A Lightweight McEliece Cryptosystem Co-processor Design

Lake Bu, Rashmi Agrawal, Haicheng, Michel A. Kinsy
Adaptive & Secure Computing Systems Lab
Boston University

Presentation Flow

- The problem of quantum computers coming real
- How pressing is the problem
- What can we do?
- Public-key systems for post-quantum era
  - Code-based encryption
  - Can we make it lighter & faster?
- Conclusion

Presentation Flow

- The problem of quantum computers coming real
- How pressing is the problem
- What can we do?
- Public-key systems for post-quantum era
  - Code-based encryption
  - Can we make it lighter & faster?
- Conclusion

What Media Have Said About the Coming of Quantum Computers

QUARTZ

WHERE THERE'S A WILL

Quantum computing could make the encryption behind every internet transaction obsolete—someday

By Joshua Holden • April 20, 2017
Professor, Rose-Hulman Institute of Technology

IEEE SPECTRUM

Tech Talk | Computing | Hardware

3 Mar 2016 | 19:03 GMT

Quantum Computer Comes Closer to Cracking RSA Encryption

Shor's algorithm performed in a system less than half the size experts expected

By Amy Nordrum

ZDNet

IBM warns of instant breaking of encryption by quantum computers: 'Move your data today'

Welcome to the future transparency of today as quantum computers reveal all currently encrypted secrets -- a viable scenario within just a few years.

By Tom Foremski for Tom Foremski: IMHO | May 18, 2018 -- 18:24 GMT (11:24 PDT) | Topic: Security

HACKADAY

HOME BLOG HACKADAY.IO TINDIE HACKADAY PRIZE SUBMIT ABOUT November 19, 2018

QUANTUM COMPUTING KILLS ENCRYPTION

by: Elliot Williams

79 Comments

SEARCH

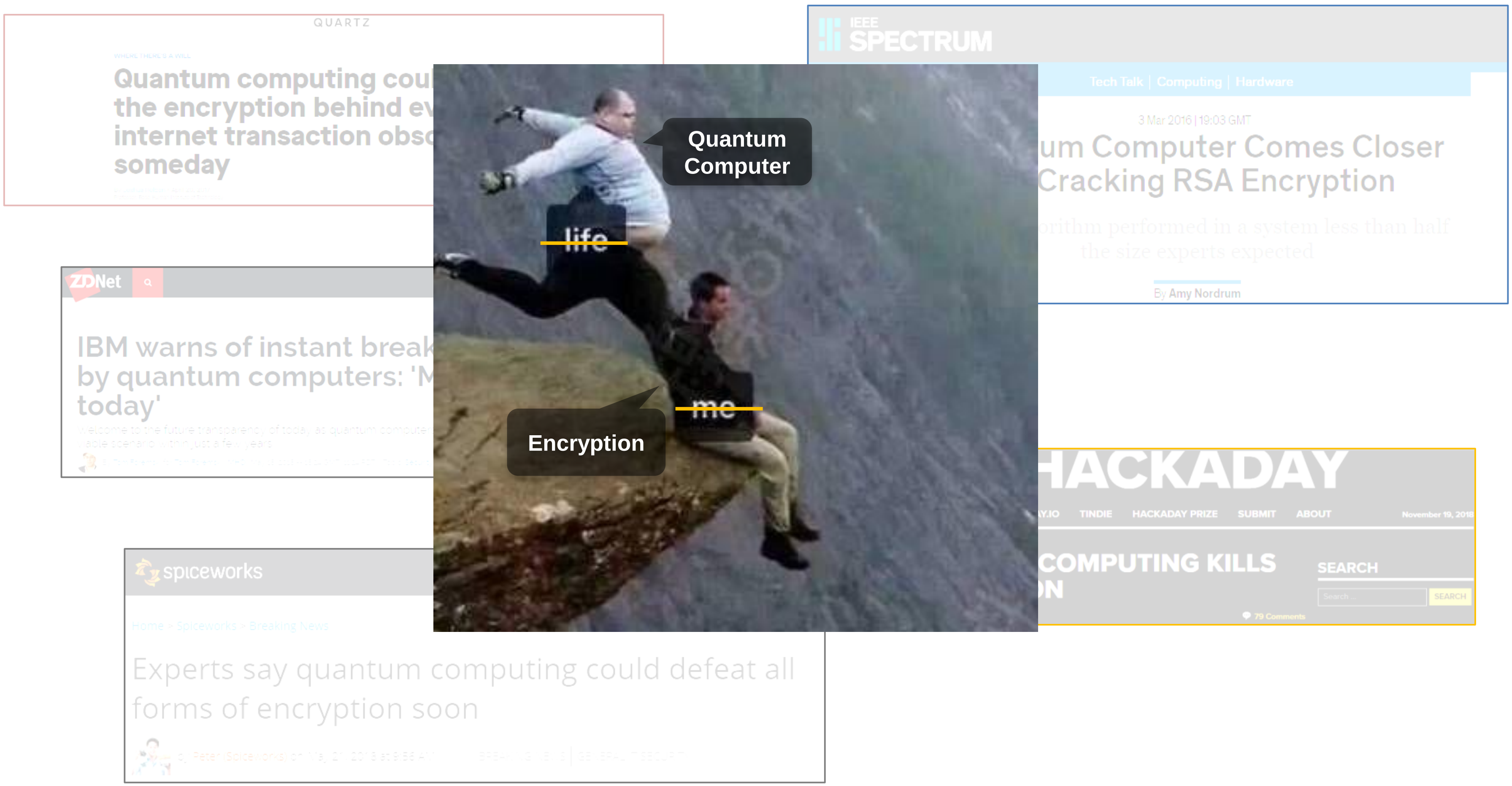
spiceworks

Home > Spiceworks > Breaking News

Experts say quantum computing could defeat all forms of encryption soon

by Peter (Spiceworks) on May 21, 2018 at 9:56 AM | BREAKING NEWS | GENERAL IT SECURITY

1. What Some Have Said About Quantum Computers



The Actual Situation Now

- Who is considered as non-post-quantum secure?

Algorithm	Secure in Post-quantum Era?
RSA-1024, -2048, -4096	No
Elliptic Curve Crypto (ECC)-256, -521	No
Diffie-Hellman	No
ECC Diffie-Hellman	No
AES-128, -192	No

[1]

[1] Zak Burkley, The Teachings of a Quantum Cat, 2016

One Question

- Can we increase the key size of some popular encryption schemes, so that they can be post-quantum secure?
  - Maybe yes, maybe no.

