Die Matrix-Verschlüsselung näher betrachtet

Stefan Schumacher

$Id: MatrixProtokoll-Input.tex,v 1.67 2023/03/11 13:23:52 stefan Exp $

Chemnitzer Linux-Tage 2023[1]

[1] https://chemnitzer.linux-tage.de/2023/de/programm/beitrag/200

Inhaltsverzeichnis

1 Einführung

2 Double Ratchet

3 Megolm: Prinzip

4 Olm/Megolm Primitiven

5 Literatur

- Chemnitzer Linux-Tage 2004: Einführung in die Kryptographie
- Chemnitzer Linux-Tage 2011: Kryptographische Dateisysteme im Detail
- Auscert Australien 2014: Security in a Post NSA Age
- PHDays Moscow 2015: Why IT-Security is fucked up – and what we can do about it
- Chemnitzer Linux-Tage 2015: Abhörsichere Internet-Telefonie mit OSTN und OSTEL
- DeepSec Vienna 2016: Assessing the Hacking Capabilities of institutional and Non-institutional Players
- Chemnitzer Linux-Tage 2019: Kryptographische Dateisysteme im professionellen Umfeld
- Friedrich-Ebert-Stiftung Erfurt: Digitale Selbstverteidigung für Journalisten

Glossar I

symmetrisches Kryptosystem ein geteilter geheimer Schlüssel für Verschlüsselung/Entschlüsselung nötig

asymmetrisches Kryptosystem kein geteilter geheimer Schlüssel für Verschlüsselung/Entschlüsselung nötig

Diffie-Hellmann Key Exchange (DH KEX): Protokoll zur Schlüsselvereinbarung, Alice und Bob können über einen öffentlichen (abhörbaren) Kanal einen geheimen Schlüssel für symmetrische Kryptographie vereinbaren

Key Derivation Function (KDF) Schlüsselableitungsfunktion, leitet $1..n$ neue Schlüssel aus einem geheimen Schlüssel ab

(Perfect) Forward Secrecy (PFS) Sitzungsschlüssel werden so vereinbart, dass nach einem Einbruch *vergangene* Sitzungsschlüssel nicht gebrochen werden können (break-backward protection)

Glossar II

Break-in recovery Sitzungsschlüssel werden so vereinbart, dass nach einem Einbruch auch *zukünftige* Sitzungsschlüssel nicht gebrochen werden können (break-forward protection)

persistent nicht-flüchtig, Langzeitschlüssel/ID-Schlüssel

ephemeral flüchtig, Sizungsschlüssel, wird aus dem Langzeitschlüssel abgeleitet

Ende-zu-Ende (E2EE) Kommunikation wird zwischen Alice und Bob so verschlüsselt, das niemand dazwischen mitlauschen kann, egal ob ISP oder Server-Betreiber

Authentizität Echtheit des Absenders (Web of Trust/Zertifikate)

Integrität Daten wurden nicht verändert (Signatur)

Zurechenbarkeit Nachrichten sind dem zuzuordnen, der sie abgeschickt hat

Verbindlichkeit Sender kann Urheberschaft gesendeter Nachrichten nicht abstreiten

Glossar III

plausible Deniability Gegenteil zu Zurechenbarkeit && Verbindlichkeit, Alice und Bob können sich sicher sein, dass sie mit Bob bzw. Alice kommunizieren, aber Mallory kann nicht beweisen, dass Alice und Bob kommuniziert haben, im Englischen auch Off-the-Record genannt

Malleability Formbarkeit kryptographischer Nachrichten, d.h. Zurechenbarkeit && Verbindlichkeit können ausgehebelt werden

Device im Matrix-Kontext *alle* End-Geräte eines Nutzers, es ist für Matrix egal ob Alice 1 Handy und/oder 42 Laptops nutzt

Mallory der böse^{TM} Angreifer

M: Message/Nachricht; K: Key pub: public $priv$: private sig: signature enc: decryption

[5] https://chemnitzer.linux-tage.de/2023/de/programm/beitrag/200
[6] https://chemnitzer.linux-tage.de/2023/de/programm/beitrag/200

Matrix

- Application-Layer-Kommunikationsprotokoll für föderierte Echtzeitkommunikation
- Ziel: generisches Messaging- und Datensynchronisationsprotkokoll
- sicheres Verbreiten und Speichern von JSON-Nachrichten
- geteilter Zustand *nicht* direkte Nachrichten zwischen Partnern
- HTTP und WebRTC
- Implementierungen für VoIP, IoT, IM

