ESIGN and Other *RSA* *Alterative Signature Schemes*

Richard Kramer, Member IEEE – Oregon State University

What is the Main Disadvantage of RSA?

Today's Objective

Provide you with a general understanding of:

RSA Alterative Signature Schemes

RSA Overview

RSA (named after RSA's inventors, Rivest, Shamir, and Adelman) was first introduced in 1977 [RivestShamirAdleman1978]

In order to understand the disadvantages of RSA, *it is FIRST important to do a recap on what RSA entails:*

- RSA generates digital signatures and cipher text, *C*, by performing exponentiation on a message, *M*, to the *e*-th power of the form:

$$C = M^e \bmod(n)$$

Where *n* is based on large prime numbers.

RSA *Encryption*

(1) Alice: Chooses and calculates:

- Choose p and q = prime numbers
- From that, calculate $n = pq$ and $\phi(n) = (p-1)(q-1)$
- From that, choose e (relatively prime) to $\phi(n)$, e.g., no common divisor > 1 that can go into e and $(p-1) \times (q-1)$

Alice

(2) Alice: Generates a Public Key (PK = *n, e*):

- n, from above
- e, from above

(4) Public Key (n, e)

(5) $C = M^e \bmod n$

Bob

(3) Alice: Generates a Secret Key (SK = *p, q, d*):

- p and q from above
- $d = e^{-1} (\text{mod}(\phi(n))$. Where: $e \times d \equiv 1 \bmod(\phi(n))$

(6) Alice: Decodes Bob's Cipher Text

$M = C^d \bmod n$

RSA *Signature*

(1) Alice: Chooses and calculates:

- Choose p and q = prime numbers
- From that, calculate $n = pq$ and $\phi(n) = (p-1)(q-1)$
- From that, choose e (relatively prime) to $\phi(n)$, e.g., no common divisor > 1 that can go into e and $(p-1) \times (q-1)$

Alice

(2) Alice: Generates a Public Key (PK = *n, e*):

- n, from above
- e, from above

(4) Sends Bob PK (n,e)

(6) Send Bob message M, and signature, S

Bob

(3) Alice: Generates a Secret Key (SK = *p, q, d*):

- p and q from above
- $d = e^{-1} (\text{mod}(\phi(n))$. Where: $e \times d \equiv 1 \text{ mod}(\phi(n))$

(5) Alice: Signs a Message (*M, d*)

$S = M^d \text{ mod } n$

(7) Bob: Validates Alice's Signature E against Message *M*

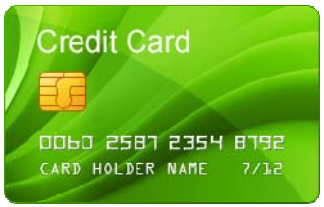
$M = S^e \text{ mod } n$?

RSA Overview

But what happens with Alice and Bob *are replaced by a low end embedded devices*?

(1) Generates a Public Key (PK = *n, e*):

- n, from above
- e, from above



(3) Sends PK (n,e)

(5) Send message M, and signature, S

(2) Generates a Secret Key (SK = *p, q, d*):

- p and q from above
- $d = e^{-1} \pmod{\phi(n)}$. Where: $e \times d \equiv 1 \pmod{\phi(n)}$

(6) Validate Signature S against Message *M*

$M = S^e \bmod n$?

(4) Signs a Message (*M, d*) **using an inverse exponential**

$S = M^d \bmod n$

Inverse exponentiation is extremely processor intensive!

Contributions / Agenda:

- Provide a comprehensive survey of ESIGN and review some important foundational work of ESIGN
- Provide the Background for ESIGN's Proof of Security
  - Factorization is at the core of ESIGN's Proof of Security
- Provide an overview of ESIGN Variants and Recommended Improvements
- ESIGN's Security and Vulnerabilities
- Review Other RSA Alternatives
  - EdDSA [LiuSeoGroßschädlKim2016]
  - ED25519 [Bernstein,DuifLangeSchwabeYang2011] / [Bernstein,DuifLangeSchwabeYang2012]

Future Reading / References

[Bernstein,DuifLangeSchwabeYang2012] Daniel J. Bernstein, Niels Duif, Tanja Lange, Peter Schwabe, and Bo-Yin Yang Daniel J. Bernstein, Niels Duif, Tanja Lange, Peter Schwabe, and Bo-Yin Yang, "High-speed high-security signatures", Journal of Cryptographic Engineering, Volume 2, Issue 2, pgs. 77–89, September 2012.

[BonehShacham2002] Dan Boneh and Hovav Shacham, "Fast Variants of RSA", CryptoBytes, Vol. 5, No. 1, pgs. 1-9, 2002.

[FouqueHowgrave-GrahamMartinetPoupard2003] Pierre-Alain Fouque, Nick Howgrave-Graham, Gwenaelle Martinet, and Guillaume Poupard, "The Insecurity of Esign in Practical Implementations", C.S. Laih (Ed.): ASIACRYPT 2003, LNCS 2894, pp. 492–506, 2003.

[FujiokaOkamotoMiyaguchi1991] Atsushi Fujioka, Tatsuaki Okamoto, Shoji Miyaguchi, "ESIGN: An Efficient Digital Signature Implementation for Smart Cards", Advances in Cryptology - EUROCRYPT '91, Volume 547 of the series Lecture Notes in Computer Science pp 446-457, 1991

[Granboulan2003] Louis Granboulan, "How to repair ESIGN", Proceeding SCN'02, Proceedings of the 3rd international conference on Security in communication networks, pgs. 234-240, 2002, © 2003.

[Howgrave-Graham] Nick Howgrave-Graham, "A Review of the ESIGN digital standard", NTRU Cryptosystems at link: https://www.ipa.go.jp/security/enc/CRYPTREC/fy15/doc/1007_esign.pdf, date unknown.

[Kobayashi Fujisaki2007] Tetsutaro Kobayashi, Eiichiro Fujisaki, "Security of ESIGN-PSS", IEICE Transactions on Fundamentals, Vol. E90, No. 7, pgs. 1395-1405, 2007.

