



Webslayer

How to brute force web applications

OWASP

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

Christian Martorella – Verizon Business
OWASP IBWAS 2010
Lisbon

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

Who am I?

- Practice Lead of Threat and Vulnerability Consulting at Verizon Business – EMEA
- Cofounder of Edge-Security
- President of FIST Conferences
- OPST, OPSA, CEH, CISSP, CISA, CISM
- OWASP Webslayer, Project Leader
- WhattheHack!, Source Conference, Hack.lu, OWASP Spain IV,VI, etc



Edge-Security

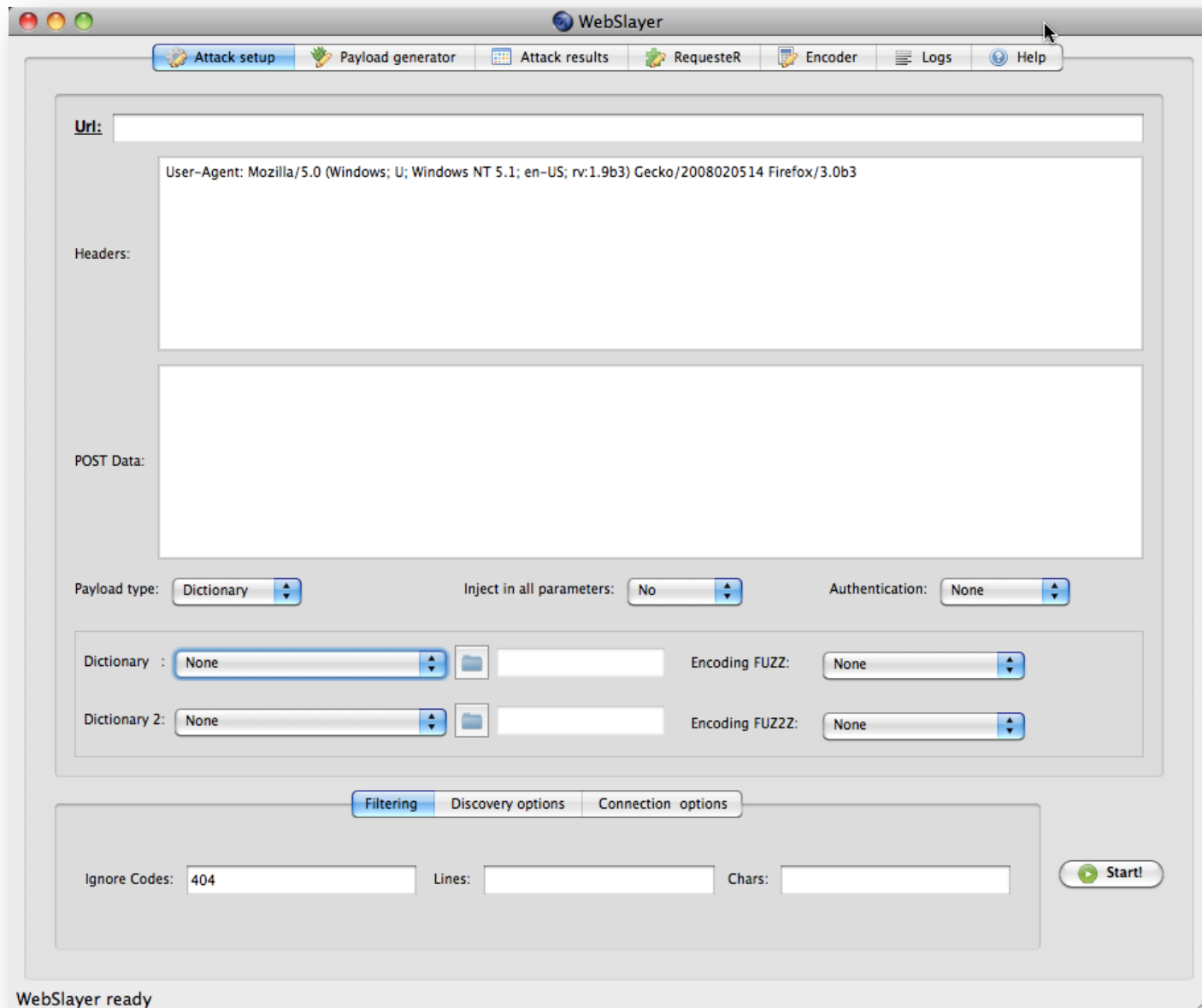
Contents

- Introduction
- Interface overview
- Payloads overview
- Basic discovery
- Working with the results
- Advanced discovery
- Login brute force
- Basic authentication brute force
- Local file inclusion abuse
- User-agent brute force
- Custom payload generation
- Advanced uses

Introduction

- Webslayer is a tool to perform brute force attacks on web applications
- It allows a security tester to brute force attacks of any kind in any part of the HTTP request (POST,GET, HEADERS, Authentication, etc)
- Is an enhancement of WFUZZ
- Multiplatform

