Post-Quantum

Cryptography Conference

Symmetric Key Exchange: Lightweight Alternatives for a Post-Quantum IoT

Bor de Kock

Assistant Professor of Cryptology at NTNU Trondheim

Symmetric Key Exchange: Lightweight Alternatives for a Post-Quantum IoT

Bor de Kock

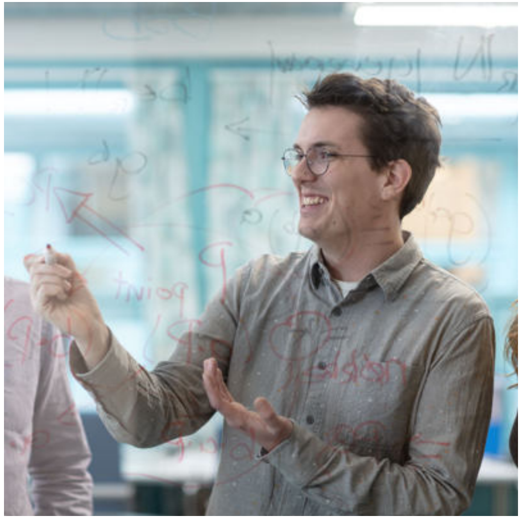
Post-Quantum Cryptography Conference

November 8, 2023

About me

Bor de Kock, assistant professor cryptology

- MSc from Eindhoven University of Technology
- PhD on Post-Quantum Key Exchange, from NTNU



My research interests:

Post-quantum cryptography, key exchange, password-based crypto, authentication, security models, ratcheting, etc.

in other words: practical crypto!


Norwegian Sea
Iceland Sea
ÍSLAND
Akureyri
Reykjavík
Gunnbjørn Fjeld
Trondheim
NORGE
SVERIGE
SUOMI
Tromsø
Oulu
Umeå
Tampere
Turku
Helsinki
Bergen
Oslo
Uppsala
Stockholm
Tallinn
EESTI
Göteborg
LATVIJA
LIETUVA
DANMARK
København
North Sea
UNITED KINGDOM
IRELAND
Glasgow
Belfast
Dublin
Liverpool
Birmingham
London
Hamburg
Bremen
Hannover
Berlin
NEDERLAND
Rotterdam
Antwerpen
Düsseldorf
Köln
DEUTSCHLAND
Leipzig
Dresden
Szczecin
Gdańsk
Bydgoszcz
Poznań
POLAND
Łódź
Warszawa
Wrocław
Lublin
Vilnius
Białystok
Мінск
БЕЛАРУСЬ
Мурманск
Архангельск
Петрозаводск
Санкт-Петербург
Москва
Нижний Новгород
Ярославль
Смоленск
Cairngorms National Park
Streymoy
Lindesnes

Norwegian University of Science and Technology

Trondheim, Gjøvik and Ålesund

Largest university in Norway

43.000 students

32 cryptographers

Let’s talk key exchange…

Let’s talk key exchange…

- Making key exchange post-quantum is an ongoing effort

Let’s talk key exchange…

- Making key exchange post-quantum is an ongoing effort
- Most serious candidates are inefficient, compared to SoA

Let’s talk key exchange…

- Making key exchange post-quantum is an ongoing effort
- Most serious candidates are inefficient, compared to SoA

Let’s talk key exchange…

- Making key exchange post-quantum is an ongoing effort
- Most serious candidates are inefficient, compared to SoA

- Symmetric algorithms such as AES are post-quantum

Let’s talk key exchange…

- Making key exchange post-quantum is an ongoing effort
- Most serious candidates are inefficient, compared to SoA

- Symmetric algorithms such as AES are post-quantum
- Symmetric algorithms such as AES are very efficient

Let’s talk key exchange…

- Making key exchange post-quantum is an ongoing effort
- Most serious candidates are inefficient, compared to SoA

- Symmetric algorithms such as AES are post-quantum
- Symmetric algorithms such as AES are very efficient
- Many security features we like are missing.

What do we want to achieve?

What do we want to achieve?

- Authenticated key exchange for very constrained devices

What do we want to achieve?

- Authenticated key exchange for very constrained devices
- Pre-shared symmetric keys

What do we want to achieve?

- Authenticated key exchange for very constrained devices
- Pre-shared symmetric keys
- Forward security

What do we want to achieve?

- Authenticated key exchange for very constrained devices
- Pre-shared symmetric keys
- Forward security
- Synchronization

What do we want to achieve?

- Authenticated key exchange for very constrained devices
- Pre-shared symmetric keys
- Forward security
- Synchronization
- Concurrent Correctness

In this talk…

Symmetric Key Exchange with Full Forward Security and Robust Synchronization

Colin Boyd, Gareth T. Davies, Bor de Kock, Kai Gellert, Tibor Jager and Lise Millerjord

In this talk…

Symmetric Key Exchange with Full Forward Security and Robust Synchronization

Colin Boyd, Gareth T. Davies, Bor de Kock, Kai Gellert, Tibor Jager and Lise Millerjord

- 3 very efficient AKE protocols with linear key evolution

In this talk…

Symmetric Key Exchange with Full Forward Security and Robust Synchronization

Colin Boyd, Gareth T. Davies, Bor de Kock, Kai Gellert, Tibor Jager and Lise Millerjord

- 3 very efficient AKE protocols with linear key evolution
- 2 AKE protocols with non-linear key evolution

In this talk…

Symmetric Key Exchange with Full Forward Security and Robust Synchronization

Colin Boyd, Gareth T. Davies, Bor de Kock, Kai Gellert, Tibor Jager and Lise Millerjord

- 3 very efficient AKE protocols with linear key evolution
- 2 AKE protocols with non-linear key evolution
- Framework for protocol analysis

In this talk…

Symmetric Key Exchange with Full Forward Security and Robust Synchronization

Colin Boyd, Gareth T. Davies, Bor de Kock, Kai Gellert, Tibor Jager and Lise Millerjord

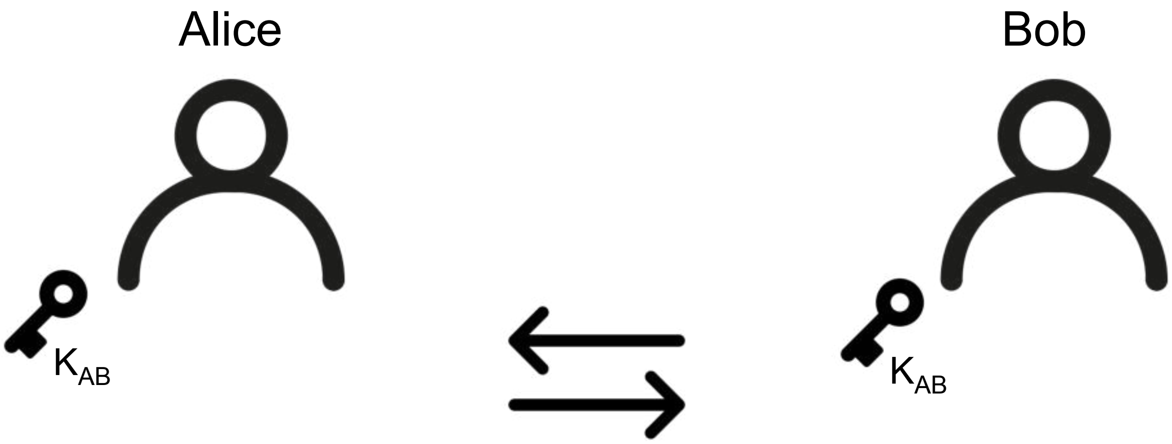
- 3 very efficient AKE protocols with linear key evolution
- 2 AKE protocols with non-linear key evolution
- Framework for protocol analysis
- Formalization of synchronization robustness as a security property

