CISPA
HELMHOLTZ CENTER FOR
INFORMATION SECURITY

The future of security protocol analysis: Towards provable guarantees for apps?

Cas Cremers | Rennes | June 2024

Historical context

- **Computational Security reductions**:
  - Goldwasser, Micali, Yao, 1970s
  - Game hopping and others: Shoup, early 2000s
- **Symbolic analysis**
  - origins in Dolev-Yao, early 1980s
  - Automated tools mid 1990s (Maude-NPA, Casper/FDR)
  - Modern tools since 2000s (Tamarin, ProVerif)
- **We have been *proving* things secure for half a century!**

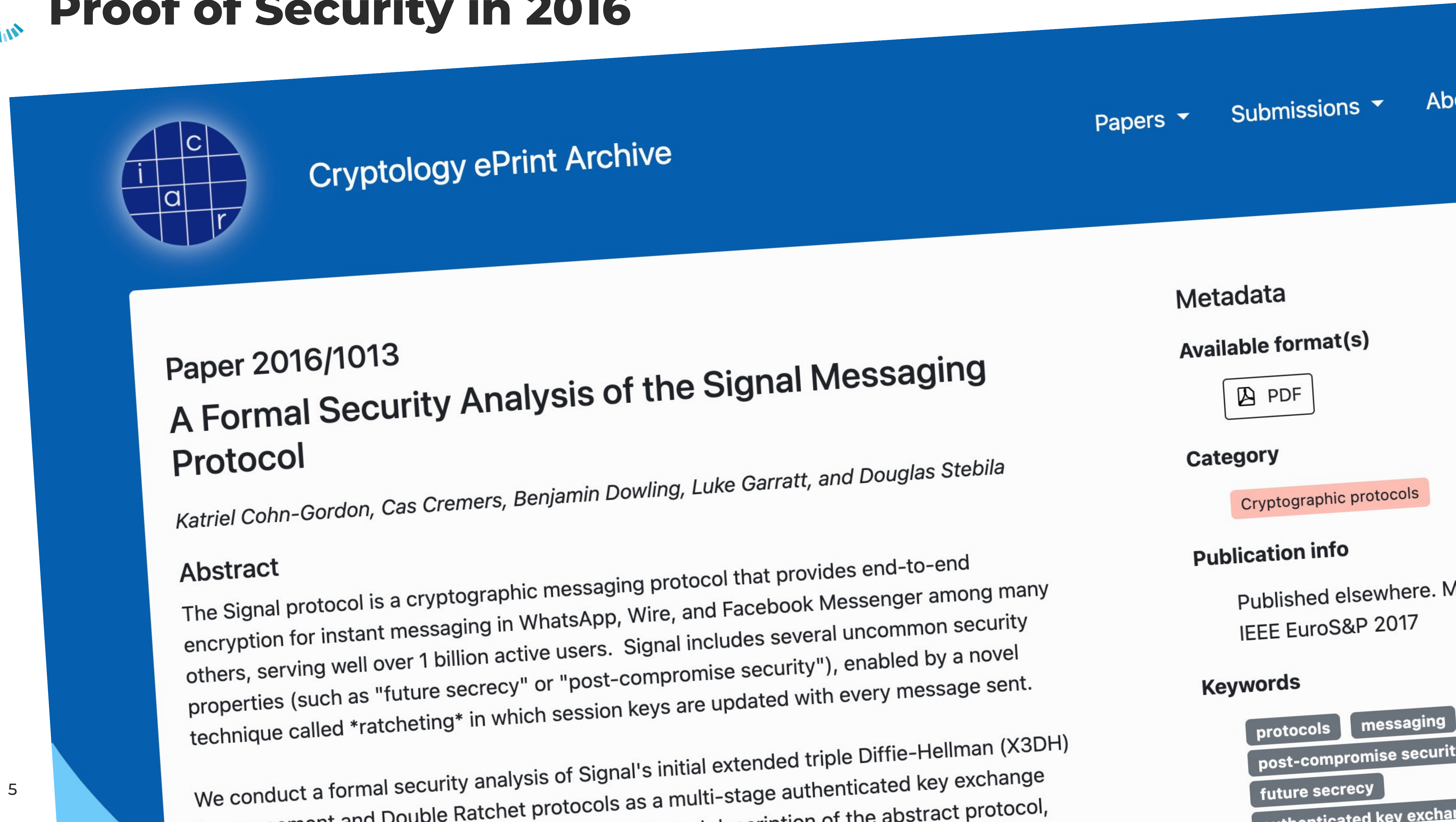
Many success stories

- Signal
- TLS 1.3
- Monero (Cryptocurrency transactions)
- SPDM 1.2
- EMV (Chip-and-Pin)
- IEEE 802.11 (WiFi)
- 5G-AKA
- Voting protocols

Case Study: Secure Messaging

- **Signal protocol** implemented in **libsignal** and integrated into many apps
  - WhatsApp
  - Signal (the App)
  - Facebook Messenger
  - Skype Private Conversations

Proof of Security in 2016



Cryptology ePrint Archive

Papers Submissions Ab

Paper 2016/1013

A Formal Security Analysis of the Signal Messaging Protocol

Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garratt, and Douglas Stebila

Abstract

The Signal protocol is a cryptographic messaging protocol that provides end-to-end encryption for instant messaging in WhatsApp, Wire, and Facebook Messenger among many others, serving well over 1 billion active users. Signal includes several uncommon security properties (such as "future secrecy" or "post-compromise security"), enabled by a novel technique called *ratcheting* in which session keys are updated with every message sent.

We conduct a formal security analysis of Signal's initial extended triple Diffie-Hellman (X3DH) ... and Double Ratchet protocols as a multi-stage authenticated key exchange ... of the abstract protocol,

Metadata

Available format(s)

PDF

Category

Cryptographic protocols

Publication info

Published elsewhere. M

IEEE EuroS&P 2017

Keywords

protocols messaging post-compromise securit future secrecy

The End

"Limitations" in IACR eprint 2016/1013

6. Limitations

As a first analysis of a complex protocol, we have chosen (some) simplicity over a full analysis of all of … our presentation and model can serve as a starting point for future analyses. … tures included in Signal which we have explicitly chosen not to model and …

… library components. The open-source libraries contain various sections of … part of the Signal protocol. For example … variant of the Double Ratchet … by Pond and included in the referenc… itself. Likewise, there is sup… t for online key exchanges instea… not intended to be part of the Signal protocol, we do no…

Out-of-band key verification. To reduce the trust requireme… mechanism for verifying public keys through an out-of-band … and medium-term public key distribution is honest, and do not analyse the … channel.

Same key for Ed25519 signing and Curve25519 DH. Signal uses the same key *ik* for DH agreement and for signing the medium-term prekeys[9]. [28, 60] prove security of a similar scheme under the Gap-DH assumption, effectively showing that the signatures can be simulated using the hashing random oracle. We conjecture a similar ar… not prove it; instead, we omit the signatures from consideration and enf… This enforced authentication means we do not capture the … ity key and then inserts a malicious signed pre-key.

… sages, users must store message keys until the messages a… d in Section 5 we do not consider this storage.

Simulta… a mechanism to deal silently with the case that Alice and Bob simultaneously initiate a session with each other. Roughly, when an agent detects that this has happened they deterministically choose one party as the initiator (e.g. by sorting identity public keys and choosing the smaller), and then complete the session as if the other party had n… amount of trial and error: agents maintain multiple states for each peer, … s in all of them. We do not consider this mechanism.

… escribes key indistinguishability of two-party multi-stage key … ionality goals which Signal may address but which we do no… rties[10], message sharing across multiple devices, voice and video call security, … (e.g. 0-round-trip modes), privacy, and deniability.

Implementation-specific threats. We make various assumptions on the components used by the protocol. In particular, we do not consider specific implementations of primitives (e.g. the particular choice of curve), instead assuming standard security properties. We also do not consider side-channel attacks.

