

# Information Security

## Networking 4: Your Topic Here

Winter 2022/2023



Jakob Heher, [www.iaik.tugraz.at](http://www.iaik.tugraz.at)

he/his

# Lecture ground rules

- We color technologies, algorithms, etc. for your convenience
  - State-of-the-art tech, no known vulnerabilities ✓
    - This is generally safe to use!
  - Outdated tech, known issues, covered for demonstration purposes ✗
    - You should not use this!
- Coloring provides a very quick-and-dirty categorization for you
  - Want to know *why*? That's what the lecture is for 😊

Recall from last time

# Meet the players



Alice  
she/hers



Bob  
he/his

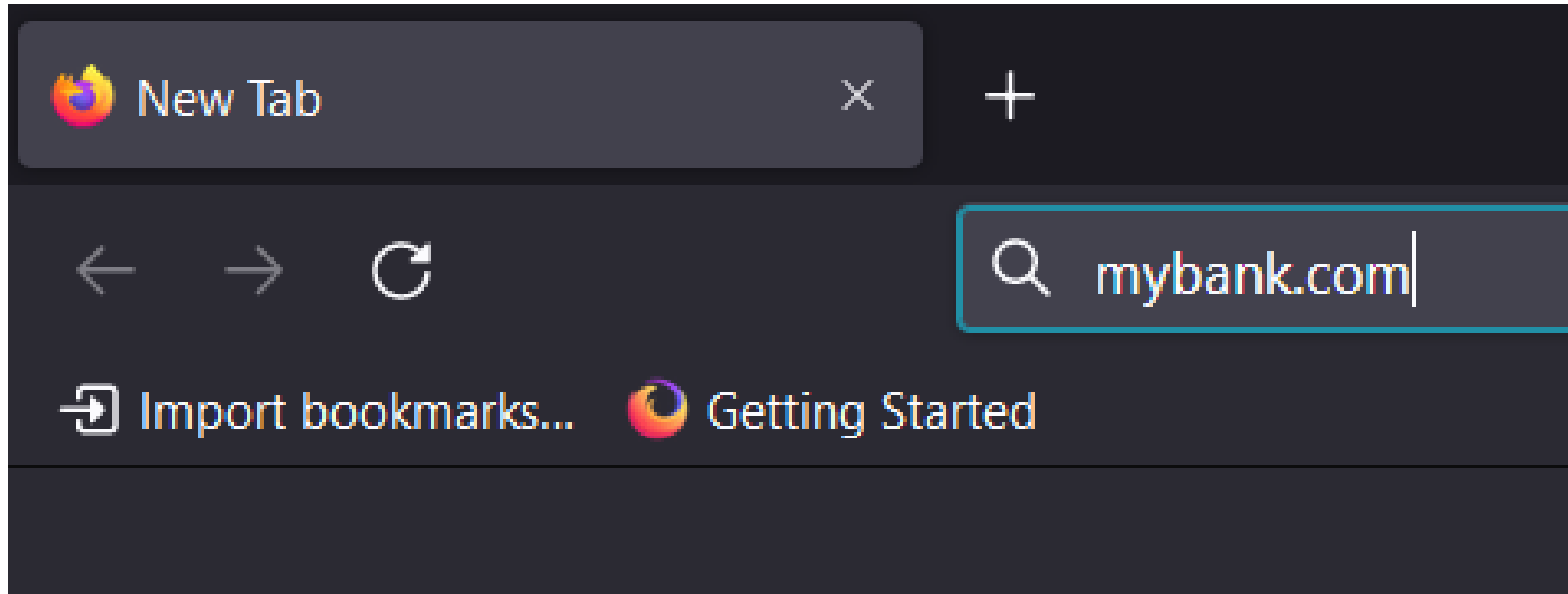


Eve  
????



Smith  
she/hers

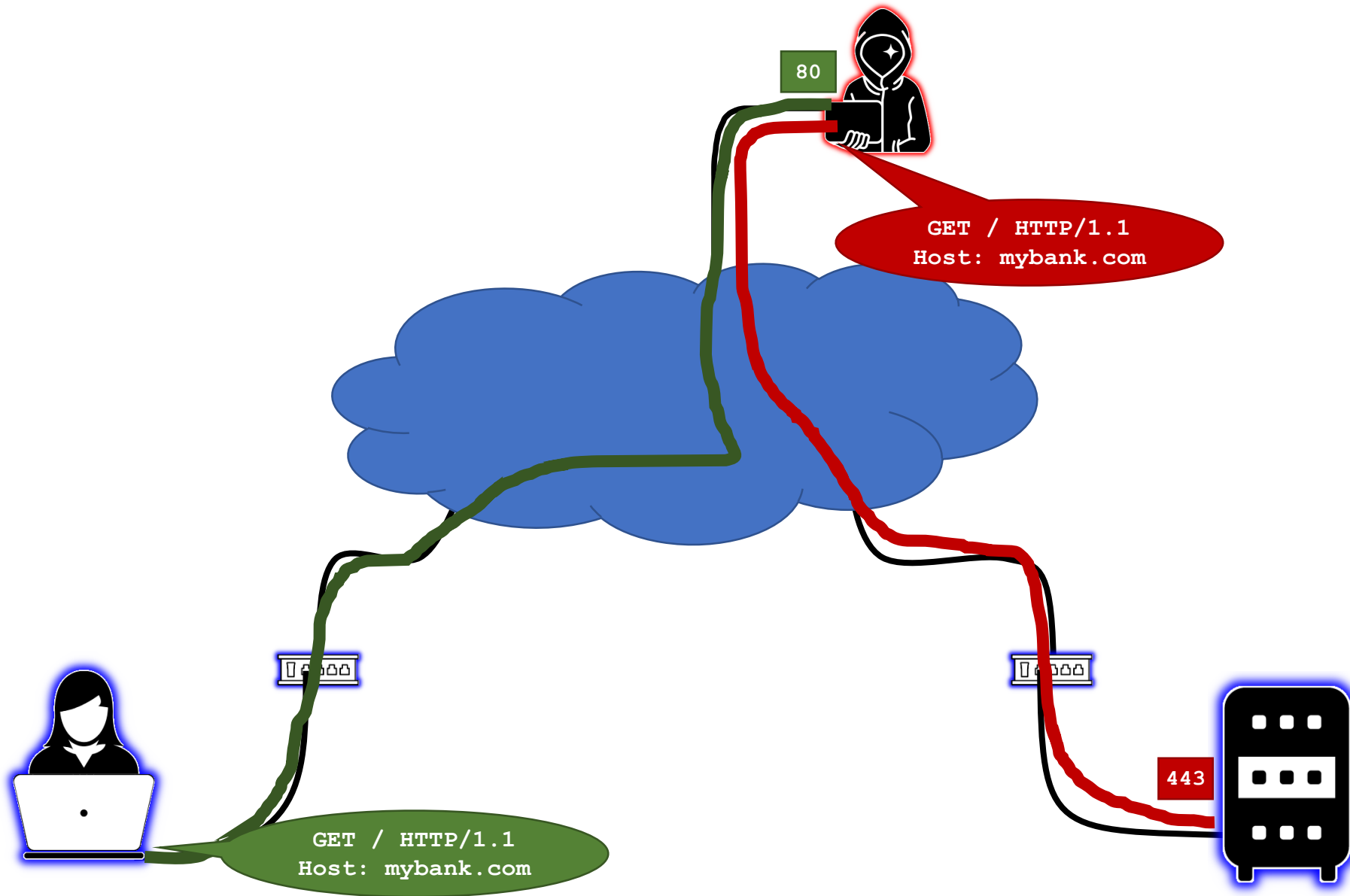
# HTTP security

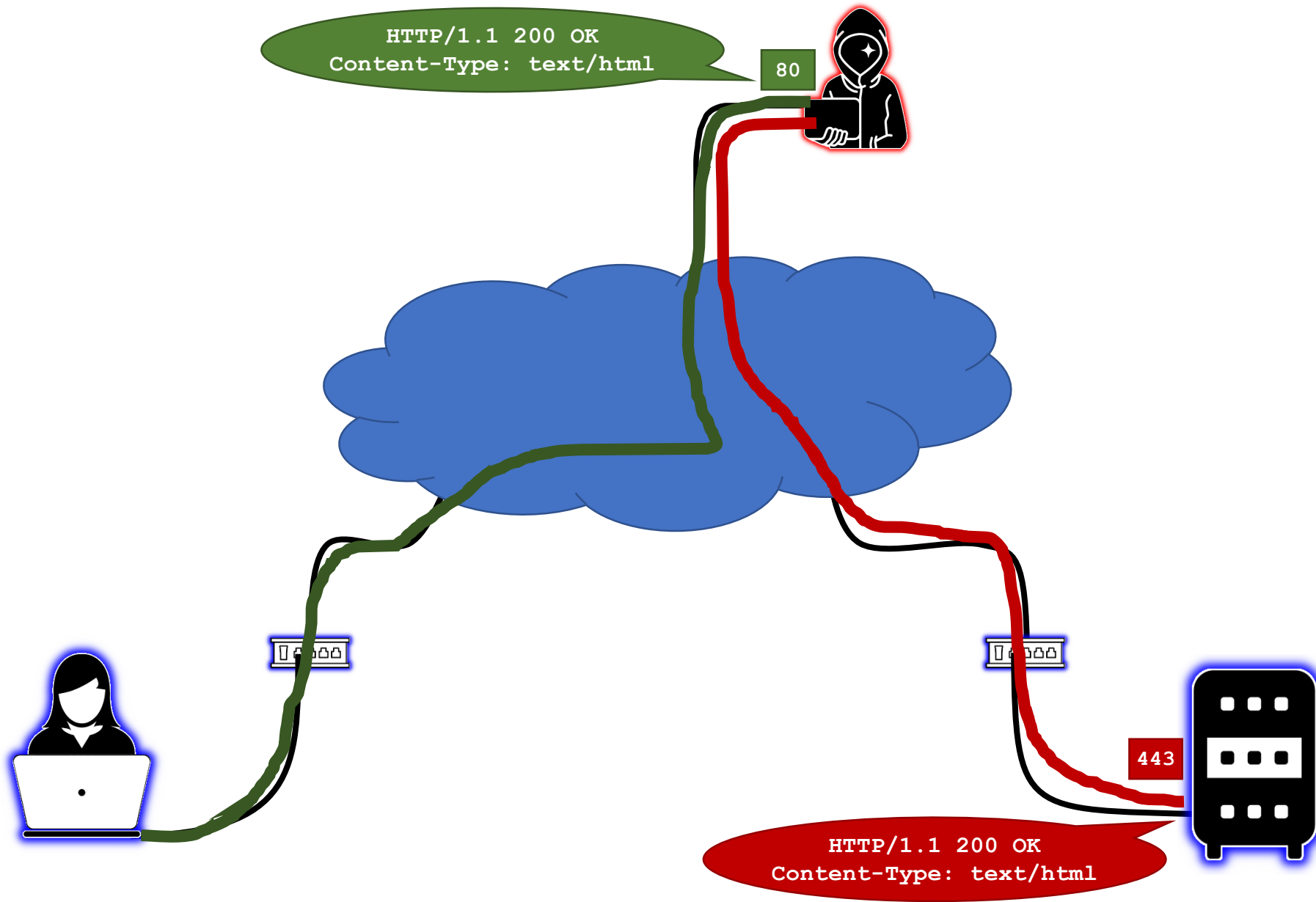


- What happens?

- DNS request for **mybank . com** -> some IP address
- Open connection to IP address on port 80
- **First HTTP response:** redirect to **https : //mybank . com/**







# HTTP **Strict-Transport-Security**



- The initial HTTP request defeats the security of the connection!
  - Only indicator of compromise: lack of HTTPS in the URL
- ✓ **Strict-Transport-Security** header
  - Server: “Only speak to me using HTTPS from now on”
  - Browser will default to HTTPS for this domain
- This still leaves the first visit vulnerable!



# HTTPS-only mode



**HTTPS-Only Mode**

HTTPS provides a secure, encrypted connection between Firefox and the websites you visit. Most websites support HTTPS, and if HTTPS-Only Mode is enabled, then Firefox will upgrade all connections to HTTPS.

[Learn more](#)

Enable HTTPS-Only Mode in all windows Manage Exceptions...

Enable HTTPS-Only Mode in private windows only


Don't enable HTTPS-Only Mode

Always use secure connections

Upgrade navigations to HTTPS and warn you before loading sites that don't support it

- New in 2022: available in all major browsers
  - Disabled by default (turn it on!)

