JUNIPER NETWORKS

# Network Configuration Example

## Configuring VPLS Pseudowires on MX Series Devices Using Dynamic Profiles

Release
12.1

Published: 2012-07-19

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net



*Network Configuration Example Configuring VPLS Pseudowires on MX Series Devices Using Dynamic Profiles*
12.1


The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

## END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at http://www.juniper.net/support/eula.html. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

# Table of Contents

## Introduction

This document describes what dynamic profiles do, how they work, and how to configure virtual private LAN service (VPLS) pseudowires using dynamic profiles.

## Dynamic Profiles Overview

A dynamic profile is a set of characteristics, defined in a type of template, that you can use to provide dynamic subscriber access and services for broadband applications. These services are assigned dynamically to interfaces. The **dynamic-profiles** hierarchy appears at the top level of the CLI hierarchy and contains many Juniper Networks configuration statements that you normally define statically.

Dynamic profile statements appear in the following subhierarchies within the **[edit dynamic-profiles]** hierarchy:

- **class-of-service**
- **firewall**
- **interfaces**
- **predefined-variable-defaults**
- **protocols**
- **routing-instances**
- **routing-options**
- **variables**

This topic covers:



### Dynamic Profile Interface Support

You can identify subscribers statically or dynamically. To identify subscribers statically, you can reference a static VLAN interface in a dynamic profile. To identify subscribers dynamically, you create variables for demux interfaces that are dynamically created when subscribers log in.

### What Dynamic Profiles Do

A dynamic profile acts as a kind of template that enables you to create, update, or remove a configuration that includes client access (for example, interface or protocol) or service (for example, CoS) attributes. Using these profiles enables you to consolidate all of the

common attributes of a client (and eventually a group of clients) and apply the attributes simultaneously.

## How Dynamic Profiles Work

After they are created, profiles reside on the router in a profile library. These profiles can contain various configurations. For example, you can create a client network access configuration, a services activation configuration, or both. When a router interface receives a join message from a client, the router applies the values configured in the specified dynamic profile to that router interface. The profile can contain interface, class-of-service (CoS), and protocol values that are applied directly to the interface. In addition, the dynamic profile can call input or output firewall filters that reside outside of the dynamic profiles hierarchy.

## Dynamic Profile Version Creation

You can create new versions of dynamic profiles that are currently in use by subscribers. Dynamic profile version creation is enabled at the **[system]** hierarchy level. When enabled, you can create multiple versions of any dynamic profiles on the router. Any subscriber that logs in following a dynamic profile modification uses the latest version of the dynamic profile. Subscribers that are already active continue to use the older version of the dynamic profile until they log out or their session terminates.

When creating versions of dynamic profiles, keep the following in mind:

- You must enable or disable dynamic profile version creation before creating or using any dynamic profiles on the router. Enabling or disabling dynamic profile version creation after dynamic profiles are configured is not supported.

  NOTE: **Before you can enable or disable dynamic profile version creation for a router on which any dynamic profiles are configured, you must first remove all dynamic profiles from the router configuration.**

- Each version of a dynamic profile is stored in the profile database as a new profile.
- The name of the new profile version is derived by appending a four-character tag string to the original base dynamic profile name. This tag string contains two dollar sign ($) characters to identify the version field of the profile name. These two characters are followed by two numerical characters that represent the version number of the dynamic profile (for example, 01).
- The dynamic profile that you modify is always stored as the latest version. You cannot create a modified dynamic profile and save it as an earlier version. For example, if you modify version three of a dynamic profile, it is saved as version four.
- You can only modify the latest version of a dynamic profile.
- If the dynamic profile version that you modify is not in use by any subscriber, the profile is overwritten with committed changes without creating a new version.
- You can create a maximum of 10 versions of each dynamic profile.

- If all 10 versions of a dynamic profile already exists, any modification to the dynamic profile results in modifying the latest version of that profile (that is, version $$10). If this version is in use, any modification attempt fails upon commit.
- You can delete a dynamic profile only when it is not in use.
- The dynamic profile version feature supports graceful restart and unified ISSU.

### Dynamic Profile Semantic Checking

Variables are applied to dynamic profiles dynamically and cannot be checked with existing CLI checks. Semantic checking validates some variables in dynamic profiles to help identify potential configuration errors.

