Accountable authentication with privacy protection: The Larch system for universal login

Emma Dauterman
UC Berkeley

Danny Lin
Woodinville High School

Henry Corrigan-Gibbs
MIT CSAIL

David Mazières
Stanford

OSDI 2023

ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL

ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

Challenging to determine extent of compromise

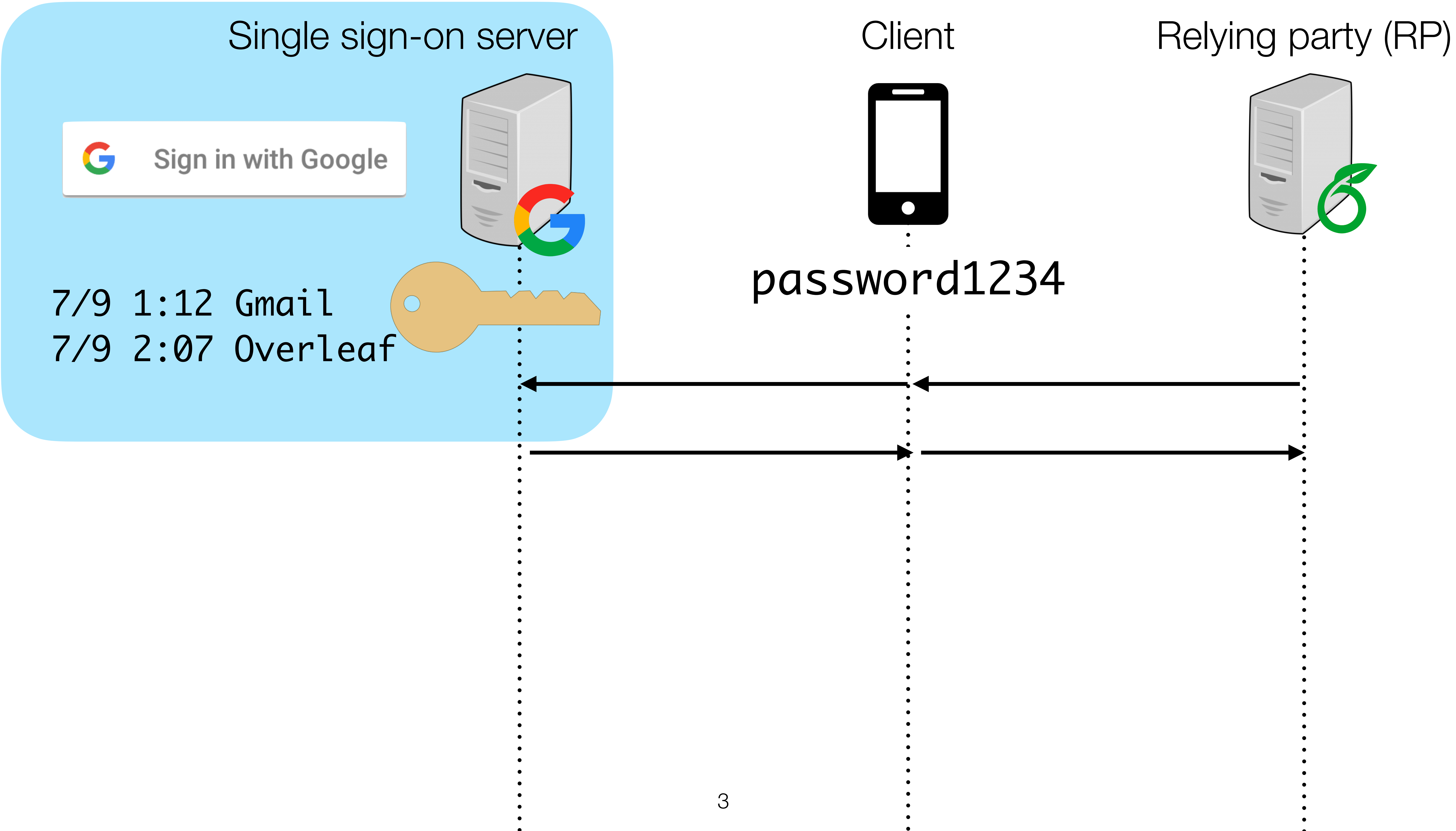
Challenging to determine extent of compromise

LastPass' latest data breach exposed some customer information

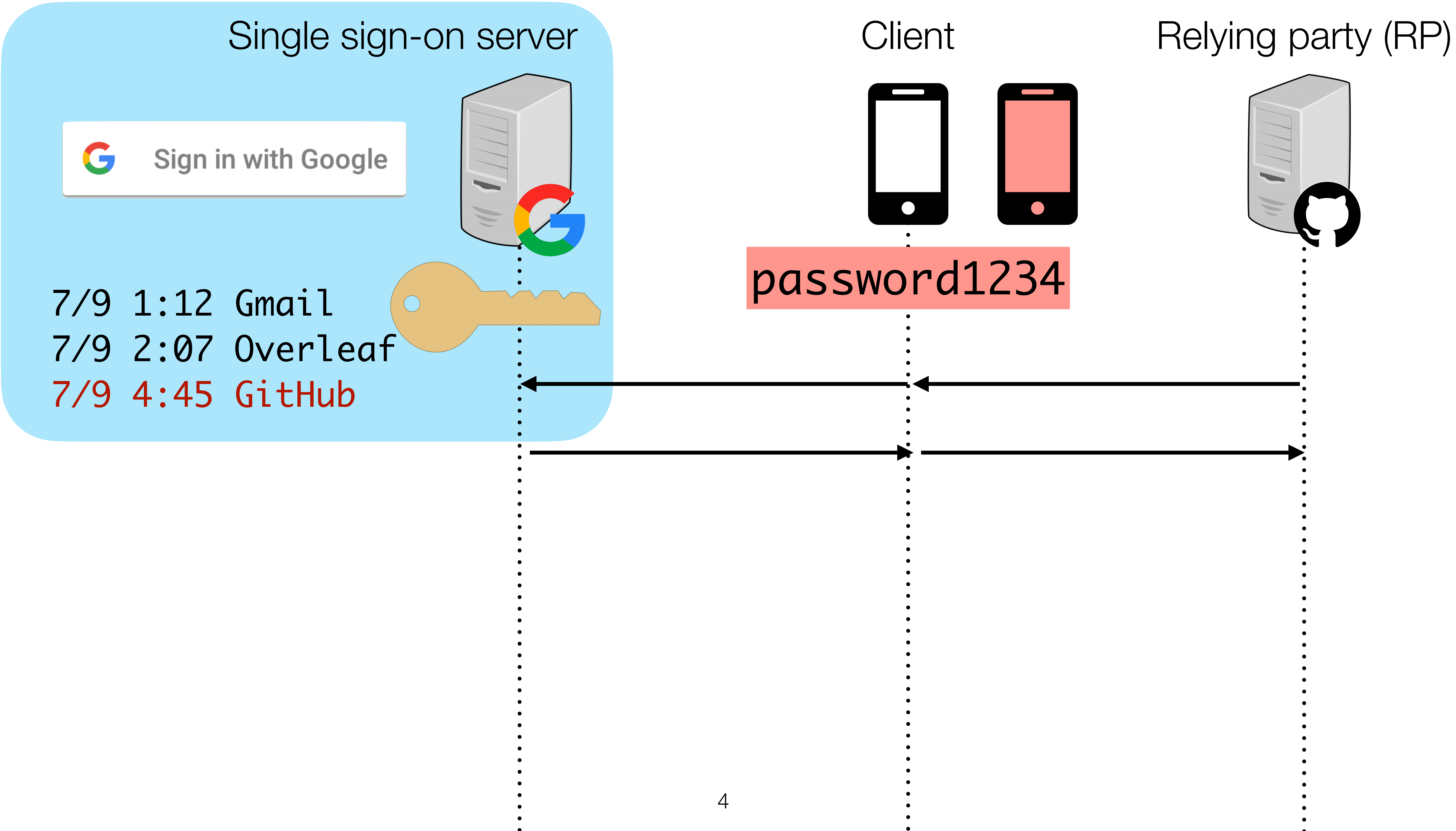
Okta ends Lapsus$ hack investigation, says breach lasted just 25 minutes

