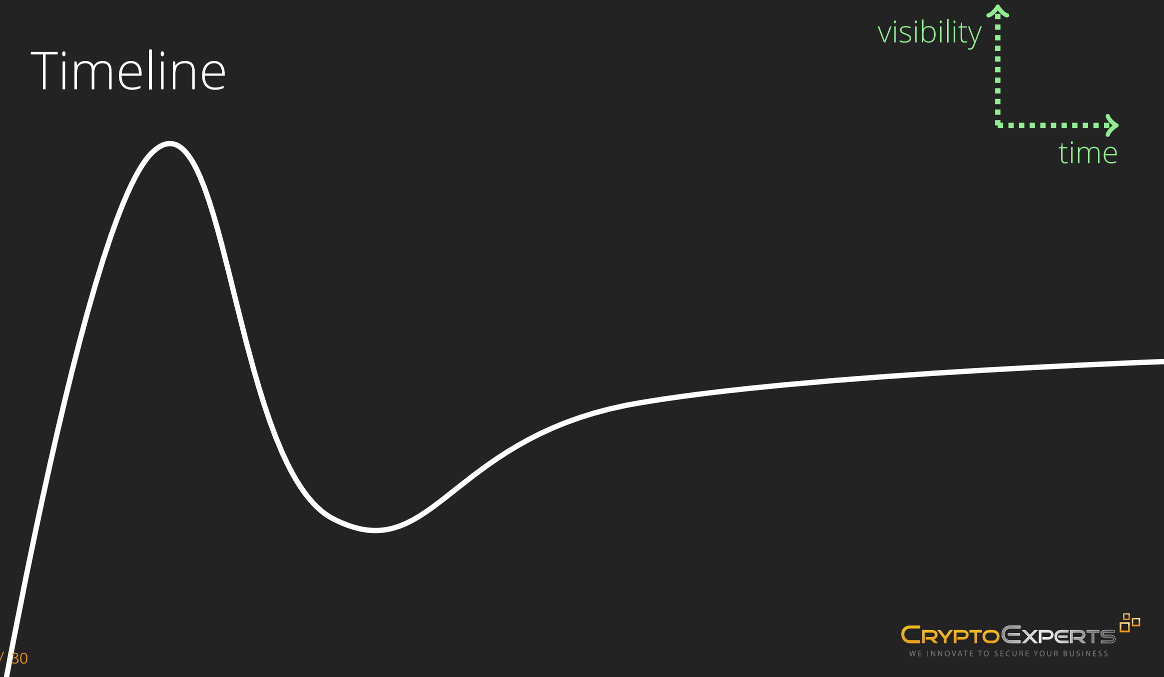
# Multilinear Maps over the Integers
# From Design to Security

