An Introduction to ModSecurity and the OWASP Core Rule Set

(OWASP Hamburg)

Christian Folini / @ChrFolini
Safety Belts
Baseline / 1st Line of Defense
Boring Bio

- Christian Folini / @ChrFolini
- Security Engineer at netnea in Switzerland
- Author, teacher and speaker
- OWASP CRS project Co-Lead
Plan for Today

- What is a WAF?
- What is ModSecurity?
- What is Core Rule Set?
- Demo
- Key concepts
- Rules
- False Positives
Web Application Firewalls

Complex • Overwhelming • Rarely Functional
ModSecurity

Embedded • Rule oriented • Granular Control
Based upon a true story!

CRS3
OwASP ModSecurity Core Rule Set v3.0

Directed by
Chaim Sanders

Starring
Walter Hop as Regex Wizard, Chaim Sanders

[Other credits and production details]

Coming soon to a server near you!
Demo Time (Installation)

Clone the repository (or download latest release):

```
$> git clone https://github.com/coreruleset/coreruleset
```

Copy the example config:

```
$> cp crs-setup.conf.example crs-setup.conf
```

Include in server config (depending on path):

Include `/path-to-owasp-crs/crs-setup.conf`

Include `/path-to-owasp-crs/rules/*.conf`
Research based on 4.5M Burp requests.

**Burp vs. OWASP ModSecurity Core Rule Set 3.0**

**CRS3 Default Install**

- **Redir.:** 0%
- **RFI:** 0%
- **LFI:** -100%
- **XSS:** -82%
- **SQLi:** -100%
Paranoia Levels

Paranoia Level 1: Minimal number of false positives  
*Baseline protection*

Paranoia Level 2: More rules, some false positives  
*Real data in the service*

Paranoia Level 3: Specialized rules, more FPs  
*Online banking level security*

Paranoia Level 4: Crazy rules, many FPs  
*Nuclear power plant level security*
Important Groups of Rules

Request Rules

REQUEST-910-IP-REPUTATION.conf
REQUEST-911-METHOD-ENFORCEMENT.conf
REQUEST-912-DOS-PROTECTION.conf
REQUEST-913-SCANNER-DETECTION.conf
REQUEST-920-PROTOCOL-ENFORCEMENT.conf
REQUEST-921-PROTOCOL-ATTACK.conf

REQUEST-930-APPLICATION-ATTACK-LFI.conf
REQUEST-931-APPLICATION-ATTACK-RFI.conf
REQUEST-932-APPLICATION-ATTACK-RCE.conf
REQUEST-933-APPLICATION-ATTACK-PHP.conf
REQUEST-941-APPLICATION-ATTACK-XSS.conf
REQUEST-942-APPLICATION-ATTACK-SQLI.conf
REQUEST-943-APPLICATION-ATTACK-SESS-FIX.conf
REQUEST-944-APPLICATION-ATTACK-JAVA.conf

REQUEST-949-BLOCKING-EVALUATION.conf
Important Groups of Rules

Response Rules

RESPONSE-950-DATA-LEAKAGES.conf
RESPONSE-951-DATA-LEAKAGES-SQL.conf
RESPONSE-952-DATA-LEAKAGES-JAVA.conf
RESPONSE-953-DATA-LEAKAGES-PHP.conf
RESPONSE-954-DATA-LEAKAGES-IIS.conf

RESPONSE-959-BLOCKING-EVALUATION.conf
Paranoia Level

Example: Protocol Enforcement Rules

Paranoia Level 1: 31 Rules
Paranoia Level 2: 7 Rules
Paranoia Level 3: 1 Rules
Paranoia Level 4: 4 Rules
Stricter Siblings

Example: Byte Range Enforcement

Paranoia Level 1:
  Rule 920270: Full ASCII range without null character

Paranoia Level 2:
  Rule 920271: Full visible ASCII range, tab, newline

Paranoia Level 3:
  Rule 920272: Visible lower ASCII range without %

Paranoia Level 4:
  Rule 920273: A-Z a-z 0-9 = - _ . , : &
Anomaly Scoring

Adjustable Limit • Blocking Mode • Iterative Tuning
Sampling Mode

Easing into CRS adoption / limit the impact

- Define a sampling rate of n
- Only n% of the requests are being funneled into CRS3
- 100% - n% of requests bypass CRS3
- Monitor performance and fix problems
- Slowly raise n in an iterative way until it reaches 100%
Fastly WAF rule set updates and maintenance

Fastly provides rule set updates to the [Fastly WAF](https://www.fastly.com) in a prompt manner to help protect customers against attacks.

For OWASP and Trustwave rules changes we use the following process:

1. We regularly review the rule changes as they happen in both the OWASP Core Rule Set and the Trustwave Rule Set.
2. We translate the rules into [Varnish Configuration Language (VCL)](https://varnish.apache.org/tapir/docs/vcl/) to run inside our cache nodes.
3. We test the rules in our platform to ensure they perform adequately. We try to maximize performance and rule efficacy while reducing false positives.
4. We correct bugs, if any are found.
5. We propagate the rule set changes to our platform worldwide.
6. Finally, we will provide customers with a notification and [instructions on how to make rule updates](https://www.fastly.com/docs/guides/security/).
## Add managed rule groups

Managed rule groups are created and maintained for you by AWS and AWS Marketplace sellers.