Data Breach Could Compromise Lawmakers' Personal Information

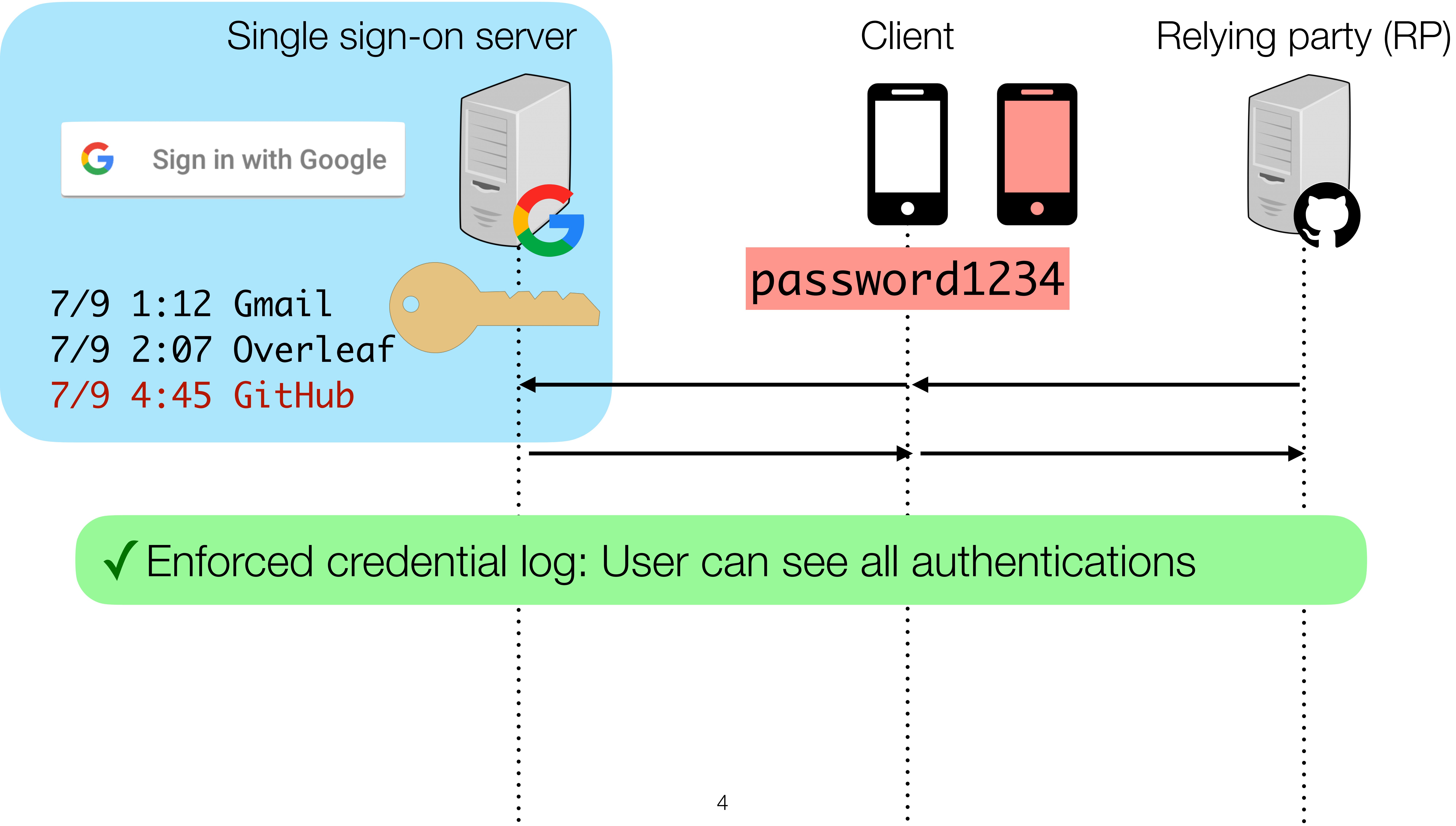
Single sign-on enforces credential logging



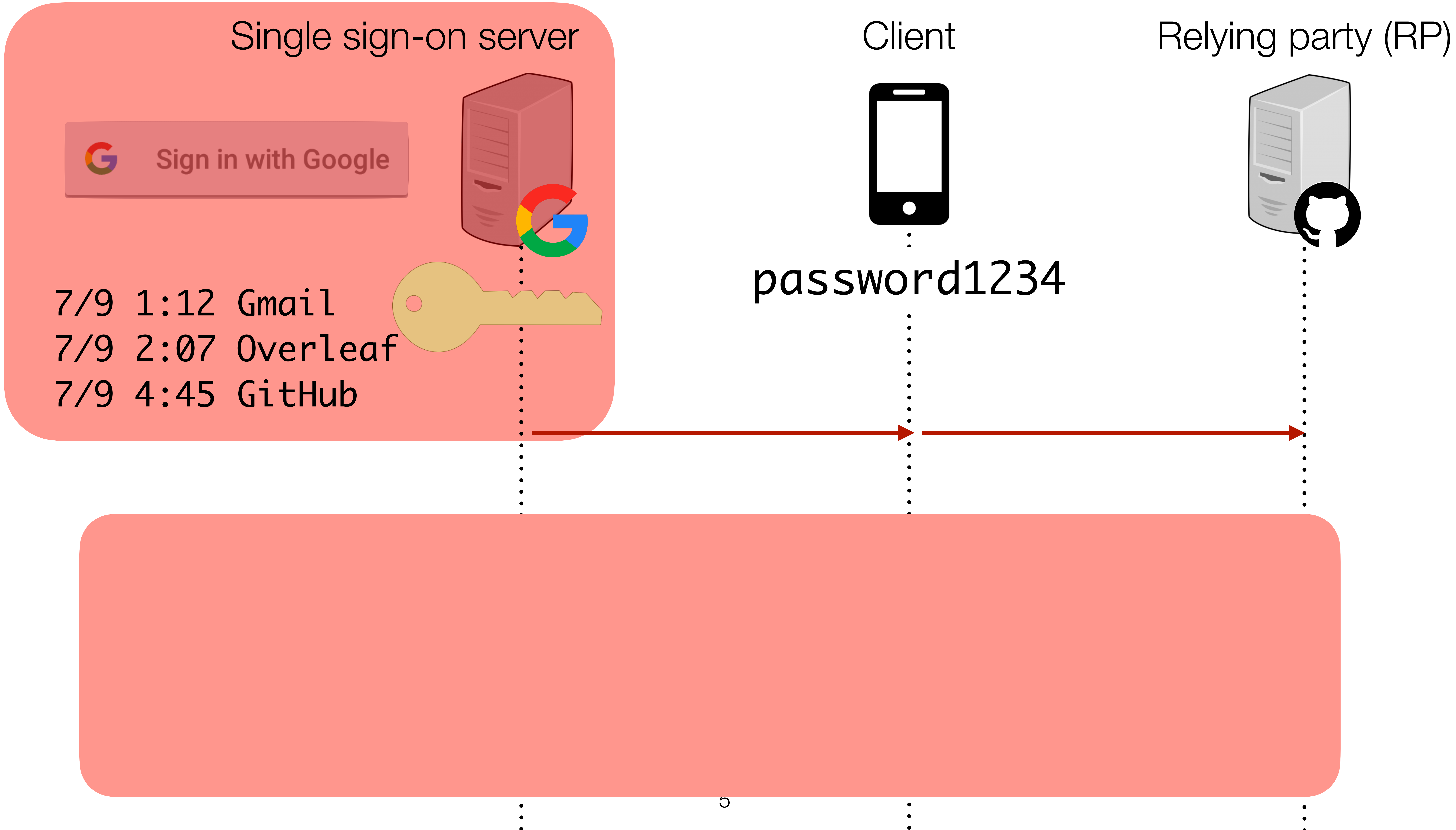
Single sign-on enforces credential logging



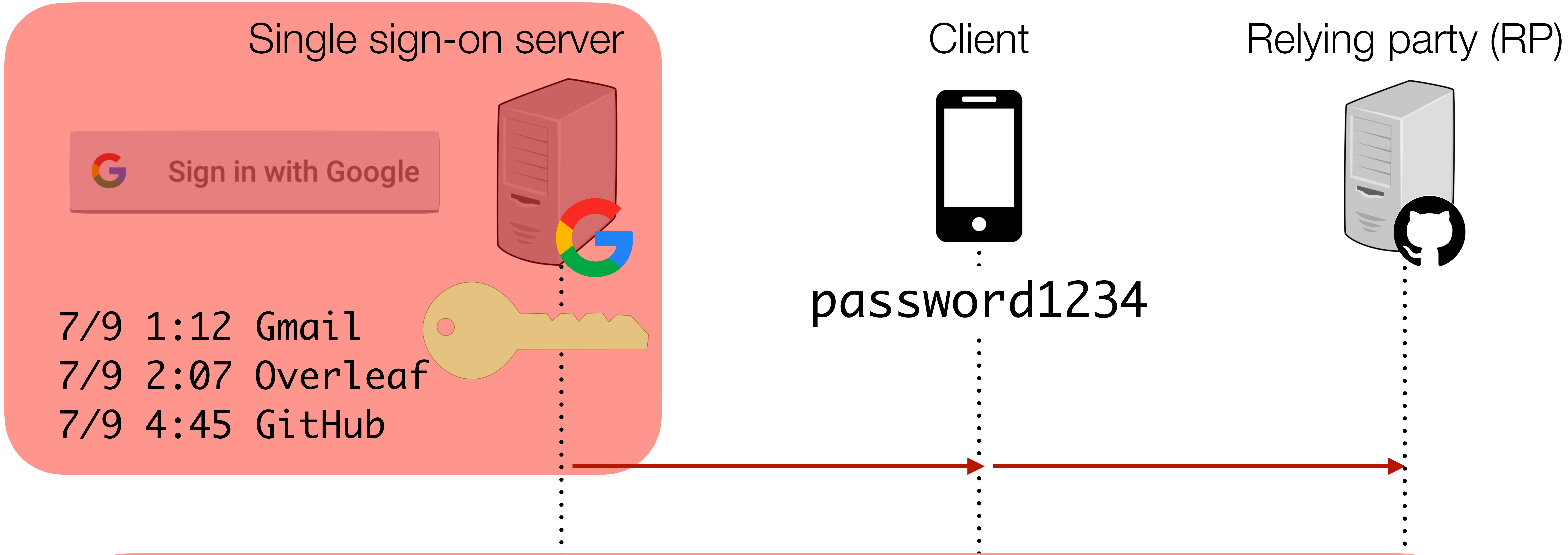
Single sign-on enforces credential logging



