Unidirectional Group Messaging: Simple, Secure, and Efficient Solutions

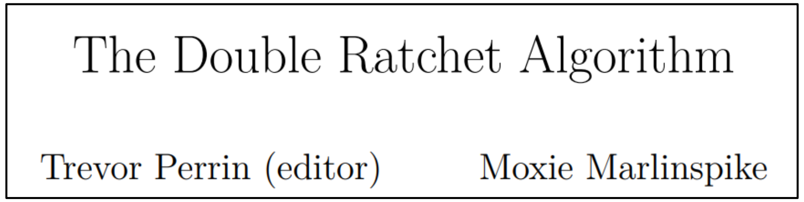
Cryptographic Applications Workshop

February 23

Real-World Cryptography Group
FAU Erlangen-Nürnberg, Germany

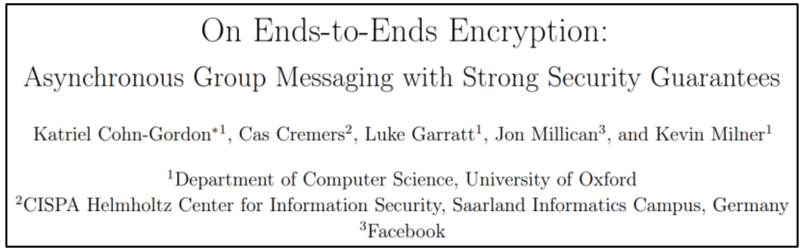
Daniel Collins and **Paul Rösler**

(Group) Messaging is Complex



The Double Ratchet Algorithm

Trevor Perrin (editor) Moxie Marlinspike

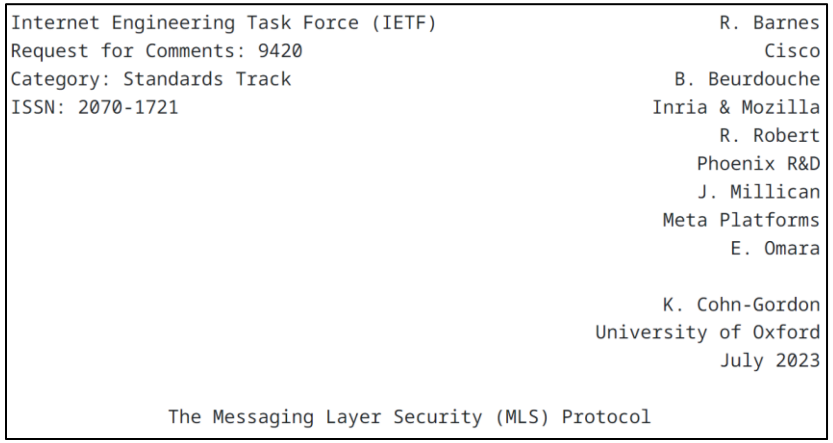


On Ends-to-Ends Encryption:

Asynchronous Group Messaging with Strong Security Guarantees

Katriel Cohn-Gordon*[1], Cas Cremers[2], Luke Garratt[1], Jon Millican[3], and Kevin Milner[1]

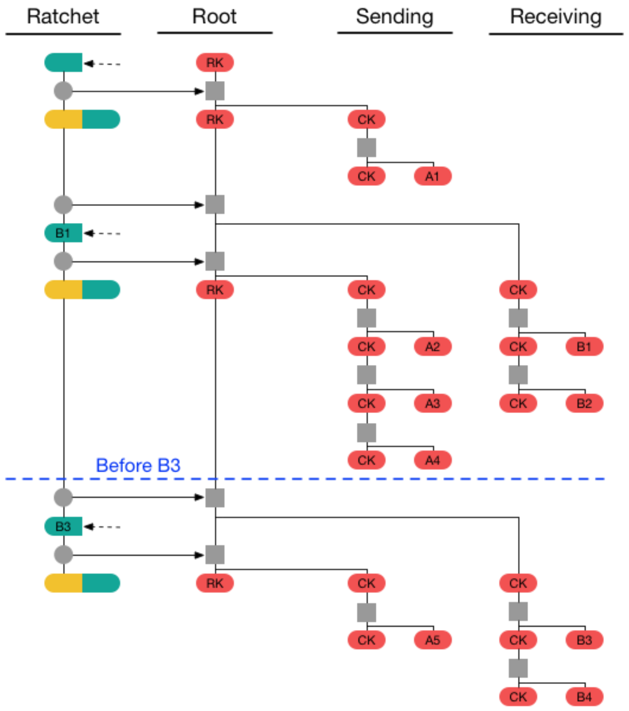
[1]Department of Computer Science, University of Oxford
[2]CISPA Helmholtz Center for Information Security, Saarland Informatics Campus, Germany
[3]Facebook

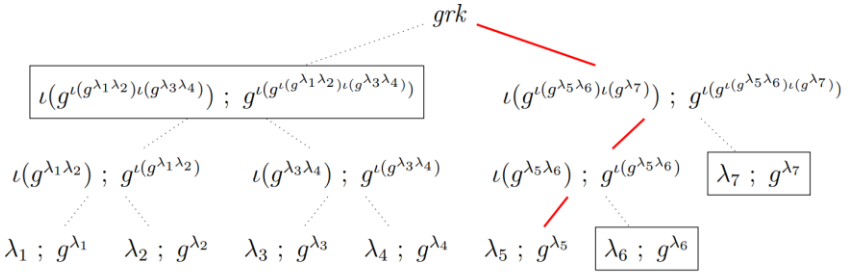


```
Internet Engineering Task Force (IETF)                        R. Barnes
Request for Comments: 9420                                        Cisco
Category: Standards Track                                 B. Beurdouche
ISSN: 2070-1721                                       Inria & Mozilla
                                                              R. Robert
                                                            Phoenix R&D
                                                            J. Millican
                                                         Meta Platforms
                                                               E. Omara

                                                         K. Cohn-Gordon
                                                   University of Oxford
                                                              July 2023

               The Messaging Layer Security (MLS) Protocol
```