### AWS managed rule groups

<table>
<thead>
<tr>
<th>Name</th>
<th>Capacity</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admin protection</td>
<td>100</td>
<td>Add to web ACL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set rules action to count</td>
</tr>
<tr>
<td>Amazon IP reputation list</td>
<td>25</td>
<td>Add to web ACL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set rules action to count</td>
</tr>
<tr>
<td>Core rule set</td>
<td>700</td>
<td>Add to web ACL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set rules action to count</td>
</tr>
<tr>
<td>Known bad inputs</td>
<td>200</td>
<td>Add to web ACL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set rules action to count</td>
</tr>
<tr>
<td>Linux operating system</td>
<td>200</td>
<td>Add to web ACL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set rules action to count</td>
</tr>
<tr>
<td>PHP application</td>
<td>100</td>
<td>Add to web ACL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set rules action to count</td>
</tr>
</tbody>
</table>
Azure Web Application Firewall on Azure Application Gateway

11/14/2019 • 8 minutes to read • .Compiler and 2.2.9 from the Open Web Application Security Project (OWASP). The WAF automatically updates to include protection against new vulnerabilities, with no additional configuration needed.
Tightly Integrated into the Oracle Cloud Infrastructure Console
Managing the OWASP rule set in the WAF

Cloudflare provides three sensitivity settings for the OWASP rule set: high, medium, and low. The table
January 29, 2019

Running a multi-tenant WAF at the edge

By Reed Morrison, Software Developer

Web Application Firewalls (WAFs) are a critical layer in modern web security, providing a website’s first line of defense against vulnerabilities. WAFs can be used to defend against and notify on attempted exploits, allowing for mitigations faster than organizations can patch vulnerable software. For a global CDN, this functionality must be implemented in a way that is sensitive to performance, providing response times on the order of milliseconds. When we first introduced a WAF engine to the Verizon Digital Media Services stack three years ago, we selected the ModSecurity Rules Engine, which we found to be first-rate for individual WAF use cases. Furthermore, ModSecurity’s support of the OWASP Core Rule Set (CRS), powerful rule language, and API access to the HTTP traffic stream in real time offered significant flexibility.

Enter waf1z

However, as the number of customers using the WAF increased, we experienced performance and resource bottlenecks. ModSecurity’s dense ruleset propagated across every customer instance drove memory and CPU utilization up across our network, increasing operational costs. Additionally, testing and deploying new rules was difficult because the rule language was often unwieldy and difficult to write and parse. These issues, along with development complexity with the existing ModSecurity library, led to the development of waf1z, an open source WAF engine, published under the Apache 2.0 license.

For Verizon Digital Media Services, waf1z is a significant improvement on ModSecurity because:

- **It consumes less memory.**
- **Offers better performance.**
- **Is API-driven.**

Waf1z supports a subset of ModSecurity capabilities, the OWASP Core rulesets 2.x and 3.x, and several third-party rulesets.
False Positives

False Positives are expected from PL2

- FPs are fought with rule exclusions
- Tutorials at https://www.netnea.com
- Get cheatsheet from Netnea
- Please report FPs at PL1 (github)
Apache / ModSecurity / CRS Tutorials

https://www.netnea.com/cms/apache-tutorials/

- Tutorial 1: Compiling Apache (Video Walk-Through)
- Tutorial 2: Configuring a Minimal Apache Web Server
- Tutorial 3: Configuring an Apache/PHP Application Server
- Tutorial 4: Enabling Encryption with SSL/TLS
- Tutorial 5: Extending and Analyzing the Access Log
- Tutorial 6: Embedding ModSecurity
- Tutorial 7: Including OWASP ModSecurity Core Rule Set
- Tutorial 8: Handling False Positives with the OWASP ModSecurity Core Rule Set
- Tutorial 9: Setting up a Reverse Proxy Server
- Tutorial 10: Efficiently Configuring and Debugging Apache and ModSecurity in the Shell
- Tutorial 11: Visualization of Apache / ModSecurity log information
- Tutorial 12: Capturing and Decrypting the Entire Traffic
ModSecurity / CRS Courses

- Offered at https://netnea.com
- 1 seat to give away for free for next week, April 22 / 23

US Time-Zone (15:00 - 23:00 CET)
Summary ModSecurity & CRS3

• 1st Line of Defense against web application attacks

• Generic set of blacklisting rules for WAFs

• Blocks 80% of web application attacks in the default installation (with a minimal number of FPs)

• Granular control over the behaviour down to the parameter level

More information at https://coreruleset.org
Questions and Answers, Contact

Contact: christian.folini@netnea.com

@ChrFolini