How to Hide MetaData in MLS-Like Secure Group Messaging: Simple, Modular, and Post-Quantum

Keitaro Hashimoto
Tokyo Tech, JP
AIST, JP

Shuichi Katsumata
AIST, JP
PQShield Ltd, UK

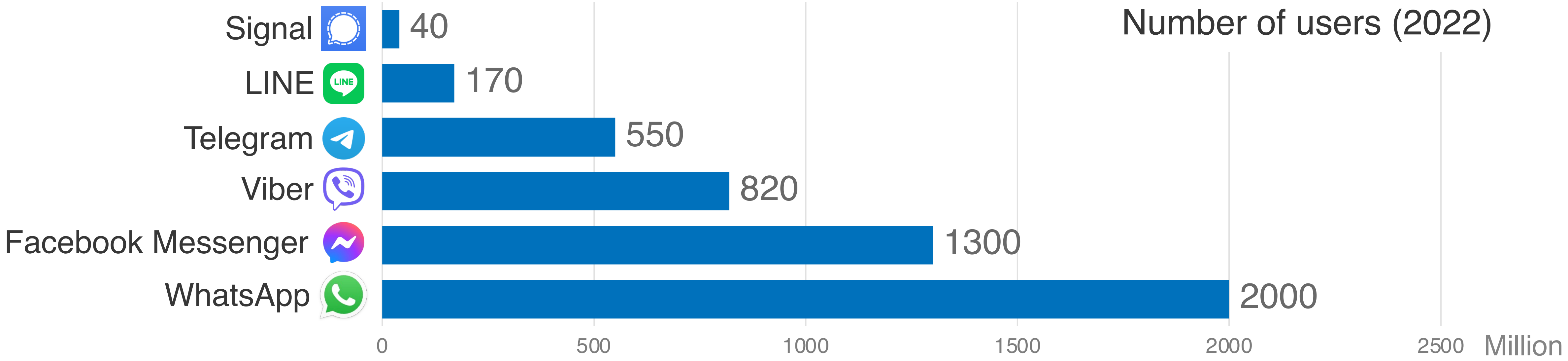
Thomas Prest
PQShield SAS, FR

ACM CCS 2022
11/8/2022@LA, USA

eprint.iacr.org/2022/1533

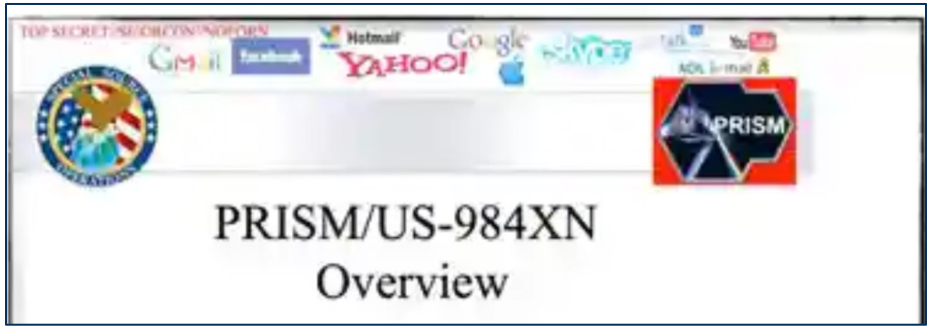
Secure group messaging (SGM)

SGM apps are used in worldwide

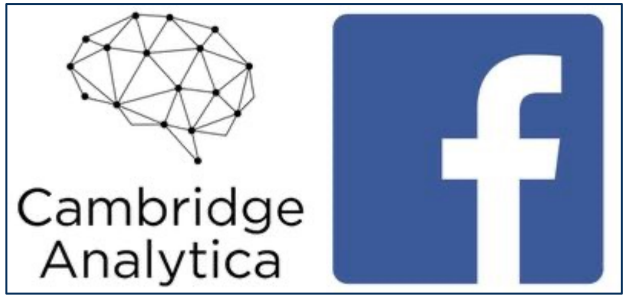


Ref: https://www.businessofapps.com/data/messaging-app-market/

Widespread data collection by governments and corporations



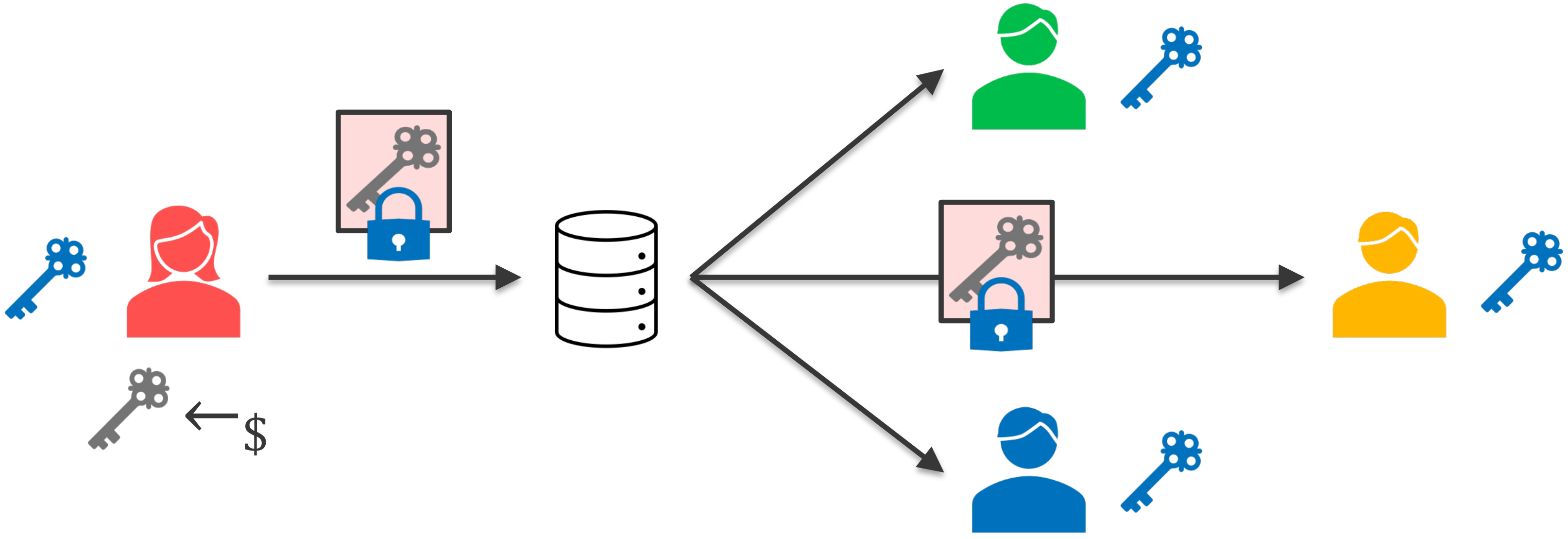
PRISM/US-984XN
Overview



Continuous Group Key Agreement (CGKA) [C:ACDT20]

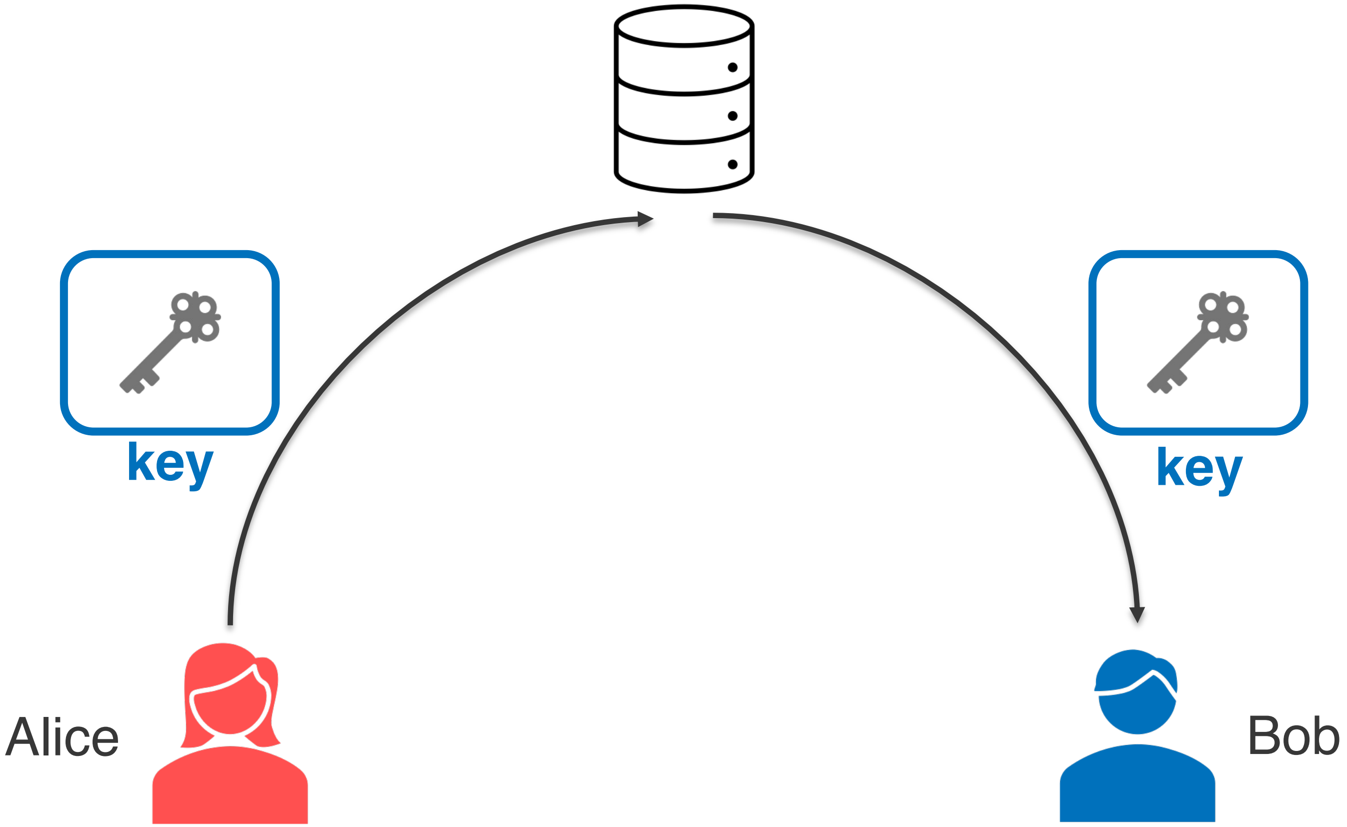
Capture the core functionality underlying SGM
e.g., TreeKEM [BBM+22,CCS:AHKM22,EC:AAC+22,…] and Chained CmPKE [CCS:**HK**P**P**W21]

- Add/Remove a party
- Update own key materials (e.g., PKE/signature keys)
- **Update group secret key** (Ratcheting)