[Kunihiro Kurosawa2007] Noboru Kunihiro and Kaoru Kurosawa, "Deterministic Polynomial Time Equivalence between Factoring and Key-Recovery Attack on Takagi's RSA", 10th International Conference on Practice and Theory in Public-Key Cryptography Beijing, China, pp 412-425, 2007.

[LiuSeoGroßschädlKim2016] Zhe Liu, Hwajeong Seo, Johann Großschädl, Howon Kim, "Efficient Implementation of NIST-Compliant Elliptic Curve Cryptography for 8-bit AVR-Based Sensor Nodes", IEEE Transactions on Information Forensics and Security, Vol. 11, Issue: 7, pgs. 1385–1397, 2016.

Future Reading / References

[MenezesQuStinsonWang2001] Alfred Menezes, Minghua Qu, Doug Stinson, Yongge Wang, "Evaluation of Security Level of Cryptography: ESIGN Signature Scheme", Certicom Research, 2001.

[MityaginPanjwan Raghavan2007] Anton Mityagin, Saurabh Panjwani, and Barath Raghavan, "Public Review for Analysis of the SPV Secure Routing Protocol", Subtitle "Analysis of the SPV Secure Routing Protocol: Weaknesses and Lessons", ACM SIGCOMM Computer Communication Review 29, Volume 37, Number 2, pgs. 29-38, 2007.

[OkamotoFujisakiMorita1998] Tatsuaki Okamoto, Eiichiro Fujisaki ,Hikaru Morita, "TSH-ESIGN: Efficient Digital Signature Scheme Using Trisection Size Hash (Submission to P1363a)", NTT Laboratories, 1998.

[OkamotoStern2003] Tatsuaki Okamoto and Jacques Stern, "Almost Uniform Density of Power Residues and the Provable Security of ESIGN", C.S. Laih (Ed.): ASIACRYPT 2003, LNCS 2894, pp. 287–301, 2003.

[RivestShamirAdleman1978] R. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public Key Cryptosystems." Commun. ACM, 21(2) pgs. 120–126, 1978.

[SelfEvaluationofESIGNSignature] Author Unknown, "Self Evaluation of ESIGN Signature", at link: security.nknu.edu.tw/crypto/Self_ESIGN.pdf, date unknown.

[Sarkar2014] Santanu Sarkar, "Small secret exponent attack on RSA variant with modulus $N = p^r q$", Des. Codes Cryptogr. 73:383–392, 2014.

[Schneier1994] Bruce Schneier, "Applied Cryptography", John Wiley and Sons Publishing, 1994.

[Wiki_TwistedEdwards_curve] at link: https://upload.wikimedia.org/wikipedia/commons/b/b7/Twisted_Edwards_curve.svg.

[ZinzindohouèBartziaBhargavan2016] Jean Karim Zinzindohou´e, Evmorfia-Iro Bartzia, Karthikeyan Bhargavan, "A Verified Extensible Library of Elliptic Curves", 2016 IEEE 29th Computer Security Foundations Symposium, pgs. 296-309, 2016.

Glossary of Terms

AERP	Approximate e-th Root Problem
AER	Approximate e-th Root
AS	Autonomous System
BGP	Border Gateway Protocol
CMA	Chosen Message Attack, (also known as EUA-CMA)
DSA	Digital Signature Algorithm
ECC	Elliptical Curve Cryptography
ECDSA	Elliptical Curve Digital Signature Algorithm
ECF	Elliptic Curve Factoring
ECM	Elliptic Curve factoring Method
EdDSA	Edwards-curve Digital Signature Algorithm
ESIGN	An efficient digital signature algorithm
ESIGN-D	ESIGN-Deterministic
ESIGN-R	ESIGN-Randomized
ESIGN-PSS	ESIGN-Probabilistic Signature Scheme
ESIGN-PSS-R	ESIGN-Probabilistic Signature Scheme with Recovery
EUA-CMA	Existentially Unforgeable Against a Chosen Message Attack (also known as CMA)
GCD	Greatest Common Divisor
LLL	A lattice algorithm developed by A. K. Lenstra, H.W. Lenstra, Jr. and L. Lovasz, usually called, thus called the LLL algorithm.
NFS	Number Field Sieve
RSA	Rivest, Shamir, and Adelman
SVP	Shortest Vector Problem (relates to factoring)
SVP	Secure Path Vector (relates to BGP)
TSH	Tri-Section Hash
TSH-ESIGN	Tri-Section Hash-ESIGN

ESIGN Overview

ESIGN was first introduced in 1985 by T. Okamoto and A. Shiraishi [OkamotoFujisakiMorita1998]

ESIGN builds on RSA by introducing the relationship $\boldsymbol{n = p^2q}$ (versus $n = pq$ for RSA) among other changes

Most importantly, ESIGN advantages:

- ***Avoids inverse exponentiation***
- ***Allows for pre-processing*** calculations of certain variables that are independent of the messages, M

What is ESIGN?

Choose and calculate:

1. Choose p *and* q = large prime numbers where $p > q$
2. Choose k: The bit length of p and/or q
3. From the above, calculate $n = p^2q$ which results in a $3k$ bits long value
4. Choose e, a positive integer
   - As will be discussed later, make $e \geq 4$

Public Key (PK = *n, e, k*):

- n, from above
- e, from above
- k, from above
- H, the hash function used

Secret Key (SK = *p, q*~~, *d*~~):

- p and q, chosen above

Primary Changes from RSA's Keys

There is no "d" based on inverse exponentiation!

What is ESIGN?

The *Signature* Process:

1. Choose r: Pick a random number $r \in Z^*_n$

2. **For a message *M***, calculate variables that will be used to later create S (the digital signature) as follows:

   a. $Z = H(M)) - r^e \bmod(n)$ where H is a one-way hash function $H(M) \in Z^*_{pq}$ for any positive integer message M

   b. $\omega_0 = \lceil ((H(M)) - r^e \bmod(n))/pq \rceil$, (is rounded to the upper integer ($\lceil\ \rceil$))

   c. $\omega_1 = \omega_0 pq - Z$

   d. If $\omega_1 > 2^{2k-1}$, then repeat the above steps starting with picking a new value r.

   e. $Y = [\omega_0/(er^{e-1})] \bmod(p)$

***3. S = r + Ypq* (the digital signature of message *M*)**

What is ESIGN?

