Code-based Cryptography

Irene Márquez Corbella
Universidad de La Laguna
Secure CAT - Kick off Meeting

Prerrequisites on error correcting codes

A **linear code** is a vector subspace $\mathcal{C} \subseteq \mathbb{F}_q^n$ where:

- $n(\mathcal{C}) = n$ is its **length**,
- $k(\mathcal{C}) = k$ is its **dimension** as $\mathbb{F}_q$-vector space.
- $d(\mathcal{C}) = d$ is its **minimum Hamming distance**.

Prerrequisites on error correcting codes

A **linear code** is a vector subspace $\mathcal{C} \subseteq \mathbb{F}_q^n$ where:

- $n(\mathcal{C}) = n$ is its **length**,
- $k(\mathcal{C}) = k$ is its **dimension** as $\mathbb{F}_q$-vector space.
- $d(\mathcal{C}) = d$ is its **minimum Hamming distance**.

The **Hamming distance** on $\mathbb{F}_q^n$ is defined by:

$$d_H(\mathbf{x}, \mathbf{y}) = |\, \{i \in \{1, \ldots, n\} \mid x_i \neq y_i\} \,|$$

Prerrequisites on error correcting codes

A **linear code** is a vector subspace $\mathcal{C} \subseteq \mathbb{F}_q^n$ where:

- $n(\mathcal{C}) = n$ is its **length**,
- $k(\mathcal{C}) = k$ is its **dimension** as $\mathbb{F}_q$-vector space.
- $d(\mathcal{C}) = d$ is its **minimum Hamming distance**.

The **Hamming distance** on $\mathbb{F}_q^n$ is defined by:

$$d_H(\mathbf{x}, \mathbf{y}) = |\{i \in \{1, \ldots, n\} \mid x_i \neq y_i\}|$$

A **t-decoder** for $\mathcal{C}$ is an algorithm $\mathcal{D}_{\mathcal{C}}$ taking as input $\mathbf{x} \in \mathbb{F}_q^n$ and returning:

- $\mathbf{c} \in \mathcal{C}$ such that $d_H(\mathbf{x}, \mathbf{c}) \leq t$ it exists.
- ? or *FAILURE* else.

Prerrequisites on error correcting codes

A **linear code** is a vector subspace $\mathcal{C} \subseteq \mathbb{F}_q^n$ where:

- $n(\mathcal{C}) = n$ is its **length**,
- $k(\mathcal{C}) = k$ is its **dimension** as $\mathbb{F}_q$-vector space.
- $d(\mathcal{C}) = d$ is its **minimum Hamming distance**.

Let $\mathcal{C} \subseteq \mathbb{F}_{q^m}^n$ be a code. Its **subfield subcode** is defined by

$$\mathcal{C} \cap \mathbb{F}_q^n$$

Many algebraic codes derive from GRS codes using this operation:
Goppa Codes, BCH codes, Srivastava codes, etc

Outline

1. History of code-based cryptography
2. Algebraic cryptanalysis in code-based cryptography
3. How to design secure schemes with codes?

Public Key Cryptography

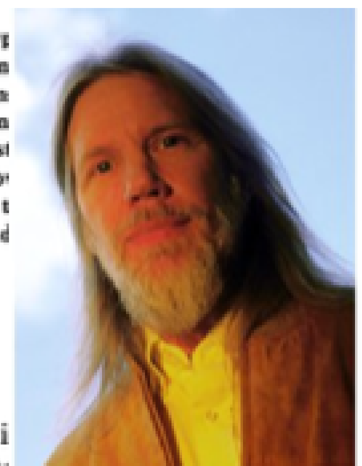
644 IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. IT-22, NO. 6, NOVEMBER 1976

New Directions in Cryptography

Invited Paper

WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEMBER, IEEE

Abstract—Two kinds of contemporary developments in cryp tography are examined. Widening applications of teleprocessin have given rise to a need for new types of cryptographic system which minimize the need for secure key distribution channels an supply the equivalent of a written signature. This paper suggest ways to solve these currently open problems. It also discusses ho the theories of communication and computation are beginning t provide the tools to solve cryptographic problems of long stand ing.



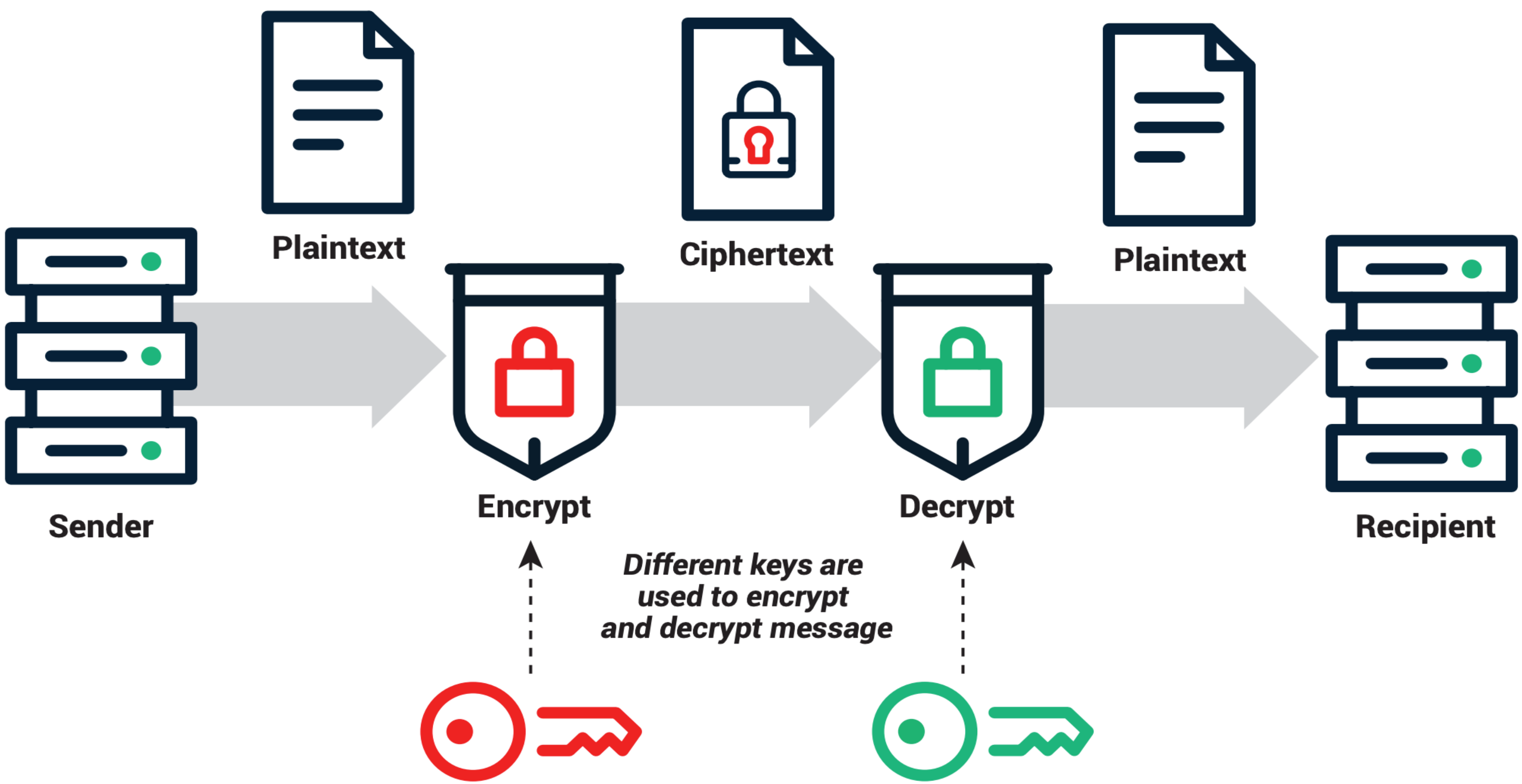
I. Introduction

WE STAND TODAY on the brink of a revolution i cryptography. The development of cheap digit hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals. In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels and supply the equivalent of a written signature. At the same time, theoretical developments in information theory

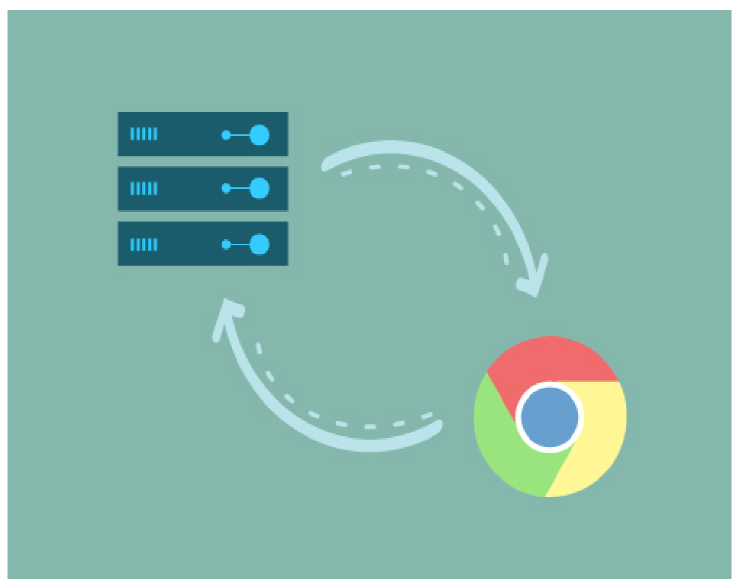
communications to large teleprocessing networks.

Section III proposes two approaches to transmitting keying information over public (i.e., insecure) channels without compromising the security of the system. In a *public key cryptosystem* enciphering and deciphering are governed by distinct keys, E and D, such that computing D from E is computationally infeasible (e.g., requiring 10^{100} instructions). The enciphering key E can thus be publicly disclosed without compromising the deciphering

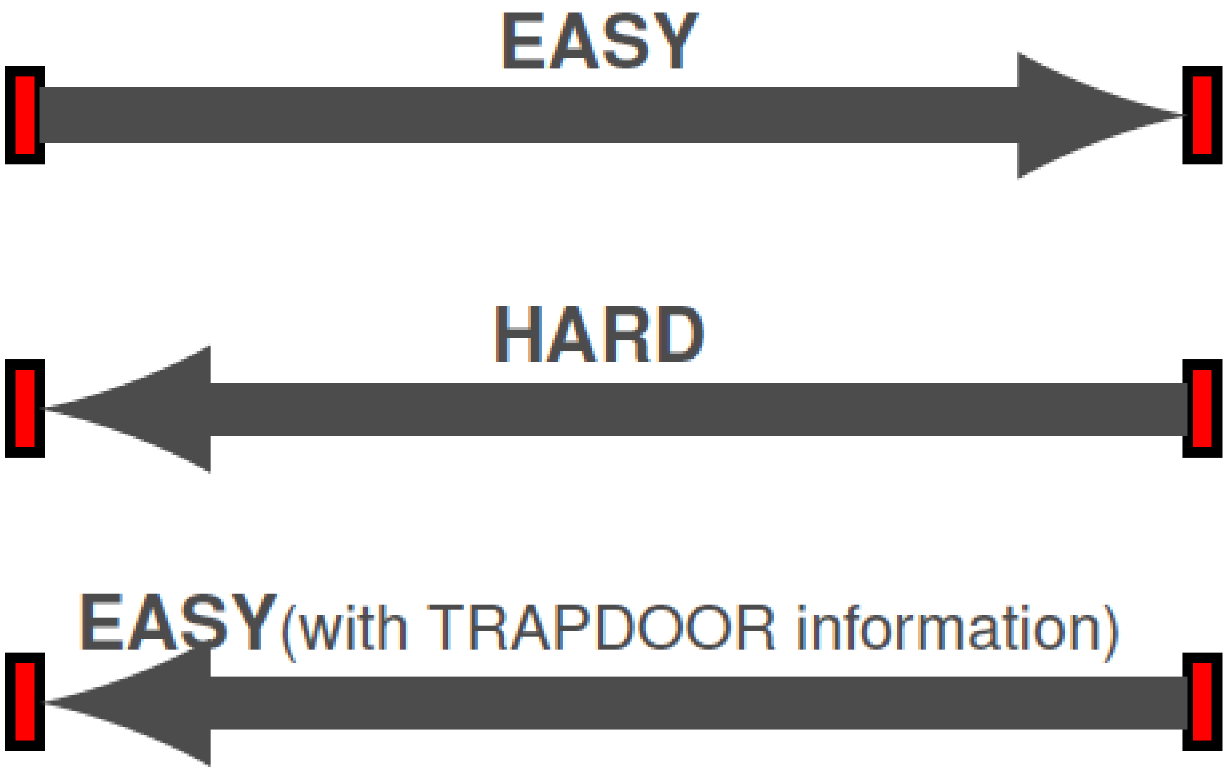
Public Key Cryptography vs Secret Key Cryptography



Do we need PKC?