Tightness of the security reduction. As pointed out in [2], a limitation … proofs for AKE protocols is that they do not provide tight reductions … depend on guessing the specific party and session under attac… huge amounts of sessions, such as Signal, this leads to an … some new AKE protocols with tight reductions, their proto… In particular, there is currently no known technique for constructi… Signal protocol. As a result, our analysis has a significantly large factor … security reduction. Specifically, we lose *at minimum* a factor of $(n_P{}^2 \cdot n_S)$ to our reduction to the Gap Diffie-Hellman assumption. Unfortunately, attempting to instantiate the Signal protocol with parameter sizes that our analysis suggests would be secure would *not* be compatible with parameter sizes that Signal implementations currently use. Indeed, implementing Signal with such parameter sizes would incur significantly more computational cost, a difficult proposition for mobile devices. One might argue about the practical relevance of such an analysis, however the current proof does provide … about the security of the Signal protocol, as well as a qualitative indicator of … over, the structure of the proof itself may be useful in future res… f the security reduction.

Application Variants. Popular applicati… Signal tend to change important details as they implement or integrate the protocol, and thus merit security analyses in their own right. For example, WhatsApp implements a re-transmission mechanism: if Bob appears to change his identity key, clients will resend messages encrypted under the new value. Hence, an adversary with control over identity registration can disconnect Bob and replace his key, and Alice will re-send the message to the adversary.

Parts of libsignal not used by Signal-the-app

Implementation threats

Using the same Curve 25519 key for signing and DH

Does tightness matter?

Out-of-order delivery

Different ways of using libsignal parts

Session handling

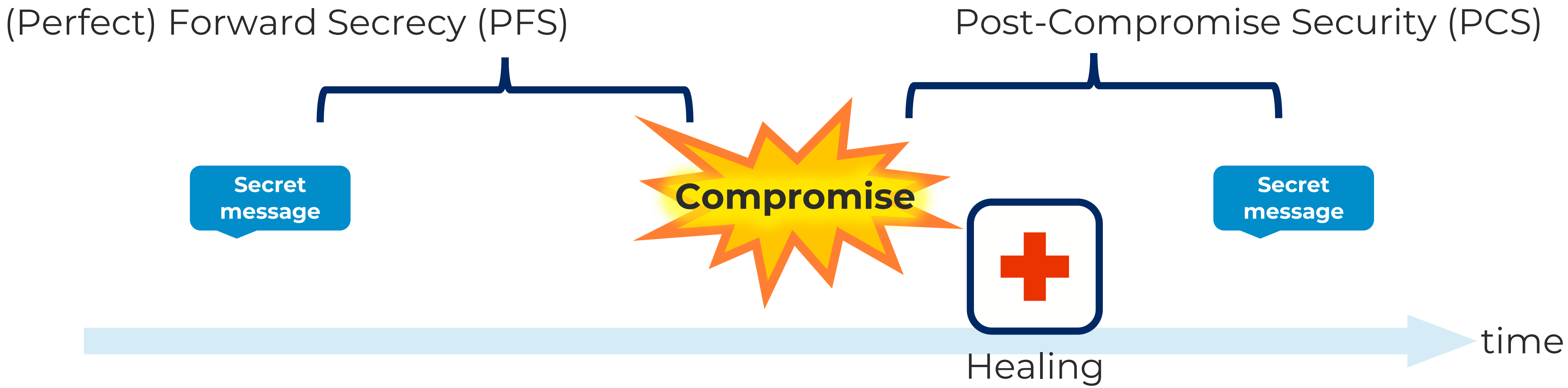
"A Formal Security Analysis of the Signal Messaging Protocol", Cohn-Gordon, Cremers, Dowling, Garratt, Stebila; Euro S&P 2017 / IACR eprint 2016/1013

Signal more in-depth

- What is Signal trying to achieve?

- “Secure Messaging”
  - Diffie-Hellman with a lot of keys and even more key rotation
  - Some form of deniability through implicit key exchange

Post-Compromise Security (PCS)



Attacker controls the network, and compromises a device at some point

“On Post-Compromise Security”; Cohn-Gordon, Cremers, Garratt; CSF 2016

Why is Post-Compromise Security (PCS) useful?

- **Older protocols do not ensure PFS**
  - Compromise allows adversary to decrypt any stored message, past or future

(E.g. TLS 1.2)

- **Newer protocols ensure PFS**
  - Compromise allows adversary to decrypt all future messages

(E.g. TLS 1.3)

- **Newest protocols also ensure PCS**
  - Compromise only allows adversary to decrypt until next *healing*
  - Thus, to maintain decryption, must interfere with all subsequent messages

(E.g. Signal)

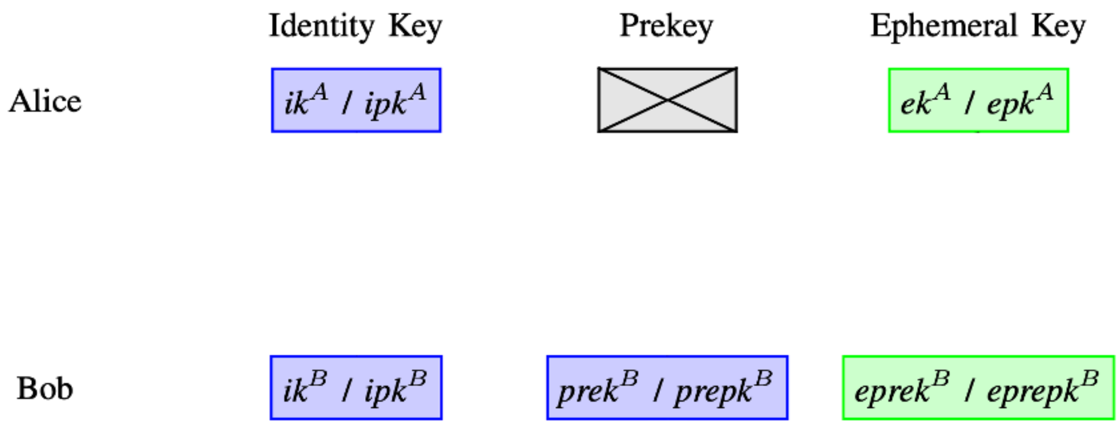
“On Post-Compromise Security”; Cohn-Gordon, Cremers, Garratt; CSF 2016

How Signal works (and achieves PCS)

- “**X3DH**”
  - Initial key exchange

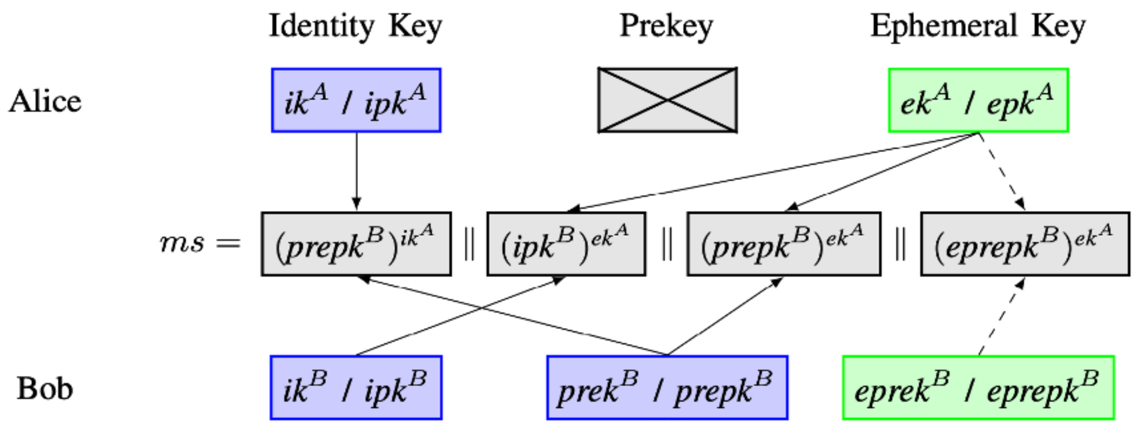
- “**Double Ratchet**”
  - **Asymmetric Ratchet**:
    - New Diffie-Hellman with each ping-pong communication, and combine this with previous secret
  - **Symmetric Ratchet**:
    - Ensure message keys are independent even if Bob does not respond

