NOTES ON
On the (in)security of the Fiat-Shamir paradigm

Shafi Goldwasser and Yael Tauman Kalai

May 10, 2005

1 Recap

1.1 Background

The Fiat-Shamir transform, which we'll refer to as the 'FS transform' from now on, is a method for converting 3-round public-coin (or *canonical*) id schemes into signature schemes using any efficiently evaluable hash function ensemble $\mathcal{H} = \{\mathcal{H}_n\}_{n\in\mathbb{N}}$, $\mathcal{H}_n = \{h_s : \{0,1\}^* \to \{0,1\}^n\}_{s\in\{0,1\}^{n^2}}$. Although here we insist on $|s| = n^2$, any polynomial $p(n)$ would have done just as well. Also, when we say $\mathcal{H}$ is 'efficiently evaluable', all we mean is that there is a function H s.t. $H(s,m) = h_s(m)$ for all $s, m \in \{0,1\}^*$, and H is computed by some deterministic TM $\mathcal{M}_H$ whose running time is polynomial in $|s|$ and $|m|$. The FS transform was first introduced, albeit implicitly, in [FS87].

Recall that a three-round public-coin id scheme $ID = (G, P, V)$ is simply a three-round id scheme where the verifier's move consists of his random bits, or 'coins'. In other words, the prover P first sends the verifier V a message α, which we think of as a kind of 'commitment', then V sends a random challenge β to P, and finally P responds with γ (see Figure 1).

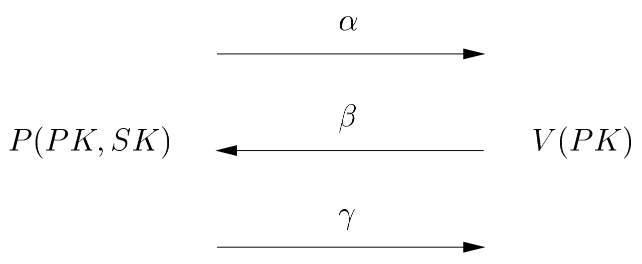

Figure 1: A three-round public-coin id scheme $ID = (G, P, V)$

Denote the signature scheme resulting from applying the FS transform to ID with respect to $\mathcal{H}$ by $SIG_H(ID) = (GEN_{\mathcal{H}}, SIGN_{\mathcal{H}}, VER_{\mathcal{H}})$.

On input 1^n (and random bits), $GEN_{\mathcal{H}}$ outputs $((PK, k), SK)$, where $(PK, SK) \leftarrow G(1^n)$ and $k \in_R \{0,1\}^{n^2}$. In order to sign a message m, one must produce a transcript $\sigma_m = (\alpha, \beta, \gamma)$ such that $\beta = h_k(\alpha, m)$ and $V(PK, \sigma_m) = 1$. $SIGN_{\mathcal{H}}(PK, k, SK, m)$ can easily come up with such a σ_m by simulating $P(PK, SK, h_k(\alpha, m))$, where α is just the first message $P(PK, SK)$ would normally send to $V(PK)$.

Ideally, we'd like the FS transform to *preserve security*: given **any** secure canonical id scheme ID, there should exist **some** ensemble $\mathcal{H}$ such that $SIG_H(ID)$ is a secure signature scheme. It was shown in [**?**] that for any such ID, the "FS-type" signature scheme one obtains by using a *truly random function* $\mathcal{O}$ instead of an ensemble $\mathcal{H}$ is secure against probabilistic polytime adversaries which have oracle access to $\mathcal{O}$. In other words, $SIG_H(ID)$ is secure *in the random oracle model.* This is taken to mean that the scheme has "no structural flaws", since it's hard to conceive of a real-world attack which wouldn't also work in the random oracle model.

However, security in the random model does not in fact imply real-world security: Canetti, Goldreich, and Halevi have exhibited a (highly contrived) signature scheme in [CGH98] which is secure in the random oracle model, yet insecure no matter what ensemble $\mathcal{H}$ the random oracle $\mathcal{O}$ is replaced with. Nevertheless, it was believed that signature schemes obtained from secure canonical id schemes via the FS transform may still be secure in the real world, since they are less "perverse" than the unrealistic scheme constructed by Canetti et al.

