Cryptography
Winter 2010/2011

Priv.-Doz. Dr. Adrian Spalka

February 10, 2011

Organization

- Lecture: Thu, 12:15–13:45, 14:00–14:45, b-it bitmax
- Tutorial: Mon, 14:00–15:30, b-it bitmax

1. Basics of computer security
1.1 Preliminary considerations

Threats due to connection to the internet

- Outsourcing of internal processes,
- launch of e-commerce,
- increasing number of entry points,
- numerous external attacks.

Practical damage:

- Eavesdropping (theft) and misuse of information,
- modification of financial documents,
- execution of harmful transactions,
- violation of patents and copyrights,
- harm to the public image of a company.

$\Longrightarrow$ Without security electronic/mobile-commerce becomes unprofitable.

Applications with need for security

Traditional applications (closed environment):

- operating systems,
- data bases.

Modern applications (open environments $\Longrightarrow$ internet):

- mutual authentication,
- exchange and storage of documents and email,
- virtual private networks,
- trade-/bank-/stock-transactions on the internet (www-sessions): e/m-commerce,
- introduction of electronic medical records,
- single sign-on.

Risk analysis

Points of attack:

- the physical media: physical properties,
- the software: central subject of computer security,
- the user: *social engineering*.

Main problem: Design of the internet.

- Preferential: interoperability.
- Neglected: security.

Position of the attacker

1. Outsider: foreign party in the net.
   - Attacker eavesdrops/manipulates data traffic in the net.
2. Insider: corrupted user as involved party.
   - User misuses his rights on purpose.
3. Hacker: foreign party directly active on the machine of an involved party.
   - Attacker sneaks into the role of the user.
4. Malicious software (virus, worm, trojan, etc.): foreign party indirectly active on the machine of an involved party.
   - Program uses the rights of the user without his knowledge/consent.

Threat potential: the reality

Military sector (1996):

- more than 500.000 break-in attempts at the US DoD
  - 10% by *red teams*, thereof 65% successful and thereof 63% undetected
  - implications also for public infrastructure.

Public sector: poll among US-enterprises (2001)

- 62% successfully attacked, thereof 55% by an unauthorized insider,
- 51% suffered losses, but only 31% thereof could specify their loss,
- 57% of the attacks over the internet.

Targets and means of attack:

- denial of service: 32%, data sabotage: 14%, fraud: 14%,
- damages by malicious software: more than USD 15 billion (USA, 1999),
- more than 40.000 webpages with hacker tools,
- hacker tools today: automated, usable by non-experts.

1.2 A systematic approach

Goals of security

- Protection of the value that the data in an information system has.
- Is any data without value? BVerfG says “No”:
  - Sensitivity depends on the context.

Methods of security

1. *Security policy*: programmatic goal approach.
   - Informal determination of the value and the sensitivity of the data by declaration of a security level.
2. *Security strategy*: translation of the security policy into the context of an information system model.
   - Declarative specification in the security model.
   - Detailed administrative regulations.
3. *Security tactic*: operational realization.
   - Practical application of a security model in a computer/network.

Ways to evaluate the values in a security policy

- Mandatory: determines all values and their protection requirements bindingly.
- Discretionary: determines at who's discretion the determination of values and their protection requirements is.

Factors for evaluation

- *Authenticity*
  - Protection from wrong attribution of actions/data,
  - identification: determination of the identity of a user,
  - authentication: proof of the claimed identity of a user.

  Example: Nobody is allowed to sign on behalf of the boss.
- *Integrity*
  - Protection from unauthorized modification of data,
  - detection of unauthorized modification of data.

  Example: No employee is allowed to change his salary.

Factors for evaluation (continued)

- *Availability*:
  - Protection from delay of access to data,
  - providing means for legitimate users to perform operations according to the rules,
  - basis of any communication.

  Example: A system failure may not exceed 4 hours.
- *Confidentiality*:
  - Protection from disclosure of data to unauthorized people,
  - concealing and hiding the intended meaning of data from unauthorized people.

  Example: Intentions to buy may not be disclosed before the date of acquisition.

Determination of values: examples

Area	*Values*	*Factors*
railroad traffic	collision freeness of trains	integrity
banks	protection from fraud	integrity
airport tower	life of passengers	availability/integrity
military	success of operations	confidentiality
medicine	health of patients	integrity/ availability/ confidentiality

Classification of protection measures for conservation of values

1. *Prevention*: should avoid loss of value.
   Example: access control and cryptography.
2. *Recovery*: should recover value afterwards.
   Example: records and protocols.
3. *Limitation of damage*: should limit the loss.
   Example: burglar alarms and intrusion detection systems.

Specific countermeasures: depending on the position of the attacker.

Securing the accessibility of resources

in case of *overload/monopolization* $\Longrightarrow$ restriction of range of use

- static/dynamic (workload dependent)/priority-based assignment of resources on *individual basis*
  $\Longrightarrow$ only applicable, if user identification possible $\Longrightarrow$ not applicable with anonymous multi-user access (internet!)

in case of *partial breakdown* $\Longrightarrow$ redundancy

- synchronized: RAIDs, multipoint-to-multipoint connections in the net, tandem processors, etc.
- time-delayed: periodic archiving, spare parts, etc.

in case of *total breakdown* $\Longrightarrow$ external physical measures

- protection against power blackout, fire, theft, etc.

a possible strategy for web-server: *request monitoring/session monitoring*

1. limit maximal number of requests/sessions
2. monitor number of requests/sessions continuously
   - in case of overload: deny requests/sessions with feedback
     $\Longrightarrow$ ongoing sessions continue unhampered
3. limit maximal increase of requests/sessions
4. monitor increase of numbers of requests/sessions continously
   - in case of overload: ignore requests/sessions without feedback
     $\Longrightarrow$ ongoing sessions continue with delay
     $\Longrightarrow$ hones new users also receive no reply

- *hardware-integrity*: modifications of the machine
  rounds of inspection
- *compliance with access routes to data*: application-specific access
  access-control of operation system
- *sanctity of data paths*: modification of data stream
  cryptographic protocols
- *user-/program-corruption*: misuse of rights
  restriction of competences, distribution of powers
- *semantic integrity*: invalid data integrity requirements (mainly in data bases)

securing confidentiality

general problem: precise definition of confidentiality

Question: Do you have an account . . .

- . . . at the Sparkasse? (*content* is confidential)
- . . . at the Castle Bank of Nassau? (*existence* is confidential)

Answers:

- Yes.
- Maybe./I don't tell./No comment.
- No.
- I don't understand the question.

Approaches to protect confidentiality:

- access can be controlled: *access restriction*
  operating systems and closed nets
- access can not be controlled: *encryption*
  open nets
- Users knows fragments of the environment:
  misinformation/uncertainty
  (statistical) databases

- *identification*: specification of an individual identification
  Examples: user-/role-/node-name/id
- *authentication*: confirmation of correctness of identification by:
  - personal/system-inherent *knowledge*: e.g. password
    advantages: arbitrarily reusable, indestructible
    disadvantages: forgettable, spy-able (written down)
  - personal *possession*: e.g. smartcard
    advantages: arbitrarily reusable, not spy-able
    disadvantages: broken, duplicated, forgotten, stolen
  - personal *characteristic* (biometry): e.g. finger print, signature, etc.
    advantage: "unforgettable"
    disadvantage: not accepted, unreliable, theft/replication of biometric data, irrevocable, legal concerns

allocation of rights

basis of allocation of rights:
trustworthiness of the user or confidentiality/integrity/accessibility
$\Longrightarrow$ *considered*: explicit assumptions on the *user's behavior*
$\Longrightarrow$ *often not considered*: implicit assumptions on the *program's behavior*
Where has malware been considered?

assumption of modern concepts: resources have to be protected from users

- *problem*: the legislation requires protection of the user from certain data
  - Example: content on the internet which is unconstitutional/inappropriate for minors
  - solutions for schools with internet access

preemptive danger elimination

- *verification*: verify the trustworthiness of programs, i.e. that a program:
  - does not contain undocumented functionality (security)
  - does have the assured functionality (safety)
- *global classification* of programs as
  - trustworthy: part of the *trusted computing base* (TCB):
    - Is demonstrably trustworthy, i.e. verifiable.
    - Can not be modified by an unauthorized party.
    - Can not be by-passed.
    - Controls all access attempts and decides on their trustworthiness.
  - possibly un-trustworthy: all the rest.
- *GAC*: design of a new access control model!
- Restriction of distribution of rights of all kind.
- Structuring of group environments.

Recovery

- traceability of actions
- attribution to initiator and executor
  logging and subsequent evaluation
  BUT: legal situation unclear.

damage limitation

1. *surveillance* of running activities
2. *alerting* on supposed breaches, i.e. on admissible but abnormal activity patterns

   intrusion detectors

   $\Longrightarrow$ privacy

2. Cryptographic systems
2.1 Introduction

Tasks of cryptography

- realization of trustworthy computer-based communication- and information-systems, which use a vulnerable medium or storage
- reliable transactions in public networks between *careful* parties

Players in a cryptographic system

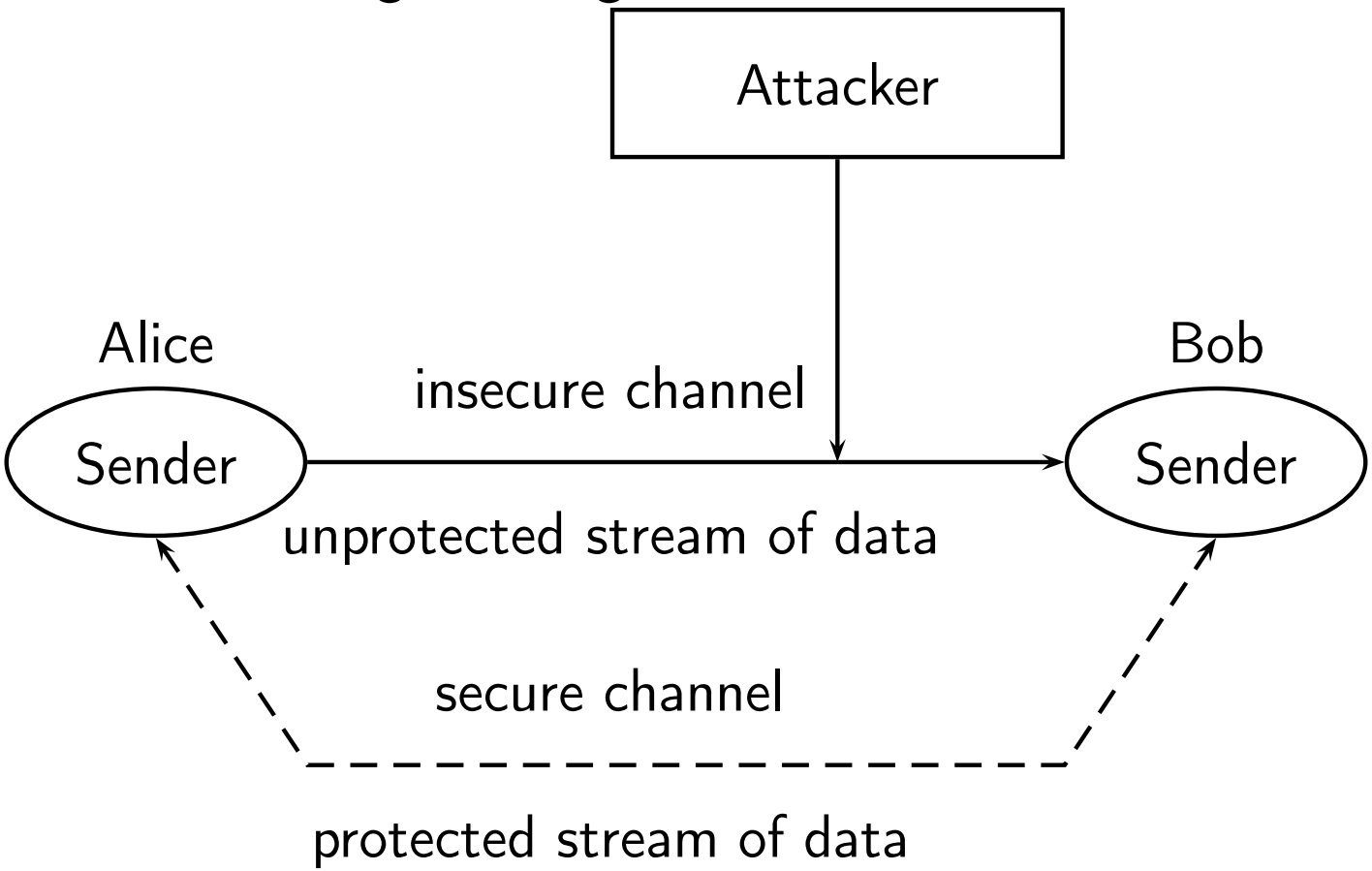
- desired parties: two or more senders and receivers (Alice and Bob)
- undesired parties:
  - eavesdropper (Eve)
  - malicious attacker (*Mallory*)

Assumptions on the desired parties

- Alice and Bob may cheat on each other, i.e. there is mutual mistrust
- The computations performed by and the messages sent by Alice and Bob are intended as such, i.e. the terminal nodes are trustworthy.

