Dokumentation

## ButterFly

### Installation

Das ButterFly Projekt wird auf einem Linux Server installiert. Die gzip Datei kann unter <http://sourceforge.net/projects/thebutterflytmp/> heruntergeladen werden. Folgende Distributionen werden für diese Installation unterstützt:

* Linux Fedora8
* Linux Debian
* Linux Gentoo
* FreeBSD 6.x, 7.x

Die Installation ist sehr einfach gestaltet. Als erstes muss die gzip Datei entpackt werden und in die richtige Ordnerstruktur eingebunden werden. Das geschieht mit Hilfe eines Befehls:

* tar -zxvpf butterfly\_1.x.tar.gz --directory /

Die gzip Datei wird entpackt und in das root Verszeichnis des users verschoben.

Durch den Aufruf eines shell Skripts wird die Installation fortgesetzt. Das Skript befindet sich unter:

* /usr/local/butterfly/start/start.sh

Wichtig: Vor der Installation sollten noch zusätzlich 2 User angelegt werden, die von ButterFly benötigt werden. Folgende Befehle sind auszuführen:

groupadd -g 2001 bfly1

groupadd -g 2002 bfly2

useradd -c \"ButterFly Apache Account\" -d /usr/local/butterfly/ -g bfly1 -s /usr/sbin/nologin -u 2001 bfly1

useradd -c \"ButterFly MySQL Account\" -d /usr/local/butterfly/ -g bfly2 -s /usr/sbin/nologin -u 2002 bfly2

Werden diese User nicht angelegt. Kommt es zu einer Fehlermeldung bei der Installation.

War die Installation erfolgreich, sollten folgende Ausgaben des Skripts zu sehen sein:

---------------------------------------------------

Butterfly application 1.0

---------------------------------------------------

Setting up the environment...

Checking tools in PATH: OK

Checking privileges of the user: OK

Does ButterFly BFLY1(Apache) user exist?: YES

Does ButterFly BFLY2(MySQL) user exist?: YES

Is port 80 available for the ButterFly?: YES

Is port 3306 available for the ButterFly?: YES

Creating DEV structure: OK

Starting APACHE: OK

Starting MYSQL: OK

Um ButterFly auf Client Systemen verwenden zu können müssen die 2 Domains des Servers in die Hosts Datei reingeschrieben werden. Diese 2 Domains lauten:

* <your server IP> insecure.butterfly.prv
* <your server IP> secure.butterfly.prv

### Komponenten

ButterFly wird komplett abgeschottet vom Hauptsystem installiert. Es ist eine chrooted Umgebung. Das heißt, dass der root Ordner des Servers auf den des ButterFly Ordners geändert wird. ButterFly liefert sämtliche Komponenten mit, die für Webentwicklung/Tests verwendet werden. Es wird Apache, MySQL und PHP 5.2.5 mit Suhosin Patch (secure), 5.1.1 (insecure) im Packet mitgeliefert. Es ist daher keine extra Installation durchzuführen.

ButterFly wurde auf einem Ubuntu Server aufgesetzt. Leider gab es bei der Installation massive Probleme mit dem MySQL deamon. Dieser wollte nicht starten. Wie schon im Kapitel Installation erwähnt ist ButterFly auch nicht unter Ubuntu getestet.

Ein Versuch in einer Debian VM auf einem lokalen Rechner war jedoch erfolgreich.

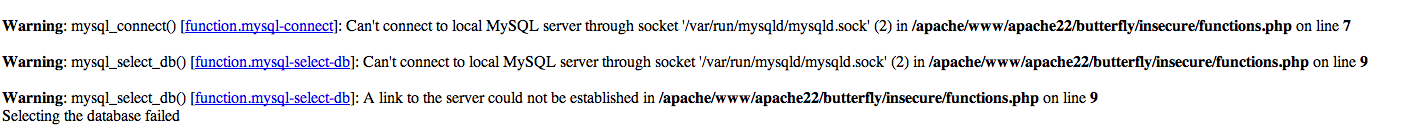
### Verwendung

ButterFly bietet zum Testen 2 User Levels. Es wird eine Bestellumgebung eines Webshops simuliert. Bestellungen können aufgegeben, verschickt oder akzeptiert werden. Es gibt eine Test Website für einen Fileupload und es wird auch eine AJAX Kommunikation simuliert.

