Code-Based Cryptography Workshop 2012

9 – 11 May 2012, Lyngby, Denmark

On the Design of Code-Based Signatures

Ayoub Otmani

ayoub.otmani@unicaen.fr

GREYC

Outline

1. Fiat-Shamir paradigm

2. Hash-and-Sign paradigm

3. “Lossy Source Coding” Signatures (joint work with J.P. Tillich)

About this Lecture . . .

▷ **Focus** on "classical" signatures

- Authentication
- Integrity
- Non-repudiation

▷ "Sophisticated" signatures are **not treated**:

Ring signature, threshold ring signature, blind signature, undeniable signature, . . .

Signature Scheme

Definition. A *signature scheme* is given by **three** algorithms:

- ▷ $(\mathrm{sk}, \mathrm{pk}) \longleftarrow \mathtt{KeyGen}(\lambda)$ where λ is a security parameter
- ▷ $\sigma \longleftarrow \mathtt{Sign}(\mathrm{sk}, \boldsymbol{m})$ where $\boldsymbol{m} \in \{0,1\}^*$
- ▷ $b \longleftarrow \mathtt{Verify}(\mathrm{pk}, \boldsymbol{m}, \sigma)$ where $b \in \{\mathtt{accept}, \mathtt{reject}\}$ and such that:

$$\mathtt{Verify}\Big(\mathrm{pk}, \boldsymbol{m}, \mathtt{Sign}(\mathrm{sk}, \boldsymbol{m})\Big) = \mathtt{accept}$$

Security Model Terminology

▷ **Forger** = Attacker

▷ Forger's **goal**

- *Universal* Forgery (key-recovery, . . .)
- *Existential* Forgery

▷ Forger's **means**

- *No*-message
- *Known* message
- *Chosen* message

I. Fiat-Shamir Paradigm

Fiat-Shamir Paradigm ('86)

▷ **Generic** method for deriving a signature scheme from **any 3-pass identification** scheme

- Replacing **Verifier's** action's by a **hash function** h
- Secure if the identification scheme is secure against **impersonation** (Abdalla-An-Bellare-Namprempre '02)

▷ **Code-based** identification scheme (zero-knowledge protocol)

- Stern ('93)
- Veron ('96)

3-Pass Identification Scheme

	$\mathbb{P}$		$\mathbb{V}$
1.	$(\texttt{sk}, \texttt{pk}) \leftarrow \texttt{KeyGen}(\lambda)$		
2.		$\xrightarrow{\textbf{a}=\texttt{Commit}(\texttt{sk},\texttt{nonce})}$	
3.		$\xleftarrow{\textbf{b}=\texttt{Challenge}(\lambda,\texttt{nonce})}$	
4.		$\xrightarrow{\textbf{c}=\texttt{Response}(\texttt{sk},a,b)}$	
5.			$\texttt{Verify}(\texttt{pk}, a, b, c)$

$$\texttt{Verify}(\texttt{pk}, a, b, c) = \texttt{accept} \quad \text{if} \quad \begin{cases} a = \texttt{Commit}(\texttt{sk}, \texttt{nonce}) \\ b = \texttt{Challenge}(\lambda) \\ c = \texttt{Response}(\texttt{sk}, a, b) \end{cases}$$

Fiat-Shamir Paradigm

▷ **Signature** σ is computed by means of the steps:

1. $a = \texttt{Commit}(\texttt{sk}, \texttt{nonce})$
2. $b = h(a, \boldsymbol{m})$
3. $c = \texttt{Response}(\texttt{sk}, a, b)$
4. $\sigma = (a, c)$

▷ **Verification** is done by computing $b' = h(a, \boldsymbol{m})$ and checking:

$$\texttt{Verify}(\texttt{pk}, a, b', c) = \texttt{accept}$$

▷ Efficiency with Stern's protocol:

- **Fast** operations
- **Large** signatures $\mathcal{O}(n \log n)$ bits
- **Large** keys $\mathcal{O}(n^2)$ (**fixed** rate)

II. Hash-and-Sign Paradigm

Introduction

- Deriving a signature scheme from a **public-key encryption** $(D_{\mathrm{sk}}, E_{\mathrm{pk}})$

- For **efficiency**, $\boldsymbol{m}$ should be a **fixed** length bit-string

  $\rightsquigarrow$ Signing a hash value $h(\boldsymbol{m})$

- Signature of $\boldsymbol{m}$ is $\sigma = D_{\mathrm{sk}}\Big(h(\boldsymbol{m})\Big)$

- Verification of $(\boldsymbol{m}, \sigma')$ checks if:

$$E_{\mathrm{pk}}(\sigma') = h(\boldsymbol{m})$$

- Random Oracle Model (ROM) $\rightsquigarrow$ h is a **random** function

Niederreiter Cryptosystem

▷ **Public key**: Parity-check matrix $\boldsymbol{H}$ of a binary Goppa code of length n and dimension k

▷ **Secret Key**: t-correcting algorithm ψ

▷ **Encryption**: $\boldsymbol{x} \rightsquigarrow \boldsymbol{y} = \boldsymbol{H}\boldsymbol{x}^T$ with $\boldsymbol{x}$ of **weight** t

▷ **Decryption**: compute $\psi(\boldsymbol{y})$ and recover $\boldsymbol{x}$

Assumption. $k = n - mt \rightsquigarrow \boldsymbol{H}$ is a $mt \times n$ matrix

