MATE Integration and Development Guide

Release 6.1
February 2015

Cisco Systems, Inc.
www.cisco.com

Cisco has more than 200 offices worldwide.
Addresses, phone numbers, and fax numbers
are listed on the Cisco website at
www.cisco.com/go/offices.

Text Part Number:

CONTENTS

Overview

This guide is for those who are integrating MATE products and doing related developmental work. Following is a summary of the chapters.

- Plan Files and Tables—Describes MATE plan file formats and how information is stored in tables.
- Importing Objects—Provides instructions on importing Layer 1 (L1) models, SRLGs, QoS models, demand groupings, and tags from other plan files.
- Importing Traffic and Growth Rates—Identifies how to import traffic and growth values into plan files, and describes the tool that enables you to create representative plan files by aggregating a series of plans over a time period.
- Exporting Routes and Tables—Provides instructions on exporting the routes of demands, IGP shortest paths, LSPs, LSP paths, circuits, and L1 circuits. It also describes how to export LSP explicit path settings, export L1 circuit Lambda Sim values, and how to export the current table showing in the MATE GUI.
- Importing Offline Discovery—Describes the MATE GUI tools for importing router configuration files, IGP databases, and SAM server information into plan files.
- Add-Ons and GUI Customizations—Provides instructions on creating add-on applications that can be accessed from the MATE GUI, as well as other means of customizing the GUI.
- Reporting Tools—Describes tools for generating MATE reports using JasperReport templates and tools for managing reports integrated into plan files.
- Command-Line Interface—Describes the file for setting default options used by calls to the tools through the CLI or through the GUI, as well as options that control the level of CLI logging.

Refer to the *Table Schema and CLI Reference* for a complete list of plan file tables and their most commonly used columns in plan tables and for a complete list of CLI tools and their Help output.

Note This guide references `$CARIDEN_HOME`, which is the directory in which the Collector server and MATE software is installed. The Linux default `$CARIDEN_HOME` is `/opt/cariden/software/mate/current`.

Integration with Other Systems and Workflows

The information in a MATE plan is readily available for integration into other systems or workflows. All plan information is saved in tables, which you can retrieve, filter, and export to ordinary .txt format plan files. The easiest way to become familiar with the tables in a plan file is from the GUI, using the Plan

Table Database Editor. This interface displays a list of all the tables in a plan and provides options for extracting plan information, from simple text searches to complex SQLite queries. The data you extract can be useful for critical tasks, such as these.

- Create reports about utilization and routing for a single simulation scenario or multi-scenario analysis.
- Extract path for node pairs, specific demands, or LSPs for analysis by third-party applications.

If you have a custom MATE task that is part of your workflow, you can add that task as a menu item to MATE, using the Add-on capability. Add-ons can simplify the task of data extraction for other systems, as well as for any MATE task that you perform frequently.

Related Topics

- *Table Schema and CLI Reference*
- `$CARIDEN_HOME/docs/table_schema`
- `$CARIDEN_HOME/bin`
- *MATE GUI Visualization Guide*
- *MATE Design User Guide*
- *MATE Live User Guide*
- *MATE Design Archive User and Administration Guide*

Plan Files and Tables

This chapter provides information about the tables used in MATE plan files, in the following sections.

- Plan Files
- Plan Tables
- Working With Tables
- External Tables

Plan Files

The unit of data storage in MATE is the *plan file*. The plan file, or plan, contains the various components of information that MATE requires to display and operate on a network.

MATE plans have the following file formats.

- .pln format plan—Complete plan information in the native MATE format.
- Complete .txt format plan—Complete plan information in a .txt format plan file.
- Simple .txt format plan—Simplified plan information in a .txt format plan file.

The GUI and CLI tools use these formats interchangeably for input and output files. The default format for the GUI is the .pln format file.

.pln Format Plan

The .pln format is the native MATE format for plan files. This format is small in size, so opening and saving the file is fast. When scripting, you generally work with some combination of .pln and .txt format plans and tables. For a description of the tables in the .pln format file, refer to the External Tables section.

.txt Format Plan

Each .txt format plan contains plan information in plain text that you can view and edit in an Excel spreadsheet or text editor. Excel is particularly useful because most of the data has tab-delimited fields that Excel can convert to a spreadsheet.

Each file contains a collection of tables, including the required <Network> table that specifies the MATE version that created the plan and other high-level plan properties. Each table is preceded by a table heading, such as and <Nodes> and <Sites>. Rows starting with a pound sign (#) are ignored and treated as comments.

There are two types of .txt format plan files.

- Complete—Contains all of the schema of tables and columns described in the External Tables section. Scripts written on complete .txt format plan files do not have to verify columns and tables are present before operating on them.
- Simple—Contains a subset of the complete format, eliminating the following unnecessary tables and table columns.
  - All empty tables.

    Example: <LSPs> table with no entries.
  - All columns that contain only default values.

    Example: Empty Protected values in the <Nodes> table default to F. So if all values in the column are F, it is not included in the .txt format plan.

    This format is smaller and easier to edit manually. However, it can be more difficult for a script to parse because the script must check for missing tables and columns. We do not recommend this format when using scripts.

Sample Simple .txt Format Plan File

To view sample tables in a simple .txt format plan, open the `samples/euro4.txt` file in Excel, or any text file editor. Following are a few `euro4.txt` excerpts for some of the most important tables in the plan.

Table 2-1 <Network>

Property	Value
Title	
Version	5.3

Note If the Version row is missing, the version property defaults to 4.1, not the current version, for backward compatibility. Plans in Version 4.1 did not require an explicit version.

Table 2-2 <Nodes> Excerpt

Name	Site	Function	Protected
cr1.ams	AMS	core	F
er1.fra	FRA	edge	F

Table 2-3 <Sites> Excerpt

Name	LocationCode	Longitude	Latitude
AMS	AMS	4.78	52.32
FRA	FRA	8.57	50.03

Table 2-4 <Interfaces>Excerpt

Node	Interface	IGPMetric
cr1.ams	{to_cr2.ams}	10
er1.fra	{to_er1.lon}	100

Table 2-5 <Circuits>Excerpt

Name	NodeA	InterfaceA	NodeB	InterfaceB	Capacity
PAR-LON	cr1.lon	{to_cr1.par}	cr1.par	{to_cr1.lon}	622
AMS_INTRA	cr1.ams	{to_cr2.ams}	cr2.ams	{to_cr1.ams}	2488
AMS_EDGE1	cr1.ams	{to_er1.ams}	er1.ams	{to_cr1.ams}	

Table 2-6 <Demands> Excerpt

Name	Source	Destination	ServiceClass
er1.lon-er1.ams	er1.lon	er1.ams	Internet
er1.lon-er1.ams	er1.lon	er1.ams	Voice

Table 2-7 <DemandTraffic> Excerpt

Name	Source	Destination	ServiceClass	TrafficLevel	Traffic
er1.lon-er1.ams	er1.lon	er1.ams	Internet	morning	200
er1.lon-er1.ams	er1.lon	er1.ams	Voice	evening	138

Convert Between Plan File Formats

The GUI and CLI use the .pln and .txt formats interchangeably. When you save a file from the GUI, the .pln format is the default. To save to a different format, select the File->Save As menu.

From the CLI, the format of a saved file depends on the extension of the output filename. A .pln extension produces a .pln format file and a .txt extension produces a .txt format file. A command-line argument (`-simple-txt-out-file`) specifies the simple .txt format plan.

To convert one format to another, use the `mate_convert` CLI tool. This tool can convert formats within the current software version, or from an earlier version to the current version. For example, `mate_convert` can upgrade a 5.6 plan file to the 6.0 file format.

Plan Tables

MATE defines all aspects of a plan using a collection of tables. Developers can access and modify all tables using MATE command-line tools. For a complete online listing of tables and their columns, refer to the `table_schema.html` file in the `$CARIDEN_HOME/docs` directory. You can also refer to the *Table Schema and CLI Reference*, which lists the most commonly used columns of each of these plan tables.

Plan Table Columns

The columns in a plan file table contain data in one of the following categories.

- **Key columns**—Columns that uniquely identify the rows of the table. Each table has one or more key columns. For example, the <Nodes> table has one key column, Name, which is the unique name of the node. The <Interfaces> table has two key columns: Node and Interface. This pair must (jointly) be unique for all entries in the table. Another example is the <Demands> table, which has key columns: Name, Source, Destination, and ServiceClass. If there are two demands from the same source to the same destination with the same service class, they must have different names.

  In the `table_schema.html` file, key columns are highlighted in orange.
- **Plan columns**—Columns that define or configure properties of entries in a table. For example, in a <Nodes> table, the Site column specifies the site that contains the node, and is therefore a plan column. Key columns are always plan columns.

  In the `table_schema.html` file, plan columns are highlighted in blue.
- **Derived columns**—Columns that provide information that is derived from the plan columns in the same table or a different table. These are not stored in plan files, but are generated by the GUI when the tables are displayed, or by `table_extract` when the tables are extracted from the plan file. For example, Remote Node in the <Interfaces> table is derived by looking up the remote node for the interface as defined in the <Circuits> table. Some derived information can be more complex to obtain. For example, the Traff Sim column is a derived column that is the result of a simulation performed on the network.

  The entries in tables generated in the GUI and from `table_extract` can depend on some pre-specified parameters. For example, the <Interfaces> table Traff Meas column is the measured traffic on that interface for a specified traffic level. For a particular QoS selection the column can be overall (*undifferentiated*) traffic, traffic on a particular queue, or traffic for a particular service class.

  In the `table_schema.html` file, derived columns are highlighted in gray.

Table Objects

Objects in MATE are represented by rows in tables, and object properties are represented by column entries in that table, or by entries in tables of related objects. For example, LSP objects are defined in the LSPs table. Columns in this table, such as Setup BW, define properties of each LSP. The paths of each LSP are also properties of the LSP, but those LSP paths are defined as objects in separate LSP Paths, Named Paths, and Named Path Hops tables.

Table 2-8 lists the notation that MATE uses to specify a plan object when the type of object is not known from the context. Except for the IP address, these notations have a one-to-one mapping with key columns for each object.

Table 2-8 Single Object Notation

Object	Format
AS	AS{ASN}

Circuit	ct{NodeA | InterfaceA | NodeB | InterfaceB}
	Example: ct{atl | POS1/1/1 | sjc | POS1/10}

External endpoint	EP{Endpoint} **Example with a member:** EP{100}:cr1.chi

Object	Format
Interface	if{Node | Interface}
IP address	ip{ipaddress} Can reference multiple objects of the same or different type. MATE first attempts to find a node with this loopback IP. If is not found, MATE chooses another interface. If there are multiple matches, MATE uses the first one it finds.
Layer 1 circuit	l1ct{L1 Node A | L1 Node B | Name}
Layer 1 circuit path	l1ctp{L1 Node A | L1 Node B | L1 Circuit | Path Option}
Layer 1 link	l1lnk{L1NodeA | L1NodeB | Name}
Layer 1 node	l1nd{Name}
Node	nd{Name}
Port	pt{Node | Port}
Port circuit	pct{NodeA | PortA | NodeB | PortB}
Site	st{Name}
SRLG	srlg{Name}

Table Schema and Plan Tables

In a text plan file, each table starts with <TableName> to identify the name of the table, for example, <Nodes>.

The first row of the table body contains the column headings, followed by rows that describe the table entries. The order of the columns is irrelevant, and only the key columns must be present.

Each plan table is defined using an excerpt from the database schema, the part that defines that table. For example, Table 2-9 lists a table schema excerpt for the <NamedPaths> table. Table 2-10 shows an example of a <NamedPaths> table that has been populated within a plan file.

In Table 2-10, the first column is Name, which is described in the first row of Table 2-9. In this case, the Name of the named path (PathA or PathB) is the same in the plan file and GUI, it's the name of the path, has a data type of text, and is a table key. Being a key means the Name column is among the columns that uniquely define a row. In this case, the Name and Source together define a unique row.

Table 2-9 NamedPaths Table Schema

Plan File Name	User Interface Name	Description	Data Type	Category
Name	Name	Name of the path	text	key
Source	Source	Name of the source of the NamedPath	text	key
Active	Active	Is the path active?	boolean	plan
Resolved	Resolved	T if all hops in path are resolved in plan, F if not, na if no hops.	boolean	derived

Table 2-10 <NamedPaths> Example

Name	Source	Active
PathA	Router1.Vostock	T
PathB	Router1.McMurdo	T

User-Defined Tables and Columns

The table structure within a plan file is extensible with namespaces. You can add new tables of your own design and new columns to the standard tables. You identify new tables or columns by adding a namespace prefix to the table or column name. MATE does not use these tables for any feature, and does not check them for errors. The tables are preserved on import and export, however, and their contents are displayed in the Plan Window under the Plan Tables hierarchy.

User-defined tables help reduce the number of extra files required in a MATE solution, and are helpful in custom scripts that need to carry information from one step to another.

The format of the namespace prefix is as follows.

<Namespace1::Namespace2:: ... ::TableName>

The following example shows a new table that defines rack space for nodes. The table title is prefixed with MyTables::, which identifies it as a user-defined table in the MyTables namespace.

Table 2-11 Example: User-Defined <MyTables::Rackspace> Table

Node	Cabinet	RUs
Node1	123	2
Node2	149	1

The next example shows how to add a user-defined column to a standard <Sites> table. The new column is prefixed with Customer::, which identifies it as a user-defined column in the Customer namespace.

Table 2-12 Example: User-Defined Column

Name	Customer::Type
Site1	Datacenter
Site2	Customer

Working With Tables

This section describes how to modify plan tables from the CLI or GUI, or how to use table files as arguments for command-line tools.

Edit Plan File Tables Using CLI Tools

This section explains how to add, modify, or delete tables in a plan file. Basically, you extract the desired table, edit the extracted file, then replace the table in the plan. Changes to columns of type Derived are ignored when replacing a table. Only columns of type Plan are relevant for changes.

The replacement tool does not validate the changed table, so the resulting plan could be incomplete, inconsistent, or have erroneous data. Such problems are flagged the next time the plan is opened in MATE.

See also the Plan Tables and SQL Queries in MATE sections.

Add Table in a Plan

Step 1 Create the desired table as a text format file, or extract one from another plan file. If you create a table manually, the table name and columns must confirm to the table definitions in this chapter.

Step 2 Add the table to the plan using `table_replace`. This overwrites an existing table if one exists.

Change Table in a Plan

Step 1 Extract the table from the plan using `table_extract` and save it to a .txt format file.

Step 2 Change the data in the table using in one of the following tools.

- Apply changes with a simple SQL statement using `table_edit`.
- Apply changes with a complex SQL statement using `mate_sql`.
- Apply changes manually using Excel or a text editor.

Step 3 Replace the table in a plan file with the modified table, using `table_replace`.

You normally replace the table in the original plan file, but the result of `table_replace` can be a different file or even a different plan format. For example, you could replace a table in a text format plan and save the result as a binary format file (.pln), leaving the original file unchanged.

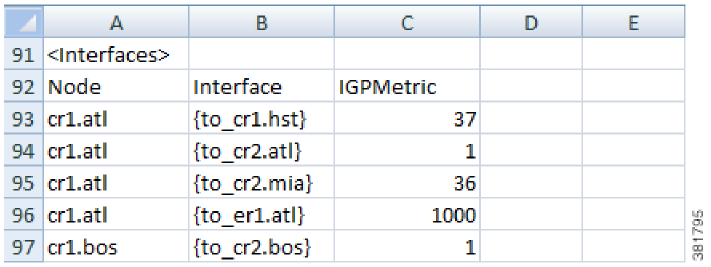
Delete Table from a Plan

Delete the table from a plan using `table_delete`.

Example of a Table Modification Using CLI Tools

This example shows how to increase the IGP Metric for interfaces on core routers in the `us-wan` plan, which is in the `$CARIDEN_HOME/samples` directory. Figure 2-1 shows an excerpt from the original plan file. The Interfaces table has three columns: Node, Interface, and IGPMetric, which are all either “key” or “plan” columns. You only need the “key” values to uniquely define an interface.

Figure 2-1 Interfaces Table in us-wan.txt



	A	B	C	D	E
91	<Interfaces>				
92	Node	Interface	IGPMetric		
93	cr1.atl	{to_cr1.hst}	37		
94	cr1.atl	{to_cr2.atl}	1		
95	cr1.atl	{to_cr2.mia}	36		
96	cr1.atl	{to_er1.atl}	1000		
97	cr1.bos	{to_cr2.bos}	1		

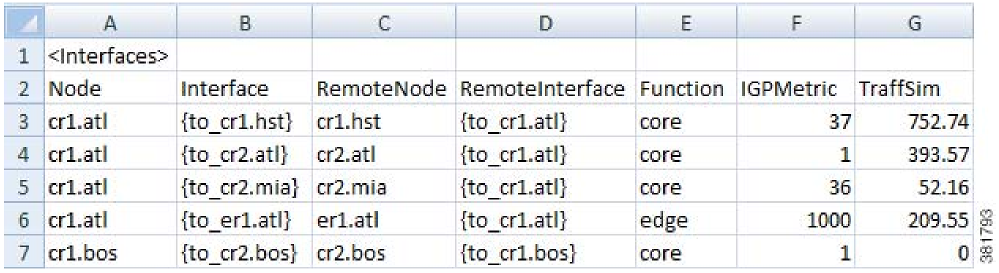
381795

In this simple example, a user could manually update the IGPMetric column, using Excel or another text editor. To script the process however, use the CLI tools.