Goldwasser and Tauman Kalai recently showed in [GTK03] that the FS transform does **not** in fact preserve security. In other words, there exists **some** secure canonical id scheme ID such that $SIG_H(ID)$ is insecure for **every** $\mathcal{H}$. Although ID is still quite "perverse", this result can be viewed as another blow to the *random oracle methodology*, a popular approach to the disgn of cryptographic protocols first stated explicitly by Bellare and Rogaway in [BR93]. The idea behind the random oracle methodology is to design a protocol which is secure in the random oracle model and "hope" that it remains secure even if the random oracle $\mathcal{O}$ is replaced by an efficient hash function such as MD5 or SHA (actually, we would need to use a family of functions rather than a single function, but that's a technical point). The utility of this approach stems from the apparent ease with which protocols secure in the random oracle model can be designed.

Although both [CGH98] and [GTK03] effectively demonstrate the failure of the random oracle methodology, the counterexamples used in those papers rely on Micali's CS proofs [**?**] and Barak's Universal Arguments [BG02], respectively, and are thus too unrealistic to rule out the methodology's practical usefulness. However, recent results such as [Nie02] and [BBP04] suggest that the methodology probably isn't a very good idea even for practical constructions. In other words, "security in the random oracle model" isn't a particularly reliable indicator of real-world security.

1.2 A high-level overview of the proof

The idea is to construct a secure canonical id scheme ID such that $SIG_H(ID)$ is insecure for **every** efficiently evaluable $\mathcal{H}$. In fact, we won't actually exhibit a single id scheme of this type; instead, we'll describe three different id schemes, ID^1, ID^2, and ID^3, such that one of them will provably have the desired properties. The only complexity-theoretic assumption we'll need is that one-way functions exist. However, if they do not exist then secure signature schemes don't exist either, which means that the FS transform can't possibly yield secure signature schemes. In this sense, the failure of the FS paradigm is *unconditional.*

1.3 What we've covered up to and including the June 17th meeting

1.3.1 The FS paradigm fails if collision-resistant hash function ensembles *d*on't exist

We first showed that the FS paradigm fails if collision-resistant hash function ensembles **do not** exist. Recall that $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$ is said to be *collision-resistant* if the probability of finding $x, y \in$

$\{0,1\}^*$ s.t. $x \neq y$ and $f_s(x) = f_s(y)$, taken over $s \in \{0,1\}^{n^2}$, is negligible in n. This is hardly surprising, since it seems intuitively clear that $\mathcal{H}$ would need to be collision-resistant for $SIG_H(ID)$ is to be secure.

Remember that a function $\mu : \mathbb{N} \to \mathbb{N}$ is said to be *negligible* if, for all $c \in \mathbb{N}$, there exists an $n_0 \in \mathbb{N}$ such that $\mu(n) < \frac{1}{n^c}$ for all $n \in \mathbb{N}, n \geq n_0$. In other words, $\mu(n)$ goes to 0 asymptotically faster than the inverse of any polynomial in n. In the sequel, we shall use $negl(n)$ to denote an anonymous negligible function.

We can now rewrite the above definition of collision resistance more formally, as follows: an ensemble $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$ is *collision-resistant* if, for all polysize circuit families $C = \{C_i\}_{i\in\mathbb{N}}$,

$$p_C(n) = \Pr_{k\in\{0,1\}^{n^2}}[C_{n^2}(k) = \langle x_1, x_2\rangle : f_k(x_1) = f_k(x_2)] = negl(n).$$

Here and elsewhere, $\langle\cdot,\cdot\rangle$ denotes an anonymous "reasonable" encoding of $\{0,1\}^* \times \{0,1\}^*$ into $\{0,1\}^*$. More formally, a "reasonable" encoding scheme is simply a pair of deterministic polytime algorithms (ENC, DEC) such that $DEC(ENC(x,y), 0^{|x|}) = x$ and $DEC(ENC(x,y), 1^{|y|}) = y$.

