Parallelized Common Factor Attack on RSA

Vineet Kumar[1] Aneek Roy[1] Sourya Sengupta[1] Sourav Sen Gupta[2]

[1]Jadavpur University, Kolkata, India
[2]Indian Statistical Institute, Kolkata, India

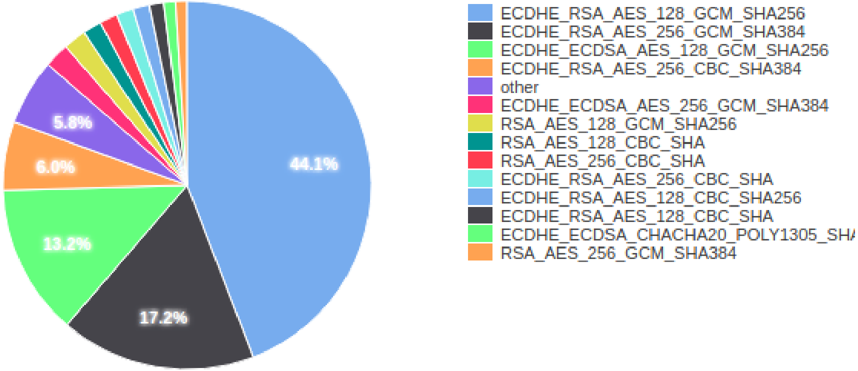
International Conference on Information System Security

IIT Bombay | 19 December 2017

RSA — Most widely used Public Key Cipher

RSA Signature is used in more than 80% SSL ciphersuites in practice.

Source of data (SSL for the last 30 days) : `https://notary.icsi.berkeley.edu/`



RSA Modulus

$$N = pq$$

Security of RSA is based on intractability / hardness of integer factorization.

RSA Modulus

$$N = pq$$

Security of RSA is based on intractability / hardness of integer factorization.

However, this assumption is violated if RSA moduli share a common factor:

$$N_1 = pq_1,\, N_2 = pq_2 \;\Rightarrow$$

$$\gcd(N_1, N_2) = p$$

RSA — The idea of Common Factor Attack

Intuitive Assumption

If two 512-bit RSA primes p and q are chosen uniformly at random, then the chance of getting the same prime twice is approx. 2^{-256} (birthday collision).

RSA — The idea of Common Factor Attack

Intuitive Assumption

If two 512-bit RSA primes p and q are chosen uniformly at random, then the chance of getting the same prime twice is approx. 2^{-256} (birthday collision).

Counter-Intuitive Reality

In 2012, Heninger et al. and Lenstra et al. independently discovered that around 0.75% of TLS certificates across the Internet shared RSA primes.

Heninger et al. also conjectured that another 1.70% may be susceptible.

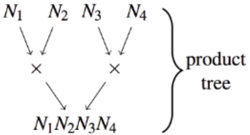
In 2013, Bernstein et al. demonstrated similar vulnerabilities in RSA moduli embedded in smart cards of Taiwan's national "Citizen Digital Certificate".

Heninger et al., USENIX Security Symposium, 2012
Lenstra et al., IACR Cryptology ePrint Archive, 2012
Bernstein et al., ASIACRYPT 2013

