Juniper Networks | Engineering Simplicity

Juniper Alert Format Relay Guide

Published
2024-05-23

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Juniper Alert Format Relay Guide

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at https://support.juniper.net/support/eula/. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

1 **Introduction**

Overview of Juniper Alert Format Relay | 2

2 **Requirements**

Virtual Machine Specifications | 6

Security Implementation and Recommendations | 8

Ports to Open | 10

Create an API Token for Juniper Alert Format Relay | 11

3 **Installation and Configuration**

Information Needed for Installation | 15

Install Juniper Alert Format Relay | 16

- Pre-Installation Checklist | **16**
- Prepare the VM Hosts | **17**
- Set Up the Local Registry | **18**
- Update Configurations | **19**
- Deploy the Installation Package | **23**

Enable Webhooks and Alerts by Using the API | 26

Enable Webhooks and Alerts by Using the Juniper Mist Portal | 29

Select Alerts and Events to Monitor | 31

Load the MIB File | 32

1
CHAPTER

Introduction

Overview of Juniper Alert Format Relay | 2

Overview of Juniper Alert Format Relay

IN THIS SECTION

- What Are Webhooks? | 2
- Information Flow | 2
- Benefits of Webhooks | 3
- Benefits of Juniper Alert Format Relay | 4
- Process Overview | 4

The Juniper Alert Format Relay provides you with Juniper Mist™ monitoring information by using selected data through webhooks. The information is parsed into pre-defined formats such as SNMP Traps and SYSLOG messages.

What Are Webhooks?

Webhooks (also known as user-defined HTTP callbacks, HTTP posts, or HTTP notifications) provide real-time notifications to a webserver. Even if your NMS cannot handle direct webhooks integrations, the Juniper Alert Format Relay ensures that you receive information about Juniper Mist events as they occur.

You can enable webhooks for Juniper Mist alarms, device events, and device up/downs.

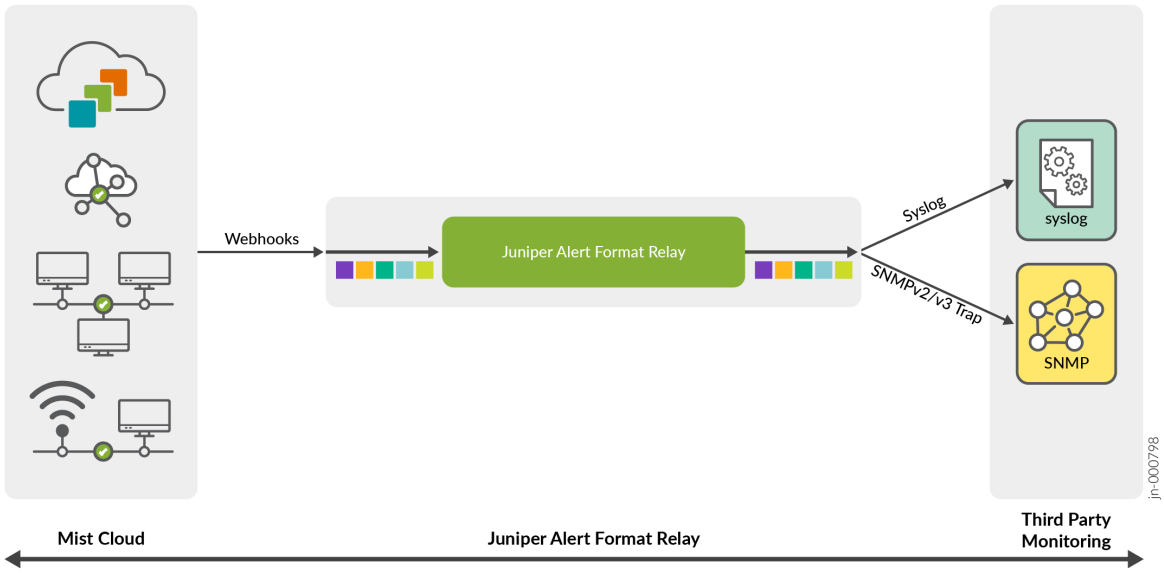
Information Flow

Here's how it works:

1. When an alarm or event occurs, Juniper Mist sends real-time notifications by using webhooks.
2. Juniper Alert Format Relay receives the notifications.
3. Juniper Alert Format Relay queries Juniper Mist using APIs to translate certain values into more readable text. For example, Juniper Alert Format Relay translates obscure IDs into site names and device names.

4. Juniper Alert Format Relay uses SYSLOG and/or SNMP v2/v3 traps to forward the messages to the network monitoring tool.
5. In the case of SNMP, the NMS uses the MIB file (MIST-ALERT-MIB) to process the messages and report them as alarms.

This diagram illustrates the information flow from Juniper Mist to Juniper Alert Format Relay to your network monitoring tool.



NOTE: Juniper Alert Format Relay does not provide support for SNMP agent integration with Juniper Mist. If the monitoring tool requires SNMP discovery, a lightweight SNMP agent can be enabled to craft a response to the SNMP queries from monitoring tool.