Tancrède Lepoint    CryptoExperts

The Mathematics of Modern Cryptography Workshop, July 10th 2015

# Timeline: The Hype Cycle of Multilinear Maps

# Timeline

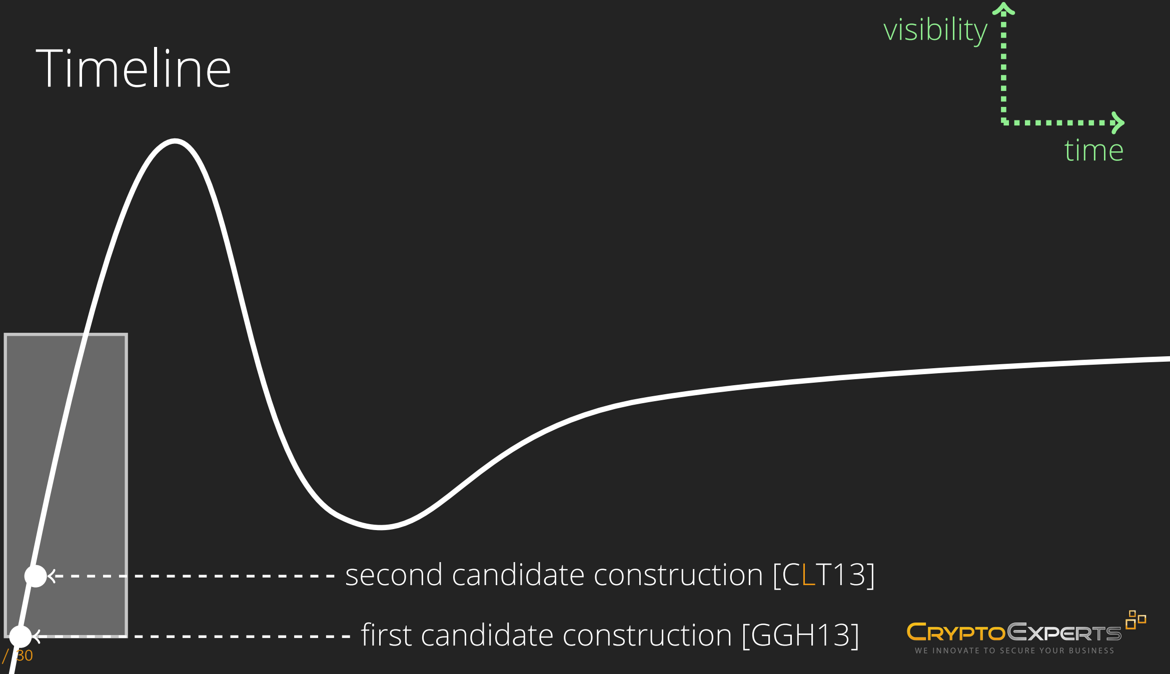





CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

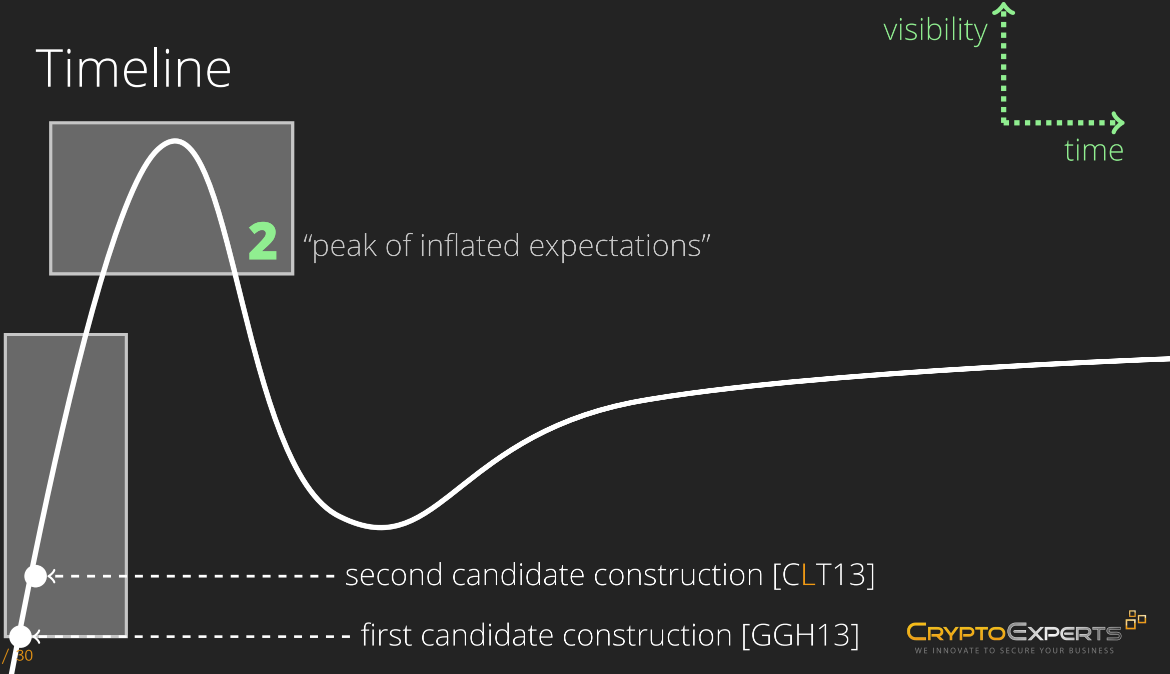
# Timeline

# Timeline



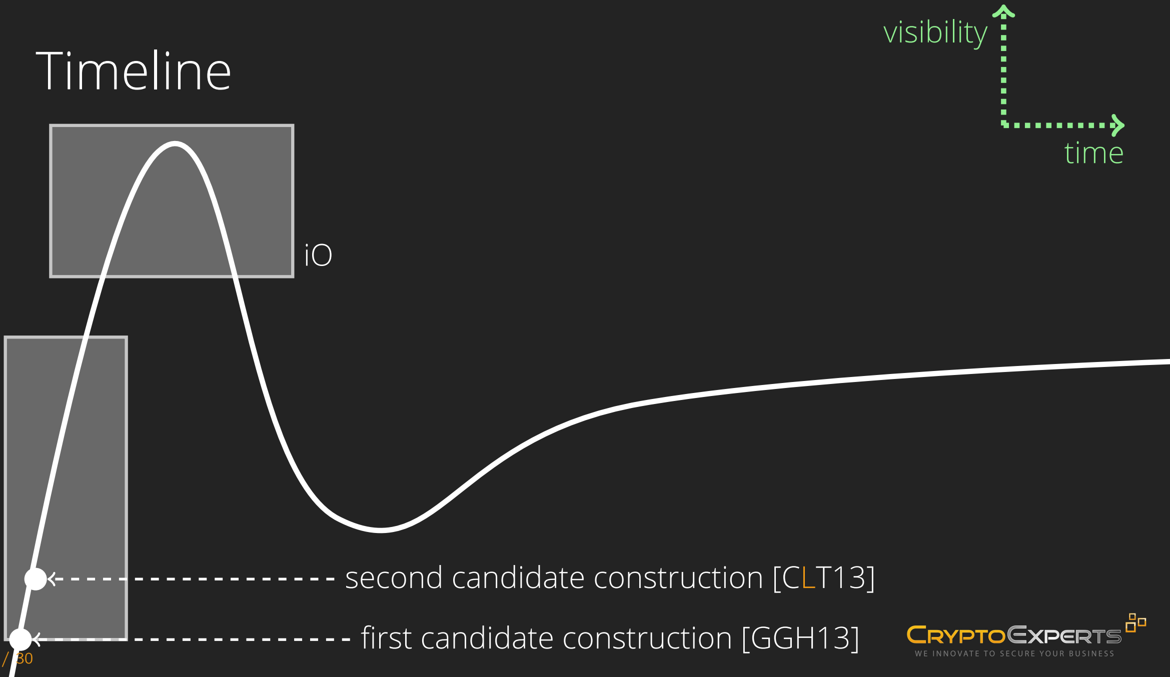

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Timeline

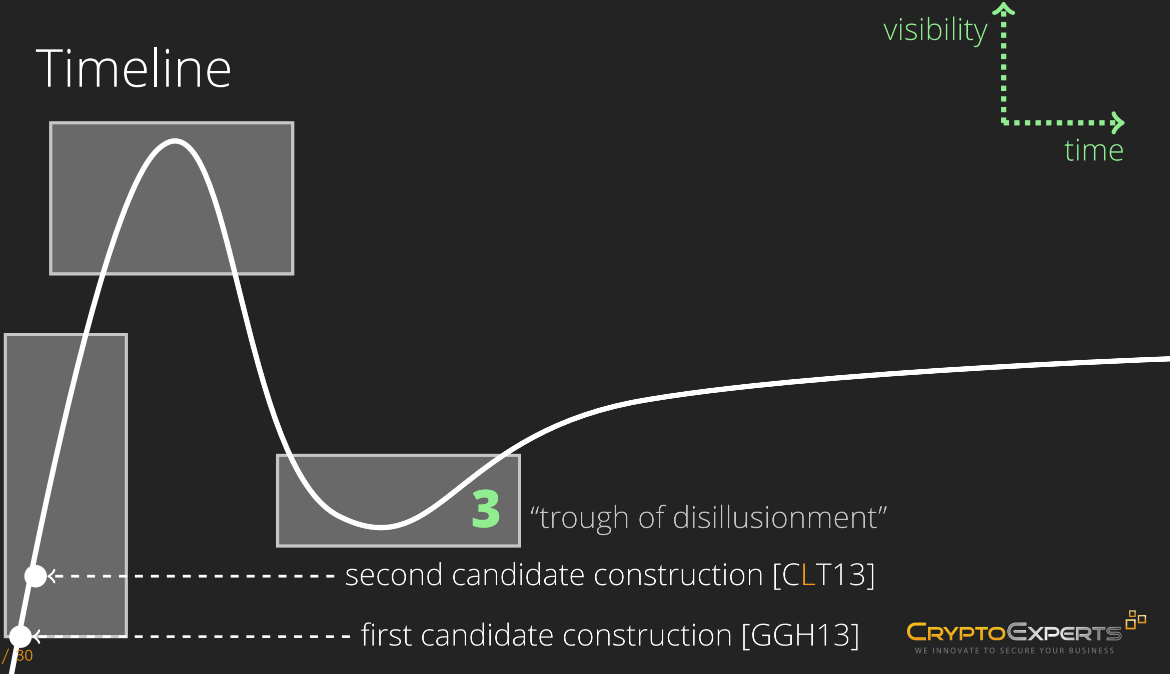



CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

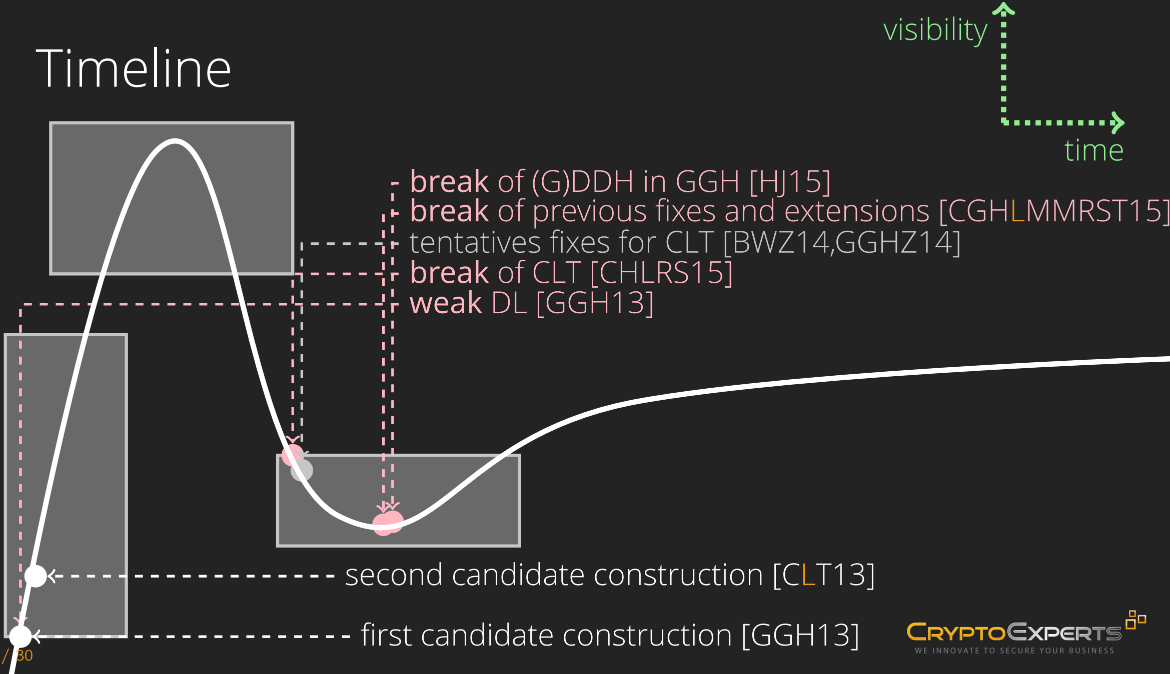
# Timeline


visibility
time
iO
second candidate construction [CLT13]
first candidate construction [GGH13]

