CRYPTO2 - 19/12/2023 + 20/12/2023

ASYMMETRIC ENCRYPTION SCHEME

TRAPDOOR ONE-WAY FUNCTION

INTRODUCTION TO RSA

SHOR ALGORITHM AND CONSEQUENCES

POST-QUANTUM CRYPTOGRAPHY

INTRODUCTION TO Mc ELIECE SCHEME

GENERATOR MATRIX

CODEBOOK

MINIMUM DISTANCE

ERROR VECTORS

DECODING RULE

ERROR CORRECTION CAPABILITY

COMPLEXITY ISSUE

PARITY CHECK MATRIX

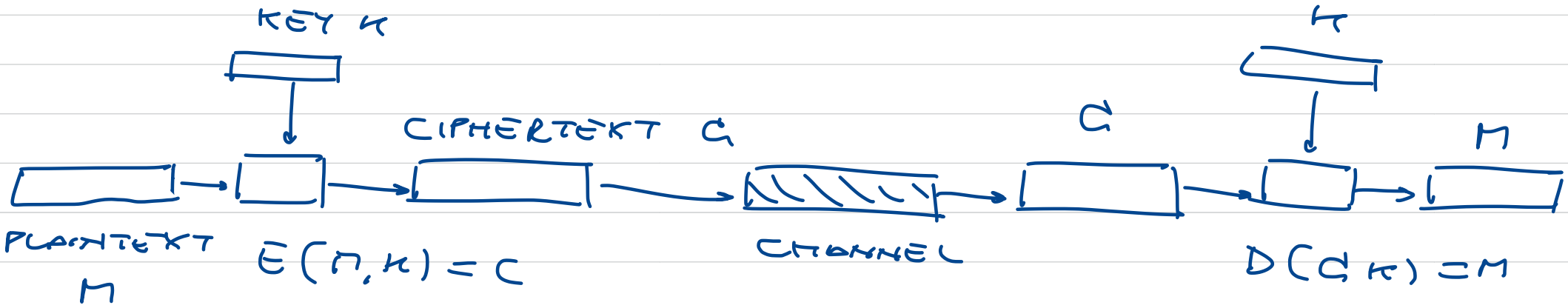
SYNDROME PROPERTY

SYNDROME DECODING

Mc ELIECE ENCRYPTION AND DECRYPTION

SYMMETRIC ENCRYPTION SCHEME

SAME KEY FOR ENCRYPTION / DECRYPTION



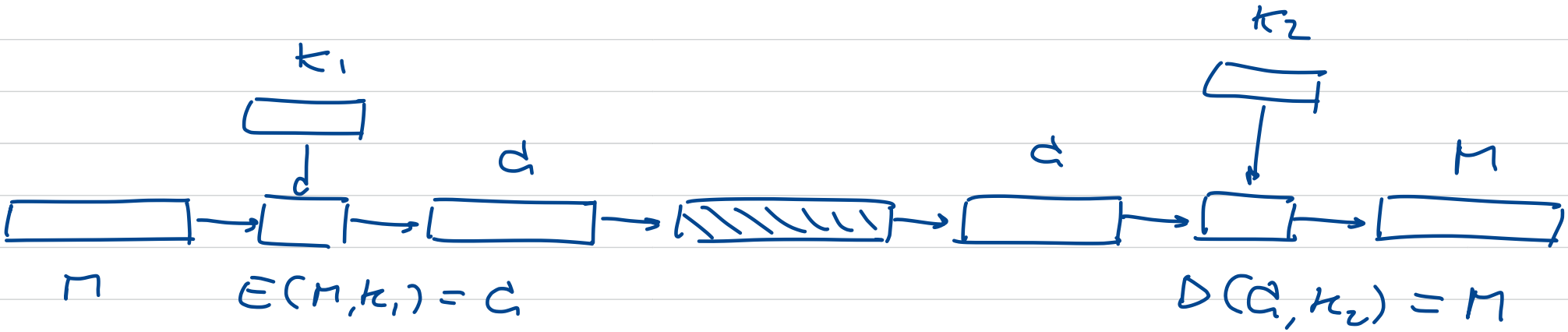
THE KEY K MUST BE EXCHANGED

ON A COMPLETELY SECURE

CHANNEL BEFORE TRANSMISSION

ASYMMETRIC (PUBLIC KEY) ENCRYPTION SCHEME

- A PUBLIC KEY K_1 USED FOR ENCRYPTION
- A PRIVATE KEY K_2 USED FOR DECRYPTION



EVERYONE HAS ACCESS TO K_1 (PUBLIC DIRECTORY, CERTIFICATE

BUT ONLY YOU HAS K_2 AND CAN DECRYPT THE MESSAGE

NO NEED TO PREVIOUSLY

EXCHANGE THE KEY

BEFORE TRANSMISSION

FOR EXAMPLE WE CAN COMBINE

- ASYMMETRIC FOR EXCHANGING

A KEY K BETWEEN USERS

- SYMMETRIC FOR ACTUAL

TRANSMISSION BY USING K

TRAPDOOR ONE-WAY FUNCTIONS

IT IS A FUNCTION VERY EASY TO BE COMPUTED, BUT VERY DIFFICULT TO BE INVERTED, UNLESS YOU HAVE SOME SIDE INFORMATION.

EXAMPLES:
- PRIME FACTORIZATION (RSA)
- DISCRETE LOGARITHM (DIFFIE HELLMAN)
- ELLIPTIC CURVES (CURVE)
- ERROR CORRECTING CODES (Mc ELIECE)

INTRODUCTION TO RSA (RIVEST. SHAMIR. ADLEMAN)

BASED ON PRIME FACTORIZATION

GIVEN 2 PRIME INTEGERS

IT IS EASY TO MULTIPLE THEN

$p \; q \rightarrow N = pq$

BUT GIVEN N IS VERY DIFFICULT

TO FACTORIZE IT AND IDENTIFY p AND q

GREATEST COMMON DIVISOR, COPRIME

A B INTEGERS

$gcd(A, B) \equiv$ LARGEST INTEGER THAT DIVIDES BOTH OF THEM

$gcd(10, 6) = 2$

A AND B ARE COPRIME

IF $gcd(A, B) = 1$

EULER PHI TOTIENT NUMBER

GIVEN A

$\phi(A)$ IS THE NUMBER OF INTEGERS $\leq A$ THAT ARE COPRIME WITH A

GIVEN A PRIME p

$$\phi(p) = p - 1$$

GIVEN TWO PRIME INTEGERS p AND q

$$\phi(pq) = (p-1)(q-1)$$

$p = 3 \qquad q = 2 \qquad n = pq = 3 \cdot 2 = 6$

$\phi(6) = 2 \qquad 1\ 2\ 3\ 4\ 5\ 6$

$\phi(6) = \phi(3 \cdot 2) = 2 \cdot 1 = 2$

PROPERTY

$A = B \mod N$ $\quad 0 \leq B \leq N-1$

B IS THE REMAINDER OF THE DIVISION BY N

$A = kN + B$

$\gcd(m, N) = 1$

$m^{\phi(N)} = 1 \mod N$

$N = 6 \quad \phi(N) = 2 \quad m = 5 \quad \gcd(5, 6) = 1$

$m^{\phi(N)} = 1 \mod N \qquad 5^2 = 25 = 1 \mod 6$

RSA: PUBLIC AND PRIVATE KEYS

(LARGE)

WE SELECT TWO PRIME INTEGERS p AND q

WE MULTIPLY THEM $N = p \cdot q$

WE COMPUTE $\phi(N) = (p-1)(q-1)$

WE CHOOSE e SUCH THAT

$$\gcd(e, \phi(N)) = 1$$

WE COMPUTE d SUCH THAT

$$e \cdot d = 1 \mod \phi(N)$$

(e, N) PUBLIC KEY

d PRIVATE KEY

RSA: ENCRYPTION AND DECRYPTION

GIVEN A MESSAGE WE REPRESENT IT AS

AN INTEGER $0 \leq m \leq N-1$

ENCRYPTION

WE REQUIRE PUBLIC KEY (e, N)

WE COMPUTE CIPHERTEXT

$$c = m^e \bmod N$$

DECRYPTION

WE TAKE PRIVATE KEY (d)

WE COMPUTE PLAINTEXT

$$m = c^d \bmod N$$

RSA: PROOF

$c^d \bmod N$ BUT $c = m^e$

SUPPOSE $\gcd(m, N) = 1$

(IF NOT $\gcd(m, p) = 1$ OR $\gcd(m, q) = 1$ → CHINESE REMAINDER THEOREM)

$m^{ed} \bmod N$

$e \cdot d = 1 \bmod \phi(N)$

$e \cdot d = k\phi(N) + 1$

$m^{k\phi(N)+1} = m^{k\phi(N)} \cdot m \bmod N = m$

$m^{k\phi(N)} = 1$

RSA toy example

$$p = 7 \quad q = 11 \quad N = pq = 77$$

$$\Phi(N) = (p-1)(q-1) = 6 \cdot 10 = 60$$

$$e = 17 \qquad gcd(e, \Phi(N)) = 1$$

$$d = 53 \qquad e \cdot d = 1 \mod \Phi(N)$$

17
53
51
85
301 | 60
1

$$m = 2$$

$$c = m^e mod \;\; N = 2^{17} mod \;\; 77 = 18$$

$$m = c^d mod \quad N = 18^{53} mod \quad 77 = 2$$

SECURITY OF RSA

THE SECURITY OF THE ALGORITHM

IS BASED ON THE DIFFICULTY

OF FACTORIZING N

WITHOUT N WE CANNOT COMPUTE $\phi(N)$

EVEN IF e IS PUBLIC WE

CANNOT COMPUTE d SUCH THAT

$$e \cdot d = 1 \mod \phi(N)$$

→ WE CANNOT DECRYPT THE MESSAGE

ONLY THE ORIGINAL USER HAS d

AND CAN DECRYPT IT

SHOR ALGORITHM

THERE IS AN ALGORITHM BASED ON QUANTUM COMPUTERS THAT ALLOW TO FACTORITE EVEN LARGE NUMBERS N

IF THIS HAPPENS → RSA IS NO MORE SECURE

SEARCH FOR NEW ALGORITHMS ABLE TO RESIST QUANTUM COMPUTERS ATTACKS

→ POST QUANTUM CRYPTOGRAPHY

NIST PQC

NIST Post-Quantum Cryptography Standardization – launched in 2016

59 encryption schemes submitted at the end of 2017

WIKIPEDIA

NIST Post-Quantum Cryptography Standardization

Post-Quantum Cryptography Standardization[1] is a program and competition by NIST to update their standards to include post-quantum cryptography.[2] It was announced at PQCrypto 2016.[3] 23 signature schemes and 59 encryption/KEM schemes were submitted by the initial submission deadline at the end of 2017[4] of which 69 total were deemed complete and proper and participated in the first round. Seven of these, of which 3 are signature schemes, have advanced to the third round, which was announced on July 22, 2020.