# HTTPS-only mode



**HTTPS-Only Mode Alert**  
**Secure Site Not Available**

You've enabled HTTPS-Only Mode for enhanced security, and a HTTPS version of **mybank.com** is not available.  
[Learn More...](#)

**What could be causing this?**

- Most likely, the website simply does not support HTTPS.
- It's also possible that an attacker is involved. If you decide to visit the website, you should not enter any sensitive information like passwords, emails, or credit card details.

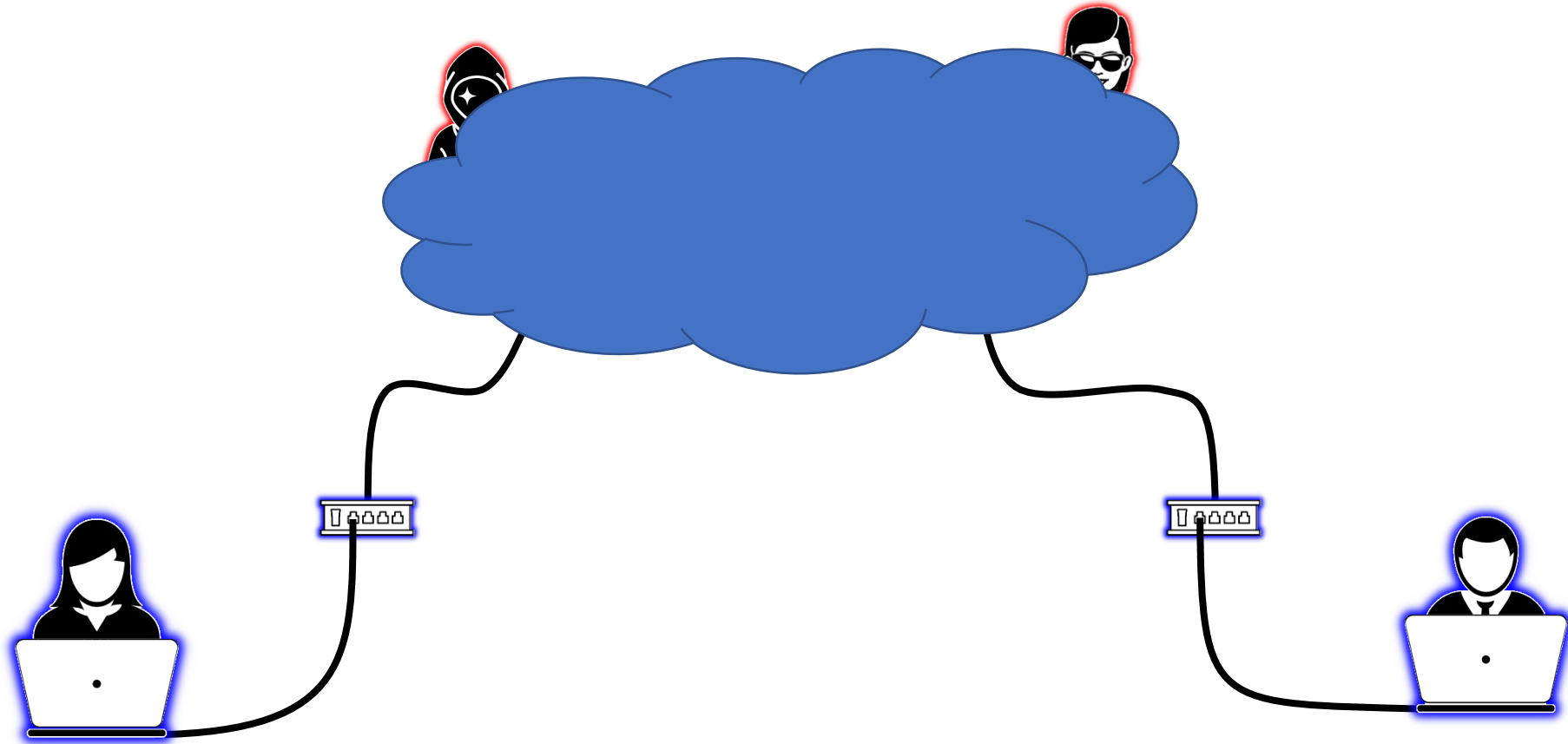
If you continue, HTTPS-Only Mode will be turned off temporarily for this site.

[Continue to HTTP Site](#) [Go Back](#)

# Virtual Private Networks

# Lower Layers – Recap

- Excellent at reliably delivering your data if everyone cooperates
- Not so excellent in the face of malicious actors
- Take-aways:
  - You cannot inherently trust that you are talking to the right person
  - You cannot inherently trust that your data is confidential
  - You cannot inherently trust that your data is unaltered
- The application layer has to take care of these things!



The Internet is a scary place...

# When you send a packet of data...

- ... it bounces between some unspecified number ...
  - ... of unknown routers somewhere on the internet ...
  - ... before you get a reply from an unknown entity ...
  - ... which claims to have the IP address you were asking for ...
- 
- ... and anyone along the way could be up to no good ...
  - ... with no safeguards.

Recall from forever ago

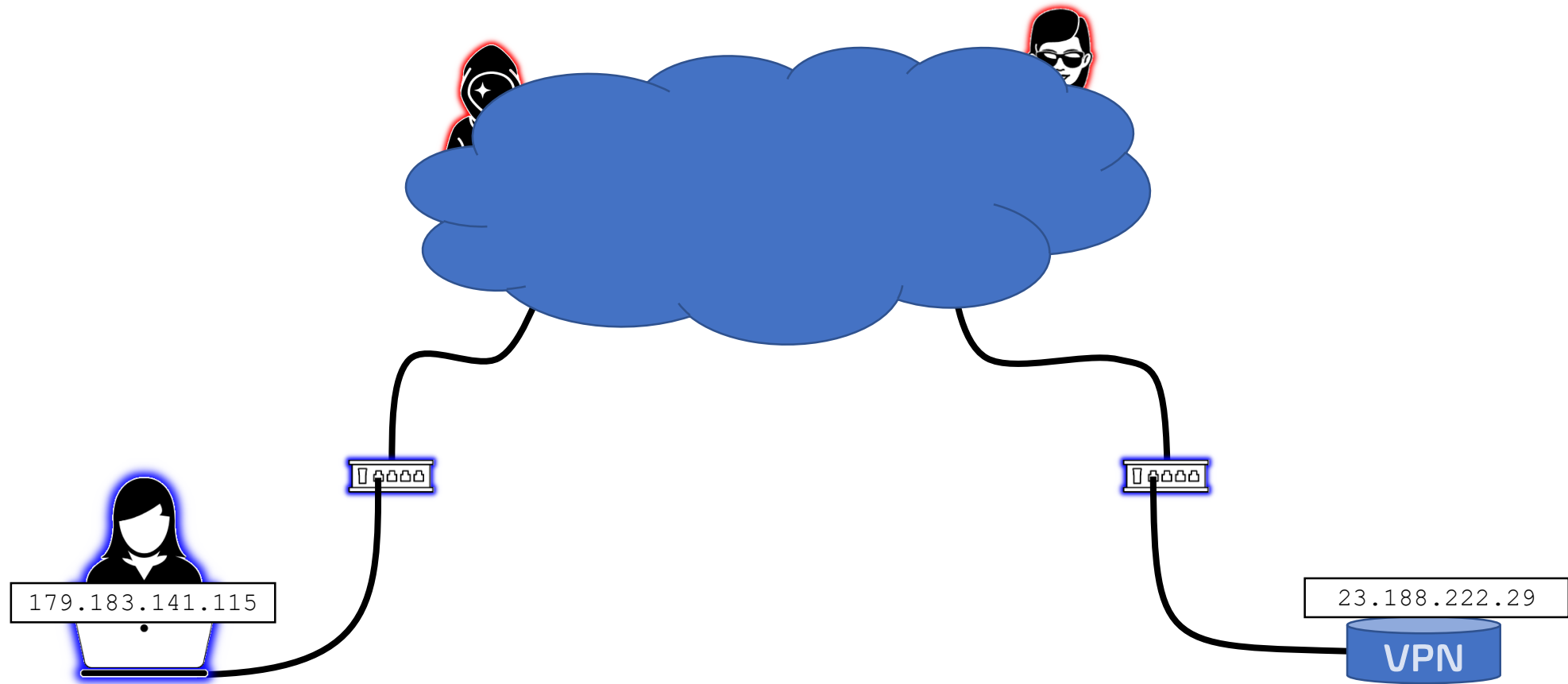
# Lower Layers – Recap

- Excellent at reliably delivering your data if everyone cooperates
- Not so excellent in the face of malicious actors
- Take-aways:
  - You cannot inherently trust that you are talking to the right person
  - You cannot inherently trust that your data is confidential
  - You cannot inherently trust that your data is unaltered
- **The application layer has to take care of these things!**

And what if it doesn't?

