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

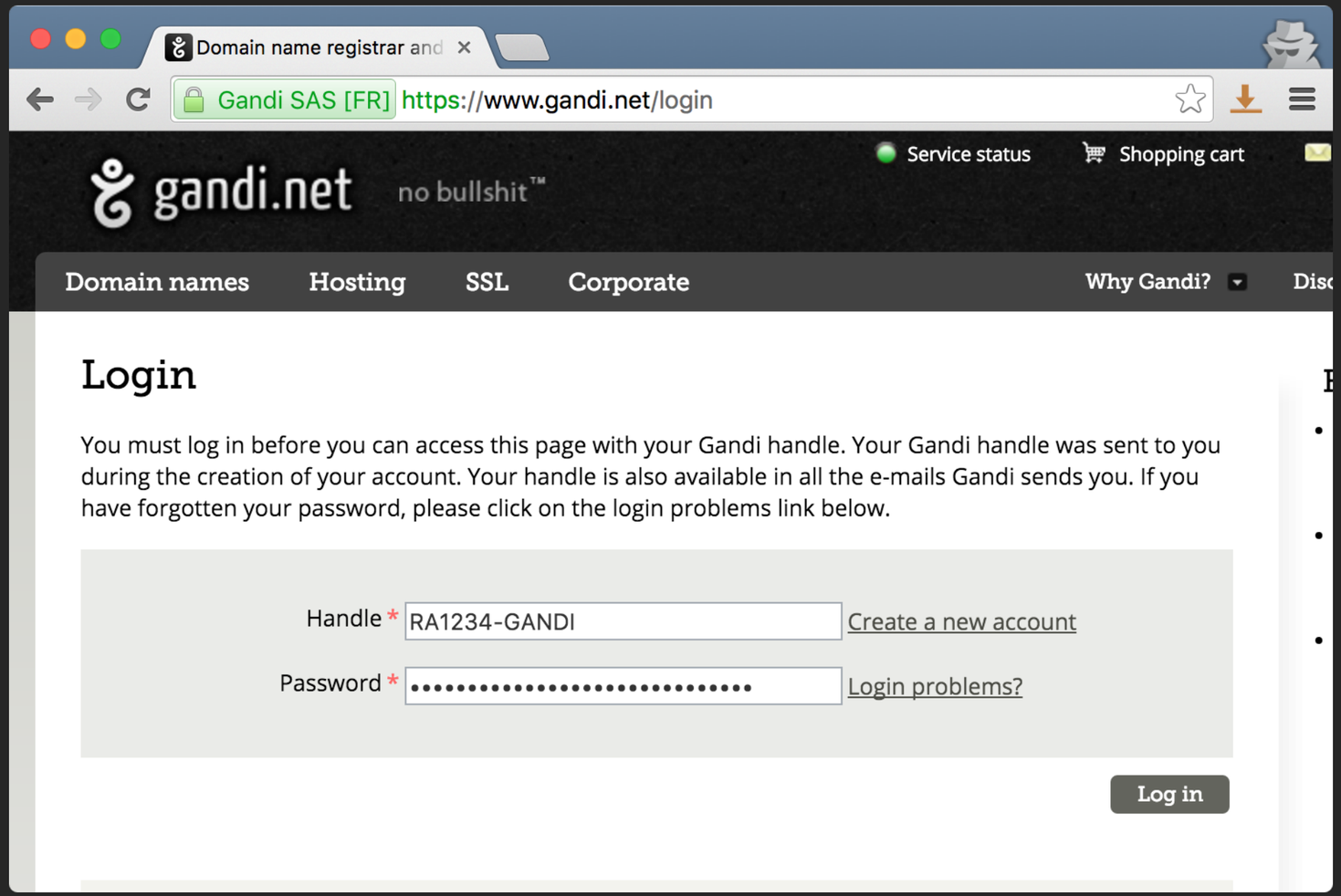
It will take 14 minutes* to crack one of your users' passwords

*English word, stored using bcrypt

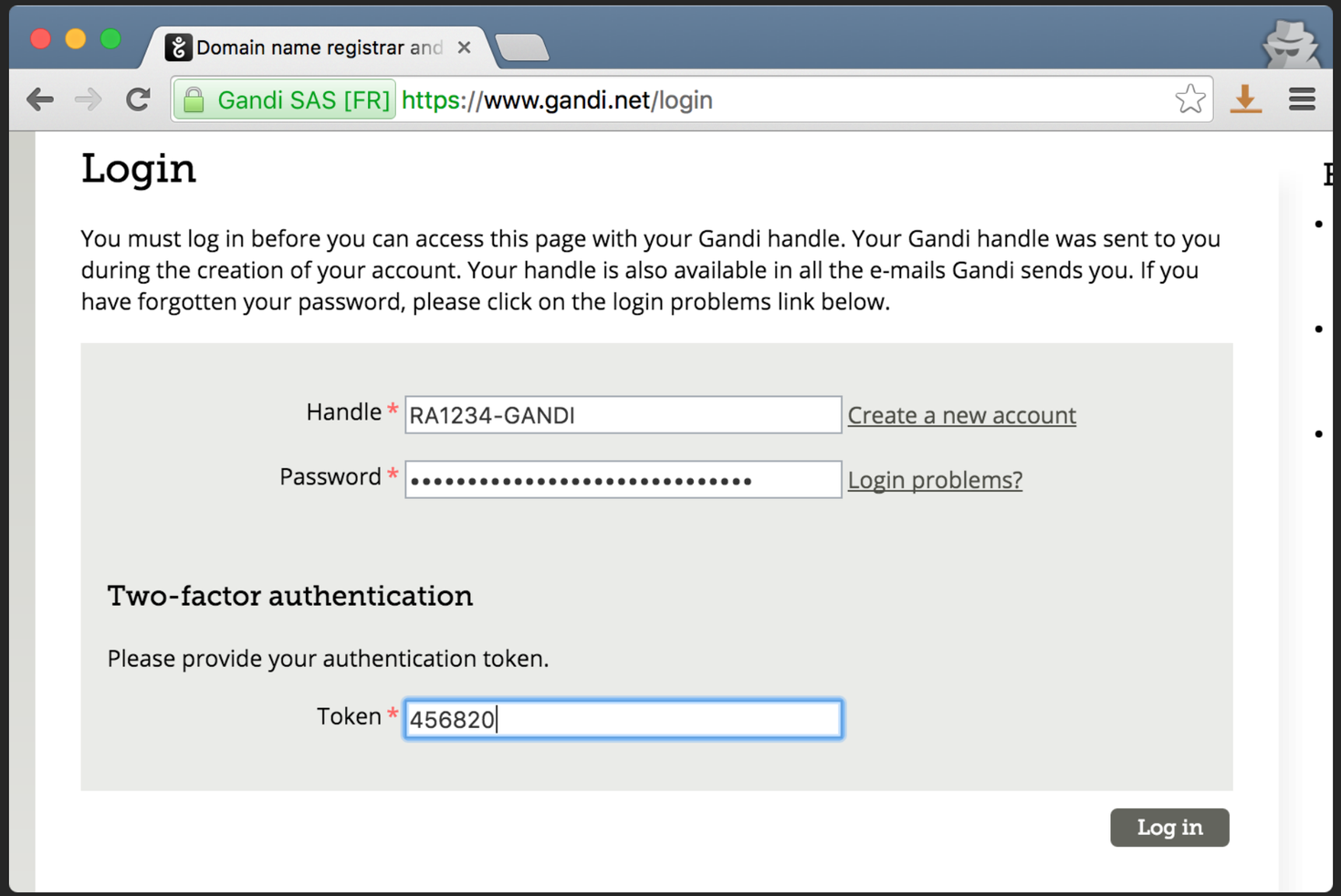
Two-factor authentication
protects your users

