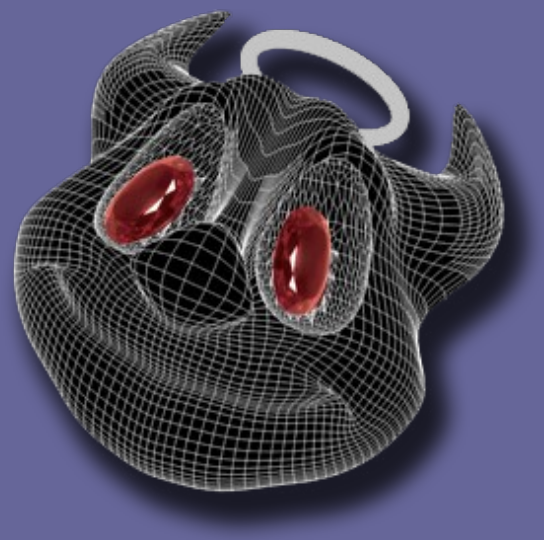
A Beginner's Guide to Security with Ruby on Rails in BSD



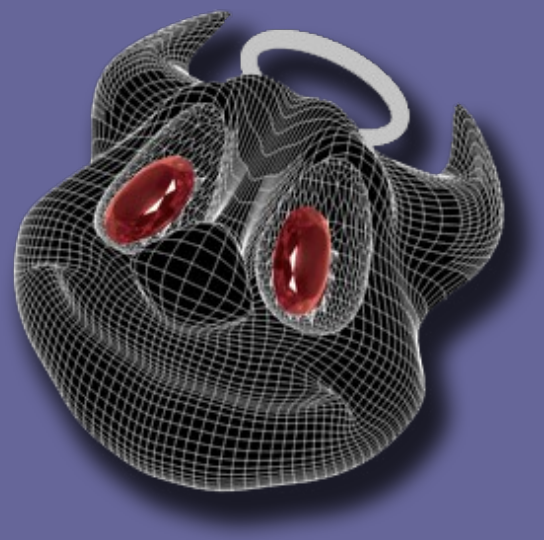
Corey Benninger

What's on Tap?

Ruby on Rails

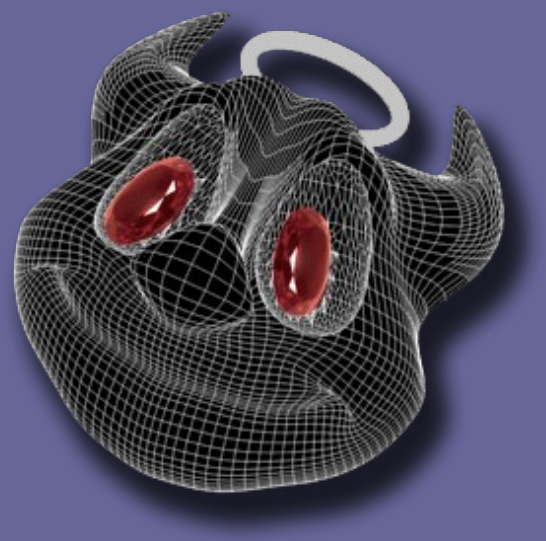
BSD

Security

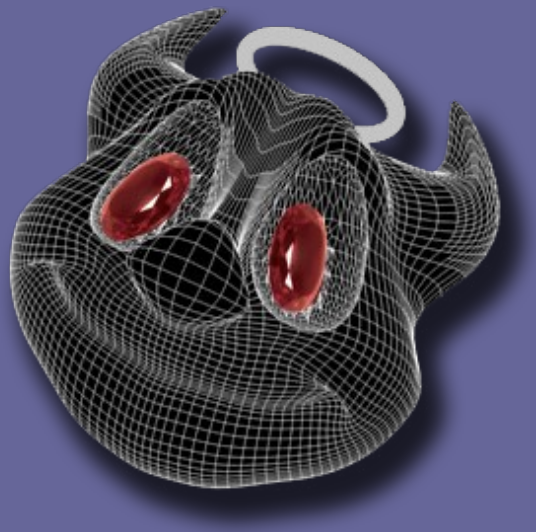


What's on Tap?

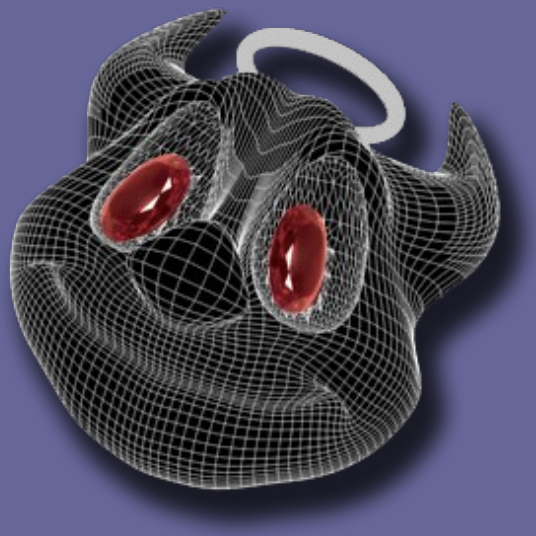
Better able to securely develop, harden, and maintain Rails based applications.



Where's the BSD?



Where's the BSD?

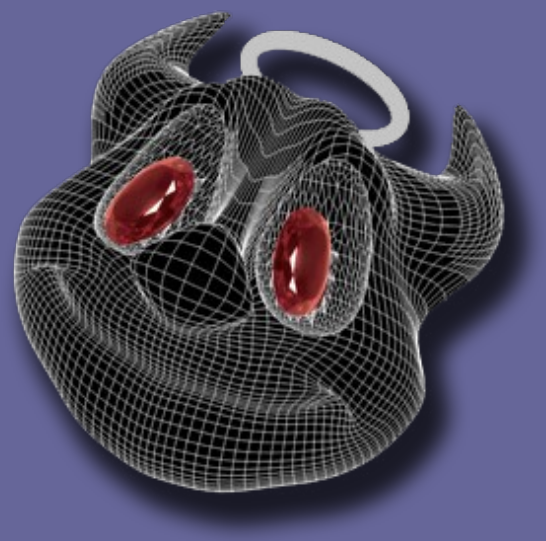


Why Ruby on Rails

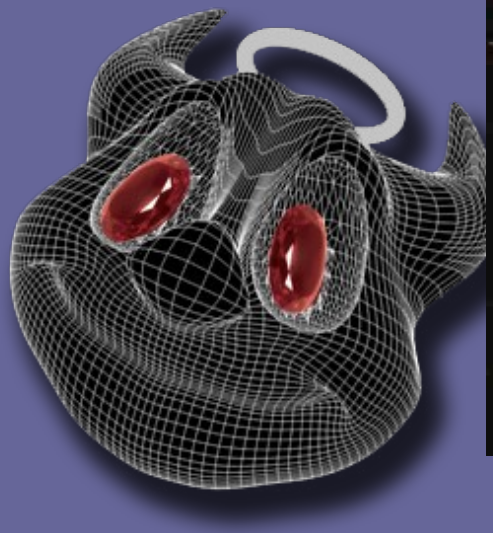
Simplicity -> Security

Don't Repeat Yourself (DRY)

Convention over Configuration

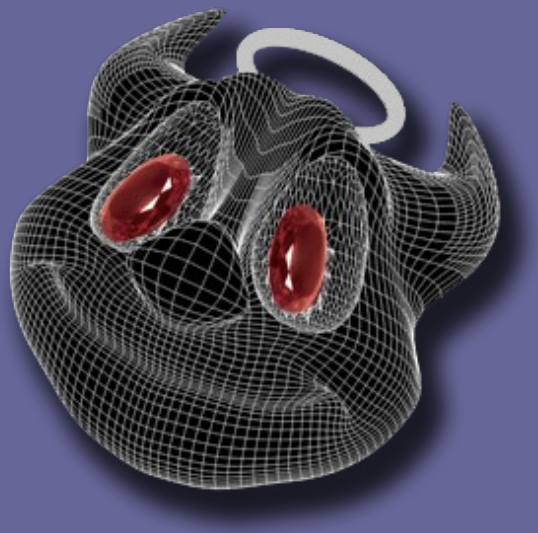


DEFENSE (in Depth)