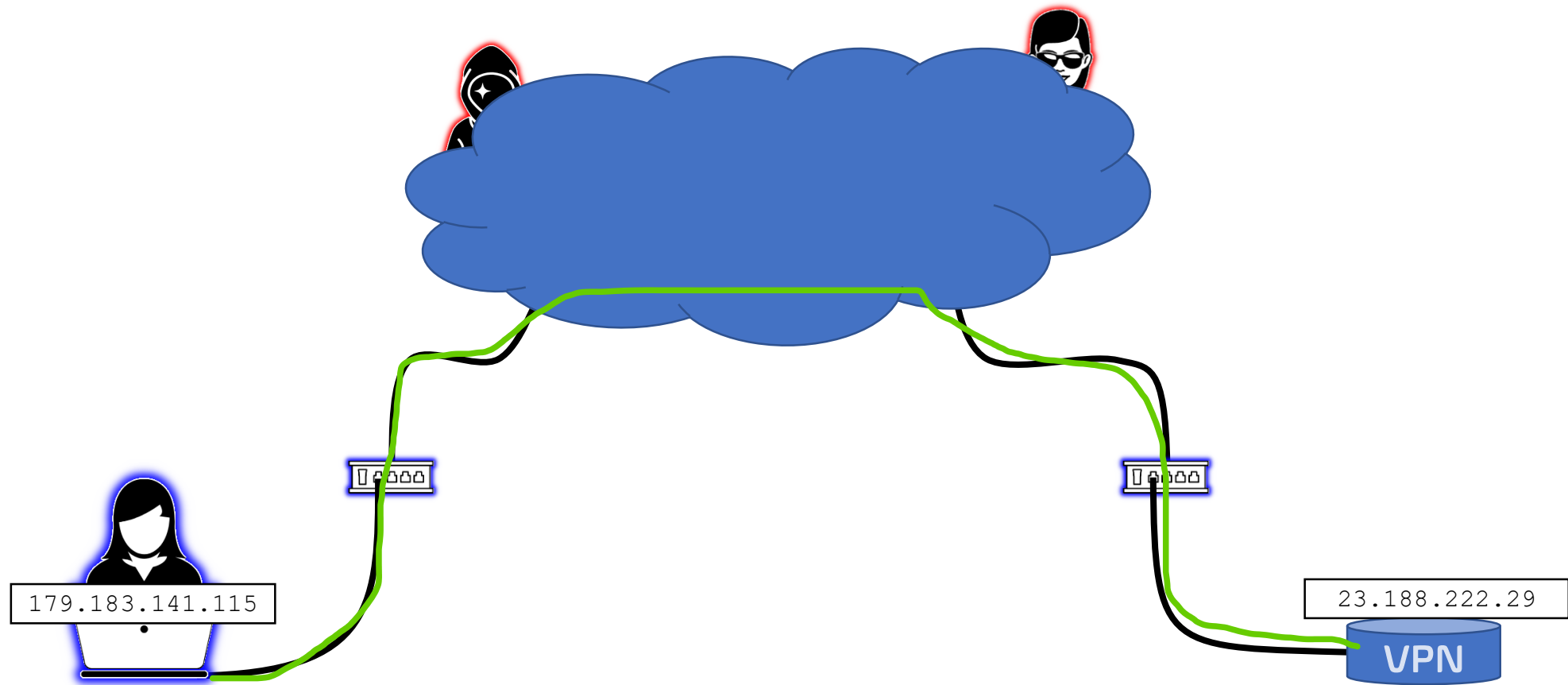
# Virtual Private Networks





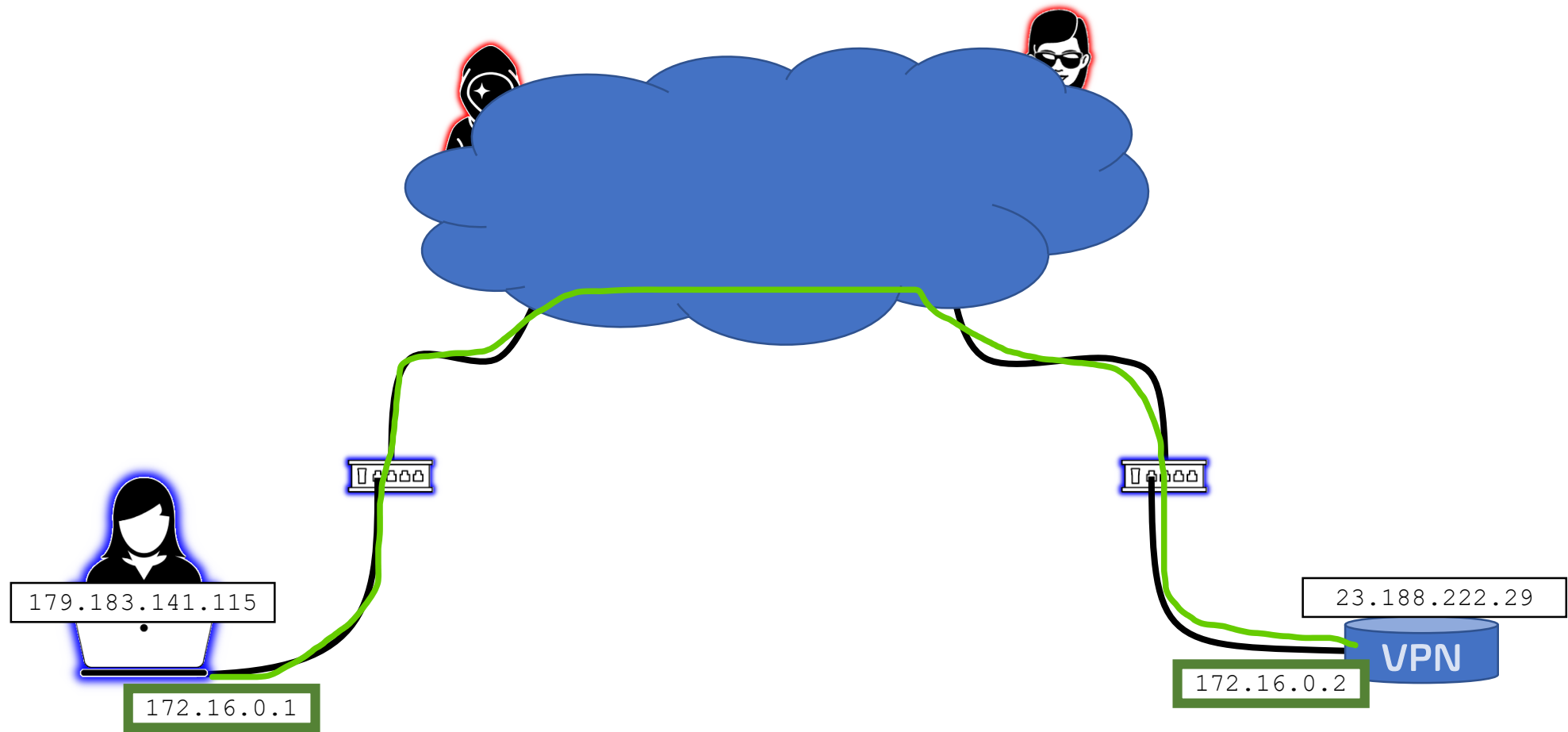
# Virtual Private Networks

1. Establish an encrypted, authenticated tunnel at the application layer



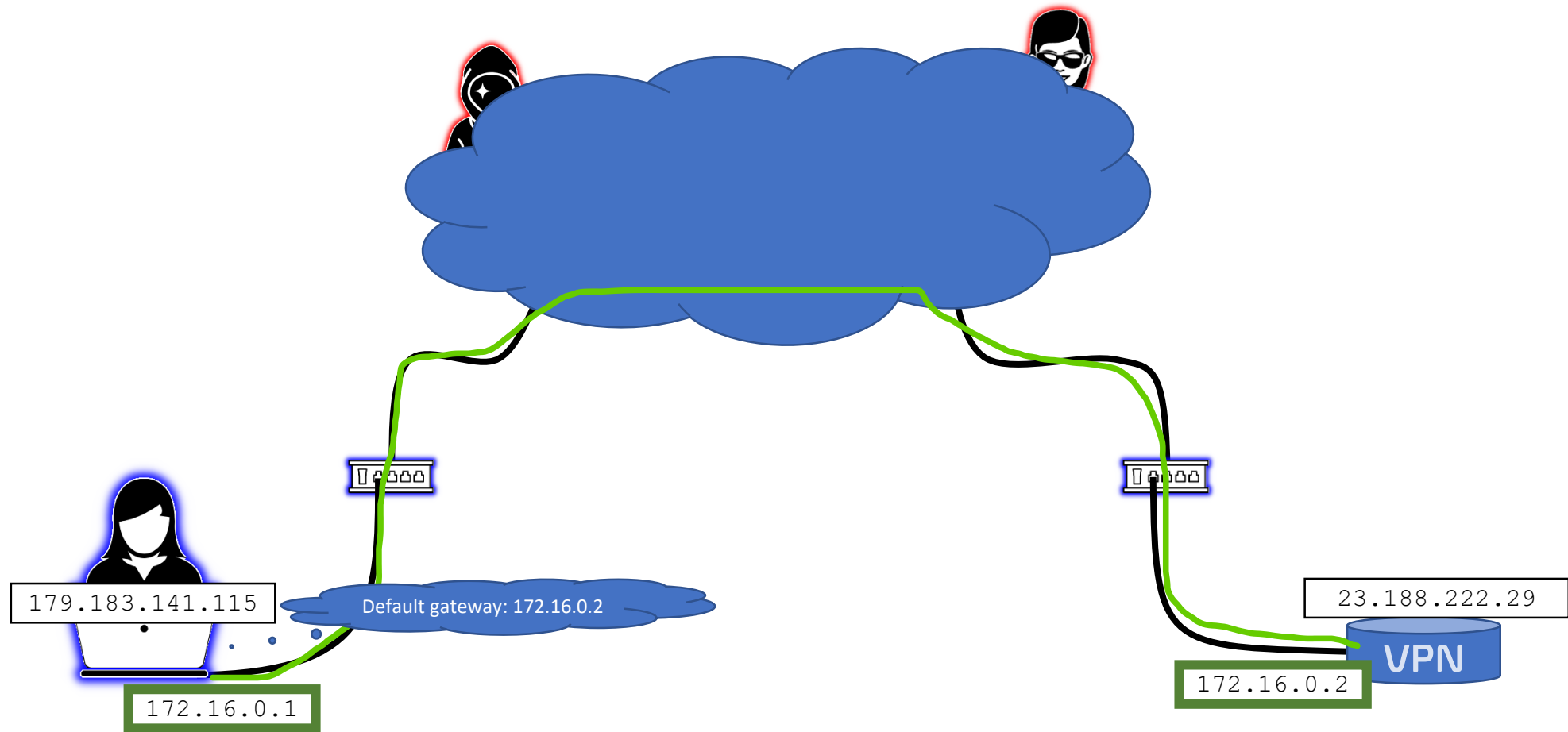
# Virtual Private Networks

1. Establish an encrypted, authenticated tunnel at the application layer
2. Slap addresses on each end and pretend it's a "real" network card



# Virtual Private Networks

1. Establish an encrypted, authenticated tunnel at the application layer
2. Slap addresses on each end and pretend it's a "real" network card
3. Re-configure the routing table so it's used to send your traffic



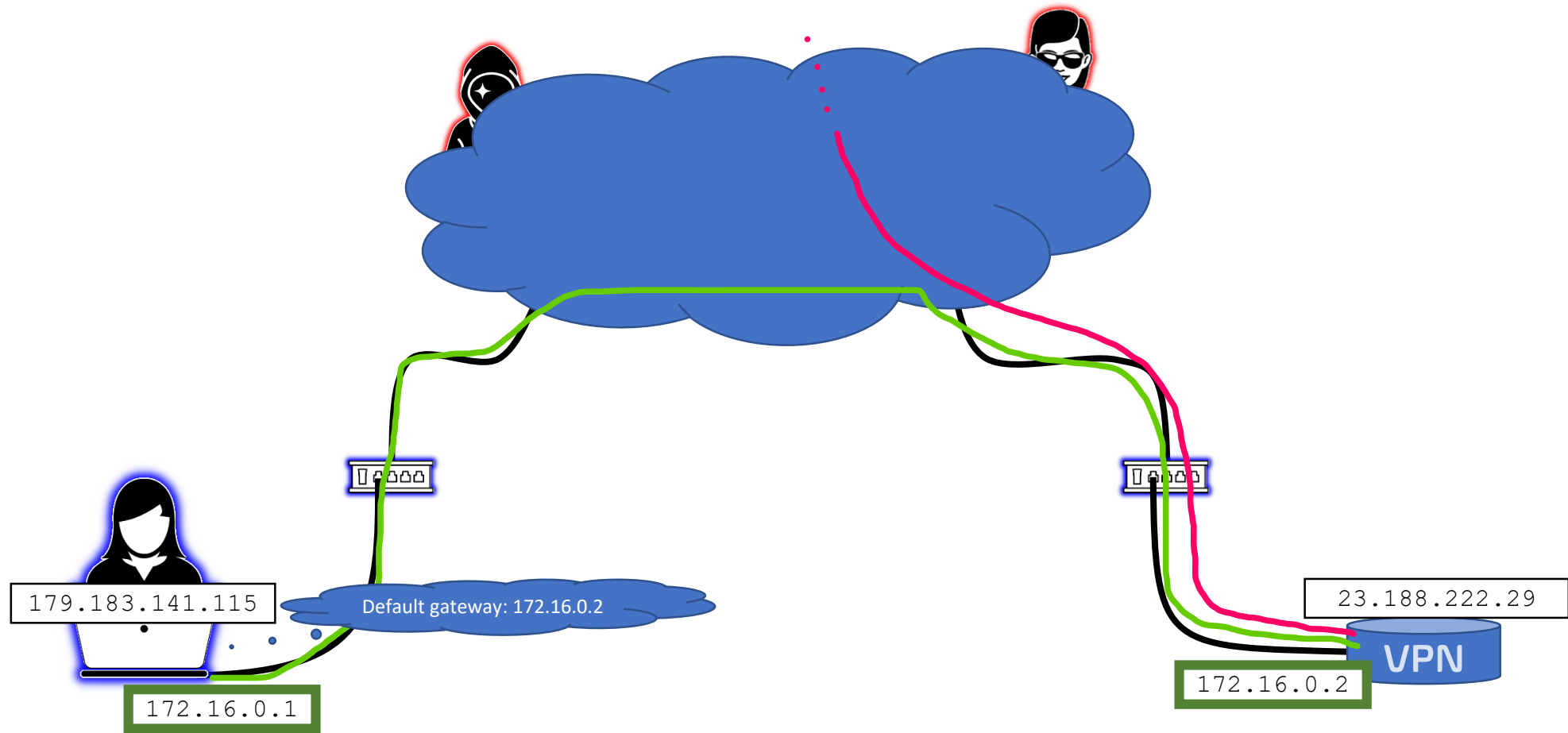
# Virtual Private Networks

1. Establish an encrypted, authenticated tunnel at the application layer
2. Slap addresses on each end and pretend it's a "real" network card
3. Re-configure the routing table so it's used to send your traffic
4. Traffic is transparently secured by the application-layer software
  
