A Formal Treatment of End-to-End Encrypted Cloud Storage

Matilda Backendal[1], Hannah Davis[2], Felix Günther[3], Miro Haller[4], Kenny Paterson[1]

[1]ETH Zurich , [2]Seagate Technology, [3]IBM Research Zurich, [4]UC San Diego

MIT, November 21, 2024

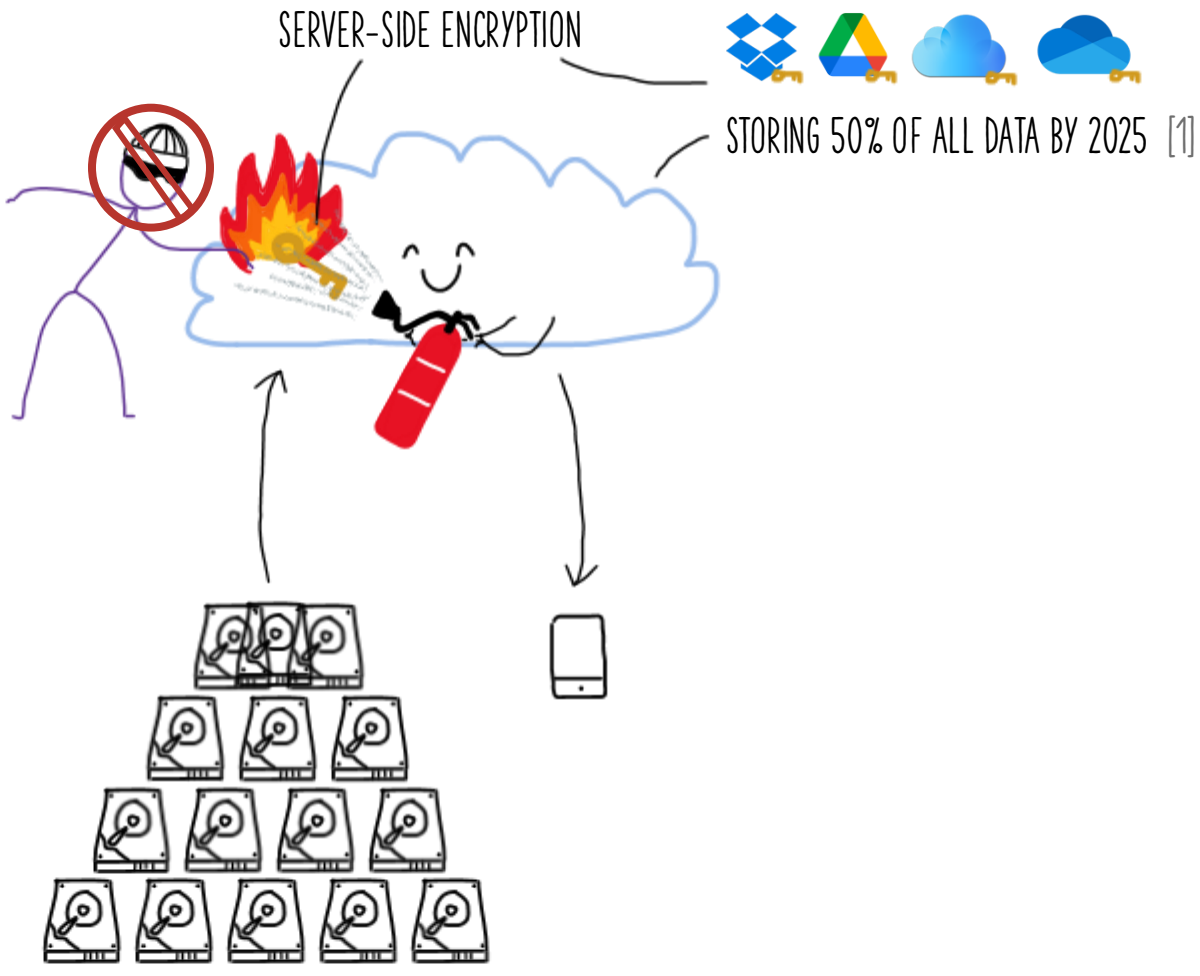
Cloud Storage

Benefits:

+ Availability
+ Redundancy
+ Scalability

Concerns:

- Data leaks to third party
  => SERVER-SIDE ENCRYPTION



[1] https://cybersecurityventures.com/the-world-will-store-200-zettabytes-of-data-by-2025/ (Sausalito, Calif., Feb. 1, 2024)

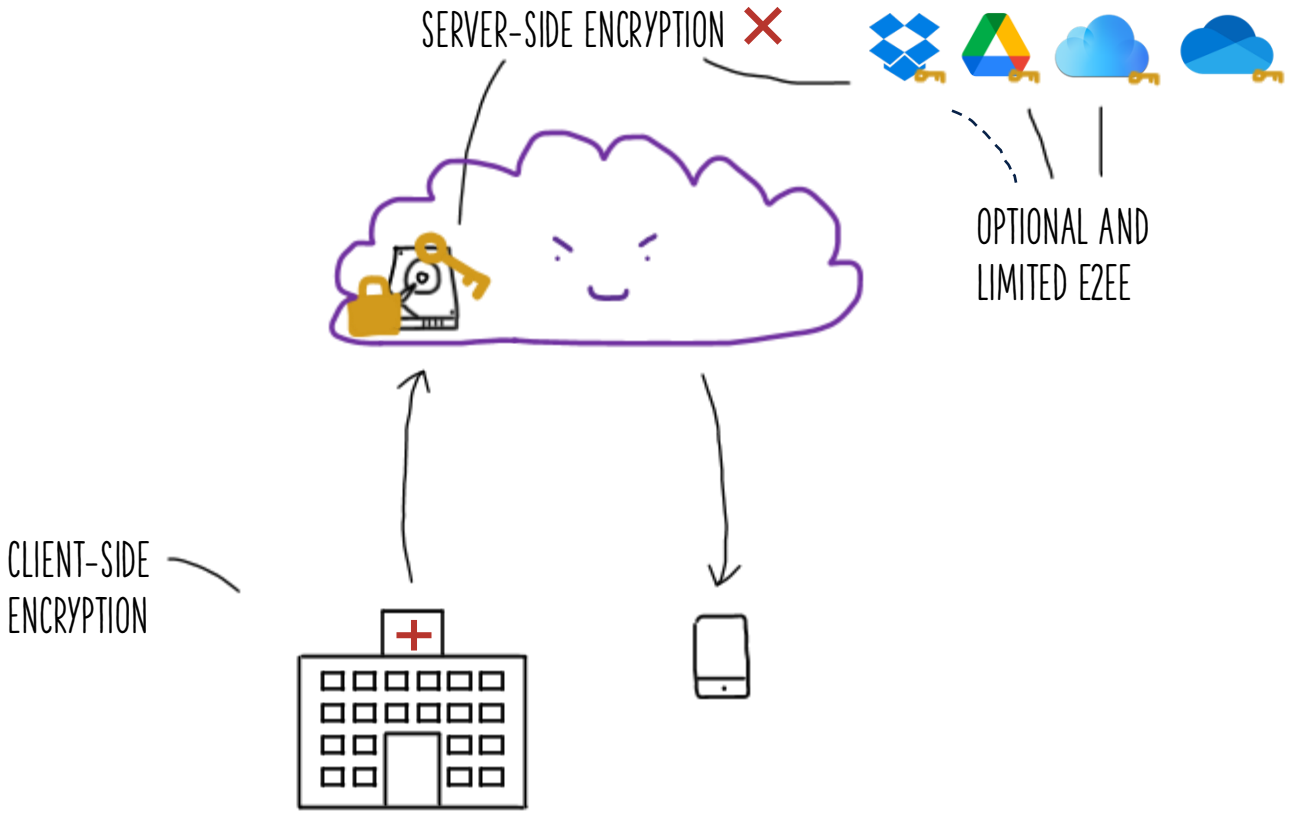
Cloud Storage

Benefits:

+ Availability
+ Redundancy
+ Scalability

Concerns:

- Data leaks to third party
  => SERVER-SIDE ENCRYPTION
- Malicious server
  => END-TO-END ENCRYPTION



https://www.hipaajournal.com/healthcare-cloud-usage-grows-but-protecting-phi-can-be-a-challenge/

E2EE Cloud Storage Providers

"WITH **MEGA**, YOU CONTROL THE ENCRYPTION"

300 MILLION USERS

THE GERMAN FEDERAL GOVERNMENT, AMNESTY INTERNATIONAL, & ETH Zurich

"ULTIMATE SECURITY"

"FREE, ENCRYPTED, AND SECURE CLOUD STORAGE. YOUR PRIVACY, SECURED BY MATH"

"EXCEPTIONALLY PRIVATE CLOUD"

"THE STRONGEST ENCRYPTED CLOUD STORAGE IN THE WORLD"

