Notes on cryptography

Ariana

August 12, 2019

Contents

Part I

Mathematical preliminaries

Since computers aren't the best at storing arbitrary reals, we usually use integers, which is how **number theory** gets involved. Encryption/Hashing is basically a function that maps integers to integers, in a way that is hard to reverse.

Furthermore, having extremely large integers would make computing extremely long, so we use finite groups to avoid having overly large numbers, which involves some **abstract algebra**.

Nowadays, elliptic curves cryptography and variants are getting quite popular. The theory behind elliptic curves have only been formulated in the mid-1900, largely from the work of the mathematician Grothendieck, who effectively kick-started **algebraic geometry**. This topic is rather math heavy so try not to have high hopes on understanding the topic in a few years(it's usually taught in math grad school for reference).

As quantum computing is getting more powerful and more accurate nowadays, quantum-resistant cryptography are also getting more important. This new form of cryptography encompasses many different ideas, from classical cryptography(not using quantum properties) such as lattices to isogeny, to quantum cryptography that uses several interesting properties of quantum systems that classical systems cannot replicate.

0.1 Notation

S
$a\}$ - The set S without the element/set a

$\mathbb{N}$ - The set of natural numbers, including 0

$\mathbb{Z}$ - The set of integers

$\mathbb{Z}^+$ - The set of positive integers

$\mathbb{Z}^-$ - The set of negative integers

$\mathbb{Z}_{|}^{+}$ - The additive group modulo p

$\mathbb{Z}_{|}^{\times}$ - The multiplicative group modulo p

$\mathbb{P}$ - The set of prime numbers

$a|b$ - b is divisible by a, $\exists m^{\in\mathbb{Z}} b = ma$

$a \perp b$ - a and b are coprime, equivalently, $GCD(a, b) = 1$

Chapter 1

Number theory

Number theory is a branch of mathematics dedicated to studying integers and equations involving integer solutions.

1.1 Divisibility and primes

The notion of divisibility and remainder appears, often implicitly, everywhere in cryptography. This section introduces the basic notion of divisibility and remainder.

Theorem 1.1.1 *Division theorem*
Given some a, b, with $b > 0$, there is a unique solution to $a = qb + r$, where $0 \leq r < |b|$.

This is quite simple to proof, and the uniqueness is proven by contradiction.
The **greatest common divisor(GCD)** of a and b is the largest natural number that divides a and b. q is called the **quotient** and r is called the **remainder**.
A extremely simple algorithm, coming all the way from Euclid, is the **Euclidean algorithm**. The algorithm has a simple recursive definition:

$$r_{-2} = a \quad r_{-1} = b$$

$$r_{k-2} = q_k r_{k-1} + r_k \quad \text{from the division theorem}$$

This stops when $r_N = 0$, and r_{N-1} is the GCD.
A worked example for $a = 1113$ and $b = 812$ is given below(this case was specially chosen, the algorithm usually converges a lot faster):

k	eqn	q and r
-2		$r_{-2} = 1113$
-1		$r_{-1} = 812$
0	$1113 = 812q_0 + r_0$	$q_0 = 1, r_0 = 301$
1	$812 = 301q_1 + r_1$	$q_1 = 2, r_1 = 210$
2	$301 = 210q_2 + r_2$	$q_2 = 1, r_2 = 91$
3	$210 = 91q_3 + r_3$	$q_3 = 2, r_3 = 28$
4	$91 = 28q_4 + r_4$	$q_4 = 3, r_4 = 7$
5	$28 = 7q_5 + r_5$	$q_5 = 4, r_5 = 0$

Thus the GCD is 7.

By reversing this algorithm, we also get solutions to a linear diophantine equation.