Es gibt zwei Modi zum Testen. Eine unsicher und eine sichere Domäne. Bei der unsicheren Domäne werden absichtlich Fehler eingebaut, die unter den OWASP Top 10 zu finden sind.

Leider konnte das Tool nicht getestet werden, da es sich am Ubuntu Server nicht installieren lässt. Der MySQL deamon lässt sich nicht starten, obwohl die Installation mit Root Rechten durchgeführt wurde. Auf einer Debian VM konnte es installiert werden, jedoch gab es Probleme bei der Nutzung.

Aufruf von der Domain ohne MySQL:



Ich würde vorschlagen, dass einer der unterstützten Linux Derivate verwendet wird, um ButterFly aufsetzen zu können.

### Sicherheitslücken

* CRLF injection / HTTP Response splitting
* Redirection attack
* Session replay (timeout of a session control)
* Session destruction vulnerability
* Stump session analysis
* Predictable session tokens
* Cross-Site Scripting (XSS)
* SQL injection
* File upload issues
* brute-forcing web server folders
* Cross-user access
* Privilege escalation
* AJAX vulnerabilities

## bWAPP

### Installation

Das bWAPP kann von <http://www.mmeit.be/bwapp/download.htm> heruntergeladen werden. Folgende Voraussetzungen müssen für eine Verwendung erfüllt sein:

* Windows, Linux, Unix, Mac OS
* Web Server (Apache, IIS)
* PHP extension
* MySQL

Einfachste Möglichkeit ist das XAMPP package am Rechner zu installieren, da hierbei alle erforderlichen Tools (Apache, MySQL, PHP extension) installiert werden.

Sind alle erforderlichen Komponenten installiert, kann das heruntergeladene bWAPP „zip“ Archive entpackt und ins das Root Verzeichnis des Web Servers verschoben werden. Danach muss noch die Datenbank Verbindung eingerichtet werden. Dazu muss das file „settings.php“ im „admin“ folder angepasst werden.

Example:

$db\_server = "localhost"; // your database server (IP/name), here 'localhost'

$db\_username = "root"; // your MySQL user, here 'root'

$db\_password = ""; // your MySQL password, here 'blank'

Um die Datenbank zu installieren muss zum Verzeichnis „install.php“ navigiert werden.

Example:

http://localhost/bWAPP/install.php

Im Fenster muss anschließend auf „here“ geklickt werden um die Datenbank zu installieren. Danach wird die Datenbank generiert und befüllt. Danach kann man sich mit dem Standard Account authentifizieren oder man richtet einen neuen Account ein.

Default credentials:

Bee/bug

### Komponenten

Unser bWAPP wurde auf einem Apache Web Server mit MySQL Datenbank auf einem Windows Server 2012 installiert. Um die Installation der einzelnen Komponenten zu vereinfachen wurde das XAMPP package installiert, welches die einzelnen Komponenten (Apache Web Server, MySQL, PHP extension) automatisch installiert. Mit dem XAMPP Control Panel kann der Web Server und MySQL Datenbank einfach gestartet werden.

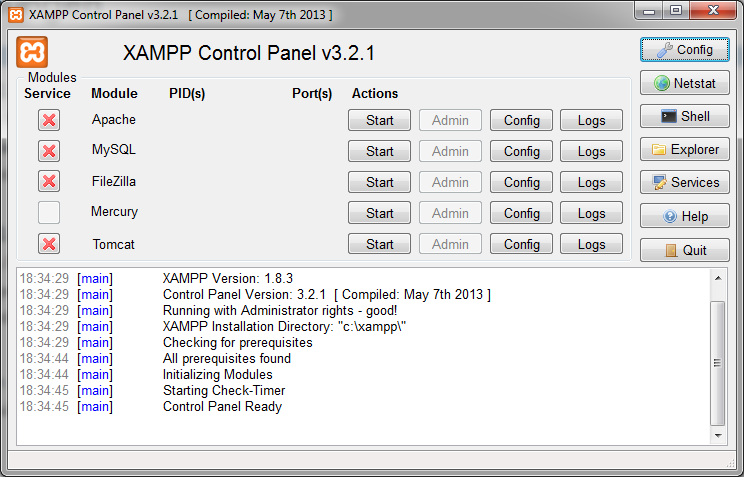


Abbildung 1: Start Screen des XAMPP Control Panel

* Apache Web Server

Falls der Apache Webserver am Standard Port nicht läuft kann dieser im Config File manuell eingestellt werden. Am einfachsten kann man das File durch klicken auf den Config button in der Zeile des Apache Web Servers erreichen.

