Homomorphic Encryption and Lattices, Spring 2011 Instructor: Shai Halevi

FHE from LWE, without bootstrapping [BV11, Gen11]

June 9, 2011 Scribe: Reut Levi

1 Starting point: Quadratic-HE

Recall the quadratic-HE (based on Regev's LWE-based scheme) from problem set 4:

- The secret key is a vector $\vec{s} = (1|\vec{s'}) \in \mathbb{Z}_q^{n+1}$ where $\vec{s'} \in_R \mathbb{Z}_q^n$.
- The public key is a $(n+1) \times m$ matrix ($m \geq 3n \log q$) of the form $P = \begin{pmatrix} \vec{b} \\ A \end{pmatrix} \in \mathbb{Z}_q^{(n+1)\times m}$ where $A \in_R \mathbb{Z}_q^{n\times m}$ and $\vec{b} = -\vec{s'}A + 2\vec{e}$ where $\vec{e} \in \lceil \Phi_{\alpha q} \rfloor^m$ is an error vector. Note: P is pseudo-random under D-LWE. Also $\vec{s}P = \vec{b} + \vec{s}A = 2\vec{e}$.
- To encrypt $m \in \{0,1\}$ choose $\vec{r} \in \{0,1\}^m$, output $\vec{c} = P\vec{r} + (m, 0 \ldots 0) \in \mathbb{Z}_q^{n+1}$. Note: If P was random then $\vec{c}$ was also random, independent of P, m.
- To decrypt, set $m' = [\langle \vec{s}, \vec{c} \rangle]_q = [\vec{s}(P\vec{r} + (m, 0 \ldots 0))]_q = [\langle 2\vec{e}, \vec{r} \rangle + \langle \vec{s}, (m, 0 \ldots 0) \rangle]_q$. If $|2\langle \vec{e}, \vec{r} \rangle| + 1 \ll q$ then $m' = 2\langle \vec{e}, \vec{r} \rangle + m$ over the integers, so $m = [m']_2$.

Since decryption is linear (as long as no wraparound occurs) then additive homomorphism (upto error) is immediate. Also, one multiplication can be done via tensor (outer) product. Given two ciphertexts $\vec{c}_1, \vec{c}_2$, can form a "product ciphertext" $C = \vec{c}_1 \otimes \vec{c}_2$. To decrypt C set:

$$m' = [\vec{s}C\vec{s}^t]_q = [\vec{s}(\vec{c}_1 \otimes \vec{c}_2)\vec{s}^t]_q = [\langle \vec{s}, \vec{c}_1 \rangle \cdot \langle \vec{c}_2, \vec{s} \rangle]_q = [(2\langle \vec{e}, \vec{r}_1 \rangle + m_1)(2\langle \vec{e}, \vec{r}_2 \rangle + m_2)]_q$$

If we still have no wraparound then $m' = m_1 m_2 \pmod 2$.

A different view of "product decryption"

Note that decryption is a bilinear form in $\vec{s}$: $\mathsf{Dec}_{\vec{s}}(C) = \vec{s}C\vec{s}^t$. So it can be expressed as a linear operation in the tensor product $\vec{s} \otimes \vec{s}$. Let $\mathsf{vec}(M)$ be the opening of the matrix M into a vector (in some fixed ordering) then $\mathsf{Dec}_{\vec{s}}(\vec{c}_1 \otimes \vec{c}_2) = \langle \mathsf{vec}(\vec{s} \otimes \vec{s}), \mathsf{vec}(\vec{c}_1 \otimes \vec{c}_2) \rangle$. So we have an "extended secret key" $\vec{s}^* = \mathsf{vec}(\vec{s} \otimes \vec{s})$ and an "extended ciphertext" $\vec{c}^* = \mathsf{vec}(\vec{c}_1 \otimes \vec{c}_2)$, but the encryption formula remains $\mathsf{Dec}_{\vec{s}^*} = [\langle \vec{s}^*, \vec{c}^* \rangle]_q \bmod 2$. This means that we can multiply more than just once, but the dimension squares with every such product.

2 Dimension-Reduction

To be able to keep multiplying without the "dimension explosion", we would like to publish some information to allow anyone to convert "extended ciphertexts" (that can be decrypted by "extended secret keys") into "normal ciphertexts" that require only "normal secret key" to decrypt. This can be thought of as a form of proxy re-encryption: we want to publish some information $P(\vec{s}^* \to \vec{t})$ that lets anyone converts a ciphertext under $\vec{s}^*$ into a ciphertext under $\vec{t}$, without breaking semantic security. In our case it is important that the dimension of $\vec{t}$ is much smaller than the dimension of $\vec{s}^*$. Very roughly, we "encrypt $\vec{s}^*$ under $\vec{t}$".

