Low Latency Privacy Preserving Inference

Alon Brutzkuz, Oren Elisha, Ran Gilad-Bachrach

Microsoft Research, June 2019

Proceedings of the 36th International Conference on Machine Learning, PMLR

Outline

- Introduction
- Possible scenarios
- More specific
  - BFV encryption
  - Encrypted inference (LoLa)
- Research directions

Outline

- Introduction
- Possible scenarios
- More specific
  - BFV encryption
  - Encrypted inference (LoLa)
- Research directions

Homomorphic encryption

- $D\big(E(m_1) * E(m_2)\big) = m_1 * m_2$
- $D\big(E(m_1) + E(m_2)\big) = m_1 + m_2$
- Partially homomorphic encryption
  - Either addition or multiplication
- Somewhat homomorphic encryption
  - Addition and multiplication with limited number of operations
- Fully homomorphic encryption
  - Addition and multiplication without limitation

Example

- $D\big(E(m_1) * E(m_2)\big) = m_1 * m_2$
- $D\big(E(m_1) + E(m_2)\big) = m_1 + m_2$

https://www.youtube.com/watch?v=pXb39wj5ShI&ab_channel=OfficeoftheDirectorofNationalIntelligence

Example

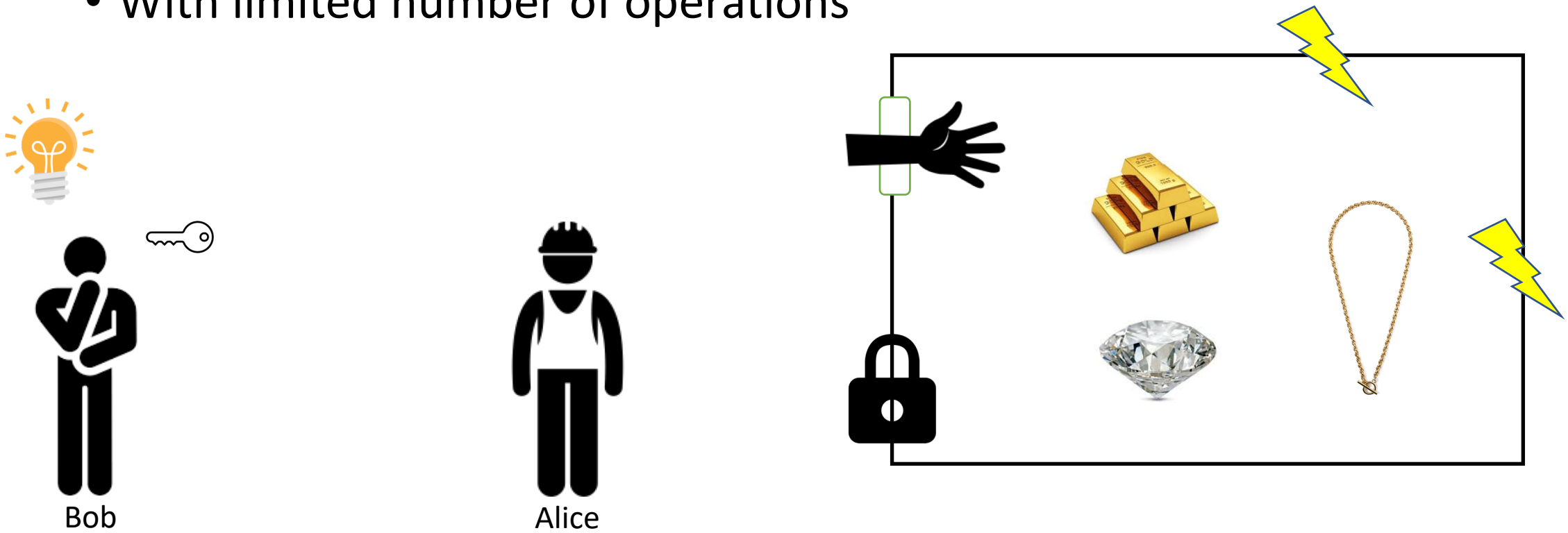
- $D\big(E(m_1) * E(m_2)\big) = m_1 * m_2$
- $D\big(E(m_1) + E(m_2)\big) = m_1 + m_2$

Bob

Alice

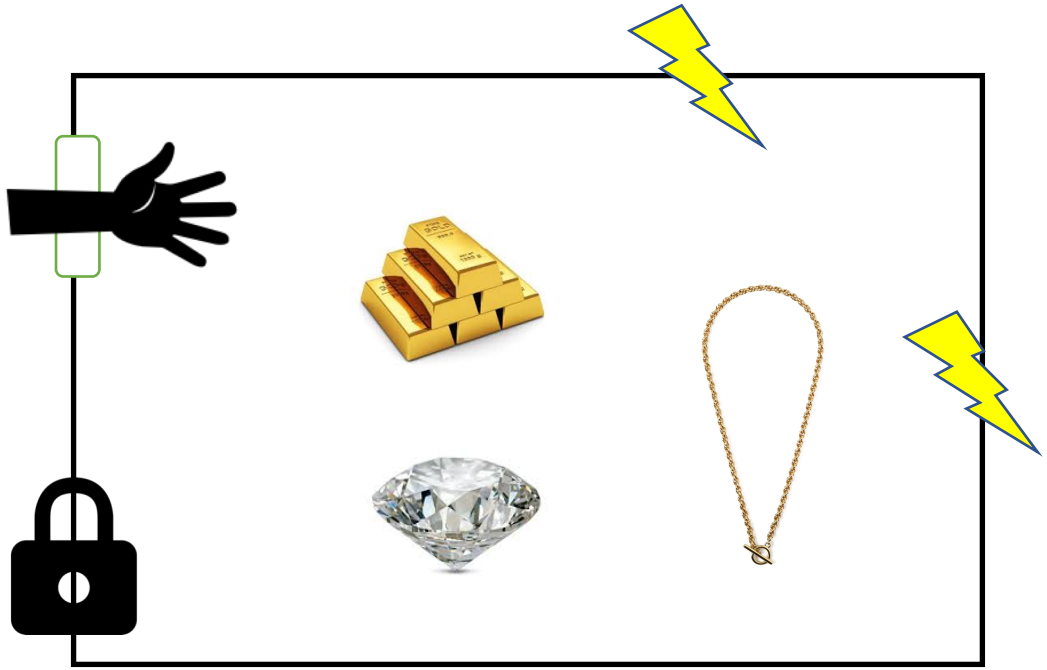
Somewhat homomorphic encryption

- $D\big(E(m_1) * E(m_2)\big) = m_1 * m_2$
- $D\big(E(m_1) + E(m_2)\big) = m_1 + m_2$
- With limited number of operations



Somewhat homomorphic encryption

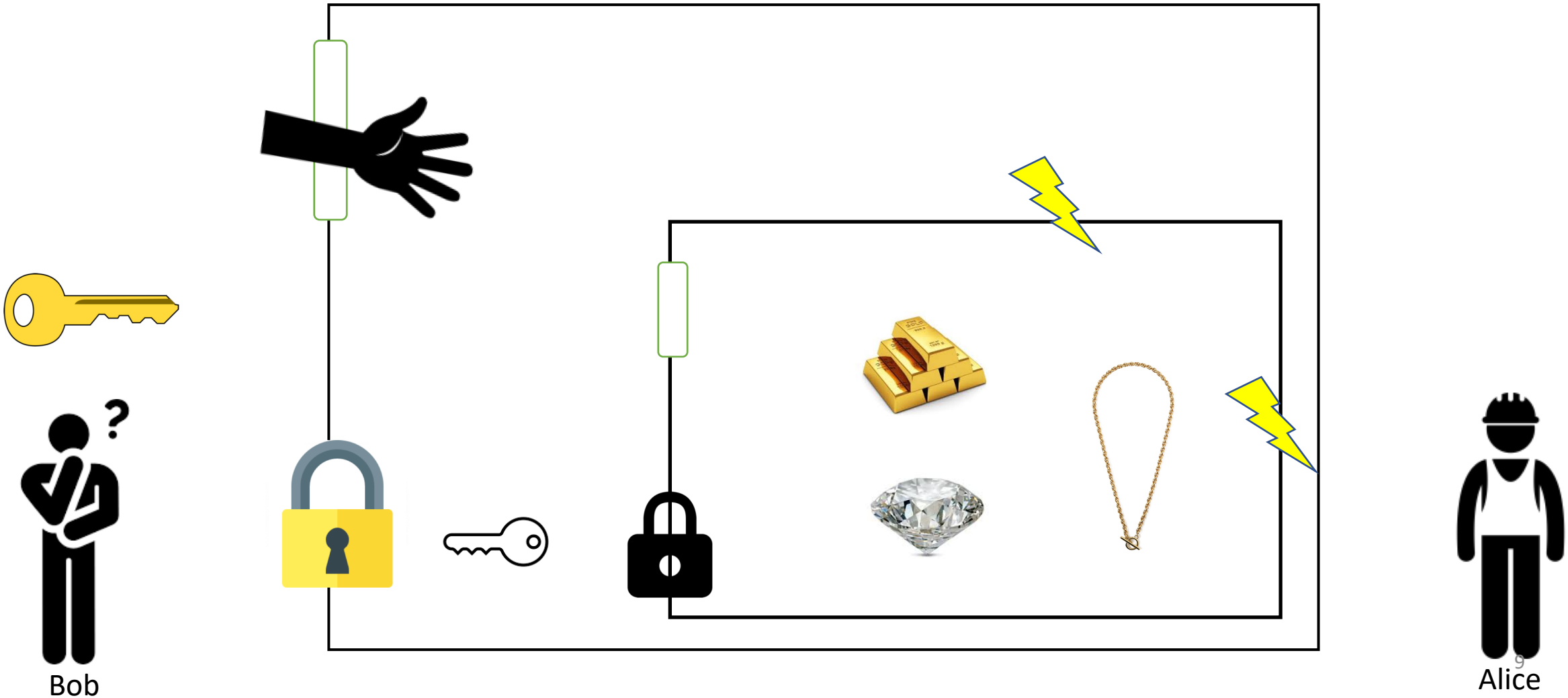
- $D\big(E(m_1) * E(m_2)\big) = m_1 * m_2$
- $D\big(E(m_1) + E(m_2)\big) = m_1 + m_2$



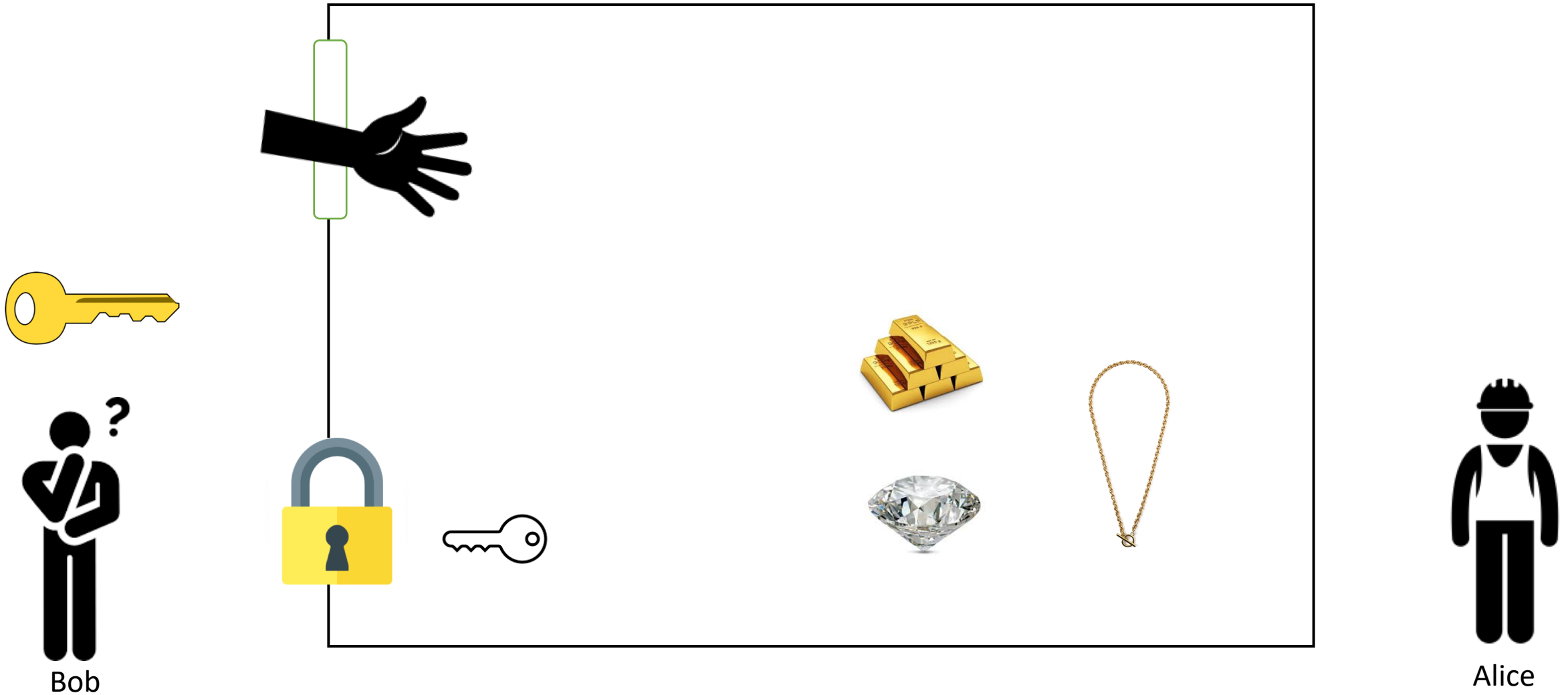
Addition

Multiplication

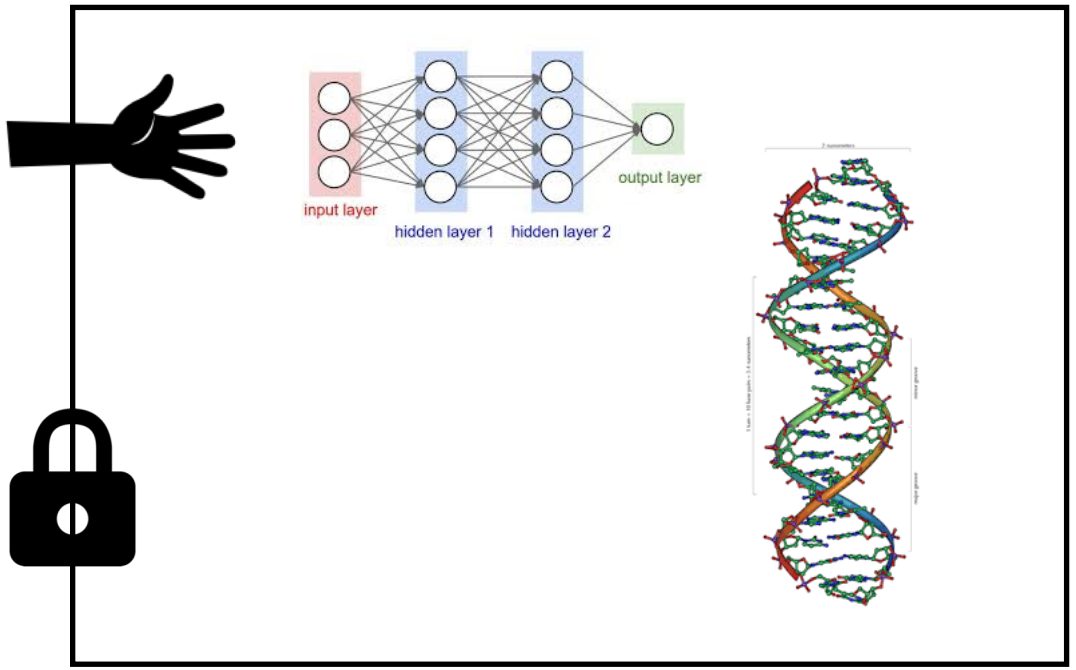
Bootstrapping



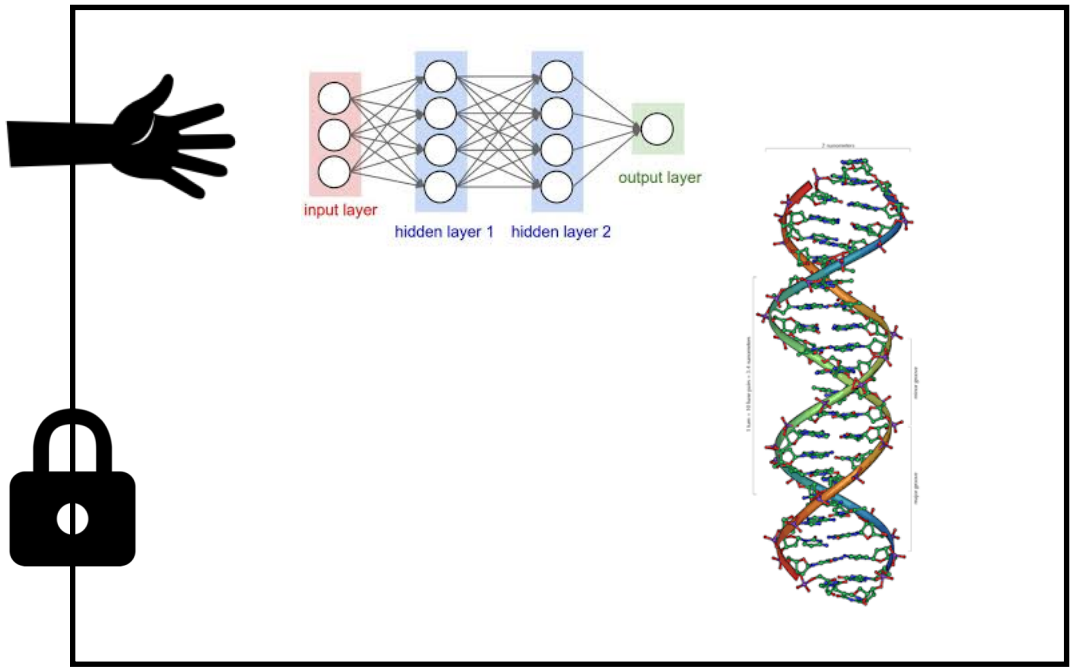
Fully homomorphic encryption

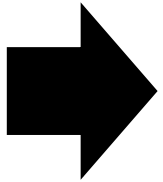


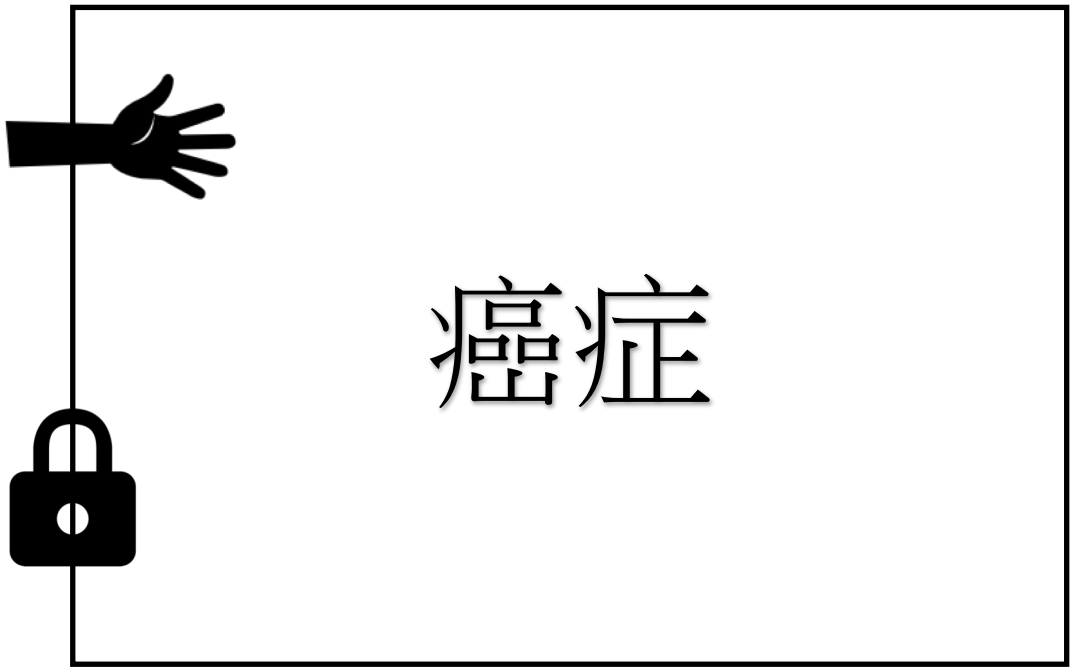
Analogy to encrypted inference



Analogy to encrypted inference



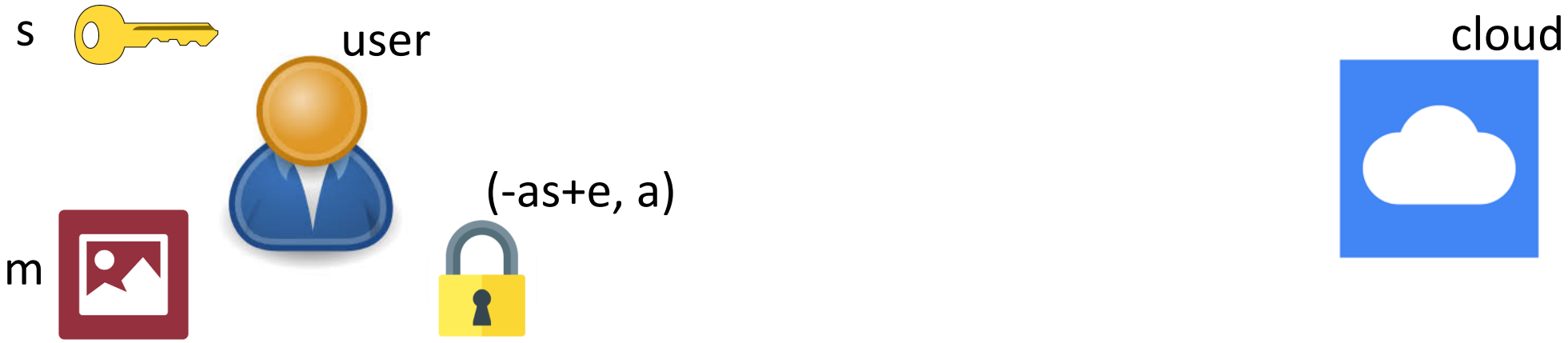




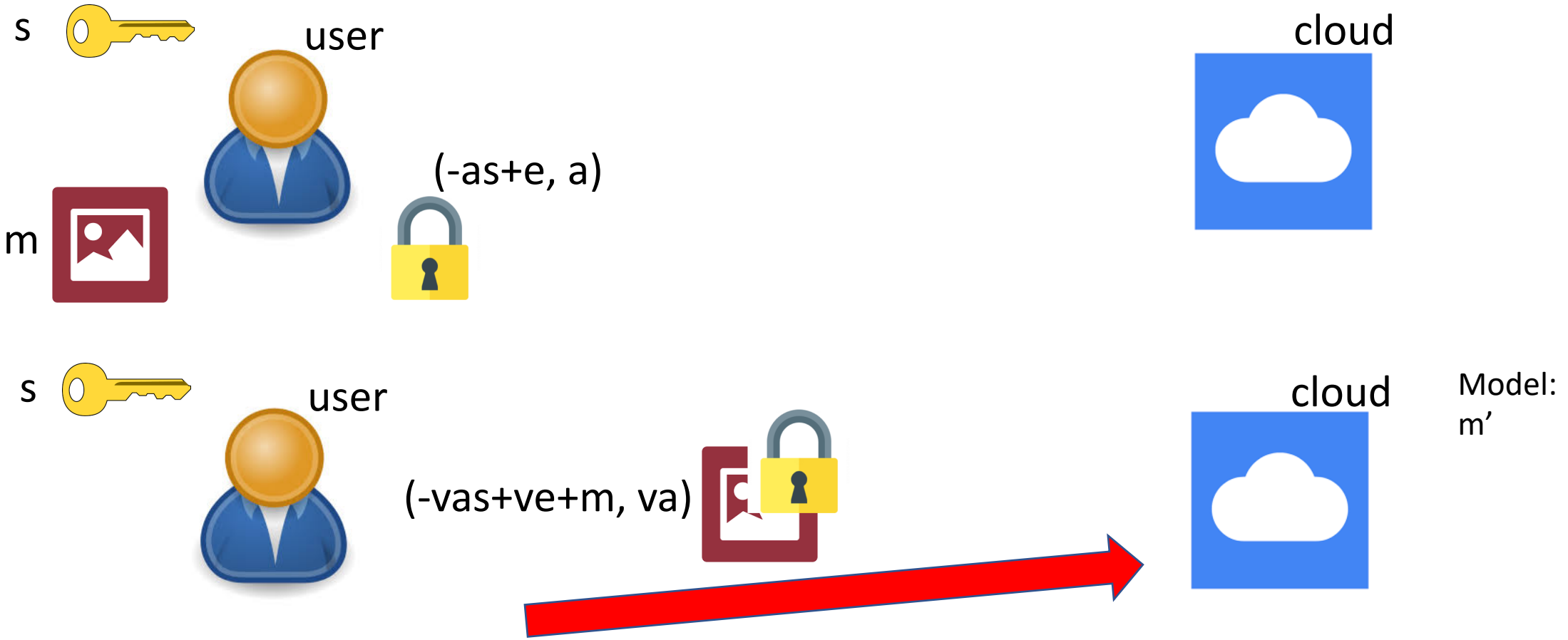
Outline

- Introduction
- Possible scenarios
- More specific
  - BFV encryption
  - Encrypted inference (LoLa)
- Research directions