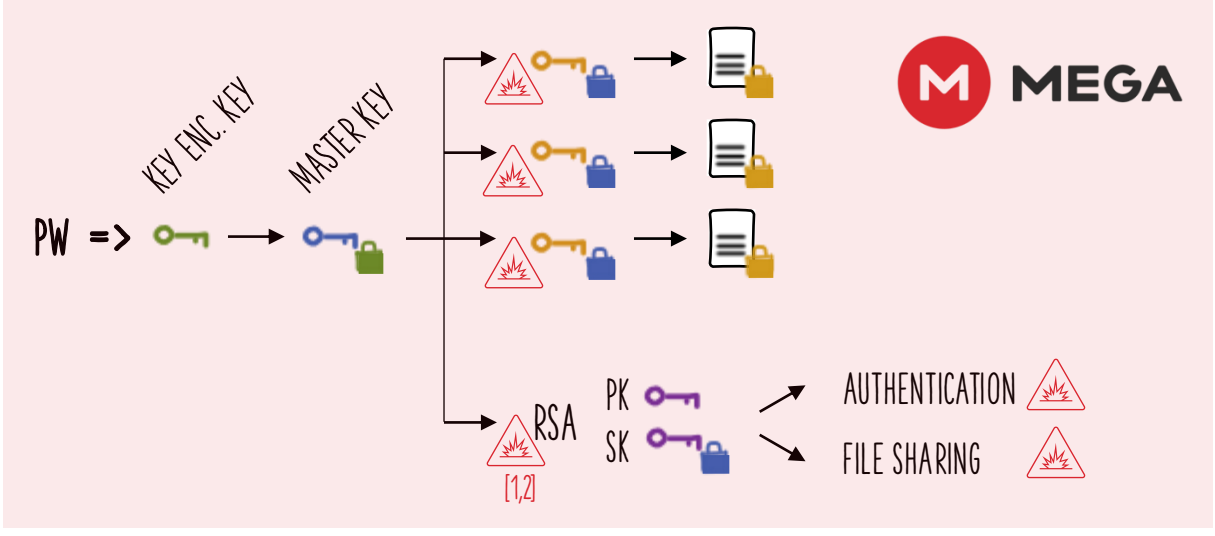
"EUROPE'S MOST SECURE CLOUD STORAGE"

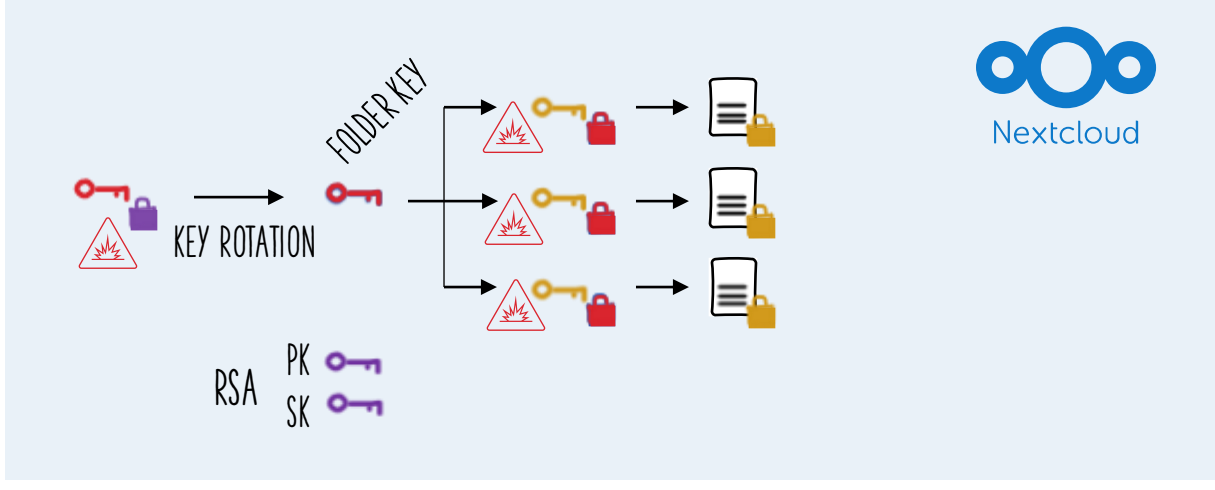
"SUPPORTS CLIENT-SIDE END-TO-END ENCRYPTION"

Case Studies: E2EE Cloud Storage

Challenges:

1	Stateless clients
2	No ciphertext integrity
3	Key recovery attacks [1,2]
4	Key reuse
5	File re-encryption infeasible
6	PKE has no authentication [3]





[1] Matilda Backendal, Miro Haller and Kenneth G. Paterson. (2023). "MEGA: Malleable Encryption Goes Awry". IEEE S&P 2023.

[2] Martin R. Albrecht, Miro Haller, Lenka Mareková, Kenneth G. Paterson. (2023). "Caveat Implementor! Key Recovery Attacks on MEGA". Eurocrypt 2023.

[3] Martin R. Albrecht, Matilda Backendal, Daniele Coppola, Kenneth G. Paterson. (2024). "Share with Care: Breaking E2EE in Nextcloud". Euro S&P 2024.

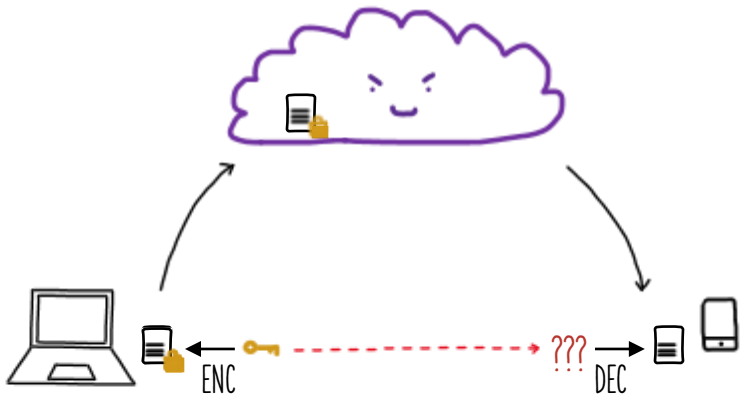
Case Studies: E2EE Cloud Storage … is surprisingly hard!

Challenges:

1	Stateless clients
2	No ciphertext integrity
3	Key recovery attacks [1,2]
4	Key reuse
5	File re-encryption infeasible
6	PKE has no authentication [3]

Implications:

- Design issues 2 4
- Key distribution problem 1



[1] Matilda Backendal, Miro Haller and Kenneth G. Paterson. (2023). "MEGA: Malleable Encryption Goes Awry". IEEE S&P 2023.

[2] Martin R. Albrecht, Miro Haller, Lenka Mareková, Kenneth G. Paterson. (2023). "Caveat Implementor! Key Recovery Attacks on MEGA". Eurocrypt 2023.

[3] Martin R. Albrecht, Matilda Backendal, Daniele Coppola, Kenneth G. Paterson. (2024). "Share with Care: Breaking E2EE in Nextcloud". Euro S&P 2024.

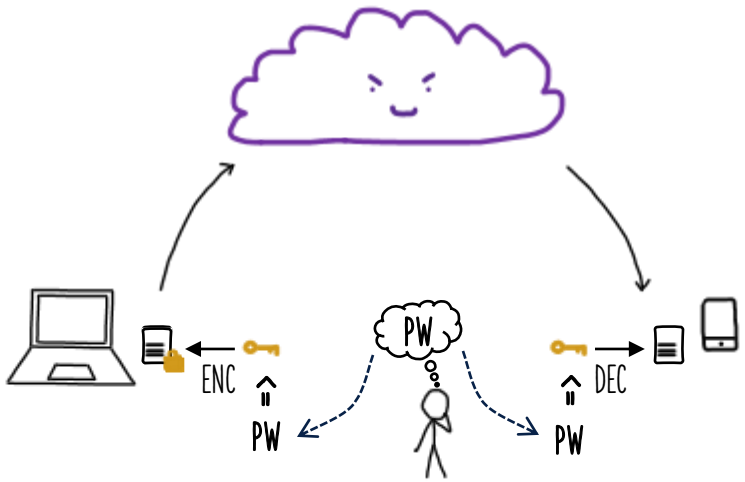
Case Studies: E2EE Cloud Storage … is surprisingly hard!

Challenges:

1	Stateless clients
2	No ciphertext integrity
3	Key recovery attacks [1,2]
4	Key reuse
5	File re-encryption infeasible
6	PKE has no authentication [3]

Implications:

- Design issues 2 4
- Key distribution problem 1
- Password-based security 1



[1] Matilda Backendal, Miro Haller and Kenneth G. Paterson. (2023). "MEGA: Malleable Encryption Goes Awry". IEEE S&P 2023.

[2] Martin R. Albrecht, Miro Haller, Lenka Mareková, Kenneth G. Paterson. (2023). "Caveat Implementor! Key Recovery Attacks on MEGA". Eurocrypt 2023.

[3] Martin R. Albrecht, Matilda Backendal, Daniele Coppola, Kenneth G. Paterson. (2024). "Share with Care: Breaking E2EE in Nextcloud". Euro S&P 2024.

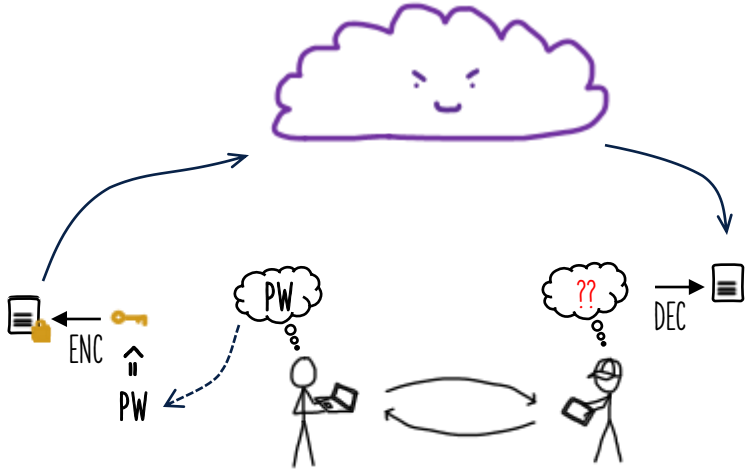
Case Studies: E2EE Cloud Storage … is surprisingly hard!