Theorem 1.1.2 *Bézout's identity*
Given some a, b, there is exactly one solution to $xa + yb = \gcd(a, b) = d, |x| \leq |\frac{b}{d}|, |y| \leq |\frac{a}{d}|$. Other solutions to this equations are of the form $\left(x - k\frac{b}{d}, y + k\frac{a}{d}\right)$

The proof of existence is simple, by solving for x, y using the extended euclidean algorithm. (x, y) are usually referred to as the **Bézout's coefficients** Using $a = 1113$ and $b = 812$:

k	eqn	q and r
0	$1113 = 812q_0 + r_0$	$q_0 = 1, r_0 = 301$
1	$812 = 301q_1 + r_1$	$q_1 = 2, r_1 = 210$
2	$301 = 210q_2 + r_2$	$q_2 = 1, r_2 = 91$
3	$210 = 91q_3 + r_3$	$q_3 = 2, r_3 = 28$
4	$91 = 28q_4 + r_4$	$q_4 = 3, r_4 = 7$
5	$28 = 7q_5 + r_5$	$q_5 = 4, r_5 = 0$

Now we 'reverse' the algorithm by starting off with eqn 4, then continuously substitute in the previous equations.

$$\begin{aligned}
7 &= 91 - 28 \cdot 3 \\
7 &= 91 - (210 - 91 \cdot 2) \cdot 3 \\
7 &= 91 \cdot 7 - 210 \cdot 3 \\
7 &= (301 - 210 \cdot 1) \cdot 7 - 210 \cdot 3 \\
7 &= 301 \cdot 7 - 210 \cdot 10 \\
7 &= 301 \cdot 7 - (812 - 301 \cdot 2) \cdot 10 \\
7 &= (1113 - 812 \cdot 1) \cdot 27 - 812 \cdot 10 \\
7 &= 1113 \cdot 27 - 812 \cdot 37
\end{aligned}$$

which are the Bézout's coefficients$(27, -37)$

Another key aspect of number theory is the concept of **prime numbers**. A prime number is simply a number that is only divisible by 1 and itself. The first few primes are:

$$2, 3, 5, 7, 11, 13, \ldots$$

This may seem pretty simple but primes appear very often, especially in analytic number theory(if anyone wants to have some fun there).
Numbers that are not prime are called **composite numbers**. These numbers are a product of primes(if they aren't they will be primes itself). We call the representation of a number as a product of primes the **prime factorization**. This comes with quite a obvious but important theorem:

Theorem 1.1.3 *Prime factorization is unique*

A important fact about primes is that for any composite number, it's prime factorization is unique. This can easily be proven by contradiction and using the fact primes are unique.
Now for a general composite number, suppose we want to count the number of integers below and coprime to itself. This function is known as the Euler's totient function($\phi(n)$). It is pretty tedious to count one at a time, however, with the prime factorization, there's a formula that relies on prime factorization(Euler's product formula):

$$\phi(n) = n\prod_{p|n}\left(1-\frac{1}{p}\right)$$

The proof of this requires 2 easier to proof facts:

1. If $m \perp n$, then $\phi(mn) = \phi(m)\phi(n)$
2. For any prime p, $\phi(p^k) = p^k\left(1-\frac{1}{p}\right)$

The first part is proven by showing that if $a \perp m$ and $b \perp n$, then $an+bm \perp mn$, and that the other direction holds too.
The second part is done by finding all the numbers that aren't coprime to p^k.
Finally using the fact prime factorization is unique, the Euler's product formula is proven.

1.2 Modular arithmetic

The proofs in this section requires some group theory, so its best to just look through the basics, and revisit this section later on.
Usually, we are not so interested in the entire number, but just the remainder after being divided by some number. This is the idea of modular arithmetic, i.e. after every operation, we take the remainder.
For example in $\mod 5$:

$$1+2 \equiv 3$$

$$1-2 \equiv 4$$

$$4 \cdot 2 \equiv 3$$

$$\frac{3}{4} \equiv 2$$

Notice that division is also possible. However, this may not always be possible. The idea of modular division is given $\frac{a}{b}$, we want to find a number c such that $a \equiv bc$, similar to normal division. In fact, with Bézout's identity, this is rather simple to solve.

$$\frac{1}{b} \equiv c \pmod{n}$$
$$bc \equiv 1 \pmod{n}$$
$$cb + kn = 1$$

So when $\gcd(b, n) = 1$, this is always solvable, with (c, k) as the Bézout's coefficients. c is known as the **modular multiplicative inverse**
In modular arithmetic, modular exponents are quite interesting, with a lot of theorems associated with it, due to its structure as a abelian group. For example,

$$5^6 \equiv 5 \pmod{1}1$$

Now, if you were given $5^x \equiv 5 \pmod{1}1$, this becomes quite intimidating rather quickly, since our notion of the real numbered log is gone. This is known as the **discrete log problem**. Fortunately there are some theorems that can assist in solving this problem.