Step 1 First, extract the Interfaces table from the plan file to a temporary file, `if-table.txt`:

```
table_extract -plan-file us_wan.txt -out-file if-table.txt -tables Interfaces
```

Figure 2-2 Extracted Interfaces Table



	A	B	C	D	E	F	G
1	<Interfaces>						
2	Node	Interface	RemoteNode	RemoteInterface	Function	IGPMetric	TraffSim
3	cr1.atl	{to_cr1.hst}	cr1.hst	{to_cr1.atl}	core	37	752.74
4	cr1.atl	{to_cr2.atl}	cr2.atl	{to_cr1.atl}	core	1	393.57
5	cr1.atl	{to_cr2.mia}	cr2.mia	{to_cr1.atl}	core	36	52.16
6	cr1.atl	{to_er1.atl}	er1.atl	{to_cr1.atl}	edge	1000	209.55
7	cr1.bos	{to_cr2.bos}	cr2.bos	{to_cr1.bos}	core	1	0

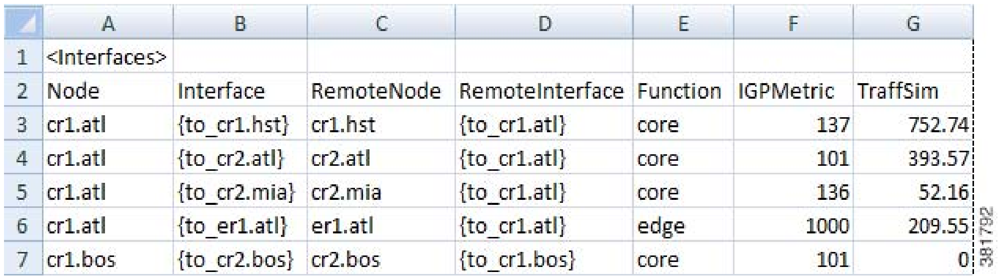
381793

Figure 2-2 shows an excerpt from the extracted Interfaces table. The interesting part of the extracted table is that it contains more than just "key" and "plan" columns, it also has "derived" columns, which MATE creates when it reads a plan file or performs simulations. These extra columns make it is easier to identify the Core routers in this example because there is now a Function column.

Step 2 The next step is to increase the IGP Metric for core router interfaces. The command below reads the extracted Interfaces file, `if-table.txt`, finds the IGPMetric column of the Interfaces table, filters the rows to those with 'core' in the Function column, adds 100 to the IGPMetric in the filtered rows, and saves the result to a new file named `if-table-edited.txt`.

```
table_edit -plan-file if-table.txt -out-file if-table-edited.txt -table Interfaces -column
IGPMetric -rowfilter "Function = 'core'" -value IGPMetric+100
```

Figure 2-3 Interfaces Table After Replacement



	A	B	C	D	E	F	G
1	<Interfaces>						
2	Node	Interface	RemoteNode	RemoteInterface	Function	IGPMetric	TraffSim
3	cr1.atl	{to_cr1.hst}	cr1.hst	{to_cr1.atl}	core	137	752.74
4	cr1.atl	{to_cr2.atl}	cr2.atl	{to_cr1.atl}	core	101	393.57
5	cr1.atl	{to_cr2.mia}	cr2.mia	{to_cr1.atl}	core	136	52.16
6	cr1.atl	{to_er1.atl}	er1.atl	{to_cr1.atl}	edge	1000	209.55
7	cr1.bos	{to_cr2.bos}	cr2.bos	{to_cr1.bos}	core	101	0

381792

Figure 2-3 shows an excerpt from the updated file, `if-table-edited.txt`, which has the expected result in the IGPMetric column: the four core routers (nodes) shown have their IGP metrics increased by 100, and the edge router metric is unchanged.

Step 3 The last step is to update the original plan file, or to create a new one. In this case, the example command creates a new plan file, one that uses the .pln format. Changing the file format in this example just shows the flexibility of the tools.

```
table_replace -table-file us_wan.txt -replace-table-file if-table-edited.txt -out-file
us_wan.pln
```

Step 4 As a confirmation, you could open the `us_wan.pln` file just created to verify the updates. The GUI table would contain the same information shown in Figure 2-3.

Edit Plan File Tables Using the GUI

Step 1 Select the Edit->Plan Tables as Database menu. The Plan Table Database Editor dialog box opens, showing all possible tables in the left pane.

Step 2 In the left pane, select the desired table. If the table already exists in the plan, it is displayed.

Step 3 Optional: Filter the table entries by using the Filter control menu or by entering text in the Search field (top right).

Step 4 Click on the field you want to change and then edit the information, or choose one of these more advanced editing options.

- Select the Set Value tab, choose the Column to modify, enter an SQL Lite expression that specifies the change, and click Set.
- Select the Advanced Filter tab, and enter an SQL Lite expression that specifies the change, and click Filter.

Step 5 Visually verify that the change is correct.

Step 6 To open the new plan, click either on Open in MATE. To save the new plan without opening it, click on Save as File.

Error Handling when Opening Edited Plans

Because plan files can be edited directly by the user, MATE must handle errors when opening plan files. The default behavior for table errors is to log a warning and ignore any row of any table that is inconsistent with information already processed. For example, if the <Nodes> table lists two nodes with the same name (which is a key column) the second row is ignored.

To simplify user editing of `.txt` format plan files, MATE gracefully handles common problems encountered during the opening process. For example, although the simplest plan can contain nodes, interfaces, and circuits, there are certain situations in which only an <Interfaces> table need be specified.

MATE makes the following changes when opening a plan with missing or incomplete information.

- Omitted tables are assumed to be present with no entries.
- Key columns are required for each row in a table. Other columns can be entirely omitted, and default values are assumed for all their entries. The default values are listed in the Table Format Reference.
- Individual entries that are left blank (in non-key columns) are filled in with default values.
- Any nodes created in the Node column of the Interfaces table are created in the <Nodes> table if they do not already exist.

- The columns Remote Node and Remote Interface in the <Interfaces> table are derived columns, which are normally ignored when MATE opens a plan. However, in this case, when the Node, Interface, Remote Node, and Remote Interface unambiguously identify another interface as the remote interface for this circuit, and the two interfaces are not already defined as belonging to any circuit in the <Circuits> table, then a circuit connecting these two interfaces is created.
- In the <Nodes> table, if the site entry is blank, a new site is created for this node with the same name as the node. Sites referenced in the <Nodes> table are added to the <Sites> table if not already present.
- If Node and Interface in the <InterfaceTraffic> table are both blank, and the (derived) column IPAddress in the table contains a valid reference to an interface in the `<Interfaces>` table, then (Node, Interface) for that interface is entered and the traffic is associated with that interface. This allows an <InterfaceTraffic> table to be added to the plan that references traffic measurements only by IP Address.
- If Source in the <LSPTraffic> table is blank, and the (derived) column SourceIP contains a valid reference to a node, and this Node and the Name entry uniquely identify an LSP, then that node name is entered into the Source column and the traffic is associated with that LSP.

Tables as Command Arguments

Many MATE CLI tools take arguments that require the specification of a set of plan objects. For example, `merge_nodes` takes a `nodes-table` parameter that specifies the list of nodes to merge. Such a list can be specified in a file containing a table in the MATE table format. Tables used as arguments must contain at least the key columns defined for that table for the tool to uniquely match the objects in the import table with the objects in the plan.

To create tables for use as command-line tools arguments, follow these steps.

Step 1 Extract the full table of objects from the plan that is the target of the command-line tool, using `table_extract`.

Step 2 Use an editor to select the appropriate rows in the table, and delete the rest. Or, typically in a script, use `mate_select` to select the rows using SQL syntax.

LSP Mesh Creation Example

To create an LSP mesh between the core routers in sites ATL, MIA, and WDC of the `us_wan.txt` plan, you must first create a file that contains the list routers in the mesh. The following example extracts the Nodes table from the plan file, and then selects the desired core routers, and saves the result as `lsp_nodes.txt`.

Step 1 First, extract the Nodes table from the plan file.

```
table_extract -plan-file us_wan.txt -out-file nodes-table.txt -tables Nodes
```

Step 2 Select the desired core routers and save the result as `lsp_nodes.txt`.

```
mate_select -table-file nodes-table.txt -out-file lsp_nodes.txt -table nodes -filter
"(Site LIKE 'ATL' OR Site LIKE 'MIA' OR Site LIKE 'WDC') AND Name REGEXP '.*cr.*'"
-show-columns Name
```

Note The SQL query uses `LIKE`, rather than `=`, when selecting sites to avoid case-insensitivity problems.

The resulting `lsp_nodes.txt` file contains the following list of nodes.

```
<nodes>
Name
cr1.atl
cr1.mia
cr1.wdc
cr2.atl
cr2.mia
cr2.wdc
```

Step 3 Create the LSP mesh by invoking the `lsp_mesh_creator` tool, specifying the `lsp_nodes.txt` file as a command-line argument.

```
lsp_mesh_creator -plan-file us_wan.pln -out-file us_wan_lsps.pln
-source-nodes-table lsp_nodes.txt -dest-equals-source true
```

SQL Queries in MATE

Three command-line tools in MATE use SQL syntax for filtering, summarizing, and manipulating plan files and reports.

Step 1 `mate_select`—Filters tables in the reports.

Step 2 `mate_summary`—Summarizes tables in the reports, primarily to provide summary data for network visualization over time, using the archive feature.

Step 3 `mate_sql`—An advanced SQL query tool.

MATE uses the SQLite implementation of the SQL language (see www.sqlite.org, especially www.sqlite.org/lang.html).

The following operators have special meaning in MATE.

- `REGEXP`—Case-insensitive matching of regular expressions. For example, SQL expression

  `Name REGEXP '^cr'`

  Is true for `Name` equal to `'CR'`, `'Cr'` or `'CR01'`. (But not for `Name` equal to `'er.cr'`)

- `MATCH`—Some columns in MATE tables contain semicolon-delimited lists, for example a list of tags in the Tags column of the Nodes and Interfaces tables, or the list of interfaces in the Actual Path column of the LSP table. The `MATCH` operator tests for membership in these lists. For example,

  `Tags MATCH 'Europe'`

  Is true for Tags equal to `'Asia;Europe'`, `'EUROPE'`, and so on. The matching is case-insensitive.

  The operator '=' is case-sensitive in SQL. The operator `LIKE`, which is case-insensitive, is often more useful for MATE tables because the case is never relevant.

- SUBNET—Selects rows if the fields look like IP addresses and are in a specific subnet.

  This function has the following syntax options.

  - `SUBNET(Column_Name, 'ip_address/prefixlen')`
  - `SUBNET(Column_Name, 'ip_address/netmask')`
  - `SUBNET(Column_Name, 'ip_address', 'prefixlen')`
  - `SUBNET(Column_Name, 'ip_address', 'netmask')`

  The following examples demonstrate usage of each syntax.

  - `SUBNET(Column_Name, '192.168.1.0/24')`
  - `SUBNET(Column_Name, '192.168.1.0/255.255.255.0')`
  - `SUBNET(Column_Name, '192.168.1.0', '24')`
  - `SUBNET(Column_Name, 'ip_address', '192.168.1.0, '255.255.255.0)`

  The following shows usage in a real SELECT statement.

  ```
  select * from Table where subnet(IPAddress, '192.168.1.0/24');
  ```

  This WHERE clause matches a row if the field IPAddress is an IP address in the 192.168.1.* network.

  SQLite does not allow arbitrary functions to have infix notation, so the following notation is impossible.

  ```
  Where IPAddress SUBNET '192.168.1.0/24'
  ```

- SUBSTITUTE—Substitutes a new value for an old value in a column.

  Syntax: `SUBSTITUTE(Column_Name, 'old', 'new')`

  where 'old' and 'new' are values like those used in the sed syntax: `s/old/new`

  The following example shows how you could replace the **m** with **c** in all node names that start with **mr**.

  ```
  table_edit -plan-file x.txt -out-file y.txt -table Nodes -column Name

  -value "SUBSTITUTE(Name,'\(mr\).*','cr\1)"
  ```

MATE Perl Library

MATE provides Perl modules that simplify access to files that use the MATE table file format. This format is used, for example, in exported plan files and report files generated from the GUI. The Perl modules are contained in the `$CARIDEN_HOME/lib/perl` directory. Documentation is available using perldoc. For example, execute the following command from the installation directory.

```
perldoc lib/perl/MATE
```

Note To run a Perl script in a Windows installation, a Perl implementation needs to be present. We recommend ActivePerl, which can be downloaded for free.

External Tables

In addition to plan tables, external tables provide input to plan files or are the result (output) of running tools on the plan file.

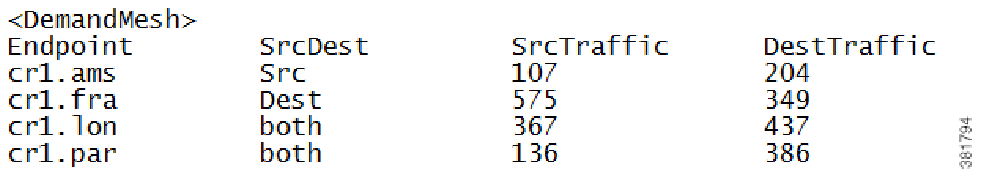
- <DemandMesh> Table
- <Edits> Table
- Tables for importing traffic and traffic growth rates; see the Importing Traffic and Growth Rates chapter.
- Tables for exporting routes and explicit LSP path settings; see the Exporting Routes and Tables chapter.

<DemandMesh> Table

A <DemandMesh> table contains columns that identify the source and destination endpoints for a demand mesh, and optionally contains columns that specify the source and destination traffic for each (Figure 2-4).

- EndPoint—Name of the demand's source or destination, for example, cr1.ams or AS{1234}.
- SrcDest—Identifies whether the endpoint is a source (Src), destination (Dest), or both. destination mesh. Default is Src. Note that an endpoint can appear twice, once in each mesh, in which case the SrcTraffic and DestTraffic should be summed. If a blank entry is summed with a non-blank entry, it becomes blank.
- SrcTraffic—Source traffic. If blank, traffic is estimated based on others and demand mesh settings
- DestTraffic—Destination traffic. If blank, traffic is estimated based on others and demand mesh settings.

Figure 2-4 Example <DemandMesh> Table



```
<DemandMesh>
Endpoint        SrcDest         SrcTraffic      DestTraffic
cr1.ams         Src             107             204
cr1.fra         Dest            575             349
cr1.lon         both            367             437
cr1.par         both            136             386
```

<Edits> Table

The `table_edit` CLI tool can optionally use a file containing an <Edits> table, which is a very time-efficient means of globally modifying MATE tables.

- Table—The name of the table to edit.
- Column—The name of the column in the table to edit. Must be one of the plan configuration columns, not derived/extended columns. If the column does not exist, it is created if the column is of the form `NameSpace::Name`.
- (RowFilter)—SQL WHERE statement selecting rows in the table. If empty, defaults to all. All columns, including derived and joined columns, are available for selection.

- Value—SQL Expression with value to insert in the column matching the filter selection. All columns, (including derived and joined columns, and including the column currently being edited, are available to use in this expression.

Example: This example shows an Edits file that change the forecast values in the demands table. The following tables show the original demands table, the edits table, and the updated demands table after running `table_edit`.

Table 2-13 Original <Demands> Table

Source	Destination	GrowthRate
A	B	1
A	C	1
B	A	1

Table 2-14 <Edits> Table

Table	Column	RowFilter	Value
Demands	GrowthRate	Source='A'	10
Demands	GrowthRate	Source='B'	20

Table 2-15 Updated <Demands> Table

Source	Destination	GrowthRate
A	B	10
A	C	10
B	A	20

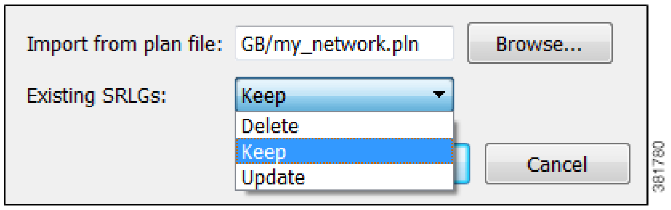
Importing Objects

Creating a plan file for modelling purposes requires specifying a number of objects that cannot be discovered directly from the network. MATE Design enables you to import shared-risk link groups (SRLGs), a Layer 1 (L1) model, a QoS model, demand groupings, external endpoints, and tags from one plan file to another, all from the File->Import menu. You can also import demand groupings from a file containing a <DemandGroupings> table.

These import tools simplify and expedite the process of importing specific objects and models. For instance, if you want to copy an L1 model into an existing plan file, you could do so using the Copy from Template tool, which copies over more information than just the Layer 1 model. Using these import tools described in this chapter allows you to copy only what is necessary.

Import SRLGs

When importing SRLGs, you can combine the imported SRLGs with existing SRLGS as follows.


Import from plan file: GB/my_network.pln
Existing SRLGs: Keep

- Delete—Delete all existing SRLGs in the destination plan file. Copy all SRLGs from the source plan file.
- Keep—Do not delete SRLGs in the destination plan file. Copy only SRLGs from the source plan file that do not have the same name as SRLGs existing in the destination plan.
- Update—Do not delete SRLGs in the destination plan file. Copy all SRLGs from the source plan file. If the two plans have SRLGs of the same name, the SRLG in the destination plan is replaced.

Step 1 Select the File->Import->SRLGs menu.

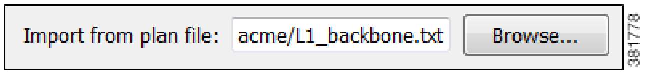
Step 2 Browse to or enter the full path of the plan file from which you are importing SRLGs.

Step 3 Select how you want the existing SRLGs handled (Delete, Keep, Update), and click OK.

For More Information ...	**See ...**
SRLGs	*MATE Design User Guide*
`import_srlgs` CLI tool	`import_srlgs` Help Output

Import Layer 1

Importing a Layer 1 model imports all Layer 1 objects, including L1 nodes, L1 links, L1 circuits, and L1 circuit hops from one plan file to another.


