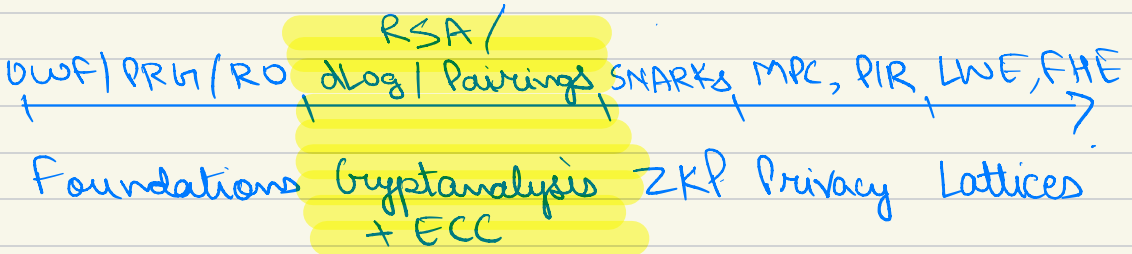


Real World Cryptanalysis

Lec 5 (Apr 13 '26)

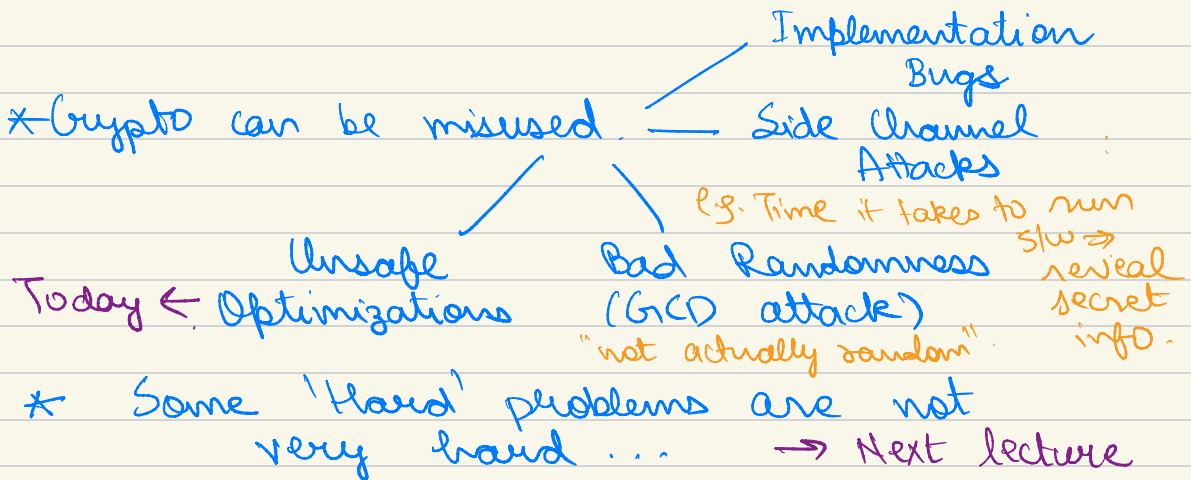
Course Progress:



'Secure' Cryptography \Rightarrow Secure Systems?

No ... ☹️

"even a small bug \rightarrow insecure"



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

Sublinear 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 $p > q$:-

$p \leftarrow \perp$

while p is not prime :

$p \leftarrow \mathbb{Z}$, n -bit integer.

↪ \exists poly-time algos to check primality.

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

Similarly, sample g .

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, Testing primality } Slow!

Optimization in Infineon Smartcards:

sample p, g of the form: -

$$R * M + (65537^a \text{ mod } M)$$

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

* Here, $M \approx$ public: $2 * 3 * 5 \dots * p_i$

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 } 2^{256} choices for N . \Rightarrow Secure. **NO!**

But, Inefficient moduli are easy to factor...

MAIN tool: Coppersmith's Attack:

If you know half of the bits of factor 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.

$$f(x_0) \equiv 0 \pmod{p} \text{ and } |x_0| \leq N^{\frac{1}{\delta}}$$

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

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

Monic: leading coefficient = 1.

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

↳ Monic; degree 3.

★ you don't NEED to know p !!!

Coppersmith's can find all x_0 s.t.

$$f(x_0) \equiv 0 \pmod{p} \text{ and } |x_0| \leq N^{\frac{1}{\delta}}$$

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

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

Algorithm for factoring N :-

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

unknown

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?

