

# What Matters in Application Security

Aaron J. Bedra  
EdgeCase, LLC.  
Central Ohio ISSA  
11/17/2006

# Who am I?

EdgeCase

software artisans

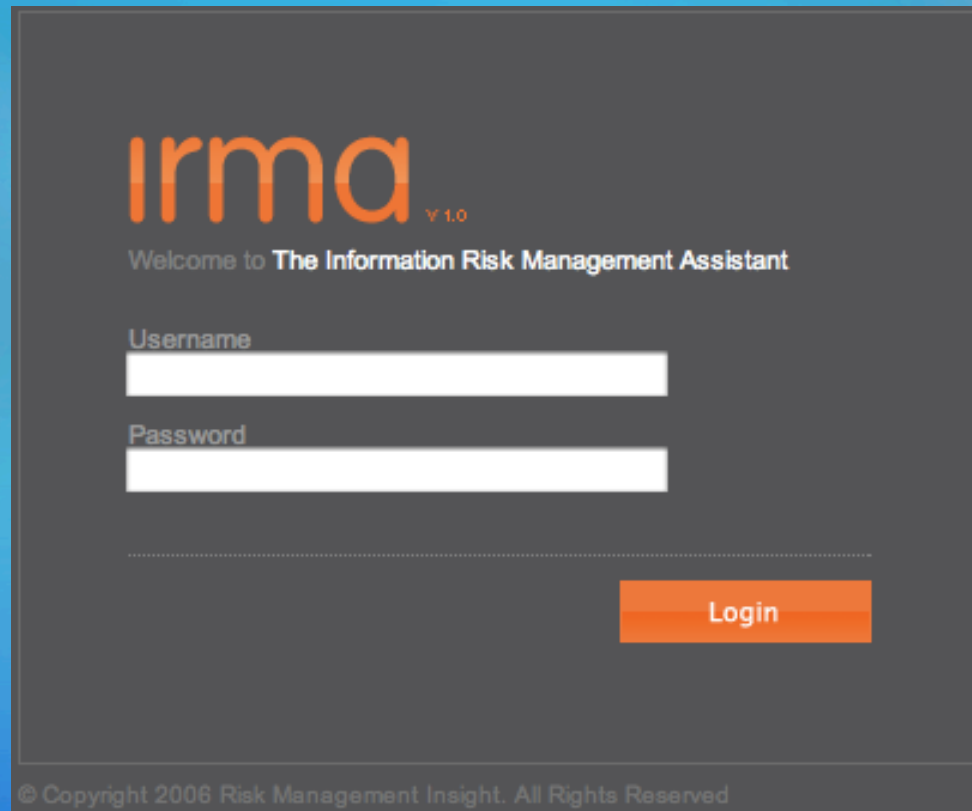


INSECURE . ORG



COLUMBUS RUBY BRIGADE

# The Goal...



The image shows a login screen for 'irma v1.0'. The interface is dark grey with orange text and buttons. It includes a logo, a welcome message, two input fields for 'Username' and 'Password', a 'Login' button, and a copyright notice at the bottom.

**irma** v 1.0

Welcome to **The Information Risk Management Assistant**

Username

Password

.....

Login

© Copyright 2006 Risk Management Insight. All Rights Reserved



```
def password=(pass)
  salt = [Array.new(6){rand(256).chr}.join].pack("m").chomp
  self.password_salt, self.password_hash = salt, Digest::SHA256.hexdigest(pass + salt)
end

def self.authenticate(username,password)
  user = User.find(:first, :conditions => ['username = ?', username])
  if user.blank? || Digest::SHA256.hexdigest(password + user.password_salt) != user.password_hash
    return nil
  end
  user
end
```

## Dashboard



### Create a New Risk Study

Start a new risk study from scratch.



### Search Risk Studies

Find an existing user to view or modify.



### Administration

Manage users, administrators, tags, and categories.



### Manage Your Account

Update your IRMA account info, including username and password.

## Welcome to IRMA

### The Information Risk Management Assistant

You are logged in as **IRMA Administrator**. Your last login was on November 12, 2006 at 02:00PM

Welcome to the Information Risk Management Assistant - IRMA! IRMA will help you develop quantitative risk analysis based on your inputs.

To get started, choose an action from the menu on your left. at any time, you can get back to this screen by choosing "Home" from the menu at the top of the web page.



http://experts.microsoft.fr/

# HACKED!

Hi Master (: Your System OwNed By Turkish Hackers!

redLine & rudeb0y & Ejder & The\_Bekir & SaCReDSeeR & ASH owNed you!

next target: microsoft.com

Ok, so tell me something I  
don't know...

# Fuzzing, the Wikiality...

Fuzz testing or fuzzing is a software testing technique. The basic idea is to attach the inputs of a program to a source of random data ("fuzz"). If the program fails (for example, by crashing, or by failing built-in code assertions), then there are defects to correct.

Fuzz testing was originated at the University of Wisconsin Madison in 1989 by Professor Barton Miller and the students in his graduate Advanced Operating Systems class. Their work can be found at <http://www.cs.wisc.edu/~bart/fuzz/>.

# Fuzzing API's

Scratch

Antiparser

fuzzball2

ip6sic

Audodafe

SMUDGE

Peach

Hydra

Fuzzy Sniff and Send

ISIC

Efuzz

ThreatX

SPIKE

PeachFuzzer

beSTORM

FuzzerCIRT

# Anatomy of a fuzz (surface attack)

- ❖ Collect a set of inputs
- ❖ Randomly insert invalid data into the inputs in an attempt to make the application produce errors.
- ❖ This creates a better set of tests and creates the ability to fix unforeseen problems and rule out faulty 3rd party software.

