SE6003 Cryptography

Fred Ezerman

Adjunct Assistant Professor (SPMS - NTU)
email: fredezerman@ntu.edu.sg

Course Information and Introduction

Thursday, 15 August 2024

Instructor

1. I grew up in East Java and Bali, Indonesia.
2. Learned philosophy (BA) and mathematics (B.Sc) as an undergraduate at Ateneo de Manila, Philippines.
3. Continued training in mathematics: M.Sc. at Ateneo followed by Ph.D. at SPMS NTU.
4. Research experiences:
   - mostly at NTU.
   - post-doc fellowship at LiQ Universite Libre de Bruxelles and CQT at NUS.
   - regular visiting research stints at Technion Haifa and Zhengzhou Univ.
5. Co-founder of several cybersecurity companies
   - CEO of SANDHIGUNA: early 2022 to end of 2023
   - Currently serving as Director of PQStation

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Coverage

1. A general view of modern cryptography and applications.
2. Mathematical background, as needed.
3. Encryption Schemes and Definitions of Security
4. Symmetric Key Cryptography
   - Block Ciphers, focusing on AES, and Stream Ciphers
   - Hash Functions
   - Message Authentication Codes
5. Public Key Cryptography
   - How to Exchange Keys in Public
   - Digital Signatures
   - Key Management
   - Secret Sharing
6. If time allows: other topics *e.g.*, Zero-Knowledge Proof, Post-Quantum Cryptography.

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Resources

There is NO REQUIRED textbook for this course.
Some recommended references are

- Heiko Knospe: A Course in Cryptography, American Mathematical Society, 2019.
  A copy of lecture notes are available in
  https://github.com/cryptobook
- Douglas Stinson and Maura Paterson: Cryptography: Theory and Practice, CRC Press
- Paar, Pelzl, and Guneysu: Understanding Cryptography, 2nd ed. Springer.

Additional resources may be provided as and when it needed. These may include official standards, software libraries, and video demos.

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Assignments and Assessment

There are four components:

1. Individual hands-on exercises (20 marks): perform specified cryptographic operations using open source tools, *e.g.*, openssl, python crypto library.
   Deadline to be agreed on.
2. Exam 1: closed book, 50-minute, in Week 7 (25 marks).
3. Exam 2: closed book, 50-minute, in Week 13 (25 marks).
4. Group Project Report (30 marks): see next slides for further details.

Group Project Report

1. Each group consists of 2 to 3 students.
2. The project can be one of the following:
   - A survey paper of a general topic *e.g.*, history of a modern cipher, blockchain, cryptocurrency, multiparty computation, privacy-preserving technologies, PKI.
   - A survey paper of a technical topic *e.g*, a secret sharing scheme, a key exchange protocol, a digital signature scheme.
   - An implementation of a scheme in software or hardware.
3. The paper should be in pdf, between 6 and 12-page long, excluding bibliography.

   If implementation is chosen, the report can be a video, a source code, or a device demo.

Deadline: 23:59 SGT on 21 November 2024. No extension will be given as I have to submit final marks by 6 December 2024

Session Replacement

Since 31 October 2024 is a public holiday, we need to agree on a replacement session.
I propose the following Saturday from 09:00 to 12:00.
Let's decide together

Week 1

Agenda for Week 1:

1. Motivating ourselves by looking at cryptography protocols in real life.
2. A brief tour of the mathematical background.

Let's get started

Modern Cryptography

Literal meaning of *cryptology* is secret communication.
Modern cryptology

1. ... also aims to protect privacy and integrity.
2. ... complexity-based, using computational assumptions.
   - All participants are computationally bounded algorithms.
   - There are computational problems that can not be solved by bounded algorithms.
3. ... in its abstract model expresses
   objects as information bits,
   actions as (digital) communications.

Mathematics in Cryptology

- Cryptology forms a great example of the unreasonable effectiveness of mathematics.
- Most research in cryptology is initially driven by curiosity.
- The economic consequences are tremendous.
- Today we revisit **some mathematical tools** that are often deployed in protecting our digital communication.

Basic Principles

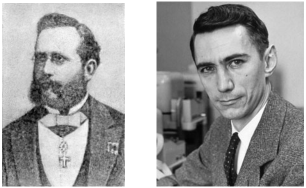


Figure: Kerckhoffs and Shannon

1. Kerckhoffs' Principle: "Desiderata de la Cryptographie Militaire", 1883

   It must not require secrecy and it can, without disadvantage, fall into the hands of the enemy.