* MySQL

In der MySQL Datenbank wurde ein Benutzer angelegt. Die Daten müssen mit dem der im File „settings.php“ konfigurierten Datenbank Verbindung übereinstimmen. Hierfür wurde mit phpMyAdmin ein neuer Datenbank Benutzer mit allen Rechten hinzugefügt.

### Verwendung

bWAPP, oder auch buggy web application eine absichtlich unsicher gehaltene Web Applikation. bWAPP hilft security Enthusiasten, Entwicklern und Studenten Web-Schwachstellen kennenzulernen bzw. zeigt auf wie diese Verhindert werden können. Es zeigt die wichtigsten Web-Schwachstellen auf, und enthalt auch die Risiken der OWASP Top 10. Auf der Seite können diese Sicherheits-Schwachstellen erforscht und getestet werden.

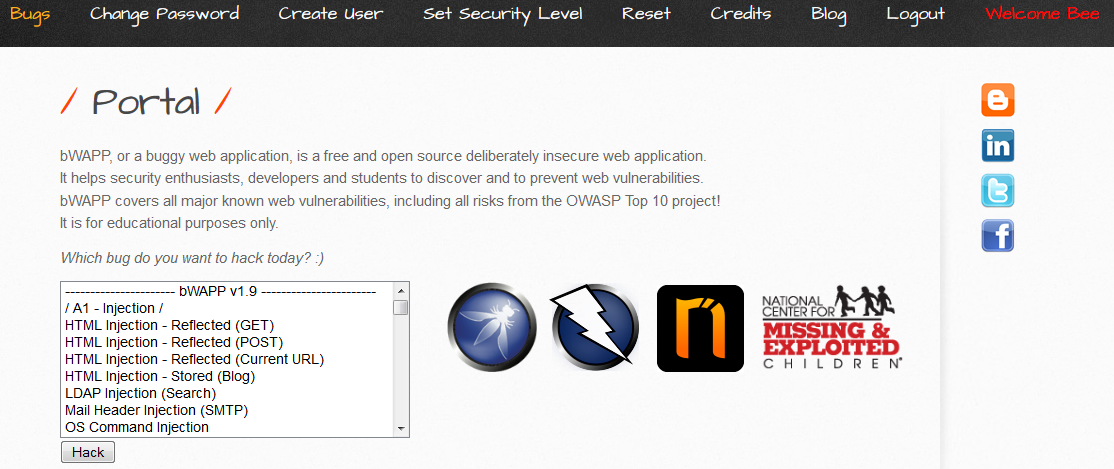


Abbildung 2: Web-Oberfläche der bWAPP

Unter Bugs können die gewünschten Sicherheitslücken ausgewählt werden und anschließend durch klicken des Buttons „Hack“ getestet werden. Beispielsweise wählt man in der Liste“ HTML Injection - Reflected (GET)“ wählt den Button „Hack“ um diese Sicherheitslücke zu testen.



Abbildung 3: Beispiel auf bWAPP HTML Injection

Man kann in hier die Sicherheitslücke testen und selbst ausprobieren. Anschließend werden einem die Ergebnisse seines „Hacks“ veranschaulicht.



Abbildung 4: Ergebnis der HTML Injection

Optional kann man die Sicherheit der bWAPP direkt auf der Website ändern. Dadurch wird die Seite sicherer und kann nicht mehr so leicht „gehackt“ werden.