Contents

- **Background**
- **Round one**
  - Round one submissions published attacks
- **Round two**
- **Round three**
  - Finalists
  - Alternate candidates
  - Intellectual property concerns
  - Round three submissions published attacks
  - Adaptations
  - Selected Algorithms 2022
- **Round four**
  - Round four submissions published attacks
- **See also**
- **References**
- **External links**

Background

Academic research on the potential impact of quantum computing dates back to at least 2001.[5] A NIST published report from April 2016 cites experts that acknowledge the possibility of quantum technology to render the commonly used RSA algorithm insecure by 2030.[6] As a result, a need to standardize quantum-secure cryptographic primitives was pursued. Since most symmetric primitives are relatively easy to modify in a

way that makes them quantum resistant, efforts have focused on public-key cryptography, namely digital signatures and key encapsulation mechanisms. In December 2016 NIST initiated a standardization process by announcing a call for proposals.[7]

The competition is now in its third round out of expected four, where in each round some algorithms are discarded and others are studied more closely. NIST hopes to publish the standardization documents by 2024, but may speed up the process if major breakthroughs in quantum computing are made.

It is currently undecided whether the future standards be published as FIPS or as NIST Special Publication (SP).

Round one

Under consideration were:[8]
(~~strikethrough~~ means it had been withdrawn)

Type	PKE/KEM	Signature	Signature & PKE/KEM
Lattice	▪ Compact LWE ▪ CRYSTALS-Kyber ▪ Ding Key Exchange ▪ EMBLEM and R.EMBLEM ▪ FrodoKEM ▪ ~~HILA5~~ (withdrawn and merged into Round5) ▪ KCL (pka OKCN/AKCN/CNKE) ▪ KINDI ▪ LAC ▪ LIMA ▪ Lizard ▪ LOTUS ▪ NewHope ▪ NTRUEncrypt[9] ▪ NTRU-HRSS-KEM ▪ NTRU Prime ▪ Odd Manhattan ▪ ~~Round2~~ (withdrawn and merged into Round5) ▪ Round5 (merger of Round2 and Hila5, announced 4 August 2018)[10] ▪ SABER ▪ Three Bears ▪ Titanium	▪ CRYSTALS-Dilithium ▪ DRS ▪ FALCON ▪ pqNTRUSign[9] ▪ qTESLA	
Code-based	▪ BIG QUAKE ▪ BIKE ▪ Classic McEliece + NTS-KEM ▪ DAGS ▪ ~~Edon-K~~ ▪ HQC ▪ ~~LAKE~~ (withdrawn and merged into ROLLO) ▪ LEDAkem ▪ LEDApkc ▪ Lepton ▪ ~~LOCKER~~ (withdrawn and merged into ROLLO)	▪ pqsigRM ▪ RaCoSS ▪ ~~RankSign~~	

	▪ McNie ▪ NTS-KEM ▪ ROLLO (merger of Ouroboros-R, LAKE and LOCKER) [11] ▪ ~~Ouroboros-R~~ (withdrawn and merged into ROLLO) ▪ QC-MDPC KEM ▪ Ramstake ▪ RLCE-KEM ▪ RQC		
Hash-based		▪ Gravity-SPHINCS ▪ SPHINCS+	
Multivariate	▪ CFPKM ▪ Giophantus	▪ DualModeMS ▪ GeMSS ▪ Gui ▪ HiMQ-3 ▪ LUOV ▪ MQDSS ▪ Rainbow	▪ ~~SRTPI~~ ▪ DME
Braid group		▪ WalnutDSA	
Supersingular elliptic curve isogeny	▪ SIKE		
Satirical submission			▪ pqRSA[12][13]
Other	▪ Guess Again ▪ ~~HK17~~ ▪ Mersenne-756839 ▪ ~~RVB~~	▪ Picnic	

Round one submissions published attacks

- Guess Again by Lorenz Panny [14]
- RVB by Lorenz Panny[15]

- RaCoSS by Daniel J. Bernstein, Andreas Hülsing, Tanja Lange and Lorenz Panny[16]
- HK17 by Daniel J. Bernstein and Tanja Lange[17]
- SRTPI by Bo-Yin Yang[18]
- WalnutDSA
  - by Ward Beullens and Simon R. Blackburn[19]
  - by Matvei Kotov, Anton Menshov and Alexander Ushakov[20]
- DRS by Yang Yu and Léo Ducas [21]
- DAGS by Elise Barelli and Alain Couvreur[22]
- Edon-K by Matthieu Lequesne and Jean-Pierre Tillich[23]
- RLCE by Alain Couvreur, Matthieu Lequesne, and Jean-Pierre Tillich[24]
- Hila5 by Daniel J. Bernstein, Leon Groot Bruinderink, Tanja Lange and Lorenz Panny[25]
- Giophantus by Ward Beullens, Wouter Castryck and Frederik Vercauteren[26]
- RankSign by Thomas Debris-Alazard and Jean-Pierre Tillich [27]
- McNie by Philippe Gaborit;[28] Terry Shue Chien Lau and Chik How Tan [29]

Round two

Candidates moving on to the second round were announced on January 30, 2019. They are:[30]

Type	PKE/KEM	Signature
Lattice	▪ CRYSTALS-Kyber[31] ▪ FrodoKEM[32] ▪ LAC ▪ NewHope[33] ▪ NTRU (merger of NTRUEncrypt and NTRU-HRSS-KEM)[9] ▪ NTRU Prime[34] ▪ Round5 (merger of Round2 and Hila5, announced 4 August 2018)[10] ▪ SABER[35] ▪ Three Bears[36]	▪ CRYSTALS-Dilithium[31] ▪ FALCON[37] ▪ qTESLA[38]
Code-based	▪ BIKE[39] ▪ Classic McEliece ▪ HQC[40] ▪ LEDAcrypt (merger of LEDAkem[41] and LEDApkc[42]) ▪ NTS-KEM[43] ▪ ROLLO (merger of Ouroboros-R, LAKE and LOCKER) [11] ▪ RQC[44]	
Hash-based		▪ SPHINCS+[45]
Multivariate		▪ GeMSS[46] ▪ LUOV[47] ▪ MQDSS[48] ▪ Rainbow
Supersingular elliptic curve isogeny	▪ SIKE[49]	
Zero-knowledge proofs		▪ Picnic[50]

Round three

On July 22, 2020, NIST announced seven finalists ("first track"), as well as eight alternate algorithms ("second track"). The first track contains the algorithms which appear to have the most promise, and will be considered for standardization at the end of the third round. Algorithms in the second track could still become part of the standard, after the third round ends.[51] NIST expects some of the alternate candidates to be considered in a fourth round. NIST also suggests it may re-open the signature category for new schemes proposals in the future.[52]

On June 7–9, 2021, NIST conducted the third PQC standardization conference, virtually.[53] The conference included candidates' updates and discussions on implementations, on performances, and on security issues of the candidates. A small amount of focus was spent on intellectual property concerns.

Finalists

Type	PKE/KEM	Signature
Lattice	▪ CRYSTALS-Kyber ▪ NTRU ▪ SABER	▪ CRYSTALS-Dilithium ▪ FALCON
Code-based	▪ Classic McEliece	
Multivariate		▪ Rainbow

Alternate candidates

Type	PKE/KEM	Signature
Lattice	▪ FrodoKEM ▪ NTRU Prime	
Code-based	▪ BIKE (https://bikesuite.org/) ▪ HQC (https://pqc-hqc.org/)	
Hash-based		▪ SPHINCS+
Multivariate		▪ GeMSS

Supersingular elliptic curve isogeny	▪ SIKE	
Zero-knowledge proofs		▪ Picnic

Intellectual property concerns

After NIST's announcement regarding the finalists and the alternate candidates, various intellectual property concerns were voiced, notably surrounding lattice-based schemes such as Kyber and NewHope. NIST holds signed statements from submitting groups clearing any legal claims, but there is still a concern that third parties could raise claims. NIST claims that they will take such considerations into account while picking the winning algorithms.[54]

Round three submissions published attacks

- Rainbow: by Ward Beullens on a classical computer[55]

Adaptations

During this round, some candidates have shown to be vulnerable to some attack vectors. It forces this candidates to adapt accordingly:

CRYSTAL-Kyber and SABER
: may change the nested hashes used in their proposals in order for their security claims to hold.[56]

FALCON
: side channel attack by . A masking may be added in order to resist the attack. This adaptation affects performance and should be considered while standardizing.[57]

Selected Algorithms 2022

On July 5, 2022, NIST announced the first group of winners from its six-year competition.[58][59]

Type	PKE/KEM	Signature
Lattice	▪ CRYSTALS-Kyber	▪ CRYSTALS-Dilithium (https://pq-crystals.org/dilithium/index.shtml) ▪ FALCON (https://falcon-sign.info/)
Hash-based		▪ SPHINCS+ (https://sphincs.org/)

Round four

On July 5, 2022, NIST announced four candidates for PQC Standardization Round 4.[60]

Type	PKE/KEM
Code-based	▪ BIKE (https://bikesuite.org/) ▪ Classic McEliece (https://classic.mceliece.org/) ▪ HQC (https://pqc-hqc.org/)
Supersingular elliptic curve isogeny	▪ SIKE (https://sike.org/)

Round four submissions published attacks

- SIKE: by Wouter Castryck and Thomas Decru on a classical computer[61]

See also

- Advanced Encryption Standard process
- CAESAR Competition – Competition to design authenticated encryption schemes
- NIST hash function competition

References

1. "Post-Quantum Cryptography PQC" (https://csrc.nist.gov/projects/post-quantum-cryptography). 3 January 2017.

