An extended abstract of this paper is published in the proceedings of the 7th International Conference on Cryptology in Africa – Africacrypt 2014.

A Second Look at Fischlin's Transformation

Özgür Dagdelen[1] and Daniele Venturi[2]

[1] *Technische Universität Darmstadt*
[2] *Sapienza University of Rome*

Abstract. Fischlin's transformation is an alternative to the standard Fiat-Shamir transform to turn a certain class of public key identification schemes into digital signatures (in the random oracle model).

We show that signatures obtained via Fischlin's transformation are existentially unforgeable even in case the adversary is allowed to get arbitrary (yet bounded) information on the entire state of the signer (including the signing key and the random coins used to generate signatures). A similar fact was already known for the Fiat-Shamir transform, however, Fischlin's transformation allows for a significantly higher leakage parameter than Fiat-Shamir.

Moreover, in contrast to signatures obtained via Fiat-Shamir, signatures obtained via Fischlin enjoy a tight reduction to the underlying hard problem. We use this observation to show (via simulations) that Fischlin's transformation, usually considered less efficient, outperforms the Fiat-Shamir transform in verification time for a reasonable choice of parameters. In terms of signing Fiat-Shamir is faster for equal signature sizes. Nonetheless, our experiments show that the signing time of Fischlin's transformation becomes, e.g., 22% of the one via Fiat-Shamir if one allows the signature size to be doubled.

Keywords. Fischlin's transformation, leakage, tightness, random oracle

1 Introduction

Digital signatures are among the most fundamental primitives in cryptography. The security of a signature scheme (as introduced by Goldwasser, Micali, and Rivest [29]) is usually defined via a game featuring a computationally bounded adversary $\mathcal{A}$, where the game models how the system can be attacked in the real world. More specifically, $\mathcal{A}$ can see valid message/signature pairs for messages of her choice, and must forge a signature on a "fresh" message, i.e., a message $\mathcal{A}$ may choose, but for which she has not seen a valid signature already. Schemes resistant against such attacks are called existentially unforgeable under adaptive chosen message attacks (in short: ufcma).

To prove that a given signature scheme is unforgeable, one typically builds a "reduction" showing that if an *efficient* $\mathcal{A}$ can win the above security game, then an adversary $\mathcal{B}$ can run $\mathcal{A}$ internally to solve a computational problem believed to be hard. Such a game-based approach is sound if: (i) the security game is a good model of reality; (ii) the constructed reduction is as "tight" as possible. We discuss these issues in detail below.

Abstraction of Reality. One assumption (implicit in the modeling above), is that $\mathcal{A}$ is assumed to interact with the signing oracle in a black-box fashion; this means that all secrets stored "inside the box" are fully hidden to the adversary. Unfortunately this assumption might be too strong and often easy to bypass. In the real world, by exploiting several characteristics of an actual implementation (e.g., timing [34], power consumption [35] and electromagnetic radiation [41]), an attacker can learn *some* information about the secret key, and this information is often sufficient to break otherwise "provably" secure schemes.

Modern-day cryptographic models (starting with [31, 37, 17, 13]) try to formalize side-channel attacks abstractly, with the goal of showing that a scheme has some form of *leakage resilience*.

The standard way of defining leakage-resilient signatures, enhances the unforgeability game by giving $\mathcal{A}$ access to a leakage oracle which outputs bounded (but arbitrary) information about the secret key sk. A signature scheme is existentially unforgeable against λ-leakage attacks if forging signatures on fresh messages is still hard given λ bits of sk-data. Throughout the paper we, in fact, consider a more general setting, where $\mathcal{A}$ leaks information not just about sk, but also about the full randomness used to generate signatures; a scheme secure in this sense is called *fully* leakage-resilient. For a more detailed discussion on leakage models for digital signatures, we refer the reder to Section 1.2.

The value λ is called the leakage parameter of the system. Note that the secret-key size s must be strictly greater than the leakage parameter λ. The quantity λ/s can be thought as the *relative leakage* of the system, with the obvious goal to make it as close to 1 as possible.

Tightness. Suppose that $\mathcal{A}$ takes time t to break the security of a primitive (e.g., a signature scheme) with probability ε. If $\mathcal{B}$ in the reduction has runtime $t' \approx t$ and solves the hard problem with probability $\varepsilon' \approx \varepsilon$, the reduction is *tight*; else it is *loose*. The ratio $(t'\varepsilon)/(t\varepsilon')$ is called the tightness gap of the reduction.

As discussed, e.g., in [33, 9], tight reductions are appealing both for theoretical purposes and because they ensure that the primitive is at least as hard to break as the underlying hard problem. A loose reduction, by contrast, only guarantees that a scheme is "plausibly" secure (see [28]). A loose reduction also results in much larger parameters, and thus much slower performances (depending on the tightness gap). In general, many researchers concerned with practice call into question the practical value of non-tight reductionist security proofs.

1.1 Our Contributions

Σ-protocols are a well-studied class of interactive protocols, run between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$, such that $\mathcal{V}$ accepts $\mathcal{P}$ as legitimate if it is convinced that $\mathcal{P}$ knows a witness w to a shared input x. Each protocol run yields a transcript of the form $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$, where com is sent by the prover. The Fiat-Shamir transform [23] is a common way of constructing efficient signature schemes (in the random oracle model [4]) from a Σ-protocol for some "hard relation".

Recently, Katz and Vaikuntanathan [32] (building on Alwen *et al.* [3]) showed that the Fiat-Shamir transform yields fully leakage-resilient signatures, provided that the underlying Σ-protocol satisfies two additional properties: (i) each theorem has exponentially many witnesses; (ii) the uncertainty of the witness conditioned on the theorem is high.

The obtained scheme has relative leakage asymptotically approaching $1/2$ and a loose reduction (with a tightness gap of about $1/\varepsilon$), due to the fact that the reduction needs to rewind the adversary in order to extract a valid witness and solve the underlying hard problem.

Fischlin's Transformation & Leakage. Fischlin's transformation [24] is an alternative to get secure signatures schemes from arbitrary Σ-protocols. Roughly, Fischlin's transformation consists of a tuple (of dimension r) of Fiat-Shamir signatures, i.e., $(\mathsf{com}_i, \mathsf{ch}_i, \mathsf{resp}_i)_{1 \leq i \leq r}$. However, instead of computing the challenge via $\mathsf{ch}_i = H(\mathsf{com}_i, m)$, the prover tries all values in the domain of challenges such that $H(m, x, \mathbf{com}, i, \mathsf{ch}_i, \mathsf{resp}_i) = 0^b$ for all $i \in [r]$ where $\mathbf{com} = (\mathsf{com}_1, \ldots, \mathsf{com}_r)$. If no such challenge can be found the challenge ch_i is chosen with the smallest output in value. Verifying includes now the check the validity of the r Fiat-Shamir signatures and whether the sum of all hash values are below a certain threshold S.

One important feature of signatures obtained via Fischlin, is that the resulting non-interactive protocol has a *straight-line* extractor. Roughly this means that there exists a probabilistic polynomial time algorithm (a.k.a. the extractor) that, upon input the theorem x, a message m, a valid signature σ on m, and all hash queries and answers made to generate σ, outputs a valid witness for x with overwhelming probability (and without further querying the signer).

Our first result is that Fischlin's transformation yields a fully leakage-resilient signature, whenever the underlying Σ-protocol satisfies properties (i) and (ii) above.

Comparing Fischlin and Fiat-Shamir. Even though the above fact is perhaps not very surprising, Fischlin's transformation comes with two important advantages over leakage-resilient signatures obtained via Fiat-Shamir. The first advantage is that for concrete schemes (e.g., the ones based on Okamoto [39] and Guillou-Quisquater [30]), the relative leakage of the resulting signatures asymptotically approaches 1. The second advantage is that the reduction to the security of the underlying hard problem is tight.[1]

As a consequence of the above observations, one might expect that for a pre-fixed level of security, signatures obtained via Fischlin can be instantiated using much smaller parameters, possibly leading to better performances than signatures obtained via Fiat-Shamir. This is surprising, as usually Fischlin's transformation is considered to be less efficient than Fiat-Shamir.