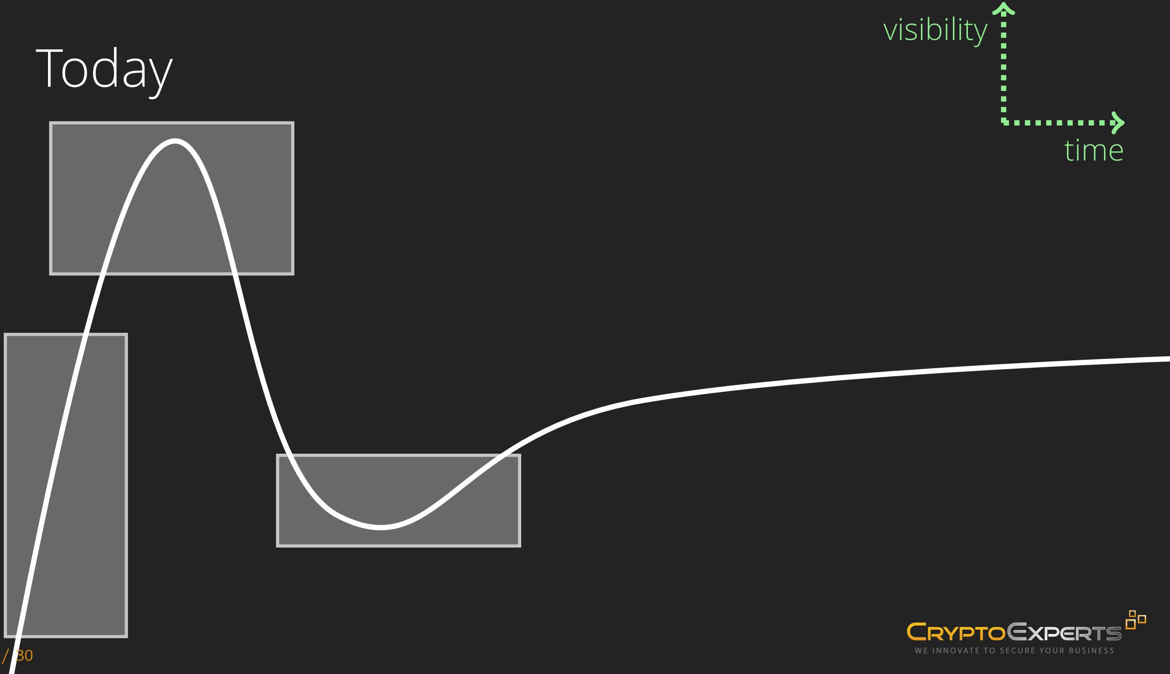
# Timeline


visibility
time
3
"trough of disillusionment"
second candidate construction [CLT13]
first candidate construction [GGH13]

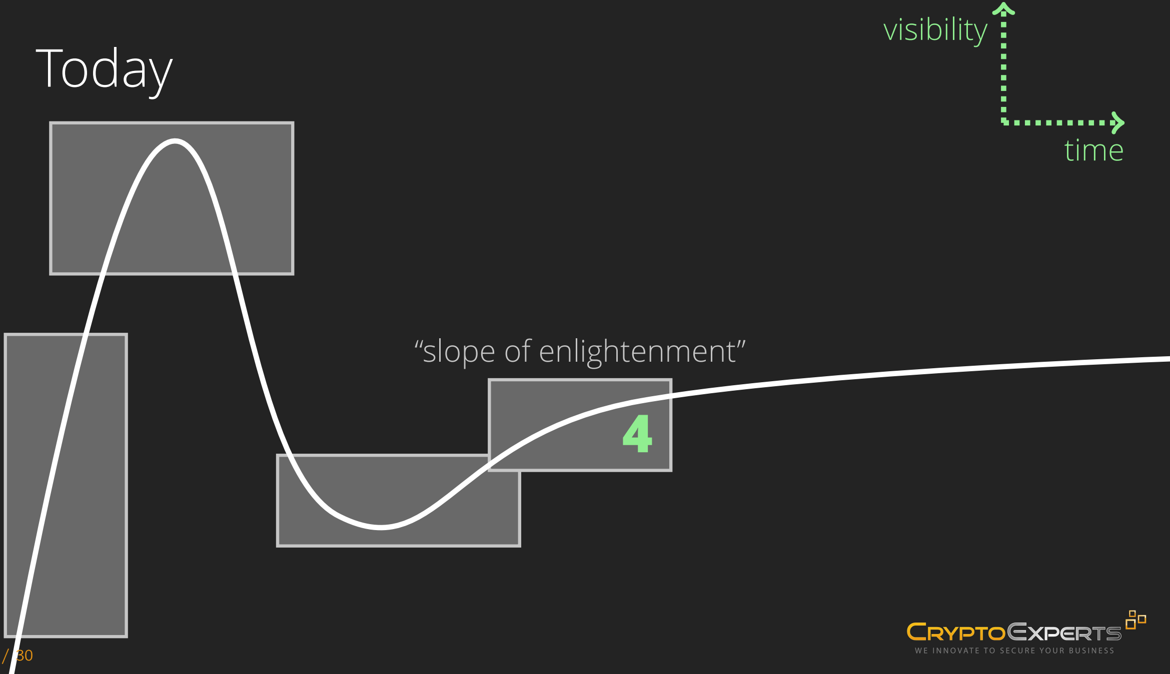
# Timeline


visibility
time
break of (G)DDH in GGH [HJ15]
break of previous fixes and extensions [CGHLMMRST15]
tentatives fixes for CLT [BWZ14,GGHZ14]
break of CLT [CHLRS15]
weak DL [GGH13]
second candidate construction [CLT13]
first candidate construction [GGH13]

# Today

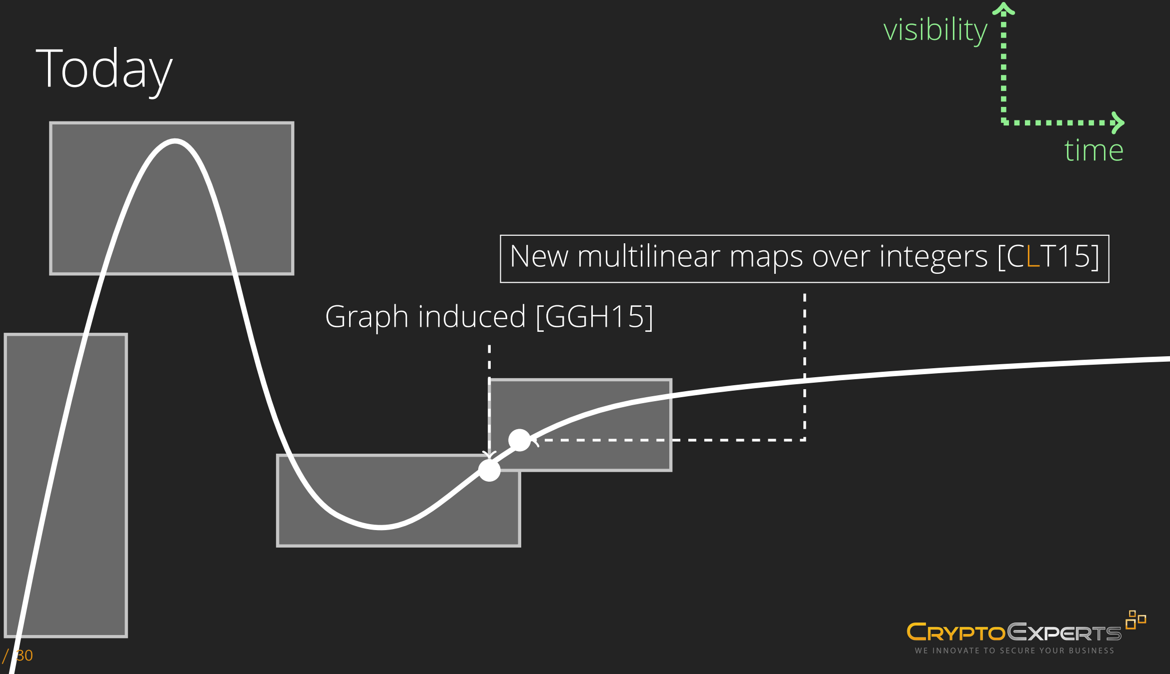


# Today




CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Today



visibility
time
New multilinear maps over integers [CLT15]
Graph induced [GGH15]

# The CLT Scheme

## Multilinear maps over the integers

[CoronL.Tibouchi'13'15]

# The CLT Scheme
## Multilinear maps over the integers

[CoronL.Tibouchi'13'15]

Second candidate construction

# The CLT Scheme

## Multilinear maps over the integers

[CoronL.Tibouchi'13'15]

Second candidate construction

Composite-order maps (different from [GGH13,GGH15])

# The CLT Scheme
## Multilinear maps over the integers

[CoronL.Tibouchi'13'15]

Second candidate construction

Composite-order maps (different from [GGH13,GGH15])

Follow [GGH13] recipe

- Level by multiplicative mask
- Zero-testing by multiplication and "shortness"

# The CLT Scheme
## Multilinear maps over the integers

[CoronL.Tibouchi'13'15]

Second candidate construction

Composite-order maps (different from [GGH13,GGH15])

Follow [GGH13] recipe

- Level by multiplicative mask
- Zero-testing by multiplication and "shortness"

Similar to FHE schemes based on Approximate-GCD

# The CLT Scheme

## Multilinear maps over the integers

[CoronL.Tibouchi'13'15]

Second candidate construction

Composite-order maps (different from [GGH13,GGH15])

Follow [GGH13] recipe

- Level by multiplicative mask
- Zero-testing by multiplication and “shortness”

Similar to FHE schemes based on Approximate-GCD

Useful for **many** applications...