Common Factor Attack using Batchwise GCD

Step 1 : Product Tree

$$P = \prod N_i$$

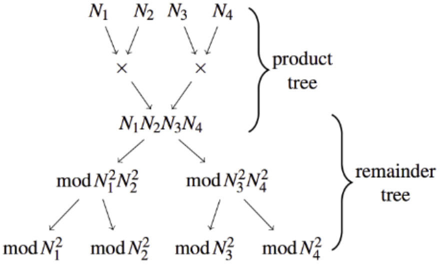


Step 1 : Product Tree

$$P = \prod N_i$$

Step 2 : Remainder Tree

$$z_i = P \bmod N_i^2$$



Common Factor Attack using Batchwise GCD

Step 1 : Product Tree

$$P = \prod N_i$$

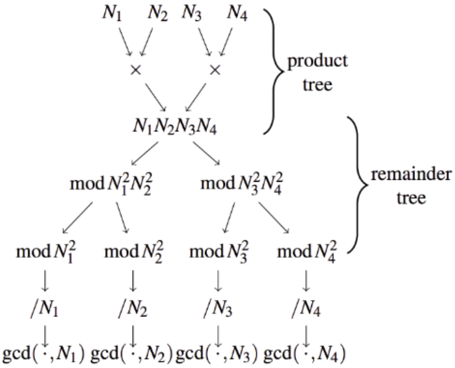
Step 2 : Remainder Tree

$$z_i = P \bmod N_i^2$$

Step 3 : Compute

$$\gcd(N_i, z_i/N_i)$$

to extract common factors (the primes)



Step 1 : Product Tree

$$P = \prod N_i$$

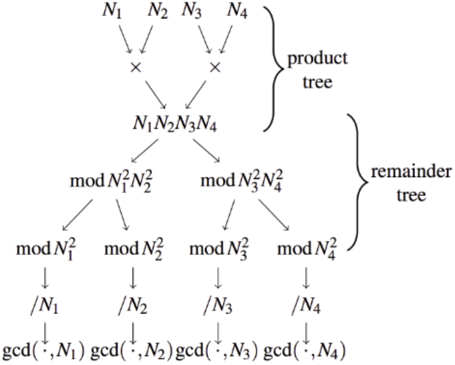
Step 2 : Remainder Tree

$$z_i = P \bmod N_i^2$$

Step 3 : Compute

$$\gcd(N_i, z_i/N_i)$$

to extract common factors (the primes)

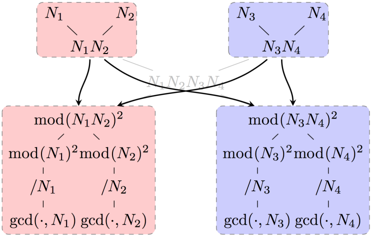


Complexity $\sim O(mn(\log n)^2 \log\log n)$

32 GB of memory and around 60 to 70 GB of storage for scratch calculations

Common Factor Attack using Parallel Batch-GCD

In 2016, Hastings et al. proposed a parallel version of batch-GCD algorithm.

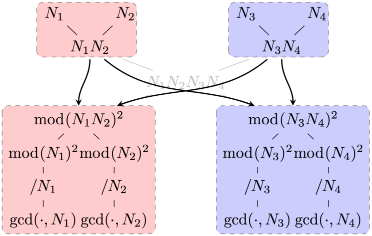


The RSA moduli dataset is partitioned into subsets and the product tree for

$$P = \prod N_i$$

is constructed independently for each subset, making this stage parallel.

In 2016, Hastings et al. proposed a parallel version of batch-GCD algorithm.



The RSA moduli dataset is partitioned into subsets and the product tree for

$$P = \prod N_i$$

is constructed independently for each subset, making this stage parallel.

But, the remainder tree is still constructed considering all subset products.

Hastings et al., Internet Measurement Conference, 2016

Our contribution — We propose a completely parallel version of batch-GCD algorithm to achieve similar results in a resource constrained environment.

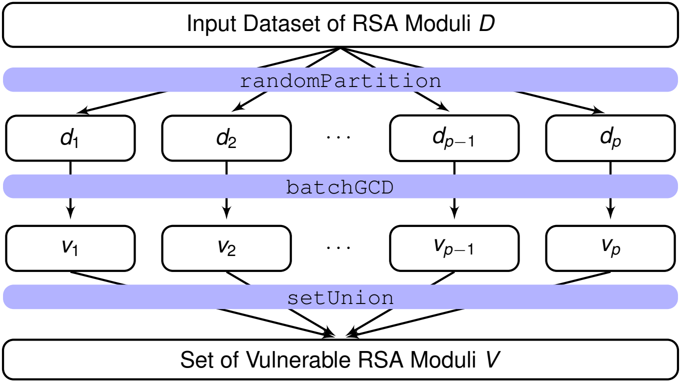


Figure : One complete iteration of our proposed Parallelized Batch-GCD

Our Idea — Parallellized Common Factor Attack

Primary concern — How many iterations are enough to recover all the factors?

