Difficulty and Inapplicability of the RSA Problem

Dan Brown

Certicom Research

Thursday, November 30, 2006

Fields Institute, Toronto, Ontario, Canada

Workshop on Cryptography: Underlying Mathematics, Provability and Foundations

Two technical reports:

Breaking RSA May Be As Difficult As Factoring

`http://eprint.iacr.org/2005/380` (also CACR 2005-37)

and

The Unprovable Security of RSA-OAEP in the Standard Model

`http://eprint.iacr.org/2006/223`

Neither refereed, yet.

Recent related refereed work:

Gregor Leander and Andy Rupp, *On the Equivalence of RSA and Factoring w.r.t. Generic Ring Algorithms*, ...

Alexandra Boldyreva and Marc Fischlin, *On the Security of OAEP*, ...

Pascal Paillier and Jorge L. Villar, *Trading One-Wayness against Chosen-Ciphertext Security in Factoring-Based Encryption*, ...

... Asiacrypt 2006, December 4–6.

Part I

The RSA Problem is Almost as Hard as Factoring

If factoring is difficult, then

no efficient algorithm can take

an RSA public key as input and then output

a straight line program (with equality based branching)

that efficiently solves

the RSA problem for the given public key, provided that

the public exponent has a small prime factor.

Miller, Rivest, Shamir, Adleman, de Laurentis, Coron, May: If factoring is hard, no efficient algorithm takes an RSA public key and outputs the private exponent.

Rabin: If factoring is hard and public exponent even, then no efficient algorithm can solve the corresponding RSA problem.

Coppersmith, Hastad, and others: Very low public exponent (such as three) makes RSA vulnerable to various indirect attacks.

Boneh and Venkatesan: No SLP reduction can prove that solving RSA problem is as hard as factoring, provided public exponent is odd and small. *Breaking RSA May Be Easier Than Factoring*

RSA Problem (aka e^{th}-root problem)

Input: (N, e, y) where

$N = pq$ and p, q prime; e a fixed RSA public exponent, such as $e = 2^{16} + 1$ or $e = 3$; y any random integer.

Output: x such that $x^e \equiv y \mod N$.

Usually ensure unique roots by arranging that

$$\gcd((p-1)(q-1), e) = 1. \tag{1}$$

Integer factoring problem is to find p given N.

A long standing open question in cryptology:

Is the RSA problem as difficult as factoring?

Straight Line Program (SLP): What They Are

A fixed sequence F of ring operations:

$F = ((i_k, j_k, \circ_k))_{1 \leqslant k \leqslant L}$

where $-1 \leqslant i_k, j_k < k$ and $\circ_k \in \{+, -, \times\}$.

Length is L.

Example: $((0, 0, \times), (-1, 1, -))$.

Action of an SLP: What They Do

Program: F of length L

Input: x

Output: $F(x)$

Action: Let $x_{-1} = 1$ and $x_0 = x$. For $k = 1$ to L, let $x_k = x_{i_k} \circ_k x_{j_k}$. Let $F(x) = x_L$.

No ifs ... $\Rightarrow$ no branching $\Rightarrow$ straight line.

Example SLP and its action:

If $F = ((0, 0, \times), (-1, 1, -))$, then

$x_{-1} = 1, \quad x_0 = x, \quad x_1 = x_0 \times x_0 = x^2$, and

$F(x) = x_2 = x_{-1} - x_1 = 1 - x^2$.

Note: program $((-1, 0, +), (-1, 0, -), (1, 2, \times))$ has same action, because $1 - x^2 = (1 + x)(1 - x)$.

SLP Variants:

Multi-input SLP: implements multi-variable integer polynomials.

SLP with division: implements rational functions in any near field, such as $\mathbb{Z}/N$.

SLP with equality testing: branches based on equality (but not size).

Proof Idea: Rabin's argument for even $e = 2$: for random r there are four values of $\sqrt{r^2}/r$. Two are 1 and -1, while the other two are $(1, -1)$ and $(-1, 1)$ taken modulo (p, q). Either of the latter two reveals the factorization of n.