Challenges:

1	Stateless clients
2	No ciphertext integrity
3	Key recovery attacks [1,2]
4	Key reuse
5	File re-encryption infeasible
6	PKE has no authentication [3]

Implications:

- Design issues 2 4
- Key distribution problem 1
- Password-based security 1
- File sharing causes complex interactions 3 6
- Need to get it right the first time 5



[1] Matilda Backendal, Miro Haller and Kenneth G. Paterson. (2023). "MEGA: Malleable Encryption Goes Awry". IEEE S&P 2023.

[2] Martin R. Albrecht, Miro Haller, Lenka Mareková, Kenneth G. Paterson. (2023). "Caveat Implementor! Key Recovery Attacks on MEGA". Eurocrypt 2023.

[3] Martin R. Albrecht, Matilda Backendal, Daniele Coppola, Kenneth G. Paterson. (2024). "Share with Care: Breaking E2EE in Nextcloud". Euro S&P 2024.

E2EE Cloud Storage Providers

"WITH **MEGA**, YOU CONTROL THE ENCRYPTION"

300 MILLION USERS

MEGA

INSECURE!

[SP:BHP23]
[EC:AHMP23]

AMNESTY INTERNATIONAL, THE GERMAN FEDERAL GOVERNMENT & ETH

"ULTIMATE SECURITY"

Nextcloud

INSECURE!

[EuroSP:ABCP23]

"FREE, ENCRYPTED, AND SECURE CLOUD STORAGE. YOUR PRIVACY, SECURED BY MATH"

Proton Drive

NOT PROVABLY SECURE

"EXCEPTIONALLY PRIVATE CLOUD"

sync.com

"EUROPE'S MOST SECURE CLOUD STORAGE"

pCloud

INSECURE!

[CCS:TH24]

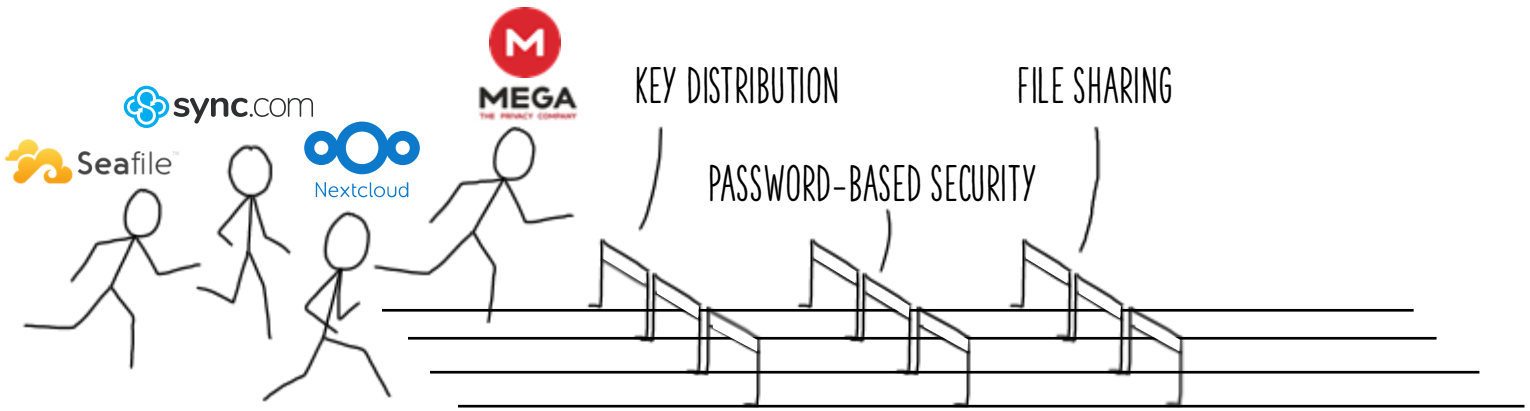
icedrive

"THE STRONGEST ENCRYPTED CLOUD STORAGE IN THE WORLD"

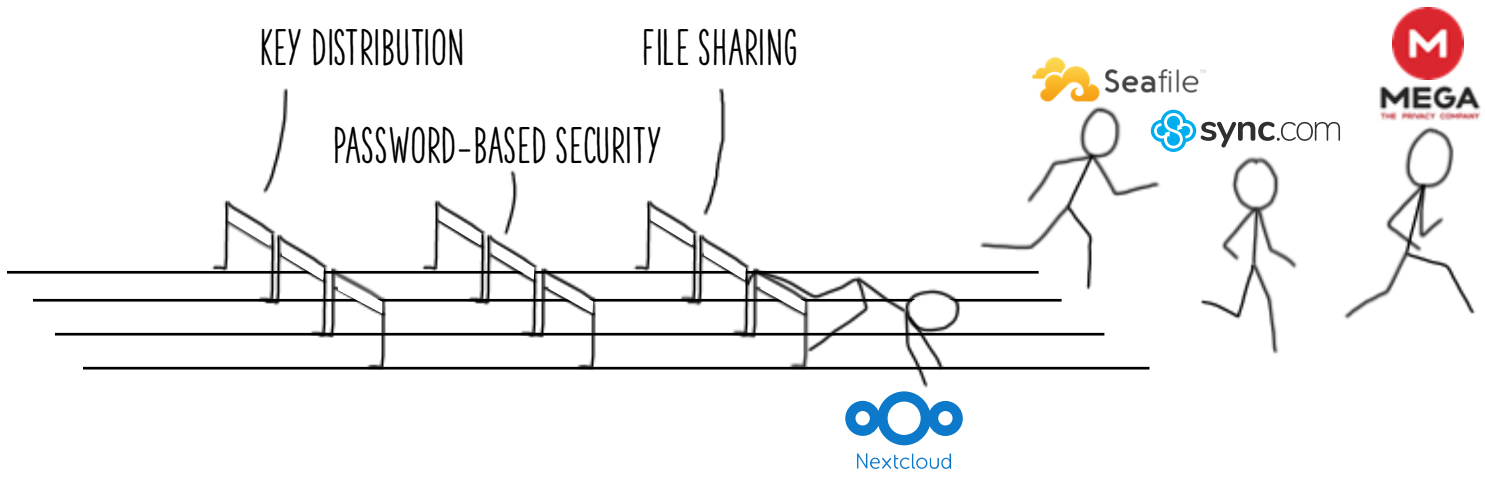
Seafile

"SUPPORTS CLIENT-SIDE END-TO-END ENCRYPTION"

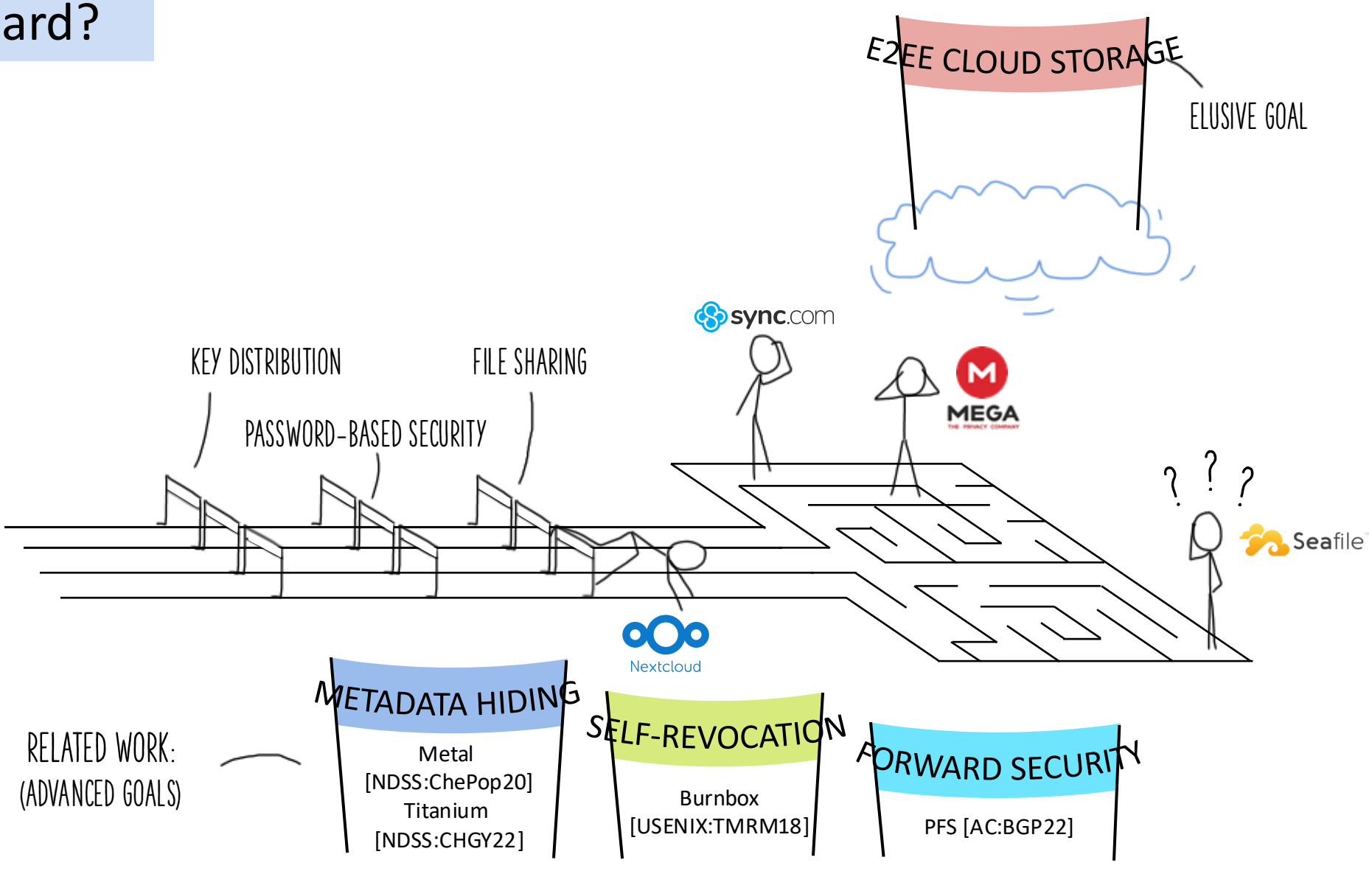
Why Is It Hard?



