

VM-Series for GCP



GCP XFF Blocking Deployment Guide

How to leverage Cloud Armor to restrict traffic based upon XFF headers in GCP

<http://www.paloaltonetworks.com>

Table of Contents

Version History	3
Overview.....	4
Required Components.....	5
Configure Cloud Armor	6
Create a Service Account Key	9
Prepare the Worker Node	11
Configure the Firewalls.....	15
Test/Verify	17
Troubleshooting	22

Version History

Version number	Comments
1.0	Initial Draft

Overview

When using the HTTP/HTTPS load balancer in the Google Cloud Environment, the load balancer will perform a source NAT on the original client IP, hiding it behind an IP address in the following ranges: 130.211.0.0/22 and 35.191.0.0/16. In addition to randomly changing the IP address used for NAT'ing each connection, IP addresses from these pools are also used for the source of health checks. To compensate for the loss of fidelity, the load balancer inserts an X-Forwarded-For (XFF) header into the request. The XFF header is of format:

X-Forwarded-For: <unverified IP(s)>, <immediate client IP>, <global forwarding rule external IP>, <proxies running in GCP>

And represents a comma-separated list of IP addresses appended by the intermediaries the request traveled through.

As the firewall sees multiple IP addresses, it is unable to distinguish the original source for the purposes of filtering future events in the event a threat is detected. One solution to this is to use a worker node to extract the original source from the XFF and use GCP Cloud Armor to filter the bad actor at the edge.

GCP Cloud Armor is a denial of service/web attack prevention tool that is integrated into the HTTP(S) load balancer and is a good option for mitigating certain types of web-based attacks. More information can be found here:

<https://cloud.google.com/armor/>

Demo Steps

- 1) This demo uses detectable threats (in this case SQL Injection attack) between a client browser and the web server to trigger action-oriented log forwarding.
- 2) The firewall detects the attack and forwards the Threat log to the worker node via an HTTP log forwarding action.
- 3) The worker node queries the firewall for the URL filtering log based on the sessionID, NAT Source Port, and received time of the detected threat. The response includes the IP address in the X-Forwarded-For HTTP header.
- 4) The worker node extracts the IP of the attacker from the X-Forwarded-For field.
The worker node determines the correct rule priority and adds a rule to the Cloud Armor security policy.

Support Policy

This document is released under an as-is, best effort, support policy. The associated scripts should be seen as community supported and Palo Alto Networks will contribute our expertise as and when possible. We do not provide technical support or help in using or troubleshooting the components of the project through our normal support options such as Palo Alto Networks support teams, or ASC (Authorized Support Centers) partners and backline support options. The

underlying product used (the VM-Series firewall) by the scripts or templates are still supported, but the support is only for the product functionality and not for help in deploying or using the template or script itself.

Unless explicitly tagged, all projects or work posted in our GitHub repository (at <https://github.com/PaloAltoNetworks/googlecloud>) or sites other than our official Downloads page on <https://support.paloaltonetworks.com> are provided under the best effort policy.

Required Components

This example will use the following components:

- GCP Cloud Armor
- HTTP(S) Load Balancer
- Palo Alto Networks Firewall(s) with URLF and Threat subscriptions (duh)
- TCP Load Balancer
- Linux Webservers
- Linux Worker Node

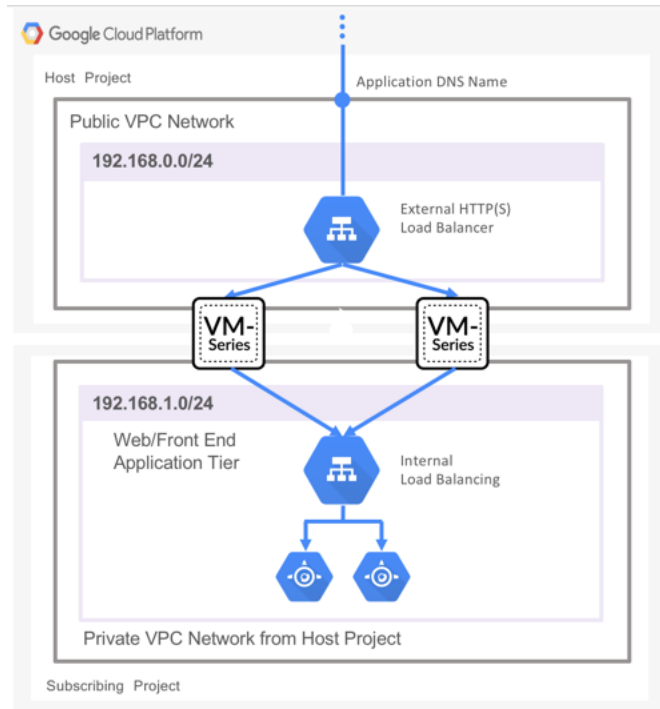
The following table lists the suggested instance sizes for this demonstration:

Instance name	Machine Type
Web Servers	f1-micro
Worker Node	f1-micro
VM Series Firewalls	n1-standard-4

Larger instances may be required/substituted as necessary.

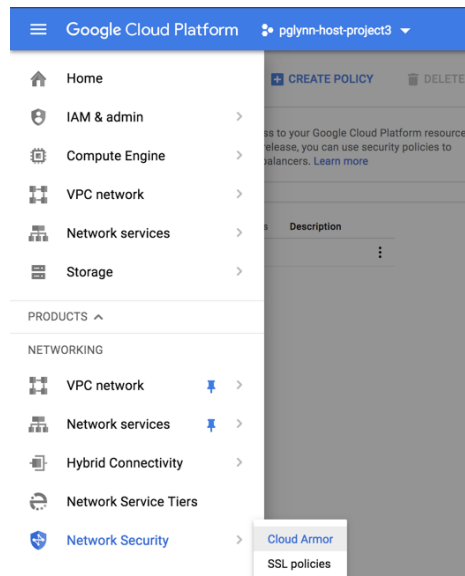
Note: There are costs associated with each machine type launched, please refer to the Google instance pricing page <https://cloud.google.com/compute/pricing>

It is assumed that the instances have already been deployed, configured, and verified to be passing traffic as expected. The topology used in this example resembles the following:



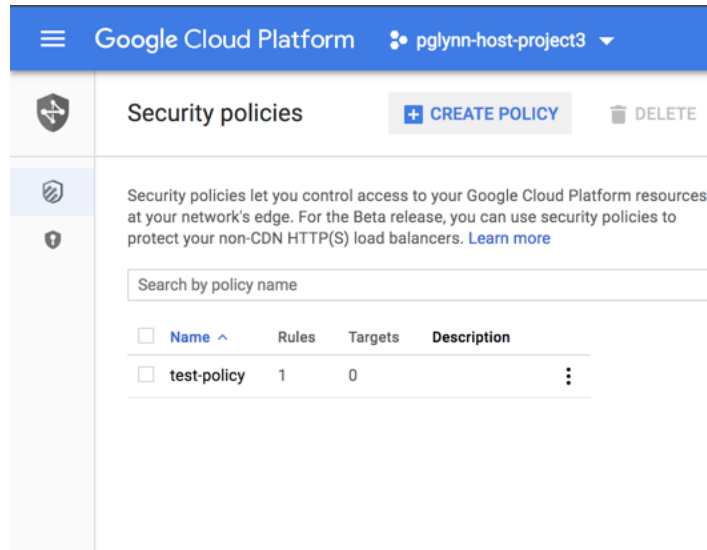
Configure Cloud Armor

1. Create the initial Cloud Armor security policy. Navigate to **Networking > Network Security > Cloud Armor**.



2. Click **CREATE POLICY**.

Palo Alto Networks GCP XFF Blocking Deployment Guide



3. Specify a Name.

The screenshot shows the 'Create security policy' form. The 'Name' field is filled with 'protect-web-apps'. The 'Description' field is empty. The 'Default rule action' is set to 'Deny'. The 'Deny status' is set to '403 (Forbidden)'. The 'Next step' button is visible at the bottom of the form.

