Security Notions

密碼學與應用

海洋大學資訊工程系
丁培毅

Unbreakable Cryptosystems ???

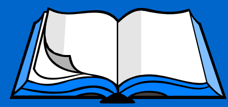
- Almost all of the practical cryptosystems are theoretically breakable given the time and computational resources.

- However, there is one system which is even theoretically unbreakable (perfectly secure): **One-time pad**.

One-time pad (Vernam Cipher)

- A kind of stream cipher
- Gilbert Vernam in 1918

shared secret
codebook



1010100 …

Encryption Key

Decryption Key

Alice — plaintext … 0101101 → $\oplus$ — ciphertext …1111001 → $\oplus$ — plaintext … 0101101 → Bob

Encrypt

Decrypt

- Nothing more about the plaintext can be deduced from the ciphertext, i.e., probability: Pr[M|C] = Pr[M] or entropy H(M|C) = H(M)
- Information-theoretical bound: for any efficient adversarial algorithm $\mathcal{A}$, Pr[$\mathcal{A}$(C)=M]=1/2.

Unbreakable Cryptosystems!!!

- One-time pad requires exchanging key that is as long as the plaintext.
- Security of one-time pad relies on the condition that keys are generated using truly random sources.
- However impractical, it is still being used in certain applications which necessitate very high-level security. Also, the "**masked by the random key**" structure is used everywhere.

Modern Cryptography

- Perfect security: possession of the ciphertext is not adding any new information to what is already known
- There may be useful information in a ciphertext, but if you can't compute it, the ciphertext hasn't really given you anything.

traditional cryptography $\Rightarrow$
modern cryptography (considering
computational difficulties of the adversary)

Modern Cryptography

- What tasks, were the adversary to accomplish them, would make us declare the system insecure?
- What tasks, were the adversary unable to accomplish, would make us declare the scheme secure?
- It is much easier to think about insecurity than security.

traditional cryptography $\Rightarrow$
modern cryptography (considering provably secure)

Provably Secure Scheme

- Provide evidence of computational security by reducing the security of the cryptosystem to some well-studied problem thought to be difficult (e.g., factoring or discrete log).
  - An encryption scheme based on some atomic primitives
  - Take some goal, like achieving privacy via encryption
  - Define the meaning of an encryption scheme to be secure
  - Choose an adversarial model with suitable capability
  - Provide a reduction statement, which shows that the only way to defeat the scheme is to break the underlying atomic primitive

Security Goals of Encryption

Various Security Definitions: ‘breakable?’

- Perfect security
- Plaintext recovery
- Key recovery
- Partial information recovery:
  - – Message indistinguishability
  - – Semantic Security
- Non-malleability
- Plaintext awareness

information-theoretically secure

Computationally secure
& provably secure

Security Goals (cont'd)

- Ex: leaking partial information about
  "buy" or "sell" a stock
  n bits, one bit per stock, 1:buy, 0:sell
  if any one bit were revealed,
  the adversary knows what I like to do.
- Changing format might avoid the above attack. However, making assumptions, or requirements, on how users format data, how they use it, or what the data content should be, is a bad and dangerous approach to secure protocol designs.

Security Goals (cont'd)

- **Simulation paradigm**: a scheme is **secure** if 'whatever a feasible adversary can obtain after attacking it, is also feasibly attainable from scratch'.

- **Semantic security**: Whatever can be obtained from the ciphertext can be computed without the ciphertext
- **Non-malleability**: Given a ciphertext, an adversary cannot produce a different ciphertext that decrypts to meaningfully related plaintext
- **Plaintext awareness**: an adversary cannot create a ciphertext y without knowing its underlying plaintext x

Adversary Models for Encryption

- Ciphertext Only
- Known Plaintext
- Chosen Plaintext
- Non-adaptive Chosen Ciphertext
- Adaptive Chosen Ciphertext

Security Goals for Signature

stingent

- **Total break** : key recovery
- **Universal forgery** : finding an efficient equivalent algorithm to produce signatures for arbitrary messages
- **Selective forgery** : forging the signature for a particular message chosen a priori by the attacker
- **Existential forgery** : forging at least one signature