Our second contribution is an accurate comparison (supported by simulations) of the performances obtained via Fischlin and Fiat-Shamir, in terms of parameters generation, signing and verification time. The comparison is carried out for the Okamoto scheme [39], whose security relies on the hardness of computing discrete logarithms over a finite field. The main findings of our analysis are sketched below:

- Key generation is always faster in Fischlin's transformation, due to the fact that a tight reduction allows to choose smaller parameters.
- In terms of verification time, signatures obtained via Fischlin are much faster than the ones obtained via Fiat-Shamir. This feature makes Fischlin's transformation particularly interesting for scenarios where one demands fast signature verification (e.g., in car2car and car2X communication [44] one might need to verify 4000-5000 signatures per second).
- In terms of signature generation time, for 80-bit security, signatures obtained via Fischlin are two times slower than the ones obtained via Fiat-Shamir if one insists for the resulting signatures having the same size. However, in case one allows signatures obtained via Fischlin to have twice the size of of the ones obtained via Fiat-Shamir, then signature generation becomes 4.5 times faster and verification reduces to 90% of that from Fiat-Shamir.

[1]We stress that the problem of finding a tight reduction for leakage-resilient signatures obtained via Fiat-Shamir remains open.

- When one takes leakage into account, and thus insists that the resulting scheme tolerates a certain amount of leakage, Fischlin's transformation outperforms the Fiat-Shamir transform in terms of security, performance, and signature size.

We remark that, even though in some cases signatures obtained via Fiat-Shamir result in better signing time, Fischlin's transformation might still be preferable in certain scenarios. For instance, an interesting feature of signatures obtained via Fischlin when used in some cryptographic protocol (e.g., for key exchange), is that parties can start verifying the signature (except checking the hash values) before the entire signature is sent. Consequently, the effort to generate and verify a signature consists essentially of the signing time plus $1/r$-th of the verification time. In contrast, signatures obtained via Fiat-Shamir have to be received in full before the verification can start. Taking this feature into account, signing and verifying for Fischlin are indeed faster than for Fiat-Shamir (for 80-bit security).[2] This property and the small key sizes let us find favor with Fischlin's transformation if the resulting signature scheme is deployed on a smartcard.

We conclude that Fischlin's transformation is a reasonable alternative to the Fiat-Shamir transform, and which transformation to use depends on the actual application.

1.2 Related Work

Tightness. Tight reductions are also discussed in e.g. [27] (for signature schemes based on the family of Diffie-Hellman problems in the random oracle model), in [40, 26, 45] (for the Schnorr and other related signature schemes in the random oracle model), and in [42] (for signature schemes in the plain model).

Leakage. Several leakage models exist so far in the literature. In our work we consider the so-called *bounded leakage* model, where the total amount of leakage is a-priory bounded to some fraction of the secret key length. A more general model is the so-called *continuous leakage model* [14, 8], where the leakage is not a priori bounded and there is some efficient procedure to "refresh" the secret key (leaving the corresponding public key unchanged).

Apart from [3, 32], other papers on leakage-resilient signatures can be found in [19, 15, 20, 7, 38]. These results are either complicated or inefficient, or do not permit optimal relative leakage, and have loose reductions.

2 Preliminaries

2.1 Notation

For $n \in \mathbb{N}$, let $[n] := \{1, 2, \ldots, n\}$. We write log for base-2 logarithms and ln for natural logarithms. We denote vectors by bold lower case letters.

For an algorithm $\mathcal{A}$, $y \leftarrow \mathcal{A}(x)$ denotes that y is output by $\mathcal{A}$ on input x; sometimes we also write $y = \mathcal{A}(x; \omega)$ to make explicit the random coins that $\mathcal{A}$ may use. Also, $\mathcal{A}^{\mathcal{O}}$ denotes that $\mathcal{A}$ has access to oracle $\mathcal{O}$. Algorithm $\mathcal{A}$ is probabilistic polynomial time (PPT) if $\mathcal{A}$ is randomized and for any input $x \in \{0, 1\}^*$ the computation of $\mathcal{A}(x)$ terminates in at most $poly(|x|)$ steps.

[2]The same conclusion does not hold for 128-bit security, though.

The min-entropy of a random variable X is $\mathbb{H}_\infty(X) = -\log \max_x \mathbb{P}[X = x]$, and measures how well X can be predicted by the best (unbounded) predictor. The conditional average min-entropy [16] of X given a random variable Z (over some set $\mathcal{Z}$) possibly dependent on X, is defined as $\widetilde{\mathbb{H}}_\infty(X|Z) := -\log \mathbb{E}_{z \leftarrow Z}[2^{-\mathbb{H}_\infty(X|Z=z)}]$. Following [3], we sometimes rephrase the notion of conditional min-entropy in terms of predictors $\mathcal{A}$ that are given some information Z, so $\widetilde{\mathbb{H}}_\infty(X|Z) = -\log(\max_{\mathcal{A}} \mathbb{P}[\mathcal{A}(Z) = X])$. We recall the following useful lemma, proven in [16], bounding the conditional average min-entropy of a random variable X given λ bits of arbitrary information on X itself.

Lemma 2.1 ([16]) *For all random variables X, Z and Λ over sets $\mathcal{X}$, $\mathcal{Z}$ and $\{0,1\}^\lambda$ such that $\widetilde{\mathbb{H}}_\infty(X|Z) \geq \beta$, we have*

$$\widetilde{\mathbb{H}}_\infty(X|Z, \Lambda) \geq \widetilde{\mathbb{H}}_\infty(X|Z) - \lambda \geq \beta - \lambda.$$

2.2 Signature Schemes

We recall here the general syntax for digital signatures.

Definition 2.2 (Signature scheme) *A signature scheme is a triple of algorithms $\mathcal{SS} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ defined as follows.*

- **Key Generation.** *Algorithm* KGen *is a probabilistic algorithm which, on input a security parameter 1^k, outputs a pair (pk, sk) where pk is the public key, and sk is the secret key.*
- **Signature.** *Algorithm* Sign *is a probabilistic algorithm which, on input a secret key sk together with message m, outputs a signature σ on m under sk.*
- **Verification.** *Algorithm* Vrfy *is a deterministic algorithm which, on input a message m and a signature σ together with the public key pk, outputs either 1 (= valid) or 0 (= invalid).*

We say that $\mathcal{SS}$ has completeness error $\varepsilon_{\text{comp}}$ if $\mathbb{P}[\mathsf{Vrfy}(pk, (m, \mathsf{Sign}(sk, m)) = 0] \leq \varepsilon_{\text{comp}}$, where $(pk, sk) \leftarrow \mathsf{KGen}(1^k)$ and the probability is over the coin tosses of Sign.

Leakage-Resilient Signatures. Consider an oracle $\mathcal{O}_\lambda(x, \cdot)$ taking as input functions $f : \{0,1\}^* \to \{0,1\}^*$ and returning $f(x)$ for a total of at most λ bits. Roughly, a signature scheme $\mathcal{SS} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ is (fully) leakage-resilient if it is hard to forge a signature even given access to oracle $\mathcal{O}_\lambda(x, \cdot)$, where x contains the secret key, plus the entire history of all random coins tossed by the signing algorithm.

More formally, consider the following experiment:

```
Experiment $\mathbf{Exp}^{\text{lkg-ufcma}}_{\mathcal{SS},\mathcal{A}}(k, \lambda)$
    $(pk, sk) \leftarrow \mathsf{KGen}(1^k)$
    $(m^\star, \sigma^\star) \leftarrow \mathcal{A}^{\mathsf{Sign}(sk,\cdot), \mathcal{O}_\lambda(\mathsf{state},\cdot)}(pk)$
    Output 1 iff
        (a) $\mathsf{Vrfy}(pk, m^\star, \sigma^\star) = 1$
        (b) $m^\star \notin \mathcal{Q}$

Set $\mathsf{state} = \{sk\}$, and $\mathcal{Q} = \emptyset$
If $\mathcal{A}$ queries $\mathsf{Sign}(sk, m)$:
    - let $\mathcal{Q} := \mathcal{Q} \cup \{m\}$
    - return $\sigma \leftarrow \mathsf{Sign}(sk, m; \omega)$
    - let $\mathsf{state} := \mathsf{state} \cup \{\omega\}$
```

Definition 2.3 (Fully leakage-resilient signature) *We say that $\mathcal{SS} = (\mathsf{KGen}, \mathsf{Sign}, \mathsf{Vrfy})$ is (t, q_s, ε)-unforgeable against chosen-message attacks (in short: -ufcma) and against λ-leakage attacks if, for every algorithm $\mathcal{A}$ running in time t and asking q_s signing queries, we have:*

$$\mathbb{P}\left[\mathbf{Exp}_{\mathcal{SS},\mathcal{A}}^{\text{lkg}-\text{ufcma}}(k, \lambda) = 1\right] \leq \varepsilon.$$

We say a signature scheme $\mathcal{SS}$ is (t, q_s, ε)-ufcma if the algorithm $\mathcal{A}$ has no access to oracle $\mathcal{O}_\lambda$ in the experiment above.

