Efficient Quantum-immune Blind Signatures
— preliminary version —

Markus Rückert
rueckert@cdc.informatik.tu-darmstadt.de

Cryptography and Computeralgebra
Department of Computer Science
TU Darmstadt

August 4, 2008

Keywords Quantum-immune, blind signatures

Abstract We present the first quantum-immune blind signature scheme. Our scheme is provably secure, efficient, and round-optimal.

1 Introduction

Since 1982, when David Chaum proposed his idea of blind signatures and a, by now classic, application in the context of digital payments, numerous blind signature schemes and other privacy-enhanced signature schemes have been developed.

Today, when building provably secure signature schemes, one has to keep emerging technologies and especially quantum computers in mind. In the quantum-age, the cryptographic assumptions change with the leap in computing power that quantum computers will provide.

To date, there are only a few cryptographic assumptions that are conjectured to be *quantum-immune*, i.e. they are considered to be able to withstand quantum computer attacks. One of those assumptions is the hardness of approximating shortest vectors (SVP) in a lattice. Although the work of Ludwig [11] suggests that todays lattice reduction algorithms can benefit from the intrinsic parallelity in quantum computation, this does not invalidate the assumption. Slightly larger security parameters are considered to be a sufficient countermeasure.

Using the SVP as our security assumption, we construct the first quantum-immune blind signature scheme. As for its efficiency, we state that it is almost as efficient as the underlying signature scheme proposed by Gentry, Peikert, and Vaikuntanathan (GPV) [9]. With its two rounds, it is even *round-optimal*. The security of both, GPV signature scheme and our blind signature scheme, is proven in the random oracle model and, due to Ajtai's result, is based on the worst case hardness of the SVP.

All previous constructions have one thing in common. They are built upon number theoretic assumptions, like the hardness of factoring large integers or computing discrete logarithms. Newer approaches, like that of [5], use pairings and bilinear maps that yield very elegant constructions. They, however, are again based on the discrete logarithm problem in this specific setting.

None of the above assumptions hold in the presence of quantum computers, where both factoring and computing discrete logarithms becomes easy due to the seminal work of Peter Shor [12].

Despite the uninstantiability result of Canetti, Goldreich, and Halevi [7], we believe that our construction is an important step towards quantum-immune blind signature schemes.

Organization. After a brief preliminaries section, we present our construction in Section 3. There, we also prove that our scheme has the well-established security properties. In Section 4, we discuss the details and the realization of the underlying trapdoor permutation. Finally, in Section 5, we propose reasonable parameters that lead to secure and efficient instantiations of our scheme.

2 Preliminaries

With n, we always denote the security parameter. $\langle \cdot \leftrightharpoons \cdot \rangle$ denotes the protocol view generated by two entities, i.e. the messages they exchange. Views are interpreted as random variables, whose output is generated by subsequent executions of the respective protocol. Two views $\mathcal{V}_1$ and $\mathcal{V}_2$ are equal if they cannot be distinguished by any computationally unbounded algorithm with non-negligible probability.

3 Construction

In this section, we describe the construction of our blind signature scheme and prove its security in terms of *blindness* and *one-more unforgeability*.

The underlying signature scheme was developed by Gentry, Peikert, and Vaikuntanathan (GPV) and presented at STOC 2008 [9]. It is built upon a family of trapdoor functions, which are arguably as good as trapdoor permutations. The family is described via a triple ($\mathsf{TrapGen}, \mathsf{SampleDom}, \mathsf{SamplePre}$) and has, among others, the following properties.

Function generation. There is an efficient algorithm $\mathsf{TrapGen}$ that outputs $(a, t) \leftarrow \mathsf{TrapGen}(n)$, where a fully defines the function f_a and t is used to sample from the inverse $f_t^{-1}(\cdot)$, which is defined as $\mathsf{SamplePre}(t, \cdot)$.

Efficiency. The function $f_a : D_n \to R_n$ is efficiently computable. Furthermore, the two sets R_n, D_n are efficiently recognizable and R_n is closed under addition. Furthermore, let $\widetilde{D_n} \subseteq D_n$.

One-wayness. Computing the function $f_t^{-1} : R_n \to \widetilde{D_n}$, is infeasible without the trapdoor t.

Domain sampling with uniform output. $\mathsf{SampleDom}(n)$ samples values from some distribution over $\widetilde{D_n}$, such that their images under f_a are uniformly distributed over R_n.

Pre-image sampling. Let $y \in R_n$. $f_t^{-1}(y)$ samples $x \leftarrow \mathsf{SampleDom}(n)$ under the condition that $f_a(x) = y$.

Linearity. Let $x_1 + x_2 \in D_n$. $f_a(x_1 + x_2) = f_a(x_1) + f_a(x_2)$.