Adversary Models for Signature

- **Key-only attack** : no-message attacks
- **Known-message attack**

powerful

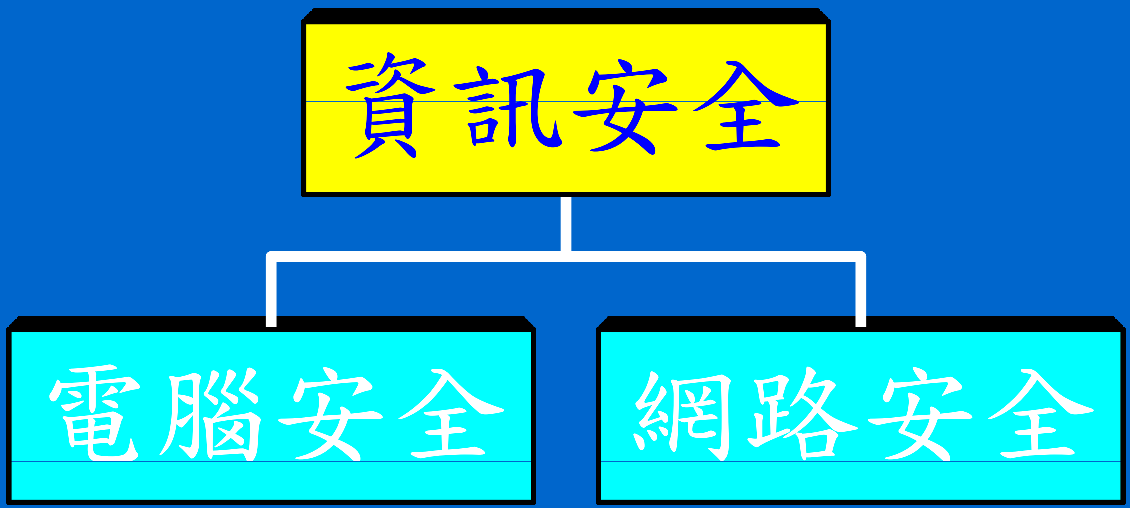
- **Generic chosen-message attack** : non-adaptive, messages not depending on public key
- **Directed chosen-message attack** : non-adaptive, messages depending on public key
- **Adaptive chosen-message attack** : messages depending on the previously seen signatures