We remark that in case the signature scheme $\mathcal{SS}$ is defined in the random oracle model (where a public function H modeled as a random oracle is available), then both the adversary $\mathcal{A}$ and the leakage functions have access to the random oracle.

2.3 Fischlin's Transformation

Let $\mathcal{L} \subseteq \mathbf{NP}$ be a language with a (polynomially computable) relation $\mathcal{R} \subset \{0,1\}^* \times \{0,1\}^*$, i.e., $x \in \mathcal{L}$ if and only if $\exists w$ such that $(x, w) \in \mathcal{R}$ and $|w| = \mathit{poly}(|x|)$. The value w is called a witness for $x \in \mathcal{L}$ (x is sometimes called a "theorem" or statement). Informally, $\mathcal{R}$ is called *hard* if, given $x \in \mathcal{L}$, it is hard to extract a valid witness for x.

Definition 2.4 (Hard relation) *A relation $\mathcal{R}$ for a language $\mathcal{L}$ is (t, ε)-hard if the following holds:*

(i) There exists an efficient algorithm Gen *that on input a security parameter k outputs $(x, w) \leftarrow$* $\mathsf{Gen}(1^k)$ *such that $(x, w) \in \mathcal{R}$ and $|w| = \mathit{poly}(|x|)$.*

(ii) For any algorithm $\mathcal{A}$ running in time t we have:

$$\mathbb{P}\left[(x, w') \in \mathcal{R} : \ w' \leftarrow \mathcal{A}(1^k, x); (x, w) \leftarrow \mathsf{Gen}(1^k)\right] \leq \varepsilon.$$

Σ-Protocols. This important class of protocols (run between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$), allows $\mathcal{P}$ to convince $\mathcal{V}$ that it knows a witness w for a shared element $x \in \mathcal{L}$, without giving $\mathcal{V}$ further information. We briefly review Σ-protocols below. Informally, a Σ-protocol consists of three messages $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ (with com sent by $\mathcal{P}$) and satisfies the following properties:

- *Completeness.* Upon interacting with an honest prover holding (x, w), the verifier accepts with overwhelming probability.
- *Special Soundness.* Given accepted proofs $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ and $(\mathsf{com}, \mathsf{ch}', \mathsf{resp}')$ for $x \in \mathcal{L}$ (with $\mathsf{ch}' \neq \mathsf{ch}$), there exists a PPT algorithm which outputs a valid witness w for x.
- *Perfect Honest-Verifier Zero Knowledge (HVZK).* There exists a PPT algorithm $\mathcal{Z}$ (the simulator) which, on input $x \in \mathcal{L}$ and a random ch, outputs an accepting conversation of the form $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$, with the same probability distribution as conversations between the honest $\mathcal{P}$, $\mathcal{V}$ on input x.[3]

In the following we also assume that com has super-logarithmic min-entropy (in the security parameter k), and that resp is quasi-unique, i.e., it is hard to find $(x, \mathsf{com}, \mathsf{ch}, \mathsf{resp}, \mathsf{resp}')$ such that both $(\mathsf{com}, \mathsf{ch}, \mathsf{resp})$ and $(\mathsf{com}, \mathsf{ch}, \mathsf{resp}')$ are accepting, with $\mathsf{resp} \neq \mathsf{resp}'$.

[3]This is also called *special* HVZK, but as argued in [24] can be assumed in general.

Let $\mathcal{R}$ be a hard relation for language $\mathcal{L}$ and $(\mathcal{P}, \mathcal{V})$ be a Σ-protocol for $\mathcal{R}$. For a hash function $H : \{0,1\}^* \to \{0,1\}^b$ (modeled as a random oracle), let r be the number of repetitions, μ the challenge size, and S the bound on the sum.

Key Generation. Upon input security parameter 1^k compute $(x, w) \leftarrow \mathsf{Gen}(1^k)$. Output $pk := x$ as public key and $sk := w$ as secret key.

Signature. Upon input a secret key $sk = w$ and a message $m \in \{0,1\}^*$, perform the following steps:

1. For all $i \in [r]$, obtain $\mathsf{com}_i \leftarrow \mathcal{P}(x)$.
2. For all $i \in [r]$ and $\mathsf{ch}_i \in [2^\mu - 1]$ compute $\mathsf{resp}_i := \mathsf{resp}(\mathsf{ch}_i) \leftarrow \mathcal{P}(\mathsf{com}_i, x, w, \mathsf{ch}_i)$. Denote ch_i which satisfies $H(m, x, \mathbf{com}, i, \mathsf{ch}_i, \mathsf{resp}_i) = 0^b$ by ch_i^*, where $\mathbf{com} = (\mathsf{com}_1, \ldots, \mathsf{com}_r)$. If no such ch_i exists, take the one with minimal hash output value.
3. Output $\sigma = (\mathsf{com}_i, \mathsf{ch}_i^*, \mathsf{resp}_i)_{i=1,\ldots,r}$.

Verification. Upon input the public key $pk = x$ and a signature $\sigma = (\mathsf{com}_i, \mathsf{ch}_i^*, \mathsf{resp}_i)_{i=1,\ldots,r}$ for message m, run the verifier of the underlying Σ-protocol to check if $\mathcal{V}(x, (\mathsf{com}_i, \mathsf{ch}_i^*, \mathsf{resp}_i)) = 1$ for all $i \in [r]$. If not, output 0. Furthermore, if $\sum_{i=1}^r H(m, x, \mathbf{com}, i, \mathsf{ch}_i^*, \mathsf{resp}_i) \leq S$ output 1; else output 0.

Figure 1: Fischlin's transformation applied to a Σ-protocol $(\mathcal{P}, \mathcal{V})$ for relation $\mathcal{R}$

The Transformation. Let $(\mathcal{P}, \mathcal{V})$ be a Σ-protocol for an **NP**-language $\mathcal{L}$ with hard relation $\mathcal{R}$, and consider a hash function $H : \{0,1\}^* \to \{0,1\}^b$, modeled as a random oracle. As proved in [24], Fischlin's transformation (represented in Fig. 1) describes a non-interactive zero-knowledge proof with a straight-line extractor, and yields an existentially unforgeable signature scheme.

Theorem 2.5 (Fischlin's transformation) *Consider the scheme in Fig. 1, where the challenge space of $(\mathcal{P}, \mathcal{V})$ has length $l = O(\log k)$. Let b, r, S, μ be functions of k such that $b \cdot r = \omega(\log k)$, $2^{\mu - b} = \omega(\log k)$, b, r, $\mu = O(\log k)$, $S = O(r)$ and $b \leq \mu \leq l$. The following holds:*

(i) The transformation describes a non-interactive zero-knowledge proof system $(\overline{\mathcal{P}}^H, \overline{\mathcal{V}}^H)$ for language $\mathcal{L}^{msg} = \{((x, m), w) : (x, w) \in \mathcal{R}\}$, where $\overline{\mathcal{P}}$ (resp. $\overline{\mathcal{V}}$) is the signer (resp. verifier) of the signature scheme. In particular, there exists a PPT simulator $\overline{\mathcal{Z}}$ which, on input (x, m) outputs a proof $\sigma = (\mathsf{com}_i, \mathsf{ch}_i^*, \mathsf{resp}_i)_{i=1,\ldots,r}$ with the same distribution as a real proof generated via $\overline{\mathcal{P}}^H$ (using x, m, w).

(ii) There exists a PPT straight-line extractor $\mathcal{K}$ and some value ε_{ext}, such that, for any PPT $\mathcal{A}$, if $(x, \sigma) \leftarrow \mathcal{A}^H(1^k)$, then

$$\mathbb{P}\Big[(x, w) \notin \mathcal{R} \wedge \overline{\mathcal{V}}^H(x, \sigma) = 1\Big] \leq \varepsilon_{ext}$$

for $w \leftarrow \mathcal{K}(x, \sigma, \mathcal{Q}_H(\mathcal{A}))$; here $\mathcal{Q}_H(\mathcal{A})$ denotes $\mathcal{A}$'s queries to (resp. answers from) the random oracle H.