Benefits of Webhooks

- Compared with APIs, which pull information from a source, webhooks push notifications as alarms and events occur.
- Compared with device-generated SNMP traps, webhooks notifications can include Juniper Mist's Marvis events as well as device events.

Benefits of Juniper Alert Format Relay

- If your network monitoring tool can't handle webhooks integrations, you can use Juniper Alert Format Relay to bridge that gap.
- Juniper Alert Format Relay provides a single integration and configuration point, reducing the configuration effort.

Process Overview

Table 1: Installation and Configuration Process

Step	Task	More Information
1	Set up the virtual machines (VMs) for the Juniper Alert Format Relay framework.	"Virtual Machine Specifications" on page 6
2	Configure your firewall rules to allow webhooks traffic.	• "Security Implementation and Recommendations" on page 8 • "Ports to Open" on page 10
3	Install Juniper Alert Format Relay.	"Install Juniper Alert Format Relay" on page 16
4	Enable the webhooks by using the API or the Juniper Mist portal.	• "Enable Webhooks and Alerts by Using the API" on page 26 • "Enable Webhooks and Alerts by Using the Juniper Mist Portal" on page 29
5	Load the Juniper Mist MIB file (MIST-ALERT-MIB) onto your network monitoring tool so that you can monitor the alerts from Juniper Alert Format Relay.	"Load the MIB File" on page 32

2

CHAPTER

Requirements

Virtual Machine Specifications | 6

Security Implementation and Recommendations | 8

Ports to Open | 10

Create an API Token for Juniper Alert Format Relay | 11

Virtual Machine Specifications

SUMMARY

To deploy Juniper Alert Format Relay, set up a framework that meets these requirements.

IN THIS SECTION

- VM Nodes | 6

VM Nodes

The Juniper Alert Format Relay platform consists of several infrastructure services and microservices, which are deployed across the three nodes by using a deployer VM. The deployer VM is a dedicated VM for hosting images and performing installation. All infrastructure service and microservices are deployed and configured as HA on three nodes and the system can handle a single VM failure.

Table 2: Hardware and Software Requirements

Requirement	Details	Comments
Number of Virtual Machines (VMs)	4	• 3 relay VMs—These VMs run the Juniper Alert Format Relay software. • 1 Deployer VM—This VM is used for installation, upgrade, and the registry for images.
Number of IP Addresses	5	• 4 IPs (1 for each VM) • 1 virtual IP in the same subnet
Number of VM Cores	• 8 for each relay VM • 6 for the deployer VM	

Table 2: Hardware and Software Requirements *(Continued)*

Requirement	Details	Comments
VM Memory	• 16 GB for each relay VM • 8 GB for the deployer VM	
VM Storage	150 GB	Root Partition – / 20G Data Partitions: • /home—10G • /opt—20G • /mnt—60G • /var—40G SSD Recommended
Network Interface	1	Configured with VM private IP address
VM OS	Ubuntu 22.04.2 LTS (Jammy Jellyfish) or RHEL8	• Enable APT/YUM, and allow access to these repositories/mirror. • Install Docker on the Deployer VM. • Enable root user access to the VMs. • Disable any OS firewall/ACL.

Security Implementation and Recommendations

IN THIS SECTION

- General Security Guidelines | 8
- SSL Certificate | 8
- Allow Inbound Traffic from Webhooks Public Source IP Addresses | 9
- VM Internet Access | 9
- Firewall and NAT Configuration | 9
- Security Hardening | 10

General Security Guidelines

- Use HTTPS with a public CA certificate. You'll need a publicly resolvable hostname and public CA certificate.
- Configure the firewall to allow only specific ports for inbound traffic.
- Configure the firewall to allow inbound traffic from only Juniper Mist™ public source IPs.
- Configure the firewall to allow only specific ports for outbound traffic.
- Update the VMs with all required security patches.
- Restrict the Juniper Alert Format Relay VM access to authorized users only.
- Allow VM and SSH access only to the internal/management network.

SSL Certificate

You need to specify a publicly resolvable FQDN for the Juniper Alert Format Relay public IP. We recommend SSL certificate validation using a public Certificate Authority (CA). If a certificate isn't available, you can disable the certificate verification requirement during the installation process.

By default, Juniper Alert Format Relay can generate and install self-signed certificates. However, it's recommended to use publicly resolvable FQDN with a public CA certificate.

Allow Inbound Traffic from Webhooks Public Source IP Addresses

Add firewall rules to allow webhooks traffic from Juniper Mist.

Juniper Mist generates the webhooks with static public IP addresses, which will not change. For the complete list of source IPs for webhooks, see in the Juniper Mist Management Guide.

As a second layer of check, Juniper Alert Format Relay allows traffic only from Juniper Mist source IPs.