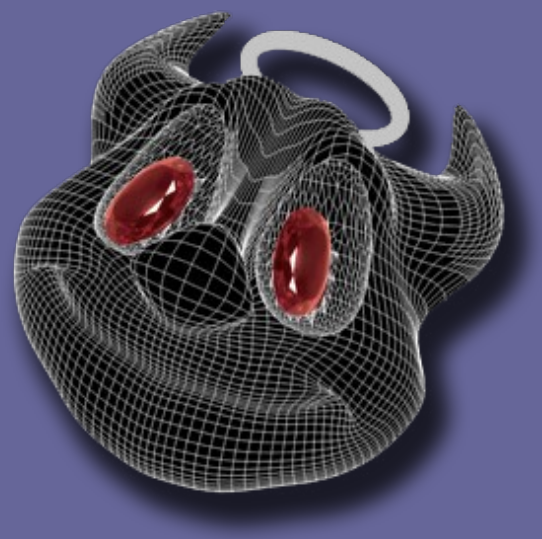
Why I'm Here


CHANNEL 1 INTERNET
A service of Cybergate Inc.
MALL
SERVICES
CHAT
StockQuotes Online
Our Very Own New England Weather
Virtual Servers
V.90
Software Store
Powered by APACHE
FreeBSD
Microsoft FrontPage
Foundstone
A Division of McAfee
Team
Services
Education
EXPERTS IN STRATEGIC SECURITY
Foundstone offers a unique combination of professional services and education to help companies continuously and measurably protect the most important assets from the most critical threats.
Business Con
Technology C
Education
Education Spotlight
What's New
Spotlight
Foundstone Professional Services
Strategic Security
Foundstone provides updated services and education programs to help you respond to your organization's ever-changing security demands.
Business Security Consulting
We thoroughly diagnose your current security posture and lay the foundation for a program to address your current and future needs.
Technology Security Consulting
Our experts uncover vulnerabilities in your network and applications and recommend ways to fortify your defenses.
Education Courses
We offer world-class training for everyone involved in security—from technical staff to management. Register Now!
Secure Software - Essentials
Learn the common application security flaws and how to prevent them by integrating security with software design and development. ...more
Foundstone in the P
Get the latest security world-renowned Found experts.
Foundstone AJAX W
New white paper on A "A Look at Flash Cook Internet Explorer Pers



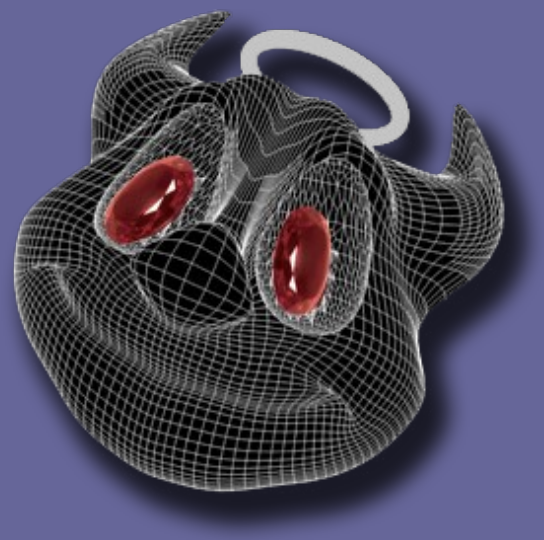
Getting the Goods

- **Ruby** – interpreted scripting language
- **Gems** – the "apt-get" for Ruby packages
- **Rails** – a framework written in Ruby for developing web applications



Getting the Goods

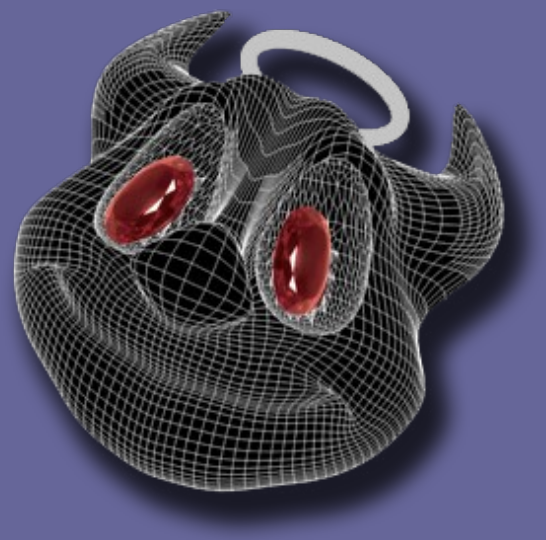
- Openbsd 3.9
  - Install Ruby ***(pkg_add ruby-1.8.4p1.tgz)***
  - Download RubyGems ***(ruby setup.rb)***
  - Install Rails ***(gem install rails --include-dependencies)***



Visit http://www.rubyonrails.org/down
for more install details

Getting the Goods

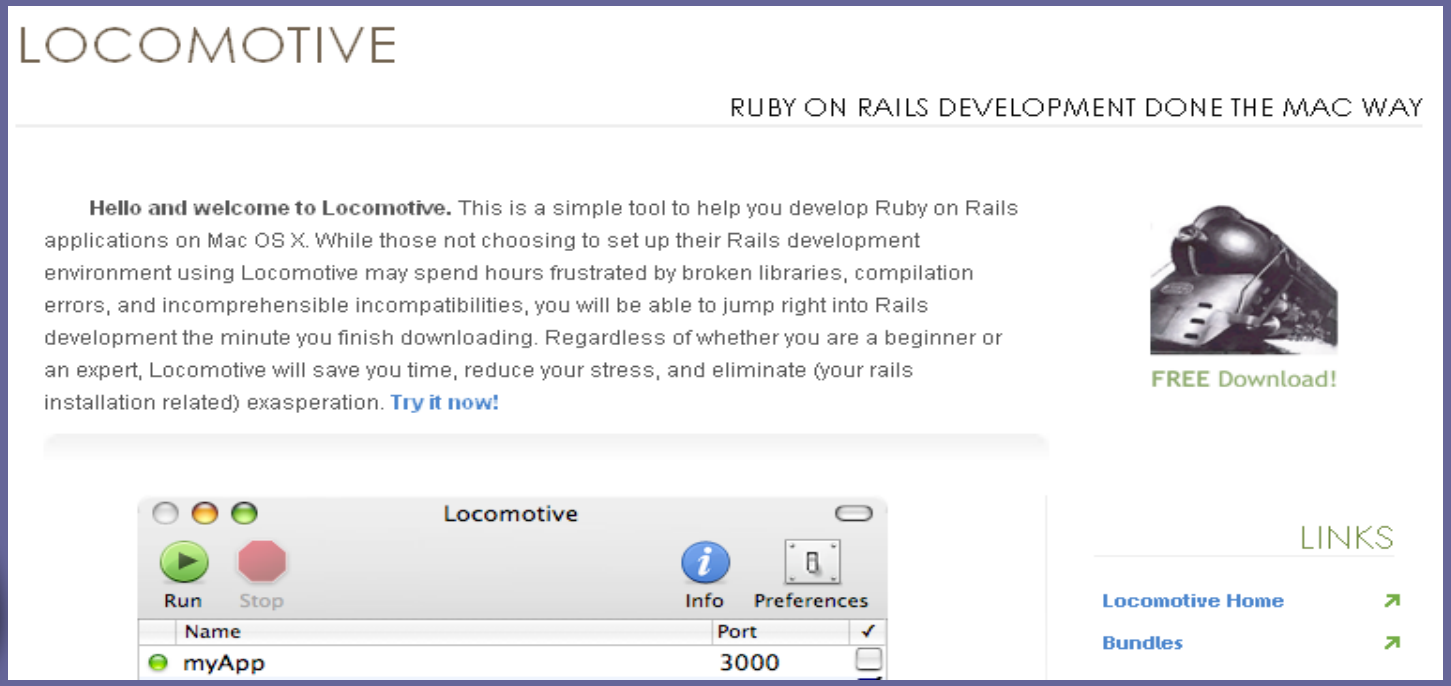
- Mac OS X 10.4
  - OS X pre 10.4.6 ships with broken Ruby
  - Ruby, Rails, and RubyGems will ship with OS X 10.5
  - The Developer Tools – Xcode 2.0 or newer
  - GNU readline

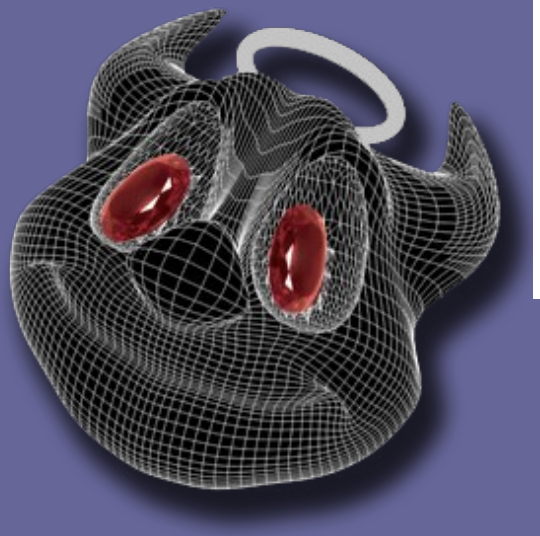


Visit http://hivelogic.com/articles/2005/12/01/ruby_rails_lighttpd_mysql_tiger
for more install details

