Rapid SNMP device onboarding in Network Performance Insight
Note

Before using this information and the product it supports, read the information in "Notices" on page 37.

This edition applies to version 1.3.0.0 of IBM Network Performance Insight and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright IBM Corporation 2015, 2018.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
Rapid SNMP device onboarding

This information helps you to understand and use the Rapid SNMP device onboarding feature that is available with Network Performance Insight® V1.3.

Intended audience

The audience who is network administrators or operations specialist responsible for installing the Network Performance Insight product suite on an enterprise network.

To install and use the Technology Pack Development Tool to create your custom Technology Pack, you must have a thorough understanding of the following subjects:

- Network Performance Insight 1.3 system
- Basic principles of network protocols and network management
- Discovery concepts
- Polling concepts
- IBM® Netcool® Operations Insight
- IBM Tivoli® Network Manager IP Edition
- SQL concepts

Network Performance Insight architecture

IBM Network Performance Insight is a network performance monitoring system. It offers both real-time and historical trends in network performance and interactive view on the network data that helps in reduced network downtime and optimized network performance.

Network Performance Insight provides IBM Netcool Operations Insight with comprehensive IP network device performance monitoring and session traffic analysis.

The following diagram shows how data is flowing through the various components in Network Performance Insight:
Network Performance Insight services

Network Performance Insight services are running on microservice architecture that has the software application as a suite of independently deployable, small, modular services in which each service runs a unique process and communicates through a well-defined, lightweight mechanism. Currently, Network Performance Insight 1.3 consists of the following microservices:

**Foundation services**
- DNS
- Event
- Manager
- Storage
- UI

**Entity Metric services**
- Cacti Collector
- Exporter
- Formula Service
- Entity Analytics
- SNMP Collector
- Threshold
- Tivoli Network Manager Collector

**Flow Metric services**
- Flow Analytics
- Flow Collector
• SNMP Discovery

For more information about these services, see their respective sections in *IBM Network Performance Insight: Product Overview*.

### Network Performance Insight additional components

Some of the additional components that are introduced in Network Performance Insight V1.3 for enhanced functions are described here:

#### Technology Packs

A set of ready-to-use Technology Packs is provided to perform second-level discovery and polling of resources to collect entity metric data. These Technology Packs can help to collect standard SNMP metrics, and Performance Metric OOTB Device Support metrics.

For more information, see *Installing the Technology Packs* section in *Installing and Configuring IBM Network Performance Insight*.

#### Network Performance Insight Dashboards

These interactive dashboards are the built-in JSON-based dashboards suite that can display aggregated network data from Network Performance Insight database with the help of REST API calls. It supports a combination of data from multiple data sources.

This feature provides a wide variety of dashboards for Network Operators, Network Engineers, and Network Capacity Planners. These dashboards help in pinpointing the troubled resources and general resource performance. A number of web-based configuration options are available to control the data that is displayed on the dashboards.

For more information, see *Network Performance Insight Dashboards* section in *IBM Network Performance Insight: Product Overview*.

**Note:** Networks for Operations Insight is a solution extension of Netcool Operations Insight that includes the following components and products:

• Tivoli Network Manager
• Tivoli Netcool Configuration Manager
• Network Performance Insight
• Network Health Dashboard
• Device Dashboard
• Topology Search

### Hortonworks Data Platform components

Hortonworks Data Platform (HDP®) can be used to help process and analyze the volume, variety, and velocity of data that continually enters your organization every day. Network Performance Insight is installed as a service extension to the installed HDP® stack.

The features of HDP® that are used in Network Performance Insight:

• HDP®
• Default support for rolling upgrades for Hadoop services
• Support for long-running applications within YARN for enhanced reliability
• Spark in-memory distributed compute engine for dramatic performance increase
Apache Ambari operational framework. Apache Ambari is an open framework for provisioning, managing, and monitoring Apache Hadoop clusters. Ambari provides an intuitive and easy-to-use Hadoop management web UI backed by its collection of tools and APIs that simplify the operation of Hadoop clusters.

Essentially includes the following open source technologies for working with Network Performance Insight:

- Apache Hadoop
- Apache Kafka
- Apache Ambari
- Apache Spark
- Apache ZooKeeper

Note: Because Zookeeper requires a majority, it is best to use an odd number of machines. For example, with four machines ZooKeeper can handle the failure of a single machine; if two machines fail, the remaining two machines do not constitute a majority. However, with five machines ZooKeeper can handle the failure of two machines.

Integrated products

Products that are integrated with Network Performance Insight 1.3:

Jazz™ for Service Management
Dashboard Application Services Hub provides visualization and dashboard services in Jazz for Service Management. It has a single console for administering IBM products and related applications. Visualization for Network Performance Insight is federated into Dashboard Application Services Hub.

IBM Tivoli Network Manager IP Edition
Tivoli Network Manager provides first-level device discovery and polling of some standard SNMP metrics.

Tivoli Netcool/OMNIbus component of IBM Netcool Operations Insight
Netcool Operations Insight is powered by the fault management capabilities of IBM Tivoli Netcool/OMNIbus. In Network Performance Insight, V1.3, Tivoli Netcool/OMNIbus is an important part of the solution for monitoring the network threshold violations.

Related information:
- IBM Network Performance Insight on IBM Knowledge Center
- Apache Hadoop YARN
- IBM Netcool Operations Insight
- IBM Tivoli Network Manager IP Edition
IBM Community

Connect, learn, and share with professionals and product support technical experts who provide their perspectives and expertise.

Access the IBM Network Performance Insight community. Use IBM Community in the following ways:
- Become involved with transparent development, an ongoing, open engagement between other users and IBM developers of Tivoli products. You can access early designs, sprint demonstrations, product roadmaps, and prerelease code.
- Connect one-on-one with the experts to collaborate and network about Tivoli and the Network and Service Assurance community.
- Read blogs to benefit from the expertise and experience of others.
- Use wikis and forums to collaborate with the broader user community.

Network Performance Insight technical training

For Tivoli technical training information, see the following Network Performance Insight Training website at https://tnpmsupport.persistentsys.com/updated_trainings.

Support information

If you have a problem with your IBM Software, you want to resolve it quickly. IBM provides the following ways for you to obtain the support you need:

Online

IBM Support Assistant
The IBM Support Assistant is a free local software serviceability workbench that helps you resolve questions and problems with IBM Software products. The Support Assistant provides quick access to support-related information and serviceability tools for problem determination. To install the Support Assistant software, go to https://www.ibm.com/software/support/isa.

Troubleshooting Guide
For more information about resolving problems, see the problem determination information for this product.

Conventions used in this publication

Several conventions are used in this publication for special terms, actions, commands, and paths that are dependent on your operating system.

Typeface conventions

This publication uses the following typeface conventions:

Bold
- Lowercase commands and mixed case commands that are otherwise difficult to distinguish from surrounding text
- Interface controls (check boxes, push buttons, radio buttons, spin buttons, fields, folders, icons, list boxes, items inside list boxes,
multicolumn lists, containers, menu choices, menu names, tabs, property sheets), labels (such as Tip:, and Operating system considerations:)

- Keywords and parameters in text

*Italic*

- Citations (examples: titles of publications, diskettes, and CDs)
- Words defined in text (example: a nonswitched line is called a point-to-point line)
- Emphasis of words and letters (words as words example: "Use the word *that* to introduce a restrictive clause."); letters as letters example: "The LUN address must start with the letter L.")
- New terms in text (except in a definition list): a *view* is a frame in a workspace that contains data.
- Variables and values you must provide: ... where *myname* represents....

