Real World Cryptanalysis

Lec 5 (Apr 16 '24)

Course Progress:

OWF / PRG / RO	RSA / dLog / Pairings	SNARKs	MPC, PIR	LWE, FHE
Foundations	Cryptanalysis + ECC	ZKP	Privacy	Lattices

'Secure' Cryptography $\overset{?}{\Rightarrow}$ Secure Systems?

No... :("even a small bug → insecure"

* Crypto can be misused
  - Implementation Bugs
  - Side Channel Attacks
    e.g. Time it takes to run s/w ⇒ reveal secret info.
  - Today ← Unsafe Optimizations
  - Bad Randomness (GCD attack)
    "not actually random".

* Some 'Hard' problems are not very hard ... → Next lecture

Infineon Attack: (2017)

Big deal! an attack on RSA KeyGen

* 10M devices ("smartcards") recalled!!

* 50% of Estonian national eIDs were vulnerable!!

* Amazing paper :) (Link online)

Background:

- RSA is surprisingly fragile to key leakage.

- Optimizations can easily break security. → smartcards are weak devices, so optimizations are welcome. : Classic e.g. of security vs perf tradeoff.

* Standard RSA KeyGen: "Rejection Sampling"
need large p, q :-

$p \leftarrow 1$

while p is not prime: → $\exists$ poly time algos to check primality.

$p \xleftarrow{\$} n$-bit integer.

for $\lambda = 128$ ("128-bit security"), $n \approx 1024$.

Similarly, sample q.

Set $N = pq$. ...

* Common to use secure H/W, e.g smartcard to securely store and use RSA secret keys.

Issue: Smartcards have limited compute.
Sampling 1024-bit numbers,] Slow!
Testing primality

Optimization in Infineon Smartcards:-

sample p, q of the form:-

$$R * M + (65537^a \bmod M)$$

* Authors inferred this structure by analysing lots of RSA keys!!

* Here, M : public : $2*3*5 \cdots * p_i$
Product of first $i=126$ primes.
For $n=1024$, M: 971 bits. $n=256$: M: 220 bits.
(different i for different key sizes)

* R, a values: different for each prime.

For $n = 1024$,

k: random ~ 53-bit number

a: random > 100-bit number.

(Flawed) Intuition:

$\geq 2^{128}$ choices for p.

$\geq 2^{128}$ " " q

$\Big\}$ 2^{256} choices for N. $\Rightarrow$ Secure.

NO!

But, Infineon moduli are easy to factor...

MAIN tool: Coppersmith's Attack:

If you know half of the bits of p OR q, then, you can factor $N = pq$ in poly time!!!

* Surprising: given only 512 bits of p, can recover all remaining bits of p and q!!!

$(512 + 1024)$ bits!

Thm: (Coppersmith '96).

Let $N = pq$ be an RSA modulus with primes $p > q$.

Let $f(x) \in \mathbb{Z}_N[x]$ be a monic polynomial of degree δ. Then, we can find all $x_0 \in \mathbb{Z}$ s.t.

$$f(x_0) = 0 \bmod p \quad \text{and} \quad |x_0| \leq N^{\frac{1}{4\delta}}.$$

in time $\text{poly}(\log N, \delta)$.

$f(x) \in \mathbb{Z}_N[x]$: polynomial with all coefficients in $\mathbb{Z}_N = \{0, 1, \ldots, N-1\}$.

Monic: Leading coefficient $= 1$.

e.g. $f(x) = \underline{\underline{x^3}} + 4x^2 + 7x + 27$

↳ Monic; degree 3.

☆ you dont NEED to know p!!!

Coppersmith's can find all x_0 s.t.

$$f(x_0) = 0 \bmod p \quad \text{and} \quad |x_0| \leq N^{\frac{1}{12}}$$

Interface: $N, \overbrace{f}^{\text{coefficients}}, \delta \rightarrow \boxed{\text{Coppersmith}} \rightarrow \{x_0\}$

Corollary: # Solutions $\leq \text{poly}(\delta, \log N)$.

Algorithm for factoring N :-

$$\Big(\underbrace{N}_{\text{known}} = \underbrace{pq}_{\text{unknown}} \quad \text{where } p = \underbrace{R}_{\text{unknown}} \cdot \underbrace{M}_{\text{known}} + \big(65537^{\overset{\text{unknown}}{a}} \bmod \underbrace{M}_{\text{known}}\big)\Big)$$

STEP 1: Guess a (i.e. Try all possible values)

STEP 2: Factorize N using coppersmith.] lets start with this.