Signal's message key derivation



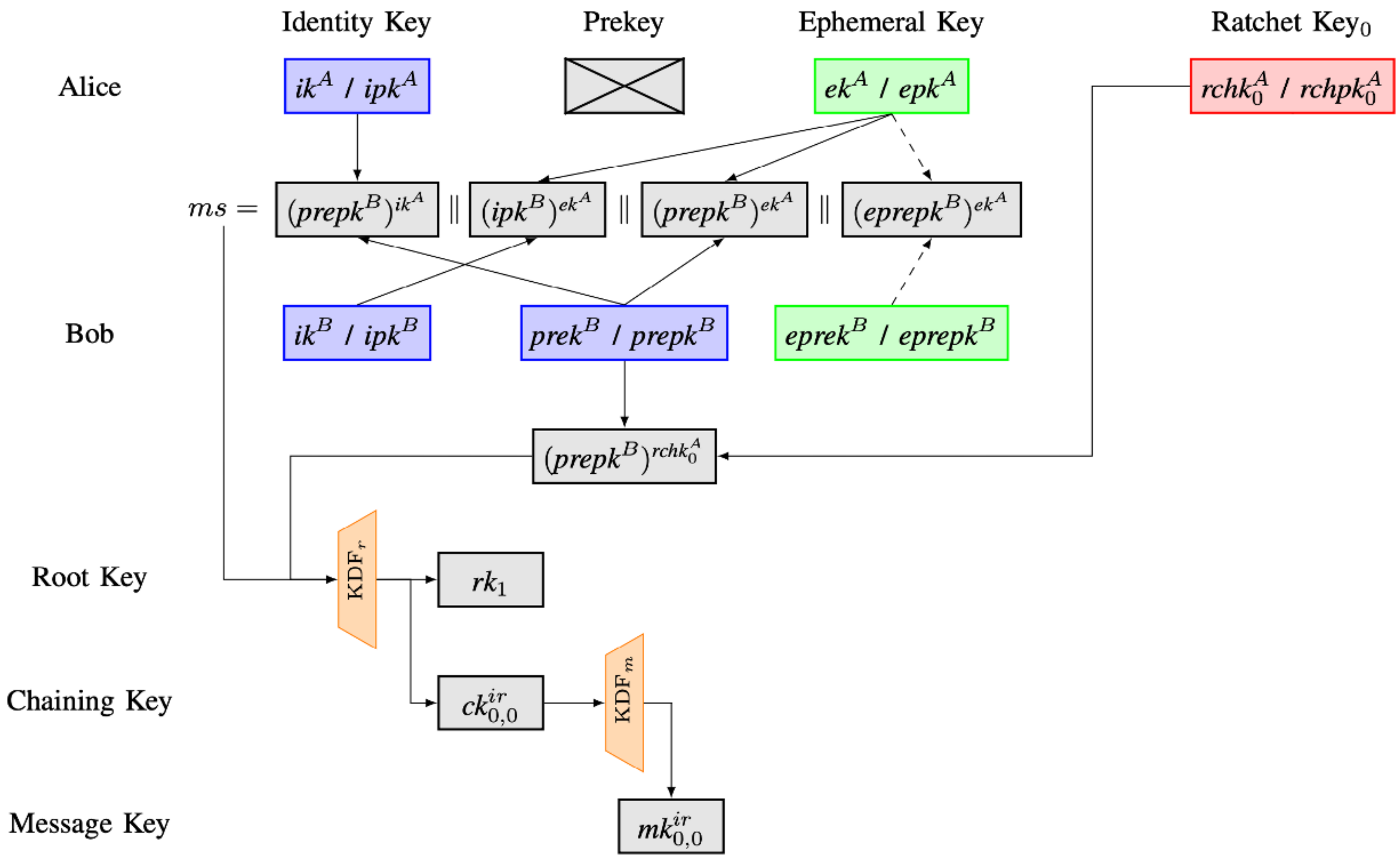
"A Formal Security Analysis of the Signal Messaging Protocol", Cohn-Gordon, Cremers, Dowling, Garratt, Stebila; Euro S&P 2017 / IACR eprint 2016/1013

Signal's message key derivation



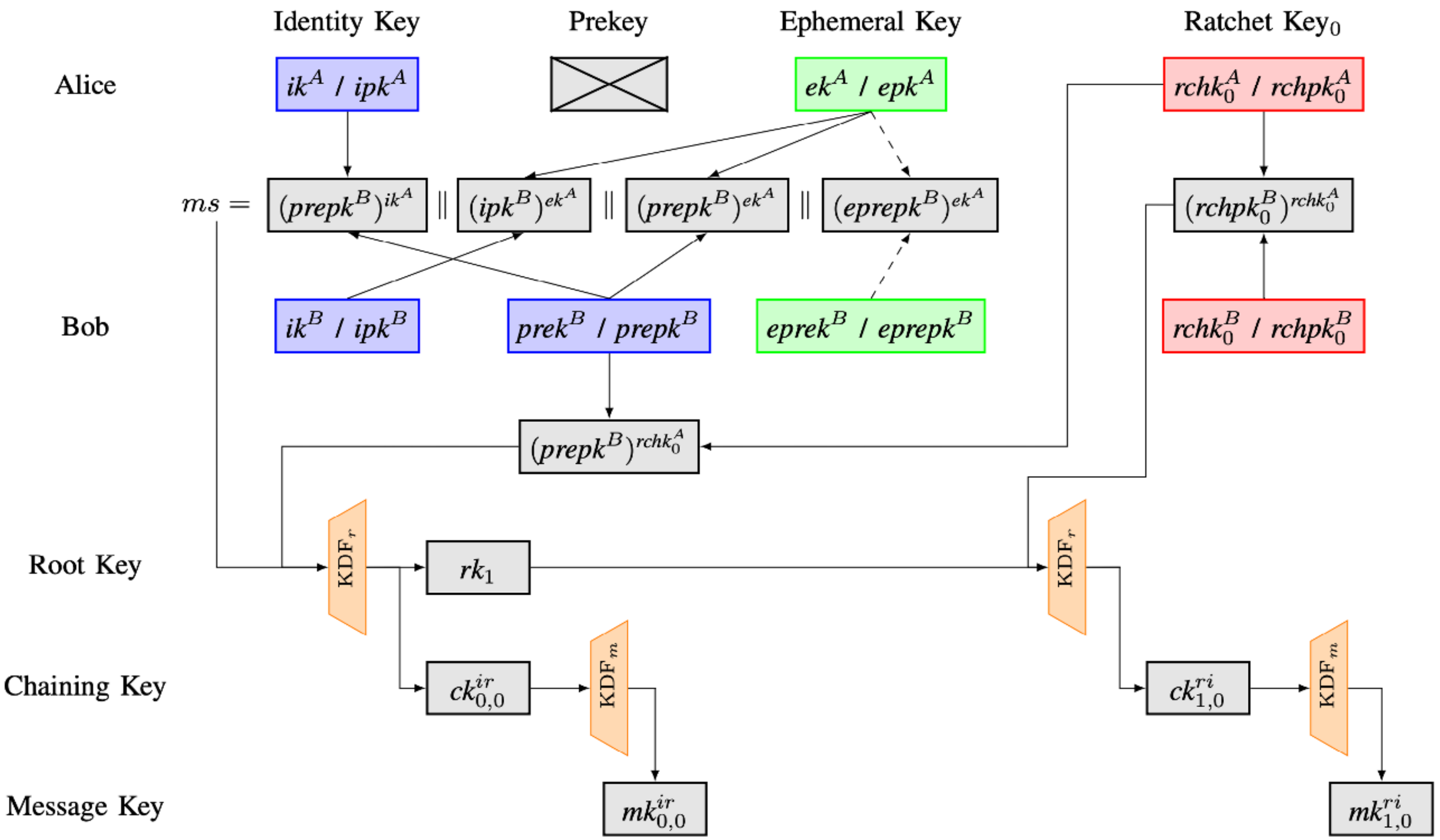
"A Formal Security Analysis of the Signal Messaging Protocol", Cohn-Gordon, Cremers, Dowling, Garratt, Stebila; Euro S&P 2017 / IACR eprint 2016/1013