First try

Let $\vec{s}_1 \in \mathbb{Z}_q^{n_1}$ be an "extended secret key" of dimension n_1, and let $\vec{s}_2 = (1|\vec{s'}_2) \in \mathbb{Z}_q^{n_2}$ be a "normal secret key" of dimension n_2. We publish a matrix $P'(\vec{s}_1 \to \vec{s}_2) = \begin{pmatrix} \vec{b} \\ A \end{pmatrix} \in \mathbb{Z}_q^{n_2 \times n_1}$, such that $A \in_R \mathbb{Z}_q^{(n_2-1)\times n_1}$ and $\vec{b}_2 = [-\vec{s'}_2 A_2 + 2\vec{e}_2 + \vec{s}_1]_q \in \mathbb{Z}_q^{n_1}$, with $\vec{e}_2$ an error vector $\vec{e}_2 \leftarrow \lceil \Phi_{\alpha q} \rfloor^{n_1}$. As before $P'(\vec{s}_1 \to \vec{s}_2)$ is pseudorandom under D-LWE (assuming that $\vec{s}_2$ is independent of $\vec{s}_1$). Note that for any vector $\vec{c}_1 \in \mathbb{Z}_q^{n_1}$ we can set $\vec{c}_2 = P'(\vec{s}_1 \to \vec{s}_2) \cdot \vec{c}_1 \in \mathbb{Z}_q^{n_2}$. Moreover, since

$$\vec{s}_2 \cdot P'(\vec{s}_1 \leftarrow \vec{s}_2) = \vec{b}_2 + \vec{s}_2 A_2 = 2\vec{e}_2 + \vec{s}_1 \pmod q ,$$

then

$$\langle \vec{s}_2, \vec{c}_2 \rangle = \vec{s}_2 \cdot P'(\vec{s}_1 \to \vec{s}_2) \cdot \vec{c}_1 = \langle 2\vec{e}_2 + \vec{s}_1, \vec{c}_1 \rangle = 2\langle \vec{e}_2, \vec{c}_1 \rangle + \langle \vec{s}_1, \vec{c}_1 \rangle \pmod q$$

This is almost what we wanted, if we didn't have wraparound the we would get

$$[\langle \vec{s}_2, \vec{c}_2 \rangle]_q = [\langle \vec{s}_1, \vec{c}_1 \rangle]_q \pmod 2 .$$

But $\vec{c}_1$ is a long vector (in Euclidean norm) so even for the short error vector $\vec{e}_2$ we are likely to get a wraparound when taking the inner-product with $\vec{c}_1$.

The fix

We represent a long vector in dimension n_1, via a short vector in dimension even larger - $n_1 \log q$. Denote $\ell = \lceil \log q \rceil$, and let $\vec{u} \in \{0, 1, \ldots, q-1\}^{n_1}$ be some vector with entries smaller than q. Let $\vec{u}_0$ be the vector of lsb's in $\vec{u}$, let $\vec{u}_1$ be the vector of 2'nd bits, etc. Namely $\vec{u} = \sum_{i=0}^{\ell-1} 2^i \vec{u}_i$ and $\vec{u}_i \in \{0,1\}^{n_1}$. Consider the following functions.

$$\mathsf{BitDecomp}(\vec{u}) = (\vec{u}_0 | \vec{u}_1 | \vec{u}_2 \ldots | \vec{u}_{\ell-1}) \in \{0,1\}^{n_1 \cdot \ell}$$

$$\mathsf{Powers2}_q(\vec{s}) = (\vec{s} | [2\vec{s}]_q | [4\vec{s}]_q \ldots [2^{\ell-1}\vec{s}]_q) \in \mathbb{Z}_q^{n_1 \cdot \ell}$$

(so $\|\mathsf{BitDecomp}(\vec{u})\| \le \sqrt{\ell n_1}$). Then for any $\vec{s}, \vec{u} \in \{0, 1, \ldots, q-1\}^{n_1}$ we have

$$\langle \mathsf{Powers2}_q(\vec{s}), \mathsf{BitDecomp}(\vec{u}) \rangle = \sum_{i=0}^{\ell-1} \langle [2^i \vec{s}]_q, \vec{u}_i \rangle = \langle \vec{s}, \sum_{i=0}^{\ell-1} 2^i \vec{u}_i \rangle = \langle \vec{s}, \vec{u} \rangle \pmod q$$

