A Simple (Leveled) Fully Homomorphic Encryption Scheme

And Thoughts on Bootstrapping

The FHE scheme is joint work with Amit Sahai (UCLA) and Brent Waters (UT Austin)

Supported by IARPA contract number D11PC20202

August 15, 2013 | Workshop on Lattices with Symmetry

Our Results

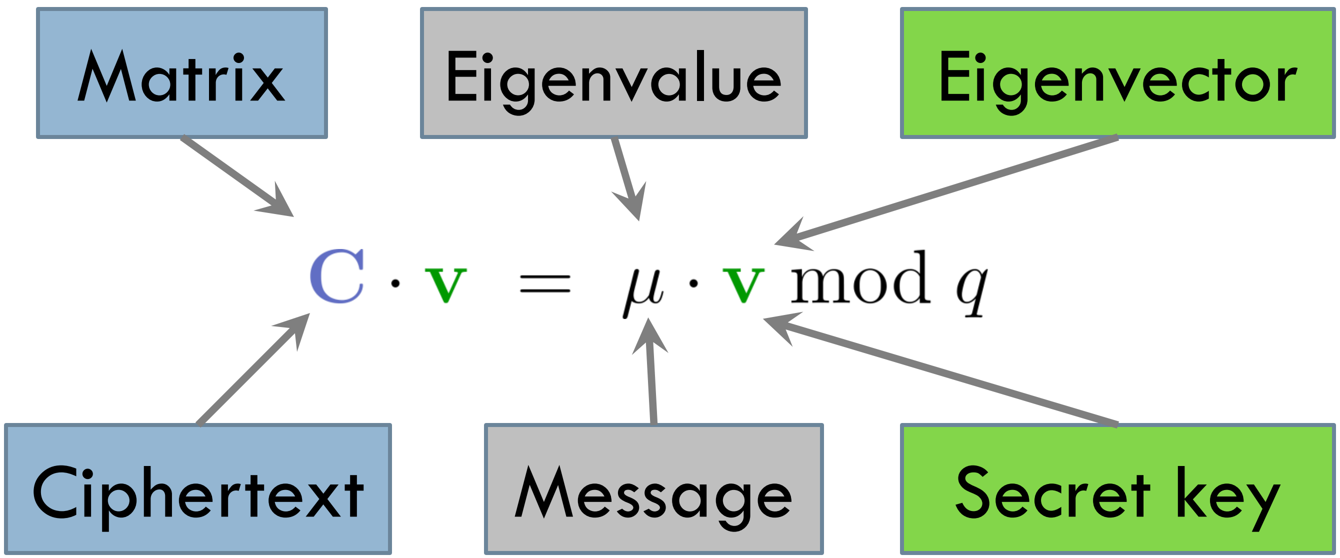
"Leveled" FHE from LWE, with nice properties:

- "Leveled" FHE: Can't go an unbounded # of levels. Can set params to enable any poly(λ) # of levels.
- Conceptual Simplicity: Ciphertexts are matrices. To add or multiply, just add or multiply matrices.
- Asymptotic Advantage: n^{ω} computation per mult
  - $\omega < 2.3727$ is the matrix multiplication constant
  - Previous schemes: "Relinearization" takes n^3 computation

Keep Good Parts of Previous Schemes

- Leveled FHE without bootstrapping [BGV12]
- Security: Based on LWE for quasi-polynomial factors (if you use bootstrapping) [BGV12]

Main Idea: Warm-Up (Toy Scheme)



- **Homomorphism:** Add or multiply ciphertexts.
  Suppose $\mathbf{C_1} \cdot \mathbf{v} = \mu_1 \cdot \mathbf{v} \bmod q$ and $\mathbf{C_2} \cdot \mathbf{v} = \mu_2 \cdot \mathbf{v} \bmod q$.
  Then $\mathbf{C_1} \cdot \mathbf{C_2} \cdot \mathbf{v} = \mu_1 \cdot \mu_2 \cdot \mathbf{v} \bmod q$

Insecurity of Toy Scheme

- Attack using encryptions of 0:

$$\mathbf{C} \cdot \mathbf{v} \;=\; \mathbf{0} \bmod q$$

  Search for $\mathbf{v}$ in the null space of $\mathbf{C}$.

