

Secure your app with 2FA

Rob Allen ~ @akrabat ~ April 2017

Your users' passwords will
be leaked

(It might not even be your fault)

Passwords have leaked from

Sony

Facebook

Adobe

Evernote

Kickstarter

HP

Gmail

Ubuntu

Zappos

Twitter

Target

Yahoo!

Home Depot

AOL

last.fm

D&B

JP Morgan

eBay

Blizzard

Steam

TK Maxx

Citigroup

Apple

Formspring

Gap

AT&T

TalkTalk

Ubisoft

Vodafone

WorldPay

SnapChat

Betfair

It will take 14 minutes* to crack
one of your users[†] passwords

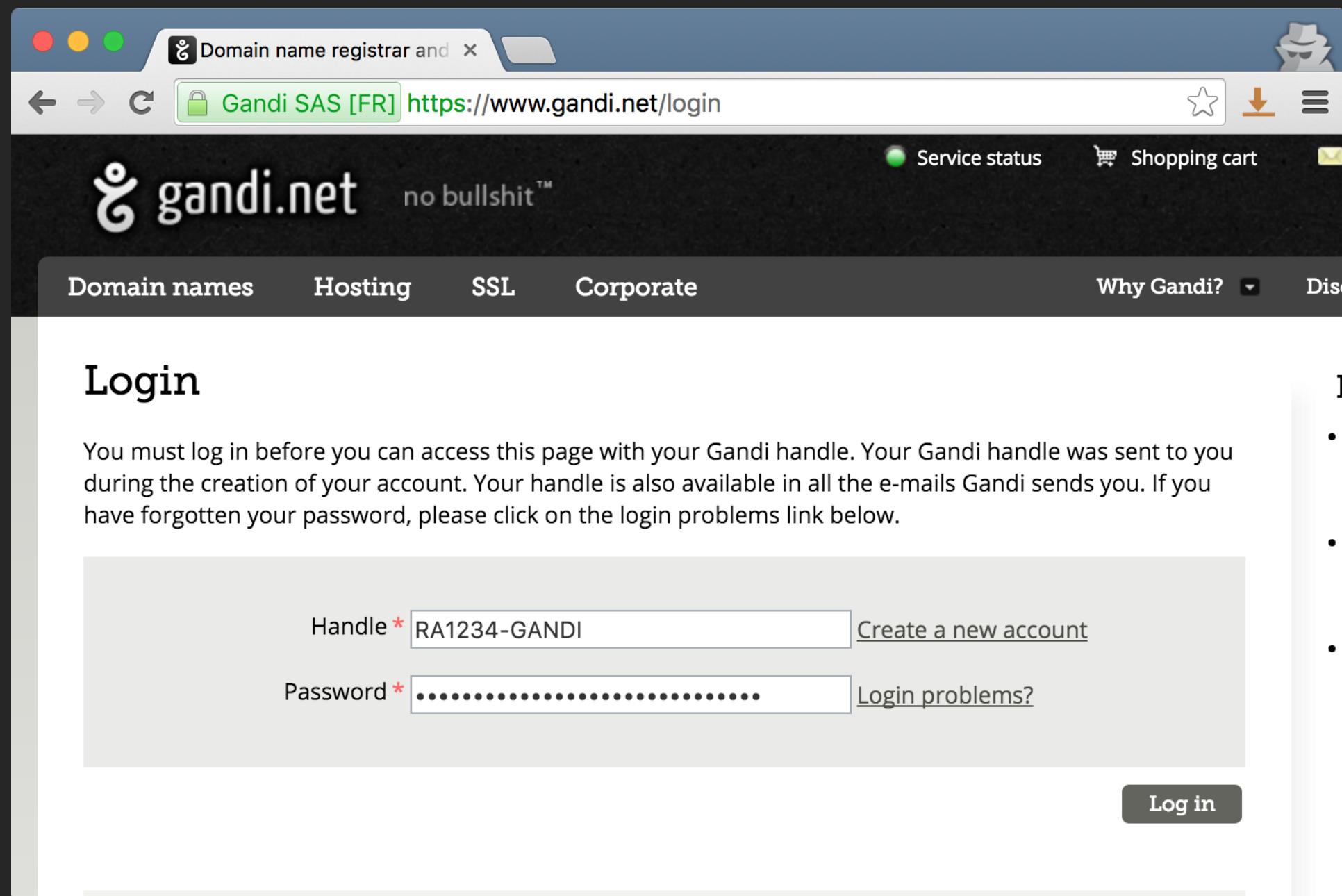
*English word, stored using bcrypt

Two-factor authentication
protects your users

What is 2FA?



Implementation on a website



Implementation on a website

The screenshot shows a web browser window with a blue header bar. The title bar says "Domain name registrar and" and the address bar shows "Gandi SAS [FR] https://www.gandi.net/login". The main content area is titled "Login". It contains instructions: "You must log in before you can access this page with your Gandi handle. Your Gandi handle was sent to you during the creation of your account. Your handle is also available in all the e-mails Gandi sends you. If you have forgotten your password, please click on the login problems link below." Below these instructions are two input fields: "Handle * RA1234-GANDI" and "Password *". To the right of the handle field is a link "Create a new account" and to the right of the password field is a link "Login problems?". A section titled "Two-factor authentication" follows, with the instruction "Please provide your authentication token." and a "Token * 456820" input field. A "Log in" button is located at the bottom right.

Handle * RA1234-GANDI [Create a new account](#)

Password * [Login problems?](#)

Two-factor authentication

Please provide your authentication token.

Token * 456820

Log in

How do we send the code?

Email

- Used by Steam
- Wide adoption (everyone has an email address!)
- Likely failures: delivery problems, blocking, spam etc
- Usually slow!
- Same system as recover password...

