Aalto University

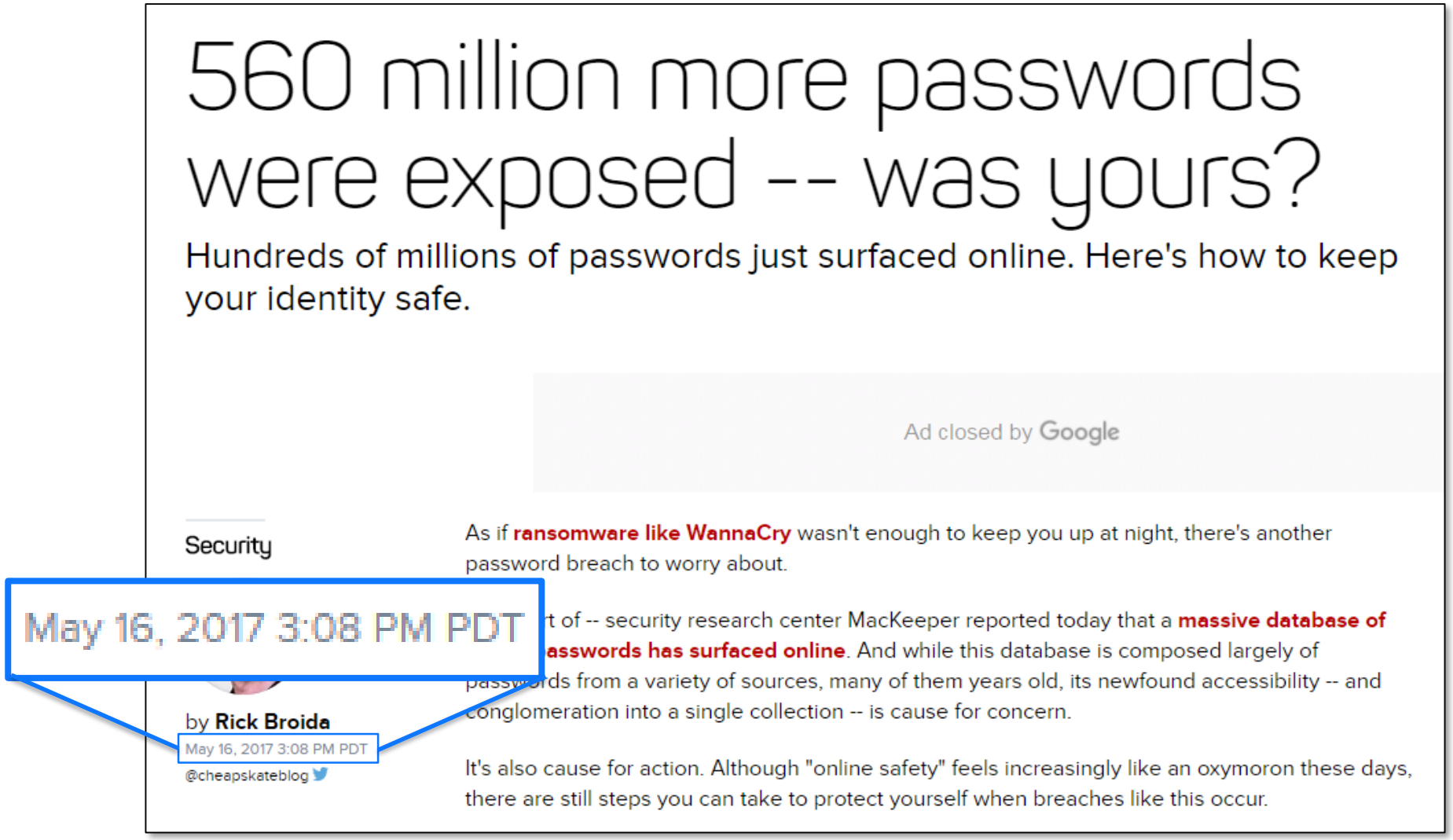
UNIVERSITÉ Concordia UNIVERSITY

intel Collaborative Research Institute for Secure Computing

Protecting Web Passwords from Rogue Servers using TEEs

Klaudia Krawiecka, Arseny Kurnikov, Andrew Paverd, Mohammad Mannan, N. Asokan

Password database breaches



560 million more passwords were exposed -- was yours?

Hundreds of millions of passwords just surfaced online. Here's how to keep your identity safe.

Security

May 16, 2017 3:08 PM PDT

by **Rick Broida**

May 16, 2017 3:08 PM PDT

@cheapskateblog

As if **ransomware like WannaCry** wasn't enough to keep you up at night, there's another password breach to worry about.

...t of -- security research center MacKeeper reported today that a **massive database of ...asswords has surfaced online**. And while this database is composed largely of pass...rds from a variety of sources, many of them years old, its newfound accessibility -- and conglomeration into a single collection -- is cause for concern.

It's also cause for action. Although "online safety" feels increasingly like an oxymoron these days, there are still steps you can take to protect yourself when breaches like this occur.

https://www.cnet.com/how-to/protect-yourself-from-the-latest-database-breach/

Phishing

August 25, 2017

Recent phishing attacks reportedly capitalize on Office 365 security holes

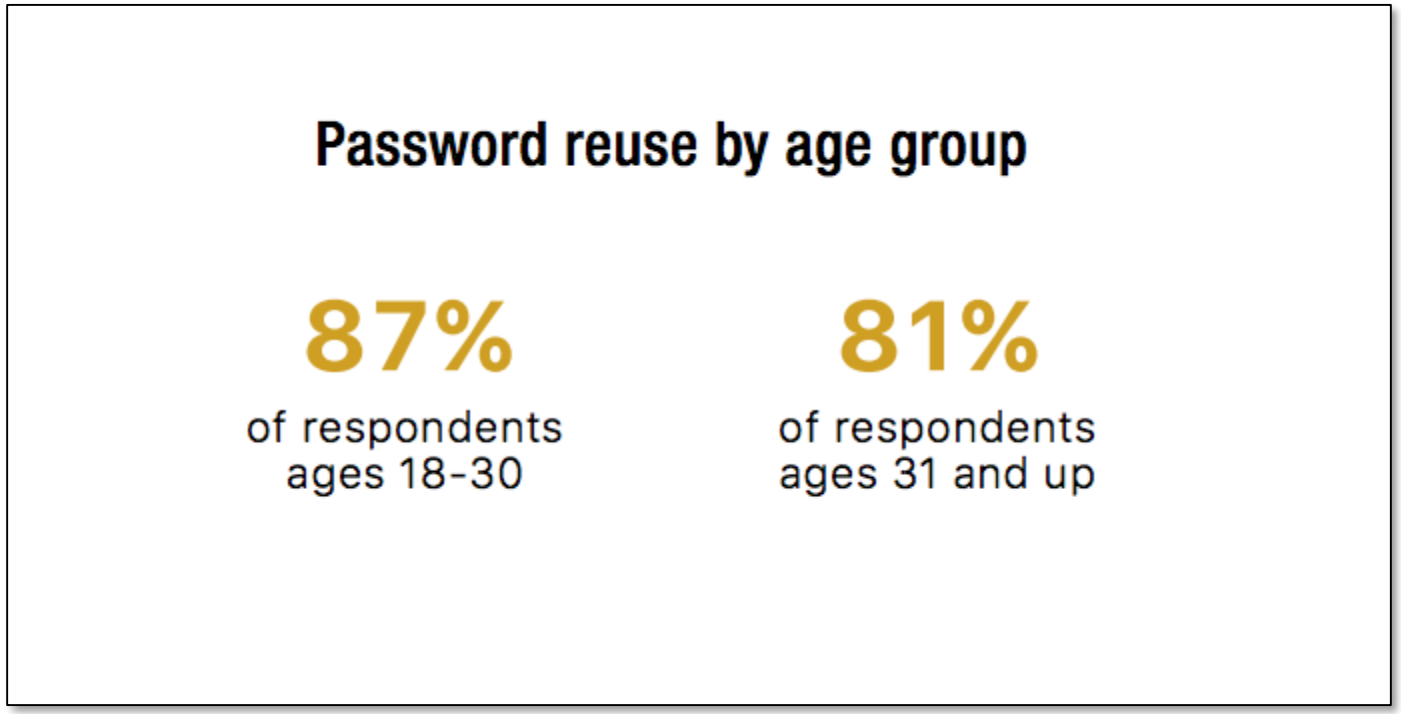
Researchers from cloud security company Avanan have reported finding two ways that phishers are evading Microsoft Office 365 Security protections: one using "hexidecimal escape characters" to conceal coding and links, and the other by compromising SharePoint files.

The first method involves emails with an HTML attachment that contain a small excerpt of JavaScript that is obscured in hexadecimal escape characters. "Therefore, no links are visible, but when opened, it presents a locally-generated phishing page with login instructions," the company explains in an Aug. 24 blog post.

In one recent example, Avanan came across a phishing email, purportedly sent by PayPal, that displayed a fraudulent login page, asking for account information such as name, address, phone number and password. By entering the information and clicking a submit button, the recipient unknowingly transmits his or her information to the cybercriminals.

https://www.scmagazine.com/recent-phishing-attacks-reportedly-capitalize-on-office-365-security-holes/article/684453/

