Sichere Verschlüsselung in Gruppen mit dem MLS-Protokoll

Stefan Schumacher

cryptomancer.de

Kieler Open Source und Linux-Tage 2025

$ Id: MLS-Protokoll-Input.tex,v 1.13 2025/09/20 06:01:35 stefan Exp $

Inhaltsverzeichnis

1. Einführung
2. Matrix - 3DH-KEX und DoubleRatchet
3. Messaging Layer Security Protocol
   - Fähigkeiten
   - Architektur
   - Ratschenbaum
   - Sicherheitseigenschaften



- Robotron KC85/3 Mitte der 80er Jahre
- Geek, Nerd, Hacker
- Kryptographie seit 30 Jahren
- einige Jahre NetBSD-Entwickler
- ehem. Sicherheitsforscher u.a. Fach-Didaktik der Kryptographie
- jetzt Sicherheitsarchitekt bei einem öffentlichen IT-Dienstleister

cryptomancer.de

Glossar

- CLT2023: Das Matrix-Protokoll näher betrachtet – Die Kryptographie-Architektur
- https://cryptomancer.de/posts/202303matrix/
-
- Glossar samt Formelzeichen und Literatur im Handout
- https://www.cryptomancer.de/messaginglayer/

Inhaltsverzeichnis

1. Einführung
2. Matrix - 3DH-KEX und DoubleRatchet
3. Messaging Layer Security Protocol
   - Fähigkeiten
   - Architektur
   - Ratschenbaum
   - Sicherheitseigenschaften

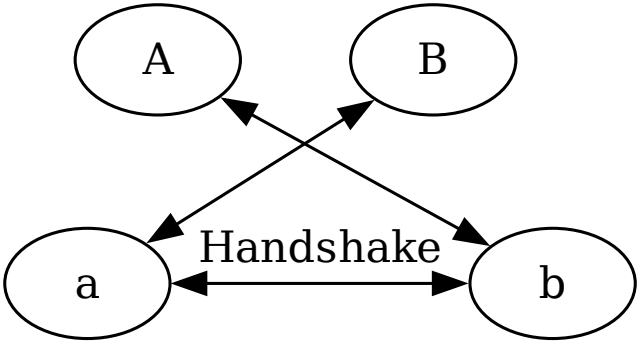
Matrix

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

Triple Diffie-Hellman Key Exchange

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

Triple Diffie-Hellman Key Exchange



cryptomancer.de

Matrix und große Räume

- für jede Nachricht wird ein eigener neuer ephemeraler Schlüssel aus dem Session-Schlüsselpaar von Alice und Bob abgeleitet
- private Schlüssel nach der Nachricht vernichtet
- Jeder Sender hat eine eigene Ratsche (*outbound session* um Nachrichten an einen Raum zu verschlüsseln
- Raum-Mitglieder können die Ratschen-Historie lokal speichern, um die Nachrichten-Historie lokal zu speichern
- Kann vorwärts gespult werden, damit neue Mitglieder alte Nachrichten nicht lesen können
- Problem: n:n-Kommunikation mit $n \gg 1$ benötigt $\frac{n*n-1}{2}$ Verbindungen
- vollständiger Graph: n=25$\rightsquigarrow$ 300 ; n=50.000$\rightsquigarrow$ 1.249.975.000

Matrix und MLS

- https://opencode.de/de/software/synapse-mit-mls-4201
- https://matrix.org/blog/2023/07/a-giant-leap-with-mls/
- https://arewemlsyet.com/

Inhaltsverzeichnis

1. Einführung

2. Matrix - 3DH-KEX und DoubleRatchet

3. Messaging Layer Security Protocol
   - Fähigkeiten
   - Architektur
   - Ratschenbaum
   - Sicherheitseigenschaften

Messaging Layer Security

- Sicherheitsschicht für Ende-zu-Ende-Verschlüsselung
- sicheres und effizientes Protokoll
- Application Layer
- für große Gruppen mit bis zur 50 000 Teilnehmern
- Idee dazu 2016 auf der IETF96 in Berlin geboren
- 2017: Asynchronous Ratcheting Trees
- 29.3.2023: MLS von der IETF als neuer Standard verabschiedet
- Draft für PQC (ML-KEM) läuft

