Nginx App Protect UDP Lab Instructions

Version 3 29th June 2022 Pete White

# Introduction

The basic lab setup is complete – networking is configured and Nginx+ and App Protect are installed on nginx-1 with a basic nginx configuration.

This lab will run through the following exercises:

1. Enabling the App Protect module
2. Install the default policy
3. Configure remote logging
4. Enable various policy features
   1. enforcementMode
   2. VIOL\_HTTP\_PROTOCOL
   3. VIOL\_EVASION
   4. VIOL\_FILETYPE
5. Enable multiple policies
6. Perform troubleshooting of issues

You will also have experience of creating and managing policies using Ansible and other useful tools

# Lab Setup

Figure 1 Lab Network Diagram

Client

10.1.1.4

Nginx-1

10.1.1.5

Server

10.1.1.6

External

10.1.10.0/24

Internal

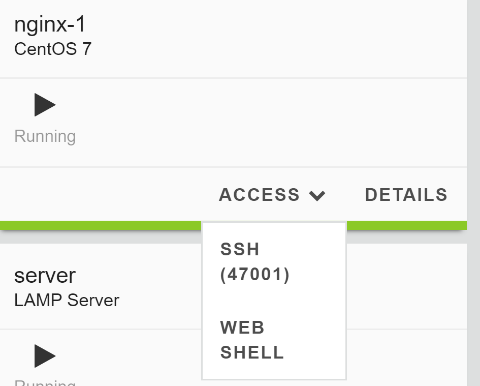
10.1.20.0/24

10 11

10 11

10.1.10.100:80

## An important note about SSH access

There are two methods by which you can access the devices – ssh or webshell. SSH is the most functional but requires some setup and web shell is the simplest but for instance doesn’t support copy and paste. See <https://help.udf.f5.com/en/articles/3347769-accessing-a-component-via-ssh> for more detailed information. **If you know how to do it, use SSH.**

# References

Use the following as references where you have issues or further questions

|  |  |
| --- | --- |
| Description | URL |
| App Protect Admin Guide | <https://docs.nginx.com/nginx-app-protect/admin-guide/> |
| App Protect Configuration Guide | <https://docs.nginx.com/nginx-app-protect/configuration/> |
| App Protect Declarative Policy Guide | <https://docs.nginx.com/nginx-app-protect/policy/> |
| App Protect Troubleshooting Guide | <https://docs.nginx.com/nginx-app-protect/troubleshooting/> |
| “Getting started with NGINX App-Protect”  F5 Internal Reference Pages | <https://docs.f5net.com/display/~melnik/Getting+started+with+NGINX+App-Protect> |

# Lab Exercises

## Lab 1 – Enable the App Protect Module

**This section demonstrates basic loadbalancing via nginx, tests the networking, and adds the App Protect module**

### Review and test basic connectivity

* Login to the client
* Run the command curl -k https://app.example.com and ensure that the web page is returned, as below:

ubuntu@ip-10-1-1-4:~$ curl -k https://app.example.com

% Total % Received % Xferd Average Speed Time Time Time Current

Dload Upload Total Spent Left Speed

100 3968 100 3968 0 0 361k 0 --:--:-- --:--:-- --:--:-- 387k

<html>

<head>

<TITLE>Using virtual server app\_example\_com and pool member 10.1.20.11 (Node #1)

</TITLE>

<meta http-equiv="Content-Type" content="text/html; charset=us-ascii" />

* Login to nginx-1 and check the nginx logs using the command tail /var/log/nginx/access.log, check the output is as per below:

[centos@ip-10-1-1-5 nginx]$ tail -5 access.log

10.1.10.10 - - [08/Jun/2020:17:34:55 +0000] "GET / HTTP/1.1" 200 3968 "-" "curl/7.47.0" "-"

### Enable the App Protect Module

* Using the command sudo vi /etc/nginx/nginx.conf, edit the nginx configuration file and add the load\_module command as the first line, as shown below:

load\_module modules/ngx\_http\_app\_protect\_module.so;

user nginx;

* Restart the nginx daemon using the command sudo nginx -s reload and ensure there are no error messages shown
* Run the command curl -k https://app.example.com?a=%3Cscript%3E and ensure the page is shown as previously
* Check the nginx logs using the command tail /var/log/nginx/access.log, check the output is as per below:

10.1.10.10 - - [08/Jun/2020:18:50:29 +0000] "GET / HTTP/1.1" 200 3968 "-" "curl/7.47.0" "-"

10.1.10.10 - - [08/Jun/2020:18:50:32 +0000] "GET /?a=%3Cscript%3E HTTP/1.1" 200 247 "-" "curl/7.47.0" "-"