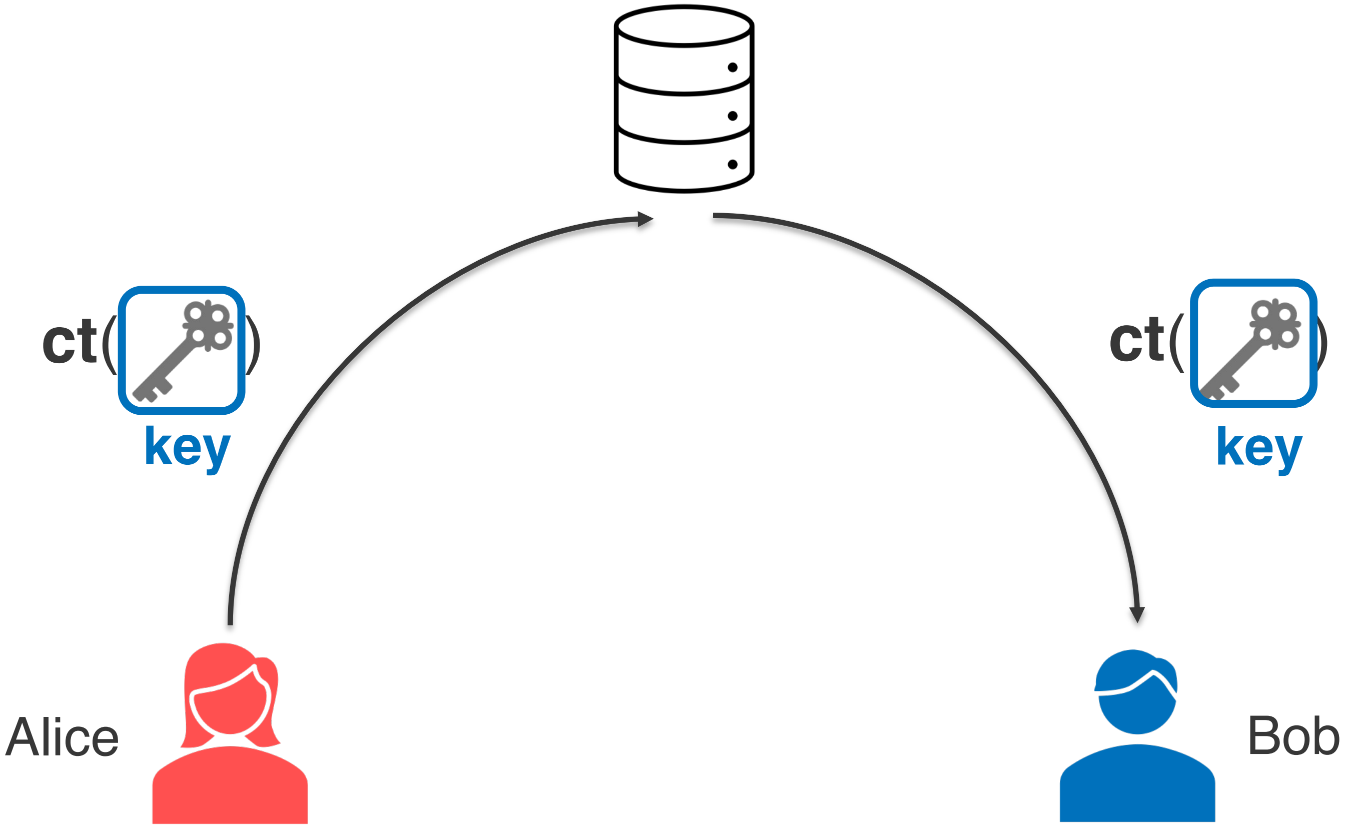
How CGKA work

- The goal is to share **secret key** among group members
  - Users communicate asynchronously through the server



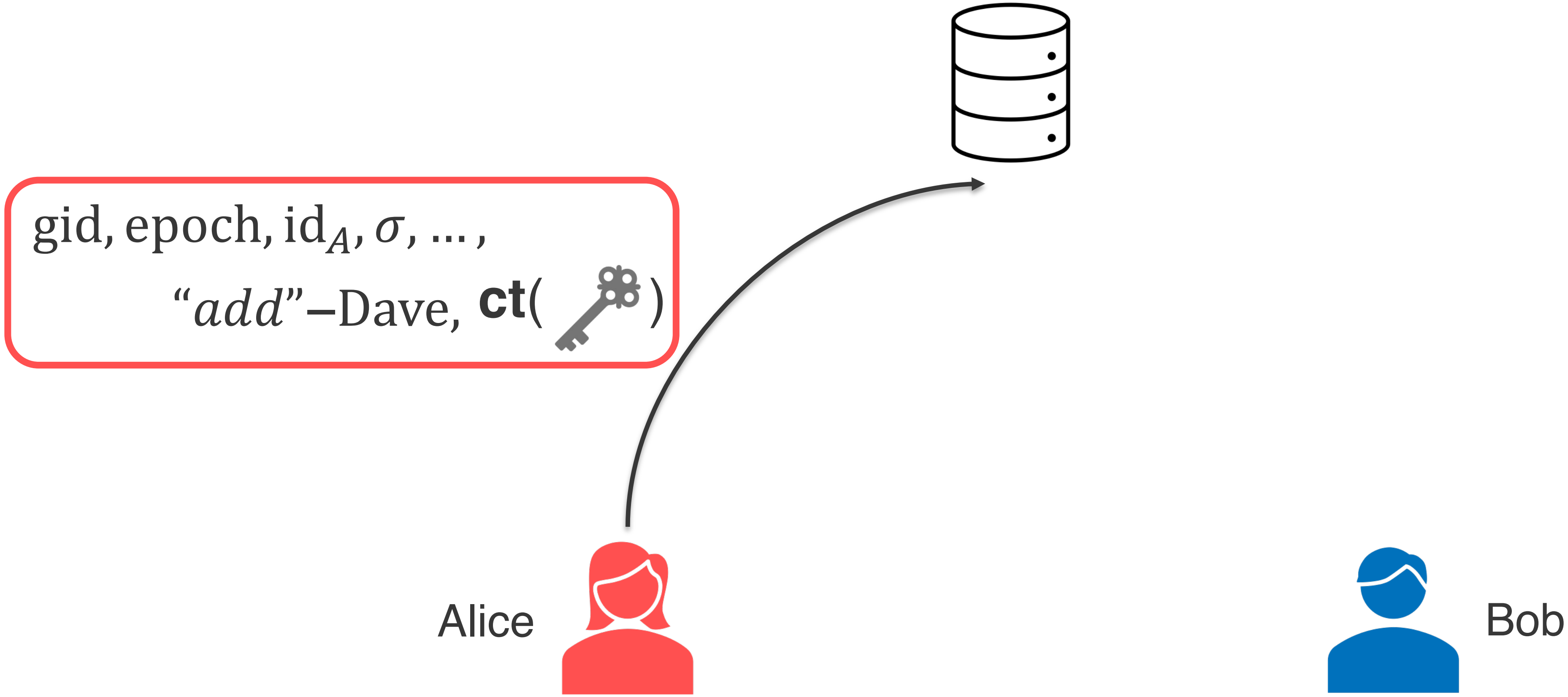
How CGKA work

- The goal is to share **<u>secret key</u>** among group members
  - Users communicate asynchronously through the server
- The secret key is protected by encryption



How CGKA work

- For delivery, the group identity and epoch are attached
- The sender’s id or the new member’s id may be included



$\mathrm{gid}, \mathrm{epoch}, \mathrm{id}_A, \sigma, \ldots,$

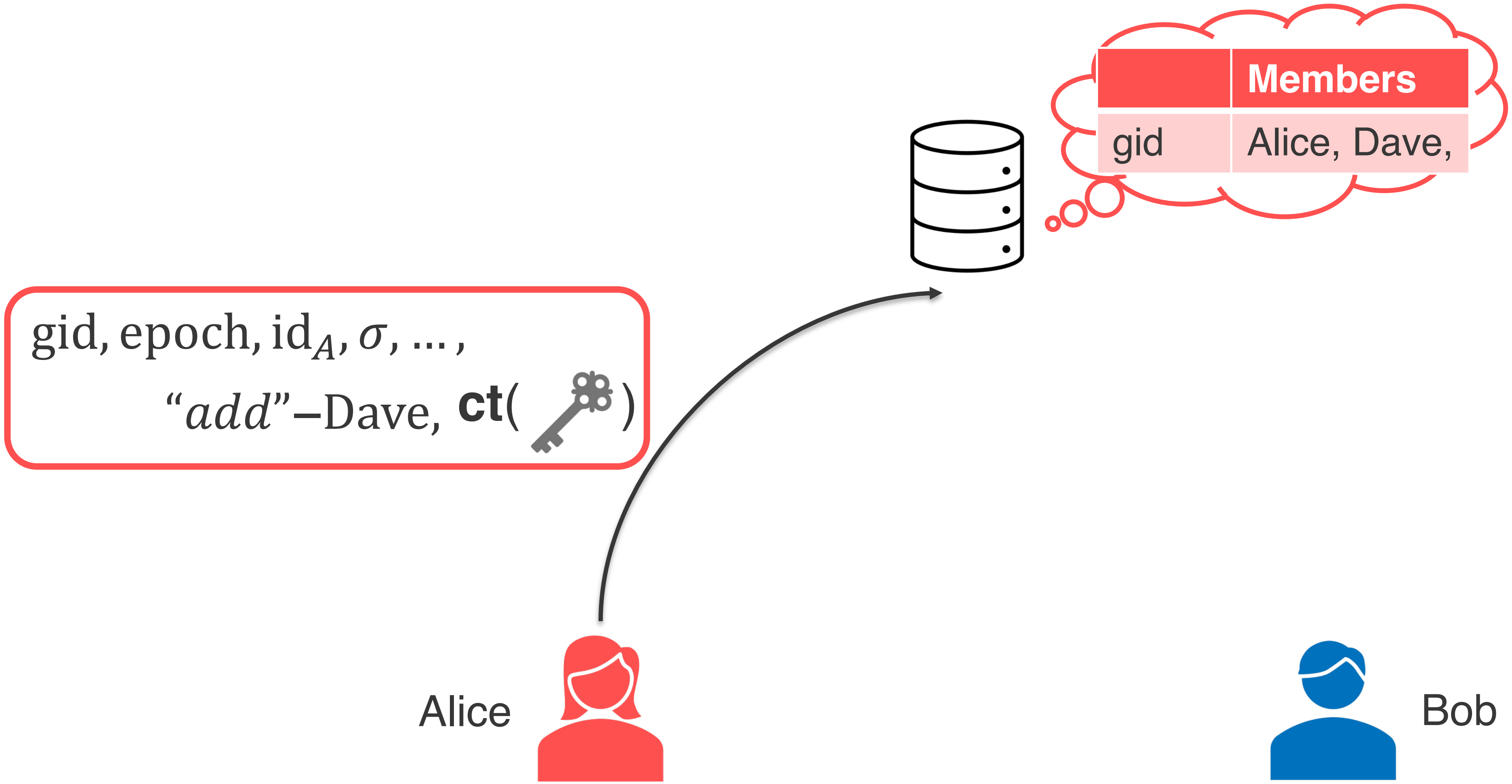
$\text{“}add\text{”–Dave},$ **ct**(🔑)

Alice

Bob

How CGKA work

- Sever explicitly obtains users’ info. from exchanged contents



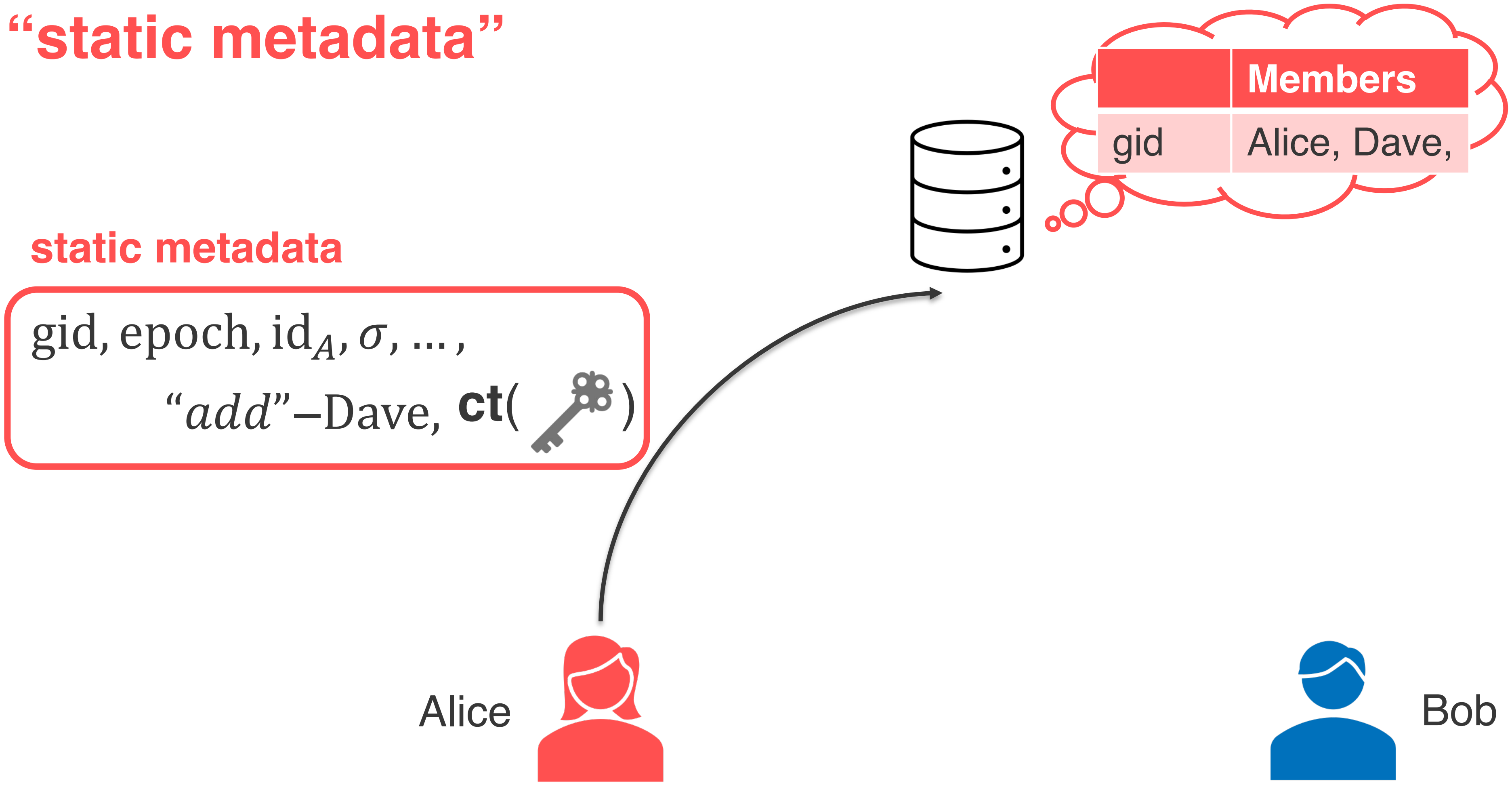
$\mathrm{gid}, \mathrm{epoch}, \mathrm{id}_A, \sigma, \ldots,$
“add”–Dave, **ct**(🔑)