Import from plan file: acme/L1_backbone.txt

- All L1 objects in the destination plan file are deleted and replaced by the newly imported L1 objects.
- If an L1 node exists in a site that also exists in the destination plan file, it is placed in that same site.
- If an L1 node exists in a site that does not exist in the destination plan file, the imported site is named after the imported L1 node.
- If the source plan file contains mappings between L3 circuits and L1 circuits, and if the destination plan file contains L3 circuits with the same key column definitions, these mappings are imported. The key columns are Node A, Interface A, Node B, and Interface B.

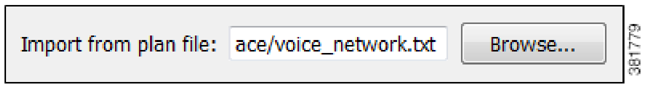
Step 1 Select the File->Import->Layer 1 menu.

Step 2 Browse to or enter the full path of the plan file from which you are importing L1 objects, and click OK.

For More Information ...	**See ...**
Layer 1 objects and Layer 1 network simulation	*MATE Design User Guide*
`import_layer1` CLI tool	`import_layer1` Help Output

Import QoS Model

The QoS model consists of the mapping between service classes and interface queues, and service class policies.


Import from plan file: ace/voice_network.txt

- All service class policies in the destination plan file, as well as all mappings between service classes and interface queues, are deleted.
- Service classes that are assigned to demands in the destination plan and that are not contained in the source plan remain, but their mappings and policies are removed.

Step 1 Select the File->Import->QoS Model menu.

Step 2 Browse to or enter the full path of the plan file from which you are importing the QoS model, and click OK.

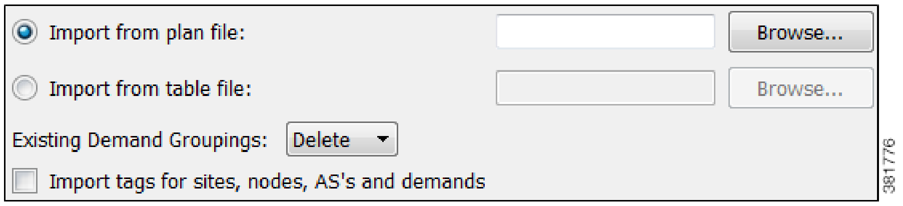
For More Information ...	**See ...**
QoS modeling, including service classes, service class policies, and interface queues	*MATE Design User Guide*
`import_qos` CLI tool	`import_qos` Help Output

Import Demand Groupings

Demand groupings define a group of demands and provide a convenient way of specifying aggregated traffic in a plan, which can be used as a basis for growth forecasting.

When importing demand groupings, you have the option of importing demand groupings from either another plan file or from a file containing a <DemandGroupings> table. You can combine the imported demand groupings with existing demand groupings as follows.

- Delete—Delete all existing demand groupings in the destination plan file. Copy all demand groupings from the source file.
- Keep—Do not delete demand groupings in the destination plan file. Copy only demand groupings from the source file that have different names than those in the destination plan.
- Update—Do not delete demand groupings in the destination plan file. Copy all demand groupings from the source file. If the two plans have demand groupings of the same name, the demand grouping in the destination plan is replaced.


● Import from plan file:
○ Import from table file:
Existing Demand Groupings: Delete
☐ Import tags for sites, nodes, AS's and demands

Step 1 Select the File->Import->Demand Groupings menu.

Step 2 Select one of these import methods.

- Browse to or enter the full path of the plan file from which you are importing a demand grouping.
- Browse to or enter the full path of the file containing the <DemandGroupings> table you are importing.

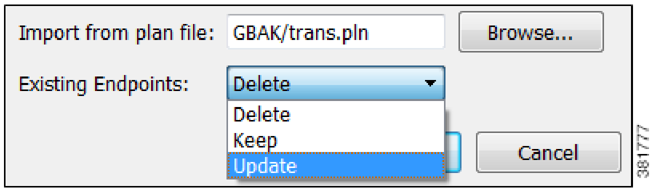
Step 3 Select how you want the existing demand groupings handled (Delete, Keep, Update).

Step 4 Select whether to import tags for sites, nodes, AS’s, and demands, and click OK.

For More Information ...	See ...
• Demands • Demand Groupings	*MATE Design User Guide*
<DemandGroupings> table	*Table Schema and CLI Reference*
Create demand groupings	*MATE Design User Guide* `insert_demand_grouping_mesh` Help output
Demand grouping traffic	*MATE Design User Guide* `import_demand_grouping_mesh` Help output
Use demand groupings in growth plans	*MATE Design User Guide* `create_growth_plans` Help output

Import External Endpoints

When importing external endpoints, you can combine them with existing external endpoints as follows.



- Delete—Delete all existing external endpoints in the destination plan file. Copy all external endpoints from the source plan file.
- Keep—Do not delete external endpoints in the destination plan file. Copy only external endpoints from the source plan file that do not have the same name as external endpoints existing in the destination plan.
- Update—Do not delete external endpoints in the destination plan file. Copy all external endpoints from the source plan file. If the two plans have external endpoints of the same name, the external endpoint in the destination plan is replaced.

Step 1 Select the File->Import->External Endpoints menu.

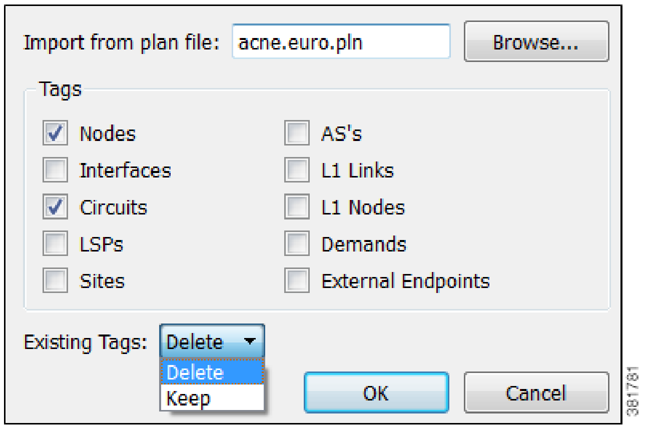
Step 2 Browse to or enter the full path of the plan file from which you are importing external endpoints.

Step 3 Select how you want the existing external endpoints handled (Delete, Keep, Update), and click OK.

For More Information ...	**See ...**
External endpoints	*MATE Design User Guide*
`import_external_endpoints` CLI tool	`import_external_endpoints` Help output

Import Tags

When importing tags, you can keep or delete existing tags as follows.



- Delete—Delete all existing tags in the destination plan file. Copy all tags from the source plan file.
- Keep—Do not delete tags in the destination plan file. Copy only tags from the source plan file that do not have the same name as tags existing in the destination plan.

Step 1 Select the File->Import->Tags menu.

Step 2 Browse to or enter the full path of the plan file from which you are importing tags.

Step 3 Select one or more object types for which you are importing tags.

Step 4 Select how you want the existing tags handled (Delete, Keep), and click OK.

For More Information ...	See ...
Tags	*MATE Design User Guide*
`import_tags` CLI tool	`import_tags` Help output

Importing Traffic and Growth Rates

This chapter describes how to import traffic and traffic growth rates into a plan file. It also describes the demand grouping traffic table that can be used in forecasting, as well as *representative* plan files, which combine traffic from multiple files.

- Traffic and Growth Rate Imports—Tables and tools for importing traffic and growth values into a plan file.
- Demand Grouping Traffic—Traffic for generating growth plans for specific time periods based on demand groupings. You can import this traffic from tables, or from MATE Live traffic reports.
- Representative Plan Files—A representative plan file is created by combining the traffic from multiple plans into a single plan based on specified times and intervals. For instance, you could collect and combine plan file snapshots for each hour of a day. These plans are representative of the current network state, and are useful for incremental planning or design purposes.

Traffic and Growth Rate Imports

MATE enables you to import the following using both the MATE GUI and the CLI tools.

- Traffic for existing interfaces, nodes, LSPs, ports, demands, and flows.
- New demands and flows with traffic.
- Growth rates for interfaces and demands. You have the option to import growth rates per interface, per demand, or per tag.

You can import traffic from one plan file into another. You can also import traffic from other data sources using files containing tab-delimited traffic tables.

Note This section describes how to import traffic and growth for interfaces, nodes, LSPs, ports, demands, and flows. You can also import growth rates for demand groupings. For information, see the Demand Grouping Traffic section.

Import Rules

Following are a few rules that MATE requires or applies when importing traffic and growth. Refer to Table 4-1 for a list of required tables, depending on what you are importing.

- Regardless of the object you are importing and regardless of whether importing traffic or growth, a traffic table must exist either within a plan file or within tab-delimited tables. When you import the traffic, the traffic tables are imported directly into the plan file.
- The interfaces, nodes, LSPs, and ports being imported must exist in the plan file into which you are importing traffic. For example, if importing interfaces, the Interfaces (<Interfaces>) table must exist and each interface identified in the traffic table must exist within the Interfaces table. If an object is in the traffic table, but is not in the plan file into which you are importing the traffic, the object is ignored.

  The same is true for demands and flows if they are not new. If they are new and do not yet exist in the plan file, you must create an object table (Demands or Flows table) for them in addition to creating their traffic table.
- Interface and demand growth rates imported using tags require an additional import growth table. This table is not directly imported into the plan, but rather is applied to the GrowthPercent column in the traffic table. The growth percent is then applied to all the interfaces or demands that have a tag corresponding to the Tag column in the import growth table.

Table 4-1 Required Tables for Importing Traffic and Growth Rates

If Importing ...	Use These Tables ...	See ...
Interface traffic	<InterfaceTraffic>	Table 4-2
Growth rates per interface	<InterfaceTraffic>	Table 4-2
Growth rates per interface tag	<InterfaceTraffic> and <InterfacesGrowth>	Table 4-2 and Table 4-8
Node traffic	<NodeTraffic>	Table 4-3
LSP traffic	<LSPTraffic>	Table 4-4
Ports traffic	<PortTraffic>	Table 4-5
Demand traffic for existing demands	<DemandTraffic>	Table 4-6
Demand traffic for new demands	<DemandTraffic> and <Demands>	Table 4-6
Growth rates per demand	<DemandTraffic>	Table 4-6
Growth rates per demand tag	<DemandTraffic> and <DemandsGrowth>	Table 4-6 and Table 4-8
Flow traffic for existing flows	<FlowTraffic>	Table 4-7
Flow traffic for new flows	<FlowTraffic> and <Flows>	Table 4-7

Traffic Tables

To import traffic, you must have a viable traffic table for each type of traffic you are importing (Table 4-1). Possible sources for these tables are as follows.

- An existing plan file
- A .txt file containing the traffic table, which might be manually created or derived from other sources.

Each traffic table must be tab-delimited and contain required “key” columns (see Table 4-2 through Table 4-7). You can modify the tables using Excel or another text editor.

Table 4-2 <InterfaceTraffic> Table

Column	Description
Node	You must either specify both the name of the source node (Node) and the interface name (Interface), or you must specify the IP address (IPAddress). The interface must exist in the <Interfaces> table using either the same node and interface combination or using the same IP address.
Interface	
IPAddress	
Queue	Use if you are importing traffic into a specific interface queue. If it does not exist, the interface queue will be created. If omitted, traffic is imported as undifferentiated.
TrafficLevel	Use if you are importing traffic into a specific travel level.If it does not exist, the traffic level will be created. If omitted, traffic is imported to the Default traffic level.
TraffMeas	Enter the amount of traffic (in Mbps) that you are importing. Required if importing traffic.
GrowthPercent	Enter the percentage by which you are growing the traffic. Required if importing growth rates.

Table 4-3 <NodeTraffic> Table

Column	Description
Node	You must either specify the node name (Node) or the IP address (IPAddress). The node must exist in the <Nodes> table using either the same node name or IP address.
IP Address	
Queue	Use if you are importing traffic into a specific interface queue. If it does not exist, the interface queue will be created. If omitted, traffic is imported as undifferentiated.
TrafficLevel	Use if you are importing traffic into a specific travel level.If it does not exist, the traffic level will be created. If omitted, traffic is imported to the Default traffic level.
SrcTraffMeas	Amount of in Mbps that is leaving the node.
DestTraffMeas	Amount of traffic in Mbps that is destined for the node.

Table 4-4 <LSPTraffic> Table

Column	Description
Name	You must either specify both the LSP name (Name) and source node name (Source), or you must specify the IP address (SourceIP). The LSP must exist in the <LSPs> table using either the same LSP name and source node name combination or using the same IP address for the source node.
Source	
SourceIP	
Queue	Use if you are importing traffic into a specific interface queue. If it does not exist, the interface queue will be created. If omitted, traffic is imported as undifferentiated.
TrafficLevel	Use if you are importing traffic into a specific travel level.If it does not exist, the traffic level will be created. If omitted, traffic is imported to the Default traffic level.
TraffMeas	Enter the amount of traffic (in Mbps) that you are importing. Required if importing traffic.

Table 4-5 <PortTraffic> Table

Column	Description
Node	Name of the node containing the port. Must reside in the <Nodes> table.
Port	Port name. Must reside in the <Ports> table with the same node that is identified in the Node column.
Queue	Use if you are importing traffic into a specific interface queue. If it does not exist, the interface queue will be created. If omitted, traffic is imported as undifferentiated.
TrafficLevel	Use if you are importing traffic into a specific travel level.If it does not exist, the traffic level will be created. If omitted, traffic is imported to the Default traffic level.
TraffMeas	Enter the amount of traffic (in Mbps) that you are importing. Required if importing traffic.

Table 4-6 Required Columns for <Demands> and <DemandTraffic> Tables

Table	Column	Description
<Demands> and <DemandTraffic>	Name	Demand name. If not specified, the default is empty.
	Source	Name of the source node. Must exist in the <Nodes> table.
	Destination	Name of the destination node. Must exist in the <Nodes> table.
	ServiceClass	Use if you are importing traffic into a specific service class. If it does not exist, the service class will be created.If omitted, traffic is imported to the Default service class.
<DemandTraffic>	TrafficLevel	Use if you are importing traffic into a specific travel level.If it does not exist, the traffic level will be created. If omitted, traffic is imported to the Default traffic level.
	Traffic	Enter the amount of traffic (in Mbps) that you are importing. Required if importing traffic.
	GrowthPercent	Enter the percentage by which you are growing the demand traffic. Required if importing growth rates.

Table 4-7 Required Columns for <Flows> and <FlowTraffic> Tables

Table	Column	Description
<Flows> and <FlowTraffic>	FlowID	Demand name. If not specified, the default is empty.
<FlowTraffic>	TrafficLevel	Use if you are importing traffic into a specific travel level.If it does not exist, the traffic level will be created. If omitted, traffic is imported to the Default traffic level.
	Queue	Use if you are importing traffic into a specific interface queue. If it does not exist, the interface queue will be created. If omitted, traffic is imported as undifferentiated.
	TraffMeas	Enter the amount of traffic (in Mbps) that you are importing.

Growth Tables

Using a growth table, you can apply growth rates to only tagged interfaces or tagged demands. The parameters identified in a growth table are applied to an associated traffic table, but are not directly imported as a table into a plan file.

There are two growth tables: <InterfacesGrowth> and <DemandsGrowth>. Each of these have only two columns.

Table 4-8 Columns in <InterfacesGrowth> and <DemandsGrowth> Tables

Column	Description
GrowthPercent	The growth rate for the undifferentiated service class in one time period. This GrowthPercent overwrites the GrowthPercent column in the <InterfaceTraffic> or <DemandTraffic> table, depending on what you are importing.
Tag	A tag value for the growth rate, which associates it with interfaces or demands that have the same tag value. Therefore, a Tag column with this tag value must exist in the <Interfaces> or <Demands> table, depending on what you are importing.

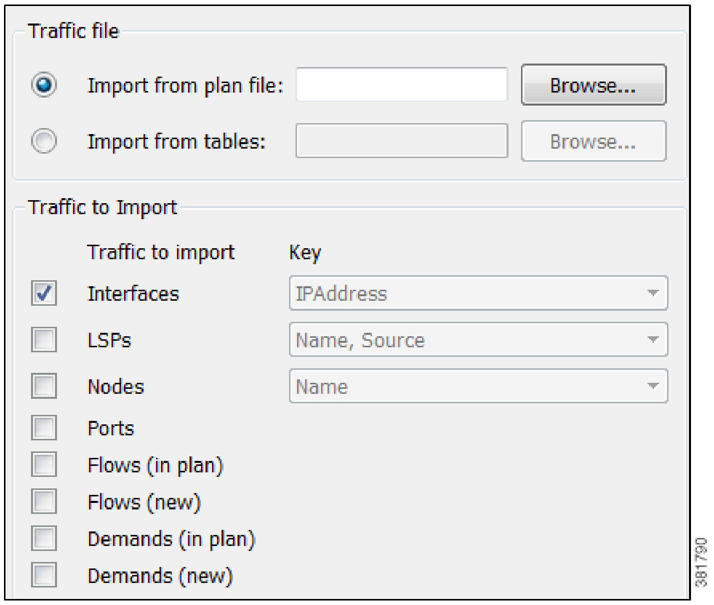
This is an efficient means of simultaneously applying growth rates to numerous interfaces or demands at one time. For example, if you have a 1,000 demands that used two tags, you could create a <DemandsGrowth> table with only two rows (one for each tag) versus creating a <DemandTraffic> table with 1,000 entries (one per demand).

Import Traffic and Growth Rates

To Import ...	Use ...	Table Prerequisites
Traffic into interfaces, nodes, ports, demands, or flows	File->Import->Traffic menu in GUI or `import_traffic` CLI tool	Object tables Traffic Tables
Traffic into LSPs	File->Import->Traffic menu in GUI or `import_lsps` CLI tool	LSP (object) table Traffic Tables
Growth rates for interfaces or demands on per-object basis	File->Import->Traffic menu in GUI or `import_traffic` CLI tool	Object tables Traffic Tables
Growth rates for interfaces or demands based on tags	File->Import->Traffic menu in GUI or `import_growth` CLI tool	Object tables Traffic Tables Growth Tables

Import Traffic and Per-Object Growth Rates Using the GUI

Step 1 Select the File->Import->Traffic menu. The Import Traffic dialog box appears.