2. Shannon Maxim: "A Mathematical Theory of Cryptography", Sept. 1945.

   The enemy knows the system

Six Goals

Paul C. van Oorschot, Computer Security and the Internet: Tools and Jewels, Springer 2020

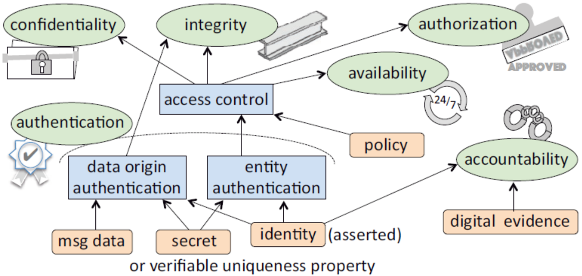


Figure 1.1: Six high-level computer security goals (properties delivered as a service). Icons denote end-goals. Important supporting mechanisms are shown in rectangles.

van Oorschot's Important Points

At the end of Chapter 1, van Oorschot distills the design principles into 24 points. Here go Points 3 and 9

P3 OPEN-DESIGN: Do not rely on secret designs, attacker ignorance, or *security by obscurity*. Invite and encourage open review and analysis. Example: undisclosed cryptographic algorithms are now widely discouraged—the *Advanced Encryption Standard* was selected from a set of public candidates by open review. Without contradicting this, leverage unpredictability where advantageous, as arbitrarily publicizing tactical defense details is rarely beneficial (there is no gain in advertising to thieves that you are on vacation, or posting house blueprints). Be reluctant to leak secret-dependent error messages or timing data, lest they be useful to attackers.

NOTE. This principle is related to *Kerckhoffs' principle*—a system's security should not rely upon the secrecy of its design details.

P9 TIME-TESTED-TOOLS: Rely wherever possible on time-tested, expert-built security tools including protocols, cryptographic primitives and toolkits, rather than designing and implementing your own. History shows that security design and implementation is difficult to get right even for experts; thus amateurs are heavily discouraged (*don't reinvent a weaker wheel*). Confidence increases with the length of time mechanisms and tools have survived (sometimes called *soak testing*).

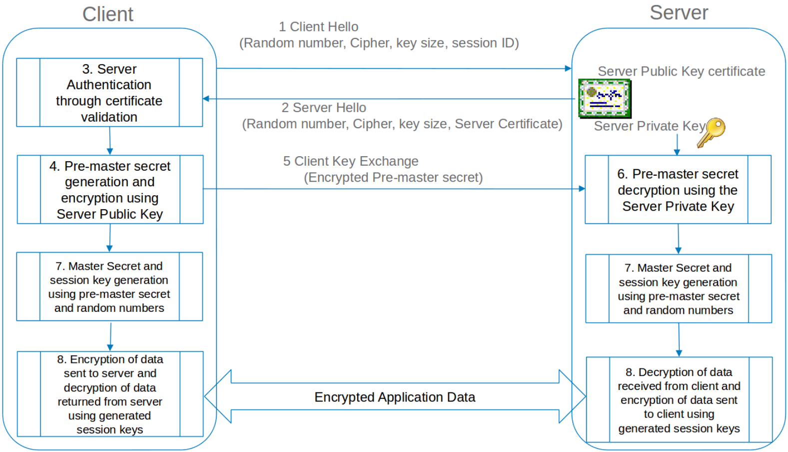
Examples of Cryptographic Tasks

Everyone has secrets or messages to specific recipients.

1. **Secret communication**: Creating a private language in public.
2. **Authentication**: The parties are who they claim to be.
3. **Integrity**: The content of the message is not changed.
4. **Nonrepudiation**: The party cannot deny their actions.
5. **Secret exchange** or **synchronization**: Alice learns Bob's secret if and only if Bob learns Alice's secret.
6. **Zero-knowledge proof** or **to convince but not reveal**: Convince others without revealing details.

Mathematics in Online Communication

A schematic picture of Transport Layer Security



Let's briefly inspect `https://www.ntu.edu.sg/`.

One Way Function with Trapdoor

Figures are mostly from Hoffstein, Pipher and Silverman, An Introduction to Mathematical Cryptography, Springer.

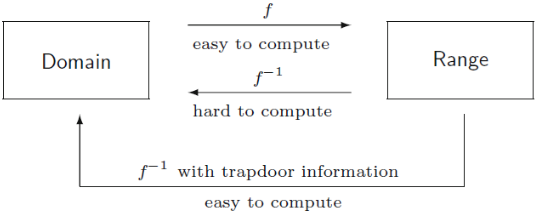


Figure 2.1: Illustration of a one-way trapdoor function

Symmetric and Asymmetric

By types of keys

- Symmetric if the keys to encrypt and decrypt are the same.
- Asymmetric (Public Key Crypto – PKC) if the keys are separated into public keys and private keys.

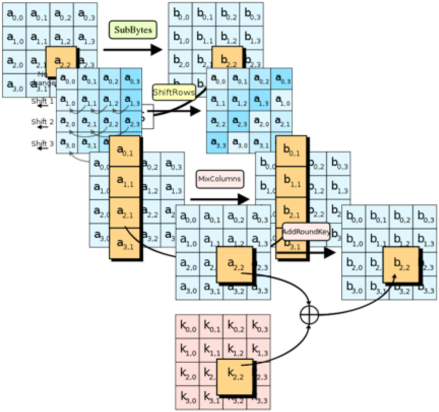
By processing mechanisms

- stream cipher: message (plaintext) is processed as a stream of bit strings.
- block cipher: message (plaintext) is processed by blocks of fixed lengths.

