Messaging Layer Security

Towards a new era of secure messaging…

Benjamin Beurdouche, Katriel Cohn-Gordon, **Raphael Robert**

BlackHat USA 2019, Las Vegas, CA

Intro

- **MLS: Messaging Layer Security**
- MLS is a new protocol for end-to-end encrypted messaging
- MLS is now an IETF working group
- Why is this important now?

- Raphael Robert: Head of Security at @wire moz://a

Inria

UNIVERSITY OF OXFORD

CISCO

ACLU

wire

MIT

NYU PARIS

wickr

CISPA
HELMHOLTZ CENTER FOR INFORMATION SECURITY

CLOUDFLARE

Google

facebook

Current status

Secure Messaging

Lots of secure messaging apps.

Some use similar protocols…

… some are quite different.

… but all have similar challenges.

Very different levels of analysis.

Everyone maintaining their own libraries.

History of security properties

PGP (OpenPGP, S/MIME, ...)

Confidentiality and Authenticity

Off-the-record protocol

Introduces Forward Secrecy and Deniability

Double Ratchet algorithm

Adapts to asynchronous communication,
introduces the notion of "future secrecy"

What about groups?

Pairwise protocols cannot “just” be extended to accommodate for groups

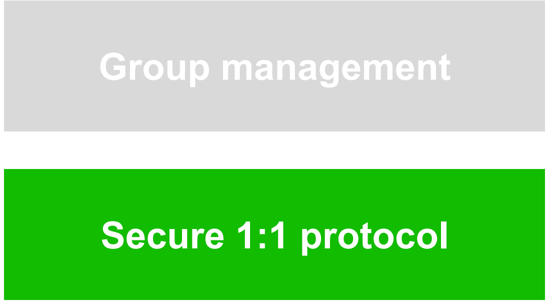
The pairwise channels can be superposed to simulate a group

Tradeoff between security properties and scalability

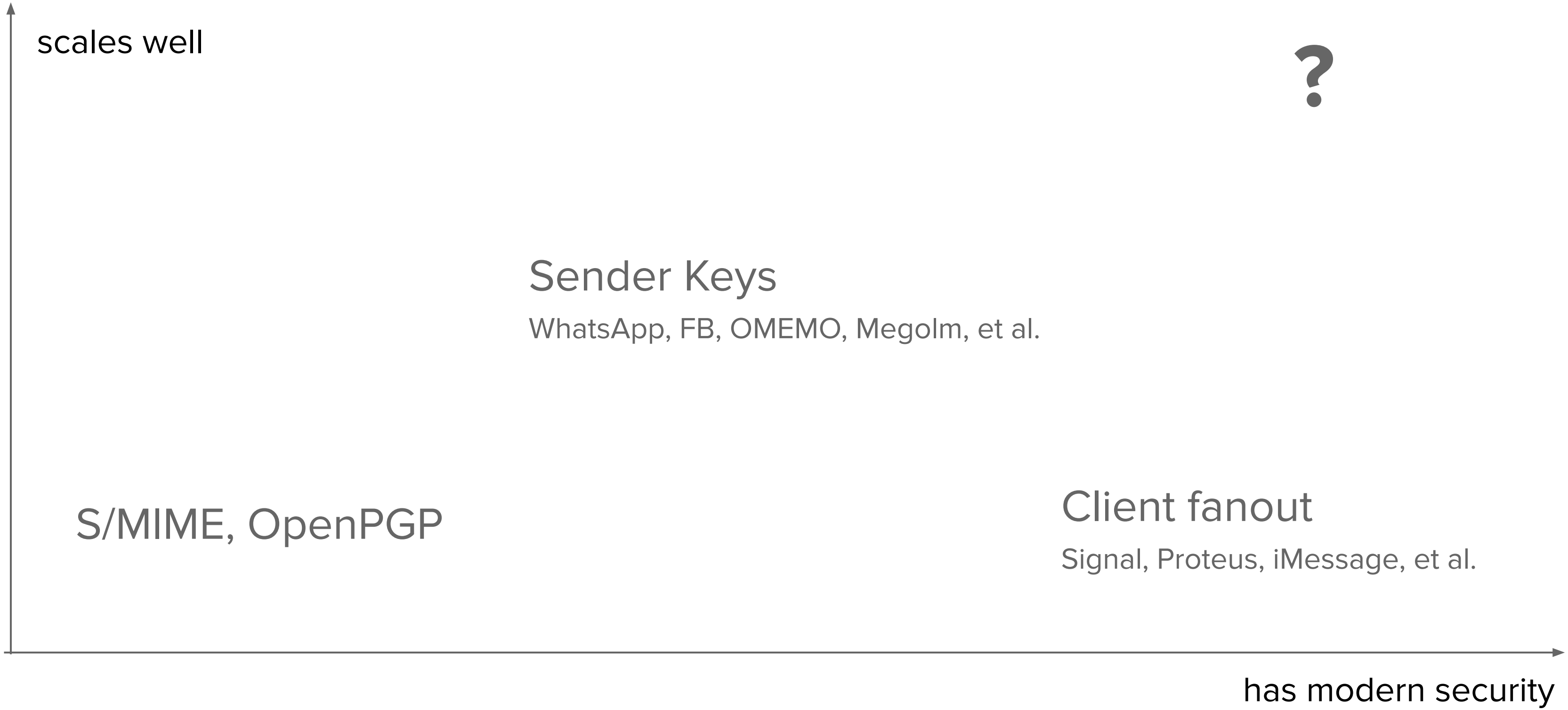
What about groups?