Signing with Niederreiter Scheme

▷ ROM implies to perform **complete decoding**

▷ **But** probability that a randomly drawn vector in $\{0,1\}^n$ is at distance t from a codeword

$$\frac{\binom{n}{t}}{2^{mt}} \geqslant \frac{\binom{n}{t}}{n^t} \simeq \frac{1}{t!} \rightsquigarrow t \text{ has to be } \textbf{small}$$

▷ Courtois-Finiasz-Sendrier ('01) proposed a method for producing Niederreiter signatures for **any** hash value:

- **Modifying** $\boldsymbol{m}$ until it lies within distance t from a codeword
- **Efficiency** implies to take small t $(t \leqslant 12)$
- **Security** implies to take large n $(n \geqslant 16)$

CFS Scheme

$\texttt{Sign}(\boldsymbol{m}, \psi)$

1. $\boldsymbol{s} = h(\boldsymbol{m})$;
2. $i = 0$;
3. Repeat
4. $\quad i = i + 1$;
5. $\quad s_i = h(\boldsymbol{s}, i)$;
6. $\quad \boldsymbol{z} = \psi(s_i)$;
7. until $\boldsymbol{z} \neq \emptyset$;
8. Return $\sigma = (\boldsymbol{z}, i)$;

CFS Scheme

$\texttt{Verify}\left(\boldsymbol{m}, (\boldsymbol{z}, i), \boldsymbol{H}, t\right)$

1. $\boldsymbol{s} = h(\boldsymbol{m})$;
2. $\boldsymbol{s}_i = h(\boldsymbol{s}, i)$
3. $\texttt{If}$ $\left(\boldsymbol{s}_i = \boldsymbol{H}\boldsymbol{z}^T \text{ and } \mathsf{wt}(\boldsymbol{z}) = t\right)$ $\texttt{then}$
4. $\quad$ $\texttt{Return accept}$;
5. $\texttt{else}$
6. $\quad$ $\texttt{Return reject}$;

Performances (80-bit)

Performances with $n = 2^m$ and $k = n - mt$

	Signature	Verification	Length	Key size (bits)
(m, t)	$t!\ t^2\ m^3$	$t^2 m$	$tm + \log_2 t$	$tm2^m$
$(21, 10)$	$2^{41.6}$	$2^{11.0}$	213.3	$2^{28.7}$
$(19, 11)$	$2^{44.9}$	$2^{11.1}$	212.4	$2^{26.7}$
$(15, 12)$	$2^{47.7}$	$2^{11.0}$	183.5	$2^{22.4}$

CFS Scheme - Alternative Way

▷ Decoding **any** syndrome by **increasing** the number of errors $t \rightsquigarrow t + \delta$ where

$$\binom{n}{t+\delta} \geqslant 2^{mt}$$

▷ These extra δ errors found through an **exhaustive search**

$\rightsquigarrow$ Signing time increased by $\binom{n}{\delta}$

▷ **Real gain** when $\displaystyle\binom{n}{\delta} < t! \rightsquigarrow$ generally $\delta \leqslant 2$

Security

▷ **Key-Recovery Attack**

- Recovering the support and the Goppa polynomial
- Best attack performs an exhaustive search on polynomials of degree t and applies Sendrier's SSA algorithm
- Time complexity $\mathcal{O}(2^{mt})$ for polynomials with coefficients in $\mathbb{F}_{2^m}$

▷ **Existential Forgery under No-Message Attack**

- Syndrome Decoding Problem

▷ **Existential Forgery under Chosen Message Attack**

- "One-out-of-many Syndrome" Decoding Problem

Existential Forgery - Algorithmic Problems

Definition. (Syndrome Decoding Problem)

- **Input.** $\boldsymbol{H}$, a syndrome $\boldsymbol{s}$ and weight t
- **Output.** word $\boldsymbol{e}$ of weight $\leqslant t$ such that $\boldsymbol{H}\boldsymbol{e}^T = \boldsymbol{s}$

Definition. ("One-out-of-many Syndrome" Decoding Problem)

- **Input.** $\boldsymbol{H}$, a list L of syndromes and weight t
- **Output.** word $\boldsymbol{e}$ of weight $\leqslant t$ and a syndrome $\boldsymbol{s}$ in L such that $\boldsymbol{H}\boldsymbol{e}^T = \boldsymbol{s}$

Existing Approaches

▷ **Syndrome Decoding Problem**

- Information Set Decoding (ISD) algoritm $\rightsquigarrow$ Time complexity $\mathcal{O}\left(2^{mt/2}\right)$

▷ **"One-out-of-many Syndrome" Decoding Problem** (Sendrier '11)

- Johansson and Jönsson's algorithm $\rightsquigarrow$ Time complexity $\mathcal{O}\left(2^{mt/2}\right)$
- Bleinchebacher's Attack $\rightsquigarrow$ Time complexity $\mathcal{O}\left(2^{mt/3}\right)$

Bleinchebacher's Attack - Preliminaries

▷ Based on the **Generalized Birthday Paradox** Problem

- **Input.** $f : E \longrightarrow \{0,1\}^r$ and $\ell \geqslant 1$
- **Output.** Finding $x_1, \ldots, x_\ell$ in E such that $\bigoplus_{i=1}^{\ell} f(x_i) = 0$

