ITIS 5246

Firewalls

Sahil Bhirud

There are 3 sections in this assignment:

1. Iptables
2. Installing ModSecurity
3. Installing OWASP Core Rule Set

1. **Iptables:**
   Iptables is a firewall that is installed by default on most of the popular distributions. In order to achieve the objectives, we will be adding the firewalls rules to the INPUT table and the NAT table.

   - **Develop a firewall policy that allows access to SSH, HTTP, and HTTPS, and port 2222**

     ```
     sudo iptables -A INPUT -p tcp -i ens33 --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
     ```

     ```
     sudo iptables -A INPUT -p tcp -i ens33 --dport 443 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
     ```

     ```
     sudo iptables -A INPUT -p tcp -i ens33 --dport 80 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
     ```

     ```
     sudo iptables -A INPUT -p tcp -i ens33 --dport 2222 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
     ```

```
Terminal - sahilbhirud@ubuntu: ~
File  Edit  View  Terminal  Tabs  Help
sahilbhirud@ubuntu:~$ sudo iptables -A INPUT -p tcp -i ens33 --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
sahilbhirud@ubuntu:~$ sudo iptables -A INPUT -p tcp -i ens33 --dport 443 -m conntrack --ctstate NEW,ESTABLISHED
sahilbhirud@ubuntu:~$ sudo iptables -A INPUT -p tcp -i ens33 --dport 443 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
sahilbhirud@ubuntu:~$ sudo iptables -A INPUT -p tcp -i ens33 --dport 80 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
sahilbhirud@ubuntu:~$ sudo iptables -A INPUT -p tcp -i ens33 --dport 2222 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
sahilbhirud@ubuntu:~$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     tcp  --  lxdbr1 any     anywhere             anywhere             tcp dpt:domain /* generated for LXD ne
twork lxdbr1 */
   43  3257 ACCEPT     udp  --  lxdbr1 any     anywhere             anywhere             udp dpt:domain /* generated for LXD ne
twork lxdbr1 */
  159 50082 ACCEPT     udp  --  lxdbr1 any     anywhere             anywhere             udp dpt:bootps /* generated for LXD ne
twork lxdbr1 */
    0     0 ACCEPT     tcp  --  ens33  any     anywhere             anywhere             tcp dpt:ssh ctstate NEW,ESTABLISHED
    0     0            tcp  --  ens33  any     anywhere             anywhere             tcp dpt:https ctstate NEW,ESTABLISHED
    0     0 ACCEPT     tcp  --  ens33  any     anywhere             anywhere             tcp dpt:https ctstate NEW,ESTABLISHED
    0     0 ACCEPT     tcp  --  ens33  any     anywhere             anywhere             tcp dpt:http ctstate NEW,ESTABLISHED
    0     0 ACCEPT     tcp  --  ens33  any     anywhere             anywhere             tcp dpt:2222 ctstate NEW,ESTABLISHED

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
 244K  630M ACCEPT     all  --  any    lxdbr1  anywhere             anywhere             /* generated for LXD network lxdbr1 */
 195K 8595K ACCEPT     all  --  lxdbr1 any     anywhere             anywhere             /* generated for LXD network lxdbr1 */

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     tcp  --  any    lxdbr1  anywhere             anywhere             tcp spt:domain /* generated for LXD ne
twork lxdbr1 */
   41  4259 ACCEPT     udp  --  any    lxdbr1  anywhere             anywhere             udp spt:domain /* generated for LXD ne
twork lxdbr1 */
  159 52152 ACCEPT     udp  --  any    lxdbr1  anywhere             anywhere             udp spt:bootps /* generated for LXD ne
twork lxdbr1 */
sahilbhirud@ubuntu:~$
```

- **Redirect incoming HTTP and HTTPS requests to your server container**

  sudo iptables -t nat -A PREROUTING -i ens33 -p tcp --dport 80 -j DNAT --to-destination 10.0.4.100:80

  sudo iptables -t nat -A PREROUTING -i ens33 -p tcp --dport 443 -j DNAT --to-destination 10.0.4.100:443

- **Redirect incoming SSH requests to your server container**

  sudo iptables -t nat -A PREROUTING -i ens33 -p tcp --dport 22 -j DNAT --to-destination 10.0.4.100:22

- **Redirect incoming requests to port 2222 to the client container's SSH service**

  sudo iptables -t nat -A PREROUTING -i ens33 -p tcp --dport 2222 -j DNAT --to-destination 10.0.4.101:22

We will now set the default policies to drop all other connections or invalid connections.