## Lab 2 - Enable the default App Protect security policy

* Using the command sudo vi /etc/nginx/nginx.conf, edit the nginx configuration file and add the *app\_protect\_enable on* command as the first line of the http stanza, as shown below:

http {

app\_protect\_enable on;

include /etc/nginx/mime.types;

* Run the command curl -k https://app.example.com?a=%3Cscript%3E and ensure that the request **IS** blocked as shown below:

ubuntu@ip-10-1-1-4:~$ curl -k https://app.example.com?a=%3Cscript%3E

<html><head><title>Request Rejected</title></head><body>The requested URL was rejected. Please consult with your administrator.<br><br>Your support ID is: 16094438418830557963<br><br><a href='javascript:history.back();'>[Go Back]</a></body></html>ubuntu@ip-10-1-1-4:~$

**Note that this request has been blocked by the default App Protect policy**

## Lab 3 – Configure Remote Logging

### Basic Setup

#### Nginx Configuration

* Using the command sudo vi /etc/nginx/nginx.conf, edit the nginx configuration file and add the *app\_protect\_security\_log\_enable* and *app\_protect\_security\_log* commands below the previous configuration, as shown below:

http {

app\_protect\_enable on;

app\_protect\_security\_log\_enable on;

app\_protect\_security\_log "/etc/nginx/log-default.json" syslog:server=10.1.20.11:514;

* Restart the nginx daemon using the command sudo nginx -s reload and ensure there are no error messages shown

#### Viewing Log Messages

* Login to the server system and run the command tailf /var/log/syslog
* Run the command curl -k https://app.example.com?a=%3Cscript%3E and ensure that the request **IS** blocked as previously
* Ensure that the log arrives at the server such as below:

Jun 8 20:26:26 ip-10-1-1-5.us-west-2.compute.internal ASM: attack\_type="Non-browser Client,Abuse of Functionality,Cross Site Scripting (XSS)",blocking\_exception\_reason="N/A",date\_time="2020-06-08 20:26:26",dest\_port="80",ip\_client="10.1.10.10",is\_truncated="",method="GET",policy\_name="app\_protect\_default\_policy",protocol="HTTP",request\_status="blocked",response\_code="0",severity="Critical",sig\_cves="N/A",sig\_ids="200001475,200000098",sig\_names="XSS script tag end (Parameter) (2),XSS script tag (Parameter)",sig\_set\_names="{Cross Site Scripting Signatures;High Accuracy Signatures},{Cross Site Scripting Signatures;High Accuracy Signatures}",src\_port="59092",sub\_violations="N/A",support\_id="6955822085203780933",unit\_hostname="N/A",uri="/",violation\_rating="5",vs\_name="38-app.example.com:1-/",x\_forwarded\_for\_header\_value="N/A",outcome="REJECTED",outcome\_reason="SECURITY\_WAF\_VIOLATION",violations="Illegal meta character in value,Attack signature detected,Violation Rating Threat detected"

* Run the command curl -k https://app.example.com and ensure that the request **IS NOT** blocked
* Ensure that the log arrives at the server as below:

Jun 8 20:27:24 ip-10-1-1-5.us-west-2.compute.internal ASM: attack\_type="N/A",blocking\_exception\_reason="N/A",date\_time="2020-06-08 20:27:24",dest\_port="80",ip\_client="10.1.10.10",is\_truncated="",method="GET",policy\_name="app\_protect\_default\_policy",protocol="HTTP",request\_status="passed",response\_code="200",severity="Informational",sig\_cves="N/A",sig\_ids="N/A",sig\_names="N/A",sig\_set\_names="N/A",src\_port="59094",sub\_violations="N/A",support\_id="6955822085203781443",unit\_hostname="N/A",uri="/",violation\_rating="0",vs\_name="38-app.example.com:1-/",x\_forwarded\_for\_header\_value="N/A",outcome="PASSED",outcome\_reason="SECURITY\_WAF\_OK",violations="N/A",violation\_details="N/A",request="GET / HTTP/1.1\r\nHost: app.example.com\r\nUser-Agent: curl/7.47.0\r\nAccept: \*/\*\r\n"#015

### Log illegal requests only

#### Nginx Configuration

* Create a new file called log-illegal.json using the command sudo vi /etc/nginx/log-illegal.json with the content below:

{

"filter": {

"request\_type": "illegal"

},

"content": {

"format": "user-defined",

"format\_string": "Request ID %support\_id%: %method% %uri% received on %date\_time% from IP %ip\_client% had the following violations: %violations%",

"max\_request\_size": "any",

"max\_message\_size": "5k"

}

}

**Note that you can try different options here by referencing the documentation at** <https://docs.nginx.com/nginx-app-protect/troubleshooting/#app-protect-logging-overview>

