NetBrain®

IE

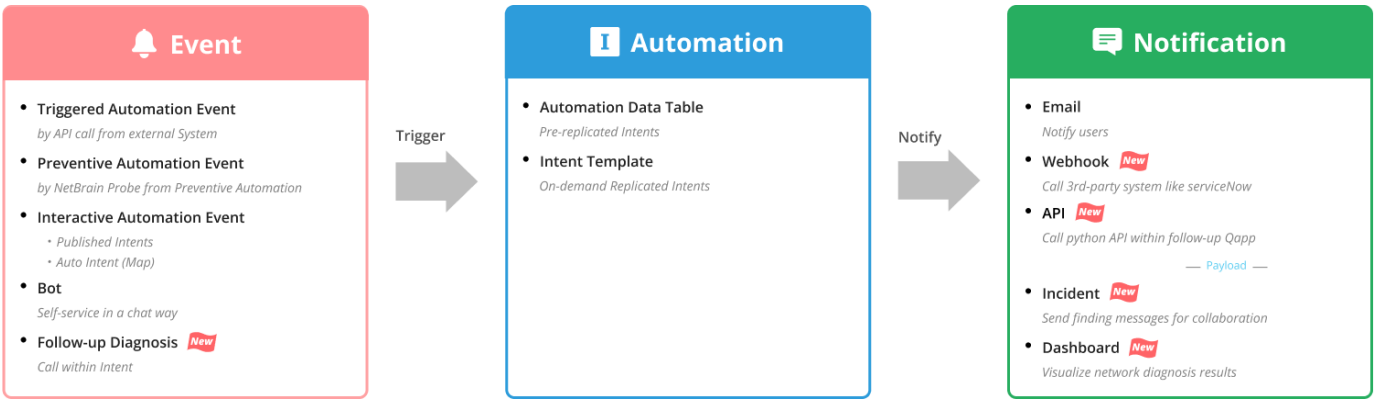
NetBrain® R11.1

Release Note

Table of Contents

1 Key New Features and Enhancements

NetBrain release R11.1 greatly enhances the Intent-based **Problem Diagnosis Automation System (PDAS),** which automates the Diagnosis of repetitive problems and enforces preventive measures across the entire network. Especially, R11.1 strengthens NetBrain as a scalable platform to build, update and execute thousands to millions of intents across your whole network with the following new features and enhancements:



- Automation Data Table (ADT) New

  An **Automation Data Table** (ADT) is an extended global data table for managing the critical network assets and network intents associated with those network assets. ADT is a database for intents, supporting intent creation and replication, and can be used as the base of the PDAS system.

- ADT-based PDAS Automation Flow

  ADT serves as a data reservoir in NetBrain and can be used across functions to provide data for the proper operations, including TAF (Triggered Automation Framework), PAF (Preventive Automation Framework), Auto Intent, Chat Bot, and follow-up Intent Diagnosis:

  - TAF Enhancements: R11.1 supports ADT intents in TAF Triggered Diagnosis and assigning ADT values to Macro Variables in the intent template.
  - PAF Enhancements: R11.1 adds a new probe, **Looking Glass Probe** New**,** and enhances PAF functions to ensure the critical assets defined within ADTs function properly by triggering the corresponding intents through the PAF probes. Users can also batch-create probes with a new feature **Auto Probe** New.

- Auto Intent Enhancements: R11.1 incorporates ADT assets into Auto Intent to support interactive troubleshooting and create the map's embedded intents and path intents.
- Path Intent Enhancements: R11.1 supports batch path intent replication from ADT, using ADT data sources as macro variables of the path intent, and path intent replication from the path browser and Auto Intent.

- Network Intent Enhancements

  R11.1 adds many new features to enhance the no-code programmability of Network Intent (NI), including:

  - Major enhancement of Intent's no-code programmability, such as follow-up diagnosis, intent dataview, and intent map.
  - Use Intent to drive programable notifications to 3rd party solutions.
  - Support intent across API-based networks such as SDN or Cloud.
  - On-Demand replication of Intent.

- Automation Bot New

  Use an Automation Bot to build an interactive, multi-step automation chatbot and execute multiple intent-based automations to solve real-world challenges without using NetBrain IE system UI. A chatbot provides self-service to a large audience without knowing the NetBrain system.

- Report and Dashboard New

Report and Dashboard provides a single interface to organize, analyze, and share data from different NetBrain automation assets, such as ADT Intent results, and a visual display of multiple automation results from PDAs.

- Other Enhancements

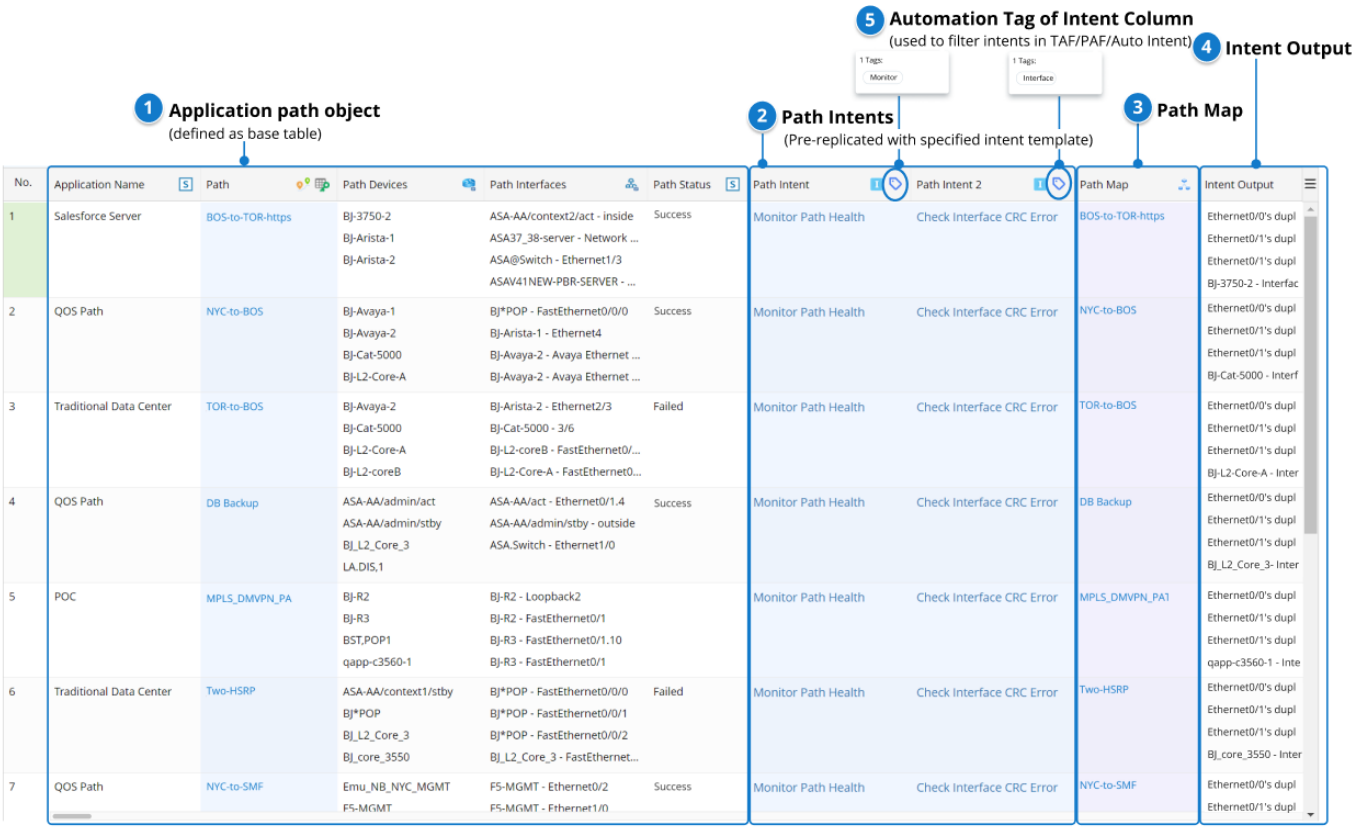
  R11.1 enhances the foundations and many features, such as **Circuit Break** New, to solve the benchmark performance issues due to the large data table, improving the data accuracy and open topology.

2 Automation Data Table (ADT)

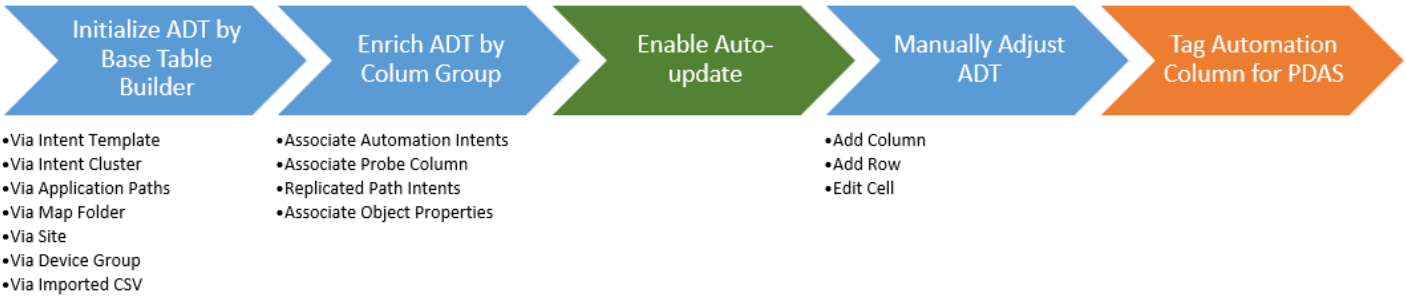
An **Automation Data Table (ADT)** is an extended global data table for managing the critical network assets and network intents associated with those network assets. ADT provides users with two capabilities:

- Use ADT to build, organize and share the critical assets, such as critical Application and Path, WAN Link, Device Failover, Subnet, and Route, then associate intents and diagnosis results for troubleshooting network problems.
- ADT is a database for intents, supporting intent creation and replication. The system provides methods (**Base Table** and **Column Group**) in Automation Data Table Builder to ensure that users can build the basic table data and create and bind the intent for each automation asset.

The following diagram illustrates what an ADT looks like:



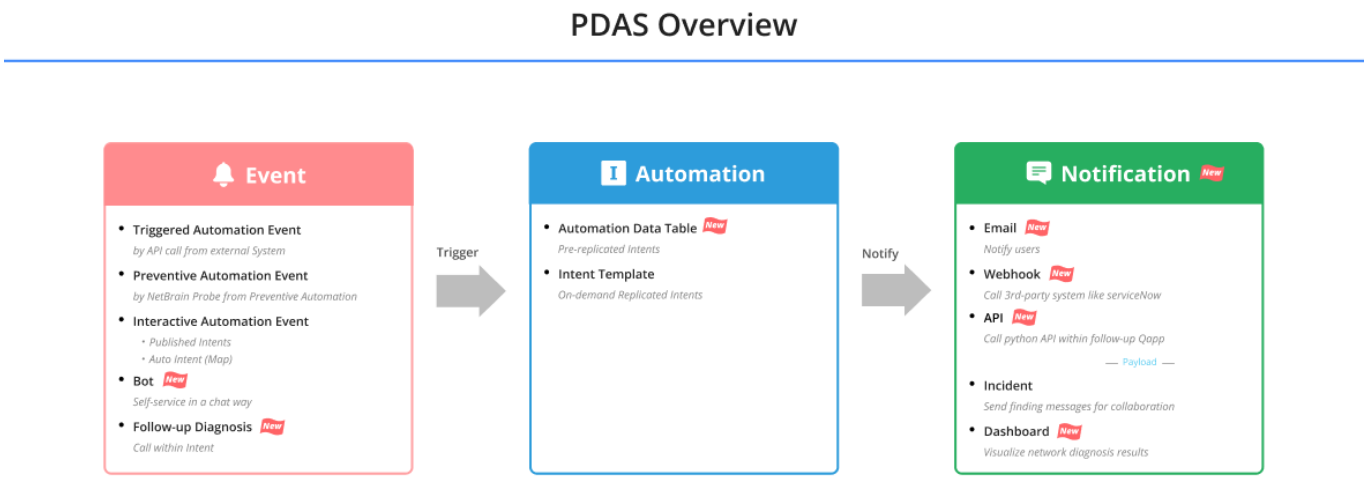
The following diagram illustrates how to create Automation Data Table:



After ADTs are created, users can use ADT for NetBrain Problem Diagnosis Automation System (PDAS), called ADT-based PDAS, which provides the following key values:

- Trigger ADT intents and deliver ADT maps in Triggered Automation Framework (TAF)
- Next-gen ADT-based Preventive Automation (PAF)
- Intent replication for application path (Path Intent)
- Troubleshoot network problems in NetBrain Bot based on ADT

The following diagram illustrates how ADT works in the PDAS system:

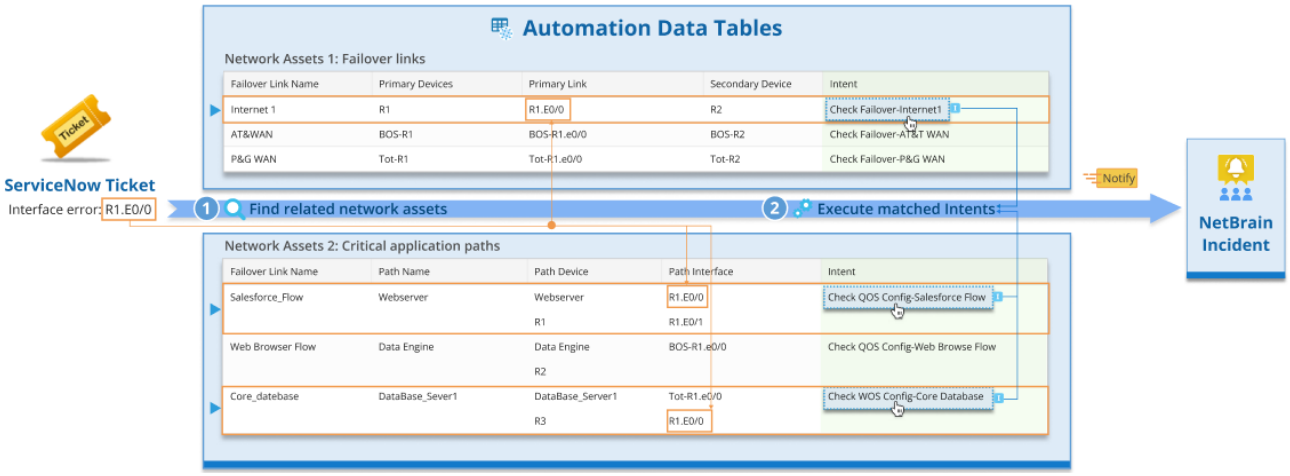


The ADT-based PDAS differs from the previous PDAS in the following two respects:

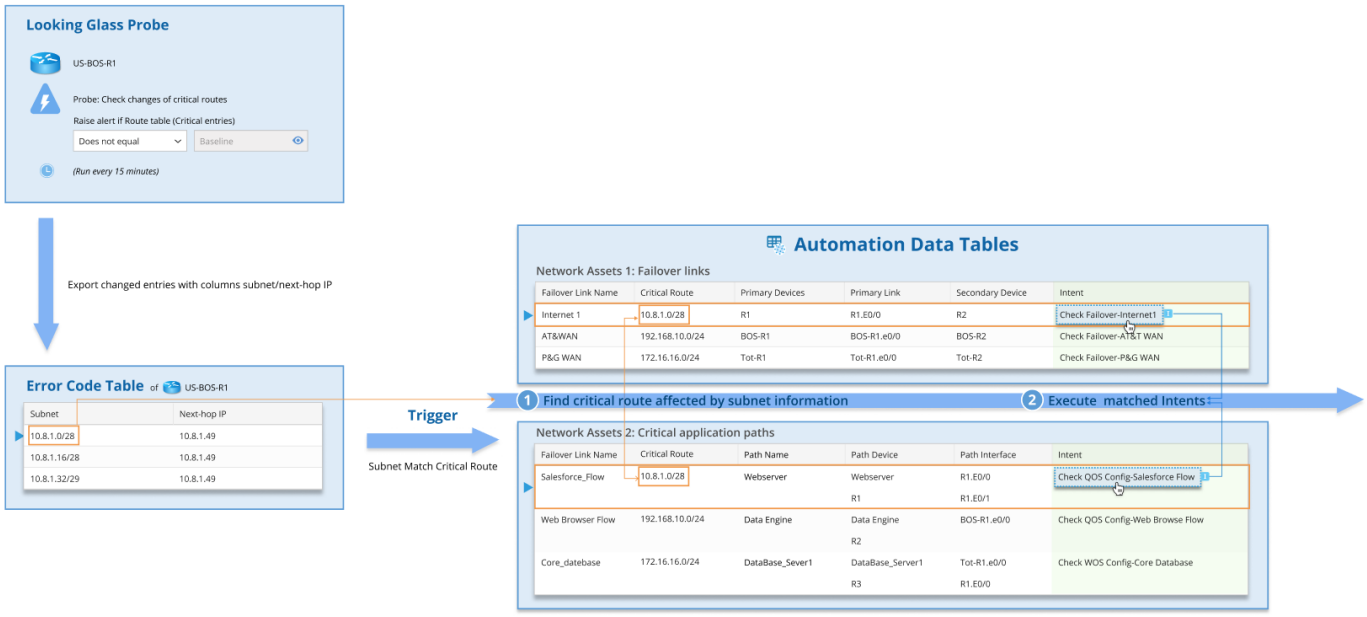
- The intents in ADT are pre-replicated, and the intents cloned from the intent template are on-demand replicated.
- The intents in ADT are referenced to automation assets.

As the ADT-based PDAS system shows its main value in TAF and PAF, we provide two diagrams to illustrate how ADT-based PDAS works in TAF and PAF, respectively.

The following diagram illustrates how ADT-based PDAS works in TAF:

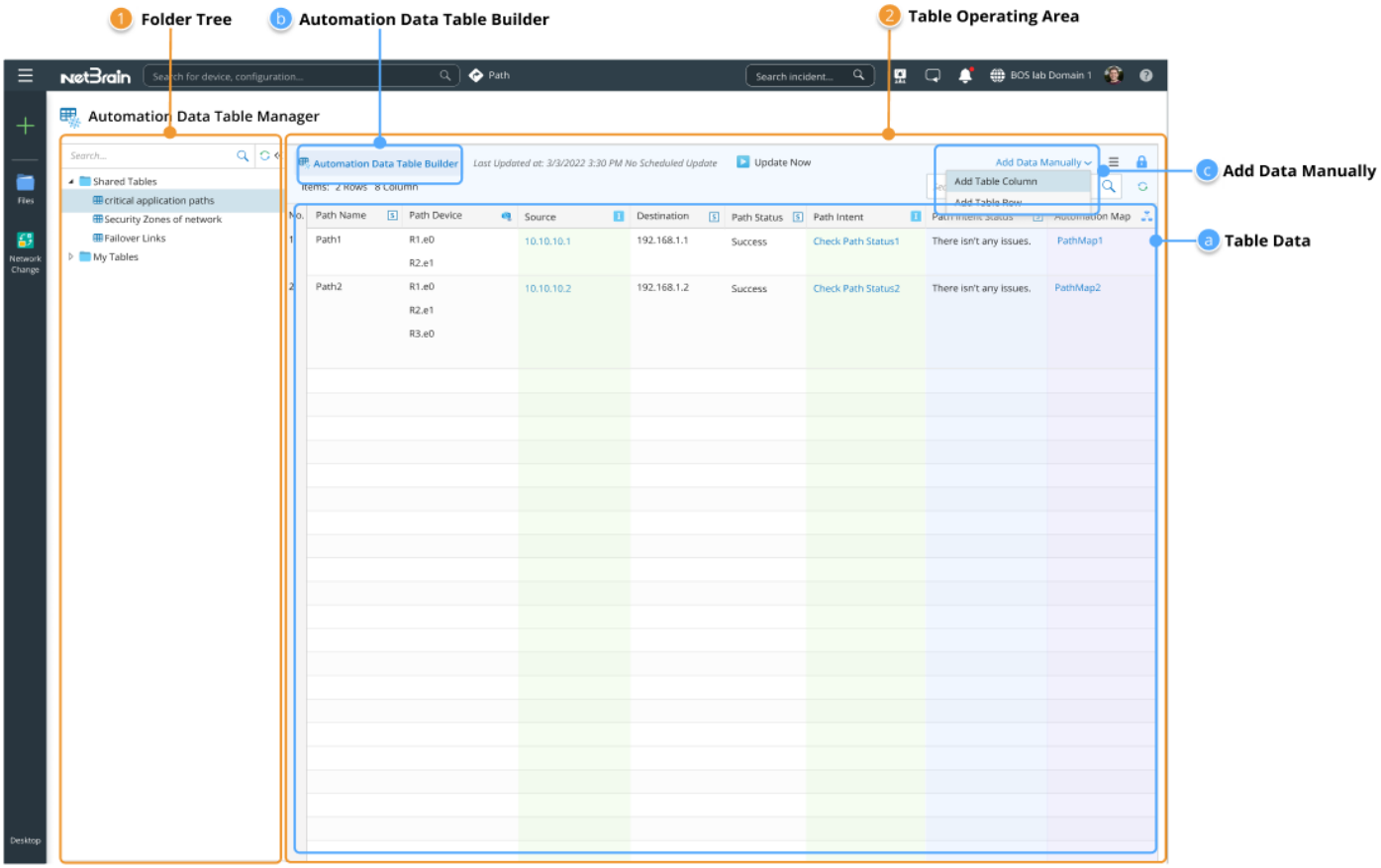


The following diagram illustrates how ADT-based PDAS works in PAF:

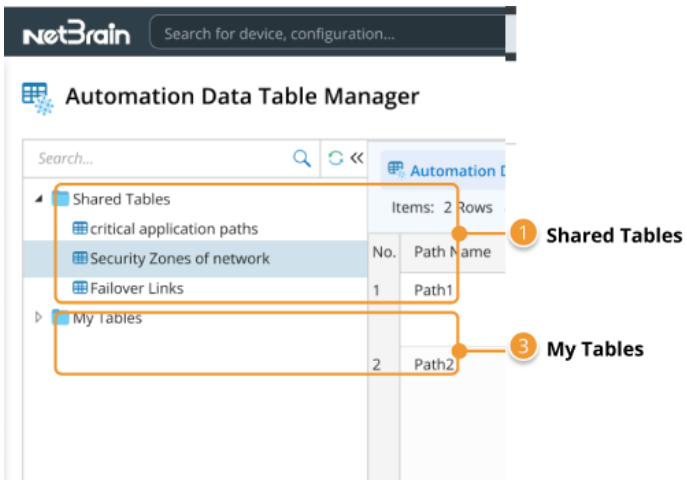


2.1 Components of Automation Data Table

Automation Data Table Manager is a centralized management interface of ADTs where end users can create, edit, and delete ADT. It consists of the following components:



1. A **Folder Tree** structure to organize shared tables and my tables.



2. **Table Operating Area** to edit, clear and export data.

- **Table Data** for displaying the ADT data.
- **Automation Data Table Builder** for creating ADT.
- Add Data Manually menu for manually editing data.

2.1.1 Table Data

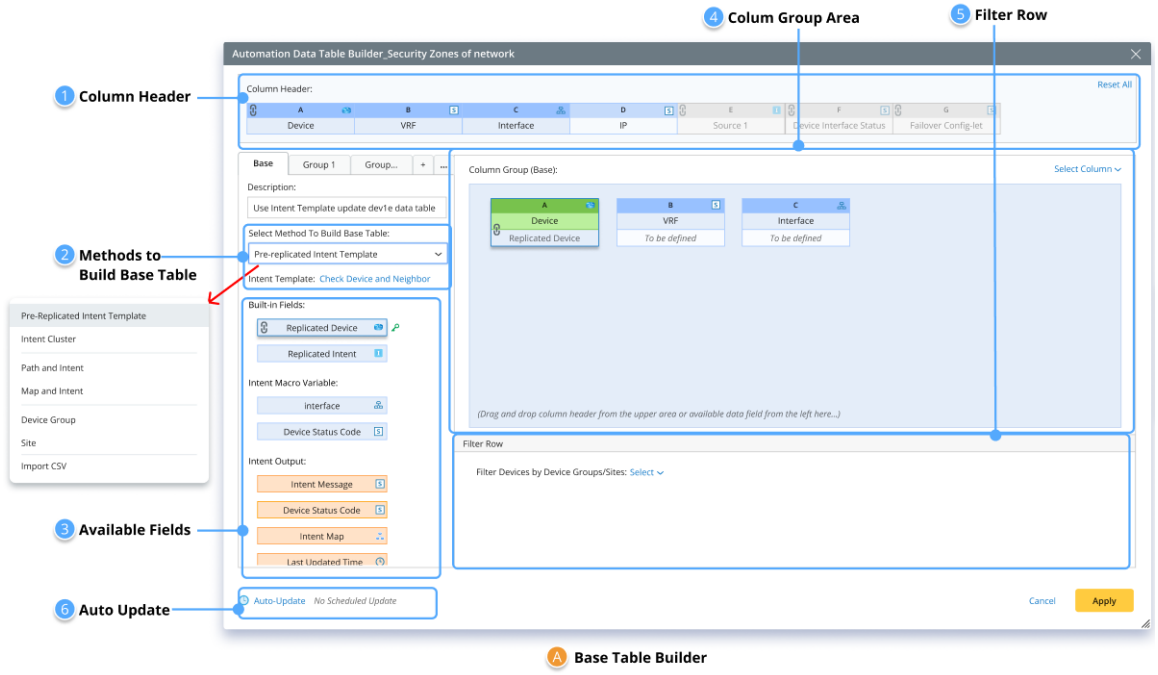
Table data, the final table data of an ADT, is shown in the diagram below and includes:

- **Table Header**: Display all the defined columns in the current ADT. Several operations on the column can be performed here (**Edit/Delete/Set as Table Key**). The basic column information, such as the display name, column name and data type, is displayed as a tip window.
- **ADT Base Table**: The base table contains basic source data for building automation assets in ADT. A base table is a way of building basic data. We provide 7 methods to create and populate the base table.
- **ADT Column Group**: The columns added to ADT by Automation Data Table Builder to enrich the ADT data of automation columns. Column groups depend on base table data and use the base table data as inputs. There are 6 methods to add a column group.

2.1.2 Automation Data Table Builder

An integrated UI with capabilities for dynamically creating table data and dynamic columns. **Base Table Builder** and **Column Group Builder** are included here to ensure users can build the table data and create/bind the intent for each automation asset.

- **Base Table Builder:** Creating a basic network table.



The system supports 7 methods to populate automation (intent/probe/map) columns, facilitating enriching/updating the ADT contents:

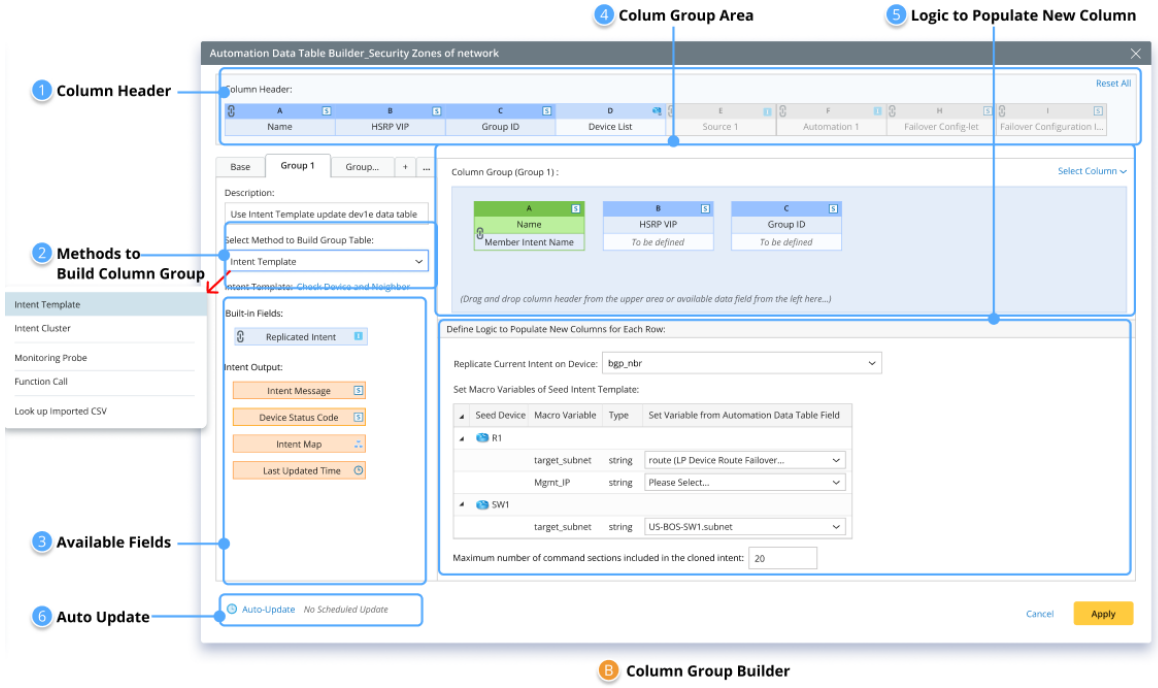
Method	Populated Columns	Populated Rows
Pre-replicated Intent Template	• Replicated Intent • Replicated Device	1 device and 1 replicated intent per row. Furthermore, the network attributes corresponding to the device can be extended by macro variables.

	• Macro Variable (Dynamic) • Intent Output *	
Intent Cluster	• Member Intent Name • Member Intent Devices • Signature Variables (Dynamic) • Intent Output *	1 member intent per row. Furthermore, the attributes corresponding to the automation asset can be added as row data.
Application Path	• Application Name • Path • Path Name • Path Devices • Path hops (device interfaces) • Path Map • Source • Destination • Path Status	1 path per row. Furthermore, the path properties can be added as row data.
Map Folder	• Map • Map Name • Map Devices • Map Device Interfaces • Map Intent • Intent Output *	1 map and 1 map intent per row. Furthermore, the map can be added as row data.
Site	• Site Name • Devices in Site • Site Map	1 site device and 1 site map intent per row. Furthermore, the site properties can be added as row data.

	• Site Map Intent • Intent Output *	
Devices of Device Group	• Device Name • Device Group Name • Device Properties (Dynamic)	1 device per row. Furthermore, the device group name and properties can be added as row data.
Imported CSV	CSV Columns (Dynamic)	1 CSV row per row.

Note: The Intent Output column includes the following data: Intent Status Code, Intent Message, Intent Status Code, Device Status Code, Intent Devices, Intent Map, Intent CLI Commands, and Last Execution Time.

- **Column Group Builder:** Creating dynamic automation columns or property columns for each automation asset.



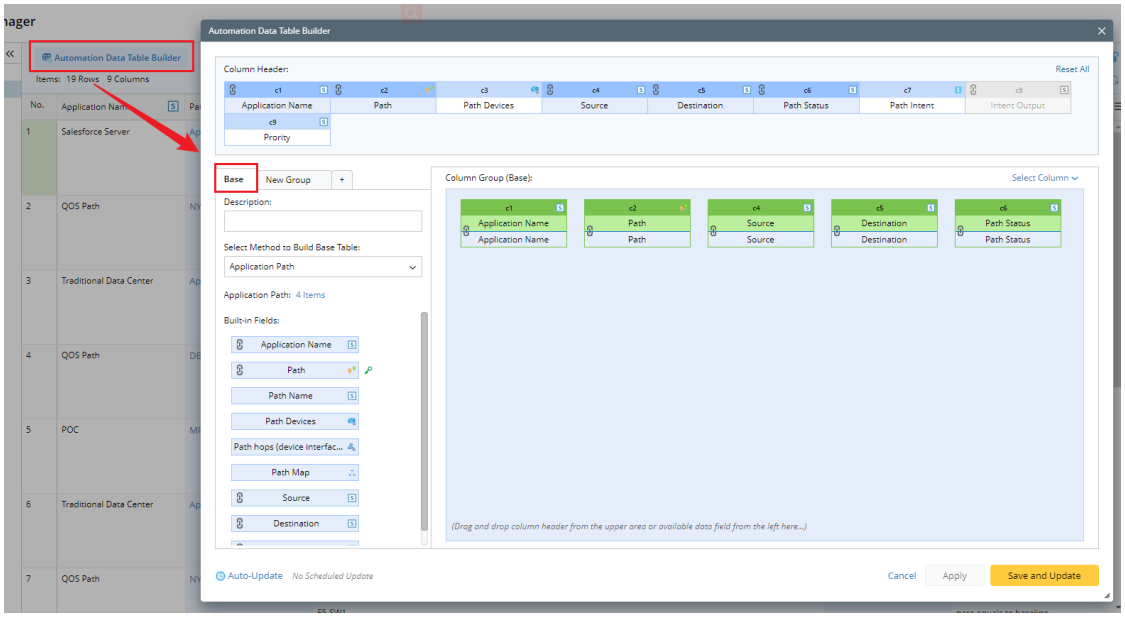
The system provides 5 column group builders:

Method	Populate Column Data	Logic
Intent Template	Cloned Intent for each Row	Clone intent for each valid row based on the defined device column. Add an intent column, then add the intent outputs to additional columns.
Intent Cluster	Member Intent	Select a NIC, then match and filter a member intent for each valid row and add the intent outputs to additional columns.
Monitoring Probe	Probe	Filter probes by the probe name using the current ADT data.
Imported CSV	CSV Column	Use selected key columns to match the rows in CSV files and append specified columns from CSV files to the ADT for new columns in the matched rows.
Function Call	Column's properties Covert source column by function	Specify a column with the type of device and use device properties and functions (e.g., transferring an IP address to a hostname) to fill in additional columns.

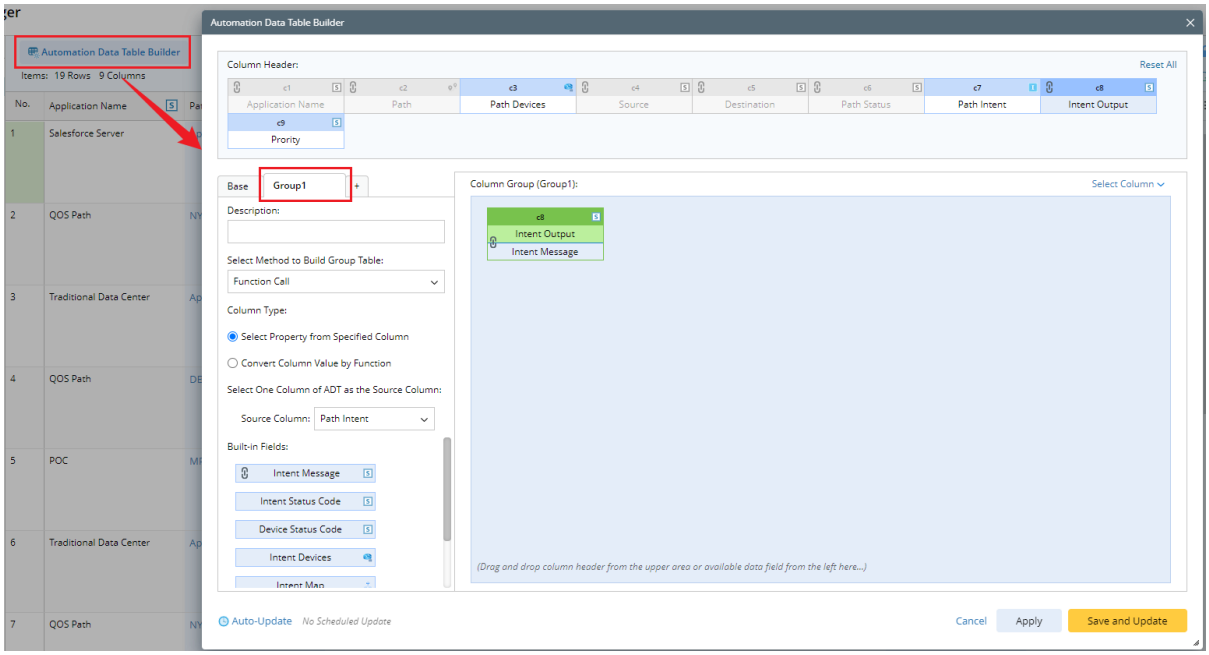
2.2 Create ADT Automatically

ADT table can be created automatically with the following steps:

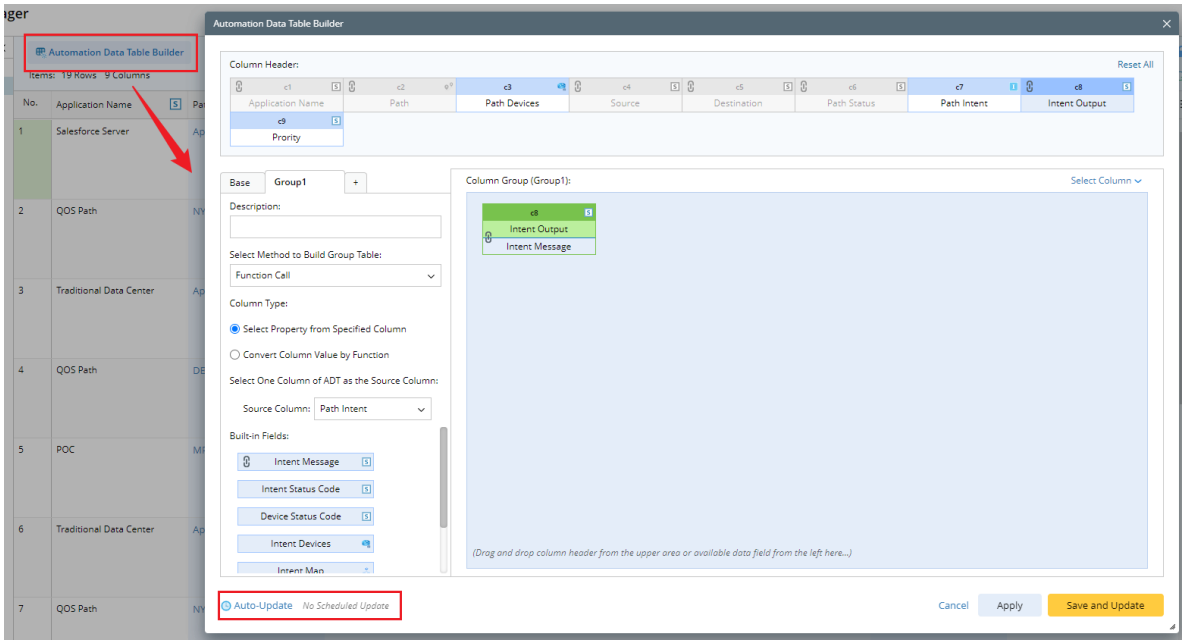
1. Go to **Automation Data Table Builder>Base** to build a base table containing data fields to include automation asset data to be used.



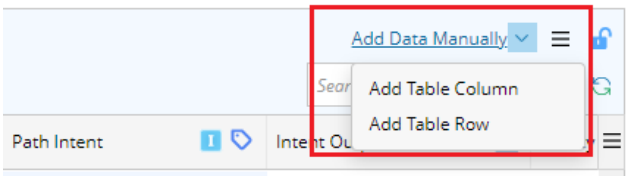
2. Go to **Automation Data Table Builder > Column Group tab** to create column groups to enrich the base table data.



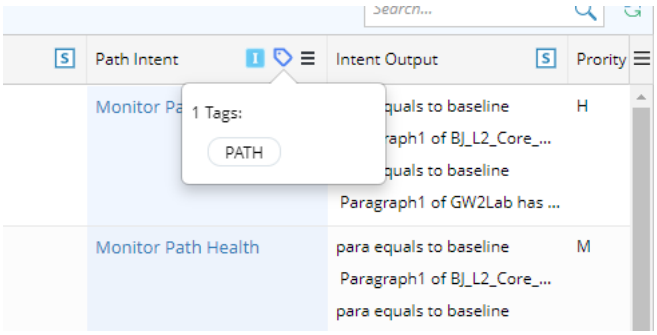
3. Go to **Automation Data Table Builder** (**Schedule Update)** to schedule updating ADT data periodically.



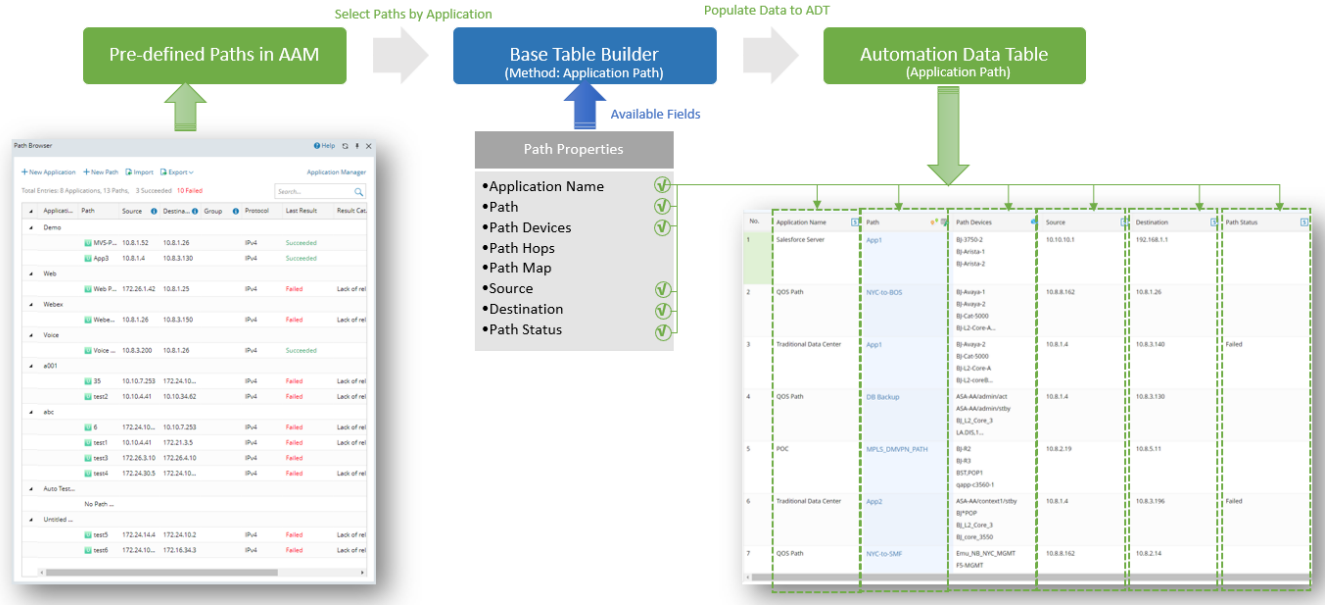
4. Go to **Automation Data Table Manager** (**Add Data Manually**), and manually add or adjust the table data.



5. Go to **Automation Data Table Manager**, then create tags for the automation column/map column from the **Tag Current Column** of the drop-down menu. The tags can filter the intents used for TAF, PAF, Bot, Auto Intent or Follow-up Intent.

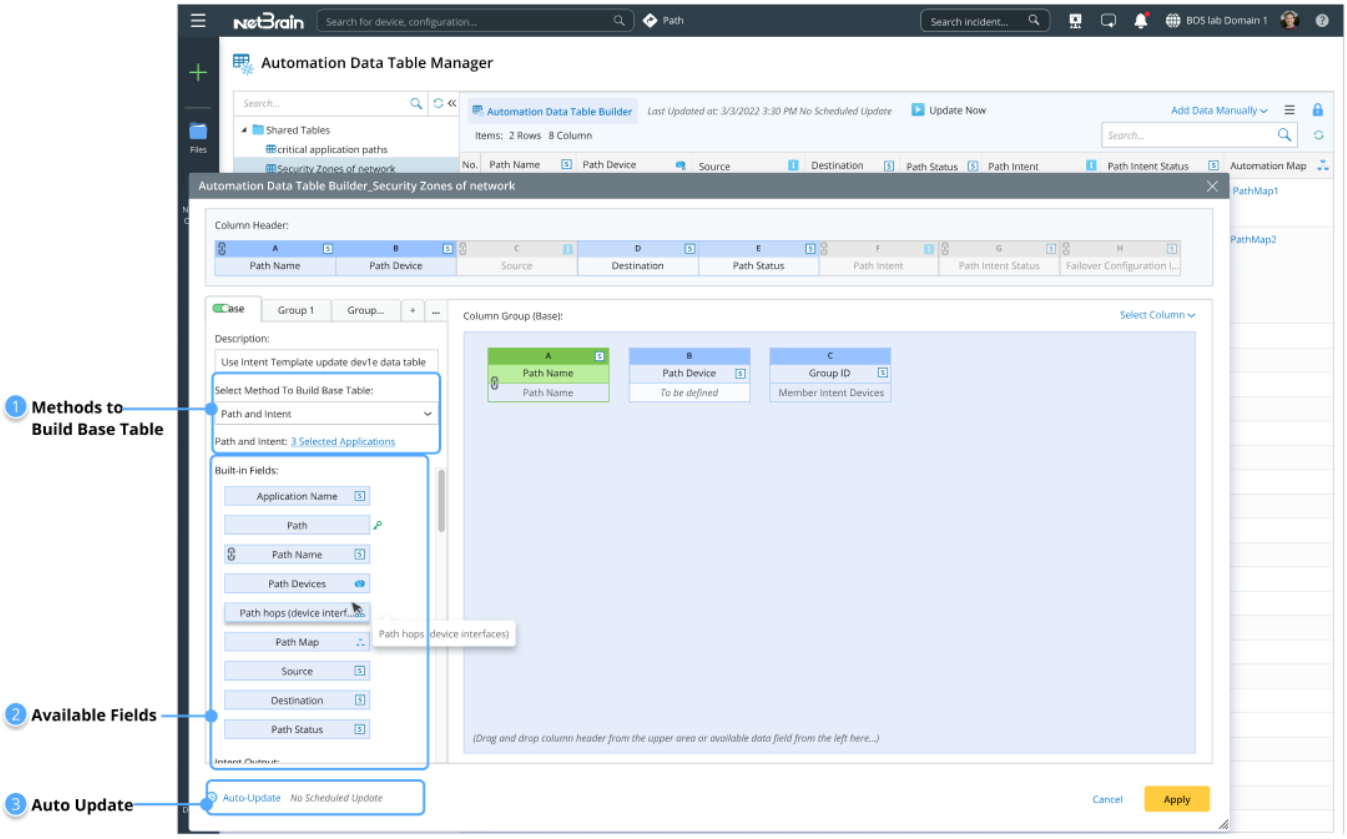


For example, users can organize all critical application paths and their automation intents into an ADT, including the key path data (such as the source and destination), then use the ADT to monitor the critical applications (PAF) and trouble any application-related issue (TAF/Chatbot/Auto Intent). The system can find the related paths for different types of events (e.g., ***a device interface has errors, the device configuration changes***) by matching the ADT column data and the event information and execute the associated path intents for the problem diagnosis and change verification.



2.2.1 Define Base Table Data

Users can use the data from the Path Browser to build the base ADT table by the method **Application Path**. Users can select paths from the Path Browser.



A Base Table Builder

Mapping Available Fields to Column Group: Drag and drop the available fields to the column group definition area to create ADT columns automatically. Select fields from the following field categories in this area:

a) Built-in Fields: The built-in fields available for this base table are listed in this area. Applicable fields included application, path (the link to this path), path name, other path properties, and path intent (see the table below for details).

Field Type	Fields	Description
Built-in Field	Application Name	The built-in fields are created from path properties.

	Path Path Name Path Devices Path hops (device interfaces) Path Map Source Destination Path Status	

b) Intent Outputs: Intent information of path intent can be displayed in the intent output columns of the ADT, such as intent message, intent status code, intent devices, intent map, intent CLI commands, and last execution time.

After ADT is created in Automation Data Table Builder, users can enable the Auto-Update for this ADT to update the data periodically.

2.2.2 Enrich Basic Table Data with Column Groups

For a base table created via the application path, column groups can be added to cover more data. The system supports several methods to populate dynamic columns for a path-based table. The table below describes each of the methods.

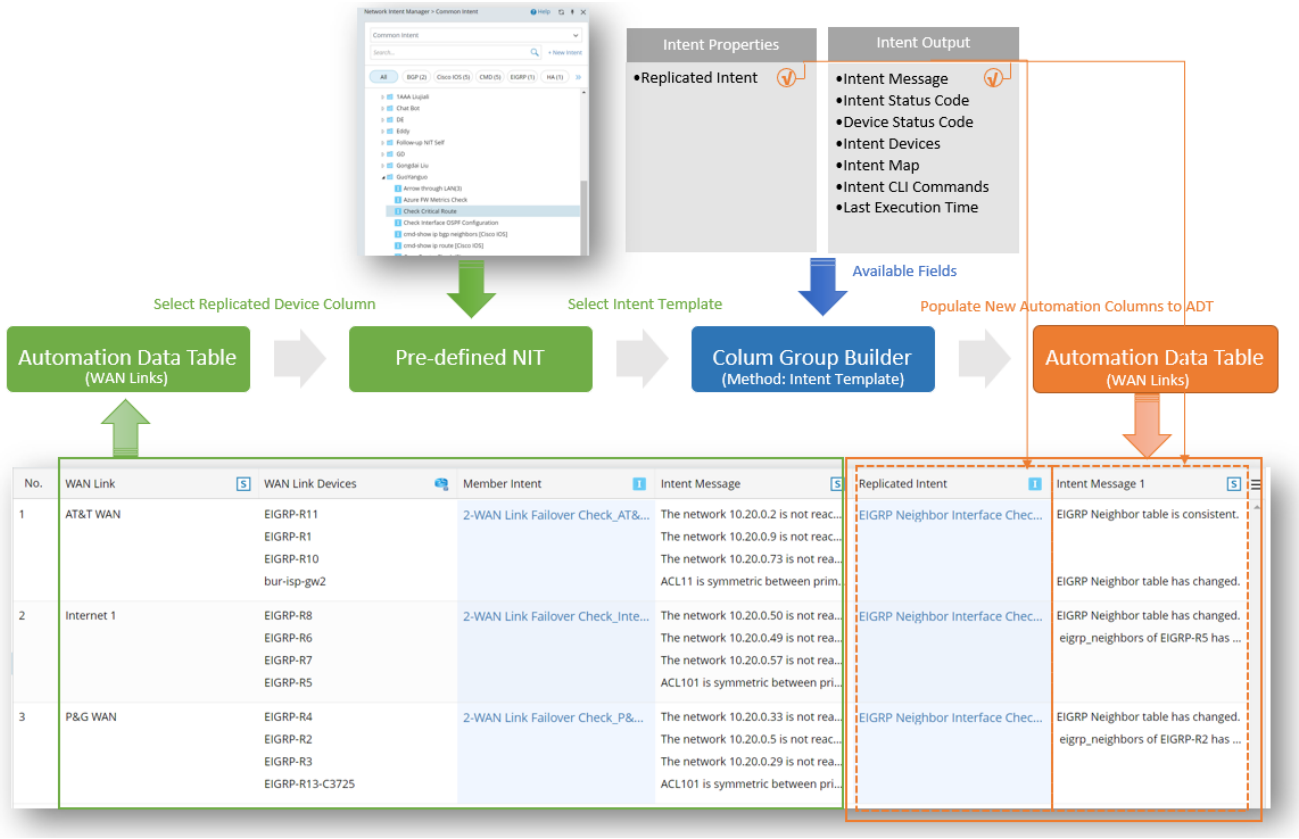
Method	When do I need to use this method?	Use Case
Use Intent Template	Add some additional intent columns for an ADT.	Intent Template (on-demand replication and Auto-Replicated) is selected and used. Users add an intent column (mandatory), then the intent outputs to additional columns.
Use Intent Cluster	Add some additional intent columns for an ADT.	Match and filter a member intent for each valid row and add the intent outputs to additional columns.
Use Auto-Probe	Add some additional probe columns for an ADT.	Filter a probe by the name using the current ADT data.

Use CSV File	Have a pre-generated CSV file and want to enrich table data with it.	Use selected key columns to match the rows in CSV files and append specified columns from CSV files to the ADT for new columns in the matched rows.
Use Function	Enrich ADT with some key column properties.	Specify a column with the type of device and use device properties and functions to fill in additional columns.
Intent Replication for Path	Want to generate some suitable path intent for application paths.	See training documentation of Intent Replication for Path for more information.

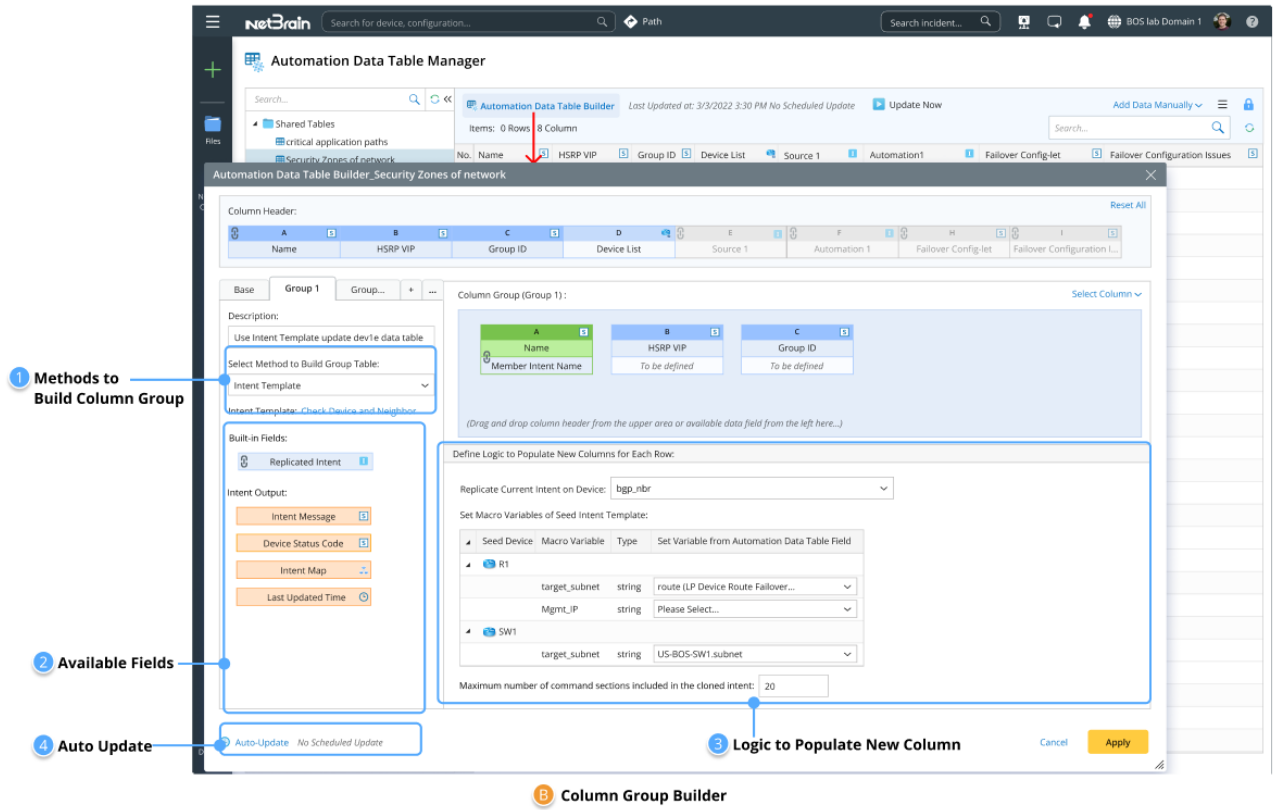
2.2.2.1 Add Automation Column from Intent Template

Based on the base data of ADT, the intent template in the automation column can be used to replicate one intent for each ADT row. This cloned intent includes the automation data in this ADT and the detailed information of the cloned intent, shown in intent output columns. Users can select a base table column containing device information for replicating intents.

The following diagram illustrates how the intent template works in Column Group Builder:



The following diagram illustrates what the Column Group Builder looks like when the **Intent Template** method is selected:



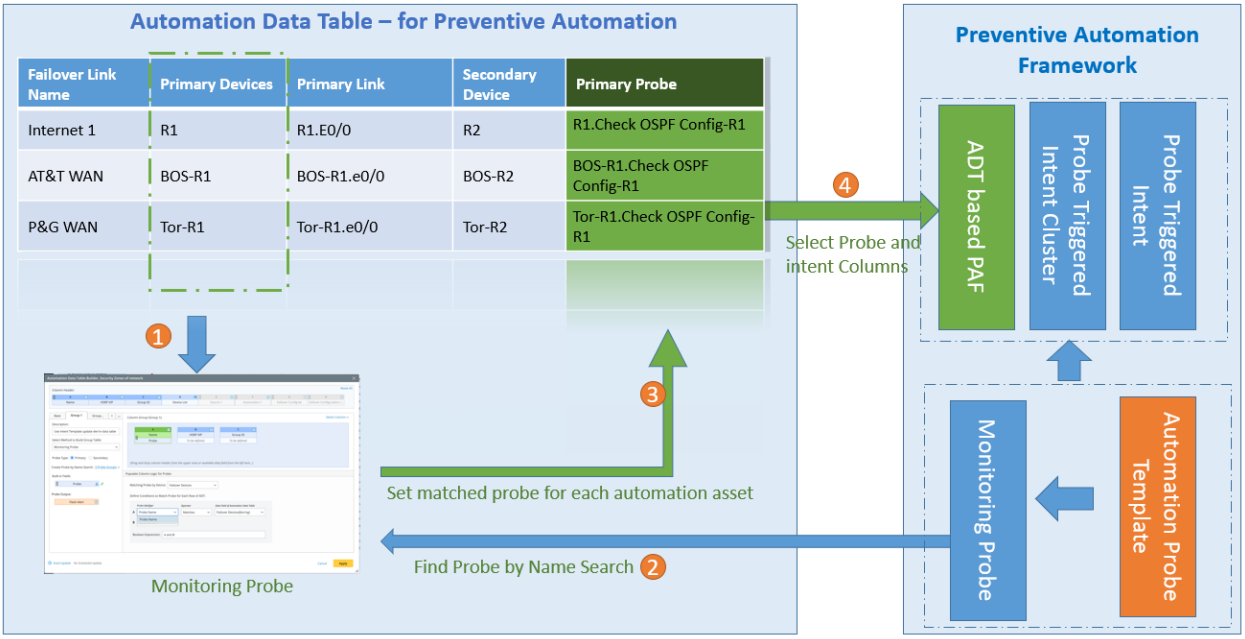
The following table describes available fields for building a group table if the intent template is the data source:

Field Type	Fields	Description
Built-in Field	Replicated Intent	The built-in fields are created from the replicated intent from the intent template.
Intent Output Field	Intent Message Intent Status Code Device Status Code Intent Devices Intent Map Intent CLI Commands	The intent output fields are created from intent details of the intents generated from the intent template.

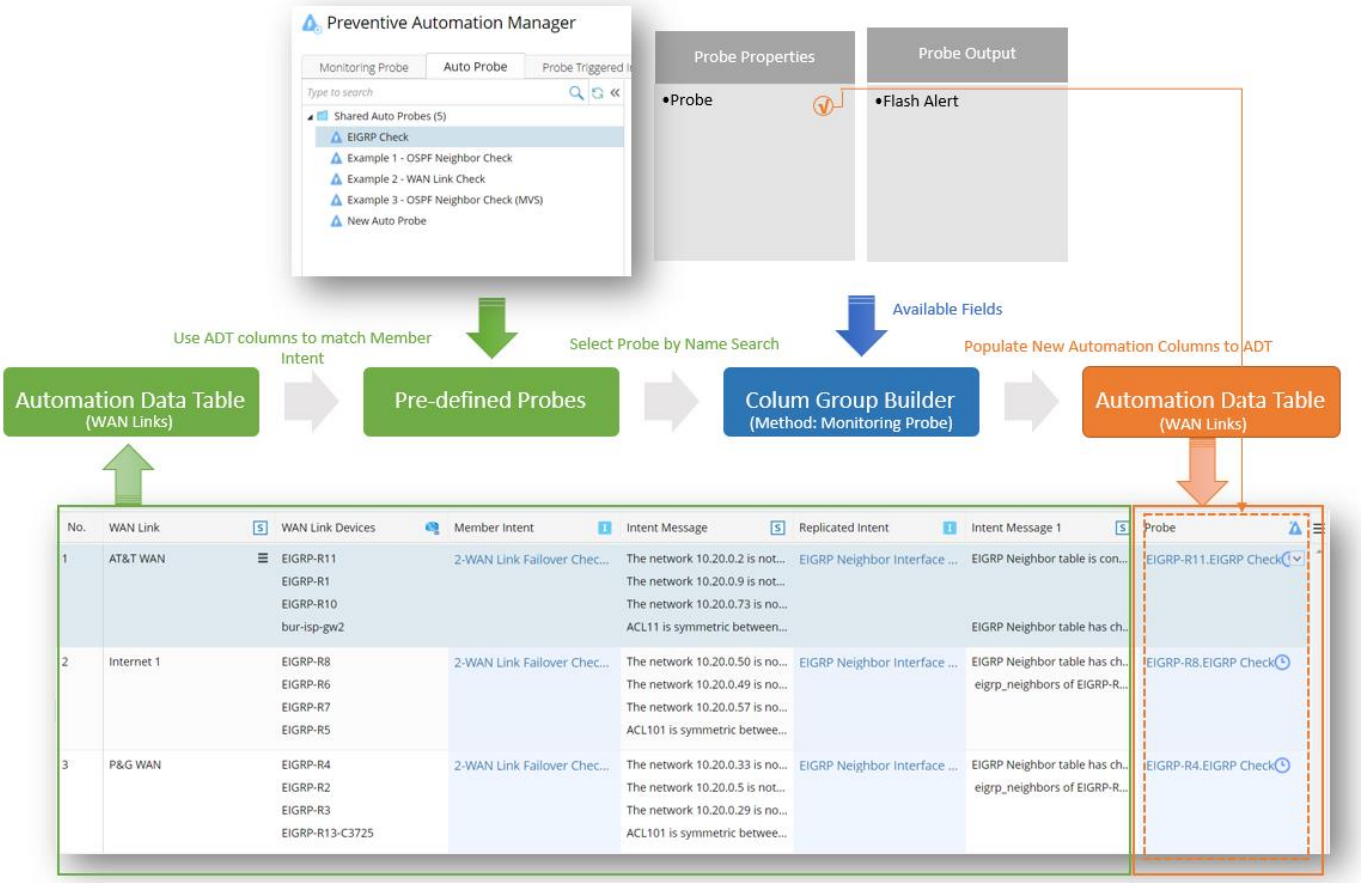
	Last Execution Time CSV Columns	

2.2.2.2 Add Probe into ADT for Prevention Automation

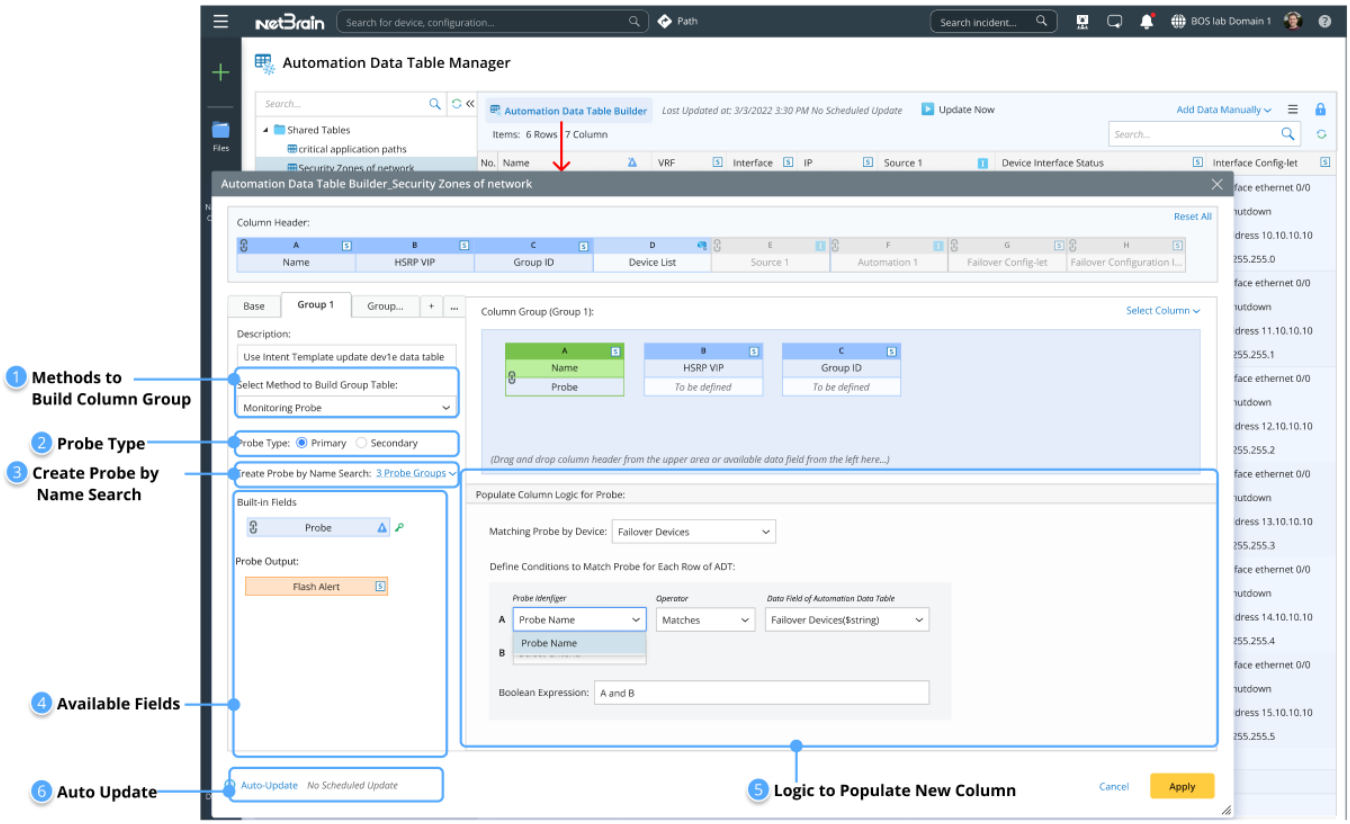
Probe installed for ADT can trigger the automation process associated with the ADT automation assets. The corresponding PDAS flow is shown in the following diagram:



A probe column can be defined for using the primary and secondary probes to trigger automation with assets in the ADT. The probe properties are used as the built-in data for the probe column. The Flash alerts generated by the probes can also be the available data source. The following diagram illustrates how Monitoring Probe works in Column Group Builder:


Preventive Automation Manager
Monitoring Probe
Auto Probe
Probe Triggered
Type to search
Shared Auto Probes (5)
EIGRP Check
Example 1 - OSPF Neighbor Check
Example 2 - WAN Link Check
Example 3 - OSPF Neighbor Check (MVS)
New Auto Probe
Probe Properties
•Probe
Probe Output
•Flash Alert
Use ADT columns to match Member Intent
Select Probe by Name Search
Available Fields
Populate New Automation Columns to ADT
Automation Data Table
(WAN Links)
Pre-defined Probes
Colum Group Builder
(Method: Monitoring Probe)
Automation Data Table
(WAN Links)
No.
WAN Link
WAN Link Devices
Member Intent
Intent Message
Replicated Intent
Intent Message 1
Probe
1
AT&T WAN
EIGRP-R11
EIGRP-R1
EIGRP-R10
bur-isp-gw2
2-WAN Link Failover Chec...
The network 10.20.0.2 is not...
The network 10.20.0.9 is not...
The network 10.20.0.73 is no...
ACL11 is symmetric between...
EIGRP Neighbor Interface ...
EIGRP Neighbor table is con...
EIGRP Neighbor table has ch...
EIGRP-R11.EIGRP Check
2
Internet 1
EIGRP-R8
EIGRP-R6
EIGRP-R7
EIGRP-R5
2-WAN Link Failover Chec...
The network 10.20.0.50 is no...
The network 10.20.0.49 is no...
The network 10.20.0.57 is no...
ACL101 is symmetric betwee...
EIGRP Neighbor Interface ...
EIGRP Neighbor table has ch...
eigrp_neighbors of EIGRP-R...
EIGRP-R8.EIGRP Check
3
P&G WAN
EIGRP-R4
EIGRP-R2
EIGRP-R3
EIGRP-R13-C3725
2-WAN Link Failover Chec...
The network 10.20.0.33 is no...
The network 10.20.0.5 is not...
The network 10.20.0.29 is no...
ACL101 is symmetric betwee...
EIGRP Neighbor Interface ...
EIGRP Neighbor table has ch...
eigrp_neighbors of EIGRP-R...
EIGRP-R4.EIGRP Check

The following diagram illustrates what the Column Group Builder looks like when the **Monitoring Probe** method is selected:



B Column Group Builder

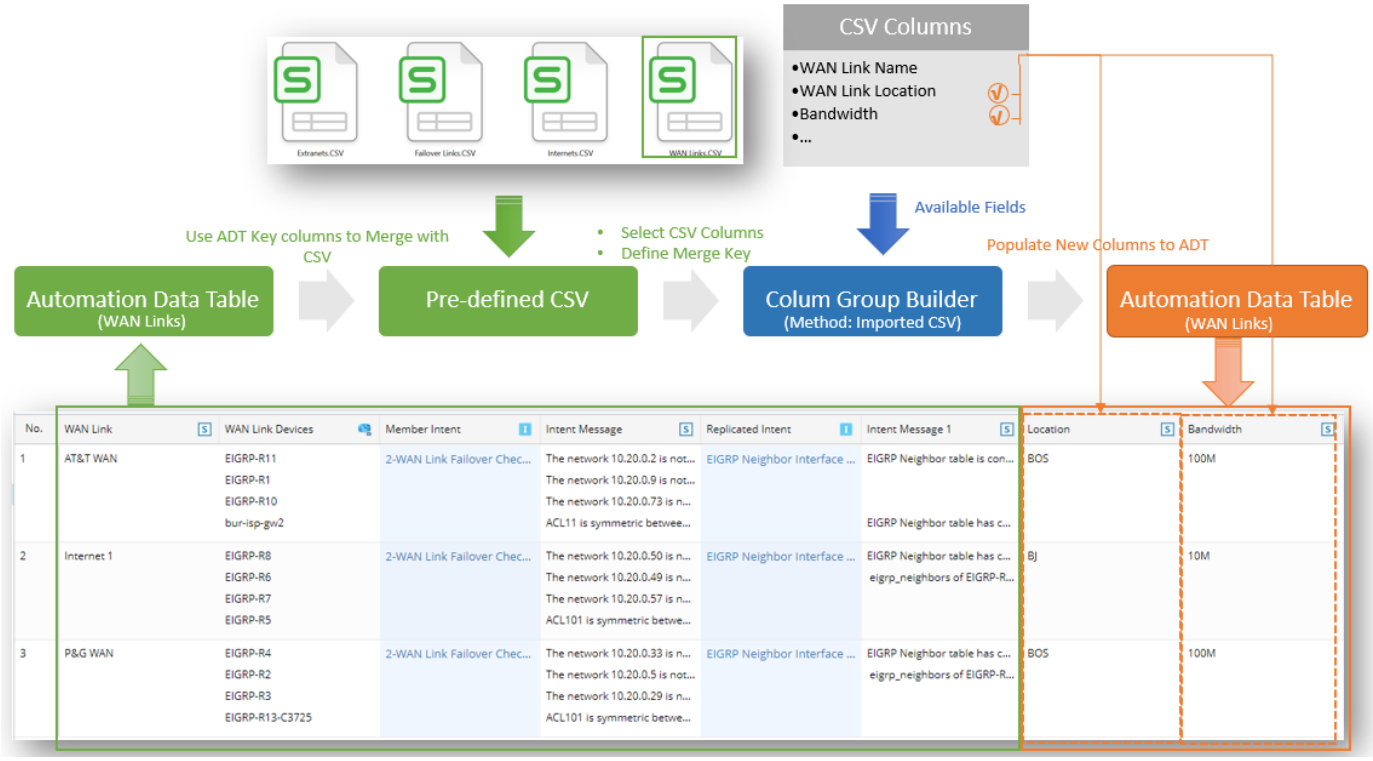
The following table describes the data fields for building group columns via a monitoring probe:

Field Type	Fields	Description
Built-in Field	Probe	The built-in field is the probe selected.
Output Field	Flash Alert	The Output Field is created by flash alerts generated by the probes.

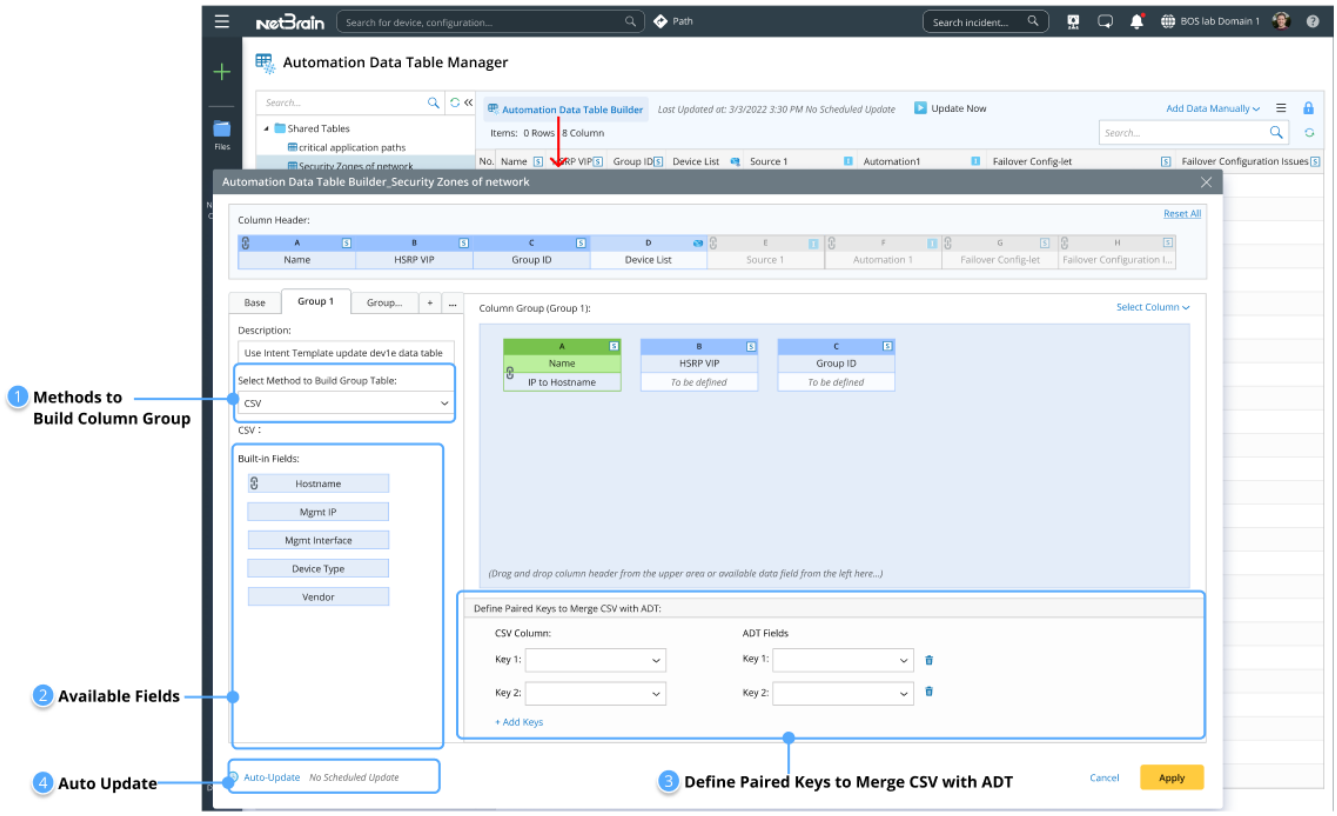
2.2.2.3 Add ADT Column from Imported CSV

If users have local CSV files containing data, the CSV file can be imported to create an ADT group table. By default, all columns in the CSV are added as available fields in ADT. To successfully merge the CSV columns to the base table, users can select fields from the CSV and fields in ADT as the table keys to be paired.

The following diagram illustrates how Imported CSV works in Column Group Builder:



The following diagram illustrates what the Column Group Builder looks like when the **Imported CSV** method is selected:

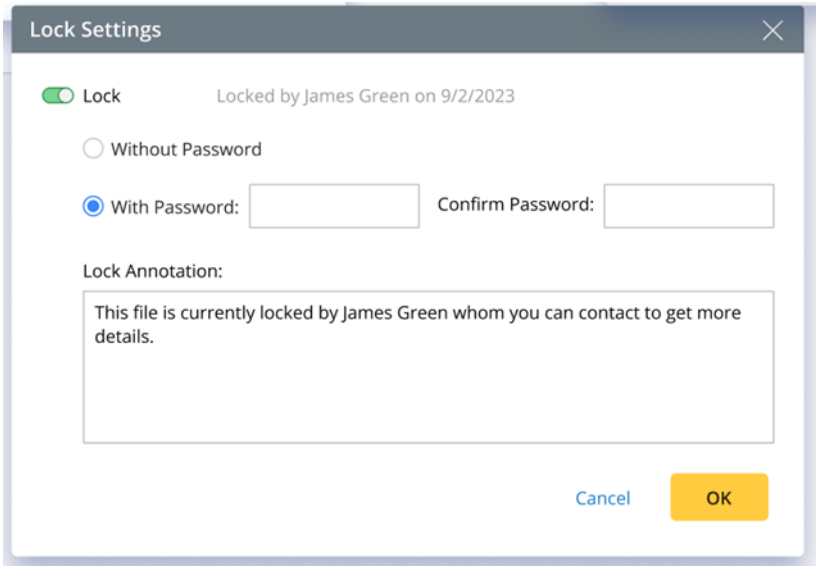


2.3 Other ADT Functions

2.3.1 Lock/Unlock ADT and ADT Editing Rights Control

To avoid mis-operations on the ADT data, ADT can be locked/unlocked by its creator or other users with the privilege. Moreover, editing rights are introduced to the ADT feature to prevent data loss due to editing conflicts.

For each ADT table, the creator of ADT can switch on the Locking function and set a locking mode (With Password or Without Password). A locked ADT is in View-Only mode and can be edited after entering the password. Lock annotation can be added to provide detailed lock information.



2.3.2 Editing Rights Control for ADT

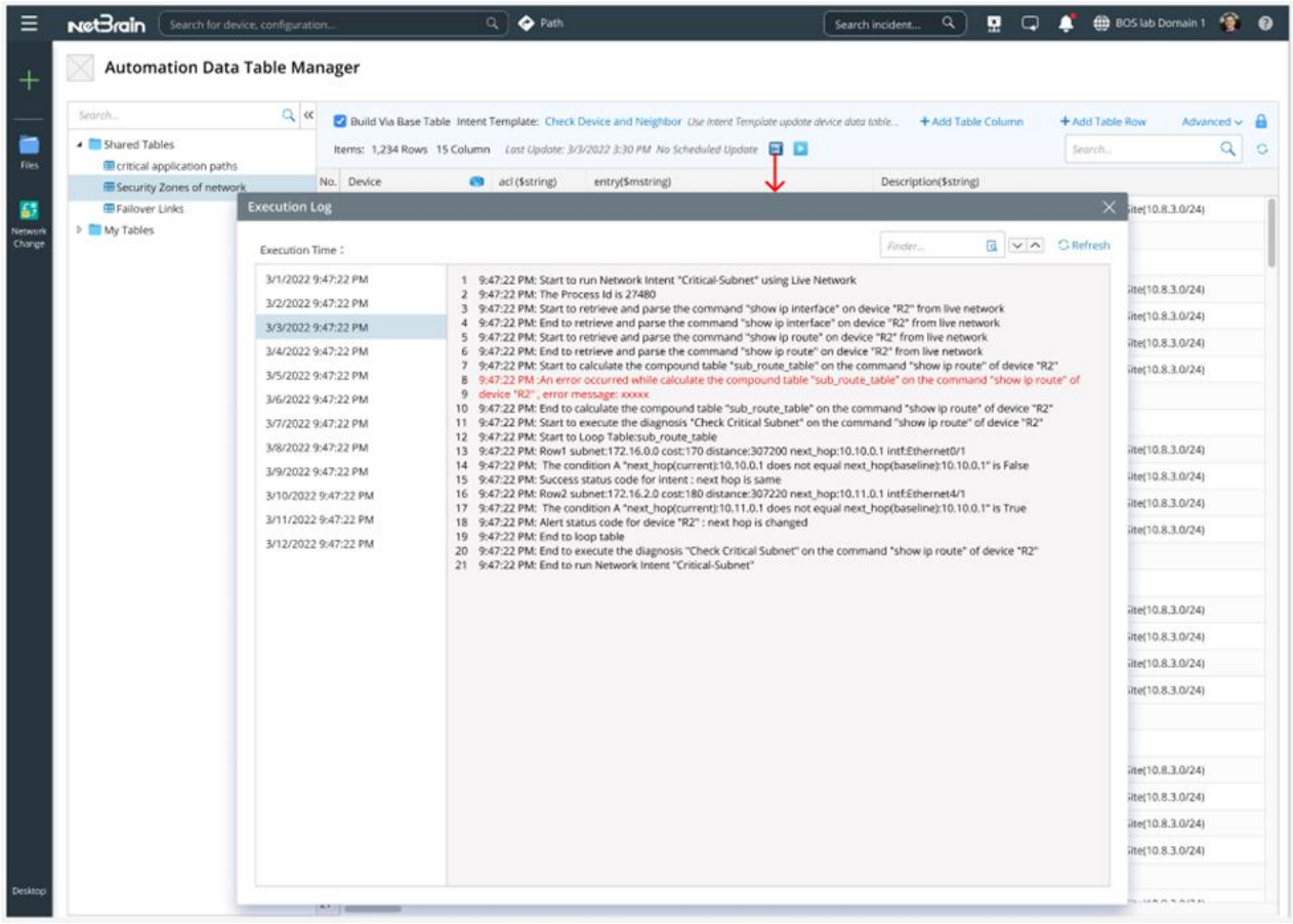
An ADT may be edited by two or more users simultaneously; in this case, one of the edited ADT cannot be saved, causing data loss. To prevent such a problem, the user editing the ADT first owns the editing rights, and only the user with editing rights can edit the ADT.

If a user without editing rights attempts to edit the ADT, a message will notify the user that the editing right is required.

2.3.3 View ADT Execution Log

The execution log is provided for an ADT to record the operations such as replicating intents, matching intent and the intent execution errors. Users can check and export the ADT execution log. Users can specify a time

range to check the execution log for the period.



2.3.4 ADT Audit Log

The audit log is provided to record and track the changes made to ADT to protect essential ADT assets. The ADT audit log records the following operations:

- Create ADT: Record relevant information when a new ADT is created.
- Delete ADT: Record relevant information when an ADT is deleted.
- Edit ADT: Record relevant information each time the ADT is edited.

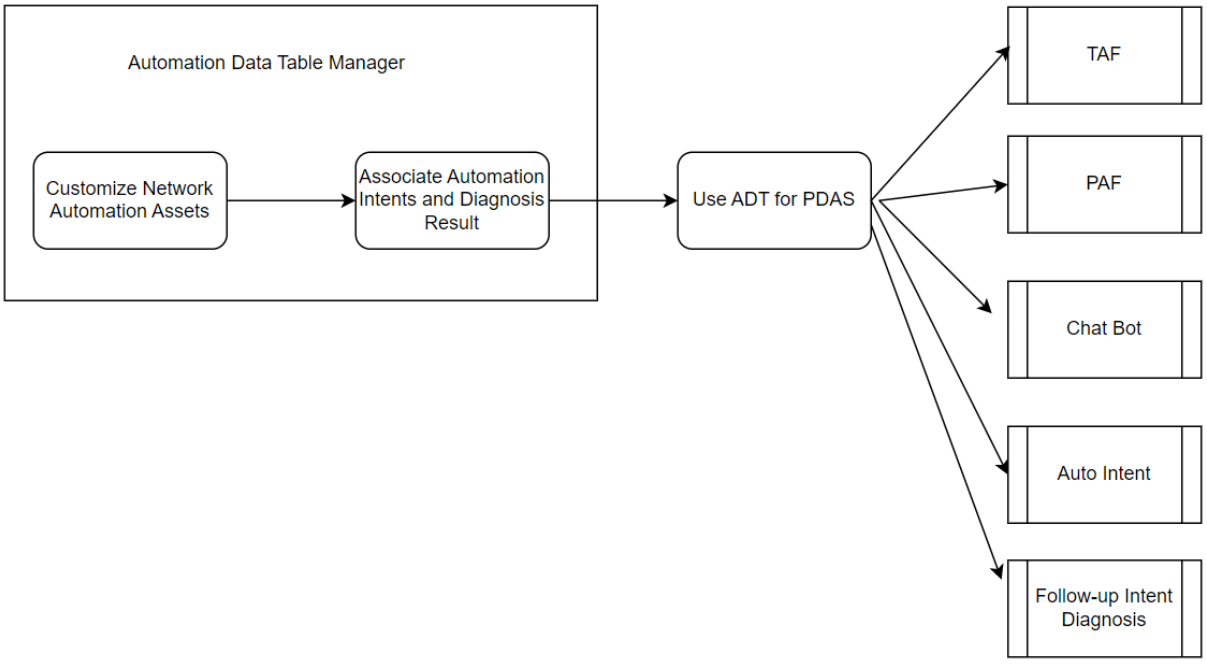
2.3.5 User Privilege for ADT

ADTs are organized in folders. Two privileges (Shared Resource and File Management and Private Resource Management privilege) are associated with ADT.

- Open ADT Manager: a user having the Shared Resource and File Management privilege or Private Resource Management privilege can open the ADT manager.
- Edit private ADT: Require Private Resource Management privilege to edit a private ADT.
- Select ADT: No privilege is required for selecting an ADT.
- Use Intent-based ADT: Require the privilege to edit intent to edit the ADT.

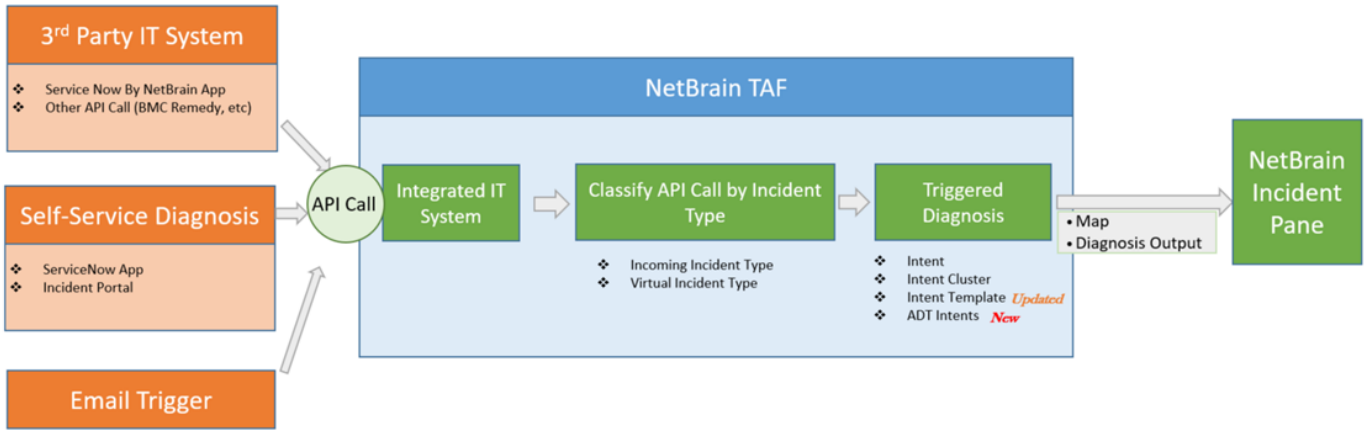
3 ADT Based PDAS

ADT serves as a data reservoir in NetBrain. It can be used across functions (TAF, PAF, Auto Intent, Chat Bot, and follow-up Intent Diagnosis) to provide data for the proper operations.



