

NGINX APP PROTECT UDF LAB

INSTRUCTIONS

VERSION 4 2ND SEPT 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
 - a. enforcementMode
 - b. VIOL_HTTP_PROTOCOL
 - c. VIOL_EVASION
 - d. 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

All completed files are in the /home/centos/lab_files/complete directory on the nginx-1 device for reference, all beginning files are in /home/centos/lab_files/complete to be used in the case of issues with creating policies.

LAB SETUP

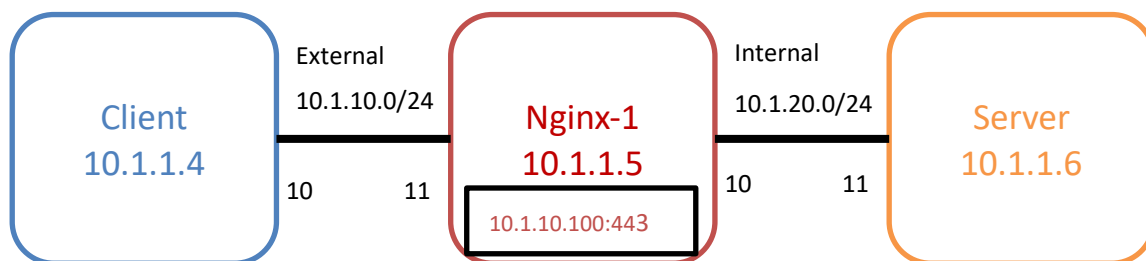
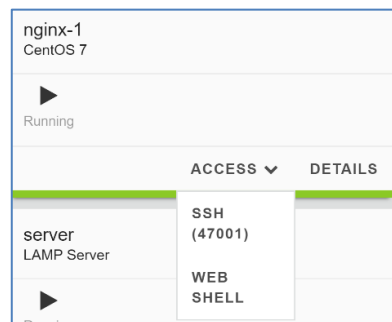


Figure 1 Lab Network Diagram

AN IMPORTANT NOTE ABOUT SSH ACCESS

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



webshell. SSH is simplest but for [component-via-SSH](#).

LAB GUIDE COLOUR CONVENTION

Throughout this lab guide, background colours are used to show on which device the commands should be run.

These are shown below:

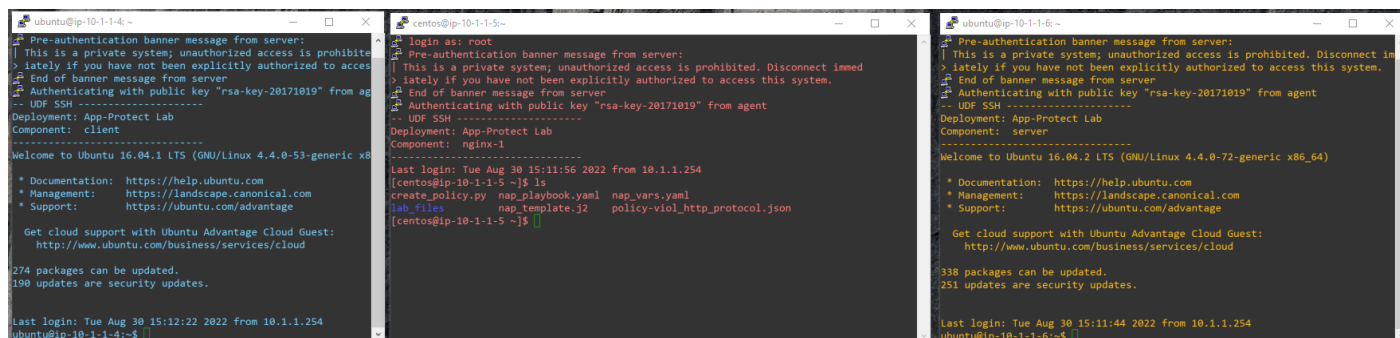
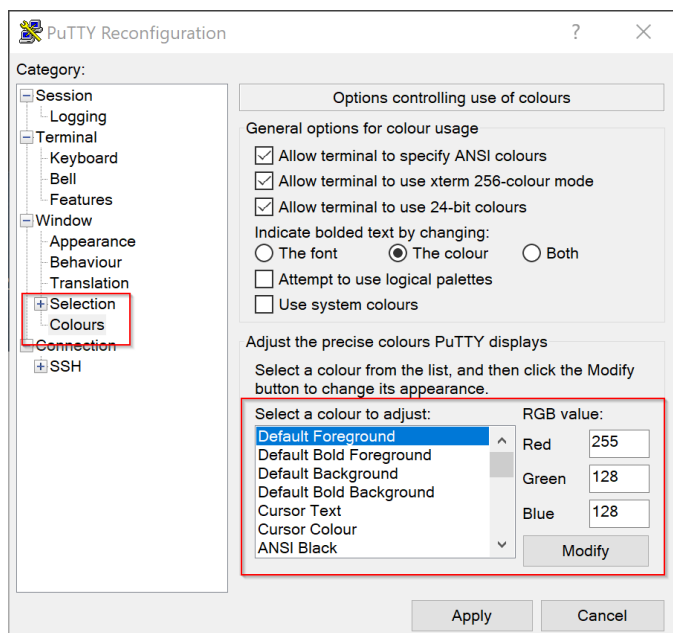
Client

