NTNU | Norwegian University of Science and Technology

VERIFIABLE RANDOM SECRETS AND SUBLIMINAL-FREE DIGITAL SIGNATURES

Master's thesis. Tjerand Aga Silde, August 2020

ABSTRACT

Contribution

We present the first post-quantum secure subliminal-free digital signature schemes. The first scheme is based purely on lattices, while the second scheme is based on collision-resistant hash-functions combined with any post-quantum "hash-then-sign" signature scheme.

ABSTRACT

- The concrete instantiation of the purely lattice-based scheme can be made non-interactive and it takes less than 10 seconds[†] to create a subliminal-free signature of total size $\approx$ 12.65 MB[‡].
- The concrete instantiation of the hash-based scheme combined with lattice-based signatures is interactive and it takes $\approx$ 1 second to generate a subliminal-free signature of size 3.3 KB, where a malicious signer has probability 2^{-10} to embed subliminal information into the signature.

[†] now only $\approx$ 5 seconds due to new optimizations
[‡] improved from $\approx$ 50 MB in Herman Galteland's Ph.D. thesis

PREFACE

- Sections §1, §4 and §5 are co-authored with Herman Galteland.
- Sections §2 and §3 are background material, where the shuffle-protocol in §3.2 is joint work with Diego, Carsten, Kristian and Thor.
- Section §6 is my own contribution‡. We conclude in §7.
- Sections §4, §5 and §6 are the main new contributions in this work.

‡new and improved compared to work published in Herman Galteland's Ph.D. thesis

OUTLINE

Introduction

Preliminaries

Lattice-Based Cryptography

Verifiable Random Secrets

Subliminal-Free Digital Signatures

Our Schemes

Conclusion

Introduction

Imagine an authentication without secrecy communication channel with a sender $\mathtt{S}$, a warden $\mathtt{W}$, a recipient $\mathtt{R}$ and a message-signature pair (m, σ):

$$\mathtt{S} \quad \xrightarrow{(m,\sigma)} \quad \mathtt{W} \quad \xrightarrow{(m,\sigma)} \quad \mathtt{R}$$

Then $\mathtt{S}$ and $\mathtt{R}$ can communicate covertly by embedding secret information into the signature, e.g., if $\mathtt{S}$ and $\mathtt{R}$ have some key-material that is shared in advance.

Introduction

Example: Schnorr-signatures

Public parameters (g, G), signature keys $(a, x = g^a)$, hash-function $\mathtt{H}$, symmetric key system $(\mathcal{E}, \mathcal{D})$ and symmetric key k. Assume that (a, k) is shared between $\mathtt{S}$ and $\mathtt{R}$. Then $\mathtt{S}$ can send a subliminal message $\hat{m}$ to $\mathtt{R}$ without $\mathtt{W}$ noticing:

$$\mathtt{S}: \quad r = \mathcal{E}(k, \hat{m}), \quad \alpha = g^r, \quad \beta = \mathtt{H}(\alpha || m), \quad \gamma = r + \beta a, \quad \sigma = (\alpha, \gamma).$$

$$\mathtt{W}: \quad \beta = \mathtt{H}(\alpha || m), \quad g^\gamma \stackrel{?}{=} \alpha x^\beta, \quad \text{if yes: forward } (m, \sigma) \text{ to } \mathtt{R}.$$

$$\mathtt{R}: \quad \beta = \mathtt{H}(\alpha || m), \quad g^\gamma \stackrel{?}{=} \alpha x^\beta, \quad \text{if yes: compute} \quad r = \gamma - \beta a, \quad \hat{m} = \mathcal{D}(k, r).$$

Introduction

To prevent such an subliminal channel, we need a procedure for creating verifiable random values that is not controlled by $\mathbb{S}$, but also hides the values from others: a *verifiable random secrets* (VRS) scheme. We combine the VRS with a signature scheme to achieve a subliminal-free signature (SFS) scheme.

There exists several SFS constructions for signatures based on the hardness of discrete logarithms, and we propose the two first post-quantum SFS schemes.

Preliminaries

- Working over the ring $R_p = \mathbb{Z}_p[X]/\langle X^N + 1\rangle$ for prime p and power-of-two N.
- The k-SUM problem is to find a subset of size k out of a set of n values $a_1, a_2, \ldots, a_n$ that sums to a given target s. The decisional and search variants are equivalent, and k-SUM takes $\mathcal{O}(n^{k/2})$ operations to solve.
- We use both randomized and deterministic discrete Gaussian sampling.

