Global Information Assurance Certification Paper

Introduction to the OWASP Mutillidae II Web Pen-Test Training Environment

GIAC (GWAPT) Gold Certification

Author: Jeremy Druin, jdruin@gmail.com

Advisor: Chris Walker

Accepted: October 18th, 2013

Abstract

Web application penetration testing is composed of numerous skills which require "hands on" practice to learn. To prepare for certification exams, master concepts learned in training, and practice pen testing, a deliberately vulnerable web application is needed. While several excellent applications exist, very few provide many types of web application vulnerabilities in a single platform. In particular, having both traditional vulnerabilities plus vulnerable web services in the same platform is rare (Eston, Abraham, & Johnson, 2011). Additionally, features such as automated recovery, built-in hints, and varying levels of difficulty are not found within the same target framework.

The OWASP Mutillidae II Web Pen-Test Training Environment provides an environment to practice exploits against approximately forty documented vulnerabilities. Two vulnerabilities are exposed as web services. Mutillidae II delivers tutorials, supporting videos, and database reset functionality. The system is designed to assist students, exam candidates, and professionals in mastering web application security testing.

1. Introduction

Web application security has become increasingly important to organizations. In 2012, statistics on data breaches show nearly 80% of records taken in data breaches were due to defects in web applications (Verizon RISK Team et al., 2012). Training developers in secure coding and awareness are essential parts of a security program. (The Open Web Application Security Project (OWASP), 2009).

Vulnerable web applications provide a safe, legal target on which aid developers in understanding and appreciating the consequences of the vulnerabilities. Security training instructors can avoid having to create custom web applications since ready-made lab environments are available. Additionally they offer an "apples to apples" testing target for potential web application vulnerability scanners. This is useful to evaluate vendor products against each other and against a target with specific, disclosed vulnerabilities (David Shelly, 2010).

Excellent applications have been provided which offer features a web application training target might implement (Crenshaw, 2009). In order to cover multiple aspects of a security program, a web application should have the following characteristics.

- Present real web, application, web service and database vulnerabilities
- Provide vulnerabilities from multiple versions of OWASP Top Ten (OWASP Foundation, 2010) (OWASP Foundation, 2010), SANS Top 25 (SANS, 2011), and others.
- Require few steps to install
- Run on multiple operating systems
- Require little programming experience to operate
- Recover from attacks
- Includes hints and tutorials
- Run in multiple security levels
- Be free to use and open source

Mutillidae II[1] (Druin, 2011) is a free, open source, deliberately vulnerable web-application target for web-security training. For users who do not want to administer a webserver Mutillidae can be installed on Linux[2] and Windows[3] operating systems[4] using the following platforms.

- LAMP (Linux, Apache HTTP Server, MySQL, and PHP, Perl or Python) (Wikipedia, 2013)
- MAMP (Macintosh, Apache, MySQL and PHP) (MAMP, 2007)
- WAMP (Windows Apache MySQL PHP) (Bourdon, 2013)
- XAMMP (Cross-platform Apache MySQL PHP) (Seidler, 2013)

With dozens of vulnerabilities and hints to help the user, Mutillidae is an easy-to-use web hacking environment designed for labs, security enthusiast, classrooms, CTF, and vulnerability assessment tool targets. Recent enhancements include an augmented version of Damn Vulnerable Web Services (DVWS) (Johnson, Damn Vulnerable Web Services, 2011).

Mutillidae has been employed for several use-cases:

- Graduate security courses[5] (Jeffrey L. Hieb, 2013)
- Corporate training[6]
- Web pen-testing training courses (SANS™ Institute, 2013) (SANS™ Institute, 2013)
- An "assess the assessor" target for vulnerability assessment software (Druin, 2011)
- Target for Web Application Firewall (WAF) testing (McHenry, 2013)
- Demonstration platform in presentations for the Open Web Applications Security Project (OWASP) Chapters (Blankenship, Hartmann, Koenig, Baso, & Sullivan, 2013) and Information

[1] Mutillidae are sometimes known as "wingless wasps". Mutillidae, a **w**eb **a**pplication **s**ecurity **p**roject (wasp) built deliberately without security can be thought as having its wings clipped. Credit to Adrian "irongeek" Crenshaw for the name.
[2] Distributions verified include Backtrack 5 R3, Kali, and Ubuntu
[3] Distributions verified include XP, Vista, and 7
[4] Mac OSX "Snow Leopard" is verified to run the project successfully on MAMP (Macintosh, Apache, MySQL and PHP).
[5] University of Louisville Speed Scientific School
[6] Multinational logistics company

Systems Security Association (ISSA) (Druin, 2012 KY ISSA Conference: Introduction to OWASP Mutillidae II Web Pen Testing Environment, 2012)

2. Project History

Mutillidae began as version 1.x on irongeek.com (Crenshaw, Irongeek.com, 2013) as a web pen testing target for vulnerabilities in the OWASP Top 10. Version 1.x was created by Adrian "irongeek" Crenshaw (Crenshaw, Irongeek.com, 2009). Although version 1.x is no longer supported or actively developed, it is still available today (Crenshaw, Mutillidae-classic (Mutillidae versions 1.x), 2012). Rather than augment version 1.x, a new project, Mutillidae II, was written using object oriented architecture. The available functionality was ported. An expanded feature set and new classes of vulnerabilities were added.

The redesign and development of Mutillidae 2.x was completed by Jeremy Druin who migrated both projects from irongeek.com to SourceForge (Druin, SourceForge: Mutillidae 2.x Download (Current Version), 2013). This aided in scaling distribution and consolidating documentation. Mutillidae 2.x recalls its roots by re-implementing the OWASP 2010 Top 10 vulnerabilities and redesigning Crenshaw's "user hint system".

3. Advantages

Mutillidae has several advantages that make the system attractive for independent study and instructor-led training. These may be summarized as vulnerabilities/challenges, built-in user assistance, ease of installation, security levels, and automated set-up/recovery.

3.1. Vulnerabilities and Challenges Provided

The project delivers vulnerabilities, skill-challenges, and capture-the-flag support in an attempt to create a comprehensive web security training environment.

3.1.1. Vulnerabilities

Mutillidae implements a large variety of vulnerabilities with every category represented from the OWASP Top Ten 2007 (OWASP Foundation, 2010) and OWASP Top Ten 2010 (OWASP Foundation, 2010). Additionally several weaknesses from the SANS TOP 25 Most Dangerous Software Errors (SANS, 2011) are included[7]. Currently there are 43 types of vulnerabilities put into operation across multiple perspectives (Druin, Documentation: Listing of Vulnerabilities in Mutillidae 2.x, 2013). For critical categories such as cross site scripting (KirstenS, 2011) and SQL injection (KirstenS, SQL Injection, 2013), multiple contexts have been implemented for each. Types of Cross-site scripting vulnerabilities provided include HTML, JavaScript, and JSON injection[8]. Pages with SQL injection allow data extraction, upload of shells, and authentication bypass[9]. A listing of vulnerabilities organized by URL is available.[10] (Druin, Documentation: Listing of Vulnerabilities in Mutillidae 2.x, 2013).

In contrast to some deliberately vulnerable web applications, Mutillidae contains "live" vulnerabilities. The user is not expected to enter statements which the platform matches against a list of correct answers. Although known solutions exist for each page, there is no particular solution required to achieve exploitation.

[7] Specifically vulnerabilities related to web applications. The SANS TOP 25 Most Dangerous Software Errors additionally contains vulnerabilities from other types of "non-web" applications.
[8] For a complete list see Appendix E: Types of Cross-site Scripting Vulnerabilities Implemented
[9] For a complete list see Appendix F: Types of SQL Injection Vulnerabilities Implemented
[10] For a complete listing of vulnerabilities in Mutillidae, refer to **Appendix A**.

For safety and legal reasons, ensure Mutillidae is only accessible via local host or over a closed virtual network and only to the intended audience[11].

3.1.2. Skill-Challenges

Several skill-challenges have been included in the latest edition[12]. These may be helpful to explore hacking scenarios, use the platform for demonstrations, and prepare for security certification exams. A list of challenges currently implemented is listed in **Appendix B**[13].

3.1.2.1. Privilege Escalation

To provide a web cryptography challenge, the "User Privileges" page[14] has a CBC bit flipping vulnerability. This challenge provides the user the ability to escalate privileges by attacking the initialization vector (Druin, Mutillidae: Introduction to CBC bit flipping attack , 2012). Privilege escalation is also possible by altering the value of "cookies".

3.1.2.2. User-Agent Impersonation

In a simulation of "captive portals", User-agent impersonation allows the user to spoof a Safari browser running on Apple iPad. By controlling the User-Agent string and custom settings, the user can appear as the target device and be accepted by the captive portal page[15].

3.1.2.3. Authentication Bypass

Multiple methods of Authentication Bypass are possible

- Session hijacking and cookie tampering (Druin, Mutillidae: Bypass Authentication via Authentication Token Manipulation, 2012)

[11] Mutillidae includes a default .htaccess file restricted to 127.0.0.1 (localhost) and 192.168.*.* in an effort to provide some default security but running the project on a production or networked host is strenuously discouraged.
[12] Version 2.5.12 as of this writing
[13] An updated list is maintained at http://sourceforge.net/projects/mutillidae/files/documentation/listing-of-vulnerabilities-in-mutillidae.txt/download
[14] By default, when installed on localhost http://localhost/mutillidae/index.php?page=view-user-privilege-level.php
[15] http://localhost/mutillidae/index.php?page=user-agent-impersonation.php

- SQL injection can be employed to become the administrative user or a user of the attackers choosing (**Figure 1**) (Druin, Mutillidae: Bypass Authentication using SQL Injection, 2012).
- Administrative pages may also be reached by brute forcing the page name[16] (Druin, Mutillidae: Brute Force Page Names using Burp-Suite Intruder, 2012). Information disclosed in robot.txt[17] and directory browsing assists the user. (Druin, Mutillidae: Manual Directory Browsing to reveal Easter Egg File , 2012).

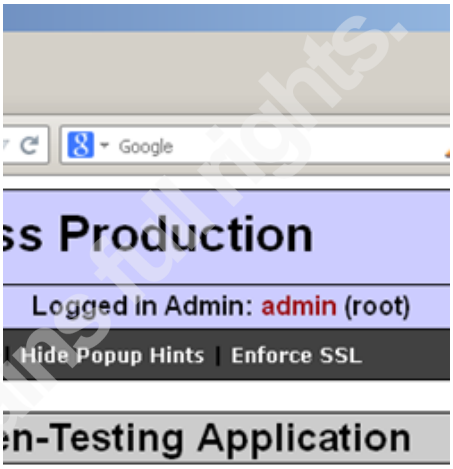

Figure 1: Authentication bypassed to gain administrative access

3.1.2.4. SSL Striping

The system includes an "Enforce SSL" feature which redirects HTTP requests to an HTTPS (SSL) connection (Druin, Mutillidae: Using Ettercap and SSLstrip to Capture Credentials , 2012). This gives an opportunity to perform "machine in the middle" (MITM) attacks with SSL Stripping (Marlinspike, 2012) to downgrade the target connection. The web browsing session proceeds in clear-text through the attacking host.

