NetFlow Data Collection

This section contains the following topics:

NetFlow Data Collection

WAE can collect and aggregate exported NetFlow and related flow measurements. These measurements can be used to construct accurate demand traffic data for WAE Design. Flow collection provides an alternative to the estimation of demand traffic from interfaces, LSPs, and other statistics using Demand Deduction. NetFlow gathers information about the traffic flow and helps to build traffic and demand matrix. Importing flow measurements is particularly useful when there is full or nearly full flow coverage of a network's edge routers. Additionally, it is beneficial when accuracy of individual demands between external autonomous systems (ASes) is of interest.

Network data collected separately by NIMOs, including topology, BGP neighbors, and interface statistics, is combined with the flow measurements to scale flows and provide a complete demand mesh between both external autonomous systems and internal nodes.

WAE gathers the following types of data to build a network model with flows and their traffic measurements aggregated over time:

- Flow traffic using NetFlow, JFlow, CFlowd, IPFIX, and Netstream flows
- Interface traffic and BGP peers over SNMP
- BGP path attributes over peering sessions

NetFlow Collection Architectures

There are two types of flow collection architectures:

Note The collection architecture to deploy depends on the measured or estimated rate of NetFlow traffic export from the network in Mbps or fps.

- Centralized NetFlow (CNF)—Typically used for small to medium networks. This is a single-server architecture.
- Distributed NetFlow (DNF)—Typically used for larger networks. This architecture consists of a JMS broker, master, and agents.

CNF Collection

The following figure shows the workflow for collecting and computing flow data in CNF. The WAE Collector CLI tools, `flow_manage` and `flow_get`, integrate with an external configuration file and the NIMO collection process, respectively. Flow-based demands and demand traffic are passed to the WAE YANG run-time system.

Figure 1: Centralized Collection and Demand Creation

- **`flow_manage`**—This CLI tool configures network connectivity and manages the collection server, including starting, stopping and configuring the flow collection process. It uses input from the <NodeFlowConfigs> table from a configuration file to generate configuration information, which it then sends to the flow collection server.
- **Flow collection server**—This background process receives configuration information from `flow_manage`, which it uses to configure the collection server and receive flow data and BGP attributes. The collection server then aggregates this data and forwards the microflows file to the `flow_get` tool.
- **`flow_get`**—This CLI tool is configured inside the `nimo_flow_get.sh script` and is executed within the external-executable-nimo. It reads flow data (microflows file) from the collection server, produces NetFlow demands and demand traffic data, and inserts this data into the WAE YANG run-time database. In addition to producing demand and traffic data, `flow_get` also produces inter-AS (IAS) flow files.

Note In production networks, do not use -log-level=INFO | DEBUG | TRACE for `flow_get`.

DNF Collection

The following figures show the DNF architecture and the DNF workflow. In this architecture, each set of network devices exports flow data to a corresponding collection server. The DNF cluster performs flow computation so that each agent is responsible for the flow computation of its corresponding flow collection server that runs the flow collector. The master node aggregates this information and passes it back to `flow_collector_ias`.