1 Configure policy

Name: protect-web-apps

Description (Optional):

Default rule action: Deny

Deny status: 403 (Forbidden)

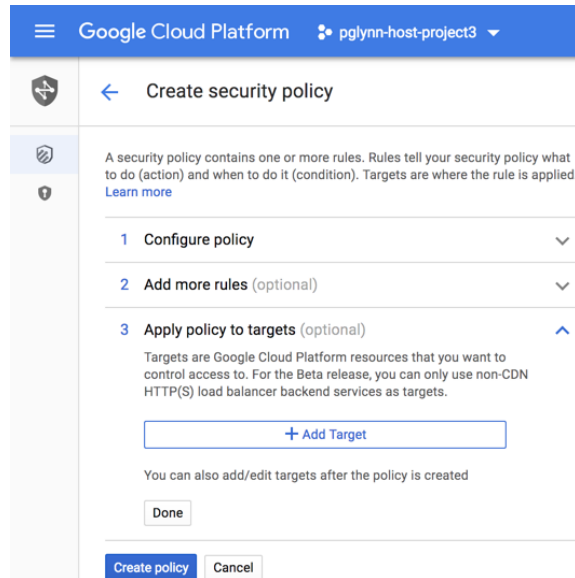
Next step

2 Add more rules (optional)

3 Apply policy to targets (optional)

4. Click **Apply policy to targets**.

Palo Alto Networks GCP XFF Blocking Deployment Guide



Google Cloud Platform pglynn-host-project3

Create security policy

A security policy contains one or more rules. Rules tell your security policy what to do (action) and when to do it (condition). Targets are where the rule is applied. [Learn more](#)

- 1 Configure policy
- 2 Add more rules (optional)
- 3 Apply policy to targets (optional)

Targets are Google Cloud Platform resources that you want to control access to. For the Beta release, you can only use non-CDN HTTP(S) load balancer backend services as targets.

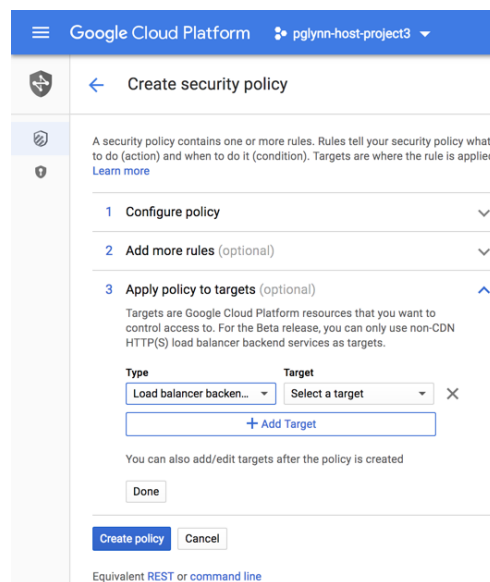
[+ Add Target](#)

You can also add/edit targets after the policy is created

[Done](#)

[Create policy](#) [Cancel](#)

5. Click **+ Add Target**.



Google Cloud Platform pglynn-host-project3

Create security policy

A security policy contains one or more rules. Rules tell your security policy what to do (action) and when to do it (condition). Targets are where the rule is applied. [Learn more](#)

- 1 Configure policy
- 2 Add more rules (optional)
- 3 Apply policy to targets (optional)

Targets are Google Cloud Platform resources that you want to control access to. For the Beta release, you can only use non-CDN HTTP(S) load balancer backend services as targets.

Type: Load balancer backen... Target: Select a target

[+ Add Target](#)

You can also add/edit targets after the policy is created

[Done](#)

[Create policy](#) [Cancel](#)

Equivalent REST or [command line](#)

6. Select the public HTTP(S) Load Balancer.

Google Cloud Platform pglynn-host-project3

Create security policy

A security policy contains one or more rules. Rules tell your security policy what to do (action) and when to do it (condition). Targets are where the rule is applied. [Learn more](#)

- 1 Configure policy
- 2 Add more rules (optional)
- 3 Apply policy to targets (optional)

Targets are Google Cloud Platform resources that you want to control access to. For the Beta release, you can only use non-CDN HTTP(S) load balancer backend services as targets.

Type	Target
Load balancer backen...	external-backend-service

[+ Add Target](#)

You can also add/edit targets after the policy is created

[Done](#)

[Create policy](#) [Cancel](#)

Equivalent REST or command line

7. Click **Create policy**. It will take a few moments to create the policy.

Google Cloud Platform pglynn-host-project3

Security policies

[+ CREATE POLICY](#) [DELETE](#)

Security policies let you control access to your Google Cloud Platform resources at your network's edge. For the Beta release, you can use security policies to protect your non-CDN HTTP(S) load balancers. [Learn more](#)

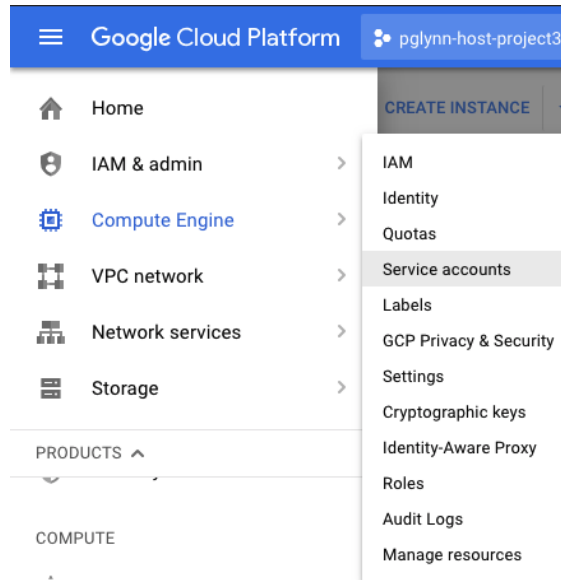
Search by policy name

Name	Rules	Targets	Description
protect-web-apps	1	1	
test-policy	1	0	

Create a Service Account Key

1. Navigate to **IAM & admin > Service accounts**.

Palo Alto Networks GCP XFF Blocking Deployment Guide



2. Click on the vertical ellipses beside the default service account and select **Create key**.

Service accounts for project "pglynn-host-project3"

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. [Learn more](#)

Filter table

<input type="checkbox"/>	Email	Name ↑	Key ID	Delete key	Key creation date	Actions
<input type="checkbox"/>	67504155973-compute@developer.gserviceaccount.com	Compute Engine default service account	No keys			<div><div></div><div>Edit</div><div>Delete</div><div>Create key</div></div>

3. Leave the key type as JSON and click **CREATE**.

Create private key for "Compute Engine default service account"

Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

Key type

☒ JSON

Recommended

☐ P12

For backward compatibility with code using the P12 format

CANCEL

CREATE

4. Save the key to a secure location as it allows API access to GCP resources.
5. (optional) Rename the key to reflect the service account to which it is attached.

Prepare the Worker Node

1. Deploy a worker node with the following settings:

Machine type: F1-micro

Network: FW management subnet

Internal IP: static

External IP (optional): ephemeral

The screenshot shows the configuration for a worker node in the Google Cloud Platform console. The node is named 'worker' and is in the 'us-central1-c' zone. It is an F1-micro machine type with 1 vCPU and 0.6 GB memory. The CPU platform is Intel Haswell. The node is connected to the 'management' network and the 'management1' subnet. The primary internal IP is 'worker-ip (10.5.0.5)'. The external IP is '35.239.140.70 (ephemeral)'. The network tier is 'Premium' and IP forwarding is 'Off'. The creation time is 'Jul 3, 2018, 6:46:22 PM'. The network interfaces table shows the 'nic0' interface connected to the 'management' network and the 'management1' subnet. The public DNS PTR record is 'None'. The firewalls section shows that HTTP and HTTPS traffic are allowed.