Because the signer knows r, e, p, q and n, the following can be pre-computed independent of any Message, M:

1. Choose r: Pick a random number $r \in Z^*_n$, where Z^*_n denotes a set of integer numbers between 0 and n-1 which is relatively prime to n
2. **For a message *M***, calculate variables that will be used to later create S (the digital signature) as follows:
   a. $Z = H(M)) - \boldsymbol{r^e \bmod (n)}$ where H is a one-way hash function H(M) $\in$ Z^*_{pq} for any positive integer message M
   b. $\omega_0 = \lceil ((H(M)) - \boldsymbol{r^e \bmod (n))/pq} \rceil$, (is rounded to the upper integer ($\lceil \ \rceil$))
   c. $\omega_1 = \omega_0 pq - Z$
   d. If $\omega_1 > 2^{2k-1}$, then repeat the above steps starting with picking a new value r.
   e. $Y = [\omega_0 \boldsymbol{/ (er^{e-1})] \bmod (p)}$

Further, if p, q and n are fixed (e.g., in a smart card, a lookup table can be built to calculate mod(n) and mod(p))

What is ESIGN?

The *Verification* Process:

- For digital signature verification, uniquely, ***ESIGN provides verification based on a range*** (versus an exact value as in RSA), by solving the following inequality:

1) $0 \leq S < n$
2) $0 \leq S^e \bmod(n)/2^{2k} < 2^{(k-1)}$
3) H'(M) computed = H'(M) received

ESIGN Overview

(1) Generates a Public Key (PK = *n*, *e*, *k*)
(2) Generates a Secret Key (SK = *p*, *q*)
(3) Choose a random number, *r*, and pre-calculate:

$r^e \bmod(n)$, $r^e \bmod(n))/pq$, and $1/[(er^{e-1}) \bmod(p)]$

(4) Sends PK (n,e, k)

(7) Send message M, and signature, S

(5) Based on a Message, *M*, calculate *Z*, ω_0, ω_1 , *Y*

(6) Sign the Message:
$S = r + Ypq$

(8) Validate Signature *S* against Message *M*

$0 \le S^e \bmod(n)/2^{2k} < 2^{(k-1)}$?

No inverse exponentiation!

Background for ESIGN's Proof of Security

The security for ESIGN is based on the intractability of the ***Approximate e-th Root Problem (AERP)***

*The **Approximate e-th Root Problem (AERP) assumption** is: there is no efficient algorithm for solving the AERP, whereas the AERP problem entails finding the e-th root of a modulo n equation*

Overall, the objective to a digital signature scheme is to be **Existentially (*i.e.*, Proven) Unforgeable Against a Chosen Message Attack (EUA-CMA)**

Background for ESIGN's Proof of Security

Numerous schemes have been developed to attempt to efficiently solve the AERP problem

Aspect related to the AERP that have been analyzed include:

1) The complexity of the e-th root problem based on the value of e
2) The size of p, q, n
3) Potential leakages of information
4) Non-randomness / bias, and various other attack scenarios

Background for ESIGN's Proof of Security

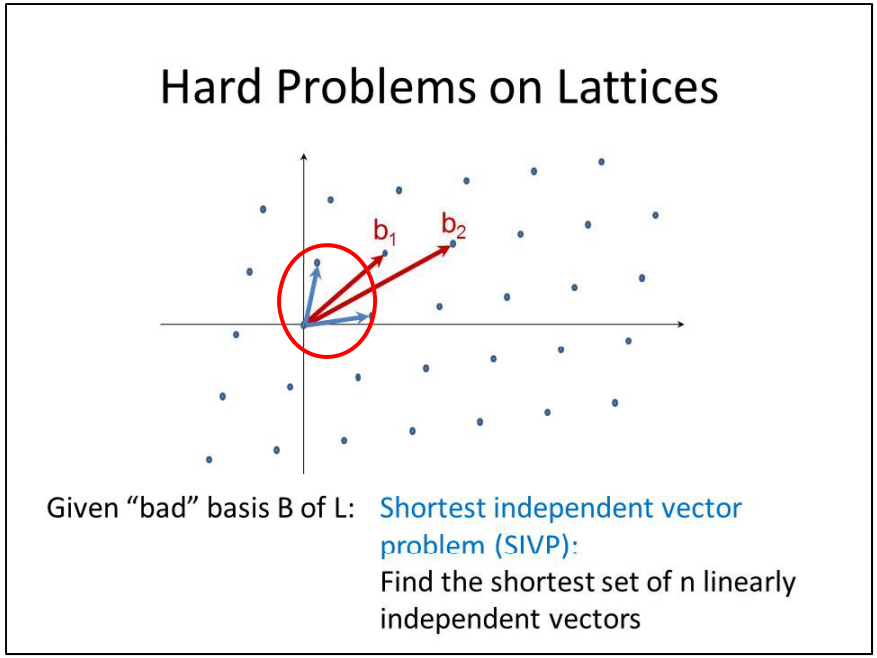
Leading Factorization Methods that have been employed to solve the AERP

They include:

- Lattice Based Reduction Factorization and LLL (Lenstra, Lenstra, Lovasz)
- Quadratic Sieve Factoring
- NFS (Number Field Sieve)
- ECF / ECM (Elliptic Curve Factoring / Elliptic Curve Method)

Factorization Methods - Lattice Based Reduction Factorization and LLL (Lenstra, Lenstra, Lovasz)

The LLL (Lenstra, Lenstra, Lovász) algorithm significantly improved lattice based factorization by reducing the lattice using short vectors in polynomial time



[SelfEvaluationofESIGNSignature] [Howgrave-Graham] OkamotoStern2003] [FouqueHowgrave-GrahamMartinetPoupard2003][Kunihiro Kurosawa2007] [Schneier1994]

Factorization Methods - Lattice Based Reduction Factorization and LLL (Lenstra, Lenstra, Lovasz)