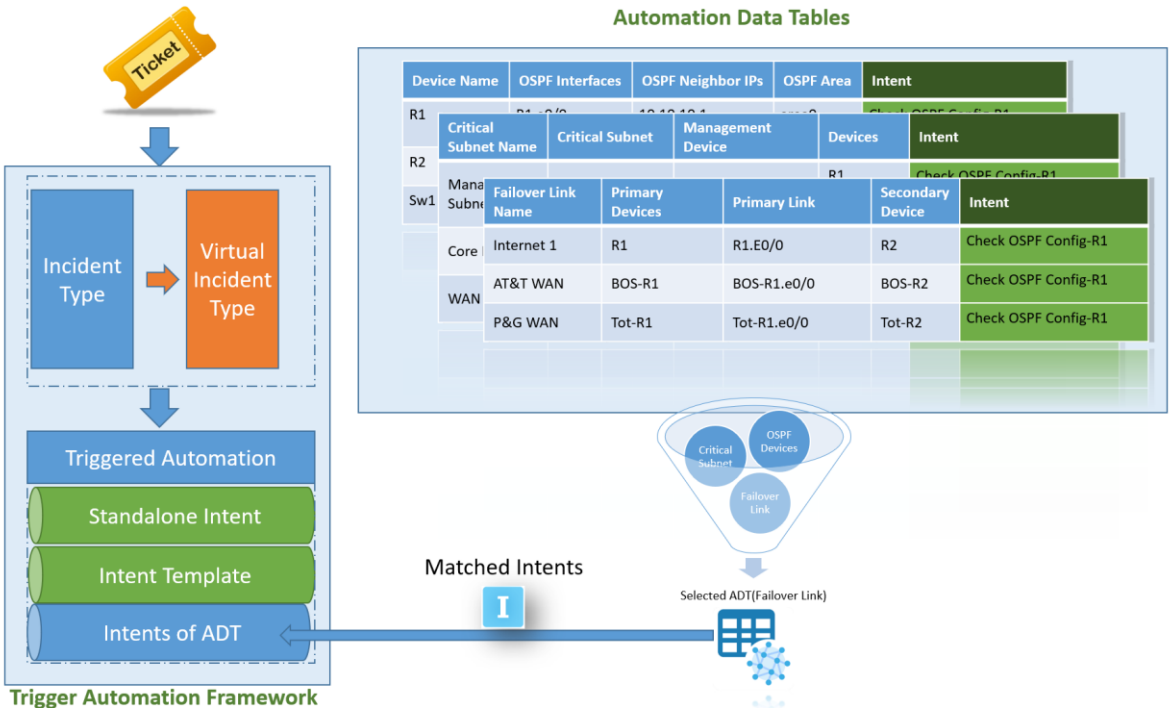
3.1 Install ADT Intents for TAF

To analyze and solve the network issues of automation assets, we can install Intents of ADT to TAF and trigger these intents by the 3rd party events. For this purpose, R11.1 improved TAF Triggered Diagnosis to support ADT intents, a set of Intents associated with specific assets (e.g., application paths, critical failover links) in ADT. Another TAF enhancement is to support assigning values to Macro Variables in the intent template.

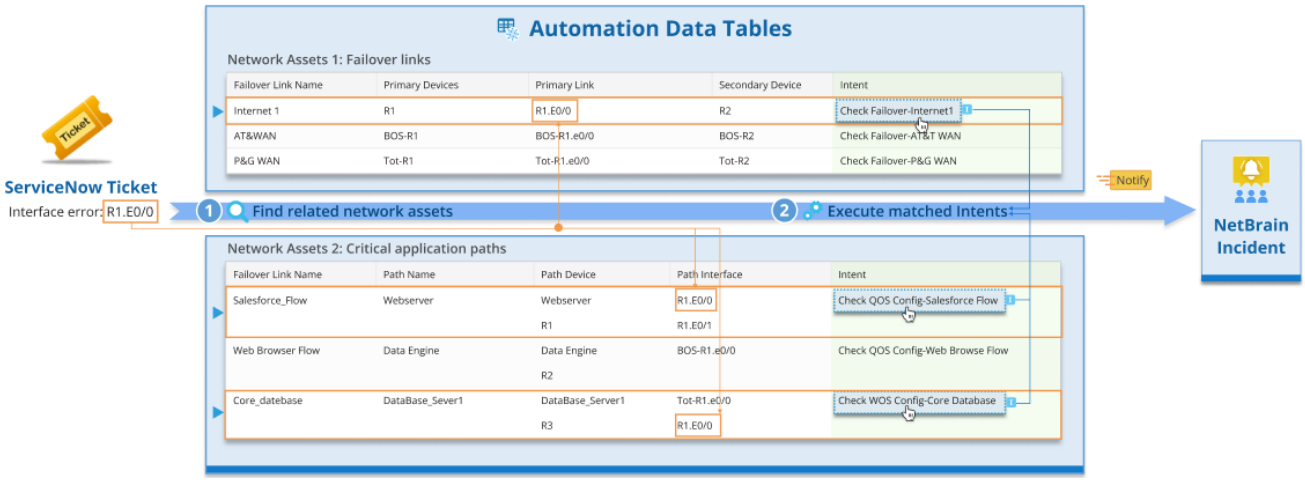


3.1.1 Trigger Diagnosis Using ADT Intents

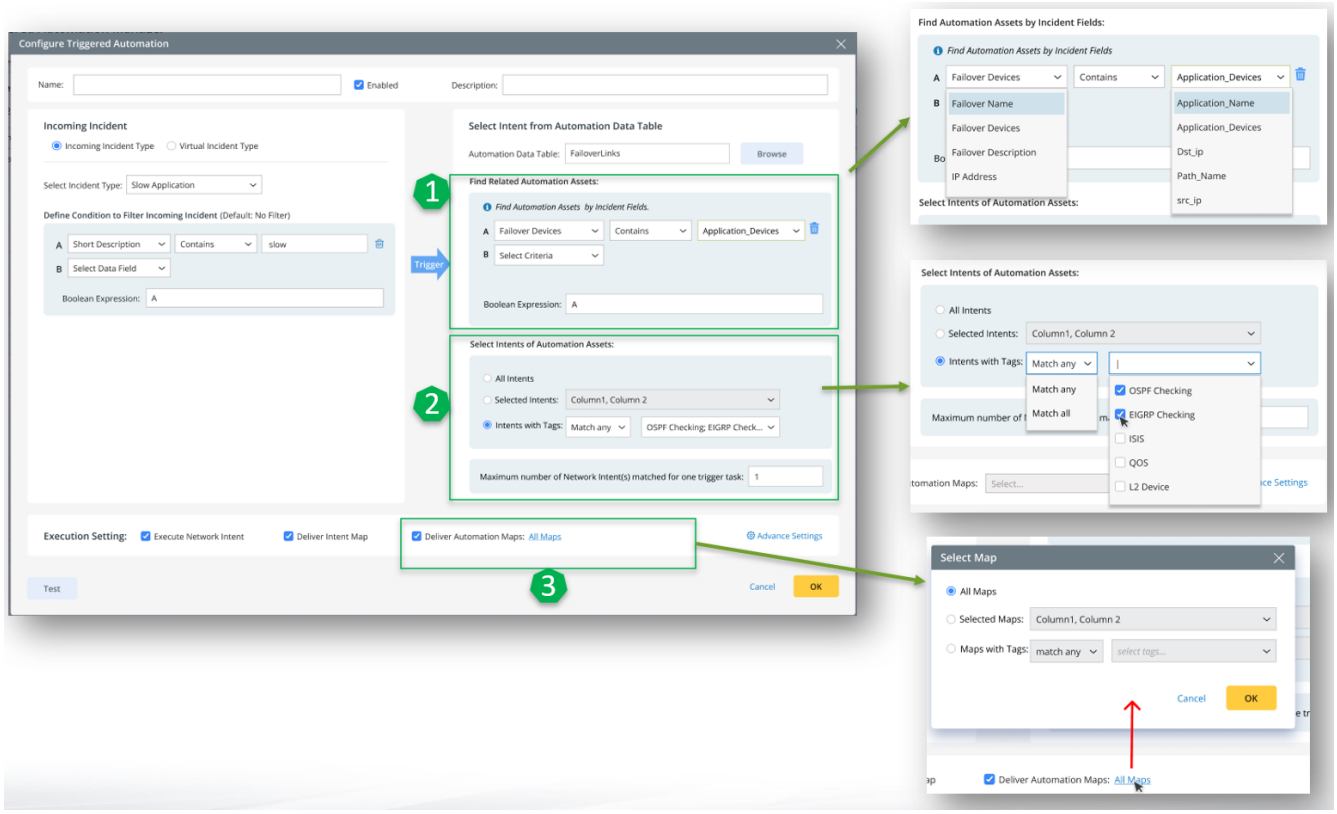
For issues related to Automation Assets, it is preferred to trigger ADT intents to troubleshoot issues related to automation assets, e.g., **checking failover links** and **checking Critical Routes**. To conveniently address and analyze the network issues of automation assets, users can install Intents of ADT to TAF and trigger these intents by events (tickets from a third party).



The following diagram illustrates an example of how ADT-based PDAS works in TAF:



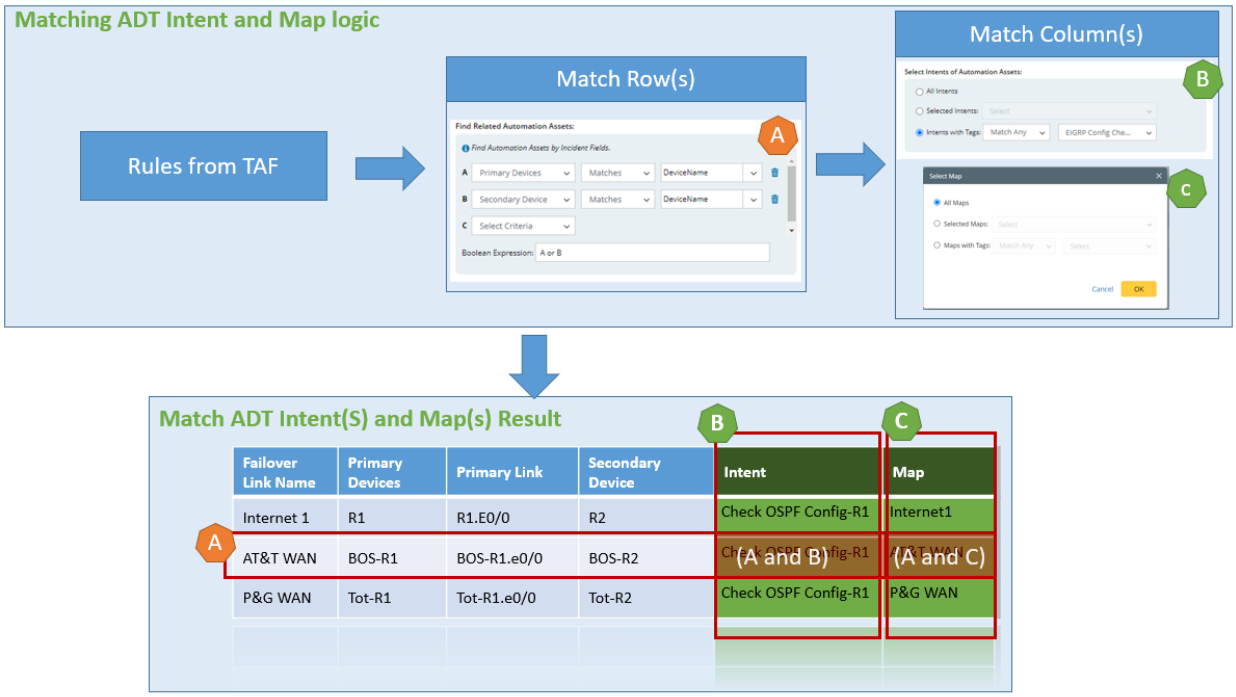
The key logic of triggering ADT intent is illustrated in the following diagram:



1. Find related Automation Assets. Find matching row data based on whether the corresponding incident field exists in the ADT column.

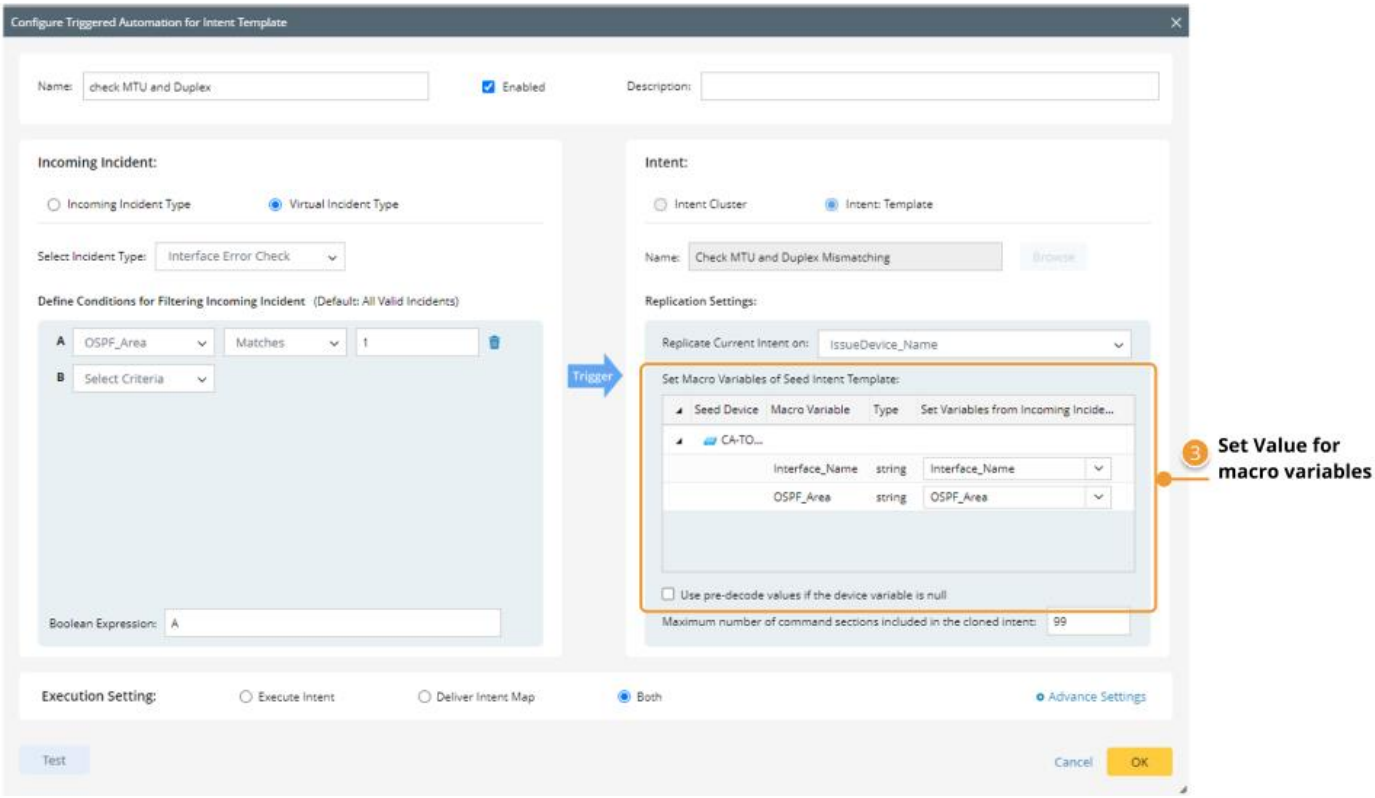
2. Trigger associated intents of Automation Assets. All the intents of the matched ADT rows will be triggered by default, and the user can filter the intents to be executed with automation tags or manually choose intents by selecting columns.
3. Deliver automation Maps of Automation Assets to the Incident pane. All the maps of the matched ADT rows will be delivered to the incident pane by default. Users can filter the delivered maps with automation tags or manually choose maps by selecting columns.

The logic for matching ADT intents in ADT-based TAF is shown in the following diagram below:



3.1.2 Trigger Diagnosis Using Intent Template

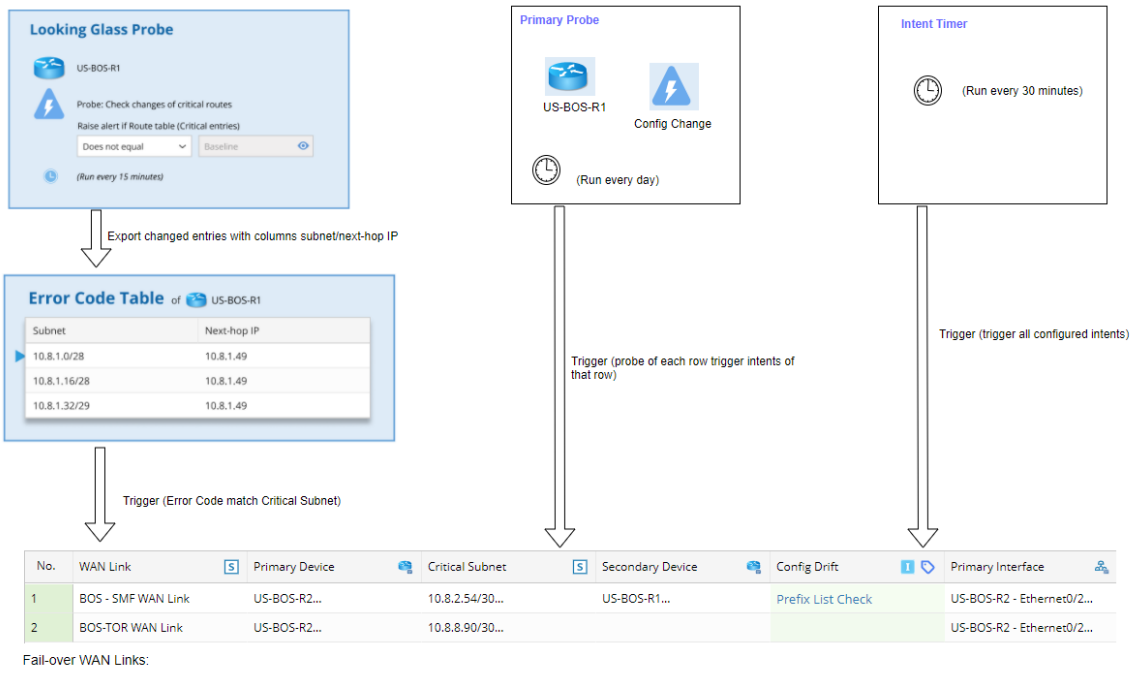
The Intents can be replicated from the intent template for troubleshooting. In the process of installing the intent template to TAF, in some cases, some macro variables of the intent template need to be set to a certain specific value, e.g., the target subnet for ***show ip route <subnet>*** command. To support TAF in these cases, users can enter values for macro variables of the intent template during the TAF installation. Data fields of incoming incident type can be used as the data source to set the value for macro variables.



3.2 Install ADT Intents to Preventive Automation

ADT can be installed in the Intent Based Automation Center (IBA) for preventive automation. R11.1 updates the PAF function based on ADT in many ways, including:

- ADT-based PAF configuration: To ensure the critical assets defined within ADTs are functioning properly, the system executes the intents of ADT by preventive automation through one of the following ways:
    - **Looking Glass Probe**: Get the monitor status of a device via SNMP/CLI. Then based on the alert data, the intents defined within ADT intents can be matched and executed.
    - **Primary Probe**: Trigger the intents defined in the same row based on the probe column defined in the ADT table.
    - **Intent Timer**: Trigger the intent execution based on the intent timer pre-defined.



Fail-over WAN Links:

- Probe Instance enhancements: Improve the monitoring probe to support ADT-based PAF function.
- Auto Probe: Provide a quick way for power users to batch-create probes on a set of qualified devices or interfaces.

3.2.1 ADT- Based Looking Glass Probe

3.2.1.1 Looking Glass Probe

A probe performs diagnosis at the device level to determine its health status for a single device. Running a probe for the entire network requires the deployment of this probe on all devices, which leads to inefficiencies and increased traffic for both network devices and the NetBrain system. Alternatively, users can run the probe on a small set of devices (**looking glass probes)**, which can detect any anomaly for the whole network. Then, these probes will parse alert information to identify problematic entries and trigger intent diagnosis for a larger group of devices. Typical looking glass probes include:

- **Route Table Check**: Filter critical route from the ***show ip route*** and compare the data with the baseline to verify the routes remain the same. Any changed routes can be exported to the error code to trigger the corresponding intent diagnosis stored in ADT.

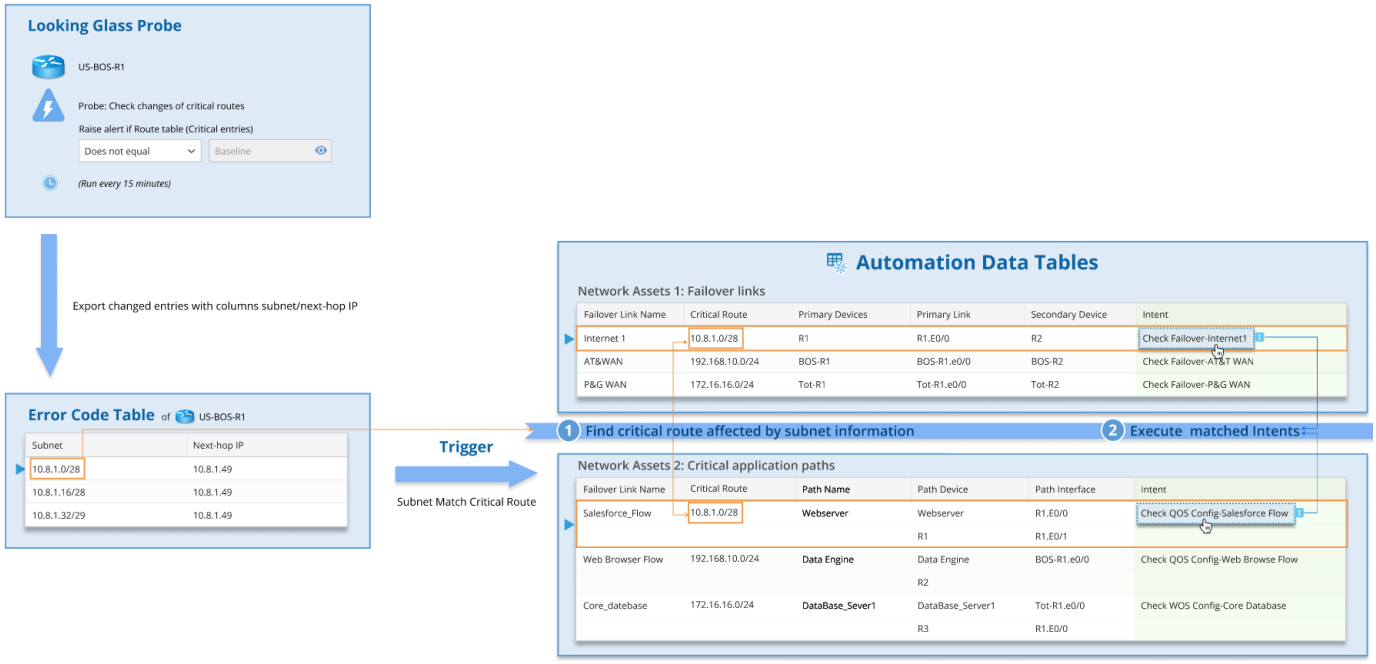
- **Neighbor Table Check**: Using the command ***show ip bgp neighbor***, compare the BGP neighbor status with baseline data to verify these neighbors are healthy. In case of any neighbor with issues, the neighbor information can be exported to the error code to trigger the intent diagnosis stored in ADT.

Users can choose the appropriate looking glass probes to monitor the network health status while utilizing minimal resources on both network devices and NetBrain System. If the looking glass probe generates an alert and the error code matches the corresponding intents in the ADT table, the configured intents will be triggered.

For example, a looking glass monitors the critical route status on a device. If there is any change in the routing entry for the looking glass probe, the changed entry (added, removed, and modified entries) will be exported into the error code. Then, the error code can be used to match the intents of ADT. The following table is an example ADT for the critical WAN link asset:

Failover Link Name	Primary Device	Critical Route	Secondary Device	Primary Link	Intent1	Intent2
Internet 1	R1	10.8.1.0/28	R2	R2.E0/0	Monitor Failover Link Health - Internet1	Verify QOS Config - Internet1
AT&T WAN	BOS-R1	192.168.10.0/24	BOS-R2	BOS-R2.e0/0	Monitor Failover Link - AT&T WAN	Verify QOS Config - AT&T WAN
P&G WAN	Tot-R1	172.16.16.0/24	Tot-R2	Tot-R2.e0/0	Monitor Failover Link Health - P&G WAN	Verify QOS Config - P&G WAN

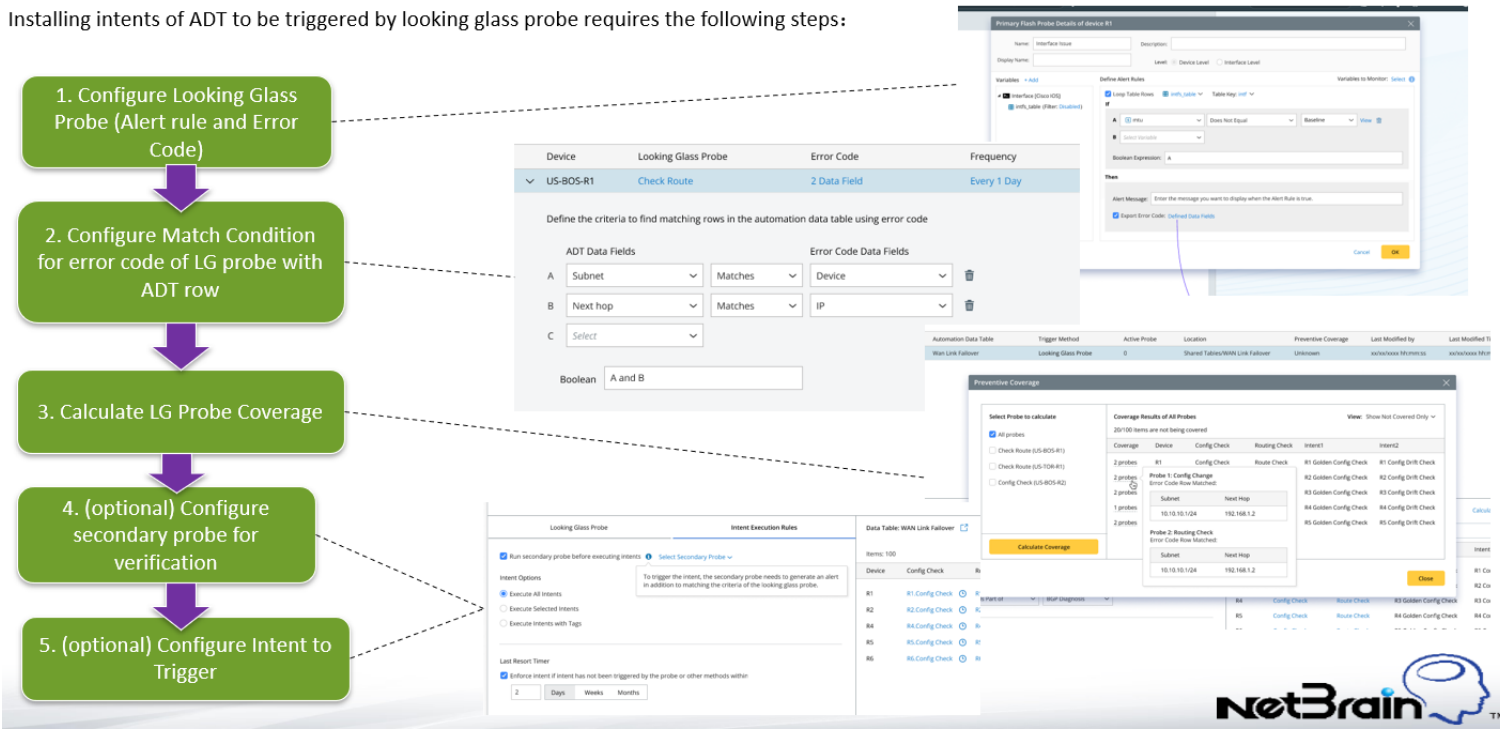
Based on the error code table contents, we use the subnet contents to look up if there is any entry match.



The above diagram shows the error code generated by looking glass probe for several subnets and the changed subnet information exported from the routing table. Then, the system matches these subnets with the ADT Table column “critical route” to check if any WAN link is affected and find the first Failover link that contains the affected critical route. Further, the system will trigger the intent execution (***Monitor Failover Link Health-Internet1*** and ***Verify QOS config – Internet1***).

3.2.1.2 ADT Creation Flow Based on Looking Glass Probe

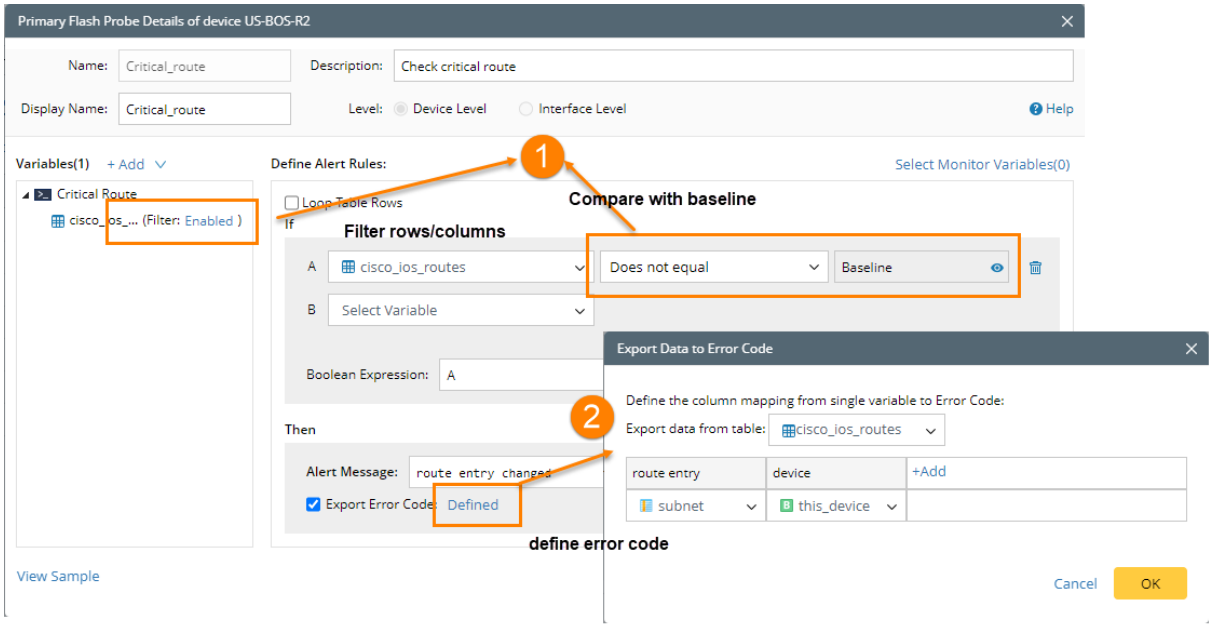
The process of configuring looking glass probes to trigger ADT involves the following steps:



R11.1 enhances the probe with the following functionalities to support the looking glass probe:

1. **Compare with Baseline Data and Filter Rows/Columns if needed.**
    - Probe diagnosis results can be compared with baseline data that is initially stored when the probe is executed, and the data can be used as the baseline to check against further data.
    - Table columns can be filtered by ADT table to keep critical ones or exclude non-important items.
    - Certain table columns can be filtered so that only critical columns will be considered.
2. **Export Error contents to error code.**

    Specified contents can be exported to the Error Code and can be used to match ADT columns.

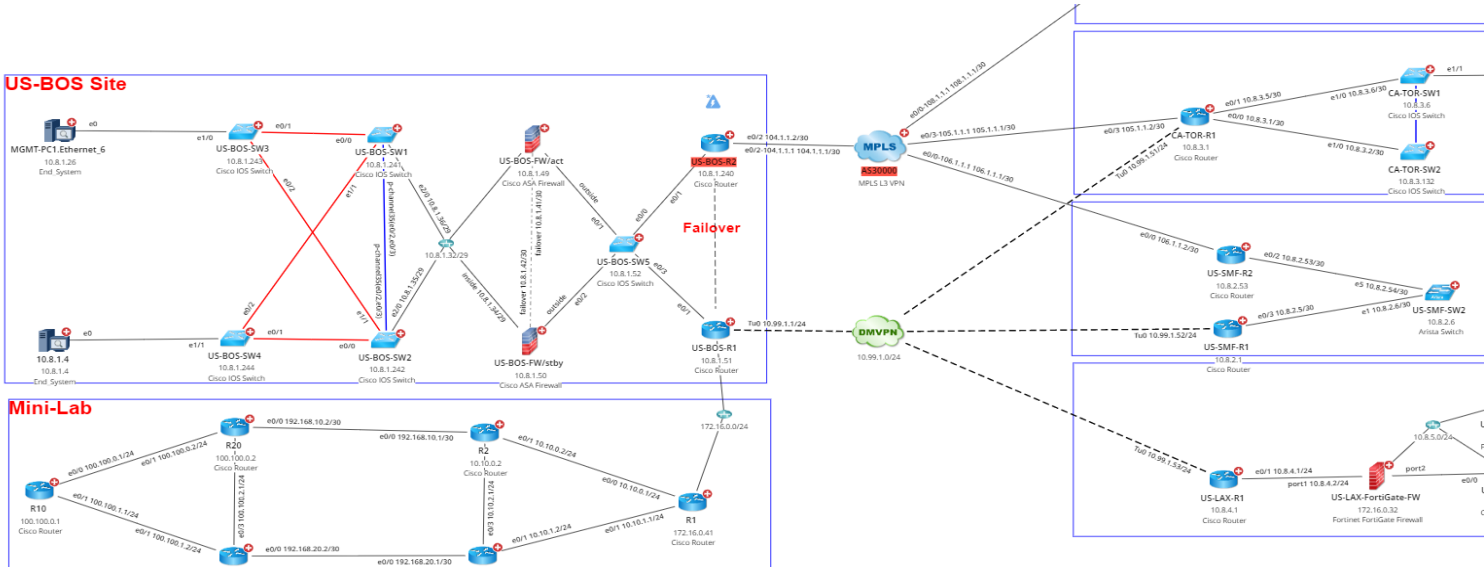


3.2.1.3 Looking Glass Probe Use Cases

Use looking glass probe to monitor critical parameters that can be used to drive the diagnosis of related network devices, such as critical routes, topology neighbor check and routing neighbor check.

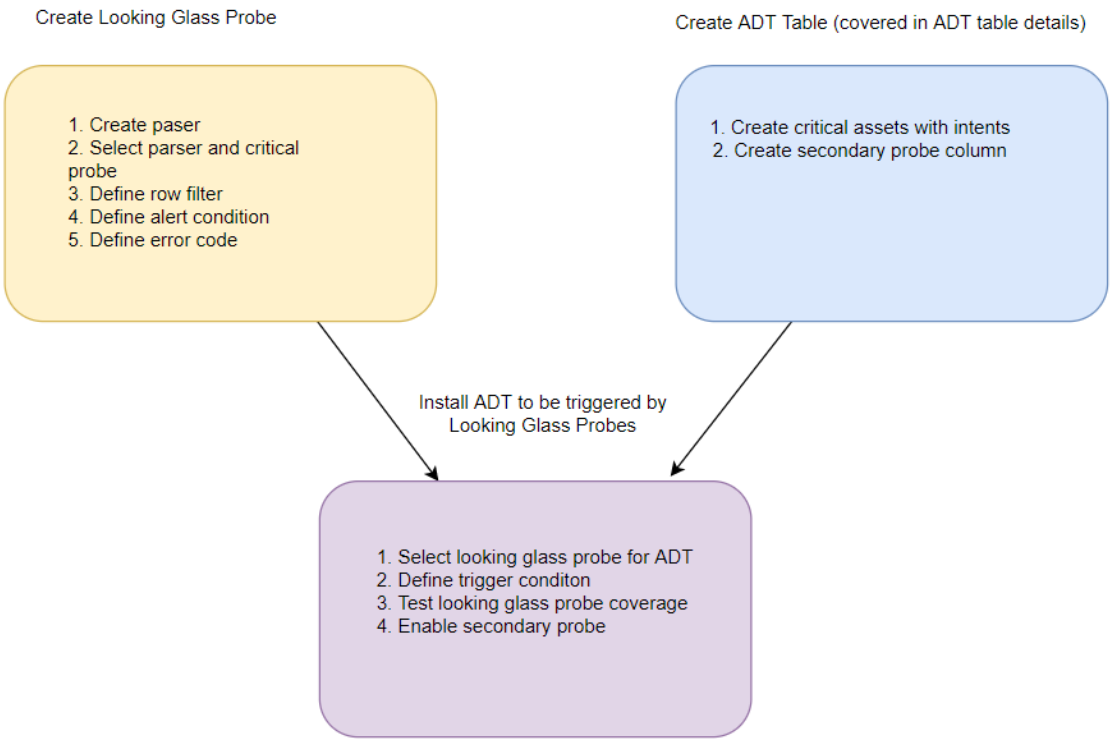
- **Routing Table Check**: You can use the command ***show ip route*** and filter critical route to compare the data with the baseline to verify the routes remain the same. Any changed route can be exported to the error code to trigger the corresponding intent diagnosis stored in ADT.
- **Topology Neighbor Check**: You can use the command ***show cpd neighbor*** or ***show lldp neighbor*** to figure out the neighbor change for a core device, and from the changed entries, you can trigger the diagnosis of neighboring devices.
- **Routing Neighbor Check**: You can use the command ***show ip bgp neighbor*** and compare the BGP neighbor status with baseline data to verify whether these neighbors are healthy. If any neighbor has issues, the neighbor information can be exported to the error code, further triggering the intent diagnosis stored in ADT.

The following example shows a looking glass probe installed on ***US-BOS-R2*** for critical route check, where the border device can identify critical route changes and trigger the intent diagnosis for each critical route.



The creation of an ADT table triggered by Looking glass probe involves the following steps:

- Create looking glass probe: Create looking glass probes to create alerts.
- Create an ADT table: Create an ADT table for critical assets.
- Install ADT to be triggered by looking glass probes: install the ADT table to be triggered by looking glass probes.



3.2.2 Use Primary Probe to Trigger Intents of ADT

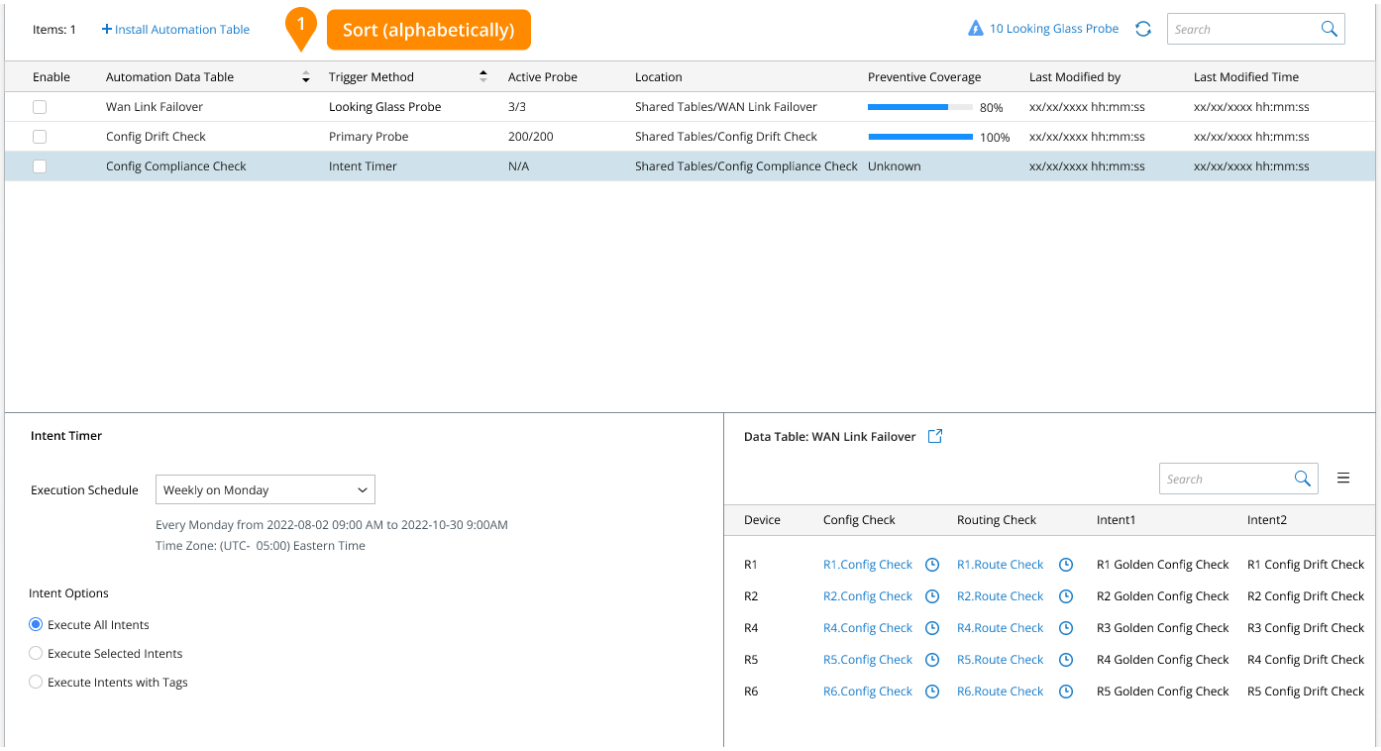
Looking glass probes may not be able to trigger all ADTs since some Intents may not find looking glass probes. In this case, use the probes defined in the ADT column to trigger the intent execution.

Users can add the primary probe column in the ADT tables, and the probe with its polling frequency can be executed periodically. The probe will trigger the configured intents within the same row it creates any alert.

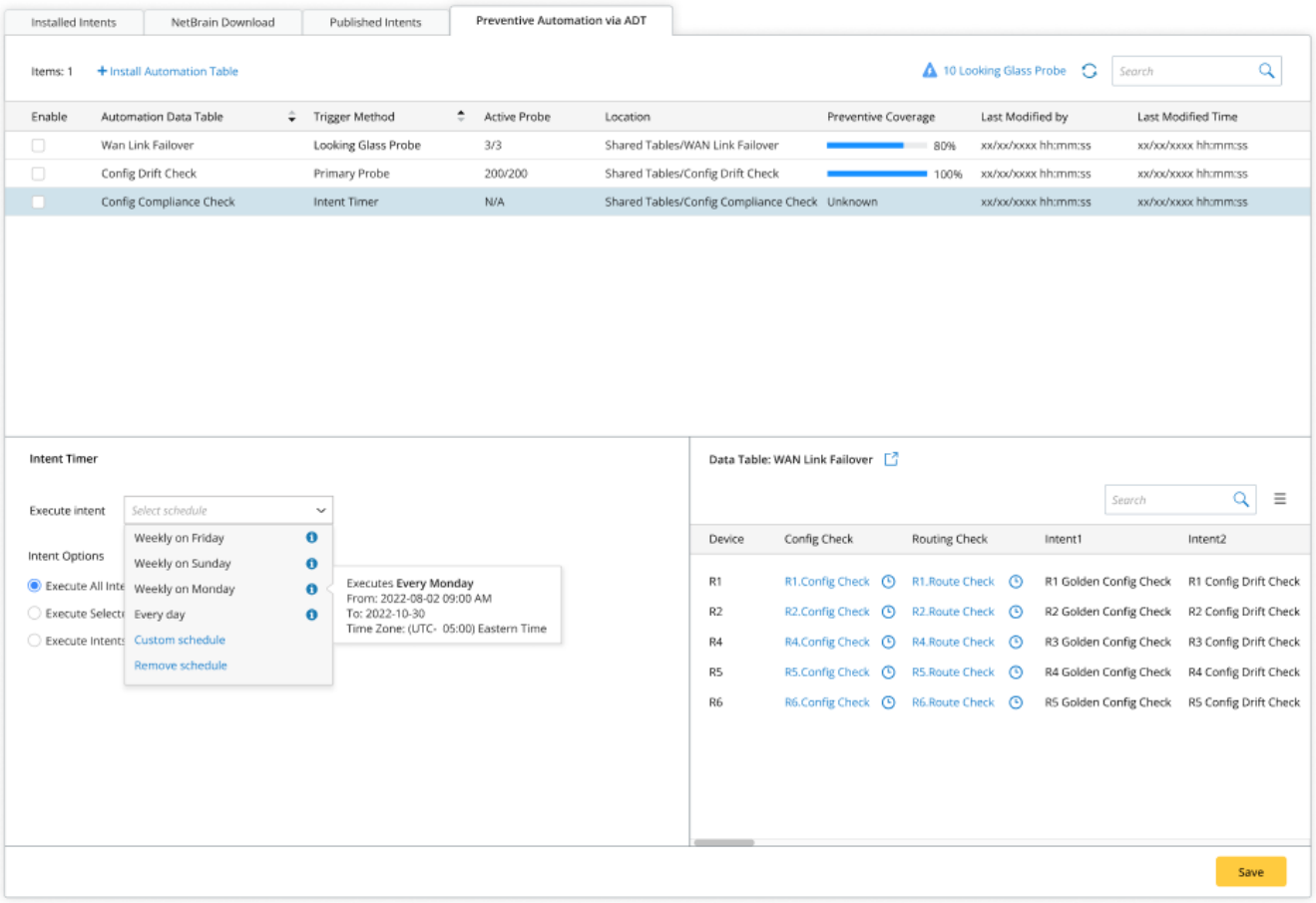
No.	Name(Primary Probe)	VRF(String)	Interface(String)	IP(String)	Source 1	Device Interface Status	Interface Config-let
1	CA-TOR-R1.CPU Check	RED Blue	e0/0 e0/1	10.10.10.10/24	Check Interface Status1	There aren't any issues for this interface	Interface ethernet 0/0 no shutdown ip address 10.10.10.10 255.255.255.0
2	CA-TOR-R1.CPU Check	Blue Green	e0/1 e0/2	12.10.10.10/24	Check Interface Status2	There aren't any issues for this interface	Interface ethernet 0/0 no shutdown ip address 11.10.10.10 255.255.255.1
3	CA-TOR-R1.CPU Check	Green RED	e0/2 e0/3	13.10.10.10/24	Check Interface Status3	There aren't any issues for this interface	Interface ethernet 0/0 no shutdown ip address 12.10.10.10 255.255.255.2
4	CA-TOR-R1.CPU Check	RED	e0/3	14.10.10.10/24	Check Interface Status4	There aren't any issues for this interface	Interface ethernet 0/0 no shutdown ip address 13.10.10.10 255.255.255.3

3.2.3 Use Intent Timer to Trigger Intents of ADT

Some ADTs may not be triggered by looking glass probes or primary probes. Instead, users can use the intent timer to run the intents of the ADT table periodically.

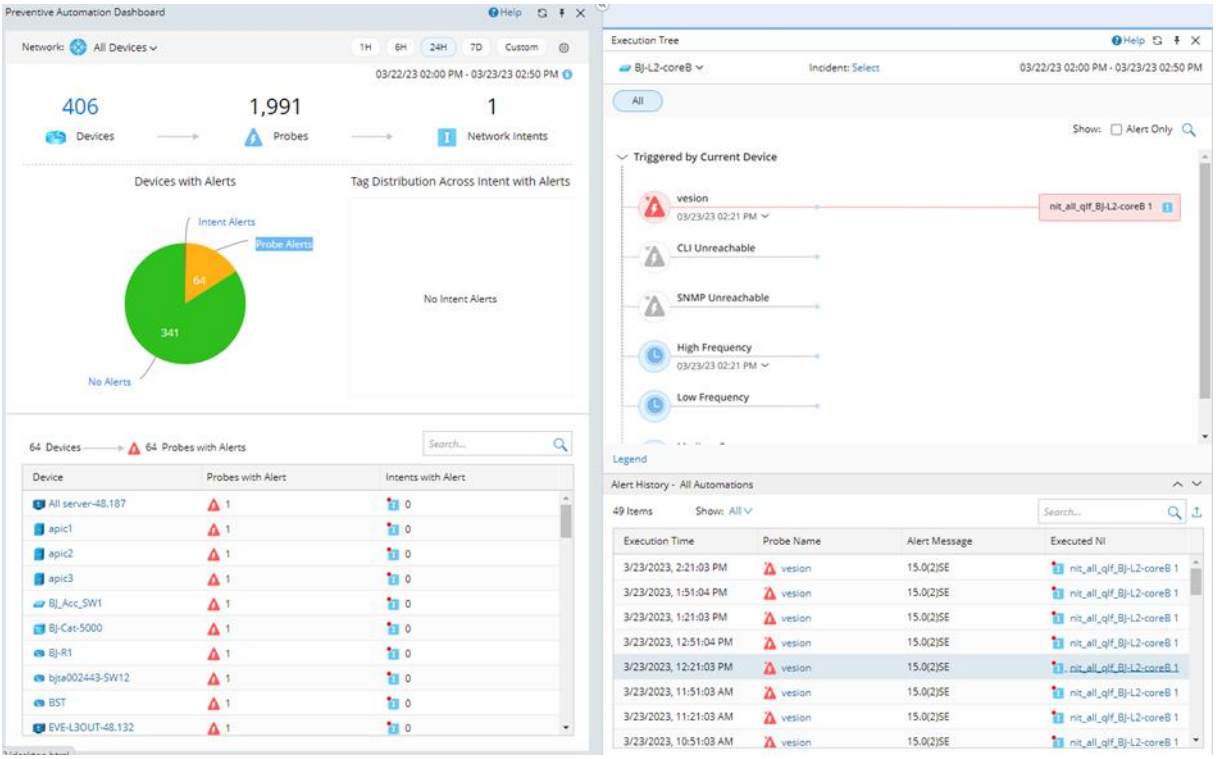


To configure the ADT executed by the Intent timer, users can install the ADT with the trigger method via Intent Timer. From the configuration window, select the intent timer to trigger the selected intents.

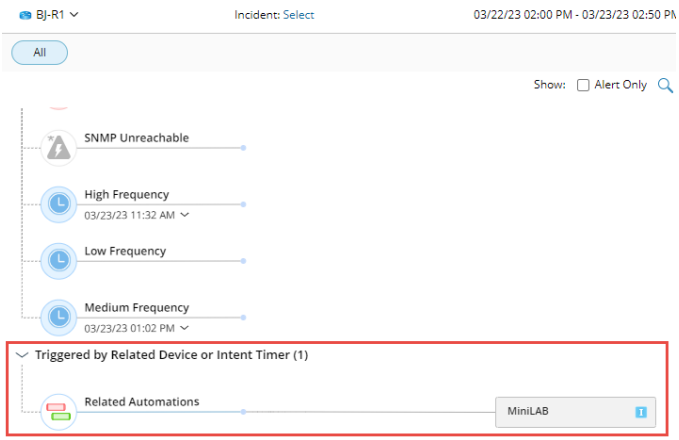


3.2.4 View Triggered Automation from PA Dashboard

PAF-based ADT results can be viewed from the PA dashboard or the **report and dashboard** new in R11.1. The following shows how the probe execution and intent trigger results can be viewed.

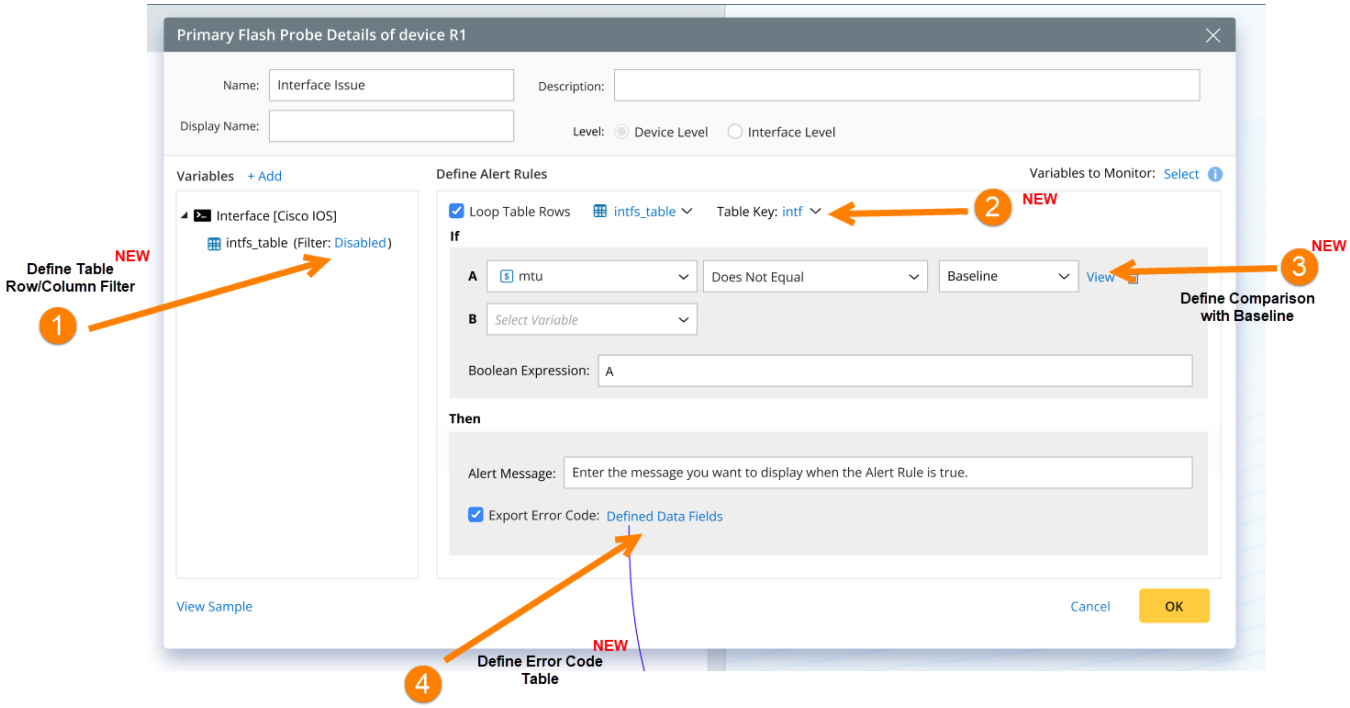


The intent results for the current device triggered by the intent timer can be found in **Triggered by Related Device or Intent Timer**.

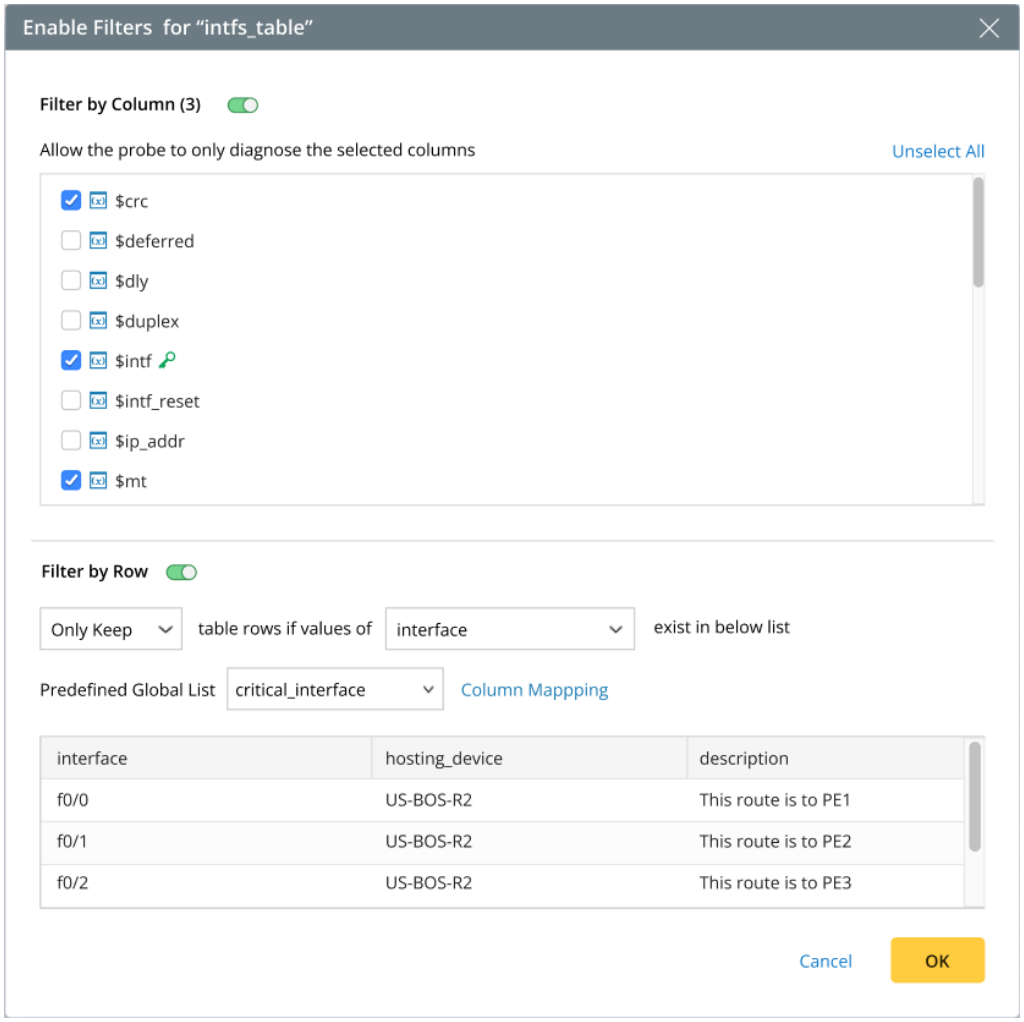


3.2.5 Probe Instance Enhancement

Monitoring Probe has been improved with the following functions in R11.1 to support the looking glass probe use case:

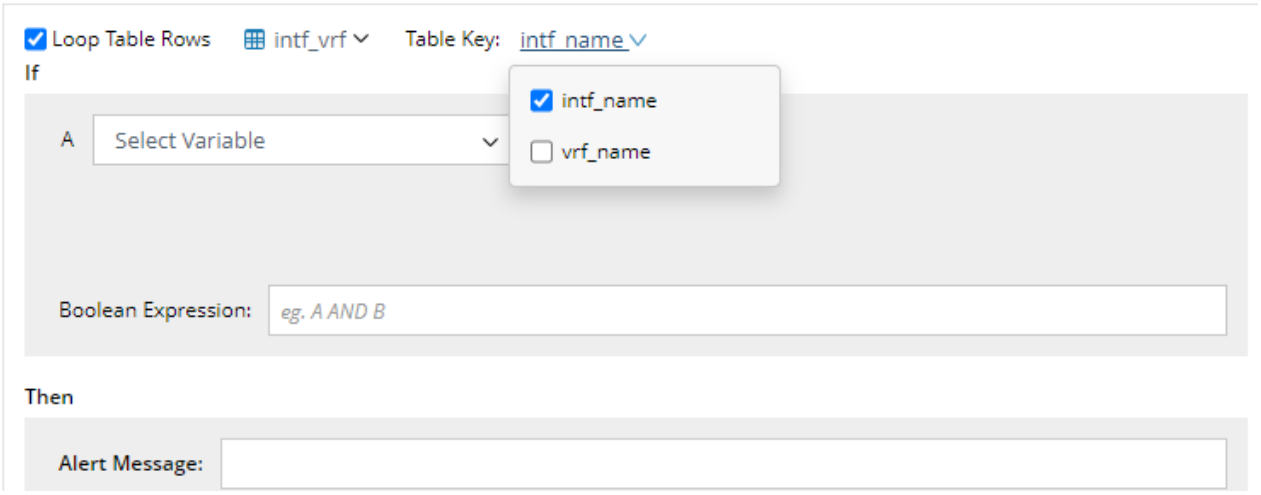


1. **Define Table row/column filter** to keep only critical contents.
    - Filter by column: Select the columns you want to keep so the probe will diagnose only certain columns.
    - Filter by row: Use the ADT table as the source for filtering the parser table. Users can keep or filter certain rows to match with ADT table contents.



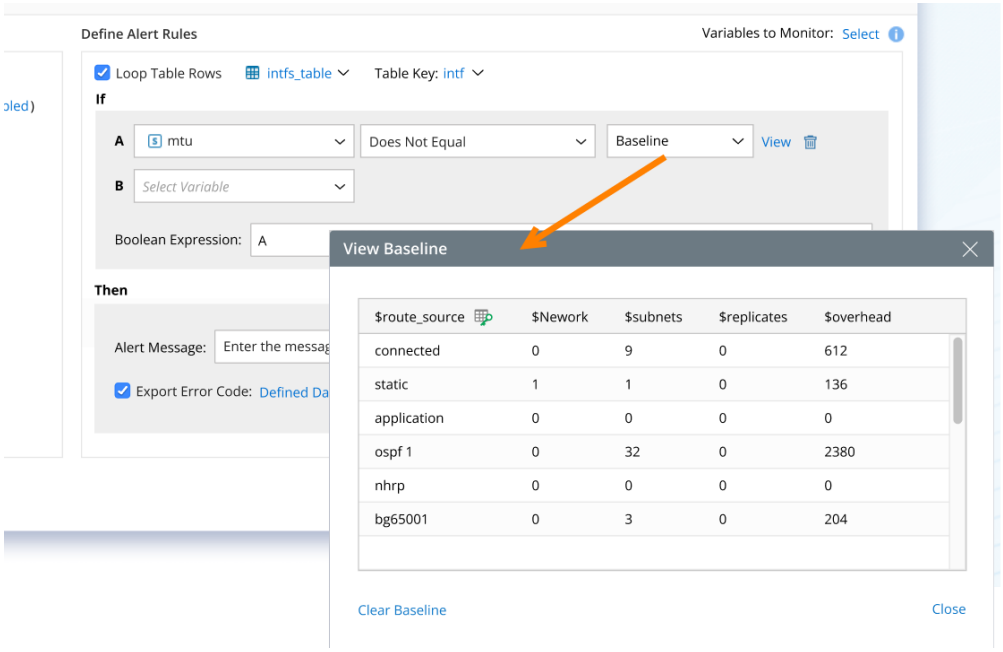
2. **Define Table key**: The table key can be defined directly within the probe. (In the earlier version, the table key comes from the parser definition).

   The table key compares the table rows against baseline/last values to identify unique rows. When looping table rows, you cannot compare the table with the baseline/last value without the table key.

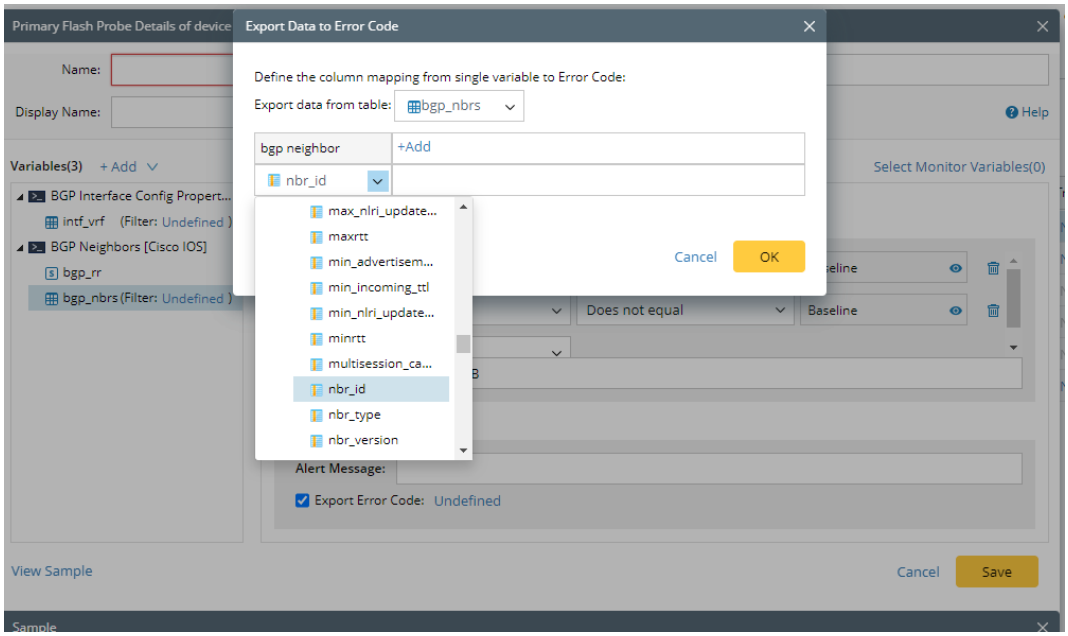


3. **Define Comparison with Baseline Logic**: use the baseline to define the comparison logic against baseline data.

   When the data is retrieved for the first time, the data will be saved and used as baseline data. The baseline data remains the same and doesn't change. If the baseline data doesn't match the current network status, you can use the Clear Baseline function to clear the existing baseline. The system will further retrieve the data and use it as baseline data.

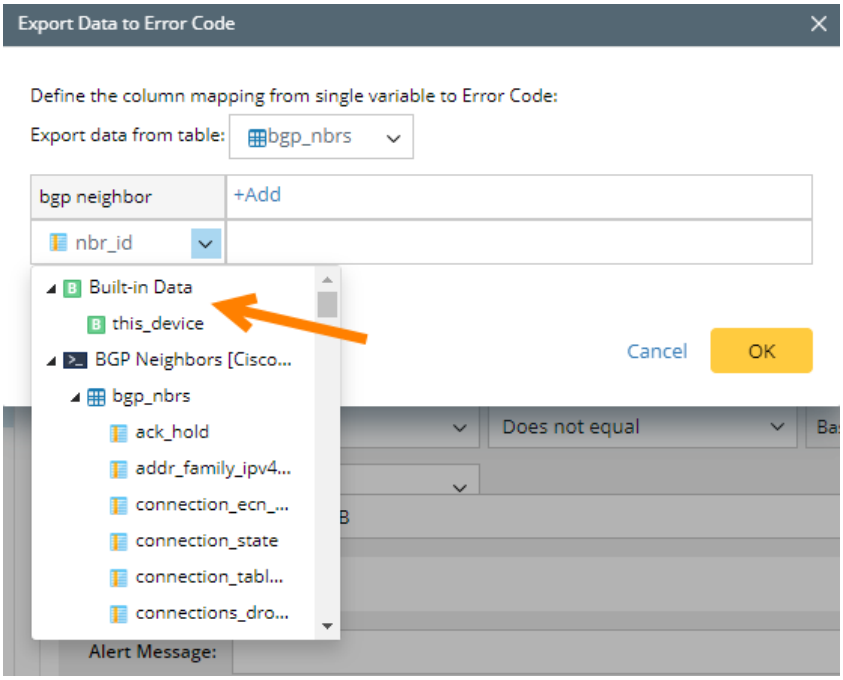


4. **Export Error Code**: Define the error code to export the error contents to trigger Intent execution within the ADT table. The error code column can be defined from the table or a single value used in the **if** condition.

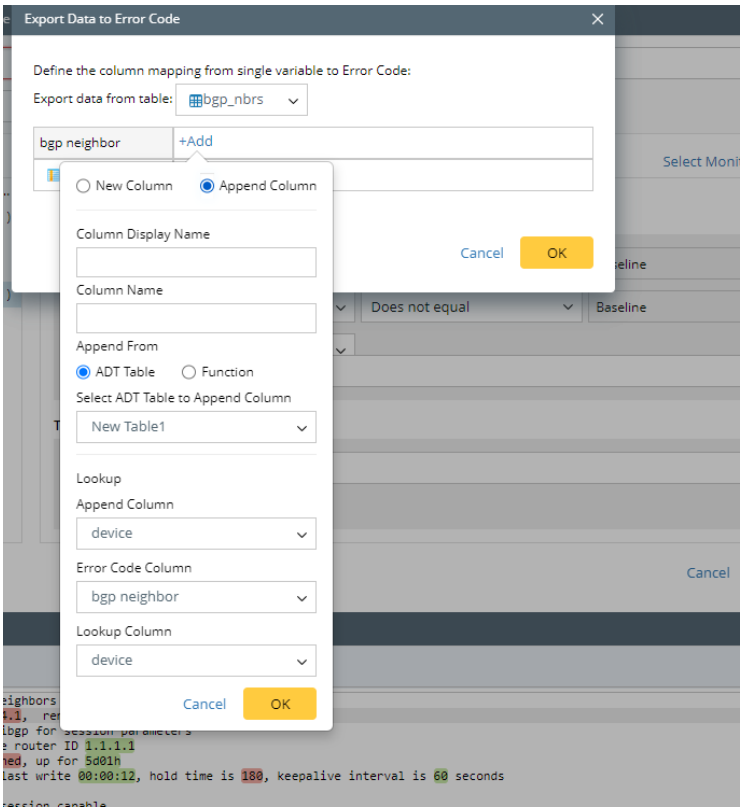


The table column can be used as an error code column besides the single value. The error code column can be further expanded with the following data types:

- Built-in data > ***this device***: this device can be exported if an alert is generated, which is very useful when there are multiple looking glass devices, and you want to define match conditions based on looking glass devices.

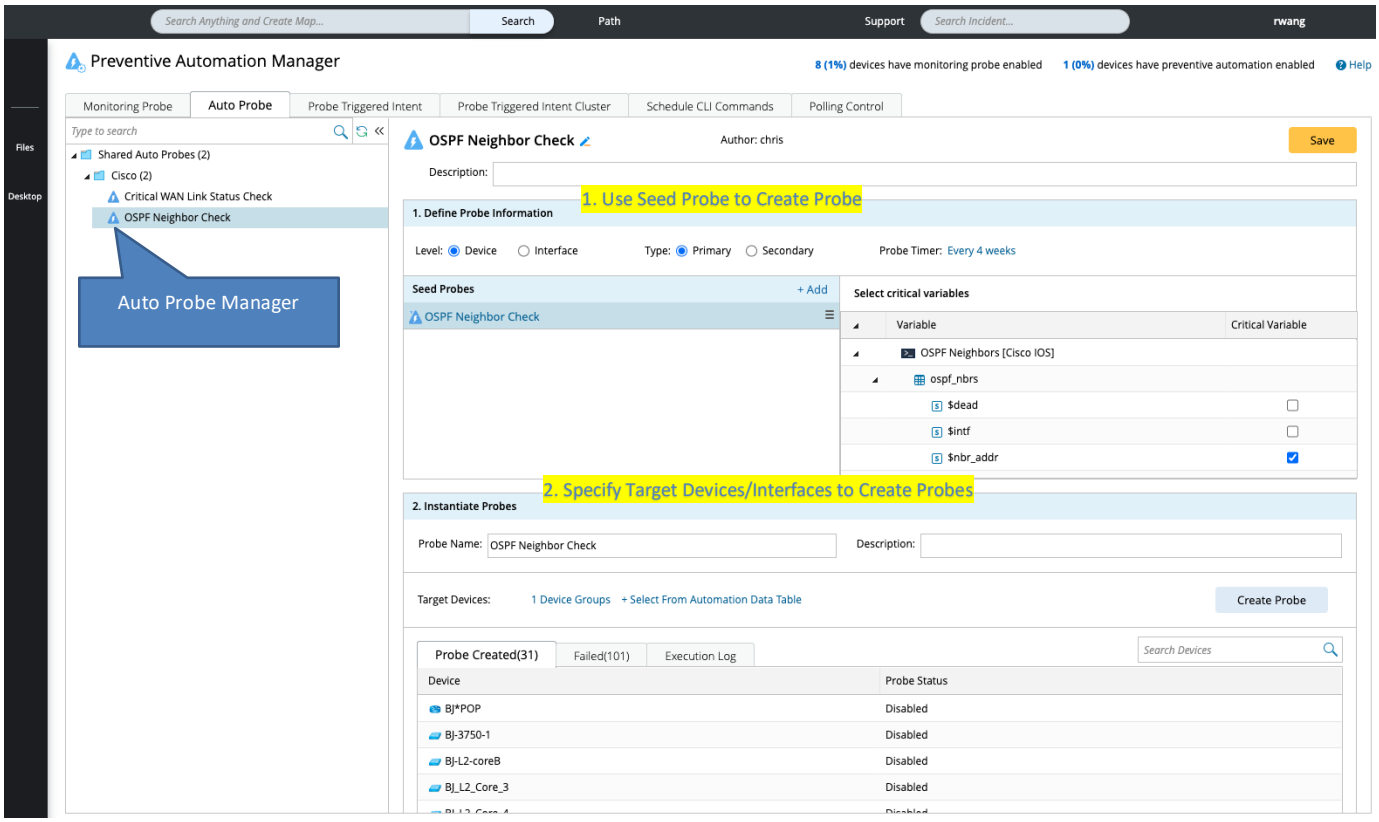


- Append column via ADT table: ADT column can be appended to the error code to expand the column.

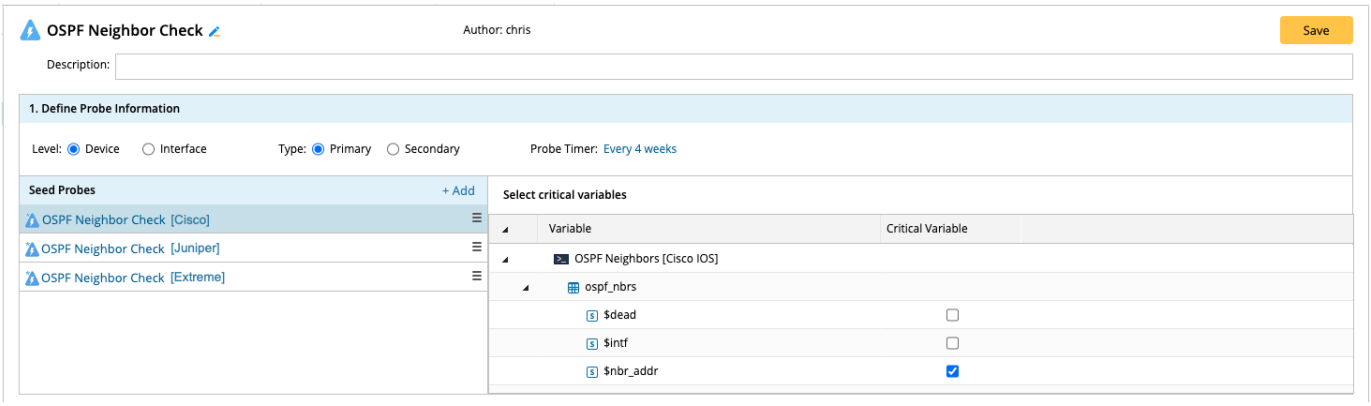


3.2.6 Auto Probe

Auto Probe provides a quick way for power users to create probes from the seed probe(s) in batches on a set of qualified devices or interfaces. Once completed, power users can use these device/interface probes to set up the ADT-based PAF.



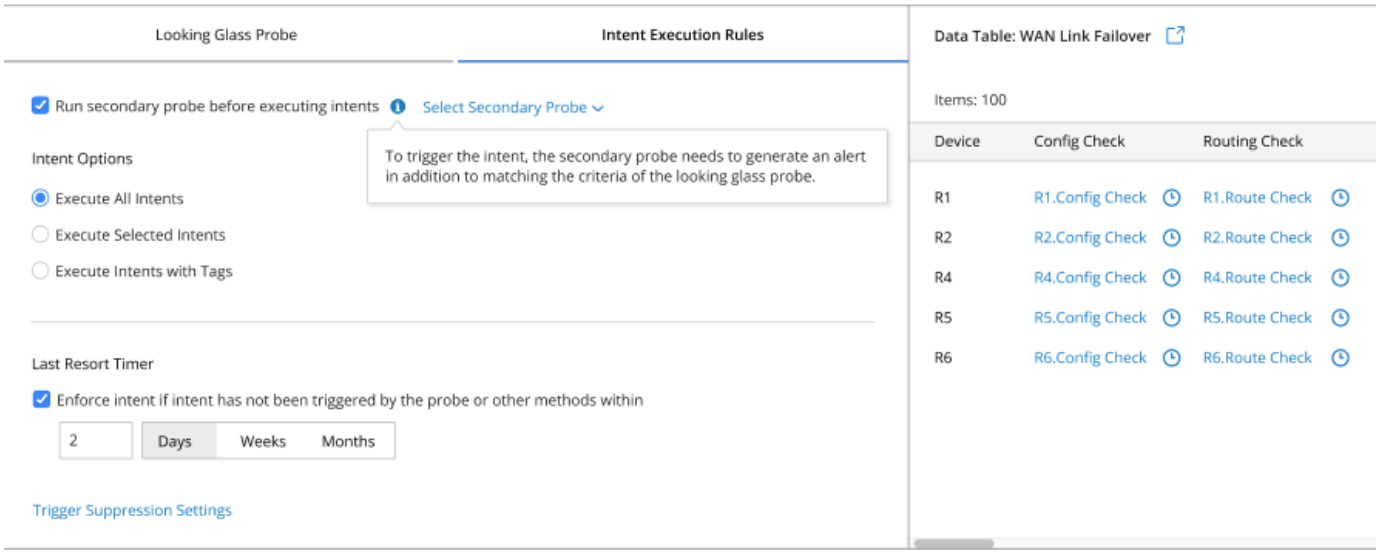
Besides selecting the seed probes and target devices or interfaces, users can specify critical variables to identify whether the probe is qualified on the target device. With the critical variables, the system can test the critical variables against the live network and find the correct seed probe for a target device.



3.2.7 Other New Features and Enhancements for PAF

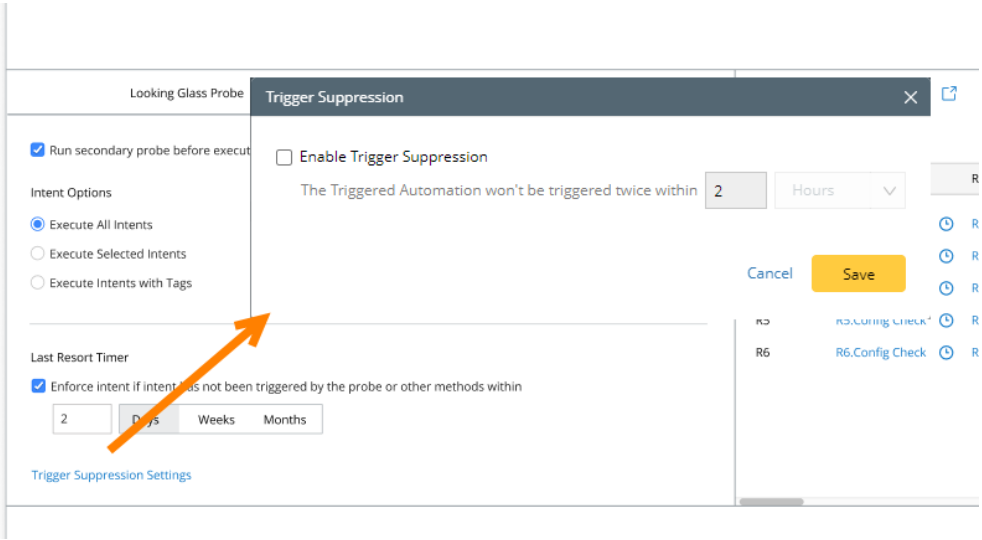
3.2.7.1 Last Resort Timer

For intents triggered by looking glass probe or primary probe, there is a chance that the probe status is healthy and probes generate no alerts for some time. To ensure that these intents are executed even if no alert is generated, enable **Last Resort Timer** and configure the condition. So if certain intents are not triggered within the defined time frame, the system will further trigger the intents to execute.



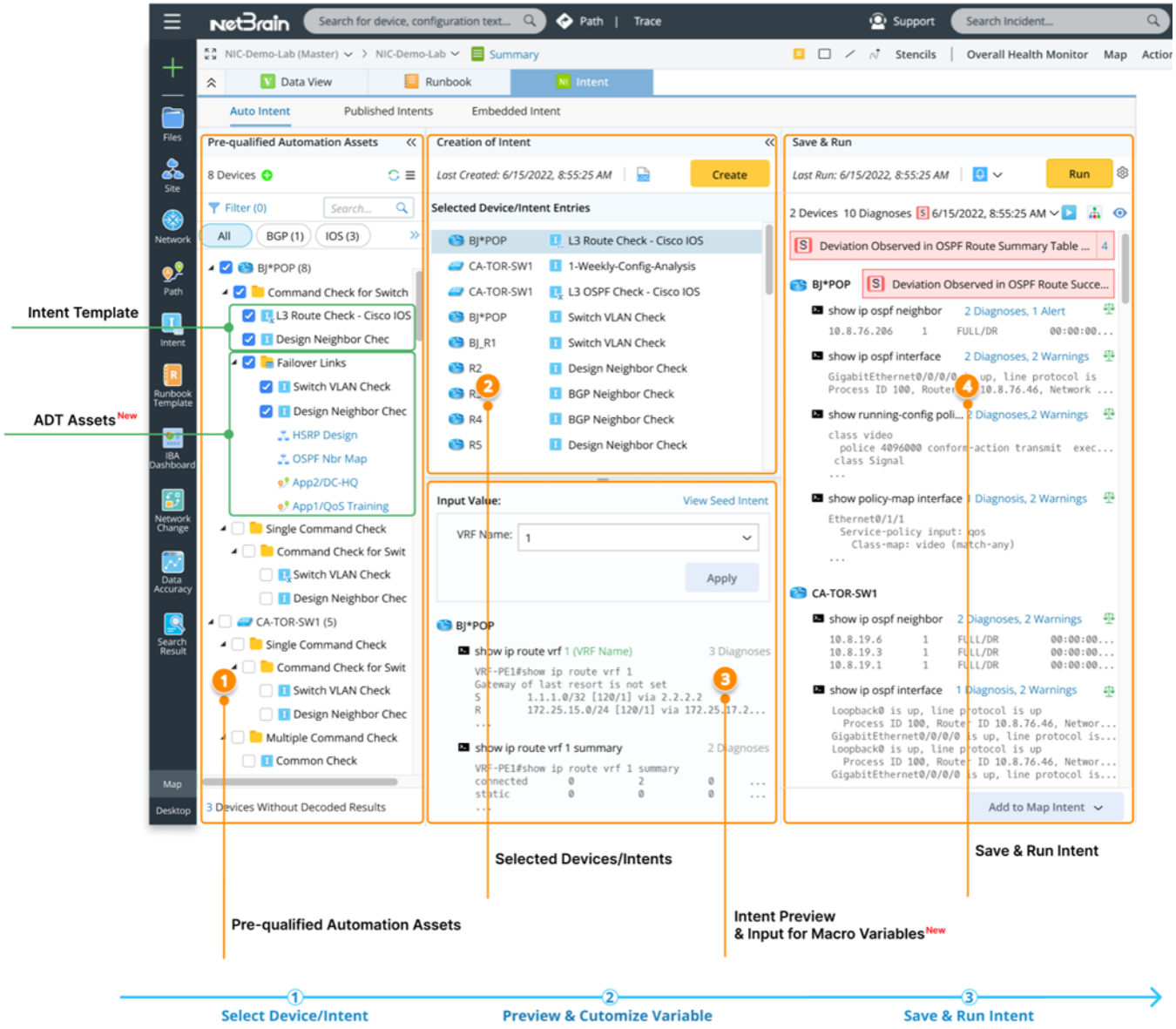
3.2.7.2 Trigger Suppression

For intents triggered by looking glass probe or primary probe, if the probe generates alerts consecutively and you don't want the intent to be triggered each time, configure the trigger suppression to specify the intents to be triggered once within the defined time frame.



3.3 Install ADT to Auto Intent

Auto Intent allows users to create and run intents on Map, which can be enabled from the Auto Intent Tab in Intent-Based Automation Center (IBA Center, Intent Library in the early releases). R11.1 makes the following enhancements for the Auto Intent:

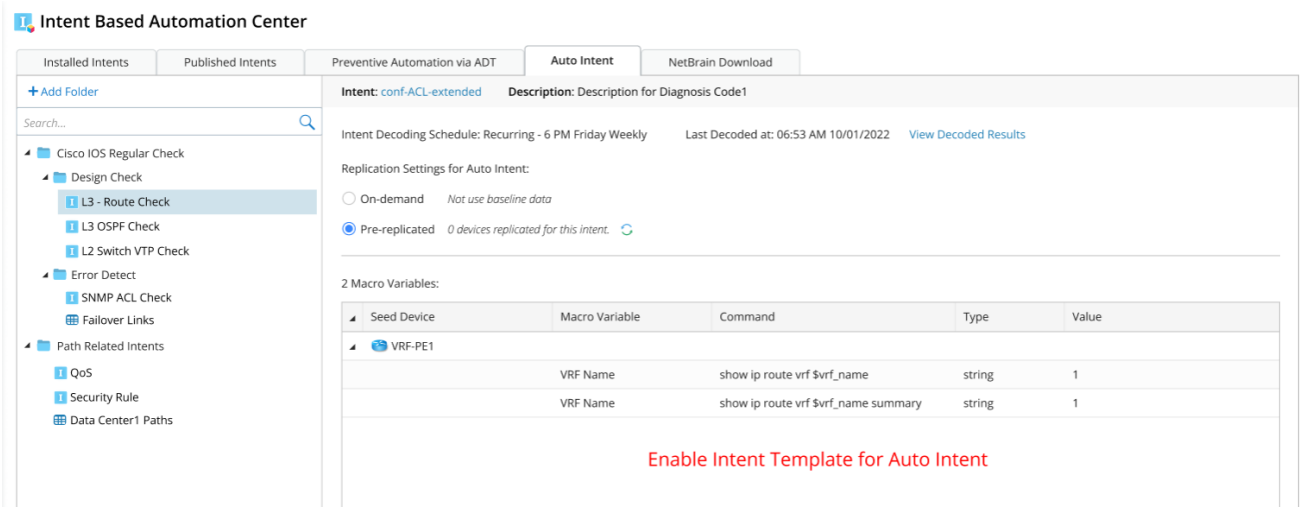


1. **Pre-qualified Automation Assets:** A folder structure to display and organize ADT's network intent templates and automation assets (intent/map/path). Users can view and use the items by category.
    - **Intent Template:** The intent templates enabled for Auto Intent are listed in the left pane.
    - **ADT Assets:** Intents/Maps/Paths of the enabled ADT are listed in the left pane.
2. **Selected Device/Intents:** Show the items selected from the left pane.
3. **Intent Preview & Input for Macro Variables:** Preview items highlighted or selected in the left pane and the items highlighted in the **Selected Device/Intents** area to facilitate network intent selection; input values for macro variables of the target device to create proper intent.
4. **Save & Run Intent:** Run intent and view the execution results, then add the created intent as map/path/common intent for further troubleshooting.

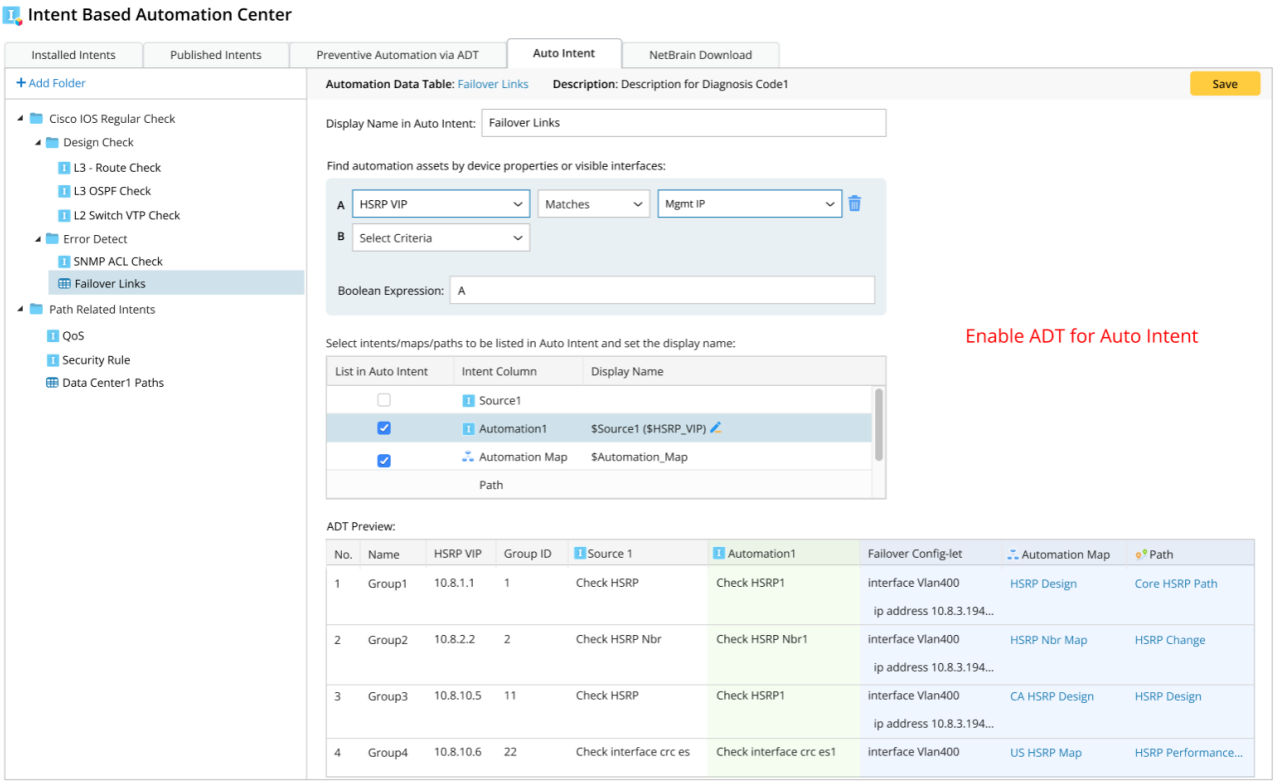
3.3.1 Auto Intent Tab in IBA Center

Generally, power users will operate on the Auto Intent tab in IBA Center.