3.1.3. "Capture the flag" support

In addition to the variety of vulnerabilities implemented, Mutillidae provides specialized features to support "capture the flag" competitions and practice using popular web application security tools.

3.1.3.1. Automated Data Capture

By forced redirection, browser "hooking", or social engineering a user can be forced to visit the capture-data.php page. The user's cookies, all request parameters (GET and POST), client IP address and user-agent string will automatically be parsed into a database table[18] and written to a text file[19]. To support the attacker, the captured-data.php page organizes captured data into time-stamped records shown on a grid (**Figure 2**).

[16] http://localhost/mutillidae/index.php?page=secret-administrative-pages.php
[17] http://localhost/mutillidae/index.php?page=robots-txt.php
[18] Assuming default database, the table is nowasp.captured-data
[19] <web root>/mutillidae/captured-data.txt

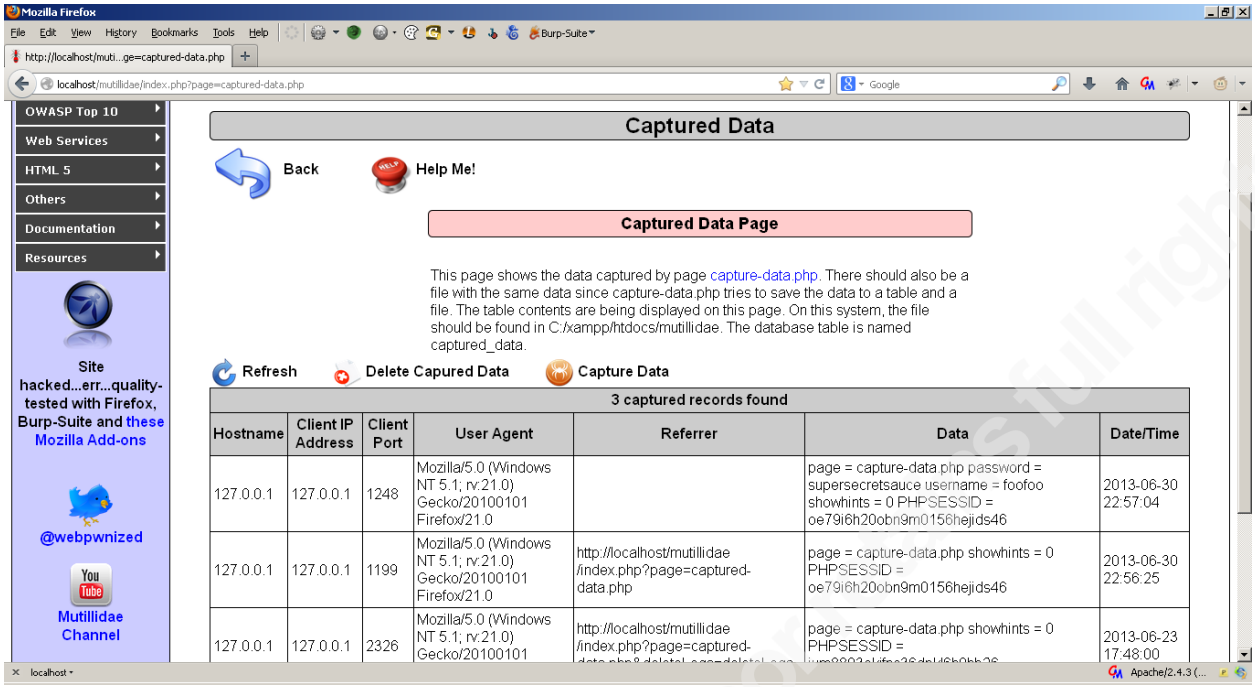

Figure 2: Captured Data Page Showing 3 Records

3.1.3.2. Vulnerabilities Geared Towards Attack Tools

Additionally, certain pages contain vulnerabilities crafted to respond to tools commonly used in labs, classes, and competitions[20]. Cross-site scripting is implemented as persistent vulnerabilities to allow the user to practice writing the exploit and lay traps such as redirection scripts, Beef Exploitation Framework (BeEF) hooks (BeEF Project), and scripts which use Asynchronous JavaScript and XML (AJAX) "background requests" (Refsnes Data, 2013) to avoid detection.

The following attack tools and web pen testing distributions are supported:

- sqlmap Automatic SQL injection and database takeover tool (Stampar & Damele): SQL injection opportunities that do not require prefixes and suffixes are provided (Druin, Basics of using sqlmap - ISSA KY Workshop - February 2013, 2013).
- Nikto 2 web server scanner (Sullo & Lodge, 2010)
- Burp-Suite security testing platform (PortSwigger Ltd., 2013)
- Web Application Attack and Audit Framework (w3af) (Riancho, 2013)
- NetSparker Community web application security scanner (Mavituna Security Ltd.)

[20] The user may choose to exploit these vulnerabilities without the toolsets

- Cenzic Hailstorm web application vulnerability scanner (Cenzic, Inc.)
- sslstrip HTTPS stripping attack tool (Marlinspike, 2012)
- hydra network logon cracker (van Hauser, 2013)
- OWASP DirBuster (Fisher, 2009)
- Samurai WTF (Johnson, Searle, & DiMaggio, Samurai Web Testing Framework, 2008)
- Rapid 7 Metasploitable 2 (Moore, 2012)
- Offensive Security Kali Linux penetration testing distribution (Offensive Security, 2013)

3.2. Help

The project contains 2 types of built-in user assistance on each page of the site plus external assistance.

- Dynamic help system
- Bubble hints
- Easter egg file
- Video tutorials

3.2.1. Built-in User Assistance

A hints system offers two levels of assistance. By default no hints are displayed[21]. Clicking the "Toggle Hints" button on the menu bar (**Figure 3**) activates level-1 hints. Upon clicking again, hints level-2 is actuated. Level-1 hints provide dynamically generated boxes which can be opened to provide information on the vulnerabilities in the page (**Figure 3**). Sections included are general description, discovery, and exploitation. Level-2 hints provide full tutorials on the subject matter.

See **Appendix G** for a list of topics covered by the level-1 hint system.

[21] The system defaults to hint-level 0. Hint levels 1 and 2 provide increasing amounts of assistance.

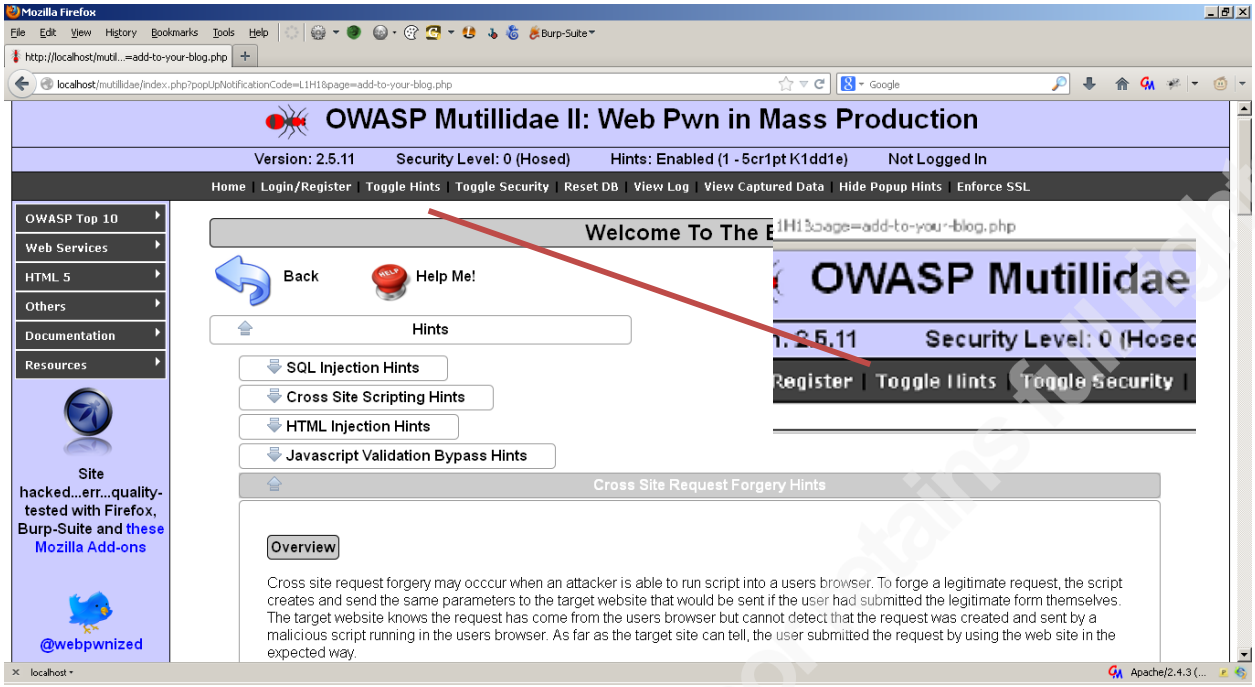

Figure 3: Clicking "Toggle Hints" activates embedded tutorials for the user. In this screenshot, Level 1 hints have been activated. The Cross Site Request Forgery hint is open for viewing.

Besides the written hints, the system will display pop-up "bubbles" when the user hovers[22] over vulnerable areas on the page. The bubbles appear in all hint levels; however, the text displayed becomes more revealing as the hint level increases. Bubble hints can be disabled[23]. The system will hide both bubble hints and dynamic help sections in the highest security level.

An Easter egg file "hidden" in the documentation[24] directory contains several hundred lines of exploit code verified to work on various pages[25]. Besides providing "known-good" exploits, the scripts can act as stubs to write injections.

3.2.2. Video Tutorial Support

Video tutorials[26] are available (Druin, webpwnized's channel, 2013) demonstrating discovery, testing, and exploiting vulnerabilities. New videos are added periodically and there are approximately 80 videos at the time of this writing (Druin, Query Results for Mutillidae on webpwnized channel, 2013).

[22] This feature requires JavaScript to be enabled since bubble hints are activated on mouse-over.
[23] The Show/Hide Bubble Hint toggle is located on the main menu bar (Figure 3)
[24] <webroot>/documentation
[25] Mutillidae-Test-Scripts.txt

Both YouTube and SourceForge allow "subscriptions" which can send an email when material is posted (SourceForge, 2013). If a single location to follow both areas is desired, announcements for both the project and videos are tweeted from a Twitter account (Druin, @webpwnized, 2013).