VM Internet Access

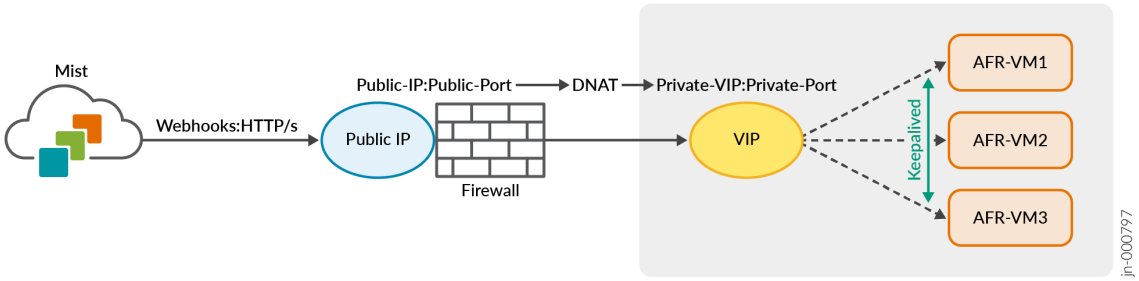
You can disable Internet access to the Juniper Alert Format Relay VMs and provide conditional access only when necessary. Juniper Alert Format Relay runs API queries to fetch inventory details, so the VMs should be able to access the Juniper Mist URL. For a complete list, see *Mist Cloud IP Addresses and Ports* in the Juniper Mist Management Guide.

On the VMs, enable APT/YUM and allow access to these repositories/mirror.

Firewall and NAT Configuration

You can configure Juniper Mist to forward the webhooks to a public IP/FDQN. This public IP can be on the firewall. Configure NAT to forward the webhook messages to the private VIP for Juniper Alert Format Relay. Juniper Alert Format Relay will respond with the HTTP status code for webhook HTTP post messages. You'll need to configure the required static/source NAT settings to allow this communication.

Here's an illustration of how the traffic from Juniper Mist moves through a firewall to the virtual IP and VMs.



Security Hardening

Here are a few ways that Juniper Alert Format Relay hardens your Juniper Mist network to make it more resistant to attacks.

- Only specific web endpoints are exposed to the Internet.
- Web endpoints are configured to receive traffic from specific Juniper Mist public IPs.
- Sensitive information is stored locally in encrypted format.

Ports to Open

IN THIS SECTION

- Firewall Recommendations | **10**

Firewall Recommendations

When configuring a firewall for your VMs, use the guidelines in the following tables.

Table 3: Ports

Application	L4	Direction	Destination Port	Communication Type
HTTPS	TCP	Outbound	443	External (only to Juniper Mist) and Internal
HTTPS	TCP	Inbound	Any TCP port One public port	External and Internal
SNMP	UDP	Outbound	162 or custom	Internal

Table 3: Ports *(Continued)*

Application	L4	Direction	Destination Port	Communication Type
SNMP	UDP	Inbound	30001, 30002, 30003	Internal
SYSLOG	UDP	Outbound	514 or custom	Internal
DNS	UDP	Outbound	53	External and Internal
NTP	UDP	Outbound	123	External and Internal
SSH	TCP	Inbound	22	Internal

Table 4: Sample Firewall Rules

Source	Destination	Port	Comments
<*VM1-Private-IP*> <*VM2-Private-IP*> <*VM3-Private-IP*>	Any	UDP-53, UDP-123, TCP-80, TCP-443 (Only to Juniper Mist)	Outbound rule
Juniper-Mist-Webhook-public-IPs	<*AFR public-IP*>	TCP-<*public port*>	Inbound rule

Table 5: Sample NAT Rule

Original Source	Original Destination	Original Service	Translated Destination IP	Translated Destination Port	Translated Service
<*Juniper-Mist-Webhook-public IPs*>	<*AFR-public-IP*>	TCP-<*public_port*>	<*VIP-Private-IP*>	<*private_port*> Default: 443	Original (TCP)

Create an API Token for Juniper Alert Format Relay

Before you install Juniper Alert Format Relay, you need to generate an API token in the Juniper Mist™ portal.

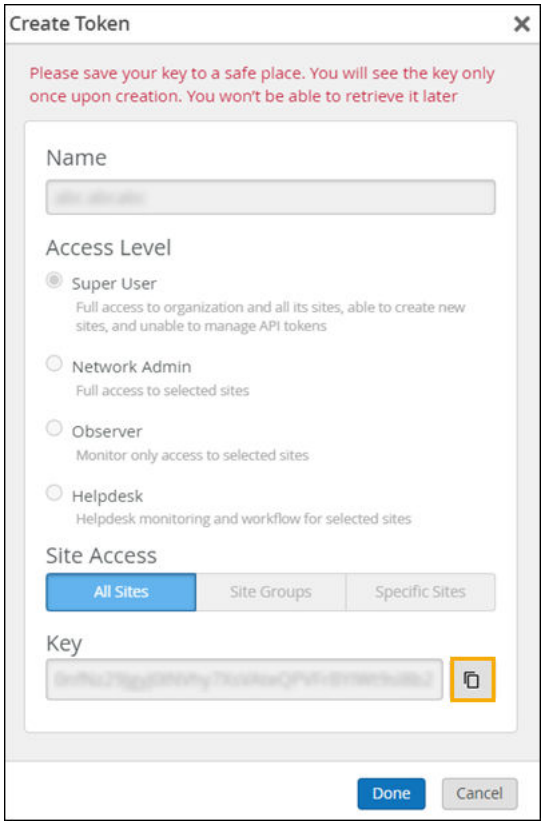
To create an API token:

1. From the left menu of the Juniper Mist portal, select **Organization** > **Settings**.
2. Click **Create Token**.
3. Enter the information.