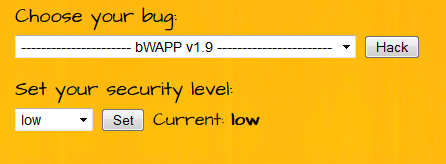


Abbildung 5: Security level der bWAPP

### Sicherheitslücken

* HTML Injection - Reflected (GET)
* HTML Injection - Reflected (POST)
* HTML Injection - Reflected (Current URL)
* HTML Injection - Stored (Blog)
* LDAP Injection (Search)
* Mail Header Injection (SMTP)
* OS Command Injection
* OS Command Injection - Blind
* PHP Code Injection
* Server-Side Includes (SSI) Injection
* SQL Injection (Search/GET)
* SQL Injection (Search/POST)
* SQL Injection (Search/CAPTCHA)
* SQL Injection (Select/GET)
* SQL Injection (Login Form)
* SQL Injection - Stored (Blog)
* SQL Injection - Stored (XML)
* SQL Injection - Blind (Search)
* SQL Injection - Blind (Web Services/SOAP)
* XML/XPath Injection (Search)
* XML/XPath Injection (Login Form)

**A2 - Broken Auth. & Session Mgmt.**

* Broken Authentication - Forgotten Function
* Broken Authentication - Insecure Login Forms
* Broken Authentication - Logout Management
* Broken Authentication - Password Attacks
* Broken Authentication - Weak Passwords
* Session Management - Administrative Portals
* Session Management - Cookies (HTTPOnly)
* Session Management - Cookies (Secure)
* Session Management - Session ID in URL
* Session Management - Strong Sessions

**A3 - Cross-Site Scripting (XSS)**

* Cross-Site Scripting - Reflected (GET)
* Cross-Site Scripting - Reflected (POST)
* Cross-Site Scripting - Reflected (JSON)
* Cross-Site Scripting - Reflected (AJAX/JSON)
* Cross-Site Scripting - Reflected (AJAX/XML)
* Cross-Site Scripting - Reflected (Back Button)
* Cross-Site Scripting - Reflected (Custom Header)
* Cross-Site Scripting - Reflected (Eval)
* Cross-Site Scripting - Reflected (HREF)
* Cross-Site Scripting - Reflected (PHP\_SELF)
* Cross-Site Scripting - Reflected (Referer)
* Cross-Site Scripting - Reflected (User-Agent)
* Cross-Site Scripting - Stored (Blog)
* Cross-Site Scripting - Stored (Change Secret)
* Cross-Site Scripting - Stored (Cookies)

**A4 - Insecure Direct Object References**

* Insecure DOR (Change Secret)
* Insecure DOR (Reset Secret)
* Insecure DOR (Order Tickets)

**A5 - Security Misconfiguration**

* Arbitrary File Access (Samba)
* Cross-Domain Policy File (Flash)
* Cross-Origin Resource Sharing (AJAX)
* Cross-Site Tracing (XST)
* Denial-of-Service (Slow HTTP DoS)
* Denial-of-Service (XML Bomb)
* Insecure FTP Configuration
* Insecure WebDAV Configuration
* Local Privilege Escalation (udev)
* Man-in-the-Middle Attack (HTTP)
* Man-in-the-Middle Attack (SMTP)
* Old/Backup & Unreferenced Files
* Robots File

**A6 - Sensitive Data Exposure**

* Host Header Attack (Reset Poisoning)
* HTML5 Web Storage (Secret)
* Insufficient Transport Layer Protection
* Text Files (Accounts)

**A7 - Missing Functional Level Access Control**

* Directory Traversal - Directories
* Directory Traversal - Files
* Host Header Attack (Cache Poisoning)
* Host Header Attack (Reset Poisoning)
* Remote & Local File Inclusion (RFI/LFI)
* Restrict Device Access
* Restrict Folder Access
* Server Side Request Forgery (SSRF)
* XML External Entity Attacks (XXE)

**A8 - Cross-Site Request Forgery (CSRF)**

* Cross-Site Request Forgery (Change Password)
* Cross-Site Request Forgery (Change Secret)
* Cross-Site Request Forgery (Transfer Amount)

**A9 - Using Known Vulnerable Components**

* PHP CGI Remote Code Execution
* PHP Eval Function

**A10 - Unvalidated Redirects & Forwards**

* Unvalidated Redirects & Forwards (1)
* Unvalidated Redirects & Forwards (2)

**Other bugs...**

* ClickJacking (Movie Tickets)
* Client-Side Validation (Password)
* HTTP Parameter Pollution
* HTTP Response Splitting
* HTTP Verb Tampering
* Information Disclosure - Favicon
* Information Disclosure - Headers
* Information Disclosure - PHP version
* Information Disclosure - Robots File
* Unrestricted File Upload