The situation

Alice and Bob exchange messages over an insecure channel.



The problems

- Attacker has unauthorized access to the data on the insecure channel. He may:
  - eavesdrop (confidentiality): hide meaning (encryption)
  - modify (integrity): detect manipulations (digital signatures)
  - discard (accessibility): bad luck
- Alice and Bob cheat on each other: detect fraud (protocols)

1. Alice uses an encryption function enc,
   which maps a message (string) m from the message space $\mathcal{M}$
   to a message (string) c from the ciphertext space $\mathcal{C}$:

$$\mathrm{enc}\colon \mathcal{M} \to \mathcal{C}, m \mapsto c.$$

2. Alice computes $c = \mathrm{enc}(m)$ and sends c to Bob.
3. Bob uses a decryption function dec with the property that $\mathrm{dec}(c) = m$ for all $c = \mathrm{enc}(m)$, i.e.

$$\mathrm{dec} : \mathcal{C} \to \mathcal{M}, \mathrm{dec}\,|_{\mathrm{enc}(\mathcal{M})} = \mathrm{enc}^{-1}\,.$$

   In most cases, $\mathcal{M}, \mathcal{C} \subseteq \Sigma^*$, where Σ is a finite set of symbols (alphabet).
   Examples: $\Sigma = \{0, 1\}$ or $\Sigma = \{A, \ldots, Z\}$

basic properties of enc and dec

- determining m from c is practically infeasible.
- enc and dec are fast and easy to use.

- Keeping enc and dec secret.
  Problem: Many people involved until maturity.
- Kerckhoff's Principle: enc and dec are public.
  Consequence: extension of enc and dec by a key parameter:

$$\mathrm{enc}(m, k_1, \ldots, k_n) \text{ and } \mathrm{dec}(c, \tilde{k}_1, \ldots, \tilde{k}_\ell)$$

  and secrecy of the keys.

Scenarios of cryptanalysis

Known are:

- only c
- some pairs (m, c)
- also enc

Relevant for the cryptanalysis

- If the message m is redundant with respect to the transmitted information, then for the recovery of m from c an incomplete/erroneous reconstruction may be sufficient.
  For natural languages this is always the case.

2.2 Encryption schemes

2.2.1 steganography

The art of "concealed writing". Hiding the existence of secrets.
plays on words: initial letters of words, parts of the text, etc.

- Examples (Richelieu):
  - encryption: message-text + *garbage-in-between*-text according to a template
  - decryption: template on the ciphertext

  laborious encryption.

Tampering of graphic files: BMP/GIF/JPG-format, etc.
Example

- size: 2048 x 3072 pixels
- 24 bit RGB-data per pixel
- The least significant bit of the RGB-data can (mostly) be manipulated without effect on the image quality.
- This enables the hiding of 2.3 MB of data in one image.

goal: classical protection of confidentiality from eavesdropping.
method: common secret key k for sender and receiver.

- encryption: $\mathrm{enc}(m, k) = c$
- decryption: $\mathrm{dec}(c, k) = m$

demand: decryption of c is practically only possible with k.

2.2.2 symmetric encryption (continued)

disadvantages:

- large number of keys and therefore expensive key-management
- useless for authenticity

advantages:

- confidentiality in closed (maybe large) user base
- easily implementable in hardware and fast (100 MB/s and more)

symmetric block ciphers

- approach: create confusion, e.g. DES (56 bit key, old) and AES (128 to 256 bit key, current)
- computation: primitive operations, e.g. bit-rotation and bit-permutation

symmetric stream ciphers

- encryption of single units (bits/bytes)
- enc can modify itself (enc may have states)
- less complex than block ciphers and therefore faster
- adequate/necessary if no buffer is available/permitted
- little/no failure propagation (good for noisy channels)
- only few public algorithms
  - SEAL (software-optimized encryption algorithm, 1993): customized for 32-bit processors
  - RC4: not public

general course of action

1. common for Alice and Bob:
   - random number generator Z
   - secret key k as start value for Z
2. encryption of $m = m_1 \dots m_n$
   - Alice starts Z with k
   - bits of m are added modulo 2 to the bits z_i output by Z, i.e. XOR

$$c_i = m_i + z_i \mod 2$$

3. decryption of $c = c_1 c_2 \dots$
   - Bob starts with Z with k

$$m_i = c_i + z_i \mod 2.$$

2.2.3 Asymmetric cryptography (public key)

Approach: difficult mathematical problems (one-way trapdoor functions):

- Prime factorization of natural numbers (RSA),
- computation of discrete logarithms in finite groups (DL),
- computation of quadratic residues in finite groups (QR),
- "division of points" on elliptic curves over finite fields (EC).

Computation: complex mathematical operations

2.2.3 Asymmetric cryptography (public key) (continued)

procedure: every participant P gets two keys pk and sk:

- public key pk for a public "telephone book"
- private key sk for his own vault

advantages:

- confidentiality, integrity and authenticity in open networks
- building block for communication protocols
- low number of keys

disadvantages:

- slow
- authenticity of public keys has to be guaranteed.

vulnerability:

- complexity is unknown, i.e. prone to *dramatic* developments.

Asymmetric basis procedure

Participants: $P_1 = (sk_1, pk_1)$ and $P_2 = (sk_2, pk_2)$

- telephone book $= (pk_1, pk_2)$

Confidentiality: P_1 sends secret message m to P_2

- encryption by P_1: $\mathrm{enc}(m, pk_2) = c$
- decryption by P_2: $\mathrm{dec}(c, sk_2) = m$

Requirement: In practice, c can only be deciphered with sk_2.

Integrity and authenticity: P_1 sends signed message m to P_2

- signing by P_1: $\text{sig-gen}(m, sk_1) = s$
- sent: (m, s) – received: (m', s')
- verification by P_2: $\text{sig-ver}(s', pk_1) = m''$

$$m'' = m' \implies m = m' \text{ and } s = s'$$

Requirement: In practice, s for m can only be generated with sk_1.

- agreement of session keys
- finding duplicate messages
- (un-)deniability of sending or receiving of messages
- general procedures:
  - contract signing
  - random decisions

reliable transactions in closed private networks:

- so far well solved with scs
  - hierarchy of keys
  - trustworthy special hardware
  - at least some trustworthy parties
  - users: banks, business, public administration, governments, military, etc.

division of tasks between scs and acs for encryption

- sks are very fast and for medium and large amounts of data indispensable
- acs are very slow and only for small amounts of data useful

$\Longrightarrow$ hybrid systems:

- scs encrypts the data
- acs encrypts the key (also: session key) of the scs

application of crypto systems to signatures (authenticity and integrity)

- infeasible with scs
- for free with acs

  $\Longrightarrow$ In practice, particularly efficient with EC-acs

3. Random numbers
3.1 Introduction

Fair coin-flipping protocol

Alice and Bob choose a one-way function f.

1. Alice chooses x, computes $f(x)$ and sends $f(x)$ to Bob.
2. Bob guesses: x even or odd and sends his guess to Alice.
3. Alice answers true or false to Bob.
4. Alice sends x to Bob.
5. Bob computes $f(x)$ and verifies that this value is equal to the value in 1.

Extension of the protocol and applications

- Alice chooses one bit at a time and generates a bit vector => Encoding of sets.
- Games that require randomness:
  - Card games,
  - board games.
- Consumers of random numbers:
  - Simulations,
  - games,
  - cryptography.

Phrasing the requirements

- Choose one element at random from a set of n numbers.
- Generate a random sequence of length m, which consists of elements from a finite set, e.g. a random bit string of length 1024.
- Clarifications:
  - All numbers and all sequences consisting of these numbers already exist - since a long time.
  - Not the *numbers* are random, but the *process* how they are chosen.
  - A random number is a number that is chosen at random.

3. Random numbers
3.1 Introduction
3.1.1 Approaches to random number generation

Hardware based

- These approaches make use of presumed randomness in physical processes.
- Examples:
  - Time in between emission of particles in radioactive decay,
  - thermal noise of semi conductors,
  - read access time of hard disks,
  - sound and image sources.

Software based

- These approaches make use of presumed randomness in events occuring during runtime of a computer.
- Examples:
  - System clock,
  - buffer content,
  - state or load of system resources.

General suggestions

- Use several sources as input.
- Deskewing (post processing) of the output in order to remove correlation (next bit) or trend (single bit).

3.2 Attempts to define randomness

Shannon (1948)

- "Father of Information Theory".
- Shannon analyses distributions, that are not ideally random.
- He defines randomness as extremum.
- A set is *perfectly random*, if its information content (entropy) is maximal.
  - It does not contain redundancy.
  - Its elements are distributed uniformly.
- Conclusion: It is not possible to generate random sequences from short random initial sequences (seeds).

Kolmogorov (1965)

- Computability theory.
- The complexity of sets is defined via the the shortest program that generates this set.

- A perfectly random set is an extremum.
- But:
  - Kolmogorov complexity is not computable.
  - It is impossible to generate sets with high Kolmogorov complexity from short random initial values (seeds).

Binary symmetric source - BSS

- Ideal source of randomness.
- **Definition 3.1**:
  - A BSS is a black box, which emits one bit at a time.
  - All outputs occur with the same probability.
  - For each output bit the probability that it is 1 or 0 is always $\frac{1}{2}$.
  - A given sequence of n bits is always generated with probability $\frac{1}{2^n}$.
  - This property of the BSS is called uniform distribution.

Problem: In reality - there exists no ideal source of randomness.

Blum/Goldwasser/Micali/Yao (1982/1984)

- **Assumption:** Applications that depend on sequences of random numbers, keep their properties to a large extend when sequences of pseudo random numbers are used.
- **Approach:** Two objects are called identical, if there exists no *efficient* algorithm that can distinguish them.
- **Result:** Randomness is no inherent property of a sequence, but perception of an observer relative to his computing power.

Definition 3.2

A sequence is pseudo random, if no algorithm running in polynomial time can distinguish this sequence from a uniform distributed sequence.

Definition 3.3

- Let $\ell : \mathbb{N} \to \mathbb{N}$, s.t. $\forall n : \ell(n) > n$.
- An efficient (deterministic) function f is a PRNG with streching function ℓ, if for every random n-bit input x,
  - the output $f(x)$ is of length $\ell(n)$ and
  - an efficient algorithm cannot distinguish $f(x)$ from a random bit sequence of of length $\ell(n)$.

Equivalent Definitions

- f is a PRNG if statistical tests running in polynomial time, cannot distinguish the output of f from a sequence of uniformly distributed bits, with probability more than $\frac{1}{2}$.
- *Next-bit test:* f is a PRNG, if there exists no algorithm running in polynomial time, which predicts the next output bit with probability higher than $\frac{1}{2}$.

Requirements for PRNGs

- **Benefit:** A PRNG generates efficient, long, pseudo random sequences from short random seeds.
- Requiremnts for PRNGs:

1. Statistic: Output should look random.
   - Theoretical definition.
   - Series of practical tests.
2. Security
   - Requirement form the BSI (AIS20, 1999): four security levels.
3. Efficiency
   - The random values should be constructed fast.
   - The application that depends on the randomness should run in a user tolerable time.

Security levels for PRNGs defined by the BSI

K1 Output should not repeat itself.

K2 Statistical properties of the output should be comparable to those of an ideal random source.

K3 An attacker observing the output should not be able to predict previous or future output or worse compute the inner state of the algorithm.

K4 An attacker knowing the inner state of the algorithm should not be able to predict previous states or outputs.

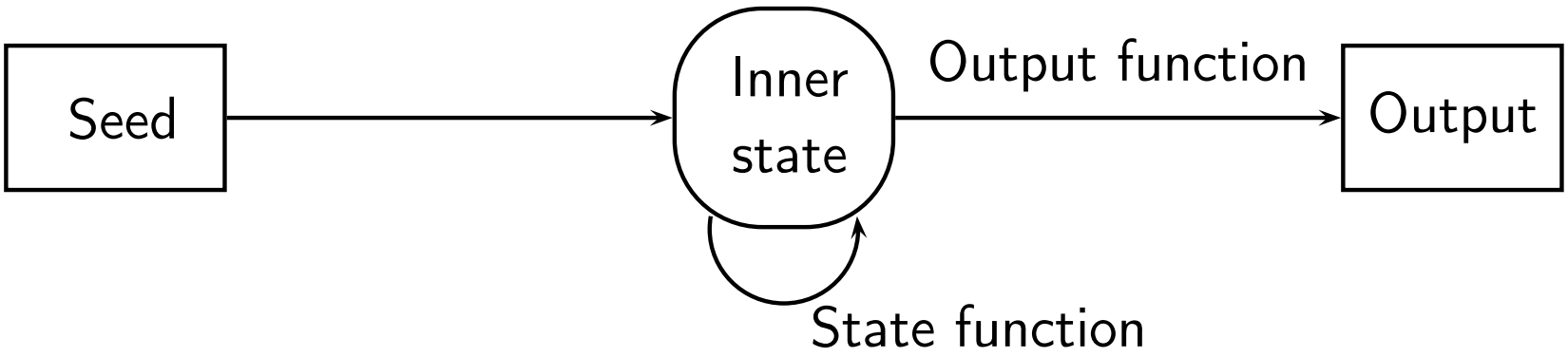
3.3 Constructions of PRNGs

Basic properties

- Statistic: Must pass practical tests (K2).
- Efficiency: Determined by practical tests and depends on the requirements of the application.
- Security: Theoretic discussions, that aim to achieve the one-way properties of K3 and K4.