In RSA, however, N is chosen so that unique e^{th} roots exist. To get around this, we will extend the ring $\mathbb{Z}/N$ in such a way that multiple roots exist. $\square$

Missing Steps: What is the extension ring?

What do we do in the extension ring?

Why are we limited to straight line programs?

Theorem: Let $N = pq$ and e form an RSA public key with $e \ll N$. Let F be an SLP of length L such that

$$\Pr\left[r \xleftarrow{\$} \mathbb{Z}/N \middle| r = F(r^e)\right] = \mu. \tag{2}$$

We can use F to build a multi-input SLP G of length about $3\phi(e)^2 L$ such that

$$\Pr\left[\mathbf{r} \xleftarrow{\$} (\mathbb{Z}/N)^{\phi(e)} \middle| p = \gcd(N, G(\mathbf{r}))\right] \approx \mu \frac{2}{\phi(e)} \frac{(e-1)}{e} \frac{(E-1)}{E} \tag{3}$$

where $E = \exp(1)$.

Proof Sketch: Let $m(X) \in \mathbb{Z}/N[X]$ be monic of degree $\phi(e)$. If $m(X)$ has a root over $\mathbb{F}_p$ but $m(X)$ is irreducible over $\mathbb{F}_q$, then

$$R = \mathbb{Z}/N[X]/m(X) \cong \mathbb{F}_p \times \mathbb{F}_{p^{d_2}} \times \cdots \times \mathbb{F}_{p^{d_t}} \times \mathbb{F}_{q^{\phi(e)}}, \qquad (4)$$

for some $d_2, \ldots, d_t$ summing to $\phi(e) - 1$.

In the copy of $\mathbb{F}_p$, the SLP F computes the unique e^{th} roots with probability at least μ, because F finds roots in $\mathbb{Z}/N \cong \mathbb{F}_p \times \mathbb{F}_q$.

In the copy of $\mathbb{F}_{q^{\phi(e)}}$, by Euler's theorem, each e^{th} power has e roots, so $F(r^e) = r$ with probability at most $\frac{1}{e}$.

Euler's theorem: $e \mid q^{\phi(e)} - 1 = \#\mathbb{F}^*_{q^{\phi(e)}}$.

Let M be the set of monic integer polynomial of degree $\phi(e)$. Modulo p,

$$\Pr\left[m \xleftarrow{\$} M \middle| m \text{ has root over } \mathbb{F}_p\right] \approx \frac{1}{1!} - \frac{1}{2!} + \cdots \pm \frac{1}{\phi(e)!}, \quad (5)$$

provided $\phi(e) \ll p$. For moderately large $\phi(e)$, the alternating sum is close to $1 - \frac{1}{E} = \frac{E-1}{E}$. Modulo q,

$$\Pr\left[m \xleftarrow{\$} M \middle| m \text{ irreducible over } \mathbb{F}_q\right] \approx \frac{1}{\phi(e)} \quad (6)$$

Algorithm: $m(X) \xleftarrow{\$} M$ and $r \xleftarrow{\$} R = \mathbb{Z}/N[X]/m(X)$, compute

$$z = F(r^e) - r. \tag{7}$$

With probability $\mu(1 - \frac{1}{e})(1 - \frac{1}{E})\frac{1}{\phi(e)}$, then perhaps:

$$\begin{aligned} z &\stackrel{?}{\equiv} 0 \bmod p \\ z &\not\equiv 0 \bmod q \end{aligned} \tag{8}$$

Not quite ...

Possible that $z(X) \not\equiv 0 \bmod p$. Really only know that $z(X)$ and $m(X)$ have a common root modulo p.

The resultant detects common roots:

$$p = \gcd(N, \mathrm{Res}(z(X), m(X))) \tag{9}$$

Implement F on R as an SLP G on vectors $(\mathbb{Z}/N)^{\phi(e)}$.

If $\phi(e)$ sufficiently small, compute resultant as extra SLP steps in G. Otherwise, just modify theorem statement. $\square$

Corollary: If factoring is difficult, no efficient algorithm A exists that takes RSA public key (N, e) as input, where there exists integer $1 < f \ll N$ with $f \mid e$, and outputs SLP F that efficiently solves the (N, e) RSA problem.