- Attack using any encryptions: Find the eigenvalues and eigenvectors of $\mathbf{C}$ by solving $\mathbf{C}$'s characteristic polynomial: $\det(x \cdot \mathbf{I} - \mathbf{C}) = 0 \bmod q$.

Patching the Toy Scheme

- ~~Method 1: Use multilinear maps to encode (entries of) ciphertext matrix.~~
  - ~~Multilinear map encoding makes it hard to search the null space of $\mathbf{C}$ or compute high-degree determinants.~~
  - ~~Can get somewhat homomorphic encryption this way.~~
- Method 2: The Approximate Eigenvector Method

$$\mathbf{C} \cdot \mathbf{v} \;=\; \mu \cdot \mathbf{v} + \mathbf{e} \bmod q$$

- $\mathbf{e}$ is a noise vector with small coefficients ($\ll q$)
- $\mathbf{v}$ is an *approximate eigenvector*

Approximate Eigenvector Homomorphisms

$$\mathbf{C_1}\cdot\mathbf{v} \;=\; \mu_1\cdot\mathbf{v}+\mathbf{e_1} \bmod q, \qquad \mathbf{C_2}\cdot\mathbf{v} \;=\; \mu_2\cdot\mathbf{v}+\mathbf{e_2} \bmod q$$

- Addition: Set $\mathbf{C}^+ \leftarrow \mathbf{C_1} + \mathbf{C_2} \bmod q$.

$$\mathbf{C}^+ \cdot \mathbf{v} \;=\; (\mu_1 + \mu_2)\cdot \mathbf{v} + (\mathbf{e_1} + \mathbf{e_2}) \bmod q$$

- Multiplication: Set $\mathbf{C}^\times \leftarrow \mathbf{C_1} \times \mathbf{C_2} \bmod q$.

$$\begin{aligned}
\mathbf{C}^\times \cdot \mathbf{v} &= \mathbf{C_1}\cdot(\mu_2\cdot\mathbf{v}+\mathbf{e_2}) \\
&= \mu_2\cdot\mathbf{C_1}\cdot\mathbf{v}+\mathbf{C_1}\cdot\mathbf{e_2} \\
&= \mu_2\cdot(\mu_1\cdot\mathbf{v}+\mathbf{e_1})+\mathbf{C_1}\cdot\mathbf{e_2} \\
&= \mu_1\cdot\mu_2\cdot\mathbf{v}+(\mu_2\cdot\mathbf{e_1}+\mathbf{C_1}\cdot\mathbf{e_2})
\end{aligned}$$

New Noise

Controlling the Noise

New Noise

$$\mathbf{C}^{\times} \cdot \mathbf{v} = \mathbf{C_1} \cdot \mathbf{C_2} \cdot \mathbf{v} = \mu_1 \cdot \lambda_2 \cdot \mathbf{v} + (\mu_2 \cdot \mathbf{e_1} + \mathbf{C_1} \cdot \mathbf{e_2})$$

- **Keep messages small**:
  Easy! Restrict messages to $\{0,1\}$ and use NAND gates.
- **Keep ciphertext entries small**:
  Suppose $\mathbf{C}$ is a product of matrices.
  Can we "Flatten" $\mathbf{C}$ to make its entries small (in $\{0,1\}$)?
- **If we could flatten ciphertexts...**
  - Homomorphic Mults increase noise by factor of at most $n+1$.
  - Can evaluate depth $\Theta(\log_{n+1} q)$ before noise reaches q.
  - Set $q = 2^{n^{\Theta(1)}}$. Then we can evaluate polynomial depth, and obtain a (leveled) FHE scheme.