Lets say we guessed a correctly. How do we factor?

$$p = \underbrace{R \cdot M}_{\text{known}} + \underbrace{\left(65537^a \bmod M\right)}_{\text{known}}$$

* we want to solve for R!

Let $C_2 = 65537^a \bmod M$.

Define $f(x) = x + \dfrac{C_2}{M} \in \mathbb{Z}_N[x]$.

$M^{-1} \bmod N$: if $\gcd(M, N) \neq 1$: can factorize N.
if $\gcd(M, N) = 1$, we can find a, b s.t. $aM + bN = 1$, then, $a = M^{-1} \bmod N$. (Extended Euclid)

1) f is Monic, $\deg(f) = 1$

2) $f(R) = R + \dfrac{C_2}{M} = \dfrac{p}{M} = 0 \bmod p$

starting to look like CS...

3) $|k| \leq N^{1/4}$ Because,

$p = k \cdot M + (\cdots \bmod M) \sim 2^{1024}$

so, $k \sim \dfrac{2^{1024}}{M} \sim 2^{54} \ll N^{1/4} = 2^{512}$.

M: ~971 bits

so, we can use Coppersmith with inputs $(N, f, \delta = \deg(f) = 1)$,

to get all solutions $k_1, k_2, \ldots$ s.t. $f(k_i) = 0 \bmod p$, $|k_i| \leq N^{1/4}$.

so,

<u>STEP 2</u>: (we guessed a in step 1)

2.1: Use Coppersmith to get $k_1, k_2 \ldots$

2.2: For each value k_i, let $p_i = k_i M + (65537^a \bmod M)$.

if N divides p_i: we're done!

[By facts 1, 2, 3 above, we are guaranteed that $k_j = k$ for some j, meaning $p_j = p$.]

Conclusion: choosing 54-bit k adds little security!

Back to STEP 1 : Guessing a :

Recall, $p = k \cdot M + (65537^a \% M)$

So, we actually need to guess

$$c_2 = 65537^a \bmod M.$$

We can Try all values of $\underline{65537^a \bmod M}$,

i.e. $\{65537^0 \% M, 65537^1 \% M, 65537^2 \% M \dots\}$

Q: How big is this set?

$\Rightarrow$ order of 65537 in $\mathbb{Z}_M^* = \{1, 2, \dots, M-1\}$

☆ section on Thursday (Apr 18)

Lets say order of 65537 in $\mathbb{Z}_M^*$ is r.

(i.e. $\text{ord}(65537) = r$)

Then, we just need to try

$$a = \{0, 1, 2, \dots, r-1\}.$$

Q: How to find r?

Smallest nonzero integer s.t

$$\underline{65537^r = 1 \bmod M.}$$

note, $\Rightarrow$ $65537^{r} \equiv 1 \bmod 2$.
$\equiv 1 \bmod 3$
$\vdots$
$\equiv 1 \bmod p_{126}$.

lets say, $ord(65537)$ in $\mathbb{Z}_{p_i}$ is r_{p_i}
Then, $r = LCM(r_2, r_3, \ldots, r_{p_{126}})$

But unfortunately, for $n = 1024$,
$M = 2 \cdot 3 \cdot \ldots \cdot p_{126}$,
$r \sim 2^{255}$: TOO BIG :(

Optimization #1 :

Observe that, we used Coppersmith to find k: 54 bits.
But, coppersmith can solve for upto $N^{1/4}$: upto $\frac{\log N}{4}$ bits : 512 Bits!!

[use CS to find more bits]

Idea : Find a different M' s.t. :-

(1) $M' \mid M$: also a product of primes

This ensures that p is still of the same form:-

(Skipped in class)

Let $M = c * M'$. then,

$p = k \cdot M + (65537^a \% M)$

Let, $65537^a = k_1 \cdot M + r_1$ for some k_1, and some $r_1 \in \{0, 1, \ldots, M-1\}$

i.e. $p = kM + r_1$. Let $r_1 = k_2 \cdot M' + r_2$ for some $k_2 \in \mathbb{N}$ and $r_2 \in \{0, 1, 2, \ldots, M'-1\}$.

Then, $p = k \cdot c \cdot M' + k_2 M' + r_2$. Also,

$65537^a \% M' = (k_1 M + (k_2 M' + r_2)) \% M' = r_2$.

So, $p = k' \cdot M' + (65537^{a'} \bmod M')$

(2). ord(65537) is small mod M', so we dont have to try too many values of a.