(iii) If the relation $\mathcal{R}$ is (t, ε)-hard, the resulting signature scheme is (t', ε')-ufcma where $t' \approx t$ and $\varepsilon' = \varepsilon + \varepsilon_{ext}$.

Note that the bound on the challenge space is without loss of generality, as for any l we can easily turn a Σ-protocol with l'-bit challenges into a Σ-protocol with l-bit challenges [11, Lemma 2]. The following corollary follows by inspection of the proof of [24, Theorem 2].

Corollary 2.6 (Concrete parameters of Fischlin's transformation) *The following holds for the transformation of Fig. 1:*

- *The completeness error is upper-bounded by*

$$\varepsilon_{\text{comp}} \leq e^{r\ln(e\cdot(2S+1))-(S+1)2^{\mu-b}}.$$

- *The failure probability of the extractor is upper-bounded by*

$$\varepsilon_{\text{ext}} \leq (q_h+1)(S+1)2^{(\log(e\cdot(S+r)/(r-1))-b)\cdot r},$$

  where $q_h = |\mathcal{Q}_H(\mathcal{A})|$.

- *The total number of hash function evaluations is upper-bounded by $r\cdot(2^\mu-1)$ (in worst case).*

3 Leakage Resilience of Fischlin's Transformation

Let $(\mathcal{P},\mathcal{V})$ be a Σ-protocol for $\mathcal{L}\subset\mathbf{NP}$, with relation $\mathcal{R}$. The main result of this section is that Fischlin's transformation applied to each such protocol yields a fully leakage-resilient signature (in the random oracle model) provided that: (i) each theorem $x\in\mathcal{L}$ has exponentially many witnesses w (and given a valid (x,w) pair is hard to find a valid, distinct (x,w') pair); (ii) the conditional min-entropy of the witness W conditioned on the public theorem X is high.

A similar result is already known for the Fiat-Shamir heuristic [3, 32]. The main difference here is that we get a fully tight reduction to the underlying hard problem and relative leakage asymptotically approaching 1—which is optimal. In comparison the best known analysis for Fiat-Shamir has a tigthness gap of roughly $1/\varepsilon$ (where ε is the hardness of the underlying relation), and relative leakage asymptotically approaching 1/2.

We start by formalizing condition (i) above, by introducing the representation problem for a relation $\mathcal{R}\subset\{0,1\}^*\times\{0,1\}^*$.

Definition 3.1 (Representation problem) *We say that the representation problem is (t,ε)-hard for a relation $\mathcal{R}$ if for all PPT adversaries $\mathcal{A}$ running in time t, we have:*

$$\mathbb{P}\Big[w\neq w'\wedge(x,w),(x,w')\in\mathcal{R}:\ (x,w,w')\leftarrow\mathcal{A}(1^k)\Big]\leq\varepsilon.$$

In many significant cases, the hardness of the representation problem for $\mathcal{R}$ is equivalent to the hardness of the underlying relation $\mathcal{R}$. We comment on two such instantiations (one based on the discrete-log assumption and one based on the factoring assumptions) in the paragraph "Concrete Instantiations" at the end of this section.

Theorem 3.2 (Fischlin's transformation is leakage-resilient) *Let $k\in\mathbb{N}$ be a security parameter and let $\mathcal{R}\subset\{0,1\}^*\times\{0,1\}^*$ be an $\mathbf{NP}$-relation such that the representation problem is (t,ε)-hard for $\mathcal{R}$. Assume that conditioned on the distribution of the public input $x\in\mathcal{X}$, the witness*

$w \in \mathcal{W}$ has high average min-entropy β, i.e., $\widetilde{\mathbb{H}}_\infty(W|X) \geq \beta$. Then, the signature scheme of Fig. 1 is $(t', q_s = poly(k), \varepsilon')$-ufcma against λ-leakage attacks, as long as

$$t' \approx t \qquad \lambda \leq \beta - r\log(3q_h) - k \qquad \varepsilon' \leq \varepsilon + \varepsilon_{\text{ext}} + 2^{-k},$$

where $q_h = poly(k)$ denotes the number of queries to the random oracle.

The proof borrows ideas from [32, Theorem 4]. The original proof requires to rewind $\mathcal{A}$, yielding a loose reduction. Intuitively, relying on the straight-line extractor of Fischlin's transformation, we can avoid rewinding and thus get a tight reduction.

Proof. By contradiction assume there exists a PPT adversary $\mathcal{A}$ running in time t' and having advantage $\varepsilon' > \varepsilon + \varepsilon_{\text{ext}} + 2^{-k}$ in the experiment $\mathbf{Exp}^{\text{lkg-ufcma}}_{\mathcal{SS},\mathcal{A}}(k,\lambda)$ (for leakage parameter λ as in the theorem statement). Consider all possible states during the execution of $\mathcal{A}$ in the unforgeability experiment, and for any such state i let h_i denote the hash query made at that state. If an execution of $\mathcal{A}$ terminates with a valid forgery $(m^\star, (\mathsf{com}_i, \mathsf{ch}_i^*, \mathsf{resp}_i)_{i\in[r]})$, we say that the forgery is associated with a set of states $\{h_i\}$ where $h_i = H(m^\star, x, \mathsf{com}, i, \mathsf{ch}_i^*, \mathsf{resp}_i)$ for all $i \in [r]$. Note that the size of this set is $\binom{q_h}{r} \leq (\frac{q_h \cdot e}{r})^r < (3q_h)^r$.

We build a PPT adversary $\mathcal{B}$ (running in time $t \approx t'$) breaking the hardness of the representation problem for $\mathcal{R}$ with advantage larger than ε (a contradiction). Without loss of generality, we assume that whenever $\mathcal{A}$ outputs a forgery $(m^\star, \sigma^\star)$: (i) $\mathcal{A}$ queried the random oracle at some point on input $(m^\star, x, \mathsf{com}, i, \mathsf{ch}_i^*, \mathsf{resp}_i)$, for $i \in [r]$; (ii) $\mathcal{A}$ never queried the signing oracle on $m^\star$. For simplicity, we further assume that every leakage query makes the same number of H evaluations. (This can always be achieved adding dummy queries.)

Adversary $\mathcal{B}$ starts by generating $(x, w) \leftarrow \mathsf{Gen}(1^k)$, where $(x, w) \in \mathcal{R}$. Hence, $\mathcal{B}$ gives the public key $pk = x$ to $\mathcal{A}$ and implicitly defines $sk = w$. Note that since $\mathcal{B}$ knows a valid witness w corresponding to x, the reduction can perfectly simulate the experiment $\mathbf{Exp}^{\text{lkg-ufcma}}_{\mathcal{SS},\mathcal{A}}(k,\lambda)$; this includes the answers to $\mathcal{A}$'s queries to both oracles $\mathsf{Sign}(sk,\cdot)$ and $\mathcal{O}_\lambda(\mathsf{state},\cdot)$, as well as the queries to the random oracle H. Adversary $\mathcal{B}$ keeps also track of all the queries $\mathcal{Q}_H(\mathcal{A})$ of $\mathcal{A}$ to H, and the corresponding answers. Eventually, $\mathcal{A}$ outputs a forgery $(m^\star, \sigma^\star)$. At this point $\mathcal{B}$ runs the straight-line extractor $\mathcal{K}$ on input $(m^\star, \sigma^\star, \mathcal{Q}_H(\mathcal{A}))$ and obtains a value $w' \leftarrow \mathcal{K}(m^\star, \sigma^\star, \mathcal{Q}_H(\mathcal{A}))$. Finally $\mathcal{B}$ outputs (x, w, w') as a solution to the representation problem for $\mathcal{R}$.