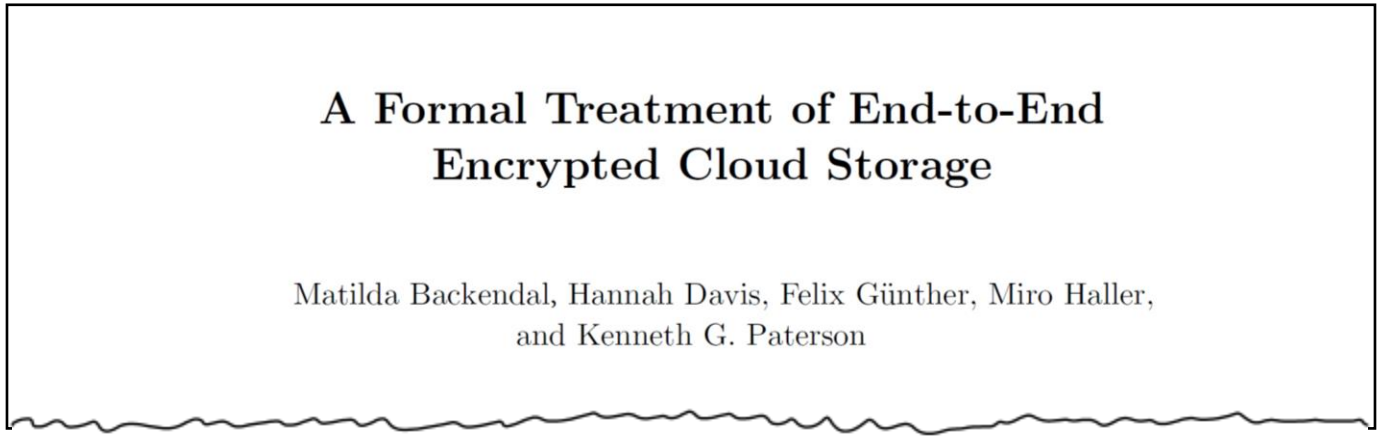
Why Is It Hard?



Why Is It Hard?



Contributions


A Formal Treatment of End-to-End Encrypted Cloud Storage

Matilda Backendal, Hannah Davis, Felix Günther, Miro Haller, and Kenneth G. Paterson

1 Formal Model

- Syntax
- Security games

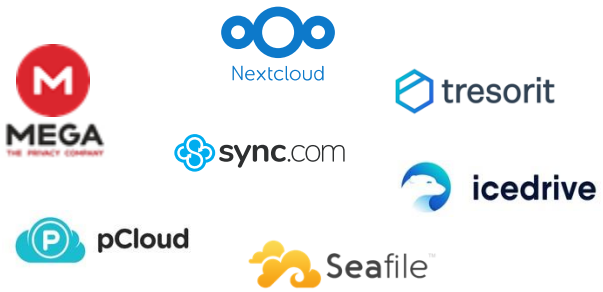
2 Construction

- CSS (Cloud Storage Scheme)
- Security proofs

1. Formalizing E2EE Cloud Storage

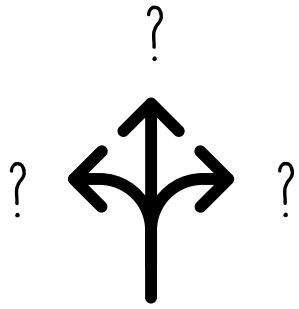
Formalizing E2EE Cloud Storage

Model Goals



Capture existing systems

Capture *real-world* systems



Capture future systems

1	Expressive
2	Faithful
3	Generic

Syntax

WHAT MAKES A CLOUD STORAGE A CLOUD STORAGE?

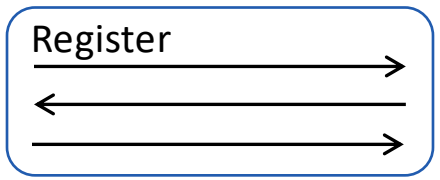
Core Functionality

1 EXPRESSIVE

- Register (create account)
- Authenticate (log in)

- Put (upload a file)
- Update (modify content)
- Get (download)
- Share
- Accept (receive share)

INTERACTIVE PROTOCOLS



Syntax

HOW DO WE MAKE THE MODEL USEFUL?

Core Functionality

1 EXPRESSIVE

- Register (create account)
- Authenticate (log in)

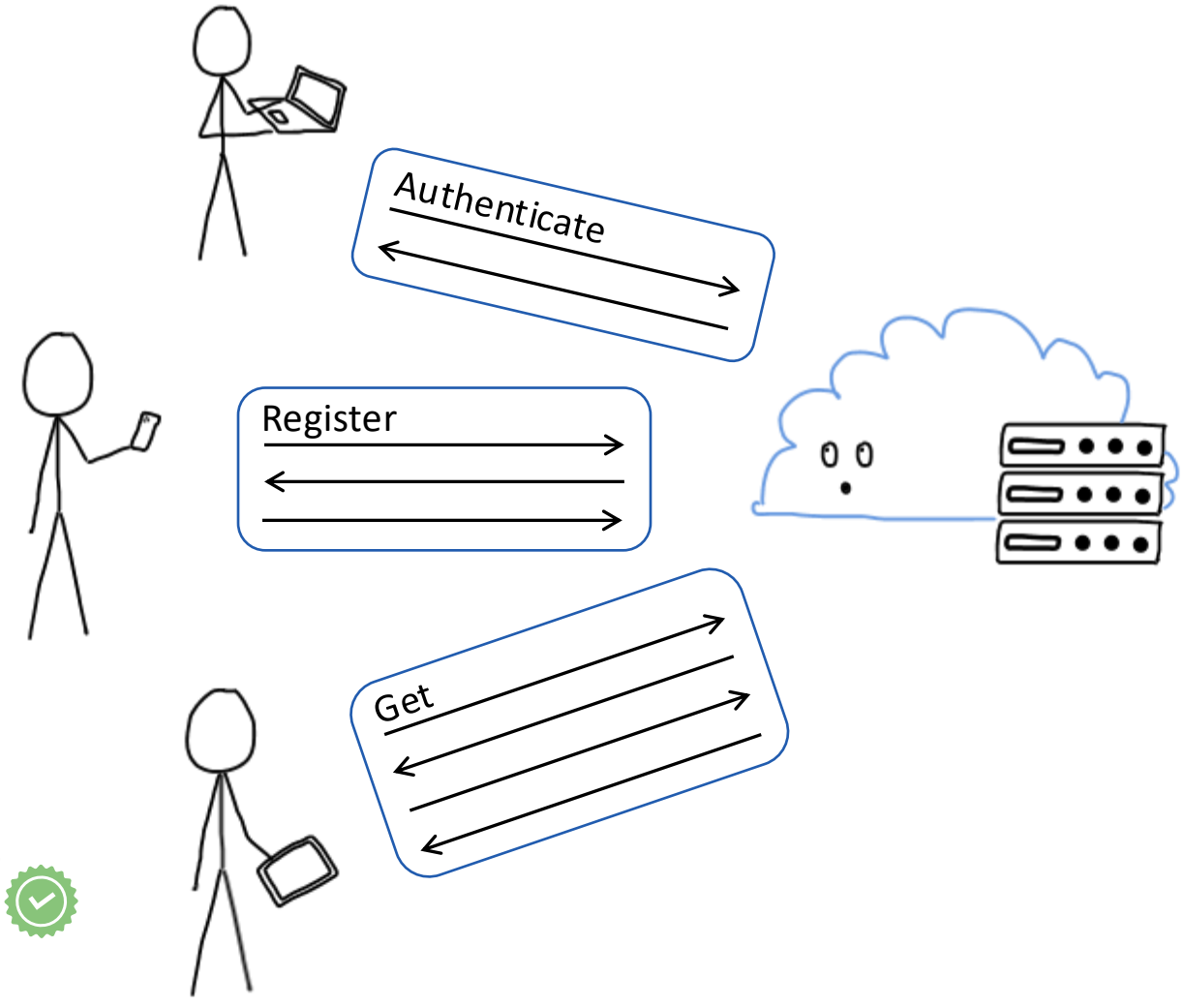
INTERACTIVE PROTOCOLS

- Put (upload a file)
- Update (modify content)
- Get (download)
- Share
- Accept (receive share)

Model Choices

- Arbitrary interleaving

2 FAITHFUL



Syntax

HOW DO WE MAKE THE MODEL USEFUL?

