北京大学
PEKING UNIVERSITY

A Practical Cold Boot Attack on RSA Private Keys

Tian Wang[1], Xiaoxin Cui[1], Yewen Ni[1], Dunshan Yu[1], Xiaole Cui[2], Gang Qu[3]

[1] Institute of Microelectronics, Peking University, Beijing, China

[2]Key Lab of Integrated Microsystems, Peking University Shenzhen Graduate School, Shenzhen, China

[3]ECE Department, University of Maryland, College Park, MD, USA

Email: cuixx@pku.edu.cn ; cuixl@pkusz.edu.cn ; gangqu@umd.edu

AsianHOST 2017

Outline

- Cold boot attacks on RSA encryption schemes

-Cold boot attack and memory decay

-Existing algorithms to recover RSA private keys

- Practical cold boot attacks on RSA private keys

-Limitations of the existing algorithms

-Our improvements

- Evaluation results and discussion
- Conclusion

北京大学
PEKING UNIVERSITY

Outline

- *Cold boot attacks on RSA encryption schemes*

-Cold boot attack and memory decay

-Existing algorithms to recover RSA private keys

- Practical cold boot attacks on RSA private keys

-Limitations of the existing algorithms

-Our improvements

- Evaluation results and discussion
- Conclusion

Cold boot attack

- J. Halderman et al, USENIX Security Symposium, August 2008
- Memory remanence effect -Data persists after power-off.
- Memory blocks are transferred to other machines to extract the encryption keys .

Fig.1 A typical cold boot attack[1]

[1] J.A. Halderman et al, “Lest we remember: cold boot attacks on encryption keys,” in *USENIX Security Symposium*, 2008, pp. 45–60

Memory decay

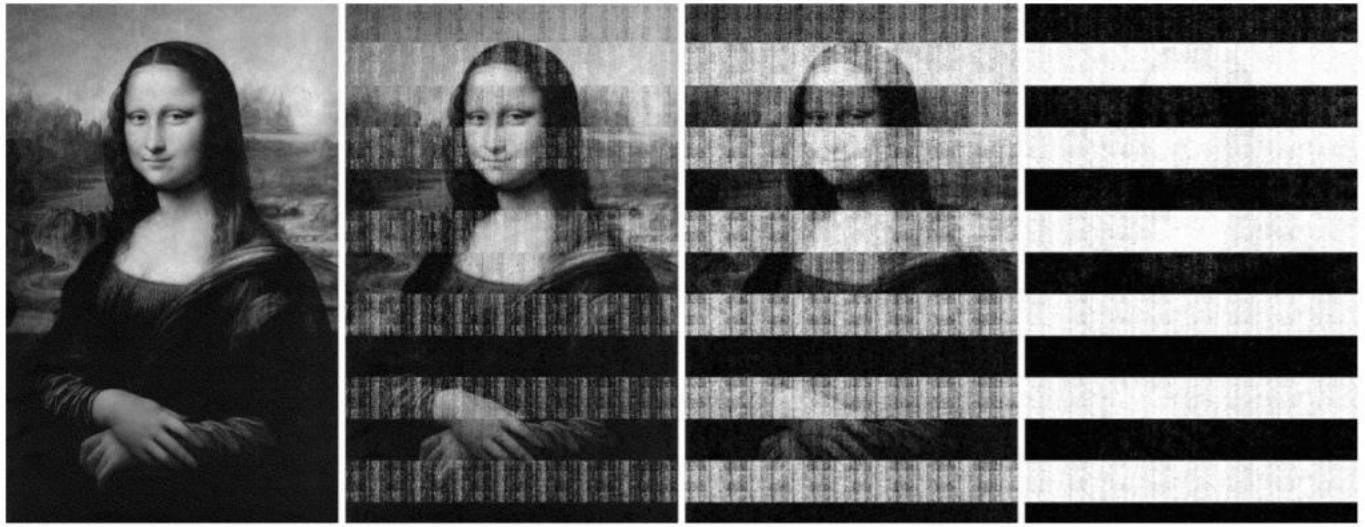


Fig.2 Visualizing memory decay[1]

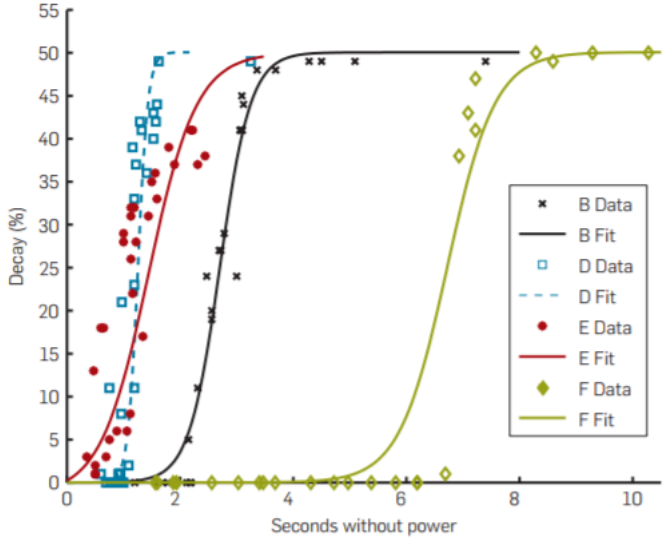


Fig.3 Measuring decay for different test systems[1]

Bit flip!

[1] J.A. Halderman et al, “Lest we remember: cold boot attacks on encryption keys,” in *USENIX Security Symposium*, 2008, pp. 45–60

Memory decay

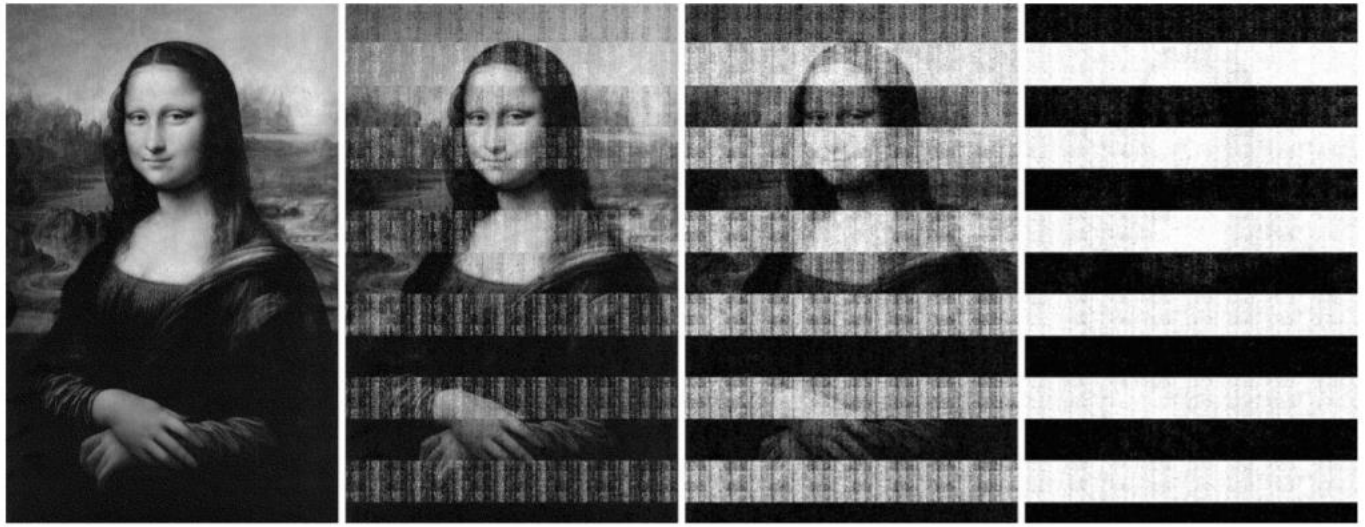


Fig.2 Visualizing memory decay[1]