Theorem 1.2.1 *Wilson's theorem*

$$(n-1)! \equiv -1 \pmod{n} \Leftrightarrow n \in \mathbb{P}$$

The forward implication is quite simple, if $n \notin \mathbb{P}$, there exists a integer a such that $a < n$ and $a|n$, so $\frac{n}{a}$ is an integer and $a \cdot \frac{n}{a} \equiv 0 \pmod{n}$.
The backwards implication is slightly trickier.
When $n = 2$, the result is trivial, so only odd primes are considered.
For every $a < n$, a unique multiplicative inverse, a^{-1}, exists.
If $a \equiv a^{-1} \pmod{n}$, then $a^2 \equiv 1 \pmod{n}$, and $(a+1)(a-1) \equiv 0 \pmod{n}$. Since $a < n$ and n is prime, the only way this is possible is when $a = 1$ or $a = n - 1$, thus all numbers, between 2 and $n - 2$(inclusive) have a different unique multiplicative inverse.
The factors of $(p-2)!$ all results in $1 \pmod{n}$. Multiplying this by $(p-1)$, we get $(p-1)! \equiv -1 \pmod{p}$. Therefore the backwards implication is proven.

Theorem 1.2.2 *Fermat little theorem*
If p is a prime number, then$a^p \equiv a \pmod{p}$. Alternatively, $a^{p-1} \equiv 1 \pmod{p}$ if $a \perp p$

The idea behind this is that there are $p-1$ elements in $\mathbb{Z}_p^\times$, since p is prime, every integer except 0 satisfies the group axioms. A simple proof involves considering the sequence

$$a, 2a, 3a, \ldots, (p-1)a$$

This sequence is simply a rearrangement of

$$1, 2, 3, \ldots, p-1$$

if $a \not\equiv 0 \pmod{p}$(simple proof by contradiction)
Now we consider the product of both sequences

$$a^{p-1}(p-1)! \equiv (p-1)! \pmod{p}$$

By Wilson's theorem,

$$-a^{p-1} \equiv -1 \pmod{p}$$

$$a^{p-1} \equiv 1 \pmod{p}$$

And the case where $a \equiv 0 \pmod{p}$ is trivial.
Notice this only works for primes. There exists a more general theorem that works for all integers:

Theorem 1.2.3 *Euler's theorem*
If $n \perp a$, then

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

Here $\phi(n)$ is the Euler's totient function.
//proof
//Carmichael function/theorem

Chapter 2

Abstract algebra

Abstract algebra is a branch of mathematics that focus on the structure of objects. Such objects include groups, fields, algebras, and many more.
Recommended readings:

- I. N. Herstein - Topics in algebra
- N. Jacobson - Basic algebra

2.1 Group

A group is a set G, along with a operator $\cdot$ defined by 4 axioms:

- If $a, b \in G$, then $a \cdot b \in G$
- $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
- There exists an element $e \in G$ such that $e \cdot a = a = a \cdot e$
- For every $a \in G$, there exists an element a^{-1} such that $a \cdot a^{-1} = e$

A few properties come directly from this axioms:

Corollary 2.1.0.1 *e is unique*

Corollary 2.1.0.2

$$a \cdot a^{-1} = a^{-1} \cdot a$$

Note that it isn't necessary that $a \cdot b = b \cdot a$

If $a \cdot b = b \cdot a$ for all $a, b \in G$, this is known as a **abelian group**
The **order** of a group is simply the number of elements in it. We denote the order of a group G as $o(G)$, alternatively using a similar notation to the cardinality of a set, $|G|$

2.1.1 Subgroups and cosets

A **subgroup** is simply a subset of the group that is a group, for example, the the even integers is a subgroup of the integers under addition.
Suppose we have a subgroup H of G, if we select an element $a \notin H$, we define the sets:

$$aH = \{ah|h \in H\}$$

$$Ha = \{ha|h \in H\}$$

as the **left and right cosets** respectively. 'coset' will usually refer to the right coset

Corollary 2.1.0.3 $|aH| = |Ha| = |H|$

Corollary 2.1.0.4 *(Left) cosets are either identical or completely disjoint*

Now we can proceed to proceed to proof one of the most important theorems in group theory, Lagrange theorem.