Single sign-on: single point of security failure



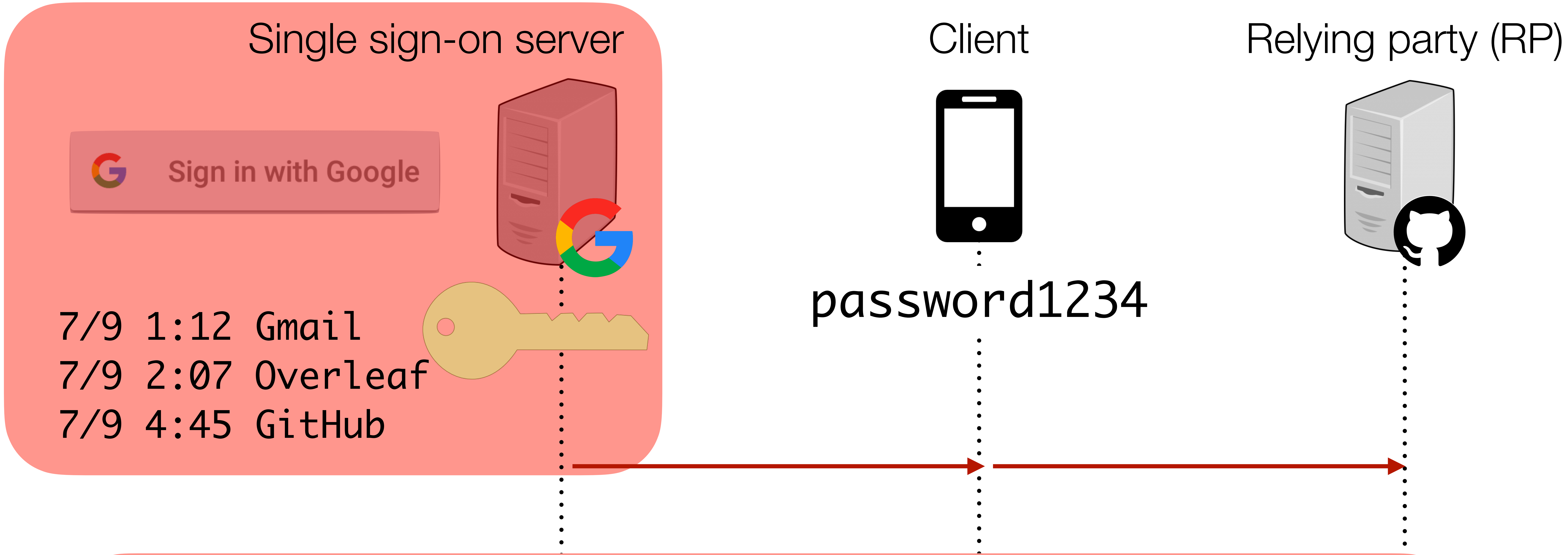
Single sign-on: single point of security failure



✓ Enforced credential log: User can see all authentications

✗ Security: Attacker can access user's accounts

Single sign-on: single point of security failure

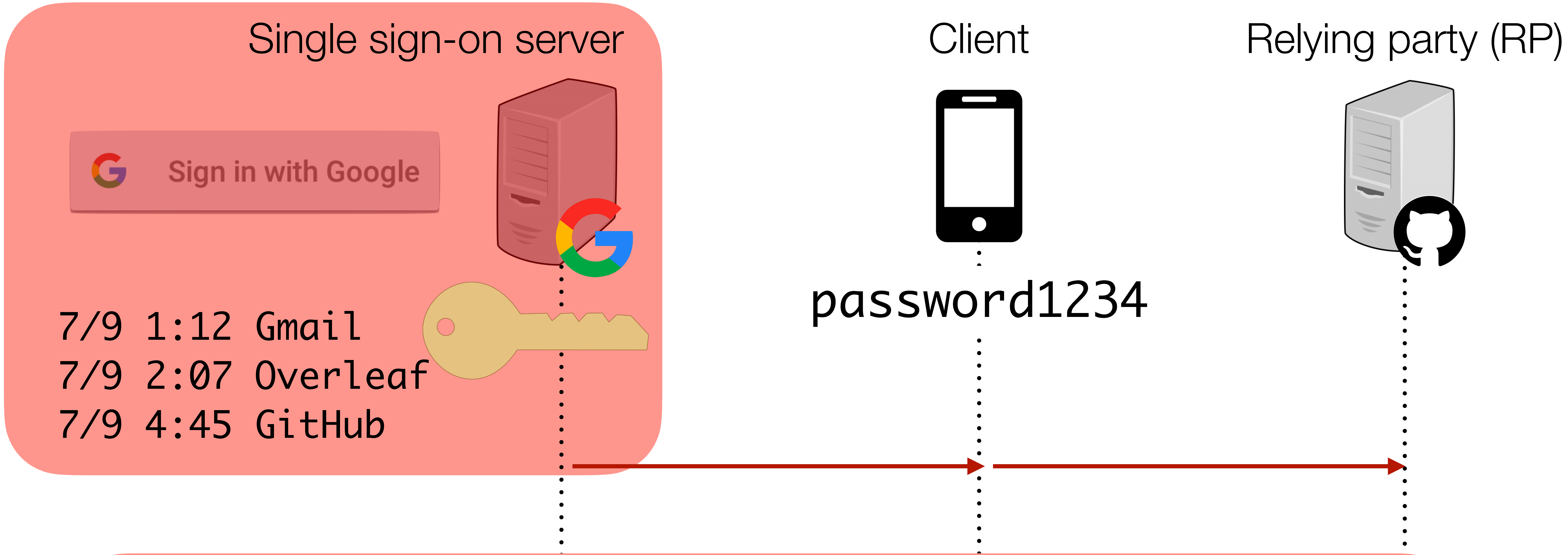


✓ Enforced credential log: User can see all authentications

✗ Security: Attacker can access user's accounts

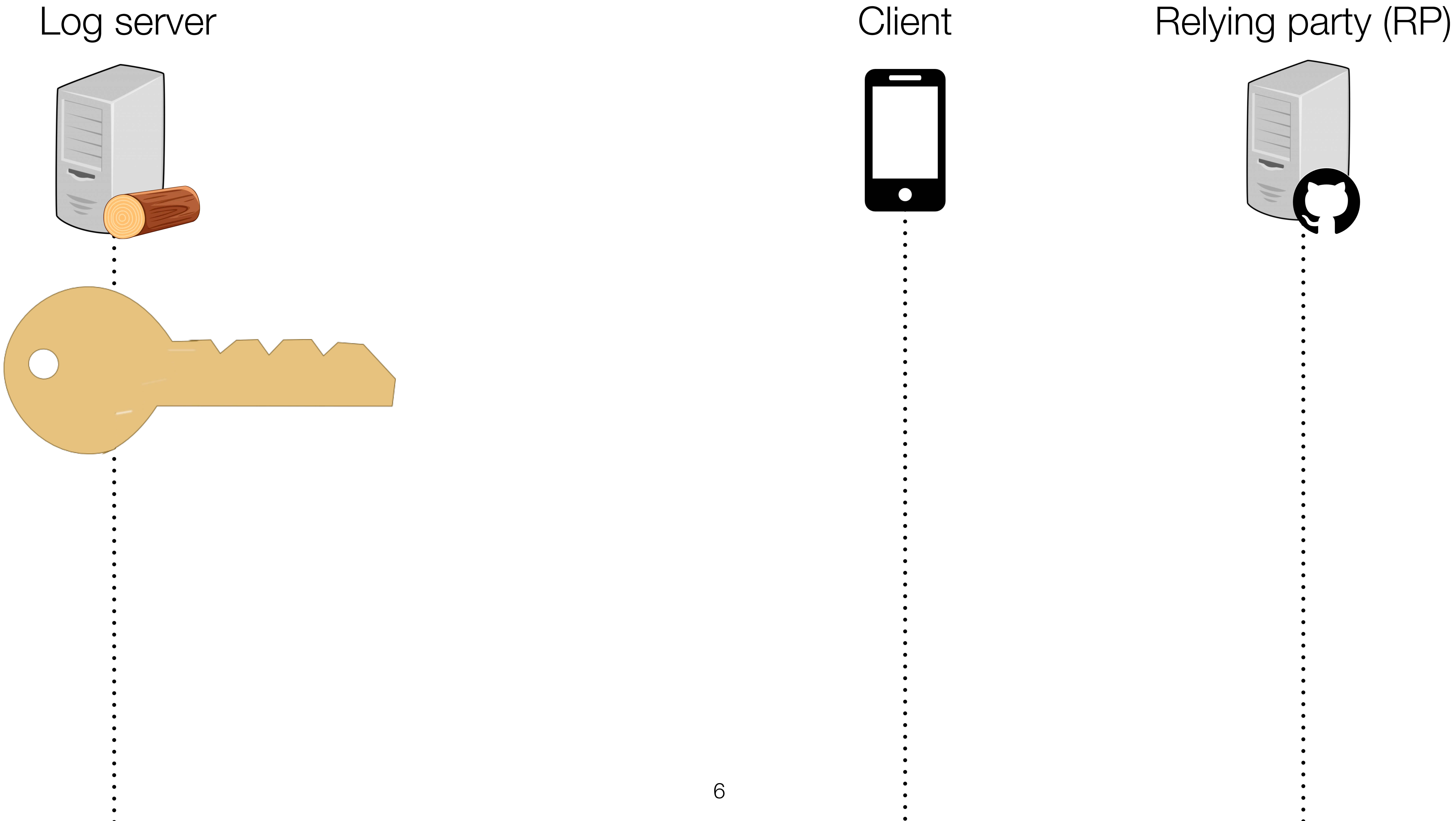
✗ Privacy: Attacker (and legitimate server) can read credential log