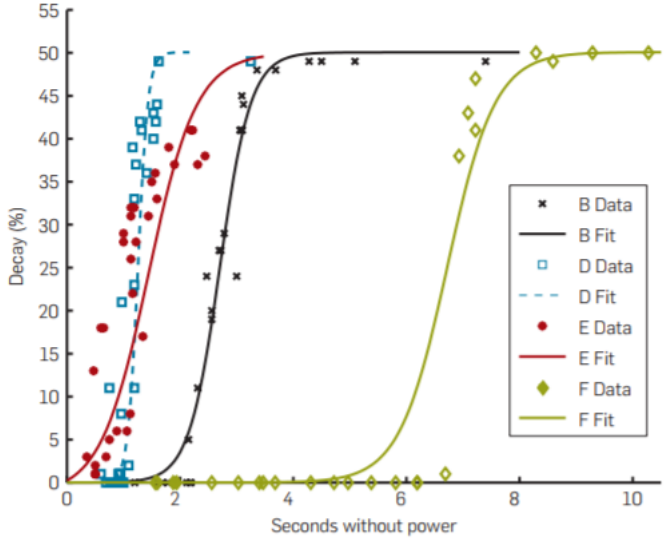
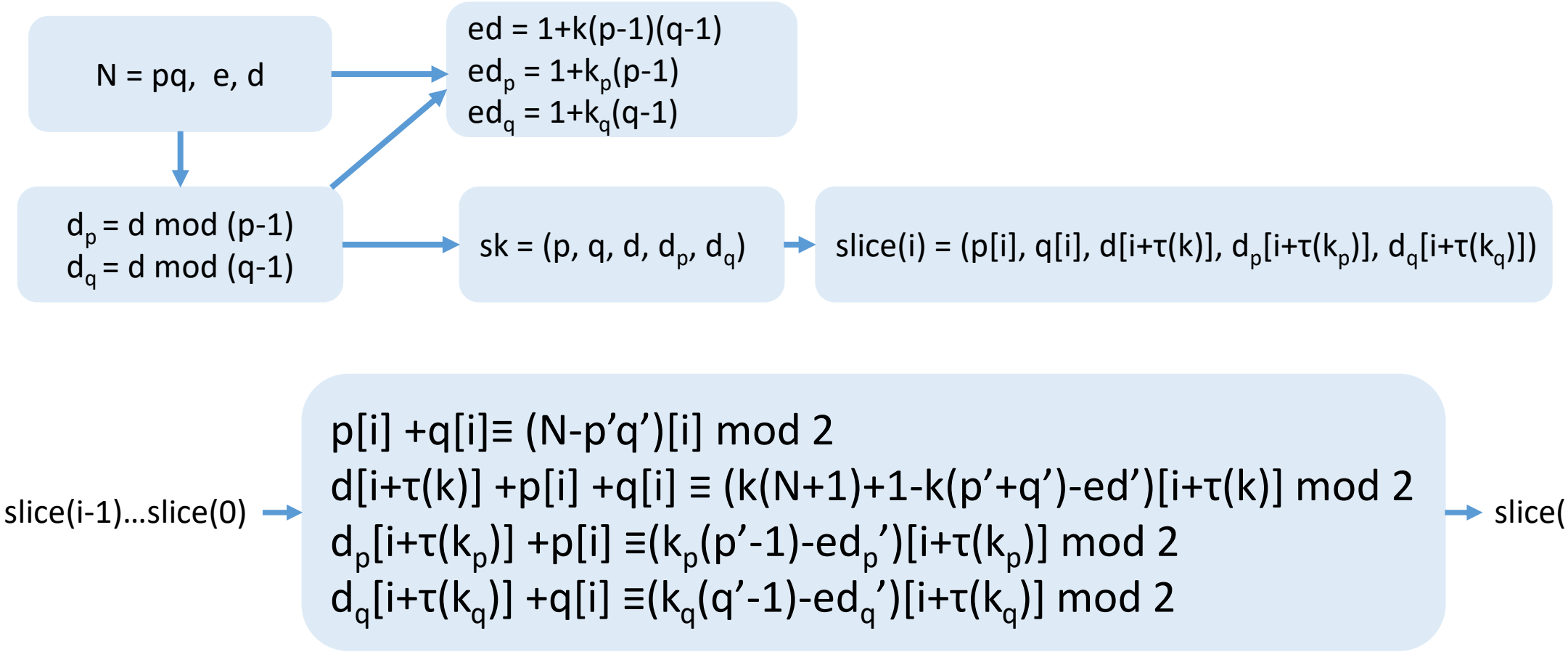


Fig.3 Measuring decay for different test systems[1]

- Cooling
- Error correction method

[1] J.A. Halderman et al, “Lest we remember: cold boot attacks on encryption keys,” in *USENIX Security Symposium*, 2008, pp. 45–60

Existing algorithms to recover RSA private keys(1/4)



Existing algorithms to recover RSA private keys(2/4)

- Hensel lifting

$p[i] + q[i] \equiv c_1$

$d[i+\tau(k)] + p[i] + q[i] \equiv c_2$

$d_p[i+\tau(k_p)] + p[i] \equiv c_3$

$d_q[i+\tau(k_q)] + q[i] \equiv c_4$

Existing algorithms to recover RSA private keys(2/4)

- Hensel lifting

$p[i] + q[i] \equiv c_1$

$d[i+\tau(k)] + p[i] + q[i] \equiv c_2$

$d_p[i+\tau(k_p)] + p[i] \equiv c_3$

$d_q[i+\tau(k_q)] + q[i] \equiv c_4$

Slice[0]

$p[0] = 1$
$q[0] = 1$
$d[\tau(k)]\ldots d[0]$
$d_p[\tau(k_p)]\ldots d_p[0]$
$d_q[\tau(k_q)]\ldots d_q[0]$

Existing algorithms to recover RSA private keys(2/4)

- Hensel lifting

$p[i] + q[i] \equiv c_1$

$d[i+\tau(k)] + p[i] + q[i] \equiv c_2$

$d_p[i+\tau(k_p)] + p[i] \equiv c_3$

$d_q[i+\tau(k_q)] + q[i] \equiv c_4$

Slice[0]

$p[0] = 1$
$q[0] = 1$
$d[\tau(k)]\ldots d[0]$
$d_p[\tau(k_p)]\ldots d_p[0]$
$d_q[\tau(k_q)]\ldots d_q[0]$

Slice[1]

$p[1] = 1$
$q[1] = 1-c_1$
$d[\tau(k)+1]=c_2-c_1$
$d_p[\tau(k_p)+1]=c_3-1$
$d_q[\tau(k_q)+1]=c_4-q[1]$

Existing algorithms to recover RSA private keys(2/4)

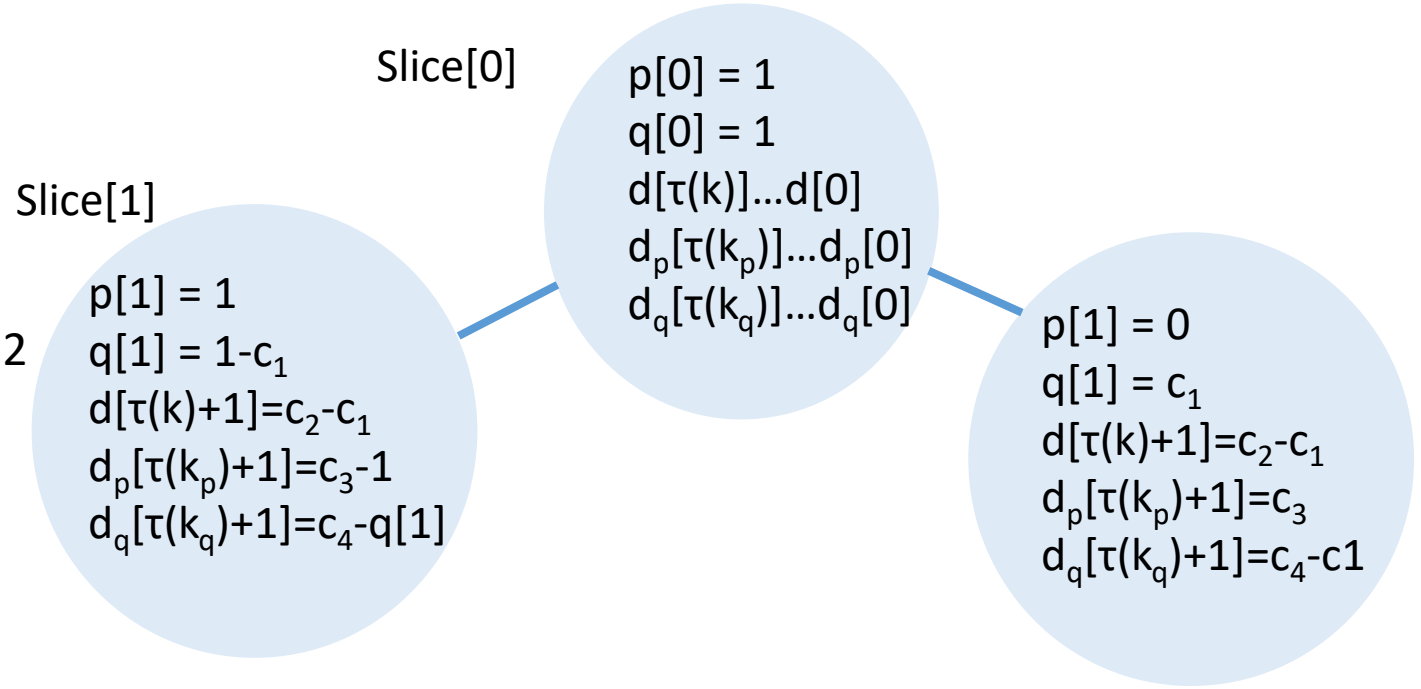
- Hensel lifting

$p[i] + q[i] \equiv c_1$

$d[i+\tau(k)] + p[i] + q[i] \equiv c_2$

$d_p[i+\tau(k_p)] + p[i] \equiv c_3$

$d_q[i+\tau(k_q)] + q[i] \equiv c_4$



Existing algorithms to recover RSA private keys(2/4)

