Project 2

• Deadlines

  • Pt 1: Tuesday, May 12

  • Pt 2: Tuesday, May 19

• Cross Site Scripting, SQL Injection, CSRF, Session Hijacking, Click-Jacking
Outline

• Ruby on Rails
• Forms
• Javascript
• Warmup Exercise Demo
Rails Basics

• Start Server: `rails server` (CTRL+C to stop)

• Database
  • `rails dbconsole` (to access SQLite)
  • `rake db:reset` (resets database)
Directory Structure

- app/
- config/routes.rb
- config/initializers
- db/schema.rb
- db/seeds.rb
Models

- Object-Relation Mapping
- User object => users table

```ruby
@user = User.new
@user.username = username
@user.salt = generate_random_salt
@userhashed_password = hash_password(password, @user.salt)
@user.profile = ""
@user.bitbars = 200
@user.save

@user = User.find_by_username(@username)
```
Views

- .html.erb files
- Located in folders associated with each controller
- Pass in variables from controller
  - `<%= @user.username %>`
  - `<% if %> ... <% end %>`
Controllers

**login_controller.rb:**

```
def login
  render :login_form
end

def post_login
  @username = params[:username]
  @password = params[:password]

  @user = User.find_by_username(@username)

  # do stuff
  if @error
    render :login_form
  else
    session[:logged_in_id] = @user.id
    render :login_success
  end
end

def logout
  if @logged_in_user then
    reset_session
  end
  redirect_to(:controller => "main", :action => "index")
end
```

- Routes
- Define “actions”
- Render views
- Set / pass variables
- Redirect to actions
Forms

- Action specifies destination URL
- GET method sends data in URL (e.g. search queries)
  - google.com/q=cats+steal+dog+beds
  - localhost:3000/profile?username=cathy&password=cat
- POST method sends data in HTTP body
Forms

- Retrieve data using params[:name]
Javascript (DOM)

- Each html document has elements
- Javascript can access, create, delete elements
- [http://www.w3schools.com/js/js_htmldom.asp](http://www.w3schools.com/js/js_htmldom.asp)
Javascript (DOM)

- document
  - document.getElementById()
  - document.getElementsByTagName()
- window
  - window.location='www.evil.com'
- element
  - element.id, element.innerHTML, element.style
- var iframe = document.getElementsByTagName('iframe')
- iframe.contentDocument (same origin as parent page)
Javascript

- XMLHttpRequest
  - var req = new XMLHttpRequest();
  - req.open("GET", "google.com", false);
  - req.send();

- Timers: setTimeout(function(){alert('hey')}, 300);

- Event Listeners: onload, onmouseover, onclick, onsubmit

- https://www.owasp.org/index.php/Cross-site_Scripting_%28XSS%29
Warmup Demo
Mechanize

- Automating web interactions

```ruby
#!/usr/bin/env ruby
require 'mechanize'
agent = Mechanize.new
page = agent.get "http://localhost:3000/login"
form = page.forms.first
form['username'] = 'attacker'
form['password'] = 'attacker'
results = agent.submit (form)
cookie = agent.cookie_jar.jar
puts cookie
```

- [http://www.rubydoc.info/gems/mechanize/Mechanize](http://www.rubydoc.info/gems/mechanize/Mechanize)
- [https://github.com/sparklemotion/mechanize](https://github.com/sparklemotion/mechanize)