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

* we want to solve for R ! To use CS, we need a monic poly f s.t. R is a root mod p .

$$\text{Let } c_2 = 65537^a \bmod M.$$

$$\text{Define } f(x) = x + \frac{c_2}{M} \in \mathbb{Z}_N[x].$$

$M^{-1} \bmod N$: 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, $\deg(f) = 1$

$$2) f(R) = R + \frac{c_2}{M} = \frac{p}{M} = 0 \bmod p$$

Starting to look like CS...

$$3) |K| \leq N^{1/4} \quad \text{Because,} \\ p = R \cdot M + (\dots \bmod M) \sim 2^{1024}$$

$$\text{so, } R \sim \frac{2^{1024}}{M} \sim 2^{54} \ll N^{1/4} = 2^{512} \\ M : \sim 971 \text{ bits}$$

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

to get all solutions R_1, R_2, \dots
s.t. $f(R_i) = 0 \bmod p, |R_i| \leq N^{1/4}$.

** one of the R_i 's is k !!!*

so, STEP 2: (we guessed a in step 1)

2-1: Use Coppersmith to get R_1, R_2, \dots

2-2: For each value R_i ,
let $p_i = R_i M + (65537^a \bmod M)$

if N divides p_i : we're done!

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

Conclusion : choosing 54-bit k adds little security!

Back to STEP 1: guessing a :-

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

So, we actually need to guess

$$C_2 = 65537^a \pmod{M}.$$

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

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

Q How big is this set?

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

Lets say order of 65537 in \mathbb{Z}_M^*

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

Then, we just need to try

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

Q How to find μ ?

Smallest nonzero integer s.t.

$$\underline{65537^\mu = 1 \pmod{M}}.$$

note, $\Rightarrow 65537^k \equiv 1 \pmod{2}$
 $\equiv 1 \pmod{3}$
 \vdots
 $\equiv 1 \pmod{P_{126}}$

lets say, ord(65537) in \mathbb{Z}_{P_i} is π_{P_i}
 Then, $\mu = \text{LCM}(\pi_2, \pi_3, \dots, \pi_{P_{126}})$

But unfortunately, for $n = 1024$,

$$\mu = 2 \cdot 3 \cdot \dots \cdot P_{126},$$

$$\mu \sim 2^{255}$$

Too Big ∞
 to BRUTEFORCE

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!

\rightarrow [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 :-

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

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

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

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

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

$$65537^a \cdot M' = (R \cdot M + (k_2 M' + r_2)) \cdot M' = r_2$$

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

(2). $\text{ord}(65537)$ is small mod M' , so we don't have to try too many values of a .

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

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

note, $p > q$, so,

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

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

Tradeoffs:

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

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

Can't 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.

Once we've found the optimal M' ,
we can factor any N sampled by
Gubincov chips as follows:-

1.) Try all values of a' in
 $\{0, 1, \dots, M'-1\}$ where

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

2.) For each value of a' , use
Coppersmith to get $\{k_1^{(a')}, \dots, k_y^{(a')}\}$
 \rightarrow Check if $k_i^{(a')} \cdot M' + (65537^a) \cdot M$ is a factor of N

FINAL Results : $\frac{1}{\text{mod}(65537)} \cdot M^1$

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
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?

Infixion public key :-

$$N = p \cdot q$$

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

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

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

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

$$[65537] = \{ 65537^0 \cdot M, 65537^1 \cdot M, \dots \}$$

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 $[65537] \subseteq \mathbb{Z}_M^*$.

↓

$$G = \{ 65537^0 \cdot M, 65537^1 \cdot M, \dots \}$$

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

$$M = 2 \cdot 3 \cdot 5 \cdot \dots \cdot p_{26} : \sim \text{smooth.}$$

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

$$\phi(M) = (2-1) \cdot (3-1) \cdot (5-1) \cdot \dots \cdot (p_{26}-1) : \sim \text{smooth}$$

so, $|G|$ also smooth!!

Q2 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 Inverse keys:

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

Informally, the probability that

$N \bmod M$ for random N falls in the subgroup $[65537]$ is Low.

Empirically: No 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

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

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

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

$$\text{where } p = R \cdot M + (65537^a) \cdot M$$

$$\text{and } q = \hat{R} \cdot M + (65537^{\hat{a}}) \cdot M$$

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

It is easy to see that at least one

$$\text{of } a, \hat{a} \in \left[\frac{c}{2}, \frac{c+r}{2} \right] = I$$

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