

# Proving tight security for Rabin-Williams signatures

D. J. Bernstein

## Public-key signatures

1976 Diffie Hellman:

Public-key signatures

would allow verification by

anyone, not just signer. Cool!

Can we build a signature system?

1977 Rivest Shamir Adleman:

verify  $s^e - m \in pq\mathbf{Z}$ .

Public  $(pq, e)$  with big random  $e$ ;

message  $m$ ; signature  $s \in \mathbf{Z}/pq$ .

1977 (and 1978) RSA was slow:

many mults in  $eth$  powering.

Even worse, horribly insecure:

e.g. forge  $(m, s) = (1, 1)$ .

1979 Rabin:  $s^2 - H(m) \in pq\mathbf{Z}$ .

Standard  $H$ . Public  $pq$ .

Message  $m$ . Signature  $s$ .

Fast; conjecturally secure; but  
can sign only  $\approx 1/4$  of all  $m$ 's.

1979 Rabin:  $s^2 - H(r, m) \in pq\mathbf{Z}$ .

Signature  $(r, s)$  instead of  $s$ .

Signer tries random  $r$ 's,  
on average  $\approx 4$  times.

1980 Williams:  $ef s^2 - \dots \in pq\mathbf{Z}$ ;

$e \in \{-1, 1\}$ ;  $f \in \{1, 2\}$ .

Each  $h \in \mathbf{Z}/pq$  is  $ef s^2$

for exactly 4 vectors  $(e, f, s)$

if  $p \in 3 + 8\mathbf{Z}$ ,  $q \in 7 + 8\mathbf{Z}$ .

Subsequent RW variations:

- Eliminate Euclid, Jacobi.
- Expand  $s$  for faster verification.
- Compress  $s$  to  $1/2$  size.
- Compress  $pq$  to  $1/3$  size.
- Compress (“recover”)  $m$  via  $H$ .

Many other signature systems  
(e.g. elliptic-curve Schnorr),  
but RW family still holds  
verification speed records.

RW is the best choice for  
verification-heavy applications:  
e.g., Internet DNS security.

## Attacks against RW

Attacker sees public key  $pq$   
and many vectors  $(m, e, f, r, s)$   
legitimately signed under that key.

Attacker forges  $(m', e', f', r', s')$   
with  $e' \in \{-1, 1\}$ ,  $f' \in \{1, 2\}$ ,  
 $r'$  of standard length,  $s' \in \mathbf{Z}/pq$ .

Forgery is **successful** if

$e' f' (s')^2 - H(r', m') \in pq\mathbf{Z}$  and  
 $m'$  wasn't legitimately signed.

Fundamental security question:

What's maximum  $\Pr[\text{ succeeds}]$   
among all feasible attacks ?

Maybe answer depends on how messages are generated.

We want  $\Pr[\text{ succeeds}]$  small for *all* message generators and all feasible attacks .

Different users have different types of message generators, communication between attacker and message generator, etc.

Would be painful to analyze each generator separately.

Similarly, would be painful to limit set of messages.

Attack 1: Blind  $H$  inversion.

Attacker chooses  $e', f', s'$ ,  
chooses  $h \in e' f' (s')^2 + pq\mathbf{Z}$ ,  
guesses uniform random  $r', m'$   
until finding  $H(r', m') = h$ ,  
forges  $(m', e', f', r', s')$ .

Obstacle to success of attack:

What's chance of finding

$H(r', m') = h$  after a

feasible number of guesses?

Conjecturally negligible

for every popular  $H$ .

Attack 2: Blind collision search.

Attacker guesses  $r', m', r'', m$   
with  $m' \neq m$  and  
 $H(r', m') = H(r'', m)$ .

Message generator gives  $m$   
to signer:  $(m, e, f, r, s)$ .

Attacker forges  $(m', e, f, r', s)$ .

Forgery succeeds if  $r = r''$ :

$$H(r, m) = H(r'', m) = H(r', m').$$

Good chance if  $r$  is short.

Same obstacle as before:

Feasible number of guesses

has conjecturally negligible

chance of finding collision in  $H$ .



Attack 3: MD5 collision search.

Was popular last decade  
to build  $H(x) = G(\text{MD5}(x))$   
for standard function  $G$ .

Assume this shape of  $H$ .

Feasible calculation,  
highly non-uniform guessing,  
finds collisions in MD5.