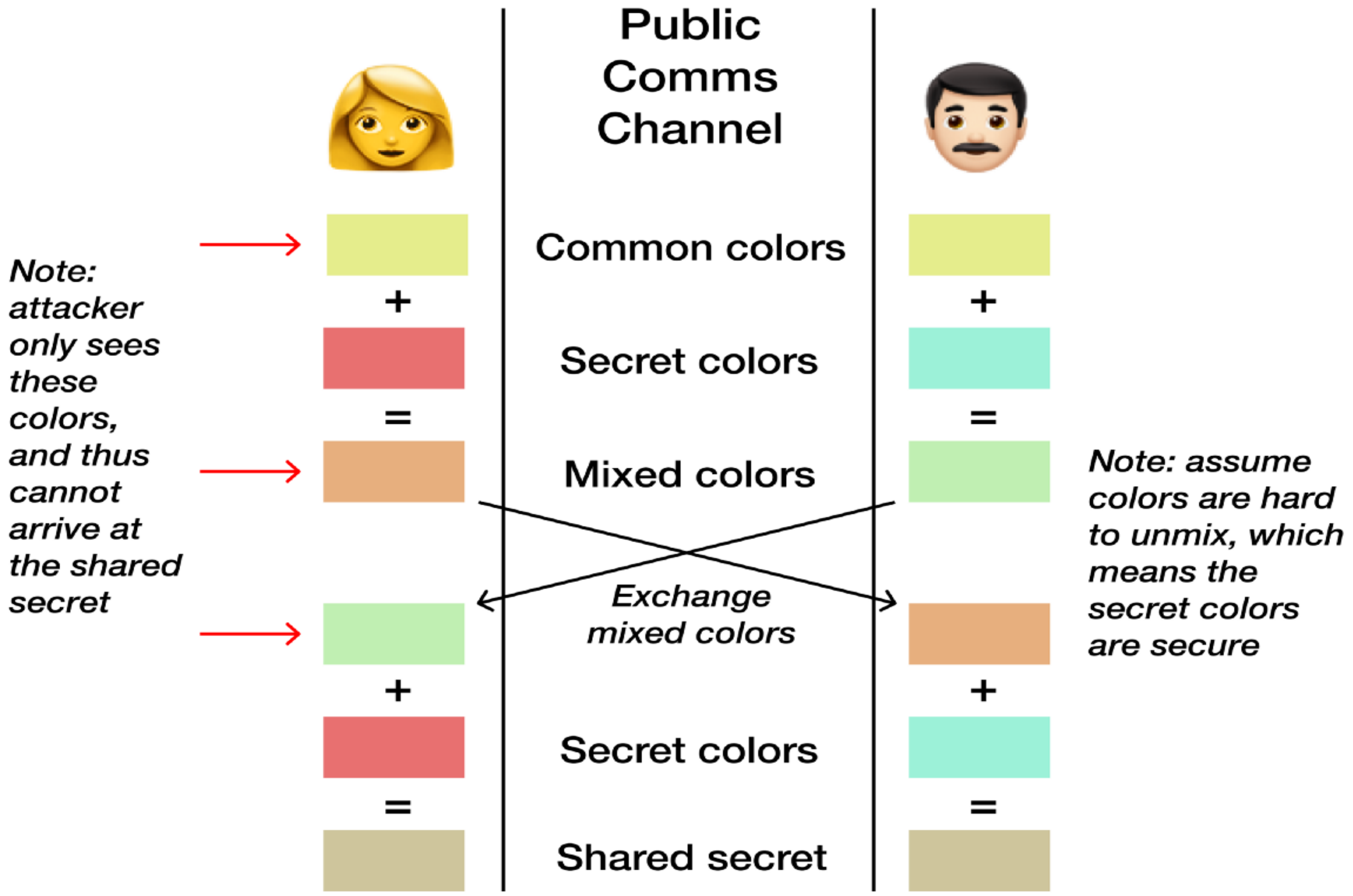


Public Key Cryptography (PKC)



PKC - Easy example

PKC - Easy example



PKC - Famous example

Trapdoor one-way functions - Integer Factorization

First Challenge: Find prime factors of 4757.

PKC - Famous example

Trapdoor one-way functions - Integer Factorization

First Challenge: Find prime factors of 4757.

Seconde Challenge: Multiply 67 and 71.

PKC - Famous example

Trapdoor one-way functions - Integer Factorization

```
4673331833592310999883355855611155212513211028177144957985823385935679234805211772074843110997402088
4962136809003804931724836744251351914436524922028678749922492363963303861930595117077052285035601177
9638644050954128274109548519743273551014325753249976993808191641040774990607027085131780854431482719
2879270515747600591825011224264939011775241470201122113881802463571203852569710311808614896188925840
6775097681495456790744215925392808604345151310705231857280062253517330504393154504927694689628526886
9674944342112985792233732337801754241421827174125670264416644353313890442672256181107628062641550510
9923842039912255378570492258674504781998501869851883957199630080387179659069436984462272457690484426
2407704045651692639000865172646299059376059542948679165463356213921674455767274649788443435352045655
6797052450980481438931349795938877105350614496693489409255155953306872814733490045565082856578190868
9333271410463787949726552668938875959796413163310288065921775297698341521241159133233652681667066444
7314331097452082351397488563625371930195040660572209571807917346778421232394122570849227616267884850
4240175619575042958633387070067162448853074807881260125089824549619209919961134580250514604063519606
6349781811986135030515811634635556335663198966158276043886903297960119840962705373835796308746810568
4364981806767810342409685957459438712247657182807255734439478038002420178354866749517974669619084362
2986643959450031252516686566594252074345358101581042723707992427571828820948849065861590065859074391
3778281730346837019251147896187873091018870042890461298730287464163869574436440285888093124299088608
8429761386038681691058654221673900140273498297190727298748366210723682751247273840980957893067062615
3249685719278922751692157130896802807496462527584643633045876649970923366331569812027362273631245871
5213560116148604399880851787687637578607326255851154570920878480432582578762864987064580881350389488
2224735180713088840316464579444631924371791325993347700122099445881579566373010228501014181466326553
7150923894600650386095599714691658518044760943228485293103850039495787904594766307709643249139518714
4359123613861061413590157894888931401420525131224866516461147016667016761431407500872246038892346552
0552808611097373063518704213113039301625336279405825183612805551854967893065836645527291551181195205
8888392595313861166137697724678782720586680474567342842176857469092018559835324800004598444782456084
4859045762973388136611991702058586520990334357374055942865874795790720345913488804917784805628948577
8804617732179296825817159750372007979156699208305582486657901255718072275107846292479424843965207746
7237403658550061799279956704411031254567465451105499362798947781210188466981867175415780089815289984
9247926557842034715220235736186498477432122404953393979535957780605535102962617356735540344891086853
8926634460813390353581444858227418449682398889148543390840713875511844096578060875650223903048389134
9117815030583034147983474364473410786162232983781676844315610390028354503758000566280031377558992067
1170248809970337175034006733019227380745118643000374192911602711339184034355819927937019521413721001
3559545759948525467412161869068267441364577423716904925542472866535799610399706977624396801087677647
5830443576635739972027953438424843103674054245431824410173440025375378765370235220916664367509996157
3987156731808048535495650986676078713033080449444052838485327694559548811642663228650685618461821860
7927872621210107894980393234159043797743621683940604454280871426803006376885767541206049493503285861
4372477002463478278523399030686898768798513296184049579718910789231320899515797161738788846785311885
1938867611102089288594969572320827894145129733289906382456243680285209396595612312525464356962608456
2680638161004265280804116438967394364534002550412238516626332322616822282772995562836029765425657159
4692470308676425544869996972655568942309011083411493144986858214357859775618228826805464134953908832
4070789327147979348300969348265766851051446870625635324729750318877018962969848005872490328704865371
4711014261765297188923940302103960017543971660344237708051436459112134068551823257845881972600579821
4432173230826258923166734660042915970100922079805049368311210507812753027221812929184183072115802000
```

PKC - Famous example

Trapdoor one-way functions - Integer Factorization

The largest known prime number (in May 2019) is

$$2^{82.589.933} - 1$$

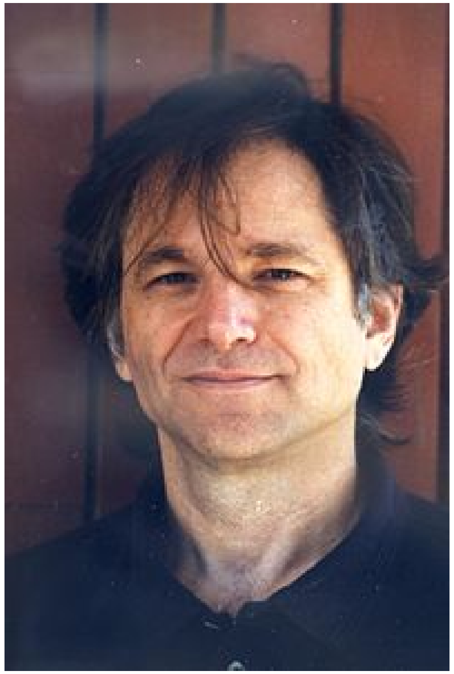
this number has 24.862.048 digits and was found by Patrick Laroche in 2018 as part of the proyect Great Internet Mersenne Prime Search (GIMPS)

3000\$ rewards

RSA

Ronald Rivest (1947)

Adi Shamir (1952)



Leonard Adleman (1945)

RSA is a public key cryptographic system developed in 1979 at the Massachusetts Institute of Technology (MIT). It is the algorithm of PKC most widely used today.
The security of the **RSA** relies on the **Integer Factorization**

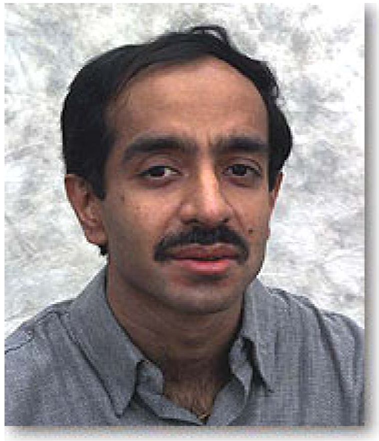
Post-quantum Cryptography

There are **2** quantum algorithms that affect cryptography:

Post-quantum Cryptography

There are **2** quantum algorithms that affect cryptography:

- Grover's Algorithm: Finds b-bit preimages in $2^{\frac{b}{2}}$ quantum operations. It requieres:
  - $2\times$ key size in Symmetric ciphers
  - Longer output sizes in Hash Functions

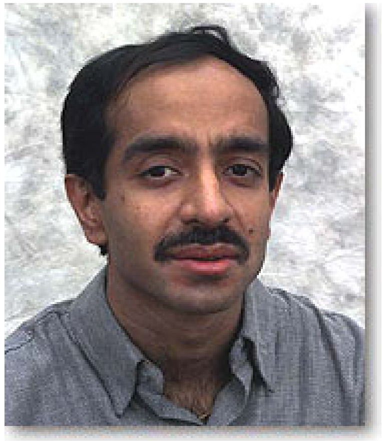


Lov Grover

Post-quantum Cryptography

There are **2** quantum algorithms that affect cryptography:

- GROVER'S ALGORITHM: Finds b-bit preimages in $2^{\frac{b}{2}}$ quantum operations. It requieres:
  - $2\times$ key size in Symmetric ciphers
  - Longer output sizes in Hash Functions
- SHOR'S ALGORITHM: Has dramatic effects on PKC, it breaks:
  - RSA cryptosystem.
  - Cryptosystems based on Discret log in finite fields and elliptic curves.