# Fuzzing Code

- ❖ A routine that arbitrarily runs through your applications source code and flips logic / changes constructs.
- ❖ Great for testing your applications test suite
- ❖ Every time the code fuzzer flips logic in your application a test should fail. (Ideal scenario)
- ❖ There are tools that help out with this process such as jester (java based), or heckle (ruby based).

# Applying fuzzing to business cases

- ❖ Combining a suite of fuzz testing you can build statistical analysis of quality between software releases, or provide actuarial information as justification for purchasing or not purchasing 3rd party software.

# Cross-site Scripting, the Wikiality...

Cross-site scripting (XSS) is a type of computer security vulnerability typically found in web applications which allow malicious web users to inject HTML or client-side script into the web pages viewed by other users. An exploited cross-site scripting vulnerability can be used by attackers to bypass access controls such as the same origin policy. Recently, vulnerabilities of this kind have been exploited to craft powerful phishing attacks and browser exploits.

# Breakdown

To be able to bookmark pages, search engines generally leave the entered variables in the URL address. In this case the URL would look like:

<http://test.searchengine.com/search.php?q=XSS%20>

Next we try to send the following query to the search engine:

```
<script type="text/javascript"> alert('This is an XSS Vulnerability') </script>
```

# Breakdown cont.

By submitting the query to search.php, it is encoded and the resulting URL would be something like:

<http://test.searchengine.com/search.php?q=%3Cscript%3Ealert%28%91This%20is%20an%20XSS%20Vulnerability%92%29%3C%2Fscript%3E>

Upon loading the results page, the test search engine would probably display no results for the search but it will display a JavaScript alert which was injected into the page by using the XSS vulnerability.

# How can I prevent this?

The best way to protect a web application from XSS attacks is ensure that your application performs validation of all headers, cookies, query strings, form fields, and hidden fields (i.e., all parameters) against a rigorous specification of what should be allowed. The validation should not attempt to identify active content and remove, filter, or sanitize it.

Make sure that none of your applications methods are publicly accessible unless they need to be. Exposing methods that directly accesses an information store leaves your application vulnerable to attack.

# So how do I apply this to what I am doing?

- ❖ Utilizing software tools to assist you in checking your applications for these types of problems, you can help detect and fix issues before attackers have a chance to gain unauthorized access to your data.
- ❖ Unfortunately this alone does not completely prevent exploitable bugs from surfacing in your application. The only way to ensure maximum effectiveness is to combine these tools with regular source code audits.

Test

# Source code auditing

- ❖ Source code audits should be performed at least once during every release cycle of your software.
- ❖ These audits should be performed by developers on other projects or by outside sources to make sure that new ideas are brought to light during the review process.
- ❖ These audits can prove to be invaluable to ensuring the success of an application release.

# Test Driven Development, the Wikiality...

Test-Driven Development (TDD) is a computer programming technique that involves repeatedly first writing a test case and then implementing only the code necessary to pass the test. Test-driven development gives rapid feedback. The technique began to receive publicity in the early 2000s as an aspect of Extreme Programming, but more recently is creating more general interest in its own right.

Practitioners emphasize that test-driven development is a method of designing software, not merely a method of testing. Along with other techniques, the concept can also be applied to the improvement and removal of software defects from legacy code that was not developed in this way.

# Testing Frameworks

- ❖ Almost every language available today has a testing framework.
- ❖ These frameworks provide you with all the tools necessary to perform source code testing throughout your application.
- ❖ Some of the more popular programming languages today not only have these tools but provide for automation of testing and nice visual feedback of test results. (JUnit and NUnit).

Yeah I've heard that before,  
we don't have time for that  
here in our organization...

# Managing the SDLC

- ❖ Two new ways of managing the Software Development Life Cycle have emerged since 2000.
- ❖ The first, Extreme Programming, evangelized by Kent Beck is geared towards the idea of programming in pairs for better productivity and dogmatically adopting the TDD philosophies.
- ❖ Shortly after the Agile method was presented as a continuation of the ideas of Extreme Programming focused on minimizing risk by developing software in short timeboxes called iterations.

Yes but the QA department already finds all the bugs in my software so why should I test first?

# Cost of Business Benefits

- ❖ If you are speaking in terms of risk management, you need to determine the business impact of releasing software that could be compromised.
- ❖ Writing tests first or adopting a test forward mentality not only helps your application but also let's you speak the rules of business with your companies business analysts so that your development team gets things right the first time.

# Ok, so when is enough enough?

- ❖ Combine your testing with code coverage tools. These tools run your tests and look at your applications source code to determine which pieces of your application have actually been tested by your developers tests.
- ❖ A safe level of code coverage in any application hovers in the realm of 90%.

# Conclusions

- ❖ Often people forget that applications have unbridled access to data that most of us deem critical.
- ❖ Testing should be done throughout the entire development process, starting with the development staff.
- ❖ Combining business analysts, QA, project management, and developers in a constant feedback situation reduces application flaws by orders of magnitude.

Questions?

# Contact Information

Email: [aaron@theedgecase.com](mailto:aaron@theedgecase.com)

Web: <http://theedgecase.com>

Skype: aaronbedra