Collision resistance. There exists no algorithm $\mathcal{A}(n, a)$ that outputs a pair $(x, x') \in D_n^2$, such that $x \neq x'$ and $f_a(x) = f_a(x')$, in time polynomial in n with non-negligible probability.

Note that we slightly modified the original construction regarding the domain D_n. In [9] is is always the same, whereas we have introduced a different $\widetilde{D_n}$ for preimage sampling. See Section 4 for details.

In addition to the above trapdoor function, Gentry, Peikert, and Vaikuntanathan use the "hash-then-sign" paradigm with a full-domain hash function (cf. [6]) $\mathsf{H} \leftarrow \mathcal{H}(n)$, where $\mathsf{H} : \{0,1\}^* \to R_n$ and $\mathcal{H}$ is a family of collision-resistant hash functions implementing the random oracle. In this setting, the GPV signature scheme is strongly unforgeable under a chosen message attack. Strong unforgeability of digital signatures means that an adversary is allowed to adaptively query a signature oracle on chosen messages. The adversary wins if it is able to output a new message-signature pair (m, σ), in the sense that the signature oracle has never answered with σ on the query m.

With the modification $\widetilde{D_n} \subseteq D_n$, the GPV signature scheme is a tuple $\mathsf{GPV} = (\mathsf{Kg}, \mathsf{Sig}, \mathsf{Vf})$, where

Key generation. $\mathsf{GPV.Kg}(1^n)$ outputs $(a, t) \leftarrow \mathsf{TrapGen}(1^n)$.

Signature issue. Let $m \in \{0,1\}^*$ be a message. $\mathsf{GPV.Sig}(t, m)$ checks whether m has been signed before and, if so, outputs the same signature. Otherwise, it computes $\sigma \leftarrow f_t^{-1}(\mathsf{H}(m))$, stores (m, σ), and returns σ.

Verification. Given a signature σ. $\mathsf{GPV.Vf}(a, \sigma, m)$ returns 1 iff $\sigma \in D_n$ and $f_a(\sigma) = \mathsf{H}(m)$.

Using a slight relaxation of the above signature scheme, we construct an equally efficient and provably secure blind signature scheme $\mathsf{BS} = (\mathsf{Kg}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Blind}, \mathsf{Unblind})$ as follows.

Key generation. $\mathsf{BS.Kg}(n)$ outputs $(a, t) \leftarrow \mathsf{TrapGen}(n)$, where a is the public verification key and t is the secret signing key.

Blinding. Let $m \in \{0,1\}^*$ be a message. $\mathsf{BS.Blind}(a, m)$ chooses a blinding value $\beta \leftarrow \mathsf{SampleDom}(n)$ and computes $m^\star \leftarrow \mathsf{H}(m) + f_a(\beta)$. The output is $(\beta, m^\star)$.

Signature issue. Let $m^\star$ be a blinded message. $\mathsf{BS.Sig}(t, m^\star)$ computes $\sigma^\star \leftarrow f_t^{-1}(m^\star)$ and returns $\sigma^\star$.

Unblinding. Let $\sigma^\star$ be a blinded signature for the message m and the blinding value β. $\mathsf{BS.Unblind}(a, m, \beta, \sigma^\star)$ computes $\sigma \leftarrow \sigma^\star - \beta$. It checks whether $\sigma \in D_n$ and $f_a(\sigma) = \mathsf{H}(m)$. If either of the conditions is violated, the algorithm aborts with fail.

Verification. $\mathsf{BS.Vf}(a, \sigma, m)$ outputs 1 iff $\sigma \in D_n$ and $f_a(\sigma) = \mathsf{H}(m)$.

If BS.Unblind aborts, it may be that the signer is dishonest. In the special setting of e-cash, if the obtained signature σ is not in the domain of f_a, the process has to be repeated with a different m and the receiver of the signature has to reveal β to prove to the signer that the signature is literally worthless. For the moment, we assume that $\sigma^\star - \beta \in D_n$. In Section 4, its becomes obvious that this always holds if both parties are honest.

Completeness. The scheme BS is complete because for all honestly generated key pairs (a, t), all messages $m \in \{0,1\}^*$, all outputs $(\beta, m^\star)$ of $\mathsf{BS.Blind}(a, m)$, and all signatures $\sigma^\star \leftarrow \mathsf{BS.Sig}(t, m^\star)$ we have

$$\sigma \leftarrow \sigma^\star - \beta \in D_n$$

and

$$f_a(\sigma) = f_a(\sigma^\star - \beta) = f_a(\sigma^\star) - f_a(\beta) = f_a(f_t^{-1}(\mathsf{H}(m) + f_a(\beta))) - f_a(\beta) = \mathsf{H}(m)\,.$$

Therefore, $\mathsf{BS.Vf}(a, \sigma, m) = 1$.

