UNIVERSITY OF LEEDS

# Practical Homomorphic Encryption Over the Integers for Secure Computation in the Cloud

**James Dyer[1,2] Martin Dyer[2] Jie Xu[2]**

[1] University of Manchester

[2] University of Leeds

IMA Cryptography & Coding 2017, 12-14 December 2017

## Motivation

- Secure single-party computation in the cloud
  - Similar approach to CryptDB [3]
  - Several encryption schemes to support operations on ciphertexts
- Encryption that supports arithmetic computation

## Our Results

- Family of schemes in 2 variants inspired by [2]:
  - HE$k$: high entropy inputs
  - HE$k$N: IND-CPA secure, any inputs
- Experimental evaluation of HE1, HE1N, HE2, HE2N

# Preliminaries

- Each scheme is the symmetric key HE scheme (KGen,Pgen,Enc,Dec,Add,Mult)
- Message space $\mathcal{M}$ is $[0, M]$
  - $n$ inputs: $m_1, m_2, \ldots, m_n$
- Ciphertext space $\mathcal{C}$ is $[0, N]$
- Security parameter space $\mathcal{S}$ and secret key space $\mathcal{K}$ are the positive integers.
- Compute degree $d$ multivariate polynomial

UNIVERSITY OF LEEDS

## Key Generation (KGen)

**Input** : $\lambda \in \mathcal{S}, \lambda > d(\lg(n+1) + \lg M)$
**Output** : $p \in \mathcal{K}$: secret key

1 $p \xleftarrow{\mathsf{p}} [2^{\lambda-1}, 2^{\lambda}]$
2 **return** $p$

UNIVERSITY OF LEEDS

## Parameter Generation (Pgen)

**Input** : $\lambda \in \mathcal{S}$
**Input** : $\rho \in \mathbb{Z}$: entropy of inputs
**Input** : $p$: secret key
**Output** : modulus $\in \mathbb{Z}$: public modulus

1 $\eta \leftarrow \lambda^2/\rho - \lambda$
2 $q \xleftarrow{\text{p}} [2^{\eta-1}, 2^{\eta}]$
3 modulus $\leftarrow pq$
4 **return** modulus

# Encryption (Enc)

**Input** : $m \in \mathcal{M}$
**Input** : $p$: secret key
**Input** : modulus: public modulus
**Output** : $c \in \mathcal{C}$

1 $q \leftarrow \text{modulus}/p$
2 $r \xleftarrow{\$} [1, q)$
3 $c \leftarrow m + rp \pmod{\text{modulus}}$
4 **return** $c$

UNIVERSITY OF LEEDS

## Decryption (Dec)

**Input** : $c \in \mathcal{C}$
**Input** : $p$: secret key
**Output** : $m \in \mathcal{M}$

1 $m \leftarrow c \pmod{p}$
2 **return** $m$

# Security

- Security from *partial approximate common divisor problem* (PACDP) and *semiprime factorisation problem*
- Need $\rho$ large to negate two obvious attacks
  - "Collision" attack: if $c_1$ and $c_2$ are encryptions of $m$ then $c_1 - c_2$ is an encryption of 0
  - Mutiply $d$ ciphertexts to try get an encryption of zero

UNIVERSITY OF LEEDS

# Key Generation (KGen)

**Input** : $\lambda \in \mathcal{S}$
**Input** : $\rho \in \mathbb{Z}$: entropy of input
**Input** : $\rho' \in \mathbb{Z}$: effective entropy of inputs
**Output** : $(\kappa, p)$: secret key

1 $\nu \leftarrow \rho' - \rho$
2 $\kappa \xleftarrow{\mathsf{p}} [2^{\nu-1}, 2^{\nu}]$ // $\kappa > (n+1)^d M^d$
3 $p \xleftarrow{\mathsf{p}} [2^{\lambda-1}, 2^{\lambda}]$ // $p > (n+1)^d (M + \kappa^2)^d$
4 **return** $(\kappa, p)$

# Encryption (Enc)

**Input** : $m \in \mathcal{M}$
**Input** : $(\kappa, p)$: secret key
**Input** : modulus: public modulus
**Output** : $c \in \mathcal{C}$