* Using the command sudo vi /etc/nginx/nginx.conf, edit the nginx configuration file and change the *app\_protect\_security\_log* command to reference the new filename, as shown below:

http {

app\_protect\_enable on;

app\_protect\_security\_log\_enable on;

app\_protect\_security\_log "/etc/nginx/log-illegal.json" syslog:server=10.1.20.11:514;

* Restart the nginx daemon using the command sudo nginx -s reload and ensure there are no error messages shown

#### Legal Requests

* Run the command curl -k https://app.example.com and ensure that the request **IS NOT** blocked
* Ensure that the log **DOES NOT** arrive at the server

#### Illegal Requests

* Run the command curl -k https://app.example.com?a=%3Cscript%3E and ensure that the request **IS** blocked as previously
* Ensure that the log arrives at the server as previously

## **Lab 4 – Enable Various Policy Features**

In this section we will create various policy files to demonstrate some of the policy features available.

Refer to the nginx documentation at <https://docs.nginx.com/nginx-app-protect/configuration/> where required.

### Enforcement Mode

#### Nginx Configuration

* Create a new file called policy-transparent.json using the command sudo vi /etc/nginx/policy-transparent.json with the content below:

{

"policy": {

"name": "policy-transparent",

"template": { "name": "POLICY\_TEMPLATE\_NGINX\_BASE" },

"applicationLanguage": "utf-8",

"enforcementMode": "transparent"

}

}

* Using the command sudo vi /etc/nginx/nginx.conf, edit the nginx configuration file and change the *app\_protect\_policy\_file* command to reference the policy file, as shown below:

http {

app\_protect\_enable on;

app\_protect\_policy\_file "/etc/nginx/policy-transparent.json";

app\_protect\_security\_log\_enable on;

app\_protect\_security\_log "/etc/nginx/log-illegal.json" syslog:server=10.1.20.11:514;

* Restart the nginx daemon using the command sudo nginx -s reload and ensure there are no error messages shown

#### Send Illegal Request

* Run the command curl -k https://app.example.com?a=%3Cscript%3E and ensure that the request **IS NOT** blocked
* Ensure that the log arrives at the server as previously

### VIOL\_HTTP\_PROTOCOL

In this lab we will enable the VIOL\_HTTP\_PROTOCOL violation globally in the Blocking Settings section and configure the alarm and block settings independently. The policy file will be created using an Ansible playbook which merges a template file and variables.

#### Nginx Configuration

* Review the Jinja2 template file *nap\_template.j2* using the command more nap\_template.j2. Note the variables within the double braces : {{ }} ( highlighted below in yellow) and the for command for looping over a list of values (highlighted below in green). View more about Ansible templating at <https://docs.ansible.com/ansible/latest/user_guide/playbooks_templating.html> and about Jinja at <https://palletsprojects.com/p/jinja/>

Template File:

[centos@ip-10-1-1-5 ~]$ more nap\_template.j2

{

"name": "policy-viol\_http\_protocol",

"template": {

"name": "POLICY\_TEMPLATE\_NGINX\_BASE"

},

"applicationLanguage": "utf-8",

"enforcementMode": "{{ enforcementMode| default (blocking) }}"

"server-technologies": [

{% for srvtech in server\_tech %}

{

"serverTechnologyName": "{{srvtech.name}}"

{% if loop.last %}

}

{% else %}

},

{% endif %}

{% endfor %}

],

"signature-settings":{

"signatureStaging": false

},

"modifications":[

--More--(34%)

* View the variables which are inserted into the *nap\_vars.yaml* file using the command more nap\_vars.yaml – note that this is in YAML syntax where a hyphen denotes a list of entries. To see more information about YAML, go to <https://docs.ansible.com/ansible/latest/reference_appendices/YAMLSyntax.html>

Variables file:

[centos@ip-10-1-1-5 ~]$ more nap\_vars.yaml

server\_tech:

- t1:

name: Unix/Linux

enforcementMode: blocking

block\_violation:

- violation1:

name: VIOL\_HTTP\_PROTOCOL

alarm\_switch: true

block\_switch: false

* Review the *nap\_playbook.yaml* file with the command more nap\_playbook.yaml. This contains tasks to be performed by Ansible. Note that the template and copy tasks are used. See more detail on these tasks at <https://docs.ansible.com/ansible/latest/collections/ansible/builtin/template_module.html> and <https://docs.ansible.com/ansible/latest/collections/ansible/builtin/copy_module.html#ansible-collections-ansible-builtin-copy-module> See a list of builtin plugins at <https://docs.ansible.com/ansible/latest/collections/ansible/builtin/index.html>

Note the source template nap\_template.j2 and destination file policy-viol\_http\_protocol.json