Single sign-on: single point of security failure

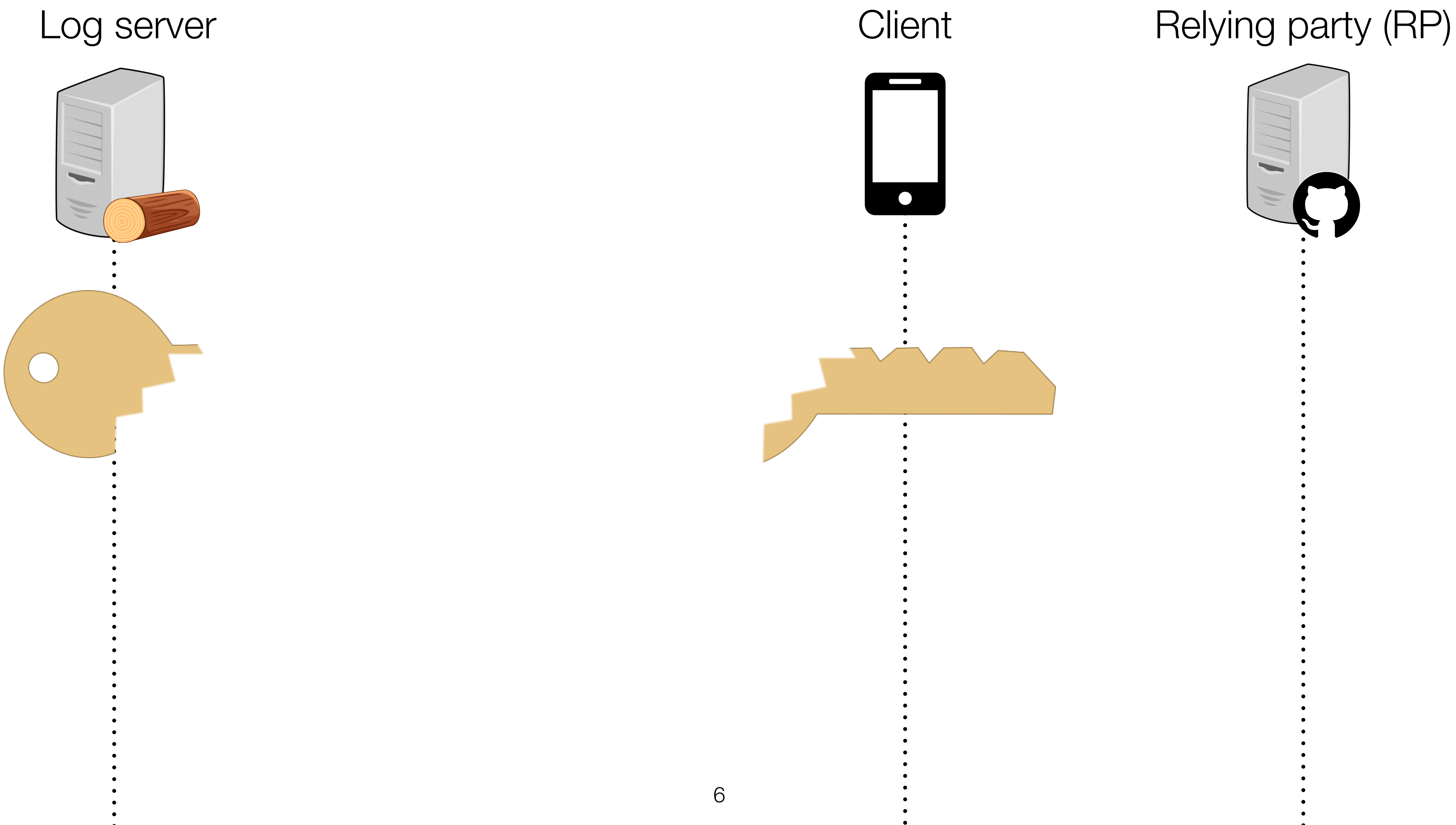


✓ Enforced credential log: User can see all authentications
✗ Security: Attacker can access user's accounts
✗ Privacy: Attacker (and legitimate server) can read credential log
✗ Universal support: Not supported by all RPs

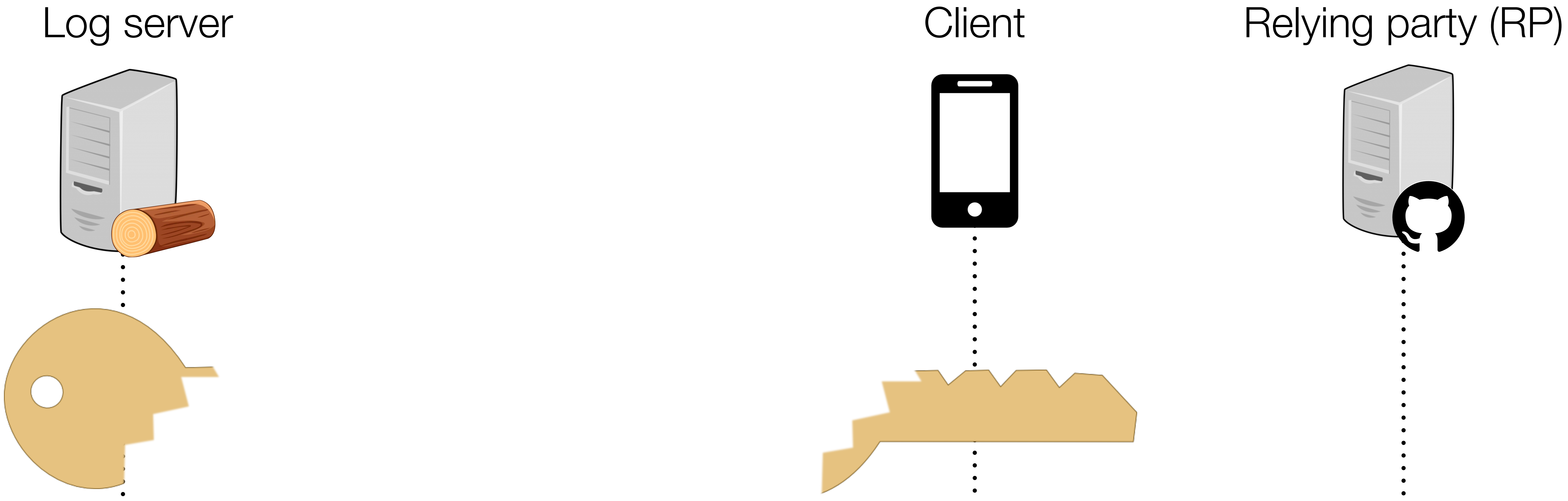
Larch: Split secret key between client and log