SMS

- Used by Twitter & LinkedIn
- Wide adoption
- But, SMS can be delayed & could cost to receive



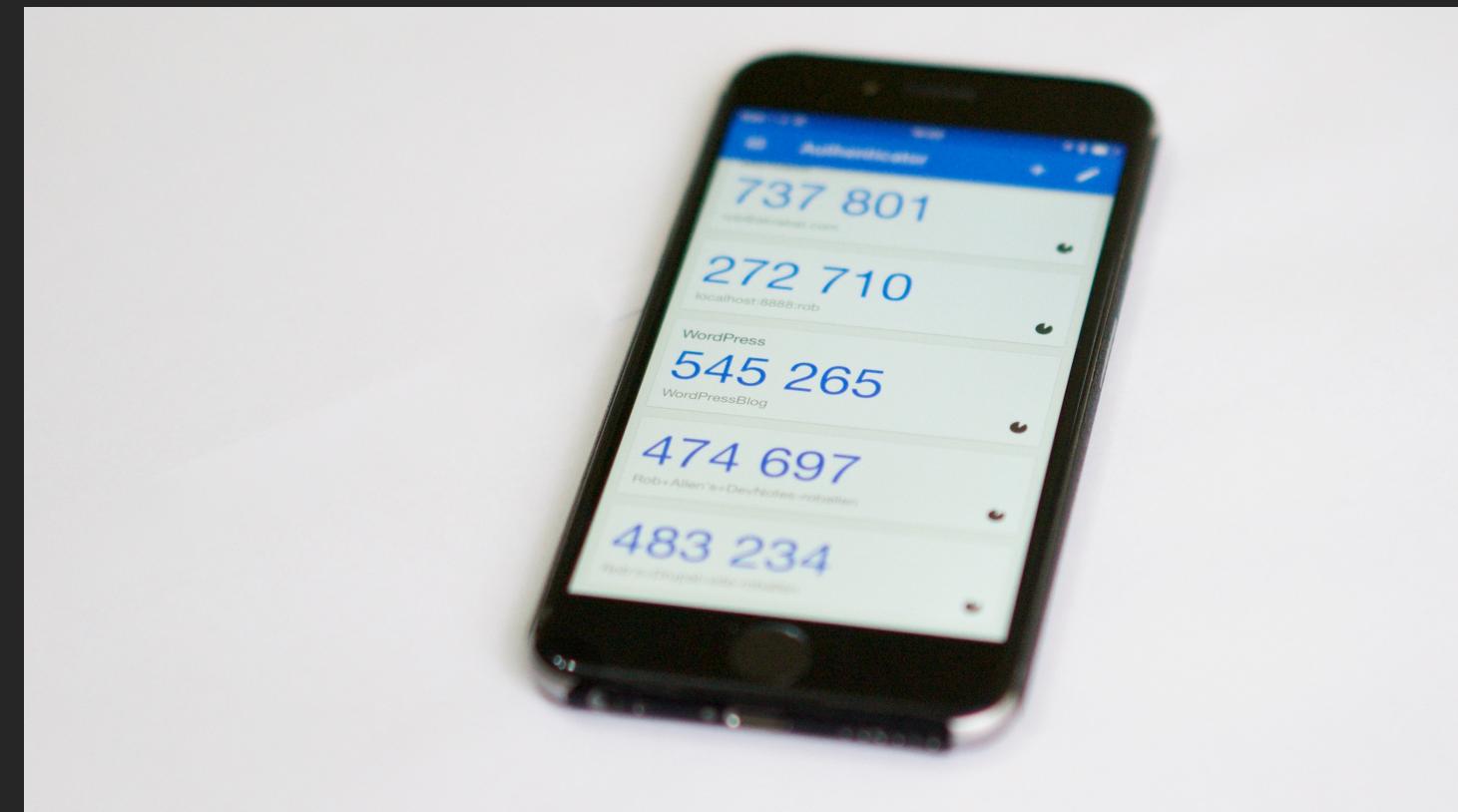
Physical device

- Used by banks, YubiKey, Blizzard, etc
- Small, long battery life
- But: expensive



App

- Easy to use
- No Internet or cellular connection required
- App is free and trusted

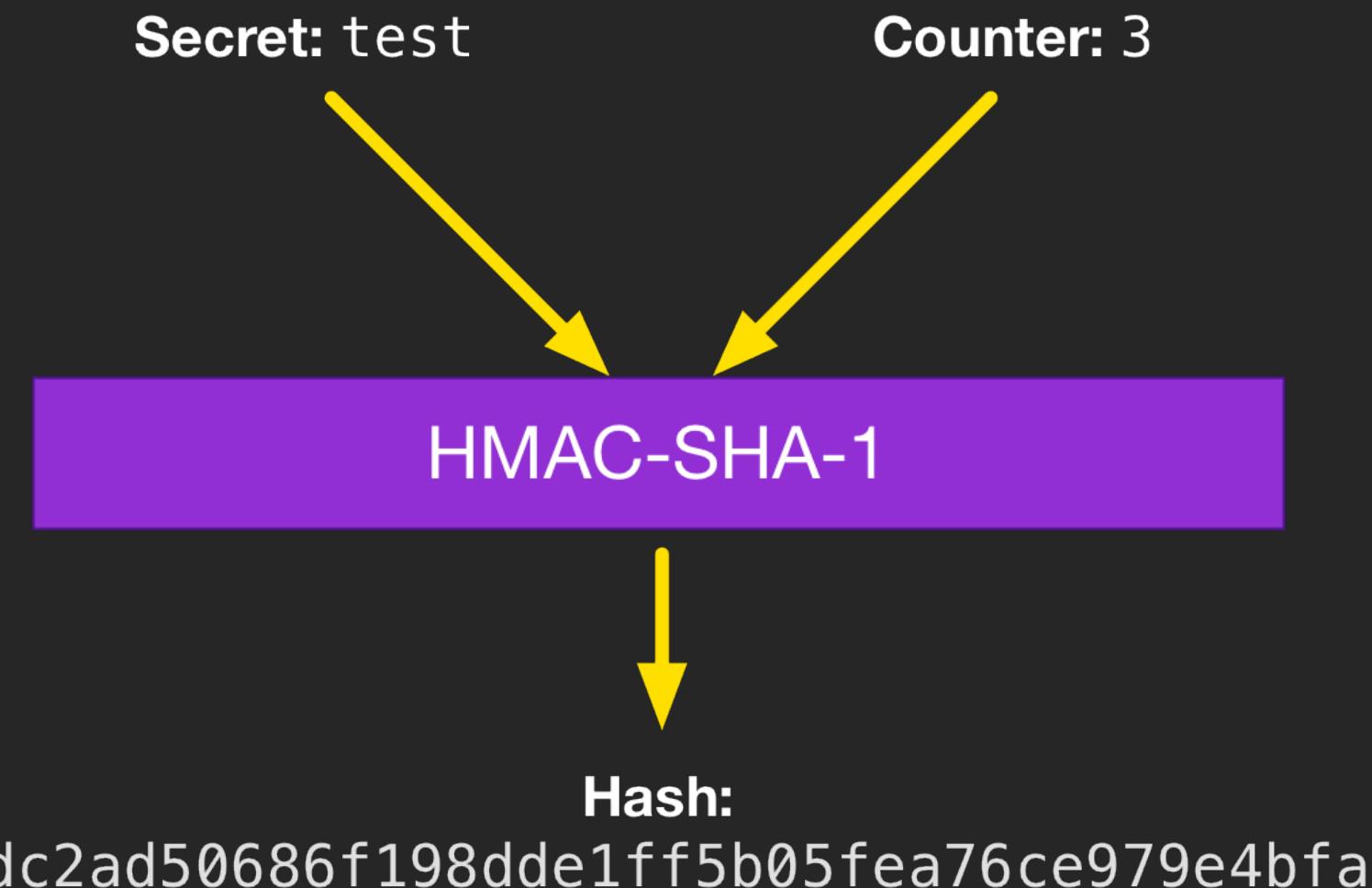


One Time Password algorithms

HOTP

- HMAC-based One-Time Password algorithm
- Computed from shared secret and counter
- New code each time you press the button
- RFC 4226