How to Flatten Ciphertexts

- Notation: $\mathbf{a} \in \mathbb{Z}_q^k, \quad \ell = \lfloor \log q \rfloor + 1, \quad N = k \cdot \ell$
- **Some definitions:**
  - $\mathsf{BitDecomp}(\mathbf{a}) = (a_{1,0}, \ldots, a_{1,\ell-1}, \ldots, a_{k,0}, \ldots, a_{k,\ell-1}) \in \{0,1\}^N$.
    Each coefficient of $\mathbf{a}$ decomposed into bits, least to most significant.
  - $\mathsf{BitDecomp}^{-1}(\mathbf{b} \in \mathbb{Z}_q^N) = (\sum_j 2^j b_{1,j} \bmod q, \ldots, \sum_j 2^j b_{k,j} \bmod q)$.
    $\mathsf{BitDecomp}^{-1}$ is defined even on inputs not in image of $\mathsf{BitDecomp}$.
  - $\mathsf{Flatten}(\mathbf{b} \in \mathbb{Z}_q^N) = \mathsf{BitDecomp}(\mathsf{BitDecomp}^{-1}(\mathbf{b}))$.
    This is a vector with coefficients in $\{0,1\}$.
  - $\mathsf{Powersof2}(\mathbf{s}) = (s_1, 2s_1, \ldots, 2^{\ell-1}s_1, \ldots, s_k, 2s_k, \ldots, 2^{\ell-1}s_k) \bmod q$
- **Some obvious facts:**
  - For any $\mathbf{a}, \mathbf{s} \in \mathbb{Z}_q^k$, it holds that $\langle \mathbf{a}, \mathbf{s} \rangle = \langle \mathsf{BitDecomp}(\mathbf{a}), \mathsf{Powersof2}(\mathbf{s}) \rangle$.
  - For any $\mathbf{b} \in \mathbb{Z}_q^N$ and $\mathbf{s} \in \mathbb{Z}_q^k$,
    $\langle \mathbf{b}, \mathsf{Powersof2}(\mathbf{s}) \rangle = \langle \mathsf{BitDecomp}^{-1}(\mathbf{b}), \mathbf{s} \rangle = \langle \mathsf{Flatten}(\mathbf{b}), \mathsf{Powersof2}(\mathbf{s}) \rangle$

How to Flatten Ciphertexts II

For $\mathbf{b} \in \mathbb{Z}_q^N, \mathbf{s} \in \mathbb{Z}_q^k$, $\langle \mathbf{b}, \mathsf{Powersof2}(\mathbf{s}) \rangle = \langle \mathsf{Flatten}(\mathbf{b}), \mathsf{Powersof2}(\mathbf{s}) \rangle$

- **Give the approximate eigenvector a special form**:
  $\mathbf{v} = \mathsf{Powersof2}(\mathbf{s})$ for some $\mathbf{s}$.
- **Flattening a ciphertext**:
  - Suppose $\mathbf{C}^{\text{NAND}} = \mathrm{I}_N - \mathbf{C_1} \cdot \mathbf{C_2} \bmod q$, for a NAND gate.
  - Set $\mathbf{C_3} \leftarrow \mathsf{Flatten}(\mathbf{C}^{\text{NAND}})$, flattening each row of $\mathbf{C}^{\text{NAND}}$. Each coefficient of $\mathbf{C_3}$ is in $\{0, 1\}$.
  - Then, $\mathbf{C_3} \cdot \mathbf{v} = \mathbf{C}^{\text{NAND}} \cdot \mathbf{v} \bmod q$. We have not changed what is decrypted, or even increased the noise.
- **We have (leveled) FHE!**

KeyGen, Encrypt, and Decrypt

- **Setup**$(1^n, 1^L)$: Set basic parameters q, $\ell = \lfloor \log q \rfloor + 1$, $N = (n+1) \cdot \ell$, $m = O(n \log q)$.
- **KeyGen**(1^n): Generate secret key $\mathbf{s} = (1, \mathbf{t}) \in \mathbb{Z}_q^{n+1}$.
  Secret key: $\mathbf{v} \leftarrow \mathsf{Powersof2}(\mathbf{s}) \in \mathbb{Z}_q^N$.
  Public key: $\mathbf{A} \in \mathbb{Z}_q^{m \times (n+1)}$ is uniform except $\mathbf{e} \leftarrow \mathbf{A} \cdot \mathbf{s}$ "small".
- **Encrypt**$(\mathbf{A}, \mu \in \{0,1\})$: For random $\mathbf{R} \in \{0,1\}^{N \times m}$, output:

$$\mathbf{C} \leftarrow \mathsf{Flatten}(\mu \cdot \mathbf{I}_N + \mathsf{BitDecomp}(\mathbf{R} \cdot \mathbf{A}))$$