Larch: Split secret key between client and log

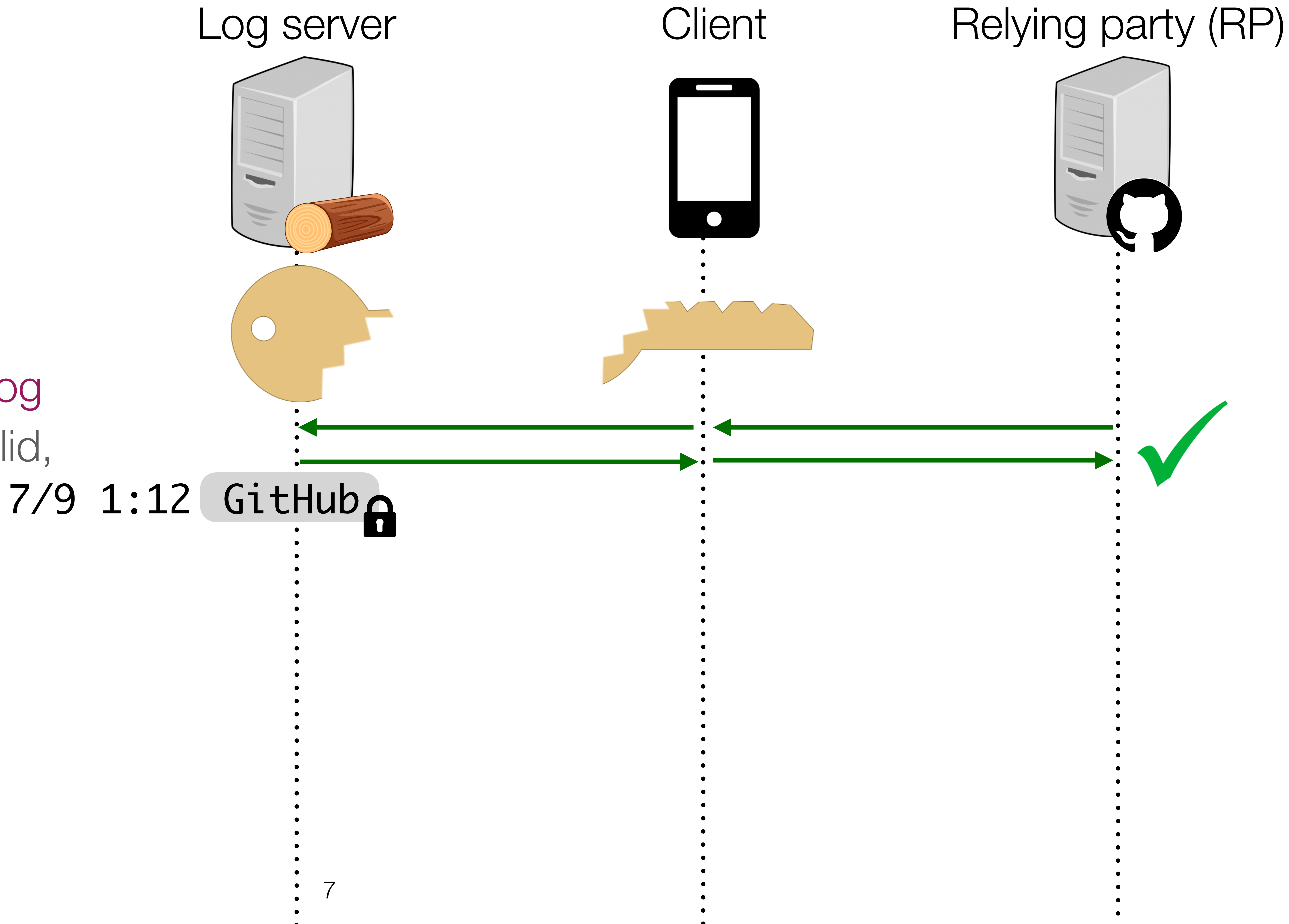


Larch: Split secret key between client and log



- ✓ Enforced credential log
- ✓ Security
- ✓ Privacy
- ✓ Universal support

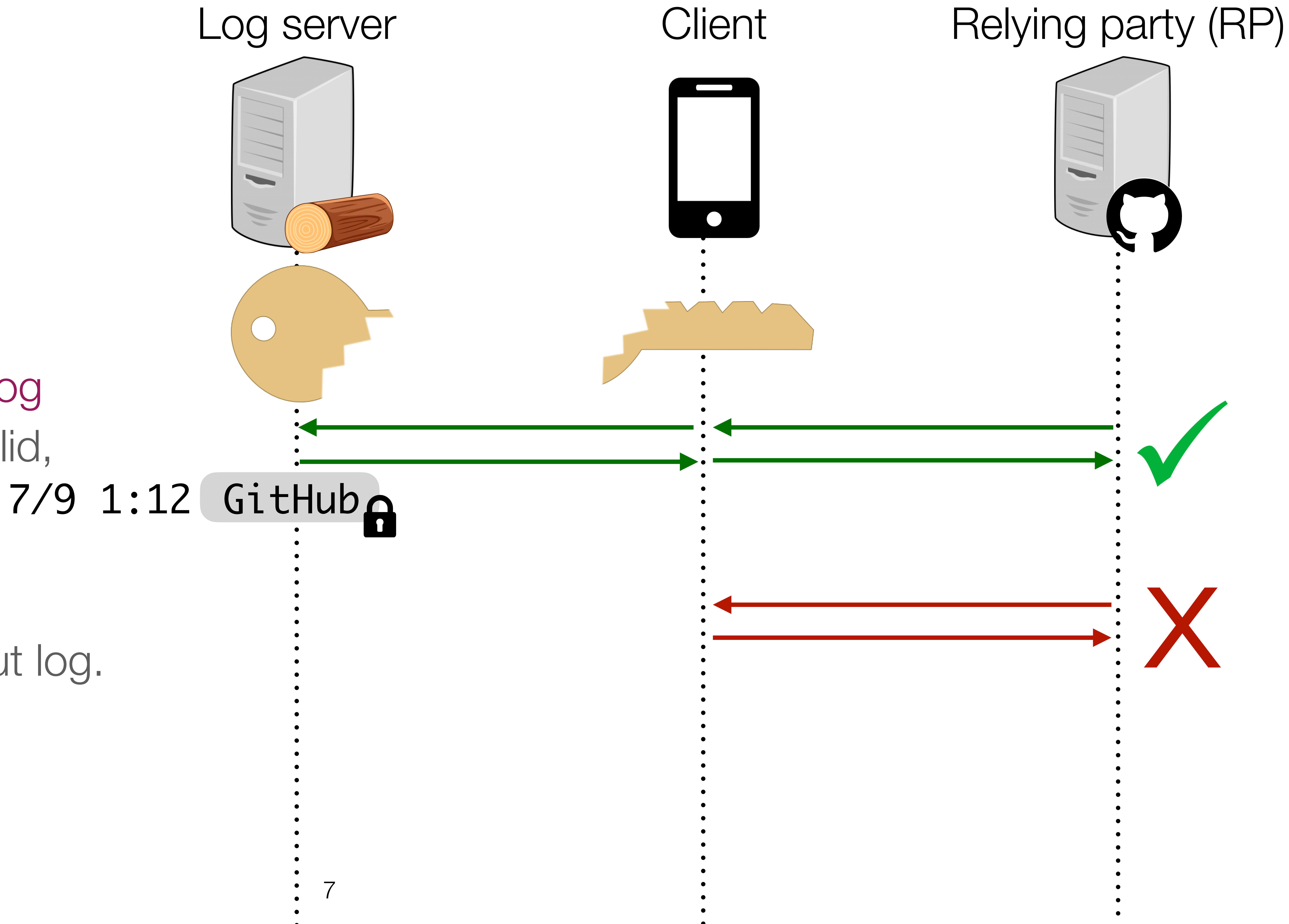
Larch: Enforced credential log with strong security



Enforced, **encrypted** credential log

Client authentications result in valid, encrypted log records.

Larch: Enforced credential log with strong security



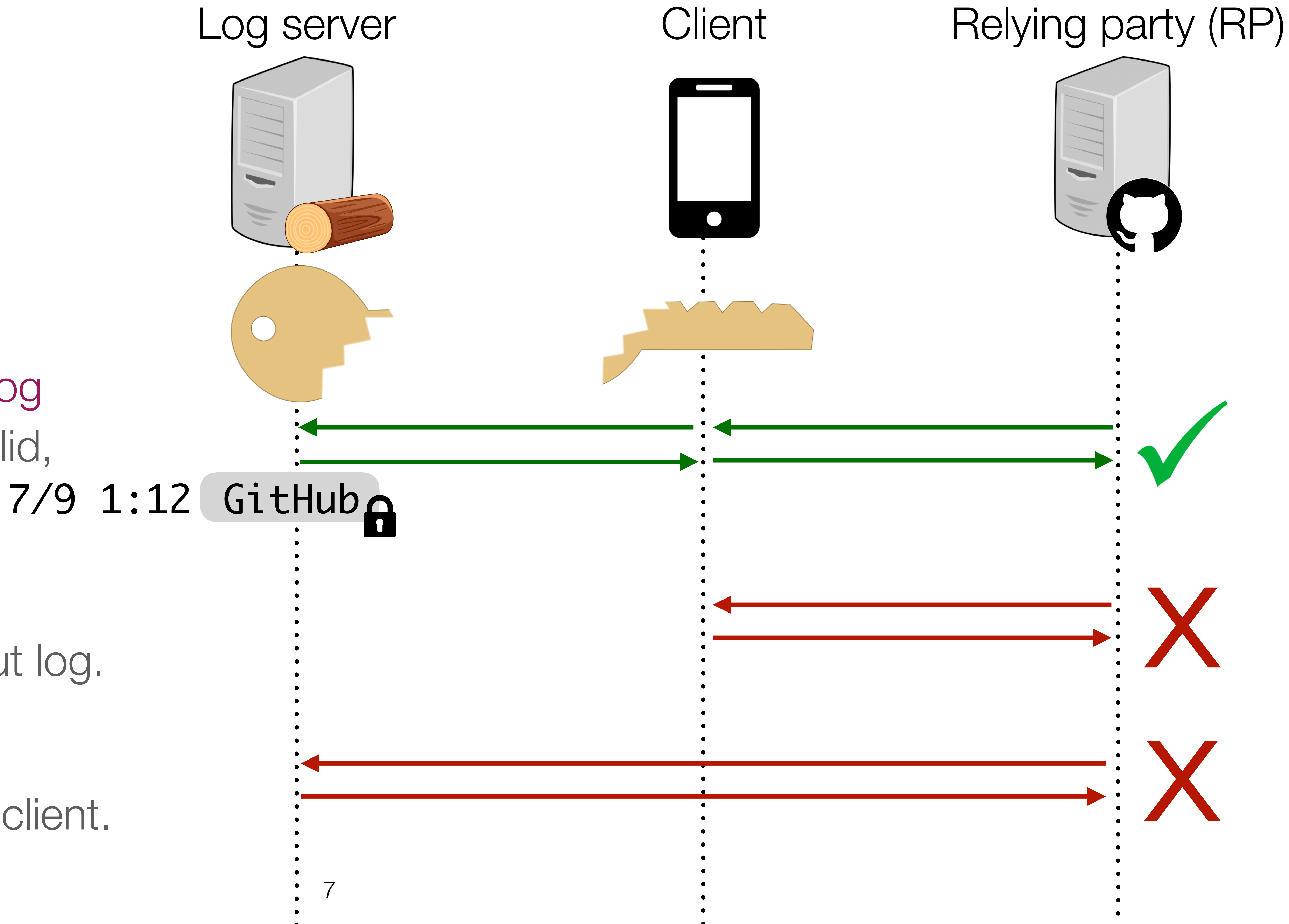
Enforced, **encrypted** credential log

Client authentications result in valid, encrypted log records.