Let $SIG = (GEN, SIGN, VER)$ be some secure signature scheme; SIG must exist under our assumption that one-way functions exist. Construct a canonical id scheme $ID = (G, P, V)$ as follows: the key generation algorithm G is identical to GEN. P's first message is some fixed string, say the empty string λ, and $V(PK, \lambda, \beta, \gamma) = 1$ if and only if $VER(PK, \beta, \gamma) = 1$, i.e. γ is a legitimate signature of the random challenge string β with respect to the key pair $(PK, SK) \leftarrow G(1^n)$. Since impersonating P would require an adversary to successfully sign a random challenge in SIG, the security of ID follows from that of SIG. However, $SIG_H(ID)$ is insecure for all $\mathcal{H}$, as we show below.

Fix a hash function ensemble $\mathcal{H} = \{\mathcal{H}_n\}_{n\in\mathbb{N}}$. A polysize adversary $ADV = \{ADV_n\}_{n\in\mathbb{N}}$ breaks the security of $SIG_H(ID) = (GEN_{\mathcal{H}}, SIGN_{\mathcal{H}}, VER_{\mathcal{H}})$ as follows: say that $((PK, k), SK)$ is generated by running $GEN_{\mathcal{H}}$ on 1^n together with some random bits, and ADV_n is given (PK, k). ADV_n first finds two distinct strings x and y s.t. $h_k(x) = h_k(y)$, where $h_k \in \mathcal{H}_n$ is uniquely specified by the key $k \in \{0,1\}^{n^2}$. Since $\mathcal{H}$ is **not** collision-resistant – because no ensemble is, by assumption – ADV_n succeeds in doing this with some non-negligible probability, which we denote by $p_{ADV}(n)$. ADV_n then queries its signing oracle $SIGN_{\mathcal{H}}(SK)$ on x to obtain a valid signature $\sigma_x = (\alpha, \beta, \gamma)$ of x, where $\beta = h_k(x)$ because $h_k(\lambda, x) = h_k(x)$. Finally, ADV_n outputs the pair (y, σ_x), where y is the message he's trying to sign and σ_x is its supposed signature.

Since $h_k(x) = h_k(y)$, σ_x is indeed a valid signature of y with respect to $((PK, k), SK)$. Therefore the probability that $VER(PK, k, y, \sigma_x) = 1$ is exactly the probability that ADV_n found such a y in the first place, i.e. $p_{ADV}(n)$. But $p_{ADV}(n)$ is non-negligible in n, by assumption, so ADV breaks the security of $SIG_H(ID)$, as promised.

1.3.2 The FS paradigm fails if collision resistant hash function ensembles *do* exist

Assume that collision resistant hash function ensembles do exist, as is widely believed, and let $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}}$, where $\mathcal{F}_n = \{f_s : \{0,1\}^{2n} \to \{0,1\}^n\}_{s\in\{0,1\}^{n^2}}$, be one such ensemble. We will use $\mathcal{F}$ as part of a special *computationally binding* commitment scheme (we'll explain what we mean by this below), called a *Merkle tree commitment* [Mer90], which will enable us to construct the id scheme we need. Let us first explain how tree committments work and what sort of properties they have.

Fix a security parameter n, and randomly choose a key $k \in \{0,1\}^{n^2}$. This key uniquely determines a function $f_k \in \mathcal{F}_n$, $f_k : \{0,1\}^{2n} \to \{0,1\}^n$.

Consider any $x \in \{0,1\}^*$ such that $|x| = n \cdot 2^m$ for some $m \in \mathbb{Z}^+ = \mathbb{N} \setminus \{0\}$. We may view x as being made of up of 2^m n-bit 'blocks': $x = x_1, \ldots, x_{2^m}$, $|x_i| = n, 1 \le i \le 2^m$. A tree committment to x under f_k, which we'll denote by $TC_{f_k}(x)$, consists of the integer m together with the label assigned to the root of the complete binary tree of height m (recall that such a tree has exactly 2^m leaves) by the following recursive labelling:

- Base case: the leaves of the tree are labelled by the n-bit blocks of x. In other words, $label_x(leaf_i) = x_i$, $1 \le i \le 2^m$, where the leaves are ordered from left to right.
- Induction step: to label an internal node, concatenate the labels of its two children together and apply f_k to the resulting $2n$-bit string, i.e. $label_x(node) = f_k(label_x(left\_child(node)) \circ label_x(right\_child(node)))$, where '$\circ$' denotes the string concatenation operator.