Getting the Goods

- Mac OS X 10.4
  - Alternatively, download Locomotive

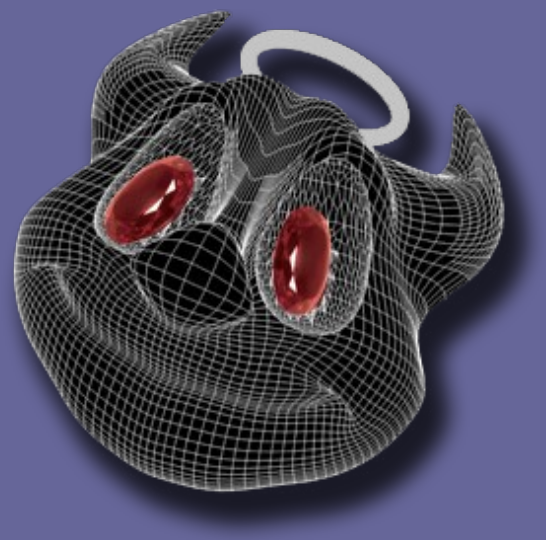




Visit http://locomotive.raaum.org to download Locomotive

Getting the Goods

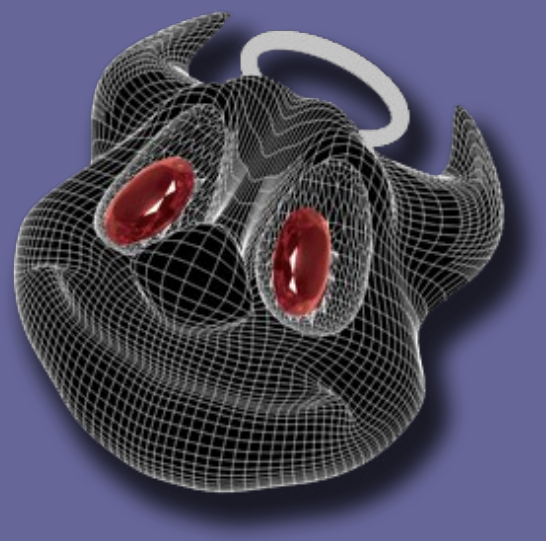
- FreeBSD 5.5 (or later)
  - ***pkg_add -r rubygem-rails***



Visit
http://wiki.rubyonrails.org/rails/pages/RailsOnFreeBSD
for more install details

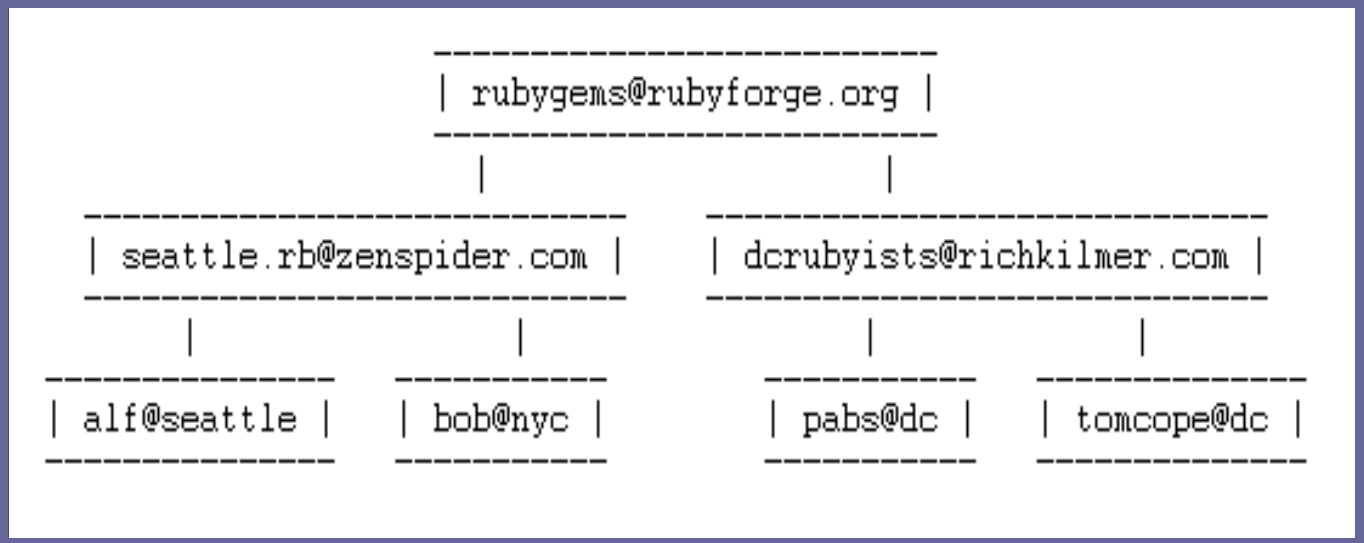
Romancing the Gems

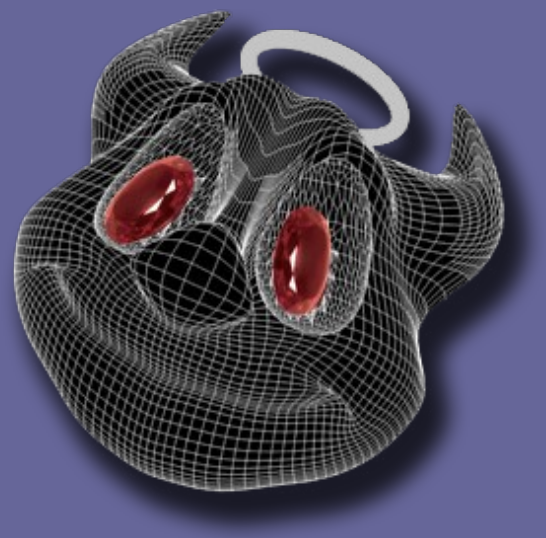
- Gems are retrieved from http://gems.rubyforge.org
  - `/usr/local/lib/ruby/gems/1.8/gems/sources-0.0.1/lib/sources.rb`
- No SSL `(confidentiality, integrity, and authenticity)`



Romancing the Gems

- RubyGems version 0.8.11 and later supports adding cryptographic signatures to gems.

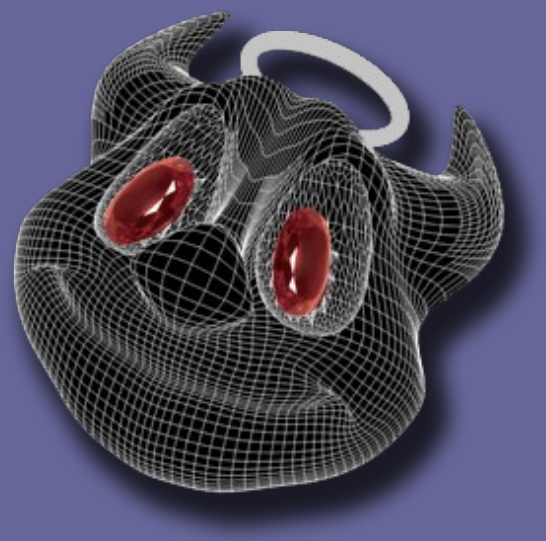




Visit
http://docs.rubygems.org/read/chapter/21
for more install details

Romancing the Gems

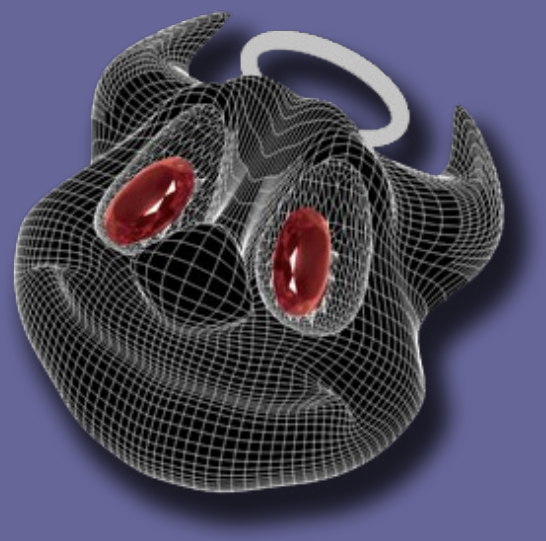
- Install the gems using the "*HighSecurity*" policy.
  - *gem install SomeGem-0.2.0.gem -P HighSecurity*



- gem must be signed
- signing cert must be valid
- signing cert must be trusted

Romancing the Gems

- Trusted Gem Certs
  - `/root/.gem/trust/cert-<SHA1(email)>.pem`

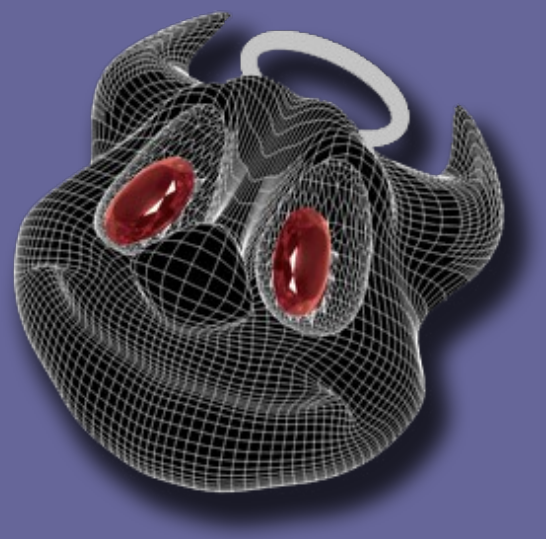


GEM Cert Directory is set to be readable by all. */root* should not be in order to restrict read access.

```
# ls -al /root/
total 36
drwx------   3 root  wheel  .
drwxr-xr-x  14 root  wheel  ..
drwxr-xr-x   3 root  wheel  .gem
```

These Go to Eleven

- Gems will typically keep older versions of packages
  - `/usr/local/lib/ruby/gems/1.8/gems`
  - Make sure to update Applications after package updates



Ruby on Rails | Screencasts | Download | Documentation | Weblog | Community | Source

RAILS

Search:

search is working now...