Messaging Layer Security

- Schlüssel-Einigungs-Protokoll für Gruppen
- für z.B. Instant Messaging
- Schutz gegen Abhören, Manipulation, gefälschte Nachrichten,
- Forward Secrecy und Post Compromise Security
- Geteilter, geschützter Gruppen-Zustand (Mitgliedschaft, krypt. Eigenschaften ...)
- Nachrichten-Konsistenz in Gruppe: jedes Mitglied erhält die gleiche Nachricht
- Architektur *und* Protokoll
- *kein* Nachrichtenprotokoll
- MLS soll in XMPP oder Matrix eingebettet werden
- MLS kann auch ohne kryptographische Absicherung eingesetzt werden
- kann föderiert werden

Logischer Aufbau

- *Clients* bilden *Gruppen*
- Gruppen bilden *Epochen*, z.B. wenn sich Eigenschaften der Gruppe ändern
- Jede Gruppe wird als Baum repräsentiert
- Gruppenmitglieder sind Blätter des Baumes
- Untergruppen des Baumes können effizient verschlüsseln
- jeder Client hat ein `LeafNode` mit Identität, Credentials und Fähigkeiten
- Fähigkeiten: des eingesetzten MLS-Protokolls und/oder Erweiterungen

Abstrakte Dienste

Authentication Service (AS)

- attestiert die Beziehung von öffentlichen MLS-Schlüsseln und der Identität
- generiert Credentials die diese Beziehung codieren
- validiert diese Credentials

Delivery Service (DS)

- verteilt Nachrichten an Gruppenmitglieder
- speichert und verteilt öffentliche Schlüssel (`KeyPackage`), die benötigt werden um geheime Gruppenschlüssel auszuhandeln

Dienste können im Netz verteilt sein

Synchronität

- jede MLS-Operation kann asynchron ausgeführt werden
- Gruppen-Schlüssel können aktualisiert werden
- Mitglieder entfernen oder reinlassen
- Nachrichten verschicken
- Transportschicht des Messengers muss dafür sorgen dass MLS-Nachrichten asynchron und verlässlich zugestellt werden

Access Control

- Alle Gruppenmitglieder mit Zugriff auf den Gruppenschlüssel können Gruppen-Operationen starten
- jedes Gruppenmitglied kann Mitglieder entfernen/reinlassen
- Messenger muss Access Control implementieren
- z.B. Gruppenadministratoren ernennen oder offene Foren zulassen
- Handshake-Nachrichten können unverschlüsselt gesendet werden z.B. für ACL
- Transportverschlüsselung wichtig!

TreeSync Protokoll

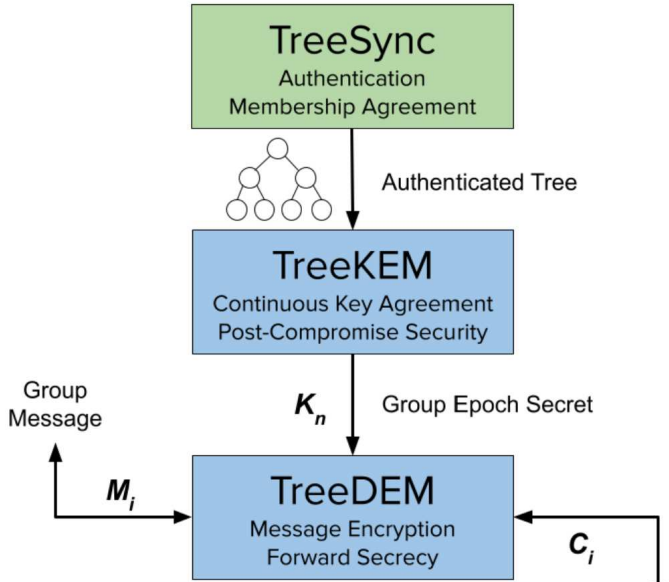
- synchronisiert den geteilten Gruppen-Zustand zwischen den Gruppenmitgliedern
- als Binär-Baum
- jedes Blatt repräsentiert ein Mitglied
- Gruppenzustand wird über Signaturen in Merkle-Bäumen authentifiziert
- Datenstruktur des Baumes stellt eine interne Integritäts-Invariante sicher
- authentifizierter, synchronisierter Zustand wird an TreeKEM übergeben

TreeKEM Protokoll

- ermöglicht es jedem Gruppenmitglied seine geheimen Schlüssel zu nutzen
- auf der Sequenz von authentifizierten Zuständen aus TreeSync
- um eine Sequenz von Gruppen-Schlüsseln (sog. Epochen-Geheimnissen) abzuleiten
- nutzt Baumstruktur für effiziente Updates: logarithmische Anzahl an Verschlüsselungen und 1 Entschlüsselungen
- Post Compromise Security und Schutz gegen ehemalige Gruppenmitglieder