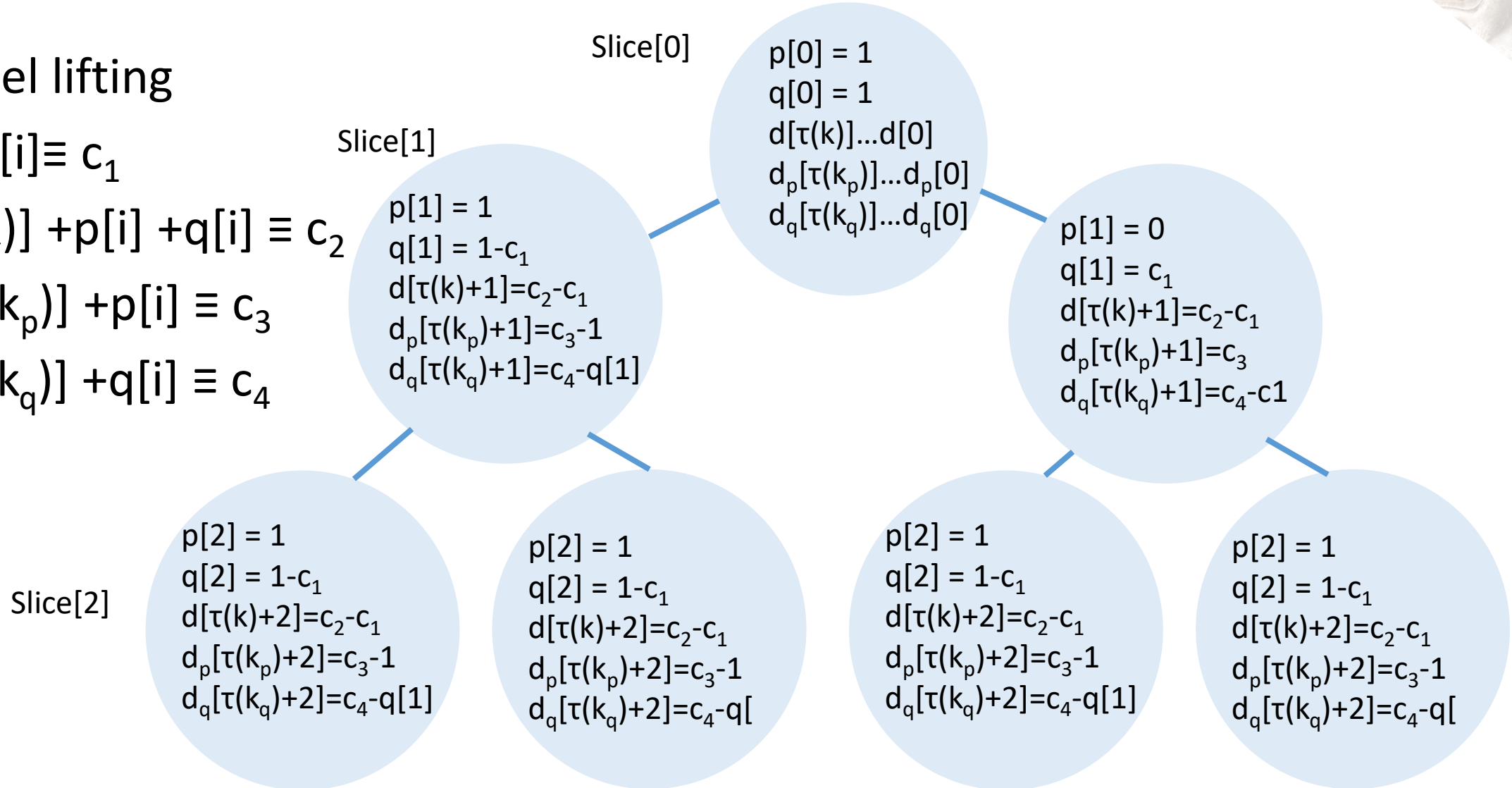
- Hensel lifting

$p[i] + q[i] \equiv c_1$

$d[i+\tau(k)] + p[i] + q[i] \equiv c_2$

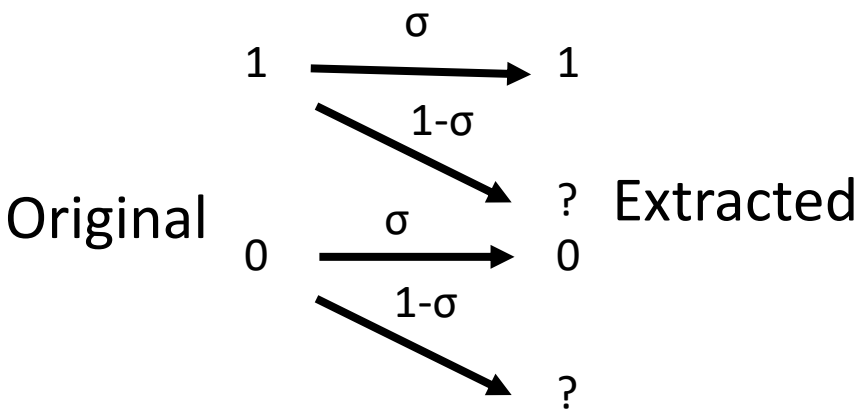
$d_p[i+\tau(k_p)] + p[i] \equiv c_3$

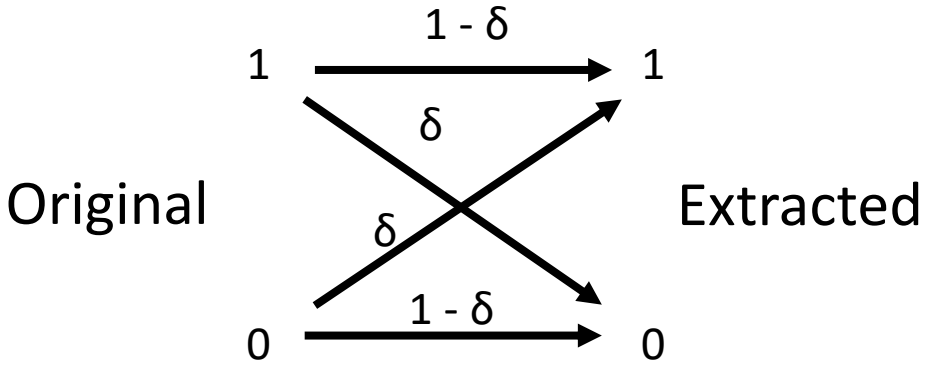
$d_q[i+\tau(k_q)] + q[i] \equiv c_4$



Existing algorithms to recover RSA private keys(3/4)

HS algorithm[2]	HMM algorithm[3]
A fraction σ(σ>0.27) of the extracted bits are known.	There exist a flipping probability δ(δ<0.237) for each bit.





[2] N. Heninger and H. Shacham, "Reconstructing RSA private keys from random key bits," *Advances in Cryptology-CRYPTO 2009.* Springer Berlin Heidelberg, 2009, pp.1-17
[3] W. Henecka, A. May and A. Meurer, "Correcting errors in RSA private keys," *Advances in Cryptology–CRYPTO 2010.* Springer Berlin Heidelberg, 2010, pp.351-369.

