Factoring RSA Modulus using Prime Reconstruction from Random Known Bits

S. Maitra, S. Sarkar and S. Sen Gupta

Cryptology Research Group, ASU
Indian Statistcal Institute, Kolkata

May 3, 2010

Background

RSA Framework

KEY-GEN

- Large (512 bits) primes p, q and $N = pq$
- $\phi(N) = (p-1)(q-1)$ and $\gcd(e, \phi(N)) = 1$
- $d = e^{-1} \bmod \phi(N)$
- Publish $\langle N, e \rangle$ and keep $\langle N, d \rangle$ Private

ENCRYPTION: $C = M^e \bmod N$ for $M \in \mathbb{Z}_N$

DECRYPTION: $M = C^d \bmod N$

Efficient Decryption: CRT-RSA (uses $d_p = d \bmod p-1$ and $d_q = d \bmod q-1$)

Motivation

RSA PROBLEM
Given RSA Public Key $\langle N, e \rangle$ and $C = M^e \bmod N$, compute M.

FACTS

- Easy to prove: "Factoring $N = pq$" $\geq$ "RSA Problem"
- As of 2010: Factoring N is *hard* for $\log_2(N) > 768$
- Practical RSA: $\log_2(N) = 1024, 2048$ (recommended)

Motivation

RSA PROBLEM
Given RSA Public Key $\langle N, e \rangle$ and $C = M^e \bmod N$, compute M.

FACTS

- Easy to prove: "Factoring $N = pq$" $\geq$ "RSA Problem"
- As of 2010: Factoring N is *hard* for $\log_2(N) > 768$
- Practical RSA: $\log_2(N) = 1024, 2048$ (recommended)

QUESTIONS

- Does factoring N get easier if we know some bits of p, q?
- How do we know the bits of p, q in the first place?

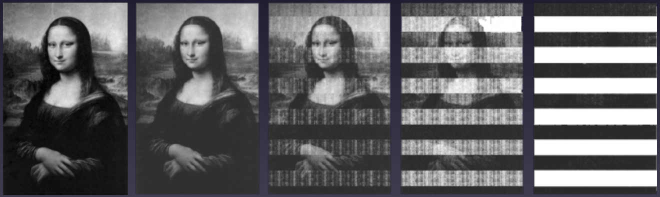
Coldboot Attack

Ref: Lest We Remember: Cold Boot Attacks on Encryption Keys.

Halderman et. al. Princeton University. 2008.

Base Logic

- System memory can be thought of as an array of capacitors
- Capacitors *take time* to charge or discharge completely
- Information can be tapped from retained charge in capacitors



Coldboot Attack

How good is it?

- Works against popular Disk Encryption systems
- Reconstruction of DES key - Halderman et. al.
- Reconstruction of AES key - Halderman et. al.
- Reconstruction of RSA keys - Heninger and Shacham

Coldboot Attack

How good is it?

- Works against popular Disk Encryption systems
- Reconstruction of DES key - Halderman et. al.
- Reconstruction of AES key - Halderman et. al.
- Reconstruction of RSA keys - Heninger and Shacham

Our Focus

- Study and analyze Heninger and Shacham (Crypto 2009)
- Suggest improvements to their results
- Propose related scheme(s) for RSA prime reconstruction

Reconstruction from LSBs

General Idea

Due to: Nadia Heninger and Hovav Shacham [Crypto 2009]
"Reconstructing RSA Private Keys from Random Key Bits"

GOAL: Reconstruct bits of primes starting at the LSB.

NOTE: Total search space (tree) size $= 2^{512}$ (for 1024 RSA)

- 4 possible choices for each pair of bits of p, q
- known RSA equation $N = pq$ rules out 2 choices

IDEA: Search tree can be pruned if we know some bits of p, q.

General Idea

Due to: Nadia Heninger and Hovav Shacham [Crypto 2009]
"Reconstructing RSA Private Keys from Random Key Bits"

GOAL: Reconstruct bits of primes starting at the LSB.

NOTE: Total search space (tree) size $= 2^{512}$ (for 1024 RSA)

- 4 possible choices for each pair of bits of p, q
- known RSA equation $N = pq$ rules out 2 choices

IDEA: Search tree can be pruned if we know some bits of p, q.

How many bits of p, q do we need to know?