To allow conversion from encryption under $\vec{s}_1 \in \mathbb{Z}_q^{n_1}$ to encryption under $\vec{s}_2 \in \mathbb{Z}_q^{n_2}$, we publish the matrix $P(\vec{s}_1 \to \vec{s}_2) = \begin{pmatrix} \vec{b} \\ A \end{pmatrix} \in \mathbb{Z}_q^{n_2 \times (n_1 \cdot \ell)}$ where $A \in_R \mathbb{Z}_q^{(n_2-1)\times(n_1\cdot\ell)}$ and $\vec{b} = -\vec{s'}_2 A + 2\vec{e}_2 + \mathsf{Powers2}_q(\vec{s}_1)$, where $\vec{e}$ is an error vector $\vec{e} \leftarrow \lceil \Phi_{\alpha q} \rfloor^{n_1 \ell}$. Then we have $\vec{s}_2 \cdot P(\vec{s}_1 \to \vec{s}_2) \equiv_q 2\vec{e} + \mathsf{Powers2}_q(\vec{s}_1)$. Now for a vector $\vec{c}_1 \in \mathbb{Z}_q^{n_1}$ we set $\vec{c}_2 = [P(\vec{s}_1 \to \vec{s}_2) \cdot \mathsf{BitDecomp}(\vec{c}_1)]_q$ and so

$$\begin{aligned} \langle \vec{s}_2, \vec{c}_2 \rangle &= \vec{s}_2 P \cdot \mathsf{BitDecomp}(\vec{c}_1) = \langle 2\vec{e} + \mathsf{Powers2}_q(\vec{s}_1), \mathsf{BitDecomp}(\vec{c}_1) \rangle \\ &= 2\langle \vec{e}, \mathsf{BitDecomp}(\vec{c}_1) \rangle + \langle \mathsf{Powers2}_q(\vec{s}_1), \mathsf{BitDecomp}(\vec{c}_1) \rangle \\ &= 2\langle \vec{e}, \mathsf{BitDecomp}(\vec{c}_1) \rangle + \langle \vec{s}_1 + \vec{c}_1 \rangle \pmod q . \end{aligned}$$

Now both $\vec{e}$, $\mathsf{BitDecomp}(\vec{c}_1)$ are small, we can avoid wraparound and therefore

$$[\langle \vec{s}_2, \vec{c}_2 \rangle]_q = [\langle \vec{s}_1, \vec{c}_1 \rangle]_q \pmod 2 .$$

What do we have so far? SWHE from LWE

- Given $\vec{c}_1, \vec{c}_2$, original ciphertext under $\vec{s}$, can generate an extended ciphertext $\mathsf{vec}(\vec{c}_1 \otimes \vec{c}_2)$. The noise in this extended ciphertext is roughly the product of the noises in the two $\vec{c}_1, \vec{c}_2$.
- If we publish the matrix $P(\mathsf{vec}(\vec{s} \otimes \vec{s}) \to \vec{t})$ then we can convert the extended ciphertext back into a normal ciphertext under $\vec{t}$. The noise only grows by an additive factor of $\approx \|\vec{e}_2\|\sqrt{\log q} \cdot n$.
- Put together we can multiply ciphertexts and the noise grows from size μ to size $\approx \mu^2 + \alpha q n^{2/3}\sqrt{\log q}$.
- To handle d levels (degree 2^d) we need to publish d such conversion matrices $P((\vec{s}_i \otimes \vec{s}_i) \to \vec{s}_{i+1})$.
- The noise grows like μ^{2^d}, so cannot do more than $\log n$ levels.

3 How to get from SWHE to FHE?

Getting from SWHE to FHE can be accomplished in several ways:

1. Squashing + bootstrapping: This works, but need to assume harness of SSSP (sparse-subset-sum problem).
2. [BV11] describe a squashing-less method to reduce decryption complexity:
   - dimension-reduction can reduce the secret key dimension to any n^ϵ, if needed.
   - A new modulus-reduction technique: Instead of $[\langle \vec{s}, \vec{c} \rangle]_q$ do $[\langle \vec{s'}, \vec{c'} \rangle]_p$ for some $p \ll q$. The noise in $\vec{e'}$ is still below $p/2$, but too large for doing more multiplications. But since we work with small dimension and small modulus then decryption has low complexity. So we get a bootstrappable scheme.
3. [Gen11] Show how to reduce the noise so that we can do poly(n) levels.

Gentry's technique for "noise control"

Recall that our decryption formula is $\mathsf{Dec}_{\vec{s}}(\vec{c}) = [[\langle \vec{s}, \vec{c} \rangle]_q]_2$.