*Monospace*

- Examples and code examples
- File names, programming keywords, and other elements that are difficult to distinguish from surrounding text
- Message text and prompts addressed to the user
- Text that the user must type
- Values for arguments or command options

*Bold monospace*

- Command names, and names of macros and utilities that you can type as commands
- Environment variable names in text
- Keywords
- Parameter names in text: API structure parameters, command parameters and arguments, and configuration parameters
- Process names
- Registry variable names in text
- Script names
Chapter 1. Rapid SNMP device onboarding

Rapid SNMP device onboarding is a new feature that is introduced in Network Performance Insight, V1.3 to bring new devices such as routers, switches, and servers into a network environment smoothly and seamlessly. Accomplishing device onboarding accurately and consistently is critical to any enterprise.

An SNMP device is a device that is managed by using the Simple Network Management Protocol (SNMP). Rapid onboarding for new SNMP devices that are introduced in the network can be completed within 1 day. Within this period, the required discovery formulas, collection formulas, and metrics can be created and deployed for Network Performance Insight to start discovery and polling.

Rapid SNMP device onboarding solutions

Rapid SNMP device onboarding can be achieved with the help of Technology Pack Development Tool that is provided free of cost for Network Performance Insight customers.

You can do the following tasks in this solution:
- Create new discovery formulas.
- Create new collection formulas.
- Create new metrics.
- Package the custom Technology Pack content.
- Deploy the custom Technology Pack on Network Performance Insight system to start discovery and polling.

Components used in Rapid SNMP device onboarding:

Technology Pack Development Environment

Pack Tool
Device Discovery Tool
SNMP Binding Tool
SNMP Formula Tool
SNMP MIB Tool
Related concepts:

Appendix A, “SNMP formula language,” on page 19

Detailed information about the SNMP formula language that can be used to create custom collection formulas, discovery formulas, and metrics.
Chapter 2. Technology Pack Development Tool

Technology Pack Development Tool is a command-line tool to design and develop Technology Pack that can be deployed in Network Performance Insight system. The content can then be used to discover and poll the resources from newly on-boarded devices.

Prepackaged Technology Pack content

Network Performance Insight V1.3 has the prepackaged Technology Pack content that can be used to discover the resources to collect the device health metrics that are termed as Performance Metric OOTB Device Support metrics.

Devices are discovered by Tivoli Network Manager as first-level discovery and resources from these devices are discovered and polled by Network Performance Insight to collect the device health metrics.

With the help of the Technology Pack Development Tool, you can create the following artifacts that are required to develop a Technology Pack that can be later deployed in the Network Performance Insight system:

- Discovery formulas
  Discovery formulas are used while discovering and analyzing the network. Discovery formulas are used against managed nodes or hosts only and their resource types and are designed to return a list of results.
- Collection formulas
  Collection formulas are used to collect information about the status of various devices in the network during polling. These formulas are applied against a resource type or managed item in the host to produce a result (numeric) that is saved in the database.
- Metrics
  A metric is a single data value, identified by the resource and displayed in a report.

All the required MIB files must be bundles and packaged.

Important: Do not use this tool to customize the built-in Technology Pack content that is provided with Network Performance Insight.

Overview of the Technology Pack Development Tool tasks

- Provides the development environment to run the various command-line tools. The following command-line tools are available:
  - Discovery tool
  - SNMP binding tool
  - Formula tool
  - SNMP MIB tool
- Validates the syntax during the development phase.
- Controls the versioning.

Process flow for creating a technology pack by using the Technology Pack Development Tool.
Software specifications for Technology Pack Development Tool

- Currently, the tool is supported on RHEL 7.x systems.
- Requires Open JDK x86_64 V1.8.0.

Installing the tool

Installation of the Technology Pack Development Tool is simple and fast.

Procedure

1. Download the latest version of the Technology Pack Development Tool from Network Performance Insight IBM Community from here: [Rapid Device Onboarding - Technology Pack Development Tool](#)
2. Extract the media to a location of choice with the following command:
   
   ```
cd /<pack_dev_tool>
$ tar zxvf npi-tpdt-0.5.0.0.tar.gz
   ```

   Where `<pack_dev_tool>` is the directory where you have extracted the tool media.

   You can see the following directories and files:
   
   - **bin**
     - device-discovery-tool
     - pack-tool
     - snmp-binding-tool
     - snmp-formula-tool
     - snmp-mib-tool
   - **lib**

     Contains the library files that are required for the tool.
Preparing your environment

Use this information to understand the steps required to prepare the environment for your Technology Pack development.

About this task

Perform the following tasks:
1. “Creating a Technology Pack project”
   A workspace that contains all the Technology Pack content is required from where you can bundle the content later.
2. “Validating the MIB files” on page 6
   Copy and validate the required standard and prerequisite MIB files that are required for the Technology Pack content.

Creating a Technology Pack project

Create a workspace directory to keep the Technology Packs project files that are developed.

About this task

Use the pack_tool script to create a workspace for your project.

Procedure

1. Create a workspace directory for the workspace with the following command:
   ```bash
   cd /<pack_dev_tool>
   $ mkdir workspace
   ```
   Where, workspace is the project directory.
2. Optional: You can also create a softlink to the repository that you are using directly. For example, to convert your Git repository as the workspace directory to develop the packs, follow these commands:
   ```bash
   $ cd ~/git/
   $ git clone git@<repo_server.ibm.com>:<my_repo>/<my_file>.git
   $ cd ~/<pack_dev_tool>
   $ ln -s ~/git/<my_pack> workspace
   ```
   A new workspace by name `<my_pack>` is created that contains the new Technology Pack with the same name as the workspace.
3. Run the following command:
   ```bash
   cd /<pack_dev_tool>
   bin/pack-tool new <my_pack>
   ```
   New pro pack by name `<my_pack>` is created that contains the new Technology Pack with the same name as the workspace.

What to do next

Verify that the Technology Pack with the name `<my_pack>` is created as follows:
```bash
$ find workspace/</my_pack>/
```
Currently, you can ignore the inventory directory and its subdirectories.

**Note:** The pack.properties file contains the technology pack version. The default is 1.0.0.

**Related reference:** Appendix B, “Command line usage,” on page 33

Use this information to understand the usage of some command line options that are available in Technology Pack Development Tool.

### Validating the MIB files

A MIB file contains object definitions, which are organized to groups. Typically, the object definitions in a MIB file are organized in groups, such as System, Interface, or TCP. Use the snmp-mib-tool script to view a MIB file source and to validate the content in the MIB files.

**About this task**

The snmp-mib-tool script is available in /<pack_dev_tool>/bin directory. Follow these steps to validate the MIB files that are required for to develop your Technology Pack content.

**Procedure**

1. Copy all the required standard and prerequisite MIB files to the following directory:
   /workspace/<my_pack>/snmp/mibs

2. View the list of MIB files available in your Technology Pack content by using the following command:
   
   ```
   bin/snmp-mib-tool <my_pack> list
   ```

   You can see the list of MIB files that you have copied in step 1.

3. View a specific MIB file source by using the following command:
   
   ```
   bin/snmp-mib-tool <my_pack> show <MIB_name>
   ```

   For example:
   
   ```
   bin/snmp-mib-tool <my_pack> show CISCO-MEMORY-POOL-MIB
   ```

   You can see the content of the CISCO-MEMORY-POOL-MIB file.
Creating discovery formulas

You can create your own discovery formulas for the resources from the devices discovered by Tivoli Network Manager. Network Performance Insight can then discover those resources.

About this task

You can create discovery formulas for new devices and their resources based on their resource types. These discovery formulas can work independent of Tivoli Network Manager for device discovery.