Lattice-Based Cryptography

Commitment Scheme

- `KeyGen`, outputs $\boldsymbol{A} = \begin{bmatrix} 1 & a_1 & a_2 \\ 0 & 1 & a_3 \end{bmatrix}$, where $a_1, a_2, a_3 \xleftarrow{\$} R_p$,
- `Com`, on input $m \in R_p$ and $\boldsymbol{r} \in R_p^3$ where $||\boldsymbol{r}||_\infty = 1$, computes $\boldsymbol{c} = \boldsymbol{A} \cdot \boldsymbol{r} + \begin{bmatrix} 0 \\ m \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$, and returns $\boldsymbol{c}$ and $\boldsymbol{d} = (m, \boldsymbol{r}, 1)$,
- `Open`, on input $\boldsymbol{c}$ and $(m, \boldsymbol{r}, f)$, verifies the opening by checking if $f \cdot \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} \stackrel{?}{=} \boldsymbol{A} \cdot \boldsymbol{r} + f \cdot \begin{bmatrix} 0 \\ m \end{bmatrix}$, and that $||r_i|| \leq 4\sigma\sqrt{N}$.

Zero-Knowledge Proof of Linear Relations

Let $[x_1], [x_2]$ and $[x_3]$ be commitments such that $x_3 = \alpha_1 x_1 + \alpha_2 x_2$ for some public values $\alpha_1, \alpha_2 \in R_p$. Then Π_{Lin} produces a zero-knowledge proof of knowledge of this relation, and Π_{LinV} verifies the proof.

Lattice-Based Cryptography

Zero-Knowledge Proof of Correct Shuffle

Given a list of elements $\hat{M}_1, \hat{M}_2, \ldots, \hat{M}_\tau$ from R_p and commitments $[M]_1, [M]_2, \ldots, [M]_\tau$, we can prove that the $[M]_i$'s are commitments to the $\hat{M}_{\gamma(i)}$'s, for some secret permutation γ. Then Π_{Shuffle} produces a zero-knowledge proof of knowledge of this relation, and Π_{ShuffleV} verifies the proof.

Verifiable Random Secrets

Definition

- $\texttt{Setup}$, on input security parameter 1^λ, outputs public parameters $\texttt{sp}$,
- $\Pi_{\texttt{Seed}}$, on input $\texttt{sp}$, outputs a random seed s,
- $\texttt{Com}$, on input seed s, outputs commitment $\tilde{c}$ of s and opening $\tilde{d}$,
- $\texttt{Challenge}$, on no input, outputs a random challenge t,
- $\texttt{Generate}$, on input commitment $\tilde{c}$, opening $\tilde{d}$ and challenge t, outputs commitment c, opening d of c (containing $r = r(s, t)$) and proof π,
- $\texttt{Check}$, on input $\tilde{c}$ and c, challenge t, and proof π, outputs 0 or 1,

Verifiable Random Secrets

VRS

Prover		Verifier
	$\mathtt{sp} \leftarrow \mathtt{Setup}(1^\lambda)$	
	$s \leftarrow \Pi_{\mathtt{Seed}}(\mathtt{sp})$	
$(\tilde{c}, \tilde{d}) \leftarrow \mathtt{Com}(s)$	$\xrightarrow{\tilde{c}}$	
	$\xleftarrow{t}$	$t \leftarrow \mathtt{Challenge}(\cdot)$
$(c, d, \pi) \leftarrow \mathtt{Generate}(\tilde{c}, \tilde{d}, t)$	$\xrightarrow{(c, \pi)}$	
		$0 \vee 1 \leftarrow \mathtt{Check}(\tilde{c}, t, c, \pi)$

Figure: Our abstract verifiable random secret scheme.

Verifiable Random Secrets

A VRS has the following properties:

- *Completeness*,
- *Binding*,
- *Prover bit-Unpredictability*, and
- *Honest-Verifier Secrecy*.

Subliminal-Free Digital Signatures

Definition (Subliminal-Free Digital Signature Scheme)

- $\texttt{KeyGen}$, on input the security parameter 1^λ, outputs public parameters $\texttt{pp}$, a signing key $\texttt{sk}$, and a verification key $\texttt{vk}$,
- $\texttt{Setup}$, on input security parameter 1^λ, outputs public parameters $\texttt{sp}$,
- $\Pi_{\texttt{Sign}}$, on input message m and $\texttt{sk}$, outputs signature σ and proof π,
- $\texttt{Verify}$, on input m, σ and $\texttt{vk}$, outputs either 0 or 1,
- $\texttt{Check}$, on input m, σ, vk and π, outputs either 0 or 1,

We require that $\texttt{Check}$ returns 1 if and only if $\texttt{Verify}$ returns 1 and π is valid.

Subliminal-Free Digital Signatures

A SFS has the following properties:

- *Completeness*,
- *Soundness*, and
- *Security against existential forgery*.

Our Schemes

Lattice-Based VRS

1. `Seed`: `V` draws τ Gaussian distributed polynomials s_i from R_p with standard deviation $\sigma/\sqrt{\kappa}$ and sends them to `P`.
2. `Commit`: `P` shuffles the polynomials using a random permutation γ, commits to them in the new order, and sends the commitments to `V`.
3. `Challenge`: `V` draws three random subset T_j, for $1 \leq j \leq 3$, each of size κ, of indices from 1 to τ and sends them to `P`.
4. `Generate`: `P` sums together the commitments for each set of indices, and sends the sums to `V` together with the proof of shuffle.
5. `Check`: `V` verifies that the sums and the proof of shuffle are correct.