1 $q \leftarrow \text{modulus}/p$
2 $r \xleftarrow{\$} [1, q)$
3 $s \xleftarrow{\$} [0, \kappa)$
4 $c \leftarrow m + s\kappa + rp \pmod{\text{modulus}}$
5 **return** $c$

UNIVERSITY OF LEEDS

## Decryption (Dec)

**Input** : $c \in \mathcal{C}$
**Input** : $(\kappa, p)$: secret key
**Output** : $m \in \mathcal{M}$

1 $m \leftarrow (c \bmod p) \bmod \kappa$
2 **return** $m$

# Key Generation (KGen)

**Input** : $\lambda \in \mathcal{S}$
**Input** : $\rho \in \mathbb{Z}$: entropy of inputs
**Output** : $(p, \boldsymbol{a})$: secret key
**Output** : modulus: public modulus

$p \xleftarrow{\mathsf{p}} [2^{\lambda-1}, 2^{\lambda}]$
$\eta \leftarrow \lambda^2/\rho - \lambda$
$q \xleftarrow{\mathsf{p}} [2^{\eta-1}, 2^{\eta}]$
modulus $\leftarrow pq$
**repeat**
  $a_i \xleftarrow{\$} [1, \text{modulus})\ (i \in [1, 2])$
**until** $a_1, a_2, a_1 - a_2 \neq 0\ (\mathrm{mod}\ p$ *and* $\mathrm{mod}\ q)$
$\boldsymbol{a} \leftarrow [a_1\ a_2]^T$
**return** $(p, \boldsymbol{a})$, modulus

# Parameter Generation (Pgen)

**Input** : $(p, \boldsymbol{a})$: secret key
**Input** : modulus: public modulus
**Output** : $R$: public re-encryption matrix

1 $q \leftarrow \text{modulus}/p$
2 $\beta \leftarrow 2(a_2 - a_1)^2$
3 $\varrho \xleftarrow{\$} [0, q)$
4 $\sigma \xleftarrow{\$} [0, \text{modulus})$
5 $\alpha_1 \leftarrow \beta^{-1}(\sigma a_1 + \varrho p - a_1^2)$
6 $\alpha_2 \leftarrow \beta^{-1}(\sigma a_2 + \varrho p - a_2^2)$
7 $R \leftarrow \begin{bmatrix} 1 - 2\alpha_1 & \alpha_1 & \alpha_1 \\ -2\alpha_2 & \alpha_2 + 1 & \alpha_2 \end{bmatrix}$
8 **return** $R$

## Encryption (Enc)

**Input** : $m \in \mathcal{M}$
**Input** : $(p, \boldsymbol{a})$: secret key
**Input** : modulus: public modulus
**Output** : $\boldsymbol{c} \in \mathcal{C}$

1 $q \leftarrow \text{modulus}/p$
2 $r \xleftarrow{\$} [0, q)$
3 $s \xleftarrow{\$} [0, \text{modulus})$
4 $\boldsymbol{c} \leftarrow (m + rp)\mathbf{1} + s\boldsymbol{a} \pmod{\text{modulus}}$
5 **return** $\boldsymbol{c}$

UNIVERSITY OF LEEDS

## Decryption (Dec)

**Input** : $\boldsymbol{c} \in \mathcal{C}$
**Input** : $(p, \boldsymbol{a})$: secret key
**Output** : $m \in \mathcal{M}$

1 $\boldsymbol{\gamma}^T \leftarrow (a_2 - a_1)^{-1}[a_2 \ \ -a_1]$
2 $m \leftarrow \boldsymbol{\gamma}^T \boldsymbol{c} \mod p$
3 **return** $m$

# **Multiplication (**Mult**)**

**Input** : $\boldsymbol{c} = [c_1\ c_2]^T \in \mathcal{C}$
**Input** : $\boldsymbol{c}' = [c'_1\ c'_2]^T \in \mathcal{C}$
**Input** : modulus $\in \mathbb{Z}$: public modulus
**Input** : $R$: re-encryption matrix
**Output** : result $\in \mathcal{C}$