[centos@ip-10-1-1-5 ~]$ more nap\_playbook.yaml

#!/usr/bin/env ansible-playbook

###############################################################################

# Ansible Playbook for NGINX App Protect Policy Management

# Borrowed from https://github.com/fbchan/nginx-app-protect-policy

# Example: ansible-playbook -i inventory nap\_play.yml

# Author: Foo-Bang (fb@f5.com)

# Version: 0.01

###############################################################################

---

- name: "### PLAYBOOK 01 ### - Create NGINX App Protect Policy"

hosts: localhost

connection: local

gather\_facts: False

vars\_files:

- nap\_vars.yaml

tasks:

- name: " # TASK 01 # - Generating NGINX App Protect policy enforcement"

template:

src: nap\_template.j2

dest: policy-viol\_http\_protocol.json

- name: "### PLAYBOOK 02 ### - Transfering App Protect Policy Enforcement"

hosts: localhost

tasks:

- name: "# TASK 01 # - Transferring policy for enforcement"

copy:

src: ./policy-viol\_http\_protocol.json

dest: /etc/nginx/policy-viol\_http\_protocol.json

owner: root

group: root

force: yes

* Run the Ansible playbook with the command ansible-playbook nap\_playbook.yaml

Example:

[centos@ip-10-1-1-5 ~]$ ansible-playbook nap\_playbook.yaml

[WARNING]: provided hosts list is empty, only localhost is available. Note that

the implicit localhost does not match 'all'

PLAY [### PLAYBOOK 01 ### - Create NGINX App Protect Policy] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK [# TASK 01 # - Generating NGINX App Protect policy enforcement] \*\*\*\*\*\*\*\*\*\*\*

ok: [localhost]

PLAY [### PLAYBOOK 02 ### - Transferring App Protect Policy Enforcement] \*\*\*\*\*\*\*\*

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [localhost]

TASK [# TASK 01 # - Transferring policy for enforcement] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [localhost]

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

localhost : ok=3 changed=1 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

* + Ensure that there are no failed tasks.
  + Ensure the policy file is created in the directory with the command ls \*.json:

[centos@ip-10-1-1-5 ~]$ ls \*.json

policy-viol\_http\_protocol.json

* + View the content of the policy with the command more policy-viol\_http\_protocol.json

[centos@ip-10-1-1-5 ~]$ more policy-viol\_http\_protocol.json

{

"name": "policy-viol\_http\_protocol",

"template": {

"name": "POLICY\_TEMPLATE\_NGINX\_BASE"

},

"applicationLanguage": "utf-8",

"enforcementMode": "blocking"

"server-technologies": [

{

"serverTechnologyName": "Unix/Linux"

}

],

"signature-settings":{

"signatureStaging": false

},

"modifications":[

],

"signatures": [

],

"blocking-settings": {

"violations": [

{

--More--(77%)

Note the server-technologies, enforcementMode and violations taken from the variables file.

* + Ensure that file has been copied to the /etc/nginx directory using the command ls -la /etc/nginx/\*.json

[centos@ip-10-1-1-5 ~]$ ls -la /etc/nginx/\*.json

-rw-r--r--. 1 root root 164 Jun 8 2020 /etc/nginx/log-default.json

-rw-r--r-- 1 root root 538 May 6 15:42 /etc/nginx/policy-viol\_http\_protocol.json

* Using the command sudo vi /etc/nginx/nginx.conf, edit the nginx configuration file and change the *app\_protect\_policy\_file* command to reference the policy file, as shown below:

http {

app\_protect\_enable on;

app\_protect\_policy\_file "/etc/nginx/policy-viol\_http\_protocol.json";

app\_protect\_security\_log\_enable on;

app\_protect\_security\_log "/etc/nginx/log-illegal.json" syslog:server=10.1.20.11:514;

* Restart the nginx daemon using the command sudo nginx -s reload and ensure there are no error messages shown

#### Send Illegal Request – blocked by WAF

* Run the command curl -H "Host: 1.2.3.4" http://app.example.com and ensure that the request **IS** blocked
* Ensure that the log arrives at the server as previously

#### Send Illegal Request – blocked by Nginx

* Run the command curl -H "Content-Length: -26" http://app.example.com and ensure that the request is rejected by NGINX which returns the 400 Bad Request
* Ensure that the log arrives at the server as previously

#### Turn Off Alarm

* Modify the nap\_vars.yaml file to set the alarm to false using the command vi nap\_vars.yaml

server\_tech:

- t1:

name: Unix/Linux

enforcementMode: blocking

block\_violation:

- violation1:

name: VIOL\_HTTP\_PROTOCOL

alarm\_switch: false

block\_switch: true

* Re-run the playbook with the command ansible-playbook nap\_playbook.yaml
* Review the policy file using the command more /etc/nginx/policy-viol\_http\_protocol.json. Check that the violation is set to block but not alarm as below:

"name": "VIOL\_HTTP\_PROTOCOL",

"alarm": false,

"block": true

* Restart the nginx daemon using the command sudo nginx -s reload and ensure there are no error messages shown

#### Send Illegal Request – blocked but no log

* Run the command curl -H "Host: 1.2.3.4" http://app.example.com and ensure that the request **IS** blocked
* Ensure that the log **DOES NOT** arrive at the server

**Note: If the log still arrives at the server, you should check that the WAF is only logging illegal requests. For reference, see Lab 3**

#### Turn Off blocking

* Modify the the nap\_vars.yaml variables file to set the violation to alarm but not block, re-run the Ansible playbook and check the policy is set as below:

"name": "VIOL\_HTTP\_PROTOCOL",

"alarm": true,

"block": false

* Restart the nginx daemon

#### Send Illegal Request – logged but no block

* Run the command curl -H "Host: 1.2.3.4" http://app.example.com and ensure that the request **IS** **NOT** blocked
* Ensure that the log **DOES** arrive at the server

### VIOL\_EVASION

In this lab we will enable the VIOL\_EVASION violation globally in the Blocking Settings section and look at how to disable blocking for individual evasion techniques. Here you will use a web-based tool for creating the policy.

#### Nginx Configuration

* Using your browser, navigate to <https://waffler.dev/prod/> and create a policy named policy-viol\_evasion.
  + In the left-hand menu, navigate to Violations under the BLOCK SETTINGS group.

Graphical user interface, application

Description automatically generated

* + Click on Add Violation and select Evasion technique detected

Graphical user interface, application

Description automatically generated

* + Click the Alarm and Block checkboxes for the violation

Graphical user interface, application, Teams

Description automatically generated

* + In the left-hand menu, navigate to Evasions under the BLOCK SETTINGS group
  + Select all of the text in the JSON lower tab and copy with Ctrl-C ( or export the file using the download function in the top right-hand side) Icon

    Description automatically generated.
* Create a new file called policy-viol\_evasion.json using the command sudo vi /etc/nginx/policy-viol\_evasion.json and paste in the JSON from the lower pane ( or use scp or your preferred method of file transfer )
* Using the command sudo vi /etc/nginx/nginx.conf, edit the nginx configuration file and change the *app\_protect\_policy\_file* command to reference the policy file, as shown below:

http {

app\_protect\_enable on;

app\_protect\_policy\_file "/etc/nginx/policy-viol\_evasion.json";

app\_protect\_security\_log\_enable on;

app\_protect\_security\_log "/etc/nginx/log-illegal.json" syslog:server=10.1.20.11:514;

* Restart the nginx daemon using the command sudo nginx -s reload and ensure there are no error messages shown

#### Send Illegal Request – blocked by WAF

* Run the command curl -k https://app.example.com/%09 and ensure that the request **IS** blocked
* Ensure that the log arrives at the server as previously

#### Disable blocking for all violations

* Set the policy so that block is set to false for VIOL\_EVASION and restart the nginx daemon:
  + Navigate to Blocking Settings>Violations and uncheck Block

Graphical user interface, application, Teams

Description automatically generated

* + Check that the policy is updated to show “block” is set to falseGraphical user interface, text

    Description automatically generated
  + Update policy file as before
* Run the curl command as above and ensure that the response is 404 Not Found ie **NOT BLOCKED** by WAF

**Note that this has disabled blocking for ALL evasion violations**

#### Enable blocking for all violations, disable Apache Whitespace violation

* Modify the policy file to disable the Apache Whitespace check as below, and restart the nginx daemon:
  + Click on Add Evasion and select Apache Whitespace

Graphical user interface, text, application, chat or text message

Description automatically generated

* + Check the Enabled button twice until the JSON text in the lower pane shows “enabled”: false

For the Apache whitespace evasion

Graphical user interface, text, application

Description automatically generated

{

"policy": {

"name": "policy-viol\_evasion",

"template": { "name": "POLICY\_TEMPLATE\_NGINX\_BASE" },

"applicationLanguage": "utf-8",

"enforcementMode": "blocking",

"blocking-settings": {

"violations": [

{

"name": "VIOL\_EVASION",

"alarm": true,

"block": true

}

],

"evasions": [

{

"description": "Apache whitespace",

"enabled": "false"

}

]

}

}

}

* Run the curl command again and ensure that the request **IS NOT** blocked by the WAF

**Note that we have enabled all violations and only disabled the one which gives us a problem. A much better solution**

### VIOL\_FILETYPE

In this lab we will enable the VIOL\_FILETYPE violation in Blocking Settings, and show the default negative policy ie all filetypes are accepted except for a specific list of disallowed file types. We will then change the default policy to block all filetypes except for the allowed type. ie positive policy. You will use another tool to initially create the policy – this one is portable and simple to use but has fewer features included.

