Customizing EVC, MPLS and MPLS-TP Policies

This chapter describes how to add custom data fields to the user interface (UI) for a specific policy and how to embed customized command line interface (CLI) templates to EVC, MPLS and MPLS-TP policies. It contains the following sections:

Customizing EVC and MPLS Policies

User Interface Customizations

This section provides a detailed explanation of UI customization features. You can extend policies by adding attributes that you define directly in the policy screen. It also helps you to define new UI attributes in a separate XML file. The new attributes defined in the policy behave in a manner similar to the existing feature, but allow you to define the templates in-line.

This section contains the following topics.

Adding User Interface Groups to Pages

While creating EVC and MPLS policies, a new **Create UI Group** button on every page of the policy enables you to create any number of UI groups on any number of pages on the policy. The name you provide for the UI Group appears as the title of the new section. Groups are used to keep related custom fields together. You can add attributes only to UI groups you create, and not to existing groups in the policy. You can further edit, delete, and reorder attributes within the UI group.

For example, edit an L3VPN policy and navigate to the VRF/VPN screen. You can create a new UI group called **Custom VRF Data** using the **Create UI Group** button.

A new **Create Global UI** button allows you to specify global attributes which are identical across all links of a service request. These global attributes appear on the first page of the service request. Like other attributes, global attributes can also be set as editable or non-editable, and have default values assigned to them.

Adding User Interface Attributes to Groups

Once you create a UI group, the **Create** button and the Settings icon is displayed in the title bar, enabling you to create attributes. Using the **Create** button, you can add custom fields to UI groups as attributes and specify their type. The types that you can specify are:

- String – regular expression and length bounds for validation
- Password – similar to the string attribute but masked in UI
- Integer – requires you to enter numbers and defines a range
- Hexadecimal – requires you to enter hexadecimal values
- Enumeration - drop-down list
- Check box – provides a check box
- IPv4 – IP v4 address, may define range
- IPv6 - IP v6 address, may define range
- Device – pick devices from the inventory – filter by device role
- Device Interface – pick device interfaces from the inventory

The list below contains the common fields that appear for all the UI attributes:

Name – refers to the value of that attribute from a CLI template. For example, if you create an attribute called *cbr*, you can refer to this new attribute in the CLI template using the variable *$cbr*.

Display Name – used as the label for the attribute in the Prime Provisioning UI.

Display Description – displayed when you hover over the tool tip icon for that attribute.

Attributes can be marked Required or Optional. To verify whether optional values are provided, you can use *#if ($my_optional_attribute)* within a CLI template. Attributes marked Required are displayed on the policy and Service Request pages.

Command Line Interface Customizations

You can extend the provisioning logic using **Provisioning CLI customizations section** of the Policy Editor. This uses data fields from both standard service model and UI Customization as input. The product of a standard service model is a custom configlet, which is merged with the standard configlet and sent to the device.

The new CLI templates in the policy are simpler to use, and allow you to create and use CLI customizations without the need for data files. However, when you upgrade using an existing database, it is not possible to convert existing templates into the new form of CLI templates automatically.

This section contains the following topics:

Creating Templates

You can now create and customize templates that consist of the CLIs, which you want to deploy on the devices using **Create CLI Template** button. Templates can refer to the data that you enter in UI groups. When you create a template, in a policy, you can specify:

- **CLI Merging Mode:**

  - External- This mode acts in a manner similar to the Template Manager customizations. It is suitable for adding the configuration that you want Prime Provisioning to generate without modifying any lines in the configuration. Extra configuration is simply sent to the device as is.

  - Combine- This mode acts in a manner similar to the XDE/PAL customizations. It is suitable for changing the configuration that Prime Provisioning generates. The content in the template is merged with the existing configuration, and is also sent to the device only when the current device configuration does not contain the required configuration. In addition to this, the output of the template is audited so that Prime Provisioning can verify the final device configuration and check that the configuration specified in the template is present on the device. Combining depends on the ability of Prime Provisioning's config parser (NOM) to parse the configuration generated by the template. To determine whether this Combine mode can be used with a given template, you need to merely preview the configuration generated for a service request. If NOM does not recognize a line from the template, you will see an error and the line is not included in the final configuration.

  - ExternalWithModify - To modify the customized template attribute value, this CLI merging mode has to be selected.

