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 $R = Z[X]/F_n(X)$
  - Also $R_q = R/qR = Z_q[X]/F(X)$
  - q is a "large" integer (e.g., $q \approx 2^{\sqrt{n}}$)
- Secrets are $\mathbf{z} \in_{\$} \boldsymbol{R_q}$ and a "small" $\boldsymbol{g} \in \mathbf{R}$
- "Plaintext space" is $\boldsymbol{R_g} = \mathbf{R}/\boldsymbol{g}\mathbf{R}$
- Level-i encoding of $\boldsymbol{\alpha} \in \boldsymbol{R_g}$ is of form $\left[{}^{\boldsymbol{e}}/_{\mathbf{z}^{\boldsymbol{i}}}\right]_{\boldsymbol{q}}$
  - e is a "small" element in the g-coset of α

The [GGH13] Construction

- Secrets are $\boldsymbol{z} \in_{\$} \boldsymbol{R_q}$ and a "small" $\boldsymbol{g} \in \boldsymbol{R}$
- "Plaintext space" is $\boldsymbol{R_g} = \boldsymbol{R}/\boldsymbol{gR}$
- Level-i encoding of $\boldsymbol{\alpha} \in \boldsymbol{R_g}$ is of form $\left[{}^{\boldsymbol{e}}/_{\boldsymbol{z^i}}\right]_{\boldsymbol{q}}$
  - e is a "small" element in the g-coset of α
- Can add, multriply encodings:
  $$[\mathbf{enc_i}(\boldsymbol{\alpha}) + \mathbf{enc_i}(\boldsymbol{\beta})]_\mathbf{q} = \mathbf{enc_i}(\boldsymbol{\alpha} + \boldsymbol{\beta})$$
  $$\left[\mathbf{enc_i}(\boldsymbol{\alpha}) \cdot \mathbf{enc_j}(\boldsymbol{\beta})\right]_\mathbf{q} = \mathbf{enc_{i+j}}(\boldsymbol{\alpha\beta})$$
  - As long as e remains smaller than q

The [GGH13] Zero-Test

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

The [CLT13] Construction

- Similar idea, but using CRT representation modulo a composite integer $\boldsymbol{N = p_1 \cdot \ldots \cdot p_t}$
  - Assuming that factoring N is hard
  - The p_i's are all the same size
- Secrets are $\boldsymbol{p_i}$'s, $\mathbf{z} \in_\$ \boldsymbol{Z_N}$, and $\boldsymbol{g_i} \ll \boldsymbol{p_i}$'s
- "Plaintext space" consists of t-vectors $\boldsymbol{(\alpha_1, \alpha_2, \ldots, \alpha_t) \in Z_{g_1} \times Z_{g_2} \times \cdots \times Z_{g_t}}$

The [CLT13] Construction

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

$\mathbf{CRT}(\boldsymbol{e_1, \ldots, e_t})$ **is the element mod** $\boldsymbol{N}$ **with this CRT decomposition**

The [CLT13] Construction

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

- Can add, multiply encodings

$$\left[\mathbf{enc}_{\boldsymbol{i}}(\vec{\boldsymbol{\alpha}}) + \mathbf{enc}_{\boldsymbol{i}}(\vec{\boldsymbol{\beta}})\right]_{\boldsymbol{q}} = \mathbf{enc}_{\boldsymbol{i}}(\overrightarrow{\boldsymbol{\alpha} + \boldsymbol{\beta}})$$

$$\left[\mathbf{enc}_{\boldsymbol{i}}(\vec{\boldsymbol{\alpha}}) \cdot \mathbf{enc}_{\boldsymbol{j}}(\vec{\boldsymbol{\beta}})\right]_{\boldsymbol{q}} = \mathbf{enc}_{\mathbf{i}+\boldsymbol{j}}(\overrightarrow{\boldsymbol{\alpha\beta}})$$

  - As long as the e_i's remain smaller than the p_i's

The [CLT13] Zero-Test

- Let $\boldsymbol{p_i^*} \stackrel{\text{def}}{=} \boldsymbol{\frac{N}{p_i}}$, $i = 1, \dots, t$

