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
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 server

Client

Relying party (RP)

7/9 1:12 Gmail
7/9 2:07 Overleaf

password1234
Single sign-on enforces credential logging

**Single sign-on server**

- 7/9 1:12 Gmail
- 7/9 2:07 Overleaf
- 7/9 4:45 GitHub

**Client**

- password1234

**Relying party (RP)**
Single sign-on enforces credential logging

Single sign-on server

Client

Relying party (RP)

✓ Enforced credential log: User can see all authentications

7/9 1:12 Gmail
7/9 2:07 Overleaf
7/9 4:45 GitHub
Single sign-on: single point of security failure

Single sign-on server

Client

Relying party (RP)

password1234

7/9 1:12 Gmail
7/9 2:07 Overleaf
7/9 4:45 GitHub

Client

Relying party (RP)
Single sign-on: single point of security failure

- Single sign-on server
  - Sign in with Google
  - 7/9 1:12 Gmail
  - 7/9 2:07 Overleaf
  - 7/9 4:45 GitHub

- Client
  - password1234

- Relying party (RP)

- ✓ Enforced credential log: User can see all authentications
- X 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

- Single sign-on server
- Client
- Relying party (RP)

7/9 1:12 Gmail
7/9 2:07 Overleaf
7/9 4:45 GitHub

✓ 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

Password: password1234
Larch: Split secret key between client and log

Log server

Client

Relying party (RP)
Larch: Split secret key between client and log

Log server

Client

Relying party (RP)
Larch: Split secret key between client and log

- Client
- Relying party (RP)
- Log server

✓ 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.
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.

Backwards compatible with:
- FIDO2
- TOTP
- Passwords
If client authenticates, log gets encrypted log record
If client authenticates, log gets encrypted log record

Two-party computation [Yao86]

Client and log only learn computation outputs
Misbehaving client cannot authenticate

Two-party computation [Yao86]
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

ECDSA signing
ECDSA threshold signing

- Extensive prior work with high costs \([\text{GGN16, Lindell17, DKLS18, GG18, CGG+20, DJN+20, GS21, ANO+22, ...}]\)
- Idea: take advantage of fact that client is honest at enrollment for precomputation
Larch for FIDO2

ECDSA threshold signing

- Extensive prior work with high costs \([\text{GGN16, Lindell17, DKLS18, GG18, CGG+20, DJN+20, GS21, ANO+22, ...}]\)

- Idea: take advantage of fact that client is honest at enrollment for precomputation

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

Precompute at enrollment
Evaluation

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

<table>
<thead>
<tr>
<th></th>
<th>FIDO2</th>
<th>TOTP</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online auth time</td>
<td>150 ms</td>
<td>91 ms</td>
<td>74 ms</td>
</tr>
<tr>
<td>Total auth time</td>
<td>150 ms</td>
<td>1.32 s</td>
<td>74 ms</td>
</tr>
<tr>
<td>Online auth comm.</td>
<td>1.73 MiB</td>
<td>201 KiB</td>
<td>3.25 KiB</td>
</tr>
<tr>
<td>Total auth comm.</td>
<td>1.73 MiB</td>
<td>65 MiB</td>
<td>3.25 KiB</td>
</tr>
<tr>
<td>Log auths/core/s</td>
<td>6.18</td>
<td>0.73</td>
<td>47.62</td>
</tr>
</tbody>
</table>
Evaluation

General-purpose two-party computation

<table>
<thead>
<tr>
<th></th>
<th>FIDO2</th>
<th>TOTP</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online auth time</td>
<td>150 ms</td>
<td>91 ms</td>
<td>74 ms</td>
</tr>
<tr>
<td>Total auth time</td>
<td>150 ms</td>
<td>1.32 s</td>
<td>74 ms</td>
</tr>
<tr>
<td>Online auth comm.</td>
<td>1.73 MiB</td>
<td>201 KiB</td>
<td>3.25 KiB</td>
</tr>
<tr>
<td>Total auth comm.</td>
<td>1.73 MiB</td>
<td>65 MiB</td>
<td>3.25 KiB</td>
</tr>
<tr>
<td>Log auths/core/s</td>
<td>6.18</td>
<td>0.73</td>
<td>47.62</td>
</tr>
</tbody>
</table>
Evaluation

Table 6: Costs for larch with FIDO2, TOTP (20 relying parties), and password (128 relying parties). We take the cost of one core on a password hash function should take 0.5 seconds using 2 cores.

<table>
<thead>
<tr>
<th></th>
<th>FIDO2</th>
<th>TOTP</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online auth time</td>
<td>150 ms</td>
<td>91 ms</td>
<td>74 ms</td>
</tr>
<tr>
<td>Total auth time</td>
<td>150 ms</td>
<td>1.32 s</td>
<td>74 ms</td>
</tr>
<tr>
<td>Online auth comm.</td>
<td>1.73 MiB</td>
<td>201 KiB</td>
<td>3.25 KiB</td>
</tr>
<tr>
<td>Total auth comm.</td>
<td>1.73 MiB</td>
<td>65 MiB</td>
<td>3.25 KiB</td>
</tr>
<tr>
<td>Log auths/core/s</td>
<td>6.18</td>
<td>0.73</td>
<td>47.62</td>
</tr>
</tbody>
</table>

Special-purpose protocols

Related work
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://github.com/edauterman/larch