Table II. Equivalent Security Levels of AES and RSA under Attacks from Classic and Quantum Computers *

Attack Platform	Symmetric Encryption			Asymmetric (Public-key) Encryption		
	Algorithm	Key Size	Security Level	Algorithm	Key Size	Security Level
Classic Computers	AES-128	128	128	RSA-2048	2,048	112
	AES-256	256	256	RSA-15360	15,360	256

* TechBeacon, Waiting for quantum computing: Why encryption has nothing to worry about, 2018

One Question

- Can we increase the key size of some popular encryption schemes, so that they can be post-quantum secure?
  - Maybe yes, maybe no.

Table II. Equivalent Security Levels of AES and RSA under Attacks from Classic and Quantum Computers *

Attack Platform	Symmetric Encryption			Asymmetric (Public-key) Encryption		
	Algorithm	Key Size	Security Level	Algorithm	Key Size	Security Level
Classic Computers	AES-128	128	128	RSA-2048	2,048	112
	AES-256	256	256	RSA-15360	15,360	256
Quantum Computers	AES-128	128	64	RSA-2048	2,048	25
	AES-256	256	128	RSA-15360	15,360	31

Grover's algorithm

Shor's algorithm

* TechBeacon, Waiting for quantum computing: Why encryption has nothing to worry about, 2018

Presentation Flow

- The problem of quantum computers coming real
- How pressing is the problem
- What can we do?
- Public-key systems for post-quantum era
    - Code-based encryption
    - Can we make it lighter & faster?
- Conclusion

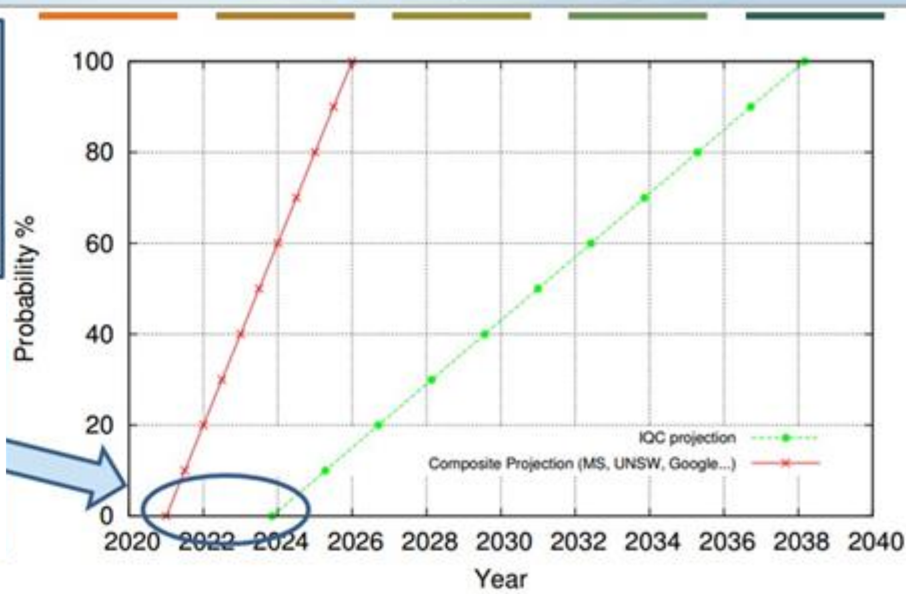
How Pressing is the Issue?

- The Timeline

Projected Probability of General Purpose* Quantum Computers Arriving By Year

The green graph is based on data from the IQC (Institute for Quantum Computing) provided earlier in 2015.

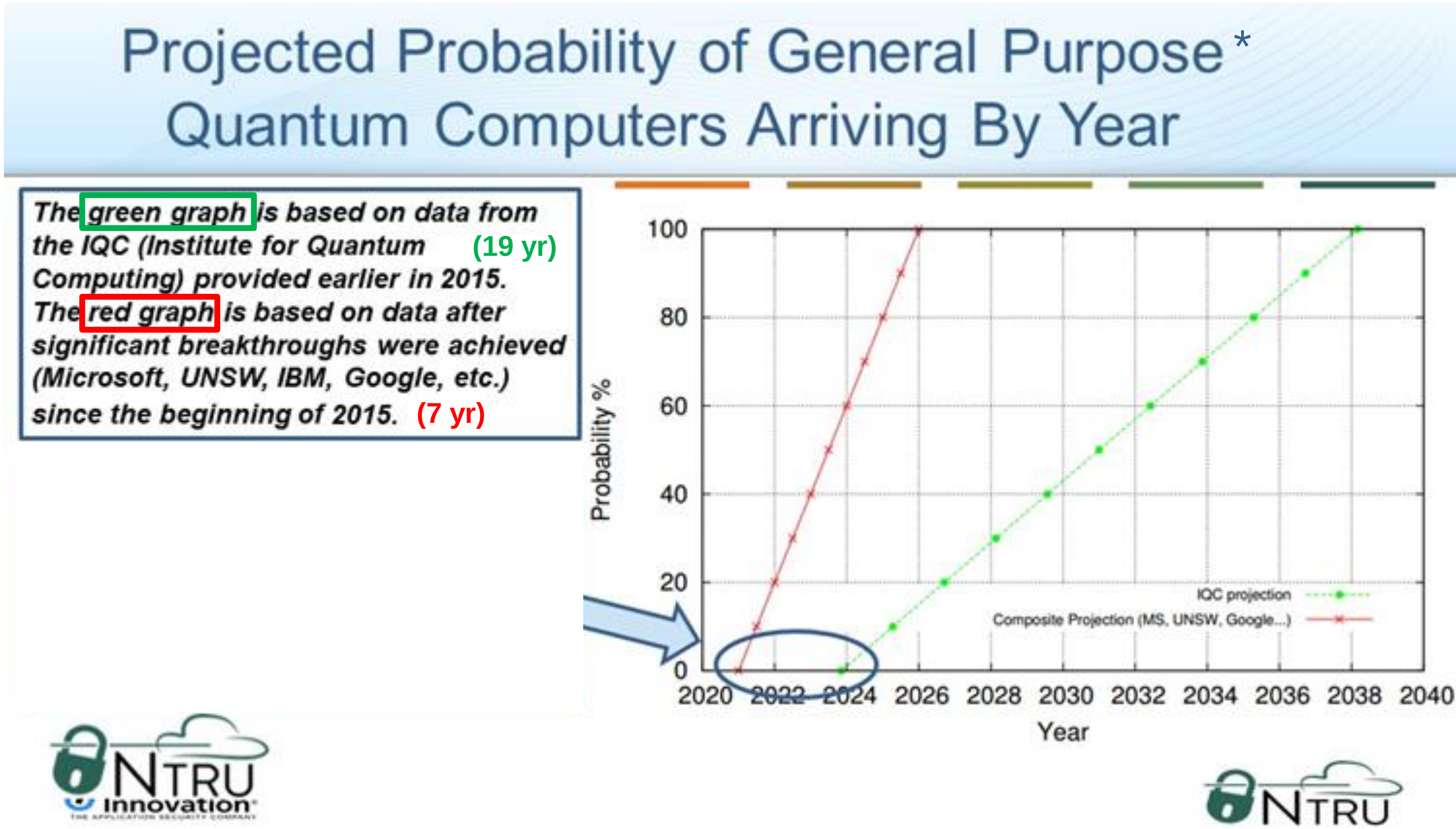
(19 yr)



* Lee Wilson, The Rise of Quantum Computers – The Current State of Cryptographic Affairs, 2016

How Pressing is the Issue?