Authenticated Key Exchange - AKE

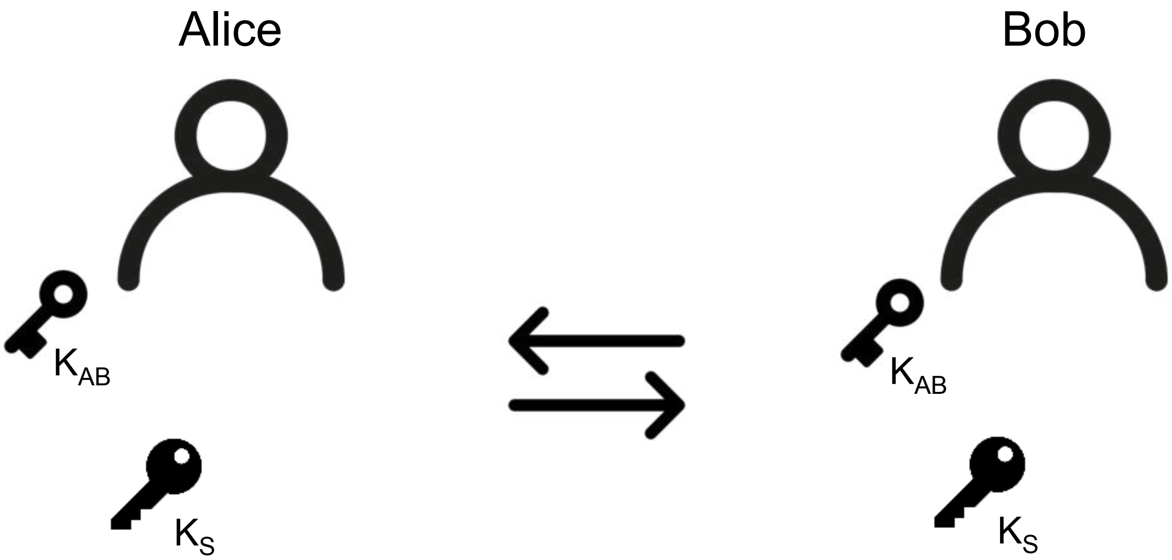
Authenticated Key Exchange - AKE

Authenticated Key Exchange - AKE

Authenticated Key Exchange - AKE



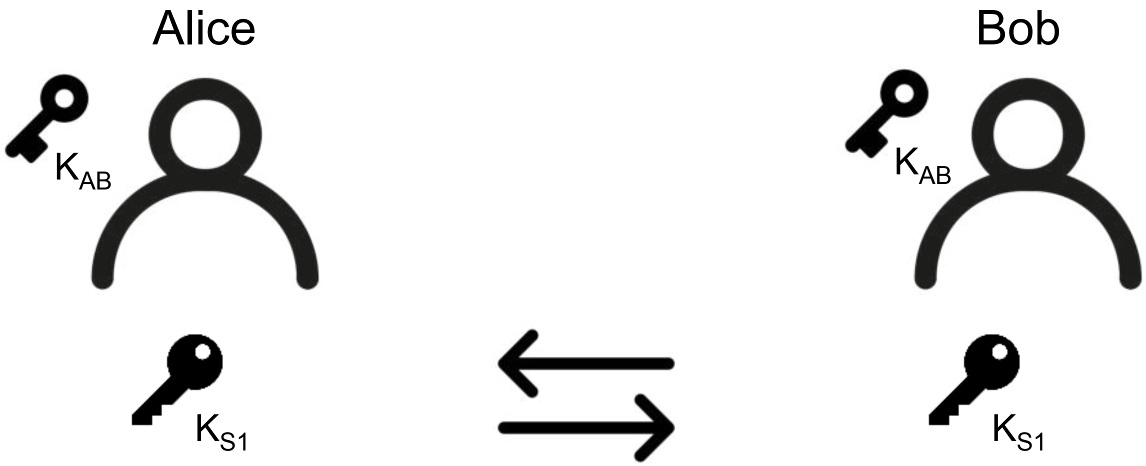
Authenticated Key Exchange - AKE



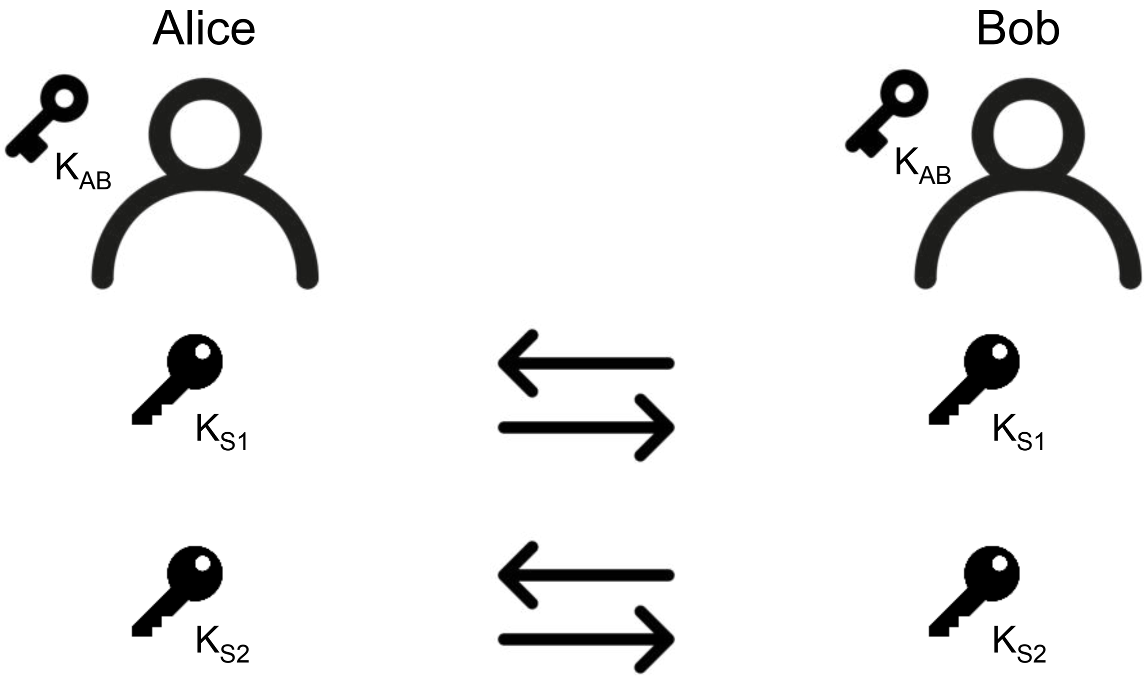
Forward Security

Forward Security

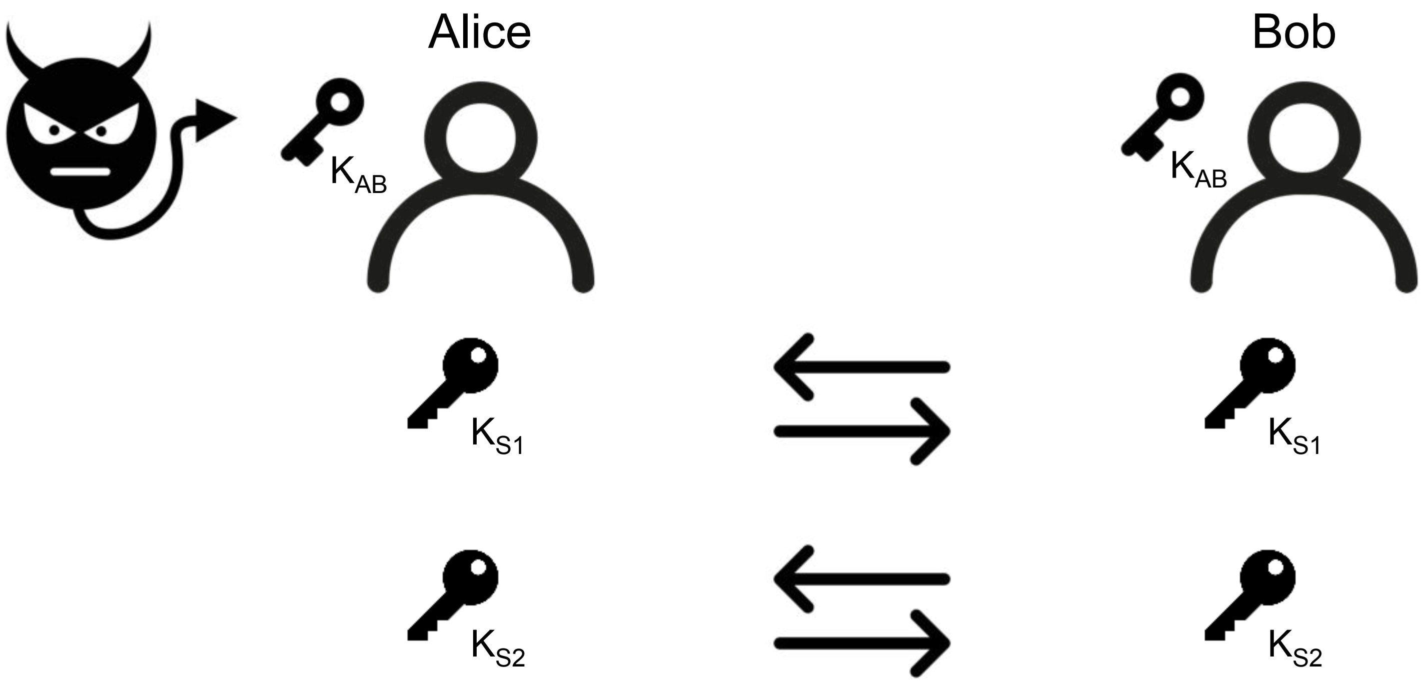
Forward Security



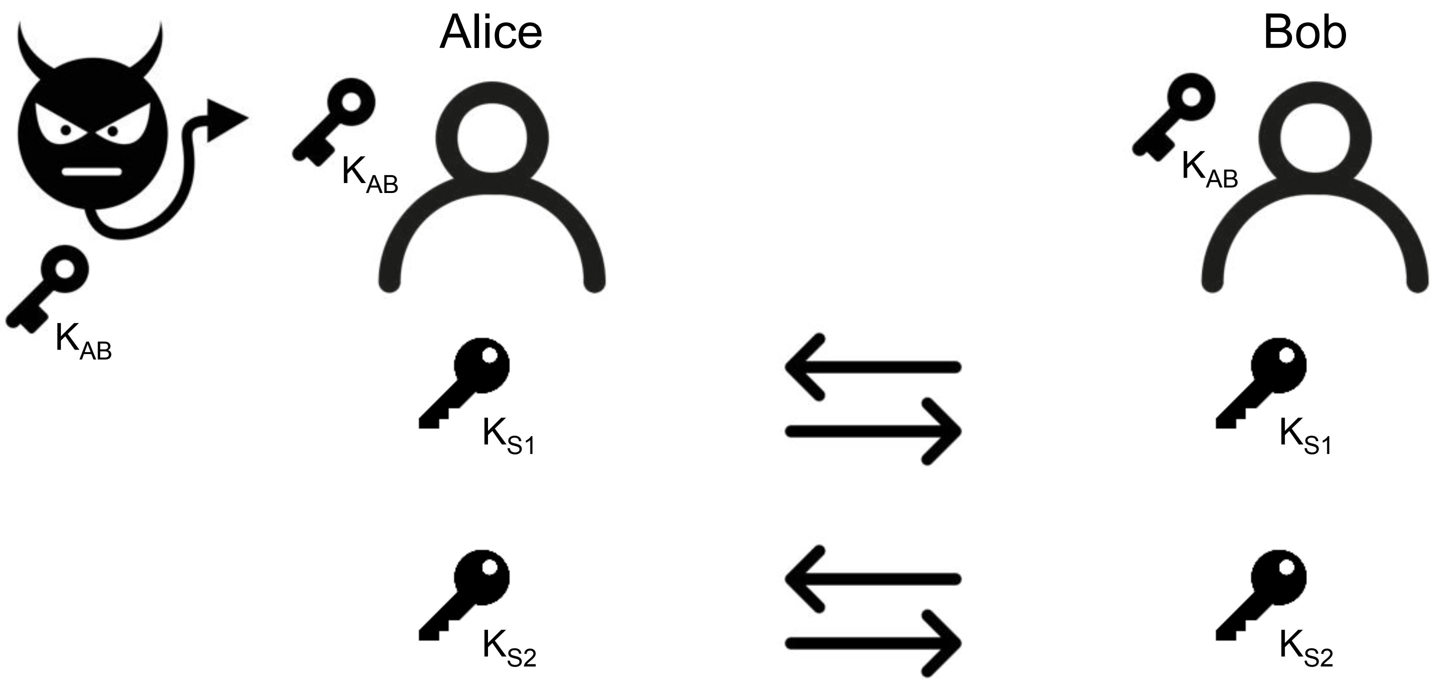
Forward Security



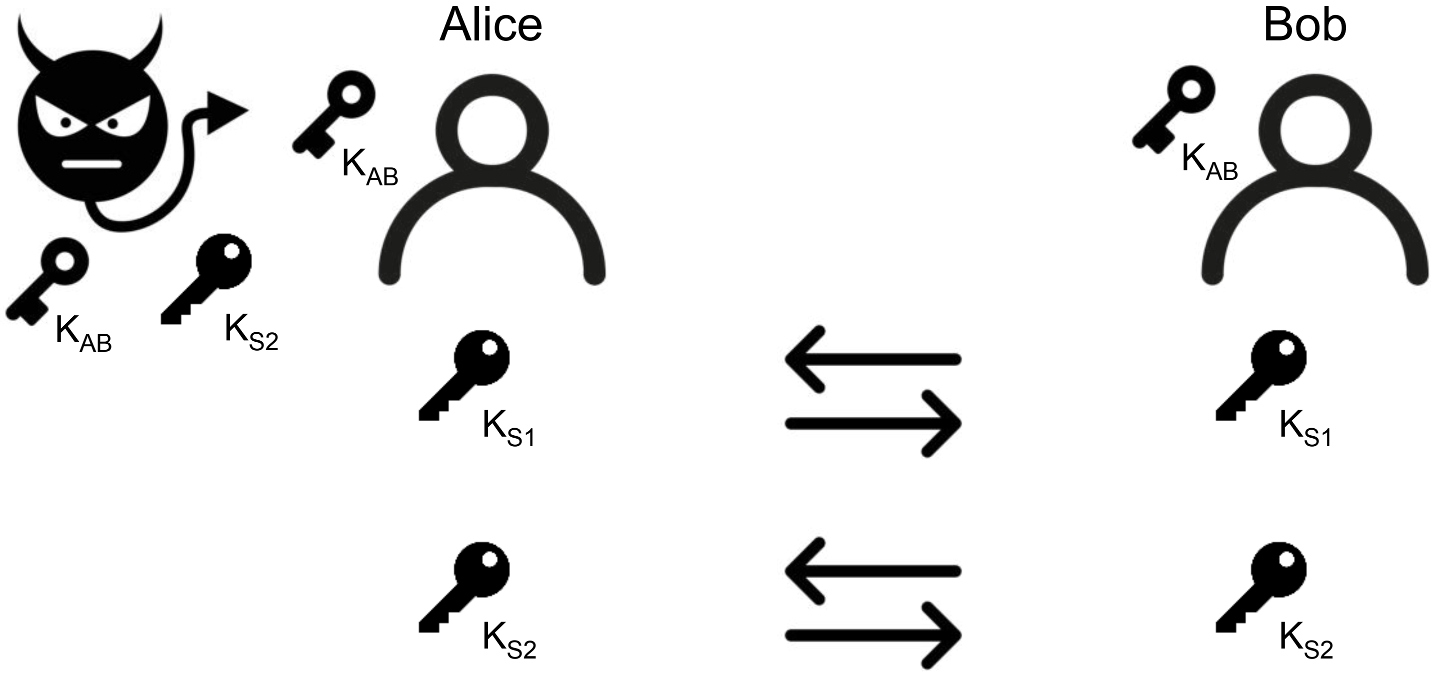
Forward Security



Forward Security

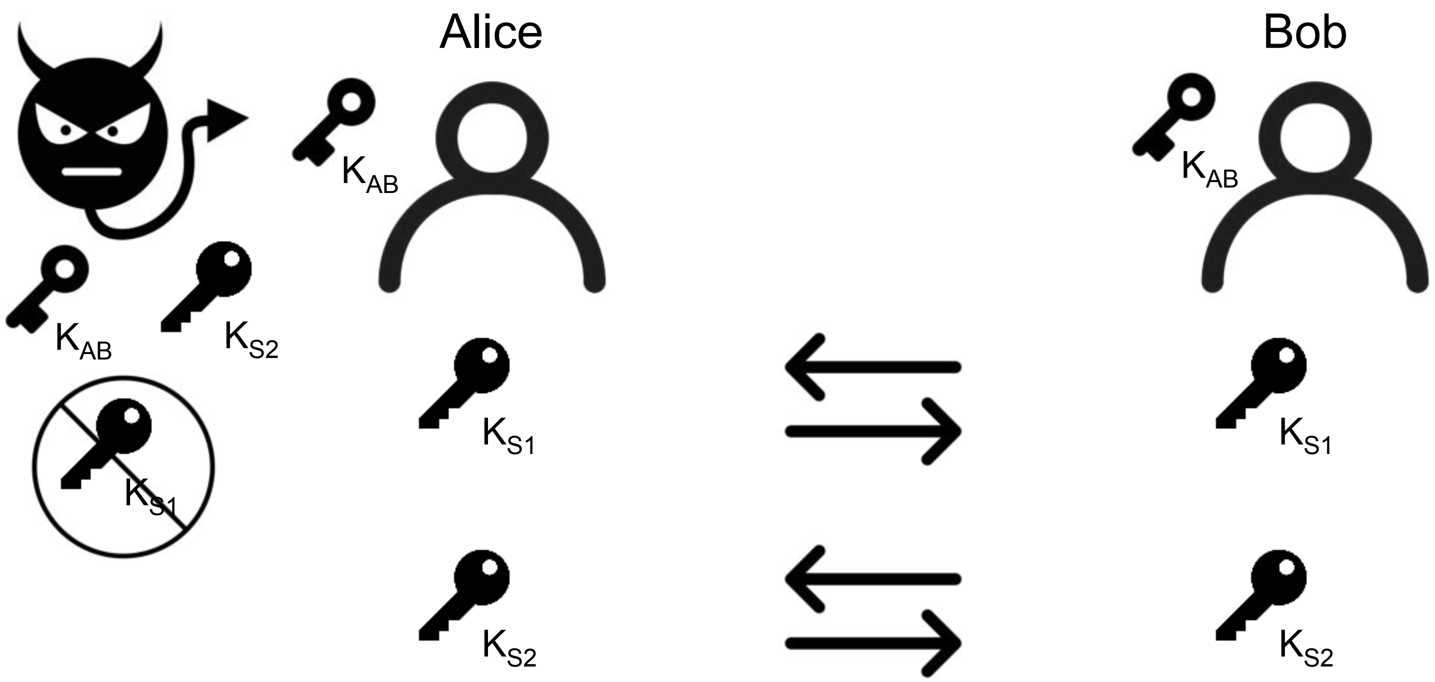


Forward Security



NTNU

Forward Security



Achieving Forward Security

Achieving Forward Security

- Evolve keys to obtain forward security

Achieving Forward Security

- Evolve keys to obtain forward security
- Time-based evolution [Dousti and Jalili, 2015]

Achieving Forward Security

- Evolve keys to obtain forward security
- Time-based evolution [Dousti and Jalili, 2015]
- Triggered evolution: evolve after session key derivation

Challenges

Challenges

- Synchronization - both parties needs to have evolved the same number of steps

Challenges

- Synchronization - both parties needs to have evolved the same number of steps
- Concurrent correctness – parallel sessions cause problems when one session evolves shared key material before the other session is ready

Challenges

- Synchronization - both parties needs to have evolved the same number of steps
- Concurrent correctness – parallel sessions cause problems when one session evolves shared key material before the other session is ready

Protocol	Auth.	# of Messages	Sync. Rob. Weak	Sync. Rob. Full	Conc. Corr.	Forward Security
SAKE [ACF20]	Mutual	5	✗	✗	✗	✓
SAKE-AM [ACF20]	Mutual	4	✗	✗	✗	✓