Existing algorithms to recover RSA private keys(4/4)

HS algorithm

HMM algorithm

Existing algorithms to recover RSA private keys(4/4)

HS algorithm

HMM algorithm

Expansion

Existing algorithms to recover RSA private keys(4/4)

	HS algorithm	HMM algorithm
Expansion	Build candidates with Hensel lifting **1** bit a time	

Existing algorithms to recover RSA private keys(4/4)

	HS algorithm	HMM algorithm
Expansion	Build candidates with Hensel lifting **1** bit a time	Build candidates with Hensel lifting **t** bits a time

Existing algorithms to recover RSA private keys(4/4)

	HS algorithm	HMM algorithm
Expansion	Build candidates with Hensel lifting **1** bit a time	Build candidates with Hensel lifting **t** bits a time
Pruning		

Existing algorithms to recover RSA private keys(4/4)

	HS algorithm	HMM algorithm
Expansion	Build candidates with Hensel lifting **1** bit a time	Build candidates with Hensel lifting **t** bits a time
Pruning	Remove the candidates that don't match with known key bits	

Existing algorithms to recover RSA private keys(4/4)

HS algorithm

Expansion: Build candidates with Hensel lifting **1** bit a time

Pruning: Remove the candidates that don't match with known key bits

HMM algorithm

Build candidates with Hensel lifting **t** bits a time

Calculate the number of matching bits

Existing algorithms to recover RSA private keys(4/4)

HS algorithm

Expansion

Build candidates with Hensel lifting **1** bit a time

Pruning

Remove the candidates that don't match with known key bits

HMM algorithm

Build candidates with Hensel lifting **t** bits a time

Calculate the number of matching bits

Calculate threshold **C**

Existing algorithms to recover RSA private keys(4/4)

HS algorithm

Expansion

Build candidates with Hensel lifting **1** bit a time

Pruning

Remove the candidates that don't match with known key bits

HMM algorithm

Build candidates with Hensel lifting **t** bits a time

Calculate the number of matching bits

Calculate threshold **C**

Remove the candidates whose number of matching bits is less than the threshold **C**

Existing algorithms to recover RSA private keys(4/4)

HS algorithm

Expansion

Build candidates with Hensel lifting **1** bit a time

Pruning

Remove the candidates that don't match with known key bits

HMM algorithm

Build candidates with Hensel lifting **t** bits a time

Calculate the number of matching bits

Calculate threshold **C**

Remove the candidates whose number of matching bits is less than the threshold **C**

Existing algorithms to recover RSA private keys(4/4)

HS algorithm

Expansion

Build candidates with Hensel lifting **1** bit a time

Pruning

Remove the candidates that don't match with known key bits

HMM algorithm

If $\delta \le 0.5 - \gamma - \varepsilon$
$t = [\ln 2n / 10\varepsilon^2]$
$\gamma = (\ln 2 (t+1)/10t)^{0.5}$
$C = 5t(0.5+\gamma)$

Build candidates with Hensel lifting **t** bits a time

Calculate the number of matching bits

Calculate threshold **C**

Remove the candidates whose number of matching bits is less than the threshold **C**

Outline

- Cold boot attacks on RSA encryption schemes

-Cold boot attack and memory decay

-Existing algorithms to recover RSA private keys

- *Practical cold boot attacks on RSA private keys*

-Limitations of the existing algorithms

-Our improvements

- Evaluation results and discussion
- Conclusion

Limitations of the previous algorithms

Limitations of the previous algorithms

- The decay of memory bits from 0 to 1 and from 1 to 0 both exist.

Limitations of the previous algorithms

- The decay of memory bits from 0 to 1 and from 1 to 0 both exist.

No bits can be known with absolute certainty.

Limitations of the previous algorithms

- The decay of memory bits from 0 to 1 and from 1 to 0 both exist.

No bits can be known with absolute certainty.

HS Algorithm

北京大学
PEKING UNIVERSITY

Limitations of the previous algorithms

- The decay of memory bits from 0 to 1 and from 1 to 0 both exist.

No bits can be known with absolute certainty. HS Algorithm

- For a given region, the decay is dominated by one direction.

Limitations of the previous algorithms

- The decay of memory bits from 0 to 1 and from 1 to 0 both exist.

No bits can be known with absolute certainty. HS Algorithm

- For a given region, the decay is dominated by one direction.

The transition probability in one direction is larger than that of the other direction.

Symmetric decay model is not accurate.

Limitations of the previous algorithms

- The decay of memory bits from 0 to 1 and from 1 to 0 both exist.

No bits can be known with absolute certainty. HS Algorithm

- For a given region, the decay is dominated by one direction.

The transition probability in one direction is larger than that of the other direction.

Symmetric decay model is not accurate. HMM Algorithm

Limitations of the previous algorithms

- The decay of memory bits from 0 to 1 and from 1 to 0 both exist.

No bits can be known with absolute certainty. HS Algorithm

- For a given region, the decay is dominated by one direction.

The transition probability in one direction is larger than that of the other direction.

Symmetric decay model is not accurate. HMM Algorithm

- The data retention phase is not long enough to mount the attack.

Limitations of the previous algorithms

- The decay of memory bits from 0 to 1 and from 1 to 0 both exist.

No bits can be known with absolute certainty. HS Algorithm ✗

- For a given region, the decay is dominated by one direction.

The transition probability in one direction is larger than that of the other direction.

Symmetric decay model is not accurate. HMM Algorithm ✗

- The data retention phase is not long enough to mount the attack.

The flipping probability of the dominating direction can be larger than 0.237.

Limitations of the previous algorithms

- The decay of memory bits from 0 to 1 and from 1 to 0 both exist.

No bits can be known with absolute certainty. HS Algorithm

- For a given region, the decay is dominated by one direction.

The transition probability in one direction is larger than that of the other direction.

Symmetric decay model is not accurate. HMM Algorithm

- The data retention phase is not long enough to mount the attack.

The flipping probability of the dominating direction can be larger than 0.237.

HMM Algorithm

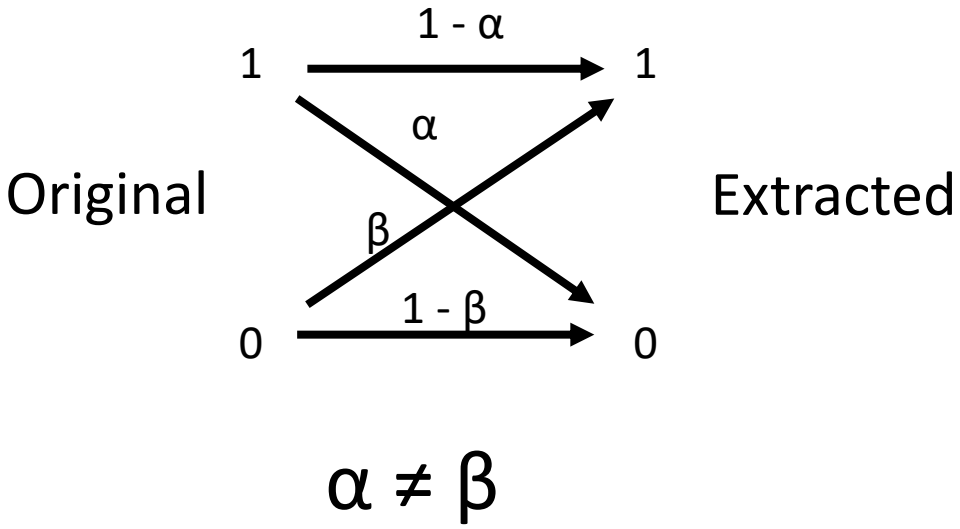
Our improvements based on HMM algorithm(1/4)

Our improvements based on HMM algorithm(1/4)

- Real-life settings.

Our improvements based on HMM algorithm(1/4)

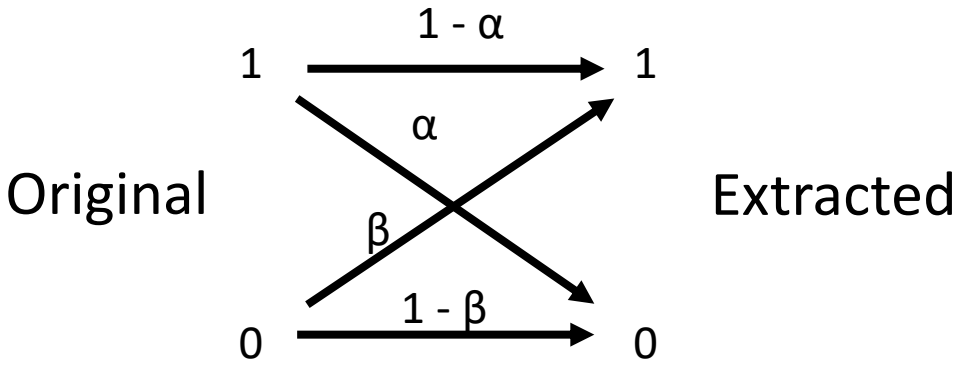
- Real-life settings.



$$\alpha \neq \beta$$

Our improvements based on HMM algorithm(1/4)

- Real-life settings.
- Asymmetric pruning strategy.