2. "Post-Quantum Cryptography Standardization – Post-Quantum Cryptography" (https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization). *Csrc.nist.gov*. 3 January 2017. Retrieved 31 January 2019.
3. Moody, Dustin (24 November 2020). "The Future Is Now: Spreading the Word About Post-Quantum Cryptography" (https://www.nist.gov/blogs/taking-measure/future-now-spreading-word-about-post-quantum-cryptography). *Nist*.
4. "Archived copy" (https://web.archive.org/web/20171229232437/https://post-quantum.ch/#). Archived from the original (https://post-quantum.ch/#) on 2017-12-29. Retrieved 2017-12-29.
5. Hong, Zhu (2001). "Survey of Computational Assumptions Used inCryptography Broken or Not by Shor's Algorithm" (http://crypto.cs.mcgill.ca/~crepeau/PDF/memoire-hong.pdf) (PDF).
6. "NIST Released NISTIR 8105, Report on Post-Quantum Cryptography" (https://csrc.nist.gov/News/2016/NIST-Released-NISTIR-8105,-Report-on-Post-Quantum). 21 December 2016. Retrieved 5 November 2019.
7. "NIST Asks Public to Help Future-Proof Electronic Information" (https://www.nist.gov/news-events/news/2016/12/nist-asks-public-help-future-proof-electronic-information). *Nist*. 20 December 2016. Retrieved 5 November 2019.
8. Computer Security Division, Information Technology Laboratory (3 January 2017). "Round 1 Submissions – Post-Quantum Cryptography – CSRC" (https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Round-1-Submissions). *Csrc.nist.gov*. Retrieved 31 January 2019.
9. "Archived copy" (https://web.archive.org/web/20171229114632/https://www.onboardsecurity.com/nist-post-quantum-crypto-submission). Archived from the original (https://www.onboardsecurity.com/nist-post-quantum-crypto-submission) on 2017-12-29. Retrieved 2017-12-29.
10. "Google Groups" (https://groups.google.com/a/list.nist.gov/forum/#!topic/pqc-forum/YsGkKEJTt5c). *Groups.google.com*. Retrieved 31 January 2019.
11. "ROLLO" (http://www.pqc-rollo.org/). *Pqc-rollo.org*. Retrieved 31 January 2019.
12. RSA using 2^{31} 4096-bit primes for a total key size of 1 TiB. "Key almost fits on a hard drive" Bernstein, Daniel (2010-05-28). "McBits and Post-Quantum RSA" (http://cr.yp.to/talks/2010.05.28/slides.pdf#page=29) (PDF). Retrieved 2019-12-10.
13. Bernstein, Daniel; Heninger, Nadia (2017-04-19). "Post-quantum RSA" (https://cr.yp.to/papers/pqrsa-20170419.pdf) (PDF). Retrieved 2019-12-10.
14. "Dear all, the following Python script quickly recovers the message from a given "Guess Again" ciphertext without knowledge of the private key" (https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/official-comments/guess-again-official-comment.pdf) (PDF). *Csrc.nist.gov*. Retrieved 30 January 2019.
15. Panny, Lorenz (25 December 2017). "Fast key recovery attack against the "RVB" submission to #NISTPQC: t …. Computes private from public key" (https://twitter.com/yx7__/status/945283780851400704). Twitter. Retrieved 31 January 2019.
16. "Comments on RaCoSS" (https://web.archive.org/web/20171226100156/https://helaas.org/racoss/). Archived from the original (https://helaas.org/racoss/) on 2017-12-26. Retrieved 2018-01-04.
17. "Comments on HK17" (https://web.archive.org/web/20180105070112/https://helaas.org/hk17/). Archived from the original (https://helaas.org/hk17/) on 2018-01-05. Retrieved 2018-01-04.
18. "Dear all, We have broken SRTPI under CPA and TPSig under KMA" (https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/official-comments/SRTPI-official-comment.pdf) (PDF). *Csrc.nist.gov*. Retrieved 30 January 2019.
19. Beullens, Ward; Blackburn, Simon R. (2018). "Practical attacks against the Walnut digital signature scheme" (https://eprint.iacr.org/2018/318). *Cryptology ePrint Archive*.
20. Kotov, Matvei; Menshov, Anton; Ushakov, Alexander (2018). "AN ATTACK ON THE WALNUT DIGITAL SIGNATURE ALGORITHM" (https://eprint.iacr.org/2018/393). *Cryptology ePrint Archive*.

21. Yu, Yang; Ducas, Léo (2018). "Learning strikes again: the case of the DRS signature scheme" (https://eprint.iacr.org/2018/294). *Cryptology ePrint Archive*.
22. Barelli, Elise; Couvreur, Alain (2018). "An efficient structural attack on NIST submission DAGS". arXiv:1805.05429 (https://arxiv.org/abs/1805.05429) [cs.CR (https://arxiv.org/archive/cs.CR)].
23. Lequesne, Matthieu; Tillich, Jean-Pierre (2018). "Attack on the Edon-K Key Encapsulation Mechanism". arXiv:1802.06157 (https://arxiv.org/abs/1802.06157) [cs.CR (https://arxiv.org/archive/cs.CR)].
24. Couvreur, Alain; Lequesne, Matthieu; Tillich, Jean-Pierre (2018). "Recovering short secret keys of RLCE in polynomial time". arXiv:1805.11489 (https://arxiv.org/abs/1805.11489) [cs.CR (https://arxiv.org/archive/cs.CR)].
25. Bernstein, Daniel J.; Groot Bruinderink, Leon; Lange, Tanja; Lange, Lorenz (2017). "Hila5 Pindakaas: On the CCA security of lattice-based encryption with error correction" (https://eprint.iacr.org/2017/1214). *Cryptology ePrint Archive*.
26. "Official Comments" (https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/official-comments/Giophantus-official-comment.pdf) (PDF). *Csrc.nist.gov*. 13 September 2018.
27. Debris-Alazard, Thomas; Tillich, Jean-Pierre (2018). "Two attacks on rank metric code-based schemes: RankSign and an Identity-Based-Encryption scheme". arXiv:1804.02556 (https://arxiv.org/abs/1804.02556) [cs.CR (https://arxiv.org/archive/cs.CR)].
28. "I am afraid the parameters in this proposal have at most 4 to 6-bits security under the Information Set Decoding (ISD) attack" (https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/round-1/official-comments/McNie-official-comment.pdf) (PDF). *Csrc.nist.gov*. Retrieved 30 January 2019.
29. Lau, Terry Shue Chien; Tan, Chik How (31 January 2019). "Key Recovery Attack on McNie Based on Low Rank Parity Check Codes and Its Reparation". In Inomata, Atsuo; Yasuda, Kan (eds.). *Advances in Information and Computer Security*. Lecture Notes in Computer Science. Vol. 11049. Springer International Publishing. pp. 19–34. doi:10.1007/978-3-319-97916-8_2 (https://doi.org/10.1007%2F978-3-319-97916-8_2). ISBN 978-3-319-97915-1.
30. Computer Security Division, Information Technology Laboratory (3 January 2017). "Round 2 Submissions – Post-Quantum Cryptography – CSRC" (https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions). *Csrc.nist.gov*. Retrieved 31 January 2019.
31. Schwabe, Peter. "CRYSTALS" (https://pq-crystals.org/). *Pq-crystals.org*. Retrieved 31 January 2019.
32. "FrodoKEM" (https://frodokem.org/). *Frodokem.org*. Retrieved 31 January 2019.
33. Schwabe, Peter. "NewHope" (https://newhopecrypto.org/). *Newhopecrypto.org*. Retrieved 31 January 2019.
34. "NTRU Prime: Intro" (https://web.archive.org/web/20190901185114/https://ntruprime.cr.yp.to/). Archived from the original (https://ntruprime.cr.yp.to/) on 2019-09-01. Retrieved 2019-01-30.
35. "SABER" (https://www.esat.kuleuven.be/cosic/pqcrypto/saber/index.html). Retrieved 17 June 2019.
36. "ThreeBears" (https://sourceforge.net/projects/threebears/). *SourceForge.net*. Retrieved 31 January 2019.
37. "Falcon" (https://falcon-sign.info/). *Falcon*. Retrieved 26 June 2019.
38. "qTESLA – Efficient and post-quantum secure lattice-based signature scheme" (https://qtesla.org/). Retrieved 31 January 2019.
39. "BIKE – Bit Flipping Key Encapsulation" (https://bikesuite.org/). *Bikesuite.org*. Retrieved 31 January 2019.
40. "HQC" (https://pqc-hqc.org/). *Pqc-hqc.org*. Retrieved 31 January 2019.
41. "LEDAkem Key Encapsulation Module" (https://www.ledacrypt.org/LEDAkem/). *Ledacrypt.org*. Retrieved 31 January 2019.