# SWHE vs. MMAPs

## Computation over encrypted data

We want to compute homomorphically over encrypted data

. . . but we do not want the same information from the result than with HE

# SWHE vs. MMAPs
## Computation over encrypted data

We want to compute homomorphically over encrypted data

encode $a$ into $[a]$ $\longleftrightarrow$ encrypt $a$ into $[a] = \mathrm{Enc}(a)$

…but we do not want the same information from the result than with HE

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# SWHE vs. MMAPs
## Computation over encrypted data

We want to compute homomorphically over encrypted data

encode $a$ into $[a]$ $\longleftrightarrow$ encrypt $a$ into $[a] = \mathrm{Enc}(a)$
in both cases, computing low-degree polys of $[a_i]$'s is possible, up to a degree $k$

…but we do not want the same information from the result than with HE



CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# SWHE vs. MMAPs
## Computation over encrypted data

We want to compute homomorphically over encrypted data

encode $a$ into $[a]$ $\longleftrightarrow$ encrypt $a$ into $[a] = \mathrm{Enc}(a)$

in both cases, computing low-degree polys of $[a_i]$'s is possible, up to a degree $k$

…but we do not want the same information from the result than with HE

MMAPS can test if it is zero, at level $k$ (and hard to compute at degree $> k$)

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# SWHE vs. MMAPs
## Computation over encrypted data

We want to compute homomorphically over encrypted data

encode $a$ into $[a]$ $\longleftrightarrow$ encrypt $a$ into $[a] = \mathrm{Enc}(a)$
in both cases, computing low-degree polys of $[a_i]$'s is possible, up to a degree $k$

…but we do not want the same information from the result than with HE

| | |
|---|---|
| MMAPS | can test if it is zero, at level $k$ (and hard to compute at degree $> k$) |
| SHWE | no information on $a$ from the result, except with secret key |

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Starting from Homomorphic Encryption

## SWHE over the integers [DGHV10,CMNT11,CNT12,CCKLLMTY13,CLT14]

# Starting from Homomorphic Encryption

## SWHE over the integers [DGHV10,CMNT11,CNT12,CCKLLMTY13,CLT14]

Secret key prime $p$

# Starting from Homomorphic Encryption

## SWHE over the integers [DGHV10,CMNT11,CNT12,CCKLLMTY13,CLT14]

| | | |
|---|---|---|
| Secret key | prime $p$ | |
| Public key | $x_0 = q_0 \cdot p$ | for very large (hard to factor) $q_0$ |

# Starting from Homomorphic Encryption

## SWHE over the integers [DGHV10,CMNT11,CNT12,CCKLLMTY13,CLT14]

Secret key — prime $p$

Public key — $x_0 = q_0 \cdot p$ for very large (hard to factor) $q_0$

Ciphertext of $m$ — $c = q \cdot p + g \cdot r + m$
for $q \leftarrow [0, q_0)$ and $r \leftarrow \chi$ "small"

# Starting from Homomorphic Encryption

## SWHE over the integers [DGHV10,CMNT11,CNT12,CCKLLMTY13,CLT14]

Secret key prime $p$

Public key $x_0 = q_0 \cdot p$ for very large (hard to factor) $q_0$

Ciphertext of $m$ $c = \mathrm{CRT}_{q_0,p}(\; q' \;,\; g \cdot r + m \;)$
for $q' \leftarrow [0, q_0)$ and $r \leftarrow \chi$ "small"

# Starting from Homomorphic Encryption

## SWHE over the integers [DGHV10,CMNT11,CNT12,CCKLLMTY13,CLT14]

Secret key — primes $p_1, \ldots, p_n$

Public key — $x_0 = q_0 \cdot p_1 \cdots p_n$ for very large (hard to factor) $q_0$

Ciphertext of $\vec{m}$ — $c = \mathrm{CRT}_{q_0, p_1, \ldots, p_n}( \; q' \; , \; g_1 \cdot r_1 + m_1, \; \ldots, \; g_n \cdot r_n + m_n \; )$

for $q' \leftarrow [0, q_0)$ and $r_1, \ldots, r_n \leftarrow \chi$ "small"

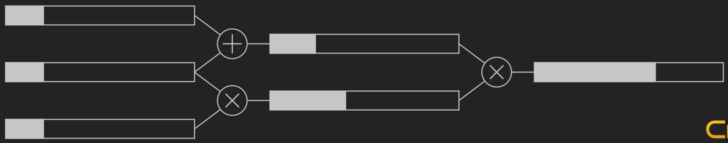
# Starting from Homomorphic Encryption

## SWHE over the integers [DGHV10,CMNT11,CNT12,CCKLLMTY13,CLT14]

Secret key primes $p_1, \ldots, p_n$

Public key $x_0 = q_0 \cdot p_1 \cdots p_n$ for very large (hard to factor) $q_0$

Ciphertext of $\vec{m}$ $c = \mathrm{CRT}_{q_0, p_1, \ldots, p_n}( \quad q' \quad , \quad g_1 \cdot r_1 + m_1, \quad \ldots, \quad g_n \cdot r_n + m_n \quad )$
for $q' \leftarrow [0, q_0)$ and $r_1, \ldots, r_n \leftarrow \chi$ "small"



CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Adding Sharp Levels

## Using multiplicative mask

[GGH13,CLT13]

# Adding Sharp Levels

## Using multiplicative mask

[GGH13,CLT13]

Let $z \leftarrow [0, x_0)$ be a random (invertible) multiplicative mask

# Adding Sharp Levels
## Using multiplicative mask

[GGH13,CLT13]

Let $z \leftarrow [0, x_0)$ be a random (invertible) multiplicative mask

Encoding of $\vec{m} \in \mathbb{Z}_{g_1} \times \cdots \times \mathbb{Z}_{g_n}$ at level $j$:

$$[\vec{m}]_j = c/z^j \bmod x_0 = \frac{\mathrm{CRT}_{q,p_1,\ldots,p_n}(q', r_1 \cdot g_1 + m_1, \ldots, r_n \cdot g_n + m_n)}{z^j} \bmod x_0$$

# Adding Sharp Levels

## Using multiplicative mask

[GGH13,CLT13]

Let $z \leftarrow [0, x_0)$ be a random (invertible) multiplicative mask

Encoding of $\vec{m} \in \mathbb{Z}_{g_1} \times \cdots \times \mathbb{Z}_{g_n}$ at level $j$:

$$[\vec{m}]_j = c/z^j \bmod x_0 = \frac{\mathrm{CRT}_{q,p_1,\ldots,p_n}(q', r_1 \cdot g_1 + m_1, \ldots, r_n \cdot g_n + m_n)}{z^j} \bmod x_0$$

Operations over $\mathbb{Z}_{x_0}$:

| | | |
|---|---|---|
| Addition | $[\vec{m}]_j + [\vec{m}']_j$ | $\simeq [\vec{m} + \vec{m}']_j$ |
| Multiplication | $[\vec{m}]_{j_1} \times [\vec{m}']_{j_2}$ | $\simeq [\vec{m} \cdot \vec{m}']_{j_1+j_2}$ |

# Main Ingredient: **Testing for Zero**

## Using the "shortness" of the noise [GGH13,CLT13]

# Main Ingredient: **Testing for Zero**

## Using the "shortness" of the noise [GGH13,CLT13]

How to test whether two degree-$k$ encodings are equal?

$$[\vec{m}]_k \simeq [\vec{\ell}]_k \text{ (i.e. } \vec{m} = \vec{\ell}) \iff [\vec{m} - \vec{\ell}]_k \simeq [\vec{0}]_k$$

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Main Ingredient: **Testing for Zero**

## Using the "shortness" of the noise

[GGH13,CLT13]

How to test whether two degree-$k$ encodings are equal?