Job Board

The MITRE Corporation is looking for a Design Leader. See more on the Job

Rails 1.1.5: Mandatory security patch (and more)

Posted by David August 09, 2006 @ 05:30 PM

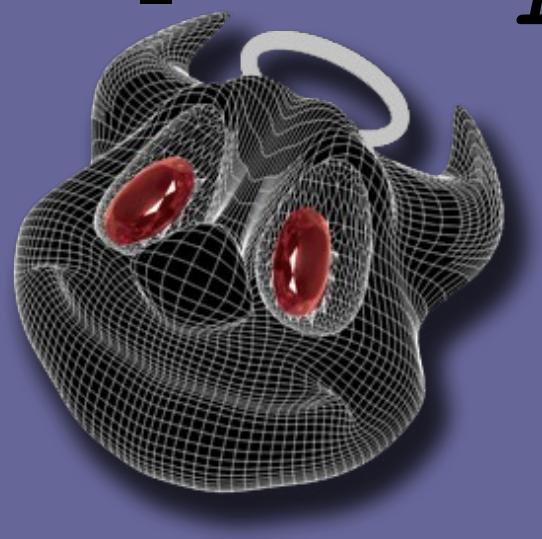
We're still hard at work on Rails 1.2, which features all the new dandy REST stuff and more, but a serious security concern has come to our attention that needed to be addressed sooner than the release of 1.2 would allow. So here's Rails 1.1.5!

This is a MANDATORY upgrade for anyone not running on a very recent edge (which isn't affected by this). If you have a public Rails site, you MUST upgrade to Rails 1.1.5. The security issue is severe and you do not want to be caught unpatched.

The issue is in fact of such a criticality that we're not going to dig into the specifics. No need to arm would-be assalients.

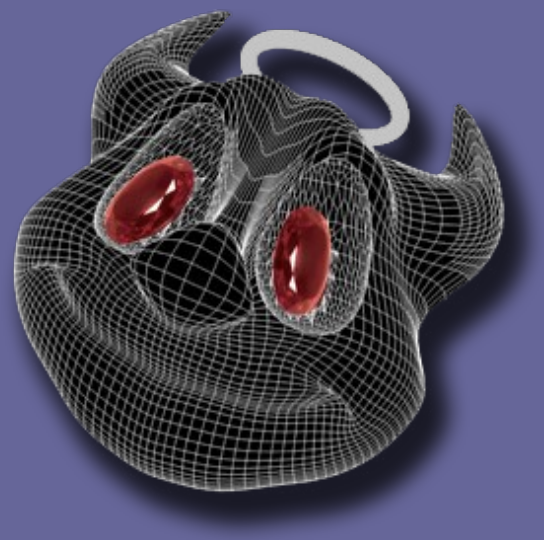
all float on ok

- When "Floating on Gems", check version number in *config/environment.rb*.
  - RAILS_GEM_VERSION = '1.1.6'
- When "Bound to Gems", (files in *vendor/rails*), make sure to rake and freeze your gems.
  - ***rake rails:freeze:gems***



No Soup For You

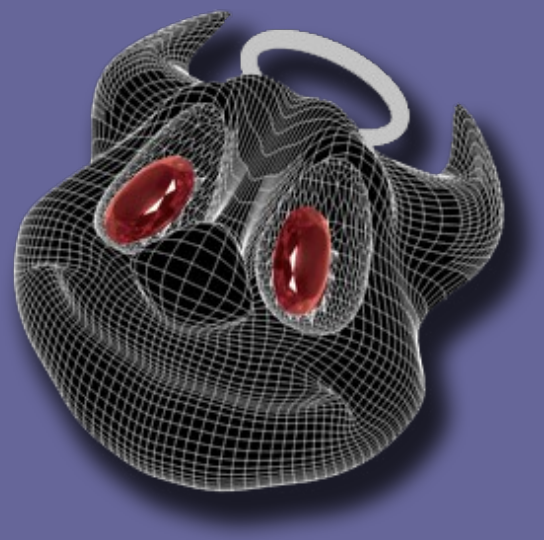
- Default Rails setup leaves weak file permissions.
- File Permissions
  - Read all to DB Config
  - Read/Write all to Log files



```
# Lock down key files
chown <owner>:<webserver> config/database.yml
chmod 640 config/database.yml
chown <owner>:<webserver> log/*.log
chmod 660 log/*.log
```

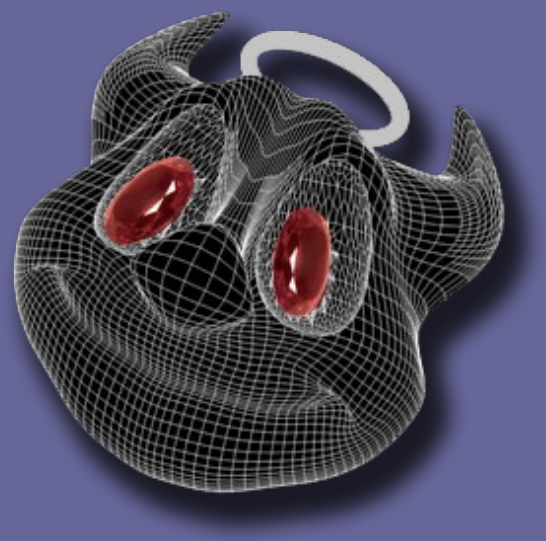
Run Away

- Run your web server with the least needed permissions.
  - sudo -u www ruby scripts/server



No Soup For You

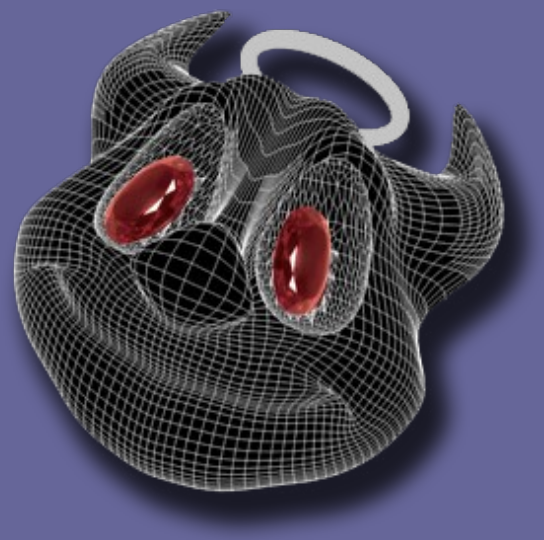
- Using defaults, Ruby will need to write to “tmp/sessions”.
- chown this directory to your ruby process. DO NOT CHMOD 777!



Things That Never Die

- Rails does not expire sessions on the server side.

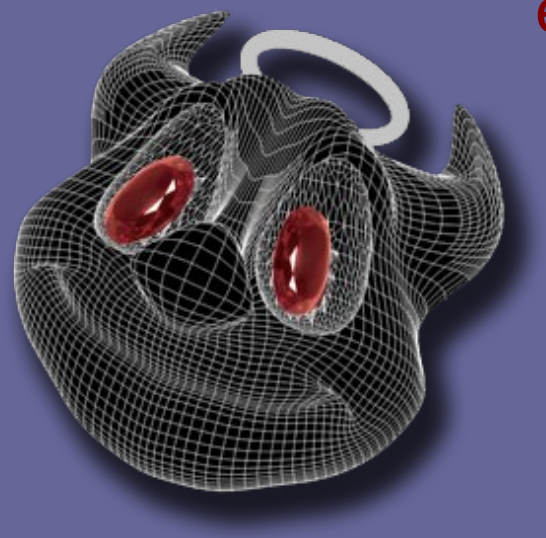
```
#NOTE: session_expire is a client side setting

class ApplicationController < ActionController::Base
  session :session_expires => 1.hour.from_now
end
```



Things That Never Die

- Rails does not expire sessions on the server side.

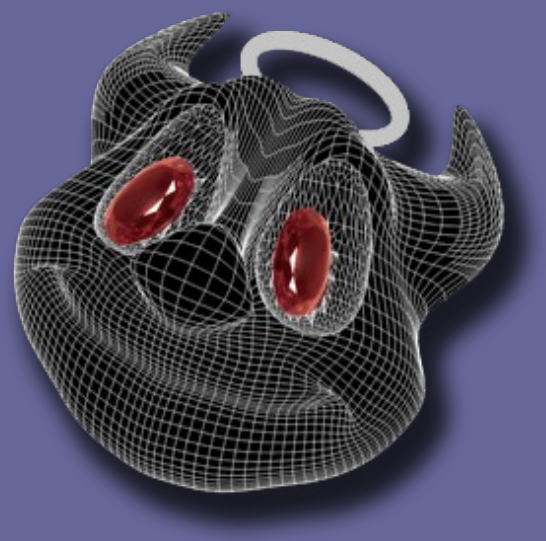
```
#Solution: Roll Your Own Clean Up
class SessionCleaner
  def self.remove_stale_sessions
    CGI::Session::ActiveRecordStore::Session.
      destroy_all( ['updated_on <?', 20.minutes.ago] )
  end
end
```



Code From Peter Donald at:
http://www.realityforge.org/articles/2006/03/01/

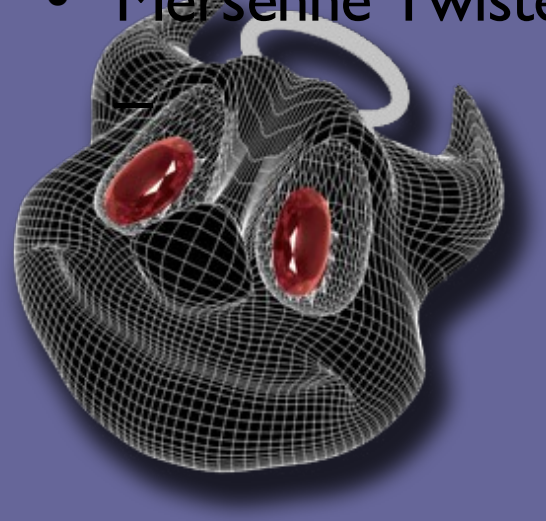
Soft Baked Cookies?

```
def create_new_id
 require 'digest/md5'
 md5 = Digest::MD5::new
 now = Time::now
 md5.update(now.to_s)
 md5.update(String(now.usec))
 md5.update(String(rand(0)))
 md5.update(String($$))
 md5.update('foobar')
 @new_session = true
 md5.hexdigest
end
private :create_new_id
```



Soft Baked Cookies?