▷ Birthday Paradox $O\left(2^{\frac{r}{2}}\right)$

▷ Wagner ('02) showed that when $\ell = 4$ then time/memory complexity $\mathcal{O}(2^{r/3})$

Bleinchebacher's Attack

▷ Searching for words $\boldsymbol{e}_1$, $\boldsymbol{e}_2$, $\boldsymbol{e}_3$ of weight $t/3$ and $h(\boldsymbol{m})$ such that

$$\boldsymbol{H}\boldsymbol{e}_1^T + \boldsymbol{H}\boldsymbol{e}_2^T + \boldsymbol{H}\boldsymbol{e}_3^T + h(\boldsymbol{m}) = 0$$

---

1. Build 3 lists L_0, L_1, L_2 of $\binom{n}{t/3}$ syndromes of words of weight $t/3$
2. New list L_0' from L_0 into L_1 by XORing and keeping the resulting syndromes whose first $mt/3$ positions are zero
3. Build one (virtual) list L_3 of $2^{mt/3}$ target hash values
4. Merge L_2 and L_3 into L_1' by XORing and keeping the resulting syndromes whose first $mt/3$ positions are zero
5. Search for a collision between L_0' and L_1' over the last $2mt/3$ bits

---

Remark.

▷ At least one solution if $\binom{n}{t/3} \geqslant 2^{mt/3}$

▷ Time/Memory is about $\mathcal{O}(2^{mt/3})$

Parallel CFS (Finiasz '10)

▷ Reparation of CFS

▷ Sign a message $\boldsymbol{m}$ twice (or i times) by means of two (or i) different hash functions h_1 and h_2 (or $\ldots, h_i$)

▷ For avoiding (trivial) attacks, the two signatures has to be related $\rightsquigarrow$ signing with second version of CFS

Finding $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$ of weight at most $t + \delta$ such that

$$\boldsymbol{H}\boldsymbol{e}_1^T = h_1(\boldsymbol{m}) \text{ and } \boldsymbol{H}\boldsymbol{e}_2^T = h_2(\boldsymbol{m})$$

▷ Time/memory complexity Bleinchebacher's attack becomes $\mathcal{O}(2^{2mt/3})$

m	t	i	Key size	Cost	Size
18	9	3	5.0 MB	$2^{20.0}$	288
19	9	2	10.7 MB	$2^{19.5}$	206
20	8	3	20.0 MB	$2^{16.9}$	294

80-bit security/$\delta = 2$

Quasi-Dyadic CFS Signature

▷ CFS-like scheme by Barreto-Cayrel-Misoczki-Niebhur ('11)

▷ Based on binary **Quasi-dyadic** Goppa codes (Cauchy matrices)

▷ **Smaller** keys than CFS scheme (reduction by a factor t)

Cauchy Matrix

- $\boldsymbol{z} = (z_0, \ldots, z_{t-1}) \in \mathbb{F}_{q^m}^t$
- $\boldsymbol{x} = (x_0, \ldots, x_{n-1}) \in \mathbb{F}_{q^m}^n$ with $x_i \neq z_j$

Definition. $\boldsymbol{C}(\boldsymbol{z}, \boldsymbol{x})$ is **Cauchy** matrix if

$$\boldsymbol{C}(\boldsymbol{z}, \boldsymbol{x}) \stackrel{\text{def}}{=} \begin{pmatrix} \frac{1}{z_0 - x_0} & \cdots & \frac{1}{z_0 - x_{n-1}} \\ \vdots & \ddots & \vdots \\ \frac{1}{z_{t-1} - x_0} & \cdots & \frac{1}{z_{t-1} - x_{n-1}} \end{pmatrix}$$

Proposition. The code defined by the parity-check $\boldsymbol{C}(\boldsymbol{z}, \boldsymbol{x})$ is a Goppa code whose polynomial is $\gamma(z) = \prod_{i=0}^{t-1} (z - z_i)$

Dyadic Matrix

Definition.

▷ $n = 2^\ell$ for some integer $\ell \geqslant 1$

▷ $\boldsymbol{h} = (h_0, \ldots, h_{n-1})$ from $\mathbb{F}_q^n$

$$\boldsymbol{\Delta}(\boldsymbol{h}) \stackrel{\text{def}}{=} \left(h_{i \oplus j}\right)_{\substack{0 \leqslant i \leqslant n-1 \\ 0 \leqslant j \leqslant n-1}}$$

▷ $\boldsymbol{\Delta}(\boldsymbol{h})$ is called a **dyadic** matrix

Proposition. (Misoczki-Barreto '09)

▷ $\boldsymbol{\Delta}(\boldsymbol{h})$ is a Cauchy matrix **if and only if** $\mathbb{F}_q$ is of characteristic 2 and

$$\frac{1}{h_{i \oplus j}} = \frac{1}{h_j} + \frac{1}{h_i} + \frac{1}{h_0}$$

▷ Furthermore, for any $\theta \in \mathbb{F}_q$, let $z_i \stackrel{\text{def}}{=} 1/h_i + \theta$ and $x_j \stackrel{\text{def}}{=} 1/h_j + 1/h_0 + \theta$

$$\boldsymbol{\Delta}(\boldsymbol{h}) = \boldsymbol{C}(\boldsymbol{z}, \boldsymbol{x})$$