Procedure

1. Copy all the standard and dependent MIB files that are required for the discovery formulas that you plan to create.

Arrange the vendor-specific MIB files by creating sub directories as follows:

```plaintext
    workspace/<my_pack>/snmp/mibs/
    workspace/<my_pack>/snmp/mibs/cisco
    workspace/<my_pack>/snmp/mibs/cisco/CISCO-ENHANCED-MEMPOOL-MIB
    workspace/<my_pack>/snmp/mibs/cisco/CISCO-ENTITY-SENSOR-MIB
    workspace/<my_pack>/snmp/mibs/cisco/CISCO-ENVMON-MIB
    workspace/<my_pack>/snmp/mibs/cisco/CISCO-MEMORY-POOL-MIB
    workspace/<my_pack>/snmp/mibs/cisco/CISCO-PROCESS-MIB
    workspace/<my_pack>/snmp/mibs/cisco/CISCO-QOS-PIB-MIB
    workspace/<my_pack>/snmp/mibs/cisco/CISCO-SMI
    workspace/<my_pack>/snmp/mibs/cisco/CISCO-TC
    workspace/<my_pack>/snmp/mibs/cisco/CISCO-ENTITY-FRU-CONTROL-MIB
    workspace/<my_pack>/snmp/mibs/cisco/INET-ADDRESS-MIB
    workspace/<my_pack>/snmp/mibs/cisco/HUAWEI-MIB
    workspace/<my_pack>/snmp/mibs/cisco/HUAWEI-ENTITY-EXTENT-MIB
    workspace/<my_pack>/snmp/mibs/cisco/HUAWEI-MIB
    workspace/<my_pack>/snmp/mibs/ietf
    workspace/<my_pack>/snmp/mibs/ietf/IANAifType-MIB
    workspace/<my_pack>/snmp/mibs/ietf/IF-MIB
    workspace/<my_pack>/snmp/mibs/ietf/SNMPv2-CONF
    workspace/<my_pack>/snmp/mibs/ietf/SNMPv2-MIB
    workspace/<my_pack>/snmp/mibs/ietf/SNMPv2-SMI
    workspace/<my_pack>/snmp/mibs/ietf/SNMPv2-TC
    workspace/<my_pack>/snmp/mibs/ietf/BRIDGE-MIB
    workspace/<my_pack>/snmp/mibs/ietf/ENTITY-MIB
    workspace/<my_pack>/snmp/mibs/ietf/HCNUM-TC
    workspace/<my_pack>/snmp/mibs/ietf/HOST-RESOURCES-MIB
    workspace/<my_pack>/snmp/mibs/ietf/IANA-ENTITY-MIB
    workspace/<my_pack>/snmp/mibs/ietf/P-BRIDGE-MIB
    workspace/<my_pack>/snmp/mibs/ietf/SNMP-FRAMEWORK-MIB
    workspace/<my_pack>/snmp/mibs/ietf/UUID-TC-MIB
    workspace/<my_pack>/snmp/mibs/ietf/RFC1155-SMI
```
2. Create the discovery formula file as follows: For example, to create a formula file by name, cisco-memorypool.discovery for a Cisco device:

   when resource.type = 'device' and resource.sysobjectid like '1.3.6.1.4.1.9'
   select index, ciscomemorypooltype as memoryPoolType, ciscomemorypoolname as memoryPoolName, ciscomemorypoolvalid as memoryPoolValid from CiscoMemoryPoolMib.ciscoMemoryPoolTable
   where ciscomemorypoolvalid = 1
   set type = 'memory'
   set vendor = 'Cisco'
   set name = context.host + '_MemoryPool<resource.index>'

3. Save the file in /workspace/<my_pack>/discovery/<vendor> folder. Arrange the discovery formulas in separate directories to avoid conflicts.

   Note: All the discovery formulas must have the file extension as .discovery.

4. Create all the required formulas to discover devices and their resources.

What to do next

- Validate the content to make sure there are no error in the formulas.

Related concepts:

"Writing custom discovery formulas" on page 20

Discovery formulas are used against devices and their resources during discovery and are designed to return a list of resources from a device.

Running the discovery

You can directly run the discovery formulas from the custom Technology Pack by using the device-discovery-tool script.

About this task

device-discovery-tool is available in the /<pack_dev_tool>/bin directory. The script runs all the discovery formulas that are available in the custom Technology Pack that is created.

The device-discovery-tool is useful in validating the formulas before the custom Technology Pack is deployed in Network Performance Insight system.

Procedure

Use the following command to run the discovery formulas from the custom Technology Pack that is created:

   $ bin/device-discovery-tool <my_pack> 'snmp://public@<device_IP>:161'
INFO: Discovered device is Resource [kind: device, name: 127.0.0.1, properties: Set(sysobjectid=1.3.6.1.4.1.2636, name=127.0.0.1, type=device)]
INFO: Found 1 supported definitions
INFO: Topology JSON =
{"resources": [{"type": "chassis", "name": "127.0.0.1_Chassis:<-1><1><0><0>", "properties": {
"index": "1.1.0.0", "jnxccontainersdescription": "chassis frame", "jnxsoperating12index": "0", "jnxccontentsserialno": "S/N TS4821", "jnxsoperating13index": "1", "jnxccontainerspartno": "710-013698", "jnxsoperatingdescription": "midplane", "vendor": "Juniper", "name": "127.0.0.1_Chassis:<-1><1><0><0>", "jnxcoperatingcontentsindex": "1", "jnxfillededescription": "chassis frame", "jnxfilledstate": "3", "jnxsoperatingl2index": "0", "type": "chassis", "jnxccontentssrevision": "REV 03", "jnxccontainerslevel": "0", "jnxccontainersview": "1", "jnxccontainersindex": "1", "jnxfilledcontainerindex": "1", "jnxccontentsscontainerindex": "1"}], "relations": []}
INFO: Successfully completed discovery, found 1 resource(s)

Note: You can copy the JSON output into any JSON viewer to see the content more effectively. For example:

```json
{ "resources": [
  {
    "type": "chassis",
    "name": "127.0.0.1_Chassis:<-1><1><0><0>",
    "properties": {
      "index": "1.1.0.0",
      "jnxccontainersdescription": "chassis frame",
      "jnxsoperating13index": "0",
      "jnxccontentsserialno": "S/N TS4821",
      "jnxsoperating13index": "1",
      "jnxccontainerspartno": "710-013698",
      "jnxsoperatingdescription": "midplane",
      "vendor": "Juniper",
      "name": "127.0.0.1_Chassis:<-1><1><0><0>",
      "jnxsoperatingcontentsindex": "1",
      "jnxfillededescription": "chassis frame",
      "jnxfilledstate": "3",
      "jnxsoperatingl2index": "0",
      "type": "chassis",
      "jnxccontentssrevision": "REV 03",
      "jnxccontainerslevel": "0",
      "jnxccontainersview": "1",
      "jnxccontainersindex": "1",
      "jnxfilledcontainerindex": "1",
      "jnxccontentsscontainerindex": "1"
    }
  }
]}
```

Related reference:
Appendix B, “Command line usage,” on page 33
Use this information to understand the usage of some command line options that are available in Technology Pack Development Tool.
Creating collection formulas

You can create your own collection formulas that you want Network Performance Insight to poll and collect the specific device performance metrics.

About this task

You can create custom collection formulas for any type of device and resources with in it.

Procedure

1. Copy all the standard and dependent MIB files that are required for the collection formulas that you plan to create and arrange them in vendor-specific directories to avoid conflicts.

   Note: Discovery and polling operations share the MIB files. So you must copy only those files that are not already available.

2. Generate the binding with all the required MIB files.

   For example, create a binding with RFC1213-MIB for a formula that required the MIB as follows:

   ```
   $ bin/snmp-binding-tool <my_pack> generate RFC1213-MIB
   Generating bindings for mib RFC1213-MIB...
   Generating bindings for mib RFC1213-MIB complete.
   ```

   Class files are generated and arranged in separate MIB folders for all the OBJECT_TYPES in the MIB file.