(2004 Wang Feng Lai Yu)

Thus obtain collision in  $H$ .

Forge as in attack 2.

Good chance of success  
if  $r$  is short. Feasible attack!

One reaction to this attack:

MD5 was a bad design.

Change choice of  $H$ .

Collisions conjecturally infeasible for many popular  $H$ 's.

Another reaction (1979 Rabin, 1989 Schnorr, et al.):

Standardize 256-bit  $r$ .

Negligible chance of  $r = r''$ .

Inversions conjecturally infeasible for many popular  $H$ 's.

Is second reaction better?

Long  $r$  is clear disadvantage.

Maybe outweighed by faster  $H$ ?

## Attack 4: Factorization by NFS.

Attacker hires computational number theorist to factor  $n$  using the number-field sieve.

Attacker chooses  $m'$ , signs  $m'$  same way as legitimate signer.

1978 RSA: “We recommend using 100-digit (decimal) prime numbers  $p$  and  $q$ , so that  $n$  has 200 digits.”

2005 Bahr Boehm Franke

Kleinjung: “We have factored RSA200 by GNFS.”

Attack 5: Leak detection.

Signer has many choices  
of signature for  $m$ :

$2^B$  choices of  $B$ -bit  $r$ ,

and then 4 choices of  $(e, f, s)$ .

Imagine idiotic signer

making successive bits of  $p$

visible to attacker by, e.g.,

copying them into bits of  $r$  or

into Jacobi symbol of  $s \bmod pq$ .

Evidently security depends on

choice of signing algorithm.

Many more attacks in literature.

Many (most?) of the attacks  
are *H*-generic:

attack works for every function *H*  
(or a large fraction of *H*'s)

if signer, attacker, verifier  
use an oracle for *H*.

It's quite embarrassing  
for a system to be broken  
by an *H*-generic attack  
faster than factorization!

Example: Signing-leak attacks  
are *H*-generic, embarrassing.

1987 Fiat Shamir:

Here's a signature system  
where embarrassment is limited.  
Can convert  $H$ -generic attack  
into factorization algorithm  
with only polynomial loss of  
efficiency and effectiveness.

1996 Bellare Rogaway:

Here's a signature system  
immune to embarrassment.  
Can convert  $H$ -generic attack  
into factorization algorithm  
with almost negligible loss of  
efficiency and effectiveness.

Many subsequent systems and “reductions in the random-oracle model.” Confusing terminology.

Common flaws in the theorems:

- Reductions aren't very tight.
- Tightness isn't quantified.
- Proofs have gaps, errors.
- The theorems don't apply to the fastest systems.

The point of this talk:

We can do better! Now have very tight proofs for some state-of-the-art RW variants.

(most recently 2006 Bernstein)

## Three state-of-the-art systems

“Fixed 0-bit unstructured RW”  
is immune to embarrassment.

“0-bit” : 0 bits in  $r$   
(despite 2002 Coron theorem  
that “FDH can’t be tight”).

“Unstructured” : Signer’s choice  
of  $(e, f, s)$  is uniform random,  
independent of  $(p, q)$ .

“Fixed” : Given same  $m$  again,  
signer chooses same signature.

For comparison, easily break

“variable 0-bit unstructured RW.”



“Fixed” needs memory  
for signatures of old  $m$ 's.

But, without memory, can  
produce indistinguishable results,  
assuming standard conjectures  
in secret-key cryptography:  
signer generates “random” bits  
by hashing  $m$  together with  
a secret independent of  $p, q$ .  
(1997 Barwood, 1997 Wigley)

Can hash  $m$  using Poly1305  
or forthcoming MAC1071;  
just a few cycles per byte.  
Scramble output with Salsa20.

“Fixed 1-bit principal RW”  
is immune to embarrassment.

“1-bit”: uniform random bit  $r$ ,  
independent of  $(p, q)$ .

“Principal”: Signer chooses  
 $e = 1$  when there’s a choice;  
 $f = 1$  when there’s a choice;  
and the unique  $s \in \mathbf{Z}/pq$   
that’s a square in  $\mathbf{Z}/pq$ .

(Same idea as 2003 Katz Wang  
reduction for fixed 1-bit RSA etc.,  
but need generalization beyond  
“claw-free permutation pairs.”)

“Fixed 128-bit |principal| RW”  
is immune to embarrassment.