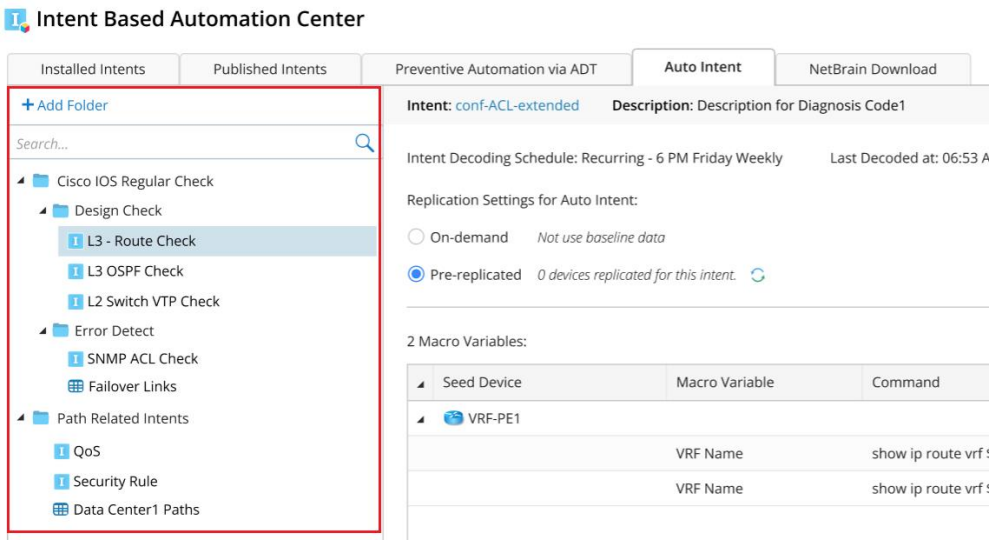
- Enable intent template for Auto Intent so that end users can replicate intent templates and customize the macro variables if necessary to create appropriate intent for map devices.



- Enable ADT (Automation Data Table) Intents/Maps/Paths for Auto Intent so that end users can find related automation assets (Intent/Map/Path) for map devices from pre-defined ADT to understand and troubleshoot network problems.

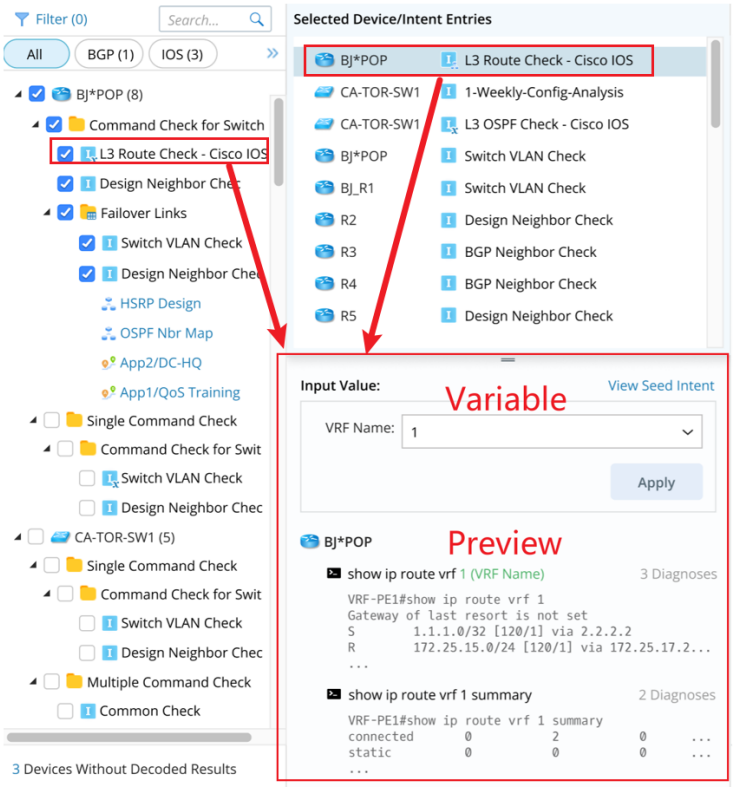


- Build a folder structure to display assets in Auto Intent Pane with a clear and meaningful organization so that end users can understand and operate on the automation assets.



3.3.2 Use Auto Intent to Create Intent for Map Devices

Auto intent supports replicating the intent template with customizable macro variables. With this functionality, end users can replicate desired intent from an intent template by simply setting the macro variables.

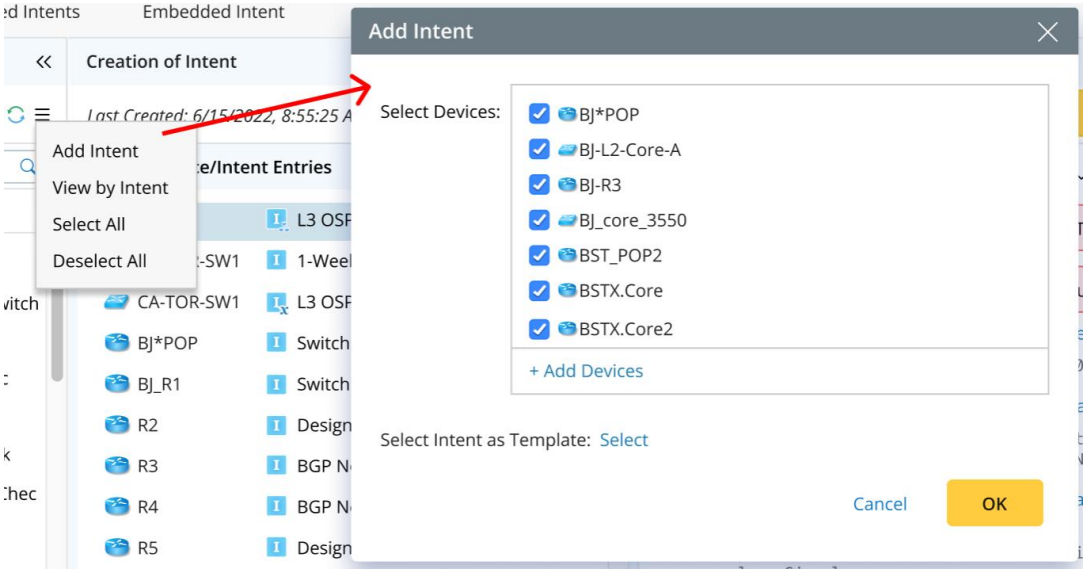


- Users can preview the data of the intent template via the following two operations:
  - Click (unselect is OK) an item in the Pre-qualified Automation Assets pane to preview its macro variables and command data.
  - Click an item in the Selected Device/Intent area to preview its macro variables and command data.
- If the intent template has macro variables defined, users can change the value, which will affect the replication results.

3.3.3 Use Intent Template Not Included in Auto Intent

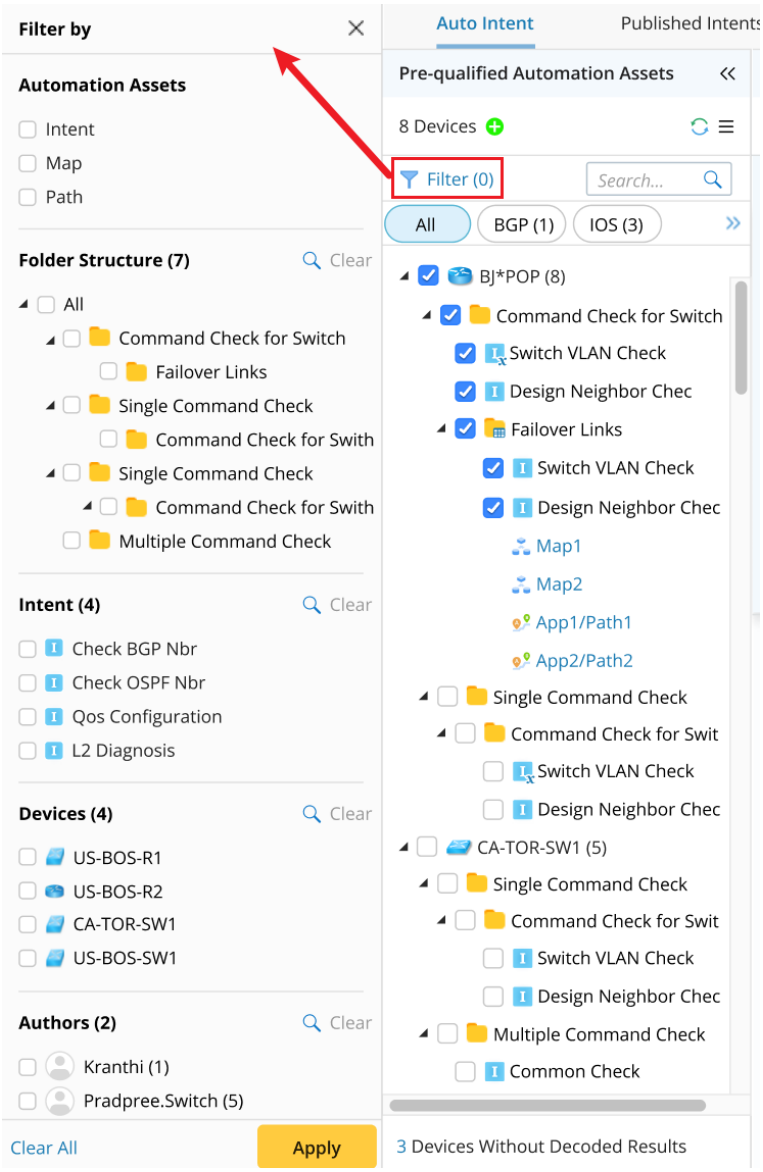
Apart from the intent templates already listed in Auto Intent, users can use additional ones not installed and decoded in IBA Center, giving users more flexibility to use intents in Auto Intent.

Select one or more devices (Devices not included in the device list can also be added and selected) and one intent template to pair and add to the Selected Device/Intent Entries area.



3.3.4 Filter Intent Template/ADT in Auto Intent

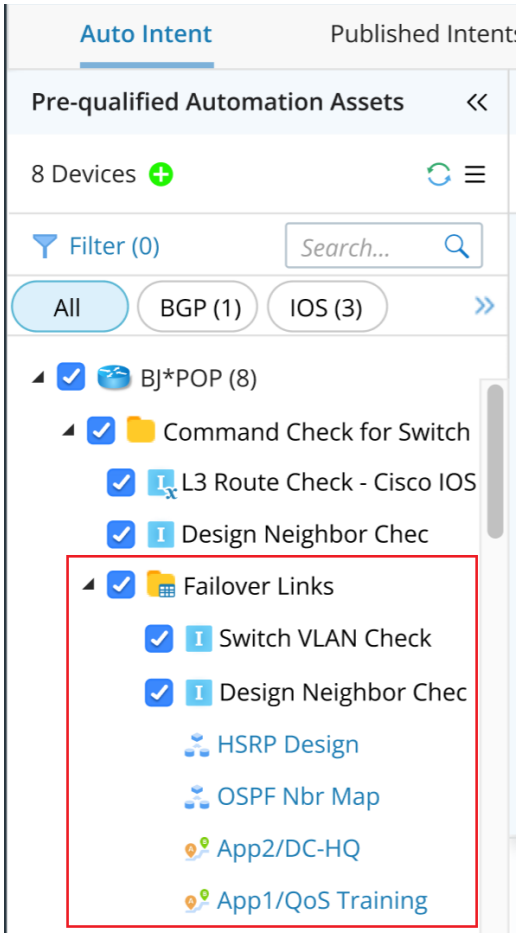
In R11.1, the filter function of Auto Intent is improved to be more intuitive and functional. When there are too many items, users can find the ones needed by filtering the items. An independent pane will appear to the left after clicking Filter (0), making it easier to define the filter conditions centrally. The improved filter function leads users to pay more attention to the automation assets and network objects enabled for Auto Intent.



3.3.5 Use ADT Automation Asset Via Auto Intent

To fully understand and troubleshoot an issue, users normally need to analyze the related maps and paths besides intents. By enabling ADT assets (Intent/Map/Path) to be used in Auto Intent, users can find automation assets for map devices, e.g., device-related failover links and the map, device-related path, and the path intents.

With the available ADT assets in Auto Intent, users can complete the following tasks:



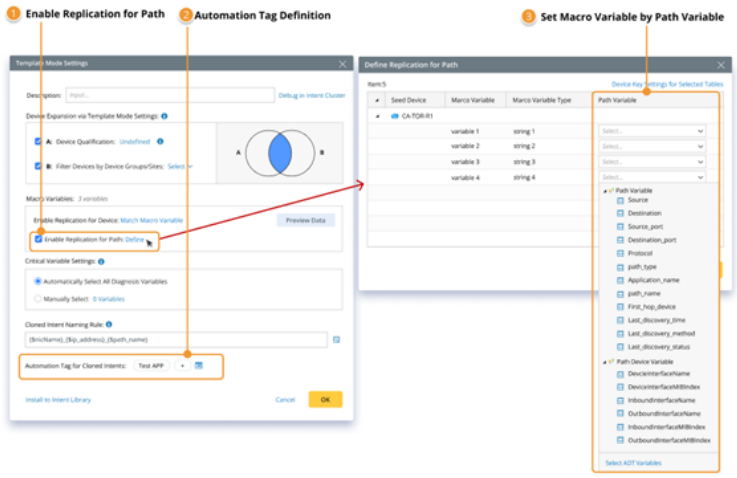
- Use intents from ADT to create an intent for troubleshooting.
- View a map and use the map for troubleshooting.
- View a path and use the path for troubleshooting.

3.4 Use ADT to Replicate Path Intent

R11.1 made the following improvements on path intent:

- Path intent replication settings in the intent template (NIT): The system allows defining path macro variables or selecting ADT data sources as macro variables for an intent template to prepare for creating

path intents. Specifically, path variables and path device variables are used as the values of the macro variables of the intent template.

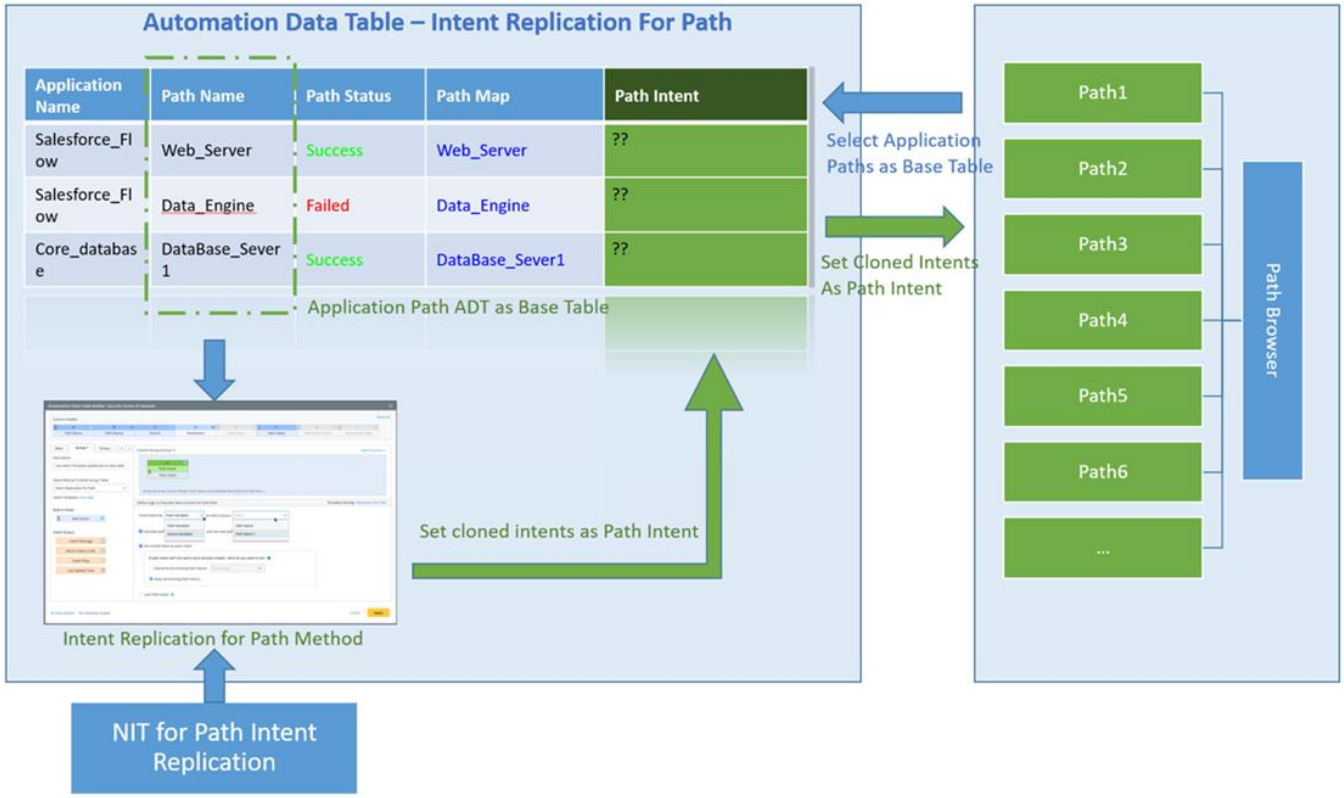


- Batch Path Intent Replication from ADT New: With the method “Intent Replication for Path” provided for building column groups in ADT, multiple path intents can be delivered from ADT with the data in ADT for running troubleshooting diagnosis.
- On-demand Path intent replication from path browser/Auto Intent [New]: Users can set to replicate path intents by selecting the intent template from an open path or Auto Intent. With the new improvements made on path intent replication, one path can have multiple path intents.

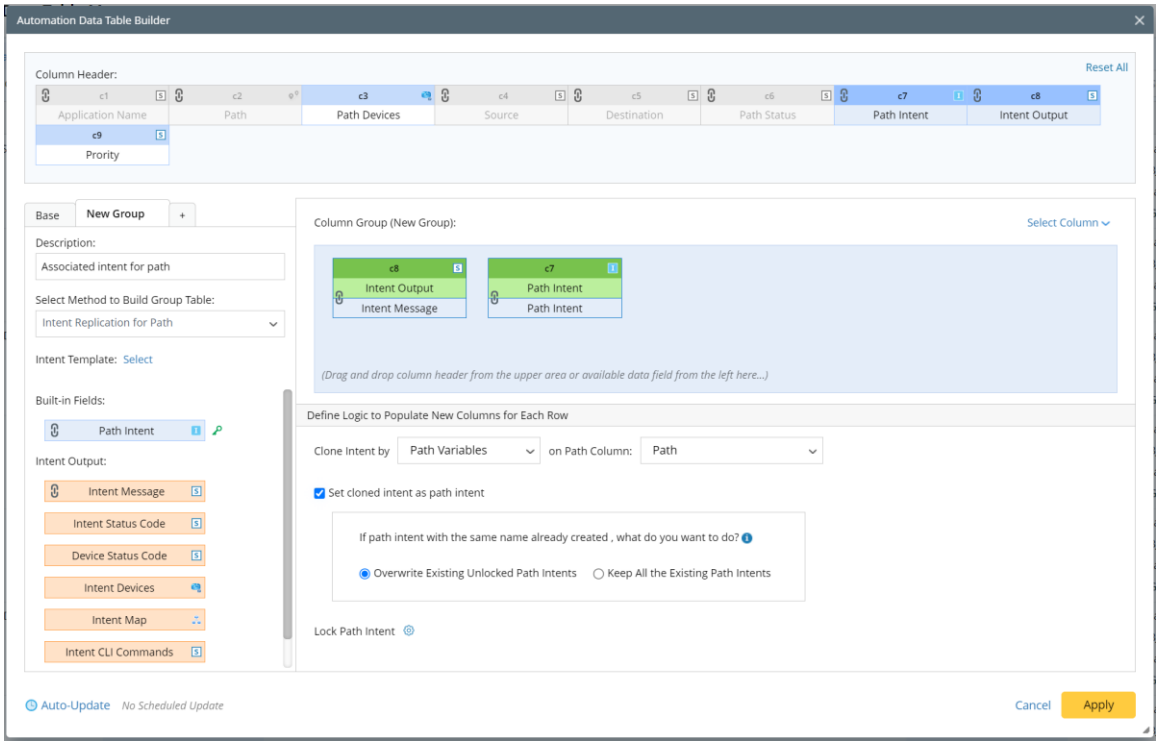
3.4.1 Batch Path Intent Replication from ADT

In R11.1, the system supports creating path intents with the new intent replication for path service in ADT.

The following diagram shows the logic of cloning path intents in batch via intent template with the data source in ADT.



In the Intent Replication for Path Service, ADT serves as a tool to provide data to replicate path intents as scheduled. Specifically, users can add a group column to ADT via the "**Intent Replication for Path**" method. The available data fields include path intent and data fields of the path intent details. With the data in ADT, users can further set the variables for path intent replication. The created intents can be saved as path intents for future use.



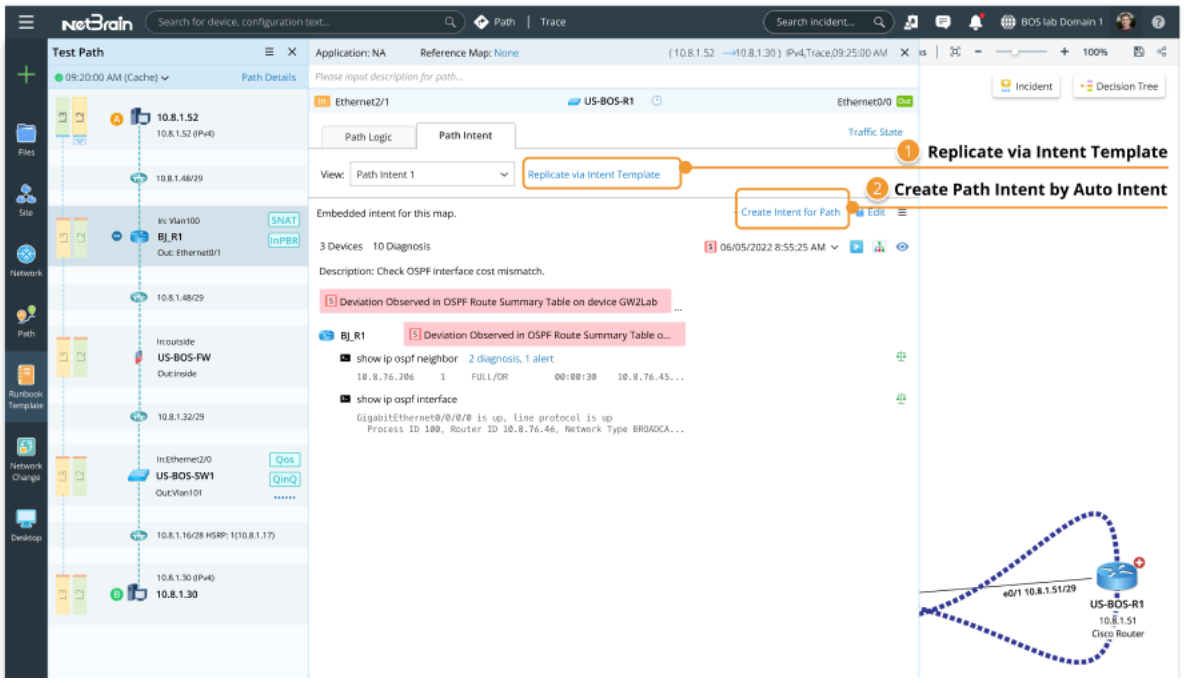
The path parameters can be passed to the Macro Variables of the seed NIT to clone a path NI.

No.	Application Name	Path	Path Devices	Source	Destination	Path Status	Path Intent	Intent Output	Prority
1	Salesforce Server	App1	Bj-3750-2 Bj-Arista-1 Bj-Arista-2	10.10.10.1	192.168.1.1		Monitor Path Health	para equals to baseline Paragraph1 of BJ_L2_Core_3 h para equals to baseline Paragraph1 of GW2Lab has ...	H
2	QOS Path	NYC-to-BOS	Bj-Avaya-1 Bj-Avaya-2 Bj-Cat-5000 Bj-L2-Core-A...	10.8.8.162	10.8.1.26		Monitor Path Health	para equals to baseline Paragraph1 of BJ_L2_Core_3 h para equals to baseline Paragraph1 of GW2Lab has ...	M
3	Traditional Data Center	App1	Bj-Avaya-2 Bj-Cat-5000 Bj-L2-Core-A Bj-L2-coreB...	10.8.1.4	10.8.3.140	Failed	Monitor Path Health	para equals to baseline Paragraph1 of BJ_L2_Core_3 h para equals to baseline Paragraph1 of GW2Lab has ...	H+
4	QOS Path	DB Backup	ASA-AA/admin/act ASA-AA/admin/stby BJ_L2_Core_3 LA.DIS.1...	10.8.1.4	10.8.3.130		Monitor Path Health	para equals to baseline Paragraph1 of BJ_L2_Core_3 h para equals to baseline Paragraph1 of GW2Lab has ...	M
5	POC	MPLS_DMVPN_PATH	BJ-R2 BJ-R3 BST,POP1 qapp-c3560-1	10.8.2.19	10.8.5.11		Monitor Path Health	para equals to baseline Paragraph1 of BJ_L2_Core_3 h para equals to baseline Paragraph1 of GW2Lab has ...	H-
6	Traditional Data Center	App2	ASA-AA/context1/stby BJ*POP BJ_L2_Core_3 BJ_core_3550	10.8.1.4	10.8.3.196	Failed	Monitor Path Health	para equals to baseline Paragraph1 of BJ_L2_Core_3 h para equals to baseline Paragraph1 of GW2Lab has ...	H
7	QOS Path	NYC-to-SMF	Emu_NB_NYC_MGMT F5-MGMT	10.8.8.162	10.8.2.14		Monitor Path Health	para equals to baseline Paragraph1 of BJ_L2_Core_3 h	L

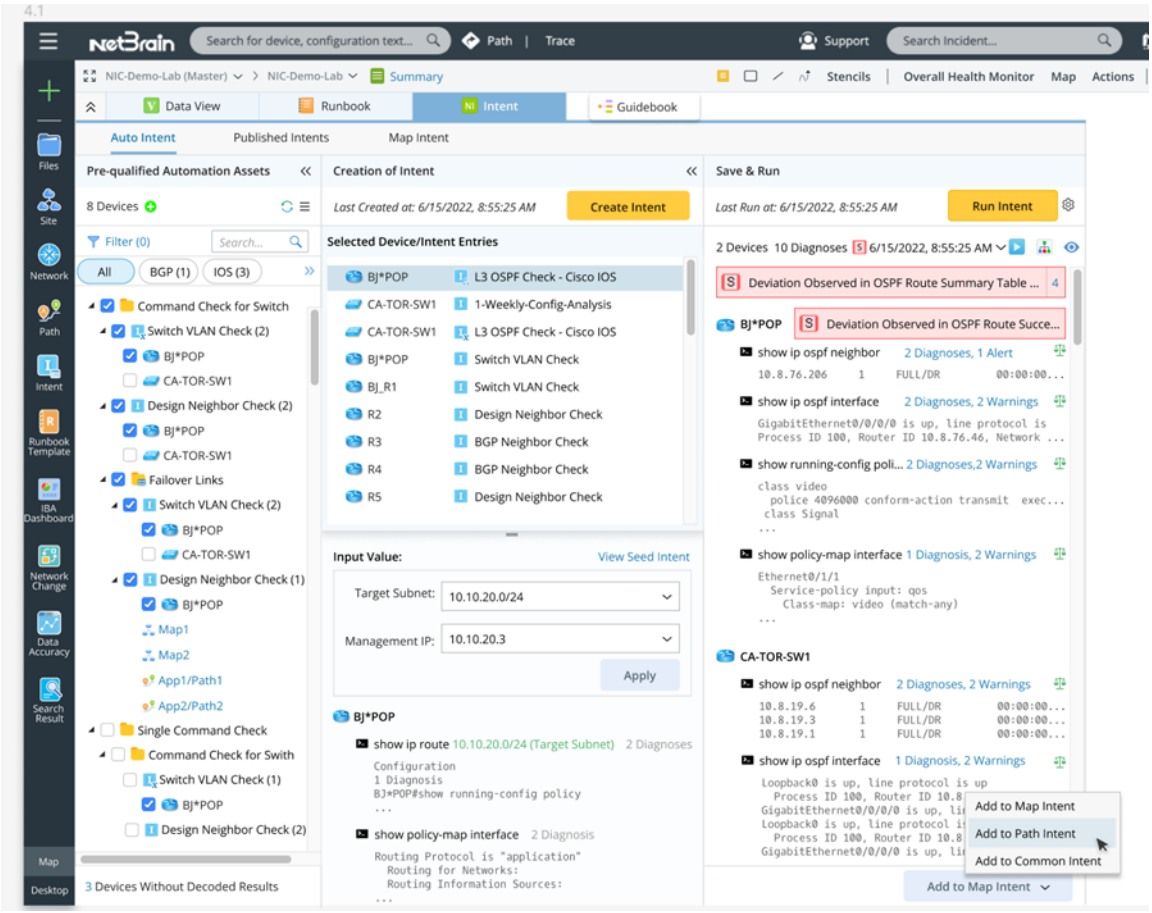
3.4.2 On-demand Path Intent Replication from Path Browser/Auto Intent

In the previous versions, a path is associated with only one path intent. In R11.1, multiple path intents can be created and associated with one path, which enhances the capability of executing path intents for troubleshooting network issues from a path. All the path intents of a path are included in a "**View**" list in the Path Intent Pane. There are two ways to create a path intent:

- Replicate the path intent via Intent Template from the **Path Detail Pane**:



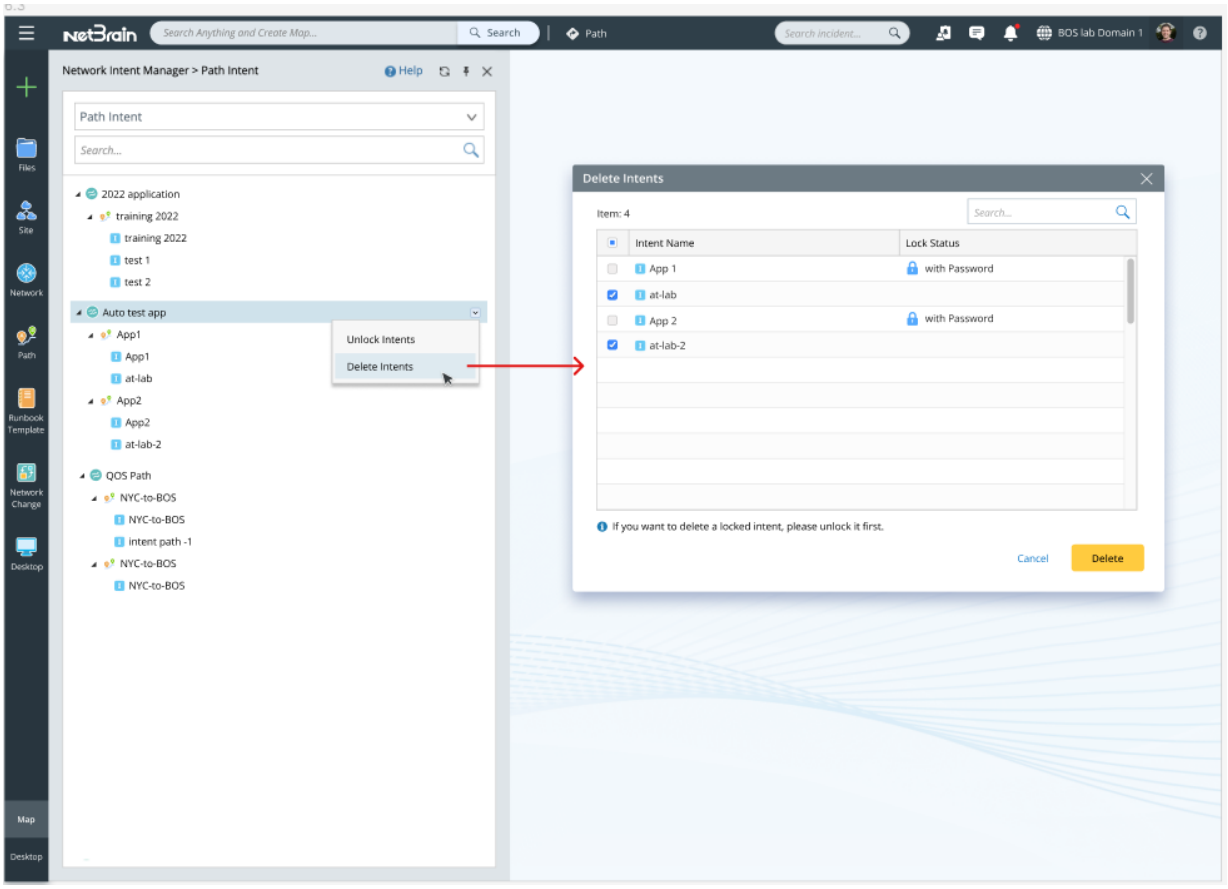
- Open **Auto Intent** to create path intent for a path: After a path intent is created with the automation assets in Auto Intent, the created intent can be associated with a path and saved as its path intent.



3.4.3 Delete/Unlock Path Intents in Batch

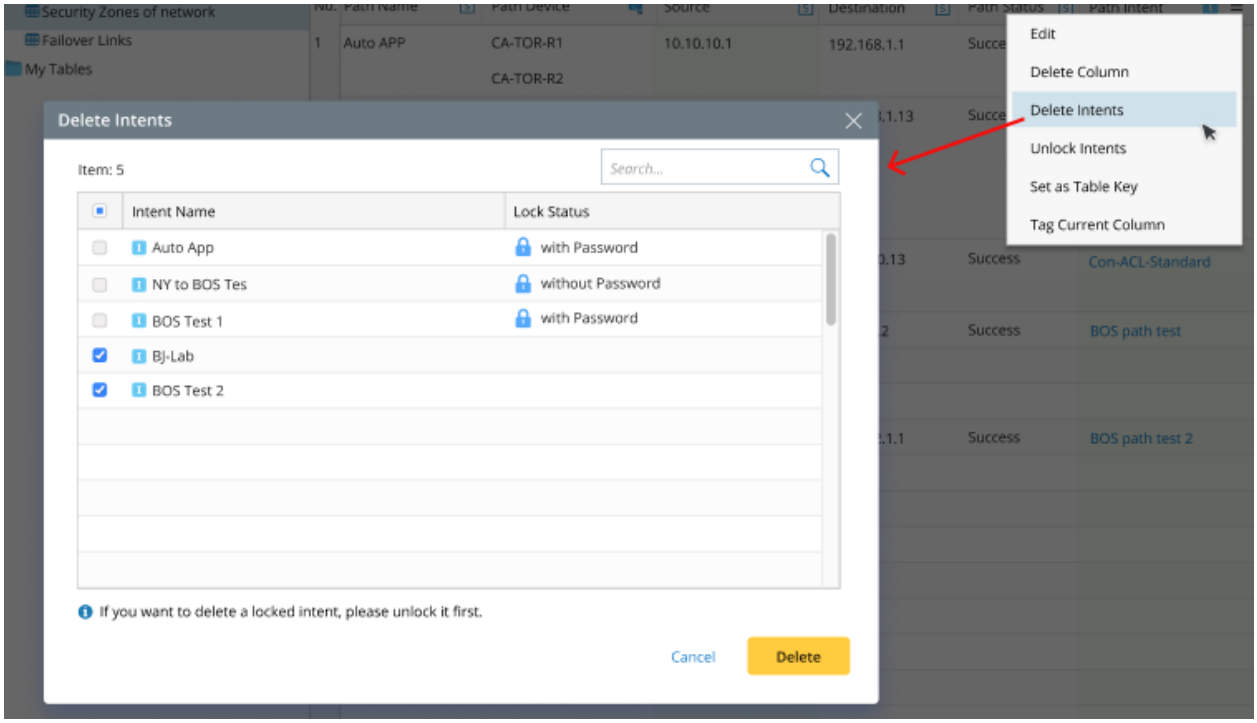
The system offers several options for deleting/unlocking path intents in batch.

- Batch Remove/Unlock Path Intents from Network Intent Manager: In **Network Intent Manager > Path Intent**, each path forms a folder to include all the associated path intents. Users can delete/unlock multiple path intents at one time in the pop-up Dialog opened from the drop-down menu.



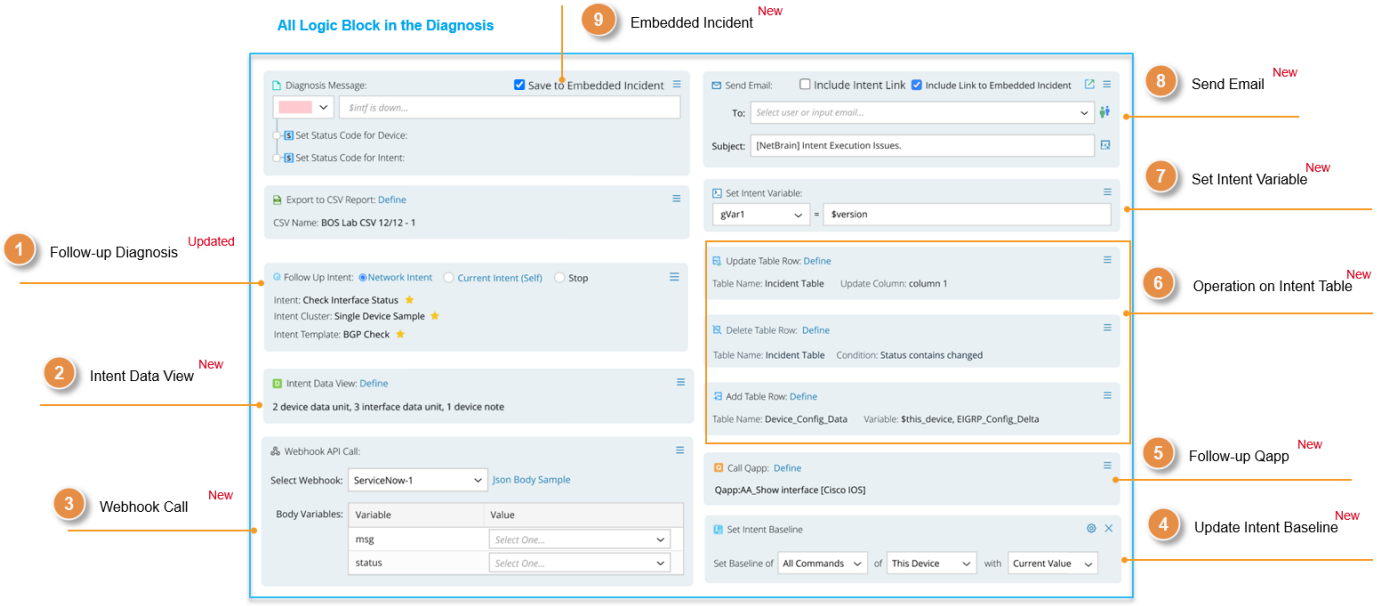
- Batch Remove/Unlock Path Intents from ADT Manager: for an ADT containing Path Intent Column, users can delete/unlock multiple path intents at a time in the pop-up dialog from the drop-down

menu of the Path Column.



4 Network Intent Enhancements

R11.1 has updated no-code intent capture, replication, and execution in many ways, including:



- **On-demand Replication of Intent (NIT)**
  - Support replication intent on-demand for auto-intent, TAF, PAF, Bot, and follow-up diagnosis.
- **Enhancements on Intent's No-code Programmability**
  - Use intent to build map and data view **2 (this number refers to the number in the picture)**
  - Embedded Incident **9**
  - Follow-up diagnosis with intent template **1**
  - Follow-up diagnosis with self **1**
  - Follow-up diagnosis with ADT (automation data table) **1**
  - Use ADT as the database for intent **6**
  - New operation on intent table **6**
  - Control the update of intent baseline by logic **4**
  - Intent Variable and Intent diagnoses **7**

- **Enhancements on Intent's Notification**
    - Send an email with payload from within intent [8]
    - Send data to 3rd party solution via webhook API [3]
    - Use Qapp as a follow-up action [5]
- **Other enhancements on intent:**
    - Scheduled execution of intent directly
    - Build intent on API-based network including cloud and SDN
    - Simplified intent debug
    - Export CSV report to Files
    - View diagnosis messages
    - Intent View UI enhancements

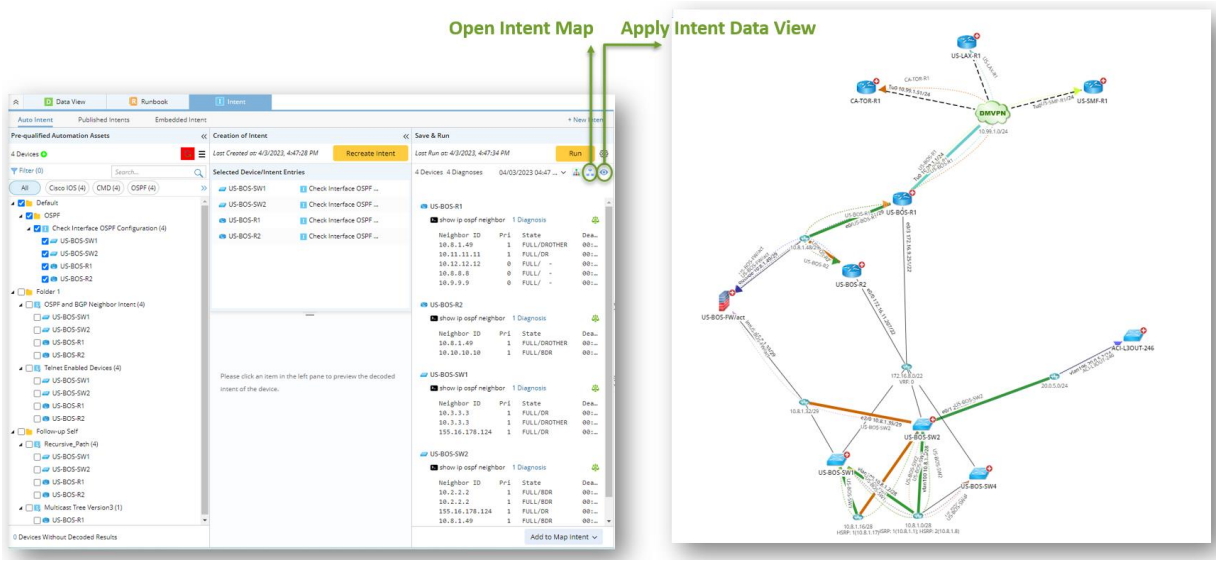
R11.1 introduces a new concept called **'Diagnosis Logic Block**' or **'Logic'**. The previous diagnosis logic is separated into 3 types of Logic: **diagnosis note & status code, export CSV,** and **follow-up intent**. R11.1 adds new logic: **Intent Data View** and **Webhook API Call**.

4.1 Create Intent Data View and Intent Map

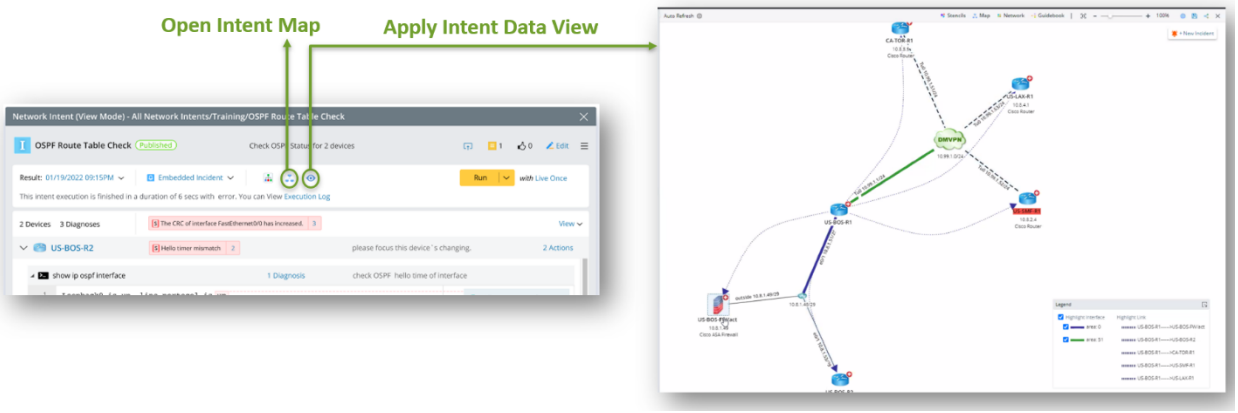
Map and data view is the foundation for troubleshooting network problems. R11.1 improves the automatic creation ability of the intent map and adds a new concept, **intent data view**, to display the results of intent on the map, such as the network design and the intent status.

- View Intent Map and Intent Data View (End User)

Under the Intent pane of a map, end users can open an intent map and the data view for the published, embedded, or auto intent.

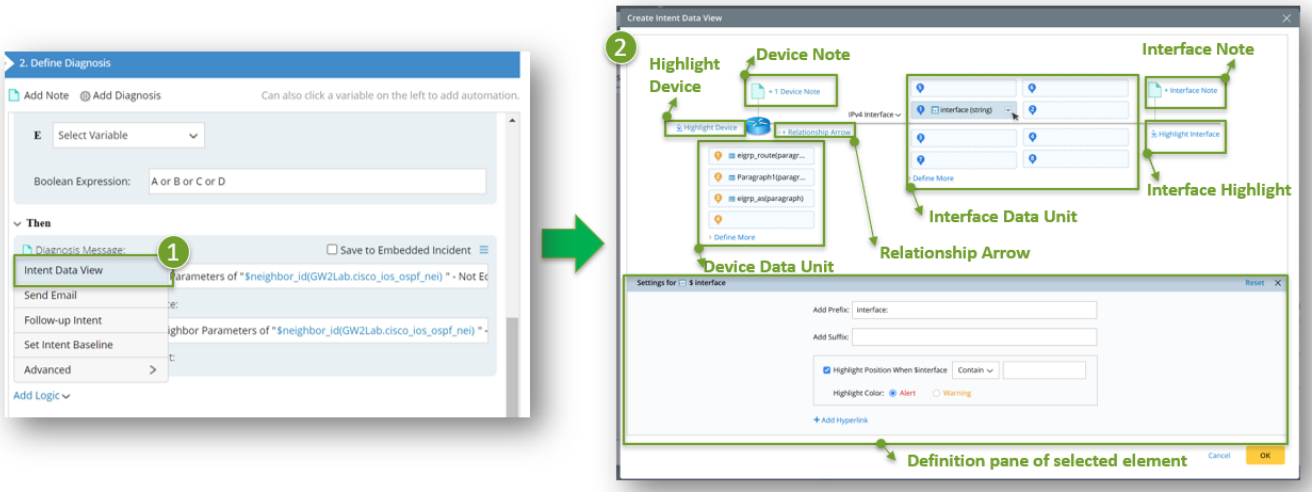


Users can also open an Intent from the Intent Manager and then Intent Map or Intent Data View from within the Intent.

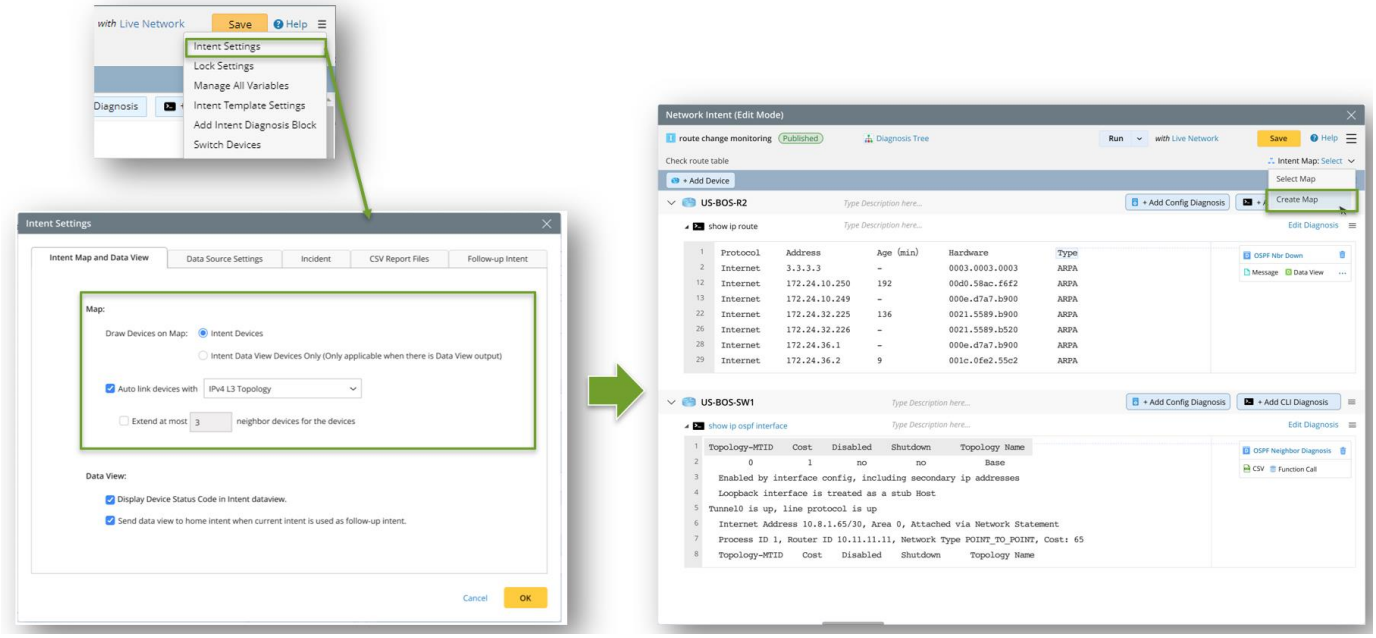


- Open Intent Map: if the intent map does not exist, the system will create one based on the Intent Map Settings.
- Apply Intent Data View: Click the eye icon to apply the intent data view to the current context map. If any device used by Intent Data View is not on the map, the system will draw it on the map.

- Build Intent Data View: Users can add a new logic, '**Intent Data View**', in the intent diagnosis and define the intent data view.



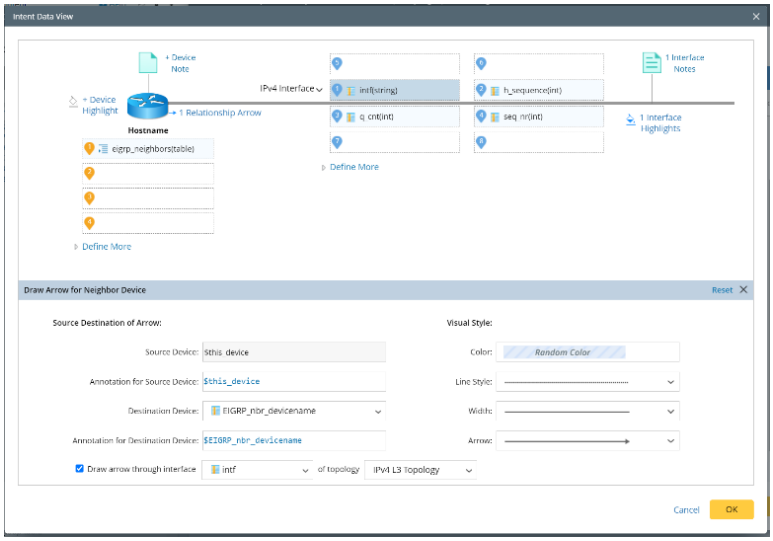
- Create Intent Map: If the intent map doesn't exist, users can define the Intent Map Settings for the system to create one and then manually optimize the map layout and annotation.



4.1.1 Build Intent Data View

The Intent Data View is designed to display the diagnosis data and results on an intent map or any map, so users can view the network design or monitor the network status on the map.

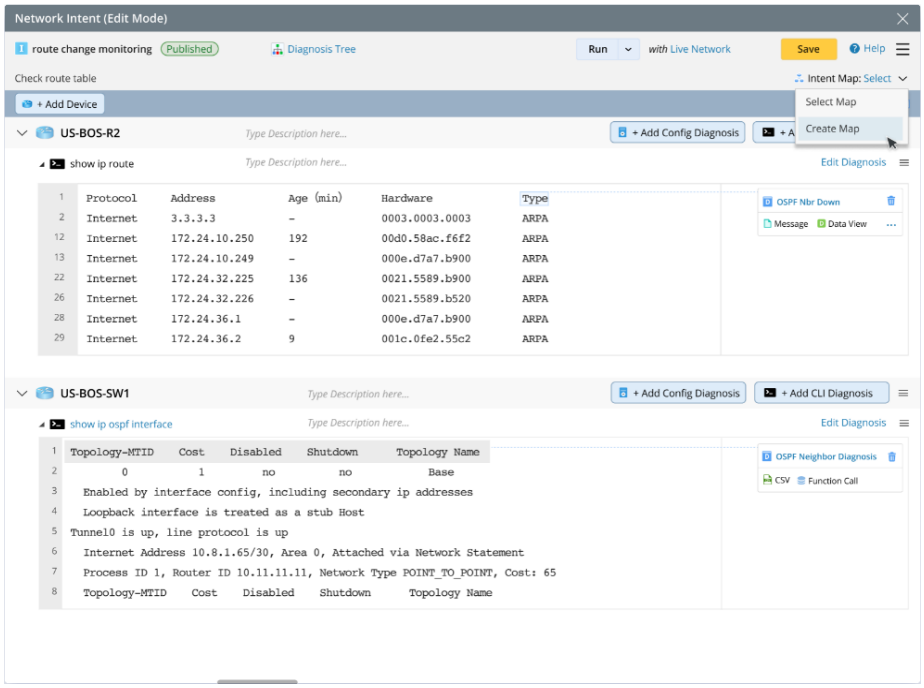
Users can add a new logic, '**Intent Data View'**, in intent diagnosis and define the intent data view with the diagnosis data. Besides defining the device and interface data units, users can define the highlight and notes and draw a relationship arrow between devices.



4.1.2 Build Intent Map

An Intent Map is a map embedded in an Intent to show the network relationship between devices of this Intent. It can be created in two ways:

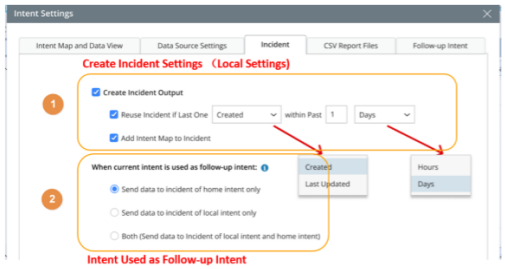
- Manually create a map first, then associate this map with an intent.
- Create an Intent Map automatically by the Intent Map Settings

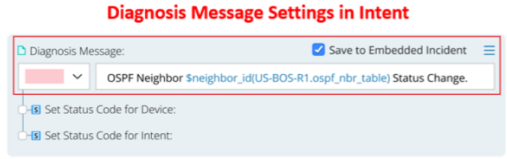


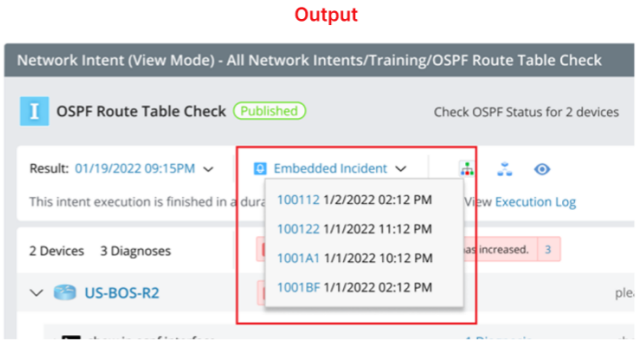
4.2 Embedded Incident

The embedded incident in intent supports:

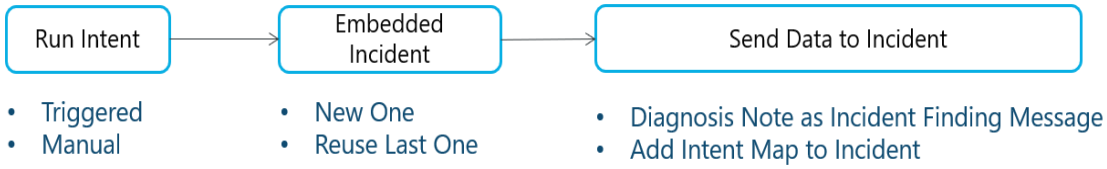
- Transient problem diagnosis – findings across multiple executions of the same intent are documented inside the Intent Incident.
- Long-term intent executions – compliance intent or published intents may be executed after the network change, and the critical findings are documented inside the intent's incident so that users can review this from within intent.
- Send an email or webhook API call to the 3rd party with a link to point back to the incident /portal with more data for the execution results.







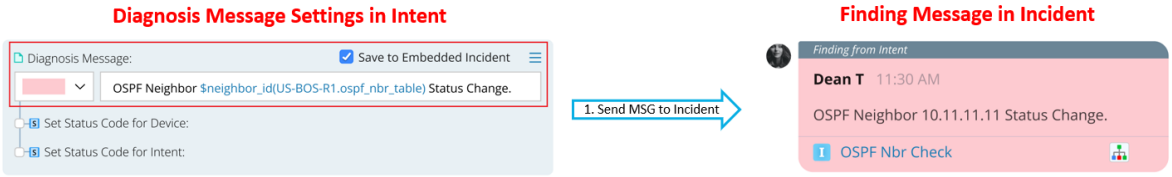
Use the embedded incident as follows:



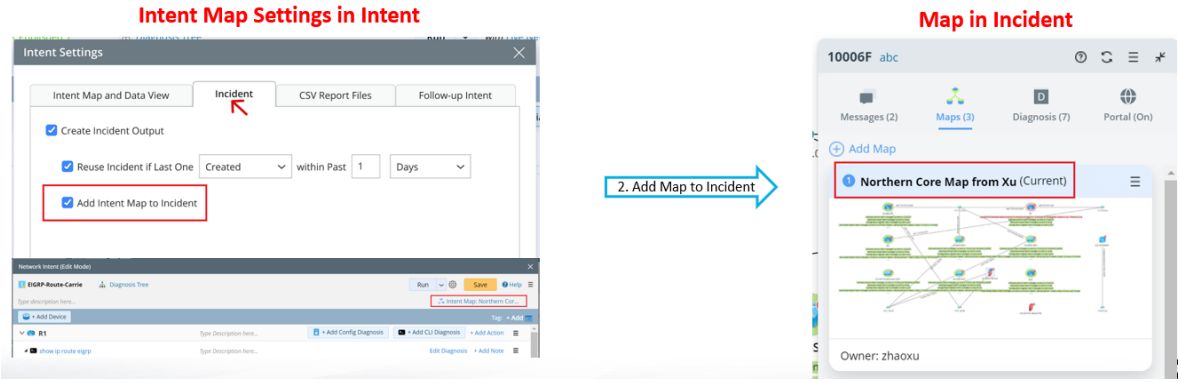
4.2.1 Document Critical Findings in Embedded Incident

Users can send messages and add a map to the incident, enabling users to track all execution history of a NI(T) from Interactive/Triggered/Preventive automation.

- Send a Message to Incident

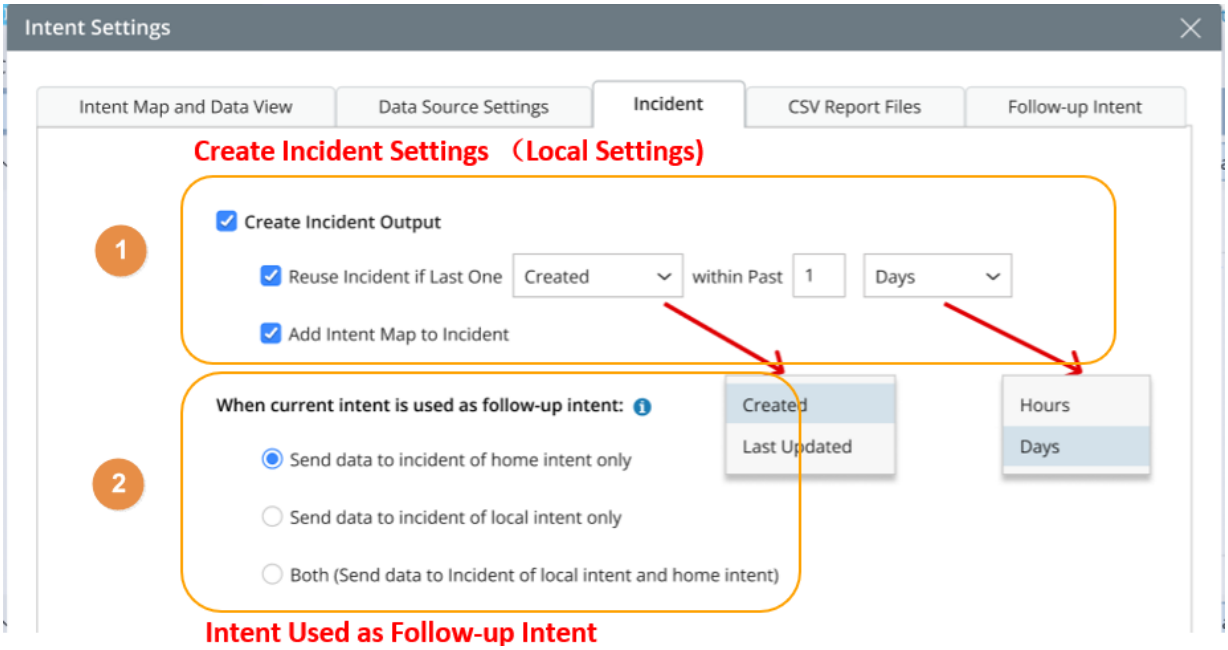


- Add Map to Incident

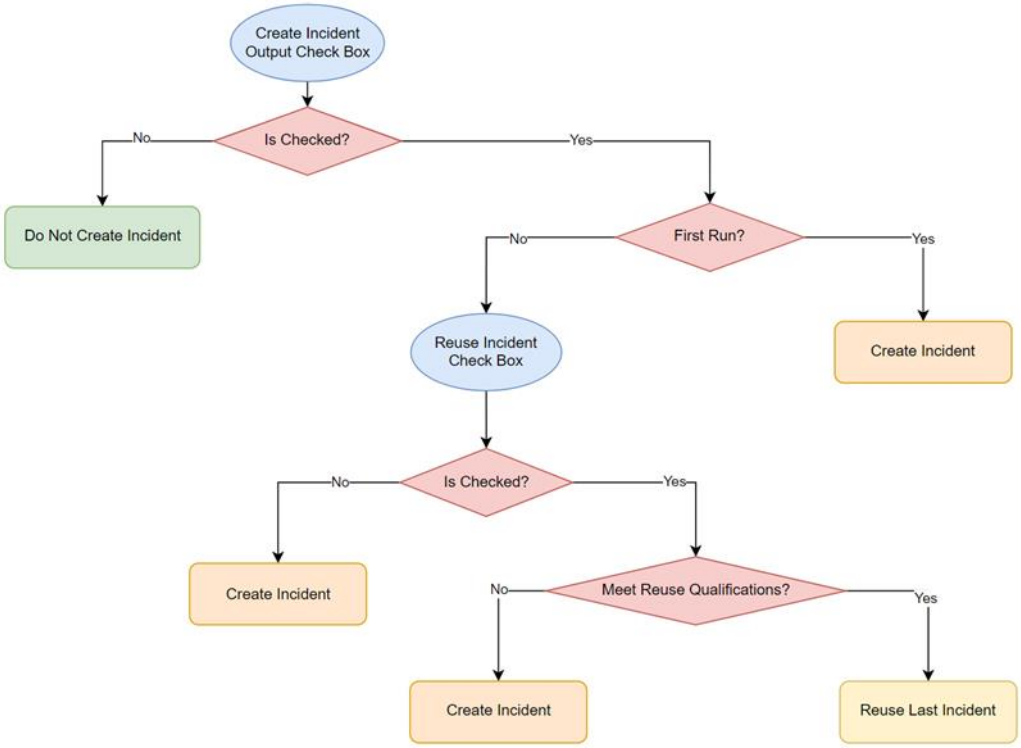


4.2.2 Create New or Reuse an Incident to Organize Results Reasonably

Users can define the incident creation settings as follow:

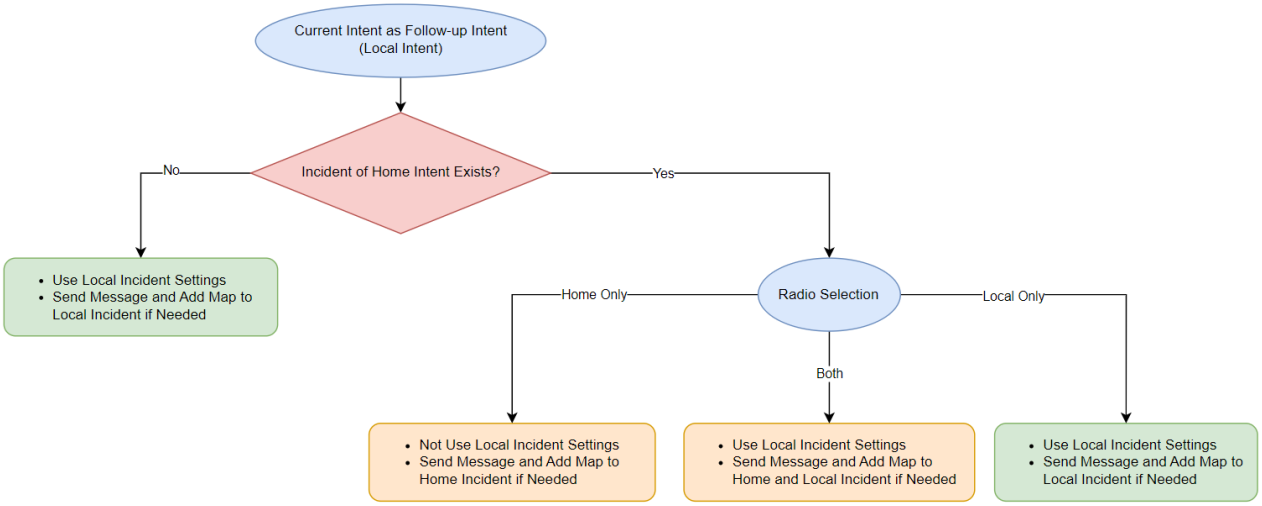


- Create Incident Output



  - If the Create Incident Output check box is selected, an incident will be generated when executing this NI for the first time; If this check box is not selected, the NI will not create an incident.
  - Reuse incident check box — indicates whether to use an existing incident from the second execution of NI. When the condition is not met, a new incident will be created. If it is not checked, a new incident is always created.

- Intent Used as Follow-up Intent

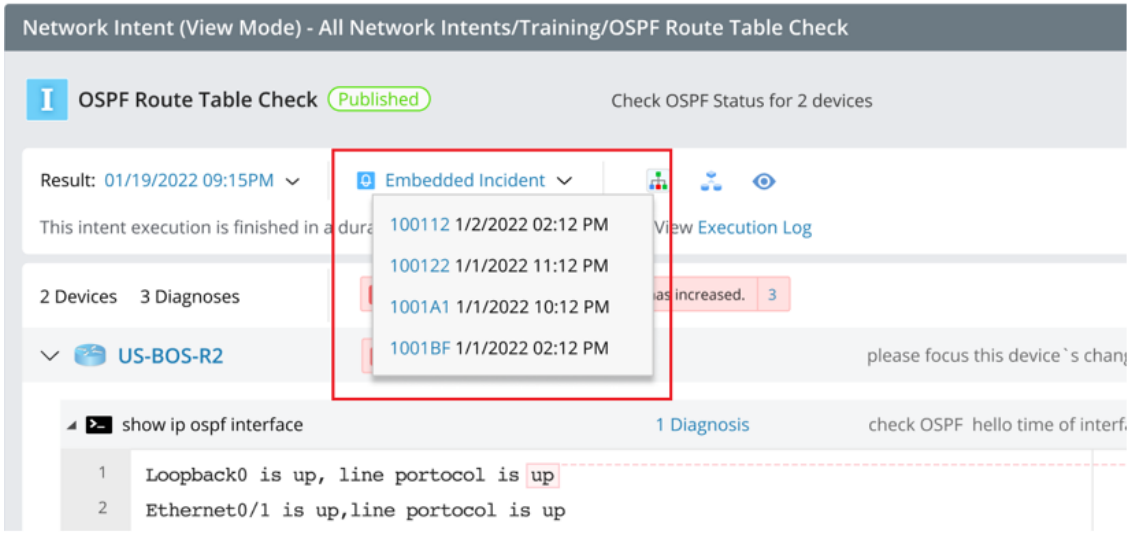


  - The setting here is based on the assumption that the home incident exists. If the home incident is not generated, the follow-up intent (local intent) always generates an incident or sends messages according to its local settings.
  - If the home incident is generated, users can set whether to use the home incident.
    - ✓ **Send to home only**: no local incident is created. Messages are sent to the home incident, and the intent map is added to the home incident.
    - ✓ **Send to local only**: local intent will create/reuse the local incident and send messages and intent map to the local incident.
    - ✓ **Both**: Indicates that the above two will work at the same time. It will send data to the home incident, and it will also create/reuse the local incident and send data to the local incident.

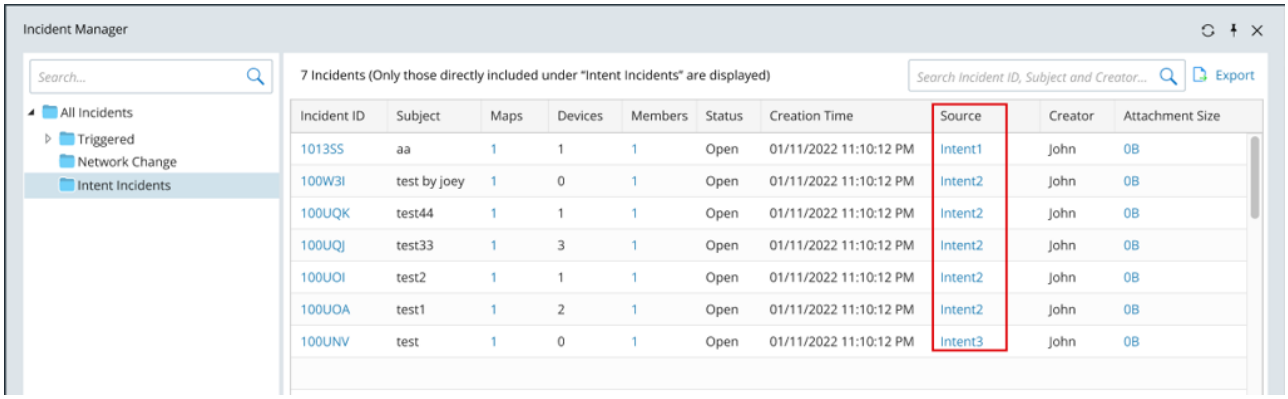
4.2.3 Manage Historical Results

Manage the embedded incidents in the following ways:

- For Single Intent: In the NI View mode, incidents generated by this intent can be displayed.

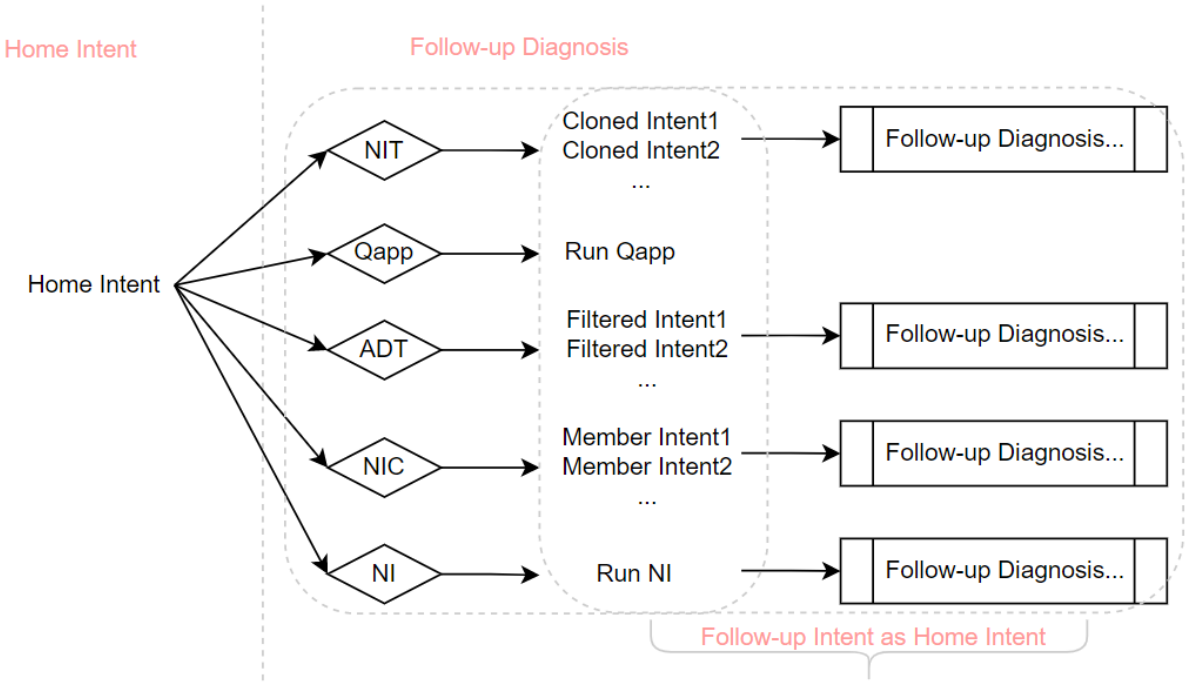


- For Entire Domain: Add the management of Intent Incidents in Incident Manager.



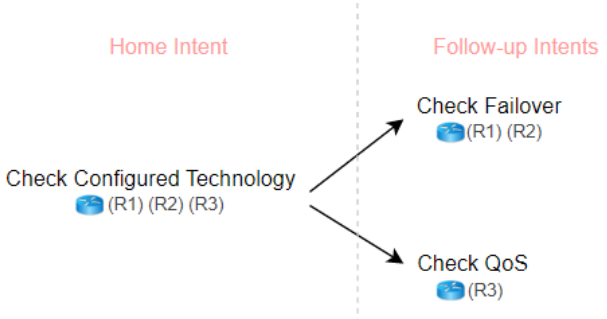
4.3 Follow-up Diagnosis

The previous version supports NI and NIC as the follow-up diagnosis. R11.1 adds NIT, Qapp, and the intents from ADT as the follow-up logic. Compared with NIC, NIT and Qapp do not replicate member Intents in advance, and the definition is simpler. Follow-up diagnosis enables multi-stage reasoning so that a follow-up intent can be taken as a home intent to do further follow-up diagnosis.

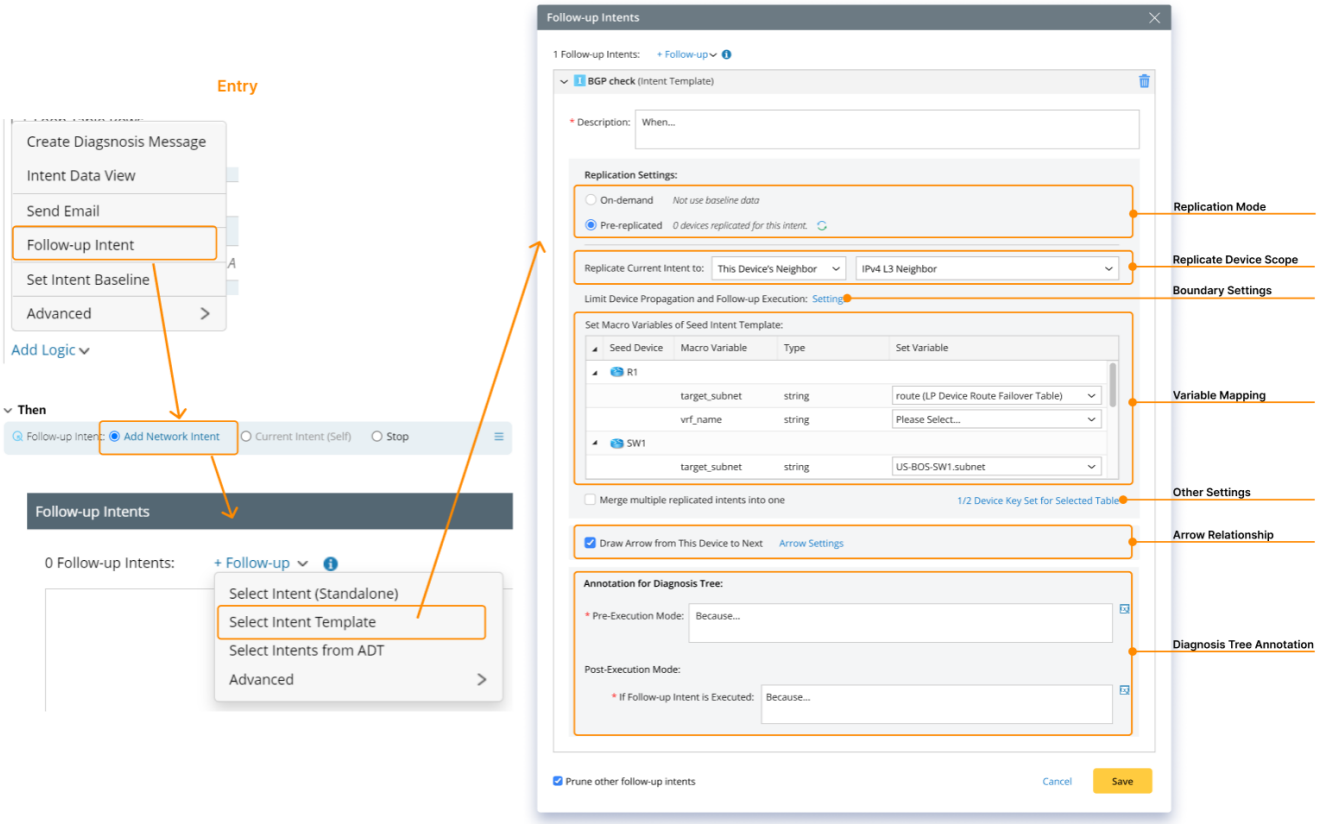


4.3.1 Follow-up Intent Template

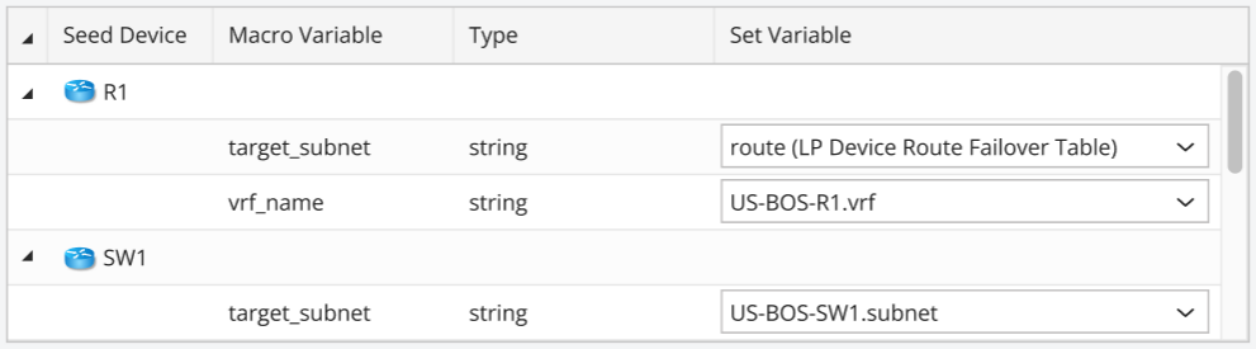
An intent (home intent) can call a specified intent template to run the follow-up diagnosis, under which the diagnosis logic in the template can be applied to downstream devices calculated from the home intent. The home intent can also transfer other variables to the macro variables of the follow-up intent templates.



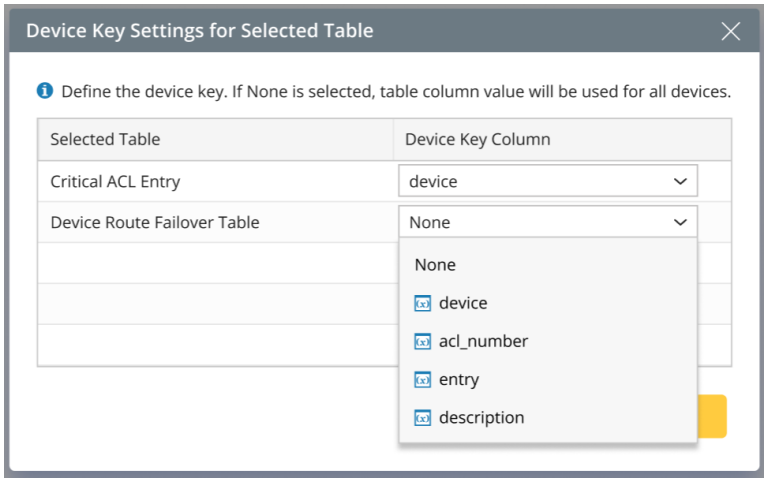
The follow-up intent supports the following setting:



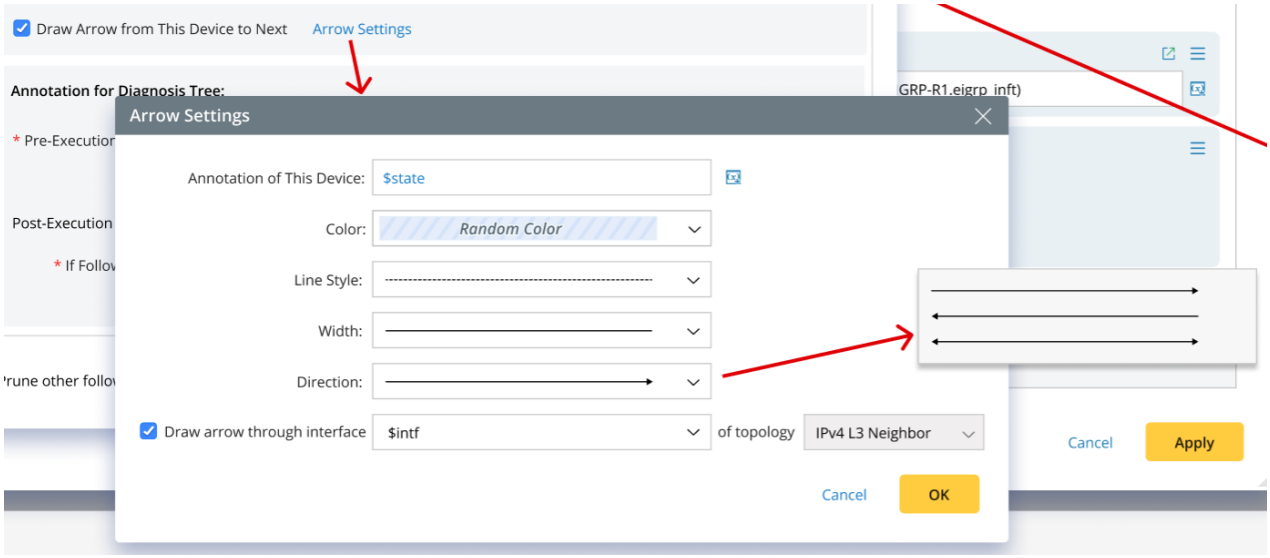
- **Replication Mode**
  - On-demand: Retrieve the data from the live network.
  - Pre-replicated: Use pre-decoded data (cached data).
- **Replicate Current Intent to**:
  - This Device's Neighbor, IPv4 L3 Neighbor, IPv6 L3 Neighbor, and L2 Neighbor can be selected.
  - Device by Variable, the range of variables that can be selected includes:
    - ✓ $this_device
    - ✓ Device Variables under Calling Intent
    - ✓ Device Macro Variables under Calling Intent
    - ✓ Global Variables under Calling Intent
- **Set Macro Variables of Follow-up Intent Template**: The range of variables that can be selected is the same as the Device by Variable in Replicate Device Scope.



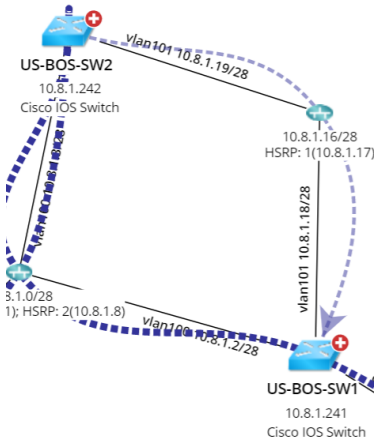
- **Merge multiple replicated intents into one**: This check box is not selected by default, which indicates each device generates an intent.
- **Set Device Key for Selected Table:** Set the device key to match table variable values precisely.



- **Draw Arrow from This Device to Next**: Draw an arrow directly from the current device to the next device whose intent is to be replicated.