- Cryptographic functions in use:
  - Hash-functions,
  - symmetric encryption schemes,
  - asymmetric encryption schemes.

Design of a PRNG



- State function generates a sequence of inner states: $I_0, I_1, \ldots$.
- Output function generates a sequence of output values: $A_0, A_1 \ldots$

Definition 3.4

Let f be a one-way function. h is a hard-core function of f, if

- h can be efficiently computed and
- for a random $x \in \{0,1\}^*$, $f(x)h(x)$ is a pseudo random distribution.

Definition 3.5

Let f be a one-way function. b is a hard-core predicate of f, if

- b can be efficiently computed and
- $b(x)$ cannot be efficiently predicted, if x is uniformly distributed, i.e.
- b preserves f's property to not be efficiently invertible.

A hard-core predicate is a special case of a hard-core function.

Definition 3.6

Let f be an efficient, length-preserving and bijective function. Let b be a hard-core predicate of f. Then

$$G(x) = b(x)b(f(x)) \dots b(f^{\ell(|x|)-1}(x))$$

is a PRNG with streching function ℓ.

Example 1 Blum-Blum PRNG

Let

- $N = pq$, p and q prime and $p, q \equiv 3 \mod 4$. [1]
- $f(x) = x^2 \mod N$, a one-way permutation.
- $lsb(x)$ the least significant bit of x.

Then

- $lsb(x)$ is a hard-core predicate of f and
- $G(x) = lsb(x)lsb(x^2 \mod N) \ldots lsb(x^{2^{\ell(|x|)-1}} \mod N)$ is a PRNG.

In general: A PRNG can be constructed from every one-way permutation.

[1] p and q must be selected at random

Blum-Blum-PRNG contd.

Input

- ℓ, the length of the sequence.

Output

- Sequence $z_1, z_2, \ldots, z_\ell$ of pseudo random bits.

Algorithm

- Pick two primes p and q at random, s.t. $p, q \equiv 3 \mod 4$.
- Compute $N = pq$.
- Select at random a seed s with $1 \leq s \leq N - 1$ and $\gcd(s, N) = 1$.
- Compute $x_0 = s^2 \mod N$
- for $i = 1$ to ℓ do
  - $x_i = x_{i-1}^2 \mod N$
  - $z_i = lsb(x_i)$
- end for.

Example 2 RSA Generator

Input

- ℓ, the length of the sequence.

Output

- Sequence $z_1, z_2, \ldots, z_\ell$ of pseudo random bits.

Algorithm

- Generate two strong RSA primes p and q.
- Compute $N = pq$.
- Compute $\phi = (p-1)(q-1)$.
- Choose e with $1 \leq e \leq \phi$ and $\gcd(e, \phi) = 1$.
- Pick at random an initial value x_0 with $1 \leq x_0 \leq n-1$.
- for $i = 1$ to ℓ do
  - $x_i = x_{i-1}^e \mod N$
  - $z_i = lsb(x_i)$
- end for.

Example 3 SHA-1 PRNG

- Cryptographic hash function SHA-1 is used as output function.
- Inner state consists of 40 bytes, which are initialized with some seed.
- The last 8 bytes are interpreted as counter. The state function increments the counter by 1.
- The SHA-1 value of the inner state is computed (20 bytes). Output the first 8 bytes as random values.
- K3 requirement is met, K4 is not!
- Promises high speed, since only one SHA-1 computation is performed per round.

Example 4 SHA-1 RIPEMD

- Cryptographic hash functions:
  - SHA-1 as output function,
  - RIPEMD-160 as state function.
- Inner state consists of 20 bytes, which are initialized with some seed.
- The inner state is advanced by applying RIPEMD-160.
- The SHA-1 value of the inner state is computed (20 bytes). Output all 20 bytes as random values.
- K3 and K4 requirements are met.
- Two values have to be computed per round.

Example 5 Secure Random

- Part of the Java cryptography architecture.
- Cryptographic hash function SHA-1 as output function and as part of the state function.
- Inner state consists of 20 bytes, which are initialized with some seed.
- Advance to next round: $I_{n+1} = I_n + A_n + 1 \mod 2^{160}$.
- The SHA-1 value of the inner state is computed (20 bytes). Output all 20 bytes as random values.
- K3 and K4 requirements are met.
- Per round only one hash value is computed.

Example 6: TripleDES

- Symmetric encryption using TripleDES as state function.
- Key cannot be hard coded, but is rather part of the inner state.
- The key and the initial value for the encryption are determined from the seed.
- The inner state is output and encrypted progressively.
- K3 requirement is met, K4 is not met.
- Per round one TripleDES computation is performed.

Example 7: Secure Rijndael

- Instead of TripleDES AES (Rijndael) is used (much faster).
- Also the key k is changed $k_{n+1} = k_n \oplus A_n$.
- Therefore K4 requirement is met.
- Problem: Changing the key in Rijndael is quite inefficient.
- Solution: Key is only changed every 10 rounds.
- Hence, K4 requirement can be met in steps (by loosing efficiency).

3.4 Statistical Tests

Monobit test and Poker test

Monobit test

- The test is passed, if the number of ones (i.e. the set bits) in a sequence of 20000 bits is in the interval [9726, 10274].

Poker test

- A sequence of 20000 bits is separated in 5000 4-bit segments.
- The frequency of occurence of the 16 different segment types is counted and stored in the variables $f(i), i = 0, \ldots, 15$.
- The test is passed if

$$2.16 \leq X \leq 46.17 \text{ with } X = \frac{16}{5000} \sum_{i=0} (f(i) - 5000)^2.$$

Runs test and longruns test

Runs test

- *run:* Sequence of equal bits (zeros or ones).
- In a sequence of 20.000 bits the frequency of occurence of equal runs (same length, same bit value) is counted.
- The test is passed if the frequency of occurence is in a given interval.

Longruns test

- The test is passed if no *run* of length 26 or more exists in a sequence of 20.000 bits.

Continuous test and auto-correlation test

Continuous test

- Two consecutive n-bit blocks ($n \geq 16$) are compared.
- The test fails if they are equal.

Auto-correlation test

- This test checks correlations within a generated bit sequence.
- A sequence of 10.000 bits $b_1 \ldots b_{10000}$ is generated. For $t = 1, \ldots, 5000$ the following values are computed:

$$Z_t = \sum_{j=0}^{5000} b_j \oplus b_{j+t}.$$

- The test is passed if all Z_t are in the interval [2327,2673].

Maurer's universal test

- Detects a very general class of possible errors of a generator.
- Basic idea: It should be impossible to compress a sequence of random numbers (without losing information).
- Measure for the compressibility of a random sequence.

Naive compression test

- Determine the compression rate (i.e. zip compression).

Careful: Seed contains the entire "randomness".

- Number of possible seeds $\geq$ number of possible sequences, keys etc.
- A seed of length 160 bits results in at most 2^{160} outputs.

4. Finding primes and primality testing

4.1 The basic algorithm

There are infinitely many primes

- How to find fast, large primes at random?
- Find a prime p in the range $B, \ldots, 2B-1$, where B is big, say $B \geq 2^{100}$.
- Let's start with the famous theorem of Euclid (about 300BC) about the number of primes:

Theorem 4.1
There are infinitely many primes.

Proof of Euclid's prime number theorem

- Assume the set of primes is finite and we can denote it as $P = \{p_1, \ldots, p_k\}$.
- The number $p = p_1 \cdot p_2 \cdots p_k + 1$ is relatively prime to each of the primes in P.
- On the other hand, according to the fundamental theorem of arithmetic on unique prime factorization, p has some prime factors.
- These prime factors must be in P.
- This contradiction proves the theorem.

Mersenne primes

- Mersenne numbers are of the form $M_n = 2^n - 1$, $n \in \mathbb{N}$.
- Such a number is not necessarily prime, for example $M_4 = 15$.
- When n is composite, then so is M_n.
- When searching for Mersenne primes, one may therefore assume n to be prime.
- The largest known prime has almost always been a Mersenne prime. Why Mersennes?
- The way the largest numbers N are proven prime, is based on the factorizations of either $N + 1$ or $N - 1$.
- For Mersennes the factorization of $N + 1$ is as trivial as possible (a power of two).
- The largeset prime known is $M_{43112609}$ – a number with 12978189 decimal digits.

Algorithm 4.1

- Input: $B \in \mathbb{Z}_{\geq 2}$
- Output: A prime p in the range $[B, 2B-1]$
  1. Repeat steps 2-3:
  2. Choose any (odd) number p in $[B, 2B-1]$ at random.
  3. Test whether p is prime.
  4. Until the test accepts p.
  5. Output p.

Analysis Algorithm 4.1

- How long does this take?
- Already testing primality of a given number N is a problem.
- Testing all possible factors would take about $\sqrt{N}$ divisions.
- This is beyond the capabilities of any existing computer.
- How many primes are there in a specific range?
- Maybe none at all?

4.2 Probabilistic algorithm

A probabilistic algorithm

- A little modification allows us to get a satisfactory result:
- Do we really need to be absolutely sure that N is prime?
- We will content ourselves with a number that is prime with a choosably small error probability!
- We replace the rigid primality test by the following algorithm – called Fermat Test.

Algorithm 4.2

- Input: A number $N \in \mathbb{Z}$, and a confidence parameter $t \in \mathbb{N}$.
- Output: Either 'N is composite' or 'N is possibly prime'.

1. Repeat t times steps 2-6:
2. Pick at random a from $\mathbb{Z}_N \backslash \{0\}$.
3. Compute $g = \gcd(a, N)$.
4. If $g \neq 1$, then return 'N is composite'.
5. Compute $b = a^{N-1} \in \mathbb{Z}_N$.
6. If $b \neq 1$, then return 'N is composite'.
7. Return 'N is possibly prime'.

Analysis of Algorithm 4.2

- The Fermat Test fails if it claims 'N is probably prime' for a composite number N.
- If N is composite the Fermat Test will correclty answer that 'N is composite'.
- Consider the case that N is composite.
  - A number a is called a *Fermat witness* if $a^{N-1} \not\equiv 1 \mod N$ and a *Fermat liar* otherwise.
  - If there is at most one witness a with $a^{N-1} \not\equiv 1 \mod N$ then at most half of all possible as are liars.
  - So if there is at least one Fermat witness, then the probability of failure is small:
  - The probability that the Fermat Test answers 'N is probably prime' under the condition that N is composite is at most $\frac{1}{2}$.
  - Since we use the Fermat Test t times independently, the failure probability is at most 2^{-t}.

Properties of the repeated Fermat Test

1. If it outputs 'N is composite', then N is composite.
2. If it outputs 'N is probably prime', then either there is no Fermat witness for N, or N is prime with probability at least $1 - 2^{-t}$.

The last statement can be rephrased: The probability that the test fails on a composite number N, with a witness is at most 2^{-t}.

4.3 Carmichael numbers and further results

Carmichael numbers

- Composite numbers N without any Fermat witness.
- Examples:
  - $561 = 3 \cdot 11 \cdot 17$
  - $1105 = 5 \cdot 13 \cdot 17$
  - $1729 = 7 \cdot 13 \cdot 19$
- Alford et al. showed that there are infinitely many Carmichael numbers.
- Problem can be fixed by using 'The strong pseudo primality test'.

Algorithm 4.3

- Input: A number $N \in \mathbb{Z}$.
- Output: Either 'N is composite' or 'N is possibly prime'.
  1. Write $N - 1 = 2^k n$, where n is odd.
  2. Choose $a \in \mathbb{Z}_N$ at random.
  3. Compute $b \leftarrow a^n \mod N$.
  4. If $b \equiv 1 \mod N$ then return 'N is probably prime'.
  5. Repeat the following k times:
  6. if $b \equiv -1 \mod N$ then return 'N is probably prime',
  7. otherwise compute $b \leftarrow b^2 \mod N$.
  8. Return 'N is composite'.