Quasi-Dyadic CFS - Key Generation

▷ Choose t and let λ be the **smallest** integer such that $t \leqslant 2^\lambda$

$$\rightsquigarrow (\mathtt{sk}, \mathtt{pk}) = (\boldsymbol{f}, \boldsymbol{G})$$

▷ $\boldsymbol{G}$ is a binary $k \times n$ generator matrix with $n = n_0 2^\lambda$ and $\boldsymbol{f} \in \mathbb{F}_{2^m}^n$ such that:

$$\boldsymbol{G}\boldsymbol{f}^T = 0$$

▷ $\boldsymbol{f}$ is "almost" the first row of a Dyadic Cauchy matrix

- "**Inside-Block**" equations: $0 \leqslant a \leqslant n_0 - 1$ and $0 \leqslant i, j \leqslant 2^\lambda - 1$

$$\frac{1}{f_{a2^\lambda + i \oplus j}} = \frac{1}{f_{a2^\lambda \oplus i}} + \frac{1}{f_{a2^\lambda \oplus j}} + \frac{1}{f_{a2^\lambda}}$$

- "**Between-Block**" equations: $0 \leqslant a \leqslant n_0 - 1$ and $0 \leqslant i \leqslant 2^\lambda - 1$

$$\frac{1}{f_{a2^\lambda + i}} + \frac{1}{f_{a2^\lambda}} = \frac{1}{f_i} + \frac{1}{f_0}$$

Algebraic Attack - Faugère -Najahi-O-Perret-Tillich ('12)

Fact.

▷ $\boldsymbol{G} = \left(\begin{array}{c|c} \boldsymbol{I}_k & \boldsymbol{R} \end{array} \right) \quad \rightsquigarrow n - k = mt$ "free" variables

▷ "Inside-Block" relations **imply** that f_i with $0 \leqslant i \leqslant 2^\lambda - 1$ is **solely determined** by $f_0, f_1, f_2, \ldots, f_{2^\lambda - 1}$

▷ **One** f_i can be fixed to an **arbitrary** value $\rightsquigarrow f_0$

Assumption. $f_1, f_2, \ldots, f_{2^\lambda - 1}$ are **known** $\rightsquigarrow mt - 2^\lambda$ "free" variables

$$0 \leqslant i \leqslant 2^\lambda - 1 : \qquad K_i \stackrel{\text{def}}{=} \frac{1}{f_i} + \frac{1}{f_0}$$

Algebraic Attack

- ▷ "Between-Block" equations become **quadratic** equations

$$K_i \; f_{a2^\lambda} f_{a2^\lambda+i} + f_{a2^\lambda+i} + f_{a2^\lambda} = 0$$

- ▷ **Number** of quadratic equations: $\left(\frac{n}{2^\lambda} - 1\right)(2^\lambda - 1)$

- ▷ Quasi-Dyadic CFS parameters are such that:
  - $t \leqslant 12 \rightsquigarrow \lambda \leqslant 4$
  - n is **large** with $n \leqslant 2^m - 2^\lambda$

$\rightsquigarrow$ Number of equations $\gg$ number of variables

Linearization Technique

▷ Each product $f_i f_j$ is replaced by a **new** variable $z_{i,j}$

$\rightsquigarrow$ Total number of new variables $\binom{mt - 2^\lambda + 2}{2}$

▷ At **least one solution** to the linearized system if:

$$\left(\frac{n}{2^\lambda} - 1\right)(2^\lambda - 1) \geqslant \binom{mt - 2^\lambda + 2}{2}$$

▷ **All** the proposed parameters **satisfy** this condition

Example.

- $t = 8 \rightsquigarrow m \geqslant 13$
- $t = 10 \rightsquigarrow m \geqslant 13$
- $t = 12 \rightsquigarrow m \geqslant 14$

Complexity of the Attack

▷ Exhaustive search for determining each $K_i \rightsquigarrow \mathcal{O}\left(2^{\lambda m}\right)$

▷ Linear algebra $\mathcal{O}\left((mt)^{2\omega}\right)$ where $2 \leqslant \omega \leqslant 3$

$(m,t)^1$	Exhaustive search ($\lambda = 4$)	Linear algebra ($\omega = 2.376$)
$(21, 10)$	2^{84}	2^{34}
$(19, 11)$	2^{76}	2^{34}
$(15, 12)$	2^{60}	2^{33}

[1] 80-bit security

▷ **Open issue.** Improving the exhaustive search part (still in progress)

Signing without Decoding (Kabatianskii-Krouk-Smeets '97)

▷ Possible if one is able to find:

- **Signing function** $\Sigma : \boldsymbol{m} \longmapsto \sigma$ of weight t
- **Verification function** χ such that $\chi(\boldsymbol{m}) = \boldsymbol{H}\sigma^T$

▷ It would allow to sign with random linear codes

▷ KKS proposed **linear maps** for Σ and χ

$$\Sigma : \boldsymbol{m} \longmapsto \boldsymbol{m}\boldsymbol{G}$$

$$\chi : \boldsymbol{m} \longmapsto \boldsymbol{F}\boldsymbol{m}^T$$

Assumption. $\boldsymbol{G}$ generates a linear code whose codewords $\boldsymbol{v}$ are such that:

$$t_1 \leqslant \mathsf{wt}(v) \leqslant t_2$$

KKS Scheme - Key Generation

▷ Security parameter $\rightsquigarrow \delta$, k, n, r, N such that $k < n < r < N$ and $0 < \delta \ll \dfrac{n}{2}$

