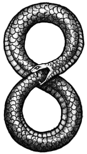
Ouroboros: a simple, secure and efficient key exchange protocol based on coding theory

Jean-Christophe Deneuville
<jean-christophe.deneuville@xlim.fr>

June the 26th, 2017
PQCrypto'17
Utrecht



Joint work with:

P. Gaborit
University of Limoges

G. Zémor
University of Bordeaux

Motivations

[ME78]

Motivations

[ME78] $\longrightarrow$ [Nie86]

Motivations

[ME78] $\longrightarrow$ [Nie86] $\longrightarrow$

RS
BCH
Goppa
RM

$\Bigg\}$ 80's $\downarrow$ 00's

Motivations

Key Sizes

[ME78] ⟶ [Nie86] ⟶ RS, BCH, Goppa, RM } 80's ↓ 00's

Security reduction to a standard problem (random codes)

Motivations

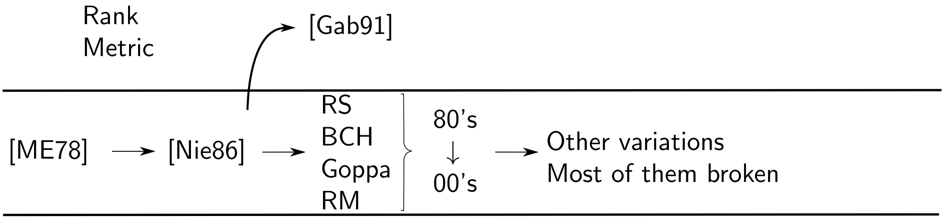
Key Sizes

[ME78] → [Nie86] → RS, BCH, Goppa, RM (80's ↓ 00's) → Other variations
Most of them broken