What is 2FA?

Implementation on a website



Implementation on a website



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

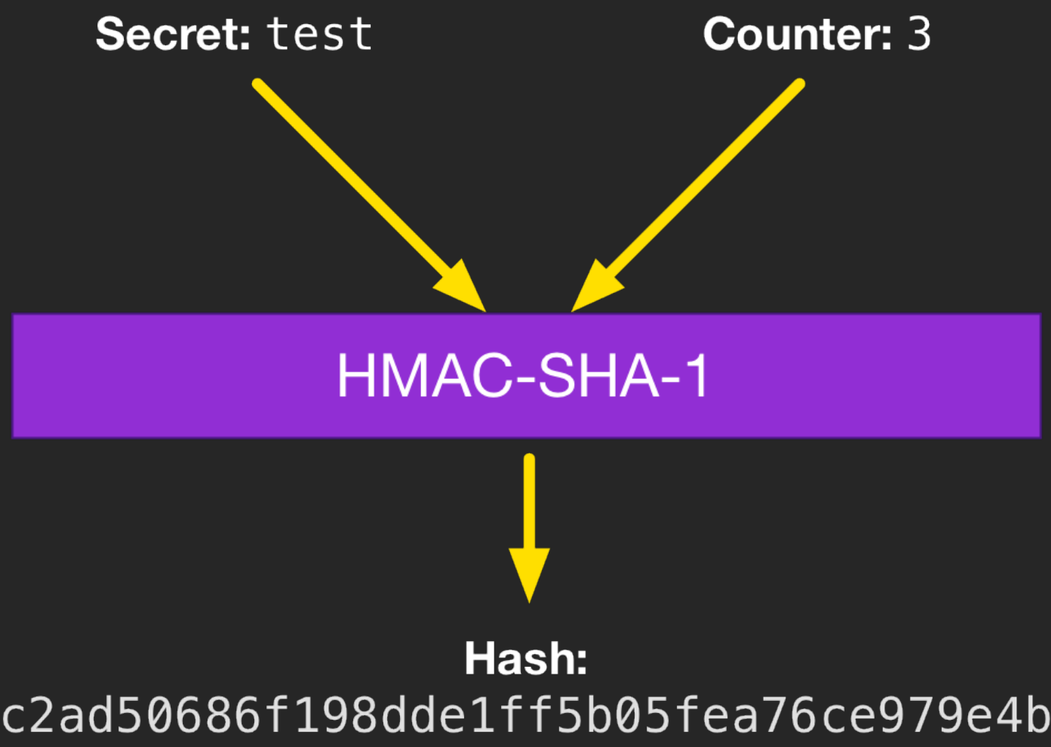
- Easy to use
- No Internet or cellular connection required
- App is free and trusted