Semantic checks are performed during commit and during profile instantiation. Commit time checks ensure that variables appear in the correct location within the dynamic profile. Checks performed before profile instantiation ensure that the values that replace the variables are correct. The checks performed on the values include the following:

- Range validation
- Variable type validation
- Existence of variables where they are mandatory
- Variable matching to regular expressions

A commit time check failure results in an error message being displayed and logged in **/var/log/messages** and the commit not taking place. An instantiation failure results in an error being logged in **/var/log/messages** and the profile instantiation failing.

**Related Documentation**

- Configuring a Basic Dynamic Profile
- Configuring a Dynamic Profile for Client Access
- Configuring a Dynamic Profile for Various Levels of Services
- Enabling Dynamic Profiles to use Multiple Versions
- Dynamic Variables Overview
- Subscriber Interface Overview

## Dynamic Profiles for VPLS Pseudowires

A router often has two types of interfaces:

- Static interfaces, which are configured before the router is booted
- Dynamic interfaces, which are created after the router is booted and while it is running

A virtual private LAN service (VPLS) pseudowire interface (such as **lsi.1048576**) is dynamically created by the system. Therefore, the logical interface unit number for the VPLS pseudowire is not available in advance to configure characteristics such as VLAN identifiers and other parameters. As a result, certain virtual local area network (VLAN)

manipulation features that are easily applied to static interfaces (such as **xe-**, **ge-**, and so on) are either not supported on dynamic interfaces or supported in a nonstandard method.

However, on MX Series routers, there is another configuration method that dynamic interfaces can use to determine their VLAN parameters when they are created by a running router: dynamic profiles. A dynamic profile is a conceptual container that includes parameters associated with a dynamic entity, parameters whose values are not known at the time the entity is configured.

A dynamic profile acts as a kind of template that enables you to create, update, or remove a configuration that includes client access (for example, interface or protocol) or service (for example, CoS) attributes. Using these profiles you can consolidate all of the common attributes of a client (and eventually a group of clients) and apply the attributes simultaneously. The router contains several predefined variables that enable dynamic association of interfaces and logical units to incoming subscriber requests. While configuring dynamic profile, use the variable **$junos-interface-ifd-name**for a dynamic physical interface and **$junos-underlying-unit-number**for a dynamic logical interface (unit). When a client accesses the router, the dynamic profile configuration replaces the predefined variable with the actual interface name or unit value for the interface the client is accessing.

Dynamic profiles for VPLS are supported only on MX Series routers. For more information about dynamic profiles, see the *Junos OS Subscriber Access Configuration Guide*

**Related Documentation**

- Ethernet Networking
- Example: Configuring VPLS Pseudowires with Dynamic Profiles—Basic Solutions
- Example: Configuring VPLS Pseudowires with Dynamic Profiles—Complex Solutions

## Use Cases for Dynamic Profiles for VPLS Pseudowires

A dynamic profile is a set of characteristics, defined in a type of template, that you can use to provide dynamic subscriber access and services for broadband applications. These services are assigned dynamically to interfaces. You can use dynamic profiles to configure the VLAN parameters of the dynamic interfaces on MX Series routers.

Two use cases for configuring VPLS pseudowires with dynamic profiles are:

- **Configuring an extra VLAN tag onto pseudowire traffic** — This is a common scenario where all the traffic received from a customer edge (CE) interface needs an additional VLAN tag toward the core. In such cases, you can use dynamic profiles at ingress and egress to control the pseudowire behavior. You can apply dynamic profiles to receive the desired frames, set the additional VLAN tag for these frames, and send these tagged frames with the desired VLAN identifier.
- **Configuring a VPLS pseudowire as a trunk interface** — This is another common scenario where the requirement is to accept traffic from a particular source and to route the traffic based on specific criteria. Dynamic profiles can be used to create

multiple pseudowire trunk interfaces to accept the traffic based on specific VLAN identifiers, and to route the accepted traffic to the desired destination.

## Example: Configuring the Router to Push an Extra VLAN Tag onto Pseudowire Traffic Using Dynamic Profiles

This example shows how dynamic profiles can be used to add an extra VLAN tag to the traffic received from a customer edge (CE) interface toward the network core.