Security proof

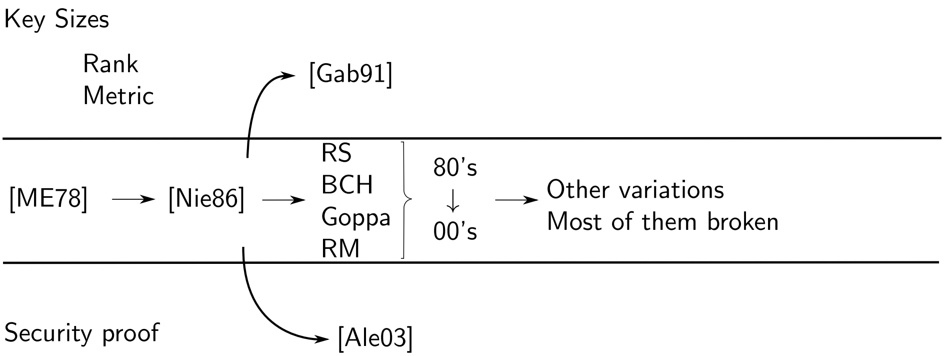
Motivations

Key Sizes

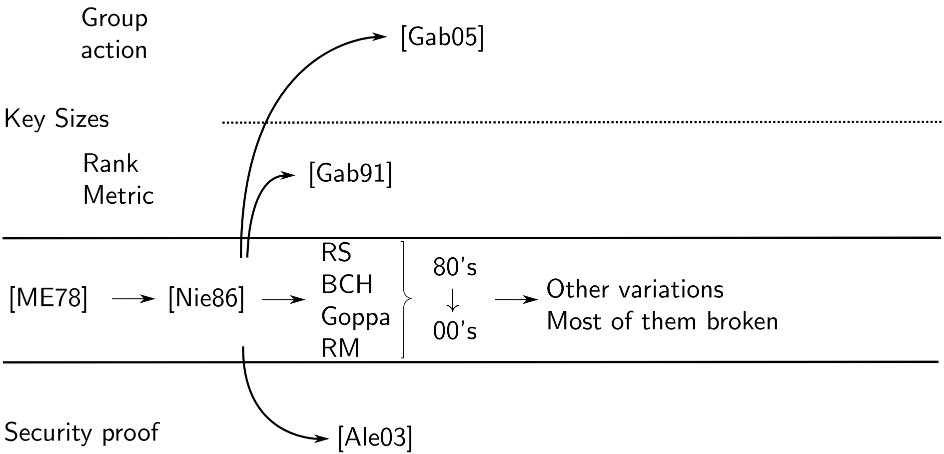


Security proof

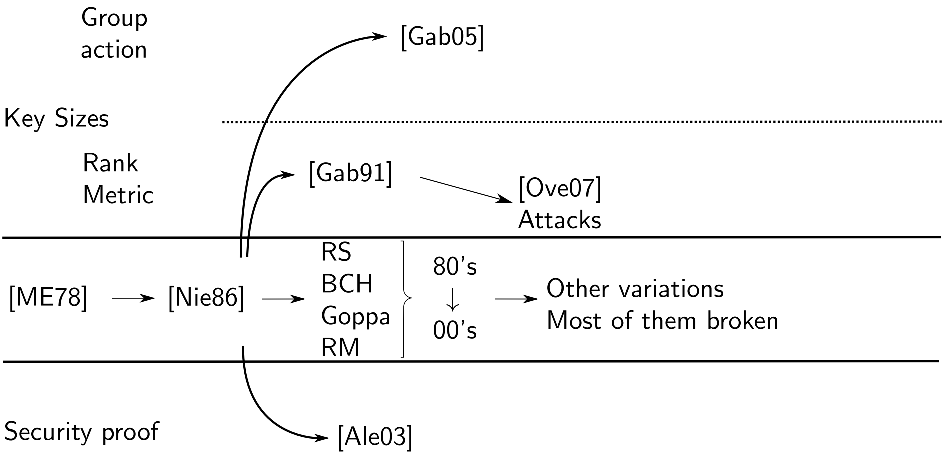
Motivations



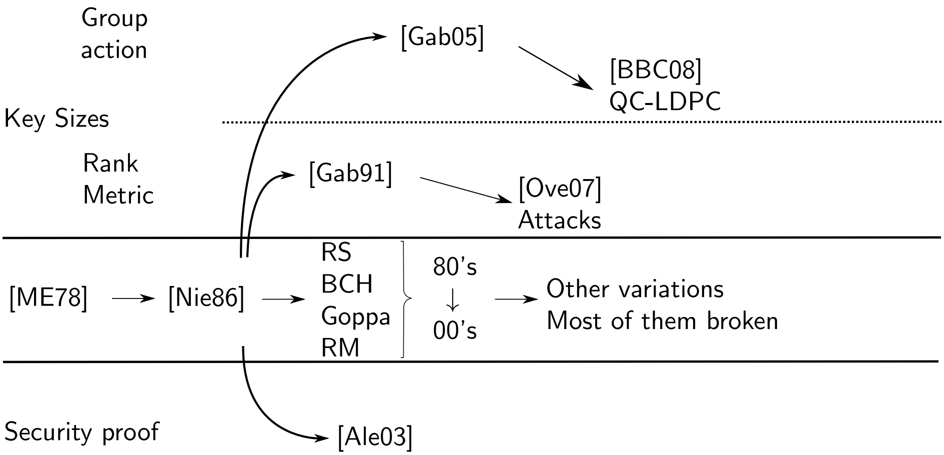
Motivations



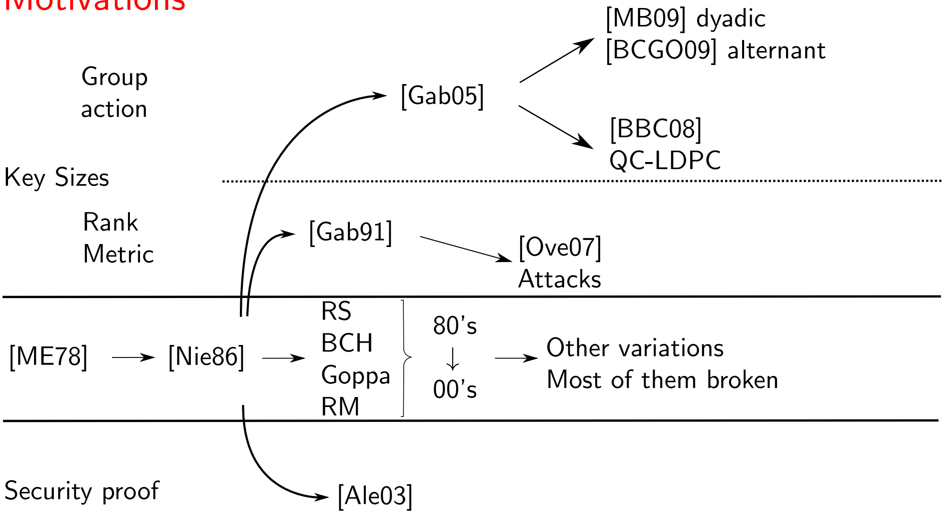
Motivations



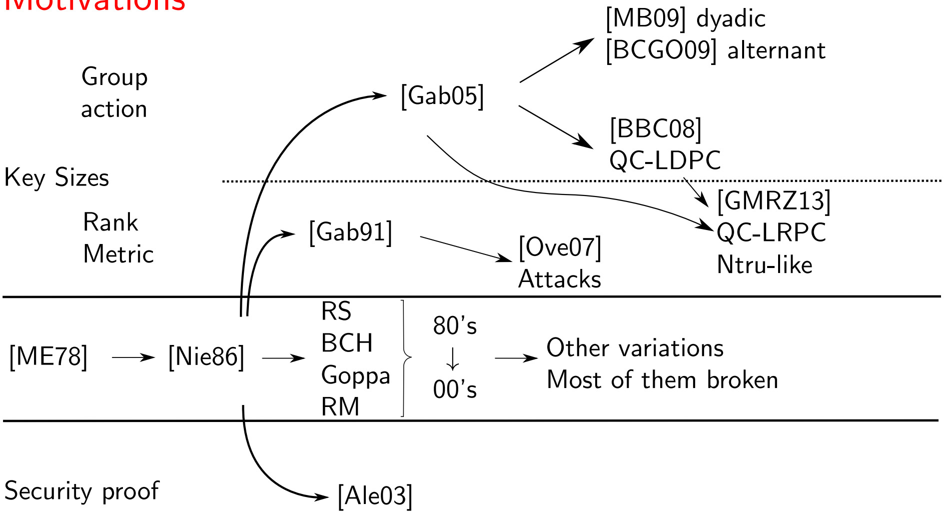
Motivations



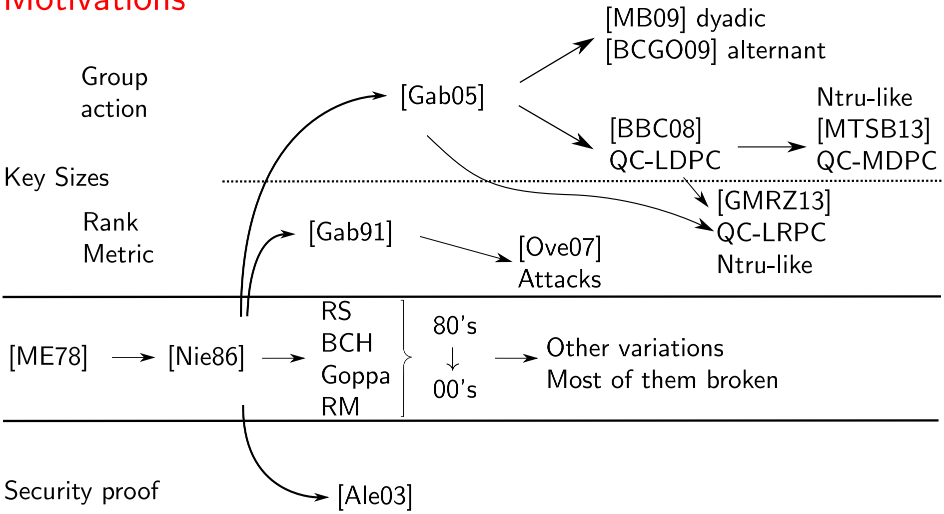
Motivations



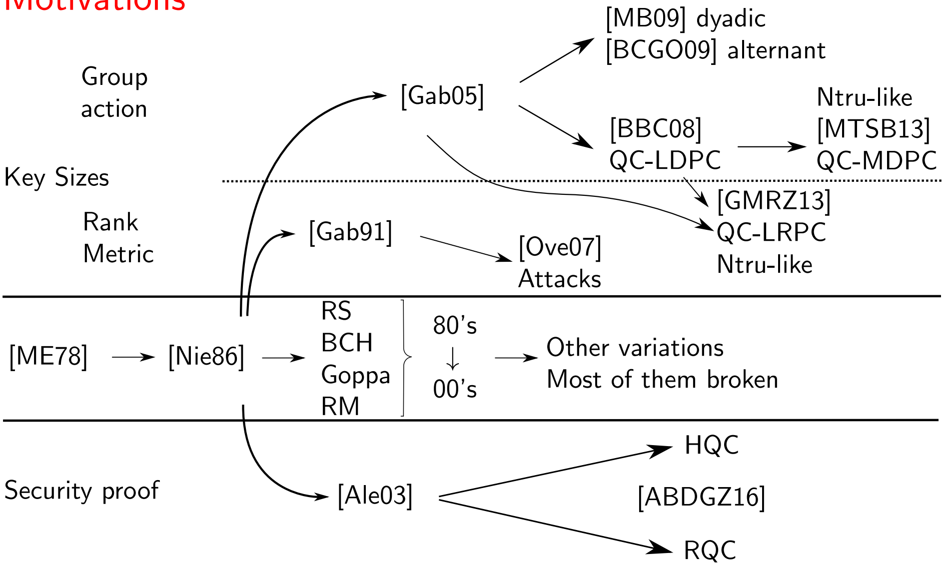
Motivations



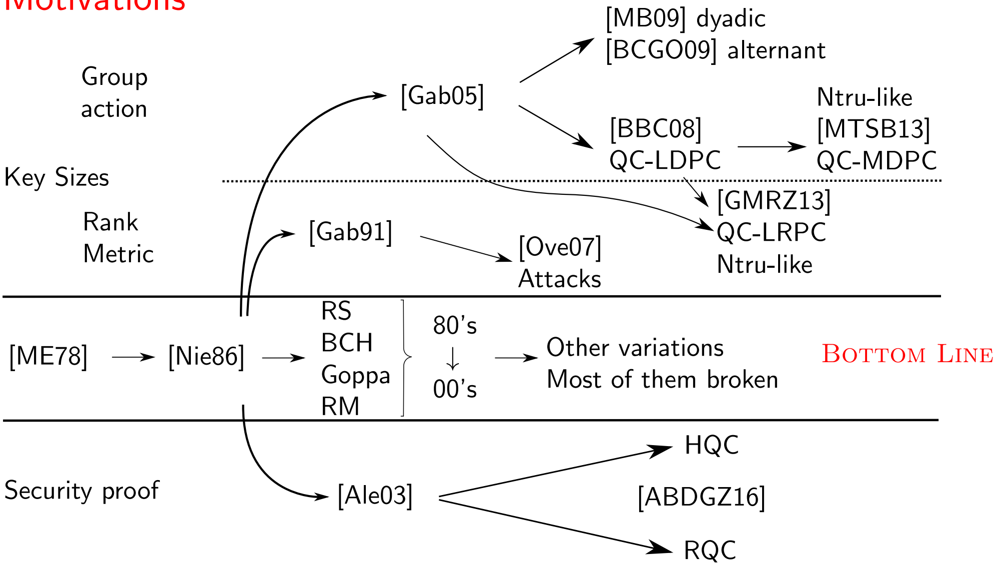
Motivations



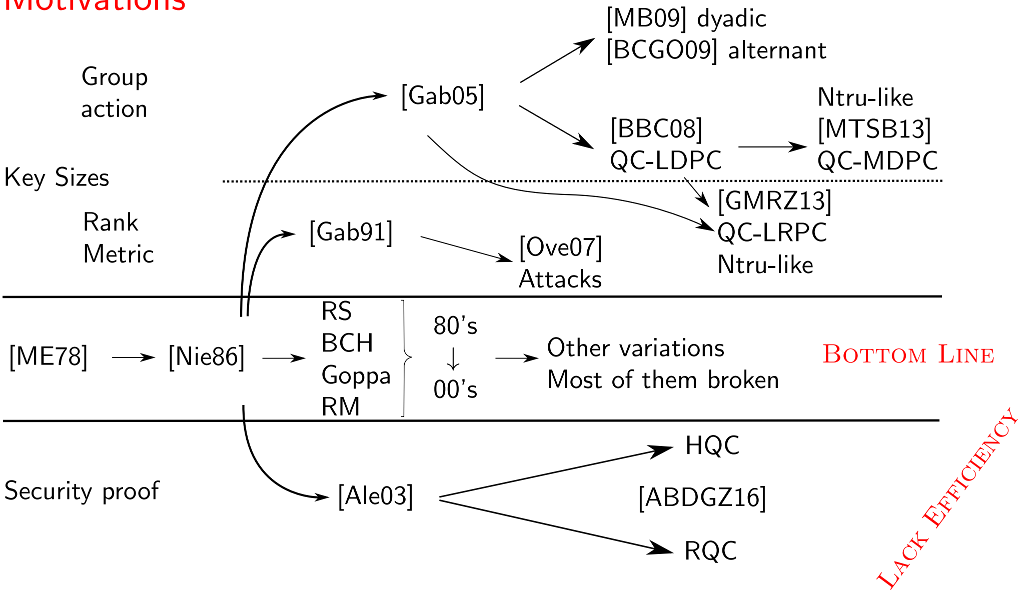
Motivations



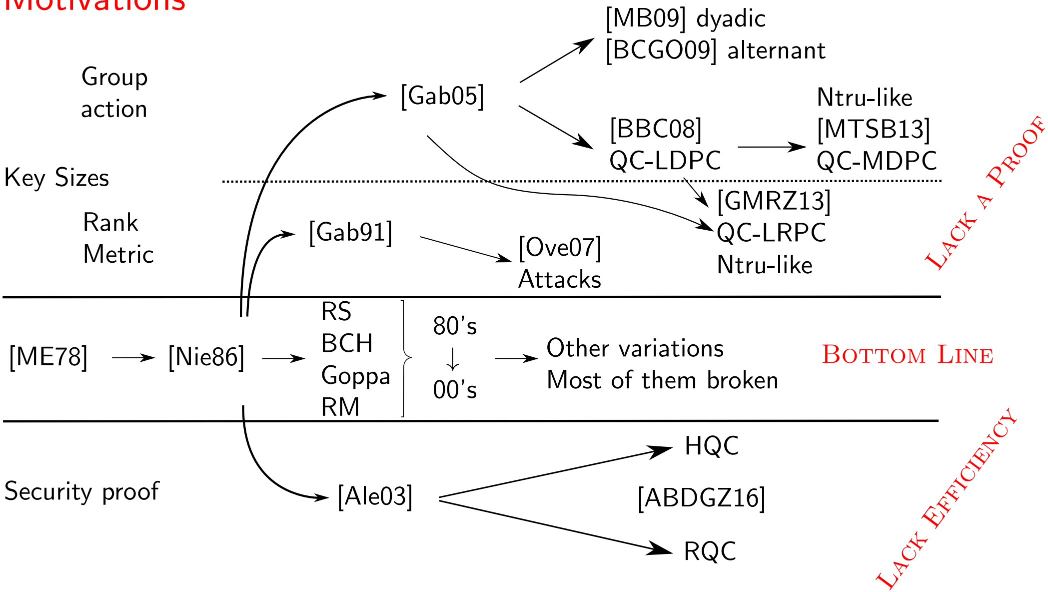
Motivations



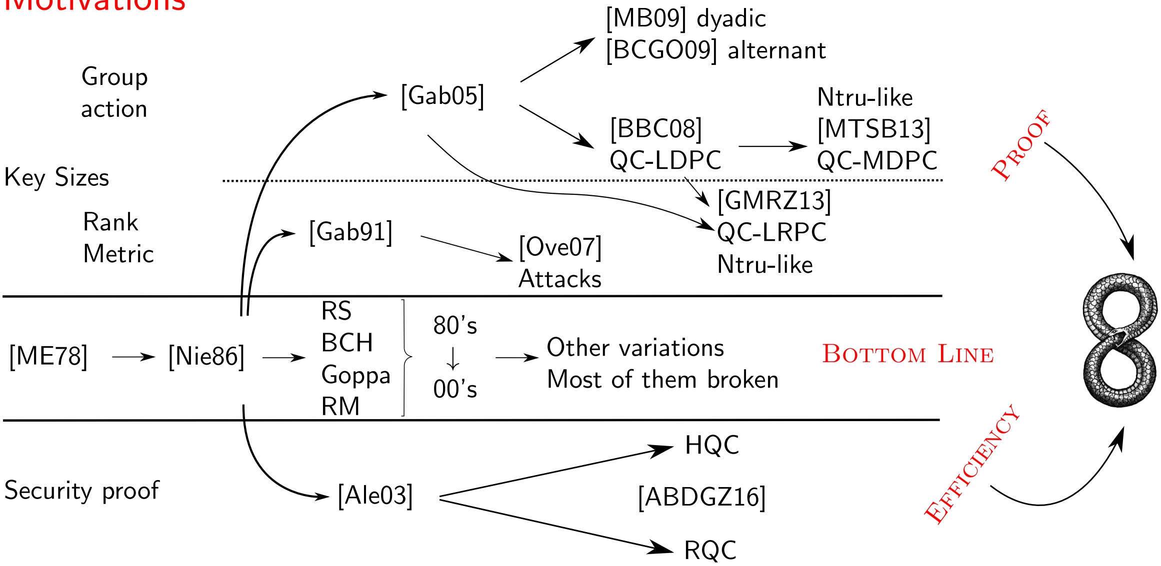
Motivations



Motivations



Motivations



Outline

1. Reminders on HQC
2. Presentation of the Ouroboros protocol
3. Security
4. Parameters

HQC Encryption Scheme [ABD+16]

Encryption scheme in **H**amming metric, using **Q**uasi-**C**yclic Codes

- Notation: Secret data - Public data - One-time Randomness
- **G** is the generator matrix of some public code $\mathcal{C}$.