Interface Overview

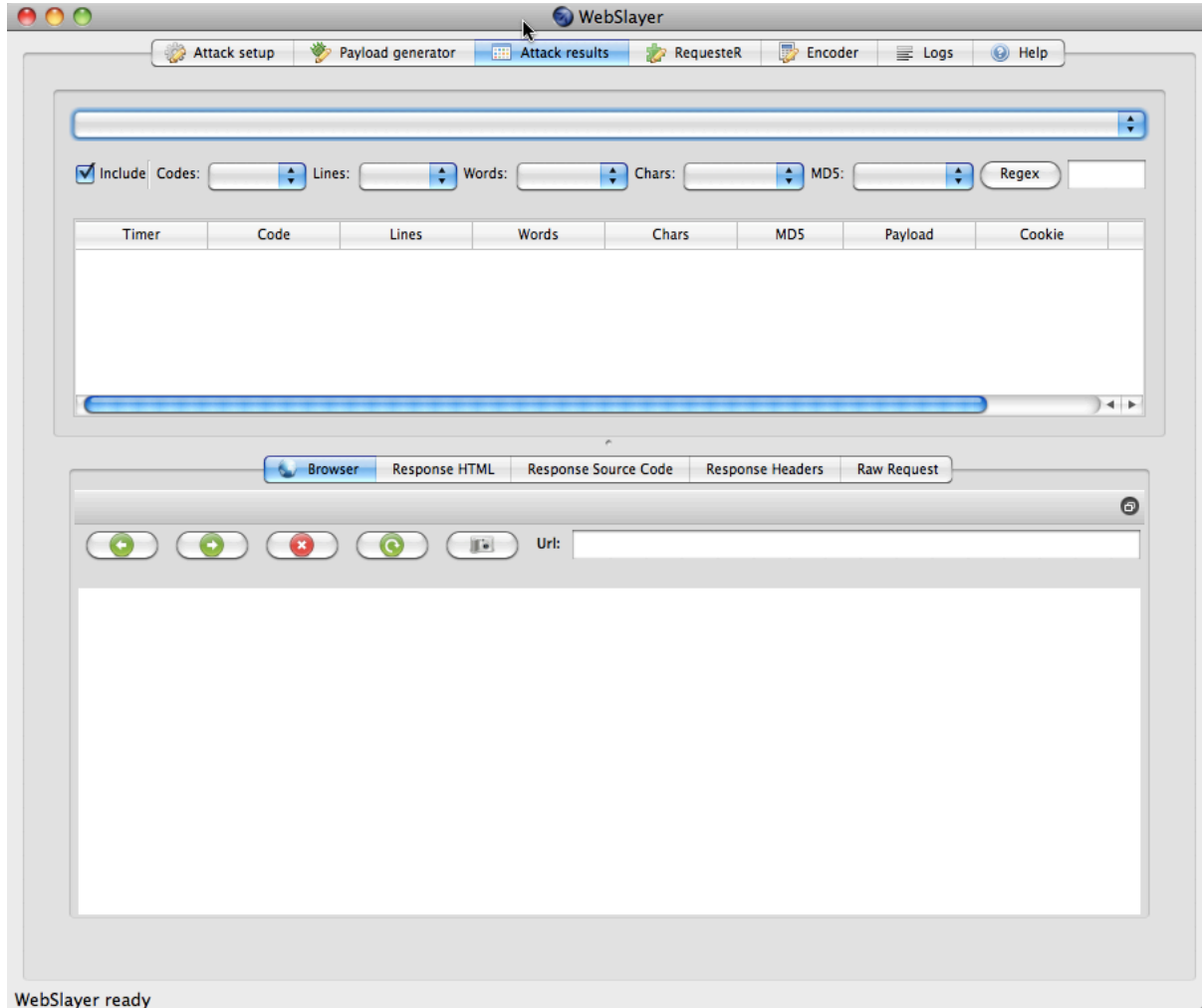


Interface Overview

The screenshot displays the OWASP WebSlayr Payload Generator interface. At the top, a navigation bar includes tabs for "Attack setup", "Payload generator" (which is active), "Attack results", "Requester", "Encoder", "Logs", and "Help". Below this, the "Payload generator" section is divided into tabs: "File", "Range", "Block", "Permutation", "Creditcards", and "Usernames". The "File" tab is selected, showing an "Add generator from:" label and a "Current encoding:" dropdown menu set to "None". A "File:" input field with a browse button is present. A text box titled "File load:" contains the instruction: "You can add a payload directly from a file, and apply an specific encoding. First select the encoding, then load the file." To the right of this section is a "Temporal Generators" area with a large empty box and a "Drop generator" button. On the far right, a "FINAL PAYLOAD:" section contains a large empty box. Below the main generator area, there are two tabs: "Payload Creator" (active) and "Payload Modifier". The "Payload Creator" tab shows a "Pattern:" input field and a "Generate PAYLOAD" button. On the right side of the interface, a vertical stack of buttons includes "Add from file", "Save Payload", "Drop Payload", and "Delete selection". At the bottom left, the text "WebSlayr ready" is displayed.

WebSlayr ready





Payloads overview

- We call payload to any list of strings that we can use to brute force the web application
- Most of the payloads are inherited from DIRB (www.open-labs.org)
- Now most of the payloads are included in FUZZDB
- Examples: common directory and files names, default installation files for different servers (jboss, apache, weblogic, etc), usernames, passwords, injections, etc

