An Efficient and Generic Construction for Signal's Handshake (X3DH): Post-Quantum, State Leakage Secure, and Deniable

Keitaro Hashimoto
Tokyo Tech/AIST, JP

Shuichi Katsumata
AIST, JP

Kris Kwiatkowski
PQShield, UK

Thomas Prest
PQShield, UK/FR

PKC 2021

Outline

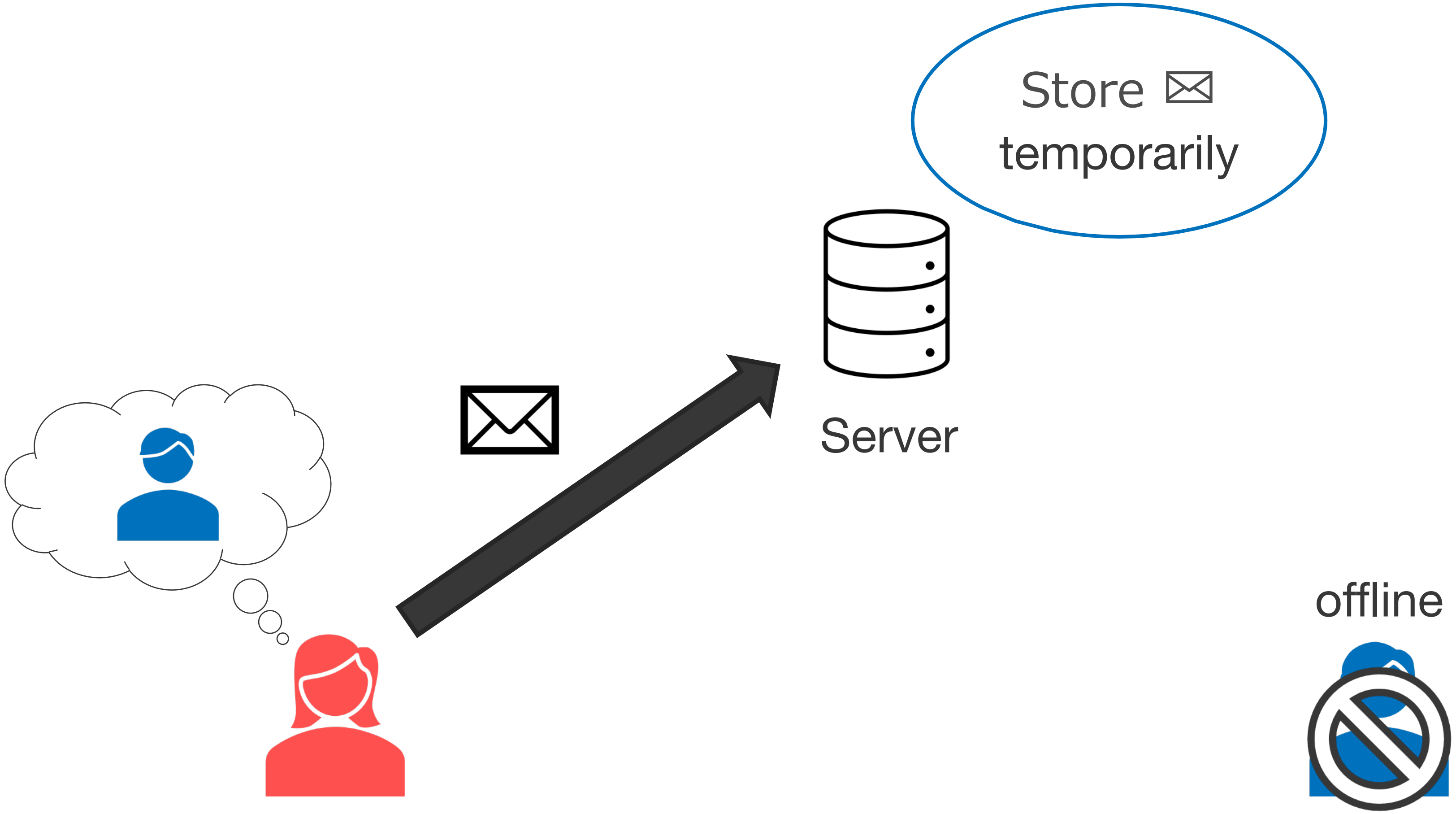
The first practical and post-quantum Signal protocol

1. Backgrounds: Instant Messaging and Signal
2. Formalization of Signal-conforming AKE (SC-AKE)
3. Generic construction of post-quantum SC-AKE
4. Implementation results

Background: Instant Messaging and Signal

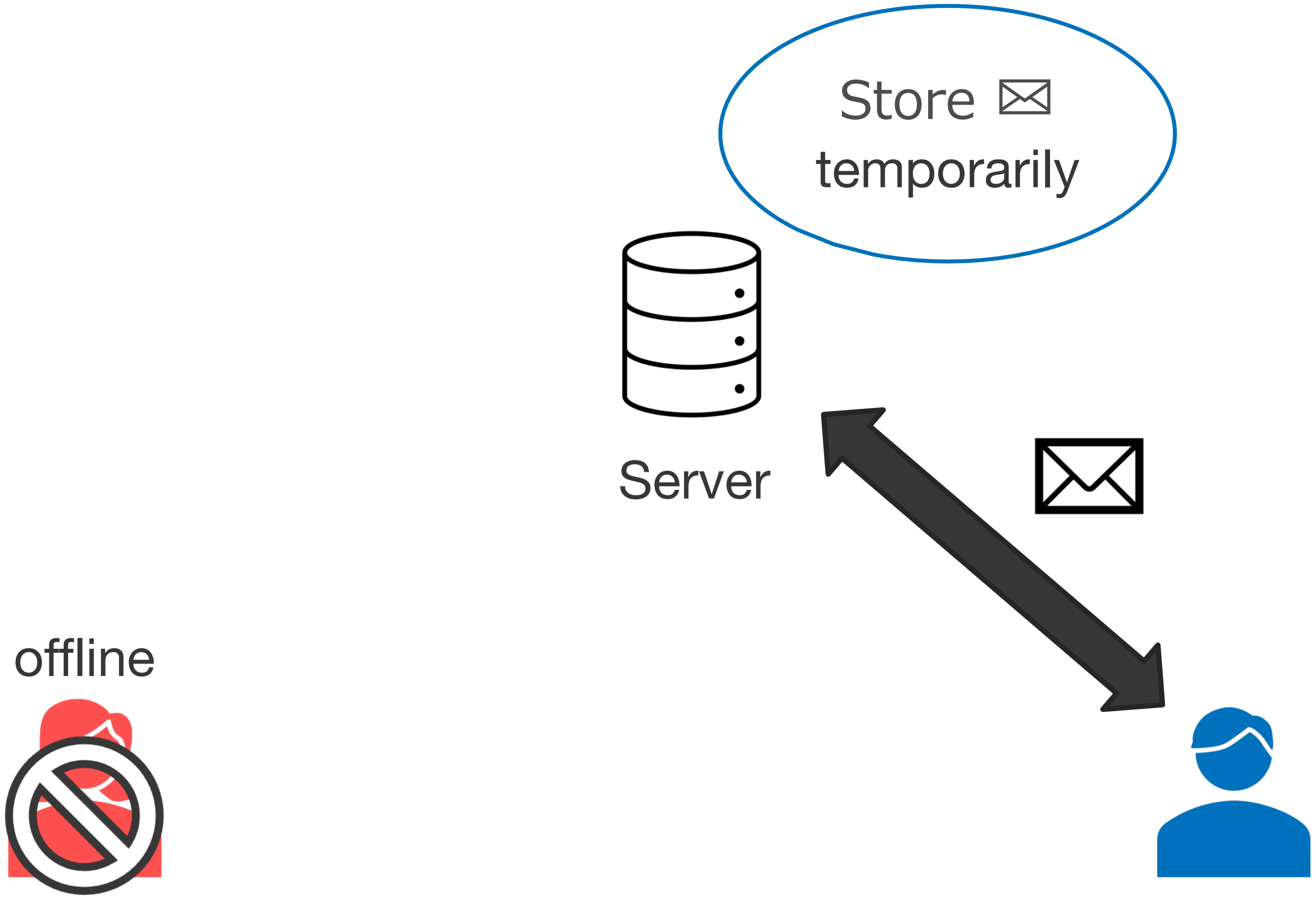
Instant Messaging

Communicate messages **asynchronously** through the server



Instant Messaging

Communicate messages **asynchronously** through server



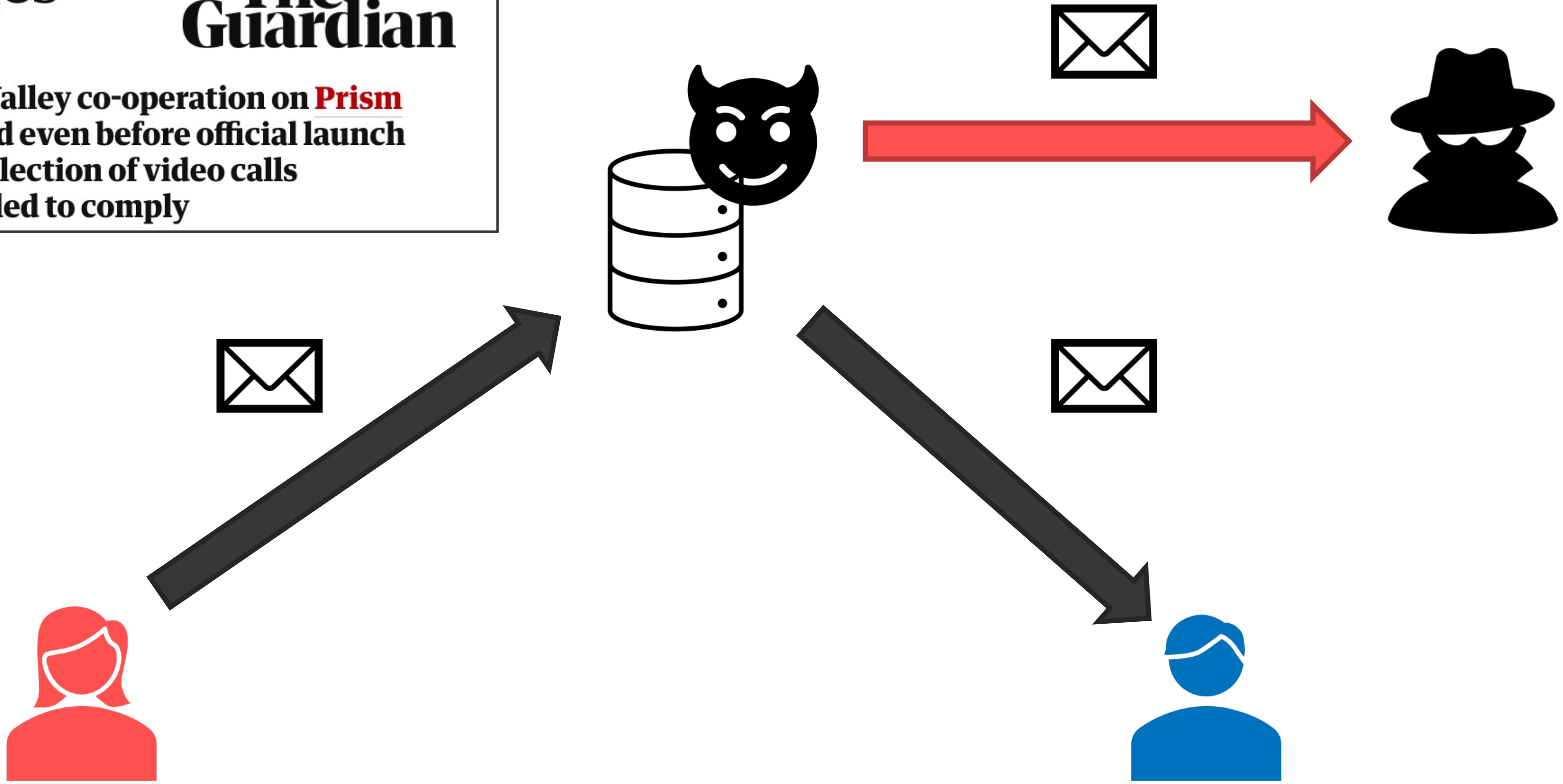
Secure Instant Messaging

- Malicious server may reveal messages
  - Ex. Sever helped an intelligence agency with collecting messages