Alice		Bob
$\mathsf{seed}_\mathbf{h} \xleftarrow{\$} \{0,1\}^\lambda$, $\mathbf{h} \xleftarrow{\mathsf{seed}_\mathbf{h}} \mathbb{F}_2^n$ $\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathcal{S}_w^n(\mathbb{F}_2)$, $\mathbf{s} \leftarrow \mathbf{x} + \mathbf{h}\mathbf{y}$	$\xrightarrow{\mathsf{seed}_\mathbf{h}, \mathbf{s}}$	
	$\xleftarrow{\mathbf{v}, \rho}$	$\mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{S}_w^n(\mathbb{F}_2)$, $\epsilon \xleftarrow{\$} \mathcal{S}_{cw}^n(\mathbb{F}_2)$ $\mathbf{v} \leftarrow \mathbf{r}_1 + \mathbf{h}\mathbf{r}_2$, $\rho \leftarrow \mu\mathbf{G} + \mathbf{s}\mathbf{r}_2 + \epsilon$
$\mu \leftarrow \mathcal{C}.\mathsf{Decode}(\rho - \mathbf{v}\mathbf{y})$		

HQC Encryption Scheme [ABD+16]

Encryption scheme in **H**amming metric, using **Q**uasi-**C**yclic Codes

- Notation: Secret data - Public data - One-time Randomness
- **G** is the generator matrix of some public code $\mathcal{C}$.

Alice		Bob
$\mathsf{seed_h} \xleftarrow{\$} \{0,1\}^\lambda$, $\mathbf{h} \xleftarrow{\mathsf{seed_h}} \mathbb{F}_2^n$ $\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathcal{S}_w^n(\mathbb{F}_2)$, $\mathbf{s} \leftarrow \mathbf{x} + \mathbf{hy}$	$\xrightarrow{\mathsf{seed_h}, \mathbf{s}}$	$\mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{S}_w^n(\mathbb{F}_2)$, $\epsilon \xleftarrow{\$} \mathcal{S}_{cw}^n(\mathbb{F}_2)$ $\mathbf{v} \leftarrow \mathbf{r}_1 + \mathbf{hr}_2$, $\rho \leftarrow \mu\mathbf{G} + \mathbf{sr}_2 + \epsilon$
$\mu \leftarrow \mathcal{C}.\mathrm{Decode}(\rho - \mathbf{vy})$	$\xleftarrow{\mathbf{v}, \rho}$	

HQC Encryption Scheme [ABD+16]

Encryption scheme in **H**amming metric, using **Q**uasi-**C**yclic Codes

- Notation: Secret data - Public data - One-time Randomness
- $\mathbf{G}$ is the generator matrix of some public code $\mathcal{C}$.

Alice		Bob
$\mathsf{seed}_{\mathbf{h}} \xleftarrow{\$} \{0,1\}^{\lambda}$, $\mathbf{h} \xleftarrow{\mathsf{seed}_{\mathbf{h}}} \mathbb{F}_2^n$ $\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathcal{S}_w^n(\mathbb{F}_2)$, $\mathbf{s} \leftarrow \mathbf{x} + \mathbf{h}\mathbf{y}$	$\xrightarrow{\mathsf{seed}_{\mathbf{h}}, \mathbf{s}}$	
	$\xleftarrow{\mathbf{v}, \boldsymbol{\rho}}$	$\mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{S}_w^n(\mathbb{F}_2)$, $\epsilon \xleftarrow{\$} \mathcal{S}_{cw}^n(\mathbb{F}_2)$ $\mathbf{v} \leftarrow \mathbf{r}_1 + \mathbf{h}\mathbf{r}_2$, $\boldsymbol{\rho} \leftarrow \boldsymbol{\mu}\mathbf{G} + \mathbf{s}\mathbf{r}_2 + \epsilon$
$\mu \leftarrow \mathcal{C}.\mathsf{Decode}(\boldsymbol{\rho} - \mathbf{v}\mathbf{y})$		

HQC Encryption Scheme [ABD+16]

Encryption scheme in **H**amming metric, using **Q**uasi-**C**yclic Codes

- Notation: Secret data - Public data - One-time Randomness
- **G** is the generator matrix of some public code $\mathcal{C}$.

Alice		Bob
$\text{seed}_{\mathbf{h}} \xleftarrow{\$} \{0,1\}^{\lambda}$, $\mathbf{h} \xleftarrow{\text{seed}_{\mathbf{h}}} \mathbb{F}_2^n$ $\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathcal{S}_w^n(\mathbb{F}_2)$, $\mathbf{s} \leftarrow \mathbf{x} + \mathbf{h}\mathbf{y}$	$\xrightarrow{\text{seed}_{\mathbf{h}}, \mathbf{s}}$	
	$\xleftarrow{\mathbf{v}, \boldsymbol{\rho}}$	$\mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{S}_w^n(\mathbb{F}_2)$, $\boldsymbol{\epsilon} \xleftarrow{\$} \mathcal{S}_{cw}^n(\mathbb{F}_2)$ $\mathbf{v} \leftarrow \mathbf{r}_1 + \mathbf{h}\mathbf{r}_2$, $\boldsymbol{\rho} \leftarrow \boldsymbol{\mu}\mathbf{G} + \mathbf{s}\mathbf{r}_2 + \boldsymbol{\epsilon}$
$\boldsymbol{\mu} \leftarrow \mathcal{C}.\text{Decode}(\boldsymbol{\rho} - \mathbf{v}\mathbf{y})$		

Correctness

Correctness Property

$$\text{Decrypt}(\text{sk}, \text{Encrypt}(\text{pk}, \boldsymbol{\mu}, \theta)) = \boldsymbol{\mu}$$

$\mathcal{C}$.Decode correctly decodes $\rho - \mathbf{v} \cdot \mathbf{y}$ whenever

the error term is **not too big**

$$\omega(\mathbf{s} \cdot \mathbf{r}_2 - \mathbf{v} \cdot \mathbf{y} + \epsilon) \leq \delta$$

$$\omega((\mathbf{x} + \mathbf{h} \cdot \mathbf{y}) \cdot \mathbf{r}_2 - (\mathbf{r}_1 + \mathbf{h} \cdot \mathbf{r}_2) \cdot \mathbf{y} + \epsilon) \leq \delta$$

$$\omega(\mathbf{x} \cdot \mathbf{r}_2 - \mathbf{r}_1 \cdot \mathbf{y} + \epsilon) \leq \delta$$

Error distribution analysis $\rightarrow$ Decryption failure probability better understood

Correctness

Correctness Property

$$\text{Decrypt}\left(\text{sk}, \text{Encrypt}\left(\text{pk}, \boldsymbol{\mu}, \theta\right)\right) = \boldsymbol{\mu}$$

$\mathcal{C}$.Decode correctly decodes $\boldsymbol{\rho} - \mathbf{v} \cdot \mathbf{y}$ whenever

the error term is **not too big**

$$\omega\left(\mathbf{s} \cdot \mathbf{r}_2 - \mathbf{v} \cdot \mathbf{y} + \boldsymbol{\epsilon}\right) \leq \delta$$

$$\omega\left(\left(\mathbf{x} + \mathbf{h} \cdot \mathbf{y}\right) \cdot \mathbf{r}_2 - \left(\mathbf{r}_1 + \mathbf{h} \cdot \mathbf{r}_2\right) \cdot \mathbf{y} + \boldsymbol{\epsilon}\right) \leq \delta$$

$$\omega(\mathbf{x} \cdot \mathbf{r}_2 - \mathbf{r}_1 \cdot \mathbf{y} + \boldsymbol{\epsilon}) \leq \delta$$

Error distribution analysis → Decryption failure probability better understood

Correctness

Correctness Property

$$\text{Decrypt}\left(\text{sk}, \text{Encrypt}\left(\text{pk}, \boldsymbol{\mu}, \theta\right)\right) = \boldsymbol{\mu}$$

$\mathcal{C}$.Decode correctly decodes $\boldsymbol{\rho} - \mathbf{v} \cdot \mathbf{y}$ whenever

the error term is **not too big**

$$\omega\left(\mathbf{s} \cdot \mathbf{r}_2 - \mathbf{v} \cdot \mathbf{y} + \boldsymbol{\epsilon}\right) \leq \delta$$

$$\omega\left(\left(\mathbf{x} + \mathbf{h} \cdot \mathbf{y}\right) \cdot \mathbf{r}_2 - \left(\mathbf{r}_1 + \mathbf{h} \cdot \mathbf{r}_2\right) \cdot \mathbf{y} + \boldsymbol{\epsilon}\right) \leq \delta$$

$$\omega(\mathbf{x} \cdot \mathbf{r}_2 - \mathbf{r}_1 \cdot \mathbf{y} + \boldsymbol{\epsilon}) \leq \delta$$

Error distribution analysis $\rightarrow$ Decryption failure probability better understood

Outline

1 Reminders on HQC

2 Presentation of the Ouroboros protocol
- Cyclic Error Decoding
- BitFlipping algorithm
- Description of the protocol

3 Security

4 Parameters

A particular decoding

- HQC requires $\mathbf{x} \cdot \mathbf{r}_2 - \mathbf{r}_1 \cdot \mathbf{y} + \epsilon$ to be "small" to correctly decode
- Ouroboros further exploits the shape of the error

Cyclic Error Decoding (CED) Problem

- Let $\mathbf{x}, \mathbf{y}, \mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{S}_w^n(\mathbb{F}_2)$ with $w = \mathcal{O}(\sqrt{n})$, and $\mathbf{e} \xleftarrow{\$} \mathcal{S}_{cw}^n(\mathbb{F}_2)$ a random error vector.
- Given $(\mathbf{x}, \mathbf{y}) \in (\mathcal{S}_w^n(\mathbb{F}_2))^2$ and $\mathbf{e}_c \leftarrow \mathbf{x}\mathbf{r}_2 - \mathbf{y}\mathbf{r}_1 + \mathbf{e}$ such that $\omega(\mathbf{r}_1) = \omega(\mathbf{r}_2) = w$, find $(\mathbf{r}_1, \mathbf{r}_2)$.