Security Notion for Secure Protocols

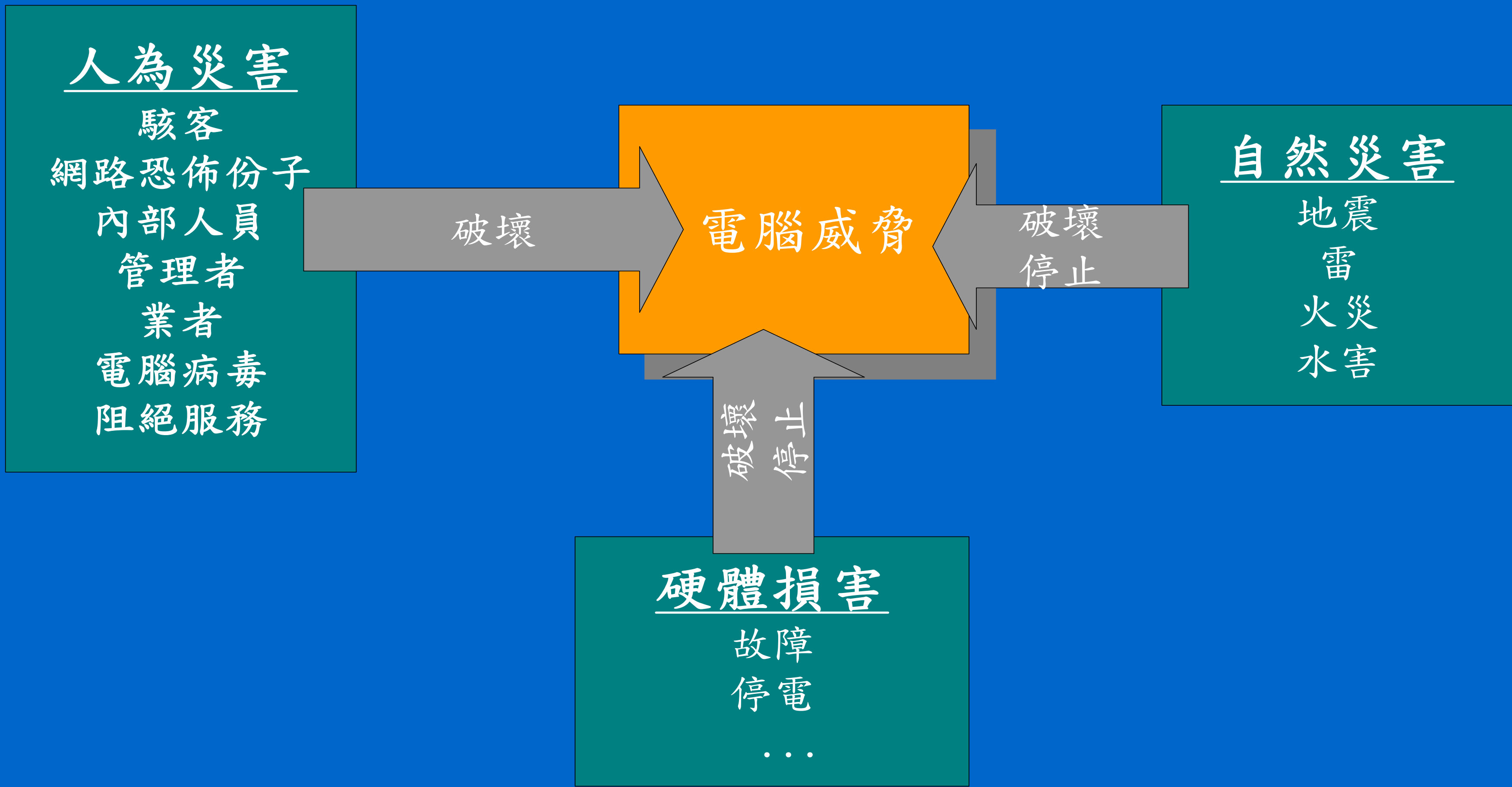
- Whatever can be obtained by a group of participants (including the adversary) during a real world protocol can also be calculated in the ideal model in which a trusted party helps every participant reaching his functional and security goals.

資訊安全的定義

- 資訊安全:利用各種方法及工具以保護靜態資訊(電腦安全)或動態資訊(網路安全)