- **Commission Sequence:** Determines whether the commission CLI is added before or after the configuration that Prime Provisioning generates. To ensure that Prime Provisioning sets up the basic service before it adds the features in the template, select **After**. If the merge mode you select is **Combine** and the commission sequence you select is **After**, the template can overwrite or remove the configuration that Prime Provisioning generated. Instead, if the commission sequence you select is **Before,** it will be Prime Provisioning's configuration that can overwrite that of the template.

- **Commission CLI:** The CLI generated during the commission sequence specified in the Velocity Template Language.

- **Decommission Sequence:** Determines whether the decommission CLI configuration is removed before or after the configuration that Prime Provisioning generates by default. This is the opposite of the Commission Sequence. To control the decommissioning sequence individually, you can create a separate template solely for the purpose of decommissioning.

- **Decommission CLI:** The CLI created during the decommission sequence.

- **Verify:** Click the **Verify** button after entering into CLI. It lists the missed out variable name, which is defined in the policy page but wrongly declared in the CLI section.

Variable Completions for Specifying CLIs

Variable completions are now available while specifying CLIs in templates. This means that you can use Ctrl-Space for completion of variables that you want to enter.

For example, when you type $ and then type Ctrl-Space, the list of all possible variables is displayed and you can select variables directly from this list without having to know them beforehand. Similarly, if you type a prefix to a variable e.g. $SR, then a filtered list of all $SR variables is listed. Further typing while the variable list is visible will further narrow the available options. When only a single option is available, it is selected automatically.

The displayed list of variables consists of:

- customized attributes that you define in the UI groups.

- $SR. standard attributes from the service request section for template attributes. These are the same attributes (names and values) as are defined for the template manager.

- the configuration of the device in the form of an XML document as parsed by NOM is present in the variable $DeviceConfig.

- the definition of the service to be configured as represented in the Database is also available as an XML document in the variable $ServiceIntent. This can be used if you need to get some aspect of the service which is not available in the $SR prefixed variables.

- $system.xpath (<XML>, <XPATH query>).

- $list.xpath (<XML>, <XPATH query>).

- $system.xpathreference (<XML>, <XPATH query>).

- $list.xpathreference (<XML>, <XPATH query>).

- variables that return sections of XML documents queried using XPATH (The $list variants will return a list of matches while the $system variants return the first matched element if any. The reference variants do not create a copy of the parts of the XML document that are returned.).

- $system.log()– logs a message in the http log.

- $system.print()– prints a message in the http.out log.

- $system.throwException() exception name, message (For example, "MPLS.customization", "MPLS service cannot be provisioned because of .."")– This is useful to throw a validation error, No configuration will be deployed. A deployed Service Request deploy that throws an exception transitions to the Invalid state and the exception message is shown in the task log and in the configuration preview.

- $DeviceCredentials. A set of device inventory related attributes for testing properties of the device.

Creating Rules for CLI Templates

While creating CLI templates, you can also determine the type of devices on which the template can be deployed by specifying a set of rules in **Device Support section**. The rules are mainly categorized based on the device type, role type, and operating system. Prime Provisioning deploys the template only when the criteria specified in these rules are fulfilled. When no rules are specified, the template is deployable on all devices.

You can create multiple rules for a given template. For example, you could have one rule for a template to be deployed on only IOS-XR devices of type N-PE; while another rule for the template to be deployed on IOS devices of type U-PE.

Importing and Exporting Customizations

You can export customizations in an XML format and save it using a text editor to create a backup of your customization. It is recommended that you create a backup of your customizations or copy the policy and modify the copy before you modify a policy with existing service requests (see Changing Customizations When a Policy is in Use), so that you can revert back to these customizations by merely importing the same XML document that you saved. To do this, an **Import/Export** button has been provided on the policy creation page. The customizations that you export are displayed in a new browser window from which you can copy the customizations onto a text editor for further use.

By exporting the customization data in an XML text format, you can:

- Apply the same customization to different policies by simply exporting the XML text and importing the same over to a new policy. This is useful when you cannot copy the whole policy for example copying a customization to a policy that is already in use with service requests.

- Edit the order in which the UI groups are placed and also edit the order in which the attributes are displayed within the UI groups.

Changing Customizations When a Policy is in Use

The new UI attributes that you define in policies can be edited even after service requests are defined based on those policies.

To introduce a new capability for only newly created services, it is recommended that you create a new policy with this capability. This can be done by copying an existing policy to create a new one and making the current policy inactive. You can also rename the policies that you copy so that operators can use the same name for the new policy. While creating new services requests, Prime Provisioning only lists the active policies so that you do not select the inactive policies used for existing service requests. This ensures that you do not face any errors while modifying in use policies.