Creating groups on top of a pairwise protocol is hiding the complexity behind a non-standard layer.

Everybody has a different solution and everybody has different security properties.



Modern security & scaling



Objectives

What do we want?

Security Protocol with modern security properties:

- Confidentiality and Authenticity
- Forward Secrecy (FS)
- Post-compromise Security (PCS)
- Membership authentication in groups
- Deniability (optional)

What do we want?

Async - Support sessions where no two members are online at the same time

Group Messaging - Support large, dynamic groups with efficient scaling

Multi-device support - Users should be able to use more than one device

Federation - Members of groups should not be limited to only one server/service

Usable - Focus on a practical drop-in for existing applications

What do we want?

Open standard - Complete specification usable by anyone

Code reuse - Robust implementations that can be used in different contexts

Security analysis - Involvement from the academic community

MLS

History

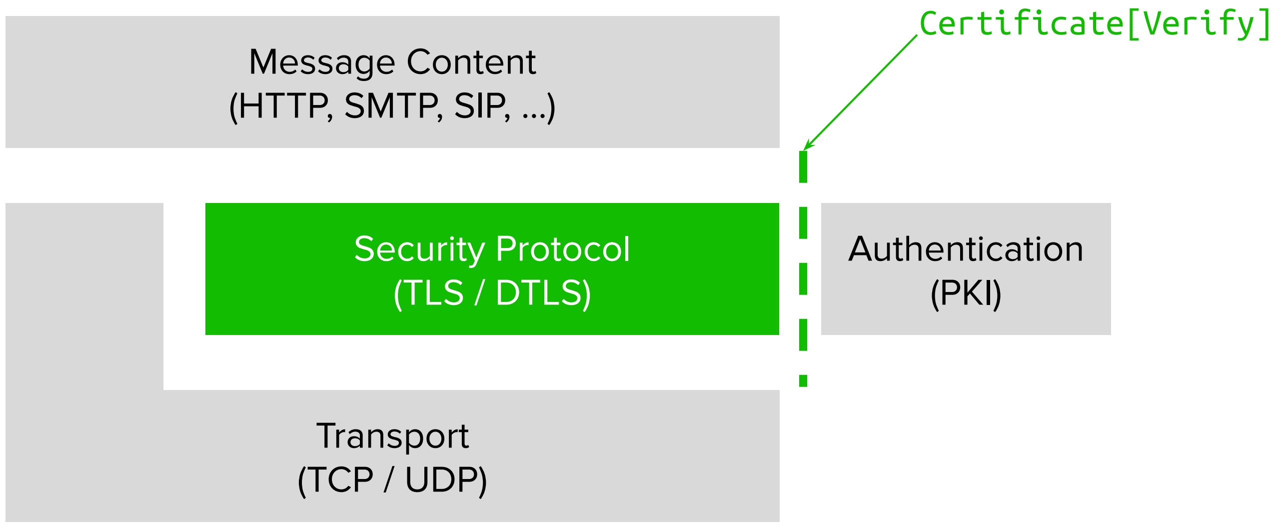
March 2018
IETF MLS BoF

May 2018
IETF MLS WG officially formed

2015: ART

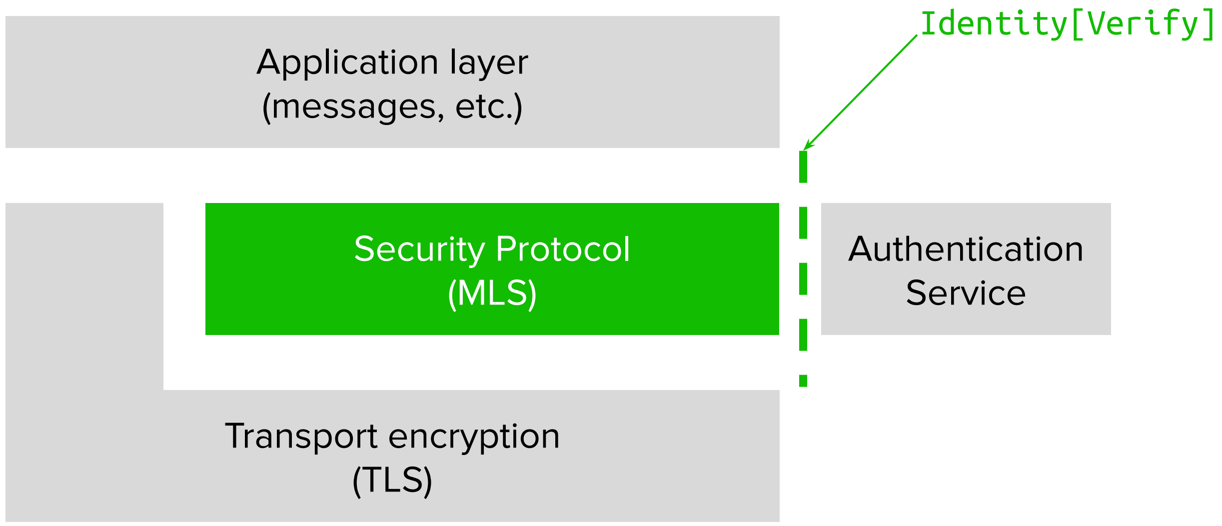
2016: Stakeholders gather

Scope of TLS

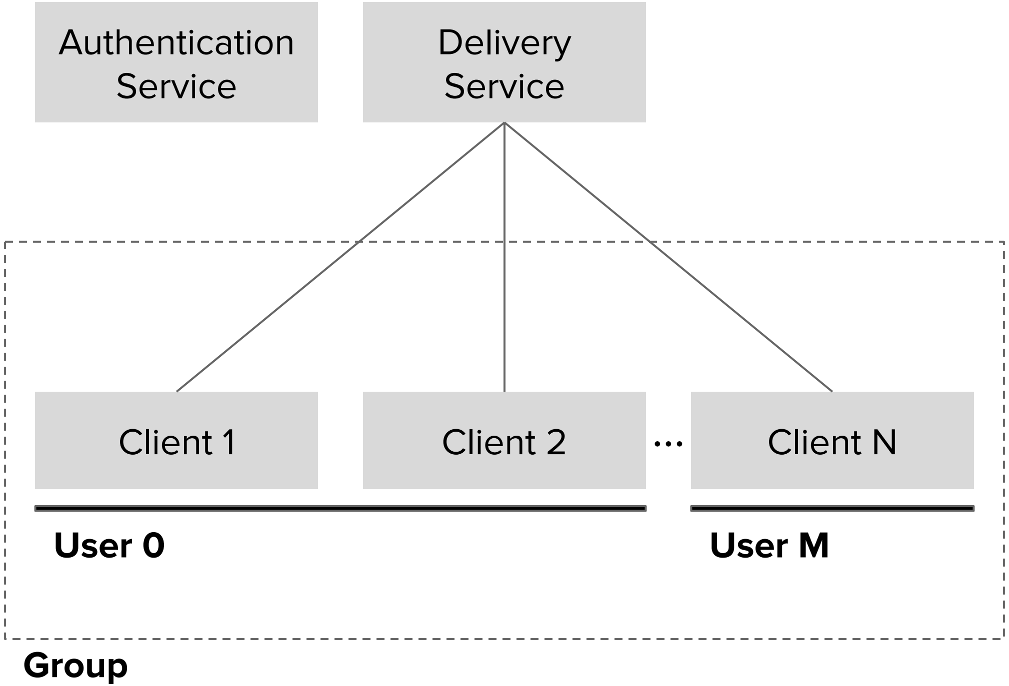


Certificate[Verify]
Message Content
(HTTP, SMTP, SIP, ...)
Security Protocol
(TLS / DTLS)
Authentication
(PKI)
Transport
(TCP / UDP)

Scope of MLS


Identity[Verify]
Application layer
(messages, etc.)
Security Protocol
(MLS)
Authentication
Service
Transport encryption
(TLS)

Architecture

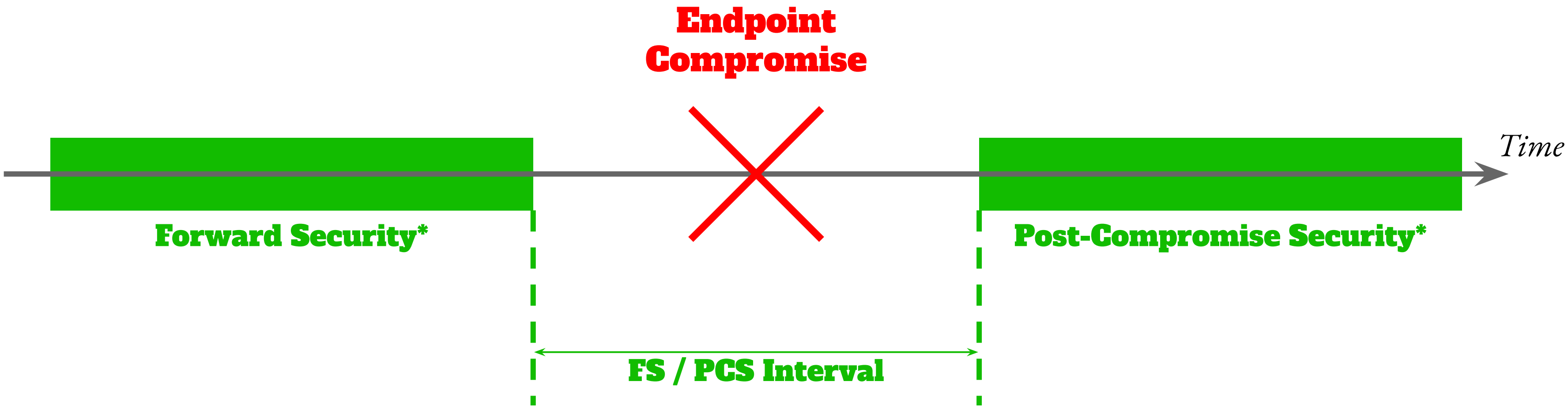


MLS vs. TLS

- Lots of people - 2 vs. 10^N
- Long lived sessions - seconds vs. months
- Lots of mobile devices involved
- Designed for human-to-human communication