Traffic file

- [✓] Import from plan file: Browse...
- [ ] Import from tables: Browse...

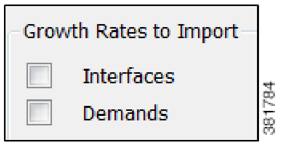
Traffic to Import

Traffic to import	Key
[✓] Interfaces	IPAddress
[ ] LSPs	Name, Source
[ ] Nodes	Name
[ ] Ports	
[ ] Flows (in plan)	
[ ] Flows (new)	
[ ] Demands (in plan)	
[ ] Demands (new)	

Step 2 Import traffic measurements.

a. In the Traffic File section, select whether to import from a plan or from a table file, and then enter or browse to the plan or table file name you are importing.

b. In the Traffic to Import section, select the traffic tables you want to import: interfaces, LSPs, nodes, ports, flows, and/or demands.



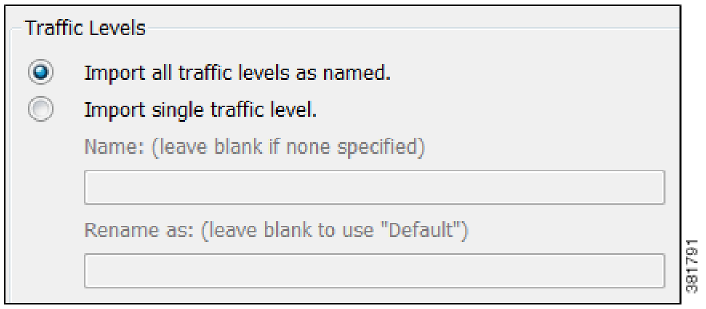
Growth Rates to Import

- [ ] Interfaces
- [ ] Demands

– If importing interfaces, LSPs, or nodes from a table, specify the columns to use as the table key. For example, for interfaces, select whether to identify the traffic imported based on the IP address or based on a combination of the node and interface name.

– For flows and demands, the "new" option imports flows or demands that do not exist in the plan file, including their traffic. The "in plan" option imports traffic only for flows or demands that currently exist in the plan.

Step 3 In the Growth Rates to Import section, select whether to import growth rates for interfaces and demands. This import is on a per-object basis. To import traffic on a per-tag basis, see the Import Growth Rates Based on Tags Using the GUI section.



Traffic Levels

- [✓] Import all traffic levels as named.
- [ ] Import single traffic level.

Name: (leave blank if none specified)

Rename as: (leave blank to use "Default")

Step 4 In the Traffic Levels section, select whether to import all the traffic levels or a single traffic level. If importing all, the traffic level names are kept on import. If importing a single traffic level when there are multiple levels available, you must specify a name. Optionally, the level can be renamed on import.

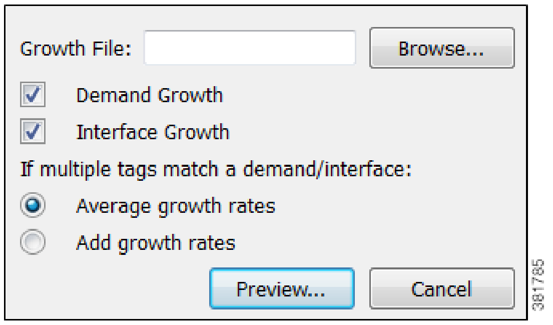
Step 5 To display a plan with the imported information, click Preview and then OK.

Step 6 Click OK.

You can also import demands by creating a demand mesh table and then import it while creating a demand mesh. See the Plan Files and Tables chapter, and see the *MATE Design User Guide*.

Import Growth Rates Based on Tags Using the GUI

Step 1 Select the File->Import->Growth menu. The Import Growth dialog box appears.



[✓] Demand Growth
[✓] Interface Growth
If multiple tags match a demand/interface:
[● Average growth rates]
Add growth rates

Step 2 Select whether to import demand or interface growth rates.

Step 3 If the interfaces or demands into which you are importing traffic have multiple tags, specify how to apply them.

- Average the growth rate evenly across the tags
- Add growth rate to each tag

Example: An interface has two tags: GrowthTrend and ExtraSales. Each tag is listed in the <InterfacesGrowth> table, but Peak has a growth of 10%, whereas Normal has a growth of 5%.

- If averaged, the resulting growth rate is 7.5%.
- If added, the resulting growth rate is 15%.

Step 4 Click Preview to see how many rows in the growth table apply to the interfaces and/or demands.

Step 5 Click OK.

Demand Grouping Traffic

MATE Design enables you to specify traffic growth per demand grouping, which in turn enables you to create growth plans that use estimates of aggregate traffic growth. Demand grouping traffic can be entered into MATE Design in several ways.

- Enter the traffic growth into the Demand Grouping Properties dialog box in the MATE GUI
- Import a single period of demand grouping traffic using the Import Demand Grouping Traffic tool.

- Create traffic reports in MATE Live and import the data using the Import Demand Grouping Traffic tool.
- Create an external <DemandGroupingTraffic> table and select the option to use this table in the Create Growth Plan GUI tool or `create_growth_plans` CLI tool.

<DemandGroupingTraffic> Table

The <DemandGroupingTraffic> table specifies traffic growth for demand groupings, over one or more periods. You can then import the traffic growth values for a single period or use the entire table to create growth plans for multiple periods.

For More Information ...	See ...
Create growth plans for multiple periods using the <DemandGroupingTraffic> table	*MATE Design User Guide* `create_growth_plans` Help output
Import single periods of the <DemandGroupingTraffic> table for use in creating growth plans	Import Demand Grouping Traffic Growth section `insert_demand_grouping` Help output

Table Content

Each row within the <DemandGroupingTraffic> table specifies forecasts for a demand grouping that must already exist in the plan file into which this table is being imported. (These demand groupings need to exist in a table called <DemandGroupings>.)

MATE calculates the expected traffic based on a combination of columns and periods where the <period> variable specifies the time period. Table 4-9 describes the columns; the examples are for the simplest forecast calculation possible where only one column and one period are used.

Table 4-9 <DemandGroupingTraffic> Columns

Column Name	Description
DemandGrouping	Name of the demand grouping. This name must exist in the <DemandGroupings> in the plan file into which you are importing the table.
TrafficLevel	Specifies that the growth is to be applied to this traffic level, which must exist in the plan file into which the table is being imported. If not specified, the default is to apply the growth to all traffic levels.
TrafficTotal:<period>	Specifies an exact traffic total in Mbps. If not specified, use the total traffic for the demand grouping of the previous period, or the base plan if this is the first period.

Column Name	Description
GrowthPercent:<period>	Specifies the percent by which to grow current traffic in Mbps. **Example:** If traffic total is 40,000 and GrowthPercent is 10, then the growth plan grows the traffic to 44,000 Mbps.
TrafficIncrement:<period>	Specifies the amount of traffic to increment by in Mbps. **Example:** If traffic total is 2600 and TrafficIncrement is 500, then the growth plan grows the traffic to 3100 Mbps **Note:** If both GrowthPercent and TrafficIncrement are used for the same <period>, the GrowthPercent is applied first.

Traffic Growth Columns and Periods

The TrafficTotal, TrafficIncrement, and GrowthPercent columns enable you to generate forecasts for multiple time periods, and each column defines a different manner in which traffic will be increased. Each column uses a <period> variable that serves several purposes. See Example <DemandGroupingTraffic> Table.

- The order in which <period> variables appear in the table defines the order in which MATE defines the growth plans.
- The <period> names the period being forecasted, for example Q1, and this name is appended to the root file name when new growth plan files are created. (Root file name is the plan file into which you are importing the table.)
- The <period> tells MATE which of the columns to combine to calculate the forecast for that one period.

  If multiple columns use the same <period> name, MATE uses all of those columns to generate one forecast for that <period>. For example, if both GrowthPercent:Q1 and TrafficIncrement:Q1 are defined, then both columns are used in the calculation.

  Columns can be (and likely are) repeated, each time with a different <period> variable. For example, if GrowthPercent:Q1 and GrowthPercent:Q2 are both defined, they are used for the creation of two separate plan files.

Traffic Growth Calculations

It is the combination of the columns and their use of <period> that MATE uses to generate growth plans, or expected traffic, for the sum of all demands within each demand grouping. For each period associated with a demand grouping the growth plan is determined using the following algorithm. See Example <DemandGroupingTraffic> Table.

Step 1 If TrafficTotal is specified, the expected traffic is set to this value.

If TrafficTotal is not specified and if this is the first period listed in table, the expected traffic is the total traffic in the demand grouping in plan file into which it is being imported (*base plan*).

If TrafficTotal is not specified and if this is not the first period, the expected traffic is the total traffic in the demand grouping for the previous period.

Step 2 If GrowthPercent is specified, the traffic growth is according to this percentage.

Step 3 If TrafficIncrement is specified, this value is added to the expected traffic after GrowthPercent is applied.

Example <DemandGroupingTraffic> Table

Table 4-10 is an example <DemandGroupingTraffic> table. Each row identifies a unique demand grouping in the ACME plan file, together with forecast data for future periods.

- TYO to All demand grouping is forecasted to grow by 1,000 Mbps in Q1 and an additional 5% in Q2.
- SEL to All demand grouping is forecasted to grow by 1,500 Mbps in Q1. In Q2, it is forecasted to grow by 5%, plus an additional 2,000 Mbps.
- PEK to All demand grouping is forecasted to have 5,000 Mbps of traffic in Q1 and grow6% in Q2.

Table 4-10 Example <DemandGroupingTraffic> Table

DemandGrouping	TrafficTotal:Q1	TrafficIncrement:Q1	GrowthPercent:Q2	TrafficIncrement:Q2
TYO to All		1000	5	
SEL to All		1500	5	2000
PEK to All	5000		6	

Table 4-11 identifies the results of applying this example <DemandGroupingTraffic> table to the ACME plan file. The newly created plan files are named ACME_Q1 and ACME_Q2.

- The Current Traffic column lists the total traffic across all demands in the associated demand grouping. This demand grouping resides in the plan file into which this table is being imported.
- The Q1 Expected Traffic column lists the traffic forecasted for that Q1 period using the rules specified in the Traffic Growth Calculations section.

  Example: SEL to All demand grouping is 1,500 Mbps increment + 3,000 Mbps of existing traffic = 4,500 Mbps.
- The Q2 Expected Traffic column lists the traffic forecasted using these same rules, only this time using Q1 traffic as the basis for calculations.

  Example: SEL to All demand grouping is (5% growth of 4,500) + 2,000 Mbps increment = 6,725 Mbps.

Table 4-11 Expected Q1 and Q2 Traffic Growth

Demand Grouping	Current Traffic	Q1 Traffic	Q2 Traffic
TYO to All	2000	3000	3150
SEL to All	3000	4500	6725
PEK to All	4000	5000	5300

Import Demand Grouping Traffic Growth

The MATE Design Create Growth Plans tool can use traffic growth information that is in the Demand Grouping table. You can import this traffic in two ways.

- Import a file containing the <DemandGroupingTraffic> table using the Import->Demand Grouping Growth menu in the MATE GUI.

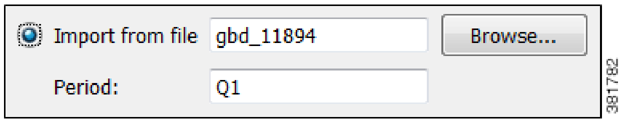
- Import MATE Live traffic trending data using the Import->Demand Grouping Growth menu in the MATE GUI.

Import Demand Grouping Traffic from a File

Requirements

- The demand groupings in the current plan must match the names of the demand groups in the <DemandGroupingTraffic> table.
- The file from which you are importing must be a .txt file.

Step 1 Select the File->Import->Demand Grouping Growth menu. The Import Traffic dialog box appears.



[✓ Import from file] gbd_11894
Period: Q1

Step 2 Select the option to import from a file.

Step 3 Either enter or browse to the file containing the <DemandGroupingTraffic> table.

Step 4 To import only traffic for a specific period, enter that Period name and then click OK. Alternatively, leave the field empty, and when you click OK you are prompted with a list of periods from which to choose.

Import Traffic Trends from MATE Live

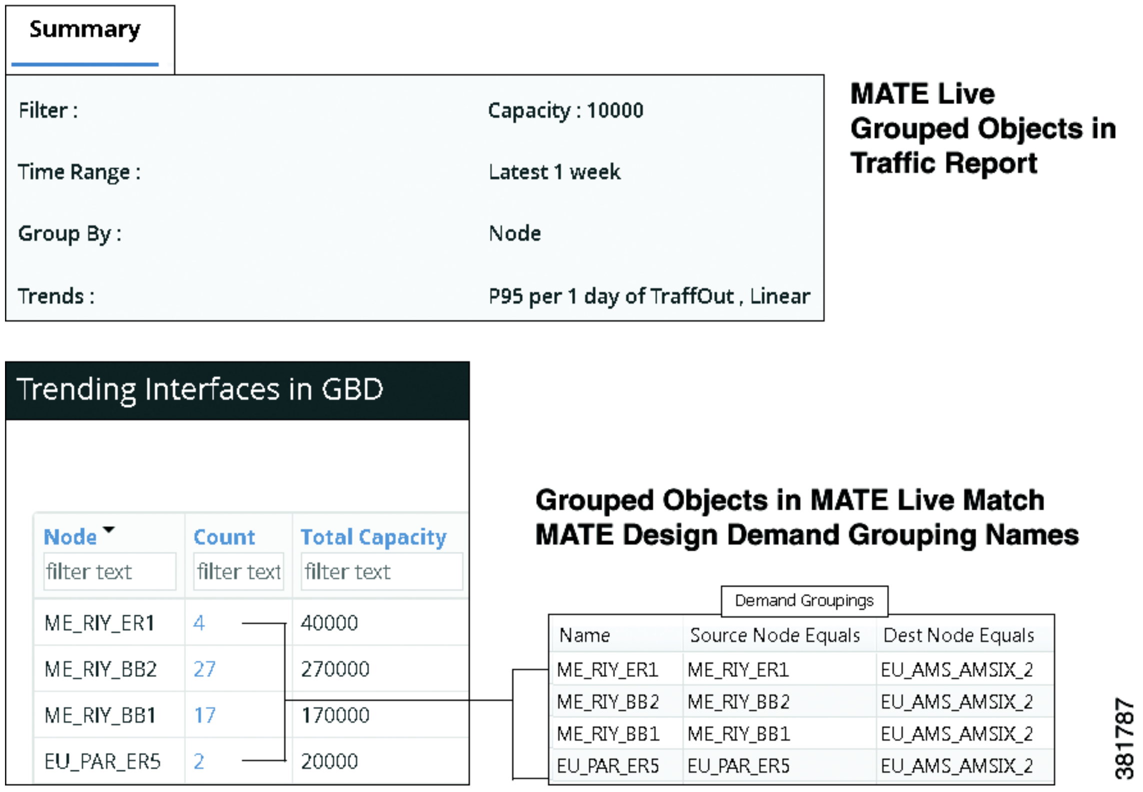
The MATE Design Create Growth Plans tool can create growth plans based on MATE Live traffic trends that are imported into the Growth Traffic (%) column of the Demand Grouping table.

Example: Figure 4-1 shows an example of a traffic report on interfaces that are grouped by node. These nodes match the names of demand groupings in MATE Design.

- A MATE Live traffic report is generated that groups ingress interfaces per node. The traffic reported per group is the total traffic entering the network at each node. Each group is named after the node.
- In MATE Design, a demand grouping is created that corresponds to each node, containing all the demands sourced at that node. The total traffic for each demand grouping is also the total traffic entering the network at that node.

The percentage growth rates calculated in MATE Live for these interface aggregates can then be imported into MATE Design and used as a percentage growth rate for the demand groupings from each node.

Figure 4-1 Example Demand Groupings in MATE Design Correlation with Grouped Objects in MATE Live


MATE Live Grouped Objects in Traffic Report
Grouped Objects in MATE Live Match MATE Design Demand Grouping Names
Summary
Trending Interfaces in GBD
Demand Groupings
381787

Filter :	Capacity : 10000
Time Range :	Latest 1 week
Group By :	Node
Trends :	P95 per 1 day of TraffOut , Linear

Node	Count	Total Capacity
ME_RIY_ER1	4	40000
ME_RIY_BB2	27	270000
ME_RIY_BB1	17	170000
EU_PAR_ER5	2	20000

Name	Source Node Equals	Dest Node Equals
ME_RIY_ER1	ME_RIY_ER1	EU_AMS_AMSIX_2
ME_RIY_BB2	ME_RIY_BB2	EU_AMS_AMSIX_2
ME_RIY_BB1	ME_RIY_BB1	EU_AMS_AMSIX_2
EU_PAR_ER5	EU_PAR_ER5	EU_AMS_AMSIX_2

Requirements

- The names of the MATE Live object groups on which you are reporting must correspond to names of MATE Design demand groupings. Figure 4-1 shows an example.
- You must know the Job ID # of the report from which you are importing traffic.
- You must know the location (path or IP address) where the report is located, and the username and password for accessing it.
- You must have an administrator role (username and password) for using MATE Live.

Generate Traffic Report in MATE Live

Step 1 **In MATE Live**, generate a traffic report for interfaces, LSPs, or demands. For information on how to create these reports, see the *MATE Live User Guide*.

- In the Group By tab, select how to group the objects.