$$[\vec{m}]_k \simeq [\vec{\ell}]_k \text{ (i.e. } \vec{m} = \vec{\ell}) \iff [\vec{m} - \vec{\ell}]_k \simeq [\vec{0}]_k$$

What is an encoding of $\vec{m} = \vec{0}$?

$$[\vec{0}]_k = \frac{\mathrm{CRT}_{q,p_1,\ldots,p_n}(q', r_1 \cdot g_1, \ldots, r_n \cdot g_n)}{z^k} \bmod x_0$$

# Main Ingredient: **Testing for Zero**

## Using the "shortness" of the noise

[GGH13,CLT13]

How to test whether two degree-$k$ encodings are equal?

$$[\vec{m}]_k \simeq [\vec{\ell}]_k \text{ (i.e. } \vec{m} = \vec{\ell}) \iff [\vec{m} - \vec{\ell}]_k \simeq [\vec{0}]_k$$

What is an encoding of $\vec{m} = \vec{0}$?

$$[\vec{0}]_k = \frac{\mathrm{CRT}_{q,p_1,\ldots,p_n}(q', r_1 \cdot g_1, \ldots, r_n \cdot g_n)}{z^k} \bmod x_0$$

Idea of [GGH13]: multiply by an element which will cancel $z^k$ and when the $r_i$'s are small ($r_i g_i \ll p_i$), yield something small compared to $x_0$.

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Simplifications for Zero-Testing

# Simplifications for Zero-Testing

$$[\vec{0}]_k = \sum_i g_i r_i \cdot ({p_i^*}^{-1}/z^k \bmod p_i) \cdot p_i^* + (\prod p_j) \cdot q'' \bmod x_0$$

where $p_i^* = \prod_{j \neq i} p_j$

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Simplifications for Zero-Testing

$$[\vec{0}]_k = \sum_i g_i r_i \cdot ({p_i^*}^{-1}/z^k \bmod p_i) \cdot p_i^* + (\prod p_j) \cdot q'' \bmod x_0$$

where $p_i^* = \prod_{j \neq i} p_j$

The random value $q''$ makes difficult to obtain something small... except if we are working modulo $\prod p_j$

# Simplifications for Zero-Testing

$$[\vec{0}]_k = \sum_i g_i r_i \cdot (p_i^{*-1}/z^k \bmod p_i) \cdot p_i^* + (\prod p_j) \cdot q'' \bmod x_0$$

where $p_i^* = \prod_{j \neq i} p_j$

The random value $q''$ makes difficult to obtain something small... except if we are working modulo $\prod p_j$

In the following $x_0 = \prod p_j$, and

$$[\vec{m}]_j = c/z^j \bmod x_0 = \frac{\mathrm{CRT}_{p_1,\ldots,p_n}(r_1 \cdot g_1 + m_1, \ldots, r_n \cdot g_n + m_n)}{z^j} \bmod x_0$$

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Zero-Testing Procedure

Multiply by the public element (where $h_i \ll p_i$)

$$p_{zt} = \sum_i h_i \cdot (g_i^{-1} z^k \bmod p_i) \cdot p_i^* \mod x_0$$

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Zero-Testing Procedure

Multiply by the public element (where $h_i \ll p_i$)

$$p_{zt} = \sum_i h_i \cdot (g_i^{-1} z^k \bmod p_i) \cdot p_i^* \mod x_0$$

$$[\vec{m}]_k = c/z^k \bmod x_0 = \frac{\mathrm{CRT}_{p_1,\ldots,p_n}(r_1 \cdot g_1 + m_1, \ldots, r_n \cdot g_n + m_n)}{z^k} \bmod x_0$$

therefore

$$[\vec{m}]_k \cdot p_{zt} = \sum_i (r_i + m_i g_i^{-1}) \cdot h_i \cdot p_i^* \bmod x_0$$

# Zero-Testing Procedure

Multiply by the public element (where $h_i \ll p_i$)

$$p_{zt} = \sum_i h_i \cdot (g_i^{-1} z^k \bmod p_i) \cdot p_i^* \mod x_0$$

$$[\vec{m}]_k = c/z^k \bmod x_0 = \frac{\mathrm{CRT}_{p_1,\ldots,p_n}(r_1 \cdot g_1 + m_1, \ldots, r_n \cdot g_n + m_n)}{z^k} \bmod x_0$$

therefore

$$[\vec{m}]_k \cdot p_{zt} = \sum_i (r_i + m_i g_i^{-1}) \cdot h_i \cdot p_i^* \bmod x_0$$

We have (we prove equivalence whp when many $p_{zt}$'s are given)

$$\vec{m} = \vec{0} \quad \Rightarrow \quad \left| [\vec{m}]_k \cdot p_{zt} \bmod x_0 \right| \ll x_0$$

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Hardness Assumptions

# Hardness Assumptions

**GDDH:** Given $(k+1)$ elements $[\vec{m}_i]_1$ and $[\vec{m}']_k$, determine whether $\vec{m}' \simeq \prod_{i=1}^{k+1} \vec{m}_i$.

# Hardness Assumptions

**GDDH:** Given $(k+1)$ elements $[\vec{m}_i]_1$ and $[\vec{m}']_k$, determine whether $\vec{m}' \simeq \prod_{i=1}^{k+1} \vec{m}_i$.

At the heart of the multipartite key echange protocol

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Hardness Assumptions

**GDDH:** Given $(k+1)$ elements $[\vec{m}_i]_1$ and $[\vec{m}']_k$, determine whether $\vec{m}' \simeq \prod_{i=1}^{k+1} \vec{m}_i$.

At the heart of the multipartite key echange protocol

Assumed to be **hard** (no reduction to Approx.-GCD)

# Hardness Assumptions

**GDDH:** Given $(k+1)$ elements $[\vec{m}_i]_1$ and $[\vec{m}']_k$, determine whether $\vec{m}' \simeq \prod_{i=1}^{k+1} \vec{m}_i$.

At the heart of the multipartite key echange protocol
Assumed to be **hard** (no reduction to Approx.-GCD)

Asymptotic parameters obtained from numerous attacks
- orthogonal lattice attack on encodings
- GCD attack on zero-testing
- hidden subset sum attack on zero-testing
- attacks on the inverse zero-testing matrix
- brute-force on the noises, ...

# But… Zeroizing Attack

## Eurocrypt 2015 best paper

[CHLRS15]

**Cryptanalysis of the Multilinear Map over the Integers**

Jung Hee Cheon[1], Kyoohyung Han[1], Changmin Lee[1], Hansol Ryu[1], Damien Stehlé[2]

[1] Seoul National University (SNU), Republic of Korea
[2] ENS de Lyon, Laboratoire LIP (U. Lyon, CNRS, ENSL, INRIA, UCBL), France.

**Abstract.** We describe a polynomial-time cryptanalysis of the (approximate) multilinear map of Coron, Lepoint and Tibouchi (CLT). The attack relies on an adaptation of the so-called *zeroizing* attack against the Garg, Gentry and Halevi (GGH) candidate multilinear map. Zeroiz-

# The Zeroizing Attack on CLT13

## Exploiting the (bi)linearity of the zero-testing procedure

# The Zeroizing Attack on CLT13

## Exploiting the (bi)linearity of the zero-testing procedure

$$[\vec{0}]_k \cdot p_{zt} = \sum_i r_i \cdot (h_i \cdot p_i^*) \in \mathbb{Z}$$

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# The Zeroizing Attack on CLT13

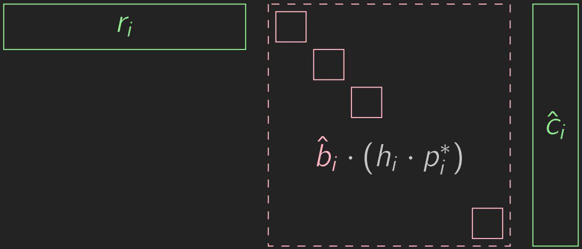
## Exploiting the (bi)linearity of the zero-testing procedure

$$[\vec{0}]_{k-2} \cdot [\vec{b}]_1 \cdot [\vec{c}]_1 \cdot p_{zt} \;= \sum_i r_i \cdot \hat{b}_i \cdot \hat{c}_i \cdot (h_i \cdot p_i^*) \in \mathbb{Z}$$