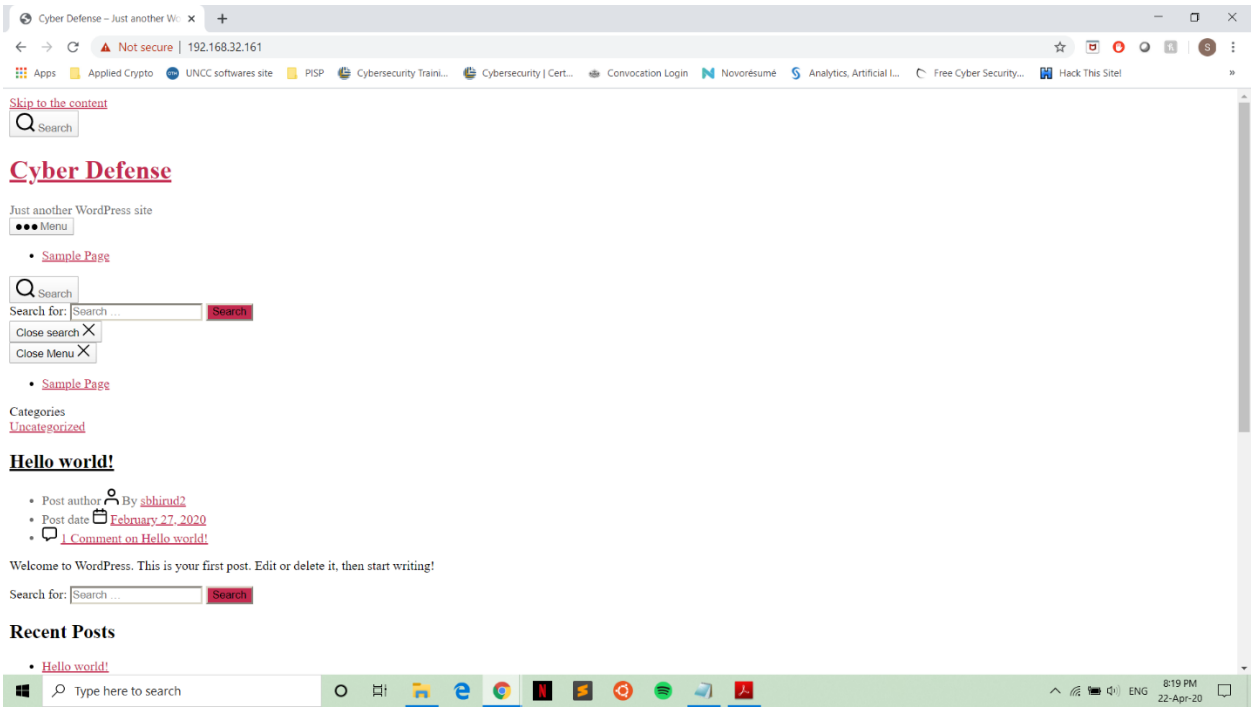
sudo iptables -P INPUT DROP

sudo iptables -P FORWARD DROP

This is a basic iptables policy which will only provide needed functionality for this assignment. To check http and https connections, you need to set up a web server on the server container or we can use telnet to check connection between two nodes.

```
sbhirud2@server: ~
Microsoft Windows [Version 10.0.18362.778]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\sahil>ssh sbhirud2@192.168.32.161
The authenticity of host '192.168.32.161 (192.168.32.161)' can't be established.
ECDSA key fingerprint is SHA256:hhgFU7SEEZy1/KfOePilDZ8SCKxV34UwH2iFbgiwqa0.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.32.161' (ECDSA) to the list of known hosts.
sbhirud2@192.168.32.161's password:
Permission denied, please try again.
sbhirud2@192.168.32.161's password:
Permission denied, please try again.
sbhirud2@192.168.32.161's password:

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

sbhirud2@server:~$
```

```
sbhirud2@client: ~

C:\Users\sahil>ssh sbhirud2@192.168.32.161 -p 2222
sbhirud2@192.168.32.161's password:

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

sbhirud2@client:~$
```