- This is essentially a *noisy* SD problem

A particular decoding

- HQC requires $\mathbf{x} \cdot \mathbf{r}_2 - \mathbf{r}_1 \cdot \mathbf{y} + \boldsymbol{\epsilon}$ to be "small" to correctly decode
- Ouroboros further exploits the shape of the error

Cyclic Error Decoding (CED) Problem

- Let $\mathbf{x}, \mathbf{y}, \mathbf{r}_1, \mathbf{r}_2 \stackrel{\$}{\leftarrow} \mathcal{S}_w^n(\mathbb{F}_2)$ with $w = \mathcal{O}(\sqrt{n})$, and $\mathbf{e} \stackrel{\$}{\leftarrow} \mathcal{S}_{cw}^n(\mathbb{F}_2)$ a random error vector.
- Given $(\mathbf{x}, \mathbf{y}) \in (\mathcal{S}_w^n(\mathbb{F}_2))^2$ and $\mathbf{e}_c \leftarrow \mathbf{x}\mathbf{r}_2 - \mathbf{y}\mathbf{r}_1 + \mathbf{e}$ such that $\omega(\mathbf{r}_1) = \omega(\mathbf{r}_2) = w$, find $(\mathbf{r}_1, \mathbf{r}_2)$.

- This is essentially a *noisy* SD problem

A particular decoding

- HQC requires $\mathbf{x} \cdot \mathbf{r}_2 - \mathbf{r}_1 \cdot \mathbf{y} + \epsilon$ to be "small" to correctly decode
- Ouroboros further exploits the shape of the error

Cyclic Error Decoding (CED) Problem

- Let $\mathbf{x}, \mathbf{y}, \mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{S}_w^n(\mathbb{F}_2)$ with $w = \mathcal{O}(\sqrt{n})$, and $\mathbf{e} \xleftarrow{\$} \mathcal{S}_{cw}^n(\mathbb{F}_2)$ a random error vector.
- Given $(\mathbf{x}, \mathbf{y}) \in (\mathcal{S}_w^n(\mathbb{F}_2))^2$ and $\mathbf{e}_c \leftarrow \mathbf{x}\mathbf{r}_2 - \mathbf{y}\mathbf{r}_1 + \mathbf{e}$ such that $\omega(\mathbf{r}_1) = \omega(\mathbf{r}_2) = w$, find $(\mathbf{r}_1, \mathbf{r}_2)$.

- This is essentially a *noisy* SD problem

A particular decoding

- HQC requires $\mathbf{x} \cdot \mathbf{r}_2 - \mathbf{r}_1 \cdot \mathbf{y} + \epsilon$ to be "small" to correctly decode
- Ouroboros further exploits the shape of the error

Cyclic Error Decoding (CED) Problem

- Let $\mathbf{x}, \mathbf{y}, \mathbf{r}_1, \mathbf{r}_2 \stackrel{\$}{\leftarrow} \mathcal{S}_w^n(\mathbb{F}_2)$ with $w = \mathcal{O}(\sqrt{n})$, and $\mathbf{e} \stackrel{\$}{\leftarrow} \mathcal{S}_{cw}^n(\mathbb{F}_2)$ a random error vector.
- Given $(\mathbf{x}, \mathbf{y}) \in (\mathcal{S}_w^n(\mathbb{F}_2))^2$ and $\mathbf{e_c} \leftarrow \mathbf{x}\mathbf{r}_2 - \mathbf{y}\mathbf{r}_1 + \mathbf{e}$ such that $\omega(\mathbf{r}_1) = \omega(\mathbf{r}_2) = w$, find $(\mathbf{r}_1, \mathbf{r}_2)$.

- This is essentially a *noisy* SD problem

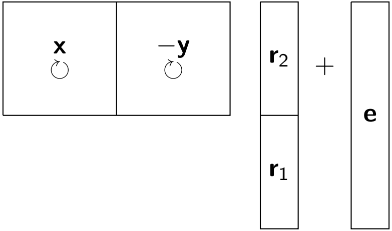
A particular decoding

- HQC requires $\mathbf{x} \cdot \mathbf{r}_2 - \mathbf{r}_1 \cdot \mathbf{y} + \epsilon$ to be "small" to correctly decode
- Ouroboros further exploits the shape of the error

Cyclic Error Decoding (CED) Problem

- Let $\mathbf{x}, \mathbf{y}, \mathbf{r}_1, \mathbf{r}_2 \stackrel{\$}{\leftarrow} \mathcal{S}_w^n(\mathbb{F}_2)$ with $w = \mathcal{O}(\sqrt{n})$, and $\mathbf{e} \stackrel{\$}{\leftarrow} \mathcal{S}_{cw}^n(\mathbb{F}_2)$ a random error vector.
- Given $(\mathbf{x}, \mathbf{y}) \in (\mathcal{S}_w^n(\mathbb{F}_2))^2$ and $\mathbf{e_c} \leftarrow \mathbf{x}\mathbf{r}_2 - \mathbf{y}\mathbf{r}_1 + \mathbf{e}$ such that $\omega(\mathbf{r}_1) = \omega(\mathbf{r}_2) = w$, find $(\mathbf{r}_1, \mathbf{r}_2)$.

- This is essentially a *noisy* SD problem



Outline

1. Reminders on HQC
2. Presentation of the Ouroboros protocol
   - Cyclic Error Decoding
   - BitFlipping algorithm
   - Description of the protocol
3. Security
4. Parameters

Hard Decision Decoding: BitFlipping

- Introduced by Gallager in 1962
- Iterative decoding for **L**ow **D**ensity **P**arity **C**heck codes
- Decoding capacity increase linearly with the code length

Intuition

1. Compute the number of unsatisfied parity-check equations for each bit of the message
2. If this number is greater than some *threshold*, flip the bit and go to 1.
3. Stop when the syndrome is null (or after a certain number of iterations).

- Easy to understand
- Easy to implement
- Pretty efficient
- The threshold value is crucial [CS16]

Hard Decision Decoding: BitFlipping

- Introduced by Gallager in 1962
- Iterative decoding for **L**ow **D**ensity **P**arity **C**heck codes
- Decoding capacity increase linearly with the code length

Intuition

1. Compute the number of unsatisfied parity-check equations for each bit of the message
2. If this number is greater than some *threshold*, flip the bit and go to 1.
3. Stop when the syndrome is null (or after a certain number of iterations).

- Easy to understand
- Easy to implement
- Pretty efficient
- The threshold value is crucial [CS16]

Hard Decision Decoding: BitFlipping

- Introduced by Gallager in 1962
- Iterative decoding for **L**ow **D**ensity **P**arity **C**heck codes
- Decoding capacity increase linearly with the code length

Intuition

1. Compute the number of unsatisfied parity-check equations for each bit of the message
2. If this number is greater than some *threshold*, flip the bit and go to 1.
3. Stop when the syndrome is null (or after a certain number of iterations).

- Easy to understand
- Easy to implement
- Pretty efficient
- The threshold value is crucial [CS16]

Hard Decision Decoding: BitFlipping

- Introduced by Gallager in 1962
- Iterative decoding for **L**ow **D**ensity **P**arity **C**heck codes
- Decoding capacity increase linearly with the code length

Intuition

1. Compute the number of unsatisfied parity-check equations for each bit of the message
2. If this number is greater than some *threshold*, flip the bit and go to 1.
3. Stop when the syndrome is null (or after a certain number of iterations).

- Easy to understand
- Easy to implement
- Pretty efficient
- The threshold value is crucial [CS16]

Hard Decision Decoding: BitFlipping

- Introduced by Gallager in 1962
- Iterative decoding for **L**ow **D**ensity **P**arity **C**heck codes
- Decoding capacity increase linearly with the code length

Intuition

1. Compute the number of unsatisfied parity-check equations for each bit of the message
2. If this number is greater than some *threshold*, flip the bit and go to 1.
3. Stop when the syndrome is null (or after a certain number of iterations).