42. "LEDApkc Public Key Cryptosystem" (https://www.ledacrypt.org/LEDApkc/). *Ledacrypt.org*. Retrieved 31 January 2019.
43. "NTS-Kem" (https://web.archive.org/web/20171229103229/https://nts-kem.io/). Archived from the original (https://nts-kem.io/) on 2017-12-29. Retrieved 2017-12-29.
44. "RQC" (http://pqc-rqc.org/). *Pqc-rqc.org*. Retrieved 31 January 2019.
45. https://sphincs.org/
46. "GeMSS" (https://web.archive.org/web/20190131040055/https://www-polsys.lip6.fr/Links/NIST/GeMSS.html). Archived from the original (https://www-polsys.lip6.fr/Links/NIST/GeMSS.html) on 2019-01-31. Retrieved 2019-01-30.
47. "LUOV -- An MQ signature scheme" (https://www.esat.kuleuven.be/cosic/pqcrypto/luov/). Retrieved 22 January 2020.
48. "MQDSS post-quantum signature" (http://mqdss.org/). *Mqdss.org*. Retrieved 31 January 2019.
49. "SIKE – Supersingular Isogeny Key Encapsulation" (https://sike.org/). *Sike.org*. Retrieved 31 January 2019.
50. "Picnic. A Family of Post-Quantum Secure Digital Signature Algorithms" (https://microsoft.github.io/Picnic/). *microsoft.github.io*. Retrieved 26 February 2019.
51. Moody, Dustin; Alagic, Gorjan; Apon, Daniel C.; Cooper, David A.; Dang, Quynh H.; Kelsey, John M.; Liu, Yi-Kai; Miller, Carl A.; Peralta, Rene C.; Perlner, Ray A.; Robinson, Angela Y.; Smith-Tone, Daniel C.; Alperin-Sheriff, Jacob (2020). "Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process" (https://csrc.nist.gov/publications/detail/nistir/8309/final). doi:10.6028/NIST.IR.8309 (https://doi.org/10.6028%2FNIST.IR.8309). S2CID 243755462 (https://api.semanticscholar.org/CorpusID:243755462). Retrieved 2020-07-23.
52. *Third PQC Standardization Conference - Session I Welcome/Candidate Updates* (https://www.nist.gov/video/third-pqc-standardization-conference-session-i-welcomecandidate-updates), retrieved 2021-07-06
53. Computer Security Division, Information Technology Laboratory (2021-02-10). "Third PQC Standardization Conference | CSRC" (https://csrc.nist.gov/Events/2021/third-pqc-standardization-conference). *CSRC | NIST*. Retrieved 2021-07-06.
54. "Submission Requirements and Evaluation Criteria" (https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf) (PDF).
55. Beullens, Ward (2022). "Breaking Rainbow Takes a Weekend on a Laptop" (https://eprint.iacr.org/2022/214.pdf) (PDF). *Eprint.iacr.org*.
56. Grubbs, Paul; Maram, Varun; Paterson, Kenneth G. (2021). "Anonymous, Robust Post-Quantum Public Key Encryption" (https://eprint.iacr.org/2021/708). *Cryptology ePrint Archive*.
57. Karabulut, Emre; Aysu, Aydin (2021). "Falcon Down: Breaking Falcon Post-Quantum Signature Scheme through Side-Channel Attacks" (https://eprint.iacr.org/2021/772). *Cryptology ePrint Archive*.
58. "NIST Announces First Four Quantum-Resistant Cryptographic Algorithms" (https://www.nist.gov/news-events/news/2022/07/nist-announces-first-four-quantum-resistant-cryptographic-algorithms). *NIST*. 2022-07-05. Retrieved 2022-07-09.
59. "Selected Algorithms 2022" (https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022). *CSRC | NIST*. 2022-07-05. Retrieved 2022-07-09.
60. "Round 4 Submissions" (https://csrc.nist.gov/Projects/post-quantum-cryptography/round-4-submissions). *CSRC | NIST*. 2022-07-05. Retrieved 2022-07-09.
61. Goodin, Dan. "Post-quantum encryption contender is taken out by single-core PC and 1 hour" (https://arstechnica.com/information-technology/2022/08/sike-once-a-post-quantum-encryption-contender-is-koed-in-nist-smackdown/). *Ars Technica*. Retrieved 6 August 2022.

External links

- NIST's official Website on the standardization process (https://csrc.nist.gov/Projects/Post-Quantum-Cryptography)
- Post-quantum cryptography website (https://pqcrypto.org/) by djb

Retrieved from "https://en.wikipedia.org/w/index.php?title=NIST_Post-Quantum_Cryptography_Standardization&oldid=1123971608"

This page was last edited on 26 November 2022, at 18:18 (UTC).

Text is available under the Creative Commons Attribution-ShareAlike License 3.0; additional terms may apply. By using this site, you agree to the Terms of Use and Privacy Policy. Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.

Round four [edit]

On July 5, 2022, NIST announced four candidates for PQC Standardization Round 4.[60]

Type	PKE/KEM
Code-based	• BIKE • Classic McEliece • HQC
Supersingular elliptic curve isogeny	• SIKE

INTRODUCTION TO Mc ELIECE SCHEME

THE Mc ELIECE IS ONE OF THE (TOGETHER WITH ELLIPTIC CURVES) CANDIDATE SCHEMES CONSIDERED FOR POST QUANTUM CRYPTOGRAPHY (ABLE TO RESIST AN ATTACK BASED ON QUANTUM COMPUTERS)

IT ALSO IS A BRIDGE BTW INF. THEORY, CODES AND CRYPTOGRAPHY

BINARY ALPHABET AND OPERATIONS

$F = \{0, 1\}$

+	0	1
0	0	1
1	1	0

·	0	1
0	0	0
1	0	1

F IS A FIELD

BINARY VECTORS

$$F^k = \{ \underline{v} = (v_1, \ldots, v_i, \ldots, v_k) \quad v_i \in F = \{0,1\} \}$$

$$F^2 = \{ (00)\ (01)\ (10)\ (11) \}$$

$$F^3 = \{ (000)\ (001) \quad - - - \}$$

$$|F^k| = 2^k$$

OPERATIONS

$\underline{v} = (v_1 \; v_2 \; v_3) \in F^3$

$\underline{w} = (w_1 \; w_2 \; w_3) \in F^3$

$\underline{v} + \underline{w} = (v_1+w_1 \;\; v_2+w_2 \;\; v_3+w_3) \in F^3$

$\underline{v} = (0\,1\,0)$

$\underline{w} = (1\,1\,0)$

$\underline{v} + \underline{w} = 1\,0\,0$

$a \in F$

$\underline{v} = (v_1 \; v_2 \; v_3) \in F^3$

$a \cdot \underline{v} = (av_1 \;\; av_2 \;\; av_3) \in F^3$

$\underline{v} = (0\,1\,0)$

$a = 1$

$a \cdot \underline{v} = (0\,1\,0)$

$a = 0$

$a \cdot \underline{v} = (0\,0\,0)$

STRUCTURE

F^k IS A VECTOR SPACE (IN PARTICULAR IT IS A GROUP)

$\underline{0} = (\underbrace{0 \cdots 0 \cdots 0}_{k \text{ BITS}})$ ALL ZERO VECTOR

$\forall \underline{v} \in F^k \quad \underline{v} + \underline{0} = \underline{v}$ IDENTITY

$\forall \underline{v} \quad \underline{v} + \underline{v} = \underline{0}$ INVERSE

$\underline{v}_1 + (\underline{v}_2 + \underline{v}_3) = (\underline{v}_1 + \underline{v}_2) + \underline{v}_3$ ASSOCIATIVE

$\underline{v}_1 + \underline{v}_2 = \underline{v}_2 + \underline{v}_1$ COMMUTATIVE

$\forall \underline{v}_1 \underline{v}_2 \quad \underline{v}_1 + \underline{v}_2 \in F^k$ CLOSURE

LINEAR ENCODING

WE START FROM A MESSAGE

$\underline{u} \in F^k$ OF k BITS

WE ADD SOME REDUNDANCY

INSTEAD OF TRANSMITTING k BITS

WE TRANSMIT $n > k$ BITS

$\underline{c} \in F^n$ OF n BITS

$n - k = r$ #OF REDUNDANCY BITS

THIS REDUNDANCY CAN BE EXPLOITED

TO DETECT / CORRECT ERRORS

$k = 2$

F^k

00

01

10

11

SUPPOSE THE SECOND BIT IS RECEIVED WRONG

TX 01 → RX 11

↑

YOU CANNOT DETECT OR CORRECT IT

WE INTRODUCE SOME REDUNDANCY: PARITY CODE

00 → 000

01 → 011

10 → 101

11 → 110

01 → 011 → 111

THE ERROR IS DETECTED

ERROR CORRECTION: REPETITION CODE

$0 \rightarrow 000$.

$1 \rightarrow 111$.

$0 \rightarrow 000 \rightarrow 100 \rightarrow 000 \rightarrow 0$

$d_H(100, 000) = 1$ ← CLOSEST

$d_H(100, 111) = 2$

How to add redundancy?