Figure 2: DNF Architecture

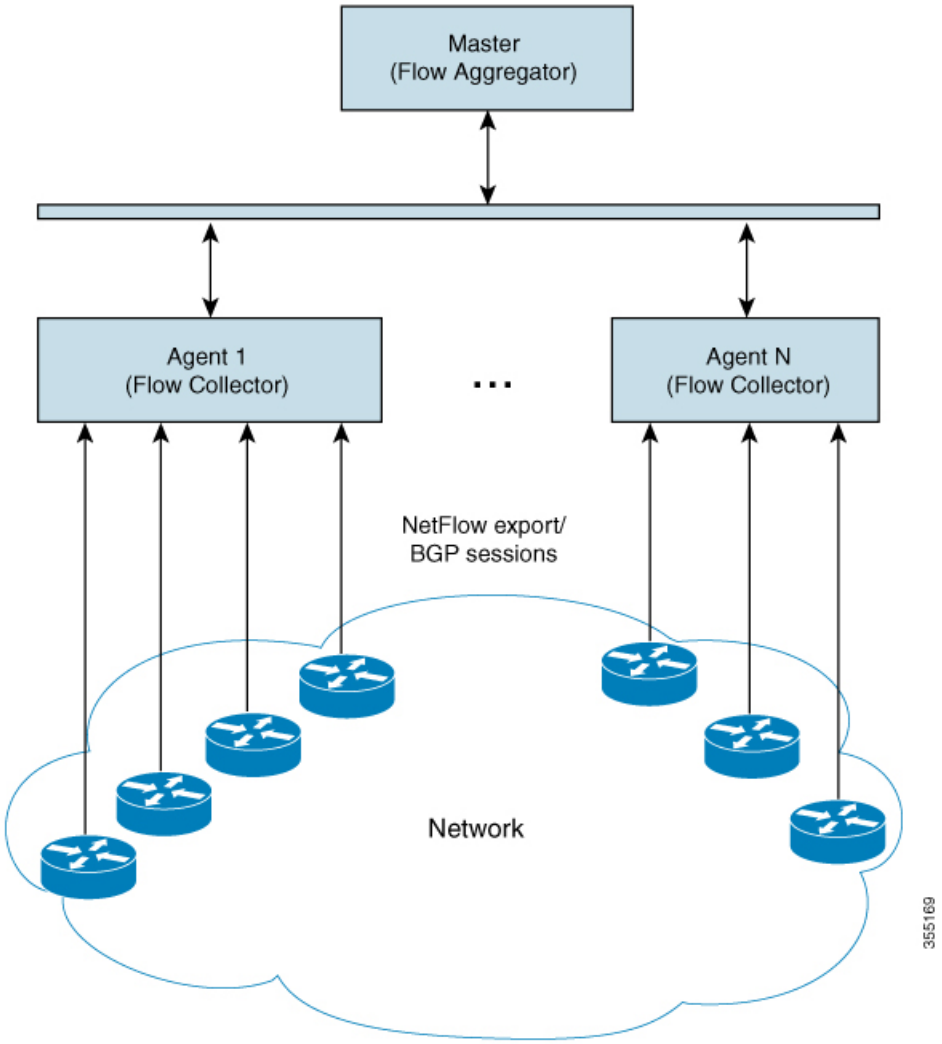

Figure 3: DNF Collection Workflow

- **`flow_cluster_manage`**—This CLI tool is used to configure and get status from the cluster. It takes a cluster configuration file and sends the configuration to the cluster.

  A REST API is also available to configure and request status from the cluster as an alternative to using `flow_cluster_manage`. For more information, see the API documentation from one of the following locations:

  - `<wae-installation-directory>docs/api/netflow/distributed-netflow-rest-api.html`
  - http://<*master-IP-address*>:9090/api-doc For example, to get the cluster configuration:

    For example, to get the cluster configuration:

    ```
    curl -X GET http://localhost:9090/cluster-config > config-file-1
    ```

For example, to set the cluster configuration:

```
curl -X PUT http://localhost:9090/cluster-config @config-file-2
```

For example, to get the cluster status:

```
curl -X GET http://localhost:9090/cluster-status > config-file-1
```

- **flow_cluster_master**—The master service collects all flow data results from all the agents and aggregates the data, which is sent back to `flow_collector_ias`.
- **flow_cluster_agent**—The agent service manages and tracks the status of the associated flow collector. Each agent receives and computes the flow data from its corresponding collection server.
- **flow_cluster_broker**—(not shown in diagram) The JMS broker service allows communication between all components within the architecture, including master and agents.
- **flow_collector_ias**—This CLI tool, which is configured inside the `nimo_flow_collector_ias_and_dmd.sh` file and is executed within the external-executable-nimo, receives the flow data from the master and produces the IAS flows file.
- **flow_collector_dmd**—This CLI tool sends NetFlow demands and demand traffic to the WAE YANG run-time database. This is configured inside the `nimo_flow_collector_ias_and_dmd.sh` file and is executed within the external-executable-nimo.

Note In production networks, do not use -log-level=INFO | DEBUG | TRACE for `flow_collector_ias` or `flow_collector_dmd`.

Centralized NetFlow Configuration Workflow

To configure CNF and start collection:

Note Unless stated otherwise, do not change permissions on files that were deployed during WAE installation.

Step 1 Confirm that the CNF NetFlow Requirements , on page 4 are met.