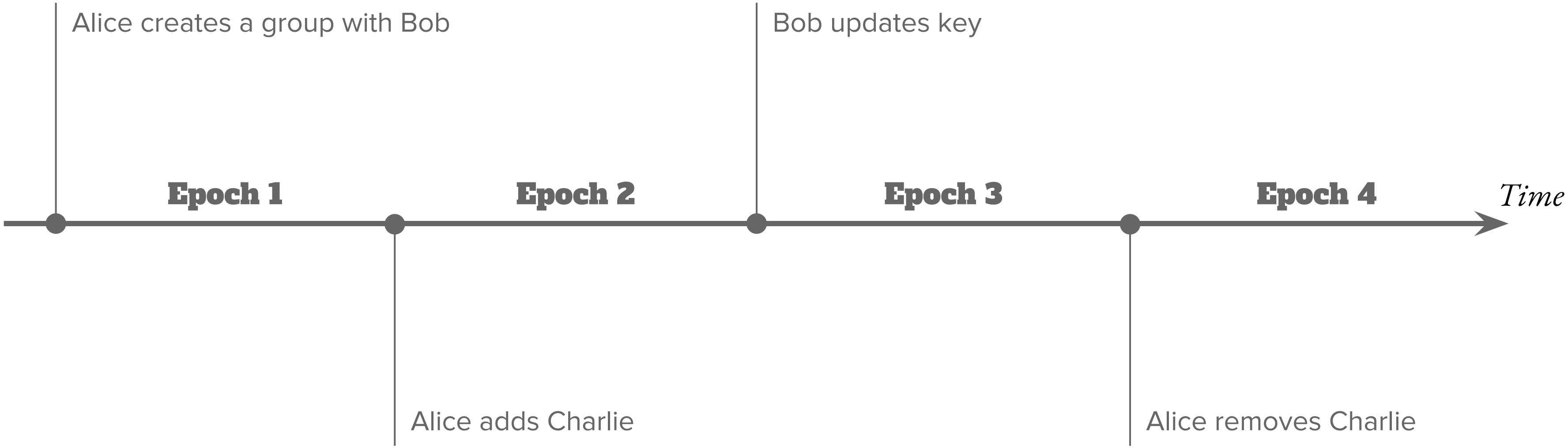
Significant probability that some member is compromised at some time in the life of the session

FS & PCS



* ... with regard to a member

Key rotation



Key rotation

Before we start

Every client/member publishes **Init Keys** ahead of time

Init Keys are handled by the **Delivery Service**

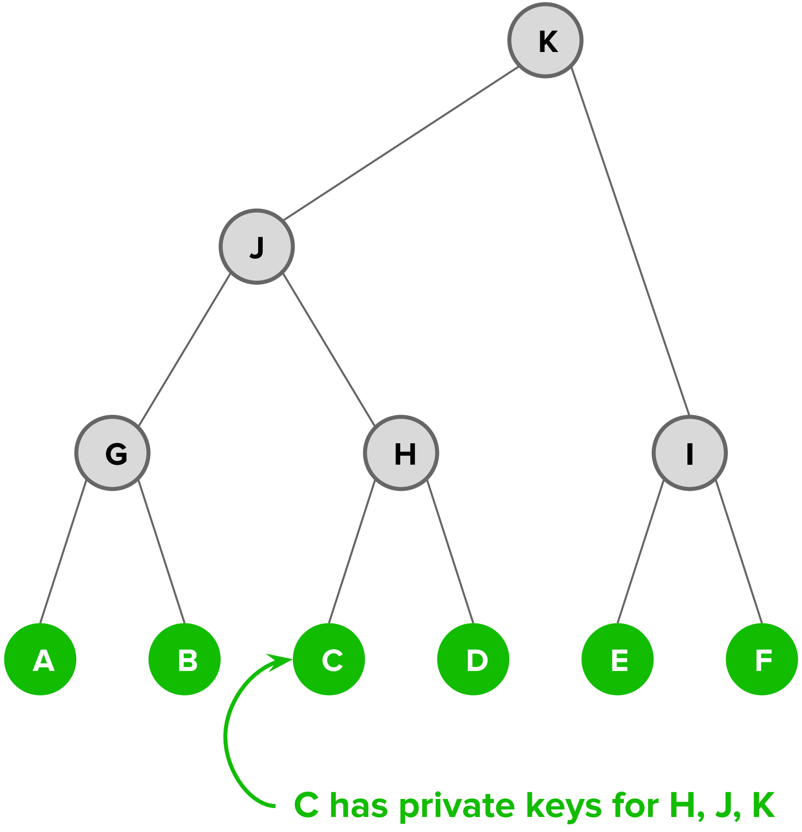
They contain credentials and a public key, so that we can encrypt data to them

The core: TreeKEM

The public state of a group is composed of a left-balanced binary tree of asymmetric public keys

Each member of the group occupies a leaf and knows all secrets in its path to the root.

Secrecy invariant: The private key for an intermediate node is only known to members of the subtree.