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`

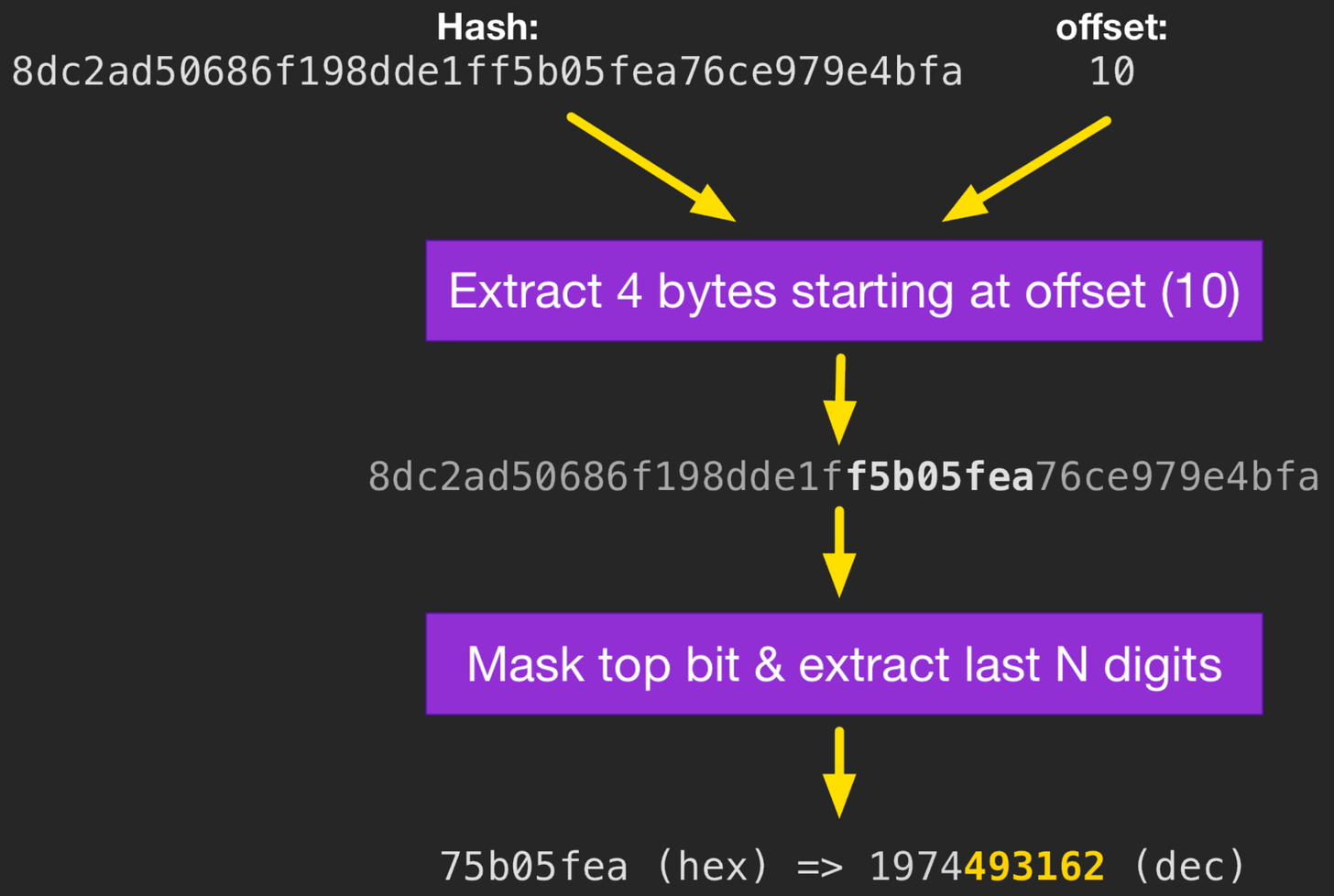
Find lower 4 bits of last byte

`8dc2ad50686f198dde1ff5b05fea76ce979e4bf`**`a`**

offset:

`0x0A` (hex) => `10` (decimal)

HOTP algorithm: step 3



HOTP in PHP

```php
function hotp($secret, $counter)
{
  $bin_counter = pack('J*', $counter);
  $hash = hash_hmac('sha1', $bin_counter, $secret, true);

  $offset = ord($hash[19]) & 0xf;

  $bin_code =
      ((ord($hash[$offset+0]) & 0x7f) << 24 ) |
      ((ord($hash[$offset+1]) & 0xff) << 16 ) |
      ((ord($hash[$offset+2]) & 0xff) <<  8 ) |
      (ord($hash[$offset+3]) & 0xff);

  return $bin_code % pow(10, 6);
}
```

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

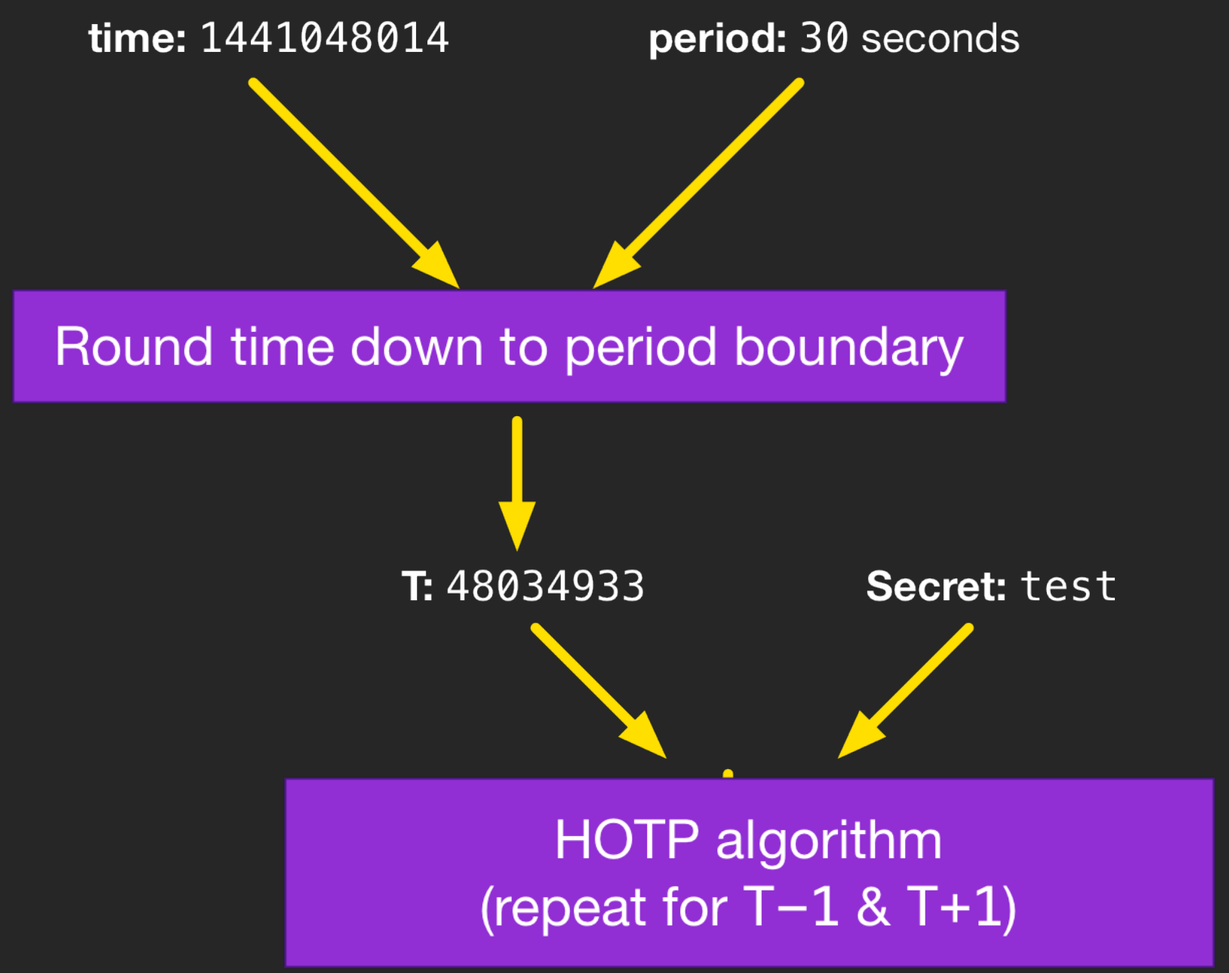
Resync if can't find in look-ahead:

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



TOTP in PHP

```php
function totp($secret)
{
    $counter = floor(time() / 30);

    return hotp($secret, $counter);
}
```

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']);
```

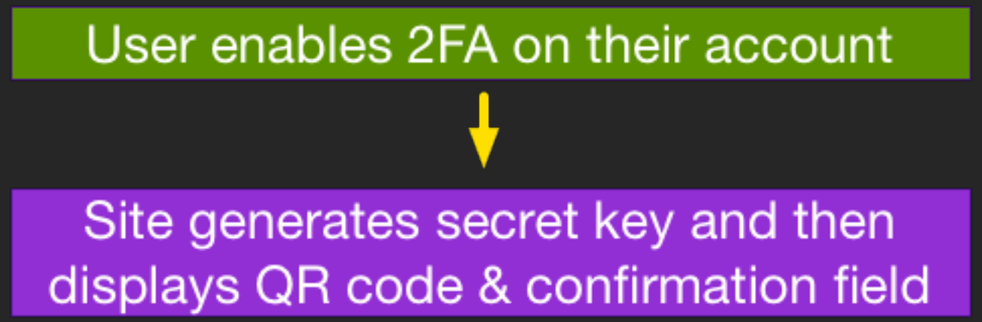
Things to do

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

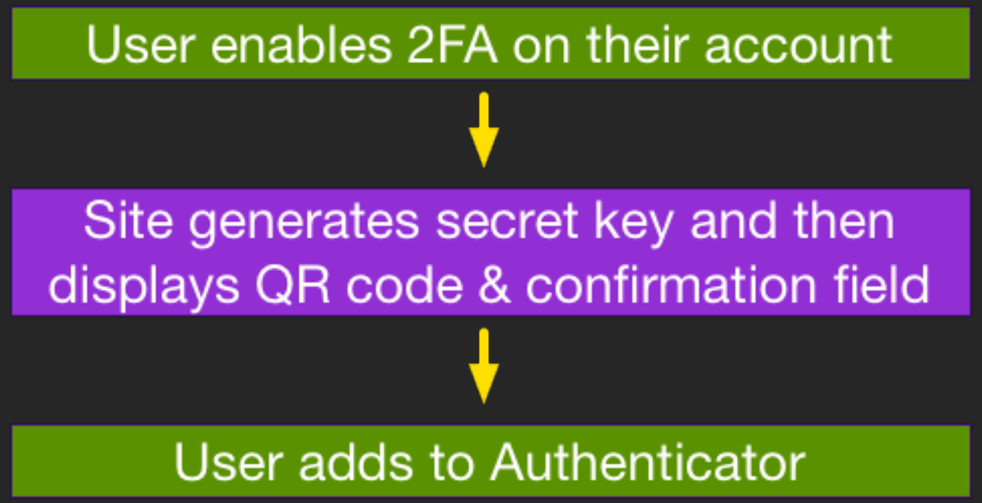
Set up 2FA

User enables 2FA on their account

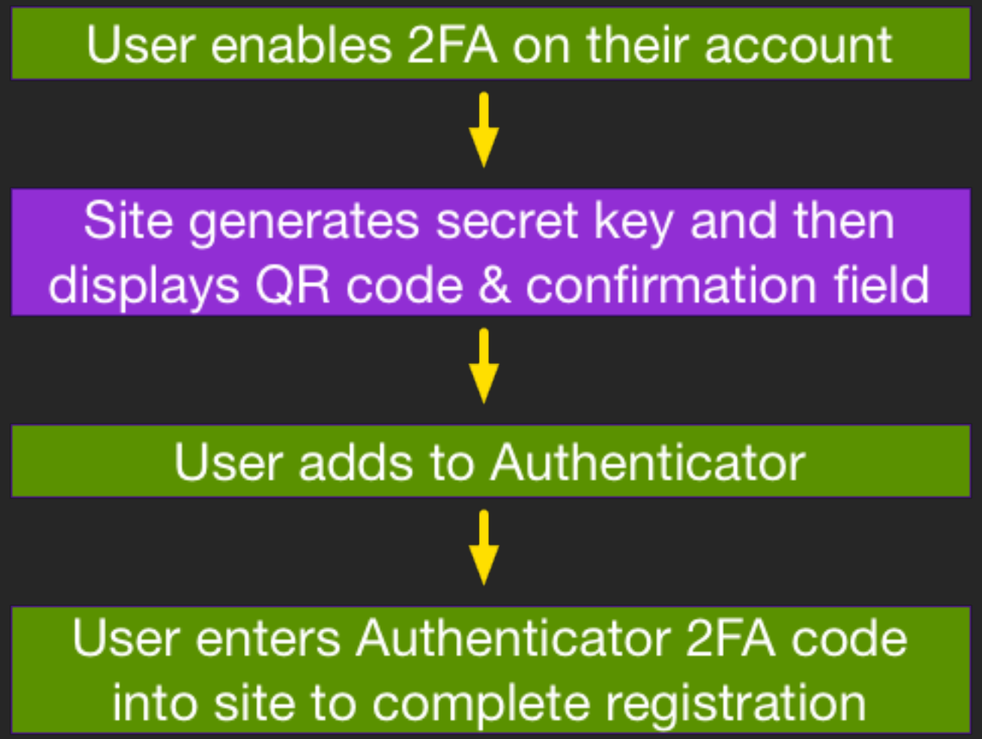
Set up 2FA


User enables 2FA on their account
Site generates secret key and then displays QR code & confirmation field

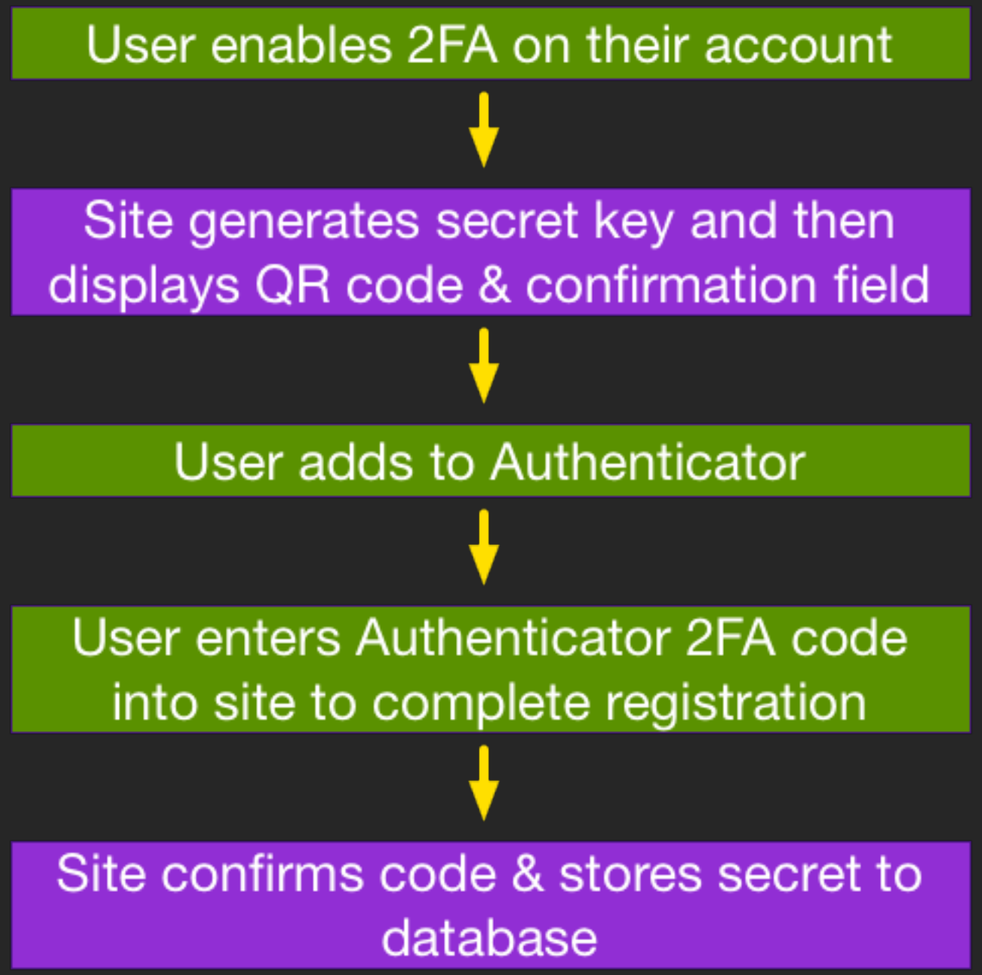
Set up 2FA


