# Let’s Authenticate

## Automated Cryptographic Authentication for the Web with Simple Account Recovery

James Conners Daniel Zappala

Brigham Young University

# HBO Is Launching A New Streaming Service Because Another Password Is Just What We Need

ANABEL PASAROW
JULY 9, 2019, 3:47 PM

PHOTO: NBCU PHOTO BANK/GETTY IMAGES.

*Friends* is leaving Netflix next year — and all 236 episodes are heading to yet another streaming service: WarnerMedia's forthcoming HBO Max. The new arm of HBO, which is coming out next spring, will include the entire HBO collection, new original content, and programming from WarnerMedia's other brands, including Warner Bros., New Line, DC Entertainment, CNN, TNT, TBS, truTV, The CW, Turner Classic Movies, and Cartoon Network. Netflix, which will also soon lose *The Office* to NBC's forthcoming streaming service, reportedly had paid $100 million last December to continue licensing *Friends* prior to this reacquisition by WarnerMedia.

Among the best offerings of HBO Max:

ONE DOES NOT SIMPLY
CREATE A SILVER BULLET

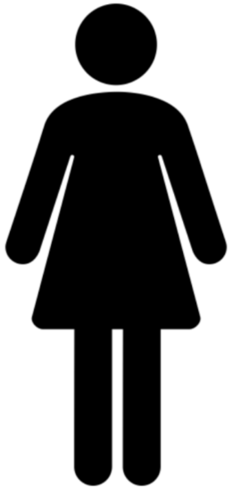
# Our focus

easy registration/login

easy account recovery

privacy by design

# What about WebAuthn?

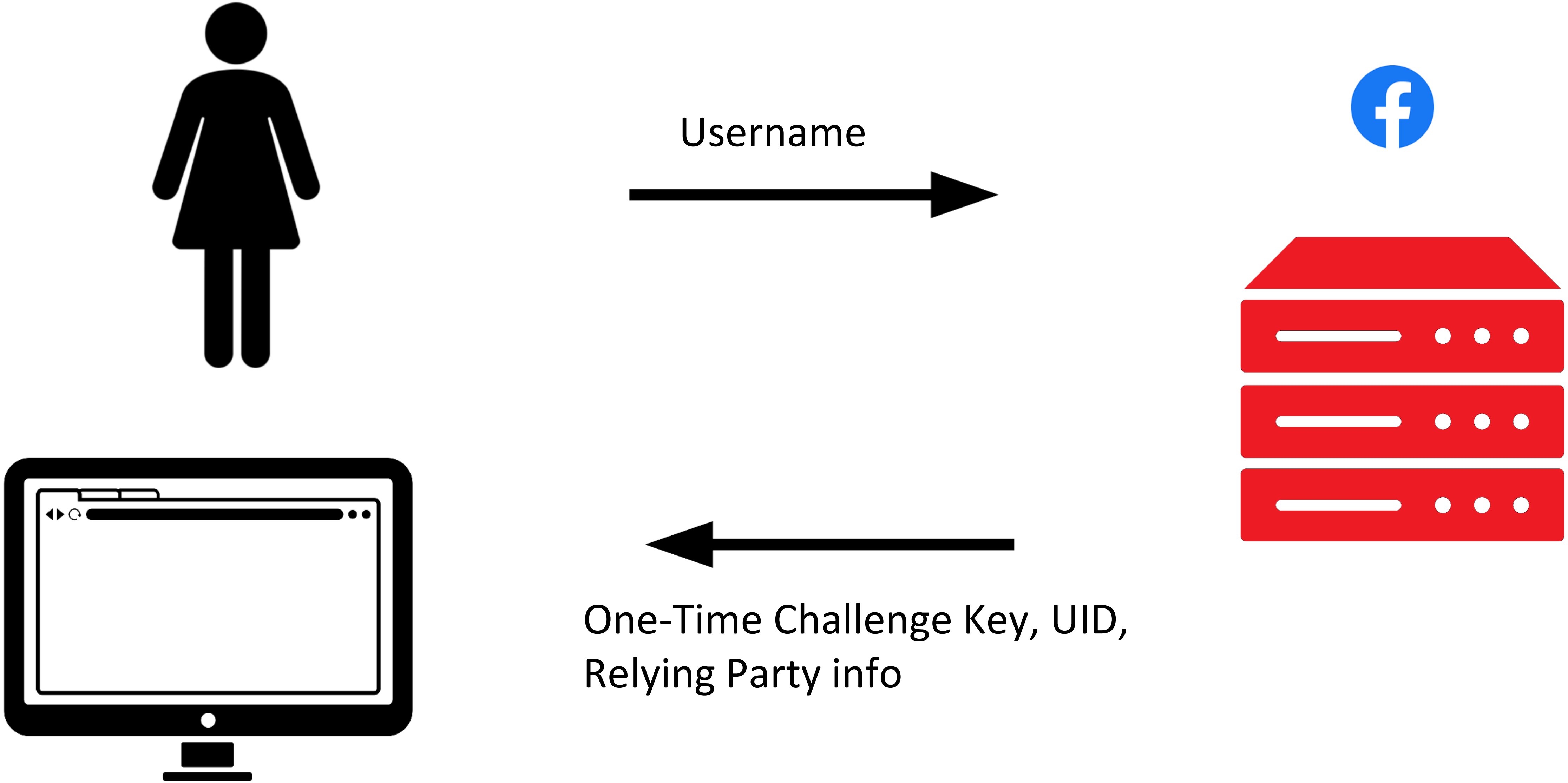


Registration/Login

Recovery

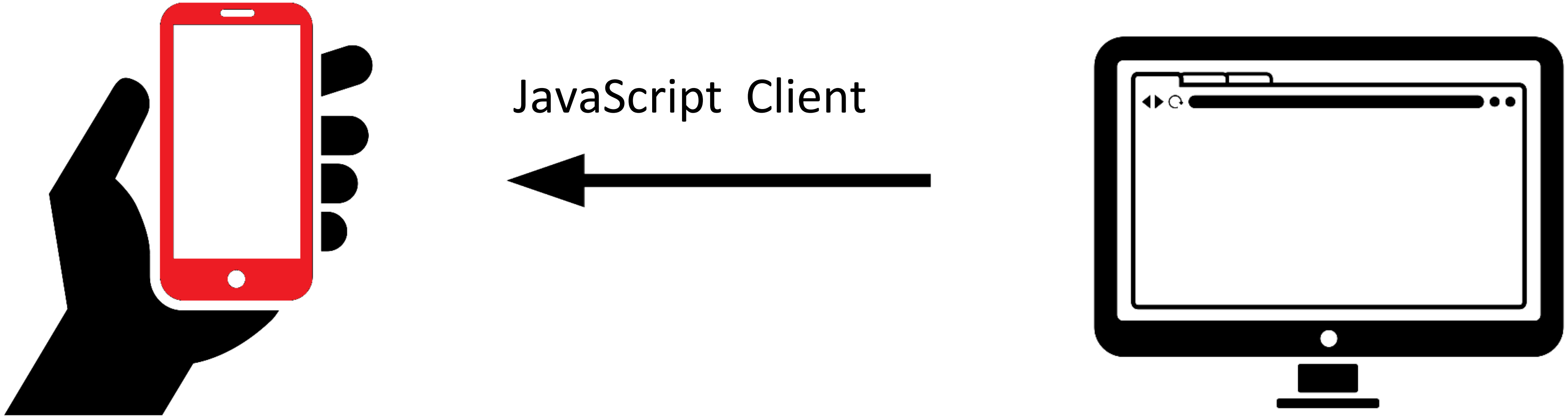
Privacy

# Alice wants to register to Facebook

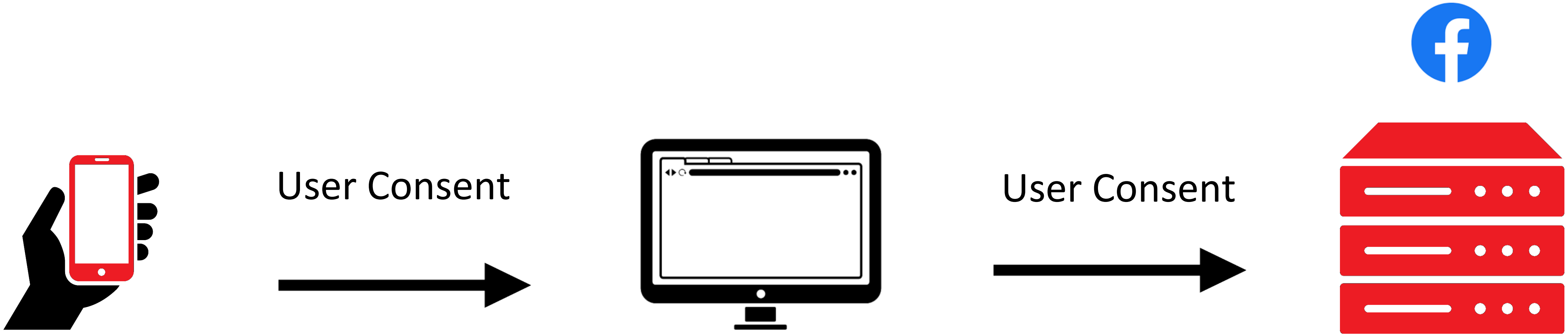