Solution Tree

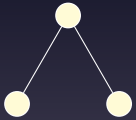
Notation

- $p[i], q[i]$ - i-th bits of p, q ($p[0] = q[0] = 1$ are LSBs)
- p_i, q_i - partial solution for p, q through bits $0 - i$
- Level i - all possibilities for p_i, q_i in the Search tree

Normal Branching

4 naive choices for $p[i], q[i]$ reduces to 2 as the known relation $N = pq$ gives

$$p[i] + q[i] = (N - p_{i-1}q_{i-1})[i] \bmod 2$$



Solution Tree

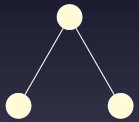
Notation

- $p[i], q[i]$ - i-th bits of p, q ($p[0] = q[0] = 1$ are LSBs)
- p_i, q_i - partial solution for p, q through bits $0 - i$
- Level i - all possibilities for p_i, q_i in the Search tree

Normal Branching

4 naive choices for $p[i], q[i]$ reduces to 2 as the known relation $N = pq$ gives

$$p[i] + q[i] = (N - p_{i-1}q_{i-1})[i] \bmod 2$$



It gets better if some bits are known ...

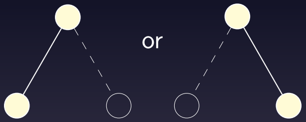
Branching Analysis

The Vital Relation

$$p[i] + q[i] = (N - p_{i-1}q_{i-1})[i] \bmod 2 \tag{1}$$

Improvised Branching

If either $p[i]$ or $q[i]$ is known,
Equation 1 fixes the other bit.



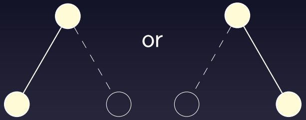
Branching Analysis

The Vital Relation

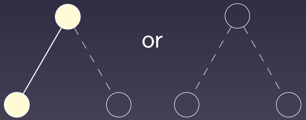
$$p[i] + q[i] = (N - p_{i-1}q_{i-1})[i] \bmod 2 \qquad (1)$$

Improvised Branching

If either $p[i]$ or $q[i]$ is known,
Equation 1 fixes the other bit.



If both $p[i]$ and $q[i]$ are known,
Equation 1 is either satisfied or not.



Branching Analysis

COLDBOOT: α fraction of p bits and β fraction of q bits known.

BRANCHING STATISTICS

- None of $p[i], q[i]$ known: 2 Branches, Prob $= (1-\alpha)(1-\beta)$.
- Only $p[i]$ known: 1 Branch, Prob $= \alpha(1-\beta)$.
- Only $q[i]$ known: 1 Branch, Prob $= (1-\alpha)\beta$.
- Both $p[i], q[i]$ known: γ Branches, Prob $= \alpha\beta$. $(1 > \gamma > 0)$

Branching Analysis

COLDBOOT: α fraction of p bits and β fraction of q bits known.

BRANCHING STATISTICS

- None of $p[i], q[i]$ known: 2 Branches, Prob $= (1-\alpha)(1-\beta)$.
- Only $p[i]$ known: 1 Branch, Prob $= \alpha(1-\beta)$.
- Only $q[i]$ known: 1 Branch, Prob $= (1-\alpha)\beta$.
- Both $p[i], q[i]$ known: γ Branches, Prob $= \alpha\beta$. $(1 > \gamma > 0)$

Total number of branches at Level i from each node at Level $i-1$:

$$2(1-\alpha)(1-\beta) + \alpha(1-\beta) + (1-\alpha)\beta + \gamma\alpha\beta = 2 - \alpha - \beta + \gamma\alpha\beta$$

Bit Requirement

Growth factor of the Search Tree: $2 - \alpha - \beta + \gamma\alpha\beta$

Natural Idea: Keep the growth factor ≈ 1 to restrict growth.

Assuming $\alpha = \beta$,

$$2 - \alpha - \beta + \gamma\alpha\beta \approx 1 \quad \Rightarrow \quad \alpha = \beta \approx \frac{1 - \sqrt{1-\gamma}}{\gamma}$$

Bit Requirement

Growth factor of the Search Tree: $2 - \alpha - \beta + \gamma\alpha\beta$

Natural Idea: Keep the growth factor ≈ 1 to restrict growth.

