COMPSCI 4CR3 - Applied Cryptography

Jake Doliskani

McMaster University

The RSA Cryptosystem

This lecture

- How RSA works
- Implementation aspects
- Finding Large Primes
- Security estimations
- Attacks and Countermeasures

RSA

- Whitfield Diffie and Martin Hellman introduced public-key cryptography in 1976.
- That opened the door to a whole new world
- People were looking for one-way functions to construct encryption algorithms
- Ron Rivest, Adi Shamir, and Leonard Adleman introduced RSA
- RSA became the most widely used public-key scheme
  - Elliptic curve schemes are now very popular

- The underlying one-way function of RSA is the integer factorization problem
  - Multiplying large integers is easy, but factoring large integers is hard

Key generation

1. Choose two large primes p, q
2. Compute $n = pq$ and $\varphi(n) = (p-1)(q-1)$
3. Select the public exponent $e \in \{0, 1, \ldots, \varphi(n) - 1\}$ such that
$$\gcd(e, \varphi(n)) = 1$$
4. Compute the private key d such that
$$de \equiv 1 \bmod \varphi(n)$$
5. Output the private-key, public-key pair (d, e)

Technically, the public key is (n, e).

Encryption, Decryption

Input: public key (n, e) and plaintext x.

1. Compute $y = x^e \bmod n$
2. Return y

Input: Input: private key d and ciphertext y.

1. Compute $x = y^d \bmod n$
2. Return x

Requirements

1. It must be computationally infeasible to find the private key d given the public key (n, e)
2. We cannot encrypt more that $\log n$ bits at a time
3. Key generation, encryption and decryption must all be fast.
   - In particular, it should be easy to do the exponentiations $x^e \mod n$ and $y^d \mod n$.
4. For any n there should be many choices of private-key, public-key pairs to avoid a brute-force attack

Also, decryption must be an inverse to encryption.

Correctness

Since $de \equiv 1 \bmod \varphi(n)$, we can write $de = 1 + t\varphi(n)$ for some integer t.
Decrypting the encryption of the message x gives

$$x^{de} = x^{1+t\varphi(n)} = x \cdot \left(x^{\varphi(n)}\right)^t \bmod n$$

Case 1: $\gcd(x, n) = 1$
By Euler's theorem

$$x \cdot \left(x^{\varphi(n)}\right)^t = x \cdot 1 = x \bmod n$$

Case 2: $\gcd(x, n) \neq 1$
Either $x = rp$ or $x = sq$. Without loss of generality assume $x = rp$. Then $\gcd(x, q) = 1$.

Correctness

Since $\gcd(x, q) = 1$, by Euler's theorem

$$\left(x^{\varphi(q)}\right)^t = 1 \bmod n$$

Then

$$\left(x^{\varphi(n)}\right)^t = \left(x^{(p-1)(q-1)}\right)^t = \left(\left(x^{\varphi(q)}\right)^t\right)^{p-1} = 1^{p-1} = 1 \bmod q$$

Using the definition of mod

$$\left(x^{\varphi(n)}\right)^t = 1 + uq$$

Therefore,

$$x \cdot \left(x^{\varphi(n)}\right)^t = x + xuq = x + (rp)uq = x + nru = x \bmod n. \qquad \square$$

Example (Encryption)

Message $x = 23$

Choose $p = 5, q = 17$

Compute $n = pq = 85$

Compute $\varphi(n) = (5-1)(17-1) = 64$

Choose $e = 9$

Compute $d = e^{-1} = 57 \bmod 64$

$$\xleftarrow{k_{pub} = (85, 9)}$$

$y = x^e = 23^9 = 28 \bmod 85$

$$\xrightarrow{y = 28}$$

$y^d = 28^{57} = 23 = x \bmod 85$

Example (Real-world keys)

p = 1752856549304473987230726928641235118933273400514716068320146879544806902156364074329188084076410592792268445892688335551609364329603747241663639351450327697639763945257700773501577541740340093080725902819842854330734068058346204053921112096804050317760601524815555186248452593375011396024678890352385010551

q = 1481231530816976963619005562070886842182597340571762806595884459175363584089103785767659998197524249341360821145577640839019206934492205212938653531463690309941571488199872319565853431464710954998657956545796867488407790074184153912170682740913215285991790505745244332885485329889540955162203434359865535 13841

e = 65537

d = 8270731525447691848920109866644402718948734149154322332508707692899813470990643355111479610052241589496601986429801098787027659442602003825604044768617910973496906327674935707608456154582404795558375060092546876668540530907110569605529914345857987602410733253117549617399829356832995974448817733855321973694473146068818297976480329131111978023675740447922214376363219325964311263169685794964505137324979863850627257870812313710932613383601428513887294014431300143728844545341433991457504484283654431245445929839440582484542426639360440946492487232674228445238040817727413118986776620332964031792742945456669333073

Exponentiation

How fast can we compute $x^d \bmod n$?

Answer 1: **repeated multiplication**. Requires $d - 1$ multiplications.

Not good enough: x, d, and n are large.

