AppSentinels.ai
Application Security Re-invented

AppSentinels API Security Platform

NGINX Deployment

Table of Contents

Revision	Date Modified	Author	Comments
1.0	02-Jan-24		Initial Draft
1.1	10-Jan-24	Sachin	Addressed minor review comments
1.2	20-Sept-24	Sachin	Updating with v2 module configs and links
1.3	28-Oct-24	Sachin	Secure logging
1.4	20-Feb-25	Sachin	Logging port change for newer modules

1. Introduction

AppSentinels supports nginx via linkable modules/plugins which can be integrated into your existing deployments.

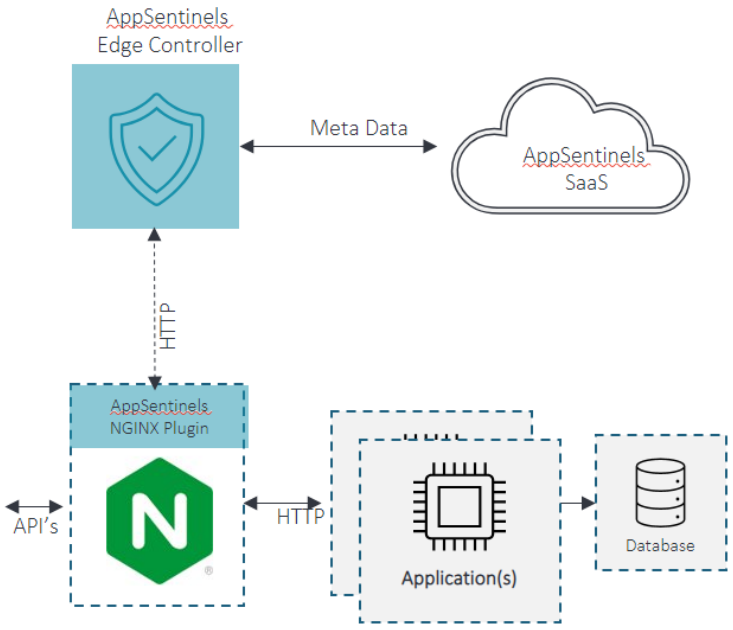

Figure 1 Nginx deployment with AppSentinels

1. AppSentinels NGINX modules gets HTTP traffic and forward the logs to AppSentinels Edge Controller for security processing
2. Modules support two modes configurable via a knob Out-of-Band(OOB)/Transparent & Service-chaining/Enforcement. In both the modes, AppSentinels process a copy of the packet
3. In OOB/transparent mode, plugin forwards the packet to the Application and Edge controller simultaneously. In service-chaining mode, the plugin forwards the packet to Edge Controller and waits for it's output before forwarding the packet to Application. This allows the plugin to enforce inline action based on response received from Edge Controller
4. AppSentinels Service-chaining mode has optional max-latency configuration. In case Edge controller response is delayed and latency crosses configured threshold, plugin gets into fail-open mode and forwards the packet to Application thereby ensuring availability and responsiveness in case of a slowness or an outage.

2. Deployment Options

AppSentinels provides various deployment options for nginx,

2.1 Baremetal nginx or nginx running on host

AppSentinels will provide a couple of loadable modules, i.e

```
nginx_ext_auth_module.so
nginx_ext_access_log_module.so
```

There are few steps to follow that are common across other deployment options as well,

2.1.1 Load modules

a. These modules will need to be copied or mounted from where nginx can pick and link with them. For eg: one can typically copy the modules onto /etc/nginx/modules/
b. Insert the below directives into your nginx.conf at the global level. Some distributions can provide for other ways of inserting this directive. Please follow the same.

```
load_module /etc/nginx/modules/nginx_ext_auth_module.so;
load_module /etc/nginx/modules/nginx_ext_access_log_module.so;
thread_pool ext_access_log_thread_pool threads=1 max_queue=10000;
thread_pool ext_monitoring_thread_pool threads=1 max_queue=10;
```

c. Please restart the nginx to check if the modules have been loaded properly

2.1.2 Integrate with application server and location blocks

2.1.2.1 OOO/Transparent mode

a. Choose the application you want to onboard onto AppSentinels. Into its server block, insert the AppSentinels directive (ext_auth_log_server and ext_stats_server) to point to the edge controller

```
http {

        server {
         listen       9000;
         server_name  front-service;

         # AppSentinels config block start
         ext_auth_log_server http://onprem-controller:9004;
         ext_stats_server http://onprem-controller:9004;
         ext_instance "<name for current instance>";
         # AppSentinels config block end

         location / {
             proxy_pass http://localhost:3000;
         }
    }
}
```

b. Ensure edge controller is reachable at the configured URL.
c. In case of secure logging over HTTPS, please change the scheme of the URL to https. You will need to ensure controller is configured to accept HTTPS
d. Similarly populate the stats server to be the edge controller itself
e. Populate the instance name for visibility. For eg: uat-region1

f. Please restart the nginx

Reference configuration can be found at https://sample-config.appsentinels.ai/appsentinels-deployment/nginx/sample_nginx_v2.conf