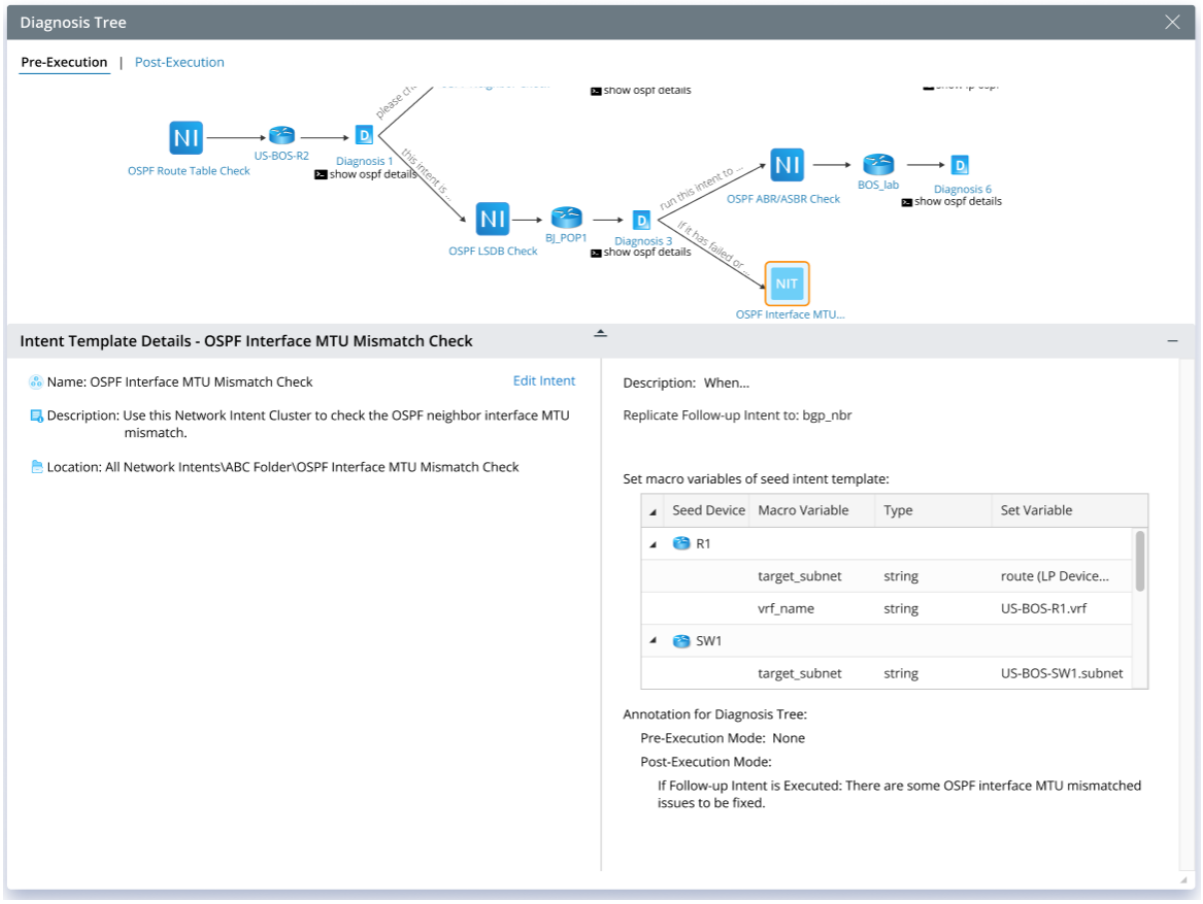


An arrow example on a map:

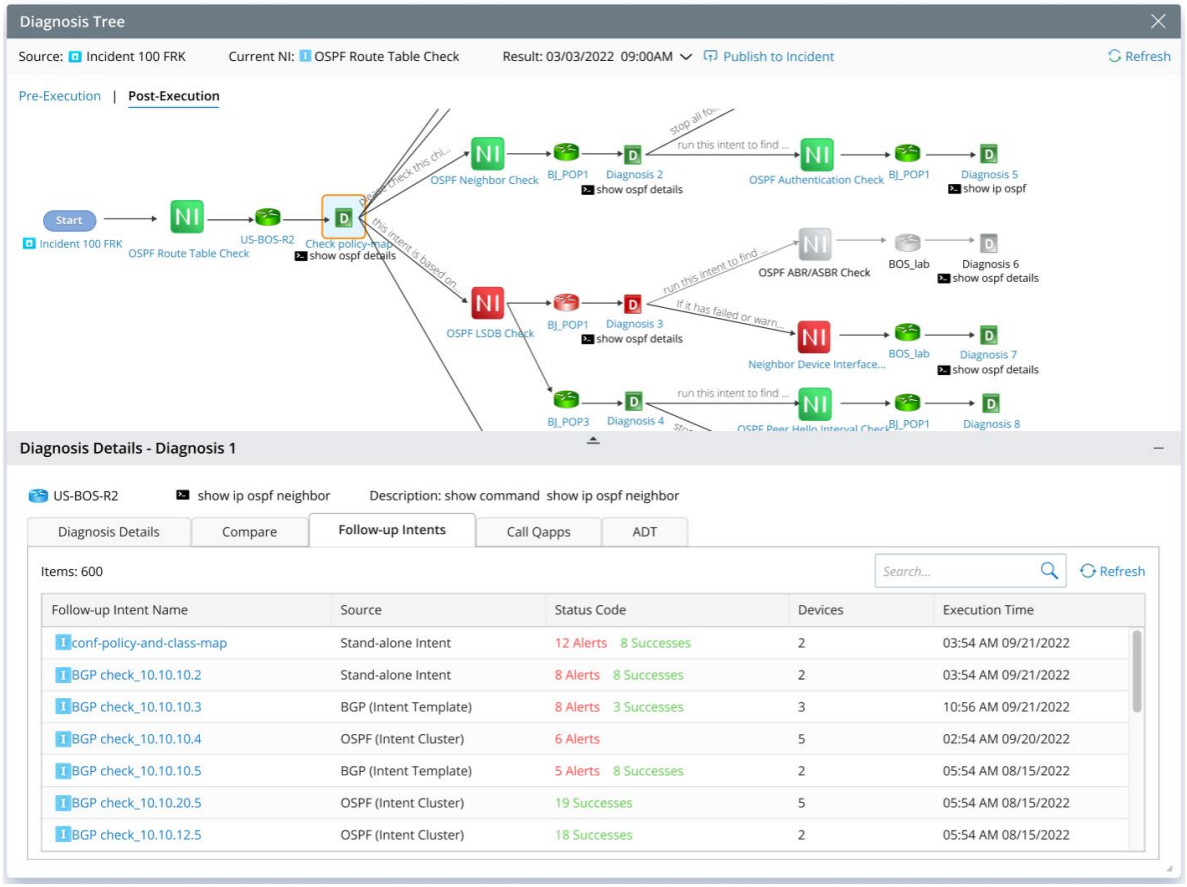


- **Annotation for Diagnosis Tree:** Annotations are displayed on the line of Diagnosis Tree. The diagnosis tree of the follow-up intent template is as follows:

- **Pre-Execution Mode**: The basic information of the intent template is displayed on the left, and the information defined in the intent is displayed on the right.



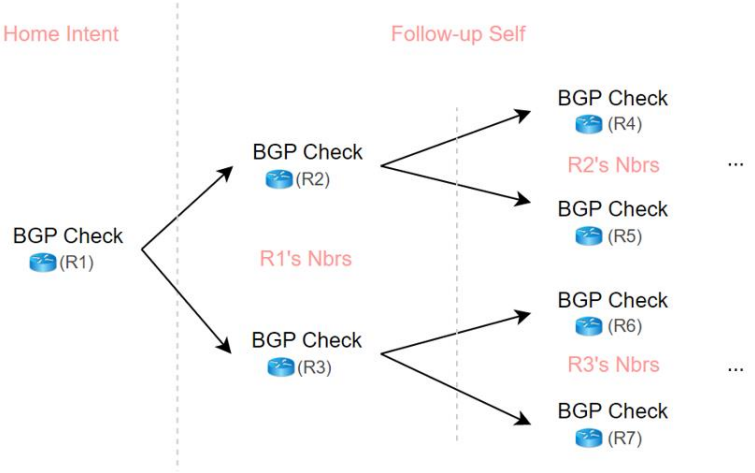
- **Post-Execution Mode**: The detailed results are displayed below. Besides the diagnosis details and comparison details, the follow-up intents/Qapps/ADT will be displayed.



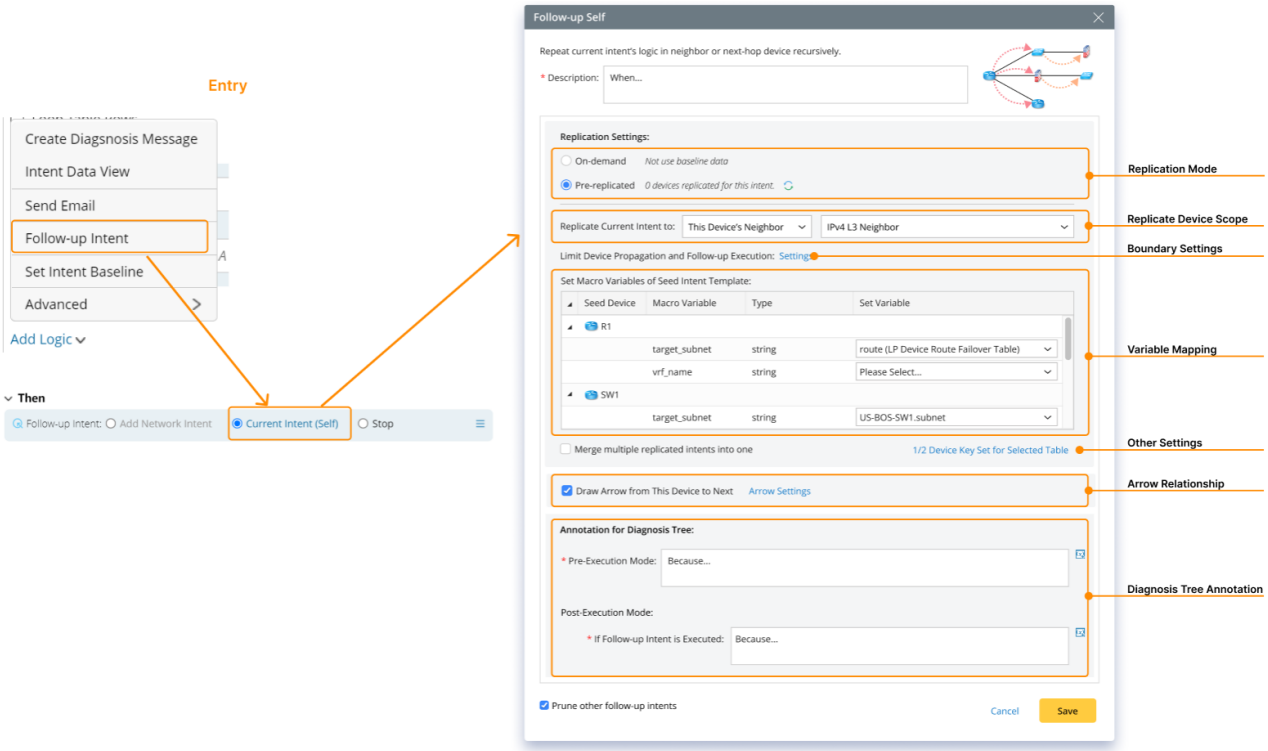
4.3.2 Follow-up Self

An Intent (home intent) can call itself to be the follow-up intent template, under which the same logic will be applied to downstream devices calculated from the home intent. The logic is recursively called upon until it

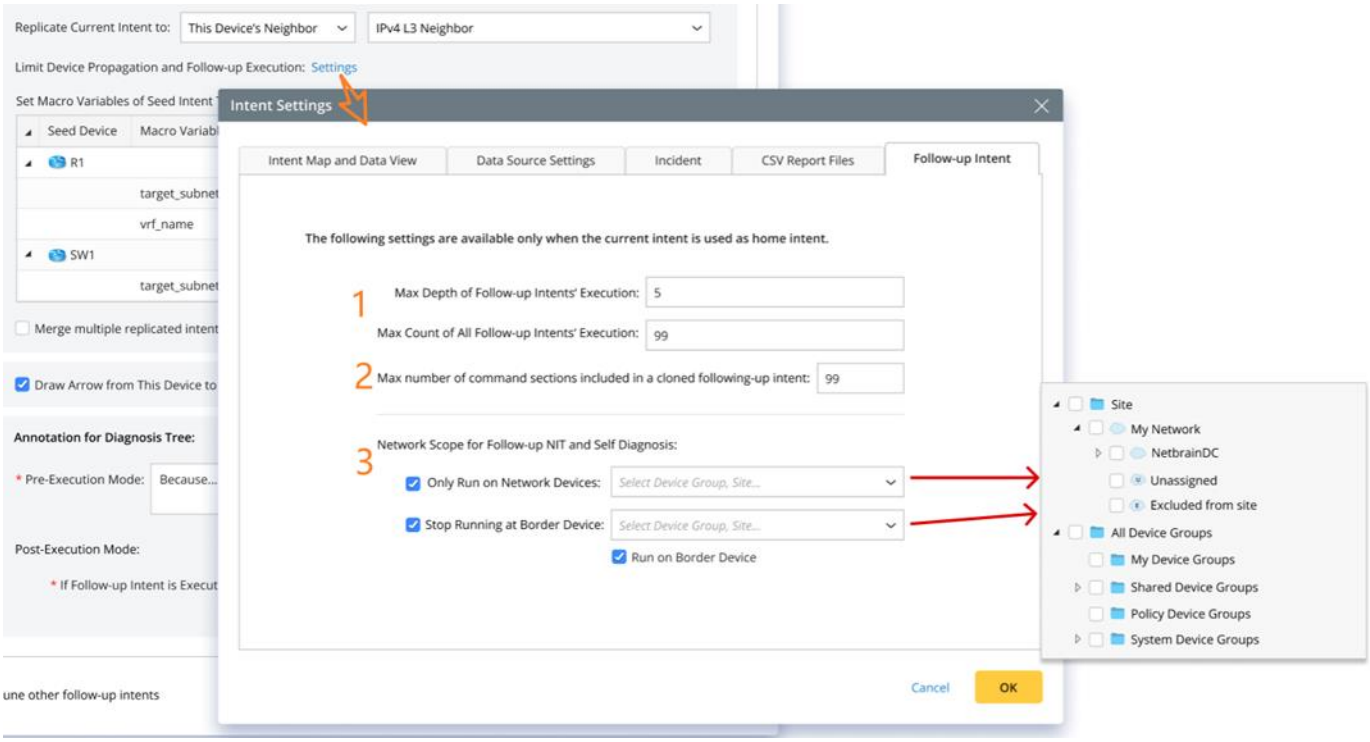
hits a boundary defined by logic or maximum depth.



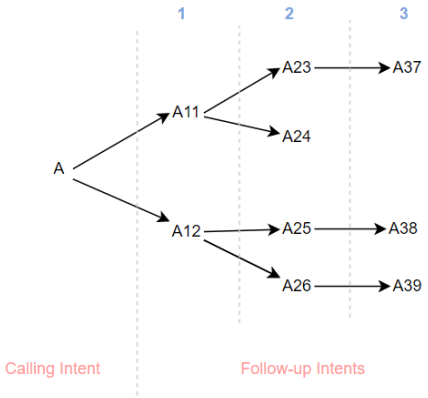
The Follow-up Self settings are like follow-up NIT:



Users can control when to hit the boundary by defining max depths and device scope:



Users can set the max depths and follow-up intent count in **Area 1**. They can also set the max command sections in one cloned intent in **Area 2**.

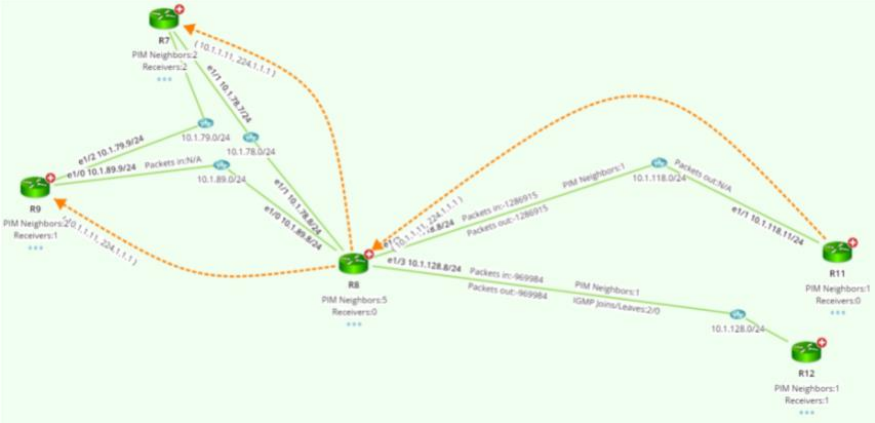


Define network scope in **area 3**:

- Limit the range of devices. The devices beyond this range will not be replicated.
- When entering a border device, it will not continue to find its neighbors, but whether to clone the intent and continue the diagnosis on the current border device can be set.
- Device groups and sites can be selected for both selections, and multi-selection is supported.

Examples of Follow-up Self:

- Execute recursive diagnosis from the current Device towards a Target IP Self.
- Draw a path with the arrow.
- Execute recursive diagnosis across Multi-level Neighbors.
- Draw the multicasting tree and execute recursive diagnosis across the multicasting tree.

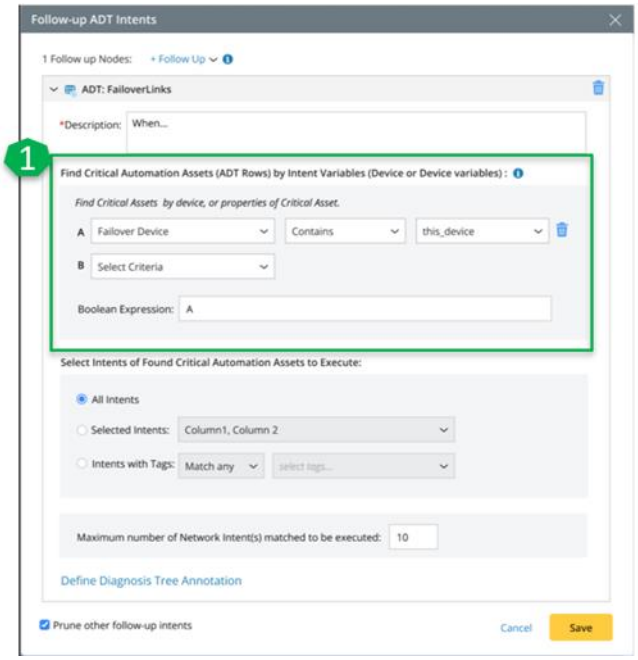


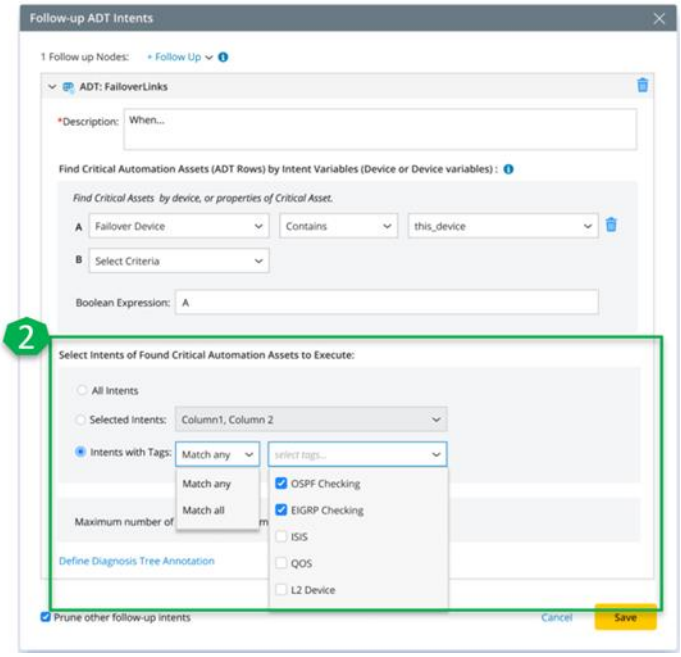
- Create a Technology Map and Highlight the Interface Area.

4.3.3 Follow-up ADT Intents

In R11.1, Automation Data Table (ADT) can reference multiple automation intent columns for each automation asset, and users can use intents (e.g., path intents) in ADT as follow-up diagnosis. The definition of follow-up

ADT includes the following:





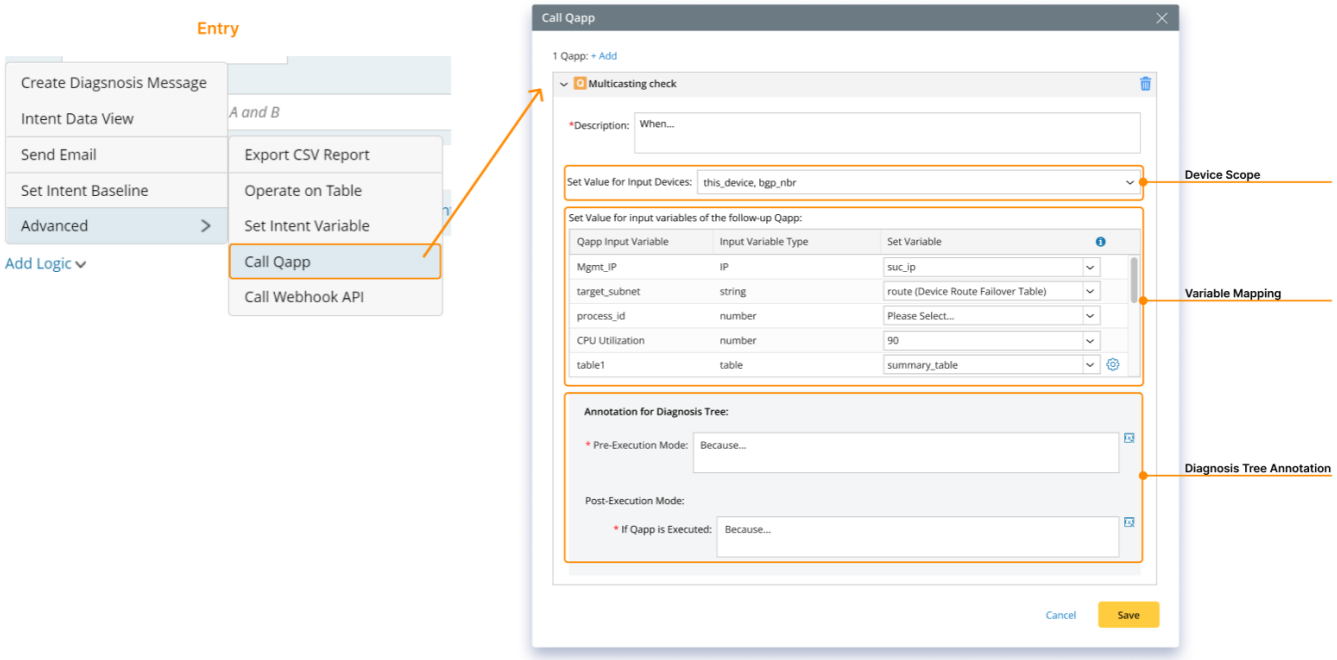
1. Find Automation Assets by Intent Variables (Context Device or other Device Variables)
2. Select the Intent of Automation Assets to be Executed.

Use the follow-up ADT intent to analyze the network problems of related automation assets. For example, check the OSPF and EIGRP state of the failover device when the HSRP state is changed.

4.3.4 Call Qapp

A new block, **Call Qapp**, can be added to the NI diagnosis. Users can select a Qapp and execute it by defining the matching relationship of input variables. The Qapp output link can be saved in NI if this Qapp is executed. Calling Qapp can execute Python scripts for advanced functions, e.g., making an API call to a 3rd party system to send the notification.

Intent Variables can be passed to follow-up Qapp. Users can execute the Qapp with a Qapp output link saved in NI.



Use Qapp to Run Advanced Functions like Making API Calls to Outside Resources. For example, users can create an intent to get the device vendor, model, and serial number, then pass these parameters to a Qapp to call ***Cisco Diagnosis API*** to get ***End of Life (EOL)***.

4.4 New Operation in Intent Table

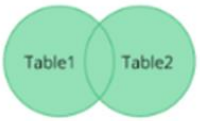
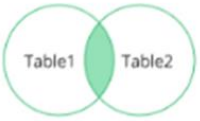
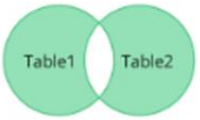
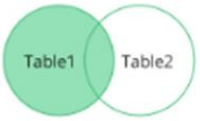
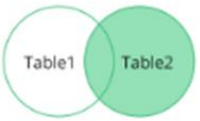
Many network data are tables. R11.1 supports more operations of tables without coding:

Table Operation	Description
Merged Table Updated	Merge two tables using DB-style programming: full/inner/outer/left/right join two tables to extract matching data rows to improve the loop table efficiency in the diagnosis.
Appended Table New	Append the data from two tables with different sources.
Sub Table (Filtered by Condition) New	Filter table rows by a condition to create a sub table, reducing the noise in the diagnosis.

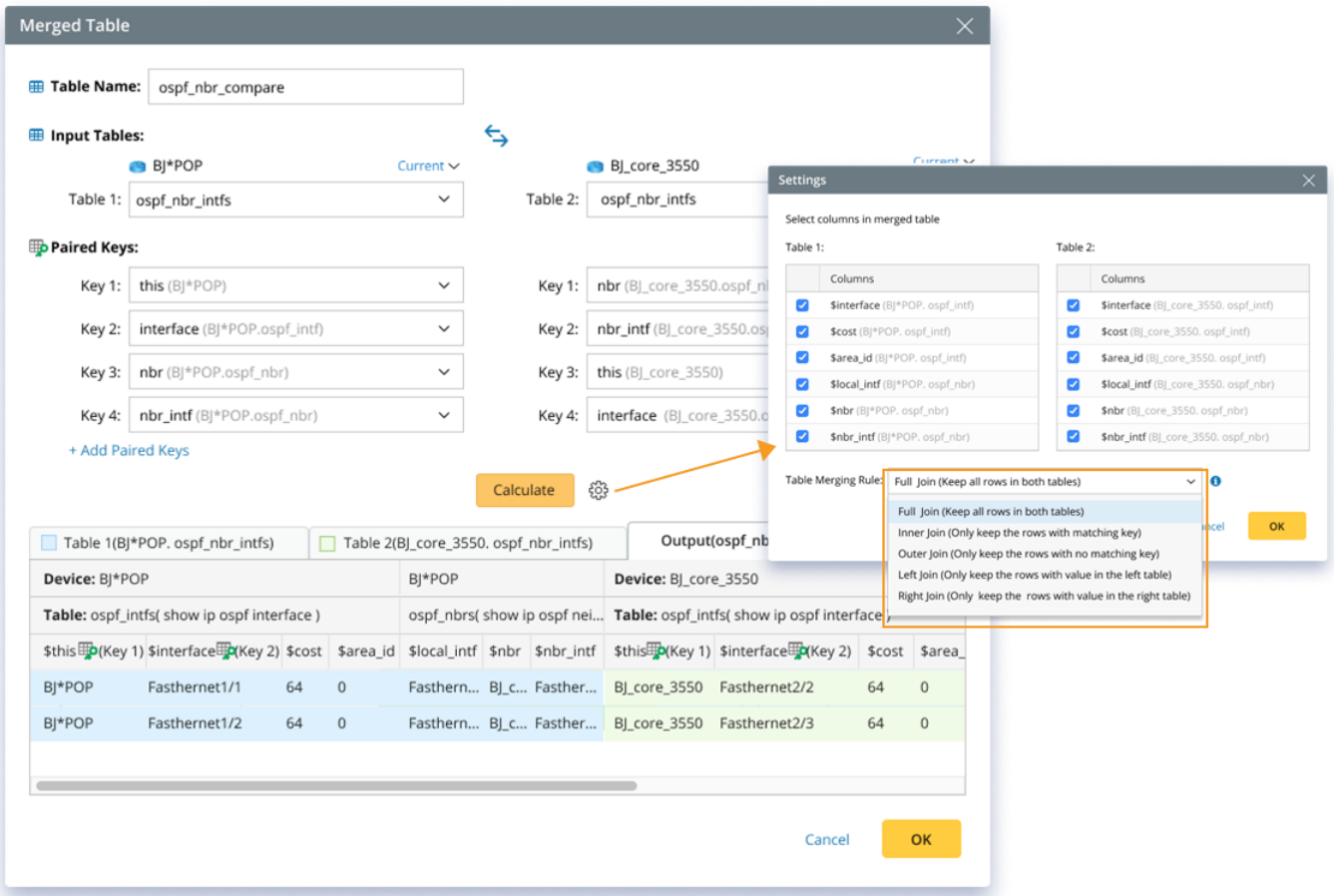
Add/Delete/Update/Query Table New	Support the “add/delete/update/query” table operation on the intent table and ADT to ensure the accuracy and completeness of table data.
Built-in Device Table Diagnosis New	Reference built-in device data, e.g., NAT table, and loop the built-in table rows to check the data accuracy and the live status for the table key metric.
Automation Assets Diagnosis New	Reference ADT, e.g., critical failover links, and loop the ADT rows to get the key assets as input for the follow-up diagnosis.

4.4.1 Merged Table

In the previous version, the system supports two Merging Rules: **Full Join** and **Inner Join**. R11.1 enhances Merged Table to perform DB-style merging rules (**outer/left/right join**) on table data.

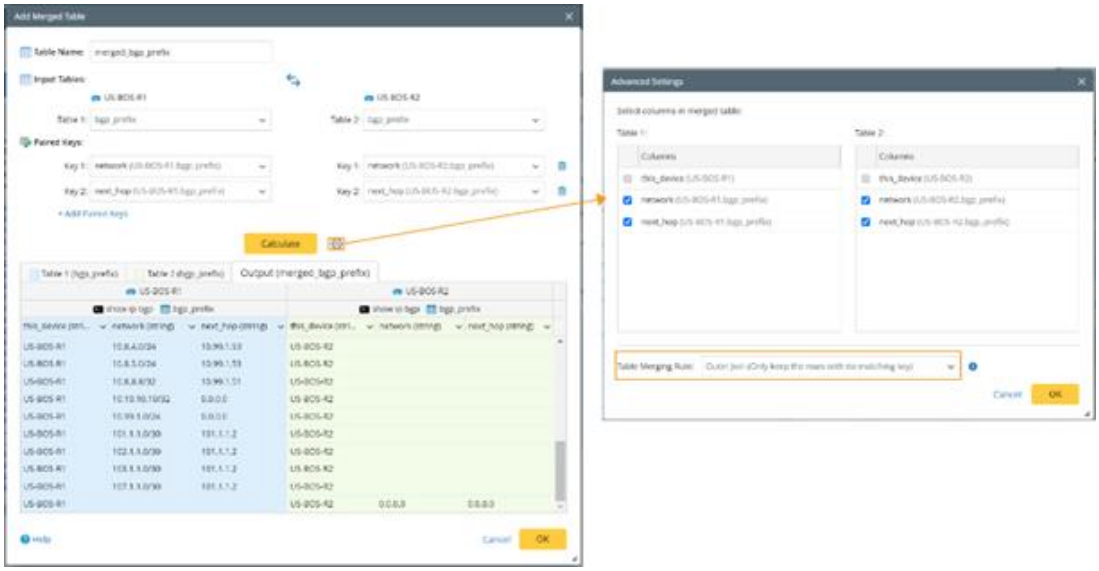
Merged Rule	Logic Diagram	Logic Description with Sample Data
Full Join		Returns all rows when there is a match in left (table1) or right (table2) table rows. **Left Table (Table1)** Column1 | Column2 1 | a 2 | b 3 | c **Right Table (Table2)** Column1 | Column2 b | x d | z **Merged Table** L. Column1 | L. Column2 | R. Column1 | R. Column2 1 | a 2 | b | b | x 3 | c d | z
Inner Join		Returns all rows that have matching values in both tables. **Left Table (Table1)** Column1 | Column2 1 | a 2 | b 3 | c **Right Table (Table2)** Column1 | Column2 b | x d | z **Merged Table** L. Column1 | L. Column2 | R. Column1 | R. Column2 2 | b | b | x
Outer Join New		Returns all rows that have no matching values in both tables. **Left Table (Table1)** Column1 | Column2 1 | a 2 | b 3 | c **Right Table (Table2)** Column1 | Column2 b | x d | z **Merged Table** L. Column1 | L. Column2 | R. Column1 | R. Column2 1 | a 3 | c d | z
Left Join New		Returns all rows from the left table (table1), and the matching rows from the right table. **Left Table (Table1)** Column1 | Column2 1 | a 2 | b 3 | c **Right Table (Table2)** Column1 | Column2 b | x d | z **Merged Table** L. Column1 | L. Column2 | R. Column1 | R. Column2 1 | a 2 | b | b | x 3 | c
Right Join New		Returns all rows from the right table (table2), and the matching rows from the left table. **Left Table (Table1)** Column 1 | Column 2 1 | a 2 | b 3 | c **Right Table (Table2)** Column 1 | Column 2 b | x d | z **Merged Table** L. Column 1 | L. Column 2 | R. Column 1 | R. Column 2 2 | b | b | x d | z

The key logic of Merged Table:



1. The output table name must follow variable naming rules.
2. Select the table variables for Table 1 and Table 2 and specify the data source (Current, Baseline, Last) for the table variable. The table variable includes the parser table, intent table, built-in table, ADT, and compound table.
3. **Add Paired Keys** to define the Paired Keys for Tables 1 and 2. The Paired Keys define the column pairs to be merged, and users can have multiple Paired Keys.
4. Select the columns and select the table merge rule in **Settings**.

Use the merged table to merge table data from two tables. For example, find the missing BGP prefixes between primary and secondary wan routers.



4.4.2 Appended Table

Appended Table is added in R11.1, which appends one table row into the other. Users must define how to map the columns of two tables.

Table 1: OSPF routes

BJ_core_3550 show ip route ospf

ospf_routes

flag (string)	subnet (string)	next_hop (string)	out_intf (string)
O E2	192.168.29.0/24	172.24.10.33	Vlan10
O	172.25.52.0	172.24.100.2	Port-channel10
O	172.25.54.0	172.24.100.2	Port-channel10
O	172.25.51.0	172.24.100.2	Port-channel10
O	172.25.50.0	172.24.100.2	Port-channel10
O E2	172.25.32.0	172.24.100.2	Port-channel10

Table 2: Connected routes

BJ_core_3550 show ip route connected

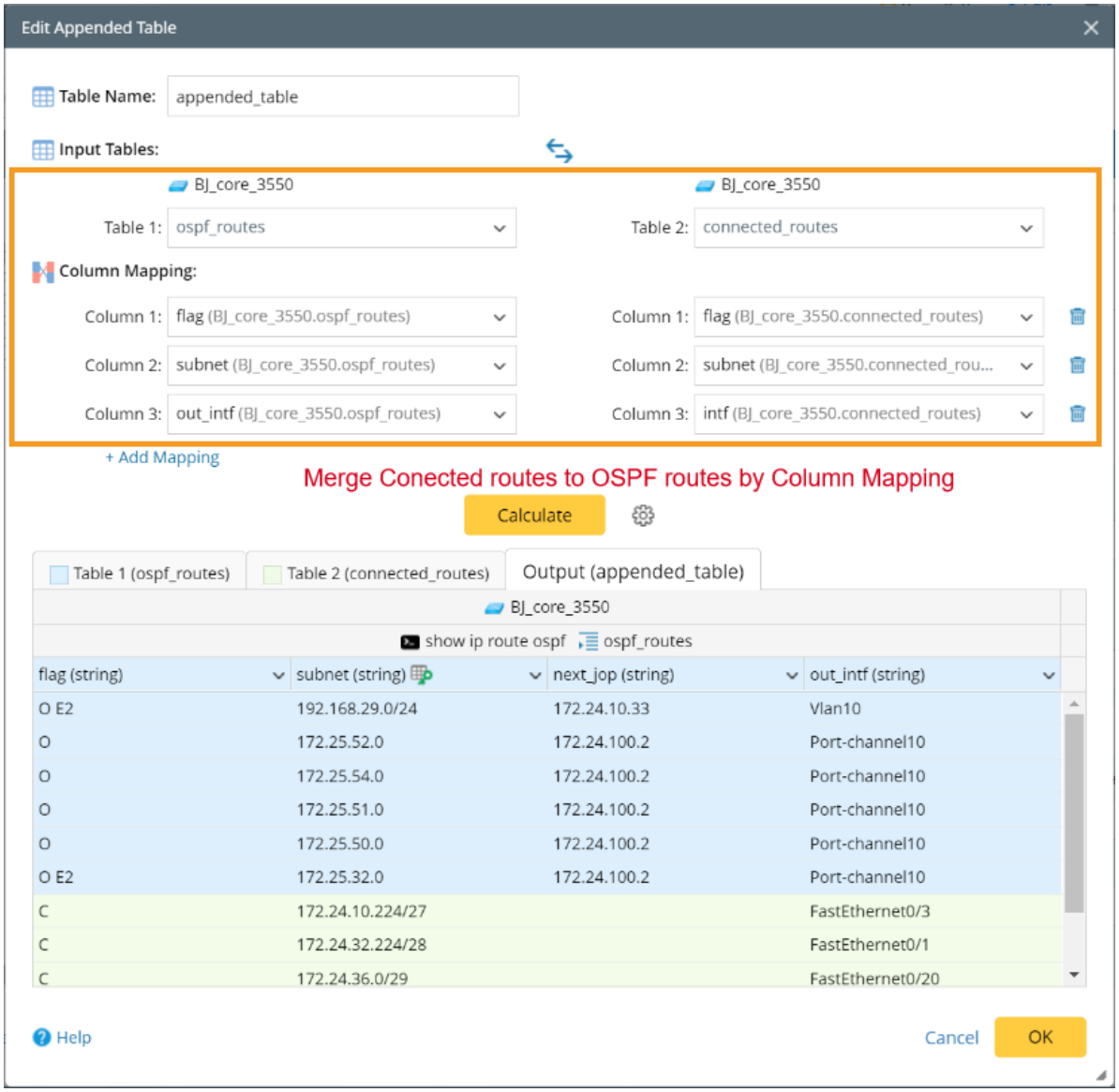
connected_routes

flag (string)	subnet (string)	intf (string)
C	172.24.10.224/27	FastEthernet0/3
C	172.24.32.224/28	FastEthernet0/1
C	172.24.36.0/29	FastEthernet0/20
C	172.24.10.32/27	Vlan10
C	172.24.100.0/30	Port-channel10

Appended Table: OSPF + Connected routes

flag (string)	subnet (string)	next_jop (string)	out_intf (string)
O E2	192.168.29.0/24	172.24.10.33	Vlan10
O	172.25.52.0	172.24.100.2	Port-channel10
O	172.25.54.0	172.24.100.2	Port-channel10
O	172.25.51.0	172.24.100.2	Port-channel10
O	172.25.50.0	172.24.100.2	Port-channel10
O E2	172.25.32.0	172.24.100.2	Port-channel10
C	172.24.10.224/27		FastEthernet0/3
C	172.24.32.224/28		FastEthernet0/1
C	172.24.36.0/29		FastEthernet0/20
C	172.24.10.32/27		Vlan10
C	172.24.100.0/30		Port-channel10

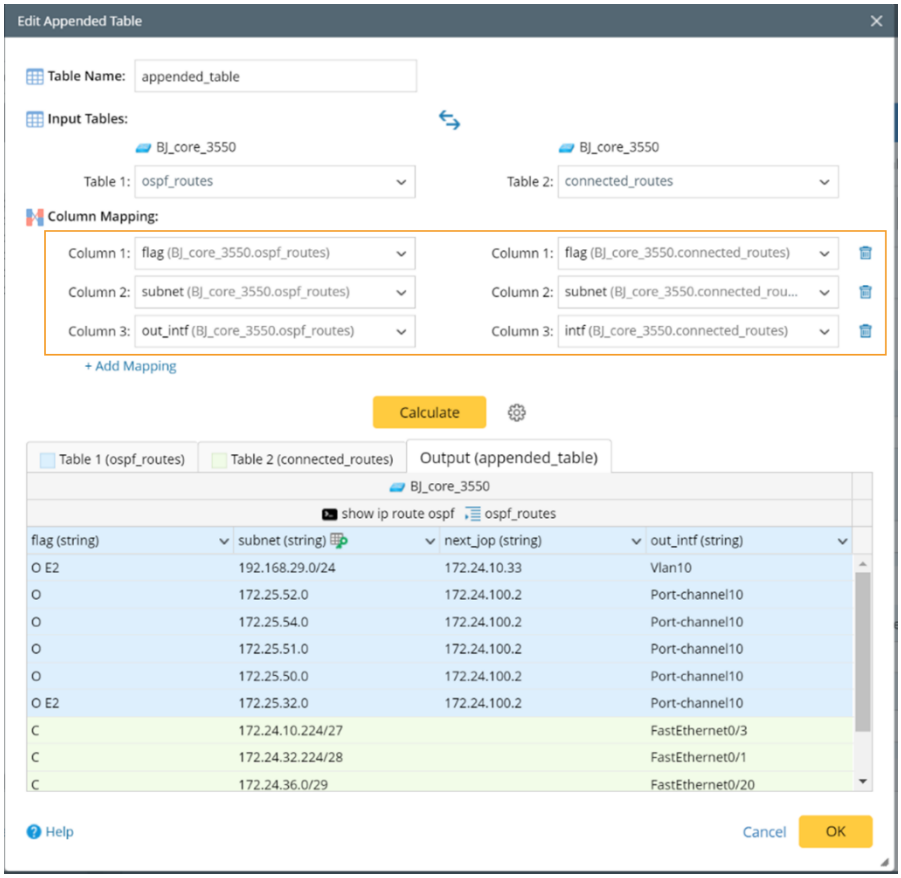
The definition of Appended Table:



The logic of Appended Table:

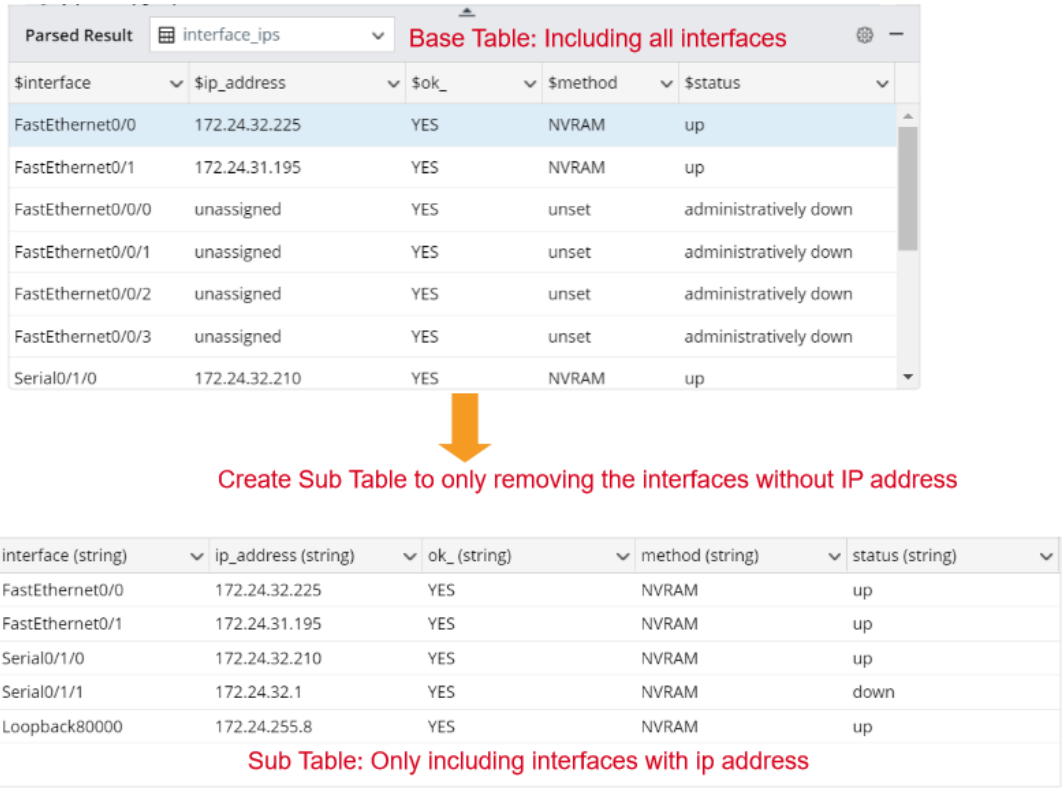
1. The output table name must follow variable naming rules.
2. Select the table variables for Input Table. The table variable includes a parser table, intent table, built-in table, ADT, and compound table.
3. **Column Mapping**: map columns of table2 to columns of table1. The column name of the Appended Table will be the same as that of Table 1.

Use Appended Table to merge rows from two tables. For example, append the connected route table and OSPF route table.

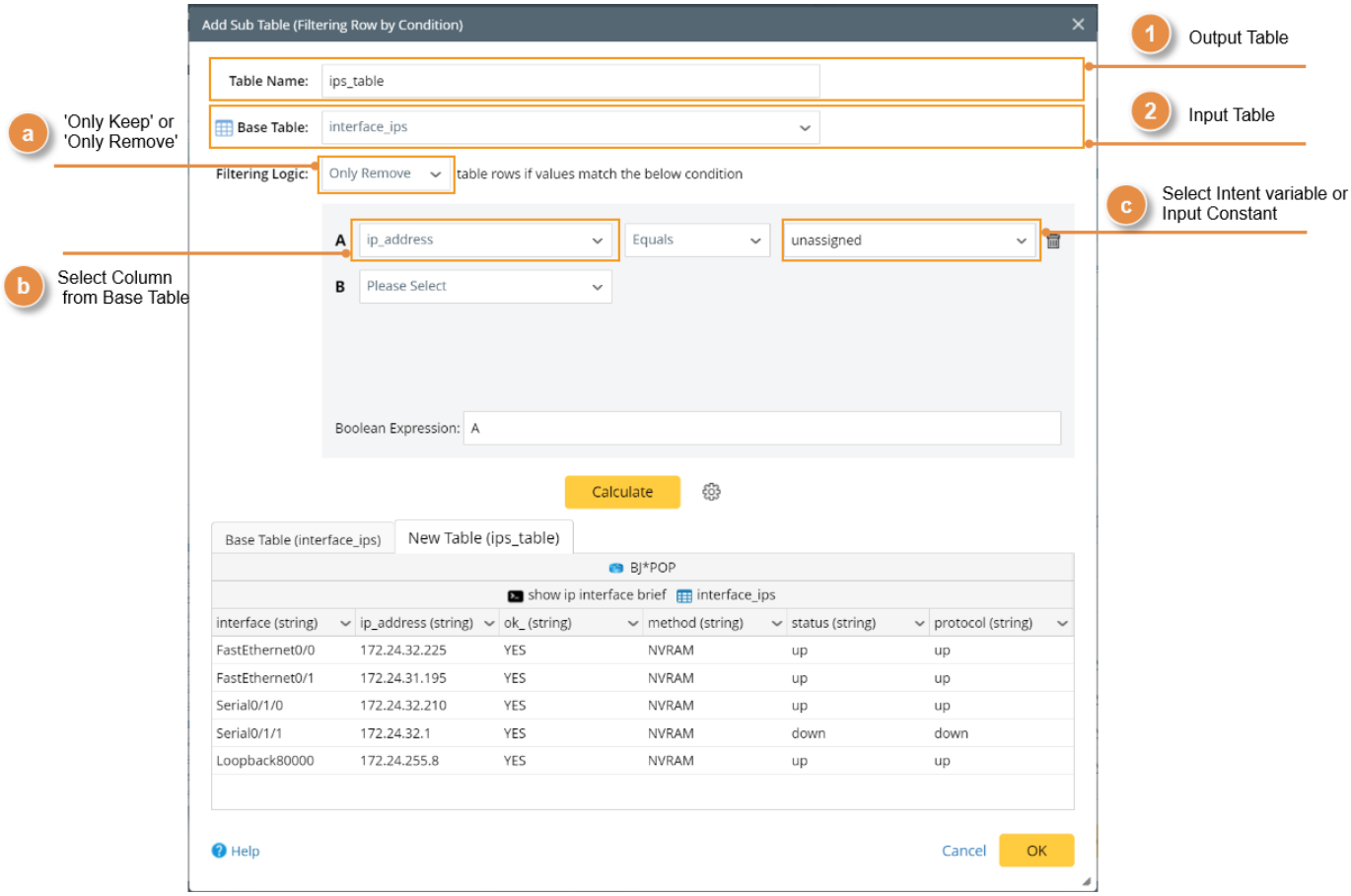


4.4.3 Sub Table Filtered by Condition

R11.1 supports defining Sub Table (Filter by Condition) to keep or remove the rows of the base table matching the conditions. The sub table can reduce the noise in the diagnosis.



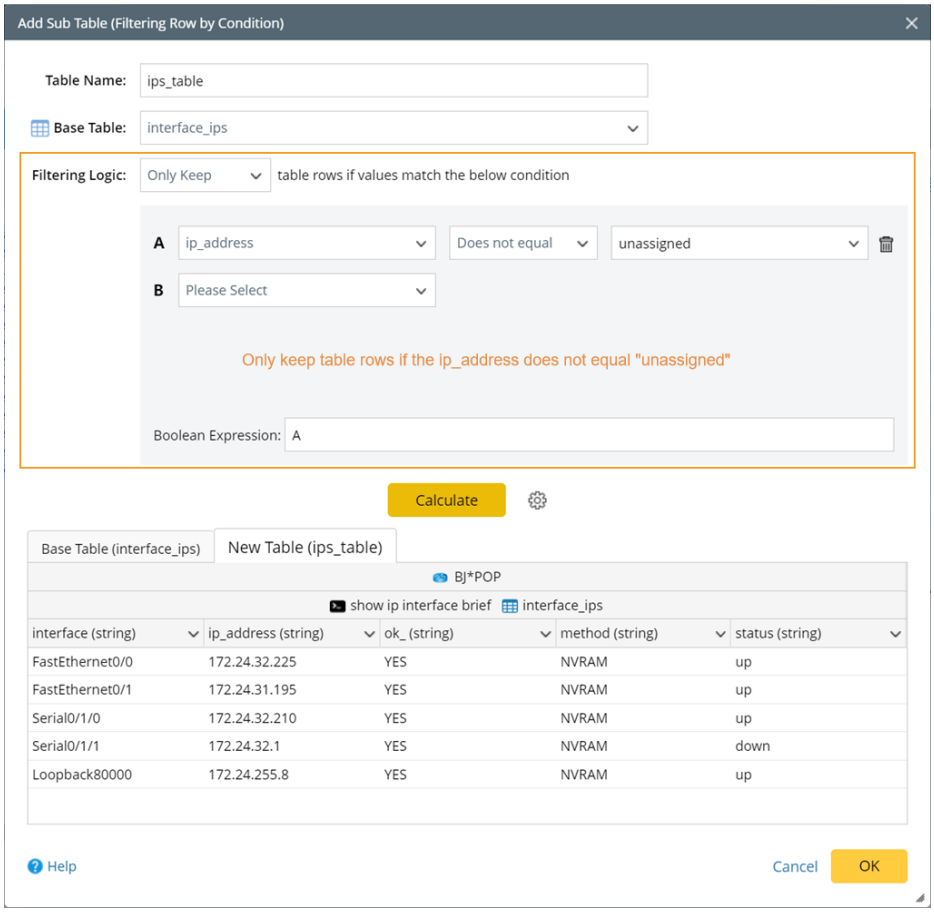
You can define the sub table (filtering row by condition):



The key logic of Sub Table (Filtering Row by Condition):

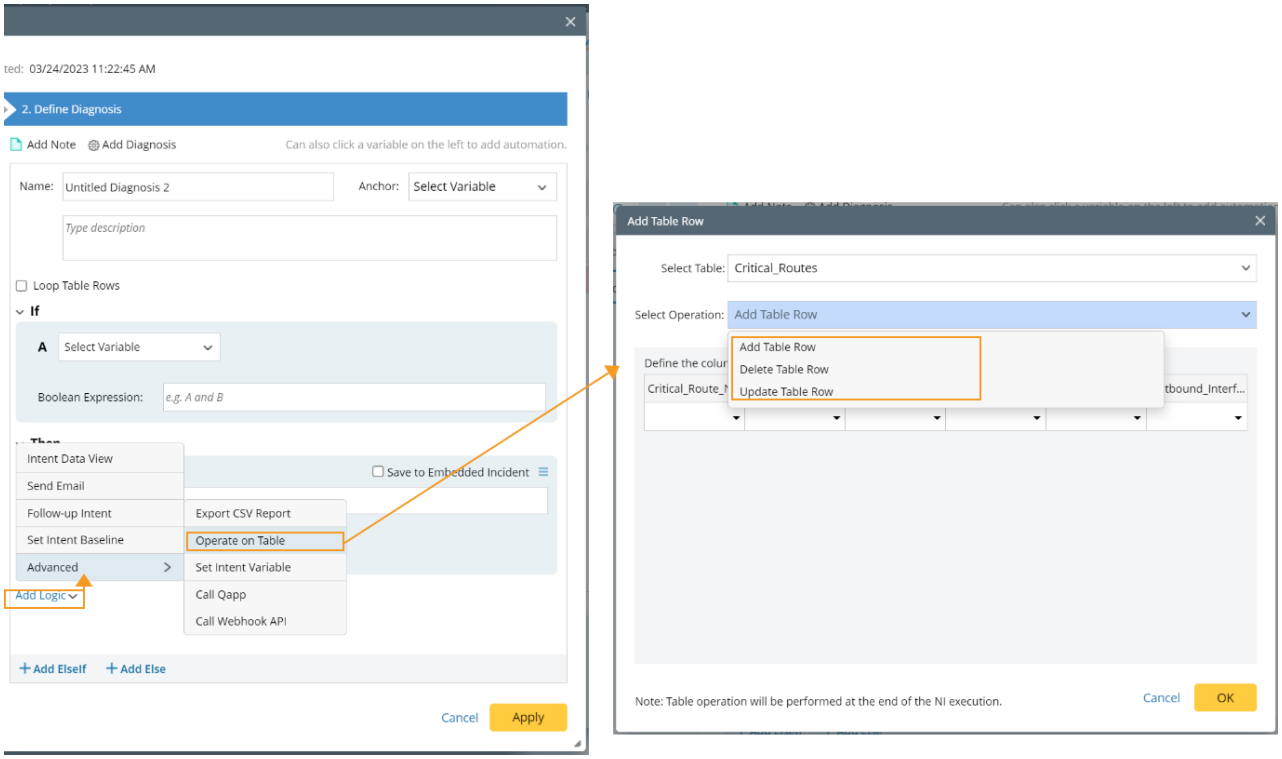
1. The output table name must follow variable naming rules.
2. Select the table variables for Base Table. The table variable includes a parser table, intent table, built-in table, ADT, and compound table.
3. **Filtering Logic:**
   a. Define filter rule: **Only Keep** or **Only Remove**.
   b. The left-side field can specify the columns of the Base Table.
   c. The right-side field can specify the Intent variable and enter a constant.

Use Sub Table to Filter Table Row by Specified Condition. For example, create a sub table to only keep the interface with the IP address configured.

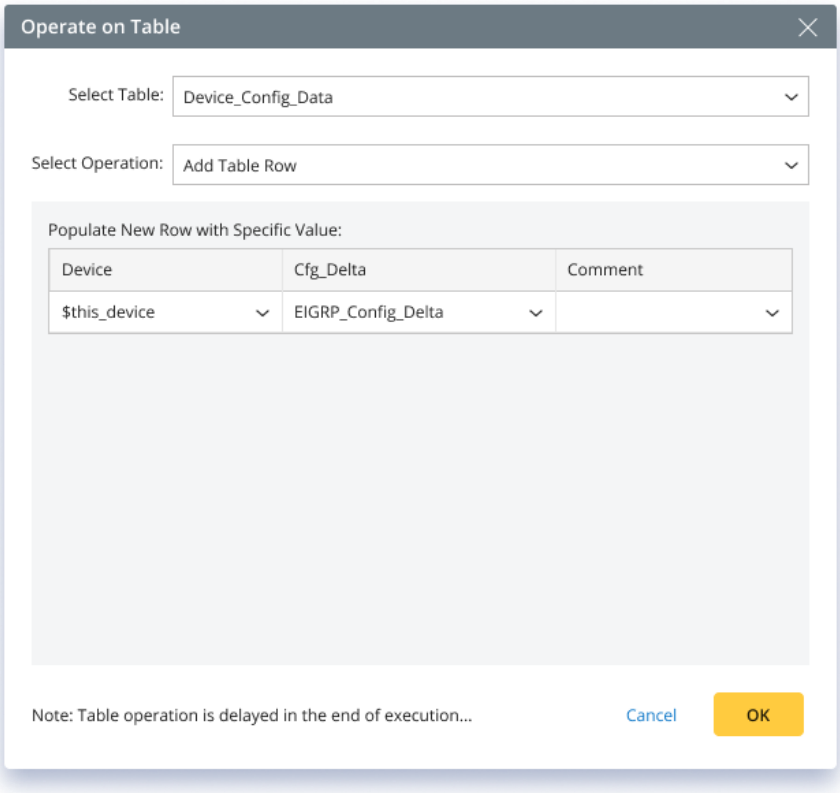


4.4.4 Enhance Operation on Table

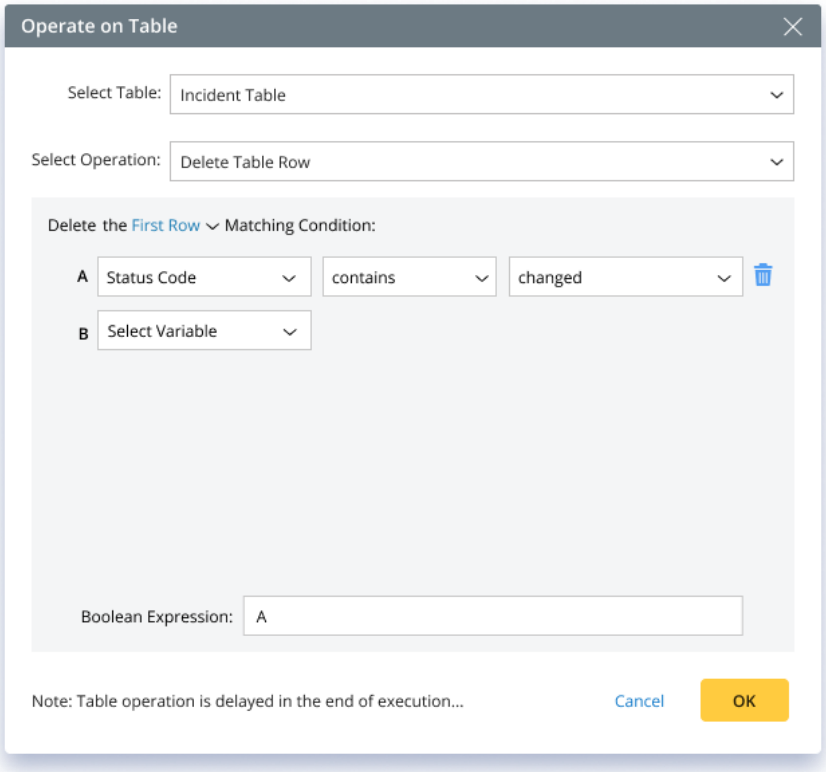
R11.1 adds table operations, “add/delete/update/query rows”, on the intent table and ADT in the intent diagnosis block:


ted: 03/24/2023 11:22:45 AM
2. Define Diagnosis
Add Note
Add Diagnosis
Can also click a variable on the left to add automation.
Name:
Untitled Diagnosis 2
Anchor:
Select Variable
Type description
Loop Table Rows
If
A
Select Variable
Boolean Expression:
e.g. A and B
Intent Data View
Send Email
Follow-up Intent
Set Intent Baseline
Advanced
Add Logic
Export CSV Report
Operate on Table
Set Intent Variable
Call Qapp
Call Webhook API
Save to Embedded Incident
+ Add ElseIf
+ Add Else
Cancel
Apply
Add Table Row
Select Table:
Critical_Routes
Select Operation:
Add Table Row
Add Table Row
Delete Table Row
Update Table Row
Define the colur
Critical_Route_
tbound_Interf...
Note: Table operation will be performed at the end of the NI execution.
Cancel
OK

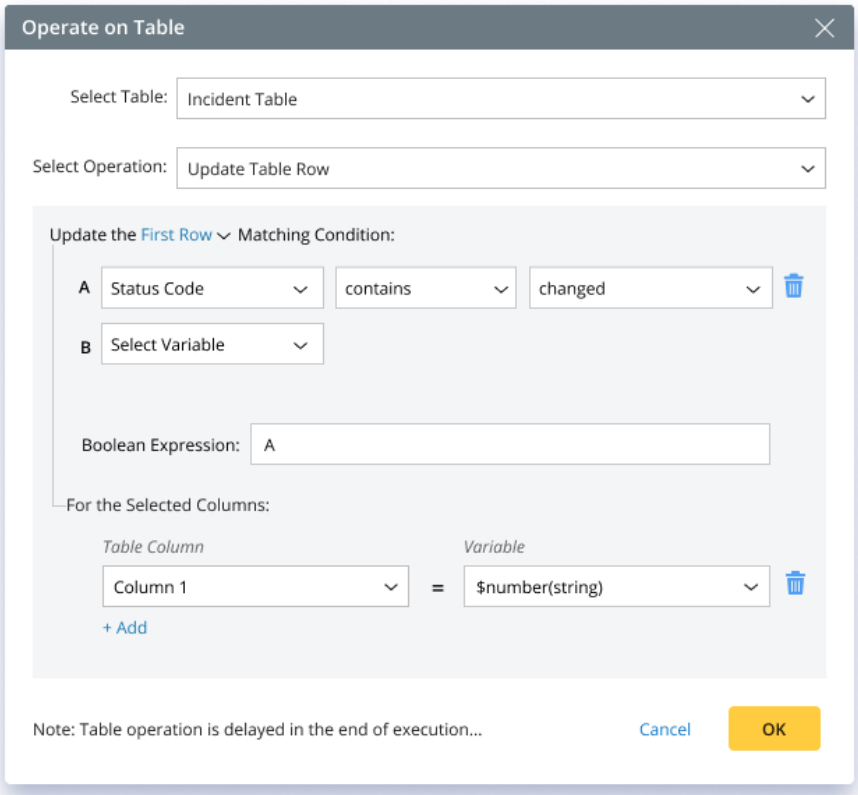
- **Add table row**: Select a table and populate a new row with a specific value.



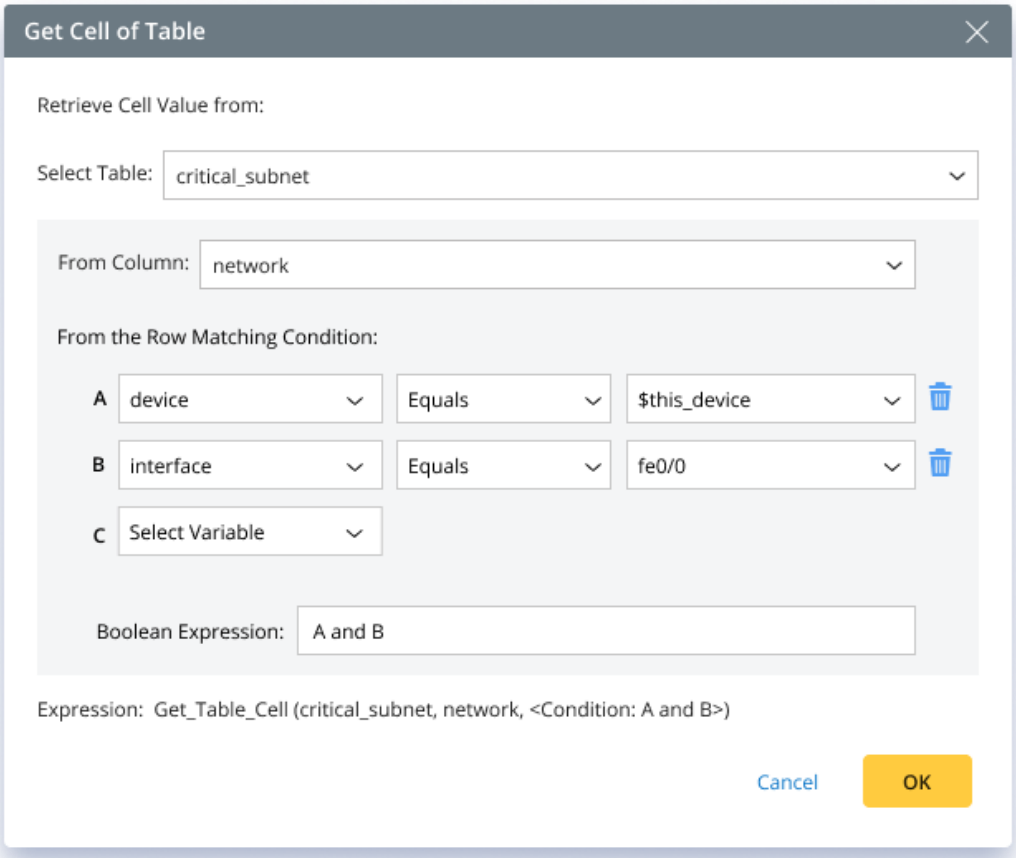
- **Delete table row**: Delete table rows (Frist Row or All Rows) matching condition.



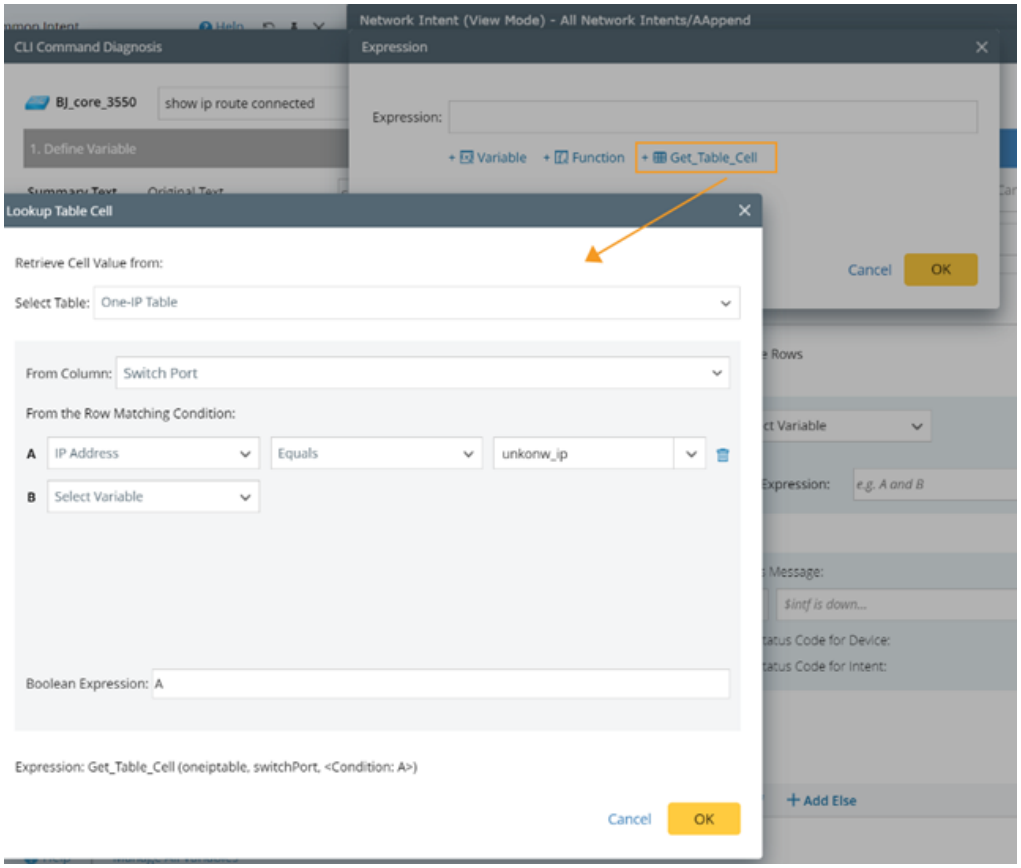
- **Update table row**: update the matching condition row (Frist Row or All Rows) to a certain value.



- **Query table cell**: Get the first table cell to match the specified conditions.



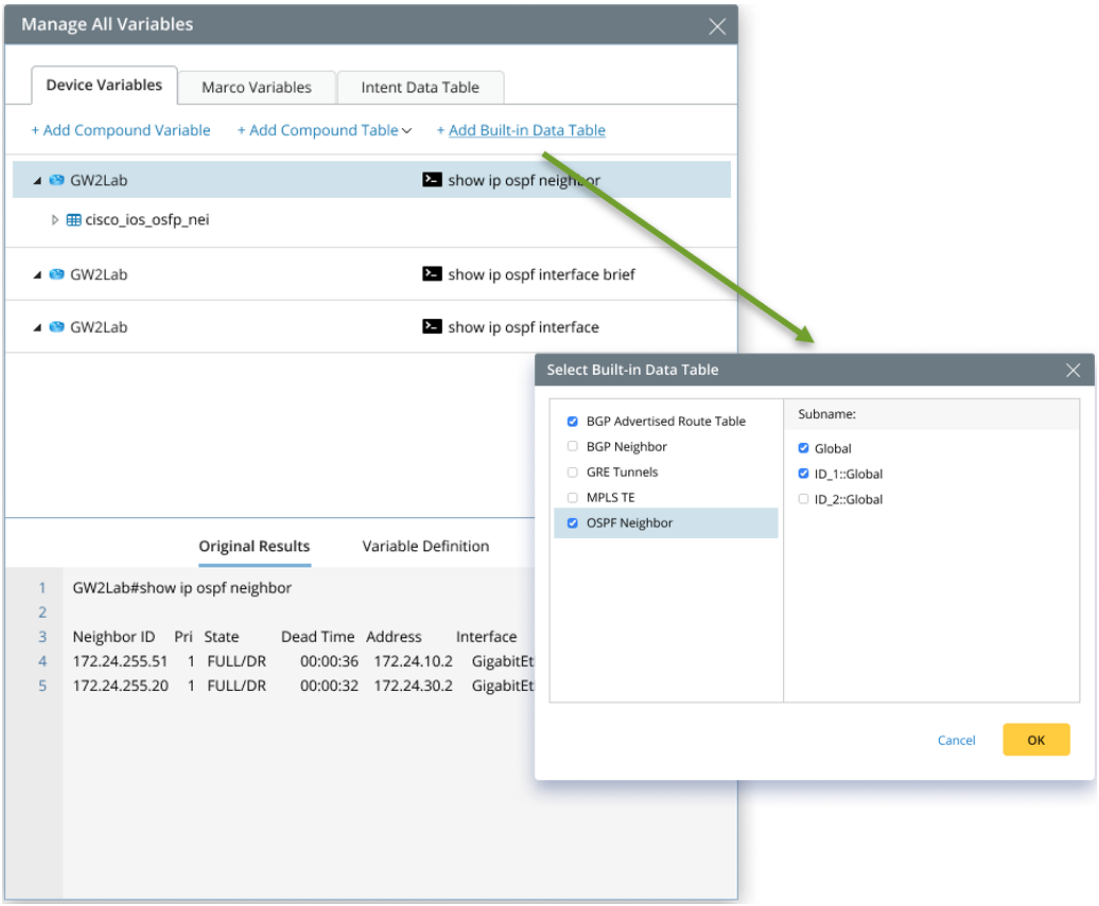
For example, users can get the **Switch Port** from **One-IP Table** by end system ip address from the ServiceNow Ticket.



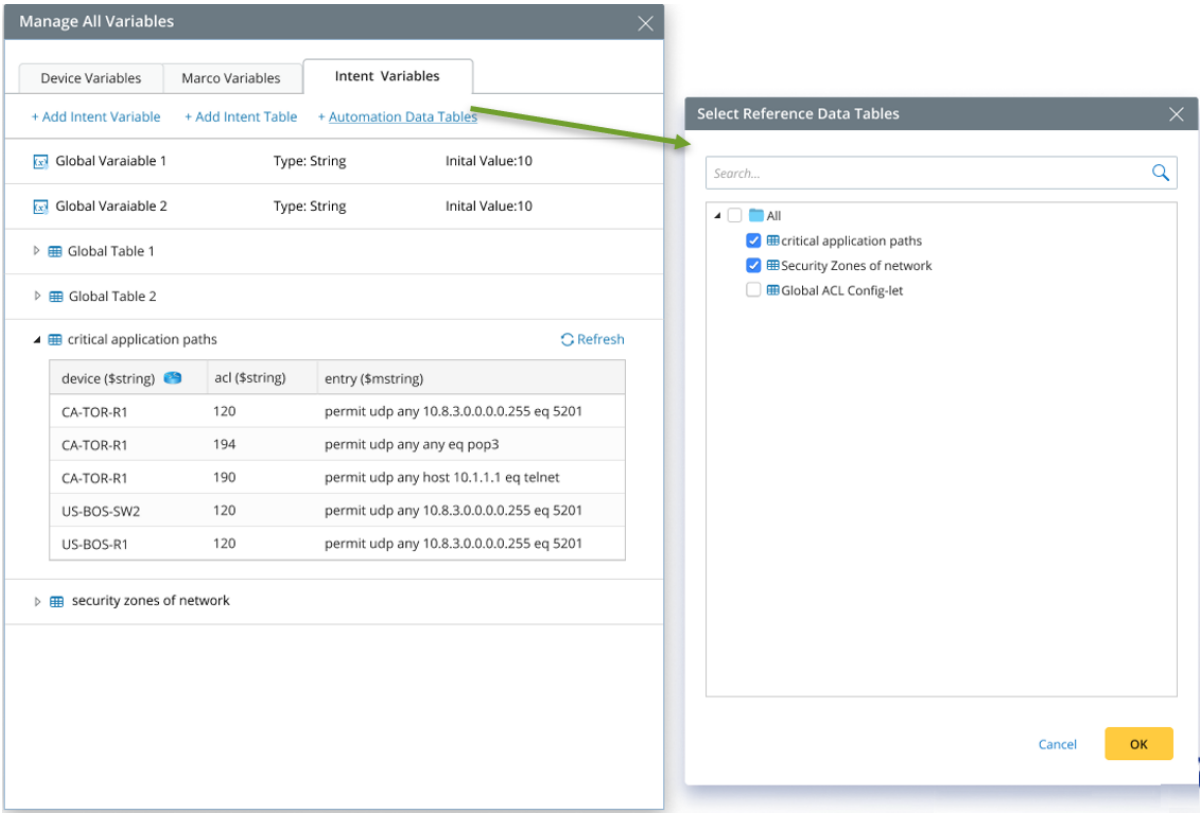
4.4.5 Support Built-in Table and ADT

In R11.1, Intent can use the built-in data tables (e.g., Route Table, NCT) and ADT to define diagnosis:

- Loop and analyze the built-in table data. For example, loop the OSPF Neighbor table to check the neighbor state.



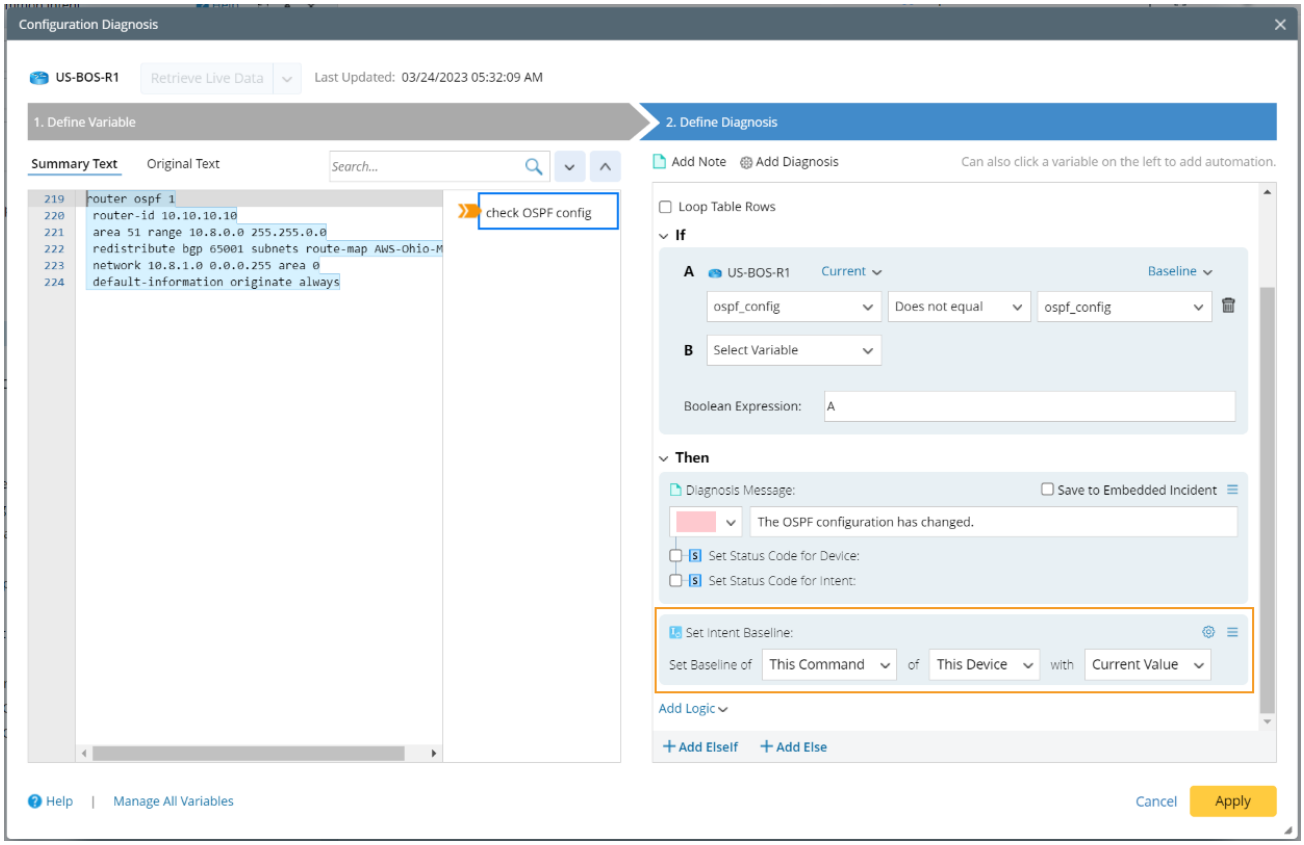
- Handle ADT Update and ADT-based follow-up intent, and loop ADT to get the multi-line text for device config-let compliance check.



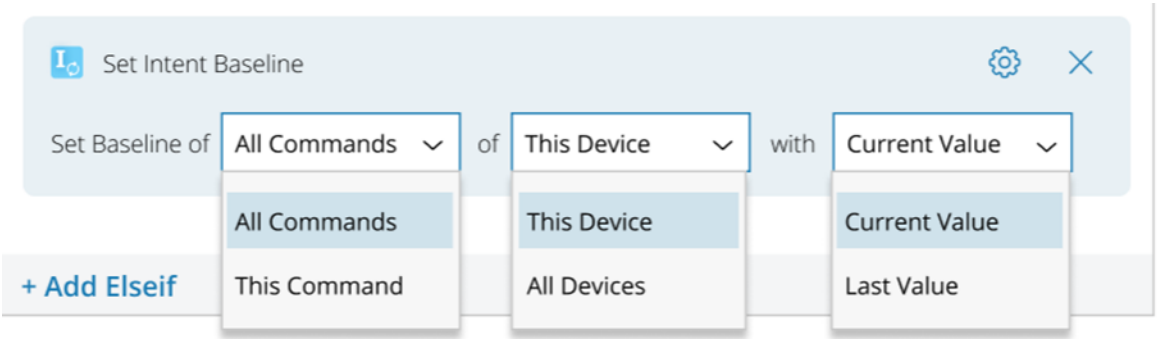
4.5 Control the Update of Intent Baseline by Logic

R11.1 supports updating the baseline in the intent diagnosis, which allows users to reset the baseline with conditions such as:

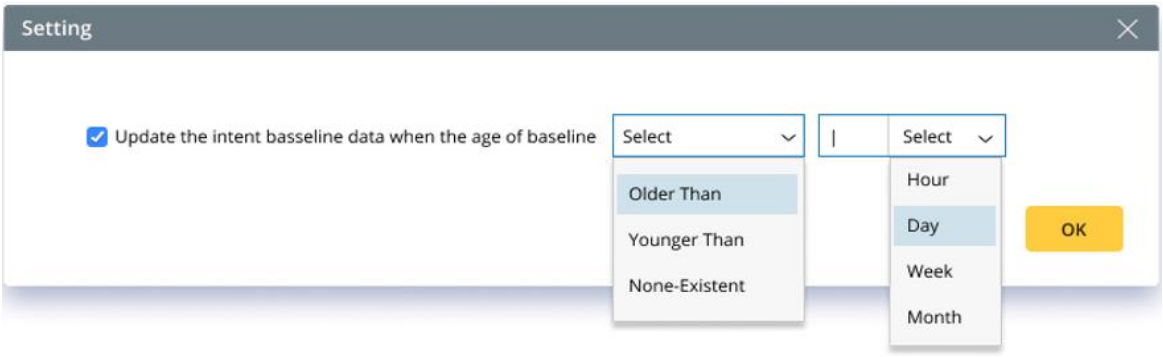
- If configuration changes are detected.
- If the baseline has not been updated for a long time.
- If the baseline is not set at all.



Users can select batch updating intent baseline based on command and device.



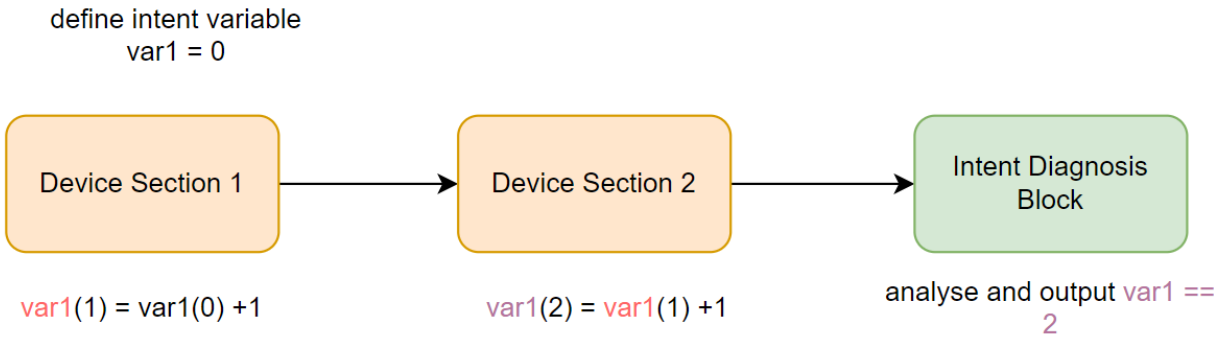
R11.1 also adds a setting to define whether to update the intent baseline based on the baseline age.



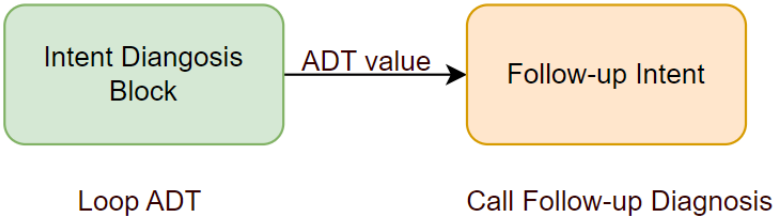
4.6 Intent Variable and Intent Diagnosis Block

Intent Variable and Intent diagnosis block provide a way to support:

- Analyze logic on a few previous device sections and use the intent diagnosis block to loop through the intent table to output the final diagnosis, including NI status code, diagnosis message, CSV report, or trigger follow-up.



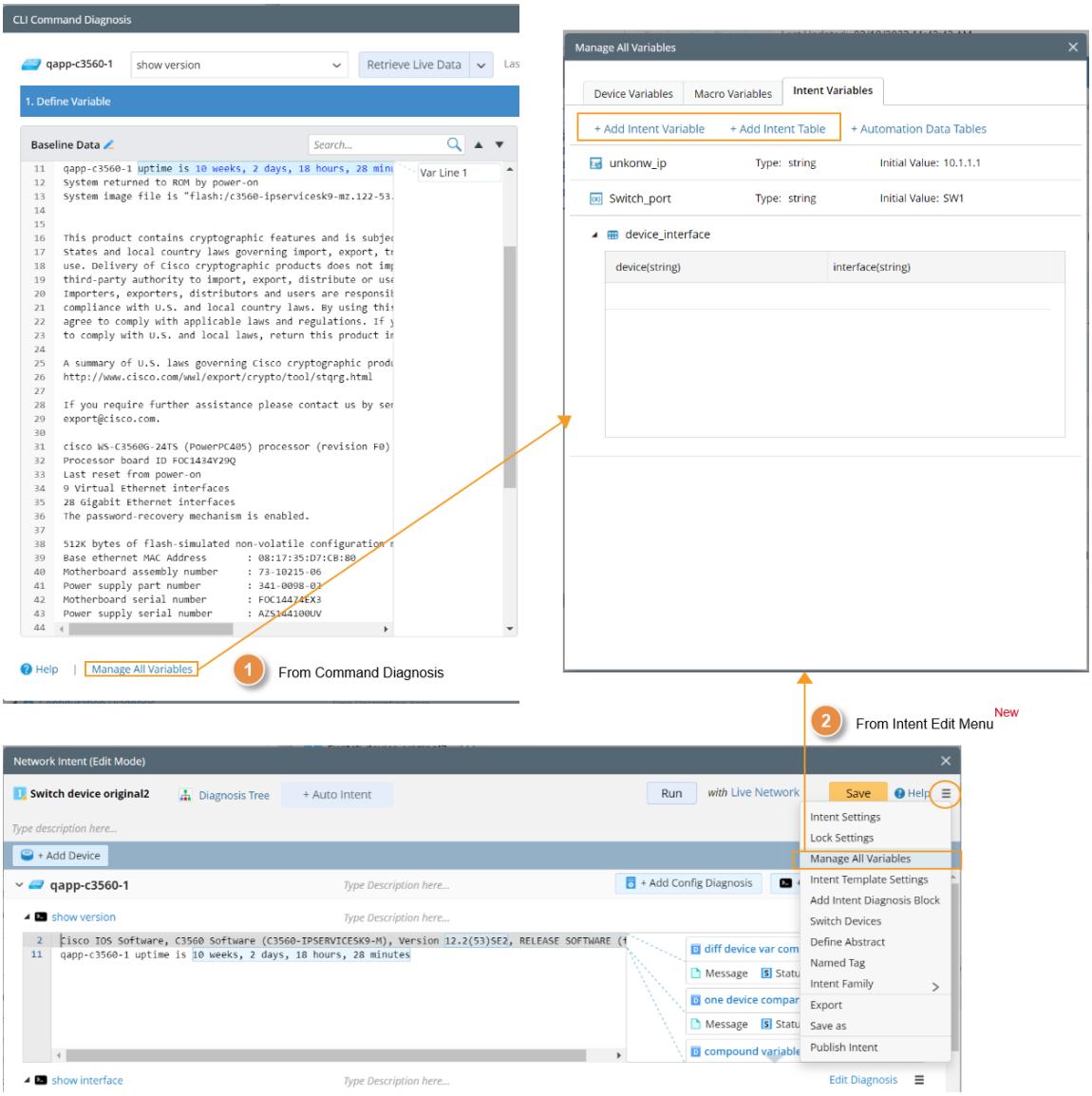
- Intent-level of diagnosis - handle ADT update and ADT-based follow-up intent (using Macro and Intent Variable to match intents inside ADT).



4.6.1 Intent Variable

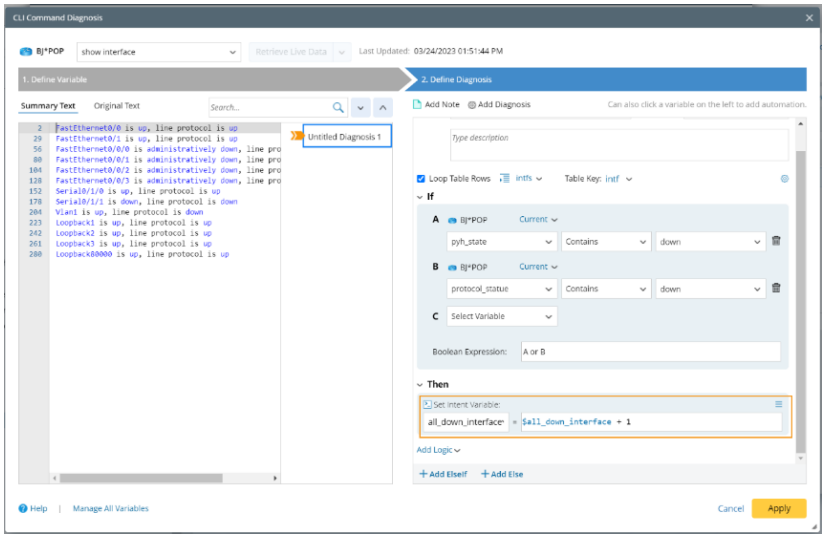
R11.1 allows users to define an intent-level variable (string, number, or table variable) that can be reached and modified in the intent:

- Add/remove/update the table row and use the data row of this table.
- Define the singleton variable (string or number) with an initial value and update the value in the device diagnosis.
- Create Intent-level Macro Variable to take the external incoming key metric as the input (e.g., ServiceNow ticket id) and use the metric value in the follow-up intent template.