grk
$\iota(g^{\iota(g^{\lambda_1\lambda_2})\iota(g^{\lambda_3\lambda_4})})\ ;\ g^{\iota(g^{\iota(g^{\lambda_1\lambda_2})\iota(g^{\lambda_3\lambda_4})})}$
$\iota(g^{\iota(g^{\lambda_5\lambda_6})\iota(g^{\lambda_7})})\ ;\ g^{\iota(g^{\iota(g^{\lambda_5\lambda_6})\iota(g^{\lambda_7})})}$
$\iota(g^{\lambda_1\lambda_2})\ ;\ g^{\iota(g^{\lambda_1\lambda_2})}$
$\iota(g^{\lambda_3\lambda_4})\ ;\ g^{\iota(g^{\lambda_3\lambda_4})}$
$\iota(g^{\lambda_5\lambda_6})\ ;\ g^{\iota(g^{\lambda_5\lambda_6})}$
$\lambda_7\ ;\ g^{\lambda_7}$
$\lambda_1\ ;\ g^{\lambda_1}$
$\lambda_2\ ;\ g^{\lambda_2}$
$\lambda_3\ ;\ g^{\lambda_3}$
$\lambda_4\ ;\ g^{\lambda_4}$
$\lambda_5\ ;\ g^{\lambda_5}$
$\lambda_6\ ;\ g^{\lambda_6}$

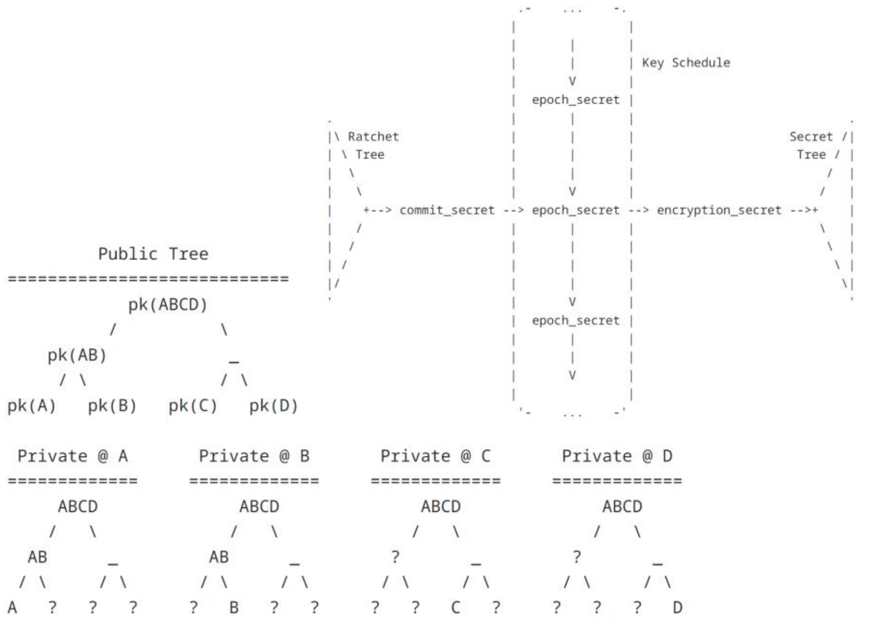
$$\begin{tikzcd}
 & \pi.tk \arrow[d] & & \pi.tk' \arrow[d] & \\
\pi.sk \arrow[r] & \boxed{\mathrm{KDF}} \arrow[r] & \pi.sk' \arrow[r] & \boxed{\mathrm{KDF}} \arrow[r] & \pi.sk''
\end{tikzcd}$$

[MLS] TreeKEM: An alternative to ART

Eric Rescorla <ekr@rtfm.com> Thu, 03 May 2018 14:27 UTCShow header

```
Hi folks,

Several of us (Karthik, Richard, and I) have been working on an
alternative to ART which we call TreeKEM. TreeKEM parallels ART in
many ways, but is more cryptographically efficient and is much better
at handling concurrent changes. The most common behaviors (updating
ones own key) can be executed completely concurrently, merging all the
requested changes.
```



```
                                  .-      ...      -.
                                  |                 |
                                  |        |        |
                                  |        |        | Key Schedule
                                  |        V        |
                                  |  epoch_secret   |
           .                      |        |        |                            .
           |\ Ratchet             |        |        |                 Secret  /|
           | \ Tree               |        |        |                  Tree  / |
           |  \                   |        |        |                       /  |
           |   \                  |        V        |                      /   |
           |    +--> commit_secret --> epoch_secret --> encryption_secret -->+    |
           |   /                  |        |        |                      \   |
           |  /                   |        |        |                       \  |
           | /                    |        |        |                        \ |
           |/                     |        |        |                         \|
           '                      |        V        |                          '
                                  |  epoch_secret   |
                                  |        |        |
                                  |        |        |
                                  |        V        |
                                  |                 |
                                  '-      ...      -'
```

```
      Public Tree
==============================
          pk(ABCD)
        /          \
    pk(AB)          _
     / \           / \
pk(A)  pk(B)  pk(C)  pk(D)

 Private @ A       Private @ B       Private @ C       Private @ D
=============     =============     =============     =============
     ABCD              ABCD              ABCD              ABCD
    /    \            /    \            /    \            /    \
   AB     _          AB     _          ?      _          ?      _
  / \    / \        / \    / \        / \    / \        / \    / \
 A   ?  ?   ?      ?   B  ?   ?      ?   ?  C   ?      ?   ?  ?   D
```

(Group) Messaging is Complex

Properties & Features:

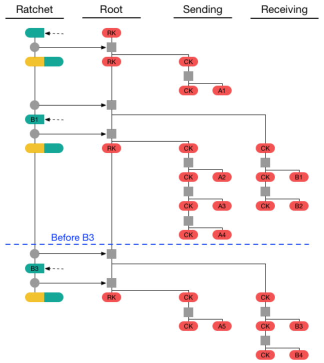
- Active security
- Unreliable network
- Dynamic membership
- Administration
- Malicious insiders
- Concurrency
- …

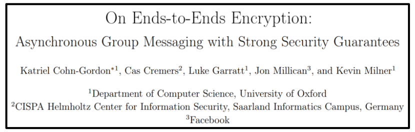
Simplifications:

- Passive adversaries
- Round-based / synchronous / reliable / etc. network
- Static group
- Honest members
    - Honest deletion
- …

The Double Ratchet Algorithm

Trevor Perrin (editor) Moxie Marlinspike

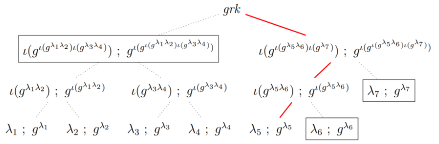




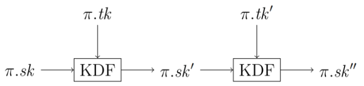
On Ends-to-Ends Encryption:
Asynchronous Group Messaging with Strong Security Guarantees

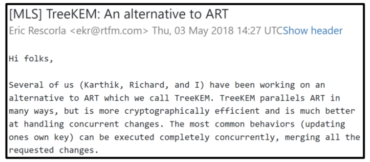
Katriel Cohn-Gordon*[1], Cas Cremers[2], Luke Garratt[1], Jon Millican[3], and Kevin Milner[1]

[1] Department of Computer Science, University of Oxford
[2] CISPA Helmholtz Center for Information Security, Saarland Informatics Campus, Germany
[3] Facebook



grk
$\iota(g^{\iota(g^{\lambda_1\lambda_2})\iota(g^{\lambda_3\lambda_4})})\ ;\ g^{\iota(g^{\iota(g^{\lambda_1\lambda_2})\iota(g^{\lambda_3\lambda_4})})}$
$\iota(g^{\iota(g^{\lambda_5\lambda_6})\iota(g^{\lambda_7})})\ ;\ g^{\iota(g^{\iota(g^{\lambda_5\lambda_6})\iota(g^{\lambda_7})})}$
$\iota(g^{\lambda_1\lambda_2})\ ;\ g^{\iota(g^{\lambda_1\lambda_2})}$
$\iota(g^{\lambda_3\lambda_4})\ ;\ g^{\iota(g^{\lambda_3\lambda_4})}$
$\iota(g^{\lambda_5\lambda_6})\ ;\ g^{\iota(g^{\lambda_5\lambda_6})}$
$\lambda_7\ ;\ g^{\lambda_7}$
$\lambda_1\ ;\ g^{\lambda_1}$
$\lambda_2\ ;\ g^{\lambda_2}$
$\lambda_3\ ;\ g^{\lambda_3}$
$\lambda_4\ ;\ g^{\lambda_4}$
$\lambda_5\ ;\ g^{\lambda_5}$
$\lambda_6\ ;\ g^{\lambda_6}$



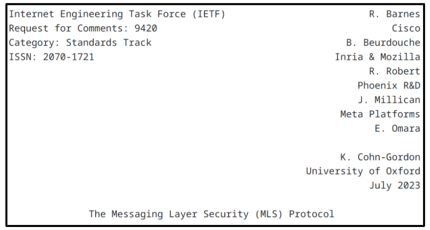


[MLS] TreeKEM: An alternative to ART

Eric Rescorla <ekr@rtfm.com> Thu, 03 May 2018 14:27 UTCShow header

Hi folks,

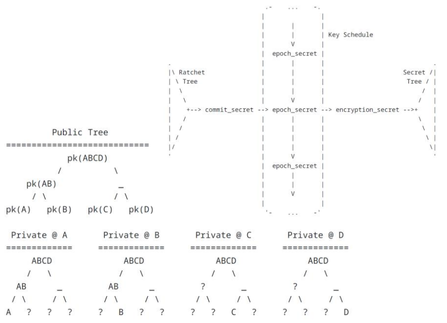
Several of us (Karthik, Richard, and I) have been working on an alternative to ART which we call TreeKEM. TreeKEM parallels ART in many ways, but is more cryptographically efficient and is much better at handling concurrent changes. The most common behaviors (updating ones own key) can be executed completely concurrently, merging all the requested changes.

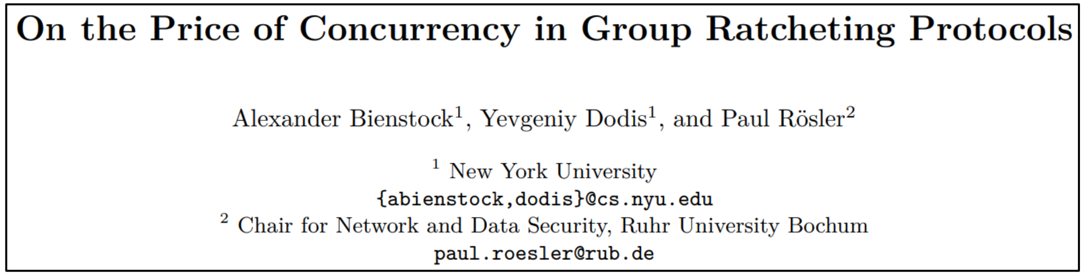


```
Internet Engineering Task Force (IETF)                         R. Barnes
Request for Comments: 9420                                         Cisco
Category: Standards Track                                  B. Beurdouche
ISSN: 2070-1721                                          Inria & Mozilla
                                                               R. Robert
                                                             Phoenix R&D
                                                              J. Millican
                                                          Meta Platforms
                                                                E. Omara

                                                          K. Cohn-Gordon
                                                    University of Oxford
                                                               July 2023


              The Messaging Layer Security (MLS) Protocol
```





On the Price of Concurrency in Group Ratcheting Protocols

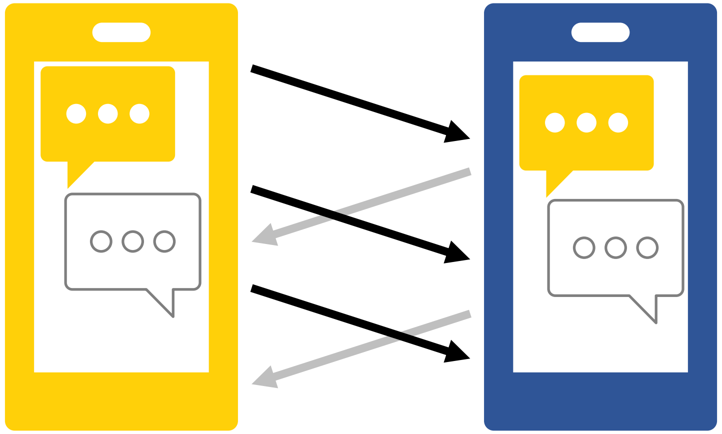
Alexander Bienstock[1], Yevgeniy Dodis[1], and Paul Rösler[2]