Our Idea — Parallellized Common Factor Attack

Primary concern — How many iterations are enough to recover all the factors?

Theorem (Optimal number of Iterations)

Suppose there exist X vulnerable RSA moduli in input dataset D. Then our algorithm recovers an expected number of ϵX vulnerable moduli if we set

$$k \approx \frac{\log(1-\epsilon)}{\log m + \log(p-1) - \log(mp-1)},$$

where ϵ is a user-defined accuracy parameter, m is the user-defined constraint of the individual computing nodes, and $p \sim |D|/m$ is the number of partitions.

Primary concern — How many iterations are enough to recover all the factors?

Theorem (Optimal number of Iterations)

Suppose there exist X vulnerable RSA moduli in input dataset D. Then our algorithm recovers an expected number of ϵX vulnerable moduli if we set

$$k \approx \frac{\log(1-\epsilon)}{\log m + \log(p-1) - \log(mp-1)},$$

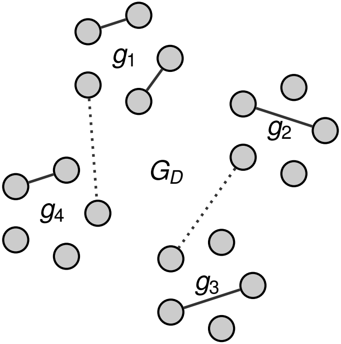
where ϵ is a user-defined accuracy parameter, m is the user-defined constraint of the individual computing nodes, and $p \sim |D|/m$ is the number of partitions.

One may interpret k given D and ϵ as : $k \approx \dfrac{\log(1-\epsilon)}{\log(|D| - |D|/p) - \log(|D|-1)}$

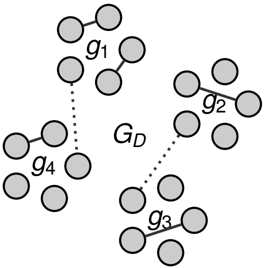
Proof of Theorem — Optimal number of Iterations

Consider the complete dataset of RSA moduli as an induced graph G_D, where the RSA moduli N_i are vertices and an edge $e_{(N_i,N_j)}$ exists iff $\gcd(N_i, N_j) > 1$.

Partitioning the RSA moduli dataset is identical to partitioning graph G_D, and thus, our algorithm discovers edges within subgraphs, and misses the others.

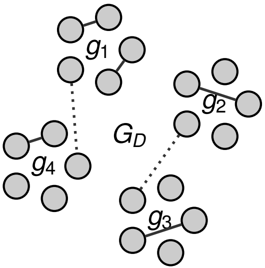


Proof of Theorem — Optimal number of Iterations



The probability that we will miss a specific edge $e_{(N_i,N_j)}$ in G_D after one execution of our algorithm:

$$
\begin{aligned}
P_{i=1} &= 1 - \frac{\text{total number of edges in } \{g_1, g_2, \ldots, g_p\}}{\text{total number of edges in } G_D} \\
&\approx 1 - \frac{\text{edges in complete supergraph of } \{g_1, g_2, \ldots, g_p\}}{\text{edges in complete supergraph of } G_D} \\
&\approx 1 - \frac{p \times \binom{m}{2}}{\binom{mp}{2}} = 1 - \frac{m-1}{mp-1} = \frac{m(p-1)}{mp-1}
\end{aligned}
$$



