nginx waf 安装配置

yum install curl-devel pcre-devel zlib-devel openssl-devel

wget <https://nginx.org/download/nginx-1.18.0.tar.gz>

wget <https://github.com/SpiderLabs/ModSecurity/releases/download/v3.0.4/modsecurity-v3.0.4.tar.gz>

//国内地址

http://www.modsecurity.cn/download/modsecurity/modsecurity-v3.0.4.tar.gz

wget <https://www.openssl.org/source/openssl-1.1.1h.tar.gz> // 启用 tls 1.3 非必须

git clone <https://github.com/SpiderLabs/ModSecurity-nginx>.git

#wget <https://github.com/SpiderLabs/ModSecurity-nginx/releases/download/v1.0.1/modsecurity-nginx-v1.0.1.tar.gz>

tar -xf modsecurity-v3.0.4.tar.gz

cd modsecurity-v3.0.4

./configure --prefix=/usr/local/modsecurity

make && make install

tar -xf openssl-1.1.1h.tar.gz

tar -xf modsecurity-nginx-v1.0.1.tar.gz

tar -xf nginx-1.18.0.tar.gz

sed -i -e 's/1.18.0//g' -e 's/nginx\//KITT/g' -e 's/"NGINX"/"KITT"/g' src/core/nginx.h

//隐藏版本号

./configure --prefix=/usr/local/nginx --with-http\_ssl\_module --with-http\_v2\_module --with-stream --without-http\_fastcgi\_module --without-http\_uwsgi\_module --without-http\_scgi\_module --add-dynamic-module=../ModSecurity-nginx-master

make && make install

wget <https://github.com/SpiderLabs/owasp-modsecurity-crs/archive/v3.2.0.tar.gz>

//国内地址

http://www.modsecurity.cn/download/corerule/owasp-modsecurity-crs-3.3-dev.zip

tar -zxvf v3.0.2.tar.gz

cd owasp-modsecurity-crs-3.0.2

mkdir –p /usr/local/nginx/conf/modsecurity

cp modsecurity-v3.0.4/modsecurity.conf-recommended /usr/local/nginx/conf/modsecurity/modsecurity.conf

cp modsecurity-v3.0.4/unicode.mapping /usr/local/nginx/conf/modsecurity

cp -r owasp-modsecurity-crs-3.2.0/rules /usr/local/nginx/conf/modsecurity/csr/

cp owasp-modsecurity-crs-3.2.0/crs-setup.conf.example /usr/local/nginx/conf/modsecurity/csr/crs-setup.conf

编辑主配置文件

//before event

load\_module modules/ngx\_http\_modsecurity\_module.so;

//server or location

 modsecurity on;

modsecurity\_rules\_file modsecurity/main.conf;

//main.conf

include modsecurity/modsecurity.conf

|  |  |
| --- | --- |
|  |  |
|  | SecRule FILES "(?i)\.php$" "t:none,log,redirect:http://cyberis.co.uk,msg:'PHP file upload blocked',id:1" |
|  |  |
|  | # Deny when a semi-colon is found in the 'ip' parameter |
|  | SecRule ARGS:ip ";" "t:none,log,deny,msg:'semi colon test',id:2" |
|  |  |
|  |  |
|  | # Disable ModSecurity for a given page - NOTE THE ESCAPE OF THE '?' |
|  | SecRule REQUEST\_URI "/joomla/index.php/component/users/\?task=registration.register" "allow,id:3" |
|  |  |
|  |  |
|  | # Fix a rule (add an exception for the password field) |
|  | SecRule REQUEST\_COOKIES|!REQUEST\_COOKIES:/\_\_utm/|REQUEST\_COOKIES\_NAMES|ARGS\_NAMES|ARGS|!ARGS:/jform\[password[12]\]/|XML:/\* "(^[\"'`´’‘;]+|[\"'`´’‘;]+$)" "phase:2,rev:'2',ver:'OWASP\_CRS/2.2.8',maturity:'9',accuracy:'8',capture,t:none,t:urlDecodeUni,block,msg:'SQL Injection Attack: Common Injection Testing Detected',id:'4',logdata:'Matched Data: %{TX.0} found within %{MATCHED\_VAR\_NAME}: %{MATCHED\_VAR}',severity:'2',tag:'OWASP\_CRS/WEB\_ATTACK/SQL\_INJECTION',tag:'WASCTC/WASC-19',tag:'OWASP\_TOP\_10/A1',tag:'OWASP\_AppSensor/CIE1',tag:'PCI/6.5.2',setvar:'tx.msg=%{rule.msg}',setvar:tx.sql\_injection\_score=+%{tx.critical\_anomaly\_score},setvar:tx.anomaly\_score=+%{tx.critical\_anomaly\_score},setvar:tx.%{rule.id}-OWASP\_CRS/WEB\_ATTACK/SQL\_INJECTION-%{matched\_var\_name}=%{tx.0}" |
|  | #...then disable the rule |
|  | SecRuleRemoveById 981318 |
|  |  |
|  | # An alternative way of donig the same thing (preferred approach) - update an existing rule. NB: target refers to the data which a pattern match is performed against, i.e. the second parameter in a rule: |
|  | SecRuleUpdateTargetById 981318 "!ARGS:/jform\[password[12]\]/" |
|  |  |
|  | # white-list the user parameter for rule #981260 only when the REQUEST\_URI is /index.php... |
|  | SecRule REQUEST\_URI "@beginsWith /index.php/component/users/" "id:5,phase:1,t:none,pass,nolog,ctl:ruleRemoveTargetById=981318;ARGS:/jform\[password[12]\]/" |
|  |  |
|  | # Only allow letters in username field - note the use of phase:2 here, as you are inspecting the request body (a form field) |
|  | SecRule ARGS:/jform\[username\]/ "[^a-zA-Z]" "t:none,phase:2,deny,id:6,log,msg:'test for whitelist rule'" |
|  |  |
|  | # Only allow certain charcters in the username field for a specific URL |
|  | SecRule REQUEST\_URI "@beginsWith /joomla/index.php/component/users/" "phase:2,id:7,deny,log,chain,msg:'test for whitelist rule, on specific URI'" |
|  |  SecRule ARGS:/jform\[username\]/ "[^a-zA-Z]" |