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Advanced Encryption System (AES)[1]

\# rounds & key lengths: $(10, 128)$, $(12, 192)$, $(14, 256)$.



[1] J. Daemen & V. Rijmen: The Design of Rijndael: The Advanced Encryption System (AES), 2nd ed., Springer, 2020.

Hash Functions

`https://passwordsgenerator.net/sha256-hash-generator/`

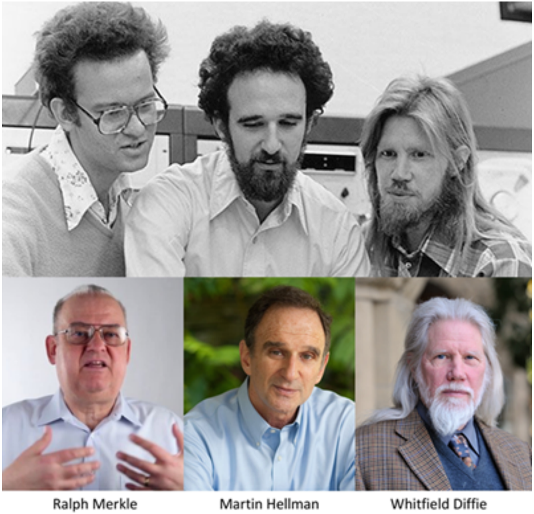
A hash function takes as input an arbitrarily long document D and returns a short, fixed length, bit string H:

1. Computation of hash(D) should be fast and easy.
2. Inversion of hash should be difficult: given a hash value H, it is hard to find D such that $\text{hash}(D) = H$.
3. Collision resistant: hard to find $D_1 \neq D_2$ such that

$$\text{hash}(D_1) = \text{hash}(D_2).$$

Commonly used algorithms include the `SHA-2` family.
Another protocol is `SHA-3` (originally named `Keccak`)
Hash functions are used, *e.g.*, in cryptocurrencies, many quantum-secure modules, and digital forensics.

Merkle, Diffie and Hellman



Diffie-Hellman: New Directions

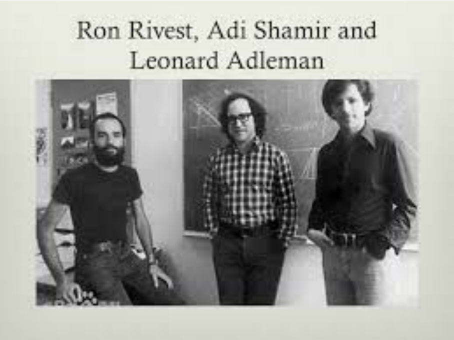
New directions in cryptography, IEEE Trans. Inform. Theory 22 (11) pp. 644–654, 1976.

Public parameter creation	
A trusted party chooses and publishes a (large) prime p and an integer g having large prime order in $\mathbb{F}_p^*$.	
Private computations	
Alice	**Bob**
Choose a secret integer a. Compute $A \equiv g^a \pmod{p}$.	Choose a secret integer b. Compute $B \equiv g^b \pmod{p}$.
Public exchange of values	
Alice sends A to Bob $\longrightarrow$	A
B $\longleftarrow$	Bob sends B to Alice
Further private computations	
Alice	**Bob**
Compute the number $B^a \pmod{p}$.	Compute the number $A^b \pmod{p}$.
The shared secret value is $B^a \equiv (g^b)^a \equiv g^{ab} \equiv (g^a)^b \equiv A^b \pmod{p}$.	

Table 2.2: Diffie–Hellman key exchange

Rivest, Shamir and Adleman

RSA: A method for obtaining digital signatures and public-key cryptosystems, Commun. ACM 21(2) pp. 120–126, 1978



Adi Shamir: How to share a secret, Commun. ACM 22 (11) pp. 612–613, 1979.

RSA Public Key Cryptosystem

Math tools from Number Theory. Hard problem: Factoring

Bob	**Alice**
Key creation	
Choose secret primes p and q. Choose encryption exponent e with $\gcd(e, (p-1)(q-1)) = 1$. Publish $N = pq$ and e.	
Encryption	
	Choose plaintext m. Use Bob's public key (N, e) to compute $c \equiv m^e \pmod{N}$. Send ciphertext c to Bob.
Decryption	
Compute d satisfying $ed \equiv 1 \pmod{(p-1)(q-1)}$. Compute $m' \equiv c^d \pmod{N}$. Then m' equals the plaintext m.	