Username

One-Time Challenge Key, UID,
Relying Party info



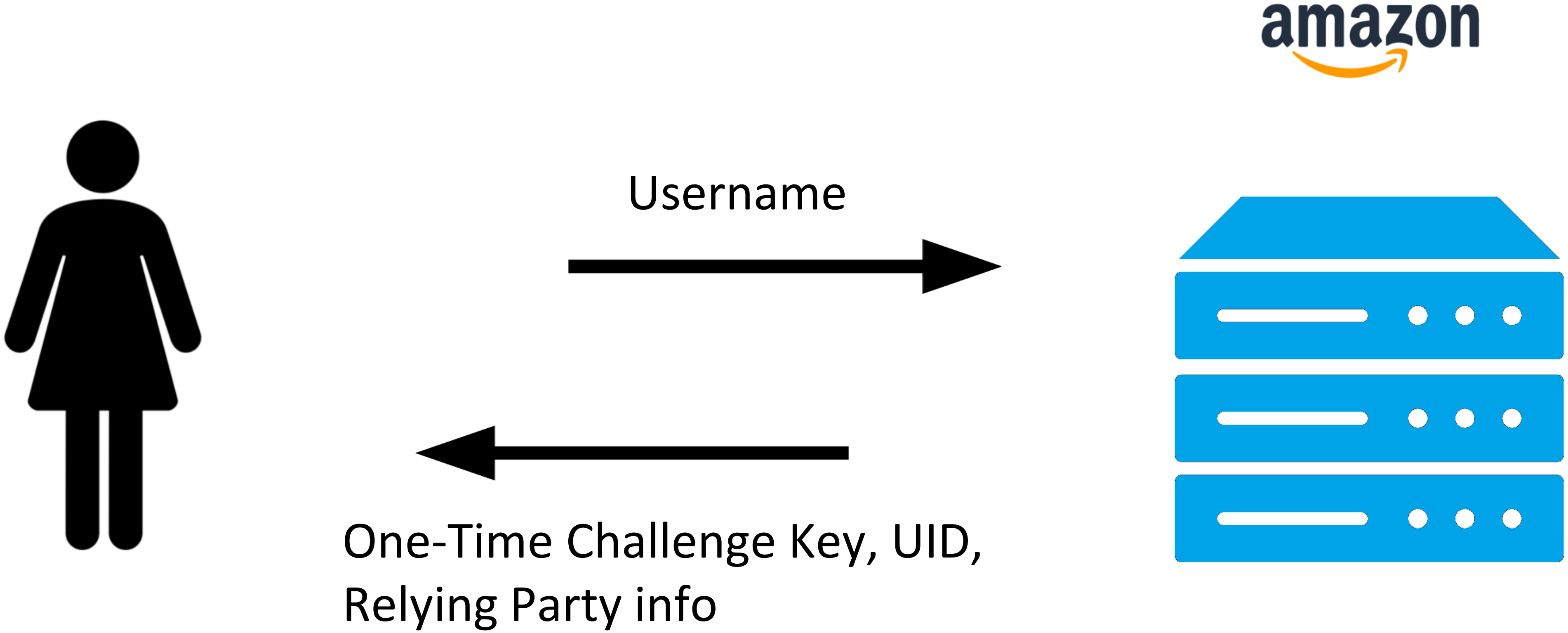
JavaScript Client



User Consent

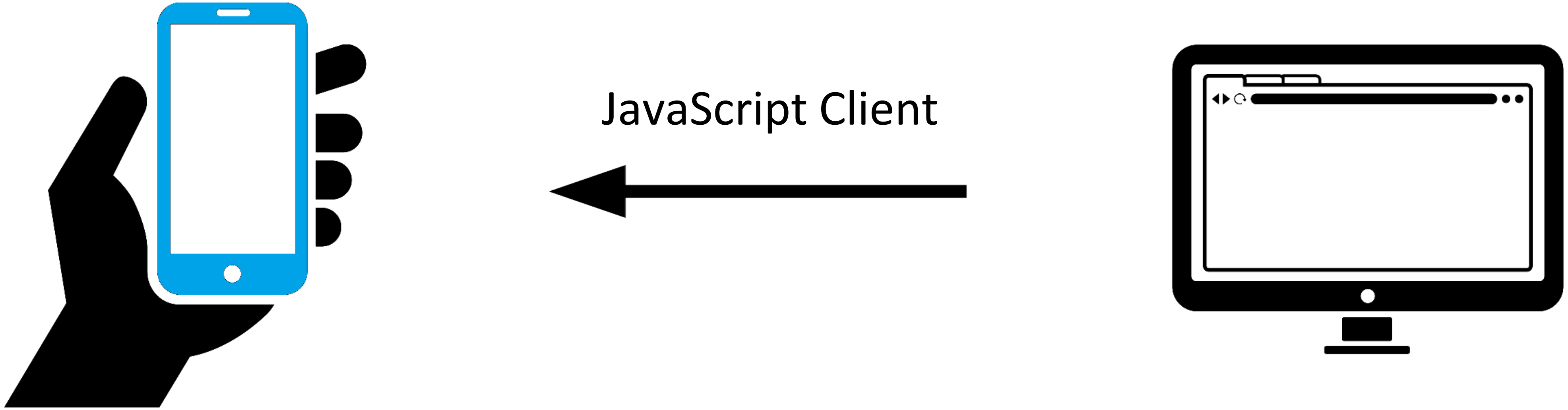
User Consent

# Alice wants to register to Amazon

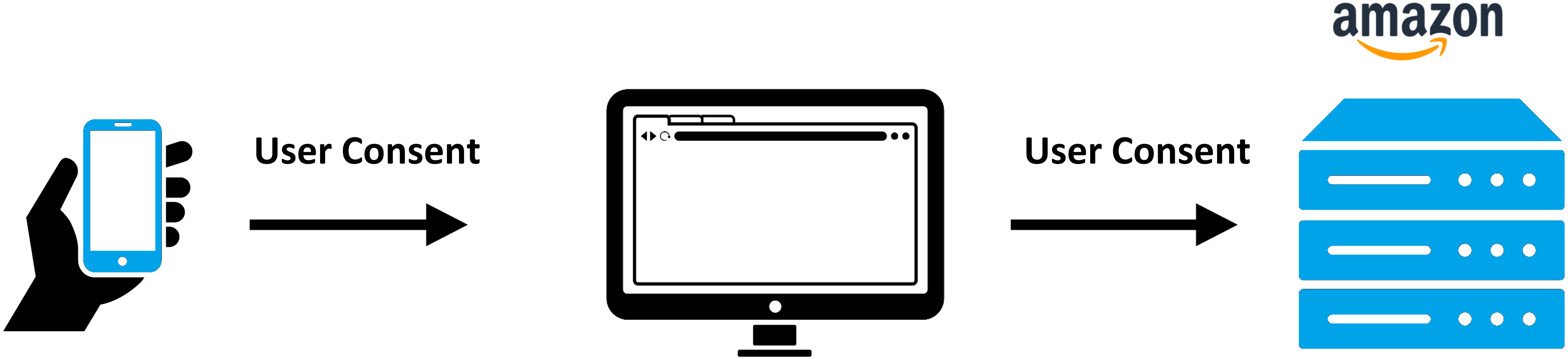


Username

One-Time Challenge Key, UID,
Relying Party info



JavaScript Client

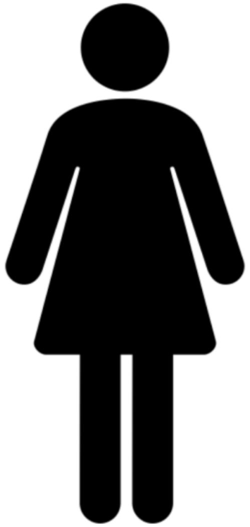


User Consent

User Consent

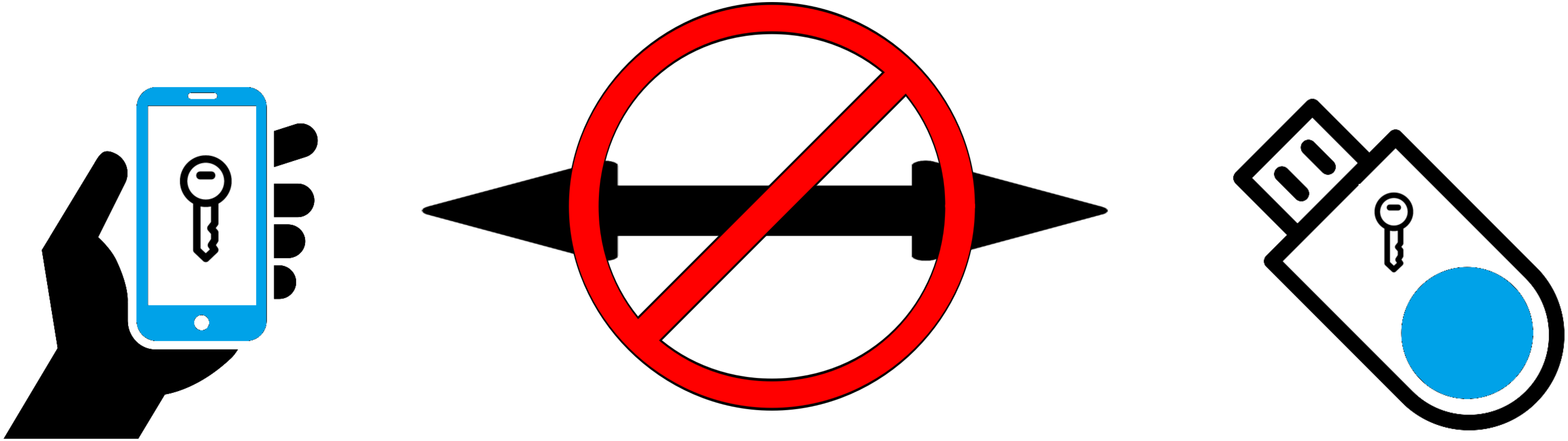
# Potential for authenticator bloat

# What happens if Alice loses her authenticator?

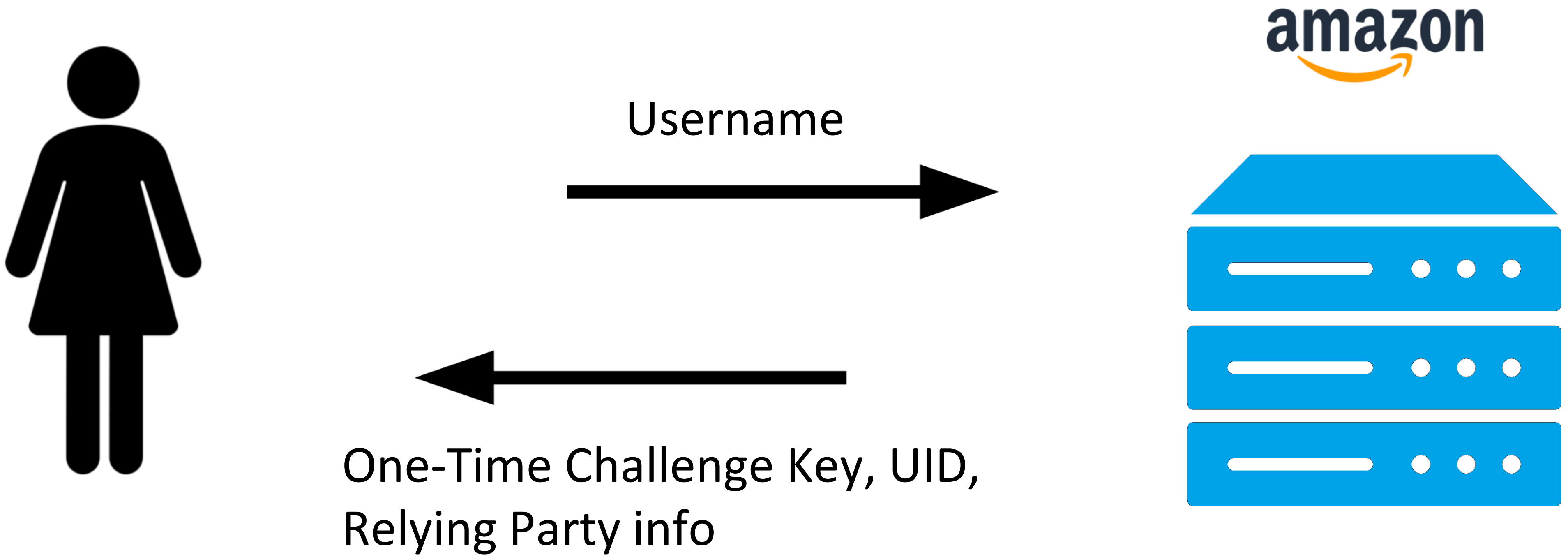


# Alice wants to recover her Amazon account

amazon

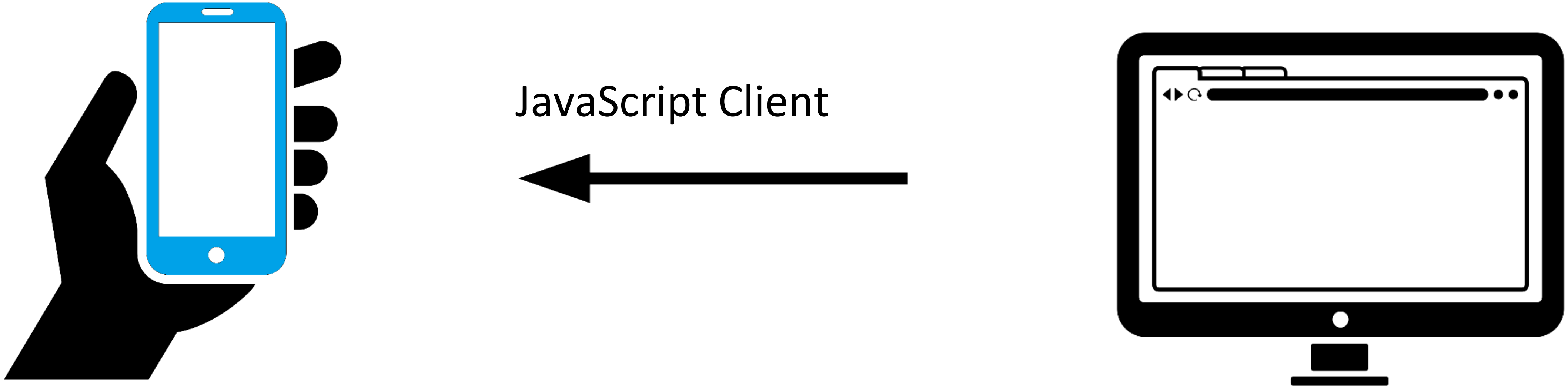