Core Functionality

1 EXPRESSIVE

- Register (create account)
- Authenticate (log in)
- Put (upload a file)
- Update (modify content)
- Get (download)
- Share
- Accept (receive share)

INTERACTIVE PROTOCOLS

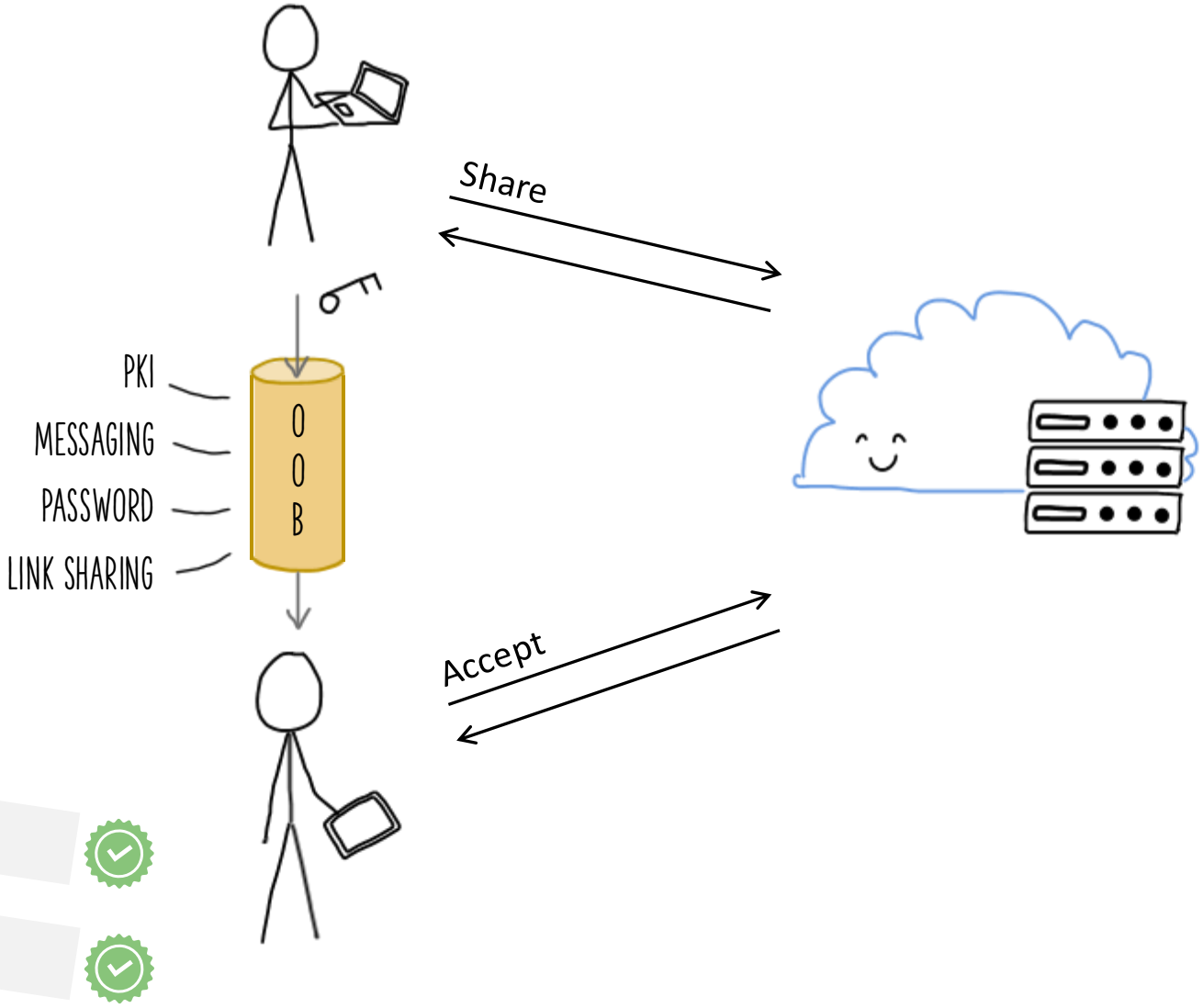
OFTEN NOT CONSIDERED IN RELATED WORK

Model Choices

- Arbitrary interleaving
- Abstract OOB channel for sharing

2 FAITHFUL

3 GENERIC



Security Notions

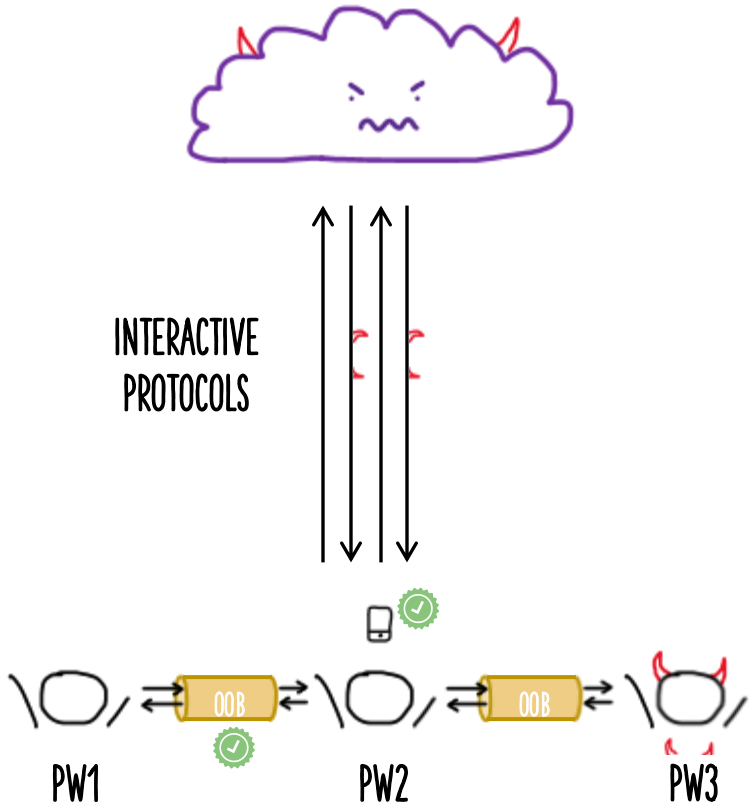
MALICIOUS SERVER SETTING

Threat model:

- Malicious cloud provider
- Trusted OOB-channels between honest users
- Trusted client code

Adversary capabilities:

- Control client protocol steps (which & when)
- Specify server responses
- Guess honest user passwords
- Compromise users (adaptive/selective)



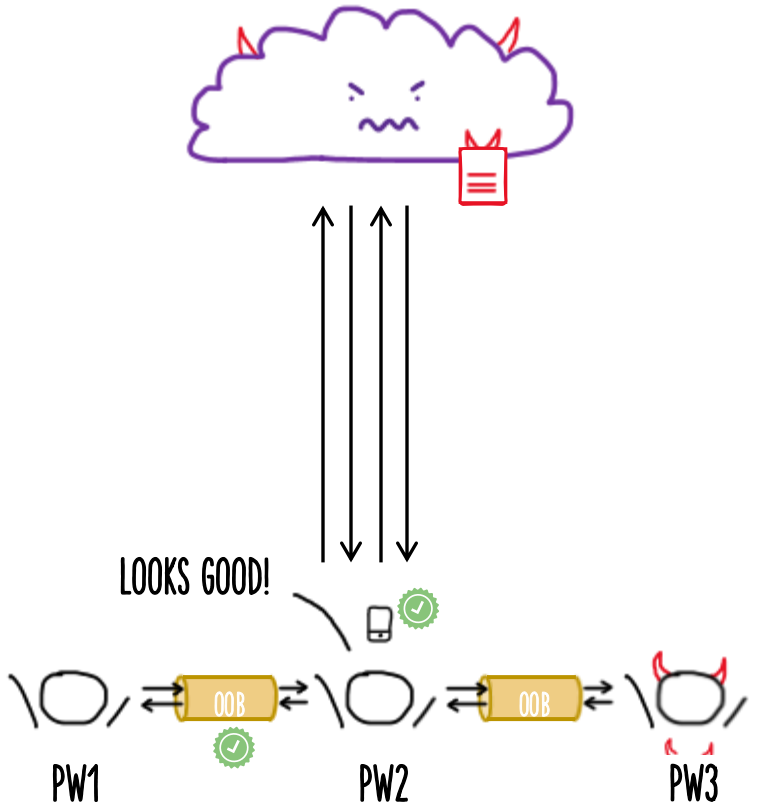
Security Notions

MALICIOUS SERVER SETTING

Integrity:

- Wins if adversary can, for an honest user,
  1. inject a file, or
  2. modify a file.

INT-PTXT-STYLE GAME



Security Notions

MALICIOUS SERVER SETTING

Integrity:

- Wins if adversary can, for an honest user,
  1. inject a file, or
  2. modify a file.