The goal is to find the shortest vector in a lattice

- Lattice based factorization is a method of building a matrix made up of portions of the polynomial to be factored,
- … and then using the output of a factorization table to complete the matrix and thus find the factors

As the complexity of the polynomial equation increases (*e.g.*, the value of *e*), the complexity of the lattice increases to the point of unmanageability:

Run time = $O(n^4(\log(i \leq n))$

Where n = the number of linear independent vectors

Factorization Methods – Quadratic Sieve (QS) Factoring

QS is very fast for factoring numbers less than 150 decimal digits

In certain cases, the QS factoring method has been proven to be more efficient than the lattice base reduction method, for example, when $e \geq 4$ given the special case of n being the product of two primes of equal size

The QS attempts to find pairs of integers x and $y(x)$ (where $y(x)$ is a function of x) such that:

$x^2 \equiv y^2 \pmod{n}$…. recall that RSA if of the form $S = M^d \bmod(n)$

QS selects a set of primes called the factor base, and attempts to find x such that the least absolute remainder of $y(x) = x^2$

Nonetheless, other methods have proved more efficient/effective including the **Number Field Sieve (NFS)** method, thus QS factoring has been surpassed by NFS

[MenezesQuStinsonWang2001] [Schneier1994]

Factorization Methods – NFS (Number Field Sieve)

NFS (Number Field Sieve) has been validated to be superior over QS (Quadratic Sieve) based on the work of numerous researchers and experimentation

Overall, NFS's expected run time is given as:

Run time =

$$O(exp((1.306 + o(1)) \ln(n)^{1/3}(\ln(\ln(n)))^{2/3}))$$

where the run time is dependent of n, the product of two prime numbers (like p and q)

[MenezesQuStinsonWang2001][SelfEvaluationofESIGNSignature]

Factorization Methods - ECF / ECM (Elliptic Curve Factoring / Elliptic Curve Method)

The Elliptical Curve Factoring Method (ECM) is the fastest factoring algorithm known *when the algorithm run time is measured in terms of small prime numbers.*

- This is specifically useful based on the smallest prime of n (*e.g,* q, where in ESIGN by definition $p > q$)
- Assumes the smallest prime number is small overall

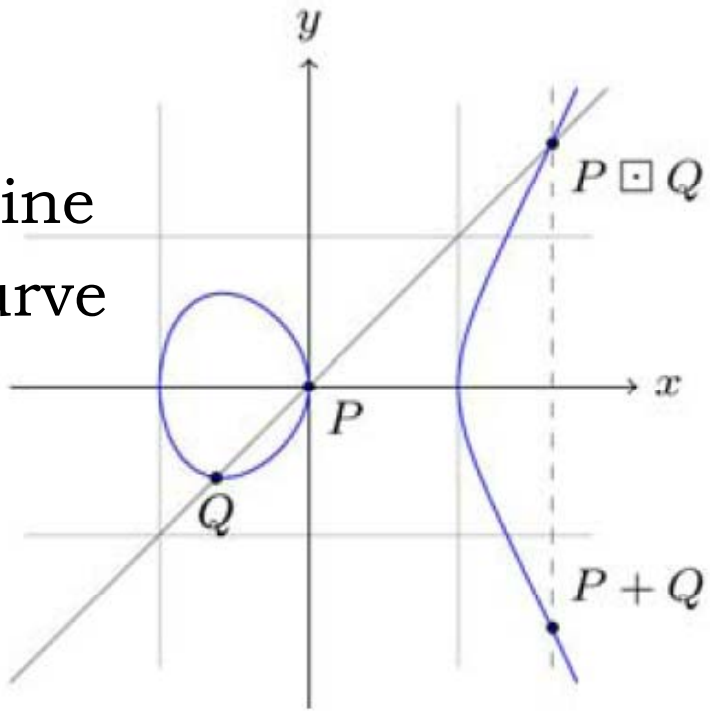
[MenezesQuStinsonWang2001][SelfEvaluationofESIGNSignature]

Factorization Methods - ECF / ECM (Elliptic Curve Factoring / Elliptic Curve Method)

ECF/ECM can be used if p and q are two prime divisors of n:

- Then the curve of the form $y^2 = x^3 + x + b \pmod{n}$ implies the same equation is also modulo p and modulo q

The method uses the fact that a line through p and q intersects the curve and is related to $p + q$



[ZinzindohouèBartziaBhargavan2016]

ESIGN Variants and Recommended Improvements

ESIGN Implementations, Improvements and Variants Overview

There are many ESIGN Variants

- TSH-ESIGN [OkamotoFujisakiMorita1998]
- ESIGN-D (Deterministic) [Granboulan2003]
- ESIGN-R (Randomized) [Granboulan2003]
- Other Variants and Recommendations for ESIGN [Howgrave-Graham]
- ESIGN-PSS (Probabilistic Signature Scheme) [BonehShacham2002]
- ESIGN-PSS-R (Probabilistic Signature Scheme with Recovery) [Kobayashi Fujisaki2007]

ESIGN Variants and Recommended Improvements - Overview

Over time, a number of ESIGN variants have been proposed to:

1) Improve security:

- For example, shortly after ESIGN was introduced, it was proven that LLL factorization could be used to factor ESIGN signatures when e = 2 or 3.