5. OK, the traffic is at the other side of the tunnel. Now what?



# Virtual Private Networks

- Scenario 1: traffic goes somewhere on the internet

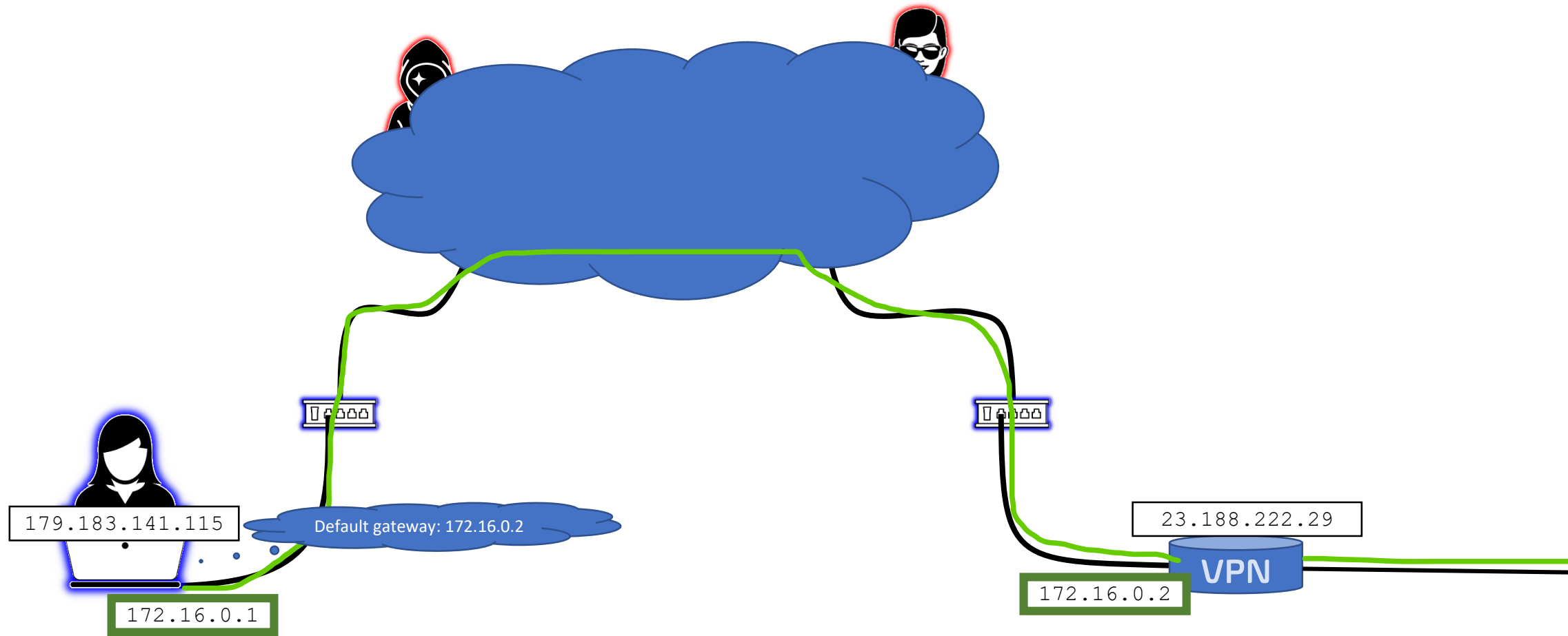


# Virtual Private Networks

- Scenario 1: traffic goes somewhere on the internet
- You are protected against:
  - Many local attacks
  - Logging from your ISP
- You are *not* protected against:
  - Any internet-based attacks
  - Bulk data inspection
- You are *slightly* impeding:
  - Figuring out the connection's source

# Virtual Private Networks

- Scenario 2: traffic stays inside the target network



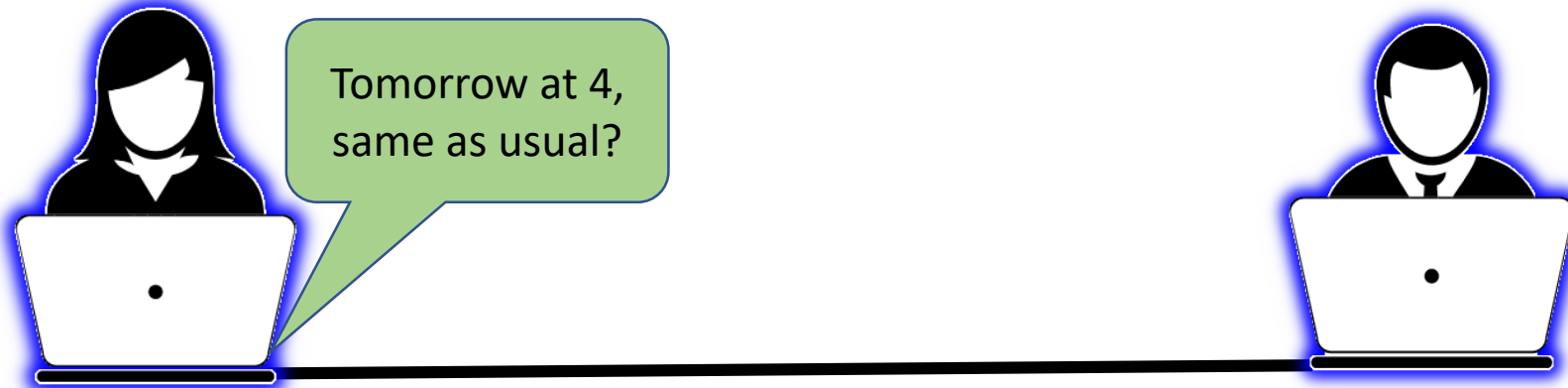
# Virtual Private Networks

- Scenario 2: traffic stays inside the target network
- You are protected against:
  - Many local attacks
  - Logging from your ISP
  - Internet-based traffic interception or monitoring
  - Routing attacks
- You are not protected against:
  - Metadata monitoring
  - Denial-of-service attacks

# Privacy & Metadata

# Metadata?

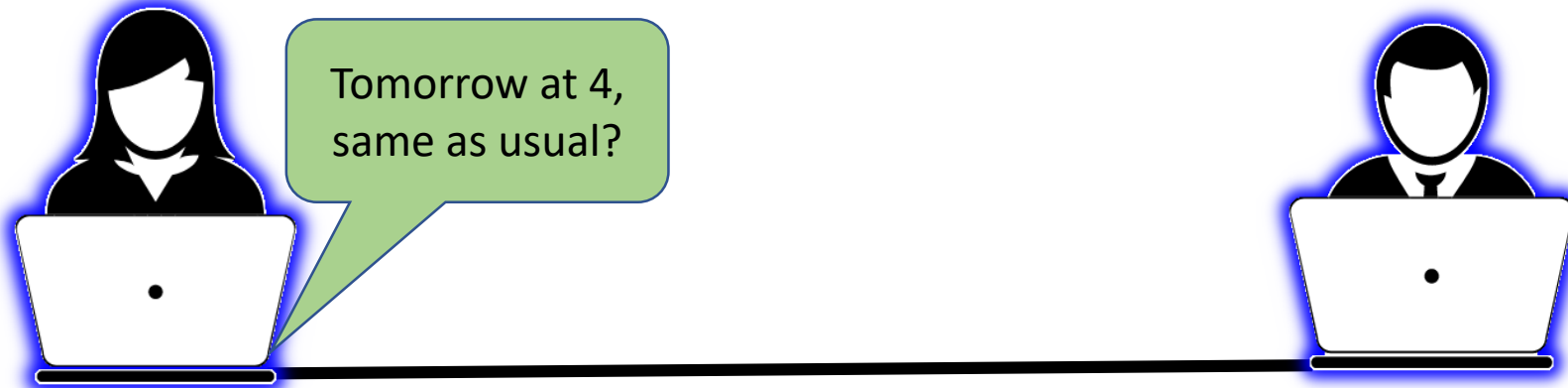
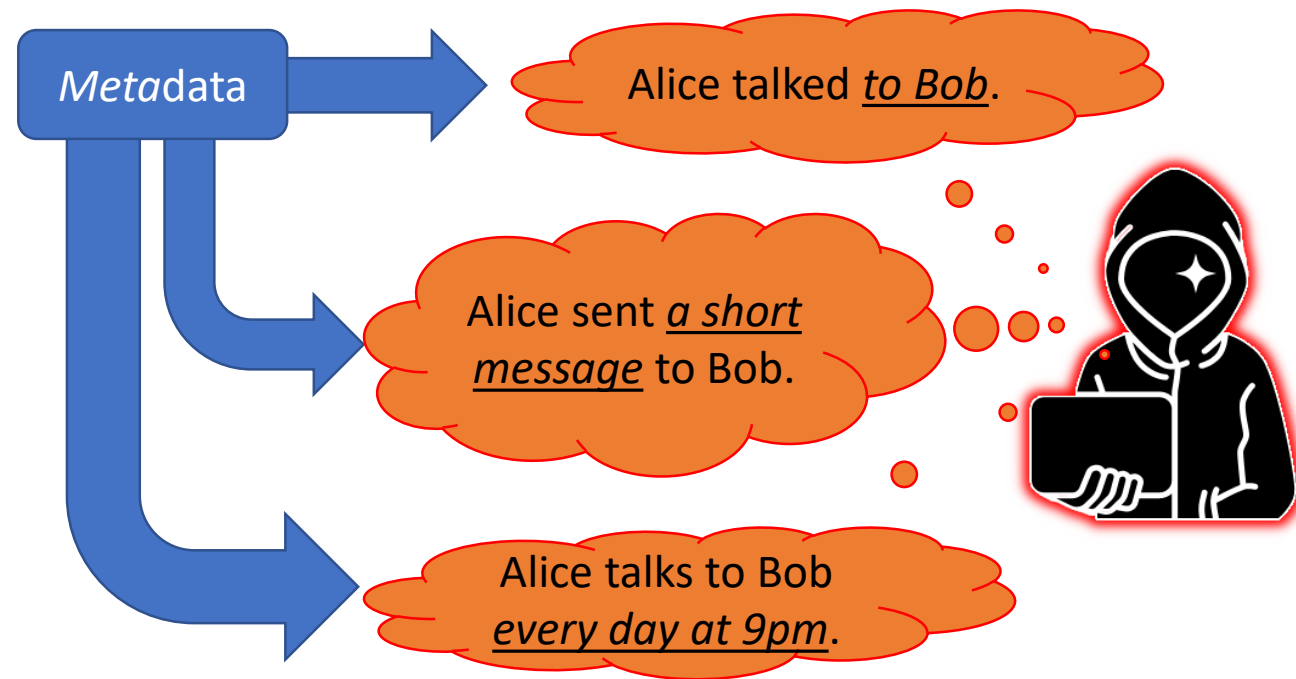
- Data *about other data*





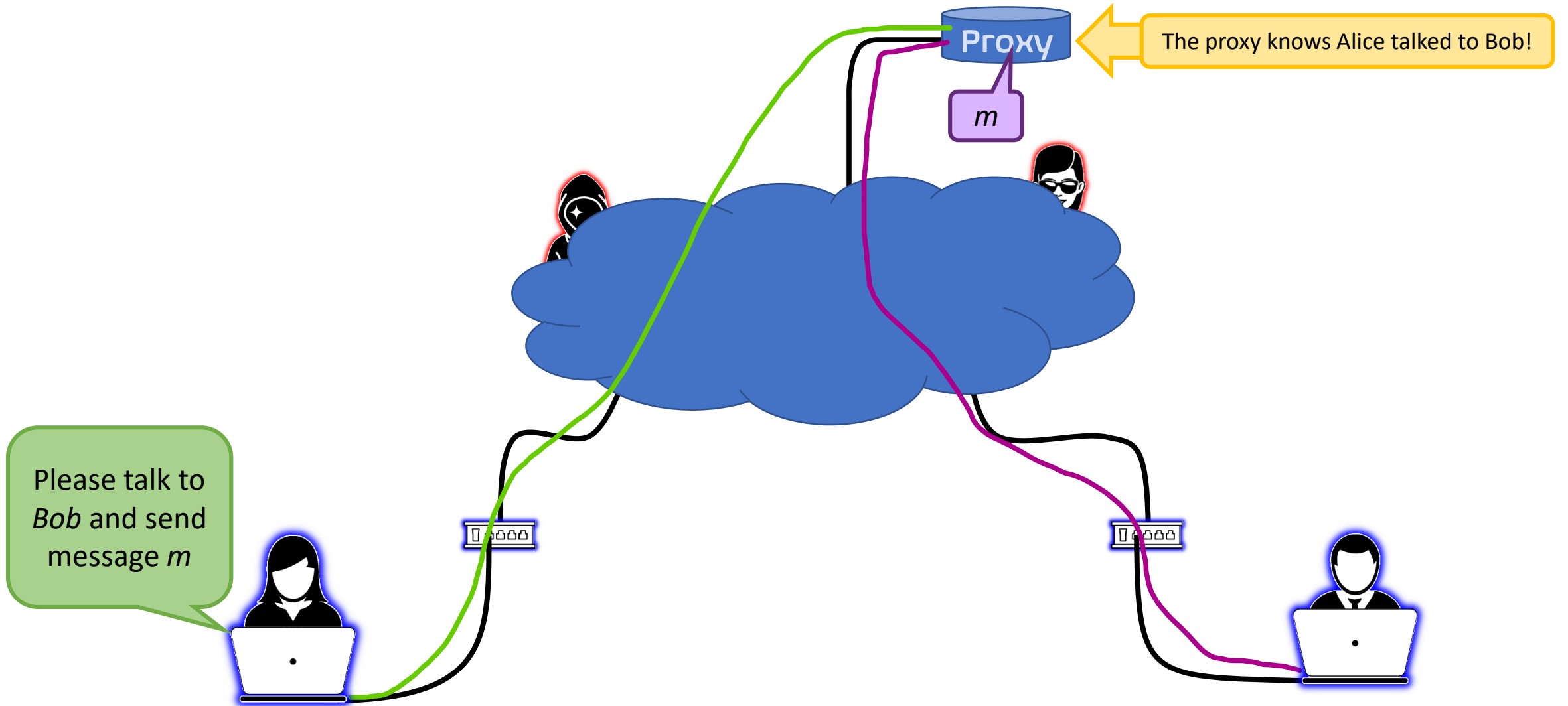
# Metadata?