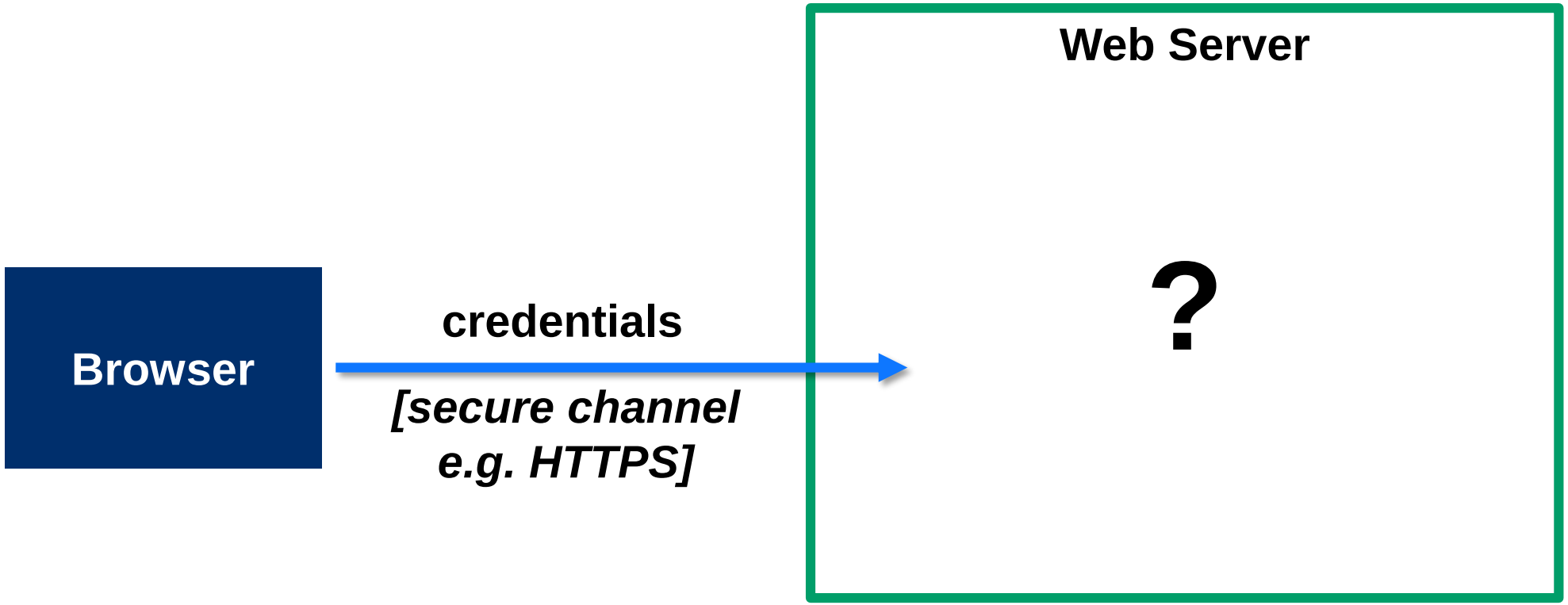
Password reuse



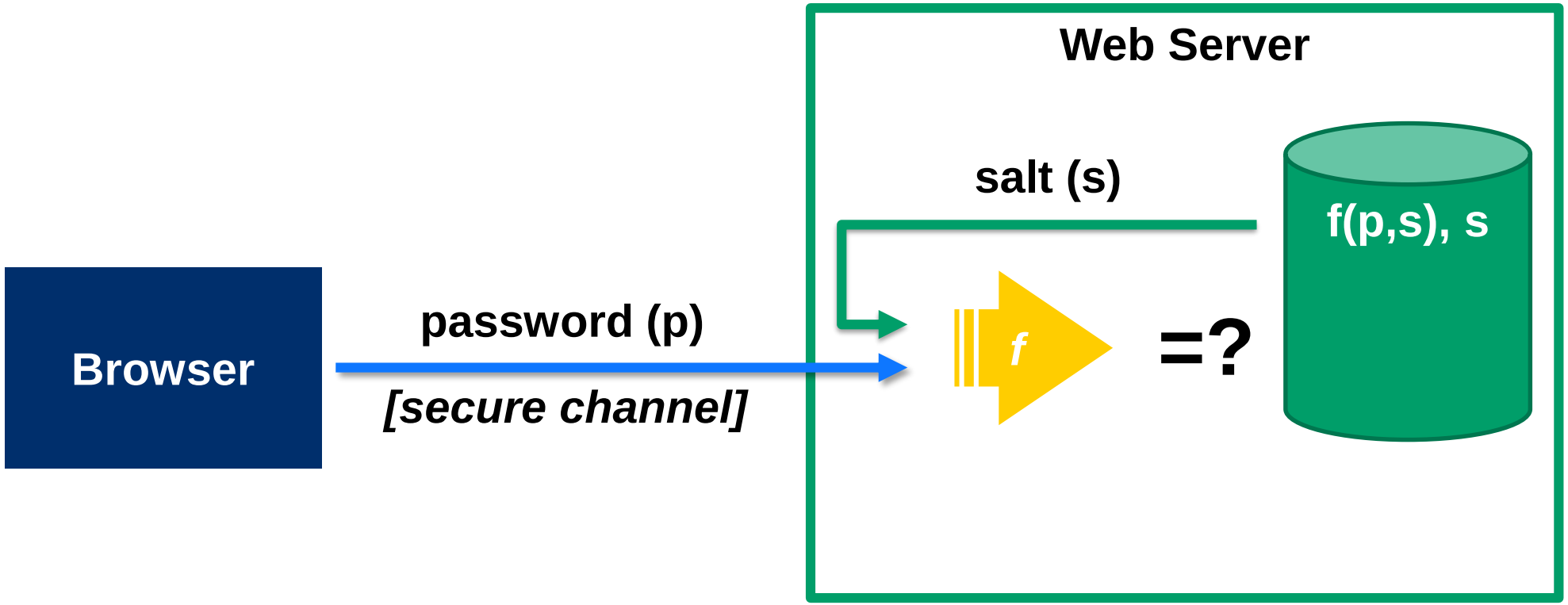
***survey of 1,000 smartphone users in 2017**

https://keepersecurity.com/assets/pdf/Keeper-Mobile-Survey-Infographic.pdf

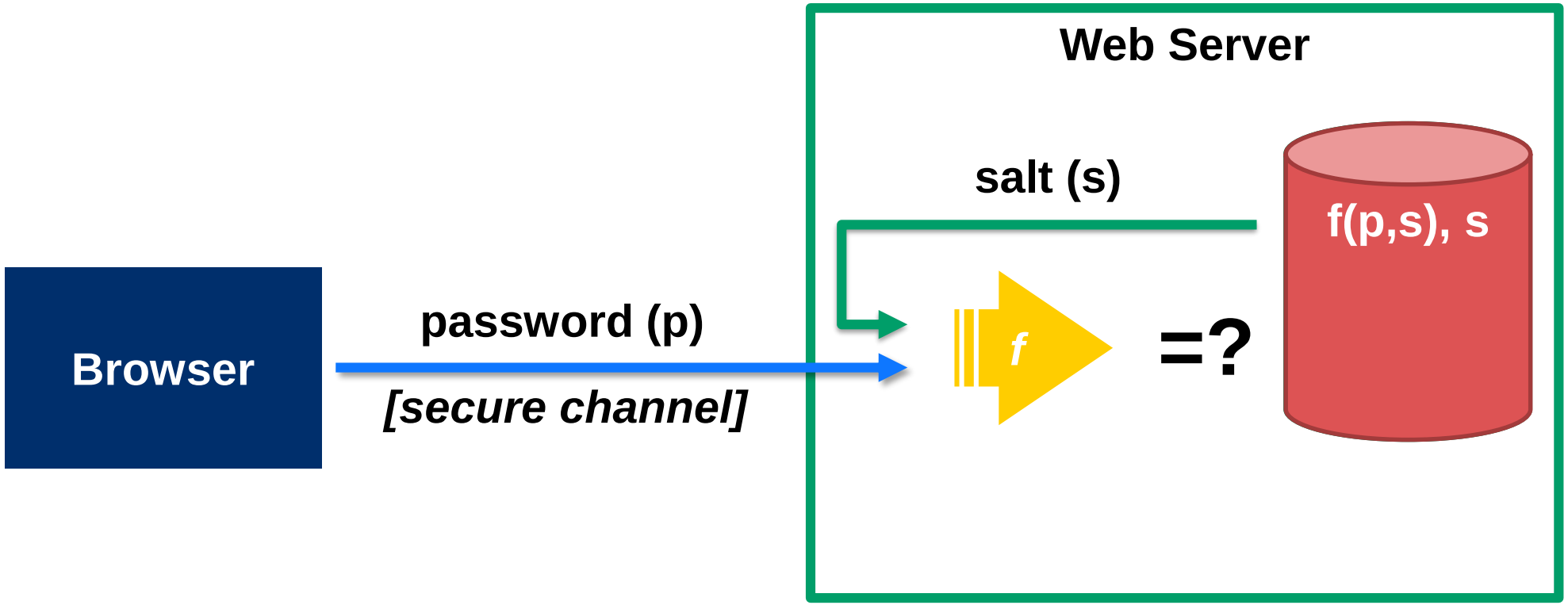
What happens on the server?



Storing passwords

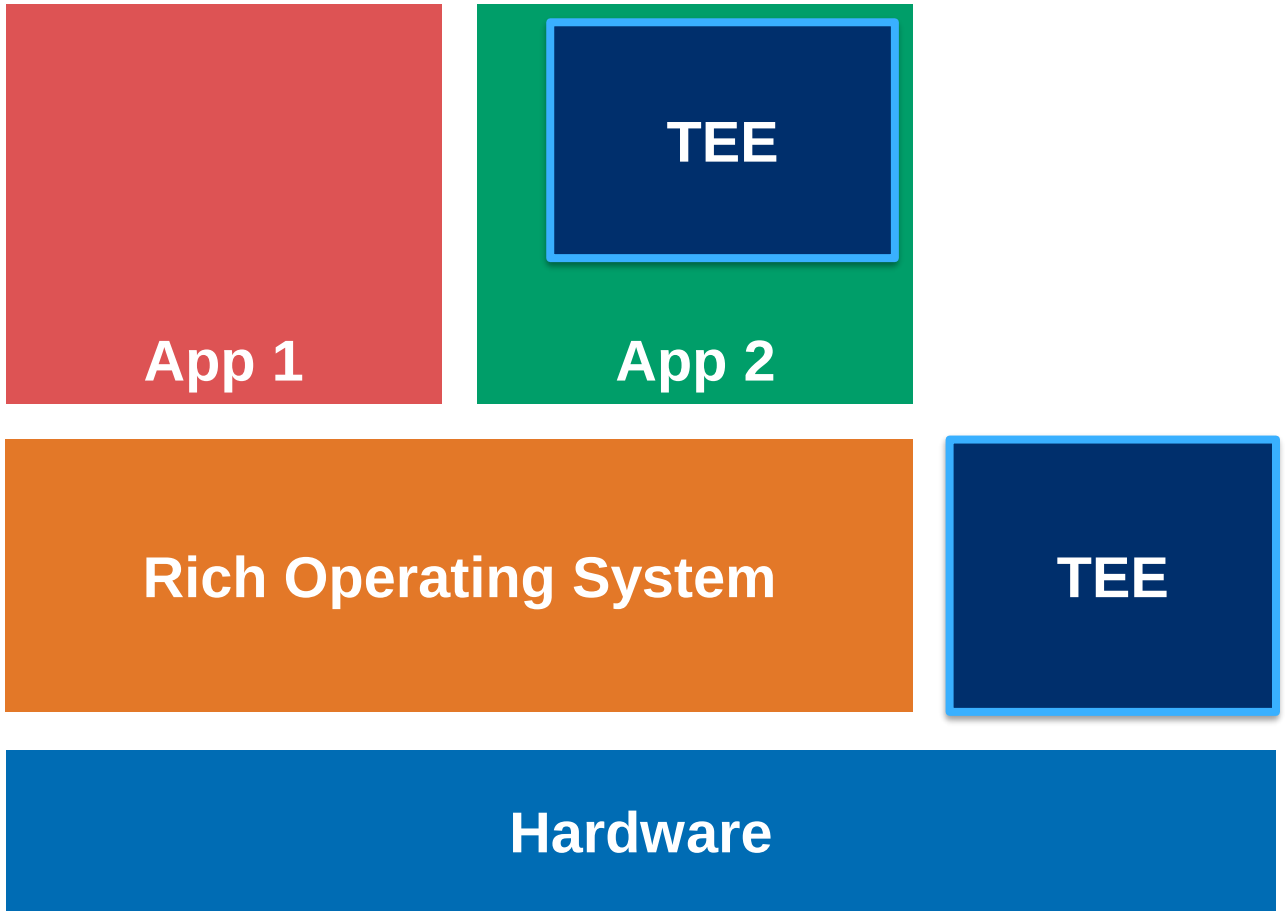


Password database breach



Offline guessing: guess passwords, apply f(), compare to leaked database.