	Members
gid	Alice, Dave,

How CGKA work

- Sever explicitly obtains users’ info. from exchanged contents

Call them “static metadata”



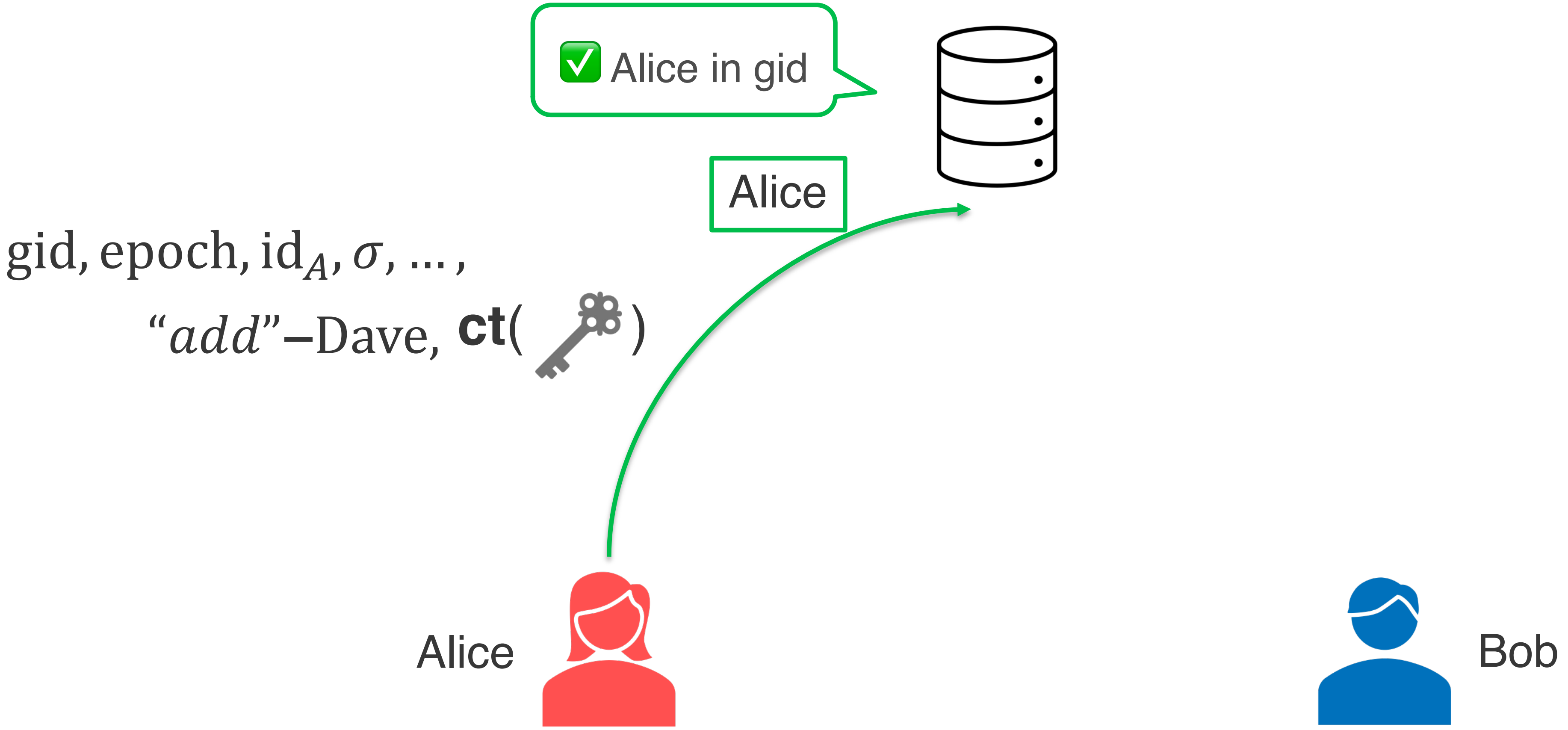
static metadata

$\mathrm{gid}, \mathrm{epoch}, \mathrm{id}_A, \sigma, \ldots,$ “add”–Dave, **ct**( )

	Members
gid	Alice, Dave,

How CGKA work

- Server authenticates users with e.g., password or certificates



✅ Alice in gid

Alice

$\mathrm{gid}, \mathrm{epoch}, \mathrm{id}_A, \sigma, \ldots,$

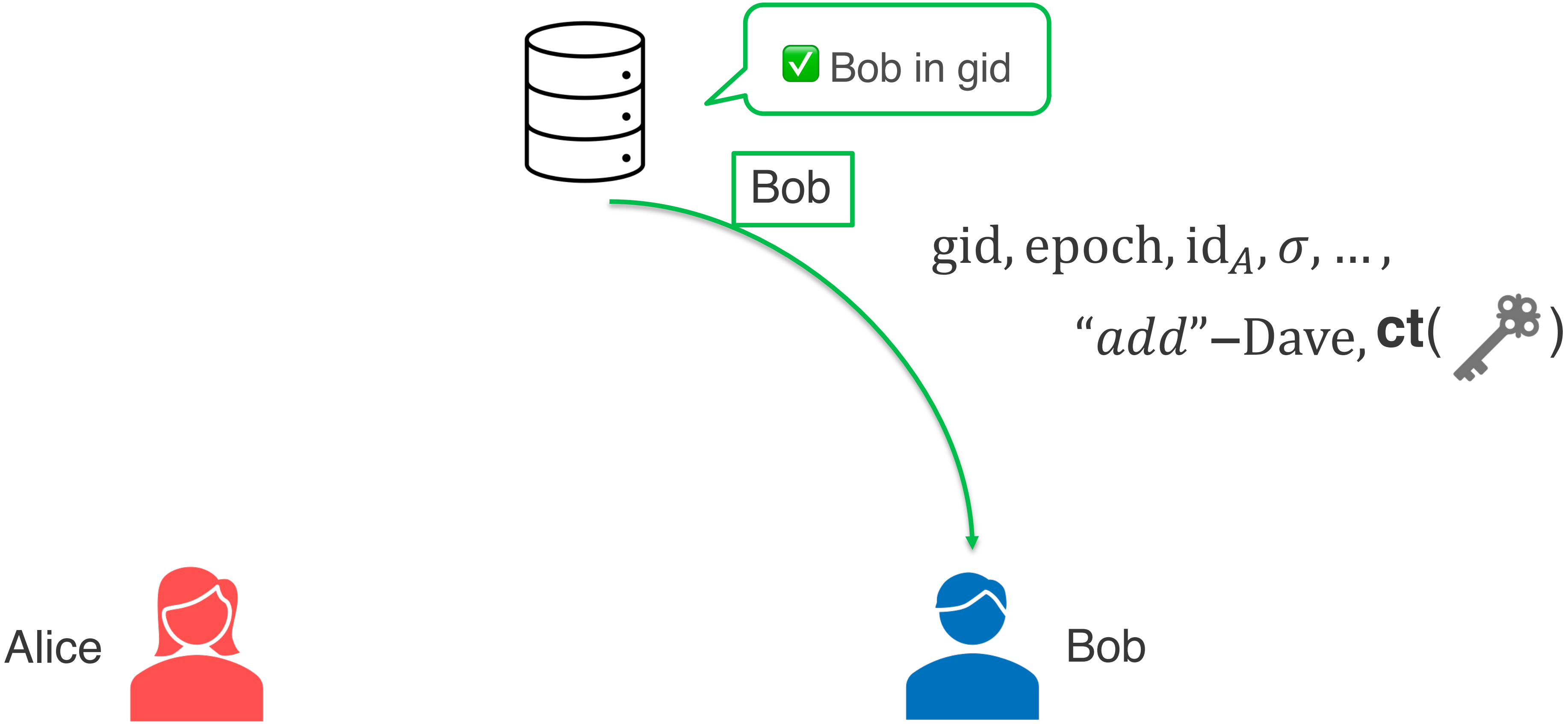
“add”–Dave, **ct**(🔑)

Alice

Bob

How CGKA work

- Server authenticates users with e.g., password or certificates



✅ Bob in gid

Bob

$\text{gid}, \text{epoch}, \text{id}_A, \sigma, \ldots,$

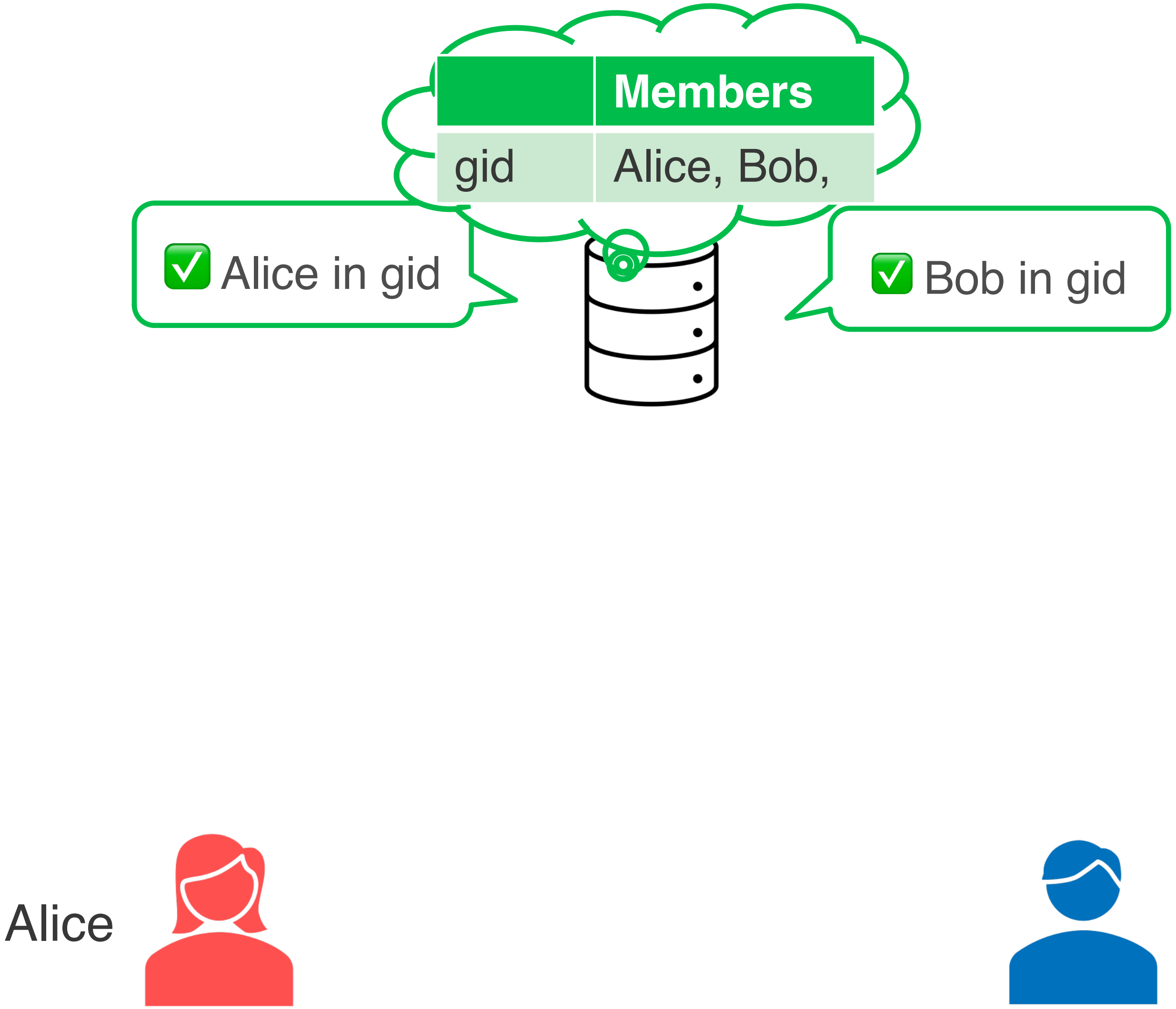
$\text{“}add\text{”–Dave},$ **ct**(🔑)