3. Create the collection formula file as follows: For example, to create a formula file by name, interface.inbound.octets.formula.

   ```
   interface.inbound.octets = value(RFC1213_MIB.ifInOctets)
   when resource.type == 'interface'
   ```

4. Save the file in workspace/<my_pack>/snmp/formulas/<vendor> folder. Arrange the discovery formulas in separate directories to avoid conflicts.

   Note: All the collection formulas must have the file extension as .formula. If the file has more than one formula in it, save it as .formulas.

5. Create all the required formulas to poll the devices and their resources.

What to do next

Validate the content to make sure there are no error in the formulas.

Related concepts:

Appendix A, “SNMP formula language,” on page 19

Detailed information about the SNMP formula language that can be used to create custom collection formulas, discovery formulas, and metrics.
Running the collection formulas

You can directly run the custom collection formulas from the custom Technology Pack by using the `snmp-formula-tool` script.

About this task

`snmp-formula-tool` is available in the `/<pack_dev_tool>/bin` directory. The script runs all the collection formulas that are available in the custom Technology Pack that is created.

The `snmp-formula-tool` is useful in validating and running the formulas before the custom Technology Pack is deployed in Network Performance Insight system.

Procedure

1. Use the following command to list all the collection formulas in the custom Technology Pack that is created:

   ```
   $ bin/snmp-formula-tool <my_pack> list
   ```

   Formulas
   - `interface.inbound.octets(ifInOctets: RFC1213_MIB.ifInOctets)`

2. Use the following command to run the collection formulas from the custom Technology Pack that is created:

   ```
   $ bin/snmp-formula-tool <my_pack> execute
   1/interface/vendor=cisco RFC1213_MIB.ifInOctets=1000,2000,3500,5500
   ```

   Executing with interface resource 1[vendor=cisco,agent=1:3030]...
   1:3030/1.3.6.1.2.1.2.2.1.10.0[300000]
   Calculated 1000.0 at timestamp 1535606392689 for interface.inbound.octets.
   Calculated 2000.0 at timestamp 1535606452690 for interface.inbound.octets.
   Calculated 3500.0 at timestamp 1535606512690 for interface.inbound.octets.
   Calculated 5500.0 at timestamp 1535606572690 for interface.inbound.octets.
   Execution complete in 1 seconds.

For example:

- If multiple properties are available in a formula, specify as:
  ```
  1/interface/vendor=cisco,name='1.1.1.1_if<1>'
  ```

- If multiple OIDs are available in a formula, specify as:
  ```
  CISCO_MEMORY_POOL_MIB.ciscoMemoryPoolFree=10,50
  CISCO_MEMORY_POOL_MIB.ciscoMemoryPoolLargestFree=20,40
  ```

Creating metrics

You can create your own specific device performance metric definitions that you want Network Performance Insight to poll and collect.

Procedure

1. Create the metric file as follows: For example, to create a metric file by name, `icmp-out-dest-unreached.metric`:

   ```
   name=icmp.out.dest.unreached
description="The number of ICMP destination unreachable message sent"
aliases=[]
properties=
   resource.type=device
   ```
2. Save the file in `/<my_project>/<my_pack>/metrics/<vendor>` folder. Arrange the
discovery formulas in vendor-specific directories to avoid conflicts.

**Note:** All the metrics must have the file extension as `.metric`.

**What to do next**

Validate the content to make sure there are no error in the formulas.

---

### Validating the pack content

As a best practice, test that the syntax of the formulas and metrics to make sure
that they work correctly. Use the validate option in the pack-tool script.

**About this task**

You can perform this step intermittently during the Technology Pack development
or at the end before packaging.

**Procedure**

Run the following command:

- Validating the discovery formulas:

```
bins/pack-tool validate <my_pack>
```

```
Validating <my_pack> pack...
Validating inventory models...
Validating discovery...
  - Validating cisco-memorypool.discovery...
Validating metrics...
Validating snmp formulas...
Validating <my_pack> pack complete.
```

- Validating the collection formulas:

```
$ bin/pack-tool validate <my_pack>
```

```
Validating <my_pack> pack...
Validating inventory models...
Validating discovery...
  - Validating cisco-memorypool.discovery...
Validating metrics...
Validating snmp formulas...
  - Validating interface.inbound.octets.formula...
Validating <my_pack> pack complete.
```

- Validating the metrics:

```
$ bin/pack-tool validate <my_pack>
```

Rapid SNMP device onboarding
Validating `<my_pack>` pack...
Validating inventory models...

Validating discovery...
  - Validating cisco-memorypool.discovery...

Validating metrics...
  - Validating icmp-out-dest-unreached.metric...
    - Error in `/home/sherpa/sdk/workspace/pack-demo/metrics/icmp-out-dest-unreached.metric`: 2:
      Expecting a value but got:
      wrong token: 'newline' (JSON does not allow unescaped new line in quoted strings, use a backslash escape)
      (if you intended 'newline' (JSON does not allow unescaped new line in quoted strings, use a backslash escape) to
      be part of a key or string value, try enclosing the key or value in double quotes, or you may be able to
      rename the file .properties rather than .conf)

Validating snmp formulas...
  - Validating interface.inbound.octets.formula...

Validating `<my_pack>` pack complete.

**Note:** The validation error is due to missing double quotation marks in the metric definition. You can fix the errors during validation and rerun.

---

**Packaging the Technology Pack bundle**

You can generate the JAR file package of the custom Technology Pack with the content that you developed by using the Technology Pack Development Tool.

**About this task**

Use the build option in `pack-tool` script to package the Technology Pack that can be deployed on the Network Performance Insight system.

**Procedure**

Run the build command as follows:

```
$ bin/pack-tool build <my_pack>
```
Building pack <my_pack> [version 1.0.0] ...
- Adding file snmp/mibs/cisco/CISCO-ENHANCED-MEMPOOL-MIB...
- Adding file snmp/mibs/cisco/CISCO-ENTITY-SENSOR-MIB...
- Adding file snmp/mibs/cisco/CISCO-ENVMON-MIB...
- Adding file snmp/mibs/cisco/CISCO-MEMORY-POOL-MIB...
- Adding file snmp/mibs/cisco/CISCO-PROCESS-MIB...
- Adding file snmp/mibs/cisco/CISCO-QOS-PIB-MIB...
- Adding file snmp/mibs/cisco/CISCO-SMI...
- Adding file snmp/mibs/cisco/CISCO-TC...
- Adding file snmp/mibs/cisco/CISCO-ENTITY-FRU-CONTROL-MIB...
- Adding file snmp/mibs/cisco/INET-ADDRESS-MIB...
- Adding file snmp/mibs/huawei/HUAWEI-ENTITY-EXTENT-MIB...
- Adding file snmp/mibs/huawei/HUAWEI-MIB...
- Adding file snmp/mibs/ietf/IANAifType-MIB...
- Adding file snmp/mibs/ietf/IF-MIB...
- Adding file snmp/mibs/ietf/SNMPv2-CONF...
- Adding file snmp/mibs/ietf/SNMPv2-MIB...
- Adding file snmp/mibs/ietf/SNMPv2-TC...
- Adding file snmp/mibs/ietf/BRIDGE-MIB...
- Adding file snmp/mibs/ietf/ENTITY-MIB...
- Adding file snmp/mibs/ietf/HCNUM-TC...
- Adding file snmp/mibs/ietf/HOST-RESOURCES-MIB...
- Adding file snmp/mibs/ietf/IANA-ENTITY-MIB...
- Adding file snmp/mibs/ietf/P-BRIDGE-MIB...
- Adding file snmp/mibs/ietf/SNMP-FRAMEWORK-MIB...
- Adding file snmp/mibs/ietf/UUID-TC-MIB...
- Adding file snmp/mibs/ietf/RFCl355-SMI...
- Adding file snmp/mibs/ietf/RFCl358-MIB...
- Adding file snmp/mibs/ietf/RFCl212...
- Adding file snmp/mibs/ietf/RFCl213-MIB...
- Adding file snmp/mibs/juniper/JUNIPER-MIB...
- Adding file snmp/mibs/juniper/JUNIPER-SMI...
- Adding file snmp/mibs/juniper/Juniper-MIBs...
- Adding file snmp/mibs/juniper/Juniper-System-MIB...
- Adding file snmp/mibs/juniper/Juniper-TC...
- Adding file snmp/mibs/juniper/Juniper-UNI-SMI...
- Adding file formulas/rfc1213/interface.inbound.octets.formula...
- Adding file discovery/cisco/cisco-memorypool.discovery...
- Adding file pack.properties...
Building pack <my_pack> complete.