> **TIP**:
>
> - For Access Level, select **Helpdesk** to give Juniper Mist Alert Format Relay read-only access to alarms and events.
> - You cannot change the site access. The default selection, All Sites, ensures that your token applies to all sites in your organization.

4. Click **Generate**.
5. Next to the **Key** field, click the copy button and save the key in a file where you can retrieve it later.

> ⚠ **CAUTION**: The key appears only when you create the token. When you close the Create Token window, the key is no longer available anywhere in the portal.



6. Click **Done** to close the window.

CHAPTER 3

Installation and Configuration

Information Needed for Installation | 15

Install Juniper Alert Format Relay | 16

Enable Webhooks and Alerts by Using the API | 26

Enable Webhooks and Alerts by Using the Juniper Mist Portal | 29

Select Alerts and Events to Monitor | 31

Load the MIB File | 32

Information Needed for Installation

Before you install Juniper Alert Format Relay, gather this information.

- Virtual Machine Details
  - Login credentials for SSH access
  - Root user credentials
  - Hostnames
- IP and Ports Details
  - Private IP address, Subnet Mask and Gateway of the VMs
  - Private port number
  - Virtual IP address for Juniper Mist Alert Format Relay
  - Public IP address/FQDN
  - Public port number
  - CA Certificate (optional)
  - NTP server IP address
  - DNS IP address
- Juniper Mist Details
  - Organization ID
  - API Token – read only. Access Level: Helpdesk (Helpdesk monitoring and workflow for selected sites)
  - Juniper Mist cloud instance (Global01/Global02/Global03/Global04/Europe01)
  - Juniper Mist API end point. (for example, api.mist.com)
- Monitoring System
  - SNMP manager or network monitoring tool IP address
  - SNMP port number
  - SNMPv3 engine ID

- SNMPv3 username
- SNMPv3 authentication key
- SNMPv3 private key
- SYSLOG server IP address
- SYSLOG protocol – TCP/UDP
- SYSLOG port number

Install Juniper Alert Format Relay

SUMMARY

To install Juniper Alert Format Relay, complete the pre-installation checklist and all these procedures.

IN THIS SECTION

- Pre-Installation Checklist | 16
- Prepare the VM Hosts | 17
- Set Up the Local Registry | 18
- Update Configurations | 19
- Deploy the Installation Package | 23

Pre-Installation Checklist

- Gather the information that you'll need to complete the installation. See "Information Needed for Installation" on page 15.
- Ensure that you have three newly installed virtual machines (VMs) available for the installation. See "Virtual Machine Specifications" on page 6.
- For all three VMs, verify:
  - You can reach the VM through SSH.
  - The VM can reach:
    - Default gateway

- APT/YUM repository/mirror
- Juniper Mist cloud API gateway
- Network monitoring tool
- SYSLOG server

- NTP is configured, and the time is synchronized across all VMs.
- DNS is configured, and DNS resolution is working.

Prepare the VM Hosts

1. Log in to the deployer VM and create the afr directory under /opt:

```
mkdir -p /opt/afr
```

2. Copy the installation bundle TAR file **afr-release-<RELEASE-TAG>.tar.gz** to the deployer VM and extract the installation bundle under /opt/afr:

```
tar -xzvf afr-release-<RELEASE-TAG>.tar.gz -C /opt/afr
```

3. In /opt/afr/, navigate afr-release-<RELEASE-TAG> directory:

```
cd /opt/afr/afr-release-<RELEASE-TAG>
```

In the remaining steps, continue to work from /opt/afr/afr-release-<RELEASE-TAG> as the current working directory.

4. Create a cluster SSH key:

```
ssh-keygen -t rsa -b 4096 -f /root/.ssh/kube_cluster_key
chmod 600 /root/.ssh/kube_cluster_key
```

Set Up the Local Registry

1. Add a afr-registry entry to /etc/hosts on all the four nodes:

```
#On Deployer VM
echo "127.0.0.1 afr-registry" >> /etc/hosts
#AFR VMs
echo "<deployer-vm-ip> afr-registry" >> /etc/hosts
```

2. Enter the necessary environment variables in the afr.env file.

```
afr.env
-------
RELEASE_TAG=v0.0.x
ANSIBLE_SSH_USER=root
ANSIBLE_SSH_KEY=/root/.ssh/kube_cluster_key
MIST_API_URL=
MIST_API_TOKEN=
SNMP_VERSION=<v2c|v3>
SNMP_COMMUNITY=
SNMP_USER_NAME=
SNMP_ENGINE_ID=
SNMP_AUTH_PASS=
SNMP_PRIV_PASS=
SNMP_ADDR=<SNMPMaangerIP>:<Port>
SNMP_AUTH_PROTO=<MD5|SHA>
SNMP_PRIV_PROTO=<DES|AES>
SYSLOG_ADDR=tcp|udp://<SyslogServerIP>:<Port>
```

3. Apply the environment variables:

```
export $(cat afr.env | xargs)
```

4. Run the setup-registry.sh to bring up the local registry:

```
chmod +x setup-registry.sh
./setup-registry.sh
```

NOTE: This script brings up the afr-registry, which serves as a container registry, file server, and Helm repository. The script relies on docker commands to bring up the container.