Alice

Bob

How CGKA work

- Server implicitly obtains users' information from access patterns

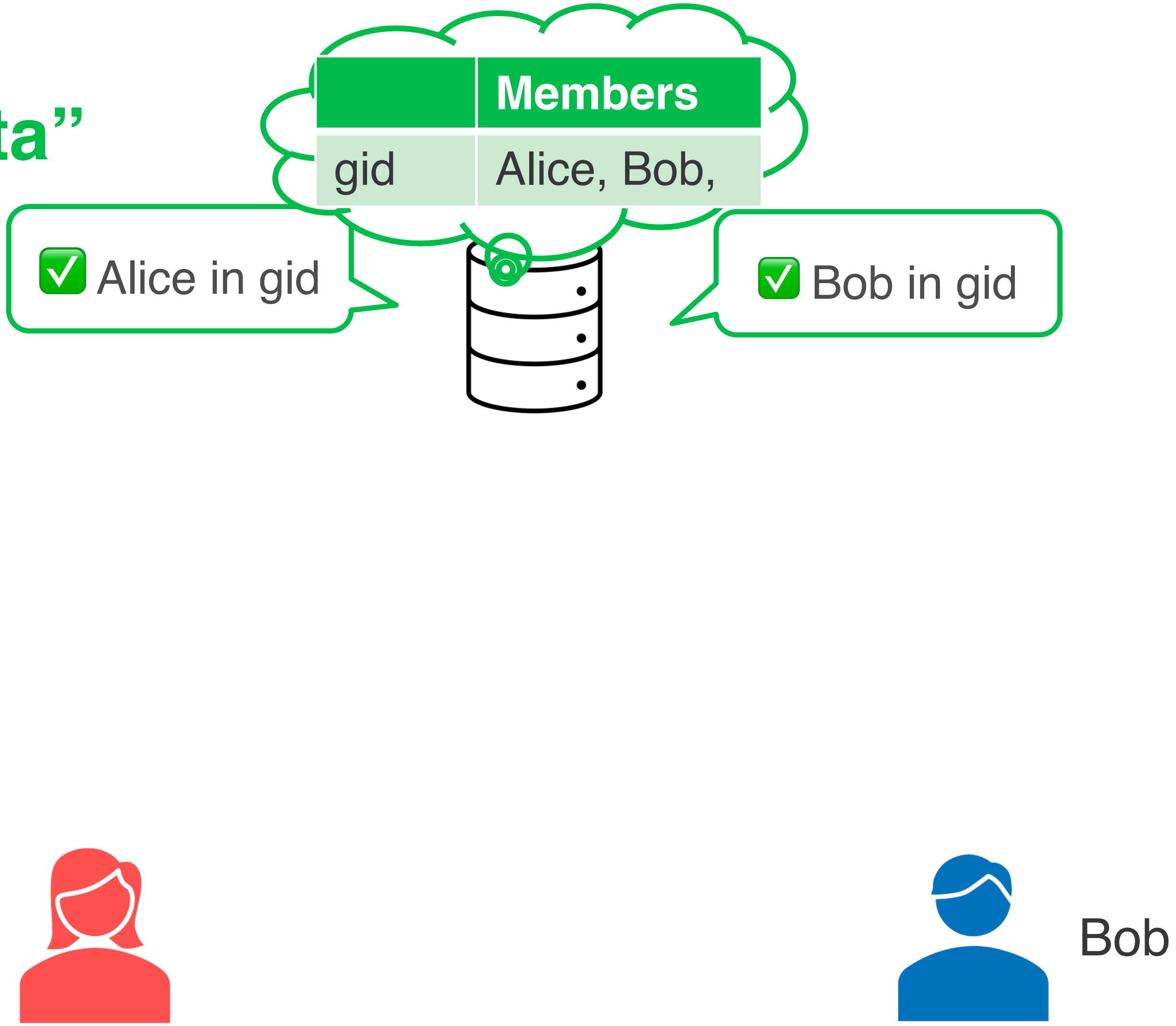


	Members
gid	Alice, Bob,

How CGKA work

- Server implicitly obtains users' information from access patterns

Call them "dynamic metadata"



	Members
gid	Alice, Bob,

✓ Alice in gid

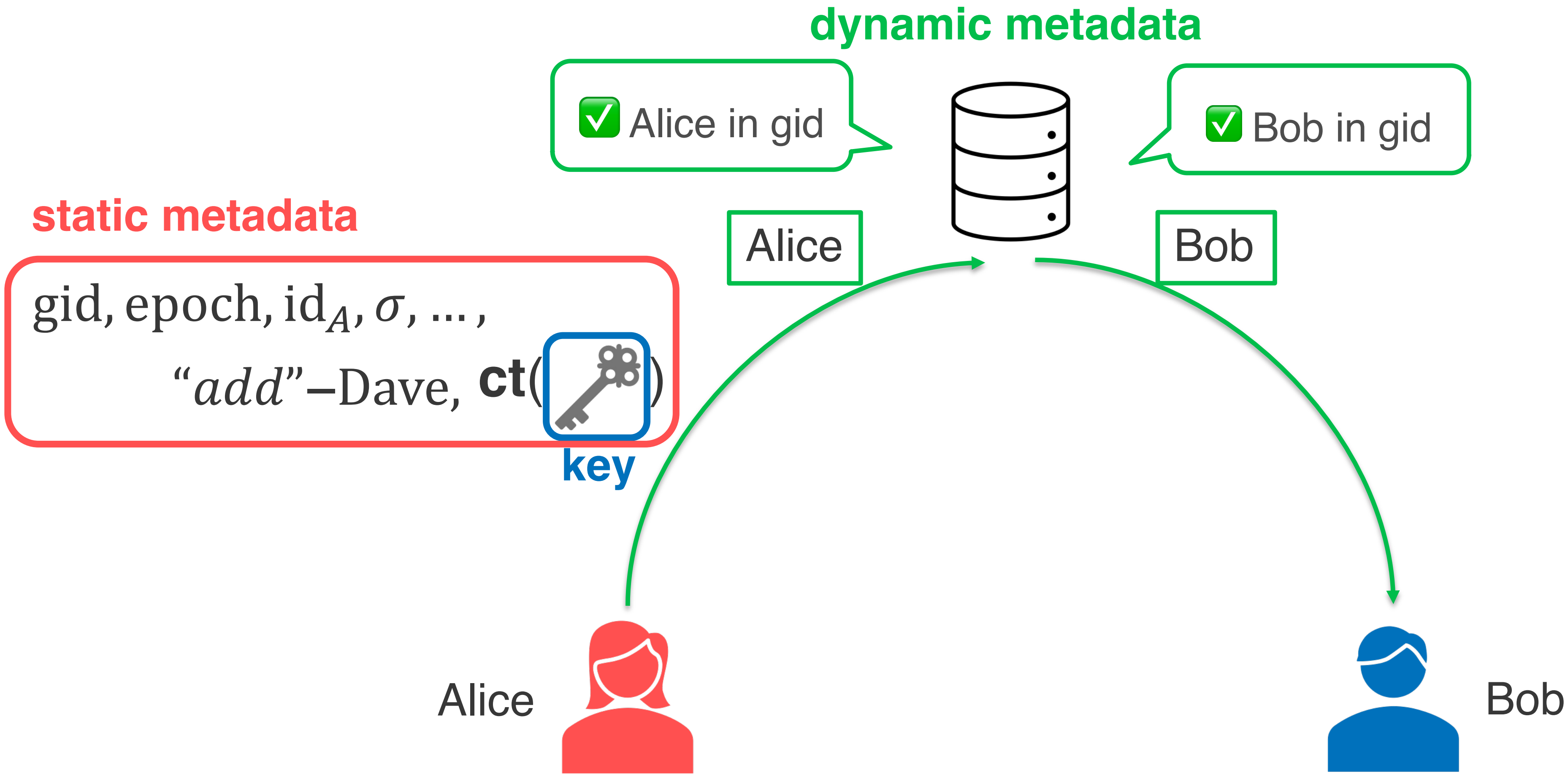
✓ Bob in gid

Alice

Bob

Summary of sensitive information in CGKA

There is three types of sensitive information:
secret key, **static metadata**, and **dynamic metadata**



dynamic metadata

✅ Alice in gid

✅ Bob in gid

Alice

Bob

static metadata

$\text{gid}, \text{epoch}, \text{id}_A, \sigma, \dots,$

$\text{“}add\text{”}\text{–Dave},$ **ct(**🔑**)**

key

Existing SGMs and protection of each data

	Secret keys	Secret keys +static metadata	Secret keys +static metadata +dynamic metadata
Signal	Vanilla Signal		Private Groups [SigPG]
Security proofs			
MLS	MLSPlaintext	MLSCiphertext	
Security proofs			

Existing SGMs and protection of each data

	Secret keys	Secret keys +static metadata	Secret keys +static metadata +dynamic metadata
Signal	Vanilla Signal		Private Groups [SigPG]
Security proofs	❌		*Only metadata [CCS:CPZ20]
MLS	MLSPlaintext	MLSCiphertext	
Security proofs			

Existing SGMs and protection of each data

	Secret keys	Secret keys +static metadata	Secret keys +static metadata +dynamic metadata
Signal	Vanilla Signal		Private Groups [SigPG]
Security proofs	✗		*Only metadata [CCS:CPZ20]
MLS	MLSPlaintext	MLSCiphertext	
Security proofs	✓ [C:ACDT20, CCS:ACDT21, C:AJM22]		

Existing SGMs and protection of each data

	Secret keys	Secret keys +static metadata	Secret keys +static metadata +dynamic metadata
Signal	Vanilla Signal		Private Groups [SigPG]
Security proofs	✗		*Only metadata [CCS:CPZ20]
MLS	MLSPlaintext	MLSCiphertext	😨
Security proofs	✓ [C:ACDT20, CCS:ACDT21, C:AJM22]	✗	

No consideration!

