On the exact security of Full Domain Hash

Jean-Sébastien Coron

Ecole Normale Supérieure
45 rue d'Ulm
Paris, F-75230, France
coron@clipper.ens.fr

Gemplus Card International
34 rue Guynemer
Issy-les-Moulineaux, F-92447, France
jean-sebastien.coron@gemplus.com

Abstract. The Full Domain Hash (FDH) scheme is a RSA-based signature scheme in which the message is hashed onto the full domain of the RSA function. The FDH scheme is provably secure in the random oracle model, assuming that inverting RSA is hard. In this paper we exhibit a slightly different proof which provides a tighter security reduction. This in turn improves the efficiency of the scheme since smaller RSA moduli can be used for the same level of security. The same method can be used to obtain a tighter security reduction for Rabin signature scheme, Paillier signature scheme, and the Gennaro-Halevi-Rabin signature scheme.

1 Introduction

Since the discovery of public-key cryptography by Diffie and Hellman [3], one of the most important research topics is the design of practical and provably secure cryptosystems. A proof of security is usually a computational reduction from solving a well established problem to breaking the cryptosystem. Well established problems include factoring large integers, computing the discrete logarithm modulo a prime p, or extracting a root modulo a composite integer. The RSA cryptosystem [9] is based on this last problem.

A very common practice for signing with RSA is to first hash the message, add some padding, and then exponentiate it with the decryption exponent. This "hash and decrypt" paradigm is the basis of numerous standards such as PKCS #1 v2.0 [10]. In this paradigm, the simplest scheme consists in taking a hash function, the output size of which is exactly the size of the modulus : this is the Full Domain Hash scheme (FDH), introduced by Bellare and Rogaway in [1]. The FDH scheme is provably secure in the random oracle model, assuming that inverting RSA, *i.e.* extracting a root modulo a composite integer, is hard. The random oracle methodology was introduced by Bellare and Rogaway in [1] where they show how to design provably secure signature schemes from any trapdoor permutation. In the random oracle model, the hash function is seen as an oracle which produces a random value for each new query.

The seminal work of Bellare and Rogaway in [1] and [2] highlights the importance, for practical applications of provable security, of taking into account the tightness of the security reduction. A security reduction is tight when breaking the signature scheme leads to solving the well established problem with sufficient probability, ideally with probability one. In this case, the signature scheme is almost as secure as the well established problem. On the contrary, if the reduction is "loose", *i.e.* the above probability is too small, the guarantee on the signature scheme can be quite weak.

In this paper, we exhibit a better security reduction for the FDH signature scheme, which gives a tighter security bound. The reduction in [2] bounds the probability ϵ of breaking FDH in time t by $\epsilon' \cdot (q_{hash} + q_{sig})$ where ϵ' is the probability of inverting RSA in time t' comparable to t and q_{hash} and q_{sig} are the number of hash queries and signature queries requested by the forger. The new reduction bounds the probability ϵ of breaking FDH by roughly $\epsilon' \cdot q_{sig}$ with the same running time t and t'. This is significantly better in practice since q_{sig} is usually much less than q_{hash}. Full domain hash is thus more secure than originally foreseen. With a tighter provable security one can safely use a smaller modulus size, which in turn improves the efficiency of the scheme.

2 Definitions

2.1 Signature schemes

A digital signature of a message is a bit string dependent on some secret known only to the signer, and on the content of the message being signed. Signatures must be verifiable : anyone can check the validity of the signature. The following definitions are based on [5].

Definition 1 (signature scheme). *A signature scheme is defined by the following :*

- The key generation algorithm Gen is a probabilistic algorithm which given 1^k, outputs a pair of matching public and secret keys, (pk, sk).

- The signing algorithm Sign takes the message M to be signed and the secret key sk and returns a signature $x = Sign_{sk}(M)$. The signing algorithm may be probabilistic.

- *The verification algorithm* $Verify$ *takes a message* M*, a candidate signature* x' *and the public key* pk*. It returns a bit* $Verify_{pk}(M, x')$*, equal to* 1 *if the signature is accepted, and* 0 *otherwise. We require that if* $x \leftarrow Sign_{sk}(M)$*, then* $Verify_{pk}(M, x) = 1$*.*

Signature schemes most often use hash functions. In the following, the hash function is seen as a random oracle : the output of the hash function h is a uniformly distributed point in the range of h. Of course, if the same input is invoked twice, identical outputs are returned.