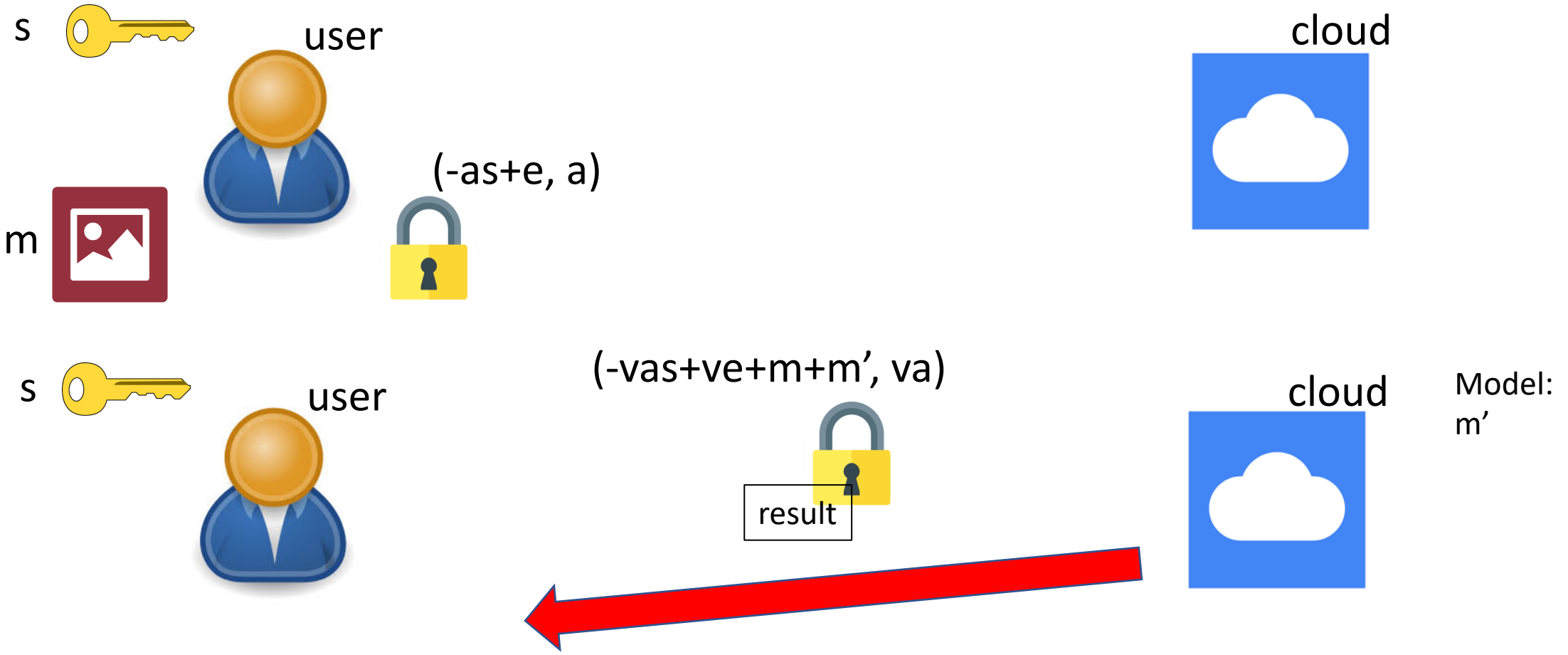
HE scenario 1 (Cloud inference)



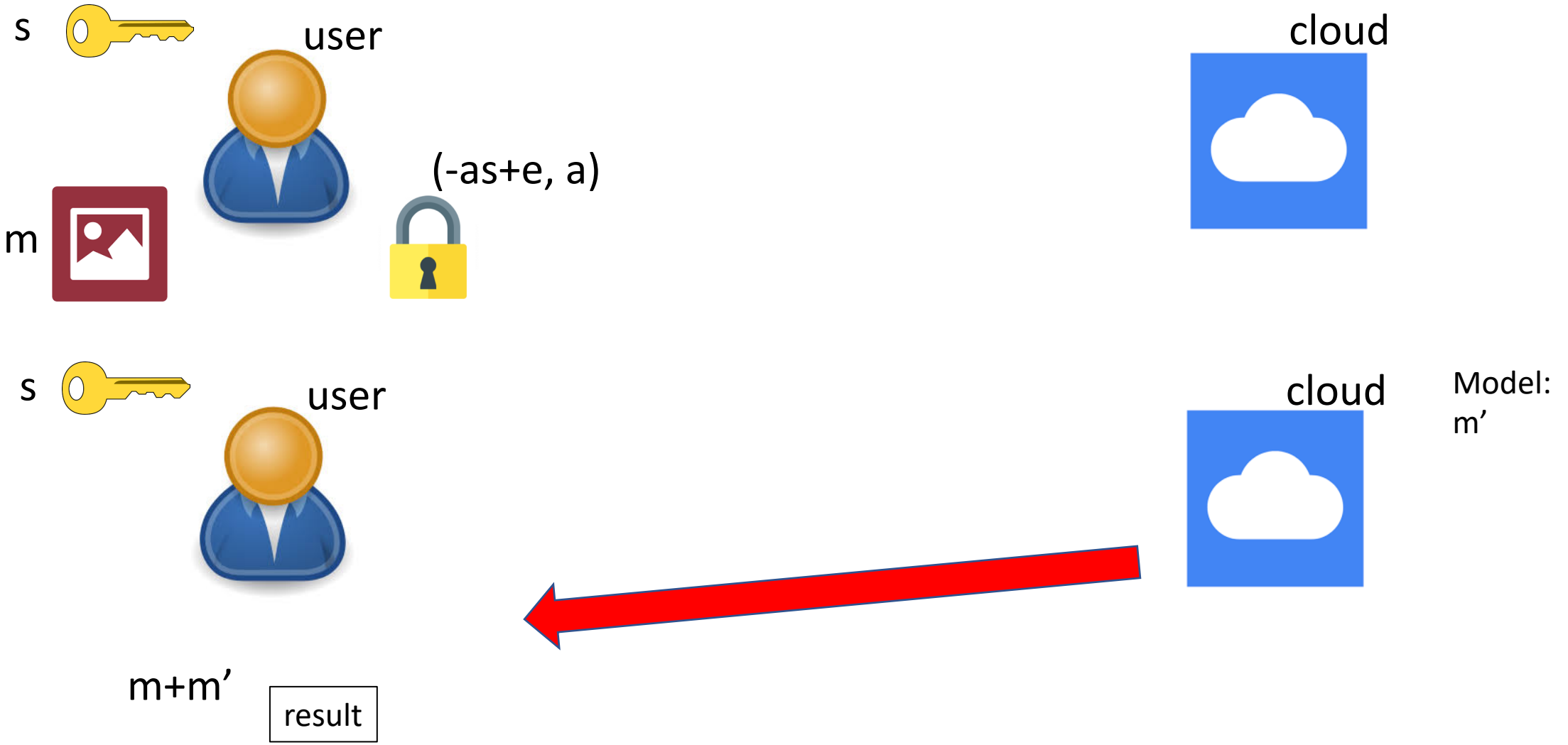
HE scenario 1 (Cloud inference)



HE scenario 1 (Cloud inference)

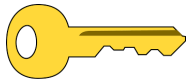


HE scenario 1 (Cloud inference)



HE scenario 2 (Cloud computation)

s



user

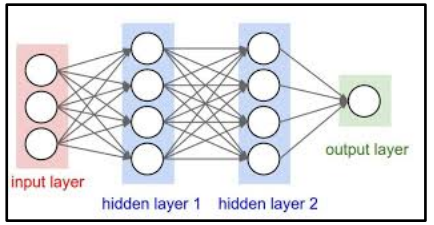
m

(-vas+ve+m, va)

s’

model provider

m’

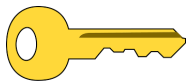


(-a’s’+e’, a’)

Computation provider

HE scenario 2 (Cloud computation)

s



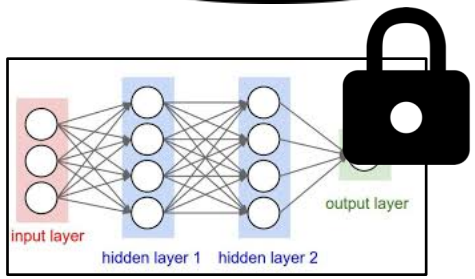
user

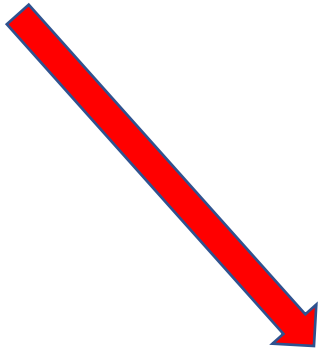
(-vas+ve+m, va)

s’

model provider

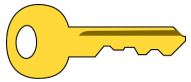
(-v’a’s’+v’e’+m’, v’a’)





Computation provider

HE scenario 2 (Cloud computation)

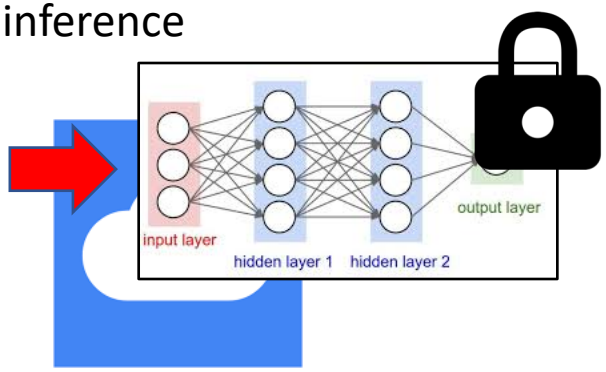
s 

user

s’ 

model provider

inference

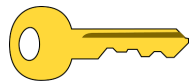


Computation provider

(-v’a’s’+v’e’+m’-vas+ve+m, va+v’a’)

HE scenario 2 (Cloud computation)

s



user

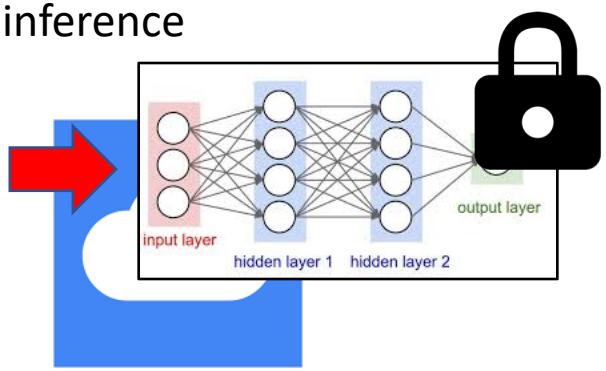
s’

model provider

Stage1 decryption (消掉a’)

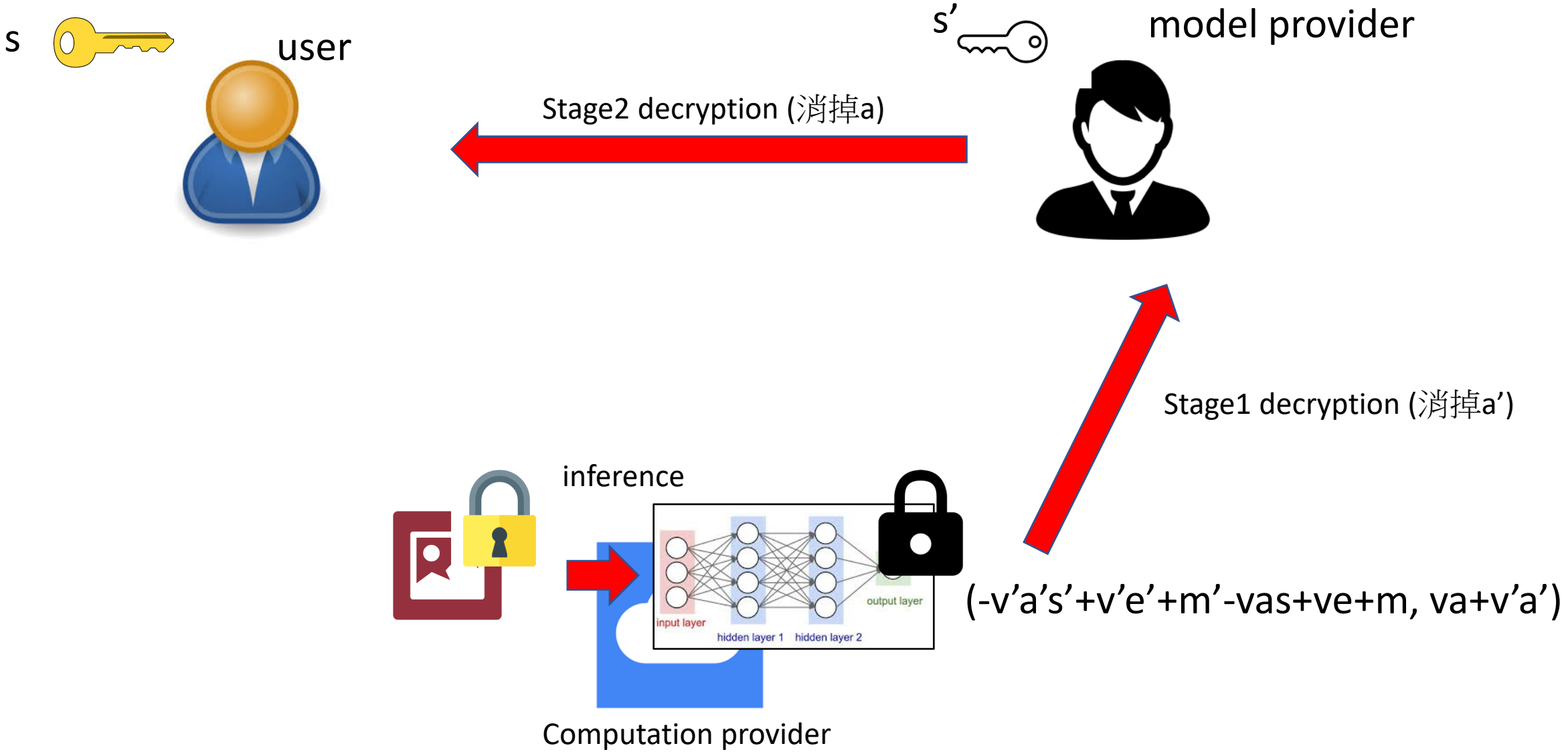
(-v’a’s’+v’e’+m’-vas+ve+m, va+v’a’)

inference



Computation provider

HE scenario 2 (Cloud computation)



Possible applications

Domain	Genomics	Health	National Security	Education
Topic	Match Maker	Billing and Reporting	Smart Grid (Municipal Service)	School Dropouts
Data Owner	Medical Institutions	Small Hospital, Clinic	Nodes and Network	School, Hospital, Welfare
Latency of Service	Hours	Hours	Quasi-Real Time	Week
Data volume (size x no)	DB O(1000X1MB) Query O(1KB)	O(10M) x O(1M)	O(1M) x O(1M)	O(1K) x O(1M)
Data persistency	Add Only	Add Only	Add Only	Add Only
Technical Issues	Comparison, Sorting Auditing Privacy	Tabulation, Linear Algebra	Comparison	Comparison Matrix Analysis
When is possible	1 years	2-3 years	Now	2-3 years
Why HE?	HIPAA	Cyber insurance	Privacy	FERPA
Who pays?	Health Insurance	Hospital	Energy Company	DoE

https://dualitytech.com/wp-content/uploads/2020/04/Covid19_onepager.pdf
https://www.researchgate.net/publication/320976976_APPLICATIONS_OF_HOMOMORPHIC_ENCRYPTION

Possible applications

Domain	Genomics	Health	National Security	Education
Topic	Match Maker	Billing and Reporting	Smart Grid (Municipal Service)	School Dropouts
Data Owner	Medical Institutions	Small Hospital, Clinic	Nodes and Network	School, Hospital, Welfare
Latency of Service	Hours	Hours	Quasi-Real Time	Week
Data volume (size x no)	DB O(1000X1MB) Query O(1KB)	O(10M) x O(1M)	O(1M) x O(1M)	O(1K) x O(1M)
Data persistency	Add Only	Add Only	Add Only	Add Only
Technical Issues	Comparison, Sorting Auditing Privacy	Tabulation, Linear Algebra	Comparison	Comparison Matrix Analysis
When is possible	1 years	2-3 years	Now	2-3 years
Why HE?	HIPAA	Cyber insurance	Privacy	FERPA
Who pays?	Health Insurance	Hospital	Energy Company	DoE

https://dualitytech.com/wp-content/uploads/2020/04/Covid19_onepager.pdf
https://www.researchgate.net/publication/320976976_APPLICATIONS_OF_HOMOMORPHIC_ENCRYPTION

Privacy-Preserving, Large-Scale COVID-19 Contact Tracing

Duality SecurePlus™ Query

Secure Query 1

Input: Phone number of confirmed case and date range

Output: Locations, dates, and times where phone was pinged

How it's privacy enhanced:

- Location data providers cannot see PII inputted or received by healthcare authorities
- Healthcare authorities cannot see PII related to individuals outside the area of exposure

Secure Query 2

Input: Locations, dates, and times where phone was pinged

Output: Individuals at the same place and same times as infected individual

How it's privacy enhanced:

- Location data providers cannot see PII inputted or received by healthcare authorities
- Healthcare authorities cannot see PII related to individuals not in the area of exposure

Action

Healthcare Authorities can

- Contact individuals suspected of exposure
- Contact exposed businesses
- Better coordinate with local health, law enforcement, and government agencies

Partners and Recognition

intel · CDA Cyber Defence Alliance · Fast Company Most Innovative Companies 2020 · DARPA · Scotiabank · University of Michigan · Gartner Cool Vendor 2019 · NIH National Institutes of Health Turning Discovery Into Health · Hearst · Homomorphic Encryption Standartization

Possible applications

感染者的電話號碼和日期範圍(加密的)
→感染者什麼時候經過哪裡

時間地點(加密的)
→哪些人在同時間經過這裡

PII: Personal Identifiable Information

https://dualitytech.com/wp-content/uploads/2020/04/Covid19_onepager.pdf
https://www.researchgate.net/publication/320976976_APPLICATIONS_OF_HOMOMORPHIC_ENCRYPTION

Open source

Name	Developer	BGV[17]	CKKS[35]	BFV[20]	FHEW [32]	CKKS Bootstrapping [39]	TFHE[33]	Description
HElib[40]	IBM	Yes	Yes	No	No	No	No	BGV scheme with the GHS optimizations.
Microsoft SEAL[41]	Microsoft	No	Yes	Yes	No	No	No	
PALISADE[42]	Consortium of DARPA–funded defense contractors and academics: New Jersey Institute of Technology, Duality Technologies, Raytheon BBN Technologies, MIT, University of California, San Diego and others.	Yes	Yes	Yes	Yes	No	Yes	General–purpose lattice cryptography library.
HEAAN[43]	Seoul National University	No	Yes	No	No	Yes	No	
FHEW[32]	Leo Ducas and Daniele Micciancio	No	No	No	Yes	No	No	
TFHE[33]	Ilaria Chillotti, Nicolas Gama, Mariya Georgieva and Malika Izabachene	No	No	No	No	No	Yes	
FV–NFLlib[44]	CryptoExperts	No	No	Yes	No	No	No	
NuFHE[45]	NuCypher	No	No	No	No	No	Yes	Provides a GPU implementation of TFHE.
Lattigo[46]		No	Yes	Yes	No	No	No	Implementation in Go along with their distributed variants enabling Secure multi–party computation.