#### Nginx Configuration

* Using your browser, navigate to <https://github.com/pwhitef5/app-protect-policy-creator>
  + Click on the green Code button and click on Download ZIP

Graphical user interface, text, application

Description automatically generated

* + Extract the zip archive and double-click on the create\_policy.html file which should open in a web browser window.
  + In Policy Name, enter ***policy-viol\_filetype***
  + Click on the [ + ] icon to the right of Violations. A frame containing a list of violations should open
  + Check both Alarm and Block checkboxes relating to VIOL\_FILETYPE
  + Click Create Policy button at the bottom of the page
  + Click Export Policy button to export policy to a file
* Create a new file called policy- viol\_evasion.json using the command sudo vi /etc/nginx/policy-viol\_filetype.json with the content below, or copy exported file to device with the correct filename:

{

"policy": {

"name": "policy-viol\_filetype",

"template": {

"name": "POLICY\_TEMPLATE\_NGINX\_BASE"

},

"applicationLanguage": "utf-8",

"enforcementMode": "blocking",

"blocking-settings": {

"violations": [

<snipped for clarity>

{

"name": "VIOL\_FILETYPE",

"alarm": true,

"block": true

}

]

}

}

}

* Using the command sudo vi /etc/nginx/nginx.conf, edit the nginx configuration file and change the *app\_protect\_policy\_file* command to reference the policy file, as shown below:

http {

app\_protect\_enable on;

app\_protect\_policy\_file "/etc/nginx/policy-viol\_filetype.json";

app\_protect\_security\_log\_enable on;

app\_protect\_security\_log "/etc/nginx/log-illegal.json" syslog:server=10.1.20.11:514;

* Restart the nginx daemon using the command sudo nginx -s reload and ensure there are no error messages shown

#### Send legal Request – allowed

* Run the command curl -k https://app.example.com/index.php and ensure that the request **IS NOT** blocked or logged

#### Send Illegal Request – blocked

* Run the command curl -k https://app.example.com/test.exe and ensure that the request **IS** blocked
* Ensure that the log arrives at the server as previously

**Note that .exe is on the list of disallowed filetypes which is the reason why this request is blocked**

#### Nginx Configuration to change the default policy

* Modify the file policy- viol\_evasion.json to match the content below, and reload nginx:

{

"policy": {

"name": "policy-viol\_filetype",

"template": { "name": "POLICY\_TEMPLATE\_NGINX\_BASE" },

"applicationLanguage": "utf-8",

"enforcementMode": "blocking",

"blocking-settings": {

"violations": [

{

"name": "VIOL\_FILETYPE",

"alarm": true,

"block": true

}

This part is adding html as an explicit allowed filetype

]

},

"filetypes": [

{

"name": "html",

"type": "explicit",

"allowed": true

}

]

},

This part is modifying the default policy to delete the wildcard filetype

"modifications": [

{

"entityChanges": {

"type": "wildcard"

},

"entity": {

"name": "\*"

},

"entityType": "filetype",

"action": "delete"

}

]

}

#### Send legal Request – allowed

* Run the command curl -k https://app.example.com/badlinks.html and ensure that the request **IS NOT** blocked or logged

#### Send Illegal Request – blocked

* Run the command curl -k https://app.example.com/index.php and ensure that the request **IS** blocked
* Ensure that the log arrives at the server as previously

## **Lab 5 – Enable multiple policies**

In this section we will demonstrate the use of two policies which are applied to two different URLs within location blocks.

#### Nginx Configuration

* Using the command sudo vi /etc/nginx/nginx.conf, edit the nginx configuration file and change the *app\_protect\_policy\_file* command to reference the default policy file, as shown below:

http {

app\_protect\_enable on;

app\_protect\_policy\_file "/etc/nginx/NginxDefaultPolicy.json";

app\_protect\_security\_log\_enable on;

app\_protect\_security\_log "/etc/nginx/log-illegal.json" syslog:server=10.1.20.11:514;

* Add a new location block as below:

server {

listen 10.1.10.100:80 default\_server;

server\_name app.example.com;

location / {

proxy\_pass http://app\_example\_com;

}

location /basic {

app\_protect\_policy\_file "/etc/nginx/policy-viol\_filetype.json";

proxy\_pass http://app\_example\_com;

}

}

* Restart the nginx daemon using the command sudo nginx -s reload and ensure there are no error messages shown

#### Send legal Request – allowed

* Run the command curl -k https://app.example.com/index.php and ensure that the request **IS NOT** blocked or logged

#### Send Illegal Request – blocked

* Run the command curl -k https://app.example.com/basic/index.php and ensure that the request **IS** blocked
* Ensure that the log arrives at the server as previously

## **Lab 6 – Troubleshooting**

In this section we will create various issues with the policy file and nginx.conf and show how to troubleshoot these.