2) To aid in the proof of security

ESIGN Variants - TSH-ESIGN

The "Tri-Section Hash" aspect is based on the fact that the hash function, H, in TSH-ESIGN is made up of ***three (tri) sections***

The previous form for Z for ESIGN was:

$$Z = \mathrm{H}(M) - r^e \bmod(n)$$

For TSH-ESIGN, the hash function, H, is such that the H'(M) must be of the proper length, thus resulting in: H'(M) = ($0||\mathrm{H}(M)$), where the hash input for Z is ($\mathrm{H}'(M)||0^{2k}$), thus ($0||\mathrm{H}(M)||0^{2k}$)

Tri-Section Hash

[OkamotoFujisakiMorita1998]

ESIGN Variants - TSH-ESIGN

Like ESIGN overall, TSH-ESIGN still offers significant reduction in processing speed and power

Schemes	*Sig. Gen. (M(1024))*	*Sig. Ver. (M(1024))*
TSH-ESIGN	9	5
RSA-based scheme (e.g., PSS or FDH-RSA)	384	17
EC-based scheme (e.g., EC-Schnorr or EC-DSA)	24	28

Relative comparison of "computation amount"

Where:

$M(1024)$ → 1024 bit multiplication

For RSA: $e = 2^{16}+1$; modulus n = 1024 bits

ESIGN: $e = 2^5$; modulus n = 1024 bits

EC (Elliptical Curve) Scheme: modulus = 160 bits

[OkamotoFujisakiMorita1998]

ESIGN Variants - ESIGN-D (Deterministic)

- While ESIGN is proven to be *e*-th root mod(*n*) secure (*e.g.*, under the AERP assumption), a number of security proofs for ESIGN have been flawed:
  - *We will discuss this further in the Section: "ESIGN's Security and Vulnerabilities"*
- To extend the provability of ESIGN, **a *deterministic* variant** of ESIGN was created called ESIGN-D (Deterministic)

[Granboulan2003]

ESIGN Variants - ESIGN-D (Deterministic)

To create the ESIGN-D variant, a number of modified steps are performed relative to the original ESIGN:

1) Instead of choosing the random number *r*, **a one-way function ϕ is created**. The function ϕ outputs a randomly distributed number that is mod(*n*).
2) **A new *k* bit (length) value** Δ is created and is included as part of the **Secret Key (SK).**
3) Instead of using the iterative calculation of ω_1 (where from above, $\omega_1 = \omega_0 pq - Z$, where ω_0 is dependent on *r; e.g.*, $\omega_0 = \lceil ((H(M)) - r^e \bmod(n))/pq \rceil$ and if $\omega_1 > 2^{2k-1}$ then a new r is chosen, instead of *r,* **a new variable is introduced, which I will call *r'***.

Where: ***r'* = ϕ(H(*M*)||Δ||*i*) for *i* = 0,1, 2, ...**

[Granboulan2003]

ESIGN Variants - ESIGN-D (Deterministic)

ESIGN-D Advantages:

- Simulation results show that $S^e \bmod(n) \bmod(2^{2k-1})$ is indistinguishable from the value generated from the signature algorithm, where Δ is sufficiently large such that ϕ is unpredictable.

ESIGN-D Disadvantages:

- Overall, ESIGN-D adds additional processing costs
- … yet at bit lengths of 1152 or 1536 there are no additional security gains as opposed to unmodified ESIGN

[Granboulan2003]

ESIGN Variants - ESIGN-R (Randomized)

ESIGN-R (ESIGN-Randomized) was created to allow **for the simulation of the probabilistic output of the ESIGN signature generator** (oracle).

Accordingly, the new signature algorithms for ESIGN-R are as follows:

1) Generate additional random number ρ, where the length of ρ is ≤ $2\log_2(qS)$
2) $H'(M) = H(M||\rho)$ where the hash output H' is the result of the combination of the hash $H(M)$ of the message M and ρ
3) $\omega_0 = \lceil ((H'(M))2^{2k} - r^e \bmod(n))/pq \rceil$, *e.g.*, the hash value $H(M)$ is replaced with $H'(M)$ and multiplied by 2^{2k}

   where from above, $S = r + Ypq$ *and* $Y = [\omega_0/(er^{e-1})] \bmod(p)$
4) The enhanced digital signature is calculated as $\sigma = M||\rho||S$

[Granboulan2003]

ESIGN Variants - ESIGN-R (Randomized)

The signature verification algorithm then becomes:

- $\mathbf{H'(}M\mathbf{) = \lfloor S^e / 2^{2k} \rfloor}$**, where** $\mathbf{H'(}M\mathbf{) = H(}M\mathbf{||}\rho\mathbf{)}$

[Granboulan2003]

ESIGN Variants - ESIGN-R (Randomized)

ESIGN-R Advantages:

- Improved security over ESIGN and ESIGN-D based on the addition of external randomness factor ρ

ESIGN-R Disadvantages:

- Like ESIGN-D, ESIGN-R adds additional processing costs
- Based on the addition of ρ, the bit length of the signature is larger as compared to ESIGN and ESIGN-D
- The addition of ρ requires an additional external random generator which adds cost
- Like ESIGN-D, for ESIGN-R at bit lengths of 1152 or 1536 there are no additional security gains as opposed to unmodified ESIGN

[Granboulan2003]

ESIGN Variants - Other Variants and Recommendations for ESIGN

A number of other ESIGN variants, recommendations and improvements have been developed to improved ESIGN, including variants to RSA that can be extended to ESIGN.

A number of these variants (and improvements) include:

1) Recommendations for the size of exponent e and length k
2) Batch RSA Extended to ESIGN
3) Multi Key Settings
4) Proposed Security Levels for ESIGN

[Howgrave-Graham][BonehShacham2002]

ESIGN Variants - Other Variants and Recommendations for ESIGN

Recommendations for the size of exponent *e* and length *k*:

- Shortly after ESIGN was first introduced, it was found that a small size (*e.g.*, $e = 2$ or 3) of the exponent, *e*, allowed ESIGN signatures to be factorable using LLL
- Accordingly, new recommendations for ESIGN were created as follows:

  1) The minimum size of exponent e, was increased to $e \geq 4$

  2) It was recommended that the length *k* (for *p* and *q*) be increased to $k \geq 320$

[Howgrave-Graham][BonehShacham2002]

ESIGN Variants - Other Variants and Recommendations for ESIGN

Batch RSA Extended to ESIGN

- The concept of batch processing of, for example multiple decryptions (or digital signatures, or signature verifications) was applied to RSA, specifically because RSA requires modulo exponential inversion to recover the original message (*e.g.*, $M = C^d \bmod(n)$ where $ed \equiv 1 \bmod(n)$).
- While ESIGN does not require exponential inversion, **batch processing as a concept** can be applied to ESIGN as well.

