Further techniques for Anonymity

Anonymity, further approaches

- Based on the idea “blending into a crowd”, that is hiding one’s actions within the actions of many others
- To execute a web transaction a user first joins a “crowd” of other users;
- Then the user’s request to a web server is passed to a random member of the crowd;
- That member can either submit the request to the server, or forward it to another randomly chosen member of the crowd and so on.

Privacy protection by the crowd

- When the request submitted to the end server, it is submitted by a random member of a crowd, so identity of an initiator is hidden (“in the crowd”) from an external observer
- Members of the crowd cannot identify initiator as well, they “just passing requests”
Crowds vs anonymizers and mixes

- Unlike an anonymizer crowds provide no single point, where an attacker can compromise anonymity of all users.
- Crowds does not provide anonymity against a global adversary able to oversee all communications. In contrast, mix-networks protect anonymity in that case.
- Crowds admit very efficient implementations in comparison with mixes: no encryption/decryption operations, no inflation of message lengths.

Tor: anonymity online

- Tor (from The onion routing project, originated in the US Naval Research Lab) – practical solution for anonymity protection: www.torproject.org
- It is free and open source and available for major platforms: Windows, Mac, Linux/Unix, Android
- Can be used for web browsing and instant messaging, prevents people from learning your location or browsing habits

Tor: main principles

- A combination of (variants of) mix-network and crowd mechanisms
  - Using a set of relay nodes (~crowd); currently >6000
  - Routing using random choice (similar to crowds)
  - Encrypted connections between neighbouring nodes (similar to mix-networks)
  - It uses public key cryptography to share the secret keys between neighbouring nodes ⇒ uses the shared secret to perform symmetric encryption in further communications between neighbours
  - temporarily available virtual channels

DC-networks

- D.Chaum, 1988: Dining Cryptographer networks

  \[
  \begin{align*}
  &A \quad K_{AC} \quad K_{AB} \\
  &B \quad K_{BC} \\
  &C
  \end{align*}
  \]

- At the preliminary stage between some pairs of nodes (at the picture between all) secret keys (sequences of bits) are exchanged
**DC-networks**

- To send a message $M$ (sequence of bits), a node, say $A$, broadcasts the value $(M + 2^1 K_{AB} + 2^1 K_{AC})$, i.e. superposition of the message and all keys of $A$, here $+$ stands for bitwise addition modulo 2 (or XOR operation).
- All other nodes broadcast superpositions of all their keys. So, $B$ broadcasts $(K_{AB} + K_{BC})$ and $C$ broadcasts $(K_{AC} + K_{BC})$.
- All nodes then superpose all received messages and get $(M + 2^1 K_{AB} + 2^1 K_{AC} + 2^1 K_{AB} + 2^1 K_{BC} + 2^1 K_{AC} + 2^1 K_{BC}) = M$
- (the initial message !!!)

---

**Anonymity by DC-networks**

- DC-networks provide for sender anonymity because an adversary is unable to decide whether the packets he may observe contain a message or not;
- DC-networks can be used in combination with other mechanisms, such as mix-networks to enhance anonymity;
- A major drawback is that DC-Networks require the preliminary stage exchanging the secret keys between participants;
- Every round of communication requires a new set of keys;
- Every node needs to participate every time a message is broadcasted => high load on the nodes => impractical in large networks.

---

**Recent developments in DC-networks**

- Dissent system (~2012):
  - Scalable to thousands nodes
  - Client-server architecture with several servers and small groups served by a server
  - Retro-active blame mechanism to deal with jamming
  - XOR together with more complicated group multiplication operations are used
  - See further details at dedis.cs.yale.edu/dissent and links to further reading at web page of COMP528
Broadcast and receiver anonymity

- Broadcast itself is a way to protect anonymity of a receiver: sender just broadcasts a message to some group of users, including intended receiver;
- Of course, as such it does not protect the content of communications;
- Better way: a sender broadcasts a message, encrypted in a way that only intended receiver can decrypt;
- It can be done by **public-key encryption**