Instant Messaging Security and Privacy

Chat and more while safeguarding your privacy

Klaus Möller

WP8-T1

Webinar, 24th of September 2020

Public

www.geant.org
Instant Messaging (IM) Introduction

- Other names: *Mobile Messaging* or simply *Online Chat*

- Originally: Sending (small) text messages to other users
  - First: on the same computer, later: world wide
  - User (person) had to be online to receive message
    - Some systems allow delivery from server later
    - Or use Chat-Bots (workaround in the beginning)

- Not limited to text anymore
  - Photos, Sounds, Video
  - File transfer between users

- Additional feature of Voice-/Videoconferencing systems
Instant Messaging Protocols

- Wide variety, some notable mentions
- HTTP(s)
  - As part of WebRTC or REST APIs
  - Discord, ...
- SIP (Telephony)
  - Skype, Facetime, ...
- SCIMP, NaCl
  - More secure messaging
  - Signal, Element(Riot)
  - Threema
- IRC, XMPP, ...
  - Legacy?
Centralized Instant Messaging Networks

- One central authority administers one or more central servers
- All users connect to these servers
- Facebook Messenger, Microsoft Live, etc.

**Pros:**
- New features can be added quickly
- Fast updates (if clients are also centrally administered with auto-updates)
- Few interoperability problems

**Cons:**
- At the mercy of the operator (Dishonesty, policy changes)
- Transparency? (Code reviews, independent audits)
- Connecting your own client? (Protocol or API documentation?)
- Government backdoor?
Federated Instant Messaging Networks

- Many authorities administer their own server(s)
- Servers are interconnected to form a (backbone) network
- Users connect to server of their choice, but still see (one) unified network

**Pros:**
- More control/trust over servers (if provided by a trusted party)
- Can run your own server on premise
- Source code (may be) available for review

**Cons:**
- Interoperability requirement makes modification of IM protocol difficult
- Users have to trust their server operator
- Conflicts between server operators may lead to network splits
Peer-to-Peer Instant Messaging Networks

- No servers, clients find each other through P2P mechanisms

**Pros:**
- Very little information exposed, nothing kept on servers
- E2EE by design (if encryption is part of the protocol)

**Cons:**
- How to initially connect to the network?
- No asynchronous delivery via server
  - But client may send message later
- IP-address still visible on the internet - P2P networks may be crawled
- Connectivity in the presence of NAT and Firewalls?
- Staying on the P2P network requires constantly sending keepalives, even if user is inactive
  - Drain on mobile device battery
General IM Risks

- Implementation Errors
- Identity theft
- Reputation
- Malware download
- Data exfiltration
- Botnet Command & Control Channels
- SPIM
- Archiving requirements, eDiscovery
Implementation Errors

- They happen and IM software is no exception
  - Client- & Server-side
- If security relevant: vulnerability
  - Server/Client is reachable from the internet: Worldwide exposure
- Mitigation
  - Patch: As soon as possible (Auto-Updates)
  - Reduce attack surface: Enable only the features/functionality you really need
  - Easiest to fix on centralized networks, OTOH: software monoculture
Identity

- Who is reachable how?
- And is this really the person you want to talk to?

https://xkcd.com/1810
https://xkcd.com/1782
Identity: Problems

• Problem 1: Finding the person we’re looking for
  – Central directory/search not on all networks (most often on centralized ones)
  – Real names do not need to have a relation to the ID on the network
    • Pseudonyms (may be necessary for self help groups, etc.)
    • Or just cool nicknames
  – Is the same name the same person on a different network?

• Problem 2: How do we know it is the real person?
  – Name collisions (John Smith)
  – Similar looking names (JohnnyS389, JohnnyS999, JohnnyS007, ...)
  – Deliberate Fakes (the_real_john_smith)
  – Phone number?
Identity: Theft

- I.e. your credentials to the IM network are compromised
  - Or even more, depending on what's in the directory
- Or someone creates an account with your name before you do
- Complain with the operators
  - How do you prove you are you?
  - And why should you have more rights to a name than the other person with the same name?

☑ Keep a good watch on your login credentials & logins
  ✓ Enable notifications from your ID provider for logins from unfamiliar sources
  ✓ Use 2FA if possible

☑ Use external sources to verify the identity of persons behind accounts
  ✓ Email, web pages, public keys, meet face-to-face
Malware Download

- Most IM protocols allow to download/share files
  - IRC, XMPP, ...
- Malware can be sent over these links
  - Custom protocols often not scanned
  - Esp. if the communication link is encrypted
- Or indirectly through HTTP links
- Esp. problematic if download is accepted automatically

☑ Disable that feature in your client, or disable downloads completely

☑ Disable that feature in your client, or disable downloads completely
Data Exfiltration

• Similar to Malware downloads, but in the other direction
• Can be used to send sensible/private information to outside parties
  − HTTP(S) or E-Mail often scanned by proxies/firewalls
  − Custom IM protocols usually not, esp. if encrypted

• How?
  − Accidentally
  − Intentionally (insider attack)
  − Client/account taken over or attacker mimics IM traffic

• Be careful when sending files
• Watch for unusual traffic patterns
  − Different servers (DNS), amount of traffic, etc.
Botnet Command & Control

• IM context: While user is offline (or occupied) a program can work as a stand-in
  - Bot: Program that holds connection to the IM network/channel
  - Level of functionality depends on programming