HOTP algorithm: step 1



HOTP algorithm: step 2

Hash:

8dc2ad50686f198dde1ff5b05fea76ce979e4bfa



Find lower 4 bits of last byte

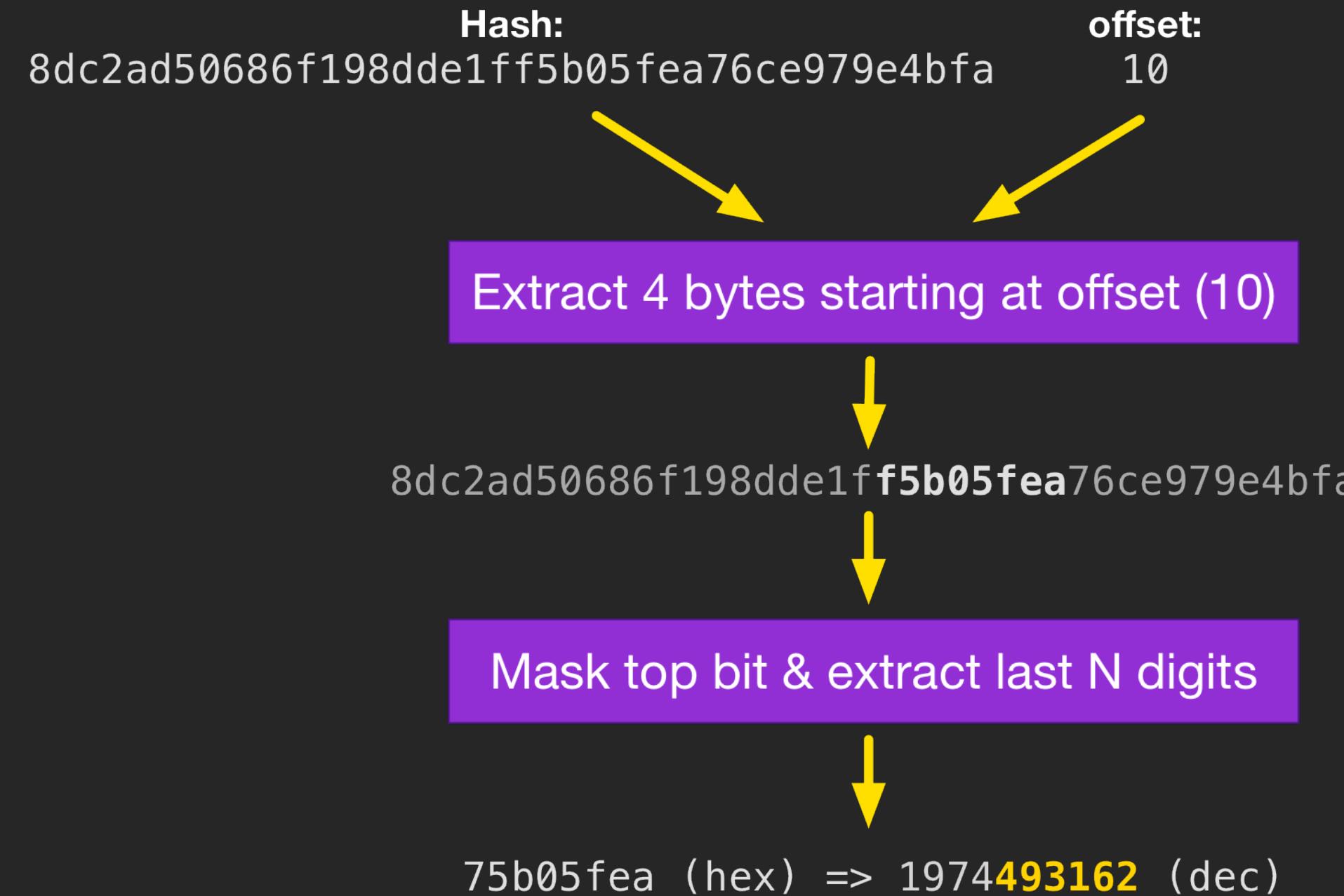


8dc2ad50686f198dde1ff5b05fea76ce979e4bf**a**

offset:

0x0A (hex) => 10 (decimal)

HOTP algorithm: step 3



HOTP in PHP

```
1 function hotp($secret, $counter)
2 {
3     $bin_counter = pack('J*', $counter);
4     $hash = hash_hmac('sha1', $bin_counter, $secret, true);
5
6     $offset = ord($hash[19]) & 0xf;
7
8     $bin_code =
9         ((ord($hash[$offset+0]) & 0x7f) << 24) |
10        ((ord($hash[$offset+1]) & 0xff) << 16) |
11        ((ord($hash[$offset+2]) & 0xff) << 8) |
12        (ord($hash[$offset+3]) & 0xff);
13
14     return $bin_code % pow(10, 6);
15 }
```

Validation process

If the user's code matches, then increment counter by 1

If the user's code does not match, then look-ahead a little

Resync if can't find in look-ahead:

1. Ask the user for two consecutive codes
2. Look ahead further from last known counter until the 2 codes are found
3. Limit look-ahead to minimise attack area. e.g. 400

TOTP

- Time-based One-Time Password algorithm
- Computed from shared secret and current time
- Increases in 30 second intervals
- RFC 6238

TOTP Algorithm

time: 1441048014

period: 30 seconds

Round time down to period boundary

T: 48034933

Secret: test

HOTP algorithm
(repeat for T-1 & T+1)

TOTP in PHP

```
1 function totp($secret)
2 {
3     $counter = floor(time() / 30);
4
5     return hotp($secret, $counter);
6 }
```

Implementing 2FA in your application

Use a library!

```
$composer require sonata-project/google-authenticator
```

Usage:

```
$g = new \Google\Authenticator\GoogleAuthenticator();

// create new secret and QR code
$secret = $g->generateSecret();
$qrcode = $g->getURL('rob', 'akrabat.com', $secret);

// validation of code
$g->checkCode($secret, $_POST['code']);
```

