Cryptanalysis using lattices

Prof. dr. ir. Frederik Vercauteren

COSIC, KU Leuven
Computer Algebra for Cryptography (B-KUL-H0E74A)

2022-2023

Small decryption exponent RSA

Knapsack cryptosystems

Coppersmith's method and applications

Small decryption exponent RSA

Knapsack cryptosystems

Coppersmith's method and applications

The RSA Cryptosystem

- Find 2 primes p and q of at least 1024 bits and set $N = p \cdot q$
- Compute Euler Phi
$$\varphi(n) = (p-1)(q-1)$$
- Choose e co-prime to $\varphi(N)$ $(\neq \pm 1)$
- Compute $d = e^{-1} \bmod \varphi(N)$
- **public key**: (e, N), **private key**: (d, N) or (p, q)
- Encryption: $c = m^e \bmod N$
- Decryption: $m = c^d \bmod N$

Wiener's attack

- Wiener '90: Let (N, e) be an RSA public key and d the corresponding private key. For
$$d \leq \frac{1}{3}\sqrt[4]{N}$$
the modulus N can be factored in time $O(\log^2 N)$
- Wiener: continued fractions, we will use lattices (with slightly worse constant)
- Idea: write RSA key equation:
$$ed = 1 + k\varphi(N) = 1 + k(N - p - q + 1)$$
and note $k = (ed - 1)/\varphi(N) < d$ (assume $e < \varphi(N)$)

Wiener's attack

- Rewriting the equation gives

$$ed - kN = 1 - k(p + q - 1)$$

- Consider the lattice

$$L = \begin{pmatrix} e & \lfloor\sqrt{N}\rfloor \\ N & 0 \end{pmatrix}$$

- Vector $\mathbf{v} = (ed - kN, d\lfloor\sqrt{N}\rfloor)$ is in the lattice and has length $\approx \sqrt{5} \cdot d \cdot \sqrt{N}$
- Comparing to volume $\approx N^{3/2}$, $\mathbf{v}$ will likely be shortest vector if

$$\sqrt{5} \cdot d \cdot \sqrt{N} < \sqrt{\frac{1}{\pi e}} N^{3/4} \qquad \Leftarrow \qquad d \leq \frac{1}{7}\sqrt[4]{N}$$

- **Exercise:** show that putting $\approx \sqrt{N}$ on top right is essentially optimal
- **Exercise:** why does this attack not work for small **encryption** exponent?

Wiener's attack

- MAGMA example:

```
> N :=
  1116870254237723980740312325116906288704452778201991357593077;
```

- ```
  > print "Decryption exp is", -Basis(L)[1][2] / Floor(Sqrt(N));
  Decryption exp is 280818091651919
  ```
  ```

Small decryption exponent RSA

Knapsack cryptosystems

Coppersmith's method and applications

Knapsack cryptosystems

- Merkle–Hellman (1978): *Hiding information and signatures in trapdoor knapsacks*
- Subset sum problem: given n different positive weights $w_1, \ldots, w_n$, and a target weight s (size of knapsack), decide whether there exist $x_i \in \{0, 1\}$ such that

$$s = \sum_{i=1}^{n} x_i w_i$$

- **Exercise:** deciding the existence is the same as finding the x_i
- Can assume number of non-zero x_i is $\leq \lfloor n/2 \rfloor$ (replace s by $\sum_i w_i - s$ if needed)
- Subset sum problem is NP-complete
- Density of sequence $w_1, \ldots, w_n$ is

$$d = \frac{n}{\log_2 \max_i\{w_i\}}$$

- Rule of thumb: $d \approx 1$ is hardest case, low density and high density are easier

Solving low density subset sum with lattices

- Given weights $w_1, \ldots, w_n$ and target sum s, form lattice

$$\begin{pmatrix} 1 & 0 & \ldots & Nw_1 \\ 0 & 1 & \ldots & Nw_2 \\ \vdots & & & \vdots \\ 0 & \cdots & 1 & Nw_n \\ 0 & 0 & \cdots & Ns \end{pmatrix}$$

- Vector $(x_1, \ldots, x_n, 0)$ is in lattice and length is $\leq \sqrt{n/2}$
- Choosing $N > \sqrt{n/2}$ forces last entry of shortest vector to be 0
- Lagarias–Odlyzko: for random weights w_i of size $2^{\beta n}$ with $\beta = 1.5473$ (density < 0.6463), really shortest vector
- Heuristically:

$$\sqrt{n/2} < \sqrt{\frac{n+1}{2\pi e}} \mathrm{vol}(L)^{\frac{1}{n+1}} \quad \text{with } \mathrm{vol}(L) = Ns \approx N(n/2)2^{\beta n}$$

- Coster et al.: replace last row by $(1/2, 1/2, \ldots, Ns)$, then density can grow to < 0.9408

Solving low density subset sum with lattices

- MAGMA example (for density 1/2):