- **Decrypt**$(\mathbf{C}, \mathbf{v})$: Compute:

$$\mathbf{C} \cdot \mathbf{v} = \mu \cdot \mathbf{v} + \mathsf{BitDecomp}(\mathbf{R} \cdot \mathbf{A}) \cdot \mathbf{v} = \mu \cdot \mathbf{v} + \mathbf{R} \cdot \mathbf{A} \cdot \mathbf{s} = \mu \cdot \mathbf{v} + \mathit{small}$$

  Recover μ from $2^{\ell-1} \cdot \mu + \mathit{small}$.

Reduction to LWE

- **Search:** Find $\mathbf{t} \in \mathbb{Z}_q^n$ given noisy inner products

$$\mathbf{a_1} \in \mathbb{Z}_q^n, \qquad b_1 = \langle \mathbf{a_1}, \mathbf{t} \rangle + e_1 \bmod q$$
$$\mathbf{a_2} \in \mathbb{Z}_q^n, \qquad b_2 = \langle \mathbf{a_2}, \mathbf{t} \rangle + e_2 \bmod q$$
$$\vdots$$

$\mathbf{a_i}$'s uniform, e_i's are "small" errors (much smaller than q)

- **Decision:** Distinguish $(\mathbf{a_i}, b_i)$ from uniform $(\mathbf{a_i}, b_i)$

Reduction to LWE

- **LWE**: Distinguish whether:
  1. $\mathbf{A} \in \mathbb{Z}_q^{m\times(n+1)}$ is uniform, or
  2. $\mathbf{A} \in \mathbb{Z}_q^{m\times(n+1)}$ is uniform conditioned on $\mathbf{A}\cdot\mathbf{s} \bmod q$ being "small" for some vector $\mathbf{s} = (1, -\mathbf{t}) \in \mathbb{Z}_q^{n+1}$.
- **Public key**: $\mathbf{A} \in \mathbb{Z}_q^{m\times(n+1)}$ is uniform except $\mathbf{e} \leftarrow \mathbf{A}\cdot\mathbf{s}$ "small". Indistinguishable from uniform under LWE.
- **Ciphertexts**: Recall $\mathbf{C} \leftarrow \mathsf{Flatten}(\mu \cdot \mathbf{I}_N + \mathsf{BitDecomp}(\mathbf{R}\cdot\mathbf{A}))$
  - $\mathbf{R}\cdot\mathbf{A}$ looks uniform by LWE and the leftover hash lemma.
  - Rows of $\mathbf{R}\cdot\mathbf{A}$ are "encryptions of 0" in Regev's scheme.
  - $\mathsf{BitDecomp}^{-1}(\mathbf{C})$ hides μ. Therefore so does $\mathbf{C}$.

Review of the Scheme

- **Approximate Eigenvector Method**:
  - $\mathbf{C} \cdot \mathbf{v} = \mu \cdot \mathbf{v} + \mathbf{e} \bmod q$
  - Secret key is $\mathbf{v} = \mathsf{Powersof2}(\mathbf{s})$, an approximate eigenvector
  - μ is the message that is encrypted
  - Homomorphic ops: Add or multiply ciphertexts, then $\mathsf{Flatten}$.
  - Bottom line: We get leveled FHE based on LWE with asymptotically better performance.

Noisiness of Ciphertexts

- Ciphertext noise grows exponentially with depth.
- Hence log q and dimension of ciphertext matrices grow linearly with depth.

Ciphertext Size Reduction

- Modulus reduction [BV11b, BGV12]:
  - Suppose c encrypts m – that is, $m = [[\langle c,v\rangle]_q]_2$.
  - Let's pick $p<q$ and set $c^* = (p/q)\cdot c$, rounded.
  - Maybe it is true that:
    - c^* encrypts m: $m = [[\langle c^*,v\rangle]_p]_2$ (new inner modulus).
    - $|[\langle c,v\rangle]_p| \approx (p/q) \cdot |[\langle c,v\rangle]_q|$ (noise is smaller).
  - This really shouldn't work… but it does…
- Also, dimension reduction: won't go over this.