Homomorphic encryption summary

- Privacy preserving computation
  - Many use cases
- Huge computational cost (10000x)
  - GPU
  - FPGA
  - ASIC

Outline

- Introduction
- Possible scenarios
- More specific
  - BFV encryption
  - Encrypted inference (LoLa)
- Research directions

RSA encrypt

- $c = Enc(m) = \ m^e \ mod \ n, \ n = pq$
- $Enc(m_1) * Enc(m_2) = (m_1 m_2)^e = Enc(m_1 * m_2)$
- Example
  - $c = Enc(m) = \ m^5 \ mod \ 35$
  - $m_1 = 2, m_2 = 3$
  - $Enc(m_1) = 32, Enc(m_2) = 33$
  - $Enc(m_1) * Enc(m_2) = 16 * 11 \ mod \ 35 = 6 = 2 * 3$
- Only multiplication

Lattice-based cryptography

- Multiplication and addition
- Quantum-resistant
- Fair enough speed and memory usage
- Easy to understand

Computation on $\frac{Z_q}{x^n+1}$

- Example $\frac{Z_7}{x^4+1}$
  - $a \rightarrow 6x^3 + 4x^2 + 1x^1 + 2x^0$
  - $b \rightarrow 1x^3 + 1x^2 + 0x^1 + 1x^0$
  - $a + b \rightarrow 0x^3 + 5x^2 + 1x^1 + 3x^0$

Computation on $\frac{Z_q}{x^n+1}$

- Example $\frac{Z_7}{x^4+1}$
  - $a \to 6x^3 + 4x^2 + 1x^1 + 2x^0$
  - $b \to 1x^3 + 1x^2 + 0x^1 + 1x^0$
  - $a + b \to 0x^3 + 5x^2 + 1x^1 + 3x^0$
  - $a * b \to 6x^6 + 10x^5 + 5x^4 + 9x^3 + 6x^2 + 1x^1 + 2x^0$
    $\to 6(-x^2) + 10(-x) + 5(-1) + 9x^3 + 6x^2 + 1x^1 + 2x^0$
    $\to 9x^3 + 0x^2 - 9x^1 - 3x^0$ (plug in $x^4 = -1$)
    $\to 2x^3 + 0x^2 + 5x^1 + 4x^0$ (mod 7)

Computation on $\frac{Z_q}{x^n+1}$

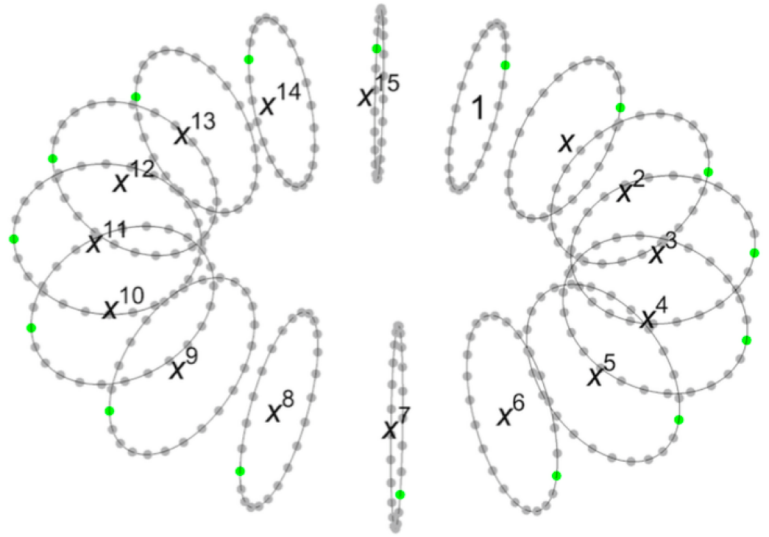


Figure 4.1: Depiction of a polynomial ring with degree 16 and plaintext modulus 24 [35].

- Example $\frac{Z_7}{x^4+1}$
  - $a \rightarrow 6x^3 + 4x^2 + 1x^1 + 2x^0$
  - $b \rightarrow 1x^3 + 1x^2 + 0x^1 + 1x^0$
  - $a + b \rightarrow 0x^3 + 5x^2 + 1x^1 + 3x^0$
  - $a * b \rightarrow 6x^6 + 10x^5 + 5x^4 + 9x^3 + 6x^2 + 1x^1 + 2x^0$

$\rightarrow 6(-x^2) + 10(-x) + 5(-1) + 9x^3 + 6x^2 + 1x^1 + 2x^0$

$\rightarrow 9x^3 + 0x^2 - 9x^1 - 3x^0$ (plug in $x^4 = -1$)

$\rightarrow 2x^3 + 0x^2 + 5x^1 + 4x^0$ (mod 7)

https://scholarworks.uark.edu/cgi/viewcontent.cgi?article=1074&context=csceuht

BFV Keygen

- Plaintext domain in $\frac{Z_t}{x^n+1}$, encrypted domain in $\frac{Z_q}{x^n+1}$, q >> t

https://eprint.iacr.org/2012/144.pdf

BFV Keygen

- Plaintext domain in $\frac{Z_t}{x^n+1}$, encrypted domain in $\frac{Z_q}{x^n+1}$, q >> t
- Example $\frac{Z_7}{x^4+1}$ (t=7, 3bits), $\frac{Z_{1153}}{x^4+1}$ (q=1153, 10bits), $\frac{q}{t} = 164$ (7 bits)

https://eprint.iacr.org/2012/144.pdf

BFV Keygen

- Plaintext domain in $\frac{Z_t}{x^n+1}$, encrypted domain in $\frac{Z_q}{x^n+1}$, q >> t
- Example $\frac{Z_7}{x^4+1}$ (t=7, 3bits), $\frac{Z_{1153}}{x^4+1}$ (q=1153, 10bits), $\frac{q}{t} = 164$ (7 bits)
- Keygen
  - $a = 123x^3 + 456x^2 + 789x^1 + 12x^0 \in \frac{Z_{1153}}{x^4+1}$ (encrypted domain)
  - $s = 0x^3 + 1x^2 + 0x^1 + 1x^0$ (**secret key**, binary coefficients)
  - $e = 1x^3 + 0x^2 + 1x^1 + 0x^0$ (**noise**, binary coefficients)
  - $pk = (e - a * s,\ a).\quad (a * s = 912\mathrm{x}^3 + 468\mathrm{x}^2 + 666\mathrm{x}^1 + 709\mathrm{x}^0)$

https://eprint.iacr.org/2012/144.pdf

BFV Keygen

- Plaintext domain in $\frac{Z_t}{x^n+1}$, encrypted domain in $\frac{Z_q}{x^n+1}$, q >> t
- Example $\frac{Z_7}{x^4+1}$ (t=7, 3bits), $\frac{Z_{1153}}{x^4+1}$ (q=1153, 10bits), $\frac{q}{t} = 164$ (7 bits)
- Keygen
  - $a = 123x^3 + 456x^2 + 789x^1 + 12x^0 \in \frac{Z_{1153}}{x^4+1}$ (encrypted domain)
  - $s = 0x^3 + 1x^2 + 0x^1 + 1x^0$ (**secret key**, binary coefficients)
  - $e = 1x^3 + 0x^2 + 1x^1 + 0x^0$ (**noise**, binary coefficients)
  - $pk = (e - a * s,\ a).\quad (a * s = 912\mathrm{x}^3 + 468\mathrm{x}^2 + 666\mathrm{x}^1 + 709\mathrm{x}^0)$
  - Only the party with s can cancel out the term $a * s$

https://eprint.iacr.org/2012/144.pdf

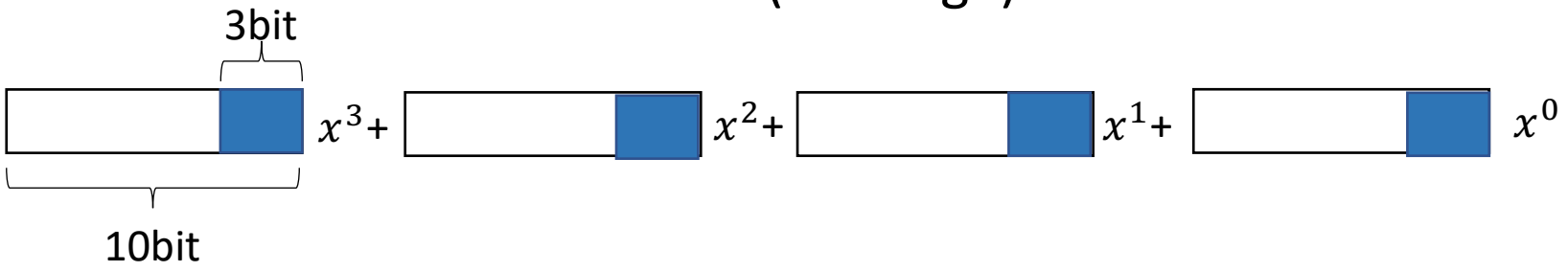
BFV Keygen

- Plaintext domain in $\frac{Z_t}{x^n+1}$, encrypted domain in $\frac{Z_q}{x^n+1}$, q >> t
- Example $\frac{Z_7}{x^4+1}$ (t=7, 3bits), $\frac{Z_{1153}}{x^4+1}$ (q=1153, 10bits), $\frac{q}{t} = 164$ (7 bits)
- Keygen
  - $a = 123x^3 + 456x^2 + 789x^1 + 12x^0 \in \frac{Z_{1153}}{x^4+1}$ (encrypted domain)
  - $s = 0x^3 + 1x^2 + 0x^1 + 1x^0$ (**secret key**, binary coefficients)
  - $e = 1x^3 + 0x^2 + 1x^1 + 0x^0$ (**noise**, binary coefficients)
  - $pk = (e - a * s,\ a).\quad (a * s = 912\mathrm{x}^3 + 468\mathrm{x}^2 + 666\mathrm{x}^1 + 709\mathrm{x}^0)$
  - Only the party with s can cancel out the term $a * s$
  - Hardness: given pk, hard to find s (reduce to SVP in lattice which is NP-Hard)

https://eprint.iacr.org/2012/144.pdf

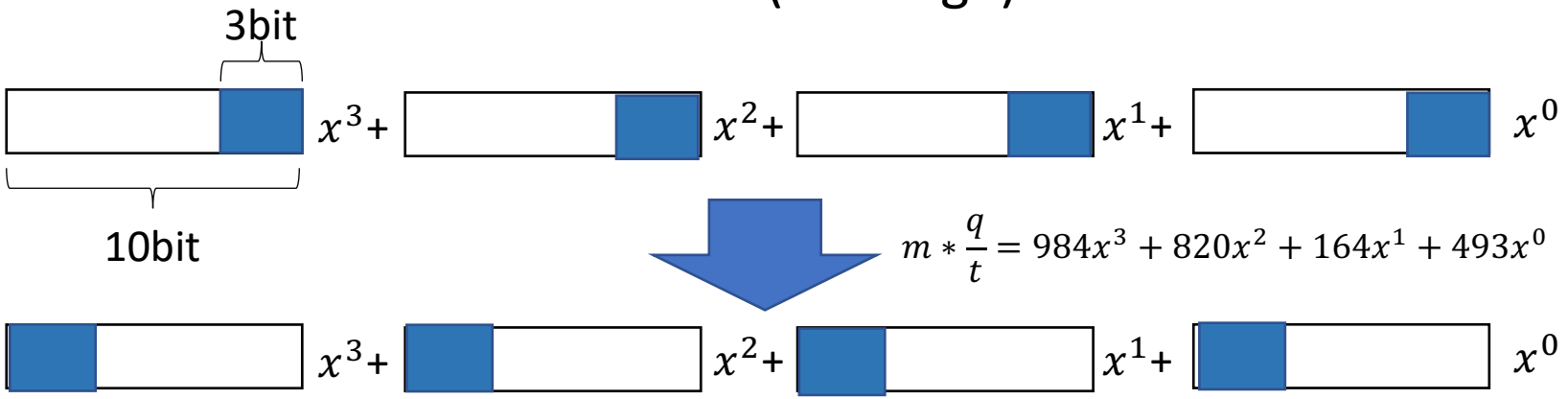
BFV Encrypt

- Encrypt $(m * \frac{q}{t} + pk[0],\ pk[1]) = (m * \frac{q}{t} + (e - a * s),\ a)$
  - $m = 6x^3 + 5x^2 + 1x^1 + 3x^0$ (message)



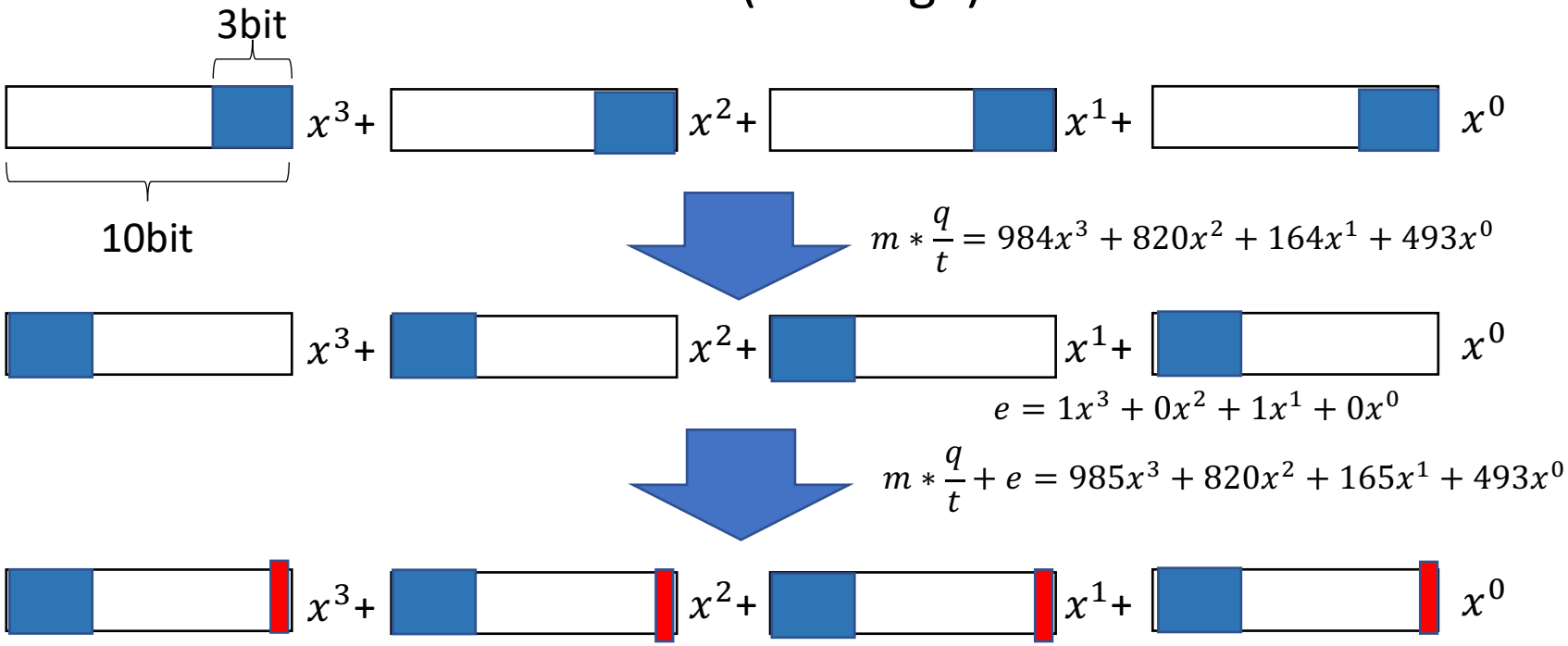
BFV Encrypt

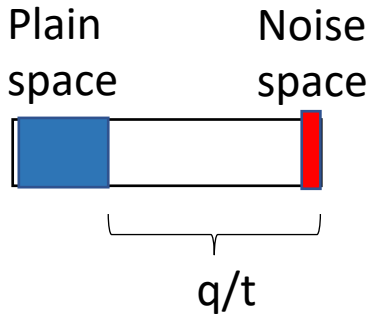
- Encrypt $(m * \frac{q}{t} + pk[0],\ pk[1]) = (m * \frac{q}{t} + (e - a * s),\ a)$
  - $m = 6x^3 + 5x^2 + 1x^1 + 3x^0$ (message)



BFV Encrypt

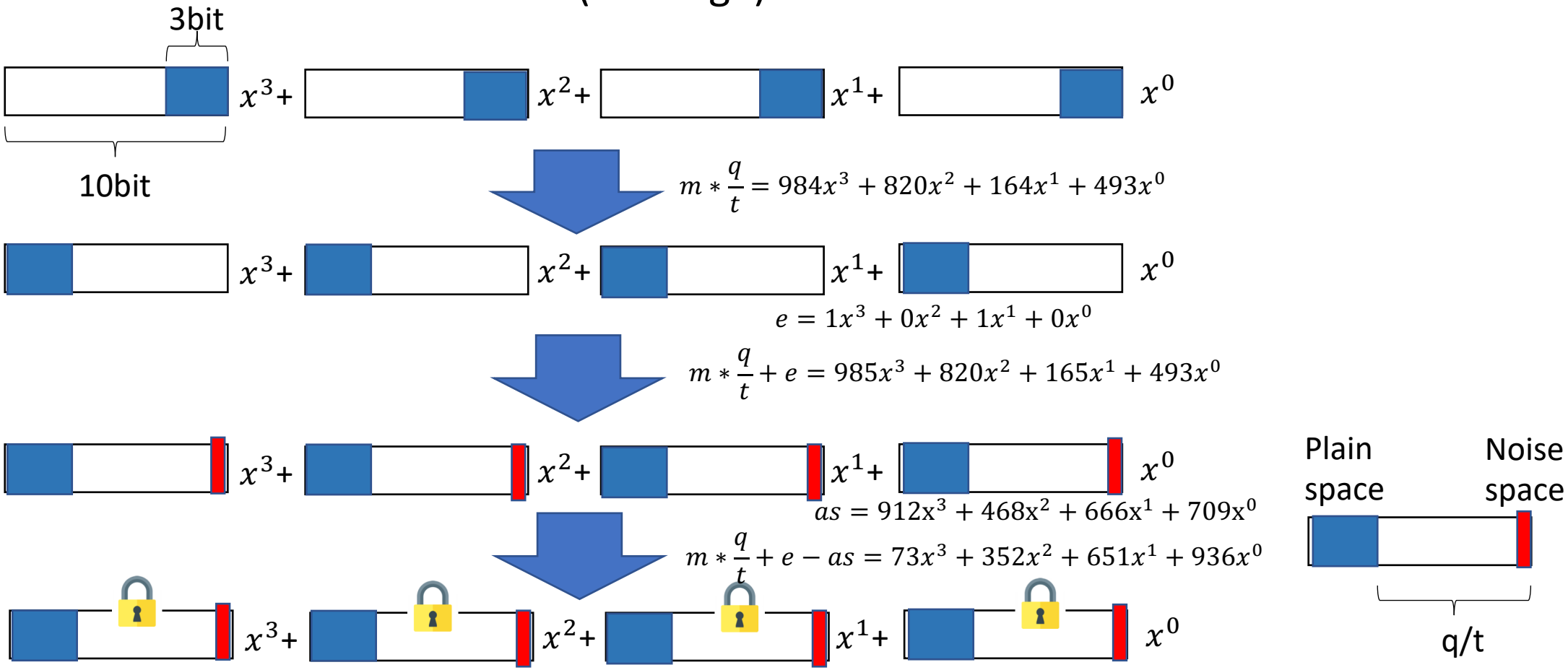
- Encrypt $(m * \frac{q}{t} + pk[0],\ pk[1]) = (m * \frac{q}{t} + (e - a * s),\ a)$
  - $m = 6x^3 + 5x^2 + 1x^1 + 3x^0$ (message)





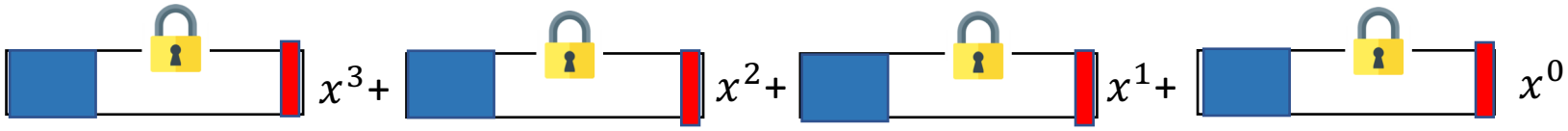
BFV Encrypt

- Encrypt $(m * \frac{q}{t} + pk[0],\ pk[1]) = (m * \frac{q}{t} + (e - a * s),\ a)$
  - $m = 6x^3 + 5x^2 + 1x^1 + 3x^0$ (message)