Trusted Execution Environments



TEE features

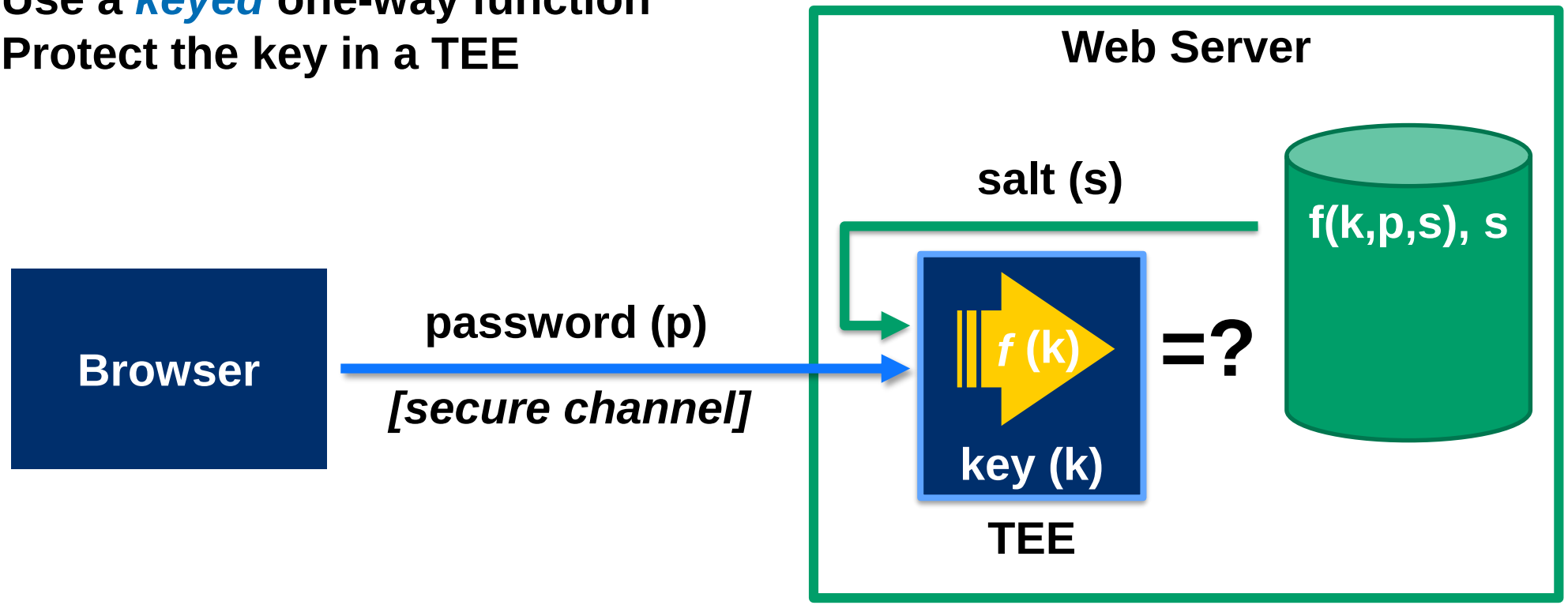
- Isolated execution
- Sealed storage
- Remote attestation

Available hardware TEEs

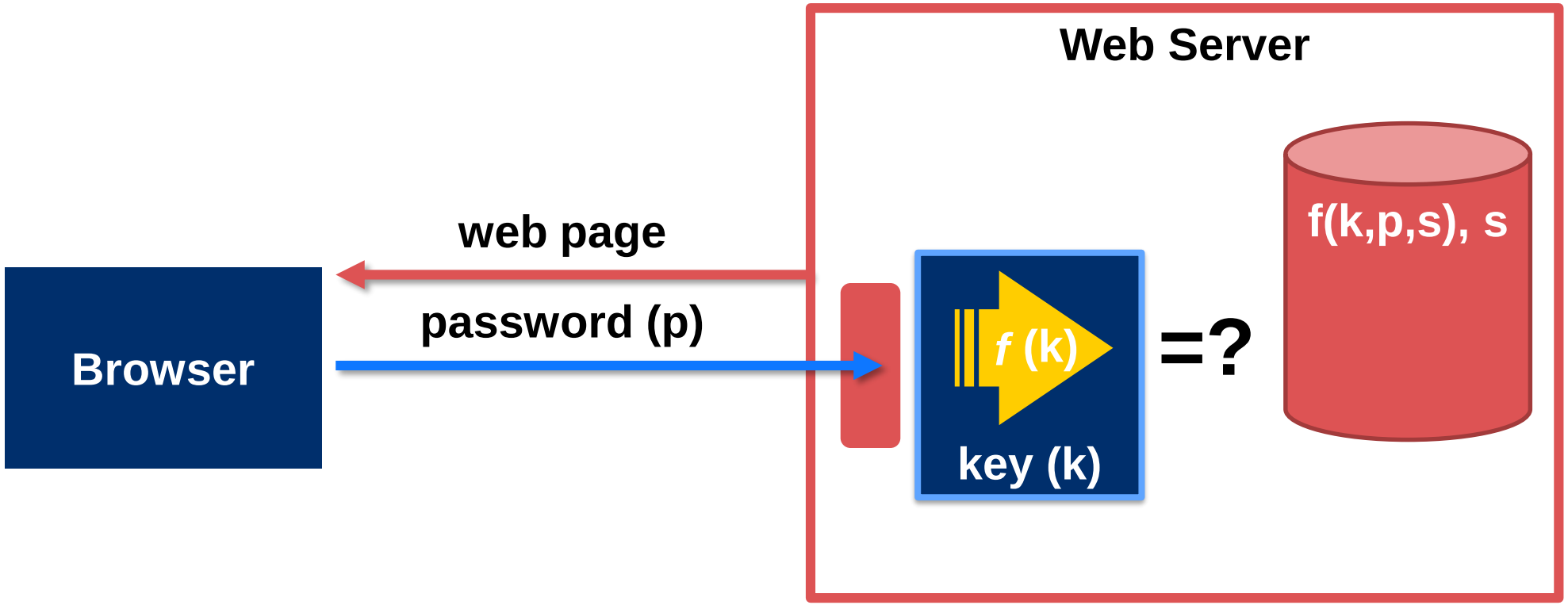
- ARM TrustZone
- Intel SGX

Storing passwords *securely*

- Use a ***keyed*** one-way function
- Protect the key in a TEE



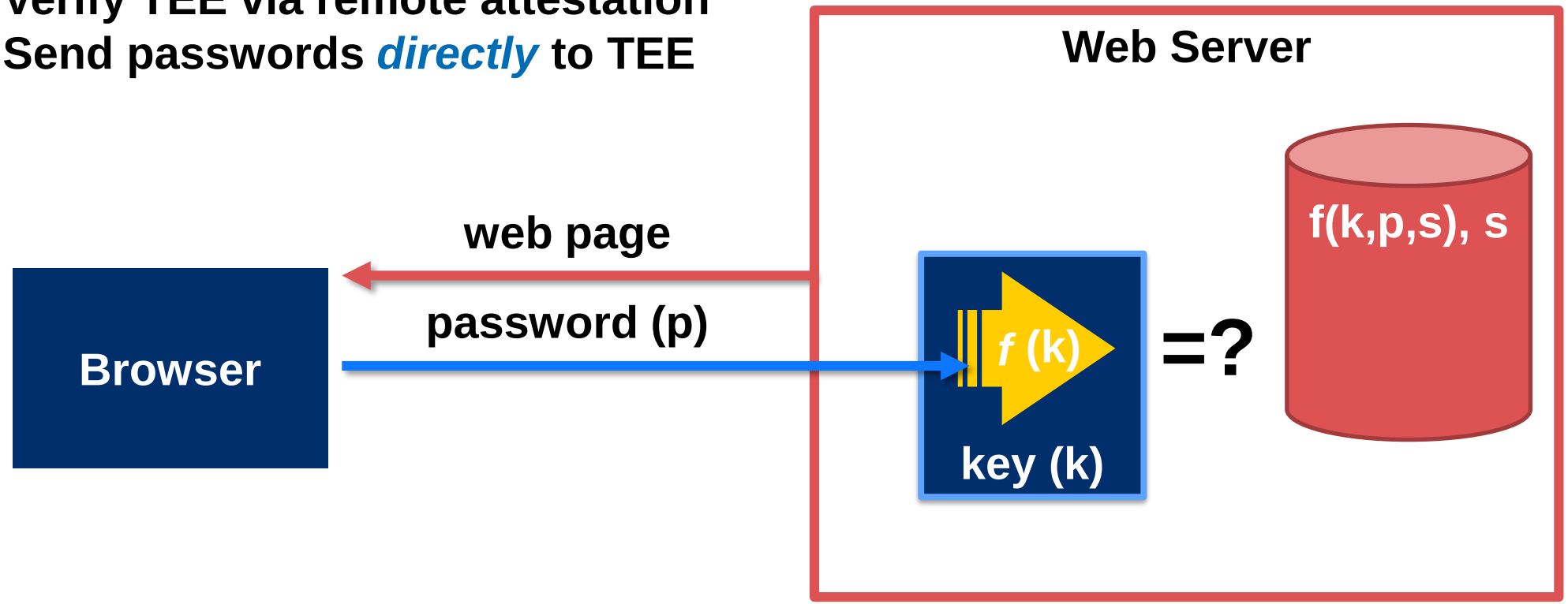
Compromised server (or Phishing)