Name	Network	Subnetwork	Primary internal IP	Alias IP ranges	External IP	Network Tier	IP forwarding	Network details
nic0	management	management1	worker-ip (10.5.0.5)	—	35.239.140.70 (ephemeral)	Premium	Off	View details

2. Copy the service account key and Python code to the worker node.

```
1. bash
DFWMACPofQG8WL:python pglynn$ ls
67504155973-compute@developer.gserviceaccount.com.json
gcp-aolf.py
DFWMACPofQG8WL:python pglynn$ scp * 35.239.140.70
35.239.140.70: No such file or directory
DFWMACPofQG8WL:python pglynn$ scp * 35.239.140.70:
Warning: Permanently added '35.239.140.70' (ECDSA) to the list of known hosts.
67504155973-compute@developer.gserviceaccount 100% 2330 33.8KB/s 00:00
gcp-aolf.py 100% 9663 53.8KB/s 00:00
DFWMACPofQG8WL:python pglynn$
```

3. Connect to the worker node and “su” to root.

Palo Alto Networks GCP XFF Blocking Deployment Guide

```
1. pglynn@worker: ~ (ssh)
DFWMACPoFQG8WL:python pglynn$ ssh 35.239.140.70
Warning: Permanently added '35.239.140.70' (ECDSA) to the list of known hosts.
Linux worker 4.9.0-6-amd64 #1 SMP Debian 4.9.88-1+deb9u1 (2018-05-07) x86_64

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Jul 5 14:36:52 2018 from 47.183.68.140
pglynn@worker:~$ ls
67504155973-compute@developer.gserviceaccount.com.json  gcp-aolf.py
pglynn@worker:~$ sudo su -
root@worker:~#
```

4. Copy the Python script and service account key to root's home directory

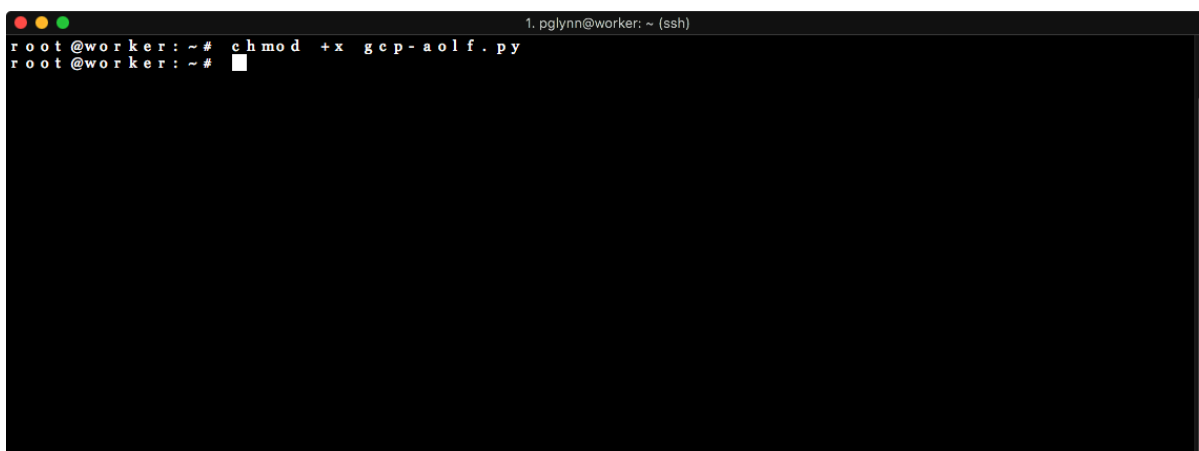
```
1. pglynn@worker: ~ (ssh)
root@worker:~# cp /home/pglynn/* .
root@worker:~# ls
67504155973-compute@developer.gserviceaccount.com.json  gcp-aolf.py
root@worker:~#
```

5. Create an environment variable pointing to the service account key. This is necessary as the Python script will need the key to authenticate to the GCP environment. The format is:

`export GOOGLE_APPLICATION_CREDENTIALS=/path/to/service_account_key.json`

```
1. pglynn@worker: ~ (ssh)
root@worker:~# cp /home/pglynn/* .
root@worker:~# ls
67504155973-compute@developer.gserviceaccount.com.json  gcp-aolf.py
root@worker:~# export GOOGLE_APPLICATION_CREDENTIALS=/root/67504155973-compute\@
developer.gserviceaccount.com.json
root@worker:~#
```

6. Add execute permission to the Python script.

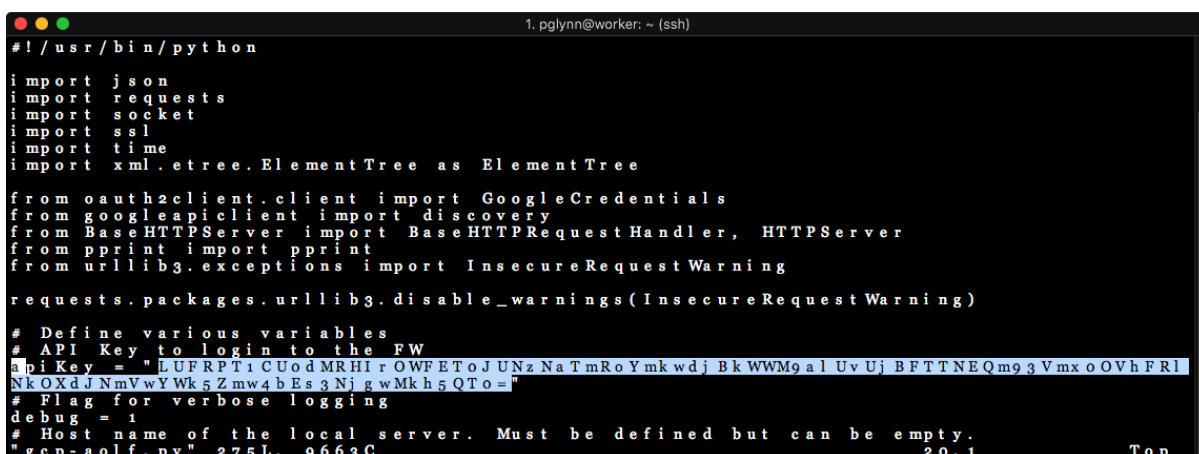


```

1. pglynn@worker: ~ (ssh)
root@worker: ~# chmod +x gcp-aolf.py
root@worker: ~#

```

7. Edit the Python script and replace the FW API key with the one specific to your implementation.



```

1. pglynn@worker: ~ (ssh)
#!/usr/bin/python

import json
import requests
import socket
import ssl
import time
import xml.etree.ElementTree as ElementTree

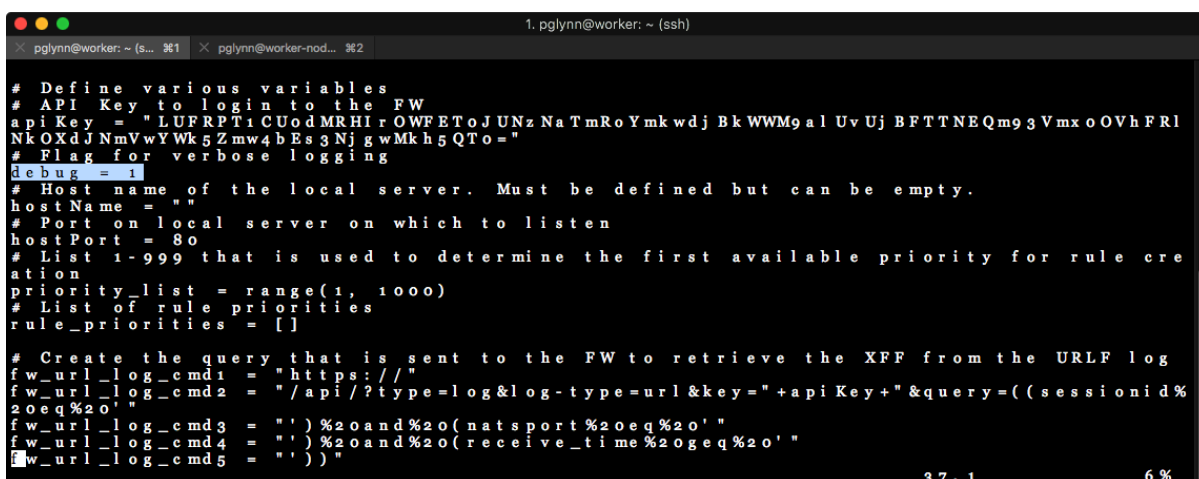
from oauth2client.client import GoogleCredentials
from googleapiclient import discovery
from BaseHTTPServer import BaseHTTPRequestHandler, HTTPServer
from pprint import pprint
from urllib3.exceptions import InsecureRequestWarning

requests.packages.urllib3.disable_warnings(InsecureRequestWarning)

# Define various variables
# API Key to login to the FW
apiKey = "LUFRPT1CUodMRHirOWFEToJUNzNaTmRoYmkwdjBkWWM9alUvUjBFTTNEQm93VmxoOVhFRlNkOXdJNmVwYWk5Zmw4bEs3NjgwMkh5QTo="
# Flag for verbose logging
debug = 1
# Host name of the local server. Must be defined but can be empty.
"gcp-aolf.py" 275L, 9663C