All the custom content that is available in the <my_pack> directory is bundled. The <my_pack>-1.0.0.jar file is available in the following directory:

$ cd <pack_dev_tool>
$ find build/
  build/
  build/<my_pack>

### Deploying the custom Technology Pack

Deploy the custom Technology Pack in Network Performance Insight system to start using the content to discover and poll the devices and resources. The collected metrics are stored in the database and can be rendered on Network Performance Insight Dashboards.

#### About this task

After the Technology Pack is bundled with the custom content, you can deploy and run discovery and polling from Network Performance Insight Dashboards as usual.
**Procedure**

1. Copy the custom Technology Pack bundle to Network Performance Insight system to a location of your choice. For example, `<DIST_DIR>`.

2. Install the Technology Pack by using the following commands:
   
   ```bash
   cd /opt/IBM/basecamp/basecamp-installer-tools/pack-installer
   ./pack-install.sh <DIST_DIR>/<my_pack>-1.0.0.jar
   ```

   The `<my_pack>-1.0.0.jar` file that is available in your `<DIST_DIR>` is extracted and the content is distributed to the following directories:

   **Discovery formulas**
   `/opt/IBM/npi/npi-itnm-collector/discovery/`

   The discovery directory has all the collection formulas and their related files.

   **Formulas**
   `/opt/IBM/npi/npi-itnm-collector/discovery/content`

   Contains all the discovery formula files that are arranged in separate vendor-specific directories from the custom Technology Pack.

   **MIB files**
   `/opt/IBM/npi/npi-itnm-collector/discovery/content/mibs`

   Contains all the MIB files that are arranged in separate directories from the custom Technology Pack.

   **Collection formulas**
   `/opt/IBM/npi/npi-formula/content/`

   The content directory has all the collection formulas and their related files.

   **Bindings**
   `/opt/IBM/npi/npi-formula/content/bindings`

   Contains all the MIB object class files that are created by the snmp-binding-tool script.

   **Formulas**
   `/opt/IBM/npi/npi-formula/content/formulas`

   Contains all the collection formula files from the custom Technology Pack.

   **MIB files**
   `/opt/IBM/npi/npi-formula/content/mibs`

   Contains all the MIB files that are arranged in separate directories from the custom Technology Pack.

   **Metrics**
   `/opt/IBM/basecamp/basecamp-timeseries/content/metrics`

   The metrics directory has all the metric files.
Troubleshooting Technology Pack Development Tool

You can use this troubleshooting and support information to troubleshoot problems with Technology Pack Development Tool.

For troubleshooting the usage of Technology Pack Development Tool for creating custom technology pack, see the related information.

Incorrect Java™ location can cause Discovery failure issues

Symptoms
If you do not set your JAVA_HOME environment variable to point to correct Java location, you might see the following error message while using Technology Pack Development Tool to create your custom technology pack content:

```java
ERROR: Discovery failed (null)
java.lang.J9VMInternals.ensureError(J9VMInternals.java:141)
java.lang.J9VMInternals.recordInitializationFailure(J9VMInternals.java:130)
persistent.itoa.discovery.function.Utils.vendor(Functions.scala)
```

Resolving the problem
To avoid this issue, make sure you point to the correct Java home as follows:

```bash
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.181-3.b13.el7_5.x86_64/jre
```

Note: Technology Pack Development Tool requires Open JDK x86_64 V1.8.0.

Errors while running the collection formulas

Symptoms
When you run the collection formulas in your custom technology pack content, you might see the ClassNotFoundException error.

Resolving the problem
To avoid this issue, make sure you follow these guidelines while creating the collection formulas:

For formula definition, the MIB name that is used in the definition must be consistent with the MIB file name inside /opt/IBM/npi/npi-formula/content/bindings folders. The formula refers to the folders that are created during bindings.

If your formula contains ETHERNET-PB-SERVICES-MIB module, it cannot find that particular MIB folder in /opt/IBM/npi/npi-formula/content/bindings folder. The correct name of the MIB file in /opt/IBM/npi/npi-formula/content/bindings folder is ETHERNET_PB_SERVICES_MIB with underscores instead of dashes. Make sure your formula uses the same notation as the MIB file name in the /opt/IBM/npi-npi-formula/content/bindings folder.

Error during discovery if sysObjectID OID value is not set in an SNMP simulator agent

Symptoms
If you are running discovery against an SNMP simulator, make sure that the agent contains all the required fields such as MIB system scalar table with sysObjectID OID value. Failing which, you might see the following error message:

```scala
ERROR: Discovery failed (head of empty stream)
scala.collection.immutable.Stream$Empty$.head(Stream.scala:1168)
```

Rapid SNMP device onboarding
You might not see this issue in real devices.

**Resolving the problem**

To resolve this issue, add the system table to the simulated agent and try to run the discovery again.
Appendix A. SNMP formula language

Detailed information about the SNMP formula language that can be used to create custom collection formulas, discovery formulas, and metrics.

Technology Pack Development Tool can be used to craft these formulas and metrics.

All the custom content can be validated, bundled, and installed in Network Performance Insight system according to your requirements.

Language basics

Specific formula language that is supported in Network Performance Insight. This formula language can be used by application engineers who need to create the custom content for new devices in network.

Formulas

A formula is a calculation that is performed against an SNMP data. It is written in a proprietary language. You can use a formula to customize the information collected by Network Performance Insight.

Two types of formulas:

Discovery formulas

Discovery formulas are used against managed nodes or hosts and their resources. These formulas are designed to return a list of resources from a device. You can write a new discovery formula to include a device that does not have an in-built support in Network Performance Insight.

Collection formulas

It is a collection of OID functions, and standard mathematical operations that are applied to a resource type to perform a computation.

Collection formulas are used to collect metrics about the status of the devices in your network during polling. These formulas are applied against a resource or managed item in the host to produce a result (numeric). The result is saved in the timeseries database. A resource type can be a physical or logical object such as port, interface, virtual circuit, or DLCI.

Network Performance Insight has many predefined collection formulas in Technology Packs that are available in the media. However, you can write additional formulas for a customized collection process.

Metrics

Collected data values. From the database perspective, a metric is a single data value that can be identified by a resource. From the reporting perspective, a metric is a data value shown in a report. It is a set of database metrics, pre-aggregated over time by using a statistical function like average or maximum. A report metric can be aggregated across a set of resources and their types.
MIB files

Management Information Base (MIB) is a collection of information that is organized hierarchically. MIBs provide definitions for the properties of a managed object within a device.

Writing custom discovery formulas

Discovery formulas are used against devices and their resources during discovery and are designed to return a list of resources from a device.