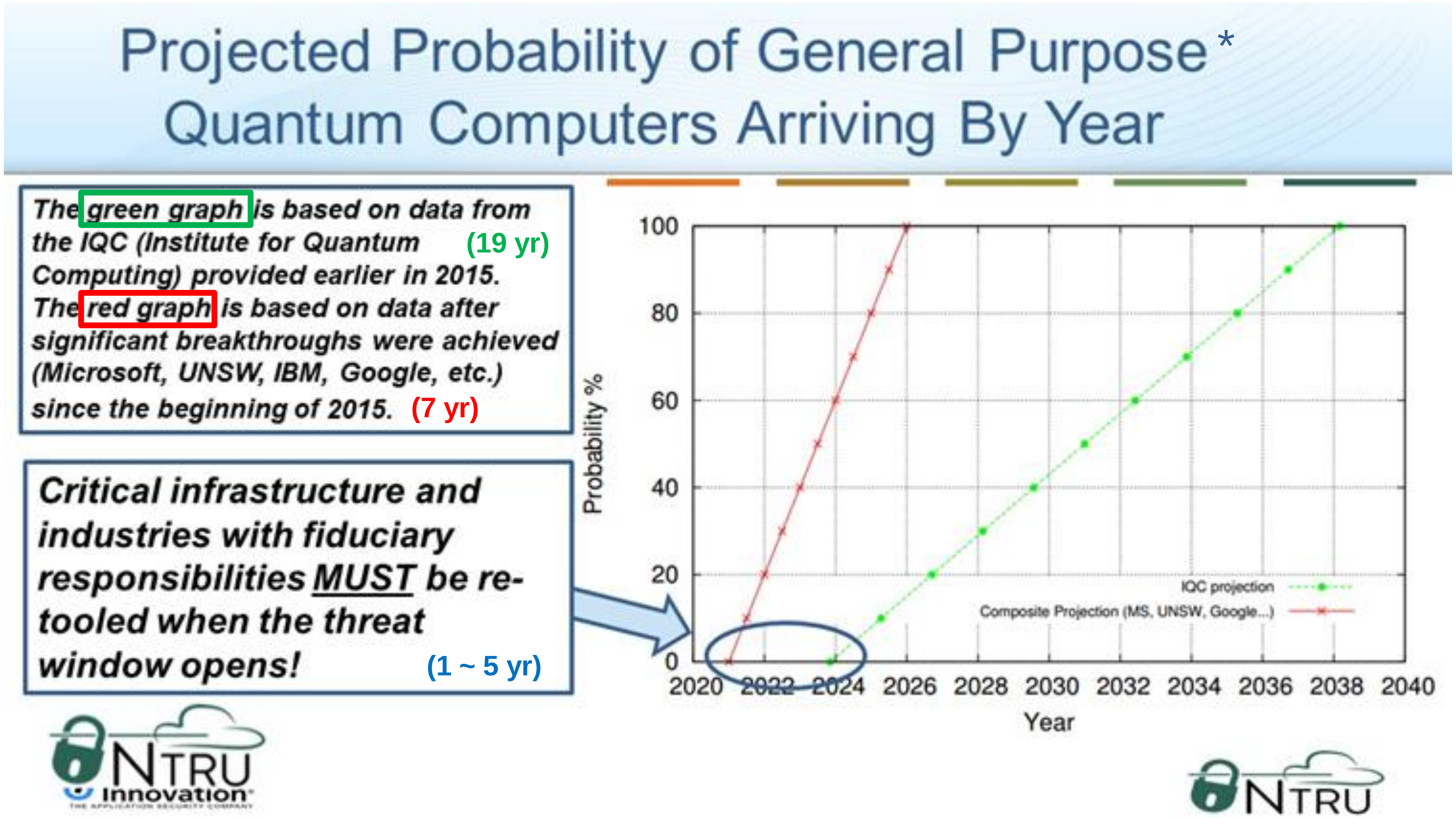
- The Timeline



* Lee Wilson, The Rise of Quantum Computers – The Current State of Cryptographic Affairs, 2016

How Pressing is the Issue?

- The Timeline



* Lee Wilson, The Rise of Quantum Computers – The Current State of Cryptographic Affairs, 2016

How Pressing is the Issue?

- What do companies/institutes say?
  - Microsoft Research
    - 5 years
  - NIST
    - 15 years

Presentation Flow

- The problem of quantum computers coming real
- How pressing is the problem
- What can we do?
- Public-key systems for post-quantum era
  - Code-based encryption
  - Can we make it lighter & faster?
- Conclusion

Post-Quantum Cryptography Standardization

NIST

Search CSRC

CSRC MENU

Information Technology Laboratory

COMPUTER SECURITY RESOURCE CENTER

CSRC

PROJECTS | POST-QUANTUM CRYPTOGRAPHY

Post-Quantum Cryptography [1]

Round 1 Submissions

Official comments on the First Round Candidate Algorithms should be submitted using the "Submit Comment" link for the appropriate algorithm. Comments from the pqc-forum Google group subscribers will also be forwarded to the pqc-forum Google group list. We will periodically post and update the comments received to the appropriate algorithm.

All relevant comments will be posted in their entirety and should not include PII information in the body of the email message.