▷ Pick at random

- $k \times n$ matrix $\boldsymbol{G}$
- $J \subset \{1, \ldots, N\}$ of cardinality n
- $r \times N$ matrix $\boldsymbol{H}$

▷ Compute $r \times k$ matrix $\boldsymbol{F} \stackrel{\text{def}}{=} \boldsymbol{H}(J)\boldsymbol{G}^T$

▷ Set $t_1 \stackrel{\text{def}}{=} \dfrac{n}{2} - \delta$ and $t_2 \stackrel{\text{def}}{=} \dfrac{n}{2} + \delta$

$$\mathtt{sk} = (J, \boldsymbol{G}) \qquad \text{and} \qquad \mathtt{pk} = (\boldsymbol{H}, \boldsymbol{F}, t_1, t_2)$$

KKS Scheme

- $\sigma \leftarrow \texttt{Sign}(\boldsymbol{m})$: Compute σ of $\{1,0\}^N$ such that:

$$\sigma_J = \boldsymbol{mG} \quad \text{and} \quad \sigma_{[1\ldots N]\setminus J} = 0$$

- $\texttt{Verify}(\boldsymbol{m}, \sigma)$

$$\boldsymbol{H}\sigma^T = \boldsymbol{F}\boldsymbol{m}^T \text{ and } t_1 \leqslant \mathrm{wt}(\sigma) \leqslant t_2$$

Preliminary Observations

Notation.

- $\mathscr{S} \overset{\text{def}}{=} \Big\{ \text{Valid KKS message/signature } (\boldsymbol{m}, \sigma) \Big\}$
- $\mathscr{C}_{\text{public}} \overset{\text{def}}{=} \Big\{ \boldsymbol{c} \in \{0,1\}^{k+N} \; : \; \left(\begin{array}{c|c} \boldsymbol{F} & \boldsymbol{H} \end{array} \right) \boldsymbol{c}^T = 0 \Big\}$

Fact.

1. $\mathscr{S}$ is a linear subspace of $\mathscr{C}_{\text{public}}$ because of $\boldsymbol{F}\boldsymbol{m}^T = \boldsymbol{H}\sigma^T$
2. $\mathscr{S}$ is of dimension k

Security of KKS Scheme

1. Basis of $\mathscr{S}$ $\rightsquigarrow$ universal forgery

   KKS scheme is a ℓ-time signature scheme with $\ell < k$

2. If $\sigma_1, \ldots, \sigma_\ell$ are ℓ signatures then $\bigcup_{i=0}^{\ell} \text{support}(\sigma_j) \subset J$

Proposition. $\sigma_1, \ldots, \sigma_\ell$ are codewords of weight of t drawn uniformly and independently

$$\mathbb{E}\left[\left|\bigcup_{i=0}^{\ell} \text{support}(\sigma_j)\right|\right] = n\left(1 - \left(1 - \frac{t}{n}\right)^{\ell}\right)$$

Remark. $t \simeq \frac{n}{2} \rightsquigarrow n(1 - \frac{1}{2^\ell})$ positions of J are known

Corollary. KKS is one-time signature

“Noisy” KKS (Barreto-Misoczki-Simplicio ’11)

Assumption. h ispublic hash function

▷ $(\sigma, \boldsymbol{v}) \leftarrow \texttt{Sign}(\boldsymbol{m})$

- Pick at random $\boldsymbol{e} \in \{0,1\}^N$ such that $\mathrm{wt}(\boldsymbol{e}) = n$
- Compute $\boldsymbol{v} \stackrel{\text{def}}{=} h(\boldsymbol{m}, \boldsymbol{H}\boldsymbol{e}^T)$
- Compute $\boldsymbol{y} \in \{0,1\}^N$ such that:

$$\boldsymbol{y}_J = \boldsymbol{v}\boldsymbol{G} \quad \text{and} \quad \boldsymbol{y}_{[1\ldots N]\setminus J} = 0$$

- $\sigma \stackrel{\text{def}}{=} \boldsymbol{y} + \boldsymbol{e}$

▷ $\texttt{Verify}(\boldsymbol{v}, \sigma)$ checks whether

$$h(\boldsymbol{m}, \boldsymbol{H}\sigma^T + \boldsymbol{F}\boldsymbol{v}^T) = \boldsymbol{v} \quad \text{and} \quad \mathrm{wt}(\sigma) \leqslant 2n$$

Further Observations

Fact.

1. $\mathscr{S}_{[k+1\ldots k+N]\setminus J} = \{0\}$
2. $\mathscr{S}_J$ is a linear code of dimension k containing low-weight words $\simeq n/2$ with

$$n/2 \ll N + k$$

Corollary.