Things to do

1. Registration of Authenticator
2. Check 2FA TOTP code during login

Set up 2FA

User enables 2FA on their account

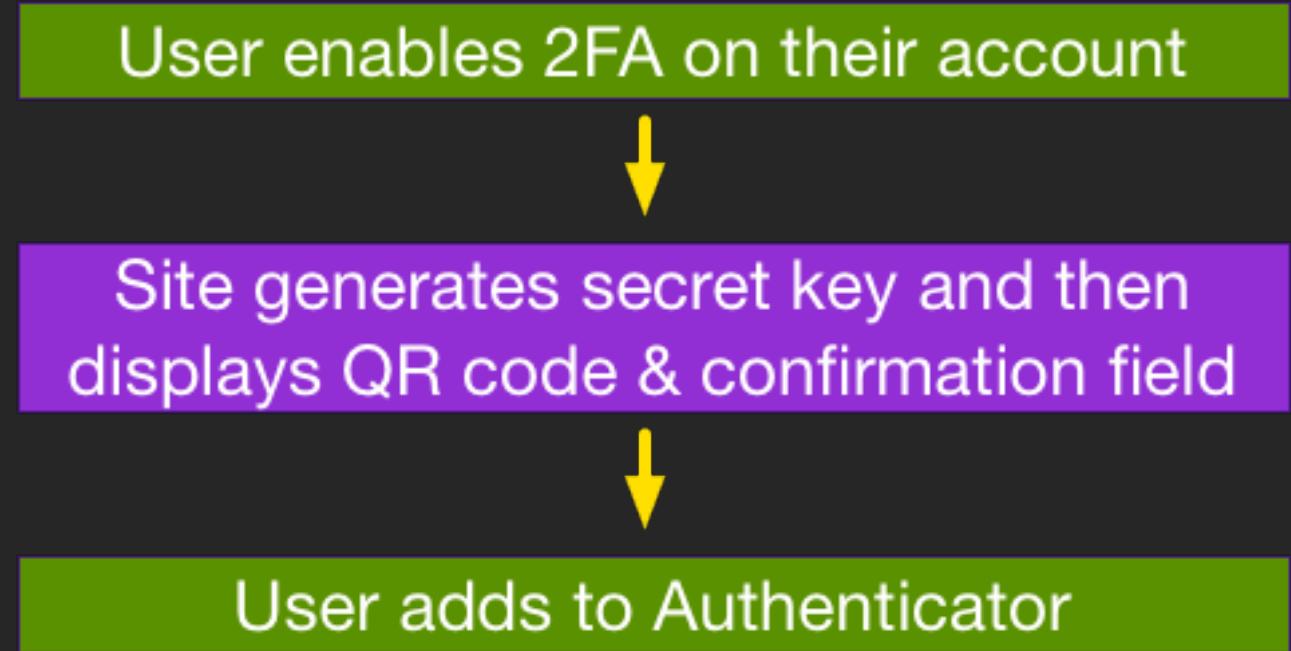
Set up 2FA

User enables 2FA on their account

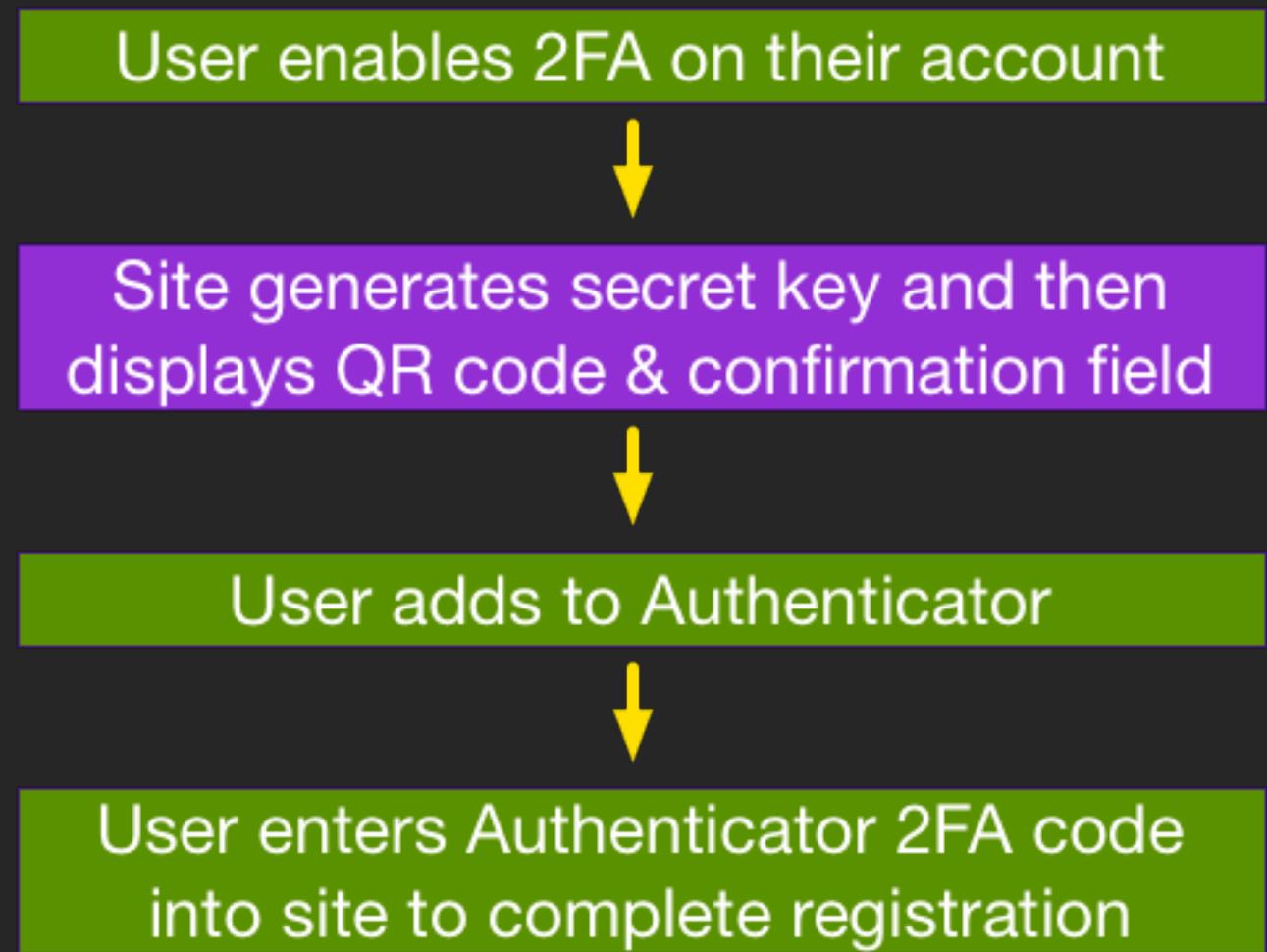


Site generates secret key and then
displays QR code & confirmation field

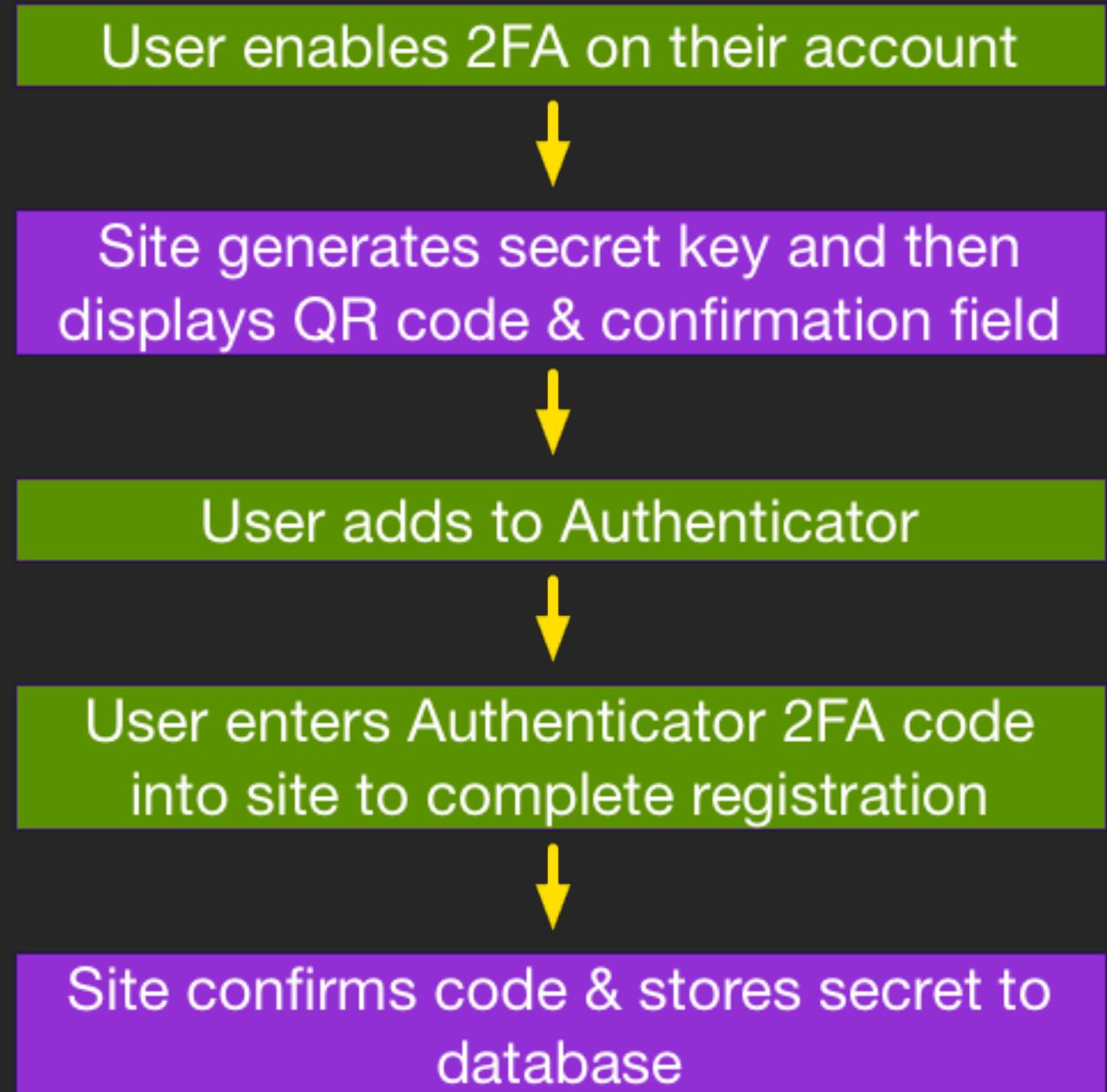
Set up 2FA



Set up 2FA



Set up 2FA



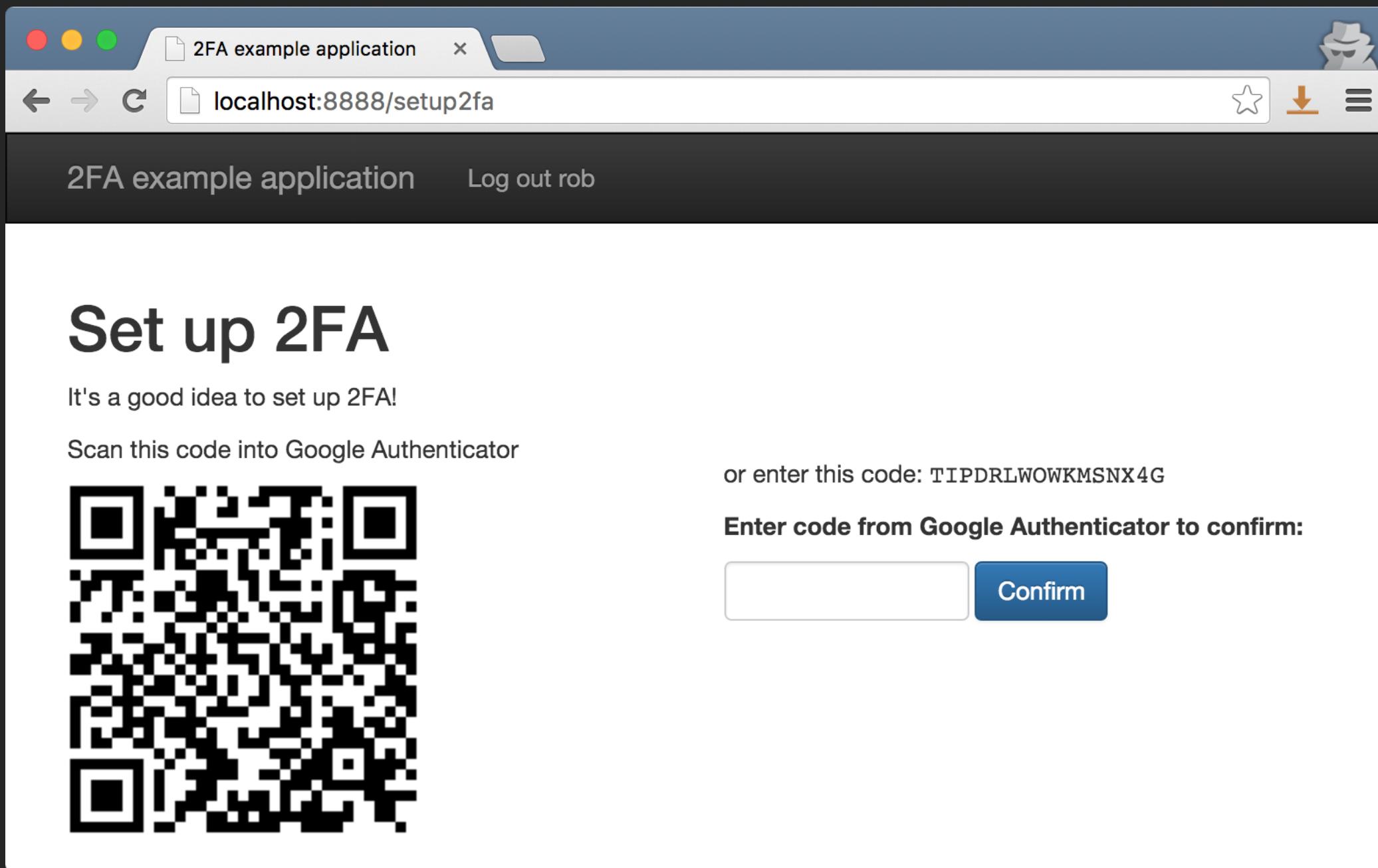
Set up 2FA: code

```
1 $app->get('/setup2fa', function () use ($app) {
2     $user = $_SESSION['user'];
3
4     $g = new \Google\Authenticator\GoogleAuthenticator();
5     $secret = $g->generateSecret();
6     $qrCodeUrl = $g->getUrl($user->getUsername(),
7         '2fa.dev', $secret);
8
9     $app->flash('secret', $secret);
10    $app->render('setup2fa.twig', [
11        'user'      => $_SESSION['user'],
12        'secret'    => $secret,
13        'qrCodeUrl' => $qrCodeUrl,
14    ]);
15});
```