Assuming $\alpha = \beta$,

$$2 - \alpha - \beta + \gamma\alpha\beta \approx 1 \quad \Rightarrow \quad \alpha = \beta \approx \frac{1 - \sqrt{1-\gamma}}{\gamma}$$

Experimental observation shows $\gamma \approx 0.5$. (open problem to prove)
Assuming this true, we get $\alpha = \beta \approx 2 - \sqrt{2} \approx 0.5857$.

Bit Requirement

Growth factor of the Search Tree: $2 - \alpha - \beta + \gamma\alpha\beta$

NATURAL IDEA: Keep the growth factor ≈ 1 to restrict growth.

Assuming $\alpha = \beta$,

$$2 - \alpha - \beta + \gamma\alpha\beta \approx 1 \quad \Rightarrow \quad \alpha = \beta \approx \frac{1 - \sqrt{1-\gamma}}{\gamma}$$

Experimental observation shows $\gamma \approx 0.5$. (open problem to prove)
Assuming this true, we get $\alpha = \beta \approx 2 - \sqrt{2} \approx 0.5857$.

Knowing 59% of bits of p, q is enough to reconstruct the primes.

Specific Cases

Case 1: Bits from just one of the primes are known (50%)

- No results till date if random bits are known.
- Requires contiguous half of one prime (Boneh, Coppersmith).

Specific Cases

Case 1: Bits from just one of the primes are known (50%)

- No results till date if random bits are known.
- Requires contiguous half of one prime (Boneh, Coppersmith).

Case 2: Bits are known in complementary fashion (25%)

- Either $p[i]$ or $q[i]$ is known at each level.
- This implies that branching is *always* just 1.
- Requires 50% of lower halves of p, q.

Specific Cases

Case 1: Bits from just one of the primes are known (50%)

- No results till date if random bits are known.
- Requires contiguous half of one prime (Boneh, Coppersmith).

Case 2: Bits are known in complementary fashion (25%)

- Either $p[i]$ or $q[i]$ is known at each level.
- This implies that branching is *always* just 1.
- Requires 50% of lower halves of p, q.

Case 3: Bits are known at random positions (30%)

- We need to construct only *half* of the primes from LSB.
- Then, use the lattice based result by Boneh et. al.
- Requires 59% of lower halves of p, q.

Experiments

Size $\|p\|, \|q\|$	Known α, β	Target t	Final W_t	max W_i	Avg. γ
256, 256	0.5, 0.5	128	30	60	0.56
256, 256	0.47, 0.47	128	106	1508	0.54
256, 256	0.45, 0.45	128	6144	6144	0.49
512, 512	0.5, 0.5	256	352	928	0.53
512, 512	0.5, 0.5	256	8	256	0.55
512, 512	0.55, 0.45	256	37	268	0.51
512, 512	0.55, 0.45	256	64	334	0.51
512, 512	0.6, 0.4	256	1648	13528	0.55
512, 512	0.6, 0.4	256	704	5632	0.56
512, 512	0.7, 0.3	256	158	1344	0.53
512, 512	0.7, 0.3	256	47	4848	0.52
1024,1024	0.55, 0.55	512	1	352	0.53
1024,1024	0.53, 0.53	512	16	764	0.53
1024,1024	0.51, 0.51	512	138	15551	0.54
1024,1024	0.51, 0.5	512	17	4088	0.52

Specific Cases

Case 4: Bits are known in a Regular Pattern (25%)

- Pattern: U bits of both unknown, P bits of p known, Q bits of q known, K bits of both known.
- Growth of tree at Level T:

$$W_T \approx \left[2^{U-K}\right]^{\frac{T}{U+P+Q+K}} = 2^{\frac{T(U-K)}{U+P+Q+K}}$$

- Required $\frac{P+K}{U+P+Q+K}$ fraction of p and $\frac{Q+K}{U+P+Q+K}$ fraction of q.
- For $P = Q$, $U = K$, this means 50% of lower halves of p, q.

Specific Cases

Case 4: Bits are known in a Regular Pattern (25%)

- Pattern: U bits of both unknown, P bits of p known, Q bits of q known, K bits of both known.
- Growth of tree at Level T:

$$W_T \approx \left[2^{U-K}\right]^{\frac{T}{U+P+Q+K}} = 2^{\frac{T(U-K)}{U+P+Q+K}}$$

- Required $\frac{P+K}{U+P+Q+K}$ fraction of p and $\frac{Q+K}{U+P+Q+K}$ fraction of q.
- For $P = Q$, $U = K$, this means 50% of lower halves of p, q.