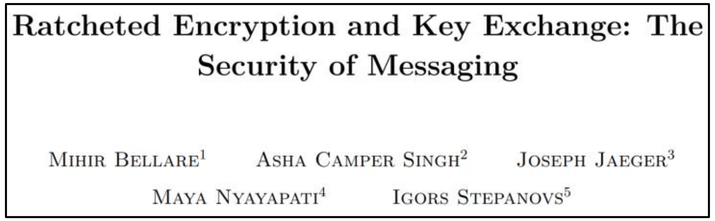
[1] New York University
{abienstock,dodis}@cs.nyu.edu
[2] Chair for Network and Data Security, Ruhr University Bochum
paul.roesler@rub.de

Systematic Simplification

JJ: Proofs (and definitions) are complicated!

*Uni*directional



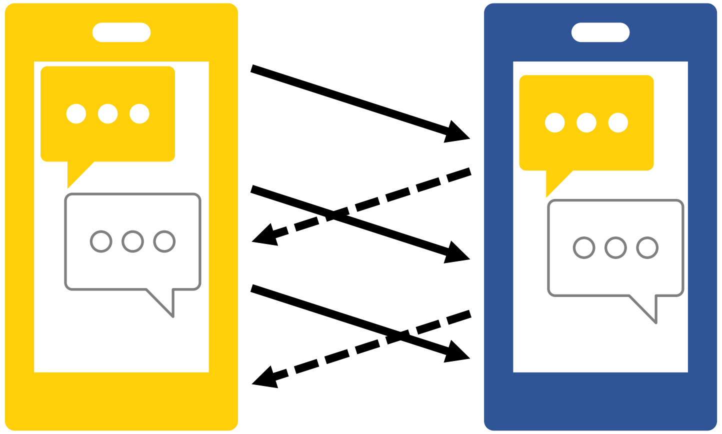


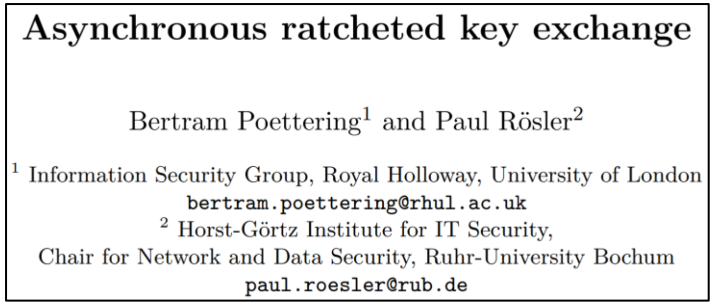
Ratcheted Encryption and Key Exchange: The Security of Messaging

Mihir Bellare[1] Asha Camper Singh[2] Joseph Jaeger[3]
Maya Nyayapati[4] Igors Stepanovs[5]

- Symmetric key chain
- Continuous PKE
- Asymmetric key chain

*Sesqi*directional





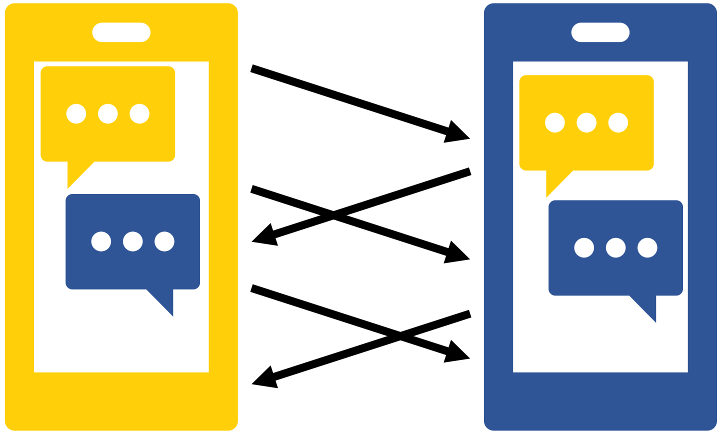
Asynchronous ratcheted key exchange

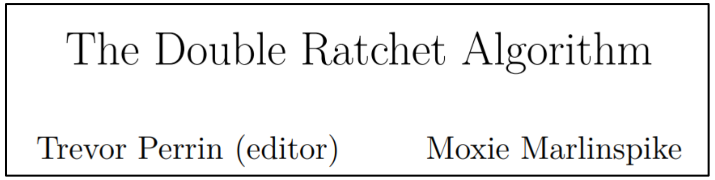
Bertram Poettering[1] and Paul Rösler[2]

[1] Information Security Group, Royal Holloway, University of London
bertram.poettering@rhul.ac.uk
[2] Horst-Görtz Institute for IT Security,
Chair for Network and Data Security, Ruhr-University Bochum
paul.roesler@rub.de