### nginx.conf Issues

In this lab we will demonstrate an issue with the nginx configuration – an incorrect filename which prevents the policy being loaded.

#### Nginx Configuration

* Using the command sudo vi /etc/nginx/nginx.conf, edit the nginx configuration file and remove the location /basic stanza created in lab 5 and also change the global policy filename to be incorrect, as shown below:

http {

app\_protect\_enable on;

app\_protect\_policy\_file "/etc/nginx/NginxDefaultPolicywrong.json";

app\_protect\_security\_log\_enable on;

app\_protect\_security\_log "/etc/nginx/log-illegal.json" syslog:server=10.1.20.11:514;

* Restart the nginx daemon using the command sudo nginx -s reload and view the error message as shown below:

[centos@ip-10-1-1-5 nginx]$ sudo nginx -s reload

{

"completed\_successfully" : false,

"error\_message" : "Failed to read '/etc/nginx/NginxDefaultPolicywrong.json'",

"error\_line\_number" : 19

}

nginx: [error] APP\_PROTECT { "event": "configuration\_load\_failure", "software\_version": "2.52.1", "error\_message":"Failed to read '/etc/nginx/NginxDefaultPolicywrong.json'","completed\_successfully":false,"error\_line\_number":19}

[centos@ip-10-1-1-5 nginx]$

**Note the error\_line\_number is the line on which the error is found and the error\_message is relevant to the error**

* View the error log using the command sudo tail /var/log/nginx/error.log, notice the error message such as below:

2020/06/26 11:59:19 [error] 2291#2291: APP\_PROTECT { "event": "configuration\_load\_failure", "software\_version": "2.52.1", "error\_message":"Failed to read ‘/etc/nginx/NginxDefaultPolicywrong.json’","completed\_successfully":false, "error\_line\_number":19}

**Note that the log level is [error] and is tagged with APP\_PROTECT. The content is in JSON format.**

* Correct the filename in nginx.conf and reload nginx, ensure the policy loads as expected

### Policy Issues

In this lab we will demonstrate issues with the policy – non-JSON format, and incorrect variable types

#### Policy Configuration

* Modify /etc/nginx/nginx.conf to use the policy as per Lab 4 - /etc/nginx/policy-viol\_filetype.json
* Modify the policy to remove the double-quote at the start of filetypes as per below:

filetypes": [

{

"name": "html",

"type": "explicit",

"allowed": true

}

]

* Restart the nginx daemon and note the failure message as below:

[centos@ip-10-1-1-5 nginx]$ sudo nginx -s reload

{

"completed\_successfully" : false,

"error\_message" : "Failed to import Policy '/etc/nginx/policy-viol\_filetype.json' from '/etc/nginx/policy-viol\_filetype.json': Fail parse JSON Policy: '\"' expected, at character offset 441 (before \"filetypes\": [\\n ...\") \n.",

"error\_line\_number" : 44

}

nginx: [error] APP\_PROTECT { "event": "configuration\_load\_failure", "software\_version": "2.52.1", "error\_message":"Failed to import Policy '/etc/nginx/policy-viol\_filetype.json' from '/etc/nginx/policy-viol\_filetype.json': Fail parse JSON Policy: '\"' expected, at character offset 441 (before \"filetypes\": [\\n ...\") \n.","completed\_successfully":false,"error\_line\_number":44}

**Note that the error indicates the error type ie ‘Fail parse JSON policy:** '**\**"' **expected’ and both the line number and the character offset**

* View the error log using the command sudo tail /var/log/nginx/error.log, notice the error message such as below:

2020/06/26 12:05:03 [error] 2358#2358: APP\_PROTECT { "event": "configuration\_load\_failure", "software\_version": "2.52.1", "error\_message":"Failed to import Policy '/etc/nginx/policy-viol\_filetype.json' from '/etc/nginx/policy-viol\_filetype.json': Fail parse JSON Policy: '\"' expected, at character offset 441 (before \"filetypes\": [\\n ...\") \n.","completed\_successfully":false,"error\_line\_number":44}

* To check the policy file for JSON errors, run the command cat /etc/nginx/policy-viol\_filetype.json | python -m json.tool and note the output showing the error in more detail as below:

[centos@ip-10-1-1-5 nginx]$ cat /etc/nginx/policy-viol\_filetype.json | python -m json.tool

Expecting property name: line 44 column 9 (char 441)

[centos@ip-10-1-1-5 nginx]$

* To find the error in the file, open the policy using the command sudo vi /etc/nginx/policy-viol\_filetype.json and use the command :16 to jump to the relevant line, use :goto 441 to jump to the character offset
* Fix the error by replacing the double-quote in the policy file and reloading nginx
* Re-check the file JSON format is correct as below ie no errors:

[centos@ip-10-1-1-5 nginx]$ cat /etc/nginx/policy-viol\_evasion.json|python -m json.tool

{

"policy": {

"applicationLanguage": "utf-8",

"blocking-settings": {

"evasions": [

{

…

* Modify the policy and change “allowed”: true to “allowed”: 12 ie change from Boolean type to integer, as per below:

"filetypes": [

{

"name": "html",

"type": "explicit",

"allowed": 12

}

]

* Reload nginx and view error as below:

[centos@ip-10-1-1-5 nginx]$ sudo nginx -s reload

{

"completed\_successfully" : false,

"error\_message" : "Failed to import Policy 'policy-viol\_filetype' from '/etc/nginx/policy-viol\_filetype.json': Could not parse/validate the File Type. Invalid value '12' for field 'allowed'.",

"error\_line\_number" : 44

}

nginx: [error] APP\_PROTECT { "event": "configuration\_load\_failure", "software\_version": "2.52.1", "error\_message":"Failed to import Policy 'policy-viol\_filetype' from '/etc/nginx/policy-viol\_filetype.json': Could not parse/validate the File Type. Invalid value '12' for field 'allowed'.","completed\_successfully":false,"error\_line\_number":44}

[centos@ip-10-1-1-5 nginx]$

* Fix the error and reload nginx

### Debugging App Protect

In this lab we will demonstrate how to turn on debugging of App Protect such as where a specific area of functionality is not working as expected.

#### Nginx Configuration

* Modify /etc/nginx/nginx.conf to use the HTTP protocol policy file from Lab 4, as below:

http {

app\_protect\_enable on;

app\_protect\_policy\_file "/etc/nginx/policy-viol\_http\_protocol.json";

app\_protect\_security\_log\_enable on;

app\_protect\_security\_log "/etc/nginx/log-illegal.json" syslog:server=10.1.20.11:514;

* Reload nginx

#### Logger Configuration File

* Modify /etc/app\_protect/bd/logger.cfg ( the default logging configuration file ) using the command sudo vi /etc/app\_protect/bd/logger.cfg
* Note the names of the modules which can be configured for debugging:

################################################################################################

#

# Logger configuration file

#

# Existing modules:

#

# IO\_PLUGIN (Requests & Responses) FTP\_PLUGIN (ftp) SMTP\_PLUGIN (smtp)

# BEM (Accumulation Responses), ECARD (Tables),

# ECARD\_POLICY (Enforcer), BD\_SSL (Communications), UMU (Memory),

# IMF (Sockets), BD\_MISC (Config and miscs),COOKIE\_MGR (Cookies), REG\_EXP (Regular expressions),

# RESP\_PARAMS (Extractions), ATTACK\_SIG (Attack Signatures), BD\_XML(XML Enforcer),

# ATTACK\_ENGINE (BF & BOT detect monitor), XML\_PARSER (all xml engine), ACY (pattern match engine),

# BD\_PB (policy builder), BD\_PB\_SAMPLING (sampling decisions for pb), LEGAL\_HASH (internal cache tables),

# CLIENT\_SIDE (Client Side infrastructre), STATS (policy builder statistics), ICAP (content inspection),

# CLUSTER\_ANOMALY (the anomaly distributed channel), PIPE (shmem channel bd-pbng, bd-lrn),

# MPP\_PARSER (Multipart parser), SA\_PLUGIN (Session awareness), DATA\_PROTECT (Data Protection Library),

# GDM (Guardium DB security), ASM\_IRULE (ASM iRule commands), LIBDATASYNC (Data Sync Library),

# BD\_CONF (BD MCP configuration), MPI\_CHANNEL (BD initiated MPI events),

# BD\_FLUSH\_TBLS(flush BD conf tables), CSRF (CSRF feature), BRUTE\_FORCE\_ENFORCER (Brute Force feature),

# LONG\_REQUEST (Long request), HTML\_PARSER (HTML parser)

* Add the following lines at the end of the file:

MODULE = IO\_PLUGIN;

LOG\_LEVEL = TS\_DEBUG;

FILE = 2;

* Run the following command to begin debugging: sudo /bin/bash -c '/opt/app\_protect/bin/set\_active.pl -g' nginx
* Capture the debug file output using the command tail -f /var/log/app\_protect/bd-socket-plugin.log
* On the client, run the command curl -H "Host: 1.2.3.4" http://app.example.com and view the debug output
* Remove the added lines from /etc/app\_protect/bd/logger.cfg and run the command sudo /bin/bash -c '/opt/app\_protect/bin/set\_active.pl -g' nginx to disable the debug logging

**Well Done! This lab is complete**