For a listing of videos available, see **Appendix D.**

3.3. Other Features

Mutillidae II implements automated system recovery to assist users in returning the backend database to the default state. Additionally, the user can switch between 3 security levels which range from completely vulnerable to secure.

3.3.1. Automated System Recovery

When the "Reset Database" button is clicked[27], the system will drop the backend database then rebuild the database, tables, and pre-configured target data[28]. This restores the system to its original state with the exception of the current security level and "help" level (**Figure 4**). These two items are stored in the session and remain intact until the user closes the browser.

[26] The primary channel is the "webpwnized" YouTube channel (http://www.youtube.com/user/webpwnized/videos). Irongeek.com mirrors most videos (Crenshaw, Web Application Pen-testing Tutorials With Mutillidae, 2013).
[27] The database reset button is located on the main menu bar (Figure 3)
[28] A listing of tables is provided in **Appendix B**

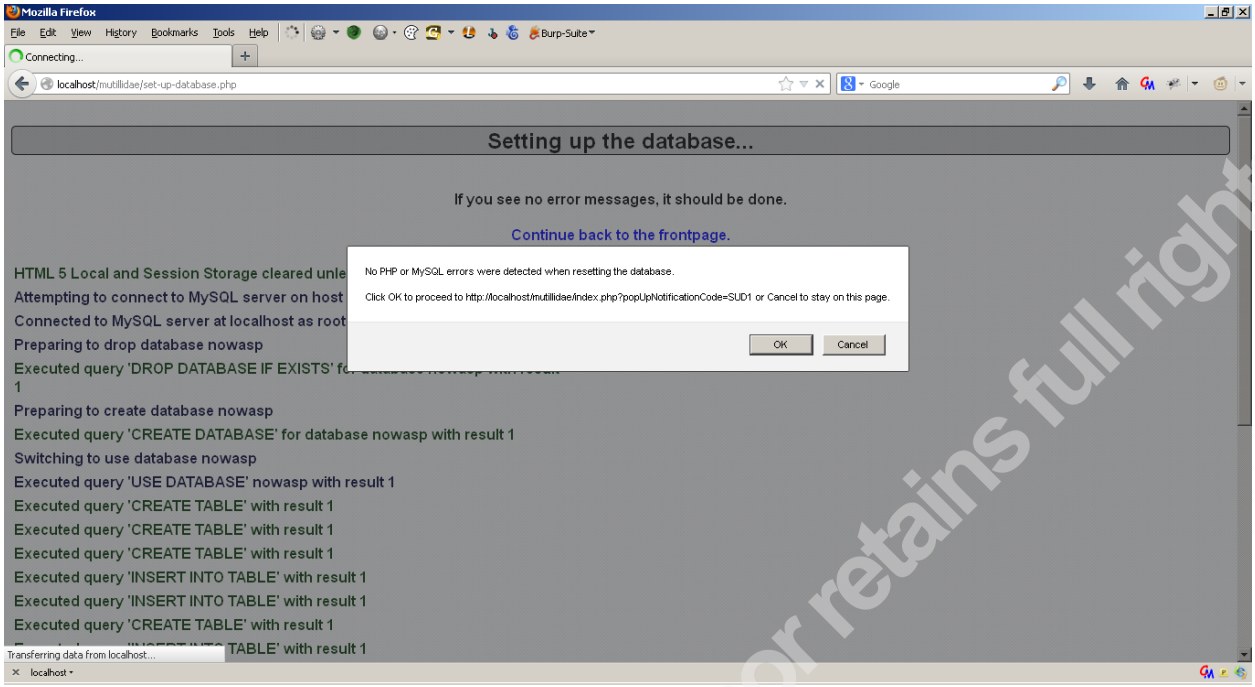

Figure 4: Clicking the "Reset DB" button causes the system to rebuilt the database effectively setting up the project or resetting the project to a known-good state

3.3.1.1. Automated System Recovery

The system restore feature is useful anytime a fresh start is desired but was created with two primary use-cases in mind.

Performing labs will change the system endangering the success of the next exercise. The restore feature ensures each lab starts from the same “clean” state.

Users may successfully exploit the system. “Persistent” malicious injections alter the system resulting in instability or annoyances. (Consider a series of popup boxes injected via cross site scripting vulnerabilities). The restore feature will drop persistent injections.

3.3.2. Security Levels

By default, Mutillidae runs insecure code but the system contains three code bases. The “security level” determines which application code executes when the user visits a page. The user can escalate from the

most insecure mode to fully secure[29]. Additionally, there is an intermediate security level[30] similar to the default mode except that JavaScript validation is activated in several pages. In this mode, the user is expected to bypass the validation[31].

The source code, including secure and insecure modes for each page, is stored in the same PHP file for easy comparison[32]. At the top of each page, variables are set which drive the page behavior.

3.4. Ease of Installation / Standard Architecture

Mutillidae may be installed on Linux and Windows hosts via SVN, Git or manually. It is available pre-installed on a few Linux "live" distributions. Installation steps are discussed later.

By running on Apache, MySQL, PHP (AMP) stacks the project will work well on both Linux and Windows hosts. Using a three-tier architecture (web server, application server, and database), the system is implemented in PHP Hypertext Preprocessor (PHP) scripts. A complete listing of database tables and columns can be found in **Appendix C**.

A central page, index.php[33], provides a common header, menu, frame, and footer for all other pages. Page content indicated by the "page" parameters is loaded into the center frame[34].

[29] Security level 0 is insecure. Security level 5 is secure as of this writing. Newly discovered vulnerabilities in security level 5 should be reported at http://sourceforge.net/p/mutillidae/bugs/.
[30] Security level 1 implements JavaScript validation on some pages and provides an alternative puzzle on the CBC bit flipping challenge
[31] Using an interception proxy such as Burp-Suite is consider best practice but disabling JavaScript works in many cases
[32] The source code may be viewed at http://sourceforge.net/p/mutillidae/code/
[33] Assuming the project is installed on localhost, http://localhost/mutillidae/index.php
[34] This creates an Insecure Direct Object Reference vulnerability

4. Usage

4.1. Installation

4.1.1. Installation on Windows

Mutillidae can be installed by copying a single folder into an Apache-MySQL-PHP (AMP) installation. For Window XP/7, Cross-platform Apache MySQL PHP (XAMPP) (Seidler, 2013) is the default platform. XAMPP is extensively tested and known to work well; however, Mutillidae is also confirmed to operate on Windows Apache MySQL PHP (WAMP) (Bourdon, 2013). An instructional video is available to assist with installation on Windows (Druin, Mutillidae: Installing OWASP Mutillidae II on Windows with XAMPP, 2012). Please see installation steps below.

4.1.2. Installation on Linux

Mutillidae will run on Linux, Apache HTTP Server, MySQL, and PHP (LAMP) (Wikipedia, 2013) stacks on Ubuntu. While not officially supported, Mutillidae also functions on Macintosh, Apache, MySQL and PHP (MAMP) (MAMP, 2007).

4.1.3. Installation Steps

- Acquired the project using one of the following options
  - Download the compressed "ZIP" file[35]
  - Subversion (SVN) (CollabNet, Inc., 2013)[36]
  - Git (git, 2013)[37]
- The entire project is contained within the folder "mutillidae"
- Place the folder into the "web server root directory". The web root for the various AMP platforms varies[38].
- Browse to the application at http://<domain>/mutillidae

[35] http://sourceforge.net/projects/mutillidae/files/mutillidae-project/
[36] svn checkout svn://svn.code.sf.net/p/mutillidae/code/ mutillidae
[37] git clone git://git.code.sf.net/p/mutillidae/git mutillidae
[38] By default, the webroot for XAMPP is C:\xampp\htdocs\. LAMP webroot is often found at /var/www/.

- Click the "Reset DB" (reset database) button to have the database tables automatically provisioned and populated with target data. In addition to creating the application tables, the database script attempts to determine if the MySQL server is available.
- A warning will be issued to the user if the database appears to be offline.[39]

4.1.4. Preinstalled on Well-known distributions

Installation provides the greatest flexibility but Mutillidae comes preinstalled on three popular platforms. These include

- Rapid7 Metasploitable 2 (Moore, 2012)
- Samurai Web Testing Framework (Samurai-WTF) (Johnson, Searle, & DiMaggio, Samurai Web Testing Framework, 2008)
- OWASP Broken Web Apps (OWASP BWA) (Willis, 2012).

These platforms will necessarily have the version of Mutillidae available at the time they were published but the project can be updated by replacing the existing files with the current files.

Samurai-WTF version 2 (InGuardians Labs, Sawyer, Searle, Johnson, & Siles, 2012), a ready-to-use bootable ISO, has a particularly low entry barrier because this platform has both the Mutillidae project plus other excellent deliberately vulnerable web application like Damn Vulnerable Web Application (DVWA) (RandomStorm, 2013) included. Additionally, many tools used in the course of web pen testing are installed (Klein Keane, 2001) (McRee, 2010).

Live distributions have advantages such as being pre-built and avoiding changes to the host hard drive[40]. Each of the distributions comes with the version of Mutillidae available at the time the distribution is created, but videos are available which explain how to upgrade the project to the latest version on Samurai-WTF.

[39] The default username/password assumed for the MySQL server is root:<blank>. Upon MySQL authentication failure, the script will attempt to user root:samurai. "samurai" is the MySQL password on Samurai-WTF.

[40] Live distributions such as Samurai-WTF and Kali Live-System run entirely in RAM unless made "persistent"

- Mutillidae: How to Upgrade to the Latest Mutillidae on Samurai WTF 2.0 (Druin, Mutillidae: How to Upgrade to the Latest Mutillidae on Samurai WTF 2.0, 2012)
- Mutillidae: How to install latest Mutillidae on Samurai WTF 2.0 (Druin, Mutillidae: How to install latest Mutillidae on Samurai WTF 2.0, 2012)

4.2. Mutillidae in action

4.2.1. Selecting an Exercise

Lab exercise are organized by the OWASP Top Ten. For example, selecting the "User Info"[41] page from the menu will load the "User Information" page[42] (**Figure 5**).