Update Configurations

1. On the deployer VM, navigate to the inventory directory and update the hosts.yml and overrides.yml files.

   NOTE: You can create a copy of the provided sample and update the details.

2. In hosts.yml file, update the VM IP addresses in the vars section.

```
# Sample Ansible hosts file
# Replace node names according to need.
---
all:
  hosts:
    node1:
      ansible_host: "{{ node1_ip }}"
      ip: "{{ node1_ip }}"
      access_ip: "{{ node1_ip }}"
      ansible_ssh_user: "{{ lookup('env', 'ANSIBLE_SSH_USER') }}"
      ansible_ssh_private_key_file: "{{ lookup('env', 'ANSIBLE_SSH_KEY') }}"
    node2:
      ansible_host: "{{ node2_ip }}"
      ip: "{{ node2_ip }}"
      access_ip: "{{ node2_ip }}"
      ansible_ssh_user: "{{ lookup('env', 'ANSIBLE_SSH_USER') }}"
      ansible_ssh_private_key_file: "{{ lookup('env', 'ANSIBLE_SSH_KEY') }}"
    node3:
      ansible_host: "{{ node3_ip }}"
      ip: "{{ node3_ip }}"
      access_ip: "{{ node3_ip }}"
      ansible_ssh_user: "{{ lookup('env', 'ANSIBLE_SSH_USER') }}"
      ansible_ssh_private_key_file: "{{ lookup('env', 'ANSIBLE_SSH_KEY') }}"
    localhost:
      ansible_connection: local
```

```
      ansible_python_interpreter: "/usr/local/bin/python3"
  children:
    ansible_controller:
      hosts:
        localhost:
    kubespray:
      hosts:
        localhost:
    kube_control_plane:
      hosts:
        node1:
        node2:
        node3:
    kube_node:
      hosts:
        node1:
        node2:
        node3:
    etcd:
      hosts:
        node1:
        node2:
        node3:
    k8s_cluster:
      children:
        kube_control_plane:
        kube_node:
    keepaliveds:
      children:
        k8s_cluster:
    calico_rr:
      hosts: {}
  vars:
    node1_ip: <AFR-VM1-IP>
    node2_ip: <AFR-VM2-IP>
    node3_ip: <AFR-VM3-IP>
```

3. In the overrides.yml file, update the VIP address and the DNS server IP address. Update webhook_whitelist_range with MIST source IPs. For the complete list of source IPs for webhooks, see in the Juniper Mist™ Management Guide.

   Depending on whether you have a public CA certificate, follow the appropriate action below.

- If you have a public CA certificate:
  - Set ingress_use_self_signed_certs as **false** in the overrides.yml file.
  - Place the certificate and the key file in the afr-deployer container.
  - Update the path for the certificate and key in the overrides.yml file.

```
ingress_use_self_signed_certs: false
ingress_tls_cert_path: "/tmp/tls.crt"
ingress_tls_key_path: "/tmp/tls.key"
```

- If you don't have a public CA certificate:
  - Set ingress_use_self_signed_certs as **true**.

  . With this configuration, a self-signed certificate is generated and applied automatically.

```
ingress_use_self_signed_certs: true
```

```
---
# Sample Ansible Overrides
## Alert Format Relay

# O F F L I N E
offline_setup: true
# Registry overrides
registry_host: "afr-registry:8443"
files_repo: "https://afr-registry:8443/files"

## Kubespray K8s installer vars
kubespray_container_name: kubespray
kubespray_config_dir: "/etc/{{ kubespray_container_name }}/inventory"
kubespray_log_dir: "/etc/{{ kubespray_container_name }}/logs"

# Kubernetes version
kube_version: v1.27.5
kube_log_level: 2

image_arch: amd64
etcd_version: v3.5.7
```

```
cni_version: v1.3.0
crictl_version: v1.27.1
calico_version: v3.25.2
helm_version: v3.12.3
containerd_version: 1.7.5
nerdctl_version: 1.5.0

# K8s networking
kube_service_addresses: "172.16.0.0/16"
kube_pods_subnet: "192.168.0.0/16"

# cluster_name: cluster.local
# container_manager: containerd
# kube_network_plugin: calico
# helm_enabled: true

# MetalLB
kube_cluster_vip: "<AFR-VIP>"

# DNS
upstream_dns_servers:
  - <DNS-Server>

# Ingress
ingress_use_self_signed_certs: true

# specify if CA cert has to be loaded
ingress_tls_cert_path: "/tmp/tls.crt"
ingress_tls_key_path: "/tmp/tls.key"

# Ingress LB service
ingress_service_name: ingress-nginx-svc
ingress_tls_secret_name: ingress-tls
ingress_controller_name: ingress-nginx-controller
ingress_controller_ns: ingress-nginx

## AFR Infra overrides

# Base Path for K8s local Persistent volumes storage
local_storage_base_dir: "/mnt/disks"

## Juniper Alert Format Relay (AFR) apps
## webhook_whitelist_range: 54.193.71.17, 54.215.237.20
```

```
webhook_whitelist_range: <MIST-Webhook-SourceIPs>

afr_overrides:
  webhook:
    app_container_name: webhook
    app_image_name: "{{ registry_host }}/webhook:{{ afr_release }}"
  transformer:
    app_container_name: transformer
    app_image_name: "{{ registry_host }}/template_engine:{{ afr_release }}"
  api-client:
    app_container_name: api-client
    app_image_name: "{{ registry_host }}/api_client:{{ afr_release }}"
  dispatcher-syslog:
    app_container_name: dispatcher-syslog
    app_image_name: "{{ registry_host }}/dispatcher:{{ afr_release }}"
  dispatcher-snmp:
    app_container_name: dispatcher-snmp
    app_image_name: "{{ registry_host }}/dispatcher:{{ afr_release }}"
```