Example protocol: LP2

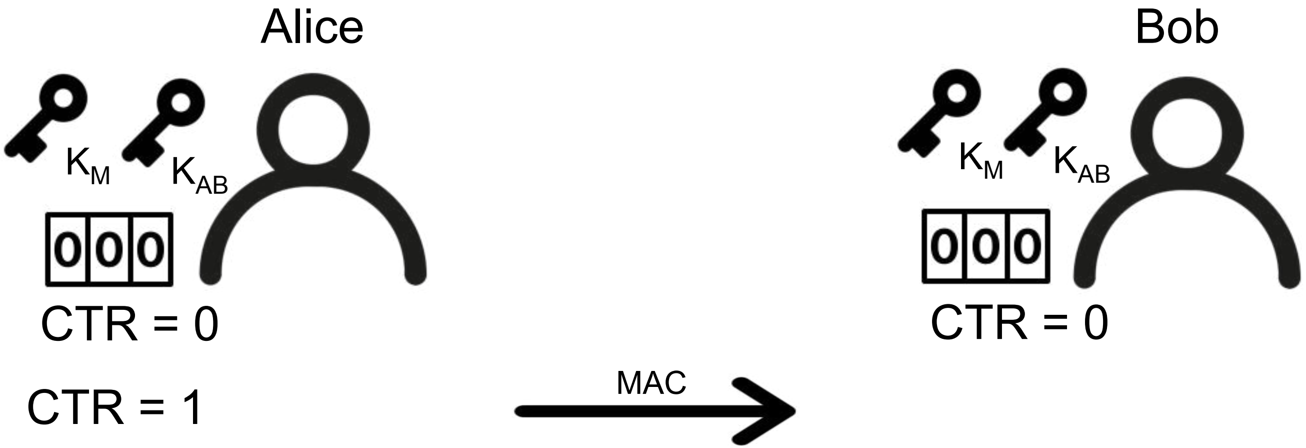
Example protocol: LP2

Example protocol: LP2

Example protocol: LP2

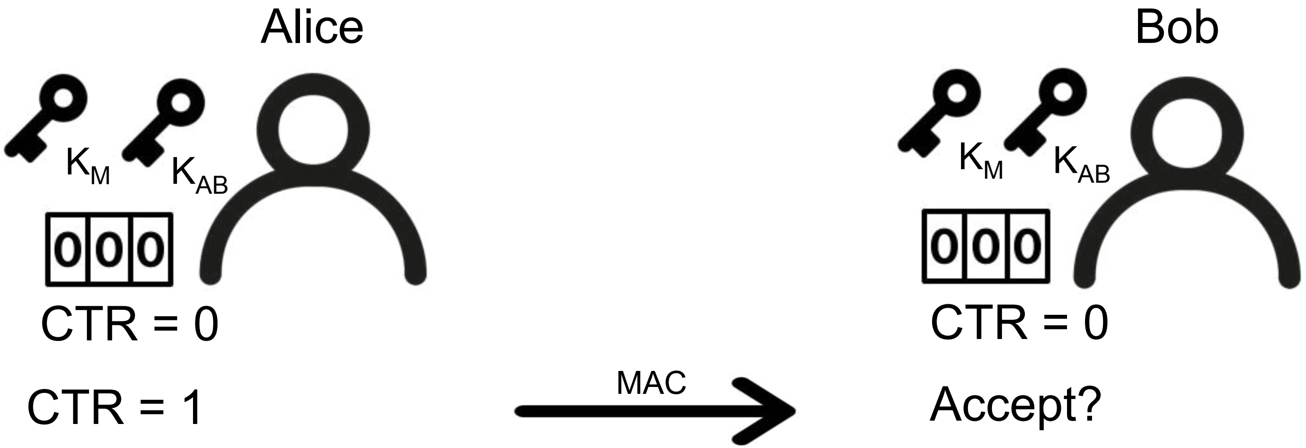
Example protocol: LP2

Example protocol: LP2

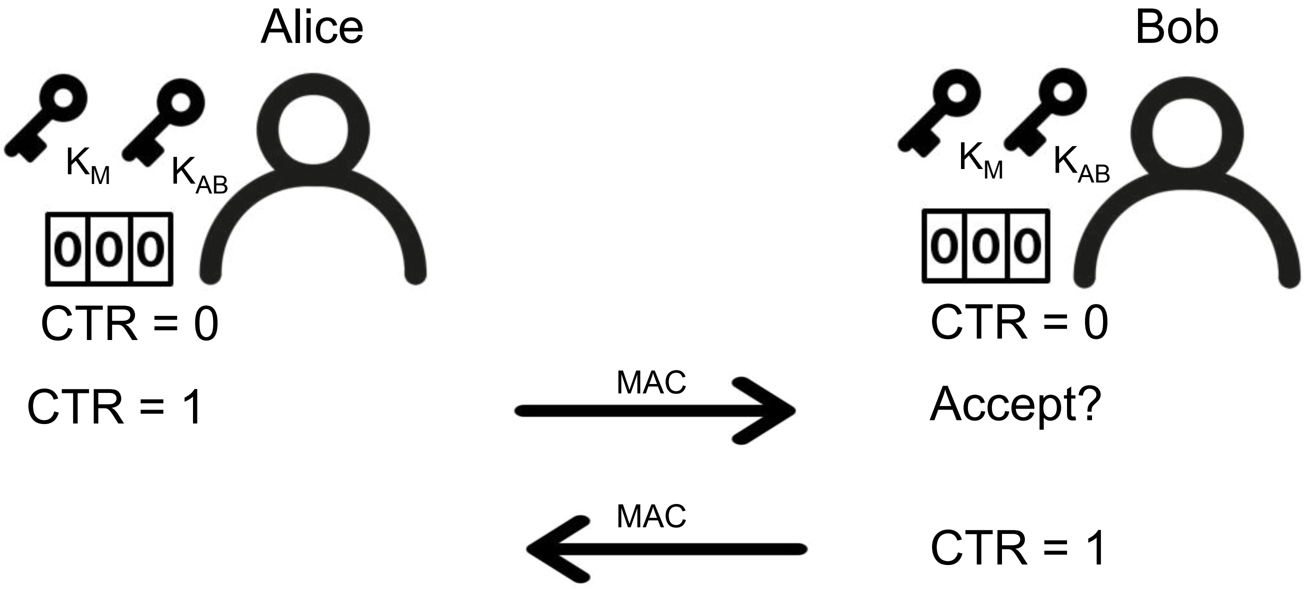


NTNU

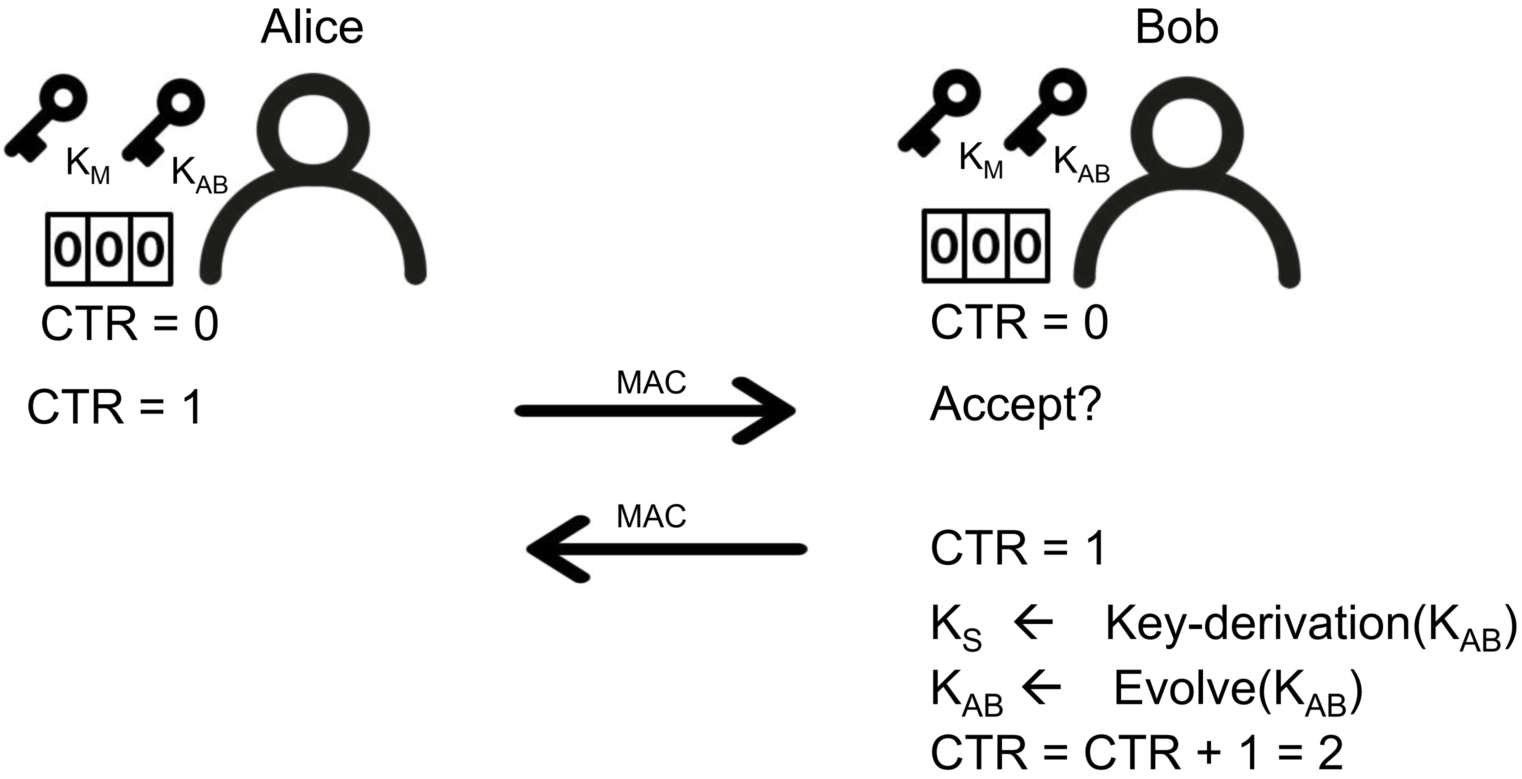
Example protocol: LP2



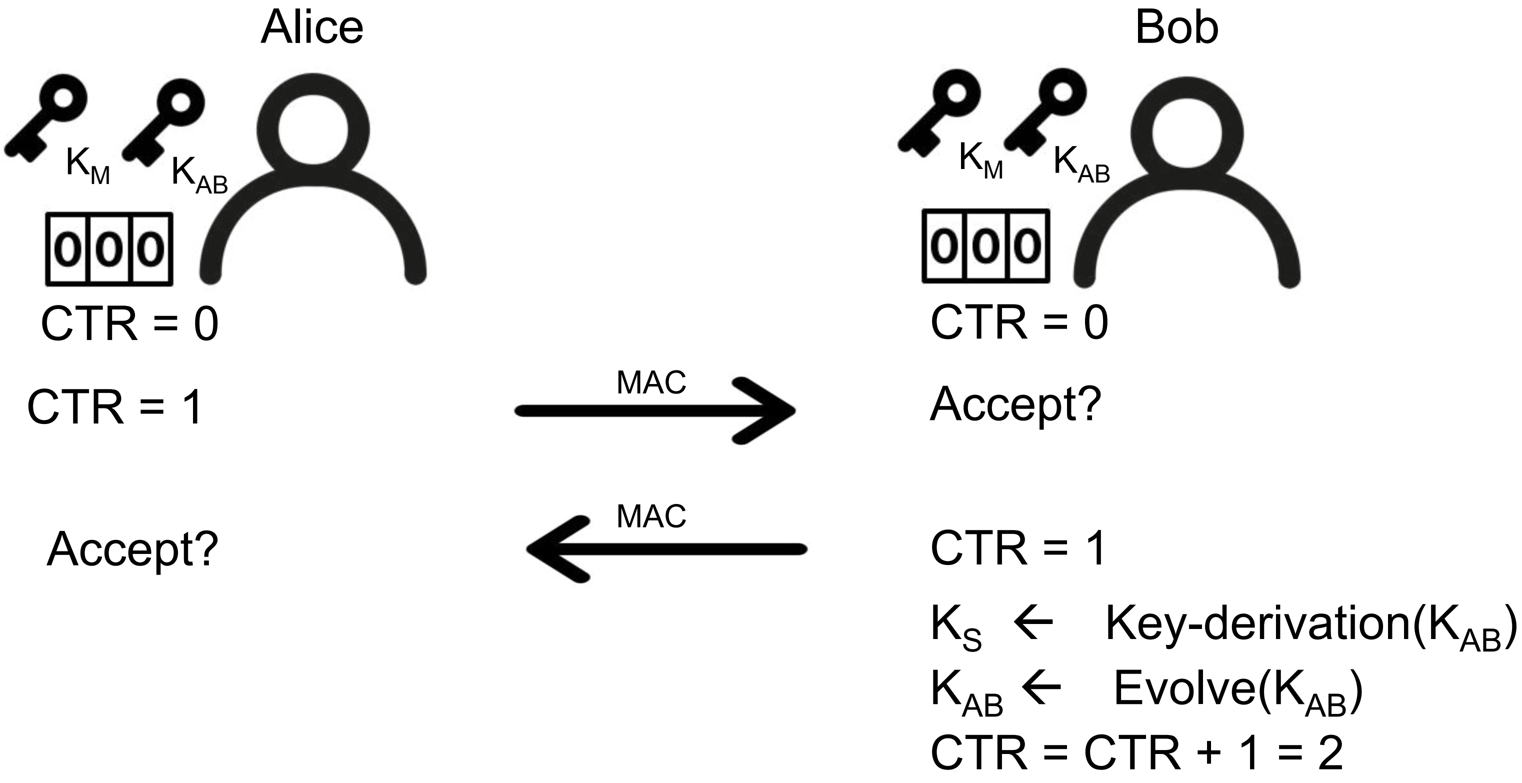
Example protocol: LP2



Example protocol: LP2