In the following, we prove the security of our blind signature scheme. A blind signature scheme is called secure if it satisfies *blindness* and *one-more unforgeability* as defined by Juels, Luby, and Ostrovsky in [10].

Blindness. The notion of blindness is defined in the following experiment, where the adversarial signer $\mathcal{S}^*$ chooses two messages m_0, m_1 and interacts with two users who obtain blind signatures for the two messages in random order. After seeing the unblinded signatures in the original order, according to m_0, m_1, the signer has to guess the message that has been signed for the first user.

Experiment $\mathsf{Exp}^{\mathsf{blind}}_{\mathcal{S}^*,\mathsf{BS}}(n)$
- $b \leftarrow \{0,1\}$
- $(pk, sk) \leftarrow \mathsf{BS.Kg}(n)$
- $(m_0, m_1) \leftarrow \mathcal{S}^*(n, pk, sk)$
- Setup users $\mathcal{U}_0(n, pk, m_b)$, $\mathcal{U}_1(n, pk, m_{1-b})$
- $\mathcal{U}_0, \mathcal{U}_1$ interact with $\mathcal{S}^*$ using the blind signature protocol of BS.
- $\mathcal{U}_0, \mathcal{U}_1$ output signatures σ_b on m_b and σ_{1-b} on m_{1-b}, where either of them might equal fail.
- If neither of the users' output equals fail
  - $d \leftarrow \mathcal{S}^*(n, sk, pk, \sigma_0, \sigma_1)$
- Else
  - $d \leftarrow \mathcal{S}^*(n, sk, pk, \mathsf{fail}, \mathsf{fail})$
  - Return 1 iff $d = b$

A signature scheme BS is (t, ϵ)-blind if there is no adversary $\mathcal{S}^*$, running in time at most t, that wins the above experiment with advantage at least ϵ, where

$$\mathsf{Adv}^{\mathsf{blind}}_{\mathcal{S}^*,\mathsf{BS}} = \Pr[\mathsf{Exp}^{\mathsf{blind}}_{\mathcal{S}^*,\mathsf{BS}}(n) = 1] - \frac{1}{2}\,.$$

The next theorem proves that, like Chaum's blind signature scheme [8], BS is unconditionally blind, i.e. $(\infty, 0)$-blind. In short, we construct a simulator that, given any b, any two messages, and any pair (β_0, β_1), constructs a pair of blinding values that generate the same view (as with b), while reversing the order in which the signatures are obtained from the signer (as with $1 - b$).

Theorem 1 (Blindness). *The blind signature scheme* BS *is* $(\infty, 0)$*-blind.*

Proof. Let $\mathcal{U}(b)$ denote the user that tries to obtain a signature on m_b. For the prove, we define a simulator $\mathcal{SIM}$ that, given a fixed b, generates a view $\langle \mathcal{S}^* \rightleftharpoons (\tilde{\mathcal{U}}(1-b); \tilde{\mathcal{U}}(b)) \rangle$ that equals $\langle \mathcal{S}^* \rightleftharpoons (\mathcal{U}(b); \mathcal{U}(1-b)) \rangle$ from $\mathcal{S}^*$'s point of view. Thus, effectively reversing the order in which the two messages are signed by $\mathcal{S}^*$. Therefore, $\mathcal{S}^*$ can only guess b with probability $1/2$ and $\mathsf{Adv}^{\mathsf{blind}}_{\mathcal{S}^*,\mathsf{BS}} = 0$. $\mathcal{SIM}$ works as follows.

Setup. $\mathcal{SIM}$ gets as input a bit b, two messages m_0, m_1, and the public parameters of BS. $\mathcal{SIM}$ chooses $\beta_0, \beta_1 \leftarrow \mathsf{SampleDom}(n)$. Let $\beta_0' \leftarrow \mathsf{H}(m_b) - \mathsf{H}(m_{1-b}) + f_a(\beta_0)$ and $\beta_1' \leftarrow \mathsf{H}(m_{1-b}) - \mathsf{H}(m_b) + f_a(\beta_1)$.
Simulated user $\tilde{\mathcal{U}}(1-b)$. Send $m_b^\star \leftarrow \beta_1' + \mathsf{H}(m_b)$ to $\mathcal{S}^*$ and receive $\sigma^\star$. Output $\sigma_{1-b} \leftarrow \sigma^\star - \beta_1$.
Simulated user $\tilde{\mathcal{U}}(b)$. Send $m_{1-b}^\star \leftarrow \beta_0' + \mathsf{H}(m_{1-b})$ to $\mathcal{S}^*$ and receive $\sigma^\star$. Output $\sigma_b \leftarrow \sigma^\star - \beta_0$.
Output. The two signatures σ_0, σ_1.

