Web Security & Rails

Perspective

- Academic Background (National Merit, BSEE, MSE, DSP, Neural Nets, Speech Recognition)
- Work Background (Medical Research, DSP Algorithm, Kernel, Telecom Embedded, CORBA, J2EE, Rails)
- Technical Background (HW 2, C 7, Java 8, Ruby 2, MGR 6)
- Rails Background (MDMS 2.5 years, HashRocket)

Why Security?

- Rails Developers = Web App Developer
- Web App Developer = Expected Security Aware
- As Internet User basic security concepts are good to know!
- Online Banking, Retirement Funds, Social Networking, etc.

GitHub Hacked March 4, 2012

- GitHub was hacked today in a way that exposed every repository. Russian hacker Egor Homakov discovered a public key form update vulnerability that allowed him (or anyone else, for that matter) to access any GitHub repository with full administrator privileges. As a result, anyone could, for example, commit to master, reopen and close issues in Issue Tracker, or even wipe the entire history of any GitHub project.
- http://www.zdnet.com/blog/security/how-github-handled-getting-hacked/10473

Rails Security Overview

- Good Sources of Information
- Identifying Vulnerabilities to Patch
- Development Practices (Web App Scenarios, Developer Security Topics)

Useful Links

- Site Security Policy:http://rubyonrails.org/security
- Rails Security Guide: http://guides.rubyonrails.org/security.html
- Open Web Applications Security Project: https://www.owasp.org/index.php/Top_10
- Egor Homakov Blog: http://homakov.blogspot.com
- OWASP Rails CheatSheet:

  https://www.owasp.org/index.php/Ruby_on_Rails_Cheatsheet

Finding Vulnerabilities

Rails Vulnerabilitiies:
http://web.nvd.nist.gov/view/vuln/search-results?query=Rails

(For Erlang):
http://web.nvd.nist.gov/view/vuln/search-results?query=Erlang

Example MDMS Vulnerabilities

- CVE-2013-0233 (MDMS ok with 1.4.9, but careful if upgrading devise)
- CVE-2013-3221 Active Record (MDMS 1.5 uses 3.0.9, MDMS 2.5 with 3.1.8)
- CVE-2013-0277 (MDMS 1.5 use Active Record 3.0.9)
- CVE-2013-0269 (MDMS 1.5 and MDMS 2.5 uses json 1.7.5)

•CVE-2013-0233 (careful if upgrading GEM)

- Summary: Devise gem 2.2.x before 2.2.3, 2.1.x before 2.1.3, 2.0.x before 2.0.5, and 1.5.x before 1.5.4 for Ruby, when using certain databases, does not properly perform type conversion when performing database queries, which might allow remote attackers to cause incorrect results to be returned and bypass security checks via unknown vectors, as demonstrated by resetting passwords of arbitrary accounts.
- Published: 04/25/2013
- Severity: 6.8

•CVE-2013-3221 (MDMS 1.5 uses 3.0.9, MDMS 2.5 with 3.1.8)

- Summary: The Active Record component in Ruby on Rails 2.3.x, 3.0.x, 3.1.x, and 3.2.x does not ensure that the declared data type of a database column is used during comparisons of input values to stored values in that column, which makes it easier for remote attackers to conduct data-type injection attacks against Ruby on Rails applications via a crafted value...
- Published: 04/22/2013
- CVSS Severity: 6.4 (MEDIUM)

CVE-2013-0277 (MDMS 1.5)

- Summary: ActiveRecord in Ruby on Rails before 2.3.17 and 3.x before 3.1.0 allows remote attackers to cause a denial of service or execute arbitrary code via crafted serialized attributes that cause the +serialize+ helper to deserialize arbitrary YAML.
- Published: 02/13/2013
- CVSS Severity: 10.0 (HIGH)

CVE-2013-0269 (MDMS 1.5/2.5)