By definition, $\mathcal{B}$ solves the representation problem for $\mathcal{R}$ whenever $\mathcal{A}$ succeeds and: (i) the extractor $\mathcal{K}$ does not fail; (ii) the extracted witness w' is different from w. Denote by Fail the event that the extractor does not return a valid witness and with Equal the event that the returned w' is equal to w. Since the event that $\mathcal{A}$ wins and the latter two events are all independent, we can write:

$$\begin{aligned}
\mathbb{P}[\mathcal{B} \text{ wins}] &= \mathbb{P}\Big[w' \neq w; (x,w),(x,w') \in \mathcal{R} :\ (x,w,w') \leftarrow \mathcal{B}(1^k)\Big] \\
&= \mathbb{P}[\mathcal{A} \text{ wins} \wedge \neg\textsc{Fail} \wedge \neg\textsc{Equal}] \\
&= \mathbb{P}[\mathcal{A} \text{ wins}] \cdot \mathbb{P}[\neg\textsc{Fail}] \cdot \mathbb{P}[\neg\textsc{Equal}] \\
&\geq \mathbb{P}[\mathcal{A} \text{ wins}] - \mathbb{P}[\textsc{Fail}] - \mathbb{P}[\textsc{Equal}] \\
&\geq \varepsilon' - \varepsilon_{\text{ext}} - \mathbb{P}[\textsc{Equal}], \qquad (1)
\end{aligned}$$

where (1) follows by our assumption on $\mathcal{A}$ and by the fact that the probability that the extractor fails is bounded by ε_{ext}.

Claim 3.3 $\mathbb{P}[\text{EQUAL}] \leq 2^{-k}$.

Proof (of claim). We show that the statement holds even in case $\mathcal{A}$ is unbounded. We will argue that $\widetilde{\mathbb{H}}_\infty(W|V) \geq k$, where W is the random variable corresponding to the witness, and V is the random variable corresponding to the view of $\mathcal{A}$ in a run of $\mathbf{Exp}^{\text{lkg-ufcma}}_{\mathcal{SS},\mathcal{A}}(k,\lambda)$. Clearly, this is sufficient as by definition of average min-entropy $\mathbb{P}[\text{EQUAL}] \leq 2^{-\widetilde{\mathbb{H}}_\infty(W|V)}$.

Notice that the view of $\mathcal{A}$ has a type $V := (\mathbf{\Sigma}, \mathbf{R}, \Lambda, X)$, where the random variable $\mathbf{\Sigma} = (\Sigma_1, \ldots, \Sigma_{q_s})$ corresponds to the signing queries of $\mathcal{A}$, the random variable $\mathbf{R} = (R_1, \ldots, R_{q_h})$ corresponds to the responses to $\mathcal{A}$'s random oracle queries, Λ corresponds to the leakage queries, and X corresponds to the public key. Now,

$$\begin{aligned}\widetilde{\mathbb{H}}_\infty(W|\mathbf{\Sigma}, \mathbf{R}, \Lambda, X) &\geq \widetilde{\mathbb{H}}_\infty(W|\mathbf{\Sigma}, X) - \lambda - r\log(3q_h) \qquad (2)\\ &\geq \widetilde{\mathbb{H}}_\infty(W|X) - \lambda - r\log(3q_h) \qquad (3)\\ &\geq k. \qquad (4)\end{aligned}$$

(2) follows by Lemma 2.1 and the fact that: (i) $\Lambda \in \{0,1\}^\lambda$ and (ii) a forgery reveals at most $r\log(3q_h)$ bits of information on the witness, corresponding to the set of random oracle queries associated with the forgery itself;[4] (3) follows by (perfect) honest-verifier zero-knowledge, as we can compute each Σ_i using only the public key X and the zero-knowledge simulator $\overline{\mathcal{Z}}$ (cf. Theorem 2.5). Finally, (4) follows by our assumption that $\widetilde{\mathbb{H}}_\infty(W|X) \geq \beta$ and the bound on $\lambda \leq \beta - r\log(3q_h) - k$.

The above claim together with our assumption that $\varepsilon' > \varepsilon + \varepsilon_{\text{ext}} + 2^{-k}$, clearly imply that $\mathbb{P}[\mathcal{B} \text{ wins}] > \varepsilon$, which contradicts the (t,ε)-hardness of the representation problem for $\mathcal{R}$. This finishes the proof.

Concrete Instantiations. Below, we discuss two concrete instantiations of Theorem 3.2, the first one based on the discrete-log assumption and the second one based on the RSA assumption (and on factoring).

Generalized Okamoto [39]. For a cyclic group $\mathbb{G}$ of prime order p, let $\mathcal{L}_{\text{DL}} := \{(g_1, \ldots, g_\ell, h) : \exists(w_1, \ldots, w_\ell) \text{ s.t. } h = \prod_{i=1}^{\ell} g_i^{w_i}\}$, where $(g_1, \ldots, g_\ell)$ are generators of $\mathbb{G}$. The tuple $w = (w_1, \ldots, w_\ell)$ is called a representation of h; the ℓ-representation problem asks to compute two distinct representations w, w' for some $x = (g_1, \ldots, g_\ell, h) \in \mathcal{L}$. As argued in [3, Lemma 4.1], the ℓ-representation problem is hard for $\mathcal{R}_{\text{DL}}$ if and only if the discrete-log problem is hard in $\mathbb{G}$.[5]

The standard Σ-protocol $(\mathcal{P}, \mathcal{V})$ to prove knowledge of a representation of an element h goes as follows: (i) $\mathcal{P}$ chooses randomly $a_1, \ldots, a_\ell$ and sets $\mathsf{com} := \prod_{i=1}^{\ell} g_i^{a_i}$; (ii) $\mathcal{V}$ selects a random $\mathsf{ch} \in \mathbb{Z}_p$; (iii) $\mathcal{P}$ returns $\mathbf{resp} = (\mathsf{ch}\cdot w_1 + a_1, \ldots, \mathsf{ch}\cdot w_\ell + a_\ell)$. Given a proof $\sigma = (\mathsf{com}, \mathsf{ch}, \mathbf{resp})$, the verifier outputs 1 if and only if $\prod_{i=1}^{\ell} g_i^{\mathsf{resp}_i} = h^{\mathsf{ch}} \cdot \mathsf{com}$.

We obtain the following result:

[4] Recall that leakage queries can depend on the random oracle; this could make the set of states associated with a forgery a function of the witness.

[5] Recall that the discrete-log problem requires to compute w such that $g^w = h$, given $(g, h, \mathbb{G}, p)$.

Corollary 3.4 *Let $\mathbb{G}$ be a cyclic group of prime order p, such that the ℓ-representation problem is hard for $\mathcal{R}_{\mathrm{DL}}$. Then, the signature scheme obtained by applying Fischlin's transformation to the generalized Okamato Σ-protocol is fully leakage-resilient for leakage parameter $\lambda \leq (1-o(1)) \cdot n$, where $n = \ell \log p$ is the length of the secret key.*

Proof. By Theorem 3.2, we get that for any desired $\delta > 0$ the leakage bound is $\lambda \leq (1-1/\ell-\delta) \cdot n$. Now, choosing $\ell > 1/\delta$ gives $\lambda \leq (1-\delta) \cdot n$ as desired.

Generalized Guillou-Quisquater [30]. For $N = p \cdot q$, where p and q are primes, let (e,d) be such that $e \cdot d = 1 \bmod \phi(N)$ and e is a prime. Consider the language $\mathcal{L}_{\mathrm{RSA}} := \{(g_1, \ldots, g_\ell, h) : \exists(\rho, (w_1, \ldots, w_\ell)) \in \mathbb{Z}_N^* \times \mathbb{Z}_e^\ell \text{ s.t. } h = \prod_{i=1}^{\ell} g_i^{w_i} \cdot \rho^e \bmod N\}$, where $(g_1, \ldots, g_\ell)$ are generators of $\mathbb{Z}_N^*$. The tuple $w = (\rho, (w_1, \ldots, w_\ell))$ is called a representation of h; the ℓ-representation problem asks to compute two distinct representations w, w' for some $x = (g_1, \ldots, g_\ell, h) \in \mathcal{L}$. As shown in [39], the ℓ-representation problem is hard for $\mathcal{R}_{\mathrm{RSA}}$ if and only if the RSA problem is hard in $\mathbb{Z}_N^*$.[6]

The standard Σ-protocol $(\mathcal{P}, \mathcal{V})$ to prove knowledge of a representation of an element h goes as follows: (i) $\mathcal{P}$ chooses randomly $a_1, \ldots, a_\ell \leftarrow \mathbb{Z}_e$ and $s \leftarrow \mathbb{Z}_N^*$ and sets $\mathsf{com} := \prod_{i=1}^{\ell} g_i^{a_i} \cdot s^e \bmod N$; (ii) $\mathcal{V}$ selects a random $\mathsf{ch} \in \mathbb{Z}_e$; (iii) $\mathcal{P}$ computes $\mathbf{z} = (\mathsf{ch} \cdot w_1 + a_1, \ldots, \mathsf{ch} \cdot w_\ell + a_\ell)$, $u = (s \cdot \rho^{\mathsf{ch}}) \bmod N$ and returns $\mathbf{resp} = (\mathbf{z}, u)$. Given a proof $\sigma = (\mathsf{com}, \mathsf{ch}, \mathbf{resp})$, the verifier outputs 1 if and only if $u^e \cdot \prod_{i=1}^{\ell} g_i^{z_i} = h^{\mathsf{ch}} \cdot \mathsf{com} \bmod N$.