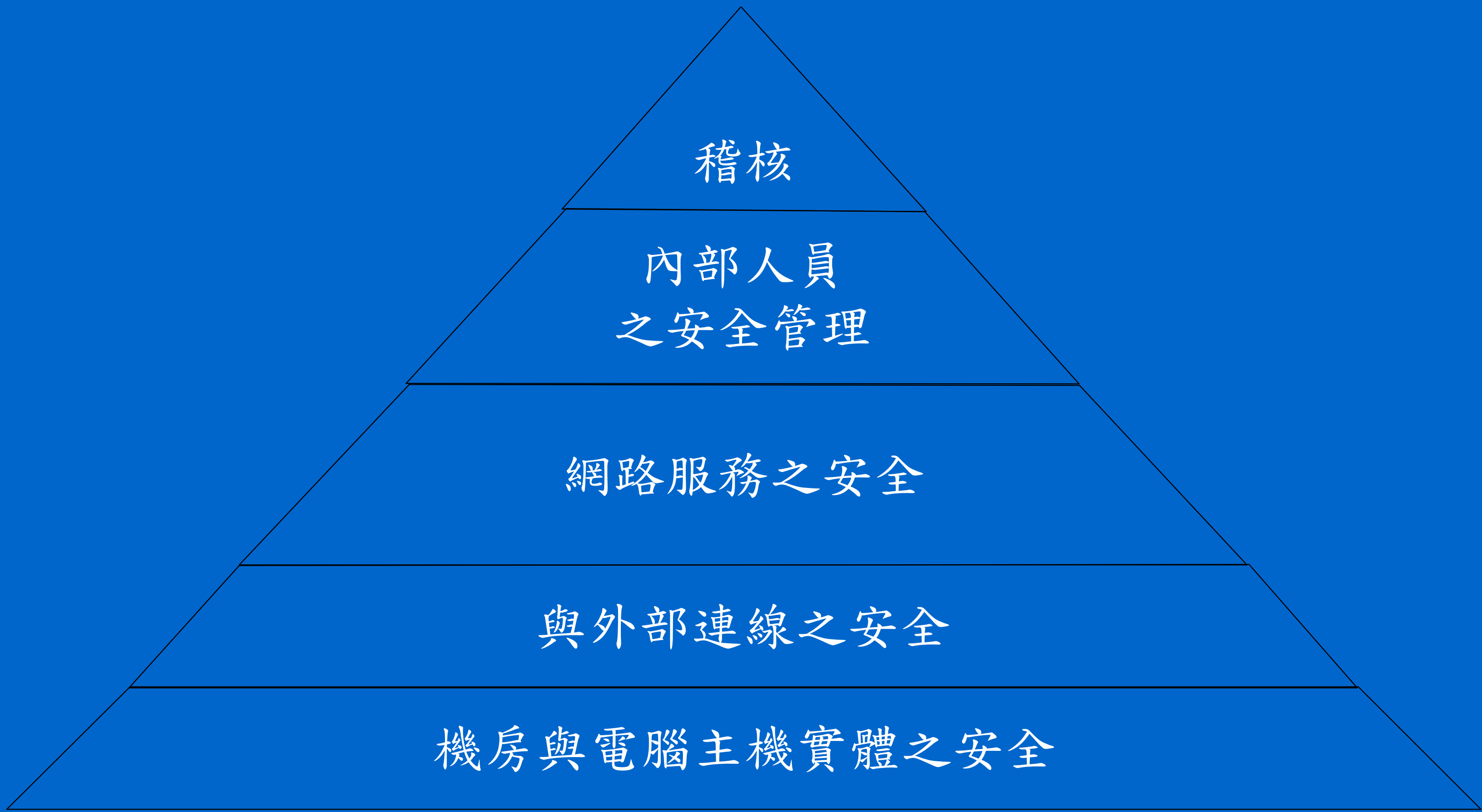


電腦安全的威脅



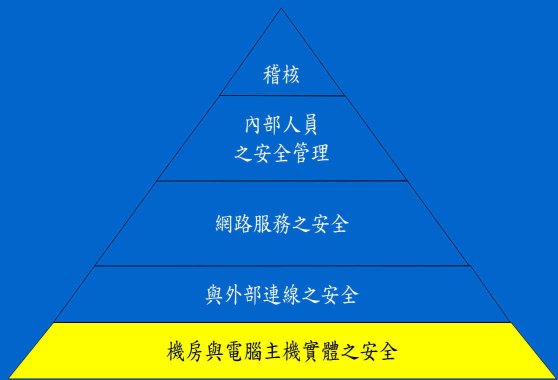
人為災害
駭客
網路恐佈份子
內部人員
管理者
業者
電腦病毒
阻絕服務
破壞
電腦威脅
破壞
停止
自然災害
地震
雷
火災
水害
破壞
停止
硬體損害
故障
停電
. . .

資訊安全課題分析



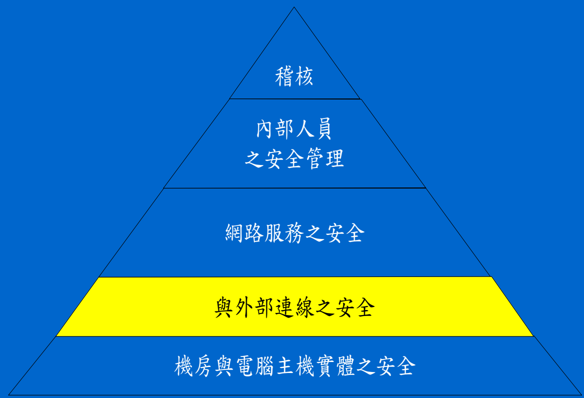
機房與電腦主機實體之安全

- 避免大自然(如水災、雷擊等)各種自然災害的危害
- 建築安全
- 避免硬體設備受到無法預測因素(如停電、 地震等)的傷害
- 備份（必須以距離隔離）
- 實體安全
- 備用電源（發電機，UPS等）