Answer 2: **square and multiply**
Idea: to compute $5^6 \bmod 9$

1. compute 5^2 and $5^4 \bmod 9$
2. compute $5^2 \cdot 5^4 \bmod 9$

three multiplications instead of five

Square-and-Multiply

Let $d = d_k d_{k-1} \cdots d_1 d_0$ be the binary representation of d. Then

$$x^d = x^{d_k 2^k + d_{k-1} 2^{k-1} + \cdots + 2^1 d_1 + 2^0 d_0} = \left(x^{2^k}\right)^{d_k} \left(x^{2^{k-1}}\right)^{d_{k-1}} \cdots \left(x^{2^1}\right)^{d_1} \left(x^{2^0}\right)^{d_0}.$$

Input: x, d, n
Output: $x^d \bmod n$

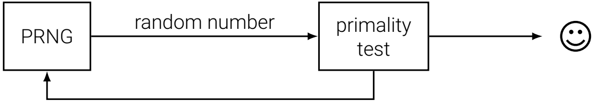
1. $y = x$
2. $z = 1$
3. Let $d = d_k d_{k-1} \cdots d_1 d_0$ be the binary representation of d
4. for $i = 0$ to k

   if $d_i = 1$ then $z = z \cdot y \bmod n$
   $y = y^2 \bmod n$

5. return z

$O(\log d)$ multiplications

How to find large primes



For this to be practical

1. We shouldn't have to test too many random numbers to find a prime
2. The primality test should be fast

Primes are common

The Prime Number Theorem

Let $\pi(x)$ = number of primes less that or equal to x. Then

$$\lim_{x\to\infty} \frac{\pi(x)}{x/\ln x} = 1.$$

- The probability of a random number number $\leq x$ being prime is $\approx 1/\ln(x)$.
- For a 512-bit prime number we need to test around

  $$\ln(2^{512}) \approx 355$$

  random numbers.

Primality testing

- Elementary tests
  e.g., try all primes $\leq \sqrt{n}$

- Probabilistic tests
  e.g., Fermat test, Miller–Rabin test

- Deterministic tests
  e.g., Agrawal-Kayal-Saxena test

Theorem

Given an odd integer p, write

$$p - 1 = 2^u r,$$

where r is odd. If we can find an integer a such that

$$a^r \neq 1 \bmod p \quad \text{and} \quad a^{r2^j} = 1 \bmod p$$

for all $j \in \{0, 1, \ldots, u-1\}$, then p is composite. Otherwise, p is probably prime.

Miller-Rabin primality test

Input: odd integer p
Output: "not prime" or "likely prime"

let $p - 1 = 2^u r$
for $i = 1$ to s do
    choose a random $a \in \{2, 3, \ldots, p-2\}$
    $z = a^r \bmod p$
    if $z \neq 1, -1 \bmod p$ then
        $j = 1$
        while $j \leq u - 1$ and $z \neq -1 \bmod p$ do
            $z = z^2 \bmod p$
            if $z = 1$ then return "not prime"
            $j = j + 1$
    if $z \neq -1 \bmod p$ then return "not prime"
return "likely prime"

- s is the number of trials
- The larger the s the more accurate the output
- The output is accurate with probability
  $$\approx 1 - 1/4^s$$
- Example: if p is 512-bit number, and we set $s = 30$, then the probability of error is less than $1/2^{60}$.

RSA in practice

- A scheme is **malleable** if the attacker can change the ciphertext into another ciphertext that corresponds to a known transformation of the plaintext.
- The plain RSA is malleable.
- Example: given a ciphertext y, we can replace it by $t^e y$ where t is some integer. Decryption gives
$$(t^e y)^d = t^{ed} x^{ed} = tx \pmod{n}.$$

- Solution: **padding**
- Example padding algorithm: Optimal Asymmetric Encryption Padding (OAEP)

Attacks

1. Protocol attacks
2. Mathematical attacks
3. Side-channel attacks

Protocol attacks:

- Exploit weaknesses in the protocols involving RSA
- There has been several such attacks
  - The malleability one we just saw
- To avoid these attacks, follow the guidelines of modern security standards

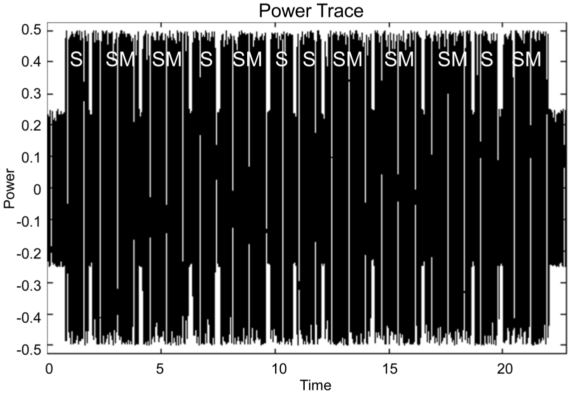
Mathematical Attacks

- The best known attack is factoring the modulus n.
- Factoring n reveals p and q, from which $\varphi(n)$ can be computed. The private key is then $d = e^{-1} \bmod \varphi(n)$.

Some recent RSA factoring records:

Modulus size	Factored on	Factored by
663 bits	2005-05-01	F. Bahr, M. Boehm, J. Franke, and T. Kleinjung
729 bits	2016-05-01	S. Bai, P. Gaudry, A. Kruppa, E. Thomé and P. Zimmermann
762 bits	2018-08-01	Samuel S. Gross.
829 bits	2020-02-01	F. Boudot, P. Gaudry, A. Guillevic, N. Heninger, E. Thomé and P. Zimmermann

Side-Channel Attacks (Example: timing attack)



Operations:	S	SM	SM	S	SM	S	S	SM	SM	SM	S	SM
Private key:	0	1	1	0	1	0	0	1	1	1	0	1