192 kbit/s for Anonymity

Motivation

Anonymity is an important feature for freedom of speech.

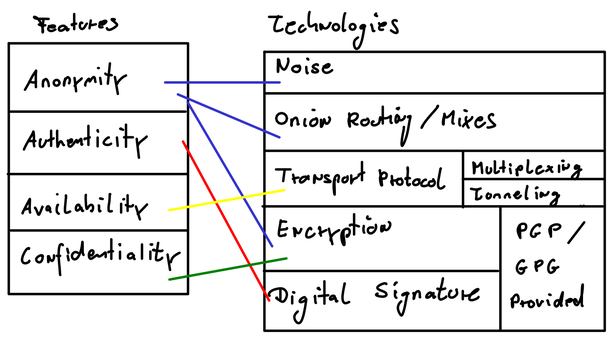
Anonymity systems are often heavy weighted or do not provide end-to-end anonymisation.

Current chat systems are not designed to provide anonymity.

Even worse, some are designed to hide from the user what they are doing.

This thesis presents a secure, peer-to-peer, decentralised anonymous chat system.

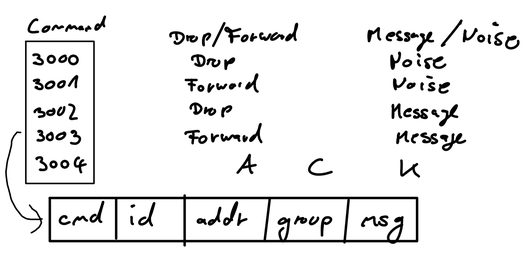
Anonymisation Techniques



Packet Types

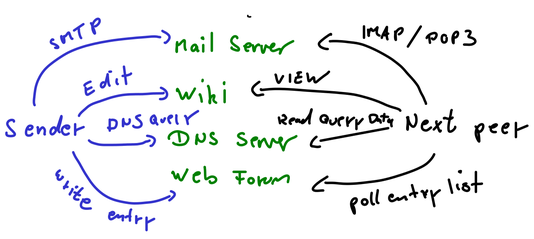
- eofmessages – plain text
- onions – multiply times encrypted
- postcards – onions + transport protocol frame

Eofmessages



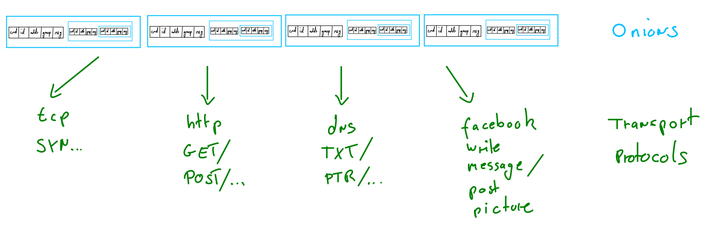
Transport Protocols

The chat system is transport protocol agnostic: It can use arbitrary protocols for transport.



Transport Protocol Tunneling

To make life harder for an attacker, onions can be tunneled via existing protocols like HTTP, SMTP, DNS or Facebook.



Transport Protocol Multiplexing

To enhance availability every peer can listen to a variety of different addresses. Every address could be supported by a different transport protocol.

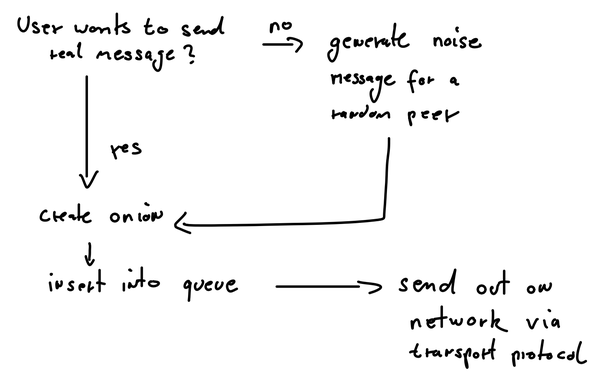
Noise

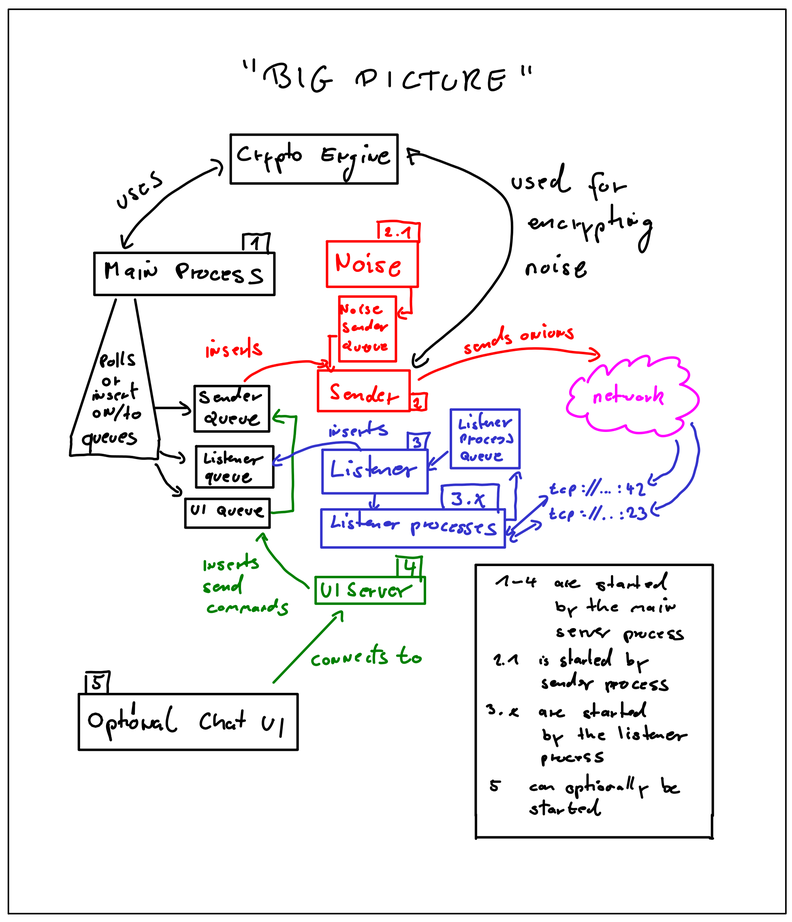
Noise is used to prevent an attacker to do statistical analysis on the network traffic.