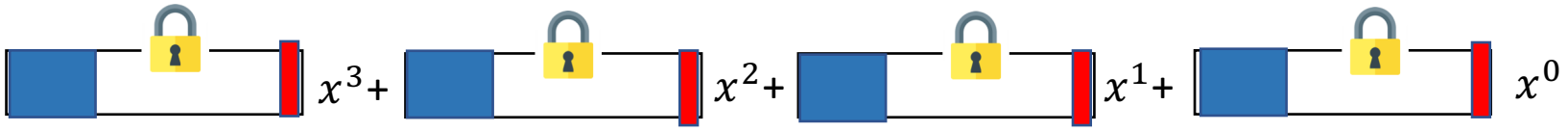
BFV Decrypt

$$m * \frac{q}{t} + e - as = 73x^3 + 352x^2 + 651x^1 + 936x^0$$



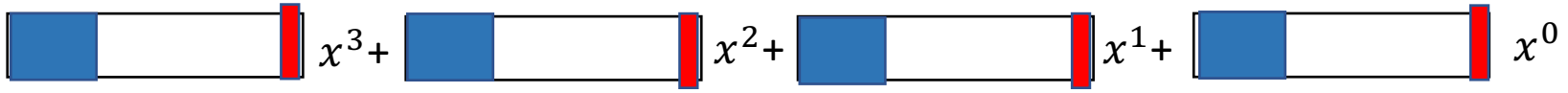
BFV Decrypt

$$m * \frac{q}{t} + e - as = 73x^3 + 352x^2 + 651x^1 + 936x^0$$



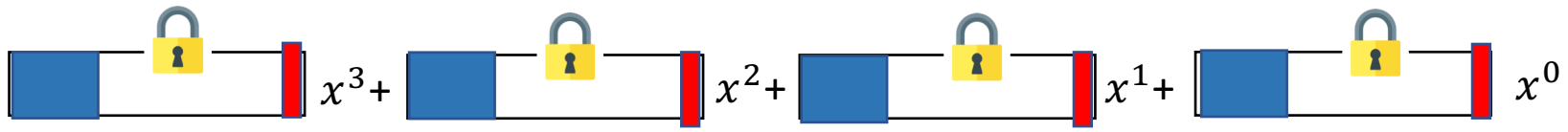
$$as = 912x^3 + 468x^2 + 666x^1 + 709x^0$$

$$m * \frac{q}{t} + e - a * s + a * s = 985x^3 + 820x^2 + 164x + 492$$



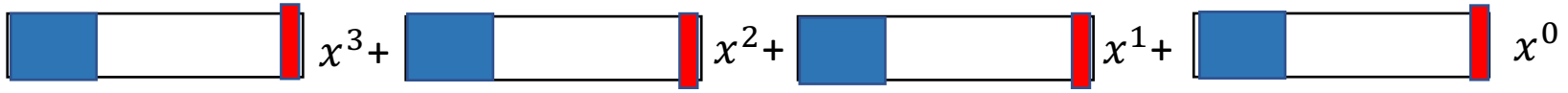
BFV Decrypt

$$m * \frac{q}{t} + e - as = 73x^3 + 352x^2 + 651x^1 + 936x^0$$



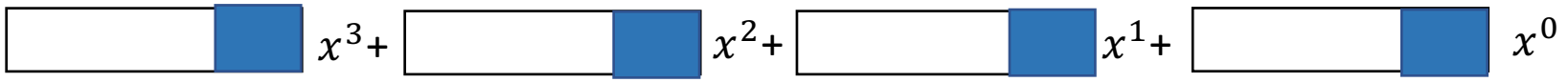
$$as = 912x^3 + 468x^2 + 666x^1 + 709x^0$$

$$m * \frac{q}{t} + e - a * s + a * s = 985x^3 + 820x^2 + 164x + 492$$



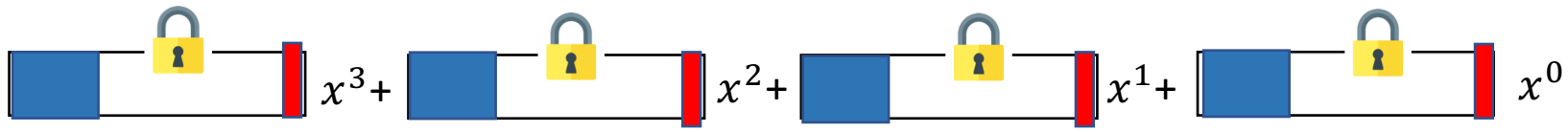
t/q=0.00607111882

$$\left(m * \frac{q}{t} + e\right) * \frac{t}{q} = 5.98005203769999x^3 + 4.9783174324x^2 + 0.9956634864799999x^1 + 2.98699045944$$



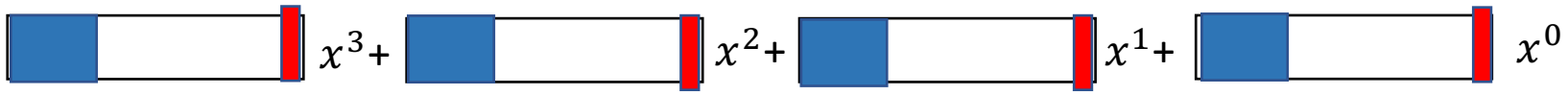
BFV Decrypt

$$m * \frac{q}{t} + e - as = 73x^3 + 352x^2 + 651x^1 + 936x^0$$



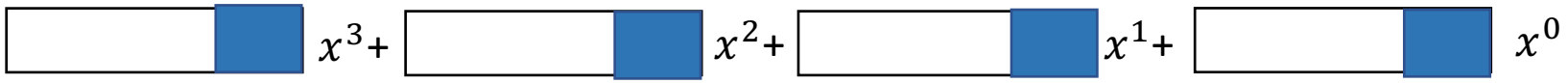
$$as = 912x^3 + 468x^2 + 666x^1 + 709x^0$$

$$m * \frac{q}{t} + e - a * s + a * s = 985x^3 + 820x^2 + 164x + 492$$



t/q=0.00607111882

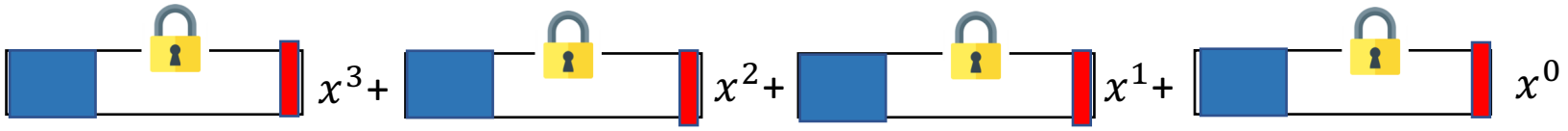
$$\left(m * \frac{q}{t} + e\right) * \frac{t}{q} = 5.98005203769999x^3 + 4.9783174324x^2 + 0.9956634864799999x^1 + 2.98699045944$$



After round: $6x^3 + 5x^2 + 1x + 3$

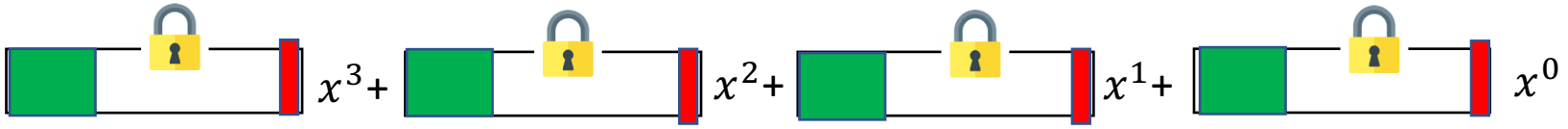
Decryption succeed if e < $\frac{q}{2t}$

Homomorphic addition

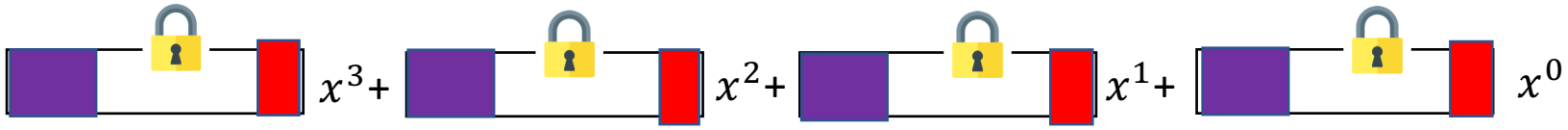


$Enc(m_1)$

+)



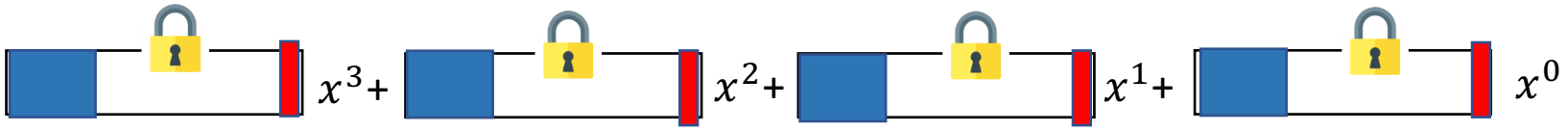
$Enc(m_2)$



$Enc(m_1 + m_2)$

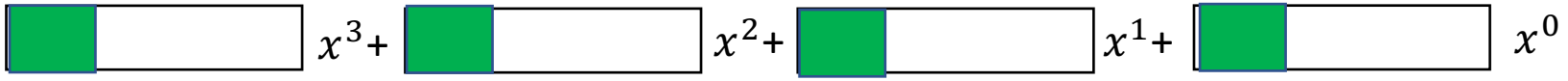
Decryption succeed if e < $\frac{q}{2t}$

Plaintext addition

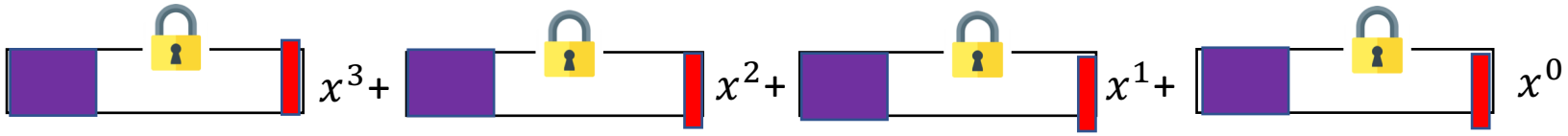


$Enc(m_1)$

+)



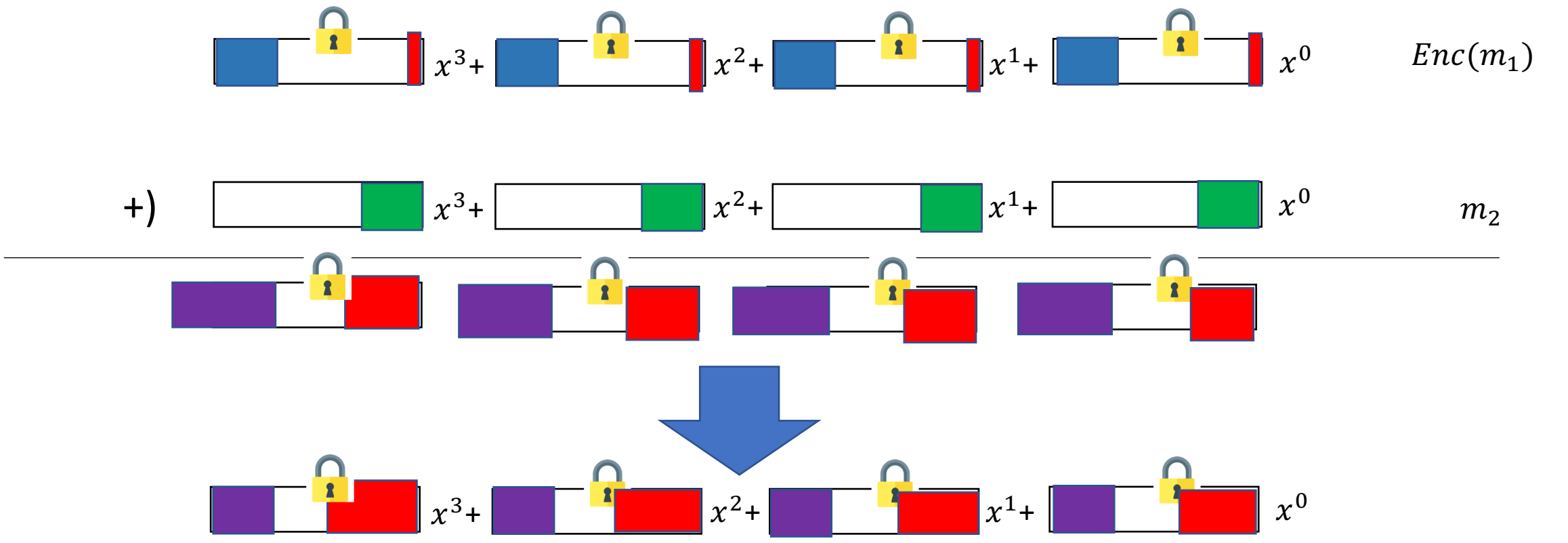
m_2



$Enc(m_1 + m_2)$

Decryption succeed if e < $\frac{q}{2t}$

Plaintext multiplication



Decryption succeed if e < $\frac{q}{2t}$

Homomorphic multiplication

- Skip, a little bit complicated
- **Consume lots of noise space**

Homomorphic multiplication

- Skip, a little bit complicated
- **Consume lots of noise space**

t = 1153 (10bits)
q = 142724769270595988105828596944949513638274700 9 (150bits)
n = 16

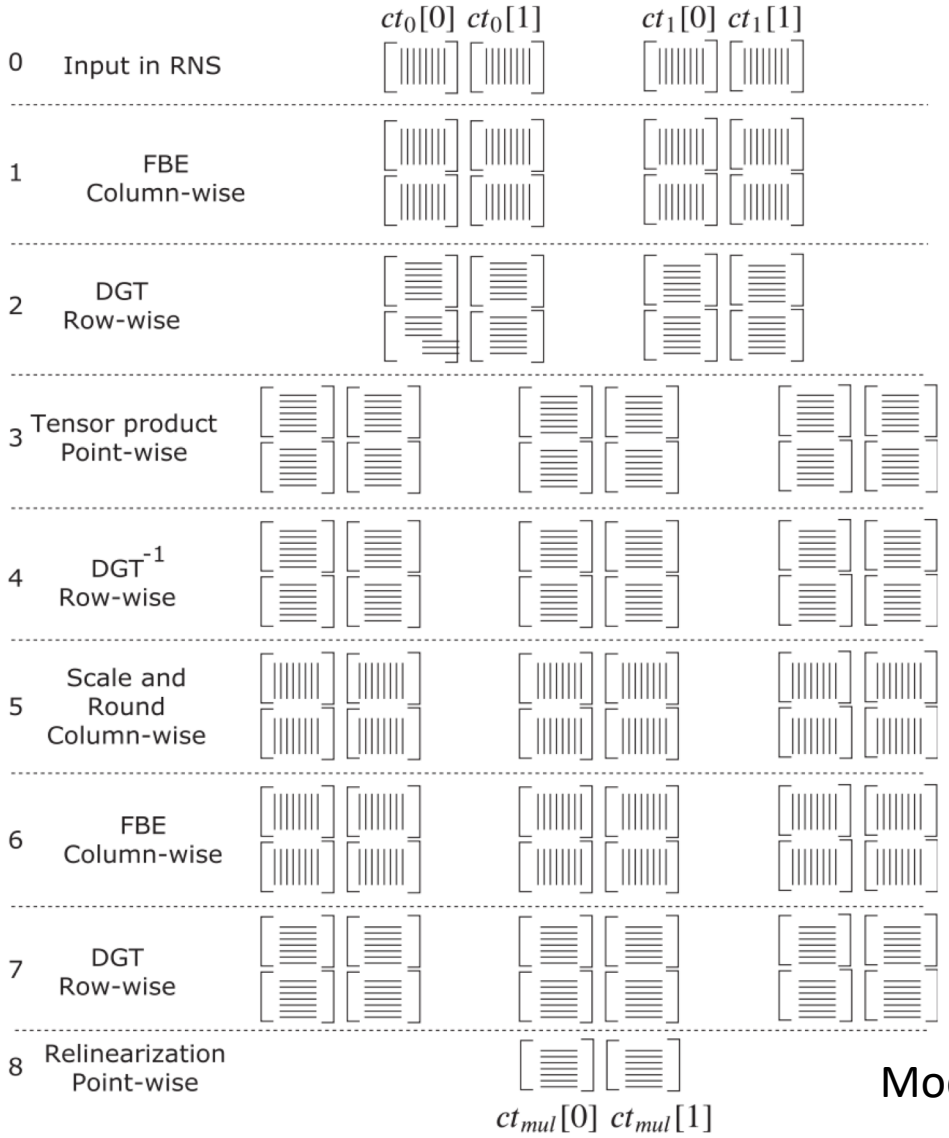
Before round, apply HMul several times:

```
583.999999999999999999999999999999999999987107703803595601743233956902
654.000000000000000000000000000000013873710056912614225047088038141725
190.999999999999999999999999999901221544276164803813128087381964752504
241.000000000000000000000064873163714006508923179433446959204615785157
4.999999999999999999817025477240480865675703938800554282174043461181871
982.999999999999998979452759689235522446449633833600466105954956067051 87
5.000000000023786969935097411863104026258438953760122481616862516745027 105
145.999997603700340162286581839958600445131043933944024583249229857815 8122
```

x^0 coefficient

Noise bits

Homomorphic multiplication



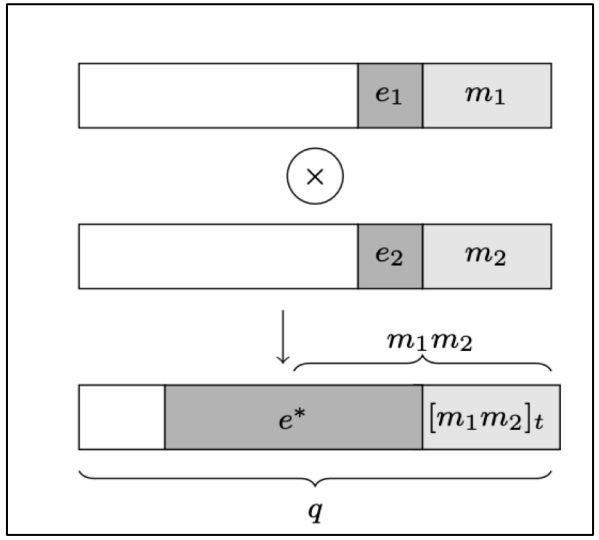
Normal NTT vs DGT (GPU-friendly)

Modulus raising vs bit-decomposition (FPGA-friendly)

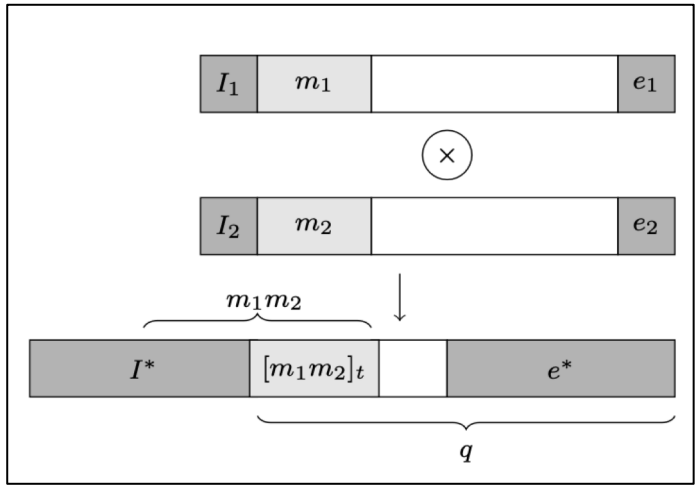
https://ieeexplore.ieee.org/document/9185077

Comparison between different schemes

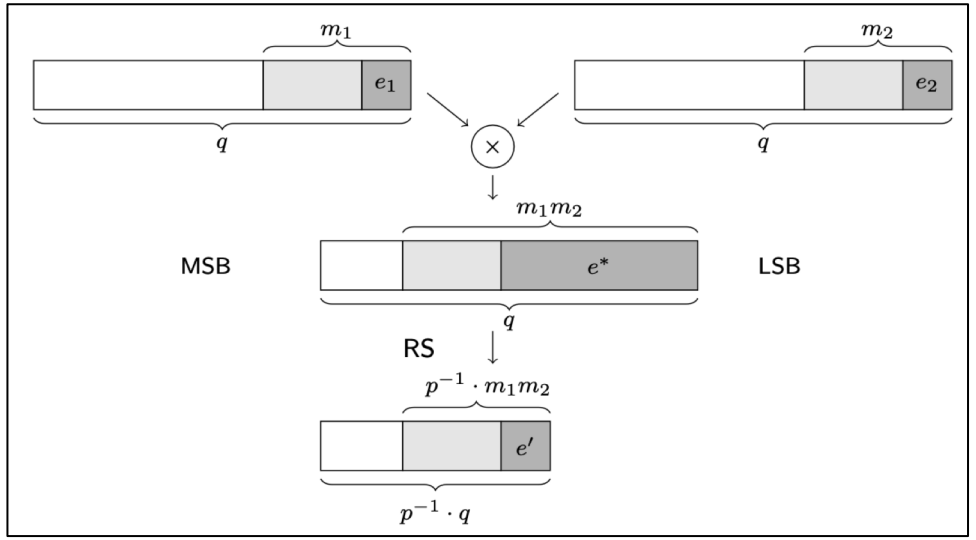
- Decryption structure



BGV, MSB is truncated



BFV, MSB is truncated (**suitable for integer operation,** can also perform fixed point operation using $m * 2^{\{precison\}}$)



CKKS, LSB is truncated (suitable for **approximate computation**)

https://eprint.iacr.org/2016/421.pdf

Encoding

$$\frac{Z_{17}}{x^4 + 1}$$

$$m_1 = 6x^3 + 5x^2 + 1x^1 + 3x^0$$
$$m_2 = 4x^3 + 5x^2 + 1x^1 + 2x^0$$
$$m_1 m_2 = 0x^3 + 2x^2 + 6x^1 + 5x^0$$

Not that useful

https://eprint.iacr.org/2018/244.pdf

Encoding

$$\frac{Z_{17}}{x^4 + 1}$$

$$m_1 = 6x^3 + 5x^2 + 1x^1 + 3x^0$$
$$m_2 = 4x^3 + 5x^2 + 1x^1 + 2x^0$$
$$m_1 m_2 = 0x^3 + 2x^2 + 6x^1 + 5x^0$$

Not that useful

[0, 1, 2, 3]

encoding

$2x^3 + 0x^2 + 5x^1 + 10x^0$

[1, 2, 3, 4]

encoding

$2x^3 + 0x^2 + 5x^1 + 11x^0$

Addition

$4x^3 + 0x^2 + 10x^1 + 4x^0$

decoding

[1, 3, 5, 7]

https://eprint.iacr.org/2018/244.pdf

Encoding

$$\frac{Z_{17}}{x^4+1}$$

Require $t \equiv 1 \pmod{2n}$

$$m_1 = 6x^3 + 5x^2 + 1x^1 + 3x^0$$
$$m_2 = 4x^3 + 5x^2 + 1x^1 + 2x^0$$
$$m_1 m_2 = 0x^3 + 2x^2 + 6x^1 + 5x^0$$

Not that useful

Require $t \equiv 1 \pmod{2n}$

Encode

$$13\begin{pmatrix} 9^{-1*0} & 9^{-3*0} & 9^{-7*0} & 9^{-5*0} \\ 9^{-1*1} & 9^{-3*1} & 9^{-7*1} & 9^{-5*1} \\ 9^{-1*2} & 9^{-3*2} & 9^{-7*2} & 9^{-5*2} \\ 9^{-1*3} & 9^{-3*3} & 9^{-7*3} & 9^{-5*3} \end{pmatrix}\begin{pmatrix} 0 \\ 1 \\ 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 10 \\ 5 \\ 0 \\ 2 \end{pmatrix}$$

Decode

$$\begin{pmatrix} 9^{1*0} & 9^{1*1} & 9^{1*2} & 9^{1*3} \\ 9^{3*0} & 9^{3*1} & 9^{3*2} & 9^{3*3} \\ 9^{7*0} & 9^{7*1} & 9^{7*2} & 9^{7*3} \\ 9^{5*0} & 9^{5*1} & 9^{5*2} & 9^{5*3} \end{pmatrix}\begin{pmatrix} 4 \\ 10 \\ 0 \\ 4 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \\ 5 \\ 7 \end{pmatrix}$$