Proof: If F finds e^{th} roots, then $G(X) = F(X)^{e/f}$ finds f^{th} roots. Now apply the theorem. $\square$

Impact for typical RSA parameters:

$e = 3$, $N \approx 2^{1024}$: Factoring assumed to cost about 2^{80}. Solving RSA problem with an SLP costs at least about 2^{75}.

$e = 2^{16} + 1$, $N \approx 2^{1024}$: Factoring assumed to cost about 2^{80}. Solving RSA problem with an SLP costs at least about 2^{30}.

Cost of 2^{30} is feasible, but we do not have an attack. Theorem only fails to provide meaningful security assurance for $e = 2^{16}+1$.

Recommendations, positive:

Hybrid exponent $e = 3(2^{16} + 1)$.

RSA public key operations almost as efficient as $2^{16} + 1$,

RSA problem at least as hard as $e = 3$,

RSA problem at least as hard as $e = 2^{16} + 1$,

Indirect RSA attacks foiled by large size.

Aside: Underlying hard problems RSA versus ECC:

Problem	RSA	ECC
Private key	IFP	ECDLP
Cryptanalyze	RSAP	ECDHP
Forge	RSAP	ECSLP*

den Boer, Maurer, Wolf, Boneh, Lipton, Gallant, B: Solving the ECDHP often almost as hard as solving the ECDLP.

Nechaev, Shoup, B: Solving ECDLP with a generic algorithm difficult.

Opposite but not Contradictory:

[BV] **Boneh and Venkatesan**: *RSA problem may be easier than factoring*: SLP reduction showing that RSA problem for small, odd e, is as hard as factoring impossible, unless factoring easy. For this, they use a metareduction.

Note: [BV] does not contradict de Laurentis, Miller, Rivest, Shamir, Adleman, or Rabin's positive results, but no common thread of SLP. Burden to justify lack of contradiction belongs to newer work.

To wit, [BV] metareduction only applies to reductions that treat the root finder as an oracle. My reduction looks inside root finder via SLP description, so is not covered by [BV].

Comparing results for relation between hardness of finding e^{th} roots and factoring:

Smallest Factor of e	Any	Miller, de Laurentis, Rivest, Shamir, Adleman, Coron, May		
	Small		Me	$-, *,$ Boneh, Venkatesan
	Two			Rabin
		Private Exponent	SLP	Any
		Root-Finding Algorithm		

Conclusion (for Part I)

Is the RSA problem as difficult as factoring?

Still open question ...

But door has been just nudged a little sliver more closed.

Special Acknowledgment (for Part I)

Steven Galbraith's generous guidance led me to this work.

Part II

Breaking RSA-OAEP May Be Easier Than the RSA Problem

What is RSA-OAEP?

Encryption:

$$c \equiv \mathrm{OAEP}(m)^e \bmod n \tag{10}$$

Decryption:

$$m = \mathrm{OAEP}^{-1}\left(c^{1/e} \bmod n\right) \tag{11}$$

What is OAEP(·)?

1. Two-step Feistel ladder built on secure hash functions, such as SHA-1 or SHA-256.

2. Probabilistic (randomized) function.

3. Adds redundancy.

Attributes leading to security proof in random oracle model.

None of these details matter for this talk!

Only property that matters for this talk is that

$$\mathrm{OAEP}^{-1}(\cdot) \tag{12}$$

exists.

Actually, more precisely, what matters is that a left inverse exists:

$$\mathrm{OAEP}^{-1}(\mathrm{OAEP}(m)) = m \tag{13}$$

Result applies to any undoable function, not just OAEP.

Applies to PKCS #1 version 1.5 (more widely deployed than OAEP, anyway). Applies to RSA-KEM.

OAEP's Claim to Fame:

Provably secure (IND-CCA2) in the random oracle model (for hash functions).

Reduction uses an RSA-OAEP breaker to solve RSA problem.

Fall to Infamy: Flaw (gap?) in original Bellare-Rogaway proof found by Shoup.