Systematic and linear encoding

$k = 3$

$\underline{v} = (v_1\ v_2\ v_3)$

$n = 6$

$\underline{c} = (c_1\ c_2\ c_3\ c_4\ c_5\ c_6)$

k | $c_1 = v_1$, $c_2 = v_2$, $c_3 = v_3$

r | $c_4 = v_1 + v_2$, $c_5 = v_1 + v_3$, $c_6 = v_2 + v_3$

$v_1\ v_2\ v_3$

$\underline{v} = (1\ 1\ 0)$

$\underline{c} = (1\ 1\ 0\ 0\ 1\ 1)$

$$
\begin{aligned}
c_1 &= v_1 \\
c_2 &= v_2 \\
c_3 &= v_3
\end{aligned}
$$

$$
\begin{aligned}
c_4 &= v_1 + v_2 \\
c_5 &= v_1 + v_3 \\
c_6 &= v_2 + v_3
\end{aligned}
$$

G $[k \times n]$ matrix (3×6)

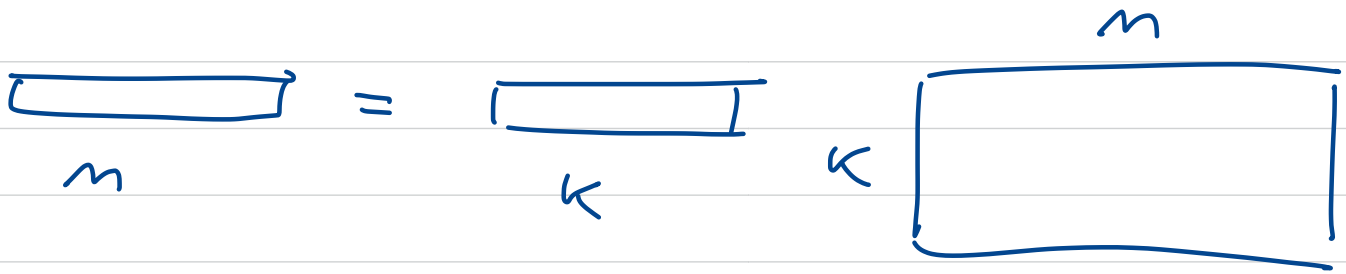
$$
G = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array}
\begin{array}{c}
\begin{array}{cccccc} c_1 & c_2 & c_3 & c_4 & c_5 & c_6 \end{array} \\
\begin{bmatrix}
1 & 0 & 0 & 1 & 1 & 0 \\
0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 1
\end{bmatrix}
\end{array}
$$

GENERATOR MATRIX

$\underline{v} \in F^k \equiv$ INFORMATION VECTOR

$\underline{c} \in F^n \equiv$ CODED VECTOR $\equiv$ CODEWORD

$$\underline{c} = \underline{v}\, G$$



$\underline{v} = (1\,1\,0)$

$\underline{c} = (1\,1\,0\,0\,1\,1)$

$$(1\,1\,0) \begin{matrix} 1 \\ 1 \\ 0 \end{matrix} \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

$$1\,1\,0\,0\,1\,1$$

DIFFERENT INF. VECTORS

ARE MAPPED INTO DIFFERENT

CODEWORDS

THE GENERATOR MATRIX HAS

FULL RANK (THE K ROWS

ARE LINEARLY

INDEPENDENT)

ONE-TO-ONE MAPPING

SINCE THE MAP $\underline{U} \rightarrow \underline{C}$

IS ONE-TO-ONE

THERE ARE 2^k DIFFERENT

CODEWORDS

CODE C IS THE SET

OF THESE 2^k CODEWORDS

NON-SYSTEMATIC GENERATOR MATRIX

$$G = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

G CAN BE NON-SYSTEMATIC

WHAT IS REQUIRED IS THE

FULL RANK → WE STILL HAVE

ONE-TO-ONE MAPPING

$$G = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

$k = 2$

$n = 5$

$F^k = F^2$

$00 \rightarrow 00000$

$10 \rightarrow 10111$

$01 \rightarrow 11100$

$11 \rightarrow 01011$

$$\begin{array}{l} 00 \\ 0 \\ 0 \end{array} \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

00000

$$\begin{array}{l} 1 \\ 0 \end{array} \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

HAMMING WEIGHT AND DISTANCE

$\underline{c}$ $w_H(\underline{c}) \equiv$ # BITS EQUAL TO 1

$$w_H(10111) = 4$$

$\underline{c_1}$ $\underline{c_2}$ $d_H(\underline{c_1}\ \underline{c_2}) \equiv$ # BITS DIFFERENT

$$d_H(10111, 11100) = 3$$

$$d_H(\underline{c_1}, \underline{c_2}) = w_H(\underline{c_1} + \underline{c_2})$$

$$\begin{array}{r} 10111 + \\ 11100 \\ \hline 01011 \end{array}$$

MINIMUM DISTANCE

$$d_{min} = \min_{\substack{\underline{c} \in \mathcal{C} \\ \underline{c} \neq \underline{0}}} w_H(\underline{c})$$

	w_H
$00 \rightarrow 00000$	0
$10 \rightarrow 10111$	4
$01 \rightarrow 11100$	3
$11 \rightarrow 01011$	3

$$d_{min} = 3$$

ADDED : NOT FOR EXAM

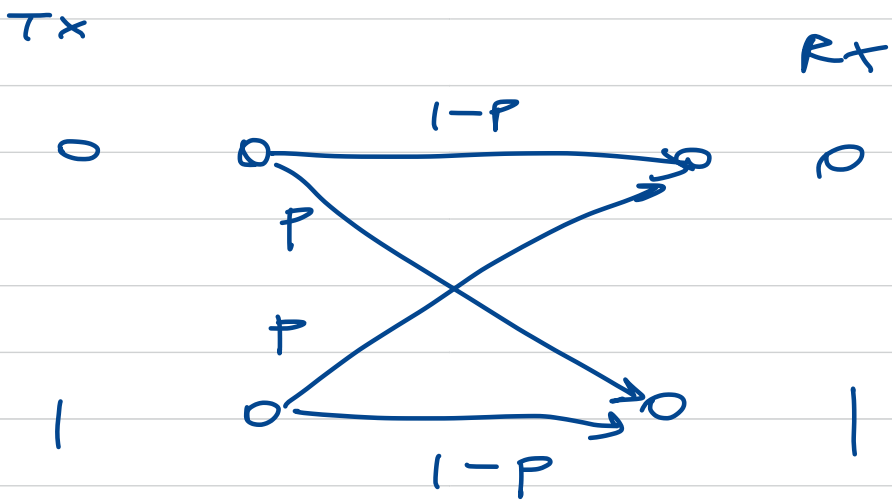
d_{min} IS INDEPENDENT FROM THE CODEWORD TAKEN AS REFERENCE

$$d_{min} = \min_{\underline{c} \in \mathcal{C}} w_H(\underline{c}) = \min_{\underline{c} \in \mathcal{C}} d_H(\underline{c}\ \underline{0})$$

$$= \min_{\underline{c} \in \mathcal{C}} d_H(\underline{c} + \underline{c}^*, \underline{c}^*)$$

$$= \min_{\underline{c}' \in \mathcal{C}} d_H(\underline{c}', \underline{c}^*)$$

BINARY SYMMETRIC CHANNEL



WHEN WE HAVE AN ERROR $0 \rightarrow 1$

$1 \rightarrow 0$

THIS IS EQUIVALENT TO SUM 1 $0+1=1$

$1+1=0$

IMPACT OF ERROR VECTOR

SUPPOSE WE TX A CODEWORD $\underline{c}$ AND ONE OF THE BITS ARE RECEIVED WRONG

$\underline{c} \rightarrow \underline{y} \neq \underline{c}$

$\underline{c} = (111)$

$\underline{y} = (110)$

$\underline{y} = \underline{c} + \underline{e}$

$\underline{e} = (001)$

$\underline{e} = (e_1 \quad e_i \quad e_n)$

$$e_i = \begin{cases} 0 & \text{IF } y_i = c_i \\ 1 & \text{IF } y_i \neq c_i \end{cases}$$

$\underline{c} = (011)$

$\underline{y} = (101)$

$\underline{e} = (110)$

ERROR VECTORS

$\underline{e} = (0 \; 0 \; \cdots \; 0)$ $\quad w_H(\underline{e}) = 0$

$\underline{e} = (1 \; 0 \; \cdots \; 0)$

$(0 \; 1 \; \cdots \; 0)$ $\quad w_H(\underline{e}) = 1$

$(0 \; 0 \; \cdots \; 1)$

MINIMUM DISTANCE DECODING RULE

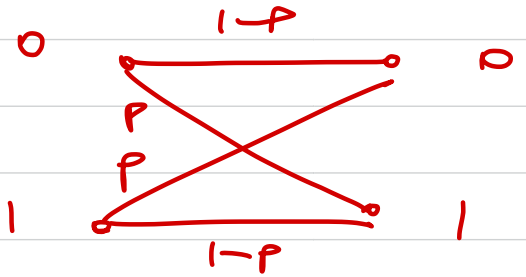
$$\underline{c} \rightarrow \underline{y} = \underline{c} + \underline{e}$$

WE WANT TO CORRECT THE ERRORS

CHOOSE

$$\underline{c_R} = \arg \min_{\underline{c} \in C} \left[d_H(\underline{y}, \underline{c}) \right]$$

ADDED: NOT FOR EXAM



$$\underline{y} = \underline{c}_T + \underline{e}$$

$$\underline{c}_R = \arg\max_{\underline{c} \in \mathcal{C}} P(\underline{c}_T = \underline{c} \mid \underline{y}) = \frac{P(\underline{y} \mid \underline{c}_T = \underline{c}) P(\underline{c}_T = \underline{c})}{P(\underline{y})}$$

$$= \arg\max_{\underline{c} \in \mathcal{C}} P(\underline{y} \mid \underline{c}_T = \underline{c}) = p^{d_H(\underline{y}, \underline{c})} (1-p)^{n - d_H(\underline{y}, \underline{c})}$$

$$= (1-p)^n \left(\frac{p}{1-p}\right)^{d_H(\underline{y}, \underline{c})}$$

$$p < 1/2 \rightarrow \left(\frac{p}{1-p}\right) < 1$$

$$\underline{c}_R = \arg\min_{\underline{c} \in \mathcal{C}} d_H(\underline{y}, \underline{c})$$

ERROR CORRECTION CAPABILITY

BY APPLYING THIS RULE

ACC $\underline{e}$ WITH HAMMING WEIGHT

$$w(\underline{e}) \leq t = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor$$