```

8. (optional) To monitor script execution or for debugging purposes, set the debug flag to "1".



```

1. pglynn@worker: ~ (ssh)
pglynn@worker: ~ (s... %1) pglynn@worker-nod... %2
# Define various variables
# API Key to login to the FW
apiKey = "LUFRPT1CUodMRHirOWFEToJUNzNaTmRoYmkwdjBkWWM9alUvUjBFTTNEQm93VmxoOVhFRlNkOXdJNmVwYWk5Zmw4bEs3NjgwMkh5QTo="
# Flag for verbose logging
debug = 1
# Host name of the local server. Must be defined but can be empty.
hostName = ""
# Port on local server on which to listen
hostPort = 80
# List 1-999 that is used to determine the first available priority for rule creation
priority_list = range(1, 1000)
# List of rule priorities
rule_priorities = []

# Create the query that is sent to the FW to retrieve the XFF from the URLF log
fw_url_log_cmd1 = "https://"
fw_url_log_cmd2 = "/api/?type=log&log-type=url&key="+apiKey+"&query=((sessionid%20eq%20'"
fw_url_log_cmd3 = "')%20and%20(natsport%20eq%20'"
fw_url_log_cmd4 = "')%20and%20(receive_time%20geq%20'"
fw_url_log_cmd5 = "')%20and%20(priority%20eq%20'"

37, 1 6%

```

9. Install pip.

```

1. pglynn@worker: ~ (ssh)
pglynn@worker: ~ (ssh)
root@worker: ~# apt-get install python-pip
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  binutils build-essential bzip2 cpp cpp-6 dbus dpkg-dev fakeroot g++ g++-6
  gcc gcc-6 gir1.2-glib-2.0 libalgorithm-diff-perl libalgorithm-diff-xs-perl
  libalgorithm-merge-perl libasan3 libatomic1 libc-dev-bin libc6-dev libcc1-0
  libcilkrts5 libdbus-1-3 libdbus-glib-1-2 libdpkg-perl libexpat1-dev
  libfakeroot libfile-fcntllock-perl libgcc-6-dev libgirepository-1.0-1
  libglib2.0-0 libglib2.0-data libgomp1 libicu57 libisl15 libitm1 liblsano
  libmpc3 libmpfr4 libmpx2 libperl5.24 libpython-all-dev libpython-dev
  libpython2.7 libpython2.7-dev libquadmath0 libstdc++-6-dev libtsano
  libubsano libxml2 linux-libc-dev make manpages manpages-dev patch perl
  perl-modules-5.24 python-all python-all-dev python-cffi-backend
  python-crypto python-cryptography python-dbus python-dev python-enum34
  python-gi python-idna python-ipaddress python-keyring python-keyrings.alt
  python-pip-whl python-pyasn1 python-secretstorage python-setuptools
  python-wheel python-xdg python2.7-dev rename sgml-base shared-mime-info
  xdg-user-dirs xml-core
Suggested packages:
  binutils-doc bzip2-doc cpp-doc gcc-6-locales default-dbus-session-bus
  | dbus-session-bus debconf debconf-i18n g++-multilib g++-6-multilib gcc-6-doc
  libstdc++6-6-dbg gcc-multilib autoconf automake libtool flex bison gdb
  gcc-doc gcc-6-multilib libgcc1-dbg libgomp1-dbg libitm1-dbg libatomic1-dbg

```

10. Install the Google API Client Library.

```

1. pglynn@worker: ~ (ssh)
pglynn@worker: ~ (ssh)
root@worker: ~# pip install google-api-python-client
Collecting google-api-python-client
  Downloading https://files.pythonhosted.org/packages/bd/40/bc3b4e7c7e65f9548024
dde5c7bad60e0e078b2d2a0ee8c426a5639c2cc9/google_api_python_client-1.7.3-py2.py3-
none-any.whl (55kB)
  100% |#####| 61kB 2.4 MB/s
Collecting google-auth>=1.4.1 (from google-api-python-client)
  Downloading https://files.pythonhosted.org/packages/53/06/6e6d5bfa4d23ee40efd7
72d6b681a7afecd859a9176e564b8c329382370f/google_auth-1.5.0-py2.py3-none-any.whl
(65kB)
  100% |#####| 71kB 4.2 MB/s
Collecting google-auth-httplib2>=0.0.3 (from google-api-python-client)
  Downloading https://files.pythonhosted.org/packages/33/49/c814d6d438b823441552
198f096fcd0377fd6c88714dbed34fd3c8c4389/google_auth_httplib2-0.0.3-py2.py3-none
-any.whl
Requirement already satisfied: six<2dev,>=1.6.1 in /usr/lib/python2.7/dist-packa
ges (from google-api-python-client)
Collecting uritemplate<4dev,>=3.0.0 (from google-api-python-client)
  Downloading https://files.pythonhosted.org/packages/f6/25/66a49231b44409d7f07c
fef2506a8b070ce3c99fc47cc256bea833f24791/uritemplate-3.0.0-py2-none-any.whl
Collecting httplib2<1dev,>=0.9.2 (from google-api-python-client)
  Downloading https://files.pythonhosted.org/packages/fd/ce/aa4a385e3e9fd351737f
d2b07eda56e7a730448465aceda6b35086a0d9b/httplib2-0.11.3.tar.gz (215kB)
  100% |#####| 225kB 3.1 MB/s
Collecting pyasn1-modules>=0.2.1 (from google-auth>=1.4.1->google-api-python-cli

```

11. Install the OAuth client.

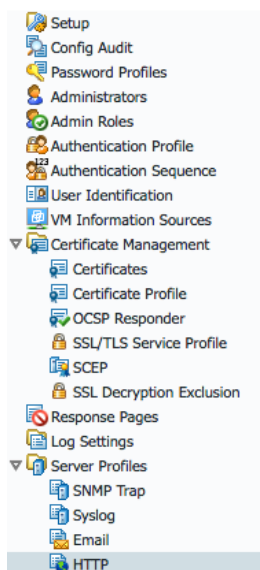
```

1. pglynn@worker: ~ (ssh)
pglynn@worker: ~ (ssh)
root@worker: ~# pip install oauth2client==1.5
Collecting oauth2client==1.5
  Downloading https://files.pythonhosted.org/packages/fd/f9/9f5a122e2a949382b85a
d56f6338552026b6d3f0837e1040b930e8a53c319/oauth2client-1.5.0.tar.gz (56kB)
  100% |#####| 61kB 2.3 MB/s
Requirement already satisfied: httplib2>=0.9.1 in /usr/local/lib/python2.7/dist-
packages (from oauth2client==1.5)
Requirement already satisfied: pyasn1-modules>=0.0.5 in /usr/local/lib/python2.7
/dist-packages (from oauth2client==1.5)
Requirement already satisfied: pyasn1>=0.1.7 in /usr/local/lib/python2.7/dist-pa
ckages (from oauth2client==1.5)
Requirement already satisfied: rsa>=3.1.4 in /usr/local/lib/python2.7/dist-packa
ges (from oauth2client==1.5)
Requirement already satisfied: six>=1.6.1 in /usr/lib/python2.7/dist-packages (f
rom oauth2client==1.5)
Building wheels for collected packages: oauth2client
  Running setup.py bdist_wheel for oauth2client ... done
  Stored in directory: /root/.cache/pip/wheels/6d/f5/c4/7d10c141f44af0c6d2a4fa22
0c483f337353c203bef022f7da
Successfully built oauth2client
Installing collected packages: oauth2client
Successfully installed oauth2client-1.5.0
root@worker: ~#