Protect against malicious client

Client cannot authenticate without log.

Larch: Enforced credential log with strong security



Enforced, **encrypted** credential log

Client authentications result in valid, encrypted log records.

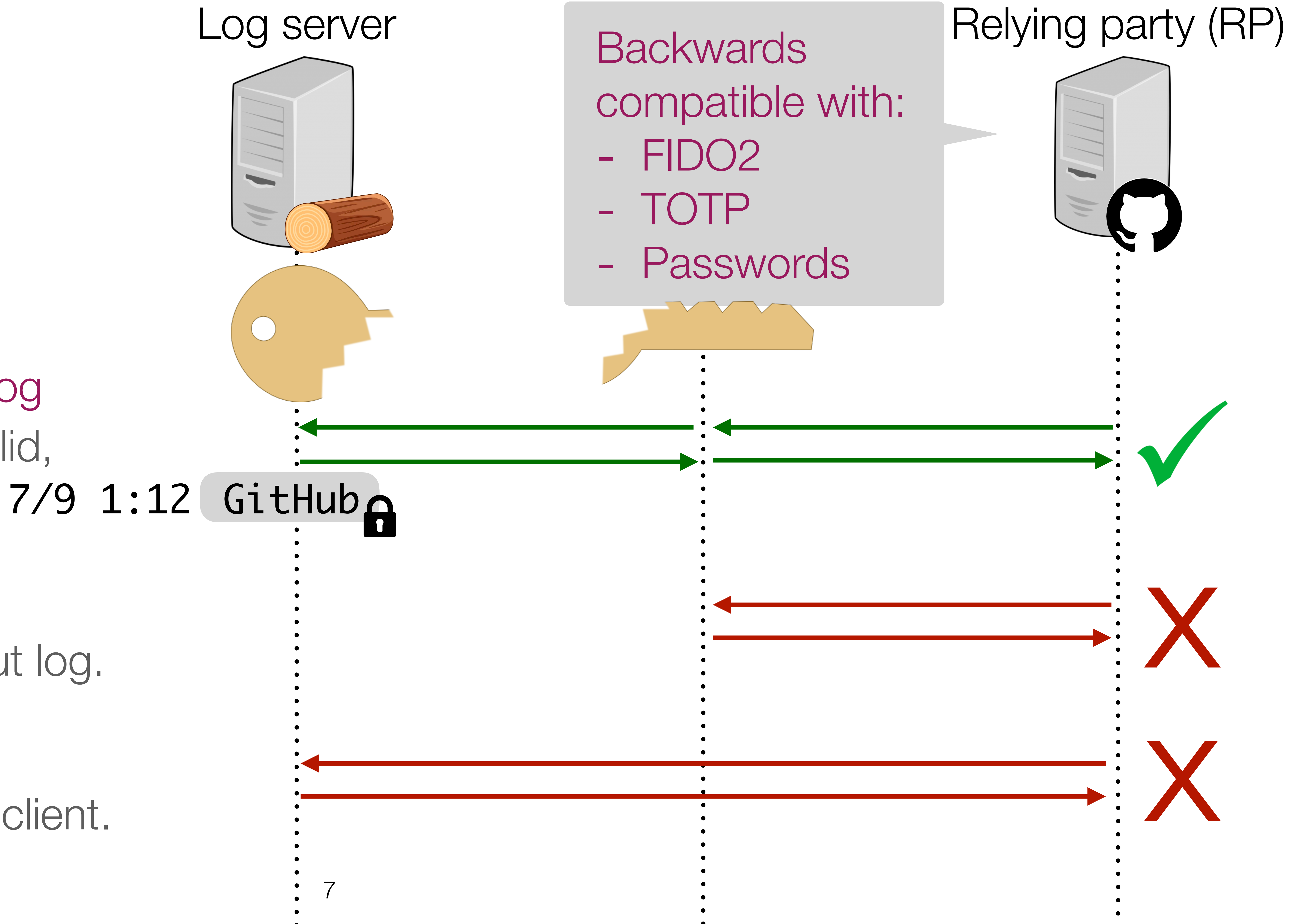
Protect against malicious client

Client cannot authenticate without log.

Protect against malicious log

Log cannot authenticate without client.

Larch: Enforced credential log with strong security



Enforced, **encrypted** credential log

Client authentications result in valid, encrypted log records.

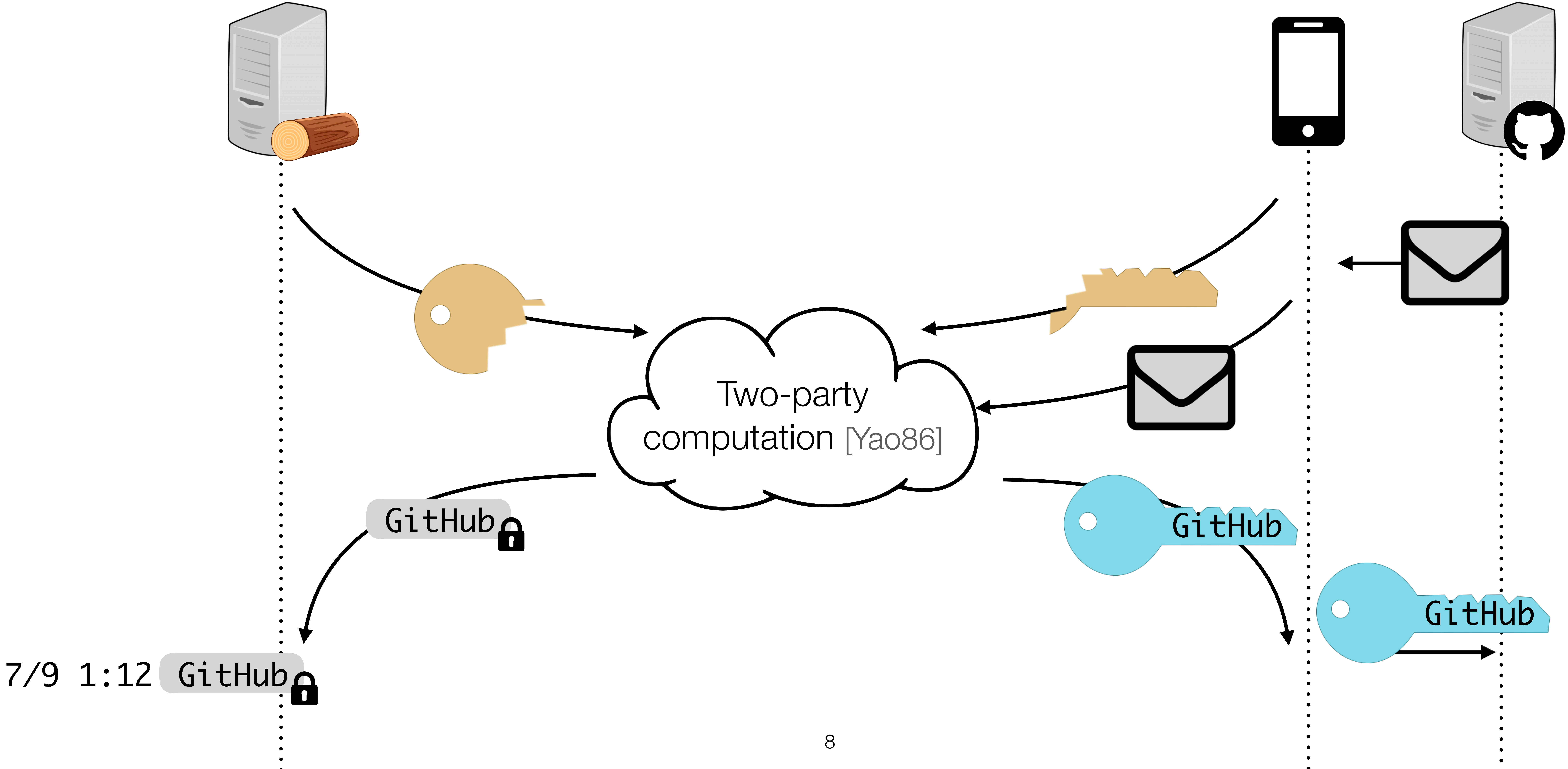
Protect against malicious client

Client cannot authenticate without log.