Discovery process

The discovery process consists of several steps.

• Tivoli Network Manager performs the first-level discovery and returns the following information:
  – Devices
  – SNMP credentials for the devices
• During the second-level discovery, Network Performance Insight runs discovery to get a list of resources from the discovered devices based on the applicable discovery formulas.
• If a new device is added to Tivoli Network Manager system, the ad hoc discovery is run. Resources are discovered on the device and updated in the database.

Note: Network Performance Insight comes with predefined vendor-specific discovery formulas and their required MIB files. All the discovery files and their MIB files are available in /opt/IBM/npi/npi-itnm-collector/discovery/content directory.

You can use the Technology Pack Development Tool to create, validate, package your own discovery formulas.

Discovery formula structure

All the discovery formula files must have the file extension as .discovery.

when clause

Specifies the resource types that are appropriate for executing a specific discovery formula. The components of when clause are as follows:

• resource.type
  resource.type = 'device'

Resource type device is discovered by Tivoli Network Manager in first-level discovery. Currently, only the device resource type is supported by Tivoli Network Manager and resources from the discovered devices are discovered by Network Performance Insight in second-level discovery.

Note: The value for resource.type must be in single quotation marks.

• resource.sysobjectid

You can narrow the discovery to a specific vendor type by specifying the sysobjectid. The sysobjectid represents the type of device and can also indicate the model number in a dotted decimal format. For example:
Table 1. Vendor and model identification numbers

<table>
<thead>
<tr>
<th>Vendor</th>
<th>OID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco</td>
<td>1.3.6.1.4.1.9</td>
</tr>
<tr>
<td>Huawei</td>
<td>1.3.6.1.4.1.201</td>
</tr>
<tr>
<td>Juniper</td>
<td>1.3.6.1.4.1.2636</td>
</tr>
<tr>
<td>Juniper Networks/Unisphere</td>
<td>1.3.6.1.4.1.4874</td>
</tr>
</tbody>
</table>

For more information about the vendor and model identification numbers, see [IANA-registered Private Enterprise Numbers](#).

All the discovery files that have the same sysobjectid as that is returned from system-objectid.discovery are run to obtain the resources from those devices.

- **name.<IP_address>**
  - It is the IP address of an interface or device.

**Example**

```sql
when resource.type = 'device' and resource.sysobjectid like '1.3.6.1.4.1.9'
```

**Note:** The value for sysobjectid must be in single quotation marks.

**select statement**

In Network Performance Insight discovery, MIB tables are represented as relational database tables. Hence you can write the discovery definitions in SQL.

Selects data from a table or multiple tables. An SQL statement that retrieves resource properties from the tables that are available in the associated vendor-specific MIB file. The `select` statement has the following components:

- **property names**
  - A list of property names and their aliases.

- **from table expression**
  - Selects data from a table or multiple tables. In this case, it is `<MIB_file_name.MIB_table_name>`. For example:

```sql
when resource.type = 'device' and resource.sysobjectid like '1.3.6.1.4.1.9'
select index, ciscomemorypooptype as memoryPoolType, ciscomemorypoolname as memoryPoolName, ciscomemorypoolvalid as memoryPoolValid from CiscoMemoryPoolMib.ciscoMemoryPoolTable where ciscomemorypoolvalid = 1
set type = 'memory'
set vendor = 'Cisco'
set name = context.host + '_MemoryPool<' + resource.index + '>
```

- `CiscoMemoryPoolMib` is the code representation of the MIB file `CISCO-MEMORY-POOL-MIB`.
  - You can get these details by running the bin/snmp-mib-tool pack-demo list command from the Technology Pack Development Tool installation directory.

- `ciscoMemoryPoolTable` is MIB table from `CISCO-MEMORY-POOL-MIB` file.
  - You can get these details by running the bin/snmp-mib-tool pack-demo show `CISCO-MEMORY-POOL-MIB` command from the Technology Pack Development Tool installation directory.

- **join table expression**
  - Joins tables to combine rows from two or more tables, based on a related column between them. The different types of joins in SQL are as follows:

  - **inner join**
Returns records that have matching values in both tables.

- **left join**
  Returns all records from the left table, and the matched records from the right table. It is also known as outer join.

- **right join**
  Returns all records from the right table, and the matched records from the left table. It is also known as outer join.

- **full join**
  Returns all records when there is a match in either left or right table.

**Note:**
- The index column is a special column that discovery adds automatically to the MIB tables. It contains the MIB index details.
- If the select statement contains joins with multiple tables and the column name is not unique for a table, then you must add the MIB table name to qualify the column name. For example, ciscoMemoryPoolTable.index.
- In V1.3, there is no support for scalar MIBs.

**where condition**
The where condition is used to extract only those records that fulfill a specified condition.

**Operators**
You can use all the SQL supported operators in the select statements. Typically, the following operators are used in the predefined discovery formulas.

- **like**
  The like operator is used in a where clause to search for a specified pattern in a column.

- **IN**
  The IN operator allows you to specify multiple values in a where clause.

- **ON**
  ON represents one or more JOIN conditions by which to match the records from one table to another.

**Example**
```sql
select hwEntityStateTable.index, hwEntityAdminStatus, hwEntityOperStatus, hwEntityStandbyStatus, hwEntityTemperature, entPhysicalTable.index, entPhysicalIndex, entPhysicalDescr, entPhysicalVendorType, entPhysicalName, entPhysicalClass from HuaweiEntityExtentMib.hwEntityStateTable left join EntityMib.entPhysicalTable on hwEntityStateTable.index = entPhysicalTable.index where entPhysicalClass = 6
```

**Note:** You can use complex nested select statements to create your discovery formulas by using the Technology Pack Development Tool.

**set definitions**
Discovery formulas provide an additional way to add resource properties. The set definition can contain a literal string, or combination of discovered properties, or special placeholder called context. For example:

```plaintext
set type = 'resource_type'
set vendor = 'vendor'
set name = context.host + '_resource_name:<' + resource.index + '>
```

The expressions in the set definitions are as follows:
- context
context is the special place holder that is used in the discovery formulas. It contains the following two components:

- host
  The context.host string returns the host name of the device. In this case, it is the IP address of the device that is returned from Tivoli Network Manager.

- port
  The context.port string returns the SNMP port number of the device.

The expression, set name = context.host + '_Huawei_PowerSupply:<' + resource.index + 'r'> results in the name as:
10.55.239.56_Huawei_PowerSupply:<1835017>

Example

```plaintext
set type = 'card'
set vendor = 'Huawei'
set name = context.host + '_Huawei_PowerSupply:<' + resource.index + 'r>'
```

unset definitions

Unset removes or deletes the specified set values.

Example

```plaintext
unset jnxfilleddescr
unset jnxoperatingcontentsindex
unset jnxoperatingllindex
unset jnxoperatingl2index
unset jnxoperatingl3index
```

Note: The set and unset definitions support JavaScript expressions.

Required properties

- type
  Resource type to which the resource that is discovered belongs to. The resource types can be configured from Resource Types page in System Configuration on Dashboard Application Services Hub portal.
  In Cisco devices, it can be memory, environment, temperature, fan, cpu, and others.
  For more information, see Configuring resource types in Installing and Configuring IBM Network Performance Insight.

- vendor
  Vendor device to which the discovery formula is applicable.

- name
  The resource on the device for which the discovery is to be done.

- index
  Index number of the resource.

