Web Application Firewalls: What the vendors do NOT want you to know

Sandro Gauci
EnableSecurity
sandro@enablesecurity.com

Wendel G. Henrique
Trustwave
whenrique@trustwave.com

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

- PT Consultant at Trustwave's SpiderLabs.
- Over 7 years in the security industry.
- Vulnerability discovery Webmails, AP, Citrix, etc.
- Spoke in YSTS 2.0, Defcon 16, H2HC and others.
- Affiliated to Hackaholic team.
$ whois SandroGauci

- Founder and CSO EnableSecurity.
- VOIPPACK (CANVAS addon).
- Security research papers.
- SIPVicious and SurfJack.
- Over 9 years in the security industry.
Introduction

- **WAF - Web Application Firewall.**

- Can be identified, detected.

- Security software is not necessarily secure.
What is WAF

- WAFs are often called 'Deep Packet Inspection Firewall'.

- Some WAFs look for attack signatures while others look for abnormal behavior.

- WAFs products: software or hardware appliance.
What is WAF

- WAFs can be installed as a reverse proxy, embedded or connected in a switch (SPAN or RAP).

- Nowadays many WAF products detect both inbound and outbound attacks.
Who uses WAF?

- Many banks around the world.
- Companies which need high protection.
- Many companies in compliance with PCI DSS (Payment Card Industry - Data Security Standard).
Operation Modes:

- Negative model (blacklist based).
- Positive model (whitelist based).
- Mixed / Hybrid (mix negative and positive model protection).
Operation Mode: Negative

A negative security model recognize attacks by relying on a database of expected attack signatures.

Example:

Do not allow in any page, any argument value (user input) which match potential XSS strings like `<script>`, `</script>`, `String.fromCharCode`, etc.
Operation Mode: Positive

A positive security model enforces positive behavior by learning the application logic and then building a security policy of valid known good requests.

Example:

Page news.jsp, the field "id" only accept numbers [0-9] and starting at 0 until 65535.
Common Weaknesses

- Bad design.
- Bad implementation.
- Vulnerable to the same flaws they intend to protect.
Detection

- Cookies: Some WAF products add their own cookie in the HTTP communication.
demo
Detection

- Header Rewrite: Some WAF products allow the rewriting of HTTP headers. The most common field is "Server", this is used to try to deceive the attackers (server cloaking).

**Example:**

Connection might be changed to Cneonction or nnCoection.
demo
Detection

- Different 404 error codes for hostile and non-existent pages.

- Different error codes (404, 400, 401, 403, 501, etc) for hostile parameters (even non-existent ones) in valid pages.
demo
Detection

WAF systems leave several signs which permit us to detect them, one of them are Drop Connection:

Example:

Drop Action: Immediately initiate a "connection close" action to tear down the TCP connection by sending a FIN packet.
Detection

- WAF systems leave several signs which permit us to detect them, one of them are Pre Built-in Rules:

- Pre Built-in Rules: All (at least all that we know) WAF systems have a built-in group of rules in negative mode, these rules are different in each products, this can help us to detect them.
Detection

- You should be thinking...

- It’s so boring.

- We have to know a lot of products to identify them correctly.

- What about create a tool for that?
WAFW00F

- That’s our answer for your prays:

- Detect over 20 different WAF products.

- Do not stop at the first WAF system found.

- Follow HTTP redirects to identify more systems.

- Much more coming soon.
waffun obscure$ python wafw00f.py --help

^      ^
 ///7/ /.' \ / __///7/ /,' \ ,' \ / __/
 | V V // o // _/ | V V // 0 // 0 // _/
 |_n_,'/_n_//_/  |_n_,' \_,' \_,'/_/
 <
 ...

WAFW00F - Web Application Firewall Detection Tool

Usage: wafw00f.py url1 [url2 [url3 ... ]]
example: wafw00f.py http://www.victim.org/

Options:
  -h, --help           show this help message and exit
  -v, --verbose        enable verbosity - multiple -v options increase
                        verbosity
  -a, --findall        Find all WAFs, do not stop testing on the first one
  -r, --disableredirect Do not follow redirections given by 3xx responses

9-6:waffun obscure$
demo
Bypassing

- WAF systems can be bypassed by changing the attack to do not match the rules:
  - Detect allowed / denied strings.
  - Detect sequences of good and bad strings together.
  - Modify your attack to match the good rules.
Bypassing

- WAF systems allow us to bypass them in different ways, one of them are using old tricks like encoding and language support:

- Spaces, comments, case sensitive mutation, Unicode, etc.

- The web server can parse, decode and interpret and HTTP request differently from the WAF.
Bypassing

WAF systems allow us to bypass them in different ways, one of them are using the flexibility of the web languages:

- HTML and JS is very flexible.

Example:

XSS Case.
demo
Bypassing

■ WAIT!

■ What about positive model?

■ They are really secure?

■ If we find a positive model we should give up?
demo
Bypassing

- You should be thinking...
- It’s time consuming.
- The are so much different techniques to remember.
- There are so many specific techniques product dependent.
- What about a tool for that?
WAFFUN

- That’s our answer for your prays:

- Test the target and point weakness in the WAF system.

- Use with WAFW00F for better results.

- Working in Windows and Unix.

- Beta version! We need the community help.
demo
Other Vulnerabilities

- XSS (in the own WAF system?)
- Overflows
- DoS
demo
Thank you!

- Do you have ideas / resources to improve our tools?

- Do you just don’t have with who talk?

  wsguglielmetti [em] gmail [ponto] com

  sandro [em] enablesecurity [ponto] com