(3.). Because we'll use Coppersmith to solve for k',

k' must be $\leq N^{\frac{1}{48}} = N^{1/4}$.

note, $p > q$, so,

$p \geq 2^{1024} = N^{1/2} \Rightarrow k'M' + \cdots \geq N^{1/2}$

so, $M' \geq N^{1/4}$. (loose lower bound)

Tradeoff:

smaller M' $\Rightarrow$ smaller $ord(65537)$
$\Rightarrow$ fewer a guesses
(GOOD)

$\Downarrow$

But also, Coppersmith slower
because need to solve for larger k'.
(BAD).

Cant try all values of M', cuz there's too many!

Paper: Use greedy heuristics to find the 'optimal' value of M'.

[Minimize $ord(65537)$ in $\mathbb{Z}^*_{M'}$, while ensuring $M' \geq N^{1/4}$ and $M' | M$.]

Note: finding the optimal M': one time process!

STEP 1: Try all values of a' in $\{0, 1, \ldots, \mu' - 1\}$ where

μ': order of 65537 in $\mathbb{Z}^*_{M'}$.

FINAL Results :-

Key Size	# a guesses	Time	AWS $	Energy $
512	$\leq 2^{20}$	2 CPU hrs.	0	0
1024	$\leq 2^{30}$	98 CPU days	76	2
		GPT3: 355 GPU yrs!!!		
2048	$\leq 2^{35}$	140 CPU yrs	40K.	~1K.
3072.	$\leq 2^{99}$.	$> 10^{25}$ CPU yrs.	>	>.

* Trivially Parallelizable !!

Bonus: Fingerprinting :-

Given an RSA public key, can I test if it is vulnerable?

Infineon public key :-

$$N = p \cdot q.$$

$$= (k \cdot M + (65537^a \bmod M))(k' M + (65537^{a'} \bmod M))$$

$$= k \cdot k' \cdot M^2 + k \cdot M \cdot (65537^{a'} \bmod M) + k' \cdot M \cdot (65537^a \bmod M) + 65537^{a+a'} \bmod M.$$

so, $N \bmod M = 65537^{a+a'}$

i.e. N is in the subgroup of $\mathbb{Z}_M^*$ generated by 65537. i.e. in

$[65537] = \{65537^0 \% M, 65537^1 \% M, \ldots\ldots\}$

so, to check if N is vulnerable, check if N is $65537^c \bmod M$ for some c.

Q1. HOW?

we basically want to test whether $\log_{65537}(N \bmod M)$ is defined.

" check whether dlog exists "

But, isn't DLog hard in groups?

NOT when group size is "smooth"; Pohlig-Hellman algo.

Here, we want DLog of $(N \bmod M)$ in subgroup $[65537] \subseteq \mathbb{Z}_M^*$.

$G = \{65537^0 \% M, 65537^1 \% M, \ldots\ldots\}$

A number is smooth if all its prime factors are small.

$M = 2 \cdot 3 \cdot 5 \cdots\cdots p_{126}$: ~ smooth.

$|G|$ divides $|\mathbb{Z}_M^*| = \phi(M)$

$\phi(M) = (2-1)\cdot(3-1)\cdot(5-1) \cdots\cdots (p_{126}-1)$: ~ smooth

so, $|G|$ also smooth!!

<u>**Q2**</u> What about false positives?

If N were fully random, then

* $N \bmod M$ could be any value in $\mathbb{Z}_M^*$.

i.e. $\phi(M)$ different values.

* But, for Infineon keys:

$N \bmod M$ is in the subgroup $[65537]$, of size $\mathrm{ord}(65537)$

Informally, the probability that

$N \% M$ for random N falls in the subgroup $[65537]$ is LOW.

Empirically: <u>NO</u> false +ves found!

WRAP UP :

* Optimizations can BREAK security !.

↓

one of MANY examples of security vs. Perf. . tradeoffs !!!

Optimization #2: (Skipped in Class).

Recall that we compute

$\log_{65537}(N \bmod M)$ for fingerprinting

$$c \in \{0, 1, 2, \ldots, r-1\}$$

where, $r = \text{ord}(65537)$ in $\mathbb{Z}_M^*$.

Then, $c = a + \hat{a} \bmod r$,

where $p = k \cdot M + (65537^{a} \,\%\, M)$

and $q = \hat{k} \cdot M + (65537^{\hat{a}} \,\%\, M)$

and $a, \hat{a} \in \{0, 1, \ldots, r-1\}$.

It is easy to see that atleast one of $a, \hat{a} \in \left[\frac{c}{2}, \frac{c+r}{2}\right] = I$

So, we only need to try all values of a in I, to find either p or q.