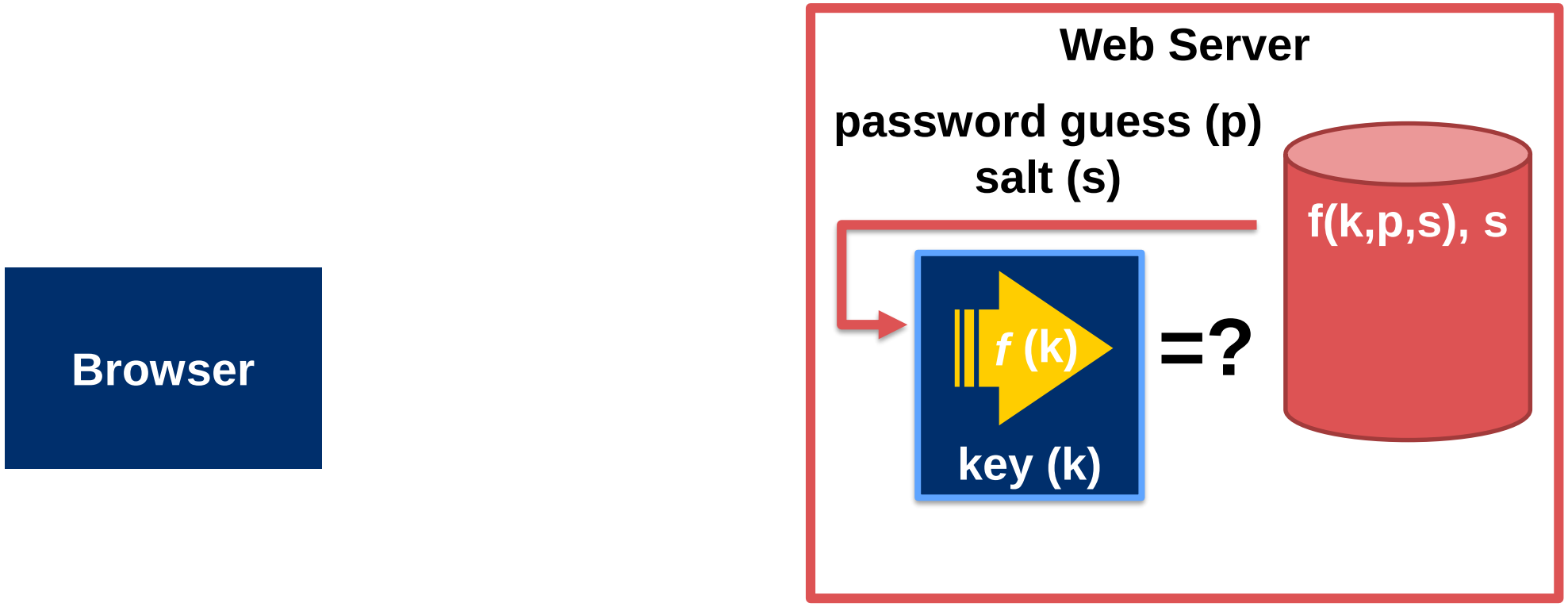
Man-in-the-Middle: intercept passwords in transit.

Transferring passwords *securely*

- Verify TEE via remote attestation
- Send passwords ***directly*** to TEE



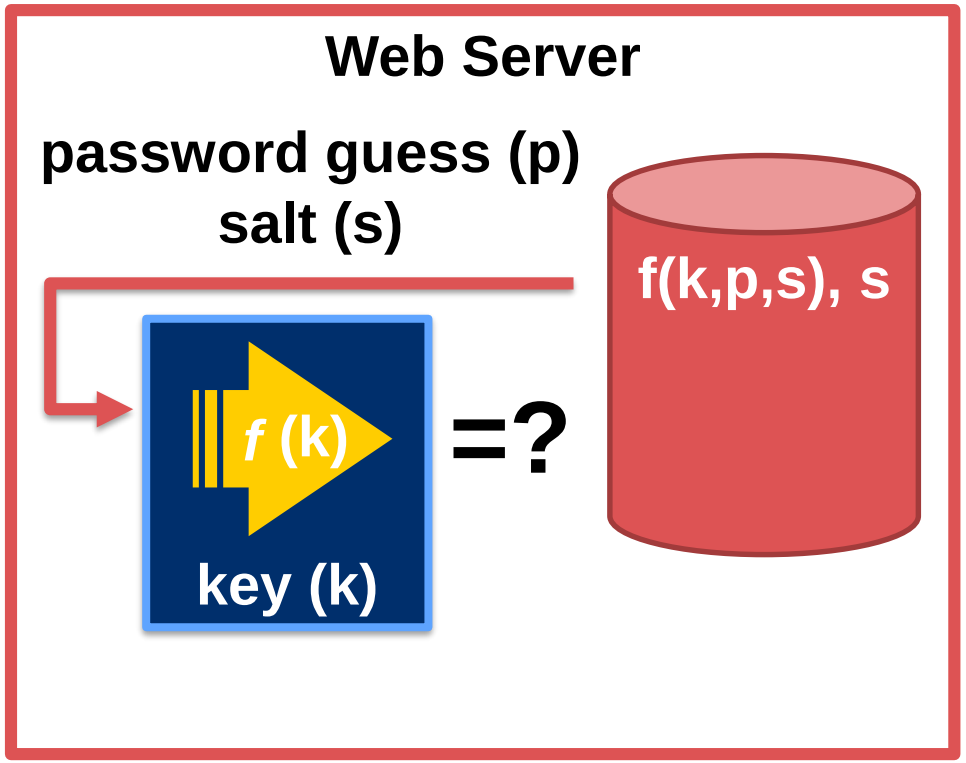
Actively malicious server



Online guessing: guess passwords, send to TEE, compare to database.

Processing passwords *securely*

- ***Rate-limiting*** **in the TEE (but can't use User ID)**



Problem definition

Adversary capabilities

- Access passwords database
- Modify web content
- Access server-client communication
- Execute server-side code
- Launch phishing attacks

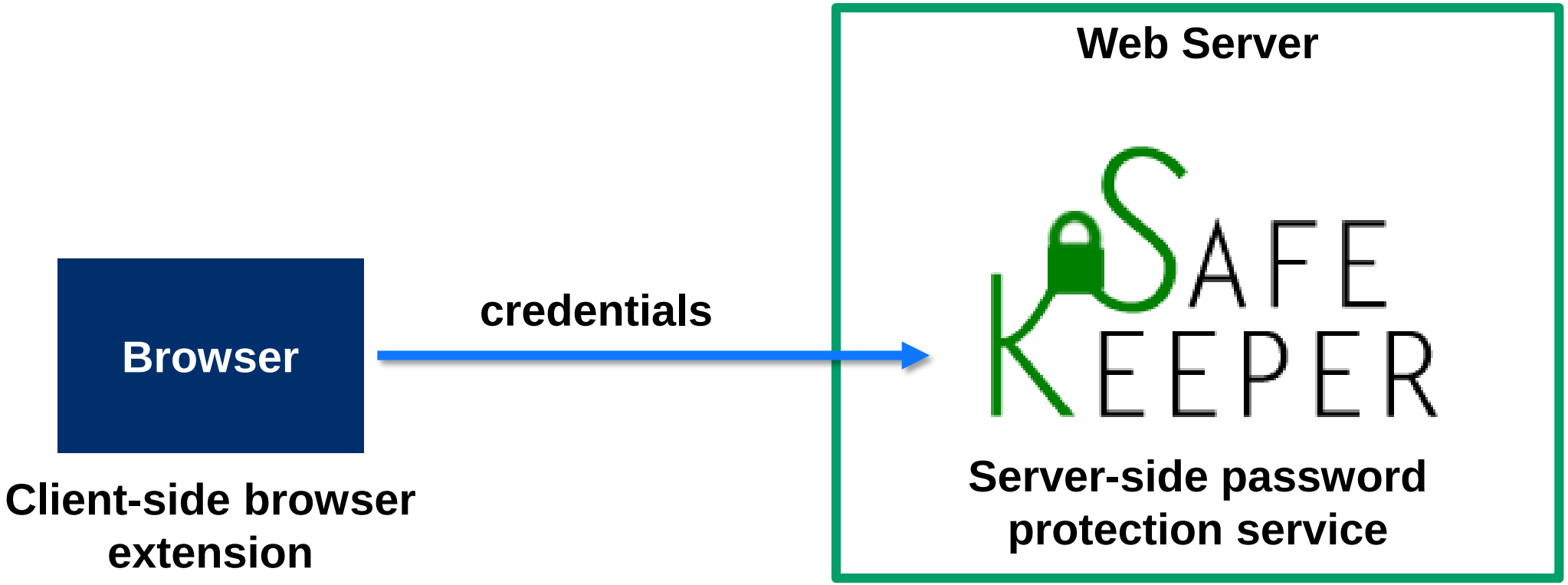
Requirements

- Password protection
  a) Passwords can only be obtained through guessing
  b) Offline guessing must be computationally infeasible
  c) Online guessing must be throttled
- User awareness

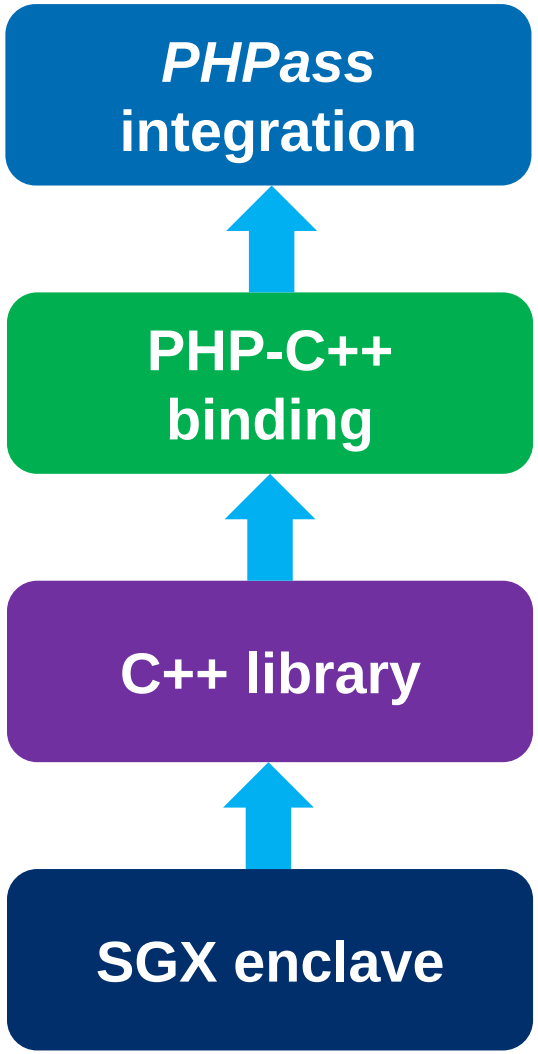
Design goals

- Minimal performance overhead
- Minimal software changes
- Ease of upgrade
- Backup and recovery

