Graded Encoding Schemes: Survey of Recent Attacks

Shai Halevi (IBM Research)
NYC Crypto Day
January 2015
Graded Encoding Schemes (GES)

- Very powerful crypto tools
  - Resembles “Cryptographic Multilinear Maps”
- Enable computation on “hidden data”
  - Similar to homomorphic encryption (HE)
- But HE is too “all or nothing”
  - No key: result is meaningless
  - Has key: can read result and intermediate values
Graded Encoding Schemes (GES)

- Leak “some information” about result
  - Can tell if results equals zero
  - Not decrypt result or intermediate values
- This partial leakage can do great things
  - Multipartite non-interactive key-exchange, Witness-encryption, Attribute-based encryption, Cryptographic code obfuscation, Functional encryption, ...
- But implementing “limited leakage” is messy
Plan for this Talk

- **Background**
  - Some details of [GGH13], [CLT13]
  - The [GGH13] “zeroizing” attack

- **New attacks (Cheon, Han, Lee, Ryu, Stehle’14)**
  - Extensions of the attacks (Coron, Gentry, H, Lepoint, Maji, Miles, Raykova, Sahai, Tibouchi’15)
  - Limitations of attacks

- **Tentative conclusions**
Constructing GES

The GGH Recipe:

- Start from some HE scheme
  - Publish a “defective secret key”
    - Called “zero-test parameter”
  - Can be used to identify encryptions of zero
    - Cannot be used for decryption
- Instantiated from NTRU in [GGH13], from approximate-GCD in [CLT13]
  - Another proposal in [GGH14] (but not today)
The [GGH13] Construction

- Works in polynomial rings $R = \mathbb{Z}[X]/F_n(X)$
  - Also $R_q = R/qR = \mathbb{Z}_q[X]/F(X)$
  - $q$ is a “large” integer (e.g., $q \approx 2^{\sqrt{n}}$)

- Secrets are $z \in \mathbb{R}_q$ and a “small” $g \in R$

- “Plaintext space” is $R_g = R/gR$

- Level-$i$ encoding of $\alpha \in R_g$ is of form $[\frac{e}{z^i}]_q$
  - $e$ is a “small” element in the $g$-coset of $\alpha$
The [GGH13] Construction

- Secrets are $z \in \mathbb{R}_q$ and a "small" $g \in \mathbb{R}$
- "Plaintext space" is $R_g = \mathbb{R}/g\mathbb{R}$
- Level-$i$ encoding of $\alpha \in R_g$ is of form $[e/z^i]_q$
  - $e$ is a "small" element in the $g$-coset of $\alpha$
- Can add, multiply encodings:
  $$[\text{enc}_i(\alpha) + \text{enc}_i(\beta)]_q = \text{enc}_i(\alpha + \beta)$$
  $$[\text{enc}_i(\alpha) \cdot \text{enc}_j(\beta)]_q = \text{enc}_{i+j}(\alpha\beta)$$
  - As long as $e$ remains smaller than $q$
The [GGH13] Zero-Test

- Level-k encoding of zero is \( u = \left[ \frac{r \cdot g}{z^k} \right]_q \)
- Zero-test parameter is \( p_{zt} = \left[ \frac{h z^k}{g} \right]_q \)
  - \( h \) is small-ish
- Multiplying we get \( \left| u \cdot p_{zt} \right|_q = \left| r \cdot h \right| \ll q \)
  - Because both \( r, h \) are small
- If \( u = enc_k(\alpha \neq 0) \) then \( \left| e \cdot p_{zt} \right|_q \approx q \)
The [CLT13] Construction

- Similar idea, but using CRT representation modulo a composite integer $N = p_1 \cdot \ldots \cdot p_t$
  - Assuming that factoring $N$ is hard
  - The $p_i$’s are all the same size

- Secrets are $p_i$’s, $z \in_{\$} Z_N$, and $g_i \ll p_i$’s

- “Plaintext space” consists of $t$-vectors $(\alpha_1, \alpha_2, \ldots, \alpha_t) \in Z_{g_1} \times Z_{g_2} \times \ldots \times Z_{g_t}$
The [CLT13] Construction

- Level-i encoding of vector \((\alpha_1 ... \alpha_t)\) has the form \(\left[\text{CRT}(e_1, ..., e_t)/z^i\right]_N\), where \(e_i = r_i g_i + \alpha_i\)
- \(e_i\)'s are small element in the \(g_i\)-cosets of \(\alpha_i\)'s

\(\text{CRT}(e_1, ..., e_t)\) is the element mod \(N\) with this CRT decomposition
The [CLT13] Construction

- Level-\(i\) encoding of vector \((\alpha_1 \ldots \alpha_t)\) has the form \(\left[\frac{\text{CRT}(e_1, \ldots, e_t)}{z_i}\right]_N\), where \(e_i = r_i g_i + \alpha_i\)
  - \(e_i\)'s are small element in the \(g_i\)-cosets of \(\alpha_i\)'s