# The Zeroizing Attack on CLT13

## Exploiting the (bi)linearity of the zero-testing procedure

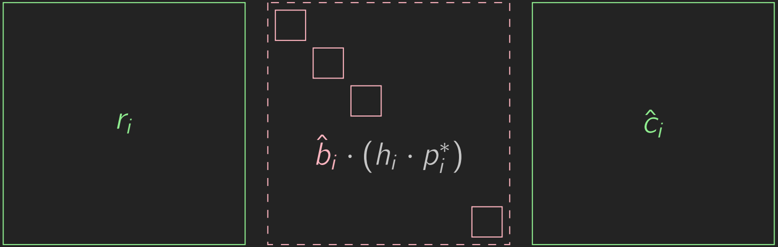
$$[\vec{0}]_{k-2} \cdot [\vec{b}]_1 \cdot [\vec{c}]_1 \cdot p_{zt} \;= \sum_i r_i \cdot \hat{b}_i \cdot \hat{c}_i \cdot (h_i \cdot p_i^*) \in \mathbb{Z}$$



# The Zeroizing Attack on CLT13

## Exploiting the (bi)linearity of the zero-testing procedure

$$[\vec{0}]_{k-2} \cdot [\vec{b}]_1 \cdot [\vec{c}]_1 \cdot p_{zt} \;=\; \sum_i r_i \cdot \hat{b}_i \cdot \hat{c}_i \cdot (h_i \cdot p_i^*) \in \mathbb{Z}$$



# The Zeroizing Attack on CLT13

## Inversion over $\mathbb{Q}$

Let's do it with many $[\vec{0}]_{k-2}$, $[\vec{c}]_1$ and two targets $[\vec{b}]_1$, $[\vec{b}']_1$

# The Zeroizing Attack on CLT13

## Inversion over $\mathbb{Q}$

Let's do it with many $[\vec{0}]_{k-2}$, $[\vec{c}]_1$ and two targets $[\vec{b}]_1$, $[\vec{b}']_1$

# The Zeroizing Attack on CLT13

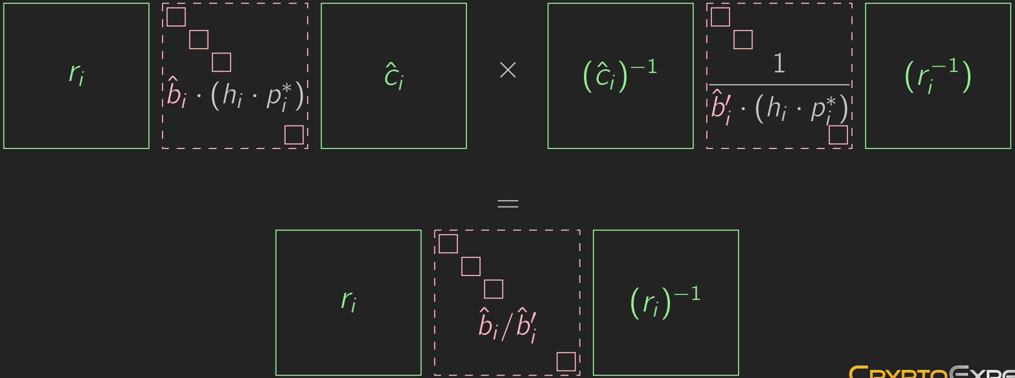
## Inversion over $\mathbb{Q}$

Let's do it with many $[\vec{0}]_{k-2}$, $[\vec{c}]_1$ and two targets $[\vec{b}]_1$, $[\vec{b}']_1$

# The Zeroizing Attack on CLT13

## Inversion over $\mathbb{Q}$

Let's do it with many $[\vec{0}]_{k-2}$, $[\vec{c}]_1$ and two targets $[\vec{b}]_1$, $[\vec{b}']_1$

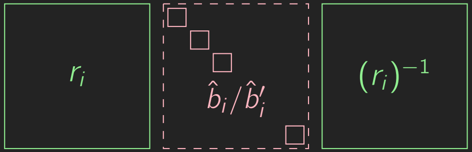


# The Zeroizing Attack on CLT13

## Computing eigenvalues

Consider the target encodings

$$[\vec{b}]_1 = \mathrm{CRT}_{p_i}(\hat{b}_i)/z, \quad [\vec{b}']_1 = \mathrm{CRT}_{p_i}(\hat{b}'_i)/z$$

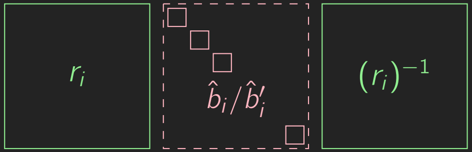


# The Zeroizing Attack on CLT13

## Computing eigenvalues

Consider the target encodings

$$[\vec{b}]_1 = \mathrm{CRT}_{p_i}(\hat{b}_i)/z, \quad [\vec{b}']_1 = \mathrm{CRT}_{p_i}(\hat{b}'_i)/z$$



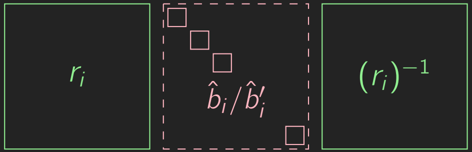

Compute the eigenvalues $\beta_i/\beta'_i = \hat{b}_i/\hat{b}'_i$

# The Zeroizing Attack on CLT13

## Computing eigenvalues

Consider the target encodings

$$[\vec{b}]_1 = \mathrm{CRT}_{p_i}(\hat{b}_i)/z, \quad [\vec{b}']_1 = \mathrm{CRT}_{p_i}(\hat{b}'_i)/z$$




Compute the eigenvalues $\beta_i/\beta'_i = \hat{b}_i/\hat{b}'_i$

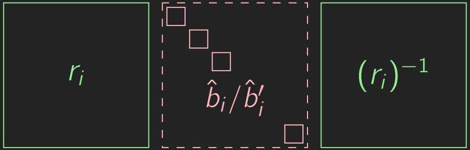
We have that

$$p_i \mid (\beta'_i \cdot [\vec{b}]_1 - \beta_i \cdot [\vec{b}']_1)$$

# The Zeroizing Attack on CLT13
## Computing eigenvalues

Consider the target encodings

$$[\vec{b}]_1 = \mathrm{CRT}_{p_i}(\hat{b}_i)/z, \quad [\vec{b}']_1 = \mathrm{CRT}_{p_i}(\hat{b}'_i)/z$$




Compute the eigenvalues $\beta_i/\beta'_i = \hat{b}_i/\hat{b}'_i$
We have that

$$p_i \mid (\beta'_i \cdot [\vec{b}]_1 - \beta_i \cdot [\vec{b}']_1)$$

Compute

$$p_i = \gcd(\beta'_i \cdot [\vec{b}]_1 - \beta_i \cdot [\vec{b}']_1, x_0)$$

# Generalizing the Zeroizing Attack on CLT13

## Zeroizing without low-level zeroes [CGHLMMRST15]

# Generalizing the Zeroizing Attack on CLT13

## Zeroizing without low-level zeroes [CGHLMMRST15]

Breaks early tentative fixes [BWZ14,GGHZ14] using zero-testing as a black-box

# Generalizing the Zeroizing Attack on CLT13

## Zeroizing without low-level zeroes [CGHLMMRST15]

Breaks early tentative fixes [BWZ14,GGHZ14] using zero-testing as a black-box

Don't need $[\vec{0}]_{k-2} \cdot [\vec{b}]_1 \cdot [\vec{c}]_1$ but $[\vec{a}]_{k-2} \cdot [\vec{b}]_1 \cdot [\vec{c}]_1 \simeq [\vec{0}]_k$

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

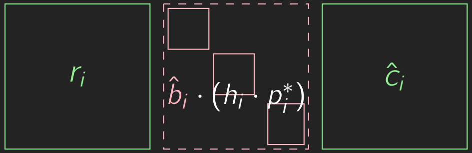
# Generalizing the Zeroizing Attack on CLT13

## Zeroizing without low-level zeroes [CGHLMMRST15]

Breaks early tentative fixes [BWZ14,GGHZ14] using zero-testing as a black-box

Don't need $[\vec{0}]_{k-2} \cdot [\vec{b}]_1 \cdot [\vec{c}]_1$ but $[\vec{a}]_{k-2} \cdot [\vec{b}]_1 \cdot [\vec{c}]_1 \simeq [\vec{0}]_k$

Can be diagonal per block. Instead of computing eigenvalues use **characteristic polynomial**.




CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Thwarting Cheon et al. Attack?