Lov Grover

Peter Shor

Preparing for the Cryptopocalypse

IBM Q system One - 2019

Preparing for the Cryptoapocalypse

Crypto-apocalypse

Preparing for the Cryptoapocalypse

Crypto-apocalypse

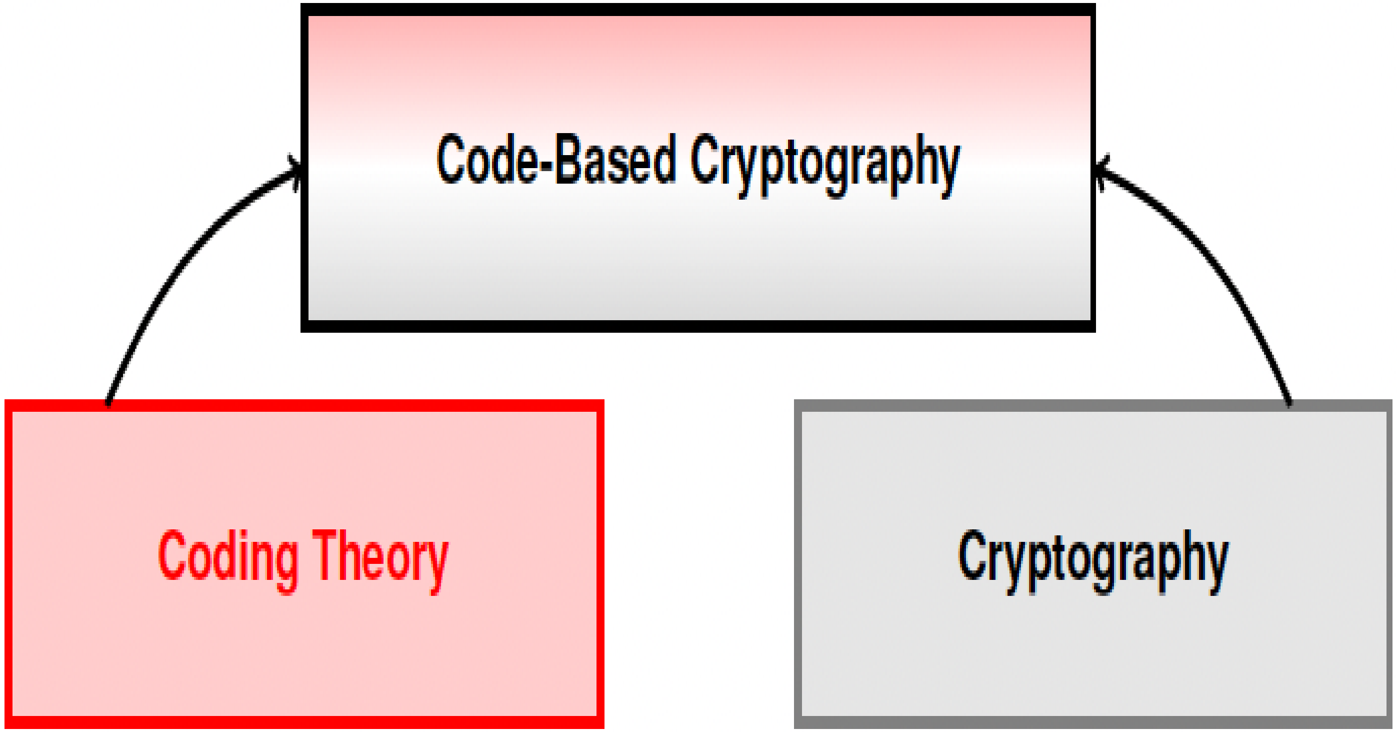
NIST (National Institute of Standards and Technology)

starts a proyect entitled:

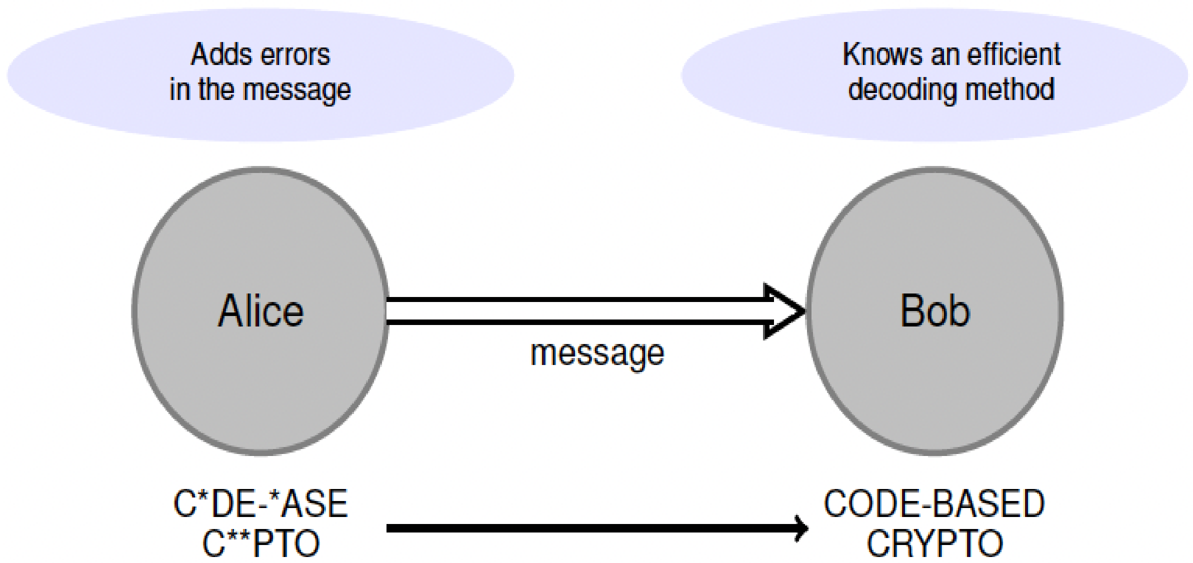
Post-Quantum Cryptography standardization.

- It start in November 2017: 69 proposals where sent.
- July 2022: 4 submissions were announced

Coding Theory vs. Cryptography



How to use Coding Theory in Cryptography?



It starts with two articles

[1] E.R. Berlekamp, R.J. McEliece and H.C.A. Van Tilborg. On the inherent intractability of certain coding problems. IEEE Trans. Inform. Theory 24(2), 1978.

[2] R.J. McEliece. A public key cryptosystem based on algebraic coding theory. DSN Progress Report 44; 1978.

It starts with two articles

[1] E.R. Berlekamp, R.J. McEliece and H.C.A. Van Tilborg. On the inherent intractability of certain coding problems. IEEE Trans. Inform. Theory 24(2), 1978.

[2] R.J. McEliece. A public key cryptosystem based on algebraic coding theory. DSN Progress Report 44; 1978.

In the article **[1]**
The proof that the following problem is NP-complete:
Bounded decoding problem. Given $\mathcal{C} \subseteq \mathbb{F}_q^n$, $\mathbf{y} \in \mathbb{F}_q^n$ and $t \geq 0$.
Does there exists $\mathbf{c} \in \mathcal{C}$ such that:

$$d_H(\mathbf{c}, \mathbf{y}) \leq t?$$

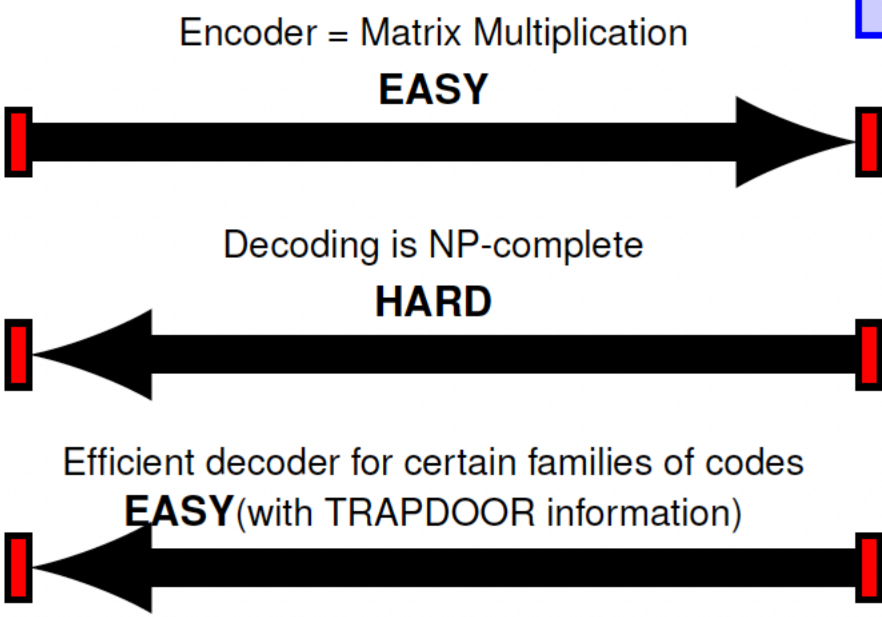
It starts with two articles

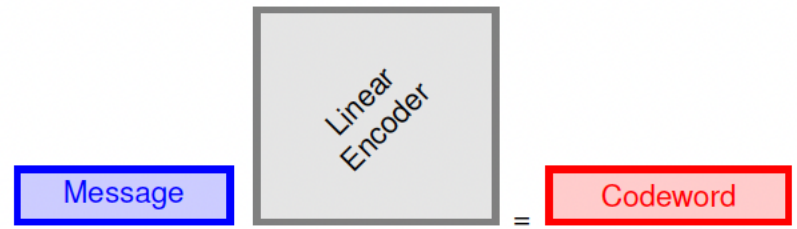
[1] E.R. Berlekamp, R.J. McEliece and H.C.A. Van Tilborg. On the inherent intractability of certain coding problems. IEEE Trans. Inform. Theory 24(2), 1978.

[2] R.J. McEliece. A public key cryptosystem based on algebraic coding theory. DSN Progress Report 44; 1978.

In article **[2]** McEliece proposes a new PKC encryption scheme.

Trapdoor One-Way Functions - McEliece





E. R. Berlekamp, R. J. McEliece and H. C. A. van Tilborg.
On the Inherent Intractability of Certain Coding Problems.
IEEE Trans. Inf. Theory. Vol. 24, pp. 384-386, 1978.

A. Barg.
Complexity Issues in Coding Theory.
Chapter 7, in Handbock of Coding Theory, 1998.

McEliece presented in the literature

Secret Key:

- G a $k \times n$ generator matrix of a code $\mathcal{C}$.
- S a $k \times k$ non-singular matrix.
- P a $n \times n$ permutation matrix.

Public Key: $(G' = SGP, t)$

Encryption: $\mathrm{Enc}(\mathbf{m}) = \mathbf{m}G' + \mathbf{e}$ with $\mathbf{e} \in \mathbb{F}_q^n$ uniformly random of weight t.

Decryption:

1. Right multiply by P^{-1}: $(\mathbf{m}G' + \mathbf{e}) \times P^{-1} = \mathbf{m}SG + \mathbf{e}P^{-1}$
2. Decode to get $\mathbf{m}S$
3. Right multiply by S^{-1} to get $\mathbf{m}$

This is what McEliece said:

A Public-Key Cryptosystem Based On Algebraic Coding Theory

R. J. McEliece
Communications Systems Research Section

Using the fact that a fast decoding algorithm exists for a general Goppa code, while no such exists for a general linear code, we construct a public-key cryptosystem which appears quite secure while at the same time allowing extremely rapid data rates. This kind of cryptosystem is ideal for use in multi-user communication networks, such as those envisioned by NASA for the distribution of space-acquired data

I. Introduction

Recently, Diffie and Hellman (Ref. 3) introduced the notion of a *public-key cryptosystem* in which communication security is achieved without the need of periodic distribution of a secret key to the sender and receiver. This property makes

Corresponding to each irreducible polynomial of degree t over $GF(2^m)$, there exists a binary irreducible Goppa code of length $n = 2^m$, dimension $k \geqslant n - tm$, capable of correcting any pattern of t or fewer errors. Moreover, there exists a fast algorithm for decoding these codes. [Algorithm due to

This is what McEliece said:

A Public-Key Cryptosystem Based On Algebraic Coding Theory

II. Description of the System

We base our system on the existence of *Goppa codes*. For the full theory of such codes the reader is referred to (Ref. 5, Chapter 8), but here we summarize the needed facts.

the code, which could be in canonical, for example row-reduced echelon, form.

Having generated G, the system designer now "scrambles" G by selecting a random dense $k \times k$ nonsingular matrix S, and a random $n \times n$ permutation matrix P. He then computes

---

$G' = SGP$, which generates a linear code with the same rate and minimum distance as the code generated by G. We call G' the public generator matrix, since it will be made known to the outside world.

and an astronomical number of choices for S and P. The dimension of the code will be about $k = 1024 \text{-} 50 \cdot 10 = 524$. Hence, a brute-force approach to decoding based on comparing $\mathbf{x}$ to each codeword has a work factor of about $2^{524} = 10^{158}$; and a brute-force approach based on coset leaders has a

We could present it differently:

A. Couvreur's IDEA

- $\mathcal{F}$ denotes a family of codes of length n and dimension k.
- $\mathcal{S}$ denotes a set of "*secrets*" with a surjective map $\mathcal{C}: \mathcal{S} \longrightarrow \mathcal{F}$ sending a secret $s \in \mathcal{S}$ into a code $\mathcal{C}(s)$.
- To any $s \in \mathcal{S}$ is associated a decoding algorithm $\mathcal{D}(s)$ for $\mathcal{C}(s)$ correcting up to t errors.