### Requirements

This example uses the following hardware and software components:

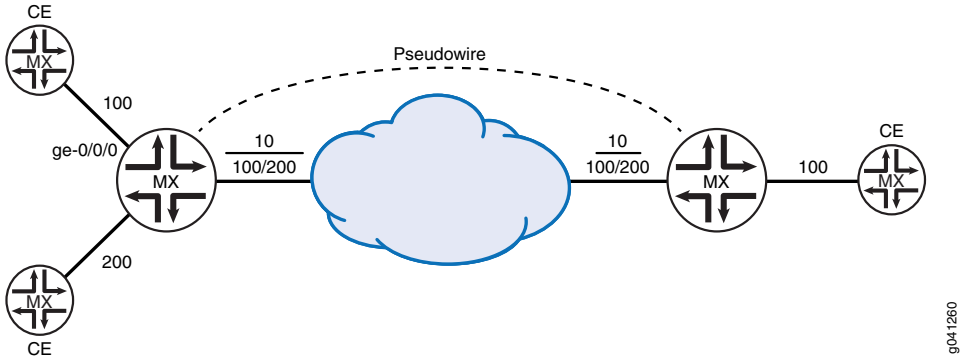
- MX Series 3D Universal Edge Routers
- Junos OS Release 9.3 or later

### Overview and Topology

A dynamic profile is a set of characteristics, defined in a type of template, that you can use to provide dynamic subscriber access and services for broadband applications. These services are assigned dynamically to interfaces. You can use dynamic profiles at ingress and egress to control the pseudowire behavior.

The topology used in this example for dynamically pushing an extra VLAN tag on to a pseudowire is shown in Figure 1 on page 5. On the ingress, the pseudowire receives dual-tagged frames with an inner VLAN ID 100, and an outer VLAN ID 10. All other frames are dropped. The outer tag is stripped at the interface. On the egress, the dual-tagged frames are transmitted with the outer vlan tag set to 10.

Figure 1: Pushing an Extra VLAN Tag on a Pseudowire



## Configuration

**CLI Quick Configuration**

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set dynamic-profiles pw-tagged interfaces $junos-interface-ifd-name unit
  $junos-underlying-interface-unit vlan-tags inner 100 outer 10
set routing-instances red instance-type vpls vlan-id 100 interface ge-0/0/0
set routing-instances red protocols vpls associate-profile pw-tagged
```

### Pushing an Additional VLAN Tag Over the Pseudowire

**Step-by-Step Procedure**

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see Using the CLI Editor in Configuration Mode in the Junos OS CLI User Guide.

NOTE: This is not a complete router configuration.

To push an additional VLAN tag over the pseudowire:

1. Create a dynamic profile named **pw-tagged**.

   ```
   user@host#set dynamic-profiles pw-tagged
   ```

2. Specify the variables **$junos-interface-ifd-name** and **$junos-underlying-interface-unit** to be used by the router to match the interface name and the unit value of the receiving interface.

   ```
   [edit dynamic-profiles pw-tagged]
   user@host# set interfaces $junos-interface-ifd-name unit
     $junos-underlying-interface-unit
   ```

3. Specify the VLAN tag to receive dual-tagged frames with an inner VLAN ID 100 and an outer VLAN ID 10.

   ```
   [edit dynamic-profiles pw-tagged interfaces "$junos-interface-ifd-name" unit
     "$junos-underlying-interface-unit"]
   user@host# set vlan-tags inner 100 outer 10
   ```

4. Create a routing instance named **red**.

   ```
   [edit ]
   user@host# set routing-instances red instance-type vpls vlan-id 100 interface
     ge-0/0/0
   ```

5. Associate the profile with the routing instance.

   ```
   [edit routing-instances red]
   user@host# set protocols vpls associate-profile pw-tagged
   ```

## Verification

### Verifying the Dynamic Profile Configuration

**Purpose** Make sure that the dynamic profile is configured correctly.

**Action** From configuration mode, enter the **show dynamic-profiles** command.

**user@host# show dynamic-profiles**