Our contributions

	Secret keys	Secret keys +static metadata	Secret keys +static metadata +dynamic metadata
Signal	Vanilla Signal		Private Groups [SigPG]
Security proofs	✗		*Only metadata [CCS:CPZ20]
MLS	MLSPlaintext	MLSCiphertext	✓ **Contrib. 2**
Security proofs	✓ [C:ACDT20, CCS:ACDT21, C:AJM22]	✓ **Contrib. 1***	✓ **Contrib. 3**

* Prove a variant of Chained CmPKE [HKPPW21]

Contribution 1: Formal analysis of static metadata

Propose a UC security model $\mathcal{F}_{CGKA}^{ctxt}$
capturing the security of key and static metadata

- Extend the state-of-the-art model [C:AJM22,CCS:HKPPW21]
  - Considers active adversaries and malicious insiders
  - Support selective downloading of contents

	Secret keys	Secret keys +static metadata	Secret keys +static metadata +dynamic metadata
MLS	MLSPlaintext	MLSCiphertext	**Contrib. 2**
Security proofs	[C:ACDT20, CCS:ACDT21, C:AJM22]	**Contrib. 1**	**Contrib. 3**

Contribution 1: Formal analysis of static metadata

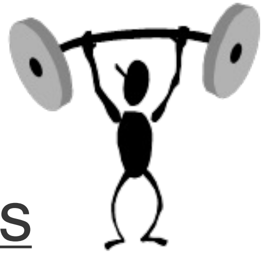
Propose a UC security model $\mathcal{F}_{CGKA}^{ctxt}$
capturing the security of key and static metadata

- Extend the state-of-the-art model [C:AJM22,CCS:HKPPW21]
  - Considers active adversaries and malicious insiders
  - Support selective downloading of contents
- Propose **Chained CmPKE$^{\text{ctxt}}$** that UC-realizes $\mathcal{F}_{CGKA}^{ctxt}$
  - Based on Chained CmPKE [CCS:HKPPW21]
  - The first provably secure **static metadata-hiding CGKA**

Contribution 1: Formal analysis of static metadata

Propose a UC security model $\mathcal{F}_{CGKA}^{ctxt}$
capturing the security of key and static metadata

- Model is parameterized by leaked metadata
  - Applicable to security analysis of other CGKAs

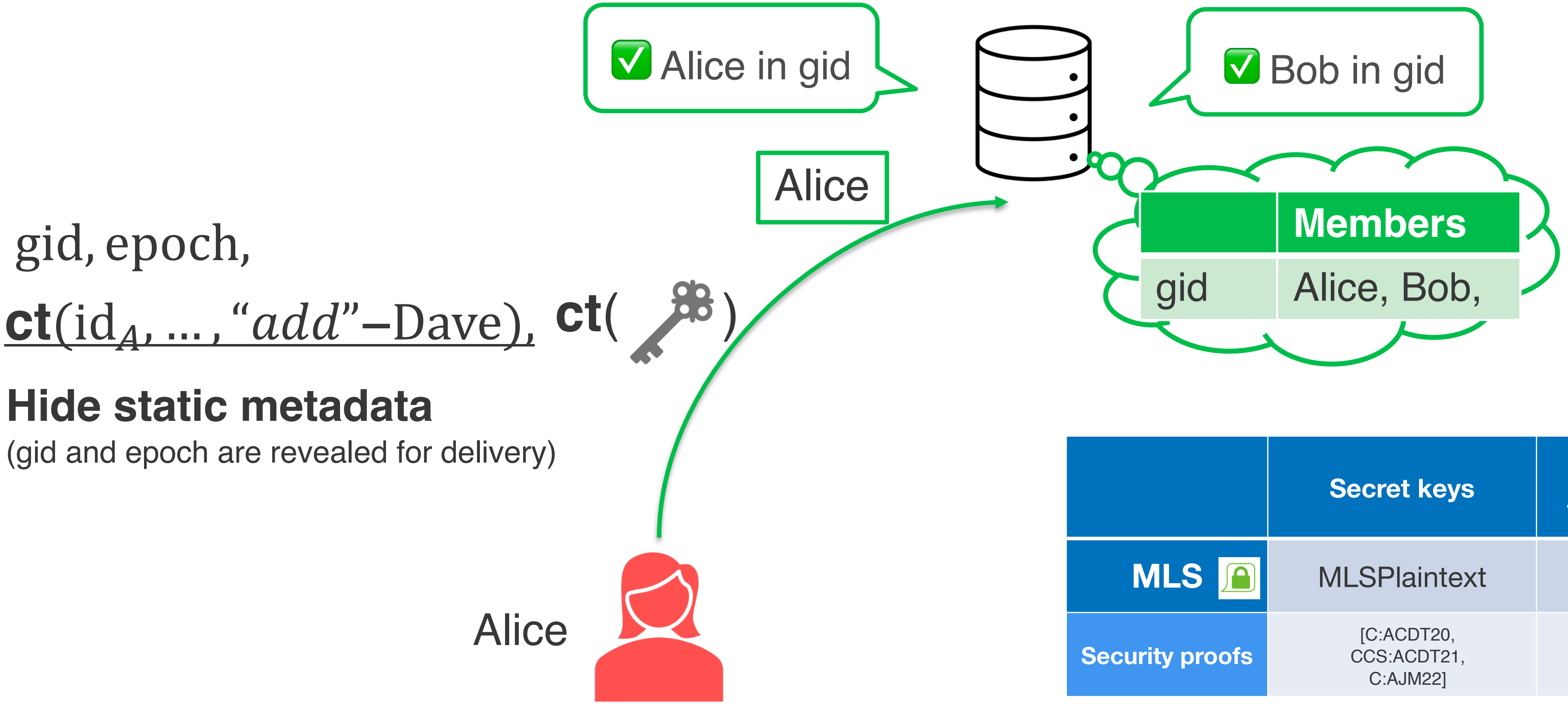


+ We analyze leaked metadata of existing CGKAs,
e.g., TreeKEM [C:AJM22], SAIK* [CCS:AHKM22], CoCoA* [EC:AAC+22]

* We analyzed the initial ePrint version.

Contribution 2: Protecting dynamic metadata

- Sevrer obtains personal information from <u>only access patterns</u>
  - Protecting static metadata alone is insufficient



gid, epoch,

$\underline{\mathbf{ct}(\mathrm{id}_A, \ldots, \text{“}add\text{”}–\mathrm{Dave})}$, **ct**(🔑)

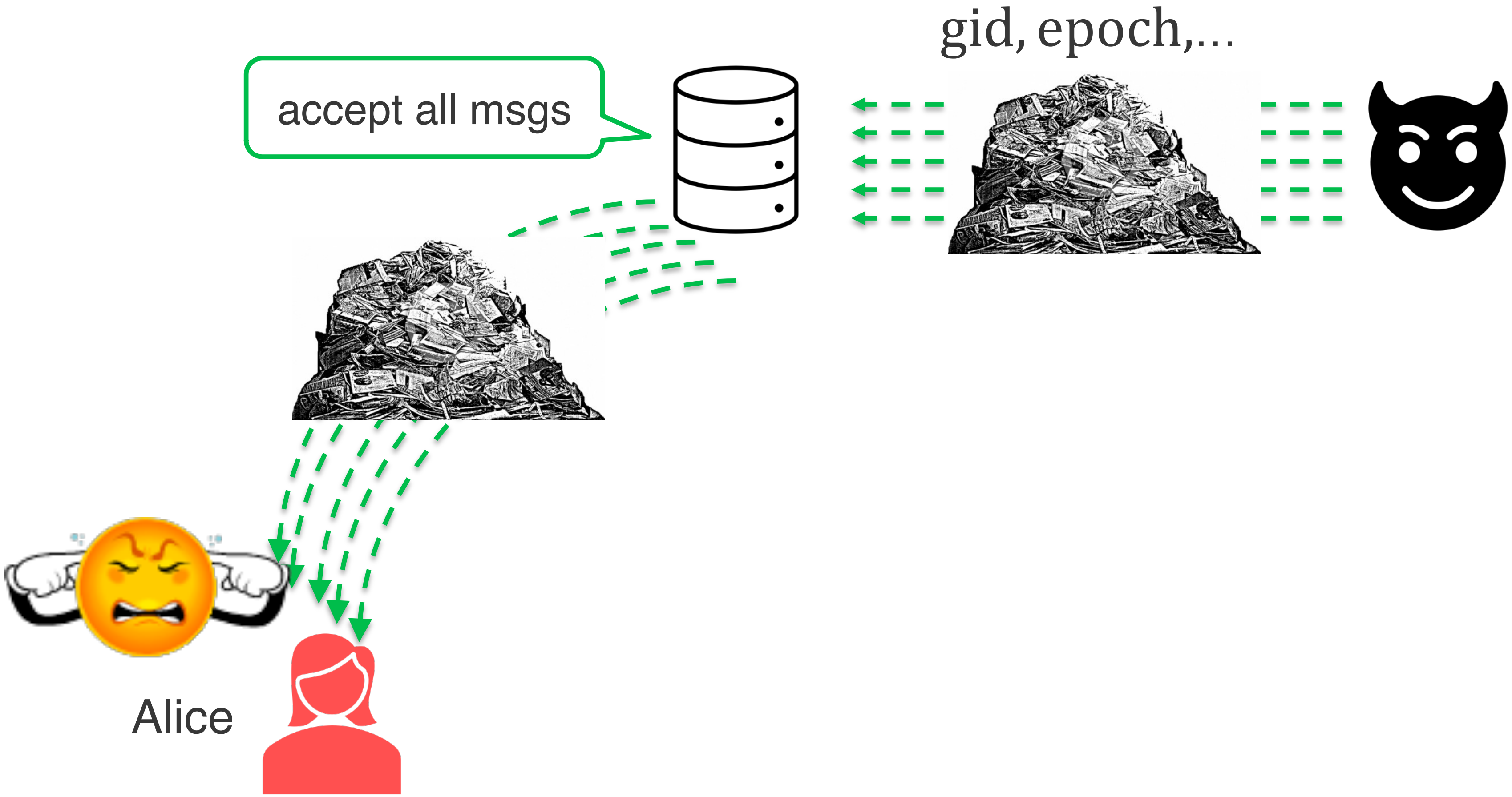
Hide static metadata

(gid and epoch are revealed for delivery)

	Secret keys	Secret keys +static metadata	Secret keys +static metadata +dynamic metadata
MLS 🔒	MLSPlaintext	MLSCiphertext	**Contrib. 2**
Security proofs	[C:ACDT20, CCS:ACDT21, C:AJM22]	**Contrib. 1**	**Contrib. 3**

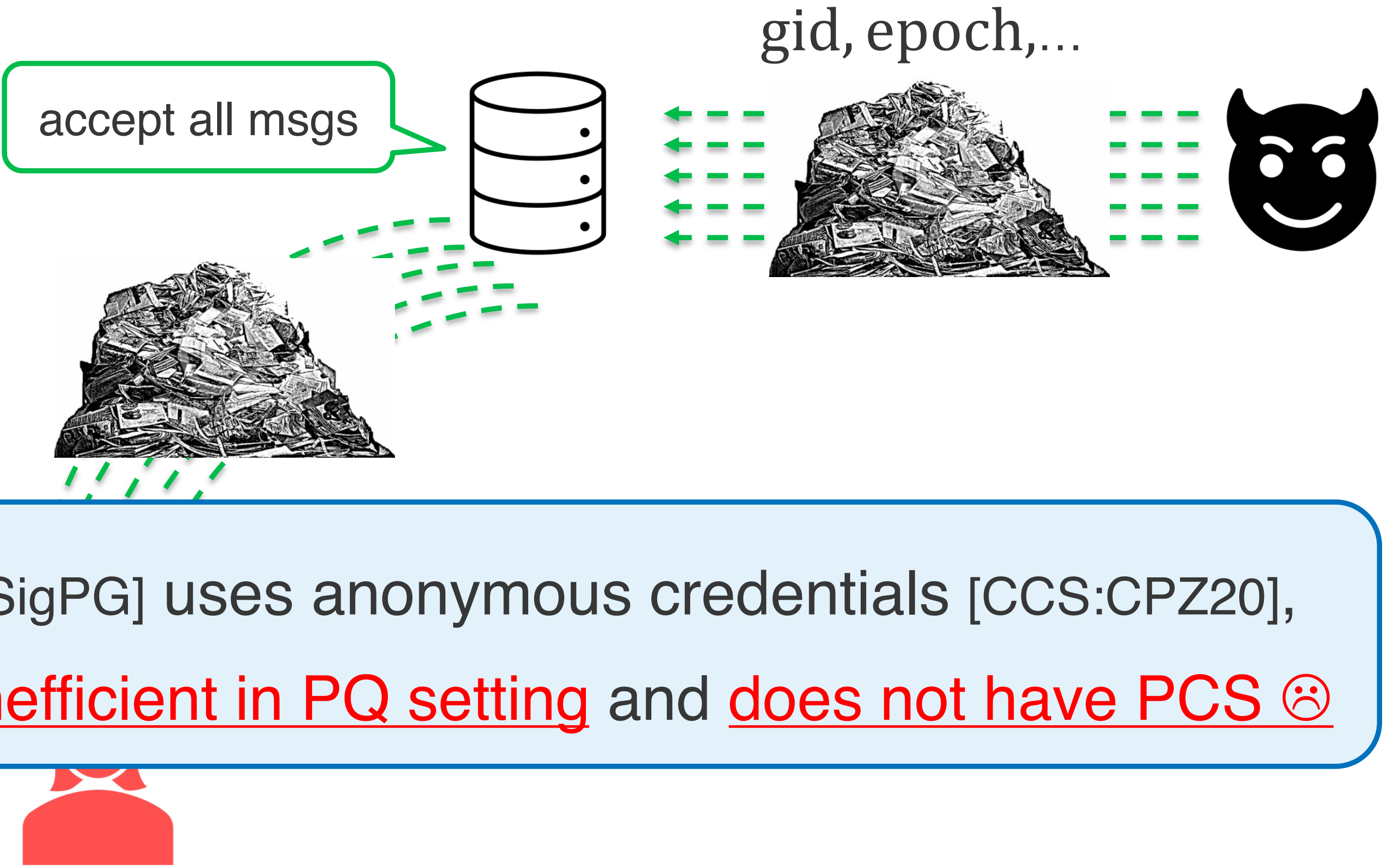
Contribution 2: Protecting dynamic metadata