## Can we remove this linearity?

[CLT15]




CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Thwarting Cheon et al. Attack?

## Can we remove this linearity?

[CLT15]

The encodings look like DGHV ciphertexts

# Thwarting Cheon et al. Attack?

## Can we remove this linearity?

[CLT15]

The encodings look like DGHV ciphertexts

Even without the randomness $q$, their form should not be an issue

# Thwarting Cheon et al. Attack?

## Can we remove this linearity? [CLT15]

The encodings look like DGHV ciphertexts

Even without the randomness $q$, their form should not be an issue

In [CoronL.Tibouchi15], we revisit the **zero-testing procedure** itself



CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Thwarting Cheon et al. Attack?

## Can we remove this linearity?

[CLT15]

The encodings look like DGHV ciphertexts
Even without the randomness $q$, their form should not be an issue

In [CoronL.Tibouchi15], we revisit the **zero-testing procedure** itself

In a nutshell:

- the zero-testing is done modulo a new prime modulus $N$;
- $x_0$ is no longer public.

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Inherent randomness in current encodings

# Inherent randomness in current encodings

Current form of encodings

$$[\vec{m}]_k = \mathrm{CRT}_{p_i}(m_i + g_i r_i)/z^k \bmod x_0$$

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Inherent randomness in current encodings

Current form of encodings

$$[\vec{m}]_k = \mathrm{CRT}_{p_i}(m_i + g_i r_i)/z^k \bmod x_0$$

$$[\vec{m}]_k = \sum_i (m_i g_i^{-1} + r_i \bmod p_i) \cdot u_i + a \cdot x_0 \quad \text{over } \mathbb{Z}$$

with $u_i = (g_i {p_i^*}^{-1} z^{-k} \bmod p_i) p_i^*$.

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Inherent randomness in current encodings

Current form of encodings

$$[\vec{m}]_k = \mathrm{CRT}_{p_i}(m_i + g_i r_i)/z^k \bmod x_0$$

$$[\vec{m}]_k = \sum_i (m_i g_i^{-1} + r_i \bmod p_i) \cdot u_i + a \cdot x_0 \quad \text{over } \mathbb{Z}$$

with $u_i = (g_i p_i^{*-1} z^{-k} \bmod p_i) p_i^*$.

The element $a$ is highly non-linear in the $r_i$'s

The element $a$ is different from the random $q'$ we had before when adapting DGHV ($\vec{m} = \vec{0} \leftrightarrow a$ is small)

# New Zero-Test Parameter

Pick a random, large prime $N \gg x_0$. We want to generate a new zero-test value $\alpha_{zt}$ such that

$$\left|[\vec{m}]_k \cdot \alpha_{zt} \bmod N\right| \ll N \iff \vec{m} = 0$$

# New Zero-Test Parameter

Pick a random, large prime $N \gg x_0$. We want to generate a new zero-test value $\alpha_{zt}$ such that

$$|[\vec{m}]_k \cdot \alpha_{zt} \bmod N| \ll N \iff \vec{m} = 0$$

In particular, we have

$$\begin{aligned}&[\vec{m}]_k \cdot \alpha_{zt} \bmod N \\ &= \sum_i (m_i g_i^{-1} + r_i \bmod p_i) \cdot (u_i \cdot \alpha_{zt}) + a \cdot x_0 \cdot \alpha_{zt} \bmod N\end{aligned}$$

# New Zero-Test Parameter

Pick a random, large prime $N \gg x_0$. We want to generate a new zero-test value $\alpha_{zt}$ such that

$$|[\vec{m}]_k \cdot \alpha_{zt} \bmod N| \ll N \iff \vec{m} = 0$$

In particular, we have

$$\begin{aligned}&[\vec{m}]_k \cdot \alpha_{zt} \bmod N \\ &= \sum_i (m_i g_i^{-1} + r_i \bmod p_i) \cdot (u_i \cdot \alpha_{zt}) + a \cdot x_0 \cdot \alpha_{zt} \bmod N\end{aligned}$$

so we want $|\alpha_{zt} \cdot u_i \bmod N| \ll N$ and $|\alpha_{zt} \cdot x_0 \bmod N| \ll N$

# How To Generate $\alpha_{zt}$?

Given $N$, the generation of $\alpha_{zt} \in \mathbb{Z}_N$ such that for all $i$, $|u_i \alpha_{zt} \bmod N|$ and $|x_0 \alpha_{zt} \bmod N|$ are small is not obvious.

# How To Generate $\alpha_{zt}$?

Given $N$, the generation of $\alpha_{zt} \in \mathbb{Z}_N$ such that for all $i$, $|u_i\alpha_{zt} \bmod N|$ and $|x_0\alpha_{zt} \bmod N|$ are small is not obvious.

The problem amounts to finding a relatively short vector in a lattice

$$\begin{pmatrix} 1 & u_1 & \cdots & u_n & x_0 \\ & N & & & \\ & & \ddots & & \\ & & & N & \\ & & & & N \end{pmatrix}$$

# How To Generate $\alpha_{zt}$?

Given $N$, the generation of $\alpha_{zt} \in \mathbb{Z}_N$ such that for all $i$, $|u_i \alpha_{zt} \bmod N|$ and $|x_0 \alpha_{zt} \bmod N|$ are small is not obvious.

The problem amounts to finding a relatively short vector in a lattice

$$\begin{pmatrix} 1 & u_1 & \cdots & u_n & x_0 \\ & N & & & \\ & & \ddots & & \\ & & & N & \\ & & & & N \end{pmatrix}$$

Use LLL? (we can tolerate an exponential approx. factor over SVP), but typically $n \geq 10^5$

# How To Generate $\alpha_{zt}$?

## Using the structure of the $u_i$'s

# How To Generate $\alpha_{zt}$?

## Using the structure of the $u_i$'s

Remember that $N \gg x_0$ and $u_i = (g_i {p_i^*}^{-1} z^k \bmod p_i) p_i^*$

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# How To Generate $\alpha_{zt}$?

## Using the structure of the $u_i$'s

Remember that $N \gg x_0$ and $u_i = (g_i p_i^{*-1} z^k \bmod p_i) p_i^*$

First note that $p_j^{-1} u_i \bmod N$ is small for all $i \neq j$

# How To Generate $\alpha_{zt}$?

## Using the structure of the $u_i$'s

Remember that $N \gg x_0$ and $u_i = (g_i {p_i^*}^{-1} z^k \bmod p_i) p_i^*$

First note that $p_j^{-1} u_i \bmod N$ is small for all $i \neq j$
Only $p_j^{-1} u_j \bmod N$ is not a priori small

# How To Generate $\alpha_{zt}$?

## Using the structure of the $u_i$'s

Remember that $N \gg x_0$ and $u_i = (g_i p_i^{*-1} z^k \bmod p_i) p_i^*$

First note that $p_j^{-1} u_i \bmod N$ is small for all $i \neq j$
Only $p_j^{-1} u_j \bmod N$ is not a priori small

Let us find $\alpha_j$ such that $\alpha_j \cdot p_j^{-1} u_j \bmod N$ is small
As before it amounts to finding a short vector in

$$\begin{pmatrix} \lceil N/B \rceil & p_j^{-1} u_j \\ & N \end{pmatrix}$$

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# How To Generate $\alpha_{zt}$?

## Using the structure of the $u_i$'s

$$\binom{\lceil N/B \rceil \quad p_j^{-1} u_j}{N}$$

# How To Generate $\alpha_{zt}$?

## Using the structure of the $u_i$'s

$$\begin{pmatrix} \lceil N/B \rceil & p_j^{-1} u_j \\ & N \end{pmatrix}$$

We chose $B$ such that LLL finds a short vector

$$(\alpha_j \cdot \lceil N/B \rceil, \beta_j)$$

where $|\alpha_j| \leq \sqrt{p_j}$ and $|\beta_j = \alpha_j \cdot p_j^{-1} u_j \bmod N| \leq N/\sqrt{p_j}$.

# How To Generate $\alpha_{zt}$?

## Using the structure of the $u_i$'s

$$\begin{pmatrix} \lceil N/B \rceil & p_j^{-1} u_j \\ & N \end{pmatrix}$$

We chose $B$ such that LLL finds a short vector

$$(\alpha_j \cdot \lceil N/B \rceil, \beta_j)$$

where $|\alpha_j| \le \sqrt{p_j}$ and $|\beta_j = \alpha_j \cdot p_j^{-1} u_j \bmod N| \le N/\sqrt{p_j}$.