- Data *about other data*



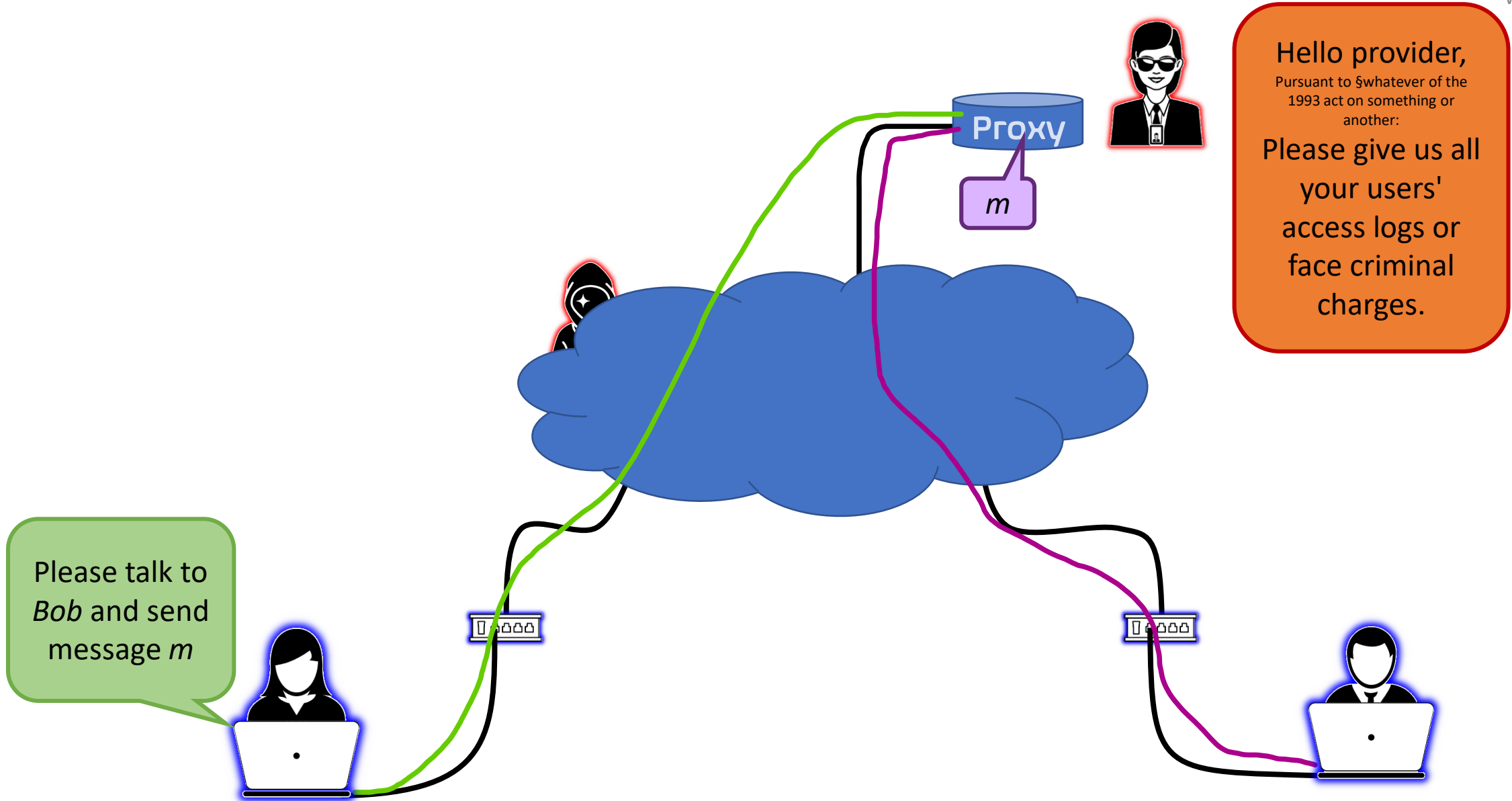
# Why does it matter?

- Undercover journalist reports home
  - Company R&D division suddenly starts talking to patent lawyers
  - Teenager starts browsing Planned Parenthood's home page
  - Iranian teenager researches queer identity
- 
- Often disclosing *association* is already dangerous!



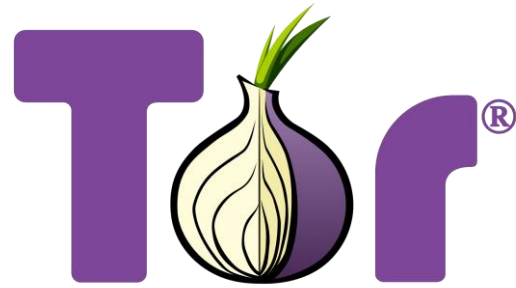
# Considerations on Trust

- The VPN server knows all of Alice's browsing history!
- But: surely, they won't tell anyone!



# Considerations on Trust

- The proxy server knows all of Alice's browsing history!
- If you *have* the data, you can ...
  - ... be subpoenaed for it
  - ... leak it through (unintentional) logging
  - ... lose it if your servers are compromised
- Keeping data 100% safe is impossible!
- Better solution: don't have the data in the first place!



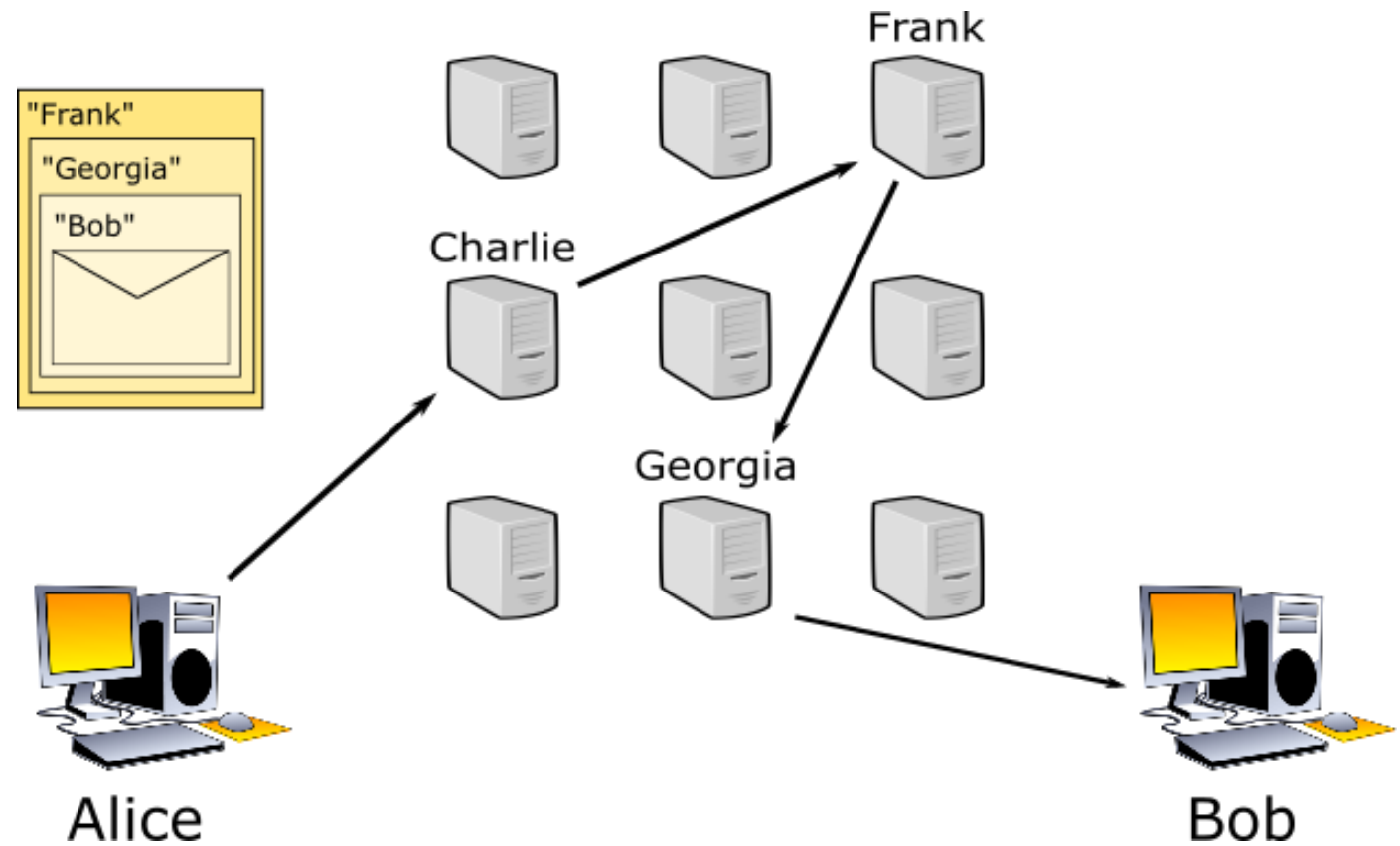
TorProject.org

- "Onion routing" concept
- Each hop only sees neighbors
- Nobody sees the entire route



couldn't compromise it as of 2013

- Thanks, Mr. Snowden!



# Privacy & Metadata – Recap

- Don't store what you don't need!
  - You can't lose data you don't have
- You might have *metadata* you hadn't considered before
  - Minimize it where possible
  
- Trust is good, protocol-level security is better
  - Want to know more about metadata avoidance?
  - Attend MSc-level information security courses!



# Authentication Factors



## GitLab Community Edition

**Username or email**

**Password**

Remember me

[Forgot your password?](#)

IAIK: Teaching related repository management.

By signing in you accept the [Terms of Use](#) and acknowledge the [Privacy Policy and Cookie Policy](#).

# Password Authentication – Issues

Leaked passwords never expire

Server can impersonate user

Phishing

Storing passwords securely

Password reuse

(Cryptographic)

# Authentication Factors

Time-Based One-Time Password (TOTP)



- User Settings
- Account
- Profile
- Security
- Notifications
- Integrations
- API Tokens
- SSH Keys
- Two-Factor Authentication
- Account Deletion

### Register Two-Factor Authenticator

Use a one-time password authenticator on your mobile device or computer to enable two-factor authentication (2FA).

We recommend using cloud-based authenticator applications that can restore access if you lose your hardware device. [What are some examples?](#)



```
otpauth://totp/git.teaching.iaik.tugraz.at:git.teaching.iaik.tugraz.at?secret=40WR2BWMIZFMG7WYHMIFTNQAKLHV43N6&issuer=git.teaching.iaik.tugraz.at
```

**Can't scan the code?**

To add the entry manually, provide the following details to the application on your phone.

Account:  
git.teaching.iaik.tugraz.at@iaik.tugraz.at

Key: 40WR 2BWM IZFM G7WY HMIF TNQA KLHV 43N6

Time based: yes

Pin code

Current password

Your current password is required to register a two-factor authenticator app.