Lemma 1. *Let p, q be two odd moduli, and let $\vec{c}, \vec{s} \in \mathbb{Z}_q^n$. Let $\vec{c'} \in \mathbb{Z}^n$ be an integer vector, where the i'th entry of $\vec{c'}$ is just $\frac{p}{q} \cdot \vec{c}_i$, rounded either up or down so that $\vec{c'} \equiv \vec{c} \pmod 2$. If*

$$|[\langle \vec{s}, \vec{c} \rangle]_q| < \frac{q}{2} - \frac{q}{p} \cdot \|\vec{s}\|_1$$

then

$$[\langle \vec{s}, \vec{c'} \rangle]_p = [\langle \vec{s}, \vec{c} \rangle]_q \mod 2$$

and

$$\left|[\langle \vec{s}, \vec{c'} \rangle]_p\right| < \frac{q}{p}\left|[\langle \vec{s}, \vec{c} \rangle]_q\right| + \|\vec{s}_1\|_1 .$$

Proof. We have $[\langle \vec{s}, \vec{c} \rangle]_q = \langle \vec{s}, \vec{c} \rangle - kp$ for some integer k. Consider now the integer $\langle \vec{s}, \vec{c'} \rangle - kp$, and we show that this integer must be in the range $\left(-\frac{p}{2}, +\frac{p}{2}\right)$ and therefore $\langle \vec{s}, \vec{c'} \rangle - kp = [\langle \vec{s}, \vec{c'} \rangle]_p$.

$$\begin{aligned} \left| \langle \vec{s}, \vec{c'} \rangle - kp \right| &= \left| \langle \vec{s}, \frac{p}{q}\vec{c} \rangle + \langle \vec{s}, \vec{c'} - \frac{p}{q}\vec{c} \rangle - kp \right| \\ &\leq \left| \langle \vec{s}, \frac{p}{q}\vec{c} \rangle - kp \right| + \left| \langle \vec{s}, \vec{c'} - \frac{p}{q}\vec{c} \rangle \right| \\ &\leq \frac{p}{q} \left| \langle \vec{s}, \vec{c} \rangle - kq \right| + \|\vec{s}\|_1 < \frac{p}{2} \,. \end{aligned}$$

On the other hand, since $\vec{c'} \equiv \vec{c} \pmod 2$ then also $\langle \vec{s}, \vec{c} \rangle = \langle \vec{s}, \vec{c'} \rangle \mod 2$, and since both p, q are odd then $kp = kq \mod 2$ so we get

$$[\langle \vec{s}, \vec{c'} \rangle]_p = \langle \vec{s}, \vec{c'} \rangle - kp = \langle \vec{s}, \vec{c} \rangle - kp = [\langle \vec{s}, \vec{c} \rangle]_q \mod 2$$

□

Example 1. *Let $\vec{s} = (2, 3)$ and $\vec{c} = (175, 212)$, so $\langle \vec{s}, \vec{c} \rangle = 2 \cdot 175 + 3 \cdot 212 = 986$.*
Let $q = 127$, so $[\langle \vec{s}, \vec{c} \rangle]_q = 986 - 8 \cdot 127 = -30$ ($= 0 \mod 2$), and let $p = 29$, so $\frac{p}{q} \cdot \vec{c} \sim (39.9, 48.4)$.

We need to round to get the first entry odd and the second even, we we have $\vec{c'} = (39, 48)$. Now $\langle \vec{s}, \vec{c'} \rangle = 2 \cdot 39 + 3 \cdot 48 = 222$, hence $[\langle \vec{s}, \vec{c'} \rangle]_p = 222 - 8 \cdot 29 = -10$ ($= 0 \mod 2$).

Note that the "noise" $[\langle \vec{s}, \vec{c'} \rangle]_p$ as a fraction of p is more than the "noise" $[\langle \vec{s}, \vec{c} \rangle]_q$ as a fraction of q, so why is this useful? Because in absolute terms the noise is getting smaller, and this lets us keep it small as we do more multiplications.

Example 2. *Assume that the noise in fresh ciphertexts has magnitude N, and set the size of the initial modulus to (say) $q_0 \sim N^{10}$, and then $q_i \sim N^{10-i}$.*

	without modulus-switching	*using modulus-switching*
fresh ciphertext	*noise/modulus* $= N/N^{10}$	N/N^{10}
level-1, deg=2	N^2/N^{10}	$N^2 \cdot \frac{N^9}{N^{10}} = N/N^9$
level-2, deg=4	N^4/N^{10}	$N^2 \cdot \frac{N^8}{N^9} = N/N^8$
level-3, deg=8	N^8/N^{10}	$N^2 \cdot \frac{N^7}{N^8} = N/N^7$
level-4, deg=16	*decryption error!* N^{16}/N^{10}	$N^2 \cdot \frac{N^6}{N^7} = N/N^6$

When using modulus switching we can evaluate #of layers which is linear in modulus-bit-size/noise-bit-size. Without it we can do only log.