- Summary: The JSON gem before 1.5.5, 1.6.x before 1.6.8, and 1.7.x before 1.7.7 for Ruby allows remote attackers to cause a denial of service (resource consumption) or bypass the mass assignment protection mechanism via a crafted JSON document that triggers the creation of arbitrary Ruby symbols or certain internal objects, as demonstrated by conducting a SQL injection attack against Ruby on Rails, aka "Unsafe Object Creation Vulnerability."
- Published: 02/13/2013
- CVSS Severity: 7.5 (HIGH)
-

Developer Practices

- Web Architecture (Scenarios)
- Developer Security Topics

Typical Web App Scenarios

Client	Protocol	Server
OS	HTTP	Linux
Browser	HTTPS	unicorn
Browser Plug-in		ruby
Cookie		Rails/Gems
Javascript		Database
Flash/PDF		

Rails Security Guide Topics

- Session / Cookie
- Cross Site Request Forgery (CSRF)
- Redirection
- File Upload/Download
- Mass Assignment
- Injection
- Cross Site Scripting Attacks (XSS)
- HTTP Headers

Session Attacks

- HiJack
- Fixation
- Cookie Theft
- Replay Attacks

MDMS

- Monaco::Application.config.session_store :cookie_store, :key => “somekey”
- Code in actionpack GEM (cookie_store, filtered_params, request, response, protect_from_forgery, nonce) and devise GEM
- Sign out button, but no SSL
- Reset_session unless operator.active?
- inactivity_timeout 60 min 20 fails locks 1hour
- Cookie Base64encoded, ApplicationController::Base has protect_from_forgery (CSRF protection)

Decoding Cookie

- Base64.decode64(“cookie”) reveals structure and info about your system
- session_id, operator_return_to, _csrf_token, warden.user.operator.key, Operator, BSON::ObjectId, warden.user.operator.session, last_request_at

Recomendations

- Prevent Hijack with SSL and logout button
- Prevent Fixation by expiration and resets
- Cookies can be stolen (can sign/encrypt)
- Nonce to prevent Replay Attacks

File Upload/Download Attacks

- Overwriting files (../../etc/password)
- Obtaining privileged information
- Public web directories or executable content
- Denial of Service attacks with sync operations

Recommendations

- Don't run App as root
- Linux directory permissions
- Sanitize and whitelist file names and paths
- Limit directories allowed

CSRF

- Browser is authenticated in App with valid cookie
- Browser views another site which executes javascript (XSS) to do operations on App's site.
- Examples: changing router dns, change user's email or password, delete or update information, etc

Recommendations

- Proper GET/POST usage (per REST)
- Make POSTS require a security token
- Note: XSS can bypass CSRF token with access to all browser info with these attacks.

Redirection Attacks

- HOW: www.joe.com/login?redirect_to=somewhere
- BAD: redirect_to params[:url]
- BAD URL: http:a.com/redirect?to=www.facebook.com
- BAD Including url param for redirect: http://www.mysite.com/stuff?id=213&url=http:www.mymainsite.com

Other Redirects

- Opera/Firefox use data protocol for XSS attack (data:text/html;base64,??badencodedattack)
- URL obfuscation methods using '@', using encoding, altering IP addresses with display using DWORD format in decimal or hex
- See http://www.pc-help.org/obscure.htm

Recommendations

- Don't allow users to supply any portion of redirect urls
- If you redirect to a URL check it with whitelist and/or regular expression

Mass Assignment

- CONTROLLER: User.new(params[:user]) or User.update_attributes(JSON.parse(request.body))
- MODEL: using attr_accessible (removed rails4) no attr_protected (blacklist) or attr_accessible (whitelist) or other restriction
- GEM strong_parameters available and packaged with rails 4.

MONGODB injection

- User.where(name: params[:name]).first
- Url: http:stuff?name=John%20%20Odendahl
- Params[:name] = “John Odendahl”
- Url: http:stuff?name%5B$lt%5D=John %20%20Odendahl
- Params[:name] = "$lt"=>"John Odendahl"}
- User.where(name: "John Odendahl").first
- User.where(name: { "$lt" => "John Odendahl"}).first

SQL Injection

- More common attacks, but we use MONGODB
- http://rails-sqli.org
- As MONGODB becomes more common, targets attacking it may become more prevalent.