- Without authentication causes denial of service attacks against groups



Contribution 2: Protecting dynamic metadata

- Without authentication causes denial of service attacks against groups

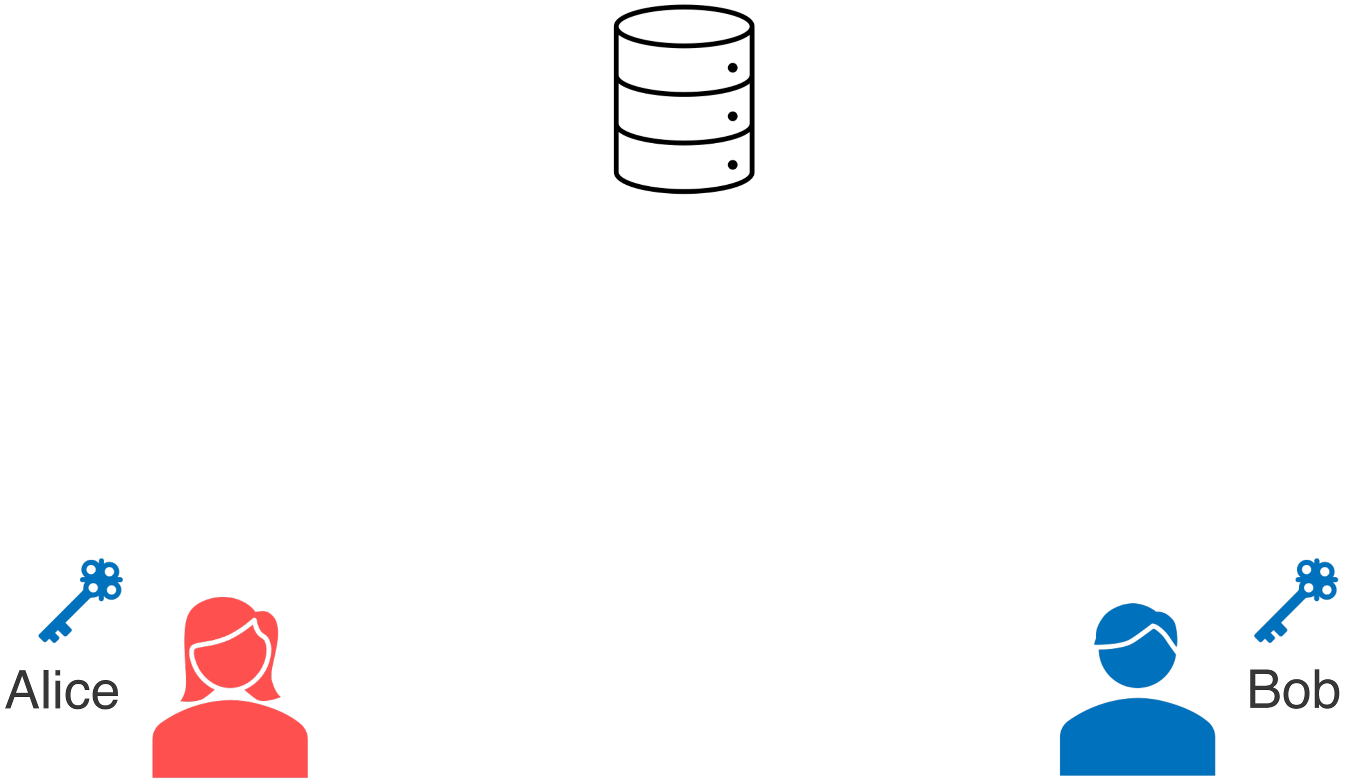


Signal [SigPG] uses anonymous credentials [CCS:CPZ20],

but it is inefficient in PQ setting and does not have PCS ☹

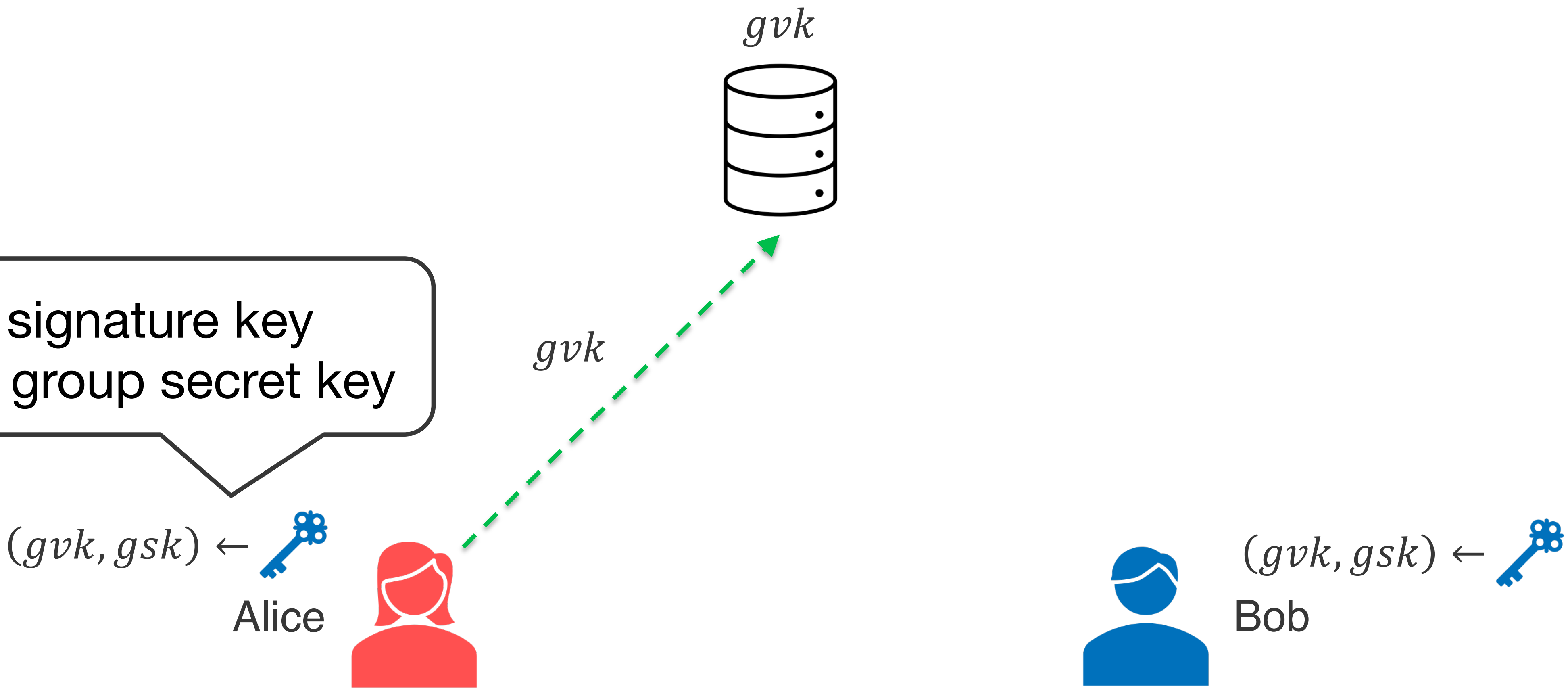
Contribution 2: Protecting dynamic metadata

💡 Use group secret key for the group membership authentication



Contribution 2: Protecting dynamic metadata

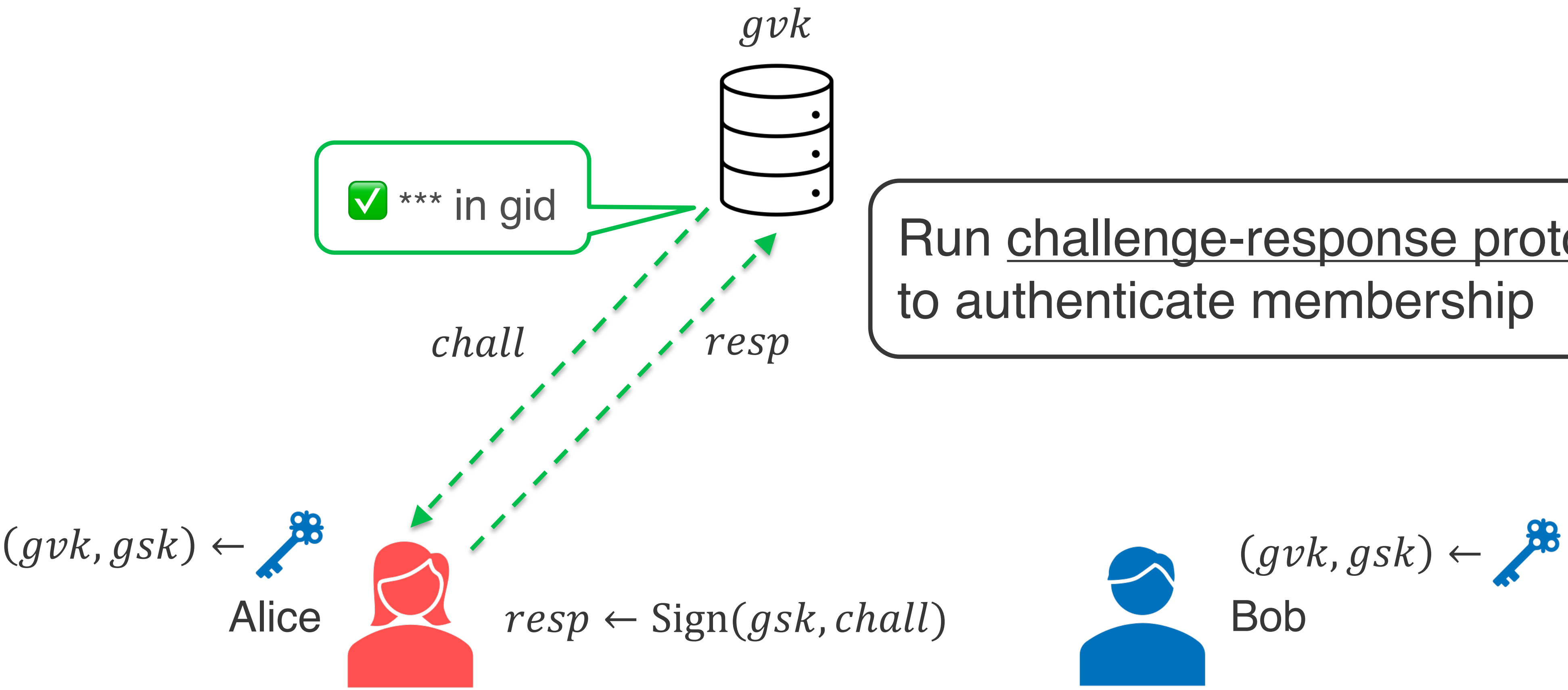
💡 Use group secret key for the group membership authentication



