Securing Ruby and Rails

Cliff Moon
Chariot Solutions

The Routing Vulnerability

- Handling
  - Good Points
    - Patches were implemented very quickly.
    - Prompt notification of the community.
    - The core team responded to users' concerns.
    - No widely reported compromised applications.
  - Bad Points
    - The 1.1.5 release did not completely fix the problem.
    - The 1.1.6 release broke Engines.
    - Full disclosure was not made right away.
    - SQL Injection red-herring in the SVN diffs.

The Routing Vulnerability

- Vulnerability in routing.rb
  - Attacker may run virtually any script within rails_root via url manipulation.
  - Denial of service.
  - Data corruption.

Application Level Vulnerabilities

- SQL Injection
  - Input is not escaped before being added to query strings.
  - Can cause data corruption, unauthorized access, and privilege escalation.
  - Use bound parameters, hash-type where clauses, or find_by_xxx.

Application Level Vulnerabilities

- Parameter Injection
  - Tamper Data plugin
  - ActiveRecord::Base#new(@params) and ActiveRecord::Base#update(@params)
  - attr_protected – marks an attribute as being unassignable via params hash, therefore it must be assigned explicitly.
  - attr_accessible – does the opposite of attr_protected.

Application Level Vulnerabilities

- Cross-Site Scripting
  - Most cases are caused by rendering unlaundered text into an HTML page.
  - sanitize - strips javascript, script tags, and form tags from the input.
  - strip_tags - strips out all HTML tags from the input.
  - textilize & markdown \u2013 formats using BlueCloth and RedCloth, respectively.

Security Plugins

- AgileWebDevelopment.com
  - Large repository of security oriented plugins.
  - All manner of authentication and access control plugins.
  - Security plugins help enforce rules in a DRY manner.

Security Plugins

- Raccess
  - Rule based system for filtering access to model objects.
  - acts_as_filterable – marks a model class as having access rules applied to it
  - invisible_if :method?, visible_if :method? - access control rules are evaluated in order of appearence: the last one to evaluate as true will determine visibility.
  - ActiveRecord::Base#security_ignore() - wraps a block in which access rules will be ignored.
  - http://rubyforge.org/projects/raccess

Security Plugins

- User Authentication
  - script/generate authenticated user account
  - Generates a login and authentication system including migrations, model, view, and controller logic.
  - Can add access control filters to controllers to DRY up login and role based access checks.
  - Can generate user activation mailings.
  - Docs include examples for adding password changing, reversible encrypted passwords, remember me checkbox, and many other modifications.

Questions?

Any questions, concerns, misgivings, or bad feelings?