Nginx-1

Server

You may wish to set the Default Foreground colour of your terminal window to match this colour scheme to make it simpler to switch between windows

Example:



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/declarative-policy/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/ESNPI/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
<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 ~]$ tail /var/log/nginx/access.log
10.1.10.10 - - [30/Aug/2022:15:26:48 +0000] "GET / HTTP/1.1" 200 3968 "-" "curl/7.47.0" "-"
```

ENABLE THE APP PROTECT MODULE

- On nginx-1, edit the nginx configuration file using the command `sudo vi /etc/nginx/nginx.conf`, and add the `load_module` command as the first line, as shown below:

```
load_module modules/ngx_http_app_protect_module.so;
user  nginx;
```

- Reload NGINX using the command `sudo nginx -s reload` and ensure there are no error messages shown
- In the client, Run the command `curl -k https://app.example.com?a=<script>` 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/2020:18:50:32 +0000] "GET /?a=<script> 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;
```

- Reload NGINX 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=<script>` 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

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/app_protect/conf/log_default.json"
    syslog:server=10.1.20.11:514;
```

- Reload NGINX using the command `sudo nginx -s reload` and ensure there are no error messages shown
- Review the default log settings using the command `cat /etc/app_protect/conf/log_default.json`

Example:

```
[centos@ip-10-1-1-5 nginx]$ cat /etc/app_protect/conf/log_default.json
{
  "filter": {
    "request_type": "illegal"
  },
  "content": {
    "format": "default",
    "max_request_size": "any",
    "max_message_size": "5k"
  }
}
```

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=<script>` and ensure that the request **IS** blocked as previously
- Ensure that the log arrives at the server such as below:

```
ubuntu@ip-10-1-1-6:~$ tailf /var/log/syslog
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"
```

Note that there are multiple violations logged – *Non-Browser Client* is logged because we are using curl to perform the request, which is not a standard browser. We will ignore this violation when viewing logs

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

```
Aug 30 16:27:11 ip-10-1-1-5.us-west-2.compute.internal ASM: Request ID  
4159779018876285411: GET / received on 2022-08-30 16:27:10 from IP 10.1.10.10 had the  
following violations: Bot Client Detected#015
```

LOG ILLEGAL REQUESTS ONLY

NGINX CONFIGURATION

- Create a new file called `log-illegal.json` using the command `sudo vi /etc/app_protect/conf/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/app_protect/conf/log-illegal.json"
    syslog:server=10.1.20.11:514;
}
```

- Reload NGINX 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=<script>` and ensure that the request **IS** blocked as previously:

```
ubuntu@ip-10-1-1-4:~$ curl -k https://app.example.com?a=<script>
<html><head><title>Request Rejected</title></head><body>The requested URL was rejected.
Please consult with your administrator.<br><br>Your support ID is:
2072338535331372578<br><br><a href='javascript:history.back();'>[Go
Back]</a></body></html>ubuntu@ip-10-1-1-4:~$
```

- Ensure that the log arrives at the server, using the new format:

```
Aug 30 16:33:39 ip-10-1-1-5.us-west-2.compute.internal ASM: Request ID
2072338535331372578: GET / received on 2022-08-30 16:33:39 from IP 10.1.10.10 had the
following violations: Illegal meta character in value,Attack signature detected,Violation
Rating Threat detected,Bot Client Detected#015
```


Note the support ID presented to the client and in the log file – this would be used for troubleshooting of client problems

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/app_protect/conf/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/app_protect/conf/policy-transparent.json";
    app_protect_security_log_enable on;
    app_protect_security_log "/etc/app_protect/conf/log-illegal.json"
    syslog:server=10.1.20.11:514;
```

- Reload NGINX 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=<script>` and ensure that the request **IS NOT** blocked
- Ensure that the log arrives at the server as previously:

```
Aug 30 16:44:11 ip-10-1-1-5.us-west-2.compute.internal ASM: Request ID
4159779018876285920: GET / received on 2022-08-30 16:44:11 from IP 10.1.10.10 had the
following violations: Illegal meta character in value,Attack signature detected,Violation
Rating Threat detected,Bot Client Detected#015
```

Note that there are multiple violations logged:

Violation	Reason
Illegal meta character in value	This is because of the < character in the value of the <i>a</i> parameter
Attack signature detected	This is because the insertion of a script tag is seen as an XSS attached signature
Violation Rating Threat detected	This is because this request is seen as a higher threat ie unlikely to be a false positive
Bot Client Detected#015	This is because we are using cURL to perform the request. As before, we will ignore this violation.

The F5 Knowledge Base article [K14230463: Reduce false positive violations in NGINX App Protect](#) is useful to review at this point

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
  },
}
--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: true
```

- 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 built-in 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.yaml
# 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 ### - Transferring App Protect Policy Enforcement"
  hosts: localhost
  tasks:
    - name: " # TASK 01 # - Transferring policy for enforcement"
      copy:
        src: ./policy-viol_http_protocol.json
        dest: /etc/app_protect/conf/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=2    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": [
    {
      "name": "VIOL_HTTP_PROTOCOL",
      "alarm": true,
      "block": true
    }
  ]
},
},
--More-- (77%)

```

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

- o Ensure that file has been copied to the /etc/app_protect/conf directory using the command `ls -la /etc/app_protect/conf/*.json`