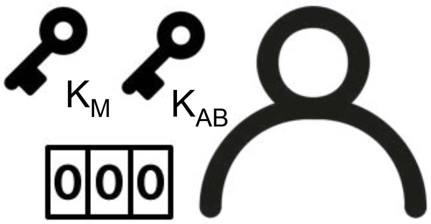


Example protocol: LP2



Example protocol: LP2

Alice



CTR = 0

CTR = 1

MAC →

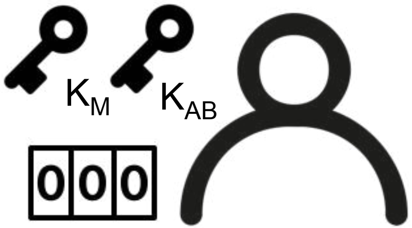
Accept?

$K_S \leftarrow$ Key-derivation(K_{AB})

$K_{AB} \leftarrow$ Evolve(K_{AB})

CTR = CTR + 1 = 2

Bob



CTR = 0

Accept?

← MAC

CTR = 1

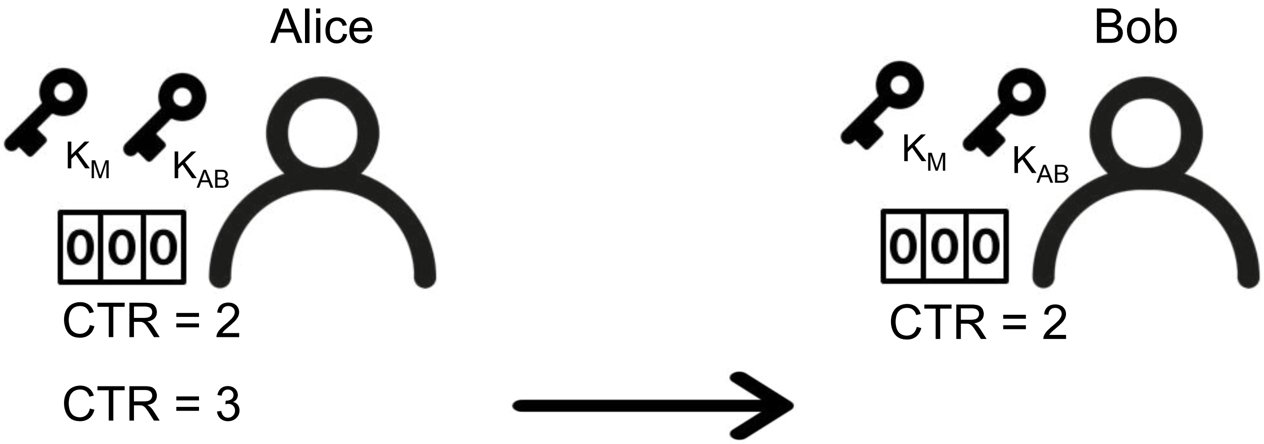
$K_S \leftarrow$ Key-derivation(K_{AB})

$K_{AB} \leftarrow$ Evolve(K_{AB})

