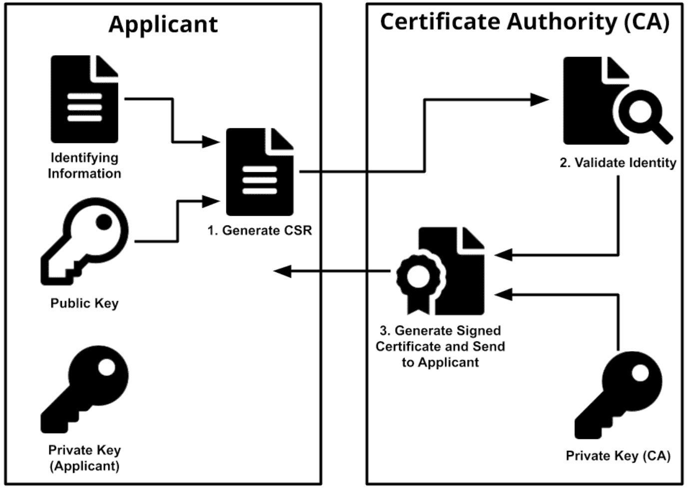
КРИПТОПРО

Towards a Proof-of-Possession мechanism For binary Goppa code-based KEM

Evgeny Alekseev
Aleksandra Babueva
Andrey Bozhko
Liliya Akhmetzyanova

Motivation

Certificate Issuance in Public Key Infrastructure (PKI)

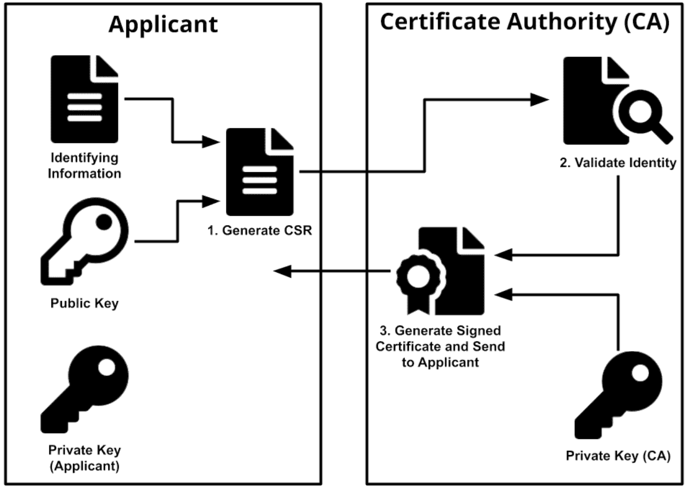


CSR (Certificate Signing Request):

- Public key pk
- Identifying info (attributes) $attr$

Motivation

Certificate Issuance in Public Key Infrastructure (PKI)



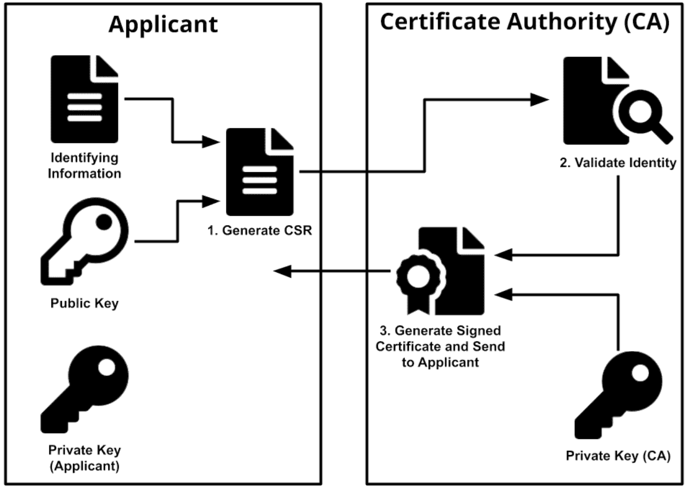
CSR (Certificate **Signing** Request):

- Public key pk
- Identifying info (attributes) $attr$
- Signature for pk and $attr$ using private key sk (**what for?**)

КРИПТОПРО

Motivation

Certificate Issuance in Public Key Infrastructure (PKI)



CSR (Certificate **Signing** Request):

- Public key pk
- Identifying info (attributes) $attr$
- Signature for pk and $attr$ using private key sk (**what for?**)

for **proof of possession** of private key as additional protection in protocols:

[1] Asokan N., Niemi V., Laitinen P. On the usefulness of proof-of-possession. 2nd Annual PKI Research Workshop – Pre-Proceedings, 2003.