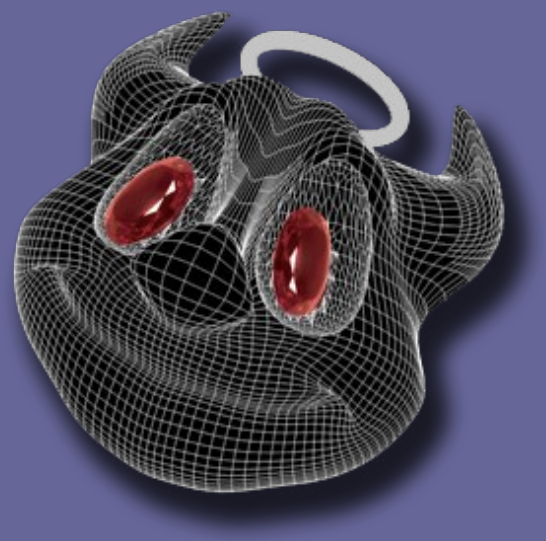
- Server Date and Time sent in response headers (except milliseconds would be unknown)
- PID is generally a limited range
- In 1995 Netscape's implementation of SSL (using time, PID, parent PID) was cracked
- "foobar" string adds no security if this is not changed
- Mersenne Twister algorithm alone is known to be cryptographically insecure



Ted Dziuba has more details at:
http://epsilondelta.net/2006/05/17/

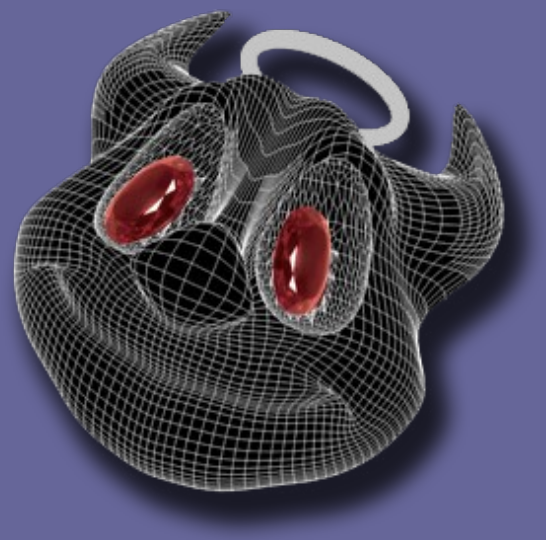
Hello Cleveland!

- Rocking Security Features
  - Protects against SQL Injection
  - Simple Validation and HTML Encoding Functions



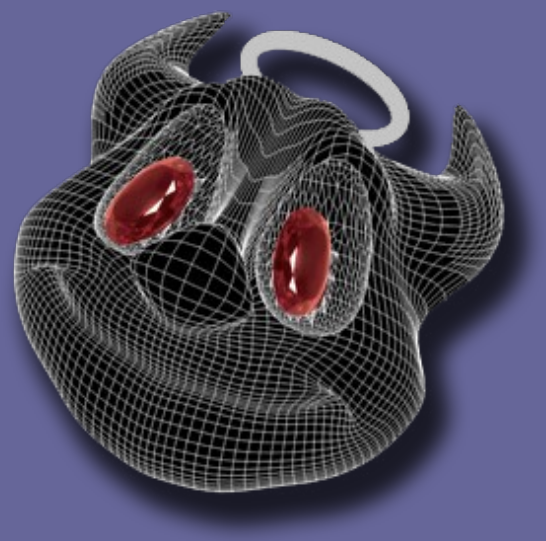
Dirty Injections

- SQL Injection can allow an attacker full access to your database or worse.
  - Example: Non-Checked input could lead to the following SQL Statement.
  - `SELECT * FROM users WHERE username = 'admin' AND password = '' OR 1=1 --';`



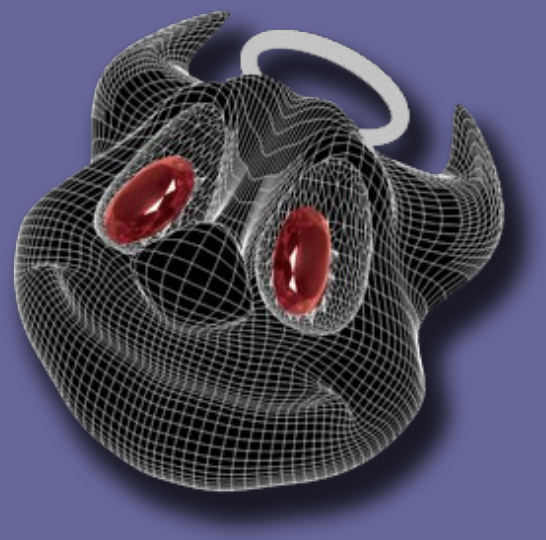
Escaped for Your Pleasure

- ActiveRecord with built in SQL Injection Protection
  - `book = Book.find(params[:id])`
  - `settings = Setting.find(:all, :conditions => [“uid=?”, user.id])`



Escaped for Your Pleasure

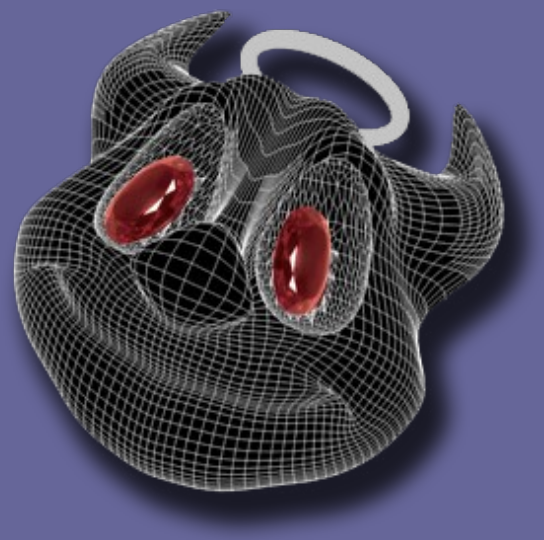
- **BUT DON'T DO THIS!!!**
  - `book = Book.find(:all`
    `:limit => #{session[:pref].id})`
  - `settings = Setting.find(:all,`
    `:conditions =>`
    `["username = '#{params[:id]}')`



This is a slide of what NOT to do.
Use Rails *bind variable* instead of #{...}
with ActiveRecords to avoid SQL Injection.

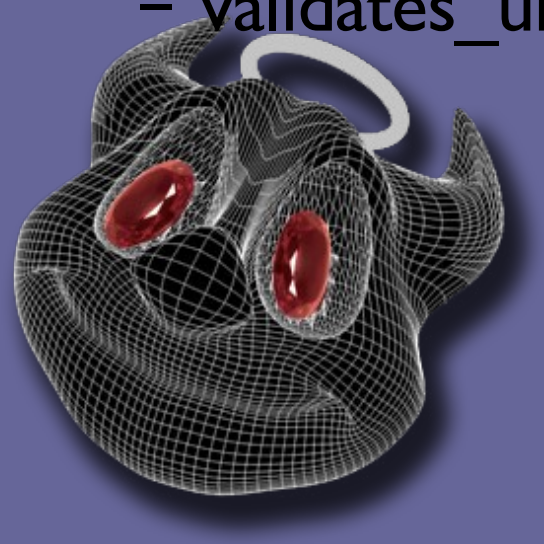
Escaped for Your Pleasure

- Data will be automatically truncated to match field length.
- Alternatively, it is easy to validate lengths of user input.
  - `validates_length_of :phone, :within => 5..16, :message => "Invalid Phone Number Length"`