Historische Entwicklung

- gar keine Verschlüsselung: (Mail, IRC, SMS): Mallory muss nur auf den Kommunikationskanal zugreifen um zu lesen/verändern zu können
- Transportverschlüsselung (SSL, TLS) Kommunikationsverbindungen werden im Netz verschlüsselt, können aufgebrochen werden, z.B. durch Vermittlungsserver, ALGs, Deep Packet Inspection ...
- Asymmetrische Verschlüsselung ohne PFS (SILC), Alice und Bob teilen Schlüssel via DHKEX, Alice und Charlie teilen Schlüssel via DHKEX Out-of-Band
- Alicens geheimer Schlüssel wird gestohlen (z.B. OpenSSL Heartbleed, Trojaner)
- Kommunikation zwischen Alice $\Leftrightarrow$ Bob und Alice $\Leftrightarrow$ Charlie kann entschlüsselt werden
- auch aufgezeichnete Kommunikation aus der Vergangenheit $\rightsquigarrow$ Problem!

Historische Entwicklung

Off-the-Record: Anforderungen

Lösung: E2EE mit PFS und Formbarkeit

1. (E2EE) Ende-zu-Ende-Verschlüsselung zwischen den Kommunikationspartnern, so dass niemand auf dem Transportweg mithören kann
2. (PFS) und auch vergangene Kommunikation bei einer Kompromittierung geheim bleibt
3. (Authentizität) und die Urheberschaft *während* der Kommunikation sichergestellt
4. (Formbarkeit) und die Urheberschaft *nach Ende* der Kommunikation abgestritten werden kann

Historische Entwicklung

Off-the-Record: Umsetzungen

Lösung: E2EE mit PFS und Formbarkeit

1. (E2EE) asymmetrische Verschlüsselung auf den Geräten von Alice und Bob (Diffie-Hellman ID-Schlüssel)
2. (PFS) aus dem ID-Schlüssel flüchtige Sitzungs-Verschlüsselungs-schlüssel ableiten und nach Sitzungsende vernichten
3. (Authentizität) Signaturschlüssel aus dem ID-Schlüssel ableiten und M signieren
4. (Formbarkeit) Signaturschlüssel flüchtig machen und nach Sitzungsende vernichten

Wichtig: Session-Management ist keine Eigenschaft der Verschlüsselung! Das muss das Anwendungsprogramm steuern

Historische Entwicklung

OTR Implementierungen

- Jabber/XMPP: Off-the-Record (OTR) (neben TLS, SASL, OpenPGP)
- XMPP Nachteil: keine Gruppenchats, beide Kommunikationspartner müssen gleichzeitig online sein
- Silent Circle (SCIMP): Authentifikation, Verschlüsselung, Forward Secrecy
- Signal: Erweiterung von SCIMP, Ziel: Perfect Forward Secrecy bei Kompromittierung eines der Kommunikationspartner
- Häufiger Wechsel der Sitzungsschlüssel zur »Selbstheilung«
- Axolotl-Protokoll, 2016 umbenannt in Double Ratchet
- aber: Signal ist auf synchrone 1:1 Kommunikation ausgelegt!
- Was ist wenn Bob 6 Monate offline ist?