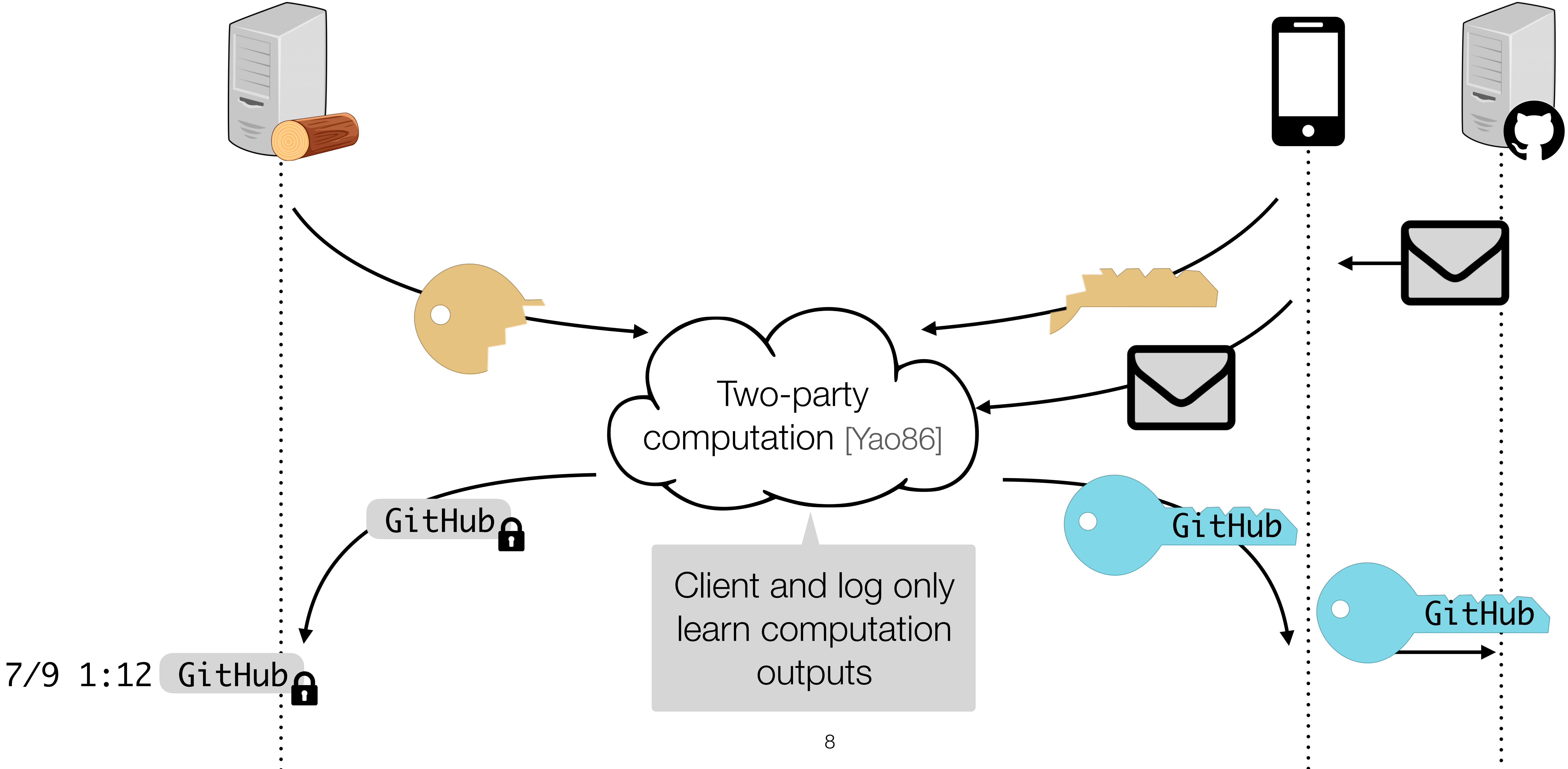
Protect against malicious log

Log cannot authenticate without client.

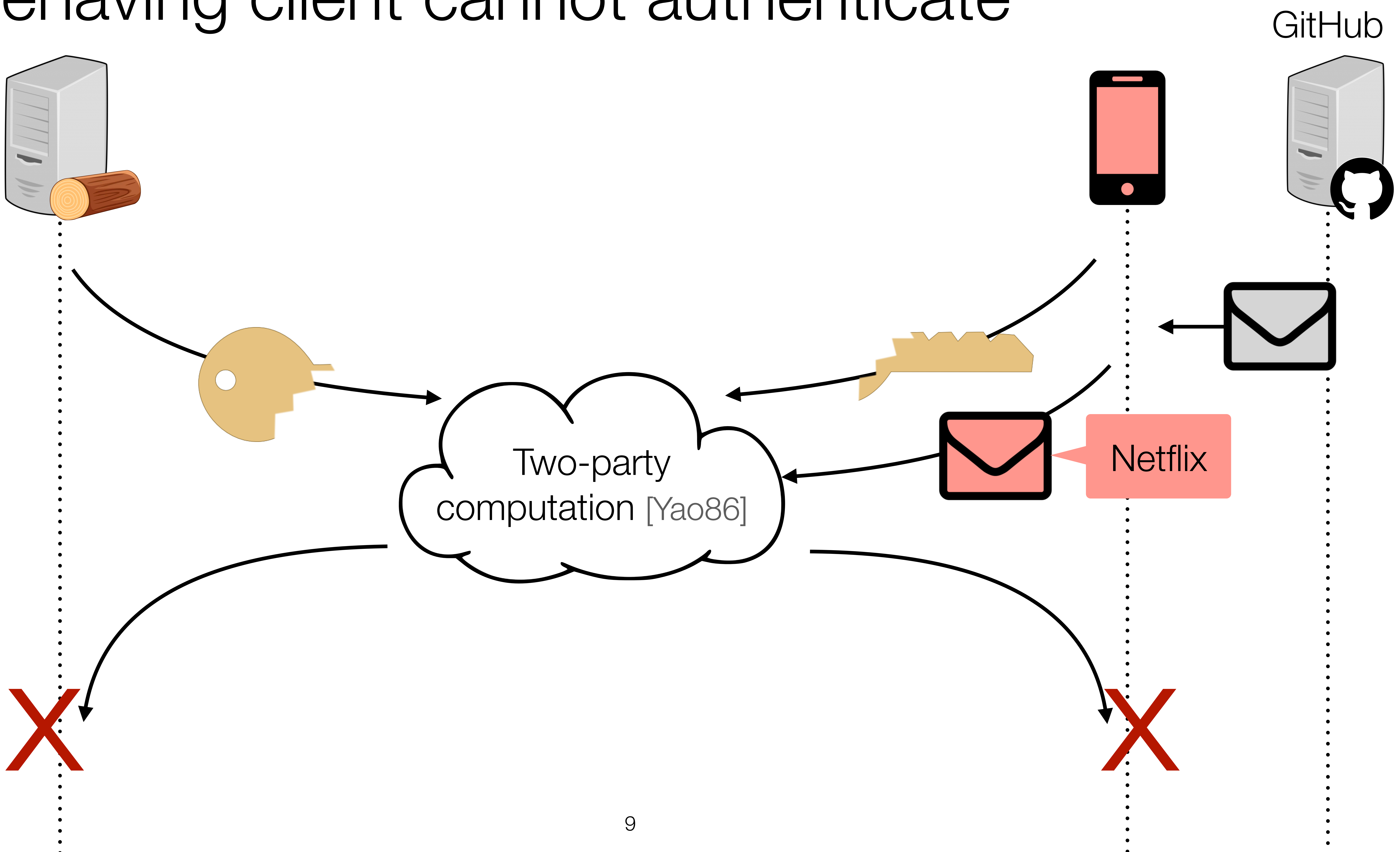
If client authenticates, log gets encrypted log record



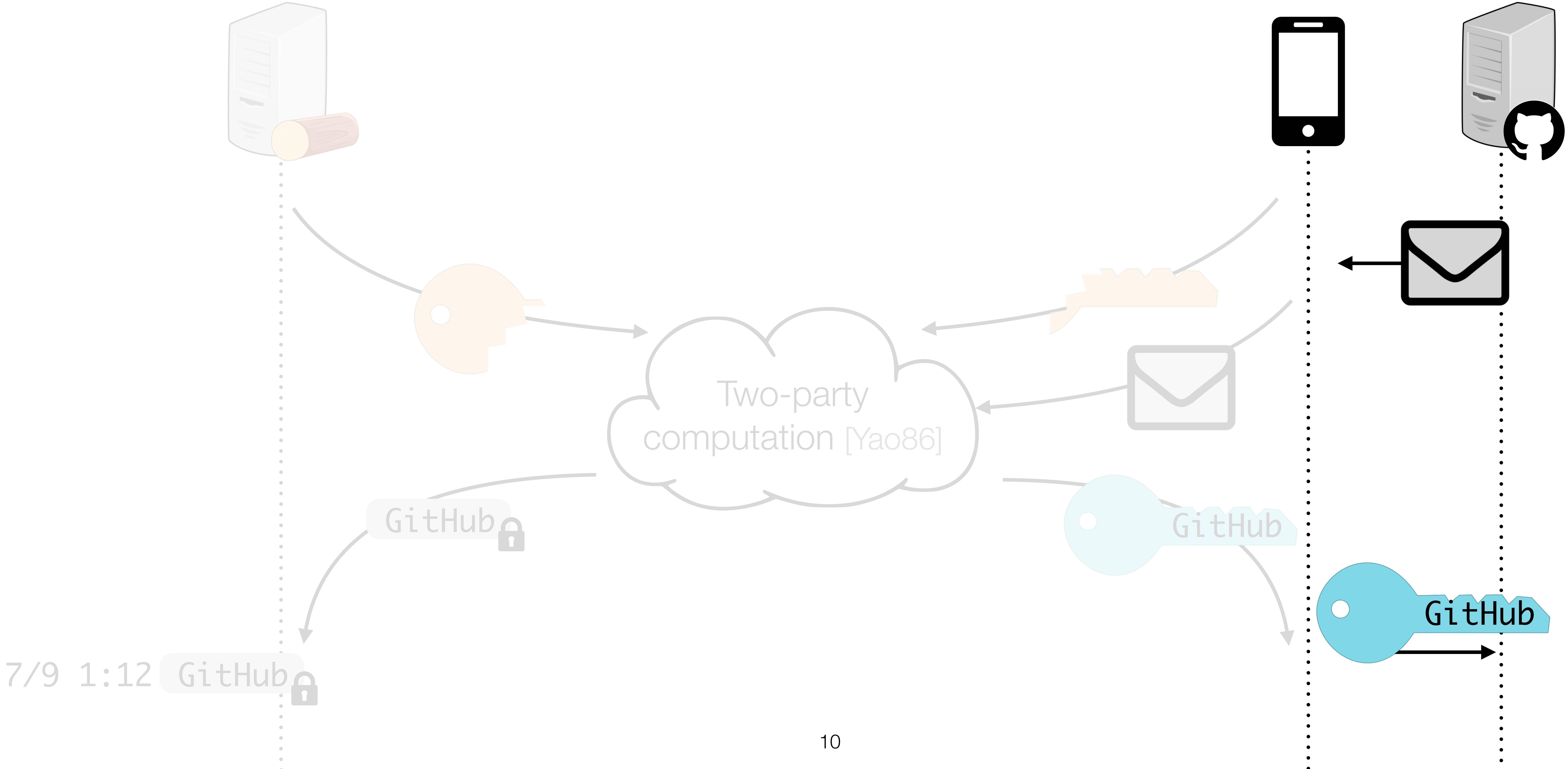
If client authenticates, log gets encrypted log record