Since f_k maps $2n$-bit strings to n-bit strings, this procedure assigns an n-bit label, $label_x(node)$, to every node in the tree as it works its way up towards the root. We call such a labelled complete binary tree *the Merkle tree corresponding to x*.

Recall that a complete binary tree with 2^m leaves has $2^{m+1} - 1$ nodes in total. We can therefore number the nodes of the Merkle tree from 1 to $2^{m+1} - 1$, starting at the root and working down towards the leaves from left to right, so that the root has index 1 and the rightmost leaf has index $2^{m+1} - 1$.

Finally, set

$$TC_{f_k}(x) = label_x(root) \circ [m]_2,$$

where '$\circ$' denotes the concatenation operator and '$[m]_2$' denotes the binary representation of m. Note that no special separator is needed to distinguish between the two parts of $TC_{f_k}(x)$, since $|label_x(root)| = n$, with n fixed and made pulbic prior to the first application of the committment.

Recall that $m = \log_2(\frac{|x|}{n})$, so that

$$\ell = |[m]_2| = \lceil \log_2(m) \rceil = \lceil \log_2(\log_2(\frac{|x|}{n})) \rceil$$

Although we've only looked at x's whose length is linear in n up till now, this means that even if $|x|$ were *doubly exponential* in n, e.g. $|x| = 2^{2^{c \cdot n}}$ for some $c \in \mathbb{Z}^+$, k would still be linear in n. Thus

$$|TC_{f_k}(x)| = n + \ell = \mathcal{O}(n)$$

for all x's we are likely to ever care about.

However, observe that although the *space* required to store strings which are exponentially long in the security parameter n is linear in n, the *time* required to compute the Merkle tree committments to such strings is still exponential in n. This is because we must examine every bit of a string in order to commit to it.

So far, we have restricted our attention to x's such that $|x| = n \cdot 2^m$ for some $m \in \mathbb{Z}^+$. How do we commit to strings that are not of this (rather special) form? The important point to realize is that *any* $x \in \{0,1\}^*$ can be padded to an x' such that $|x'| = n \cdot 2^m$ for some $m \in \mathbb{Z}^+$ in a uniquely decodable way: simply let $\ell = \lceil \log_2(\frac{|x|}{n}) \rceil$, $m = n \cdot 2^\ell - |x| - 1$, and set $x' = 0^m 1 x$. Since $|x'| = (n \cdot 2^\ell - |x| - 1) + 1 + |x| = n \cdot 2^\ell$, we have $\log_2(\frac{|x'|}{n}) = \log_2(2^\ell) = \ell$, a positive integer. Notice that x should technically be padded with a leading '1' before m is computed, for otherwise we will have $m = -1$ for x's that are already of the right form. We can easily recover x from x' by stripping away all of the leading characters up to and including the first '1'.

In light of this fact, we will henceforth assume (wlog) that $m = \log_2(\frac{|x|}{n}) \in \mathbb{Z}^+$.

What about decommittment? The standard way to decommit to $x \in \{0,1\}^*$ is to simply send x to the other party. Having received a supposed committment $y = y_1 \circ y_2$, where $y_1 \in \{0,1\}^n$ and $y_2 \in \{0,1\}^\ell$, followed by $x \in \{0,1\}^*$, that party will accept if and only if $|x| = n \cdot 2^m$, where m is the integer y_2 is the binary representation of, and $label_x(root) = y_1$.

Now that we've specified both how to commit and how to decommit to strings using the Merkle tree committment scheme, it remains to explain what sort of properties it possesses.

CLAIM: If $\mathcal{F}$ is collision resistant, then $TC_{\mathcal{F}}$, the Merkle tree committment with respect to $\mathcal{F}$, is *computationally binding*. That is, for every security parameter n, given a randomly chosen $k \in \{0,1\}^{n^2}$, which uniquely determines a function $f_k \in \mathcal{F}_n$, it is infeasible to find two distinct strings of equal length whose Merkle tree committments with respect to f_k coincide.

More formally, for all polysize circuit families $C = \{C_i\}_{i \in \mathbb{N}}$,

$$q_C(n) = \Pr_{k \in \{0,1\}^{n^2}}[C_{n^2}(k) = \langle x_1, x_2 \rangle : |x_1| = |x_2| \wedge x_1 \neq x_2 \wedge TC_{f_k}(x_1) = TC_{f_k}(x_2)] = negl(n).$$

PROOF: The proof uses the standard "randomized reduction" technique, which Goldreich refers to as a 'reducibility argument' in [Gol01].