[2] Алексеев Е., Зинюк Б. Об одной проблеме при выдаче сертификатов открытых ключей постквантовых алгоритмов инкапсуляции ключа. PKI-форум, 2023.

Motivation

(public key pk, private key sk):

→ signature key pair: no problem to sign CSR for proving

→ encryption key pair:

Motivation

(public key pk, private key sk):

→ signature key pair: no problem to sign CSR for proving

→ encryption key pair:

✓ Dlog-based cryptography

Same key pair in DH-based encryption and ElGamal-type signature

✓ Lattice-based cryptography

Several specific methods were proposed [3]

X **Code-based cryptography**

(?) Only generic methods on zk-nark

КРИПТОПРО

[3] Guneysu T. et al. Proof-of-possession for KEM certificates using verifiable generation.

Proof-of-Possession for KEM

Key Encapsulation Mechanism (KEM)

- $\mathsf{KGen}() \rightarrow (\mathsf{sk}, \mathsf{pk})$
- $\mathsf{Encaps}(\mathsf{pk}) \rightarrow (K, c)$
- $\mathsf{Decaps}(\mathsf{sk}, c) \rightarrow K$

$$\mathbf{Exp}^{\mathrm{IND-CCA2}}_{\mathsf{KEM}}(\mathcal{A})$$

$$
\begin{aligned}
&(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KEM.KGen}(\ )\\
&st \leftarrow \mathcal{A}^{\mathsf{Decaps}(\mathsf{sk},\cdot)}(\mathsf{pk})\\
&b \xleftarrow{\mathcal{U}} \{0,1\}\\
&(K_0^*, c^*) \leftarrow \mathsf{KEM.Encaps}(\mathsf{pk})\\
&K_1^* \xleftarrow{\mathcal{U}} \mathcal{K}\\
&b' \leftarrow \mathcal{A}^{\mathsf{Decaps}(\mathsf{sk},\cdot)}(st, c^*, K_b^*)\\
&\mathbf{return}\ (b' = b)
\end{aligned}
$$

Target security property: **Indistinguishability** (IND-CCA2): only owner of sk can obtain K

Proof-of-Possession for KEM

Proof-of-Possession (PoP)

- $\mathsf{PGen}(\mathsf{sk}, \mathsf{pk}, \mathsf{attrs}) \to \pi$
- $\mathsf{Vf}(\mathsf{pk}, \mathsf{attrs}, \pi) \to b \in \{0, 1\}$

$$\mathbf{Exp}^{\mathrm{UF}}_{\mathsf{PoP},\mathsf{KEM}}(\mathcal{A})$$

$$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KEM.KGen}(\,)$$

$$(\mathsf{attrs}^*, \pi^*) \leftarrow \mathcal{A}^{\mathsf{PGen}(\mathsf{sk},\mathsf{pk},\cdot)\,\boxed{\mathsf{Decaps}(\mathsf{sk},\cdot)}}(\mathsf{pk})$$

$$\mathbf{return}\ \mathsf{PoP.Vf}(\mathsf{pk}, \mathsf{attrs}^*, \pi^*) \wedge ((\mathsf{attrs}, \pi) \neq (\mathsf{attrs}^*, \pi^*))$$

Target security properties: **Unforgeabilty** (UF-CMA): only owner of sk can provide valid proof π must hold in the presence of Decaps oracle

КРИПТОПРО

Proof-of-Possession for KEM

Proof-of-Possession (PoP)

- $\mathsf{PGen}(\mathsf{sk}, \mathsf{pk}, \mathsf{attrs}) \to \pi$
- $\mathsf{Vf}(\mathsf{pk}, \mathsf{attrs}, \pi) \to b \in \{0, 1\}$

$$\mathbf{Exp}^{\mathrm{ZK}}_{\mathsf{PoP},\mathsf{Sim}}(\mathcal{A})$$

$$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathcal{A}(\ )$$

$$b \xleftarrow{\mathcal{U}} \{0, 1\}$$

$$\mathbf{if}\ b = 1 : b' \leftarrow \mathcal{A}^{\mathsf{PGen}(\mathsf{sk},\mathsf{pk},\cdot),\mathsf{Hash}(\cdot)}()$$

$$\mathbf{if}\ b = 0 : b' \leftarrow \mathcal{A}^{\mathsf{Sim}(\mathsf{pk},\cdot),\mathsf{Hash}(\cdot)}()$$

$$\mathbf{return}\ b = b'$$

Target security properties:

Unforgeabilty (UF-CMA): only owner of sk can provide valid proof π must hold in the presence of Decaps oracle

Zero Knowledge (ZK): proof π doesn't leak info about sk

required for preserving IND-CCA2 security of KEM

Binary Goppa code-based KEM

«Classical» approach for constructing KEM:

Public Key Encryption scheme (PKE) + Fujisaki-Okamoto(FO)-transformation

- $\mathsf{KGen}() \to (\mathsf{sk}, \mathsf{pk})$
- $\mathsf{Enc}(\mathsf{pk}, m) \to c$
- $\mathsf{Dec}(\mathsf{sk}, c) \to m$

FO = T + U

- T makes PKE «rigid»
- U makes KEM from «rigid» PKE
  - implicit/explicit rejection
  - K depends on c or not

Rigidity: $\forall\ (\mathsf{sk}, \mathsf{pk}), c : \mathsf{Dec}(\mathsf{sk}, c) = \perp \text{ or } \mathsf{Enc}(\mathsf{pk}, \mathsf{Dec}(\mathsf{sk}, c)) = c$

Binary Goppa code-based KEM

Our «patient»:

Public Key Encryption scheme (PKE) + Fujisaki-Okamoto(FO)-transformation

Niederreiter scheme
based on binary Goppa code

FO = U

Note: Niederreiter scheme is already «rigid»

- used in Classic McEliece KEM [4]
- similar scheme used in «Кодиеум» (TC26)

[4] Classic McEliece Team. Classic mceliece: cryptosystem specification, 2022.

Binary Goppa code-based KEM

Binary Goppa code

Fix the following parameters:

- $m, n \in \mathbb{N}, n \leq 2^m$,
- $\alpha = (\alpha_0, \dots, \alpha_{n-1}) \subseteq \mathbb{F}_{2^m}$, α_i are distinct,
- $g(x) \in \mathbb{F}_{2^m}[x],\ \deg g(x) = t$ s.t. $g(\alpha_i) \neq 0\ \forall\ i$

Binary Goppa code C of length n is

$$C = \Gamma(\alpha, g) = \left\{ c = (c_0, \dots, c_{n-1}) \in \mathbb{F}_2^n : \sum_{i=1}^{n} \frac{c_i}{x - \alpha_i} = 0 \bmod g(x) \right\}$$

If $g(x)$ square-free then minimal distance $d \geq 2t + 1$

Binary Goppa code-based KEM

Binary Goppa code

Fix the following parameters:

- $m, n \in \mathbb{N}, n \leq 2^m$
- $\alpha = (\alpha_0, \dots, \alpha_{n-1}) \subseteq \mathbb{F}_{2^m}, \alpha_i$ are distinct,
- $g(x) \in \mathbb{F}_{2^m}[x],\ \deg g(x) = t$ s.t. $g(\alpha_i) \neq 0\ \forall\ i$

We consider

- $n = 2^m$
- $g(x)$ is irreducible

Binary Goppa code C of length n is

$$C = \Gamma(\alpha, g) = \left\{ c = (c_0, \dots, c_{n-1}) \in \mathbb{F}_2^n : \sum_{i=1}^{n} \frac{c_i}{x - \alpha_i} = 0 \bmod g(x) \right\}$$

If $g(x)$ square-free then minimal distance $d \geq 2t + 1$

КРИПТОПРО

Binary Goppa code-based KEM

ND.KGen( ):

1 : $\alpha_0, \dots, \alpha_{n-1} \xleftarrow{\mathcal{U}} \mathbb{F}_{2^m} : \alpha_i \neq \alpha_j, i \neq j$

2 : $\alpha \leftarrow (\alpha_0, \dots, \alpha_{n-1})$

3 : $g \xleftarrow{\mathcal{U}} \{f \in \mathbb{F}_{2^m}[x] : \deg(f) = t\}$

4 : **if** `IsIrreducible`$(g) = 0$: go to 3

5 : Compute $\widetilde{H} = \{h_{ij}\} \in \mathbb{F}_{2^m}^{t \times n} : \; h_{ij} \leftarrow \alpha_j^{i-1}/g(\alpha_j)$