New zero-testing element:

$$\alpha_{zt} = \sum_j h_j \cdot \alpha_j \cdot p_j^{-1} \bmod N$$

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# How To Generate $\alpha_{zt}$? Using the structure of the $u_i$'s

New zero-testing element (sizes to keep in mind $N \approx x_0 \cdot p_j, \quad \alpha_j \approx \sqrt{p_j}$):

$$\alpha_{zt} = \sum_j h_j \cdot \alpha_j \cdot p_j^{-1} \bmod N$$

When applied on an encoding $[\vec{m}]_k$:

$$\begin{aligned} &[\vec{m}]_k \cdot \alpha_{zt} \bmod N \\ &= \sum_i (m_i g_i^{-1} + r_i \bmod p_i) \cdot (u_i \cdot \alpha_{zt}) + a \cdot x_0 \cdot \alpha_{zt} \bmod N \end{aligned}$$

# How To Generate $\alpha_{zt}$?

## Using the structure of the $u_i$'s

New zero-testing element (sizes to keep in mind $N \approx x_0 \cdot p_j, \quad \alpha_j \approx \sqrt{p_j}$):

$$\alpha_{zt} = \sum_j h_j \cdot \alpha_j \cdot p_j^{-1} \bmod N$$

When applied on an encoding $[\vec{m}]_k$:

$$\begin{aligned} &[\vec{m}]_k \cdot \alpha_{zt} \bmod N \\ &= \sum_i (m_i g_i^{-1} + r_i \bmod p_i) \cdot (h_i \beta_i + \sum_{j \neq i} h_j \alpha_j \cdot u_i / p_j) \\ &+ a \cdot x_0 \cdot \alpha_{zt} \bmod N \end{aligned}$$

# An Important Caveat

## Cannot work directly modulo $x_0$

# An Important Caveat

## Cannot work directly modulo $x_0$

$x_0$ cannot be made public, contrary to [CLT13]

# An Important Caveat

## Cannot work directly modulo $x_0$

$x_0$ cannot be made public, contrary to [CLT13]

However, define $v_0 = x_0 \cdot \alpha_{zt} \bmod N$, and

$$
\begin{aligned}
&([\vec{0}]_k \cdot \alpha_{zt} \bmod N) \bmod v_0 \\
&= \Big(\sum_i r_i \cdot \big(h_i\beta_i + \sum_{j \neq i} h_j\alpha_j \cdot u_i/p_j\big) + a \cdot v_0 \in \mathbb{Z}\Big) \bmod v_0 \\
&= \sum_i r_i \cdot \big(h_i\beta_i + \sum_{j \neq i} h_j\alpha_j \cdot u_i/p_j\big) \bmod v_0
\end{aligned}
$$

# An Important Caveat

## Cannot work directly modulo $x_0$

$x_0$ cannot be made public, contrary to [CLT13]

However, define $v_0 = x_0 \cdot \alpha_{zt} \bmod N$, and

$$
\begin{aligned}
&([\vec{0}]_k \cdot \alpha_{zt} \bmod N) \bmod v_0 \\
&= \Big(\sum_i r_i \cdot \big(h_i\beta_i + \sum_{j\neq i} h_j\alpha_j \cdot u_i/p_j\big) + a \cdot v_0 \in \mathbb{Z}\Big) \bmod v_0 \\
&= \sum_i r_i \cdot \big(h_i\beta_i + \sum_{j\neq i} h_j\alpha_j \cdot u_i/p_j\big) \bmod v_0
\end{aligned}
$$

We can apply Cheon et al. attack modulo $v_0$

# An Important Caveat

## A Ladder of encodings

# An Important Caveat

## A Ladder of encodings

Making $x_0$ secret is somewhat inconvenient:

- when we add or multiply encodings, we cannot reduce them modulo $x_0$ anymore to keep them of the same size

# An Important Caveat
## A Ladder of encodings

Making $x_0$ secret is somewhat inconvenient:
when we add or multiply encodings, we cannot reduce them modulo $x_0$ anymore to keep them of the same size

Solution (taken from [DGHV10]): publish a ladder of encodings of 0 of increasing size

- encodings
$$X_i^{(j)} = (\mathrm{CRT}_{p_i}(r_i g_i)/z^j \bmod x_0) + q_i \cdot x_0$$
with $q_i \leftarrow [0, 2^i)$ for $i = 1, \ldots, \log(x_0)$
- do the operation over $\mathbb{Z}$, and remove $X_i^{(j)}$ for decreasing $i$'s

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Concrete Attempt

# Concrete Attempt

Consider $u = [\vec{0}]_{k-2} \cdot [\vec{b}]_1 \cdot [\vec{c}]_1$

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Concrete Attempt

Consider $u = [\vec{0}]_{k-2} \cdot [\vec{b}]_1 \cdot [\vec{c}]_1$

Apply the ladder to reduce its size to the size of $x_0$:

$$u' = u + \sum s_i X_i^{(k)}$$

# Concrete Attempt

Consider $u = [\vec{0}]_{k-2} \cdot [\vec{b}]_1 \cdot [\vec{c}]_1$
Apply the ladder to reduce its size to the size of $x_0$:

$$u' = u + \sum s_i X_i^{(k)}$$

Write $u'$ over $\mathbb{Z}$:

$$u' = \sum_i (r_i \cdot \hat{b}_i \cdot \hat{c}_i + s_i \cdot r_{X,i,k}) \cdot u_i - a \cdot x_0$$

# Concrete Attempt

Consider $u = [\vec{0}]_{k-2} \cdot [\vec{b}]_1 \cdot [\vec{c}]_1$
Apply the ladder to reduce its size to the size of $x_0$:

$$u' = u + \sum s_i X_i^{(k)}$$

Write $u'$ over $\mathbb{Z}$:

$$u' = \sum_i (r_i \cdot \hat{b}_i \cdot \hat{c}_i + s_i \cdot r_{X,i,k}) \cdot u_i - a \cdot x_0$$

All $s_i$'s and $a$ come up in the way of Cheon et al. attack



CRYPTOEXPERTS
WE INNOVATE TO SECURE YOUR BUSINESS

# Proof-of-concept Implementation

https://github.com/tlepoint/new-multilinear-maps

| Instantiation | $\lambda$ | $\kappa$ | $n$ | $\eta$ | $\Delta$ | $\rho$ | $\gamma = n \cdot \eta$ | pp size |
|---|---|---|---|---|---|---|---|---|
| Small | 52 | 6 | 540 | 1679 | 23 | 52 | $0.9 \cdot 10^6$ | 27 MB |
| Medium | 62 | 6 | 2085 | 1989 | 45 | 62 | $4.14 \cdot 10^6$ | 175 MB |
| Large | 72 | 6 | 8250 | 2306 | 90 | 72 | $19.0 \cdot 10^6$ | 1.2 GB |
| Extra | 80 | 6 | 25305 | 2619 | 159 | 85 | $66.3 \cdot 10^6$ | 6.1 GB |

| Setup | Publish | KeyGen |
|---|---|---|
| 5.9 s | 0.10 s | 0.17 s |
| 36 s | 0.33 s | 1.06 s |
| 583 s | 2.05 s | 6.17 s |
| 4528 s | 7.8 s | 23.9 s |

# Conclusion

# Conclusion

The CLT scheme has many interesting features:

- composite order maps,
- assumed hardness of GDDH but also of DLIN & SubM

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Conclusion

The CLT scheme has many interesting features:

- composite order maps,
- assumed hardness of GDDH but also of DLIN & SubM

Concrete targets to attack in practice if desired

Same efficiency as original CLT13

# Conclusion

The CLT scheme has many interesting features:

composite order maps,

assumed hardness of GDDH but also of DLIN & SubM

Concrete targets to attack in practice if desired

Same efficiency as original CLT13

Open problems for CLT15:

- Analyze the reparation
- Improve the efficiency
- Adapt the technique to [GGH13]?

CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Thank You

## Questions & Discussion




CryptoExperts
WE INNOVATE TO SECURE YOUR BUSINESS

# Discussion

1. Design
   - public encoding space / inversion
2. Attacks
3. Assumptions
   - what sort of assumptions can be made?
   - base multilinear maps on well-known problems
4. Applications
   - something that look different from obfuscation
   - what can you do with a small number of levels?
   - relation between 2-multilinear maps / pairings in applications