We could present it differently:
A. Couvreur's IDEA

- $\mathcal{F}$ denotes a family of codes of length n and dimension k.
- $\mathcal{S}$ denotes a set of *"secrets"* with a surjective map $\mathcal{C} : \mathcal{S} \longrightarrow \mathcal{F}$ sending a secret $s \in \mathcal{S}$ into a code $\mathcal{C}(s)$.
- To any $s \in \mathcal{S}$ is associated a decoding algorithm $\mathcal{D}(s)$ for $\mathcal{C}(s)$ correcting up to t errors.

Secret Key: $s \in \mathcal{S}$

Public Key: (G, t) where G is a $k \times n$ generator matrix for $\mathcal{C}(s)$.

Encryption: $\mathrm{Enc}(\mathbf{m}) = \mathbf{m}\mathrm{G} + \mathbf{e}$ where $\mathbf{e} \in \mathbb{F}_q^n$ is a uniformly random vector of weight t.

Decryption: Apply $\mathcal{D}(s)$ to $\mathbf{m}G + \mathbf{e}$ to recover $\mathbf{m}$.

Example - GRS codes

- Let $n,\ k$ be positive integers with $1 \leq k \leq n \leq q$.
- $\mathbf{a} = (a_1, \ldots, a_n) \in \mathbb{F}_q^n$ with $a_i \neq a_j$ for all $i \neq j$.
- $\mathbf{b} = (b_1, \ldots, b_n) \in \mathbb{F}_q^n$ with $b_i \neq 0$ for all i.
- **Polynomial space:**

$$L_k = \mathbb{F}_q[X]_{<k} = \{f(X) \in \mathbb{F}_q[X] \mid \deg(f) < k\}$$

  L_k is a vector space of dimension k, with canonical basis: $\mathcal{B} = \{1, \ldots, x^{k-1}\}$

- **Evaluation map:**

$$\begin{array}{rccc} \mathrm{ev}_{\mathbf{a},\mathbf{b}}: & L_k & \longrightarrow & \mathbb{F}_q^n \\ & f(X) & \longmapsto & \mathbf{b} * f(\mathbf{a}) = (b_1 f(a_1), \ldots, b_n f(a_n)) \end{array}$$

Definition: GRS codes

$$\mathrm{GRS}_k(\mathbf{a}, \mathbf{b}) = \{\mathrm{ev}_{\mathbf{a},\mathbf{b}}(f) \mid f \in L_k\}$$

Example - GRS codes

Definition: GRS codes

$$\mathrm{GRS}_k(\mathbf{a},\mathbf{b}) = \{\mathrm{ev}_{\mathbf{a},\mathbf{b}}(f) \mid f \in L_k\}$$

- $\mathcal{F}$ the set of $[n,k]$ GRS codes,
- $\mathcal{S} = \left\{(\mathbf{a},\mathbf{b}) \in \mathbb{F}_q^n \times \mathbb{F}_q^n \mid a_i \neq a_j \text{ and } b_i \neq 0, \ \ \forall i,j \in \{1,\dots,n\}, \ \ i \neq j\right\}$
- $\mathcal{D}(s)$ is our favorite decoder for GRS, e.g. Berlekamp Welch algorithm with $t = \lfloor \frac{n-k}{2} \rfloor$

Example - Alternant codes

Definition: Alternant codes

Let $\mathbf{a} = (a_1, \ldots, a_n) \in \mathbb{F}_{q^m}^n$ be a vector with distinct entries and $\mathbf{b} = (b_1, \ldots, b_n) \in \mathbb{F}_{q^m}^n$ with $b_i \neq 0$. An alternant code of degree r is the code:

$$\mathcal{A}_r(\mathbf{a}, \mathbf{b}) = \mathrm{GRS}_r(\mathbf{a}, \mathbf{b})^{\perp} \cap \mathbb{F}_q^n$$

Example - Alternant codes

Definition: Alternant codes
Let $\mathbf{a} = (a_1, \dots, a_n) \in \mathbb{F}_{q^m}^n$ be a vector with distinct entries and $\mathbf{b} = (b_1, \dots, b_n) \in \mathbb{F}_{q^m}^n$ with $b_i \neq 0$. An alternant code of degree r is the code:

$$\mathcal{A}_r(\mathbf{a}, \mathbf{b}) = \mathrm{GRS}_r(\mathbf{a}, \mathbf{b})^\perp \cap \mathbb{F}_q^n$$

- $\mathcal{F}$ the set of alternant codes of length n and degree r.
- $\mathcal{S} = \left\{(\mathbf{a}, \mathbf{b}) \in \mathbb{F}_{q^m}^n \times \mathbb{F}_{q^m}^n \mid a_i \neq a_j \text{ and } b_i \neq 0, \ \ \forall i, j \in \{1, \dots, n\}, \ \ i \neq j\right\}$
- $\mathcal{D}(s)$ is our favorite decoder for alternant codes, e.g. Berlekamp Welch algorithm.

Example - Classical Goppa codes - McEliece 1978

Definition: Classical Goppa codes

Let $\mathbf{a} \in \mathbb{F}_{q^m}^n$ be a vector with distinct entries and $g \in \mathbb{F}_{q^m}[x]_{\leq t}$ be a polynomial sucha that $g(a_i) \neq 0 \forall i$. The Goppa code associated to $(\mathbf{a}, g)$ is

$$\mathcal{G}(\mathbf{a}, g) = \mathcal{A}_{\deg(g)}(\mathbf{a}, g(\mathbf{a})^{-1}) \cap \mathbb{F}_q^n$$

where $g(\mathbf{a})^{-1} = (g(a_1)^{-}1, \ldots, g(a_n)^{-1})$

.

Example - Classical Goppa codes - McEliece 1978

Definition: Classical Goppa codes
Let $\mathbf{a} \in \mathbb{F}_{q^m}^n$ be a vector with distinct entries and $g \in \mathbb{F}_{q^m}[x]_{\leq t}$ be a polynomial sucha that $g(a_i) \neq 0 \forall i$. The Goppa code associated to $(\mathbf{a}, g)$ is

$$\mathcal{G}(\mathbf{a}, g) = \mathcal{A}_{\deg(g)}(\mathbf{a}, g(\mathbf{a})^{-1}) \cap \mathbb{F}_q^n$$

where $g(\mathbf{a})^{-1} = (g(a_1)^-1, \ldots, g(a_n)^{-1})$

- $\mathcal{F}$ the set of classical Goppa codes of length n and degree r.
- $\mathcal{S} = \left\{(\mathbf{a}, g) \in \mathbb{F}_{q^m}^n \times \mathbb{F}_{q^m}[x]_{\leq t} \mid \ldots\right\}$
- $\mathcal{D}(s)$ is our favorite decoder for Goppa codes.

Example - MDPC codes

Definition: QC-MDPC codes

Let n be a positive even integer and $f, g \in \mathbb{F}_2[x]_{\leq n}$ be two polynomials of weight $\mathcal{O}(\sqrt{n})$. An $[2n, n]$ QC-MPDC code is the kernel of the sparse matrix:

$$\left(\begin{array}{cccc|cccc} f_0 & f_1 & \cdots & f_{n-1} & g_0 & g_1 & \cdots & g_{n-1} \\ f_{n-1} & f_0 & \cdots & f_{n-2} & g_{n-1} & g_0 & \cdots & g_{n-2} \\ \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & \end{array}\right)$$

.

Example - MDPC codes

Definition: QC-MDPC codes

Let n be a positive even integer and $f, g \in \mathbb{F}_2[x]_{\leq n}$ be two polynomials of weight $\mathcal{O}(\sqrt{n})$. An $[2n, n]$ QC-MPDC code is the kernel of the sparse matrix:

$$\left(\begin{array}{cccc|cccc} f_0 & f_1 & \cdots & f_{n-1} & g_0 & g_1 & \cdots & g_{n-1} \\ f_{n-1} & f_0 & \cdots & f_{n-2} & g_{n-1} & g_0 & \cdots & g_{n-2} \\ \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & \end{array}\right)$$

- $\mathcal{F}$ the set of $[2n, n]$ MDPC codes
- $\mathcal{S} = \{(f, g) \in \mathbb{F}_q[x]_{\leq n}$ of weight $\mathcal{O}(\sqrt{n})\}$
- $\mathcal{D}(s)$ is our favorite decoder for MDPC codes, e.g. Bit Flipping algorithm.

Example - Algebraic geometry codes

Definition: Algebraic geometry codes

Let $\mathcal{X}$ be a smooth projective geometrically connected curve over $\mathbb{F}_q$, G be a divisor on $\mathcal{X}$ and $\mathcal{P} = (P_1, \ldots, P_n)$ be a set of $\mathbb{F}_q$-points of $\mathcal{X}$. We define

$$\mathcal{C}_L(\mathcal{X}, \mathcal{P}, G) = \{(f(P_1), \ldots, f(P_n) \mid f \in \mathcal{L}(G)\}$$

Example - Algebraic geometry codes

Definition: Algebraic geometry codes
Let $\mathcal{X}$ be a smooth projective geometrically connected curve over $\mathbb{F}_q$, G be a divisor on $\mathcal{X}$ and $\mathcal{P} = (P_1, \ldots, P_n)$ be a set of $\mathbb{F}_q$-points of $\mathcal{X}$. We define

$$\mathcal{C}_L(\mathcal{X}, \mathcal{P}, G) = \{(f(P_1), \ldots, f(P_n) \mid f \in \mathcal{L}(G)\}$$

- $\mathcal{F}$ the set of AG codes of length n from the curve $\mathcal{X}$
- $\mathcal{S} = \left\{(\mathcal{P}, G) \in \mathcal{X}(\mathbb{F}_q)^n \times \mathrm{Div}_{\mathbb{F}_q}(\mathcal{X}) \mid P_i \neq P_j \forall i \neq j\right\}$
- $\mathcal{D}(s)$ is our favorite decoder for AG codes, e.g. Error Correcting Pairs algorithm.

Efficient Decoding Algorithms

The following classes of codes:

- CGeneralized Reed-Solomon codes (GRS codes).
- Cyclic codes
- Alternant codes
- Goppa codes
- Algebraic geometry codes (AG codes)

... have efficient decoding algorithms:

- Arimoto, Peterson, Gorenstein, Zierler
- Berlekamp, Massey, Sakata
- Justensen et al. Vladut-Skorobatov
- Error-correcting pairs (ECP)

Attacks on the McEliece PKC

We have mainly 2 different ways of cryptanalyzing the McEliece cryptosystem:

1. Generic Decoding Attacks - Message recovery attacks The best known techniques needs **exponential** time in the code length.

2. Structural Attacks - Key recovery attacks Retrieve the code structure rather than use an unspecific decoding algorithm, i.e. recover $s \in \mathcal{S}$ such that the public key $\mathcal{C}_{\mathrm{pub}} = \mathcal{C}(s)$.

   Requirement: Distinguishing a prescribed structure code from a random one.

We focus on **Key Recovery Attacks** on this talk.

Security Proofs of McEliece

We reduce the problem of **Bounded decoding problem** to the security of McEliece under the assumption:

The generator matrix of the public $[n, k]$ code looks random.

That is:

The uniform distribution on the public $[n, k]$ code in family $\mathcal{F}$ is computationally indistinguishable from the uniform distribution on the whole family of $[n, k]$ codes.