Use the force.

Basic discovery

- We are going to launch a basic directory discovery
 - Target: X.X.X.X
 - Payload: Dictionary (common)
- Filtering: 404 (will hide all the responses with 404)
- Non Standard Code detection:
 - ▶ Will try to determine what is the default error response, and will hide the responses that match this one.

Working with the results

The screenshot shows the WebSlayer application interface. The top navigation bar includes tabs for Attack setup, Payload generator, Attack results (active), Requester, Encoder, Logs, and Help. The main area displays the current attack configuration: `1 | http://localhost:8888/ejemplos/FUZZ | Dictionary | /Volumes/ZumoDrive-501/Development/webslayer-google/wordlist/general/common.txt`. Below this are filters for Include (checked), Codes, Lines, Words, Chars, MD5, and a Regex field.

	Timer	Code	Lines	Words	Chars	MD5	Payload	Cookie	Location
1	0.005925	401	14	58	492	6c0f54a80b025fb695908fb8d83523ca	Stats		
2	0.007668	301	9	30	334	4f61b794635f2155253f3e8b7222478d	cat		http://localhost:8888/ejemplos/cat/
3	0.001850	200	3	19	101	b1df2bc4e4b5d94a17cce2b529c072a3	db		
4	0.001883	401	14	58	492	6c0f54a80b025fb695908fb8d83523ca	stats		
5	0.004048	301	9	30	337	7f48c62fb872755b86bfc5d8dc8cd94a	upload		http://localhost:8888/ejemplos/upload/

Below the table, the Browser tab is active, showing the response for the URL `http://localhost:8888/ejemplos/upload/`. The browser content displays the title "Index of /ejemplos/upload" and a directory listing:

Name	Last modified	Size	Description
Parent Directory	-	-	-
up.php	12-Dec-2010 18:46	766	

At the bottom of the browser view, it says "Apache/2.0.63 (Unix) PHP/5.3.2 DAV/2 Server at localhost Port 8888".