```

[centos@ip-10-1-1-5 ~]$ ls -la /etc/app_protect/conf/*.json
-rw-r--r-- 1 nginx  nginx   184 Jun 16 18:58 /etc/app_protect/conf/log_default.json
-rw-r--r-- 1 root   root    341 Aug 30 16:22 /etc/app_protect/conf/log-illegal.json
-rw-r--r-- 1 nginx  nginx  3655 Jun 16 18:58
/etc/app_protect/conf/NginxApiSecurityPolicy.json
-rw-r--r-- 1 nginx  nginx   134 Jun 16 18:58
/etc/app_protect/conf/NginxDefaultPolicy.json
-rw-r--r-- 1 nginx  nginx  8491 Jun 16 18:58
/etc/app_protect/conf/NginxStrictPolicy.json
-rw-r--r-- 1 root   root    513 Aug 30 17:03 /etc/app_protect/conf/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/app_protect/conf/policy-viol_http_protocol.json";
    app_protect_security_log_enable on;
    app_protect_security_log "/etc/app_protect/conf/log-illegal.json"
    syslog:server=10.1.20.11:514;
}

```

- Reload NGINX 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" -k https://app.example.com` and ensure that the request **IS** blocked

```
ubuntu@ip-10-1-1-4:~$ curl -H "Host: 1.2.3.4" -k https://app.example.com
<html><head><title>Request Rejected</title></head><body>The requested URL was
rejected. Please consult with your administrator.<br><br>Your support ID is:
8531451594106347485<br><br><a href='javascript:history.back();'[Go
Back]</a></body></html>
```

- Ensure that the log arrives at the server as previously

```
Sep  1 12:19:00 ip-10-1-1-5.us-west-2.compute.internal ASM: Request ID
8531451594106347485: GET / received on 2022-09-01 12:18:59 from IP 10.1.10.10 had
the following violations: HTTP protocol compliance failed,Bot Client Detected#015
```

SEND ILLEGAL REQUEST – BLOCKED BY NGINX

- Run the command `curl -H "Content-Length: -26" -k https://app.example.com` and ensure that the request is rejected by NGINX which returns the 400 Bad Request

```
ubuntu@ip-10-1-1-4:~$ curl -H "Content-Length: -26" -k https://app.example.com
<html>
<head><title>400 Bad Request</title></head>
<body>
<center><h1>400 Bad Request</h1></center>
<hr><center>nginx/1.21.6</center>
</body>
</html>
```

- Ensure that the log arrives at the server as previously

```
Sep  1 12:20:20 ip-10-1-1-5.us-west-2.compute.internal ASM: Request ID
8531451594106347995: GET N/A received on 2022-09-01 12:20:20 from IP 10.1.10.10
had the following violations: HTTP protocol compliance failed#015
```

TURN OFF BLOCKING

- Modify the `nap_vars.yaml` file to set the block to false using the command `vi nap_vars.yaml`

```
server_tech:
- tl:
  name: Unix/Linux

enforcementMode: blocking

block_violation:
- violation1:
  name: VIOL_HTTP_PROTOCOL
  alarm_switch: true
  block_switch: false
```

- Re-run the playbook with the command `ansible-playbook nap_playbook.yaml`
- Review the policy file using the command `more /etc/app_protect/conf/policy-viol_http_protocol.json`. Check that the violation is set to block but not alarm as below:

```
"name": "VIOL_HTTP_PROTOCOL",
"alarm": true,
"block": false
```

- Reload NGINX using the command `sudo nginx -s reload` and ensure there are no error messages shown

SEND ILLEGAL REQUEST – NOT BLOCKED, WITH LOG

- Run the command `curl -H "Host: 1.2.3.4" -k https://app.example.com` and ensure that the request **IS NOT** blocked

```
ubuntu@ip-10-1-1-4:~$ curl -H "Host: 1.2.3.4" -k https://app.example.com
<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" />
...
```

- Ensure that the log arrives at the server as previously

```
Sep  1 12:24:35 ip-10-1-1-5.us-west-2.compute.internal ASM: Request ID
10015829823190528429: GET / received on 2022-09-01 12:24:35 from IP 10.1.10.10 had the
following violations: HTTP protocol compliance failed,Bot Client Detected#015
```


TURN OFF LOGGING

- Modify the `nap_vars.yaml` file to set `alarm` to `false` and `block` to `true` 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/app_protect/conf/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
```

- Reload NGINX using the command `sudo nginx -s reload` and ensure there are no error messages shown

SEND ILLEGAL REQUEST – BLOCKED

- Run the command `curl -H "Host: 1.2.3.4" -k https://app.example.com` and ensure that the request **IS** blocked

```
</html>ubuntu@ip-10-1-1-4:~$ curl -H "Host: 1.2.3.4" -k http.example.com
<html><head><title>Request Rejected</title></head><body>The requested URL was rejected.
Please consult with your administrator.<br><br>Your support ID is:
5847644888864021580<br><br><a href='javascript:history.back();'>[Go
Back]</a></body></html>
```

- Ensure that the log arrives at the server as previously

```
Sep  1 12:34:36 ip-10-1-1-5.us-west-2.compute.internal ASM: Request ID
5847644888864021580: GET / received on 2022-09-01 12:34:36 from IP 10.1.10.10 had the
following violations: HTTP protocol compliance failed,Bot Client Detected#015
```

Hold on - if we turned off the alarm setting in the violation, why did we still receive a log entry?