Register with two-factor app

Key: 40WR 2BWM IZFM G7WY HMIF TNQA KLHV 43N6

base32 decode

e3 ad 1d 06 cc 46 4a c3 7e d8  
3b 10 59 b6 00 52 cf 5e 6d be

key

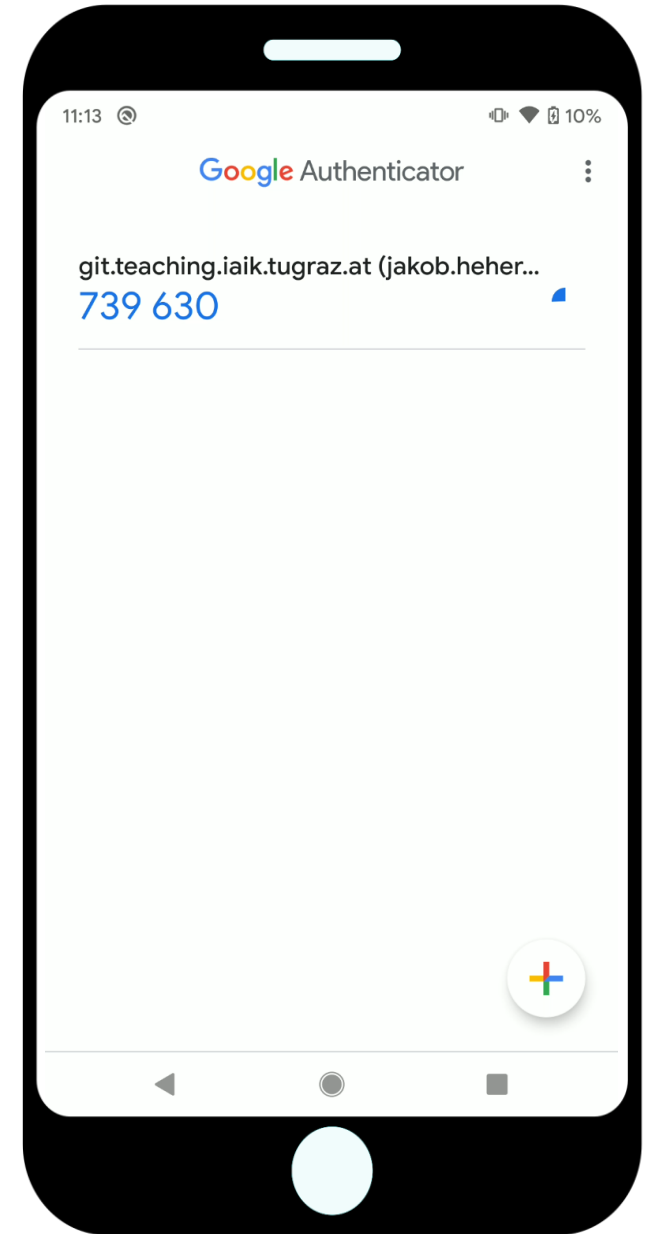
message

HMAC\_SHA1

11:13:22

The RFC 4648 Base 32 alphabet

Value	Symbol	Value	Symbol	Value	Symbol	Value	Symbol
0	A	8	I	16	Q	24	Y
1	B	9	J	17	R	25	Z
2	C	10	K	18	S	26	2
3	D	11	L	19	T	27	3
4	E	12	M	20	U	28	4
5	F	13	N	21	V	29	5
6	G	14	O	22	W	30	6
7	H	15	P	23	X	31	7



Key: 40WR 2BWM IZFM G7WY HMIF TNQA KLHV 43N6

base32 decode

e3 ad 1d 06 cc 46 4a c3 7e d8  
3b 10 59 b6 00 52 cf 5e 6d be

key

HMAC\_SHA1

message

digest

33 33 ac ec 5e d8 27 15 ca ee  
7d f0 27 41 9a 4f 80 2c b5 a6

extract 4 bytes

27 15 ca ee

treat as big-endian integer

655739630

modulo  $10^6$

739630

11:13:22

Tuesday, 8 November 2022

Unix timestamp

1667902402

divide by 30s

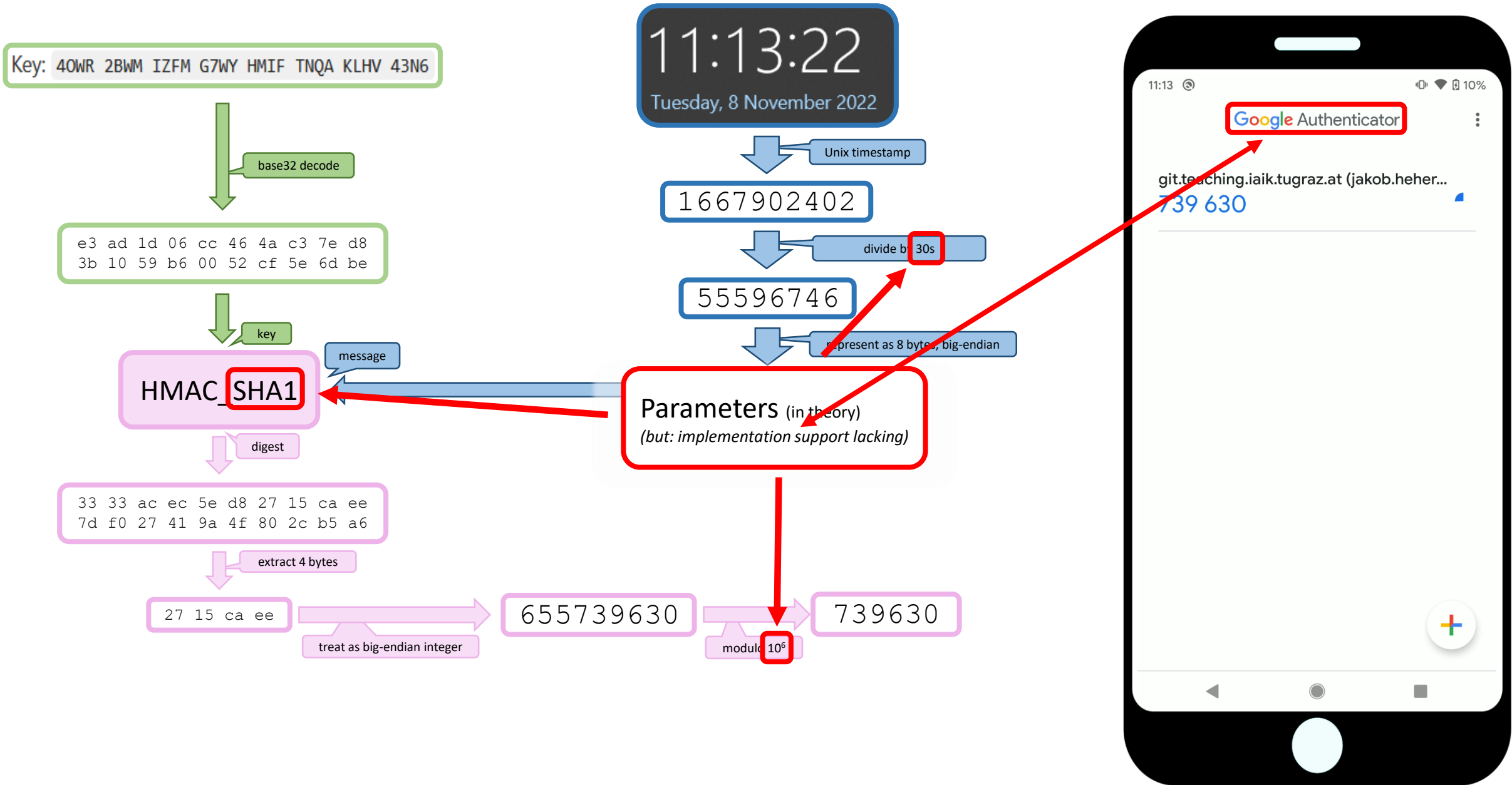
55596746

represent as 8 bytes, big-endian

00 00 00 00 03 50 56 ca

Least significant half-byte = 0x6  
⇒ Take 4 bytes starting at index 0x6







Key: 40WR 2BWM IZFM G7WY HMIF TNQA KLHV 43N6

Device lost?

(Just backup the key!)

e3 ad 1d 06 cc 46 4a c3 7e d8  
3b 10 59 b6 00 52 cf 5e 6d be

HMAC\_SHA1

33 33 ac ec 5e d8 27 15 ca ee  
7d f0 27 41 9a 4f 80 2c b5 a6

27 15 ca ee

655739630

739630

11:13:22

Tuesday, 8 November 2022

1667902402

55596746

00 00 00 00 03 50 56 ca



Key: 40WR 2BWM IZFM G7WY HMIF TNQA KLHV 43N6

base32 decode

e3 ad 1d 06 cc 46 4a c3 7e d8  
3b 10 59 b6 00 52 cf 5e 6d be

key

HMAC\_SHA1

message

digest

33 33 ac ec 5e d8 27 15 ca ee  
7d f0 27 41 9a 4f 80 2c b5 a6

extract 4 bytes

27 15 ca ee

treat as big-endian integer

655739630

modulo  $10^6$

739630

11:13:22

Tuesday, 8 November 2022

Clock Synchronization

(common: try n intervals before/after)

divide by 30s

55596746

represent as 8 bytes, big-endian

00 00 00 00 03 50 56 ca



Key: 40WR 2BWM IZFM G7WY HMIF TNQA KLHV 43N6

base32 decode

e3 ad 1d 06 cc 46 4a c3 7e d8  
3b 10 59 b6 00 52 cf 5e 6d be

key

HMAC\_SHA1

message

digest

33 33 ac ec 5e d8 27 15 ca ee  
7d f0 27 41 9a 4f 80 2c b5 a6

extract 4 bytes

27 15 ca ee

treat as big-endian integer

655739630

modulo

739630

Only 6 characters  
(Lockout/delay for failed attempts)

11:13:22

Tuesday, 8 November 2022

Unix timestamp

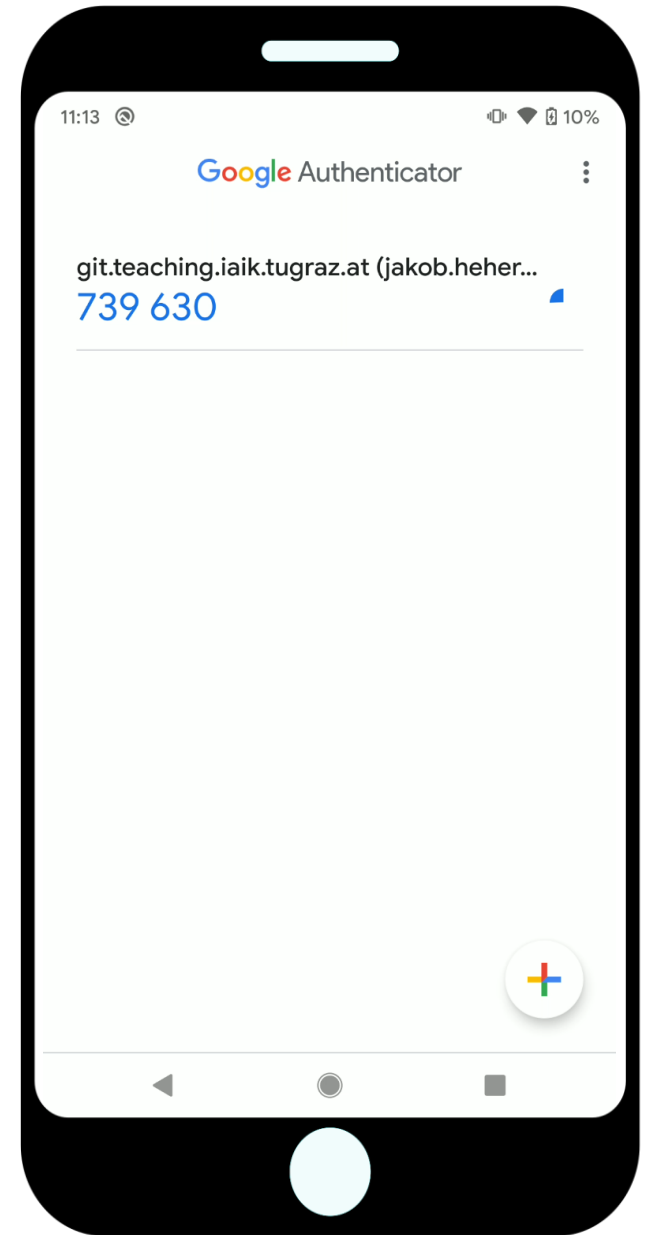
1667902402

divide by 30s

55596746

represent as 8 bytes, big-endian

00 00 00 00 03 50 56 ca



# Time-Based One-Time Password (TOTP)

- Shared secret key + current timestamp  $\Rightarrow$  six-digit passcode
- ✓ Random secret
  - Users cannot reuse passcode between websites
- ✓ Passcode changes every 30 seconds
  - Phished credentials quickly become stale