```
> n := 10;
> w := [];
> for i in [1..n] do
>     w cat:= [Random([1..2^(2*n)])]; // density 1/2
> end for;
> ind := Random(Subsets({1..n}, Floor(n/2)));
> s := &+[w[i] : i in ind];
> print "Weights are", w;
> print "Target is", s, "with weight indices", ind;
> L := IdentityMatrix(Integers(), n + 1);
> for i in [1..n] do L[i][n + 1] := N*w[i]; end for;
> L[n + 1][n + 1] := Ceiling(Sqrt(n/2))*s;
> ShortestVectors(Lattice(L));
```

Knapsack cryptosystems

- Subset sum is easy for superincreasing sequences, where

$$w_i > \sum_{k=1}^{i-1} w_k$$

- Example: $w_i = 2^i$ is superincreasing, the x_i are the bits of s

Knapsack cryptosystems

- Subset sum is easy for superincreasing sequences, where

$$w_i > \sum_{k=1}^{i-1} w_k$$

- Example: $w_i = 2^i$ is superincreasing, the x_i are the bits of s
- Merkle–Hellman idea: try to hide a superincreasing sequence $\{w_i\}$ as follows
  - Choose modulus $N > \sum_{i=1}^{n} w_i$, a random multiplier W and a permutation π of the integers $\{1, \ldots, n\}$
  - Hide the superincreasing sequence by giving out sequence

$$a_i = W w_{\pi(i)} \bmod N$$

Knapsack cryptosystems

- Public key: sequence $a_1, \ldots, a_n$
- Private key: $W, \pi, N, w_1, \ldots, w_n$
- Encryption: given message of n bits $m_1, \ldots, m_n$ simply compute

$$c = \sum_{i=1}^{n} m_i a_i$$

- Decrypt: multiply by $W^{-1} \bmod N$, solve subset sum problem using $w_1, \ldots, w_n$ and invert permutation π

Knapsack cryptosystems: attack

- Goal: to find multiplier $U = W^{-1} \bmod N$ and modulus N such that $Ua_i \bmod N$ are elements of superincreasing sequence
- Assume there is no permutation, then
$$Ua_i - k_i N = w_i$$
and $w_i < N/2^{n-i}$ (for small i very small compared to N)
- Like Wiener attack on RSA but now do not know k_i or N
- **Exercise:** show that
$$|a_i k_1 - a_1 k_i| < N/2^{n-i-1}$$
which again is very small (compared to $a_i k_1$ and $a_1 k_i$)
- $\Rightarrow$ $U = k_1$ and $N = a_1$ good candidate for superincreasing $Ua_i \bmod N$
- Likely not equal to original U and N but still useful

Knapsack cryptosystems: attack

- Build lattice (for small ℓ)

$$\begin{pmatrix} \lambda & a_2 & \dots & a_\ell \\ 0 & -a_1 & \dots & 0 \\ \vdots & & & \vdots \\ 0 & \dots & & -a_1 \end{pmatrix}$$

- Vector $[\lambda k_1, k_1 a_2 - k_2 a_1, \dots, k_1 a_\ell - k_\ell a_1]$ is short
- Choose λ such that λk_1 similar size as other entries
- Use $U = k_1$ and $N = a_1$, hopefully Ua_i mod N superincreasing
- Already good enough in practice to take $\ell = 5$, so only very small lattices involved
- Try all ℓ-element sequences in turn to find correct permutation: $O(n^5)$ tries
- Conclusion: knapsack cryptosystems completely broken

Knapsack cryptosystems: attack example (without π)

- Private key:
  - $N = 2609$,
  - $W = 2525$ (so $U = 528$),
  - $w_1 = 7, w_2 = 20, w_3 = 35, w_4 = 71, w_5 = 140, w_6 = 307, w_7 = 651, w_8 = 1301$
- Public key:
  - $a_1 = 2021, a_2 = 929, a_3 = 2278, a_4 = 1863, a_5 = 1285, a_6 = 302, a_7 = 105, a_8 = 294$
- Build lattice for $\ell = 3$ with $\lambda = 1/8$:

$$\begin{pmatrix} \frac{1}{8} & 929 & 2278 \\ 0 & -2021 & 0 \\ 0 & 0 & -2021 \end{pmatrix} \quad \text{with LLL-reduction} \quad \begin{pmatrix} \frac{409}{8} & 13 & 21 \\ \frac{63}{8} & -82 & 23 \\ \frac{385}{8} & -52 & -84 \end{pmatrix}$$

- Guess $U = k_1 = 409$, $N = a_1 = 2021$
- Note: wrong guess, yet $Ua_i \bmod N$ yields

$$0, 13, 21, 50, 105, 237, 504, 1007$$

which is superincreasing, so can be used to decrypt (except m_1, can be guessed)

Small decryption exponent RSA

Knapsack cryptosystems

Coppersmith's method and applications

Coppersmith's method

- Coppersmith '96: find small roots of a modular polynomial
- Setup:
  - integer N of unknown factorisation (e.g. RSA modulus)