This is because the log configuration `request_type` we have is “illegal” which includes requests with violations (ie either alerted or blocked)

Element	Meaning	Type/Values	Default
request_type	Log according to what App Protect detected in the request.	Enumerated values: <ul style="list-style-type: none">all: all requests, both legal and illegal.illegal: requests with violations (i.e., either alerted or blocked).blocked: requests with violations that were blocked.	all

Figure 2 Taken from <https://docs.nginx.com/nginx-app-protect/configuration-guide/configuration/#security-logs>

If the policy were in transparent mode, setting `alarm` to `true` would ensure that the violation is logged, or `false` to not log the violation.

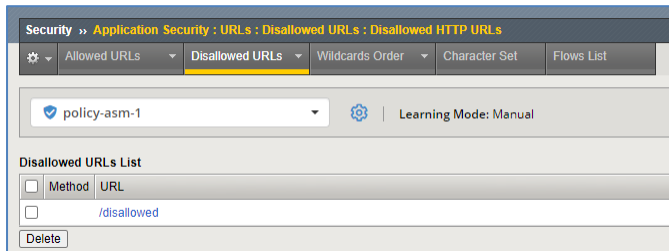
If the policy is in blocking mode and the entity is set to `block`, then any matching violations will be logged.

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 convert an existing ASM policy in XML format to be a JSON formatted App Protect policy. See <https://docs.nginx.com/nginx-app-protect/configuration-guide/configuration/#policy-converter> for more information about this Policy Converter

POLICY CONVERSION

The policy to be converted is a standard Fundamental AWAf security policy created on TMOS v16.x which has the /disallowed Disallowed URL configured, exported as XML:



- View the /disallowed URL configuration in the AWAf template using the command `grep /disallowed policy-asm-1.xml`

```
[centos@ip-10-1-1-5 ~]$ grep /disallowed policy-asm-1.xml
<last_change>URL Content Profile Any:Any on /disallowed [add] { audit: policy =
/Common/policy-asm-1, username = admin, client IP = 172.18.19.220 }</last_change>
<url method="*" protocol="HTTP" type="explicit" name="/disallowed">
```

- Convert the policy using the command `sudo /opt/app_protect/bin/convert-policy -i policy-asm-1.xml -o policy-viol_evasion.json`

```
[centos@ip-10-1-1-5 ~]$ sudo /opt/app_protect/bin/convert-policy -i policy-asm-1.xml -o
policy-viol_evasion.json
{"warnings":["Element '/graphql-profiles' is unsupported.", "Default header '*-bin' cannot
be deleted.", "Traffic Learning, Policy Building, and staging are unsupported.", "/signature-
settings/signatureStaging must be 'false' (was 'true').", "/csrf-urls/enforcementAction
value 'verify-csrf-token' is unsupported.", "/blocking-settings/violations/name value
'VIOL_BLOCKING_CONDITION' is unsupported.", "/blocking-settings/violations/name value
'VIOL_BRUTE_FORCE' is unsupported.", "/blocking-settings/violations/name value
'VIOL_CONVICTION' is unsupported.", "/blocking-settings/violations/name value
'VIOL_CSRF_EXPIRED' is unsupported.", "/blocking-settings/violations/name value
'VIOL_GEOLOCATION' is unsupported.", "/blocking-settings/violations/name value
'VIOL_HOSTNAME_MISMATCH' is unsupported.", "/blocking-settings/violations/name value
'VIOL_MALICIOUS_DEVICE' is unsupported.", "/blocking-settings/violations/name value
'VIOL_MALICIOUS_IP' is unsupported.", "/blocking-settings/violations/name value
'VIOL_REDIRECT' is unsupported.", "/blocking-settings/violations/name value
'VIOL_SERVER_SIDE_HOST' is unsupported.", "/blocking-settings/violations/name value
'VIOL_SESSION_AWARENESS' is unsupported.", "/blocking-settings/violations/name value
'VIOL_WEBSOCKET_BAD_REQUEST' is unsupported.", "/blocking-settings/violations/name value
'VIOL_WEBSOCKET_BINARY_MESSAGE_LENGTH' is unsupported.", "/blocking-
settings/violations/name value 'VIOL_WEBSOCKET_BINARY_MESSAGE_NOT_ALLOWED' is
unsupported.", "/blocking-settings/violations/name value 'VIOL_WEBSOCKET_EXTENSION' is
unsupported.", "/blocking-settings/violations/name value
'VIOL_WEBSOCKET_FRAMES_PER_MESSAGE_COUNT' is unsupported.", "/blocking-
settings/violations/name value 'VIOL_WEBSOCKET_FRAME_LENGTH' is unsupported.", "/blocking-
settings/violations/name value 'VIOL_WEBSOCKET_FRAME_MASKING' is unsupported.", "/blocking-
settings/violations/name value 'VIOL_WEBSOCKET_FRAMING_PROTOCOL' is
unsupported.", "/blocking-settings/violations/name value
'VIOL_WEBSOCKET_TEXT_MESSAGE_NOT_ALLOWED' is unsupported.", "/blocking-
settings/http-protocols/description value 'No Host header in HTTP/1.1 request' is
unsupported.", "/blocking-settings/http-protocols/description value 'CRLF characters before
request start' is unsupported.", "/blocking-settings/http-protocols/description value
'Content length should be a positive number' is unsupported.", "/blocking-settings/http-
protocols/description value 'Bad host header value' is
unsupported.", "/general/enableEventCorrelation must be 'false' (was 'true').", "Element
'/websocket-urls' is unsupported.", "Element '/redirection-protection' is
unsupported.", "Element '/gwt-profiles' is unsupported.", "/signature-sets/learn value true
```

```
is unsupported", "/performStaging must be 'false' (was  
'true')."], "file_size": 22901, "filename": "/home/centos/policy-  
viol_evasion.json", "completed_successfully": true}
```