We obtain the following result:

Corollary 3.5 *Let (N, e) be such that the ℓ-representation problem is hard for $\mathcal{R}_{\mathrm{RSA}}$. Then, the signature scheme obtained by applying Fischlin's transformation to the generalized Guillou-Quisquater Σ-protocol is fully leakage-resilient for leakage parameter $\lambda \leq (1-o(1)) \cdot n$, where $n = \ell \log(e) + \log \phi(N)$ is the length of the secret key.*

A similar statement can be obtained based on factoring, following Fischlin and Fischlin [25].

4 Comparison

In this section we investigate the efficiency of signature schemes obtained via the Fiat-Shamir transform and Fischlin's transformation. We do so by comparing the performance of the two paradigms for an implementation of the signature scheme resulting from the Generalized Okamoto scheme [39] (see Section 3). Our implementation was carried out in the Charm cryptographic framework [2] in `Python`. The experiments were performed on a single core of a 3 GHz Intel Core i7. We instantiated the random oracle by `SHA-2`.

In our implementation of Fischlin's transformation we do not impose an upper bound on the challenge size (i.e., $\mu = \infty$). As a consequence we do not require a threshold S because given $b \cdot r = \omega(\log k)$ the probability of finding appropriate challenges ch_i mapping $H(m, x, \mathbf{com}, i, \mathsf{ch}_i, \mathsf{resp}_i)$ to 0^b is negligibly close to 1 in the security parameter k. Having no threshold in the loop-clause yields to a signature scheme with expected (rather than strict) polynomial running time. Nonetheless,

[6]Recall that the RSA problem requires to compute ρ such that $\rho^e = u \bmod N$, given (u, N, e).

the running times of the original version and ours do not differ noticeably. Our experiments have shown, in fact, that the full span of challenge candidates was rarely explored.

We stress that our implementations are not optimized. In particular, we expect faster running times when carrying out the implementation in `C/C#`. However, a prototype implementation in `Python` gives still reasonable timings for the purpose of comparing the two transformations.

4.1 Parameter Selection

In order to select reasonable parameters for the two signature schemes we have to assess the hardness of the underlying hardness assumption and take the tightness gap of the reduction into account. The security of the Generalized Okamoto scheme relies on the hardness of the representation problem in a group $\mathbb{G}$ of prime order p, which is equivalent to the discrete logarithm problem [3, Lemma 4.1]. Here, we focus on the case where $\mathbb{G} = GF(p)$ is a Galois fields. More precisely, we take the multiplicative group $\mathbb{Z}_p^*$ where p is a safe prime, i.e. $p = 2q + 1$, and p and q are both prime.

The best known algorithm to solve the discrete logarithm problem in $\mathbb{G} = GF(p)$ is the Number Field Sieve (NFS), with complexity $L_p[1/3, (64/9)^{1/3}]$ for modulus p, where the complexity function $L_p[t, s]$ is defined as $L_p[t, s] = e^{s(1+o(1))(\ln p)^t (\ln \ln p)^{1-t}}$. When estimating security parameters we take previously known attacks and timings into account by saying that if computing discrete logarithms in groups of order p takes time t, than we expect that computing discrete logarithms in groups of order p' takes time roughly $t' \approx t \frac{L_p[1/3, \sqrt[3]{64/9}]}{L_{p'}[1/3, \sqrt[3]{64/9}]}$. If the difference between p' and p is not too large, the term $o(1)$ goes to zero. A similar strategy was recommended in [36]. We take as reference the 2009 factorization of a 768-bit modulus, which offers roughly 66-bit security ($t \approx 2^{66}$).

Let us now consider both schemes with their respective security reduction. If an adversary $\mathcal{A}$ (t', ε')-breaks the signature scheme obtained via Fiat-Shamir (see [3, 32]), there exists an adversary $\mathcal{B}$ with runtime $\approx t'$ that solves the ℓ-representation problem in $\mathbb{G}$ with probability $(\varepsilon')^2/q_h$, where q_h is the number of queries to the random oracle H. Thus, we need to run $\mathcal{B}$ around $O(q_h/(\varepsilon')^2)$, yielding a runtime $t \approx t' \cdot q_h/(\varepsilon')^2$. For ε' large enough (say $\varepsilon' > 0.1$) and for $q_h \approx t'$, we have $t \approx 2^{165}$ for 80-bit security. Thus, the parameters must be chosen such that computing the discrete logarithm in $\mathbb{G}$ with NFS takes time roughly 2^{165}. This holds for a prime p of roughly 5400 bits. Analogously, for 128-bits of security one needs a prime p of roughly 15000 bits.

We compare these results to Fischlin's scheme. Since the reduction is tight ($\varepsilon \approx \varepsilon' - \varepsilon_{ext} - 2^{-k}$), an adversary $\mathcal{B}$ solves the ℓ-representation problem in $\mathbb{G}$ in time $t \approx t' \cdot (\varepsilon' - \varepsilon_{ext} - 2^{-k})^{-1}$. Recall that ε_{ext} is the extractor's success probability to extract the witness in the security reduction. We have to set parameters r and b such that $\varepsilon_{ext} \leq q_h \cdot 2^{-br}$ is smaller than the advantage of solving the representation problem ε.[7] We can choose a 1130-bit prime p for $\mathbb{G} = GF(p)$ for 80-bit security and 3048-bit prime for 128-bit security, respectively.

In the following we compare both signature schemes in terms of performance and signature size. We stress that for some signature schemes obtained via Fiat-Shamir or Fischlin (e.g., the Schnorr signature scheme [43]), signatures can be shortened by removing the commitment(s) from the signature because the commitment is re-computable from the challenge and response alone. (This holds in particular for the signature derived from the Generalized Okamoto scheme.) Given the above system parameters, a signature computed via Fiat-Shamir consists of one hash value (the challenge, from $\mathbb{Z}_p^*$), and ℓ elements from $\mathbb{Z}_p^*$ (the response), yielding a signature of size 16200 bits

[7]We derive the bound $\varepsilon_{ext} \leq q_h \cdot 2^{-br}$ by adapting the proof of Corollary 2.6 to the proposed modifications of the scheme, as described above.

	80-bit security				128-bit security			
	FS	Fischlin $r=7$ $b=12$	Fischlin $r=14$ $b=6$	Fischlin $r=6$ $b=14$	FS	Fischlin $r=7$ $b=19$	Fischlin $r=19$ $b=7$	Fischlin $r=11$ $b=12$
Signing time (in sec)	0.463	1.037	**0.103**	3.531	5.3	290.262	**1.889**	4.715
Verification (in sec)	1.16	**0.060**	0.117	0.062	30.89	**0.993**	2.552	1.451
Signature size (in kB)	1.98	1.94	3.87	**1.67**	5.49	**5.22**	14.15	8.2
Public-key size (in kB)	1.98	**0.41**	**0.41**	**0.41**	5.49	**1.12**	**1.12**	**1.12**
Secret-key size (in kB)	1.32	**0.28**	**0.28**	**0.28**	3.66	**0.37**	**0.37**	**0.37**

Table 1: Comparison between Fiat-Shamir (FS) and Fischlin for the Generalized Okamoto signature scheme. The table shows performance and sizes for $\ell = 2$.

for 80-bit security (resp. 45000 bits for 128-bit security) and $\ell = 2$. On the other hand, a signature computed via Fischlin consists of r challenge values of expected size b bits, and $r \cdot \ell$ elements from $\mathbb{Z}_p^*$. We obtain comparable signature sizes with $r = 7$ for both 80 and 128 bits of security.

In Table 1 and Figures 2 and 3 we illustrate the performances of both schemes in terms of key generation, signing time, verification time, and leakage resilience. The result of the comparison are discussed in the following subsections.

4.2 On Key Generation

The key generation algorithm is exactly the same for both schemes. However, due to the loose reduction of the Fiat-Shamir transform, the resulting signature scheme requires much larger groups than the one derived via Fischlin's transformation.