Please refrain from using OFFICIAL COMMENT to ask administrative questions, which should be sent to pqc-comments@nist.gov

By selecting the "Submitter's Website" links, you will be leaving NIST.gov. We have provided links to submitter web sites because they may have information that would be of interest to you. No inferences should be drawn o.n account of other sites being referenced, or not, from this page. There may be other web sites that are more appropriate for your purpose. NIST does

PROJECT LINKS

Overview
FAQs
News
Events
Publications
Presentations

ADDITIONAL PAGES

Post-Quantum Cryptography Standardization
Call for Proposals

[1] https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions

Post-Quantum Cryptography (PQC) Standardization

- NIST
  - Jan 2017 - present
  - Evaluating 69 (5 withdrawn) submissions of PQC, to bring up a standard (just like AES or RSA):
    - 21 lattice-based
    - 18 code-based
    - Some hash-based
    - Some others

[1]

Algorithm	Algorithm Information *KAT files are included in zip file unless they were too large*	Submitters	Comments
BIG QUAKE	Zip File (4MB) IP Statements Website	Alain Couvreur Magali Bardet Elise Barelli Olivier Blazy Rodolfo Canto-Torres Philippe Gaborit Ayoub Otmani Nicolas Sendrier Jean-Pierre Tillich	Submit Comment View Comments
BIKE	Zip File (10MB) IP Statements Website	Nicolas Aragon Paulo Barreto Slim Bettaieb Loic Bidoux	Submit Comment View Comments
CFPKM	Zip File (<1MB) IP Statements Website	O. Chakraborty J.-C. Faugere L. Perret	Submit Comment View Comments
Classic McEliece	Zip File (<1MB) KAT Files (26MB) IP Statements Website	Daniel J. Bernstein Tung Chou Tanja Lange Ingo von Maurich Rafael Misoczki Ruben Niederhagen Edoardo Persichetti Christiane Peters Peter Schwabe Nicolas Sendrier Jakub Szefer Wen Wang	Submit Comment View Comments
Compact LWE	Zip File (1MB) IP Statements Website	Dongxi Liu Nan Li Jongkil Kim Surya Nepal	Submit Comment View Comments

Submission deadline Nov 30, 2017. List updated Dec 20, 2018.

[1] https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions

Why Code-based?

1. It has withstood the test of time
   - Published in 1978, 40 years of examination
2. The security reduction (hardness) is decoding of linear codes without knowing the encoding algorithm
   - NP-hard for quantum computers
3. Although:
   - Its key size is large: 1MB
   - Its decryption is highly parallel: easy for both good & bad guys

Presentation Flow

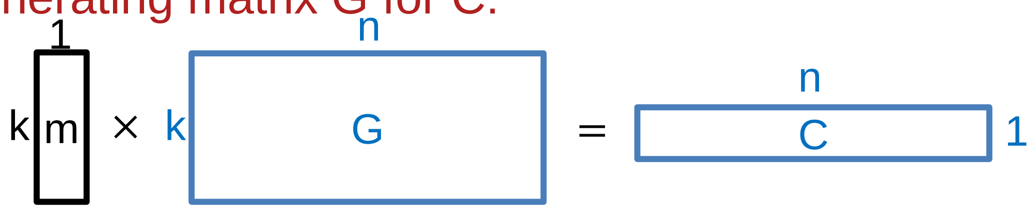
- The problem of quantum computers coming real
- How pressing is the problem
- What can we do?
- Public-key systems for post-quantum era
  - Code-based encryption
  - Can we make it lighter & faster?
- Conclusion

Code-based Encryption

- McEliece cryptosystem
  - Error Correction Code (ECC)
    - A (n, k, t) ECC code C:
      - k: size of your message (information)
      - n: size of the encoded message (codeword)
      - t: # of random errors C can tolerate

Code-based Encryption

- McEliece cryptosystem
  - Error Correction Code (ECC)
    - A (n, k, t) ECC code C:
      - k: size of your message (information)
      - n: size of the encoded message (codeword)
      - t: # of random errors C can tolerate
    - Generating matrix G for C:



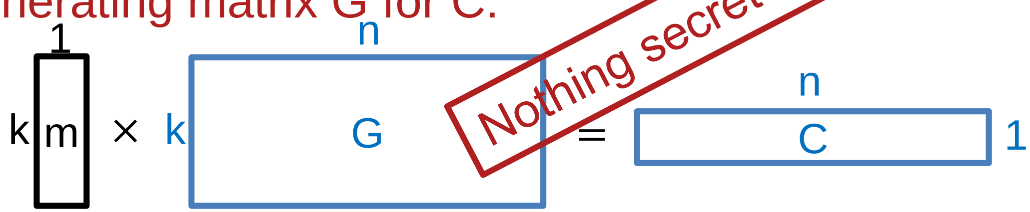
    - When m is encoded by G to C, it is able to be recovered even after being distorted by t errors
      - Err-correct (C+ e) = m

Code-based Encryption

- McEliece cryptosystem
  - Error Correction Code (ECC)
    - A (n, k, t) ECC code C:
      - k: size of your message (information)
      - n: size of the encoded message (codeword)
      - t: # of random errors C can tolerate

Nothing secret to anyone.

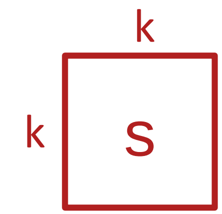
    - Generating matrix G for C:

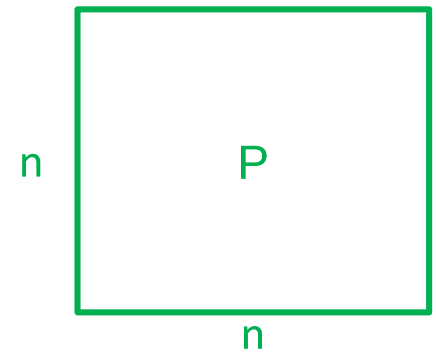


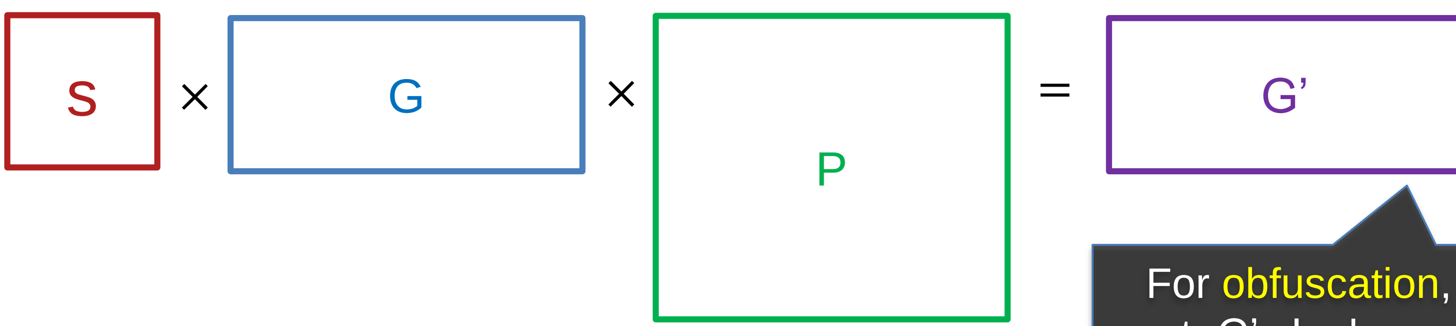
    - When m is encoded by G to C, it is able to be recovered even after being distorted by t errors
      - Err-correct (C+ e) = m