Signal's message key derivation

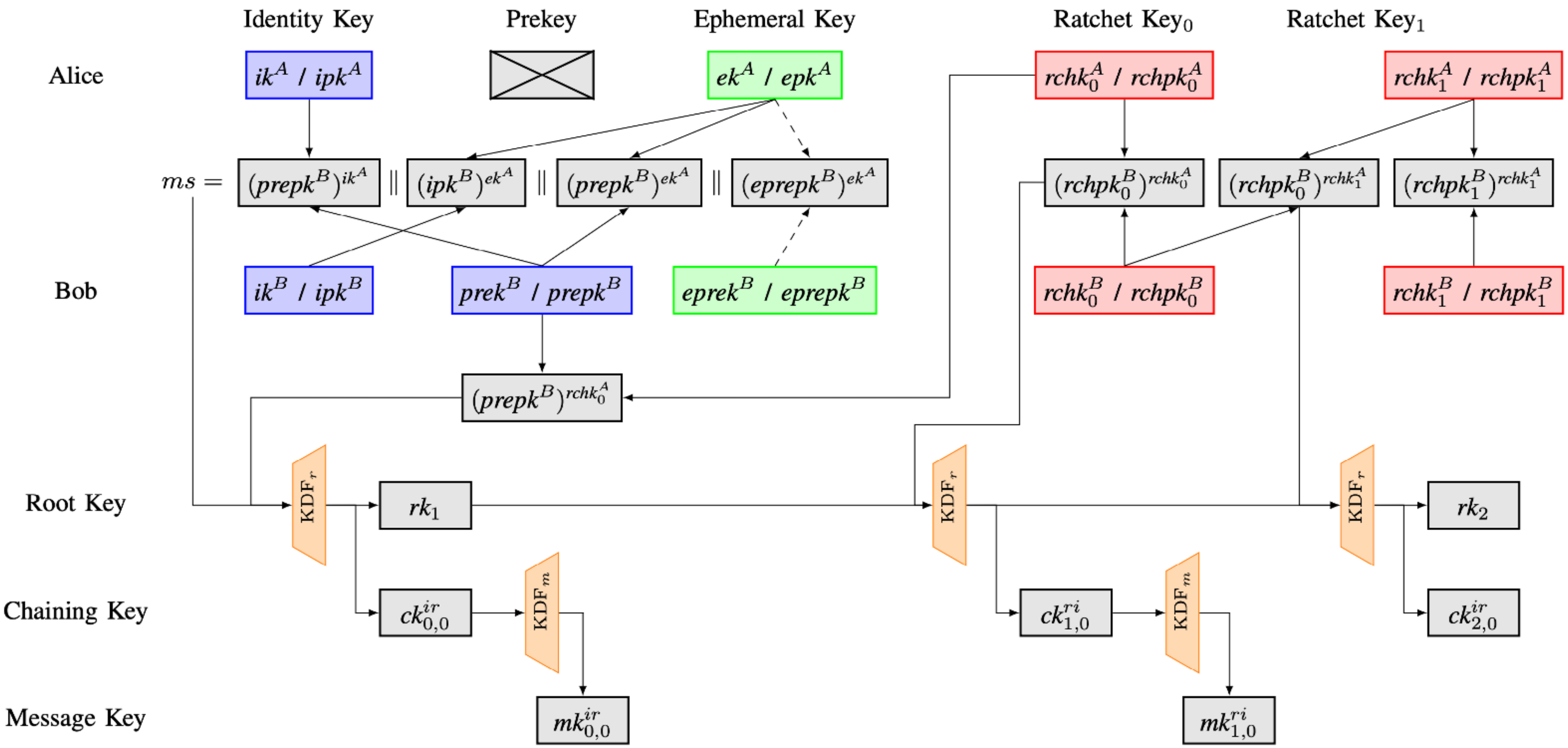


"A Formal Security Analysis of the Signal Messaging Protocol", Cohn-Gordon, Cremers, Dowling, Garratt, Stebila; Euro S&P 2017 / IACR eprint 2016/1013

Signal's message key derivation

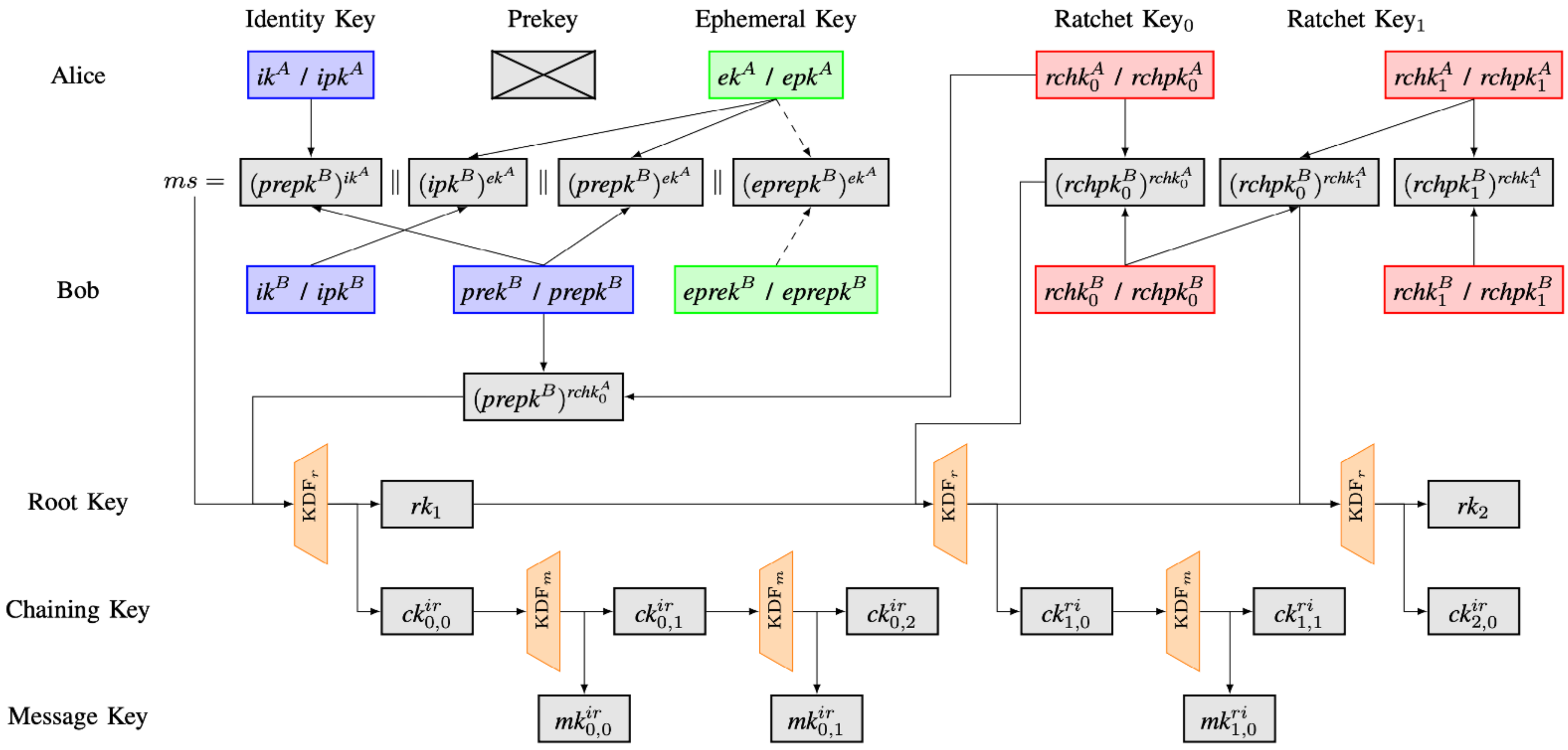


Signal's message key derivation



"A Formal Security Analysis of the Signal Messaging Protocol", Cohn-Gordon, Cremers, Dowling, Garratt, Stebila; Euro S&P 2017 / IACR eprint 2016/1013