Every peer sends a postcard every 250 ms or 4 postcards per second.

With an average postcard size of 6 KiB, this results in a continious network flow of 24 KiB/s or 192 kBit/s.

Noise Workflow

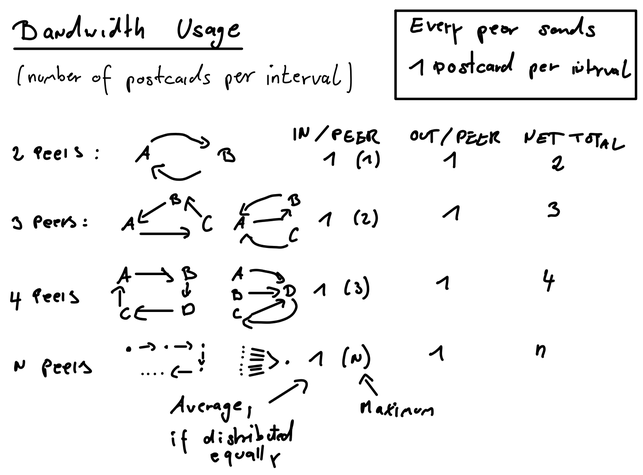




Bandwidth Usage

(number of postcards per interval)

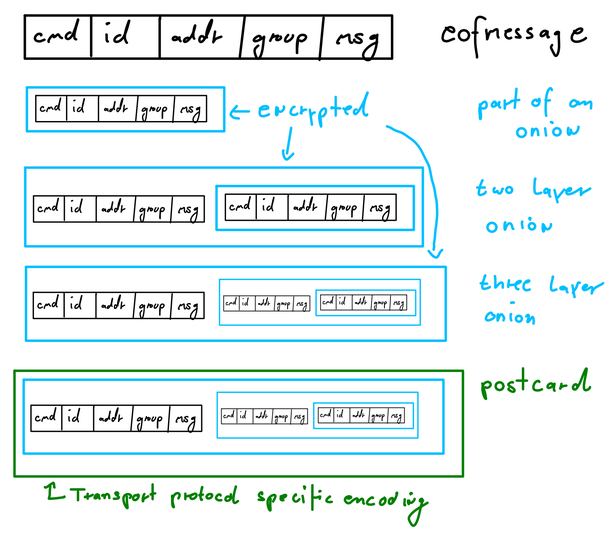
Every peer sends 1 postcard per interval



Onion Routing

An Onion is a multiply encrypted message. Every recipient can read one part of an onion and knows only about its predecessor and successor.

The readable part is called eofmessage and always consists of the same fields.

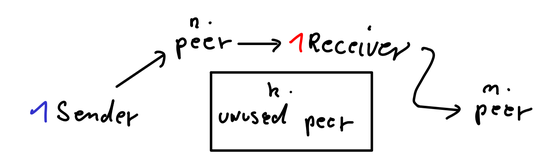


To hide the real recipient of a message, the recipient is inserted at a random position of the onion.

$n \geq 0$ proxy-peers $= n + m$

$m \geq 0$ route-peers $=$ proxy-peers $+ 1$

$k \geq 0$ network-peers $=$ route-peers $+ k$



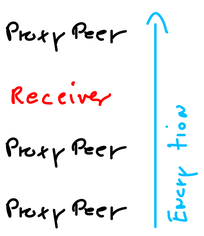
De-Anonymise:

- Compromise 1 Sender
- Compromise proxy-peers
- Compromise 1 Receiver

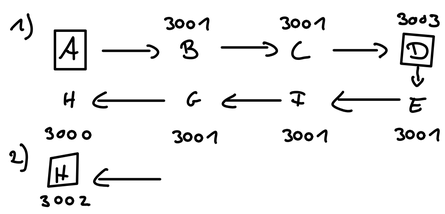
How to create an onion

Sender

1) Generate route
   - Receiver is at random position
   - Use random address of each peer
2) Create Onion
   - Encrypt for last peer first
   - Re-encrypt until first peer



Example Onion Route



Conclusions

- The proposed chat system works – proven by tests of the prototype.
- For real world usage, many participants are required.
- Usability should be improved.