- Recovering $\mathscr{S}$ by applying algorithms searching for low-weight codewords
- $\boldsymbol{F} = \boldsymbol{H}(J)\boldsymbol{G}^T \rightsquigarrow \mathscr{C}_{\text{public}}$ is **not a random** code

Universal Forgery under No-Message Attack (O-Tillich '11)

$$\left(\; \boldsymbol{F} \; \middle| \; \boldsymbol{H} \; \right) \leadsto \mathscr{S} = \text{ Secret}$$

▷ Dumer's ISD algorithm: ℓ, p with p very small

- Random $I \subset \{1, \ldots, N+k\}$ of cardinality $k + K + \ell$
- Outputs $\boldsymbol{x}$ of weight $\simeq n/2$ such that $\boldsymbol{x}_I$ is of weight $2p$

▷ Analysis shows that the attack performs **better** when

- $I \subset \{k+1, \ldots, N+k\}$
- Rates of $\mathscr{S}$ and $\mathscr{C}_{\text{public}}$ are close
- n is small

▷ **Bootstrapping** Second codeword $\boldsymbol{y}$ is found **more easily** from $\boldsymbol{x}$

- Take at random $I \subset \{k+1, \ldots, N+k\} \setminus \text{support}(\boldsymbol{x})$

Open issue. Finding "good" parameters immune against this attack

Instead of Correcting?

▷ "Hash-and-Sign" Paradigm considers $h(\boldsymbol{m})$ as a "noisy" version of signature

$\rightsquigarrow h(m)$ should not be changed

▷ CFS scheme simulates complete decoding

$\rightsquigarrow h(m)$ has to be changed

▷ With J.P. Tillich we propose to rephrase the problem in the framework of **Rate-Distortion Theory** (also called **lossy source coding**)

III. “Lossy Source Coding” Signatures

Rate-Distorsion Theory

▷ Aiming at **representing/estimating/quantizing** a source (= random variable $X(\omega)$) taking **infinite** numbers of values by means of a **finite** number N of values

$$X(\omega) \in \mathcal{X} \rightsquigarrow \mathcal{R}(X) \overset{\text{def}}{=} \left\{ \hat{X}(\omega_1), \ldots, \hat{X}(\omega_N) \right\}$$

Example.

- Representation of real numbers with a fixed number of bits
- Lossy-data compression

▷ Representation **cannot be done exactly** $\rightsquigarrow$ **maximum distorsion** D

$$\forall \omega : \quad \text{dist}\Big(\hat{X}(\omega), X(\omega) \Big) \leqslant D$$

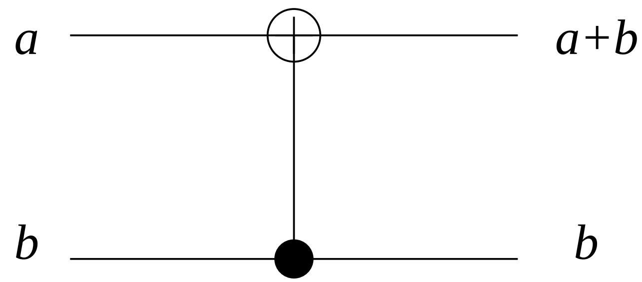
▷ Choosing N **optimal** values

$$X(\omega) \rightsquigarrow \text{Find the } \textbf{closest} \text{ point in } \mathcal{R}(X)$$

Polar Codes (Arikan '07)

▷ Length $N = 2^n$

▷ Encoding based on Fast Fourier Transform architecture



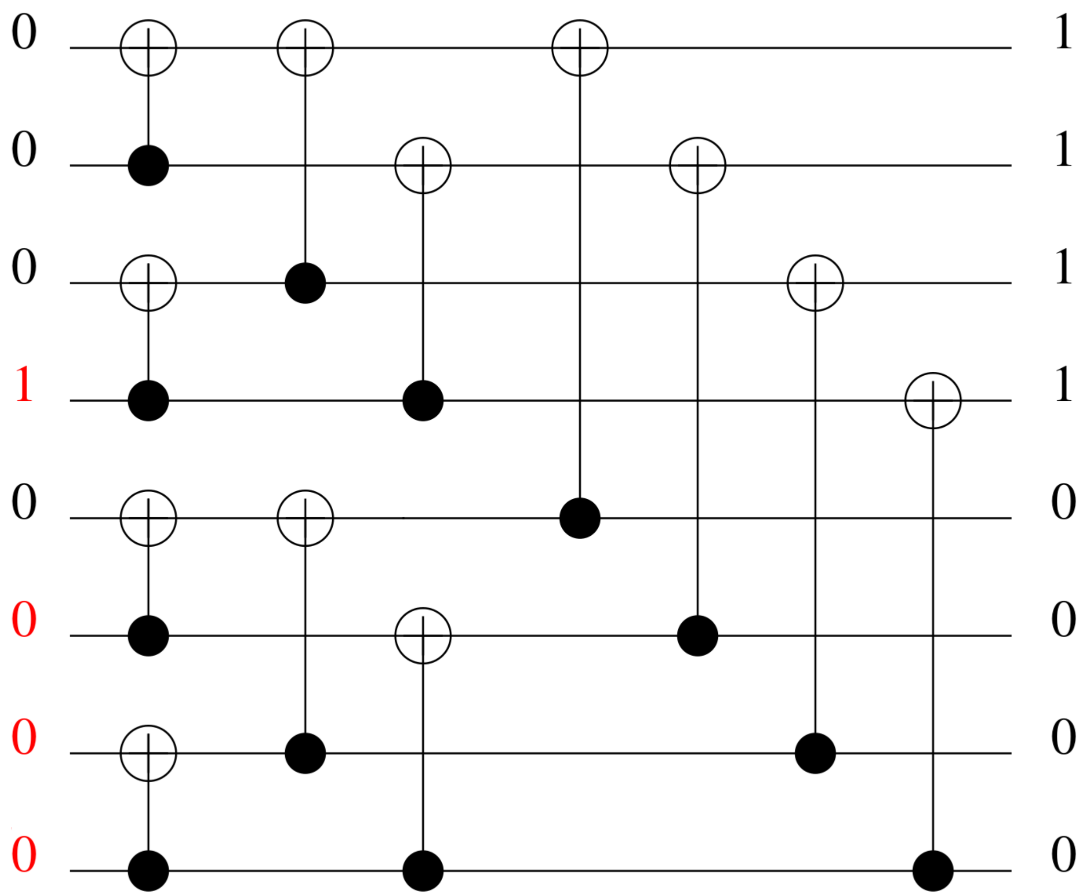
▷ **Encoding**/**Decoding** can be made in $\mathcal{O}(N \log N)$ operations