- Check that the policy is created with the command `more policy-viol_evasion.json`

```
[centos@ip-10-1-1-5 ~]$ more policy-viol_evasion.json
{
  "policy" : {
    "blocking-settings" : {
      "evasions" : [
        {
          "description" : "Directory traversals",
          "enabled" : false
        },
        {
          "description" : "Multiple decoding",
          "enabled" : false,
          "maxDecodingPasses" : 3
        }
      ]
    }
  }
}
```

- Note that VIOL_EVASION is set to block:

```
"block": true,  
"name": "VIOL_CSRF"  
},  
  
{"block": true,  
"name": "VIOL_DATA_GUARD"  
},  
  
{"block": true,  
"name": "VIOL_ENCODING"  
},  
  
{"block": true,  
"name": "VIOL_EVASION"  
},  
  
{"block": true,  
"name": "VIOL_FILETYPE"  
},  
  
{"block": true,  
"name": "VIOL_FILE_UPLOAD"  
},  
  
{"block": true,  
"name": "VIOL_FILE_UPLOAD_IN_BODY"  
}
```

- Note that the /disallowed Disallowed URL is converted:

```
{
  "signature-settings" : {},
  "template" : {
    "name" : "POLICY_TEMPLATE_NGINX_BASE"
  },
  "urls" : [
    {
      "metacharsOnUrlCheck" : false,
      "method" : "*",
      "name" : "*",
      "protocol" : "http",
      "type" : "wildcard"
    },
    {
      "disallowFileUploadOfExecutables" : false,
      "isAllowed" : false,
      "mandatoryBody" : false,
      "method" : "*",
      "name" : "/disallowed",
      "protocol" : "http",
      "type" : "explicit"
    }
  ],
  "xml-profiles" : [
```

- Copy the policy file to the App Protect configuration directory with the command `sudo cp policy-viol_evasion.json /etc/app_protect/conf`
- 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/app_protect/conf/policy-viol_evasion.json";
    app_protect_security_log_enable on;
    app_protect_security_log "/etc/app_protect/conf/log-illegal.json"
    syslog:server=10.1.20.11:514;
}
```

- Reload NGINX using the command `sudo nginx -s reload` and ensure there are no error messages shown

SEND ILLEGAL REQUEST – NOT BLOCKED BY WAF

In this request, we are performing multiple decoding - %25 is the ASCII code for % and %2E is the code for "." , so once this is decoded the URL becomes "%.". ie multiple decoding

- Run the command `curl -k https://app.example.com/%25252E` and ensure that the response is **404 Not Found** ie request **NOT** blocked

```
/html>ubuntu@ip-10-1-1-4:~$ curl -k https://app.example.com/%25252E
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL /%252E was not found on this server.</p>
</body></html>
```

- Ensure that the log **DOES NOT** arrive at the server

ENABLE MULTIPLE DECODING EVASION

- Edit the security policy with the command `sudo vi /etc/app_protect/conf/policy-viol_evasion.json`
- Under policy>blocking-settings>evasions, set Multiple decoding to be enabled

```
{
  "policy": {
    "blocking-settings": {
      "evasions": [
        {
          "description": "Directory traversals",
          "enabled": false
        },
        {
          "description": "Multiple decoding",
          "enabled": true,
          "maxDecodingPasses": 3
        },
        {
          "description": "%u decoding",
          "enabled": false
        }
      ]
    }
  }
}
```

- Reload NGINX using the command `sudo nginx -s reload` and ensure there are no error messages shown
- Run the command `curl -k https://app.example.com/%25252E` and ensure that the response is **BLOCKED**

```
ubuntu@ip-10-1-1-4:~$ curl -k https://app.example.com/%25252E
<html><head><title>Request Rejected</title></head><body>The requested URL was rejected.
Please consult with your administrator.<br><br>Your support ID is:
1185827382183798318<br><br><a href='javascript:history.back(); '>[Go
Back]</a></body></html>
```

- Ensure that the log arrives at the server as previously

