A review of the ESIGN digital signature standard

Nick Howgrave-Graham

NTRU Cryptosystems
5 Burlington Woods,
Burlington, MA 01803 USA.
Email: nhowgravegraham@ntru.com

Abstract. This document is a review for CRYPTREC 2001 regarding the ESIGN digital signature standard. In this report we make no progress towards a better solution to the approximate e'th roots problem, but we do introduce a new attack on the ESIGN standard. This (lattice) attack is based on an attacker having some knowledge of the bits of the ephemeral keys used while signing. Under suitable conditions this can lead to recovery of the private key (namely the factorisation of n) quickly, and thus should be taken very seriously. Similar attacks have been shown for the digital signature algorithm (DSA), and they have had a wide impact on practical implementations of cryptosystems. The existence of this attack stresses the importance of using truly uniform pseudorandom numbers, and the careful implementation of the ESIGN standard. The attack does not apply to a correctly implemented version of the standard.

Keywords: greatest common divisor, approximations, hidden number problems, lattice attacks

1 Introduction

In this report we assess the security of the ESIGN digital signature standard as specified by [16–18]. The algorithm was designed to address the problem of RSA signatures, namely the asymmetric computation times in signing (relatively expensive) and verification (relatively cheap). In section 2 we sum up the scheme.

In the "standard model", where the algorithm is correctly implemented, we can find no weakness with the algorithm, i.e. no progress has been made on the e'th root approximation problem for any $e \geq 4$. However it is arguably prudent for a signature scheme to be secure against slightly stronger attacks models. In [8] Howgrave-Graham and Smart examined the security of the digital signature algorithm (DSA) assuming that an adversary has knowledge of some of the bits of the ephemeral keys used in signing. Many practical instances of the DSA did, and perhaps still do, leak such information, and so this is a realistic attack model. As well as from poor implementation, this information may be leaked from side channel attacks such as a timing or power analysis[1]. Much subsequent work has been done on this topic, see [12, 13, 2], and so ESIGN is assessed under this security model in section 4. Unfortunately (and rather remarkably) the situation for ESIGN is exactly the same as for the DSA, i.e. the secret key can be leaked quickly from this information. The existence of such a property in a signature scheme may be considered a design flaw; at very least it means that implementors must be careful.

In section 5 we sum up some other minor points to do with the standard, and in section 6 we recap the main results of the review.

2 A summary of the ESIGN standard

Keygen For some security parameter k, pick two random primes, p and q, of length k bits, and set $n = p^2q$. If n does not have $3k$ bits then one should choose different primes. Also pick some positive integer $e \geq 8$.

The ESIGN standard requires that $k \geq 320$ and $e \geq 8$. The suggested parameters are $k = 384$ and $e = 2^{10} = 1024$.

Signing In order to sign a message representative m of length $k-1$ bits (resulting from a hash of the true message) one does the following steps:

1. Choose a random nonce r uniformly at random from Z^*_{pq}.
2. Set $z = 2^{2k}m$.
3. Set $\alpha = z - r^e \bmod n$.
4. Let w_0, w_1 be positive integers such that $w_0pq - w_1 = \alpha$ for some $w_1 < pq$.
5. Set $t = w_0(er^{e-1})^{-1} \bmod p$.
6. Set $s = r + tpq$.

After which one outputs the signature s of the message m.

Verification On input a signature s of length $3k$ bits and a message representative m of length $k-1$ bits one can verify that s is a signature of the message m by performing the following steps:

1. Set $t = s^e \bmod n$.
2. Check that the bits of t in positions $2k \ldots 3k-1$ agree with the bits of m, and that the top bit of t is zero. If so then set $v = 1$ else set $v = 0$.

After which one returns the verification bit v.

[1] Such attacks are less applicable to ESIGN since the random nonce has very little impact on the computation time.

2.1 Proof of validity

The verification works simply because

$$\begin{aligned} s^e &= (r + tpq)^e \bmod n \\ &= r^e + etpqr^{e-1} + \ldots \bmod n \\ &= z + w_1 \end{aligned}$$

and since $w_1 < pq$ it cannot effect the top k bits of z, since the bottom $2k$ bits of z are all zero.