Set up 2FA: template

```
1 <h1>Set up 2FA</h1>
2
3 <p>Scan this code into Google Authenticator:</p>
4 
5 <p>or enter this code: <tt>{{ secret }}</tt></p>
6
7 <p>Enter code from Google Authenticator to confirm:</p>
8 <form method="POST" action="/setup2fa">
9   <div class="form-group">
10     <input name="code" maxlength="6">
11     <button type="submit">Confirm</button>
12   </div>
13 </form>
```

Set up 2FA



A screenshot of a web browser window titled "2FA example application". The address bar shows "localhost:8888/setup2fa". The page itself is titled "Set up 2FA" and contains the text "It's a good idea to set up 2FA!". It instructs the user to "Scan this code into Google Authenticator" and provides a QR code. An alternative method is given as "or enter this code: TIPDRLWOWKMSNX4G". Below this, there is a text input field labeled "Enter code from Google Authenticator to confirm:" followed by a blue "Confirm" button.

2FA example application Log out rob

Set up 2FA

It's a good idea to set up 2FA!

Scan this code into Google Authenticator



or enter this code: TIPDRLWOWKMSNX4G

Enter code from Google Authenticator to confirm:

 Confirm

Set up 2FA: process submission

```
1 $app->post('/setup2fa', function () use ($app) {
2     $secret = $app->environment['slim.flash']['secret'];
3     $code = $app->request->post('code');
4
5     $g = new \Google\Authenticator\GoogleAuthenticator();
6     if ($g->checkCode($secret, $code)) {
7         /* successful registration: store secret into user's row in db */
8
9         $app->redirect('/');
10    }
11
12    $app->flash('error', 'Failed to confirm code');
13    $app->redirect('/setup2fa');
14 }));
```