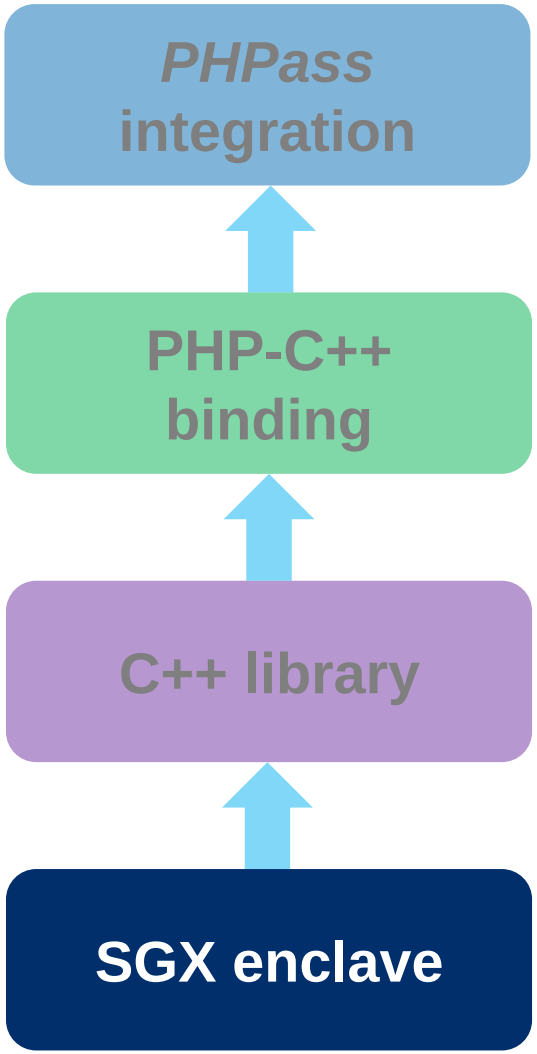
SafeKeeper


Web Server
Browser
credentials
SAFE KEEPER
Client-side browser extension
Server-side password protection service

SafeKeeper – server side



SafeKeeper – server side



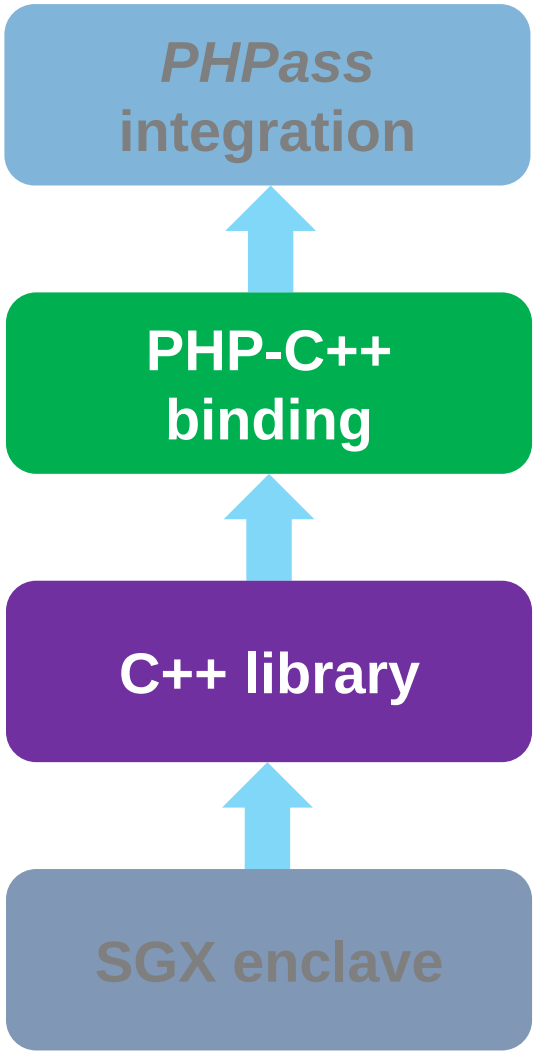
Password processing

- Key generated in enclave
- CMAC from sgx_tcrypto library
  - 128 bit key
  - AES-NI hardware acceleration

Rate limiting

- Per-user rate limiting based on *salt*
- In-TEE map of salts and attempts
  - Uses SGX trusted time

SafeKeeper – server side



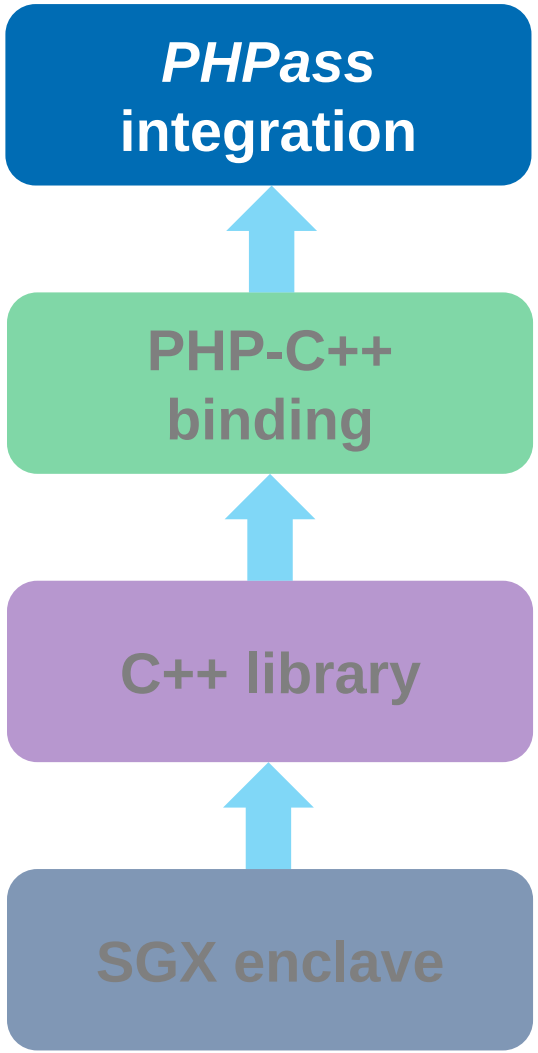
PHP-CPP

- C++ library for writing PHP extensions
  http://www.php-cpp.com/

C++ Library

- Enclave initialization
- Sealed data storage/retrieval

SafeKeeper – server side

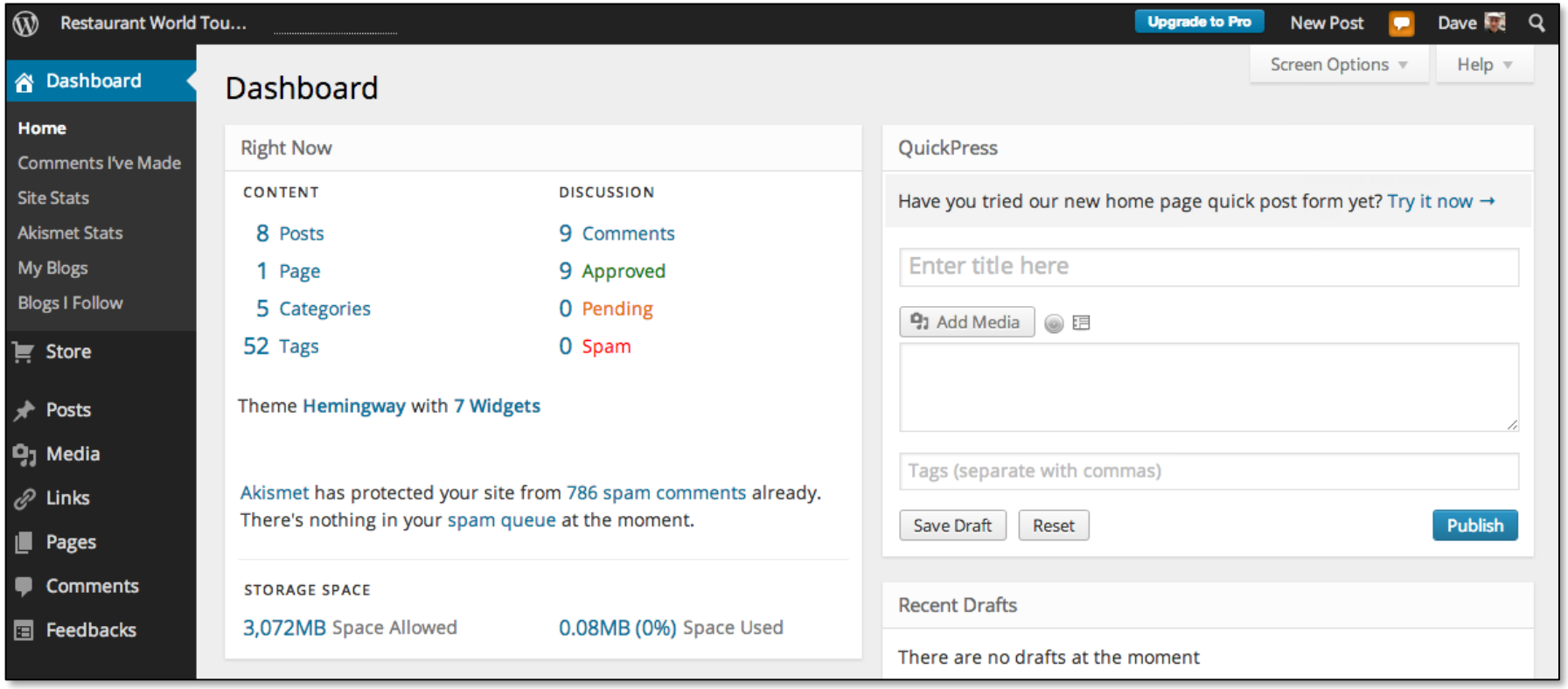


PHPass library

- Used by WordPress, Joomla, etc.
- Default: multi-round MD5 (!)

