



# The Evolution of Web Application Penetration Testing

Daniel Cuthbert, OWASP Testing Guide  
OWASP  
Daniel.Cuthbert@owasp.org

**OWASP  
AppSec  
DC**  
October 2005

Copyright © 2005 - The OWASP Foundation  
Permission is granted to copy, distribute and/or modify this document  
under the terms of the GNU Free Documentation License.

**The OWASP Foundation**  
<http://www.owasp.org/>

---

# Introduction

- Who am I?
- What's this talk really about?



---

# Quick Poll

## ■ Who here:

- ▶ Performs Web Application Penetration Tests (WAPT's)
- ▶ Has been doing it for less than 2 years?
- ▶ Has hired 3rd party consultancies to perform WAPT's
- ▶ Were actually satisfied with the work that was performed?



# Brief History Of Penetration Testing

- 1998 - Web App form field attacks
- 1998 - Encoding flaws
- 1999 - IIS / ASP flaws
- 2000 - SQL injection on ASP
- 2000 - More encoding flaws UNICODE/UTF-8
- 2001 - SQL injection on Oracle, MySQL, DB2 etc.
- 2001 - First attempt at attack & pen web proxies
- 2001 - Mainstream fuzzing of web-apps
- 2002 - Java app server flaws
- 2002 - MIME / chunked encoding (protocol level)
- 2003 - ASP.net hacking / Blind SQL injection
- 2003 - DotNetReflector
- 2004 - LDAP injection
- 2005 - Nada!



# Current Problems in WAPT

- Recent influx of non-security IT people joining IT Security
- Very few new IT security consultants come from a development background
- The level of penetration testing skill isn't as advanced as it used to be (Hacking Exposed Phenomenon)
- The reliance on automated tools is growing



---

# Changes In Software Development

- Vendors attitude changed from “it’s a bug” to “it’s a feature” to finally “it’s a security bug!”
- Web applications are no longer simple front-end’s to databases
- Developers are now expected (and required) to have an understanding of security techniques and vulnerabilities
- Customers are far more aware of security issues today than ever before



---

# Legislative Changes

- VISA's Cardholder Information Security Program (CISP) & MasterCard's Site Data Protection Program (SDP)
- Sarbanes-Oxley Act of 2002
- Basel II
- UK's Data Protection Act
- UK's Computer Misuse Act 1990



---

# Reality check

- Application deployments are highly time critical
- Application deployments are usually late
- Application development is often over-budget
- These pressure *can* lead developers to overlook security problems and controls



---

# Changes Required From Security Testers

- Understanding of Threat Modeling
- Risk Assessment
- Testing Methodologies
- Integrating Security Testing into the SDLC
- Performing “deep” security tests



---

# Threat Modeling

- Objectives of a threat modeling exercise
- Value of performing threat modeling
- DREAD
- STRIDE
- Example



---

## DREAD (Ranking Threats)

- Threats should be ranked using DREAD (Damage, Reproducibility, Exploitability, Affected Users, Discovery)
- Threats should be prioritized like any other software problem



# DREAD

	Rating	High (3)	Medium (2)	Low (1)
D	Damage potential	The attacker can subvert the security system; get full trust authorization; run as administrator; upload content.	Leaking sensitive information	Leaking trivial information
R	Reproducibility	The attack can be reproduced every time and does not require a timing window.	The attack can be reproduced, but only with a timing window and a particular race situation.	The attack is very difficult to reproduce, even with knowledge of the security hole.
E	Exploitability	A novice programmer could make the attack in a short time.	A skilled programmer could make the attack, then repeat the steps.	The attack requires an extremely skilled person and in-depth knowledge every time to exploit.
A	Affected users	All users, default configuration, key customers	Some users, non-default configuration	Very small percentage of users, obscure feature; affects anonymous users
D	Discoverability	Published information explains the attack. The vulnerability is found in the most commonly used feature and is very noticeable.	The vulnerability is in a seldom-used part of the product, and only a few users should come across it. It would take some thinking to see malicious use.	The bug is obscure, and it is unlikely that users will work out damage potential.



# DREAD

Threat	D	R	E	A	D	Total	Rating
Attacker obtains authentication credentials by monitoring the network.	3	3	2	2	2	12	High
SQL commands injected into application.	3	3	3	3	2	14	High



---

# STRIDE

- Spoofing
- Tampering
- Repudiation
- Information Disclosure
- Denial of Service
- Elevation of Privilege



---

# Risk Assessment

- Why should we perform a risk assessment?
- What's the difference between a risk assessment and a threat modeling exercise?



# Testing Methodologies

- The need for a defined testing methodology:
  - ▶ Consistency
  - ▶ Reproducibility
  - ▶ Quality control
- Should integrate with the organization's Security Standard for Web Applications
- Methodology should be detailed - no room for interpretation
- Threats should be prioritized like any other software problem



# Integrating Testing in the Dev Lifecycle

- Educate developers about security threats and common vulnerabilities
- Educate QA testers about how to perform basic security tests
- Work with management to ensure that testing is conducted throughout the project lifecycle
- Create a Security Standard for Web Applications that defines:
  - ▶ What security functions an application must implement
  - ▶ How these functions should be implemented



# Implementing Defense In-depth

- The value of Security Architecture Reviews
  - ▶ Identify risks at project inception - save \$\$\$
  - ▶ Define security functions early on
- Applications are continually expanding: Today a web app, tomorrow a web service and Thursday a mobile Java front-end
- Assessments should test the code in depth - not only through the web front-end
- Security tests can be embedded in Unit tests and Integration testing



# On the Horizon

- What are the new technologies being developed?
- New toys! - Improved Assessment tools
  - ▶ Improved automated tools - less false positives and negatives
  - ▶ Improved manual testing tools - e.g. E-or from [www.sensepost.com](http://www.sensepost.com)
  - ▶ Secure coding features integrated in IDE's
  - ▶ Security tests built into QA testing tools



# On the Horizon cont..

- Improved Methodologies
  - ▶ More clearly defined
  - ▶ Tailored to different classes of applications e.g. Banking, eCommerce, B2B etc.
- Security requirements specified in software contracts
  - ▶ Let the law and economics solve the problem!
- How will security be perceived?
  - ▶ More integrated, less animosity



---

# Questions?