Table 3.1: RSA key creation, encryption, and decryption

Computational Gap

Online calculator `http://magma.maths.usyd.edu.au/calc/`

```
p:=130796034785235721893734614731585906278З;
```

```
p:=1307960347852357218937346147315859062783;
p;
IsPrime(p);
q:=2833419889721787128217599;
q;
IsPrime(q);
time N:=p*q;
N;
time F:=PrimeDivisors(N);
F;
```

Clear Submit

```
1307960347852357218937346147315859062783
true
2833419889721787128217599
true
Time: 0.000
3706000864572296322968648767501458399706778297175696873426518017
Time: 4.450
[ 2833419889721787128217599, 1307960347852357218937346147315859062783 ]
```

(a) `time gap`

```
//p:=195845982777569926302400511;
```

```
//p:=195845982777569926302400511;
p:=1307960347852357218937346147315859062783;
p;
IsPrime(p);
//q:=2833419889721787128217599;
q:=4776913109852041418248056622882488319;
//q:=22525179859446666140991543177471319574581426704487890973300733139039351000268
q;
IsPrime(q);
time N:=p*q;
N;
time F:=PrimeDivisors(N);
F;
```

Clear Submit

The computation exceeded the time limit and so was terminated prematurely.

```
1307960347852357218937346147315859062783
true
4776913109852041418248056622882488319
true
Time: 0.000
6248012932822561585488935787002145600282202669364198280301935725382285131777
```

(b) `time overflow`

Figure: Computational gaps: multiplication & factoring.

RSA Signature

Note: Digital signing uses a *hash-then-encrypt* mechanism.
The document's fingerprint (output of hash fcn) is signed.
Current typical key length in RSA Signature is 2048.

Samantha	**Victor**
Key creation	
Choose secret primes p and q. Choose verification exponent e with $\gcd(e, (p-1)(q-1)) = 1$. Publish $N = pq$ and e.	
Signing	
Compute d satisfying $de \equiv 1 \pmod{(p-1)(q-1)}$. Sign document D by computing $S \equiv D^d \pmod{N}$.	
Verification	
	Compute $S^e \bmod N$ and verify that it is equal to D.

Table 4.1: RSA digital signatures

Relevant Computational Problems for RSA

1. How to decide if a large number is PRIME[2].
   - Miller-Rabin Algorithm:
     Fast but with a (very small) probability of error.
   - Agrawal-Kayal-Saxena (AKS) Algorithm:
     Much slower but always gives the accurate answer.
2. How secure is RSA? Survey by Dan Boneh
   Twenty Years of Attacks on the RSA Cryptosystem,
   Notices of the AMS, Feb. 1999, pp. 203 – 213.
3. Implementation issues in actual devices.
   Prime and Prejudice: Primality Testing Under Adversarial Conditions by Albrecht *et al.*
   `https://eprint.iacr.org/2018/749`

We try simple simulation by using an online tool:
`www.cryptool.org/en/cto/highlights/rsa-step-by-step`

[2] `https://prime-numbers.info/list/first-10000-primes`

Two Theorems on PRIMES

Theorem 1 (Fermat Little Theorem)

Let p be a prime number. Then

$$a^p \equiv a \pmod{p} \text{ for any integer } a.$$

Theorem 2 (Prime Number Theorem)

For $X \in \mathbb{N}$, let

$$\Pi(X) := \text{ the number of primes } p \text{ satisfying } 2 \leq p \leq X.$$

Then

$$\lim_{X \to \infty} \frac{\Pi(X)}{X/\ln(X)} = 1.$$

How Many PRIMES?

- Prime Number Theorem counts the number of primes of 2048 bit length to choose as p dan q in RSA-2048.
- Such a prime has 617 digits ($2^{2048} \approx 10^{616.5094}$)

Corollary 3

The number of 2048 *bit primes is* $\Pi(2^{2048}) - \Pi(2^{2047})$

$$\approx \frac{2^{2048}}{\ln(2^{2048})} - \frac{2^{2047}}{\ln(2^{2047})} \approx 2^{2036.528} > 10^{613}$$

- This is the number of keys to choose one pair from.
- We can all see that it will be stupid for the adversary to try to guess which pair p and q of primes are used as keys from such a large number of choices.

New and Emerging Topics

- Distributed Ledgers
- Privacy Preserving Cryptography e.g. Confidential Computing in the Cloud.
- Quantum-Secure Cryptography.
- Threshold Cryptology
- ...

Summary

Here are some important points to keep in mind.

- Modern cryptology lies in the intersection of physics, mathematics and computations.
- It exploits computational (time or memory) **gaps** in executing functions and their inverses.
- Knowing the theory does **not** always lead to correct and efficient implementation of the required protocols.
- Cryptology is a highly interdisciplinary endeavour.

We now continued to Knospe's slides to review some mathematical tools that we use to properly implement the above cryptographic primitives.

Cryptography

Fundamentals

Prof. Dr. Heiko Knospe

November 4, 2022

Mathematical Fundamentals

Modern cryptography relies on mathematical structures and methods.

We briefly discuss a number of fundamental topics from discrete mathematics, elementary number theory, computational complexity and probability theory.