Inhaltsverzeichnis

1 Einführung

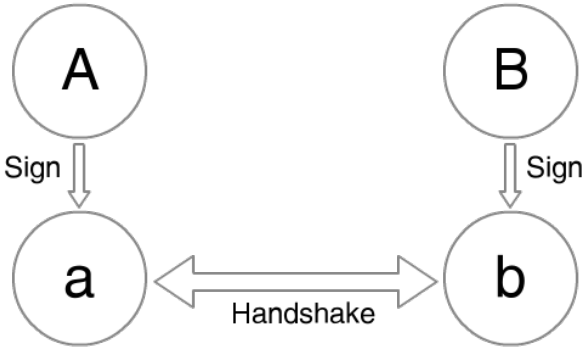
2 Double Ratchet

3 Megolm: Prinzip

4 Olm/Megolm Primitiven

5 Literatur

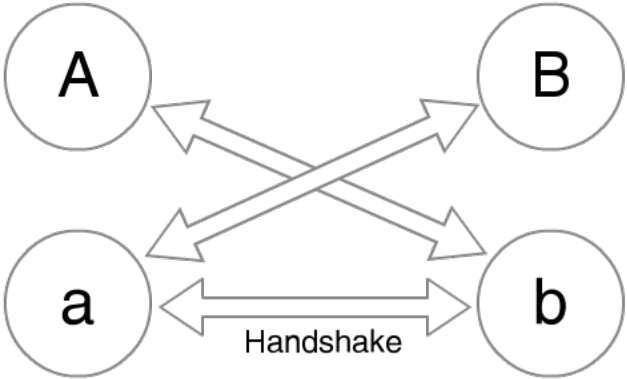
Off-the-Record Key Exchange



♣

OTR-KEX: Alice und Bob erzeugen DSA-Signatur aus DH-Schlüsselpaar und führen Handshake auf den Signaturen aus

Extended Triple Diffie-Hellman Key Exchange



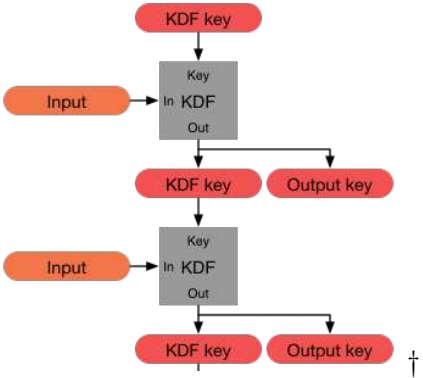
♣

X3DH-KEX: Alice und Bob leiten ephemerale Schlüssel (a,b) ab und führen 3 Handshakes zwischen den ID-Schlüsseln und ephemeralen Schlüsseln aus

Triple Diffie-Hellman Key Exchange

- X3DH-KEX: Alice und Bob leiten ephemerale Schlüssel (a,b) ab und führen 3 Handshakes zwischen den ID-Schlüsseln und ephemeralen Schlüsseln aus
- Danach werden 3 geteilte Geheimnisse abgeleitet: g^{aB}, g^{Ab}, g^{ab}
- Algorithmische und Protokoll-Komplexität reduziert, Formbarkeit erhöht, Nutzlast verringert, Forward Secrecy beibehalten
- X3DH-KEX: Nur zum Aufbau einer Session, danach nicht mehr verwendet
- bietet Forward Secrecy über den folgenden Sessions hinweg (inter-session)

KDF-Chain



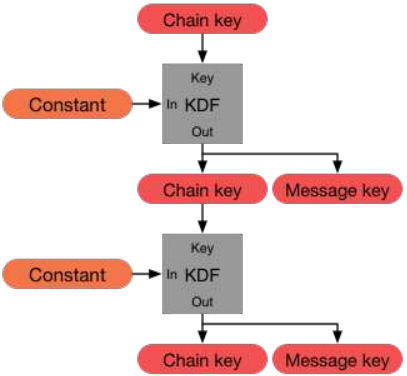
†

- Verkettete KDFs
- Teil der Ausgabe n_1 wieder Eingabe in n_2
- Output-Key erscheint für Mallory als Zufallsdaten
- selbst wenn Mallory den Input kontrolliert!
- vergangene Output-Keys sind sicher (PFS)
- zukünftige Output-Key sind sicher *wenn* zukünftige Inputs hinreichend Entropie beinhalten

Vorwärts immer - Rückwärts nimmer!



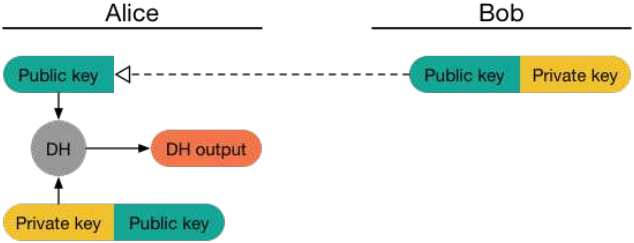
Symmetric Key Ratchet



†

- Jede M wird mit dem Message-Key K_n^M verschlüsselt
- K_n^M wird *nur* für M_n verwendet
- wird K_n^M kompromittiert, wird *nur* M_n kompromittiert
- K_n^M kann beliebig lange gespeichert werden
- gesamte Historie kann beliebig lange gespeichert werden
- aber: *symmetrisch*, funktioniert nicht mit anderen Kommunikationspartnern