- Signature chain
- Asynchronous key updates

*Bi*directional



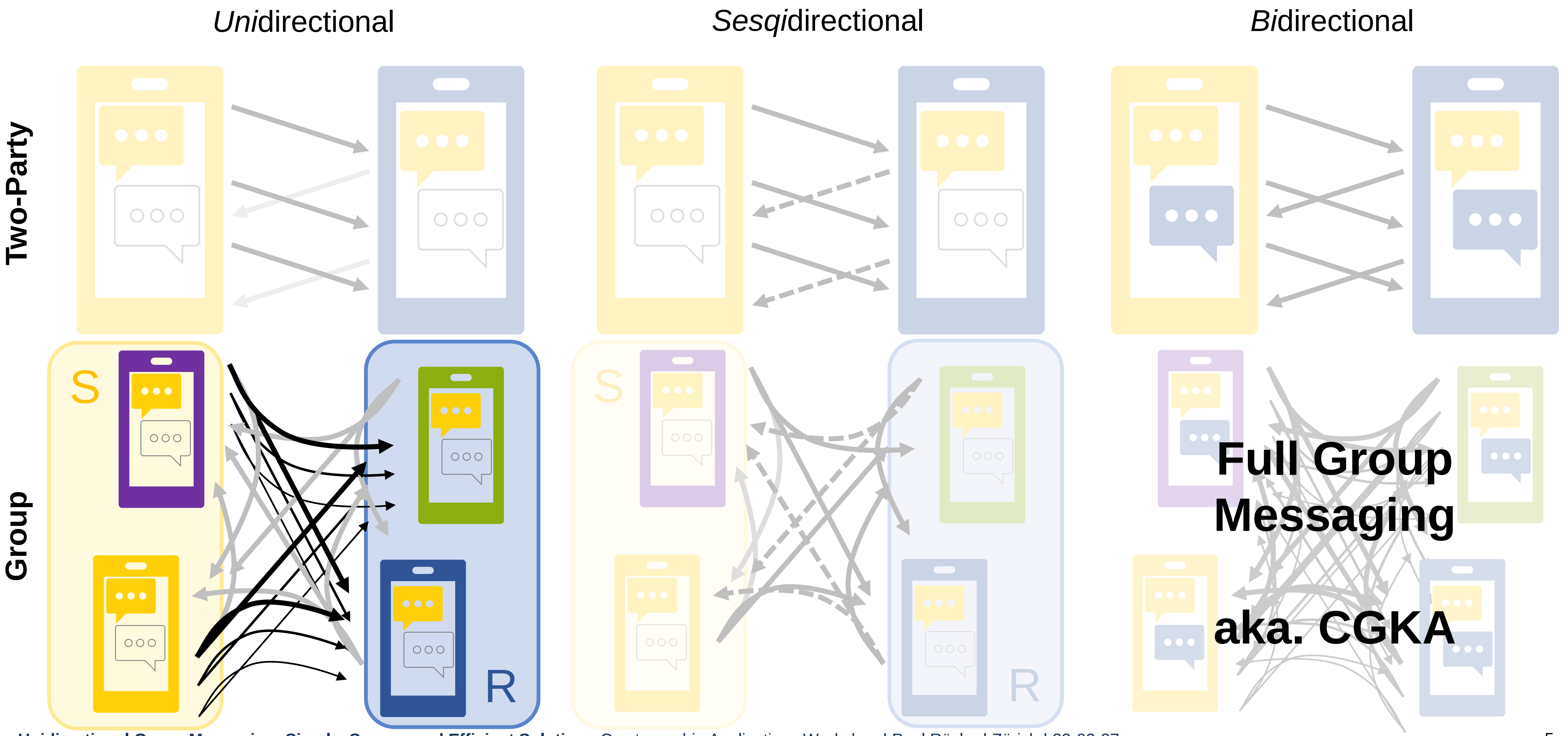


The Double Ratchet Algorithm

Trevor Perrin (editor) Moxie Marlinspike

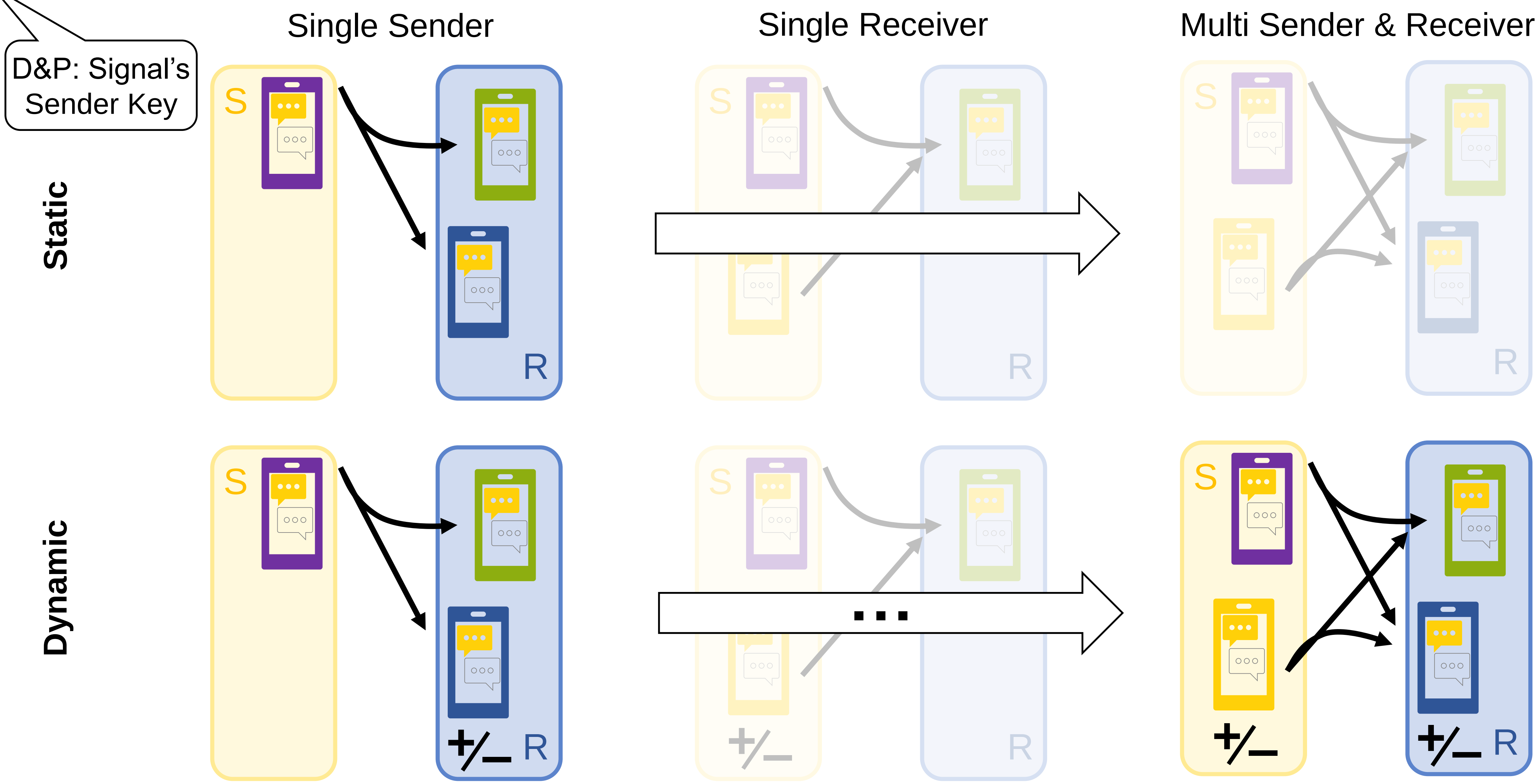