Step 2 Prepare the Operating System for CNF, on page 5

Step 3 Create the CNF Configuration File, on page 6

Step 4 Configure CNF Collection, on page 7

a) Configure the netflow-nimo for CNF, on page 7

CNF NetFlow Requirements

For system requirements, see the *Cisco WAE System Requirements* document.

Licensing

Confirm with your Cisco WAE representative that you have the correct licenses for getting flow and flow demands when using the `flow_manage` and `flow_get` tools.

Prepare the Operating System for CNF

To prepare the OS for CNF, run the following `flow_manage` command from the WAE CLI:

```
sudo -E ./flow_manage -action prepare-os-for-netflow
```

The `prepare-os-for-netflow` option does the following:

- Uses the `setcap` command to allow non-root users limited access to privileged ports (0-1023). This is necessary when configuring the flow collector to use a port under 1024 to listen to BGP messages.
- Configures the OS instance to reserve up to 15,000 of file descriptors to account for the large number of temporary files that may be produced by `flow_get` in a CNF architecture.

Note After executing this command, you must reboot the server.

NetFlow Collection Configuration

The flow collection process supports IPv4 and IPv6 flows captured and exported by routers in the ingress direction. It also supports IPv4 and IPv6 iBGP peering.

Routers must be configured to export flows to and establish BGP peering with the flow collection server. Note the following recommendations:

- NetFlow v5, v9, and IPFIX datagram export to the UDP port number of the flow collection server, which has a default setting of 2100. Export of IPv6 flows requires NetFlow v9 or IPFIX.
- Configure the flow collection server on the routers as an iBGP route reflector client so that it can send BGP routes to edge or border routers. If this is not feasible, configure a router or route server that has a complete view of all relevant routing tables.
- Configure the source IPv4 address of flow export data grams to be the same as the source IPv4 address of iBGP messages if they are in the same network address space.
- Explicitly configure the BGP router ID.
- Configure static routing.
- If receiving BGP routes, the maximum length of the BGP **`AS_path`** attribute is limited to three hops. The reason is to prevent excessive server memory consumption, considering that the total length of BGP attributes, including **`AS_path`**, attached to a single IP prefix can be very large (up to 64 KB).

Create the CNF Configuration File

The <NodeFlowConfigs> table contains basic node configuration information used by the `flow_manage` tool when generating configuration information that it passes to the flow collection server. Thus, prior to executing `flow_manage`, you must construct this table as follows:

- Use a tab or comma delimited format.
- Include one row per node (router) from which you are collecting flow data.
- Enter contents described in the following table for each of these nodes. The BGP columns are required only if collecting BGP information.

Table 1: <NodeFlowConfigs> Table Columns

Column	Description
Name	Node name
SamplingRate	Sampling rate of the packets in exported flows from the node. For example, if the value is 1,024, then one packet out of 1,024 is selected in a deterministic or random manner.
FlowSourceIP	IPv4 source address of flow export packets.
BGPSourceIP	IPv4 or IPv6 source address of iBGP update messages. This column is needed if the `flow_manage -bgp` option is true.
BGPPassword	BGP peering password for MD5 authentication. Use this column if the `flow_manage -bgp` option is true and if BGPSourceIP has a value.

The following is a <NodeFlowConfigs> Table example:

Name	SamplingRate	FlowSourceIP	BGPSourceIP	BGPPassword
paris-er1-fr	1024	192.168.75.10	69.127.75.10	ag5Xh0tGbd7
chicago-cr2-us	1024	192.168.75.15	69.127.75.15	ag5Xh0tGbd7
chicago-cr2-us	1024	192.168.75.15	2001:db9:8:4::2	ag5Xh0tGbd7
tokyo-br1-jp	1024	192.168.75.25	69.127.75.25	ag5Xh0tGbd7
brazilia-er1-bra	1024	192.168.75.30	2001:db8:8:4::2	ag5Xh0tGbd7

Configure CNF Collection

Configure the netflow-nimo for CNF

Before you begin

- You must have a source network model. This is the final network model which includes topology collection and any other NIMO collections you want to include.

Step 1 From the Expert Mode, navigate to **/wae:networks**.

Step 2 Click the plus (+) sign and enter a network model name. We recommend a unique name that is easily identifiable; for example, networkABC_CNF_flow_get.

Step 3 Click the **nimo** tab.

Step 4 From the **Choice - nimo-type** drop-down list, choose **netflow-nimo**.