2.2 Security of signature schemes

The security of signature schemes was formalized in an asymptotic setting by Goldwasser, Micali and Rivest [5]. Here we use the definitions of [1] and [2] which take into account the presence of an ideal hash function, and give a concrete security analysis of digital signatures. Resistance against adaptive chosen-message attacks is considered : a forger $\mathcal{F}$ can dynamically obtain signatures of messages of its choice and attempts to output a valid forgery. A *valid forgery* is a message/signature pair (M, x) such that $Verify_{pk}(M, x) = 1$ but the signature of M was never requested by $\mathcal{F}$.

Definition 2. *A forger* $\mathcal{F}$ *is said to* $(t, q_{sig}, q_{hash}, \epsilon)$*-break the signature scheme* $(Gen, Sign, Verify)$ *if after at most* $q_{hash}(k)$ *queries to the hash oracle,* $q_{sig}(k)$ *signatures queries and* $t(k)$ *processing time, it outputs a valid forgery with probability at least* $\epsilon(k)$ *for all* $k \in \mathbb{N}$*.*

Definition 3. *A signature scheme* $(Gen, Sign, Verify)$ *is* $(t, q_{sig}, q_{hash}, \epsilon)$*-secure if there is no forger who* $(t, q_{sig}, q_{hash}, \epsilon)$*-breaks the scheme.*

2.3 The RSA cryptosystem

The RSA cryptosystem [9] is the most widely used public-key cryptosytem. It can be used to provide both secrecy and digital signatures.

Definition 4 (The RSA cryptosystem). *The RSA cryptosystem is a family of trapdoor permutations. It is specified by :*

- *The RSA generator* $\mathcal{RSA}$*, which on input* 1^k*, randomly selects* 2 *distinct* $k/2$*-bit primes* p *and* q *and computes the modulus* $N = p \cdot q$*. It randomly picks an encryption exponent* $e \in \mathbb{Z}^*_{\phi(N)}$ *and computes the corresponding decryption exponent* d *such that* $e \cdot d = 1 \bmod \phi(N)$*. The generator returns* (N, e, d)*.*

- *The encryption function* $f : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$ *defined by* $f(x) = x^e \bmod N$.

- *The decryption function* $f^{-1} : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$ *defined by* $f^{-1}(y) = y^d \bmod N$.

2.4 Quantifying the security of RSA

We follow the definitions of [2]. An *inverting algorithm* $\mathcal{I}$ for RSA gets input N, e, y and tries to find $f^{-1}(y)$. Its success probability is the probability to output $f^{-1}(y)$ when N, e, d are obtained by running $\mathcal{RSA}(1^k)$ and y is set to $f(x)$ for an x chosen at random in $\mathbb{Z}_N^*$.

Definition 5. *An inverting algorithm $\mathcal{I}$ is said to (t, ϵ)-break RSA if after at most $t(k)$ processing time its success probability is at least $\epsilon(k)$ for all $k \in \mathbb{N}$.*

Definition 6. *RSA is said to be (t, ϵ) secure if there is no inverter which (t, ϵ)-breaks RSA.*

3 The Full Domain Hash signature scheme

3.1 Definition

The Full Domain Hash $(GenFDH, SignFDH, VerifyFDH)$ signature scheme [1] is defined as follows. The key generation algorithm, on input 1^k, runs $\mathcal{RSA}(1^k)$ to obtain (N, e, d). It outputs (pk, sk), where $pk = (N, e)$ and $sk = (N, d)$. The signing and verifying algorithm have oracle access to a hash function $H_{FDH} : \{0, 1\}^* \to \mathbb{Z}_N^*$. Signature generation and verification are as follows :

$SignFDH_{N,d}(M)$
  $y \leftarrow H_{FDH}(M)$
  return $y^d \bmod N$

$VerifyFDH_{N,e}(M, x)$
  $y \leftarrow x^e \bmod N; y' \leftarrow H_{FDH}(M)$
  if $y = y'$ then return 1 else return 0.