[0, 1, 2, 3]

encoding

$$2x^3 + 0x^2 + 5x^1 + 10x^0$$

[1, 2, 3, 4]

encoding

$$2x^3 + 0x^2 + 5x^1 + 11x^0$$

Addition

$$4x^3 + 0x^2 + 10x^1 + 4x^0$$

decoding

[1, 3, 5, 7]

https://eprint.iacr.org/2018/244.pdf

Rotation

$\frac{Z_{17}}{x^8+1}$

[0, 1, 2, 3]
[4, 5, 6, 7]

Rotate column

[3, 0, 1, 2]
[7, 4, 5, 6]

$\frac{Z_{17}}{x^8+1}$

[0, 1, 2, 3]
[4, 5, 6, 7]

Rotate row

[4, 5, 6, 7]
[0, 1, 2, 3]

https://eprint.iacr.org/2012/099.pdf

Rotation

- Using a set of automorphism $R_n r$ of the form

$$\theta_t : R_{p^r} \longrightarrow R_{p^r}, \quad a(X) \longmapsto a(X^t) \pmod{(p^r, \Phi_m(X))}.$$

  - Example: $\frac{Z_{17}}{x^4+1}$
    - $2x^3 + 3x^2 + 1x^1 + 2$
    - → $2(x^3)^3+3(x^3)^2+1(x^3)^1+2$
    - →$2x^9 + 3x^6 + 1x^3 + 2$
    - →$2x^9 + 3x^6 + 1x^3 + 2$ $(x^4 = -1)$
    - →$2x + 17x^2 + 1x^3 + 2$ (Z_{17})
- Then perform key-switching from $s(X^t)$ to original s
- Require $2 * \log_2 n/2$ (either direction) galois keys in Microsoft SEAL

https://eprint.iacr.org/2012/099.pdf

Routines

$\frac{Z_{17}}{x^4+1}$

$\frac{Z_{1033}}{x^4+1}$

[0, 1, 2, 3]

encode

$2x^3 + 0x^2 + 5x^1 + 10x^0$

encrypt

$(948x^3 + 1022x^2 + 439x^1 + 456x^0, 489x^3 + 696x^2 + 55x^1 + 297x^0)$

[1, 2, 3, 4]

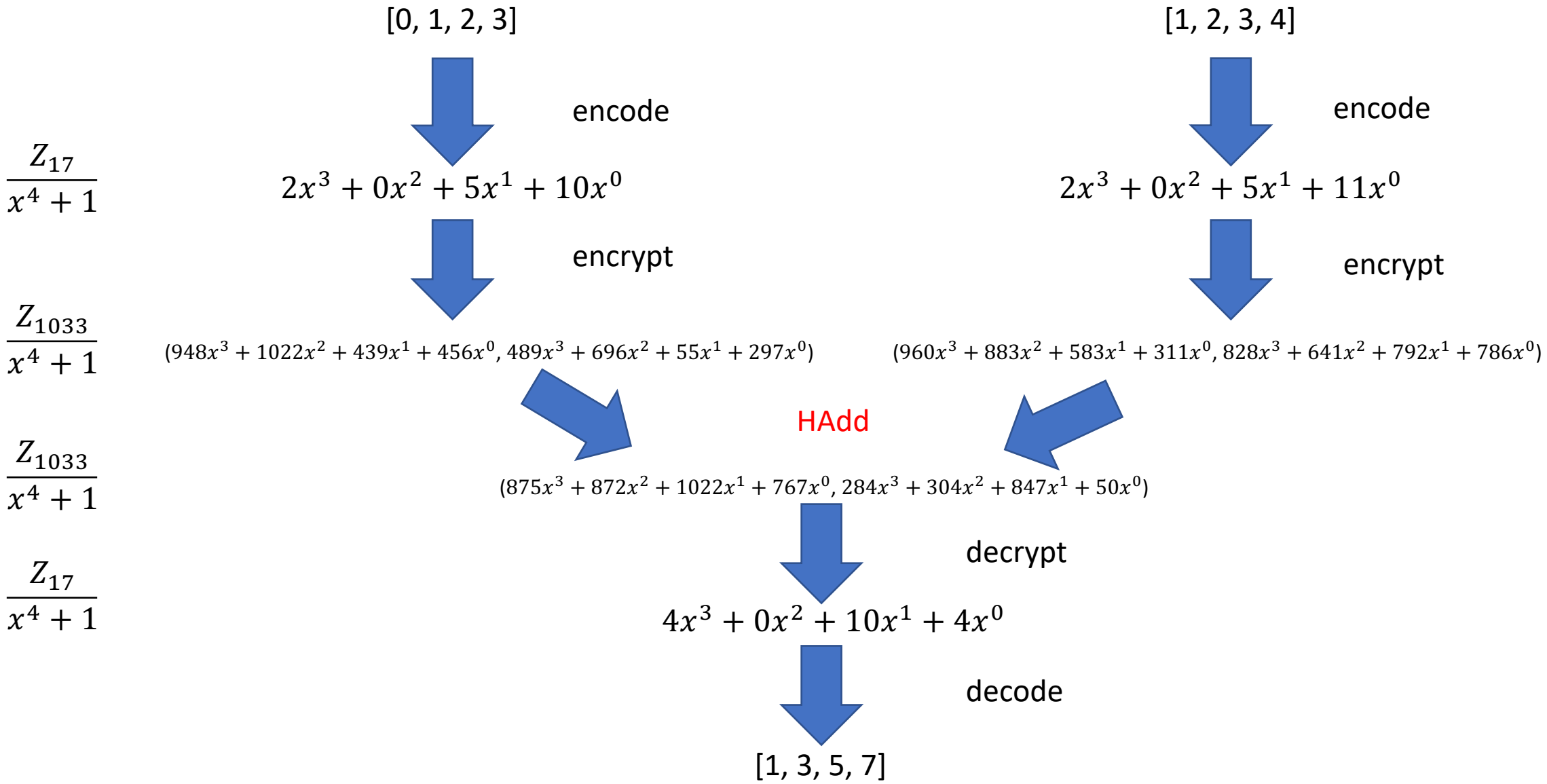
encode

$2x^3 + 0x^2 + 5x^1 + 11x^0$

encrypt

$(960x^3 + 883x^2 + 583x^1 + 311x^0, 828x^3 + 641x^2 + 792x^1 + 786x^0)$

Routines


[0, 1, 2, 3]
encode
$2x^3 + 0x^2 + 5x^1 + 10x^0$
encrypt
$(948x^3 + 1022x^2 + 439x^1 + 456x^0, 489x^3 + 696x^2 + 55x^1 + 297x^0)$
[1, 2, 3, 4]
encode
$2x^3 + 0x^2 + 5x^1 + 11x^0$
encrypt
$(960x^3 + 883x^2 + 583x^1 + 311x^0, 828x^3 + 641x^2 + 792x^1 + 786x^0)$
HAdd
$(875x^3 + 872x^2 + 1022x^1 + 767x^0, 284x^3 + 304x^2 + 847x^1 + 50x^0)$
decrypt
$4x^3 + 0x^2 + 10x^1 + 4x^0$
decode
[1, 3, 5, 7]
$\frac{Z_{17}}{x^4+1}$
$\frac{Z_{1033}}{x^4+1}$
$\frac{Z_{1033}}{x^4+1}$
$\frac{Z_{17}}{x^4+1}$

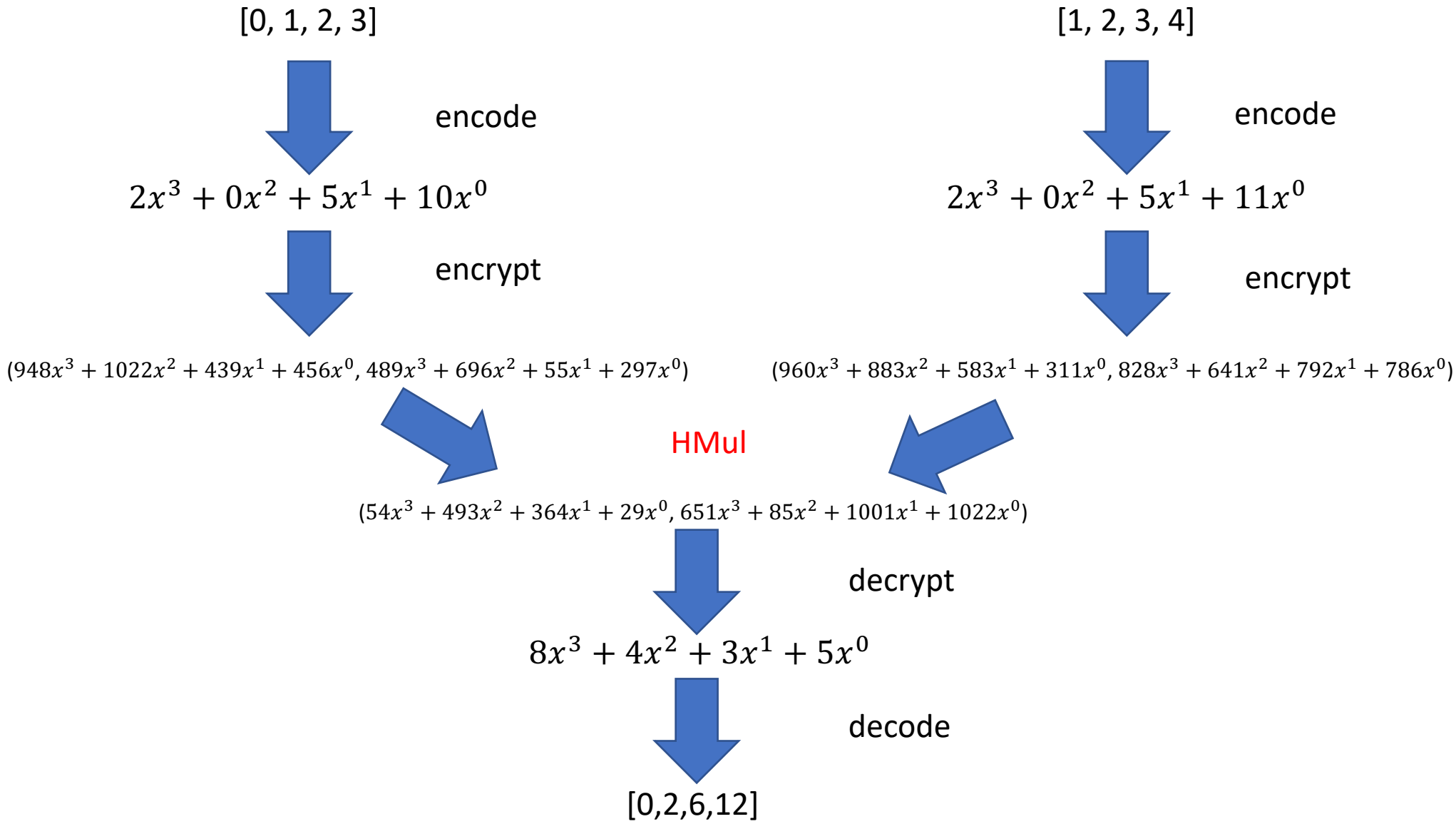
Routines

$\frac{Z_{17}}{x^4+1}$

$\frac{Z_{1033}}{x^4+1}$

$\frac{Z_{1033}}{x^4+1}$

$\frac{Z_{17}}{x^4+1}$



Routines

[157, 1014, 88, 516] [105, 86, 707, 13]

[0, 1, 2, 3]

encode

$2x^3 + 0x^2 + 5x^1 + 10x^0$

encrypt

$\frac{Z_{1033}}{x^4+1}$

$(948x^3 + 1022x^2 + 439x^1 + 456x^0, 489x^3 + 696x^2 + 55x^1 + 297x^0)$

Rotate

$\frac{Z_{1033}}{x^4+1}$

$(516x^3 + 88x^2 + 1014x^1 + 157x^0, 13x^3 + 707x^2 + 86x^1 + 105x^0)$

decrypt

$\frac{Z_{17}}{x^4+1}$

$5x^3 + 0x^2 + 2x^1 + 10x^0$

decode

[1, 0, 3, 2]

$\frac{Z_{17}}{x^4+1}$

[1, 2, 3, 4]

encode

$2x^3 + 0x^2 + 5x^1 + 11x^0$

encrypt

$(960x^3 + 883x^2 + 583x^1 + 311x^0, 828x^3 + 641x^2 + 792x^1 + 786x^0)$

Summary

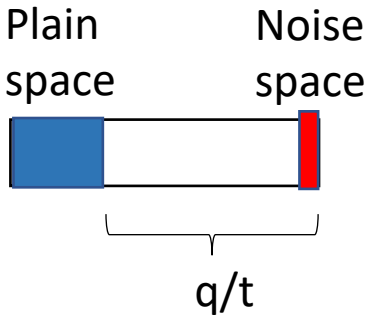
- Three main schemes
  - BGV
  - BFV (integer arithmetic)
  - CKKS (suitable for approximation arithmetic)
- Function call
  - **Encode, Decode**
  - **Encrypt, Decrypt**
  - Input a, b should be encrypted:
    - **H_Add, H_Mul, H_Rotate**
  - One input a, b can be plaintext
    - **PlainMul, PlainAdd**

Parameter selections

- t
  - Select first so that all plaintext operation is in t
  - Example: t = 20bits, a=5bits, can perform up to a^4

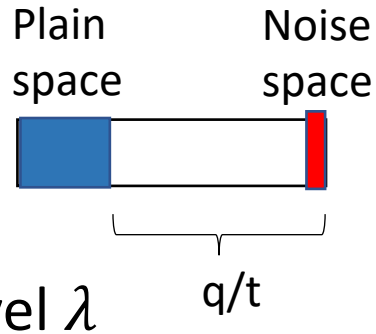
Parameter selections

- t
  - Select first so that all plaintext operation is in t
  - Example: t = 20bits, a=5bits, can perform up to a^4
- q
  - q/t should be large enough to ensure success of decryption



Parameter selections

- t
  - Select first so that all plaintext operation is in t
  - Example: t = 20bits, a=5bits, can perform up to a^4
- q
  - q/t should be large enough to ensure success of decryption
- n
  - n/log(q) should be large enough to ensure enough security level λ



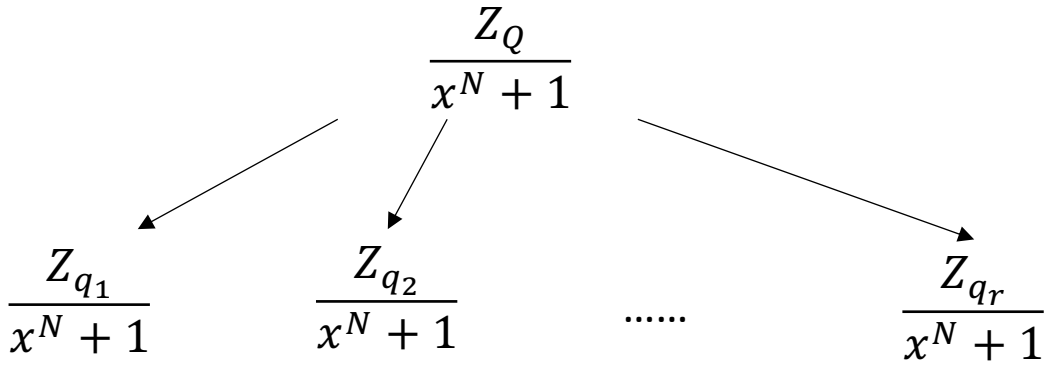
$N \geq \frac{\lambda+110}{7.2} log(Q)$ to get λ-bit security level

n	Bit-length of default q		
	128-bit security	192-bit security	256-bit security
1024	27	19	14
2048	54	37	29
4096	109	75	58
8192	218	152	118
16384	438	300	237
32768	881	600	476

https://www.microsoft.com/en-us/research/uploads/prod/2017/11/sealmanual-2-3-1.pdf

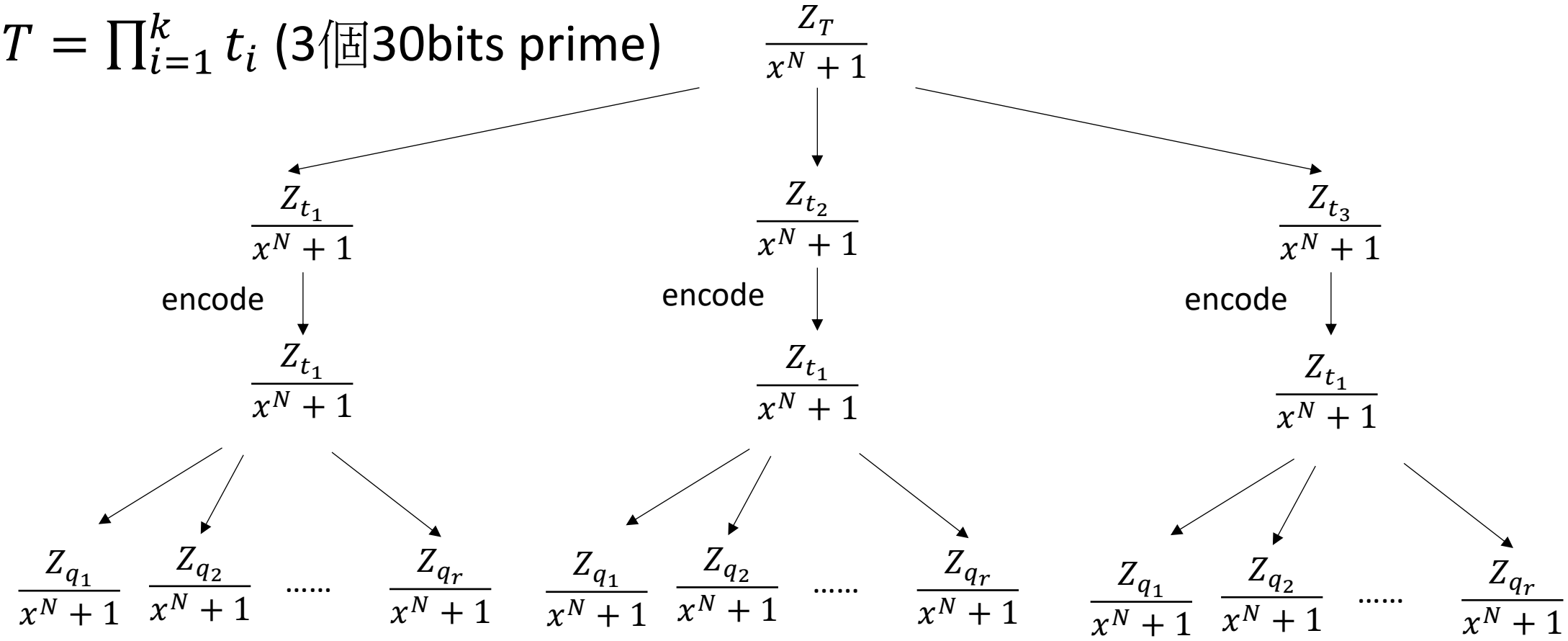
RNS (CRT)

- q may be huge (i.e. 300bits), which cannot fit in a machine word
- $Q = \prod_{i=1}^{r} q_i$ (10個30bits prime)
  - Operation in Q, equals operation in each q_i
  - More efficient, parallelizable



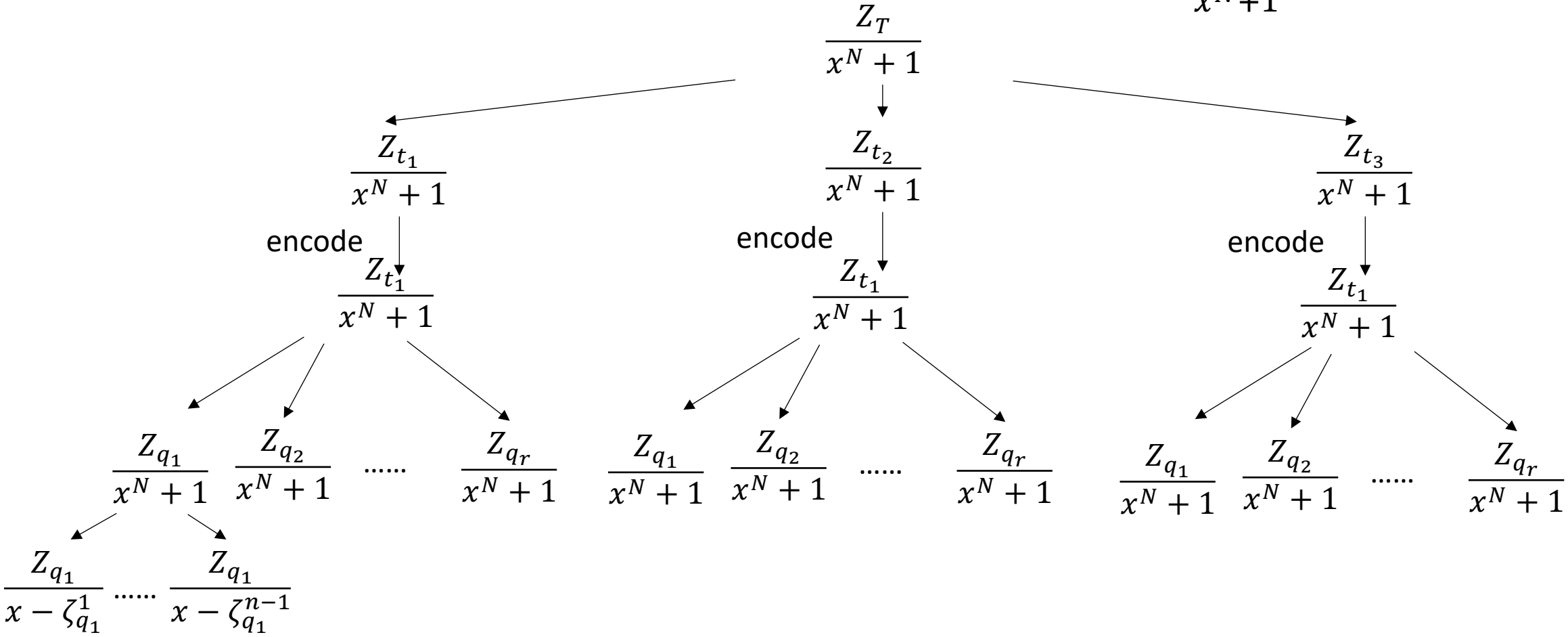
Double CRT and RNS

- t may also be huge (i.e. 90 bits)
- $T = \prod_{i=1}^{k} t_i$ (3個30bits prime)



Even further beyond

- Faster polynomial multiplication
  - Polynomial multiplication can be done in O(N logN) under $\frac{Z_{q_1}}{x^N+1}$ using NTT/FFT



Time, space, noise and security

Table 1. The asymptotic costs of homomorphic operations for the CKKS and RNS-CKKS scheme variants in HEAAN v1.0 and SEAL v3.1 respectively. $M(Q)$ is the complexity of multiplying large integers and is $O(\log^{1.58} Q)$ for HEAAN.