Chronology

1978: McEliece - Binary Goppa Codes

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2
2008: Berger, Cayrel, Gaborit, Otmani - QC Alternant codes

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2
2008: Berger, Cayrel, Gaborit, Otmani - QC Alternant codes
2010: Bernstein-Lange-Peters: q-ary "wild" Goppa codes

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2
2008: Berger, Cayrel, Gaborit, Otmani - QC Alternant codes
2010: Bernstein-Lange-Peters: q-ary "wild" Goppa codes
2010: Otmani, Tillich, Dallot, Faugère, Perret, Otmani - Attacks on QC-codes

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2
2008: Berger, Cayrel, Gaborit, Otmani - QC Alternant codes
2010: Bernstein-Lange-Peters: q-ary "wild" Goppa codes
2010: Otmani, Tillich, Dallot, Faugère, Perret, Otmani - Attacks on QC-codes

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2
2008: Berger, Cayrel, Gaborit, Otmani - QC Alternant codes
2010: Bernstein-Lange-Peters: q-ary “wild” Goppa codes
2010: Otmani, Tillich, Dallot, Faugère, Perret, Otmani - Attacks on QC-codes
2010: Wieschebrink's $\mathcal{C} \star \mathcal{C}$ attack

Proposals
Attacks
Broken
Partially Broken

Chronology

Proposals
Attacks
Broken
Partially Broken

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2
2008: Berger, Cayrel, Gaborit, Otmani - QC Alternant codes
2010: Bernstein-Lange-Peters: q-ary "wild" Goppa codes
2010: Otmani, Tillich, Dallot, Faugère, Perret, Otmani - Attacks on QC-codes
2010: Wieschebrink's $\mathcal{C} \star \mathcal{C}$ attack

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2
2008: Berger, Cayrel, Gaborit, Otmani - QC Alternant codes
2010: Bernstein-Lange-Peters: q-ary "wild" Goppa codes
2010: Otmani, Tillich, Dallot, Faugère, Perret, Otmani - Attacks on QC-codes
2010: Wieschebrink's $\mathcal{C} \star \mathcal{C}$ attack
2011: Faugère, Gautier, Otmani, Perret, Tillich - Distinghisher for High rate Goppa codes

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2
2008: Berger, Cayrel, Gaborit, Otmani - QC Alternant codes
2010: Bernstein-Lange-Peters: q-ary "wild" Goppa codes
2010: Otmani, Tillich, Dallot, Faugère, Perret, Otmani - Attacks on QC-codes
2010: Wieschebrink's $\mathcal{C} \star \mathcal{C}$ attack
2011: Faugère, Gautier, Otmani, Perret, Tillich - Distinghisher for High rate Goppa codes
2012: Misoczki, Tillich, Sendrier, Barreto - MDPC codes

Proposals
Attacks
Broken
Partially Broken

Chronology

Proposals
Attacks
Broken
Partially Broken

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2
2008: Berger, Cayrel, Gaborit, Otmani - QC Alternant codes
2010: Bernstein-Lange-Peters: q-ary "wild" Goppa codes
2010: Otmani, Tillich, Dallot, Faugère, Perret, Otmani - Attacks on QC-codes
2010: Wieschebrink's $\mathcal{C} \star \mathcal{C}$ attack
2011: Faugère, Gautier, Otmani, Perret, Tillich - Distinghisher for High rate Goppa codes
2012: Misoczki, Tillich, Sendrier, Barreto - MDPC codes
2014: Couvreur-Márquez-Corbella-Pellikaan - Attacks on AG codes

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2
2008: Berger, Cayrel, Gaborit, Otmani - QC Alternant codes
2010: Bernstein-Lange-Peters: q-ary "wild" Goppa codes
2010: Otmani, Tillich, Dallot, Faugère, Perret, Otmani - Attacks on QC-codes
2010: Wieschebrink's $\mathcal{C} \star \mathcal{C}$ attack
2011: Faugère, Gautier, Otmani, Perret, Tillich - Distinghisher for High rate Goppa codes
2012: Misoczki, Tillich, Sendrier, Barreto - MDPC codes
2014: Couvreur-Márquez-Corbella-Pellikaan - Attacks on AG codes
2014: Couvreur-Otmani-Tillich - Goppa codes with $m = 2$
2014: Faugère-Perret-Portzamparc : Some Goppa codes with $m = 2, 3$

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2
2008: Berger, Cayrel, Gaborit, Otmani - QC Alternant codes
2010: Bernstein-Lange-Peters: q-ary "wild" Goppa codes
2010: Otmani, Tillich, Dallot, Faugère, Perret, Otmani - Attacks on QC-codes
2010: Wieschebrink's $\mathcal{C} \star \mathcal{C}$ attack
2011: Faugère, Gautier, Otmani, Perret, Tillich - Distinghisher for High rate Goppa codes
2012: Misoczki, Tillich, Sendrier, Barreto - MDPC codes
2014: Couvreur-Márquez-Corbella-Pellikaan - Attacks on AG codes
2014: Couvreur-Otmani-Tillich - Goppa codes with $m = 2$
2014: Faugère-Perret-Portzamparc : Some Goppa codes with $m = 2, 3$

Proposals
Attacks
Broken
Partially Broken

Chronology

1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2
2008: Berger, Cayrel, Gaborit, Otmani - QC Alternant codes
2010: Bernstein-Lange-Peters: q-ary "wild" Goppa codes
2010: Otmani, Tillich, Dallot, Faugère, Perret, Otmani - Attacks on QC-codes
2010: Wieschebrink's $\mathcal{C} \star \mathcal{C}$ attack
2011: Faugère, Gautier, Otmani, Perret, Tillich - Distinghisher for High rate Goppa codes
2012: Misoczki, Tillich, Sendrier, Barreto - MDPC codes
2014: Couvreur-Márquez-Corbella-Pellikaan - Attacks on AG codes
2014: Couvreur-Otmani-Tillich - Goppa codes with $m = 2$
2014: Faugère-Perret-Portzamparc : Some Goppa codes with $m = 2, 3$
2014: More attacks on QC and QD codes

Proposals
Attacks
Broken
Partially Broken

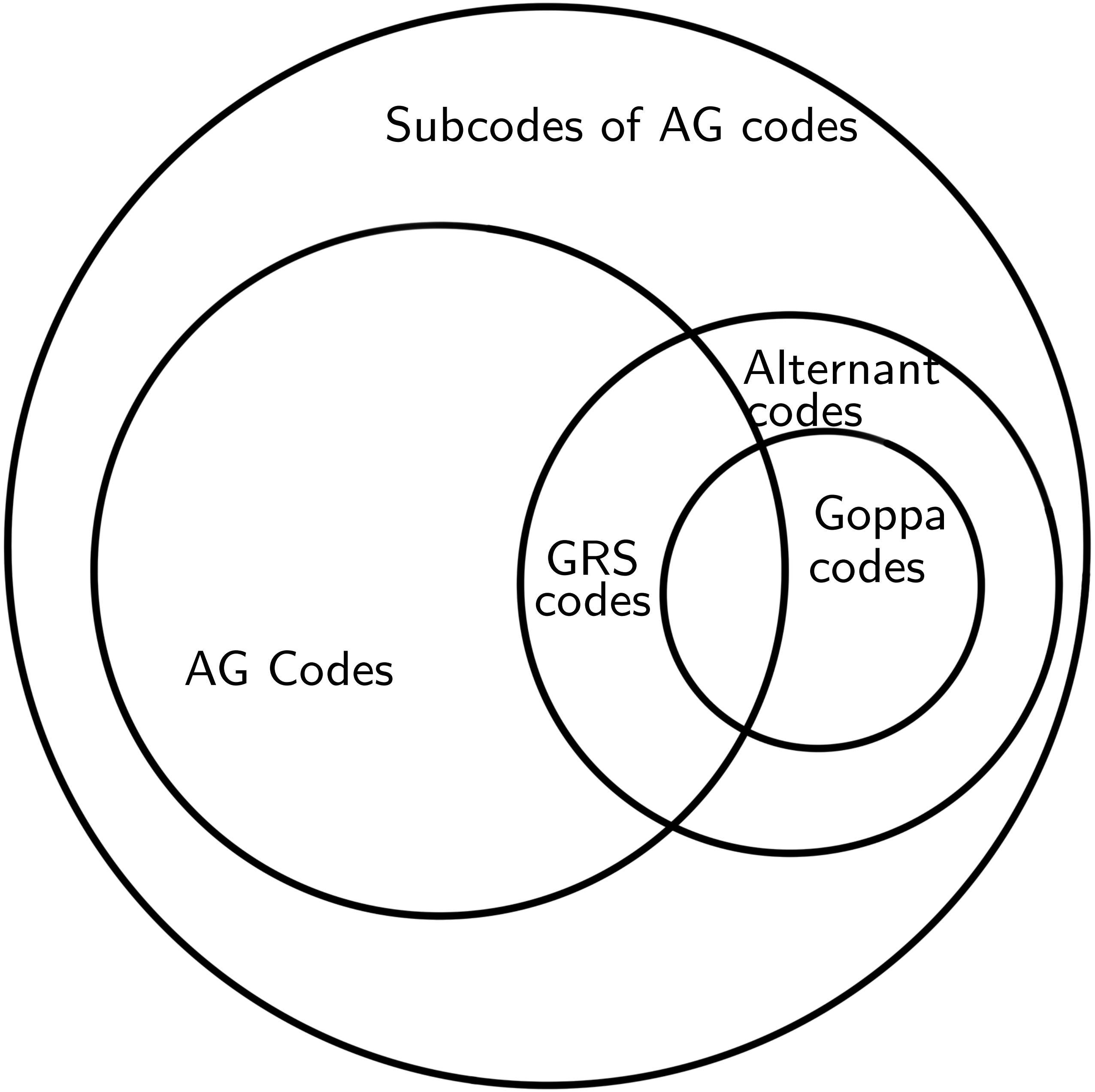
Chronology

Proposals
Attacks
Broken
Partially Broken

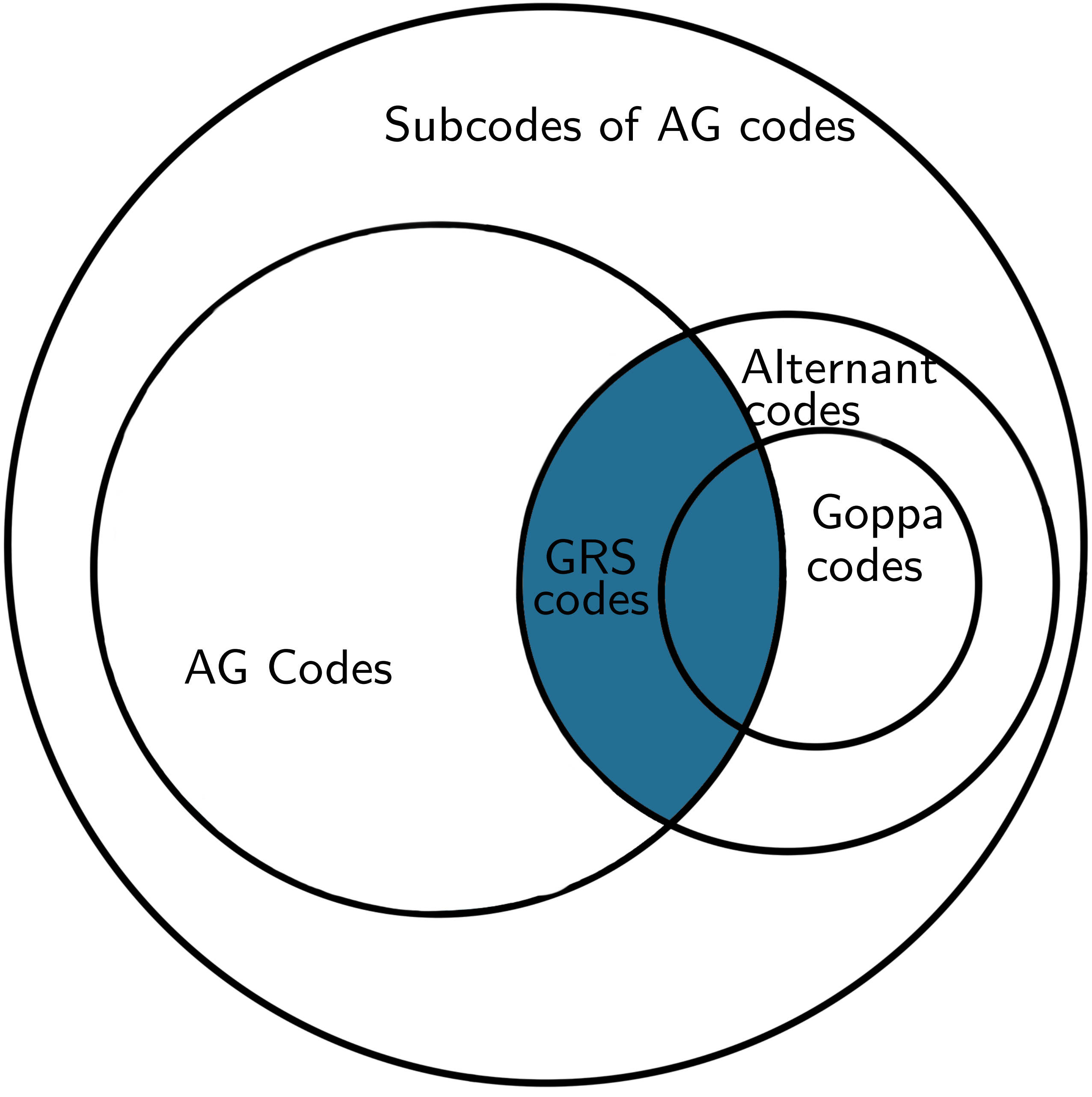
1978: McEliece - Binary Goppa Codes
1986: Niederreiter - GRS codes
1992: Sidelnikov-Shestakov
1994: Sidelnikov-Shestakov - Reed-Muller codes
1996: Janwa-Moreno - AG codes and their subfield subcodes
2001: Berger-Loidreau - Subcodes of GRS codes
2005: Gaborit - Quasi-Cyclic subcodes of BCH codes
2007: Minder Shokrollahi Subexponential time attack on RM codes
2008: Faure-Minder: Attack on AG codes for genus ≤ 2
2008: Berger, Cayrel, Gaborit, Otmani - QC Alternant codes
2010: Bernstein-Lange-Peters: q-ary "wild" Goppa codes
2010: Otmani, Tillich, Dallot, Faugère, Perret, Otmani - Attacks on QC-codes
2010: Wieschebrink's $\mathcal{C} \star \mathcal{C}$ attack
2011: Faugère, Gautier, Otmani, Perret, Tillich - Distinghisher for High rate Goppa codes
2012: Misoczki, Tillich, Sendrier, Barreto - MDPC codes
2014: Couvreur-Márquez-Corbella-Pellikaan - Attacks on AG codes
2014: Couvreur-Otmani-Tillich - Goppa codes with $m = 2$
2014: Faugère-Perret-Portzamparc : Some Goppa codes with $m = 2, 3$
2014: More attacks on QC and QD codes