```

Configure the Firewalls

1. Login to the firewall and navigate to **Device > Server Profiles > HTTP**.



2. Add a new server profile with the following parameters:

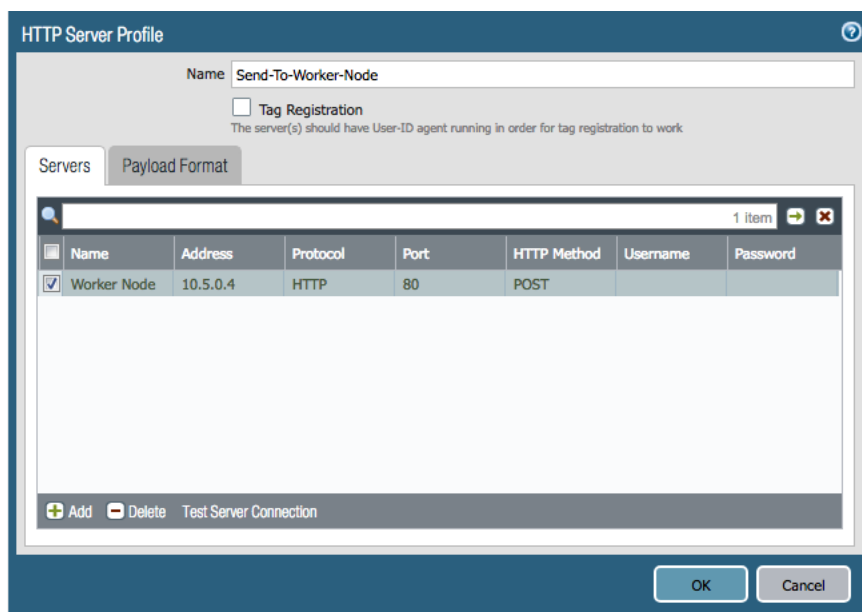
Name: free-form text (e.g. Worker Node)

Address: Internal IP of the worker node

Protocol: HTTP

Port 80

HTTP Method: POST



3. Under **Payload Format**, edit the log type for **Threat** and create a new payload format:

Palo Alto Networks GCP XFF Blocking Deployment Guide

Name: Free-form text

Content-type: application/json

Payload:

```
{"SessionID":"$sessionid","NATSRCPort":"$natsport","ReceiveTime":"$receive_time","SecurityPolicy":"protect-web-apps"}
```

(replace “protect-web-apps” with the name of the Cloud Armor security policy created earlier!)

Payload Format

Pre-defined Formats: HTTP-To-Worker

Name: HTTP-To-Worker

URI Format:

HTTP Headers

Headers	Value
content-type	application/json

+ Add - Delete

Parameters

Parameters	Value
------------	-------

+ Add - Delete

Payload:

```
{"SessionID":"$sessionid","NATSRCPort":"$natsport","ReceiveTime":"$receive_time","SecurityPolicy":"protect-web-apps"}
```

Send Test Log OK Cancel

4. Navigate to **Objects > Log Forwarding** and add a new log forwarding profile with the following parameters:

Name: Free-form text

Log Type: Threat

Filter: (severity geq medium)

Forward Method: HTTP > (your HTTP Sever Profile)

The screenshot shows the 'Log Forwarding Profile Match List' configuration window. The 'Name' field is set to 'Forward Logs'. The 'Log Type' is set to 'threat'. The 'Filter' field contains the expression '(severity geq medium)'. The 'Forward Method' section is expanded, showing four methods: SNMP, Email, Syslog, and HTTP. Each method has an 'Add' and 'Delete' button. The 'Built-in Actions' section is also expanded, showing a table with columns 'Name' and 'Type'. The table is currently empty, with 'Add' and 'Delete' buttons at the bottom. The 'OK' and 'Cancel' buttons are at the bottom right.

5. Edit the policy permitting web traffic from the untrust/internet side of the FW and add the log forwarding profile to the **Options**.

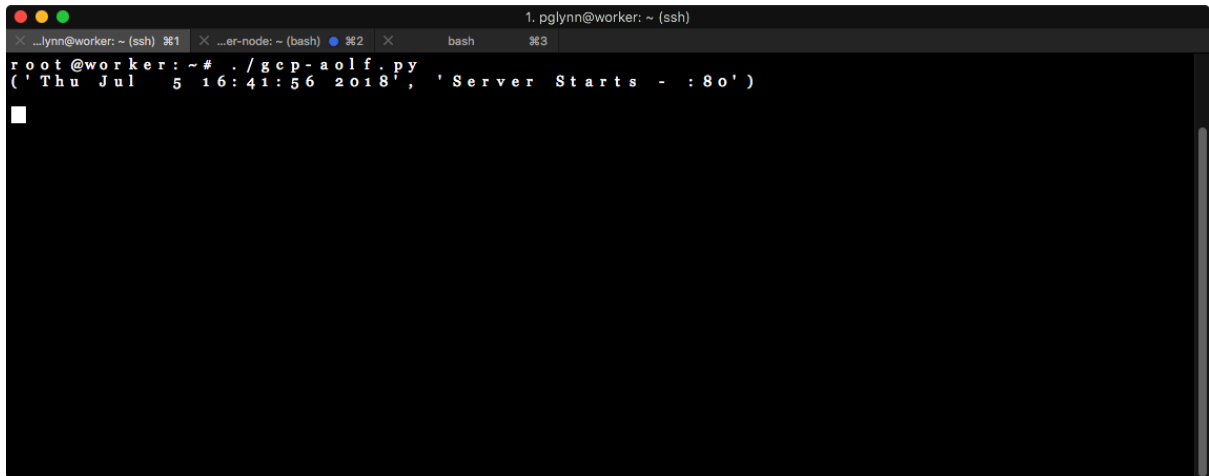
The screenshot shows the 'Options' configuration window. The 'Log Setting' section is expanded, showing 'Log at Session Start' (unchecked) and 'Log at Session End' (checked). The 'Log Forwarding' dropdown is set to 'Forward Logs'. The 'Other Settings' section is also expanded, showing 'Schedule' set to 'None', 'QoS Marking' set to 'None', and 'Disable Server Response Inspection' (unchecked). The 'OK' and 'Cancel' buttons are at the bottom right.

6. Commit the changes and replicate to the other FW.

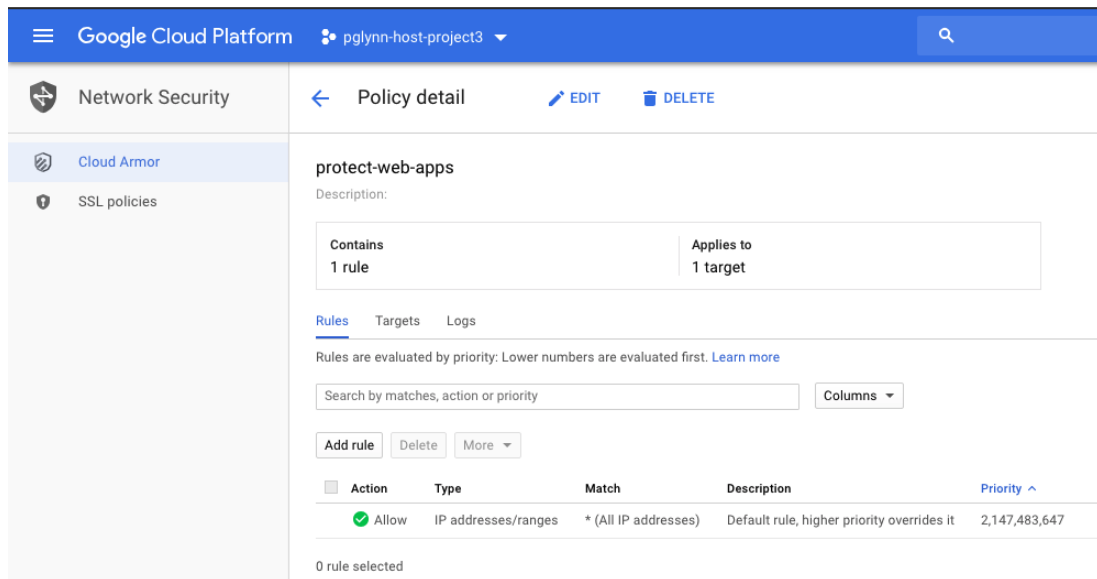
Test/Verify

1. Launch the Python script from the worker node. You will have to be root if launching it manually as the script listens on a well-known port (TCP/80).