- Composition of 2x *Sesqui*directional

Systematic Simplification



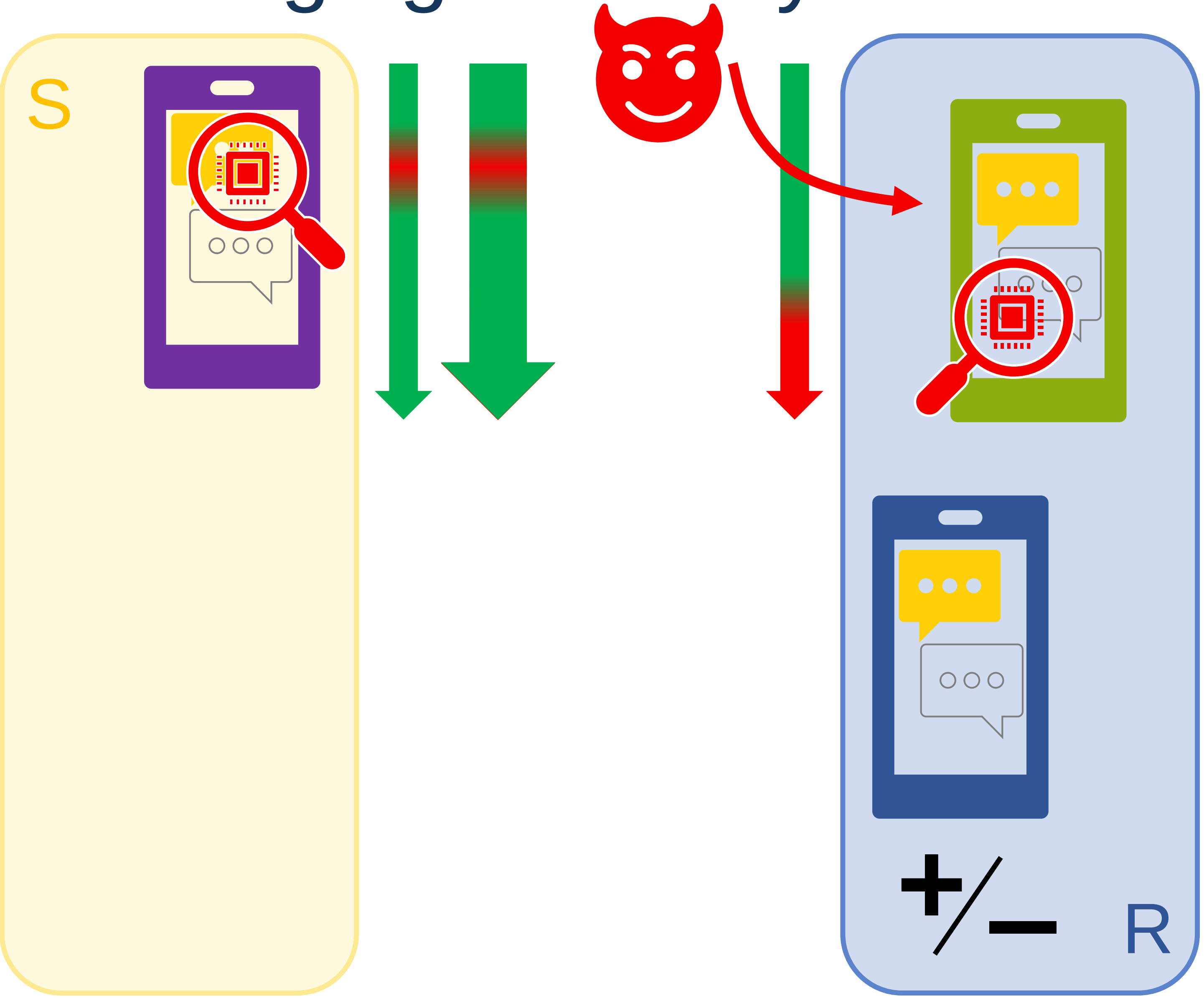
Unidirectional Group Messaging

D&P: Signal's Sender Key



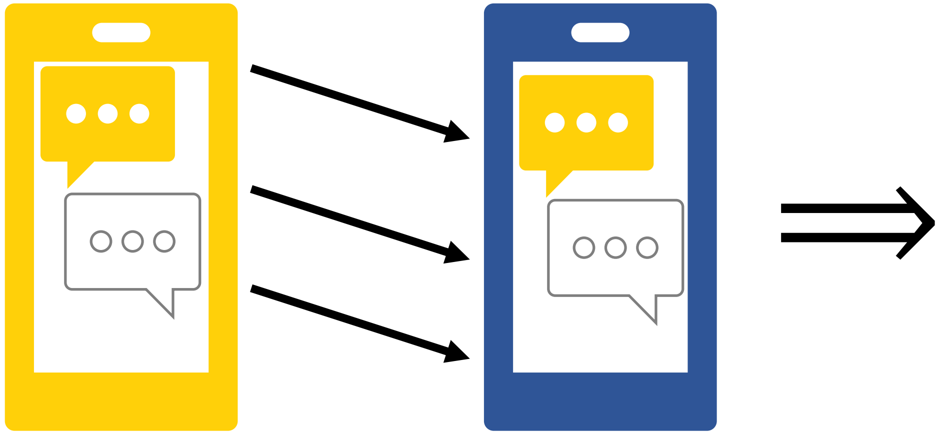
Unidirectional Group Messaging: Security