```
Sep  2 13:15:29 ip-10-1-1-5.us-west-2.compute.internal ASM: Request ID
1185827382183798318: GET /. received on 2022-09-02 13:15:29 from IP 10.1.10.10 had the
following violations: Evasion technique detected#015
```

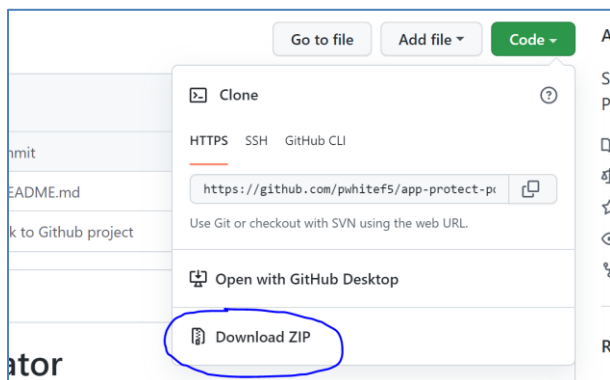
Note that the violation VIOL_EVASION is enabled globally, but individual checks are enabled separately. In this way, if there were an issue with a false positive on an evasion technique, this individual check can be disabled whilst still checking for other evasion techniques.

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 an open source tool to initially create the policy.

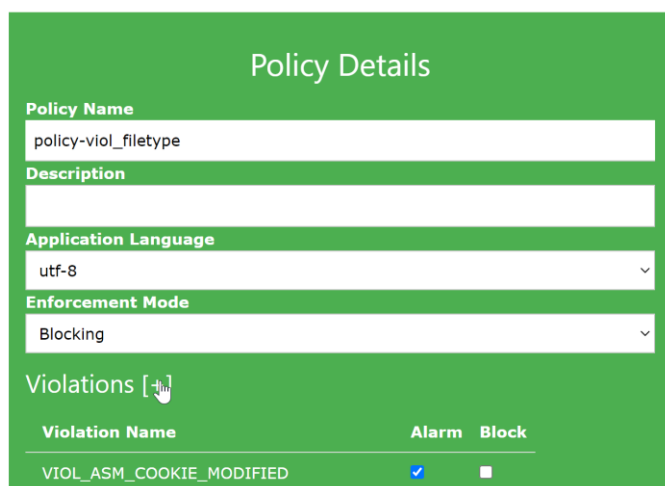
NGINX CONFIGURATION

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



- 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

NGINX App Protect Policy Creator



Violation Name	Alarm	Block
VIOL_ASM_COOKIE_MODIFIED	<input checked="" type="checkbox"/>	<input type="checkbox"/>

- 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/app_protect/conf/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/app_protect/conf/policy-viol_filetype.json";
    app_protect_security_log_enable on;
    app_protect_security_log "/etc/app_protect/conf/log-illegal.json"
    syslog:server=10.1.20.11:514;
}

```

- Reload NGINX 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

```

/ubuntu@ip-10-1-1-4:~$ curl -k https://app.example.com/index.php
<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" />
<script language="javascript">
    function showCookieLink() {
        var ele = document.getElementById("CookieLink");
        ele.style.display = "block";
    }
}

```

Note that we still expect the log referring to Bot Client Detected, but not Illegal file type

SEND ILLEGAL REQUEST – BLOCKED

- Run the command `curl -k https://app.example.com/test.exe` and ensure that the request **IS** blocked

```

ubuntu@ip-10-1-1-4:~$ curl -k https://app.example.com/test.exe
<html><head><title>Request Rejected</title></head><body>The requested URL was rejected.
Please consult with your administrator.<br><br>Your support ID is:
18284023369949473805<br><br><a href='javascript:history.back();'>[Go
Back]</a></body></html>

```

- Ensure that the log arrives at the server as previously

```

Sep  2 13:35:01 ip-10-1-1-5.us-west-2.compute.internal ASM: Request ID
18284023369949473805: GET /test.exe received on 2022-09-02 13:35:00 from IP 10.1.10.10 had
the following violations: Illegal file type,Bot Client Detected#015

```

Note that .exe is on the standard list of disallowed filetypes which is the reason why this request is blocked. View the list of disallowed filetypes here: <https://docs.nginx.com/nginx-app-protect/configuration-guide/configuration/#disallowed-file-types>

NGINX CONFIGURATION TO CHANGE THE DEFAULT POLICY

- Modify the file policy- viol_evasion.json to match the content below, and reload nginx:

```
{
  "modifications": [
    {
      "entityChanges": {
        "type": "wildcard"
      },
      "entity": {
        "name": "*"
      },
      "entityType": "filetype",
      "action": "delete"
    }
  ],
  "policy": {
    "name": "policy-viol_filetype",
    "description": "",
    "template": {
      "name": "POLICY_TEMPLATE_NGINX_BASE"
    },
    "applicationLanguage": "utf-8",
    "enforcementMode": "blocking",
    "filetypes": [
      {
        "name": "html",
        "type": "explicit",
        "allowed": true
      }
    ],
    "blocking-settings": {
      "violations": [
        {
          "name": "VIOL_ASM_COOKIE_MODIFIED",
          "alarm": true,
          "block": false
        }
      ],
    },
    ...
  }
}
```

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

This part is adding html as an explicit allowed filetype

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

```
ubuntu@ip-10-1-1-4:~$ curl -k https://app.example.com/badlinks.html
<html>
<head>
<title>Broken Links</title>
<meta http-equiv="pragma" content="no-cache" />
</head>
```

SEND ILLEGAL REQUEST – BLOCKED

- Run the command `curl -k https://app.example.com/index.php` and ensure that the request **IS blocked**

```
ubuntu@ip-10-1-1-4:~$ curl -k https://app.example.com/index.php
<html><head><title>Request Rejected</title></head><body>The requested URL was rejected.
Please consult with your administrator.<br><br>Your support ID is:
1185827382183799338<br><br><a href='javascript:history.back();'>[Go
Back]</a></body></html>
```