  For example, opportunities exist to batch process certain portions of the ESIGN algorithms for multiple messages:

  1) Batch selection of large prime numbers p and q
  2) Batch selection of random number r sets
  3) Batch calculation to variables that are based on *p*, *q*, and *r*, independent of the message, *M*

[Howgrave-Graham][BonehShacham2002]

ESIGN Variants - Other Variants and Recommendations for ESIGN

Multi Key Settings

- As an extension of ESIGN to prevent an attacker from forging keys at each level within a multi key setting
- To resolve this, the hash value H(*M*) can be modified to be H'(*M*) such that *the message M is hashed with a public key at each level* of security within a multi key environment, thus:
  - **H'(*M*) = H(*M*| |PK_x) where *x* is the Public Key at a specific level, *x***

    … and at the next level H''(*M*) = H'(*M*| |PK_{x+1}) … and so on

[Howgrave-Graham][BonehShacham2002]

ESIGN Variants - Other Variants and Recommendations for ESIGN

Proposed Security Levels for ESIGN

- To continue to increase the difficulty of the AERP, two levels of security have been recommended for the length of k as follows:
  - k_{Level_1} = 384 bits
  - k_{Level_2} = 768 bits
  - Where e = 1024

[Howgrave-Graham]

ESIGN Variants - ESIGN-PSS (Probabilistic Signature Scheme)

As ESIGN continues to evolve and benefit from the advancements of RSA have also been applied to ESIGN

- As a result, ESIGN-PSS (ESIGN-Probabilistic Signature Scheme) was created
- ESIGN-PSS is part of the ISO/IEC14888-2:2008 standard

[Kobayashi Fujisaki2007]

ESIGN Variants - ESIGN-PSS (Probabilistic Signature Scheme)

The primary difference between the baseline ESIGN scheme and the ESIGN-PSS scheme is the addition of the variable v which is chosen as follows:

- $8 \leq v$, 2^{3k-1}
- GCD(v, n) =1
- GCD(v, $\phi(n) \leq 3k$, where recall that $\phi(n)$

[Kobayashi Fujisaki2007]

ESIGN Variants - ESIGN-PSS (Probabilistic Signature Scheme)

From this, the following is the Public Key (PK) and the Secret Key (SK) for ESIGN-PSS:

Public Key (PK = from Standard ESIGN: *n, k;* + *new value: v*):

- Where *n, k* are comparable to the values in the previously discussed for ESIGN

Secret Key (SK = from Standard ESIGN: *p, q, k;* + *new value v*):

- Where *p, q*, and *k* are comparable to the values in the previously discussed for ESIGN

[Kobayashi Fujisaki2007]

ESIGN Variants - ESIGN-PSS (Probabilistic Signature Scheme)

The primary changes from ESIGN to ESIGN-PSS relates to the digital signature generation (and verification)

1) ESIGN-PSS adds a random salt E, where E is then combined with the hash of the message M to form the digital signature

2) The hashed message M is then hashed (again) using hash function h:

$$\mathbf{HH} = h(\mathbf{0^{64}} || \mathbf{H}(M) || E)$$

3) And then hashed again using hash function g :

$$R = \mathbf{g(HH)} \oplus (\mathbf{0}^{kg-kE}\mathbf{-1||1}||E)$$

[Kobayashi Fujisaki2007]

ESIGN Variants - ESIGN-PSS (Probabilistic Signature Scheme)

For the digital signature verification, this process is reversed using the Public Key (PK).

ESIGN-PSS Advantages:

- Proof of security under many different models

ESIGN-PSS Disadvantages:

- Added complexity (for example performing a triple hash) in computational power (cost)
- Added length to signatures

[Kobayashi Fujisaki2007]

ESIGN Variants - ESIGN-PSS-R (Probabilistic Signature Scheme with Recovery)

Uniquely, ESIGN-PSS-R provides a *recoverable message length*, defined as follows:

$$\boldsymbol{Len_{M1}}\textbf{, where } \boldsymbol{Len_{M1}} < \boldsymbol{k^{g}} - \boldsymbol{k^{E}} - \mathbf{1}$$

- Relative to ESIGN-PSS, ESIGN-PSS-R includes the additional parameter message recovery parameter Len_{M1} **within the respective hash functions** used in the ESIGN-PSS scheme:

$$\mathbf{HH} = \boldsymbol{h}(\boldsymbol{Len_{M1}} \,||\, \boldsymbol{M_1} \,||\, \mathbf{H}(\boldsymbol{M_1}) \,||\, \boldsymbol{E})$$

$$\boldsymbol{R} = \boldsymbol{g}(\mathbf{HH}) \oplus (\mathbf{0}^{\boldsymbol{kg}-\boldsymbol{kE}} - \mathbf{1} \,||\, \mathbf{1} \,||\, \boldsymbol{E})$$

[Kobayashi Fujisaki2007]

ESIGN Variants - ESIGN-PSS-R (Probabilistic Signature Scheme with Recovery)

ESIGN-PSS-R Advantages:

- Proof of security under many different models
- Improved ability to verify message length

ESIGN-PSS Disadvantages:

- Added complexity (triple hash) in computational power (cost)
- Added length to signatures

[Kobayashi Fujisaki2007]

ESIGN's Security and Vulnerabilities

As noted: ESIGN was found to be vulnerable when the exponent e, was = 2 or 3 – **that was quickly resolved**

As with other signature schemes that use a hash function, ESIGN can be compromised if the hash function H(*M*), is:

1) Not collision resistant

2) Not pre-image collision resistant; e.g., if H(*M*) = H(*M'*), then the adversary that were to receive signature S for *M'* would also be able to get the signature for *M*

[MenezesQuStinsonWang2001]

ESIGN's Security and Vulnerabilities

While it is said that the mathematics of the AERP are not fully understood for large prime numbers (at least when e is not prime to $\phi(n)$:

Consistently ESIGN has been proven secure based on AERP absent of leakage or non-randomness / bias

In fact, ESIGN has been proven to provide near uniform distribution over a large interval :

For a fixed message M, the *e-th* power $S^e \bmod(n)$ of the output S is uniformly distributed over the set of e-th powers of elements Z_n^*, lying in interval: Y to $Y + 2^{2k-1}$, (where Y = the $0||\mathrm{H}(M)||0^{2k}$)

ESIGN's Security and Vulnerabilities

On the other hand, ESIGN is highly compromised in the presence of leakage and/or non-randomness / bias

- ***For example: The impact of ESIGN security based on leakage***

  Using the LLL algorithm with a 1152 bit mod (n) and a random variable r bit length is 768 bits:

  IF ONLY 8 BITS of the 768 bits for r are exposed, based on simply obtaining 57 signatures: the Secret Key (SK) can be recovered in a matter of minutes (using 2003 computing speed estimates).