Microsoft handed the NSA access to encrypted messages

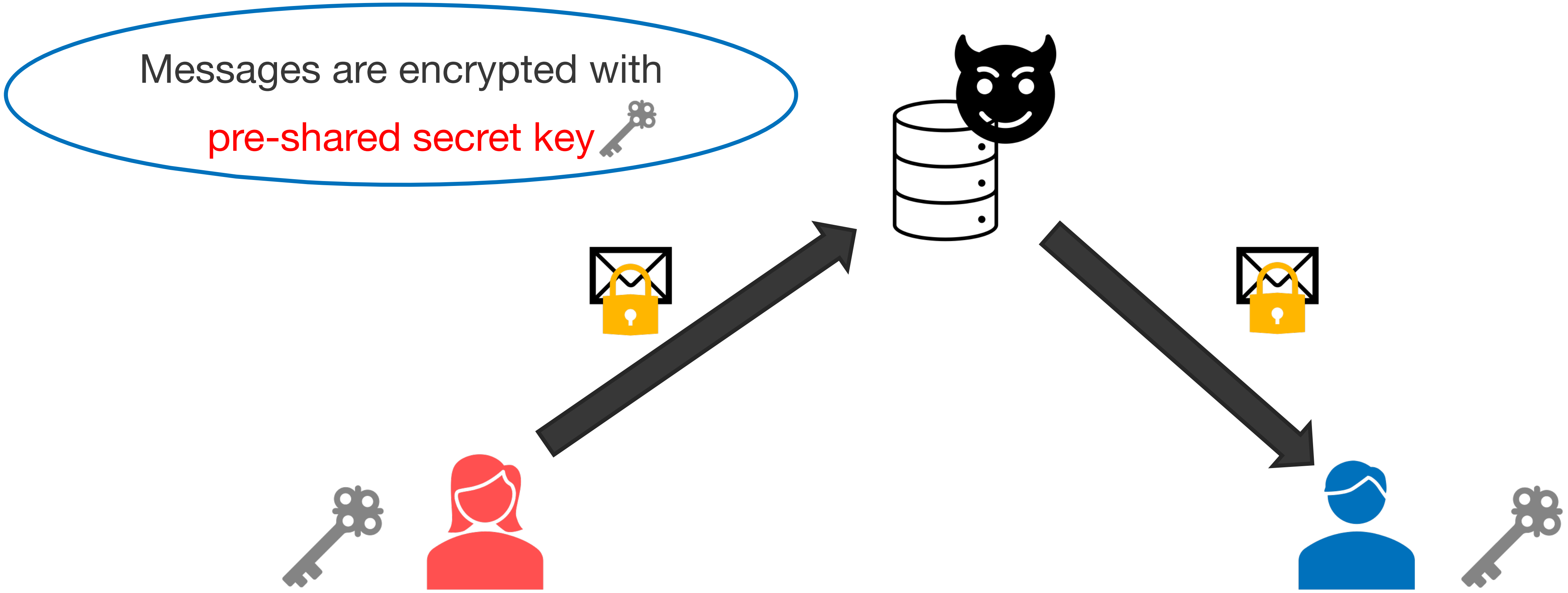
The Guardian

- **Secret files show scale of Silicon Valley co-operation on Prism**
- **Outlook.com encryption unlocked even before official launch**
- **Skype worked to enable Prism collection of video calls**
- **Company says it is legally compelled to comply**



Secure Instant Messaging

- Malicious server may reveal messages
  - Ex. Sever helped an intelligence agency with collecting messages
- To ensure security and privacy, secure instant messaging is widely used



Signal

- Widespread secure instant messaging application
- Use Signal protocol based on Diffie-Hellman assumption
- **Signal protocol** is deployed in Signal, WhatsApp, Facebook Messenger, etc.
    - **Billions of users** in the world

Source of photo:
- https://commons.wikimedia.org/wiki/File:Signal_ultramarine_icon.png
- https://commons.wikimedia.org/wiki/File:WhatsApp.svg
- https://commons.wikimedia.org/wiki/File:Facebook_Messenger_logo_2020.svg

Signal protocol

Signal protocol

X3DH

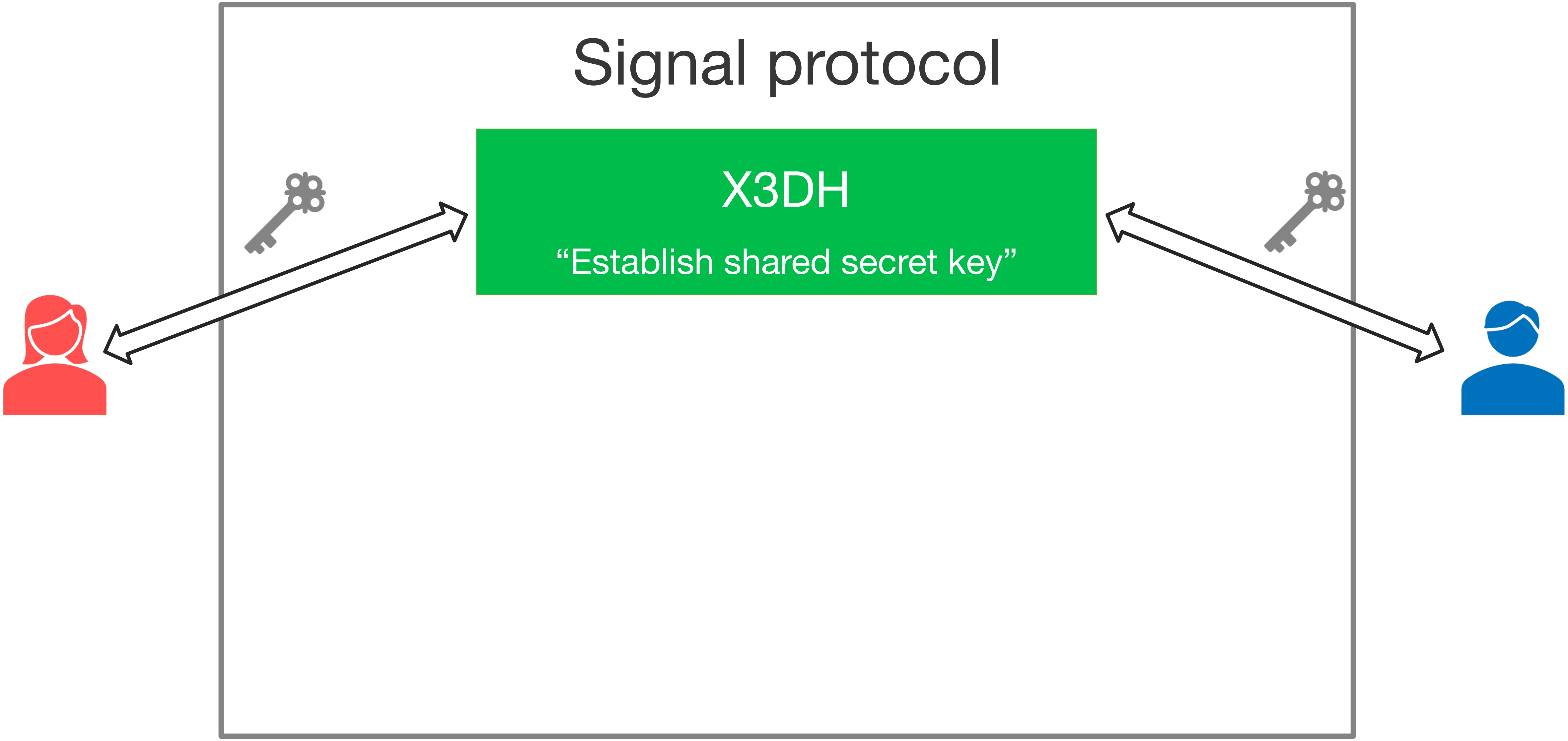
"Establish shared secret key"

↓

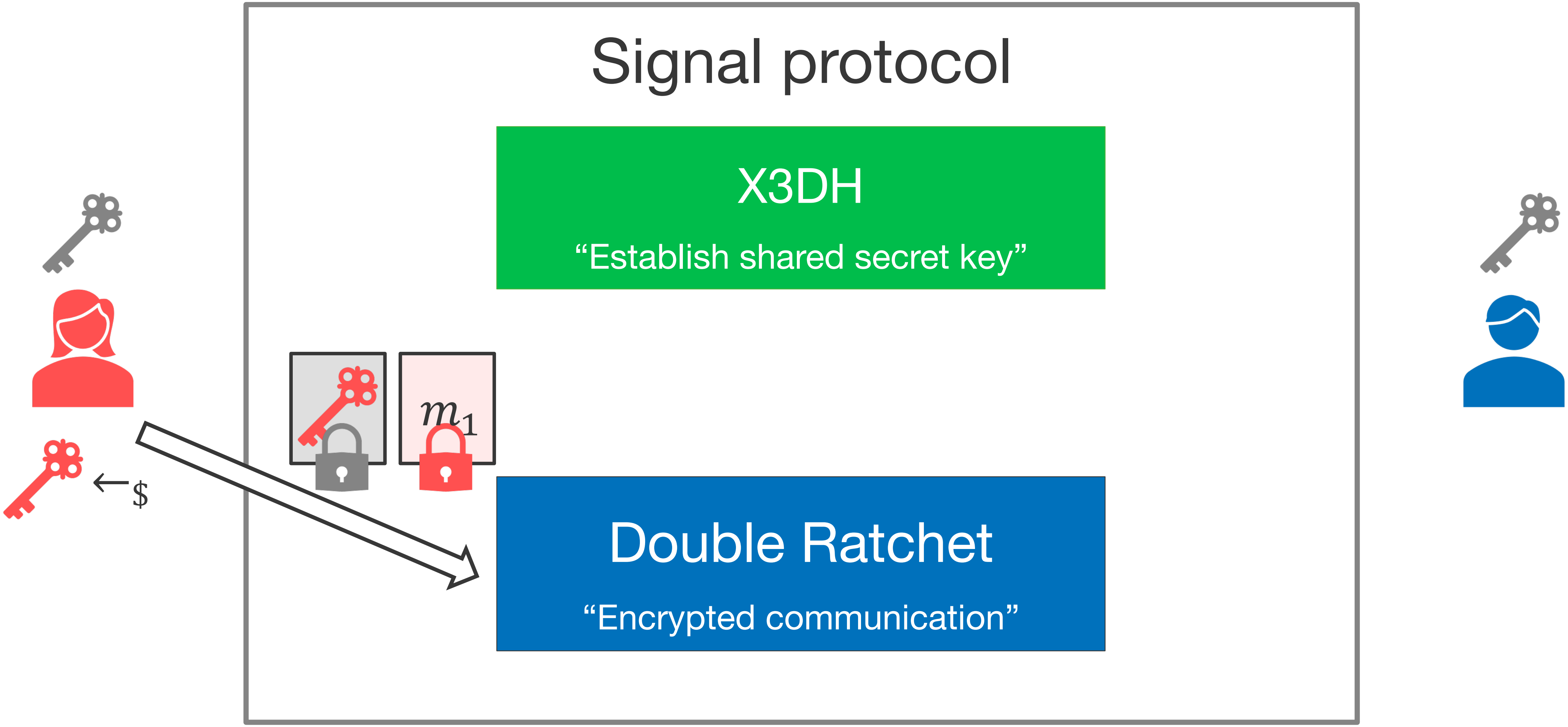
Double Ratchet

"Encrypted communication"