6 : Compute $\widehat{H} \in \mathbb{F}_2^{mt \times n}$, replacing each $h_{ij} = h_{ij}^0 + \dots + h_{ij}^{m-1} z^{m-1}$ with column $(h_{ij}^0, \dots, h_{ij}^{m-1})^T \in \mathbb{F}_2^m$

7 : $[I_{n-k} \mid T] \leftarrow \mathsf{Systematic}(\widehat{H})$. If error go to 1

8 : **return** $((g, \alpha), T)$

Rejection Sampling:
generate random g until g is irreducible
Probability of success $\approx 1/t$ [4]

Reduce to systematic form

КРИПТОПРО

[5] Ben-Or M., Probabilistic algorithms in finite fields

Binary Goppa code-based KEM

$\mathsf{ND.Enc}(\mathsf{pk}, e)$:

$T \leftarrow \mathsf{pk}$
$c \leftarrow [E_{n-k} \mid T]e$
$\mathbf{return}\ c$

c is a syndrome for $e, wt(e) = t$

$\mathsf{ND.Dec}(\mathsf{sk}, c)$:

$v \leftarrow c \| 0^k$
$c \leftarrow \mathtt{decode}(\mathsf{sk}, v)$
$\mathbf{if}\ c = \perp : \mathbf{return}\ \perp$
$e \leftarrow v + c$
$\mathbf{if}\ \boxed{\mathrm{wt}(e) = t \wedge c = [E_{n-k} \mid T]e :}$
$\quad \mathbf{return}\ e$
$\mathbf{return}\ \perp$

Syndrome decoding:

- Paterson's algorithm
- Berlekamp-Massey algorithm

makes PKE rigid

PoP: Naive approach

Use CFS [6] signature scheme (hash-and-sign paradigm):

- h = H(**attr**) $\in \mathbb{F}_2^n$
- signature = decode(h) using α, g

Problem:

The probability that h is decodable is $\approx \dfrac{1}{t!}$

Example: for Classic McEliece-128 $(t = 64)$: $\approx 2^{-296}$

Hash-and-sign paradigm doesn't work here

КРИПТОПРО

[6] Courtois N. T., Finiasz M., Sendrier N. How to achieve a McEliece-based digital signature scheme

PoP: Naive approach

<u>Another techniques for signatures</u>

Interactive ZKP + Fiat-Shamir transformation:

- Random permutations (Stern approach [7])
- MPC-in-the-head [8]

[7] V. V. Vysotskaya, I. V. Chizhov. The security of the code-based signature scheme based on the Stern identification protocol.
.
[8] Feneuil T., Joux A., and Rivain M. Syndrome Decoding in the Head: Shorter Signatures from Zero-Knowledge Proofs

Core idea

sk = $s \in \mathbb{F}_2^n, wt(s) \leq w$
pk = $y \in \mathbb{F}_2^{n-k}$ and $H \in \mathbb{F}_2^{(n-k)\times n}$

given y and H

prove the knowledge of

$s \; s.t. Hs = y, wt(s) \leq w$

without revealing s

КРИПТОПРО

PoP: Naive approach

We can’t just add
s to sk $= (g(x), \alpha)$ and $y = [I_{n-k}\,|T]s$ to pk $= T$

Such a signature proves nothing
Anyone can compute it without knowing $g(x), \alpha$

PoP: Naive approach

BUT: if s is a codeword of minimal weight and $y = 0$
it potentially can prove something…

<u>Problem</u>: even knowing $g(x), \alpha$ it is hard to find
codeword of minimal weight

PoP: A new approach

1. Change KGen: for a random word of small weight generate a Goppa code such that the random word is a codeword of the generated code

Input: $\alpha, c \in \mathbb{F}_2^n,\ wt(c) = 2t + 1$
Output: $g(x)$ of degree t s.t $c \in \Gamma(\alpha, g)$

$\texttt{PolyFromWord}(t, c, \alpha)$

1 : $(\alpha_0, \ldots, \alpha_{n-1}) \leftarrow \alpha$

2 : $\delta \leftarrow \prod_{i:c_i=1} (x - \alpha_i)$

3 : $\delta' \leftarrow derivative(\delta)$

4 : $\beta^2 \leftarrow \delta'$

5 : **return** β

A new approach

1. Change KGen: for a random word of small weight generate a Goppa code such that the random word is a codeword of the generated code