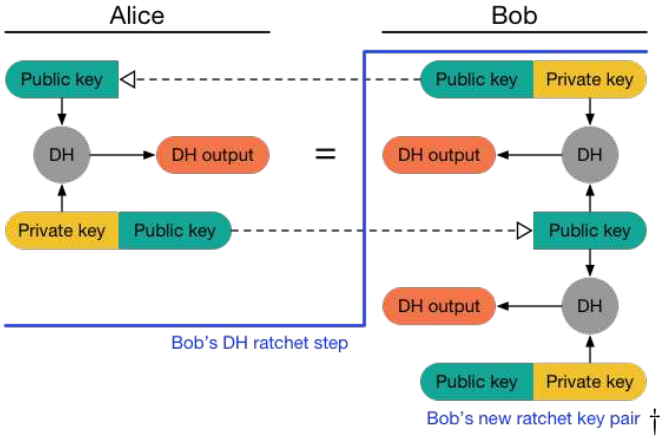
Diffie Hellman Ratchet



†

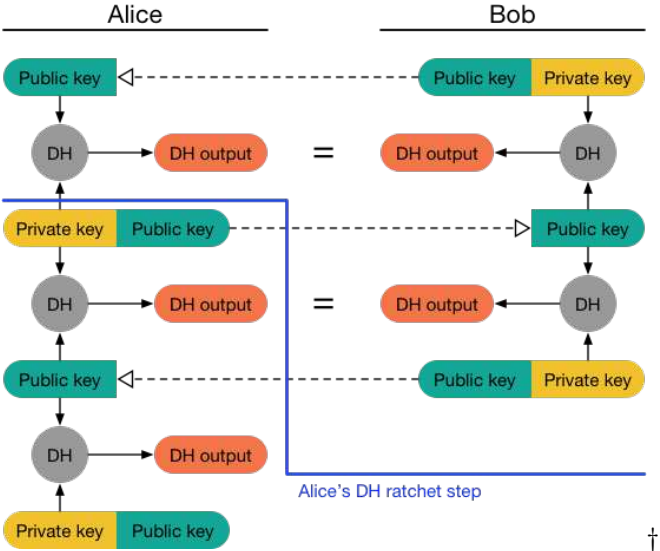
Alice nimmt Bobs öffentlichen Schlüssel als Input und führt mit ihrem privaten Schlüssel eine KDF-Operation durch

Diffie Hellman Ratchet



Bob nimmt Alicens öffentlichen Schlüssel und führt mit seinem privaten Schlüssel eine KDF-Operation durch

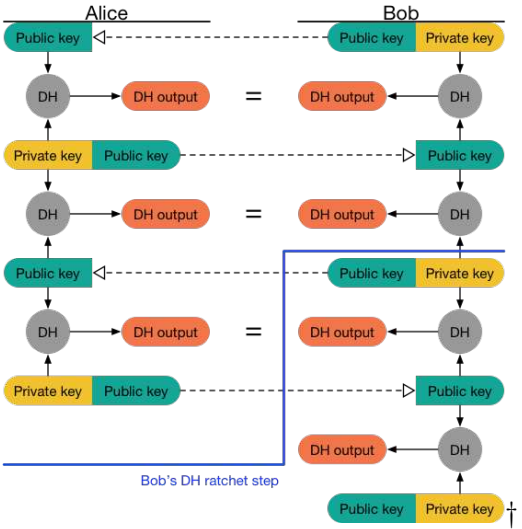
Diffie Hellman Ratchet



†

Neue DH-Schlüssel für jede Nachricht

Diffie Hellman Ratchet



ad infinitum

Verschränkungen zur Double Ratchet

- jede Nachricht enthält einen Header mit dem jeweiligen DH-Key
- neuer DH-Key $\rightsquigarrow$ Ratschen-Schritt
- modifiziert die KDF-Chain
- Output der DH-Ratchet ist Input für die Send/Receive-Chain

Alice:

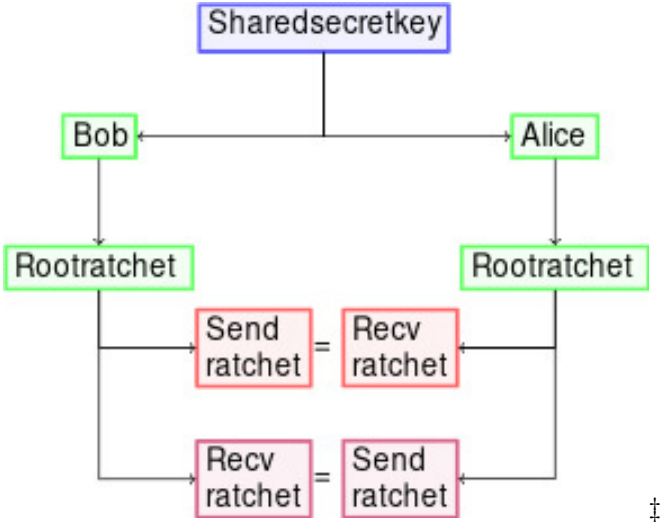
- DH-Chain
- Send-Ratchet
- Receive-Ratchet

Bob:

- DH-Chain
- Receive-Ratchet
- Send-Ratchet

Aber: Kein Schutz der Meta-Daten durch das Protokoll! Mallory sieht wer wann mit wem mit welcher Datenmenge kommuniziert hat!

Verschränkungen zur Double Ratchet



Die 3 bzw. 6 Ratchets im Überblick

Inhaltsverzeichnis

1 Einführung

2 Double Ratchet

3 Megolm: Prinzip

4 Olm/Megolm Primitiven

5 Literatur

Megolm

- Matrix will große Räume unterstützen *mit* Verschlüsselung, Signal nur 1:1
- Problem: n:n-Kommunikation mit $n \gg 1$ benötigt $\frac{n*n-1}{2}$ Verbindungen
- vollständiger Graph: n=25$\rightsquigarrow$ 300 ; n=100$\rightsquigarrow$ 4.950; n=1.000$\rightsquigarrow$ 499.500
- eine eigene Lösung muss her!
- Megolm: Gruppen-Ratsche für verschlüsselte Gruppen-Kommunikation
- Matrix nutzt immer Megolm, auch in 1:1 Chats
- zentrale Idee bei Matrix: geteilten Zustand synchronisieren
- Jeder Sender hat eine eigene Ratchet (*outbound session* um Nachrichten an einen Raum zu verschlüsseln
- Raum-Mitglieder können die Ratchet-Historie (Schlüssel) lokal speichern, um die Nachrichten-Historie lokal zu entschlüsseln
- verschlüsselte Nachrichten können auf dem öffentlichen Server bleiben
- Kann vorwärts gespult werden, damit neue Mitglieder alte Nachrichten nicht lesen können

Megolm

1. Nutzer tritt Raum bei $\rightsquigarrow$ Session wird initiiert mit Signatur-Schlüssel und einzelner Ratchet
2. K_{pub}^{sig} und Ratchet-Zustand wird n:n mit jedem Teilnehmer verschlüsselt via Olm ausgetauscht
3. Diese Session wird auf allen Nutzer-Geräten geteilt, Megolm hat nur 1 Session per User per Raum
4. Für jede gesendete Nachricht M wird K^{enc} aus einem Ratschen-Schritt abgeleitet
5. M wird mit K^{sig} signiert
6. Empfänger der Nachricht können K^{enc} ebenfalls aus einem Schritt auf ihrer Ratsche ableiten
7. M wird mit dem gesendeten K^{sig} verifiziert

Megolm

Caveats

- Trust Management vereinfacht, nur 1 Device/User
- Kein Zufalls-Seed beim Fortschreiten der Ratsche: M kann beliebig of entschlüsselt werden, wenn der Zustand der Ratsche bekannt ist
- Alte M können auf neuen Geräten gelesen werden!
- E2EE sinnvoll in großen Räumen? Mindestens Zutrittskontrolle!