Analyis. In short, for each choice of b and blinding values $f_a(\beta_0), f_a(\beta_1)$, there is exactly one pair of blinding values β_1', β_0' that generates a view for $1-b$ that is equal to the view with b.

The choice of β_0', β_1' is fully random due to the fact that $f_a(\beta_0)$ and $f_a(\beta_1)$ are distributed uniformly at random over R_n and because H is a random oracle. Therefore, these blinding values look exactly like the blinding values randomly chosen by the users $\mathcal{U}(b)$ and $\mathcal{U}(1-b)$. Because of that, $\langle \mathcal{S}^* \rightleftharpoons (\tilde{\mathcal{U}}(1-b); \tilde{\mathcal{U}}(b)) \rangle$ equals $\langle \mathcal{S}^* \rightleftharpoons (\mathcal{U}(b); \mathcal{U}(1-b)) \rangle$. The simulator, however, obtains the signatures in reverse order w.r.t. the users $\mathcal{U}(b)$, $\mathcal{U}(1-b)$, i.e. first for m_{1-b} and then for m_b.

Furthermore, note that the simulator outputs fail whenever the users would. Therefore, $\mathcal{S}^*$ cannot distinguish the simulator from two honest users via aborts. $\square$

One-more unforgeability. Unforgeability in the context of blind signatures is defined in the experiment $\mathsf{Exp}^{\mathsf{omf}}_{\mathcal{U}^*,\mathsf{BS}}$. There, a malicious user $\mathcal{U}^*$ is successful if it is possible to obtain $\ell+1$ distinct signatures on $\ell+1$ messages from ℓ interactions with the signer. More formally:

Experiment $\mathsf{Exp}^{\mathsf{omf}}_{\mathcal{U}^*,\mathsf{BS}}(n)$
$b \leftarrow \{0,1\}$
$\mathsf{H} \leftarrow \mathcal{H}(n)$
$(pk, sk) \leftarrow \mathsf{BS.Kg}(n)$
$\{(m_1, \sigma_1), \ldots, (m_\jmath, \sigma_\jmath)\} \leftarrow \mathcal{U}^{*\mathsf{H}(\cdot), \mathsf{BS.Sig}(sk, \cdot)}(n, pk)$
Let ℓ be the number of (complete) interaction between $\mathcal{U}^*$ and the signer.
Return 1 iff $\mathsf{BS.Vf}(pk, \sigma_i, m_i) = 1$ for all $i = 1, \ldots, \jmath$ and $\ell < \jmath$.

A signature scheme BS is $(t, q_{\mathsf{Sig}}, q_{\mathsf{H}}, \epsilon)$-one-more unforgeable if there is no adversary $\mathcal{A}$, running in time at most t, making at most q_{Sig} signature queries and at most q_{H} hash oracle queries, that wins the above experiment with probability at least ϵ.

We prove that our blind signature scheme is provably secure under a reasonable assumption, namely that the following "one-more trapdoor inversion problem" is hard.

Definition 1 (Chosen target trapdoor inversion problem (CTTI)). *The chosen target trapdoor inversion problem is defined via the following experiment, where the adversary $\mathcal{A}$ has access to a challenge oracle O_{R_n} and to an inversion oracle f_t^{-1}. The adversary wins, if it outputs $\jmath$ preimages for challenges obtained from O_{R_n}, while making only $\ell < \jmath$ queries to f_t^{-1}.*

Experiment $\mathsf{Exp}_{\mathcal{A}}^{\mathsf{ctti}}(n)$
$(a,t) \leftarrow \mathsf{TrapGen}(n)$
$(\pi, x_1, \ldots, x_\jmath) \leftarrow \mathcal{A}^{\mathsf{O}_{R_n}, f_t^{-1}(\cdot)}(n,a)$
Let $y_1, \ldots, y_\ell$ be the challenges returned by O_{R_n}.
Let $\imath$ be the number of queries to f_t^{-1}.
Return 1 iff
1. $\pi : \{1, \ldots, \jmath\} \to \{1, \ldots, \ell\}$ *is injective and*
2. $f_a(x_i) = y_{\pi(i)}$ *for all* $i = 1, \ldots, \jmath$ *and*
3. $\imath < \jmath$.

The problem is $(t, q_{\mathsf{I}}, q_{\mathsf{O}}, \epsilon)$-hard if there is no algorithm $\mathcal{A}$, running in time at most t, making at most q_{I} inversion queries and at most q_{O} queries to O_{R_n}, which wins the above experiment with probability larger than ϵ.

The one-wayness of f_a gives us $(\mathsf{poly}(n), 0, 1, \epsilon)$-hardness, which we will extend to $(\mathsf{poly}(n), \mathsf{poly}(n), \mathsf{poly}(n), \epsilon')$-hardness for a negligible ϵ'. With our definition and this assumption, we follow the line of thought of Bellare, Namprempre, Pointcheval, and Semanko in [4]. They define a collection of "one-more" problems in the RSA context.