Case 5: Bits are known only at the top half - discussed later.

Specific Cases

Case 4: Bits are known in a Regular Pattern (25%)

- Pattern: U bits of both unknown, P bits of p known, Q bits of q known, K bits of both known.
- Growth of tree at Level T:

$$W_T \approx \left[2^{U-K}\right]^{\frac{T}{U+P+Q+K}} = 2^{\frac{T(U-K)}{U+P+Q+K}}$$

- Required $\frac{P+K}{U+P+Q+K}$ fraction of p and $\frac{Q+K}{U+P+Q+K}$ fraction of q.
- For $P = Q$, $U = K$, this means 50% of lower halves of p, q.

Case 5: Bits are known only at the top half - discussed later.

Case 6: Large chunk of bits not known at the beginning - problem!

Missing Bits Issue

Suppose we are missing u contiguous bits of both p, q.

We may miss these bits

- at the very beginning (bits 1 to u), or
- somewhere in the middle (bits $k + 1$ to $k + u$)

In either case, size of search tree grows to at least 2^u.

Missing Bits Issue

Suppose we are missing u contiguous bits of both p, q.

We may miss these bits

- at the very beginning (bits 1 to u), or
- somewhere in the middle (bits $k+1$ to $k+u$)

In either case, size of search tree grows to at least 2^u.

If u is large enough ($u \geq 50$), this will require huge memory (2^{50}) to store the search tree, even one level at a time.

If storage fails, the reconstruction algorithm fails!

Lattice Solution

THEOREM (In simple words)

- *τl_N many meast significant bits are unknown for primes p, q*
- *the subsequent ηl_N bits are known for both*

The τl_N bits can be recovered in poly$(\log N)$ time if $\eta > 2\tau$.

PROOF OUTLINE

- Let p_0, q_0 known and p_1, q_1 unknown portions of p, q.

$$\left(2^{\tau l_N} p_0 + p_1\right)\left(2^{\tau l_N} q_0 + q_1\right) = N \bmod 2^{(\tau+\eta) l_N}$$

- Solve $f(x, y) = \left(2^{\tau l_N} p_0 + x\right)\left(2^{\tau l_N} q_0 + y\right) - N$ over $\mathbb{Z}_T$.
- Lattice techniques to solve bivariate modular polynomial.

Lattice Solution

# of Unknown bits (τl_N)	# of Known bits (ηl_N)	Time in Seconds		
		LLL Algo	Resultant	Root
40	90	36.66	25.67	< 1
50	110	47.31	35.20	< 1
60	135	69.23	47.14	< 1
70	155	73.15	58.04	< 1

TABLE: Experimental runs with lattice dimension 64.

Lattice Solution

# of Unknown bits (τl_N)	# of Known bits (ηl_N)	Time in Seconds		
		LLL Algo	Resultant	Root
40	90	36.66	25.67	< 1
50	110	47.31	35.20	< 1
60	135	69.23	47.14	< 1
70	155	73.15	58.04	< 1

TABLE: Experimental runs with lattice dimension 64.

NOTE

- Advantage: Complements the original Algorithm nicely.
- Requires $\eta > 2\tau + 2k/l_N$ if bits are missing after Level k.
- Disadvantage: Requires more than double bits for both primes.

Reconstruction from MSBs

General Idea

Notation

- $p[i], q[i]$ - i-th bits of p, q ($p[0] = q[0] = 1$ are MSBs)
- p_i, q_i - partial solution for p, q through bits $0 - i$

Idea

- Suppose we get chunks of bits from p, q via Coldboot attack.
- The basic idea is that of a recursive mutual reconstruction.