▷ **Capacity-achieving** codes for **any** binary memoryless channel

▷ **Optimal** for lossy source coding of a binary symetric source (Korada '10)

Encoding with Polar Codes (I)

Example. $n = 3$



▷ Which code do we get?

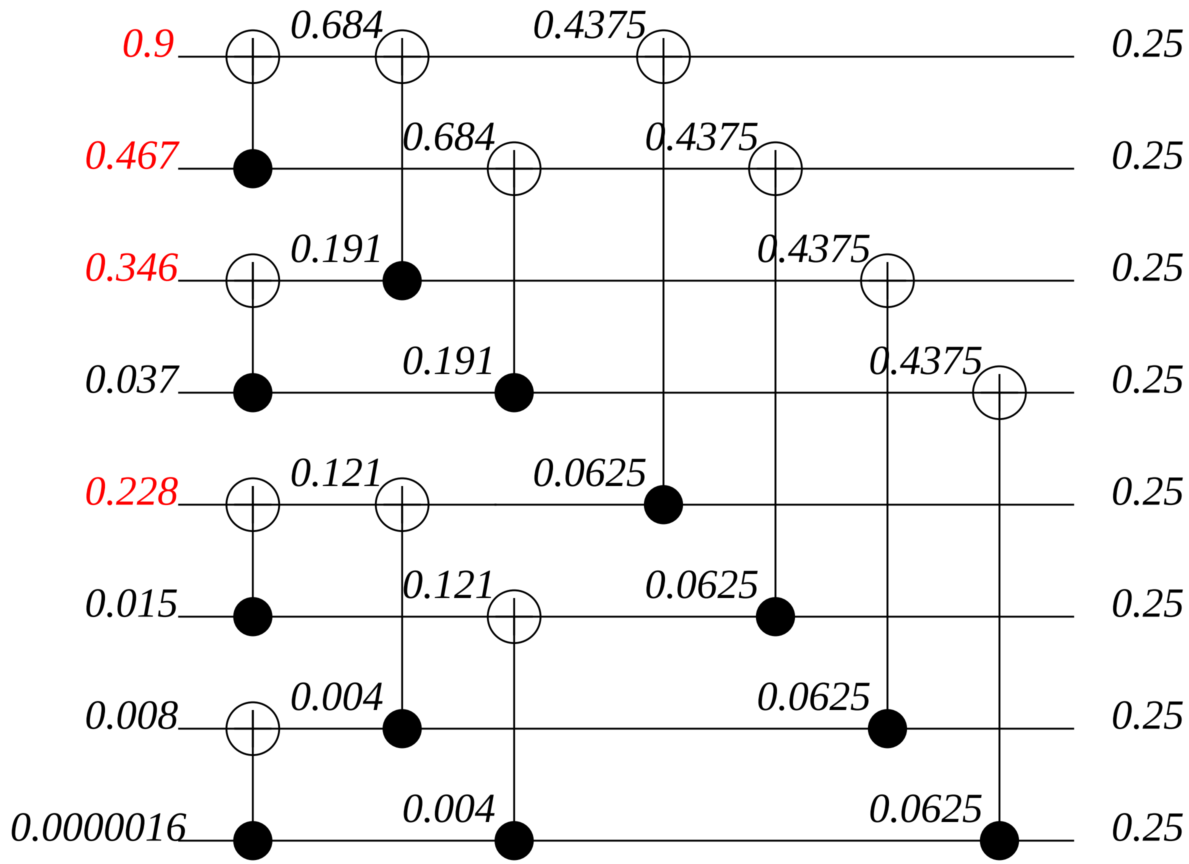
Encoding with Polar Codes (II)

Extended Hamming code $[8, 4, 4]$ defined by the generator matrix:

$$\boldsymbol{G} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$\rightsquigarrow$ Which entries have to be kept zero?