TreeDEM Protokoll

- nimmt das Epochen-Geheimnis von TreeKEM
- leitet davon Nachrichten-Verschlüsselungs-Schlüssel für jedes Gruppenmitglied ab
- nach jeder Nachricht wird der Nachrichten-Verschlüsselungs-Schlüssel weiter geratscht
- um Forward Secrecy zu bekommen

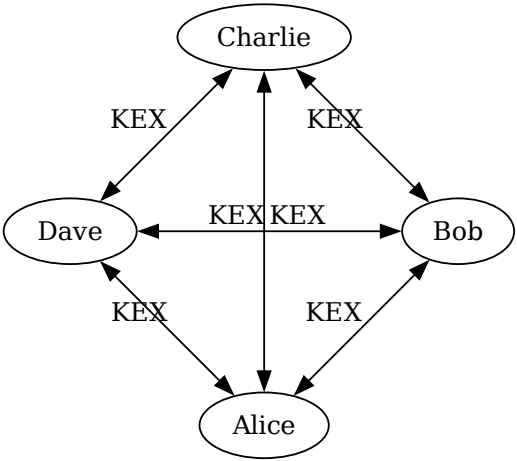


aus [1]

cryptomancer.de

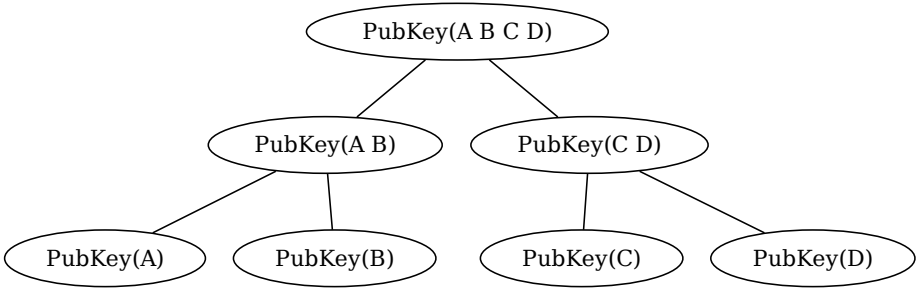
Ratschenbäume

- MLS nutzt ebenfalls Ratschen-Bäume
- als perfekt ausbalancierte Binär-Bäume
- Jedes Gruppenmitglied ist einem Blatt zugeordnet
- Pfad zur Wurzel enthält alle geheime Schlüssel, die das Mitglied kennt
- Der geheime Schlüssel in einem Knoten ist einem Mitglied nur bekannt, wenn der Unterbaum des betreffenden Knoten das Blatt des Mitglied beinhaltet



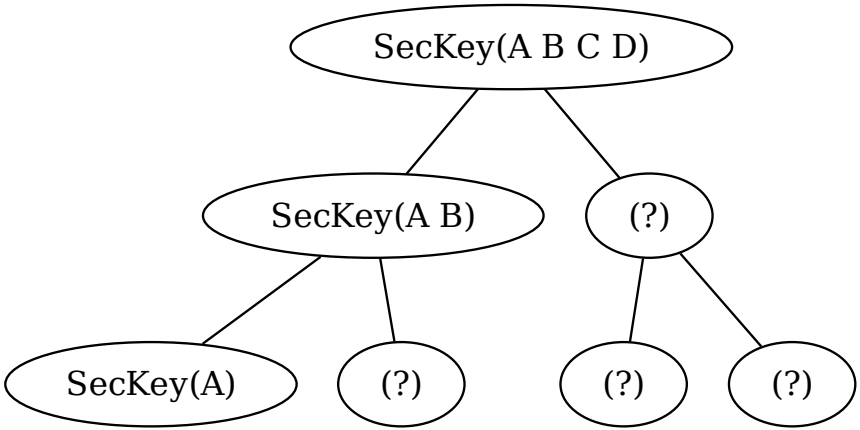
N to N (5 to 5 = 10)