3 Existing attacks

If an adversary manages to forge an ESIGN signature on a message m then clearly he has solved the approximate e'th root problem for the specific message m, i.e. he has found an s for which the top k bits of $s^e \bmod n$ are equal to $0||m$. Assuming this problem is hard on the average case implies that the attacker can not perform such an action given merely m and the public parameters.

In section 3.1 we recap previous results which show that the e'th root problem is not hard for the cases $e = 2$ and $e = 3$, so these parameters should not be allowed.

In section 3.2 we also consider the problem of directly retrieving the private key (p and q) from the public parameter n. Of course this knowledge immediately implies that one can sign in the standard way, and thus forge any message one wants.

Both of the above attacks do not assume that useful information can be obtained from a set of signature transcripts, indeed in section 3.3 we explain why it is very likely that transcripts leak no information.

Finally in section 3.4 we examine whether any of these attacks may be improved on in the near future.

3.1 The cases of $e = 2$ and $e = 3$

As noted in [6] the e'th root problem is actually easy for the cases $e = 2$ and $e = 3$. The reason for this is that we can simply disregard the modulo n part, and work with real numbers. To be more specific suppose we wish to forge a signtaure on a message m then let $z = 2^{2k}m$ and let s, d be such that $s = z^{1/e} + d$, for some real number d, $|d| \leq 1/2$. In this case

$$s^e = (z^{1/e} + d)^e = z + edz^{e-1/e} + \ldots$$

When $e = 2$ this means that $s^e - z$ is only about $\sqrt{n}$, and thus approximately the top half of the bits of s^e and z are the same, i.e. s is a valid siganture for the message m. For the case $e = 3$ the size of $s^e - z$ is about $1.5n^{2/3}$, so now the top 1/3 of the bits of z and s^e will almost match but there may be some non-matching toward the lesser significant bits. By exhaustively testing $s' = s \pm x$ for some small x, one may find a signature for the message m with non-negligible probability of success.

3.2 Factoring $n = p^2q$

As discussed in [16], the best current way to factor a number of the form $n = p^2q$ which has ≥ 960 bits, is to apply the number field sieve (NFS), i.e. the same technique as for other integers of a similar size (e.g. an RSA modulus). For an integer of this size the running time of this algorithm is currently infeasible, although just as with RSA it may well be prudent to increase the recommeded bit size of n to allow for improvements in factoring integers, e.g. n with $6 \times 384 = 2304$ may be recommended for a higher degree of security.

Another alternative for factoring n is to use the standard elliptic curve factoring method (ECM), or slight modifications thereof for $n = p^2q$. However in both these cases finding factors whose bit size is atleast 320 bits is beyond the reach of feasibility.

A third possibility is to use lattice techniques to factor $n = p^rq$ (see [5]), however for $r = 2$ this technique is inferior to both the NFS and ECM.

3.3 Transcript analysis

In this section we examine what information a transcript analysis of ESIGN signatures leaks, i.e. we assume we are given many pairs (m_i, s_i) such that the bits in positions $2k \dots 3k-1$ of s_i^e are equal to the bits of m_i, and the top bit s_i^e is 0. The m_i correspond to the message digests resulting from a hash function, and this hash function is modelled as a random oracle [1].

If we do model the hash function as a random oracle, then such a transcript clearly leaks no useful information. This follows because an attacker with no private information can simulate such a transcript by simply choosing random s_i uniformly from Z_n, and checking whether the top bit of $s_i^e \bmod n$ is 0. If so (which will happen half of the time on average), then the attacker can set the bits of m_i to be equal to the bits of $s_i^e \bmod n$ in positions $2k \dots 3k-1$.

Such (s_i, m_i) will be indistinguishable from random, i.e. uniformly distributed in the space of valid transcripts.

3.4 On extending these attacks

It is clear that an adversary that can break the e'th root problem can forge signatures, and in section 3.1 we showed that this was is possible for $e = 2$ and $e = 3$. However for all $e \geq 4$ the simple trick of ignoring the modulo n part, and finding the closest integer to the e'th root over the reals will not work: the error will simply be too large.