Examples

Some vendor-specific discovery formula examples:

**Cisco**
`cisco-powersupply.discovery`

When resource.type = 'device' and resource.sysobjectid like '1.3.6.1.4.1.9'
select index, ciscoenvmonsuppliesstatusdescr as envPowerSupplyStatusDescription,
ciscoenvmonsuppliesstatus as envPowerSupplyState, ciscoenvmonsuppliessource as
envPowerSupplySource from CiscoEnvmonMib.ciscoEnvMonSupplyStatusTable
where ciscoenvmonsuppliesstatus!=5
Related information:

- **Validating the MIB files** on page 6:

A MIB file contains object definitions, which are organized to groups. Typically, the object definitions in a MIB file are organized in groups, such as System, Interface, or TCP. Use the `snmp-mib-tool` script to view a MIB file source and to validate the content in the MIB files.

**Related tasks:**

- "Validating the MIB files" on page 6
Writing custom collection formulas

A collection formula is applied against a resource to produce a result that is always a number. The result is then saved to the timeseries database. The collection formulas are executed regularly based on the polling interval and applied on a larger scale than discovery formulas. These collection formulas must be as simple as possible.

About this task

Explains how to create custom collection formulas.

Formula syntax

Use this information to understand the syntax for writing the collection formulas easily. A collection formula is a calculation performed against raw SNMP data. It is a combination of functions, conditions, and standard mathematical operations.

Collection formula structure

All collection formulas must have the extension .formula. If a formula file has more than one formula, save it with the extension .formulas.

Name of the formula

Typically, formula names follow a dot notation that includes the following structure:

<resource_group>.<metric_name>.<units>.formula

For example:

- Environment.Temperature.Level.Celcius.formula
- Network.Outbound.Broadcast.pps.formulas

function

Built-in functions that can be used in the collection formula language that can perform a calculation on the metrics and return a value. See "Common functions" on page 27.

when clause

Filter condition to pick from the specified values or to compare the expression to determine the value. It contains a set of key value pairs.

when resource.type == 'card' && resource.vendor == 'Huawei'

Note: All string values must be in single quotation marks.

resource type

Typically, the when condition must specify the resource type from which to poll that is followed by other conditions. For example:

when resource.type == 'interface' && number(resource.ifSpeed)>429496729

- CPU.Utilization.Percent.formula

  CPU.Utilization.Percent = value(CISCO_PROCESS_MIB.cpmCPUTotal5minRev) / when resource.type == 'cpu' && resource.vendor == 'Cisco'

- Network.Inbound.Broadcast.pps.formulas

  Network.Inbound.Broadcast.pps = positive
  (delta(IF_MIB.ifHCInBroadcastPkts))/duration(IF_MIB.ifHCInBroadcastPkts)*100
  when resource.type == 'interface' && number(resource.ifSpeed)>4294967295
Network.Inbound.Broadcast.pps = positive
(delta(IF_MIB.ifInBroadcastPkts))/duration(IF_MIB.ifInBroadcastPkts)*100
when resource.type == 'interface' && number(resource.ifSpeed)<4294967295

Operators
The operators supported by the formula language.

Mathematical conventions
The formula language uses standard priority rules between operators. Expressions are evaluated from left to right, respecting parentheses and mathematical precedence rules. You can use parentheses to alter priority or to increase readability.

Binary Operators
A list of the mathematical operators supported by the formula language.

Table 2. Mathematical Operators

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>Addition</td>
<td>When used between numbers, this is mathematical addition. When used between strings, this is the concatenation operator.</td>
</tr>
<tr>
<td>-</td>
<td>Subtraction or negation</td>
<td>This symbol is used either as a subtraction operator, or in front of an expression to negate the value of the expression.</td>
</tr>
<tr>
<td>*</td>
<td>Multiplication</td>
<td></td>
</tr>
<tr>
<td>/</td>
<td>Division</td>
<td></td>
</tr>
<tr>
<td>%</td>
<td>Modulo</td>
<td>Returns the remainder. For example, 155 % 10 -&gt; 5.</td>
</tr>
<tr>
<td>b&amp;</td>
<td>Binary AND</td>
<td>Converts numbers to binary and performs an AND operation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example: 252 b&amp; 63 = 60</td>
</tr>
<tr>
<td></td>
<td></td>
<td>11111100 b&amp; 00111111 = 00111100</td>
</tr>
<tr>
<td>b</td>
<td></td>
<td>Binary OR</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example: 127 b</td>
</tr>
<tr>
<td></td>
<td></td>
<td>01111111 b</td>
</tr>
</tbody>
</table>

The result of a Boolean operation is an integer value: 0 for False, 1 for True. The value is considered to be False if it is 0, or True for any value other than 0.
Boolean Operators

Table 3. Boolean Operators

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;</td>
<td>Greater than</td>
<td></td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater than or equal</td>
<td></td>
</tr>
<tr>
<td>&lt;</td>
<td>Less than</td>
<td></td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less than or equal</td>
<td></td>
</tr>
<tr>
<td>==</td>
<td>Equal</td>
<td></td>
</tr>
<tr>
<td>!=</td>
<td>Not equal</td>
<td></td>
</tr>
<tr>
<td>Like</td>
<td>Like</td>
<td>This is a string comparison tool. The right argument is a string that can contain wildcard characters. The wildcard characters are as follows:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>* -- Replaces 0 or more characters</td>
</tr>
<tr>
<td></td>
<td></td>
<td>? -- Replaces one character.</td>
</tr>
<tr>
<td>&amp;&amp;</td>
<td>Logical AND</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Common functions

The common functions supported in collection formulas.

delta

Purpose

Returns the difference between the current value of binding $x$ and its previous value. If `delta()` of a MIB object with counter syntax is negative, it’s not considered a problem.

This function is used exclusively in polling formulas.

Syntax

```
delta (x)
```

Example

```
positive(delta(CISCO_ENHANCED_MEMPOOL_MIB.cempMemPoolFreeHit))
when resource.type == 'memory' && resource.vendor == 'Cisco' && resource.name.indexOf('Enhanced_Memory_Pool') > 0
```

value

Purpose

Returns the value of a binding $x$.

Syntax

```
value (x)
```
**x** | The binding whose value you want to find.
---|---

**Examples**
  `value(CISCO_ENVMON_MIB.ciscoEnvMonSupplyState) when resource.type == 'environment' && resource.vendor == 'Cisco' && resource.name.indexOf('PowerSupply') > 0`

**values**

**Purpose**
Returns the values of a binding `x` that is stored in current state.

**Syntax**
`values(x)`

**x** | The binding whose values from its stored current state that you want to find.
---|---

**Example**
`positive(values(CISCO_ENHANCED_MEMPOOL_MIB.cempMemPoolFreeHit)(0)) when resource.type == 'memory' && resource.vendor == 'Cisco' && resource.name.indexOf('Enhanced_Memory_Pool') > 0`

**duration**

**Purpose**
Returns the difference of the current value collection time and the previous value collection time of a binding `x`.