Signal's message key derivation



"A Formal Security Analysis of the Signal Messaging Protocol", Cohn-Gordon, Cremers, Dowling, Garratt, Stebila; Euro S&P 2017 / IACR eprint 2016/1013

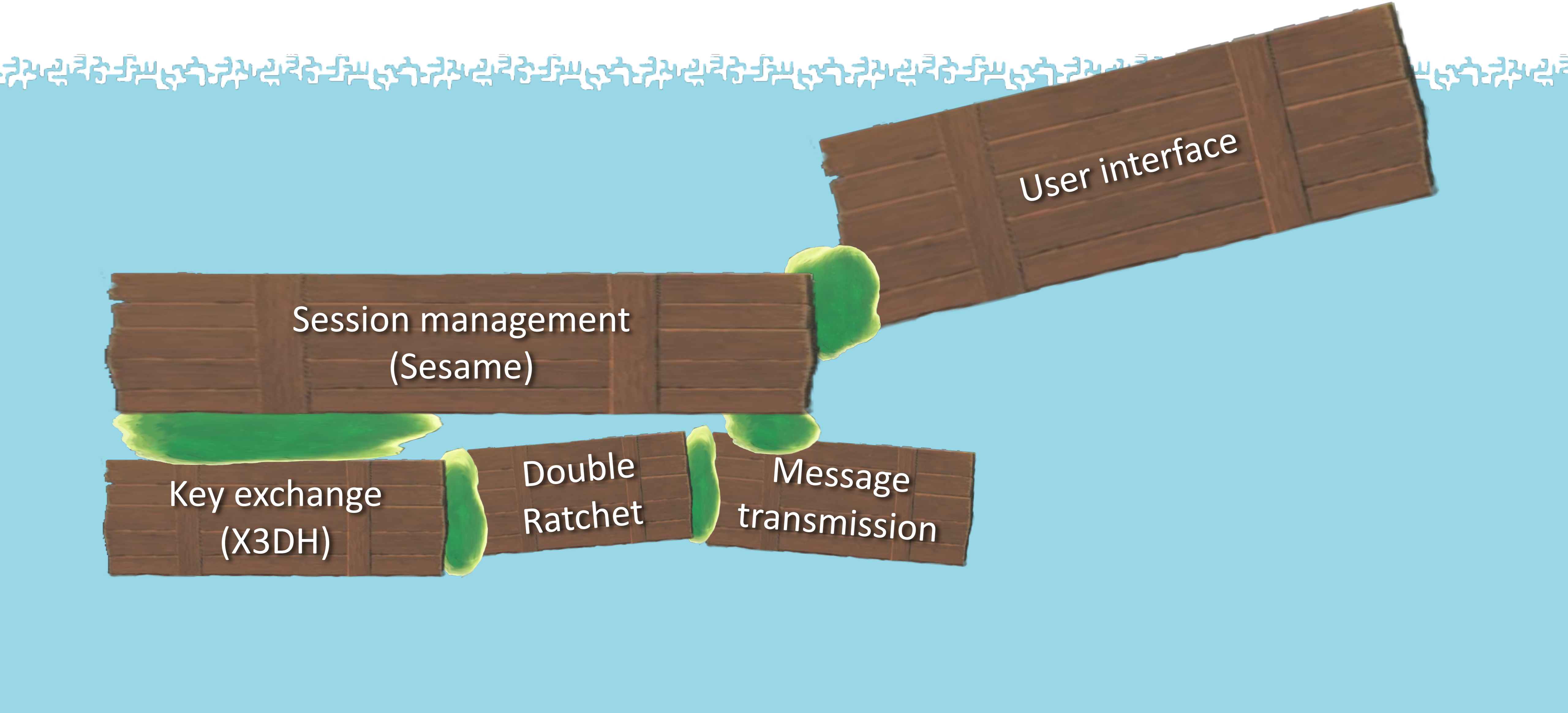
State-of-the-art

- **2016**: ”A Formal Security Analysis of the Signal Messaging Protocol” (IACR eprint 2016/1013)
- **2018**: “The Double Ratchet: Security Notions, Proofs, and Modularization for the Signal Protocol“ (2018/1037)
- **2019**: “A Unified and Composable Take on Ratcheting” (2019/694)
- **2019**: “Multi-Device for Signal” (2019/1363)
- **2022**: "A more complete analysis of the Signal Double Ratchet” (2022/355)
- **2022**: “Universally Composable End-to-End Secure Messaging” (2022/376)

- +lots of works on faster healing variants and trade-offs, for example:
  - **2023**:“How fast do you heal? A taxonomy for post-compromise security in secure-channel establishment”, Blazy et al
- **2024**: PQXDH Post-quantum Signal initial key exchange

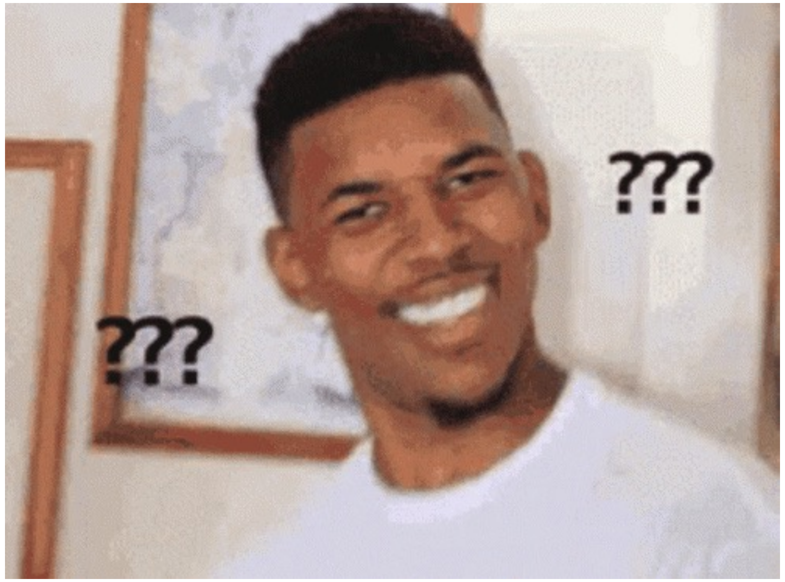
There is more!

There is more to Signal-the-App



What Signal is actually doing

~~One unbroken chain between Alice and Bob~~



Alice stores up to 40 sessions with Bob (per device)

One session is marked as the “active” session

User interface merges messages from all sessions and chains (invisible to user)

When Alice receives on a session, she sets it as the active one

When sending, Alice sends to active session

In case of a decryption error, Alice will start a new session

“Formal Analysis of Session-Handling in Secure Messaging: Lifting Security from Sessions to Conversations”; Cremers, Jacomme, Naska; USENIX 2023

Session handling: Why?