Attack finished OK

Advanced discovery

■ We are going to play with the rest of settings:

- Threads
- NSC Detection (Non Standard Code)
- Filtering
- Recursion
- File extensions
- Proxies

Login Brute force

■ We are going to brute force a login form. We need to get a request template for the login.

1. Open firefox
2. Enable LiveHTTPHeaders
3. Perform a login
4. Copy the request information to Webslayer
5. Replace password value by FUZZ
6. Select a dictionary with common password
7. Launch the attack

■ hints: `common_pass.txt` `user:admin`

Basic authentication Brute force

- Now we are going to brute force the BASIC authentication that protects a directory.
 - Select Authentication: BASIC
 - Type the admin:FUZZ
 - Select Dictionary
 - Analyze results
- Hints: `common_pass.txt`

Local file inclusion Brute Force

- Suppose that we find a Local file inclusion vulnerability, and we want to search for valid files in the server.
- We can use a LFI dictionary, and launch the attack.
- Target: `/training/php_include.php?file=hello.html`

User-agent brute force

- Some applications can have a different set of functionalities depending on the user-agent.
- We can try to perform a brute force attack on the User-Agent and analyze for changes in the responses
- We are going to replace the User-Agent by FUZZ
 - User-Agent: FUZZ
- Select mobile-agents.txt from the root directory
- Analyze the results

Custom payload generation

The screenshot shows the 'WebSlayer' application window with the 'Payload generator' tab selected. The interface is divided into several sections:

- Navigation:** A top menu bar with tabs for 'Attack setup', 'Payload generator', 'Attack results', 'Requester', 'Encoder', 'Logs', and 'Help'.
- Generator Selection:** A row of tabs: 'File', 'Range', 'Block', 'Permutation', 'Creditcards', and 'Usernames'. The 'File' tab is active.
- Configuration:** 'Add generator from:' (empty) and 'Current encoding:' (set to 'None').
- File Input:** A 'File:' field with a browse button ('...').
- File Load Instructions:** A text box stating: 'File load: You can add a payload directly from a file, and apply an specific encoding. First select the encoding, then load the file.'
- Temporal Generators:** An empty vertical list box.
- Drop generator:** A button with a red minus sign and the text 'Drop generator'.
- Pattern Input:** A 'Pattern:' text field.
- Generate Payload:** A button with a green plus sign and the text 'Generate PAYLOAD'.
- Final Payload:** A large empty vertical box labeled 'FINAL PAYLOAD:'.
- Actions:** A vertical stack of buttons: 'Add from file', 'Save Payload', 'Drop Payload', and 'Delete selection'.

At the bottom left, the status bar reads 'WebSlayer ready'. At the bottom right, there is a logo for 'SP' and a globe icon.

Custom payload generation

■ We have the following options to create a payload:

- File
- Range
- Block
- Permutation
- Credit Cards
- Usernames

■ Patterns -> Final Payload

Custom payload generation

- Let 's create a payload with the following pattern:
 - ▶ admin-001
 - ▶ admin-020
 - ▶ guest-001

Encoded parameters brute force

Many times we see parameters encoded with different methods like MD5, base64, we can easily brute force this parameters with webslayer

■ Target:

/training/encoded.php?