  This can be done by solving $r^e \bmod(n)$ which allows the recovery of p and q *using LLL*

ESIGN's Security and Vulnerabilities

Experimental results of ESIGN vulnerabilities should l bits of leakage occur is as follows:

$n = 3k$	$2k$	$\log(r)$	experimental value for ℓ	theoretical bound for ℓ	d	time to factor
512	340	335	5	8	55	2 min 10
768	512	506	6	9	55	2 min 20
1024	682	674	8	11	56	2 min 30
1152	768	760	8	11	57	3 min
1536	1024	1013	11	14	57	4 min 10
2048	1364	1349	15	17	57	5 min 50

Where:

- l = bits of leakage from random number r.
- d = the number of signatures that would need to be received by an adversary A prior to being able to factor p and q

[FouqueHowgrave-GrahamMartinetPoupard2003]

ESIGN - Future Applications

The paper, "Analysis of the SPV Secure Routing Protocol: Weaknesses and Lessons" provides one example of the value and future application of ESIGN:

- The intercommunication of gateways that form the Internet require extremely fast signature generation, and as a result such signature schemes compromise speed for security including those employed in the Border Gateway Protocol (BGP)

ESIGN could be a solution to provide both speed and solid proof of security

[MityaginPanjwan Raghavan2007]

ESIGN - Conclusion

ESIGN / ESIGN's variants have been proven to provide a sufficiently intractable AERP

... and thus is as secure as RSA

... yet ESIGN affords lower computational costs at faster speeds, and allows for pre-processing

Other RSA Alternatives - EdDSA

EdDSA (Edwards Digital Signature Algorithm) is an extension of ECC (Elliptical Curve Cryptography)

ECC offers at least the following benefits:

1) Small key and signatures
2) High security equivalence: a 160-bit ECC scheme is approximately equivalent to a 1024 bit RSA scheme
3) Very fast key and signature generation speeds

Thus ideally suited for embedded applications

[LiuSeoGroßschädlKim2016]

Other RSA Alternatives - EdDSA

ECC, and by extension EdDSA, utilizes algebraic (birational) relationships between points on an Elliptical Curve

An example of an Elliptical Curve equation, *E*, is as follows (e.g., a short Weierstraβ equation):

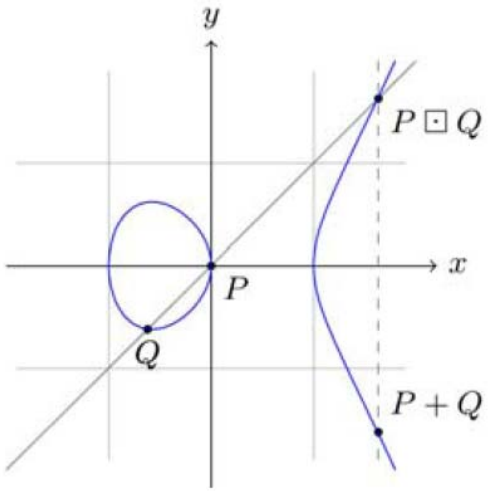
$$\boldsymbol{E:\ y^2 = x^3 + ax + b}$$

Where:

E is over an finite prime field F_p

$a, b \in F_p$

$4a^3 + 27b^2 \neq 0$



[LiuSeoGroßschädlKim2016]

[ZinzindohouèBartziaBhargavan2016]

Other RSA Alternatives - EdDSA

EdDSA builds upon ECC by introducing "Edwards Twisted Curves"

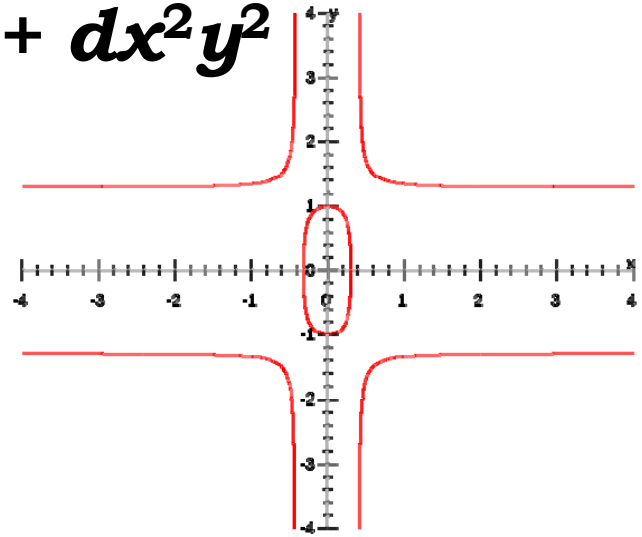
An example of an Edwards Twisted Curve is:

$$\boldsymbol{E:\ ax^2 + y^2 = 1 + dx^2y^2}$$

Where:

E is over an finite prime field F_p

$a, d \in F_p$



Graph of Edwards Twisted Curve equation:
$10x^2 + y^2 = 1 + 6x^2y^2$
[Wiki_TwistedEdwards_curve]

[LiuSeoGroßschädlKim2016]

Other RSA Alternatives – ED25519

And ED25519 is a specific implementation of an Edwards Twisted Curve:

$$E: \; x^2 + y^2 = 1 + 121665/121666x^2y^2$$

As proposed by Bernstein, ED25519 offers:

1) Very fast key and signature generation speeds
2) Fast signature verification
   … that can be further batch processed
3) Small 256 bit keys and 512 bit signatures
4) High security – the equivalent to a 3000 bit RSA scheme
5) Collision Hash resiliency

[Bernstein,DuifLangeSchwabeYang2011/2012]

Results / Comparisons

ESIGN was fully implemented against ECDSA and ED25519.