Deploy the Installation Package

1. Navigate to afr directory:

```
cd /opt/afr/afr-release-<RELEASE-TAG>
```

2. Set up the deployer container and log in to the container shell:

```
# Bring up the deployer container
docker run \
  -v "${PWD}/inventory:/deployer/ansible/inventory" \
  -v "/root/.ssh/kube_cluster_key:/root/.ssh/kube_cluster_key" \
  -v "/etc/kubespray:/etc/kubespray" \
  -v "/var/run/docker.sock:/var/run/docker.sock" \
  --env-file "${PWD}/afr.env" \
  --add-host=afr-registry:host-gateway \
  --name afr-deployer \
  -d afr-registry:8443/deployer:${RELEASE_TAG} sleep infinity
```

```
# Shell into the container
docker exec -it afr-deployer bash
```

NOTE: You can monitor the installation progress by opening a new session and opening the install.log:

```
tail -f /etc/kubespray/logs/install.log
```

3. Deploy Kubernetes:

```
# Container Shell

(afr-deployer) cd /deployer/ansible
(afr-deployer)
  ansible-playbook \
    -i inventory/hosts.yml \
    -e @inventory/overrides.yml \
    playbooks/deploy-kubernetes.yml

# Post Deploy
(afr-deployer)
  ansible-playbook \
    -i inventory/hosts.yml \
    -e @inventory/overrides.yml \
    playbooks/post-deploy.yml
```

4. Deploy the infrastructure applications:

```
(afr-deployer) cd /deployer/ansible

# OLM
(afr-deployer)
  ansible-playbook \
    -i inventory/hosts.yml \
    -e @inventory/overrides.yml \
    playbooks/deploy-olm.yml

# Kafka
```

```
(afr-deployer)
  ansible-playbook \
    -i inventory/hosts.yml \
    -e @inventory/overrides.yml \
    playbooks/deploy-kafka.yml

# Redis
(afr-deployer)
  ansible-playbook \
    -i inventory/hosts.yml \
    -e @inventory/overrides.yml \
    playbooks/deploy-redis.yml

# Prometheus
(afr-deployer)
  ansible-playbook \
    -i inventory/hosts.yml \
    -e @inventory/overrides.yml \
    playbooks/deploy-prometheus.yml

# Grafana
(afr-deployer)
  ansible-playbook \
    -i inventory/hosts.yml \
    -e @inventory/overrides.yml \
    playbooks/deploy-grafana.yml
```

5. Deploy the Juniper Alert Format Relay applications:

```
(afr-deployer) cd /deployer/ansible

# Webhook
(afr-deployer)
  ansible-playbook \
    -i inventory/hosts.yml \
    -e @inventory/overrides.yml \
    playbooks/deploy-webhook.yml

# Transformer
(afr-deployer)
  ansible-playbook \
```

```
    -i inventory/hosts.yml \
    -e @inventory/overrides.yml \
    playbooks/deploy-transformer.yml

# Dispatcher
(afr-deployer)
  ansible-playbook \
    -i inventory/hosts.yml \
    -e @inventory/overrides.yml \
    playbooks/deploy-dispatcher.yml

# API Client
(afr-deployer)
  ansible-playbook \
    -i inventory/hosts.yml \
    -e @inventory/overrides.yml \
    playbooks/deploy-api-client.yml
```

6. To complete the deployment process, get the MIB file and load it onto your network monitoring tool to process the traps.

```
/opt/afr/afr-release-<RELEASE-TAG>/mibs
```

See "Load the MIB File" on page 32.

Enable Webhooks and Alerts by Using the API

SUMMARY

Here's how to set up webhooks by using the Juniper Mist™ API.

You'll go through this procedure three times to set up three webhooks for Juniper Alert Format Relay:

- Alarms

- Device Events
- Device Up/Downs

> **NOTE**: You can enable webhooks by using either the API or the Juniper Mist portal. Also see "Enable Webhooks and Alerts by Using the Juniper Mist Portal" on page 29.