- Ensure that the log arrives at the server as previously

```
Sep  2 13:43:24 ip-10-1-1-5.us-west-2.compute.internal ASM: Request ID
1185827382183799338: GET /index.php received on 2022-09-02 13:43:24 from IP
10.1.10.10 had the following violations: Illegal request length,Illegal URL
length,Illegal file type,Bot Client Detected#015
```


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/app_protect/conf/NginxDefaultPolicy.json";
    app_protect_security_log_enable on;
    app_protect_security_log "/etc/app_protect/conf/log-illegal.json"
    syslog:server=10.1.20.11:514;
```

- Add a new location block as below:

```
server {
    listen 10.1.10.100:443 ssl default_server;
    server_name app.example.com;
    ssl_certificate /etc/ssl/certs/dummy.crt;
    ssl_certificate_key /etc/ssl/certs/dummy.key;
    location / {
        proxy_pass http://app_example_com;
    }
    location /basic {
        app_protect_policy_file "/etc/app_protect/conf/policy-viol_filetype.json";
        proxy_pass http://app_example_com;
    }
}
```

- Reload NGINX 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

```
ubuntu@ip-10-1-1-4:~$ curl -k https://app.example.com/index.php
<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" />
```

SEND ILLEGAL REQUEST – BLOCKED

- Run the command `curl -k https://app.example.com/basic/index.php` and ensure that the request **IS** blocked

```
ubuntu@ip-10-1-1-4:~$ curl -k https://app.example.com/basic/index.php
<html><head><title>Request Rejected</title></head><body>The requested URL was rejected.
Please consult with your administrator.<br><br>Your support ID is:
6853458070205590127<br><br><a href='javascript:history.back(); '>[Go
Back]</a></body></html>
```

- Ensure that the log arrives at the server as previously

```
Sep  2 14:02:21 ip-10-1-1-5.us-west-2.compute.internal ASM: Request ID
6853458070205590127: GET /basic/index.php received on 2022-09-02 14:02:20 from IP
10.1.10.10 had the following violations: Illegal request length,Illegal URL length,Illegal
file type,Bot Client Detected#015
```

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 change the global policy filename to be incorrect, as shown below:

```
http {
    app_protect_enable on;

    app_protect_policy_file "/etc/app_protect/conf/NginxDefaultPolicywrong.json";
    app_protect_security_log_enable on;
    app_protect_security_log "/etc/app_protect/conf/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 ~]$ sudo nginx -s reload
{
  "completed_successfully": false,
  "error_message": "Failed to read /etc/app_protect/conf/NginxDefaultPolicywrong.json:
open /etc/app_protect/conf/NginxDefaultPolicywrong.json: no such file or directory",
  "error_line_number": 16,
  "software_version": "10.87.0"
}
nginx: [error] APP_PROTECT { "event": "configuration_load_failure", "software_version":
"3.954.0", "completed_successfully":false,"error_message":"Failed to read
/etc/app_protect/conf/NginxDefaultPolicywrong.json: open
/etc/app_protect/conf/NginxDefaultPolicywrong.json: no such file or
directory","error_line_number":16,"software_version":"10.87.0"}
```

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:

```
2022/09/02 14:05:46 [error] 3157#3157: APP_PROTECT { "event":
"configuration_load_failure", "software_version": "3.954.0",
"completed_successfully":false,"error_message":"Failed to read
/etc/app_protect/conf/NginxDefaultPolicywrong.json: open
/etc/app_protect/conf/NginxDefaultPolicywrong.json: no such file or
directory","error_line_number":16,"software_version":"10.87.0"}
```

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 with `sudo nginx -s reload`, 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 the `/etc/app_protect/conf/policy-viol_filetype.json` policy to remove the double-quote at the start of filetypes using the command `sudo vi /etc/app_protect/conf/policy-viol_filetype.json` as per below:

```
filetypes": [
  {
    "name": "html",
    "type": "explicit",
    "allowed": true
  }
]
```

- Restart the nginx daemon using the command `sudo nginx -s reload` and note the failure message as below:

```
[centos@ip-10-1-1-5 ~]$ sudo nginx -s reload
{
  "error_message": "Failed to import Policy '/etc/app_protect/conf/policy-viol_filetype.json' from '/etc/app_protect/conf/policy-viol_filetype.json': Failed to parse JSON Policy\n\"' expected, at character offset 493 (before \"filetypes\": [\n...\")",
  "completed_successfully": false,
  "error_line_number": 46
}
nginx: [error] APP_PROTECT { "event": "configuration_load_failure", "software_version": "3.954.0", "error_message": "Failed to import Policy '/etc/app_protect/conf/policy-viol_filetype.json' from '/etc/app_protect/conf/policy-viol_filetype.json': Failed to parse JSON Policy\n\"' expected, at character offset 493 (before \"filetypes\": [\n...\")", "completed_successfully": false, "error_line_number": 46}
```

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:

```
2022/09/02 14:14:27 [error] 3230#3230: APP_PROTECT { "event": "configuration_load_failure", "software_version": "3.954.0", "error_message": "Failed to import Policy '/etc/app_protect/conf/policy-viol_filetype.json' from '/etc/app_protect/conf/policy-viol_filetype.json': Failed to parse JSON Policy\n\"' expected, at character offset 493 (before \"filetypes\": [\n...\")", "completed_successfully": false, "error_line_number": 46}
```

- To check the policy file for JSON errors, run the command `cat /etc/app_protect/conf/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 ~]$ cat /etc/app_protect/conf/policy-viol_filetype.json | python -m json.tool
Expecting property name: line 22 column 9 (char 493)
```

- To find the error in the file, open the policy using the command `sudo vi /etc/app_protect/conf/policy-viol_filetype.json` and use the command `:16` to jump to the relevant line, use `:goto 441` to jump to the character offset

```

    },
    "applicationLanguage": "utf-8",
    "enforcementMode": "blocking",
    "filetypes": [
      {
        "name": "html",
        "type": "explicit",

```

- Fix the error by replacing the double-quote in the policy file and reloading nginx with the command `sudo nginx -s reload`
- Re-check the file JSON format is correct as below ie no errors:

```

[centos@ip-10-1-1-5 ~]$ cat /etc/app_protect/conf/policy-viol_filetype.json | python -m
json.tool
{
  "modifications": [
    {
      "action": "delete",
      "entity": {
        "name": "*"
      },
      "entityChanges": {
        "type": "wildcard"
      }
    }
  ]
}

```

- 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 ~]$ sudo nginx -s reload
{
  "error_message": "Failed to import Policy 'policy-viol_filetype' from
'/etc/app_protect/conf/policy-viol_filetype.json': Could not parse/validate the File Type.
Invalid value '12' for field 'allowed'.",
  "completed_successfully": false,
  "error_line_number": 46
}
nginx: [error] APP_PROTECT { "event": "configuration_load_failure", "software_version":
"3.954.0", "error_message":"Failed to import Policy 'policy-viol_filetype' from
'/etc/app_protect/conf/policy-viol_filetype.json': Could not parse/validate the File Type.
Invalid value '12' for field
'allowed'.", "completed_successfully": false, "error_line_number": 46}

```

- 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/app_protect/conf/policy-viol_http_protocol.json";
    app_protect_security_log_enable on;
    app_protect_security_log "/etc/app_protect/conf/log-illegal.json"
    syslog:server=10.1.20.11:514;
```

- Reload nginx and ensure no errors are seen

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`

```
[centos@ip-10-1-1-5 ~]$ sudo /bin/bash -c '/opt/app_protect/bin/set_active.pl -g' nginx
0
```

- Capture the debug file output using the command `tail -f /var/log/app_protect/bd-socket-plugin.log`

```
[centos@ip-10-1-1-5 ~]$ tail -f /var/log/app_protect/bd-socket-plugin.log
New ALL level: TS_ERR
New ALL level: TS_CRIT
New ALL level: TS_WARNING
New ALL level: TS_NOTICE
New ALL file num: 2
New module: IO_PLUGIN
New level: TS_DEBUG
New file num: 2
BD_MISC|NOTICE|Sep 02 14:35:40.273|0584|/builds/d3347dec/11/waf/waf-general/secore/common/bd_tbls/table_funcs.cpp:2799|manifest curr_generation: 42
BD_MISC|NOTICE|Sep 02 14:35:40.273|0584|/builds/d3347dec/11/waf/waf-general/secore/common/bd_tbls/table_funcs.cpp:2806|{"component":"BD","datetime":"2022-09-02T14:35:39Z","jobId":"5a3cfbe92829f4bdadb997c3b6de8415","jobStartDatetime":"2022-09-02T14:35:39Z","jobStatus":"completed"}
```

- On the client, run the command `curl -H "Host: 1.2.3.4" -k https://app.example.com` and view the debug output

```
IO_PLUGIN|DEBUG |Sep 02 14:36:06.994|3276|socket_plugin_events.c:0031|read 506 bytes
IO_PLUGIN|DEBUG |Sep 02 14:36:06.994|3276|socket_plugin_events.c:0259|msg size: 482, evt: 269:TMEVT_REQUEST
IO_PLUGIN|DEBUG |Sep 02 14:36:06.994|3276|socket_plugin_events.c:0050|ctx id: 0 is_server: 0
IO_PLUGIN|DEBUG |Sep 02 14:36:06.994|3276|socket_plugin_events.c:0065|vs_name_len: 23, logging_str_len: 1
IO_PLUGIN|DEBUG |Sep 02 14:36:06.994|3276|socket_plugin_events.c:0075|vs_name: 37-app.example.com:1-/, logging_str:
IO_PLUGIN|DEBUG |Sep 02 14:36:06.994|3276|socket_plugin_events.c:0079|hdrs_count: 5
IO_PLUGIN|DEBUG |Sep 02 14:36:06.994|3276|socket_plugin_events.c:0085|hdr 0 len: 16
IO_PLUGIN|DEBUG |Sep 02 14:36:06.994|3276|socket_plugin_events.c:0090|hdr content: GET / HTTP/1.1

IO_PLUGIN|DEBUG |Sep 02 14:36:06.994|3276|socket_plugin_events.c:0085|hdr 1 len: 15
IO_PLUGIN|DEBUG |Sep 02 14:36:06.994|3276|socket_plugin_events.c:0090|hdr content: Host: 1.2.3.4
```

- 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