"Polarization" Phenomenon



▷ **General rule** For a code of length N and dimension K then set to 0 the $N-K$ **worst** positions

▷ **Entries** set to zero are called "frozen" (red)

Using Polar Codes in Cryptography

▷ Adding **diversity**

- Changing the alphabet from binary to $GF(4) = \{0, 1, w, w^2\}$
- Not considering only one transform $\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$ but a set of transforms

$$\left\{ \begin{pmatrix} 1 & w \\ w & 1 \end{pmatrix}, \begin{pmatrix} w^2 & w \\ 1 & 1 \end{pmatrix}, \begin{pmatrix} w^2 & 1 \\ w & 1 \end{pmatrix} \right\}$$

- Randomly picking 2^{n-1} transforms at each level i of $\{1, \dots, n\}$

▷ **Expanding** from $GF(4)$ to $GF(2)$ $\rightsquigarrow$ **binary** linear code of length and dimension **twice** as large

▷ **Masking** the structure like McEliece

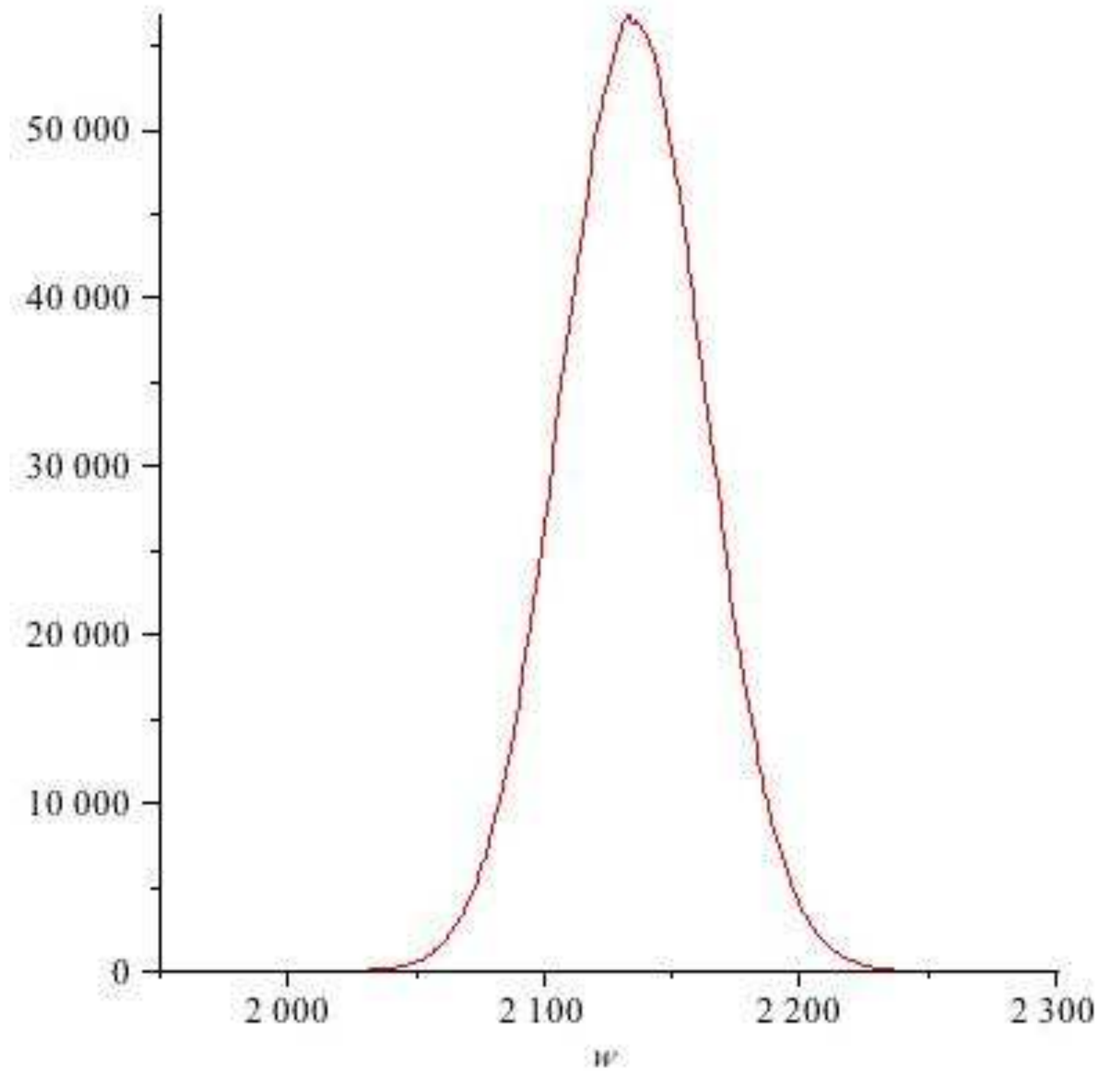
Estimating Minimum Distance

Proposition. Minimum distance of a polar code with information set containing only integers whose binary representation does not contains less than ℓ zeros is at least 2^ℓ.

▷ **Proposed parameters** (over $GF(4)$)

- $N = 4,096$, $K = 1,255$, $\ell = 7 \rightsquigarrow$ minimum distance $\geqslant 128$
- 80-bit security (Peters' q-ary version of ISD)

Binary Distorsion Values ($4,000,000$ **tests**)



Maximum distorsion $\leqslant 2,268$

Performances

- Binary code of length $8,182$ and dimension $2,510$
- **Maximum** distorsion $\leqslant 2,268 \rightsquigarrow$ 1400-bit security (ISD for binary codes)
- Average time for one signature: $\simeq$ 4ms
- Key size: 6.5 Mbyte