- Can add, multiply encodings
  \[
  \left[\text{enc}_i(\overrightarrow{\alpha}) + \text{enc}_i(\overrightarrow{\beta})\right]_q = \text{enc}_i(\overrightarrow{\alpha + \beta})
  \]
  \[
  \left[\text{enc}_i(\overrightarrow{\alpha}) \cdot \text{enc}_j(\overrightarrow{\beta})\right]_q = \text{enc}_{i+j}(\overrightarrow{\alpha \beta})
  \]
  - As long as the \(e_i\)'s remain smaller than the \(p_i\)'s
The [CLT13] Zero-Test

- Let $p_i^* \overset{\text{def}}{=} \frac{N}{p_i}$, $i = 1, \ldots, t$

- Observation: Fix any $(e_1, \ldots, e_t)$. Then

$$\text{CRT}(p_1^*e_1, \ldots, p_t^*e_t) = \sum_i p_i^* e_i \mod N$$

- The CLT zero-test parameter is

$$p_{zt} = \left[\text{CRT}(p_1^* h_1 g_1^{-1}, \ldots, p_t^* h_t g_t^{-1}) \cdot z^k\right]_N$$

- $|h_i| \ll p_i$
The [CLT13] Zero-Test

- \( p_{zt} = \left[ CRT(p_1^* h_1 g_1^{-1}, \ldots, p_t^* h_t g_t^{-1}) \cdot z^k \right]_N \)

- An encoding of \((0, \ldots, 0)\) at level \(k\) has the form \( u = \left[ CRT(r_1 g_1, \ldots, r_t g_t) / z^k \right]_N \)

  - So \( u \cdot p_{zt} = CRT(p_1^* h_1 r_1, \ldots, p_t^* h_t r_t) = \sum_i p_i^* h_i r_i \)

  - \(|h_i r_i| \ll p_i\), and therefore \(|p_i^* h_i r_i| \ll N\)

  - The sum is still much smaller than \(N\)

- If \(u\) is an encoding of non-zero at level \(k\) then \(|u \cdot p_{zt}| \approx N\)
Common properties of GGH, CLT

- Plaintext is a vector of elements
  - Size-1 vector in GGH
  - There is also a GGH variant with longer vectors
- An encoding $u$ of $(\alpha_1, ..., \alpha_t)$ is “related” to a vector $(e_1, ..., e_t)$ with $e_i = r_i g_i + \alpha_i$
  - We will write $u \sim (e_1, ..., e_t)$
  - Finding the $e_i$’s means breaking the scheme
- Add/mult act on the $e_i$’s over the integers
  - No modular reduction
Common properties of GGH, CLT

- If \( u \) is an encodings of zero at the top level
  - \( u \sim (r_1g_1, \ldots, r_tg_t) \)
- then by zero-testing we get \( ztst(u) = \sum_i \sigma_i r_i \)
  - \( \sigma_i \)'s are system parameters, independent of \( u \)
    - \( \sigma = h \) for GGH, \( \sigma_i = p_i^*h_i \) for CLT
- The computation is over the integers, without modular reduction

(If \( u \) encodes non-zero then we do not get an equality over the integers)
Attacks
The [GGH13] “zeroizing” attack

- Say we have level-\(i\) GGH encoding of zero
  - \(u_0 \sim (r_0 g)\)
- … and many other level-\((k - i)\) encodings
  - \(u_j \sim (e_j)\)
- Then \(u_0 u_j \sim (e_j r_0 g)\), using zero-test we get
  \[
  y_j = ztst(u_0 u_j) = hr_0 \cdot e_j
  \]
- We recover the \(e_j\)'s upto the factor \(h' = hr_0\)
- Can compute GCDs to find, remove \(h'\)
The [GGH13] “zeroizing” attack

- This attack does not work for CLT
  - At least not “out of the box”
  - Also doesn’t work on the “vectorised” GGH variant
- We have vectors $u_j \sim (e_{j,1}, \ldots, e_{j,t})$
- Applying the same procedure gives the inner products $y_j = \sum_i r_{0,i} \sigma_i \cdot e_{j,i}$
  - Only one $y_j$ per vector of $e_{j,i}$’s
  - Not enough to do GCD’s
The Cheon et al. Attack [CHLRS14]

- A major “upgrade” of the [GGH13] attack
- When applicable, completely breaks CLT
  - i.e., you can factor $N$, learn all the plaintext
- Also works for the “vectorised” GGH
  - Not a complete break, but as severe as zeroizing attacks on the non-vectorised GGH
The Cheon et al. Attack [CHLRS14]