- The **SESAME protocol** (part of libsignal) deals with session handling

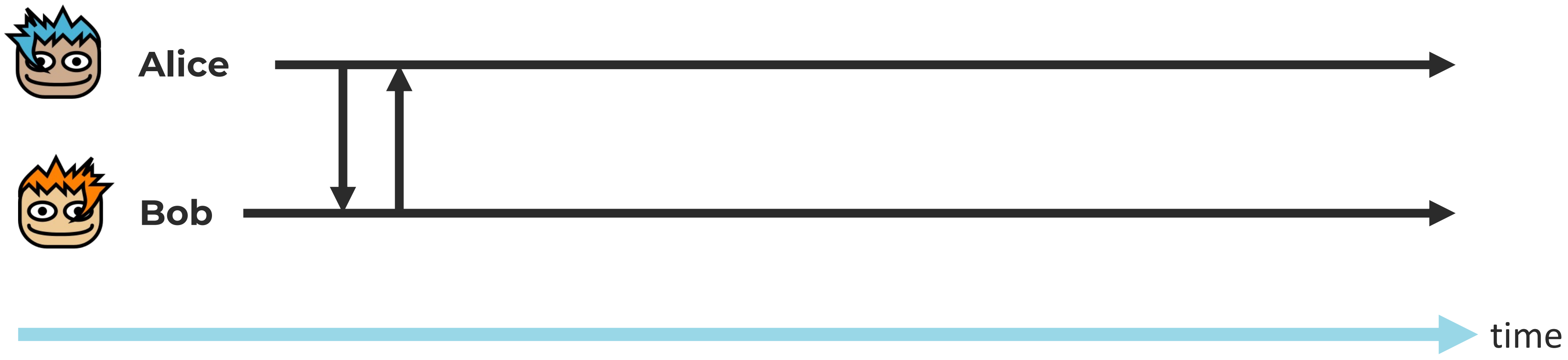
- Clients might *lose* chain state
  - A bit is flipped in memory
  - Restore a backup
  - Phone broken/lost, get a new one
  - ...
- Messages might be delayed in-flight

- If Bob loses his chain state, can he ever talk to Alice again?

“Formal Analysis of Session-Handling in Secure Messaging: Lifting Security from Sessions to Conversations”; Cremers, Jacomme, Naska; USENIX 2023

Impact on PCS

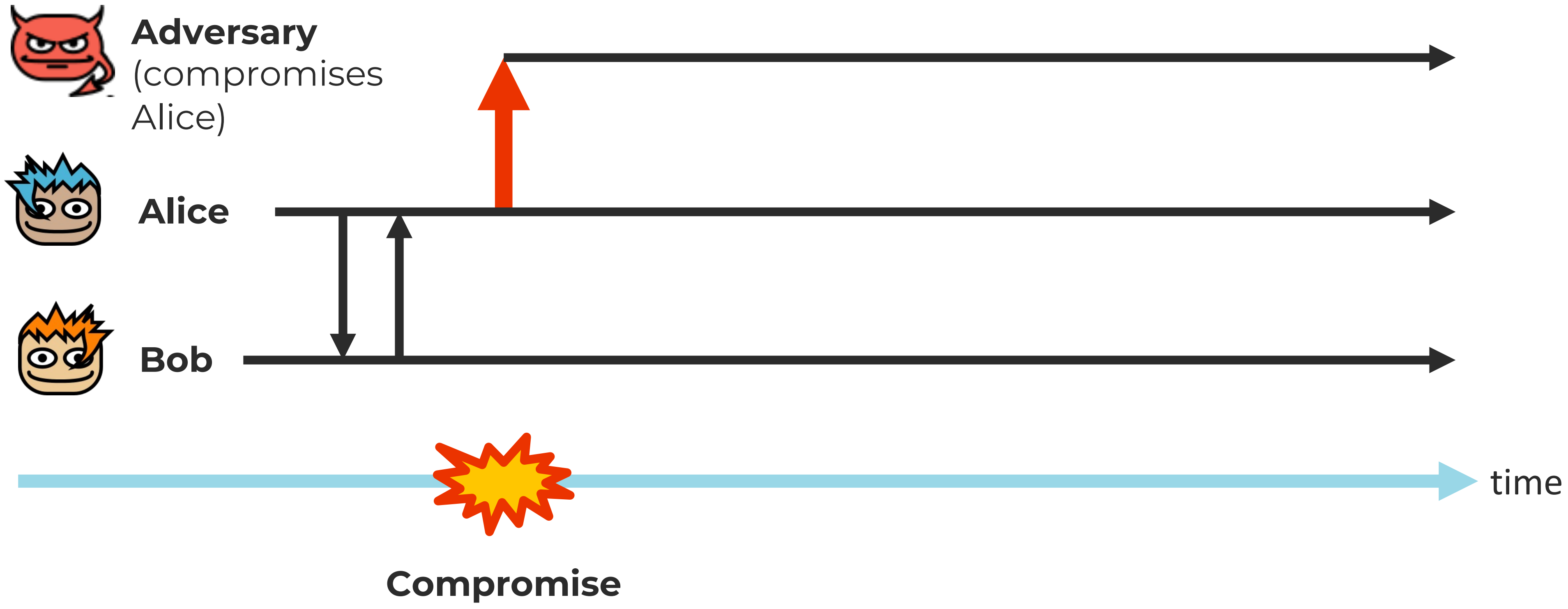
- Post-Compromise Security **is not achieved** for users in reality



“Formal Analysis of Session-Handling in Secure Messaging: Lifting Security from Sessions to Conversations”; Cremers, Jacomme, Naska; USENIX 2023

Impact on PCS

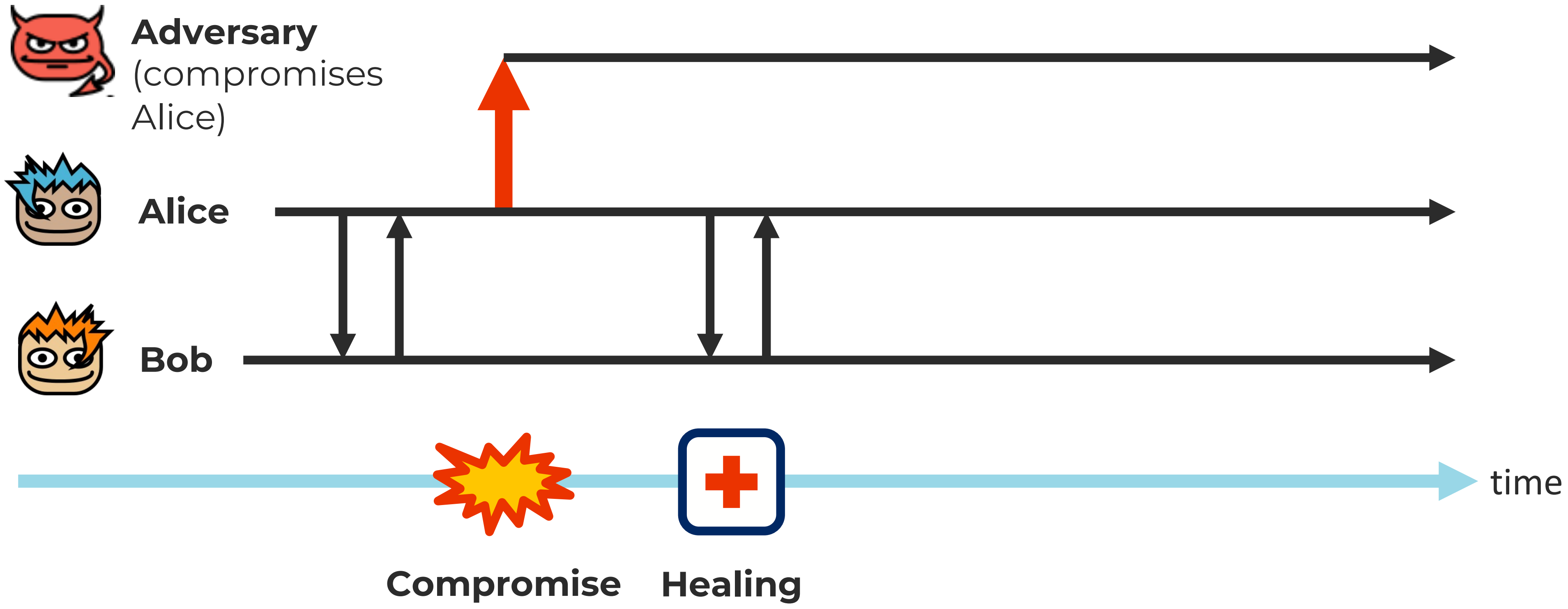
- Post-Compromise Security **is not achieved** for users in reality