As for its hardness, we show that it is as hard as forging GPV signatures.

Theorem 2. *The* CTTI *is* $(t, q_{\mathsf{I}}, q_{\mathsf{O}}, \epsilon)$*-hard if and only if the* GPV *signature is* $(t, q_{\mathsf{I}}, q_{\mathsf{O}}, \epsilon)$*-strongly unforgeable.*

Proof. We show both directions separately.

$\mathsf{CTTI} \Rightarrow \mathsf{GPV}$*:* Let's assume that GPV is not $(t, q_{\mathsf{I}}, q_{\mathsf{O}}, \epsilon)$-strongly unforgeable. Thus, there exists a forger $\mathcal{A}$ against strong unforgeability. Using $\mathcal{A}$, we construct an adversary $\mathcal{B}$ that solves the CTTI. The adversary $\mathcal{B}$ works as follows.

Setup. $\mathcal{B}$ sets up a list $L_{\mathsf{H}} \leftarrow \emptyset$ of triples (m, c, s), which is indexed by the first component, and a counter $\ell \leftarrow 0$. It gets as input the public trapdoor parameter a and executes $\mathcal{A}$ on input a in a black-box simulation. $\mathcal{B}$ has access to O_{R_n} and $f_t^{-1}(\cdot)$.

Random oracle H. For each query m of $\mathcal{A}$ to the random oracle H, algorithm $\mathcal{B}$ searches L_{H} for a triple $(m, c, *)$. If it exists, $\mathcal{B}$ outputs c. Otherwise, $\mathcal{B}$ increments ℓ, queries its challenge oracle $c_\ell \leftarrow \mathsf{O}_{R_n}$, stores $(m_\ell \leftarrow m, c_\ell, \square)$ in L_{H}, and outputs c_ℓ. $\square$ serves as a placeholder for "uninitialized".

Signature queries. When $\mathcal{A}$ queries its signature oracle on m, algorithm $\mathcal{B}$ searches L_{H} for a triple (m_i, c_i, s_i). If it exists, $\mathcal{B}$ outputs s_i. Otherwise, $\mathcal{B}$ queries f_t^{-1} with $\mathsf{H}(m_i)$, receives s_i, stores (m_i, c_i, s_i) in L_{H}, and returns s_i to $\mathcal{A}$.

Output. When $\mathcal{A}$ stops, it outputs a forgery (m^*, σ^*). Assume $m^* = m_\jmath$. Let $L'_{\mathsf{H}} = \{(m^{(1)}, c^{(1)}, s^{(1)}), \dots, (m^{(q_{\mathsf{I}})}, c^{(q_{\mathsf{I}})}, s^{(q_{\mathsf{I}})})\}$ be the set of all triples in L_{H}, excluding those of form $(*, *, \Box)$. $\mathcal{B}$ sets

$$\pi = \{(i,j) : \exists a^{(i)} \in L'_{\mathsf{H}}\, \exists b_j \in L_{\mathsf{H}} : a^{(i)} = b_j\} \cup \{(q_{\mathsf{I}} + 1, \jmath)\}$$

and outputs $(\pi, s^{(1)}, \dots, s^{(q_{\mathsf{I}})}, \sigma^*)$.

Analyis. Note that $\mathcal{B}$ perfectly simulates $\mathcal{A}$'s environment. Since f_a is collision resistant, we can safely assume that $\mathcal{A}$ outputs a forgery on a message m^* that has never been sent to the signing oracle. Thus, $\mathcal{B}$ has not queried f_t^{-1} on $\mathsf{H}(m^*)$. Therefore, $\mathcal{B}$ makes $\imath = q_{\mathsf{I}}$ queries to f_t^{-1} and outputs $q_{\mathsf{I}}+1$ preimages along with an injective map π. Thus, the first and last requirements in the CTTI experiment are met. As for the second requirement, we state that $f_a(s_i) = \mathsf{H}(m_{\pi(i)})$ for all $i \neq \jmath$ and $f_a(\sigma^*) = \mathsf{H}(m^*) = \mathsf{H}(m_{\pi(\jmath)})$. Therefore, $\mathcal{B}$ is successful whenever $\mathcal{A}$ is.

$\mathsf{CTTI} \Leftarrow \mathsf{GPV}$*:* Now, assume that the CTTI is not $(t, q_{\mathsf{I}}, q_{\mathsf{O}}, \epsilon)$-hard, i.e. there exists an adversary $\mathcal{A}$ that efficiently solves the problem. We show that $\mathcal{A}$ can be used to break strong unforgeability of GPV. We construct a forger $\mathcal{B}$ as follows.