**Syntax**
`duration (x)`

**x** | The binding whose value you want to find.
---|---

**Example**
`positive(delta(RFC1213_MIB.ipInReceives))/duration(RFC1213_MIB.ipInReceives)*100 when resource.type == 'device'`

**time**

**Purpose**
Returns the collection time of the current value of a binding `x`.

**Syntax**
`time (x)`

**Example**
`positive(time(CISCO_ENHANCED_MEMPOOL_MIB.cempMemPoolFreeHit)) when resource.type == 'memory' && resource.vendor == 'Cisco' && resource.name.indexOf('Enhanced_Memory_Pool') > 0`

**times**

**Purpose**
Returns all the collection times of all the values of a binding `x` that is stored in the current state.

**Syntax**
`times(x)`
Example
positive(times(CISCO_ENHANCED_MEMPOOL_MIB.cempMemPoolFreeHit)(0))
when resource.type == 'memory' && resource.vendor == 'Cisco' &&
resource.name.indexOf('Enhanced_Memory_Pool') > 0

currentTime
Purpose
Returns the collection time of the current value of a binding x.
Syntax
currentTime (x)
Example
currentTime(CISCO_ENHANCED_MEMPOOL_MIB.cempMemPoolFreeHit)
when resource.type == 'memory' && resource.vendor == 'Cisco' &&
resource.name.indexOf('Enhanced_Memory_Pool') > 0

previousTime
Purpose
Returns the collection time of the previous value of a binding.
Syntax
previousTime (x)
Example
previousTime(CISCO_ENHANCED_MEMPOOL_MIB.cempMemPoolFreeHit)
when resource.type == 'memory' && resource.vendor == 'Cisco' &&
resource.name.indexOf('Enhanced_Memory_Pool') > 0

rate
Purpose
Returns the rate of change between the current value and previous
value of a binding.
Syntax
rate (x)
Example
rate(CISCO_ENHANCED_MEMPOOL_MIB.cempMemPoolFreeHit)
when resource.type == 'memory' && resource.vendor == 'Cisco' &&
resource.name.indexOf('Enhanced_Memory_Pool') > 0

currentValue
Purpose
Returns the value of a binding x.
Syntax
currentValue (x)
Example
currentValue(CISCO_ENHANCED_MEMPOOL_MIB.cempMemPoolFreeHit)
when resource.type == 'memory' && resource.vendor == 'Cisco' &&
resource.name.indexOf('Enhanced_Memory_Pool') > 0

previousValue
Purpose
Returns the previous value of a binding.
Syntax
previousValue (x)
Writing custom metrics

A metric is a single collected data value that is identified by its generic name, timestamp, and value. From the reporting perspective, a metric is a data value that is shown in a report. Metric files contain metadata such as metric name, description, and alias. For every collection formula, a corresponding metric file must be created.

About this task

Explains how to create custom metrics and use them in creating custom collection formulas. All performance metrics in Network Performance Insight are stored in timeseries database. These metrics can be displayed in real-time on Timeseries Data dashboard.

Metric structure

Typical structure of a metric that is used in a collection formula. A metric has these components. Metrics structure follows HOCON format, which is derived from JSON.

name

It is the normalized metric name for the same metric across all supported vendors. For example, the built-in metrics that are available in Technology Packs has the following structure:

<resource_group>-<metric_name>-<units>.metric

Note: All metrics must have an extension .metric.

description

Provide a simple description of what SNMP data that the metric returns. The data type for description is string.

Note: Description must be in double quotation marks.

units

Provide an applicable unit for the metric based on the information that it represents and its result. For example:

- Number (count,type)
- Percent
- Bytes
- Volt
- Celsius
- Milliseconds
aliases

An array of comma-separated vendor-specific formula names by which the metric can be identified. It must match the formula name that can be mapped to the metric name.

properties

This field can be left blank with open and closed braces.

Some examples of metrics are as follows:

- Environment-Voltage-Level-Volts.metric
  name=Environment.Voltage.Level.Volts
  description="The voltage of entity"
  units=volt
  aliases=[CiscoDevice Voltage Level,Huawei Voltage Level]
  properties={}

- CPU-Utilization-Percent.metric
  name=CPU.Utilization.Percent
  description="The percentage of CPU utilization"
  units=Percent
  aliases=[cpuBusy,JuniperERX CPU Utilization,JuniperChassis CPU Utilization avg last 5min,Huawei CPU Utilization, CiscoDevice System CPU Utilization CPM]
  properties={}

- IP-Fragment-Failure-Count.metric
  name=IP.Fragment.Failure.Count
  description="The number of IP datagrams that is discarded because they needed to be fragmented at this entity but might not be."
  units=Number
  aliases=[RFCMIBII Fragmentation Failures]
  properties={}

Related information:

Using HOCON, the JSON Superset
Appendix B. Command line usage

Use this information to understand the usage of some command line options that are available in Technology Pack Development Tool.

**pack_tool**

pack-tool command can be run as root user. The script is available in /<pack_dev_tool>/bin directory. Where, <pack_dev_tool> is the directory where Technology Pack Development Tool is extracted.

**Usage** Run the following command as root user:

```bash
$ bin/pack-tool
```

```
- pack-tool [command] [options]

commands:
  new <pack> - creates a new pack project
  build <pack> - builds the specified pack project otherwise all projects
  clean <pack> - cleans the specified pack project otherwise all projects
  validate <pack> - validates the specified pack project otherwise all projects
  help - displays help
```

**Parameters**

- **new**
  Creates a project workspace to contain the Technology Packs that are developed.

- **build**
  Builds and packages the Technology Pack after the development is completed.

- **clean**
  Cleans the specified Technology Pack project. If a project name is not specified, it cleans all projects.

- **validate**
  Validates that content within the Technology Packs after the development is complete.

- **help**
  Provides the usage of the pack-tool command.

**device-discovery-tool**

The device-discovery-tool command is used to run the discovery formulas that are created for your custom Technology Pack. The script is available in /<pack_dev_tool>/bin directory. Where, <pack_dev_tool> is the directory where Technology Pack Development Tool is extracted.

**Usage** Run the following command as root user:

```bash
$ bin/device-discovery-tool
```

```
- device-discovery-tool [options]
```

© Copyright IBM Corp. 2015, 2018
discovery-device-tool [pack] snmp://[credential]@[host]:[port]?[options]

credential:
  version 1/2c - [read community string]:[write community string]
  version 3 - [user name]:[authentication]:[encryption]:[context]

authentication: 
  md5(password) - MD5 password
  sha1(password) - SHA1 password
  none - No authentication

encryption:
  des(password) - DES key
  aes(password) - AES key
  aes128(password) - AES 128-bit key
  aes192(password) - AES 192-bit key
  aes256(password) - AES 256-bit key
  none - No encryption

options:
  version - SNMP Version (2,3). Default is auto-detect
  timeout - SNMP timeout in milisecond. Default is 5 seconds
  retries - Number of retries before giving up. Default is 3 times

Parameters

pack
  Name of the Technology Pack.

credential
  Credentials for two the SNMP versions:
  • version 1/2c - [read community string]:[write community string]
    - read community string: name of the SNMP read community.
    - write community string: name of the SNMP write community.
  • version 3 - [user name]:[authentication]:[encryption]:[context]
    - user name: the user name to be used for this SNMP V3 community name.
    - authentication: the password to be used for authentication (MD5 or SHA1) for this SNMP V3 element.
    - encryption: the type of encryption for the privacy password. The following types of encryption are available:
      - 3-DES
      - AES 128
      - AES 192
      - AES 256
    - context: the context name to be used for this SNMP V3 community name.

Host
  Host name of the device associated with this SNMP configuration.

port
  The port associated with this SNMP configuration. By default, it is 161.

options
  Provide the additional information as follows:
  • version - the SNMP Version (2 or 3) associated with this SNMP configuration.
  • timeout - the length of time (in seconds) to wait for a response from a request. Default is 5 seconds.
• retries - the number of times that a request is tried again if a request failure. Default is 3 times
Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user’s responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those
websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
US

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

The client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM’s future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM’s suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:
This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (your company name) (year).
Portions of this code are derived from IBM Corp. Sample Programs.
© Copyright IBM Corp. _enter the year or years_.

Trademark

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information" at [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

Adobe, Acrobat, PostScript and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

ITIL is a registered trademark, and a registered community trademark of The Minister for the Cabinet Office, and is registered in the U.S. Patent and Trademark Office.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.
Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ulitrion logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

**Terms and conditions for product documentation**

Permissions for the use of these publications are granted subject to the following terms and conditions.

**Applicability**

These terms and conditions are in addition to any terms of use for the IBM website.

**Personal use**

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

**Commercial use**

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

**Rights**

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.