Misbehaving client cannot authenticate



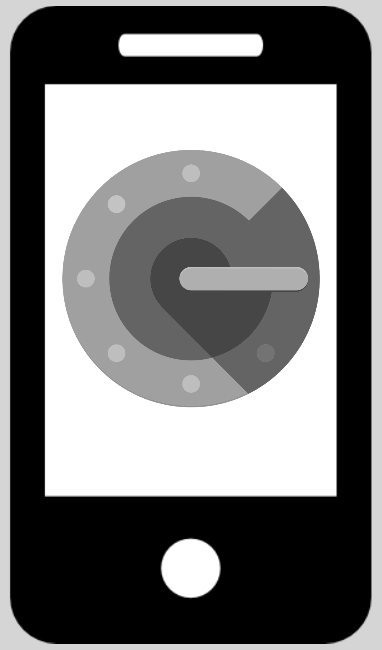
Relying party is unaware client is running larch



Larch is compatible with relying parties running:

FIDO2
(this talk)

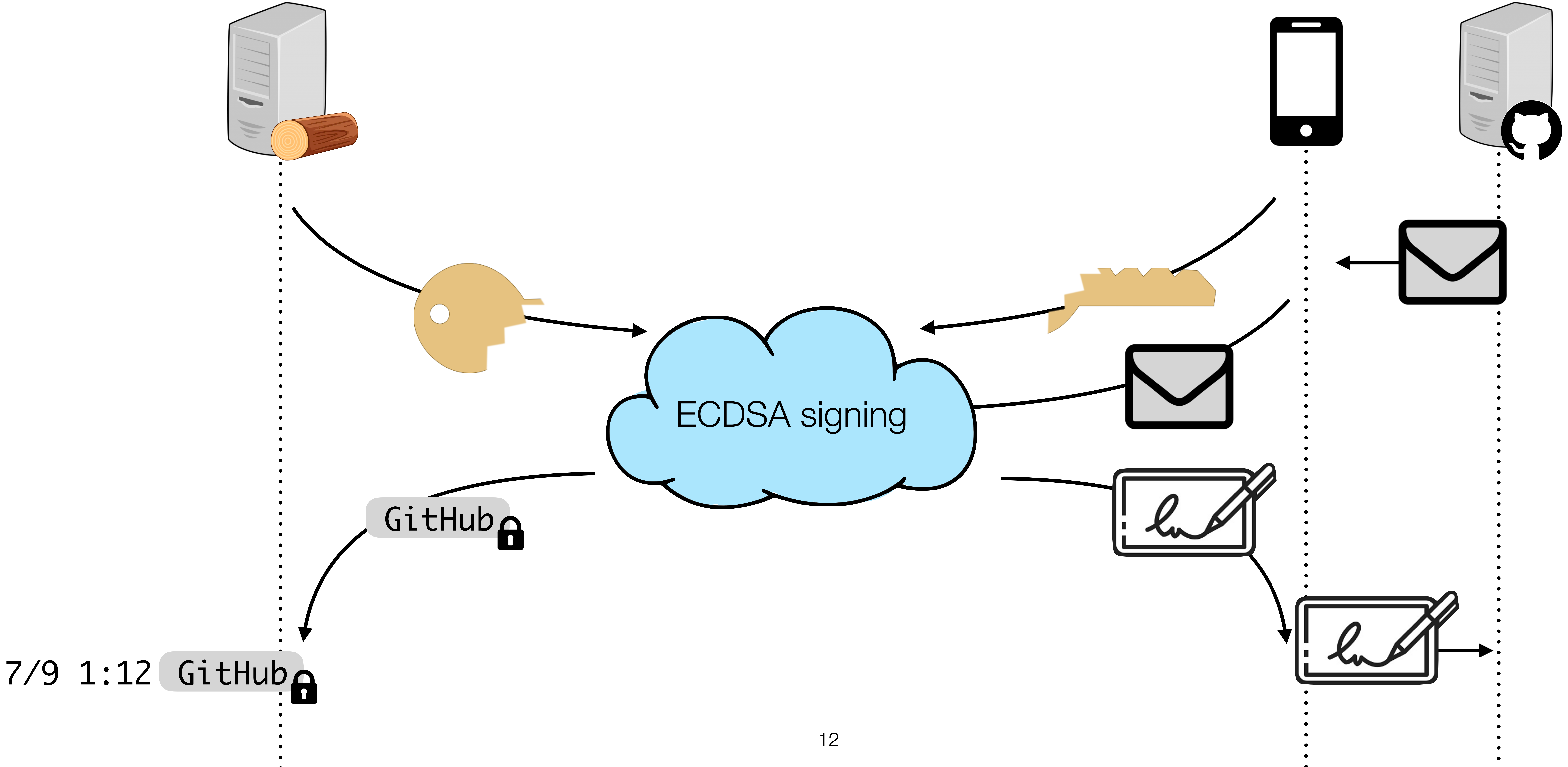
TOTP
(see paper)



Passwords
(see paper)

67bZ!9g92&

Larch for FIDO2



Larch for FIDO2

ECDSA threshold signing

- Extensive prior work with high costs [GGN16, Lindell17, DKLS18, GG18, CGG+20, DJN+20, GS21, ANO+22, …]
- Idea: take advantage of fact that client is honest at enrollment for *precomputation*

7/9 1:12 GitHub

Larch for FIDO2

ECDSA threshold signing

- Extensive prior work with high costs [GGN16, Lindell17, DKLS18, GG18, CGG+20, DJN+20, GS21, ANO+22, …]
- Idea: take advantage of fact that client is honest at enrollment for *precomputation*

To sign a message m with signing nonce r,
compute $f_1(r) \cdot (m + f_2(r) \cdot \mathsf{sk})$

7/9 1:12 GitHub

Larch for FIDO2

ECDSA threshold signing

- Extensive prior work with high costs [GGN16, Lindell17, DKLS18, GG18, CGG+20, DJN+20, GS21, ANO+22, …]
- Idea: take advantage of fact that client is honest at enrollment for *precomputation*

To sign a message m with signing nonce r,

compute $f_1(r) \cdot (m + f_2(r) \cdot \mathsf{sk})$

Precompute at enrollment

7/9 1:12 GitHub

Evaluation

ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL

ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

Code available at: `https://github.com/edauterman/larch`

Experiment setup:

- Log server on c5.4xlarge (8 cores, 32 GiB memory)
- Client on c5.2xlarge (4 cores, 16 GiB memory)
- 20ms RTT
- Bandwidth 100Mbps
- TOTP with 20 accounts; passwords with 128 accounts
- Do not include network latency between client and RP in measurements

Evaluation

ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL

ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

	FIDO2	**TOTP**	**Password**
Online auth time	150 ms	91 ms	74 ms
Total auth time	150 ms	1.32 s	74 ms
Online auth comm.	1.73 MiB	201 KiB	3.25 KiB
Total auth comm.	1.73 MiB	65 MiB	3.25 KiB
Log auths/core/s	6.18	0.73	47.62

Evaluation

ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL

ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

		General-purpose two-party computation	
	FIDO2	**TOTP**	**Password**
Online auth time	150 ms	91 ms	74 ms
Total auth time	150 ms	1.32 s	74 ms
Online auth comm.	1.73 MiB	201 KiB	3.25 KiB
Total auth comm.	1.73 MiB	65 MiB	3.25 KiB
Log auths/core/s	6.18	0.73	47.62

Evaluation

ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL

ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

Special-purpose protocols

	FIDO2	**TOTP**	**Password**
Online auth time	150 ms	91 ms	74 ms
Total auth time	150 ms	1.32 s	74 ms
Online auth comm.	1.73 MiB	201 KiB	3.25 KiB
Total auth comm.	1.73 MiB	65 MiB	3.25 KiB
Log auths/core/s	6.18	0.73	47.62

Credential compromise will happen

✓ Enforced credential log: easy to determine extent of account compromise

✓ Security: log cannot access user's accounts

✓ Privacy: log records are encrypted

✓ Universal support: compatible with unmodified relying parties

Key idea: splitting authentication secret between client and log

Moving forward: need tools to make it easier to recover from compromise

Emma Dauterman

edauterman@berkeley.edu

https://arxiv.org/pdf/2305.19241.pdf

https://github.com/edauterman/larch