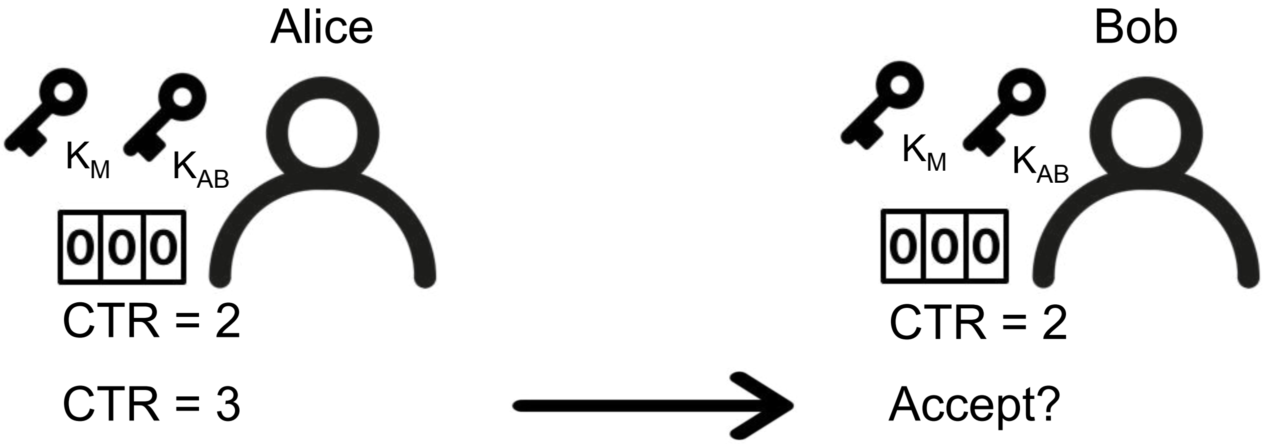
CTR = CTR + 1 = 2

Example protocol: LP2

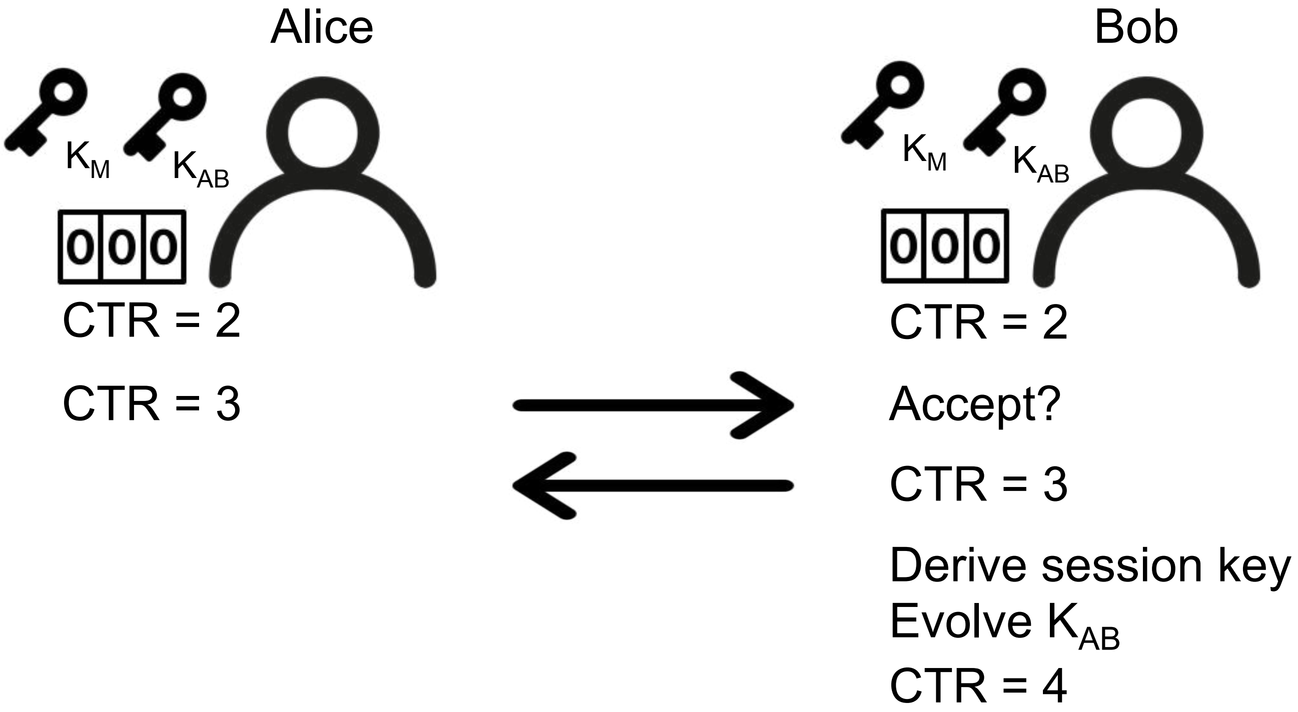
Example protocol: LP2



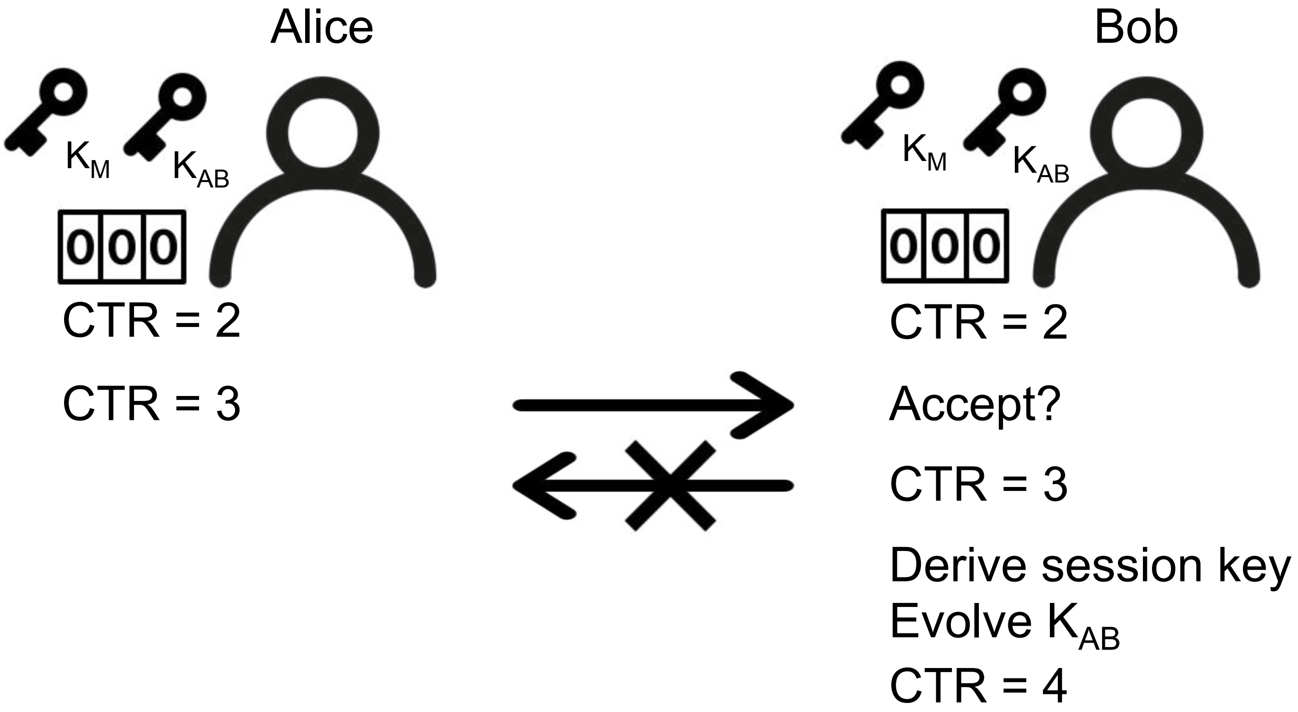
Example protocol: LP2



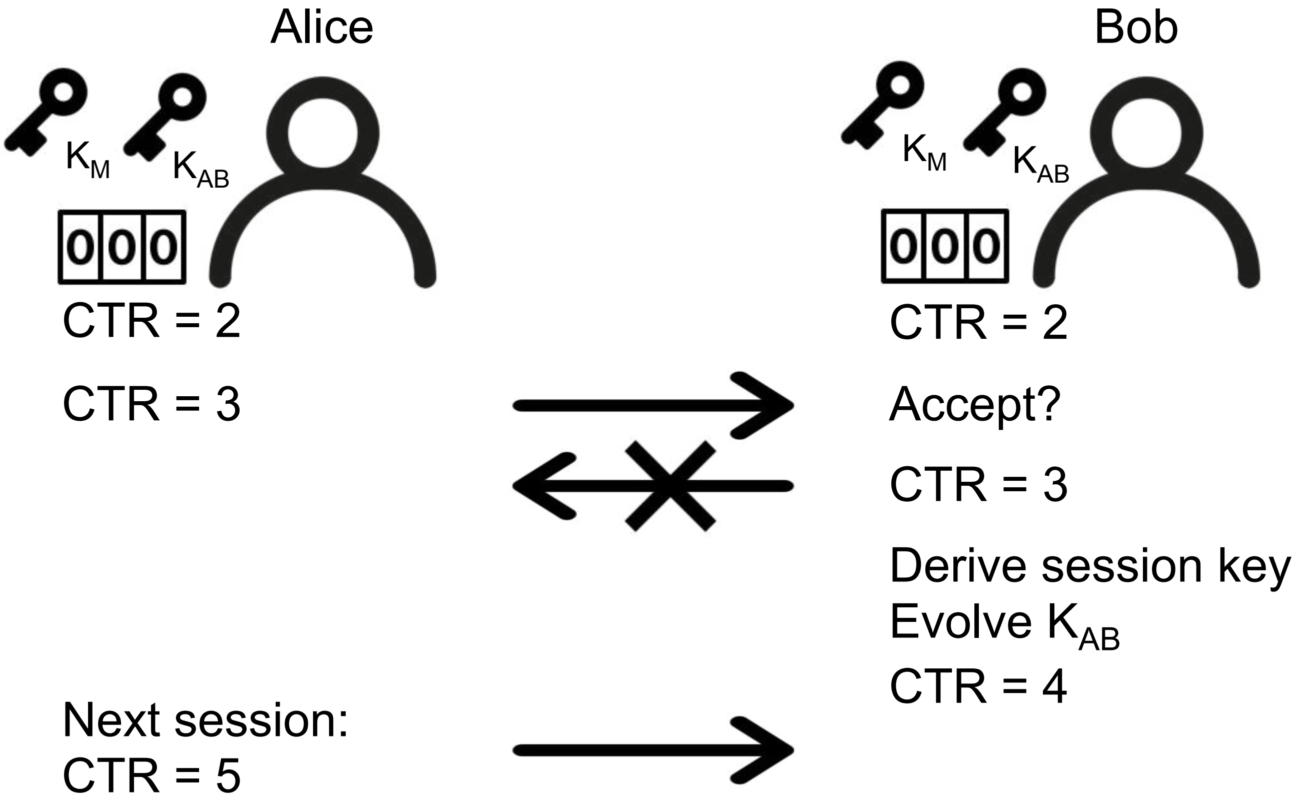
Example protocol: LP2



Example protocol: LP2



Example protocol: LP2



Linear-evolving protocols

Linear-evolving protocols

- 3 protocols – 1, 2 and 3 messages

Linear-evolving protocols

- 3 protocols – 1, 2 and 3 messages
- 1-message protocol: one-way authentication

Linear-evolving protocols

- 3 protocols – 1, 2 and 3 messages
- 1-message protocol: one-way authentication
- 3-message protocol: key confirmation, bounded gap

Linear-evolving protocols

- 3 protocols – 1, 2 and 3 messages
- 1-message protocol: one-way authentication
- 3-message protocol: key confirmation, bounded gap