Enhanced to use our SGX enclave

WordPress using SafeKeeper



"WordPress was used by more than 27.5% of the top 10 million websites as of February 2017"
https://w3techs.com/technologies/overview/content_management/all/

Performance

Scalability (PHPass)

- Unmodified: 446 (±10) passwords/second
- SafeKeeper: 1653 (±70) passwords/second

Scalability (Enclave only)

- 101,337 (±4186) passwords/second

Memory Requirements

- 110 MB for in-enclave map of 1 million users

Setup: Intel Core i5 6500 3.2 GHz, 8 GB RAM, Ubuntu 16.04

Deployability

Software changes

- Drop-in replacement for hash function in PHPass
- Fewer than 10 lines of PHPass code changed

Upgrade path

- Can transparently upgrade existing databases without user input

Backup and recovery

- Can be distributed across multiple enclaves for scalability and failure tolerance

Problem definition

Adversary capabilities

- Access passwords database
- Modify web content
- Access server-client communication
- Execute server-side code
- Launch phishing attacks

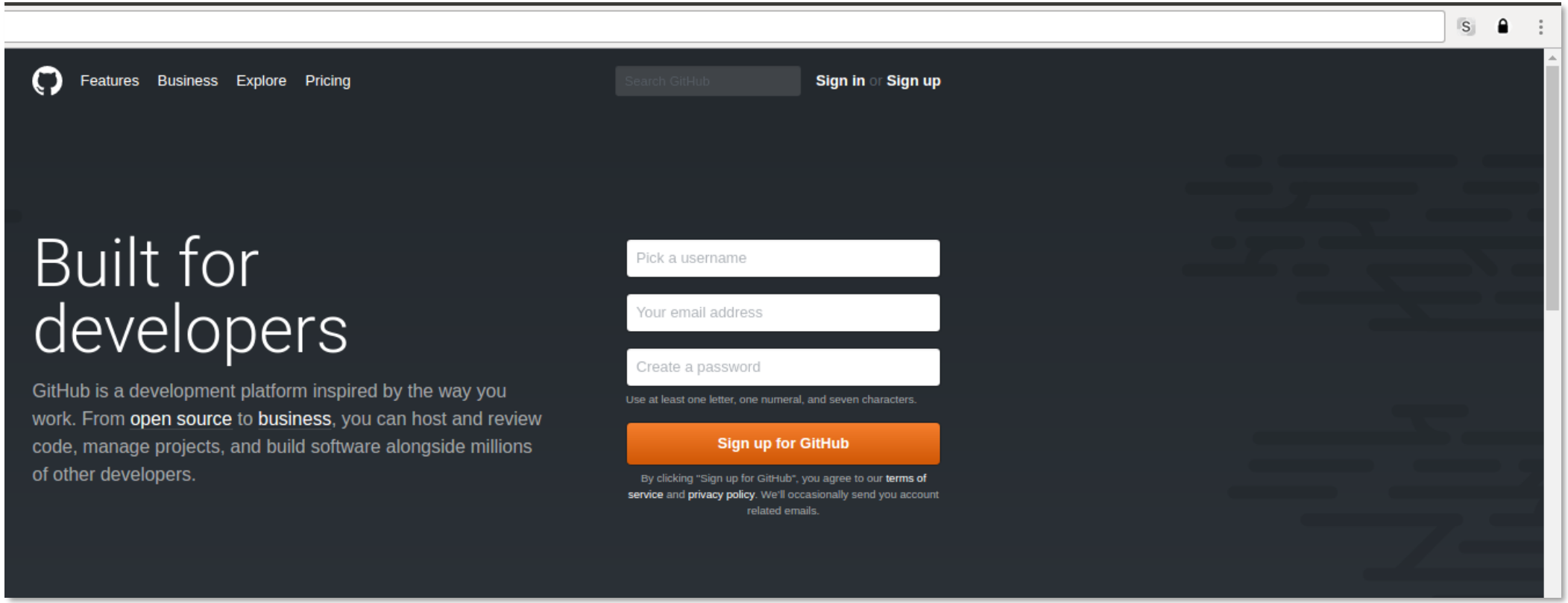
Requirements

- Password protection
  a) Passwords can only be obtained through guessing
  b) Offline guessing must be computationally infeasible
  c) Online guessing must be throttled
- User awareness

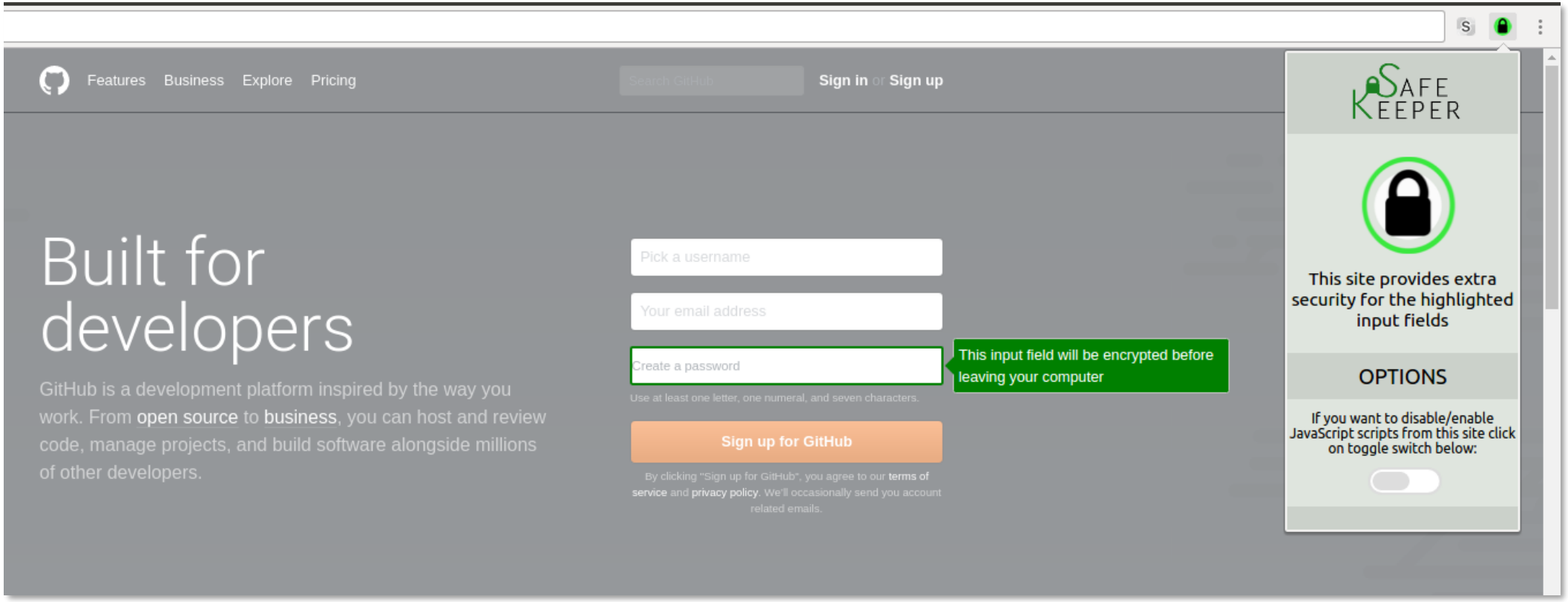
Design goals

- Minimal performance overhead
- Minimal software changes
- Ease of upgrade
- Backup and recovery

SafeKeeper Browser Add-on



SafeKeeper Browser Add-on



Can users use this effectively?

User study

86-participant on-site user study

Participants recruited using:

- Social media
- Email lists

Broad range of disciplines:

- Computer science
- MBA
- Design
- Consumer psychology
- …

Age	
18 – 28 years	81%
29 – 38 years	19%

Highest qualification	
High school	9%
Bachelors	41%
Masters	34%
PhD	2%
Not specified	14%

Gender	
Male	72%
Female	28%

User study

"Does this website use SafeKeeper to protect your password?"

