Privacy

• Protect **confidential** information from non-authorized use

• Protection Mechanisms: *Crypto, Access control, ...*

• Problem: information leakage in many applications is **unavoidable**
I. Leakage that happens intentionally

- eg: extract statistics from a dataset

- Problem: inference of personal information

- eg: “what is the median age of cancer patients”
II. Leakage due to side channels

- ge: OpenSSL timing attack [BonehBrumley03]

- Also: cache misses, power, radiation, faults, …

- Completely preventing such channels is impossible
III. Leakage in exchange to **efficiency**

- eg: Anonymous Communication Systems (Crowds, Onion Routing, …)

  ![Diagram showing anonymous communication systems]

  - **Strong anonymity** is achievable (eg Dining Cryptographers)
  - But such systems are not efficient!
IV. Leakage in exchange to a service

- eg: Location Based Services
  - Points Of Interest / Dating / social networks / games / …
- Privacy issues: tracking, profiling, identification
Quantitative Information Flow

How can we measure information leakage?
Plan of the course

• Motivation, application examples
• Secrets and vulnerability
• Channels and leakage
• Robustness and capacity
• Comparing systems, the lattice of information
• Applications and exercises
Example: analysis of a password checker

Password checker 1

Password: $K_1 K_2 \ldots K_N$
Input by the user: $x_1 x_2 \ldots x_N$
Output: $out$ (Fail or OK)

Intrinsic leakage

By learning the result of the check the adversary learns something about the secret

```plaintext
out := OK
for i = 1, ..., N do
  if $x_i \neq K_i$ then
    out := FAIL
  end if
end for
```
Example: analysis of a password checker

Password checker 2

Password: $K_1K_2\ldots K_N$
Input by the user: $x_1x_2\ldots x_N$
Output: $\text{out}$ (Fail or OK)

Side channel attack

If the adversary can measure the execution time, then he can also learn the longest correct prefix of the password
The Dining Cryptographers protocol

Example of Anonymity Protocol: DC Nets [Chaum’88]

- A set of nodes with some communication channels (edges).
- One of the nodes (source) wants to broadcast one bit \( b \) of information.
- The source (broadcaster) must remain anonymous.
The Dining Cryptographers protocol

Chaum’s solution

• Associate to each edge a fair binary coin

• Toss the coins

• Each node computes the binary sum of the incident edges. The source adds $b$. They all broadcast their results

• Achievement of the goal:
  Compute the total binary sum: it coincides with $b$
Observables: An (external) attacker can only see the declarations of the nodes

Question: Does the protocol protect the anonymity of the source?
Dining Cryptographers, limitations

- DC is **not practical** for a large number of users
- In practice we might want to **trade anonymity for efficiency**
- Crowds offers a **weaker** notion of anonymity called **probable innocence**
- Designed for **anonymous web surfing**
The Crowds protocol

The **initiator:**
- Forwards the message

A **forwarder:**
- With pb $p_f$ forwards
- With pb $1 - p_f$ delivers
- The path is used in the **opposite direction** for the reply
- The **same** path is used in **future** requests
The Crowds protocol: anonymity

• We consider sender anonymity

• Attacker model
  - **Cannot** see the whole network
  - Only messages **sent to him**

• The server:
  - only sees the **last user**
  - Strong anonymity is satisfied
Corrupted users:

- They can see **forwarding requests** and “detect” a user $i$
- User $i$ can still claim that he was **forwarding** the message for user $j$
- Is strong anonymity satisfied?

Compare the **probab. to detect** $i$:
- when $i$ is the payer
- when $j$ is the payer

- They are **different**: strong anonymity is violated
A **location-based system** is a system that uses geographical information in order to provide a service.

- Retrieval of Points of Interest (POIs).
- Mapping Applications.
- Deals and discounts applications.
- Location-Aware Social Networks.
Location-Based Systems

- Find restaurants within 300 meters.
- Hide location, not identity.
- Provide approximate location.
Solution: obfuscation

area of interest

reported position
Solution: obfuscation

area of retrieval

area of interest
Solution: obfuscation
Issues to study

How can we generate the noise?

What kind of formal privacy guarantees do we get?

Which mechanism gives optimal utility?

What if we use the service repeatedly?
Timing Attacks in Cryptosystems

Remote timing attack [BonehBrumley03]

1024-bit RSA key recovered in 2 hours from standard OpenSSL implementation across LAN

Response time depends on the key!
Timing Attacks in Cryptosystems

What counter-measures can we use?

Make the decryption time constant
   Too slow!

Force the set of possible decryption times to be small
   Is it enough?
   Must be combined with blinding
   Careful analysis of the privacy guarantees is required