Recall that, in order to resist special discrete logarithm solvers, we have to instantiate the groups in $\mathbb{Z}_p^*$ where p is a safe prime, i.e., $p = 2q + 1$ with p, q prime. Finding safe primes is an expensive task. Especially, signatures derived via Fiat-Shamir require to sample a safe prime p of size 15000 which may take several days or weeks.[8] Even finding a safe prime of 5400 bits (for 80-bit security) took us roughly 12 hours using the safe prime generator of OpenSSL.

As a consequence Fischlin's transformation is preferable when it comes to key generation in both performance and size. Both public and secret-key size of Fischlin's scheme are roughly 80% shorter than the ones in Fiat-Shamir.

4.3 On Signature Generation

For what concerns the signing time, we paid attention to perform the comparison between the two schemes as fair as possible. In particular group operations were implemented in the same way, and the experiments were run on the same machine using the same crypto libraries. As recommended by Fischlin [24], we enhanced the computation of the hash function in Fischlin's scheme by pre-computing and saving part of the hash values $H(m, x, \mathbf{com}, \cdot, \cdot, \cdot)$ (since this part is fixed throughout all loops).[9] We also note that a clever implementation of Fischlin's scheme, requires just additions

[8]For this reason we took a slightly larger safe prime p of size 16384. More precisely we took the publicly available prime $p = 2^{16384} - 364486013$ [5].

[9]We note that doing so, we distance our implementation from the random-oracle proof.

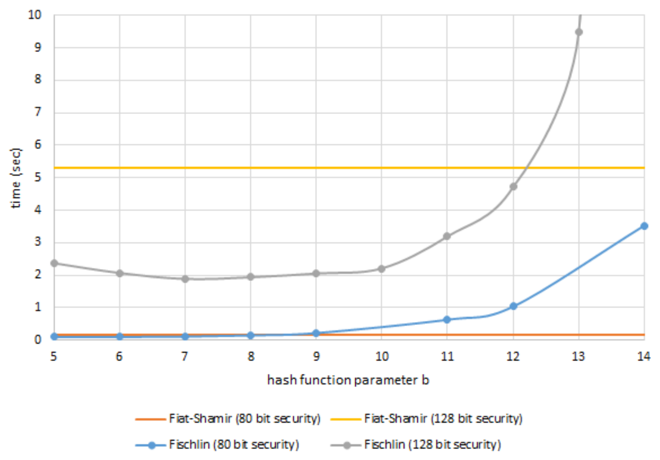

Figure 2: Runtime of the signature algorithm for the Fiat-Shamir and Fischlin transformation.

in the group $\mathbb{G}$ when searching for an appropriate challenge (see step 2. in Fig. 1). Here, one can re-use the previously computed response resp instead of computing it from scratch each time in the loop.

Fischlin's transformation offers a trade-off between performance and signature size. In Fig. 2 we illustrate the running time of the signature algorithm for different choices of the parameters r and b, and compare it with the case of signatures obtained via Fiat-Shamir. We observe that for similar signature sizes and 80-bit security, generating Fiat-Shamir signatures is twice as fast as generating signatures via Fischlin. For 128-bit security, the situation is even worse (Fischlin is roughly 58 times slower). Nonetheless, we see that if one is flexible with respect to the signature size, Fischlin's transformation yields a signing time which is up to 4.5 times faster than the one for Fiat-Shamir, while the signature size only doubles.

The signing time for Fischlin's transformation increases rapidly from a certain threshold $b = b^*$.[10] For instance, if $b = 14$ and we stick to 80-bit security, the signing algorithm takes time roughly 3.5 seconds. Taking 128-bit security, any $b > 13$ leads to huge signing time. For $b = 13$, a signature requires 9.47 seconds. The signing time for Fischlin and Fiat-Shamir becomes of the same magnitude for $r = 11$ and $b = 12$ (128-bit security).

4.4 On Verification

Recall that, as mentioned in Section 4.1, we implemented the compressed form of the Generalized Okamoto scheme. That is, signatures consist of $(\mathsf{ch}, \mathbf{resp})$ for the Fiat-Shamir transform and $(\mathsf{ch}_i, \mathbf{resp}_i)_{1 \le i \le r}$ for Fischlin's transformation. Consequently, the verification algorithm first has to reconstruct the commitment (vector) and then check for the validity of the response(s).

Fig. 3 shows the running time of the verification algorithm for the two schemes. Interestingly, verifying signatures obtained via Fischlin's transformation is significantly faster than verifying

[10]We believe that the value for the threshold b^* is dependent on the implementation and programming language. For example, for certain parameters, the time needed to compute all hash values exceeds the time needed to perform all necessary group operations. If the gap between the former and the latter changes (e.g., due to a different programming language), one might expect that the threshold b^* shifts, eventually leading to a different outcome for the comparison of signing time.

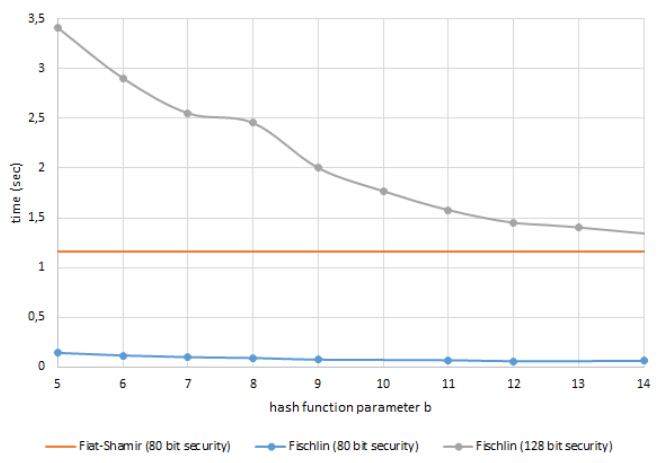

Figure 3: Runtime of the verification algorithm for the Fiat-Shamir and Fischlin transformation.

signatures obtained via Fiat-Shamir. For comparable signature sizes, Fischlin's transformation yields a verification time which is roughly 19 times faster than Fiat-Shamir for 80-bit security, and roughly 30 times faster for 128-bit security. The latter choice of parameters, however, leads to high signing times (209.262 seconds). For comparable signing times, Fischlin's transformation offers still 21 times faster verification with 37% signature size increase.

4.5 On Leakage Resilience

As discussed in Section 3 (see Corollary 3.4), our analysis of Fischlin's transformation applied to the Generalized Okamoto scheme yields relative leakage asymptotically approaching 1. (This is in contrast to [3, 32], where the relative leakage is always smaller than 1/2.)

In Table 2 one can find the time for running the signing algorithm when considering different values for the desired tolerated leakage λ. Note that to allow higher relative leakage, the parameter ℓ must be increased. This generates a linear blow-up in both the running times and signature size. However, if we set parameters b, r, ℓ such that signatures resulting by applying Fischlin's transformation match Fiat-Shamir in terms of signature size and amount of tolerated leakage, the signing time for Fischlin's transform decreases considerably. This is because the parameter ℓ can be fixed to 2 if we want to tolerate relative leakage less than 1/2. Thus, one can choose larger r—and smaller b—to meet the signature size of Fiat-Shamir, and end-up with an efficiency gain.

In conclusion, when we take leakage resilience into account, Fischlin's transformation outperforms the Fiat-Shamir transform in signing time, verification time and signature size.

5 Discussion

We have shown that Fischlin's transformation is a viable approach to get fully leakage-resilient signatures with a tight reduction to the underlying hard problem, and asymptotically optimal relative leakage (in the random oracle model). This is in contrast to the situation for signatures obtained via Fiat-Shamir (having a non-tight reduction and non-optimal relative leakage). When one takes into account this gap (as demonstrated by our implementation), it becomes evident that,

Signature running time (in sec)	80-bit security			
	$\lambda \leq 1/4$	$\lambda \leq 3/8$	$\lambda \leq 7/16$	$\lambda \leq 3/4$
	$\ell = 2$	$\ell = 4$	$\ell = 8$	—
	$\ell' = 2$	$\ell' = 2$	$\ell' = 2$	$\ell' = 4$
	$\|\sigma\| \approx 1.98$	$\|\sigma\| \approx 3.3$	$\|\sigma\| \approx 5.93$	$\|\sigma\| \approx 5.52$
Fiat-Shamir (with ℓ)	0.463	0.951	1.858	—
Fischlin (with ℓ')	1.037	0.114	0.103*	0.287

Table 2: Comparison of Fischlin's transformation and the Fiat-Shamir transform for the Generalized Okamoto signature scheme, with different leakage parameter λ. Fischlin is instantiated with r and b such that the resulting signature size is comparable in both schemes. For the timing (*) we selected the fastest parameters r, b where the resulting signature size is even smaller.

in many important scenarios, Fischlin's transformation might be preferable (or at least comparable) to the Fiat-Shamir transformation in terms of verification time, signing time and key generation.