- Easy to understand
- Easy to implement
- Pretty efficient
- The threshold value is crucial [CS16]

Outline

1. Reminders on HQC
2. Presentation of the Ouroboros protocol
   - Cyclic Error Decoding
   - BitFlipping algorithm
   - Description of the protocol
3. Security
4. Parameters

Ouroboros

- Requires a hash function Hash : $\{0,1\}^* \longrightarrow \mathcal{S}^n_{cw}(\mathbb{F}_2)$ [Sen05]
- ϵ of HQC plays the role of the exchanged secret in Ouroboros
- CE-Decoder is a modified BitFlipping algorithm to solve the CED problem

Alice		Bob
$\mathsf{seed}_{\mathbf{h}} \xleftarrow{\$} \{0,1\}^\lambda$, $\mathbf{h} \xleftarrow{\mathsf{seed}_{\mathbf{h}}} \mathbb{F}_2^n$		
$\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathcal{S}^n_w(\mathbb{F}_2)$, $\mathbf{s} \leftarrow \mathbf{x} + \mathbf{h}\mathbf{y}$	$\xrightarrow{\mathbf{h},\mathbf{s}}$	$\mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{S}^n_w(\mathbb{F}_2)$
		$\mathbf{e}_r \leftarrow \mathrm{Hash}(\mathbf{r}_1, \mathbf{r}_2)$, $\epsilon \xleftarrow{\$} \mathcal{S}^n_{w_e}(\mathbb{F}_2)$
	$\xleftarrow{\mathbf{s}_r, \mathbf{s}_e}$	$\mathbf{s}_r \leftarrow \mathbf{r}_1 + \mathbf{h}\mathbf{r}_2$, $\mathbf{s}_e \leftarrow \mathbf{s}\mathbf{r}_2 + \mathbf{e}_r + \epsilon$
$\mathbf{e}_c \leftarrow \mathbf{s}_e - \mathbf{y}\mathbf{s}_r = \mathbf{x}\mathbf{r}_2 - \mathbf{y}\mathbf{r}_1 + \epsilon'$		
$(\mathbf{r}_1, \mathbf{r}_2) \leftarrow \text{CE-Decoder}(\mathbf{x}, \mathbf{y}, \mathbf{e}_c, t, w, w_e)$		
$\epsilon \leftarrow \mathbf{e}_c - \mathbf{x}\mathbf{r}_2 + \mathbf{y}\mathbf{r}_1 - \mathrm{Hash}(\mathbf{r}_1, \mathbf{r}_2)$		
ϵ	SHARED SECRET	ϵ

Ouroboros

- Requires a hash function Hash : $\{0,1\}^* \longrightarrow \mathcal{S}^n_{cw}(\mathbb{F}_2)$ [Sen05]
- ϵ of HQC plays the role of the exchanged secret in Ouroboros
- CE-Decoder is a modified BitFlipping algorithm to solve the CED problem

Ouroboros

- Requires a hash function Hash : $\{0,1\}^* \longrightarrow \mathcal{S}^n_{cw}(\mathbb{F}_2)$ [Sen05]
- ϵ of HQC plays the role of the exchanged secret in Ouroboros
- CE-Decoder is a modified BitFlipping algorithm to solve the CED problem

Alice		Bob
$\mathsf{seed_h} \xleftarrow{\$} \{0,1\}^\lambda$, $\mathbf{h} \xleftarrow{\mathsf{seed_h}} \mathbb{F}_2^n$		
$\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathcal{S}^n_w(\mathbb{F}_2)$, $\mathbf{s} \leftarrow \mathbf{x} + \mathbf{hy}$	$\xrightarrow{\mathbf{h},\mathbf{s}}$	$\mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{S}^n_w(\mathbb{F}_2)$
		$\mathbf{e_r} \leftarrow \mathrm{Hash}(\mathbf{r}_1, \mathbf{r}_2)$, $\epsilon \xleftarrow{\$} \mathcal{S}^n_{w_e}(\mathbb{F}_2)$
	$\xleftarrow{\mathbf{s_r},\mathbf{s_e}}$	$\mathbf{s_r} \leftarrow \mathbf{r}_1 + \mathbf{hr}_2$, $\mathbf{s_e} \leftarrow \mathbf{sr}_2 + \mathbf{e_r} + \epsilon$
$\mathbf{e_c} \leftarrow \mathbf{s_e} - \mathbf{ys_r} = \mathbf{xr}_2 - \mathbf{yr}_1 + \epsilon'$		
$(\mathbf{r}_1, \mathbf{r}_2) \leftarrow \text{CE-Decoder}(\mathbf{x}, \mathbf{y}, \mathbf{e_c}, t, w, w_e)$		
$\epsilon \leftarrow \mathbf{e_c} - \mathbf{xr}_2 + \mathbf{yr}_1 - \mathrm{Hash}(\mathbf{r}_1, \mathbf{r}_2)$		
$\boxed{\epsilon}$	SHARED SECRET	$\boxed{\epsilon}$

Ouroboros

- Requires a hash function Hash : $\{0,1\}^* \longrightarrow \mathcal{S}^n_{cw}(\mathbb{F}_2)$ [Sen05]
- ϵ of HQC plays the role of the exchanged secret in Ouroboros
- CE-Decoder is a modified BitFlipping algorithm to solve the CED problem

Alice		Bob
$\mathsf{seed_h} \xleftarrow{\$} \{0,1\}^\lambda$, $\mathbf{h} \xleftarrow{\mathsf{seed_h}} \mathbb{F}_2^n$ $\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathcal{S}^n_w(\mathbb{F}_2)$, $\mathbf{s} \leftarrow \mathbf{x} + \mathbf{h}\mathbf{y}$	$\xrightarrow{\mathbf{h},\mathbf{s}}$	$\mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{S}^n_w(\mathbb{F}_2)$ $\mathbf{e_r} \leftarrow \mathrm{Hash}(\mathbf{r}_1, \mathbf{r}_2)$, $\epsilon \xleftarrow{\$} \mathcal{S}^n_{w_e}(\mathbb{F}_2)$ $\mathbf{s_r} \leftarrow \mathbf{r}_1 + \mathbf{h}\mathbf{r}_2$, $\mathbf{s_e} \leftarrow \mathbf{s}\mathbf{r}_2 + \mathbf{e_r} + \epsilon$
$\mathbf{e_c} \leftarrow \mathbf{s_e} - \mathbf{y}\mathbf{s_r} = \mathbf{x}\mathbf{r}_2 - \mathbf{y}\mathbf{r}_1 + \epsilon'$ $(\mathbf{r}_1, \mathbf{r}_2) \leftarrow \text{CE-Decoder}(\mathbf{x}, \mathbf{y}, \mathbf{e_c}, t, w, w_e)$ $\epsilon \leftarrow \mathbf{e_c} - \mathbf{x}\mathbf{r}_2 + \mathbf{y}\mathbf{r}_1 - \mathrm{Hash}(\mathbf{r}_1, \mathbf{r}_2)$	$\xleftarrow{\mathbf{s_r},\mathbf{s_e}}$	
$\boxed{\epsilon}$	SHARED SECRET	$\boxed{\epsilon}$

Ouroboros

- Requires a hash function Hash : $\{0,1\}^* \longrightarrow \mathcal{S}^n_{cw}(\mathbb{F}_2)$ [Sen05]
- ϵ of HQC plays the role of the exchanged secret in Ouroboros
- CE-Decoder is a modified BitFlipping algorithm to solve the CED problem

Alice

$\mathsf{seed}_\mathbf{h} \xleftarrow{\$} \{0,1\}^\lambda$, $\mathbf{h} \xleftarrow{\mathsf{seed}_\mathbf{h}} \mathbb{F}_2^n$

$\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathcal{S}^n_w(\mathbb{F}_2)$, $\mathbf{s} \leftarrow \mathbf{x} + \mathbf{h}\mathbf{y}$

$\xrightarrow{\mathbf{h}, \mathbf{s}}$

Bob

$\mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{S}^n_w(\mathbb{F}_2)$

$\mathbf{e}_r \leftarrow \mathrm{Hash}(\mathbf{r}_1, \mathbf{r}_2)$, $\epsilon \xleftarrow{\$} \mathcal{S}^n_{w_e}(\mathbb{F}_2)$

$\mathbf{s}_r \leftarrow \mathbf{r}_1 + \mathbf{h}\mathbf{r}_2$, $\mathbf{s}_e \leftarrow \mathbf{s}\mathbf{r}_2 + \mathbf{e}_r + \epsilon$

$\xleftarrow{\mathbf{s}_r, \mathbf{s}_e}$

Alice

$\mathbf{e}_c \leftarrow \mathbf{s}_e - \mathbf{y}\mathbf{s}_r = \mathbf{x}\mathbf{r}_2 - \mathbf{y}\mathbf{r}_1 + \epsilon'$

$(\mathbf{r}_1, \mathbf{r}_2) \leftarrow \text{CE-Decoder}(\mathbf{x}, \mathbf{y}, \mathbf{e}_c, t, w, w_e)$

$\epsilon \leftarrow \mathbf{e}_c - \mathbf{x}\mathbf{r}_2 + \mathbf{y}\mathbf{r}_1 - \mathrm{Hash}(\mathbf{r}_1, \mathbf{r}_2)$

$\boxed{\epsilon}$ SHARED SECRET $\boxed{\epsilon}$

Ouroboros

- Requires a hash function Hash : $\{0,1\}^* \longrightarrow \mathcal{S}^n_{cw}(\mathbb{F}_2)$ [Sen05]
- ϵ of HQC plays the role of the exchanged secret in Ouroboros
- CE-Decoder is a modified BitFlipping algorithm to solve the CED problem

Alice		Bob
$\mathbf{seed_h} \xleftarrow{\$} \{0,1\}^\lambda$, $\mathbf{h} \xleftarrow{\mathbf{seed_h}} \mathbb{F}_2^n$		
$\mathbf{x},\mathbf{y} \xleftarrow{\$} \mathcal{S}^n_w(\mathbb{F}_2)$, $\mathbf{s} \leftarrow \mathbf{x} + \mathbf{hy}$	$\xrightarrow{\mathbf{h},\mathbf{s}}$	$\mathbf{r}_1,\mathbf{r}_2 \xleftarrow{\$} \mathcal{S}^n_w(\mathbb{F}_2)$
		$\mathbf{e_r} \leftarrow \mathrm{Hash}(\mathbf{r}_1,\mathbf{r}_2)$, $\epsilon \xleftarrow{\$} \mathcal{S}^n_{w_e}(\mathbb{F}_2)$
$\mathbf{e_c} \leftarrow \mathbf{s_e} - \mathbf{ys_r} = \mathbf{xr}_2 - \mathbf{yr}_1 + \epsilon'$	$\xleftarrow{\mathbf{s_r},\mathbf{s_e}}$	$\mathbf{s_r} \leftarrow \mathbf{r}_1 + \mathbf{hr}_2$, $\mathbf{s_e} \leftarrow \mathbf{sr}_2 + \mathbf{e_r} + \epsilon$
$(\mathbf{r}_1,\mathbf{r}_2) \leftarrow \text{CE-Decoder}(\mathbf{x},\mathbf{y},\mathbf{e_c},t,w,w_e)$		
$\epsilon \leftarrow \mathbf{e_c} - \mathbf{xr}_2 + \mathbf{yr}_1 - \mathrm{Hash}(\mathbf{r}_1,\mathbf{r}_2)$		
$\boxed{\epsilon}$	SHARED SECRET	$\boxed{\epsilon}$

Ouroboros

- Requires a hash function Hash : $\{0,1\}^* \longrightarrow \mathcal{S}^n_{cw}(\mathbb{F}_2)$ [Sen05]
- ϵ of HQC plays the role of the exchanged secret in Ouroboros
- CE-Decoder is a modified BitFlipping algorithm to solve the CED problem