Proposition

Fix

- $\alpha = (\alpha_0, \dots, \alpha_{n-1}) \subseteq \mathbb{F}_{2^m}, \alpha_i$ are distinct, $n = 2^m$,
- $c \in \mathbb{F}_2^n, wt(c) = 2t + 1$

If g = $\texttt{PolyFromWord}(t, c, \alpha)$ is irreducible, then $c \in \Gamma(\alpha, g)$

Follows from the fact, that $g(x)$ should divide derivative of «locator poly» of any codeword

КРИПТОПРО

A new approach

Changes in KEM:

ND.KGen′( ):

1 : $\alpha_0, \ldots, \alpha_{n-1} \xleftarrow{\mathcal{U}} \mathbb{F}_{2^m} : \alpha_i \neq \alpha_j, i \neq j$

2 : $\alpha \leftarrow (\alpha_0, \ldots, \alpha_{n-1})$

3 : $c \xleftarrow{\mathcal{U}} \mathbb{F}_2^n : \mathrm{wt}(c) = 2t + 1$

4 : $g \leftarrow \mathtt{PolyFromWord}(t, c, \alpha)$

changes №1

5 : **if** $\mathtt{IsIrreducible}(g) = 0$: go to 3

6 : Compute $\widetilde{H} = \{h_{ij}\} \in \mathbb{F}_{2^m}^{t \times n} : \ h_{ij} \leftarrow \alpha_j^{i-1}/g(\alpha_j)$

7 : Compute $\widehat{H} \in \mathbb{F}_2^{mt \times n}$, replacing each $h_{ij} = h_{ij}^0 + \cdots + h_{ij}^{m-1} z^{m-1}$ with column $(h_{ij}^0, \ldots, h_{ij}^{m-1})^T \in \mathbb{F}_2^m$

8 : $[I_{n-k} \mid T] \leftarrow \mathsf{Systematic}(\widehat{H})$. If error go to 1

9 : **return** $((g, \alpha, c), T)$

changes №2

The (experimental) probability of picking

- irreducible poly is $\approx \frac{1}{t}$

A new approach

2. Prove the knowledge of this word (with attributes)

given $y \in \mathbb{F}_2^{mt}$ and $H \in \mathbb{F}_2^{mt \times n}$

prove the knowledge of

$s \in \mathbb{F}_2^n \; s.t. Hs = y, wt(s) \le w$

without revealing s

In our case:

$y = 0^{mt}, H = [I_{n-k} \mid T\,]$ from ND.KGen′( ):

$s = c,\ w = 2t + 1$

Security analysis

Theorem 8 [3] (Informal). Let **H** be modeled as a random oracle. PoP for KEM provides unforgeability if the following conditions hold:

- $\mathsf{PoP.PGen}$ is zero-knowledge
- $\mathsf{PoP.Vf}$ is extractable
- $\mathsf{KEM.Decaps}$ is simulatable
- $\mathsf{KEM.KGen}$ is one-way function

КРИПТОПРО

[3] Guneysu T. et al. Proof-of-possession for KEM certificates using verifiable generation.

Security analysis

Theorem 8 [3] (Informal). Let **H** be modeled as a random oracle. PoP for KEM provides unforgeability if the following conditions hold:

- $\mathsf{PoP.PGen}$ is zero-knowledge
- $\mathsf{PoP.Vf}$ is extractable
- $\mathsf{KEM.Decaps}$ is simulatable
- $\mathsf{KEM.KGen}$ is one-way function

usually doesn't depend on the **code properties**
depend on well-studied security of
commitment/hash/PRG

КРИПТОПРО

Security analysis

Theorem 8 [3] (Informal). Let **H** be modeled as a random oracle. PoP for KEM provides unforgeability if the following conditions hold:

- $\mathsf{PoP.PGen}$ is zero-knowledge
- $\mathsf{PoP.Vf}$ is extractable
- $\mathsf{KEM.Decaps}$ is simulatable

usually doesn't depend on the **code properties**
depend on well-studied security of commitment/hash/PRG

- $\mathsf{KEM.KGen}$ is one-way function

necessary condition for KEM security:
if we «believe» that KEM is IND-CCA2-secure that this property **must already hold**

КРИПТОПРО

Security analysis

Theorem 8 [3] (Informal). Let **H** be modeled as a random oracle. PoP for KEM provides unforgeability if the following conditions hold:

- $\mathsf{PoP.PGen}$ is zero-knowledge
- $\mathsf{PoP.Vf}$ is extractable
- $\mathsf{KEM.Decaps}$ is simulatable
- $\mathsf{KEM.KGen}$ is one-way function

usually doesn't depend on the **code properties**
depend on well-studied security of commitment/hash/PRG

BUT: $\mathsf{KEM.KGen}$ is changed

- Original: compute $g(x), \alpha$ from T
- Modified: compute c, α from T

КРИПТОПРО

Security analysis

Hypothesis

Let

- $G = \{g(x) \in \mathbb{F}_{2^m}[x] \mid \deg g(x) = t, g(x) - irreducible\};$
- $\Omega = \{c \in \mathbb{F}_2^n \mid wt(c) = 2t + 1\};$
- $(\Omega, 2^\Omega, U)$ – probability space with uniform distribution.

Distribution of the random variable $\mathtt{PolyFromWord}(t, \cdot, \alpha) : \Omega \to \mathbb{F}_{2^m}[x]$ conditioned on G is computationally indistinguishable from uniform distribution over G.

Experimentally verified on small parameters

- $m = 5, n = 2^m, t = 3$
- $m = 6, n = 2^m, t = 3$

КРИПТОПРО

Security analysis

Reusing known signature schemes [7,8] and its analysis

Signature scheme (SS)

- $\mathsf{KGen}(\ ) \rightarrow (\mathsf{sk}, \mathsf{pk})$
- $\mathsf{Sig}(\mathsf{sk}, m) \rightarrow \sigma$
- $\mathsf{Vf}(\mathsf{pk}, m, \sigma) \rightarrow b \in \{0, 1\}$

Signature scheme as a PoP mechanism:

$$\mathsf{PoP.PGen}(\mathsf{sk}, \mathsf{pk}, \mathsf{attrs}) := \mathsf{SS.Sig}(\mathsf{sk}, \mathsf{attrs})$$

$$\mathsf{PoP.Vf}(\mathsf{pk}, \mathsf{attrs}, \pi) := \mathsf{SS.Vf}(\mathsf{pk}, \mathsf{attrs}, \pi)$$

КРИПТОПРО

Security analysis

Reusing known signature schemes [7,8] and its analysis

Alternative proof way (Informal). Let **H** be modeled as a random oracle. PoP for KEM provides unforgeability if the following conditions hold:

- $\mathsf{SS.KGen}(\,) = \mathsf{KEM.KGen}(\,)$
- SS provides unforgeability
- $\mathsf{KEM.Decaps}$ is simulatable

КРИПТОПРО

Security analysis

<u>Reusing known signature schemes [7,8] and its analysis</u>

! Signature schemes from [7, 8] use random linear code, need to change to code from ND

KGen( ):

$1: \quad H \xleftarrow{\mathcal{U}} \mathbb{F}_2^{(n-k)\times n}$

$2: \quad s \xleftarrow{\mathcal{U}} \mathbb{F}_2^n$

$3: \quad y \leftarrow Hs$

$4: \quad \textbf{return } (s, (H, y))$

Syndrome Decoding (SD) for random linear code

KGen′( ):

$1: \quad \alpha_0, \ldots, \alpha_{n-1} \xleftarrow{\mathcal{U}} \mathbb{F}_{2^m} : \alpha_i \neq \alpha_j, i \neq j$

$2: \quad \alpha \leftarrow (\alpha_0, \ldots, \alpha_{n-1})$

$3: \quad c \xleftarrow{\mathcal{U}} \mathbb{F}_2^n : \mathrm{wt}(c) = 2t + 1$

$4: \quad g \leftarrow \mathtt{PolyFromWord}(t, c, \alpha)$

$5: \quad \textbf{if } \mathtt{IsIrreducible}(g) = 0 : \text{go to } 3$

$6: \quad \text{Compute } \widetilde{H} = \{h_{ij}\} \in \mathbb{F}_{2^m}^{t\times n} : \ h_{ij} \leftarrow \alpha_j^{i-1}/g(\alpha_j)$

$7: \quad \text{Compute } \widehat{H} \in \mathbb{F}_2^{mt\times n}, \text{ replacing each } h_{ij} = h_{ij}^0 + \cdots + h_{ij}^{m-1} z^{m-1}$ with column $(h_{ij}^0, \ldots, h_{ij}^{m-1})^T \in \mathbb{F}_2^m$