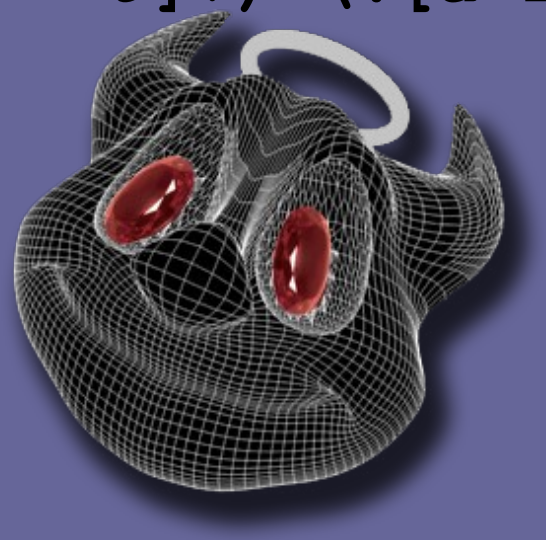
Validate Me

- Rails comes with a number of input validations built in.
  - validates_length_of
  - validates_presence_of
  - validates_format_of
  - validates_uniqueness_of

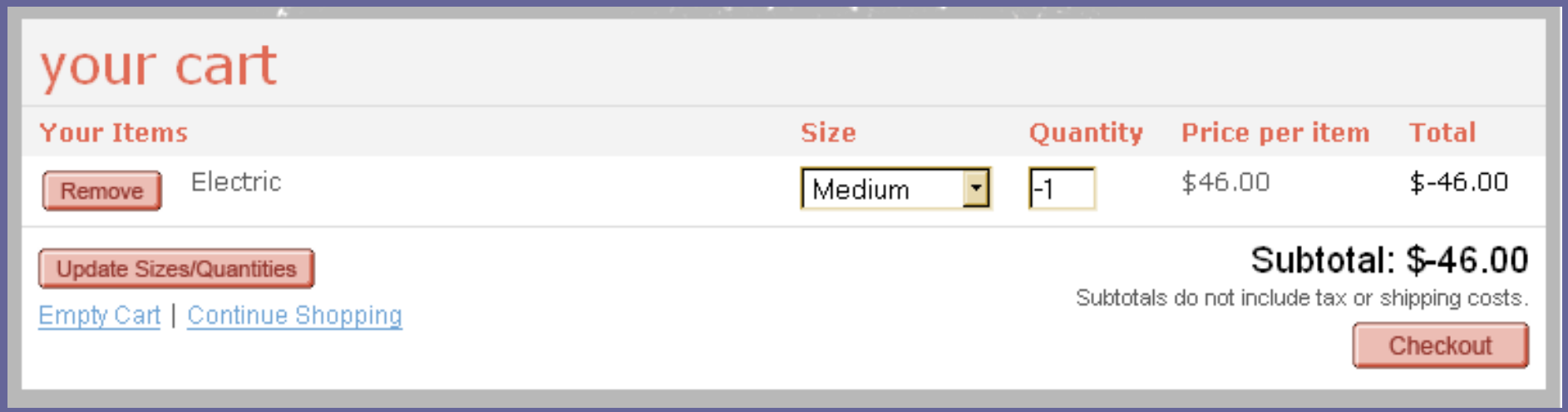


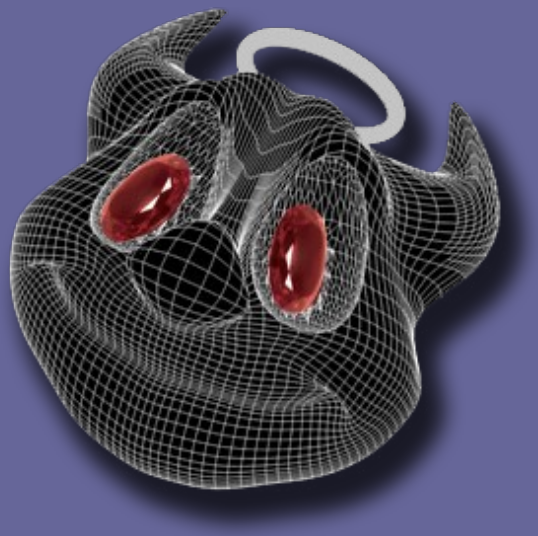
Validate Me

```
validates_length_of :phone, :within => 5..16
validates_format_of :phone, :with => /^[+\/\-
 () 0-9]+$/, :message => "Invalid Phone
 Number"

validates_format_of :url, :with => /^(http|
 https):\/\/[a-z0-9]+([\-\.]{1}[a-z0-
 9]+)*\.[a-z]{2,5}(([0-9]{1,5})?\/.*)?$/ix
```



Money Back Guarantee





Veni Vidi XSSdi

- Cross-Site Scripting is the web's new number one bad guy.


ZDNet Where Technology Means Business

NEWS | BLOGS | WHITE PAPERS

Page One | All News |

SECURITY

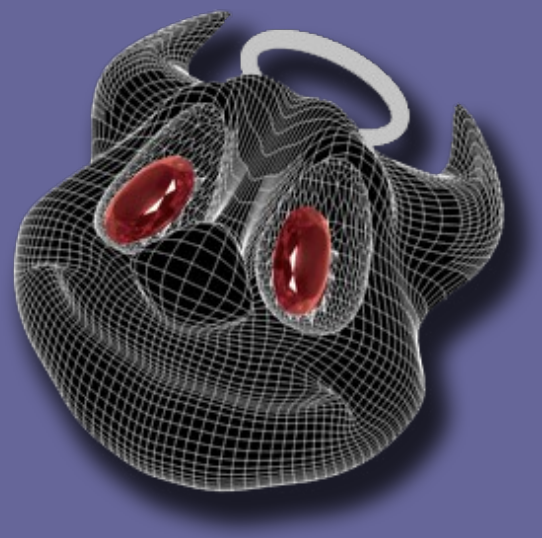
Samy opens new front in worm war

By Munir Kotadia, CNET News.com
Published on ZDNet News: October 17, 2005, 11:40 AM PT

TALKBACK ADD YOUR OPINION Forward in EMAIL Format for PRINT

ZDNet Tags: Security threats Viruses and worms Web browsers Security

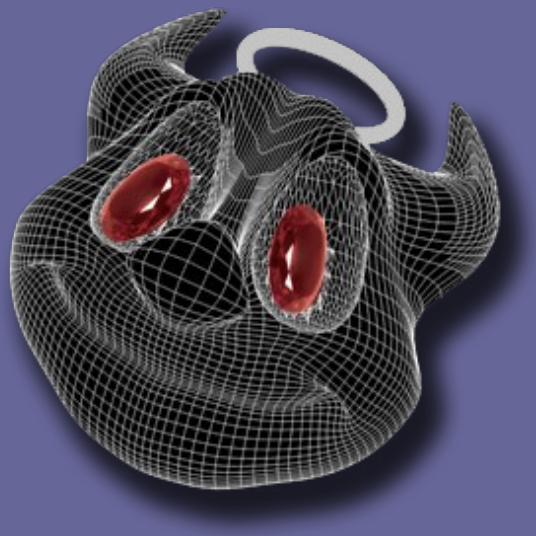
The newly discovered Samy worm is one of the first to exploit a cross-site scripting vulnerability, a technique security experts fear could be used to open a new front in attacks.

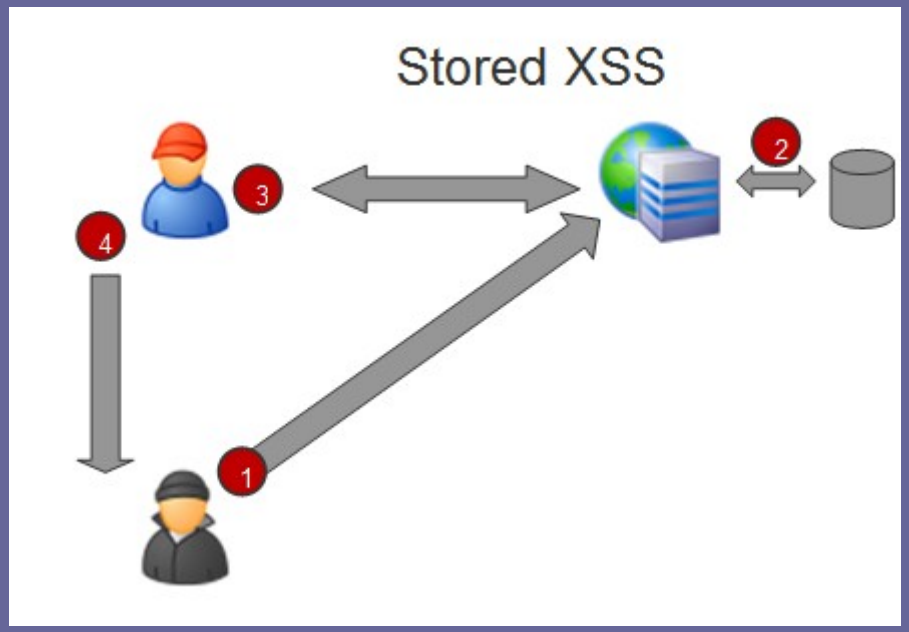


Probably the best site about XSS
http://ha.ckers.org

Veni Vidi XSSdi

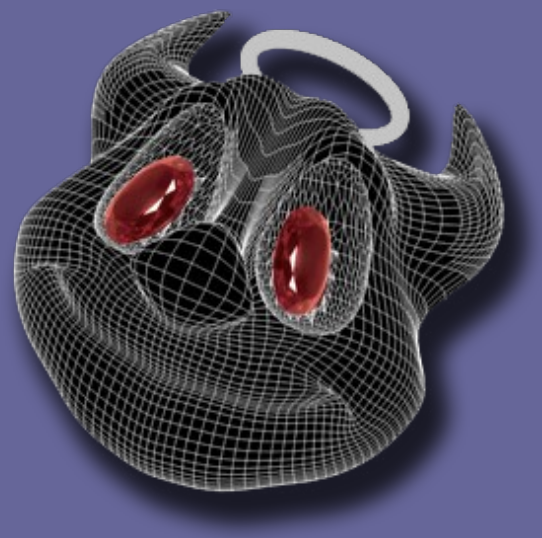
- Cross-Site Scripting (XSS) will take advantage of a flaw in a website to run unexpected code on a victim's web browser.





