exactly L layers of "$\times$" gates;
  - "$\times$" gate: fan-in 2; "$+$" gate: arbitrary fan-in.
- Note: Any boolean circuit can be converted to this form for some L.
- Evaluate the circuit layer by layer and gate by gate.

Evaluation of addition gates

$\text{SH.Eval}_{evk}(\text{mult}, c_1, \ldots, c_t)$

- Input: $c_1, \ldots, c_t$, where $c_i = \left((\mathbf{v}_i, w_i), \ell\right)$.
- Output: $c_{\text{add}} = \left((\mathbf{v}_{\text{add}}, w_{\text{add}}), \ell\right)$ where
  - $\mathbf{v}_{\text{add}} := \sum \mathbf{v}_i$
  - $w_{\text{add}} := \sum w_i$

$\text{SH.Eval}_{evk}(\text{mult}, c, c')$

- Input: $c = ((\mathbf{v}, w), \ell), \; c' = ((\mathbf{v}', w'), \ell)$.
- Output: $c_{\text{mult}} = ((\mathbf{v}_{\text{mult}}, w_{\text{mult}}), \ell + 1)$ where
  - $\mathbf{v}_{\text{mult}} := \sum_{\substack{0 \le i \le j \le n \\ 0 \le \tau \le \lfloor \log q \rfloor}} h_{i,j,\tau} \cdot \mathbf{a}_{\ell+1,i,j,\tau}$
  - $w_{\text{add}} := \sum_{\substack{0 \le i \le j \le n \\ 0 \le \tau \le \lfloor \log q \rfloor}} h_{i,j,\tau} \cdot b_{\ell+1,i,j,\tau}$

- In the above, recall:
  - $evk = \{\psi_{\ell,i,j,\tau}\} = \{(\mathbf{a}_{\ell,i,j,\tau},\ b_{\ell,i,j,\tau})\}.$
  - $h_{i,j} = \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,j,\tau} \cdot 2^{\tau}$ (in binary).
  - $h_{i,j}$ are the coefficients of $\Phi_{(\mathbf{v},\ w),\ (\mathbf{v}',\ w')}(\mathbf{x})$ and can be computed from $(\mathbf{v},\ w),\ (\mathbf{v}',\ w'),$ where

    $$\Phi_{(\mathbf{v},\ w),\ (\mathbf{v}',\ w')}(\mathbf{x})$$

    $$= \left(w - \sum \mathbf{v}[i] \cdot \mathbf{x}[i]\right) \cdot \left(w' - \sum \mathbf{v}'[i] \cdot \mathbf{x}[i]\right)$$

    $$= \sum_{0 \le i \le j \le n} h_{i,j} \cdot \mathbf{x}[i] \cdot \mathbf{x}[j]$$

Make the SH scheme bootstrappable

$$\mathbf{s}_0 \quad \mathbf{s}_1 \quad \cdots \quad \mathbf{s}_{L-1} \quad \mathbf{s}_L \quad \hat{\mathbf{s}}$$

$$\overline{\mathbf{s}_0} \quad \overline{\mathbf{s}_1} \quad \cdots \quad \overline{\mathbf{s}_{L-1}} \quad \overline{\mathbf{s}_L}$$

encrypted as

$$\underbrace{\frac{p}{q}\cdot 2^{\tau}\cdot \mathbf{s}_L[i]}_{\approx\ \cdot \hat{b}_{i,\tau} - \langle \hat{\mathbf{a}}_{i,\tau},\, \hat{\mathbf{s}} \rangle}$$

Key generation BTS.Keygen(1^κ)

- Run SH.Keygen(1^κ) to obtain the secret key $\mathbf{s}_L$, public key $(\mathbf{A},\ \mathbf{b})$, and evaluation key $\mathbf{\Psi}$.
- Generate a short secret key $\hat{\mathbf{s}} \leftarrow_R \mathbb{Z}_p^k$, and for $0 \le i \le n$, $0 \le \tau \le \lfloor \log q \rfloor$, compute
  - $\hat{\mathbf{a}}_{i,\tau} \leftarrow_R \mathbb{Z}_p^k$ and $\hat{e}_{i,\tau} \leftarrow_R \hat{\chi}$
  - $\hat{b}_{i,\tau} := \langle \hat{\mathbf{a}}_{i,\tau},\ \hat{\mathbf{s}} \rangle + \hat{e}_{i,\tau} + \left\lfloor \frac{p}{q} \cdot \left(2^\tau \cdot \mathbf{s}_L[i] \right) \right\rceil \bmod p$
  - $\hat{\psi}_{i,\tau} := \left(\hat{\mathbf{a}}_{i,\tau},\ b_{i,\tau} \right)$. Let $\mathbf{\Psi} = \left\{ \hat{\psi}_{i,\tau} \right\}$.