Theorem 2.1.1 *Let H be a subgroup of G, then $o(H)|o(G)$*

Suppose there are k distinct cosets of H.
Since e is in H, the union of all the cosets must be G.
All cosets are also either distinct or identical, so the union of all k distinct cosets must be G.
Since the cosets are distinct, the size is simply the sum of the individual cosets, therefore

$$ko(H) = o(G)$$

2.2 Lattices(groups)

//what are lattices even
//integer lattice
//lll
//using lll in weird ways

Part II

Theoretical concepts

//we dont really care for actual attacks tbh...

Part III

Block ciphers

Chapter 3

Schemes

3.1 S and P boxes

3.2 Feistel network

used everywhere

3.3 Lai Massey network

Chapter 4

Hashes

hashes are kinda blockciphery

Chapter 5

DES

Chapter 6

AES

Chapter 7

Alternate block ciphers

7.1 ChaCha

used by google apparently?

Part IV

Classical public-key cryptosystems

Chapter 8

Old ciphers

//just describe general techniques going through all is quite rarted

Chapter 9

DH

Chapter 10

RSA

10.1 Introduction

RSA is a private key cryptography algorithm that relies on prime factorization being hard.
The algorithm:

1. Generate 2 distinct primes, p and q
2. Let $n = pq$
3. Compute either $o = \phi(n)$ or $\lambda(n)$
4. Choose $1 < e < o$, such that $\gcd(e, o) = 0$
5. Calculate d such that $ed = 1 \pmod{o}$
6. Public: n, e
   Private: p, q, o, d
7. To encrypt m, calculate $c = m^e \mod n$
8. To decrypt c, calculate $m = c^d \mod n$

10.2 Variants

10.2.1 Speed up

Sometimes to speed up decryption, CRT is used to compute mod n.
Let $d_p = d \pmod{p-1}$, $d_q = d \pmod{q-1}$

$$m_p \equiv c^d \equiv c^{d_p} \pmod{p}$$

$$m_q \equiv c^d \equiv c^{d_q} \pmod{q}$$

Define h as $m = m_q + qh$(by CRT). Now consider $m \pmod p$.

$$m_q + qh \equiv m_p \pmod p$$
$$h \equiv q^{-1}(m_p - m_q) \pmod p$$

10.2.2 $e = 2$

10.2.3 Multiple primes

10.3 Attacks

There are quite a few of attacks on RSA if the system is improperly setup:

- Small n/Week p
- Small d
- Small $p - q$
- Partial d exposure
- Partial p exposure
- Partial m exposure
- Partial decryption oracle(LSB kind) oracle
- Padding oracle
- Constant n, m different e
- Constant e, m different n
- Constant e, related m, different n
- Timing attack
- Power trace
- Fault attack

10.3.1 Important papers

:

- https://crypto.stanford.edu/ dabo/papers/RSA-survey.pdf contains most of the attacks on RSA

10.3.2 Small n

To factor n, I suggest using yafu, Alpertron online factorization or looking up on factordb(may also work for larger primes)

10.3.3 Small d

//wiener atk

10.3.4 Small $p - q$

The basic attack for this is **Fermat factorization**.
Notice that

$$(a+b)(a-b) = a^2 - b^2$$

Thus if we can find a, b such that $a^2 - N = b^2$, we have factored N.

```
def fermat(n):
        a=ceil(sqrt(n))
        b2=a*a-n
        while !is_square(b2):
                a+=1
                b2=a*a-n
        b=sqrt(b2)
        return (a-b,a+b)
```

There are several other speedups to this method.
//https://eprint.iacr.org/2009/318.pdf