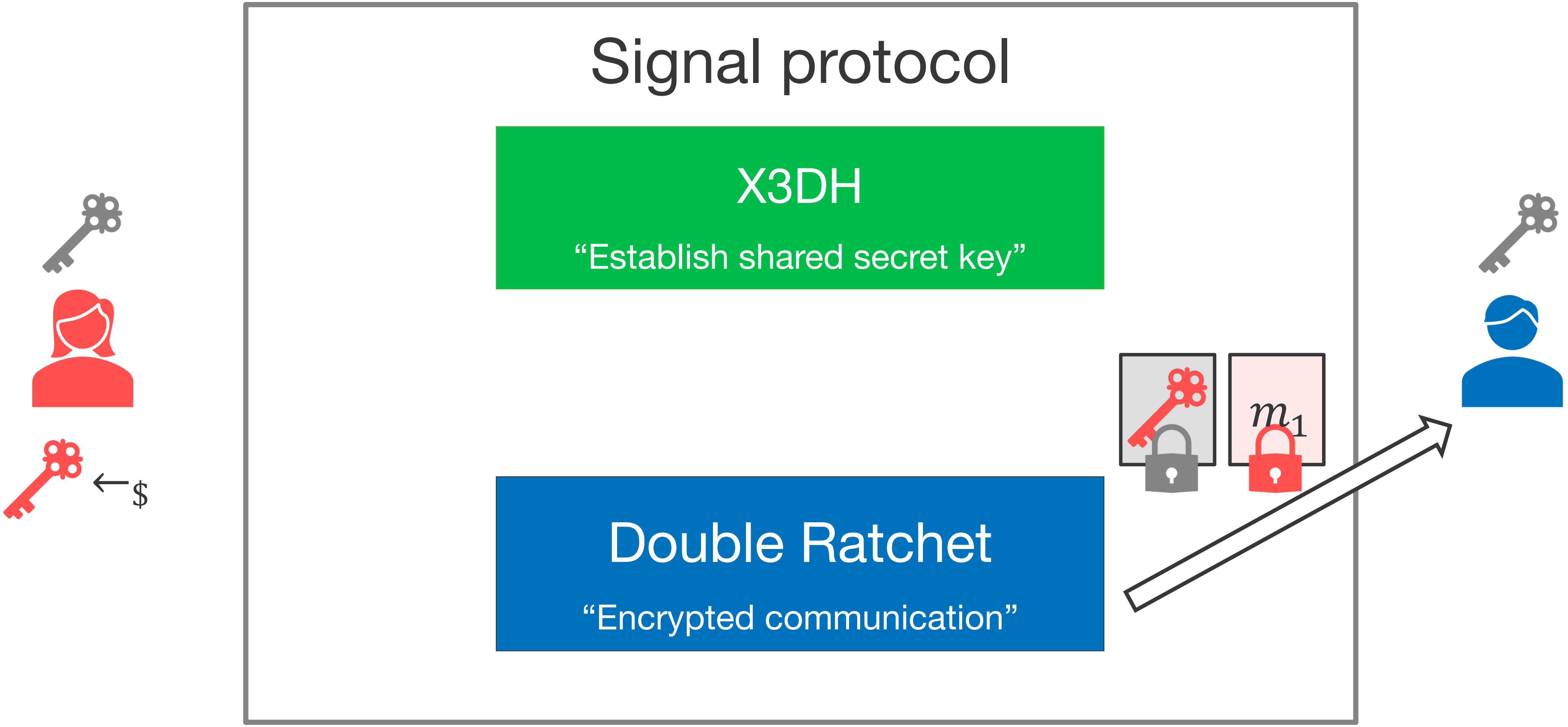
Signal protocol



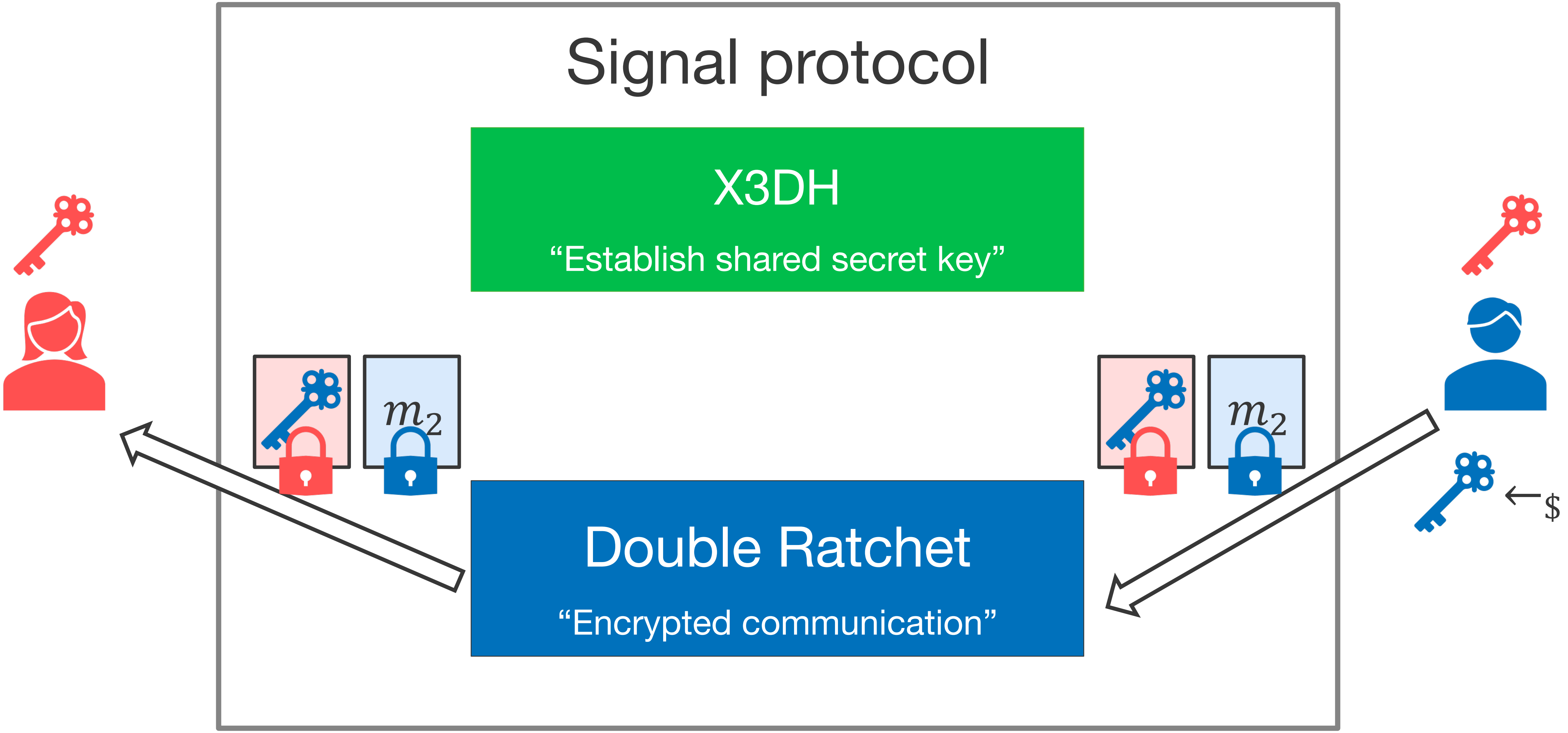
Signal protocol



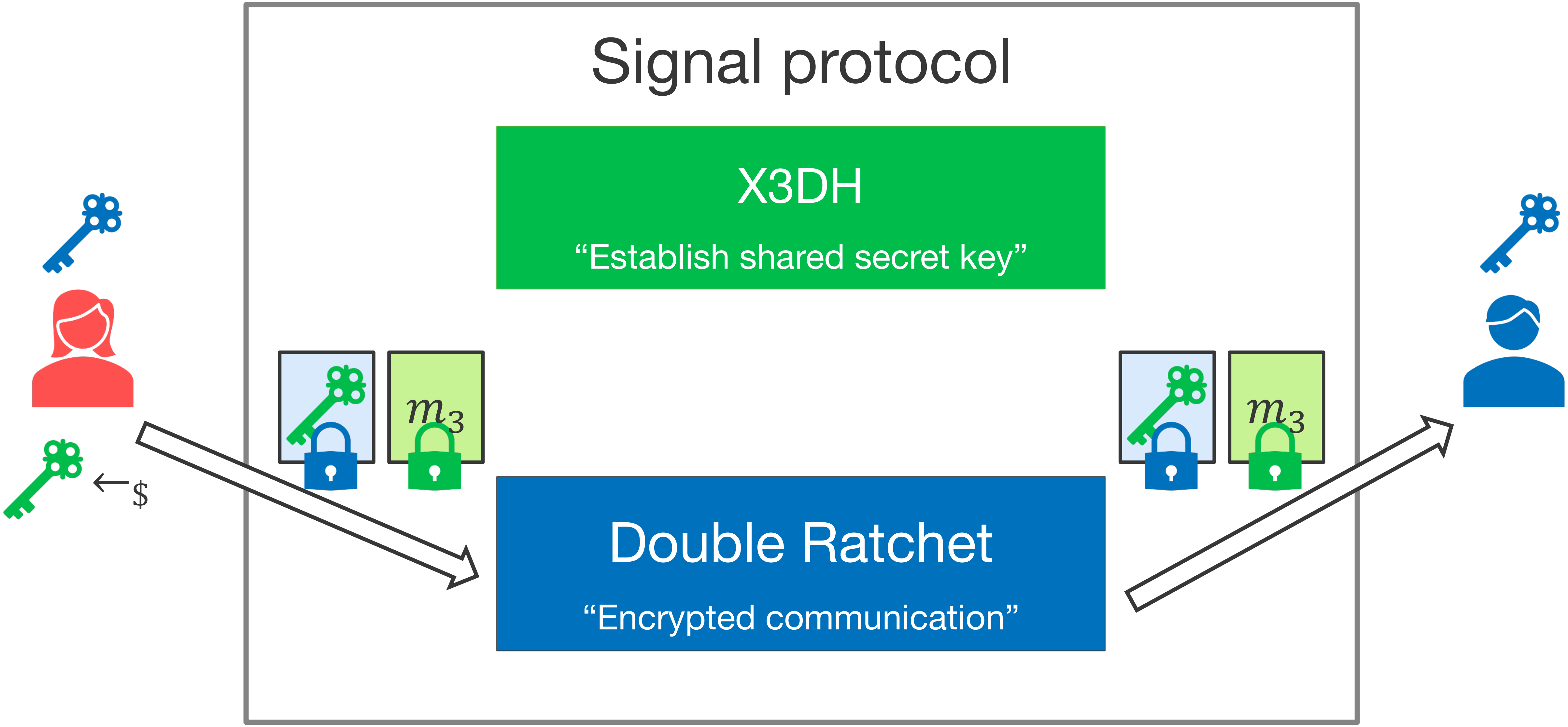
Signal protocol



Signal protocol



Signal protocol



Related works

2016 Double Ratchet protocol [MP16a]
X3DH protocol [MP16b]
were proposed in white paper

2017 Cohn-Gordon et al. [CGC+17] analyzed Signal protocol

2019 Alwen et al. [ACD19]
- formalized security models of Double Ratchet protocol
- proposed generic construction of DR protocol instantiable from post-quantum assumptions ☺

Related works

2016

Double Ratchet protocol [MP16a]
X3DH protocol [MP16b]

were proposed in white paper

2017

Cohn-Gordon et al. [CGC+17] analyzed Signal protocol

2019

Alwen et al. [ACD19]
- formalized security models of Double Ratchet
- proposed generic construction of DR protocol instantiable from post-quantum assumptions ☺

As for X3DH protocol:

- Security models has not been formalized
  (White paper [MP16b] provides overview of its security)
- Constructions from other than DH assumption are unknown ☹
  (Generic construction does not exist either)

Related works

2016 Double Ratchet protocol [MP16a]
X3DH protocol [MP16b]
were proposed in white paper

2017 Cohn-Gordon et al. [CGC+17] analyzed Signal protocol

2019 Alwen et al. [ACD19]
- formalized security models of Double Ratchet
- proposed generic construction of DR protocol instantiable from post-quantum assumptions ☺

Purpose

- Formalize security models of X3DH protocol
- Design generic construction of X3DH protocol

Our contribution

Design and Implementation of generic construction
as alternative to X3DH protocol

Theory

Practice

Our contribution

Design and Implementation of generic construction
as alternative to X3DH protocol

Theory

- Formalize X3DH protocol as a specific type of AKE
  - Call Signal-conforming AKE (SC-AKE)
- Define functionality and security for SC-AKE

Practice

Our contribution

Design and Implementation of generic construction as alternative to X3DH protocol

Theory

- Formalize X3DH protocol as a specific type of AKE
  - Call Signal-conforming AKE (SC-AKE)
- Define functionality and security for SC-AKE
- Propose generic construction of post-quantum SC-AKE based on KEM & SIG

Practice

Our contribution

Design and Implementation of generic construction
as alternative to X3DH protocol

Theory

- Formalize X3DH protocol as a specific type of AKE
  - Call Signal-conforming AKE (SC-AKE)
- Define functionality and security for SC-AKE
- Propose generic construction of post-quantum SC-AKE based on KEM & SIG

Practice

- Implement our SC-AKE using NIST PQC candidates
- Evaluate computation and communication costs

Our contribution

Design and Implementation of generic construction
as alternative to X3DH protocol

Theory

- Formalize X3DH protocol as a specific type of AKE
  - Call Signal-conforming AKE (SC-AKE)
- Define functionality and security for SC-AKE
- Propose generic construction of post-quantum SC-AKE based on KEM & SIG

Practice

- Implement our SC-AKE using NIST PQC candidates
- Evaluate computation and communication costs

Realize the first practical and post-quantum Signal protocol!

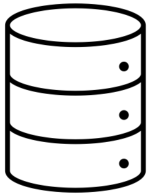
Contribution 1

Theory: Formalizing SC-AKE

Recap: X3DH protocol

<u>Asynchronous</u> key exchange protocol with the help of server

Initialization phase



1. Gen long-term key (g^a, a)
2. Gen first message g^x
3. Store x as state

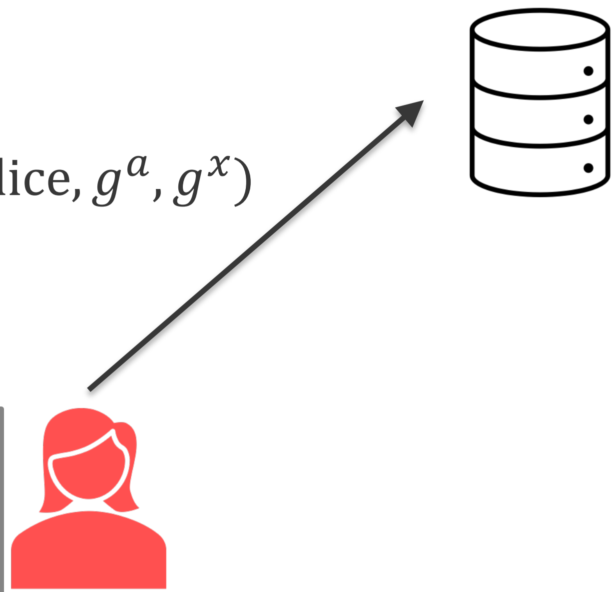
Key pair (g^a, a)
State x

Recap: X3DH protocol

<u>Asynchronous</u> key exchange protocol with the help of server

Initialization phase

(Alice, g^a, g^x)