Protocol	Auth.	# of Messages	Sync. Rob. Weak	Sync. Rob. Full	Conc. Corr.	Forward Security
LP1	R only	1	✓	✗	✗	✓
LP2	Mutual	2	✓	✗	✗	✓
LP3	Mutual	3	✓	✗	✗	✓

Security model

Security model

- Framework for protocol analysis

Security model

- Framework for protocol analysis
- AKE model from [Bellare Rogaway 94, Li et al 2014]

Security model

- Framework for protocol analysis
- AKE model from [Bellare Rogaway 94, Li et al 2014]
- Model lacks notion of concurrent correctness and synchronization

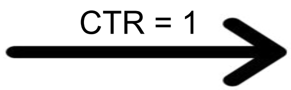
Security model

- Framework for protocol analysis
- AKE model from [Bellare Rogaway 94, Li et al 2014]
- Model lacks notion of concurrent correctness and synchronization
- Formalization of synchronization robustness – the ability to compute keys in future sessions if something goes wrong

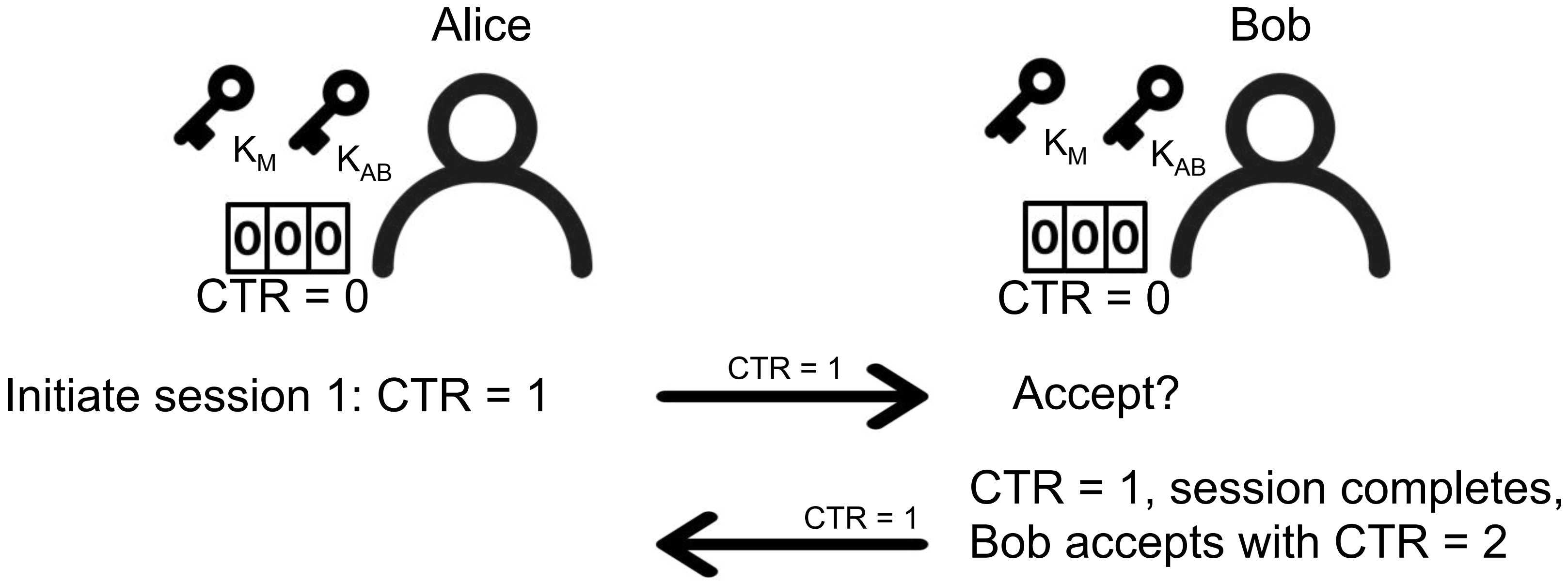
Concurrent Correctness

Concurrent Correctness

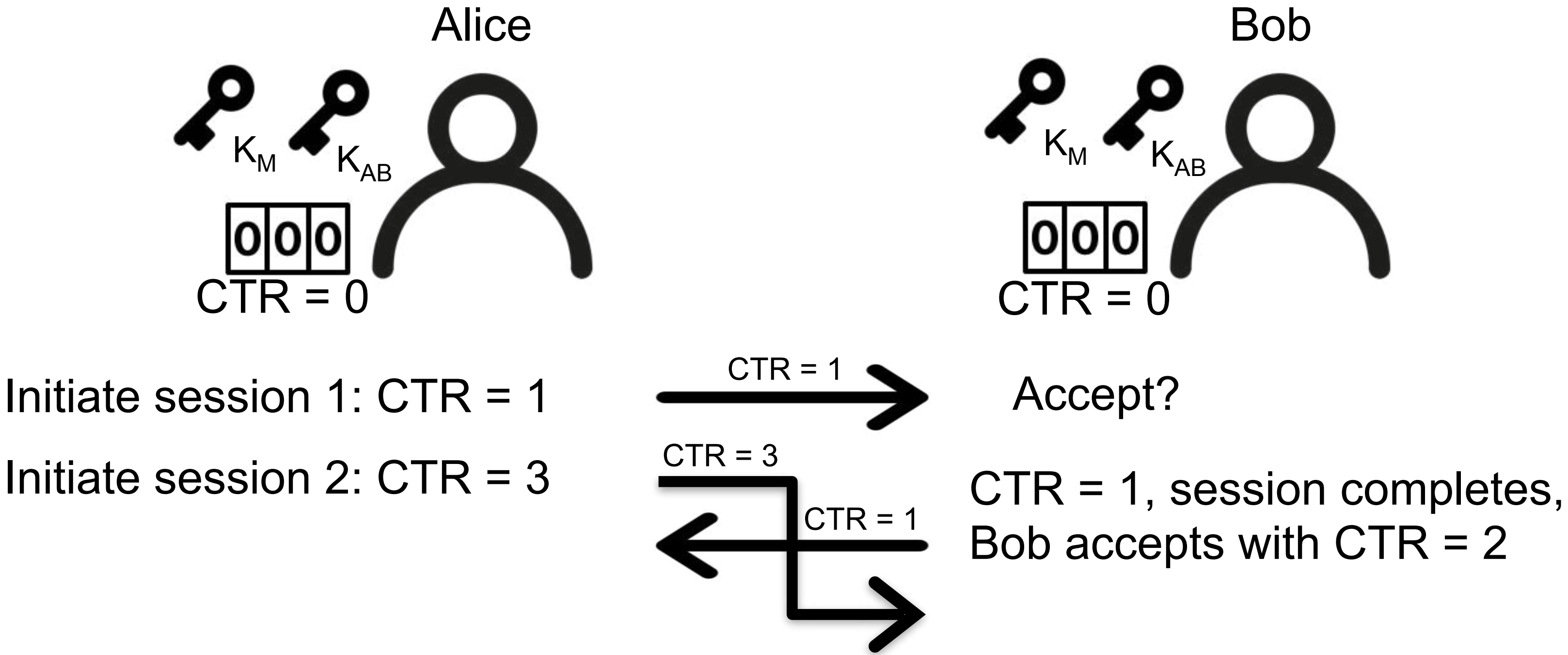
Initiate session 1: CTR = 1



Concurrent Correctness

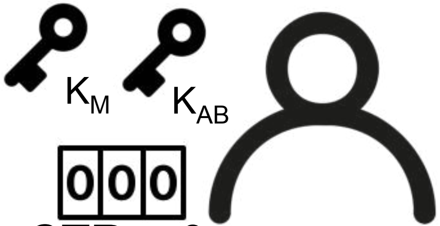


Concurrent Correctness



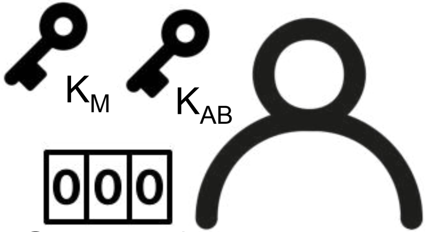
Concurrent Correctness

Alice



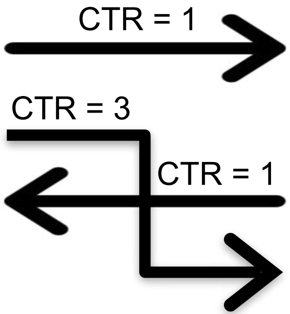
CTR = 0

Bob



CTR = 0

Initiate session 1: CTR = 1



Accept?

Initiate session 2: CTR = 3

CTR = 1, session completes,
Bob accepts with CTR = 2

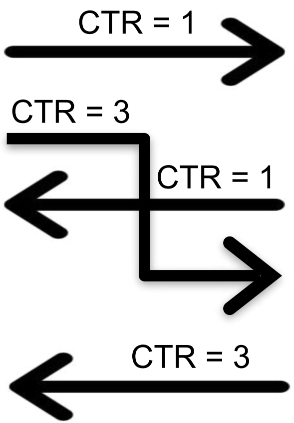
Accept?
Alice is at CTR = 3,
Aborts session 1

Concurrent Correctness

Initiate session 1: CTR = 1

Initiate session 2: CTR = 3

Accept?
Alice is at CTR = 3,
Aborts session 1



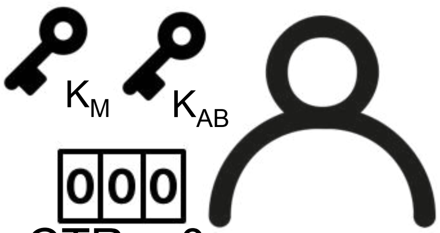
Accept?