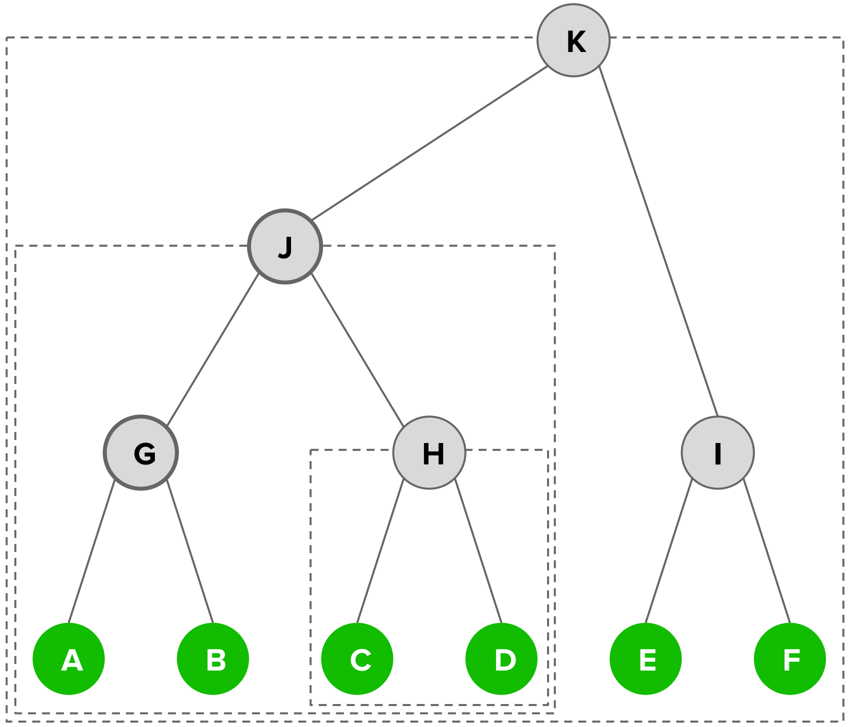
Trees of Keys

This has a couple of nice consequences:

Intermediate nodes represent subgroups you can KEM / encrypt to

Root private key is known to everyone in the group at a given time

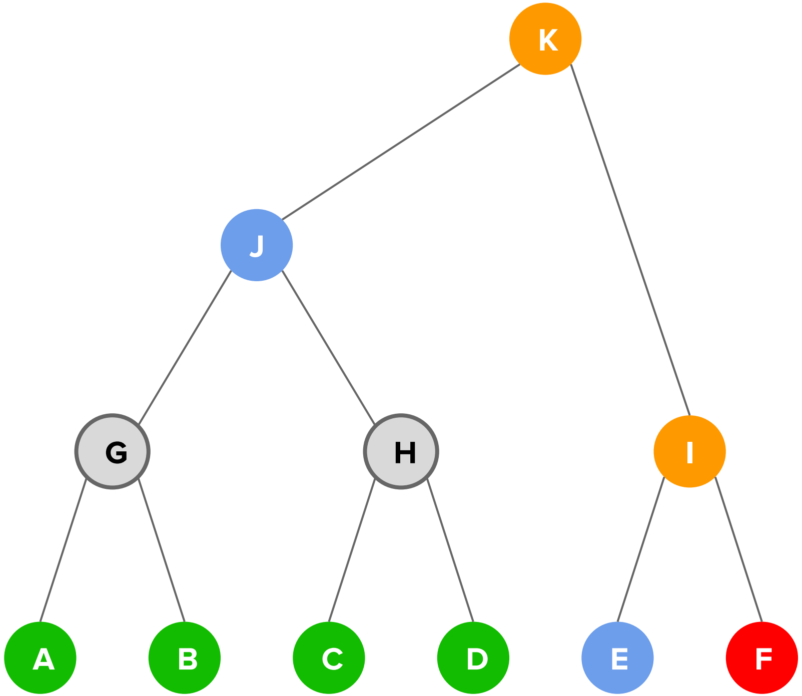
Protocol maintains this state through group operations (**Create**, **Add**, **Update**, **Remove**)



Update

F wants to do an **Update**

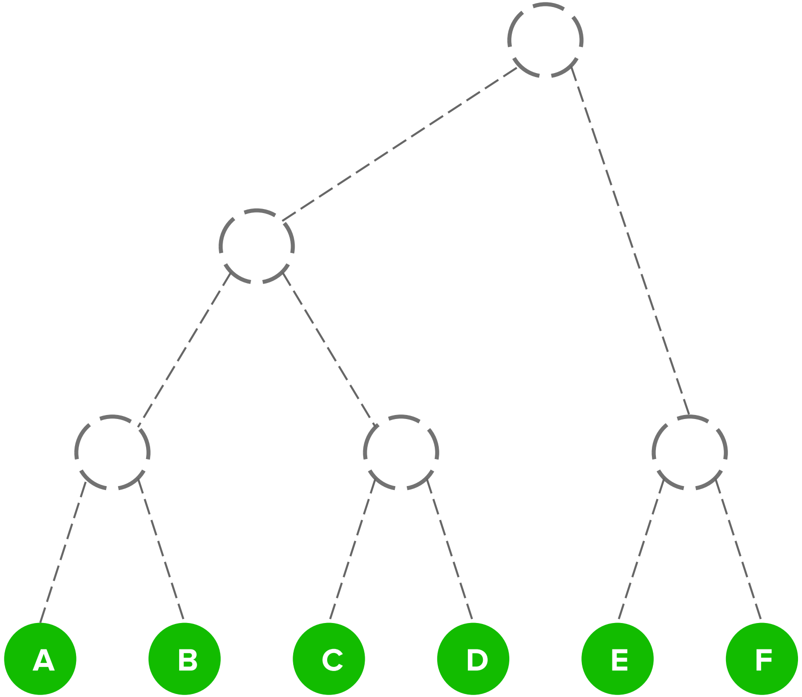
- It generates a fresh leaf key
- Hashes up to the root along the direct path
- Encrypts new values to the co-path