ARE CERTAINLY CORRECTED

d_{min}	t
3	1
5	2
7	3

ADDED: NOT FOR EXAM

WE TX c
WE RX y WITH

$$d_H(y, c) = a \leq t$$

SUPPOSE WE INSTEAD CHOOSE c'

$$d_H(y, c') = b \leq a$$

$$d_H(c, c') \leq a + b \leq 2a \leq 2t$$

$$< d_{MIN}$$

IMPOSSIBLE

COMPLEXITY ISSUE

PROBLEM: WHEN WE APPLY MINIMUM DISTANCE RULE

WE MUST COMPARE y AGAINST ALL POSSIBLE CODEWORDS

→ THERE ARE 2^k CODEWORDS

IMPOSSIBLE IF k IS NOT SMALL

WE NEED ALTERNATIVE DECODING ALGORITHMS

WE PRESENT SYNDROME DECODING

PARITY CHECK MATRIX

$$c_1 = v_1$$
$$c_2 = v_2$$
$$c_3 = v_3$$

$$\left.\begin{aligned} c_4 &= v_1 + v_2 \\ c_5 &= v_1 + v_3 \\ c_6 &= v_2 + v_3 \end{aligned}\right\}$$

$$c_4 = c_1 + c_2$$
$$c_5 = c_1 + c_3$$
$$c_6 = c_2 + c_3$$

$$\begin{cases} 0 = c_1 + c_2 + c_4 \\ 0 = c_1 + c_3 + c_5 \\ 0 = c_2 + c_3 + c_6 \end{cases}$$

$$\begin{cases} 0 = c_1 + c_2 + c_4 \\ 0 = c_1 + c_3 + c_5 \\ 0 = c_2 + c_3 + c_6 \end{cases}$$

$t = n - k$ $\qquad$ n

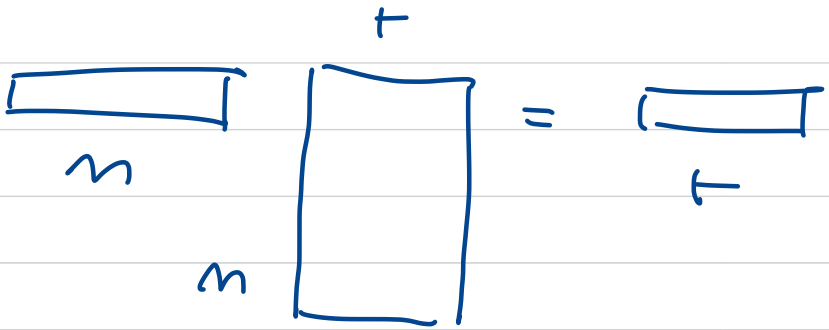
PARITY CHECK MATRIX

H

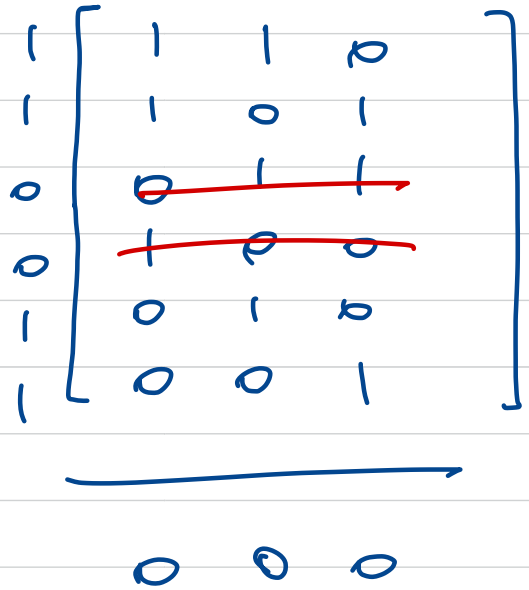
$n \times t$

	I	II	III
c_1	1	1	0
c_2	1	0	1
c_3	0	1	1
c_4	1	0	0
c_5	0	1	0
c_6	0	0	1

$\underline{c}\, H = \underline{0}$



1 1 0 0 1 1



$\underline{c}\ H = \underline{0}$

$G = [\ \equiv\ ]$ $\qquad$ $G H = \underline{0}$

ALTERNATIVE DEFINITION FOR H

IT IS AN $m \times t$ BINARY MATRIX

WITH FULL RANK SUCH THAT

$$G \cdot H = \underline{\underline{0}}$$

SYNDROME

FOCUS ON THE MATRIX H

FOR EACH CODEWORD $\underline{c}$

$$\underline{c} H = \underline{0}$$

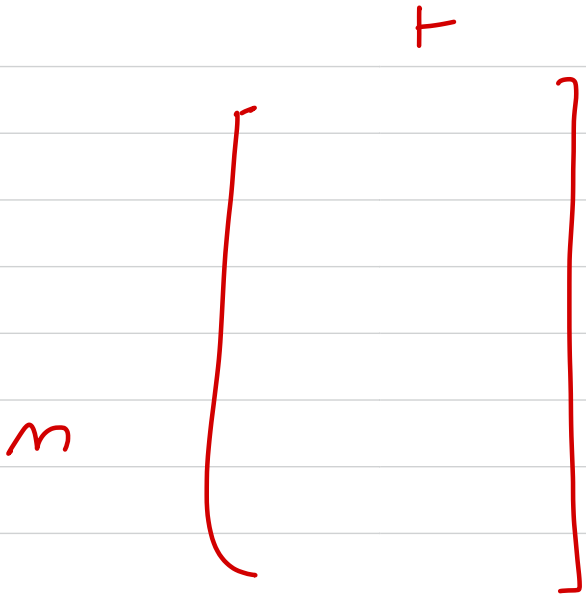
$$\underline{c} \in F^{n}$$

SYNDROME

$\underline{y} \in F^n$ $\quad\quad$ $\underline{y} H = \underline{s}$

PROPERTY: THE SYNDROME $\underline{s} = \underline{0}$

IF AND ONLY IF $\underline{y} \in C$



$$n = \dim(\text{KERNEL}) + \dim(\text{Im})$$

$$\dim(\text{Im}) = \text{RANK}(A)$$

K

SYNDROME AND ERROR VECTORS

WE TX $\underline{c}$

WE RX $\underline{y} = \underline{c} + \underline{e}$

COMPUTE SYNDROME $\underline{s} = \underline{y}\,H =$

$= \underline{c}H + \underline{e}\,H = \underline{e}\,H$

$\underline{0}$

THE SYNDROME OF THE RX VECTOR $\underline{y}$

ONLY DEPENDS ON ERROR VECTOR $\underline{e}$

ALL $\underline{e}$ WITH $w(\underline{e}) \leq t$

HAVE DIFFERENT SYNDROMES

SIMPLE DECODING ALGORITHM:

SYNDROME DECODING

ADDED: NOT FOR EXAM

$\underline{e}_1 \quad w(\underline{e}_1) \leq t$

$\underline{e}_2 \quad w(\underline{e}_2) \leq t$

$\underline{s}_1 = \underline{s}_2$ IMPOSSIBLE

IN FACT $\underline{e}_1 H = \underline{e}_2 H \rightarrow (\underline{e}_1 + \underline{e}_2) H = \underline{0}$

$\rightarrow \quad \underline{e}_1 + \underline{e}_2 \in \mathcal{C} \rightarrow$

$w_H(\underline{e}_1 + \underline{e}_2) \leq 2t < d_{MIN}$

IMPOSSIBLE

BEFORE STARTING TRANSMISSION

WE TAKE ALL $\underline{e}$ $w_H(\underline{e}) \in t$

AND FOR EACH OF THEM

WE COMPUTE $\underline{S} = \underline{e} H$

AND WE STORE IT INTO A LOOKUP TABLE

$\underline{S}$	$\underline{e}$

WE RECEIVE $\underline{y} = \underline{c} + \underline{e}$

WE COMPUTE $\underline{s} = \underline{y} H = \underline{e} H$

LUT $\underline{s} \rightarrow \underline{e}$

$\underline{y} + \underline{e} = \underline{c}$

$k = 2 \qquad n = 5$

$$G = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

$$H = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

$\underline{v}$	$\underline{c}$	w_H
$00 \rightarrow$	$0\ 0\ 0\ 0\ 0$	0
$10 \rightarrow$	$1\ 0\ 1\ 1\ 1$	4
$01 \rightarrow$	$1\ 1\ 1\ 0\ 0$	3
$11 \rightarrow$	$0\ 1\ 0\ 1\ 1$	3

$d_{min} = 3$

$$G H = \underline{0}$$

$$t = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor = 1$$

$$w_H(\underline{e}) \leq t = 1$$

$$H = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

$\underline{s} = \underline{e}\, H$

$w_H(\underline{e}) \leq 1$

$\underline{e}$	$\underline{s}$
00000	000
10000	011
01000	001
00100	010
00010	101
00001	100

e	s
00000	000
10000	011
01000	001
00100	010
00010	101
00001	100

$\underline{c} = 10111$ *

$\underline{e} = 00001$

$\underline{y} = \underline{c} + \underline{e} = 10110$

$\underline{s} = \underline{y}\,H$

$\underline{y} + \underline{e} = \begin{matrix} 10110 \\ 00001 \end{matrix} = 10111$ *

1	011
0	001
1	010
1	101
0	100

$100 = \underline{s}$

$\underline{e} = 00001$

Mc ELIECE CRYPTO SYSTEM

USER SELECTS

A GENERATOR MATRIX G — k ROWS, n COLUMNS

IT COMPUTES

THE PARITY CHECK MATRIX H — n ROWS, $r = n - k$ COLUMNS

$$GH = \underline{\underline{0}}$$

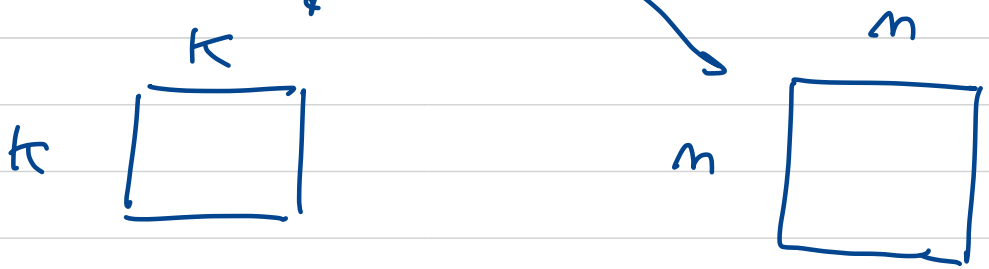
IT COMPUTES d_{min}

AND $$t = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor$$

INSTEAD OF PUBLISHING G

THE USER COMPUTES

$$G_1 = S G P$$



FULL RANK
(NON-SINGULAR)