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

$$\mathbf{CRT}(\boldsymbol{p_1^* e_1, \dots, p_t^* e_t}) = \boldsymbol{\sum_i p_i^* e_i \bmod N}$$

- The CLT zero-test parameter is

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

  - $|h_i| \ll p_i$

The [CLT13] Zero-Test

- $\boldsymbol{p_{zt} = \left[CRT\left(p_1^* h_1 g_1^{-1}, \ldots, p_t^* h_t g_t^{-1}\right) \cdot z^k\right]_N}$
- An encoding of $(0, \ldots, 0)$ at level k has the form $\boldsymbol{u = \left[{}^{\mathrm{CRT}(\mathrm{r}_1 g_1, \ldots, r_t g_t)} / {}_{z^k}\right]_N}$
  - So $\boldsymbol{u \cdot p_{zt} = CRT(p_1^* h_1 r_1, \ldots, p_t^* h_t r_t) = \sum_i p_i^* h_i r_i}$
    - $|h_i r_i| \ll p_i$, 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 $(\alpha_1, \ldots, \alpha_t)$ is “related” to a vector $(e_1, \ldots, e_t)$ with $e_i = r_i g_i + \alpha_i$
  - We will write $\boldsymbol{u} \sim (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_t)$
  - 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
  - $\boldsymbol{u} \sim (\boldsymbol{r_1 g_1}, \dots, \boldsymbol{r_t g_t})$
- then by zero-testing we get $\mathbf{ztst}(\boldsymbol{u}) = \sum_i \boldsymbol{\sigma_i r_i}$
  - $\boldsymbol{\sigma_i}$'s are system parameters, independent of $\boldsymbol{u}$
    - $\boldsymbol{\sigma} = \boldsymbol{h}$ for GGH, $\boldsymbol{\sigma_i} = \boldsymbol{p_i^* h_i}$ 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
  - $\boldsymbol{u_0} \sim (\boldsymbol{r_0 g})$
- … and many other level-$(k-i)$ encodings
  - $\boldsymbol{u_j} \sim (\boldsymbol{e_j})$
- Then $u_0 u_j \sim \left(e_j r_0 g\right)$, using zero-test we get

$$\boldsymbol{y_j = ztst(u_0 u_j) = hr_0 \cdot e_j}$$

  - 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 $\boldsymbol{u_j} \sim (\boldsymbol{e_{j,1}}, \ldots, \boldsymbol{e_{j,t}})$
- Applying the same procedure gives the inner products $\boldsymbol{y_j} = \sum_i \boldsymbol{r_{0,i}} \boldsymbol{\sigma_i} \cdot \boldsymbol{e_{j,i}}$
  - 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
  - $\mathbf{u_j} \sim \left(\boldsymbol{a}_{\mathbf{j,1}}\mathbf{g_1}, \ldots, \boldsymbol{a}_{\mathbf{j,t}}\mathbf{g_t}\right),\ \mathbf{j = 1, 2, \ldots}$
- … two level-i' encodings
  - $\boldsymbol{v} \sim (\boldsymbol{b}_{\mathbf{1}}, \ \ldots \ , \ \boldsymbol{b}_{\mathbf{t}}), \boldsymbol{v}' \sim (\boldsymbol{b}'_{\mathbf{1}}, \ \ldots \ , \ \boldsymbol{b}'_{\boldsymbol{t}})$
- … and many encodings at level $k - i - i'$
  - $\mathbf{w_j} \sim \left(\boldsymbol{c}_{\mathbf{j,1}}, \ \ldots \ , \ \boldsymbol{c}_{\mathbf{j},\boldsymbol{t}}\right),\ \mathbf{j = 1, 2, \ldots}$
- For each j_1, j_2, we have a level-k encoding
  - $u_{j_1} v\, w_{j_2} \sim (a_{j_1,1} b_1 c_{j_2,1} \cdot g_1, \ldots, a_{j_1,t} b_t c_{j_2,t} \cdot g_t)$
  - Similarly for $u_{j_1} v'\, w_{j_2}$

The Cheon et al. Attack [CHLRS14]