Megolm

Angriffsvektoren

- Message Replay möglich, daher Ratschen-Index in der Anwendung tracken
- Transcript Consistency: Haben alle Raummitglieder die (gleiche) Nachricht empfangen? Forschungsfrage!
- Keine Post Compromise Security in Megolm für die Dauer einer Session! Lebenserwartung einer Session ist Eigenschaft des Raumes und kann konfiguriert werden, z.B. auf $M = 1$
- Perfect Forward Secrecy: Endanwendung kann alte Sessions löschen oder Ratschen-Indizes weiterdrehen
- Die gesamte Sicherheit einer Megolm-Session beruht auf der Sicherheit des Kanals für den Schlüsselaustausch (hier: Olm)
- Verschlüsselung in einem Raum aktivieren: $O(n)$
- Messaging Layer Security: $O(log\ n)$

Inhaltsverzeichnis

1 Einführung

2 Double Ratchet

3 Megolm: Prinzip

4 Olm/Megolm Primitiven

5 Literatur

Olm Primitiven

Ed25519 Fingerprint-Schlüssel-Paar Für jedes Device, identifiziert diese, K_{pub} wird im Matrix-Netz publiziert

Curve25519 ID Schlüsselpaar etabliert ein geteiltes Geheimnis, jedes Device hat ein langlebiges ID-Paar um Olm-Sitzungen zu etablieren. K_{25519} wird mit dem Ed25519 Fingerprint-Schlüssel signiert und im Matrix-Netz geteilt. Das Curve25519-Paar sollte idealerweise regelmäßig rotiert werden

Curve25519 Einmal-Schlüssel-Paare werden ebenfalls zum Aufbau der Olm-Session verwendet, aber jeweils nur ein mal
Alice erzeugt eine Anzahl an Einmal-Schlüssel-Paaren und veröffentlicht die öffentlichen Schlüssel auf dem Homeserver. Wenn Bob eine Olm-Session mit Alice starten will, muss er einen dieser Schlüssel beanspruchen und ein eigenes Schlüssel-Paar erzeugen. Zusammen mit den ID-Schlüsseln wird damit die Olm-Session aufgebaut.

Megolm Primitiven

Megolm Verschlüsselungsschlüssel verschlüsselt Gruppen-Nachrichten, initialisiert mit Zufallsdaten. Mit jeder Nachricht wird ein neuer Hash aus dem Megolm-Schlüssel abgeleitet zur Verschlüsselung der nächsten Nachricht Alice kann Bob den aktuellen Hash schicken, so dass dieser zukünftige Nachrichten entschlüsseln kann, nicht aber vergangene

Ed25519 Megolm Signaturschlüssel Für jede Megolm-Session wird ein Ed25519 Signaturschlüssel erzeugt, um den Absender zu authentifizieren. K_{pub} wird im Raum geteilt.

Device-Schlüssel Device-Schlüssel werden nur einmal erzeugt und registriert, wenn das Device zu ersten mal startet. Diese Schlüsselpare werden via Olm erzeugt in mit dem Homeserver geteilt.

Matrix-Nachricht I


```
{
  "content": {
    "algorithm": "m.megolm.v1.aes-sha2",
    "ciphertext": "AwgAEsABj4z8dhrwnjTELLPsONiSCEa/3IfE58+aTBGG9s4fQ
    "device_id": "MNCTGZADAJ",
    "sender_key": "9Vt/swlq0x1CvShirCArWMumdMGbq7/NHr46i88Njyk",
    "session_id": "z2/f3G/SVvYzWxPbQc7JfalWZ37YKV/NeNA3M6Tp6PI"
  },
  "origin_server_ts": 1678349743104,
  "room_id": "!HPiGXHIZsSZGXuSLxg:matrix.org",
  "sender": "@0xkaishakunin:matrix.org",
  "type": "m.room.encrypted",
  "unsigned": {
    "age": 21462277
```

Matrix-Nachricht II


```
  },
  "event_id": "$3NM8H6QGPQ_7bha9JeNWkVm3n2vj0alYuTP6azLgWy8",
  "user_id": "@0xkaishakunin:matrix.org",
  "age": 21462110
}
```


[34] https://chemnitzer.linux-tage.de/2023/de/programm/beitrag/200

Inhaltsverzeichnis

1 Einführung

2 Double Ratchet

3 Megolm: Prinzip

4 Olm/Megolm Primitiven

5 Literatur

Albrecht, M. R., Celi, S., Dowling, B. & Jones, D. (2022). Practically-exploitable Cryptographic Vulnerabilities in Matrix. *2023 IEEE Symposium on Security and Privacy (SP)*, 1419–1436.