Algebraic structures are covered in a separate chapter.

Sets

Sets are the most elementary mathematical structure. Finite sets play an important role in cryptography.

Example: $M = \{0, 1\}^{128}$ is the set of binary strings of length 128. Elements in M can be written in the form $b_1 b_2 \dots b_{128}$ or

$$(b_1, b_2, \dots, b_{128})$$

in vectorial notation. An element of M could, for example, represent one block of plaintext or ciphertext data. The cardinality of M is very large:

$$|M| = 2^{128} \approx 3.4 \cdot 10^{38}$$

Small and Large Numbers

It is important to help understand the difference between small, big and inaccessible numbers in practical computations. For example, one can easily store one terabyte (10^{12} bytes, i.e., around 2^{43} bits) of data. On the other hand, a large amount of resources are required to store one exabyte (one million terabytes) or 2^{63} bits and more than 2^{100} bits are out of reach.

The number of computing steps is also bounded: less than 2^{40} steps (say CPU clocks) are easily possible, 2^{60} operations require a lot of computing resources and take a significant amount of time, and more than 2^{100} operations are unfeasible. It is for example impossible to test 2^{128} different keys with conventional (non-quantum) computers.

Functions

Definition

A *function* or a *map*

$$f : X \to Y$$

consists of two sets (the *domain* X and the *codomain* Y) and a rule which assigns an output element (an *image*) $y = f(x) \in Y$ to each input element $x \in X$. The set of all $f(x)$ is a subset of Y called the *range* or *image* $im(f)$. Any $x \in X$ with $f(x) = y$ is called a *preimage* of y. Let $B \subset Y$, then we say that

$$f^{-1}(B) = \{x \in X \mid f(x) \in B\}$$

is the *preimage* or *inverse image* of B under f.

Injective, Surjective and Bijective Maps

Definition

Let $f : X \to Y$ be a function.

- f is *injective* if different elements of the domain map to different elements of the range: for all $x_1,\ x_2 \in X$ with $x_1 \neq x_2$, we have $f(x_1) \neq f(x_2)$. Equivalently, f is injective if for all $x_1,\ x_2 \in X$:

$$f(x_1) = f(x_2) \Rightarrow x_1 = x_2$$

- f is *surjective* or *onto* if every element of the codomain Y is contained in the image of f, i.e., for every $y \in Y$ there exists an $x \in X$ with $f(x) = y$. In other words, f is surjective if $im(f) = Y$.
- f is *bijective* if it is both injective and surjective. Bijective functions are invertible and possess an inverse map $f^{-1} : Y \to X$ such that $f^{-1} \circ f = id_X$ and $f \circ f^{-1} = id_Y$.

Relations

Definition

A relation R on X is a subset of $X \times X$. R is called an *equivalence relation* if it satisfies the following conditions:

1. R is reflexive, i.e., $(x, x) \in R$ for all $x \in X$, and
2. R is symmetric, i.e., if $(x, y) \in R$ then $(y, x) \in R$, and
3. R is transitive, i.e., if $(x, y) \in R$ and $(y, z) \in R$ then $(x, z) \in R$.

If $(x, y) \in R$, then x and y are called *equivalent* and we write $x \sim y$. For $x \in X$, the subset $\overline{x} = \{y \in X \mid x \sim y\} \subset X$ is called the *equivalence class* of x. The set of equivalence classes of X gives the *quotient set*

$$X/\sim .$$

Residue Classes modulo n

Let $n \in \mathbb{N}$, $n \geq 2$. Define the following equivalence relation R_n on $\mathbb{Z}$:

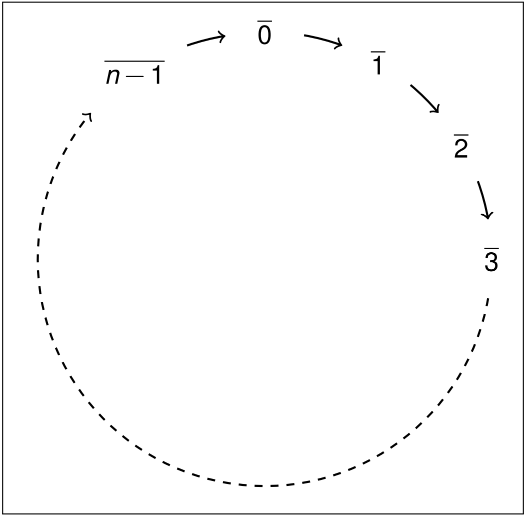
$$R_n = \{(x,y) \in \mathbb{Z} \times \mathbb{Z} \mid x - y \in n\mathbb{Z}\}$$

Note: $(x,y) \in R_n$ if the difference $x - y$ is divisible by n.
The equivalence class of $x \in \mathbb{Z}$ is the set

$$\overline{x} = \{\ldots, x-2n, x-n, x, x+n, x+2n, \ldots\}.$$

Now we have n *different* equivalence classes and the quotient set $\mathbb{Z}/\sim$ has n elements. We call this set the *residue classes modulo n* or *integers modulo n* and denote it by $\mathbb{Z}_n$ or $\mathbb{Z}/(n)$. Each residue class has a *standard representative* in the set $\{0, 1, \ldots, n-1\}$ and elements in the same residue class are called *congruent modulo n*.