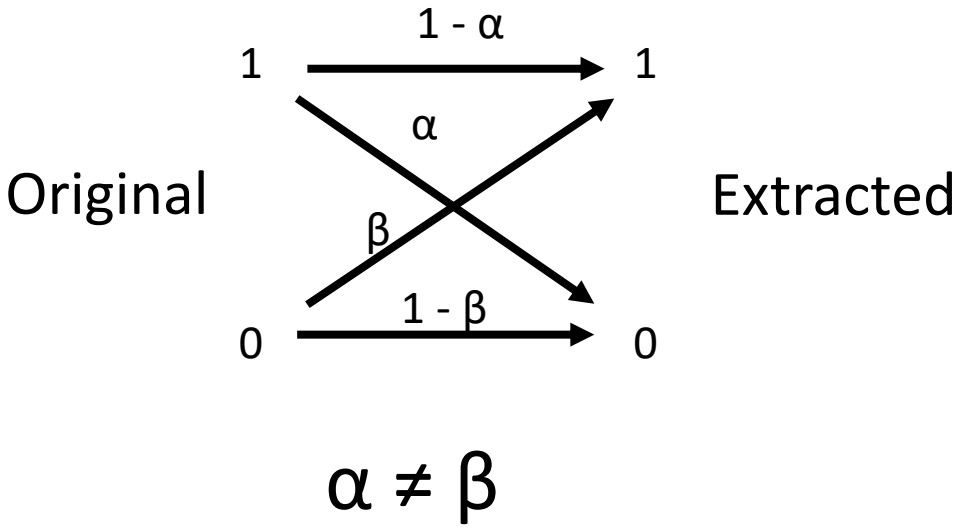


$$\alpha \neq \beta$$

Our improvements based on HMM algorithm(1/4)

- Real-life settings.
- Asymmetric pruning strategy.

Our improvements



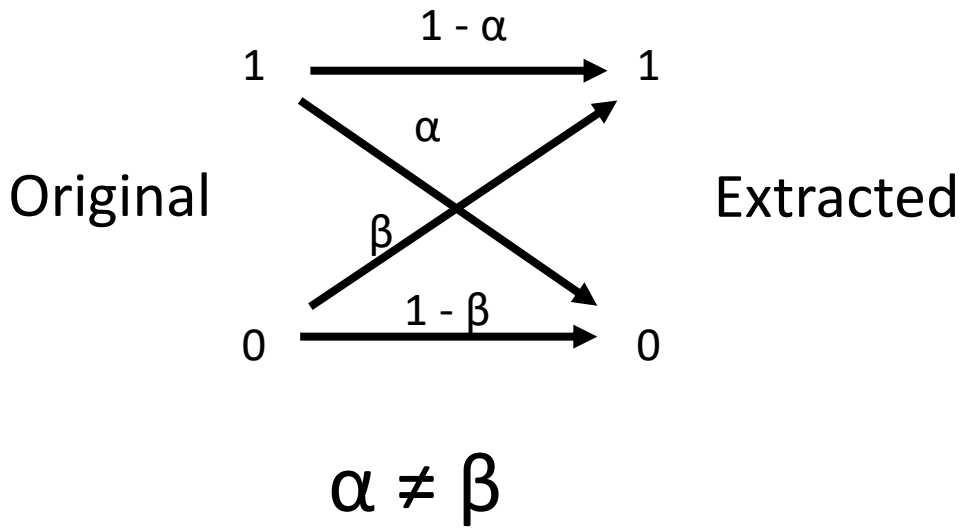
$$\alpha \neq \beta$$

Our improvements based on HMM algorithm(1/4)

- Real-life settings.
- Asymmetric pruning strategy.

Our improvements

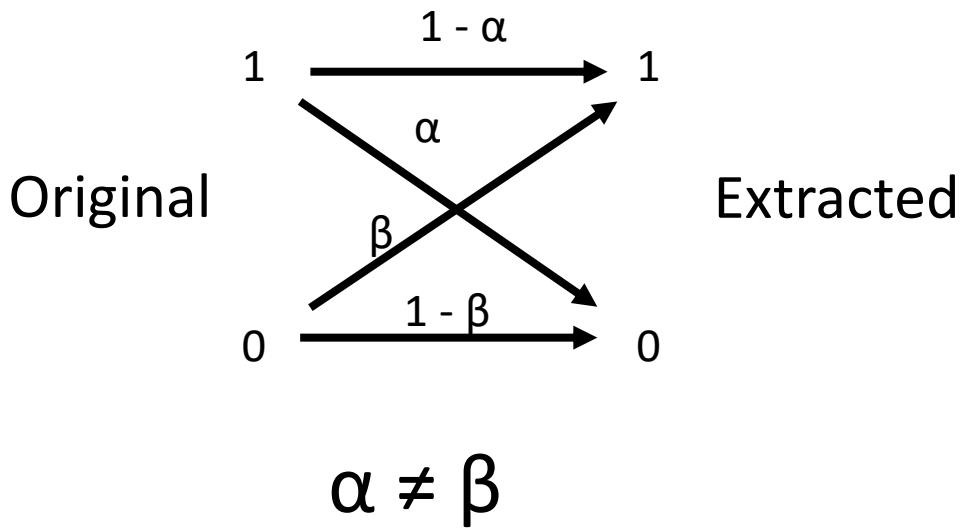
Expansion



$$\alpha \neq \beta$$

Our improvements based on HMM algorithm(1/4)

- Real-life settings.
- Asymmetric pruning strategy.



$$\alpha \neq \beta$$

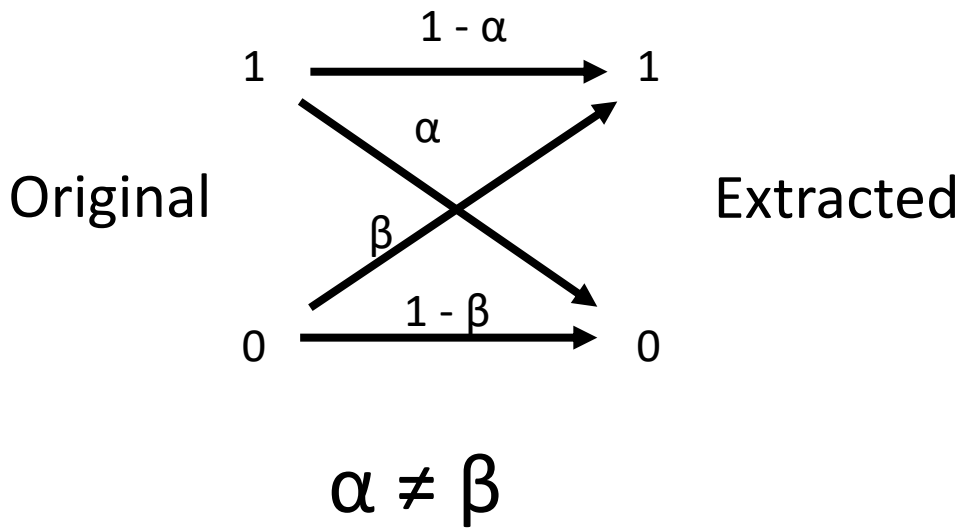
Our improvements

Build candidates with Hensel lifting **t** bits a time

Expansion

Our improvements based on HMM algorithm(1/4)

- Real-life settings.
- Asymmetric pruning strategy.



$\alpha \neq \beta$

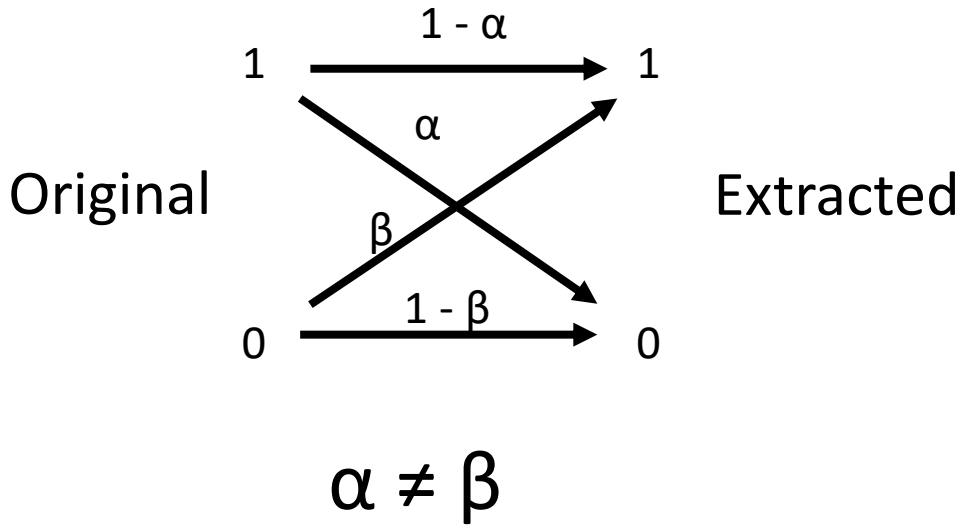
Our improvements

Build candidates with Hensel lifting **t** bits a time — Expansion

Pruning

Our improvements based on HMM algorithm(1/4)

- Real-life settings.
- Asymmetric pruning strategy.