We conclude with a number of remarks on our results.

- Our model of leakage resilience assumes that the overall amount of leakage is bounded by some a-priori fixed parameter. Using the techniques of [3, 1], our results extend to continuous leakage resilience in the so-called "Floppy model" (see also [12]), where one assumes the existence of a (leakage free) floppy that can be used to update the secret key (leaving the corresponding public key unchanged).
- Our analysis can be extended to the context of memory tampering (see, e.g., [18, 21, 22] and references therein). In particular Theorem 3.2 can be generalized to the setting of bounded leakage and tamper resilience [12], where the adversary is not only allowed to leak from the state of the signer but can also inject (an a-priori bounded number of) faults into the secret key and then obtain access to a "faulty" signing oracle.
- We remark that in general Fischlin's transformation can be preferable to Fiat-Shamir in all settings where simulation extractability is needed [20].

Furthermore, we ask ourselves how secure signature schemes are against quantum adversaries if the underlying identification scheme is quantum immune. More precisely, does the Fischlin's transformation yield secure signature schemes in the quantum random oracle model [6]. Dagdelen et al. [10] have shown that this does not always hold for the Fiat-Shamir transformation.

Acknowledgments

Özgür Dagdelen is supported by the German Federal Ministry of Education and Research (BMBF) within EC-SPRIDE.

References

[1] Shweta Agrawal, Yevgeniy Dodis, Vinod Vaikuntanathan, and Daniel Wichs. On continual leakage of discrete log representations. In *ASIACRYPT (2)*, pages 401–420, 2013.

[2] Joseph A. Akinyele, Christina Garman, Ian Miers, Matthew W. Pagano, Michael Rushanan, Matthew Green, and Aviel D. Rubin. Charm: a framework for rapidly prototyping cryptosystems. *Journal of Cryptographic Engineering*, 3(2):111–128, 2013.

[3] Joël Alwen, Yevgeniy Dodis, and Daniel Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. In *CRYPTO*, pages 36–54, 2009.

[4] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[5] Ken's blog. Mostly on computers and mathematics. Website Blog. `http://kenta.blogspot.de/2011/01/cvogqzhd-some-large-safe-primes.html` Last acess: 31.01.2013.

[6] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In *ASIACRYPT*, pages 41–69, 2011.

[7] Elette Boyle, Gil Segev, and Daniel Wichs. Fully leakage-resilient signatures. *J. Cryptology*, 26(3):513–558, 2013.

[8] Zvika Brakerski, Yael Tauman Kalai, Jonathan Katz, and Vinod Vaikuntanathan. Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage. In *FOCS*, pages 501–510, 2010.

[9] Sanjit Chatterjee, Alfred Menezes, and Palash Sarkar. Another look at tightness. *IACR Cryptology ePrint Archive*, 2011:442, 2011.

[10] Özgür Dagdelen, Marc Fischlin, and Tommaso Gagliardoni. The Fiat-Shamir transformation in a quantum world. In *ASIACRYPT (2)*, pages 62–81, 2013.

[11] Ivan Damgård. On Σ-protocols. Technical report, Aarhus University, 2013. Available at `www.daimi.au.dk/~ivan/Sigma.pdf`.

[12] Ivan Damgård, Sebastian Faust, Pratyay Mukherjee, and Daniele Venturi. Bounded tamper resilience: How to go beyond the algebraic barrier. In *ASIACRYPT (2)*, pages 140–160, 2013.

[13] Francesco Davì, Stefan Dziembowski, and Daniele Venturi. Leakage-resilient storage. In *SCN*, pages 121–137, 2010.

[14] Yevgeniy Dodis, Kristiyan Haralambiev, Adriana López-Alt, and Daniel Wichs. Cryptography against continuous memory attacks. In *FOCS*, pages 511–520, 2010.

[15] Yevgeniy Dodis, Kristiyan Haralambiev, Adriana López-Alt, and Daniel Wichs. Efficient public-key cryptography in the presence of key leakage. In *ASIACRYPT*, pages 613–631, 2010.

[16] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.

[17] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *FOCS*, pages 293–302, 2008.

[18] Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-malleable codes. In *ICS*, pages 434–452, 2010.

[19] Sebastian Faust, Eike Kiltz, Krzysztof Pietrzak, and Guy N. Rothblum. Leakage-resilient signatures. In *TCC*, pages 343–360, 2010.

[20] Sebastian Faust, Markulf Kohlweiss, Giorgia Azzurra Marson, and Daniele Venturi. On the non-malleability of the Fiat-Shamir transform. In *INDOCRYPT*, pages 60–79, 2012.

[21] Sebastian Faust, Pratyay Mukherjee, Jesper Buus Nielsen, and Daniele Venturi. Continuous non-malleable codes. In *TCC*, pages 465–488, 2014.

[22] Sebastian Faust, Pratyay Mukherjee, Daniele Venturi, and Daniel Wichs. Efficient non-malleable codes and key-derivation for poly-size tampering circuits. In *Eurocrypt*, 2014.

[23] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, pages 186–194, 1986.

[24] Marc Fischlin. Communication-efficient non-interactive proofs of knowledge with online extractors. In *CRYPTO*, pages 152–168, 2005.

[25] Marc Fischlin and Roger Fischlin. The representation problem based on factoring. In *CT-RSA*, pages 96–113, 2002.

[26] Sanjam Garg, Raghav Bhaskar, and Satyanarayana V. Lokam. Improved bounds on security reductions for discrete log based signatures. In *CRYPTO*, pages 93–107, 2008.

[27] Eu-Jin Goh, Stanislaw Jarecki, Jonathan Katz, and Nan Wang. Efficient signature schemes with tight reductions to the Diffie-Hellman problems. *J. Cryptology*, 20(4):493–514, 2007.

[28] Oded Goldreich. On the foundations of modern cryptography. In *CRYPTO*, pages 46–74, 1997.

[29] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

[30] Louis C. Guillou and Jean-Jacques Quisquater. A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In *CRYPTO*, pages 216–231, 1988.

[31] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In *CRYPTO*, pages 463–481, 2003.

[32] Jonathan Katz and Vinod Vaikuntanathan. Signature schemes with bounded leakage resilience. In *ASIACRYPT*, pages 703–720, 2009.

[33] Neal Koblitz and Alfred Menezes. Another look at "provable security". *J. Cryptology*, 20(1):3–37, 2007.

[34] Paul C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In *CRYPTO*, pages 104–113, 1996.

[35] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *CRYPTO*, pages 388–397, 1999.

[36] Arjen K. Lenstra and Eric R. Verheul. Selecting cryptographic key sizes. In Hideki Imai and Yuliang Zheng, editors, *Public Key Cryptography*, volume 1751 of *Lecture Notes in Computer Science*, pages 446–465. Springer Berlin Heidelberg, 2000.

[37] Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In *TCC*, pages 278–296, 2004.

[38] Jesper Buus Nielsen, Daniele Venturi, and Angela Zottarel. Leakage-resilient signatures with graceful degradation. In *Public Key Cryptography*, pages 362–379, 2014.

[39] Tatsuaki Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In *CRYPTO*, pages 31–53, 1992.

[40] Pascal Paillier and Damien Vergnaud. Discrete-log-based signatures may not be equivalent to discrete log. In *ASIACRYPT*, pages 1–20, 2005.

[41] Jean-Jacques Quisquater and David Samyde. Electromagnetic analysis (EMA): Measures and counter-measures for smart cards. In *E-smart*, pages 200–210, 2001.

[42] Sven Schäge. Tight proofs for signature schemes without random oracles. In *EUROCRYPT*, pages 189–206, 2011.

[43] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In *Advances in CryptologyCrypto89 Proceedings*, pages 239–252. Springer, 1990.

[44] Torsten Schütze. Automotive security: Cryptography for car2x communication. In *Embedded World Conference*, 2011.

[45] Yannick Seurin. On the exact security of schnorr-type signatures in the random oracle model. In *EUROCRYPT*, pages 554–571, 2012.