“128-bit”: uniform random  
128-bit  $r$ , independent of  $(p, q)$ .

“|Principal|”: Signer chooses  
 $e = 1$  when there’s a choice;  
 $f = 1$  when there’s a choice;  
and  $s \in \{0, 1, \dots, (pq - 1)/2\}$   
with  $s$  or  $-s$  square in  $\mathbf{Z}/pq$ .

Implementation note: Can  
continue to avoid Euclid, Jacobi.

## Blind attacks

Consider algorithm  $A_1$  that, given  $pq \in \{2^{2048}, \dots, 2^{2049} - 1\}$  and  $h'$ , computes  $(e', f', s')$ .

How large is  $\Pr[e' f' (s')^2 = h']$  for uniform random 2048-bit  $h'$ ?

Build factorization algorithm  $A_0$ :  
choose uniform random  $(e, f, s)$ ;  
compute  $h' = e f s^2$ ;  
start over if  $h' \geq 2^{2048}$ ;  
compute  $(e', f', s') = A_1(pq, h')$ ;  
compute  $\gcd\{pq, s' - s\}$ .  
Comparable efficiency to  $A_1$ .

Define  $A_1$  as **successful**

if  $e' f'(s')^2 = h'$ .

If  $A_1$  is always successful

then  $e' f'(s')^2 = e f s^2$ ;

$s', s$  are coprime to  $pq$

with probability  $1 - \epsilon$ , tiny  $\epsilon$ ;

$s', s$  are independent square roots;

so  $\gcd\{pq, s' - s\} \in \{p, q\}$

with probability  $\geq (1 - \epsilon)/2$ .

More generally:

If  $\Pr[A_1 \text{ succeeds}] = \alpha$  for

uniform random 2048-bit  $h'$  then

$\Pr[A_0 \text{ factors } pq] \geq \alpha(1 - \epsilon)/2$ .

## Seeing a signature

Consider algorithm  $A_2$  that, given  $h, e, f, s, h', pq$  with  $h = efs^2$ , computes  $(e', f', s')$ .

How large is  $\Pr[e' f' (s')^2 = h']$  for independent uniform random 2048-bit  $h, h'$ ?

Three versions of question:

1. Unstructured  $(e, f, s)$ .
2. Principal  $(e, f, s)$ .
3. |Principal|  $(e, f, s)$ .

Analogy: Attack sees signature of  $m$ , forges signature of  $m' \neq m$ .

Intuition:  $A_2$  learns nothing from seeing  $h, e, f, s$ .

Formalization:

**Simulated signer**, given  $pq$ , generates random  $(h, e, f, s)$  with exactly the right distribution.

Thus can build  $A_1$  from  $A_2$ .

$A_1$ , given  $pq, h'$ ,

generates  $h, e, f, s$ ;

runs  $A_2$  with  $h, e, f, s, h', pq$ .

$\Pr[A_2 \text{ succeeds}] = \Pr[A_1 \text{ succeeds}]$ .

How to generate  $h, e, f, s$ ?

How does simulated signer work?

For unstructured:

Generate uniform random  $e, f, s$ ;

compute  $h = efs^2$ ;

start over if  $h \geq 2^{2048}$ .

For principal:

Generate uniform random  $e, f, x$ ;

compute  $s = x^2$ ;

tweak  $e, f$  if  $\gcd\{x, pq\} > 1$ ;

compute  $h = efs^2$ ;

start over if  $h \geq 2^{2048}$ .

For  $|principal|$ : Similar.



## Seeing many signatures

Consider  $A_3$  that, given  $pq, h'$ ,  
 $(h_1, e_1, f_1, s_1), \dots, (h_q, e_q, f_q, s_q)$   
with each  $h_i = e_i f_i s_i^2$ ,  
computes  $(e', f', s')$ .

How large is  $\Pr[e' f' (s')^2 = h']$   
for independent uniform  
random  $h_1, \dots, h_q, h'$ ?

Again three versions of question.

Analogy: Attack sees signatures  
of distinct  $m_1, \dots, m_q$ , forges  
signature of  $m' \notin \{m_1, \dots, m_q\}$ .

Intuition:  $A_3$  learns nothing from seeing  $h_i, e_i, f_i, s_i$ .

Formalize exactly as before.

Build  $A_1$  from  $A_3$

by generating  $h_i, e_i, f_i, s_i$  using same simulated signer.