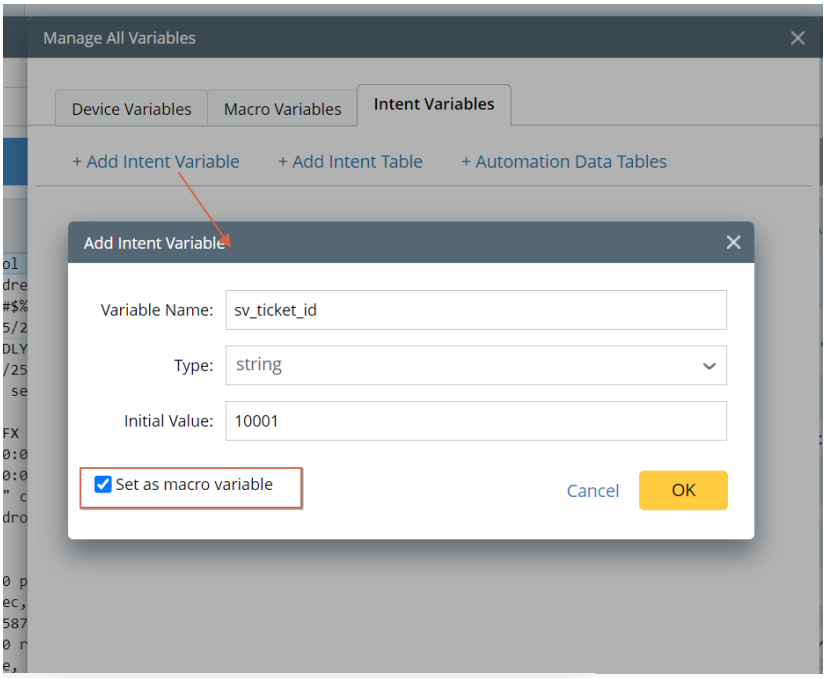


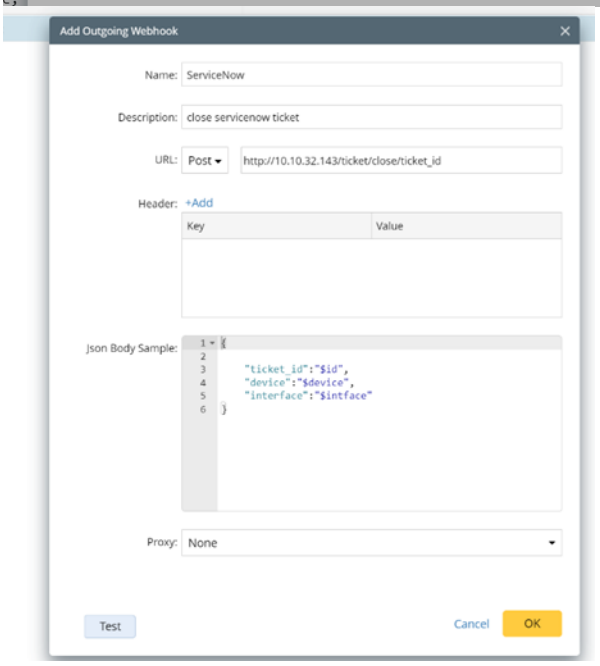
Use the intent variable to:

- Store the device statistic data, which can be referred to by the intent status code. For example, Count the shutdown interfaces on the devices:



- Receive external incoming key metric (e.g., ServiceNow ticket id) and pass the metric value in the follow-up intent template. For example, check the interface state and close the ticket from ServiceNow.

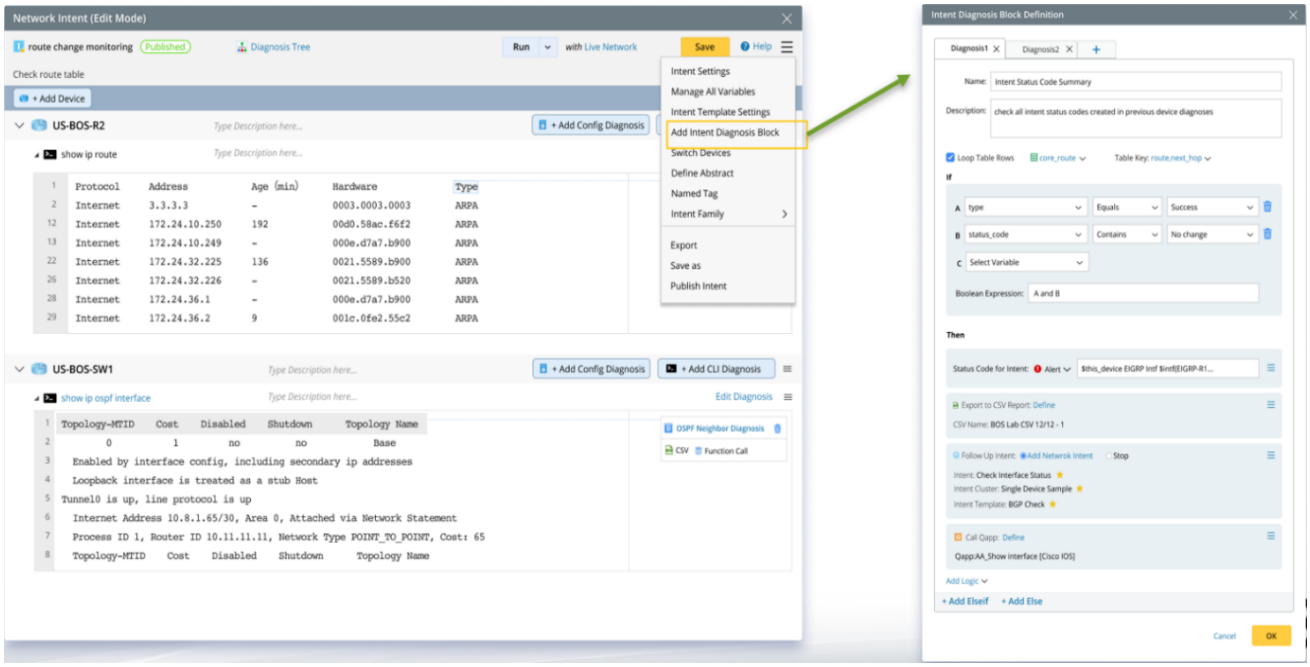




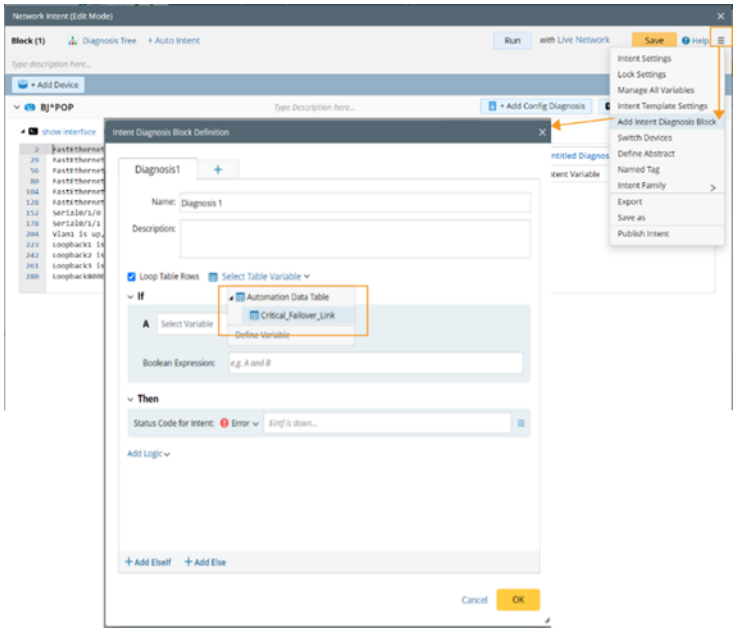
4.6.2 Intent Diagnosis Block

Difference between **Intent Diagnosis** and **Device Diagnosis**:

- Each intent can have only one intent diagnosis block, which will be displayed and executed after all device diagnoses.
- Intent Diagnosis has no parser and command baseline data and includes diagnosis blocks (if/then/else).
- Device diagnosis can access all variables of the current device and intent variables. Intent diagnosis only can access intent variables.
- Device diagnosis can use all supported logic, and intent diagnosis can only use the device-independent logic, including Intent Status Code, Follow-up Intent, Send Email, Call Webhook API, Call Qapp, and Export CSV Report.



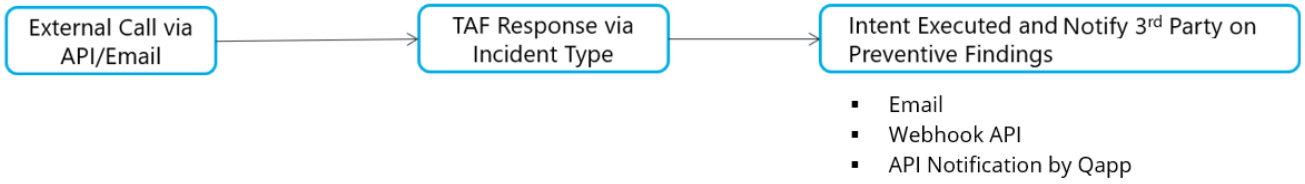
Users can use ADT as input to define an intent diagnosis. For example, check failover link change.



4.7 Programmable Notification to 3rd-party system

R11.1 adds three diagnosis logics: **Send Email, Call Webhook API** and **Call Qapp** so that the intent diagnosis results can be sent to users or 3rd systems via **Email, Webhook API,** and **Qapp**. (e.g., email ServiceNow to create a ticket or pass the data to Splunk via Webhook API).

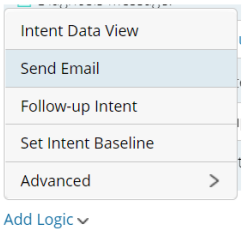
The basic flow of sending diagnosis results is as follows:



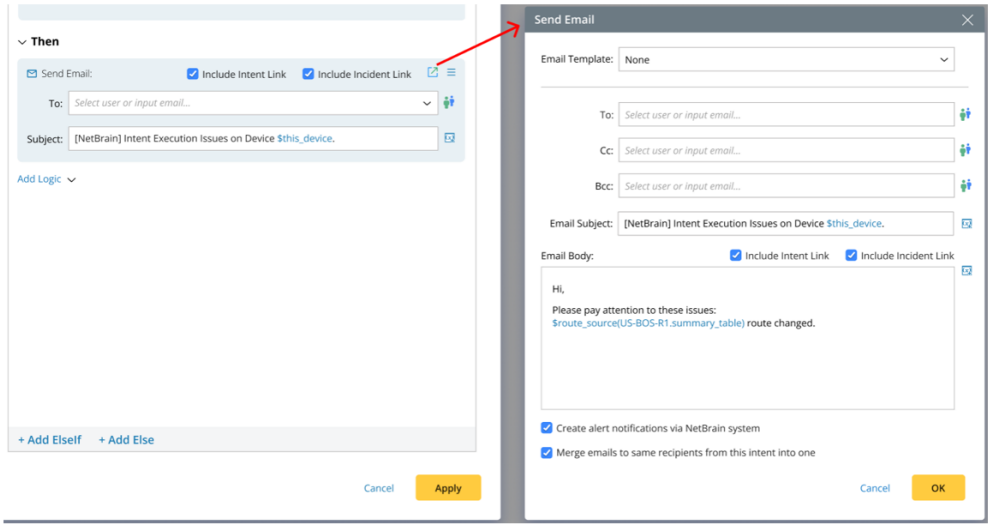
4.7.1 Send Email

R11.1 provides an entrance for adding **Send Email** block in NI diagnosis. Users can define the email settings, such as **subject, body,** and **intent link**.

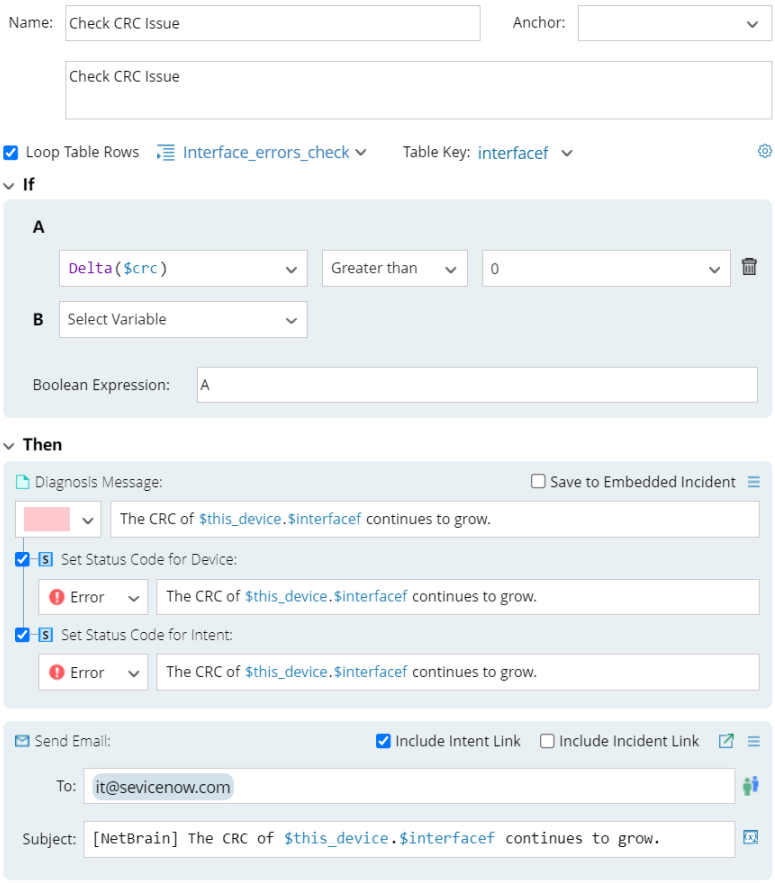
Entrance :



Email Settings:



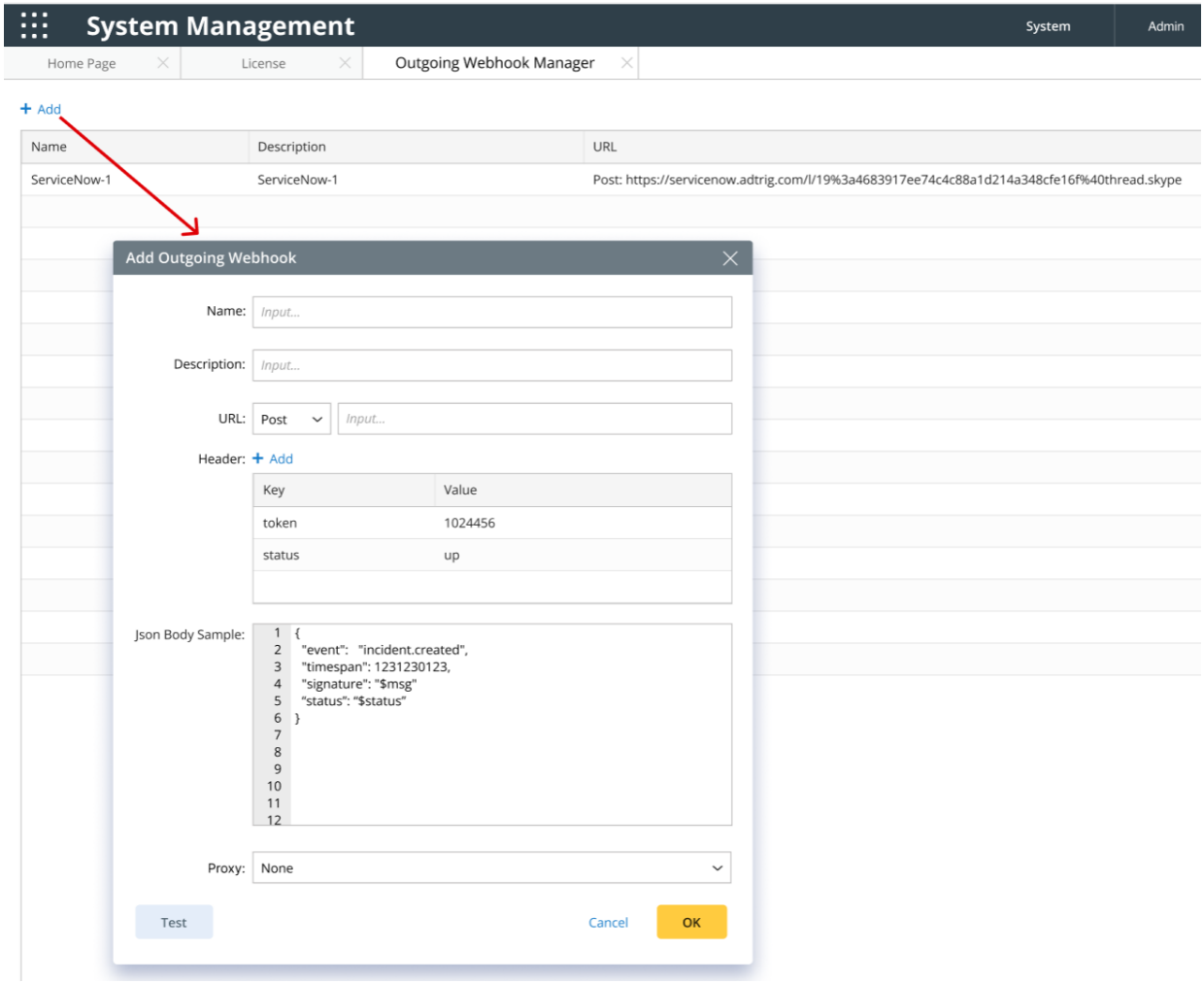
For example, send an email if the interface CRC error grows.



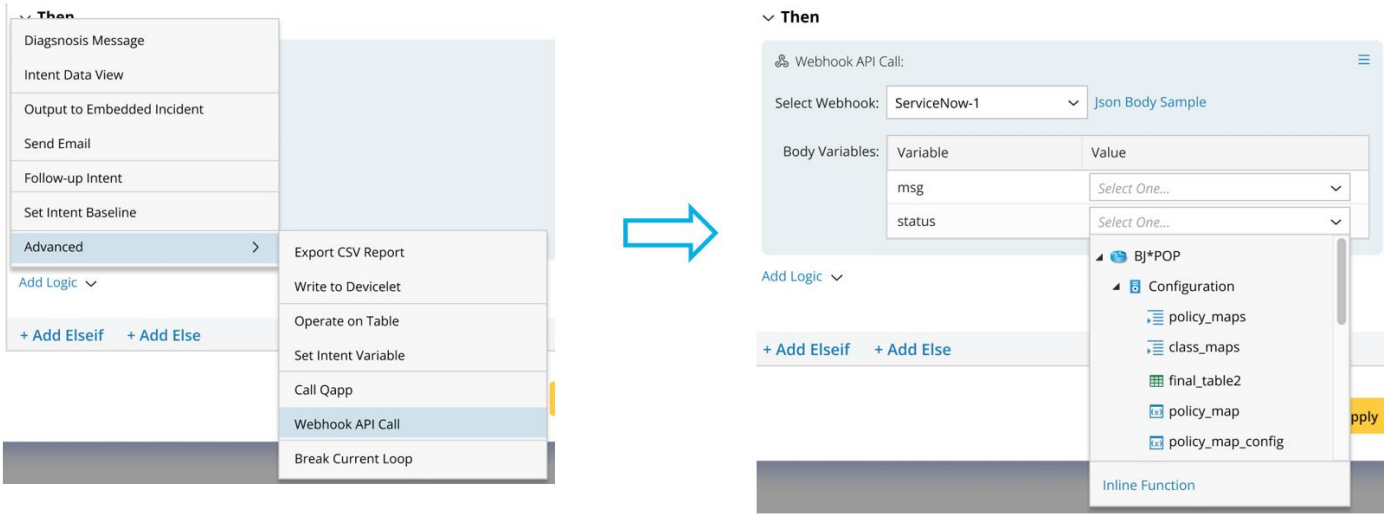
4.7.2 Call Webhook

Outgoing Webhook Manager is added in the System Management. Users can refer to the defined **Webhook** in the intent without knowing the technical details, such as the **URL** and **Header**, to send diagnosis results to

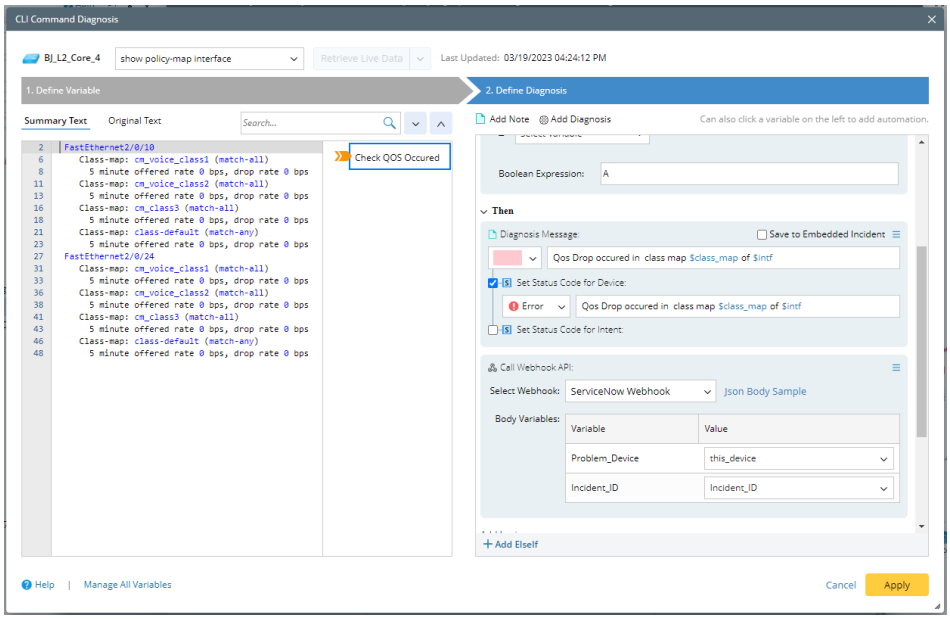
a 3rd party system.



Inside an intent, users can add a Webhook API Call block.



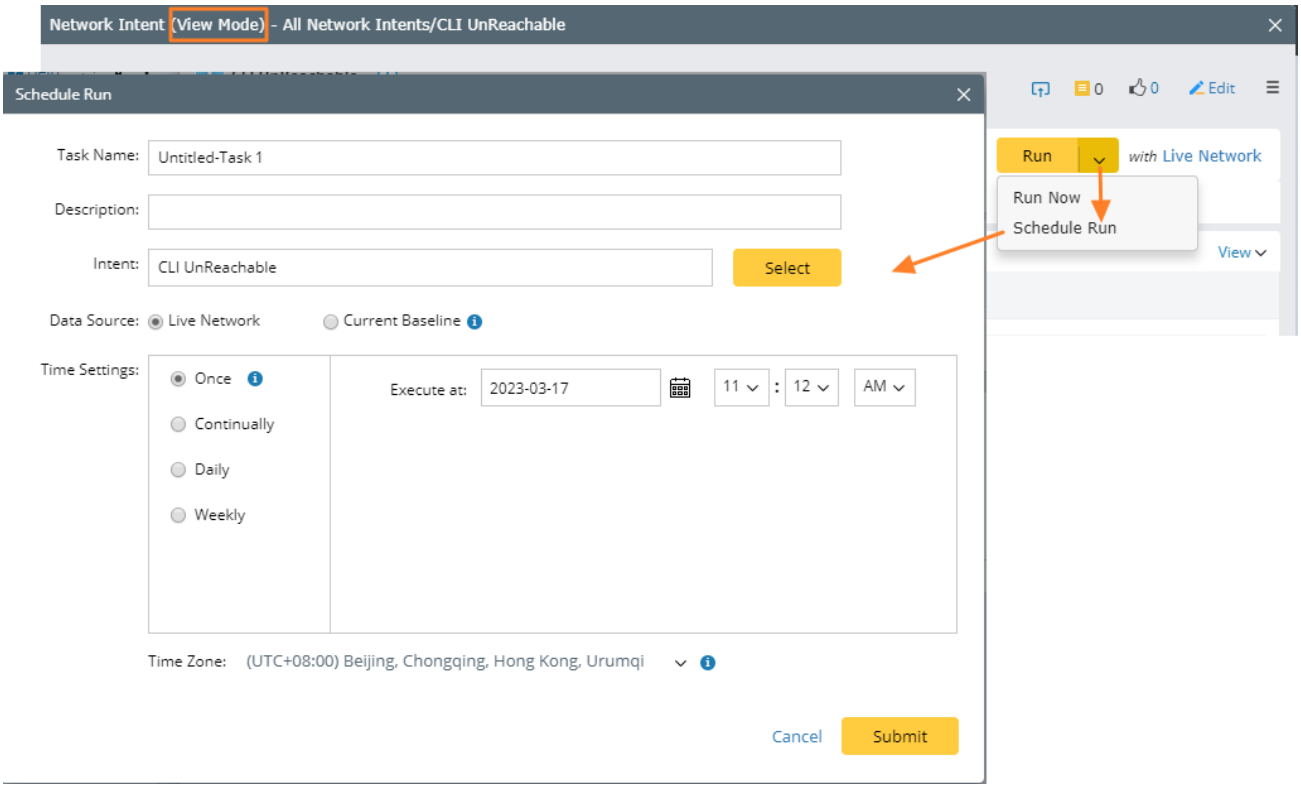
Use **Webhook Call** to Send Alerts to External Ticket System, for example, check application performance and return found issues to ServiceNow via calling webhook.



4.8 Schedule Intent

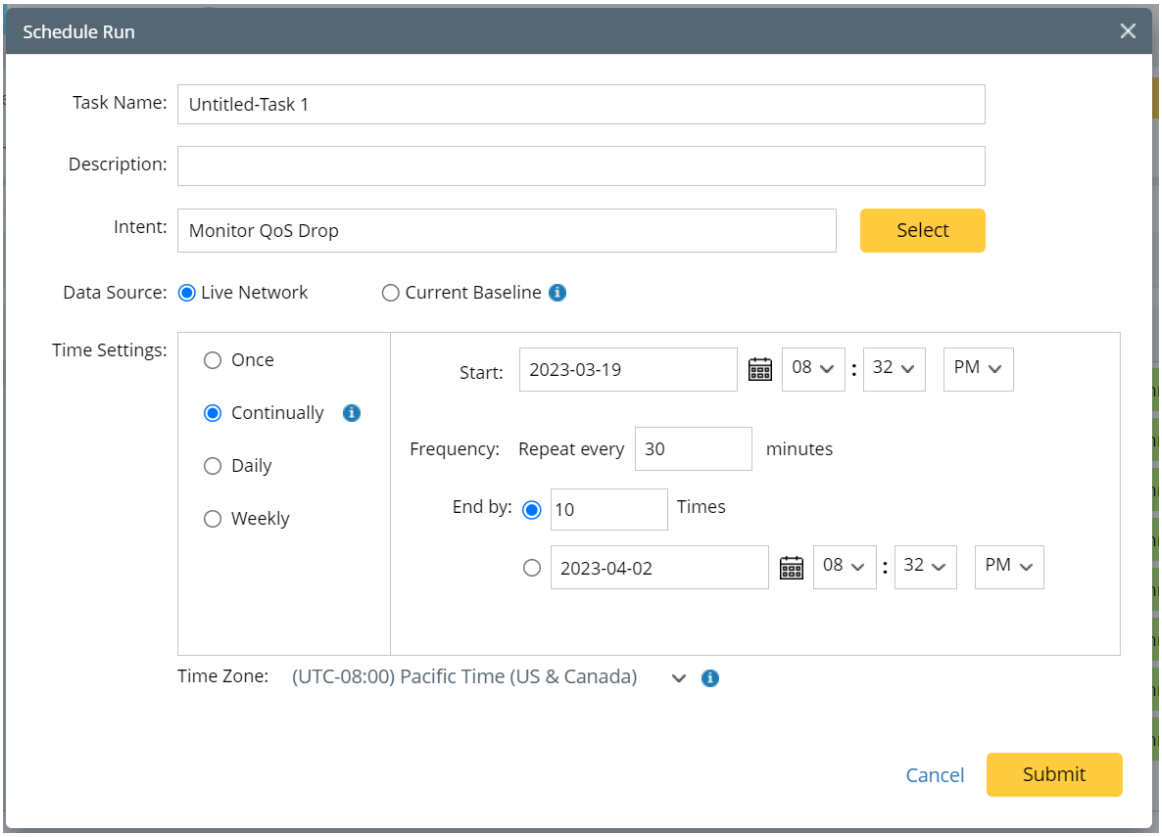
R11.1 supports setting the schedule to run intents. Users can schedule an intent to diagnose the transient problems, such as:

- Schedule an Intent "**Monitor ASA Failover status**" to continuously monitor the core application path.
- Schedule an Intent to regularly check whether the standby switch has been configured with the same ACL as the active switch.
- Schedule an Intent to ping devices before regular network changes or identify variation after network changes, then send email alerts to engineers.



When a user defines a scheduled task:

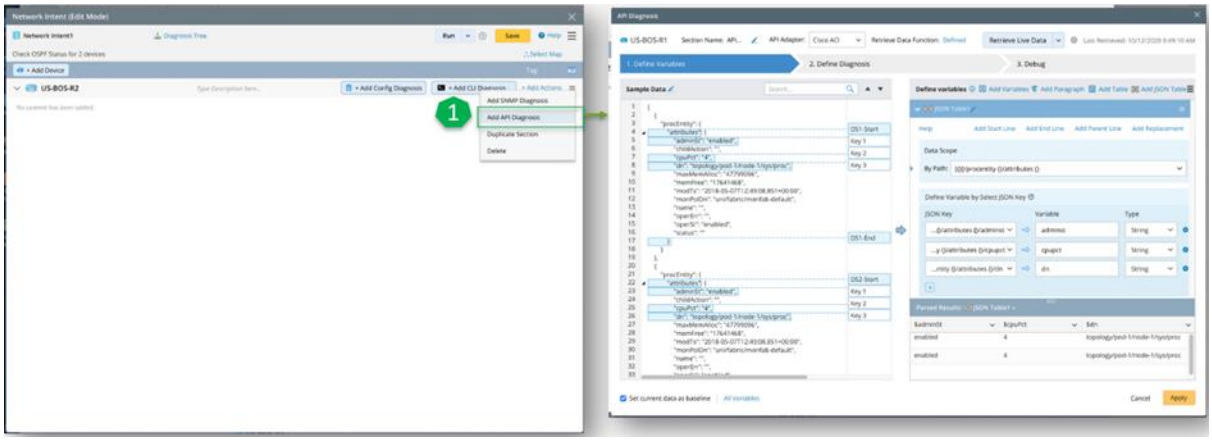
- One scheduled intent task can only have one intent.
- Able to set the data source, execution time, and frequency (**Once, Continually, Daily,** and **Weekly**).
- For the **Continually** settings, users must set the **end by times** and **end by date** (cannot be longer than 2 weeks from the start). If the **Times** checkbox is selected, the **Times*Frequency** must be less than 2 weeks.



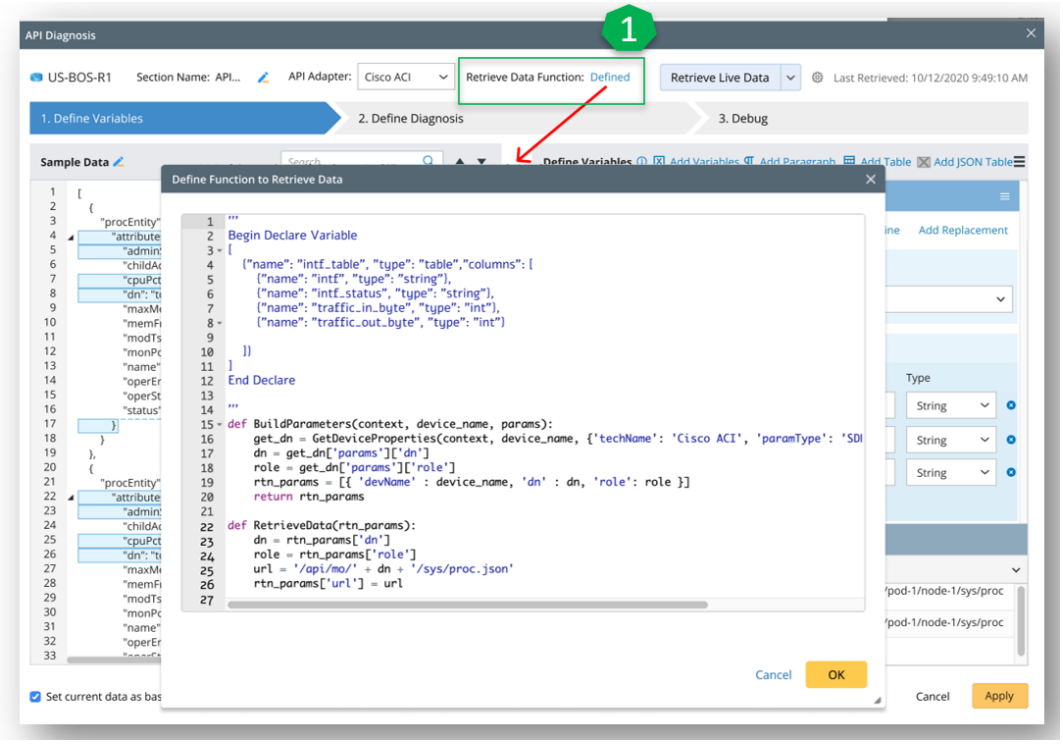
4.9 API Diagnosis in Intent Definition

The Intent in the previous version supports the data retrieved from CLI and SNMP but not from APIs, which limits the intent used for the public cloud and SDN technologies. R11.1 adds support for API data, which enables the Diagnosis of SDN/Cloud Network and SPOG Data View from an external system via REST API (such as Splunk, SolarWinds, and ServiceNow).

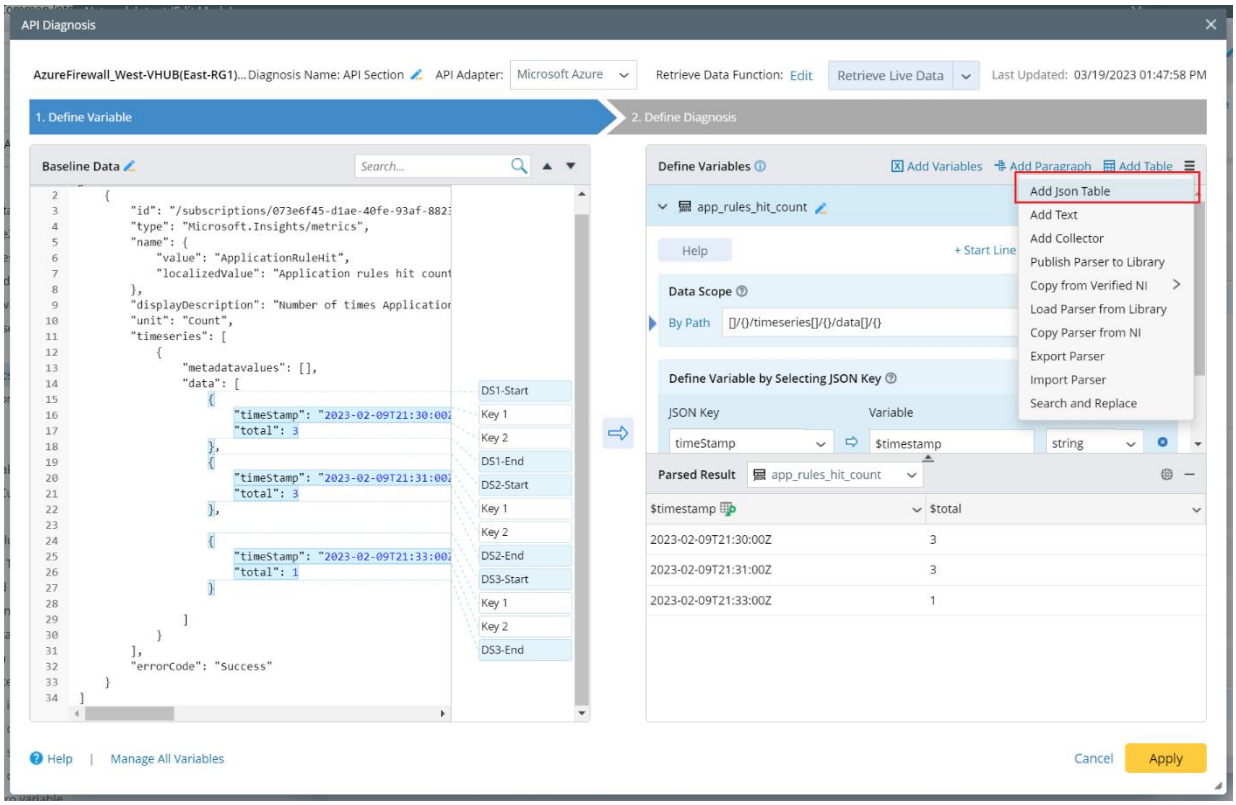
R11.1 adds a new API diagnosis type to support API diagnosis in intent. After defining the API diagnosis, a new API Diagnosis section will be added:



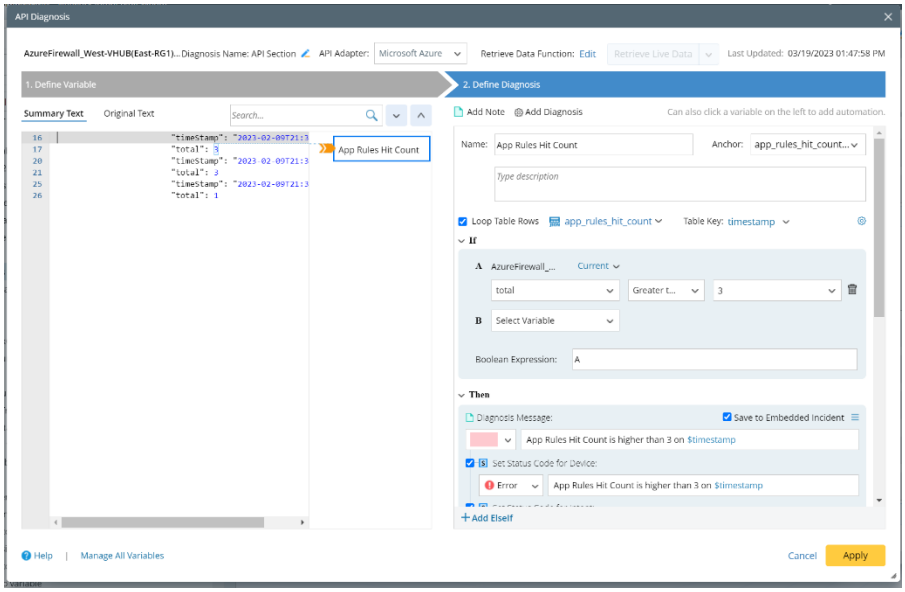
1. Add Retrieve Function to Retrieve data by REST API: In the API diagnosis definition, after selecting API Adapter for a specific vendor, users can define the retrieve data function to get the network data of SDN/Cloud/Other External System via REST API. The **Retrieve Data Function** is defined by Python Script, and it is defined the same as the advanced API parser.



2. New **JSON Group** to parse the JSON Result: After retrieving the JSON data by the defined retrieve data function, add a new Parser Group, **JSON Group**, to parser the JSON result. JSON Group will read all the keys from the JSON result and convert the JSON result to Table Format by selecting the JSON key.



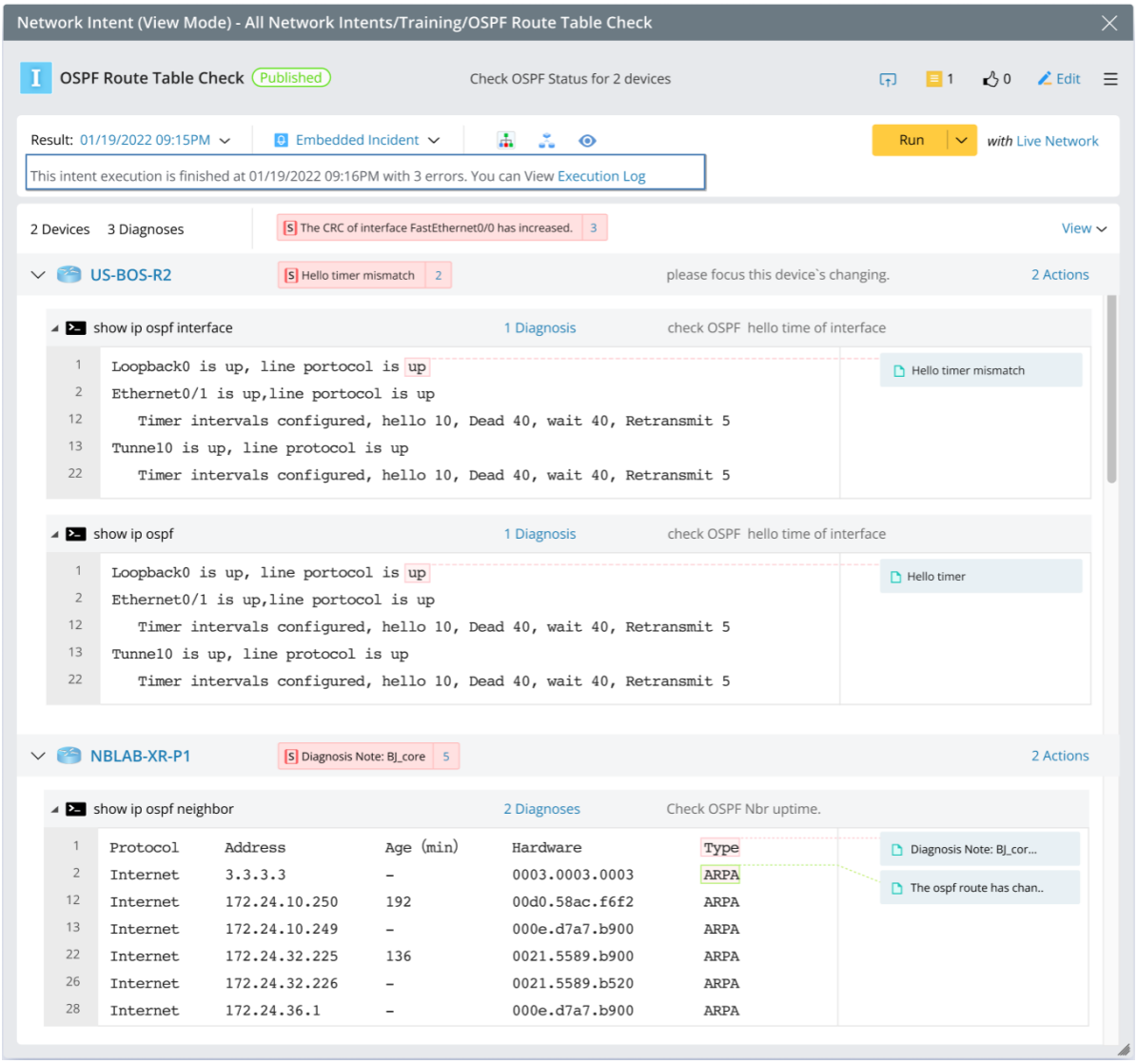
For example, users can build an intent with API diagnosis to monitor the number of times application rules are hit on all Azure firewalls.



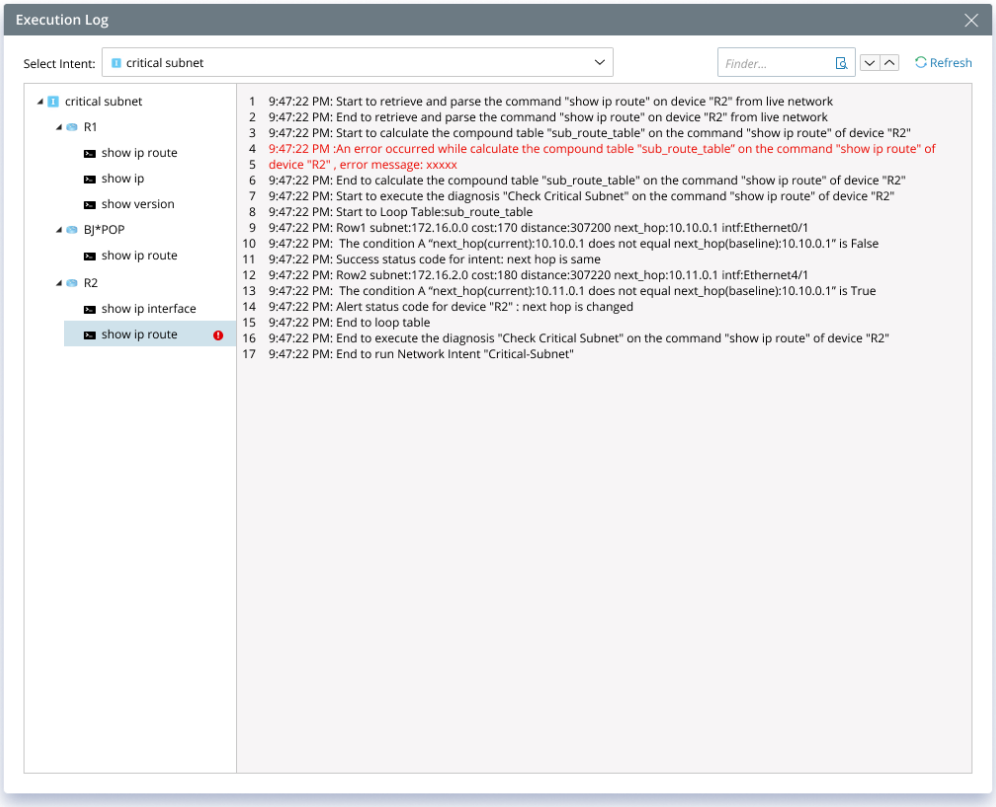
4.10 Simplified Intent debug

R11.1 removes the old Intent Debug function since it is hard to use and maintain. Instead, R11.1 enriches the Execution Log in intent Debug Mode.

- Support the centralized viewing of home intent and follow-up intent/intent template/intent cluster execution log for multi-stage diagnosis.



- Support outputting richer error logs for troubleshooting intent execution.

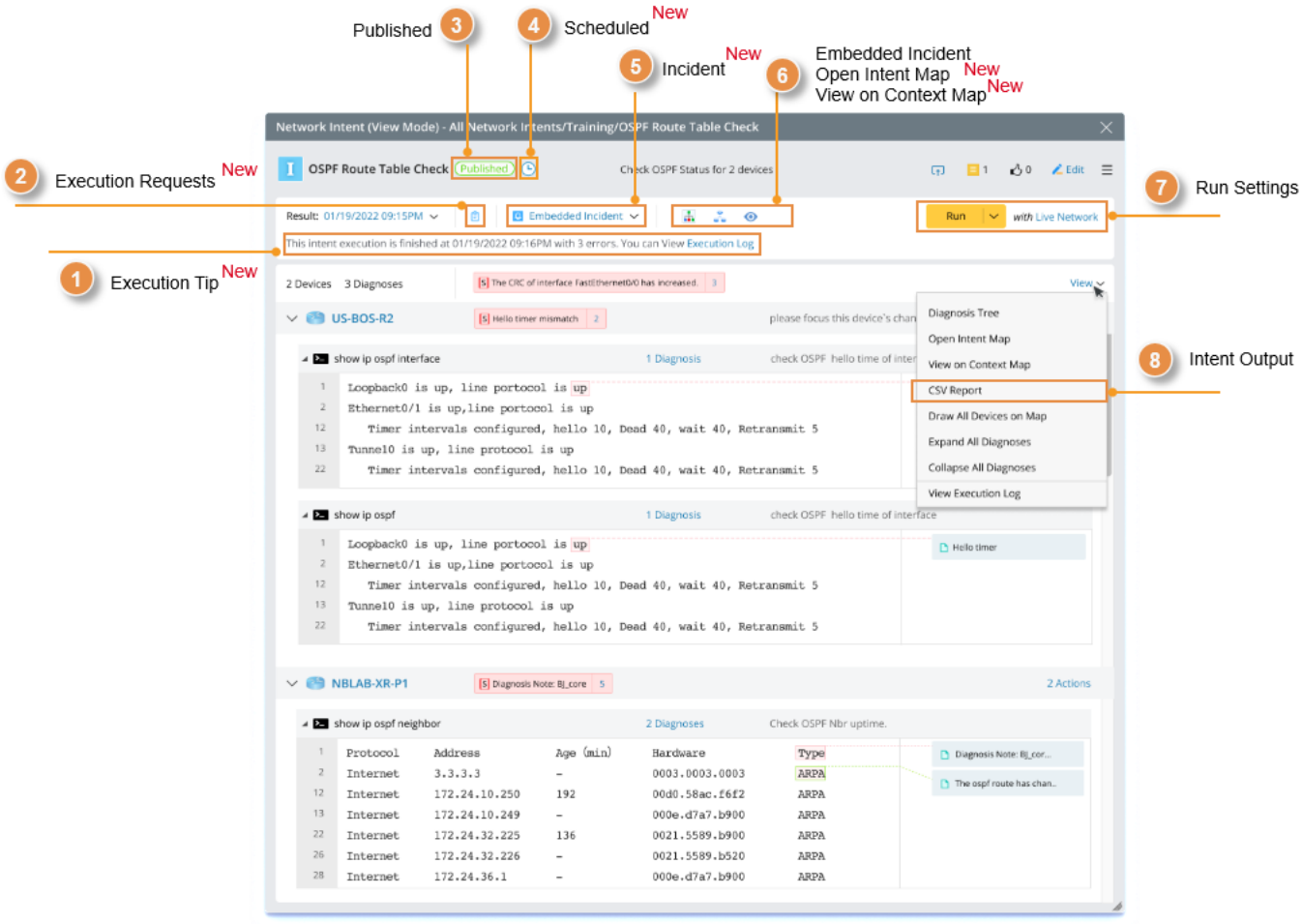


- Support outputting richer intermediate raw data of intent execution process for debug intent.

4.11 Other Intent's Improvements

4.11.1 Intent View

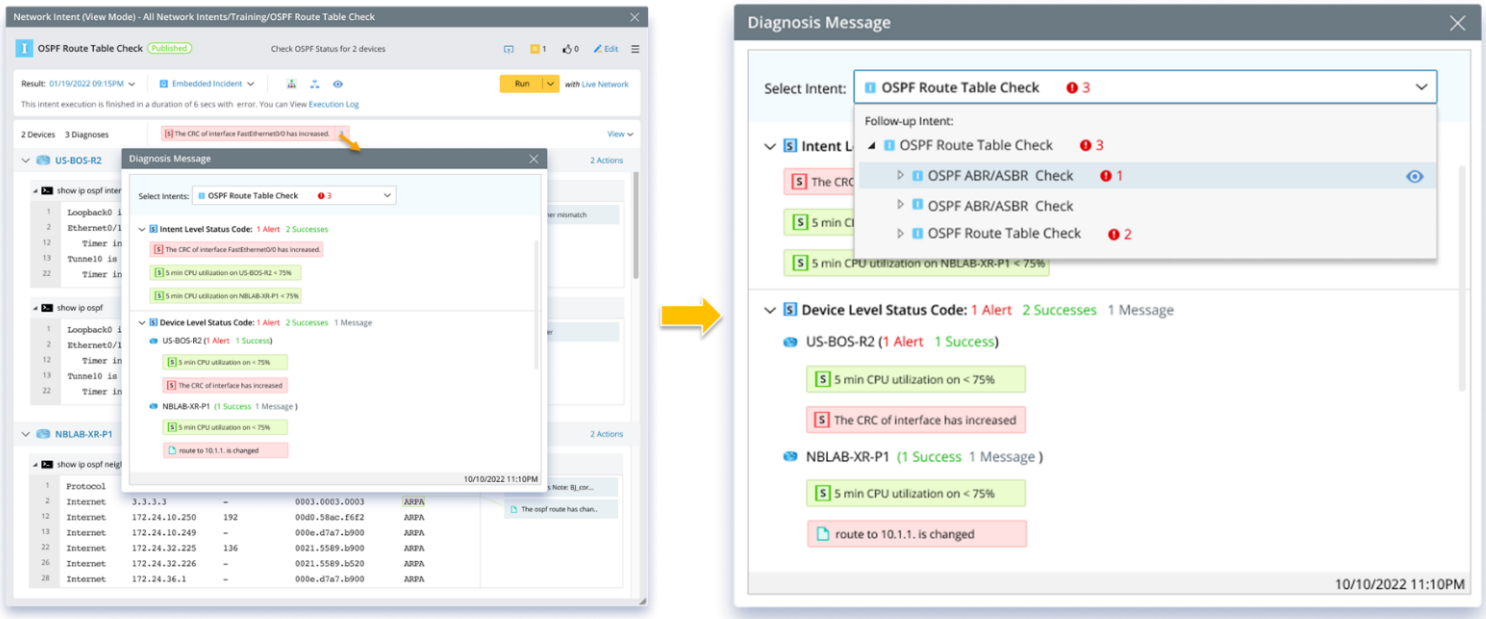
In R11.1, the UI layout of Intent View is optimized. The first layer UI displays the current running status of NI and frequently used operations. Moreover, the CSV Report is moved to the secondary menu to keep the main interface clean.



4.11.2 View Diagnosis Message

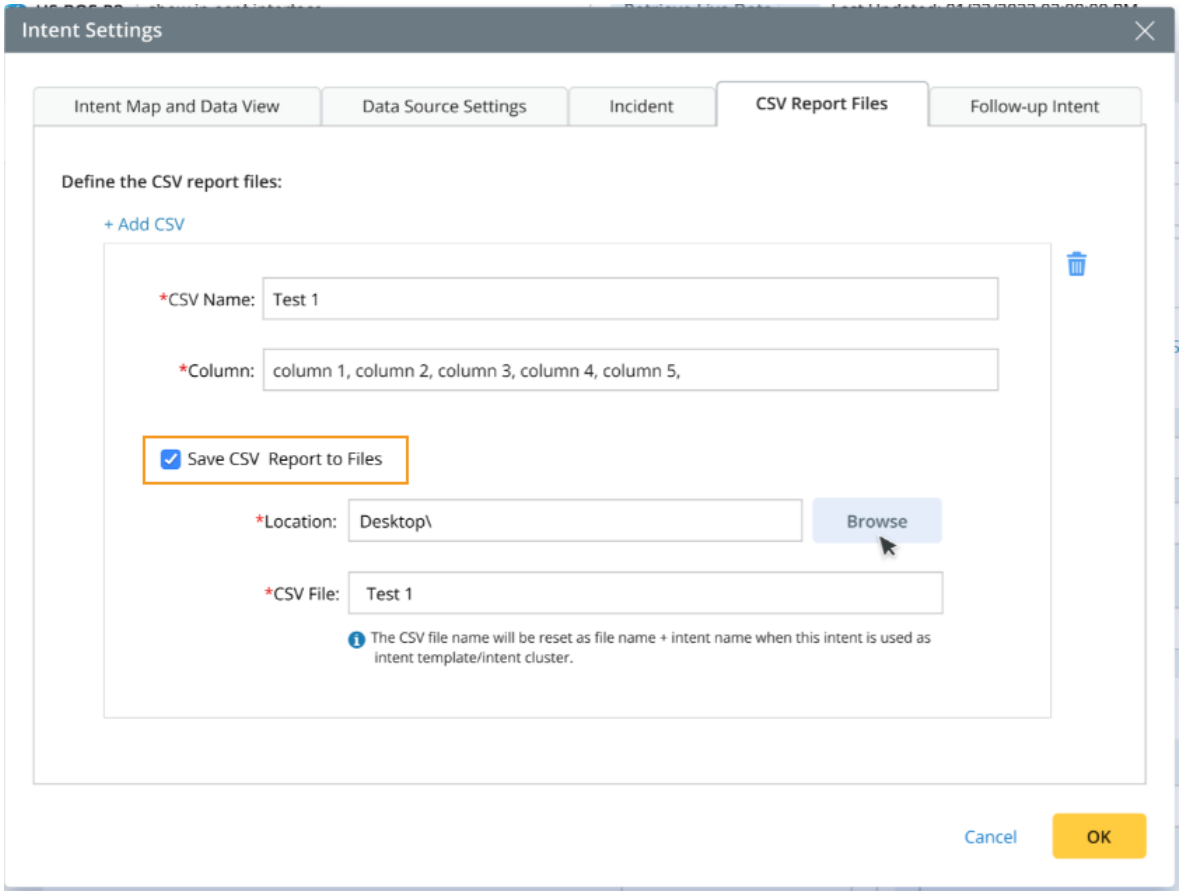
The Diagnosis Message dialog supports switching to view the status code and message generated by the execution of Home Intent and Follow-up Intents, and clicking the Open button can open the follow-up intent

to view the details:



4.11.3 Export CSV Report to Files

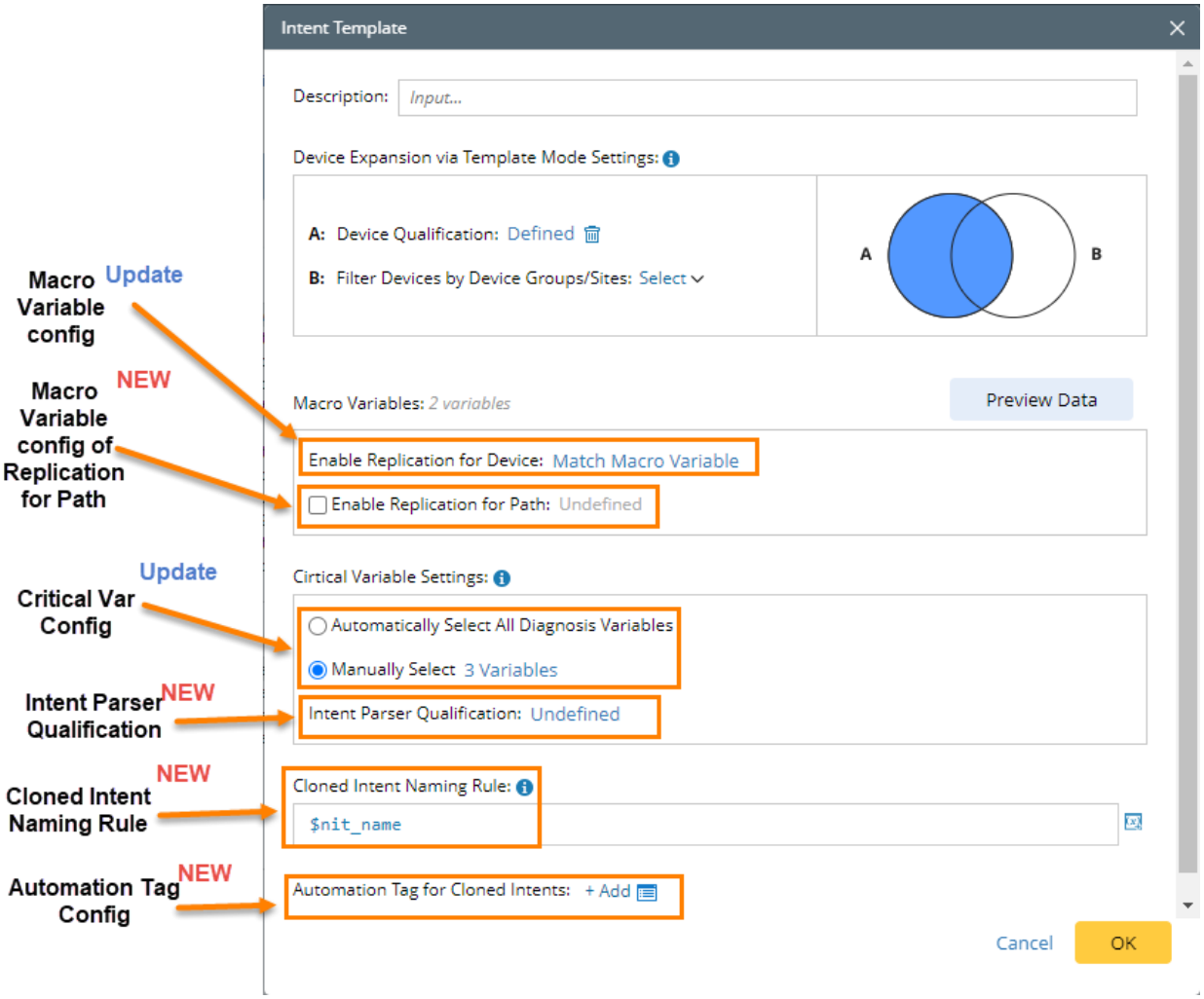
In R11.1, users can specify the NetBrain Files folder for CSV reports from intent.



4.12 Intent Replication Improvements

The configuration logic in the Intent Template Setting is improved to be more intuitive.

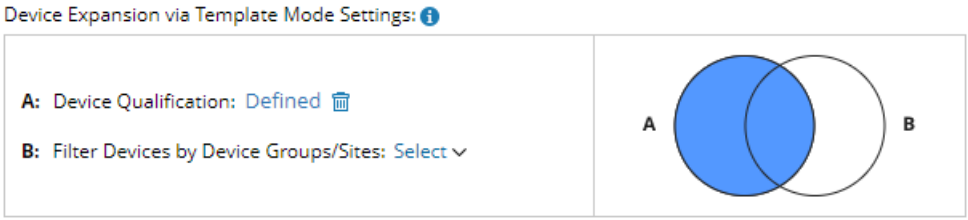
R11.1 makes the following improvements to the Intent Template:



- **Macro variable config**: config how to define Macro variables using ADT.
- **Macro Variable config of Replication for Path**: enable replication for path and how to pass a variable value to a Macro variable using built-in path variable (source, destination, source port etc.)
- **Critical Var Config**: define how to match a seed device command based on auto-test results of critical variables.
- **Intent Parser Qualification**: define intent parser qualification to filter qualified devices based on qualification to improve unnecessary command testing.
- **Cloned Intent Naming Rule**: define how cloned intent can be named with the variable value of Macro variable and path variable values.
- **Automation Tag Config**: configure automation tag for cloned intents.

4.12.1 Device Scope Configuration

For Device Scope configuration, the existing **Device Qualification** and **Filter Devices by Device Group/Sites** options are merged into one section for configuration.

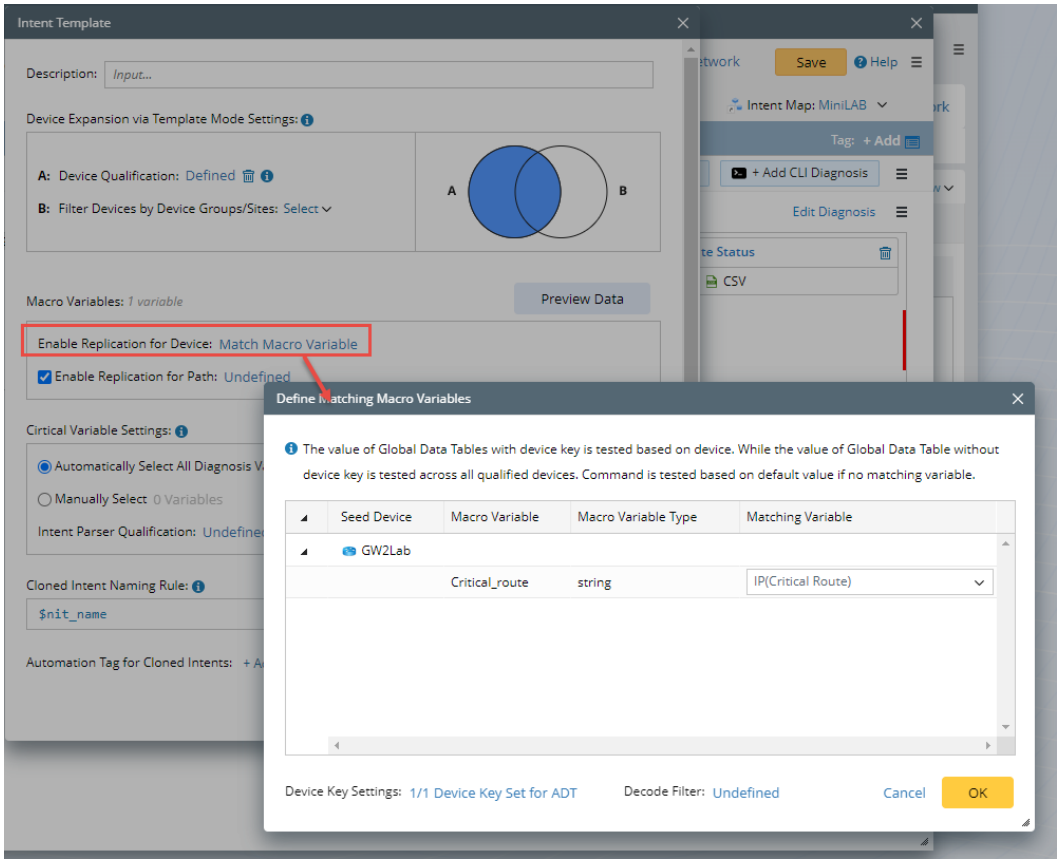


Both options **A** and **B** can be defined separately with the following options:

- **Define A only**: Filter Intent Template devices based on the **Device Qualification** of the configuration item.
- **Define B only**: Filter Intent Template devices based on **Device Group/Site configuration**.
- **Define both A and B**: Filter Intent Template devices based on the intersection of A and B.
- **Both A and B are undefined**: Intent Template cannot match any device.

4.12.2 Set Macro Variable

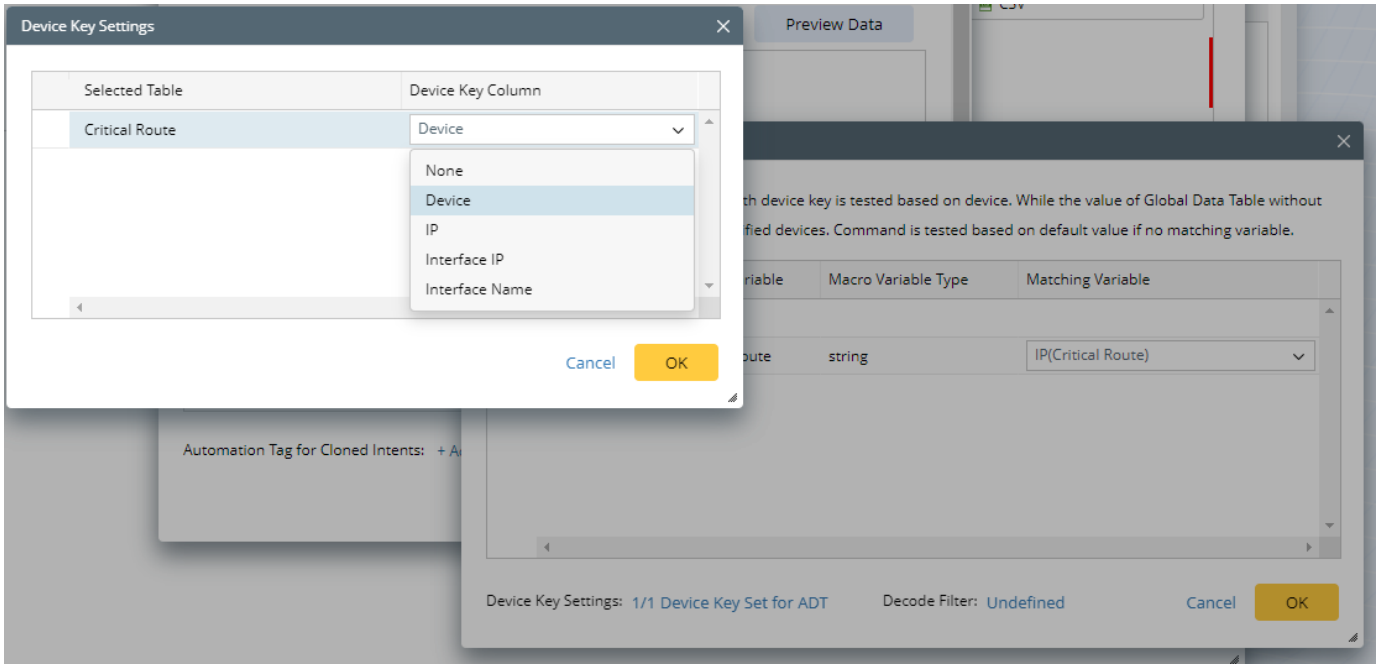
The method to set Macro Variable can be done only through GDT in the current version (the method of configuring Macro Variable through CSV file is removed. You can create a GDT table from a CSV file instead).



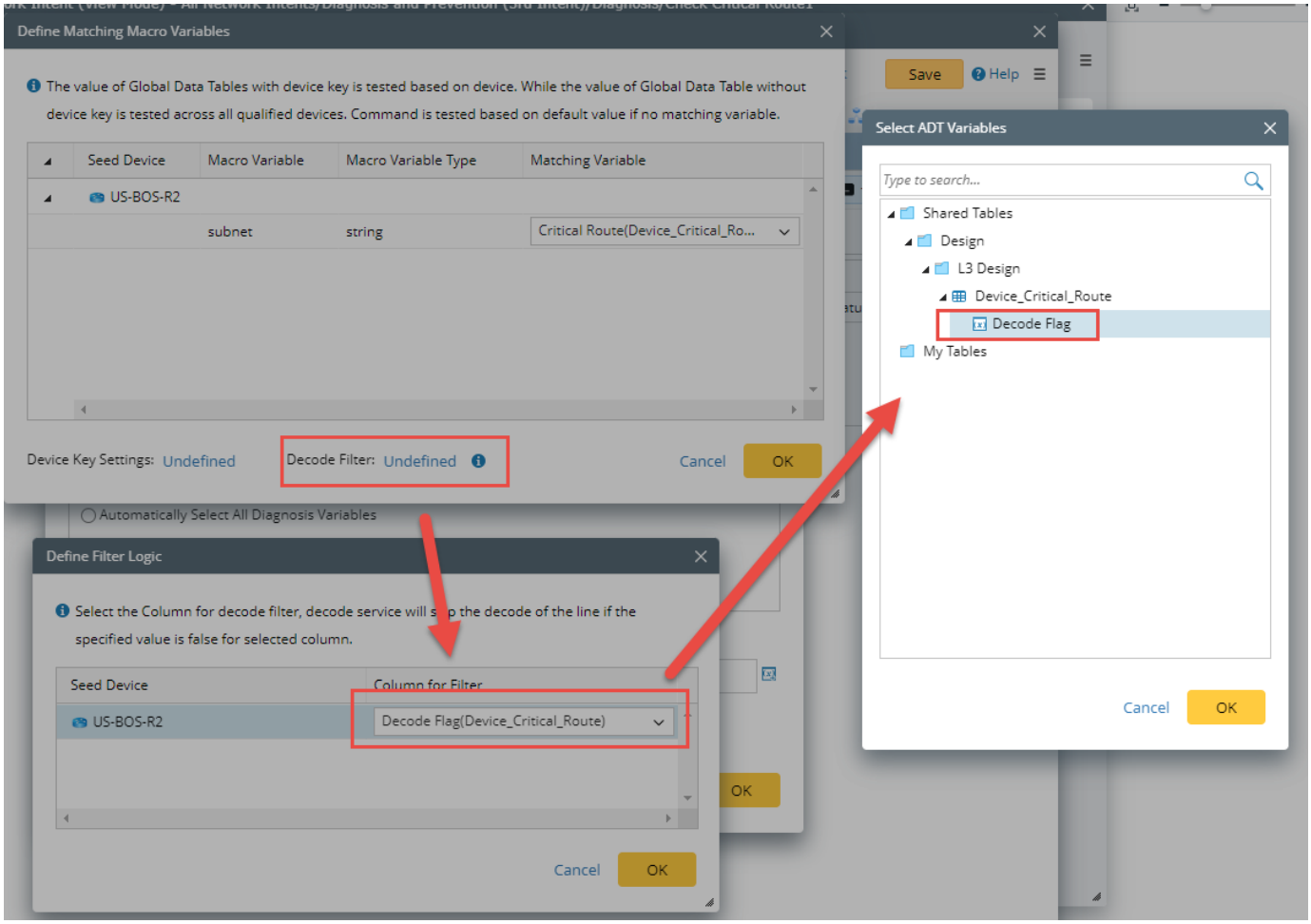
Define **Macro Variable** for the seed device as follows:

1. If an ADT column is selected and the device key is configured for the ADT, all Qualified Devices will use values of the column for each device defined in ADT to test against Critical Variables.
2. If an ADT column is selected and no Device Key is configured for the ADT, all Qualified Devices will use all values in the ADT column for Auto Test to test against Critical Variables.
3. If no ADT column is selected, the default value in the Seed Intent will be used to test the effectiveness of the command for all Qualified Devices (without going through the Critical Variable check).

By default, the system will set the first available device/interface column as the default device key column. You can manually modify the device key column settings.



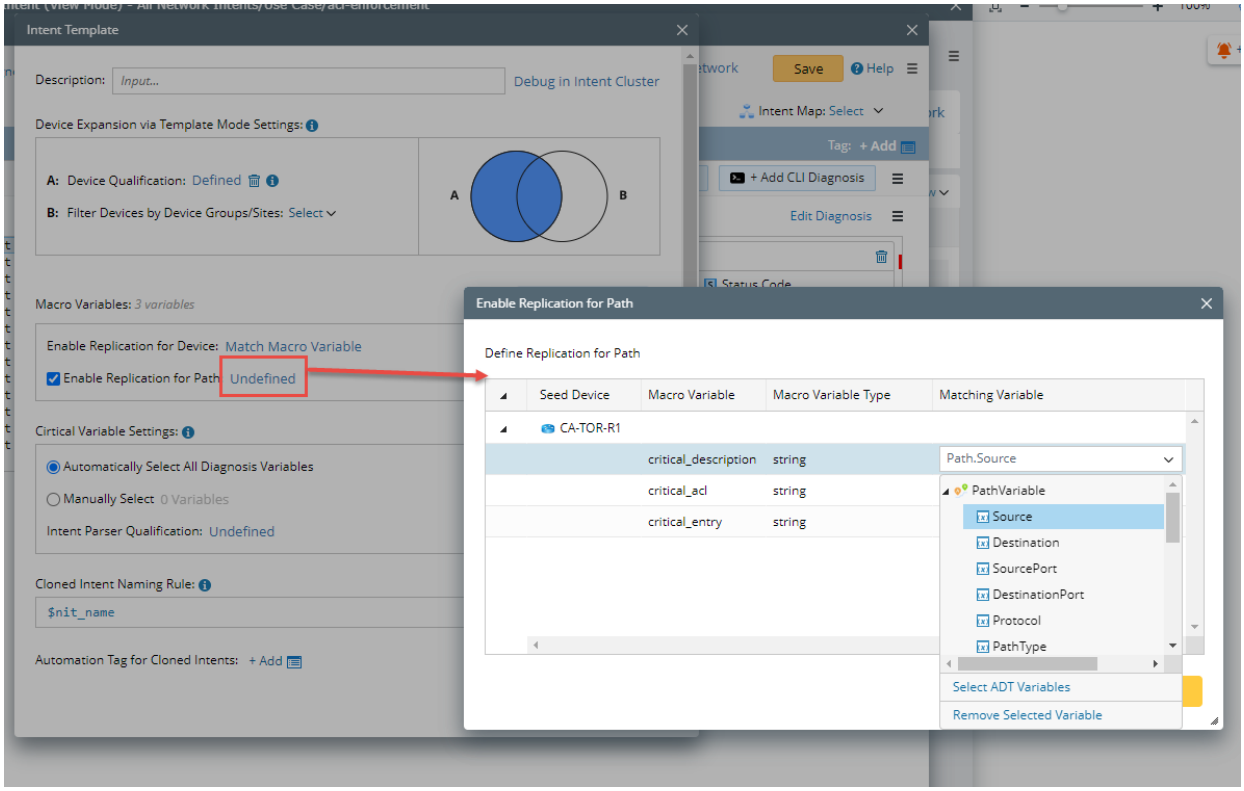
To reduce the possibly large number of Candidate Values to perform decoding operations, users can set a **decode flag** for the values.



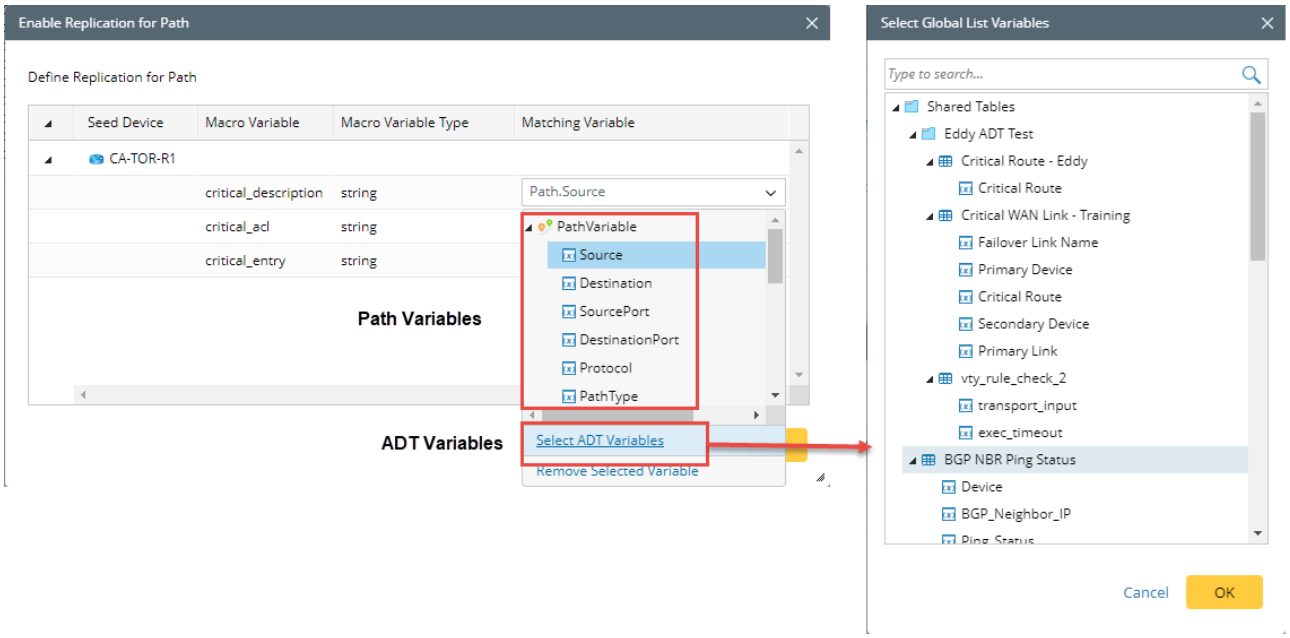
Users can define a filter logic for each Seed Device by selecting an ADT column. By using an ADT column with Boolean Type, it can decide whether the row will be used for decoding.

4.12.3 Intent Replication for Path

When an Intent Template is used for a Path, it can be used as input and passed to the Macro Variable for replication. Therefore, the ability to enable replication for Path is added here, which can replicate for Path Input.

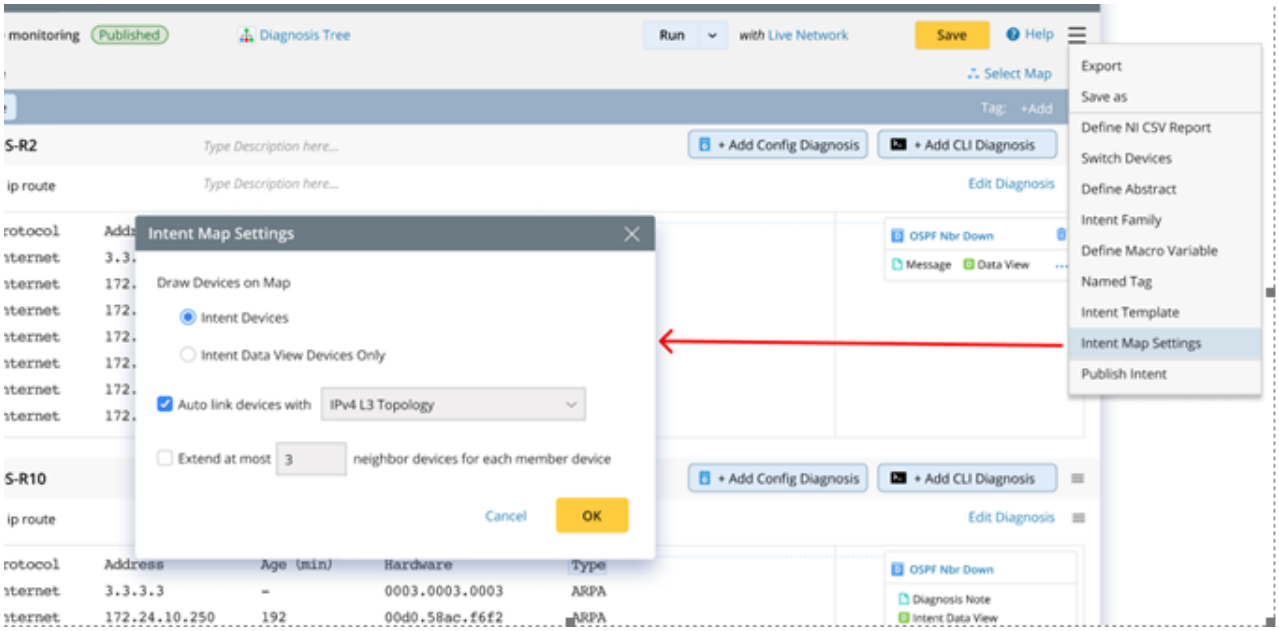


When the user enables the replication for Path, a window "**Enable Replication for Path"** popups for the user to select the Path Variable used for the Seed Device Macro Variable.



4.12.4 Intent Map Setting Configuration

The Intent Map configuration in the Intent Template is moved to the **Intent Map settings** inside the Intent.

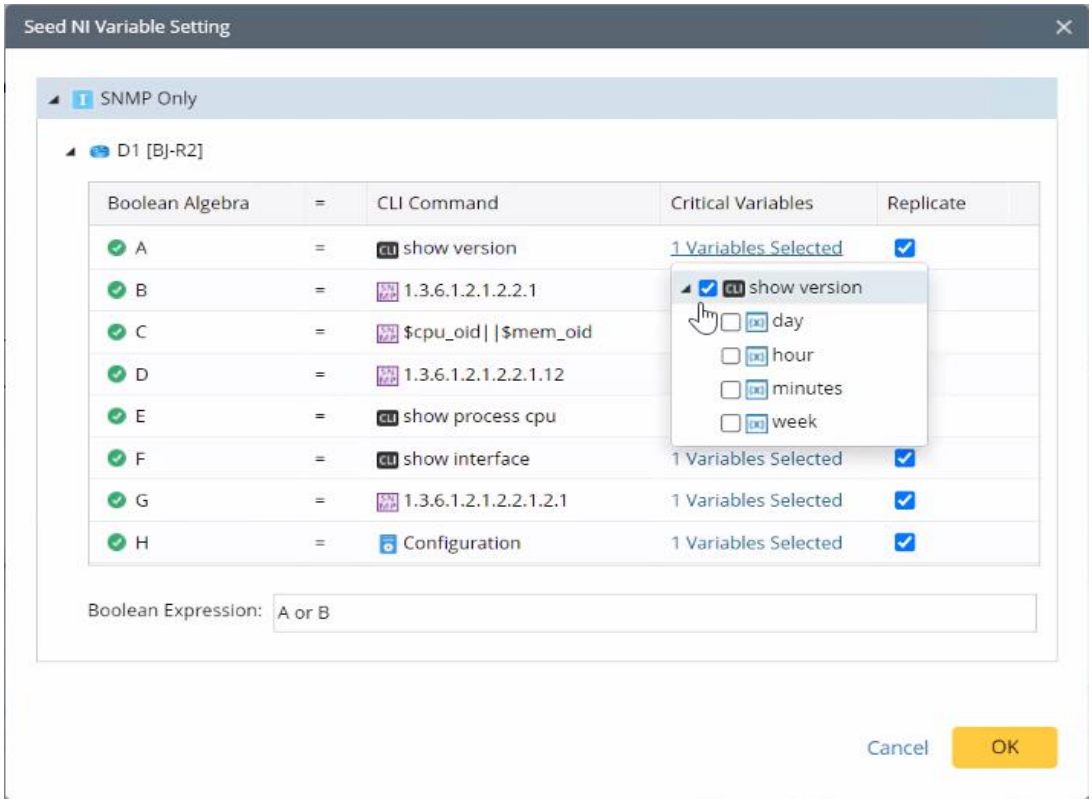


When generating Cloned Intent, the Cloned Intent Setting will be set according to the **Intent Map Settings** in the Seed Intent.

4.12.5 Configuration of Critical Variables

The configuration of Critical Variables is consistent with the configuration items of the improved Intent Cluster. The logic is consistent for generating Cloned Intent for PAF/TAF, and the match is determined based on the Boolean expression of all Critical Variables.

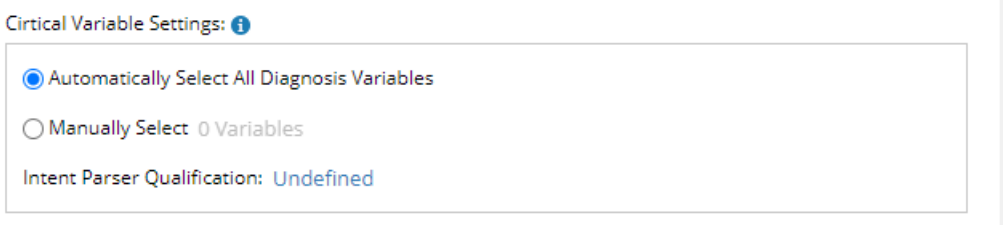
Users can select any variable as the Critical Variable and combine them with the Boolean operations (AND/OR).

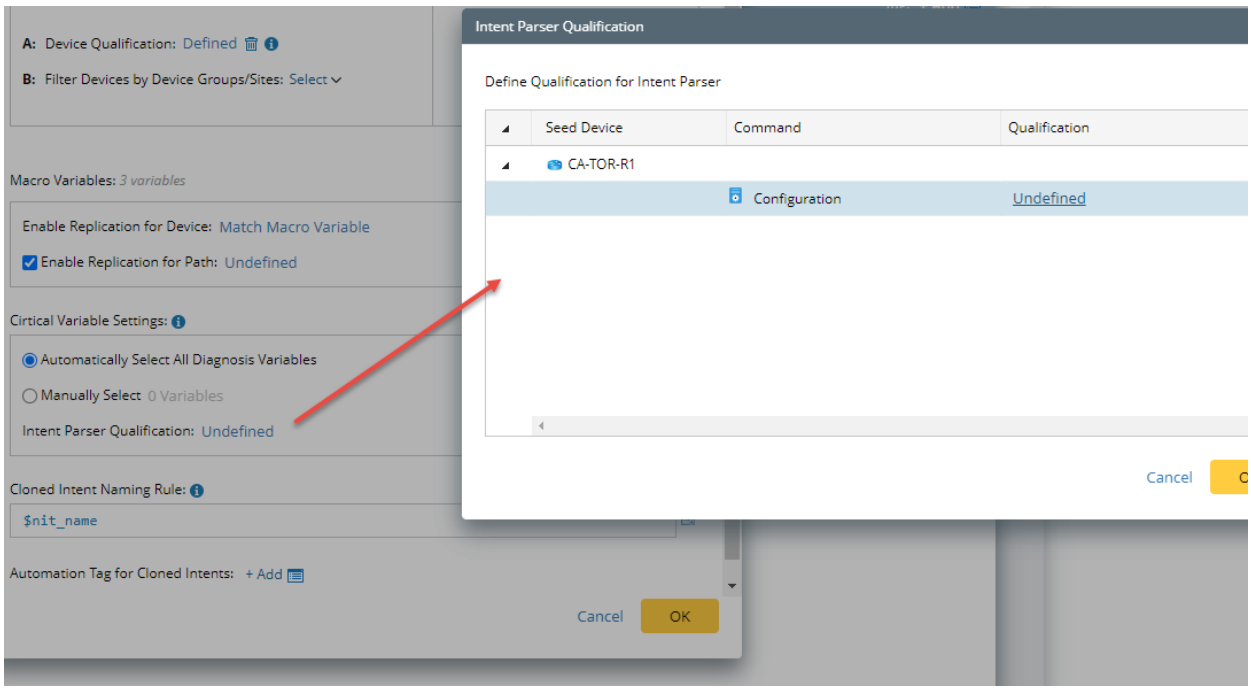


A Replicate checkbox is added to each device section, indicating whether the corresponding CLI/Config will be replicated. Dependencies across CLI Command will be passed through even if not selected for replication.

4.12.6 Intent Parser Qualification:

An option **Intent Parser Qualification** is added in the intent template settings to reduce the possibly large number of commands during the decoding process. With this option checked, the devices will be filtered according to the qualification defined in the **Intent Parser Qualification** (for example, per different Vendor Devices).