2017: NIST's call for post-quantum crypto

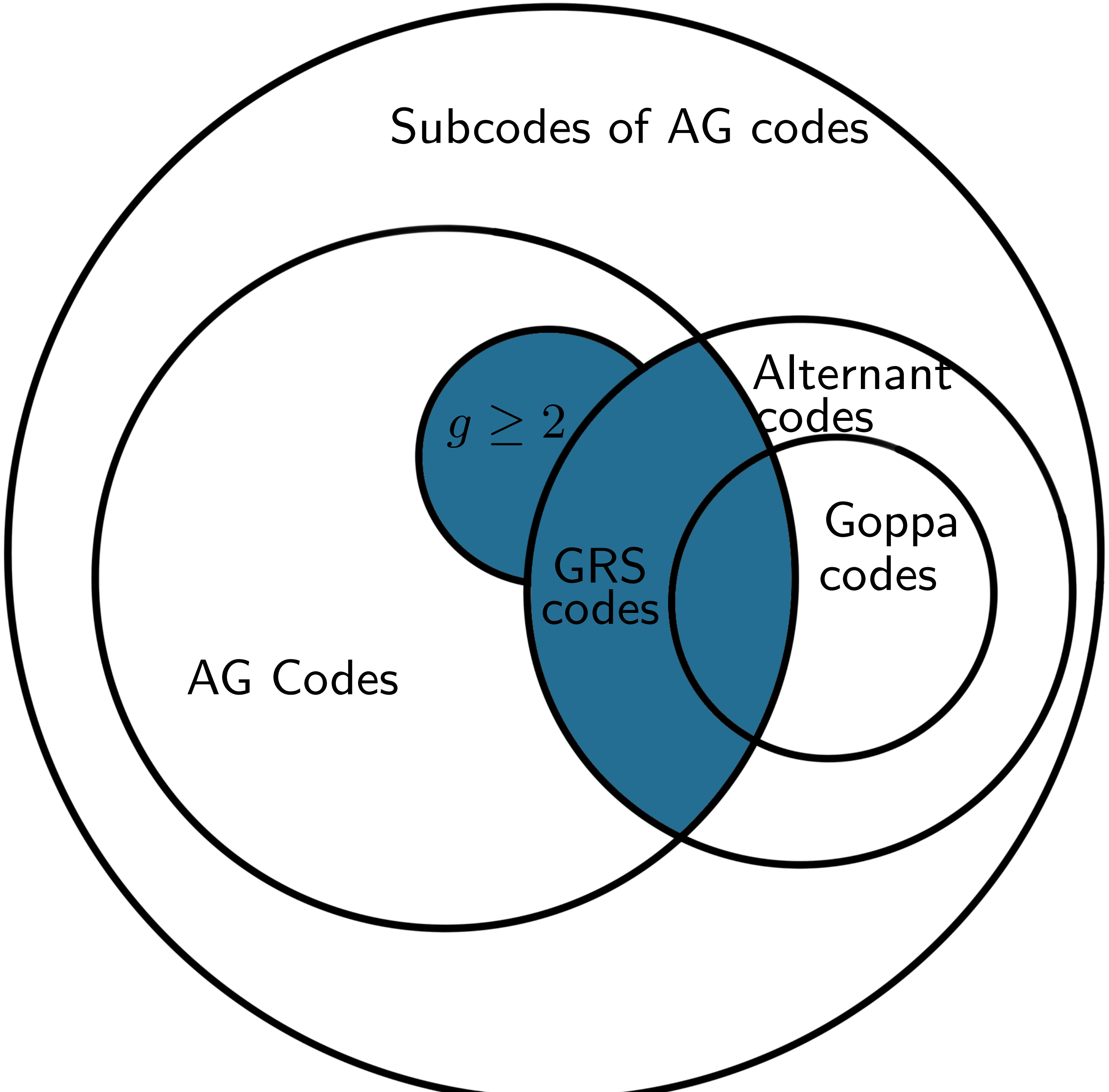
What is still surviving?



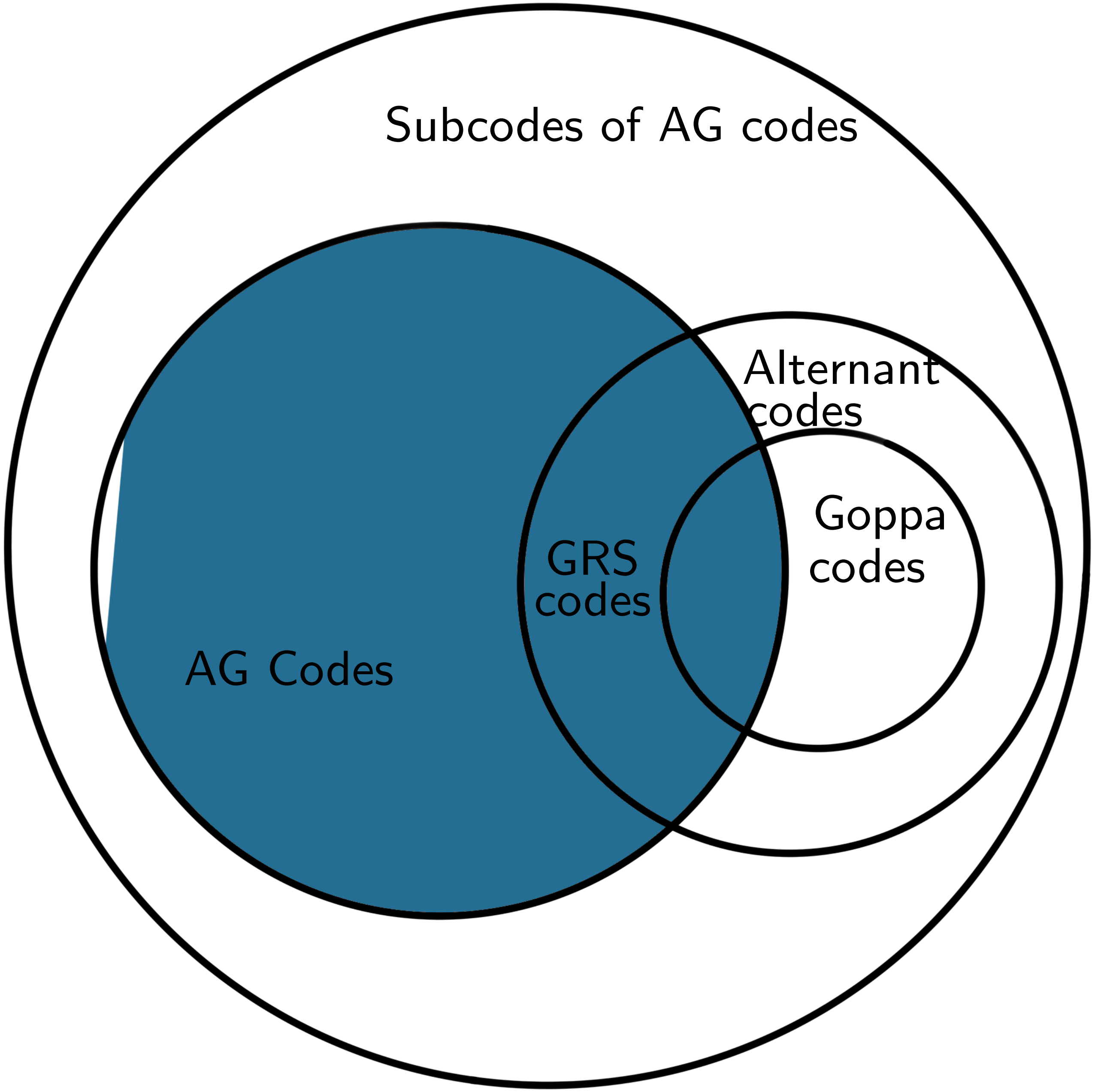
What is still surviving? - Sidelnikov-Shestakov 1992



What is still surviving? - Faurer-Minder 2008

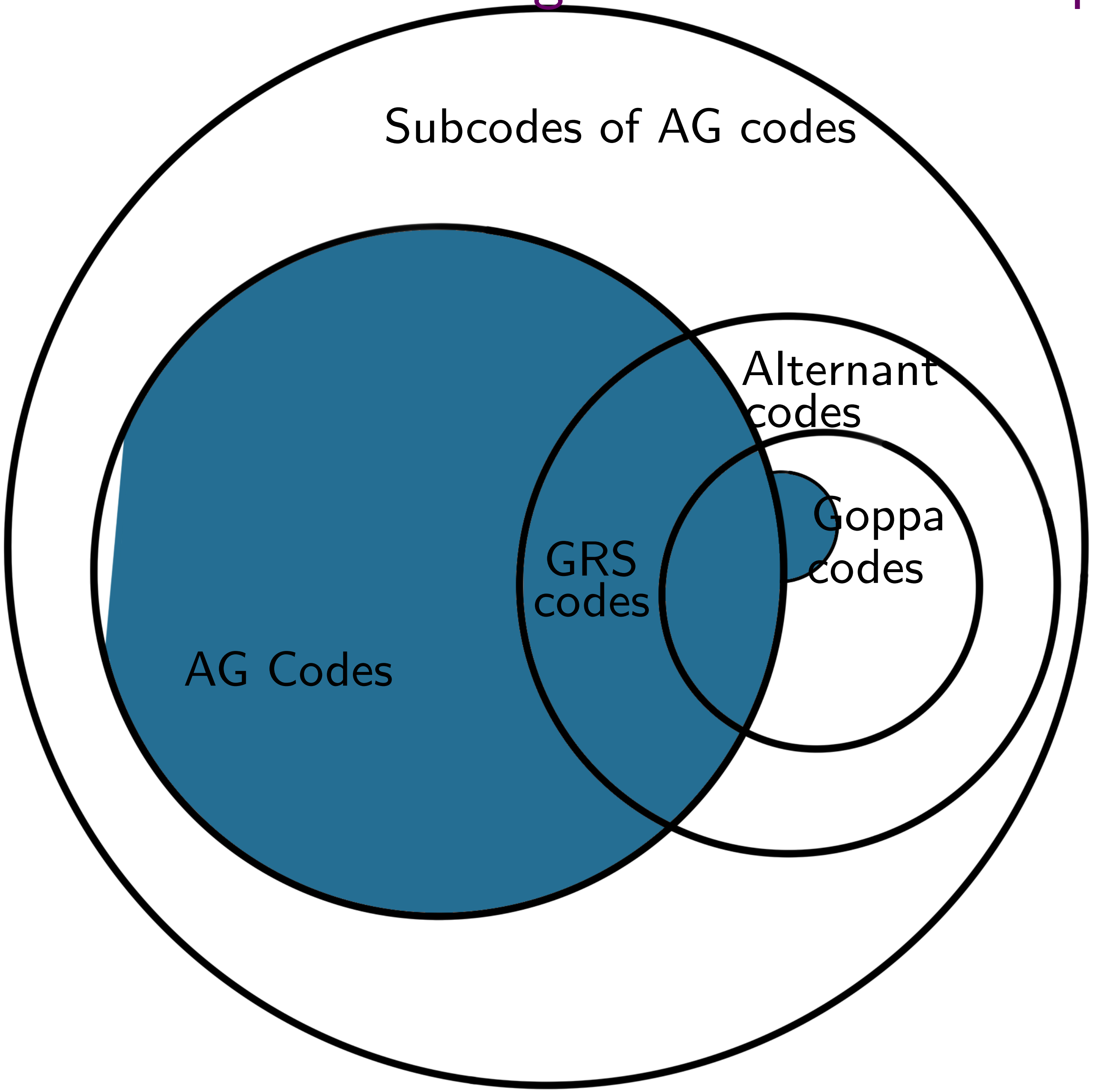


What is still surviving? Couvreur-M.-Pellikaan 2014



What is still surviving?