A similar situation holds for low exponent RSA: when there is no wrapping modulo n, an encrypted (small) message m can be easily recovered by taking e'th roots over the integers, however when there is significant wrapping modulo n (i.e. for general low exponent RSA, when m is no longer small) the problem appears to be intractable [3].

With regards to ESIGN and in the presence of significant wrapping modulo n, as is the case for any $e \geq 4$, it appears that this is a suitably hard problem on which to base the security of the signature scheme.

The recommendation that $e \geq 8$ for practical implementations seems reasonable.

Further lattice work has been done in [7, 15] but these techniques, while interesting, do not shed any new light on how to attack ESIGN for $e \geq 4$.

Improvements to integer factorisation (and thus key recovery for ESIGN) are always happening. It is this reviewers opinion that two levels of security should be recommended, one with $k = 384$ and the other with $k = 768$. The latter should be immune to progress in integer factorisation for a substantial amount of time; see [10] for realistic estimates.

4 The weakness in the ephemeral keys

ESIGN signatures are of the form $s = r + tpq$ for some integers $r < pq$ and $t < p$. In this section we assume that we have some information about the bits of the ephemeral key r. Such a situation might come from the following sources:

1. the use of a weak pseudorandom number,
2. probing attacks,
3. a poor implementation of ESIGN.

Since ESIGN uses SHA-1 as the pseudorandom number generator (for which no significant weaknesses are know) source 1 is not really applicable. Source 2 is certainly a realistic attack, but is more dependent on the instantiation of ESIGN in hardware, so should probably be left to a later stage. Source 3 is also dependent on the implementation of ESIGN, this time in software, however from observations made with the practical implementations of the DSA, it is very important to stress to implementors, even at the standards stage, that r

cannot be chosen to have any bias. When the range of a random nonce does not end on a power of 2 (as in the case of ESIGN), implementations often simply set the top bit to 0 to ensure $r < pq$. Such efficiency gains do introduce biases to r.

It is interesting to note that in ESIGN the random nonce r has only a very marginal impact on the computation time of signing.

If we now assume that we know l integers s_i of the form

$$s_i = r_i + t_i pq \tag{1}$$

for which r_i is a little less than pq we will describe a technique to recover these r_i.

Clearly equation 1 implies that $ps_i = r_i p + t_i n$ which suggest building the following lattice

$$L = \begin{pmatrix} D & s_1 & s_2 & \dots & s_l \\ & n & & & \\ & & n & & \\ & & & \ddots & \\ & & & & n \end{pmatrix} \tag{2}$$

where $D \approx n^{2/3}$.

This lattice was first studied by Boneh and Venkatesan in [4], in which they examined a general problem known as the *hidden number problem*, and its application to proving the hardness of the bits of the Diffie–Helman secret.

The applicability of this lattice to attacking signature schemes was first noticed by Howgrave-Graham and Smart in [8], and since then much work has been done on extending the scheme; see [12, 13, 2].

To quickly summarise the attack note that by the creation of L we know that the vector $xL = (pD, pr_1, pr_2, \dots, pr_l)$ is in the lattice, where $x = (p, -t_1, -t_2, \dots, -t_l)$. This has norm approximately pR where R is a bound on the size of the r_i. If this is significantly below the smallest vector anticipated in a lattice of determinant Dn^l, then one may use lattice reduction techniques such as [11] to recover the small vector, and hence determine the secret key p (which is the first entry of x).

The results in [13] are the strongest in a provable sense. Also in practice they show that when only the top 3 bits of each of the r_i is known then lattice reduction techniques will recover the remaining bits.

It is worth mentioning that for any fixed number of signatures one may use the techniques of [9] to achieve better bounds, but in general it is better to use the lattice L and as many signatures as possible.

Very recently Bleichenbacher has developed non-lattice techniques [2] to recover the r_i. These techniques are certainly applicable when there is a bias corresponding to knowing the top 1.5 bits of each of the r_i. For smaller biases the size of the computation time, and size of the required transcript increases significantly.

The mere existence of these kinds of attack mean that one should be very careful leaking any information of this kind.

5 Other minor comments

Report [17] appears to be a little out of date. In the keygen algorithm specification, it does not say to reject n for which the bit size is less than $3k$, and it also allows any $e > 4$ rather than $e \geq 8$.