# Alice needs to register a 2nd authenticator with Amazon

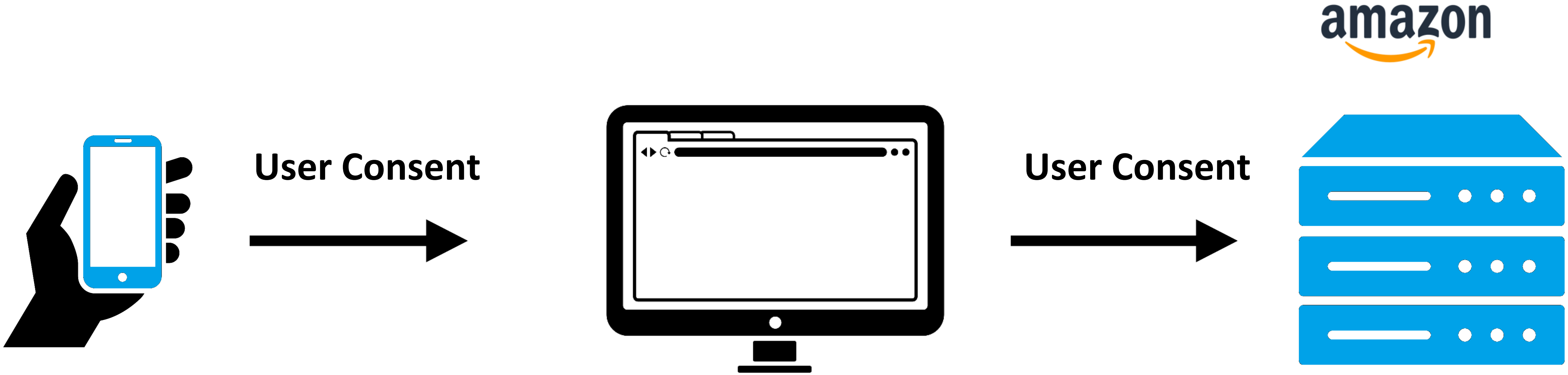


Username

One-Time Challenge Key, UID,
Relying Party info



JavaScript Client



User Consent

User Consent

# Privacy leaks and Tracking are possible

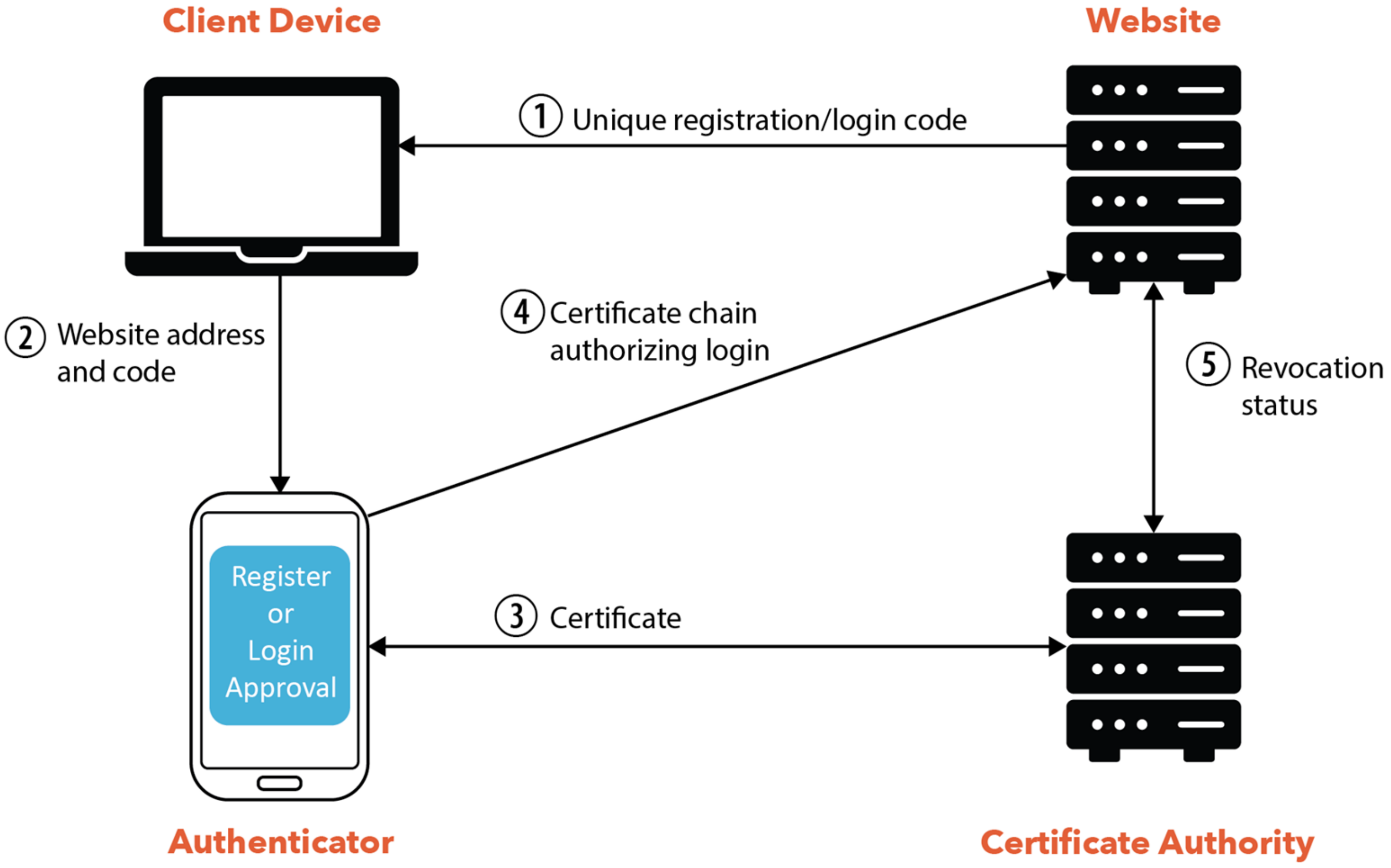
# Let’s Authenticate

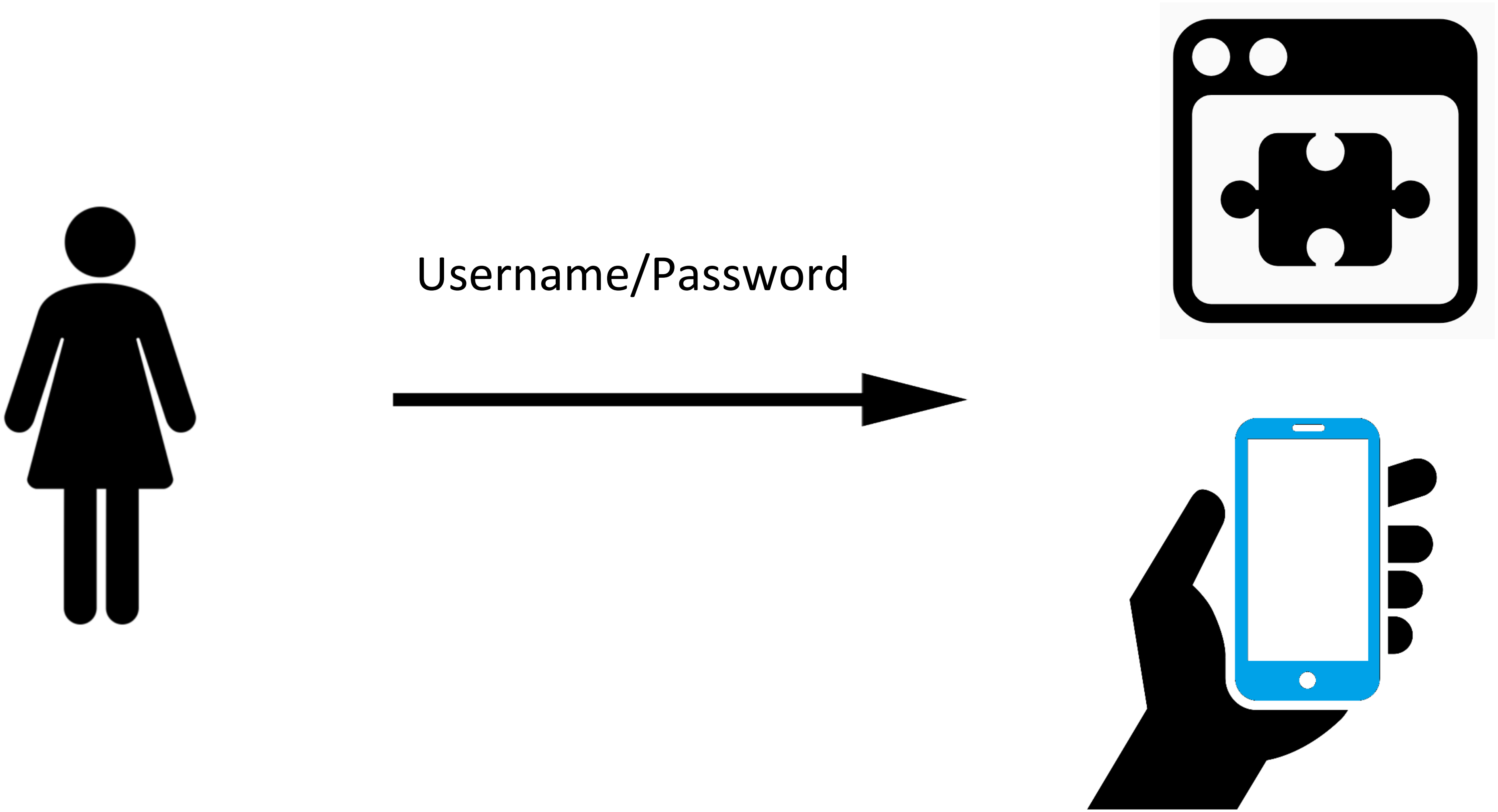
Easy Registration/Login

Easy account recovery

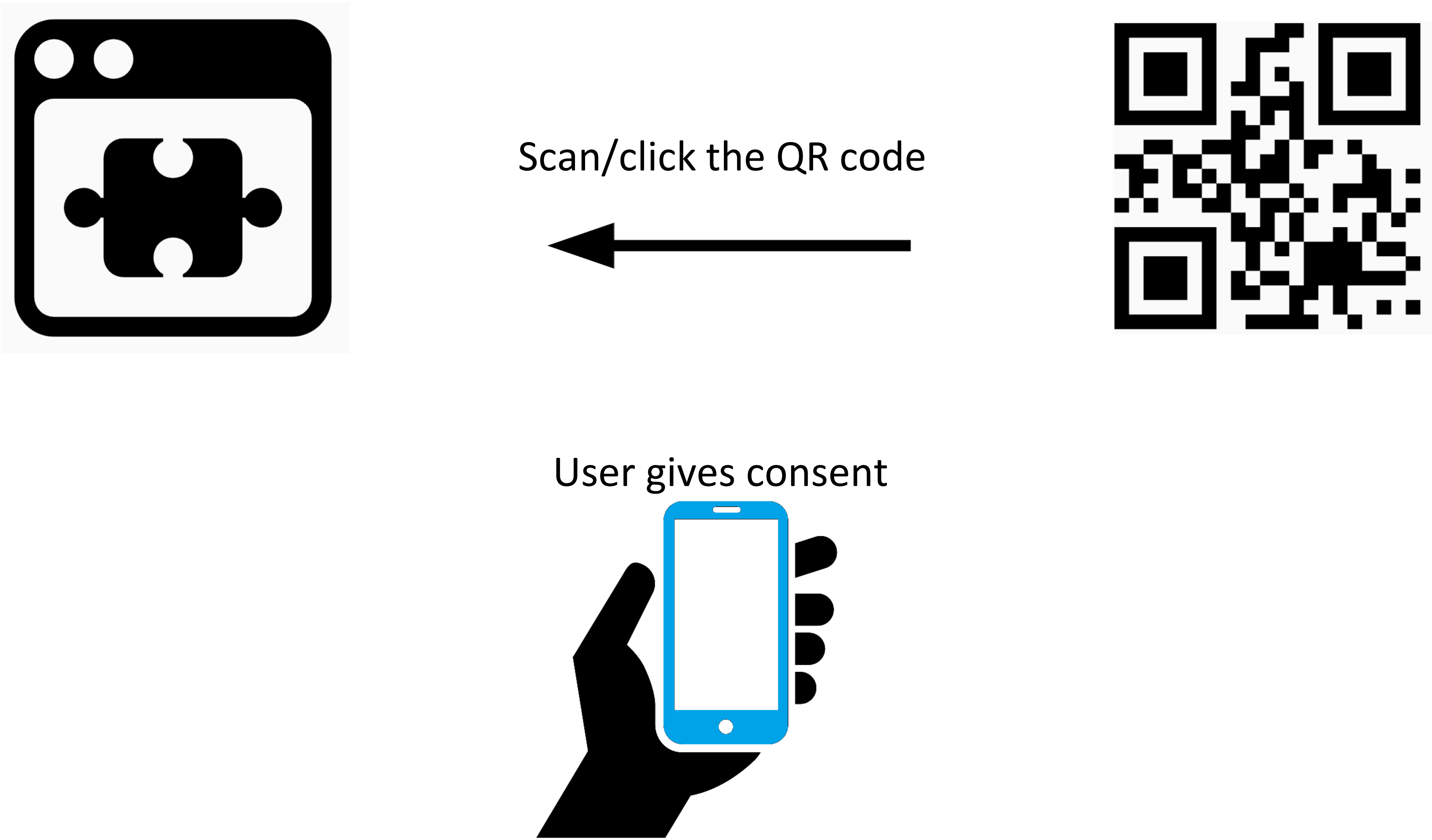
Privacy

# Let’s Authenticate Registration/Login


Client Device
Website
① Unique registration/login code
② Website address and code
④ Certificate chain authorizing login
⑤ Revocation status
Register or Login Approval
③ Certificate
Authenticator
Certificate Authority