Before You Begin

You'll need information from your pre-installation tasks to replace the placeholder values in the sample payload. See "Information Needed for Installation" on page 15.

Specifically, you'll need:

- The public IP address and the port that you configured for Juniper Mist Alert Format Relay
- An API token with Super User access
- Your Juniper Mist organization ID

To enable webhooks and alerts by using the API:

1. In your API platform, paste the code shown below. You'll edit it in the next step.

```
POST: https://api.mist.com/api/v1/orgs/<org-id>/webhooks
Headers:
'Authorization': 'token <token>',
'Content-Type': 'application/json'
 Body:
{
 "name": "AFR-Alarms",
 "url": "http/s://<AFR-FQDN/PublicIP>:<Port>/webhook/mist/v1/alarms/",
 "secret": "",
 "enabled": true,
 "verify_cert": true,
 "topics": [
 "alarms"
 ],
 "for_site": false,
 "org_id": “<Org_id>” # eg:"9819c633-08da-4a9a-a267-7e349d215e64
 }
 Body:
 {
 "name": "AFR-Events",
```

```
"url": " http/s://<AFR-FQDN/PublicIP>:<Port>/webhook/mist/v1/device-events/",
"secret": "",
"enabled": true,
"verify_cert": false,
"topics": [
"device-events"
],

"for_site": false,
"org_id": “<Org_id>” # eg:"9819c633-08da-4a9a-a267-7e349d215e64
}
 Body:
 {
       "name": "AFR-Events",
       "url": "https://<AFR-FQDN/PublicIP>:<PublicPort>/webhook/mist/v1/device-updowns/",
       "secret": "",
       "enabled": true,
       "verify_cert": true, #false - if CA Cert is not available
       "topics": [
           "device-updowns"
       ],

       "for_site": false,
       "site_id": "00000000-0000-0000-0000-000000000000",
       "org_id": “<Org_id>” # eg:"9819c633-08da-4a9a-a267-7e349d215e64
   }
```

2. In the AFR-Alarms and AFR-Events sections of the code, replace the placeholder values:
   - Replace <AFR-FQDN/PublicIP>:<Port> with the public IP address and the port that you configured for Juniper Alert Format Relay.
   - Replace <token> with the key for the API token that you created for Juniper Alert Format Relay.
   - Replace <Org_id> with the actual ID for your Juniper Mist organization.
3. Send the API request.
4. In the Juniper Mist portal, select the exact alarms and events that you want to monitor.

   See "Select Alerts and Events to Monitor" on page 31.

Enable Webhooks and Alerts by Using the Juniper Mist Portal

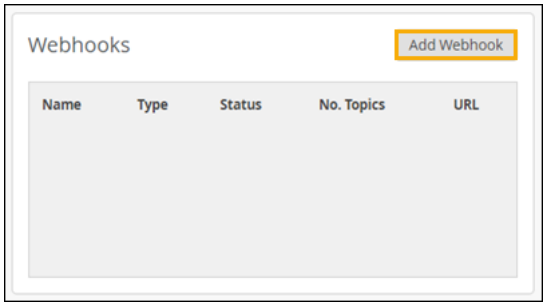
Here's how to set up webhooks in the Juniper Mist™ portal.

NOTE: You can enable webhooks by using either the API or the Juniper Mist portal. Also see "Enable Webhooks and Alerts by Using the API" on page 26.

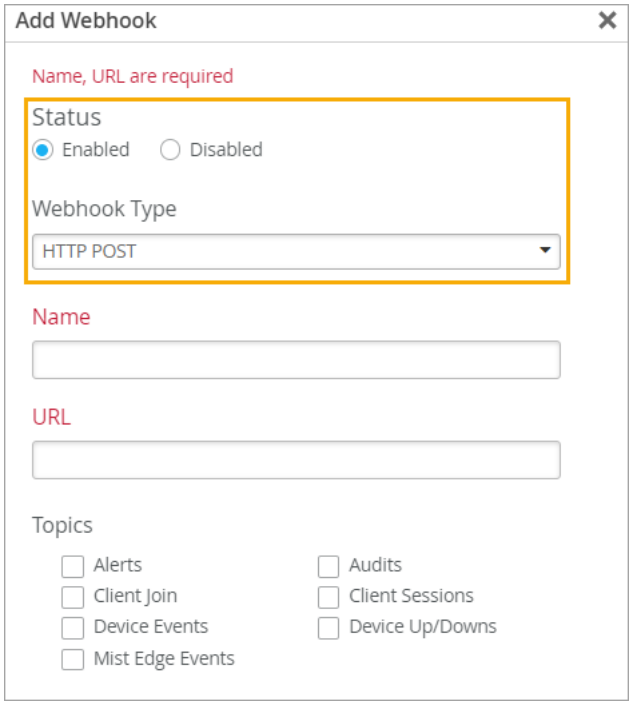
You'll go through this procedure three times to set up three webhooks:

- Alarms
- Device Events
- Device Up/Downs

1. From the left menu of the Juniper Mist portal, select **Organization** > **Settings**.
2. In the Webhooks section, click **Add Webhook**.