Logging in

After login form, prompt for 2FA code

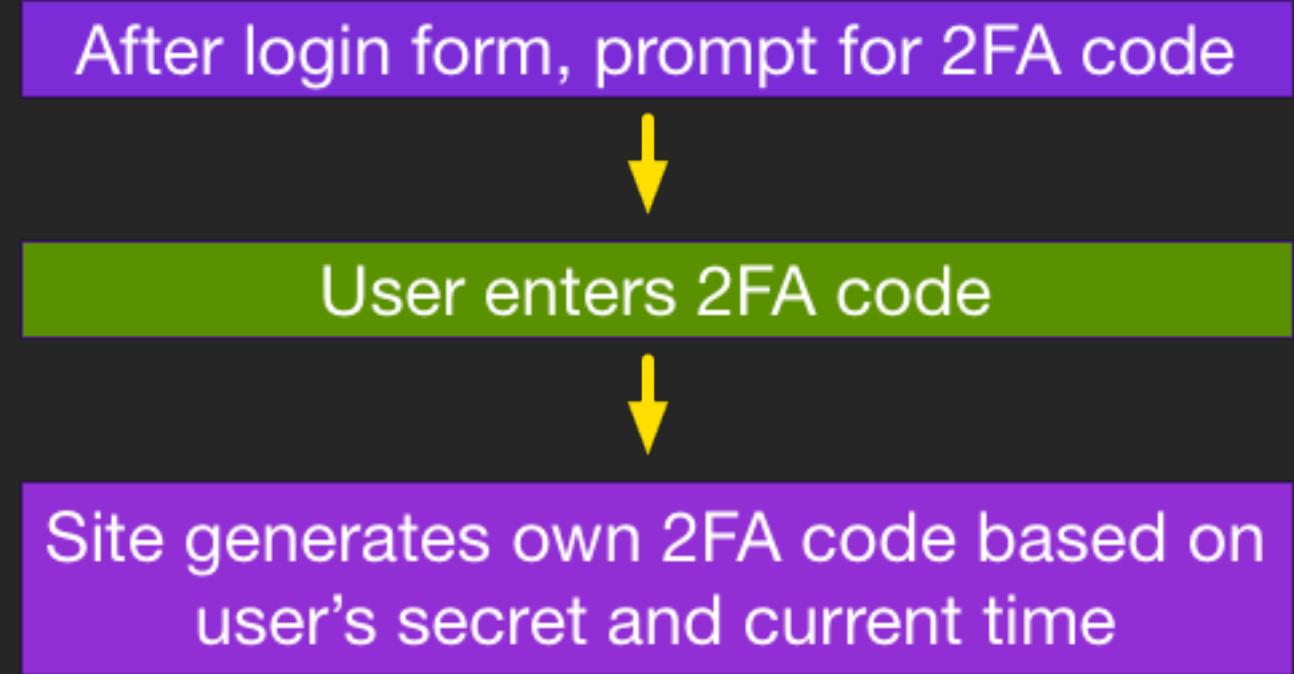
Logging in

After login form, prompt for 2FA code

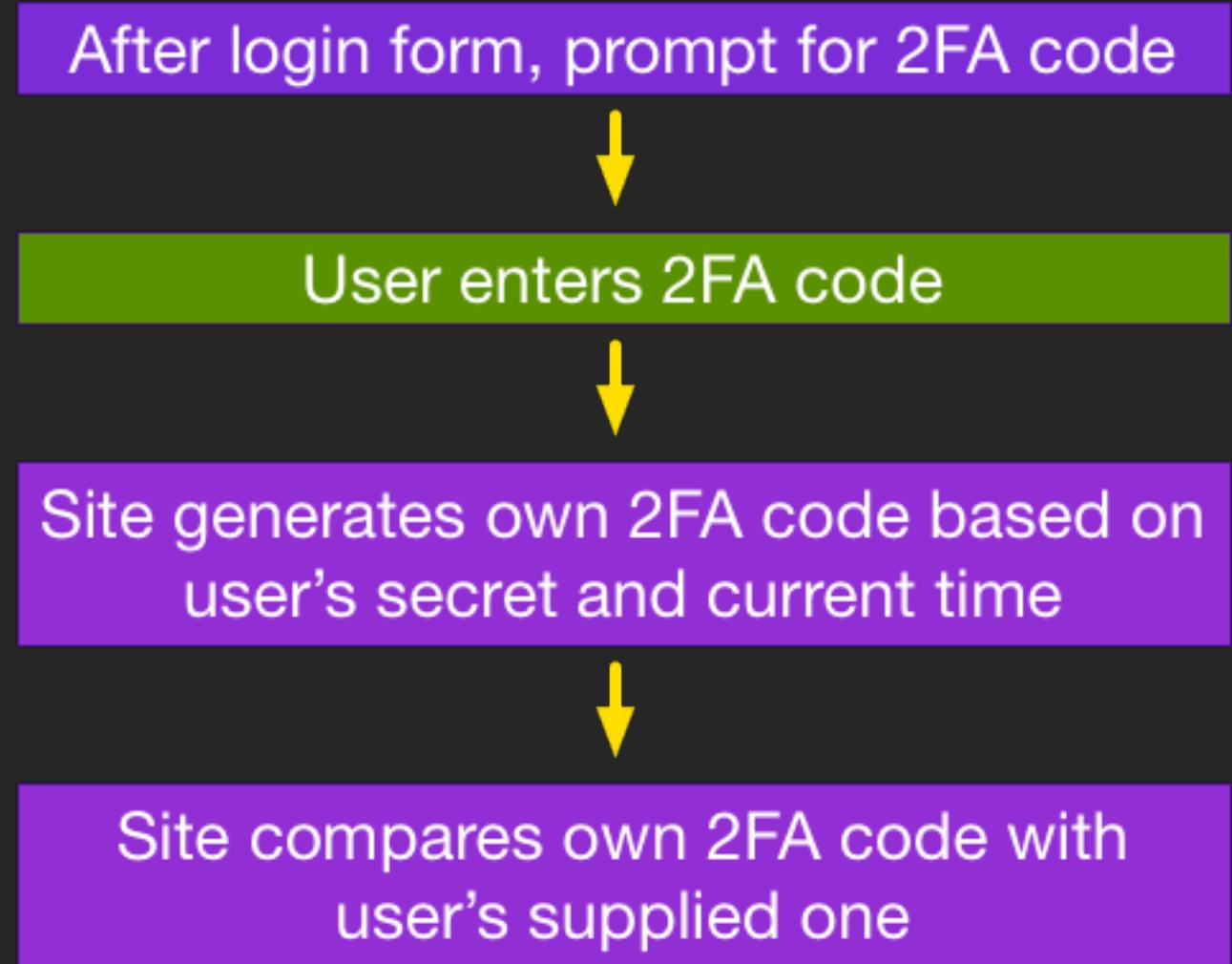


User enters 2FA code

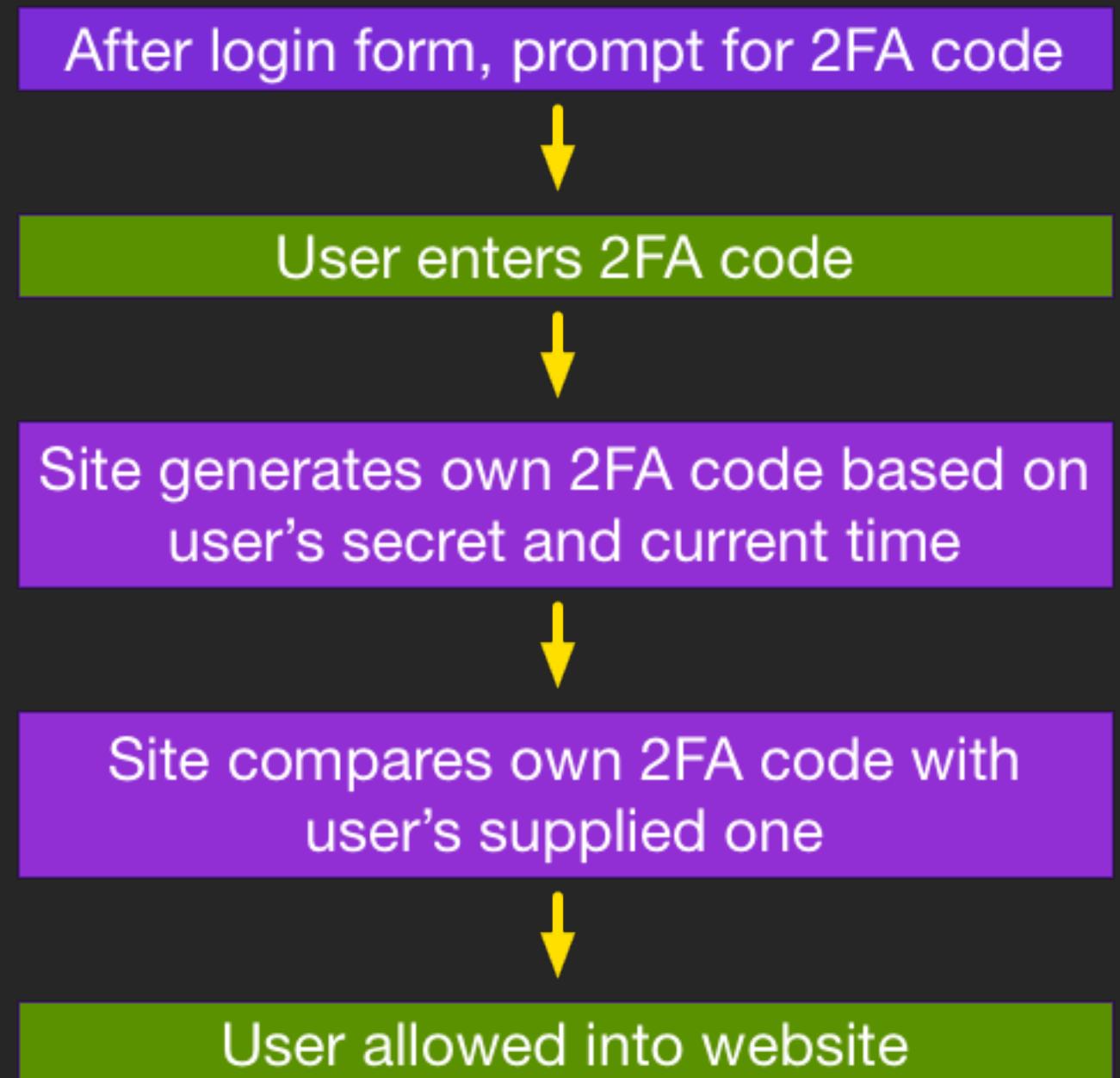
Logging in



Logging in



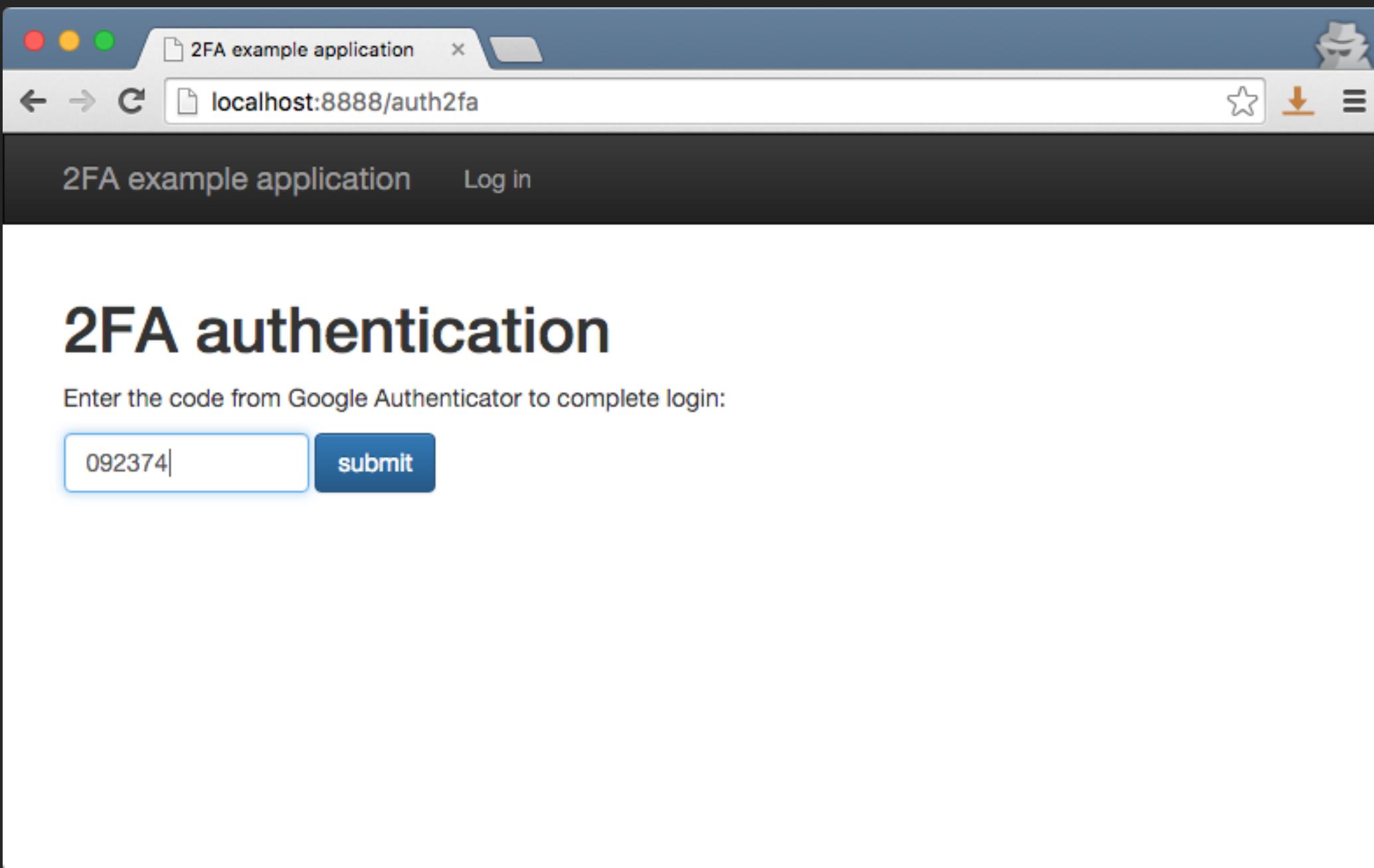
Logging in



Login: 2FA code form

```
1 <h1>2FA authentication</h1>
2
3 <p>Enter the code from Google Authenticator to complete login:</p>
4
5 <form method="POST" action="/auth2fa">
6   <div>
7     <input name="code" maxlength="6">
8     <button type="submit">Submit</button>
9   </div>
10 </form>
```

2FA code



Login: process 2FA code

```
1 $app->post('/auth2fa', function () use ($app) {
2     $user    = $_SESSION['user_in_progress'];
3     $secret  = $user->getSecret();
4     $code    = $app->request->post('code');
5
6     $g = new \Google\Authenticator\GoogleAuthenticator();
7     if ($g->checkCode($secret, $code)) {
8         /* code is valid! */
9         $_SESSION['user'] = $_SESSION['user_in_progress'];
10
11        $app->redirect('/');
12    }
13
14    $app->flash('error', 'Failed to confirm code');
15    $app->redirect('/auth2fa');
16});
```

Round out solution

- Prevent brute force attacks
- Consider adding a "remember this browser" feature
- Need a solution for a lost/new phone

Example project: <https://github.com/akrabat/slim-2fa>

Hardware OTP: YubiKey



Operation

1. Insert YubiKey into USB slot
2. Select input field on form
3. Press button to fill in OTP field
4. Server validates OTP with YubiCloud service

Yubikey's OTP

cdcccccetfjnfigkgkgudlkcbjnfrkhkbelbfvnhcj
cdcccccetfjnfuhtrehjekcciljdbifgfrlccbnjhkf
cdcccccetfjnfljrhettkvijcgddkenbtuiknktejgnvgb



Public id



Yubico OTP

6 byte private identity field
Counter
Timer field
Random number
CRC16 checksum

Coding it

```
$composer require enygma/yubikey
```

Usage:

```
$v = new \Yubikey\Validate($apiKey, $clientId);
$response = $v->check($_POST['yubikey_code']);

if ($response->success() === true) {
    // allow into website
}
```

That's all there is to it!

Thank you!

<https://joind.in/talk/b52c8>

Rob Allen - <http://akrabat.com> - @akrabat