Correctness of the strong pseudo primality test

- Correctness is based on the following two facts:
  - In a field F, the only elements a with $a^2 = 1$ are $a = \pm 1$.
  - If N is prime then $\mathbb{Z}_N$ is a field.
- Illustration of the strong pseudo primality test:
  - Choose $a = 113$.
  - Three examples:

N	$N-1 = 2^k n$	b_0	b_1	b_2	b_3	b_4	Output
553	$2^3 \cdot 69$	407	302	512	22		composite
557	$2^2 \cdot 139$	556	1				probably prime
561	$2^4 \cdot 35$	56	331	116	67	1	composite

Error probabilty of Algorithm 4.3

- An integer is squarefree if no square of a prime number divides it.
- **Fact 4.2** A Carmichael number is squarefree.
- We obtain the following characterization of a Carmichael number:

$$\begin{aligned} & N \text{ is a Carmichael number} \\ \Leftrightarrow \quad & N \text{ is squarefree and } \forall p | N, p-1 | N-1 \end{aligned}$$

Example

- The smallest Carmichael number is $N = 561 = 3 \cdot 11 \cdot 17$.
- $N - 1 = 560 = 2^4 \cdot 5 \cdot 7$.
- $\phi(N) = 2 \cdot 10 \cdot 16 = 320$.
- For any $a \in \mathbb{Z}_N^\times$, we have $a^2 = 1$ in $\mathbb{Z}_3$, $a^{10} = 1$ in $\mathbb{Z}_{11}$, and $a^{16} = 1$ in $\mathbb{Z}_{17}$.
- Hence $a^{80} = 1$ in $\mathbb{Z}_3, \mathbb{Z}_{11}$, and $\mathbb{Z}_{17}$, and by the CRT also in $\mathbb{Z}_N$.
- Now $80 \mid 560 = N - 1$, so that also $a^{N-1} = 1$ in $\mathbb{Z}_N^\times$.

Properties of the strong pseudo primality test

(i) If N is prime, the test returns 'probably prime'.

(ii) If N is composite and not a Carmichael number, the test returns 'composite' with probability at least $1/2$.

(iii) If N is a Carmichael number, the test returns a proper factor of N with probability at least $1/2$.

(iv) For an n-bit input N, the test uses $O(n^3)$ bit operations.

Repeating the Algorithm 4.3

- The probability that the strong pseudoprimality test answers incorrectly 'N is probably prime' for a composite N is at most $1/2$.
- When we use this test t times independently, the error probability is at most 2^{-t}.

Theorem 4.3

The strong pseudo primality test, repeated t times, has the following properties.

(i) If it outputs 'N is composite', then N is composite.

(ii) If it outputs 'N is probably prime', then N is prime with probability at least $1 - 2^{-t}$.

What does it mean when a primality test returns 'probably prime'?

- Is N then 'probably prime'?
- Of course not: N is either prime or it is not.
- 'probably' refers to the random choices made within the algorithm.
- If the test is run 1001 times, then it means the following: if N is not prime, then an event has been witnessed whose probability is at most 2^{-1001}.
- If you fly in an airplane whose safety depends on the actual primality of such an "industrial-strength pseudoprime", then this fact should not worry you unduly, since other things are much more likely to fail :-)

Note: The strong pseudoprimality test is the algorithm of choice for testing the primality of a given number N, unless deterministic security is required.

4.4 Finding prime numbers

Algorithm 4.4

- Want to find all prime numbers up to a value x.
  1. Start with a list $(2, 3, 4, ..., x)$ of all integers up to x.
  2. Initially, let p equal 2, the first prime number.
  3. Strike from the list all multiples of p less than or equal to x. $(2p, 3p, 4p, etc.)$
  4. Find the first number remaining on the list after p (this number is the next prime); replace p with this number.
  5. Repeat steps 3 and 4 until p^2 is greater than x.
  6. All the remaining numbers in the list are prime.

Analysis of Algorithm 4.4

- This method is old (from about 200 BC) and pretty, but its running time and space are about x log x 4and look pretty old in view of cryptographic requirements.
- Cryptographic requirements must be polynomial in log x.
- The following algorithm finds a large pseudo prime at the latter cost, in the range required by the RSA crypto system.

Algorithm 4.5

- Input: An integer n and a confidence parameter t.
- Output: A number N in the range from $2^{(n-1)/2}$ to $2^{n/2}$.
  1. Set $x = 2^{(n-1)/2}$.
  2. Repeat steps 3 and 4 until some N is accepted.
  3. Pick N at random from the set $\{\lceil x \rceil, \ldots, \lfloor \sqrt{2}x \rfloor\}$.
  4. Call the strong pseudoprimality test t times. Input is N and everytime a new a independently chosen from $\{1, \ldots, N-1\}$. Accept N if and only if all these tests return 'N is probably prime'.
  5. Return N.

Analysis of Algorithm 4.5

- In order to find a prime, choose numbers and then test them until we find a prime.
- But if there is no prime in the given range?
- Then the algorithm would never stop.
- If there are only very few primes, the algorithm would run for a long time, this is also undesireable.
- Good news: There are abundant primes in a given range.

Prime number theorem

- The prime number theorem says approximately how many primes there are up to some bound x.
- Let $\pi(x)$ denote the number of primes p with $p \leq x$.
- Besides $\pi(x)$, the function p_n is also useful. It denotes the nth prime number, for example $p_3 = 5$.

Theorem 4.4

We have approximately

$$\pi(x) \approx \frac{x}{lnx} \text{ and } p_n \approx n \text{ ln } n.$$

Theorem 4.5

On input $n \geq 11$ and t, the output of the algorithm is prime with probability at least $1 - 2^{-t+1}\, n$. It uses an expected number of $O(tn^4)$ bit operations.

A word on deterministic prime tests

- There exist deterministic prime tests.
- The first one running in polynomial time was proposed in 2002 by Agrawal, Kayal and Saxena.
- The running time of this test is approximately $O(\log^6(N))$ operations, where N is the number to be tested.

5. One-way functions and Hash-functions

One way function

Definition 5.1

- One-way function $f : A \to B$.
  - Linear or polynomial runtime.
  - A single computer can calculate it in less than 50ms.
- $f^{-1}(y) = x$ cannot be computed in practice for most randomly selected values.
  - At least exponential runtime or NP-complete.
  - Cannot be computed in 1000 years, even if all computers in the world work on this task.

Applications of one-way functions

- First used by R.M. Needham and Wilkes in 1968:
  - Storage of encrypted passwords.
  - Example of Purdy (1974):

$$f : \mathbb{Z}/p\mathbb{Z} \to \mathbb{Z}/p\mathbb{Z} \text{ with } p = 2^{64} - 59$$

$$f(x) = x^{2^{24}+17} + a_1 x^{2^{24}+3} + a_2 x^3 + a_3 x^2 + a_4 x + a_5, a_i \approx 10^{19}$$

Applications of one-way functions contd.

- Cryptographic Hash-functions,
- pseudo random number generators,
- secret sharing,
- zero-knowledge proofs.

5.1 Cryptographic Hash-functions

Definition 5.2

- A one-way function f with:
  - Result $f(x) = y$ has constant length.
  - Second preimage resistant: For a given value v it is practically impossible to compute a different value w, s.t. $f(v) = f(w)$.
  - Collision free: It should be difficult to find two different values v and w, such that $f(v) = f(w)$.

Widely used Hash-functions

- Message Digest, MD2, MD4, MD5 (weak),
- Secure Hash Algorithm 1, SHA-1: 160-bit Output, NIST-Standard (weak),
- SHA-2 family: SHA-224/512 (good).

Example SHA-256

- Input: Document with up to 2^{64} bit length.
- Input is processed in 512 bit blocks. Document might need padding.
- Output: 256 bit message digest.

5.2 One-way functions for asymmetric crypto systems

Definition 5.3

- Let a and b be two numbers (keys),
- m be the plaintext, and
- $f(a, m) = c$ the ciphertext.

Then f is a one-way trapdoor function, if

- every function g in a and c with $g(a, c) = f^{-1}$ cannot be computed in practice, i.e. f cannot be inverted knowing only public information.
- There exists a function $h(a, b, c) = f^{-1}$ (also with b as a parameter), which can be easily computed, i.e. f can be easily inverted knowing some secret.

6. RSA

Basic idea

- It is easy to find to large primes p and q and to compute $N = pq$.
- It is hard to factor N.

6.1 Theoretical Foundations

Definition 6.1 Euler's φ-function

- For $N \in \mathbb{N}$ we have

$$\varphi(N) = N \prod_{p|N} (1 - \frac{1}{p})$$

where the p are distinct.

Carmichael λ-function

- For $p = 2$ and $m = 2$

$$\lambda(2^2) = \varphi(2^2) = 2$$

- For $p = 2$ and $m \geq 3$, $m \in \mathbb{N}$

$$\lambda(2^m) = \frac{1}{2}\varphi(2^m)$$

- For all primes $p \geq 3$, $m \in \mathbb{N}$

$$\lambda(p^m) = \varphi(p^m) = p^m - p^{m-1}$$

- For $N = p_1^{m_1} p_2^{m_2} \dots p_k^{m_k}$

$$\lambda(N) = \operatorname{lcm}(\lambda(p_1^{m_1}), \lambda(p_2^{m_2}), \dots, \lambda(p_k^{m_k}))$$

Summary

- Now we have for all $m, N \in \mathbb{N}$, m, N coprime:
  - Theorem of Euler: $m^{\varphi(N)} \equiv 1 \mod N$.
  - Theorem of Carmichael: $m^{\lambda(N)} \equiv 1 \mod N$.
- Let $N = p_1 p_2 \dots p_k$, p_i distinct, then we have for all $m \in \mathbb{N}$

$$m^{\lambda(N)+1} \equiv m \mod N$$

and for all $k \in \mathbb{N}$

$$m^{k\lambda(N)+1} \equiv m \mod N.$$

- Let $N = pq$, p and q prime, $p \neq q$, and $k, m \in \mathbb{N}$ then we obtain an important formula for RSA:

$$m \equiv m^{k(p-1)(q-1)+1} \mod N.$$

6.2 Construction of the RSA one-way function

Setup

- $N = pq$ with p, q prime and $p \neq q$.
- $e \in \mathbb{N}$ with $e > 1$ and $\gcd(e, \lambda(N)) = 1$.
- $\mathrm{enc}(m, (e, N)) \equiv m^e \mod N$ with $m < N$, $m \in \mathbb{N}$.
- $d \in \mathbb{N}$ with $de \equiv 1 \mod \lambda(N)$, i.e. $m^{de} \equiv m \mod N$ for all $m \in \mathbb{N}$.
- $\mathrm{dec}(c, (d, N)) \equiv c^d \mod N$ with $c \in \mathbb{N}$ and $c < N$.

Properties

- $\mathrm{enc}(m, (e, N))$ and $\mathrm{dec}(c, (d, N))$ are easy to compute.
- d can be easily computed from p and q or from $\lambda(N)$.
- It is hard to compute $\mathrm{dec}(c, (d, N))$ without knowledge of d.
- $\mathrm{dec} = \mathrm{enc}^{-1}$

$\Rightarrow \mathrm{enc}(m)$ is a one-way function with trapdoor d.

Usage of RSA

- Most important task for using RSA in practice is to find large prime numbers.
- Today, the largest threat for RSA is the development of efficient algorithms for factoring integers.

6.3 Generating RSA keys

Step 1 and 2

1. Select two primes p and q, s.t.:
   - p and q are sufficiently large.
   - $|p - q|$ large, but not too large. IEEE P1363 suggests $\frac{1}{2} < |\log_2 p - \log_2 q| < 30$.
   - $p \equiv -1 \mod 12$ and $q \equiv -1 \mod 12$.
   - $p_1 = \frac{1}{2}(p-1)$, $q_1 = \frac{1}{2}(q-1)$, $\frac{1}{12}(p+1)$ and $\frac{1}{12}(q+1)$ are prime as well.
2. Compute $N = pq$ and $z = 2p_1q_1$.

Steps 3 and 4

3. Select at random $e \in \mathbb{N}$, such that:
   - $\gcd(e, z) = 1$.
   - $e - 1$ is neither a multiple of p_1 nor q_1.
4. Compute – using the Extended Euclidean Algorithm – d and z', s.t.:
   - $0 < d < z$,
   - $de + zz' = 1$.

Steps 5 to 7

5. Destroy p, q, z and z'.
6. Keep $S = d$ secret (secret key).
7. Publish $P = (e, N)$ (public key).

Selecting RSA primes

- Let n be the bit length of N.
- Use the strong pseudo prime algorithm with a confidence parameter $t = \lceil \log_2(2n) \rceil + s$.
- Then the algorithm returns a prime with probability at least $1 - 2^{-s}$ and it uses an expected number of $O((s + \log n)n^4)$ bit operations.
- With $s = 20 + \log_2 n$ the probability of failure is at most $0.000001/n$ and we expect $O(n^4 \log n)$ operations.

RSA efficiency

Find $n/2$-bit primes at random	$O(n^4 \log n)$,
Calculate d from e	$O(n^2)$,
Calculate powers modulo N	$O(n^3)$.

Altogether the key generation can be done in time $O(n^4)$, and the encryption in RSA of one plaintext block needs $O(n^3)$ bit operations.