For each command and each Seed Device, users can define the following:

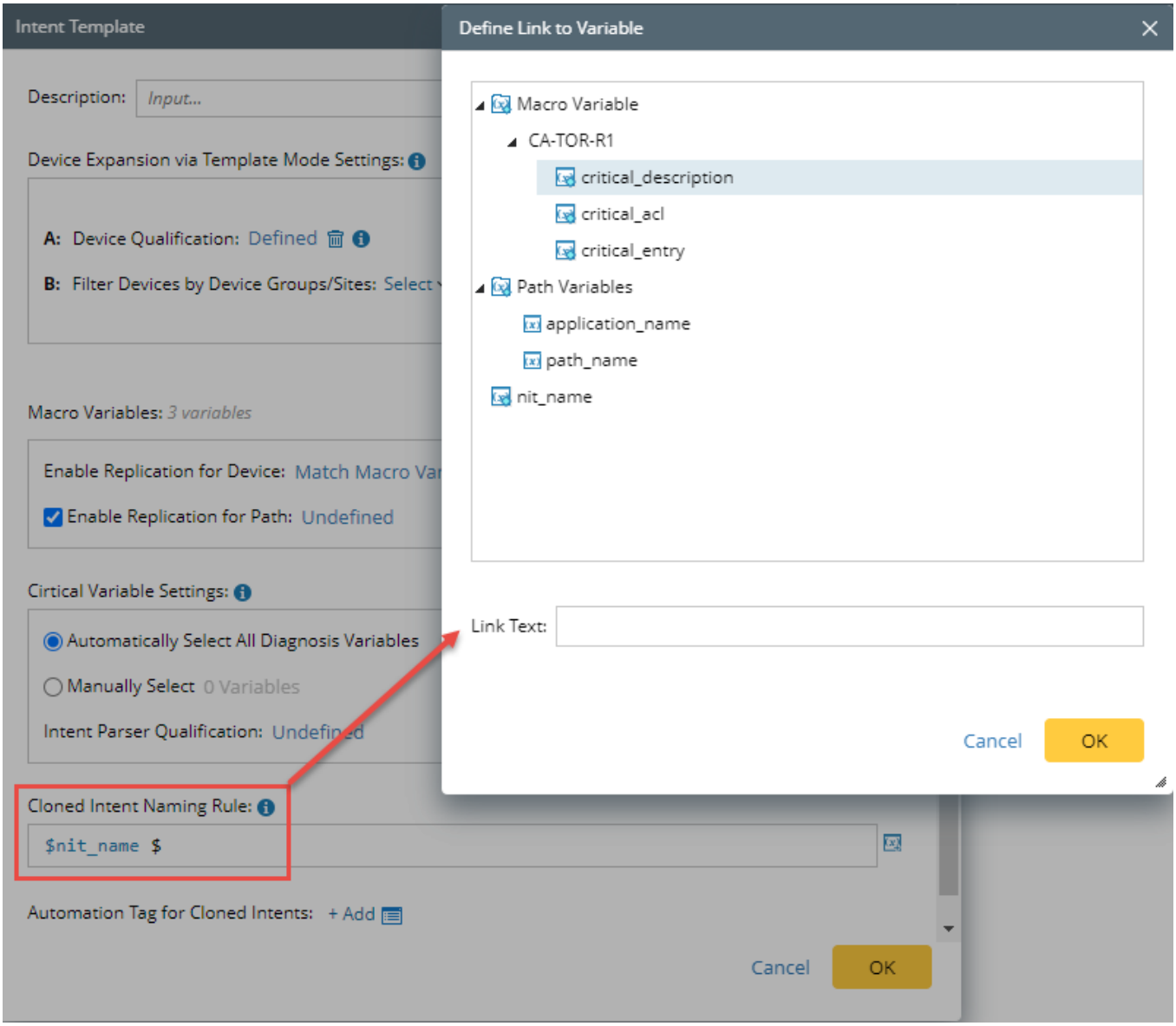
- **Device Type Qualification:** Specifies the Device Type Qualification that will be effective for Device Types.

4.12.7 Cloned Intent Name Rule

To make the name of Cloned Intent more meaningful, users can define the rule to name the cloned intent.

The following variables be inserted into the name:

- NIT_name
- Macro Variable Name
- Path variables (application_name and path_name)

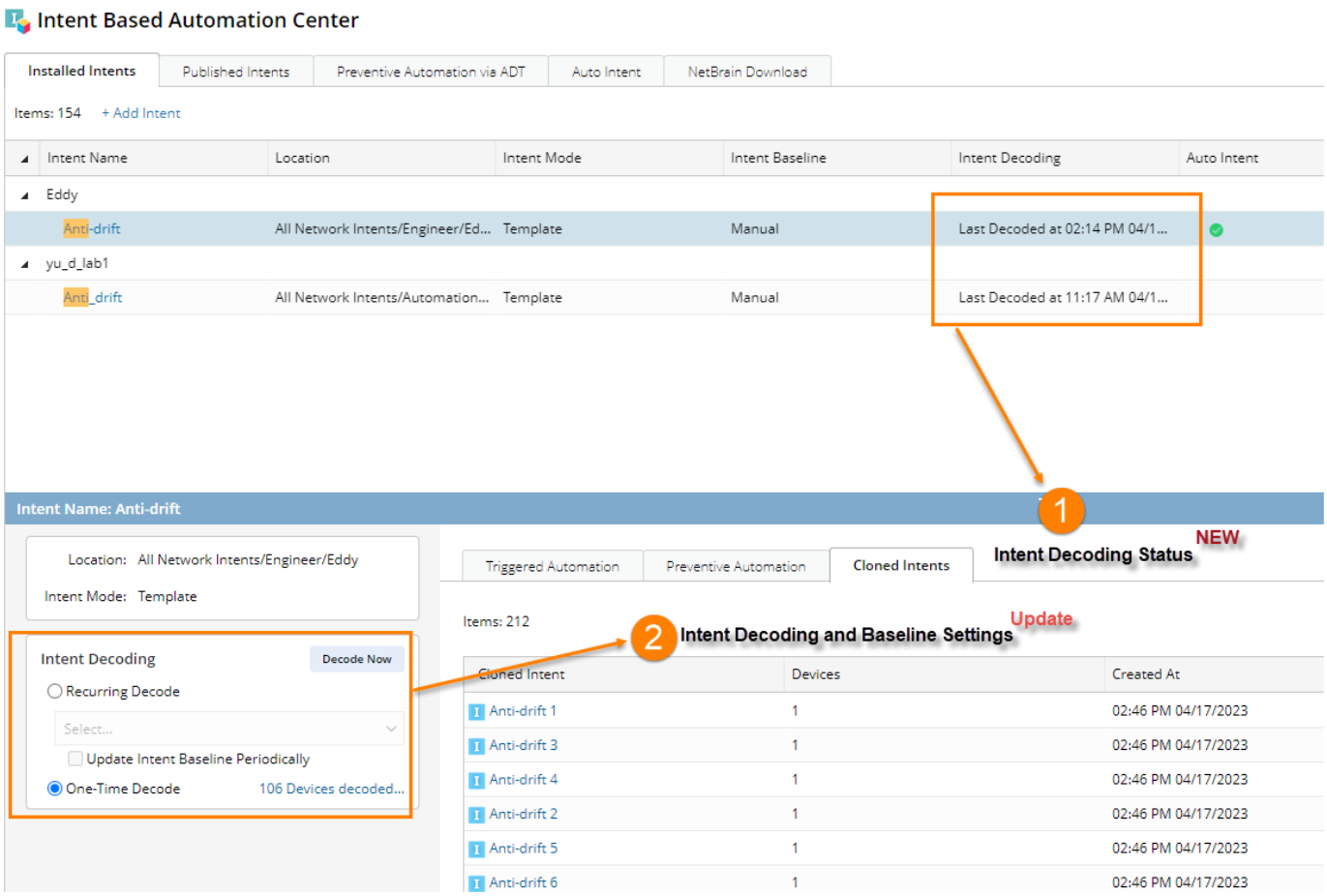


4.13 Intent Decode and Baseline Improvements

R11.1 improves Intent Decode and Baseline Service to remove the complexity in understanding the Intent Decode/Baseline and the Intent Decode cycle. The key improvements are:

1. Added detailed status indicators to the column **Intent Decoding**

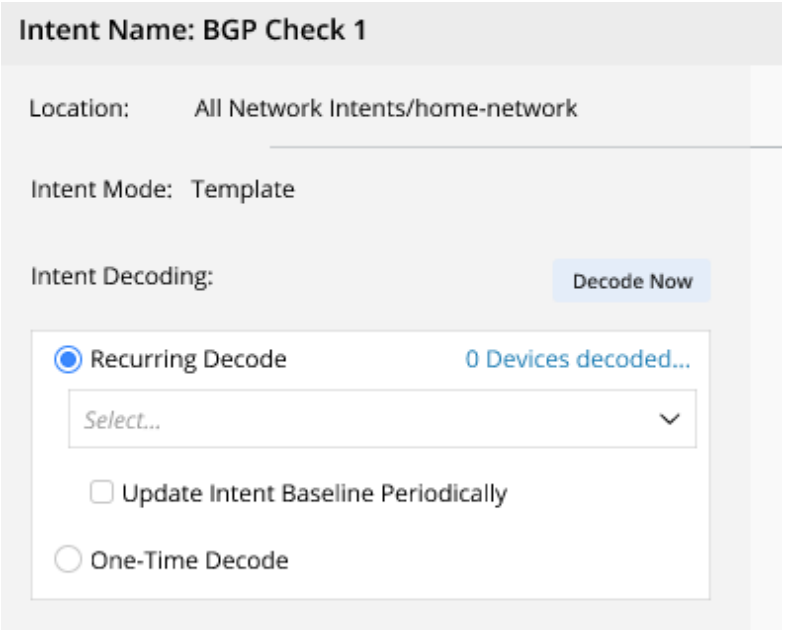
2. Updated Intent Decoding and Baseline Settings pane.



4.13.1 Intent Decode / Baseline Settings Adjustment

R11.1 version adds the option, **Update Intent Baseline Periodically**, under Recurring Decode.

After selecting **Recurring Decode**, users can select a timer and check **Update Intent Baseline Periodically**.



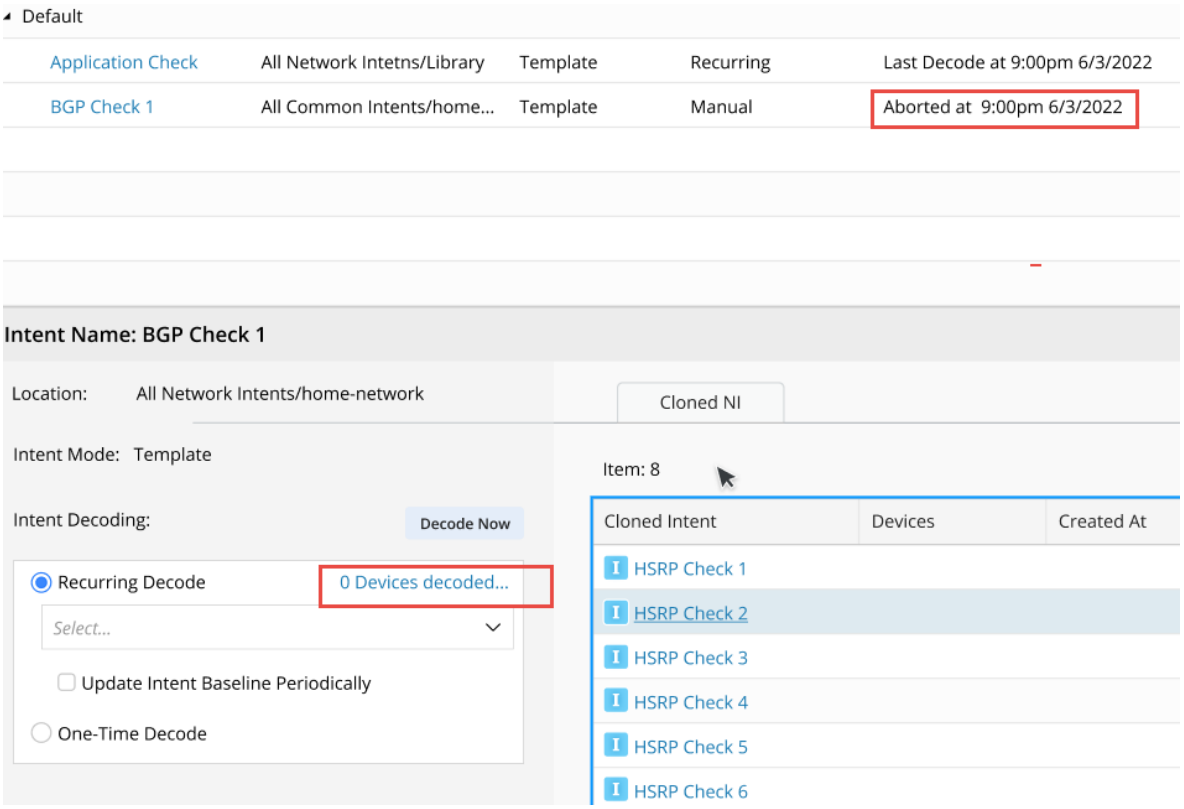
The logic is:

- The intent Baseline cycle is completely consistent with the Intent Decode cycle, and there is no longer a scenario of an inconsistent Intent Decode/Baseline cycle as in the previous version.
- Intent Decode is completely decoded according to the period selected here, and there is no longer a minimum daily period limit.

4.13.2 Intent Decode Displaying Status

More intent decoding statuses are added:

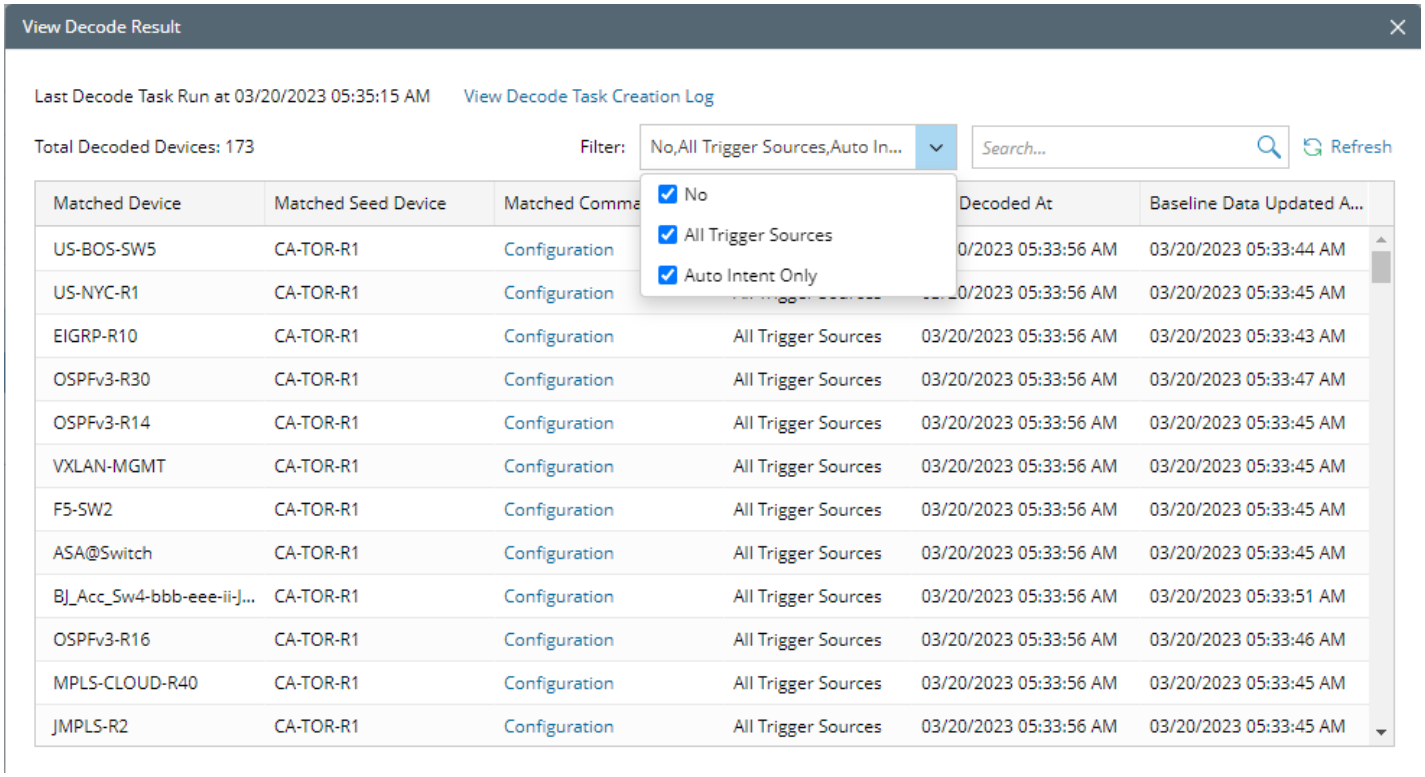
- **Aborted at xxx**: **No Qualified Devices** – A corresponding status label is added when the Intent Decode task ends due to no qualified devices. If the decoding result is 0 devices, "**0 Devices decoded...**" is displayed next to the **Recurring Decode** field.



- **Not Started:** The Decode Task has not started.
- **Decode Now task scheduled at xxx:** The task is scheduled in the future.
- **50/180 devices processed... 24 devices matched:** The Decode Task is going, showing the total number of devices (180), the number of devices that were already decoded (**50)**, and the number of matched devices (24).
- **Last decoded at 06:50:30 PM 12/2/2022:** The date and time that the task was done.

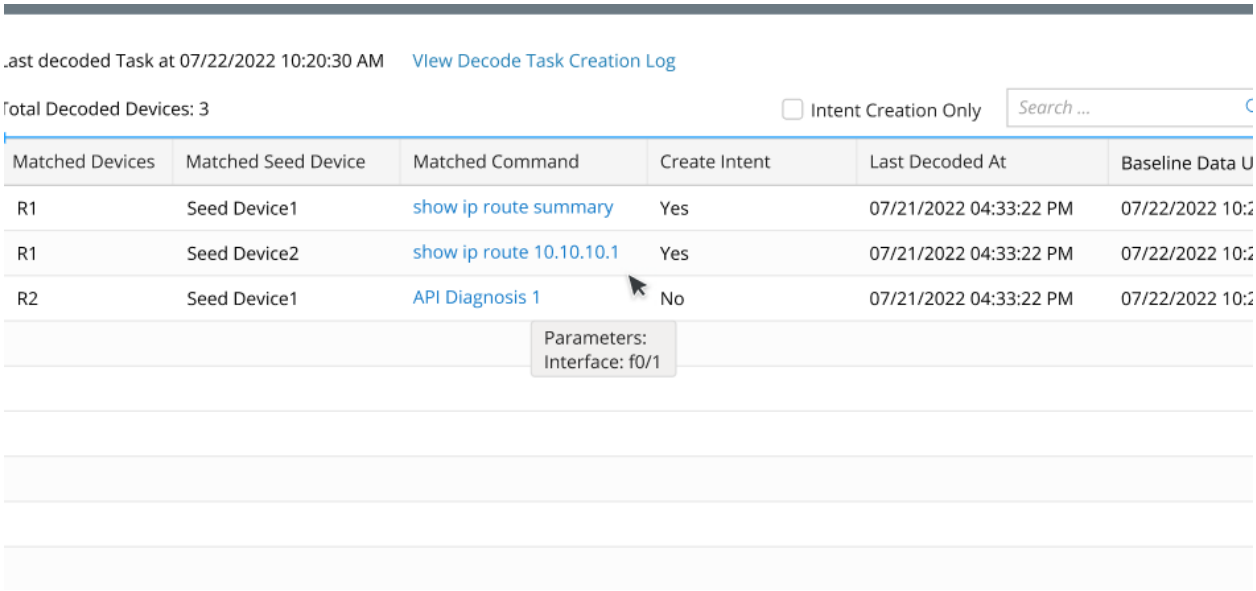
4.13.3 Intent Decode Results Improvements

If the Intent Decode process succeeds the command test but fails the Critical Variable test, the result will still be stored to be used by Auto Intent. This scenario will result in a situation where a Command Test is successful, but Create NI is **NO** (due to a failure in the Critical Variable result). And when the user hovers over the **Create Intent,** it will display a prompt "**Command tested successfully with default macro variable value of Seed NI**".

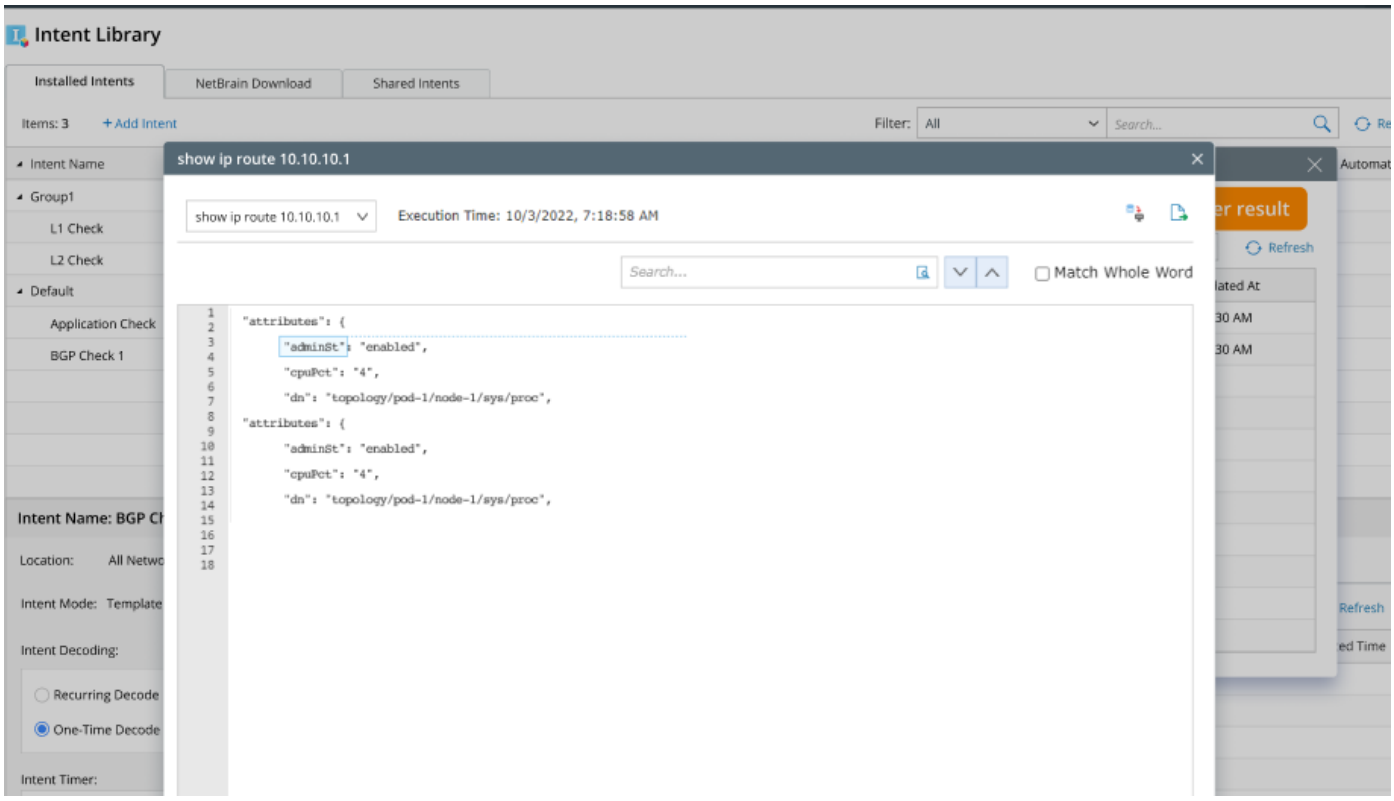


4.13.4 Show Decode Results for API Parser

When displaying the results of the API parser, the API names and the API parser may be the same but correspond to different parameters. If users hover the cursor over the matched command, the **Parameters** of the API parser are displayed.



When users click on the API Parser in the Matched Command column, it pops up a window with the Baseline data:



5 Automation Bot

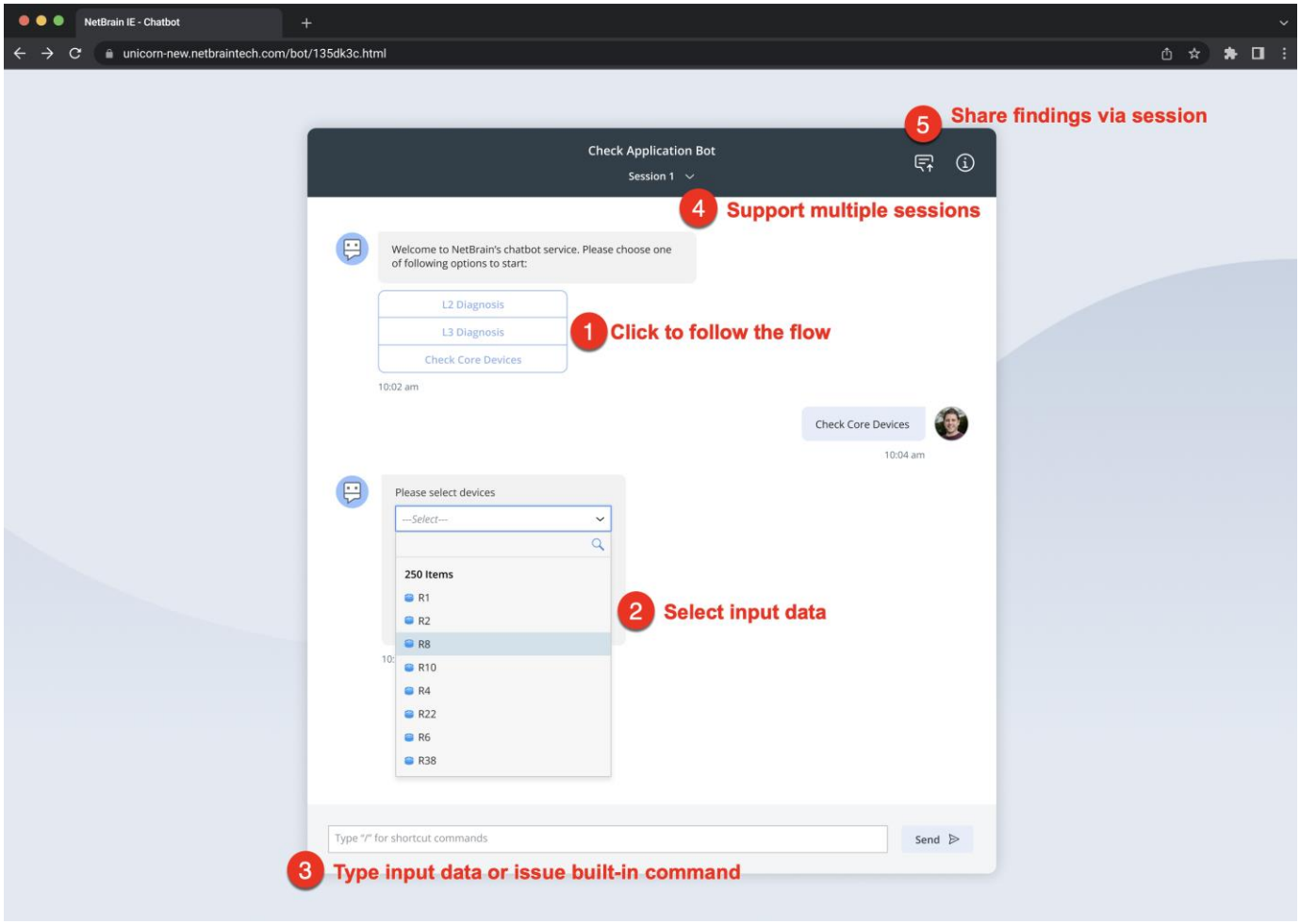
Automation Bot can be used to build an interactive, multi-step automation chatbot and execute multiple intent-based automations to solve real-world challenges without using NetBrain IE system UI. Using a chatbot doesn't require NetBrain know-how or any seat license and is a great way to provide self-service to a large audience.

Automation bot is considered a new event to launch automation in PDAS. The use cases of the Automation Bot:

- Use a Bot to deliver the dynamic map for specific IT objects, such as applications, data centers, sites, and device groups.
- Use a Bot to execute various diagnoses on a network by following a simple chat conversation.
- Use a Bot to prompt end users to interactively provide input devices and other related data to execute intent.
- Share important findings in the Bot with others for collaboration.

5.1 Use Chat Bot (End User)

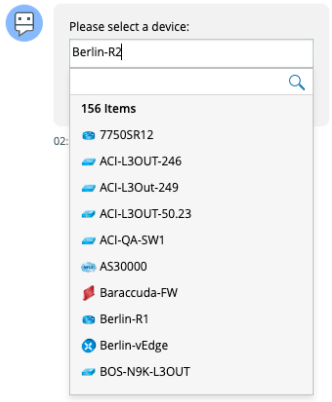
The end user uses a web-based chatbot to complete the self-service automations. The chatbot can be opened within the NetBrain IE system or a bot URL from others sharing from the bot editor.



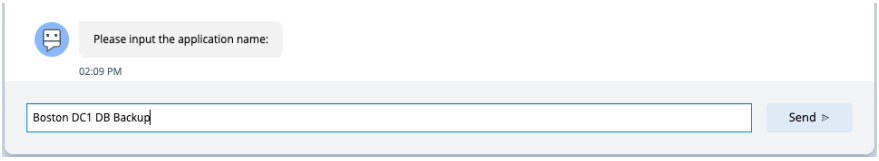
With a chatbot, users can:

1. Follow the conversation by clicking the button prompted by Bot to follow the conversation flow.
2. Interactively select or input value for automation execution: depending on the automation flow defined by the power user, end users might provide the following input to trigger the network intent for diagnosis:
    - Single selection and Multiple selections. For example, select one device or select multiple devices for NIT replication.

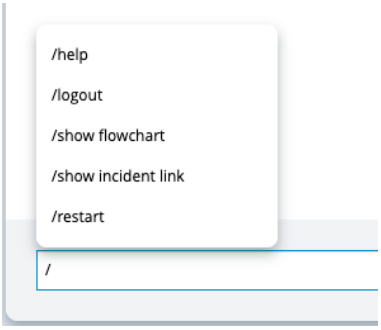
- Power users can allow end users to input string directly in the drop-down box instead of selecting an existing item from the list.



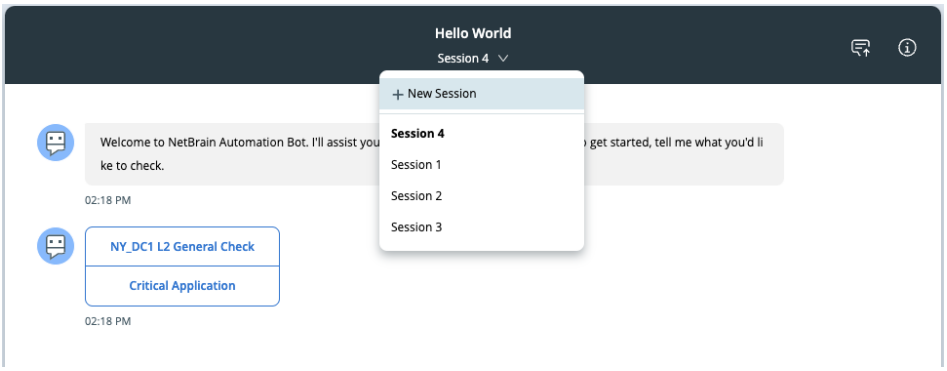
- Input text in the chat input box directly.



3. The chat input box at the bottom
    - Users can enter the text in the field instead of clicking the button.
    - Users can type"/" to show all built-in commands.



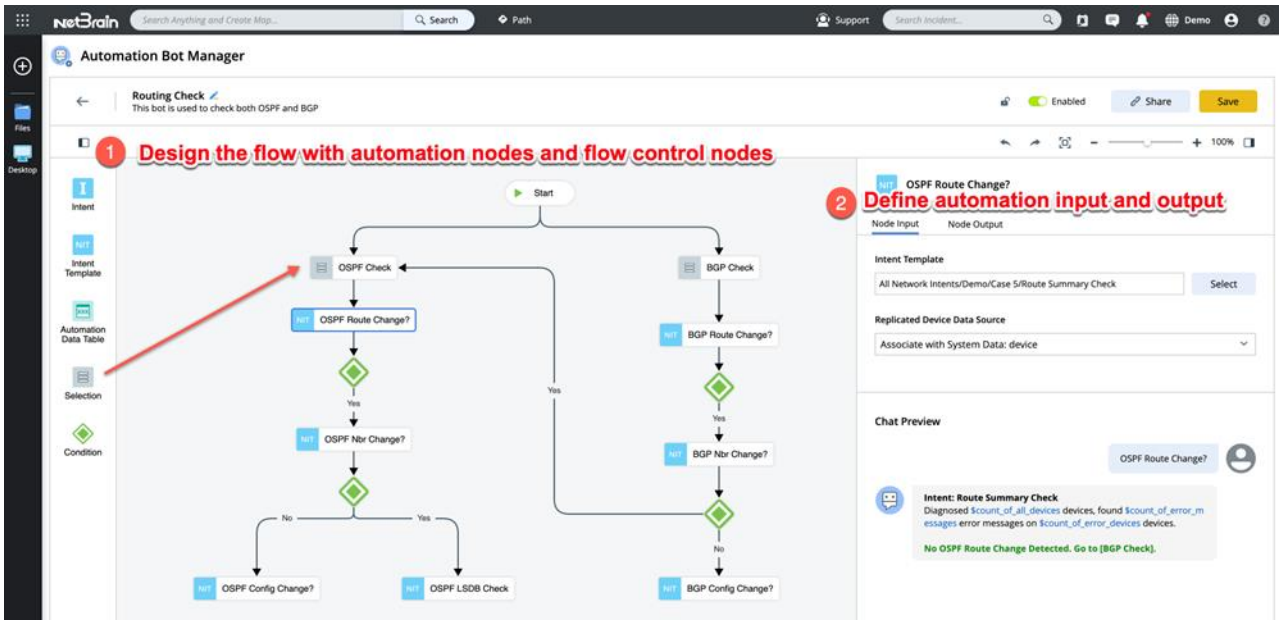
4. Sessions: Every chat session will be saved automatically. Users can view previous sessions and start a new session at any time.



5. Publish the current session with important findings to others for collaboration.

5.2 Chatbot Creation Flow

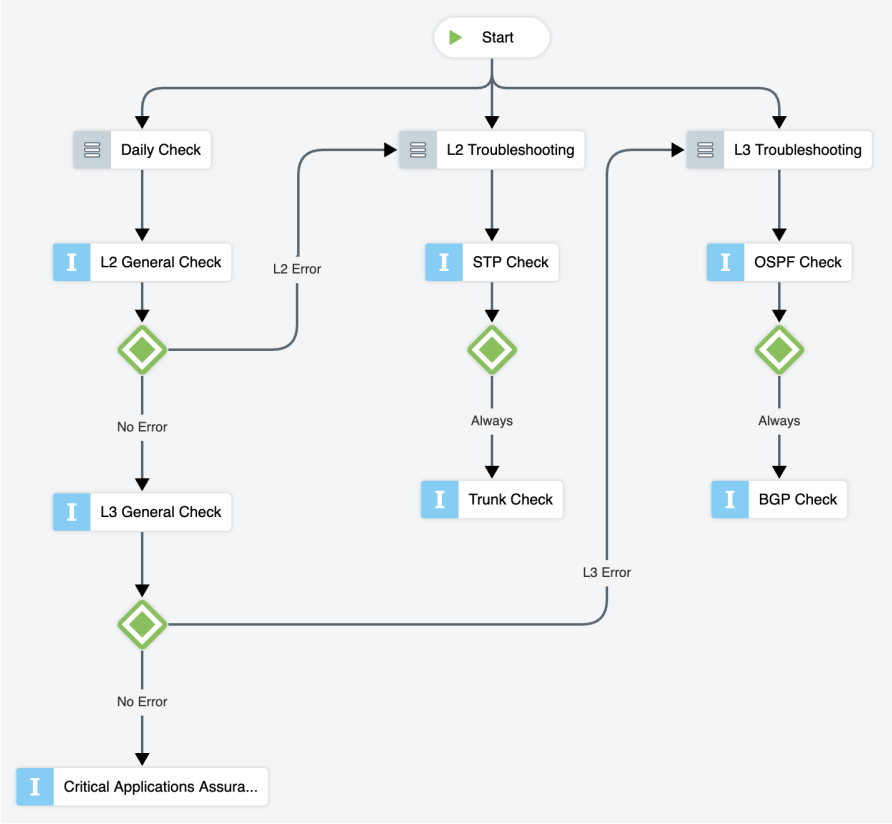
Power users can use Automation Bot Manager to build the bot conversation flow via just drag-and-dropping a node to the flow. The system provides the following types of nodes:



5.2.1 Intent (NI) Node

If power users do not want end users to interactively change devices for the intent during using this Bot, use the intent node.

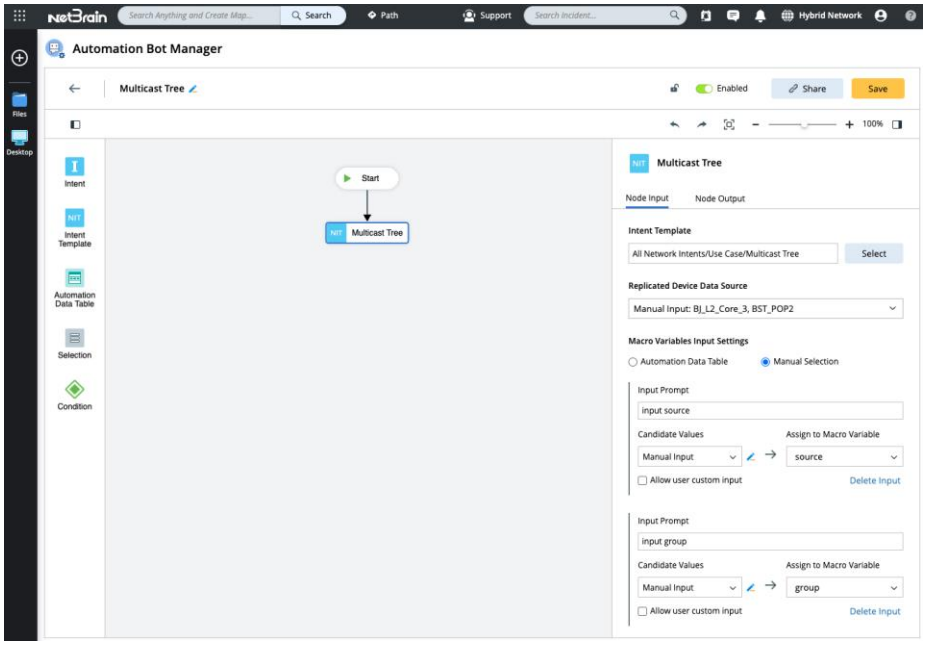
For example, power users create a series of intents for ***NY_DC1*** network daily checks and troubleshooting. The best way to enable end users to leverage this automation knowledge in their daily work is for power users to group these intents into a bot. Power users don't need end users to change the target devices defined in the intent. End users just need to chat with this Bot without accessing the NetBrain IE system and execute the pre-defined automation to ensure everything works as expected.



5.2.2 Intent Template (NIT) Node

When power users want end users to interactively change devices and other input data for the macro variables to replicate the intent for execution, use the Intent Template node. The Intent devices and macro variable value can be replaced by other devices and data for replication. During end user interactions with the Bot, it will guide the end user to interactively provide input devices and macro variable value for intent template replication.

For example, create a bot to guide end users to draw the multicasting tree of the source/group that end users interactively specify and execute multicasting diagnosis.

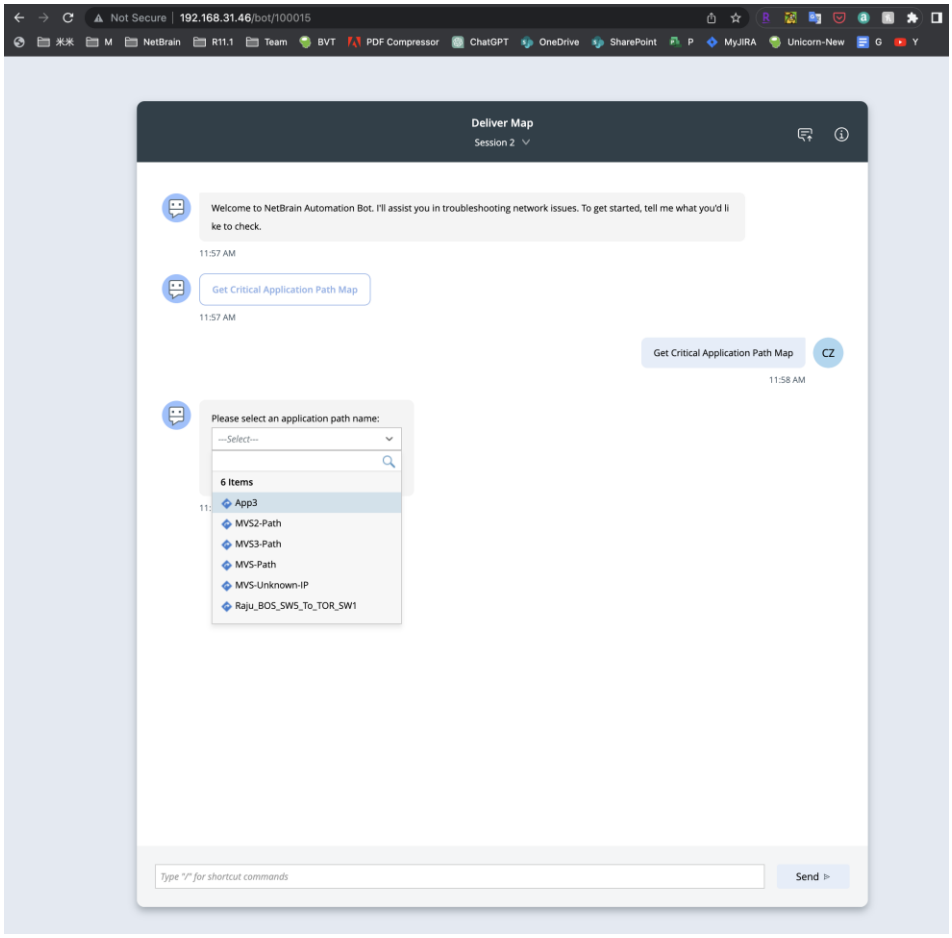


5.2.3 Automation Data Table (ADT) Node

ADT provides a flexible way to organize and extend the critical network assets and their associated intent and map. Two typical use cases can be achieved by ADT node:

- Use the ADT node to deliver a map.
- Use the ADT node to dynamically search matched critical assets and execute its associated intent automation.

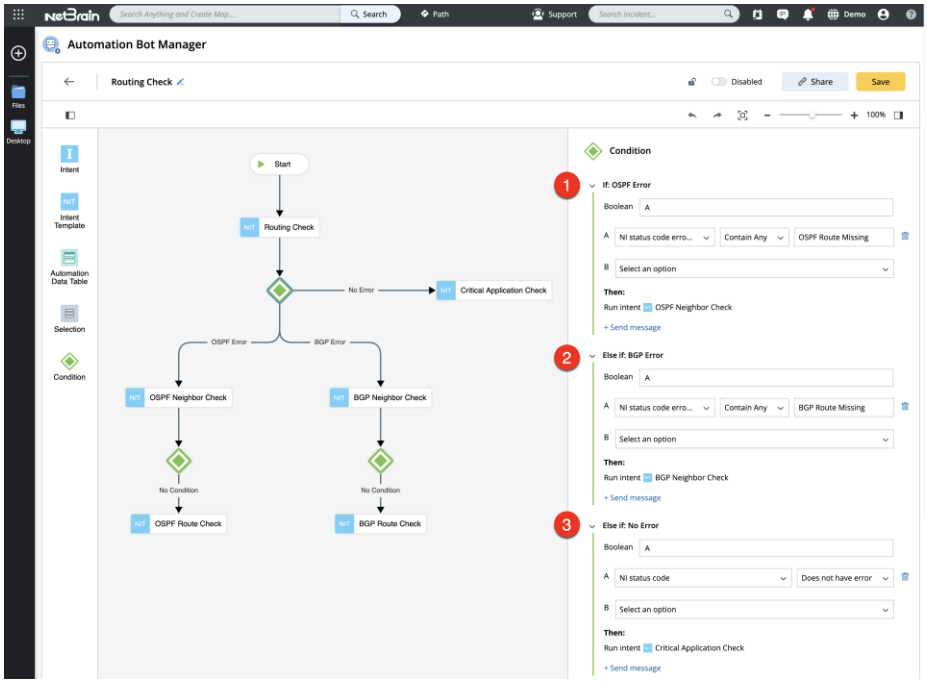
For example, create a bot to deliver a critical application path map to bot end users.



5.2.4 Condition Node to Control Flow

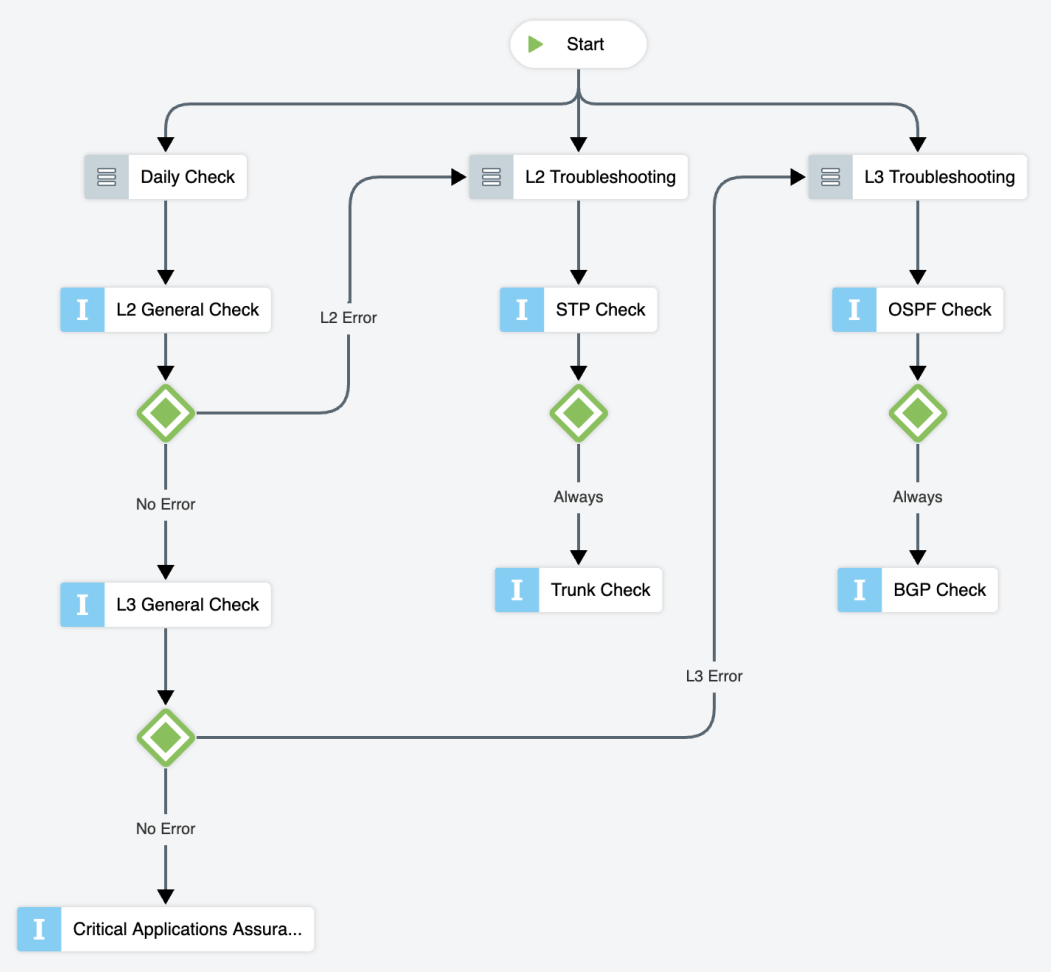
A completed conversation flow usually consists of multiple steps. Power users can connect multiple automation steps with conditions to decide the next flow to send users in an automation flow based on the previous diagnosis result. For example, if a previous diagnosis detects an error, go to flow1; if there is no error, go to flow2. No condition is also available to connect automations without any condition, and the system will execute multiple intents individually.

For example, use condition node to control: if OSPF errors are detected after a Routing Check, then go to OSPF detailed check branch; if BGP errors are detected, then go to BGP detailed check branch; if no error is detected, then go to critical application check.



5.2.5 Selection Node

Power users can leverage the selection node to organize bot conversation flow by grouping automations. Just double-click the node to rename it. No additional settings are needed.

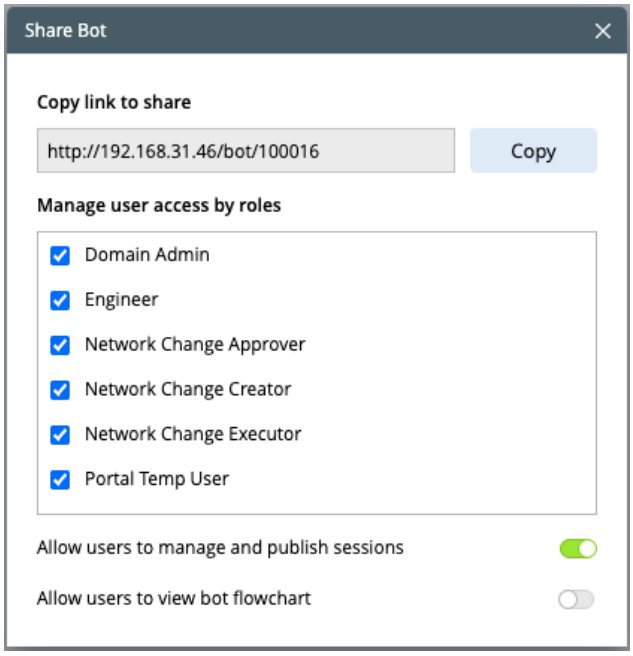


5.3 Additional Features

The system provides the following functions to help users manage the bots:

1. Lock/Unlock a bot with/without a password to avoid unauthorized modification.
2. Enable/Disable a bot.

3. Share the Bot and control which user roles can access this Bot, and whether to allow the session feature and show flowchart command in the end user chat console interface.



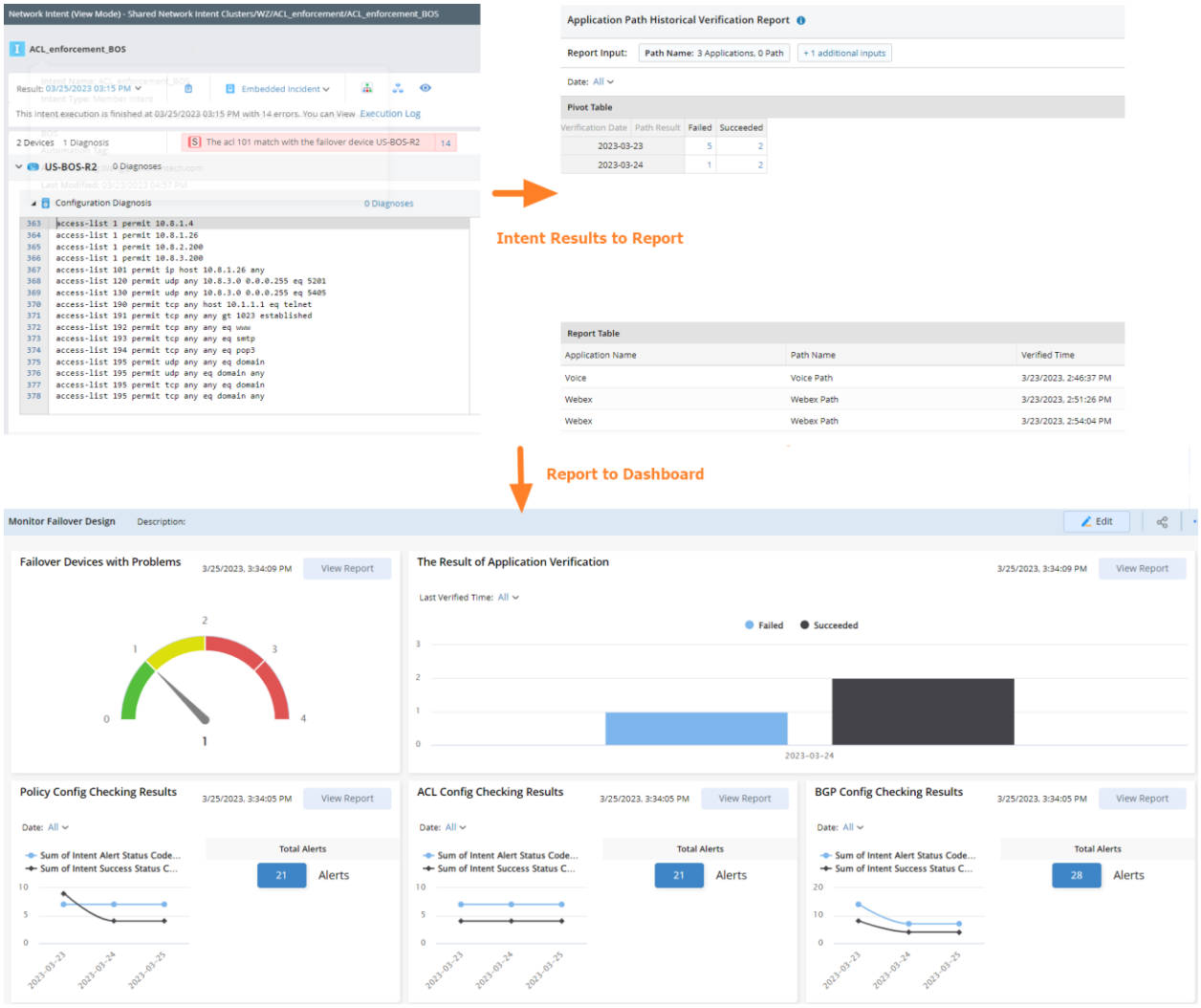
4. Customize Bot Login Page Logo, Content, and About Bot Content for different purposes.

6 Report and Dashboard

NetBrain system lacks a function to view the results across the whole network in the early versions. R11.1 adds the Report and Dashboard with two capabilities:

- Organize, analyze, and share data from different NetBrain automation assets, such as thousands of ADT Intent results.
- Provide a visual display of multiple automation results from PDAs on a single GUI.

The picture below illustrates an example that uses Report and Dashboard to analyze Intent results and visually display the data to help monitor a network's failover designs.

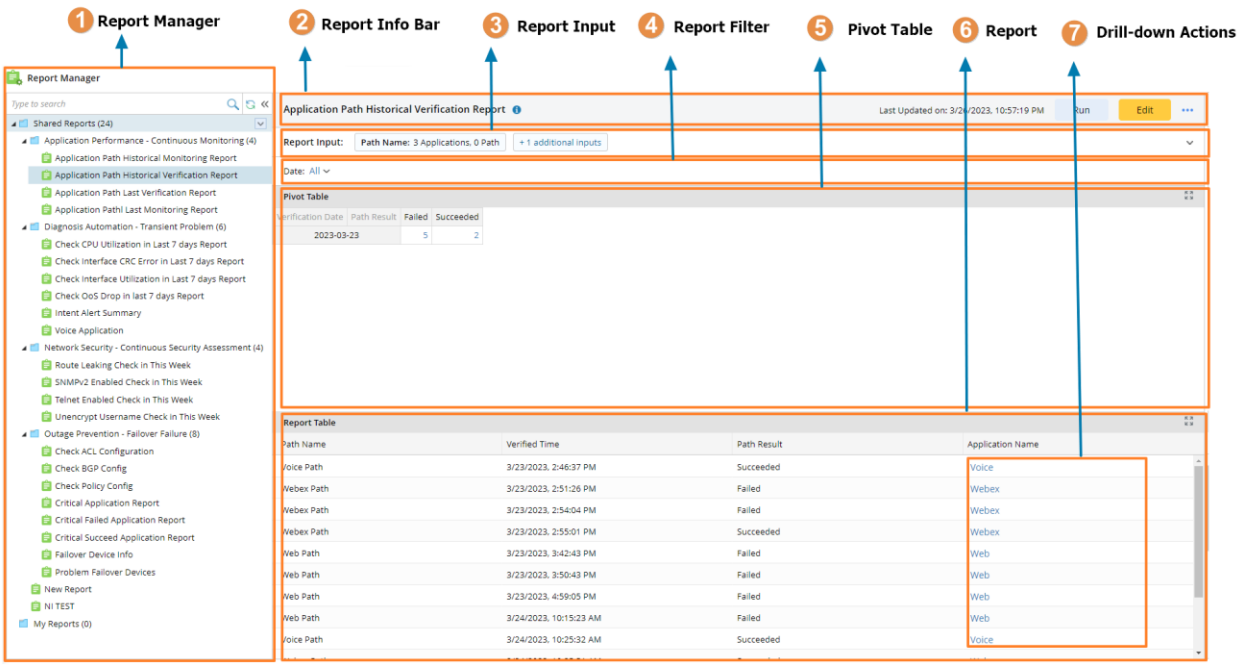


Besides the example mentioned above, other common use cases for the Report and Dashboard:

- Capture the Configuration Drifts for Outage Prevention
- Capture transient problems for Diagnosis Automation
- Perform Security Assessment for Network Security
- Display the result of Change Management Tasks for Protective Change
- Display the application information for Application Performance

6.1 Key Components to View Report and Dashboard

6.1.1 Key Components to View Report

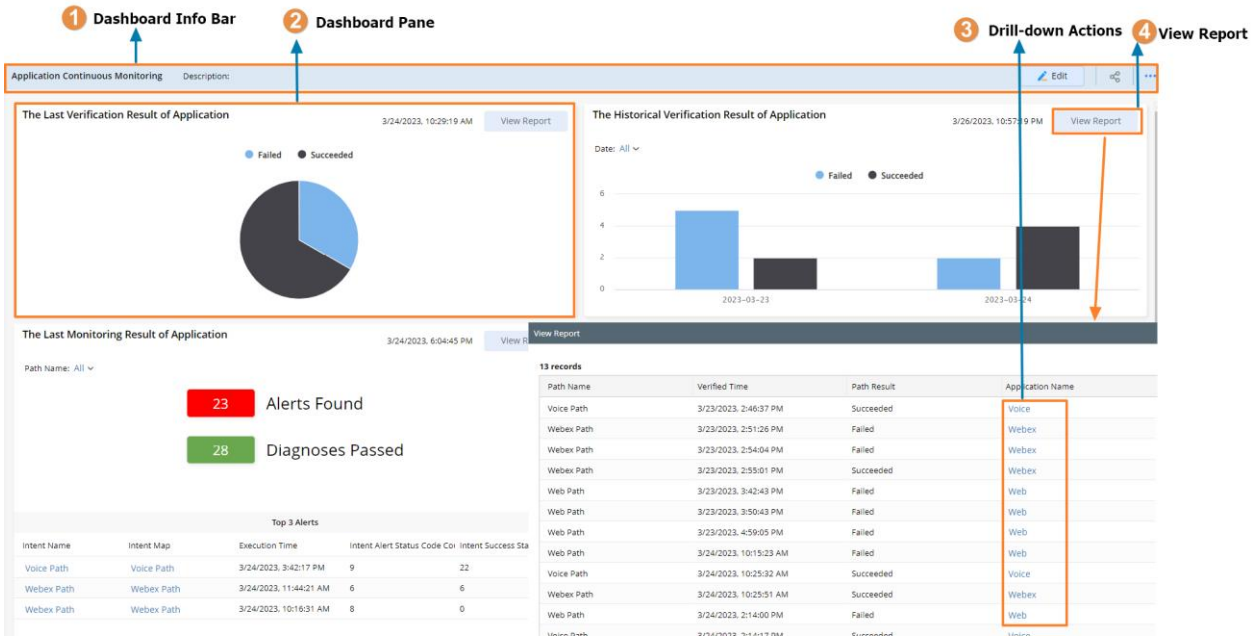


The GUI of the View Report page consists of seven key components:

1. **Report Manager**: Lists all the accessible reports in the domain. From the Report Manager, users can choose a report for viewing, editing, or renaming; or create new folders and reports.
2. **Report Info Bar**: Display the basic information of the report, such as the title and last update time. You can mouse over the info icon to view more information, including the report's creator, last modified by and last modified time. It also provides the option to edit or run the Report.
3. **Report Input**: Display the Report Input and the criteria to select and generate the report data.
4. **Report Filter**: Filters used to refine the report data or narrow it down to subsets of report data.
5. **Pivot Table**: Display the Pivot Table, which helps analyze and summarize information from a large amount of Report data.
6. **Report:** Display the content of the report.

7. **Drill-down Actions:** Hyperlinks in report columns that are linked to other NetBrain features for users to drill down to underlying data.

6.1.2 Key Components to View Dashboard

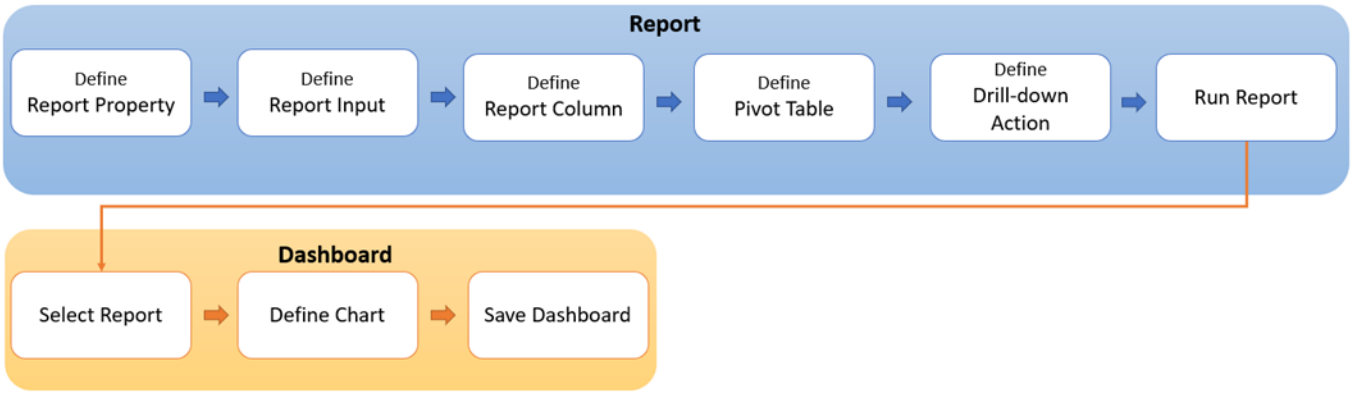


The graphic user interface of the View Dashboard page consists of four key components:

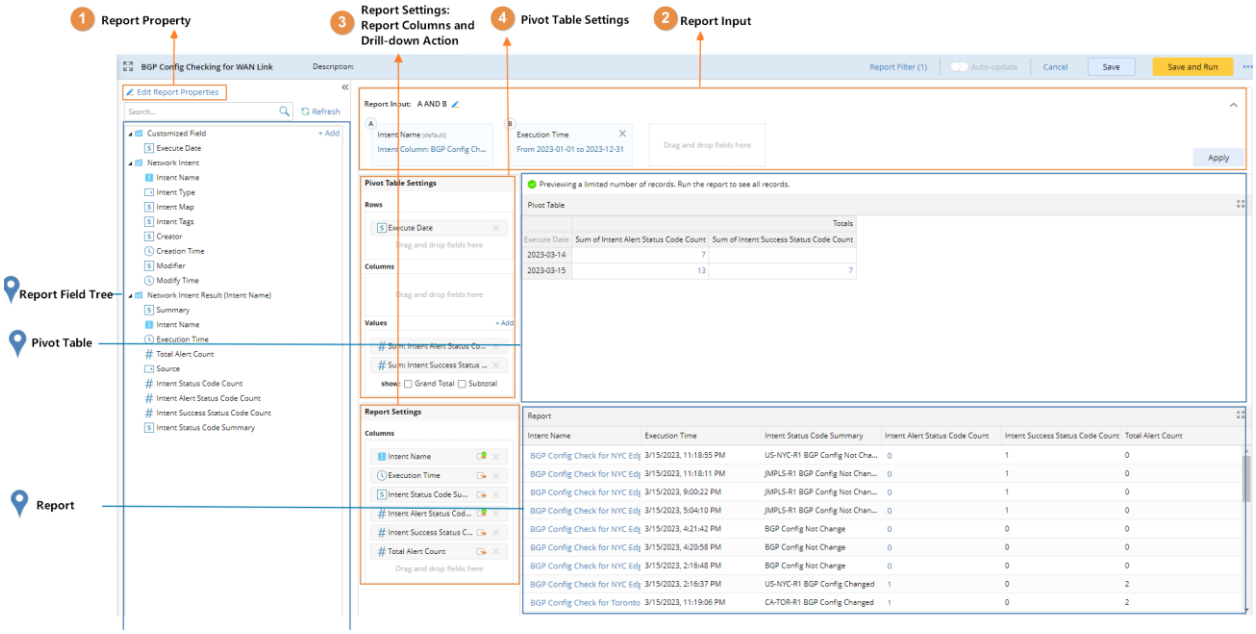
1. **Dashboard Info Bar**: Displays the basic information of the Dashboard, such as title and description. It also provides the option to edit, share or update the Dashboard.
2. **Dashboard Pane:** Major components of the Dashboard that display report data in various charts.
3. **Drill-down Actions:** Hyperlinks in report columns linked to other NetBrain features.
4. **View Report**: Display the report data used for the Dashboard Pane in a pop-out window.

6.2 Key Components to Create the Report and Dashboard

The user flow to create the Report and Dashboard:



6.2.1 Key Components to Create a Report



The GUI of the Report editor consists of 3 major areas and 4 basic settings. You can define a basic report by defining the following four settings:

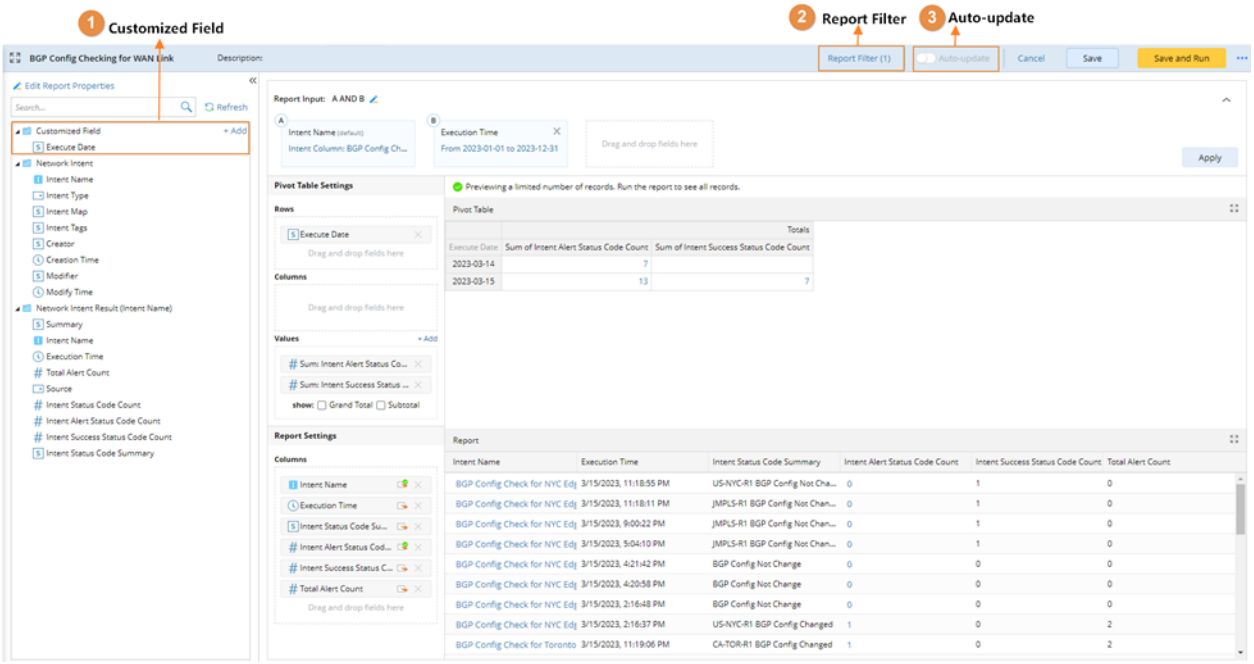
1. **Report Property:** Define the basic properties of a report, including report name, creation method and Report Field Tree. Report Objects and Report Fields define the Report Field Tree.
    - **Report Object**: Report Objects are analogous to database tables that organize NetBrain data by categories, such as Device Property, Change Management, and Intent Property.

    - **Report Field:** Report Fields are analogous to columns in database tables. Each Report Object has multiple Report Fields. For example, the Intent Property Object has Report Fields such as Intent name, creation time, Intent type, etc.
2. **Report Input:** Define the Report rows with filtering criteria. Each report has one built-in default input that can be edited but not deleted; additional report inputs can be added by drag-and-dropping Report Fields to the Report Input area. The logical relationships between each input can also be defined.
3. **Report Settings:** Define the Report Columns and Drill-down Actions.
    - **Report Columns**: Define the Report Columns by drag-and-drop available Report Fields from Report Field Tree to the Report Column area.
    - **Drill-down Actions**: Define user interactions with the Report, such as opening the Device detail pane or the Network Change Manager. Those interactions will appear as hyperlinks in the Report columns for viewers to click.
4. **Pivot Table Settings:** Define the Rows, Columns and Values of the Pivot Table.

These four settings help to define the three major areas on the Report Editor page:

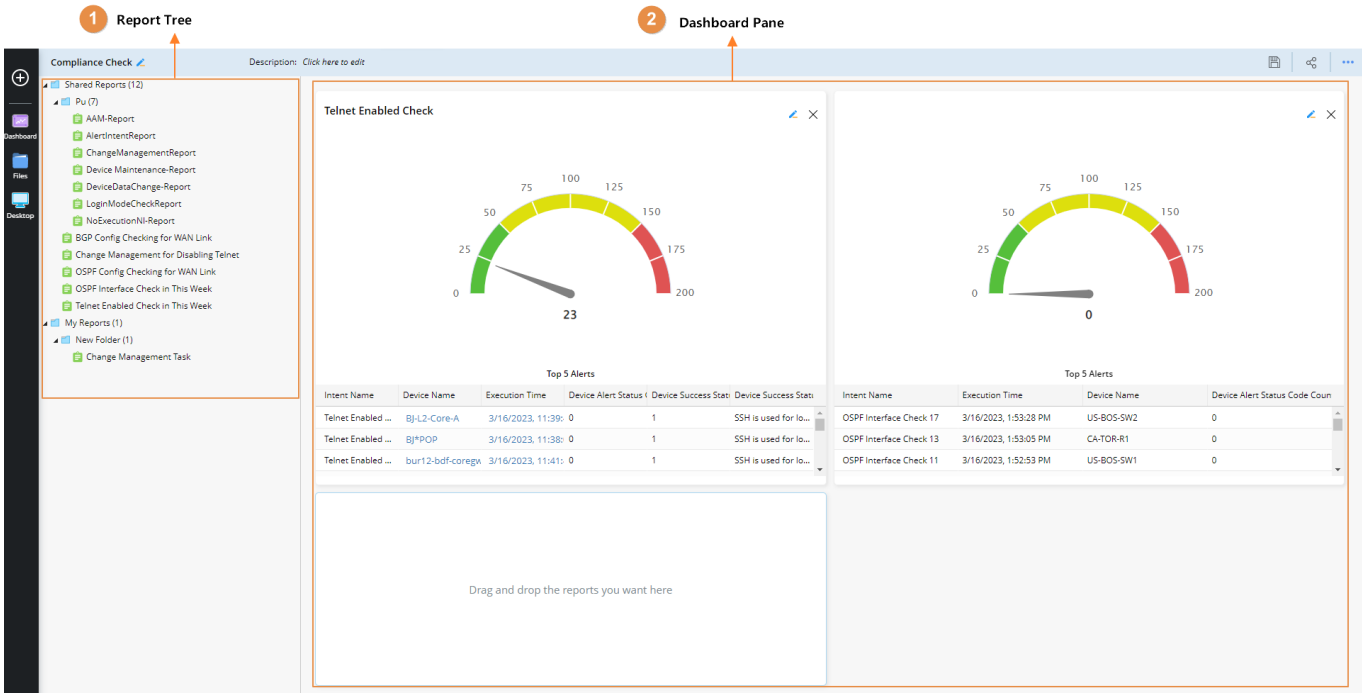
- **Report Field Tree**: Report Field Tree lists all the available fields that can be used in the Report, including Customized Fields defined by users and built-in Report Fields defined by Report Property.
- **Pivot Table**: Pivot Table helps aggregate the report data by selecting Report Fields to help analyze and summarize information from the Report. It can also be used in Dashboard to create charts.
- **Report:** The report area displays limited sample data of the report. The Report rows are defined by Report Input, and the Report columns are defined by Report Columns.

Besides the four basic settings mentioned above, the following advanced settings are offered as an extension to the basic Report functionality:



- **Customized Field:** Define customized fields besides built-in Report Fields using calculations, functions, and conditions.
- **Report Filter:** Define filters for users to drill down data when viewing a Report or Dashboard.
- **Auto-update:** Besides running Report manually, users can update the Report data automatically on the specified frequency.

6.2.2 Key Components to Create a Dashboard



The GUI of the Report editor consists of 2 major areas: Report Tree and Dashboard Pane.

- **Report Tree**: Lists all the available reports that can be used in the Dashboard. Drag and drop a report to the pane area on the right to add a Dashboard pane.
- **Dashboard Pane**: A Dashboard pane is defined by the data source and chart settings. Each pane can have a different report from Report Tree as its data source. Also, different chart types are provided to satisfy different use case scenarios, such as Line Charts, Gauge charts, Pie charts, etc. Besides the chart, each pane could have an optional call-out chart to help provide extra information.

6.3 Create Report

The report organizes data from different automation assets into reports, such as device, intent, application, change management, and site. Besides that, the Pivot Table feature in Report can help further analyze and summarize the report data.

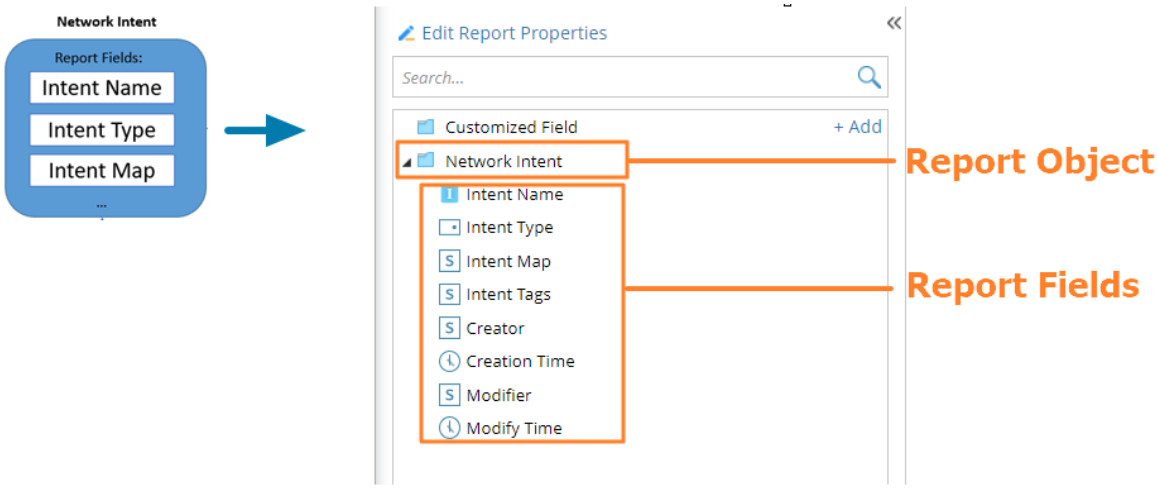
6.3.1 Define Report Properties

When creating a new Report from Report Manager, the Report Properties window will automatically pop out, then you can follow the steps on GUI to define the Report Properties. There are three steps to define the Report Properties:

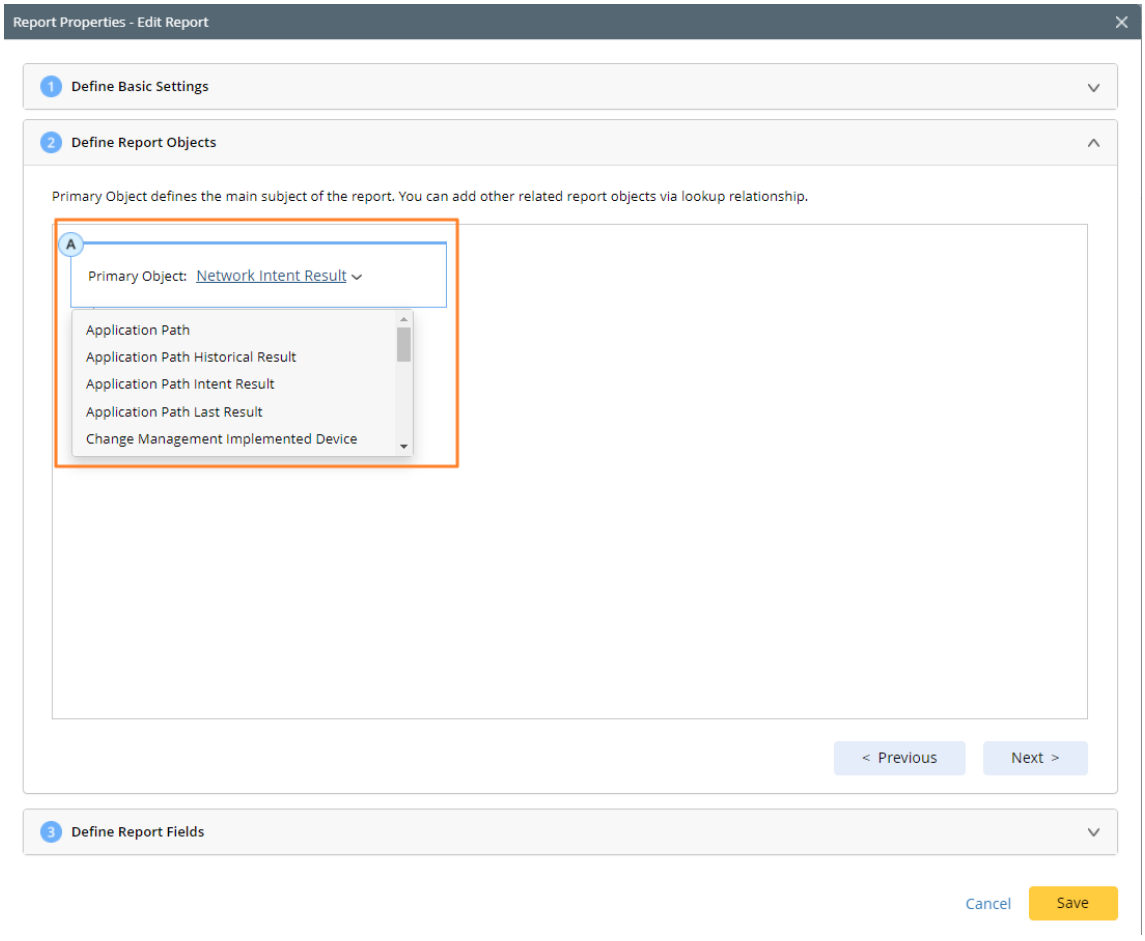
- **Basic Settings**: The name and description of the Report.
- **Report Properties**: Select the Primary Object, Related Object and their relationships.
    - **Report Object**: Report Objects are analogous to database tables that organize NetBrain data by categories, such as Device Property, Change Management, and Intent Property. Each report must select one Primary Report Object and can also have up to three Related Report Objects.
    - **Primary Object**: Primary Object defines the main subject of the report. Each report must have and can only have one Primary Object.
    - **Related Object**: Related Objects can supplement the main subject's information. Other Report Objects can be added to Report as Related Objects via a lookup relationship.
- **Report Fields**: Preview the Report Fields of the Report and add Related Fields if necessary.

6.3.1.1 Report Object

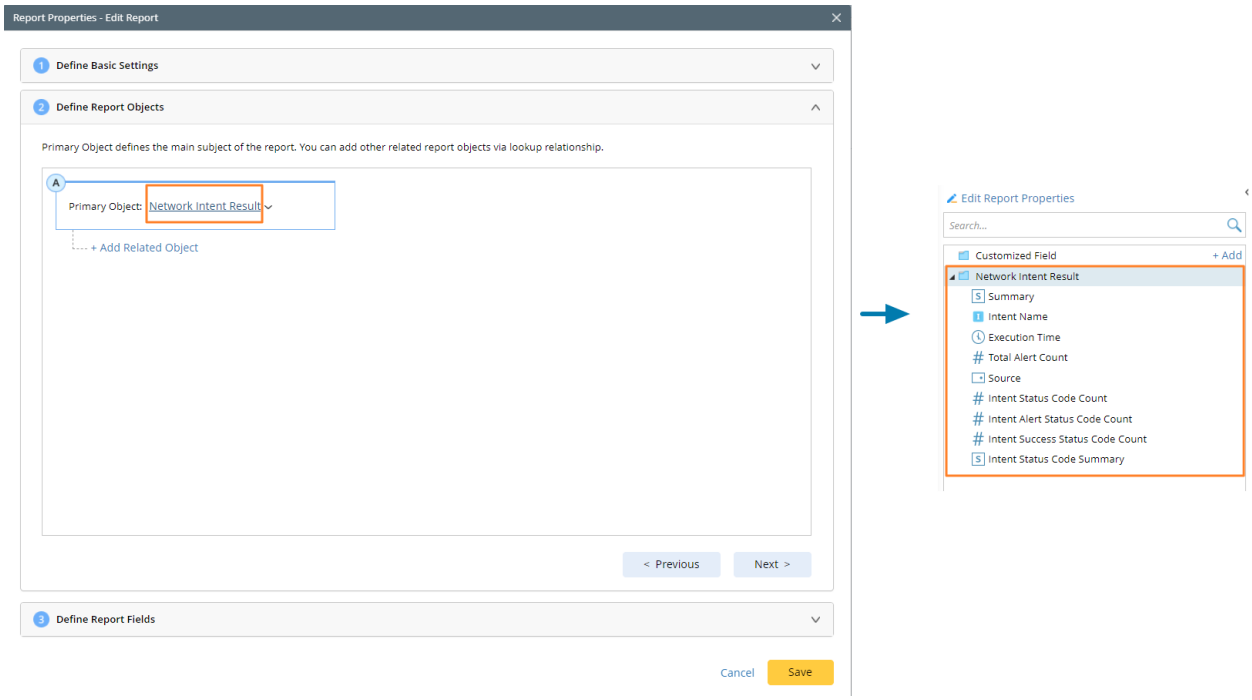
Report Objects are analogous to database tables that organize NetBrain data by categories. The Report Object includes multiple Report Fields that can be used as Report Columns to build report content. In R11.1, the system provides Report Objects for device, intent, application, change management and site. We will continue to provide more report objects in the future release. As illustrated in the picture below, the blue box represents a Report Object, such as Network Intent, which contains multiple Report Fields such as **Intent Name, Intent Type, Intent Map**, etc. The Report Fields will be listed in Report Field Tree for use in the Report.



Report Objects are mainly used in Report Properties to help define the basic data structure of a report. Users must select one Primary Object and can add up to three Related Objects for the report.



Each report must have and can only have one **Primary Object**. The major purpose of the report should determine Primary Objects. For example, for a report of Intent Results, Report Object ***Network Intent Result*** should be selected as Primary Object. After the Primary Object is selected, the Report Fields under the Report Object will be listed in Report Field Tree for use in the Report.

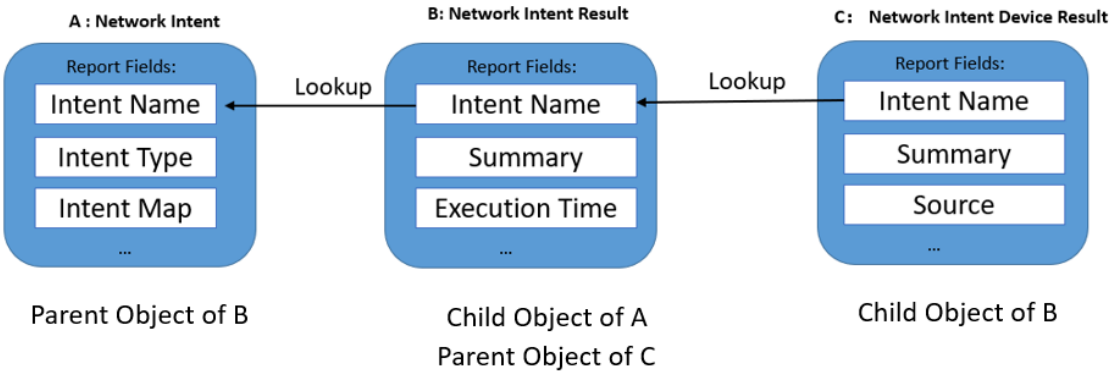


6.3.1.2 Lookup Relationships between Report Objects

In the database, the Report Objects are related via **Lookup Relationships**. Through the lookup relationship, different Report Objects are related via key Report Fields so that multiple Report Objects can be used in the same report.

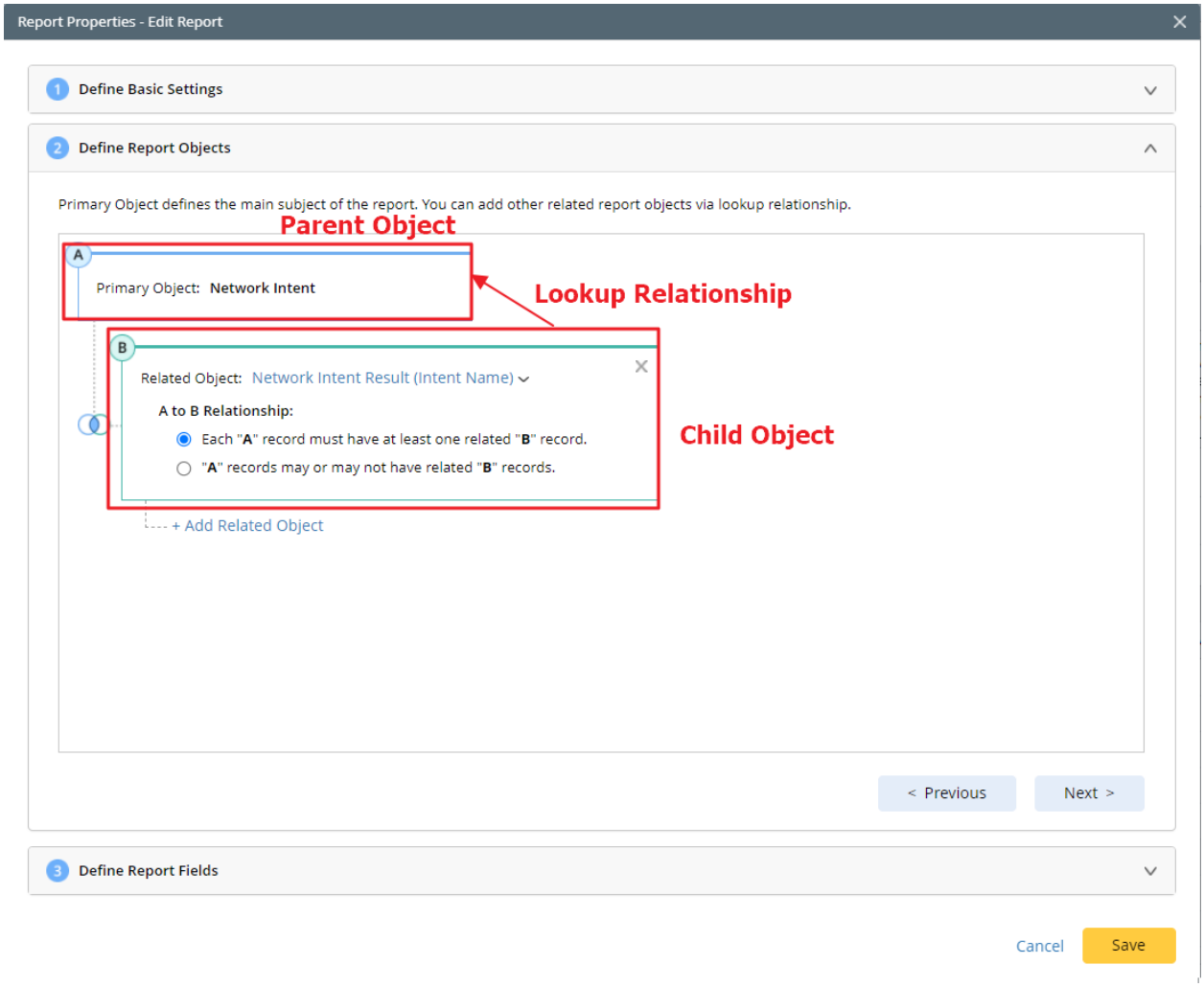
The diagram below illustrates an example of the Report Objects and their relationships. The Network Object B (***Network Intent Result***) lookup to A (***Network Intent***) via Report Field ***Intent Name***, and A (***Network Intent Result***) can be considered as the Parent Object to B (***Network Intent***). Similarly, Report Object C (***Network Intent Device Result***) lookup to B (***Network Intent Result***) via Report Field ***Intent Name***. The B (***Network Intent Result***) can be considered as the Parent Object to C (***Network Intent Device Result***).

Please note that this lookup relationship is one-way, so it makes a difference when adding related objects and related fields in Report Property.