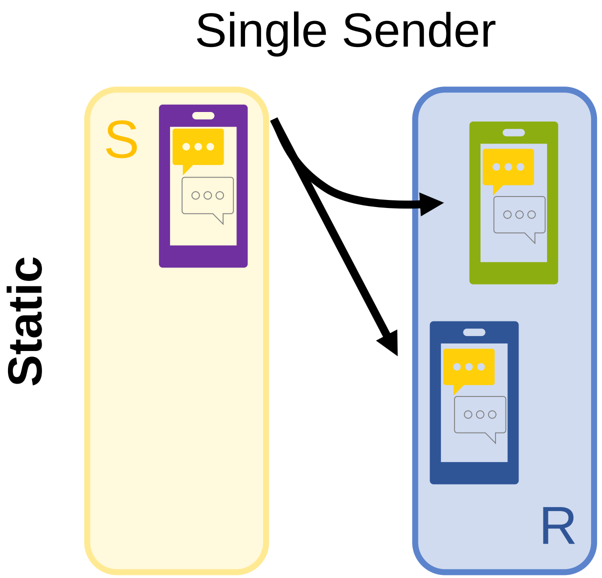
- Forward Security for both
- Post-Compromise Security for Sender
- Diverging upon Impersonation



Static Group: Construction

- Forward Security for both
- Post-Compromise Security for Sender
- Diverging upon Impersonation
- Optimal Performance ☺

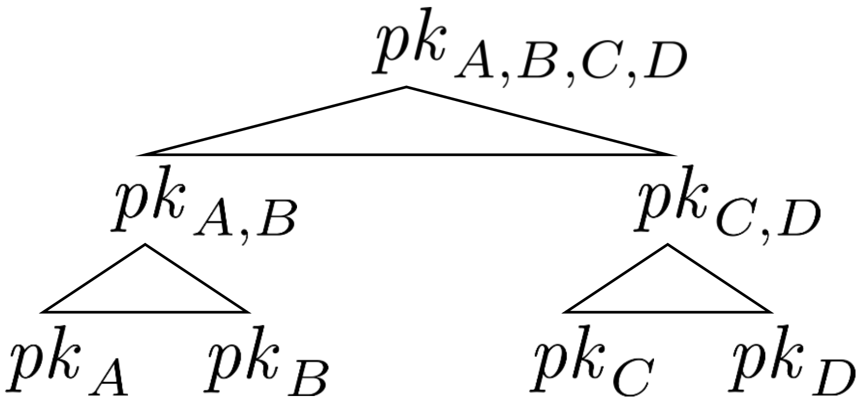


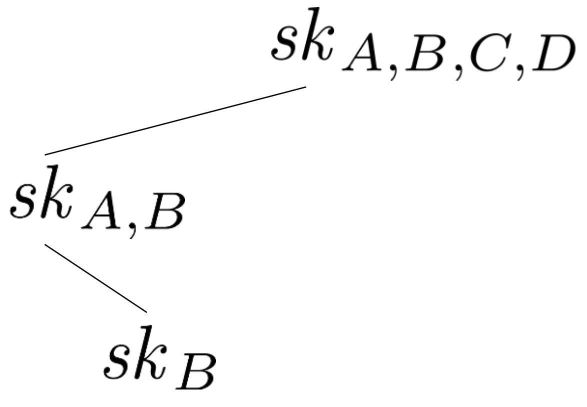


```
$\underline{\mathrm{send}(pk, m):}$
$k \leftarrow_{\$}$
$c \leftarrow \mathrm{enc}(pk, (k, m))$
$sk \leftarrow \mathrm{H}(k, pk, c)$
$pk \leftarrow \mathrm{gen}(sk)$
Return $(pk, c)$
```

```
$\underline{\mathrm{recv}(sk, c):}$
$(k, m) \leftarrow \mathrm{dec}(sk, c)$
$sk \leftarrow \mathrm{H}(k, pk, c)$
Return $(sk, m)$
```

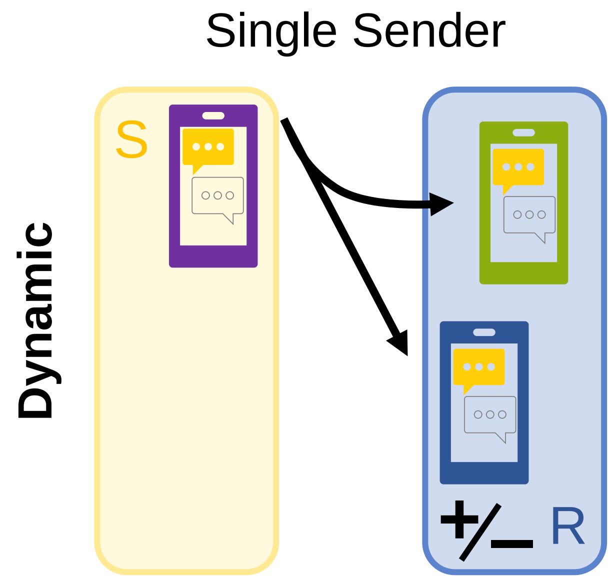
Dynamic Group, Single Sender: Construction





```
$(sk^*, pk^*) \leftarrow \text{gen}$
For all $i$ in tree:
    $k \leftarrow \text{eval}(sk^*, pk_i)$
    $sk_i \leftarrow \text{H}(k, pk_i, pk^*)$
    $pk_i \leftarrow \text{gen}(sk_i)$
```

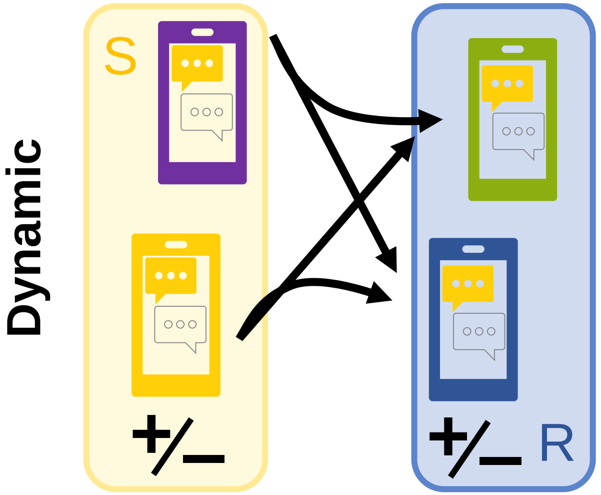
```
For all $i$ on path:
    $k \leftarrow \text{eval}(sk_i, pk^*)$
    $sk_i \leftarrow \text{H}(k, pk_i, pk^*)$
```