2.1.2.2 Service Chaining/Enforcement mode

a. Enforcement will require incoming API request to be vetted against Edge controller. This will require server level definition of /auth subrequest block and a location specific directive as below

```
server {
    listen       9000;
    server_name  front-service;

    # AppSentinels config block start
    ext_auth_fail_allow on;
    location /auth {
        internal;
        proxy_pass http://onprem-controller:9004;
    }
    ext_stats_server http://onprem-controller:9004;
    ext_instance "<name for current instance>";
    # AppSentinels config block end

    location / {
        # AppSentinels config block start
        ext_auth_request /auth;
        # AppSentinels config block end
        proxy_pass http://localhost:3000;
    }
}
```

b. Ensure edge controller is reachable at the configured auth URL
c. Similarly populate the stats server to be the edge controller itself
d. Populate the instance name for visibility. For eg: uat-region1
e. Please restart the nginx

Reference configuration can be found at https://sample-config.AppSentinels.ai/AppSentinels-deployment/nginx/sample_nginx_enforcement.conf

2.2 Container based nginx deployments

The above highlighted method of integration can be used in deployments that use nginx as containers.

2.2.1 Load modules

a. AppSentinels modules will need to be mounted into the nginx container instance from where the nginx can pick and link with them. One way to do this would be to perform a

container mount of the modules. Another way could be to package the modules into a new container image.

The below example shows one way of using volume mount in a docker-compose spec to mount the modules. Here the modules are copied onto,
/usr/local/openresty/nginx/modules/

```
openresty:
  image: openresty/openresty:alpine
  container_name: openresty
  ports:
    - 7001:7001
    - 9000:9000
  extra_hosts:
    - "onprem-controller:172.17.0.1"
  volumes:
    - ./nginx_ext_auth_module.so:/usr/local/openresty/nginx/modules/nginx_ext_auth_module.so
    - ./nginx_ext_access_log_module.so:/usr/local/openresty/nginx/modules/nginx_ext_access_log_module.so
    - ./openresty/nginx.conf:/usr/local/openresty/nginx/conf/nginx.conf
    - ./openresty/http.conf:/etc/nginx/conf.d/http.conf
```

Ref spec:
docker-compose.yaml:
https://sample-config.appsentinels.ai/appsentinels-deployment/nginx/openresty-conf/docker-compose.yaml

b. Insert the below directives into your nginx.conf. Some distributions can provide other ways of inserting this directive into hooks provided by the configuration. In the absence of it, the existing nginx.conf will have to be modified and remounted as in section

```
    load_module /etc/nginx/modules/nginx_ext_auth_module.so;
    load_module /etc/nginx/modules/nginx_ext_access_log_module.so;
    thread_pool ext_access_log_thread_pool threads=1 max_queue=10000;
    thread_pool ext_monitoring_thread_pool threads=1 max_queue=10;
```

c. Please restart the container to ensure modules are loading fine

2.2.2 Integrate with application server and location blocks

a. The process to integrate AppSentinels modules into server and location blocks remain the same as described for baremetal nginx. Please refer to section and pickup the right configurations based on the mode of deployment (transparent or service chaining mode)
f. Ensure edge controller is reachable at the configured URL
b. Please restart the nginx container

3. Verify Deployment

AppSentinels Edge Controllers deployed in your environment will be listed on the System Health page on AppSentinels Dashboard.

Generate application traffic in the monitored application and check the API catalogue on AppSentinels Dashboard for the discovered APIs.

4. Deployment/Debugging

1. Deploy edge controller
2. Once the container is up, you should be able to see the same under the system health page of your organization dashboard
3. Nginx will ship logs to edge controller's (default port 9006) usually running on a different host. To check if there is any connectivity issue between nginx and edge controller, run the below tcpdump on the edge controller VM.

```
tcpdump -i <ingress device eg: eth0, ens3> -A  'port 9006 and (((ip[2:2] - ((ip[0]&0xf)<<2)) - ((tcp[12]&0xf0)>>2)) != 0)' | egrep --line-buffered "^........(GET |HTTP\/|POST |HEAD )|^[A-Za-z0-9-]+: " | sed -r 's/^........(GET |HTTP\/|POST |HEAD )/\n\1/g'
```

The above command should output API requests called **POST /auth** which signals, that connectivity is good between nginx and edge controller's VM.
The above command assumes that nginx is sending logs to port 9006, however, if you have setup where nginx ship logs to a different port, please change the port according in tcpdump

4. If edge controller is connected to AppSentinels cloud, you should be able to see your APIs on the dashboard.

5. Upgrades

If nginx or openresty is to be upgraded, AppSentinels modules will need to be recompiled for the latest nginx or openresty. Recompilation process will require sharing the complete output of the below commands relevant to the latest system,
> cat /etc/os-release
> nginx –V (capital v)

Once the newer modules are available, the upgrade process can proceed as required.