Setup. $\mathcal{B}$ gets as input the public trapdoor parameter a. It sets up a list $L_{\mathsf{H}} \leftarrow \emptyset$ of triples (m, c, x), indexed by m. Furthermore it initializes a counter $\ell \leftarrow 0$. It runs a black-box simulation of $\mathcal{A}$ on input a.

Random oracle H. On input m, $\mathcal{B}$ searches L_{H} for a triple $(m, c, *)$. If it exists, it outputs c. Otherwise, $\mathcal{B}$ increases ℓ, chooses a new $c_\ell \leftarrow R_n$, and stores $(m_\ell \leftarrow m, c_\ell, \Box)$ in L_{H}, where "$\Box$" denotes "uninitialized". Finally, $\mathcal{B}$ returns c_ℓ.

Challenge oracle queries. $\mathcal{B}$ chooses a new $m \leftarrow D_n$, computes $c \leftarrow \mathsf{H}(m)$, and returns c.

Inversion queries. On input c, $\mathcal{B}$ searches L_{H} for a triple (m_i, c, x_i). If it exists and $x_i \neq \Box$ then $\mathcal{B}$ outputs x_i. If it does not exist then $\mathcal{B}$ increments ℓ, sets $i \leftarrow \ell$, chooses a new $m_\ell \leftarrow D_n$, and adds $(m_\ell, c, \Box)$ to L_{H}. Finally, $\mathcal{B}$ queries $x_\ell \leftarrow f_t^{-1}(c)$, stores (m_i, c, x_i), and returns x_i.

Output. When $\mathcal{A}$ stops, it outputs $(\pi, x_1, \dots, x_\jmath)$. Algorithm $\mathcal{B}$ searches the lowest index i, for which $(m_{\pi(i)}, c_{\pi(i)}, \Box) \in L_{\mathsf{H}}$. It outputs the forgery $(m_{\pi(i)}, x_i)$.

Analysis. First of all, note that $\mathcal{B}$ perfectly simulates all of $\mathcal{A}$'s oracles. Since $\mathcal{A}$ is a successful chosen target trapdoor inverter, there is an index i with $f_a(x_i) = c_{\pi(i)}$, such that $\mathcal{A}$ never queried the inversion oracle on $c_{\pi(i)}$. Therefore, $\mathcal{B}$ has never queried its signature oracle on $\mathsf{H}(m_{\pi(i)}) = c_{\pi(i)}$ and x_i is a valid forgery on the message $m_{\pi(i)}$.

In both proofs, the number of inversion queries equals the number of signature queries and the number of challenge oracle oracle queries equals the number of queries to the random oracle. The overhead of handling $\mathcal{A}$'s queries is minimal and consists mainly of list operations that can be neglected because they are essentially the same in both reductions. This concludes the proof. □

Using the last theorem, we can now prove one-more unforgeability of our blind signature scheme.

Theorem 3. *The* BS *blind signature scheme is* $(t, q_{\mathsf{Sig}}, q_{\mathsf{H}}, \epsilon)$*-one-more unforgeable if the* CTTI *is* $(t, q_{\mathsf{Sig}}, q_{\mathsf{H}}, \epsilon)$*-hard.*

Proof. Towards contradiction, we assume that there exists a successful forger $\mathcal{A}$ against one-more unforgebility of BS. Using $\mathcal{A}$, we construct an algorithm $\mathcal{B}$ via a black-block simulation, such that $\mathcal{B}$ solves the respective instance of the CTTI. The simulation works as follows.

Setup. $\mathcal{B}$ gets as input the public trapdoor parameter a an has access to the challenge oracle O_{R_n} and to a trapdoor inversion oracle f_t^{-1}. $\mathcal{B}$ initializes a list $L_{\mathsf{H}} \leftarrow \emptyset$ of pairs (m, c), indexed by m, a list $L_{\mathsf{I}} \leftarrow \emptyset$ of pairs $(m^\star, \sigma^\star)$, indexed by $m^\star$, and two counters $\ell \leftarrow 0$, $\imath \leftarrow 0$. It runs $\mathcal{A}$ on input a in a black-box simulation.

Random oracle queries. On input m, $\mathcal{B}$ looks up m in L_{H}. If it finds a pair (m, c) then it returns c. Otherwise, $\mathcal{B}$ increments $\imath$, chooses a new $c_\imath$, stores $(m_\imath \leftarrow m, c_\imath)$ in L_{H}. Afterwards, $\mathcal{B}$ returns $c_\imath$.

Blind signature queries. On input $m^\star$, algorithm $\mathcal{B}$ searches a pair $(m^\star, \sigma^\star)$ in L_{I}. If it exists, $\mathcal{B}$ returns $\sigma^\star$. Otherwise, algorithm $\mathcal{B}$ increments ℓ, queries its inversion oracle $\sigma_\ell^\star \leftarrow f_t^{-1}(m^\star)$, stores $(m_\ell^\star \leftarrow m^\star, \sigma_\ell^\star)$ in L_{I}, and returns $\sigma_\ell^\star$.