6.4 Confidentiality with RSA

Encryption

- Alice sends a secret message m to Bob.
- Encryption by Alice:
  1. Represent plaintext as $m \in \mathbb{N}$.
  2. Divide $m = \ldots m_i \ldots$ in blocks, with $m_i < N_B$.
  3. Encrypt every m_i with $P_B = (e_B, N_B)$, the public key of Bob:

$$\mathrm{enc}(m_i, P_B) = m_i^{e_B} \mod N_B = R_i.$$

  4. Send R_i to Bob.

Decryption

- Decryption by Bob:
  - Decrypt every R_i with $S_B = d_B$, his secret key:

$$\mathrm{dec}(R_i, S_B) = R_i^{d_B} \mod N_B = m_i$$

6.5 Integrity and authenticity with RSA

Signing

- Alice signs a message m and sends it to Bob.
- Signing by Alice:
  1. Represent message as $m \in \mathbb{N}$.
  2. Choose a hash-function h and compute:

$$\bar{m} = h(m)$$

  3. Separate $\bar{m}$ in blocks, with $\bar{m}_i < N$, if necessary.
  4. Sign $\bar{m}$ with $S_A = d_A$, her secret key:

$$\text{sig-gen}(\bar{m}, S_A) = \bar{m}^{d_A} \mod N_A = \sigma.$$

  5. Send (m, σ) to Bob.

Verification

- Verification by Bob:
  1. Receives (m, σ).
  2. Computes – using the shared hash function:

$$\bar{m} = h(m).$$

  3. Separate $\bar{m}$ in blocks, with $\bar{m}_i < N$, if necessary.
  4. Verifies the signature with $P_A = (e_A, N_A)$, with the public key from Alice.

$$\text{sig-ver}(\sigma, P_A) = \sigma^{e_A} \mod N_A = \bar{m}'$$

  - The verification is successful if $\bar{m} = \bar{m}'$, then we can be sure about integrity and authenticity.
  - Otherwise: Discard message and signature.

6.6 Computing the secret key and factoring the modulus are equally hard

Proof - Part 1

- Given p and q, the factors of N. Find d.
- Since we know e, the public key, we can easily compute d by solving:

$$de \equiv 1 \mod (p-1)(q-1)$$

Proof - Part 2

- Let d be known. Find p and q. From $de \equiv 1 \mod (p-1)(q-1)$, we know there is some $k \in \mathbb{Z}$, such that $ed - 1 = k(p-1)(q-1)$.
- We know for all $a \in \mathbb{Z}_N^*$

$$a^{ed-1} \equiv 1 \mod N$$

- Let $ed - 1 = 2^s t$ with $t = 2i + 1$.
- Then for about half the elements $a \in \mathbb{Z}_N^*$ we have:

$$a^{2^{s-1}t} \not\equiv \pm 1 \mod N$$

- For an a that fulfills the above equation we have that $\gcd(a^{2^{s-1}t}, N)$ is a proper factor of N.
- Choose some $a \in \mathbb{Z}_N^*$ and compute $\gcd(a^{2^{s-1}t}, N)$.

In general: Every method, which can compute $\phi(N)$ or d, can also factor N.

6.7 Attacks on RSA

Homomorphic properties

- Encryption of products, without knowledge of factors:

$$m_1^e m_2^e \mod N = (m_1 m_2)^e \mod N = R_1 R_2$$

Forging signatures

- Let $\sigma_1 = m_1^d \mod N$ and $\sigma_2 = m_2^d \mod N$ be public.
- The attacker computes:
  - $\sigma_1\sigma_2 \mod N = m_1^d m_2^d \mod N = (m_1 m_2)^d \mod N$,
  - $\sigma_1^{-1} = (m_1^d)^{-1} \mod N = (m_1^{-1})^d \mod N$,
  - $-\sigma_1 = -(L_1^d) \mod N = (-m_1)^d \mod N$.
- Most of the time $m_1 m_2$ does not make sense for language, but it can be useful for numbers.
- Fix: insert redundancy (human semantics).

Adaptive chosen ciphertext attack

- Attacker Charlie wants to decrypt a message $R = m^e \mod N$ which was meant for Alice.
- Alice decrypts messages – except R – chosen by Charlie and shows him the outcome.
- Charlie chooses $x \in \mathbb{Z}_N^*$ and computes $R_1 = Rx^e \mod N$.
- Alice computes for Charlie $m_1 = R_1^d \mod N$.
- We get
$$m_1 \equiv R_1^d \equiv R^d (x^e)^d \equiv mx \mod N.$$
- So Charlie can compute the message R.

Using the same small exponent e for several users

- Let for example $e = 3$ (for best performance this is even recommended).
- If we have three participants with different modulus and the same plaintext we have:
  - $R_1 = m^3 \mod N_1$
  - $R_2 = m^3 \mod N_2$
  - $R_3 = m^3 \mod N_3$
- Then by applying the Chinese Remainder Theorem we get:

$$R = m^3 \mod N_1 N_2 N_3 = m^3.$$

- A suggested solution: split messages in shorter blocks and pad with random values.

Using the same modulus for several users

- Knowledge of one pair (e_0, d_0) allows factoring N.
- Then all pairs (e_k, d_k) can be computed.

$\Rightarrow$ Everybody can read all messages - despite having different keys.

Decryption of algebraically "related" messages

- Choose for example $e = 3$.
- Consider

$$R_1 = m^3 \mod N \quad \text{and} \quad R_2 = (m+1)^3 \mod N$$

- Then it holds:

$$\frac{R_2 + 2R_1 - 1}{R_2 - R_1 + 2} = \frac{(m+1)^3 + 2m^3 - 1}{(m+1)^3 - m^3 + 2} = \frac{3m^3 + 3m^2 + 3m}{3m^2 + 3m + 3} = m$$

Fixpoint attack

- Let o be the order of e in $\frac{\mathbb{Z}}{\lambda(N)\mathbb{Z}}$. Then it holds that

$$e^o \equiv 1 \mod \lambda(N)$$

- and for all messages m it holds

$$m^{e^o} \equiv m \mod N.$$

- Solution: pick e with large order.

Wiener's attack on short secret keys

- Small d can be found in polynomial time by using partial fraction decomposition.
- Solution: the length of d and N should be about the same.

Conclusion

- None of the above mentioned attacks is dangerous if RSA is used with the recommended protection mechanisms.
- At the moment: Only advances in developing efficient factoring algorithms cause pressure.

7. Foundations of asymmetric crypto systems based on groups

Basis for asymmetric Cryptosystems

- A finite cyclic abelian group G with prime order $|G|$ and a generator $g \in G$.
- If the order of $|G|$ is not prime then a prime order subgroup H of G is selected.
- The notation of the group operation is sometimes written multiplicativly and sometimes additivly – depending on the context.

Basis for elliptic curve asymmetric crypto systems

- The additive group of points on an elliptic curve.

Properties of G for asymmetric crypto systems

- Multiplication/exponentiation is easy.
- Inverse operation (division/ computing logarithms) is hard.
- It is possible to generate elements in G that are distributed close to uniform.

Dicrete log problem (DLP)

Definition 7.1:

- Let G be a finite group.
- Given $g, h \in G$, find the smalles $n \in \mathbb{N}$, if one exists, such that $g^n = h$

7.1 Cryptosystem based on the discrete log problem

General rule

- If it is possible to break a crypto system in G then all other crypto systems working in G are insecure as well.

Diffie-Hellman key exchange

Goal:

- Alice and Bob want to chose a random element from G as their secret.

Prerequisites:

- Public: group G and an element $g \in G$ with large order.

Diffie-Hellman key exchange cntd.

Method:

i) Alice generates $a \in [1, |G| - 1]$ and sends $x = g^a$ to Bob.

ii) Bob generates $b \in [1, |G| - 1]$ and sends $y = g^b$ to Alice.

iii) Alice computes $y^a = (g^b)^a = g^{ab}$.

iv) Bob computes $x^b = (g^a)^b = g^{ab}$.

$\Longrightarrow$ g^{ab} is only known to Alice and Bob.

Attack (Diffie-Hellman problem, DHP

- Evesdropper Eve knows: $G, g, x = g^a$ and $y = g^b$.
- If Eve can compute g^{ab}, then she has solved the DHP in G.
- It is conjectured (e.g. by Maurer (1994)) that for most groups used in cryptography the DHP and the DLP is equivalent.

ElGamal encryption(1985)

Goal:

- Alice wants to send to Bob the message $m \in G$ in a confidential way.

Prerequisites:

- Public: Group G and some element $g \in G$ with large order.
- Secret key of Bob: $b \in [1, |G| - 1]$.
- Public key of Bob: $B = g^b$.

ElGamal encryption cntd.

Method:

i) Alice generates a random $a \in [1, |G| - 1]$.

ii) She computes $x = g^a$ and $c = B^a m$, and sends (x, c) to Bob.

iii) Bob computes
$cx^{-b} = B^a m (g^a)^b = (g^b)^a m (g^a)^{-b} = g^{ab-ab} m = m.$

ElGamal Signatures (1985)

Goal:

- Bob signs a message $m \in \mathbb{Z}_d$.

Prerequisites:

- Public: Group G and some element $g \in G$ with large order d.
- Public: $f : G \to \mathbb{Z}_d$, bijective.
- Secret key of Bob: $b \in \mathbb{Z}_d$.
- Public key of Bob: $B = g^b$.

Method:

i) Signing by Bob:

a) Generates a random $k \in \mathbb{Z}_d^\times$.

b) Computes $K = g^k$.

c) Solves:

$$k\sigma + bf(K) \equiv m \mod d$$

for $\sigma \in \mathbb{Z}_d$.

d) Sends (m, K, σ) to Alice. The signature of m is (K, σ).

ii) Alice computes:

$$z = B^{f(K)} K^\sigma =$$

and verifies $z = g^m$?

Digital signature according to DSA without Hash-function

DSA: Digital Signature Algorithm.

- Motivation: Speed up the verification of the digital signature based on ElGamal.
- Goal and prerequisites as above.

Method:

i) Signing by Bob:

a) Generates a random $k \in \mathbb{Z}_d^\times$.

b) Computes $K = g^k$.

c) Solves:

$$-bf(K) + \sigma k \equiv m \mod d$$

for $\sigma \in \mathbb{Z}_d$.

d) Sends (m, K, σ) to Alice. The signature of m is (K, σ).

ii) Verification by Alice:

a) $u = m\sigma^{-1}$

b) $v = f(K)\sigma^{-1} \mod d$

c) $w = g^u B^v = g^{m\sigma^{-1}} g^{vb} = g^{m\sigma^{-1} + bf(K)\sigma^{-1}} = g^{\sigma^{-1}(m+bf(K))}$

d) Verify if $w = s$.

Advantage: Verification via two exponentiations in G, ElGamal needs three.

Encryption due to Massey-Omura

- Rarely used, but very elegant.

Goal:

- Alice wants to send a message m to Bob in a confidential way.

Prerequisites:

- Public: group G with large order.

Massay-Omura cntd.

Method:

i) Alice generates at random $a \in [1, |G| - 1]$ with $\gcd(a, |G|) = 1$ and sends to Bob $x = m^a$.

ii) Bob generates at random $b \in [1, |G| - 1]$ with $\gcd(b, |G|) = 1$ and sends to Alice $y = x^b = m^{ab}$.

iii) Alice sends $z = y^{a^{-1}} = m^{aba^{-1}} = m^b$.

iv) Bob computes $z^{b^{-1}} = m^{bb^{-1}} = m$

Digital Signatures according to Nyberg-Rueppel (1996)

Properties:

- A series of signature schemes with message recovery.
- Difference to ElGamal: Message is $m \in G$.
- In the lecture we look at the variant of Piveteau (1993), without message recovery.

Goal:

- Bob signs a message $m \in G$.

Nyberg-Rueppel cntd.

Prerequisits:

- Public: Group G and $g \in G$ with large order.
- Public: $f : G \to \mathbb{Z}/|G|\,\mathbb{Z}$, bijective.
- Secret key of Bob: $b \in [1, |G| - 1]$ with $\gcd(b, |G|) = 1$.
- Public key of Bob: $P_B = g^b$.

Nyberg-Rueppel cntd.

i) Signing by Bob:
   a) Generates a random $k \in [1, |G| - 1]$ with $\gcd(k, |G|) = 1$.
   b) Computes $s = mg^{-k}$.
   c) Solves:
   $$1 \equiv bf(s) + tk \mod |G|$$
   for $t \in [1, |G| - 1]$.
   d) Sends (m, s, t) to Alice. The signature of m is (s, t).

ii) Verification by Alice:
   - Compute
   $$P_B^{-f(s)} s^t = g^{tk-1-tk} m^t = z$$
   - Check $z = m^t g^{-1}$.

Reduction of problems

- It is always conjectured, but there is no proof that breaking one of the above schemes also breaks RSA.
- There is no proof that breaking RSA is equivalent to factoring the modulus. In fact, $Boneh/Venkatesan(1998)$ back up the conjecture that this is not the case.
- $DHP = DLP$ can only be proven for some special cases.

The function $f : G \to \mathbb{Z}/\left|G\right|\mathbb{Z}$

- Using $g = \mathbb{F}_p^\times$, f is canonical.
- For other groups one can weaken the condition that f has to be bijective. We can look for a set M, with about the same order as G, $|G| \sim |M|$, such that

$$f : G \to \mathbb{Z}/\left|M\right|\mathbb{Z}$$

is almost injective.

Usage of f in Elliptic curve cryptography

- (x, y) are coordinates of points, but only x is inserted into f.
- Using $G = \mathbb{F}_p$, p prime $|G| \sim p$ is chosen and $x \in \mathbb{F}_p$ is considered an integer.
- Using $G = F$, F a field with $char(F) = 2$, i.e. $F = \mathbb{F}_{2^n}$, x has to be denoted as integer as well. In practice x is considered a basis of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$ and the coordinates of x are digits of a binary number.

