CUCKOO COMMITMENTS: REGISTRATION-BASED ENCRYPTION & KEY-VALUE MAP COMMITMENTS FOR LARGE SPACES

WORK BY DARIO FIORE $^1$, DIMITRIS KOLONELOS $^{1,2}$ & PAOLA DE PERTHUIS $^{3,4}$
FULL VERSION: HTTPS://EPRINT.IACR.ORG/2023/1389
PUBLISHED AT ASIACRYPT 2023
PRESENTATION BY PAOLA DE PERTHUIS
REGISTRATION-BASED ENCRYPTION

MOTIVATIONS
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]

Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.

updatable RBE
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE), but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

Stemming from [TCC:Garg-Hajiabadi-Mahmoody-Rahimi-18]
Like Identity-Based Encryption (IBE),
but without the key curator holding secrets.
REGISTRATION-BASED ENCRYPTION (RBE)

State-of-the-art:
- first constructions were very inefficient;
- efficient black-box constructions in [Glaeser-Kolonelos-Malavolta-Rahimi-22]
  but identity-space of polynomial size
- and [EC:Döttling-Kolonelos-Lai-Lin-Malavolta-Rahimi-23] with lattices, but
  ciphertexts in GB
REGISTRATION-BASED ENCRYPTION (RBE)

State-of-the-art:
- first constructions were very inefficient;
- efficient black-box constructions in [Glaeser-Kolonelos-Malavolta-Rahimi-22] but identity-space of polynomial size
- and [EC:Döttling-Kolonelos-Lai-Lin-Malavolta-Rahimi-23] with lattices, but ciphertexts in GB

| Setting | TD | Compactness | |ct| | #updates | |pp| + |crs|
|---|---|---|---|---|---|---|
| [HLWW23] Pairings (C) \{0, 1\}* | Adaptive | \(O(\lambda \log n)\) | \(\log n\) | \(O(\lambda r^{2/3} \log n)\) |
| [GKMR22] Pairings (P) \{1, n\} | Adaptive | \(4 \log n\) | \(\log n\) | \(O(\sqrt{n \log n})\) |
| Ours P1 Pairings (P) \{0, 1\}* | Adaptive | \(6 \lambda \log n\) | \(\log n\) | \(O(\sqrt{\lambda n \log n})\) |
| Ours P2 Pairings (P) \{0, 1\}* | Selective | \(12 \log n\) | \(\log n\) | \(O(\sqrt{n \log n})\) |
| [DKL*23] Lattices \{0, 1\} | Adaptive | \((2\lambda + 1) \log n\) | \(\log n\) | \(O(\log n)\) |
| Ours L Lattices \{0, 1\}* | Selective | \(4 \log^2 n\) | \(\log n\) | \(O(\log n)\) |

Table 1: Comparison of the schemes resulting from different instantiations of our compiler. \(n\) is the maximum number of users to be registered. Pairings (P) indicates prime order groups and Pairings (C) composite order groups respectively. |ct| in the pairing construction is measured in group elements and in the Lattice constructions LWE ciphertexts.
A TECHNIQUE BASED ON CUCKOO HASHING

A NEW SETTING FOR CUCKOO HASHING
CUCKOO HASHING

A powerful technique
CUCKOO HASHING

A powerful technique
CUCKOO HASHING
A powerful technique
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest...
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest
Cuckoo Hashing

A powerful technique; an example with two hash functions, and one animal per nest.
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest.
Cuckoo Hashing

A powerful technique; an example with two hash functions, and one animal per nest
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest.
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest and with a stash
Cuckoo Hashing

A powerful technique; an example with two hash functions, and one animal per nest and with a stash.
Cuckoo Hashing

A powerful technique; an example with two hash functions, and one animal per nest and with a stash.
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest and with a stash.
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest and with a stash.
Cuckoo Hashing

A powerful technique; an example with two hash functions, and one animal per nest and with a stash.
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest and with a stash.
CUCKOO HASHING