"Formal Analysis of Session-Handling in Secure Messaging: Lifting Security from Sessions to Conversations"; Cremers, Jacomme, Naska; USENIX 2023

Impact on PCS

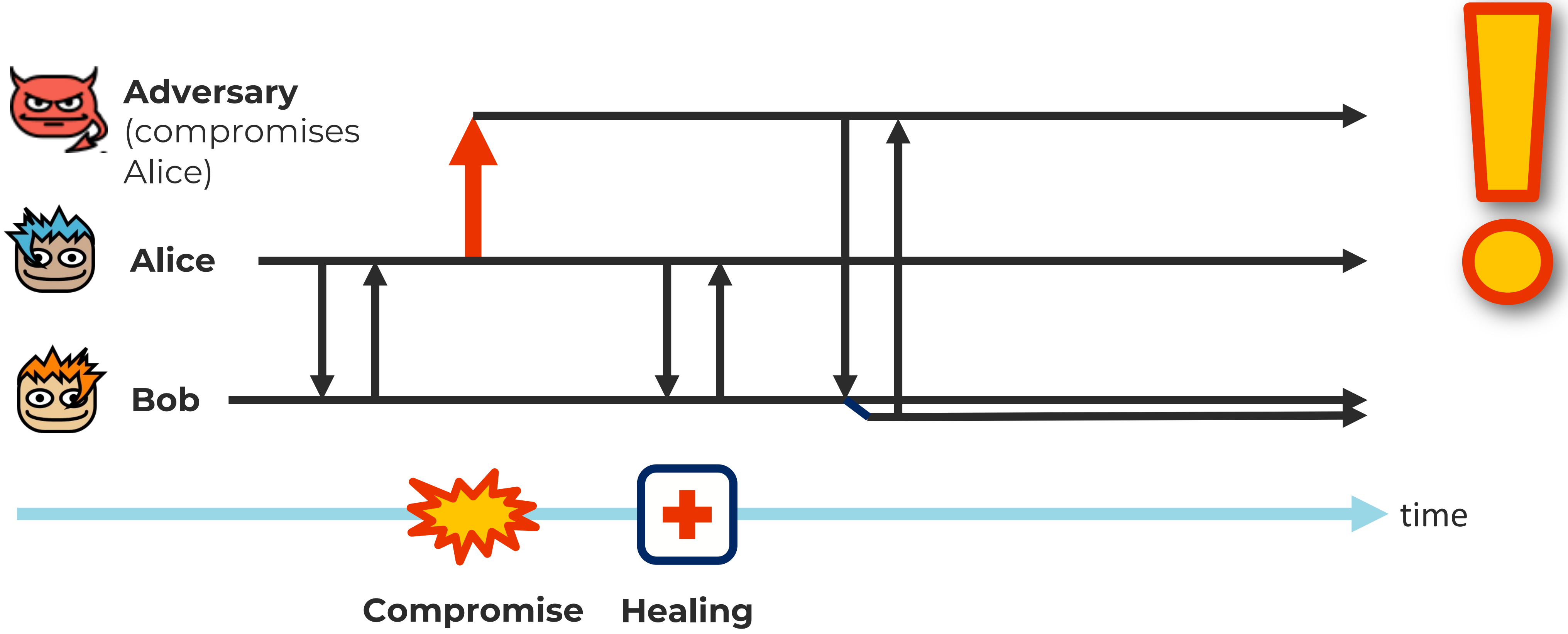
- Post-Compromise Security **is not achieved** for users in reality



"Formal Analysis of Session-Handling in Secure Messaging: Lifting Security from Sessions to Conversations"; Cremers, Jacomme, Naska; USENIX 2023

Impact on PCS

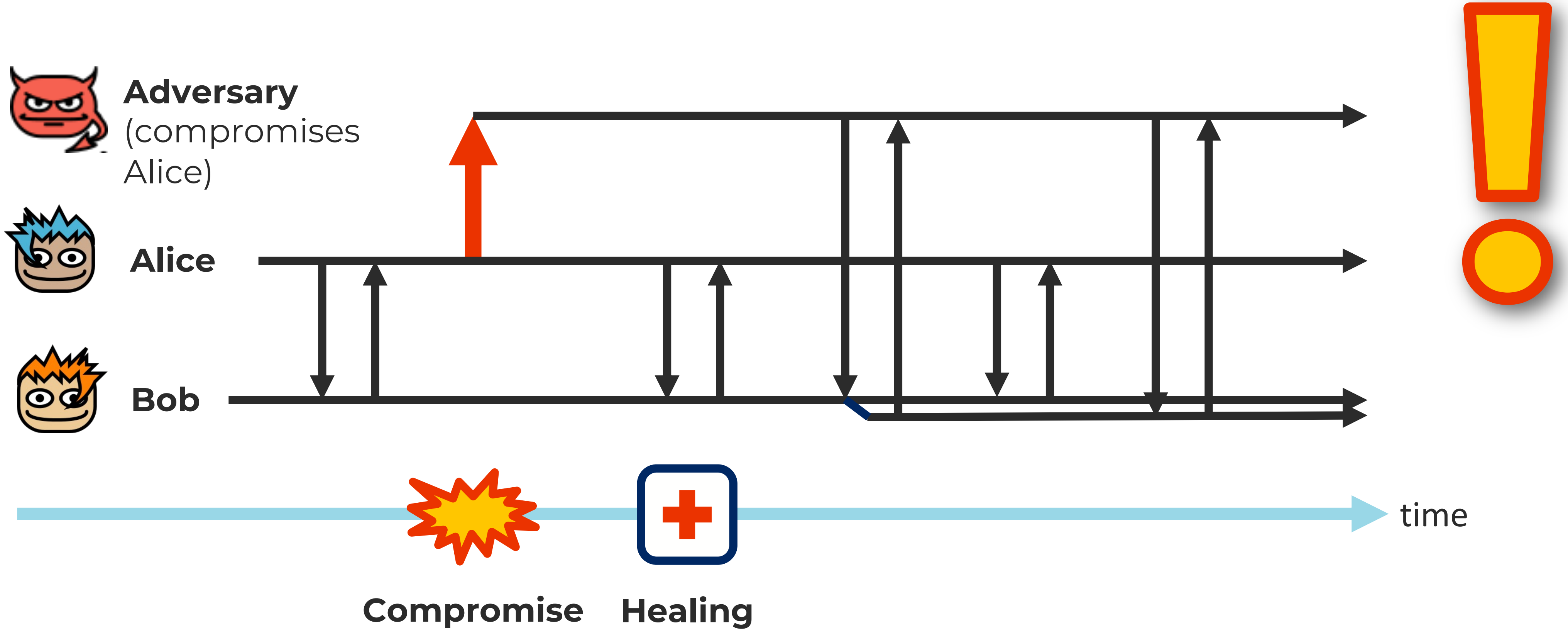
- Post-Compromise Security **is not achieved** for users in reality



"Formal Analysis of Session-Handling in Secure Messaging: Lifting Security from Sessions to Conversations"; Cremers, Jacomme, Naska; USENIX 2023