- Output of key generation:
  - Secret key: $sk = \hat{\mathbf{s}}$.
  - Public key: $pk = (\mathbf{A},\ \mathbf{b})$.
  - Evaluation key: $evk = \left(\mathbf{\Psi},\ \hat{\mathbf{\Psi}}\right)$.

Encryption $\text{BTS.Enc}_{pk}(\mu)$

- Same as $\text{SH.Enc}_{pk}(\mu)$.

Decryption $\text{BTS.Enc}_{sk}(\hat{c})$

- To decrypt ciphertext $\hat{c} = (\hat{\mathbf{v}}, \hat{w}) \in \mathbb{Z}_p^k \times \mathbb{Z}_p$, compute $\mu^* := \left(\hat{w} - \left\langle \hat{\mathbf{v}}, \hat{\mathbf{s}} \right\rangle\right) \bmod p \bmod 2$.

- It's correct if $\hat{w} - \left\langle \hat{\mathbf{v}}, \hat{\mathbf{s}} \right\rangle = \mu + 2\hat{e} \bmod p$ and $\hat{e}$ is small.

Evaluation $\text{BTS.Eval}_{evk}(f, c_1, \ldots, c_t)$

- Run $\text{SH.Eval}_{\boldsymbol{\Psi}}$ to obtain a ciphertext $c_f \in \mathbb{Z}_q^n \times \mathbb{Z}_q \times \{L\}$:

$$c_f = \big((\mathbf{v}, w), L\big) \leftarrow \text{SH.Eval}_{\boldsymbol{\Psi}}(f, c_1, \ldots, c_t)$$

- Reduce the dimension and modulus of c_f to k, p.

  The new ciphertext is $\hat{c} = (\hat{\mathbf{v}}, \hat{w})$, where

$$\hat{w} = 2 \cdot \sum_{i=0}^{n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \cdot \hat{b}_{i,\tau} \mod p \qquad \in \mathbb{Z}_p$$

$$\hat{\mathbf{v}} = 2 \cdot \sum_{i=0}^{n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \cdot \hat{\mathbf{a}}_{i,\tau} \mod p \qquad \in \mathbb{Z}_p^k$$

- Theorem. If the ciphertext $c_f = \left((\mathbf{v},\ w),\ L\right)$ satisfies

$$w - \left\langle \mathbf{v},\ \mathbf{s}_L \right\rangle = \mu + 2e \mod q,$$

then the reduced ciphertext $c = \left(\hat{\mathbf{v}},\ \hat{w}\right)$ satisfies

$$\hat{w} - \left\langle \hat{\mathbf{v}},\ \hat{\mathbf{s}} \right\rangle = \mu + 2\hat{e} \mod p$$

where $\hat{e} \approx \dfrac{p}{q} \cdot e$ (an appropriately scaled version of e).

- Recall decryption: $\mu^* := \left(\hat{w} - \left\langle \hat{\mathbf{v}},\ \hat{\mathbf{s}} \right\rangle\right) \mod p \mod 2.$

Remark

- The coefficients $h_{i,\tau}$ are obtained as follows.
  - Let $\phi(\mathbf{x}) = \phi_{\mathbf{v},w}(\mathbf{x}) \triangleq \frac{p}{q} \cdot \left(\frac{q+1}{2} \cdot \underbrace{(w - \langle \mathbf{v}, \mathbf{x} \rangle)}_{\bmod q} \right) \bmod p.$
  - Let $h_0, \ \ldots, \ h_n \in \mathbb{Z}_q$ s.t. $\phi(\mathbf{x}) = \sum_{i=0}^{n} h_i \cdot \left(\frac{p}{q} \cdot \mathbf{x}[i] \right) \bmod p$

$$= \sum_{i=0}^{n} \sum_{\tau=0}^{\lfloor \log q \rfloor} h_{i,\tau} \cdot \left(\frac{p}{q} \cdot 2^{\tau} \cdot \mathbf{x}[i] \right) \bmod p$$

- The h_i's in slide 21 are coefficients of $w - \langle \mathbf{v}, \mathbf{x} \rangle \bmod q$.

Security

- Theorem (informal). If (average-case) $\mathrm{DLWE}_{n,q,\chi}$ and $\mathrm{DLWE}_{k,p,\hat{\chi}}$ are both (t,ε)-hard, then the BTS scheme is $\left(t-\mathrm{poly}(\kappa),\ 2(L+1)(2^{-\kappa}+\varepsilon)\right)$-sematically secure.

- (t,ε)-hard: any adversary with running time t may have advantage at most ε.

- Worst-case SVP $\leq$ average-case DLWE $\leq$ BTS.