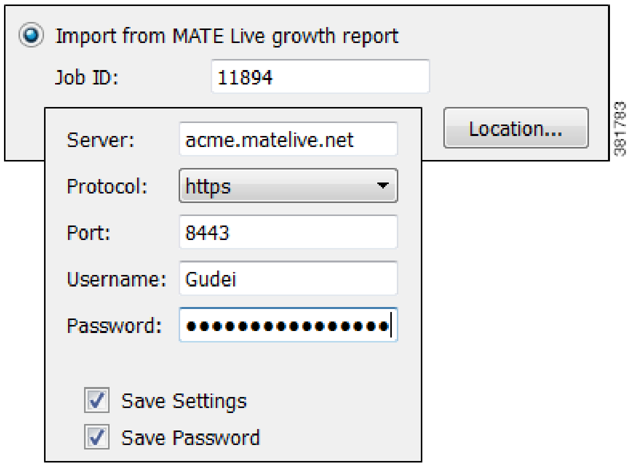
  These grouped objects are listed in the first column of the resulting report. It is this list of names that must match the demand grouping names in MATE Design. For an example, see Figure 4-1.
- In the Trends tab, select the trending value. For interfaces, this must be TraffOut, and for LSPs and demands, there is only one value, which is Traff.
- In the Trending Algorithm drop-down list (in the Trends tab), select the method of aggregating the traffic trend data.

Step 2 After running the report, note its Job ID# from the Analytics, Report Log page.

Note For more information on generating traffic reports in MATE Live, see the *MATE Live User Guide*.

Import Traffic Report Data into MATE Design

Step 1 **In MATE Design**, select the File->Import->Demand Grouping Growth menu.



Import from MATE Live growth report [✓]

Job ID: 11894

Server: acme.matelive.net

Protocol: https

Port: 8443

Username: Gudei

Password: ••••••••••••••••

Save Settings ✓

Save Password ✓

Step 2 Select the option to import a traffic report from MATE Live.

Step 3 Enter the MATE Live Job ID number for the traffic report you are importing.

Step 4 Click the Location button.

a. Enter a fully qualified path or IP address for the server name.

b. Identify how to connect to the server by selecting the appropriate protocol and entering the port number (for example, 8443).

c. Enter an administrative username and password that gives you access to the server.

d. To save all settings except for the password for future use, click Save Settings.

e. To save the password for future use, click Save Password. This option is not available unless you save settings.

f. Click OK.

Step 5 Click OK in the Import Demand Grouping Traffic dialog box.

Step 6 To verify the import was successful, show the Demand Groupings table, Growth Traffic Total column and notice the traffic is there.

Representative Plan Files

Plan files created by Collector are snapshots of a network state at a point in time. Transitory events, such as failures, are captured in the snapshot if they occur during this time. The traffic collected is the specific traffic that occurred during that collection window, which can be five minutes or less. As such, a snapshot usually does not represent the network state over the course of a typical day or week of network operation, and thus, is inadequate to use as the basis for long-term design and planning tasks. To address these needs, the `create_representative_plan` tool uses multiple snapshots from an archive to construct a single plan that is more representative of the general network state.

- The topology is extracted from a base plan, which by default is the most recent snapshot. You can, however, specify any plan file as the base plan. An archive template plan file is particularly useful since it does not contain transitory failures.
- The representative plan contains multiple traffic levels, one per time interval specified. For example, there could be one traffic level per hour of the day.
- Demands are extracted from snapshots that are selected from each of these time intervals. A single demand in the representative plan file contains a range of traffic values representative of the different amounts of traffic for that demand over the course of the specified time period.
- Interface, LSP, and node measurements are extracted from snapshots that are selected from each of these time intervals.

A representative plan file is particularly useful when planning networks where peak utilizations occur at unknown or different times of the day across different interfaces. A simulation analysis performed over all traffic levels identifies the time intervals when peaks occur.

You can fine-tune the results to include specified snapshots for multiple traffic levels across specific intervals. These time parameters are divided into two sets.

- One set includes `-time-period`, `-time-interval-length`, and `-time-interval-starts`. These parameters define time intervals during a day or during a week (defined by `-time-period`) that are created for the resulting plan file.
- The other set includes `-sample-time-end`, `-sample-time-length`, and `-time-zone`. These parameters define which periods of data in the archive are used to populate the specified time intervals.

Following is the sequence of steps the `create_representative_plan` tool uses to create the representative plan file.

- Examines all snapshots that fall into the specified time interval during the sample time range. Of these snapshots, MATE selects the one with the least number of failed circuits and the most number of active interfaces in the common base plan. If there is a tie, the snapshot with the highest amount of demand traffic is selected. Only snapshots with single traffic levels are used.
- Removes all traffic intervals from the base plan.
- Creates new traffic intervals in the base plan, using the format HHMM-HHMM or DDDHHMM-DDDHHMM, depending on whether the time period is a day or a week.

  Examples: 03:00-04:00 and Fri17:00-Fri18:00
- Imports all demands from the corresponding snapshot for the time interval.
  - If a snapshot demand matches one existing in the base plan, then MATE uses that demand and the snapshot demand traffic for it.
  - If there is no matching demand, MATE creates the demand with 0 traffic.
  - If a demands exists in the base plan, but not in the snapshot, the demand is used with 0 traffic.

  Note that multicast demands are imported with the required multicast flows and multicast destinations.
- Imports measured traffic for interfaces, LSPs, and nodes that exist in both the base plan and the snapshot.

Each representative plan file includes a report section where each traffic level is defined per row. Refer to the Examples section to see the information included per traffic level.

Examples

In this first example, the network peak time is between 4 PM and 7 PM daily. To better understand this peak traffic, we could create a representative traffic level for each hour of this range, which includes 4 PM, 5 PM, and 6 PM. For weekly forecast purposes, we are interested in sampling the last 5 days to construct the traffic levels. We are naming the output file “representative_day.pln.” We are choosing to use the latest snapshot in the time period, 110502_0347_UTC.pln, as the base plan, and it is located in the archive that is named “backbone.” Following is the command and sample output.

```
create_representative_plan -out-file representative_day.pln -archive backbone
-time-interval-length 60 -sample-time-length 1 -time-interval-starts 1600,1700,1800
```

Traffic Level	Snapshot	Matching Interfaces	Total Demand Traffic	Total Demands	Demands Not Imported
16:00-17:00	110502_0347_UTC.pln	25	43534.32	453	4
17:00-18:00	110502_0347_UTC.pln	25	47583.23	454	3
18:00-19:00	110502_0347_UTC.pln	25	50771.49	454	3

In this second example, we know the network peak times are around 4 PM daily, as well as 8 PM on Fridays. We need to get a representative traffic level for each of these six periods for the last two weeks. Today is Tuesday, so yesterday’s 4-5 PM range and last week’s Monday 4-5 PM range are used to construct the 4 PM traffic level. The name of the representative plan file we are creating is “weekly_peak.pln.” The base plan, 110407_0423_UTC.pln, is in the acme directory. Following is the command and sample Traffic Levels output.

```
create_representative_plan -plan-dir acme -base-plan 110407_0423_UTC.pln -outfile
weekly_peak.pln -time-period week -time-interval-length 60 -sample-time-length 14
-time-interval-starts Mon1600,Tue1600,Wed1600,Thu1600,Fri1600,Fri2000
```

Traffic Level	Snapshot	Matching Interfaces	Total Demand Traffic	Total Demands	Demands Not Imported
Mon16:00-Mon17:00	110407_0423_UTC.pln	97	53245.14	6702	21
Tue16:00-Tue17:00	110407_0423_UTC.pln	97	53413.36	6702	21
Wed16:00-Wed17:00	110407_0423_UTC.pln	95	49985.27	6701	22
Thu16:00-Thu17:00	110407_0423_UTC.pln	97	56831.91	6702	21
Fri16:00-Fri17:00	110407_0423_UTC.pln	93	48732.18	6700	23
Fri120:00-Fri21:00	110407_0423_UTC.pln	97	53692.39	6702	21

Related Topics

- Plan Files and Tables
- *Table Schema and CLI Reference*
- `$CARIDEN_HOME/docs/table_schema`
- *MATE Design User Guide*

Exporting Routes and Tables

MATE enables you to export the following.

- Export Routes—Routes of demands, shortest IGP, TE, and latency paths, LSPs, LSP paths, circuits, Layer 1 (L1) circuits, and L1 circuit paths. Most of these have options for specifying the hop type.
- Export Explicit LSP Path Settings
- Export L1 Link Wavelength Utilization
- Export Current Table that is showing

This ability to export routes and tables facilitates the exchange of information in a text editable format. It also supplements the analysis of routes and plan file objects. For instance, you could export demand routes using L1 link to determine which L1 links are used by demands. Since you can export routes that changed due to failure states, you can fine-tune your analysis of how failures impact the network.

Export Routes

You can export routes using the MATE GUI or using `export_routes` CLI tool.

This tool creates one of the following tables, depending on what you are exporting.

- <CircuitHops>
- <DemandHops>
- <L1CircuitHops>
- <L1CircuitPathHops>
- <LSPHops>
- <LSPPathHops>
- <ShortestIGPPathHops>
- <ShortestLatencyPathHops>
- <ShortestTEPathHops>

The table generated is output to a .txt file containing only this table. Their key columns are a combination of the key column identified for their object type plus a Step column. The Step column is an integer that identifies the sequence of this hop in the route path. For example, the <Demands> table has key columns of Name, Source, Destination, and ServiceClass. Therefore, the <DemandHops> table has these four columns to uniquely identify the demand, plus a Step column to identify the hop sequence.

Each exported hops table includes the following columns to specify details of the hop and its position with respect to other hops in the route. Note that the routes of demands might be split to account for ECMP routing or multicast. So demand hops might not be listed sequentially as a hop could be followed by two next hops as the route and traffic split into two.

- UnresolvedHop—The hop type and complete definition. For example, if{cr1.atl|to_cr1.hst} means this hop is an interface with a source node of cr1.atl and an egress interface of to_cr1.hst. For a complete list of these object notations, see the Plan Files and Tables chapter.
- PreviousStep—Identifies the previous step or steps in the route. As such, this identifies the previous hop in the route. For example, if you are on the row corresponding to Step 4, and the PreviousStep is 2, then the previous hop is contained in the row corresponding to Step 2.

  If the PreviousStep column is empty, then this is the first hop in the route.
- NextStep—Identifies the next step in that table. As such, this identifies the next hop in the route.

  If the NextStep column is empty, then this is the last hop in the route.
- TrafficProportion—The proportion of traffic on this hop compared to the other hops in the route. Although this value is usually 1, in the case of ECMP routing, it could be less than 1.

Optionally, you can fail objects prior to exporting their routes and then export only the routes that changed due to that failed state.

For More Information ...	**See ...**
Export routes through MATE GUI	Export Routes Using the GUI
Export routes using CLI tool	`export_routes` Help output
Key column definitions per table	`$CARIDEN_HOME/docs/table_schema.html` file; key columns are highlighted in orange
Key columns	Plan Files and Tables chapter
MATE tables	Plan Files and Tables chapter
Object notation	Plan Files and Tables chapter
Fail objects prior to exporting routes	*MATE Design User Guide*

Example Export Route Tables

Table 5-1 is an example <LSPHops> table that was created using active LSPs with interface hops. The cr2.par_cr1.fra LSP has no interface hops in its route. In the table, it has only one step and no previous or next steps. The cr1.ams_cr1.par LSP has three interface hops (steps 1, 2, and 3) between source (cr1.ams) and destination (cr1.par).

Table 5-1 Example <LSPHops> Table

Name	Source	Destination	Step	UnresolvedHop	Previous Step	Next Step	Traffic Proportion
cr2.par_cr1.fra	cr2.par	cr1.fra	1	if{cr2.par|{to_cr1.fra}}			1
cr1.ams_cr1.par	cr1.ams	cr1.par	1	if{cr1.ams | to_cr2.lon}}		2	1

Name	Source	Destination	Step	UnresolvedHop	Previous Step	Next Step	Traffic Proportion
cr1.ams_cr1.par	cr1.ams	cr1.par	2	if{cr2.lon|{to_cr1.lon}}	1	3	1
cr1.ams_cr1.par	cr1.ams	cr1.par	3	if{cr1.lon|{to_cr1.par}}	2		1

Table 5-2 is an example <DemandHops> table that was created using demands with interface hops (Figure 5-1). The demand route in this example splits into multiple ECMP paths.

Figure 5-1 Example Demand Route Shown in <DemandHops> Table

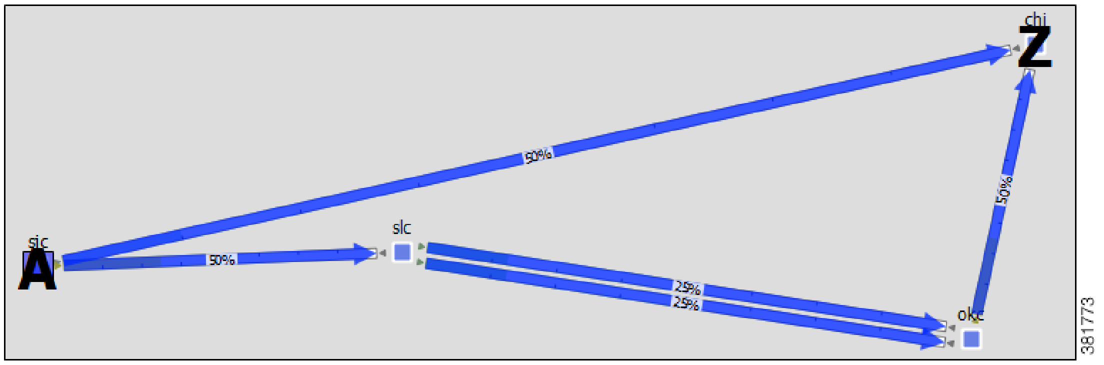


Table 5-2 Example <DemandHops> Table with ECMP Route

Name	Source	Destination	Service Class	Step	UnresolvedHop	Previous Step	Next Step	Traffic Proportion
	sjc	chi	Default	1	if{sjc|{to_slc}}		3;4	0.5
	sjc	chi	Default	2	if{sjc|{to_chi}}			0.5
	sjc	chi	Default	3	if{slc|{to_okc}}	1	5	0.25
	sjc	chi	Default	4	if{slc|{to_okc}[1]}	1	5	0.25
	sjc	chi	Default	5	if{okc|{to_chi}}	3;4		0.5

If you were to fail the circuit between sjc and slc and export only routes that changed during a failed state, the resulting exported route table would be as shown in Table 5-3.

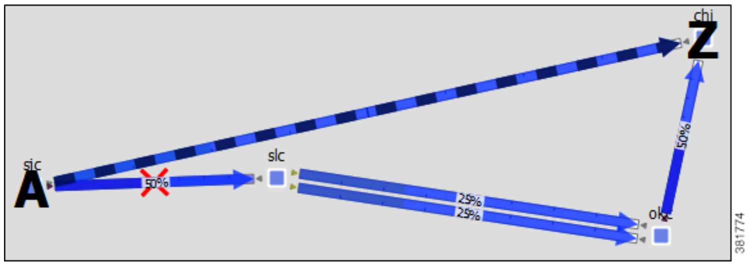


Table 5-3 Example <DemandHops> Table After Failing a Circuit

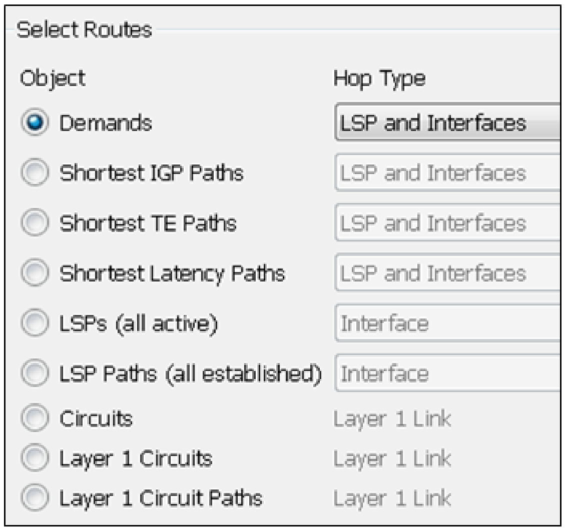
Name	Source	Destination	Service Class	Step	UnresolvedHop	Previous Step	Next Step	Traffic Proportion
	sjc	chi	Default	1	if{sjc|{to_slc}}			1

Export Routes Using the GUI

From the MATE GUI, follow these steps to export routes to the hops tables. Hops tables are created for the object that is selected in the dialog box. For example, if you select Circuits, MATE creates a <CircuitHops> table containing all circuits in the plan file.

If you want to export routes only for those objects whose route changed due failure state, fail all relevant objects before exporting the routes and select the “Only export routes changed by failure state” option. Alternatively, you could run Simulation Analysis and then fail one or more objects to the worst-case failure. For more information, see the *MATE Design User Guide*.

When exporting circuits, only L3 circuits using L1 hops are exported.


Select Routes

Object	Hop Type
(●) Demands	LSP and Interfaces
( ) Shortest IGP Paths	LSP and Interfaces
( ) Shortest TE Paths	LSP and Interfaces
( ) Shortest Latency Paths	LSP and Interfaces
( ) LSPs (all active)	Interface
( ) LSP Paths (all established)	Interface
( ) Circuits	Layer 1 Link
( ) Layer 1 Circuits	Layer 1 Link
( ) Layer 1 Circuit Paths	Layer 1 Link

When exporting L3 circuits and L1 circuits, the Active Path Sim property in the L1 Circuits table is considered. This property identifies which L1 circuit path is currently being used by the L1 circuit. If an L1 circuit path is not available due to failure, the L1 circuit’s Active Path Sim is updated to the L1 circuit path that is being used. When exporting L1 circuit paths, all routed paths are exported.

Step 1 Select the Export->Export Routes menu. The Export Routes dialog box appears.

Step 2 Select the object for which you want to export routes.

Step 3 Except for circuits and L1 circuits, select the hop type that you want to export. For circuits, L1 circuits, and L1 circuit paths, the only available option is Layer 1 links hops.

Step 4 If exporting only routes that changed due to a failure, select the associated option.

Step 5 Click OK, and save the .txt file to the location of your choice.