Derive a signature key from the group secret key

Contribution 2: Protecting dynamic metadata

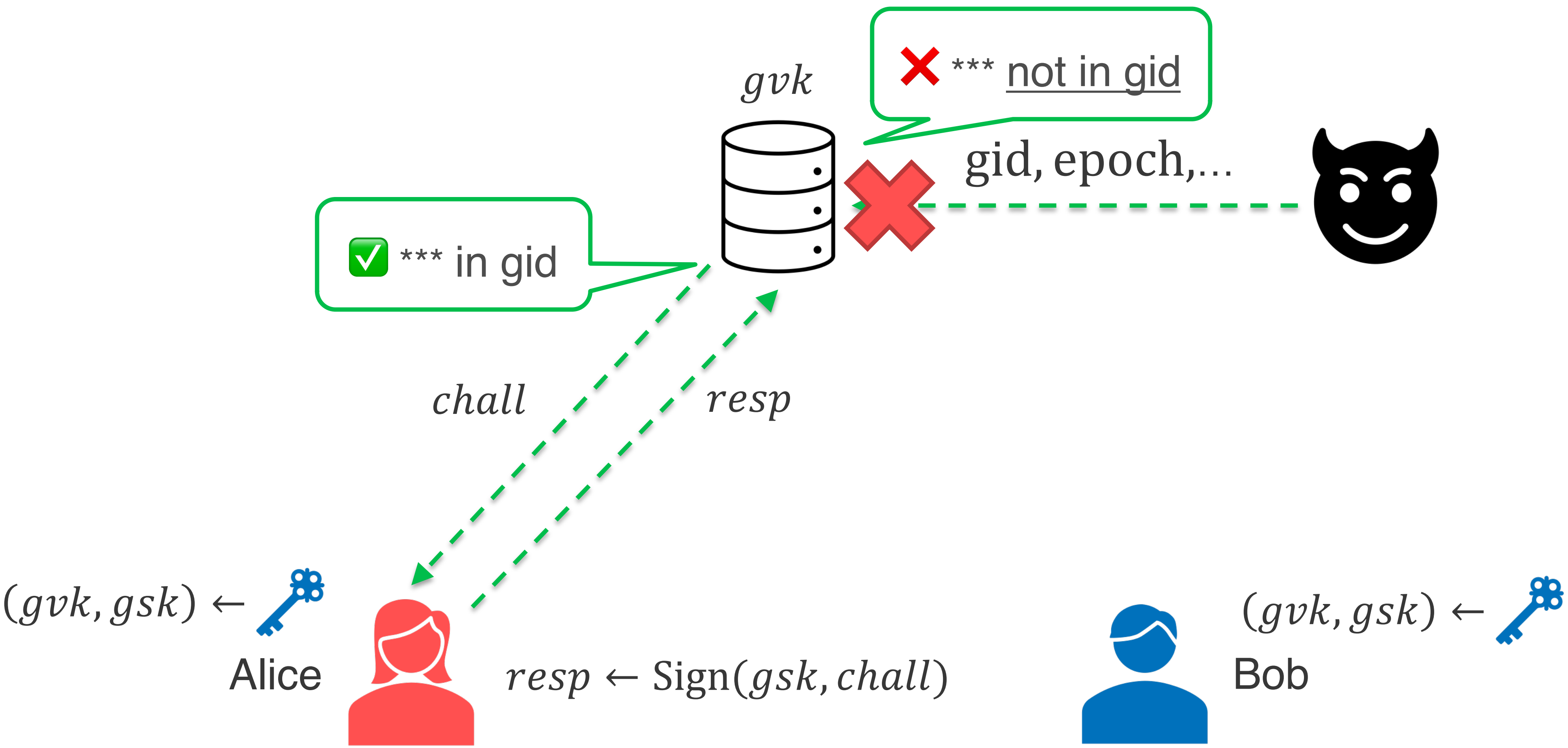
Use group secret key for the group membership authentication



Run challenge-response protocol to authenticate membership

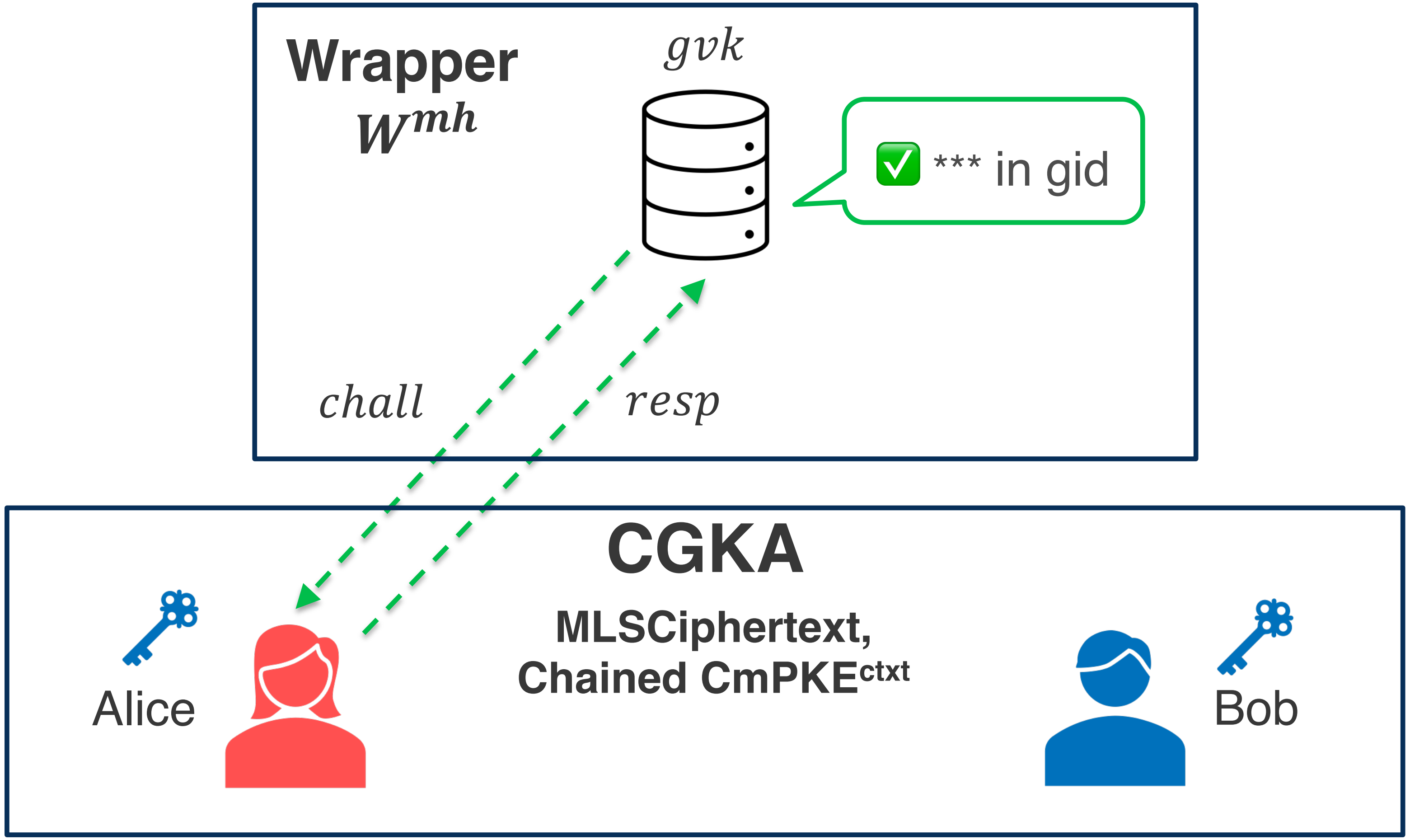
Contribution 2: Protecting dynamic metadata

Server can authenticate users without knowing other information



Contribution 2: Protecting dynamic metadata

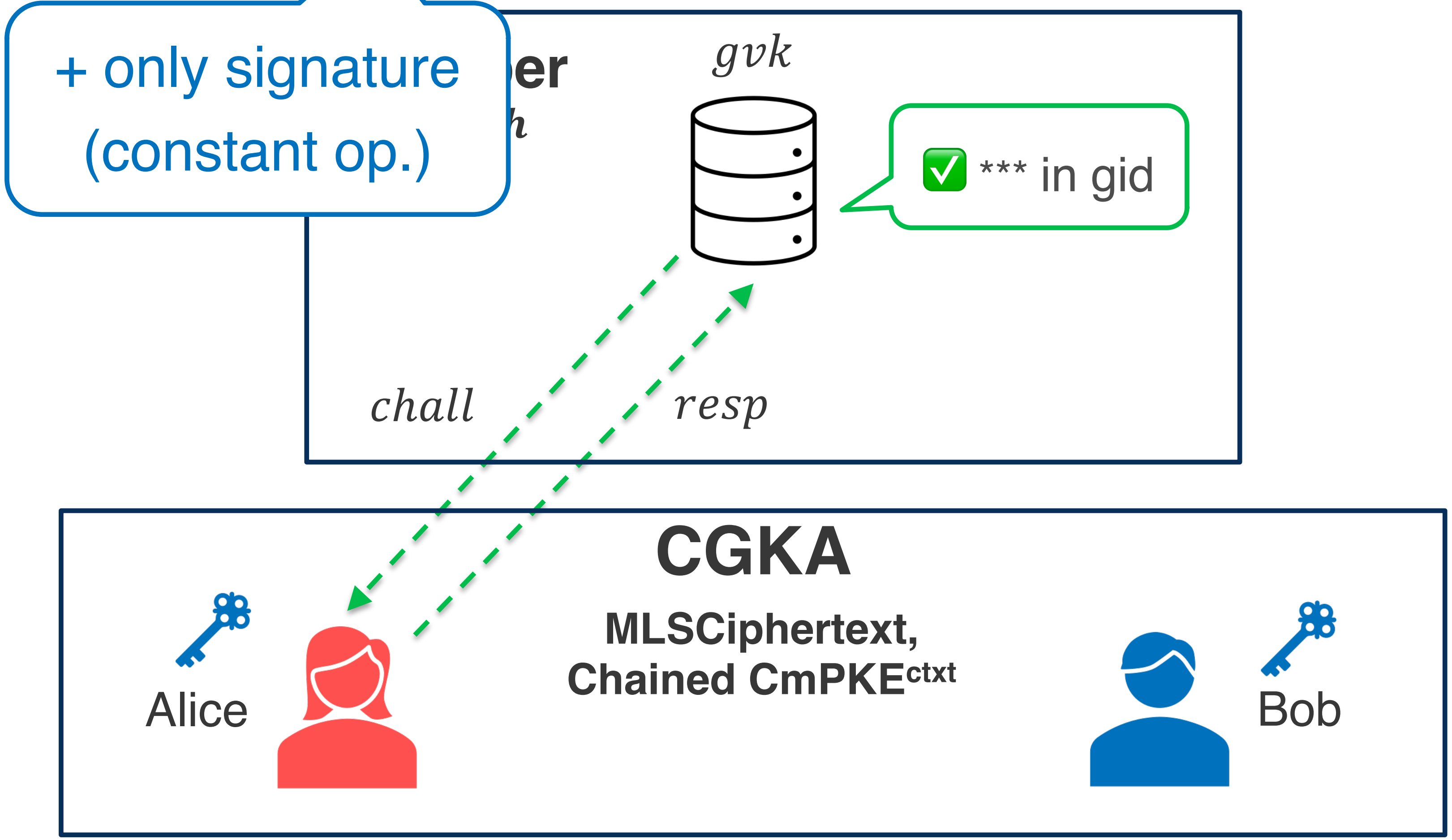
Construct an efficient and generic wrapper protocol W^{mh}



Contribution 2: Protecting dynamic metadata

Construct an efficient and generic wrapper protocol W^{mh}

+ only signature (constant op.)