Code-based Encryption

- Key Generation (Alice)
  - Pick two random matrices:
    - $k \times k$ non-singular binary matrix S:
    - $n \times n$ permutation binary matrix P:
  - Compute:
    - $k \times n$ matrix $G' = S \times G \times P$





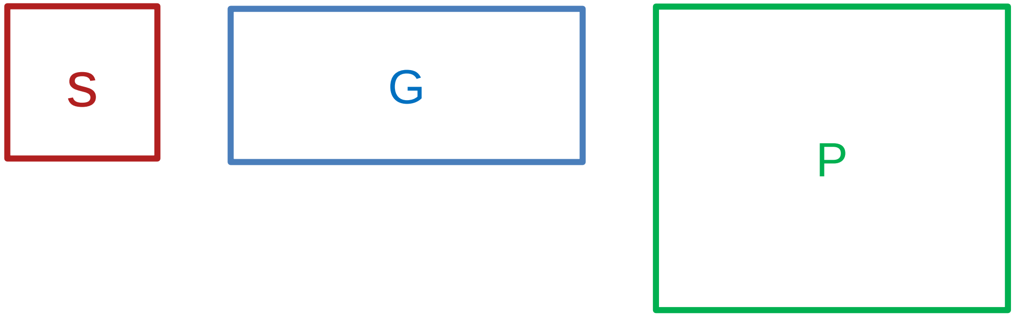


For obfuscation, s.t. G's leaks no information of G

Code-based Encryption

- Key Generation
  - Public key (t, G’):
    - t,

  - Private key (S, G, P):
    - Remember G’ = S×G×P



Code-based Encryption

- McEliece cryptosystem
  - Encryption (Bob to Alice):
    1. Message (plaintext) encrypted to codeword

       $$\boxed{m} \times \boxed{G'} = \boxed{C}$$

    2. Codeword obfuscated by error to cipher

       $$\boxed{C} + \boxed{e \text{ (t of 1's)}} = \boxed{\text{Cipher}}$$

Code-based Encryption

- McEliece cryptosystem
  - Encryption (Bob to Alice):
    1. Message (plaintext) encrypted to codeword

       $\boxed{m} \times \boxed{G'} = \boxed{C}$

    2. Codeword obfuscated by error to cipher

       $\boxed{C} + \boxed{e\text{ (t of 1's)}} = \boxed{\text{Cipher}}$

Can be arbitrarily chosen. Will be eliminated anyways but makes attack difficult.

Code-based Encryption

- McEliece cryptosystem
  - Encryption (Bob to Alice):
    1. Message (plaintext) encrypted to codeword

       $m \times G' = C$

    2. Codeword obfuscated by error to cipher

       $C + e \text{ (t of 1's)} = \text{Cipher}$

       Can be arbitrarily chosen. Will be eliminated anyways but makes attack difficult.

- Decryption (Alice computes):
  - Remember $G' = S \times G \times P$

  1) $\text{Cipher}' = \text{Cipher} \times P^{-1} = (m \times G' + e) \times P^{-1} = (m \times S \times G) + (e \times P^{-1})$
  2) Err-correct $(m \times S \times G + e') = m \times S$
  3) $m \times S \times S^{-1} = m$

Code-based Encryption

- McEliece cryptosystem
  - Encryption (Bob to Alice):
    1. Message (plaintext) encrypted to codeword

       $\boxed{m} \times \boxed{G'} = \boxed{C}$

    2. Codeword obfuscated by error to cipher

       $\boxed{C} + \boxed{e \text{ (t of 1's)}} = \boxed{\text{Cipher}}$

Can be arbitrarily chosen. Will be eliminated anyways but makes attack difficult.

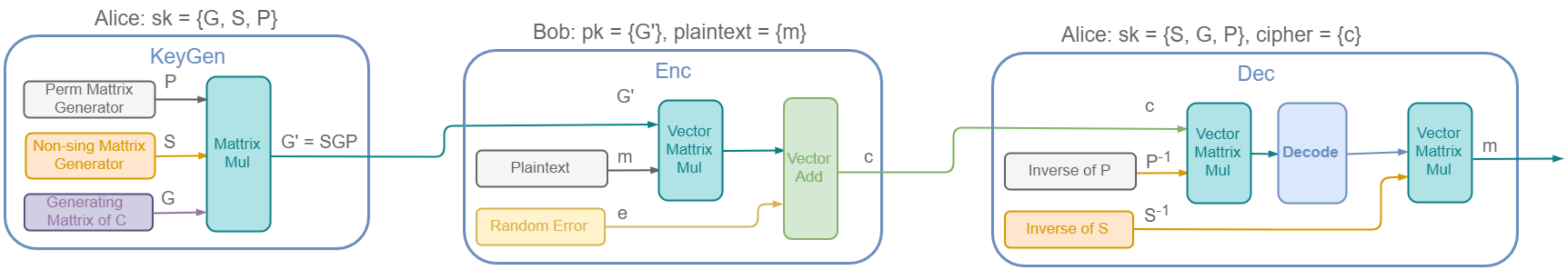
  - Decryption (Alice computes):
    - Remember $G' = S \times G \times P$

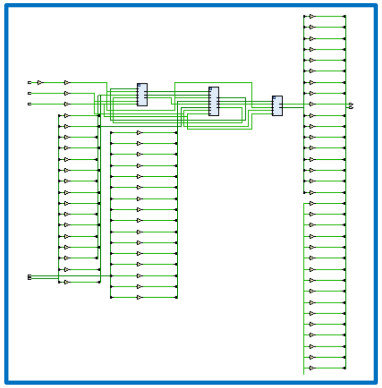
    1) $\text{Cipher}' = \text{Cipher} \times P^{-1} = (m \times G' + e) \times P^{-1} = (m \times S \times G) + (e \times P^{-1})$
    2) $\text{Err-correct}(m \times S \times G + e') = m \times S$
    3) $m \times S \times S^{-1} = m$

Attacker does not know: S, G, P
Thus he/she cannot compute:
S^{-1}, Err-correct(e), P^{-1}
Therefore, he/she does not know m