Export Explicit LSP Path Settings

Tab-delimited .txt files listing all explicit LSP path settings are created by selecting the File->Export->Explicit LSP Settings menu in the MATE GUI, or using `export_routes` CLI tool. The Hops column lists the explicitly routed path of each LSP in MATE object notation. For a complete list of these object notations, see the Plan Files and Tables chapter.

Table 5-4 is an example output file. In addition to these columns, the file also contains SetupBW, SetupPri, and HoldPri columns. Figure 5-2 shows the first LSP path option listed in Table 5-4.

Figure 5-2 Path Option 1 for the cr1.ams_er1.BRU LSP

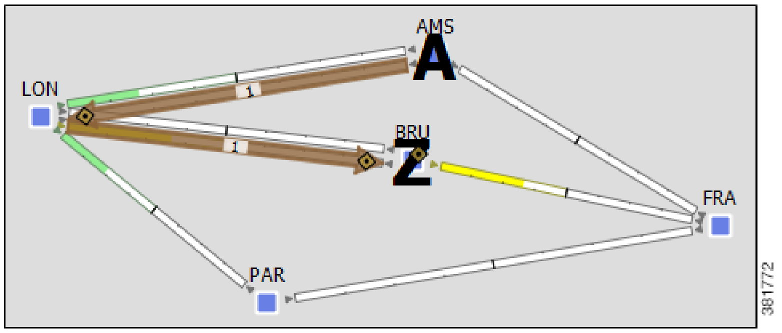
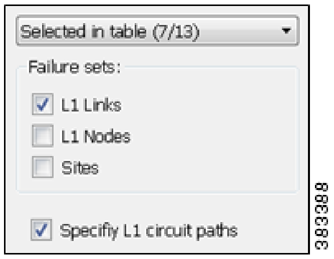


Table 5-4 Example File Containing Explicit LSP Path Settings

Name	LSP	Source	Destination	Preference	Hops
cr1.ams_er1.BRU_1	cr1.ams_er1.BRU	cr1.ams	er1.BRU	1	if{er1.BRU|{to_cr1.BRU}},if{cr1.BRU|{to_cr2.lon}},if{cr2.lon|{to_cr1.ams}}
cr1.ams_er1.BRU_1	cr1.ams_er1.BRU	cr1.ams	er1.BRU	2	if{er1.BRU|{to_cr2.BRU}},if{cr2.BRU|{to_cr1.lon}},if{cr1.lon|{to_cr2.ams}},if{cr2.ams|{to_cr1.ams}}

Export L1 Link Wavelength Utilization

Changes in the routing and Lambdas of L1 circuits paths affect the utilized wavelengths on L1 links. Any L1 failure or a site failure could cause the L1 circuit path to be rerouted. If Auto Lambda is enabled, then the L1 circuit path could be forced to select a Lambda on a different path option in order to avoid lambda blocking. For more information about Layer 1, see the *MATE Design User Guide*.



- [✓] L1 Links
- [ ] L1 Nodes
- [ ] Sites
- [✓] Specify L1 circuit paths

The Export Lambda Utilization tool enables you to create and export a file containing a <LambdaUtil> table that identifies all of the L1 circuit path Lambda Sim values that are routed on each L1 link. This is useful for determining how many L1 circuits paths are using an L1 link, to determine how fragmented the wavelength usage is across L1 links, and to perform more elaborate wavelength assignments.

The table shows whether there is a failure, the L1 link, and the Lambda values of the L1 circuit paths traversing it. Each Lambda column correlates to a different L1 circuit path. To see the names of the L1 circuit paths, use the “Specify L1 circuit paths” option.

Example: In this simple network there are no failures, and only four Lambdas are used. The ams-lon L1 link has an L1 circuit path using Lambda1 and none using the remaining Lambdas. The lon-par L1 link has four L1 circuit paths using it.

<LambdaUtil>							
Failure	Name	L1NodeA	L1NodeB	Lambda1	Lambda2	Lambda3	Lambda4
none	ams-lon	cr1.ams	cr2.lon	T	F	F	F
none	lon-par	cr2.lon	cr1.par	T	T	T	T

To see the effects of L1 link, L1 node, and sites failures on the wavelength utilization of L1 links, select the relevant failure sets.

Example: Row 2 shows that the L1lnk{rom|lon|rom-lon} failure causes an L1 circuit path to switch from one Lambda 1 to another lambda on the ams-lon L1 link or causes it to switch path options that it is using. Row 3 shows that the L1lnk{par|rom|par-rom} failure prevents L1 circuit paths from using either Lambda1 or Lambda2 when traversing the ams-lon L1 link.

	<LambdaUtil>							
	Failure	Name	L1NodeA	L1NodeB	Lambda1	Lambda2	Lambda3	Lambda4
1	none	ams-lon	cr1.ams	cr2.lon	T	F	F	F
2	L1lnk{rom|lon|rom-lon}	ams-lon	cr1.ams	cr2.lon	F	T	T	T
3	L1lnk{par|rom|par-rom}	ams-lon	cr1.ams	cr2.lon	F	F	T	T

Step 1 If you want to export the lambda utilization for only selected L1 links, first select those L1 links.

Step 2 Select the File->Export->Lambda Utilization menu.

Step 3 Select which L1 links to use: All, those selected, or a selected tag.

Step 4 If you want to run Simulation Analysis on L1 links, L1 nodes, or sites, select one or more of these options.

Step 5 If you want to know the name of the L1 circuit path for each Lambda referenced in the table, select "Specify L1 circuit paths."

Step 6 Click OK.

Export Current Table

To export the table that is currently showing in the MATE GUI, select the File->Export->Table menu. Only the columns that are showing appear in the exported table. To show and hide tables and columns, use the View->Tables menus.

You can also export a table or portions of a table using a simple cut-and-paste method. Select the rows and columns, use Ctrl-C (Cmd-C on Mac) to copy the selection, and then paste it to the desired document. If you paste this information into an Excel document, the data is correctly placed into the Excel rows and columns.

Importing Offline Discovery

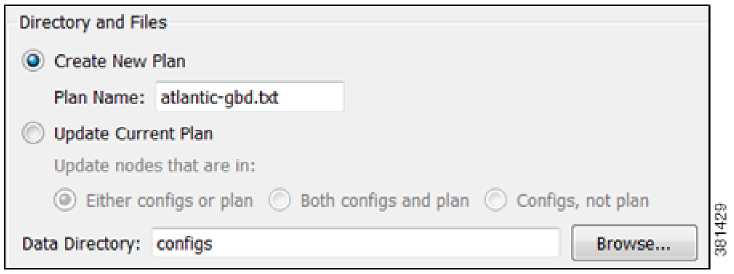
This chapter describes the MATE GUI tools available to discover and retrieve information from router configuration tools and from the Alcatel-Lucent 5620 Server Aware Manager (SAM) server. These tools are useful for capturing and importing network information into the MATE GUI.

- Parse Configs—Import Router Configuration Files
- Parse IGP—Import IGP Database
- SAM Get Plan—Import SAM Server Information

This chapter gives high-level information on accessing the tools from the MATE GUI. These same tools, plus several other offline tools, are available from the CLI and are described in more detail in the references listed in the Related Topics section.

Import Router Configuration Files

The Parse Configs tool reads a set of Cisco, Juniper Networks, and/or Huawei router configuration files, creates a MATE plan file of the network, and imports it into the MATE GUI.


Directory and Files

[✓ Create New Plan]

Plan Name: atlantic-gbd.txt

Update Current Plan

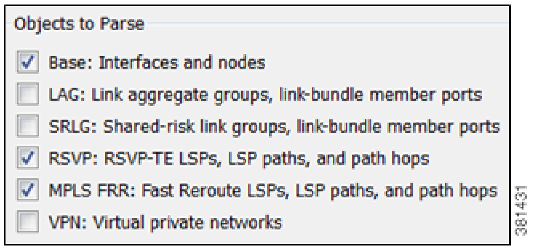
Update nodes that are in: [✓ Either configs or plan] Both configs and plan Configs, not plan

Data Directory: configs

Step 1 Select the File->Get Plan from->Configs menu.

Step 2 Select whether to create a new plan file or add information to an existing plan.

- For new plans, enter the complete path and plan file name you are creating.
- If updating an existing plan file, select how to update nodes.
  - Update only if the nodes exist in either config files or a plan file.
  - Update only if the nodes exist in both config files and a plan file.
  - Update only if the nodes exist in config files and do not exist in a plan file.


Objects to Parse
- [✓] Base: Interfaces and nodes
- [ ] LAG: Link aggregate groups, link-bundle member ports
- [ ] SRLG: Shared-risk link groups, link-bundle member ports
- [✓] RSVP: RSVP-TE LSPs, LSP paths, and path hops
- [✓] MPLS FRR: Fast Reroute LSPs, LSP paths, and path hops
- [ ] VPN: Virtual private networks

Step 3 Enter or browse to the name of the directory containing the router configurations to be parsed.

Step 4 Select one or more configuration objects to parse.

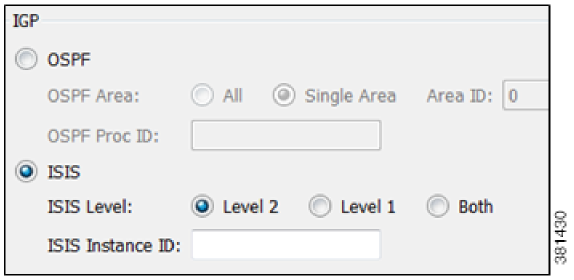
Step 5 Select whether IGP is OSPF or IS-IS.

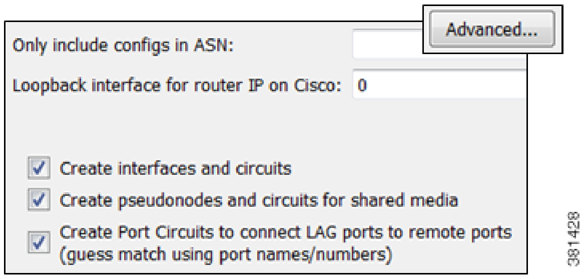
- If parsing OSPF, either use all OSPF areas or select one and enter its area ID as an integer or IP address. The default is area 0.

  If multiple process IDs exist, MATE uses the first one in the config unless you specify otherwise.

- If parsing multi-level IS-IS, select whether to use Level 1, Level 2, or both.

  If multiple instance IDs exist, MATE uses the first one in the config unless you specify otherwise.


IGP
- OSPF
  - OSPF Area: All [✓ Single Area] Area ID: 0
  - OSPF Proc ID:
- [✓ ISIS]
  - ISIS Level: [✓ Level 2] Level 1 Both
  - ISIS Instance ID:



Only include configs in ASN:
Loopback interface for router IP on Cisco: 0
- [✓] Create interfaces and circuits
- [✓] Create pseudonodes and circuits for shared media
- [✓] Create Port Circuits to connect LAG ports to remote ports (guess match using port names/numbers)

Step 6 To set advanced options, click Advanced.

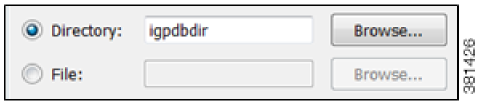
- Enter values for these options, as needed, if you prefer not to use the following MATE defaults.
  - Ignore BGP ASNs.
  - Loopback interface for Cisco routers is 0.
- Select these options, as needed.
  - Create interfaces and circuits to build a network topology after parsing the configs.
  - Create pseudonodes and interfaces for matching circuits for shared media, such as Ethernet LANs.
  - Create port circuits to connect LAG ports to remote ports. The match between the two is created in ascending order using a combination of port names and port numbers.

Step 7 To save the imported SAM information as a MATE plan file, select File->Save As.

Import IGP Database

The Parse IGP tool converts IGP information from router `show` commands and imports the IGP database into a new or existing plan file.

Step 1 Select the File->Get Plan From->IGP Database menu.



Step 2 Either enter or browse to one of the following.

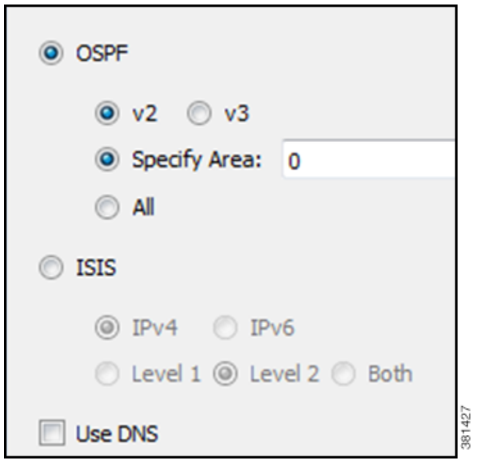
- Directory name containing multiple IGP database files.
- File name containing the IGP database. This file is the equivalent of the `show` command output for Cisco and Juniper routers, which is described in the Offline Discovery chapter of the*Wae Platform Configuration Guide*.

Step 3 Select whether to import OSPF or IS-IS database.

- For OSPF, select whether to import OSPFv2 or OSPFv3. Note that OSPFv3 is the protocol that runs in IPv6 networks.

  By default, MATE collects OSPF area 0 LSDB. To import topologies from non-zero area LSDBs, select the All option.
- For IS-IS, select whether to import IS-IS for IPv4 or IPv6. IS-IS topologies must be congruent.

  Choose Level 1, Level 2, or both metrics. If you select the Both option, MATE combines both levels into a single network, and Level 2 metrics take precedence.



Step 4 Select whether to use DNS to resolve IP addresses (node names) in the resulting plan file. (In the GUI, routers are called *nodes*.)

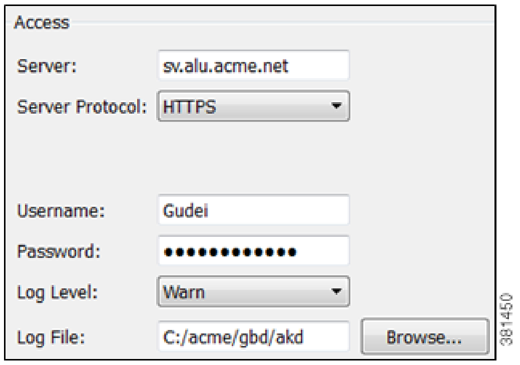
Step 5 Click OK.

Step 6 To save the imported information as a MATE plan file, select File->Save As.

Import SAM Server Information

The SAM Get Plan retrieves network topology, as well as current or historical interface traffic measurements, from 5620 SAM servers, and imports it into the MATE GUI. The tool queries each router in the discovered network to obtain additional information, such as router name, vendor, and model. It also creates a MATE plan file.

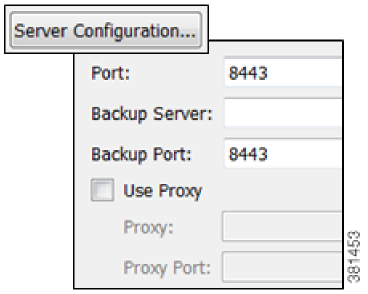
Step 1 Select the File->Get Plan from->SAM menu.


Access

Server:	sv.alu.acme.net
Server Protocol:	HTTPS
Username:	Gudei
Password:	●●●●●●●●●●●●
Log Level:	Warn
Log File:	C:/acme/gbd/akd

Step 2 In the Access section, specify the following.

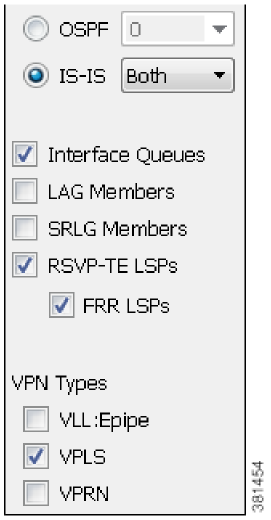
- Server—Enter the name or IP address of a SAM server.
- Server Protocol—Select a protocol that the SAM server supports: either HTTP (not encrypted) or HTTPS (encrypted).
- Username—Enter the user name for the SAM server. The user account must have the oss role as one of its Scope of Commands Roles. The admin group does not have this role by default. See the SAM Integration chapter in the *WAE Platform Configuration Guide* for information on setting it and its password.
- Password—Enter the password associated with the user account configured on the SAM server.
- Log Level—Select the logging level you wish to record when SAM Get Plan is executed.
- Log File—Enter or browse to the location where you want to store the log file.
- Server Configuration—Click to specify a port number, backup server, backup port, or proxy server.



Port:	8443
Backup Server:	
Backup Port:	8443
☐ Use Proxy	
Proxy:	
Proxy Port:	

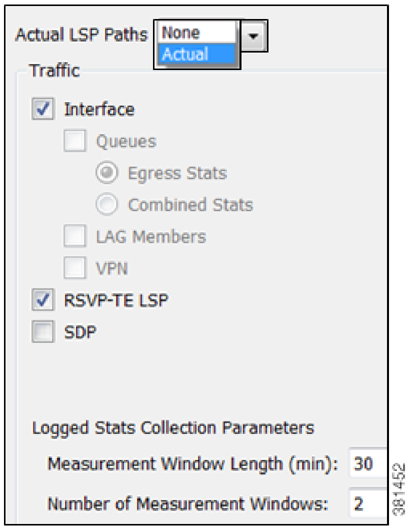
Step 3 In the Configuration section, specify the IGP topology discovered by SAM that you want to import.

- OSPF—Select if the IGP is OSPF, and then either select or enter its area.
- IS-IS—Select if the IGP is IS-IS, and then select the level as 1, 2, or both.
- Optional selections—Select each topology option according to the configuration information SAM has been configured to collect and that you need to import into the plan file. You can make multiple selections.


( ) OSPF 0
(● IS-IS) Both
[✓] Interface Queues
[ ] LAG Members
[ ] SRLG Members
[✓] RSVP-TE LSPs
[✓] FRR LSPs
VPN Types
[ ] VLL:Epipe
[✓] VPLS
[ ] VPRN

Step 4 In the Operational Data section, identify whether to include actual LSP paths and traffic statistics in the MATE plan file. The default is to not include them.