User enables 2FA on their account
Site generates secret key and then displays QR code & confirmation field
User adds to Authenticator

Set up 2FA


User enables 2FA on their account
Site generates secret key and then displays QR code & confirmation field
User adds to Authenticator
User enters Authenticator 2FA code into site to complete registration

Set up 2FA


User enables 2FA on their account
Site generates secret key and then displays QR code & confirmation field
User adds to Authenticator
User enters Authenticator 2FA code into site to complete registration
Site confirms code & stores secret to database

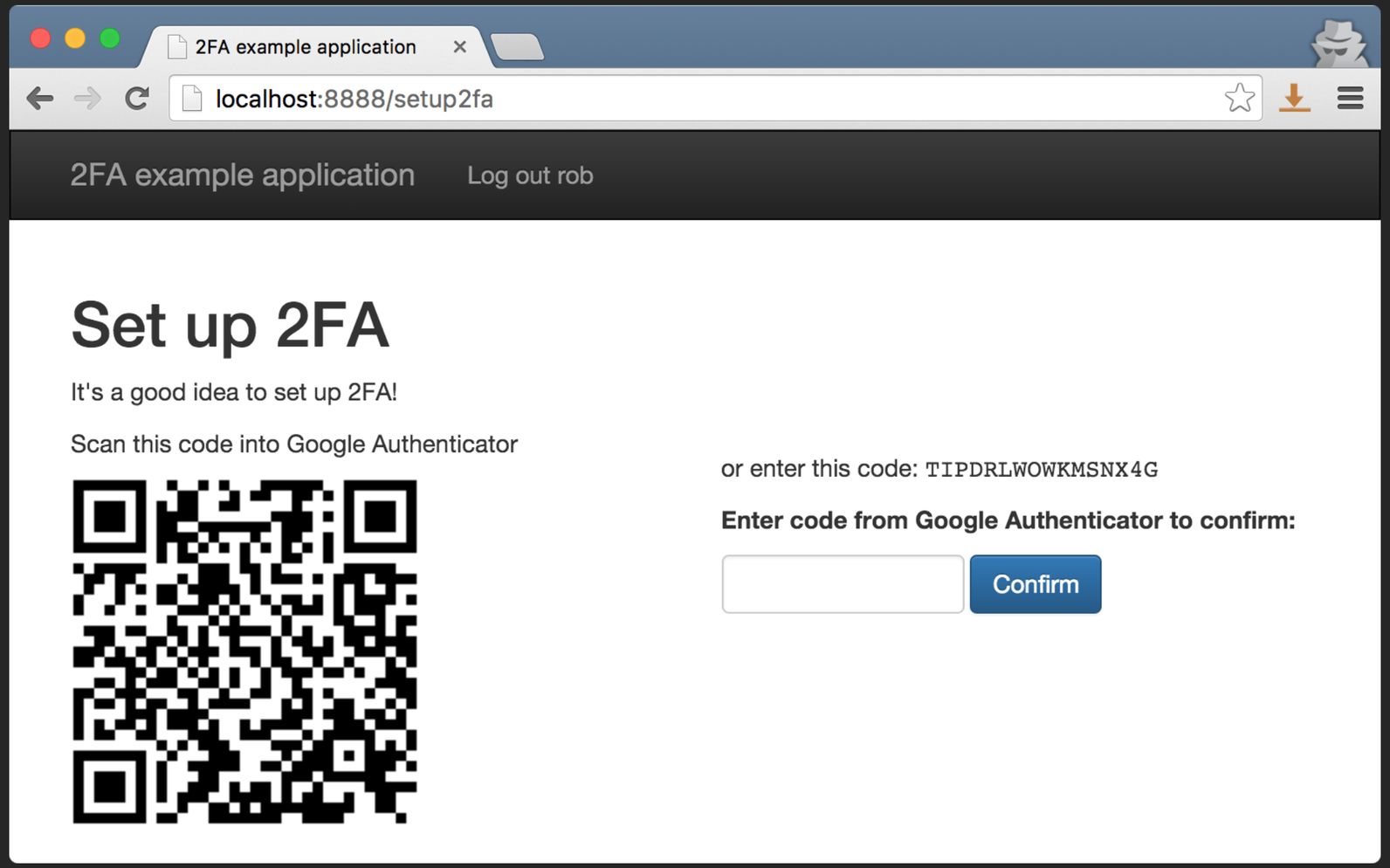
Set up 2FA: code