Code-based Encryption

- Architecture
  - Key Generation
  - Encryption
  - Decryption





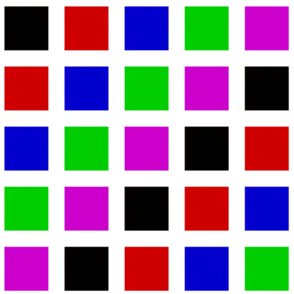
Code-based Encryption

- Can we make it faster?
  - The original scheme uses binary Goppa codes
    - Encoding & decoding involves
      - Long polynomial division
      - Solving equations over finite fields
  - We propose using Orthogonal Latin Square Codes
    - Encoding & decoding involves
      - Binary operations only
      - Can be fully parallel

Code-based Encryption

- Orthogonal Latin Square Codes (OLSC)

Aα	Bγ	Cδ	Dβ
Bβ	Aδ	Dγ	Cα
Cγ	Dα	Aβ	Bδ
Dδ	Cβ	Bα	Aγ



- Information: q^2
- Redundancy: $2mq$
- Errors to correct: m
- $\left|\begin{array}{c:c} M_0 & \\ M_1 & I_{2mq} \\ \cdots & \\ M_{2m-1} & \end{array}\right|$

1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
0	0	0	1	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
1	0	0	0	0	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	1	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	1	0	1	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	1	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Code-based Encryption

- OLSC-based McEliece Cryptosystem
  - Information: q^2
  - Redundancy: $2mq$
  - Errors to correct: m
  - $$\left[\begin{array}{c:c} \begin{matrix} M_0 \\ M_1 \\ \cdots \\ M_{2m-1} \end{matrix} & I_{2mq} \end{array}\right]$$

Algorithm 1: OLSC-based McEliece Cryptosystem

Let $G' = SGP$ and t be the public key, and $\{G, S, P\}$ the private key, where G is a $k \times n$ OLSC encoding matrix with random permutation of columns, and H as its corresponding decoding matrix. Let each Latin square size $q \times q$. Let m be the plaintext and c the encrypted cipher.

Precompute: S^{-1}, P^{-1} as the inverse to S, P respectively.

$c' \leftarrow cP^{-1}$

$u \leftarrow Hc' \times H$

for i=0 to n

  $m_i' \leftarrow (u_i > q/2)? \sim c_i' : c_i'$

$m \leftarrow m'S^{-1}$

return m

Code-based Encryption

- OLSC-based McEliece Cryptosystem

Code	Time Complexity
Binary-Goppa	$O(n^2)$
OLSC	$O(1)$

Algorithm 1: OLSC-based McEliece Cryptosystem

Let $G' = SGP$ and t be the public key, and $\{G, S, P\}$ the private key, where G is a $k \times n$ OLSC encoding matrix with random permutation of columns, and H as its corresponding decoding matrix. Let each Latin square size $q \times q$. Let m be the plaintext and c the encrypted cipher.

Precompute: S^{-1}, P^{-1} as the inverse to S, P respectively.

$c' \leftarrow cP^{-1}$

$u \leftarrow Hc' \times H$

for i=0 to n

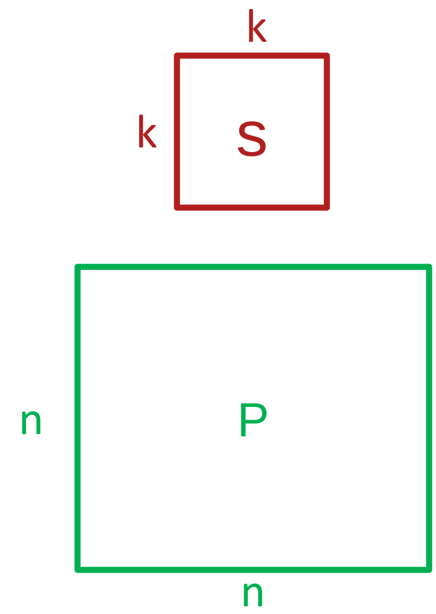
    $m'_i \leftarrow (u_i > q/2)? \sim c'_i : c'_i$

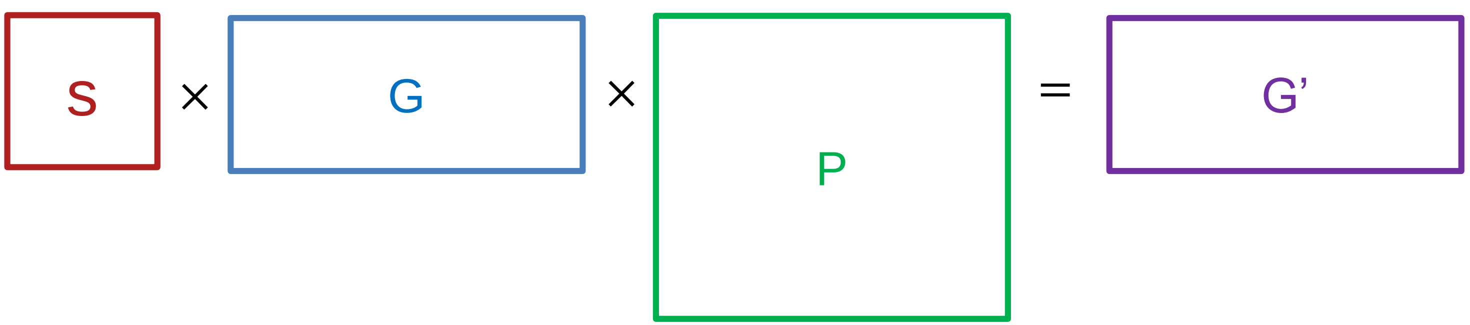
$m \leftarrow m'S^{-1}$

return m

Recall that …

- McEliece cryptosystem
  - Pick two random matrices:
    - $k \times k$ binary matrix S:
    - $n \times n$ binary permutation matrix P:
  - Compute:
    - $k \times n$ matrix $G' = S \times G \times P$





Code-based Encryption

- G (the private key)
  - From a t= 2 double error-correcting OLSC code

Generate G

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0
2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0
3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1
4	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	1	0
5	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	1
6	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	1	0	0	0
7	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0	1	0	0
8	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	1
9	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	1	0
10	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	1	0	0
11	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0	1	0	0	1	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	0	1	0	1	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	1
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	0	1	0

Code-based Encryption

- S and its inverse (the private key)

S

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
2	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
3	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
4	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
5	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
6	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
10	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
11	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
12	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
13	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1
14	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1
15	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

S-1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
3	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
5	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
6	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
7	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
9	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
10	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
11	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
12	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
13	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
14	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Code-based Encryption

- P and its inverse (the private key)