Main study

- 64 participants

Follow-up study

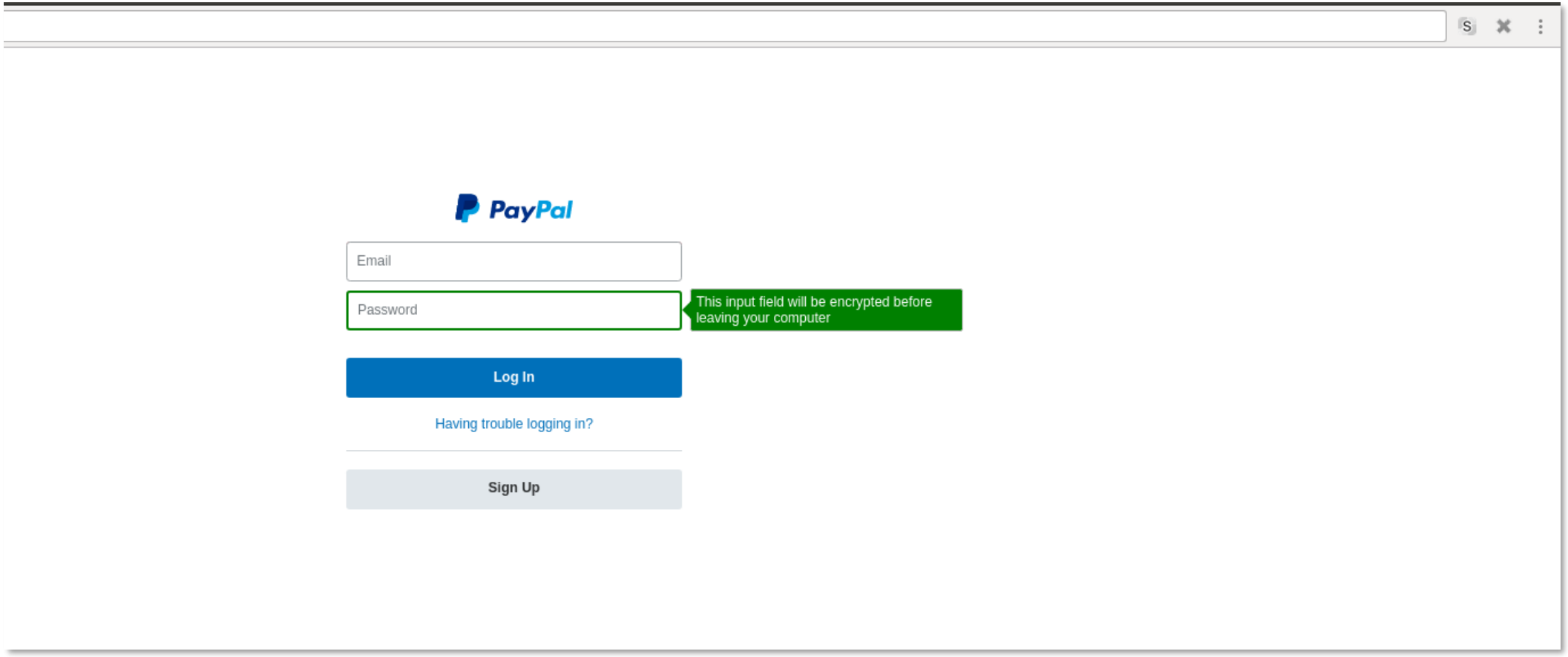
- 20 participants, 2 months later

Control group

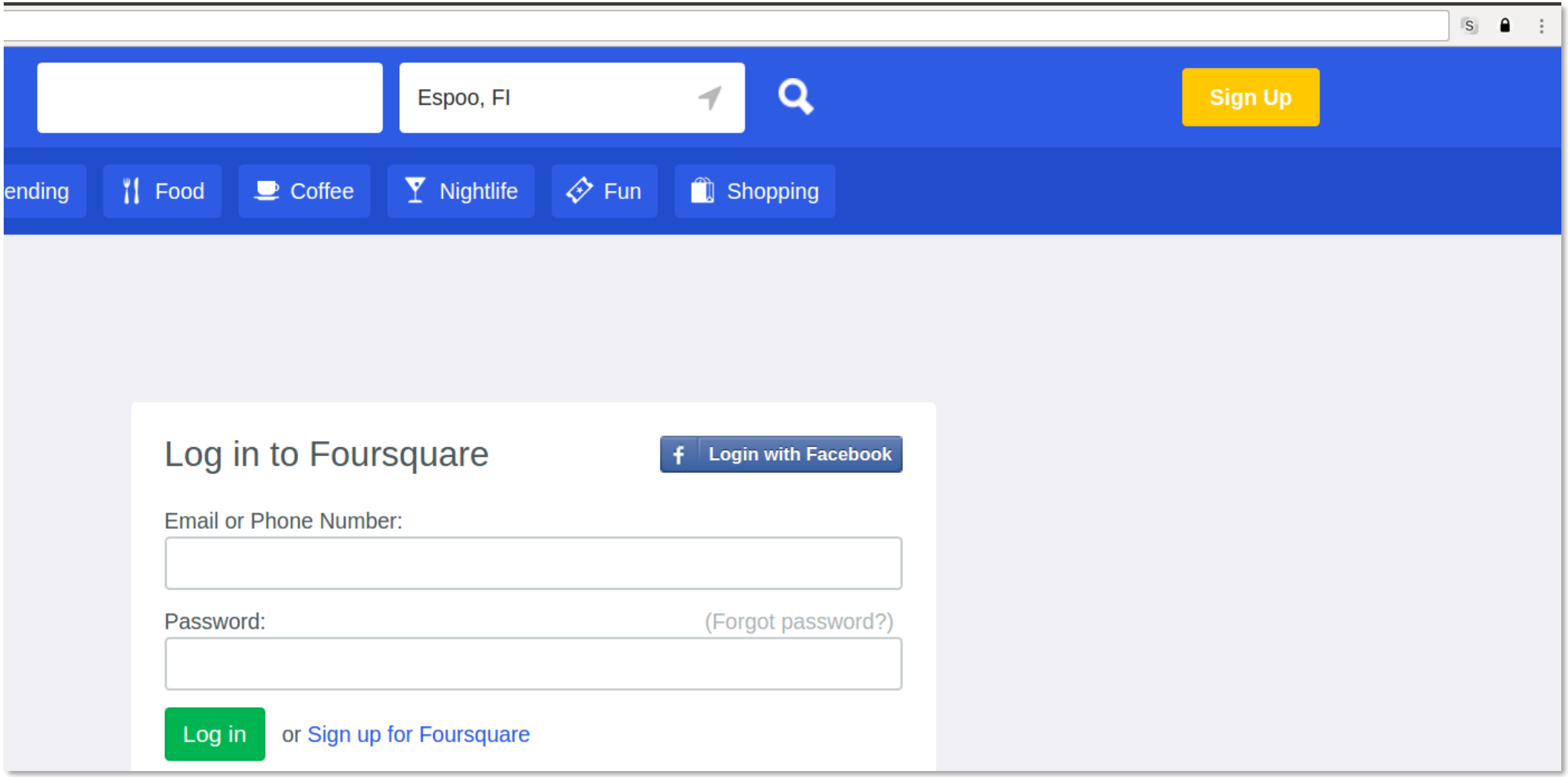
- 22 participants, no instructions

#	Protected	Type of spoofing
6	No	None
7	No	Password field highlighted
3	No	Password field highlighted after time delay
4	Yes	None
5	Yes	Other fields highlighted