Alice		Bob
$\mathsf{seed_h} \xleftarrow{\$} \{0,1\}^\lambda$, $\mathbf{h} \xleftarrow{\mathsf{seed_h}} \mathbb{F}_2^n$		
$\mathbf{x}, \mathbf{y} \xleftarrow{\$} \mathcal{S}^n_w(\mathbb{F}_2)$, $\mathbf{s} \leftarrow \mathbf{x} + \mathbf{hy}$	$\xrightarrow{\mathbf{h},\mathbf{s}}$	$\mathbf{r}_1, \mathbf{r}_2 \xleftarrow{\$} \mathcal{S}^n_w(\mathbb{F}_2)$
		$\mathbf{e_r} \leftarrow \text{Hash}(\mathbf{r}_1, \mathbf{r}_2)$, $\epsilon \xleftarrow{\$} \mathcal{S}^n_{w_e}(\mathbb{F}_2)$
$\mathbf{e_c} \leftarrow \mathbf{s_e} - \mathbf{ys_r} = \mathbf{xr}_2 - \mathbf{yr}_1 + \epsilon'$	$\xleftarrow{\mathbf{s_r},\mathbf{s_e}}$	$\mathbf{s_r} \leftarrow \mathbf{r}_1 + \mathbf{hr}_2$, $\mathbf{s_e} \leftarrow \mathbf{sr}_2 + \mathbf{e_r} + \epsilon$
$(\mathbf{r}_1, \mathbf{r}_2) \leftarrow \text{CE-Decoder}(\mathbf{x}, \mathbf{y}, \mathbf{e_c}, t, w, w_e)$		
$\epsilon \leftarrow \mathbf{e_c} - \mathbf{xr}_2 + \mathbf{yr}_1 - \text{Hash}(\mathbf{r}_1, \mathbf{r}_2)$		
$\boxed{\epsilon}$	SHARED SECRET	$\boxed{\epsilon}$

Outline

1. Reminders on HQC
2. Presentation of the Ouroboros protocol
3. Security
   - Security Model and Hybrid Argument
   - Ouroboros Security
4. Parameters

Security Model and Hybrid Argument

- Key exchange as an encryption scheme
- Same as Ding *et al.* [Din12, DXL12], Peikert's [Pei14], BCNS [BCNS15] and NewHope [ADPS16]
- Usual game:

Exp$_{\mathcal{E},\mathcal{A}}^{\text{ind}-b}(\lambda)$
1. param ← Setup(1^λ)
2. (pk, sk) ← KeyGen(param)
3. $(\epsilon_0, \epsilon_1) \leftarrow \mathcal{A}(\texttt{FIND} : \text{pk})$
4. $\mathbf{c}^* \leftarrow \text{Encrypt}(\text{pk}, \epsilon_b, \theta)$
5. $b' \leftarrow \mathcal{A}(\texttt{GUESS} : \mathbf{c}^*)$
6. RETURN b'

- Hybrid argument:
  1. Construct a sequence of games transitioning from Enc(ϵ_0) to Enc(ϵ_1)
  2. Prove they are indistinguishable one from another

Security Model and Hybrid Argument

- Key exchange as an encryption scheme
- Same as Ding *et al.* [Din12, DXL12], Peikert's [Pei14], BCNS [BCNS15] and NEWHOPE [ADPS16]

- Usual game:

Exp$_{\mathcal{E},\mathcal{A}}^{\text{ind}-b}(\lambda)$
1. param $\leftarrow$ Setup(1^λ)
2. (pk, sk) $\leftarrow$ KeyGen(param)
3. $(\epsilon_0, \epsilon_1) \leftarrow \mathcal{A}(\texttt{FIND} : \text{pk})$
4. $\mathbf{c}^* \leftarrow$ Encrypt$(\text{pk}, \epsilon_b, \theta)$
5. $b' \leftarrow \mathcal{A}(\texttt{GUESS} : \mathbf{c}^*)$
6. RETURN b'

- Hybrid argument:
  1. Construct a sequence of games transitioning from Enc(ϵ_0) to Enc(ϵ_1)
  2. Prove they are indistinguishable one from another

Security Model and Hybrid Argument

- Key exchange as an encryption scheme
- Same as Ding *et al.* [Din12, DXL12], Peikert's [Pei14], BCNS [BCNS15] and NewHope [ADPS16]
- Usual game:

Exp$_{\mathcal{E},\mathcal{A}}^{\text{ind}-b}(\lambda)$
1. param $\leftarrow$ Setup(1^λ)
2. (pk, sk) $\leftarrow$ KeyGen(param)
3. $(\epsilon_0, \epsilon_1) \leftarrow \mathcal{A}(\texttt{FIND} : \text{pk})$
4. $\mathbf{c}^* \leftarrow$ Encrypt$(\text{pk}, \epsilon_b, \theta)$
5. $b' \leftarrow \mathcal{A}(\texttt{GUESS} : \mathbf{c}^*)$
6. RETURN b'

- Hybrid argument:
  1. Construct a sequence of games transitioning from Enc(ϵ_0) to Enc(ϵ_1)
  2. Prove they are indistinguishable one from another

Security Model and Hybrid Argument

- Key exchange as an encryption scheme
- Same as Ding *et al.* [Din12, DXL12], Peikert's [Pei14], BCNS [BCNS15] and NewHope [ADPS16]

- Usual game:

$\mathbf{Exp}_{\mathcal{E},\mathcal{A}}^{\mathsf{ind}-b}(\lambda)$
1. param $\leftarrow$ Setup(1^λ)
2. (pk, sk) $\leftarrow$ KeyGen(param)
3. $(\epsilon_0, \epsilon_1) \leftarrow \mathcal{A}(\texttt{FIND} : \text{pk})$
4. $\mathbf{c}^* \leftarrow$ Encrypt$(\text{pk}, \epsilon_b, \theta)$
5. $b' \leftarrow \mathcal{A}(\texttt{GUESS} : \mathbf{c}^*)$
6. RETURN b'

- Hybrid argument:
  1. Construct a sequence of games transitioning from Enc(ϵ_0) to Enc(ϵ_1)
  2. Prove they are indistinguishable one from another

Outline

1. Reminders on HQC
2. Presentation of the Ouroboros protocol
3. Security
   - Security Model and Hybrid Argument
   - Ouroboros Security
4. Parameters

Security

Definition (SD Distribution)

For positive integers, n, k, and w, the *SD(n, k, w) Distribution* chooses $\mathbf{H} \stackrel{\$}{\leftarrow} \mathbb{F}^{(n-k)\times n}$ and $\mathbf{x} \stackrel{\$}{\leftarrow} \mathbb{F}^n$ such that $\omega(\mathbf{x}) = w$, and outputs $(\mathbf{H}, \mathbf{H}\mathbf{x}^\top)$.

Definition (Decisional s-QCSD Problem)

For positive integers n, k, w, s, a random parity check matrix $\mathbf{H}$ of a QC code $\mathcal{C}$ and $\mathbf{y} \stackrel{\$}{\leftarrow} \mathbb{F}^n$, the *Decisional s-Quasi-Cyclic SD Problem s-DQCSD(n, k, w)* asks to decide with non-negligible advantage whether $(\mathbf{H}, \mathbf{y}^\top)$ came from the s-QCSD(n, k, w) distribution or the uniform distribution over $\mathbb{F}^{(n-k)\times n} \times \mathbb{F}^{n-k}$.

Theorem

Ouroboros is IND-CPA *under the 2-DQCSD and 3-DQCSD assumptions.* $\longrightarrow$ sketch of proof

Security

Definition (SD Distribution)

For positive integers, n, k, and w, the *SD*(n, k, w) *Distribution* chooses $\mathbf{H} \xleftarrow{\$} \mathbb{F}^{(n-k)\times n}$ and $\mathbf{x} \xleftarrow{\$} \mathbb{F}^n$ such that $\omega(\mathbf{x}) = w$, and outputs $(\mathbf{H}, \mathbf{H}\mathbf{x}^\top)$.

Definition (Decisional s-QCSD Problem)

For positive integers n, k, w, s, a random parity check matrix $\mathbf{H}$ of a QC code $\mathcal{C}$ and $\mathbf{y} \xleftarrow{\$} \mathbb{F}^n$, the *Decisional s-Quasi-Cyclic SD Problem s-DQCSD*(n, k, w) asks to decide with non-negligible advantage whether $(\mathbf{H}, \mathbf{y}^\top)$ came from the s-QCSD(n, k, w) distribution or the uniform distribution over $\mathbb{F}^{(n-k)\times n} \times \mathbb{F}^{n-k}$.

Theorem

Ouroboros is IND-CPA *under the 2-DQCSD and 3-DQCSD assumptions.* $\rightarrow$ sketch of proof

Security

Definition (SD Distribution)

For positive integers, n, k, and w, the *SD*(n, k, w) *Distribution* chooses $\mathbf{H} \xleftarrow{\$} \mathbb{F}^{(n-k)\times n}$ and $\mathbf{x} \xleftarrow{\$} \mathbb{F}^{n}$ such that $\omega(\mathbf{x}) = w$, and outputs $(\mathbf{H}, \mathbf{H}\mathbf{x}^{\top})$.

Definition (Decisional s-QCSD Problem)

For positive integers n, k, w, s, a random parity check matrix $\mathbf{H}$ of a QC code $\mathcal{C}$ and $\mathbf{y} \xleftarrow{\$} \mathbb{F}^{n}$, the *Decisional s-Quasi-Cyclic SD Problem s-DQCSD*(n, k, w) asks to decide with non-negligible advantage whether $(\mathbf{H}, \mathbf{y}^{\top})$ came from the s-QCSD(n, k, w) distribution or the uniform distribution over $\mathbb{F}^{(n-k)\times n} \times \mathbb{F}^{n-k}$.