Homomorphic Operation	CKKS	RNS-CKKS with $Q = \Pi_{i=1}^{r} Q_i$
addition, subtraction	$O(N \cdot \log Q)$	$O(N \cdot r)$
scalar multiplication	$O(N \cdot M(Q))$	$O(N \cdot r)$
plaintext multiplication	$O(N \cdot \log N \cdot M(Q))$	$O(N \cdot r)$
ciphertext multiplication	$O(N \cdot \log N \cdot M(Q))$	$O(N \cdot \log N \cdot r^2)$
ciphertext rotation	$O(N \cdot \log N \cdot M(Q))$	$O(N \cdot \log N \cdot r^2)$

https://www.cs.utexas.edu/~roshan/CHET.pdf

Time, space, noise and security

Table 1. The asymptotic costs of homomorphic operations for the CKKS and RNS-CKKS scheme variants in HEAAN v1.0 and SEAL v3.1 respectively. $M(Q)$ is the complexity of multiplying large integers and is $O(\log^{1.58} Q)$ for HEAAN.

Homomorphic Operation	CKKS	RNS-CKKS with $Q = \Pi_{i=1}^{r} Q_i$
addition, subtraction	$O(N \cdot \log Q)$	$O(N \cdot r)$
scalar multiplication	$O(N \cdot M(Q))$	$O(N \cdot r)$
plaintext multiplication	$O(N \cdot \log N \cdot M(Q))$	$O(N \cdot r)$
ciphertext multiplication	$O(N \cdot \log N \cdot M(Q))$	$O(N \cdot \log N \cdot r^2)$
ciphertext rotation	$O(N \cdot \log N \cdot M(Q))$	$O(N \cdot \log N \cdot r^2)$

Memory usage (bits) (RNS)

Private key:	2*N
Public key:	$2*\log_2 Q$*N
Relinearization key:	$r * \log_2 Q$*N
Rotation key:	$r * \log_2 Q$*N*(#rot)

https://www.cs.utexas.edu/~roshan/CHET.pdf

Time, space, noise and security

Table 1. The asymptotic costs of homomorphic operations for the CKKS and RNS-CKKS scheme variants in HEAAN v1.0 and SEAL v3.1 respectively. $M(Q)$ is the complexity of multiplying large integers and is $O(\log^{1.58} Q)$ for HEAAN.

Homomorphic Operation	CKKS	RNS-CKKS with $Q = \Pi_{i=1}^{r} Q_i$
addition, subtraction	$O(N \cdot \log Q)$	$O(N \cdot r)$
scalar multiplication	$O(N \cdot M(Q))$	$O(N \cdot r)$
plaintext multiplication	$O(N \cdot \log N \cdot M(Q))$	$O(N \cdot r)$
ciphertext multiplication	$O(N \cdot \log N \cdot M(Q))$	$O(N \cdot \log N \cdot r^2)$
ciphertext rotation	$O(N \cdot \log N \cdot M(Q))$	$O(N \cdot \log N \cdot r^2)$

Operation	Time Cost	Noise Cost
Constant Addition	cheap	cheap
Addition	cheap	cheap
Constant Mult.	cheap	moderate[2]
Multiplication	expensive	expensive
Rotation	expensive	cheap

Memory usage (bits) (RNS)

Private key: 2*N

Public key: $2*\log_2 Q*N$

Relinearization key: $r * \log_2 Q*N$

Rotation key: $r * \log_2 Q*N*(\#rot)$

https://www.cs.utexas.edu/~roshan/CHET.pdf

Time, space, noise and security

Table 1. The asymptotic costs of homomorphic operations for the CKKS and RNS-CKKS scheme variants in HEAAN v1.0 and SEAL v3.1 respectively. $M(Q)$ is the complexity of multiplying large integers and is $O(\log^{1.58} Q)$ for HEAAN.

Homomorphic Operation	CKKS	RNS-CKKS with $Q = \Pi_{i=1}^{r} Q_i$
addition, subtraction	$O(N \cdot \log Q)$	$O(N \cdot r)$
scalar multiplication	$O(N \cdot M(Q))$	$O(N \cdot r)$
plaintext multiplication	$O(N \cdot \log N \cdot M(Q))$	$O(N \cdot r)$
ciphertext multiplication	$O(N \cdot \log N \cdot M(Q))$	$O(N \cdot \log N \cdot r^2)$
ciphertext rotation	$O(N \cdot \log N \cdot M(Q))$	$O(N \cdot \log N \cdot r^2)$

Memory usage (bits) (RNS)

Private key: 2*N
Public key: $2*\log_2 Q*N$
Relinearization key: $r * \log_2 Q*N$
Rotation key: $r * \log_2 Q*N*(\#rot)$

Operation	Time Cost	Noise Cost
Constant Addition	cheap	cheap
Addition	cheap	cheap
Constant Mult.	cheap	moderate[2]
Multiplication	expensive	expensive
Rotation	expensive	cheap

n	Bit-length of default q		
	128-bit security	192-bit security	256-bit security
1024	27	19	14
2048	54	37	29
4096	109	75	58
8192	218	152	118
16384	438	300	237
32768	881	600	476

https://www.cs.utexas.edu/~roshan/CHET.pdf

Outline

- Introduction
- Possible scenarios
- More specific
  - BFV encryption
  - Encrypted inference (LoLa)
- Research directions

Comparison of existing encrypted inference

Table 2: MNIST performance comparison. Solutions are grouped by accuracy levels.

Method	Accuracy	Latency	
FHE–DiNN100	96.35%	1.65	(Bourse et al., 2017)
LoLa-Small	96.92%	**0.29**	
CryptoNets	98.95%	205	(Dowlin et al., 2016)
nGraph-HE	98.95%[9]	135	(Boemer et al., 2018)
Faster-CryptoNets	98.7%	39.1	(Chou et al., 2018)
CryptoNets 2.3	98.95	24.8	
HCNN	99%	14.1	(Badawi et al., 2018)
LoLa-Dense	98.95%	7.2	
LoLa	98.95%	**2.2**	

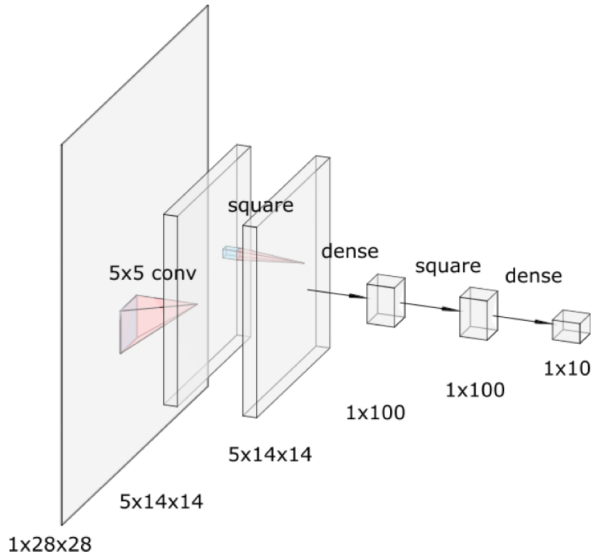


Figure 3: The structure of the network used for MNIST classification.

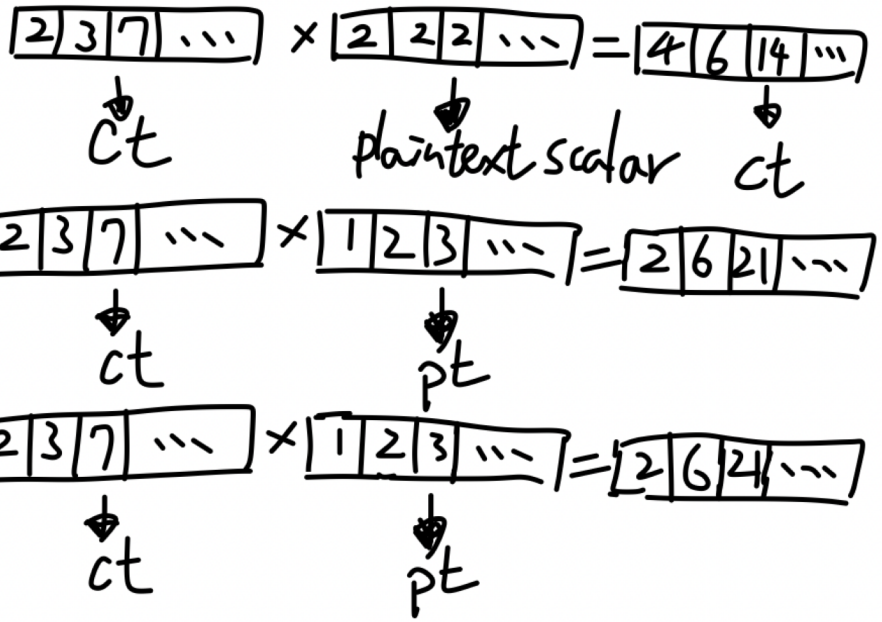
85625個double

Supported operations

Ciphertext × Scalar = Ciphertext

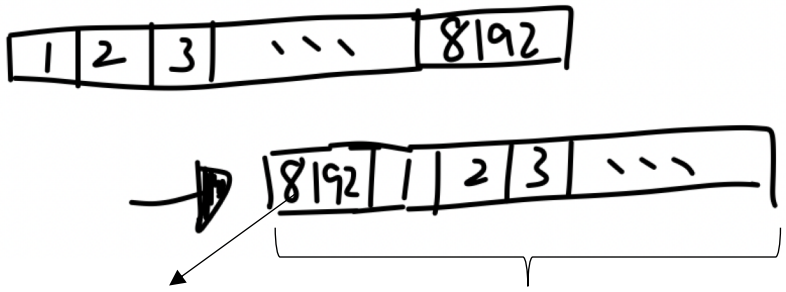
Ciphertext × Plaintext = Ciphertext

Ciphertext × Ciphertext = Ciphertext



加法一樣 (elementwise)

Rotate



Scenario

- Inference MNIST (28x28)

Here,
N=8192, k=4, $\lambda = 128$
Q={43,43,44,44,44} bits, r=5
$\log_2 Q \approx 218$
#rot ≈ 20

Memory usage (bits) (RNS)		
Private key:	2*N	2KB
Public key:	$2*\log_2 Q*N$	420KB
Relinearization key:	$r * \log_2 Q*N$	1MB
Rotation key:	$r * \log_2 Q*N*(\#rot)$	20MB
Encrypted 28x28 image:	$k*25*2*\log_2 Q*N$	42MB
Result returned:	$2*\log_2 Q*N$	420KB

Scenario

- Inference MNIST (28x28)

Here,
N=8192, k=4, $\lambda = 128$
Q={43,43,44,44,44} bits, r=5
$\log_2 Q \approx 218$
#rot ≈ 20

Memory usage (bits) (RNS)		
Private key:	2*N	2KB
Public key:	$2*\log_2 Q*N$	420KB
Relinearization key:	$r * \log_2 Q*N$	1MB
Rotation key:	$r * \log_2 Q*N*(\#rot)$	20MB
Encrypted 28x28 image:	$k*25*2*\log_2 Q*N$	42MB
Result returned:	$2*\log_2 Q*N$	420KB

420KB+1MB+20MB+42MB
≈ 64MB

Cf:
RSA-2048, public key 2048bits
Message 245B→256B

Neural network

- Cannot perform ReLU or sigmoid straightforward
  - Approximation using taylor expansion
  - Using HE friendly activation function
    - Ex: x^2

Neural network

- Cannot perform ReLU or sigmoid straightforward
  - Approximation using taylor expansion
  - Using HE friendly activation function
    - Ex: x^2

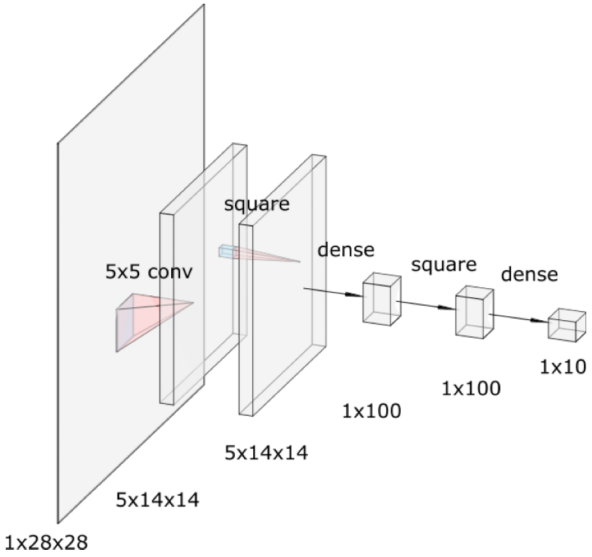
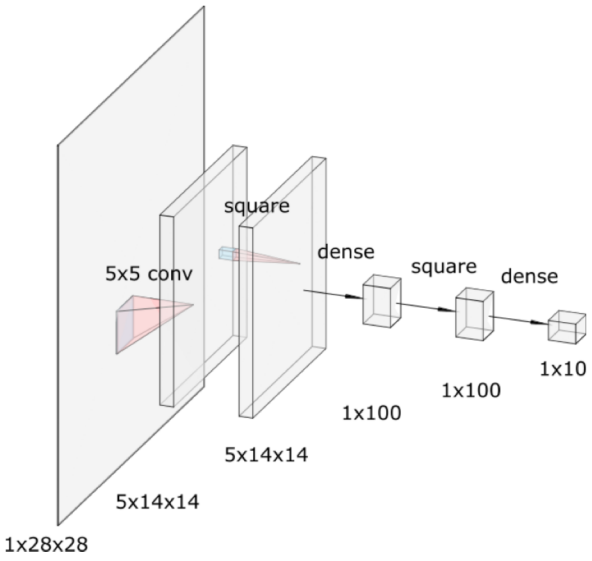


Figure 3: The structure of the network used for MNIST classification.

Neural network

- Cannot perform ReLU or sigmoid straightforward
  - Approximation using taylor expansion
  - Using HE friendly activation function
    - Ex: x^2



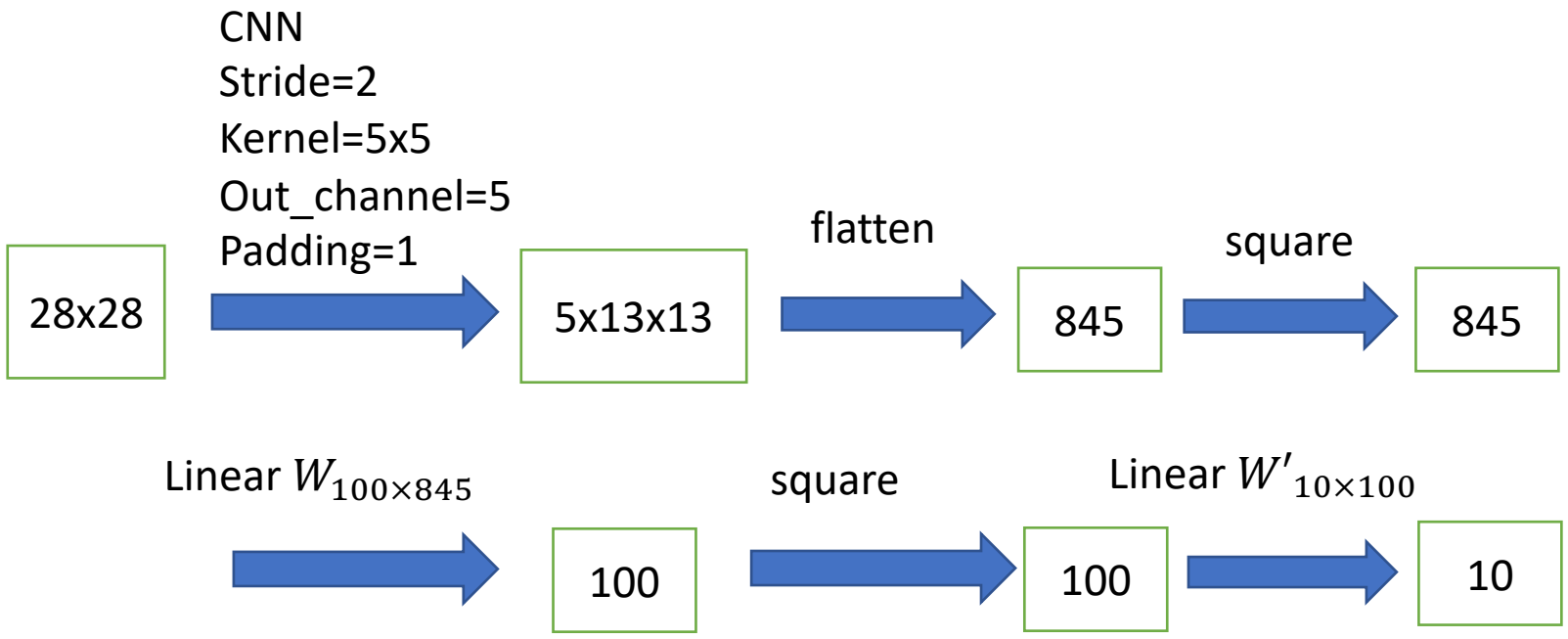


Figure 3: The structure of the network used for MNIST classification.

Training phase

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 5, 5, 2, padding=1)
        self.fc1 = nn.Linear(5*169, 100)
        self.fc2 = nn.Linear(100, 10)