PERMUTATION
MATRIX

GIVEN G_1 IT'S VERY DIFFICULT TO

COMPUTE G

PUBLIC KEY

(G, t)

ENCRYPTION

WE RECOVER PUBLIC KEY (G_1, t)

NOTE: G_1 IS A $k \times n$ BINARY MATRIX

WE REPRESENT OUR MESSAGE AS A BINARY VECTOR OF k BITS

$$\underline{v} \in F^k$$

WE COMPUTE $\underline{c} = \underline{v} G_1 \in F^n$

WE RANDOMLY EXTRACT $\underline{e}$ $w_H(e) \leq t$

CIPHERTEXT $\underline{y} = \underline{c} + \underline{e}$

DECRYPTION

USER RECEIVES $\underline{Y} = \underline{C} + \underline{e}$

IT KNOWS THAT $G_1 = S G P$

$\underline{Y} \rightarrow \underline{Y_1} = P^{-1} \underline{Y}$

IT DECODES $\underline{Y_1}$ → IT APPLIES A DECODING ALGORITHM (FOR EXAMPLE SYNDROME DECODING) TO CORRECT THE ERROR AND RECOVER THE CODEWORD

$\underline{Y_1} \xrightarrow{\text{DECODING}} \underline{C_1}$

$$G_1 = S G P$$

GIVEN $\underline{C_1}$ IT RECOVERS $\underline{U_1}$

BY SOLVING

$$\underline{U_1} G = \underline{C_1}$$

GIVEN $\underline{U_1}$ IT RECOVERS THE ORIGINAL MESSAGE AS

$$\underline{V} = \underline{U_1} S^{-1}$$

IMPORTANT

ONLY THE LEGITIM USER

CAN DECODE $\underline{y}$ BECAUSE

IT KNOWS S, G, P AND H

WHICH ARE NEEDED TO

RECOVER $\underline{s}$

FROM THE PUBLIC KEY $G_1 = SGP$

IT'S IMPOSSIBLE TO RECOVER THEM

CURRENTLY, NO ALGORITHMS BASED ON QUANTUM COMPUTERS ABLE TO BREAK THE Mc ELIECE SCHEME ARE KNOWN

→ Mc ELIECE SCHEME IS ONE OF THE 2 CANDIDATES FOR POST QUANTUM CRYPTOGRAPHY

McEliece toy example

$$k = 2 \quad n = 5$$

$$G = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix} \qquad H = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

v	c	$w_H(c)$
$00 \to$	00000	0
$10 \to$	10111	4
$01 \to$	11100	3
$11 \to$	01011	3

$d_{min} = 3$ $\qquad t = \left\lfloor \frac{d_{min}-1}{2} \right\rfloor = 1$

$$S = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

$P^{-1} = P^t$

$$G_1 = \underbrace{SGP} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{bmatrix} \qquad t_1 = 1$$

PUBLIC KEY

$\underline{v} \in F^k \qquad k = 2$

$$\underline{v} = 11$$

$$\underline{c} = \underline{v} G_1 = 10101$$

$$\underline{e} = 00100$$

$$\underline{y} = \underline{c} + \underline{e} = 10001$$

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \qquad P^{-1} = P^t = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

(14253) (13524)

$$S = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \qquad S^{-1} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$$

Matlab

```
inv(gf(S))
```

$$\underline{y}_1 = \underline{y}P^{-1} = 10100$$

$$H = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

$\underline{e}$ $\qquad\qquad$ $\underline{s} = \underline{e}H$

LUT $\quad w(e)$ = 1

$$w(e)=0 \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

$$\underline{y}_1 = 10100$$

$$\underline{s} = \underline{y}_1 H = 001$$

$$\underline{\hat{e}} = 01000$$

$$\underline{c}_1 = \underline{y}_1 + \underline{\hat{e}} = 11100$$

$$\underline{v}_1 G = \underline{c}_1$$

$$\underline{c}_1 = 11100$$

$$G = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

$$\underline{v}_1 = 01$$

Matlab

$$v_1' = gflineq(G', c_1', 2)$$

$$\underline{v}_1 = 01$$

$$S^{-1} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$$

$$\underline{v} = \underline{v}_1 S^{-1} = 11$$

McEliece cryptosystem

The McEliece cryptosystem is a public key algorithm based on error correcting codes.

The security of the algorithm is based on the complexity of decoding a generic linear code.

Currently (December 2022), the McEliece scheme is one of the four schemes admitted at round 4 of NIST (National Institute of Standards and Technology) Post-Quantum Cryptography Standardization competition.

Summary

Given a linear block code $C(n, k)$ able to correct t errors per block, the user U_1 public key is (G_1, t_1), where G_1 is an obfuscated generator matrix $G_1 = SGP$ (S a non-singular matrix, G a generator matrix of C and P a permutation matrix).

If the user U_2 wants to sends a message to U_1, it represents the message by a vector $\underline{v}$ of k-bits, encodes it by G_1 to obtain $\underline{c} = \underline{v}G_1$ and adds t_1 random errors to obtain the transmitted vector $\underline{y}$.

The user U_1 decodes $\underline{y}$ by applying a low-complexity decoding algorithm designed for C and recover the message vector $\underline{v}$.

Thanks to obfuscation, the true code is unknown to the attacker. The decoding of a generic linear code is known to be NP-HARD then it becomes unfeasible if k and n are large.

Binary Alphabet

We represent the binary alphabet by the symbol $F = \{0, 1\}$ (or GF(2)).
The binary sum has addition table:

$+$	0	1
0	0	1
1	1	0

The binary multiplication sum has multiplication table

$\cdot$	0	1
0	0	0
1	0	1

With these two operations, $(F, +, \cdot)$ is a field.

Vector of bits

We represent the set of all possible k-bit vectors as

$$F^k = \{\underline{v} = (v_1, \cdots, v_i, \cdots, v_k) \quad v_i \in F = \{0, 1\}\}$$

(ex: $F^2 = \{(00), (01), (10), (11)\}$)
Given two binary vectors we define their sum as the bit-by-bit sum:

$$\underline{v} = (v_1, \cdots, v_i, \cdots, v_k) \in F^k \quad \underline{w} = (w_1, \cdots, w_i, \cdots, w_k) \in F^k$$

$$\underline{a} = \underline{v} + \underline{w} = (v_1 + w_1, \cdots, v_i + w_i, \cdots, v_k + w_k) \in F^k$$

(ex $11 + 10 = 01$)
and the multiplication of a vector by a bit as:

$$b \in F = \{0, 1\} \quad \underline{v} = (v_1, \cdots, v_i, \cdots, v_k) \in F^k$$

$$\underline{c} = b \cdot \underline{v} = (b \cdot v_1, \cdots, b \cdot v_i, \cdots, b \cdot v_k) \in F^k$$

(ex $1 \cdot 11 = 11$ and $0 \cdot 11 = 00$)

Vector space and group properties

It is easy to show that F^k is a vector space.
In particular, $(F^k, +)$ is a commutative group:
Closure:

$$\forall \underline{v}, \underline{w} \in F^k \quad \underline{v} + \underline{w} \in F^k$$

Sum identity:

$$\underline{0} = (0, \cdots, 0, \cdots, 0) \in F^k \quad \forall \underline{v} \in F^k \quad \underline{v} + \underline{0} = \underline{v}$$

Additive inverse:

$$\forall \underline{v} \in F^k \quad \underline{v} + \underline{v} = \underline{0}$$

Associative property

$$\forall \underline{v}_1, \underline{v}_2, \underline{v}_3 \in F^k \quad \underline{v}_1 + (\underline{v}_2 + \underline{v}_3) = (\underline{v}_1 + \underline{v}_2) + \underline{v}_3$$

Commutative property

$$\forall \underline{v}_1, \underline{v}_2 \in F^k \quad \underline{v} + \underline{w} = \underline{w} + \underline{v}$$

Code $C(n, k)$

A code $C(n, k)$ is characterized by two parameters

- k = information vector length
- n = codeword length

The encoder maps any k-bit information vector $\underline{v} \in F^k$ into an n-bit codeword $\underline{c} \in F^n$:

$$\begin{array}{rccc} e: & F^k & \longrightarrow & F^n \\ & \underline{v} & \longrightarrow & \underline{c} \end{array}$$

The map is linear and one-to-one, then it can be represented by a $k \times n$ generator matrix G, with rank k:

$$\underline{c} = \underline{v}G$$

Example

$C(n = 7, k = 4)$

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

$$\underline{v} = (1000)$$

$$\underline{c} = \underline{v}G = (1000111)$$

Codebook

Given a code $C(n,k)$ the codebook is the set of all possible n-bit codewords $\underline{c} \in F^n$ obtained by encoding all the possible 2^k information vectors $\underline{v} \in F^k$. Since the matrix G has full rank, all the codewords are distint and the codebook cardinality is 2^k, too.

$$C = \left\{ \underline{c} = \underline{v}G \ \ \forall \underline{v} \in F^k \right\}$$

$$|C| = 2^k$$

Example

In the previous example $C(n = 7, k = 4)$

$$
\begin{array}{lcl}
0000 & \longrightarrow & 0000000 \\
0001 & \longrightarrow & 0001000 \\
0010 & \longrightarrow & 0010000 \\
0011 & \longrightarrow & 0011000 \\
0100 & \longrightarrow & 0100000 \\
0101 & \longrightarrow & 0101000 \\
0110 & \longrightarrow & 0110000 \\
1000 & \longrightarrow & 1000000 \\
1001 & \longrightarrow & 1001000 \\
1010 & \longrightarrow & 1010000 \\
1011 & \longrightarrow & 1011000 \\
1100 & \longrightarrow & 1100000 \\
1101 & \longrightarrow & 1101000 \\
1110 & \longrightarrow & 1110000 \\
1111 & \longrightarrow & 1111000
\end{array}
$$