$8: \quad [I_{n-k} \mid T] \leftarrow \mathsf{Systematic}(\widehat{H}). \text{ If error go to } 1$

$9: \quad \textbf{return } (c, (T, 0^{mt}))$

Find c, α for random irreducible Goppa code

КРИПТОПРО

Security analysis

Reusing known signature schemes [7,8] and its analysis

But! Usually unforgeability analysis [7,8] consists of the similar steps proving that

- $\mathsf{SS.Sig}$ is zero-knowledge (ZK)
- $\mathsf{SS.Vf}$ is extractable (Ext)
- $\mathsf{SS.KGen}$ is one-way function (OW)

Security analysis

Reusing known signature schemes [7,8] and its analysis

But! Usually unforgeability analysis [7,8] consists of the similar steps proving that

- $\mathsf{SS.Sig}$ is zero-knowledge (ZK)
- $\mathsf{SS.Vf}$ is extractable (Ext)
- $\mathsf{SS.KGen}$ is one-way function (OW) ← changed

Security analysis

Example from [8]

Theorem 5. *Suppose the PRG used is (t, ε_{PRG})-secure and any adversary running in time t has at most an advantage ε_{SD} against the underlying d-split syndrome decoding problem. Model* Hash_0, Hash_1 *and* Hash_2 *as random oracles where* Hash_0, Hash_1 *and* Hash_2 *have 2λ-bit output length. Then chosen-message adversary against the signature scheme depicted in Figure 1, running in time t, making q_s signing queries, and making q_0, q_1, q_2 queries, respectively, to the random oracles, succeeds in outputting a valid forgery with probability*

$$\Pr[\mathsf{Forge}] \leq \underbrace{\frac{(q_0 + \tau N q_s)^2}{2 \cdot 2^{2\lambda}} + \frac{q_s(q_s + q_0 + q_1 + q_2)}{2^{2\lambda}} + q_s \cdot \tau \cdot \varepsilon_{PRG}}_{\text{ZK}} + \underbrace{\varepsilon_{SD}}_{\text{OW}} + \underbrace{q_2 \cdot \varepsilon^{\tau}}_{\text{Ext}},$$

where $\varepsilon = p + \frac{1}{N} - p \cdot \frac{1}{N}$ and p defined in Equation (3).

SD for random code → find (c, α) for Goppa code

What should be done

For security analysis of proposed PoP in ROM:

- Analyze hypothesis

Thank you for your attention!

lah@cryptopro.ru

Security analysis

Theorem 8 [3] (Informal). Let **H** be modeled as a random oracle. PoP for KEM provides unforgeability if the following conditions hold:

- $\mathsf{PoP.PGen}$ is zero-knowledge → ∃ «simulator» computing correct proof without sk (in RO)
- $\mathsf{PoP.Vf}$ is extractable → ∃ «extractor» computing sk from correct proof
- $\mathsf{KEM.Decaps}$ is «simulatable» → ∃ «simulator» computing Decaps without sk (in RO)
- $\mathsf{KEM.KGen}$ is one-way function → computing sk from pk is an intractable task

КРИПТОПРО

[3] Guneysu T. et al. Proof-of-possession for KEM certificates using verifiable generation.

Security analysis

<u>Example</u>: zero-knowledge for the Stern-based signature scheme

Theorem 3 [6] about
connection between EUF-CMA (chosen message attack) and EUF-NMA (no message attack)
security proves that Sig algorithm is zero-knowledge.

Theorem 3. Let $\mathcal{A}$ be an adversary in the EUF-CMA model for the Stern signature scheme making at most q_f queries to the hashing oracle F and at most q_s queries to the signing oracle Sign. Then there exists an adversary $\mathcal{B}$ in the EUF-NMA model for the Stern signature scheme making at most q_f queries to the hashing oracle and

$$\mathsf{Adv}^{\text{EUF-NMA}}_{\text{Stern}}(\mathcal{B}) \geqslant \mathsf{Adv}^{\text{EUF-CMA}}_{\text{Stern}}(\mathcal{A}) - q_s \left(\frac{14\,\tilde{c}\,\delta\,q_f}{T_{\text{Coll}}}\right)^{\delta},$$

where T_{Coll} is the complexity of optimal algorithm solving $\text{Coll}(h)$ problem with probability of success at least $1 - 1/e$ and $\tilde{c}$ is a constant depending on the model of computation.