Step 5 Click **netflow-nimo** and select the **source-network** and **storage-format** from the drop down list.

Note We recommend that the storage-format is set to "native" for faster performance. If the storage-format is set to "native", then source should be set to "file".

Step 6 Click the **config** tab.

Step 7 Click **common** and enter the following information:

- **split-as-flows-on-ingress**—Select the traffic aggregation strategy for external ASNs.
- **asn**—Enter the ASN of the internal AS in the network.
- **address-family**—Select the protocol version to include in IAS flows and demands computation.
- **number-of-threads**—Enter the maximum number of threads to be used in parallel computation.
- **ext-node-tags**—Enter a list of one or more node tags separated by a comma.
- **extra-aggregation**—Enter a list of aggregation keys separated by a comma.
- **log-level**—Select the log level of the tool.

Step 8 Click **ias-flows** and enter the following information:

- **delegated-computation-timeout-seconds**—Enter the number of seconds to wait before timing out when delegating the computation of IAS flows into the distributed cluster.
- **trim-inter-as-flows**—Enter the value in MBits/sec below which the inter-as-flows for traffic is strictly discarded.
- **match-on-bgp-external-info**—Select whether to match egress IP addresses in the BGP peer relation.
- **flows-dir**—Enter the directory containing flow matrix files to import. The file will be removed immediately after imported.
- **flows-file**—Enter the file path containing flow matrix files to import. The file will be removed immediately after imported.
- **ingress-interface-flow-filter**—Enter a filter of node and interface in the form Node:InterfaceName that will be applied while reading the flow matrix to filter in only those ingress interfaces.
- **egress-interface-flow-filter**—Enter a filter of node and interface in the form Node:InterfaceName that will be applied while reading the flow matrix to filter in only those egress interfaces.
- **backtrack-micro-flows**—Select whether to generate files showing a relationship between micro flows from the input file and those demands or inter-as-flows that aggregate them.
- **flow-import-flow-ids**—Enter comma separated flow IDs to import data from. Use " to import from all flows.

Step 9 Click **demands** and enter the following information:

- **demand-name**—Enter a namee for any new demands.
- **demand-tag**—Enter a tag for any new demands, or to be appended to existing tag demands.
- **trim-demands**—Specify the value in MBits/sec below which the demands are strictly discarded.
- **service-class**—Specify the demand service class.
- **traffic-level**—Specify the demand traffic level.
- **missing-flows**—Enter the path where file with interfaces that are missing flows is generated.

Step 10 Click **run-netflow-collection** > **Invoke run-netflow-collection**

DNF NetFlow Configuration Workflow

To configure DNF and start collection:

Note Unless stated otherwise, do not change permissions on files that were deployed during WAE installation.

Step 1 Confirm that the Distributed NetFlow Requirements, on page 8 are met.

Step 2 Set Up the DNF Cluster, on page 10

a) Modify the DNF Configuration Files, on page 10

Step 3 Configure DNF Cluster, on page 11

a) Create the DNF Cluster Configuration File, on page 11

Step 4 Configure DNF Collection, on page 14

a) Configure the netflow-nimo for DNF, on page 14

Distributed NetFlow Requirements

For system requirements, see the *Cisco WAE System Requirements* document.

In addition, the following are required for all cluster elements (master, agents, JMS Broker):

- Ansible 2.1 or later.
- Java virtual machine (JVM) has the same installation path for all elements. The java executable should be in the path readable for all users.
- A sudo SSH user with the same name in each server dedicated for the cluster (broker, master, and all the agents) must exist. Make a note of this user name because it is used in the group_vars/all Ansible file (discussed later in this section).

  WAE Planning software must be installed on a server (installation server) with the appropriate license file.

- Agent system requirements meet the same requirements needed for WAE installation.
- The flow collection process supports IPv4 and IPv6 flows captured and exported by routers in the ingress direction. It also supports IPv4 and IPv6 iBGP peering. Routers must be configured to export flows to and establish BGP peering with the flow collection server.

Licensing

Confirm with your Cisco WAE representative that you have the correct licenses for getting flow and flow demands when using the `flow_cluster_master`, `flow_collector_ias`, and `flow_collector_dmd` tools.