Output. Finally, $\mathcal{A}$ stops and outputs $((m_1, \sigma_1), \ldots, (m_\jmath, \sigma_\jmath))$, $\ell < \jmath$. W.l.o.g., assume that $(m_i, c_i) \in L_{\mathsf{H}}$, for all $i = 1, \ldots, \jmath$. Algorithms $\mathcal{B}$ sets

$$\pi = \{(i, j) : f_a(\sigma_i) = c_j\}$$

and outputs $(\pi, \sigma_1, \ldots, \sigma_\jmath)$.

Analysis. First, observe that all of $\mathcal{A}$'s oracles are perfectly simulated. When $\mathcal{A}$ calls H, algorithm $\mathcal{B}$ draws a new challenge from its challenge oracle. Whenever $\mathcal{A}$ queries its signature oracle on a new blinded message, $\mathcal{B}$ calls its inversion oracle. Therefore, when $\mathcal{A}$ outputs a one-more forgery, $\mathcal{B}$ can use it to solve the CTTI. $\mathcal{B}$'s output is valid in the CTTI experiment because all preimages evaluate to challenges received from O_{R_n} and the number of output inversions $\jmath$ is greater than the number of inversion queries ℓ. As for the map π, we state that it is injective. Otherwise, there would be a a pair $\sigma \neq \sigma'$ in $\mathcal{A}$'s output with $f_a(\sigma) = f_a(\sigma') = \mathsf{H}(m_i)$, which contradicts the collision resistance of f_a. Thus $\mathcal{B}$ is successful when $\mathcal{A}$ is.

Again, the overhead of handling $\mathcal{A}$'s queries is dominated by simple list processing and can be neglected. □

Together with Theorem 2, our construction is one-more unforgeable if the GPV signature is strongly unforgeable.

4 Realization

The underlying signature scheme was developled by Gentry, Peikert, and Vaikuntanathan (GPV) and presented at STOC 2008 [9]. It uses a modified Babai nearest plane algorithm [3] and two famous results by Ajtai [1,2] in order to build a trapdoor function that is arguably "as good as" a trapdoor permutation. It's security is proven in the random oracle model and reduces to the collision resistance of f_a, which in turn reduces to the hardness of finding short vectors in a lattice.

GPV *trapdoor function.* The trapdoor function from [9] is defined as follows.

Parameters. Depending on the security parameter n, the other parameters in [9] are the following.

Modulus	$q = n^3$
Domain dimension	$m = 5\,n\,\log(q)$
Basis length bound	$L = m^{1+\epsilon}, \epsilon > 0$
Gaussian parameter	$s = L\,\omega(\sqrt{\log(m)})$

The above paramters will be made explicit in Section 5.

Spaces. The range is

$$R_n = \mathbb{Z}_q^n$$

and the domain D_n is

$$D_n = \{\mathbf{e} \in \mathbb{Z}^m : \|\mathbf{e}\|_\infty \leq s\,\omega(\sqrt{\log(m)})\}\,.$$

In addition, we have introduced a second set

$$\widetilde{D_n} = \{\mathbf{e} \in \mathbb{Z}^m : \|\mathbf{e}\|_\infty \leq 2\,s\,\omega(\sqrt{\log(m)})\}\,,$$

which is the range of the inversion function f_t^{-1}.

Trapdoor description. The public trapdoor key a describes the above public parameters and the public matrix

$$\mathbf{A} \in \mathbb{Z}_q^{n\times m}\,.$$

The set

$$\Lambda^\perp(\mathbf{A}) = \{\mathbf{v} \in \mathbb{Z}^m : \mathbf{A}\,\mathbf{v} \equiv \mathbf{0} \pmod q\}$$

describes a lattice, for which the secret trapdoor paramter t describes a basis $\mathbf{T}$, such that

$$\left\|\tilde{\mathbf{T}}\right\| \leq L\,.$$

Here, $\tilde{\mathbf{T}}$ is the Gram-Schmidt orthogonalized matrix $\mathbf{T}$ and the norm of a matrix is defined as

$$\|\mathbf{X}\| = \left\| \begin{pmatrix} | & & | \\ \mathbf{x_1} & \cdots & \mathbf{x_c} \\ | & & | \end{pmatrix} \right\| = \max_{i=1,\ldots,c} \|\mathbf{x_i}\|_2\,.$$

Trapdoor evaluation. On input $\mathbf{x}$, the trapdoor function $f_a(\mathbf{x})$ evaluates to

$$\mathbf{y} \leftarrow \mathbf{A}\,\mathbf{x} \mod q.$$

Preimage sampling. Sampling from f_t^{-1} is performed via a modified Babai nearest plane algorithm. The algorithm explicitly uses $\mathbf{T}$ and relies on its short length. On input $\mathbf{y}$, it performs the following steps.