Famously fixed up by Shoup, Fujisaki, Okamoto, Pointcheval, and Stern.

The end .

But wait ...

SHA-1 not a random oracle. Neither is SHA-256.

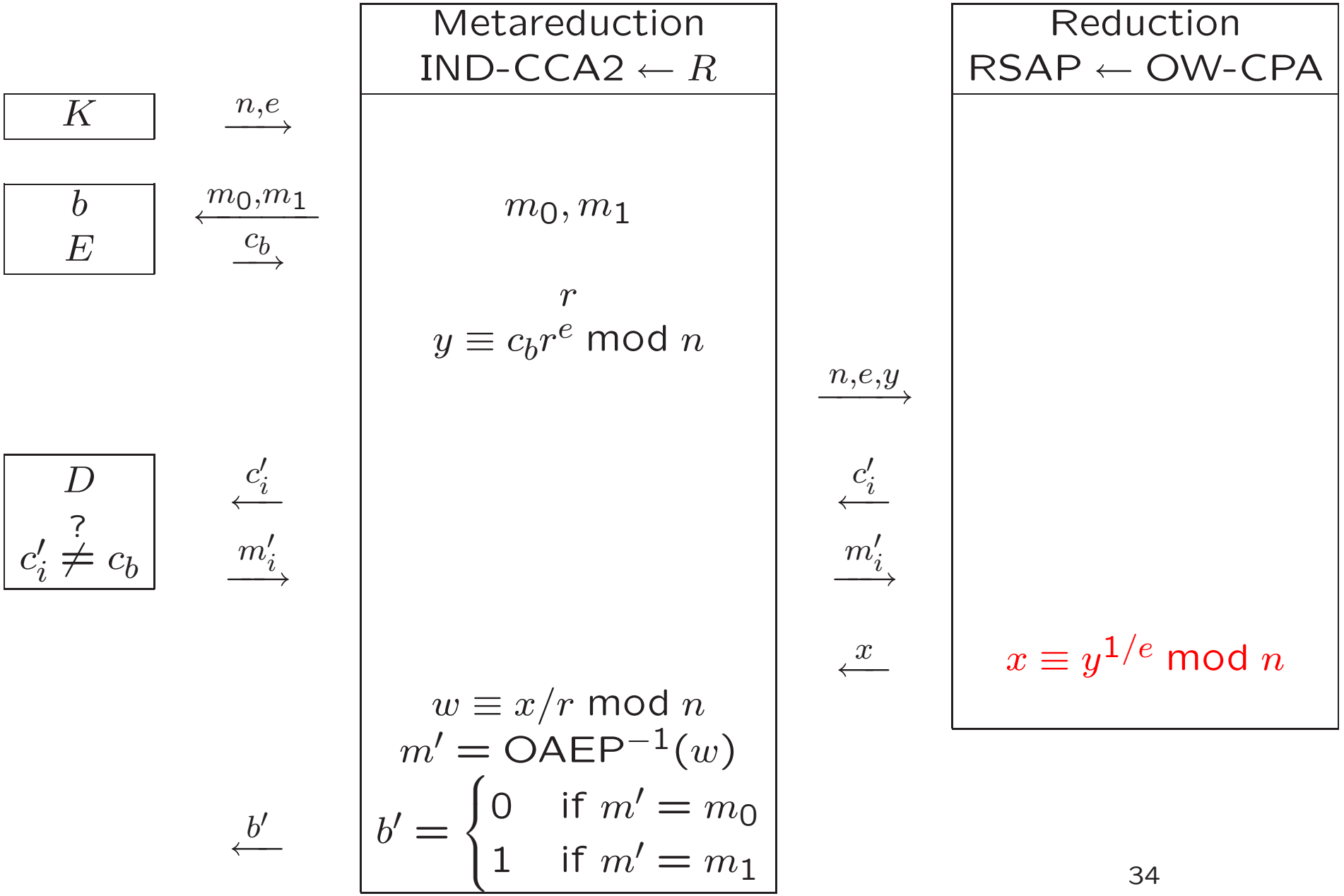
Random oracles are practical for provers but impractical for implementers.