```php
$app->get('/setup2fa', function () use ($app) {
    $user = $_SESSION['user'];

    $g = new \Google\Authenticator\GoogleAuthenticator();
    $secret = $g->generateSecret();
    $qrCodeUrl = $g->getURL($user->getUsername(),
        '2fa.dev', $secret);

    $app->flash('secret', $secret);
    $app->render('setup2fa.twig', [
        'user'      => $_SESSION['user'],
        'secret'    => $secret,
        'qrCodeUrl' => $qrCodeUrl,
    ]);
});
```

Set up 2FA: template

```
<h1>Set up 2FA</h1>

<p>Scan this code into Google Authenticator:</p>
<img src="{{ qrCodeUrl }}">
<p>or enter this code: <tt>{{ secret }}</tt></p>

<p>Enter code from Google Authenticator to confirm:</p>
<form method="POST" action="/setup2fa">
    <div class="form-group">
        <input name="code" maxlength="6">
        <button type="submit">Confirm</button>
    </div>
</form>
```

Set up 2FA


2FA example application
localhost:8888/setup2fa
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

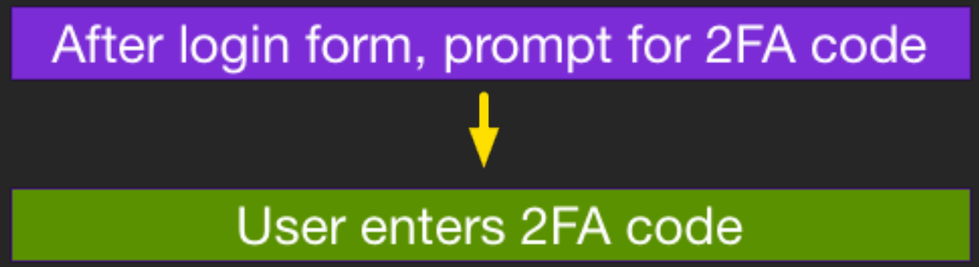
Set up 2FA: process submission

```php
$app->post('/setup2fa', function () use ($app) {
    $secret = $app->environment['slim.flash']['secret'];
    $code = $app->request->post('code');

    $g = new \Google\Authenticator\GoogleAuthenticator();
    if ($g->checkCode($secret, $code)) {
        /* successful registration: store secret into user's row in db */