Furthermore, if the complexity of $\mathcal{A}$ is T, then the complexity of $\mathcal{B}$ is upper bounded by $T + c''(q_f + q_s T^{\text{Sig}}_{\text{Stern}})$, where $T^{\text{Sig}}_{\text{Stern}}$ is the complexity of the signature generation algorithm and c'' is a constant depending on the model of computation.

doesn't depend on code properties

[6] V. V. Vysotskaya, I. V. Chizhov. The security of the code-based signature scheme based on the Stern identification protocol.

Binary Goppa code-based KEM

FO-transformation U with implicit rejection and dependence on c

KEM.KGen

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{PKE.KGen}(\,)$

$s \xleftarrow{\mathcal{U}} \mathcal{M}$

$\mathsf{sk}' \leftarrow (\mathsf{sk}, s)$

$\mathbf{return}\ (\mathsf{sk}', \mathsf{pk})$

KEM.Encaps(pk)

$m \xleftarrow{\mathcal{U}} \mathcal{M}$

$c \leftarrow \mathsf{PKE.Enc}(\mathsf{pk}, m)$

$K \leftarrow \mathsf{H}(m, c)$

$\mathbf{return}\ (K, c)$

KEM.Decaps(sk′, c)

$(\mathsf{sk}, s) \leftarrow \mathsf{sk}'$

$m' \leftarrow \mathsf{PKE.Dec}(\mathsf{sk}, c)$

$\mathbf{if}\ m' \neq \perp:$

$\quad \mathbf{return}\ \mathsf{H}(m', c)$

$\mathbf{else}\ :$

$\quad \mathbf{return}\ \mathsf{H}(s, c)$

Straight-line Extractability

ε-extractability (information-theoretic)

Hash is a random oracle modeling H. There exists efficient algorithm Ext (extractor) modeling Hash such that for any adversary A

$$\mathsf{Adv}^{\mathrm{EXT}}_{\mathsf{PoP},\mathsf{Ext}}(\mathcal{A}) = \Pr\left[\mathbf{Exp}^{\mathrm{EXT}}_{\mathsf{PoP},\mathsf{Ext}}(\mathcal{A}) \to 1\right] < \varepsilon$$

$$\begin{array}{l}
\underline{\mathbf{Exp}^{\mathrm{EXT}}_{\mathsf{PoP},\mathsf{Ext}}(\mathcal{A})} \\
(\mathsf{pk}, \pi, \mathsf{attrs}) \leftarrow \mathcal{A}^{\mathsf{Hash}(\cdot)}() \\
\textbf{if } \mathsf{Vf}(\mathsf{pk}, \mathsf{attrs}, \pi) \neq 1 \\
\quad \textbf{return } 0 \\
\mathsf{sk} \leftarrow \mathsf{Ext}(\mathsf{pk}, \mathsf{attrs}, \pi) \\
\textbf{return } (\mathcal{R}(\mathsf{pk}, \mathsf{sk}) \neq 1)
\end{array}$$

Decapsulation Sumulatibility

ε-simulatability (information-theoretic)

Hash is a random oracle modeling H. There exists efficient algorithm Sim (simulator) modeling Hash such that for any adversary A

$$\mathsf{Adv}^{\mathrm{KEM-SIM}}_{\mathsf{KEM},\mathsf{Sim}}(\mathcal{A}) = 2\Pr\left[\mathbf{Exp}^{\mathrm{KEM-SIM}}_{\mathsf{KEM},\mathsf{Sim}}(\mathcal{A}) \to 1\right] - 1 < \varepsilon$$

$$\underline{\mathbf{Exp}^{\mathrm{KEM-SIM}}_{\mathsf{KEM},\mathsf{Sim}}(\mathcal{A})}$$

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KGen}()$

$b \xleftarrow{\mathcal{U}} \{0,1\}$

if $(b = 0)$

$\quad b' \leftarrow \mathcal{A}^{\mathsf{Decaps}(\mathsf{sk},\cdot),\mathsf{Hash}(\cdot)}$

else

$\quad b' \leftarrow \mathcal{A}^{\mathsf{Sim}(\mathsf{pk},\cdot),\mathsf{Hash}(\cdot)}$

return $(b = b')$