Theorem

Ouroboros is IND-CPA *under the* 2*-DQCSD and* 3*-DQCSD assumptions.* $\rightarrow$ sketch of proof

Outline

1. Reminders on HQC
2. Presentation of the Ouroboros protocol
3. Security
4. Parameters
   - Reduction Compliant
   - Optimized Parameters

Reduction Compliant Parameters

	Ouroboros Parameters					
Instance	n	w	w_e	threshold	security	DFR
Low-I	$5,851$	47	94	30	80	$0.92 \cdot 10^{-5}$
Low-II	$5,923$	47	94	30	80	$2.3 \cdot 10^{-6}$
Medium-I	$13,691$	75	150	45	128	$0.96 \cdot 10^{-5}$
Medium-II	$14,243$	75	150	45	128	$1.09 \cdot 10^{-6}$
Strong-I	$40,013$	147	294	85	256	$4.20 \cdot 10^{-5}$
Strong-II	$40,973$	147	294	85	256	$< 10^{-6}$

Table : Parameter sets for Ouroboros

Outline

1 Reminders on HQC

2 Presentation of the Ouroboros protocol

3 Security

4 Parameters
- Reduction Compliant
- Optimized Parameters

Optimized Parameters wrt Best Know Attacks

	Ouroboros Optimized Parameters					
Instance	n	w	w_e	threshold	security	DFR
Low-I	$4,813$	41	123	27	80	$2.23 \cdot 10^{-5}$
Low-II	$5,003$	41	123	27	80	$2.60 \cdot 10^{-6}$
Medium-I	$10,301$	67	201	42	128	$1.01 \cdot 10^{-4}$
Medium-II	$10,837$	67	201	42	128	$< 10^{-7}$
Strong-I	$32,771$	131	393	77	256	$< 10^{-4}$
Strong-II	$33,997$	131	393	77	256	$< 10^{-7}$

Table : Optimized parameter sets for Ouroboros in Hamming metric

Conclusion

In this talk

- Ouroboros: a *secure*, *simple*, and *efficient* code-based key exchange protocol
- Efficient decoding through BitFlipping
- Competitive parameters

Further Improvements

Conclusion

In this talk

- Ouroboros: a *secure*, *simple*, and *efficient* code-based key exchange protocol
- Efficient decoding through BitFlipping
- Competitive parameters

Further Improvements

Conclusion

In this talk

- Ouroboros: a *secure*, *simple*, and *efficient* code-based key exchange protocol
- Efficient decoding through BitFlipping
- Competitive parameters

Further Improvements

Conclusion

In this talk

- Ouroboros: a *secure*, *simple*, and *efficient* code-based key exchange protocol
- Efficient decoding through BitFlipping
- Competitive parameters

Further Improvements

- Improve BitFlipping threshold [CS16]
- Switching to Rank metric drastically improves parameters! $\longrightarrow$ interlude?
- Optimize implementation
- OpenSSL TLS integration

Conclusion

In this talk

- Ouroboros: a *secure*, *simple*, and *efficient* code-based key exchange protocol
- Efficient decoding through BitFlipping
- Competitive parameters

Further Improvements

- Improve BitFlipping threshold [CS16]
- Switching to Rank metric drastically improves parameters! → interlude?
- Optimize implementation
- OpenSSL TLS integration

Conclusion

In this talk

- Ouroboros: a *secure*, *simple*, and *efficient* code-based key exchange protocol
- Efficient decoding through BitFlipping
- Competitive parameters

Further Improvements

- Improve BitFlipping threshold [CS16]
- Switching to Rank metric drastically improves parameters! → interlude?
- Optimize implementation
- OpenSSL TLS integration

Conclusion

In this talk

- Ouroboros: a *secure*, *simple*, and *efficient* code-based key exchange protocol
- Efficient decoding through BitFlipping
- Competitive parameters

Further Improvements

- Improve BitFlipping threshold [CS16]
- Switching to Rank metric drastically improves parameters! $\longrightarrow$ interlude?
- Optimize implementation
- OpenSSL TLS integration

Conclusion

In this talk

- Ouroboros: a *secure*, *simple*, and *efficient* code-based key exchange protocol
- Efficient decoding through BitFlipping
- Competitive parameters

Further Improvements

- Improve BitFlipping threshold [CS16]
- Switching to Rank metric drastically improves parameters! $\longrightarrow$ interlude?
- Optimize implementation
- OpenSSL TLS integration

Thanks!

Paper available @ http://unil.im/ouroboros

Thanks!

Carlos Aguilar Melchor, Olivier Blazy, Jean Christophe Deneuville, Philippe Gaborit, and Gilles Zémor.
Efficient encryption from random quasi-cyclic codes.
CoRR, abs/1612.05572, 2016.

Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe.
Post-quantum key exchange - A new hope.
In Thorsten Holz and Stefan Savage, editors, *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016.*, pages 327–343. USENIX Association, 2016.

Michael Alekhnovich.
More on average case vs approximation complexity.
In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pages 298–307, 2003.

Joppe W. Bos, Craig Costello, Michael Naehrig, and Douglas Stebila.
Post-quantum key exchange for the TLS protocol from the ring learning with errors problem.
In *2015 IEEE Symposium on Security and Privacy*, pages 553–570. IEEE Computer Society Press, May 2015.

Julia Chaulet and Nicolas Sendrier.
Worst case qc-mdpc decoder for mceliece cryptosystem.
In *Information Theory (ISIT), 2016 IEEE International Symposium on*, pages 1366–1370. IEEE, 2016.

Jintai Ding.
New cryptographic constructions using generalized learning with errors problem.
Cryptology ePrint Archive, Report 2012/387, 2012.

Jintai Ding, Xiang Xie, and Xiaodong Lin.
A simple provably secure key exchange scheme based on the learning with errors problem.
Cryptology ePrint Archive, Report 2012/688, 2012.

Rafael Misoczki, Jean-Pierre Tillich, Nicolas Sendrier, and Paulo SLM Barreto.
Mdpc-mceliece: New mceliece variants from moderate density parity-check codes.
In *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pages 2069–2073. IEEE, 2013.

Chris Peikert.
Lattice cryptography for the internet.
In Michele Mosca, editor, *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014, Waterloo, ON, Canada, October 1-3, 2014. Proceedings*, volume 8772 of *Lecture Notes in Computer Science*, pages 197–219. Springer, 2014.

Nicolas Sendrier.
Encoding information into constant weight words.
In *Information Theory, 2005. ISIT 2005. Proceedings. International Symposium on*, pages 435–438. IEEE, 2005.

Paper available @ http://unil.im/ouroboros

Rank Metric Interlude (1/2)

Rank metric defined over (finite) extensions of finite fields

- $\mathbb{F}_q$ a finite field with q a power of a prime.
- $\mathbb{F}_{q^m}$ an extension of degree m of $\mathbb{F}_q$.
- $\mathbb{F}_{q^m}$ can be seen as a vector space on $\mathbb{F}_q$.
- $\mathcal{B} = (b_1, ..., b_m)$ a basis of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$.

Let $\mathbf{v} = (v_1, \ldots, v_n)$ be a word of length n in $\mathbb{F}_{q^m}$.

Any coordinate $v_j = \sum_{i=1}^{m} v_{ij} b_i$ with $v_{ij} \in \mathbb{F}_q$.

$$\mathbf{v} = (v_1, ..., v_n) \rightarrow \mathbf{V} = \begin{pmatrix} v_{11} & v_{12} & \cdots & v_{1n} \\ v_{21} & v_{22} & \cdots & v_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ v_{m1} & v_{m2} & \cdots & v_{mn} \end{pmatrix}$$

Rank weight of word

$\mathbf{v}$ has rank $r = \mathrm{rank}(\mathbf{v})$ iff the rank of $\mathbf{V} = (v_{ij})_{ij}$ is r.
Equivalently $\mathrm{rank}(\mathbf{v}) = r \Leftrightarrow v_j \in V_r \subset \mathbb{F}_{q^m}^n$ with $\dim(V_r)$=r.

Rank Metric Interlude (1/2)

Rank metric defined over (finite) extensions of finite fields

- $\mathbb{F}_q$ a finite field with q a power of a prime.
- $\mathbb{F}_{q^m}$ an extension of degree m of $\mathbb{F}_q$.
- $\mathbb{F}_{q^m}$ can be seen as a vector space on $\mathbb{F}_q$.
- $\mathcal{B} = (b_1, ..., b_m)$ a basis of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$.

Let $\mathbf{v} = (v_1, \ldots, v_n)$ be a word of length n in $\mathbb{F}_{q^m}$.

Any coordinate $v_j = \sum_{i=1}^{m} v_{ij} b_i$ with $v_{ij} \in \mathbb{F}_q$.

$$\mathbf{v} = (v_1, ..., v_n) \rightarrow \mathbf{V} = \begin{pmatrix} v_{11} & v_{12} & \cdots & v_{1n} \\ v_{21} & v_{22} & \cdots & v_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ v_{m1} & v_{m2} & \cdots & v_{mn} \end{pmatrix}$$

Rank weight of word

$\mathbf{v}$ has rank $r = \mathrm{rank}(\mathbf{v})$ iff the rank of $\mathbf{V} = (v_{ij})_{ij}$ is r.
Equivalently $\mathrm{rank}(\mathbf{v}) = r \Leftrightarrow v_j \in V_r \subset \mathbb{F}_{q^m}^n$ with $\dim(V_r)$=r.

Rank Metric Interlude (1/2)

Rank metric defined over (finite) extensions of finite fields

- $\mathbb{F}_q$ a finite field with q a power of a prime.
- $\mathbb{F}_{q^m}$ an extension of degree m of $\mathbb{F}_q$.
- $\mathbb{F}_{q^m}$ can be seen as a vector space on $\mathbb{F}_q$.
- $\mathcal{B} = (b_1, ..., b_m)$ a basis of $\mathbb{F}_{q^m}$ over $\mathbb{F}_q$.

Let $\mathbf{v} = (v_1, \ldots, v_n)$ be a word of length n in $\mathbb{F}_{q^m}$.

Any coordinate $v_j = \sum_{i=1}^{m} v_{ij} b_i$ with $v_{ij} \in \mathbb{F}_q$.

$$\mathbf{v} = (v_1, ..., v_n) \rightarrow \mathbf{V} = \begin{pmatrix} v_{11} & v_{12} & \cdots & v_{1n} \\ v_{21} & v_{22} & \cdots & v_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ v_{m1} & v_{m2} & \cdots & v_{mn} \end{pmatrix}$$

Rank weight of word

$\mathbf{v}$ has rank $r = \mathrm{rank}(\mathbf{v})$ iff the rank of $\mathbf{V} = (v_{ij})_{ij}$ is r.
Equivalently $\mathrm{rank}(\mathbf{v}) = r \Leftrightarrow v_j \in V_r \subset \mathbb{F}_{q^m}^n$ with dim(V_r)=r.