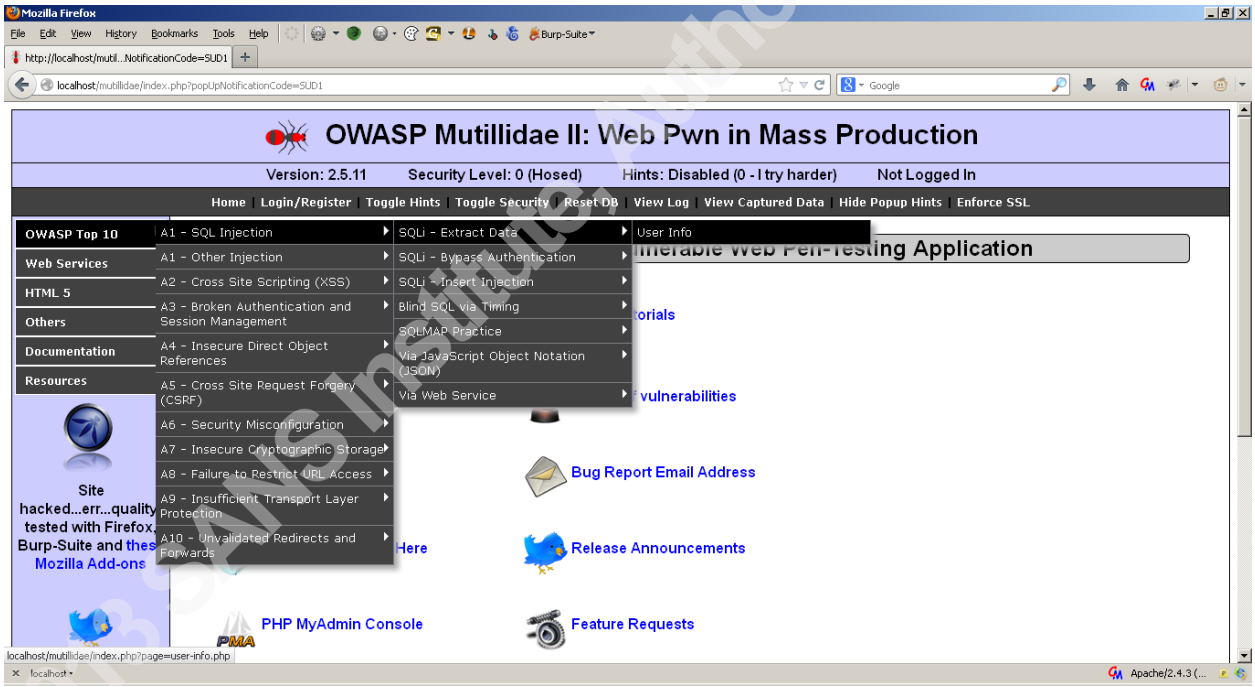

Figure 5: Selecting "OWASP Top 10" -> "A1 – SQL Injection" -> "SQLi – Extract Data" -> "User Info" will direct the user to the "User Information" page

[41] "OWASP Top 10" → "A1 – SQL Injection" → "SQLi – Extract Data" → "User Info"
[42] Assuming default installation the "User Information" page is located at http://localhost/mutillidae/index.php?page=user-info.php

This page displays an account's profile if the user enters the username and password of the account wanted. Assuming the security level is set to "zero" or "one", the page contains a defect which results in a Structured Query Language (SQL)[43] injection vulnerability.

Users are reminded to "reset" between exercises.

4.2.2. Basic Example

SQL injection can be performed quickly by activating the hints, inserting the example provided into the "username" field, and clicking the "Lookup User" button. The hint is provided by opening the "SQL Injection" hint and referring to the section on "Exploitation" (**Figure 6**).

The SQL injection will result in the display of all accounts (**Figure 7**).

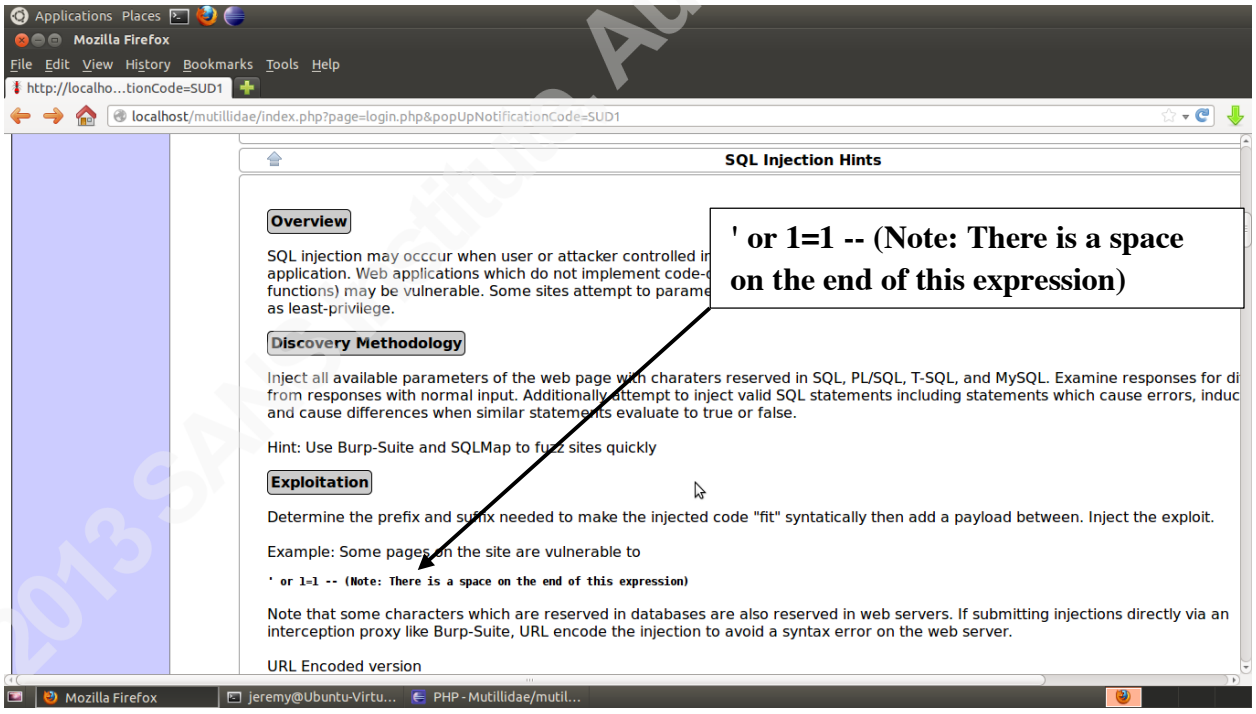

Figure 6: SQL Injection Hint

[43] Structured Query Language (SQL) - http://www.w3schools.com/sql/

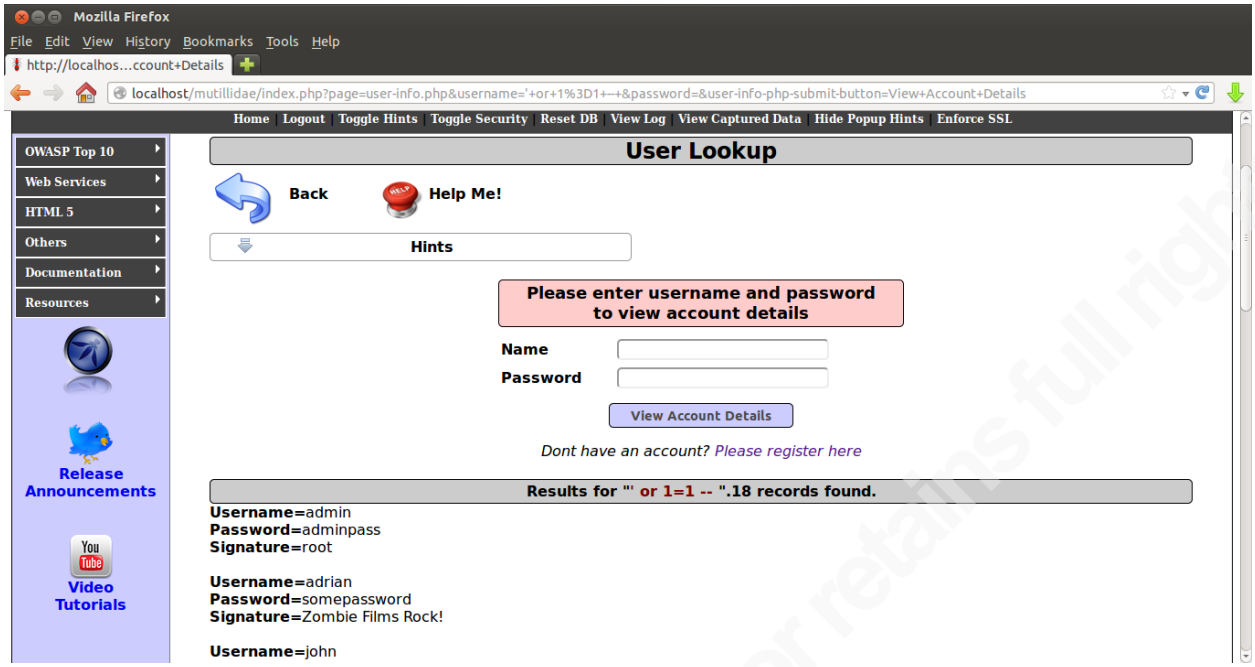

Figure 7: Result of SQL Injection using Built-in help

4.2.3. Intermediate Example

To perform an SQL injection following the map-discover-exploit methodology

1. Induce an error to reveal internal page structure and/or logic
2. Determine SQL fragments needed to form a syntactically correct SQL statement
3. Encode the fragments in order to create a valid injection
4. Inject the fragment into the vulnerable input parameter

This process begins by injecting characters which are meaningful within an SQL query such as single-quote, block-open-comment (/*), inline-comment (--), or others (Oracle, 2013). This will cause the system to throw an exception when attempting to pull account information. Because the page exhibits an "application exception information leakage" vulnerability, specifically an "SQL disclosure" defect, internal system information such as the database query is displayed (Whalen, Khant, & KirstenS, 2013).

In version 2.5.8, this query is the following where "username submitted" and "password submitted" are the form values passed by the user.

*SELECT * FROM accounts WHERE username='**<username submitted>**' AND password='**<password submitted>**'*

If the user only wishes to dump all accounts, they can use the information gained to alter the query by passing not only data but also SQL query syntax that completes the query such that the query will always evaluate to TRUE despite the "WHERE" clause (refer to example query above). One solution would be to pass the following as the "username" form value. The value of the password form field is not relevant in this example. As a placeholder, a value of "anything" might be passed. Note carefully that there is a space at the end of the statement.

Username: ' or 'r' = 'r' --
Password: <not relevant> (example uses the string "anything")

When the user injection is combined with the SQL supplied by the system, the resulting statement would be:

*SELECT * FROM accounts WHERE username='**' or 'r' = 'r' --** ' AND password='**anything**'*

With the comment symbol cancelling the password portion of the query and the username always resulting to TRUE regardless (since the letter "r" is always equal to itself), the effective SQL statement executed is equivalent to:

SELECT * FROM accounts

As such the system will display all records from the accounts table (**Figure 8**).