7.2 Groups for asymmetric crypto systems

DLP today

The presented crypto schemes are based on arbitrary abelian groups, but for implementation simple operations are required, e.g. a simple algebraic expression.

DLP today

- Easily solvable in additive subgroups of finite fields.
- Not solvable on elliptic curves over finite fields.
- Usually for crypto systems the group $\mathbb{F}_p^\times$ over large p is considered.
- DLP can be solved in $\mathbb{F}_p^\times$ in subexponential time (McCurley (1990) and Adleman (1994)). Methods are based on ideas of factoring with number field sieves (Lenstra (1993)).

Increasing complexity using number theory

- Replace $\mathbb{F}_p^\times$ by $E(\mathbb{F}_p)$, the set of rational points on an elliptic curve in $\mathbb{F}_p$.
- Result: DLP in the additive group $E(\mathbb{F}_p)$ is several orders of magnitude harder that DLP in the multiplicative group $\mathbb{F}_p^\times$ of same order.

7.3 Comparison of key lengths

Time complexity

- DLP in $E(\mathbb{F}_q)$: exponential to $n = \log_2 q$
- DLP in $\mathbb{F}_p^\times$ subexponential to $N = \log_2 p$

Key length in elliptic curve crypto systems grows proportional to the 3. root of the key length in conventional crypto systems.

Comparison of key length

AES	*RSA/DLP*	*ECC/DLP*
80	1024	160
112	2048	224
128	3072	256
192	8192	384
256	15360	512

Complexity of implementation

- The operation in $E(\mathbb{F}_q)$ is more complex than in $\mathbb{F}_p^\times$: otherwise there is no difference.

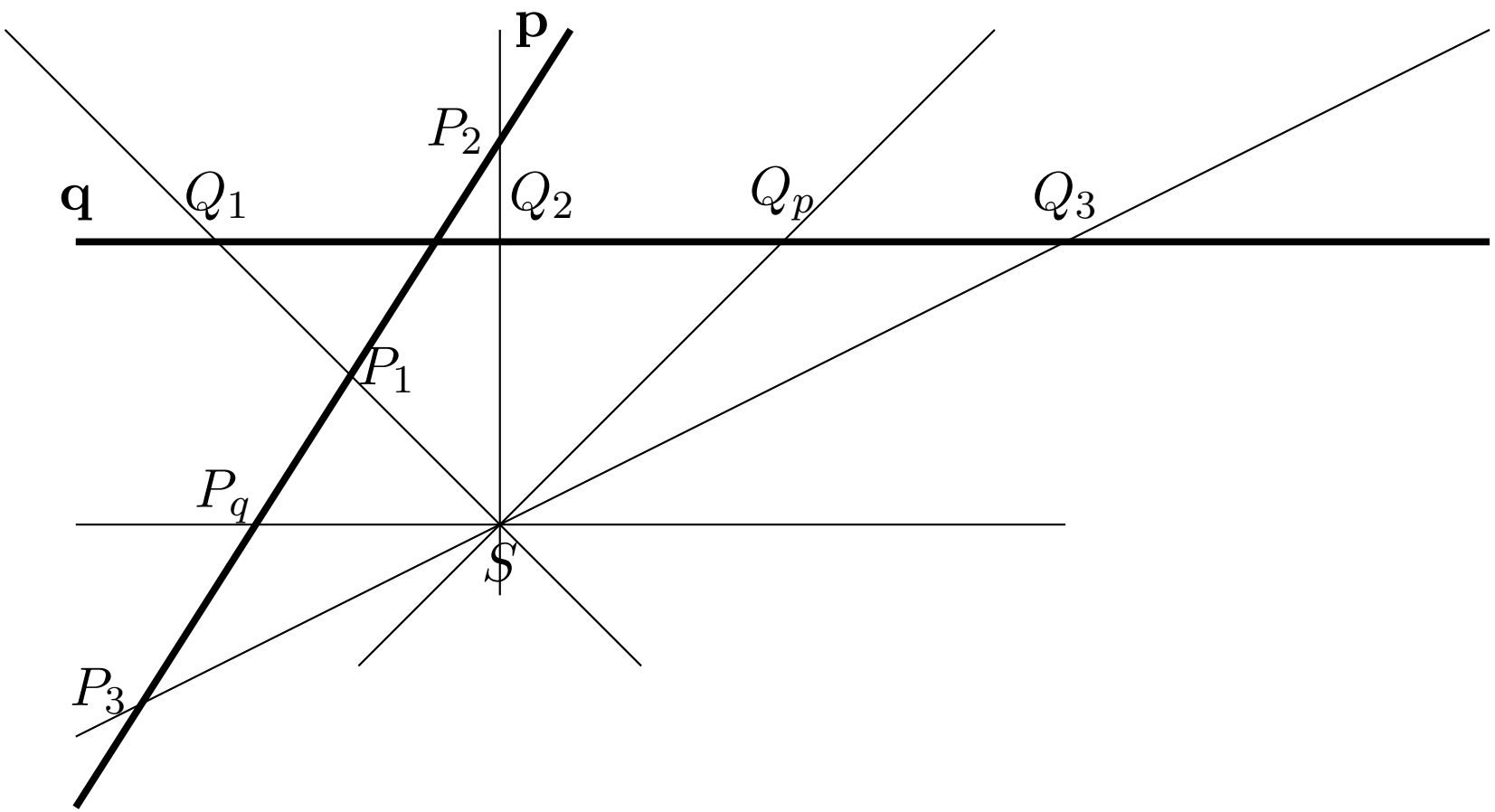
8. Elliptic curves in finite fields

8.1 Projective geometry

Introduction

- Content: properties of structures that do not change during projection.
- Relation to elliptic curves: results in an additional point, which is important for the group definition.
- History:
  - da Vinci, Dürer, 15th century: study of perspective in paintings.
  - Monge, 18th century: descriptive geometry (forms of projections).
  - Poncelet, 19th century: Projective geometry.
- Observation:
  - In images of the central projection parallel lines intersect.
  - The difference between parallel and intersecting lines is lost.

Idea: Consider artificial elements



Artificial points

- Consider a line p.
- Select a second line q and a point S, the auxiliary point, which neither lies on p nor q.
- Draw lines through S: Q_i is is the point on q that was projected by P_i on p.
- Relating P_i to Q_i is bijective, except for points P_q and Q_p.
- Define:
  - the point Q_p on q has an image on p: the artificial point P_∞.
  - the point P_q on p has an image on q: the artificial point Q_∞.

Artificial points cntd.

- Q_p is reached, if one follows the line p in both directions.
- With Q_p, p becomes a closed line in the projection.
- Every line has only one artificial point.
- In projections it does not make sense to distinguish inner and outer points for line segments.
- Instead for a pair of points we say:
  - they divide each other, e.g. (P_1, P_3) or (P_2, P_q) or
  - they do not divide each other, e.g. (P_1, P_q) or (P_2, P_3) .

Distinguish using artificial points

- Consider an additional line r in the image.
  - If p and r are parallel, then r has also the artificial point P_∞ (Q_p in the projection onto q).
  - In case p and r intersect, then r has an artificial point different from P_∞ (Q_r in the projection onto q).
- The images of all parallel lines intersect in one point during a central projection from one plane into an inclined plane.
  - i) The artificial point depends only on the direction of the line and can be associated with the direction.
  - ii) The artificial point is considered as a regular point of a projective line.
  - iii) P_∞ and Q_∞ determine the artificial line:
    - the set of artificial points of all lines,
    - the set of all intersections of parallel lines.

In general two lines intersect in the projective plane in exactly one point.

8.1.1 Projective coordinates on a line

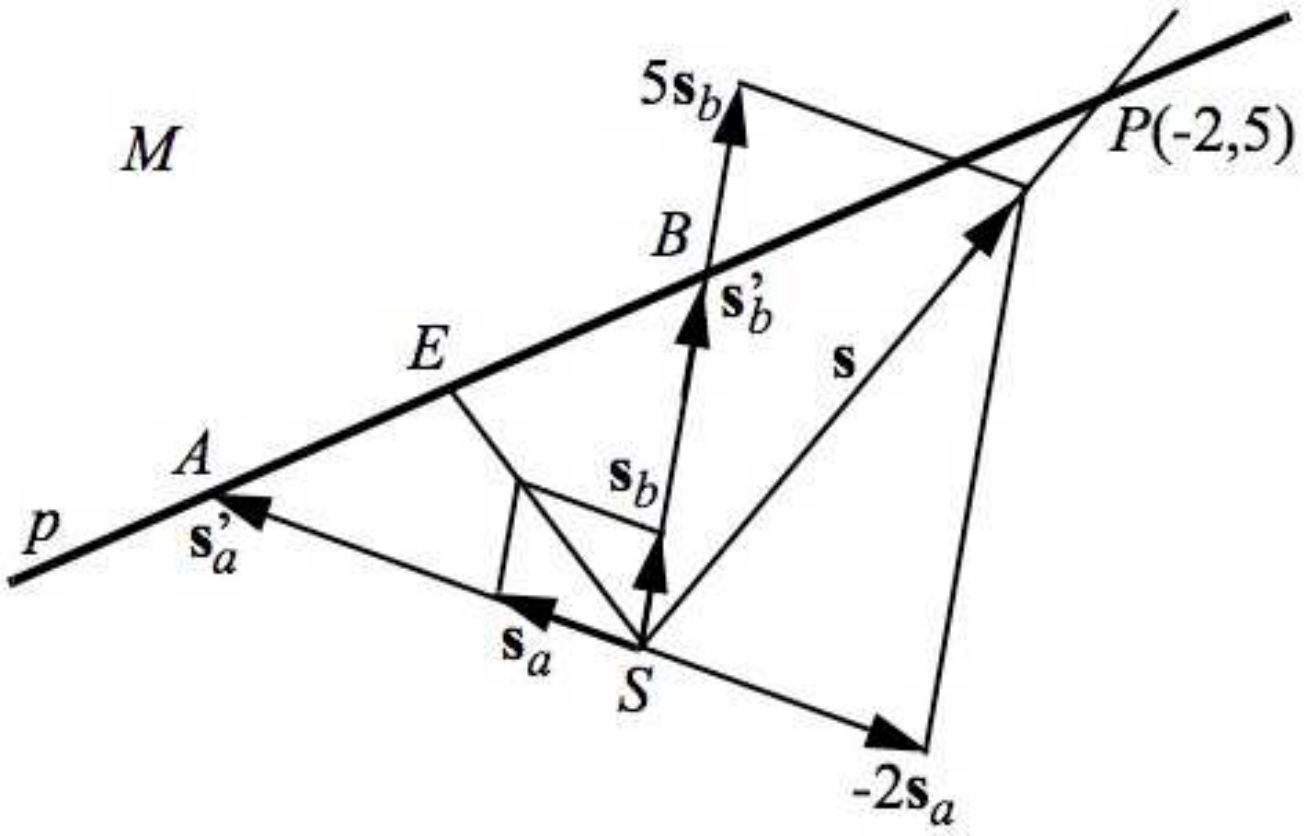
Projective coordinates

Problem:

- the position of a point on an affine line is determined by a coordinate. The coordinate depends on the choice of the origin and the unit line segment.
- Affine maps between lines preserve the coordinate ratios of original and image points, respectively.

Solution: projective coordinates.

Approach



Approach cntd.

- Consider the line p in the plane M, i.e. a structure in the next dimension.
- Select a basis on M with two linearly independent vectors $\mathbf{s}_a$ and $\mathbf{s}_b$ and a point S not lying on p as the origin.
- A unique line with the auxiliary point S is assigned to every point P on p.
- Therefore P is uniquely determined by the coordinates (a, b) of a vector

$$\mathbf{s} = a\mathbf{s}_a + b\mathbf{s}_b.$$

Approach contd.

- With every $k \neq 0$ also (ka, kb) determine the point P.
- (a, b) are called the homogeneous coordinates of P.
- The points $A(1, 0)$ and $B(0, 1)$ on p are the basic points and $E(1, 1)$ is the unit point on p corresponding to the basis M.
- The homogeneous coordinates of P_∞ are determined by the slope of p.

8.1.2 Algebraic approach in the projective plane

Definition 8.1 Let $\mathbb{F}$ be a field then

- $\mathbf{A}^1(\mathbb{F}) = \{(a)|a \in \mathbb{F}\}$ is the one dimensional affine space over $\mathbb{F}$.
- $\mathbf{A}^2(\mathbb{F}) = \{(a_1, a_2)|a_1, a_2 \in \mathbb{F}\}$ is the two dimensional affine space over $\mathbb{F}$.