# Time-Based One-Time Password (TOTP)

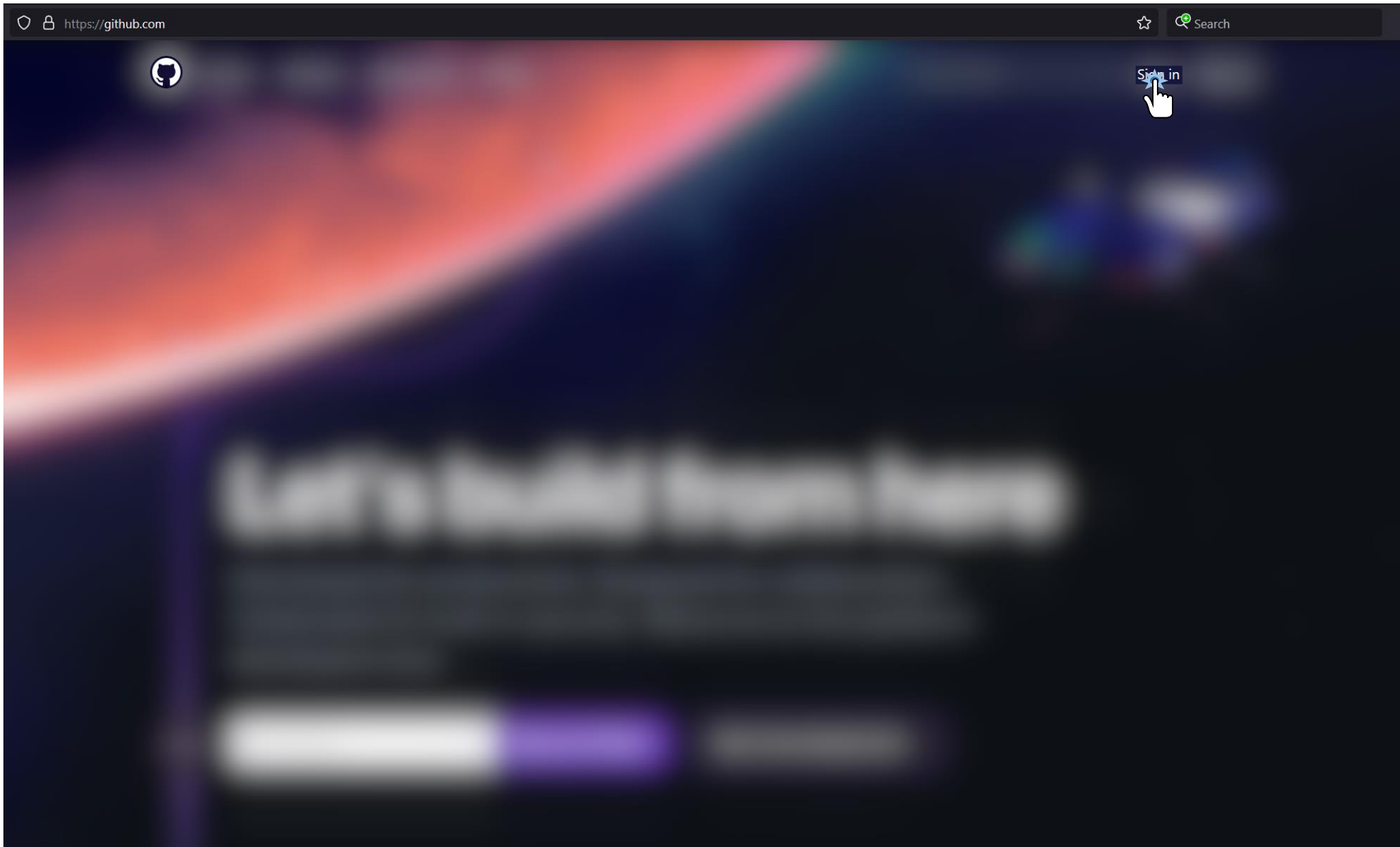
- Shared secret key + current timestamp  $\Rightarrow$  six-digit passcode
- × **Server can still impersonate user**
  - Authentication is based on a symmetric, shared secret
- × **Secure storage is still paramount**
  - ... and more difficult, since you can't hash a secret key
- × **Real-time phishing still works**

(Cryptographic)

# Authentication Factors

Web Authentication











## Two-factor authentication



### Security key

When you are ready to authenticate, press the button below.

Use security key

- Use this method for future logins  
Future logins on this device will prompt you to use a security key by default.

Unable to verify with your security key?

- [Enter two-factor authentication code](#)
- [Use a recovery code or request a reset](#)

Windows Security



## Making sure it's you

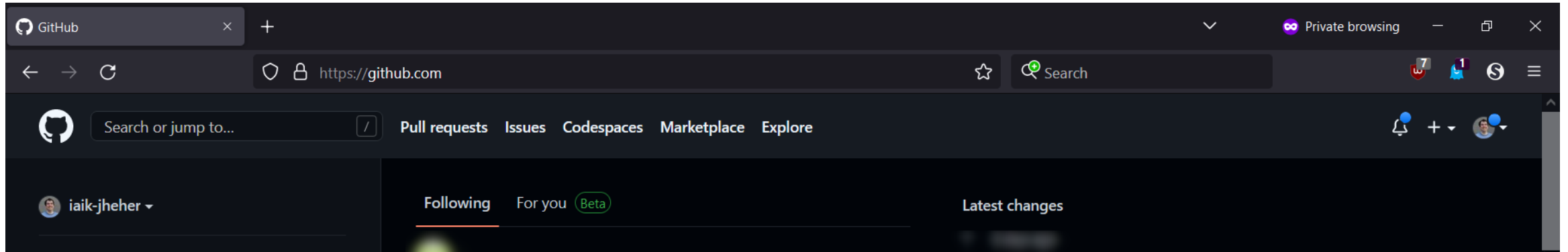
Please sign in to github.com.

This request comes from Firefox, published by Mozilla Corporation.

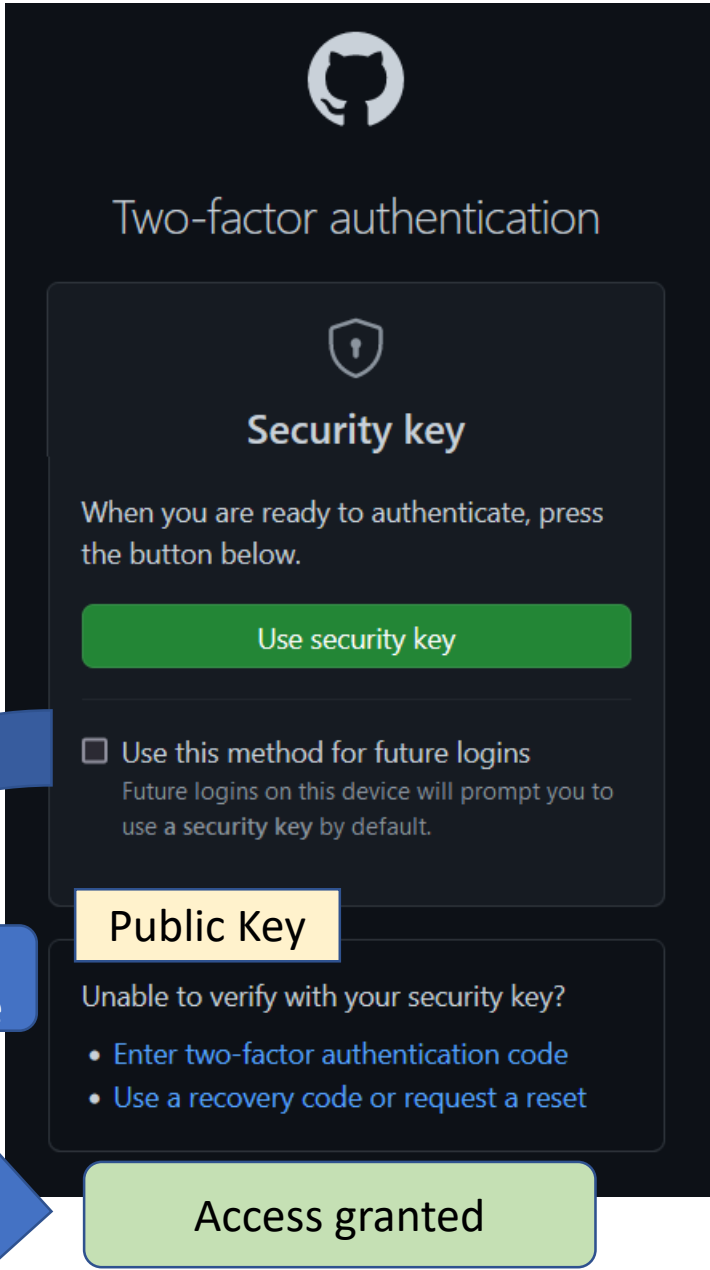


Touch your security key.

Cancel



- OK, what just happened?



Challenge value

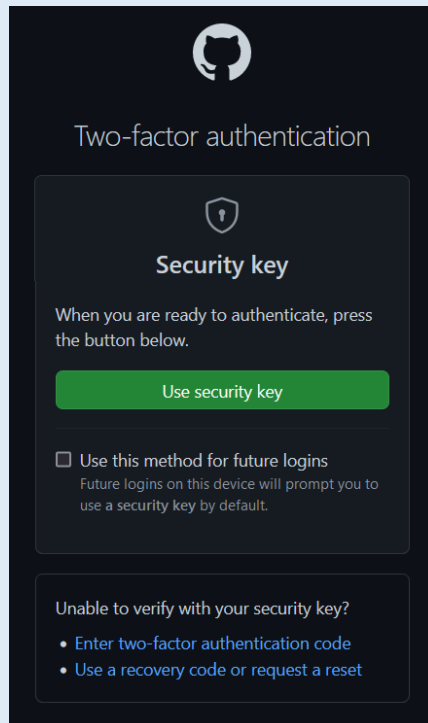


Verify User Presence

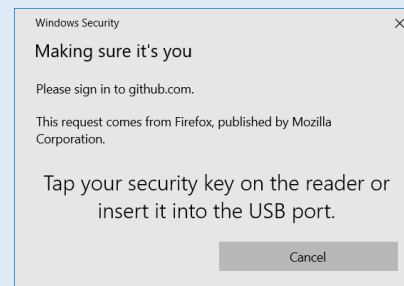
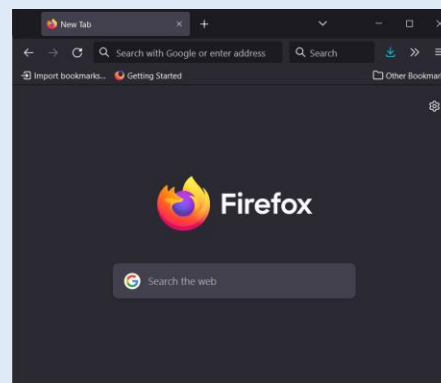
Signature

Verify Signature

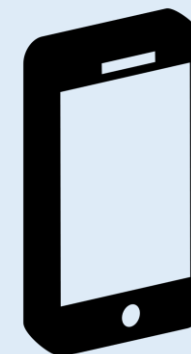
## Relying Party



## Client



## Authenticator



### Relying Party

Two-factor authentication

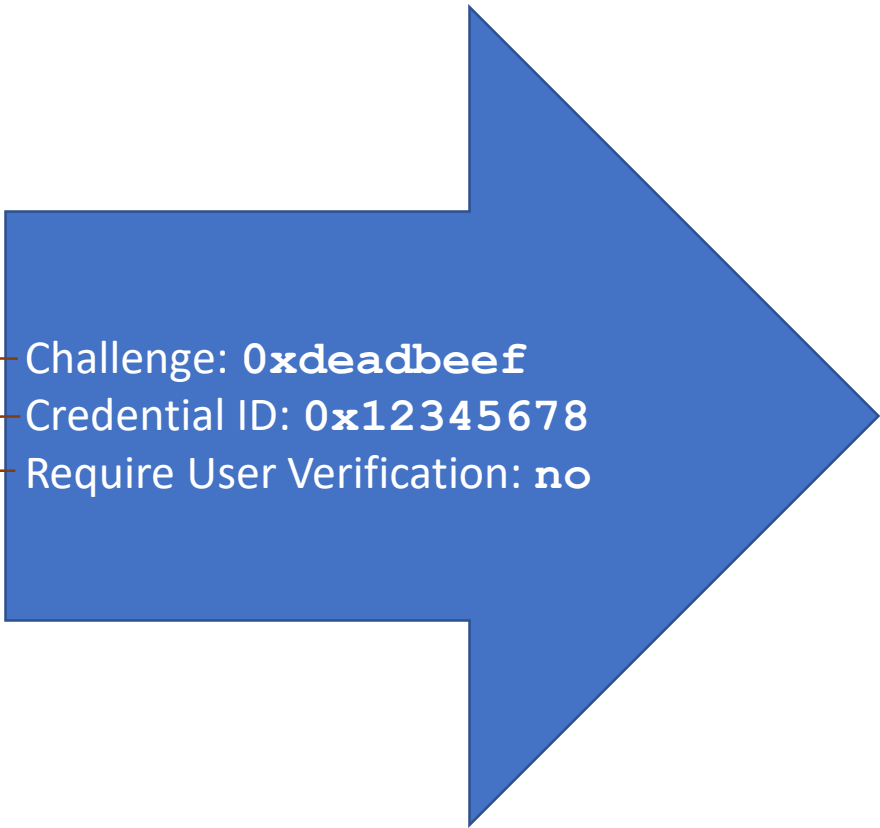
Security key

Use security key

Should the token check a 2<sup>nd</sup> factor?