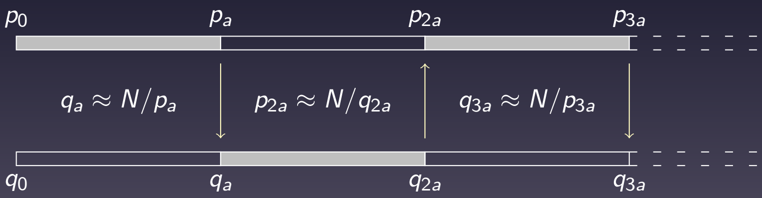
General Idea

Notation

- $p[i], q[i]$ - i-th bits of p, q ($p[0] = q[0] = 1$ are MSBs)
- p_i, q_i - partial solution for p, q through bits $0 - i$

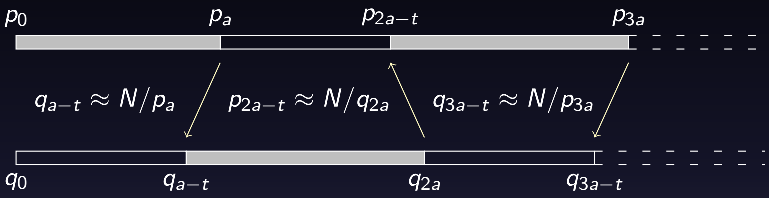
Idea

- Suppose we get chunks of bits from p, q via Coldboot attack.
- The basic idea is that of a recursive mutual reconstruction.



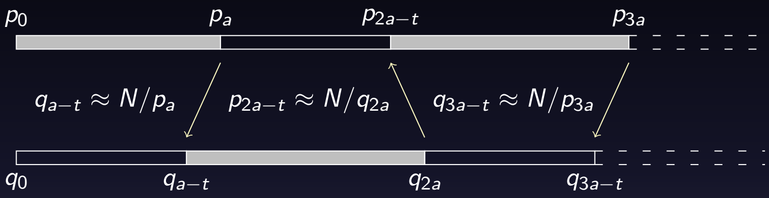
Detailed Approach

Practical Scenario



Detailed Approach

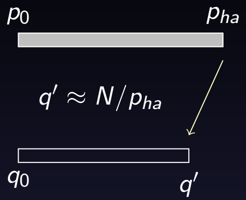
Practical Scenario



Issues to resolve

- How accurate are the approximations?
- How probable is the success of the reconstruction process?
- How many bits of the primes do we need to know?

Approximations

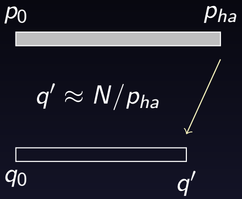


Assume $\sqrt{N/2} < q < \sqrt{N} < p < \sqrt{2N}$
We have $|p - p_{ha}| < 2^{l_p - ha}$, and thus

$$|q - q'| = \left| \frac{N}{p} - \frac{N}{p_{ha}} \right| = \frac{N}{pp_{ha}} |p - p_{ha}| < 2^{l_p - ha}$$

Suppose $q' = q + X$, where $X < 2^{l_p - ha}$ is of size $l_p - ha$ or less.

Approximations



Assume $\sqrt{N/2} < q < \sqrt{N} < p < \sqrt{2N}$
We have $|p - p_{ha}| < 2^{l_p - ha}$, and thus

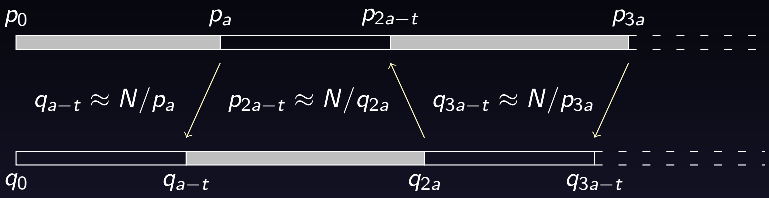
$$|q - q'| = \left| \frac{N}{p} - \frac{N}{p_{ha}} \right| = \frac{N}{pp_{ha}} |p - p_{ha}| < 2^{l_p - ha}$$

Suppose $q' = q + X$, where $X < 2^{l_p - ha}$ is of size $l_p - ha$ or less.