• Say we have many level-$i$ zero-encodings
  • $u_j \sim (a_{j,1}g_1, \ldots, a_{j,t}g_t)$, $j = 1, 2, \ldots$
• … two level-$i'$ encodings
  • $v \sim (b_1, \ldots, b_t)$, $v' \sim (b'_1, \ldots, b'_t)$
• … and many encodings at level $k - i - i'$
  • $w_j \sim (c_{j,1}, \ldots, c_{j,t})$, $j = 1, 2, \ldots$
• For each $j_1, j_2$, we have a level-$k$ encoding
  • $u_{j_1}v w_{j_2} \sim (a_{j_1,1}b_1c_{j_2,1} \cdot g_1, \ldots, a_{j_1,t}b_t c_{j_2,t} \cdot g_t)$
  • Similarly for $u_{j_1}v' w_{j_2}$
The Cheon et al. Attack [CHLRS14]

- Zero-testing we get
  - \( y_{j_1,j_2} = \text{ztst}(u_{j_1} v w_{j_2}) = \sum_i a_{j_1,i} b_i c_{j_2,i} \cdot \sigma_i \)
  - Similarly for \( y'_{j_1,j_2} = \text{ztst}(u_{j_1} v' w_{j_2}) \)

- In vector form: \( y_{j_1,j_2} = \)

\[
(a_{j_1,1}, \ldots, a_{j_1,t}) \times \begin{pmatrix}
b_1 \sigma_1 \\
0 \\
\vdots \\
b_t \sigma_t
\end{pmatrix} \times \begin{pmatrix}
c_{j_2,1} \\
\vdots \\
c_{j_2,t}
\end{pmatrix}
\]
The Cheon et al. Attack [CHLRS14]

- Zero-testing we get
  - \( y_{j_1,j_2} = \text{ztst}(u_{j_1} v w_{j_2}) = \sum_i a_{j_1,i} b_i c_{j_2,i} \cdot \sigma_i \)
  - Similarly for \( y'_{j_1,j_2} = \text{ztst}(u_{j_1} v' w_{j_2}) \)

- In vector form: \( y_{j_1,j_2} = \)
  
  \[
  \begin{pmatrix}
  (a_{j_1,1}, \ldots, a_{j_1,t}) \times & V \\
  \end{pmatrix}
  \]
The Cheon et al. Attack [CHLRS14]

- Putting the $y_{j_1,j_2}$’s in a $t \times t$ matrix we get
  \[ Y = \begin{bmatrix} y_{j_1,j_2} \end{bmatrix} = U \times V \times W \]
  - $U$ has the $u_{j_1}$”s as rows
  - $V$ is as before
  - $W$ has the $w_{j_2}$’s as columns

- Similarly $Y' = \begin{bmatrix} y'_{j_1,j_2} \end{bmatrix} = U \times V' \times W$

- We know $Y, Y'$ but not $U, V, V', W$

- Importantly, equalities hold over the integers Whp $U, V, W$ are invertible
The Cheon et al. Attack [CHLRS14]

Once we have $Y, Y'$ we compute

$$Z = Y^{-1} \times Y' = (UVW)^{-1} \times (UV'W) = W^{-1} \times (V^{-1} \times V') \times W$$

Recall that $V^{-1} \times V' = \begin{pmatrix} b'_1/b_1 & \cdots & 0 \\ 0 & \ddots & \vdots \\ 0 & \cdots & b'_t/b_t \end{pmatrix}$

- Eigenvalues of $V^{-1} \times V'$ are $b'_i/b_i, \ i = 1, \ldots, t$
- Same for $Z$ (since $V^{-1} \times V', Z$ are similar)
The Cheon et al. Attack [CHLRS14]

- After computing $Z$, compute its eigenvalues $\{b'_i/b_i : i = 1, ..., t\}$
  - We get $b_i, b'_i$ upto the factor $\text{GCD}(b_i, b'_i)$

- Often knowing the ratios $b'_i/b_i$ is enough to violate hardness assumption

- For CLT, can use $b'_i/b_i$ to factor $N$:
The Cheon et al. Attack [CHLRS14]

For CLT, can use $b_i'/b_i$ to factor $N$:

- Recall $\nu = \left[\text{CRT}(b_1, ..., b_i, ..., b_t)/z^{i'}\right]_N$
  $\nu' = \left[\text{CRT}(b_1', ..., b_i', ..., b_t')/z^{i'}\right]_N$

- Express $b_i'/b_i$ as a simple fraction $b_i'/b_i = d_i'/d_i$
  - $d_i, d_i'$ are co-prime

- $x_i = [d_i\nu' - d_i'\nu]_N$ has 0 CRT component for $p_i$
  - Whp the other CRT components are not zero