Because Mutillidae is genuinely vulnerable[44], more realistic exercises may be carried out. In some cases instructors will want to see a complete discovery phase combined with methodical injections.

[44] The user is not expected to enter a predetermined sequence of characters

4.2.4. Advanced Example

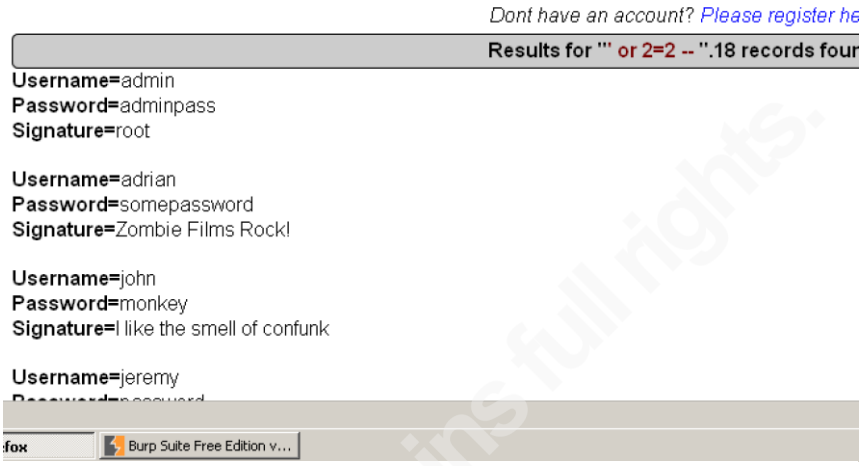

Figure 8: 18 records extracted via SQL injection on User Information page

SQL Injection is capable of reading and writing files when preconditions are met[45]. Building on the "Intermediate Example", a "union" type SQL injection can read files hosted on the database server using database functionality. For example, MySQL includes the LOAD_FILE() function (Oracle Corporation, 2013). Union type SQL injection allows the web application query to run with a query injected by the attacker[46].

Username: ' union select null, null, LOAD_FILE('../../../../../etc/passwd') AS username, null, null--

Password: <not relevant> (example uses the string "anything")

When the user injection is combined with the SQL supplied by the system, the resulting statement is:

*SELECT * FROM accounts WHERE username=*' **union select null, null, LOAD_FILE('../../../../../etc/passwd') AS username, null, null--** ' *AND password=*'***anything***'

The LOAD_FILE() function reads the contents of /etc/passwd[47] file which are displayed by the web application (**Figure 9**).

[45] The database account used by the web application need permissions to read/write files to the operating system
[46] Technically both queries are sub-queries of the union
[47] On Windows systems, boot.ini may provide a good test target

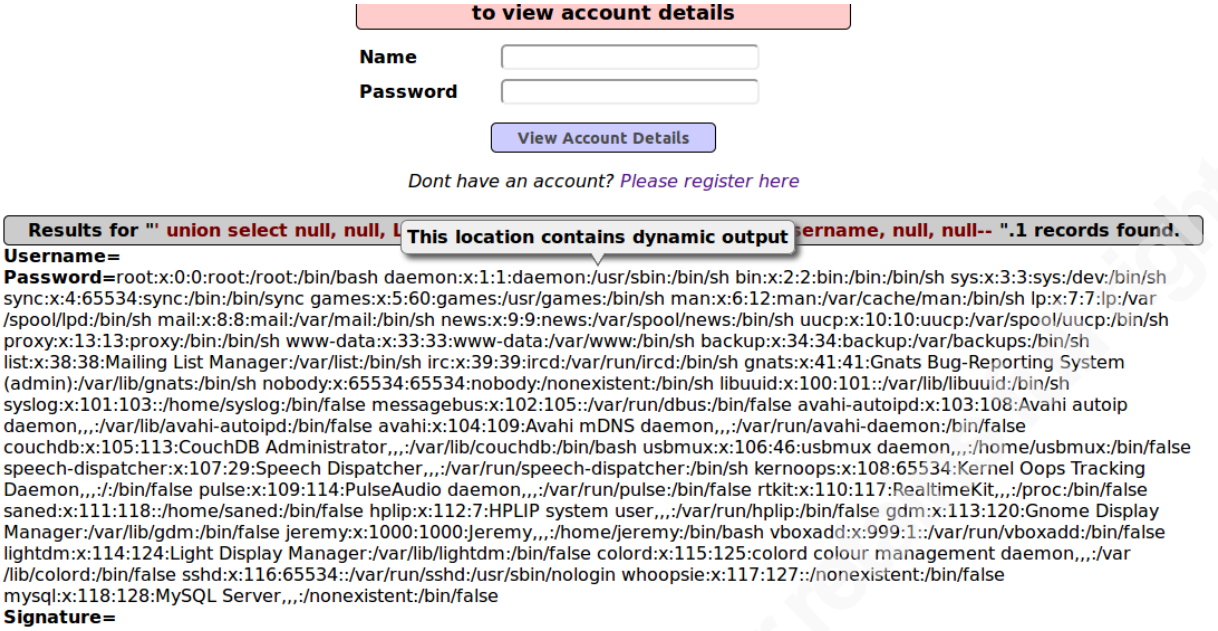

Figure 9: SQL Injection is used to display the etc-passwd file

SQL injection is an interesting example but uses only two vulnerabilities. As noted previously, a complete listing of vulnerabilities is listed in **Appendix A**.

5. Conclusion

With secure coding training and security awareness becoming increasing important portions of a security program, deliberately vulnerable web applications can provide a target on which to practice application security. The Mutillidae web pen testing system presents web, application, web service and database vulnerabilities from the OWASP Top Ten and SANS Top 25 through which exploits and remediation can be performed safely and legally.

Very few target platforms expose a large variety of vulnerabilities in a single system; especially the inclusion of web services. By incorporating these into one system, Mutillidae offers a superior product suitable to prepare for certification exams and master concepts learned in training.

Since the application can be installed relatively easily, operates on both Linux and Windows, and does not require programming experience to operate there is a low barrier to entry. Mutillidae includes hints and tutorials allowing the system to be used in a classroom environment, independent study, or instructor-led training courses. The automated setup and recovery feature provides reproducible results from a known-good state. These features and others make Mutillidae a good candidate to consider using in web application security programs.

Appendix A: Listing of Vulnerabilities (as of July 31, 2013)[48]

- SQL Injection
- O/S Command injection
- JSON injection
- HTML injection
- JavaScript Injection
- DOM injection
- Cascading style sheet injection
- Log injection
- Reflected Cross Site Scripting via GET, POST, Cookies, and HTTP Headers
- Stored Cross Site Scripting
- Cross Site Request Forgery
- Authentication Bypass via SQL injection
- Privilege Escalation via Cookie Injection
- Unencrypted database credentials
- Directory Browsing
- JavaScript validation bypass
- Remote File Inclusion
- Frame source injection
- PHPMyAdmin Console
- SSL Stripping
- Application Exception
- Un-validated Redirects and Forwards
- Phishing
- Click-jacking
- CBC bit flipping (latest)
- Brute force "secret admin pages"
- PHP server configuration disclosure

[48] For the latest list: http://sourceforge.net/projects/mutillidae/files/documentation/listing-of-vulnerabilities-in-mutillidae.txt/download

- Application path disclosure
- Platform path disclosure
- Information disclosure via HTML comments
- robots.txt information disclosure
- Parameter addition
- HTTP Parameter Pollution
- Buffer overflow
- Denial of Service
- Loading of any arbitrary file
- Method Tampering
- Forms caching
- Local File Inclusion
- Comments with sensitive data
- Insecure Cookies
- XML External Entity Injection
- Unrestricted File Upload

Appendix B: Listing of Challenges

- Authentication bypass
- Brute forcing secret administrative pages
- CBC bit flipping
- Privilege escalation
- SSL Stripping
- User agent impersonation

Appendix C: Database Structure

Default database name: (Can be changed by user)

Tables, columns:

blogs_table
- cid
- blogger_name
- comment
- date

accounts
- cid
- username
- password
- mysignature
- is_admin

hitlog
- cid
- hostname
- ip
- browser
- referrer
- date

credit_cards
- ccid
- ccnumber
- ccv
- expiration

pen_test_tools
- tool_id
- tool_name
- phase_to_use
- tool_type
- comment

captured_data
- data_id
- ip_address
- hostname
- port

user_agent_string
referrer
data
capture_date

page_hints
page_name
hint_key
hint

page_help
page_name
help_text_key
order_preference

level_1_help_include_files
level_1_help_include_file_key
level_1_help_include_file

balloon_tips
tip_key
hint_level
tip

Appendix D: Listing of Video Tutorials

Video Tutorials[22]		
Topic	**Video**	**URL**
Installation	Mutillidae: Installing OWASP Mutillidae II on Windows with XAMPP	http://www.youtube.com/watch?v=1hF0Q6ihvjc
	Mutillidae: Installing Metasploitable-2 with Mutillidae on Virtual Box	http://www.youtube.com/watch?v=e0vpBKRZPGc
	Mutillidae: How to install latest Mutillidae on Samurai WTF 2.0	http://www.youtube.com/watch?v=y-Cz3YRNc9U
	Mutillidae: Introduction to Installing, Configuring, and Using Burp-Suite Proxy	http://www.youtube.com/watch?v=L4un5IppoY4
	Mutillidae: How to install and configure Burp-Suite with Firefox	http://www.youtube.com/watch?v=Fj0n17Jtnzw
	Mutillidae: How to remove PHP errors after installing Mutillidae on Windows XAMPP	http://www.youtube.com/watch?v=kDo52RySRME
	Building a Virtual Lab to Practice Pen Testing	http://www.youtube.com/watch?v=5egBxI5Nnwo
	Mutillidae: How to Upgrade to the Latest Mutillidae on Samurai WTF 2.0	http://www.youtube.com/watch?v=obOLDQ-66oQ
Tools and Techniques	Mutillidae: Spidering Web Applications with Burp-Suite	http://www.youtube.com/watch?v=P52EuH9MnTs
	Mutillidae: Basics of Burp-Suite Targets Tab	http://www.youtube.com/watch?v=q3iG7YMjcmE
	Mutillidae: Brute Force Page Names using Burp-Suite Intruder	http://www.youtube.com/watch?v=4Fz9mJeMNkI
	Mutillidae: Using Burp Intruder Sniper to Fuzz Parameters	http://www.youtube.com/watch?v=la5hTlSDKWg
	Mutillidae: Comparing Burp-Suite Intruder Modes Sniper, Battering-ram, Pitchfork, Cluster-bomb	http://www.youtube.com/watch?v=2Ehp33BLRmI
	Mutillidae: Introduction to Burp-Suite Comparer Tool	http://www.youtube.com/watch?v=KxqY_bp13gc
	Mutillidae: Using Burp-Suite Sequencer to Compare CSRF-token strengths	http://www.youtube.com/watch?v=YGRoFU0USRA
	Mutillidae: Basics of Web Request and Response Interception with Burp-Suite	http://www.youtube.com/watch?v=qsE04AhlJrc
	ISSA 2013 Web Pen-testing Workshop - Part 1 - Intro to Mutillidae, Burp Suite & Injection	http://www.youtube.com/watch?v=JPd2YtgJm8Q
	Mutillidae: Overview of Useful Pen Testing Add-ons for Firefox	http://www.youtube.com/watch?v=Id_JC3hJzhk