- Zero-testing we get
  - $\boldsymbol{y_{j_1,j_2} = \mathrm{ztst}\left(\mathrm{u_{j_1}\,v\,w_{j_2}}\right) = \sum_\mathrm{i} a_{\mathrm{j_1,i}} b_\mathrm{i} c_{\mathrm{j_2,i}} \cdot \sigma_\mathrm{i}}$
  - Similarly for $y'_{j_1,j_2} = \mathrm{ztst}\left(u_{j_1} v'\, w_{j_2}\right)$
- In vector form: $y_{j_1,j_2} =$

$$\left(a_{j_1,1}, \dots, a_{j_1,t}\right) \times \begin{pmatrix} b_1\sigma_1 & & 0 \\ & \ddots & \\ 0 & & b_t\sigma_t \end{pmatrix} \times \begin{pmatrix} c_{j_2,1} \\ \vdots \\ c_{j_2,t} \end{pmatrix}$$

The Cheon et al. Attack [CHLRS14]

- Zero-testing we get
  - $\boldsymbol{y_{j_1,j_2} = \mathrm{ztst}\left(u_{j_1}\, v\, w_{j_2}\right) = \sum_i a_{j_1,i} b_i c_{j_2,i} \cdot \sigma_i}$
  - Similarly for $y'_{j_1,j_2} = \mathrm{ztst}\left(u_{j_1}\, v'\, w_{j_2}\right)$
- In vector form: $y_{j_1,j_2} =$

$$\underbrace{\left(a_{j_1,1}, \ldots, a_{j_1,t}\right)}_{\overrightarrow{u_{j_1}}} \times \underbrace{\begin{pmatrix} b_1\sigma_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & b_t\sigma_t \end{pmatrix}}_{\boldsymbol{V}} \times \underbrace{\begin{pmatrix} c_{j_2,1} \\ \vdots \\ c_{j_2,t} \end{pmatrix}}_{\overrightarrow{w_{j_2}}}$$

The Cheon et al. Attack [CHLRS14]

- Putting the y_{j_1,j_2}'s in a $t \times t$ matrix we get

$$\boldsymbol{Y} = [\boldsymbol{y}_{\boldsymbol{j_1},\boldsymbol{j_2}}] = \boldsymbol{U} \times \boldsymbol{V} \times \boldsymbol{W}$$

  - U has the $\overrightarrow{u_{j_1}}$'s as rows
  - V is as before
  - W has the $\overrightarrow{w_{j_2}}$'s as columns

Whp U,V,W are invertible

- Similarly $\boldsymbol{Y}' = [\boldsymbol{y}'_{\boldsymbol{j_1},\boldsymbol{j_2}}] = \boldsymbol{U} \times \boldsymbol{V}' \times \boldsymbol{W}$
- We know Y, Y' but not U, V, V', W
- Importantly, equalities hold over the integers

The Cheon et al. Attack [CHLRS14]

- Once we have Y, Y' we compute
$$\boldsymbol{Z = Y^{-1} \times Y' = (UVW)^{-1} \times (UV'W)}$$
$$\boldsymbol{= W^{-1} \times (V^{-1} \times V') \times W}$$

- Recall that $V^{-1} \times V' = \begin{pmatrix} b_1'/b_1 & & 0 \\ & \ddots & \\ 0 & & b_t'/b_t \end{pmatrix}$
  - Eigenvalues of $V^{-1} \times V'$ are $\boldsymbol{b_i'/b_i}$, $i = 1, \ldots, t$
  - Same for Z (since $V^{-1} \times V', Z$ are similar)

The Cheon et al. Attack [CHLRS14]

- After computing Z, compute its eigenvalues

$$\{b_i'/b_i : i = 1, \ldots, t\}$$

  - We get b_i, b_i' upto the factor $GCD(b_i, b_i')$

- 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 $\boldsymbol{v = \left[CRT(b_1, \dots, b_i, \dots, b_t)/z^{i'}\right]_N}$
    $\boldsymbol{v' = \left[CRT(b_1', \dots, b_i', \dots, b_t')/z^{i'}\right]_N}$
  - Express b_i'/b_i as a simple fraction $b_i'/b_i = d_i'/d_i$
    - d_i, d_i' are co-prime
  - $\boldsymbol{x_i = [d_i v' - d_i' v]_N}$ has 0 CRT component for p_i
  - Whp the other CRT components are not zero