$\Rightarrow$ Recover $p_i = \text{GCD}(N, x_i)$
Extending the Attack

- Easy to see that the same attack still works as long as $u_{j_1} \cdot v \cdot w_{j_2}$ and $u_{j_1} \cdot v' \cdot w_{j_2}$ are encoding of zeros for every $j_1, j_2$
- Don’t need the $u_{j_1}$’s themselves to encode zero
- e.g.

\[ u_j \sim (a_{j,1} g_1, a_{j,2}, a_{j,3}), \]
\[ v \sim (b_1, b_2 g_2, b_3) \text{ and } v' \sim (b'_1, b'_2 g_2, b'_3), \]
\[ w_j \sim (c_{j,1}, c_{j,2}, c_{j,3} g_3) \]
Attack Consequences
Some Schemes are Broken

- For example, schemes that publish low-level encoding of zeros are likely broken
  - Publishing zero-encoding would be useful
  - E.g., to re-randomeize encodings by adding a subset-sum of these zero encodings

- Even some obfuscation schemes
  - E.g., the “simple IO scheme” from [Zim14] (this requires further extending the attacks)
Many Assumptions are Broken

- “Source Group” assumptions:
  - Given level-1 encodings of elements $\alpha_1, \alpha_2, \ldots$, cannot tell if $\text{expr}(\vec{\alpha}) = 0$
  - $\text{expr}(\ast)$ has degree $\leq k - 3$ (say)
- Generally broken, use the attack with
  - $u_j \sim \text{expr}(\vec{\alpha}) \cdot \alpha_j$
  - $v \sim \alpha_1, v' \sim \alpha_2$
  - $w_j \sim \alpha_j$
Many Assumptions are Broken

- Subgroup-Membership assumptions:
  - Input: encoding of \((\alpha, $, ..., $, 0, ..., 0)\)
    - And some other encodings too
  - Goal: distinguish \(\alpha = 0\) from \(\alpha = $\)
    - Would be easy if we could get an encoding of \((*, 0, ..., 0, \phi, ..., \phi)\)
    - Assumption: it is hard otherwise
  - Broken if we can get encoding of the form \((0, 0, ..., 0, \phi, ..., \phi)\)
Many Assumptions are Broken

- Currently we have no candidate GES with hard source-group or subgroup-membership problems
A Suggested Fix

• Instead of $u_j v w_j \sim \vec{0}$, maybe we can use
  $$\delta = u_j v w_j - \hat{u}_j \hat{v} \hat{w}_j \sim \vec{0}$$

• For encodings $u, v, w$ and $\hat{u}, \hat{v}, \hat{w}$

• This was suggested as a fix to the attacks
  • It is always possible to convert $u_j v w_j \sim \vec{0}$ to get the weaker condition [BWZ14]
  • Similar fix mentioned in [GGHZ14]

• But the attack can be extended to defeat it
Further Extending the Attack

- We mount the same attack, using vectors of double the length
  \[ ztst(\delta) = \left( \sum_i a_{j1,i} b_i c_{j2,i} \cdot \sigma_i - \sum_i \hat{a}_{j1,i} \hat{b}_i \hat{c}_{j2,i} \cdot \sigma_i \right) / g \]

- Similar to before, but now we have \( 1/g \) factor
  - \( g = CRT(g_1, ..., g_t) \) in CLT

- Equality holds over the integers/rationals!

- So \( Y = U \times V \times W \cdot \frac{1}{g} \), and the same for \( Y' \)

- When setting \( Z = Y^{-1} \times Y' \), the \( \frac{1}{g} \) falls off
Limitations of the Attacks

- Rely on partitioning $y_{j_1,j_2} = u_{j_1} \cdot v \cdot w_{j_2} \sim \vec{0}$
  - We can vary $u_{j_1}$ without affecting $v, w_{j_2}$
  - Similarly can vary $w_{j_2}$ without affecting $v, u_{j_1}$
- Many applications do not give such nicely partitioned encoding of zeros
  - E.g., [GGHRSW13] use Barrington BPs
    - You get encoding of zeros in the form $\vec{u} \times \prod_i V_i \times \vec{w}$
    - But changing any bit in the input affects many $V_i$’s
  - Some applications have explicit binding factors
Final Musings About Security

- Current Graded Encoding Schemes “hide” encoded values behind mod-$q$ relations
  - Solving mod-$q$ relations directly involves solving lattice problems (since we need small solutions)
- But zero-test parameter lets you “strip” the mod-$q$ part, get relations over the integers
  - No more lattice problems, any solution will do
  - Can only get these relations when you have an encoding of zero
Final Musings About Security

- Security relies on the adversary’s inability to solve these relations
  - By the time you get a zero, the relations are too complicated to solve

- Security feels more like HFE than FHE
  - HFE: Hidden Field Equations
  - FHE: Fully-Homomorphic Encryption

- It’s going to be a bumpy ride..