P

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
4	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
6	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
8	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
10	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
12	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
14	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

p-1

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Code-based Encryption

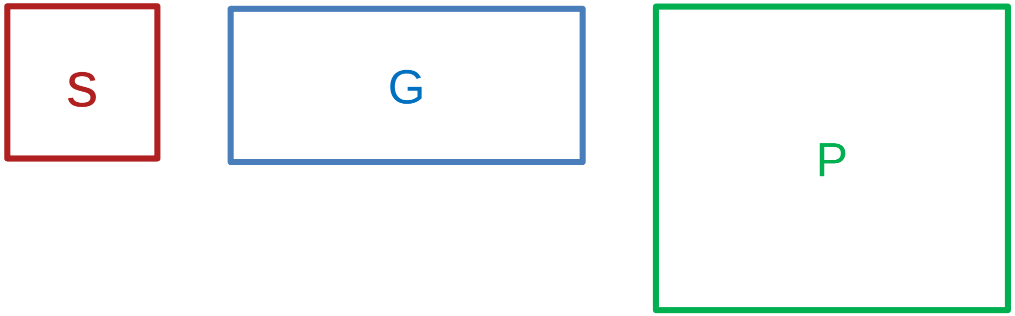
- G’ = S×G×P the public key

S*G*P = PK	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	1	1	1
2	1	1	1	0	0	0	0	0	0	0	0	1	0	1	1	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	1	1
3	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	0	1	1	1	0	1	1	0	1	0	0	0	0	0	0	1	1
4	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1
5	1	1	0	0	0	0	0	0	1	0	1	1	1	1	1	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1
6	1	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1
7	1	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	1	0	0	0	1	1	1	1	1	1	0	0	1	1	1	1	0	1	1	0	0	0	0	0	0
11	0	0	0	0	0	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0
12	0	0	0	0	0	0	1	1	0	1	1	1	1	1	0	1	1	1	1	0	0	1	0	0	1	1	1	0	0	0	0	0
13	0	0	0	0	0	1	1	1	0	0	1	0	0	1	0	1	1	0	1	0	0	1	0	0	1	1	1	0	0	0	0	0
14	0	0	0	0	0	1	1	1	0	1	1	0	0	1	0	0	1	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0
15	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

Code-based Encryption

- Key Generation
  - Public key (t, G'): Alice to Bob
    - t = 2,

  - Private key (S, G, P): Alice to herself
    - Remember G' = S×G×P



Code-based Encryption

- Encryption (Bob to Alice)
    1. Message (plaintext) encrypted to codeword

m × G' = C

7777

0	0	0	1	1	1	1	0	0	1	1	0	0	0	0	1

×

S*G*P = PK

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	1	1	1
2	1	1	1	0	0	0	0	0	0	0	0	1	0	1	1	0	0	1	1	0	1	0	0	0	0	0	0	0	0	1	1	1
3	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	0	1	1	1	0	1	1	0	1	0	0	0	0	0	0	1	1
4	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1
5	1	1	0	0	0	0	0	0	1	0	1	1	1	1	1	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1
6	1	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1
7	1	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	1	0	0	0	1	1	1	1	1	1	0	0	1	1	1	1	0	1	1	0	0	0	0	0	0
11	0	0	0	0	0	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0
12	0	0	0	0	0	0	1	1	0	1	1	1	1	1	0	1	1	1	1	0	0	1	0	0	1	1	1	0	0	0	0	0
13	0	0	0	0	0	1	1	1	0	0	1	0	0	1	0	1	1	0	1	0	0	1	0	0	1	1	1	0	0	0	0	0
14	0	0	0	0	0	1	1	1	0	1	1	0	0	1	0	0	1	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0
15	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

=

0	1	1	0	1	1	1	1	1	0	1	1	0	0	1	0	0	0	0	1	0	1	1	1	1	0	1	1	0	0	0	0

Code-based Encryption

- Encryption (Bob to Alice)
    1. Message (plaintext) encrypted to codeword
    2. Codeword obfuscated by error to cipher

C + e (t of 1's) = Cipher

0	1	1	0	1	1	1	1	1	0	1	1	0	0	1	0	0	0	0	1	0	1	1	1	1	0	1	1	0	0	0	0

+

1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

=

1	1	1	0	1	1	1	1	1	0	1	1	0	0	1	1	0	0	0	1	0	1	1	1	1	0	1	1	0	0	0	0

Code-based Encryption

- Decryption (Alice)

  (Remember $G' = S \times G \times P$)

  1) $\text{Cipher} \times P^{-1} = (m \times G' + e) \times P^{-1} = (m \times S \times G) + (e \times P^{-1})$

$$\begin{pmatrix}1\\1\\1\\0\\1\\1\\1\\1\\1\\1\\0\\1\\1\\0\\0\\1\\1\\0\\0\\0\\1\\0\\1\\1\\1\\1\\0\\1\\1\\0\\0\\0\end{pmatrix} \times$$

p-1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

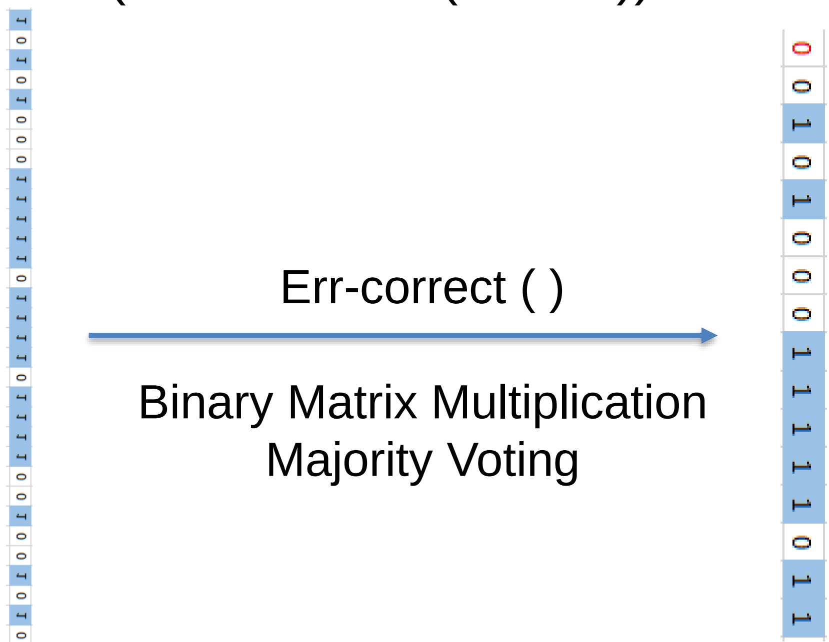
$$= \begin{pmatrix}0\\1\\0\\1\\0\\0\\1\\0\\0\\1\\1\\1\\1\\0\\1\\1\\1\\1\\0\\1\\1\\1\\1\\1\\0\\0\\0\\1\\0\\1\\0\\1\end{pmatrix}$$

Code-based Encryption

- Decryption (Alice)