➔Recover $p_i = GCD(N, x_i)$

Extending the Attack

- Easy to see that the same attack still works as long as $\boldsymbol{u}_{j_1} \cdot \boldsymbol{v} \cdot \boldsymbol{w}_{j_2}$ and $\boldsymbol{u}_{j_1} \cdot \boldsymbol{v}' \cdot \boldsymbol{w}_{j_2}$ are encoding of zeros for every j_1, j_2
  - Don't need the u_{j_1}'s themselves to encode zero
  - e.g.
    $\mathbf{u_j} \sim \left(\boldsymbol{a}_{\mathrm{j},1}\boldsymbol{g}_1, \boldsymbol{a}_{j,2}, \ \boldsymbol{a}_{j,3}\right)$,
    $\boldsymbol{v} \ \sim \ (\boldsymbol{b}_1, \ \ \boldsymbol{b}_2\boldsymbol{g}_2, \boldsymbol{b}_3)$ and $\boldsymbol{v}' \sim (\boldsymbol{b}'_1, \ \ \boldsymbol{b}'_2\boldsymbol{g}_2, \boldsymbol{b}'_3)$,
    $\boldsymbol{w}_j \sim \ (\boldsymbol{c}_{j,1}, \ \ \boldsymbol{c}_{j,2}, \ \ \boldsymbol{c}_{j,3}\boldsymbol{g}_3)$

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-randomize 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 $\boldsymbol{\alpha_1, \alpha_2, \ldots,}$ cannot tell if $\mathbf{expr}(\vec{\boldsymbol{\alpha}}) = \mathbf{0}$
  - $expr(*)$ has degree $\leq k - 3$ (say)
- Generally broken, use the attack with
  - $\mathbf{u_j} \sim \mathbf{expr}(\vec{\boldsymbol{\alpha}}) \cdot \boldsymbol{\alpha}_{\mathbf{j}}$
  - $\mathbf{v} \sim \boldsymbol{\alpha_1}, \mathbf{v}' \sim \boldsymbol{\alpha_2}$
  - $\mathbf{w_j} \sim \boldsymbol{\alpha}_{\mathbf{j}}$

Many Assumptions are Broken

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

Many Assumptions are Broken

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

A Suggested Fix

- Instead of $u_{j_1} v\, w_{j_2} \sim \vec{0}$, maybe we can use

$$\boldsymbol{\delta = u_{j_1} v\, w_{j_2} - \widehat{u}_{j_1} \widehat{v} \widehat{w}_{j_2} \sim \vec{0}}$$

  - For encodings u_j, v, w and $\widehat{u_j}, \hat{v}, \widehat{w_j}$
- This was suggested as a fix to the attacks
  - It is always possible to convert $\boldsymbol{u_{j_1} v\, w_{j_2} \sim \vec{0}}$ 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
  $$\boldsymbol{ztst}(\boldsymbol{\delta}) = \left(\sum_i \boldsymbol{a}_{\mathrm{j_1,i}} \boldsymbol{b}_{\mathrm{i}} \boldsymbol{c}_{\mathrm{j_2,i}} \cdot \boldsymbol{\sigma}_{\mathrm{i}} - \sum_i \hat{\boldsymbol{a}}_{\mathrm{j_1,i}} \widehat{\boldsymbol{b}}_{\mathrm{i}} \hat{\boldsymbol{c}}_{\mathrm{j_2,i}} \cdot \boldsymbol{\sigma}_{\mathrm{i}}\right) / \boldsymbol{g}$$
  - Similar to before, but now we have $1/g$ factor
    - $g = CRT(g_1, \ldots, g_t)$ in CLT
- Equality holds over the integers/rationals!
- So $\boldsymbol{Y} = \boldsymbol{U} \times \boldsymbol{V} \times \boldsymbol{W} \cdot {}^{\boldsymbol{1}}/_{\boldsymbol{g}}$, and the same for $\boldsymbol{Y'}$
- When setting $Z = Y^{-1} \times Y'$, the ${}^{1}/_{g}$ 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..