  - degree d polynomial $f(x) = x^d + f_{d-1}x^{d-1} + \cdots + f_1 x + f_0$
  - some bound $B > 0$
- Problem: find all integers x_0 with

$$|x_0| < B \quad \text{and} \quad f(x_0) \equiv 0 \bmod N$$

- If N is prime power: foot finding algorithms (see Lecture 8)
- If factorization of N is known: Chinese Remainder Theorem
- Power of method lies in fact that **factorization of N need not be known**

Coppersmith's method: bound B

- B depends on degree d and on N
- for $d > 1$, cannot have $B \approx N$ since then could solve

$$x^3 - c \equiv 0 \bmod N$$

- Note: RSA equation easy to solve when $m < N^{1/3}$, since then

$$x^3 - c = 0 \quad \text{over } \mathbb{Z}$$

- **Coppersmith**: can find in time polynomial in $(\log N, d, 1/\epsilon)$ all roots x_0 of $f(x) \equiv 0 \bmod N$ with $|x_0| \leq \frac{1}{2} N^{1/d - \epsilon}$

Coppersmith's method

- Finding roots of a polynomial over $\mathbb{Z}$ is easy
- Idea: build polynomial $F(x)$ over $\mathbb{Z}$ with same roots x_0

Coppersmith's method

- Finding roots of a polynomial over $\mathbb{Z}$ is easy
- Idea: build polynomial $F(x)$ over $\mathbb{Z}$ with same roots x_0
- Assume that $F(x_0) \equiv 0 \bmod N^m$ for some $m \geq 1$ and $|F(x_0)| < N^m$, then $F(x_0) = 0$ over integers.

Coppersmith's method

- Finding roots of a polynomial over $\mathbb{Z}$ is easy
- Idea: build polynomial $F(x)$ over $\mathbb{Z}$ with same roots x_0
- Assume that $F(x_0) \equiv 0 \bmod N^m$ for some $m \geq 1$ and $|F(x_0)| < N^m$, then $F(x_0) = 0$ over integers.
- Problem: only have a bound B on x_0, not actual value!
- Let $F(x) = \sum_{i=0}^{n-1} F_i x^i$, then clearly

$$|F(x_0)| \leq \sum_{i=0}^{n-1} |F_i| B^i$$

- Cauchy-Schwartz: on vectors $[|F_i|B^i]_i$ and $[1, \cdots, 1]$

$$|F(x_0)| \leq \sqrt{n} \cdot ||F(xB)||$$

Howgrave-Graham's lemma

- Let $F(x)$ be a polynomial with n monomials and

$$F(x_0) \equiv 0 \bmod N^m \quad \text{with} \quad |x_0| < B$$

$$||F(xB)|| < \frac{N^m}{\sqrt{n}}$$

- Then $F(x_0) = 0$ over the integers

Coppersmith's method

- Given $f(x_0) \equiv 0 \bmod N$, for $g_{i,j}(x) = N^{m-i}x^j f(x)^i$

$$g_{i,j}(x_0) \equiv 0 \bmod N^m$$

- Also holds for every linear combination of the $g_{i,j}$
- Search for linear combination satisfying H-G lemma
- In other words: look for linear combination F with

$$||F(xB)|| < \frac{N^m}{\sqrt{n}}$$

Coppersmith's method

- Order the polynomials $g_{i,j}(x) = N^{m-i}x^j f(x)^i$ for $0 \leq i \leq m$ and $0 \leq j < d$ by degree
- Build the lattice L with rows coefficient vectors of $g_{i,j}(xB)$
- $\dim(L) = (m+1)d$ and is the number of monomials n of lin. comb. F
- **Exercise:** volume of L is given by

$$\mathrm{vol}(L) = B^{n(n-1)/2} N^{nm/2}$$

- LLL returns vector $F(xB)$ with norm

$$||F(xB)|| \leq 2^{(n-1)/4} \mathrm{vol}(L)^{1/n} = 2^{(n-1)/4} B^{(n-1)/2} N^{m/2}$$

Coppersmith's method

- Howgrave-Graham's condition finally gives

$$2^{(n-1)/4} B^{(n-1)/2} N^{m/2} < \frac{N^m}{\sqrt{n}}$$

- Given f and N, for every choice of m, obtain bound B for which method works

$$B < \frac{N^{m/(n-1)}}{\sqrt{2}n^{1/(n-1)}}$$

- Since $n = (m+1)d$, obtain the asymptotic bound $N^{1/d}$ for $n \to \infty$
- Note that LLL suffices, no need for SVP oracle

Attack on RSA with small e and stereotyped m

- Assume (N, e) is an RSA public key with e small
- Assume the encrypted message is stereotyped, i.e.

$$m = M2^k + x_0$$

  where M is known and $x_0 < 2^k$ unknown
- Coppersmith: if $|x_0| < N^{1/e}$, then can find x_0 in polynomial time in $(\log N, e)$
- Apply Coppersmith to $f(x) = (M2^k + x)^e - c \equiv 0 \bmod N$
- **Exercise:** what if unknown part is not the least significant part?

Attack on RSA with small e and stereotyped m in MAGMA