Chowdhury, P. D., Sameen, M., Blessing, J., Boucher, N., Gardiner, J., Burrows, T., Anderson, R. & Rashid, A. (2023). Threat Models over Space and Time: A Case Study of E2EE Messaging Applications. *arXiv preprint arXiv:2301.05653*.

Cremers, C. & Feltz, M. (2011). One-round Strongly Secure Key Exchange with Perfect Forward Secrecy and Deniability [https://eprint.iacr.org/2011/300]. https://eprint.iacr.org/2011/300

Degabriele, J. P., Lehmann, A., Paterson, K. G., Smart, N. P. & Strefler, M. (2011). On the Joint Security of Encryption and Signature in EMV [https://eprint.iacr.org/2011/615]. https://eprint.iacr.org/2011/615

van der Have, F. (2022). *The X3DH Protocol: A Proof of Security* (Diss.). Raboud Uni.

Borisov, N., Goldberg, I. & Brewer, E. (2004). Off-the-record communication, or, why not to use PGP. *Proceedings of the 2004 ACM workshop on Privacy in the electronic society*, 77–84.

Di Raimondo, M., Gennaro, R. & Krawczyk, H. (2005). Secure off-the-record messaging. *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society*, 81–89.

Alexander, C. & Goldberg, I. (2007). Improved user authentication in off-the-record messaging. *Proceedings of the 2007 ACM workshop on Privacy in electronic society*, 41–47.

Moscaritolo, V., Belvin, G. & Zimmermann, P. (2012). Silent circle instant messaging protocol protocol specification. *Online, White Paper*, 47.

Marlinspike, M. (2016, 20. November). *The Double Ratchet Algorithm* (T. Perrin, Hrsg.; Tech Report).

Alwen, J., Coretti, S. & Dodis, Y. (2018). The Double Ratchet: Security Notions, Proofs, and Modularization for the Signal Protocol. *IACR Cryptol. ePrint Arch.* https://cs.nyu.edu/~dodis/ps/signal.pdf

Cohn-Gordon, K., Cremers, C. J. F., Dowling, B., Garratt, L. & Stebila, D. (2017). A Formal Security Analysis of the Signal Messaging Protocol. *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, 451–466. https://eprint.iacr.org/2016/1013.pdf

Alwen, J., Coretti, S., Jost, D. & Mularczyk, M. (2020). Continuous group key agreement with active security. *Theory of Cryptography Conference*, 261–290. https://eprint.iacr.org/2020/752.pdf

Jaeger, J. & Stepanovs, I. (2018). Optimal Channel Security Against Fine-Grained State Compromise: The Safety of Messaging.. *IACR Cryptology ePrint Archive, 2018*. https://eprint.iacr.org/2018/553.pdf

Vehkaoja, V. (2017). End-to-end encryption protocol for internet of things devices. *University of Oulu, degree programme in computer science and engineering, Master's Thesis, 38*. http://jultika.oulu.fi/files/nbnfioulu-201706022469.pdf

Bienstock, A., Fairoze, J., Garg, S., Mukherjee, P. & Raghuraman, S. (2022). A More Complete Analysis of the Signal Double Ratchet Algorithm [https://eprint.iacr.org/2022/355]. Cryptology ePrint Archive, Paper 2022/355. https://eprint.iacr.org/2022/355

[37] https://chemnitzer.linux-tage.de/2023/de/programm/beitrag/200
[38] https://chemnitzer.linux-tage.de/2023/de/programm/beitrag/200
[39] https://chemnitzer.linux-tage.de/2023/de/programm/beitrag/200

Bilderquellen:

- ♣ https://signal.org/blog/simplifying-otr-deniability/
- † https://signal.org/docs/specifications/doubleratchet
- ‡ https://nfil.dev/coding/encryption/python/double-ratchet-example/

Quellen:

- https://blog.jabberhead.tk/2019/03/10/a-look-at-matrix-orgs-olm-megolm-encryption-protocol/
- https://matrix.org/docs/spec/clientserver/r0.4.0.html#m-room-encrypted
- https://gitlab.matrix.org/matrix-org/olm/blob/master/docs/olm.md
- https://matrix.org/docs/guides/end-to-end-encryption-implementation-guide
- https://gitlab.matrix.org/matrix-org/olm/blob/master/docs/megolm.md#session-sharing-format
- https://matrix.org/docs/spec/clientserver/r0.4.0.html#m-room-encrypted