$Pr[q' = q_{ha-t}] =$ Probability that Carry propagates less than t bits

$$Pr[q' = q_{ha-t}] > 1 - \frac{1}{2^t}$$

Success Probability

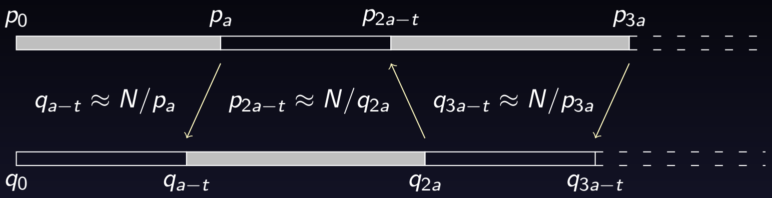


Total number of approximations: $\frac{target}{blocksize} = \left\lfloor \frac{l_p/2}{a} \right\rfloor = \left\lfloor \frac{l_N}{4a} \right\rfloor$

Probability of success = Probability that each approx. is correct

$$P_{a,t} > \left(1 - \frac{1}{2^t}\right)^{\lfloor l_N/4a \rfloor}$$

Bit Requirement



Bits needed at each approximation level $\approx a + t$

Total bit requirement is approximately

$$\left\lfloor \frac{l_N}{4a} \right\rfloor (a+t) = \left\lfloor \frac{l_N}{4} \left(1 + \frac{t}{a}\right) \right\rfloor$$

Experiments

a	$t = 1$	$t = 2$	$t = 3$	$t = 4$	$t = 5$
10	0, 0	2.5, 0.07	16.8, 3.55	41.5, 19.9	64.5, 45.2
20	1.8, 0.02	18.7, 3.17	44.5, 20.1	65.7, 46.1	81.9, 68.3
40	15.5, 1.6	42.8, 17.8	66.7, 44.9	81.8, 67.9	90.8, 82.7
60	29.1, 6.3	55.6, 31.6	75.7, 58.6	86.6, 77.2	91.7, 88.1
80	41.9, 12.5	66.4, 42.2	82.9, 67.0	91.0, 82.4	95.7, 90.9
100	50.6, 25.0	74.4, 56.2	86.6, 76.6	93.7, 87.9	97.1, 93.8

Each cell: Practical probability, Theoretical probability of success

Practical probability: 10000 experiments each with 1024 RSA

Highlights: Bit requirement $< 70\%$ with success probability $> \frac{1}{2}$

Runtime of algorithm $= O(\log^2 N)$

Experiments

a	$t = 6$	$t = 7$	$t = 8$	$t = 9$	$t = 10$
10	82.1, 67.5	90.6, 82.2	95.0, 90.7	97.2, 95.2	-
20	90.6, 82.8	94.8, 91.0	97.5, 95.4	98.5, 97.7	99.3, 97.6
40	95.2, 91.0	97.8, 95.4	98.6, 97.7	99.3, 98.8	99.9, 99.4
60	95.3, 93.9	97.4, 96.9	98.9, 98.4	99.5, 99.2	99.9, 99.7
80	98.3, 95.4	99.1, 97.7	99.4, 98.8	99.7, 99.4	100, 99.7
100	98.8, 96.9	99.6, 98.4	99.8, 99.2	99.9, 99.6	100, 99.8

Each cell: Practical probability, Theoretical probability of success

Practical probability: 10000 experiments each with 1024 RSA

Highlights: Bit requirement $< 70\%$ with success probability $> \frac{1}{2}$

Runtime of algorithm $= O(\log^2 N)$

Conclusion

Summary

PREMISE

- Coldboot Attack: Bits of RSA primes can be obtained
- Crypto 2009: RSA primes can be reconstructed from LSB side
- Crypto 2009: 59% of the bits suffice for reconstruction

Summary

PREMISE

- Coldboot Attack: Bits of RSA primes can be obtained
- Crypto 2009: RSA primes can be reconstructed from LSB side
- Crypto 2009: 59% of the bits suffice for reconstruction

THIS TALK

- LSB Reconstruction: Analysis of Crypto 2009 idea

Summary

PREMISE

- Coldboot Attack: Bits of RSA primes can be obtained
- Crypto 2009: RSA primes can be reconstructed from LSB side
- Crypto 2009: 59% of the bits suffice for reconstruction

THIS TALK

- LSB Reconstruction: Analysis of Crypto 2009 idea
- LSB Reconstruction: Just 50% bits from lower halves suffice

Summary

PREMISE

- Coldboot Attack: Bits of RSA primes can be obtained
- Crypto 2009: RSA primes can be reconstructed from LSB side
- Crypto 2009: 59% of the bits suffice for reconstruction