Unable to verify with your security key?

- Enter two-factor authentication code
- Use a recovery code or request a reset



### Client

J  
a  
v  
a  
S  
c  
r  
i  
p  
t

Windows Security

Making sure it's you

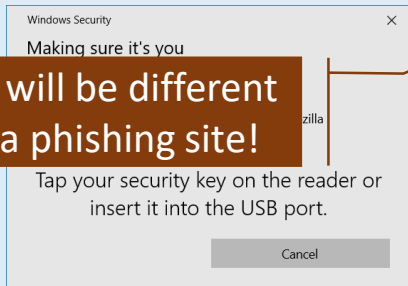
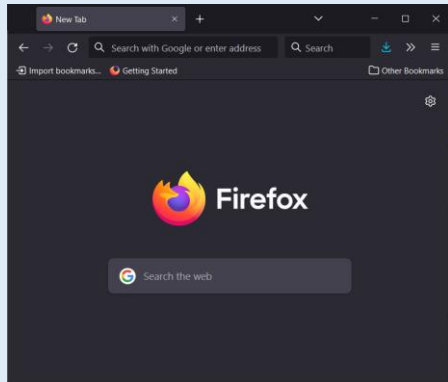
Please sign in to github.com.

This request comes from Firefox, published by Mozilla Corporation.

Tap your security key on the reader or insert it into the USB port.

Cancel

## Client

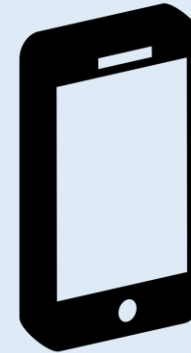


This will be different  
on a phishing site!

Challenge: `0xdeadbeef`  
Credential ID: `0x12345678`  
Require User Verification: `no`

Type: `webauthn.get`  
Origin: `https://github.com`

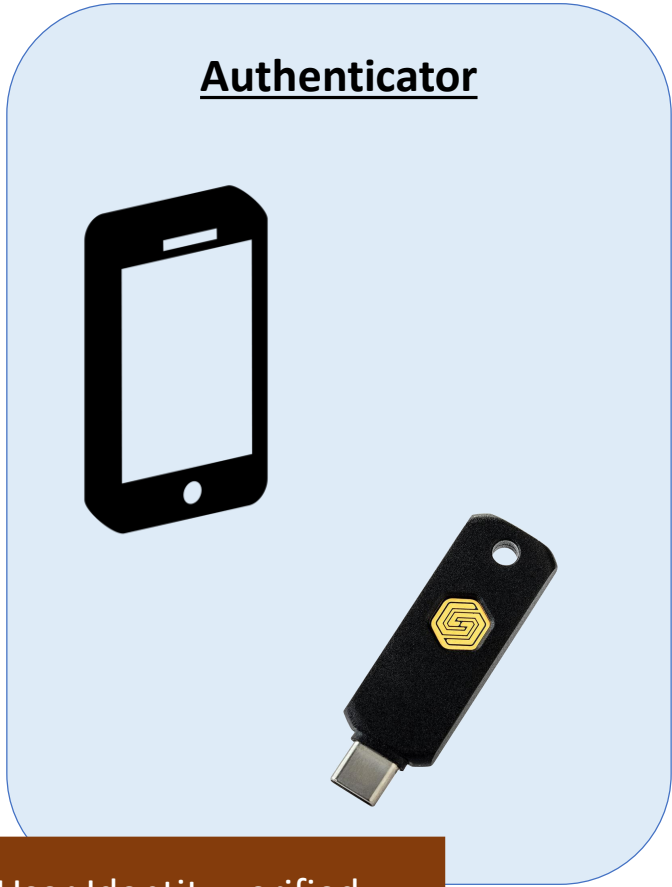
## Authenticator





Challenge: 0xdeadbeef  
Credential ID: 0x12345678  
Require User Verification: no

Type: webauthn.get  
Origin: https://github.com



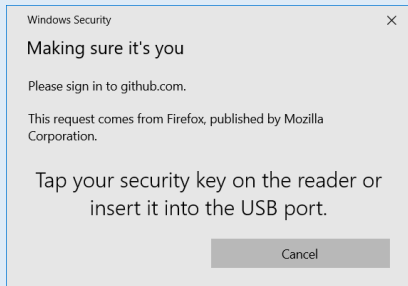
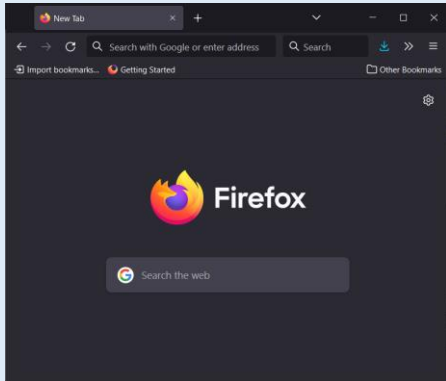
- Find Credential by ID
- Compare Origin
- Verify User Identity  
(if requested)
- Verify User Presence
- Sign Assertion  
(using Private Key)

Signature & Flags

e.g. User Identity verified

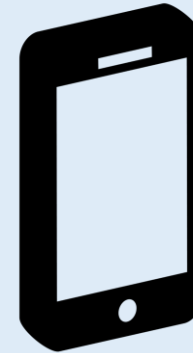
Over all data transmitted

## Client

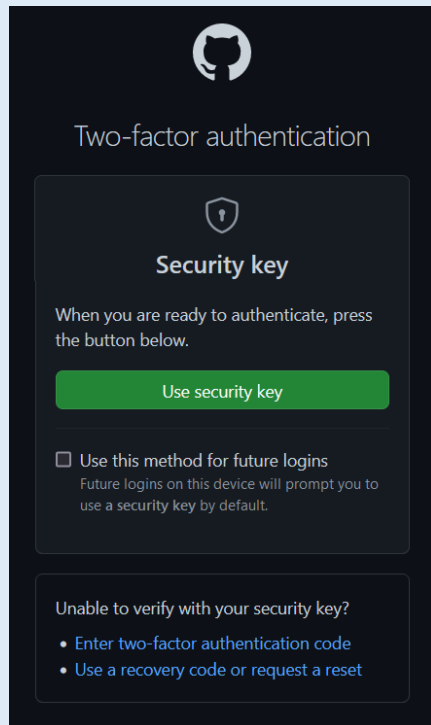


Signature & Flags

## Authenticator



## Relying Party

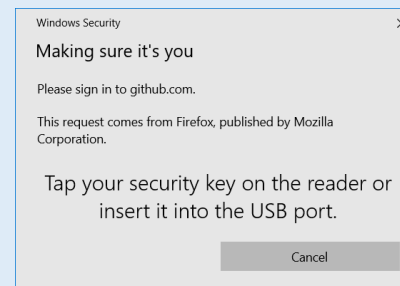


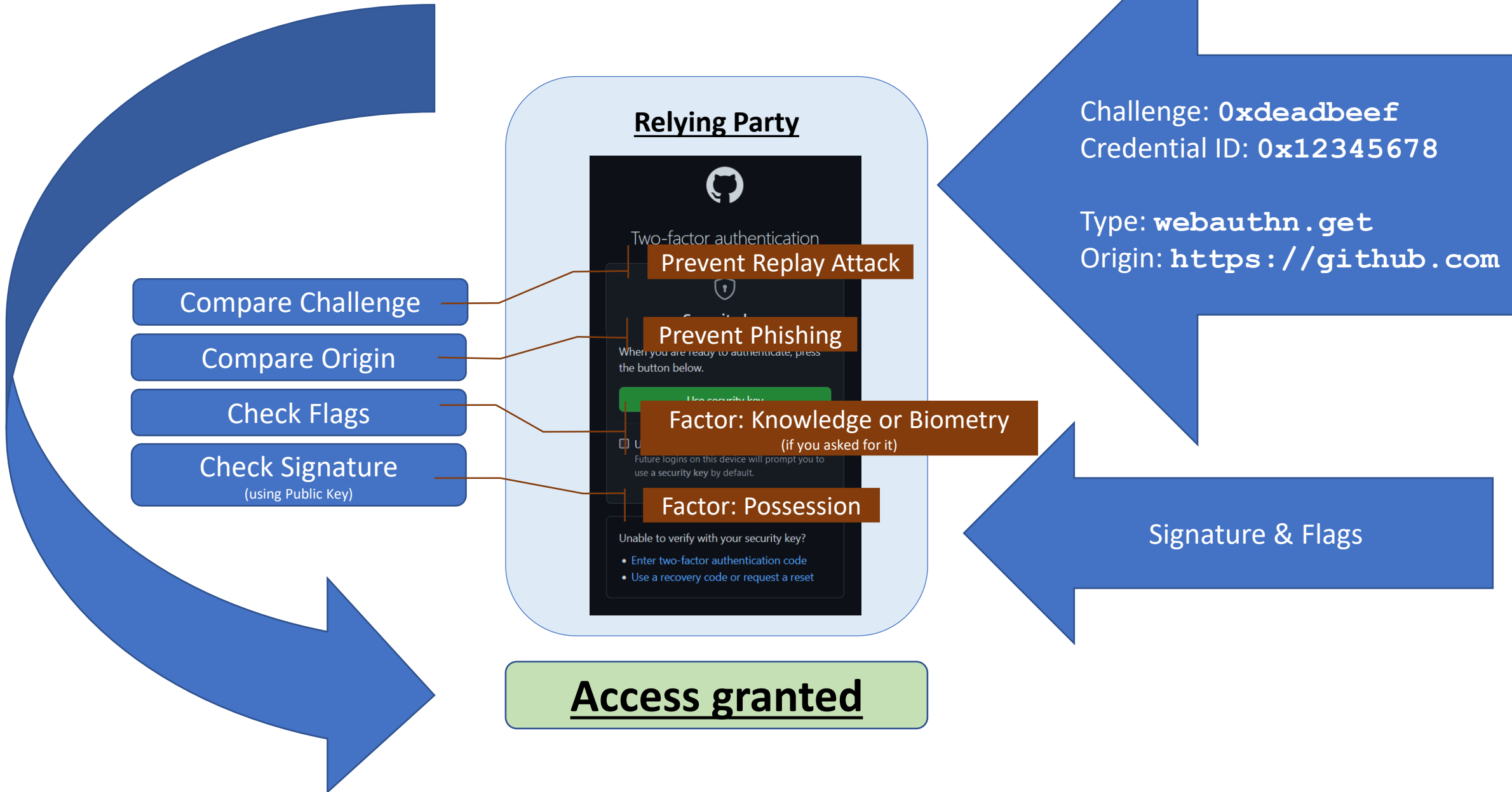
Challenge: `0xdeadbeef`  
Credential ID: `0x12345678`

Type: `webauthn.get`  
Origin: `https://github.com`

Signature & Flags

## Client





# Web Authentication (WebAuthn)

- Public key cryptography using hardware tokens
- ✓ No secure server storage necessary
  - Public keys are not sensitive information
- ✓ Phishing impossible
  - The browser embeds the current origin into the signed data

# Web Authentication (WebAuthn)

- Public key cryptography using hardware tokens
- × Users might lose hardware tokens or devices
  - Your system is only as secure as the recovery factor...

(Cryptographic)

# Authentication Factors

Web Authentication – Passkeys

# Web Authentication (WebAuthn)

- Public key cryptography using hardware tokens

× Users might lose hardware tokens or devices

- Your system is only as secure as the recovery factor...

What if we don't tie each credential to a single device?



# WebAuthn Passkeys

- Public key cryptography with automated key synchronization
- ✓ No secure server storage necessary
  - Public keys are not sensitive information
- ✓ Phishing impossible
  - The browser embeds the current origin into the signed data
- ✓ Credentials survive device failure or loss
  - Synchronized via “sync providers” (Microsoft, Apple, Google)

# WebAuthn Passkeys

- Public key cryptography with automated key synchronization
- × Sync providers' implementation is a *huge* point of failure
  - A vulnerability would expose *billions* of single-factor credentials
- × Dependency on sync platforms leads to customer lock-in
  - Switching loses every single credential you use to log in, everywhere
- × Lack of interoperability reinforces existing cross-sector monopolies
  - Want to use a phone OS, made by Google, to log in?
  - Only if you're using a specific browser that's made by Google!

# WebAuthn Passkeys

- Public key cryptography with automated key synchronization
- ✓ Definitely more secure than “standard” password usage
- ? Difficult to compare with password manager usage
- × Less secure than hardware token usage
- × Has some pretty glaring ethical issues