The probability that we will miss a specific edge $e_{(N_i,N_j)}$ in G_D after one execution of our algorithm:

$$P_{i=1} = 1 - \frac{\text{total number of edges in } \{g_1, g_2, \ldots, g_p\}}{\text{total number of edges in } G_D}$$

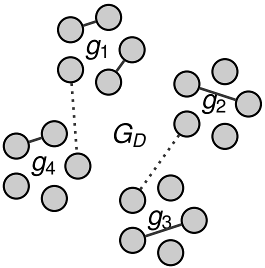
$$\approx 1 - \frac{\text{edges in complete supergraph of } \{g_1, g_2, \ldots, g_p\}}{\text{edges in complete supergraph of } G_D}$$

$$\approx 1 - \frac{p \times \binom{m}{2}}{\binom{mp}{2}} = 1 - \frac{m-1}{mp-1} = \frac{m(p-1)}{mp-1}$$

The probability that we will miss a specific edge $e_{(N_i,N_j)}$ in G_D after k iterations:

$$P_{i=k} = (P_{i=1})^k \approx \left(\frac{m(p-1)}{mp-1}\right)^k$$

Proof of Theorem — Optimal number of Iterations



The probability that we will miss a specific edge $e_{(N_i,N_j)}$ in G_D after one execution of our algorithm:

$$\begin{aligned}
P_{i=1} &= 1 - \frac{\text{total number of edges in } \{g_1, g_2, \ldots, g_p\}}{\text{total number of edges in } G_D} \\
&\approx 1 - \frac{\text{edges in complete supergraph of } \{g_1, g_2, \ldots, g_p\}}{\text{edges in complete supergraph of } G_D} \\
&\approx 1 - \frac{p \times \binom{m}{2}}{\binom{mp}{2}} = 1 - \frac{m-1}{mp-1} = \frac{m(p-1)}{mp-1}
\end{aligned}$$

The probability that we will miss a specific edge $e_{(N_i,N_j)}$ in G_D after k iterations:

$$P_{i=k} = (P_{i=1})^k \approx \left(\frac{m(p-1)}{mp-1}\right)^k$$

The fraction of edges recovered after k iterations is $\epsilon \approx 1 - \left(\frac{|D| - |D|/p}{|D| - 1}\right)^k$

Input : Set of moduli D, constraint m, accuracy ϵ
Output: V — set of vulnerable moduli in D

$p \leftarrow$ `ceiling(`$|D|/m$`)` ;
$k \leftarrow$ `chooseIteration(`m, p, ϵ`)` ;
for $i \leftarrow 1$ **to** k **do**
  $\{d_1, d_2, \ldots, d_p\} \leftarrow$ `randomPartition(`D, p`)` ;
  $\{v_1, v_2, \ldots, v_p\} \leftarrow$ `batchGCD(`$\{d_1, d_2, \ldots, d_p\}$`)` ;
  $V_i \leftarrow$ `setUnion(`$\{v_1, v_2, \ldots, v_p\}$`)` ;
end
$V \leftarrow$ `setUnion(`$\{V_1, V_2, \ldots, V_k\}$`)` ;

Our Algorithm — Parallelized Batch-GCD

Input : Set of moduli D, constraint m, accuracy ϵ
Output: V — set of vulnerable moduli in D