Can one prove RSA-OAEP secure without random oracles?

This is the standard model, in which more and more often cryptologists have striven to obtain security.

In other words, can RSA-OAEP be proven secure in the standard model?

Let's see ...


Metareduction
IND-CCA2 ← R
Reduction
RSAP ← OW-CPA
K
n,e
b
E
m_0,m_1
c_b
m_0, m_1
r
$y \equiv c_b r^e \bmod n$
n,e,y
D
?
$c'_i \neq c_b$
c'_i
m'_i
c'_i
m'_i
x
$x \equiv y^{1/e} \bmod n$
$w \equiv x/r \bmod n$
$m' = \mathrm{OAEP}^{-1}(w)$
$b' = \begin{cases} 0 & \text{if } m' = m_0 \\ 1 & \text{if } m' = m_1 \end{cases}$
b'

Why doesn't this contradict the random oracle model proofs?

Why can't our metareduction M use the random oracle model reduction R_{OM} to attack RSA-OAEP?

Reduction R_{OM} has extra inputs (oracle queries from an OW-CPA attacker) and outputs (simulated oracle responses).

Metareduction M does not have metareduction for the corresponding communication.

Feel free to try to extend M — could be fun.

Note well:

1. Metareduction can be made into OW-CCA2 attacker.

2. Reduction for IND-CCA2 security $\Rightarrow$ reduction for OW-CPA security.

Limitations:

CCA2 is crucial in the metareduction.

A OW-CPA security reduction $\not\Rightarrow$ IND-CCA1 attack (afaik).

If reduction algorithm R exists, then Alice could use it to solve any instance of the RSA problem for his public key just by sending him a ciphertext to decrypt. But if she can do solve the RSA problem, she can decrypt any existing ciphertext sent to Bob. So ...

Isn't this metareduction blindingly obvious?

Yes, it's obvious, but so what?

It's still important, even if it is obvious. Occasionally the obvious is important.

At least it's important enough to warn newbies before they go off and try to prove security of RSA-OAEP in the standard model based on the RSA problem.

But, isn't this already known?

Perhaps in folklore.

Williams, Rivest, Shamir and Adleman noted that the provable security of Rabin(-Williams) encryption leads to an adaptive attack.

The attack seems to have been prevented with padding schemes like OAEP.

Myth may have taken root that it is therefore (not) provable in standard model too.

Another metareduction

A strong reduction for RSA-OAEP treats two functions used inside of OAEP($\cdot$) as random oracles.

This specialization of reductions is somewhat analogous to what [BV] do with SLP reductions.

Suppose R is a strong reduction showing RSA-OAEP to be OW-CPA secure if the RSA problem is hard.

A metareduction M is constructed that uses R to solve the RSA problem.

The metareduction M mimics the Bellare-Rogaway proof of security for RSA-OAEP.

In a nutshell, M uses simulated random oracles to eliminate any advantage R can gain from access to an RSA decryption oracle.

This is essentially how the IND-CCA$\{1,2?\}$ security of RSA-OAEP is proven in the random oracle model.

So, when R solves the RSA problem, metareduction M outputs this solution to the RSA problem.

Conclusion to Part II

Impossible: A reduction that proves, in the standard model, based on the assumption that the RSA problem is hard, that RSA-OAEP is

- IND-CCA2 secure,
- OW-CCA2 secure,
- OW-CPA secure unless it is also OW-CCA2 insecure.
- OW-CPA secure — using a strong reduction.

Still possible: Alternative proofs based on different assumptions.

Two conclusions = confusion.

Taken together, what does it all mean?

Heavy questions call for enlightening answers.

... so, please forgive me,

it's time for ...

Rhyme for a metaconclusion

One big step forward, we took in part one:

RSA's a bit harder, so what's left to do?

Two backward steps, we took in part two.

Third roots are no use; it's back to square none,

For OAEP is again undone.

O wild goose chase! O roller coaster ride!

What more obvious secrets in cryptology hide?

Sorry![3] mod doggerel

Thanks again, I hope you had fun!