cryptomancer.de



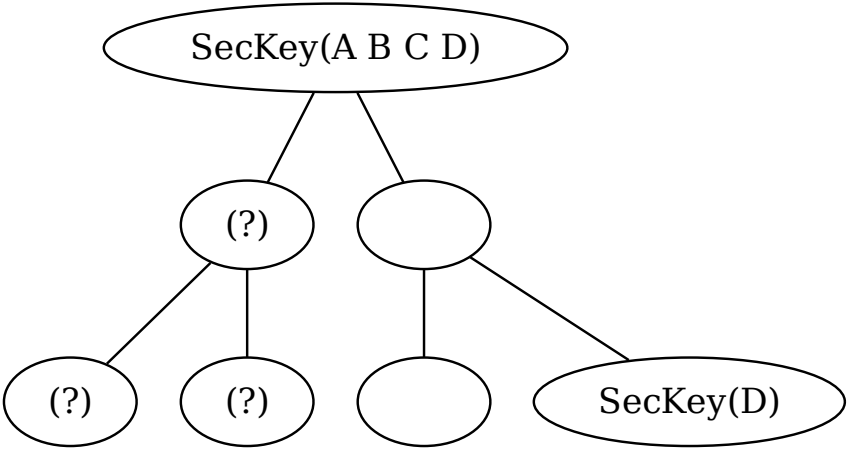
Public Tree

cryptomancer.de



Full Secret Tree for Alice

cryptomancer.de



Secret Tree with hole for Dave

cryptomancer.de

Sicherheitseigenschaften

- Nachrichten: E2EE sichert Vertraulichkeit und Integrität *aber* Metadaten ohne TLS sichtbar
- Gruppengeheimnisse: werden aus dem Ratschenbaum abgeleitet, Private Schlüssel sind nur Mitgliedern des Unterbaumes (der Gruppe) bekannt, C können an alle $n-1$ Mitglieder mit logarithmischen Aufwand verschickt werden. Schlüsselrotation und Mitglieder-Entfernung effizient möglich (logarithmisch)
- Vertraulichkeit des Absenders: Nachricht und Absenderdaten werden mittels AEAD-Ciphersuiten verschlüsselt, die einen privaten Schlüssel des Absenders nutzen. Nachrichten sind damit de facto kollisionsresistent

Sicherheitseigenschaften

- Authentizität: AEAD-Ciphersuiten, kryptographische Signatur unter jeder Nachricht
  Signaturschlüssel verifiziert Identität des Mitglieds!
- Authentizität: *zusätzliche* Absicherung durch PSK in Applikation
- Gruppen-Fragmentierung: böswilliges Gruppenmitglied kann individuelle Update-Pfade erzeugen und Opfer aus dem Ratschen-Braum zwingen
  Applikation sollte Proposals von n Gruppenmitgliedern akzeptieren bevor diese committed werden

Gruppen-Metadaten

Unverschlüsselt:

- KeyPackage Nachrichten
- GroupInfo Nachrichten
- Der unverschlüsselte Teil der Welcome-Nachricht
- Proposal oder Commits die als PublicMessage gesendet werden
- Unverschlüsselte Header in PrivateMessage Nachrichten
- Die Länge von verschlüsselten Welcome und PrivateMessage Nachrichten

Transportverschlüsselung empfohlen!

Forward Secrecy

Forward Secrecy

- zwischen einzelnen Epochen: alte private Schlüssel werden aus dem Ratschen-Baum gelöscht
- in einer Epoche: private Nachrichtenschlüssel werden nach Verschlüsselung der Nachricht gelöscht
- NB: private Nachrichtenschlüssel werden zwischen allen Gruppenmitgliedern geteilt!
- Gefahr bei Offline-Mitgliedern! Applikation sollte Offline-Mitglieder entfernen können

Post-Compromise Security

Post-Compromise Security zwischen einzelnen Epochen:

- AktualPost Quantum Cryptography isierung der Blatt-Schlüssel der Mitglieder im Ratschen-Baum
  zukünftige Nachrichten werden mit neuen Schlüsseln verschlüsselt
- NB: PCS gilt erst gruppenweit, wenn alle Mitglieder den Update-Proposal commited haben!

KeyPackages

- enthalten u.a. den InitKey
- InitKey muss zur gewählten CipherSuite passen
- und einzigartig sein
- Mehrfach genutzte KeyPackages gefährden beides!
- Wiederbenutzung ist Ultima Ration gegen DoS-Attacke
- Applikation sollte KeyPackages regelmäßig neu generieren
- KeyPackage darf nicht kompromittiert werden!

Post Quantum Cryptography

- Post Quantum Cryptography
- Draft offen
- https://datatracker.ietf.org/doc/draft-ietf-mls-pq-ciphersuites/
- Mehr dazu in meinem 3. Vortrag zur Kryptokalypse

Fragen?

- cryptomancer.de
- kaishakunin.com
- https://mastodon.social/@0xKaishakunin
- 7B2B 45B1 330E 579E 3C3E 9B34 089D 8068 8050 ECCF
- Matrix: @SchumaSte:matrix.org

cryptomancer.de