Javascript XSS Paths

- <script>alert('Hello')</script>
- <img src=javascript :alert('Hello')>
- <table background=”javascript:alert('Hello')”>
- Http://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet

HTTP Headers

- Webrick puts out the version info which makes it easy to attack. It is the default server for rails apps on Heroku.
- Consider Secure_headers Gem or better default behavior in Rails 4 for headers

Webrick Response HEADER #1

- Webrick Response Header: Content-Type text/html; charset=utf-8
- X-Ua-CompatibleIE=Edge
- Etag "dbadbd499d7378af0b2c540471383e0c"
- Cache-Control max-age=0, private, must-revalidate
- X-Runtime 0.448210
- Content-Length 4408
- Server WEBrick/1.3.1 (Ruby/1.9.3/2012-04-20)

WebRick Respone Header #2

- Date Mon, 05 Aug 2013 18:57:13 GMT
- Set-Cookie _monaco_session=BAh7B0kiD3Nlc3Npb25faWQGOgZFRkkiJWM1ZTdjYzI2YzVkZjA2ZjRiZTJiODE1OGM3MWI1ZmRmBjsAVEkiEF9jc3JmX3Rva2VuBjsARkkiMXg5SCtOTFF6WXBYdEZVc1luc2JMdExBN05md2lXVG9YTlU1Q0oyaUduSXc9BjsARg%3D%3D--82ec905926672efdbf2aeb92e765be96d84685ee; path=/; HttpOnly

Everything in Source Control?

- Common to perhaps not locate database.yml and initializer/secret_token.rb in the source control to help prevent disclosure of important information.

Don't have Time?

- Problem: We aren't public facing and don't have time/resources to track and stay on top of all of these issues?
- Answer: Maybe we can just use Brakeman GEM and catch most of the big issues with very little work?

Brakeman WARNINGS#1

- High,,json gem version 1.7.5 has a remote code vulnerablity: upgrade to 1.7.7
- High,,json_pure gem version 1.7.5 has a remote code vulnerablity: upgrade to 1.7.7
- High,,Rails 3.1.8 has a remote code execution vulnerability: upgrade to 3.1.10 or disable XML parsing
- High,,"All versions of Rails before 3.0.18, 3.1.9, and 3.2.10 contain a SQL Injection Vulnerability: CVE-2012-5664; Upgrade to 3.2.10, 3.1.9, 3.0.18"

Brakeman WARNING#2

- High,,"All versions of Rails before 3.0.19, 3.1.10, and 3.2.11 contain a SQL Injection Vulnerability: CVE-2013-0155; Upgrade to 3.2.11, 3.1.10, 3.0.19"
- High,,Session secret should not be included in version control near line 7
- Weak,,Dynamic Render Path,"Render path contains parameter value near line 25: render(action => actions[+params[:action].to_sym+], {})"

Brakeman CONTROLLER #1

- High,,Cross Site Scripting,Unescaped model attribute in JSON hash near line 39: +CriteriaTemplate.cdn_key_options.to_json+
- High,,Cross Site Scripting,"Unsafe parameter value in link_to href near line 41: link_to(""Transactions"", +params.merge(:session_query => ({ :view => :transactions }))+)"
- High,,Cross Site Scripting,"Unsafe parameter value in link_to href near line 48: link_to(""Sessions"",params.merge(:session_query => ({ :view => :sessions }))+)"

Brakeman CONTROLLER #2

- Medium,,Cross Site Scripting,"Unsafe model attribute in link_to href near line 2: link_to(+(Unresolved Model).new+.key, [:copy_or_edit, component, hub, headend, +(Unresolved Model).new+].compact, :class => ""modal"")"
- Medium,,Cross Site Scripting,Unescaped model attribute near line 39: CriteriaTemplate.cdn_key_options.to_json

Brakeman's To Far Behind on dependencies? Other options?

- Source Code Scanner
- https://github.com/codesake/codesake_dawn
- Commercial Source Revision Monitor:
- Https://gemnasium.com
- Https://gemcanary.com