1 $c_3 \leftarrow 2c_1 - c_2$
2 $\boldsymbol{c}_\star \leftarrow [c_1\ c_2\ c_3]^T$
3 $c'_3 \leftarrow 2c'_1 - c'_2$
4 $\boldsymbol{c}'_\star \leftarrow [c'_1\ c'_2\ c'_3]^T$
5 result $\leftarrow R(\boldsymbol{c}_\star \circ \boldsymbol{c}'_\star)\ (\mathrm{mod}$ modulus$)$
6 **return** result

# Key Generation (KGen)

**Input** : $\lambda \in \mathcal{S}$
**Input** : $\rho \in \mathbb{Z}$: entropy of input
**Input** : $\rho' \in \mathbb{Z}$: effective entropy of inputs
**Output** : $(\kappa, p, \boldsymbol{a})$: secret key
**Output** : modulus: public modulus

$\nu \leftarrow \rho' - \rho$
$\kappa \xleftarrow{\mathsf{p}} [2^{\nu-1}, 2^{\nu}]$
$p \xleftarrow{\mathsf{p}} [2^{\lambda-1}, 2^{\lambda}]$
$\eta \leftarrow \lambda^2/\rho' - \lambda$
$q \xleftarrow{\mathsf{p}} [2^{\eta-1}, 2^{\eta}]$
modulus $\leftarrow pq$
**repeat**
  $a_i \xleftarrow{\$} [1, \text{modulus})$ $(i \in [1, 2])$
**until** $a_1, a_2, a_1 - a_2 \neq 0 \ (\mathrm{mod}\ p$ *and* $\mathrm{mod}\ q)$
$\boldsymbol{a} \leftarrow [a_1\ a_2]^T$
**return** $(\kappa, p, \boldsymbol{a})$, modulus

# Encryption (Enc)

**Input** : $m \in \mathcal{M}$
**Input** : $(\kappa, p, \boldsymbol{a})$: secret key
**Input** : modulus: public modulus
**Output** : $\boldsymbol{c} \in \mathcal{C}$

1 $q \leftarrow \text{modulus}/p$
2 $r \xleftarrow{\$} [0, q)$
3 $s \xleftarrow{\$} [0, \kappa)$
4 $t \xleftarrow{\$} [0, \text{modulus})$
5 $\boldsymbol{c} \leftarrow (m + rp + s\kappa)\mathbf{1} + t\boldsymbol{a} \pmod{\text{modulus}}$
6 **return** $\boldsymbol{c}$

UNIVERSITY OF LEEDS

## Decryption (Dec)

**Input** : $\boldsymbol{c} \in \mathcal{C}$
**Input** : $(\kappa, p, \boldsymbol{a})$: secret key
**Output** : $m \in \mathcal{M}$

1 $\boldsymbol{\gamma}^T \leftarrow (a_2 - a_1)^{-1}[a_2 \ \ -a_1]$
2 $m \leftarrow (\boldsymbol{\gamma}^T \boldsymbol{c} \mod p) \mod \kappa$
3 **return** $m$

# Experimental Setup

- Implemented in pure, unoptimised Java using JScience [1] library
- Simple MapReduce algorithm to compute degree $d$ multivariate polynomial ($n$ inputs)
  - Input: $n/d$ rows of $d$ encrypted integers
  - Map phase multiplies the integers on a row
  - Reduce phase sums these products
- 24,000 inputs (scaled to 106,000,000 inputs)
- Secure 16GB client to generate, encrypt and decrypt data
- Hadoop 2.7.3 cluster on 17 2GB VMs (one master, 16 slaves)
- Results compared with literature
  - 10 to 1000 times faster to compute products and sums