$\alpha \neq \beta$

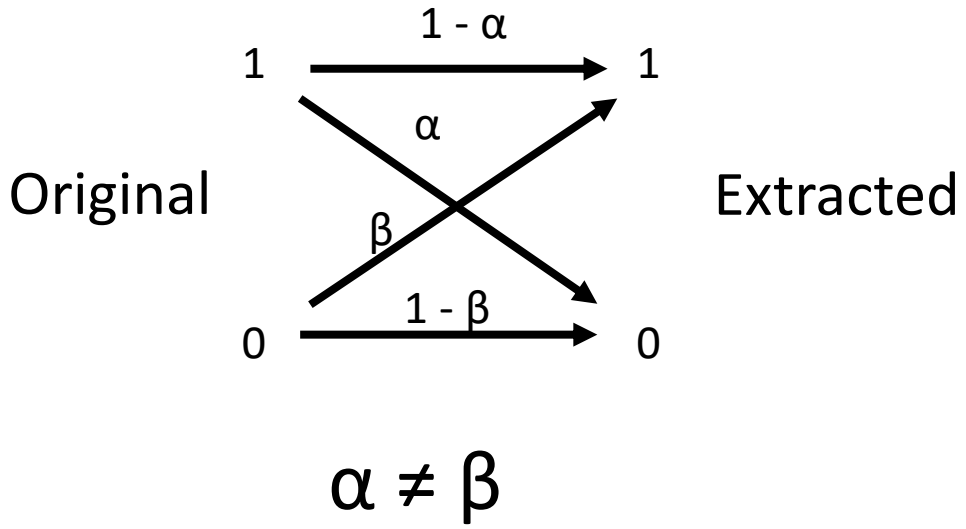
Our improvements

Build candidates with Hensel lifting **t** bits a time — Expansion

Calculate the number of matching bit 0 and matching bit 1 respectively — Pruning

Our improvements based on HMM algorithm(1/4)

- Real-life settings.
- Asymmetric pruning strategy.



$\alpha \neq \beta$

Our improvements

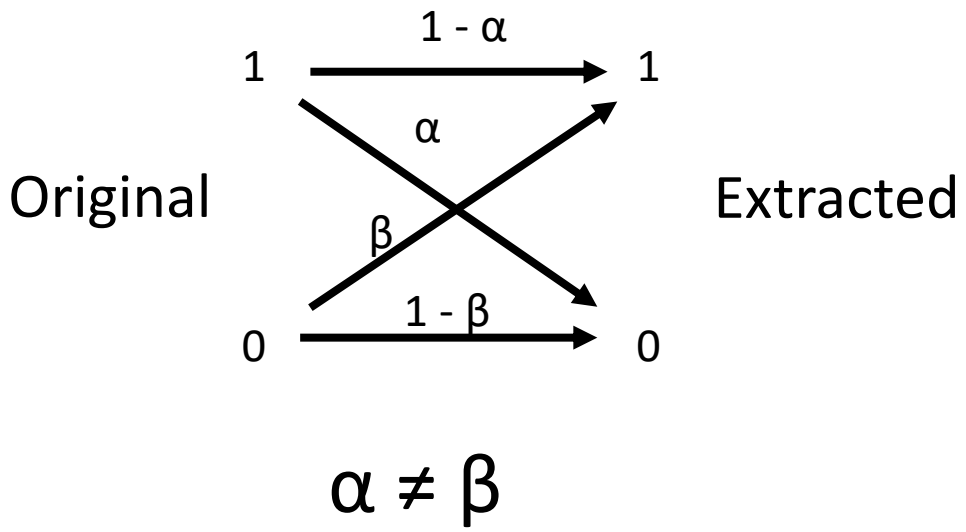
Build candidates with Hensel lifting **t** bits a time — Expansion

Calculate the number of matching bit 0 and matching bit 1 respectively — Pruning

Calculate two thresholds for bit 0 and bit 1 $\mathbf{C_0}$ and $\mathbf{C_1}$

Our improvements based on HMM algorithm(1/4)

- Real-life settings.
- Asymmetric pruning strategy.



$\alpha \neq \beta$

Our improvements

Build candidates with Hensel lifting **t** bits a time — Expansion

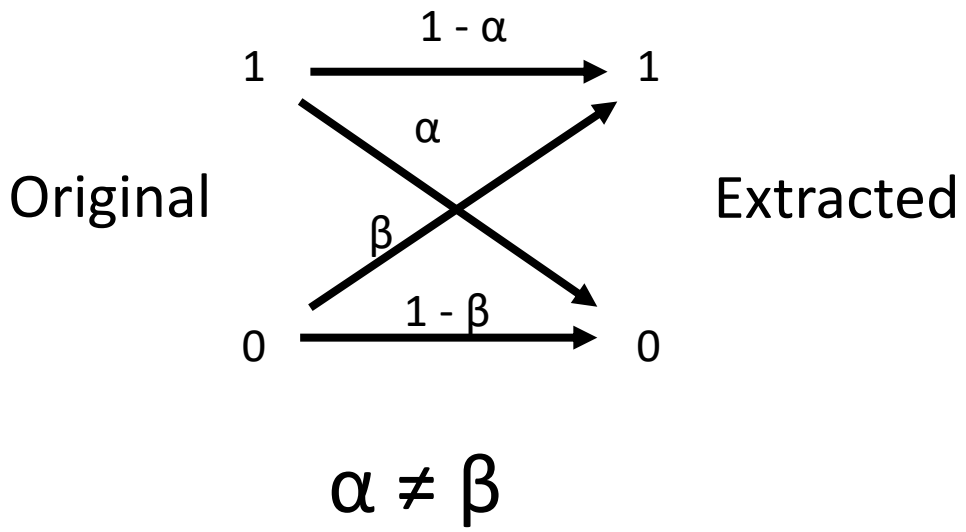
Calculate the number of matching bit 0 and matching bit 1 respectively — Pruning

Calculate two thresholds for bit 0 and bit 1 $\mathbf{C_0}$ and $\mathbf{C_1}$

Remove the candidates whose number of matching bit 0 (or 1) is less than the corresponding threshold $\mathbf{C_0(C_1)}$