OWASP 2013 A1: Injection[49] **SQL Injection**	Mutillidae: Bypass Authentication using SQL Injection	http://www.youtube.com/watch?v=0_AN5FKsxaw
	Mutillidae: Automate SQL Injection using sqlmap	http://www.youtube.com/watch?v=dmYp2drEwwE
	Mutillidae: Basics of SQL Injection Timing Attacks	http://www.youtube.com/watch?v=3qOgt9IYgaI
	Mutillidae: Introduction to Union-Based SQL Injection	http://www.youtube.com/watch?v=UcbZUmuMy3U
	Mutillidae: Basics of Inserting Data with SQL Injection	http://www.youtube.com/watch?v=WLmKK4LKdl0
	Mutillidae: Inject Web Shell Backdoor via SQL Injection	http://www.youtube.com/watch?v=lcaqam-CyBE
	Mutillidae: Basics of using SQL Injection to Read Files	http://www.youtube.com/watch?v=EBk0-lJ2LrM
	Mutillidae: Generate Cross Site Scripts with SQL Injection	http://www.youtube.com/watch?v=UH9gx4TyFlk
	Mutillidae: SQL Injection via AJAX request with JSON response	http://www.youtube.com/watch?v=RFEqlg21mkE
	Mutillidae: Basics of using sqlmap - ISSA KY Workshop - February 2013	http://www.youtube.com/watch?v=vTB3Ze901pM
OWASP 2013 A3: Cross-site Scripting[50]	Mutillidae: Explanation of HTTPOnly Cookies in Presence Cross-Site Scripting	http://www.youtube.com/watch?v=YCfInEFWbVA
	Mutillidae: Two Methods to Steal Session Token using Cross-Site Scripting	http://www.youtube.com/watch?v=sTTdFujJIAA
	Mutillidae: Injecting a Cross Site Script via Cascading Stylesheet Context	http://www.youtube.com/watch?v=DXtLNGqfgMo
	Mutillidae: Basics of Injecting Cross-Site Script into HTML onclick Event	http://www.youtube.com/watch?v=C_3I6IpbP78
	Mutillidae: Introduction to locating Reflected Cross-site Scripting	http://www.youtube.com/watch?v=XnOqNCd31B4
	Mutillidae: Sending Persistent Cross-site Scripts into Web Logs to Snag Web Admin	http://www.youtube.com/watch?v=dIGJ7kuj9Qo
	Mutillidae: Generate Cross Site Scripts with SQL Injection	http://www.youtube.com/watch?v=UH9gx4TyFlk
	Mutillidae: Injecting Cross Site Scripts (XSS) into Log Page via Cookie	http://www.youtube.com/watch?v=bj9IcclYG1k
	Introduction to HTML Injection (HTMLi) and Cross Site Scripting (XSS) Using Mutillidae	http://www.youtube.com/watch?v=efHdM5grGkI
	Mutillidae: Introduction to Cross Site Scripting (XSS) via JavaScript String Injection	http://www.youtube.com/watch?v=zs30qw4CF2U
	Mutillidae: Explanation of HTTPOnly Cookies in Presence Cross-Site Scripting	http://www.youtube.com/watch?v=YCfInEFWbVA

[49] (Smithline & Gigler, Top 10 2013-A1-Injection, 2013)
[50] (Smithline, Wichers, Wetter, & Gigler, 2013)

OWASP 2013 A6[51]: Sensitive Data Disclosure **HTML5 Web Storage**	Mutillidae: Adding Values to DOM Storage using Cross-site Scripting	http://www.youtube.com/watch?v=uJoMHujjo_I
	Mutillidae: Alter Values in HTML5 Web Storage using Cross-site Script	http://www.youtube.com/watch?v=N1-FXp7WrC4
	Mutillidae: Alter Values in HTML5 Web Storage using Persistent Cross-site Script	http://www.youtube.com/watch?v=F4I9XfTAsJk
	Mutillidae: Alter Values in HTML5 Web Storage using Reflected Cross-site Script	http://www.youtube.com/watch?v=luMyYV70bk4
	Web Pen Testing HTML 5 Web Storage using JSON Injection	http://www.youtube.com/watch?v=MNvAib9KWzo
	Mutillidae: Stealing HTML5 Storage via JSON Injection	http://www.youtube.com/watch?v=_tGcZKBXsFU
	Mutillidae: Reading Hidden Values from HTML5 Dom Storage	http://www.youtube.com/watch?v=3nAqRp9wt8g
OWASP 2013 A1: Injection[52] **Command Injection**	Mutillidae: Command Injection to Dump Files, Start Services, and Disable Firewall	http://www.youtube.com/watch?v=1bXTq_qaa_U
	Mutillidae: How to Locate the Easter egg File using Command Injection	http://www.youtube.com/watch?v=VWZYyH0VewQ
	Mutillidae: Gaining Administrative Shell Access via Command Injection	http://www.youtube.com/watch?v=GRuRK-bejgM
	Mutillidae: Using Command Injection to Gain Remote Desktop	http://www.youtube.com/watch?v=if17nCdQfMg
OWASP 2013 A1: Injection[53] **HTTP Parameter Pollution**	Mutillidae: Introduction to HTTP Parameter Pollution	http://www.youtube.com/watch?v=Tosp-JyWVS4
OWASP 2013 A2: Broken Authentication and Session Management[54]	Mutillidae: Using Hydra to Brute Force Web Forms Based Authentication	http://www.youtube.com/watch?v=SsUUyizhS60
	Mutillidae: Bypass Authentication using SQL Injection	http://www.youtube.com/watch?v=0_AN5FKsxaw
	Mutillidae: Bypass Authentication via Authentication Token Manipulation	http://www.youtube.com/watch?v=mEbmturLljU
	Mutillidae: Brute Force Authentication using Burp-Intruder	http://www.youtube.com/watch?v=frtkNB5G3vI
	Mutillidae: Analyze Session Token Randomness using Burp-Suite Sequencer	http://www.youtube.com/watch?v=kNSAhKiXctA

[51] (Gigler & Smithline, 2013)
[52] (Smithline & Gigler, Top 10 2013-A1-Injection, 2013)
[53] (Smithline & Gigler, Top 10 2013-A1-Injection, 2013)
[54] (Gigler & Smithline, Top 10 2013-A2-Broken Authentication and Session Management, 2013)

OWASP 2013 A6[55]: Sensitive Data Disclosure	Mutillidae: Determine Server Banners using Netcat, Nikto, and w3af	http://www.youtube.com/watch?v=goCm1TCJ29g
	Mutillidae: Using Nmap to Fingerprint HTTP servers and Web Applications	http://www.youtube.com/watch?v=VQV-y_-AN80
	Mutillidae: Finding Comments and File Metadata using Multiple Techniques	http://www.youtube.com/watch?v=Ouqubk9H-KY
Local File Inclusion	Mutillidae: How to Exploit Local File Inclusion Vulnerability using Burp-Suite	http://www.youtube.com/watch?v=t8w6Bd5zxbU
	ISSA 2013 Web Pen-testing Workshop - Part 6 - Local/Remote File Inclusion	http://www.youtube.com/watch?v=0fODWaeupV0
Bypassing Client Restrictions	Mutillidae: Two Methods to Bypass JavaScript Validation	http://www.youtube.com/watch?v=e0M-qJYhCnk
	Mutillidae: XSS bypassing JavaScript Validation	http://www.youtube.com/watch?v=9EaNr_8D65A
	Mutillidae: How to Bypass Maxlength Restrictions on HTML Input Fields	http://www.youtube.com/watch?v=mjeBPCGA7ko
OWASP A5: Security Misconfiguration[56] **CBC Bit Flipping**	Mutillidae: Introduction to CBC Bit Flipping Attack	http://www.youtube.com/watch?v=TNt2rJcxdyg
OWASP A5: Security Misconfiguration[57] **SSL Stripping**	Mutillidae: Using Ettercap and SSLstrip to Capture Credentials	http://www.youtube.com/watch?v=n_5NGkOnr7Q
OWASP 2013 A1: Injection[58] **XML External Entity Injection**	Mutillidae: Introduction to XML External Entity Injection	http://www.youtube.com/watch?v=DJaX4HN2gwQ
	ISSA KY September 2013 Workshop - Introduction to XML External Entity Injection	http://www.youtube.com/watch?v=Zl8U2YVp2lw
HTTP Methods	Mutillidae: Determine HTTP Methods using	http://www.youtube.com/wat

[55] (Gigler & Smithline, 2013)
[56] (Gigler & Smithline, Top 10 2013-A5-Security Misconfiguration, 2013)
[57] (Gigler & Smithline, Top 10 2013-A5-Security Misconfiguration, 2013)
[58] (Smithline & Gigler, Top 10 2013-A1-Injection, 2013)

	Netcat	ch?v=MxiVx7e_FbM
	Mutillidae: Examination of Cache-Control and Pragma no-cache Headers	http://www.youtube.com/watch?v=awMpx8Bhir0
	Mutillidae: Demonstration of Frame-busting JavaScript and X-Frame-Options Header	http://www.youtube.com/watch?v=KWcckqnUGo8
	Mutillidae: Three Methods for Viewing HTTP Request and Response Headers	http://www.youtube.com/watch?v=NkdtMiUG30A
	Mutillidae: Introduction to User-agent Impersonation	http://www.youtube.com/watch?v=VAGG4uC1ogw
Web Services	ISSA KY August 2013 Workshop - Introduction to Pen Testing Web Services	http://www.youtube.com/watch?v=e6HAQnvuaic

Appendix E: Types of Cross-site Scripting Vulnerabilities Implemented

1. Hypertext Markup Language (HTML) (Druin, Introduction to HTML Injection (HTMLi) and Cross Site Scripting (XSS) Using Mutillidae, 2013)
2. JavaScript strings (Druin, Mutillidae: Introduction to Cross Site Scripting (XSS) via Javascript String Injection, 2013)
3. JavaScript Object Notation (JSON) strings (Druin, Web Pen Testing HTML 5 Web Storage using JSON Injection , 2012)
4. Cascading style attributes (Druin, Mutillidae: Injecting a Cross Site Script via Cascading Stylesheet Context , 2012)
5. Extensible Markup Language (XML) (Druin, Mutillidae: Introduction to XML External Entity Injection, 2013)
6. HTTP "cookies"/headers (Druin, Mutillidae: Injecting Cross Site Scripts (XSS) into Log Page via Cookie , 2012)