Definition 8.2 Let $a_i, b_i, t \in \mathbb{F}$, $t \neq 0$,

- $(a_1, \ldots, a_k)$ and $(b_1, \ldots, b_k)$ are homogeneously equivalent $[a_1, \ldots, a_k] \sim [b_1, \ldots, b_k]$ if $a_i = tb_i$.
- The one dimensional projective space over $\mathbb{F}$, i.e. the projective line, is

$$\mathbf{P}^1(\mathbb{F}) = \{[a_1, a_2] | (a_1, a_2) \neq (0, 0)\}$$

  $[a_1, a_2]$ are the homogeneous coordinates of a point on $\mathbf{P}^1(\mathbb{F})$.
- The two dimensional projective space over $\mathbb{F}$, i.e. the projective plane, is

$$\mathbf{P}^2(\mathbb{F}) = \{[a_1, a_2, a_3] | (a_1, a_2, a_3) \neq (0, 0, 0)\}$$

  $[a_1, a_2, a_3]$ are the homogeneous coordinates of a point on $\mathbf{P}^2(\mathbb{F})$.
- The homogeneous coordinates $[a_1, \ldots, a_k]$ are normalized, if $a_1, \ldots a_k$ are coprime.

Geometric and algebraic considerations

- Geometrically speaking one can consider $\mathbf{P}^2$ as the affine space in addition with the set of directions, i.e.

$$\mathbf{P}^2 = \mathbf{A}^2 \cup \{\texttt{set of directions in } \mathbf{A}^2\}.$$

- Algebraically, one considers the maps

$$\begin{aligned} \mu\colon \mathbf{A}^2(\mathbb{F}) &\to \mathbf{P}^2(\mathbb{F}) \\ (a,b) &\mapsto [a,b,1] \\ v\colon \mathbb{F} &\to \mathbf{P}^2(\mathbb{F}) \\ a &\mapsto [a,1,0] \end{aligned}$$

  as well as the point $[1,0,0]$.
- The map μ embeds the affine space canonically into the projective space.
- The map v captures with the base field all affine directions.
- $[1,0,0]$ is the artificial point of the direction line.

- It holds that

$$\mathbf{P}^2(\mathbb{F}) = \mu(\mathbf{A}^2(\mathbb{F})) \cup v(\mathbb{F}) \cup \{[1,0,0]\}$$

or simply $\mathbf{P}^2 = \mathbf{A}^2 \cup \mathbf{P}^1$.

8.1.3 The line in the plane

The line in the plane

- Line L in the affine plane:
  Set of zeros of

$$f(x,y) = ax + by + c \text{ with } (a,b) \neq (0,0) \text{ in } \mathbf{A}^2(\mathbb{F}),$$

  i.e. the affine coordinates

$$L = \{(x,y) \in \mathbf{A}^2(\mathbb{F}) | f(x,y) = 0\}$$

- Line L in the projective plane:
  Set of zeros of

$$F(X,Y,Z) = aX + bY + cZ \text{ with } (a,b,c) \neq (0,0,0) \text{ in } \mathbf{P}^2(\mathbb{F}),$$

  i.e. the homogeneous coordinates

$$L = \{(X,Y,Z) \in \mathbf{P}^2(\mathbb{F}) | F(X,Y,Z) = 0\}$$

Projective case

- For the projective line we have $F(tX, tY, tZ) = tF(X, Y, Z), t \neq 0$, i.e. with $[X, Y, Z] \in L$ also $[tX, tY, tZ] \in L$.
- To consider projective solutions of polynomials – i.e. in order to tell that $[X, Y, Z]$ is a solution – with zero (X, Y, Z) also (tX, tY, tZ) for all $t \neq 0$ must be a zero.

Homogeneous polynomial

Definition 8.3 A polynomial $F(X,Y,Z) = \sum a_r X^i Y^j Z^k$ of degree d is called homogeneous if always $i+j+k=d$ for all $a_r \neq 0$.
A homogeneous polynomial fulfills

$$F(tX,tY,tZ) = t^d F(X,Y,Z),$$

i.e. it is possible to consider its solutions in the projective plane.

8.1.4 Plane curves

Definition 8.4 Let $f \in \mathbb{F}[x, y]$ be a polynomial over the field $\mathbb{F}$.

- The zeros of $f(x, y)$ define an affine plane curve

$$C_f(\mathbb{F}) = \{|(x, y) \in \mathbf{A}^2(\mathbb{F})|f(x, y) = 0\}$$

- $(a, b) \in \mathbf{A}^2(\mathbb{F})$ is a singular point of $C_f(\mathbb{F})$,if
  - (a, b) is a point on $C_f(\mathbb{F})$, $(a, b) \in C_f(\mathbb{F})$, i.e. $f(a, b) = 0$.
  - Both derivatives of f vanish in (a, b), i.e.

$$\frac{\delta f}{\delta x}(a, b) = \frac{\delta f}{\delta y}(a, b) = 0$$

- $C_f(\mathbb{F})$ is singular over $\mathbb{F}$, if it has a singular point $(a, b) \in \mathbf{A}^2(\mathbb{F})$.
- $C_f(\mathbb{F})$ is non-singular if it has no singular point, even over the algebraic closure.

Definition 8.5 Let $F \in \mathbb{F}[X, Y, Z]$ be a homogeneous polynomial over the field $\mathbb{F}$.

- The zeros of $F(X, Y, Z)$ define a homogeneous plane curve

$$C_F(\mathbb{F}) = \{[X, Y, Z] \in \mathbf{P}^2(\mathbb{F}) | F(X, Y, Z) = 0\}$$

- $[A, B, C] \in \mathbf{P}^2(\mathbb{F})$ is a singular point of $C_F(\mathbb{F})$, if $[A, B, C] \in C_F(\mathbb{F})$ and

$$\frac{\delta F}{\delta X}(A, B, C) = \frac{\delta F}{\delta Y}(A, B, C) = \frac{\delta F}{\delta Z}(A, B, C) = 0$$

Moving affine to projective coordinates

- Let $f \in \mathbb{F}[x, y]$. With

$$x = \frac{X}{Z} \text{ and } y = \frac{Y}{Z}$$

we move $f(x, y)$ into its homogeneous form $F(X, Y, Z)$.

8.2 Elliptic curves

Definition 8.6 Let $\mathbb{F}$ be a field and

$$F(X,Y,Z) = Y^2Z + a_1XYZ + a_3YZ^2 - X^3 - a_2X^2Z - a_4XZ^2 - a_6Z^3$$

be a homogeneous polynomial of degree 3 with $a_1, a_2, a_3, a_4, a_6 \in \mathbb{F}$. In case the projective curve

$$C_F(\mathbb{F}) = \{[X,Y,Z] \in \mathbf{P}^2(\mathbb{F}) | F(X,Y,Z) = 0\}$$

is non-singular, then $C_F(\mathbb{F})$ is an elliptic curve in homogeneous form.
The Weierstraßequation of C_F is

$$C\colon Y^2Z + a_1XYZ + a_3YZ^2 = X^3 + a_2X^2Z + a_4XZ^2 + a_6^Z 3$$

The point at infinity

- The affine part of $C_F(\mathbb{F})$ are points $[X, Y, 1]$, i.e. points on $C_f(\mathbb{F})$ with

$$f(x, y) = y^2 + a_1xy + a_3y - x^3 - a_2x^2 - a_4x - a_6.$$

- The projective part of $C_F(\mathbb{F})$ are points $[X, Y, 0]$.
- Let $[X, Y, 0] \in \mathbf{P}^2(\mathbb{F})$. Plugged into C, one obtains $X^3 = 0$ and $Y \neq 0$, i.e. a triple zero and we define

$$\mathcal{O} = [0, Y, 0] = [0, 1, 0] \in C_F(\mathbb{F}).$$

- Therfore, $\mathcal{O} = [0, 1, 0]$ is the only non-affine point on an elliptic curve.

The point at infinity cntd.

- $\mathcal{O}$ is for all polynomials F a regular point of the curve $C_F(\mathbb{F})$ with
$$\frac{\delta F}{\delta Z}(0,1,0) = 1,$$
i.e. testing for singularity can be done in the affine world.
- When considering elliptic curves $\mathcal{O}$ is called the point at infinity and $\mathcal{O}$ is a rational point.
- $\mathcal{O}$ is also:
  - the projective intersection of vertical lines,
  - a point of inflection of $C_F(\mathbb{F})$.

Observation:

- There are different elliptic curves which have the same rational points.

Task: Find the following elements:

- A transformation that projects the rational points of two elliptic curves bijectively on each other.
- An elliptic curve of simple form, i.e. with few terms, onto which the rational points of all/many curves can be projected.

Weierstraßnormal form

Result: Weierstraßnormal form

- Curves in the original and in the Weierstraß normal form have different shapes.
- Examining rational points on elliptic (in general on cubic) curves can be reduced to Weierstraß normal form.

We will now show:

- The Weierstraß normal form of an elliptic curve depends on the characteristic of the base field.
- The larger the characteristic of the base field the easier the Weierstraß normal form.

Theorem 8.7 (Weierstraß reduction) Let $C_F(\mathbb{F})$ be an elliptic curve over the field $\mathbb{F}$ with

$$F(X,Y,Z) = Y^2Z + a_1XYZ + a_3YZ^2 - X^3 - a_2X^2Z - a_4XZ^2 - a_6Z^3$$

i) Let char $\mathbb{F} = 2$ and $a_1 \neq 0$. Then

a) The transformation $\Gamma_1 : \mathbf{P}^2(\mathbb{F}) \to \mathbf{P}^2(\mathbb{F})$ with

$$\Gamma_1([X,Y,Z]) = \left[\frac{X}{a_1^2} - \frac{a_3Z}{a_1^3}, \frac{Y}{a_1^3} - \frac{(a_1^2a_4 + a_3^2)Z}{a_1^6}, Z\right]$$

is bijective.

b) Let

$$G_1(X,Y,Z) = Y^2Z + XYZ - X^3 - b_2X^2Z - b_6Z^3$$

with

$$b_2 = \frac{a_3 + a_1a_2}{a_1^3} \text{ and}$$

$$b_6 = \frac{a_1^6a_6 + a_1^5a_3a_4 + a_1^4a_2a_3^2 + a_1^4a_4^2 + a_1^3a_3^3 + a_3^4}{a_1^{12}}.$$

Then we have

$$\Gamma_1(C_F(\mathbb{F})) = C_{G_1}(\mathbb{F})$$

The Weierstraß normal form of an elliptic curve with new coefficients over a field with characteristic 2 is

$$Y^2Z + XYZ = X^3 + a_2X^2Z + a_6Z^3 \text{(homogeneous)}$$

$$y^2 + xy = x^3 + a_2x^2 + a_6 \text{(affine)}$$

i.e. $a_1 = 1$ and $a_3, a_4 = 0$.

Theorem Weierstraß reduction cntd.

ii) Let char $\mathbb{F} \neq 2$. Then

a) The transformation $\Gamma_2 \colon \mathbf{P}^2(\mathbb{F}) \to \mathbf{P}^2(\mathbb{F})$ with

$$\Gamma_2([X, Y, Z]) = \left[X, Y + \frac{a_1 x}{2} + \frac{a_3 Z}{2}, Z\right]$$

is bijective.

b) Let

$$G_2(X, Y, Z) = Y^2 Z - X^3 - \frac{1}{4} c_2 X Z^2 - \frac{1}{4} c_6 Z^3$$

with $c_2 = a_1^2 + 4a_2$, $c_4 = 2a_4 + a_1 a_3$ and $c_6 + a_3^2 + 4a_6$. Then we have

$$\Gamma_2(C_F(\mathbb{F})) = C_{G_2}(\mathbb{F})$$

The Weierstraß normal form of an elliptic curve over a field with characteristic not equal 2 is

$$Y^2 Z = X^3 + a_2 X^2 Z + a_4 X Z^2 + a_6 Z^3 \text{(homogeneous)}$$

$$y^2 = x^3 + a_2 x^2 + a_4 x + a_6 \text{(affine)}$$

i.e. $a_1, a_3 = 0$.

Theorem Weierstraß reduction cntd.

iii) Let char $\mathbb{F} \neq 2, 3$. Then

a) The transformation $\Gamma_3 \colon \mathbf{P}^2(\mathbb{F}) \to \mathbf{P}^2(\mathbb{F})$ with

$$\Gamma_3([X, Y, Z]) = [36X + 3c_2Z, 216Y, Z]$$

is bijective.

b) Let

$$G_3(X, Y, Z) = Y^2Z - X^3 + 27d_4XZ^2 + 54d_6Z^3$$

with $d_4 = c_2^2 - 24c_4$, $d_6 = -c_2^3 + 36c_2c_4 - 216c_6$. Then we have

$$\Gamma_3(C_{G_2}(\mathbb{F})) = C_{G_3}(\mathbb{F})$$

The Weierstraß normal form of an elliptic curve over a field with characteristic not equal 2 and 3 is

$$Y^2Z = X^3 + a_4XZ^2 + a_6Z^3 \text{(homogeneous)}$$
$$y^2 = x^3 + a_4x + a_6 \text{(affine)}$$

i.e. $a_1, a_2, a_3 = 0$.

Proof of Weierstraß normal form construction

The steps for the proof for (i),(ii) and (iii):

- Construction of a transformation Γ_i^{-1}, an inverse to Γ_i, e.g.

$$\Gamma_2^{-1}([X,Y,Z]) = \left[X, Y - \frac{a_1 X}{2} - \frac{a_3 Z}{2}, Z\right]$$

- Moving to G_i by substituting the variables in F (respectively G_2) according to Γ_i^{-1}, e.g.: Evaluate

$$F(X, Y - \frac{1}{2}a_1 X - \frac{1}{2}a_3 Z, Z)$$

  with the goal $G_2(X,Y,Z)$.