1. Compute $\mathbf{t} \in \mathbb{Z}_q^m$, such that $\mathbf{A}\,\mathbf{t} \equiv \mathbf{y} \pmod q$. This is done by linear algebra and most likely yields a $\mathbf{y} \notin \widetilde{D_n}$.
2. Use the trapdoor basis $\mathbf{T}$ to sample a vector $\mathbf{v}$ from a gaussian distribution around $-\mathbf{t}$ and output $\mathbf{x} = \mathbf{t} + \mathbf{v}$

The described trapdoor function has all the properties mentioned in Section 3. As for the required linearity in our blind signature scheme, note that f_a is linear in the sense that for for all $\mathbf{x_1} + \mathbf{x_2} \in D_n$:

$$f_a(\mathbf{x_1} + \mathbf{x_2}) = f_a(\mathbf{x_1}) + f_a(\mathbf{x_2}) \mod q\,.$$

Therefore, all computations in D_n, $\widetilde{D_n}$, and R_n have to be performed modulo q.

Concerning security of the GPV signature scheme, we state that it is unforgeable if the problem of finding short integer solutions $\mathbf{v} \in \mathbb{Z}^m$, $\|\mathbf{v}\|_\infty \leq s\,\omega(\sqrt{\log(m)}$, of the equation

$$\mathbf{A}\,\mathbf{v} \equiv \mathbf{0} \pmod q$$

is hard [9]. As for our modified setting, with $\widetilde{D_n}$ and D_n, we need a slightly stronger assumption, i.e. the above problem has to be hard with $\|\mathbf{v}\|_\infty \leq 2\,s$ $\omega(\sqrt{\log(m)}$. Furthermore, we claim that this special setting cannot be exploited to forge a signature $\sigma' \in \widetilde{D_n}$ from two valid signatures $\sigma_1, \sigma_2 \in D_n$ by simply adding them as $\sigma' \leftarrow \sigma_1 + \sigma_2$ because of the collision resistance of the full-domain hash H.

5 Parameters

In this section, we analyze the choice of parameters in the GPV signature scheme and show how to apply their choice to our blind signature scheme. Then, we assess the practical efficiency of our construction.

This section will appear in the final version.

Acknowledgements

The auther would like to thank Dominique Schröder for numerous fruitful discussions on the subject.

References

1. Miklós Ajtai. *Generating Hard Instances of Lattice Problems (Extended Abstract).* Proceedings of the Annual Symposium on the Theory of Computing (STOC) 1996, Lecture Notes in Computer Science, pages 99–108. Springer-Verlag, 1996.
2. Miklós Ajtai. *Generating Hard Instances of the Short Basis Problem.* 1999, Lecture Notes in Computer Science, pages 1–9. Springer-Verlag, 1999.
3. László Babai. *On Lovász' lattice reduction and the nearest lattice point problem. Combinatorica*, 6(1):1–13, 1986.
4. Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. *The One-More-RSA-Inversion Problems and the Security of Chaum's Blind Signature Scheme. Journal of Cryptology*, 16(3):185–215, 2003.
5. Alexandra Boldyreva. *Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme.* Public-Key Cryptography (PKC) 2003, Volume 2567 of Lecture Notes in Computer Science, pages 31–46. Springer-Verlag, 2003.
6. Mihir Bellare and Pil Rogaway. *Random Oracles are Practical: A Paradigm for Designing Efficient Protocols.* Proceedings of the Annual Conference on Computer and Communications Security (CCS). ACM Press, 1993.
7. Ran Canetti, Oded Goldreich, and Shai Halevi. *The random oracle methodology, revisited. J. ACM*, 51(4):557–594, 2004.
8. David Chaum. *Blind Signatures for Untraceable Payments.* Advances in Cryptology — Crypto 1982, pages 199–203. Plemum, New York, 1983.
9. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. *Trapdoors for hard lattices and new cryptographic constructions.* Proceedings of the Annual Symposium on the Theory of Computing (STOC) 2008, Lecture Notes in Computer Science, pages 197–206. Springer-Verlag, 2008.
10. Ari Juels, Michael Luby, and Rafail Ostrovsky. *Security of Blind Digital Signatures.* Advances in Cryptology — Crypto 1997, Volume 1294 of Lecture Notes in Computer Science, pages 150–164. Springer-Verlag, 1997.
11. Christoph Ludwig. *A Faster Lattice Reduction Method Using Quantum Search.* ISAAC 2003, Volume 2906 of Lecture Notes in Computer Science, pages 199–208. Springer-Verlag, 2003.
12. Peter W. Shor. *Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM Journal on Computing*, 26(5):1484–1509, 1997.