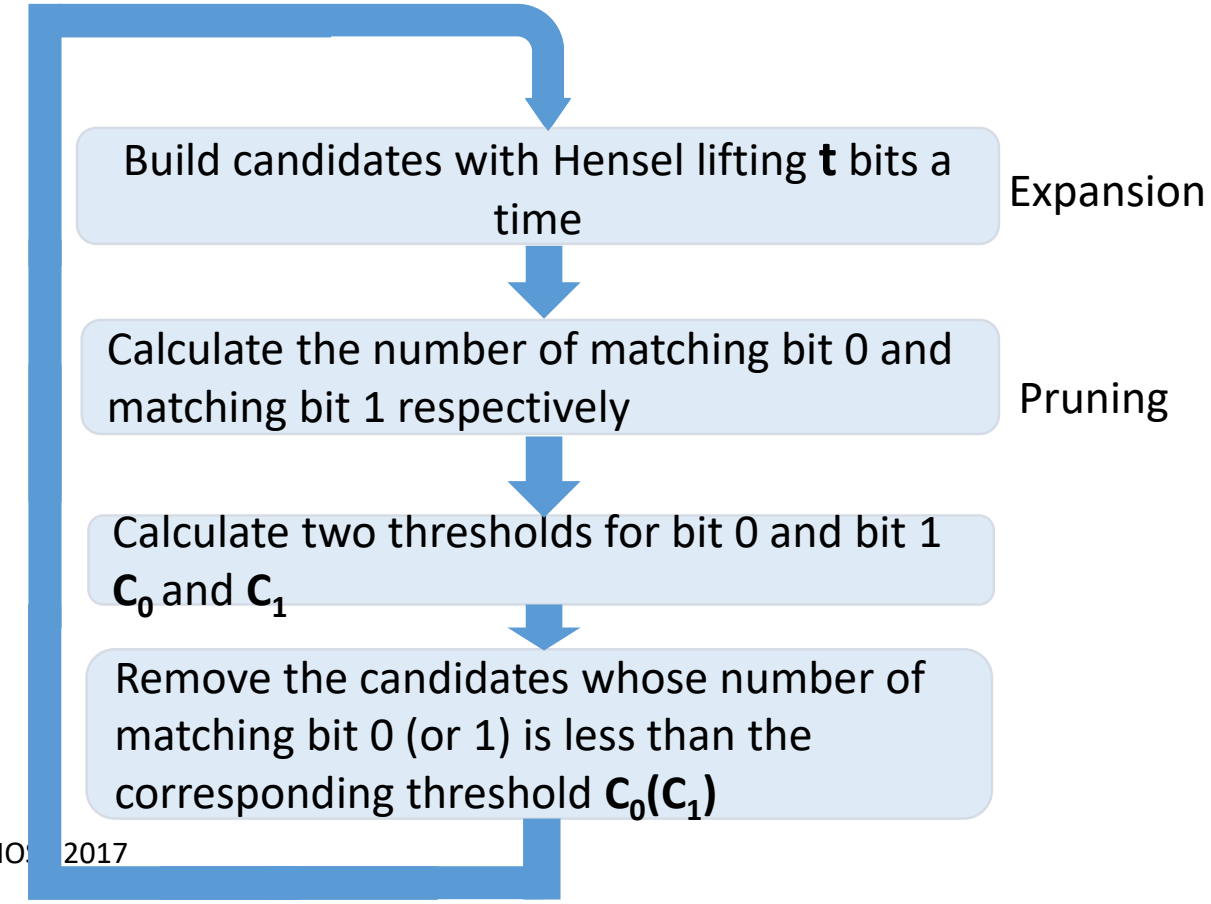
Our improvements based on HMM algorithm(1/4)

- Real-life settings.
- Asymmetric pruning strategy.



$\alpha \neq \beta$

Our improvements



Our improvements based on HMM algorithm(2/4)

The Improved HMM Algorithm
Input: (N,e), erroneous sk, error probabilities α and β
Output: the correct sk.
Initialization: Calculate (k, k_p, k_q) and slice(0)
For round = 1 to n/(2t)+1 Expansion: Expand t times for every candidate with slice(0)…slice(round-1)t using the Hensel lifting, which generates 2^t different partial candidates gc_1, $gc_{2\ldots\ldots}$ gc_2t in slice((round-1)t+1)… slice(round t).

Pruning:

For i = 1 to 2^t
--Count the number of matching bit 0(1) $X_{i0(1)}$ of gc_i in slice((round-1)t+1)… slice(round t) with the given erroneous sk.
--Compute the threshold $C_{i0(1)}$ in the ith round.
If (X_{i0} <= C_{i0} or X_{i1} <= C_{i1}) Discard the candidate.
Else Keep the candidate
end for
end for

Finalization: Run trial decryption to determine the correct private key.

Our improvements based on HMM algorithm(3/4)

- Notation

For any ε_0 ε_1>0, (N,e) as the n-bit modulus and public key.

Given flipping probability α, β, η =min((1- α) ε_1^2, ε_0^2)

$p_{1(0)}$: the error probability of bit 1(0) in the extracted key

$N_{0(1)}$: the number of bit 0 (1) in each *t* slices of the extracted key

$N_{0(1)}$': the number of bit 0 (1) in each *t* slices of the candidate key

- Choice of parameters

We choose t, t_0, t_1 so that t = [ln 2n /5η] $t = 1+t_0+t_1$

Our improvements based on HMM algorithm(4/4)

- Performance analysis

$\gamma_1 = (\ln 2\ (t_1+1)/2N_1)^{\wedge}0.5$ $\gamma_0 = (\ln 2\ (t_0+1)/2N_0')^{\wedge}0.5$

$C_1=N_1(0.5+\gamma_1)$ $C_0=N_0'(N_0/5t+\gamma_0)$

If $p_1 \leq 0.5-\gamma_1-\varepsilon_1$, $\beta \leq 1-N_0'-\gamma_0-\varepsilon_0$

The algorithm can output the correct sk in polynomial time **$O(n^{2+\ln 2/2.5\eta})$**

with success probability greater than $1-5\eta/2\ln(2n)-1/n$.

*success probability: The possibility that correct candidate isn't pruned.

Outline

- Cold boot attacks on RSA encryption schemes

-Cold boot attack and memory decay

-Existing algorithms to recover RSA private keys

- Practical cold boot attacks on RSA private keys

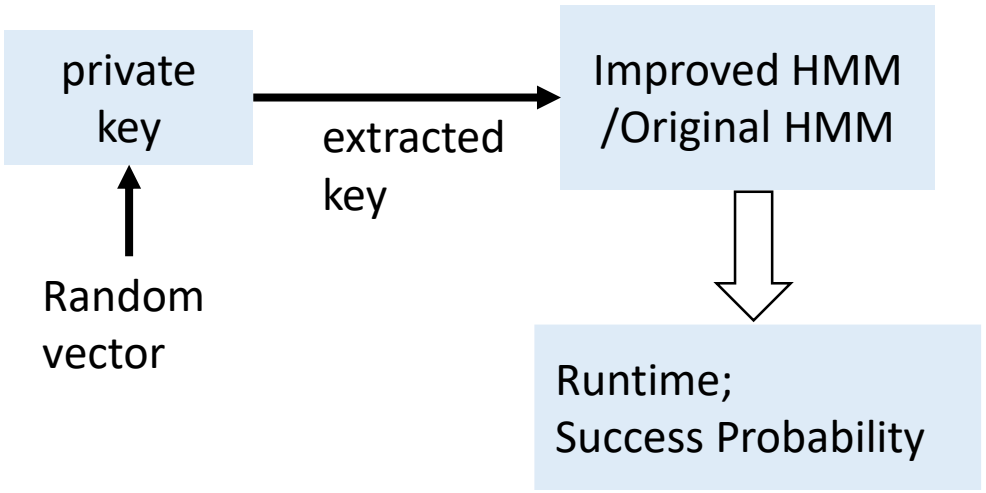
-Limitations of the existing algorithms

-Our improvements

- *Evaluation results and discussion*
- Conclusion

Evaluation results(1/3)

- $e = 2^{16}+1$
- $0.05 \le \alpha \le 0.4$ $\beta = 0.001, 0.01$
- 100 different 1024-bit private keys
- 100 repetitions for each private key



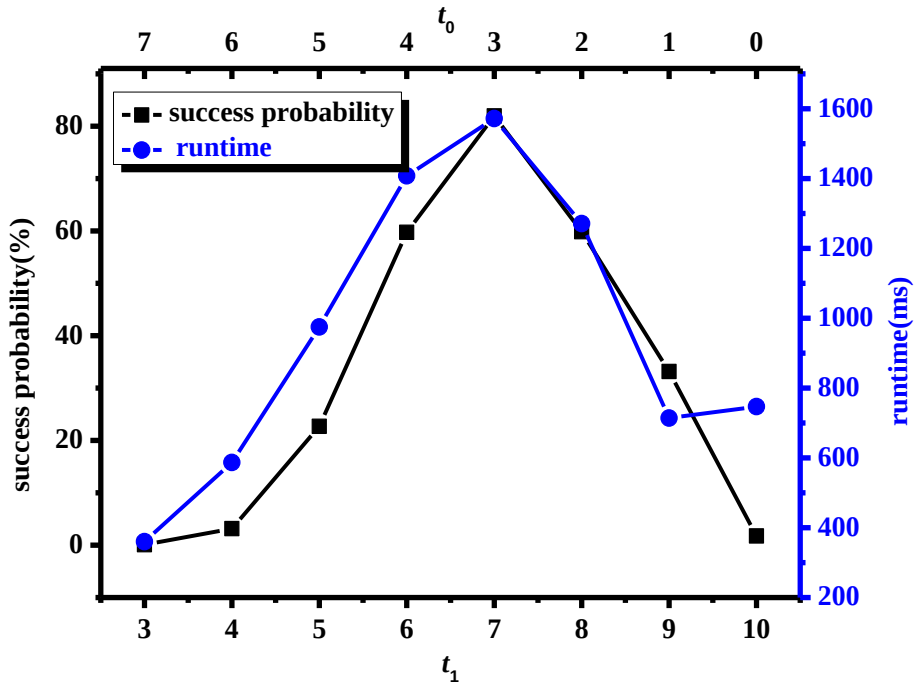


Fig. 4 The success probability and runtime for different combinations of (t_0,t_1) when $\alpha = 0.25$, $\beta = 0.001$ and $t = 11$.