    def forward(self, x):
        x = self.conv1(x)
        x = x**2
        x = torch.flatten(x, 1)
        x = self.fc1(x)
        x = x**2
        x = self.fc2(x)
        output = F.log_softmax(x, dim=1)
        return output
```

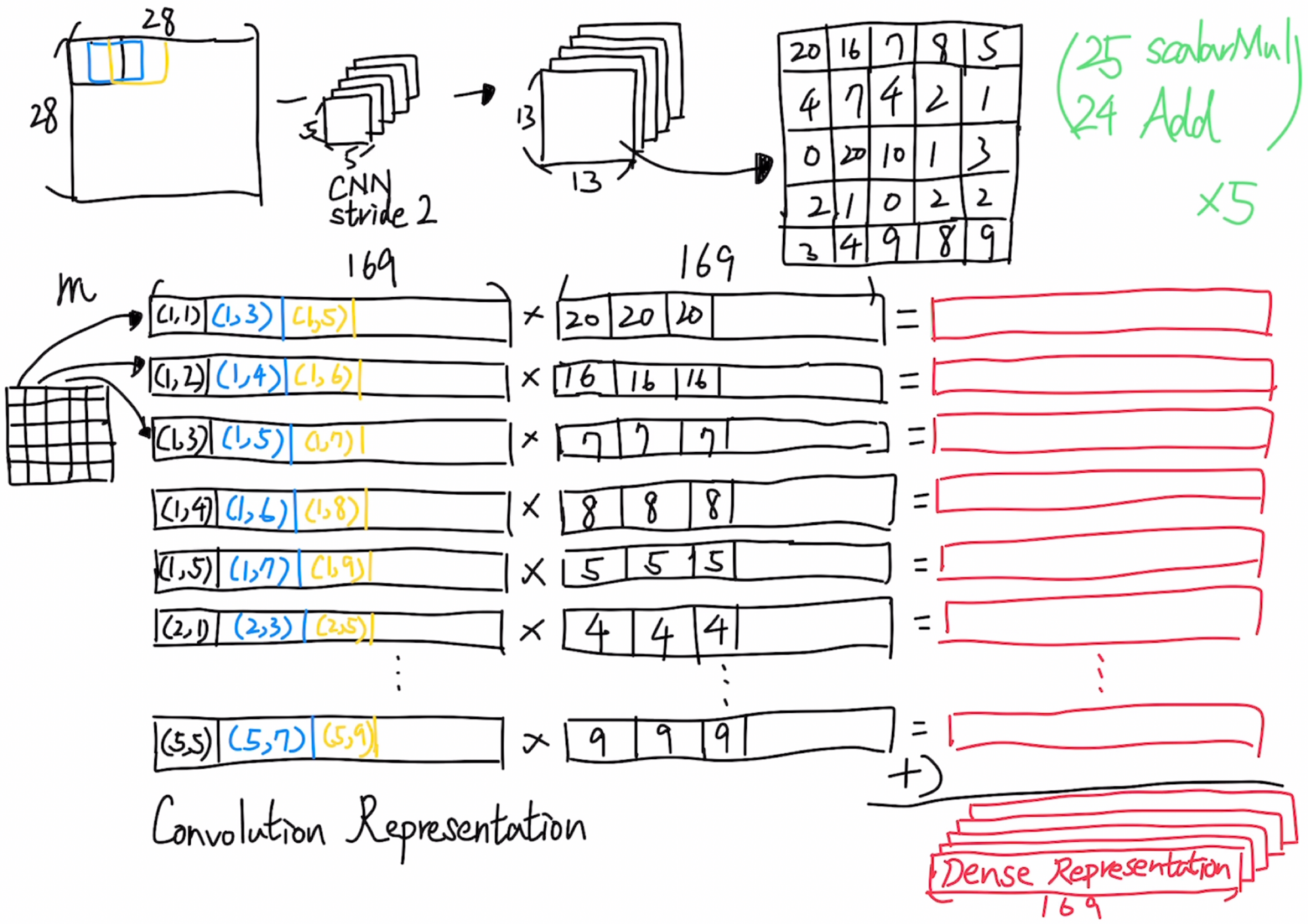
```
Test set: Average loss: 0.0837, Accuracy: 9874/10000 (99%)
```

Inference

Here,
N=8192, k=4, $\lambda = 128$
Q={43,43,44,44,44} bits, r=5
$\log_2 Q \approx 218$
#rot ≈ 20

- Take the model parameters out
- Using fixed point to represent model coefficients and inputs
- Divide the result with scale afterwards

CNN



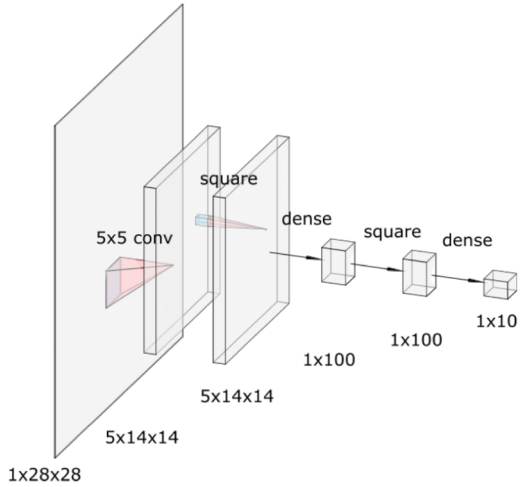


Figure 3: The structure of the network used for MNIST classification.

Flatten

4 Rotation
4 Add

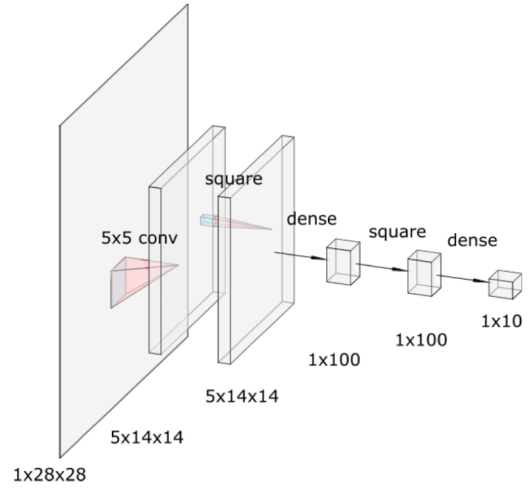
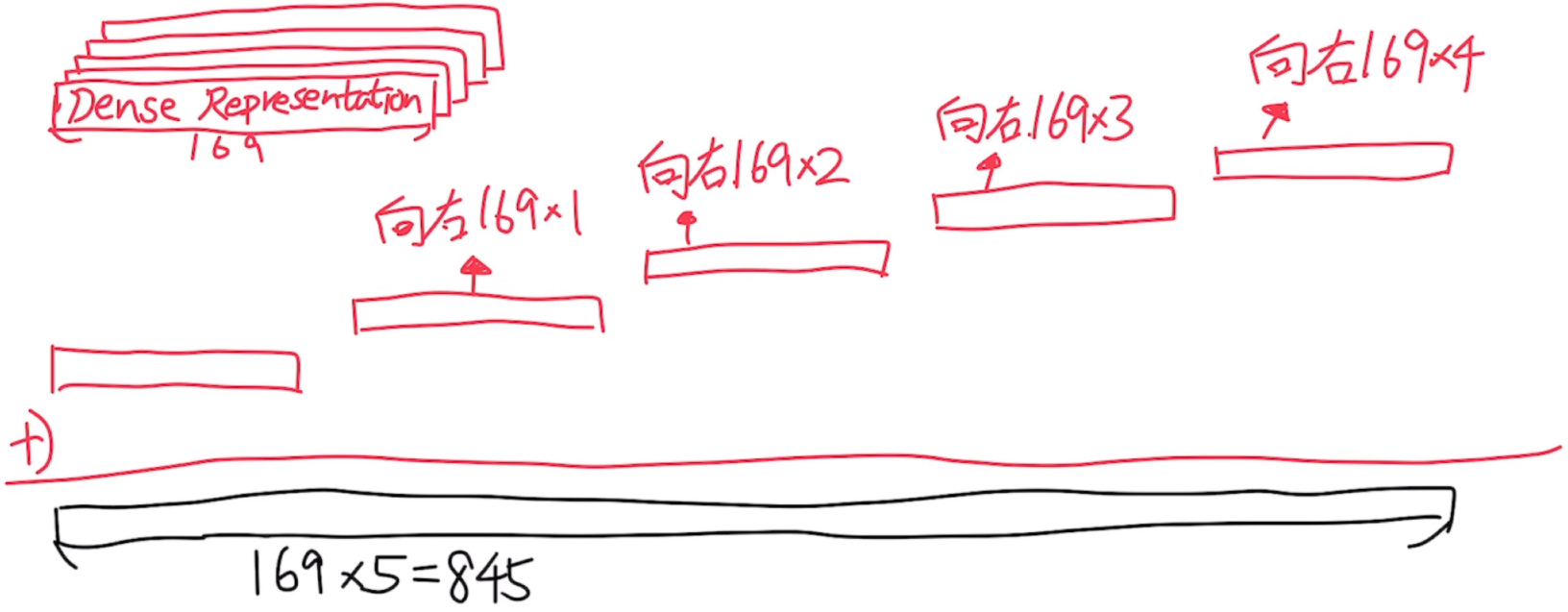
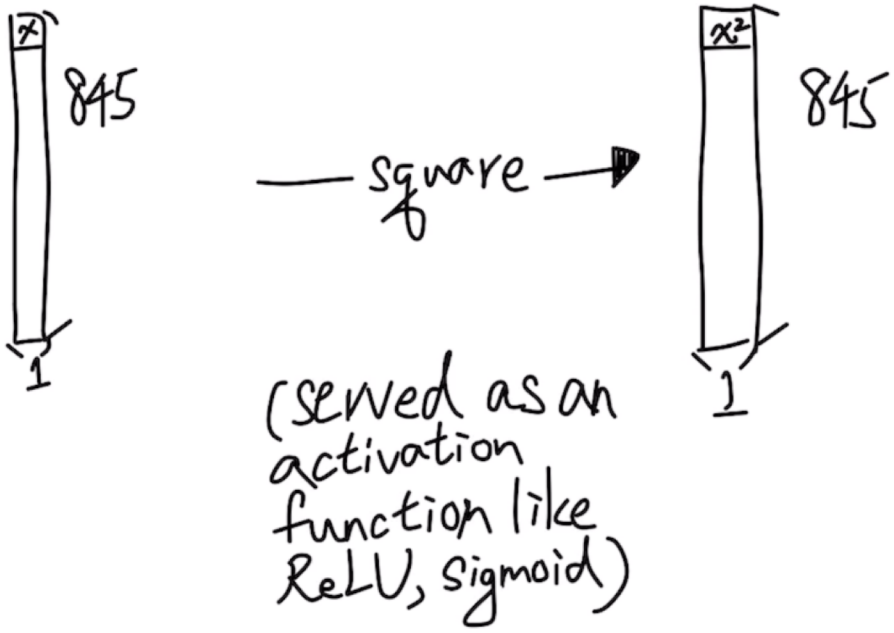


Figure 3: The structure of the network used for MNIST classification.



Square



1 cipherMul

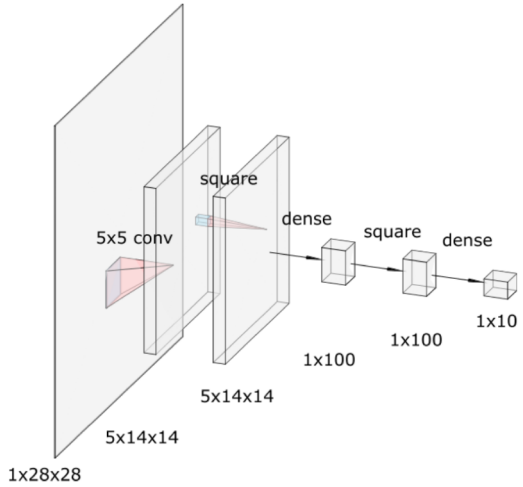
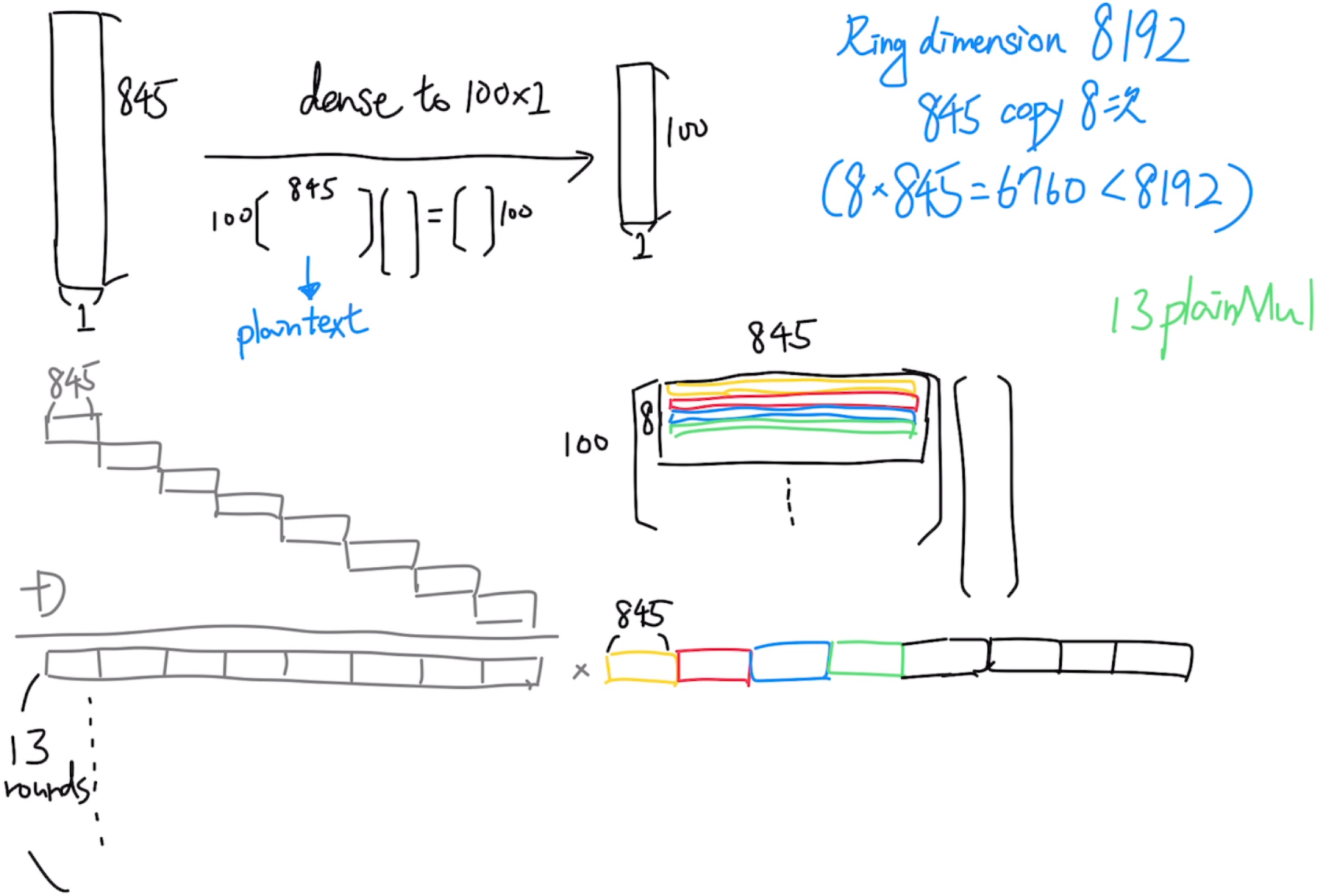


Figure 3: The structure of the network used for MNIST classification.

Dense layer 845 → 100



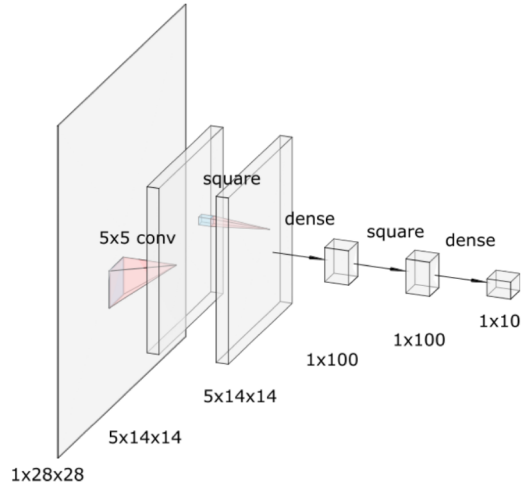
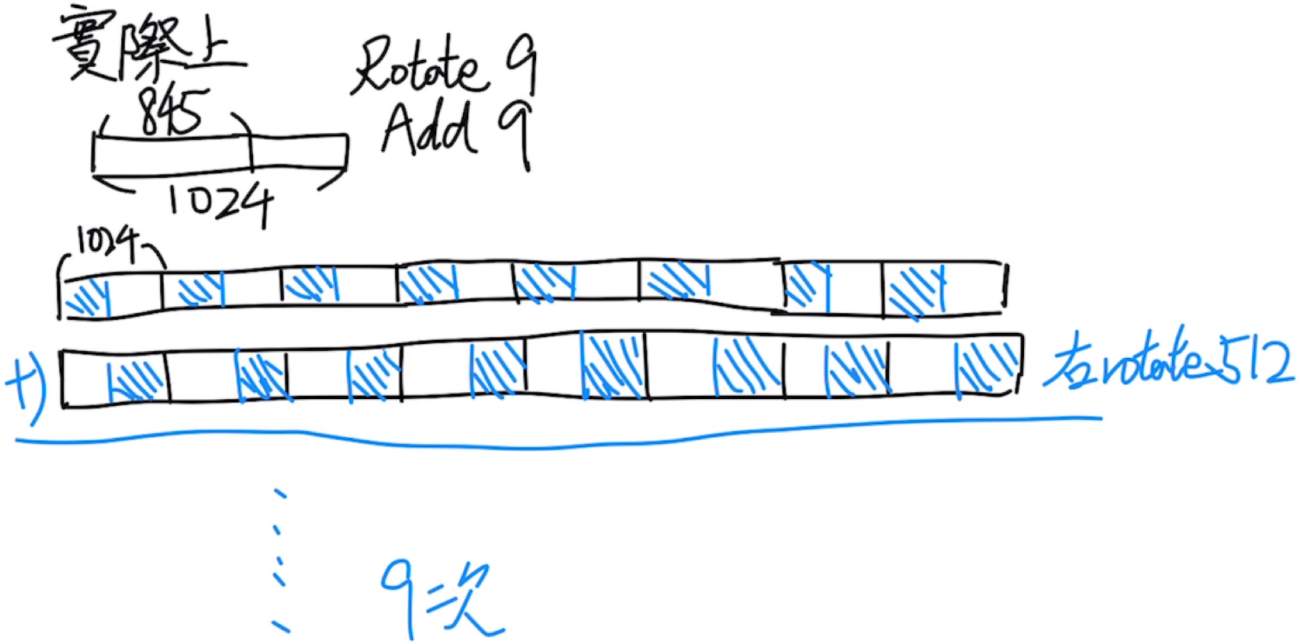
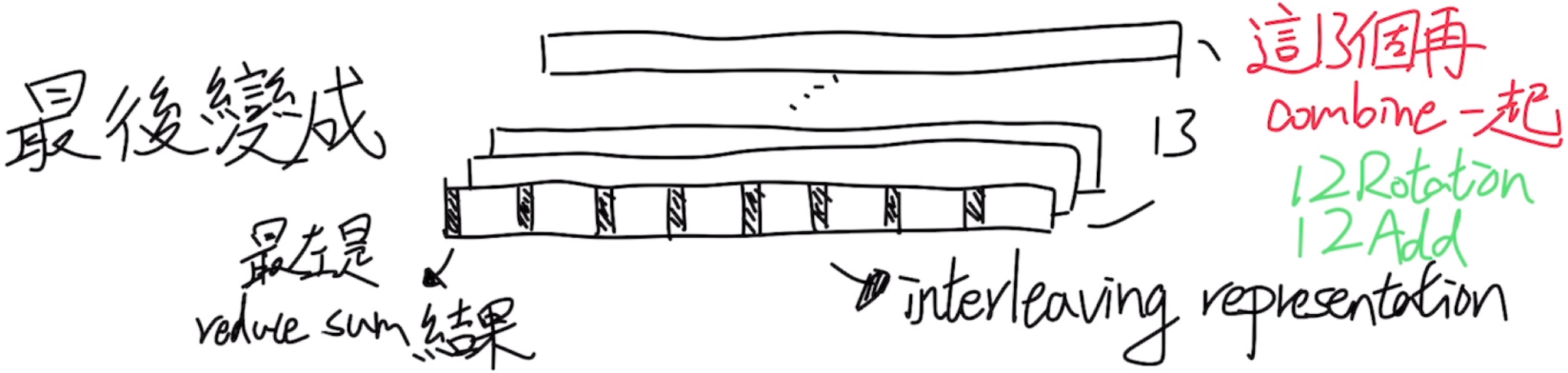


Figure 3: The structure of the network used for MNIST classification.

Dense layer 845 → 100





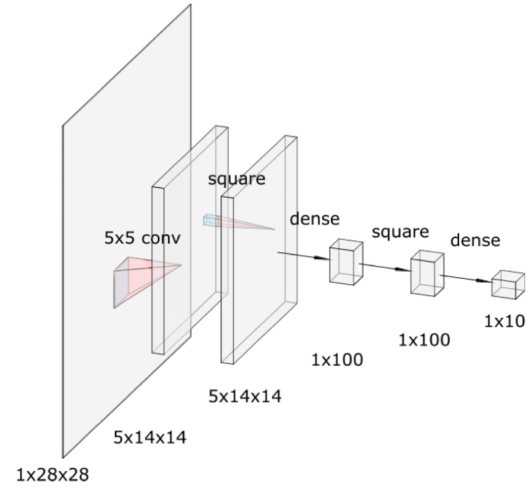
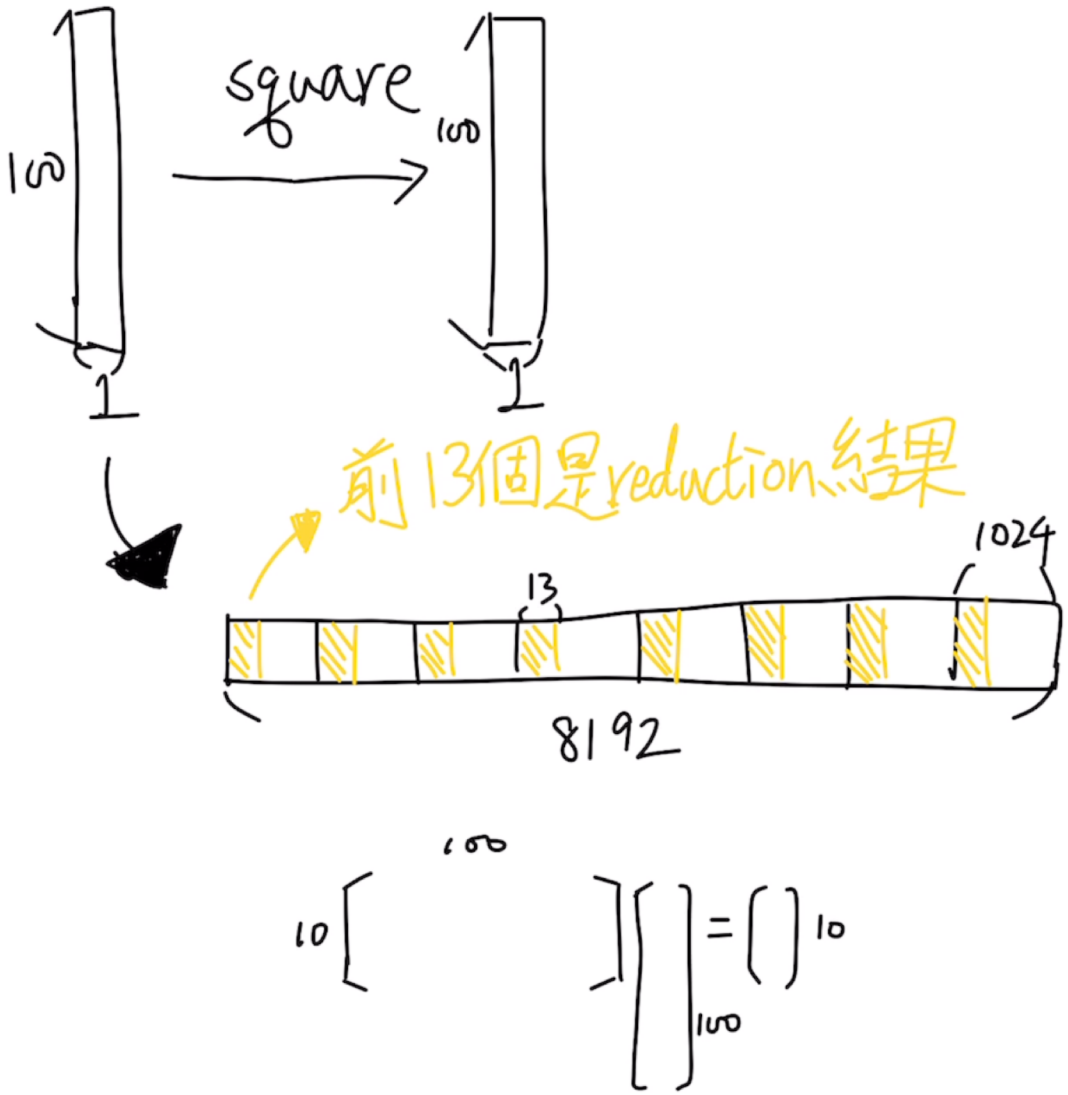


Figure 3: The structure of the network used for MNIST classification.

Square



Square：1 cipherMul

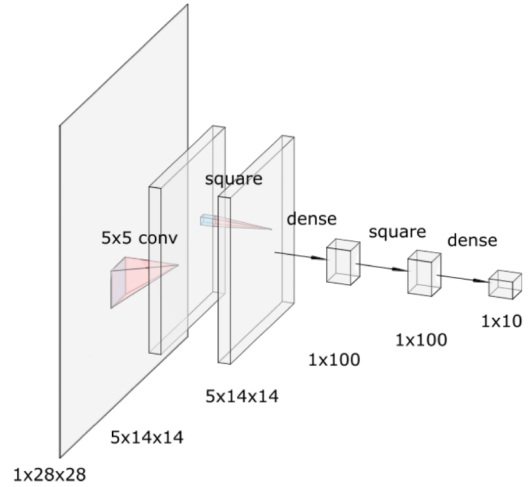
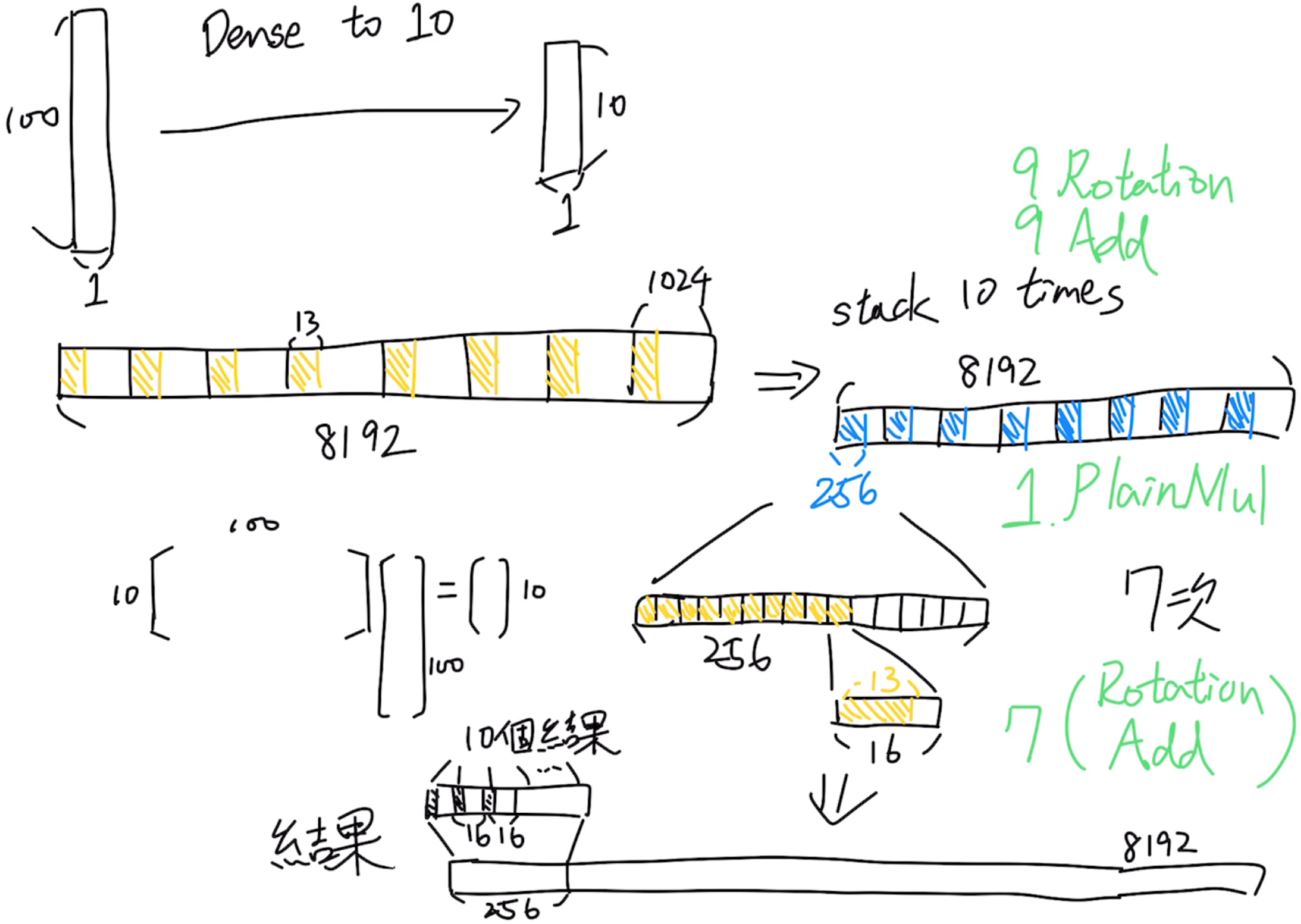


Figure 3: The structure of the network used for MNIST classification.

Dense layer 100 → 10



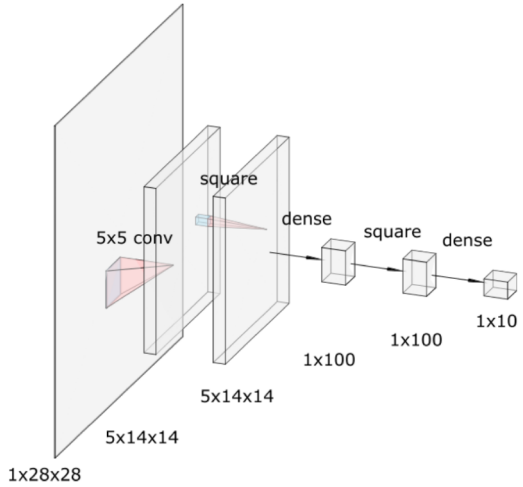
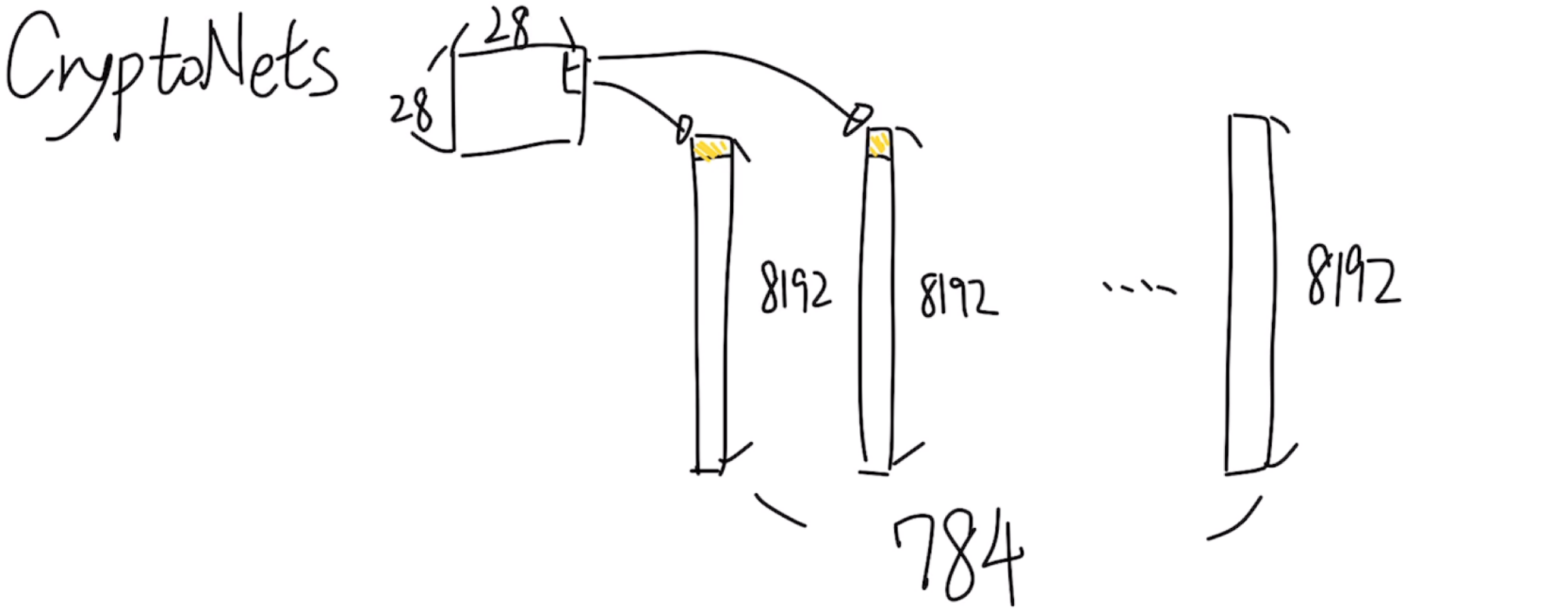


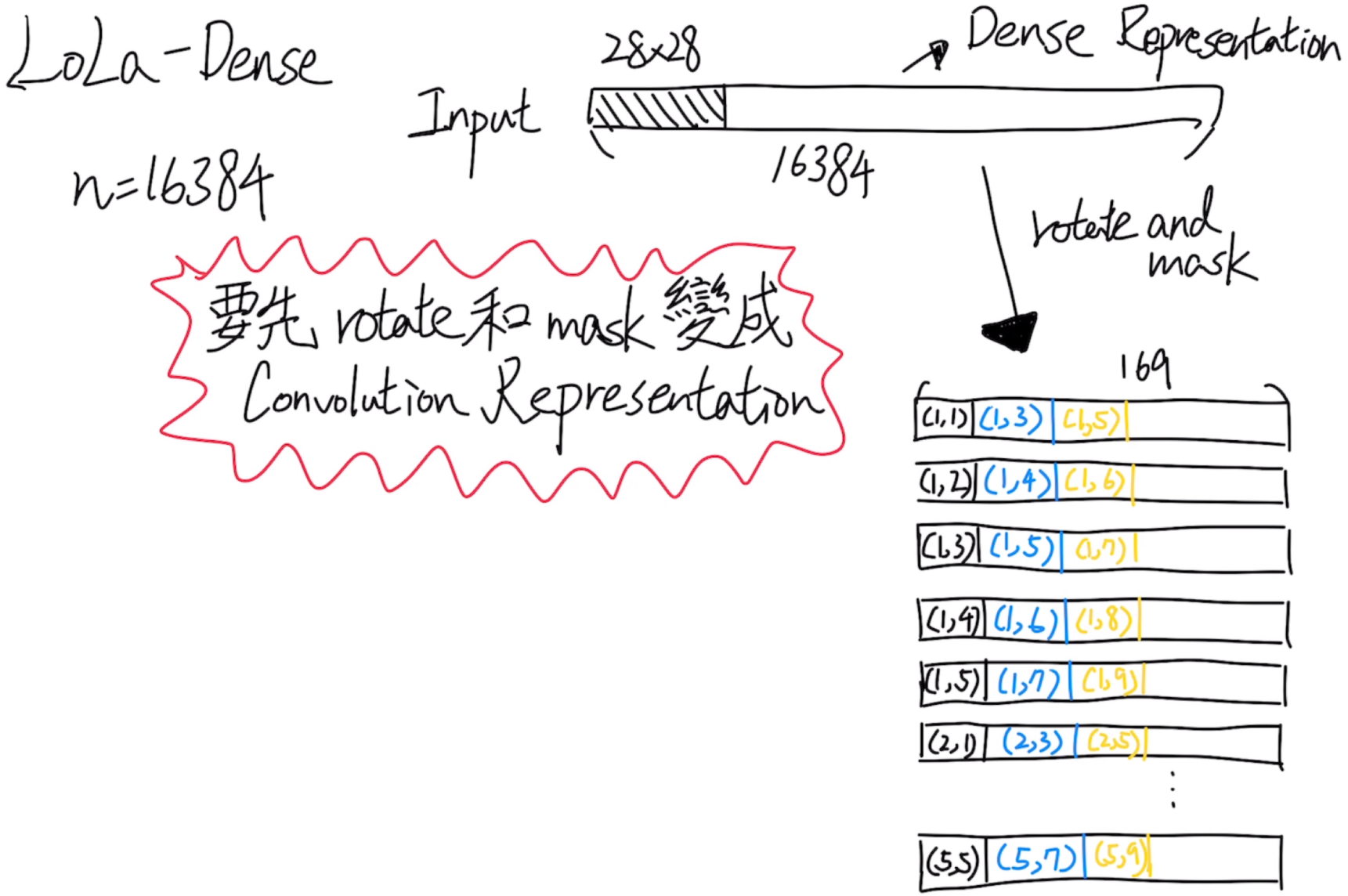
Figure 3: The structure of the network used for MNIST classification.

CryptoNets



High-latency, but can perform batch-operation (bs=8192) (amortized latency)

LoLa-Dense


LoLa-Dense
n=16384
28x28
Input
Dense Representation
16384
rotate and mask
要先 rotate和 mask 變成 Convolution Representation
169
(1,1) (1,3) (1,5)
(1,2) (1,4) (1,6)
(1,3) (1,5) (1,7)
(1,4) (1,6) (1,8)
(1,5) (1,7) (1,9)
(2,1) (2,3) (2,5)
(5,5) (5,7) (5,9)

LoLa-Small

LoLa small

One CNN → square activation

→ dense to 10

125 + 845×10 = 8575 doubles

Comparison

Azure standard B8ms virtual machine with 8 vCPUs and 32GB of RAM

Table 2: MNIST performance comparison. Solutions are grouped by accuracy levels.

Method	Accuracy	Latency	
FHE–DiNN100	96.35%	1.65	(Bourse et al., 2017)
LoLa-Small	96.92%	**0.29**	
CryptoNets	98.95%	205	(Dowlin et al., 2016)
nGraph-HE	98.95%[9]	135	(Boemer et al., 2018)
Faster-CryptoNets	98.7%	39.1	(Chou et al., 2018)
CryptoNets 2.3	98.95	24.8	
HCNN	99%	14.1	(Badawi et al., 2018)
LoLa-Dense	98.95%	7.2	
LoLa	98.95%	**2.2**	

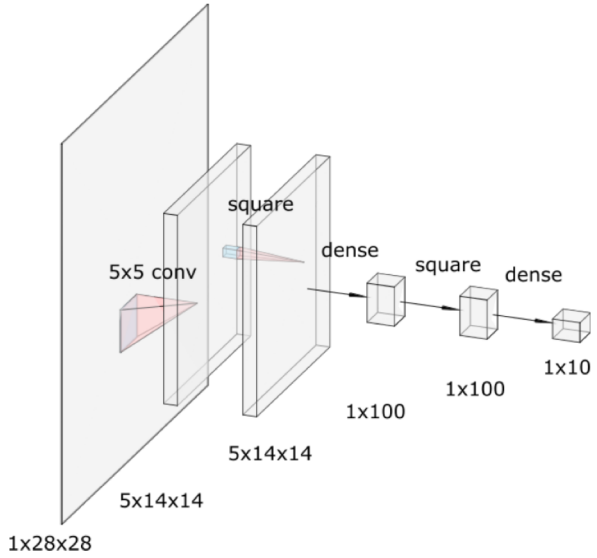


Figure 3: The structure of the network used for MNIST classification.

85625個double

11× faster than CryptoNets 2.3 and 93× faster than CryptoNets

Scaling

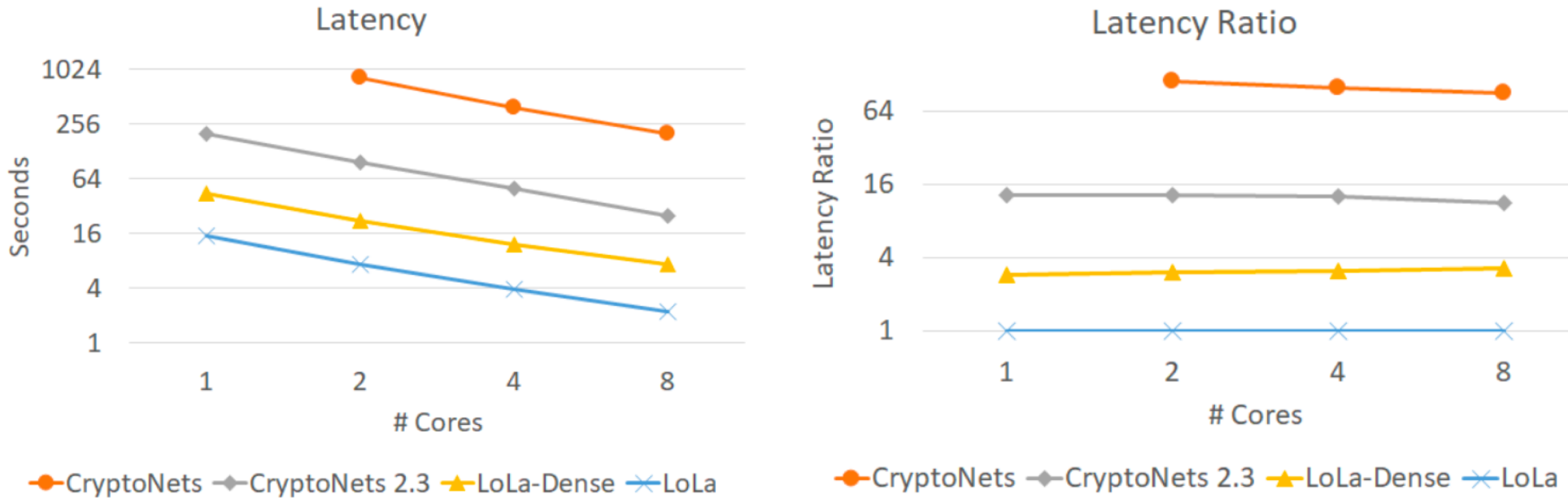


Figure 2: Left: latency of the different network implementations for the MNIST task with respect to the number of available cores. Right: ratio between the latency of each solution and the latency of LoLa

Testing

- Intel(R) Core(TM) i7-6700K CPU @ 4.00GHz, 8 cores, 64GB RAM