Our Schemes

Lattice-Based Subliminal-Free Signature Scheme

Prover		Verifier
		Seed:
	$\xleftarrow{s = \{s_i\}}$	$s_i \xleftarrow{\$} \mathcal{N}_{\sigma/\sqrt{\kappa}}, 1 \leq i \leq \tau$
Com:		
$\gamma \xleftarrow{\$} S_\tau$		
$(\tilde{c}_i, \tilde{d}_i) \leftarrow \mathtt{Com}(s_{\gamma(i)})$	$\xrightarrow{\tilde{c} = \{\tilde{c}_i\}}$	
		Challenge:
$\pi_S \leftarrow \Pi_{\text{Shuffle}}(\{\tilde{c}_i\}, \{s_i\}, \gamma)$		$T_j \overset{\$}{\subset} \{1, \ldots, \tau\}$,
	$\xleftarrow{t = \{T_j\}}$	$\lvert T_j \rvert = \kappa, 1 \leq j \leq 3$
Generate:		
$(c_j, d_j) \leftarrow \sum_{l \in T_j} \mathtt{Com}(s_{\gamma^{-1}(l)})$		
$\pi_L \leftarrow \Pi_{\text{Lin}}(\{c_j\}, t', (1, a_1, a_2))$		
$(t', \boldsymbol{z}) \leftarrow \mathtt{Sign}(m, \mathtt{sk})$	$\xrightarrow{(m, (t', \boldsymbol{z})), (\{c_j\}, (\pi_S, \pi_L))}$	
		Check:
		$1 \overset{?}{=} \Pi_{\text{ShuffleV}}(\{\tilde{c}_i\}, \{s_i\}, \pi_S)$
		$1 \overset{?}{=} \Pi_{\text{LinV}}(\{c_j\}, t', (1, a_1, a_2), \pi_L)$
		Verify:
		$1 \overset{?}{=} \mathtt{Verify}(\mathtt{vk}, m, (t', \boldsymbol{z}))$
		If all algorithms output 1:
		Send $(m, (t', \boldsymbol{z}))$ to the receiver.

Our Schemes

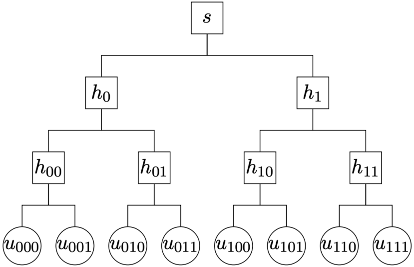

Figure: Merkle-tree

Our Schemes

Hash-Based VRS

1. `Seed`: `P` chose a random bit string s of length 3λ and keeps this private.
2. `Commit`: `P` generates the full tree applying the algorithm `BuildTree` on s, and sends the root $\tilde{c}$ to `V` as a commitment.
3. `Challenge`: `V` draws a random index $t = l$, where $0 \leq l \leq M - 1$, and sends t to `P`.
4. `Generate`: `P` publishes $c = w_l$ and the proof π_l, generated by applying the algorithm `SubTrees` on s and l, which contains the roots of the subtrees not on the path between s and u_l.
5. `Check`: `V` verifies that w_l and π_l generates the tree by applying the algorithm `CompleteTree` to w_l and π_l and comparing the root to $\tilde{c}$.

Our Schemes

Hash-Based Subliminal-Free Signature Scheme

Prover		Verifier
Seed:		
$s \xleftarrow{\$} \{0,1\}^{3\lambda}$		
Com:		
$(\tilde{c}, \tilde{d}) \leftarrow \texttt{BuildTree}(s)$	$\xrightarrow{\tilde{c}}$	
		Challenge:
	$\xleftarrow{t = I}$	$I \xleftarrow{\$} \{0, ..., M-1\}$
Generate:		
$(c, d) \leftarrow (w_I, v_I)$		
$\pi_I \leftarrow \texttt{SubTrees}(s, I)$		
$(w_I, z) \leftarrow \texttt{Sign}(m, \texttt{sk})$	$\xrightarrow{(c, \pi_I, (w_I, z))}$	
		Check:
		$\tilde{c} \stackrel{?}{=} \texttt{CompleteTree}(w_I, \pi_I)$
		Verify:
		$1 \stackrel{?}{=} \texttt{Verify}(\texttt{vk}, m, (w_I, z)))$
		If all algorithms output 1:
		Send $(m, (w_I, z))$ to the receiver.

Conclusion

- The concrete instantiation of the purely lattice-based scheme can be made non-interactive and it takes less than 5 seconds to create a subliminal-free signature of total size $\approx$ 12.65 MB.
- The concrete instantiation of the hash-based scheme combined with lattice-based signatures is interactive and it takes $\approx$ 1 second to generate a subliminal-free signature of size 3.3 KB, where a malicious signer has probability 2^{-10} to embed subliminal information into the signature.

Thank you! Any questions?

Presentation available at `tjerandsilde.no/talks`.

Norwegian University of Science and Technology