

## Introduction

WordPress is a state-of-the-art semantic personal publishing platform with a focus on aesthetics, web standards, and usability. Unfortunately it is also missing the vital security functions that protect the application from malicious attacks. A default install of WordPress is not as secure as Web Application Security Professionals would like, hence the need for extra layers of defence to ensure that the application remains secure at all times.

## What is ModSecurity and why do I need it?

ModSecurity works as a layer of defence between the Web Server and the application and works on a series of rules that define how it needs to react to certain request types and behaviour. With an ever increasing attack rate aimed at web applications, ModSecurity helps add an external security layer that increases security, detects, and prevents attacks before they reach web applications.

## Installing ModSecurity

This guide won't go into the details of installing ModSecurity as there are many excellent guides already out there. If you are not already running ModSecurity and would like to know more, we suggest you visit:

- <http://modsecurity.org/documentation/index.html>
- <http://atomicplayboy.net/blog/2005/01/30/an-introduction-to-mod-security/>

## General Configuration

The following configuration settings are included as part of the standard install, so ensure that yours look similar to the ones below.

```
SecFilterEngine On
SecServerSignature "Go Away"
SecFilterCheckURLEncoding On
SecFilterCheckUnicodeEncoding Off
SecFilterForceByteRange 1 255
SecAuditEngine RelevantOnly
SecAuditLog /var/log/httpd/modsec_log
SecFilterScanPOST On
SecFilterDefaultAction "deny,log,status:500"
```

## Additional Web Application Security Rules

Whilst not directly related to WordPress, the following rules prevent known attacks and malicious activity and therefore increase the overall security to your blog.

```
# Do not accept GET or HEAD requests with bodies
SecFilterSelective REQUEST_METHOD "^(GET|HEAD)$" chain
SecFilterSelective HTTP_Content-Length "!^$"
# Require Content-Length to be provided with
# every POST request
```

```
SecFilterSelective REQUEST_METHOD "^POST$" chain
SecFilterSelective HTTP_Content-Length "^$"

# Generic SQL Injection Prevention
SecFilter "delete[[:space:]]+from"
SecFilter "insert[[:space:]]+into"
SecFilter "select.+from"
SecFilterSelective ARGS "(or.+1[[:space:]]*=[[:space:]]1|(or 1=1|'+)--)"
"\"id:300014,rev:1,severity:2,msg:'Generic SQL injection protection'"
SecFilterSelective ARGS
"\"((alter|create|drop)[[:space:]]+(column|database|procedure|table)|delete[[:space:]]+from|update.
+set.+)=)" \"id:300015,rev:1,severity:2,msg:'Generic SQL injection protection'"

# PHP
#Protecting from XSS attacks through the PHP session cookie
SecFilterSelective ARG_PHPSESSID "!^[0-9a-z]*$"
SecFilterSelective COOKIE_PHPSESSID "!^[0-9a-z]*$"
SecFilterSelective REQUEST_URI
"\"(&(cmd|command)=(id|uname)\\x20|cmd\\?(cmd|command)=|(spy|cmd|cmd_out|sh)\\. (gif|jpg|png|
g|bmp|txt)\\?&(cmd|command)=|\\.php\\?(cmd|command)=)"

#Generic PHP exploit signatures
SecFilterSelective POST_PAYLOAD|REQUEST_URI "<\\?php
(chr|fwrite|fopen|system|chr|passthru|popen|proc_open|shell_exec|exec|proc_nice|proc_termin
ate|proc_get_status|proc_close|psockopen|leak|apache_child_terminate|posix_kill|posix_mkfifo|
posix_setpgid|posix_setsid|posix_setuid|phpinfo)\\(.*)\\;" \"id:330002,rev:1,severity:2,msg:'Generic
PHP exploit pattern denied'"

#Generic PHP remote file injection
SecFilterSelective REQUEST_URI "!(/do_command)" chain
SecFilterSelective REQUEST_URI "\\ .php\\?.*= (https?|ftp) \\. *(cmd|command) ="
#General [url] php forum protections (phpbb and others, to protect against script injection attacks in
url links)
SecFilterSelective THE_REQUEST "\\ .php\\?" chain
SecFilter "\\ [url=(script|javascript|applet|about|chrome|activex) \\. * \\. * \[ /url \]"

#Generic PHP exploit signatures
SecFilterSelective THE_REQUEST
"\"(chr|fwrite|fopen|system|e?chr|passthru|popen|proc_open|shell_exec|exec|proc_nice|proc_termin
ate|proc_get_status|proc_close|psockopen|leak|apache_child_terminate|posix_kill|posix_mkfi
fo|posix_setpgid|posix_setsid|posix_setuid|phpinfo)\\(.*)\\;"
"\"id:330001,rev:1,severity:2,msg:'Generic PHP exploit pattern denied'"
```

As with any hardening guide, the above rules may prevent other applications from functioning on the server. We are not to be held responsible for anything, no matter what happens.