        $app->redirect('/');
    }

    $app->flash('error', 'Failed to confirm code');
    $app->redirect('/setup2fa');
});
```

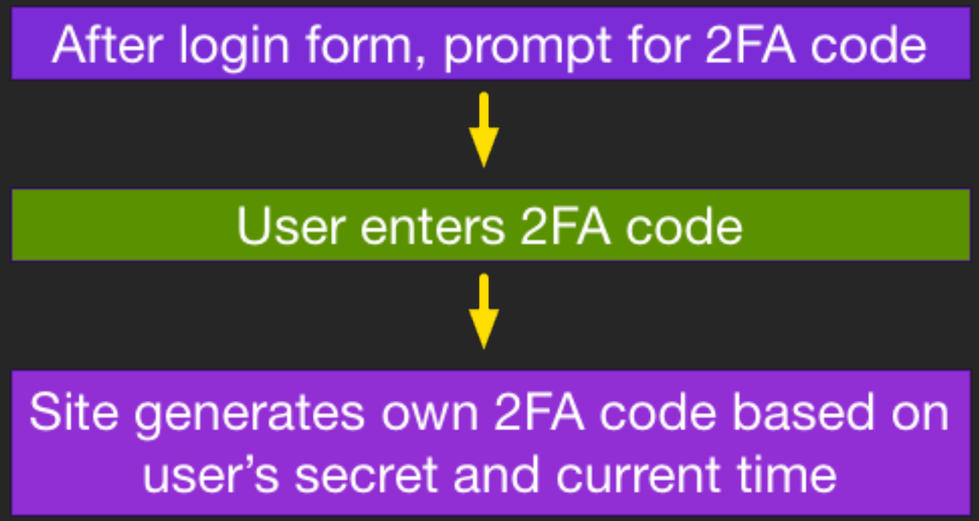
Logging in

After login form, prompt for 2FA code

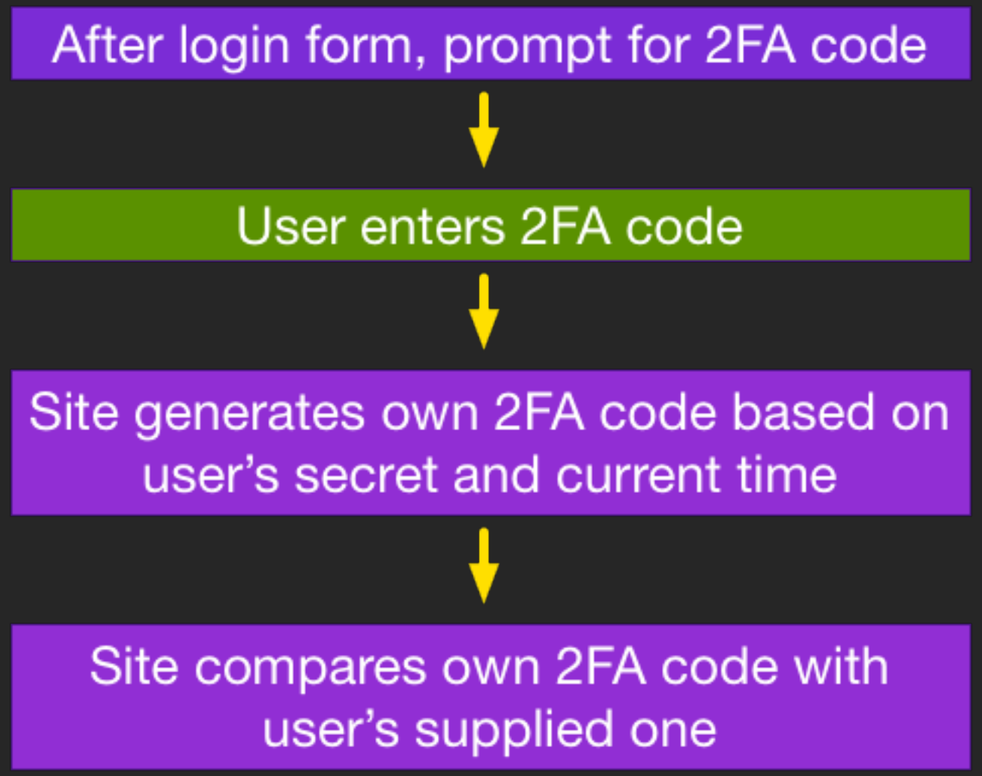
Logging in



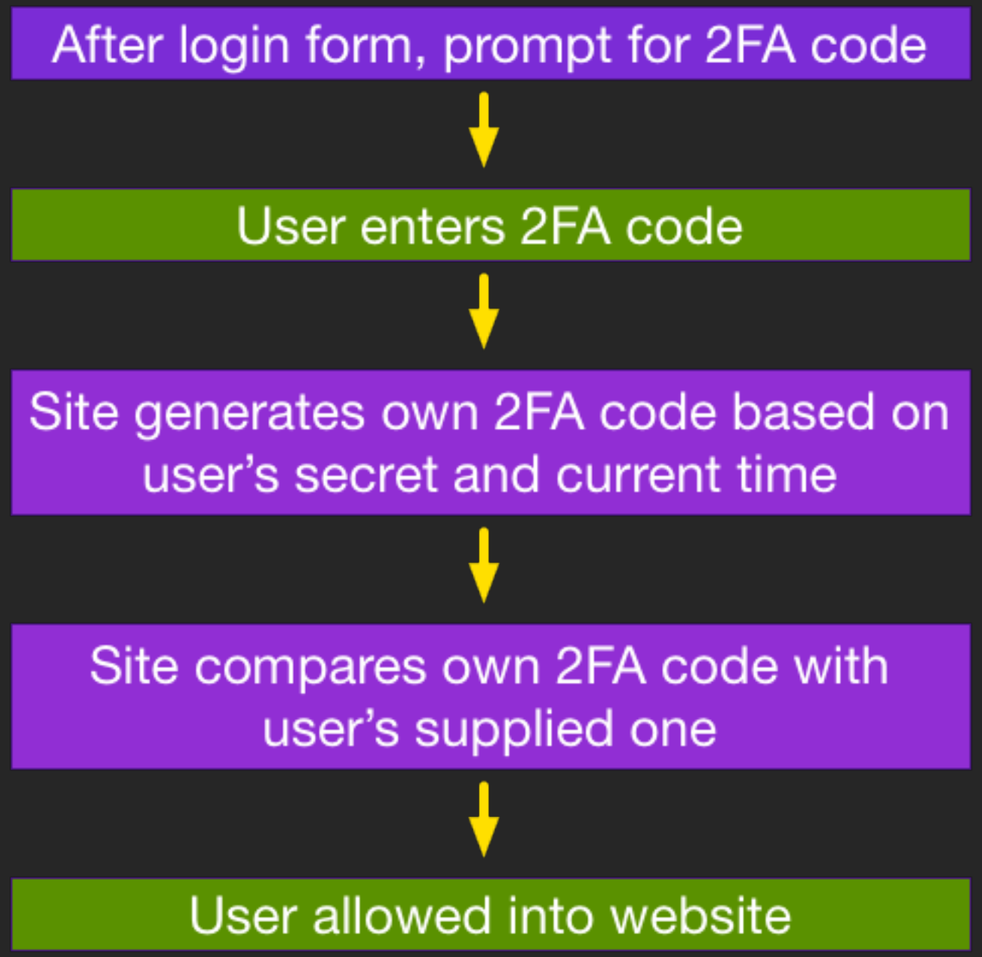
Logging in


After login form, prompt for 2FA code
User enters 2FA code
Site generates own 2FA code based on user's secret and current time

Logging in


After login form, prompt for 2FA code
User enters 2FA code
Site generates own 2FA code based on user's secret and current time
Site compares own 2FA code with user's supplied one

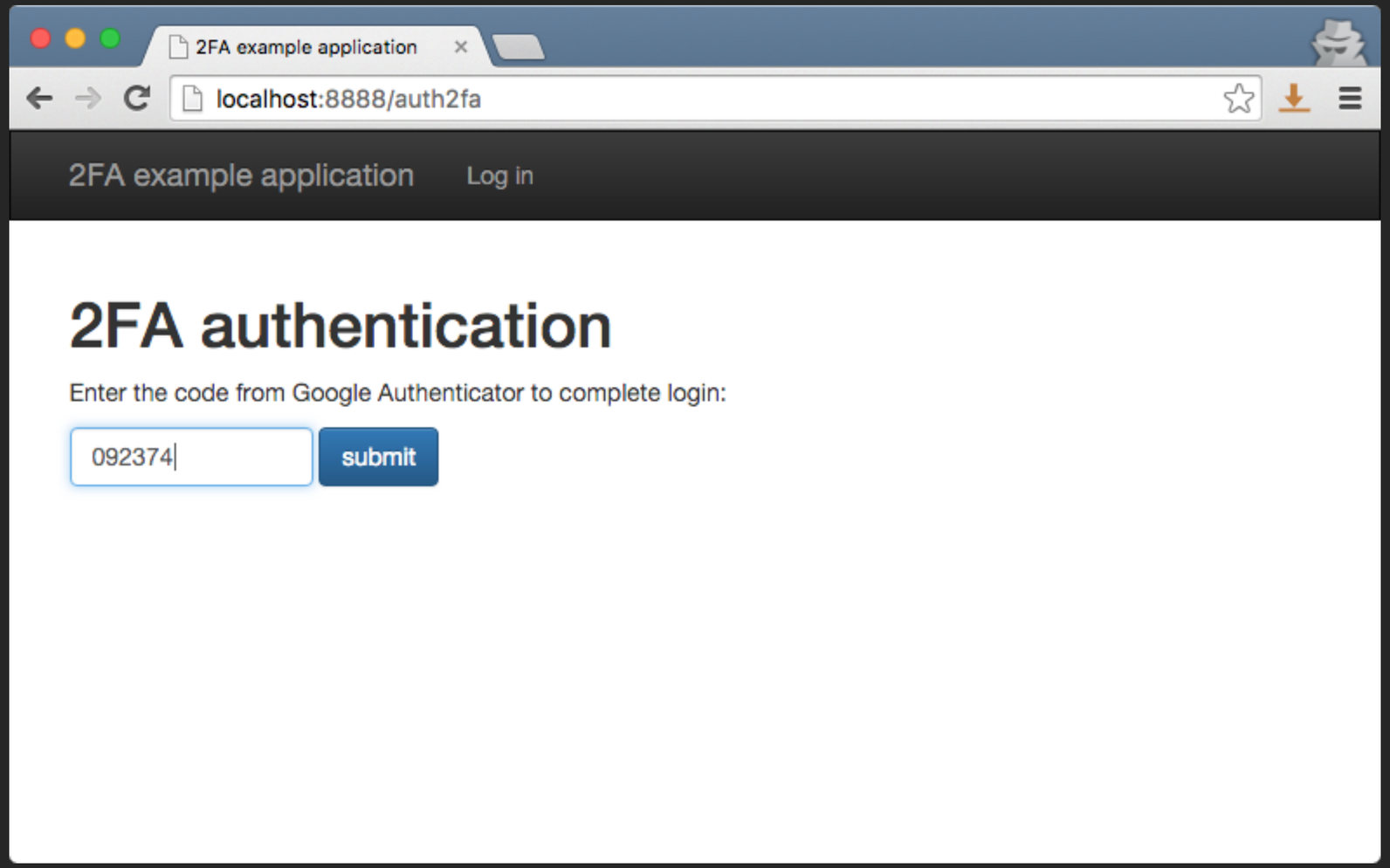
Logging in


After login form, prompt for 2FA code
User enters 2FA code
Site generates own 2FA code based on user's secret and current time
Site compares own 2FA code with user's supplied one
User allowed into website

Login: 2FA code form

```
<h1>2FA authentication</h1>

<p>Enter the code from Google Authenticator to complete login:</p>

<form method="POST" action="/auth2fa">
    <div>
        <input name="code" maxlength="6">
        <button type="submit">Submit</button>
    </div>
</form>
```

2FA code


2FA example application
localhost:8888/auth2fa
2FA example application
Log in
2FA authentication
Enter the code from Google Authenticator to complete login:
092374
submit

Login: process 2FA code

```php
$app->post('/auth2fa', function () use ($app) {
    $user   = $_SESSION['user_in_progress'];
    $secret = $user->getSecret();
    $code   = $app->request->post('code');

    $g = new \Google\Authenticator\GoogleAuthenticator();
    if ($g->checkCode($secret, $code)) {
        /* code is valid! */
        $_SESSION['user'] = $_SESSION['user_in_progress'];

        $app->redirect('/');
    }

    $app->flash('error', 'Failed to confirm code');
    $app->redirect('/auth2fa');
});
```

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

```
cdccccetfjnfigkgkgudlkcbjnjnfrkhkbelbhfvnhcj
cdccccetfjnfuuhtrebhjekcciljdbifgfrlccbnjhkf
cdccccetfjnfljrhetkvijcgddkenbtuiknktejgnvgb
```

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