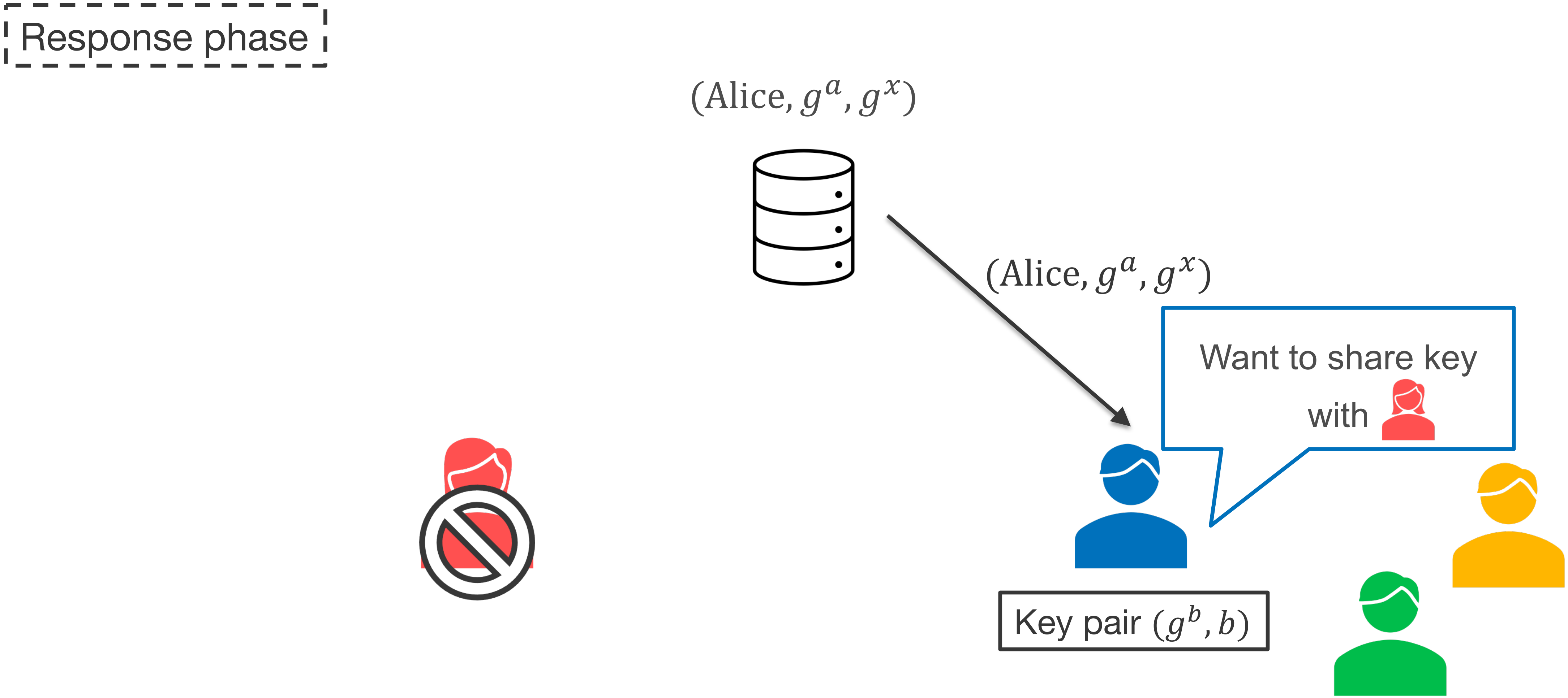
(Alice, g^a, g^x)

1. Gen long-term key (g^a, a)
2. Gen first message g^x
3. Store x as state

Key pair (g^a, a)
State x

Recap: X3DH protocol

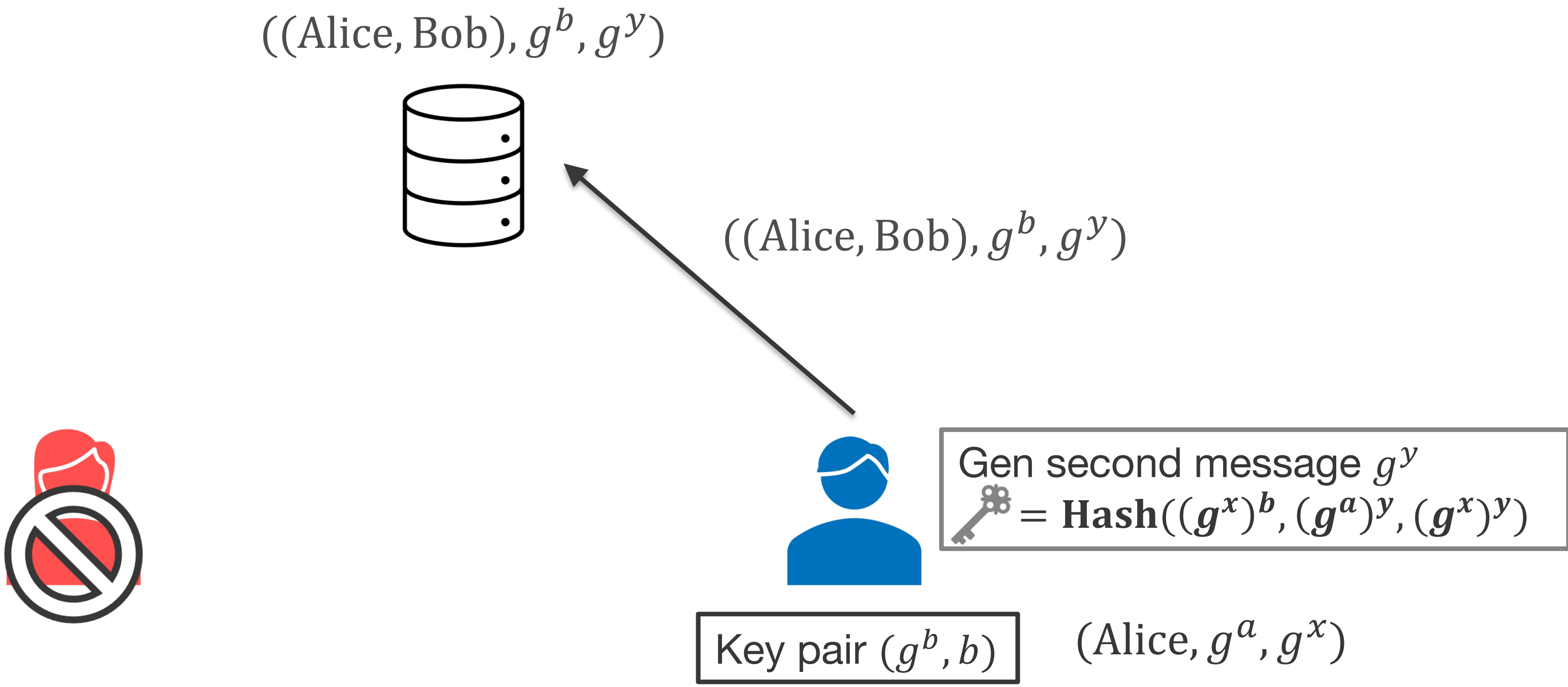
Asynchronous key exchange protocol with the help of server



Recap: X3DH protocol

Asynchronous key exchange protocol with the help of server

Response phase

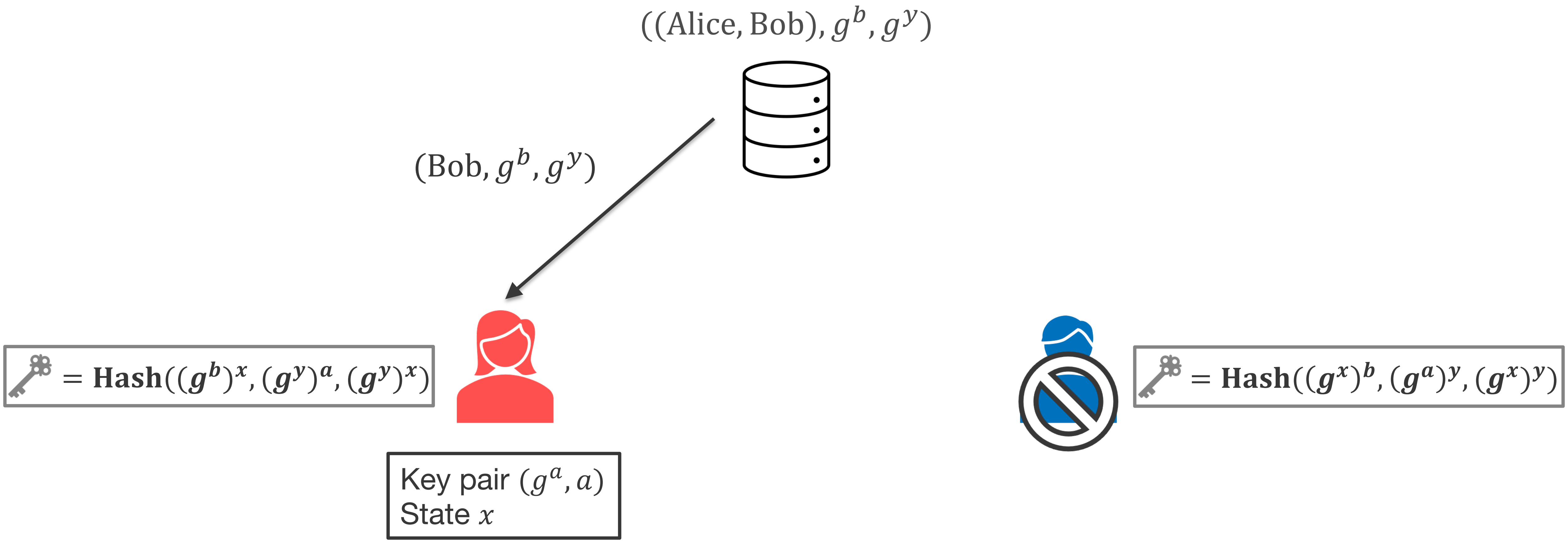


※ He sends ciphertexts to Alice at the same time

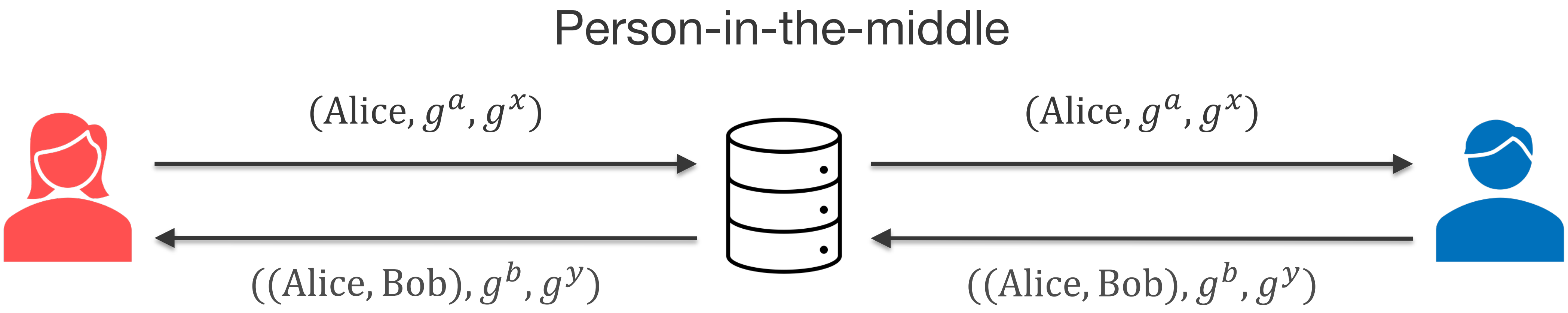
Recap: X3DH protocol

<u>Asynchronous</u> key exchange protocol with the help of server

Finalize phase



On a closer look



$= \mathbf{Hash}((g^b)^x, (g^y)^a, (g^y)^x)$

$= \mathbf{Hash}((g^x)^b, (g^a)^y, (g^x)^y)$

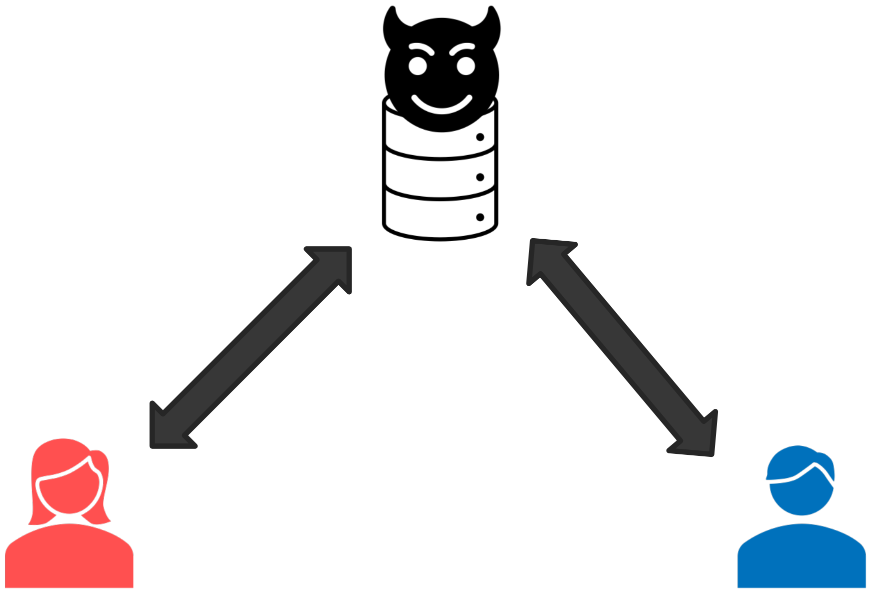
X3DH protocol looks like a general authentication key exchange (AKE)

Starting point: X3DH ≈ Authenticated Key Exchange

Consider X3DH protocol as **a specific type of AKE protocol**

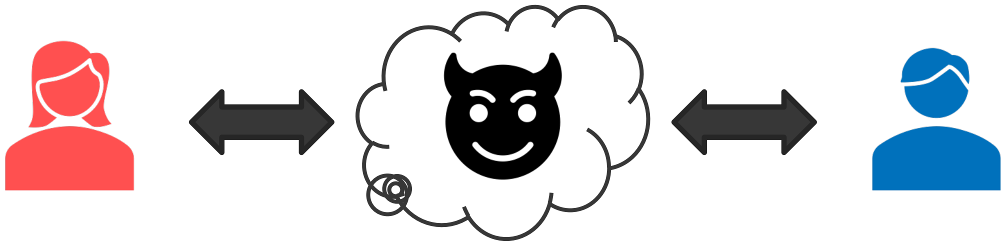
Signal-conforming AKE (SC-AKE)