Java Message Server (JMS) Broker

Each distributed flow collection setup must have a single JMS broker instance in order for the master, agents, and client within a cluster to exchange information. All information is interchanged through the broker and enables all the components to communicate with each other. DNF supports a dedicated JMS broker.

The broker must have the following features enabled in order for all JMS clients (master, agents, and `flow_collector_ias` instances) to work:

- Out of band file messaging
- Support of obfuscated passwords in configuration files

Master and Agents

Ansible files are used to install and run DNF configuration on the JMS broker, master, and agent servers.

Master

The master node provides the following services in the cluster:

- Monitors and tracks agent status.
- Monitors and tracks the status of the last completed IAS computation.
- Aggregates IAS flow data coming from all agents back to the client.
- Handles configuration and status requests from the cluster.

Agents

Only one agent per server is supported. Agents cannot be on the WAE installation or data collection server. Each agent receives and computes flow data from its corresponding collection server.

Note You have the option to deploy only one agent in the cluster. This is an alternative to CNF for networks that are expected to expand in size or grow in traffic.

Set Up the DNF Cluster

Modify the DNF Configuration Files

If you use default WAE installation options, there are only a few mandatory parameters that must be changed. These will be noted in the applicable configuration topics. The topics described in this section assume the following:

- The master server (installation server) is where the WAE planning software has been installed and default directories are used. In particular, the configuration files used for DNF on the installation server are located in *<wae_installation_directory>*/etc/netflow/ansible.
- A dedicated JMS broker will be used in DNF configuration.
- In configuration examples, the following values are used:
  - Master and JMS broker IP address—198.51.100.10
  - Agent 1 IP address—198.51.100.1
  - Agent 2 IP address—198.51.100.2
  - Agent 3 IP address—198.51.100.3

group_vars/all

The file is located in *<WAE_installation_directory>*/etc/netflow/ansible/group_vars/all. This file is the Ansible file that contains the variable definitions that are used in the playbook files.

Edit the following options:

Option	Description
LOCAL_WAE_INSTALLATION_DIR_NAME	The local path that contains the WAE installation file.
WAE_INSTALLATION_FILE_NAME	The filename of the WAE installation file.
TARGET_JDK_OR_JRE_HOME	The full path and filename of the Oracle JRE file. All machines in the cluster (broker, master, and all the agents) should have the JRE previously installed under this variable.
LOCAL_LICENSE_FILE_PATH	The full path to the license file.
SSH_USER_NAME	The SSH user name created or used when SSH was enabled on each machine. This sudo user is used by Ansible to deploy the cluster over SSH.

For example (comments removed):

```
LOCAL_WAE_INSTALLATION_DIR_NAME: "/wae/wae-installation"
WAE_INSTALLATION_FILE_NAME: "wae-linux-v16.4.8-1396-g6114ffa.rpm"
TARGET_JDK_OR_JRE_HOME: "/usr/lib/jvm/java-1.8.0-openjdk-1.8.0_45"
LOCAL_LICENSE_FILE_PATH: "/home/user1/.cariden/etc/MATE_Floating.lic"
TARGET_SSH_USER: ssh_user
```

Configure DNF Cluster

Create the DNF Cluster Configuration File

To more easily create the cluster configuration file for `flow_manage_cluster`, you can use the CNF configuration file produced from `flow_manage` as a template for the cluster configuration file.

For example:

Step 1 Use the following sample file to create the .json file.

Source the `waerc` file.

```
${CARIDEN_HOME}/flow_manage \
-action produce-config-file \
-node-flow-configs-table <input-path> \
-cluster-config-file <output-path> \
-interval 120 \
-bgp true \
-bgp-port 10179 \
-port 12100 \
-flow-size lab \
-server-ip ::
```