Report [18] is generally well written and comprehensive. The only recommendation for this report is that it be more specific about how to generate $r \in \{1, 2, \ldots, pq - 1\}$ such that $\gcd(r, n) = 1$ in the section on SP-ESIGN (assuming the random data is obtained from SHA-1 say).

Report [16] is a good introduction to ESIGN, although the timing results at the end use $e = 32$, which should really be replaced by the recommended values of $e = 1024$. Future versions of this document should also include the dangers inherent in mishandling the random number generation.

6 Conclusions

The major contribution of this report is to emphasize the importance of the correct use of random numbers in ESIGN, where this point has not be made in the past. It is the opinion of this reviewer that the underlying hard problem behind ESIGN, namely calculating approximate e'th roots is indeed a suitably hard problem on which to build a cryptosystem. Properly implemented, ESIGN is an attractive and efficient signature scheme.

It is recommended that two levels of security are given to the users of ESIGN, i.e. recommending $k = 384$ for standard security, and $k = 768$ for a signature scheme that will need to remain secure for a long time. This is to allow for advancements in integer factorization.

The recommended choice of $e = 2^{10} = 1024$ seems sensible.

References

1. M. Bellare and P. Rogaway Random oracles are practical: a paradigm for designing efficient protocols *Proc. of the first ACM CCS* pp. 62-73, 1993.

2. D. Bleichenbacher Private communication.
3. D. Boneh Twenty years of attacks on the RSA cryptosystem. *Notices of the American Mathematical Society (AMS)* Vol. 46, No. 2, pp. 203–213, 1999.
4. D. Boneh and R. Venkatesan Hardness of computing the most significant bits of secret keys in Diffie–Hellman related schemes *Proc. of Crypto '96* Vol. 1109 Springer. 1996
5. D. Boneh, G. Durfee and N. Howgrave-Graham Factoring $N = p^r q$ for large r. *In Proceedings Crypto '99*, Lecture Notes in Computer Science, Vol. 1666, Springer-Verlag, pp. 326–337, 1999.
6. E. Brickell and J. DeLaurentis An attack on a signature scheme proposed by Okamoto and Shiraishi *In Proc. of Crypto '85*, LNCS, Vol. 218, pp.28–32, 1986 Springer-Verlag
7. D. Coppersmith Small solutions to polynomial equations, and low exponent RSA vulnerabilities *J. of Crypto*, Vol. 10(4), pp.233–260, 1997 Springer-Verlag
8. N. A. Howgrave-Graham and N. P. Smart Lattice attack on digital signature schemes *Design, Codes and Cryptography 2001* Vol. 23, pp. 283–290, 2001 Kluwer Academic Press.
9. N. A. Howgrave-Graham Approximate integer common divisors *Cryptography and Lattices, CaLC 2001* LNCS Vol. 2146, pp. 51–66, 2001 Springer–Verlag
10. A. K. Lenstra Unbelievable security; matching AES security using public key systems *In Proc. Asiacrypt 2001* LNCS Vol. 2248, pp. 67–86, 2001 Springer–Verlag
11. A. K. Lenstra, H. W. Lenstra and L. Lovász. Factoring polynomials with integer coefficients *Mathematische Annalen*, Vol. 261, pp. 513–534, 1982.
12. P. Q. Nguyen. The dark side of the hidden number problem; lattice attacks on DSA *Proceedings of CCNT '99* pp. 321–330, 2001. Birkhauser.
13. P. Q. Nguyen and I. E. Shparlinski The insecurity of the digital signature algorithm with partially known nonces *To appear in J. Crypto*
14. T. Okamoto. Fast public–key cryptosystem using congruent polynomial equations *Electronic letters*, Vol. 22, No. 11, pp. 581–582, 1986.
15. B. Vallée, M. Girault and P. Toffin. How to break Okamoto's cryptosystem by reduing lattice bases *Proceedings of Eurocrypt '88* LNCS vol. 330, pp. 281–291, 1988.
16. REVIEW ITEM. Self-evaluation of ESIGN.
17. REVIEW ITEM. Specification of ESIGN Signatures.
18. REVIEW ITEM. ESIGN Specification. NTT Information Sharing Platform Laboratories, NTT Corporation.