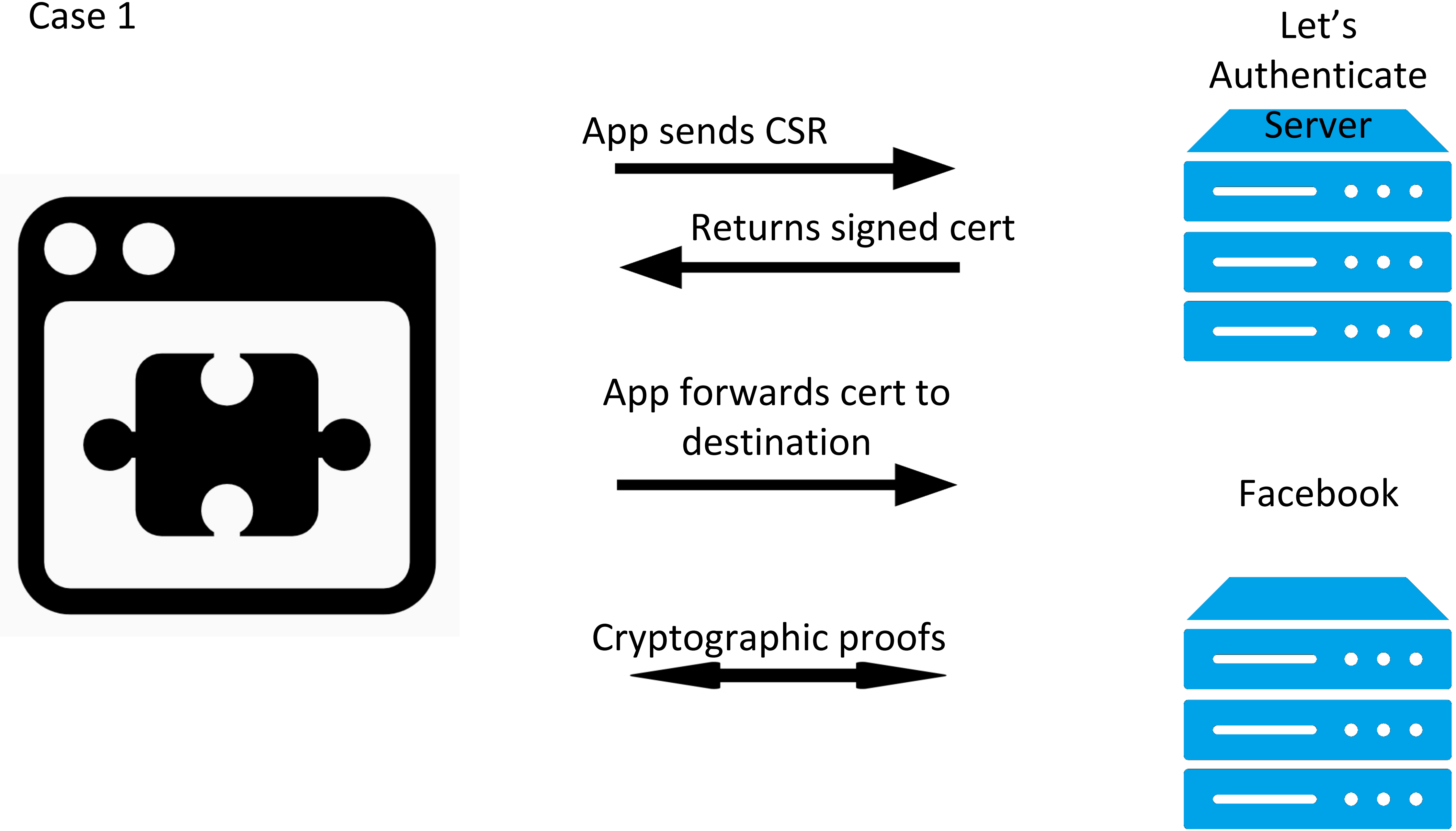
Username/Password



Scan/click the QR code

User gives consent

Case 1



Let’s Authenticate Server

App sends CSR

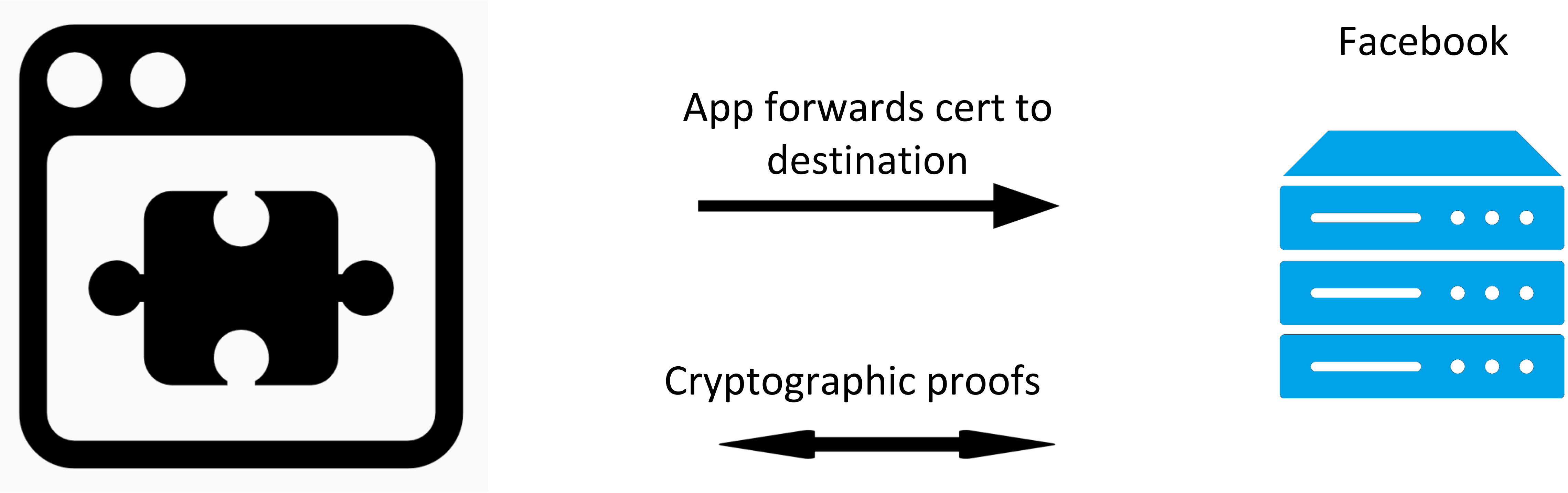
Returns signed cert

App forwards cert to destination

Facebook

Cryptographic proofs

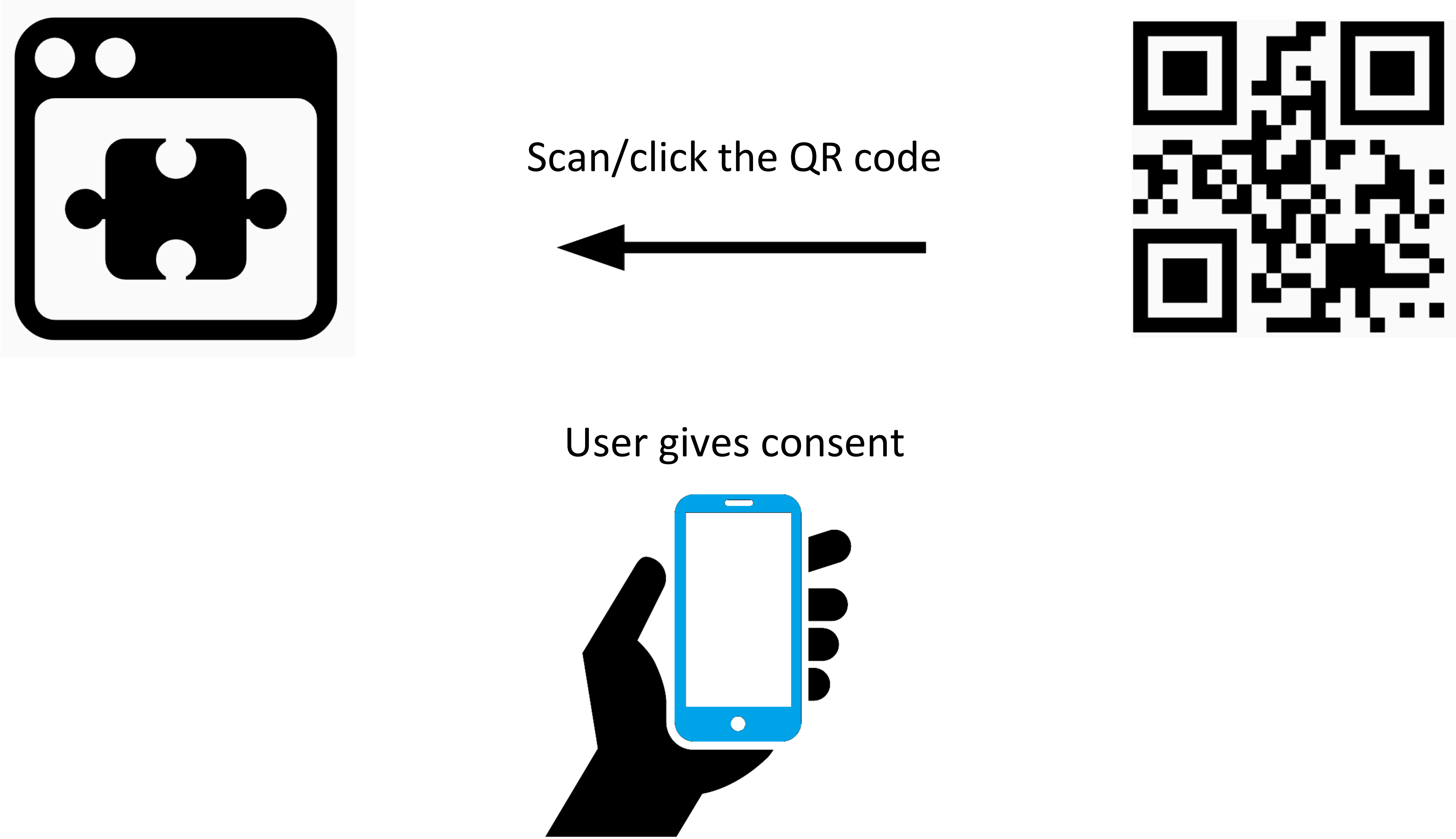
# Case 2



Facebook

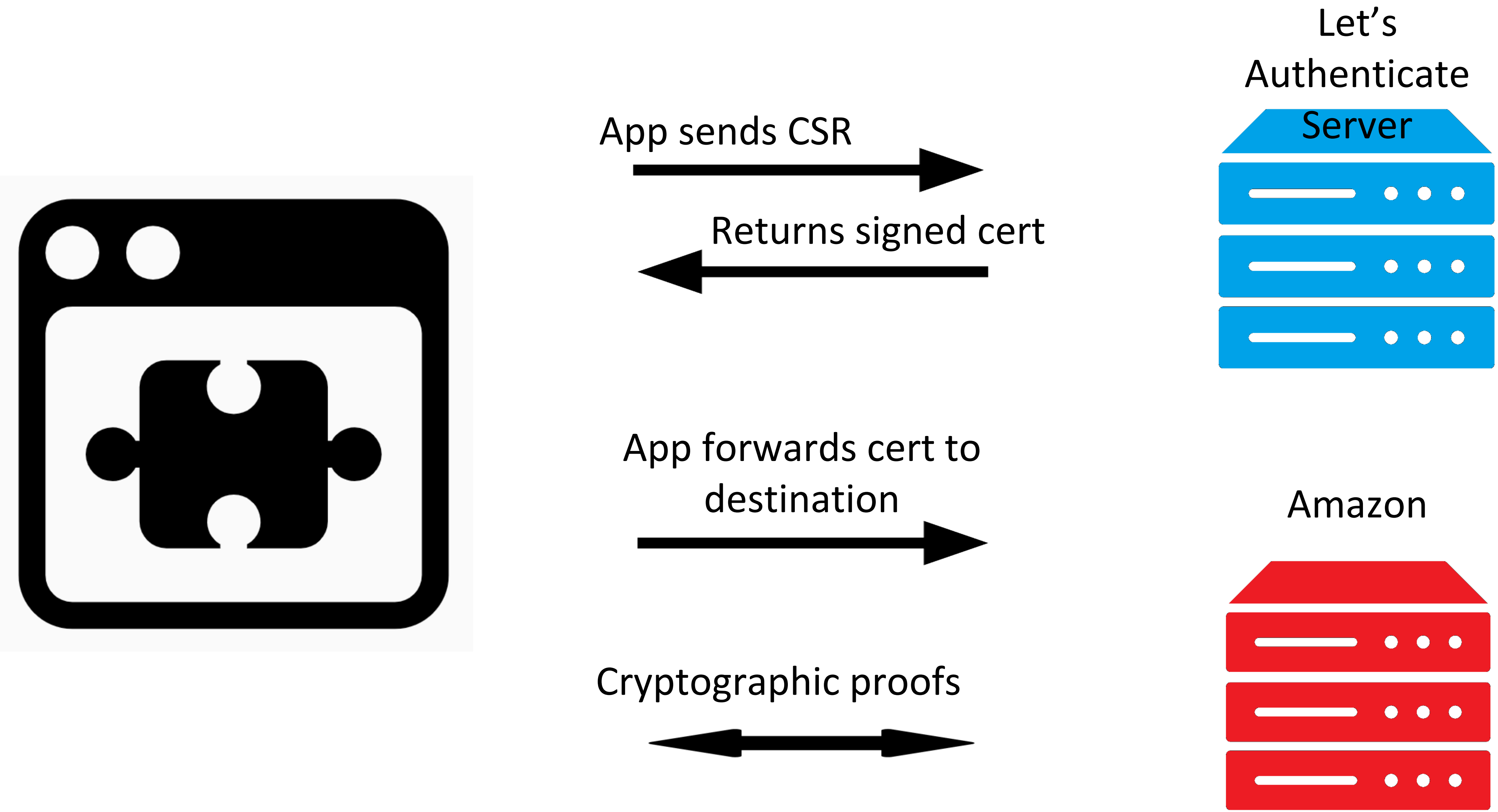
App forwards cert to destination

Cryptographic proofs



Scan/click the QR code

User gives consent



Let's Authenticate Server

App sends CSR

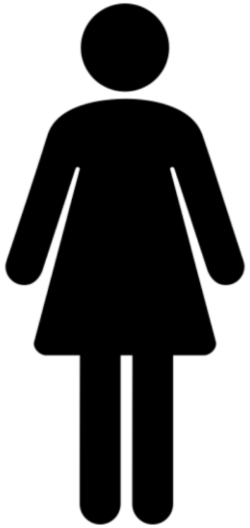
Returns signed cert

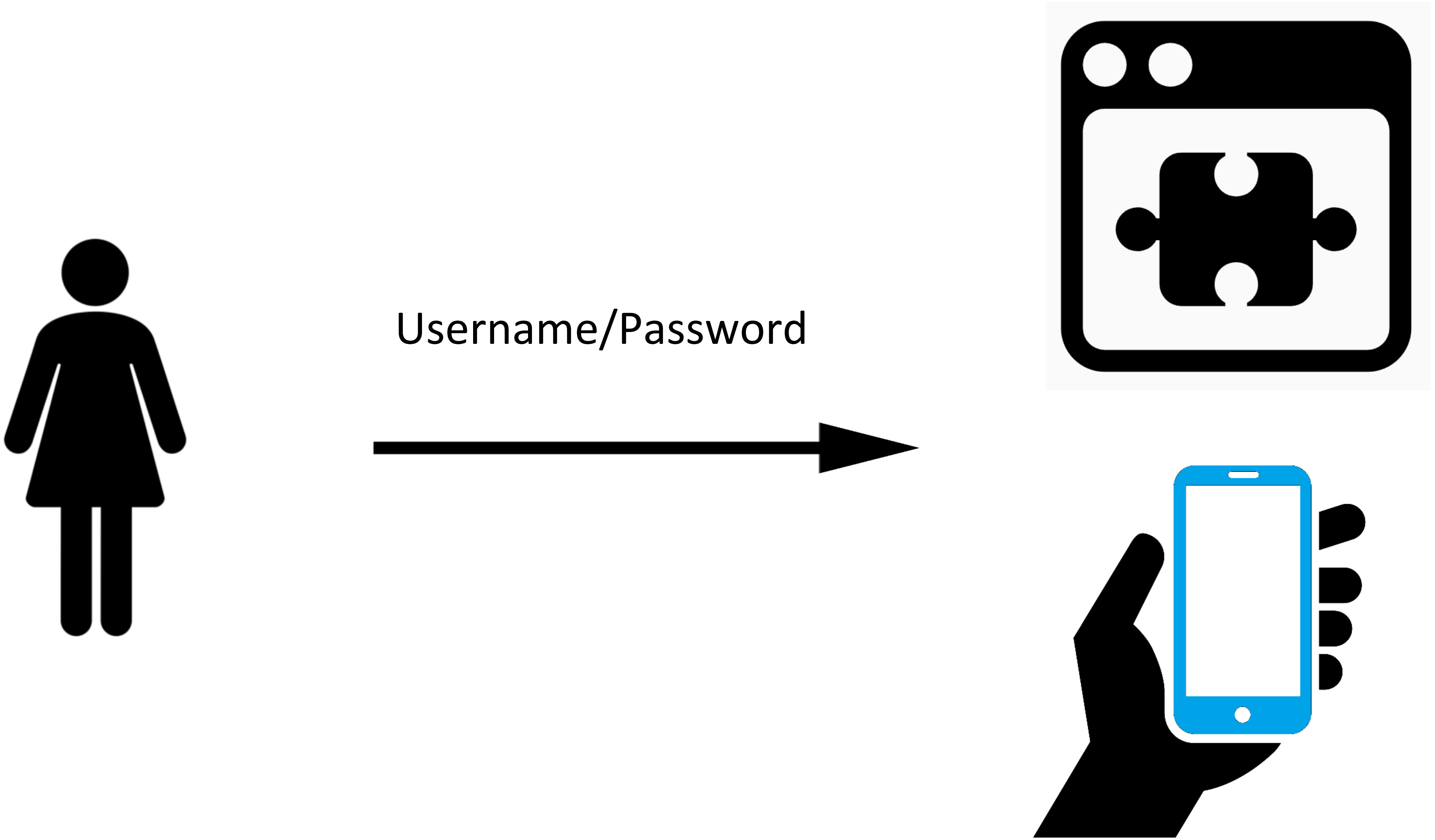
App forwards cert to destination

Amazon

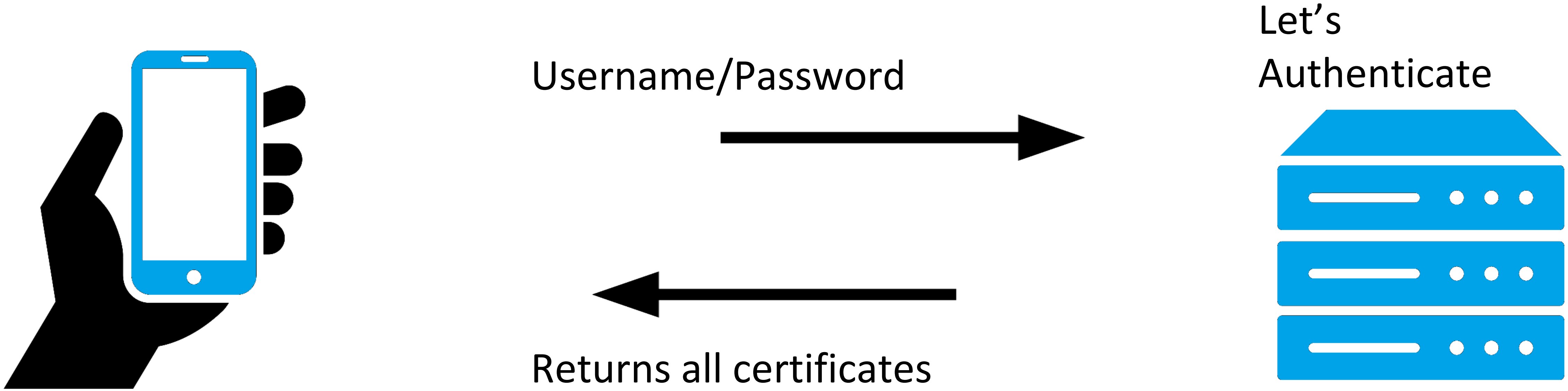
Cryptographic proofs

# What happens if Alice loses her authenticator?





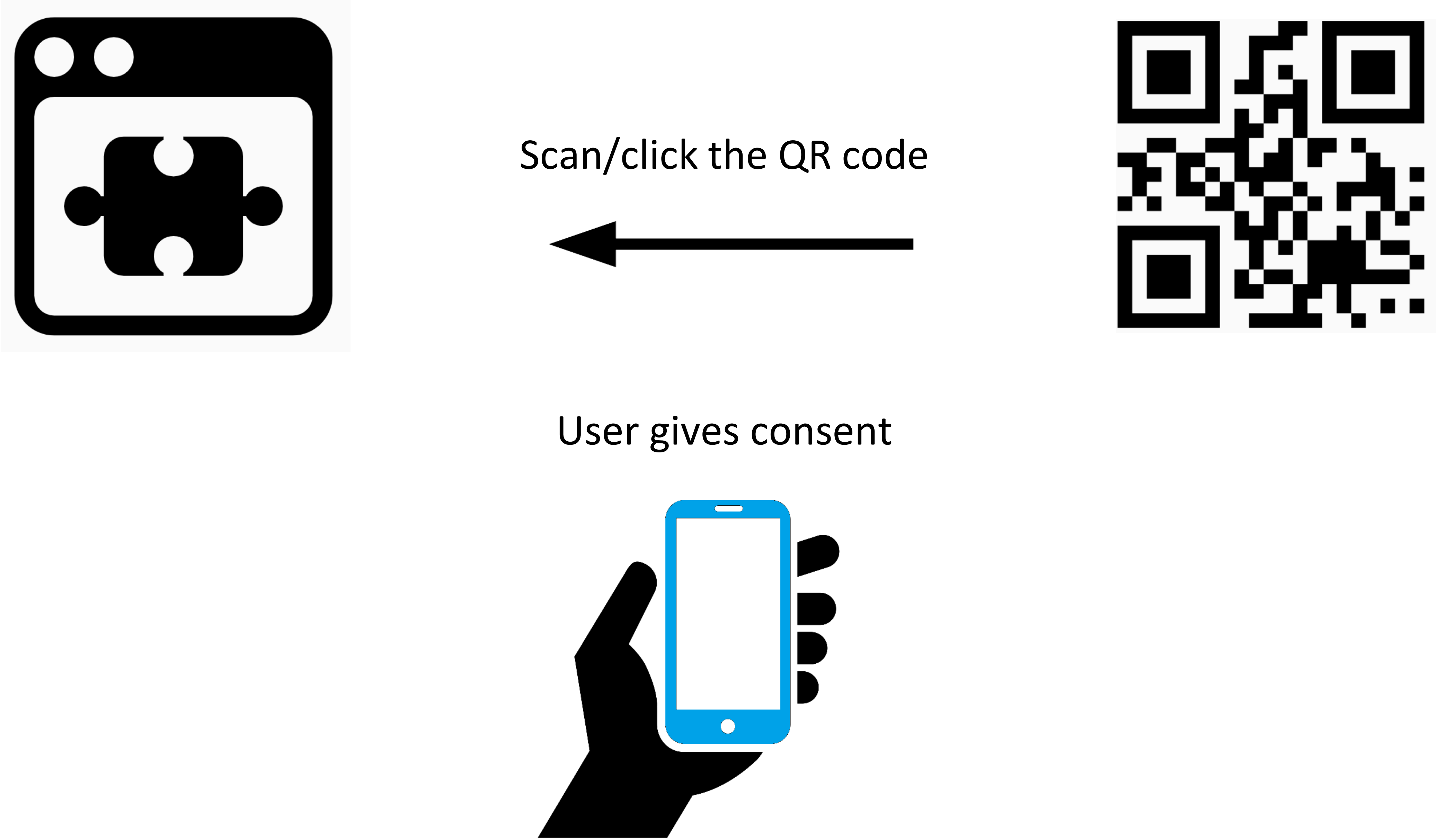