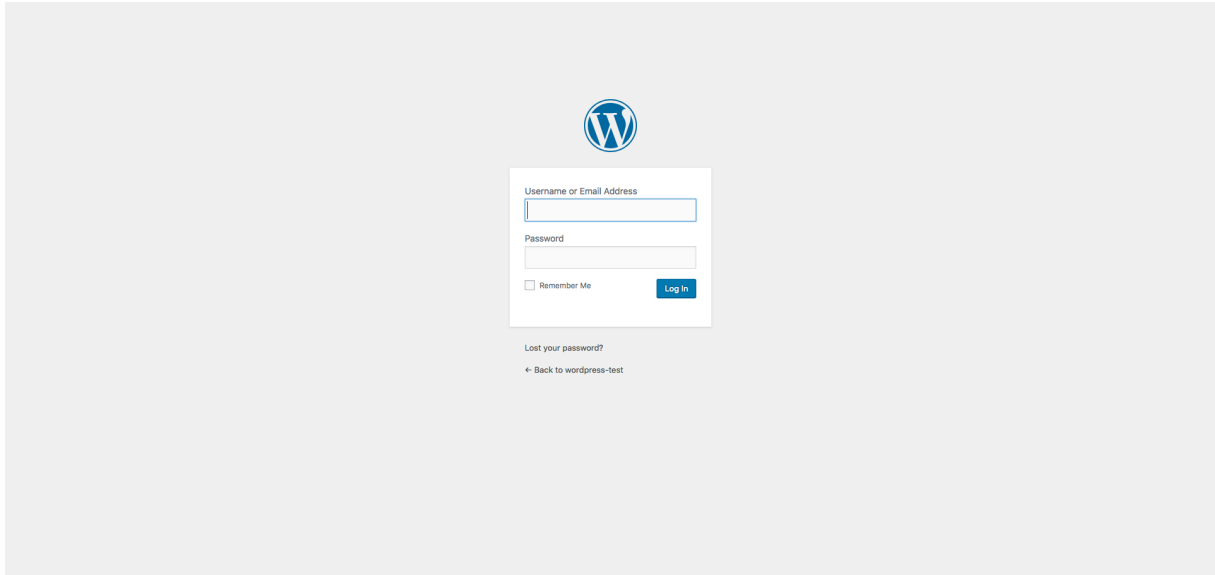
Palo Alto Networks GCP XFF Blocking Deployment Guide



2. Check the Security Policy. In a production environment, there may be multiple rules blocking/permitting access.



3. Navigate to the web page (we are using a wordpress server for this example).

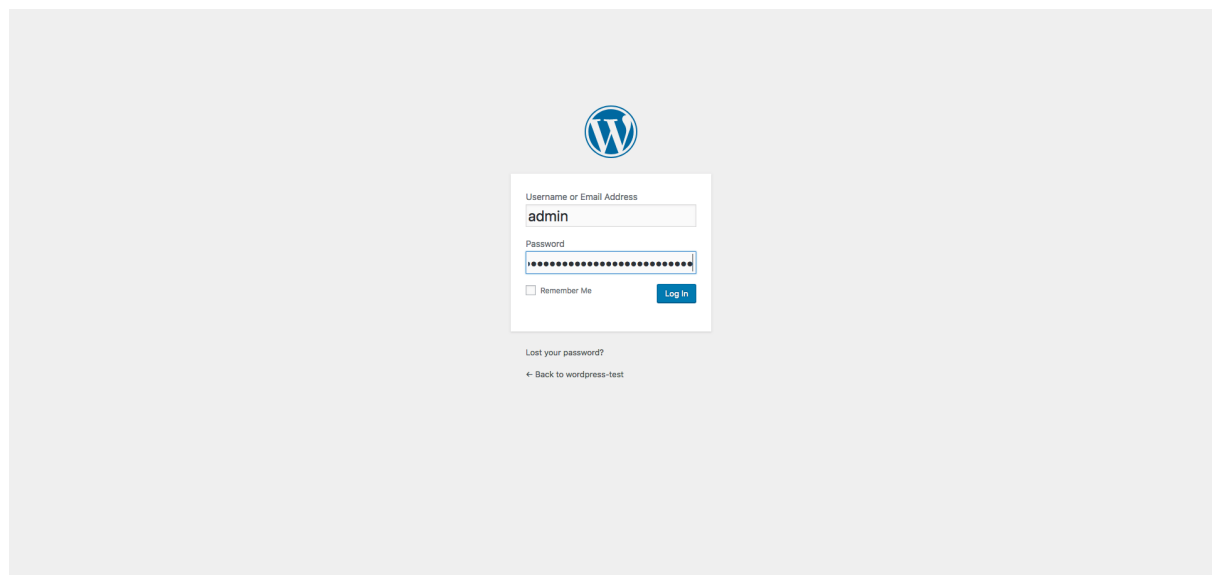


4. Input a valid username and for the password, simulate an XSS or SQL Injection attack. Examples include:

`<script>alert("HI")</script>`

`%' or '0'='0`

`1' or '1' = '1`



5. The firewall should block the attempted login.

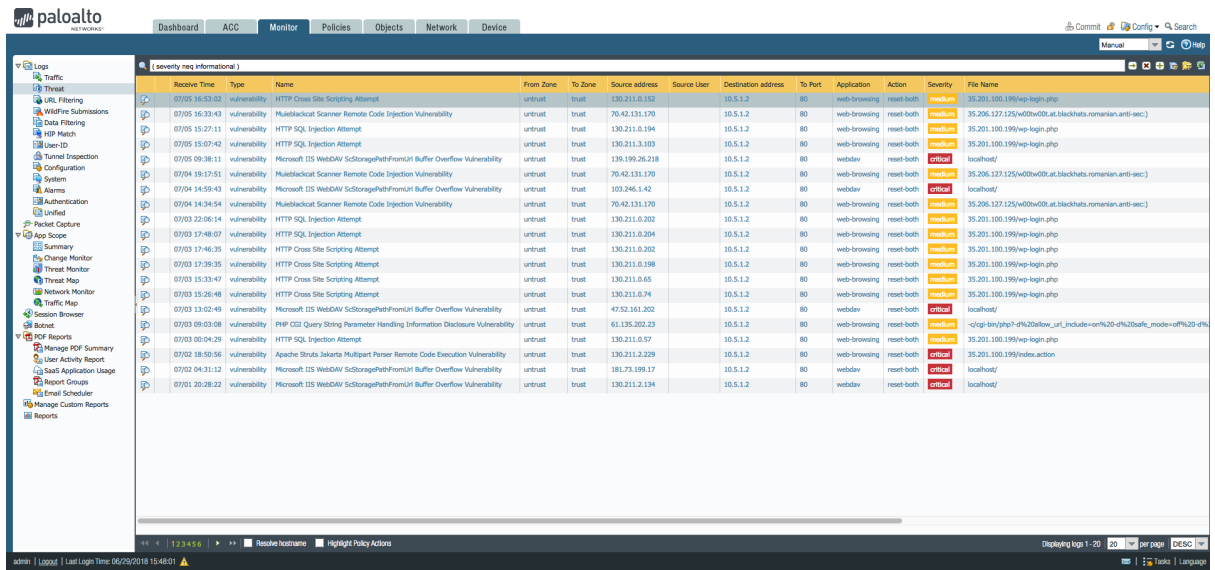
Palo Alto Networks GCP XFF Blocking Deployment Guide

Error: Server Error

The server encountered a temporary error and could not complete your request.

Please try again in 30 seconds.

6. A Threat log event will be generated.

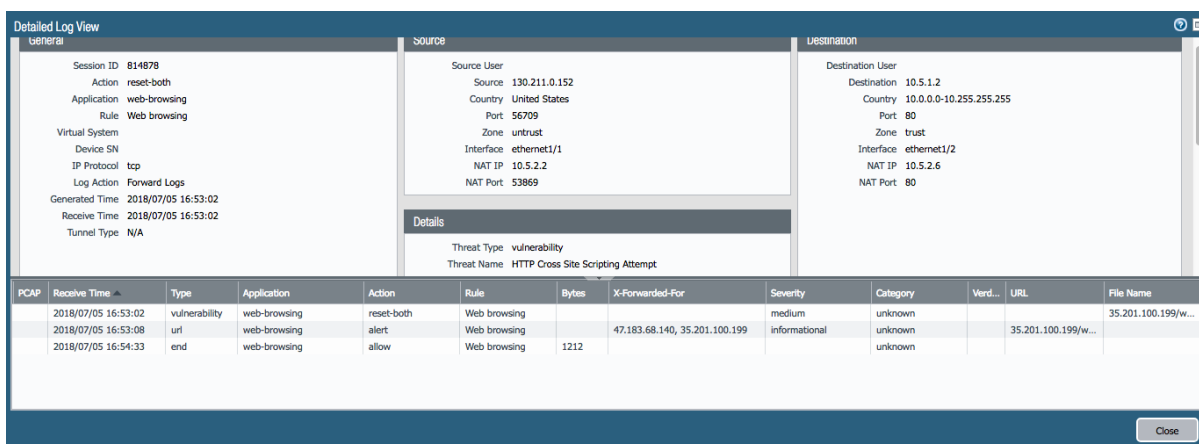


The screenshot displays the Palo Alto Networks Threat Log interface. The left sidebar contains navigation options such as Traffic, Threat, URL Filtering, Wildfire Submissions, Data Filtering, HOP Match, User ID, Tunnel Inspection, Configuration, System, Alarms, Authentication, Unified, App Scope, Packet Capture, App Scope, Summary, Change Monitor, Threat Monitor, Threat Map, Network Monitor, Session Browser, Snort, PDF Reports, Manage PDF Summary, User Activity Report, SaaS Application Usage, Report Groups, Email Scheduler, Manage Custom Reports, and Reports. The main panel shows a table of threat log entries with columns: Receive Time, Type, Name, From Zone, To Zone, Source address, Source User, Destination address, To Port, Application, Action, Severity, and File Name. The table lists various security events, including HTTP Cross Site Scripting Attempts, Microsoft IIS WebDAV SCSStoragePathFromUrl Buffer Overflow Vulnerabilities, and PHP CGI Query String Parameter Handling Information Disclosure Vulnerabilities. The severity levels range from medium to critical.