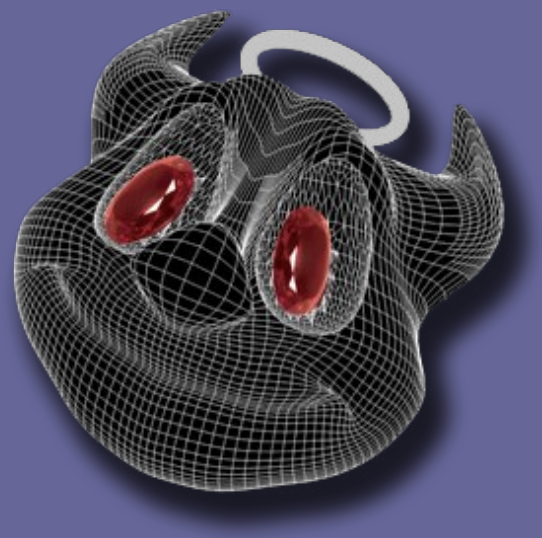
It's not just for breakfast any more...

- XSS is not limited to stealing cookie/session data.
  - Force a user to browse and submit data
  - Cause a browser to scan an internal network
  - Call vulnerable 3rd party controls
  - Attacker has endless possibilities for exploit payloads...



Veni Vidi XSSdi

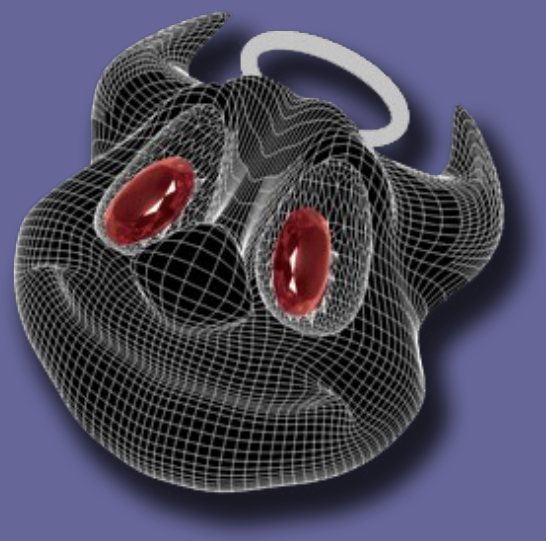
- Any user input, when redisplayed back to the same or other users, should be properly encoded to avoid allowing an attacker to write their own HTML/Javascript code.



Ruby to the Rexsscue

- Use the “h” html_escape method when writing user data back out.

```
<% for comment in @post.comments %>
        <%=h comment.body %>
<% end %>
```



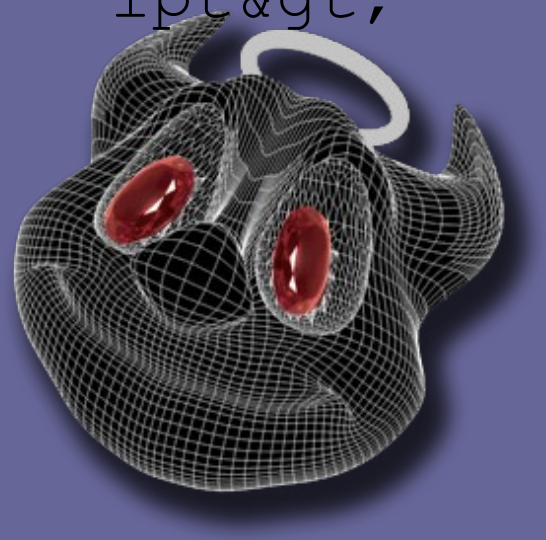
Ruby to the Rexsscue

Before HTML_Encoding:

```
    Watch me steal your cookie:
<script>alert(document.cookie)</script>
```

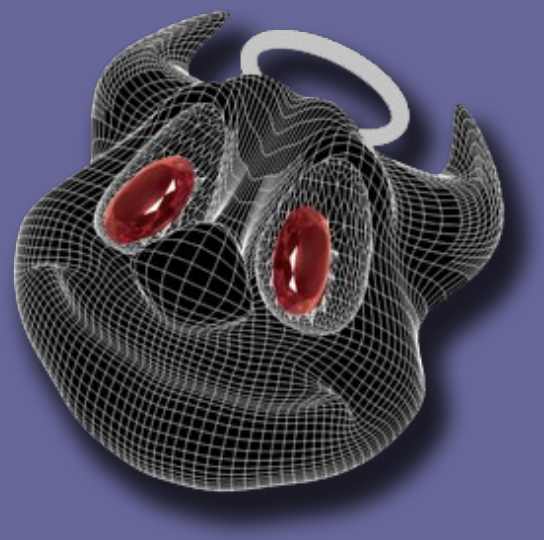
After HTML_Encoding:

```
    Watch me steal your cookie:
&lt;script&gt;alert(document.cookie)&lt;/scr
 ipt&gt;
```



Ruby to the Rexsscue

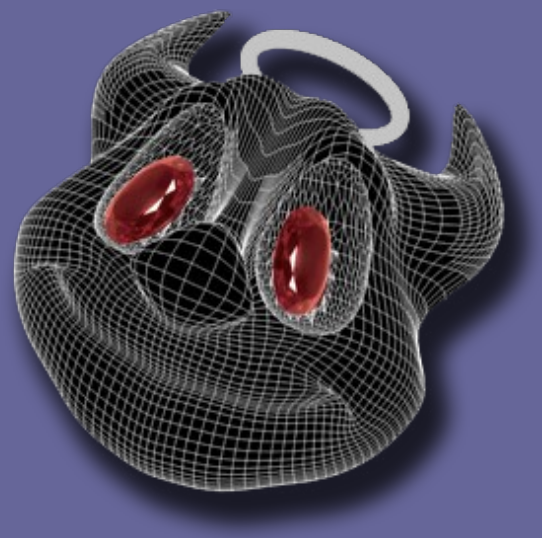
- Safe ERB
  - Plug-in that will ensure all strings written through rhtml templates are checked or escaped before written out. (Ruby's built in “$SAFE” can not be properly used with Rails.)



Shinya Kasatani wrote Safe ERB
which can be found at
http://wiki.rubyonrails.org/rails/pages/Safe+ERB

Checking the AJAX

- Rails has a built in check for XML Http Requests.
  - ***request.xhr?*** simply checks for header "X-Requested-With=XMLHttpRequest". This can be forged by an attacker.



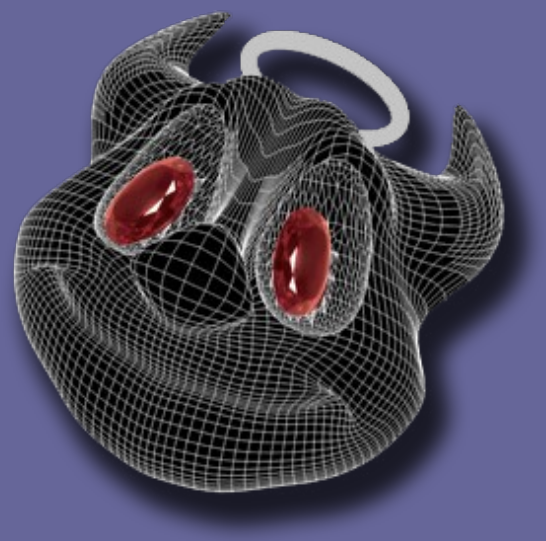
Would you like fries with that?

- Bulk database assignments, like “create” and “new”, can add data for any column in a table.