Warning regarding distinctness:

Reasonable to vary problem, forcing  $h_i = h_j$  for various  $(i, j)$ ; analogous to attacker forcing repetitions in  $m_1, \dots, m_q$ .

Same simulated signer is fine for fixed signatures but not for variable signatures.

## Hashing first

The “FDH” case ( $B = 0$ , no  $r$ ):

Consider algorithm  $A_4$  that  
is given  $pq, h_1, h_2, \dots, h_{q+1}$ ;  
selects  $i$ , sees  $e_i, f_i, s_i$ ;  
repeats for any number of  $i$ 's;  
computes  $i', e', f', s'$ .

Algorithm is **successful**

if  $h_{i'} = e' f' (s')^2$  and  $i'$  is new.

How large is  $\Pr[A_4 \text{ succeeds}]$ ?

Analogy: Attack chooses  
messages to feed to legitimate  
signer *after* inspecting hashes.

Conventional treatment of FDH  
(1996 Bellare Rogaway, etc.):

Easily build  $A_3$  from  $A_4$   
by guessing  $i'$  in advance.

Guess is correct  
with probability  $1/(q + 1)$ .

Can increase probability  
from  $1/\#\{\text{hash queries}\}$   
to  $\Theta(1/\#\{\text{signing queries}\})$ .

(2000 Coron)

“We show . . . it is not possible to  
further improve the security proof  
of FDH.” (2002 Coron)

New treatment (2006 Bernstein):

Easily build  $A_0$  from  $A_4$

for unstructured signatures.

Tight! No guessing required.

Warning: construction fails  
for principal,  $|\text{principal}|$ , etc.

For each  $i \in \{1, \dots, q + 1\}$

choose independent uniform  
random  $(e_i, f_i, s_i)$ .

No need to distinguish  $i'$   
from the signing queries.

2002 Coron theorem

assumes “unique signature.”

Signatures here aren't “unique.”

The non-“FDH” case,  $B \geq 1$ :

Consider algorithm  $A_4$  that  
is given random access to  $pq$ ,  
 $h_1(0), \dots, h_1(2^B - 1)$ ,  
 $h_2(0), \dots, h_2(2^B - 1), \dots,$   
 $h_{q+1}(0), \dots, h_{q+1}(2^B - 1)$ ;  
selects  $i$ , sees  $r_i, h_i(r_i), e_i, f_i, s_i$ ;  
repeats for any number of  $i$ 's;  
computes  $i', e', f', r', s'$ .

Algorithm is **successful** if

$h_{i'}(r') = e' f'(s')^2$  and  $i'$  is new.

Analogy:  $h_i(r) = H(r, m_i)$ ,

distinct  $m_1, \dots, m_{q+1}$ .

Unstructured: as for  $B = 0$ .

What about principal etc.?

Resort to 2003 Katz Wang.

Katz-Wang theorem is limited to  
“claw-free permutation pairs”  
but idea also works for RW.

Build  $h_i(r)$  for  $r \neq r_i$   
using unstructured simulator.

Build  $h_i(r)$  for  $r = r_i$   
using principal simulator.

If  $i'$  new then  $r', r_{i'}$  independent  
so  $\Pr[r' \neq r_{i'}] = 1 - 1/2^B$ .

Probability loss is 2 for  $B = 1$ ;  
converges rapidly to 1 as  $B \rightarrow \infty$ .

## The final reduction

Consider attack  $A_5$  that  
is given  $H$  oracle and  $pq$ ;  
selects  $m$ , sees signature;  
repeats for any number of  $m$ 's;  
computes  $(m', e', f', r', s')$ .

Easily convert into  $A_4$ ,  
thanks to fixed signatures.

$\Pr[A_5 \text{ succeeds}] = \Pr[A_4 \text{ succeeds}]$   
assuming uniform random  $H$ .

Credit for exposition:

I stole  $A_4$  shape from

2004 Koblitz Menezes ("RSA1").



## The “random-oracle” debate

“These proofs are required!  
Security must be proven!  
Cryptosystems without theorems  
are bad cryptosystems!”

“No! These proofs are useless!  
Some attacks aren’t generic!”

My view: Insisting on proofs  
exposes many security problems,  
weeds out many awful systems,  
saves time for cryptographers.

Reconsider this insistence  
if it sacrifices system speed;  
but today there’s no conflict.