Model of X3DH



≈

Model of AKE

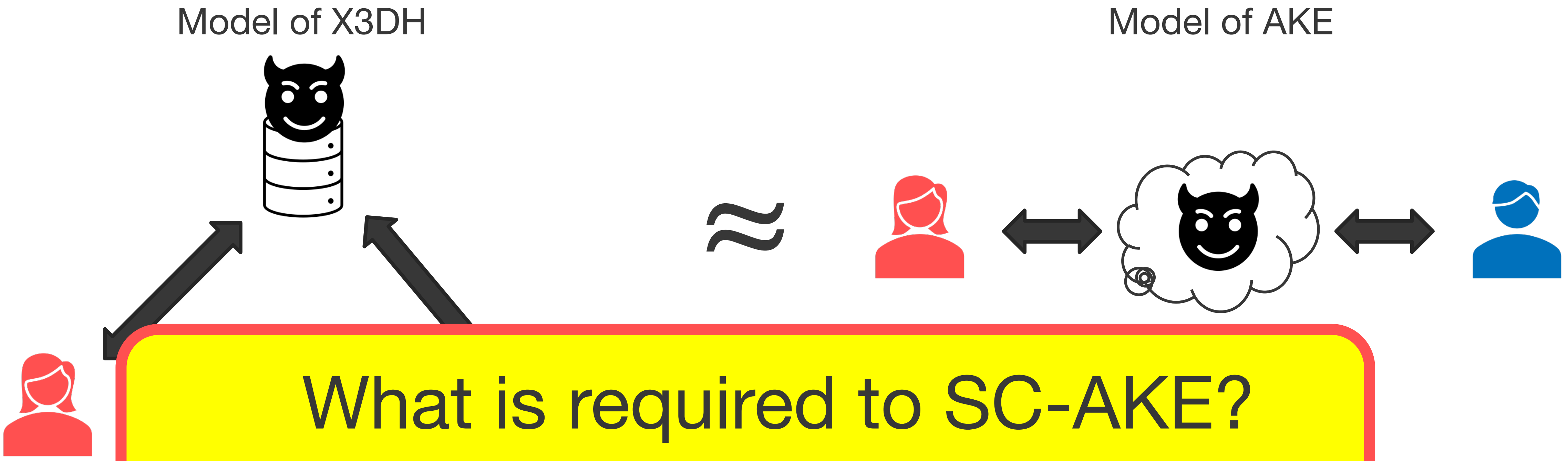


By viewing “server” as ”AKE adversary controlling channel",
X3DH protocol can be considered as an AKE protocol

Starting point: X3DH ≈ Authenticated Key Exchange

Consider X3DH protocol as **a specific type of AKE protocol**

Signal-conforming AKE (SC-AKE)



What is required to SC-AKE?

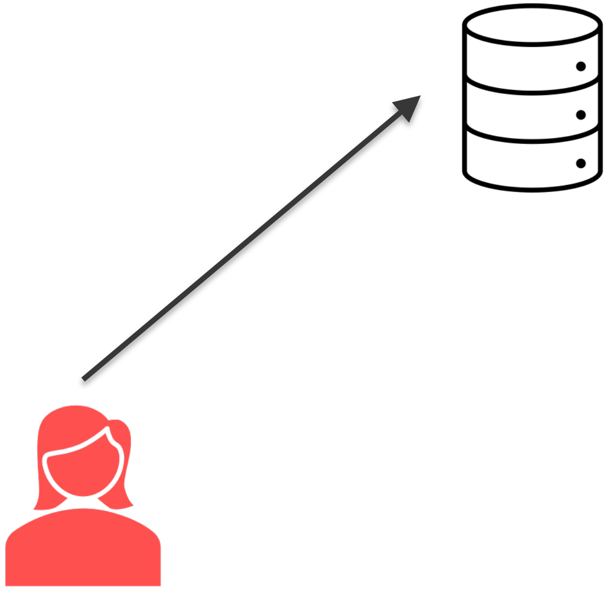
"**Functionality**" and "**Security**"

X3DH protocol can be considered as a kind of AKE protocol

Requirement (1): Functionality of SC-AKE

1. 2-round
2. First-message must be independent from communication partners

Initialization phase



1. Gen long-term key (g^a, a)
2. Gen first message g^x
3. Store x as state

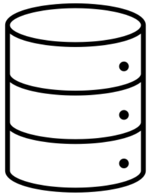
Key pair (g^a, a)
State x

Requirement (1): Functionality of SC-AKE

1. 2-round
2. First-message must be independent from communication partners

Initialization phase

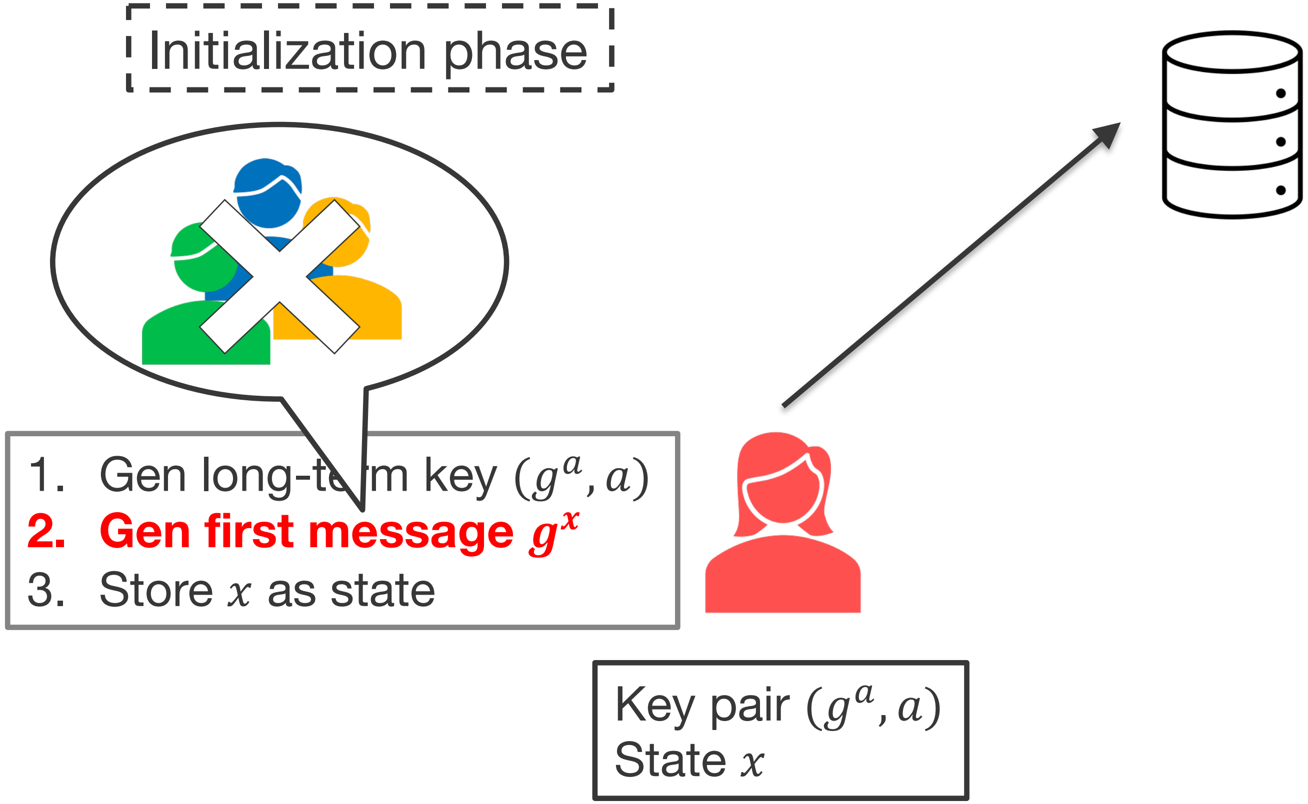
1. Gen long-term key (g^a, a)
2. **Gen first message g^x**
3. Store x as state



Key pair (g^a, a)
State x

Requirement (1): Functionality of SC-AKE

1. 2-round
2. First-message must be independent from communication partners

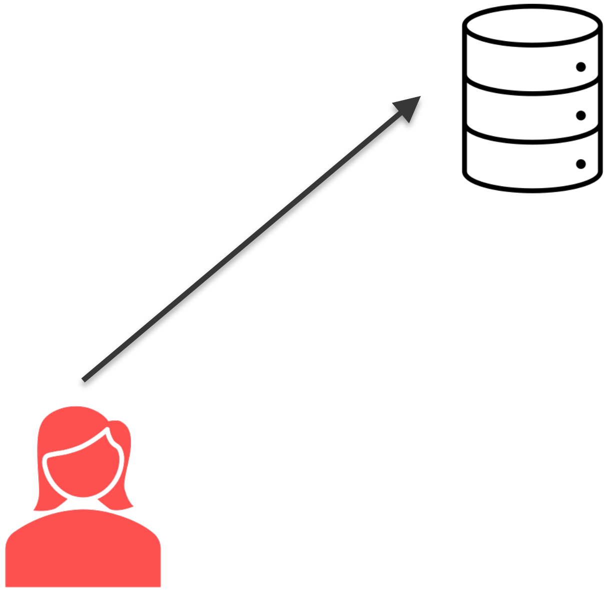


(2-round)
Receiver Obliviousness

Requirement (2): Security of SC-AKE

Double Ratchet protocol is secure against state leakage
⇒ SC-AKE also needs the same level of security

Initialization phase



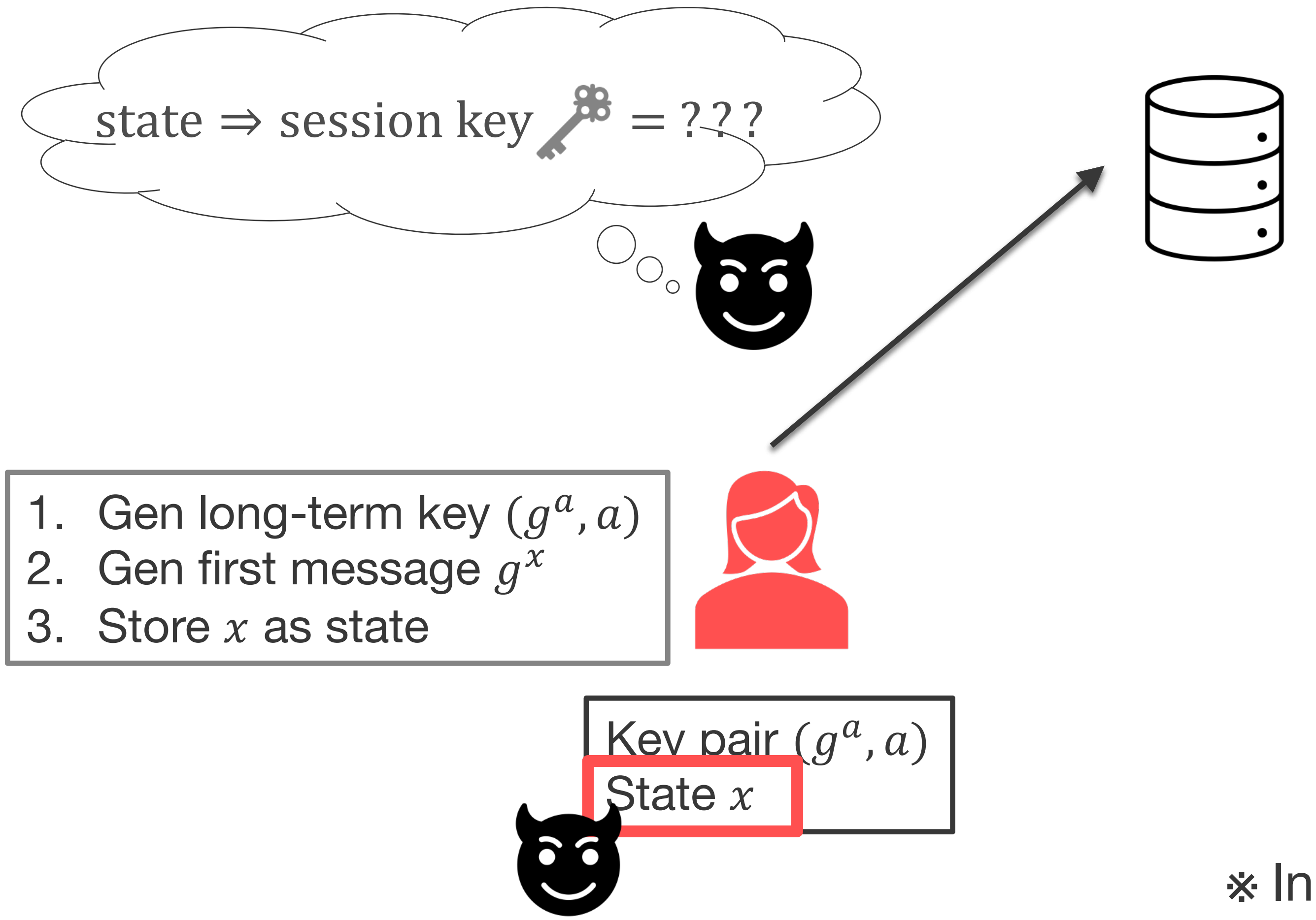
1. Gen long-term key (g^a, a)
2. Gen first message g^x
3. **Store x as state**