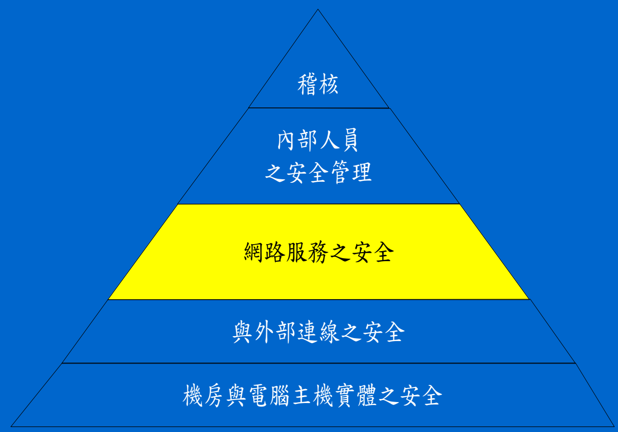
與外部連線之安全

- 利用密碼器、電子簽章及識別協定等資訊安全技術建立安全之通道及使用者連線之認證機制
- 保護自己在與外部連線通訊之隱私性及認證性



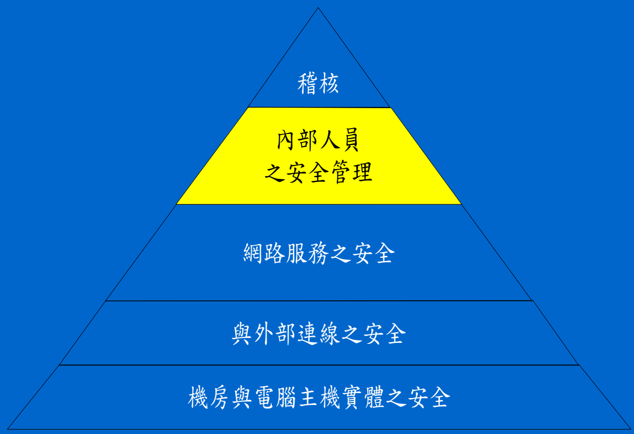
網路服務之安全

- 避免遭外部駭客之入侵及病毒之散播
- 確保網路能正常服務
- 定期安全健康檢查
- 危機應變處理



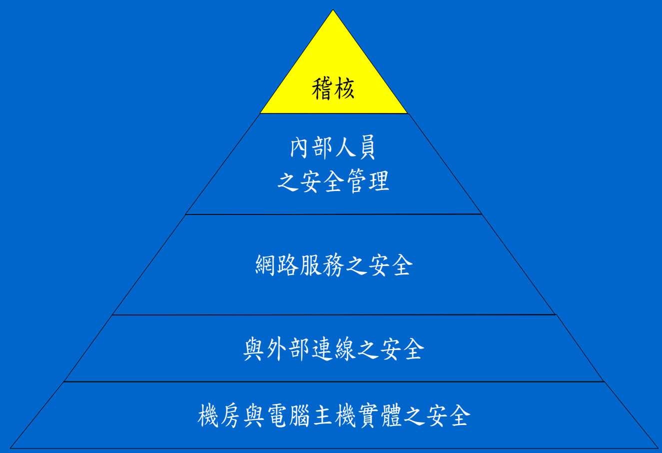
內部人員之安全管理

- 員工、管理者及電腦管理者應有不同的存取權限，以避免內部人員對機密資訊的危害
- 加強人員的資訊安全教育
- 關閉離職員工的存取權限
- 人員違反安全政策的處理



稽核

- 詳細制定安全政策並確保安全政策及措施能順利進行
- 持續保護與追蹤



Fundamental Cryptographic Services

- Confidentiality
  - Hiding the contents of the messages exchanged in a transaction
- Authentication
  - Ensuring that the origin of a message or the identity is correctly identified
- Integrity
  - Ensuring that only authorized parties are able to modify computer system assets and transmitted information
- Non-repudiation
  - Requires that neither of the authorized parties deny the aspects of a valid transaction

Cryptographic Applications

- **Digital Signatures:** allows electronically sign (personalize) the electronic documents, messages and transactions
- **Identification / authentication:** replace password-based authentication methods with more powerful (secure) techniques.
  - Identification: presenting the unique identity
  - Authentication: associate the individual with his unique identity by something he knows, something he possesses and some specific features of him

Cryptographic Applications

- **Key Establishment:** To communicate a key to your correspondent (or perhaps actually mutually generate it with him) whom you have never physically met before.
- **Secret Sharing:** Distribute the parts of a secret to a group of people who can never exploit it individually.
- **Zero Knowledge Proof**: Peggy proves to Victor that she has a particular knowledge without letting Victor learn the knowledge throught the interaction.