Table: users Schema: rubyblog_devel Comment:

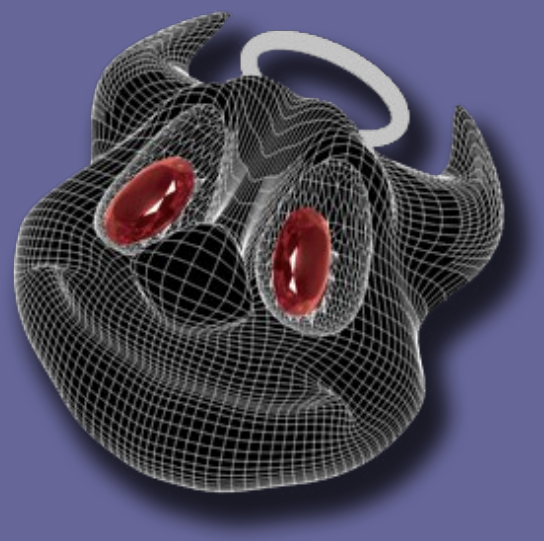
Columns/Indices | Table Options | Options

Name	Data Type	NOT NULL	AUTO INC	Flags	Default Value	Comments
id	INTEGER	✓	✓		NULL	
username	VARCHAR(255)				NULL	
password	VARCHAR(255)				NULL	
is_admin	TINYINT(1)				0	
approved	TINYINT(1)				0	



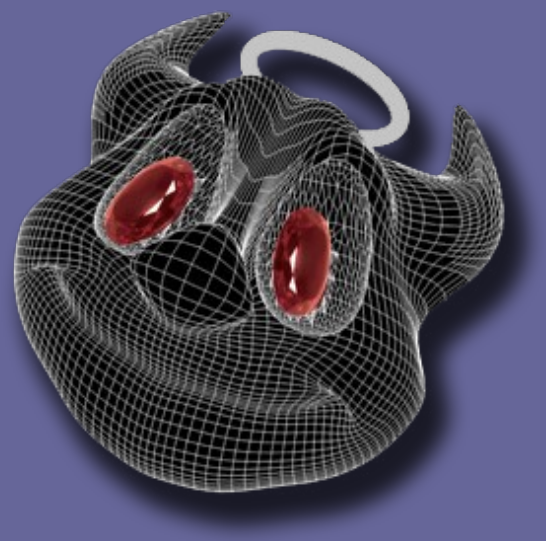
Would you like fries with that?

- Black List Column Exclusion
  - attr_protected :approved, :is_admin
- White List Column Exclusion
  - attr_accessible :username, :password



We're Safer

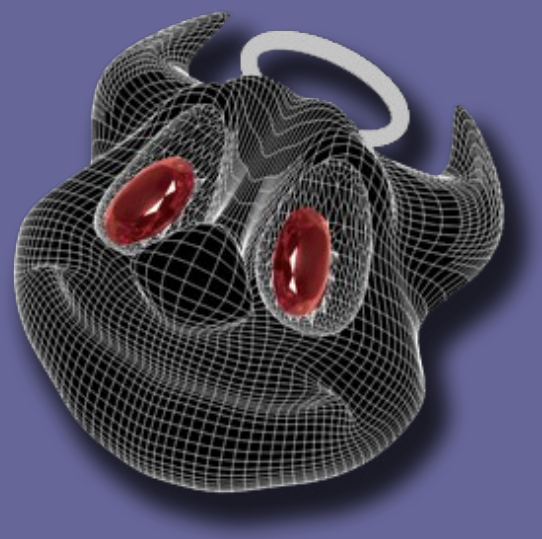
- "Contrary to popular belief, buffer overflow exploits do not occur in custom web applications. While technically possible, the truth is that they are just not seen in the real world." - Jeremiah Grossman (Mar 2006)
- Ruby is an interpreted, strongly typed dynamic language without direct memory access.
- Pure Ruby code will not contain buffer overflows.



Looking for that warm fuzzy feeling?
http://www.owasp.org/index.php/Buffer_Overflows
or
http://www.whitehatsec.com/articles/mythbusting_buffer_overflow.pdf

But not Safe

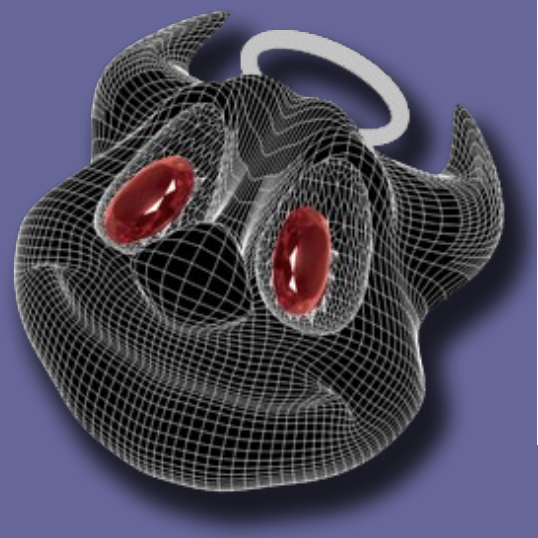
- A buffer overflow could exists in the interpreter.
- Using “RubyInline” you could embed C code in with Ruby.

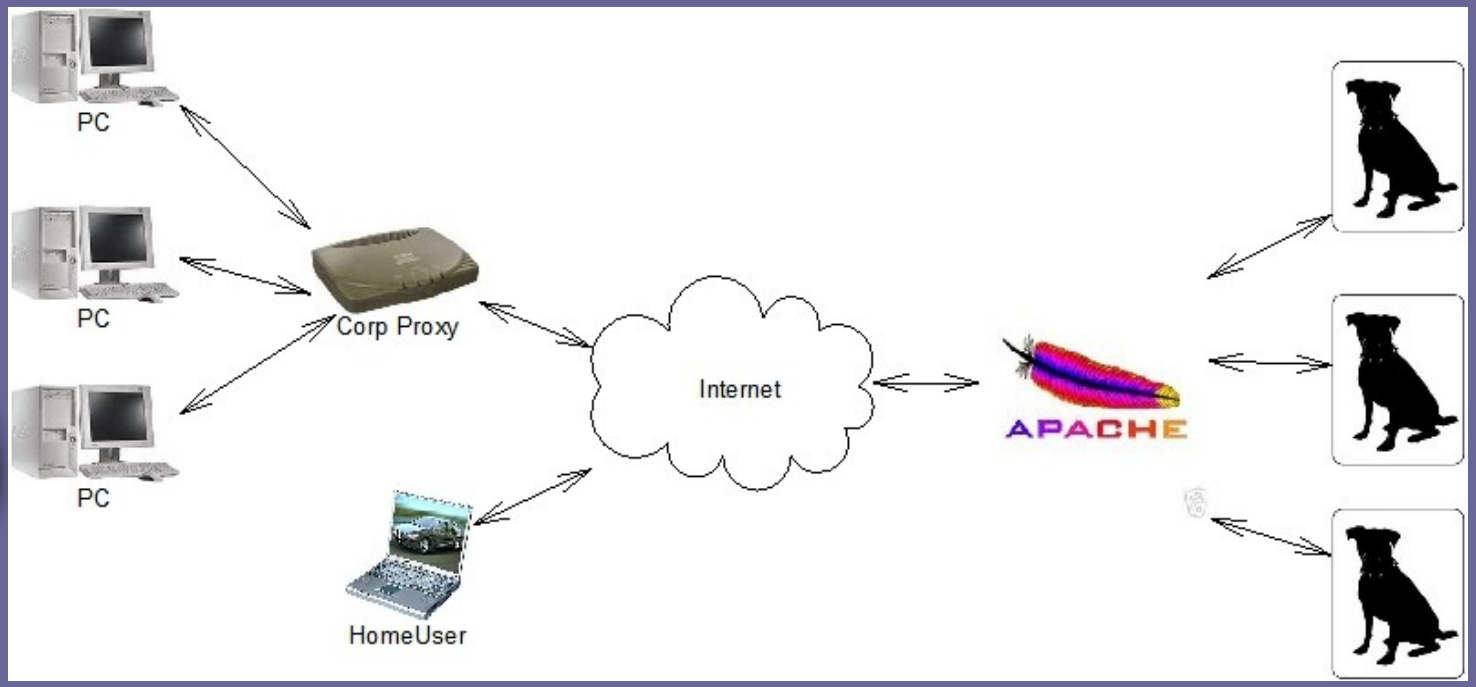


```
require 'rubygems'
require_gem 'RubyInline'

class << self
 inline do |builder|
    builder.c "
          int badcopy(char *input[]) {
          char buffer[10];
          strcpy(buffer, input[]);
          return 0;
     } "
 end
end
```

haeY haeY haeY

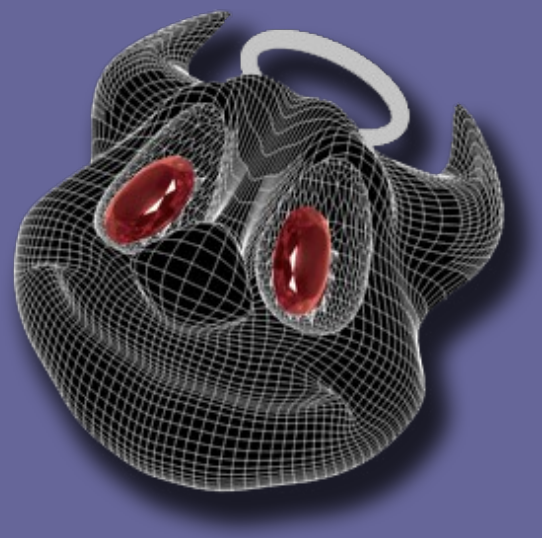
- Rails is single threaded. It can only handle one request at a time.
- **Reverse Proxy** to the Performance Rescue





haeY haeY haeY

- Response Splitting
  - Attacker will try to forge *response headers* to split a response and craft malicious content.
  - Validate filenames, cookies, and other data that may be used in *response headers* (particularly watch out for newline characters like %0D and %0A).



... for all the fish

- Validate Installs (if possible)
- Lockdown File Permissions (restrict reads)
- Validate User Input ("validates..." methods)
- Properly Escape Output (Safe ERB)
- Expire Sessions
- Use Bind Variables in ActiveRecords