Key pair (g^a, a)
State x

※ In the literature of AKE, it is called CK security

Requirement (2): Security of SC-AKE

Double Ratchet protocol is **secure against state leakage**
⇒ <u>SC-AKE also needs the same level of security</u>



<u>State Leakage Secure</u>

※ In the literature of AKE, it is called CK security

Contribution 2

Theory: Generic construction of SC-AKE

Existing post-quantum AKE are insufficient for Signal

Constructions (2-round)	Post-quantum	Receiver obliviousness	State leakage secure
DH-type construction [BFG+20, dKGV20, KTAT20]	△ Gap-CSIDH	○	✕*
SIG-KEM-SIG construction [Shoup99]	○	○	✕*
KEM-KEM-KEM construction [FSXY12, FSXY13, XLL+18, HKSU20, XAY+20]	○	✕	○

*: NAXOS trick makes it secure against state leakage
(NAXOS trick: store ephemeral randomness instead of actual state and reconstruct state)

Proposed construction

Proposed construction satisfies all necessary requirements

Constructions (2-round)	Post-quantum	Receiver obliviousness	State leakage secure
DH-type construction [BFG+20, dKGV20, KTAT20]	△ Gap-CSIDH	○	✕*
SIG-KEM-SIG construction [Shoup99]	○	○	✕*
KEM-KEM-KEM construction [FSXY12, FSXY13, XLL+18, HKSU20, XAY+20]	○	✕	○
Proposed generic construction	○	○	○

*: NAXOS trick makes it secure against state leakage
(NAXOS trick: store ephemeral randomness instead of actual state and reconstruct state)

Starting point: Existing generic construction of post-quantum AKE

SIG-KEM-SIG

(vk_A, sk_A) (vk_B, sk_B)

$(ek_T, dk_T) \leftarrow \mathrm{KEM.Gen}()$
$\sigma_A \leftarrow \mathrm{SIG.Sign}(sk_A, ek_T)$

$\xrightarrow{ek_T,\ \sigma_A}$

Verify σ_A
$(K_T, C_T) \leftarrow \mathrm{KEM.Enc}(ek_T)$
$\sigma_B \leftarrow \mathrm{SIG.Sign}(sk_B, sid)$
$= \mathrm{Hash}(K_T)$

$\xleftarrow{C_T,\ \sigma_B}$

Verify σ_B
$K_T \leftarrow \mathrm{KEM.Dec}(dk_T, C_T)$
$= \mathrm{Hash}(K_T)$

※ $sid = id_A || id_B || vk_A || vk_B || ek_T || C_T$

KEM-KEM-KEM

(ek_A, dk_A) (ek_B, dk_B)

$(ek_T, dk_T) \leftarrow \mathrm{KEM.Gen}()$
$(K_B, C_B) \leftarrow \mathrm{KEM.Enc}(ek_B)$

$\xrightarrow{ek_T,\ C_B}$

$(K_T, C_T) \leftarrow \mathrm{KEM.Enc}(ek_T)$
$(K_A, C_A) \leftarrow \mathrm{KEM.Enc}(ek_A)$
$K_B \leftarrow \mathrm{KEM.Dec}(dk_T, C_T)$
$= \mathrm{Hash}(K_T, K_A, K_B)$

$\xleftarrow{C_T,\ C_A}$

$K_T \leftarrow \mathrm{KEM.Dec}(dk_T, C_T)$
$K_A \leftarrow \mathrm{KEM.Dec}(dk_A, C_A)$
$= \mathrm{Hash}(K_T, K_A, K_B)$

Cons of existing generic construction

SIG-KEM-SIG

(vk_A, sk_A) (vk_B, sk_B)

$(ek_T, \boxed{dk_T}) \leftarrow \mathrm{KEM.Gen}()$

$\sigma_A \leftarrow \mathrm{SIG.Sign}(sk_A, ek_T)$

$\xrightarrow{ek_T,\ \sigma_A}$

Verify σ_A

$(K_T, C_T) \leftarrow \mathrm{KEM.Enc}(ek_T)$

$\sigma_B \leftarrow \mathrm{SIG.Sign}(sk_B, sid)$

$= \mathrm{Hash}(K_T)$

$dk_T + C_T \Rightarrow K_T =$

$\xleftarrow{C_T,\ \sigma_B}$

Verify σ_B

$K_T \leftarrow \mathrm{KEM.Dec}(dk_T, C_T)$

$= \mathrm{Hash}(K_T)$

※ $sid = id_A || id_B || vk_A || vk_B || ek_T || C_T$

KEM-KEM-KEM

(ek_A, dk_A) (ek_B, dk_B)

$(ek_T, dk_T) \leftarrow \mathrm{KEM.Gen}()$

$(K_B, C_B) \leftarrow \mathrm{KEM.Enc}(ek_B)$

$\xrightarrow{ek_T,\ C_B}$

$(K_T, C_T) \leftarrow \mathrm{KEM.Enc}(ek_T)$

$(K_A, C_A) \leftarrow \mathrm{KEM.Enc}(ek_A)$

$K_B \leftarrow \mathrm{KEM.Dec}(dk_T, C_T)$

$= \mathrm{Hash}(K_T, K_A, K_B)$

$\xleftarrow{C_T,\ C_A}$

$K_T \leftarrow \mathrm{KEM.Dec}(dk_T, C_T)$

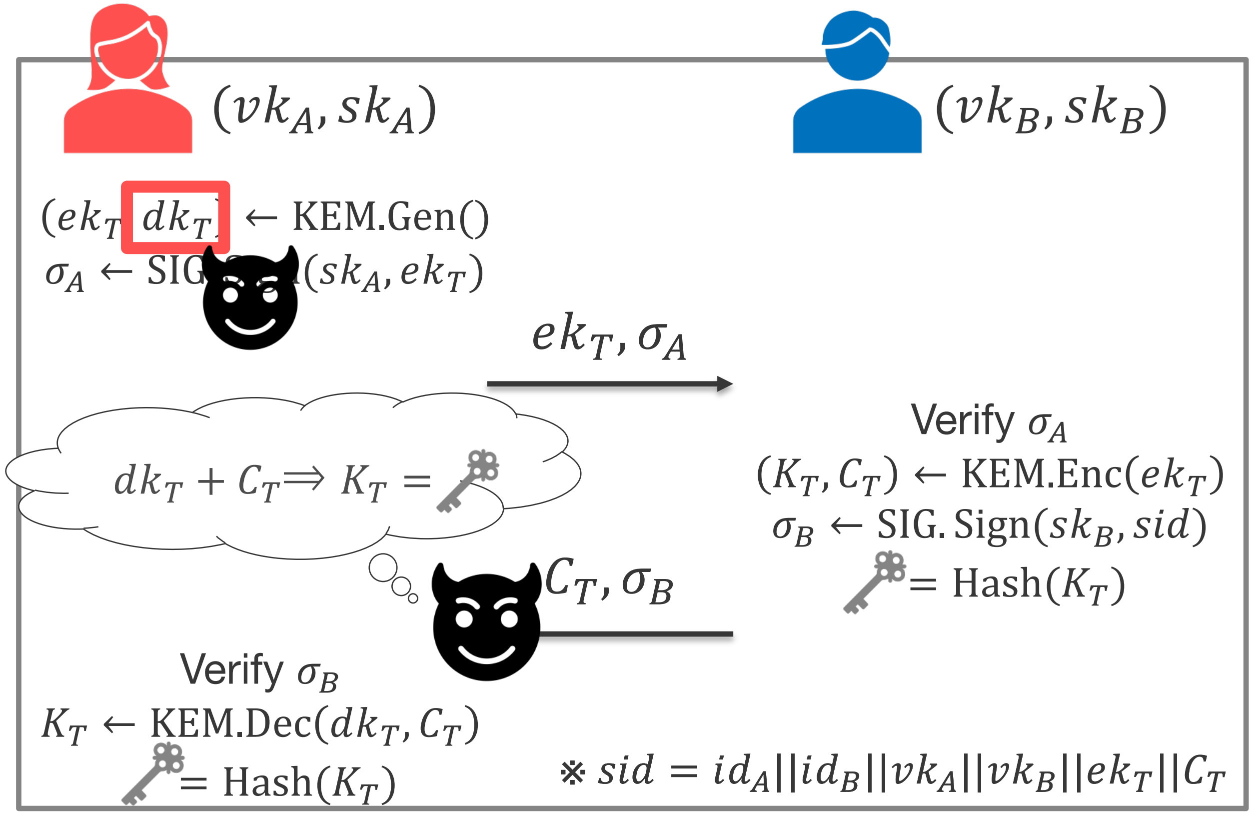
$K_A \leftarrow \mathrm{KEM.Dec}(dk_A, C_A)$

$= \mathrm{Hash}(K_T, K_A, K_B)$

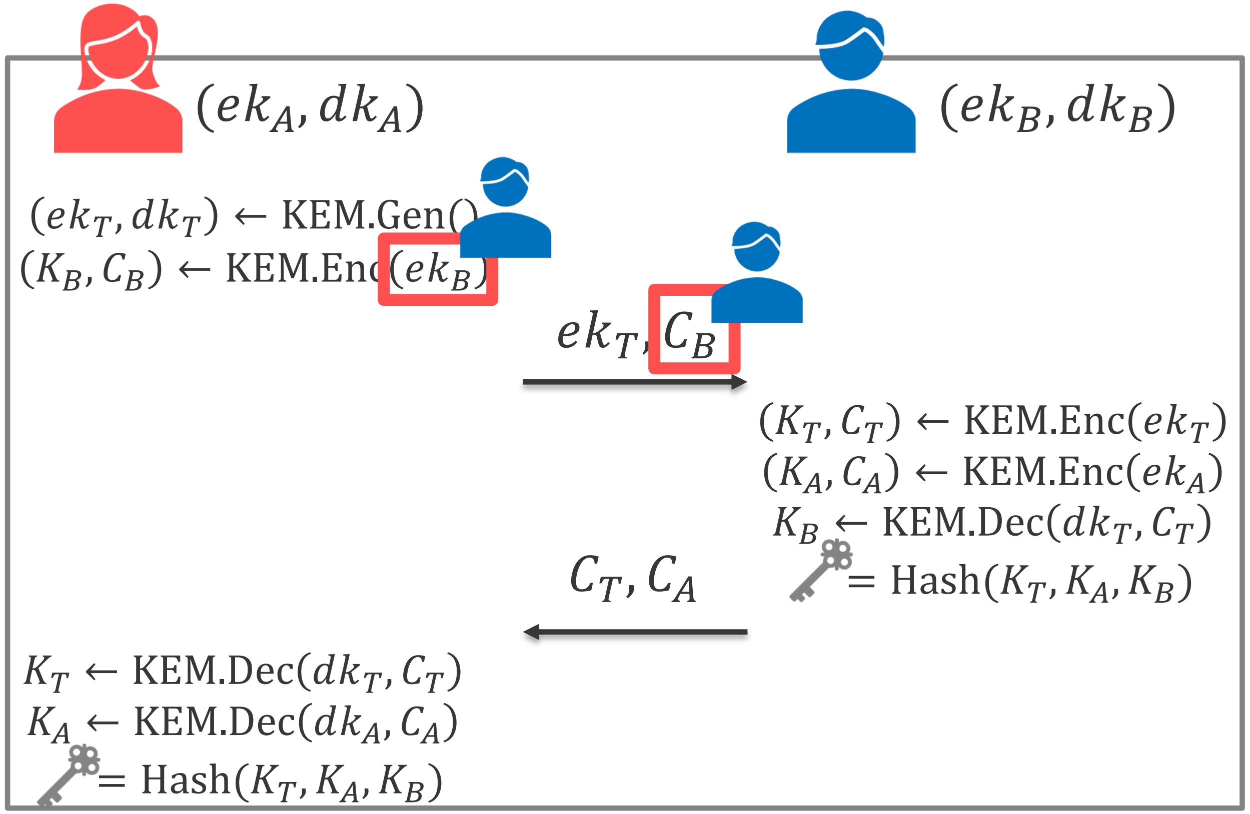
If state (dec. key dk_T) is exposed, session key is also exposed

Cons of existing generic construction

SIG-KEM-SIG



KEM-KEM-KEM



✖ If state (dec. key dk_T) is exposed, session key is also exposed

✖ First message depends on the peer

Pros of existing generic construction

SIG-KEM-SIG

(vk_A, sk_A) (vk_B, sk_B)

$\boxed{\begin{array}{l}(ek_T, dk_T) \leftarrow \text{KEM.Gen}() \\ \sigma_A \leftarrow \text{SIG.Sign}(sk_A, ek_T)\end{array}}$

$\xrightarrow{ek_T, \sigma_A}$

Verify σ_A
$(K_T, C_T) \leftarrow \text{KEM.Enc}(ek_T)$
$\sigma_B \leftarrow \text{SIG.Sign}(sk_B, sid)$
$= \text{Hash}(K_T)$

$\xleftarrow{C_T, \sigma_B}$

Verify σ_B
$K_T \leftarrow \text{KEM.Dec}(dk_T, C_T)$
$= \text{Hash}(K_T)$

※ $sid = id_A || id_B || vk_A || vk_B || ek_T || C_T$

KEM-KEM-KEM

(ek_A, dk_A) (ek_B, dk_B)