THIS TALK

- LSB Reconstruction: Analysis of Crypto 2009 idea
- LSB Reconstruction: Just 50% bits from lower halves suffice
- LSB Reconstruction: Works same or better for special cases

Summary

PREMISE

- Coldboot Attack: Bits of RSA primes can be obtained
- Crypto 2009: RSA primes can be reconstructed from LSB side
- Crypto 2009: 59% of the bits suffice for reconstruction

THIS TALK

- LSB Reconstruction: Analysis of Crypto 2009 idea
- LSB Reconstruction: Just 50% bits from lower halves suffice
- LSB Reconstruction: Works same or better for special cases
- LSB Reconstruction: Lattice solution to 'missing bits' issue

Summary

PREMISE

- Coldboot Attack: Bits of RSA primes can be obtained
- Crypto 2009: RSA primes can be reconstructed from LSB side
- Crypto 2009: 59% of the bits suffice for reconstruction

THIS TALK

- LSB Reconstruction: Analysis of Crypto 2009 idea
- LSB Reconstruction: Just 50% bits from lower halves suffice
- LSB Reconstruction: Works same or better for special cases
- LSB Reconstruction: Lattice solution to 'missing bits' issue
- MSB Reconstruction: A completely new idea for MSB side

Summary

PREMISE

- Coldboot Attack: Bits of RSA primes can be obtained
- Crypto 2009: RSA primes can be reconstructed from LSB side
- Crypto 2009: 59% of the bits suffice for reconstruction

THIS TALK

- LSB Reconstruction: Analysis of Crypto 2009 idea
- LSB Reconstruction: Just 50% bits from lower halves suffice
- LSB Reconstruction: Works same or better for special cases
- LSB Reconstruction: Lattice solution to 'missing bits' issue
- MSB Reconstruction: A completely new idea for MSB side
- MSB Reconstruction: Analysis and experimental verification

Current Goal

Open question mentioned in the paper:
"what if random bits, not blocks, are known at MSB side?"

One of the reviewers for this paper suggested:
"why don't you extend the LSB algorithm to the MSB case?"

Current Goal

Open question mentioned in the paper:
"what if random bits, not blocks, are known at MSB side?"

One of the reviewers for this paper suggested:
"why don't you extend the LSB algorithm to the MSB case?"

IDEA

- At any level $i-1$, we have 4 choices for next bits $p[i], q[i]$
- Each choice gives an upper and lower bound on p_i, q_i
- Multiply to get upper and lower bounds on $N^{(i)} = p_i q_i$
- Accept the option for $p[i], q[i]$ if this bound suits N
- Trim and prune the search tree as the known bits come in

Current Goal

Open question mentioned in the paper:

"what if random bits, not blocks, are known at MSB side?"

One of the reviewers for this paper suggested:

"why don't you extend the LSB algorithm to the MSB case?"

IDEA

- At any level $i - 1$, we have 4 choices for next bits $p[i], q[i]$
- Each choice gives an upper and lower bound on p_i, q_i
- Multiply to get upper and lower bounds on $N^{(i)} = p_i q_i$
- Accept the option for $p[i], q[i]$ if this bound suits N
- Trim and prune the search tree as the known bits come in

to be included in the extended journal version

Open Problems

1. Completion of the idea we just stated
    - Formalizing the reconstruction algorithm
    - Experimental verification of the idea
    - Statistical analysis of the branching and bit requirement

Open Problems

1. Completion of the idea we just stated
    - Formalizing the reconstruction algorithm
    - Experimental verification of the idea
    - Statistical analysis of the branching and bit requirement

2. Provide a theoretical value for the pruning parameter γ
    - Heninger and Shacham conjectured that $\gamma = 0.5$
    - Our experiments showed that $\gamma > 0.5$ in most cases
    - No theoretical proof could be provided till date

Open Problems

1. Completion of the idea we just stated
   - Formalizing the reconstruction algorithm
   - Experimental verification of the idea
   - Statistical analysis of the branching and bit requirement

2. Provide a theoretical value for the pruning parameter γ
   - Heninger and Shacham conjectured that $\gamma = 0.5$
   - Our experiments showed that $\gamma > 0.5$ in most cases
   - No theoretical proof could be provided till date

3. Can we do any better than what we saw?
   - Better the bit requirements and pruning in LSB case
   - Better the probability of success in the MSB case

Thank You