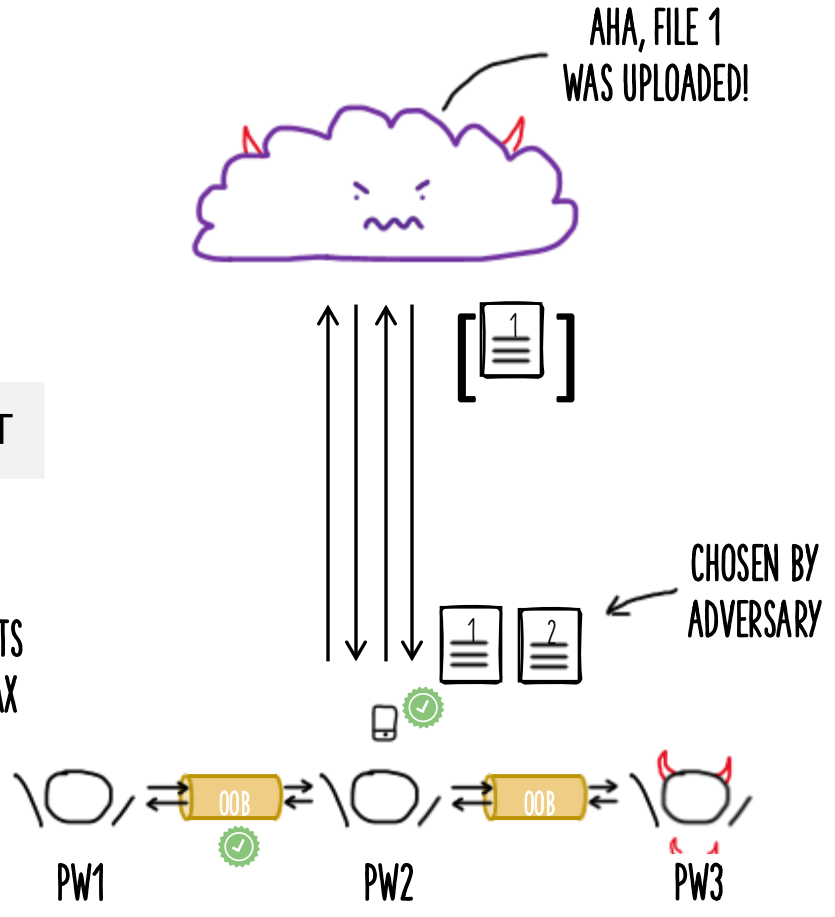
INT-PTXT-STYLE GAME → ! Not INT-CTXT

Confidentiality:

- Wins if adversary can, for an honest user,
  - learn any information and distinguish files

IND-CCA-STYLE GAME → ! Not IND$

NO CIPHERTEXTS IN OUR SYNTAX



Security Notions

MALICIOUS CLIENT SETTING [ONGOING WORK]

Threat model:

- ~~Malicious~~ honest cloud provider, malicious clients
- Trusted OOB-channels between honest users
- Trusted client code
- \+ Trusted client-to-server channels?

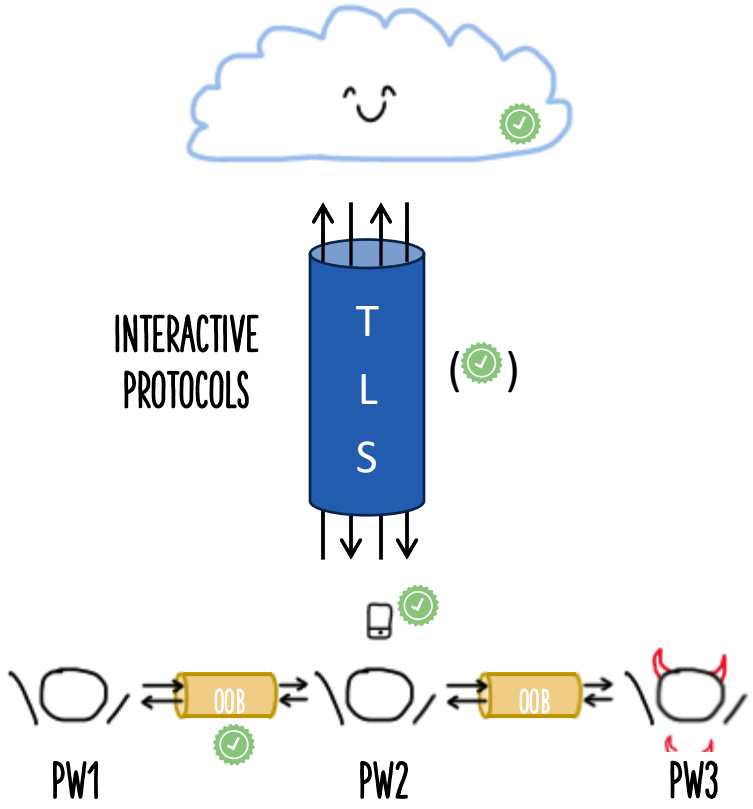
Adversary capabilities:

- Control client protocol steps (which & when)
- ~~Specify server responses~~
- Guess honest user passwords
- Compromise users (adaptive/selective)

Additional goals:

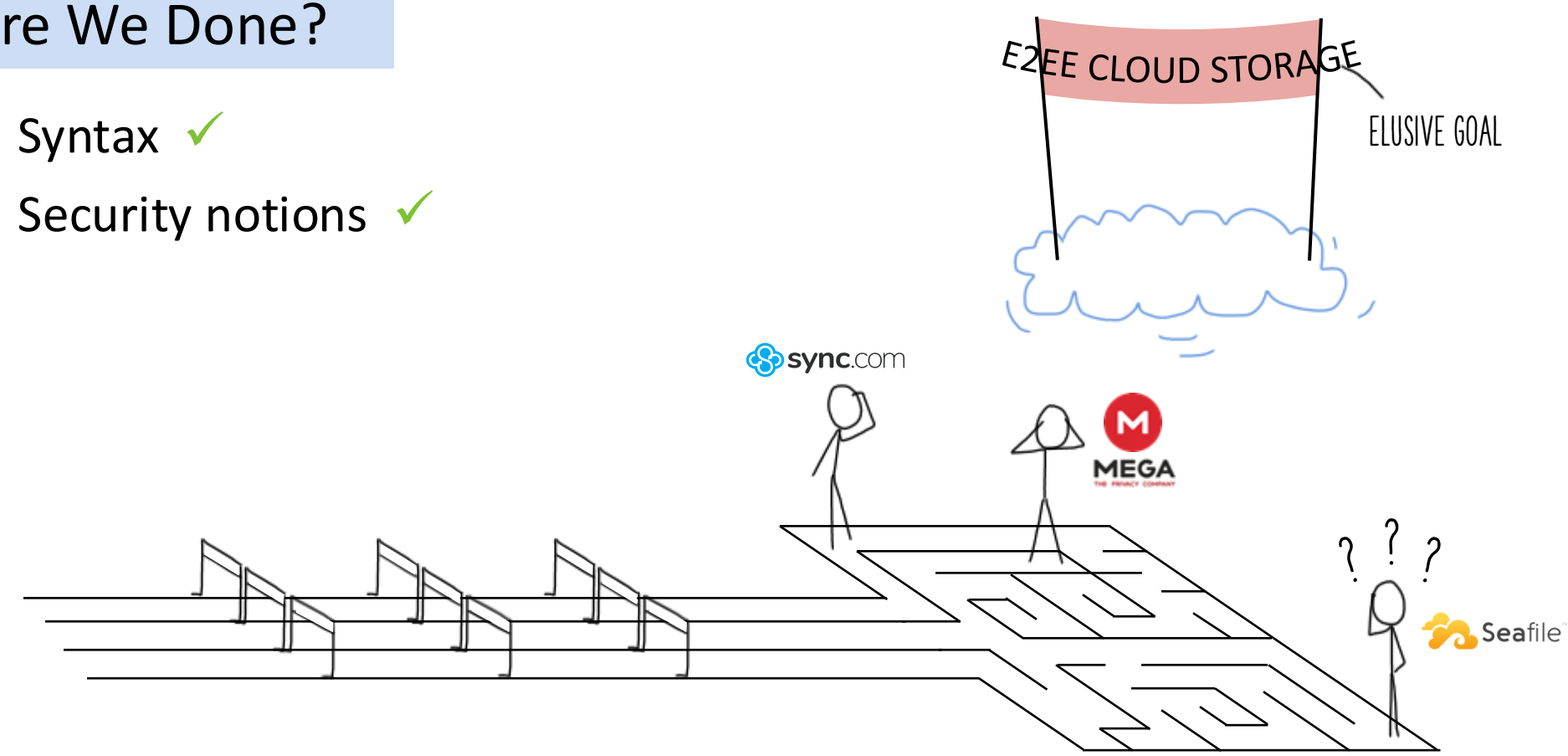
INFEASIBLE IN THE MALICIOUS SERVER SETTING!