$(ek_T, dk_T) \leftarrow \text{KEM.Gen}()$
$(K_B, C_B) \leftarrow \text{KEM.Enc}(ek_B)$

$\xrightarrow{ek_T, C_B}$

$(K_T, C_T) \leftarrow \text{KEM.Enc}(ek_T)$
$(K_A, C_A) \leftarrow \text{KEM.Enc}(ek_A)$
$K_B \leftarrow \text{KEM.Dec}(dk_T, C_T)$
$= \text{Hash}(K_T, K_A, K_B)$

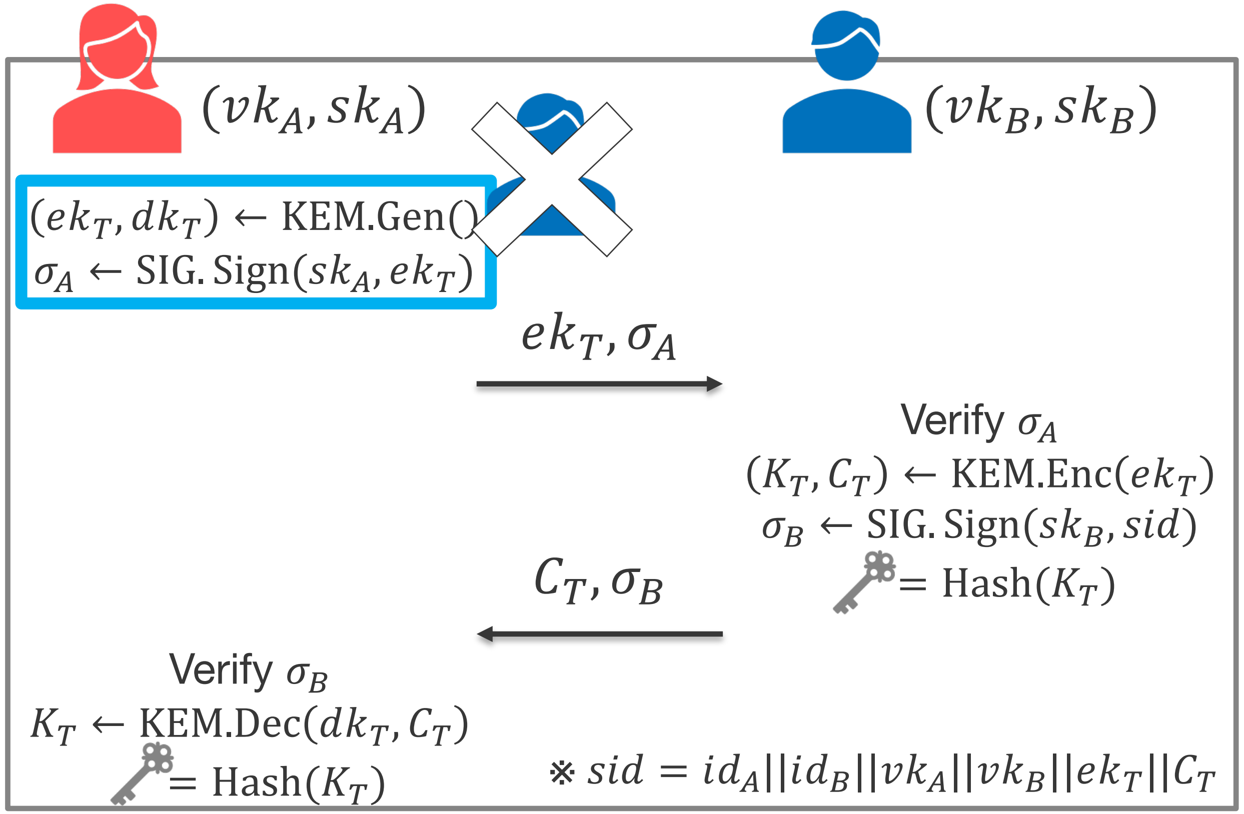
$\xleftarrow{C_T, C_A}$

$K_T \leftarrow \text{KEM.Dec}(dk_T, C_T)$
$K_A \leftarrow \text{KEM.Dec}(dk_A, C_A)$
$= \text{Hash}(K_T, K_A, K_B)$

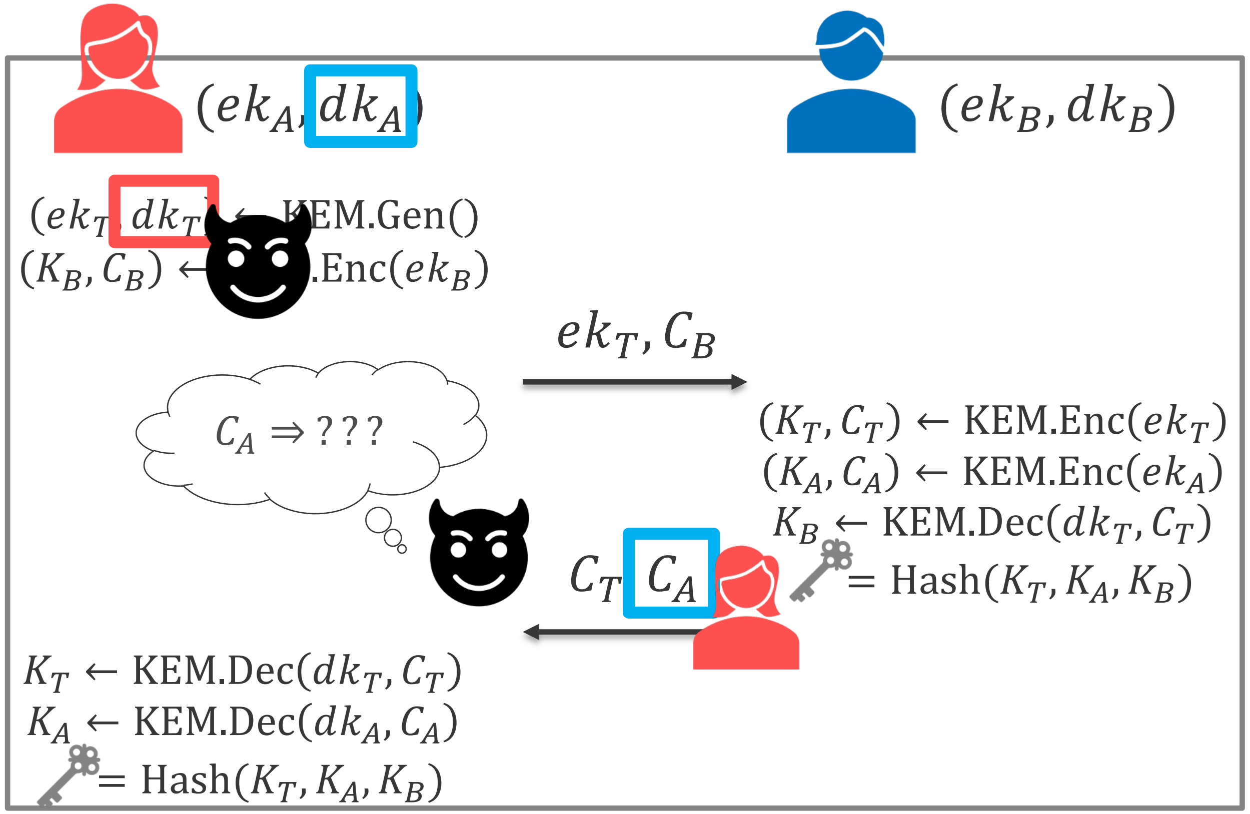
Receiver oblivious

Pros of existing generic construction

SIG-KEM-SIG



KEM-KEM-KEM



✔ Receiver oblivious

✔ State leakage secure

Pros of existing generic construction

SIG-KEM-SIG

(vk_A, sk_A) (vk_B, sk_B)

$(ek_T, dk_T) \leftarrow \text{KEM.Gen}()$
$\sigma_A \leftarrow \text{SIG.Sign}(sk_A, ek_T)$

ek_T, σ_A →

Verify σ_A
$(K_T, C_T) \leftarrow \text{KEM.Enc}(ek_T)$
$\sigma_B \leftarrow \text{SIG.Sign}(sk_B, sid)$
$= \text{Hash}(K_T)$

← C_T, σ_B

Verify σ_B
$K_T \leftarrow \text{KEM.Dec}(dk_T, C_T)$
$= \text{Hash}(K_T)$

※ $sid = id_A \| id_B \| vk_A \| vk_B \| ek_T \| C_T$

KEM-KEM-KEM

$(ek_A, \boxed{dk_A})$ (ek_B, dk_B)

$(ek_T, \boxed{dk_T}) \leftarrow \text{KEM.Gen}()$
$(K_B, C_B) \leftarrow \text{KEM.Enc}(ek_B)$

ek_T, C_B →

$C_A \Rightarrow ???$

$(K_T, C_T) \leftarrow \text{KEM.Enc}(ek_T)$
$(K_A, C_A) \leftarrow \text{KEM.Enc}(ek_A)$
$K_B \leftarrow \text{KEM.Dec}(dk_T, C_T)$
$= \text{Hash}(K_T, K_A, K_B)$

← $C_T, \boxed{C_A}$

$K_T \leftarrow \text{KEM.Dec}(dk_T, C_T)$
$K_A \leftarrow \text{KEM.Dec}(dk_A, C_A)$
$= \text{Hash}(K_T, K_A, K_B)$

Receiver oblivious

State leakage secure

Can we make the best of both worlds?

Recap: existing generic construction of post-quantum AKE

SIG	KEM	SIG
Authenticate Alice “explicitly”	session key	Authenticate Bob “explicitly”


Receiver oblivious


Insecure if Alice's state is exposed

KEM KEM KEM

Recap: existing generic construction of post-quantum AKE

SIG	KEM	SIG	
⇩	⇩	⇩	
Authenticate Alice “explicitly”	session key	Authenticate Bob “explicitly”	✓ Receiver oblivious ✗ Insecure if Alice's state is exposed

KEM	KEM	KEM	
⇩	⇩	⇩	
Authenticate Bob “implicitly” + session key	session key	Authenticate Alice “implicitly” + session key	✓ State leakage secure ✗ First message depends on Bob for authentication

Construction of proposed SC-AKE

SIG

Authenticate Alice
“explicitly”

KEM

session key

SIG

Authenticate Bob
“explicitly”

KEM

Authenticate Bob
“implicitly”
+ session key

KEM

session key

KEM

Authenticate Alice
“implicitly”
+ session key

Construction of proposed SC-AKE

SIG
⇩
Authenticate Alice
“explicitly”

KEM
⇩
session key

SIG
⇩
Authenticate Bob
“explicitly”

KEM
⇩
Authenticate Bob
“implicitly”
+ session key

KEM
⇩
session key

KEM
⇩
Authenticate Alice
“implicitly”
+ session key

State leakage secure

Construction of proposed SC-AKE

SIG

⇩

Authenticate Alice
“explicitly”

KEM

⇩

session key

SIG

⇩

Authenticate Bob
“explicitly”

Receiver oblivious

KEM

⇩

Authenticate Bob
“implicitly”
+ session key

KEM

⇩

session key

KEM

⇩

Authenticate Alice
“implicitly”
+ session key

State leakage secure

Construction of proposed SC-AKE

Proposed = ⊥-KEM-(KEM, SIG) construction

$((ek_A, vk_A), (dk_A, sk_A))$

$((ek_B, vk_B), (dk_B, sk_B))$

$(ek_T, dk_T) \leftarrow \mathrm{KEM.Gen}()$

$\xrightarrow{\quad ek_T \quad}$

$(K_T, C_T) \leftarrow \mathrm{KEM.Enc}(ek_T)$

$(K_A, C_A) \leftarrow \mathrm{KEM.Enc}(ek_A)$

$\sigma_B \leftarrow \mathrm{SIG.Sign}(sk_B, sid)$

$= \mathrm{Hash}(K_T, K_A)$

$\xleftarrow{\quad C_T, C_A, \sigma_B \quad}$

Verify σ_B

$K_T \leftarrow \mathrm{KEM.Dec}(dk_T, C_T)$

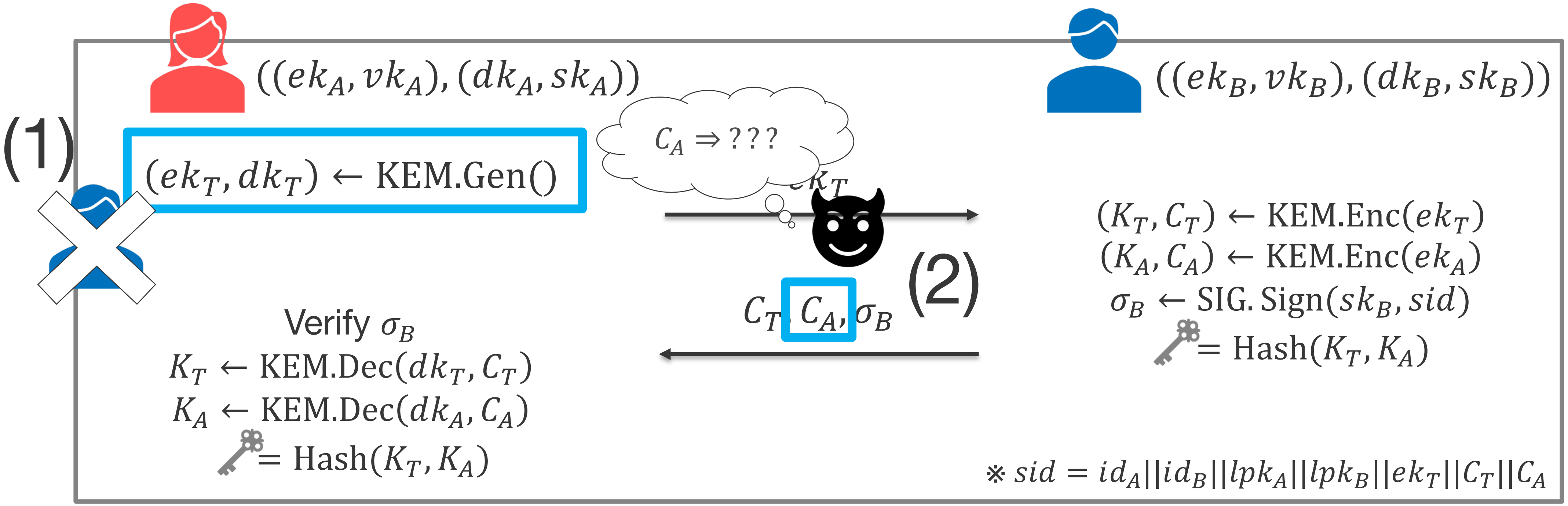
$K_A \leftarrow \mathrm{KEM.Dec}(dk_A, C_A)$