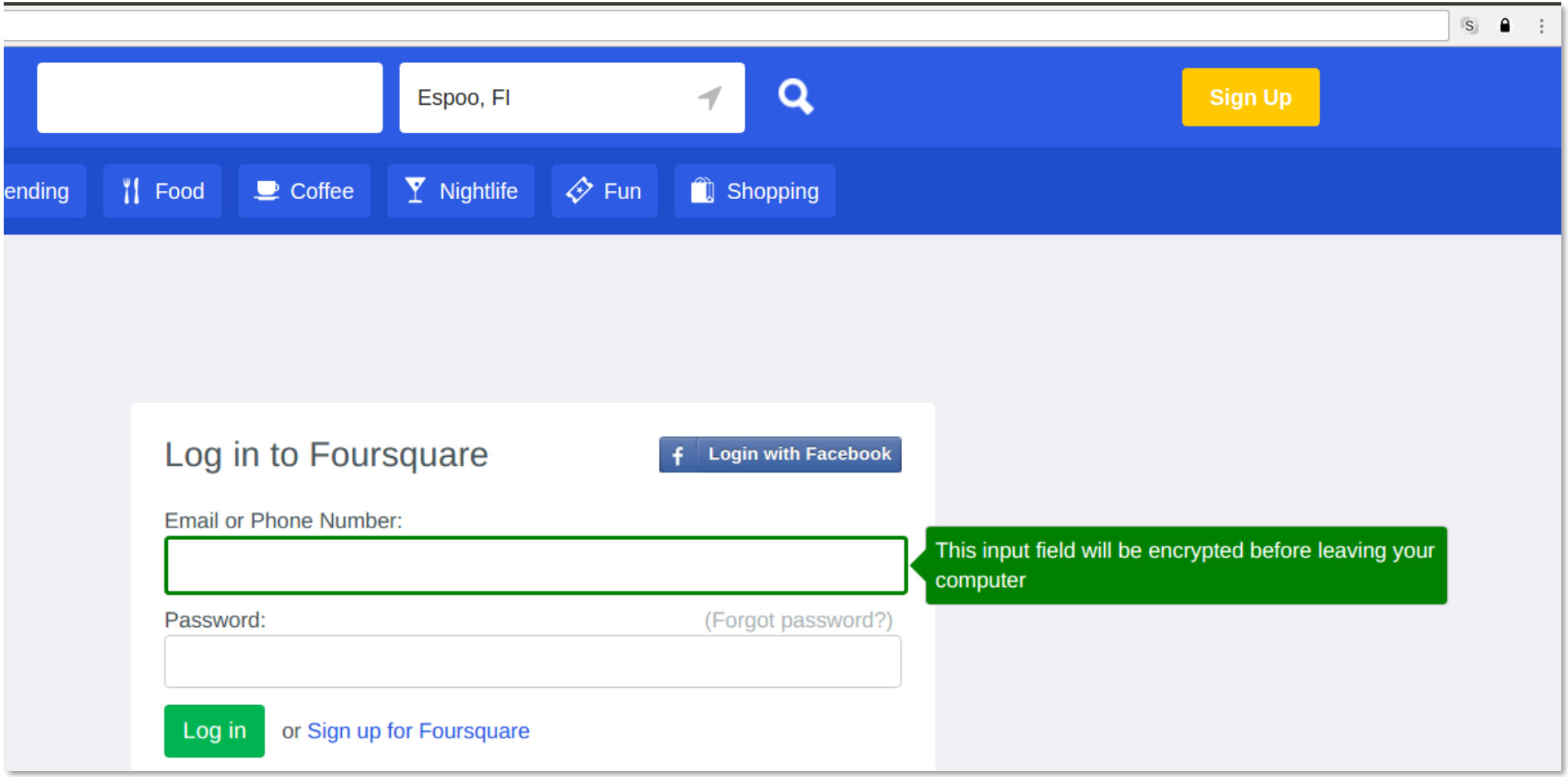
Attempted UI spoofing



Attempted UI Spoofing

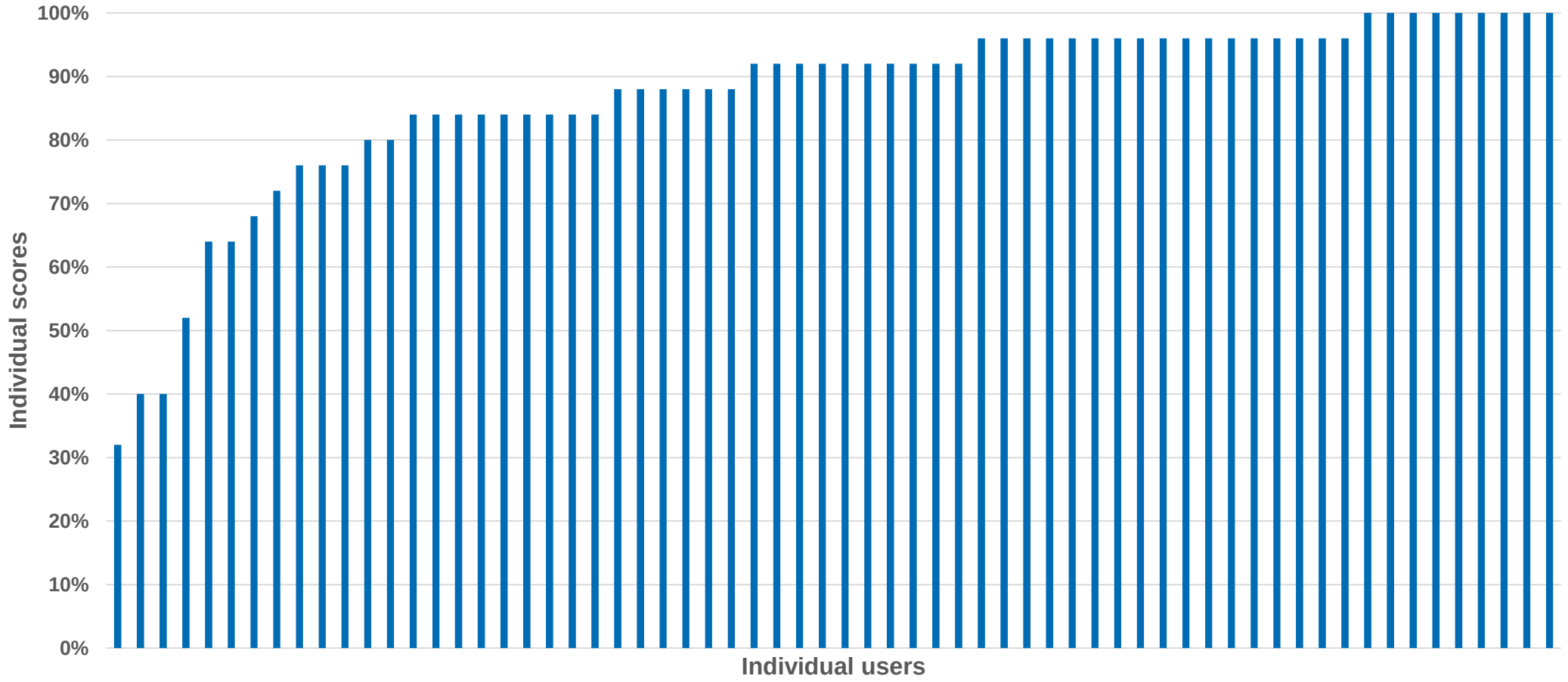


Attempted UI Spoofing



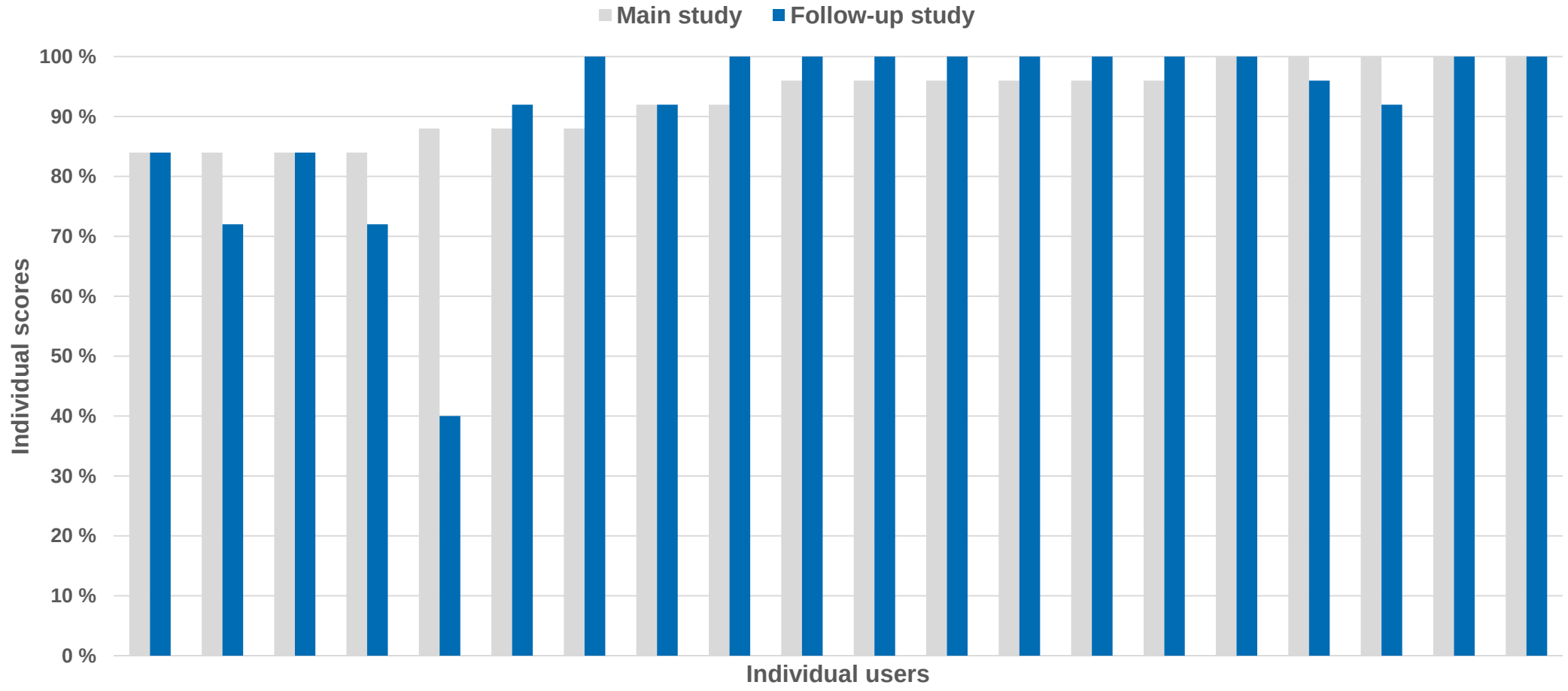
Main study results

Average effectiveness: 87%



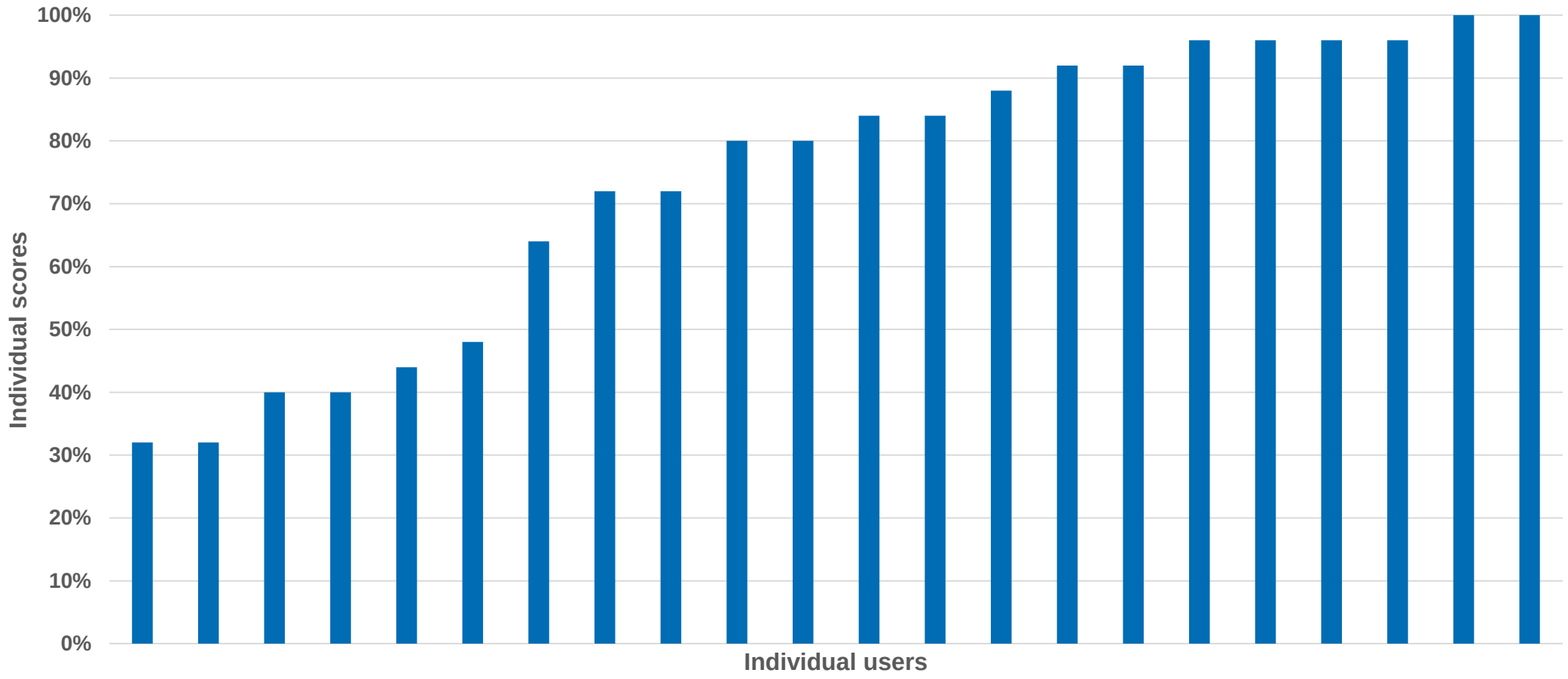
Follow-up study results

Main study effectiveness: 93% **Follow-up study effectiveness: 91%**

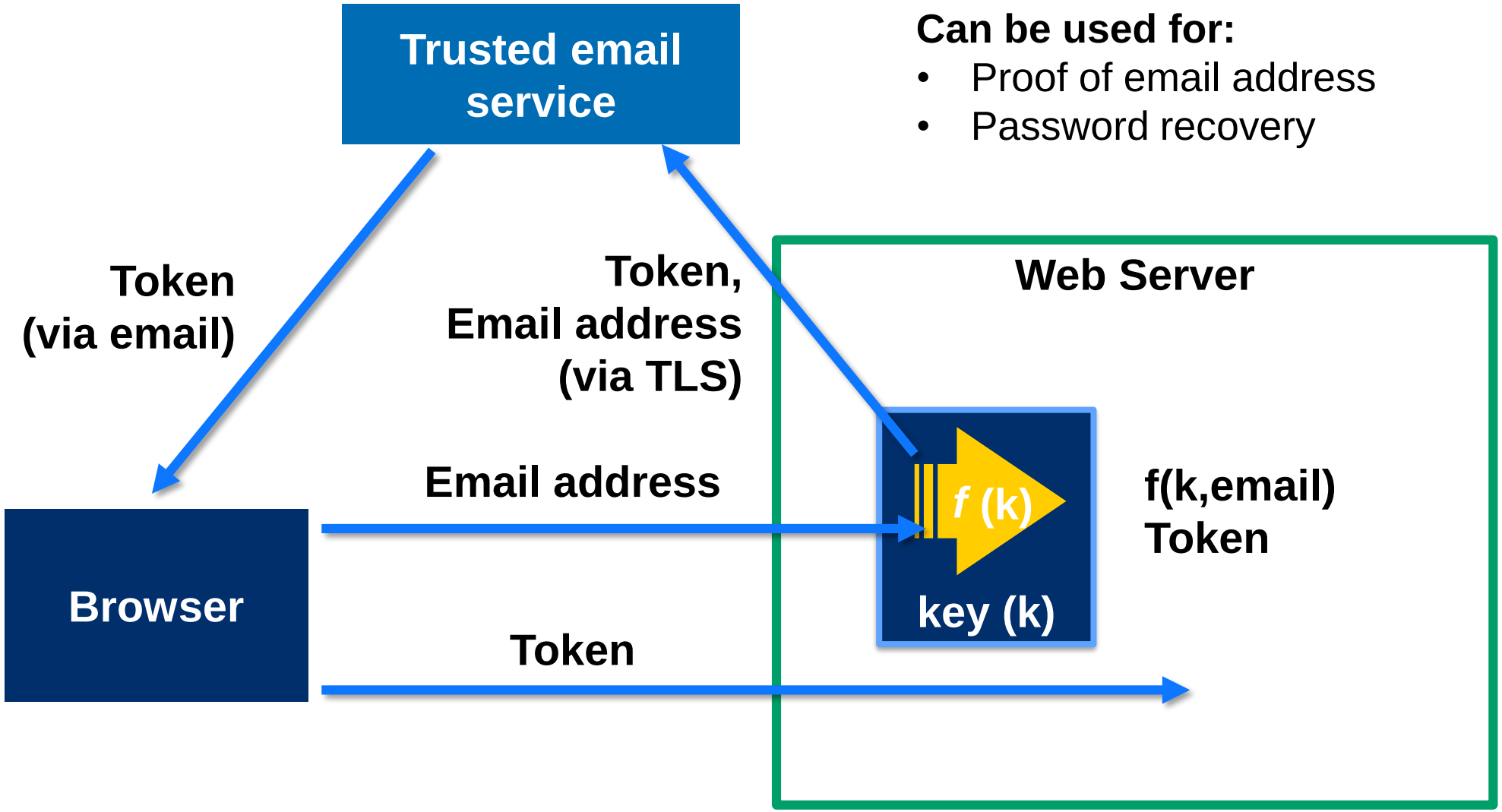


Control group results

Control group effectiveness: 74%



Future work: Protecting email addresses



Can be used for:

- Proof of email address
- Password recovery

Conclusions

- TEEs can help to protect password databases
- Can be integrated into existing systems
- Can achieve web-scale performance
- Can protect real users
- Potential for future work

https://ssg.aalto.fi/projects/passwords/