Evaluation results(2/3)

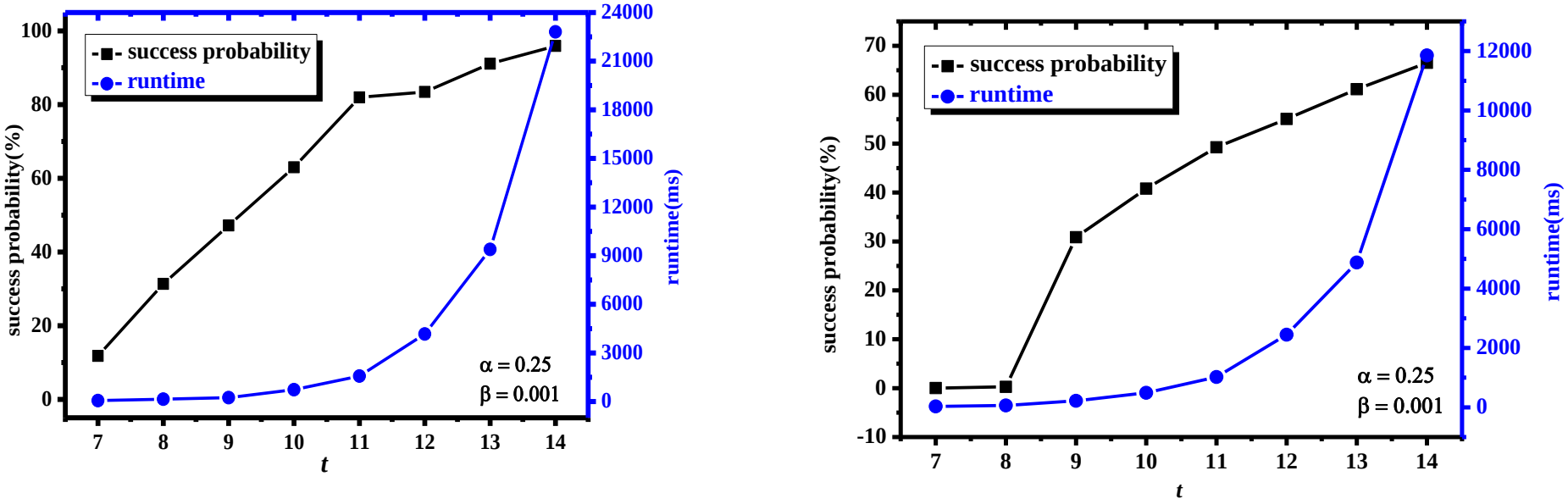


Fig. 5 Success probability and runtime of the improved HMM algorithm(a) and the original HMM algorithm(b)

Optimized t!

Evaluation results(3/3)

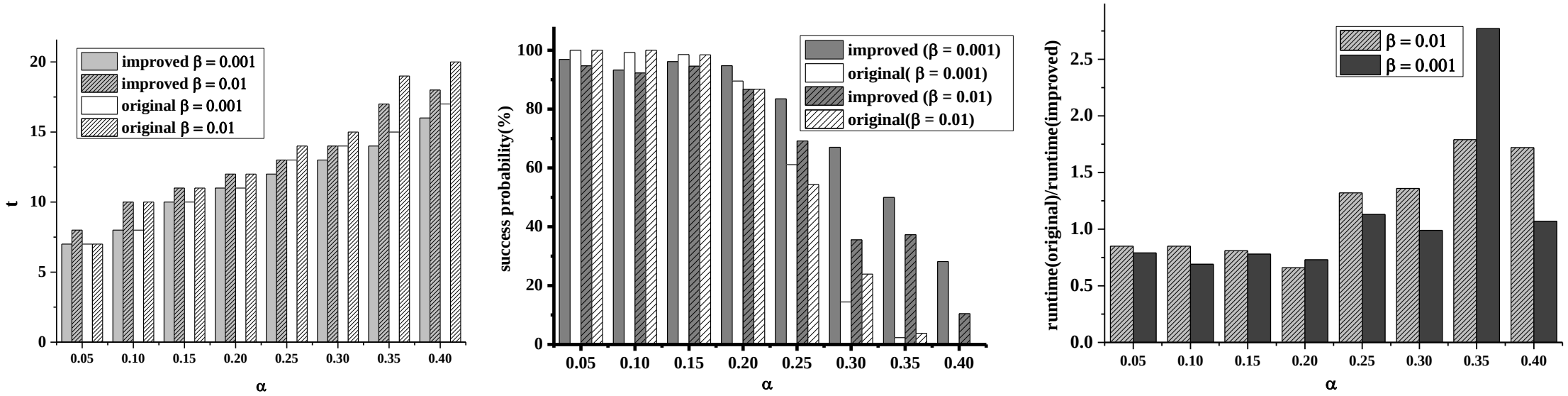


Fig. 5 Success probability and runtime of the improved HMM algorithm and the original HMM algorithm

- $\alpha\uparrow$ or $\beta\uparrow$ -> $t\uparrow$ -> success probability $\downarrow$
- $\alpha < 0.2$
- $\alpha > 0.2$

Discussion

- $C = C_1' + C_0' = N_0'(0.5+\gamma) + (5t - N_0')(0.5+\gamma)$
- $C_1 = N_1(0.5+\gamma_1)$ $\quad C_0 = N_0'(N_0/5t+\gamma_0)$
- $\alpha < 0.2 \quad C_1' \approx C_1 \quad C_0' < C_0$

$$P(X_{c0} + X_{c1} > C_0' + C_1') > P(X_{c0} + X_{c1} > C_0 + C_1)$$

$$> P(X_{c0} > C_0 \cap X_{c1} > C_1)$$

- $\alpha > 0.2 \quad C_1' > C_1$

$$P(X_{c0} + X_{c1} > C_0' + C_1') < P(X_{c0} > C_0 \cap X_{c1} > C_1)$$

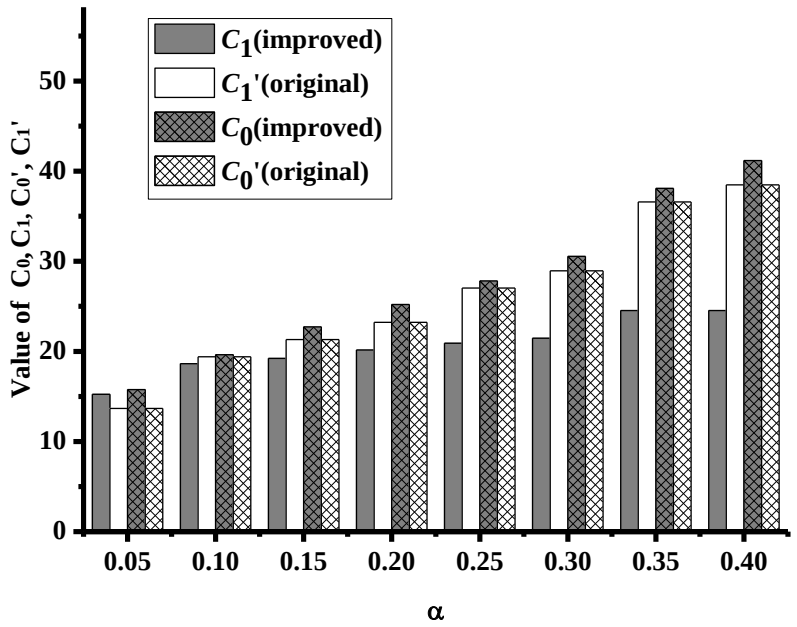


Fig. 6 Estimated threshold paramters of the improved and original HMM algorithm for different error probabilities α when β = 0.001

Outline

- Cold boot attacks on RSA encryption schemes

-Cold boot attack and memory decay

-Existing algorithms to recover RSA private keys

- Practical cold boot attacks on RSA private keys

-Limitations of the existing algorithms

-Our improvements

- Evaluation results and discussion
- *Conclusion*

Conclusion

- A practical improvement of the cold boot attack based on HMM algorithm

  -Real life settings

  -Asymmetric pruning strategy

- Performance analysis

Thank you!