# HE1 Results

| Alg. | Parameters | | Encryption | | MR Job | | | Decrypt |
|---|---|---|---|---|---|---|---|---|
| | $d$ | $\rho$ | Init(s) | Enc($\mu$s) | Exec(s) | Prod($\mu$s) | Sum($\mu$s) | (ms) |
| HE1 | 2 | 32 | 0.12 | 13.52 | 23.82 | 54.41 | 9.06 | 0.21 |
| HE1 | 2 | 64 | 0.12 | 16.24 | 23.85 | 60.38 | 8.04 | 0.49 |
| HE1 | 2 | 128 | 0.15 | 25.73 | 23.77 | 84.69 | 8.43 | 0.28 |
| HE1 | 3 | 32 | 0.17 | 22.98 | 23.65 | 87.75 | 11.46 | 0.35 |
| HE1 | 3 | 64 | 0.19 | 34.63 | 24.72 | 95.68 | 12.37 | 0.45 |
| HE1 | 3 | 128 | 0.42 | 54.83 | 26.05 | 196.71 | 14.07 | 0.55 |
| HE1 | 4 | 32 | 0.28 | 43.36 | 24.48 | 108.72 | 13.75 | 0.5 |
| HE1 | 4 | 64 | 0.53 | 58.85 | 26.41 | 227.44 | 15.85 | 3.59 |
| HE1 | 4 | 128 | 1.36 | 104.95 | 28.33 | 484.95 | 16.92 | 5.67 |



# HE1N Results

| Alg. | Parameters | | | Encryption | | MR Job | | | Decrypt |
|---|---|---|---|---|---|---|---|---|---|
| | $d$ | $\rho$ | $\rho'$ | Init(s) | Enc($\mu$s) | Exec(s) | Prod($\mu$s) | Sum($\mu$s) | (ms) |
| HE1N | 2 | 1 | 32 | 0.22 | 32.99 | 22.94 | 88.38 | 8.53 | 3.35 |
| HE1N | 2 | 1 | 64 | 0.39 | 52.63 | 26.24 | 168.54 | 12.39 | 3.56 |
| HE1N | 2 | 1 | 128 | 1.2 | 89.01 | 26.18 | 226.2 | 13.16 | 8.1 |
| HE1N | 2 | 8 | 32 | 0.6 | 57.88 | 25.9 | 177.36 | 11.17 | 7.18 |
| HE1N | 2 | 8 | 64 | 0.32 | 43.93 | 26.53 | 96.78 | 12.18 | 2.27 |
| HE1N | 2 | 8 | 128 | 1.13 | 78.11 | 24.42 | 212.75 | 11.07 | 8.4 |
| HE1N | 2 | 16 | 64 | 0.33 | 53.97 | 27.15 | 168 | 13.67 | 4.47 |
| HE1N | 2 | 16 | 128 | 0.63 | 68.73 | 25.22 | 194.42 | 11.01 | 7.65 |
| HE1N | 3 | 1 | 32 | 8.54 | 183.19 | 24.24 | 522.07 | 12.06 | 9.09 |
| HE1N | 3 | 1 | 64 | 3.67 | 125 | 29.49 | 467.36 | 18.22 | 11.43 |
| HE1N | 3 | 1 | 128 | 27.84 | 313.76 | 26.94 | 1235.77 | 15.04 | 11.75 |
| HE1N | 3 | 8 | 32 | 115 | 462.45 | 32.61 | 1556.17 | 21.11 | 19.79 |
| HE1N | 3 | 8 | 64 | 9.75 | 180.08 | 25.87 | 500.62 | 15.03 | 10.39 |
| HE1N | 3 | 8 | 128 | 36.05 | 259.15 | 30.1 | 836.27 | 20.68 | 11.45 |
| HE1N | 3 | 16 | 64 | 30.96 | 378.99 | 28.24 | 1338.33 | 15.51 | 13.3 |
| HE1N | 3 | 16 | 128 | 8.13 | 226.32 | 27.92 | 621.95 | 18.01 | 10.89 |