A powerful technique; an example with two hash functions, and one animal per nest and with a stash...
CUCKOO HASHING

Performance
CUCKOO HASHING

Performance, for negligible failure (in $\lambda$):
- $h = 2$ hash functions, $N = 2^{hn}$ nests with capacity one, to store $n$ animals:
  average constant insertion time, worst-case $\log(n)$ stash.
CUCKOO HASHING

Performance, for negligible failure (in $\lambda$):
- $h = 2$ hash functions, $N = 2^{hn}$ nests with capacity one, to store $n$ animals:
  average constant insertion time, worst-case $\log(n)$ stash.

Security against adversaries wanting to fill the stash?
(choosing animals maliciously)
Cuckoo Hashing

Performance, for negligible failure (in $\lambda$):
- $h = 2$ hash functions, $N = 2hn$ nests with capacity one, to store $n$ animals:
  average constant insertion time, worst-case $\log(n)$ stash.

Security against adversaries wanting to fill the stash?
  (choosing animals maliciously)
- $h = 2$, $N = 2kn$ nests: average constant insertion, worst-case $n$ stash
Performance, for negligible failure (in $\lambda$):
- $h = 2$ hash functions, $N = 2hn$ nests with capacity one, to store $n$ animals:
  average constant insertion time, worst-case $\log(n)$ stash.
Security against adversaries wanting to fill the stash?
  (choosing animals maliciously)
- $h = 2$, $N = 2kn$ nests: average constant insertion, worst-case $n$ stash
- $h = \lambda$, $N = 2\lambda n$ nests: average $\lambda$ time insertion, worst-case empty stash

CUCKOO HASHING
**Cuckoo Hashing**

Performance, for negligible failure (in $\lambda$):
- $h = 2$ hash functions, $N = 2hn$ nests with capacity one, to store $n$ animals:
  average constant insertion time, worst-case $\log(n)$ stash.

Security against adversaries wanting to fill the stash?
(choosing animals maliciously)
- $h = 2, N = 2kn$ nests: average constant insertion, worst-case $n$ stash
- $h = \lambda, N = 2\lambda n$ nests: average $\lambda$ time insertion, worst-case empty stash

reference for parameters in cryptography: [C:Yeo23]
OUR CONSTRUCTION USING CUCKOO HASHING WITH VECTOR COMMITMENTS (VC), AND WITNESS ENCRYPTION FOR VC (VCWE)
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]

\[ \text{crs} = g_1, g_2, g_3, g_5, g_6 \]

\[ g_1 = 1 \]

\[ g_2, g_3 \]

\[ g_2 \]

\[ g_3, g_5 \]

\[ g_2 \]
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC-LY10]
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE,
USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC-LY10]
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]

I need an opening of for the second vector component.
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]

\[ g_1, g_2, g_3, g_5, g_6 \]

I need an opening of \( g_2, g_3 \) for the second vector component and also to know \( = 1 \)

\( = 2 \)
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]

Updates

\[ g_1, g_2, g_3, g_5, g_6 \]

\[ g_2, g_3 \]

\[ = 1 \]

\[ g_5, g_6 \]

\[ = 2 \]

\[ \rightarrow \quad \text{g3} \]
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]

Updates

\[ g_1, g_2, g_3, g_5, g_6 \]

\[ g_2, g_3 \]

\[ g_5, g_6 \]

\[ = 1 \]

\[ = 2 \]
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]

Updates

\[ g_1, g_2, g_3, g_5, g_6 \]

\[ g_2, g_3 \]

\[ g_5, g_6 \]

\[ = 1 \]

\[ = 2 \]
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]

Updates

\[ g_1, g_2, g_3, g_5, g_6 \]

\[ g_2 \cdot g_3 \]

\[ g_5 \cdot g_6 \]

\[ = 1 \]

\[ = 2 \]
THE GLAESER-KOLONELLOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]

Updates

\[ g_1, g_2, g_3, g_5, g_6 \]