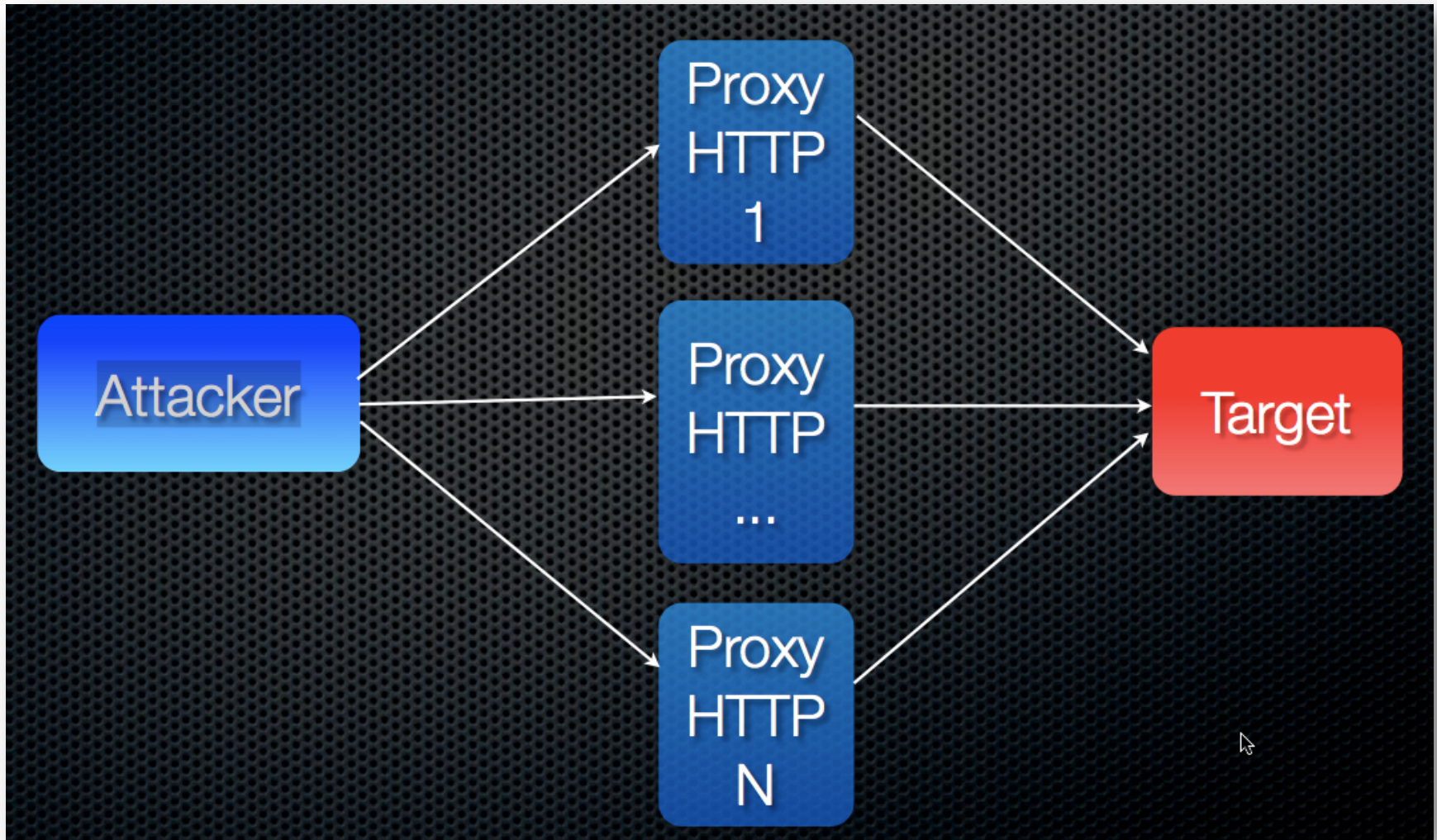
var=126b7016a916a4b178dd72b947c15123

■ Hint: Encode the payload, test

Advanced uses & techniques

- Finding the same file in different servers
- Finding a file in different directories
- Multiple servers discovery
- Proxy discovery
- Source ip balancing
- Random order (diagonal, horizontal, vertical)
 - FUZZ2Z-FUZZ Horizontal

Source ip balancing



Advanced techniques

- **Horizontal scanning:** we try un password for all the users
- **Diagonal scanning:** different username/ password (randomized)
- **Three dimension:** (H,V,D) + Time
- **Four dimension:** (H,V,D) + Time + Balancing source IP

Question, ideas?

?

**"MAY THE FORCE
BE WITH YOU"**



Thank you

Christian Martorella
Webslayer Project Leader
Verizon Business

Cmartorella at edge-security.com

OWASP

16/12/2010

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

The OWASP Foundation

<http://www.owasp.org>