where *<input-path>* is the path of the node configuration .txt file used in CNF (see Configure and Run the Collector Server for more information on creating this file) and *<output-path>* is the path where you want the resulting seed cluster configuration file to reside. Verify that the output of the seed cluster configuration file is similar to the following:

```
{
  "agentConfigMapInfo": {
    "wae-netflow-agent::agent-1": {
      "perAgentDebugMode": null,
      "flowManageConfiguration": {
        "maxBgpdPeers": 150,
        "useBgpPeering": true,
        "outfileProductionIntervalInSecs": 60,
        "networkDeploymentSize": "lab",
        "bgpTcpPort": 179,
        "netflowUdpPort": 2100,
        "daemonOutputDirPath": "<user.home>/.cariden/etc/net_flow/flow_matrix_interchange",
        "keepDaemonFilesOnStartStop": false,
        "purgeOutputFilesToKeep": 3,
        "routerConfigList": [
          {
            "name": "rr3",
            "bGPSourceIP": "172.20.164.147",
            "flowSourceIP": "10.1.1.3",
            "bGPPassword": "",
            "samplingRate": "1"
          }
        ],
        "ipPrefixFilteringList": [],
        "appendedProperties": null,
        "daemonOutputFileMaskPrefix": "out_matrix_",
        "daemonOutputSoftLinkName": "flow_matrix_file-latest",
        "extraAggregation": [],
        "listValidExtraAggregationKeys": false
      }
```

```
    },
    "wae-netflow-agent::agent-2": {
      "perAgentDebugMode": null,
      "flowManageConfiguration": {
        "maxBgpdPeers": 150,
        "useBgpPeering": true,
        "outfileProductionIntervalInSecs": 60,
        "networkDeploymentSize": "lab",
        "bgpTcpPort": 179,
        "netflowUdpPort": 2100,
        "daemonOutputDirPath": "<user.home>/.cariden/etc/net_flow/flow_matrix_interchange",
        "keepDaemonFilesOnStartStop": false,
        "purgeOutputFilesToKeep": 3,
        "routerConfigList": [
          {
            "name": "rr3",
            "bGPSourceIP": "172.20.164.147",
            "flowSourceIP": "10.1.1.3",
            "bGPPassword": "",
            "samplingRate": "1"
          }
        ],
        "ipPrefixFilteringList": [],
        "appendedProperties": null,
        "daemonOutputFileMaskPrefix": "out_matrix_",
        "daemonOutputSoftLinkName": "flow_matrix_file-latest",
        "extraAggregation": [],
        "listValidExtraAggregationKeys": false
      }
    },
  },
  "aggregationMode": "okIfNotAllPortionsArePresent",
  "debugMode": {
    "bypassAnyNfacctdOperation": false
  }
}
```

Step 2 Edit the file to include each agent configuration. Copy, paste, and edit each section as it applies to each agent in the cluster. This example shows two agents:

```
{
    "agentConfigMapInfo": {
        "cluster_1::instance_1":
        {
            "flowManageConfiguration":
            {
                "maxBgpdPeers": 150,
                "bgpTcpPort": 179,
                "flowType": "Netflow",
                "useBgpPeering": true,
                "outfileProductionIntervalInSecs": 900,
                "networkDeploymentSize": "medium",
                "netflowUdpPort": 2100,
                "keepDaemonFilesOnStartStop": true,
                "purgeOutputFilesToKeep": 3,
                "daemonOutputFileMaskSuffix": "%Y.%m.%d.%H.%M.%s",
                "daemonOutputDirPath":
"<user.home>/.cariden/etc/net_flow/flow_matrix_interchange",
                "daemonOutputFileMaskPrefix": "out_matrix_",
                "daemonOutputSoftLinkName": "flow_matrix_file-latest",
                "extraAggregation": [],
                "routerConfigList":
                    [
                        {
```

```
                            "name": "ar1.dus.lab.anyname.com",
                            "bGPSourceIP": "1.2.3.4",
                            "flowSourceIP": "1.2.3.5",
                            "bGPPassword": "bgp-secret",
                            "samplingRate": "666"
                        },
                        {
                            "name": "cr1.ams.lab.anyname.com",
                            "bGPSourceIP": "1.2.3.51",
                            "flowSourceIP": "1.2.3.53",
                            "bGPPassword": "bgp-secret-3",
                            "samplingRate": "8000"
                        }
                    ],
                "appendedProperties":
                {
                    "key1": "value1",
                    "key2": "value2"
            }
        }
},
```