Couvreur-Otmani- Tillich & Faugère-Perret-Portzamparc 2014



What is still surviving?

Algebraic world:

- Binary Goppa code (NIST's classic McEliece and NTS KEM).
- Goppa codes for $m \geq 2$.
- Goppa codes with "*small*" automorphism group.
- Subfield subcodes of AG codes.

Advantages: Short ciphertexts, no decoding failure.

What is still surviving?

Algebraic world:

- Binary Goppa code (NIST's classic McEliece and NTS KEM).
- Goppa codes for $m \geq 2$.
- Goppa codes with "*small*" automorphism group.
- Subfield subcodes of AG codes.

Advantages: Short ciphertexts, no decoding failure.

Probabilistic world:

- QC MDPC codes

Advantages: Short Keys

What is still surviving?

Algebraic world:

– Binary Goppa code (NIST's classic McEliece and NTS KEM).
– Goppa codes for $m \geq 2$.
– Goppa codes with "*small*" automorphism group.
– Subfield subcodes of AG codes.

Advantages: Short ciphertexts, no decoding failure.

Probabilistic world:

– QC MDPC codes

Advantages: Short Keys

Promising alternatives:

– HQC, RQC.

◦ **Advantages:** do not rely on indistinguishability. Promising application of algebraic codes.

Original Proposal - Binary Goppa codes

1978: McEliece's - Binary Goppa codes

Public Key Size: 32kB for 65 bits of security

(with respect to Prange algorithm).

2018: NIST proposals with Binary Goppa codes:

– **Classic McEliece**
  Public Key Size: 1-1.3MByte for > 256 bits of security.
– **NTS KEM** 319 KBytes for > 128 bits security.

Original Proposal - Binary Goppa codes

1978: McEliece's - Binary Goppa codes

Public Key Size: 32kB for 65 bits of security

(with respect to Prange algorithm).

2018: NIST proposals with Binary Goppa codes:

- **Classic McEliece**
  Public Key Size: 1-1.3MByte for > 256 bits of security.
- **NTS KEM** 319 KBytes for > 128 bits security.

During these 40 years many attempts to get shorter keys... but **HOW?**

IDEA 1 : Reducing the extension degree

Definition: Alternant codes

Let $\mathbf{a} = (a_1, \ldots, a_n) \in \mathbb{F}_{q^m}^n$ be a vector with distinct entries and $\mathbf{b} = (b_1, \ldots, b_n) \in \mathbb{F}_{q^m}^n$ with $b_i \neq 0$. An alternant code of degree r is the code:

$$\mathcal{A}_r(\mathbf{a}, \mathbf{b}) = \mathrm{GRS}_r(\mathbf{a}, \mathbf{b})^{\perp} \cap \mathbb{F}_q^n$$

IDEA 1 : Reducing the extension degree

Definition: Alternant codes
Let $\mathbf{a} = (a_1, \ldots, a_n) \in \mathbb{F}_{q^m}^n$ be a vector with distinct entries and $\mathbf{b} = (b_1, \ldots, b_n) \in \mathbb{F}_{q^m}^n$ with $b_i \neq 0$. An alternant code of degree r is the code:

$$\mathcal{A}_r(\mathbf{a}, \mathbf{b}) = \mathrm{GRS}_r(\mathbf{a}, \mathbf{b})^\perp \cap \mathbb{F}_q^n$$

Fact. The larger the m the worse the parameters. But:

- The case $m = 1$ is broken (Sidelnikov-Shestakov 1992).
- Some specific cases of m=2 and 3 called wild Goppa codes are broken too:
  - Couvreur, Otmani, Tillich, 2014.
  - Faugère, Perret, de Portzamparc, 2014.

IDEA 1 : Reducing the extension degree

Further construction from GRS codes

- **2001:** Berger-Loidreau

  Subcodes of GRS codes.
- **2006:** Wieschebrink

  Adds random columns in GRS code's generator matrix.
- **2013:** Baldi, Bianchi, Chiaraluce, Rosenthal, Schipani

  Multiply the GRS code by a sparse matrix.
- **2016:** Wang's RLCE system

  Replaces some columns of a GRS's generator matrix by linear combinations of GRS and random columns.

IDEA 2 : Using codes with a non trivial automorphism group

Definition: Given a code $\mathcal{C} \subseteq \mathbb{F}_q^n$ with a group action $\mathcal{G}$, one can define the invariant code:

$$\mathcal{C}^{\mathcal{G}} = \{\mathbf{x} \in \mathcal{C} \mid \forall \sigma \in \mathcal{G} \sigma(\mathbf{x}) = \mathbf{x}\}$$

If the action of $\mathcal{G}$ is public, then $\mathcal{C}^{\mathcal{G}}$ is computable in polynomial time.

IDEA 2 : Using codes with a non trivial automorphism group

Definition: Given a code $\mathcal{C} \subseteq \mathbb{F}_q^n$ with a group action $\mathcal{G}$, one can define the invariant code:

$$\mathcal{C}^{\mathcal{G}} = \{\mathbf{x} \in \mathcal{C} \mid \forall \sigma \in \mathcal{G} \sigma(\mathbf{x}) = \mathbf{x}\}$$

If the action of $\mathcal{G}$ is public, then $\mathcal{C}^{\mathcal{G}}$ is computable in polynomial time.

In 2005 Gaborit proposes to use codes with a non-trivial automorphism group $\mathcal{G}$:

- Quasi-cyclic codes (QC-codes) : $\mathcal{G} = \mathbb{Z}/l\mathbb{Z}$
- Quasi-dyadic codes (QC-codes) : $\mathcal{G} = (\mathbb{Z}/2\mathbb{Z})^{\gamma}$

IDEA 2 : Using codes with a non trivial automorphism group

Definition: Given a code $\mathcal{C} \subseteq \mathbb{F}_q^n$ with a group action $\mathcal{G}$, one can define the invariant code:

$$\mathcal{C}^{\mathcal{G}} = \{\mathbf{x} \in \mathcal{C} \mid \forall \sigma \in \mathcal{G} \sigma(\mathbf{x}) = \mathbf{x}\}$$

If the action of $\mathcal{G}$ is public, then $\mathcal{C}^{\mathcal{G}}$ is computable in polynomial time.

In 2005 Gaborit proposes to use codes with a non-trivial automorphism group $\mathcal{G}$:

- Quasi-cyclic codes (QC-codes) : $\mathcal{G} = \mathbb{Z}/l\mathbb{Z}$
- Quasi-dyadic codes (QC-codes) : $\mathcal{G} = (\mathbb{Z}/2\mathbb{Z})^{\gamma}$

- Advantage: Permits to reduce the public key size.
- Advantage: No incidence on the security w.r.t. generic decoding.
- Disadvantage: Affect the security w.r.t. key recovery attacks.

IDEA 2 : Using codes with a non trivial automorphism group

Some key recovery attacks:

- QC-BCH codes
  Otmani-Tillich-Dallot (2008)

- QC-Alternant codes
  Faugère - Otmani - Perret - Tillich (2010)

- QC and QD - Alternant codes
  Faugère - Otmani - Perret - Tillich - Portzamparc (2016)

- QD - Alternant codes (DAGS)
  Barelli - Couvreur (2018)

Outline

1. History of code-based cryptography

2. Algebraic cryptanalysis in code-based cryptography

   2.1. Sidelnikov-Shestakov like attack.
   2.2. Algebraic attacks by solving a polynomial system.
   2.3. $\star$-product.

3. How to design secure schemes with codes?

Sidelnikov-Shestakov - 1992

Theorem: Sidelnikov-Shestakov
Given as input any matrix G generating the code $\mathrm{GRS}_k(\mathbf{a}, \mathbf{b})$, there exists an algorithm running in time $\mathcal{O}(n^4)$ that outputs $\mathbf{a}', \mathbf{b}'$ such that:

$$\mathrm{GRS}_k(\mathbf{x}, \mathbf{y}) = \mathrm{GRS}_k(\mathbf{x}', \mathbf{y}')$$

Moreover,

$$\mathbf{x} = \tfrac{a\mathbf{x}+b\mathbf{1}}{c\mathbf{x}+d\mathbf{1}} \quad \text{and} \quad \mathbf{y}' = \lambda \mathbf{y}$$

Sidelnikov-Shestakov - 1992

Public Key: $\mathcal{C} \subseteq \mathrm{GRS}_k(\mathbf{x}, \mathbf{y})$

Secret Key: $s = (\mathbf{x}, \mathbf{y})$

Sidelnikov-Shestakov Attack:

Step 1. In our search for $\mathbf{x}$, one can arbitrarily fix 3 points, say:

$$x_{n-2} = 1, \quad x_{n-1} = 0 \quad \text{and} \quad x_n = \infty$$

Step 2. From a generator matrix G of a code $\mathrm{GRS}_k(\mathbf{x}, \mathbf{y})$ compute two minimum weight codewords whose supports are close.

From G, by Gaussian elimination we can find:

$$\begin{array}{lllllllllll} \mathbf{u} = (& 0 & \dots & 0 & 0 & 1 & u_{k+1} & \dots & u_n &) & \longrightarrow & f(x) \\ \mathbf{v} = (& 0 & \dots & 0 & 1 & 0 & v_{k+1} & \dots & v_n &) & \longrightarrow & g(x) \end{array}$$

Sidelnikov-Shestakov - 1992

$$\begin{array}{lcccccccccl} \mathbf{u} = (& 0 & \dots & 0 & 0 & 1 & u_{k+1} & \dots & u_n &) & \longrightarrow & f(x) \\ \mathbf{v} = (& 0 & \dots & 0 & 1 & 0 & v_{k+1} & \dots & v_n &) & \longrightarrow & g(x) \end{array}$$

Lemma If two elements $f, g \in \mathbb{F}_q[x]_{\leq k}$ share $k-2$ zeroes, then

$$\phi(x) = \frac{f(x)}{g(x)} = \frac{\alpha x + \beta}{\gamma x + \delta}$$

$$\mathbf{u} \star \mathbf{v}^{-1} = (\quad 0 \quad \dots \quad 0 \quad \perp \quad \frac{u_{k+1}}{v_{k+1}} \quad \dots \quad \frac{u_n}{v_n} \quad) \quad \longrightarrow \quad f(x)$$

1. **Solve** in $(\alpha, \beta, \gamma, \delta)$ the system $\phi(x_i) = \frac{u_i}{v_i}$ with $i = n-2, n-1, n$ we find ϕ
2. **Solve** the equation $\phi(x_i) = \frac{u_i}{v_i}$ for each $i \in [k+1, n-3]$ we find $x_{k+1}, \dots, x_{n-3}$

Step 3. Once $\mathbf{x}$ is known, one can easily find a valid $\mathbf{y}$ by solving a linear system.

Sidelnikov-Shestakov - 1992

Remark: Computing minimum weight codewords is hard but...

is only **Gaussian elimination** for **GRS codes**!!!

Sidelnikov-Shestakov - 1992

Remark: Computing minimum weight codewords is hard but…

is only **Gaussian elimination** for **GRS codes**!!!

Some attacks deriving form Sidelnikov-Shestakov:

- Minder-Shokrollahi (2007) - Broke Sidelnikov's proposal based on binary Reed-Muller codes.

  Subexponential time attack

- Faure-Minder (2008) - Broke AG codes from hyperelliptic curves.

  The attack has exponential cost in the curve's genus

  In orange due to the cost of computing minimum weight codewords.

Algebraic attacks by polynomial system solving

Idea: A code $\mathcal{A}_r(\mathbf{x}, \mathbf{y})$ is contained in the kernel of a matrix of the form:

$$H = \begin{pmatrix} y_1 & \cdots & y_n \\ x_1 y_1 & \cdots & x_n y_n \\ \vdots & \ddots & \vdots \\ x_1^{r-1} y_1 & \cdots & x_n^{r-1} y_n \end{pmatrix}$$

Put x_i, y_i as formal variables X_i, Y_i and solve the polynomial system:

$$H(X_i, Y_i)^T \cdot G = 0$$

For usual McEliece parameters, the resolution of such a polynomial system is out of reach. But...