3. Keep the default settings for **Status** (Enabled) and **Webhook Type** (HTTP POST).



4. Refer to the information below to enter the Name, URL, and Topic.

   Refer to the Alarms Webhook column the first time that you go through this procedure. Refer to the other columns when you repeat the procedure for the other two webhooks.

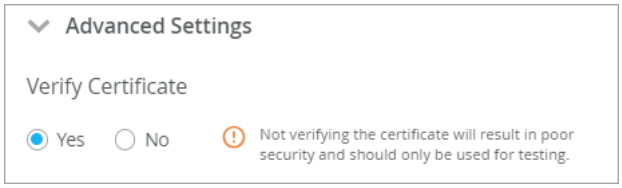
Table 6: Webhook Setup Details

	Alarms Webhook	Events Webhook	Up/Downs Webhook
Name	AFR-Alarms	AFR-Events	AFR-Updowns
URL	https://<AFR-FQDN/PublicIP>:<PublicPort>/webhook/mist/v1/alarms/	https://<AFR-FQDN/PublicIP>:<PublicPort>/webhook/mist/v1/device-events/	https://<AFR-FQDN/PublicIP>:<PublicPort>/webhook/mist/v1/device-updowns/
Topics	Alerts	Device Events	Device Up/Downs

NOTE: The Device Up/Downs webhooks topic is a subset of device events that include only disconnected, reconnected, and restarted events. This currently supports APs, switches, and gateways (WAN Edges). If you use the Device Events topic, you will receive duplicates of these events.

It's not necessary to configure both AFR-events and AFR-updowns. You only need to configure one.

5. Click **Advanced Settings**, and then select the appropriate option:
   - If you have a CA certificate, click **Yes**.
   - If you do not have a CA certificate, click **No**.



6. Click **Add**.
7. Repeat this procedure until you've added all three webhooks.

Select Alerts and Events to Monitor

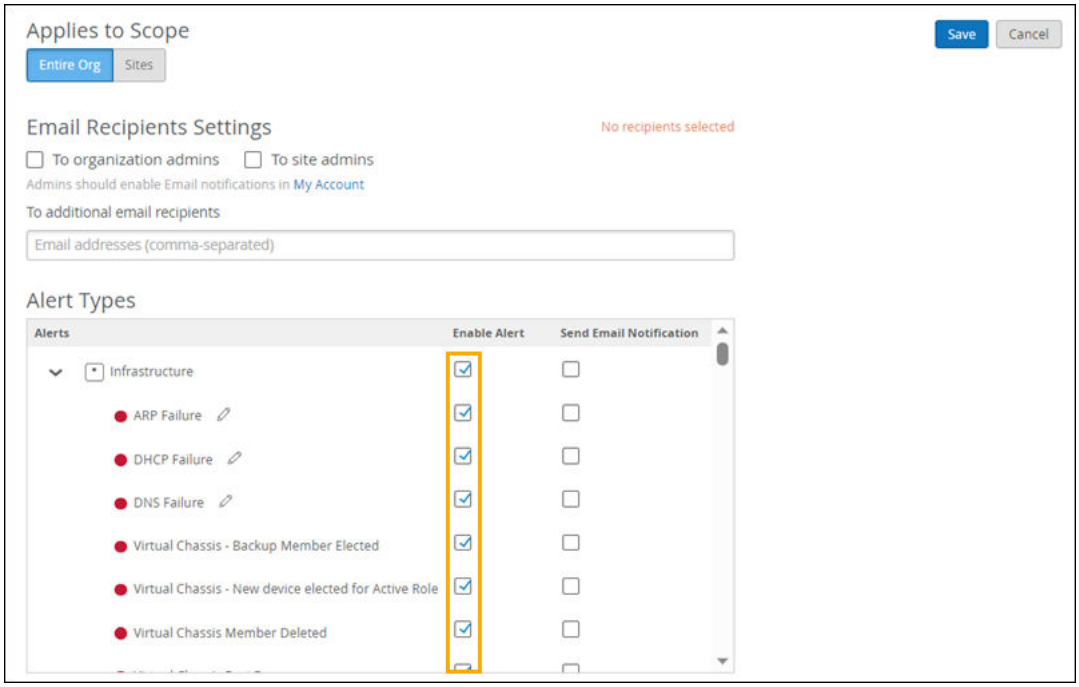
After you enable webhooks, select the alerts and events that you want to monitor.

1. From the left menu of the Juniper Mist™ portal, select **Monitor** > **Alerts**.
2. Click **Alerts Configuration** (near the top right corner of the page).

3. Select the check box for each type of alert or event that you want to monitor.

NOTE: If you want to select all options in a category, such as Infrastructure, select the check box at the top of that section.

Example



Load the MIB File

The final step in the setup process is to load the MIB file in your network monitoring tool.

This file defines the SNMP trap generated from Juniper Alert Format Relay so that your network monitoring tool can display the alerts that it receives through the relay.

You can find the MIB file at the following path on the deployer VM.

```
/opt/afr/afr-release-<RELEASE-TAG>/mibs
```