OWASP and Rails Security

RoR Meetup, December 10, 2013

PayNearMe™

What could happen if your source code was leaked?

PayNearMe™

MongoDB Hacked

- CI and Static Analysis companies used them to store GitHub Keys
- Those GitHub keys could have granted the attacker access to your source code

PayNearMe™

Prezi Source Leaked

- One of their developers inadvertently posted his repo credentials
- White hat hacker was able to grab all of their source without their knowledge

Prezi

PayNearMe™

What is OWASP?

OWASP Background

- Founded in 2001
- Non-profit organization
- Produces lots of material on how to secure web applications
- Top 10 is an ongoing list of the most important web app vulnerabilities

PayNearMe™

OWASP Top 10, 2013

1. Injection
2. Broken Authentication
3. Cross Site Scripting
4. Insecure Direct Object References
5. Security Misconfiguration
6. Sensitive Data Exposure
7. Missing Function Level Access Control
8. CSRF
9. Vulnerable Components
10. Unvalidated Forward/Redirect

OWASP Top 10, 2013

1. Injection
2. Broken Authentication
3. Cross Site Scripting
4. Insecure Direct Object References
5. Security Misconfiguration
6. Sensitive Data Exposure
7. Missing Function Level Access Control
8. CSRF
9. Vulnerable Components
10. Unvalidated Redirects

PayNearMe™

Injection

Allowing non-sanitized user data into persistent data queries.

Injection

Most obvious example

```
User.where("email LIKE '%#{params[:email]}%'")
```

Less obvious example

```
User.find(params[:id]).update_attributes(params[:user])
```

Injection

Solution is to use scopes or Squeel

Turn this bad code

```
User.where("email LIKE '%#{params[:email]}%'")
```

into this sanitized code

```
User.where{email =~ “%#{params[:email]}%”}
```

Injection

This update allows them to get the admin role

```
User.update_attributes(params[:user])
```

This does not

```
good_params = params[:user].slice(:name, :email)
User.update_attributes(good_params)
```

Cross Site Scripting

Allowing user injected Javascript to run in your site.

Cross Site Scripting

Most obvious example

```
%p=@user.biography.html_safe
```

Less obvious example

```
%p=link_to @user.name.html_safe, @user.homepage
```

Cross Site Scripting

Solution #1

`Don't use html_safe`

Solution #2

`Use a sanitizer gem like rgrove/sanitize`

PayNearMe™

Insecure Direct Object Reference

Allowing a user to access data they should not access.

PayNearMe™

Insecure Direct Object Reference

Using file references

```
send_file “docs/#{params[:id]}.pdf”
```

Using a UNIX command

```
`rake gen:test_data[#{params[:test_id]}]`
```

Insecure Direct Object Reference

Use whitelists for file references

Use thoughtbot/cocaine for UNIX commands

Or just don’t use UNIX commands

Missing Authorization

Every secure page needs to authorize the user against the used data

Missing Authorization

1. Non-admin user can read, but not update an order
2. They navigate to the order show page
    - /site/123/order/456
3. They hand edit the url to this
    - /site/123/order/456/edit

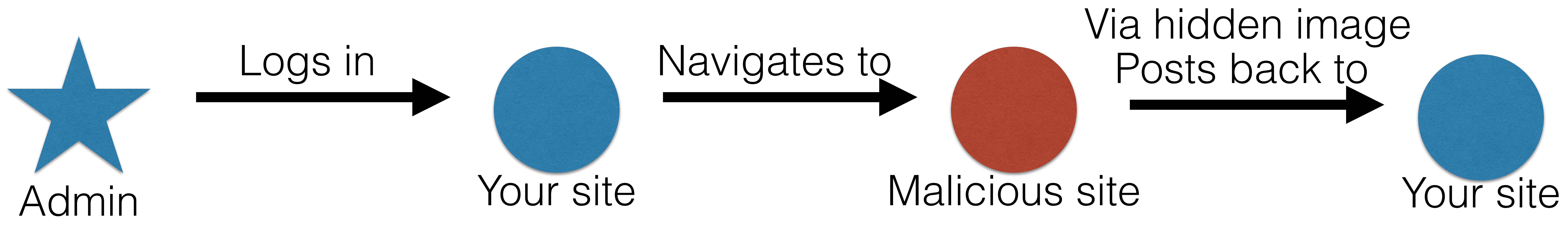
With authorization they should not see that page

Cross Site Request Forgery

Accepting potentially dangerous data from other domains

PayNearMe™

Cross Site Request Forgery



PayNearMe™

Cross Site Request Forgery

Solution Part 1

`Disable posting from other domains.`
`See rhk/rack_protection.`

Solution Part 2

`Always use POST for editing data`

PayNearMe™

Unvalidated Redirects

Redirecting a user to an unvalidated URL

PayNearMe™

Unvalidated Redirects

```
render params[:page]
```

Unvalidated Redirects

```
redirect_to @user.website
```

Unvalidated Redirects

Dynamic Rendering

```
case params[:page]
  when ‘show’
   render :show
  when ‘edit’
   render :edit
  else
   render :index
end
```

Unvalidated Redirects

Dynamic Redirect

Either avoid doing it or use an interstitial page.

Never trust user input!

How many vulnerabilities in your code?

Tools to Answer That

- Brakeman gem
  - good to get started
- Code Climate
  - good for ongoing analysis
  - Coupon! IFU15MA2

CODE CLIMATE

PayNearMe™

Demo Time

PayNearMe™