Accessing WordPress webserver from Host Machine

2. Installing ModSecurity:

ModSecurity is a toolkit for real-time web application monitoring, logging, and access control. To correctly install ModSecurity and integrate it with nginx server, we have to go through the following steps:

1. Get right version of Nginx
2. Download and install ModSecurity
3. Download nginx connector and compile it as a Dynamic Module for nginx
4. Load it to nginx
5. Configuring and testing of the integration.

It is likely that you will have to reinstall nginx server. To upgrade it, follow the method mentioned on https://nginx.org/en/linux_packages.html#Ubuntu

(Before doing this operation, make sure you back up your old installation files just in case.)

Now, install all the pre-requisites required for ModSecurity by running the following command:

```
apt-get install -y apt-utils autoconf automake build-essential git libcurl4-openssl-dev libgeoip-dev liblmdb-dev libpcre++-dev libtool libxml2-dev libyajl-dev pkgconf wget zlib1g-dev
```

Next, download, compile and install ModSecurity 3.0:

```
git clone --depth 1 -b v3/master --single-branch https://github.com/SpiderLabs/ModSecurity
```

```
cd ModSecurity
git submodule init
git submodule update
./build.sh
./configure
make
make install
```

Next, download the nginx connector for ModSecurity and compile it as a dynamic module:

```
git clone --depth 1 https://github.com/SpiderLabs/ModSecurity-nginx.git
```

Now, we have to get the source code of the exact version which we are installing.

Check nginx version using the following command:

```
nginx -v
```

then, do the following:

```
wget http://nginx.org/download/nginx-<version-number>.tar.gz
```

Extract the tar and configure the module:

```
tar zxvf nginx-<version-number>.tar.gz
```

```
cd nginx-<version-number>
./configure --with-compat --add-dynamic-module=../ModSecurity-nginx
make modules
```

Copy the modsecurity module to *"/etc/nginx/modules"* or any other location.

```
cp objs/ngx_http_modsecurity_module.so /etc/nginx/modules
```

Load the Nginx ModSecurity Connector Dynamic Module by adding the following in " *"/etc/nginx/nginx.conf".*

```
load_module /etc/nginx/modules/ngx_http_modsecurity_module.so;
```

The path here is the one where we have copied the file.

To configure and test, run the following commands:

```
mkdir /etc/nginx/modsec

wget -P /etc/nginx/modsec/

https://raw.githubusercontent.com/SpiderLabs/ModSecurity/v3/master/modsecurity.
conf-recommended

mv /etc/nginx/modsec/modsecurity.conf-recommended
etc/nginx/modsec/modsecurity.conf
```

Go to */etc/nginx/modsec/modsecurity.conf* and edit the following line:

```
SecRuleEngine On
```

To check if ModSecurity is working, put the following in */etc/nginx/modsec/main.conf*:

```
# From https://github.com/SpiderLabs/ModSecurity/blob/master/

# modsecurity.conf-recommended

#

# Edit to set SecRuleEngine On

Include "/etc/nginx/modsec/modsecurity.conf"

# Basic test rule

SecRule ARGS:testparam "@contains test" "id:1234,deny,status:403"

#modsecurity and modsecurity_rules_file directives to the NGINX configuration to
#enable

ModSecurity:

server {

modsecurity on;

modsecurity_rules_file /etc/nginx/modsec/main.conf;

}
```

Here, we are instructing nginx to load modsecurity.conf and defining a new test rule to drop a connection if there is a param=test.