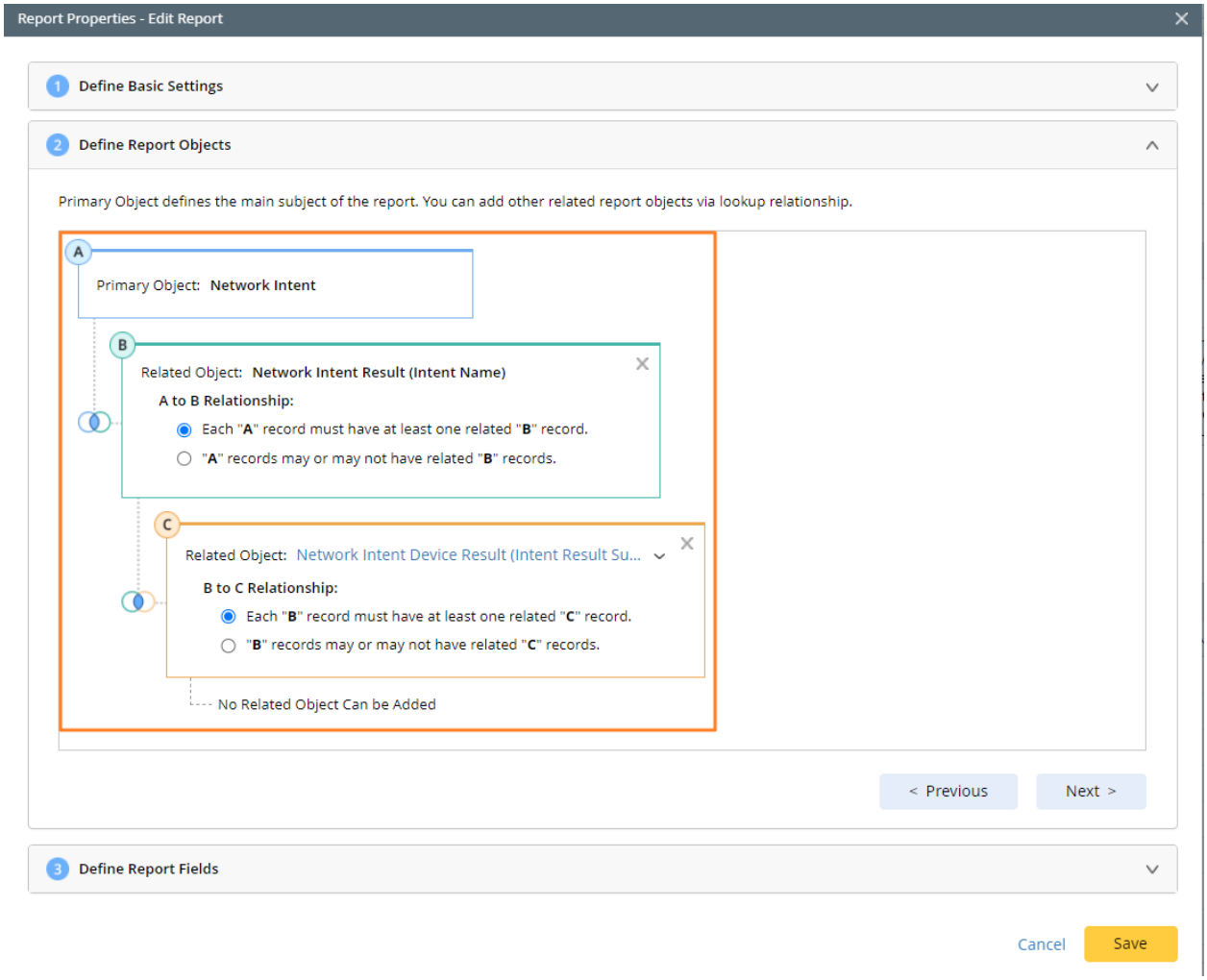


6.3.1.3 Define Complex Report Using Multiple Report Objects

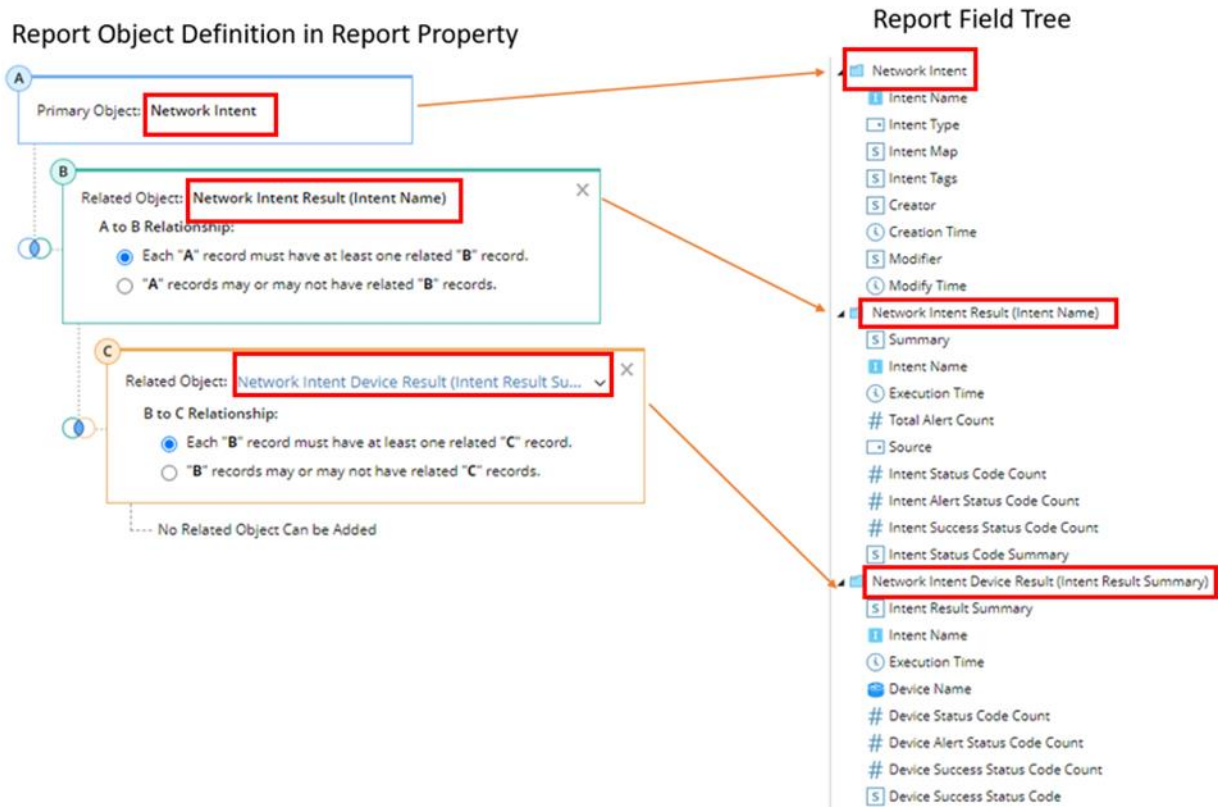
For Reports that require complex data structures, users can have multiple Report Objects in a Report by adding Related Report Objects via a lookup relationship. Also, users can add up to three Related Fields to provide additional context to the Report. And the Primary Object should be the Report Object that occupies the highest level in the hierarchy of the Report Object relationship.



For example, to build a report that includes **Intent type**, **Intent results**, and **Intent device results** as report columns, use three Report Objects: **Network Intent, Network Intent Result** and **Network Intent Device Result**. Report Object A (***Network Intent***), which occupies the highest level in the lookup relationship, is selected as Primary Object, then Report Object B (***Network Intent Result***) is added as a Related Object, and Report Object C (***Network Intent Device Result***) is added last.



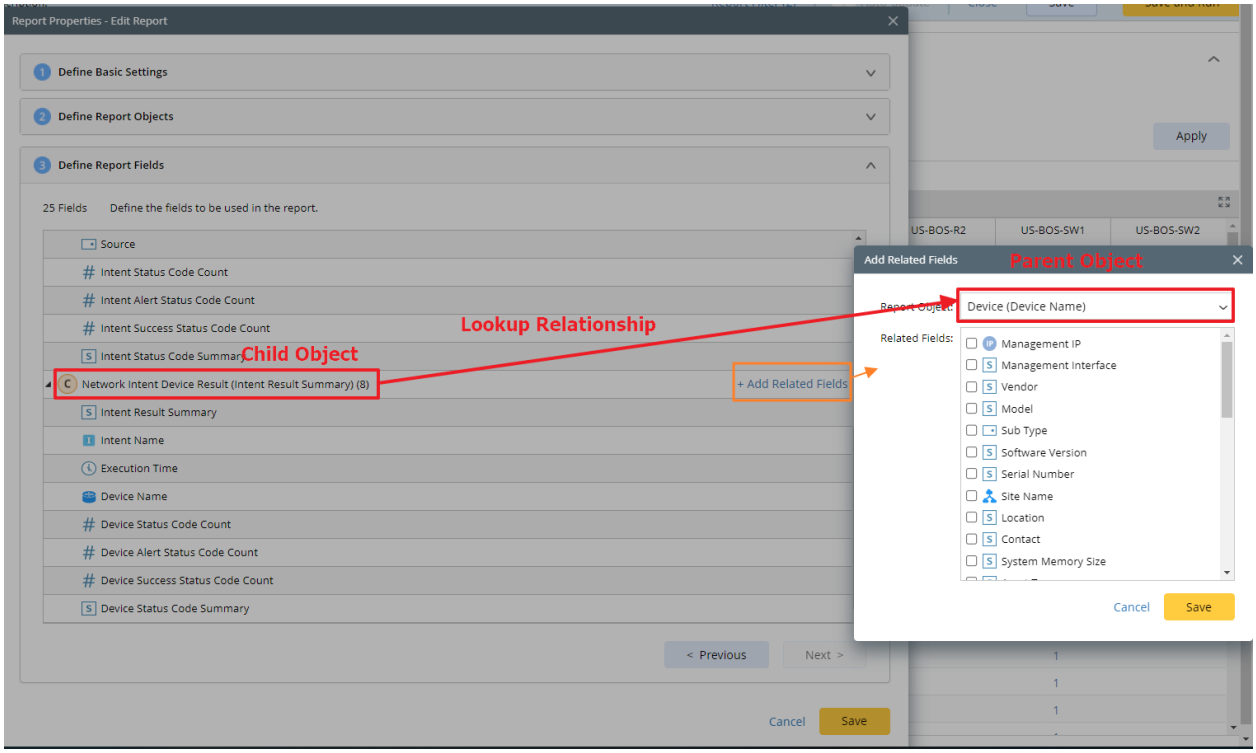
The Selected Report Object and the report fields under them will help form the barebone of the Report Field Tree, which specifies the basic data structure of the report. The picture below illustrates the relationship between the Report Object definition in Report Property and the Report Field Tree formed accordingly.



6.3.1.4 Define Report Fields

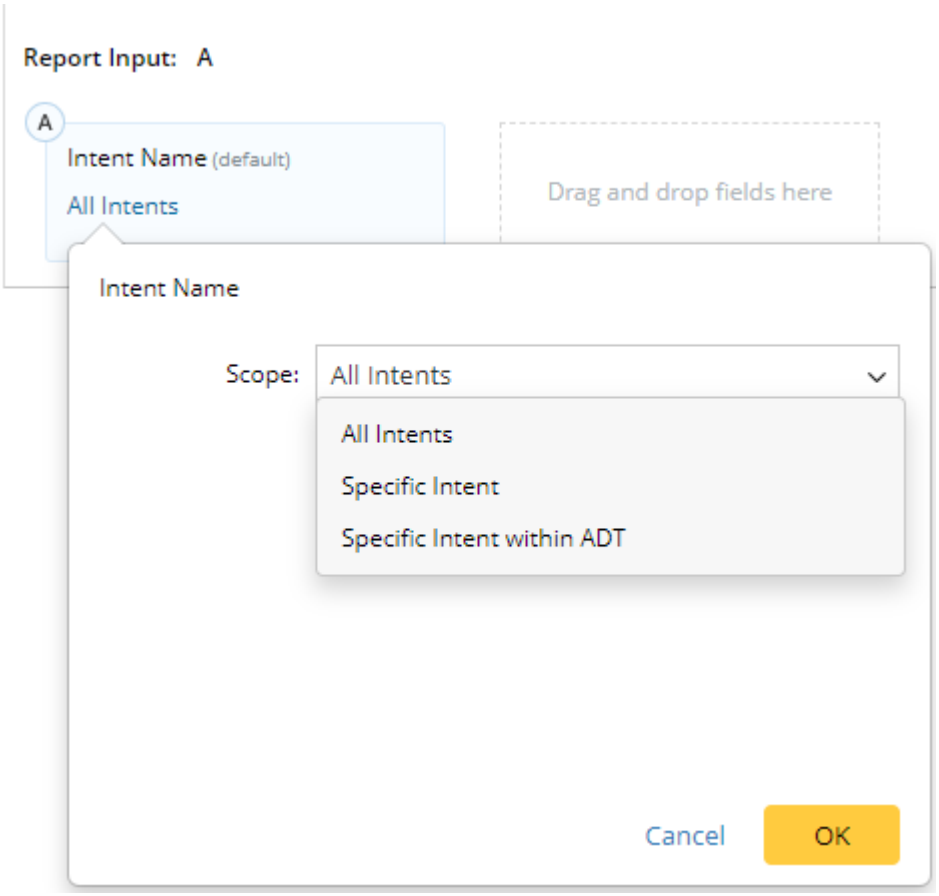
All Report Fields of the selected Report Objects are listed to provide a preview of the Report Field Tree. Besides these Report Fields, users can also add Related Fields to help provide additional information to the report.

To add Related Fields, select the Report Object. The drop-down list will list all the Parent Objects of Report Object A.

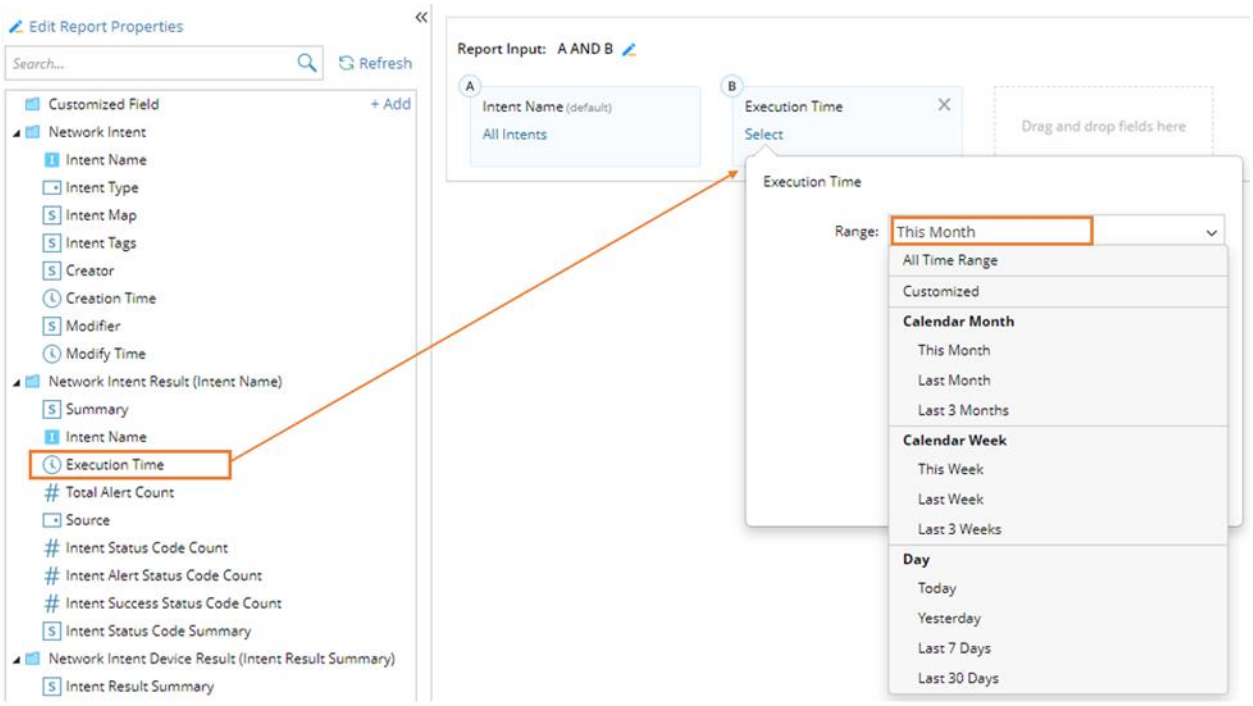


6.3.2 Define Report Input

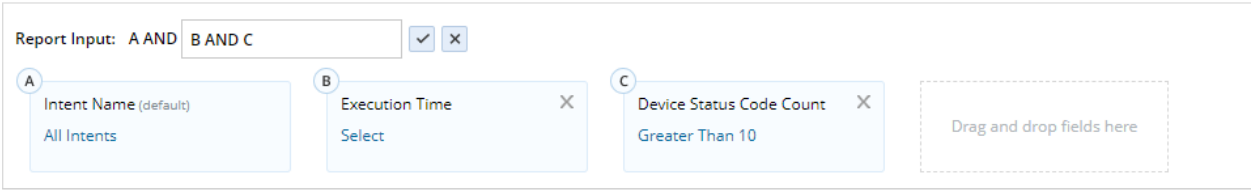
Report Input defines the report rows with filtering criteria. It selects the data in the NetBrain database, and only records meeting the criteria specified in Report Input are included in the report. Each report has one built-in default input and can have multiple additional report inputs.



Default Input is included in the report by default and cannot be deleted. It helps provide first-level filtering to narrow down the data scope. The default input is usually related to the Primary Object of the report. For example, if the Primary Object of the report is Network Intent, then the default input is Intent Name, allowing you to select the desired Intent by name or by ADT.



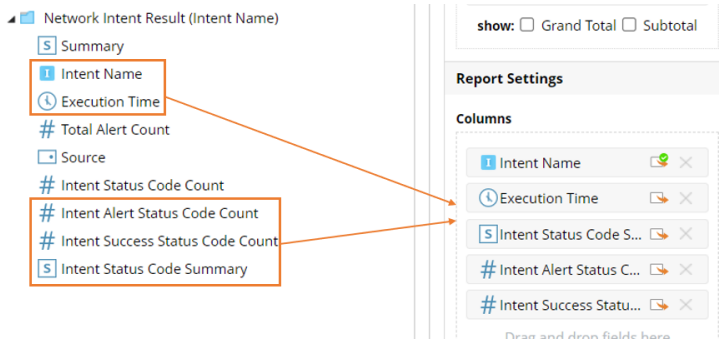
Besides default input, you can add multiple additional inputs by drag-and-drop Report Fields from the Report Field Tree to Report Input. For example, to filter the report content by execution time, you can drag and drop *execution time* to Report Input and set the time range as *This Month*.



The relationship between Report Inputs can be managed by clicking the edit icon beside the logical expression. Please note that the default input will always have an 'AND' relationship with the other additional inputs.

6.3.3 Define Report Columns

By drag-and-drop report fields from the Report Tree to Report Columns, users can define the report's columns. Users can also adjust the column orders by moving the report fields up and down in the Report Columns.



Besides built-in report fields, users can define customized fields using formulas and use them in the report. There are two ways to define a customized field: **Calculated Field** and **Conditional Field**.

6.3.3.1 Calculated Field

Calculated Field allows you to define a customized field using functions and mathematical and logical operators. To define the formula, select the available functions and fields from the tree on the left side and insert them into the formula. All the report fields in the Report Field Tree can be used except for other customized fields.

Besides the standard mathematical formula and logical operators, the following functions are supported for this release:

Function Name	Example	Description
ToString	*string* **ToString**(*number* or *datetime* or *boolean* $input)	Converts a number, a datetime or a boolean to string format.
Duration	*number Duration (datetime $startDate, datetime $endDate, string $unit)*	Calculate the time difference between the start date and the end date. The output is calculated by the unit and given in number format. Supported unit includes week, day, hour and minute.
Year	*number* **Year** (*datetime* $date)	Subtract the year from a date

Month	*number* **Month** (*datetime* $date)	Subtract the month from a date, a number between 1(January) and 12(December)
Day	*number* **Day** (*datetime* $date)	Subtract the day from a date, a number between 1 and 31
Hour	*number* **Hour** (*datetime* $date)	Subtract the hour from a date, a number between 0 and 23
WeekofYear	*number* **WeekofYear** (*datetime* $date)	Subtract the week from a date, a number between 0 and 53
Now	*datetime* **Now** ()	Returns the current time in datetime format
Ceil	*number* **Ceil** (*number* $input)	Returns the smallest integer that is greater or equal to the given number.
Floor	*number* **Floor** (*number* $input)	Returns the largest integer that is smaller or equal to the given number.
IF	*number* or *string* or *datetime* **IF** (*logical expression* $expression, $value_if_true, $value_if_false)	First verifies if an expression is true or false, then returns a given value based on the verification result. The data type of variable $value_if_true and $value_if_false should be the same.

6.3.3.2 Conditional Field

Unlike Calculated Field, which is based on user-defined formulas, Conditional Field provides quick and easy-to-use UI-based conditions for the most common customized fields. For the Conditional Field, the report field can only be of data type **'number', 'picklist' and 'DataTime'**.

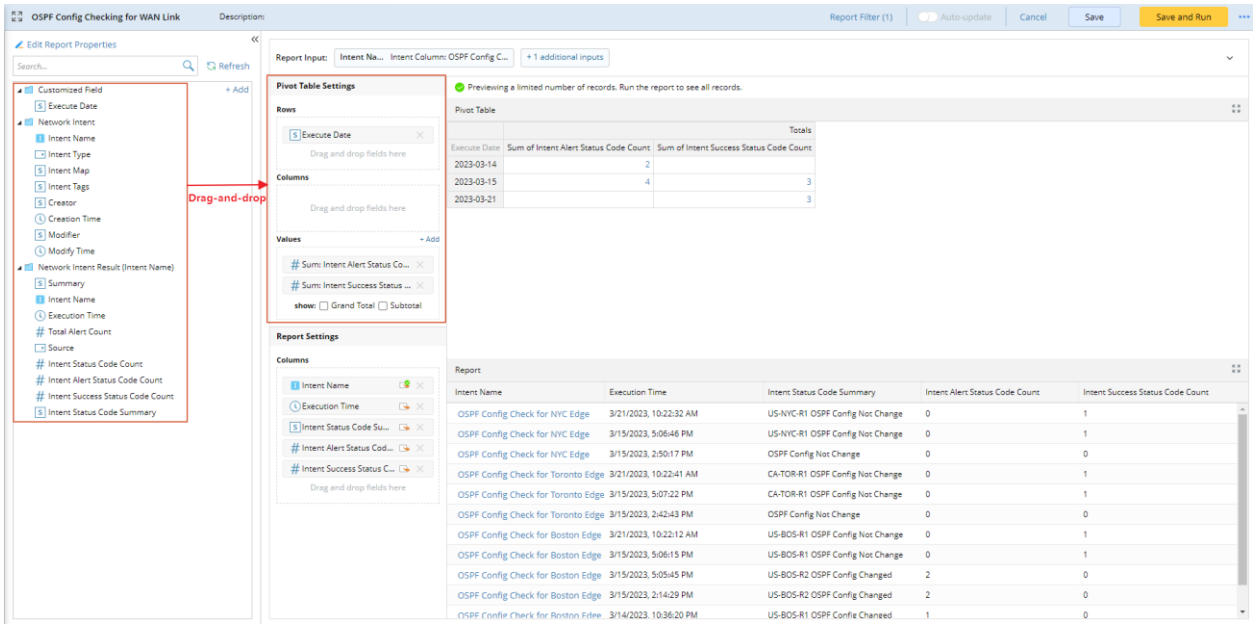
6.3.4 Pivot Table

A **pivot table** is a data summarization and analysis tool that organizes, summarizes, and analyzes data from a report. In a Pivot Table, data is organized into rows and columns, and users can reorganize the data to look at different aspects of it, enabling users to identify trends and patterns in the data and perform various calculations and comparisons.

There are two primary purposes for the Pivot table:

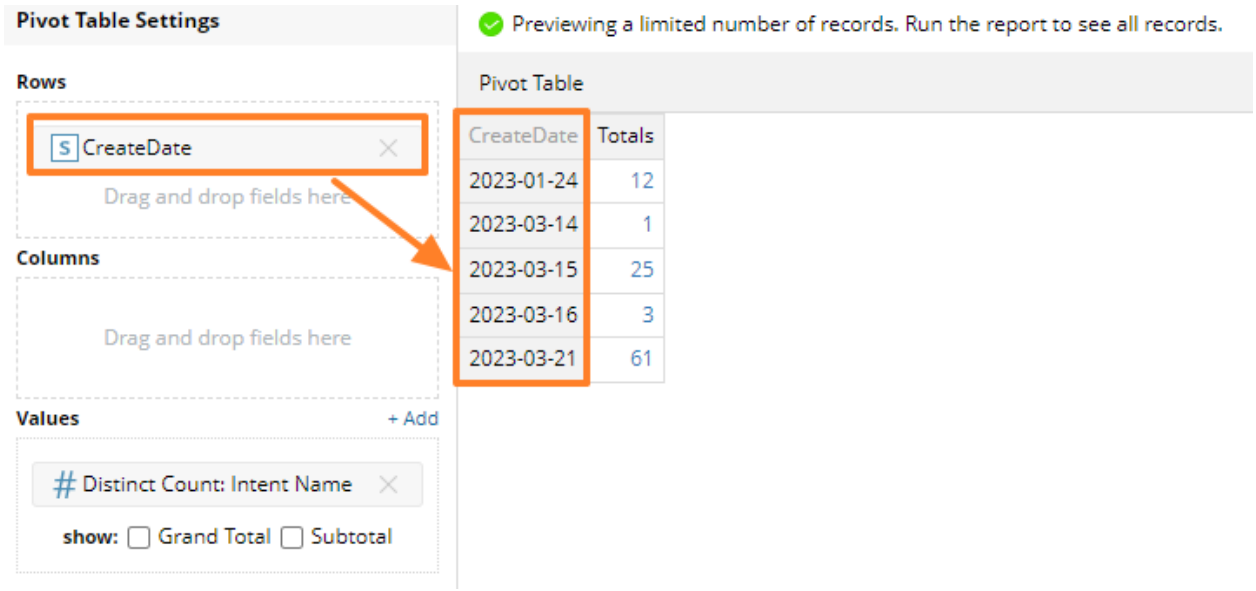
- To analyze large amounts of data in a Report by summarizing it into a more manageable and meaningful format.
- To assist in defining the Dashboard by providing the data needed for Dashboard charts.

In the report editor page, drag-and-drop fields from Report Field Tree to Pivot Table Settings to define the Rows and Columns of Pivot Table. Then click **+Add** to define the values of the Pivot Table.

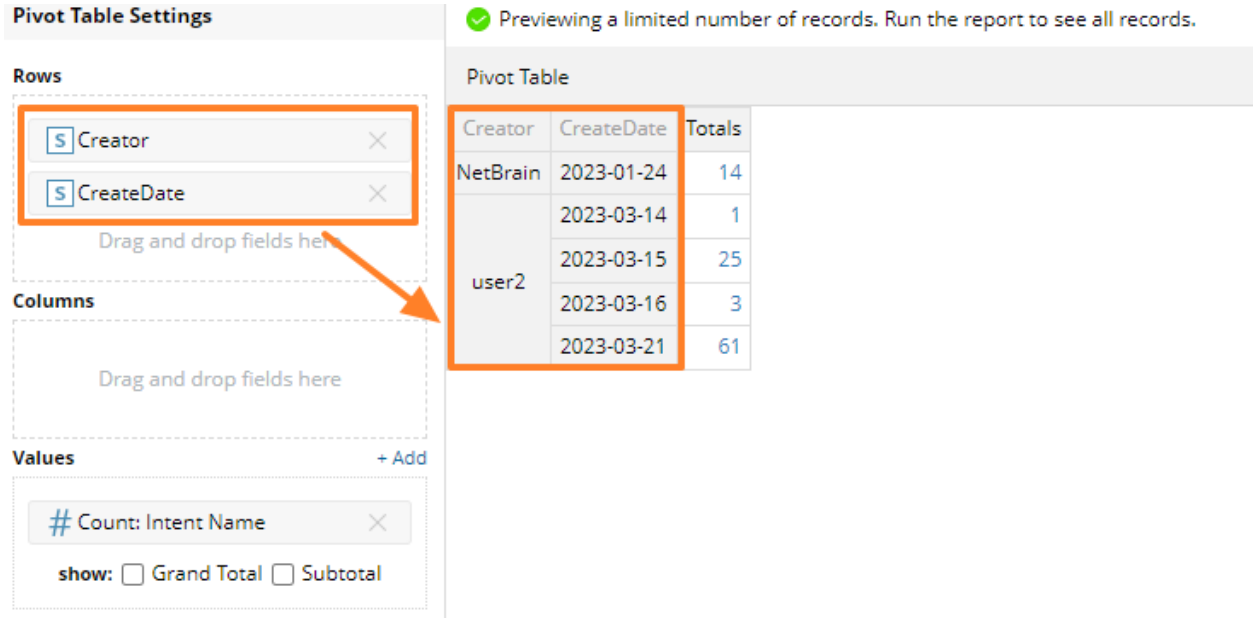


6.3.4.1 Pivot Table Rows

The rows in Pivot Table represent the categories by which the data is grouped and summarized. Any available fields in the Report Field Tree can be used as Pivot Table rows. As illustrated in the picture below, the Rows for Pivot Table will become the first column in Pivot Table.

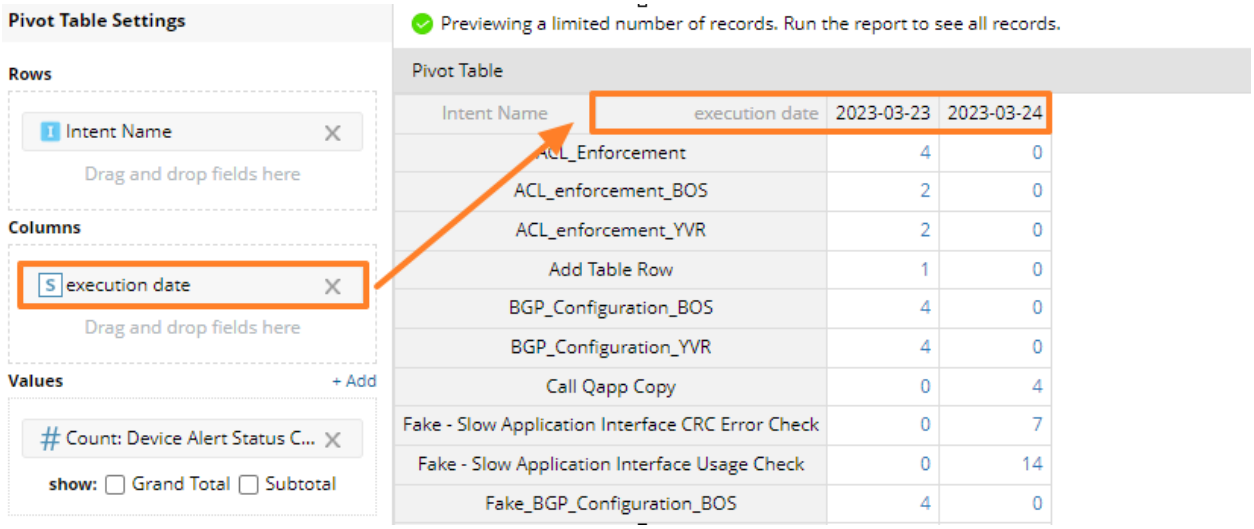


A Pivot Table can have multiple grouping by rows. It provides a multi-layer of data grouping and can be used to drill down into more details. Each Pivot Table can have up to four groupings by Rows or columns. When there are multiple rows in a Pivot Table, the data is grouped hierarchically based on the order of rows. Each row represents a grouping level, with the first row being the highest level and the last row being the lowest level. As the picture below illustrates, the Pivot Table will group the data first by Creator, then by CreateDate. Each Row will represent a grouping level, with the Creator row being the highest level and the CreateDate Row being the lowest.

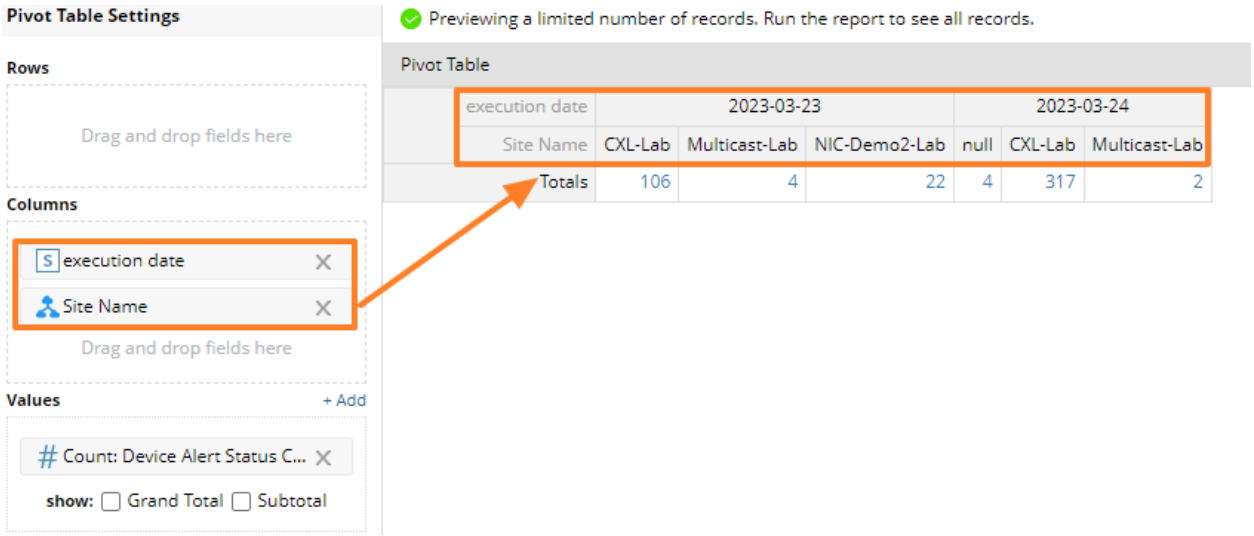


6.3.4.2 Pivot Table Columns

Like Pivot Table Rows, the Column in Pivot Table also allows you to group data by certain report fields. Pivot Table Columns provide another way to aggregate data and help you identify important trends that may not be obvious when viewing data by rows. The Pivot Table will create a new column for each unique value in the selected report fields when grouping by columns. The Pivot Table will then display the summary data for each column.



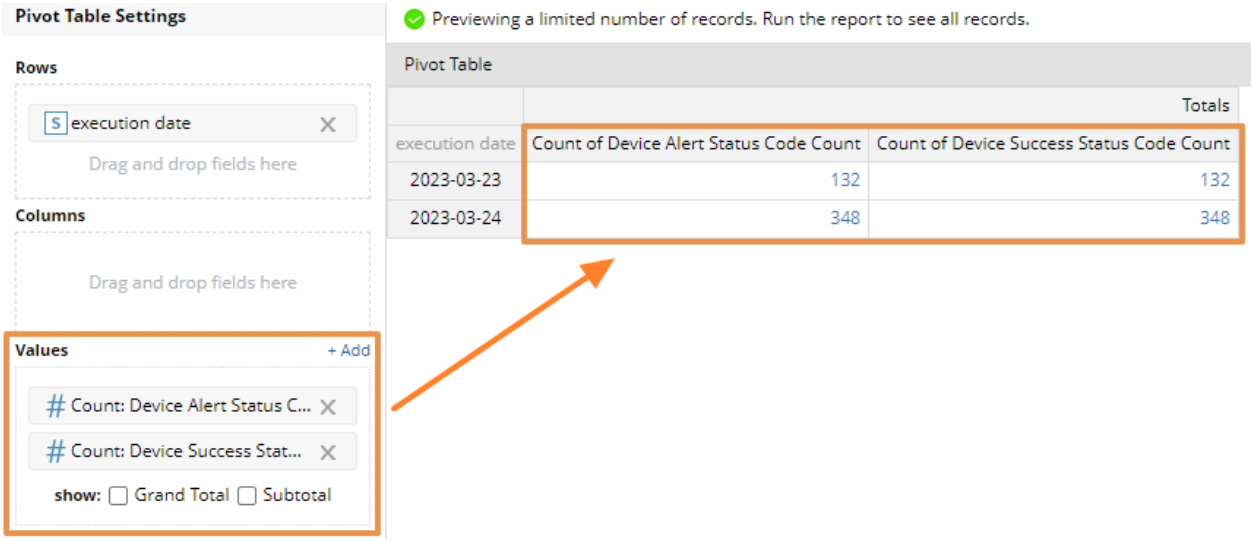
Like Pivot Table Rows, a Pivot Table can have multiple groupings by columns. Each Pivot Table can have up to four groupings by Rows or columns. When there are multiple columns in a Pivot Table, the data is grouped hierarchically based on the order of columns. As illustrated in the picture below, the Pivot Table will group the data by execution date, then by Site Name.



6.3.4.3 Pivot Table Values

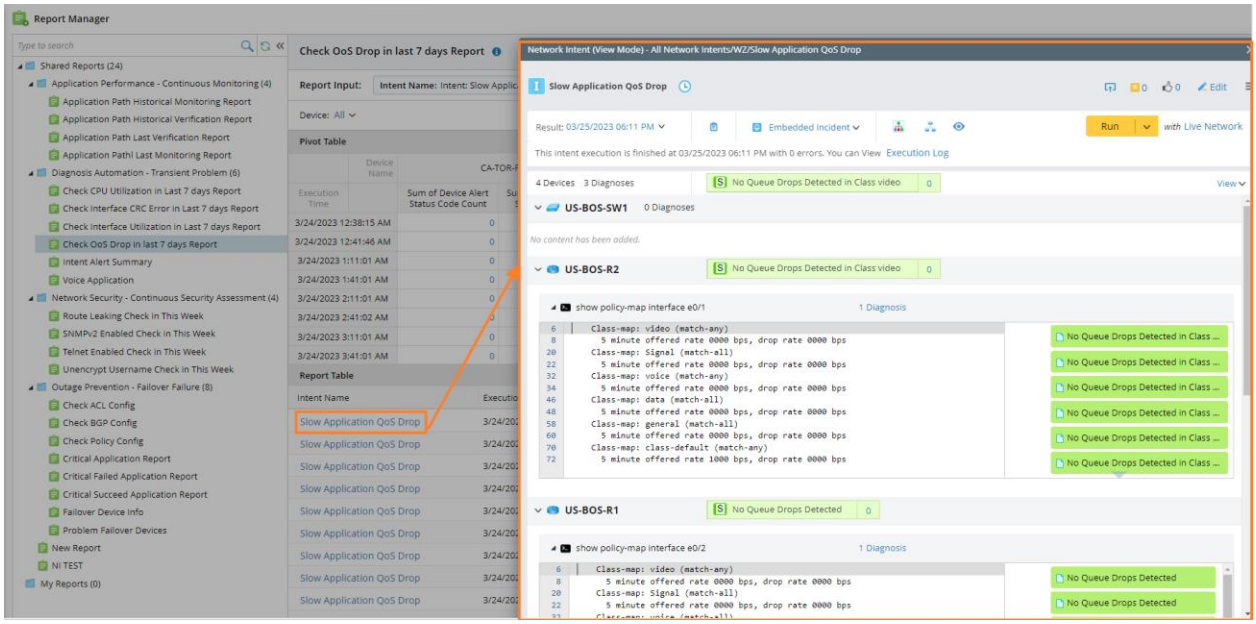
In Pivot Table, the Values are the numerical data being summarized and displayed. When defining values for Pivot Table, you can choose one or more report columns to use as Values. Then you can also specify the calculation method for the Values, such as **sum, average, minimum, count**, etc. The available calculation method varies depending on the data type of the Report Field selected.

These Values are displayed in the body of the Pivot Table, with Rows and Columns defining the categories or dimensions you want to analyze.



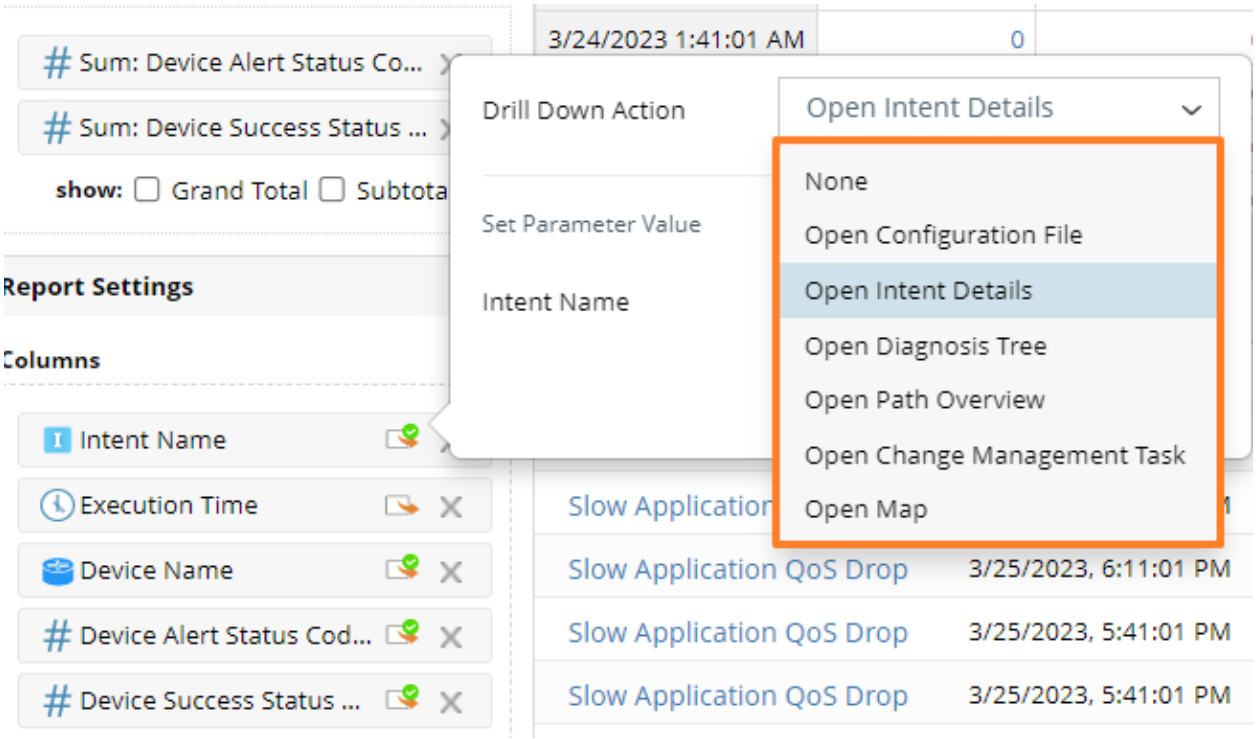
6.3.5 Drill-down Actions

Drill-down actions in the Report provide navigation from Report data to related NetBrain features by hyperlinks in report columns, providing deeper insights for Reports by drilling down into the underlying data.



There are six types of actions to select from for Drill-down actions. Besides that, you can also select one of the Report Columns as the parameter for the Drill-down actions. The parameter defines the details of the Drill-down Actions, such as the name of the Intent to be opened.

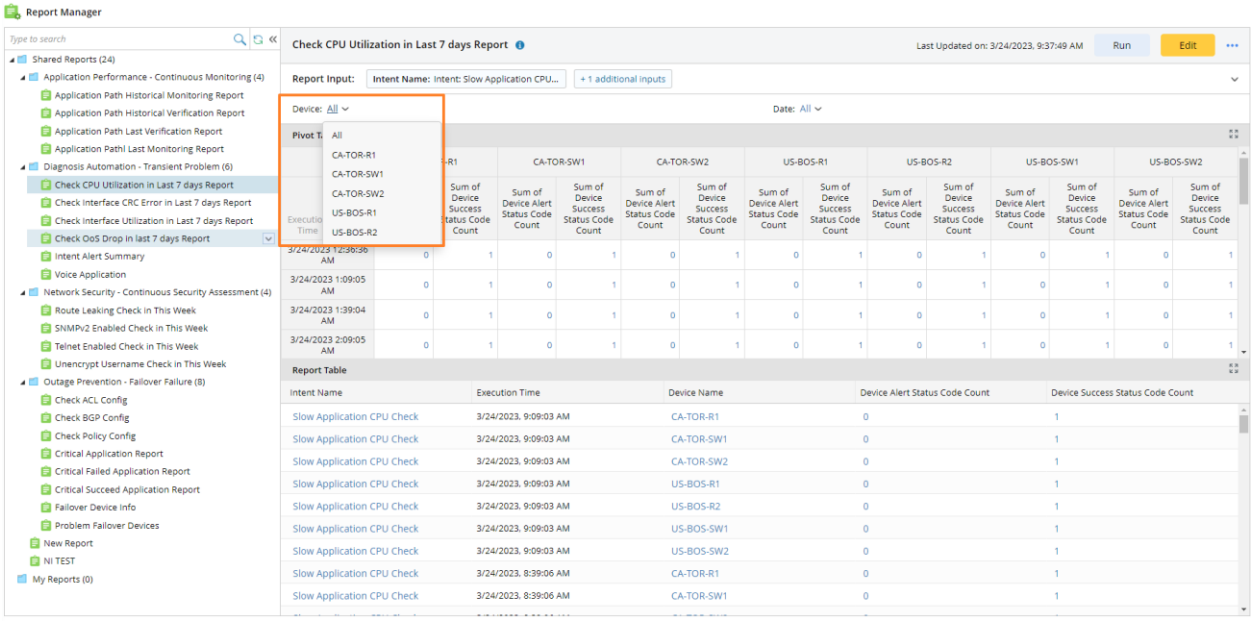
- **Open Configuration File** – Open the configuration file for the device. The parameter defines the value of the device name.
- **Open Intent Details** – Open the Network Intent in View Mode. The name of the Intent is defined by the parameter.
- **Open Diagnosis Tree** - Open the Diagnosis Tree of an Intent execution result. The name of the Intent and execution time of the result are defined by the parameter.
- **Open Path Overview** – Open the Path Overview pane. The name of the Path and Verified Time is defined by the parameter.
- **Open Change Management Task** – Open the Change Management Task. The name of the Change Management Task is defined by the parameter.
- **Open Map** – Open the map in another window. The map name is defined by the parameter.



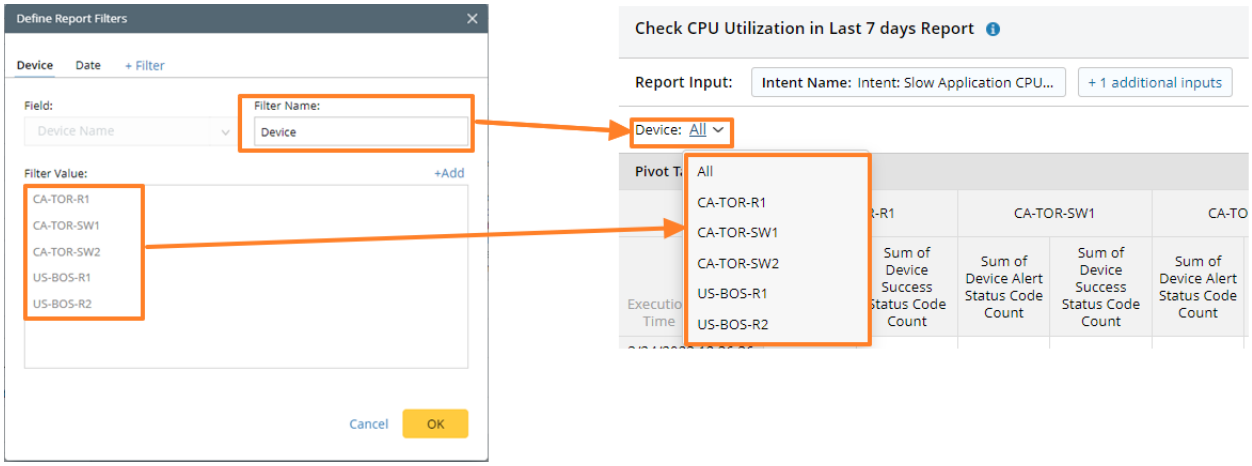
6.3.6 Report Filter

Report Filters can filter the report data by certain criteria when viewing a report. There are two purposes for using Report Filters:

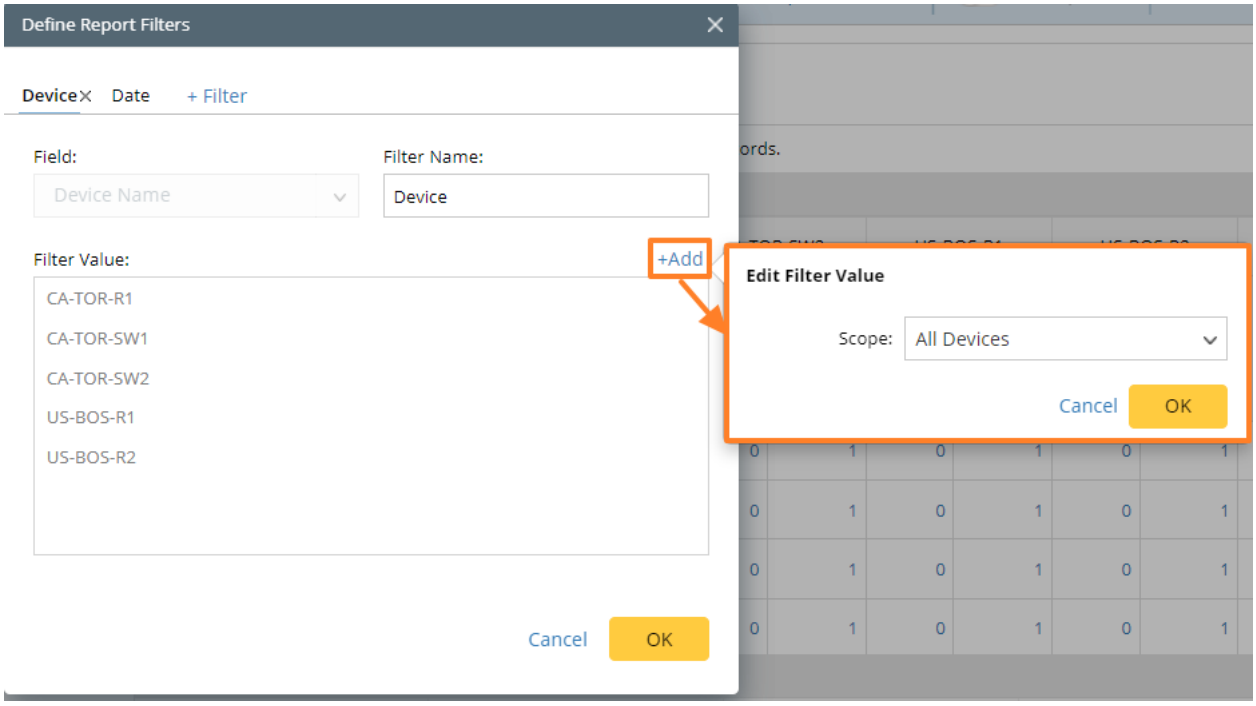
- Helps users to sort through a large amount of data and focus on important information.
- Displays data from different perspectives or scopes to help identify the trends or patterns that may not be obvious when viewing the report.



You can define up to three Report Filters for each Report on Edit Report Page. Each Report Filter has three key elements:



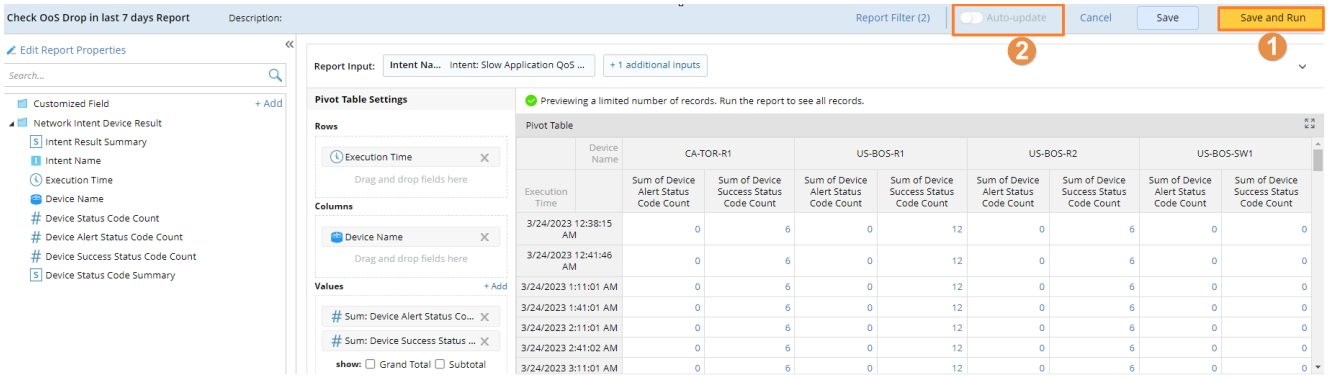
- **Field** – Select one of the Report Columns as Field from the drop-down list to define which Report Column the filtering criteria is applied to.
- **Filter Name** – Define the name of the Report Filter displayed on the View Report page.
- **Filter Value** –Define the options in the drop-down list for Report Filters. Click on +Add to add Filter Values. You can add up to ten Filter Values for each filter.



6.3.7 Run Report

Once the report is defined, you can run it to generate the data. The Report Edit page only provides a preview of the limited number of sample data, and you need to run the report to achieve accurate and complete data.

There are two ways to run a report: manually run it through the Save and Run button, and schedule to run it through Auto-update.

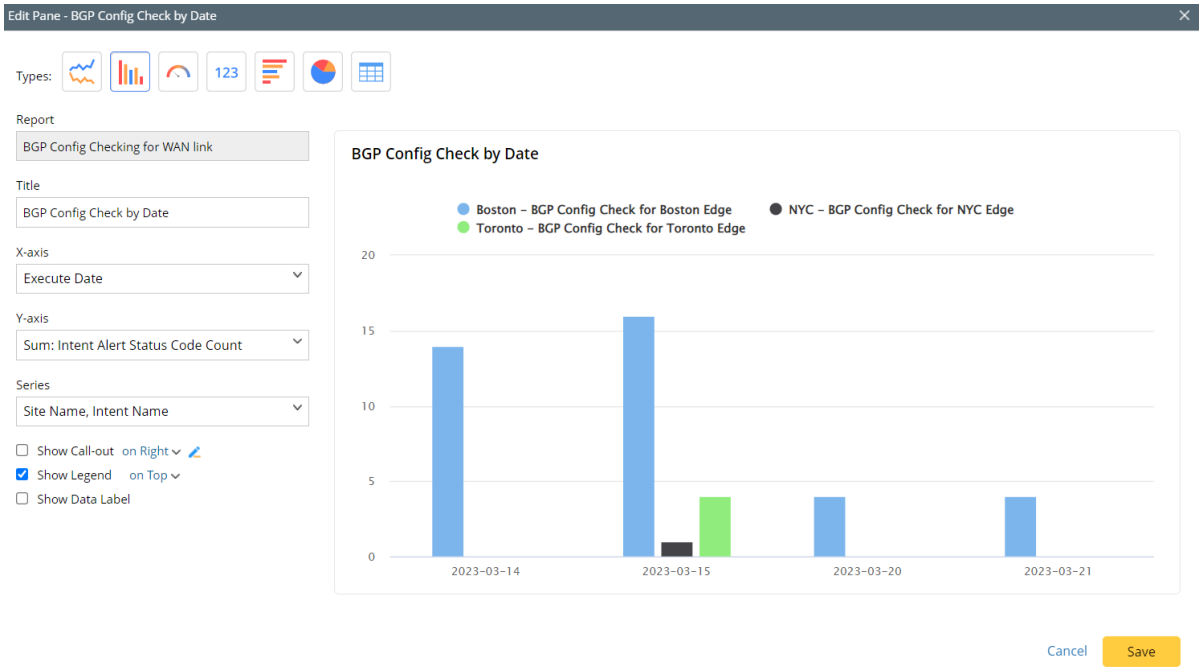


6.4 Create Dashboard

The Dashboard visually displays multiple reports in different charts for a specific purpose.

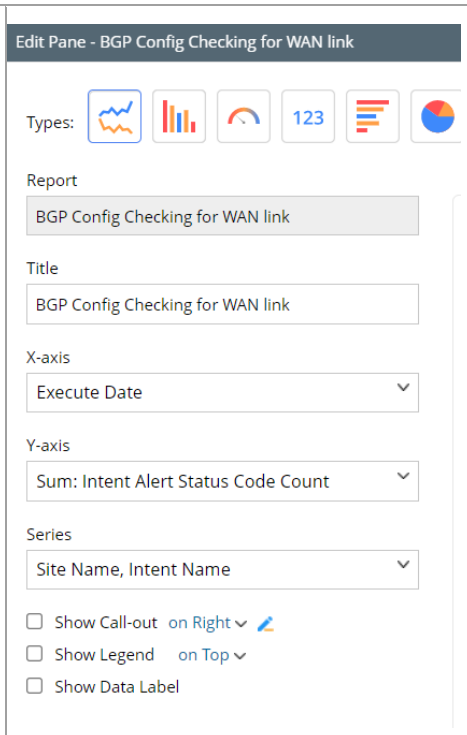
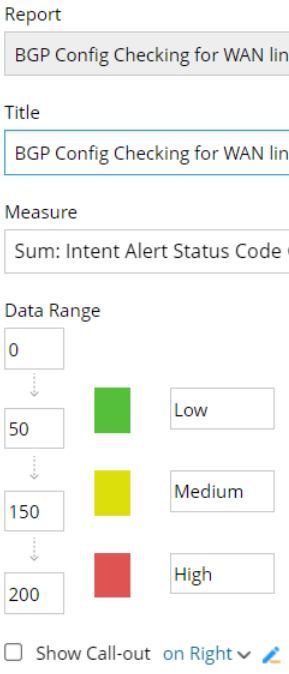
6.4.1 Chart

The dashboard uses various charts to display the analyzed result of the Report. Based on different pivot table settings, Dashboard provides a **column chart, bar chart, line chart, metric chart, gauge chart, pie chart** and **table chart** to display data.



As illustrated in the table below, the data that can be selected and used in the chart directly relates to the Pivot Table settings in the report.

Chart Type	UI Annotation for Chart Settings	Chart Settings

Line Chart, Bar Chart, Column Chart		• The x-axis of the line chart can be one or more of the Pivot Table Rows. • The y-axis of the line chart can be one or more of the Pivot Table Values. • The series of the line chart can be one or more of the Pivot Table Columns. • Check Show Call-out to display call-out on the chart. You can display the call-out on the right or bottom of the chart. • Check Show Legend to display the data legends on the chart. You can display the legend on the chart's top, right, or bottom. • Check Show Data Label to display the data details on the chart.
Gauge Chart		• The Measure of the Gauge Chart can be one or more of the grand totals of Pivot Table values, whether the total is enabled in Pivot Table. • You can modify the data range, colors, and labels for Gauge Chart.

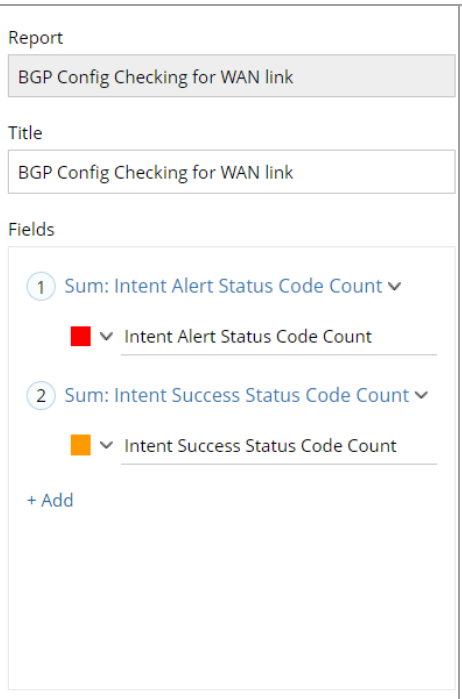
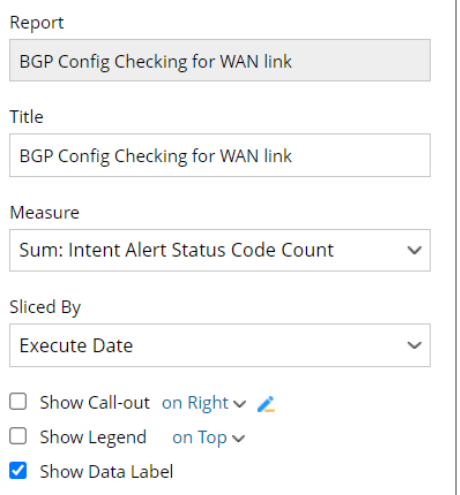
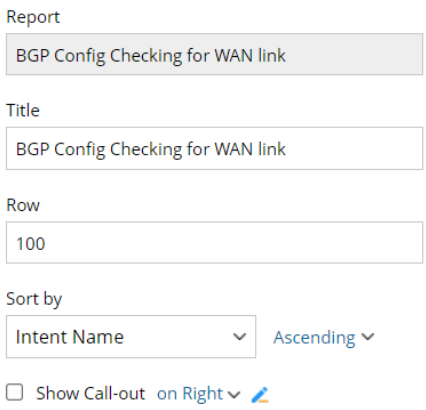
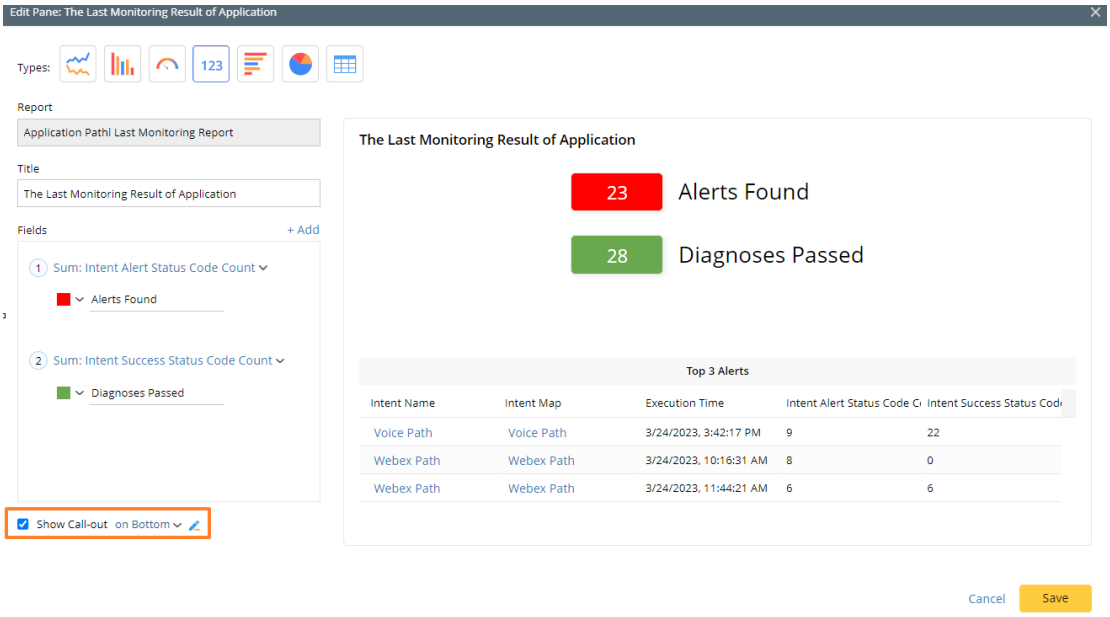
Metric Chart		• The data of the Fields in the Metric Chart can be one of the grand totals of Pivot Table values, whether the grand total is enabled in Pivot Table or not. • You can modify the data, colors, and labels for Fields in Metric Chart. • You can add more than one Field in one Metric Chart.
Pie Chart		• The Measure of the Pie Chart can be one of the Pivot Table Values. • The 'Sliced By' of the Pie Chart can be one of the Pivot Table Rows or Columns. • Check Show Call-out to display call-out on the chart. You can display the call-out on the right or bottom of the chart. • Check Show Legend to display the data legends on the chart. You can display the legend on the chart's top, right, or bottom. • Check Show Data Label to display the data details on the chart.

Table Chart		• The Table Chart can display the top N records of the report, where N can be specified by users. • The Table Chart can be sorted by one of the Report Columns in ascending or descending order.

6.4.2 Call-Out

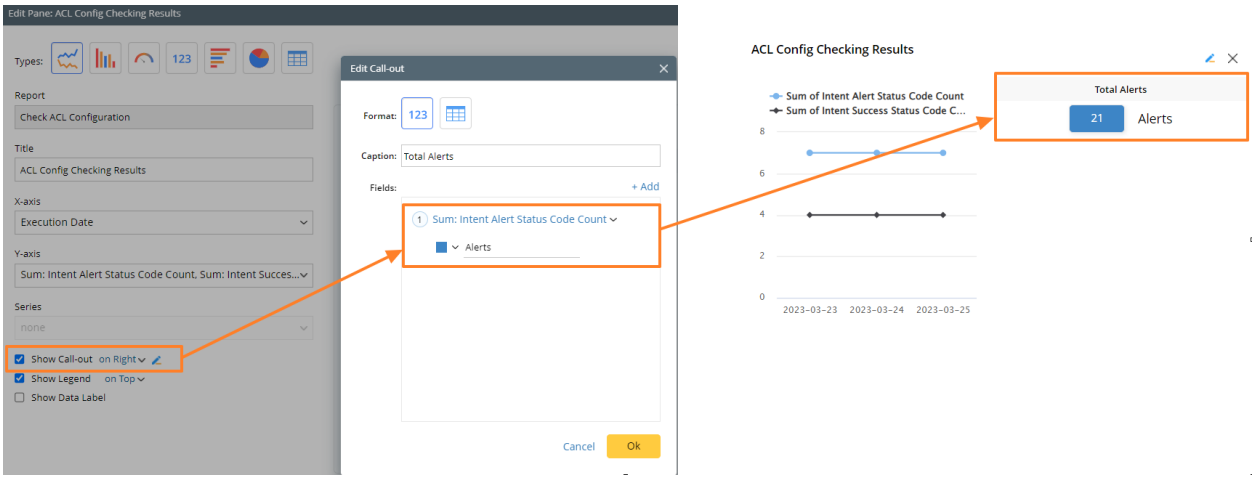
Besides the chart, each dashboard pane can have one call-out to help provide additional information. There are two main purposes for using the call-out:

- Provide additional details that are not obvious from the chart.
- Highlight important data or trends in the chart.



6.4.2.1 Define Metric Call-out

This type of call-out displays numerical data along with their corresponding units of measurement. It helps highlight important data in a clear and concise format.



Select the Metric Chart as the format in the Edit Call-out window to define a Metric Call-out. There are two key elements for Metric Call-out:

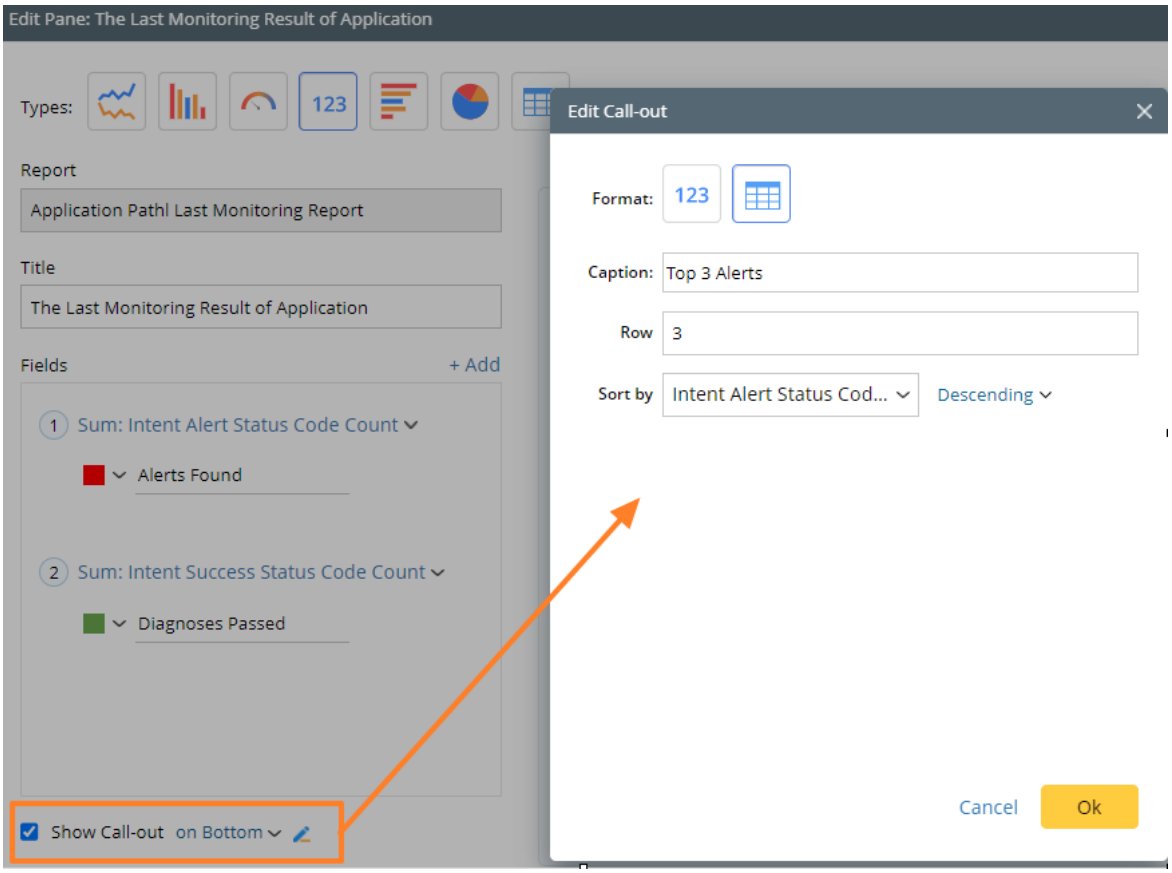
- **Captions** – Captions are descriptive labels on the top of the Call-out. It helps provide an overview of the content or purpose of a call-out.
- **Fields** - Fields are the data displayed in the Metric Call-out. Each Metric Call-out can have up to six Fields, while each Field displays one summarized data from the report. You can define a unique background color and label each field to help annotate the data.

6.4.2.2 Define Table Call-out

This type of call-out displays the top-N records of the original report. It helps provides additional details for the chart.

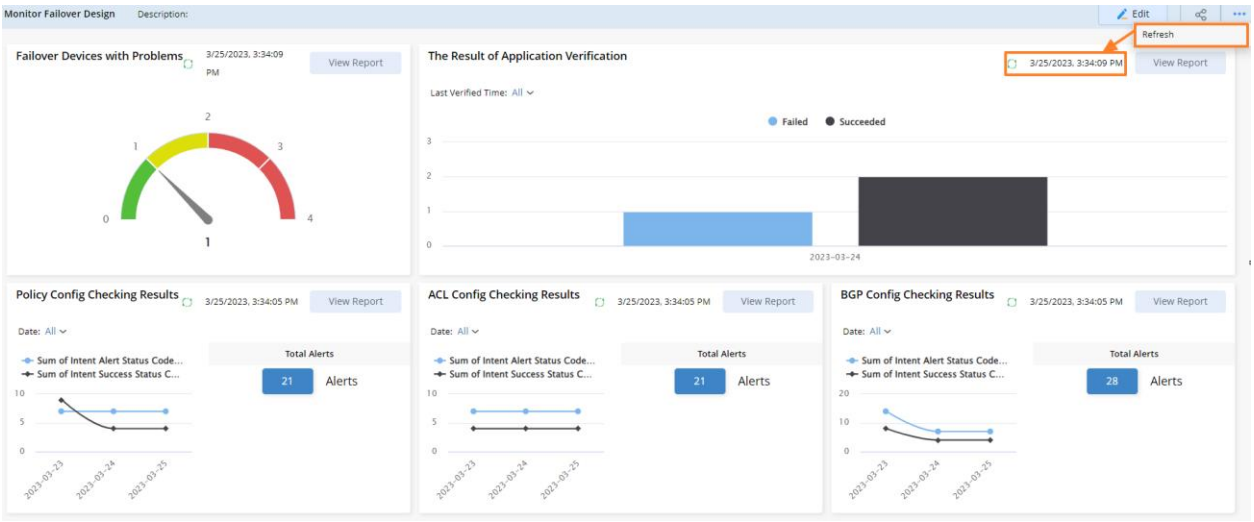
To define the Table Call-out, select the Table Chart as the format in the Edit Call-out window. There are three key elements for a Table Call-out:

- **Captions** – Captions are descriptive labels that help provide an overview of the content or purpose of the call-out.
- **Row** - Row defines how many rows of data appear in the Table Call-out.
- **Sort by** – Sort by defines by which report column the report data is sorted. You can choose to sort it in ascending or descending order.



6.4.3 Refresh Dashboard

Dashboard caches report data in the back end for faster loading speed. Refresh the Dashboard from the drop-down menu to get the latest report data and update the dashboard accordingly.



6.5 Report and Dashboard Examples

6.5.1 Monitor Failover Failure for Outage Prevention

Failover design is essential to network outage prevention to ensure the reliability of critical network services. However, sometimes failover failure could happen due to inadequate testing or configuration errors. To monitor the failover designs and find potential risks in advance, a dashboard could be built to answer the following question:

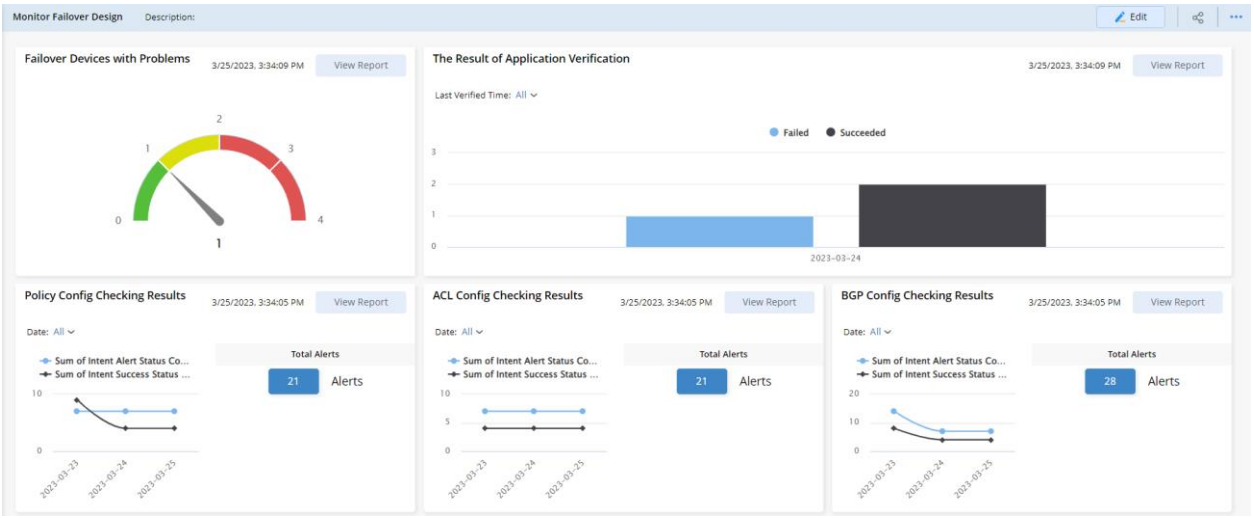
How is the robustness of the failover design in the network?

To answer this question, a few follow-up questions could be asked:

- Are the ACLs consistent between the primary and failover devices?

- Are the BGP configurations consistent between the primary and failover devices?
- Are the policy maps consistent between the primary and failover devices?
- Which devices have potential risks?
- When are the risks detected?
- Is there any impact on critical applications?

For each follow-up question, you can build a report to answer it. Then a dashboard could summarize and display those reports on a single screen.



6.5.2 Automate Diagnosis of Transient Problems

Diagnosing transient problems can be challenging because they occur intermittently or briefly, making them difficult to reproduce and troubleshoot. An Intent is scheduled to run every 10 minutes to troubleshoot the slowness in the voice application. A dashboard can be built to answer the following questions:

What are the possible causes of slowness in the voice application?

To answer this question, a few follow-up questions could be asked:

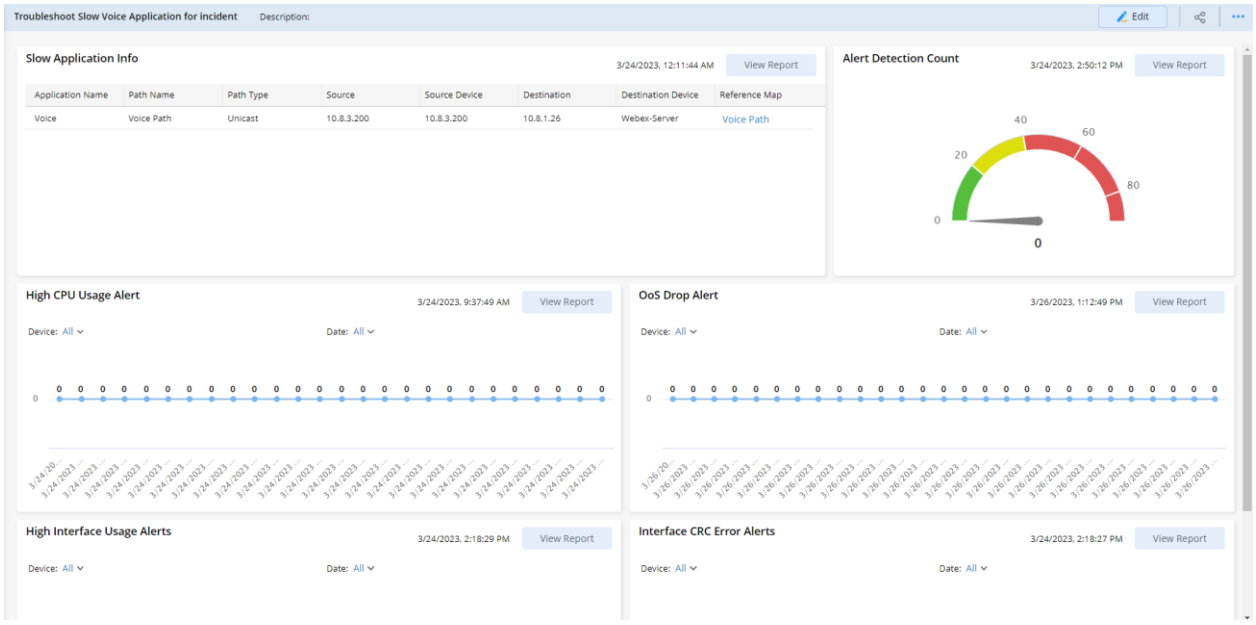
- Is there high CPU usage in Voice Application?

- Is there QoS drop in Voice Application?
- Is there high interface usage in Voice Application?
- Are there interface CRC errors in Voice Application?
- When are the alerts detected?
- How many alerts in total are detected?

Reports and Dashboards can be created to analyze and display the results from the following Intents:

- **Check CPU Utilization**: This Path Intent is scheduled to collect results of the command, show process CPU, for all the devices in the path and generate an alert if the CPU usage is above 80%.
- **Check QoS Packet Drop**: This Path Intent is scheduled to generate an alert whenever there is a QoS packet drop for all devices in the path.
- **Check Interface Usage:** This Path Intent is scheduled to monitor the interface usage for all the devices in the path and generate an alert if the interface usage is above 70%.
- **Check Interface CRC Error**: This Path Intent is scheduled to check the interface CRC error for all the devices in the path. It will generate an alert if there is any CRC error detected.

For each follow-up question, you can build a report to answer it. Then a dashboard could summarize and display those reports on a single screen.



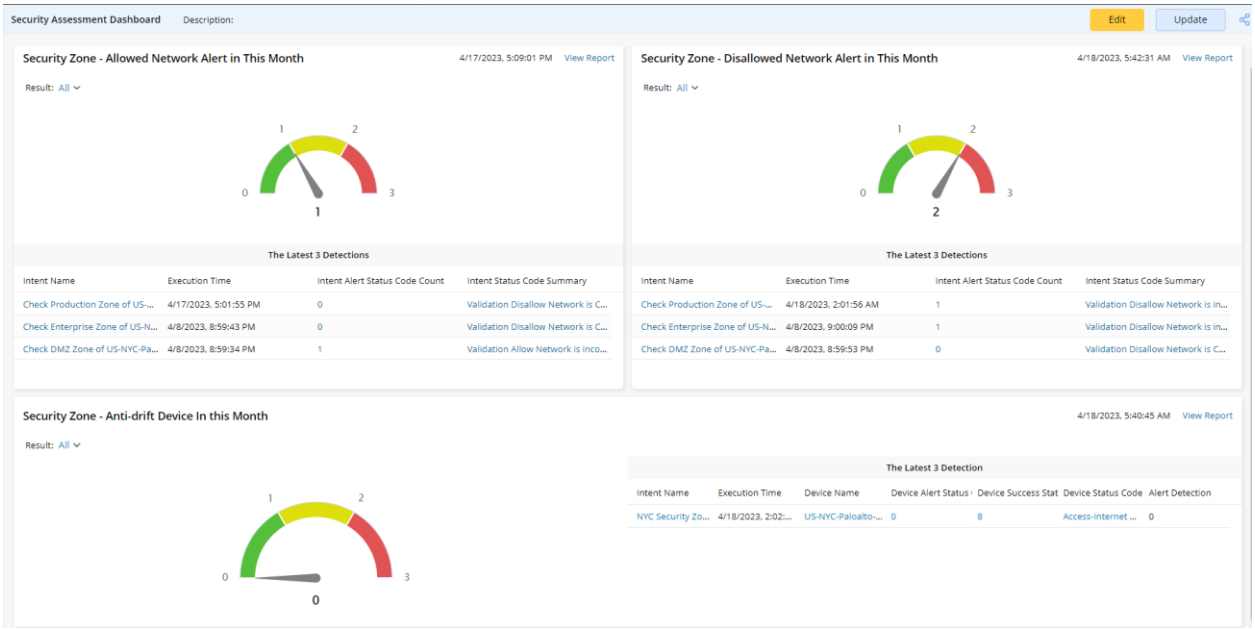
6.5.3 Perform Security Assessment for Network Security

Network Security Assessment is critical to maintaining a secure and resilient network infrastructure. To conduct the security assessment regularly, a series of Network Intents are scheduled to run weekly. A dashboard can be built to answer the following questions:

Is there any security weakness in the network?

To answer this question, a few follow-up questions could be asked:

- Can the allowed devices successfully access the devices in the security zone?
- Are the disallowed devices successfully blocked by the security zone?
- Is there any configuration drift in the security zone?



6.5.4 Perform Continuous Monitoring of Application Performance

Continuous Monitoring of critical applications and paths can help identify potential problems, improve performance, and analyze network traffic patterns. Two Intent are scheduled to run continuously to monitor

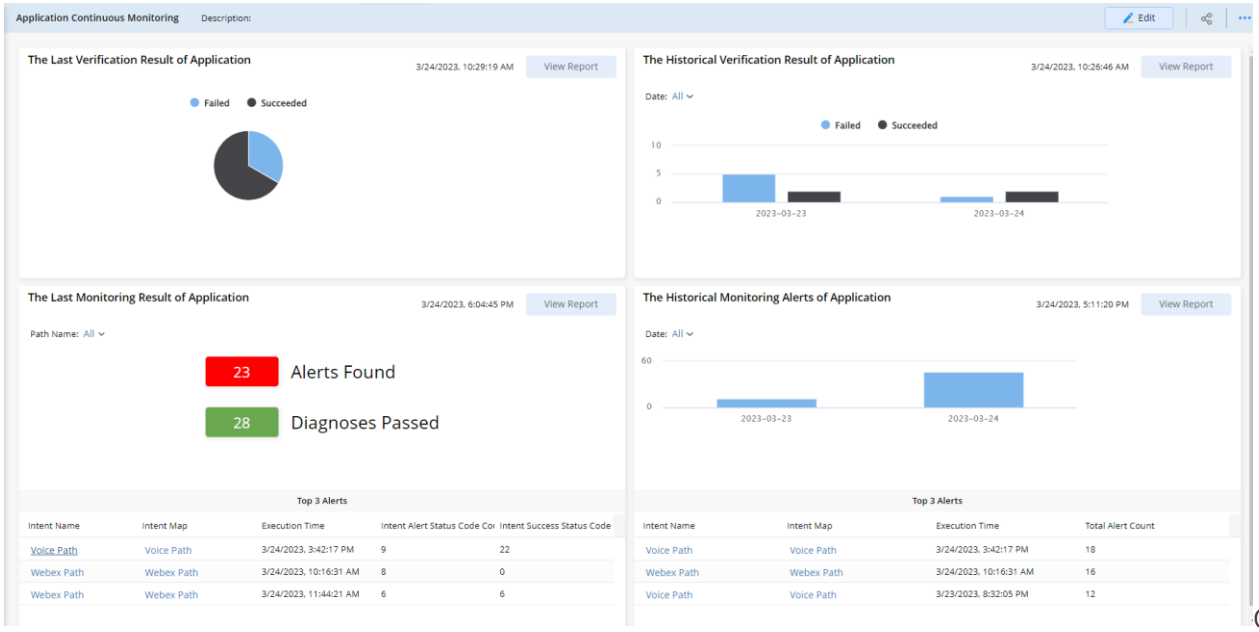
the key applications Voice Path and Webex Path in the network. A dashboard can be built to answer the following questions:

Are there any potential problems in critical applications?

To answer this question, a few follow-up questions could be asked:

- Which applications failed to be verified that require attention?
- Which applications are verified successfully?
- For the application failed to be verified, are there any failed to ping destination issues?
- For the application failed to be verified, are there any ACL misconfigurations?

For each follow-up question, you can build a report to answer it. Then a dashboard could summarize and display those reports on a single screen.



7 Other Enhancements

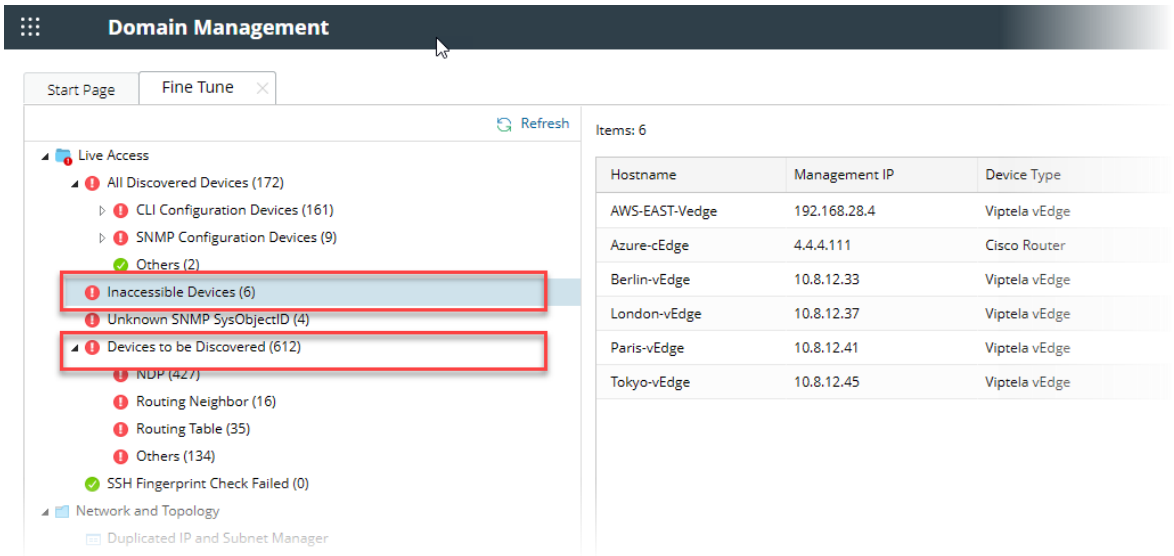
7.1 Domain Setup and Data Accuracy Improvements

7.1.1 Discovery and Fine Tune Improvements

7.1.1.1 Unknown IP and Do-Not-Scan Improvement

After the discovery, many entries will be generated in the table **Fine Tune\Devices to be Discovered**. Some of these IP addresses are not accessible or invalid, which makes cleaning up this table difficult. R11.1 provides automated ways to clean up these entries.