Hamming weight and Hamming distance

The Hamming weight w_H of a binary vector of F^n is the number of bits equal to one. The Hamming distance d_H between two binary vectors of F^n is the number of bits where they are different. It is easy to show that

$$d_H\left(\underline{c_1}, \underline{c_2}\right) = w_H\left(\underline{c_1} + \underline{c_2}\right)$$

ex $w_H(101) = 3$
ex $d_H(101, 110) = w_H(011) = 2$

Minimum distance of a code

The minimum distance of a code is the minimum Hamming distance between different codewords

$$d_{\min} = \min_{\left[\forall \underline{c}_1, \underline{c}_2 \in C \;\; \underline{c}_1 \neq \underline{c}_2\right]} d_H\left(\underline{c}_1, \underline{c}_2\right)$$

It is easy to show that it can be computed as teh minimum Hamming weight of a non-zero codeword

$$d_{\min} = \min_{\left[\forall \underline{c} \in C \;\; \underline{c} \neq \underline{0}\right]} w_H\left(\underline{c}\right)$$

Error vector

Suppose to transmit a codeword $\underline{c} \in C$ and to receive a vector $\underline{y}$ where, for some reason (noise, intentionally added errors, ...) some of the bits are received wrong, i.e., inverted.
We can represent the wrong bits by an error vector $\underline{e} = (e_1, \cdots, e_i, \cdots e_n) \in F^n$ where $e_i = 0$ if $y_i = c_i$ and $e_i = 1$ if $y_i \neq c_i$.
Example: $\underline{c} = (1000), \underline{y} = (0101), \underline{e} = (1101)$.

Minimum distance decoding rule

Given $\underline{y}$ we can choose the codeword that minimizes the error probability by the minimum distance decoding rule:

$$\hat{\underline{c}} = arg \min_{\underline{c} \in C} d_H \left(\underline{y}, \underline{c} \right)$$

Unfortunately, the complexity of this decoding rule is proportional to the code cardinality 2^k and becomes rapidly unfeasible for non-trivial k.

Error correction capability

The minimum distance of the code determines the number of errors that a code is certainly able to correct by a minimum distance rule:

$$t = \left\lfloor \frac{d_{\text{min}} - 1}{2} \right\rfloor$$

Parity check matrix

Given a code $C(n,k)$ we define $r = n - k$.
Given an n-bit vector $\underline{y}$ we want to establish a test to determine if it belongs to the code or not. Let us introduce an $n \times r$ binary matrix H and consider this linear function:

$$\begin{array}{rccc} f: & F^n & \longrightarrow & F^r \\ & \underline{y} & \longrightarrow & \underline{s} = \underline{y}H \end{array}$$

We call the r-bit vector $\underline{s} \in F^r$ the syndrome of $\underline{y}$.
We want to build a matrix H such that $\underline{s} = \underline{0}$ if and only if $\underline{y} \in C$.
It is easy to show that this matrix must have these two properties:

- rank(H)=r
- $GH = 0$

Syndrome decoding

The syndrome decoding is a simplified decoding algorithm that we can apply when we know the parity check matrix H of the code.
First we note that, given $\underline{y} = \underline{c} + \underline{e} \Longrightarrow \underline{s} = \underline{y}H = \underline{e}H$, i.e., the syndrome only depends on the error vector.
Now, suppose we want to correct all the error vectors up to weight t. For each of them we precompute the syndrome and we store it into a LookUpTable (LUT) mapping each syndrome into the corresponding error vector.
At the received side, given $\underline{y}$ we compute the syndrome $\underline{s} = \underline{y}H$.
We check if $\underline{s}$ is listed in the LUT: in this case we get the corresponding error vector $\underline{e}$ and we build the received vector as

$$\hat{\underline{c}} = \underline{y} + \underline{e}$$

McEliece cryptosystem

The user U_1 selects a $k \times n$ generator matrix G.
It obfuscates it by computing

$$G_1 = SGP$$

where

- S is a non-singular (full rank) $k \times k$ binary matrix
- P is an $n \times n$ permutation matrix

Note that if k and n are big enough, it is computationally hard to identify the true generator matrix G.
The public key is (G_1, t_1) where $t_1 \leq t$ is the number of errors that U_1 wants to correct.

Encryption

The user U_2 partition its message into k-bit vectors $\underline{v}$, then for each vector $\underline{v}$:

- Computes $\underline{c} = \underline{v}G_1$
- Generates a random error vector $\underline{e}$ with t_1 errors: $w_H(\underline{e}) = t_1$.
- Computes $\underline{y} = \underline{c} + \underline{e}$: this is the ciphertext sent to U_1.

McEliece cryptosystem

The user U_1 receives $\underline{y}$ then

- Generates $\underline{y}_1 = \underline{y}P^{-1}$.
- Decodes $\underline{y}_1$ to obtain $\underline{c}_1$
- From $\underline{c}_1 = G\underline{v}_1$ recovers $\underline{v}_1$
- Generates $\hat{\underline{v}} = \underline{v}_1 S^{-1}$.

Finally we have $\hat{\underline{v}} = \underline{v}$ and the original message is retrieved.

Exercise

You will receive by email:

- the generator matrix G
- the parity check matrix H
- the non-singular matrix S
- the permutation matrix P
- the number of errors $t_1 = 1$
- a sequence of ciphertexts corresponding to a message

$k = 4 \qquad n = 7$

```
% matrix G
G = [1,0,1,0,1,0,1;1,1,0,0,0,1,1;0,0,1,0,0,1,1;0,0,1,1,1,0,0];
% matrix H
H = [0,1,0;1,1,0;1,0,0;0,1,1;1,1,1;1,0,1;0,0,1];
% matrix S
S = [1,1,1,0;0,0,0,1;0,1,0,0;0,1,1,1];
% matrix P
P = [0,0,1,0,0,0,0;0,0,0,1,0,0,0;0,1,0,0,0,0,0;0,0,0,0,0,1,0;0,0,0,0,1,0,0;0,0,0,0,0,0,1;1,0,0,0,0,0,0];
% ciphertext
cipher =
[0,1,0,1,1,1,0,1,0,0,1,0,0,1,0,1,0,1,0,0,1,1,0,0,1,0,0,1,1,1,0,0,0,1,0,0,0,1,0,0,0,1,0,0,1,1,0,0,1,0,0,1,0,0,0,1,0,1,1,1,1,0,1,1,
1,1,1,0,0,1,1,1,0,0,1,0,1,1,0,1,1,1,0,1,];
```

Exercise

- Write a program to implement the deciphering of the McEliece scheme to recover the transmitted message.

In the report, write:

- All the received inputs (string, G, H, ...)
- All the matrices you computed to solve the problem (S^{-1}, P^{-1})
- The LUT
- The name of the city

Exercise [redacted]: ASCII

The message is the name of a city. Each capital letter is encoded into an 8-bit vector by using ASCII Windows-1252 encoding according to this table:

letter	ASCII code	binary	letter	ASCII code	binary
A	65	01000001	N	78	01001110
B	66	01000010	O	79	01001111
C	67	01000011	P	80	01010000
D	68	01000100	Q	81	01010001
E	69	01000101	R	82	01010010
F	70	01000110	S	83	01010011
G	71	01000111	T	84	01010100
H	72	01001000	U	85	01010101
I	73	01001001	V	86	01010110
J	74	01001010	W	87	01010111
K	75	01001011	X	88	01011000
L	76	01001100	Y	89	01011001
M	77	01001101	Z	90	01011010

Exercise [redacted]: Suggestion

We strongly suggest to implement the entire McEliece scheme (encryption + decryption) and test the toy example presented on the handwritten notes.

Exercise [redacted]: LUT

Since (in this simplified example) we have $t_1 = 1$ you must consider $n + 1$ error vectors $\underline{e}$:

- the all-zero error vector
- the n error vectors of weight 1

For each error vector compute the corresponding syndrome $\underline{s} = \underline{e}H$ and store all the syndromes/error vectors pairs.

Exercise : McEliece decryption

- Divide your string into n-bit vectors $\underline{y}$
- Compute $\underline{y}_1 = \underline{y}P^{-1}$
- Decode $\underline{y}_1$ to obtain $\underline{v}_1$ by applying syndrome decoding

Exercise [redacted]: Syndrome decoding

- Compute the syndrome $\underline{s} = \underline{y}_1 H$
- Obtain the corresponding error vector $\underline{e}$ from the LUT
- Obtain the received codeword $\underline{c}_1 = \underline{y}_1 + \underline{e}$
- Obtain the corresponding information vector $\underline{v}_1$.

Exercise [redacted]: Final result

- Obtain $\underline{\hat{v}} = \underline{v}_1 S^{-1}$
- Collect together all information vectors $\underline{\hat{v}}$
- Apply the ASCII decoding to obtain the name of the city

```
lett=binaryVectorToDecimal(flip(vett));
L(i)=char(lett);
```

Exercise [redacted]: Question

3. [redacted] Which values of k and n are considered for practical applications of the McEliece scheme? Add the references where you found the results.

Exercise 4.4

Important: this is an alternative to Exercise 4.3. You cannot solve both of them.

Exercise 4.4: Signal/TLS

1. (pt. 5) The Signal Protocol (WhatsApp encryption): what is it, how the keys are generated and exchanged, which algorithms are used for encryption (no detailed description of the algorithm, but write on which technique they are based on (e.g., prime factorization or discrete log or elliptic curves or ...), which modes are used, how group chats are managed, role of symmetric and asymmetric encryption.
2. (pt.5) The TLS Protocol 1.3 (https encryption): what is it, how the keys are generated and exchanged, which algorithms are used for encryption (no detailed description of the algorithm, but write on which technique they are based on (e.g., prime factorization or discrete log or elliptic curves or ...), which modes are used, role of symmetric and asymmetric encryption..

Important:

- Max 5 pages for each question.
- Properly cite all the used references