Add/Init

We want to add members to a group

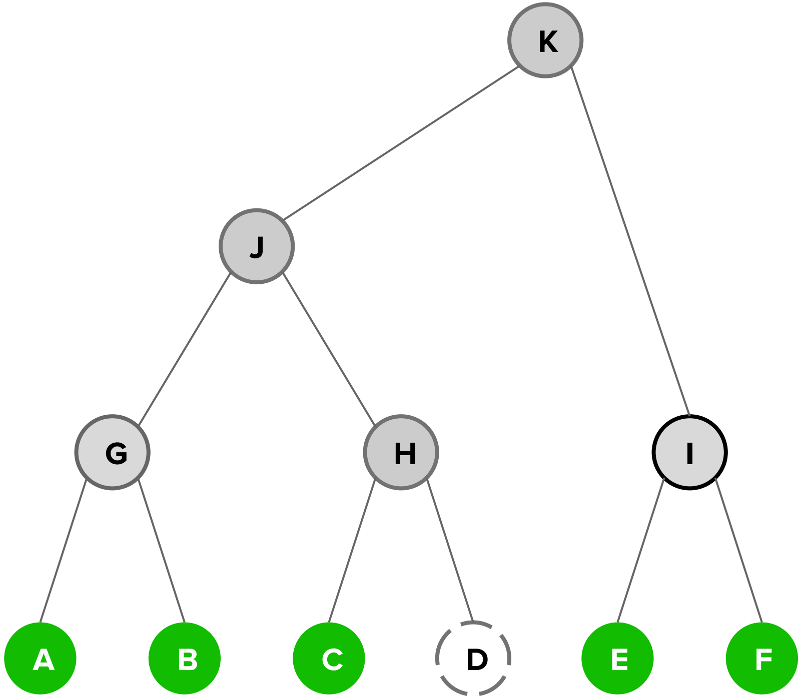
- We fetch init keys for every member
- New members get added as new leaves to the tree
- Newly added members will do an **Update** when they come online



Remove

A wants to remove **D**

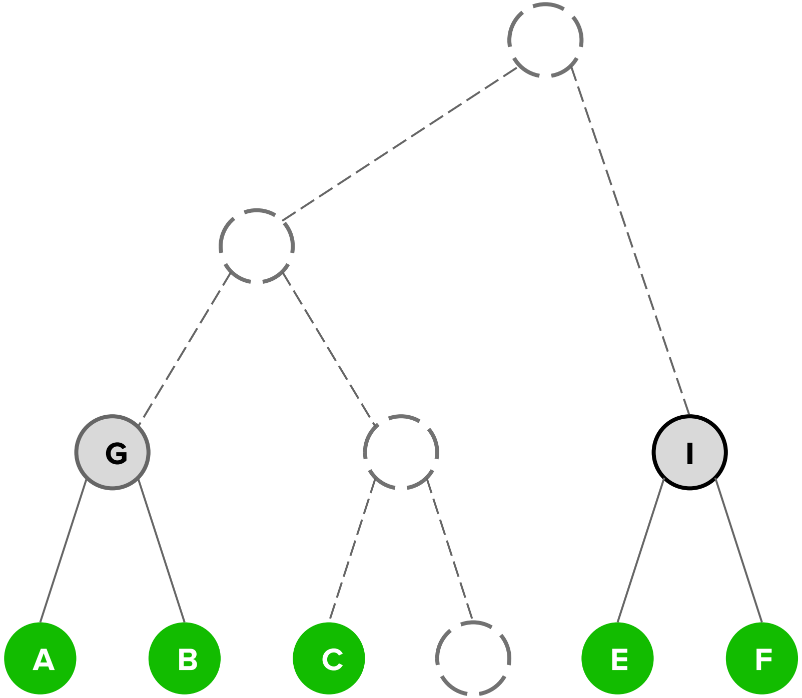
- **A** sends a message to the group saying **D** should be removed



Remove

A wants to remove **D**

- **A** sends a message to the group saying **D** should be removed
- The direct path of **D** is blanked
- Therefore **D** does not know any tree secret
- **A** can do an **Update** to derive a group secret unknown to **D**



Messages

Handshake messages

- Control messages (**Create**, **Add**, **Update**, **Remove**) with global order

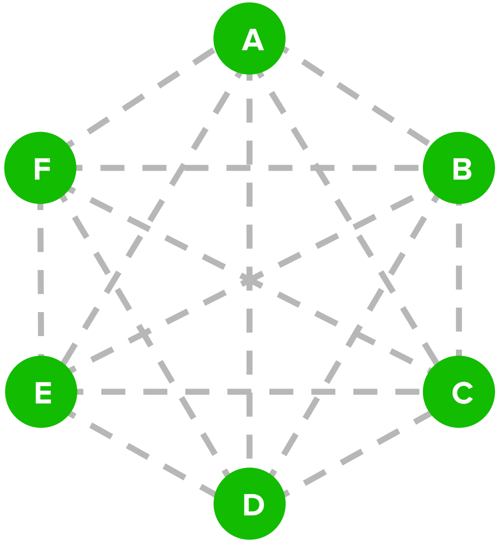
Application messages

- Typically text messages, but could be any data, with per-sender order

The Delivery Service enforces ordering of handshake messages

Efficiency

Pairwise protocols superpose 1:1 connections in a group (full mesh)



Efficiency

Pairwise sending:

- Sending messages is in **O(N)**

Sender keys:

- Fan-out an **encryption key** to everyone and use it for messages
- Sending the **encryption key** out is still in **O(N)**, sending a **message** is in **O(1)**
- Problem: if a member leaves the group, everyone has to fan-out a new key in **O(N^2)**