```
+-----------------------------------------------------+
|         BFV Performance Test with Degree: 8192      |
+-----------------------------------------------------+
/
| Encryption parameters :
|   scheme: BFV
|   poly_modulus_degree: 8192
|   coeff_modulus size: 218 (43 + 43 + 44 + 44 + 44) bits
|   plain_modulus: 786433
\
```

```
Average batch: 123 microseconds
Average unbatch: 152 microseconds
Average encrypt: 4859 microseconds
Average decrypt: 1395 microseconds
Average add: 60 microseconds
Average multiply: 14989 microseconds
Average multiply plain: 2456 microseconds
Average square: 10842 microseconds
Average relinearize: 5398 microseconds
Average rotate rows one step: 5425 microseconds
Average rotate rows random: 24049 microseconds
Average rotate columns: 5414 microseconds
```

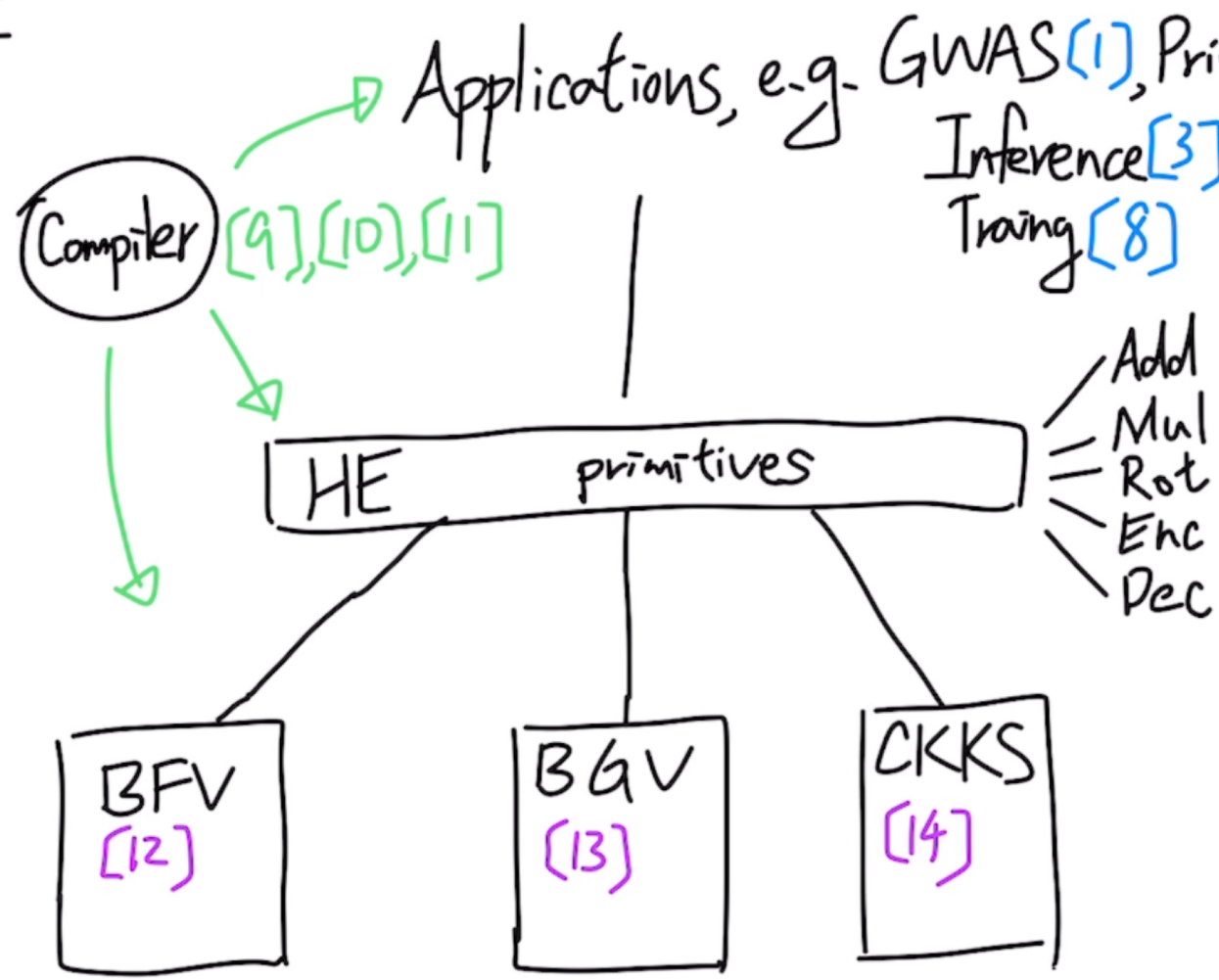
μs	Without HE	With HE
CNN	**222**	MulPlain:125 Add:129 Rotate:4 **=336440**
Square_1	**9**	**CipherMul: 1** **10842**
Dense to 100	**952**	Rotate: 136 Add: 149 MulPlain: 13 **=778668**
Square_2	**9**	**CipherMul: 1** **10842**
Dense to 10	**12**	Rotate: 16 Add: 17 MulPlain: 16 (9+7)*(5480)+2456 **=90292**
Total	**1204**	**4*1685488** **=4908336**

Outline

- Introduction
- Possible scenarios
- More specific
  - BFV encryption
  - Encrypted inference (LoLa)
- Research directions

FL + Hybrid scheme [21], [22]

HE



Scheme acceleration [15], [16], [17]

GPU [18], [19], [20]

FPGA [23]

ASIC [24]

RISC-HE

Reference

- On the explanation of FHE: https://www.youtube.com/watch?v=pXb39wj5ShI&ab_channel=OfficeoftheDirectorofNationalIntelligence
- Possible HE application: https://www.researchgate.net/publication/320976976_APPLICATIONS_OF_HOMOMORPHIC_ENCRYPTION
- Covid-19 contact tracing: https://dualitytech.com/wp-content/uploads/2020/04/Covid19_onepager.pdf
- Open source HE library: https://scholarworks.uark.edu/cgi/viewcontent.cgi?article=1074&context=csceuht
- Somewhat Homomorphic encryption: https://eprint.iacr.org/2012/144.pdf
- Homomorphic Encryption for Arithmetic of Approximate Numbers: https://eprint.iacr.org/2016/421.pdf
- Faster Homomorphic Linear Transform in HELib: https://eprint.iacr.org/2018/244.pdf
- Homomorphic Evaluation of the AES Circuit: https://eprint.iacr.org/2012/099.pdf
- SEAL-manual: https://www.microsoft.com/en-us/research/uploads/prod/2017/11/sealmanual-2-3-1.pdf
- CHET: https://www.cs.utexas.edu/~roshan/CHET.pdf
- Multi-GPU RNS: https://ieeexplore.ieee.org/document/9185077