Username/Password



Username/Password

Let’s
Authenticate

Returns all certificates



Scan/click the QR code

User gives consent

# Privacy

- Want to avoid colluding websites tracking users
- Want to avoid giving Let’s Auth CA information about sites a user authenticates to

# Privacy

- Each certificate is bound to a unique email address: <uniquecode>@letsauth.org
- <uniquecode> = hash(username,password,websiteDomain,salt)
- Also makes it easy to reclaim accounts after lost authenticator

# Comparing Let’s Authenticate to WebAuthn

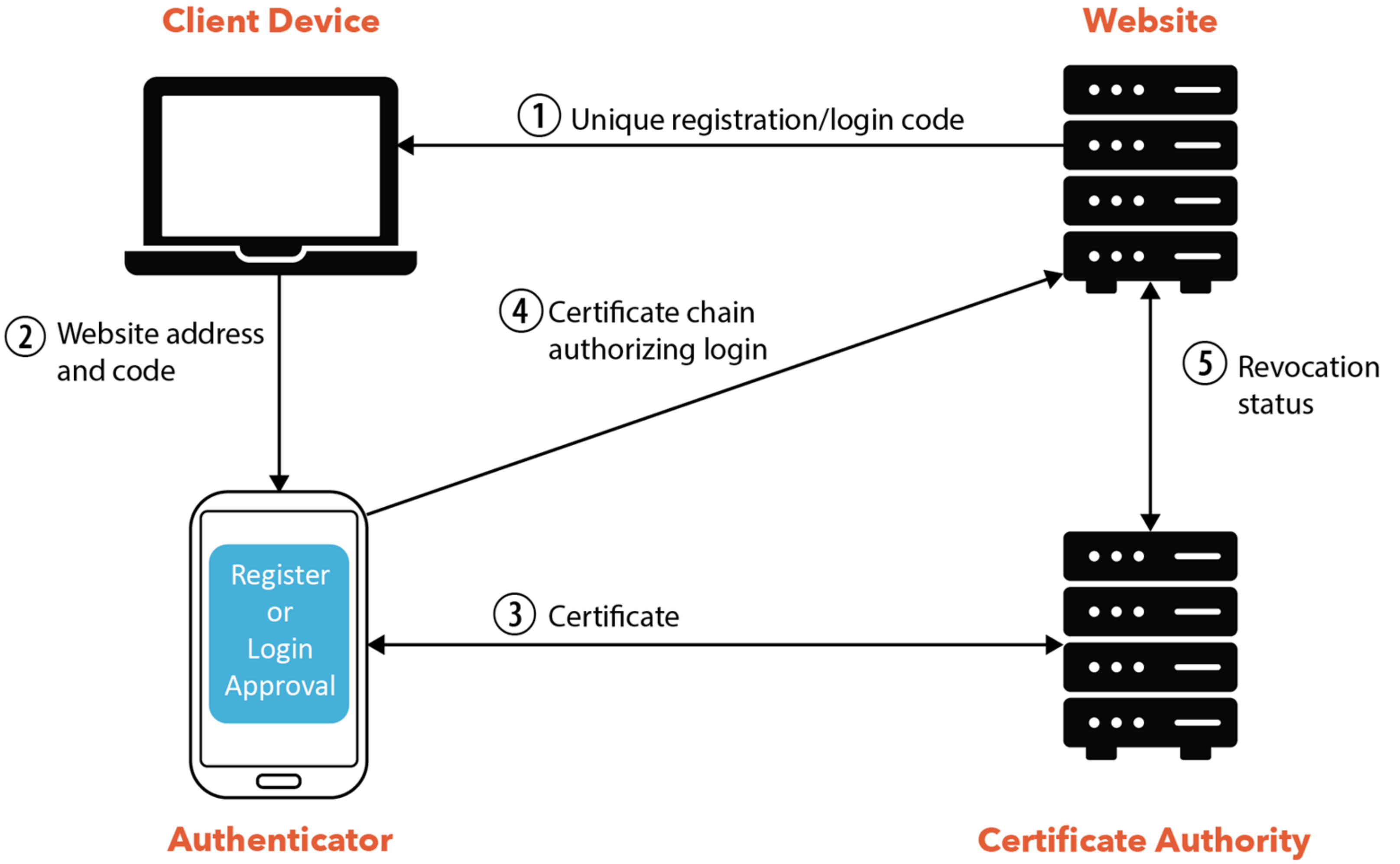
| System | *Memory-wise Effortless* | *Scalable for Users* | *Nothing to Carry* | *Easy to Learn* | *Efficient to Use* | *Easy Recovery from Loss* | *Negligible Cost to User* | *Server