```
pw-tagged {
    interfaces {
        "$junos-interface-ifd-name" {
            unit "$junos-underlying-interface-unit" {
                vlan-tags inner 100 outer 10;
            }
        }
    }
}
```

From configuration mode, enter the **show routing-instances** command.

**user@host# show routing-instances**

```
red {
    instance-type vpls;
    interface ge-0/0/0.0;
    vlan-id 100;
    protocols {
        vpls {
            associate-profile pw-tagged;
        }
    }
}
```

**Meaning** In the ingress direction, the pseudowire receives dual-tagged frames with an inner VLAN ID 100, and an outer VLAN ID 10. All other frames are dropped and the outer tag is stripped at the interface. In the egress direction, dual-tagged frames are transmitted with an inner VLAN ID 100 and an outer VLAN ID 10.

**Related Documentation**

- Dynamic Profiles Overview on page 1
- Dynamic Profiles for VPLS Pseudowires on page 3
- Use Cases for Dynamic Profiles for VPLS Pseudowires on page 4
- Example: Configuring a VPLS Pseudowire as a Trunk Interface Using Dynamic Profiles on page 8

## Example: Configuring a VPLS Pseudowire as a Trunk Interface Using Dynamic Profiles

This example shows how dynamic profiles can be used to create multiple pseudowire trunk interfaces to accept the traffic based on specific VLAN identifiers, and to route the accepted traffic to the desired destination.



### Requirements

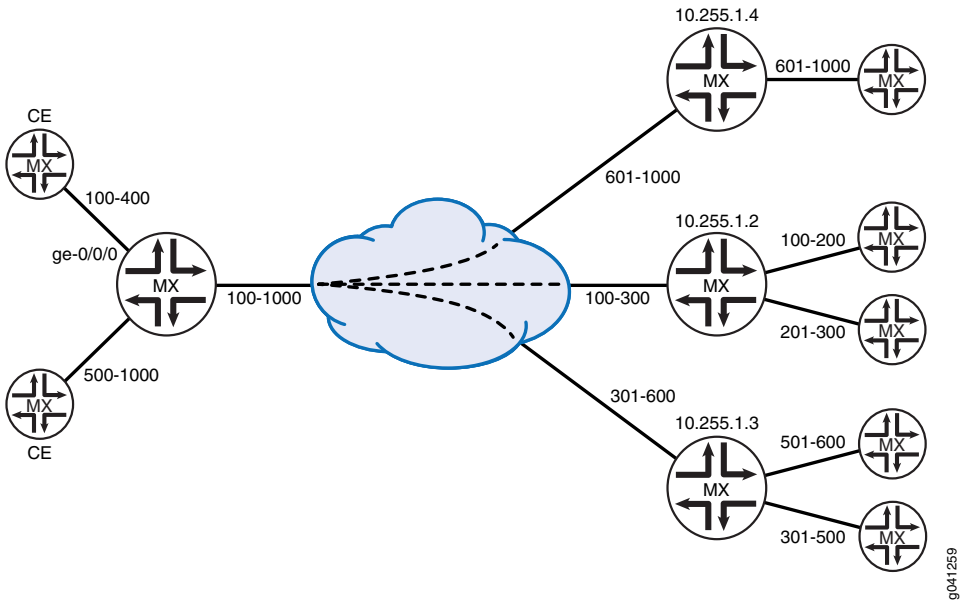
This example uses the following hardware and software components:

- MX Series 3D Universal Edge Routers
- Junos OS Release 9.3 or later

### Overview and Topology

The topology used in this example for configuring VPLS pseudowires as trunk interfaces is shown in Figure 2 on page 8. Three trunk interfaces, **pw-trunk-1** with VLAN ID 100–300, **pw-trunk-2** with VLAN ID 301–600, and **pw-trunk-3** with VLAN ID 601—100, are configured.

Figure 2: Using Pseudowires as Trunk Interfaces



## Configuration

Configuring VPLS pseudowires as trunk interfaces using dynamic profiles involves performing these tasks:

- Creating a Dynamic Profile on page 9
- Creating a Routing Instance with a Bridge Domain on page 11
- Associating Dynamic Profiles with a Routing Instance on page 11

**CLI Quick Configuration**

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set dynamic-profiles pw-trunk-1 interfaces $junos-interface-ifd-name unit
  $junos-underlying-interface-unit family bridge interface-mode trunk vlan-id-list
  [100-300]
set dynamic-profiles pw-trunk-2 interfaces $junos-interface-ifd-name unit
  $junos-underlying-interface-unit family bridge interface-mode trunk vlan-id-list
  [301-600]
set dynamic-profiles pw-trunk-3 interfaces $junos-interface-ifd-name unit
  $junos-underlying-interface-unit family bridge interface-mode trunk vlan-id-list
  [601-1000]
set routing-instances red instance-type virtual-switch bridge-domains vlan-100 vlan-id
  100
set routing-instances red instance-type virtual-switch bridge-domains vlan-200 vlan-id
  200
set routing-instances red instance-type virtual-switch bridge-domains vlan-1000 vlan-id
  1000
set routing-instances red protocols vpls neighbor 10.255.1.2 associate-profile pw-trunk-1
set routing-instances red protocols vpls neighbor 10.255.1.3 associate-profile pw-trunk-2
set routing-instances red protocols vpls neighbor 10.255.1.4 associate-profile pw-trunk-3
```