Changes to the attributes in the policy will cause no change to the data in the associated service requests. The changes can only be noticed in the user interface and the way the service request is configured.

To create a backup of the previous version of a customization, refer to Importing and Exporting Customizations. This enables you to revert back to the previously saved version after modifying a policy that has existing service requests.

Some types of changes that you make to a policy can result in undesired changes to a service and hence it's recommended that you review existing service requests before you make these changes to the policy. The types of changes that requires you to review existing service requests are:

- Removing an attribute:
  When an attribute is removed from the policy page with its declaration existing still in the CLI template, an appropriate error with link "**Has errors**" is enabled in the "**Provisioning CLI Customizations**" page. On rolling over the mouse over the "Question mark" icon, the necessary details are shown. Although the removed attribute is no longer displayed and referenced from the provisioning logic and templates, it continues to exist in the service request.The saved value reappears only if you add an attribute with the same name. This behavior is to ensure that the removal of attributes is reversible step. When you remove attributes that continue to be referenced from the provisioning logic or from templates, the templates fail because they are referring to undefined attributes. Thus it is recommended that you first remove all references to the attributes, before you proceed with the removal of these attributes.

- Removing values from the valid range of an attribute:
  This can be done by changing a string validation regular expression, restricting an integer range, and removing values for an enumeration. After you remove these values and then edit the service

request, while retaining the invalid values, you will not be able to save the service request. You will need to either change the value of the attribute or cancel your edit. Thus it is recommended that you edit service requests and not use the invalid values before you change the policy.

- Making an attribute non-editable:
  The attribute can not be modified and will be hidden from the service request page create using the policy with the non-editable attribute. Attribute values modified during service request creation, are not visible to the operated modifying services. To ensure that different service requests do not have different values for the same attribute, it is recommended that all service requests created with the policy contain the same default values before they are marked non-editable. Thus new service requests can only be created with the default values.

Different changes that you make to existing service requests can have varied results. The results are:

- Adding an attribute: The next time you create or edit the service request, this attribute will be added with its default value and can be referred from the templates and provisioning logic.

- Expanding the valid range of an attribute: No changes to the existing service request, however, you can edit the service request to select the new value.

- Editing the default value for an attribute: No change to the existing service requests. Only newly created service requests will take the default value.

- Make an attribute editable: The value can be modified while creating or editing existing services. The attribute will contain its former value.

After you add new attributes to a service, which translate to more lines of configuration, and re-deploy the service, managing the transition is easy since the template will be activated automatically.

However, if you remove template configurations and replace them with new configurations, you need to ensure that you maintain the decommissioning sequence of the old features before you add the new features. Once the service is migrated, you must no longer use the old features. To do this, you can introduce an additional attribute that represents whether the service is migrated or not. This can be used as a condition with an 'if' statement in the template to decide whether an old extension has to be decommissioned or not. For advanced help in migrating from template solutions to customizing policies, you can contact the Advanced Services team.

Customizing MPLS-TP Policies

In MPLS-TP policy editor page, a new accordion "Additional Attributes" has been introduced. This accordion consists of a CLI Customizations text box where you can paste the customization in XML format. The customization data can be applied to either a new policy or to an existing policy. The customization is applied to the policy only when you click **Finish** or else the policy will ignore the changes introduced by the customization.

When you open the MPLS-TP policy again, you can view the customization changes listed as additional attributes in Tunnel Characteristics and Tunnel End-Points accordions. You can change the value of these additional attributes anytime by modifying the customization XML of the policy. These values gets reflected on the service request associated with the policy once you click the Finish button. But you cannot change the value of the additional attributes through UI of a policy. It will not update the associated service request.

The different types of attributes supported in MPLS-TP customization are mentioned below:

- String – regular expression and length bounds for validation

- Integer – requires you to enter numbers and defines a range

- Enumeration - drop-down list

- Check box – provides a check box

- IPv4 – IP v4 address, may define range

You can also mention the different CLI merging modes in the customization XML. The supported modes are: External, Combine, ExternalWithModify. For detailed information on this, refer to the CLI Merging Mode section of Creating Templates.

The Additional Attributes added through customization can be edited or removed from the UI even after the Service Request is defined for the policy but this can be done only by pasting the modified customization XML in the Additional Attributes accordion.