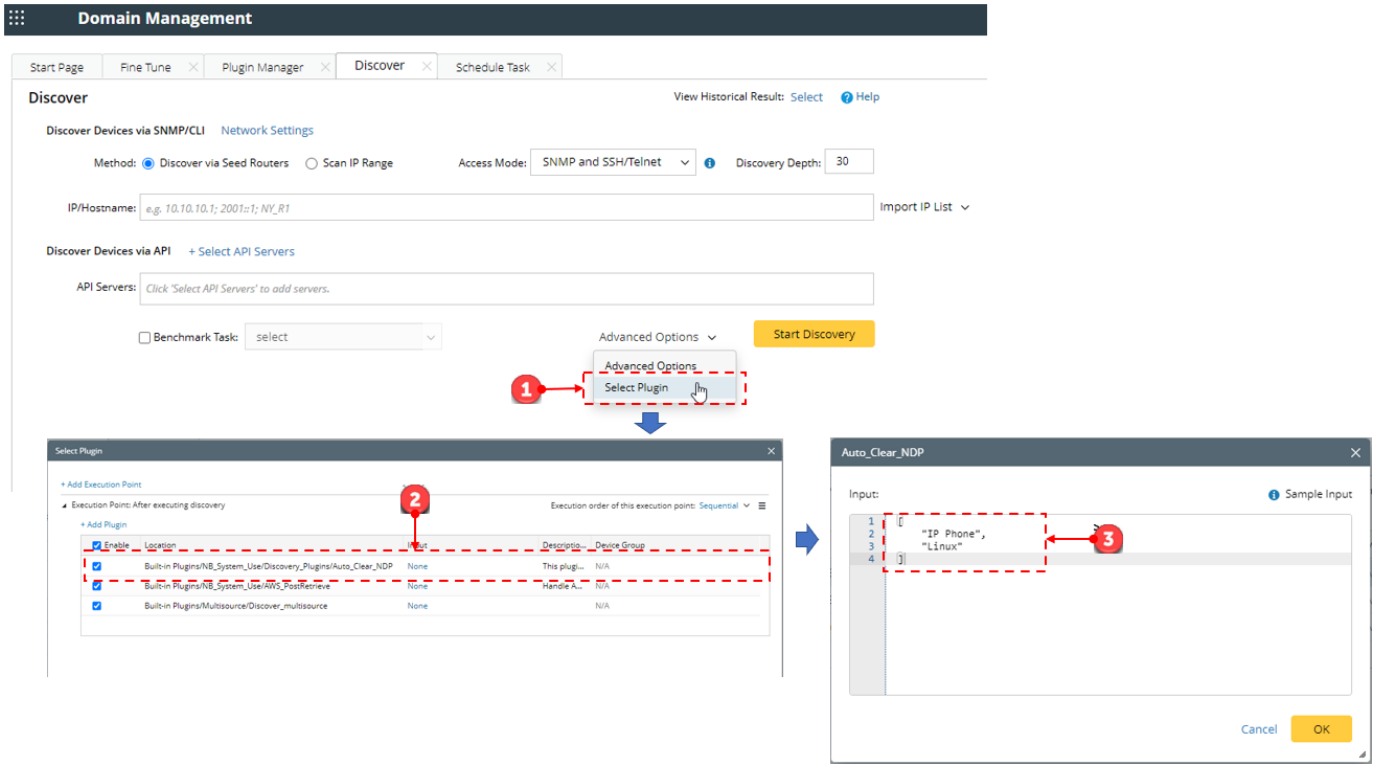
1. Changed the field name, **Missed Devices** to **Inaccessible Devices**, and **Unknown IPs** to **Devices to be Discovered**.



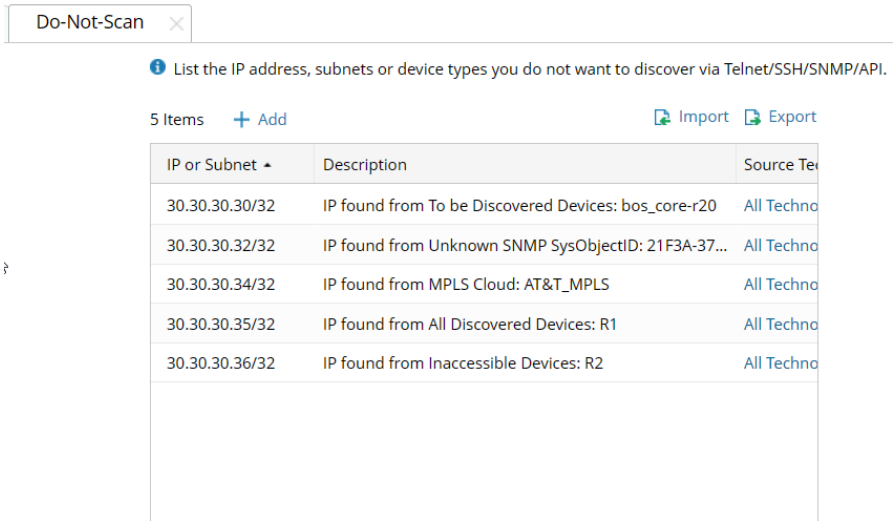
2. Automatically add MPLS PE IPs to the **Do-Not-Scan** table after creating and updating MPLS Cloud. The PE IPs already in the **Devices to be Discovered** table will be removed.
3. After the Generic Device is added, the IP address belonging to the Generic Device will be automatically deleted from the **Devices to be Discovered** table.

4. Provide a System Plugin, ***Auto_Clear_NDP,*** that automatically runs after the discovery to move the entries from the **Devices to be Discovered/NDP table** to the **Do-Not-Scan table**.

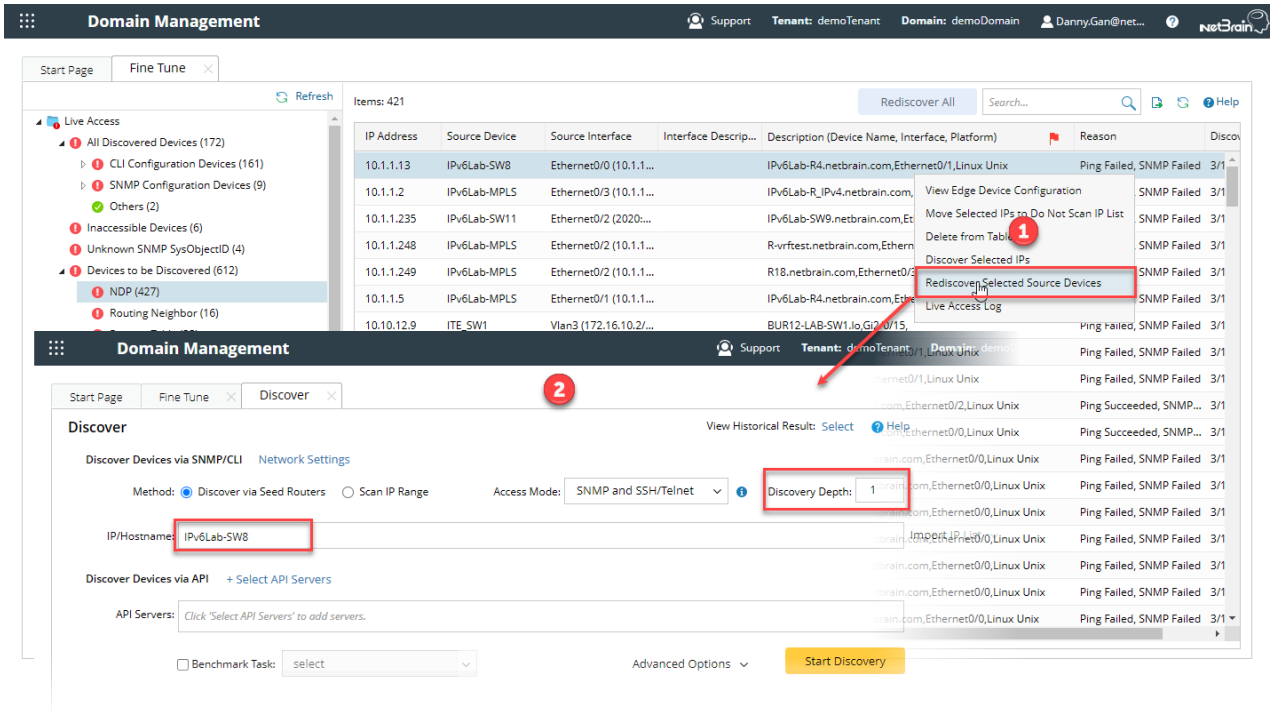
   The ***Auto_Clear_NDP*** plugin matches the keyword in the description column and moves them to the Do-Not-Scan table to reduce the noise. The user can modify the keyword, for example, Linux, IP Phone, etc.



5. Record the IP source in the Do-Not-Scan Table to help the user understand where the IP is from.

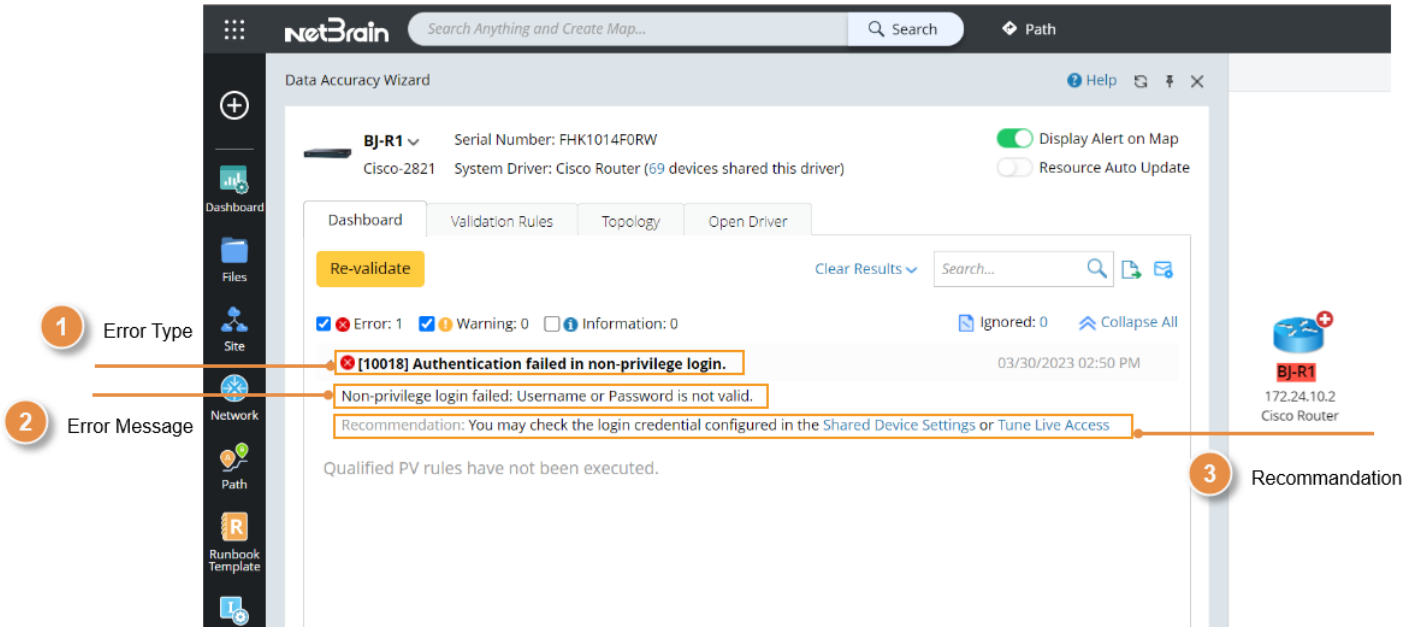


6. When the Source Device is deleted from the domain, its associated entries in the **Devices to be Discovered** table are automatically cleared.
7. Add **Rediscover Selected Source Devices** function in the **Devices to be Discovered** table, which will launch a new discover task using the Source Device as the seed and the depth set to 1. After rediscovering the source device, the system automatically clears the associated entries that are no longer in the Source Device's NDP table.



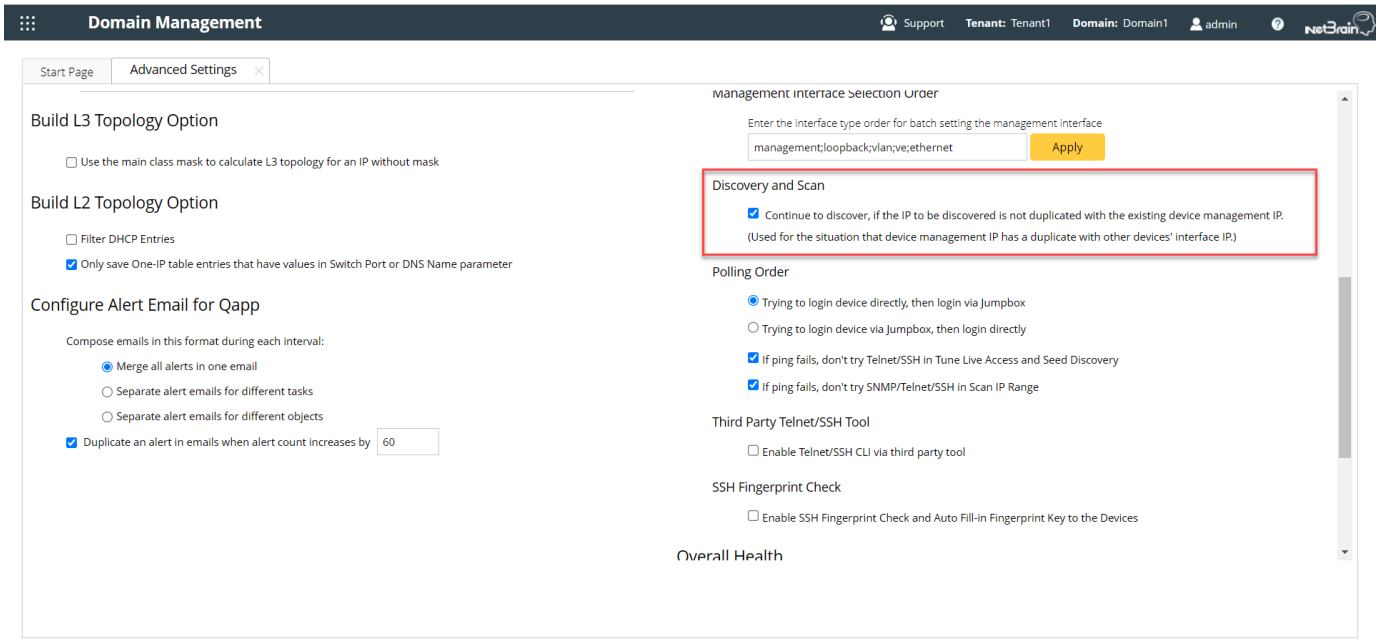
7.1.1.2 System Validation Improvements

R11.1 adds new error types to improve the CLI, API and server access errors. Also, the recommendation of existing error types is updated to guide the users more accurately in solving live access issues in the Data Accuracy Wizard pane.



7.1.1.3 Discover a Device with Duplicate IP

R11.1 provides an option in the Advanced Settings of Domain Management to enable the discovery of duplicate IPs in such customer networks. This option can be used to discover the device which has duplicated interface IP address (the management IP address must be unique). By default, this option is disabled. Enabling this option will impact the discovery performance.

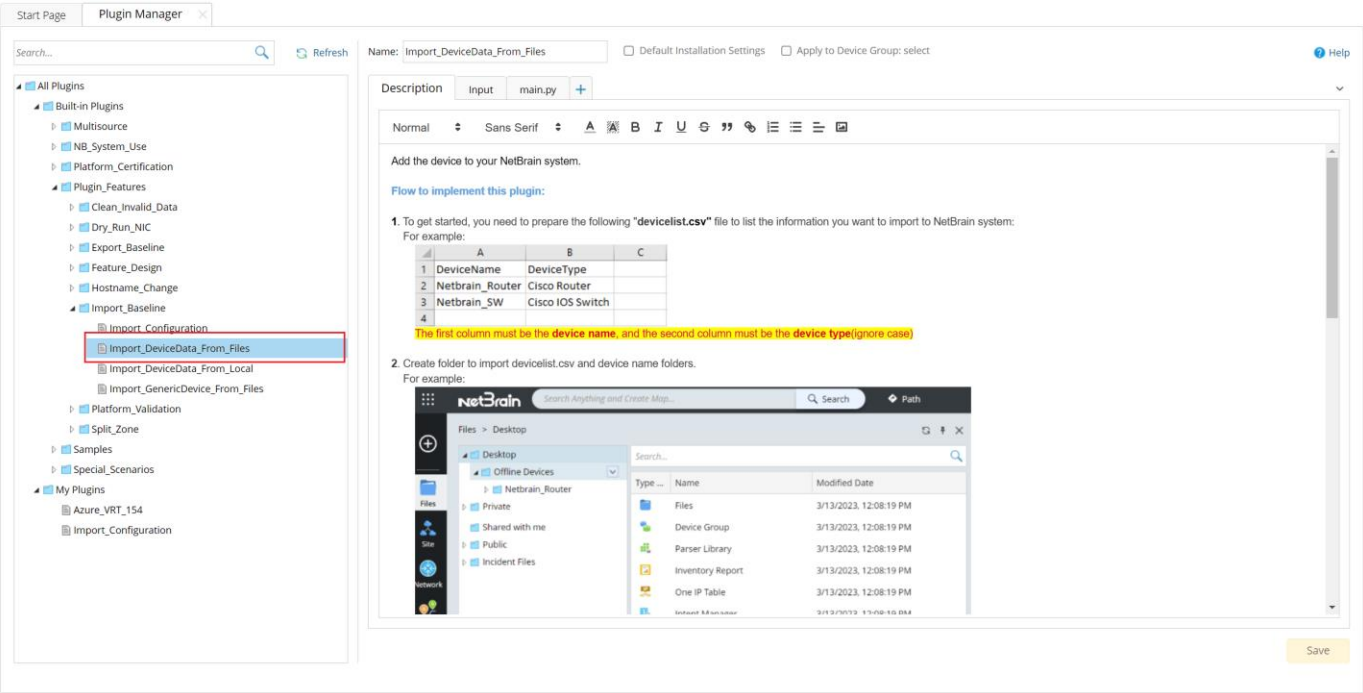


7.1.1.4 Import Devices Via Config Files/Pre-defined CSV

R11.1 provides two built-in plugins to import inaccessible devices to the NetBrain system.

- **Import Devices via Config Files:**

The plugin, ***Import_DeviceData_From_Files***, under the folder **/All Plugin/Built-in Plugins/Plugin_Features/Import_Baseline/** in the **plugin manager**, imports the devices by the configuration file.



- **Import Devices via Pre-defined CSV**

The plugin ***Import_GenericDevice_From_Files,*** under the folder **/All Plugin/Built-in Plugins/Plugin_Features/Import_Baseline/** in the **plugin manager**, imports the devices by CSV file.

The CSV file must contain the following information:

- If there is no IP address, the corresponding physical interface data will be generated.
- If there is an IP address, the corresponding physical and IP interface data will be generated.

The fields ***DeviceName, ManagementIP, DeviceType, DriverName,*** and ***IntfName*** are required. The other fields are optional.

DeviceName	ManagementIP	DeviceType	DriverName	IntfName	IPv4Address	IntfType	IntfVrf	IntfMode	IntfVlan
Device1	10.10.10.10	Cisco IOS Switch	Cisco IOS Switch	Ethernet0/0	192.168.25.1	*VLAN*	*global_vrf*		40
Device2	11.11.11.11	Juniper Router	Juniper Router	Ethernet0/1		*Physical*		*access*	100

7.1.2 Benchmark Improvements

R11.1 made the following improvements on the scheduled benchmark task:

- Remove the **BGP Advertised Route Table** from the System Table since we already have the NCT Table with the same information. Collecting the same data twice is a waste of resources and time-consuming.
- Modify the data collecting logic for **BGP Advertised Route Table** and **BGP All VPNv4 Advertised Route Table** in the Server Benchmark to only collect the data on the MPLS CE device and the BGP neighbor connected to MPLS Cloud. This logic also applies to Public Cloud VRT. The related nodes in Public Cloud are:
    - AWS: DX Router, VGW
    - Azure: MSEE, Virtual Network Gateway, and VPN Gateway
    - GCP: Cloud Router, Partner Interconnect, and VPN Gateway
- Add Circuit Breaks

  Collecting a large amount of data may be time-consuming, which could cause Server Benchmarks to run for several hours without completion. This delay in updating critical data such as topology could affect the regular system operations. However, by implementing a Circuit Break, the initial Server Benchmark may take longer to complete as it detects and records the big data items. The Circuit Break Benchmark can then collect these items at a lower frequency, resulting in smoother and quicker subsequent Server Benchmarks. Refer to Circuit Break in Server Benchmark for more information.

7.2 Topology Accuracy Improvements

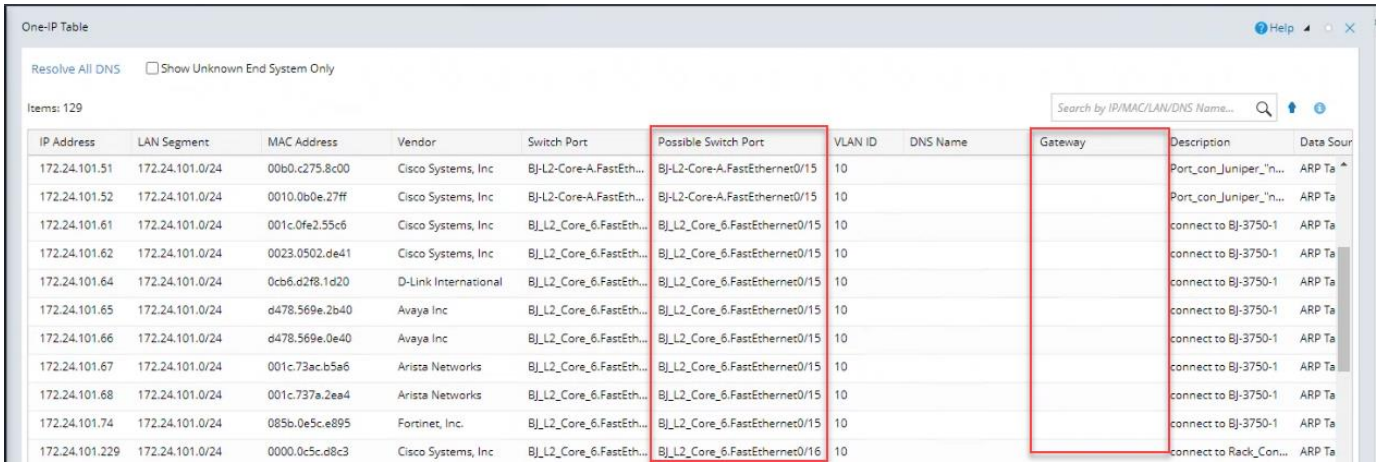
7.2.1 Open Topology Improvements

The Open Topology team has continuously improved and updated the main logic of the Open Topology algorithm:

- Improve the Proxy ARP in the VLAN Group algorithm to support more complex network situations.
- Support “Generic Device” in Open Topology.
- Optimize the performance.
- Use the Adaptive Plugin to address more customer cases, including:
    - Improve the HA support for more Vendors and technologies.
    - Improve the VSS/VDS logic.
    - Support the sub-Interface case with Duplicate IP in different VRFs.
    - Support the Palo Alto Firewall using L2 Zone to connect different VLANs.
- Remodel the Virtualization model to restrict the related interfaces to the Child device only.
- Provide more plugins to improve the troubleshooting ability.

7.2.2 One IP Table Improvement

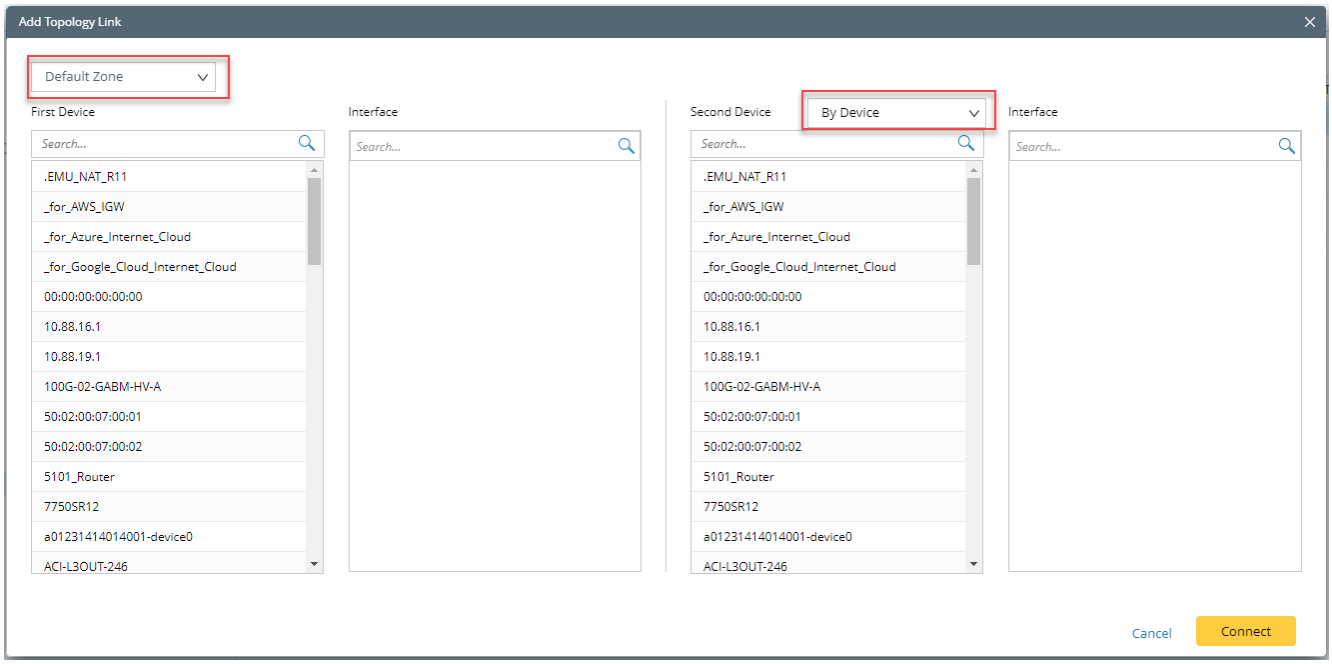
Certain IP/MAC entries in the One IP table do not have a corresponding switchport because of the NDP neighbor in the domain. Due to this, users cannot determine the connected switchport and hence cannot locate the IP/MAC in their network. To solve this issue, R11.1 adds two columns, **gateway** and **possible switch port** into One IP table.



IP Address	LAN Segment	MAC Address	Vendor	Switch Port	Possible Switch Port	VLAN ID	DNS Name	Gateway	Description	Data Sour
172.24.101.51	172.24.101.0/24	00b0.c275.8c00	Cisco Systems, Inc	BJ-L2-Core-A.FastEth...	BJ-L2-Core-A.FastEthernet0/15	10			Port_con_Juniper_"n...	ARP Ta
172.24.101.52	172.24.101.0/24	0010.0b0e.27ff	Cisco Systems, Inc	BJ-L2-Core-A.FastEth...	BJ-L2-Core-A.FastEthernet0/15	10			Port_con_Juniper_"n...	ARP Ta
172.24.101.61	172.24.101.0/24	001c.0fe2.55c6	Cisco Systems, Inc	BJ_L2_Core_6.FastEth...	BJ_L2_Core_6.FastEthernet0/15	10			connect to BJ-3750-1	ARP Ta
172.24.101.62	172.24.101.0/24	0023.0502.de41	Cisco Systems, Inc	BJ_L2_Core_6.FastEth...	BJ_L2_Core_6.FastEthernet0/15	10			connect to BJ-3750-1	ARP Ta
172.24.101.64	172.24.101.0/24	0cb6.d2f8.1d20	D-Link International	BJ_L2_Core_6.FastEth...	BJ_L2_Core_6.FastEthernet0/15	10			connect to BJ-3750-1	ARP Ta
172.24.101.65	172.24.101.0/24	d478.569e.2b40	Avaya Inc	BJ_L2_Core_6.FastEth...	BJ_L2_Core_6.FastEthernet0/15	10			connect to BJ-3750-1	ARP Ta
172.24.101.66	172.24.101.0/24	d478.569e.0e40	Avaya Inc	BJ_L2_Core_6.FastEth...	BJ_L2_Core_6.FastEthernet0/15	10			connect to BJ-3750-1	ARP Ta
172.24.101.67	172.24.101.0/24	001c.73ac.b5a6	Arista Networks	BJ_L2_Core_6.FastEth...	BJ_L2_Core_6.FastEthernet0/15	10			connect to BJ-3750-1	ARP Ta
172.24.101.68	172.24.101.0/24	001c.737a.2ea4	Arista Networks	BJ_L2_Core_6.FastEth...	BJ_L2_Core_6.FastEthernet0/15	10			connect to BJ-3750-1	ARP Ta
172.24.101.74	172.24.101.0/24	085b.0e5c.e895	Fortinet, Inc.	BJ_L2_Core_6.FastEth...	BJ_L2_Core_6.FastEthernet0/15	10			connect to BJ-3750-1	ARP Ta
172.24.101.229	172.24.101.0/24	0000.0c5c.d8c3	Cisco Systems, Inc	BJ_L2_Core_6.FastEth...	BJ_L2_Core_6.FastEthernet0/16	10			connect to Rack_Con...	ARP Ta

7.2.3 Remove Zone Selection from Add Topology Link

Each VLAN Group generates a zone in the Open Topology, causing many zones in the domain. R11.1 removed the zone selection and media option from **Add Topology Link** UI. Users can add point-to-point L3 and L2 links only.



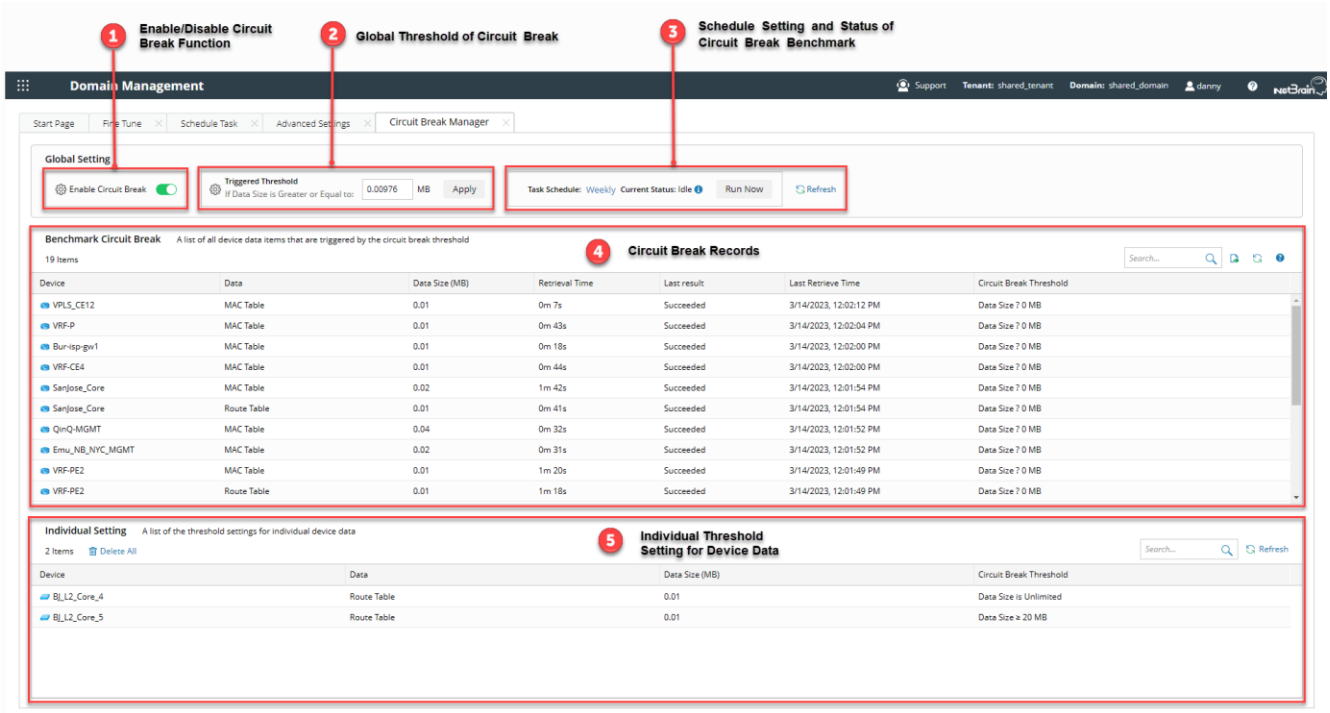
7.3 Circuit Break in Server Benchmark

Collecting a large amount of data may be time-consuming, which could cause Server Benchmarks to run for several hours without completion. This delay in updating critical data such as topology could affect the regular system operations. However, by implementing a Circuit Break, the initial Server Benchmark may take longer to complete as it detects and records the big data items. The Circuit Break Benchmark can then collect these items at a lower frequency, resulting in smoother and quicker subsequent Server Benchmarks.

The Domain admin can use Circuit Break to detect and record large data items in the regular Benchmark task and move them to the Circuit Break Benchmark automatically for low-frequency data collection, which will

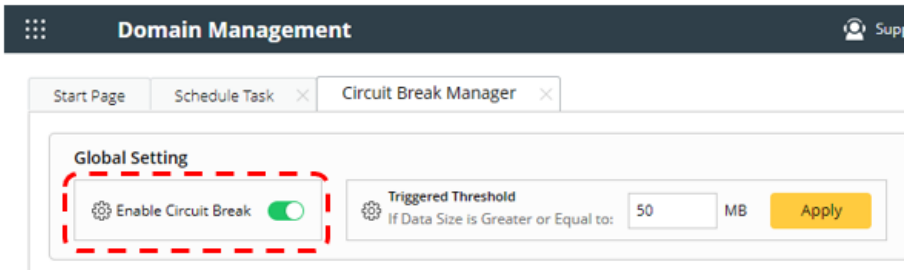
ensure the regular Benchmark operates at a faster pace. Take the following steps to configure the circuit break:

1. Enable/Disable the Circuit Break function globally.
2. Define the global threshold of Circuit Break checking.
3. Define the scheduling of the Circuit Break Benchmark and view the current status.
4. Review all device data items that are triggered by the Circuit Break threshold.
5. Review and update customized Circuit Break threshold settings for the individual device data item.

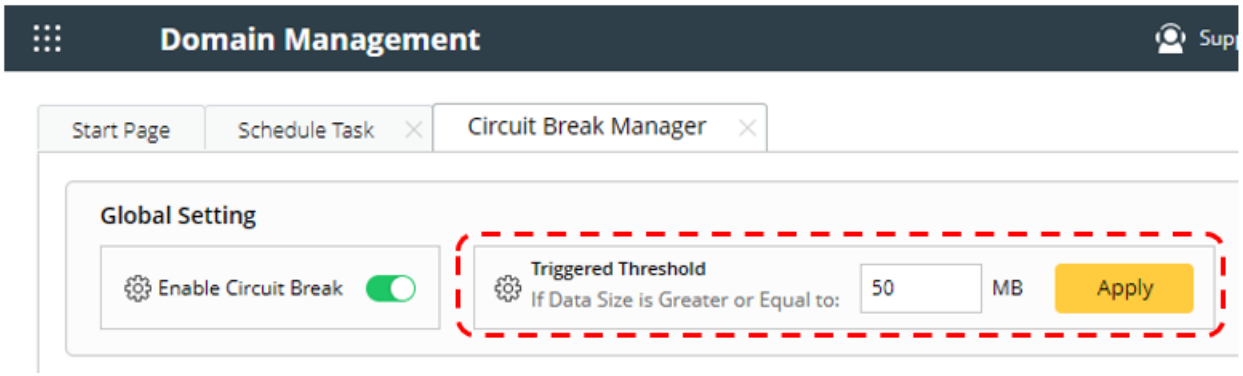


7.3.1 Circuit Break Global Settings

By default, the Circuit Break function is turned OFF, and it can be turned ON when there is a large amount of data in the user network, causing delays or inability to complete the Server Benchmark.

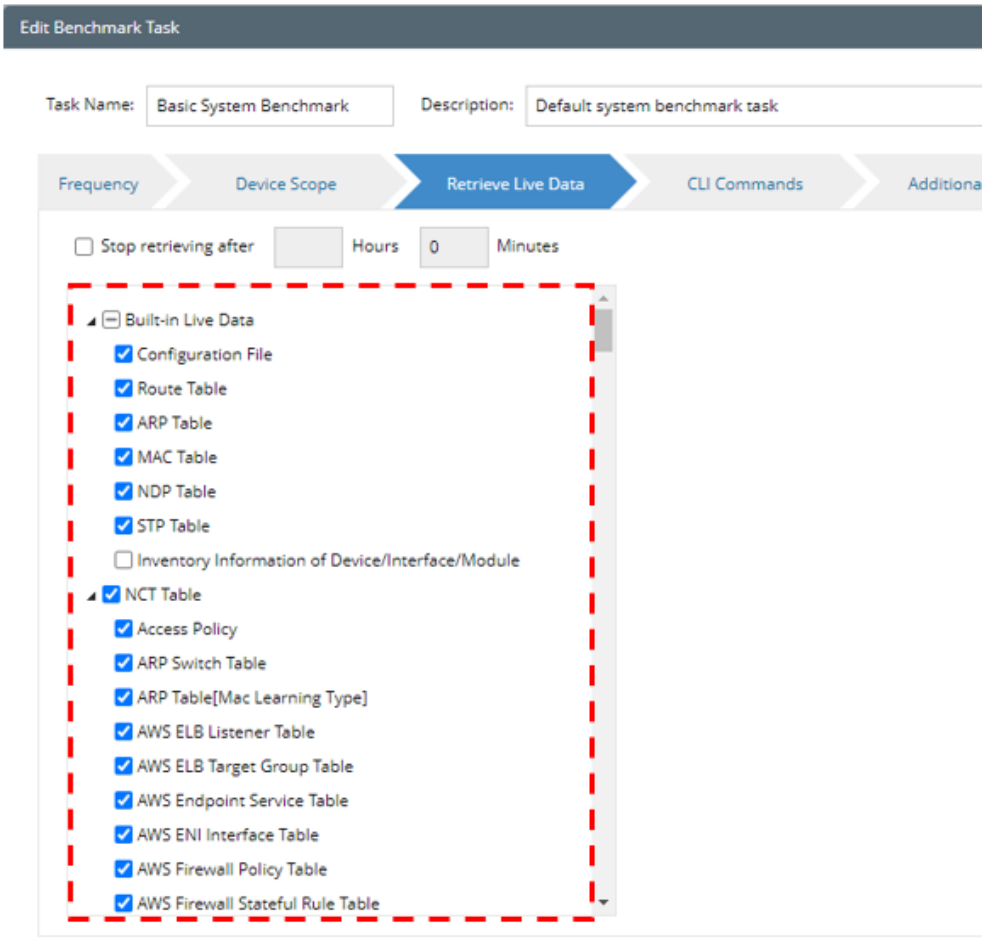


The Circuit Break uses the data size as the threshold for retrieving live data in the Server Benchmark. The default system-wide threshold is 50 MB.

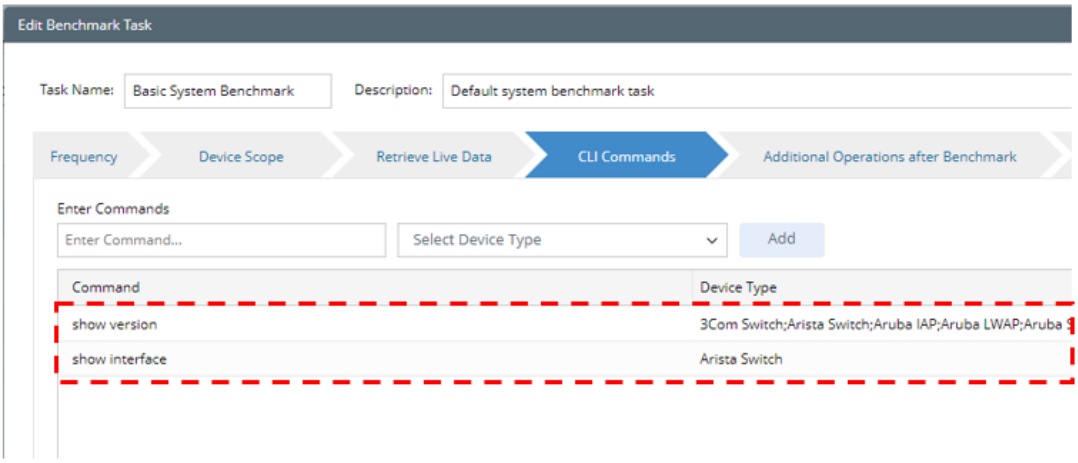


The Circuit Break will check the following data retrieved in the Server Benchmark:

- Each data item in the Retrieve Live Data section, including the System Tables and NCT Tables.



- Each CLI Command in the CLI Commands section.



The data size is the size of the raw data in each item listed above for an individual device. If a data item has sub-items, the data size is the total amount of all raw data of the main item and sub-items.

For example: If R1 has 5 VRFs configured, the Route Table of R1 will include one default routing table and 5 VRF routing tables. The data size of the Route Table for R1 is the total amount of the default routing table plus 5 VRF routing tables.

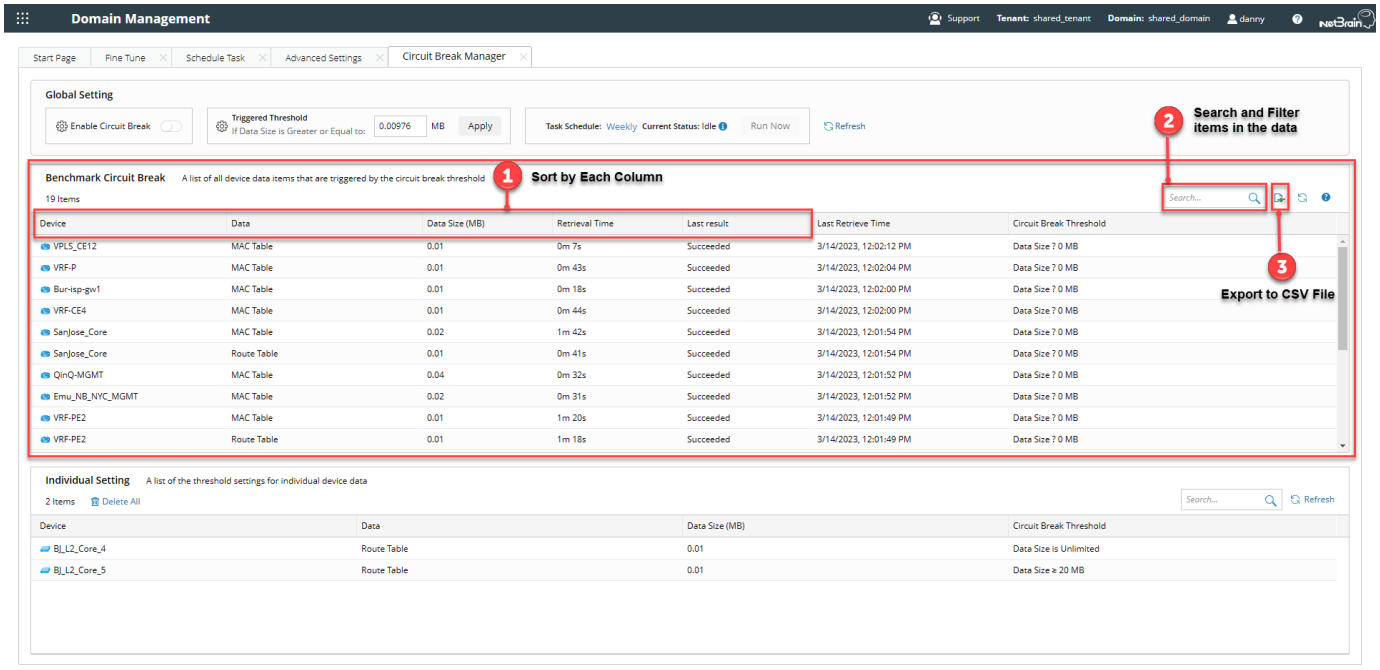
7.3.2 Execute Circuit Break Check in Server Benchmark

When executing Server Benchmark, two checks need to be performed for each data to be collected:

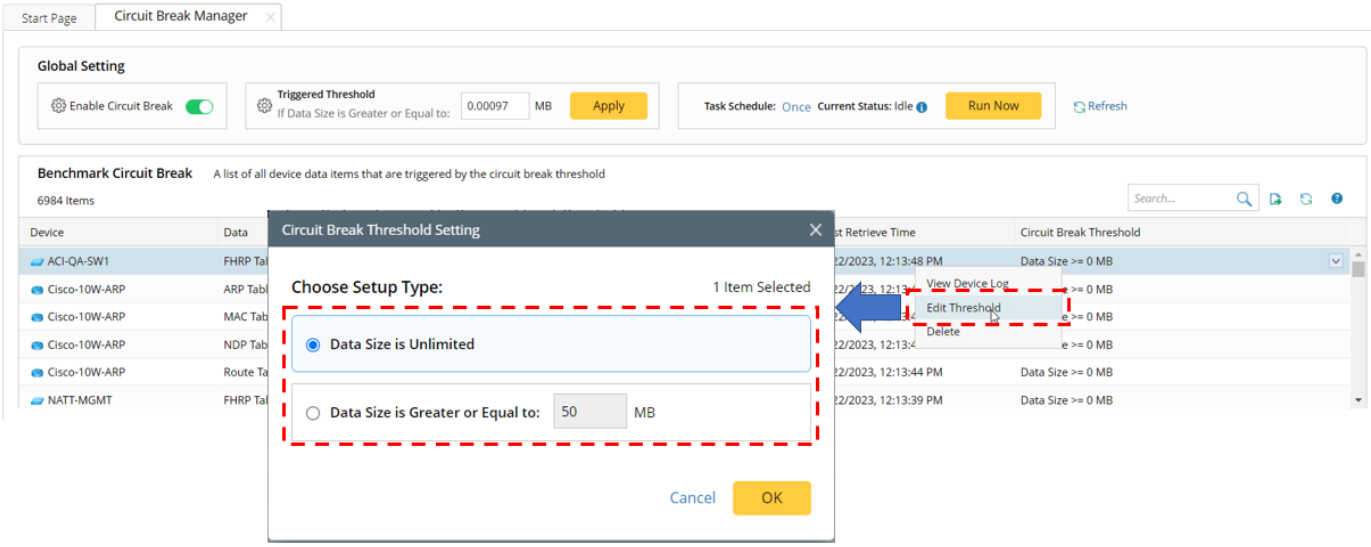
- Before collecting each piece of data, check whether it has been recorded in the Circuit Break Manager. If it has been recorded, do not collect it.
- After collecting each data, check whether it triggered the Circuit Break threshold. If triggered, record it in Circuit Break Manager.

7.3.3 Manage Circuit Break Data

All Device Data items that trigger Circuit Breaks in Server Benchmark will be recorded in the Circuit Break Manager.

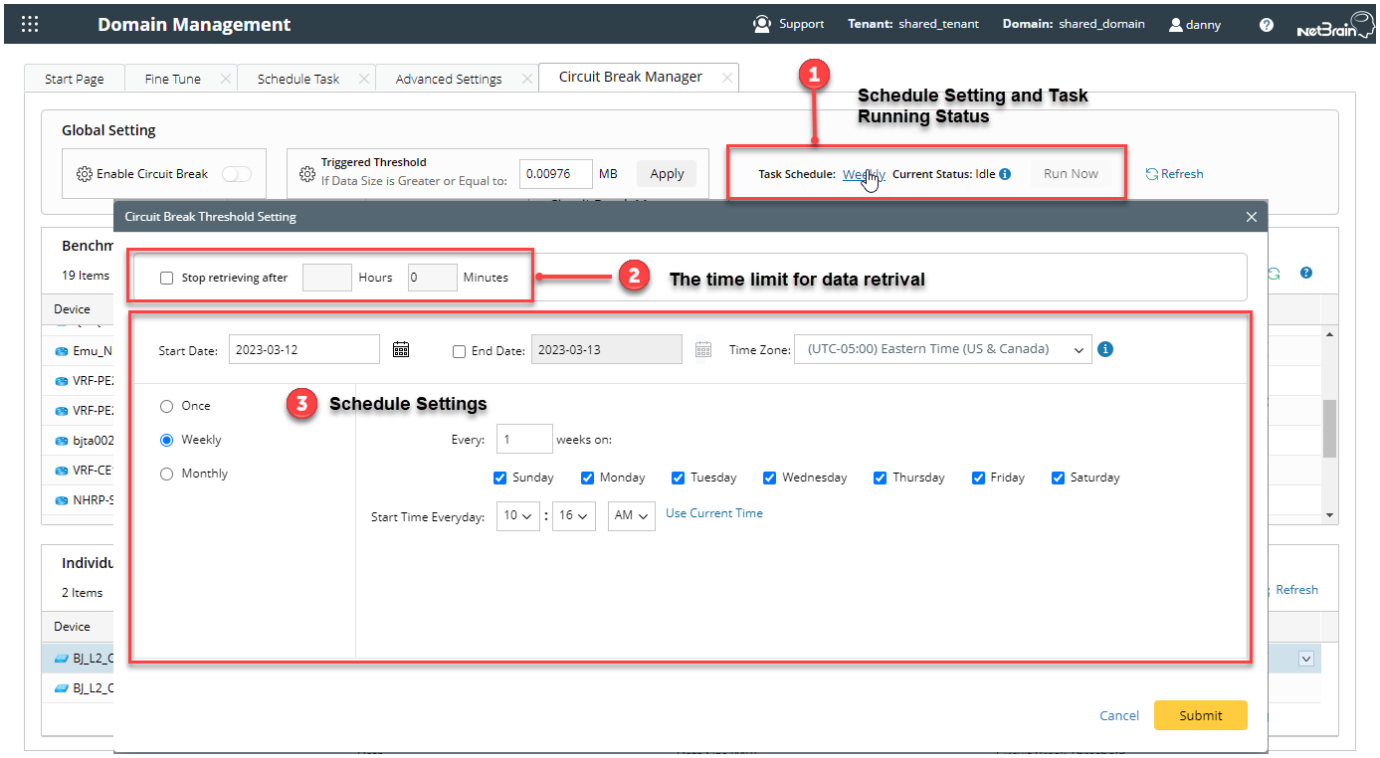


Users can set individual Device Data of the Circuit Break thresholds according to their actual situation. For example, if the data size for the routing table of the core router is larger than the default Circuit Break threshold, but the user still wants to get data in the Server Benchmark, it can be achieved by modifying the thresholds.



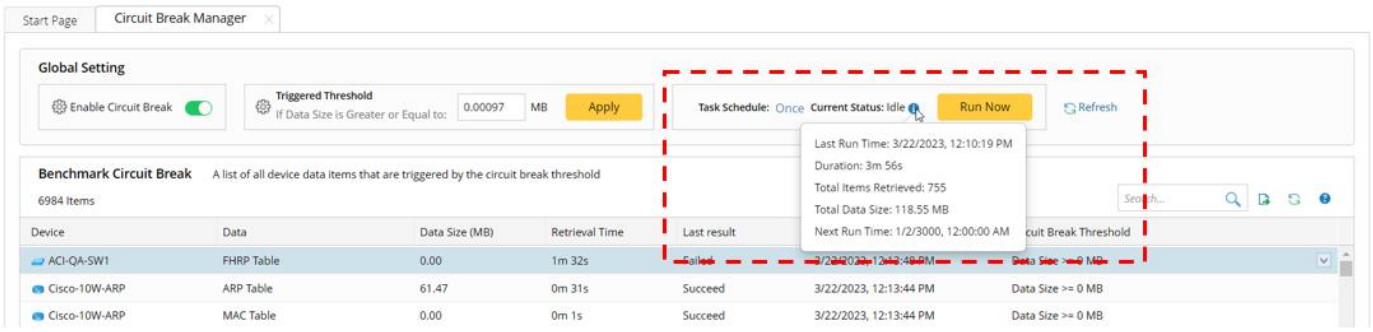
- Choose the “Data Size is Unlimited”, which means that this data item does not run Circuit Break checking, which is the default selection.
- The “Data Size is Greater or Equal to” can be chosen to assign a specific data size value to the data item.

7.3.4 Run Circuit Break Benchmark



1. When the Circuit Break function is enabled, the system will automatically create a Circuit Break Benchmark task, and all device data items in the Circuit Break table will be included in this task.
2. In some user networks, device access is allowed only within a specified time range. The “Stop retrieving after” option can define the time limit for data collection.
3. Using a lower frequency to collect the device data items in the Circuit Break Benchmark task is recommended. Circuit Break Benchmark supports run Once (manual execution), Weekly and Monthly.

Information summary can be viewed after the Circuit Break Benchmark is finished.



7.4 System-wide Intent Execution Workload Control

The system is expected to process the intent execution from different sources efficiently, and meanwhile, the stable operation of the system will not be affected by a large number of requests.

Moreover, the priority of intents and urgency to run intents differ. For example, an intent of compliance checks may be executed once a month, and their delay time can be long, such as 1 week, while the intents to help troubleshoot must be executed in a short time.

R11.1 introduces a new setting for intent: ***max_execution_delay,*** and the system utilization can also be controlled in this version:

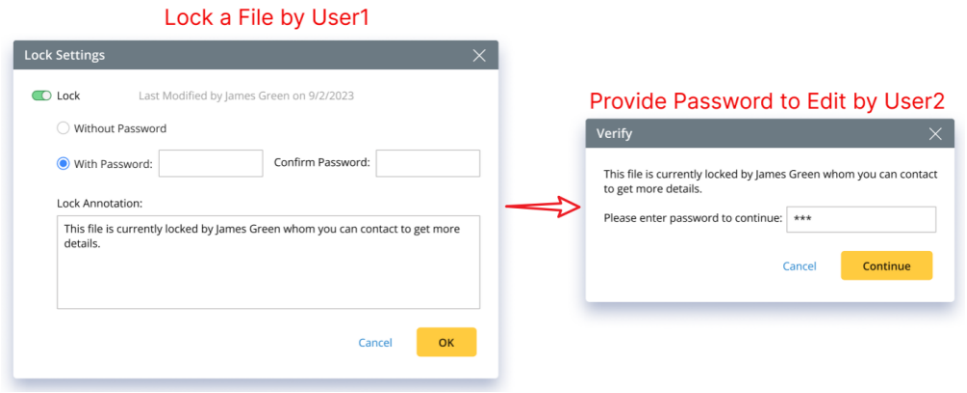
- Inside Intent, a field "***max_execution_delay***" has been added for the maximum time gap of a NI from submission to execution. By default, it is set as 5 minutes. When executing an intent, the service will check if the interval between the current time and the submission time is greater than the max execution delay time. If so, it means the intent was submitted too long ago, and the execution of the intent will be aborted.
- Before each NI is executed, it will be checked whether it needs to be executed: If the same NI has been executed before and there is a result, then the execution of the same NI within the max execution delay time will be skipped, and the result of the previous NI will be reused for the new execution request. If the system is still executing an NI and receives the same NI execution request, the execution of the

subsequent NI request will be skipped if the request time interval is within ***2 * max_execution_delay,*** and the result of the previous NI execution will be reused.

- The backend process can monitor the system utilization (server CPU, memory, etc.) in real-time. Requests will be processed only when the utilization is within the specified range (80%, configurable in the database), preventing the system from being busier by more requests.
- The intent requests have priority: all intent execution requests from PAF are submitted to the backend system for execution at LOW priority, while those from TAF are at HIGH priority (priority is divided into SUPPER, HIGH, and LOW).
- To protect the system from being overwhelmed by the API requests, users can set the API request threshold (5000 in the API request queue, configurable in the database by the system administrator) that the system can accept. If the accumulated API request exceeds the threshold, the API requests from the 3rd-party system will be discarded and cannot enter the system.
- Users can set the number of waiting tasks supported by TAF to prevent the impact of many tasks. By default, the number of waiting tasks is up to 30 in a worker server, and this value can be configured in the database.

7.5 Lock and Single Editing Control

The files of many key features (Network Intent/Map/Bot/ADT...) in NetBrain can be created by a user and used by many others. R11.1 improves the lock/unlock function with the editing rights control function in NetBrain.



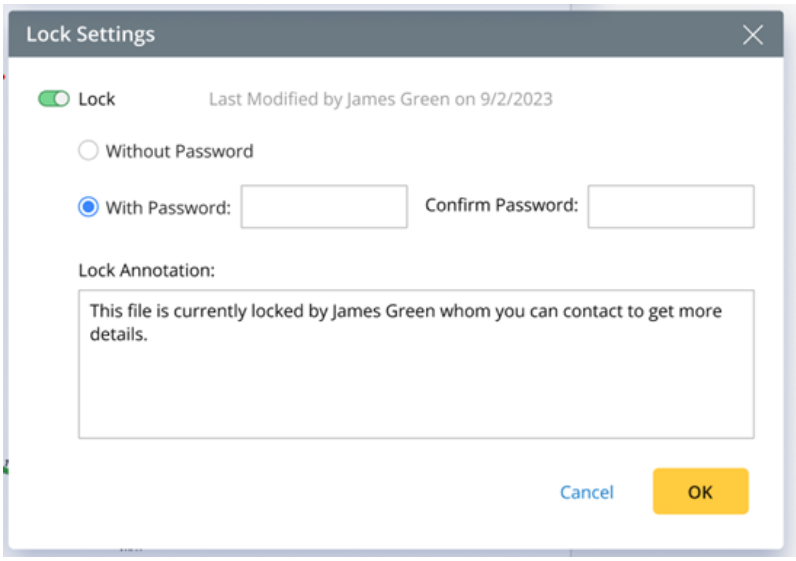
- Users with editing rights can lock a file with or without a password.
- Other users who want to edit the file must provide the password if needed.
- The creator and users with the “Domain Admin” role can freely edit a file without a password.

7.5.1 Lock Function

The Lock function protects users’ important files from being modified casually. Here are two typical use scenarios for applying this function:

- The creator of a file (such as intent and map) can lock it with a password to prevent other users from modifying it.
- For teamworking, the creator of a file locks it with a password and shares it with other team members. Users who are not team members can only view the file.

Users can lock files with or without a password and provide lock annotation to other users. If the file is locked with a password, other users can edit the file after entering the password.



- The information about who modified the lock settings and the modification time are displayed on the Lock Settings dialog.
- To define the lock settings, users must meet one of the following conditions:
  - A creator of the file.

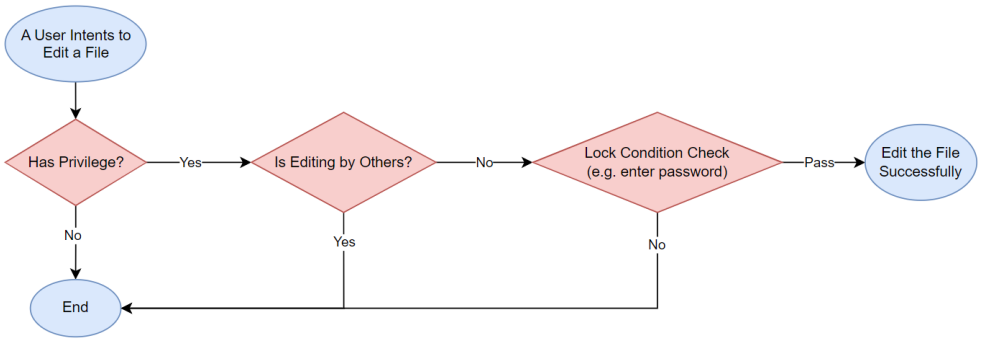
- A user (who is not the creator of the file) with Editing Rights, e.g., the "Shared Resource and File Management" privilege.
- A user with the "Domain Admin" role.

- The lock settings of the original file will not be kept in the new one after saving as or copying the file. However, the settings will be kept after the file is exported/imported.

7.5.2 Single-editing Control

Users in the same domain may simultaneously operate with the same file in NetBrain. If several users edit the file simultaneously, the file may be overwritten or cannot be saved. To address this issue, we introduce "Single-editing Control" to ensure that only one user can edit a file simultaneously.

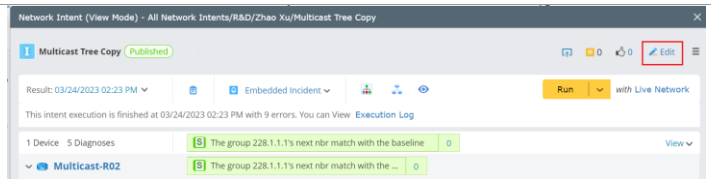
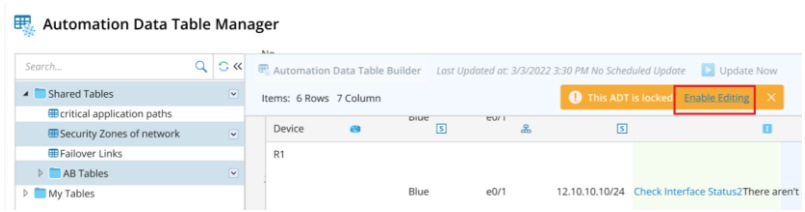
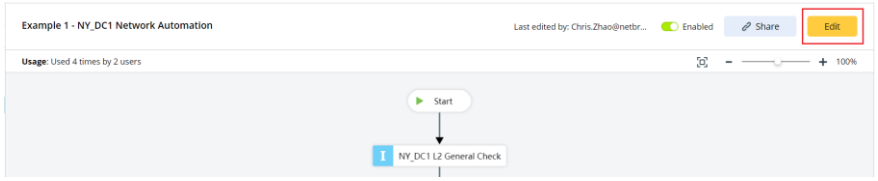
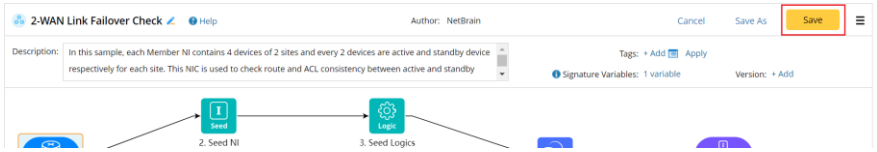
When a user intends to edit a file, the verification flow is shown in the diagram below:



7.5.3 Related Features

The above logic can be applied to Network Intent/Map/ADT/Bot/Network Intent Cluster in R11.1. However, there may be adjustments for every feature based on each one's specificity.

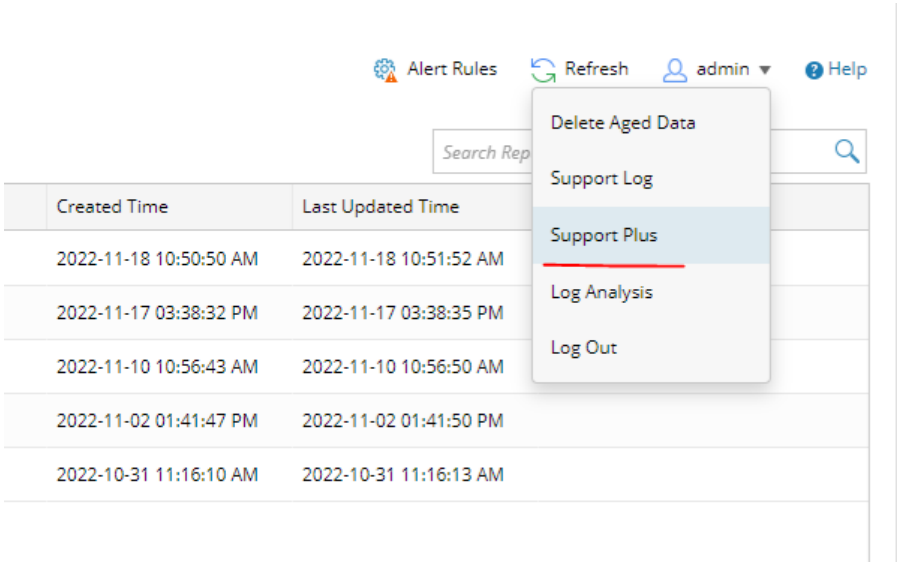
Feature	Lock Function	Single-editing Control
Network Intent	Editing Rights: "Shared Resource and File Management" privilege. When to Verify: users click "Edit", e.g.	The same as the basic concept.

		
Map	Editing Rights: Map Owner When to Verify: users click “Save”, e.g. 	Use the “Map Owner” control as before.
ADT (Automation Data Table)	Editing Rights: “Shared Resource and File Management” privilege. When to Verify: users click “Enable Editing”, e.g. 	This concept is not used in R11.1.
Bot	Editing Rights: “Shared Resource and File Management” privilege. When to Verify: users click “Edit”, e.g. 	The same as the basic concept.
Network Intent Cluster	Editing Rights: “Shared Resource and File Management” privilege. When to Verify: users click “Save”, e.g. 	The same as the basic concept. Check the condition when clicking “Save”.

7.6 Support Plus

Support Plus aims to check the health status of NetBrain IE systems online/offline so that system improvements can be made promptly according to the report.

In R11.1, Support Plus provides more metrics in the report, helping customers monitor and have better insights into the system's health and performance.



Number	System Health List	Items
1	Server Firewall & Port Information	Check if the firewall of each server is turned on and if the port of each component is open. • Whether the firewall is enabled • Whether the incoming ports of MongoDB, Elasticsearch, RabbitMQ, and Redis are open • Whether the outgoing port of the Front Server and the incoming port of the Front Server Controller are open • Whether the outgoing ports of SMTP and IMAP are open
2	External Network connectivity	Check the network connectivity required for specific needs: • Whether the email-sending port is connected • Whether Teams can be connected to the Internet • Whether ShareFile can be connected to the Internet

3	Multi-DC Configuration	• Whether Multi-DC is implemented • Whether Multi-DC is configured properly
4	Auto Data Clean Check	• List the time when the data clean was last executed. Users can determine if the data clean works properly from it. • List the data clean status of each data type, i.e., whether it's enabled • List the clean rule for each data type, i.e., when the old data should be deleted
5	Notification Setting	• Check if the System Email Setting has been enabled and defined. • Check if the Service Monitor Alert Setting has been defined. The email receiver and the alert rule must be defined.
6	Auto-update Information	• Check if the auto-update has been enabled. • Check the last result for auto-update

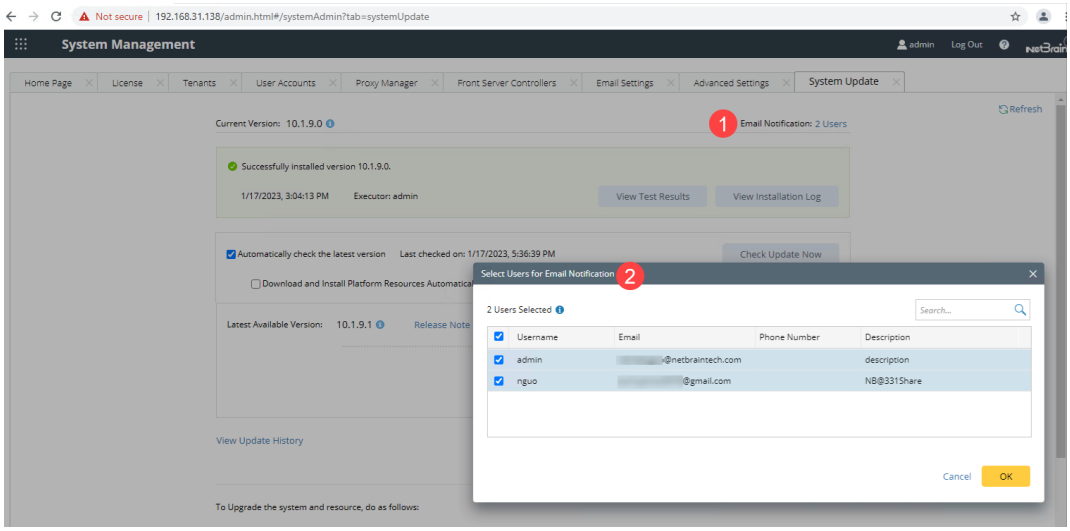
7	MongoDB	• MongoDB Metrics
		  ○ Check the status of the replica set
		  ○ Check the status and role(master/secondary/arbiter) of each node
		  ○ The number of currently open files
		  ○ If the Majority Read Concern has been enabled for the MongoDB cluster
		• MongoDB Data Size Check: The data size in each domain
		• MongoDB Slow Queries: Slow queries and their correlated metrics

8	RabbitMQ	• RabbitMQ Metrics: ○ Check the status of the RabbitMQ cluster

		• RabbitMQ Queue Check: Check if there is an abnormal queue in RabbitMQ and if consumers exist

7.7 Auto Update Improvements

In previous versions, all system admins received an email notification when a system update succeeded or failed, or when a new patch was available. In R11.1, a system admin can select who will receive these emails. Only the selected users can receive the notifications. Admins can make the selection at any time, and the selection will be applied when the next update is performed.



Running the pre-check before auto-update can verify the environment's readiness, preventing issues such as malfunctions of the service monitor.

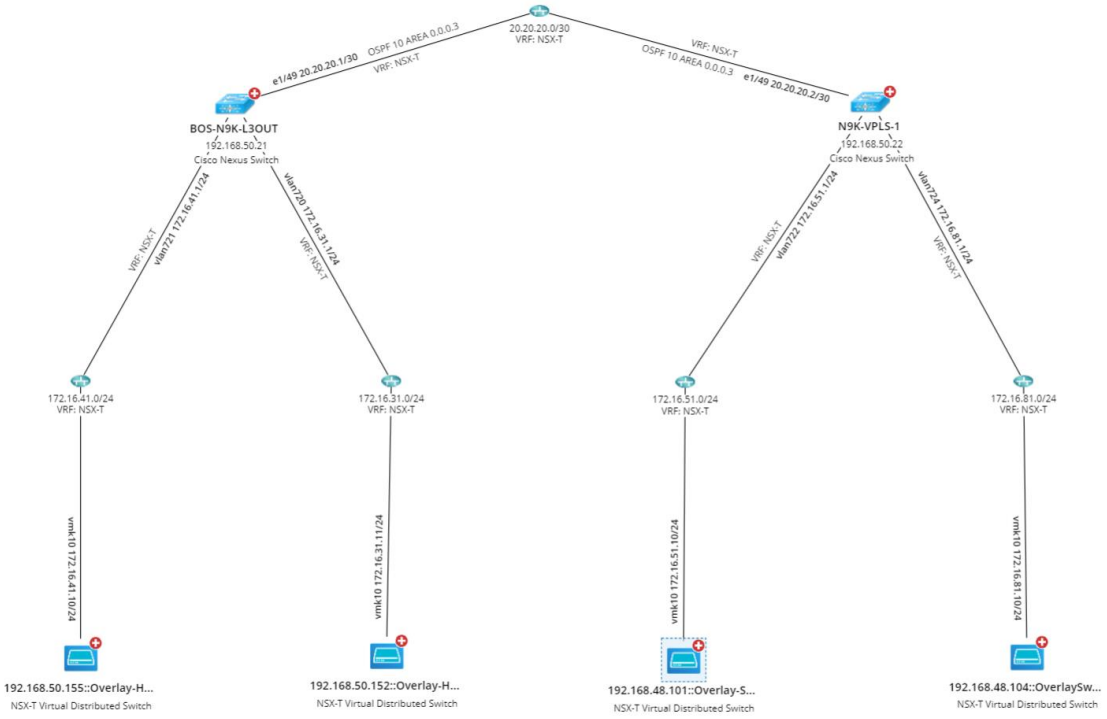
To prompt users to run the pre-check tool, a pop-up window will appear when they upload a patch and click **Schedule**. Clicking the **running the pre-check tool** link in the window will take users to a webpage with the user guide, which includes instructions for completing the system and environment check.

7.8 New Vendor Support

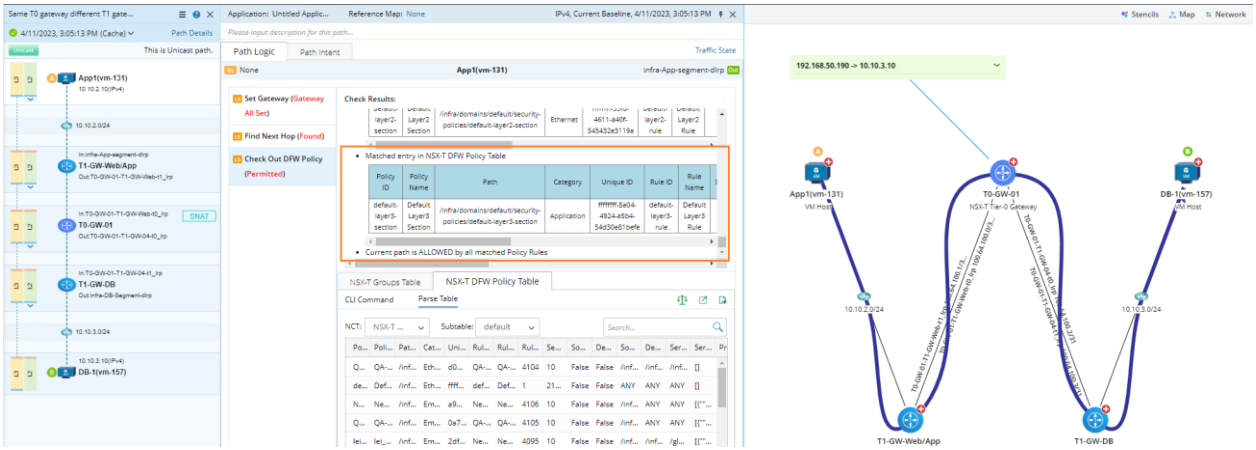
7.8.1 Support NSX-T

R11.1 supports NSX-T with the following functions:

- Discover NSX-T components such as T0/T1 Gateways, Segments, N-VDS Transport Nodes, Edge Nodes, etc. This enriched data helps understand better the network's configuration, dependencies, and potential bottlenecks.
- Create the map to visualize the NSX-T infrastructure, such as the topology of Logic Components like T0/T1 Gateways, Segments and Virtual Machines, which provides a comprehensive view of the network topology, enabling users to identify and troubleshoot potential issues in their NSX-T environment.



- Discover the AB path between the two endpoints inside or outside the NSX-T networks. The path calculation can also support checking DFW Policy, Gateway Policy, NAT and Load Balancer, which can help analyze the security, traffic flow or load balancing quickly and visually.



- View NSX-T operation status by applying data views driven by the API parser.

7.8.2 Other Device Type Support

Besides NSX-T, R11.1 also supports the following new device type:

- **Juniper MIST**: support the discovery of Juniper MIST AP and the L2 Wireless Topology.
- **Aruba Orchestrator** (Silver Peak, or Aruba SD-WAN): supports Aruba WAN Optimizer and Router. The system retrieves the configuration files, routing tables, APR tables, and NDP tables through the API.
- **Fortinet SD-WAN**: support the discovery of the ForiManager and Forinet ForiGate Firewall through API and retrieve the configuration files and routing tables through API.

7.9 Security Improvement

R11.1 resolved the following CVEs in R11.1 from 3rd Party Libraries (C/H/M/L is the priority, meaning Critical/High/Medium/Low):

CVE	Associated Library	Fixed version
CVE-2021-29921 **C** CVE-2022-37454 **C** CVE-2020-27619 **C** BZip2 in Python (CVE-2019-12900) **C** CVE-2015-20107 **H** CVE-2022-0391 **H** CVE-2021-28861 **H** CVE-2020-10735 **H** CVE-2022-42919 **H** CVE-2022-45061 **H** CVE-2021-3737 **H** Zlib in Python (CVE-2018-25032) **H** CVE-2021-3733 **M** CVE-2021-3426 **M** CVE-2021-23336 **M**	Python	3.10.9

CVE	Associated Library	Fixed version
CVE-2021-4189 M CVE-2013-0340 M		
CVE-2022-35737 H	SQLite3	3.40.0
CVE-2020-10543 H CVE-2020-10878 H CVE-2020-12723 H	Perl	5.32.1.1
CVE-2022-29117 H	Microsoft.Owin	4.2.2
CVE-2022-41881 H	Netty-Codec-Haproxy	4.1.90
CVE-2022-41854 M CVE-2022-38752 M	SnakeYAML	1.33
CVE-2022-38900 H	Decode-URI-Component	0.2.2
CVE-2022-24999 H	QueryString	6.11.0
CVE-2022-0536 M	Follow-Redirects	1.15.2
CVE-2022-21189 C	Dexie.js	3.2.2
CVE-2022-31160 M	JQuery UI	1.13.2
CVE-2021-32840 C CVE-2021-32842 M	SharpZipLib	1.3.3
CVE-2021-44732 C CVE-2021-43666 H CVE-2020-36475 H CVE-2020-36476 H	MbedTLS (used in Redis C++ client)	Removed from 10.1.9/R11.1 Replaced with () redis-plus-plus

CVE	Associated Library	Fixed version
CVE-2020-36478 **H** CVE-2020-22741 **H** CVE-2020-36426 **H** CVE-2020-36477 **M** CVE-2021-24119 **M** CVE-2020-10941 **M** CVE-2019-16910 **M**		

8 Cases Resolved in R11.1

The followings are customer cases resolved in R11.1 since the last major release of R11. Please note that this is a completed list.

- Case **00153154**: Fixed the issue that the dynamic search with an IP address is not working when proceeding with the site segregation operation due to an Elasticsearch synchronization problem.
- Case **00152949**: Support for manual gateway selection in path calculation.
- Case **00150847**: It takes a long time to log in if there are many domains.
- Case **00153653**: Benchmark task running for 24+ hours when updating site maps.
- Case **00151489**: Unable to run password change scripts on Cisco Routers and Switches due to lack of <send enter>.
- Case **00152785**: CLI access issue with HP ProCurve Switch caused by vt100 context handle logic error.
- Case **00153828**: Velocloud Path tracing is not working with SDWAN devices via Live Data source.
- Case **00153762**: Aruba WLC configuration file update failed due to an extra underline in the hostname.
- Case **00153684**: PAF not displaying/sending alerts.
- Case **00153578**: Search/Map for End Hosts not working properly.
- Case **00154011**: Incorrect switchport info on One-IP Table and wrong l2 connections to other switches.
- Case **00152299**: Site Visio maps size is 0 KB when exported using benchmark task.
- Case **00153524**: MPLS Cloud path failing - no route to next hop device.
- Case **00152920**: Benchmark takes a long time due to CLI threads reinitiating issue.
- Case **00153517**: Recurring decode not running without selecting update baseline.
- Case **00153256**: Cannot search some devices in Site search and global search if the config files are larger than 16MB.

- Case **00152886**: Show counters with the Include bar for only "0 0 0 0" cannot be displayed in the runbook.
- Case **00151857**: Incorrect L2 topology and One IP table entries based on PAN virtual Mac.
- Case **00146775**: Portal SSO Integration Support.
- Case **00150245**: L3 topology for the switch device is not built for the IP unnumbered interfaces.
- Case **00152343**: Unify search return data format by adding object-group and object network.
- Case **00151947**: Decommissioned devices are removed from Missed Devices table after System Benchmark.
- Case **00149717**: New API for Domain Metrics Plugin.
- Case **00152482**: Sites display problem when moving devices from Unsigned to leaf sites.
- Case **00152335**: Failed to update device info due to null PyFuncContext.
- Case **00151710**: Configuration comparison does not work properly if API config is selected for Firepower.
- Case **00151988**: Schedule Task Succeeded with warnings due to exceptional data in HP switch configurations.
- Case **00150356 & 00150449**: Error while exporting Site details in Excel File format.
- Case **00151491**: Integrating Audit and system monitor logs to Splunk.
- Case **00151991**: The help link is incorrect on the Missed Device page.
- Case **00151600**: Checkpoint NCT Policy Table not complete after live path.
- Case **00151420**: NI Global Data Table - Subtable showing empty.
- Case **00150970**: Remove "The site was rebuilt" messages received on ALL site maps.
- Case **00151563**: Delete old, cached files to reduce swap usage on the Worker server automatically.
- Case **00150778**: New plugin API to delete all Fine Tune-unknown ip.
- Case **00151110**: L2 topo build stuck during the benchmark.

- Case **00149980**: MongoDB high disk utilization due to data clean not working for some domains.
- Case **00150996**: Solve domain creating performance issue.
- Case **00150951**: L2 topology build takes a long time during the Benchmark.
- Case **00076382**: New feature to export the user list in NetBrain.
- Case **00150652**: Add support for new BGP peer template command to MPLS L3 VPN Cloud.
- Case **00150108**: Getting UnicodeDecodeError when tracing path using current baseline.
- Case **00147921**: Unable to generate domain health report with error message "The domain health report is being created by other users. Please try again later. "
- Case **00148718**: Support Ignore config block (multi-lines) for Comparison logic.
- Case **00150652**: Add support for new BGP peer template command for MPLS L3 VPN Cloud.
- Case **00148962**: The user can not choose the email subject identifier.
- Case **00150179**: IP from one IP table cannot be displayed on the map with the error message "TypeError: Cannot read properties of undefined (reading 'id')".
- Case **00150219**: Servers getting disconnected in the service monitor due to ES performance problems.
- Case **00149896**: SNMP access timed out for Cisco Meraki Firewall and Cisco ACI Leaf Switch, Spine, APIC in Validation.
- Case **00150400**: Path failure due to setting the wrong port mode after switchport command - Arista Switch.
- Case **00148570**: Device login session is not cleaned in front server - Juniper Router.
- Case **00148665**: Support a new format when building the cloud routing table and comparing as a path.
- Case **00150277**: Miss L2 topology from the Palo Alto Firewall to the Juniper EX SW.
- Case **00149376**: Inaccessible devices not being sent to the "Missed Devices" list.
- Case **00149138**: Velocloud Exception: in ngJson::Value::asCString(): requires string Value.

- Case **00149745**: Triggered Diagnosis events are long pending and are not getting through.
- Case **00149286**: Error while making API calls to Netbrain due to duplicate site name.
- Case **00149220**: MPLS path not working as expected due to destination IP load logic defect.
- Case **00149388**: Policy Table Comparisons are not working on Checkpoint firewalls.
- Case **00149507**: VRT in Business Cloud missing B Routes from CE device BGP Advertised Route.
- Case **00149714**: Azure route lookup should check the route AS-Path value to show the active path with a bold line.
- Case **00150041**: Change Management templates don't load due to multiple selection templates.
- Case **00148553**: Add more error logs for SNOW app integration.
- Case **00149469**: Portal login issue with access code with error message "The user has no permission to login".
- Case **00148694**: The Next hop device is not available in the routing table as the next hop IP is a duplicated IP on the HA device interface.
- Case **00148841**: The uploaded logo does not appear on the login page.
- Case **00149016**: Guidebook diagram fails to initialize.
- Case **00149340**: NCT Compare is not working on some Azure nodes due to duplicate data in the data engine.
- Case **00149419**: L2 topology building time is long, and the performance needs to be optimized.
- Case **00148659**: Cisco ACI Leaf Switches are wrongly shown in the Missed Devices after being rediscovered.
- Case **00148786**: More missing switch port entries are found in the One IP table due to the L2 logic changes since IEv10.
- Case **00148915**: Feature request–API for retrieving portal access key.
- Case **00149122**: Request to restrict API access to the parser if the user does not have private resource management permission.

- Case **00147645**: The device count shown in the Fine Tune result, and Inventory Report is different. A plugin is provided to delete duplicated devices.
- Case **00148655**: Duplicated VRF subtables are shown in the BGP Received Route table, but there is only one VRF subtable - Fortinet FortiGate Firewall.
- Case **00148735**: The Interface description in the subnet manager is not populated properly.
- Case **00148339**: NIC Failed to update Baseline with an error "An item with the same key has already been added."
- Case **00147510**: Support Office 365 SMTP/IMAP OAuth2.0.
- Case **00147590**: Webserver shut down with the error "Cannot read configuration file".
- Case **00147284**: Display N/A instead of retrieval failure since Interface Information is not defined in the End System driver.
- Case **00147917**: Fail to update Palo Alto configuration with empty hostname error due to large configuration file not saved in drive in time.
- Case **00146200**: Checkpoint Opsec manager is not working.
- Case **00147939**: Entries in One IP-table doubling during the build process.
- Case **00147709**: Management ip disappears in Tune live device of the silver peak.
- Case **00147678**: Ping result inconsistency in Device Log and Fine Tune Report.
- Case **00146542**: Cisco Meraki Firewall CLI Config Retrieval Failed.
- Case **00148399**: There is no topology for the Baracuda firewall due to failure to parse interface information.
- Case **00145332**: SNMP proxy forwarder support for Netbrain Front Server.
- Case **00147739**: Config parser does not show data for Cisco IOS XR device for IEv10.1.
- Case **00147366**: Benchmark crash in Build VLAN Group step.
- Case **00144319**: Unable to Delete Parser in the Shared Folder, error "Unable to perform the operation due to invalid Knowledge Cloud authentication".

- Case **00147123**: No L2 Topo for ACI leaf device's specific VPC port. The physical interface on the VPC port needs to be displayed on the ACI leaf device.
- Case **00147473**: Traceroute node throws an unknown error because Traceroute is not using the login script from the driver.
- Case **00147701**: Getting ACI L2 neighbor servers with BOND. The physical interface on the VPC port needs to be displayed on the ACI leaf device.
- Case **00133340**: Add export to Excel for Audit Log.
- Case **00144876**: [Public Cloud] Add Northbound API to Edit API Server Include/Exclude VNets.
- Case **00145923**: All Palo Alto firewall configs retrieve failed in benchmark and discovery.
- Case **00146264**: Cannot import Automation Library caused by the security policy.
- Case **00146624** Netbrain is not loading the config for the selected dates.
- Case **00146335**: Benchmark succeeded even though vCenter failed due to insufficient license.
- Case **00146657**: Issues with L3 mapping NIC.
- Case **00146729**: Last Accessed Time in Fine Tune wording issue.
- Case **00146771**: Subtask LiveRetrieveMultiDevice ended but failed with an error.
- Case **00146846**: Unable to create Device Health report with error "Failed to generate a new report".
- Case **00146928**: Support Full-Text Search.
- Case **00147153**: Unable to create Device Health report with error "Failed to generate a new report" due to multiple records isCurent=true in DB.
- Case **00147233:** Import ticket does not work.
- Case **00144967** Path failure for ACI endpoints with traceback error in next hop check due to too many subnets in one VLAN interface.
- Case **00146043**: Cannot adjust the L2 connections for the port channel link.

- Case **00146048**: Basic System benchmark warning "Subtask BuildVlanGroup ended, but failed with error..."
- Case **00146603:** Schedule discovery - Discovery over node license limit not saved.
- Case **00145635**: Device cannot be found during Dynamic Discovery for MPLS cloud.
- Case **00146157**: Interface parsing issues- Aryaka SD-WAN.
- Case **00145165**: Unable to pull any data(config, route) from Aryaka SDWAN devices via SNMP.
- Case **00145496**: Network Change stuck on one device when checking Pre-check Hostname before Execution.
- Case **00145854**: Unable to find Layer2 after benchmark due to an error when dealing with HRSP data.
- Case **00145352**: Unable to create Device Health report with error "Failed to generate a new report".
- Case **00145634**: Resolve Latest CLI Configuration Retrieval Failed on Palo Alto Firewall.
- Case **00144583**: L2 Path failing -Fortinet FortiGate Firewall.
- Case **00144445**: Benchmark error - Subtask BuildVlanGroup ended but failed with an error.
- Case **00143758**: Site Maps L2 & L3 connection is not populating post 10.1 version upgrade.
- Case **00143492**: Path is not moving forward to Viptela boxes.
- Case **00144180**: Ignored DAW alerts still making devices red on site maps.
- Case **00144384**: Getting Old Domain Health Report While Exporting.
- Case **00142952**: The map data was not completely exported.
- Case **00143818**: Dynamic Map L3 Topology-PortChannel Links Issue.
- Case **00144150**: Retrieve live Data won't run when we play runbook.
- Case **00143667**: Path not working after upgrade due to MPLS Cloud L3 Topo issue.
- Case **00143693**: Adding cloud objects to L2 site maps.
- Case **00143844**: Benchmark Timeout error due to too many NetScreen Firewalls having the same IP.

- Case **00142752**: Benchmark is slow because all the telnet sessions are taken and not released automatically.
- Case **00143017**: After upgrade, unable to collect logs on Dell EMC switches when the last line contains 'bel' char(0x07).
- Case **00143316**: Path gateways must always be selected, even for old paths.
- Case **00143450**: SSH connection failed when setting "version=1" in fix_livesetting.ini on Linux FS.
- Case **00143637**: When calculating a path, the gateway resolve is very slow due to too many interfaces in one device.