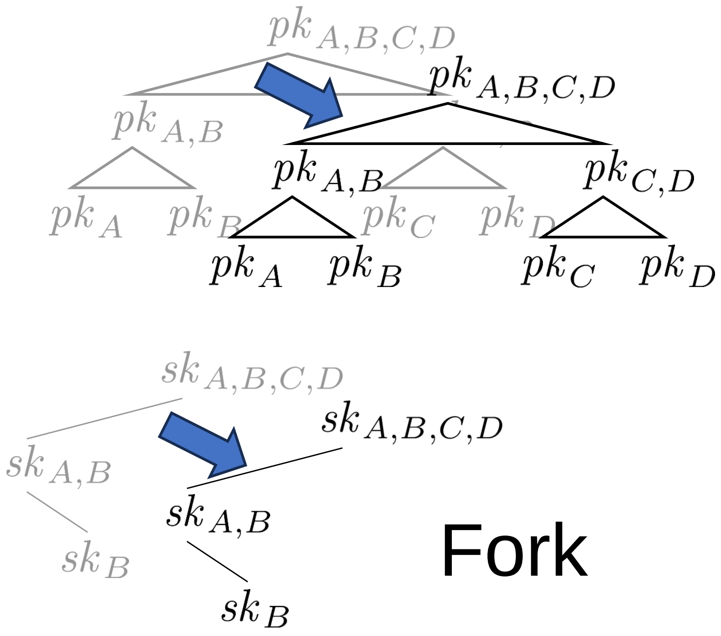


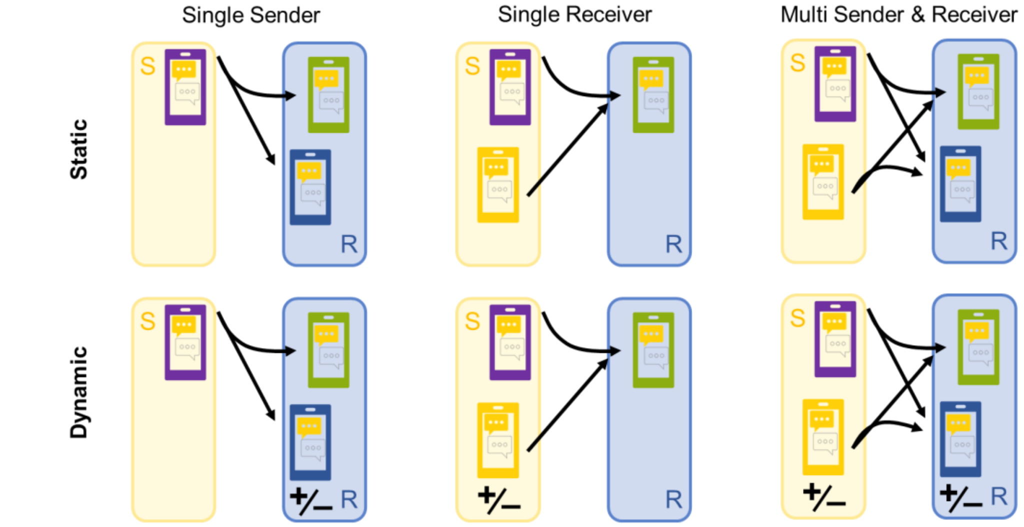
- FS for both
- PCS for Sender
- Diverging upon Impersonation
- Small ciphertexts

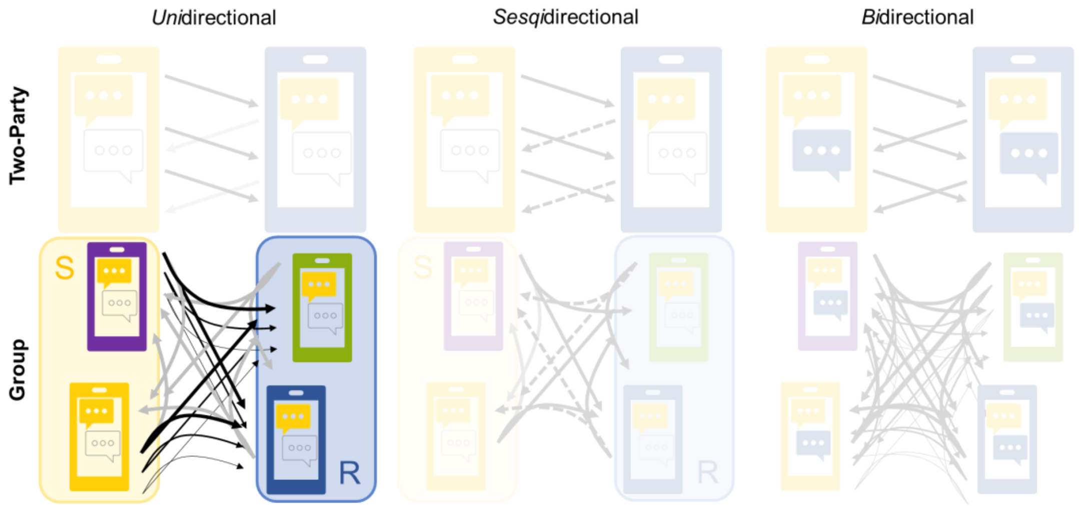
Outlook & Summary

Multi Sender & Receiver









- Efficient
- Stronger Security
- Open:
  - *Sesqui*directional
  - Malicious Senders
  - Unreliable Network

roeslpa.de

Open Discussion

- Sender Keys and Unidirectional Messaging:
  - Simple
  - Core: Forward Security
- Simplicity $\overset{?}{\Rightarrow}$ Verifiability / Trust
- MLS flexible but complex
- What are your thoughts?