- To import the actual LSP path, select Actual.
- Select one or more types of traffic statistics to import. These are not available unless you are collecting the associated topology.
- If including traffic statistics, identify the criteria for the logged statistics to import.
  - Measurement Window Length (min)—Amount of time over which the traffic is averaged. For example, if this is set to 30, the amount of traffic over a 30-minute time period is averaged. The recommended value is greater than twice the Collection Interval of the accounting policy on the SAM server.
  - Number of Measurement Windows—Number of samples being collected. For example, if you enter a 2 here and a 30 in the Length field, you would generate two 30-minute samples.


Actual LSP Paths: None / Actual
Traffic
[✓] Interface
[ ] Queues
(● Egress Stats)
( ) Combined Stats
[ ] LAG Members
[ ] VPN
[✓] RSVP-TE LSP
[ ] SDP
Logged Stats Collection Parameters
Measurement Window Length (min): 30
Number of Measurement Windows: 2

Step 5 To import the SAM information, click OK.

Step 6 To save the imported SAM information as a MATE plan file, select File->Save As.

Related Topics

- *Table Schema and CLI Reference*
- `$CARIDEN_HOME/bin`
- *WAE Platform Configuration Guide*

Add-Ons and GUI Customizations

MATE supports tools for customizing the MATE GUI, enabling you to tailor it to meet your specific needs.

- Add-On Applications—Tools for creating add-ons, which are user-defined applications that can be called from the MATE GUI to edit and process open plan files.
- Customized Network Plot Views—Tools for creating customized network plot views.

Add-On Applications

MATE provides basic tools for incorporating add-ons (scripts or executables) into the MATE GUI.

- Perl library for editing table files, for example .txt format plan files (see the Plan Files and Tables chapter).
- CLI tools for editing plan files, such as `table_extract`, `table_edit`, and `table_replace` (see the *Table Schema and CLI Reference*).
- A framework for creating dialog boxes for data entry to the add-on (see the Create Add-Ons section).

Once registered with the MATE GUI, add-ons are accessible from the Add-Ons menu. If you do not see an add-on that you expect, select the Add-Ons->Find Add-Ons menu.

Create Add-Ons

Creating an add-on involves these steps.

Step 1 Create an executable to perform the add-on task. This executable processes specific CLI parameters, which the GUI then uses to pass details of plan files, reports, and options specified through the add-on dialog box. See Construct Add-On Executable.

Step 2 Construct the add-on's configuration file (`addon.txt`), which contains two tables. See Construct addon.txt.

- `<AddOnConfigs>`—Defines the add-on name and description, as well as how the executable should be invoked.
- `<AddOnOptions>`—Defines the MATE GUI dialog box from which the executable is invoked from the GUI.

Step 3 Construct the `return-config-file` parameter of the executable to specify the MATE GUI behavior upon exiting the add-on. See Construct return-config-file.

Step 4 Register the add-on with the MATE GUI, which is putting the executable and `config.txt` file where they can be discovered by MATE. See Register Add-Ons.

Construct Add-On Executable

The add-on can invoke any executable file, such as a shell script, Perl script, or binary executable. The executable must recognize the parameters listed in Table 7-1, which are passed to the add-on executable when it is called from the GUI.

All parameters except `-plan-file` are used by the MATE GUI to pass a temporary file or directory name to the add-on script. The script then optionally adds the contents to create the desired file or report. For example, you can create an add-on that generates reports on the existing plan file, as well as one that returns a new plan for display in the GUI.

Table 7-1 Parameters Passed by the MATE GUI to the Add-On Executable

Parameter	Description
plan-file	Required: The plan file in .pln format that is currently active in the GUI.
out-file	Optional: A file name to which the add-on saves a modified version of the plan file. The MATE GUI opens this new plan file after the add-on has completed.
pdf-report-file	Optional: A file name to which the add-on writes a report in PDF format. When exiting the add-on, this report is opened using the default PDF viewer. For information on creating PDF reports, see the Reporting Tools chapter.
report-dir	Optional: A directory name in which the add-on creates a report that is imported into the currently active plan file and opens the report to its first section. The ShowReport value in the <AddOnReturnConfig> table can show this same report. The ShowReport value also overrides the report-dir value. See the Construct return-config-file section. For information on creating reports that can be stored in plan files, see the Reporting Tools chapter.
report-file	Optional: A file name to which the add-on writes a plain text report section. The MATE GUI places this report section in an Add-Ons report in the active plan file. This parameter is available for backward compatibility. We recommend you use the report-dir parameter instead. The ShowReport value in the <AddOnReturnConfig> table overrides this property value. See the Construct return-config-file section.
return-config-file	The file name to which the add-on writes the instructions on MATE GUI behavior after the add-on has completed. This file contains an <AddOnReturnConfig> table. For information, see theConstruct return-config-file section. This parameter is required if using the report-file parameter.

Error Messages and Warnings

The add-on can generate error messages and warnings by writing them to the standard output to the GUI log window (accessed via the Window->Log Window menu).

- Error messages must start with the word 'Error:' on a new line.
- Warning messages must start with the word 'Warning:' on a new line.

The add-on can inform the MATE GUI of its progress by writing the percentage completed to the standard output. This must be a single line containing a number followed by the `'%'` character. The GUI displays this progress in a progress bar while the add-on is running.

A return value of 0 from the add-on indicates that the add-on ran successfully. Any other value indicates an add-on failure.

Construct addon.txt

The `addon.txt` configuration file contains two tables.

- `<AddOnConfigs>`—Defines the add-on name, provides a description of the add-on, and identifies how to invoke the executable should be invoked (Table 7-2).
- `<AddOnOptions>`—Optional: Defines input fields in the add-on dialog box (Table 7-3). A CLI parameter is specified for each field, and these are passed to the executable, together with the input values. The rows are sequentially read.

Table 7-2 <AddOnConfigs>

Property	**Value**
Description	Description displayed in the dialog box.
Name	Name displayed in the dialog box. The name displayed in the Add-Ons menu is a subdirectory name.
Planformat	This value determines the type of column information are provided to the add-on. The advantage to using .txt or .db is that you do not have to extract the derived information using other means, such as `table_extract`. Default = pln; only plan file column information Options = txt, db; plan file and derived column information
Run	Required: The command used to execute the add-on.
Version	Version number displayed in the dialog box.
WindowsRun	The command used to execute the add-on under Windows, if different than the Run parameter.

Table 7-3 <AddOnOptions> Table

Column	Description
Tab	Create tabs and group options within them. • If left empty, the tab is unnamed. If only an unnamed tab is present, it is not displayed and instead the options are shown in a dialog box without tabs. • If the Tab column has the same value as a previously named Tab value, the option is added to the same tab, and is placed below the previous option. If the value is different, a new tab is created and the option is placed in that tab.
Group	Create groups and place options within them. • If there is no Group value, an option is not grouped, and if applicable, is placed below the last ungrouped option. • If the Group value is new, a new group is created within its associated tab. If that row does not have a Tab value, the group is created within an unnamed tab. • If the Group column has the same value as a previously named Group value, the option is added to that same group and is placed below the previous option provided they have the same Tab value.
OptionName	The name of the command-line option; this is prepended with a -(dash) when passed to the executable. The following names are reserved and cannot be used. • mate-version • out-file • pdf-report-file • plan-file • report-dir • report-file • return-config-file
Prompt	Text string that prompts for the desired value.

Column	Description
Type	Type of entry. • Bool—Drop-down list with True and False selections. Passes “true” or “false” to the script. • CheckBox—Toggle entry; if selected, it is true, and if not selected, it is false. • ComboBox—Selection of items to appear in a drop-down menu defined by a semicolon-delimited set of options. The user can either select from the menu or type an entry explicitly. • DropDown—Selection of items to appear in a drop-down menu defined by a semicolon-delimited set of options. The user can select from the menu. • Float—Floating point number. Range is expressed as x;y. • Integer—Integer. Range is expressed as x;y. • Password—Text string. Characters typed by the user are hidden. • OpenDir—Directory name. Same as OpenFile but for directory names. • OpenFile—Name of an existing file. Default specifies the file name (no path) in the current directory. Can browse with a browse button to a different directory. The complete path is passed to the script. • SaveFile—File name. Same as OpenFile, except it can create a file. • String—Text string. Range is ignored. • TableSelection—Drop-down list for the objects in a specified table. For example, if you specify TableSelection in the Type column and Interfaces in the Table column, the drop-down list gives you standard MATE Design choices of selected, all, none, or tags. The Range and Default columns are ignored if this type is specified. • TableEntry— Drop-down list containing the information for traffic levels, service classes, or interface queues. For example, if you have multiple traffic levels in the plan file, this drop-down list enables you to choose one of the available traffic levels. Specify the list in the Table column. The selected item is passed to the script as an argument for the option in the OptionName column.
Table	The table for the TableSelection type or the TableEntry type. • TableSelection options can be any table that is accessible from the MATE GUI using the View->Tables->Show/Hide Tables menu except for the Shortest Paths table. • TableEntry is either Queues, ServiceClasses, or TrafficLevels.
Range	Varies depending on the Type identified.
Default	Varies depending on the Type identified.

Construct return-config-file

The file named by the `return-config-file` parameter stores the return configuration information in the `<AddOnReturnConfig>` table (Table 7-4). This table contains a pair of columns, Property and Value, where the values define the MATE GUI behavior upon exiting the add-on (Table 7-4). For information on the GUI behavior of views, traffic levels, service classes, and queues see the *MATE Design User Guide*. For information on plot layouts and traffic utilization colors, see the *MATE GUI Visualization Guide*.

Table 7-4 <AddOnReturnConfig> Properties and Values

Property	**Value**
OutputFileName	Output (resulting) plan file name. The string can contain a $1 variable, which is the name of the input plan file (without file extension). The default is $1-out. **Example:** If the input plan file name is euro_geo.pln, by default the output file name is euro_geo-out.pln.
PlotLayout	The GUI displays the specified layout view. If not specified, the GUI displays the Default layout, which is customary for a newly opened plan file.
Queue	The GUI displays traffic for the specified queue. If both ServiceClass and Queue are specified, the ServiceClass value has priority.
ServiceClass	The GUI displays traffic for the specified service class. If not specified, the GUI displays undifferentiated traffic.
ShowReport	The GUI opens the report, including the report-dir report, which should be inserted into the plan before this rule is applied.
ShowReportSection	The report opened by ShowReport opens to this section. If not specified, the report opens to the first section.
TrafficLevel	The GUI displays this traffic level. If not specified, the GUI displays the Default traffic level, which is customary for a newly opened plan file.
UtilColorList	The GUI fills interfaces with these colors to show levels of traffic utilization. If not specified, the default threshold settings and colors are used. Following is the format, where threshold is a real number and colors are of the form #RRGGBB (standard HTML color names). Format: threshold%<=color<=threshold% **Example:** 0% <= red <= 25%, 25% <=green<=50%, 50%<=purple<= 75%, 75%<=blue<=100%
UserView	The GUI displays this user-defined plot view. If both View and UserView are specified, the View value has priority. For information on creating plot views, see the Customized Network Plot Views section.
View	The GUI displays one of these views. If not specified, the GUI displays whichever view is appropriate for the output plan file. • FailureImpact—Failure Impact view • MeasUtil—Measured Traffic view • SimLSPRsrv—LSP Reservations view • SimUtil—Simulated Traffic view • SimWCUtil—Worst-Case Traffic view

Register Add-Ons

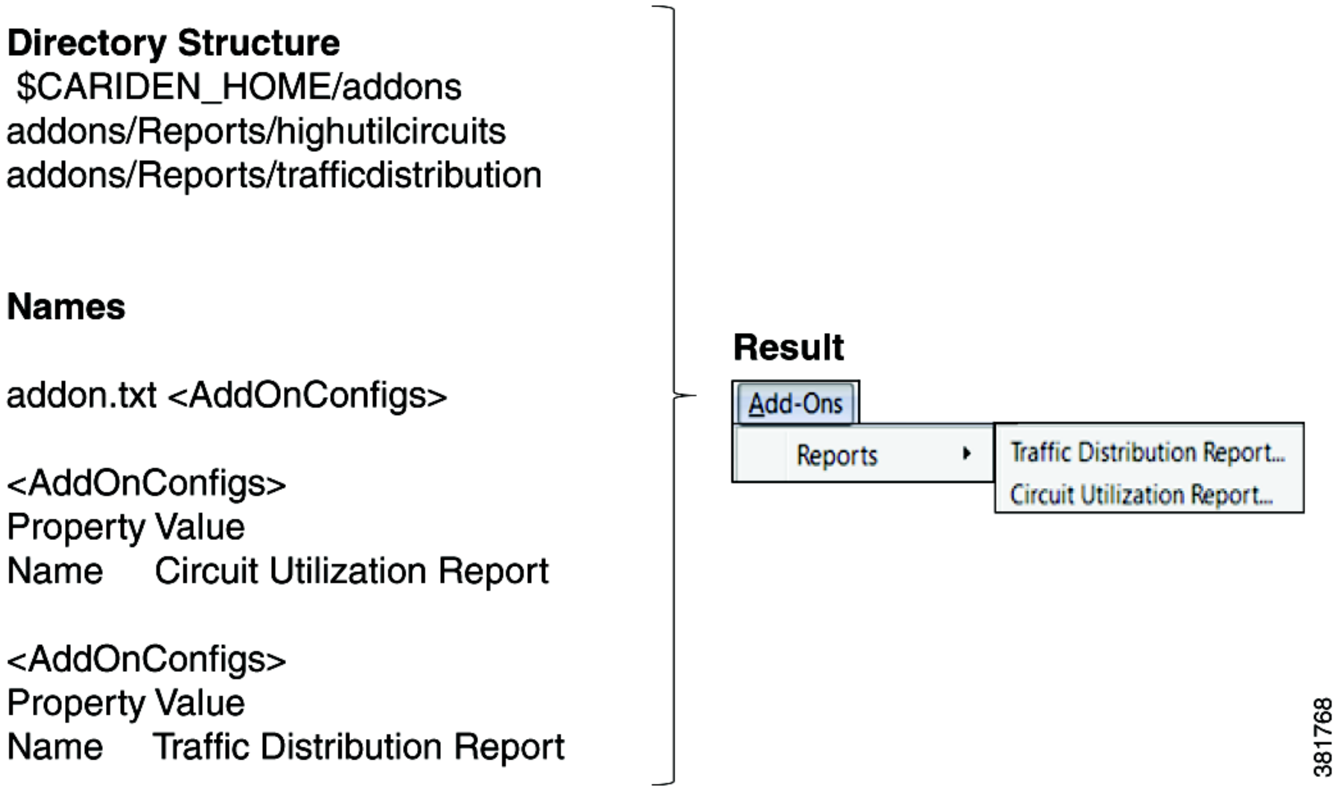
To register an add-on with MATE, create a subdirectory for the add-on in the `$CARIDEN_HOME/addons` directory. At a minimum, this directory must contain the `addon.txt` configuration file that defines the dialog box and information about the executable.

MATE adds a menu item in the Add-Ons menu using the Name parameter in the <AddOnConfigs> table. If this parameter is omitted, MATE uses the name of the subdirectory.

To create multiple levels of sub-menus in the MATE GUI, simply create nested subdirectories. See Figure 7-1 for an example.

Add-ons and their sub-menus appear in the Add-Ons menu the next time you launch the GUI, or after you select the Add-Ons->Find Add-Ons menu.

Figure 7-1 Example Add-On Submenus


Directory Structure
$CARIDEN_HOME/addons
addons/Reports/highutilcircuits
addons/Reports/trafficdistribution
Names
addon.txt <AddOnConfigs>
<AddOnConfigs>
Property Value
Name Circuit Utilization Report
<AddOnConfigs>
Property Value
Name Traffic Distribution Report
Result
Add-Ons
Reports
Traffic Distribution Report...
Circuit Utilization Report...
381768

Example Add-Ons

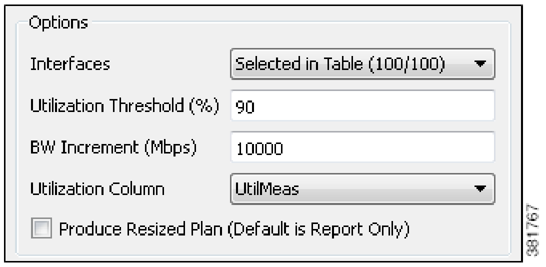
MATE Design includes three example add-ons: Circuit Upgrade, Circuit Utilization Report, and Traffic Distribution Report.

Circuit Upgrade Add-On

The `$CARIDEN_HOME/addons/CircuitUpgrade` directory includes a sample add-on named CircuitUpgrade that analyzes the Interfaces table Utilization columns. Upon exiting the add-on dialog box (Figure 7-2), the add-on executable does the following.

- For every interface whose utilization (in the specified utilization column) exceeds a threshold, the circuit capacity is increased by the specified increment until the condition is met.
- A report named *Circuit Upgrade* is generated. The report contains only one section, which is named *List of Upgrades*.
- The MATE GUI opens a new plan file using the same name with -upgrade added as a suffix.