10.3.5 Partial d exposure

//coppersmith lll

10.3.6 Partial p exposure

//coppersmith lll

10.3.7 Partial m exposure

//normal modulo math
//coppersmith lll

10.3.8 Partial decryption oracle(LSB kind) oracle

//multiply by k^e and abuse modulo

10.3.9 Padding oracle

//multiply by k^e and abuse modulo
//Bleichenbacher

10.3.10 Constant n, m different e

Suppose we are given $c_1 = m^{e_1} \pmod{n}$ and $c_2 = m^{e_2} \pmod{n}$.
Using Bézout's identity, we compute a, b such that $ae_1 + be_2 = 1$, then we compute

$$c_1^a c_2^b = m^{ae_1} m^{be_2} = m \pmod{n}$$

10.3.11 Constant e, m different n

The idea of this comes from when m is small, we can easily take the eth root. For example, $c = 6369690780153, e = 3, n = 25160293800283$, we can take the cube root of c and get $m =$ 0x4869 =‘Hi’. If we were given multiple $m^e mod n_i$, we can ‘increase’ the modulus using chinese remainder theorem
Suppose we receive $c_i = m^e \pmod{n}_i$ for e such i, we can simply use CRT to find $c = m^e \pmod{\Pi}_i n_i$, and take the eth root of c, since $m < n$, so $m^e \lessapprox \Pi_i n_i$.

10.3.12 Constant e, related m, different n

//ok now have magic with Hastad or FR atks

10.3.13 Timing attack

//https://www.paulkocher.com/TimingAttacks.pdf
//http://crypto.stanford.edu/ dabo/papers/ssl-timing.pdf

10.3.14 Power trace

actually just 0 and 1

10.3.15 Fault attack

lol just f up one of the crt modulos ez

Chapter 11

ECC

algebraic geometry hell here we go

Part V

Quantum cryptography

Chapter 12

Basic theory of qubits

Qubits are like normal bits, but probabilistic and much harder to imagine. We represent qubits as probability density, as compared to classical probability.

12.1 States

We represent states with **ket vectors**. This notation will become clear once more quantum theory is introduced.
For a given state A, we call the state $|A\rangle$. Qubits have 2 states, 0 and 1, $|0\rangle$ and $|1\rangle$ respectively.
Classically we think of probability as a real positive number less than 1, however for qubits, it's easier to think in terms of **probability density**, which can be complex. The probability of an event occurring is given by the square of the probability density.
Basic properties of probability density:

1. A complex number with magnitude less than 1
2. The chance is given by the squared magnitude of the probability density
3. Squared sum is 1(square then sum)

A general qubit, $|\psi\rangle$ has some chance of being $|0\rangle$ and some chance of being $|1\rangle$, we represent this as

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$$

where $P(|0\rangle) = |\alpha|^2$, $P(|1\rangle) = |\beta|^2$
Now rewriting this into the more familiar vector notation, we can represent $|0\rangle$ as $\begin{pmatrix}1\\0\end{pmatrix}$ and $|1\rangle$ as $\begin{pmatrix}0\\1\end{pmatrix}$, so a arbitrary quantum state is written as

$$|\psi\rangle = \begin{pmatrix}\alpha\\\beta\end{pmatrix}$$

Now we introduce the dual of the ket notation, the **bra vector**. These forms a bra-c-ket when placed together. The bra is written as the opposite of a ket, $\langle 0|$, and it's defined as(using vectors)

$$\langle\psi| = |\psi\rangle^\dagger$$

where † represents conjugate transpose(complex conjugate and transpose)
So $\langle\psi| = \begin{pmatrix}\alpha^* & \beta^*\end{pmatrix}$, and when we place them together, we get

$$\langle\psi|\psi\rangle = \begin{pmatrix}\alpha^* & \beta^*\end{pmatrix}\begin{pmatrix}\alpha \\ \beta\end{pmatrix} = |\alpha|^2 + |\beta|^2 = 1$$

//projecting stuff with braket
//multiqubit state
//entanglement ayy
//chsh with the funky integration proof

Chapter 13

BB84

Chapter 14

E91

Part VI

Post-quantum crypto

fancy math