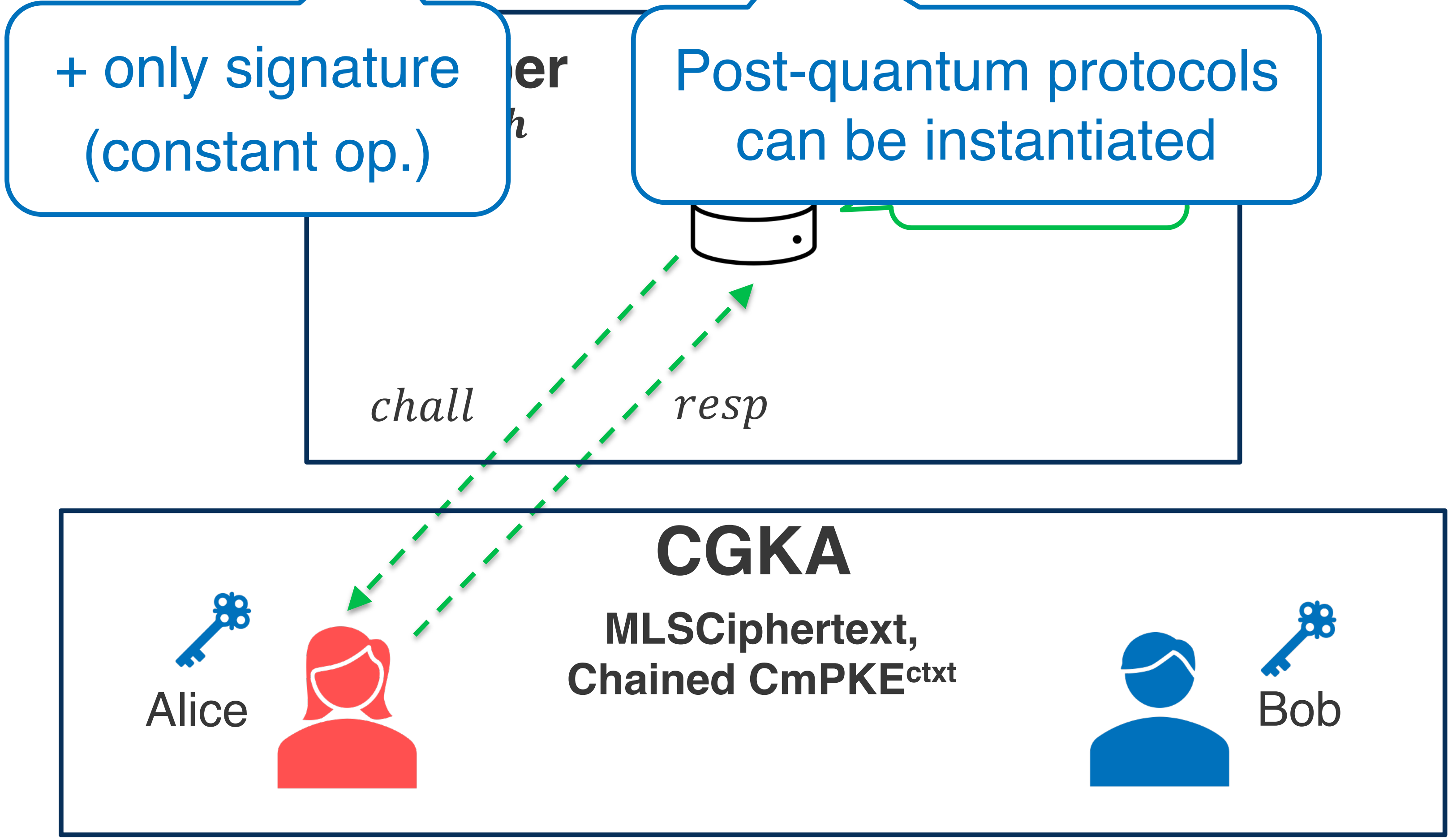
MLSCiphertext, Chained CmPKE^{ctxt}

Contribution 2: Protecting dynamic metadata

Construct an efficient and generic wrapper protocol W^{mh}

+ only signature (constant op.)

Post-quantum protocols can be instantiated



Contribution 3: Formal analysis of all metadata

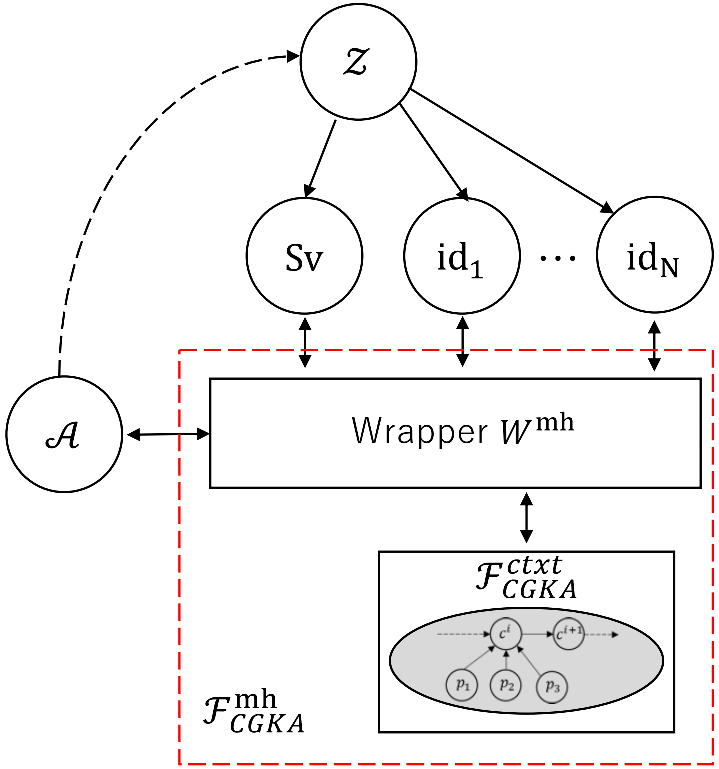
Propose a UC security model $\mathcal{F}_{CGKA}^{mh}$ capturing the security of **key**, **static metadata** and **dynamic metadata**

	Secret keys	Secret keys +static metadata	Secret keys +static metadata +dynamic metadata
MLS 🔒	MLSPlaintext	MLSCiphertext	**Contrib. 2**
Security proofs	[C:ACDT20, CCS:ACDT21, C:AJM22]	**Contrib. 1**	**Contrib. 3**

Contribution 3: Formal analysis of all metadata

Propose a UC security model $\mathcal{F}_{CGKA}^{mh}$ capturing the security of key, static metadata and dynamic metadata

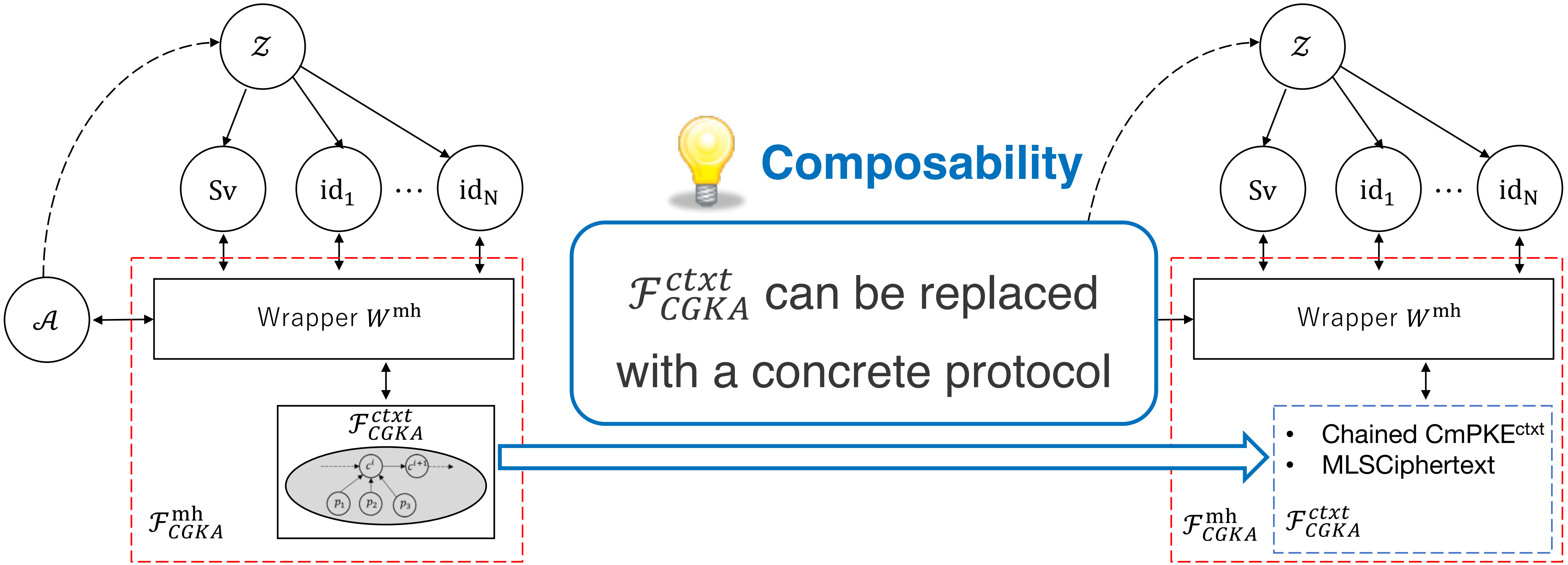
- Prove our wrapper W^{mh} realize $\mathcal{F}_{CGKA}^{mh}$ in $\mathcal{F}_{CGKA}^{ctxt}$-hybrid model



Contribution 3: Formal analysis of all metadata

Propose a UC security model $\mathcal{F}_{CGKA}^{mh}$ capturing the security of key, static metadata and dynamic metadata

- Prove our wrapper W^{mh} realize $\mathcal{F}_{CGKA}^{mh}$ in $\mathcal{F}_{CGKA}^{ctxt}$-hybrid model


$\mathcal{Z}$
Sv
id_1
$\cdots$
id_N
$\mathcal{A}$
Wrapper W^{mh}
$\mathcal{F}_{CGKA}^{ctxt}$
c^i
c^{i+1}
p_1
p_2
p_3
$\mathcal{F}_{CGKA}^{\mathrm{mh}}$
$\mathcal{Z}$
Sv
id_1
$\cdots$
id_N
Wrapper W^{mh}
$\mathcal{F}_{CGKA}^{\mathrm{mh}}$
$\mathcal{F}_{CGKA}^{ctxt}$

Composability

$\mathcal{F}_{CGKA}^{ctxt}$ can be replaced with a concrete protocol

- Chained CmPKE$^{\mathrm{ctxt}}$
- MLSCiphertext

Summary

	Secret keys	Secret keys +static metadata	Secret keys +static metadata +dynamic metadata
Signal	Vanilla Signal		Private Groups [SigPG]
Security proofs	✗		*Only metadata [CCS:CPZ20]
MLS	MLSPlaintext	MLSCiphertext	✓ **Contrib. 2**
Security proofs	✓ [C:ACDT20, CCS:ACDT21, C:AJM22]	✓ **Contrib. 1**	✓ **Contrib. 3**

The first probably secure metadata-hiding CGKA based on Chained CmPKE [CCS:HKPPW21]

References

- [SigPG] Technology Preview: Signal Private Group System. https://signal.org/ blog/signal-private-group-system/, 2019.
- [CCS:CPZ20] M. Chase, T. Perrin, and G. Zaverucha. The Signal Private Group System and Anonymous Credentials Supporting Efficient Verifiable Encryption. In ACM CCS 2020.
- [C:ACDT20] J. Alwen, S. Coretti, Y. Dodis, and Y. Tselekounis. Security Analysis and Improvements for the IETF MLS Standard for Group Messaging. In CRYPTO 2020.
- [CCS:ACDT21] J. Alwen, S. Coretti, Y. Dodis, and Y. Tselekounis. Modular Design of Secure Group Messaging Protocols and the Security of MLS. In ACM CCS 2021.
- [CCS:HKPPW21] K. Hashimoto, S. Katsumata, E. Postlethwaite, T. Prest, and B. Westerbaan. A Concrete Treatment of Efficient Continuous Group Key Agreement via Multi-Recipient PKEs. In ACM CCS 2021.
- [C:AJM22] J. Alwen, D. Jost, and M. Mularczyk. On The Insider Security of MLS. In CRYPTO 2022.
- [EC:AAC+22] J. Alwen, B. Auerbach, M. Cueto Noval, K. Klein, G. Pascual-Perez, K. Pietrzak, and M. Walter, “CoCoA: Concurrent Continuous Group Key Agreement.” In EUROCRYPT 2022.
- [CCS:AHKM22] J. Alwen, D. Hartmann, E. Kiltz, and M. Mularczyk, “Server-Aided Continuous Group Key Agreement.” In ACM CCS 2022.
- [BBM+22] R. Barnes, B. Beurdouche, J. Millican, E. Omara, K. Cohn-Gordon, and R. Robert. The Messaging Layer Security (MLS) Protocol. Internet Engineering Task Force. 2022.
- [OBR+22] E. Omara, B. Beurdouche, E. Rescorla, S. Inguva, A. Kwon, and A. Duric. The Messaging Layer Security (MLS) Architecture. Internet Engineering Task Force. 2022.