The information for the second agent starts here:

```
        "cluster_1::instance_2":
        {
            "flowManageConfiguration":
            {
                "maxBgpdPeers": 150,
                "bgpTcpPort": 179,
                "flowType": "Netflow",
                "useBgpPeering": true,
                "outfileProductionIntervalInSecs": 900,
                "networkDeploymentSize": "medium",
                "netflowUdpPort": 2100,
                "keepDaemonFilesOnStartStop": true,
                "purgeOutputFilesToKeep": 3,
                "daemonOutputFileMaskSuffix": "%Y.%m.%d.%H.%M.%s",
                "daemonOutputDirPath":
"<user.home>/.cariden/etc/net_flow/flow_matrix_interchange",
                "daemonOutputFileMaskPrefix": "out_matrix_",
                "daemonOutputSoftLinkName": "flow_matrix_file-latest",
                "extraAggregation": [],
                "routerConfigList":
                    [
                        {
                            "name": "ar1.dus.lab.anyname.com",
                            "bGPSourceIP": "5.6.7.8",
                            "flowSourceIP": "5.6.7.9",
                            "bGPPassword": "bgp-secret-2",
                            "samplingRate": "666"
                        },
                        {
                            "name": "cr1.ams.lab.anyname.com",
                            "bGPSourceIP": "5.6.7.81",
                            "flowSourceIP": "5.6.7.83",
                            "bGPPassword": "bgp-secret-4",
                            "samplingRate": "8000"
                        }
                    ],
                "appendedProperties":
                {
                    "key1": "value1",
                    "key2": "value2"
```

```
            }
        }
},
```

Note The .jason file configuration is only needed for Master ans is not required for Slave.

Configure DNF Collection

Configure the netflow-nimo for DNF

Before you begin

- You must have a source network model. This is the final network model which includes topology collection and any other NIMO collections you want to include.

Step 1 From the Expert Mode, navigate to **/wae:networks**.

Step 2 Click the plus (+) sign and enter a network model name. We recommend a unique name that is easily identifiable; for example, networkABC_CNF_flow_get.

Step 3 Click the **nimo** tab.

Step 4 From the **Choice - nimo-type** drop-down list, choose **netflow-nimo**.

Step 5 Click **netflow-nimo** and select the **source-network** and **storage-format** from the drop down list.

Note We recommend that the storage-format is set to "native" for faster performance. If the storage-format is set to "native", then source should be set to "file".

Step 6 Click the **config** tab.

Step 7 Click **common** and enter the following information:

- **split-as-flows-on-ingress**—Select the traffic aggregation strategy for external ASNs.
- **asn**—Enter the ASN of the internal AS in the network.
- **address-family**—Select the protocol version to include in IAS flows and demands computation.
- **number-of-threads**—Enter the maximum number of threads to be used in parallel computation.
- **ext-node-tags**—Enter a list of one or more node tags separated by a comma.
- **extra-aggregation**—Enter a list of aggregation keys separated by a comma.
- **log-level**—Select the log level of the tool.

Step 8 Click **ias-flows** and enter the following information:

- **delegated-computation-timeout-seconds**—Enter the number of seconds to wait before timing out when delegating the computation of IAS flows into the distributed cluster.
- **trim-inter-as-flows**—Enter the value in MBits/sec below which the inter-as-flows for traffic is strictly discarded.
- **match-on-bgp-external-info**—Select whether to match egress IP addresses in the BGP peer relation.
- **flows-dir**—Enter the directory containing flow matrix files to import. The file will be removed immediately after imported.

- **flows-file**—Enter the file path containing flow matrix files to import. The file will be removed immediately after imported.
- **ingress-interface-flow-filter**—Enter a filter of node and interface in the form Node:InterfaceName that will be applied while reading the flow matrix to filter in only those ingress interfaces.
- **egress-interface-flow-filter**—Enter a filter of node and interface in the form Node:InterfaceName that will be applied while reading the flow matrix to filter in only those egress interfaces.
- **backtrack-micro-flows**—Select whether to generate files showing a relationship between micro flows from the input file and those demands or inter-as-flows that aggregate them.
- **flow-import-flow-ids**—Enter comma separated flow IDs to import data from. Use " to import from all flows.

Step 9 Click **demands** and enter the following information:

- **demand-name**—Enter a namee for any new demands.
- **demand-tag**—Enter a tag for any new demands, or to be appended to existing tag demands.
- **trim-demands**—Specify the value in MBits/sec below which the demands are strictly discarded.
- **service-class**—Specify the demand service class.
- **traffic-level**—Specify the demand traffic level.
- **missing-flows**—Enter the path where file with interfaces that are missing flows is generated.

Step 10 Click **run-netflow-collection** > **Invoke run-netflow-collection**