Rank Metric Interlude (2/2)

- Best Known Attacks have worse complexity in rank metric ($2^{\mathcal{O}(n^2)}$) than in Hamming metric ($2^{\mathcal{O}(n)}$)
- Consequence: worse attacks $\Rightarrow$ better parameters

Instance	Ouroboros-R Parameters						
	key size (bits)	n	m	q	w	security	decoding failure
Ouroboros-R-I	1,591	37	43	2	5	100	10^{-4}
Ouroboros-R-II	2,809	53	53	2	5	128	10^{-8}
Ouroboros-R-III	3, 953	59	67	2	6	192	10^{-7}
Ouroboros-R-IV	5, 293	67	79	2	7	256	10^{-5}
Ouroboros-R-V	5, 618	53	53	4	6	256	10^{-10}

Parameter sets for Ouroboros-R in rank metric.

back to conclusion

Rank Metric Interlude (2/2)

- Best Known Attacks have worse complexity in rank metric ($2^{\mathcal{O}(n^2)}$) than in Hamming metric ($2^{\mathcal{O}(n)}$)
- Consequence: worse attacks $\Rightarrow$ better parameters

Instance	Ouroboros-R Parameters						
	key size (bits)	n	m	q	w	security	decoding failure
Ouroboros-R-I	1,591	37	43	2	5	100	10^{-4}
Ouroboros-R-II	2,809	53	53	2	5	128	10^{-8}
Ouroboros-R-III	3, 953	59	67	2	6	192	10^{-7}
Ouroboros-R-IV	5, 293	67	79	2	7	256	10^{-5}
Ouroboros-R-V	5, 618	53	53	4	6	256	10^{-10}

Parameter sets for Ouroboros-R in rank metric.

back to conclusion

Rank Metric Interlude (2/2)

- Best Known Attacks have worse complexity in rank metric ($2^{\mathcal{O}(n^2)}$) than in Hamming metric ($2^{\mathcal{O}(n)}$)
- Consequence: worse attacks $\Rightarrow$ better parameters

Instance	Ouroboros-R Parameters						
	key size (bits)	n	m	q	w	security	decoding failure
Ouroboros-R-I	1,591	37	43	2	5	100	10^{-4}
Ouroboros-R-II	2,809	53	53	2	5	128	10^{-8}
Ouroboros-R-III	3,953	59	67	2	6	192	10^{-7}
Ouroboros-R-IV	5,293	67	79	2	7	256	10^{-5}
Ouroboros-R-V	5,618	53	53	4	6	256	10^{-10}

Parameter sets for Ouroboros-R in rank metric.

back to conclusion

Sketch of proof

Sequence of games from $\mathsf{Enc}(\epsilon_0)$ to $\mathsf{Enc}(\epsilon_1)$

$\mathsf{Enc}(\epsilon_0) \longrightarrow \mathsf{Enc}_{\mathbf{s}^\star}(\epsilon_0) \longrightarrow \mathsf{Enc}_{\mathbf{s}^\star,\mathbf{r}^\star}(\epsilon_0)$

$\downarrow$

$\mathsf{Enc}(\epsilon_1) \longleftarrow \mathsf{Enc}_{\mathbf{s}^\star}(\epsilon_1) \longleftarrow \mathsf{Enc}_{\mathbf{s}^\star,\mathbf{r}^\star}(\epsilon_1)$

$$\mathbf{Adv}^{\mathrm{ind}}_{\mathcal{E},\mathcal{A}}(\lambda) \leq 2 \cdot \left(\mathbf{Adv}^{\text{2-DQCSD}}(\lambda) + \mathbf{Adv}^{\text{3-DQCSD}}(\lambda)\right)$$

back to security

Sketch of proof

Sequence of games from $\text{Enc}(\epsilon_0)$ to $\text{Enc}(\epsilon_1)$

$\text{Enc}(\epsilon_0) \longrightarrow \text{Enc}_{\mathbf{s}^\star}(\epsilon_0) \longrightarrow \text{Enc}_{\mathbf{s}^\star,\mathbf{r}^\star}(\epsilon_0)$

$\downarrow$

$\text{Enc}(\epsilon_1) \longleftarrow \text{Enc}_{\mathbf{s}^\star}(\epsilon_1) \longleftarrow \text{Enc}_{\mathbf{s}^\star,\mathbf{r}^\star}(\epsilon_1)$

$$\mathbf{Adv}^{\text{ind}}_{\mathcal{E},\mathcal{A}}(\lambda) \leq 2 \cdot \left(\mathbf{Adv}^{\text{2-DQCSD}}(\lambda) + \mathbf{Adv}^{\text{3-DQCSD}}(\lambda)\right)$$

back to security

Sketch of proof

Sequence of games from $\text{Enc}(\epsilon_0)$ to $\text{Enc}(\epsilon_1)$

$\text{Enc}(\epsilon_0) \longrightarrow \text{Enc}_{\mathbf{s}^\star}(\epsilon_0) \longrightarrow \text{Enc}_{\mathbf{s}^\star,\mathbf{r}^\star}(\epsilon_0)$

$\downarrow$

$\text{Enc}(\epsilon_1) \longleftarrow \text{Enc}_{\mathbf{s}^\star}(\epsilon_1) \longleftarrow \text{Enc}_{\mathbf{s}^\star,\mathbf{r}^\star}(\epsilon_1)$

$$\mathbf{Adv}^{\text{ind}}_{\mathcal{E},\mathcal{A}}(\lambda) \leq 2 \cdot \left(\mathbf{Adv}^{\text{2-DQCSD}}(\lambda) + \mathbf{Adv}^{\text{3-DQCSD}}(\lambda)\right)$$

back to security

Sketch of proof

Sequence of games from $\mathsf{Enc}(\epsilon_0)$ to $\mathsf{Enc}(\epsilon_1)$

$\mathsf{Enc}(\epsilon_0) \longrightarrow \mathsf{Enc}_{\mathbf{s}^\star}(\epsilon_0) \longrightarrow \mathsf{Enc}_{\mathbf{s}^\star,\mathbf{r}^\star}(\epsilon_0)$

$\downarrow$

$\mathsf{Enc}(\epsilon_1) \longleftarrow \mathsf{Enc}_{\mathbf{s}^\star}(\epsilon_1) \longleftarrow \mathsf{Enc}_{\mathbf{s}^\star,\mathbf{r}^\star}(\epsilon_1)$

$$\mathbf{Adv}^{\mathrm{ind}}_{\mathcal{E},\mathcal{A}}(\lambda) \leq 2 \cdot \left(\mathbf{Adv}^{\text{2-DQCSD}}(\lambda) + \mathbf{Adv}^{\text{3-DQCSD}}(\lambda)\right)$$

back to security

Sketch of proof

Sequence of games from $\mathsf{Enc}(\epsilon_0)$ to $\mathsf{Enc}(\epsilon_1)$

$\mathsf{Enc}(\epsilon_0) \longrightarrow \mathsf{Enc}_{\mathbf{s}^\star}(\epsilon_0) \longrightarrow \mathsf{Enc}_{\mathbf{s}^\star,\mathbf{r}^\star}(\epsilon_0)$

$\downarrow$

$\mathsf{Enc}(\epsilon_1) \longleftarrow \mathsf{Enc}_{\mathbf{s}^\star}(\epsilon_1) \longleftarrow \mathsf{Enc}_{\mathbf{s}^\star,\mathbf{r}^\star}(\epsilon_1)$

$$\mathbf{Adv}^{\text{ind}}_{\mathcal{E},\mathcal{A}}(\lambda) \leq 2 \cdot \left(\mathbf{Adv}^{\text{2-DQCSD}}(\lambda) + \mathbf{Adv}^{\text{3-DQCSD}}(\lambda) \right)$$

back to security

Sketch of proof

Sequence of games from $\mathsf{Enc}(\epsilon_0)$ to $\mathsf{Enc}(\epsilon_1)$

$$\mathsf{Enc}(\epsilon_0) \longrightarrow \mathsf{Enc}_{\mathbf{s}^\star}(\epsilon_0) \longrightarrow \mathsf{Enc}_{\mathbf{s}^\star,\mathbf{r}^\star}(\epsilon_0)$$

$$\downarrow$$

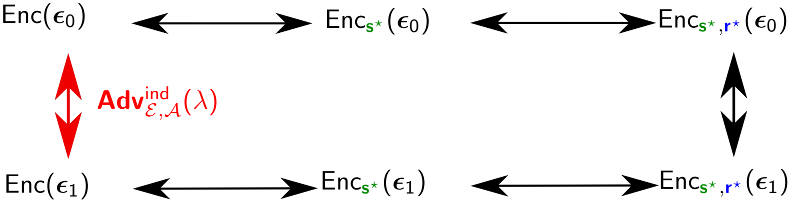
$$\mathsf{Enc}(\epsilon_1) \longleftarrow \mathsf{Enc}_{\mathbf{s}^\star}(\epsilon_1) \longleftarrow \mathsf{Enc}_{\mathbf{s}^\star,\mathbf{r}^\star}(\epsilon_1)$$

$$\mathbf{Adv}^{\mathrm{ind}}_{\mathcal{E},\mathcal{A}}(\lambda) \leq 2 \cdot \left(\mathbf{Adv}^{\text{2-DQCSD}}(\lambda) + \mathbf{Adv}^{\text{3-DQCSD}}(\lambda) \right)$$

back to security

Sketch of proof

Sequence of games from $\mathsf{Enc}(\epsilon_0)$ to $\mathsf{Enc}(\epsilon_1)$

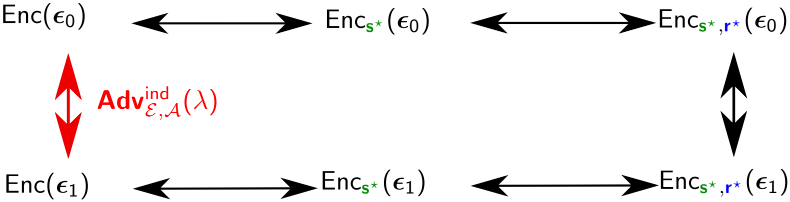


$$\mathbf{Adv}^{\mathsf{ind}}_{\mathcal{E},\mathcal{A}}(\lambda) \leq 2 \cdot \left(\mathbf{Adv}^{\text{2-DQCSD}}(\lambda) + \mathbf{Adv}^{\text{3-DQCSD}}(\lambda)\right)$$

back to security

Sketch of proof

Sequence of games from $\mathsf{Enc}(\epsilon_0)$ to $\mathsf{Enc}(\epsilon_1)$



$$\mathbf{Adv}^{\mathsf{ind}}_{\mathcal{E},\mathcal{A}}(\lambda) \leq 2 \cdot \left(\mathbf{Adv}^{\mathsf{2\text{-}DQCSD}}(\lambda) + \mathbf{Adv}^{\mathsf{3\text{-}DQCSD}}(\lambda)\right)$$

back to security