# HE2 Results

| Alg. | Parameters | | Encryption | | MR Job | | | Decrypt |
|---|---|---|---|---|---|---|---|---|
| | $d$ | $\rho$ | Init(s) | Enc($\mu$s) | Exec(s) | Prod($\mu$s) | Sum($\mu$s) | (ms) |
| HE2 | 2 | 32 | 0.16 | 85.79 | 26.82 | 305.52 | 11.68 | 4.83 |
| HE2 | 2 | 64 | 0.17 | 95.92 | 29.71 | 354.79 | 16.9 | 3.26 |
| HE2 | 2 | 128 | 0.22 | 132.53 | 32.84 | 540.78 | 22.83 | 4.92 |
| HE2 | 3 | 32 | 0.23 | 130.3 | 31.18 | 513.93 | 23.77 | 6.52 |
| HE2 | 3 | 64 | 0.29 | 145.62 | 32.84 | 615.9 | 24.61 | 6.3 |
| HE2 | 3 | 128 | 0.52 | 249.47 | 29.54 | 1443.82 | 16.56 | 18.34 |
| HE2 | 4 | 32 | 0.39 | 175.63 | 29.5 | 733.23 | 20.69 | 6.01 |
| HE2 | 4 | 64 | 0.7 | 255.3 | 29.55 | 1578.39 | 18.29 | 16.24 |
| HE2 | 4 | 128 | 2.7 | 465.51 | 37.47 | 2943.91 | 22.15 | 15.41 |

# HE2N Results

| Alg. | Parameters | | | Encryption | | MR Job | | | Decrypt |
|---|---|---|---|---|---|---|---|---|---|
| | $d$ | $\rho$ | $\rho'$ | Init(s) | Enc($\mu$s) | Exec(s) | Prod($\mu$s) | Sum($\mu$s) | (ms) |
| HE2N | 2 | 1 | 32 | 0.27 | 147.83 | 29.74 | 571.94 | 16.58 | 5.66 |
| HE2N | 2 | 1 | 64 | 0.43 | 202.74 | 33.36 | 1291.68 | 18.3 | 13.23 |
| HE2N | 2 | 1 | 128 | 1.58 | 354.19 | 33.76 | 1977.51 | 17.13 | 12.46 |
| HE2N | 2 | 8 | 32 | 0.59 | 234.83 | 31.42 | 1413.31 | 15.21 | 14.92 |
| HE2N | 2 | 8 | 64 | 0.33 | 163.78 | 27.42 | 635.64 | 13.6 | 6.18 |
| HE2N | 2 | 8 | 128 | 0.9 | 307.68 | 36.32 | 1850.83 | 21.71 | 15.79 |
| HE2N | 2 | 16 | 64 | 0.42 | 208.1 | 29.96 | 1230.56 | 13.41 | 13.16 |
| HE2N | 2 | 16 | 128 | 0.73 | 274.48 | 30.82 | 1585.1 | 14.85 | 15.04 |
| HE2N | 3 | 1 | 32 | 5.72 | 651.1 | 36.49 | 3438.96 | 18.67 | 19.05 |
| HE2N | 3 | 1 | 64 | 4.45 | 477.52 | 35.33 | 3073.46 | 18.75 | 19.77 |
| HE2N | 3 | 1 | 128 | 26.83 | 1192.79 | 43.23 | 6416.43 | 22.48 | 25.12 |
| HE2N | 3 | 8 | 32 | 87.38 | 1658.36 | 49.63 | 8139.19 | 23.71 | 27.24 |
| HE2N | 3 | 8 | 64 | 5.21 | 607.75 | 36.54 | 3337.1 | 22.28 | 17.39 |
| HE2N | 3 | 8 | 128 | 17.14 | 945.64 | 40.49 | 4620.69 | 25.91 | 22.41 |
| HE2N | 3 | 16 | 64 | 39.19 | 1368.18 | 44.88 | 7005.7 | 24.1 | 28.3 |
| HE2N | 3 | 16 | 128 | 11.39 | 774.07 | 36.05 | 3845.1 | 20.29 | 20.74 |

# Conclusion

- 10 to 1000 times faster than results reported in literature
- Generalisable to $k$ dimension ciphertexts (see Appendix)
  - This HE$k$ scheme can be transformed to an FHE scheme.

# References I


J.-M. Dautelle. *JScience*. Version 4.3.1. Sept. 2014. URL: http://jscience.org.


M. van Dijk et al. 'Fully Homomorphic Encryption over the Integers'. In: *Proc. EUROCRYPT '10*. 2010, pp. 24–43.


R. A. Popa et al. 'CryptDB: Protecting Confidentiality With Encrypted Query Processing'. In: *Proc. SOSP '11*. 2011, pp. 85–100.