## ModSecurity Specific Rules for WordPress

The rules discussed in this article related to the 1.9.x version of ModSecurity. As time permits, we will update the document to reflect the 2.x release of ModSecurity. The rules below are not a definitive list of rules, but are offered as a guideline on stopping known attack vectors and methods aimed at WordPress installations only. As with any software, new vulnerabilities are discovered all the time so ensure you keep on checking the site for any new rules.

```
# Add access control to the login page (nb remember to change the IP address!)
SecFilterSelective REMOTE_ADDR "!192\.\168\.\0\.\69. chain
SecFilterSelective REQUEST_URI "/wp-login.php" \
log,deny,redirect:http://www.yoursite.com/nologin.html
# WordPress SQL injection vulnerability prevention
SecFilterSelective REQUEST_URI "/index\.php" chain
SecFilterSelective ARG_poll|ARG_category|ARG_ctg
"((select|grant|delete|insert|drop|alter|replace|truncate|update|create|rename|describe)[[:space:]]+[A-Z|a-z|0-9|\*| |,|]+[[:space:]]+(from|into|table|database|index|view)[[:space:]]+[A-Z|a-z|0-9|\*| |,|])\|'|UNION.*SELECT.*INTO.*FROM)"
SecFilterSelective REQUEST_URI "/wp-trackback\.php\?tb_id=*(select|grant|delete|insert|drop|do|alter|replace|truncate|update|create|rename|describe)[[:space:]]+[A-Z|a-z|0-9|\*| |,|]+[[:space:]]+(from|into|table|database|index|view)"
SecFilterSelective REQUEST_URI "/wp-trackback\.php" chain
SecFilterSelective ARG_tb_id
"(select|grant|delete|insert|drop|do|alter|replace|truncate|update|create|rename|describe)[[:space:]]+[A-Z|a-z|0-9|\*| |,|]+[[:space:]]+(from|into|table|database|index|view)"
SecFilterSelective REQUEST_URI
"/index\.php\?cat=.*(select|grant|delete|insert|drop|do|alter|replace|truncate|update|create|rename|describe)[[:space:]]+[A-Z|a-z|0-9|\*| |,|]+[[:space:]]+(from|into|table|database|index|view)"
SecFilterSelective REQUEST_URI "wp-pass\.php\?_\ wp_\ http_\ referer="
SecFilterSelective HTTP_USER_AGENT "WordPress Hash Grabber"

#WordPress cat vulnerability prevention
SecFilterSelective REQUEST_URI "/WordPress/" chain
SecFilterSelective ARG_cat "!^[0-9]*$"

#WordPress shell injection vulnerability
SecFilterSelective REQUEST_URI "/cache/user.*\.\.php\?cmd="
"id:390064,rev:1,severity:2,msg:'JITP: WordPress shell injection Vulnerability'"

#WordPress "cache_lastpostdate" PHP code insertion prevention
SecFilterSelective ARG_cache_lastpostdate "<?\?php"

#WordPress SQL injection and feed path disclosure vulnerability prevention
SecFilterSelective REQUEST_URI "/\?feed=\rss2\&p=-1"
SecFilterSelective REQUEST_URI "/wp\WordPress\?\?feed=\rss2\&p=-1"

# WordPress Experimental XML-RPC generic attack sigs
SecFilter "\\\|'\\|\\|;"
SecFilter "\<param\>\<name\>.*\\|";
```

```
SecFilter "(\\<xml|\\<.*xml)" chain
SecFilter "(echo(
|\\(|\\').*\\;|chr|fwrite|fopen|system|echr|passthru|popen|proc_open|shell_exec|exec|proc_nice|pr
oc_terminate|proc_get_status|proc_close|pfsockopen|leak|apache_child_terminate|posix_kill|posi
x_mkfifo|posix_setpgid|posix_setsid|posix_setuid|phpinfo)\\(.*)\\;" chain
SecFilter "methodCall\\>"

# WordPress Specific XML-RPC attacks on xmlrpc.php
SecFilterSelective THE_REQUEST "(/xmlrpc|. *xmlrpc_services)\\.php" chain
SecFilter "(\\<xml|\\<.*xml)" chain
SecFilter "(echo(
|\\(|\\').*\\;|chr|fwrite|fopen|system|echr|passthru|popen|proc_open|shell_exec|exec|proc_nice|pr
oc_terminate|proc_get_status|proc_close|pfsockopen|leak|apache_child_terminate|posix_kill|posi
x_mkfifo|posix_setpgid|posix_setsid|posix_setuid|phpinfo)\\(.*)\\;"

# WordPress XML-RPC SQL injection generic signature
SecFilterSelective THE_REQUEST "(/xmlrpc|. *xmlrpc_services)\\.php" chain

SecFilter
"<methodName>.*</methodName>.*<value><string>.*(select|grant|delete|insert|drop|do|alter|re
place|truncate|update|create|rename|describe)[[:space:]]+[A-Z|a-z|0-9|\\*|
|,]+[[:space:]](from|into|table|database|index|view). *methodName\\>"

# Prevent WP Theme Based XSS: WordPress Search
SecFilterSelective THE_REQUEST "\\?s=" chain
SecFilter "<[[:space:]]*(script|javascript|applet|about|chrome|activex|object|iframe|img)"

# WordPress XML-RPC generic attack sigs
SecFilterSelective POST_PAYLOAD "^Content-Type\\: application/xml" chain
```

## Conclusion

Installing and using ModSecurity will add the necessary extra layer of security that is needed to ensure your blog remains secure. We plan to develop this whitepaper as we go. Feedback and comments welcome. A special thanks to Daniel Cuthbert who was the primary author of this Whitepaper and for his great research and contributions to the BlogSecurity project.

## Credits

Author: Daniel Cuthbert <http://danielcuthbert.com/>

Co-Author: David Kierznowski <http://blogsecurity.net>