- Proof of non-singularity of $C_{G_i}(\mathbb{F})$ by computing partial derivatives, e.g. by using the chain rule.

$$\frac{\delta G_2}{\delta X}(X, YZ) = \frac{\delta F}{\delta X}(\Gamma_2^{-1}(X,Y,Z)) - \frac{a_1}{2}\frac{\delta F}{\delta Y}((X,Y,Z))$$

Definition 8.8 Let $C_F(\mathbb{F})$ with

$$F(X,Y,Z) = Y^2Z + a_1XYZ + a_3YZ^2 - X^3 - a_2X^2Z - a_4XZ^2 - a_6Z^3$$

and

$$\begin{aligned} b_2 &= a_1^2 + 4a_2 \\ b_4 &= 2a_4 + a_1a_3 \\ b_6 + a_3^2 &+ 4a_6 \\ b_8 &= a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2 \end{aligned}$$

Discriminant and j-Invariant cntd.

i) The discriminant of the elliptic curve $C_F(\mathbb{F})$ is

$$\Delta = -b_2^2 b_8 - 8b_4^3 - 27b_6^2 + 9b_2 b_4 b_6.$$

ii) The j-Invariant of $C_F(\mathbb{F})$ is

$$j = \frac{(b_2^2 - 24b_4)^3}{\Delta}$$

- The j-Invariant determines the isomorphism class of an elliptic curve over the algebraic closure.
- The discriminant gives information about the singularity of an elliptic curve.

Theorem 8.9 $C_F(\mathbb{F})$ is singular if and only if, $\Delta = 0$.
Proof: Exercise.

8.3 The group law

Operation on points of an elliptic curve

We can define an operation on the set of points of an elliptic curve which leads to an abelian group.
Ways to introduce the operation:

- Using algebra only: not very descriptive,
- traditional geometry (affine): motivation incomplete,
- projective geometry: combines clear description with analytics.

8.3.1 Projective lines

Projective lines

- A projective line $L(a, b, c)$ is a projective curve $C_F(\mathbb{F})$ with

$$F(X, Y, Z) = aX + bY + cZ \text{ and } (a, b, c) \neq (0, 0, 0).$$

- The affine reduction of $L(a, b, c), l(a, b, c)$ is $C_F(\mathbb{F})$ with

$$f(x, y) = ax + by + c.$$

- If $(a, b) \neq (0, 0)$, then $l(a, b, c)$ is a line in the affine plane.
- Otherwise, $L(a, b, c)$ has no affine part and $l(a, b, c)$ is empty.

Properties of projective lines

- A projective line is non-singular if there is no point in which all partial derivatives vanish simultaneously.
- Given two projective points, there is exactly one projective line that runs through both of them.
  - Let $P_1 = [x_1, y_1, z_1]$ and $P_2 = [x_2, y_2, z_2]$, $P_1 \neq P_2$.
  - Plugging these points into a projective line results in two linearly independent equations

$$ax_1 + by_1 + cz_1 = 0$$
$$ax_2 + by_2 + cz_2 = 0$$

  which have a linear solution space.

Intersection of projective lines

- Two projective lines intersect always in one point:
  - i) The lines have an affine intersection point.
  - ii) The lines do not have an affine intersection, i.e. are parallel in the affine space:
    - Consider two parallel affine lines

$$f_1(x, y) = y - ax$$
$$f_2(x, y) = y - ax - c$$

and their projctive description:

$$F_1(X, Y, Z) = Y - aX$$
$$F_2(X, Y, Z) = Y - aX - cZ$$

    - F_1 and F_2 intersect in $P = [1, a, 0]$, i.e. at infinity in the affine space.

8.3.2 Points of intersection of projective lines and curves

Tangent in a point

Definition 8.10 Let $P = [a, b, c] \in C_F(\mathbb{F})$ be a non-singular point on the projective curve $C_F(\mathbb{F})$, i.e. not all partial derivatives vanish in P. The tangent in P on $C_F(\mathbb{F})$ is the line

$$L\left(\frac{\delta F}{\delta X}(a,b,c), \frac{\delta F}{\delta Y}(a,b,c), \frac{\delta F}{\delta Z}(a,b,c)\right)$$

Definition 8.11 Let $P_1, P_2 \in L(a,b,c), P_1 = [x_1, y_1, z_1], P_2 = [x_2, y_2, z_2]$. The multiplicity of the intersection of $L(a,b,c)$ and $C_F(\mathbb{F})$ in P_1, denoted as

$$m(P, L(a,b,c), C_F(\mathbb{F}))$$

is the multiplicity of a zero in $t = 0$ of the polynomial

$$g(t) = F(x_1 + tx_2, y_1 + tY_2, z_1 + z_2)$$

Multiple zeros cntd.

- If P_1 is not on $C_F(\mathbb{F})$, then $g(0) \neq 0$, i.e. $g(t)$ has no zero at 0.
- In case $L(a, b, c)$ is the tangent in P on $C_F(\mathbb{F})$, then

$$m(P, L(a, b, c), C_F(\mathbb{F})) \geq 2$$

  - $g(0) = 0$
  - The order of the zero is the smalles exponent of t in g(t).
  - If all derivatives up the the kth derivative are 0, then the order is k.
  - Continue with the chain rule.

Intersection of a line with an elliptic curve

Theorem 8.12 A projcetive line $L(a, b, c)$ and an elliptic curve $E(\mathbb{F})$ do not intersect, intersect once, or intersect three times, i.e.

$$\sum_{P \in \mathbf{P}^2(\mathbb{F})} m(P, L(a, b, c), E(\mathbb{F})) = 0 \text{ or } 1 \text{ or } 3$$

Proof idea: case distinction for a and b.

Intersection of a line with an elliptic curve

Corollary 8.13 The line, that runs through two points on an elliptic curve, also intersects this curve in a third point.

Corollary 8.14 The tangent in a point on an elliptic curve, also intersects this curve in another point.

8.3.3 The group operation

- Let $E(\mathbb{F}) = C(\mathbb{F})$ with:

$$F(X,Y,Z) = Y^2Z + a_1XYZ + a_3YZ^2 - X^3 - a_2X^2Z - a_4XZ^2 - a^6Z^3$$

Definition 8.15 Let $P, Q \in E(\mathbb{F})$

i) If $P \neq Q$ then let
   - L_1 be the line through P and Q,
   - R the third intersection point of L_1 with $E(\mathbb{F})$, i.e. $E(\mathbb{F}) \cap L_1 = \{P, Q, R\}$.

ii) If $P = Q$ then let
   - L_1 be the tangent in P,
   - R is the intersection point of L_1 with $E(\mathbb{F})$, i.e. $E(\mathbb{F}) \cap L_1 = \{P, R\}$.

Intersection with $\mathcal{O}$

Consider $\mathcal{O}$:

i) If $R \neq \mathcal{O}$ then let
- L_2 be the line through R and $\mathcal{O}$,
- S the third intersection point of L_2 with $E(\mathbb{F})$, i.e. $E(\mathbb{F}) \cap L_2 = \{R, \mathcal{O}, S\}$.

ii) If $R = \mathcal{O}$ then let
- L_2 be the tangent in $\mathcal{O}$,
- S the intersection point of L_2 with $E(\mathbb{F})$, i.e. $E(\mathbb{F}) \cap L_2 = \{\mathcal{O}, S\}$.

Using this we define

$$S = P + Q, \text{ if } P \neq Q$$
$$S = P + P = 2P, \text{ if } P = Q$$

Theorem 8.16 According to Definition 8.15 $(E(\mathbb{F}), +)$ is an abelian group. The neutral element is $\mathcal{O}$.

8.3.4 Addition formula

Addition with non-affine points

- The only non-affine point is $\mathcal{O}$.
- One can easily check

$$\mathcal{O} + P = P$$
$$\mathcal{O} + \mathcal{O} = \mathcal{O}$$

and with $P = -Q$ it holds $P + Q = \mathcal{O}$.

Limiting addition to the affine part

With the above observation we can now limit the addtion to the affine part, i.e. to $E(\mathbb{F}) = C_f(\mathbb{F})$ with

$$f(x, y) = y^2 + a_1xy + a_3y - x^3 - a_2x^2 - a_4x - a_6$$

Find explicit coordinates of

- $P + Q$ with $P \neq Q$ and $P \neq -Q$: addition formula,
- $P + P = 2P$: point doubling,

for elliptic curves over

- arbitrary fields,
- fields $\mathbb{F}$ with $\mathrm{char}\mathbb{F} \neq 2, 3$.

Theorem 8.17

i) Let $P = (x, y) \in E(\mathbb{F})$. Then

$$-P = (x, -y - a_1 x - a_3)$$

ii) Let $P_1, P_2 \in E(\mathbb{F}), P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$, s.t. $P_1 \neq P_2$ and $P_1 \neq -P_2$. Then

$$P_3 = (x_3, y_3) = P_1 + P_2$$

with

$$x_3 = \lambda^2 + a_1\lambda - a_2 - x_1 - x_2$$
$$y_3 = -(\lambda - a_1)x_3 - \nu - a_3$$

and

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$$
$$\nu = \frac{y_1 x_2 - y_2 x_1}{x_2 - x_1}$$

iii) Let $P = (x, y) \in E(\mathbb{F})$. Then

$$2P = (x', y')$$

with

$$x' = \lambda^2 + a_1\lambda - a_2 - 2x$$
$$y' = -(\lambda - a_1)x' - \nu - a_3$$

and

$$\lambda = \frac{3x^2 - 2a_2x + a_4 - a_1y}{2y + a_1x + a_3}$$
$$\nu = \frac{-x^3 + a_4x + 2a_6 - a_3y}{2y + a_1x + a_3}$$

Theorem 8.18 Let $E(\mathbb{F}) = C_f(\mathbb{F})$ with

$$f(x, y) = y^2 - x^3 - ax - b$$

i) Let $P = (x, y) \in E(\mathbb{F})$. Then

$$-P = (x, -y)$$

Addition in fields with char $\neq 2, 3$ contd.

ii) Let $P_1, P_2 \in E(\mathbb{F}), P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$, s.t. $P_1 \neq P_2$ and $P_1 \neq -P_2$. Then

$$P_3 = (x_3, y_3) = P_1 + P_2$$

with

$$x_3 = \lambda^2 - x_1 - x_2$$
$$y_3 = \lambda(x_1 - x_3) - y_1$$

and

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$$

Addition in fields with char $\neq 2, 3$ contd.

iii) Let $P = (x, y) \in E(\mathbb{F})$. Then

$$2P = (x', y')$$

with

$$x' = \lambda^2 - 2x$$
$$y' = \lambda(x - x') - y$$

and

$$\lambda = \frac{3x^2 + a}{2y}$$

Example 1

Let $y^2 = x^3 + 17$, $P_1 = (-1, 4)$ and $P_2 = 2, 5$.

- Line through P_1 and P_2:

$$y = \frac{1}{3}x + \frac{13}{3}$$

This results in:

$$x_3 = -\frac{8}{9} \text{ and } y_3 = \frac{109}{27}$$

Hence,

$$P_1 + P - 2 = \left(-\frac{8}{9}, -\frac{109}{27}\right)$$

- Tangent on P_1:

$$\lambda = \frac{f'(-1)}{8} = \frac{3}{8}$$

Therefore,

$$2P_1 = \left(\frac{137}{64}, -\frac{2651}{512}\right)$$

Example 2

Let $y^2 = x^3 + x + 1 \in \mathbb{F}_5$. One obtains by testing all possibilities

$$E(\mathbb{F}_5) = \{\mathcal{O}, (0, \pm 1), (2, \pm 1), (3, \pm 1), (4, \pm 2)\}.$$

Hence, $E(\mathbb{F}_5)$ is an abelian group of order 9.

- Must be cyclic or the direct product of two groups of order 3.
- Decision is made via the group table.
- Let $P = (0, 1)$, then

  $$2P = (4, 2), 3P = (2, 1), 4P = (3, -1) \text{ etc.}$$

  therfore $E(\mathbb{F}_5)$ is cyclic of order 9.
- The points $(2, \pm 1)$ have order 3, all others, except for $\mathcal{O}$ have order 9.

8.3.5 Further research on elliptic curves

Further topics with elliptic curves

- Avoiding bad curves.
- Determining good curves.
- Efficient implementation.
- Embedding into cryptographic protocols.

Uses of elliptic curves

- Factoring.
- Prime number proofs.
- Keep your eyes open in all areas of mathematics:
  - A natural number n is *congruent*, if it is the area of a right triangle with rational sides, i.e. there are rational numbers a, b, and c, s.t.
$$n = \frac{1}{2}ab \text{ and } a^2 + b^2 = c^2$$
    - Fermat: 1,2 and 3 are not congruent.
    - Fibonacci: 5 and 6 are congruent.
    - Euler: 7 is congruent.
    - General result: Tunnell (1983) with elliptic curves.
  - Fermats Last Theorem: Proven by Wiles(1995) using elliptic curves.
- Further methodes.
- Hyperelliptic curves.