Residue Classes modulo n



Example: $\mathbb{Z}_2$

$\mathbb{Z}_2 = \{\overline{0}, \overline{1}\}$ has only two elements. One has $\overline{-1} = \overline{1} = \overline{3}$ and $\overline{-2} = \overline{0} = \overline{2}$. The difference of two elements which are in the same class is divisible by 2 (i.e., their difference is even).

The standard representatives are 0, 1 and we have

$$\overline{0} = \{\dots, -4, -2, 0, 2, 4, \dots\},$$
$$\overline{1} = \{\dots, -3, -1, 1, 3, 5, \dots\}.$$

We may simple write 0 and 1 for these two classes.

XOR, AND, OR

Elements of $\mathbb{Z}_2$ can be added modulo 2, and addition is the same as the XOR operation on bits. The multiplication modulo 2 corresponds to the AND operation.

$\oplus$	0	1
0	0	1
1	1	0

$\cdot$	0	1
0	0	0
1	0	1

Table: *XOR and AND operations.*

Furthermore, there is an OR operation on bits and
$x \ OR \ y = x \oplus y \oplus x \cdot y$.

OR	0	1
0	0	1
1	1	1

Table: *OR operation*

Example: $\mathbb{Z}_{26}$

$\mathbb{Z}_{26} = \{\overline{0}, \overline{1}, \ldots, \overline{25}\}$ has 26 elements. For example, one has $\overline{-14} = \overline{38}$, since $-14 - 38 = -52$ is a multiple of 26. The integers -14 and -38 are congruent modulo 26 and we write

$$-14 \equiv 38 \mod 26.$$

The standard representative of this residue class is 12 and

$$\overline{12} = \{\ldots, -14, 12, 38, 64, \ldots\}.$$

Computations with Residue Classes

Residue classes can be added, subtracted and multiplied. An arbitrary integer representative can be used, and it is reasonable to choose a small representative.

Examples: a) $79 - 180 \mod 26 \equiv 1 - 24 \equiv 1 + 2 = 3 \mod 26$.
b) $234577 \cdot 2328374 \cdot 2837289374 \mod 3 \equiv 1 \cdot 2 \cdot 2 \equiv 1 \mod 3$.

However, division is more tricky since rational numbers $\frac{b}{a}$ are not representatives of residue classes. We say that a is invertible modulo n if there exists $x \in \mathbb{Z}$ such that

$$ax \equiv 1 \mod n.$$

Then $x \equiv (a \mod n)^{-1}$.

Example: $(3 \mod 10)^{-1} \equiv 7$, since $3 \cdot 7 \equiv 1 \mod 10$.

Invertible Residue Classes

Proposition

An integer a is invertible modulo n if and only if $\gcd(a, n) = 1$*, i.e., if the greatest common divisor of a and n is* 1*.*

Example: 3 is invertible modulo 10, but 2 is not invertible modulo 10.

Definition

The invertible integers modulo *n* are called units mod *n*. The subset of units of $\mathbb{Z}_n$ is denoted by $\mathbb{Z}_n^*$.

Example: $\mathbb{Z}_{10}^* = \{\overline{1}, \overline{3}, \overline{7}, \overline{9}\}$.

Prime Numbers

Definition

An integer $p \geq 2$ is called a prime number if p is only divisible by ± 1 and $\pm p$.

If p is prime, then

$$\mathbb{Z}_p^* = \{\overline{1}, \ldots, \overline{p-1}\}.$$

Prime numbers play an important role in public-key cryptography. The *Prime Number Theorem* states that the density of primes in the first N integers is approximately

$$\frac{1}{\ln(N)}.$$

Extended Euclidean Algorithm

One of the key algorithms in elementary number theory is the *Extended Euclidean Algorithm*. The algorithm takes two nonzero integers a, b as input and computes $\gcd(a, b)$ as well as two integers $x, y \in \mathbb{Z}$ such that

$$\gcd(a, b) = ax + by.$$

The Extended Euclidean Algorithm is very efficient and can be used to compute the multiplicative inverse of $a \mod n$. If $\gcd(a, n) = 1$ then the algorithm outputs $x, y \in \mathbb{Z}$ such that

$$1 = ax + ny.$$

Then

$$1 \equiv ax \mod n$$

and thus $x \equiv (a \mod n)^{-1}$.