Figure 7-2 Example Circuit Upgrade Dialog Box


Options
Interfaces: Selected in Table (100/100)
Utilization Threshold (%): 90
BW Increment (Mbps): 10000
Utilization Column: UtilMeas
☐ Produce Resized Plan (Default is Report Only)

Tab and Group Customizations

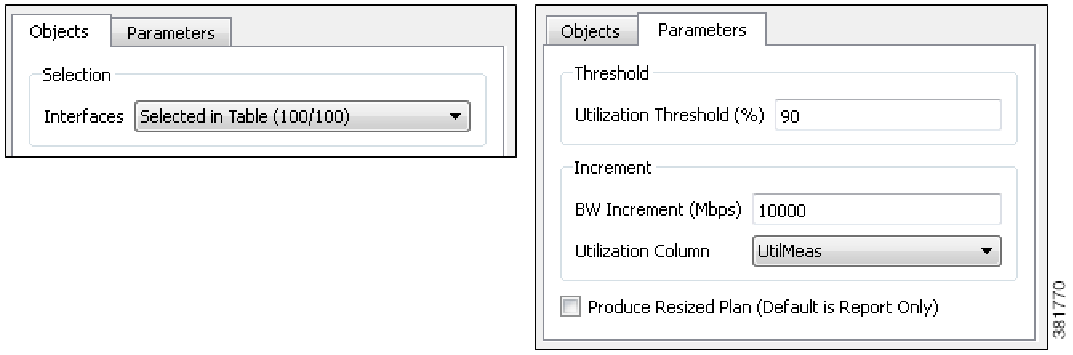
This example <AddOnOptions> table customizes the default sample addon.txt file as follows (Figure 7-3).

- Creates two tabs: Objects and Parameters
- In the Objects tab, creates a group named Selection.
- In the Parameters tab, creates two groups: Threshold and Increment. Another available selection is created to give the option to produce the resided plan file is left ungrouped.

<AddOnOptions>

Tab	Group	OptionName	Prompt	Type	Default	Range	Table
Objects	Selection	interfaces	interfaces	TableSelection			Interfaces
Parameters	Threshold	thresh	Utilization Threshold (%)	Float	90	0;100	
Parameters	Increment	increment	BW Increment (Mbps)	Float	10000		
Parameters	Increment	col	Utilization Column	DropDown	UtilMeas	UtilMeas;UtilSim; WCUtil	
Parameters		resize	Produce Resized Plan (Default is Report Only)	CheckBox	FALSE		

Figure 7-3 Example Customizable Tabs and Groups



Selection
Interfaces: Selected in Table (100/100)

Threshold
Utilization Threshold (%): 90

Increment
BW Increment (Mbps): 10000
Utilization Column: UtilMeas

Produce Resized Plan (Default is Report Only)

Reports Add-Ons

The `$CARIDEN_HOME/addons/Reports` directory contains two sub-directories, each containing an add-on, thus demonstrating how to create sub-menus. Each of these add-ons, Circuit Utilization Report and Traffic Distribution Report, demonstrate two key add-on features.

- Passing information you input to the `mate_jasper` tool to generate a report based on the current plan file, and opening the resulting PDF report. Each report is based on the .jrxml JasperReports template file input to `mate_jasper`. For information on using `mate_jasper`, see the Reporting Tools chapter and `mate_jasper` Help output.
- Using derived column information for the resulting report.

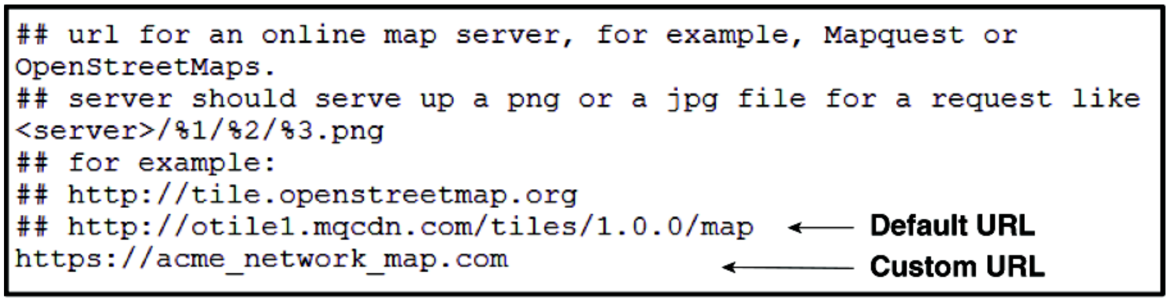
Note To run these example report add-ons, you must have ActivePerl installed if you are running a Windows operating system. (ActivePerl is an open-source Perl scripting language provided by ActiveState.)

Customized Network Plot Views

Map Server

MATE uses an online detailed geographic map database to draw the detailed map background, which is one of the canvas plot options available in the MATE GUI. By default, MATE uses a public server whose location is specified in the `$CARIDEN_HOME/etc/onlinemapurl.txt` file. You can change this URL to use a different server. If you do change it, we recommend that you comment out the line, and add a new one. This way you can easily revert to the default if needed.

For information on viewing a detailed background map and changing plot options, see the *MATE GUI Visualization Guide*.



```
## url for an online map server, for example, Mapquest or
OpenStreetMaps.
## server should serve up a png or a jpg file for a request like
<server>/%1/%2/%3.png
## for example:
## http://tile.openstreetmap.org
## http://otile1.mqcdn.com/tiles/1.0.0/map
https://acme_network_map.com
```

Static Backgrounds

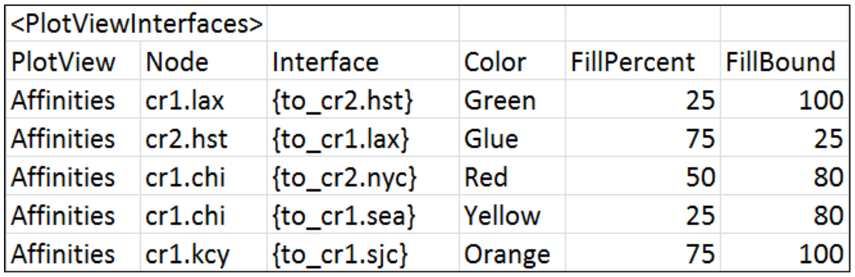
By editing the plan file directly (for example, through an add-on), you can add a user-defined static view to display interfaces with customized color fills and fill percentages. The view then appears as an option in the Network Plot drop-down list located in the far left corner of the visualization toolbar.

To create this custom network plot view, create or edit a `<PlotViewInterfaces>` table (Table 7-5). You must have an entry (row) for each interface you want you want color filled, and each of these interfaces must have the same PlotView name. For example, Figure 7-4 shows four interfaces in the user-defined plot view named Affinities.

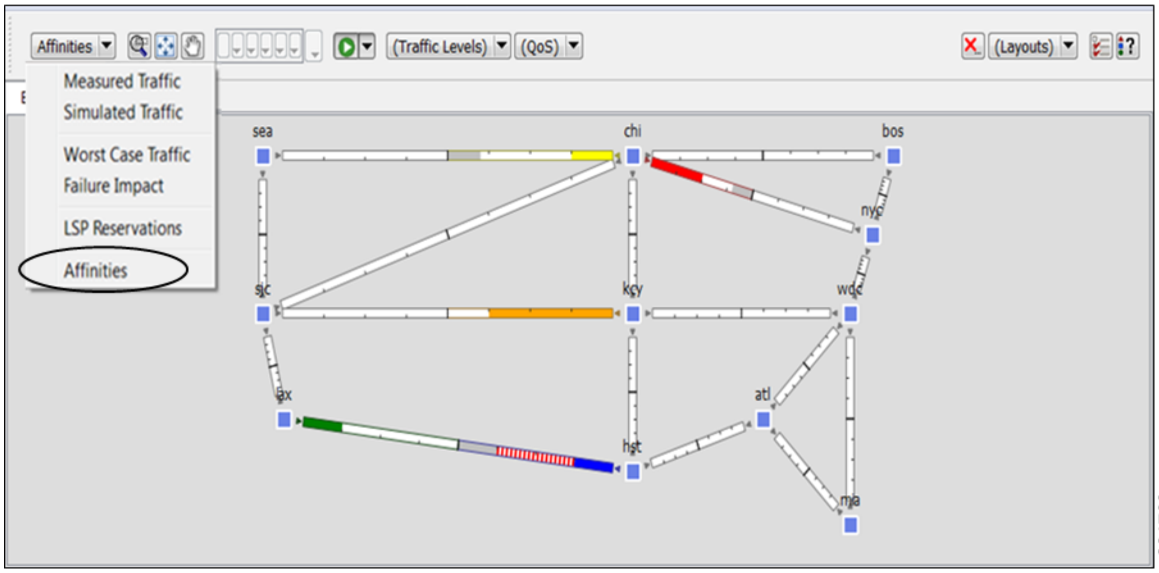
Table 7-5 <PlotViewInterfaces> Columns

Column	Description
PlotView	Name of the plot view; this name appears in the Network Plot drop-down list.
Node	Node containing the interfaces.
Interface	Interface name. **Example:** POS1/1/1
FillPercent	The percent of the interface to be filled with color.
FillBound	Used only if the FillPercent color represents a utilization percentage. FillBound is the maximum utilization percentage possible, or the QoS bound. The remainder of the interface is gray. The default is 100%. • FillBound less FillPercent is colored white. For example, on cr1.chi to cr2.nyc in Figure 7-4, the FillBound is 80 and the FillPercent is 50, thus filling 30 percent of the interface with white. • If the FillBound is greater than the FillPercent, then the difference is filled with red and white stripe, which represents a QoS violation. For example, on the cr2.hst to cr1.lax interface in Figure 7-4, the FillBound is 25 and the FillPercent is 75, thus filling 50% of the interface as a QoS violation.

Figure 7-4 ***Example <PlotViewInterfaces> Table Defining an Affinities View and Coloring Five Interfaces***



<PlotViewInterfaces>					
PlotView	Node	Interface	Color	FillPercent	FillBound
Affinities	cr1.lax	{to_cr2.hst}	Green	25	100
Affinities	cr2.hst	{to_cr1.lax}	Glue	75	25
Affinities	cr1.chi	{to_cr2.nyc}	Red	50	80
Affinities	cr1.chi	{to_cr1.sea}	Yellow	25	80
Affinities	cr1.kcy	{to_cr1.sjc}	Orange	75	100



Reporting Tools

MATE supports two CLI reporting tools. One is `mate_jasper`, which integrates with the JasperReports library. The other is `manage_reports`, which provides a way to manage reports that are generated in a plan file.

JasperReports Tool

JasperReports is an open-source reporting library, which is distributed with MATE. The `mate_jasper` tool provides integration with this library. The tool enables you to produce reports in PDF or HTML format using either a plan file or arbitrary SQLite database file as input and using a .jrxml (JasperReports) file as a template.

You can call the `mate_jasper` tool from an add-on so as to produce reports directly from the GUI. You can find two such example add-ons in the following locations.

- `$CARIDEN_HOME/addons/Reports/highutilcircuits`
- `$CARIDEN_HOME/addons/Reports/trafficdistribution`

For information regarding add-ons, see the Add-Ons and GUI Customizations chapter. For more information on using this tool, see the `mate_jasper` Help output.

Example: This shows the `mate_jasper` tool generating a report named `peak_util.pdf` and placing it in the `$CARIDEN_HOME/addons/peak_traffic/report` directory. It uses the `apac_backbone.pln` plan file as input and the `traffic_template.jrxml` file as a template.

```
mate_jasper -template traffic_template.jrxml -out-file
$CARIDEN_HOME/addons/peak_traffic/report/peak_util.pdf -plan-file apac_backbone.pln
```

Reports in Plan Files

MATE creates reports when certain tools are run, such as Simulation Analysis (`sim_analysis`) and Demand Deduction (`dmd_deduct`). These reports are stored in the plan file on which the tool is run and can be accessed through the Window->Reports menu. Add-ons and other scripts can use this reporting functionality.

The `manage_reports` tool provides access to the reports in a plan file for any of the following functions.

- Insert a report into a plan file
- Delete a report from a plan file

- Extract a report from a plan file
- Rename a report within a a plan file
- List the reports in a plan file
- Print a report in a plan file

Once a report is generated, use the manage_reports tool to insert the report into a plan file. If `manage_reports` is used to extract a report from a plan file, the extracted report uses this same report format.

Plan File Report Format

The plan file report format consists of a directory containing the data and formatting details for each section in the report. The directory must contain a `config.txt` configuration file that includes a `<Sections>` table identifying sections of the report. All columns in this `<Sections>` table are required (Table 8-1).

Optionally, the `config.txt` file contains a `<TableColumns>` table that provides formatting information on how to display the TABLE sections of the report (Table 8-2). If the `<TableColumns>` table is not included, MATE defaults are used.

The name of the report is the name of the directory containing the report. If you create a report in an add-on executable using the `report-dir` option, in which case you cannot name the directory, the report is named after the add-on.

Table 8-1 Columns of the Required <Sections> Table in the config.txt File

Column	Description
Name	Section name that appears on the report.
Type	Output format of the report: HTML, TEXT, or TABLE.
Filename	The file in the report directory containing the source data for the section. • If the ContentType is HTML, this file must contain HTML. • If the ContentType is TEXT, this file must contain plain text. • If the ContentType is TABLE, this file must contain MATE tables. One of those tables must use the same name as the section name (Name column) of this row. This table is used as the contents of this section. **Example:** If the Name column of this row is Interfaces, then this file must contain a table named <Interfaces>.
Index	An integer representing the order in which the sections appear in the report.

Table 8-2 Columns in the Optional <TableColumns> Table in the config.txt File

Column	Description
Table	The name of the section being defined. This name must be the same as a TABLE section name in the <Sections> table.
Column	The name of the column.

Column	Description
DisplayName	Optional: Alternative column name. If not specified, the Column name is used. **Example:** You could set the column name to TraffMinusReservation, and the display name to Traffic Reservation.
Shown	Optional: Whether the column is shown when the plan file opens (T) or hidden (F, default). If a report is imported into the plan using an add-on's `report-dir` option or through `manage_reports`, then the Shown setting is copied from that report.
ToolTip	Optional: The information to be displayed when you hover the cursor over the column heading in the GUI.
Type	Optional: The sorting order for the column: REAL, TEXT (default), INTEGER, or BOOLEAN.
Decimals	Optional: If the column Type is REAL, this value specifies the number of decimal places to use. The default is empty, which means do not constrain the decimal places.

Example Plan File Report Directory and Tables

This example shows a sample report in the `/QuarterlyReport` directory.

```
% ls QuarterlyReport/

config.txt

summary.txt

file.txt

log.txt
```

Table 8-3 shows the `<Sections>` table in the `config.txt` file. Notice that each section name has an associated file that corresponds with the files listed in the above directory. Both the Interfaces and the Nodes sections of this report draw their contents from the same `file.txt` file. The report output lists these sections in the order identified in the Index column.

Table 8-3 Example <Sections> Table

Name	Filename	Type	Index
Summary	summary.txt	HTML	1
Interfaces	file.txt	TABLE	2
Nodes	file.txt	TABLE	3
Log	log.txt	TEXT	4

Table 8-4 shows the columns of the Nodes and Interfaces TABLE sections being defined in the `<TableColumns>` table in the `config.txt` file. All empty fields use the MATE defaults.

Table 8-4 Example <TableColumns> Table

Table	Column	DisplayName	Type	Decimals	Shown	ToolTip
Node	Name					
Node	Status					Upgrade, remove, or keep
Interfaces	Capacity		REAL	2		Required capacity
Interfaces	Node					Interface node
Interfaces	Active		BOOLEA N		F	Is it running?
Interfaces	Interface	Interface Name				

Command-Line Interface

The command-line interface (CLI) tools allow scripting of various functions performed by the MATE GUI, such as import, export, initialization, simulation and optimization. CLI tools can, in some cases, provide additional options that are not available from the GUI, such as network interface tools. This chapter describes the CLI options file and logging levels.

- CLI Options File—Format of the `options.txt` file, which can be used to set default options for use by calls to the tools through the CLI or through the GUI.
- Logging Levels—Options that control the level of logging.

CLI Options File

Options for any tool can be invoked by specifying these options at the CLI, or by specifying the options in an `options.txt` file. By default, each CLI tool looks for this options file in the `$CARIDEN_HOME/etc` directory. A different options file can be specified when calling any of the tools by using the `-options-file` option.

Note Options specified at the command-line take precedence over options specified in the options file.

The Options file format consists of zero or more lines of the following form.

```
<tool-name>:<option>=<option-value>
```

Here `<tool-name>` is the tool name as specified in the table above, `<option>` and `<option-value>` are an option and its value, specific to that tool.

Lines starting with the character `#` are treated as comments and ignored.

If a filename is specified as an parameter to an option in `options.txt` using a relative path or without a path, the path will be relative to the directory in which the tool is called, and not the directory containing the `options.txt` file.

When tools are called through the GUI, the default options displayed in the dialog correspond to the options file settings. Note that changes in the options dialog entries will however remain in effect for the remainder of the session, or until changed.

Options to each command-line tool are available by calling the tool with the `-help` option.

Logging Levels

Many commands have an option (-verbosity) that controls the level of logging.

Table 9-1 Log Levels

Level	Type	Description
0	Fatal	Premature termination of a CLI tool or GUI process under circumstances beyond our control, so output might be corrupt. Minimal effort to handle it elegantly. **Example:** disk full, cannot allocate memory.
1-10	Error	An error occurred so that the results of the tool might be wrong, but there will still be output. Probably premature termination. **Example:** couldn't log in to the seed router for snmp_find_igp_db, bad import format, bad .pln file format.
11-20	Warning	Use of deprecated API, poor use of API, unexpected or undesirable situations. MATE might prefer to have more information to continue, but will make a best guess. **Examples:** Replacing unsupported characters on import, updating SRLG table format on import, unequal capacities on import
21-30	Notice	Results of a tool that can be used to make a decision about what to do next. **Examples:** which routers could not find communities.
31-40	Info	Useful information to further assist users. Can be per object. Timing information for long processes. **Examples:** Metric Optimization completed in 45 seconds, Tried to get community for node A, succeeded
41-50	Debug	Detailed information of flow through system to help trace back an error or warning. SNMP queries: router needed to be retried.
51-60	Trace	Detailed information of flow through system to help trace back an error or warning. **Examples:** individual SNMP queries, Full Mosek output for demand deduction, metric optimization

Related Topics

- *Table Schema and CLI Reference*
- `$CARIDEN_HOME/bin`, where `$CARIDEN_HOME` is the directory containing the Collector server and MATE software (on Linux, the default is `/opt/cariden/software/mate/current`)