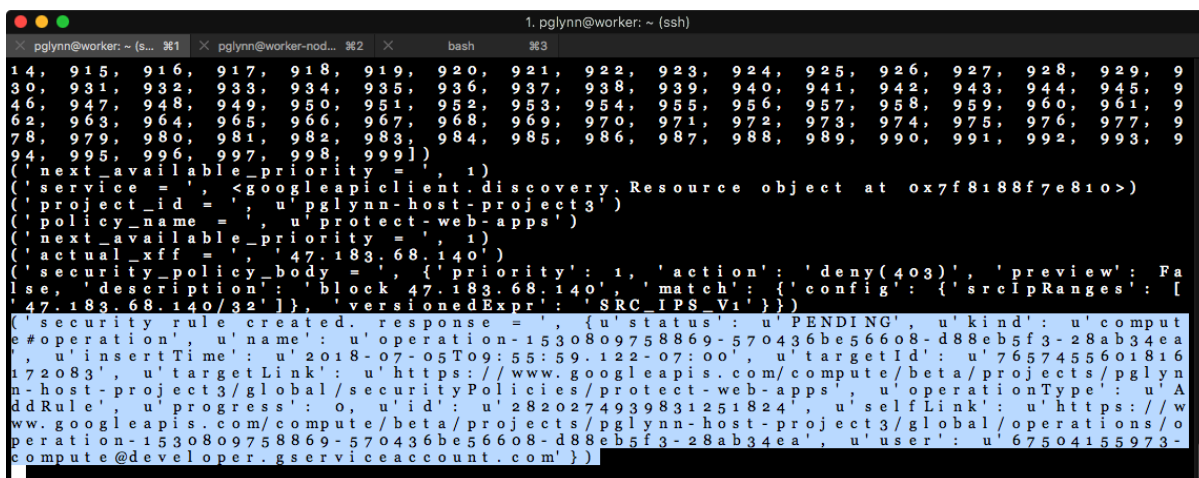
Receive Time	Type	Name	From Zone	To Zone	Source address	Source User	Destination address	To Port	Application	Action	Severity	File Name
07/05 16:53:02	vulnerability	HTTP Cross Site Scripting Attempt	untrust	trust	130.211.0.152		10.5.1.2	80	web-browsing	reset-both	medium	35.201.100.199/wp-login.php
07/05 16:53:43	vulnerability	Mueblackcat Scanner Remote Code Injection Vulnerability	untrust	trust	70.42.131.170		10.5.1.2	80	web-browsing	reset-both	medium	35.206.127.125/w00w00t.at.blackhats.romanian.anti-sec)
07/05 15:27:11	vulnerability	HTTP SQL Injection Attempt	untrust	trust	130.211.0.194		10.5.1.2	80	web-browsing	reset-both	medium	35.201.100.199/wp-login.php
07/05 15:07:42	vulnerability	HTTP SQL Injection Attempt	untrust	trust	130.211.3.103		10.5.1.2	80	web-browsing	reset-both	medium	35.201.100.199/wp-login.php
07/05 09:38:11	vulnerability	Microsoft IIS WebDAV SCSStoragePathFromUrl Buffer Overflow Vulnerability	untrust	trust	139.199.26.218		10.5.1.2	80	webdav	reset-both	critical	localhost/
07/04 19:17:51	vulnerability	Mueblackcat Scanner Remote Code Injection Vulnerability	untrust	trust	70.42.131.170		10.5.1.2	80	web-browsing	reset-both	medium	35.206.127.125/w00w00t.at.blackhats.romanian.anti-sec)
07/04 14:59:43	vulnerability	Microsoft IIS WebDAV SCSStoragePathFromUrl Buffer Overflow Vulnerability	untrust	trust	103.246.1.42		10.5.1.2	80	webdav	reset-both	critical	localhost/
07/04 14:34:54	vulnerability	Mueblackcat Scanner Remote Code Injection Vulnerability	untrust	trust	70.42.131.170		10.5.1.2	80	web-browsing	reset-both	medium	35.206.127.125/w00w00t.at.blackhats.romanian.anti-sec)
07/03 22:06:14	vulnerability	HTTP SQL Injection Attempt	untrust	trust	130.211.0.202		10.5.1.2	80	web-browsing	reset-both	medium	35.201.100.199/wp-login.php
07/03 17:48:07	vulnerability	HTTP SQL Injection Attempt	untrust	trust	130.211.0.204		10.5.1.2	80	web-browsing	reset-both	medium	35.201.100.199/wp-login.php
07/03 17:46:35	vulnerability	HTTP Cross Site Scripting Attempt	untrust	trust	130.211.0.202		10.5.1.2	80	web-browsing	reset-both	medium	35.201.100.199/wp-login.php
07/03 17:39:35	vulnerability	HTTP Cross Site Scripting Attempt	untrust	trust	130.211.0.198		10.5.1.2	80	web-browsing	reset-both	medium	35.201.100.199/wp-login.php
07/03 15:33:47	vulnerability	HTTP Cross Site Scripting Attempt	untrust	trust	130.211.0.65		10.5.1.2	80	web-browsing	reset-both	medium	35.201.100.199/wp-login.php
07/03 15:26:48	vulnerability	HTTP Cross Site Scripting Attempt	untrust	trust	130.211.0.74		10.5.1.2	80	web-browsing	reset-both	medium	35.201.100.199/wp-login.php
07/03 13:02:48	vulnerability	Microsoft IIS WebDAV SCSStoragePathFromUrl Buffer Overflow Vulnerability	untrust	trust	47.52.161.202		10.5.1.2	80	webdav	reset-both	critical	localhost/
07/03 09:03:08	vulnerability	PHP CGI Query String Parameter Handling Information Disclosure Vulnerability	untrust	trust	61.135.202.23		10.5.1.2	80	web-browsing	reset-both	medium	-c/cgi-bin/php?d%20below_url_include=on%20-d%20safe_mode=off%20-d%
07/03 00:04:29	vulnerability	HTTP CGI Query String Parameter Handling Information Disclosure Vulnerability	untrust	trust	130.211.0.57		10.5.1.2	80	web-browsing	reset-both	medium	35.201.100.199/wp-login.php
07/02 18:50:56	vulnerability	Apache Struts Jakarta Multipart Parser Remote Code Execution Vulnerability	untrust	trust	130.211.2.229		10.5.1.2	80	web-browsing	reset-both	critical	35.201.100.199/index.action
07/02 04:31:12	vulnerability	Microsoft IIS WebDAV SCSStoragePathFromUrl Buffer Overflow Vulnerability	untrust	trust	181.73.199.17		10.5.1.2	80	webdav	reset-both	critical	localhost/
07/01 20:28:22	vulnerability	Microsoft IIS WebDAV SCSStoragePathFromUrl Buffer Overflow Vulnerability	untrust	trust	130.211.2.134		10.5.1.2	80	webdav	reset-both	critical	localhost/