The concrete security analysis of the FDH scheme is provided by the following theorem [1] :

Theorem 1. *Suppose RSA is (t', ϵ')-secure. Then the Full Domain Hash signature scheme is (t, ϵ)-secure where $t = t' - (q_{hash} + q_{sig} + 1) \cdot \mathcal{O}(k^3)$ and $\epsilon = (q_{hash} + q_{sig}) \cdot \epsilon'$.*

As stated in [2], the disadvantage of this result is that ϵ' could be much smaller than ϵ. For example, if we assume like in [2] that the forger is allowed to request $q_{sig} = 2^{30}$ signatures and computes hashes on $q_{hash} = 2^{60}$ messages, even if the RSA inversion probability is as low as 2^{-61}, then all we obtain is that the forging probability is at most $1/2$, which is not satisfactory. To obtain an acceptable level of security, we must use a larger modulus, which will affect the efficiency of the scheme.

To obtain a better security bound, Bellare and Rogaway designed a new scheme, the *probabilistic signature scheme* (PSS), which achieves a tight security reduction : the probability of forging a signature is almost equally low as inverting RSA ($\epsilon \simeq \epsilon'$). Instead, we show in the next section that a better security bound can be obtained for the original FDH scheme.

3.2 The new security reduction

We exhibit a different reduction which gives a better security bound for FDH. Namely, we prove the following theorem :

Theorem 2. *Suppose RSA is (t', ϵ')-secure. Then the Full Domain Hash signature scheme is (t, ϵ)-secure where*

$$t = t' - (q_{hash} + q_{sig} + 1) \cdot \mathcal{O}(k^3)$$

$$\epsilon = \frac{1}{(1 - \frac{1}{q_{sig}+1})^{q_{sig}+1}} \cdot q_{sig} \cdot \epsilon'$$

For large q_{sig},

$$\epsilon \simeq \exp(1) \cdot q_{sig} \cdot \epsilon'$$

Proof. Let $\mathcal{F}$ be a forger that $(t, q_{sig}, q_{hash}, \epsilon)$-breaks FDH. We assume that $\mathcal{F}$ never repeats a hash query or a signature query. We build an inverter $\mathcal{I}$ which (t', ϵ')-breaks RSA.

The inverter $\mathcal{I}$ receives as input (N, e, y) where (N, e) is the public key and y is chosen at random in $\mathbb{Z}_N^*$. The inverter $\mathcal{I}$ tries to find $x = f^{-1}(y)$ where f is the RSA function defined by N, e. The inverter $\mathcal{I}$ starts running $\mathcal{F}$ for this public key. Forger $\mathcal{F}$ makes hash oracle queries and signing queries. $\mathcal{I}$ will answer hash oracle queries and signing queries itself. We assume for simplicity that when $\mathcal{F}$ requests a signature of the message M, it has already made the corresponding hash query on M. If not, $\mathcal{I}$ goes ahead and makes the hash query itself. $\mathcal{I}$ uses a counter i, initially set to zero.

When $\mathcal{F}$ makes a hash oracle query for M, the inverter $\mathcal{I}$ increments i, sets $M_i = M$ and picks a random r_i in $\mathbb{Z}_N^*$. $\mathcal{I}$ then returns $h_i = r_i^e \bmod N$ with probability p and $h_i = y \cdot r_i^e \bmod N$ with probability $1-p$. Here p is a fixed probability which will be determined later.

When $\mathcal{F}$ makes a signing query for M, it has already requested the hash of M, so $M = M_i$ for some i. If $h_i = r_i^e \bmod N$ then $\mathcal{I}$ returns r_i as the signature. Otherwise the process stops and the inverter has failed.

Eventually, $\mathcal{F}$ halts and outputs a forgery (M, x). We assume that $\mathcal{F}$ has requested the hash of M before. If not, $\mathcal{I}$ goes ahead and makes the hash query itself, so that in any case $M = M_i$ for some i. Then if $h_i = y \cdot r_i^e \bmod N$ we have $x = h_i^d = y^d \cdot r_i \bmod N$ and $\mathcal{I}$ outputs $y^d = x/r_i \bmod N$ as the inverse for y. Otherwise the process stops and the inverter has failed.