One caveat: Recall that in LWE we can choose the secret $\vec{s} \in_R \mathbb{Z}_q^n$, so actually the "additive term" $\|\vec{s}\|_1$ is huge ($> q_0$).

Reducing the secret-key size

First try. We proved in homework that $\vec{s}$ can be chosen from the error distribution, rather than at random. This yields smaller $\vec{s}$, but small enough. If we have $\alpha = 1/\text{poly}$, then the bit-size of αq_0 is a constant fraction of the bit-size of q_0. Thus every multiplication increases the bit-size of the noise by a constant fraction of $|q_0|$, so we need to reduce the modulus by at least that much, so we cannot do more than constant many levels.

The fix. We again use $\mathsf{BitDecomp}(\cdot)$, this time on the secret $\vec{s}$. This is interleaved with the dimension reduction.

The [Gen11] scheme

We have the security parameter n, and let d be the number of levels in the circuit that we want to evaluate.

- Choose the moduli $q_0, q_1 \dots q_d$, with $q_i/q_{i+1} \geq B$ (where B is a parameter polynomial in n).
- For $i = 0 \dots d$ choose a secret key $\vec{s}_i \in_R \mathbb{Z}_{q_i}^n$ (or from the noise distribution $\lceil \Phi_{\alpha q_i} \rfloor^n$). Let $\vec{t}_i = \mathsf{vec}(\vec{s}_i \otimes \vec{s}_i)$ and $\vec{u}_i = \mathsf{Powers2}_{q_i} \in \{0,1\}^{n^2 \log q_i}$.
- Let $pk = P_0$ which is chosen as in the key-generation of quadratic-HE from the beginning of today's class, except with modulus q_0. Then for $i = 1, \dots, d$ let $P_i = P(\vec{u}_{i-1} \to \vec{s}_i) \in \mathbb{Z}_{q_i}^{n \times (n^2 \log^2 q_i)}$.
- Given $\vec{c}_1$, $\vec{c}_2$ that we want to multiply, both encrypted with respect to the same level i, do the following
  - Multiply: compute $\vec{c}^{\otimes} = \mathsf{vec}(\vec{c}_1 \otimes \vec{c}_2)$, which is decrypted under $\vec{t}_i$ to the correct value, modulo q_i.
  - Expand: Set $\vec{d} = \mathsf{Powers2}_{q_i}(\vec{c}^{\otimes}) \in \mathbb{Z}_q^{n^2 \log q_i}$, which is decrypted under $\vec{u}_i$ to the correct value, modulo q_i.
  - Switch-modulus: Set $\vec{d}' = \mathsf{round}\left(\frac{q_{i+1}}{q_i} \cdot \vec{d}\right)$, where rounding is so that $\vec{d}' \equiv \vec{d} \pmod 2$. Note that since $\vec{u}$ is a 0-1 vector (and so $\|\vec{u}\|_1 < n^2 \log q_i$) then the noise in $\vec{d}'$ is small, and so $\vec{d}'$ is decrypted under $\vec{u}_i$ to the correct value modulo q_{i+1}.
  - Reduce-dimension: Set $\vec{c} = P(\vec{u}_i \to \vec{s}_{i+1}) \cdot \mathsf{BitDecomp}(\vec{d}')$, so now $\vec{c}$ is decrypted under $\vec{s}_{i+1}$ to the correct value modulo q_{i+1}

If the noise in $\vec{c}_1, \vec{c}_2$ is bounded by some μ, then the noise in $\vec{c}^{\otimes}$ is bounded by μ^2, and so is the noise in $\vec{d}$. Then the noise in $\vec{d}'$ is bounded by $\mu^2/B + n^2 \log q_i$, and the noise in the final $\vec{c}$ is bounded by $\mu^2/B + n^2 \log q_i + n^2 \log^2 q_i \cdot (\alpha q_{i+1})$. If α is small enough and B is large enough then this can be made at most μ, so the noise does not increase!!

References

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. Manuscript, 2011.

[Gen11] Craig Gentry. Fully homomorphic encryption without bootstrapping. Cryptology ePrint Archive, Report 2011/277, 2011.