Cryptographic Applications

- **E-commerce:** carry out the secure transaction over an insecure channel like Internet.
- **E-cash / E-contract**
- **E-voting / E-auction**
- **Games**
- **Anonymous secret broadcast and tracing**
- **Stenography (digital watermarking)**
- **Software protection (IPR)**
- **Crypto currency & Blockchain**

Focus of this course

- Analysis of the fundamental primitives and protocols
- Security of the fundamental primitives and protocols

Why Staying in This Class???

- Most of the time in the future you won't be coding the cryptography primitives.
- You will be using these cryptography primitives (as they are from the software libraries or packages).
- Why do you need to stay in this class to understand the background materials of these primitives?

Why Staying in This Class???

- CATCHES: the usage of these primitive has to follow strict security notions
  – insecure SSL mechanism ==> TLS
  – 2002 MSIE SSL implementation faults
  – most textbook's plain RSA and ElGamal system is insecure without preprocessing

Why Staying in This Class???

– Double DES
– Symmetric encryption with ECB mode
– Chosen ciphertext attacks on CBC / OFB / CFB / Counter mode of DES/AES
– Subliminal channels
– Signature scheme without non-repudiation
– SSH (Secure SHell) Authentication & Encryption
– SSL Authentication

Why Staying in This Class???

- Standards would be established on most cryptographic primitives. These primitives will be at your disposal when you design your application systems.
- You need to understand clearly these primitives in order to design any customized secure protocol.
- You need to follow the 'provably security' methodology to base your protocols on the security guarantees of the underlying primitives.

Aspects of Modern Cryptography

- One way function assumption
- Model adversaries such that they need to solve computationally intractable problems
- Refined security definitions
- Provably secure methodology
- Reduce intractability assumptions
- Reduce trust assumptions
- Reduce physical assumptions

Quantum Computer

- Peter Shor 1994
- Both number factoring and discrete log problems can be solved in probabilistic polynomial time (actually linear) if the quantum computer of sufficient cubits (e.g 2048) were built successfully.
- There are some physical phenomenon at the atom level, which will change its state when being measured in any way.

Goal of Modern Cryptography

- Create schemes (protocols) that are easy to operate (properly) but hard to foil!

Complexity Classes

- P: problems that can be solved by an algorithm with computation complexity **O(p(n))**
  - ex. Bubble sort $O(n^2)$ Quick sort O(n logn)
  - there are many problems which are not P
    - ex. 2^n knapsack(subset sum)
    - n! Travelling Salesman Problem (TSP)
    - unsolvable halting problem
- NP: decision problems that have solutions which can be verified by a polynomial time algorithm (problems that might still have polynomial time solutions) ex. decision-TSP, Satisfiability (SAT), knapsack, Factoring, ...

Complexity Classes

- NP-hard:
  - all NP problems have a poly-time mapping reduction to them. Once you have a poly-time solution for any one of NP-hard problems, you have a poly-time solution for every NP problem. However, an NP-hard problem itself might not be an NP problem. Usually, a problem is NP-hard if you find an NP-complete problem that reduces to it.
  - ex. search-TSP, SVP, TQBF, halting problem (unsolvable)
- NP-complete:
  - Def 1: NP problems, all NP problems can be reduced to them
  - Def 2: NP problems, to which SAT can be reduced
  - Def 3: NP $\cap$ NP-Hard
  - ex. SAT, decision-TSP, G3C, Knapsack ...

Complexity Classes

- **reduction**

$$P_1 \leq_T P_2$$

means "if P_2 were solved by a poly-time algorithm $\mathcal{A}$, P_1 can also be solved by calling poly-times of the same algorithm $\mathcal{A}$" or equivalently "if P_1 is unsolvable polynomially, P_2 is also unsolvable polynomially".