• Bot as Malware
  - Bot as a method to remotely control a system
  - For sending SPAM, conducting DDoS, exfiltrating data, etc.
  - Bots with connection to IRC channels (much declined, but still there)
  - Today: HTTPS connections
  - Rare: Bots with other IM protocols (JabberBot: XMPP)
**SPIM**

- **SPam over Instant Messaging**
- Text with links or images sent through IM
- Primarily on public networks with open groups
  - Also on private servers, if not sealed off from the internet
- **Mitigation**
  - Block/Ignore/Ban/Report SPIM account – spammer will move to another
  - Stay on invite-only groups/channels – doesn’t work if IM net allows direct messages (once your ID is known)
  - Receive only messages from IDs on your contact list – may get in the way of finding new contacts
  - Limiting the number messages users can send (server-side)
Archiving

- Laws may require relevant communications/documents to be archived
  - HIPPA, Sarbanes-Oxley, etc.
- This will include IM data if used for business relevant communication
- Legal Risk: What if IM data is not preserved/archived?
  - Related Problem: How to find communications/documents in the organizations archive/storage (eDiscovery)
- May collide with E2EE
  - Key escrow for business?
- OTOH: How long to keep logs of sessions (privacy protection)
  - Check the logging settings of client (and server)
Encryption

- Most desirable: End-to-End Encryption (E2EE)
  - Messages get encrypted at the sender and decrypted by the receiver
- Second best: Transport Encryption (most often: TLS)
  - Message is encrypted on the way to the server, but unencrypted there
- Problem: Nontransparent, it is often unclear whether
  - Is encryption used by default?
  - Is it E2EE or Transport Encryption?
  - What crypto-algorithms/key lengths are used?
  - Do the algorithms allow Forward Secrecy?
  - Do the algorithms allow deniability?
Encryption: Key Management

• With the messages encrypted, how is the key management done?
  • By the network/server operator?
    – Must be trustworthy
    – Transparency of the process?
    – Can users notice when the operator changes or discloses keys?
  • By the end user?
    – Eliminates the trust problem with the network/server operator
    – But must be done right
    – Do they have the required knowledge?
    – How is the key publication/revocation done?
Deniable Authentication in Instant Messaging

• Cryptography enables encrypted and integrity protected messages
  – But: The sender can’t deny that messages were from him
  – This “non-repudiation” property is often desired
  – I.e. business communications

• Use case/problem: Outsider breaks into channel (i.e. knows session key)
  – Can participants (later) deny that messages were send by them?
  – While still maintaining integrity (among them)?

• Why?
  – Participants may face prosecution (i.e. dissidents, whistle-blowers, ...)
Deniable Authentication Protocols

- Basic Idea: Authentication/Integrity Key is derived from the session key
  - If outsiders can get/break the session key, they also get the authentication key
  - And can thus forge (authentic) messages
  - So all participants can later deny that a message was sent from them
  - While the session key is unbroken, everything is fine (for the participants)

- Sample Protocols:
  - Off The Record (OTR) Messaging
    - On top of other protocols like XMPP, often through plug-ins
  - Silent Circle Instant Messaging Protocol (SCIMP)
    - Client: Silent Circle Phone
Deniable Authentication Caveats

• Metadata analysis of communication is still possible
  – Esp. for P2P-Networks
  – Need for VPN/TOR

• Human factor
  – What if somebody records the messages?
  – Someone discloses who was participating

• Endpoint security
  – I.e. somebody breaks into your device

• Legal
  – Judges have to believe in the cryptographic (technical) argument
  – As of yet unproven in courts
Instant Messaging Recommendations

• By the Electronic Frontier Foundation (EFF)
  - Communications encrypted in transit between all the links in the communication path
  - Communications encrypted with keys the provider does not have access to (E2EE)
  - Users can independently verify their correspondent's identity eg. by comparing key fingerprints
  - Past communications are secure if the encryption keys are stolen (forward secrecy)
  - Having the source code open to independent review (open source)
  - Having the software's security designs well-documented
  - Having a recent independent security audit
Instant Messaging Recommendations (cont.)

• Further (recommendations to users)
  ✓ Do not log or store any information regarding any message or its contents
  ✓ Do not log or store any information regarding any session or event
  ✓ Do not rely on a central authority for the relaying of messages (decentralized computing)

• Recommended Messengers (from privacytools.io)
  – Centralized: Signal
  – Federated: Element (formerly Riot)
  – P2P: Briar, Jami
Thank you

Any questions?

Next module: Videoconferencing Security & Privacy
28th of September 2020
www.geant.org
References

- JabberBot Analysis: https://www.welivesecurity.com/2013/01/30/walking-through-win32jabberbot-a-instant-messaging-cc/
- EFF Recommendations: https://web.archive.org/web/20191022070029/https://www.eff.org/node/82654
- OTR: https://otr.cypherpunks.ca/index.php
- Matrix Specifications: https://matrix.org/docs/spec/
Tools

- https://www.privacytools.io/software/real-time-communication/
- Signal: https://signal.org/
- Matrix: https://matrix.org/docs/guides/introduction
- Element: https://element.io/
- Briar: https://briarproject.org/
- Jami: https://jami.net/
- Keybase: https://keybase.io/