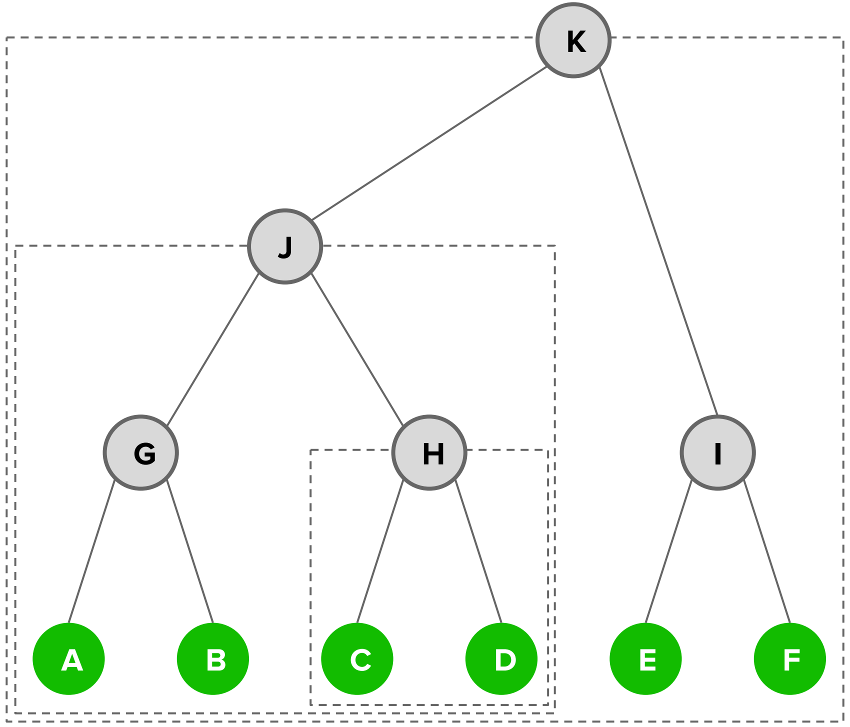
Efficiency

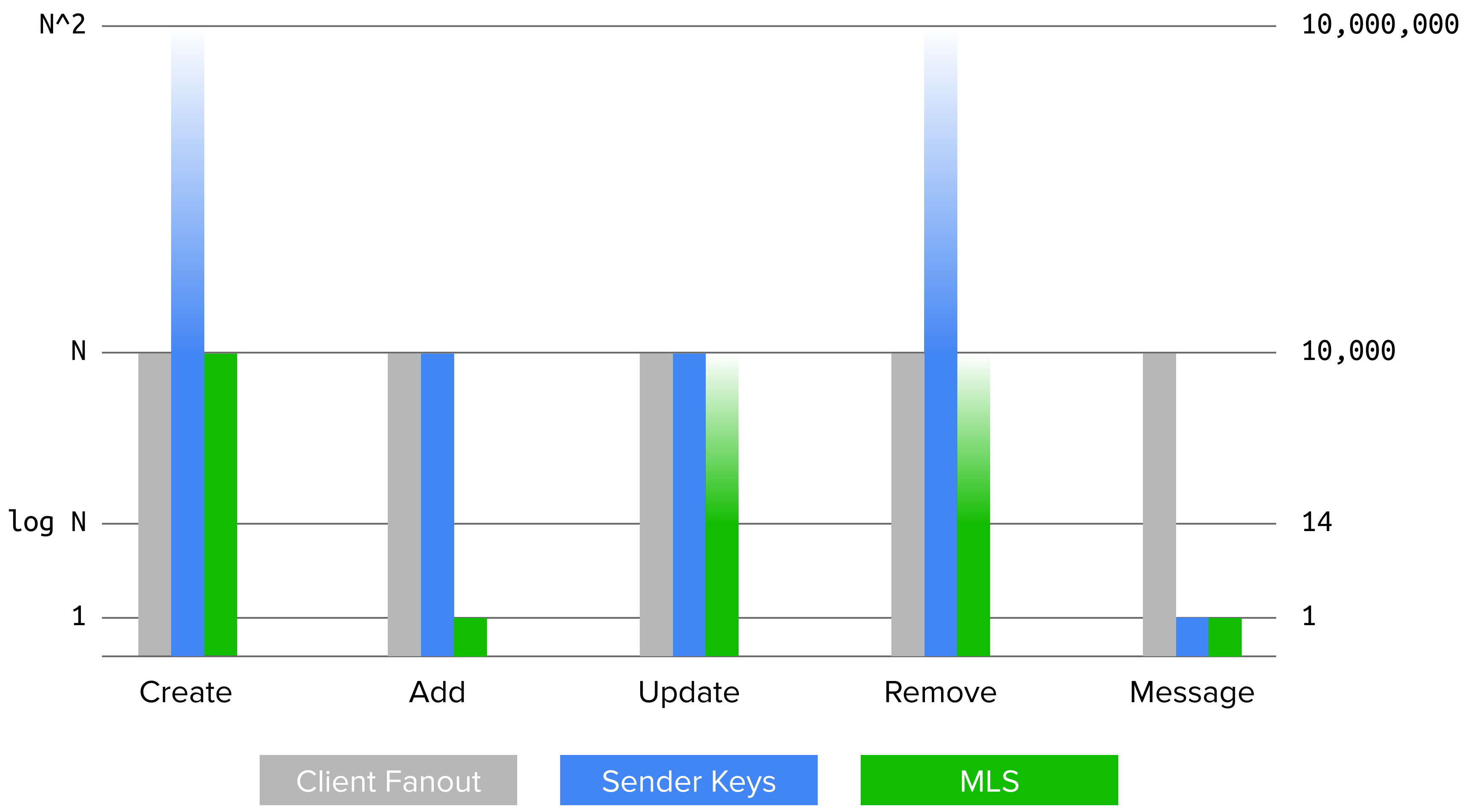
MLS allows to maintain a group secret in **O(log N)** by using left-balanced binary trees