- Authentication & authorization
- No offline dictionary attacks on pw
- Availability for honest user files



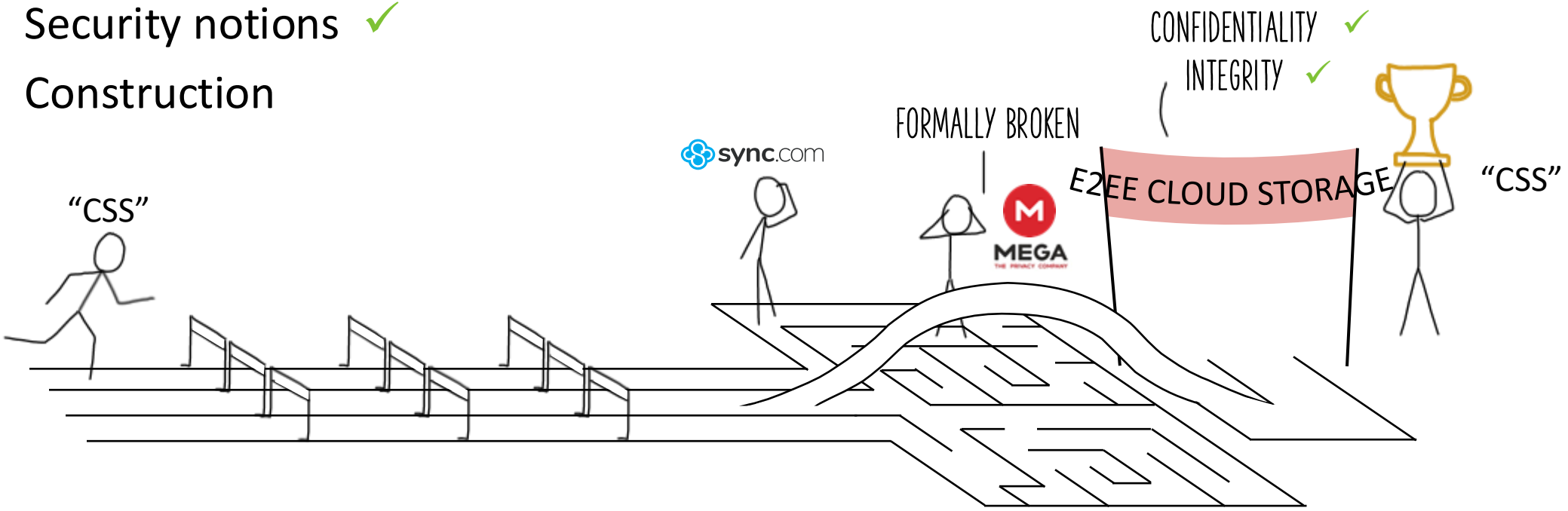
Are We Done?

- Syntax ✓
- Security notions ✓



Are We Done?

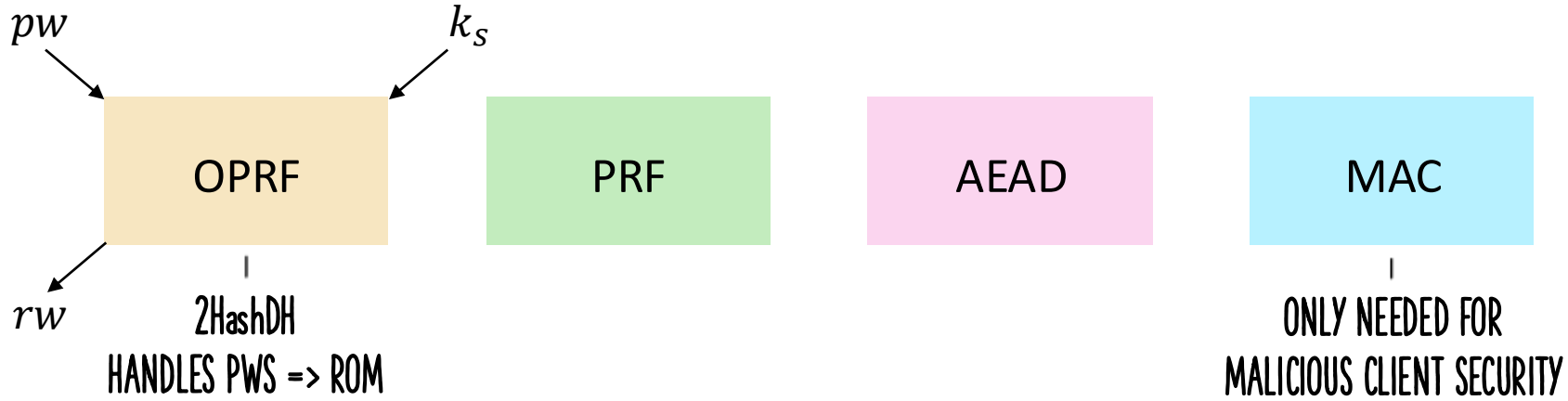
- Syntax ✓
- Security notions ✓
- Construction



2. Constructing E2EE Cloud Storage

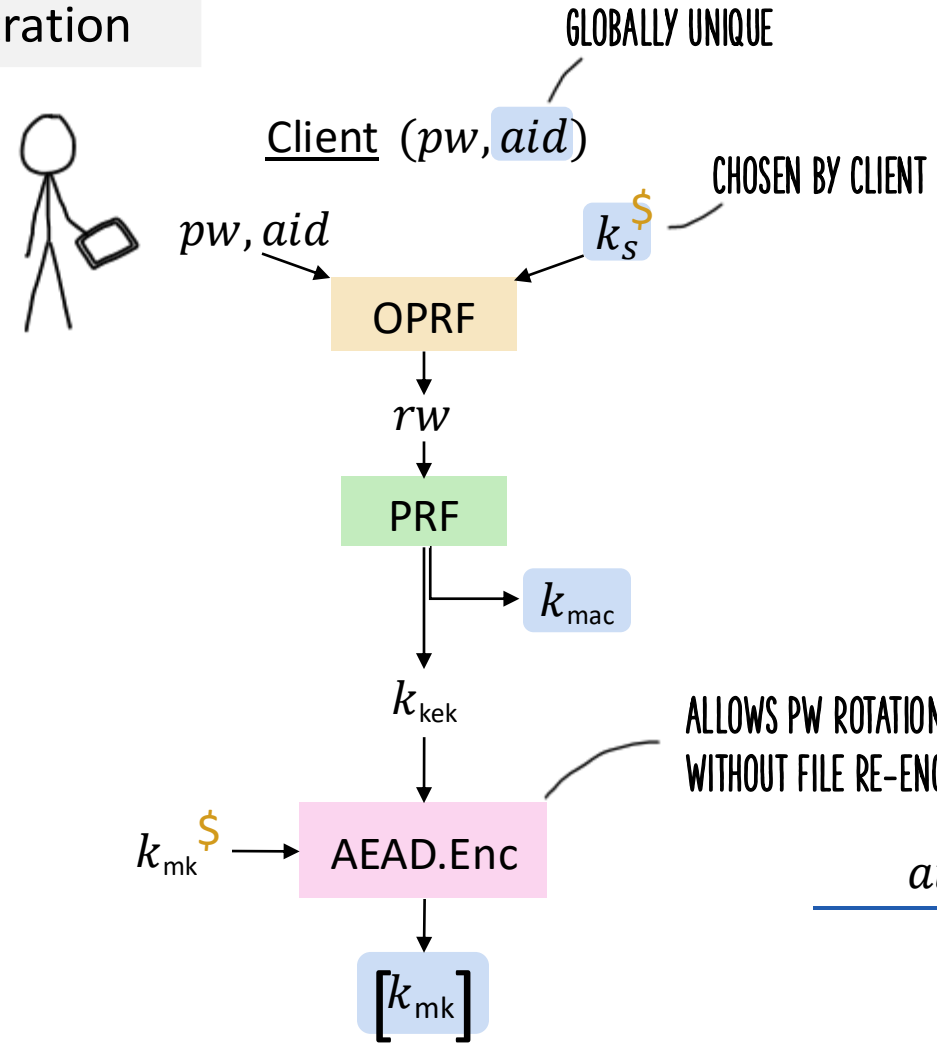
CSS (Cloud Storage Scheme)

Building Blocks



CSS (Cloud Storage Scheme)

Registration



Server

STORE

$User[aid] \leftarrow k_s, k_{mac}, [k_{mk}]$

$aid, k_s, k_{mac}, [k_{mk}]$

CSS (Cloud Storage Scheme)

Authentication

GLOBALLY UNIQUE

Client (pw, aid)

Server

PASSWORD HARDENING

pw, aid

k_s

OPRF

rw

PRF

k_{kek}

AEAD.Dec

$[k_{mk}]$

k_{mk}

CHALLENGE-RESPONSE AUTHENTICATION

sid $

sid

k_{mac}

MAC.Tag

τ

τ

τ sid

k_{mac}

MAC.Verify

$\perp / \top$

$[k_{mk}]$ (IF $\top$)

FETCH

$k_s, k_{mac}, [k_{mk}] \leftarrow User[aid]$

IF $\top$:

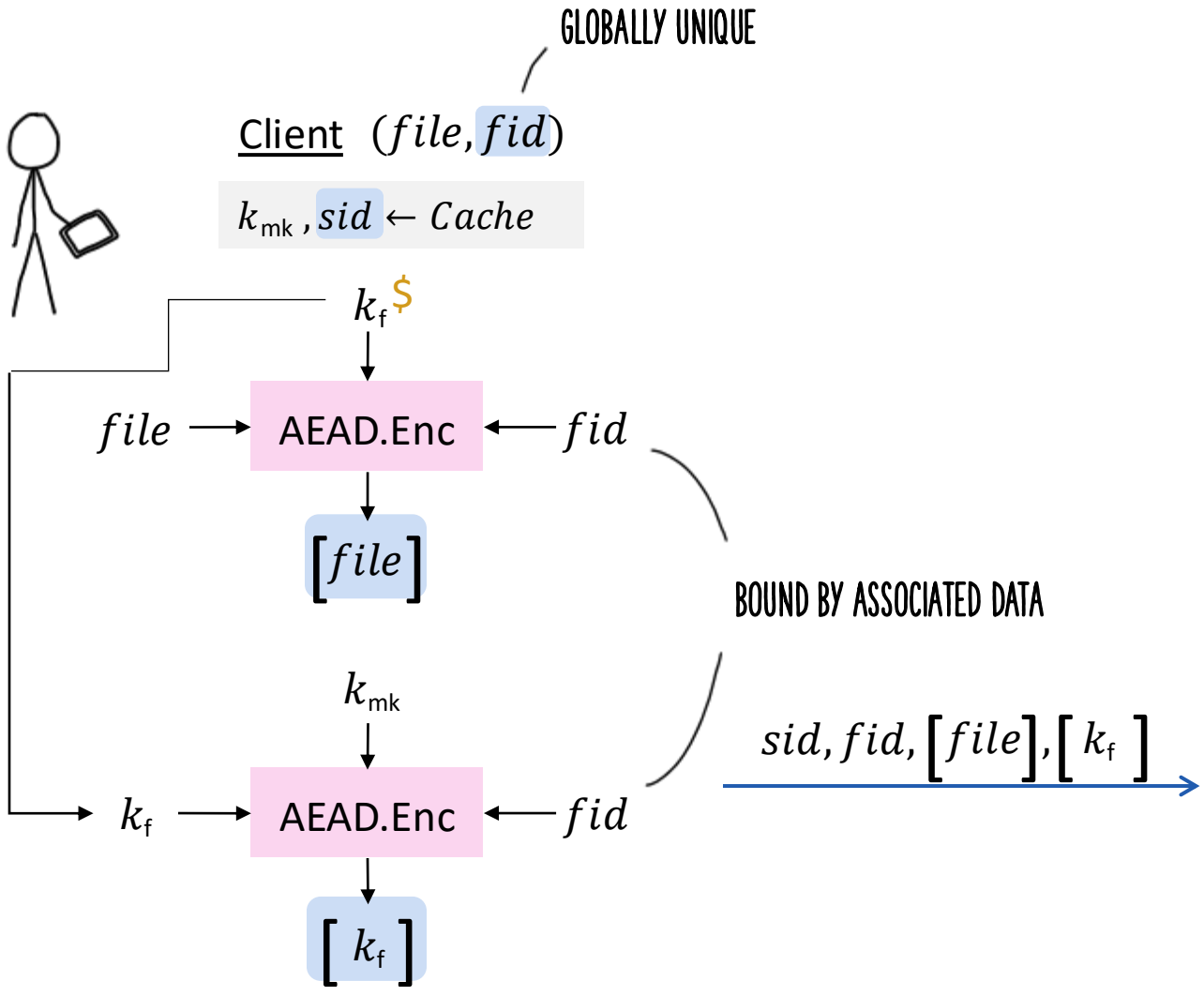
$Session[sid] \leftarrow aid$

STORE

$Cache \leftarrow k_{mk}, sid$

CSS (Cloud Storage Scheme)

Put



Server

FETCH

$aid \leftarrow Session[sid]$

STORE

$File[fid] \leftarrow [file]$ — SHARED

$Key[aid, fid] \leftarrow [k_f]$ — UNIQUE PER USER

CSS (Cloud Storage Scheme)

Share

*SIMPLIFIED

RECIPIENT ACCOUNT ID

Client $(fid, raid)$

$k_{mk}, sid \leftarrow Cache$

Server

$sid, fid, raid$

FETCH

$aid \leftarrow Session[sid]$

$[k_f] \leftarrow Key[aid, fid]$

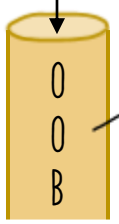
$[k_f]$

k_{mk}

$[k_f]$ → AEAD.Dec ← fid

k_f

SEND TO: $raid$

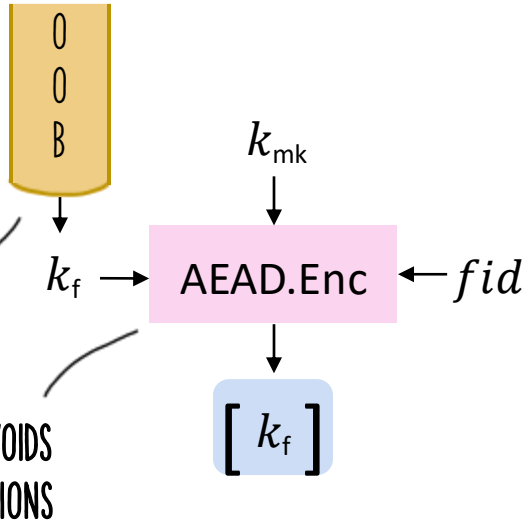


CSS (Cloud Storage Scheme)

Accept *SIMPLIFIED

Client (fid)

$k_{mk}, sid \leftarrow Cache$



$sid, fid, [k_f]$

Server

FETCH

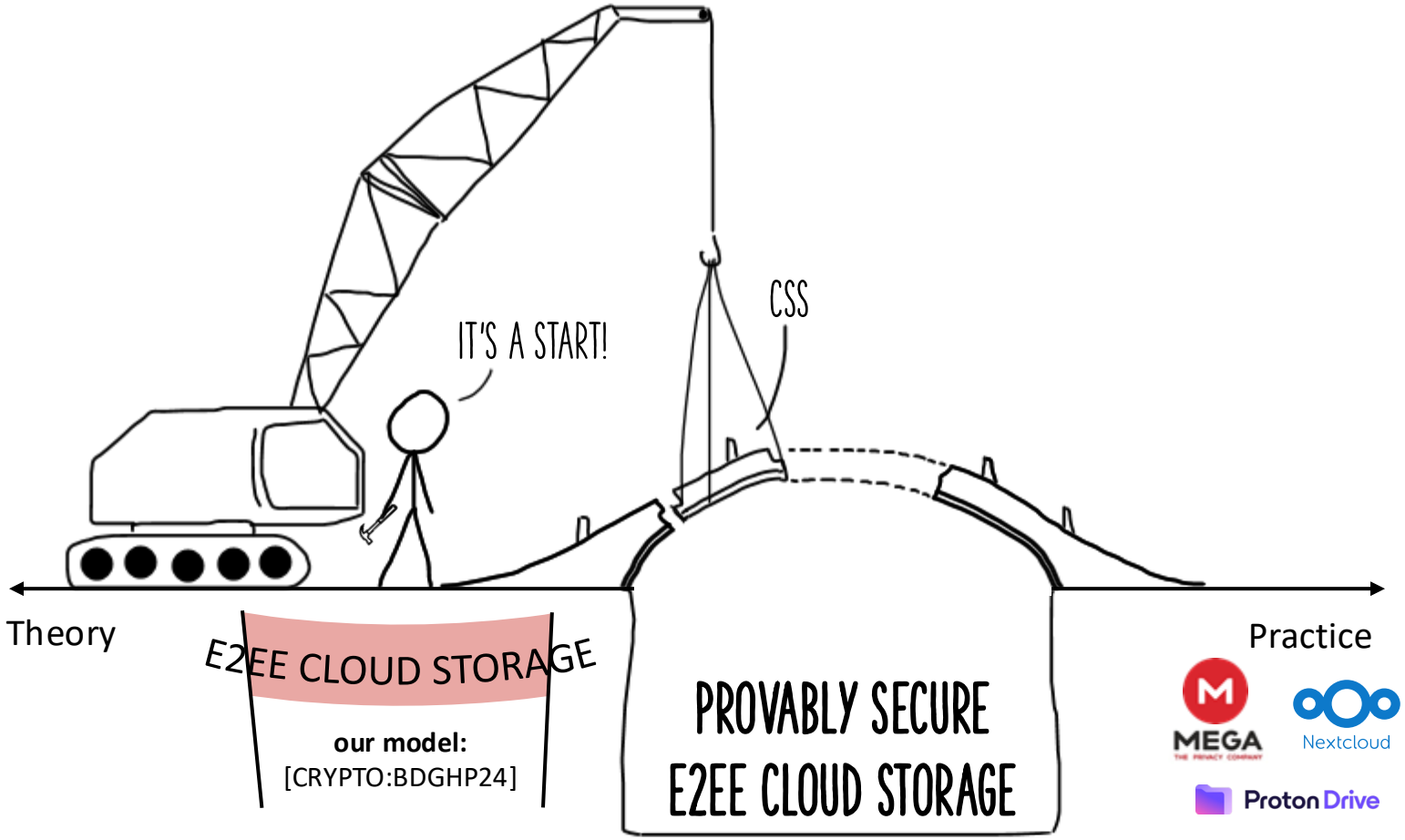
$aid \leftarrow Session[sid]$

STORE

$Key[aid, fid] \leftarrow [k_f]$

Conclusion

- E2EE cloud storage in practice:
  - Plagued by attacks
- E2EE cloud storage in theory
  - Novel security notions [BDGHP24]
  - CSS
- Future work
  - Adaptive security proof
  - Large-scale deployment
  - Prove existing E2EE cloud storage secure



A Formal Treatment of End-to-End Encrypted Cloud Storage

Matilda Backendal, Hannah Davis, Felix Günther, Miro Haller, Kenny Paterson
mbackendal@inf.ethz.ch mhaller@ucsd.edu

eprint.iacr.org/2024/989