Algebraic attacks by polynomial system solving

Idea: A code $\mathcal{A}_r(\mathbf{x}, \mathbf{y})$ is contained in the kernel of a matrix of the form:

$$H = \begin{pmatrix} y_1 & \cdots & y_n \\ x_1 y_1 & \cdots & x_n y_n \\ \vdots & \ddots & \vdots \\ x_1^{r-1} y_1 & \cdots & x_n^{r-1} y_n \end{pmatrix}$$

Put x_i, y_i as formal variables X_i, Y_i and solve the polynomial system:

$$H(X_i, Y_i)^T \cdot G = 0$$

For usual McEliece parameters, the resolution of such a polynomial system is out of reach. But...

- Attacks on QC and QD Alternant codes
  Faugère- Otmani- Perret- Portzamparc, Tillich (2010)

$\star$ product

For all $\mathbf{a}, \mathbf{b} \in \mathbb{F}_q^n$ we define

$\star$ **product:**

$$\mathbf{a} * \mathbf{b} = (a_1 b_1, \ldots, a_n b_n) \in \mathbb{F}_q^n$$

$\star$ **product of two codes:** Let $A,\ B \subseteq \mathbb{F}_q^n$ we define

$$A * B = \langle \{\mathbf{a} * \mathbf{b} \mid \mathbf{a} \in A \text{ and } \mathbf{b} \in B\} \rangle$$

For $B = A$ then we denote by $A^2 = A * A$

$\star$ product - Attack

Theorem: Cascudo-Cramer-Mirandola-Zémor 2013
Let $\mathcal{C}$ be a random $[n, k]$-code then

$$\mathrm{Prob}\left(\dim(\mathcal{C}^2) < \min\left(n, \binom{k+1}{2}\right)\right) \xrightarrow{n\to\infty} 0$$

$\star$ product - Attack

Theorem: Cascudo-Cramer-Mirandola-Zémor 2013

Let $\mathcal{C}$ be a random $[n,k]$-code then

$$\mathrm{Prob}\left(\dim(\mathcal{C}^2) < \min\left(n, \binom{k+1}{2}\right)\right) \xrightarrow{n\to\infty} 0$$

Proposition:

- If $k \leq \frac{n+1}{2}$. Then, $\mathrm{GRS}_k(\mathbf{a},\mathbf{b})^2 = \mathrm{GRS}_{2k-1}(\mathbf{a},\mathbf{b}*\mathbf{b})$
- If $k > \frac{n+1}{2}$ then,
  $\left(\mathrm{GRS}_k(\mathbf{a},\mathbf{b})^{\perp}\right)^2 = \mathrm{GRS}_{2(n-k)-1}(\mathbf{a},\mathbf{b}^{\perp}*\mathbf{b}^{\perp})$

$\star$ product - Attack

Theorem: Cascudo-Cramer-Mirandola-Zémor 2013
Let $\mathcal{C}$ be a random $[n,k]$-code then

$$\mathrm{Prob}\left(\dim(\mathcal{C}^2) < \min\left(n, \binom{k+1}{2}\right)\right) \xrightarrow{n\to\infty} 0$$

Proposition:

- If $k \leq \frac{n+1}{2}$. Then, $\mathrm{GRS}_k(\mathbf{a},\mathbf{b})^2 = \mathrm{GRS}_{2k-1}(\mathbf{a},\mathbf{b}*\mathbf{b})$
- If $k > \frac{n+1}{2}$ then,
  $\left(\mathrm{GRS}_k(\mathbf{a},\mathbf{b})^{\perp}\right)^2 = \mathrm{GRS}_{2(n-k)-1}(\mathbf{a},\mathbf{b}^{\perp}*\mathbf{b}^{\perp})$

Proposition: Similar result for AG codes

$$\mathcal{C}_L(\mathcal{X},\mathcal{P},G)^2 = \mathcal{C}_L(\mathcal{X},\mathcal{P},2G)$$

under some condition on $\deg(G)$.

First use of $\star$-product - Wieschebrink 2010

- Berger-Loidreau propose in 2005 to use **subcodes of GRS codes**
- This proposal was broken by Wieschebrink in 2010.

First use of $\star$-product - Wieschebrink 2010

- Berger-Loidreau propose in 2005 to use **subcodes of GRS codes**
- This proposal was broken by Wieschebrink in 2010.

Public Key: $\mathcal{C} \subseteq \mathrm{GRS}_k(\mathbf{a}, \mathbf{b})$

Secret Key: $s = (\mathbf{a}, \mathbf{b})$

Fact: With high probability:

$$\mathcal{C}^2 = \mathrm{GRS}_k(\mathbf{a}, \mathbf{b})^2 = \mathrm{GRS}_{2k-1}(\mathbf{a}, \mathbf{b} * \mathbf{b})$$

First use of $\star$-product - Wieschebrink 2010

- Berger-Loidreau propose in 2005 to use **subcodes of GRS codes**
- This proposal was broken by Wieschebrink in 2010.

Public Key: $\mathcal{C} \subseteq \mathrm{GRS}_k(\mathbf{a}, \mathbf{b})$

Secret Key: $s = (\mathbf{a}, \mathbf{b})$

Fact: With high probability:

$$\mathcal{C}^2 = \mathrm{GRS}_k(\mathbf{a}, \mathbf{b})^2 = \mathrm{GRS}_{2k-1}(\mathbf{a}, \mathbf{b} * \mathbf{b})$$

Wieschebrink's attack:

Step 1. Compute $\mathcal{C}^2$

Step 2. Perform Sidelnikov-Shestakov attack on $\mathcal{C}^2$ to recover $(\mathbf{a}, \mathbf{b} * \mathbf{b})$

Step 3. Deduce $(\mathbf{a}, \mathbf{b})$.

Filtration Attack (Application of $\star$-product)

Illustrative example on GRS codes

Suppose we know the codes

$$\mathcal{C}_k = \mathrm{GRS}_k(\mathbf{a}, \mathbf{b}) \quad \text{and} \quad \mathcal{C}_{k-1} = \mathrm{GRS}_{k-1}(\mathbf{a}, \mathbf{b})$$

Filtration Attack (Application of $\star$-product)

Illustrative example on GRS codes

Suppose we know the codes

$$\mathcal{C}_k = \mathrm{GRS}_k(\mathbf{a}, \mathbf{b}) \quad \text{and} \quad \mathcal{C}_{k-1} = \mathrm{GRS}_{k-1}(\mathbf{a}, \mathbf{b})$$

Proposition: If $2k-1 \leq n-2$, then:

$$\mathcal{C}_{k-2} = \mathrm{GRS}_{k-2}(\mathbf{a}, \mathbf{b})$$

can be computed as the set

$$\mathbf{c} \in \mathcal{C}_{k-1} \quad \text{y} \quad \mathbf{c} * \mathcal{C}_k \subseteq \left(\mathcal{C}_{k-1}\right)^2$$

.

Filtration Attack (Application of $\star$-product)

Illustrative example on GRS codes

Suppose we know the codes

$$\mathcal{C}_k = \mathrm{GRS}_k(\mathbf{a}, \mathbf{b}) \quad \text{and} \quad \mathcal{C}_{k-1} = \mathrm{GRS}_{k-1}(\mathbf{a}, \mathbf{b})$$

Proposition: If $2k-1 \leq n-2$, then:

$$\mathcal{C}_{k-2} = \mathrm{GRS}_{k-2}(\mathbf{a}, \mathbf{b})$$

can be computed as the set

$$\mathbf{c} \in \mathcal{C}_{k-1} \quad \text{y} \quad \mathbf{c} * \mathcal{C}_k \subseteq (\mathcal{C}_{k-1})^2$$

Proof: [Sketch]

$$\begin{array}{ccccccc} & \mathcal{C}_{k-1} & * & \mathcal{C}_k & = & (\mathcal{C}_{k-1})^2 \\ & (\mathbf{b} * f(\mathbf{a})) & * & (\mathbf{b} * g(\mathbf{a})) & = & (\mathbf{b} * \mathbf{b})\,(fg)(\mathbf{a}) \\ \text{with} & \deg(f) < k-1 & , & \deg(g) < k & \Rightarrow & \deg(fg) < 2k-2 \end{array}$$

Filtration Attack (Application of $\star$-product)

Illustrative example on GRS codes

Suppose we know the codes

$$\mathcal{C}_k = \mathrm{GRS}_k(\mathbf{a}, \mathbf{b}) \quad \text{and} \quad \mathcal{C}_{k-1} = \mathrm{GRS}_{k-1}(\mathbf{a}, \mathbf{b})$$

Proposition: If $2k - 1 \leq n - 2$, then:

$$\mathcal{C}_{k-2} = \mathrm{GRS}_{k-2}(\mathbf{a}, \mathbf{b})$$

can be computed as the set

$$\mathbf{c} \in \mathcal{C}_{k-1} \quad \text{y} \quad \mathbf{c} * \mathcal{C}_k \subseteq \left(\mathcal{C}_{k-1}\right)^2$$

Then, reiterate the process we deduce the filtration:

$$\mathrm{GRS}_k(\mathbf{a}, \mathbf{b}) \supseteq \mathrm{GRS}_{k-1}(\mathbf{a}, \mathbf{b}) \supseteq \mathrm{GRS}_{k-2}(\mathbf{a}, \mathbf{b}) \supseteq \cdots \supseteq \mathrm{GRS}_1(\mathbf{a}, \mathbf{b})$$

Thus we get: $\mathrm{GRS}_1(\mathbf{a}, \mathbf{b}) = \left\{\alpha \mathbf{b} \mid \alpha \in \mathbb{F}_q^*\right\}$

Where we can deduce $\mathbf{b}$ and $\mathbf{a}$ solving a linear system.

Filtration Attack (Application of $\star$-product)

Illustrative example on GRS codes

Suppose we know the codes

$$\mathcal{C}_k = \mathrm{GRS}_k(\mathbf{a}, \mathbf{b}) \quad \text{and} \quad \mathcal{C}_{k-1} = \mathrm{GRS}_{k-1}(\mathbf{a}, \mathbf{b})$$

Proposition: If $2k - 1 \le n - 2$, then:

$$\mathcal{C}_{k-2} = \mathrm{GRS}_{k-2}(\mathbf{a}, \mathbf{b})$$

can be computed as the set

$$\mathbf{c} \in \mathcal{C}_{k-1} \quad \text{y} \quad \mathbf{c} * \mathcal{C}_k \subseteq (\mathcal{C}_{k-1})^2$$

Then, reiterate the process we deduce the filtration:

$$\mathrm{GRS}_k(\mathbf{a}, \mathbf{b}) \supseteq \mathrm{GRS}_{k-1}(\mathbf{a}, \mathbf{b}) \supseteq \mathrm{GRS}_{k-2}(\mathbf{a}, \mathbf{b}) \supseteq \cdots \supseteq \mathrm{GRS}_1(\mathbf{a}, \mathbf{b})$$

Remark: We do not need to know both $\mathrm{GRS}_k(\mathbf{a}, \mathbf{b})$ and $\mathrm{GRS}_{k-1}(\mathbf{a}, \mathbf{b})$ but $\mathrm{GRS}_{k-1}(\mathbf{a}, \mathbf{b})$ can be replaced by a shortening of $\mathrm{GRS}_k(\mathbf{a}, \mathbf{b})$ at one position.

Filtration Attack (Application of $\star$-product)

Same idea is behind:

- Alternative attack on GRS codes

  Couvreur-Gautier-Gaborit-Otmani-Tillich (2015)
- AG codes and their subcodes

  Couvreur-M.-Pellikaan (2014-17)
- Wild Goppa codes for $m = 2$

  Couvreur-Otmani-Tillich (2014-17)

Remark: No more need to compute minimum weight codewords!!

Outline

1. History of code-based cryptography

2. Algebraic cryptanalysis in code-based cryptography

3. How to design secure schemes with codes?

How to evaluate the security of algebraic codes?

- **Sufficiently many codes in the family**
  Support Splitting Algorithm (N. Sendrier).

- **Low weight codewords should be hard to compute.**
  Avoid Sidelnikov-Shestakov like attacks.

- **No square code distinguiser**.
  – $\mathcal{C}^2$, $(\mathcal{C}^{\perp})^2$ should behave like random codes.
  – Also their shortenings.

- And if you use some automorphism group, check the above properties for both $\mathcal{C}$ and $\mathcal{C}^{\mathcal{G}}$.

- It should resist attacks by algebraic systems solving – This is difficult to analyze.

Thanks