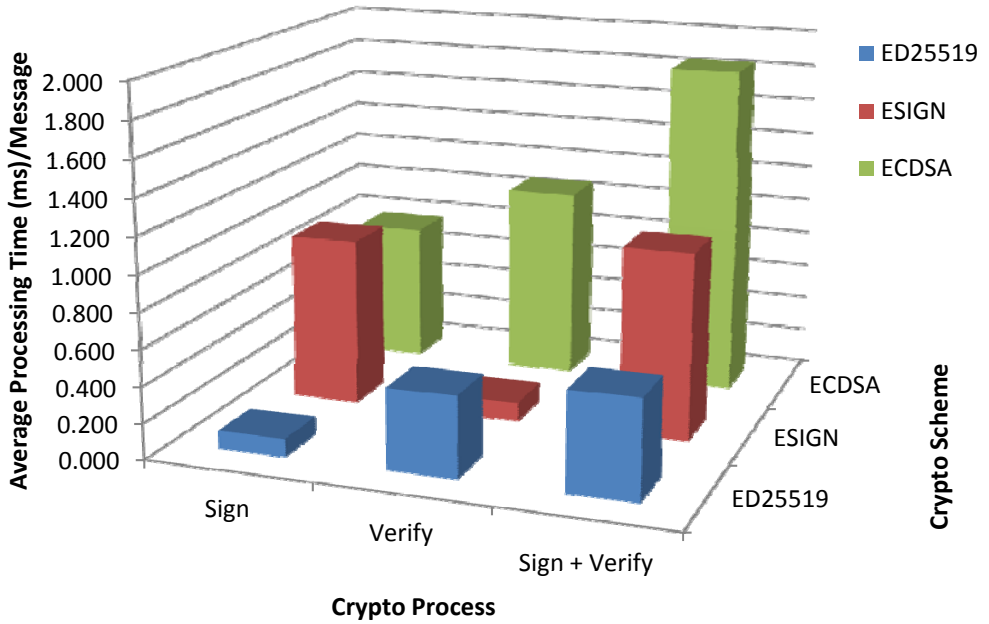
Summary:

1) **ESIGN outperformed <u>ECDSA</u> overall**. For example, on the Raspberry Pi Cortex A53 processor, ESIGN Verify versus ECDSA Verify **outperformed ECDSA by 901%** *while at the expense of only -12% for ESIGN Sign versus ECDSA Sign.*

2) **ESIGN as compared to <u>ED25519</u>**, on the Intel i5 64 bit processor, **ESIGN Verify outperformed ED25519 Verify by 930%** *while at the expense of -77% comparing ESIGN Sign to ED25519 Sign.*

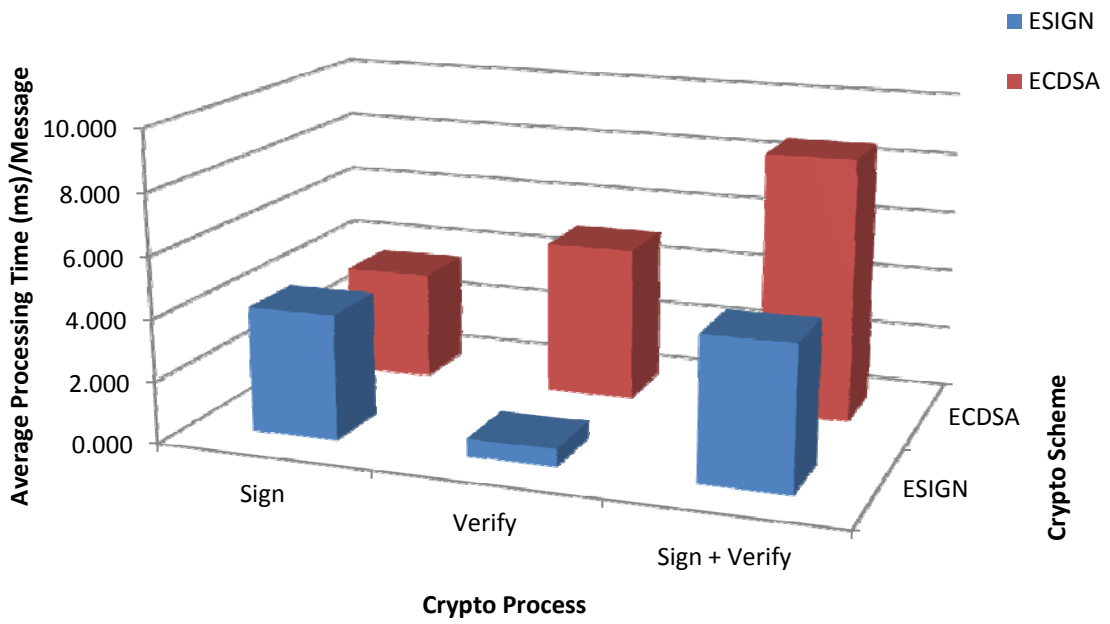
Results / Comparison

Average Processing Time/Message versus Crypto-Scheme (100,000 Messages - 64 bit Intel i5 at 1.7 GHz):



Results / Comparison

Average Processing Time/Message versus Crypto-Scheme
(100,000 Messages - Broadcom BCM2837 Cortex A53):



Summary

While RSA is still wide spread, ESIGN is proven to be equally secure

… yet **much faster** for verify

… yet at an un-optimized disadvantage as compared to ED25519

Future Work

Very little processor speed benchmarking appears to have been done to compare ESIGN to ECC methods (including EdDSA and ED25519

Thus: Review optimization opportunities for ESIGN *including a bit of a catch-22:*

1) A loop occurs when ω_1 is invalid… thus requires the re-calculation of any would be pre-computed values … which defeats the whole purpose of pre-computed values!
2) And then implement the pre-computing of values.

Questions?