Extended Euclidean Algorithm

Input: $a,\ b \in \mathbb{N}$
Output: $\gcd(a, b),\ x,\ y \in \mathbb{Z}$ such that $\gcd(a, b) = ax + by$
Initialisation: $x_0 = 1$, $x_1 = 0$, $y_0 = 0$, $y_1 = 1$, $sign = 1$

while $b \neq 0$ **do**
    $r = a \mod b$ // remainder of the integer division $a : b$
    $q = a/b$ // integer quotient
    $a = b$
    $b = r$
    $xx = x_1$
    $yy = y_1$
    $x_1 = q \cdot x_1 + x_0$
    $y_1 = q \cdot y_1 + y_0$
    $x_0 = xx$
    $y_0 = yy$
    $sign = -sign$
end while
$x = sign \cdot x0$
$y = -sign \cdot y0$
$gcd = a$
return $\gcd,\ x,\ y$

Modular Exponentiation I

Modular exponentiation with a large basis, exponent and modulus plays an important role in cryptography. How can we efficiently compute

$$x^a \mod n\,?$$

If $a = 2^k$ then k-fold squaring modulo n gives the result:

$$x^a \mod n = ((((x^2 \mod n)^2 \mod n)^2 \mod n)^2 \dots)^2 \mod n$$

For example, $x^{256} \mod n$ can be computed with only 8 squaring operations. After each squaring, reduce mod n in order to reduce the size of the result.

Modular Exponentiation II

If the exponent is not a power of 2, then it can still be written as *a sum of powers* of 2. This gives a product of factors of type $x^{(2^k)} \mod n$, and each factor can be computed by k modular squarings. We call this the *Fast Exponentiation Algorithm*.

Example: Compute $6^{41} \mod 59$. We have $41 = 2^5 + 2^3 + 2^0$ and first compute the following sequence of squares:

$$
\begin{aligned}
6^2 &\equiv 36 \mod 59 \\
6^4 &\equiv 36^2 \equiv 57 \mod 59 \\
6^8 &\equiv 57^2 \equiv 4 \mod 59 \\
6^{16} &\equiv 4^2 \equiv 16 \mod 59 \\
6^{32} &\equiv 16^2 \equiv 20 \mod 59
\end{aligned}
$$

Then $6^{41} = 6^{32} \cdot 6^8 \cdot 6 \equiv 20 \cdot 4 \cdot 6 \equiv 8 \mod 59$.

Cardinality

Proposition

Let X and Y be finite sets of cardinality $|X|$ and $|Y|$, respectively. Then:

1. $|X \times Y| = |X| \cdot |Y|$ *and* $|X^k| = |X|^k$ *for* $k \in \mathbb{N}$.
2. *Suppose* $|X| = n$ *and* $k \leq n$. *Then the number of subsets of X of cardinality k is given by the binomial coefficient* $\binom{n}{k} = \frac{n!}{k!(n-k)!}$.

Example: There are $\binom{128}{2} = \frac{128 \cdot 127}{2} = 8128$ different binary words of length 128 with exactly two ones and 126 zeros.

Euler's φ-Function

Definition

Let $n \in \mathbb{N}$. Then Euler's φ-function is defined by the cardinality of the units mod n, i.e.,

$$\varphi(n) = |\mathbb{Z}_n^*|$$

Examples: a) $\varphi(10) = 4$.
b) If p is a prime number, then $\varphi(p) = p - 1$.
c) If p and q are different prime numbers, then $\varphi(pq) = (p-1)(q-1)$ (Why?).

Permutations

Definition

Let S be a finite set. A *permutation* of S is a bijective map $\sigma : S \to S$.

Proposition

Let S be a finite set and $|S| = n$. Then there are $n!$ permutations of S.

Note: the factorial increases very fast, for example

$$50! \approx 3.04 \cdot 10^{64}.$$

Permutations in Cryptography

Cryptographic operations often use permutations. A randomly chosen family of permutations of a set like $M = \{0,1\}^{128}$ would constitute an ideal block cipher. However, it is impossible to write down or store a general permutation since M has 2^{128} elements. Much simpler (and much less secure) are *bit permutations*, which permute only the *position* of the bits.

Example: $(5\ 7\ 1\ 2\ 8\ 6\ 3\ 4)$ defines a permutation on $X = \{0,1\}^8$: a byte $(b_1, b_2, \ldots, b_8)$ is mapped to $(b_5, b_7, b_1, b_2, b_8, b_6, b_3, b_4)$. There are 8! bit permutations of X (a small number), but $(2^8)!$ general permutations (a very large number).

Big-O Notation

We often need to analyze the computational complexity of algorithms, i.e., the resources (running time and space) as a function of the input size.

Definition

Let $f, g : \mathbb{N} \to \mathbb{R}$ be two functions on $\mathbb{N}$. Then we say that g is an *asymptotic upper bound* for f, if there exists a real number $C \in \mathbb{R}$ and an integer $n_0 \in \mathbb{N}$ such that

$$|f(n)| \leq C\,|g(n)| \text{ for all } n \geq n_0.$$

One writes $f = O(g)$ or $f \in O(g)$.