= 1

= 2

Key

Key
THE GLAESER-KOLONELOS-MALAVOLTA-RAHIMI (GKMR) RBE, USING LIBERT-YUNG VECTOR COMMITMENTS [TCC:LY10]

Updates

\[ g_1, g_2, g_3, g_5, g_6 \]

= 1

= 2
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING

$g_1, g_2, g_3, g_5, g_6$
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING

$g_1, g_2, g_3, g_5, g_6$

Cuckoo hashing

$\rightarrow 2$

$\rightarrow 1$

etc. for ids 2 and 1

etc. for ids 1 and 3
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING

$g_1, g_2, g_3, g_5, g_6$

For id 2

For id 1

2

1

2

1

3

1

0, etc.

0, etc.
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING

$g_1, g_2, g_3, g_5, g_6$
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING

Problem

$g_1, g_2, g_3, g_5, g_6$

2

$\rightarrow$

1

3

$\rightarrow$

1

$\rightarrow$

2

$\rightarrow$

1
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING

Problem:
what if encryptors use the wrong hash function?
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING

Encryption needs to be not only with respect to the position, but also the identity.
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING

commitment of $g_1, g_2, g_3, g_5, g_6$
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING

commitment of
commitment of

$g_1, g_2, g_3, g_5, g_6$
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING

commitment of

opening of

in \( g_2 \)

(in position 2)
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING

Without both openings for position 1, without committed to in in position 1, without made for this cuckoo hashing, nothing could be inferred.
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING
updatable as before
OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING

updatable as before
if the cuckoo hashing changes,
the commitments and opening change.
**OUR SCHEME, COMBINING GKMR WITH CUCKOO HASHING**

| Setting       | ID       | Compactness | $|ct|$            | #updates | $|pp| + |crs|$ |
|---------------|----------|-------------|-----------------|----------|------------------|
| [HLWW23]      | Pairings (C) $\{0,1\}^*$ | Adaptive     | $O(\lambda \log n)$ | $\log n$ | $O(\sqrt{n \log n})$ |
| [GKMR22]      | Pairings (P) $[1, n]$ | Adaptive     | $4\log n$       | $\log n$ | $O(\sqrt{n \log n})$ |
| Ours P1       | Pairings (P) $\{0,1\}^*$ | Adaptive     | $6\lambda \log n$ | $\log n$ | $O(\sqrt{n \log n})$ |
| Ours P2       | Pairings (P) $\{0,1\}^*$ | Selective    | $12\log n$      | $\log n$ | $O(\sqrt{n \log n})$ |
| [DKL+23]      | Lattices $\{0,1\}^*$ | Adaptive     | $(2\lambda + 1) \log n$ | $\log n$ | $O(\log n)$ |
| Ours L        | Lattices $\{0,1\}^*$ | Selective    | $4\log^2 n$     | $\log n$ | $O(\log n)$ |

Table 1: Comparison of the schemes resulting from different instantiations of our compiler. $n$ is the maximum number of users to be registered. Parings (P) indicates prime order groups and Pairings (C) composite order groups respectively. $|ct|$ in the pairing construction is measured in group elements and in the Lattice constructions LWE ciphertexts.
OTHER CONTRIBUTION

KEY-VALUE MAP COMMITMENTS
for large keys, with updates, using pairings
OTHER CONTRIBUTION

KEY-VALUE MAP COMMITMENTS
for large keys, with updates, using pairings

equivalence of vector commitments
and universal accumulators
THANK YOU

WORK BY DARIO FIORE ¹, DIMITRIS KOLONELOS ¹,² & PAOLA DE PERTHUIS ³,⁴
FULL VERSION: HTTPS://EPRINT.IACR.ORG/2023/1389
PUBLISHED AT ASIACRYPT 2023
PRESENTATION BY PAOLA DE PERTHUIS

1: IMdea Institute for Advanced Studies
2: Universidad Politécnica de Madrid
3: cosmian
4: École Normale Supérieure