Modulus Reduction Magic Trick

- Scaling lemma: Let $p < q$ be odd moduli.
  - Given c with $m = [[\langle c,s\rangle]_q]_2$. Set $c' = (p/q)c$. Set c'' to be
    - the integer vector closest to c', such that $c'' = c \bmod 2$.
  - If $|[\langle c,s\rangle]_q| < q/2 - (q/p)\cdot \ell_1(s)$, then:
    - c'' is a valid encryption of m with possibly much less noise!
    - $m = [[\langle c'',s\rangle]_p]_2$, and $|[\langle c'',s\rangle]_p| < (p/q) \cdot |[\langle c,s\rangle]_q| + \ell_1(s)$

Annotated Proof

1. For some k, $[\langle c,s\rangle]_q = \langle c,s\rangle - kq$.
2. $(p/q)|[\langle c,s\rangle]_q| = \langle c',s\rangle - kp$.
3. $|\langle c''-c',s\rangle| < \ell_1(s)$.

1. Imagine $\langle c,s\rangle$ is close to kq.
2. Then $\langle c',s\rangle$ is close to kp.
3. $\langle c'',s\rangle$ also close to kp if s is small.

4. Thus, $|\langle c'',s\rangle - kp| < (p/q)\ |[\langle c,s\rangle]_q| + \ell_1(s) < p/2$.
5. So, $[\langle c'',s\rangle]_p = \langle c'',s\rangle - kp$.
6. Since $c' = c$ and $p = q \bmod 2$, we have $[[\langle c'',s\rangle]_p]_2, = [[\langle c,s\rangle]_q]_2$.

Modulus Reduction: Shortcomings

- Reduces size of modulus (q to p) and size of ciphertext
- *Does not* reduce *ratio* of modulus to noise.

Thoughts on Bootstrapping

Bootstrapping: What Is It?

- So far, we can evaluate bounded depth funcs F:

x_1
x_2
…
x_t

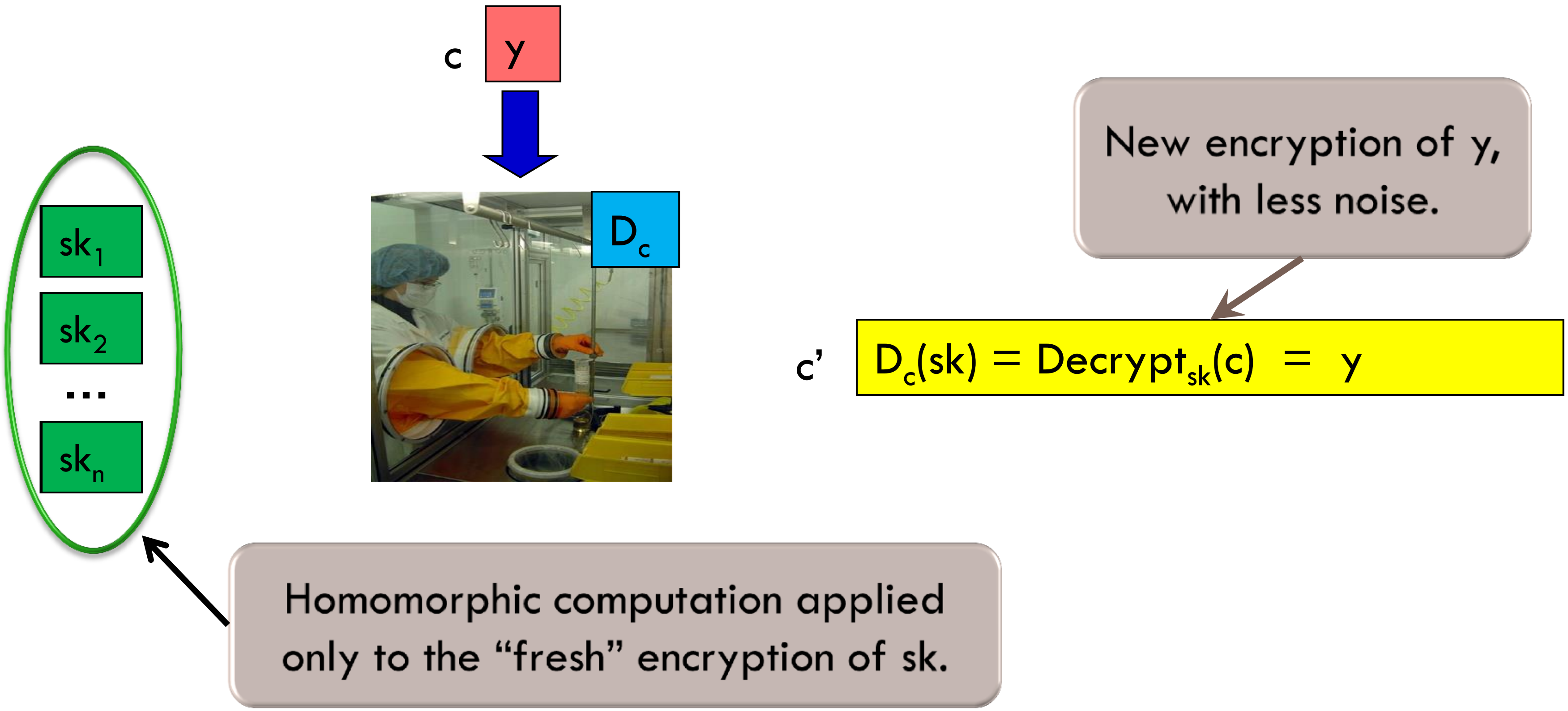
F

c $F(x_1, x_2, \ldots, x_t)$

- We have a noisy evaluated ciphertext c.
- We want to get a less noisy c' that encrypts the same value, but with less noise.
  - Modulus reduction is not enough…
- Bootstrapping *refreshes* ciphertexts, using the *encrypted secret key*.

Bootstrapping: What Is It?

- For ciphertext c, consider $D_c(sk) = \mathrm{Decrypt}_{sk}(c)$
  - Suppose $D_c(\cdot)$ is a low-depth polynomial in sk.
- Include in the public key also $\mathrm{Enc}_{pk}(sk)$.



Bootstrapping: A Mixed Blessing

- Good news: Gives us unbounded depth

- Bad news: Computationally very expensive!
  - Involves running Decrypt circuit *homomorphically*.
  - Decrypt is rather expensive already. Why?
    - Decryption formula must have high (polynomial) degree (log depth).
  - Decrypting with the overhead of homomorphic encryption is too much.

Gentry-Halevi Implementation (Eurocrypt '11): The Somewhat Homomorphic Scheme

Dimension	KeyGen	Enc (amortized)	Dec
512 200,000-bit integers	0.16 sec	4 millisec	4 millisec
2048 800,000-bit integers	1.25 sec	60 millisec	23 millisec
8192 3,200,000-bit integers	10 sec	0.7 sec	0.12 sec
32728 13,000,000-bit integers	95 sec	5.3 sec	0.6 sec

Gentry-Halevi Implementation (Eurocrypt '11): The FHE Scheme

Dimension	KeyGen	PK size	Re-Crypt
512 200,000-bit integers	2.4 sec	17 MByte	6 sec
2048 800,000-bit integers	40 sec	70 MByte	31 sec
8192 3,200,000-bit integers	8 min	285 MByte	3 min
32728 13,000,000-bit integers	2 hours	2.3 GByte	30 min

We Want a New Approach for FHE

- Do we really need "noisy" ciphertexts?
- Can we "refresh" ciphertexts (reduce their noise) without "bootstrapping", or a radically streamlined version of it?
- Can we at least allow q to be only polynomial in the security parameter (rather than quasi-polynomial)?

"Polly Cracker": An Attempt at No-Noise FHE [Fellows-Koblitz '93]

Main Idea

Encryptions of 0 evaluate to 0 at the secret key.