The probability that $\mathcal{I}$ answers to all signature queries is at least $p^{q_{sig}}$. Then $\mathcal{I}$ outputs the inverse of y for f with probability $1-p$. So with probability at least $\alpha(p) = p^{q_{sig}} \cdot (1-p)$, $\mathcal{I}$ outputs the inverse of y for f. The function $\alpha(p)$ is maximal for $p_{max} = 1 - 1/(q_{sig}+1)$ and

$$\alpha(p_{max}) = \frac{1}{q_{sig}} \left(1 - \frac{1}{q_{sig}+1}\right)^{q_{sig}+1}$$

Consequently we obtain :

$$\epsilon(k) = \frac{1}{(1 - \frac{1}{q_{sig}+1})^{q_{sig}+1}} \cdot q_{sig} \cdot \epsilon'(k)$$

and for large q_{sig}, $\epsilon(k) \simeq \exp(1) \cdot q_{sig} \cdot \epsilon'(k)$.

The running time of $\mathcal{I}$ is the running time of $\mathcal{F}$ added to the time needed to compute the h_i values. This is essentially one RSA computation, which is cubic time (or better). This gives the formula for t.

$\square$

3.3 Discussion

In many security proofs in the random oracle model (including [2]), the inverter has to "guess" which hash query will be used by the adversary to produce its forgery, resulting in a factor of q_{hash} in the success probability. This paper shows that a better method is to include the challenge y in the answer of many hash queries so that the forgery is useful to the inverter with greater probability. This observation also applies to the Rabin signature scheme [8], the Paillier signature scheme [7] and also the Gennaro-Halevi-Rabin signature scheme [4], for which the q_{hash} factor in the random oracle security proof can also be reduced to q_{sig}.

4 Conclusion

We have improved the security reduction of the FDH signature scheme in the random oracle model. The quality of the new reduction is independent from the number of hash calls performed by the forger, and depends only on the number of signatures queries. This is of practical significance since in real-world applications, the number of hash calls is only limited by the computational power of the forger, whereas the number of signature queries can be deliberately limited : the signer can refuse to sign more than 2^{20} or 2^{30} messages. However, the reduction is still not tight and there remains a sizable gap between the exact security of FDH and the exact security of PSS.

5 Acknowledgements

I would like to thank Jacques Stern, David Pointcheval and Alexey Kirichenko for helpful discussions and the anonymous referees for their constructive comments.

References

1. M. Bellare and P. Rogaway, *Random oracles are practical : a paradigm for designing efficient protocols.* Proceedings of the First Annual Conference on Computer and Commmunications Security, ACM, 1993.
2. M. Bellare and P. Rogaway, *The exact security of digital signatures - How to sign with RSA and Rabin.* Proceedings of Eurocrypt'96, LNCS vol. 1070, Springer-Verlag, 1996, pp. 399-416.
3. W. Diffie and M. Hellman, *New directions in cryptography*, IEEE Transactions on Information Theory, IT-22, 6, pp. 644-654, 1976.
4. R. Gennaro, S. Halevi, T. Rabin, *Secure hash-and-sign signatures without the random oracle*, proceedings of Eurocrypt'99, LNCS vol. 1592, Springer-Verlag, 1999, pp. 123-139.
5. S. Goldwasser, S. Micali and R. Rivest, *A digital signature scheme secure against adaptive chosen-message attacks*, SIAM Journal of computing, 17(2):281-308, april 1988.
6. A. Lenstra and H. Lenstra (eds.), *The development of the number field sieve*, Lecture Notes in Mathematics, vol 1554, Springer-Verlag, 1993.
7. P. Paillier, *Public-key cryptosystems based on composite degree residuosity classes.* Proceedings of Eurocrypt'99, Lecture Notes in Computer Science vol. 1592, Springer-Verlag, 1999, pp. 223-238.
8. M.O. Rabin, *Digitalized signatures and public-key functions as intractable as factorization*, MIT/LCS/TR-212, MIT Laboratory for Computer Science, 1979.

9. R. Rivest, A. Shamir and L. Adleman, *A method for obtaining digital signatures and public key cryptosystems*, CACM 21, 1978.
10. RSA Laboratories, PKCS #1 : *RSA cryptography specifications*, version 2.0, September 1998.