```
repeat
   p := RandomPrime(1024 : Proof := false);
   q := RandomPrime(1024 : Proof := false);
until GCD((p-1)*(q-1), 3) eq 1;  // not efficient but to save space

N := p*q;   ZN := Integers(N);

e := 3;  // encryption exponent
k := 256;  // key length we want to encrypt
M := Random(2^760);  // fixed random padding of length 760 bits
key := Random(2^256);  // key we want to transport
c := Integers() ! (ZN ! (key + 2^k*M))^e;  // encryption of the key
                                           // using the fixed padding
```

Attack on RSA with small e and stereotyped m in MAGMA

```
Zx<x> := PolynomialRing(Integers());
f := (x + 2^k*M)^e - c;  // polynomial used in encryption
time s := SmallRoots(f, N, 2^256);  // Coppersmith on f with bound 2^256
----> Time: 0.000
s[1] eq key;
----> true
```

Attack on RSA with partial knowledge of secret key

- **Coppersmith (general)**: Let N be an integer of unknown factorization, with unknown divisor $b \geq N^{\beta}$. Then for $f(x)$ a monic polynomial of degree d, can find in polynomial time in $(\log N, d, 1/\epsilon)$ all roots x_0 of

$$f(x) \equiv 0 \bmod b \text{ with } |x_0| \leq \frac{1}{2} N^{\beta^2/d-\epsilon}$$

- Assume you know a good approximation $\tilde{p}$ to p, i.e.

$$\tilde{p} = p + \Delta \quad \text{with } |\Delta| < N^{1/4}$$

- Note that the degree 1 polynomial $x - \tilde{p}$ has zero Δ modulo p, so can apply general Coppersmith with $\beta = 1/2$

Attack on RSA with partial knowledge of secret key in MAGMA

```
repeat
   p := RandomPrime(1024 : Proof := false);
   q := RandomPrime(1024 : Proof := false);
until GCD((p-1)*(q-1), 3) eq 1;  // not efficient but to save space

N := p*q;
beta := Log(N, p) - 0.01;  // this is around 0.5
Zx<x> := PolynomialRing(Integers());

ptilde := p + Random(2^460);   // differing lowest 460 bits
f := x - ptilde;
time s := SmallRoots (f, N, 2^460 : Beta:=beta);  // beta explicit
----> Time: 10.750   // closer to 512, the longer it takes

p eq (ptilde - s[1]);
----> true
```

Other applications in cryptanalysis

- Factoring $N = p^r q$ for large r
- Hidden Number Problem
- Attacks on digital signature schemes
- Attacks on Approximate Common Divisor problem
- Attacks on (Short) Principal Ideal Problem
- Attacks on NTRU and LWE cryptosystems (see next week)

Reading material

- Chapters 16 to 19 of Steven Galbraith's book: `https://www.math.auckland.ac.nz/~sgal018/crypto-book/crypto-book.html`
- Public key cryptanalysis by Phong Q. Nguyen:
  See Toledo
- A deterministic algorithm for finding r-power divisors by David Harvey and Markus Hittmeir:
  `https://arxiv.org/pdf/2202.12401.pdf`