Appendix F: Types of SQL Injection Vulnerabilities Implemented

1. Extract data (Druin, Mutillidae: Introduction to union-based SQL Injection, 2012)
2. Bypass authentication (Druin, Mutillidae: Bypass Authentication using SQL Injection, 2012)
3. Upload web backdoor shells (ISSA 2013 Web Pen-testing Workshop - Part 3 - Uploading Web Shells via SQL Injection, 2013)

Appendix G: Built-in Help Topics

- Html5 Web Storage
- Unrestricted File Upload
- JSON Injection
- SQL Injection
- Html Injection
- Cross Site Scripting
- Cross Site Request Forgery
- Xml External Entity Attack
- Authentication Bypass
- JavaScript Injection
- Command Injection
- Local File Inclusion
- Insecure Direct Object Reference
- CBC Bit Flipping Attack
- Unvalidated Redirects And Forwards
- SSL Misconfiguration
- Cascading Style Sheet Injection
- Buffer Overflow
- Click Jacking
- Parameter Addition
- Dom Injection
- Secret Administrative Pages
- Parameter Pollution
- Method Tampering
- Information Disclosure
- JavaScript Validation Bypass
- User Agent Impersonation
- Platform Path Disclosure
- Application Path Disclosure
- Directory Browsing
- Remote File Inclusion
- Frame Source Injection
- Robots.Txt

6. References

Apache.org. (2013, June 18). *Apache™ Subversion.* Retrieved July 6, 2013, from Apache.org: http://subversion.apache.org/

BeEF Project. (n.d.). *BeEF: The Browser Exploitation Framework Project*. Retrieved June 30, 2013, from BeEFProject.com: http://beefproject.com/

Bennetts, S., & Neumann, A. (2013, July 3). *OWASP Zed Attack Proxy Project.* Retrieved July 6, 2013, from The Open Web Application Security Project (OWASP): https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

Blankenship, B., Hartmann, K., Koenig, C., Baso, S., & Sullivan, K. (2013, April 30). *Louisville Chapter.* Retrieved 7 2013, July, from The Open Web Application Security Project (OWASP): https://www.owasp.org/index.php/Louisville#Past_Events

Bourdon, R. (2013, June 25). *WampServer, a Windows web development environment*. Retrieved June 25, 2013, from WampServer: http://www.wampserver.com/en/

Brandel, M. (2008, March 3). *How to Evaluate (and Use) Web Application Security Scanners.* Retrieved July 6, 2013, from CSO Online: http://www.csoonline.com/article/221326/how-to-evaluate-and-use-web-application-security-scanners

Byrne, D. (2012, August 22). *Grendel-Scan*. Retrieved July 6, 2013, from SourceForge: http://sourceforge.net/projects/grendel/

Cenzic, Inc. (n.d.). *Cenzic Hailstorm Technology*. Retrieved June 30, 2013, from Cenzic.com: http://www.cenzic.com/technology/index.html

CollabNet, Inc. (2013, June 18). *Apache Subversion*. Retrieved July 1, 2013, from Subversion: http://subversion.apache.org/

Crenshaw, A. (2009, September 23). *Deliberately Insecure Web Applications for Learning Web App Security*. Retrieved June 25, 2013, from Irongeek: http://www.irongeek.com/i.php?page=security/deliberately-insecure-web-applications-for-learning-web-app-security

Crenshaw, A. (2009, March 2). *Irongeek.com*. Retrieved June 30, 2013, from Deliberately Insecure Web Applications for Learning Web App Security: http://www.irongeek.com/i.php?page=security/deliberately-insecure-web-applications-for-learning-web-app-security

Crenshaw, A. (2012, November 29). *Mutillidae-classic (Mutillidae versions 1.x).* Retrieved June 30, 2013, from Sourceforge: http://sourceforge.net/projects/mutillidae/files/mutillidae-project/mutillidae-classic%20%28mutillidae%20versions%201.x%29/

Crenshaw, A. (2013, June 21). *Irongeek.com*. Retrieved June 27, 2013, from Irongeek.com: http://www.irongeek.com/

Crenshaw, A. (2013). *Web Application Pen-testing Tutorials With Mutillidae.* Retrieved June 30, 2013, from Irongeek.com: http://www.irongeek.com/i.php?page=videos/web-application-pen-testing-tutorials-with-mutillidae

David Shelly, R. M. (2010, November 8). *Closing the Gap: Analyzing the Limitations of Web Application Vulnerability Scanners.* Retrieved June 25, 2013, from The Open Web Application Security Project (OWASP): http://www.owasp.com/index.php/Closing_the_Gap:_Analyzing_the_Limitations_of_Web_Application_Vulnerability_Scanners

die.net. (2013). *unzip(1) - Linux man page*. Retrieved July 6, 2013, from die.net: http://linux.die.net/man/1/unzip

Druin, J. (2011, March 24). *OWASP Mutillidae II Web Pen-Test Practice Application*. Retrieved June 25, 2013, from Sourceforge: http://sourceforge.net/projects/mutillidae/

Druin, J. (2012, October 7). *2012 KY ISSA Conference: Introduction to OWASP Mutillidae II Web Pen Testing Environment.* Retrieved July 7, 2013, from YouTube: https://www.youtube.com/watch?v=CYsiNYeAS6U

Druin, J. (2012, February 4). *Mutillidae: Brute Force Page Names using Burp-Suite Intruder.* Retrieved June 30, 2013, from YouTube: https://www.youtube.com/watch?v=4Fz9mJeMNkI

Druin, J. (2012, January 9). *Mutillidae: Bypass Authentication using SQL Injection.* Retrieved June 30, 2013, from YouTube: http://www.youtube.com/watch?v=0_AN5FKsxaw

Druin, J. (2012, January 10). *Mutillidae: Bypass Authentication via Authentication Token Manipulation.* Retrieved July 29, 2013, from YouTube: https://www.youtube.com/watch?v=mEbmturLljU

Druin, J. (2012, September 16). *Mutillidae: How to install latest Mutillidae on Samurai WTF 2.0.* Retrieved July 1, 2013, from YouTube: http://www.youtube.com/watch?v=y-Cz3YRNc9U

Druin, J. (2012, September 17). *Mutillidae: How to Upgrade to the Latest Mutillidae on Samurai WTF 2.0.* Retrieved July 1, 2013, from YouTube: http://www.youtube.com/watch?v=obOLDQ-66oQ

Druin, J. (2012, January 26). *Mutillidae: Injecting a Cross Site Script via Cascading Stylesheet Context .* Retrieved June 30, 2013, from YouTube: https://www.youtube.com/watch?v=DXtLNGqfgMo

Druin, J. (2012, March 23). *Mutillidae: Injecting Cross Site Scripts (XSS) into Log Page via Cookie .* Retrieved June 30, 2013, from YouTube: http://www.youtube.com/watch?v=bj9IcclYG1k

Druin, J. (2012, February 17). *Mutillidae: Installing OWASP Mutillidae II on Windows with XAMPP.* Retrieved July 6, 2013, from YouTube: http://www.youtube.com/watch?v=1hF0Q6ihvjc

Druin, J. (2012, August 7). *Mutillidae: Introduction to CBC bit flipping attack .* Retrieved June 30, 2013, from YouTube: https://www.youtube.com/watch?v=TNt2rJcxdyg

Druin, J. (2012, January 21). *Mutillidae: Introduction to union-based SQL Injection.* Retrieved June 30, 2013, from YouTube: http://www.youtube.com/watch?v=UcbZUmuMy3U

Druin, J. (2012, March 27). *Mutillidae: Manual Directory Browsing to reveal Easter Egg File .* Retrieved June 30, 2013, from YouTube: https://www.youtube.com/watch?v=zWZ33xdDvk4

Druin, J. (2012, August 26). *Mutillidae: Using Ettercap and SSLstrip to Capture Credentials .* Retrieved June 30, 2013, from YouTube: https://www.youtube.com/watch?v=n_5NGkOnr7Q

Druin, J. (2012, June 1). *Web Pen Testing HTML 5 Web Storage using JSON Injection .* Retrieved June 30, 2013, from YouTube: http://www.youtube.com/watch?v=MNvAib9KWzo

Druin, J. (2013). *@webpwnized.* Retrieved 6 2013, July, from Twitter: http://en.twitter.com/webpwnized

Druin, J. (2013, February 1). *Basics of using sqlmap - ISSA KY Workshop - February 2013.* Retrieved June 30, 2013, from YouTube: http://www.youtube.com/watch?v=vTB3Ze901pM

Druin, J. (2013, June 23). *Code Tree*. Retrieved July 6, 2013, from SourceForge: http://sourceforge.net/p/mutillidae/code/HEAD/tree/

Druin, J. (2013, June 23). *Documentation: Listing of Vulnerabilities in Mutillidae 2.x.* Retrieved June 30, 2013, from SourceForge: http://iweb.dl.sourceforge.net/project/mutillidae/documentation/listing-of-vulnerabilities-in-mutillidae.txt

Druin, J. (2013, March 2). *Introduction to HTML Injection (HTMLi) and Cross Site Scripting (XSS) Using Mutillidae.* Retrieved June 30, 2013, from YouTube: http://www.youtube.com/watch?v=efHdM5grGkI

Druin, J. (2013, June 23). *Mutillidae II Features.* Retrieved July 6, 2013, from SourceForge: http://sourceforge.net/projects/mutillidae/

Druin, J. (2013). *Mutillidae Installation on XAMPP (Win7).* Retrieved July 6, 2013, from SourceForge: http://sourceforge.net/projects/mutillidae/files/documentation/mutillidae-installation-on-xampp-win7.pdf/download

Druin, J. (2013, June 30). *Mutillidae: Introduction to Cross Site Scripting (XSS) via Javascript String Injection.* Retrieved June 30, 2013, from YouTube: http://www.youtube.com/watch?v=zs30qw4CF2U

Druin, J. (2013, June 30). *Mutillidae: Introduction to XML External Entity Injection.* Retrieved June 30, 2013, from YouTube: http://youtu.be/DJaX4HN2gwQ

Druin, J. (2013, June 30). *Query Results for Mutillidae on webpwnized channel.* Retrieved June 30, 2013, from YouTube: http://www.youtube.com/user/webpwnized/search?query=mutillidae

Druin, J. (2013, June 23). *SourceForge: Mutillidae 2.x Download (Current Version).* Retrieved June 30, 2013, from Sourceforge: http://sourceforge.net/projects/mutillidae/files/