Next, copy the unicode.mapping file from source folder to modsec folder in */etc/nginx/modsec/*

```
cp ~/ModSecurity/unicode.mapping /etc/nginx/modsec/
```

Restart nginx server:

```
sudo systemctl restart nginx
```

Check if the connection is being dropped by issuing the following curl command. The 403 status code will confirm that the connection is dropped and the rule is working:

```
curl localhost?testparam=test
```

3. **Installing OWASP Core Rule Set:**

Download OWASP CRS from github and extract the rules into some location.

```
wget https://github.com/SpiderLabs/owasp-modsecurity-crs/archive/v3.0.2.tar.gz
tar -xzvf v3.0.2.tar.gz
sudo mv owasp-modsecurity-crs-3.0.2 /usr/local
```

I have downloaded the rules in /usr/local directory.

Create crs-setup.conf file as a copy of rs-setup.conf.example:

```
cd /usr/local/owasp-modsecurity-crs-3.0.2
sudo cp crs-setup.conf.example crs-setup.conf
```

Next, I added all the rules to a .conf file and included the path of this file in the */etc/nginx/modsec/main.conf*

```
root@server:/usr/local/owasp-modsecurity-crs-3.0.2/rules# ls
REQUEST-900-EXCLUSION-RULES-BEFORE-CRS.conf.example   RESPONSE-954-DATA-LEAKAGES-IIS.conf
REQUEST-901-INITIALIZATION.conf                       RESPONSE-959-BLOCKING-EVALUATION.conf
REQUEST-903.9001-DRUPAL-EXCLUSION-RULES.conf          RESPONSE-980-CORRELATION.conf
REQUEST-903.9002-WORDPRESS-EXCLUSION-RULES.conf       RESPONSE-999-EXCLUSION-RULES-AFTER-CRS.conf.example
REQUEST-905-COMMON-EXCEPTIONS.conf                    crawlers-user-agents.data
REQUEST-910-IP-REPUTATION.conf                        iis-errors.data
REQUEST-911-METHOD-ENFORCEMENT.conf                   java-code-leakages.data
REQUEST-912-DOS-PROTECTION.conf                       java-errors.data
REQUEST-913-SCANNER-DETECTION.conf                    lfi-os-files.data
REQUEST-920-PROTOCOL-ENFORCEMENT.conf                 php-config-directives.data
REQUEST-921-PROTOCOL-ATTACK.conf                      php-errors.data
REQUEST-930-APPLICATION-ATTACK-LFI.conf               php-function-names-933150.data
REQUEST-931-APPLICATION-ATTACK-RFI.conf               php-function-names-933151.data
REQUEST-932-APPLICATION-ATTACK-RCE.conf               php-variables.data
REQUEST-933-APPLICATION-ATTACK-PHP.conf               restricted-files.data
REQUEST-941-APPLICATION-ATTACK-XSS.conf               scanners-headers.data
REQUEST-942-APPLICATION-ATTACK-SQLI.conf              scanners-urls.data
REQUEST-943-APPLICATION-ATTACK-SESSION-FIXATION.conf  scanners-user-agents.data
REQUEST-949-BLOCKING-EVALUATION.conf                  scripting-user-agents.data
RESPONSE-950-DATA-LEAKAGES.conf                       sql-errors.data
RESPONSE-951-DATA-LEAKAGES-SQL.conf                   sql-function-names.data
RESPONSE-952-DATA-LEAKAGES-JAVA.conf                  unix-shell.data
RESPONSE-953-DATA-LEAKAGES-PHP.conf                   windows-powershell-commands.data
root@server:/usr/local/owasp-modsecurity-crs-3.0.2/rules# cd ..
root@server:/usr/local/owasp-modsecurity-crs-3.0.2# cd /etc/nginx/modsec/
root@server:/etc/nginx/modsec# ls
```

Reload the settings of nginx:

```
sudo systemctl restart nginx
```

Test the OWASP CRS by running the following command:

```
curl -H "User-Agent: Nikto" http://localhost/
```

If you get a 404 error, then ModSecurity has dropped the connection.

```
Terminal - root@server: /etc/nginx
File  Edit  View  Terminal  Tabs  Help
root@server:/etc/nginx# curl -H "User-Agent: Nikto" http://localhost/
<html>
<head><title>403 Forbidden</title></head>
<body bgcolor="white">
<center><h1>403 Forbidden</h1></center>
<hr><center>nginx/1.14.2</center>
</body>
</html>
root@server:/etc/nginx#
```