Impact on PCS

- Post-Compromise Security **is not achieved** for users in reality



“Formal Analysis of Session-Handling in Secure Messaging: Lifting Security from Sessions to Conversations”; Cremers, Jacomme, Naska; USENIX 2023

The Messaging Layer Security (MLS) Protocol
IETF RFC 9420

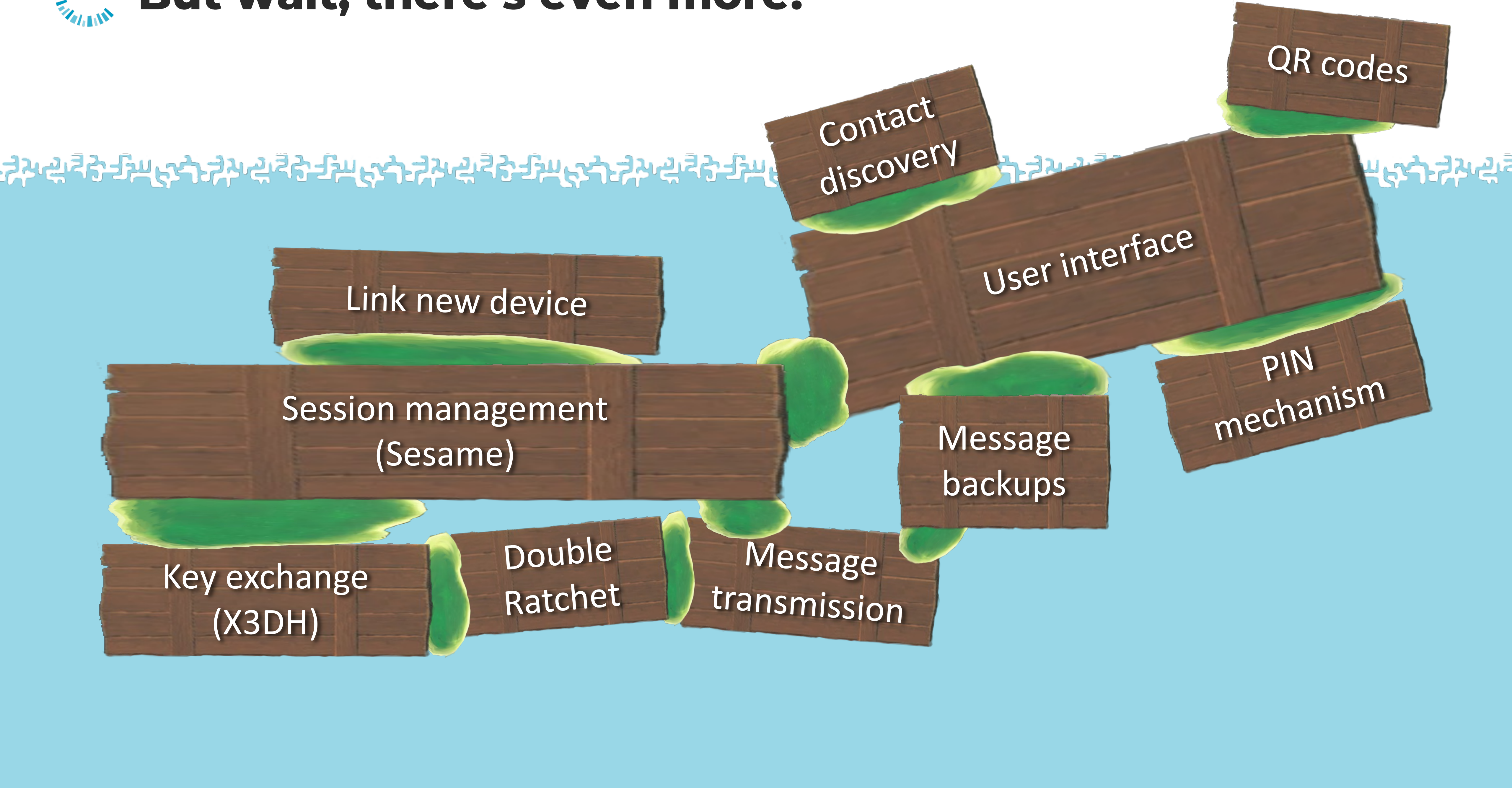
(Surely this is solved for the newest group messaging standard, MLS?)

“On Ends-to-Ends Encryption: Asynchronous Group Messaging with Strong Security Guarantees”, Cohn-Gordon , Cremers, Garratt, Millican, Milner; ACM CCS 2018

So why use a complex ratchet?

- We are currently writing up how to solve this and what cannot be achieved
  - Possibility & Impossibility results
- ETA: A few weeks (IACR eprint)

But wait, there's even more!



The wider picture

beyond Signal & beyond messaging

Structural problems

I. Proof methodologies scale badly

II. Composition results too limited

III. No consistency of threat models/properties across abstraction levels

Proof methodologies do not scale sufficiently

- Analyzing concurrent instances against network attacker remains a challenging problem

- Many brave attempts at large objects with various methodologies!

- Symbolic tools such as Tamarin can reach the largest scope for protocols
  - we expect growth, but not two orders-of-magnitude

Composition results too limited

If monolithic proofs don't scale, split into smaller parts? Literature has many!

- P(X) & P'(Y) => P''(Composition(X,Y))

- but often only consider a limited
  - Class of protocols,
  - Class of properties, or
  - Adversary model

- Examples:
  - Symbolic results often very limited adversary (eg lacking equational theories)
  - Real-world protocols share state, keying material, and primitives
  - Properties like PCS not covered

Universal Composability?

- The UC framework:
  1. Way to specify security properties (ideal functionality),
  2. Methodology to prove that a construction realizes a functionality, and
  3. Guaranteeing that realized functionalities safely compose with others

- In reality, we see it mostly used to specify & prove (1 & 2).

3. Inconsistency of guarantees and threat models

- Problem with multiple levels of abstraction:
  - We prove great properties at the lowest level…
  - Which we don’t need and don’t use at the next level

- Examples:
  - **Fine-grained TLS 1.3 security versus “secure channel”**
    - At level of a banking app:
      - cannot even formulate forward secrecy anymore
      - many subtly different channels in reality

3. Inconsistency of guarantees and threat models

- Problem with multiple levels of abstraction:
  - We prove great properties at the lowest level…
  - Which we don’t need and don’t use at the next level

- Examples:
  - **Resilience against state-reveal of protocol**
    - → app level state-reveal?
    - take messages from history?
    - Resending messages upon adding new device?

3. Inconsistency of guarantees and threat models

- Problem with multiple levels of abstraction:
  - We prove great properties at the lowest level…
  - Which we don't need and don't use at the next level

- Examples:
  - **Key indistinguishability of Key Exchange protocols**
    - → use with AEAD next?
    - Adversary can tell key from random at next level

Theory versus practice

- For messaging, super secure ratcheting is awesome!
  - But does it still make sense when
    - Users must still communicate after failures
    - State might become corrupted
    - Backend servers are not synchronized
    - ….?

- Should security researchers insist engineers use their building blocks as-is?
  - No, because researchers only see a fraction of the requirements

Picture: Billy Grace Ward (CC-BY)

Good news: Formal analysis becoming the norm

- Post-quantum versions of…
  - **Apple iMessage (PQ3)**
    - Manual game-hopping computational analysis
    - Tamarin analysis
  - **Signal (PQXDH)**
    - ProVerif
    - CryptoVerif
- But we need this for more types of applications, and more complete analyses

Conclusions!

- **Amazing progress in provable security**
- But we are **very far from done**
  - We prove properties of a fraction of small systems, and
  - These properties often do not hold at application level
- **Invitation**: ***let's work towards apps!***
- All of it will involve
  - More consistency across abstraction levels
  - Many more connection & composition results

Cas Cremers – cremers@cispa.de