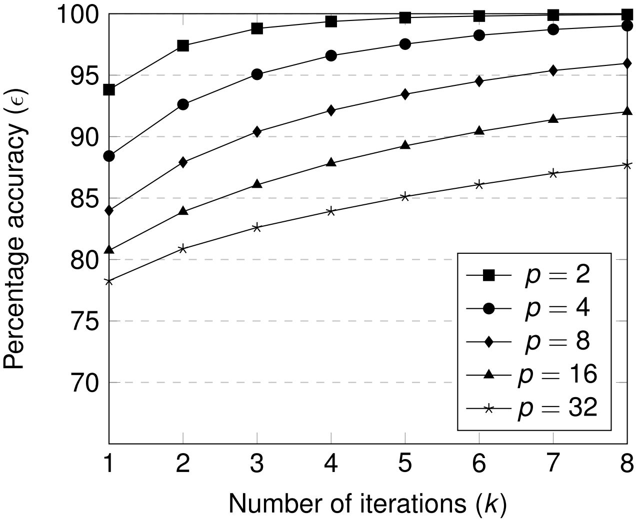
1 $p \leftarrow$ `ceiling(`$|D|/m$`)` ;
2 $k \leftarrow$ `chooseIteration(`m, p, ϵ`)` ;
3 **for** $i \leftarrow 1$ **to** k **do**
4 $\quad \{d_1, d_2, \ldots, d_p\} \leftarrow$ `randomPartition(`D, p`)` ;
5 $\quad \{v_1, v_2, \ldots, v_p\} \leftarrow$ `batchGCD(`$\{d_1, d_2, \ldots, d_p\}$`)` ;
6 $\quad V_i \leftarrow$ `setUnion(`$\{v_1, v_2, \ldots, v_p\}$`)` ;
7 **end**
8 $V \leftarrow$ `setUnion(`$\{V_1, V_2, \ldots, V_k\}$`)` ;

Line 2 : k given D and ϵ is chosen as $k \approx \dfrac{\log(1-\epsilon)}{\log(|D| - |D|/p) - \log(|D| - 1)}$

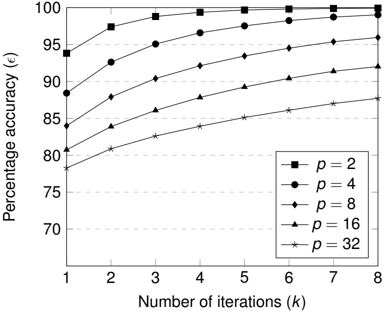
The algorithm recovers ϵ fraction of vulnerable RSA moduli from the dataset.

Our Algorithm — Practical Performance Results

Checked ϵ for various choices of $p = 2, 4, 8, 16, 32$, and $k = 1, 2, 3, \ldots, 9$



Checked ϵ for various choices of $p = 2, 4, 8, 16, 32$, and $k = 1, 2, 3, \ldots, 9$



In practice, with Intel Core i5 4210U CPU, 4 GB RAM

$p = 8$ partitions and $k = 3$ iterations resulted in $> 90\%$ recovery

$p = 32$ partitions and $k = 5$ iterations resulted in $> 85\%$ recovery

Scope — Potential extensions of Our Proposal

Extend our algorithm to include the partially parallel tree of Hastings et al.

Extend our proposal to include the more sophisticated approaches of finding vulnerable RSA moduli, using Coppersmith-type lattice based attacks, as done by Bernstein et al. on the Taiwan's national "Citizen Digital Certificate".

Lenstra, A.K., Hughes, J.P., Augier, M., Bos, J.W., Kleinjung, T., Wachter, C.:
Ron was wrong, Whit is right.
IACR Cryptology ePrint Archive **2012** (2012) 64

Heninger, N., Durumeric, Z., Wustrow, E., Halderman, J.A.:
Mining your ps and qs: Detection of widespread weak keys in network devices.
In: Proceedings of the 21th USENIX Security Symposium, Bellevue, WA, USA, August 8-10, 2012. (2012) 205–220

Hastings, M., Fried, J., Heninger, N.:
Weak keys remain widespread in network devices.
In: Proceedings of the 2016 ACM on Internet Measurement Conference, IMC 2016, Santa Monica, CA, USA, November 14-16, 2016. (2016) 49–63

Bernstein, D.J., Chang, Y., Cheng, C., Chou, L., Heninger, N., Lange, T., van Someren, N.:
Factoring RSA keys from certified smart cards: Coppersmith in the wild.
In: Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II. (2013) 341–360

Thank You!


A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.