Suppose that there exists a polysize circuit family $ADV = \{ADV_i\}_{i \in \mathbb{N}}$ which breaks the computational unambiguity of $TC_{\mathcal{F}}$, so that $q_{ADV}(n) > \frac{1}{n^c}$ for some $c \in \mathbb{Z}^+$ and infinitely many n's. We will use ADV to construct a polysize circuit family ADV' for which $p_{ADV'}(n) > \frac{1}{n^c}$, which means that ADV' breaks the collision resistance of $\mathcal{F}$. Since $\mathcal{F}$ is collision-resistant by assumption, this is a contradiction.

Fix an n such that $q_{ADV}(n) > \frac{1}{n^c}$.

On input $k \in \{0,1\}^{n^2}$, ADV'_{n^2} first runs ADV_{n^2} on k to obtain a pair of strings $\langle x_1, x_2 \rangle$. It then constructs the Merkle trees corresponding to x_1 and x_2, and compares the labels of the two trees node by node, starting with node $2^m - 1$ (i.e. the rightmost node in the level directly above the leaves) and working up towards the root from right to left. If ADV'_{n^2} finds an index i such that $label_{x_1}(node_i) \neq label_{x_2}(node_i)$, it outputs $\langle y_1, y_2 \rangle$, where $y_1 = label_{x_1}(left\_child(node_i)) \circ label_{x_1}(right\_child(node_i))$ and $y_2 = label_{x_2}(left\_child(node_i)) \circ label_{x_2}(right\_child(node_i))$, so that $|y_1| = |y_2| = 2n$.

What is the probability $p_{ADV'}(n)$ that $y_1 \neq y_2$, yet $f_k(y_1) = f_k(y_2)$?

We know that $x_1 \neq x_2$, $TC_{f_k}(x_1) = TC_{f_k}(x_2)$ with probability $q_{ADV}(n) > \frac{1}{n^c}$. Let $x_1 = x_1^1 \circ x_1^2 \circ \cdots \circ x_1^{2^m}$ and $x_2 = x_2^1 \circ x_2^2 \circ \cdots \circ x_2^{2^m}$, where $m \in \mathbb{Z}^+$ and $|x_1^i| = |x_2^i| = n, 1 \leq i \leq 2^m$.

References

[BBP04] Mihir Bellare, Alexandra Boldyreva, and Adriana Palacio. An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In *Proceedings of Advances in Cryptology—EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques*, volume 3027 of *Lecture Notes in Computer Science*, pages 171–188. Springer-Verlag, 2004.

[BG02] Boaz Barak and Oded Goldreich. Universal arguments and their applications. In *Proceedings of the 17th IEEE Annual Conference on Computational Complexity—CCC '02*, pages 162–171. IEEE Computer Society, 2002.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: a paradigm for designing efficient protocols. In *Proceedings of the 1st ACM conference on Computer and communications security—CCS '93*, pages 62–73. ACM Press, 1993.

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *Proceedings of STOC '98: Thirtieth Annual ACM Symposium on Theory of Computing*, pages 209–218. ACM Press, 1998.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: practical solutions to identification and signature problems. In *Proceedings of Advances in cryptology—CRYPTO '86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer-Verlag, 1987.

[Gol01] Oded Goldreich. *Foundations of Cryptography: Basic Tools*. Cambridge University Press, 2001.

[GTK03] Shafi Goldwasser and Yael Tauman Kalai. On the (in)security of the fiat-shamir paradigm. In *Proceedings of FOCS '03: 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 102–113. IEEE Computer Society, 2003.

[Mer90] Ralph C. Merkle. A certified digital signature: That antique paper from 1979. In *Proceedings of Advances in Cryptology—CRYPTO '89, 9th Annual International Cryptology Conference*, volume 435 of *Lecture Notes in Computer Science*, pages 218–238. Springer-Verlag, 1990.

[Nie02] Jesper Buus Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In *Proceedings of Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference*, volume 2442 of *Lecture Notes in Computer Science*, pages 111–126. Springer-Verlag, 2002.