Compatible* | *Resilient to Phishing* | *Resilient to Theft* | *No Trusted Third Party* | *Unlinkable* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Passwords | | | ● | ● | ● | ● | ● | ● | | ● | ● | ● |
| LastPass | ○ | ● | ○ | ● | ● | ○ | ○ | ● | ● | ● | | ● |
| Mozilla Persona | ○ | ● | ● | ● | ● | ● | ● | | | ● | | |
| WebAuthn | ◖ | | ◖ | ● | ◖ | | ◖ | | ● | ◖ | ◖ | ◖ |
| Let’s Authenticate | ○ | ● | ◖ | ● | ● | ● | ● | | ● | ● | | ● |

● full support, ○ *quasi* support, ◖ support depends on the authenticator type or the website, (blank) no support

# What’s Next?

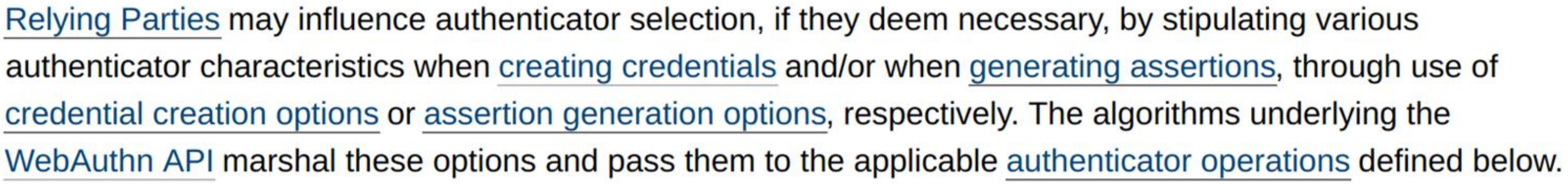
- In-depth Security and privacy analysis
- In lab and longitudinal user studies
- Exploration of different account challenges
- Consideration of short-lived certificates VS revocation