- KeyGen: Secret = some point $\mathbf{s} = (s_1, \ldots, s_n) \in Z_q^n$.
  Public key: Polynomials $\{a_i(x_1,\ldots,x_n)\}$ s.t. $a_i(\mathbf{s})=0 \bmod q$.
- Encrypt: From $\{a_i\}$, generate a *random* polynomial $b(\mathbf{x})$ such that $b(\mathbf{s}) = 0 \bmod q$. For m in $\{0,1\}$, ciphertext is:

$$c(\mathbf{x}) = m + b(\mathbf{x}) \bmod q.$$

- Decrypt: Evaluate ciphertext at secret: $c(\mathbf{s})=m \bmod q$.
- ADD and MULT: Output sum or product of ciphertexts.

Polly Cracker Cryptanalysis

- An Attack if # of monomials in ciphertexts is small:
  - Collect lots of encryptions $\{c_i\}$ of 0.
  - If the challenge ciphertext also encrypts 0, it will likely be in linear span of the given encryptions of 0.
    - Use Gaussian elimination (linear algebra).
- Avoiding the attack:
  - Can # of monomials in ciphertext be exponential?
  - But ciphertext can be efficiently represented?
  - Without introducing other attacks?

Noisy Polly Cracker: A Framework for Most Somewhat Homomorphic Schemes

Main Idea

Encryptions of 0 evaluate to something small and even (smeven) at the secret key.

- KeyGen: Secret = some point $\mathbf{s} = (s_1, \ldots, s_n) \in Z_q^{\,n}$. gcd(q,2)=1. Public key: Polynomials $\{a_i(x_1,\ldots,x_n)\}$ s.t. $a_i(\mathbf{s})=2e_i \bmod q$, $|e_i| \ll q$.
- Encrypt: From $\{a_i\}$, generate a *random* polynomial $b(\mathbf{x})$ such that $b(\mathbf{s})$ = smeven mod q. For m in {0,1}, ciphertext is:

$$c(\mathbf{x}) = m + b(\mathbf{x}) \bmod q.$$

- Decrypt: Evaluate ciphertext at secret: $c(\mathbf{s})$=m+smeven mod q. Then, reduce mod 2 to get m.
- ADD and MULT: Output sum or product of ciphertexts.

Noisy Polly Cracker: A Framework for Most Somewhat Homomorphic Schemes

<u>Main Idea</u>

Encryptions of 0 evaluate to something small and even (smeven) at the secret key.

- KeyGen: Secret = some point $\mathbf{s} = (s_1, \ldots, s_n) \in Z_q^n$. gcd(q,2)=1. Public key: Polynomials $\{a_i(x_1,\ldots,x_n)\}$ s.t. $a_i(\mathbf{s}$

  $\{a_i\}$, generate a *random* polyn

  mod q. For m in {0,1}, ciphertext is:

$$c(\mathbf{x}) = m + b(\mathbf{x}) \bmod q.$$

- Decrypt: <u>Evaluate ciphertext at secret</u>: $c(\mathbf{s})$=m+smeven mod q. Then, reduce mod 2 to get m.
- ADD and MULT: Output sum or product of ciphertexts.

ADDs and MULTs make the "noise" grow.

We call [c($\mathbf{s}$) mod q] the "noise" of the ciphertext.

Confining Noise to Tight Orbits

- Ciphertexts have “noise”
- But want that noise doesn’t grow with # of operations
- Noise remains always in one of two distinct orbits O_0 and O_1, depending on which bit is encrypted.
- Noise maintains high entropy, without growing larger.
  - Can we find make the following maps efficiently computable, even when the orbits have high entropy, and when distinguishing elements of the two orbits is hard?

$$f_{ADD} : O_{m1} \times O_{m2} \rightarrow O_{m1+m2}$$

$$f_{MULT} : O_{m1} \times O_{m2} \rightarrow O_{m1 \times m2}$$

Confining Noise to Tight Orbits

- An Obstacle?
  - (Cohen, Shpilka, Tal): Other than linear polynomials, the min degree of a polynomial $f : [1,n] \to [1,n]$ is $n-o(n)$.
  - Suggests perhaps f_{ADD} and f_{MULT} must have very high degree – not a "simple" transformation.
- But is this really an obstacle?
  - Bootstrapping uses a polynomial of very high degree for free:
    - It decomposes a ciphertext into bits (mod 2) – this is a high-degree transformation viewed modulo $p \neq 2$.
    - Modulus reduction is also a "free" high-degree transformation.

Thank You! Questions?