Asymptotic Complexity: Examples

1. $f(n) = 2n^3 + n^2 + 7n + 2$. Since $n^2 \leq n^3$, $n \leq n^3$ and $1 \leq n^3$ for $n \geq 1$, one has $f(n) \leq (2+1+7+2)n^3$. Set $C = 12$ and $n_0 = 1$. Thus $f = O(n^3)$ and so f has cubic growth in n.
2. $f(n) = 100 + \frac{20}{n+1}$. Set $C = 101$ and $n_0 = 19$. Since $\frac{20}{n+1} \leq 1$ for $n \geq 19$, we have $f = O(1)$. Hence f is asymptotically bounded by a constant.
3. $f(n) = 5\sqrt{2^{n+3} + n^2 - 2n}$. Then $f = O(2^{n/2})$, and so f grows exponentially in n.

Complexity of Algorithms

Definition

If the running time of an algorithm is $f(n)$, where f is a *polynomial* and n is the input *size*, then the algorithm has *polynomial running time* and belongs to the complexity class **P**.

Polynomial-time algorithms are usually regarded as *efficient*.

In computer science, one is usually interested in the *worst-case* complexity of algorithms. However, when looking at the complexity of attacks against cryptographic schemes, their *average-case* complexity is much more important.

Complexity of Algorithms: Examples

- The functions in the above examples (1) and (2) are polynomial.
- The running time of the Extended Euclidean Algorithm on input $a, b \in \mathbb{N}$ is $O(\text{size}\,(a)\,\text{size}\,(b))$, so the algorithm is polynomial on the maximal input size.
- The running time of multiplying two numbers modulo n is $O(\text{size}\,(n)^2)$, which is polynomial.
- The running time of fast exponentiation modulo n is $O(\text{size}\,(n)^3)$, which is also polynomial.
- An algorithm which loops through $N = 2^n$ items has exponential running time in n.

Negligible Functions

We need the notion of a *negligible* function in the context of the probability of successful attacks.

Definition

Let $f : \mathbb{N} \to \mathbb{R}$ be a function. We say that f is *negligible* in n, if $f = O(\frac{1}{q(n)})$ for all polynomials q, or equivalently, if $f = O(\frac{1}{n^c})$ for all $c > 0$.

Negligible functions are eventually smaller than any inverse polynomial. This means that $f(n)$ approaches zero faster than any of the functions $\frac{1}{n}, \frac{1}{n^2}, \frac{1}{n^3}, \ldots$

Example: $f(n) = 10e^{-n}$ and $2^{-\sqrt{n}}$ are negligible in n.
$f(n) = \frac{1}{n^2+3n}$ is not negligible, since $f(n) = O(\frac{1}{n^2})$, but $f \neq O(\frac{1}{n^3})$.

Probability

We refer to textbooks on probability theory. We only consider *discrete probability spaces* and need the following notions:

- Probability space $(\Omega, \mathcal{S}, Pr)$, where Ω is a sample space, $\mathcal{S} = \mathcal{P}(\Omega)$ the set of events and $Pr : \mathcal{S} \to [0, 1]$ a probability distribution.
- Independent events A, B, i.e., $P(A \cap B) = P(A) \cdot P(B)$, and mutually independent events $A_1, \ldots, A_n$.
- The conditional probability $P[A|B] = \frac{P(A \cap B)}{P(B)}$ of events A, B.
- Random variables $X : \Omega \to \mathbb{R}$, their expectation $E[X]$ and variance $V[X]$.
- Probability mass function (pmf) $p_X(x) = Pr[X = x]$ of a random variable X.

Uniform Distribution and Random Bits

Definition

Pr has a uniform distribution if all elementary events have equal probability: $Pr[\{\omega\}] = \frac{1}{|\Omega|}$ for all $\omega \in \Omega$.

Random bits (or random numbers) are quite important in cryptography (but difficult to generate).

Definition

A random bit generator (RBG) outputs a sequence of bits such that the corresponding random variables $X_1, X_2, X_3, \ldots$ satisfy

1. $Pr[X_n = 0] = Pr[X_n = 1] = \frac{1}{2}$ for all $n \in \mathbb{N}$ (uniform distribution), and
2. $X_1, X_2, \ldots, X_n$ are mutually independent for all $n \in \mathbb{N}$.

Birthday Paradox

Let $x_1, x_2, \ldots, x_n$ be a sequence in a sample space Ω. We say that there is a *collision* if at least two elements in the sequence are identical.

Proposition

Let Pr be a uniform distribution on a set Ω of cardinality n. If we draw $k = \left\lceil \sqrt{2 \ln(2) n} \right\rceil \approx 1.2\sqrt{n}$ independent samples from Ω, then the probability of a collision is around 50%.

This fact is called *birthday paradox*: only $k = 23$ random birthdays ($n = 365$) are on average sufficient for a collision.

For $|\Omega| = 2^n$, around $2^{n/2}$ independent samples probably give a collision.