  (Remember $G' = S \times G \times P$)

  1) $\text{Cipher} \times P^{-1} = (m \times G' + e) \times P^{-1} = (m \times S \times G) + (e \times P^{-1})$
  2) $\text{Err-correct}\,(m \times S \times G + (e \times P^{-1})) = m \times S$



Code-based Encryption

Attacker does not know: S, G, P
Thus he/she cannot compute:
S^{-1}, Err-correct( ), P^{-1}
Therefore, he/she does not know m

- Decryption (Alice)

(Remember $G' = S \times G \times P$)

1) $\text{Cipher} \times P^{-1} = (m \times G' + e) \times P^{-1} = (m \times S \times G) + (e \times P^{-1})$
2) $\text{Err-correct}(m \times S \times G + (e \times P^{-1})) = m \times S$
3) $m \times S \times S^{-1} = m$

0
0
1
0
1
0
0
0
1
1
1
1
1
0
1
1

×

S-1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
3	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
5	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
6	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
7	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
9	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
10	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
11	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
12	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
13	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
14	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

=

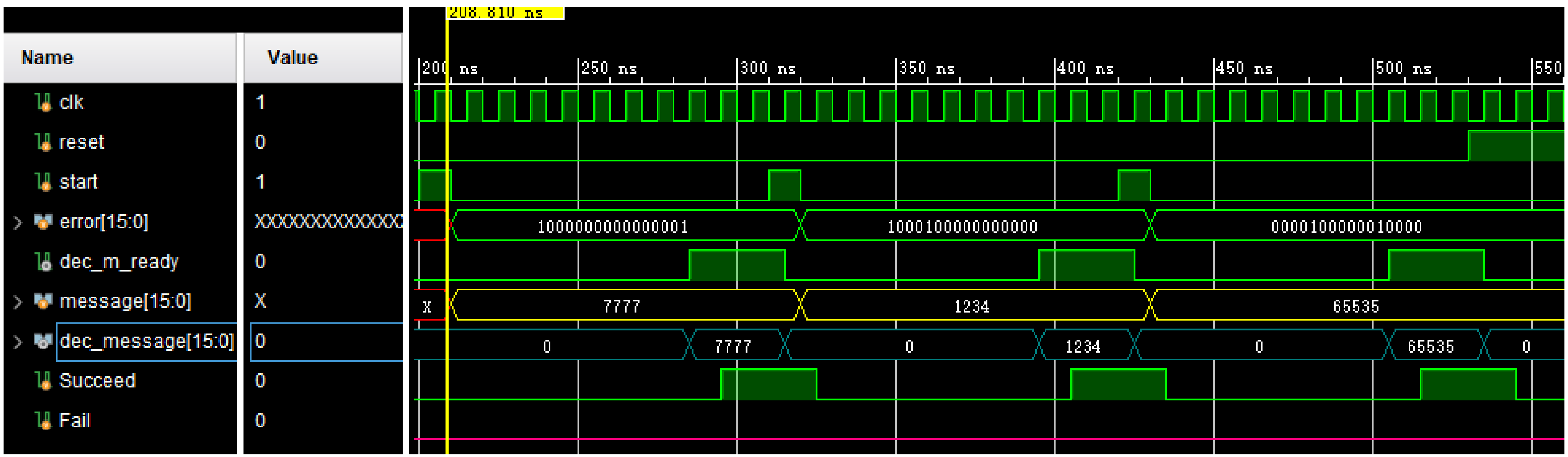
7777

0
0
0
1
1
1
1
0
0
1
1
0
0
0
0
1

= m

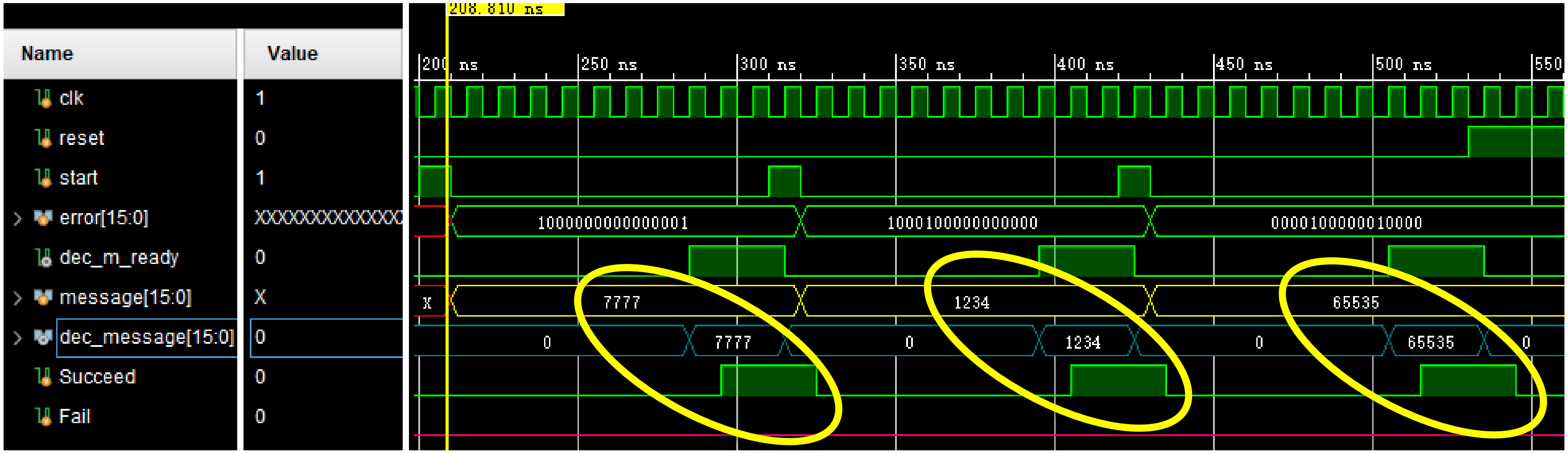
Code-based Encryption

- Simulation successful
  - Able to encrypt & decrypt 16-bit plaintexts



Code-based Encryption

- Simulation successful
  - Able to encrypt & decrypt 16-bit plaintexts



Presentation Flow

- The problem of quantum computers coming real
- How pressing is the problem
- What can we do?
- Public-key systems for post-quantum era
  - Code-based encryption
  - Can we make it lighter & faster?
- Conclusion

What needs to be done?

- McEliece cryptosystem
  - Post-quantum secure key size:
    - $k \times n$ matrix G': k = 5413, n = 6960, t = 119
    - Key size: 1mb for 128-bit security (RSA usually 2048 bits)

G'

- Crypto-analysis
  - Key space (G, S, P)
  - Decoding techniques

Any Questions?