# Discussion

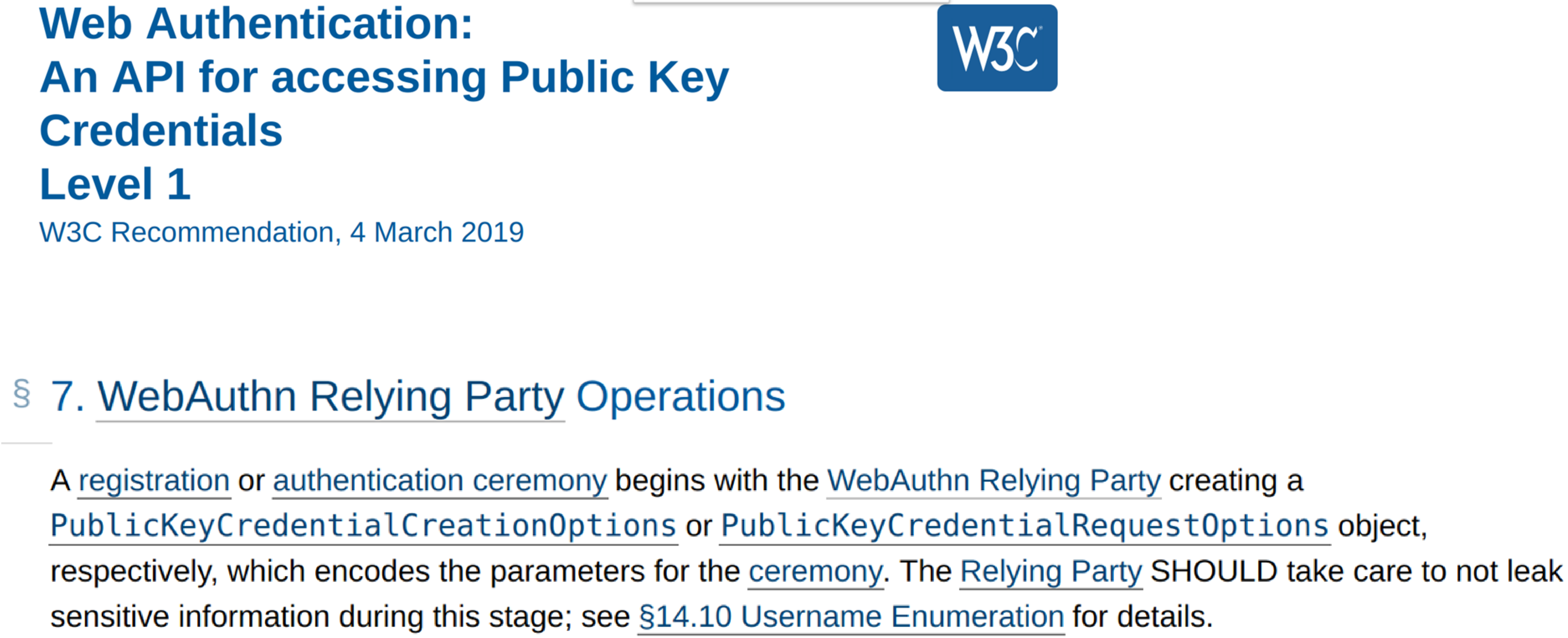

Client Device
Website
① Unique registration/login code
② Website address and code
④ Certificate chain authorizing login
⑤ Revocation status
Register or Login Approval
③ Certificate
Authenticator
Certificate Authority

# Web Authentication: An API for accessing Public Key Credentials Level 1

W3C Recommendation, 4 March 2019



Relying Parties may influence authenticator selection, if they deem necessary, by stipulating various authenticator characteristics when creating credentials and/or when generating assertions, through use of credential creation options or assertion generation options, respectively. The algorithms underlying the WebAuthn API marshal these options and pass them to the applicable authenticator operations defined below.

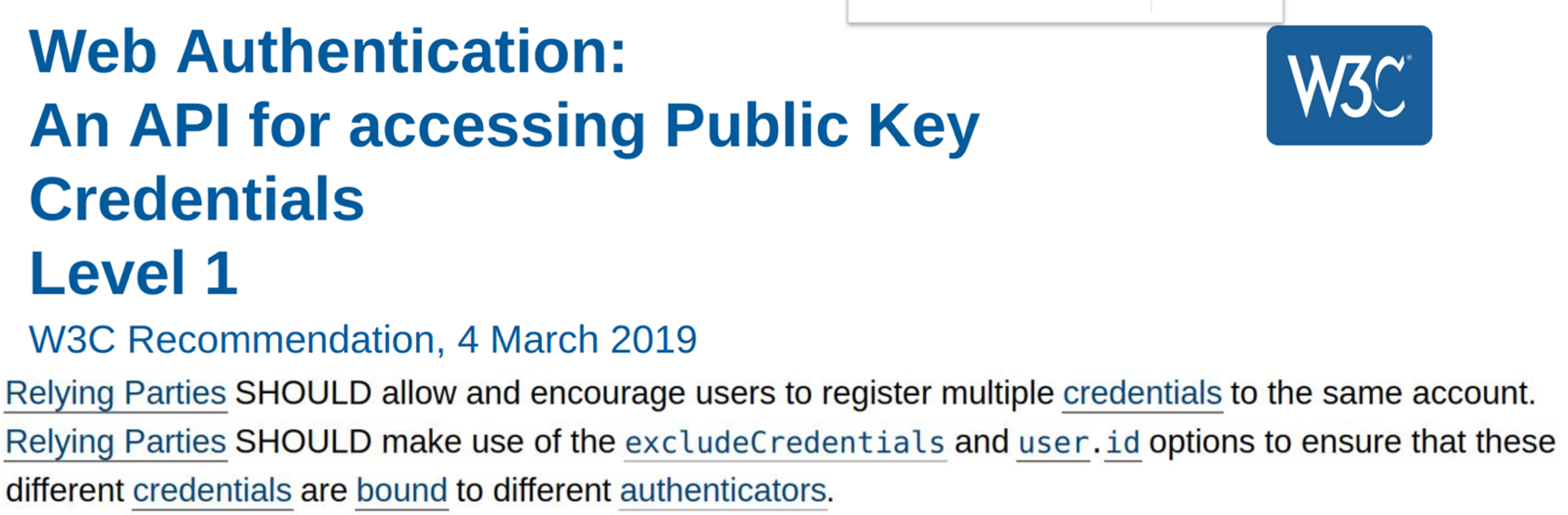


# Web Authentication: An API for accessing Public Key Credentials Level 1

W3C Recommendation, 4 March 2019

## § 7. WebAuthn Relying Party Operations

A registration or authentication ceremony begins with the WebAuthn Relying Party creating a `PublicKeyCredentialCreationOptions` or `PublicKeyCredentialRequestOptions` object, respectively, which encodes the parameters for the ceremony. The Relying Party SHOULD take care to not leak sensitive information during this stage; see §14.10 Username Enumeration for details.



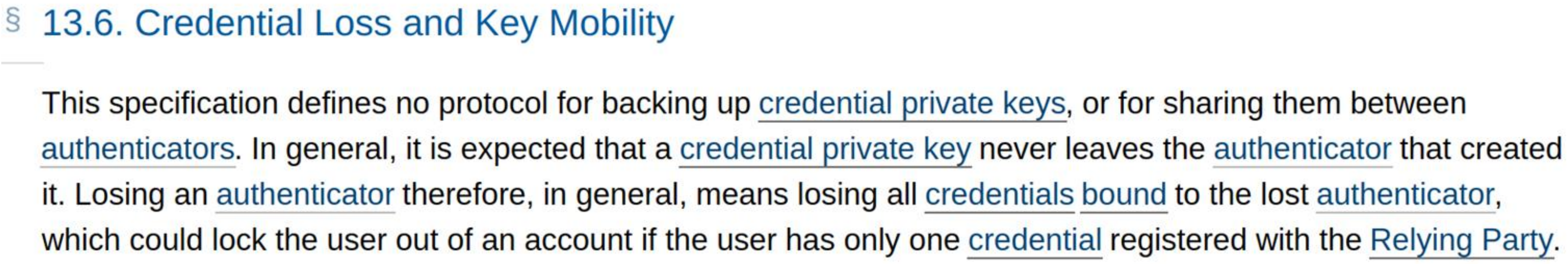
# Web Authentication: An API for accessing Public Key Credentials Level 1

W3C Recommendation, 4 March 2019

Relying Parties SHOULD allow and encourage users to register multiple credentials to the same account. Relying Parties SHOULD make use of the `excludeCredentials` and `user.id` options to ensure that these different credentials are bound to different authenticators.

# Web Authentication: An API for accessing Public Key Credentials Level 1

W3C Recommendation, 4 March 2019



## § 13.6. Credential Loss and Key Mobility

This specification defines no protocol for backing up credential private keys, or for sharing them between authenticators. In general, it is expected that a credential private key never leaves the authenticator that created it. Losing an authenticator therefore, in general, means losing all credentials bound to the lost authenticator, which could lock the user out of an account if the user has only one credential registered with the Relying Party.

# Persona

- Allowed email providers to issue certificates to a user
- Simpler registration process since their email was verified
- Tracking still possible, unless a user creates a different email for each service
- Adoption was an issue as well
  - 4 entities of adoption (Users, Websites, Browsers and Email providers)
- They did provide a fallback identity provider and a cross-browser library, but they were short term solutions