7. Details on the log entry will show the Traffic, URL, and Threat logs as well as the log forwarding action.

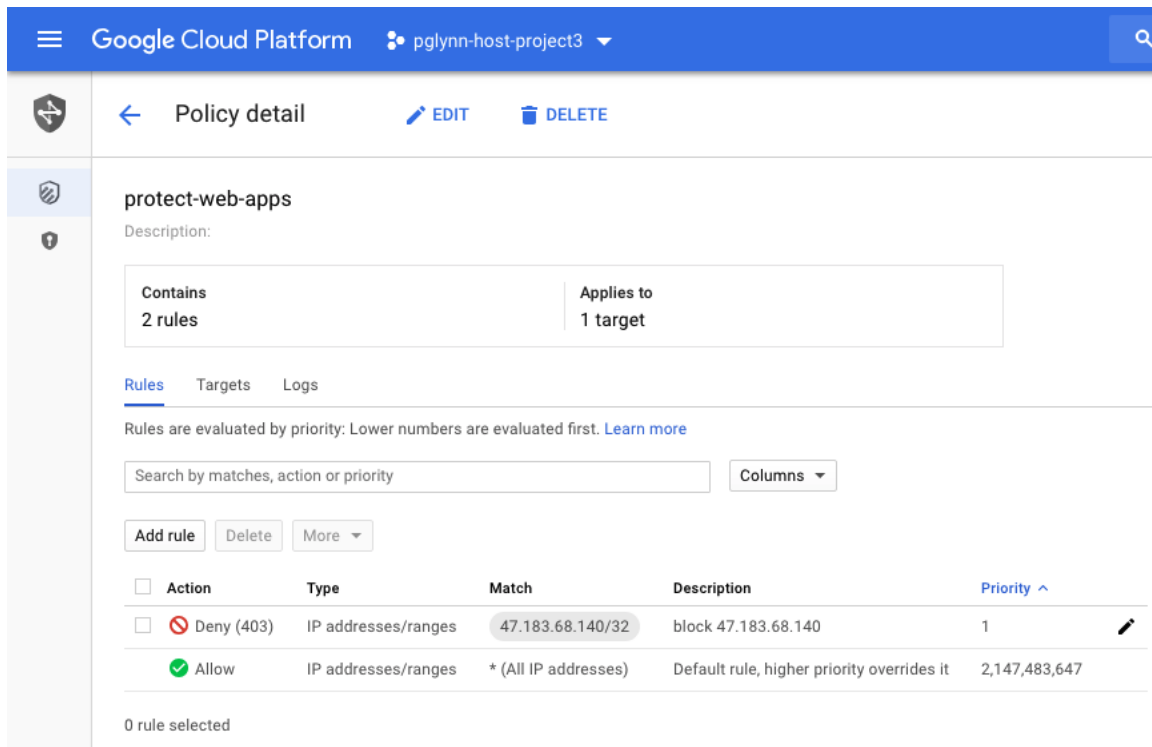
Palo Alto Networks GCP XFF Blocking Deployment Guide



8. If debugging is enabled, you will see a large amount of output culminating in the acceptance of the request to create a new rule.



9. Check the security policy after a few moments (a browser refresh may be required).



10. Verify by re-attempting the XSS/SQL Injection attempt from the browser. You should see a **403 Forbidden** error.

11. When done, interrupt the Python script with <CTRL>-<C>.

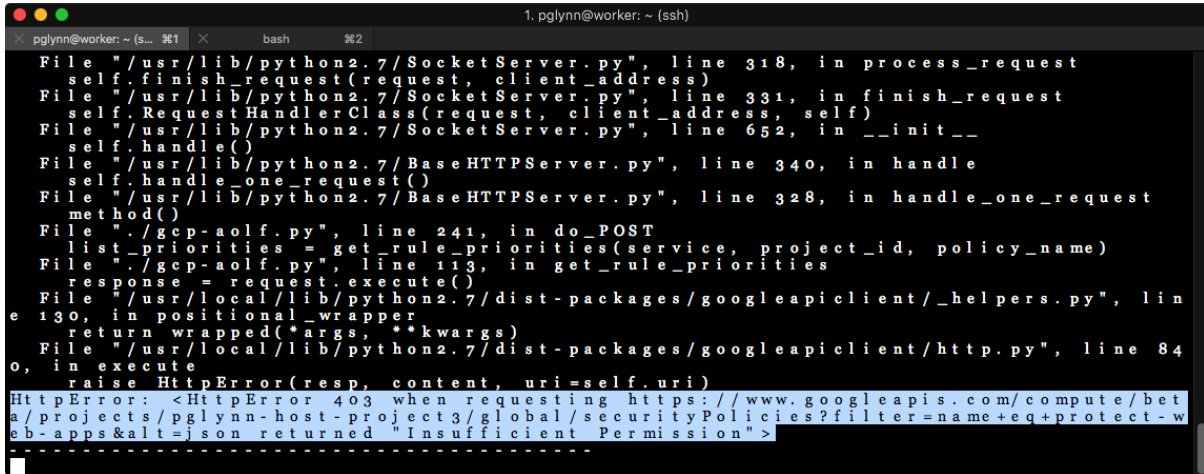
```
1. pglynn@worker: ~ (ssh)
pglynn@worker: ~ (ssh)
('next_available_priority' = 1)
('service' = <googleapiclient.discovery.Resource object at 0x7f8188f7e810>)
('project_id' = 'pglynn-host-project3')
('policy_name' = 'protect-web-apps')
('next_available_priority' = 1)
('actual_xff' = '47.183.68.140')
('security_policy_body' = {'priority': 1, 'action': 'deny(403)', 'preview': False, 'description': 'block 47.183.68.140', 'match': {'config': {'srcIpRanges': ['47.183.68.140/32']}, 'versionedExpr': 'SRC_IPS_V1'}})
('security_rule_created_response' = {'status': 'PENDING', 'kind': 'compute#operation', 'name': 'operation-1530809758869-570436be56608-d88eb5f3-28ab34ea', 'insertTime': '2018-07-05T09:55:59.122-07:00', 'targetId': '7657455601816172083', 'targetLink': 'https://www.googleapis.com/compute/beta/projects/pglynn-host-project3/global/securityPolicies/protect-web-apps', 'operationType': 'AddRule', 'progress': 0, 'id': '2820274939831251824', 'selfLink': 'https://www.googleapis.com/compute/beta/projects/pglynn-host-project3/global/operations/operation-1530809758869-570436be56608-d88eb5f3-28ab34ea', 'user': '67504155973-compute@developer.gserviceaccount.com'})

^C('Thu Jul 5 17:31:20 2018', 'Server Stops - :80')
root@worker: ~#
```

Troubleshooting

- For the script to be able to execute, it needs to load two python libraries: google-api-python3-client and oauth2client==1.5. If those two libraries are not installed prior to running the script, it will exit with an error.

- If the service account does not have the correct permissions or the authentication key has not been loaded, the script will run but fail when attempting to query the GCP environment.



```
1. pglynn@worker: ~ (ssh)
pglynn@worker: ~ (ssh)
File "/usr/lib/python2.7/SocketServer.py", line 318, in process_request
    self.finish_request(request, client_address)
File "/usr/lib/python2.7/SocketServer.py", line 331, in finish_request
    self.RequestHandlerClass(request, client_address, self)
File "/usr/lib/python2.7/SocketServer.py", line 652, in __init__
    self.handle()
File "/usr/lib/python2.7/BaseHTTPServer.py", line 340, in handle
    self.handle_one_request()
File "/usr/lib/python2.7/BaseHTTPServer.py", line 328, in handle_one_request
    method()
File "./gcp-aolf.py", line 241, in do_POST
    list_priorities = get_rule_priorities(service, project_id, policy_name)
File "./gcp-aolf.py", line 113, in get_rule_priorities
    response = request.execute()
File "/usr/local/lib/python2.7/dist-packages/googleapiclient/_helpers.py", line
130, in positional_wrapper
    return wrapped(*args, **kwargs)
File "/usr/local/lib/python2.7/dist-packages/googleapiclient/http.py", line 84
o, in execute
    raise HttpError(resp, content, uri=self.uri)
HttpError: <HttpError 403 when requesting https://www.googleapis.com/compute/bet
a/projects/pglynn-host-project3/global/securityPolicies?filter=name+eq+protect-w
eb-apps&alt=json returned "Insufficient Permission">
```

For more details on the API calls, including required IAM permissions, see:

<https://cloud.google.com/compute/docs/reference/rest/beta/securityPolicies/list>

<https://cloud.google.com/compute/docs/reference/rest/beta/securityPolicies/addRule>