CTR = 1, session completes,
Bob accepts with CTR = 2

Accept?
CTR = 3, session completes,
Bob accepts with CTR = 4

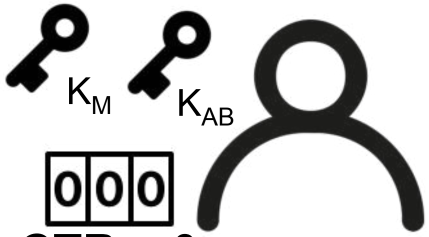
Concurrent Correctness

Alice



CTR = 0

Bob



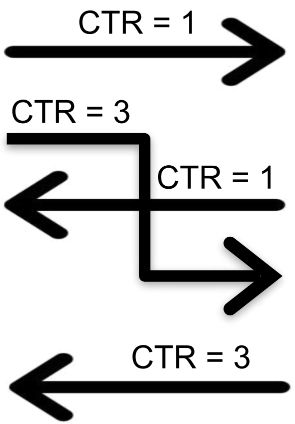
CTR = 0

Initiate session 1: CTR = 1

Initiate session 2: CTR = 3

Accept?
Alice is at CTR = 3,
Aborts session 1

Accept?
Alice accepts with CTR = 4



Accept?

CTR = 1, session completes,
Bob accepts with CTR = 2

Accept?
CTR = 3, session completes,
Bob accepts with CTR = 4

Synchronization Robustness

Synchronization Robustness

- Captures the ability of two parties succeeding in exchanging a session key in the future, no matter what has happened previously.

Synchronization Robustness

- Captures the ability of two parties succeeding in exchanging a session key in the future, no matter what has happened previously.
- If the parties get out of sync, we need to be able to resynchronize.

Synchronization Robustness

- Captures the ability of two parties succeeding in exchanging a session key in the future, no matter what has happened previously.
- If the parties get out of sync, we need to be able to resynchronize.
- This definition formalizes this requirement and comes in a weak and a strong flavour.

Weak Synchronization Robustness

Weak Synchronization Robustness

- Definition: Any honestly executed, uninterrupted session will succeed no matter what has happened before.

Weak Synchronization Robustness

- Definition: Any honestly executed, uninterrupted session will succeed no matter what has happened before.
  - Concurrent sessions were initiated

Weak Synchronization Robustness

- Definition: Any honestly executed, uninterrupted session will succeed no matter what has happened before.
  - Concurrent sessions were initiated
  - Messages in previous sessions were dropped, reordered or altered (so that they were not accepted)

Weak Synchronization Robustness

- Definition: Any honestly executed, uninterrupted session will succeed no matter what has happened before.
  - Concurrent sessions were initiated
  - Messages in previous sessions were dropped, reordered or altered (so that they were not accepted)
  - Parties are arbitrarily many steps out of sync

Weak Synchronization Robustness

- Definition: Any honestly executed, uninterrupted session will succeed no matter what has happened before.
  - Concurrent sessions were initiated
  - Messages in previous sessions were dropped, reordered or altered (so that they were not accepted)
  - Parties are arbitrarily many steps out of sync
  - Either way: the next session Alice and Bob are allowed to execute without any interruption will succeed

Weak Synchronization Robustness

- Definition: Any honestly executed, uninterrupted session will succeed no matter what has happened before.
  - Concurrent sessions were initiated
  - Messages in previous sessions were dropped, reordered or altered (so that they were not accepted)
  - Parties are arbitrarily many steps out of sync
  - Either way: the next session Alice and Bob are allowed to execute without any interruption will succeed
- LP2: Allowing role reversal will make the protocol fail to meet this requirement

Full Synchronization Robustness

Full Synchronization Robustness

- Definition: Any honestly executed session will succeed, no matter what else is going on with concurrent sessions or previous sessions.

Full Synchronization Robustness

- Definition: Any honestly executed session will succeed, no matter what else is going on with concurrent sessions or previous sessions.
  - Arbitrary many concurrent sessions are allowed

Full Synchronization Robustness

- Definition: Any honestly executed session will succeed, no matter what else is going on with concurrent sessions or previous sessions.
  - Arbitrary many concurrent sessions are allowed
  - The adversary may interleave messages with concurrent sessions arbitrarily

Full Synchronization Robustness

- Definition: Any honestly executed session will succeed, no matter what else is going on with concurrent sessions or previous sessions.
  - Arbitrary many concurrent sessions are allowed
  - The adversary may interleave messages with concurrent sessions arbitrarily
  - The adversary may make any previous or concurrent sessions fail, but the one that is allowed to complete honestly will succeed

Full Synchronization Robustness

- Definition: Any honestly executed session will succeed, no matter what else is going on with concurrent sessions or previous sessions.
  - Arbitrary many concurrent sessions are allowed
  - The adversary may interleave messages with concurrent sessions arbitrarily
  - The adversary may make any previous or concurrent sessions fail, but the one that is allowed to complete honestly will succeed
- Linearly evolving protocols fail this requirement

Non-linear key evolution

Non-linear key evolution

- Need something different to achieve full synchronization robustness

Non-linear key evolution

- Need something different to achieve full synchronization robustness
- Use puncturable pseudorandom functions [Sahai Waters 2014]

Non-linear key evolution

- Need something different to achieve full synchronization robustness
- Use puncturable pseudorandom functions [Sahai Waters 2014]
- Definition: A PPRF is a PRF with an extra algorithm PUNCT(k, x) such that
  - Evaluating on a punctured value fails
  - Puncturing on an already punctured value returns the same key
  - Puncturing is commutative – the order in which you puncture values does not matter

Non-linear key evolution

- Need something different to achieve full synchronization robustness
- Use puncturable pseudorandom functions [Sahai Waters 2014]
- Definition: A PPRF is a PRF with an extra algorithm PUNCT(k, x) such that
  - Evaluating on a punctured value fails
  - Puncturing on an already punctured value returns the same key
  - Puncturing is commutative – the order in which you puncture values does not matter
- Session key is determined by evaluating on the session nonce

Non-linear key evolution

- Need something different to achieve full synchronization robustness
- Use puncturable pseudorandom functions [Sahai Waters 2014]
- Definition: A PPRF is a PRF with an extra algorithm PUNCT(k, x) such that
  - Evaluating on a punctured value fails
  - Puncturing on an already punctured value returns the same key
  - Puncturing is commutative – the order in which you puncture values does not matter
- Session key is determined by evaluating on the session nonce
- All concurrent sessions can succeed: puncturing only affects key material of that particular session

The non-linearly evolving protocols

Protocol	Auth.	# of Messages	Sync. Rob. Weak	Sync. Rob. Full	Conc. Corr.	Forward Security
PP1	R only	1	✓	✓	✓	✓
PP2	Mutual	2	✓	✓	✓	✓

All our protocols

Protocol	Auth.	# of Messages	Sync. Rob. Weak	Full	Conc. Corr.	Forward Security
SAKE [ACF20]	Mutual	5	✗	✗	✗	✓
SAKE-AM [ACF20]	Mutual	4	✗	✗	✗	✓
LP1	R only	1	✓	✗	✗	✓
LP2	Mutual	2	✓	✗	✗	✓
LP3	Mutual	3	✓	✗	✗	✓
PP1	R only	1	✓	✓	✓	✓
PP2	Mutual	2	✓	✓	✓	✓

Concluding…

Concluding…

- Symmetric cryptography: it’s more relevant than you think

Concluding…

- Symmetric cryptography: it’s more relevant than you think
- Post-Quantum ≠ Key Exchange and Signatures

Concluding…

- Symmetric cryptography: it’s more relevant than you think
- Post-Quantum ≠ Key Exchange and Signatures
- We need to rethink our *systems*, not just our protocols

Concluding…

- Symmetric cryptography: it's more relevant than you think
- Post-Quantum ≠ Key Exchange and Signatures
- We need to rethink our *systems*, not just our protocols

Concluding…

- Symmetric cryptography: it’s more relevant than you think
- Post-Quantum ≠ Key Exchange and Signatures
- We need to rethink our *systems*, not just our protocols

With regard to our work…

Concluding…

- Symmetric cryptography: it’s more relevant than you think
- Post-Quantum ≠ Key Exchange and Signatures
- We need to rethink our *systems*, not just our protocols

With regard to our work…

- Implementation efforts are underway

Concluding…

- Symmetric cryptography: it’s more relevant than you think
- Post-Quantum ≠ Key Exchange and Signatures
- We need to rethink our *systems*, not just our protocols

With regard to our work…

- Implementation efforts are underway
- No real world test data yet, but theoretical analysis promising

Concluding…

- Symmetric cryptography: it’s more relevant than you think
- Post-Quantum ≠ Key Exchange and Signatures
- We need to rethink our *systems*, not just our protocols

With regard to our work…

- Implementation efforts are underway
- No real world test data yet, but theoretical analysis promising
- Let me know if you want to get involved!

More info?

Symmetric Key Exchange with Full Forward Security and Robust Synchronization

Colin Boyd, Gareth T. Davies, Bor de Kock, Kai Gellert, Tibor Jager and Lise Millerjord

IACR ePrint 2021 / 702

bor.dekock@ntnu.no

@bordekock

NTNU

Post-Quantum Cryptography Conference

PKI Consortium

ENTRUST

HID

PQ SHIELD

Fortanix

KEYFACTOR

NOREG

QRL

THALES

d-trust.

01 COMMUNIQUE

amsterdam convention bureau

ascertia

亞洲誠信 TRUSTAsia