$= \mathrm{Hash}(K_T, K_A)$

※ $sid = id_A || id_B || lpk_A || lpk_B || ek_T || C_T || C_A$

Construction of proposed SC-AKE

Proposed = ⊥-KEM-(KEM, SIG) construction



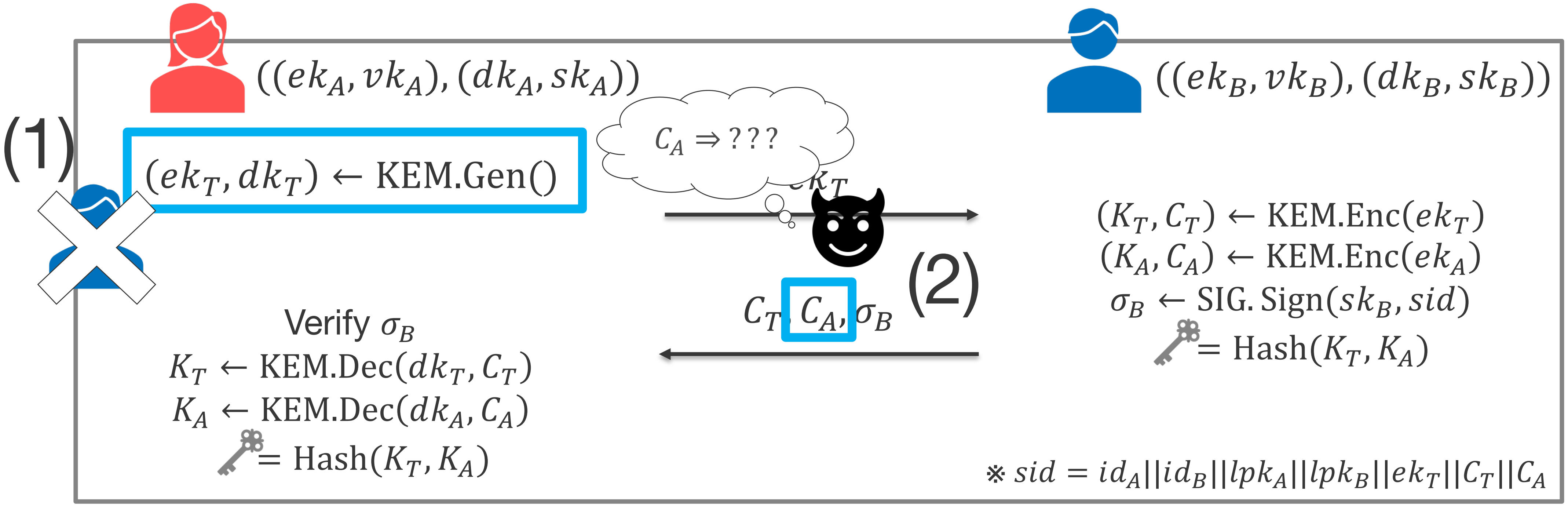
(1) Receiver obliviousness

(2) State leakage secure

To compute the session key, both dk_A and dk_T are needed

Construction of proposed SC-AKE

Proposed = ⊥-KEM-(KEM, SIG) construction



(1) Receiv

(2) State l

We can make the best of both worlds!

To compute the session key, both dk_A and dk_T are needed

Summary of our results

1. Generic construction of Signal-conforming AKE based on KEM and SIG
   - ✓ 2-round and receiver oblivious
   - ✓ State leakage secure
2. <u>Deniable</u> SC-AKE using ring signatures and NIZKs

Signal protocol

PQ SC-AKE = KEM + SIG

“Establish shared secret key”

PQ Double Ratchet [ACD19]

“Encrypted communication”

⇨ **The first post-quantum Signal protocol!**

Contribution 3

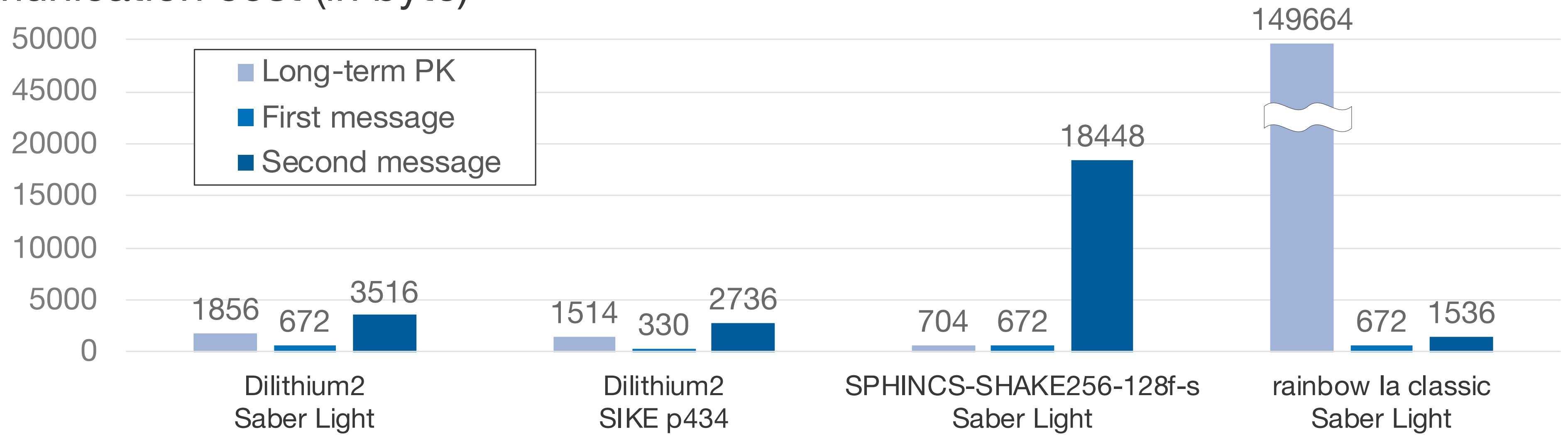
Practice: Implementation of proposed SC-AKE

Implementation details

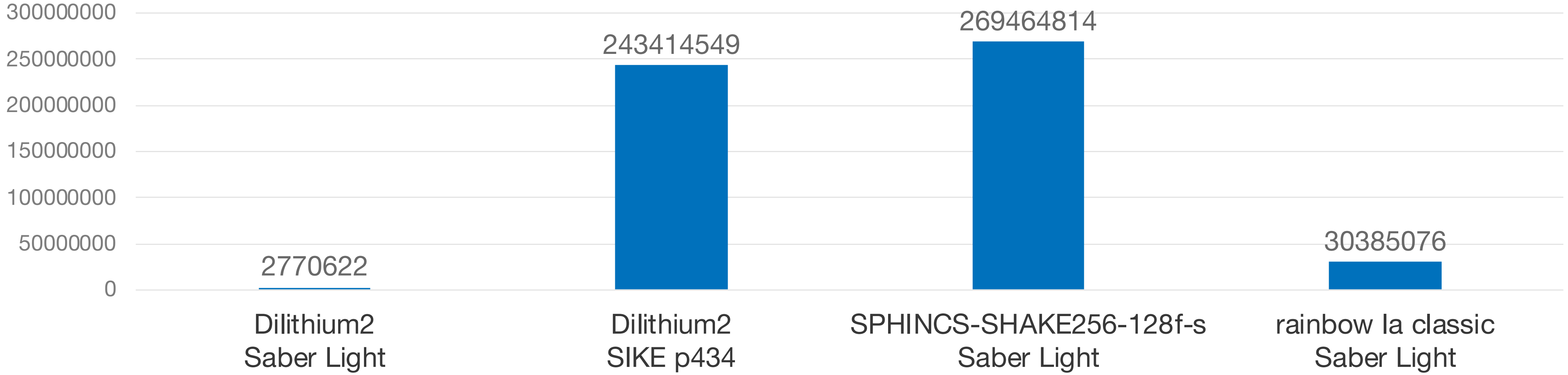
- Use post-quantum KEMs and signature schemes submitted for the NIST PQC standardization
- Pair variants of KEMs and signature schemes corresponding to the same security level (levels 1, 3 and 5)
  - Obtain 128 different instantiations of post-quantum SC-AKE
- Evaluate computation cost (CPU cycles) and communication cost (data size)

Implementation results (only 4 instantiations, NIST level I)

Communication cost (in byte)



Computation cost (in CPU cycle)



Conclusion

Design and implementation of generic construction of Signal-conforming AKE protocol

Theory

- Formalization of X3DH protocol as a specific type of AKE (SC-AKE)
  - Define required functionality and security
- Generic construction of <u>post-quantum</u> SC-AKE from KEM and signature

Practice

- Implementation of proposed SC-AKE with NIST PQC candidates
  - Evaluate computation and communication costs

Realize the first <u>practical and post-quantum</u> Signal protocol!

References

- [Shoup99] V. Shoup, On Formal Models for Secure Key Exchange, Theory of Cryptography Library, https://www.shoup.net/papers/skey.pdf, 1999.
- [FSXY12] A. Fujioka, K. Suzuki, K. Xagawa, and K. Yoneyama. Strongly secure authenticated key exchange from factoring, codes, and lattices. PKC 2012, pp. 467–484.
- [FSXY13] A. Fujioka, K. Suzuki, K. Xagawa, and K. Yoneyama. Prac- tical and post-quantum authenticated key exchange from one-way secure key encapsulation mechanism. ASIACCS 13, pp. 83–94.
- [MP16a] M. Marlinspike and T. Perrin. The Double Ratchet Algorithm. https://signal.org/docs/specifications/doubleratchet/.
- [MP16b] M. Marlinspike and T. Perrin. The x3dh key agreement protocol. https://signal.org/docs/specifications/x3dh/.
- [CGC+17] K. Cohn-Gordon, C. Cremers, B. Dowling, L. Garratt, and D. Stebila, “A Formal Security Analysis of the Signal Messaging Protocol,” in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2017, pp. 451–466.

References

- [XLL+18] H. Xue, X. Lu, B. Li, B. Liang, and J. He. Understanding and constructing AKE via double-key key encapsulation mechanism. ASIACRYPT 2018, pp. 158–189.
- [ACD19] J. Alwen, S. Coretti, and Y. Dodis. The double ratchet: Security notions, proofs, and modularization for the Signal protocol. EUROCRYPT 2019, pp. 129–158.
- [BFG+20] J. Brendel, M. Fischlin, F. Günther, C. Janson, and D. Stebila. Towards post-quantum security for signal's x3dh hand- shake. In SAC 2020.
- [dKGV20] B. d Kock, K. Gjøsteen, and M. Veroni. Practical isogeny-based key exchange with optimal tightness. In SAC 2020.
- [KTAT20] T. Kawashima, K. Takashima, Y. Aikawa, and T. Takagi. An efficient authenticated key exchange from random self-reducibility on csidh. In ICISC 2020.
- [HKSU20] K. Hövelmanns, E. Kiltz, S. Schäge, and D. Unruh. Generic authenticated key exchange in the quantum random oracle model. PKC 2020, pp. 389–422.
- [XAY+20] H. Xue, M. H. Au, R. Yang, B. Liang, and H. Jiang. Com- pact authenticated key exchange in the quantum random or- acle model. Cryptology ePrint Archive, Report 2020/1282.

References

- [MP16a] M. Marlinspike and T. Perrin. The Double Ratchet Algorithm. https://signal.org/docs/specifications/doubleratchet/.
- [MP16b] M. Marlinspike and T. Perrin. The x3dh key agreement protocol. https://signal.org/docs/specifications/x3dh/.
- [CGC+17] K. Cohn-Gordon, C. Cremers, B. Dowling, L. Garratt, and D. Stebila, “A Formal Security Analysis of the Signal Messaging Protocol,” in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2017, pp. 451–466.
- [ACD19] J. Alwen, S. Coretti, and Y. Dodis. The double ratchet: Security notions, proofs, and modularization for the Signal protocol. EUROCRYPT 2019, pp. 129–158.