Example: 100k members and message size of 1kb

Pairwise: 100k operations and payload of 100k * 1kb = 100mb

MLS: 17 operations and payload of 17 * 1kb = 17kb




N^2
N
log N
1
10,000,000
10,000
14
1
Create
Add
Update
Remove
Message
Client Fanout
Sender Keys
MLS

Metadata protection

Message content is secret because of end-to-end encryption

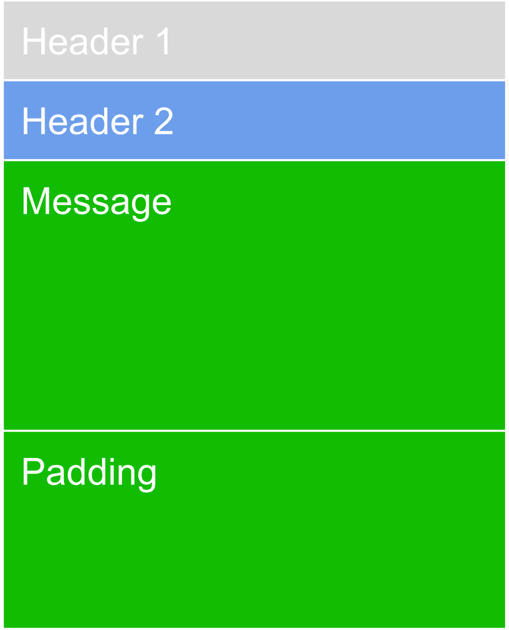
What data should we try to protect additionally?

There are two kinds of metadata:

- Observable metadata
- Persisted metadata

Metadata protection

- Servers will keep messages in queues, we just need to tell the server in which queue to save the message
- We can encrypt the sender of a message, the server doesn't need to have that information
- We can have arbitrary padding, so that clients can make messages indistinguishable from each other



Deniability

Messages are signed for authenticity

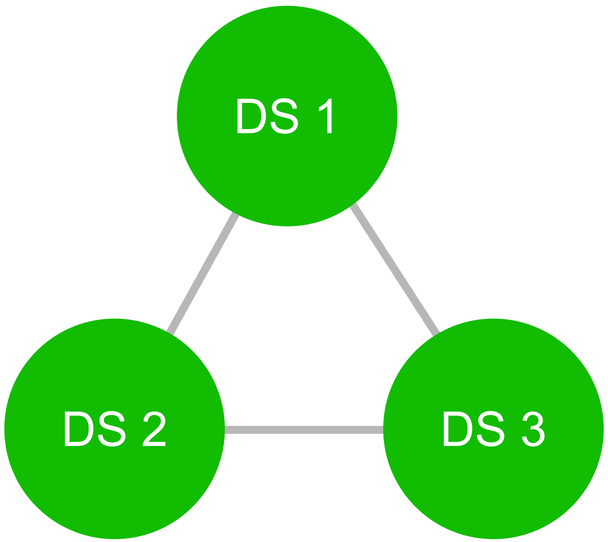
- Identity keys can be transferred over 1:1 deniable channels
- Authorship of text messages becomes deniable
- Participation in a group becomes deniable
- Messages are also encrypted under the group key, therefore opaque to the server

Federation

Are we limited to one **Delivery Service**?

Ordering for handshake messages is important

If we can **distribute** the ordering problem across multiple delivery services, **federation** becomes possible.



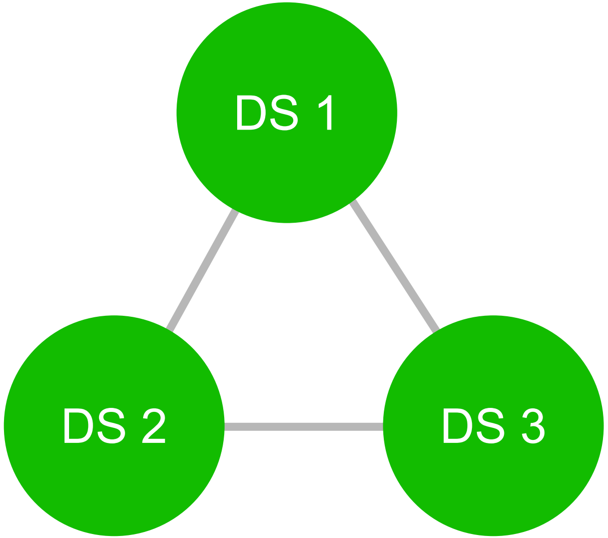
Federation

<u>**Federation without redundancy**</u>

Simple approach: designate which **Delivery Service** is responsible for the ordering

<u>**Federation with redundancy**</u>

More advanced approach: have some consensus among the **Delivery Services** on which one is responsible for ordering



Business messaging

Business communication is seeing a transformation from using email towards using messaging.

This change is driven by consumer experience.

Business messaging

The encryption challenge

Most solutions only use transport encryption (TLS) to protect messages and files.

End-to-end encryption is challenging at scale.

Business messaging

<u>**The feature challenge**</u>

Most solutions only enable users of the same organisation to talk to each other.

Email is still popular as a legacy technology, because anyone can be reached.

Federation contributed to the popularity of email.

Summary

- MLS aims to be new standard for secure messaging, especially in (large) groups
- Modern security properties
- Robust, usable open specification
- Usable solution for new and existing products

More information: messaginglayersecurity.rocks

Thanks

Benjamin Beurdouche
@beurdouche – inria.fr

Katriel Cohn-Gordon
@katrielalex – katriel.co.uk

Raphael Robert
@raphaelrobert – wire.com