Druin, J. (2013, June 15). *sqlmap-targets.php.* Retrieved July 6, 2013, from Sourceforge: http://sourceforge.net/p/mutillidae/code/36/tree/sqlmap-targets.php

Druin, J. (2013). *webpwnized's channel*. Retrieved June 30, 2013, from YouTube: http://www.youtube.com/user/webpwnized

Eston, T., Abraham, J., & Johnson, K. (2011, August 21). *Don't Drop the SOAP: Real World Web Service Testing.* Retrieved July 6, 2013, from Blackhat Media: http://media.blackhat.com/bh-us-11/Johnson/BH_US_11_JohnsonEstonAbraham_Dont_Drop_the_SOAP_WP.pdf

Fisher, J. (2009, October 22). *OWASP DirBuster Project*. Retrieved June 30, 2013, from The Open Web Application Security Project (OWASP): https://www.owasp.org/index.php/Category:OWASP_DirBuster_Project

GIAC. (2013). *GIAC Web Application Penetration Tester (GWAPT) - Exam Certification Objectives & Outcome Statements.* Retrieved July 6, 2013, from GIAC.org: http://www.giac.org/certification/web-application-penetration-tester-gwapt

Gigler, T., & Smithline, N. (2013, June 23). *Top 10 2013-A2-Broken Authentication and Session Management*. Retrieved August 2, 2013, from The Open Web Application Security Project (OWASP): https://www.owasp.org/index.php/Top_10_2013-A2-Broken_Authentication_and_Session_Management

Gigler, T., & Smithline, N. (2013, June 23). *Top 10 2013-A5-Security Misconfiguration*. Retrieved August 2, 2013, from The Open Web Application Security Project (OWASP): https://www.owasp.org/index.php/Top_10_2013-A5-Security_Misconfiguration

Gigler, T., & Smithline, N. (2013, June 23). *Top 10 2013-A6-Sensitive Data Exposure*. Retrieved July 31, 2013, from The Open Web Application Security Project (OWASP): https://www.owasp.org/index.php/Top_10_2013-A6-Sensitive_Data_Exposure

git. (2013, July 30). *Git Fast Version Control*. Retrieved July 30, 2013, from Git-SCM.com: http://git-scm.com/about

InGuardians Labs, Sawyer, J., Searle, J., Johnson, K., & Siles, R. (2012, August 17). *Samurai.* Retrieved July 6, 2013, from SourceForge: http://sourceforge.net/projects/samurai/files/SamuraiWTF%202.0%20Branch/

ISSA 2013 Web Pen-testing Workshop - Part 3 - Uploading Web Shells via SQL Injection. (2013, May 20). Retrieved June 30, 2013, from YouTube: http://www.youtube.com/watch?v=u3Wf_3SI_zE

Jeffrey L. Hieb, P. (2013, June 13). Assistant Professor, University of Louisville Speed-Scientific Department of Engineering Fundamentals. (J. Druin, Interviewer)

Johnson, K. (2011, March 8). *Damn Vulnerable Web Services.* Retrieved July 6, 2013, from Secure Ideas LLC: http://dvws.secureideas.net/index.html

Johnson, K., Searle, J., & DiMaggio, F. (2008). *Samurai Web Testing Framework*. Retrieved June 27, 2013, from Inguardians: http://samurai.inguardians.com/

KirstenS. (2011, December 8). *Cross-site Scripting (XSS).* Retrieved June 30, 2013, from The Open Web Application Security Project (OWASP): http://www.owasp.com/index.php/Cross_site_scripting

KirstenS. (2013, April 27). *SQL Injection.* Retrieved June 30, 2013, from The Open Web Application Security Project (OWASP): http://www.owasp.com/index.php/SQL_Injection

Klein Keane, J. C. (2001). *Samurai Web Testing Framework Applications.* Retrieved July 6, 2013, from Mad Irish . net: http://www.madirish.net/188

MAMP. (2007, May 2). *MAMP: Mac - Apache - MySQL - PHP*. Retrieved June 25, 2013, from MAMP: http://www.mamp.info/en/mamp/index.html

Marlinspike, M. (2012). *sslstrip.* Retrieved June 30, 2013, from Thoughtcrime.org: http://www.thoughtcrime.org/software/sslstrip/

Mavituna Security Ltd. (n.d.). *Netsparker Community Edition*. Retrieved June 30, 2013, from mavituna security: https://www.mavitunasecurity.com/communityedition/

McHenry, B. (2013, August 5). Security Solutions Architect F5 Networks. (J. Druin, Interviewer)

McRee, R. (2010, December). *SamuraiWTF: The Life Cycle of a Web Application Vulnerability Analysis.* Retrieved July 6, 2013, from Toolsmith, ISSA Journal: http://holisticinfosec.org/toolsmith/pdf/december2010.pdf

Moore, H. (2012, May 31). *Metasploitable 2 Exploitability Guide.* Retrieved June 27, 2013, from Rapid7 Security Street: https://community.rapid7.com/docs/DOC-1875

Offensive Security. (2013). *Kali Linux*. Retrieved June 30, 2013, from Kali Linux: http://www.kali.org/

Oracle. (2013). *Chapter 13. SQL Statement Syntax.* Retrieved July 6, 2013, from MySQL 5.7 Reference Manual: http://dev.mysql.com/doc/refman/5.7/en/sql-syntax.html

Oracle. (2013). *Chapter 19. INFORMATION_SCHEMA Tables.* Retrieved July 6, 2013, from MySQL Server Documentation: http://dev.mysql.com/doc/refman/5.7/en/information-schema.html

Oracle Corporation. (2013). *MySQL 5.7 Manual 12.5 String Functions*. Retrieved August 7, 2013, from MySQL Documentation: http://dev.mysql.com/doc/refman/5.7/en/string-functions.html#function_load-file

OWASP Foundation. (2010, April 20). *Top 10 2007*. Retrieved June 25, 2013, from The Open Web Application Security Project (OWASP): https://www.owasp.org/index.php/Top_10_2007

OWASP Foundation. (2010, April 27). *Top 10 2010-Main*. Retrieved June 25, 2013, from The Open Web Application Security Project (OWASP): https://www.owasp.org/index.php/Top_10_2010-Main

Pavlov, I. (2013). *7-Zip file archiver*. Retrieved July 6, 2013, from 7-Zip: www.7-zip.org/

PortSwigger Ltd. (2013). *Burp Suite*. Retrieved June 30, 2013, from Portswigger: http://portswigger.net/burp/

RandomStorm. (2013). *Damn Vulnerable Web Application (DVWA).* Retrieved July 6, 2013, from DVWA: http://www.dvwa.co.uk/

Refsnes Data. (2013). *The XMLHttpRequest Object*. Retrieved June 30, 2013, from w3schools.com: http://www.w3schools.com/dom/dom_httprequest.asp

Riancho, A. (2013). *w3af*. Retrieved June 30, 2013, from w3af.org: http://w3af.org/

Russell, R., Quinlan, D., & Yeoh, C. (2004, January 28). *Filesystem Hierarchy Standard.* Retrieved July 6, 2013, from Linux Foundation: http://refspecs.linuxfoundation.org/FHS_2.3/fhs-2.3.pdf

SANS. (2011, June 27). *CWE/SANS TOP 25 Most Dangerous Software Errors*. Retrieved June 25, 2013, from SANS: http://www.sans.org/top25-software-errors/

SANS™ Institute. (2013). *SEC542: Web App Penetration Testing and Ethical Hacking*. Retrieved 6 2013, July, from SANS.org: http://www.sans.org/course/web-app-penetration-testing-ethical-hacking

SANS™ Institute. (2013). *SEC642: Advanced Web App Penetration Testing and Ethical Hacking*. Retrieved July 6, 2013, from SANS.org: http://www.sans.org/course/advanced-web-app-penetration-testing-ethical-hacking

Seidler, K. '. (2013, June 26). *XAMPP: Cross-platform Apache MySQL PHP*. Retrieved June 27, 2013, from Apache Friends: http://www.apachefriends.org/en/xampp.html

Smithline, N., & Gigler, T. (2013, June 23). *Top 10 2013-A1-Injection*. Retrieved July 31, 2013, from The Open Web Application Security Project (OWASP): https://www.owasp.org/index.php/Top_10_2013-A1-Injection

Smithline, N., Wichers, D., Wetter, D., & Gigler, T. (2013, June 14). *Top 10 2013-A3-Cross-Site Scripting (XSS)*. Retrieved July 31, 2013, from The Open Web Application Security Project (OWASP): https://www.owasp.org/index.php/Top_10_2013-A3-Cross-Site_Scripting_(XSS)

SourceForge. (2013). *SourceForge Mailing Lists.* Retrieved July 6, 2013, from SourceForge: http://sourceforge.net/apps/trac/sourceforge/wiki/Mailing%20lists

Stampar, M., & Damele, B. (n.d.). *sqlmap Automatic SQL injection and database takeover tool*. Retrieved June 30, 2013, from sqlmap.org: http://sqlmap.org/

Sullo, C., & Lodge, D. (2010). *Nikto 2*. Retrieved June 30, 2013, from CIRT.net: http://www.cirt.net/nikto2

The Open Web Application Security Project (OWASP). (2009, March 18). *OWASP Education Track: What Developers Should Know on Web Application Security.* Retrieved June 25, 2013, from The Open Web Application Security Project: https://www.owasp.org/index.php/Education_Track:_What_Developers_Should_Know_on_Web_Application_Security

van Hauser. (2013, January 7). *thc.org*. Retrieved June 30, 2013, from THC-Hydra: http://www.thc.org/thc-hydra/

Verizon RISK Team et al. (2012). *Verizon 2012 Data Breach Investigations Report.* Retrieved June 25, 2013, from Verizon Enterprise: http://www.verizonenterprise.com/resources/reports/rp_data-breach-investigations-report-2012-ebk_en_xg.pdf

Whalen, S., Khant, A., & KirstenS. (2013, June 17). *Information Leakage.* Retrieved July 6, 2013, from The Open Web Application Security Project (OWASP): https://www.owasp.org/index.php/Information_Leakage

Wikipedia. (2013, June 17). *LAMP (software bundle)*. Retrieved June 25, 2013, from Wikipedia: http://en.wikipedia.org/wiki/LAMP_(software_bundle)

Willis, C. (2012, July 24). *OWASP Broken Web Applications Project*. Retrieved June 27, 2013, from The Open Web Application Security Project (OWASP): https://www.owasp.org/index.php/OWASP_Broken_Web_Applications_Project

WinZip Computing, S.L., A Corel Company. (2013). *WinZip® 17.5*. Retrieved July 6, 2013, from WinZip: http://www.winzip.com/win/en/index.htm

Zalewski, M. (2008, July 2). *ratproxy*. Retrieved June 6, 2013, from Google code: https://code.google.com/p/ratproxy/