### Creating a Dynamic Profile

**Step-by-Step Procedure**

Perform the following steps to create the **pw-trunk-1** dynamic profile. This dynamic profile is then associated with the routing instance to filter the ingress and egress frames.

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see Using the CLI Editor in Configuration Mode in the Junos OS CLI User Guide.

NOTE: This is not a complete router configuration.

1. Create a dynamic profile named **pw-trunk-1**.

```
[edit]
user@host# set dynamic-profiles pw-trunk-1
```

2. Specify the variables **$junos-interface-ifd-name** and **$junos-underlying-interface-unit** to be used by the router to match the interface name and the unit value of the receiving interface.

```
[edit dynamic-profiles pw-trunk-1]
user@host# set interfaces $junos-interface-ifd-name unit
  $junos-underlying-interface-unit
```

3. Specify the protocol family, interface mode, and the trunk mode VLAN membership for the interface.

```
[edit dynamic-profiles pw-trunk-1 interfaces "$junos-interface-ifd-name" unit
  "$junos-underlying-interface-unit"]
user@host# set family bridge family bridge interface-mode trunk vlan-id-list
  [100-300]
```

**Step-by-Step Procedure**

Perform the following steps to create the **pw-trunk-2** dynamic profile. This dynamic profile is then associated with the routing instance to filter the ingress and egress frames.

1. Create a dynamic profile named **pw-trunk-2**.

```
[edit ]
user@host# set dynamic-profiles pw-trunk-2
```

2. Specify the variables **$junos-interface-ifd-name** and **$junos-underlying-interface-unit** to be used by the router to match the interface name and the unit value of the receiving interface.

```
[edit dynamic-profiles pw-trunk-2]
user@host# set interfaces $junos-interface-ifd-name unit
  $junos-underlying-interface-unit
```

3. Specify the protocol family, interface mode, and the trunk mode VLAN membership for the interface.

```
[edit dynamic-profiles pw-trunk-2 interfaces "$junos-interface-ifd-name" unit
  "$junos-underlying-interface-unit"]
user@host# set family bridge family bridge interface-mode trunk vlan-id-list
  [301-600]
```

**Step-by-Step Procedure**

Perform the following steps to create the **pw-trunk-3** dynamic profile. This dynamic profile is then associated with the routing instance to filter the ingress and egress frames.

1. Create a dynamic profile named **pw-trunk-3**.

```
[edit]
user@host# set dynamic-profiles pw-trunk-3
```

2. Specify the variables **$junos-interface-ifd-name** and **$junos-underlying-interface-unit** to be used by the router to match the interface name and the unit value of the receiving interface.

```
[edit dynamic-profiles pw-trunk-3]
user@host# set interfaces $junos-interface-ifd-name unit
  $junos-underlying-interface-unit
```

3. Specify the protocol family, interface mode, and the trunk mode VLAN membership for the interface.

```
[edit dynamic-profiles pw-trunk-3 interfaces "$junos-interface-ifd-name" unit
  "$junos-underlying-interface-unit"]
user@host# set family bridge family bridge interface-mode trunk vlan-id-list
  [601-1000]
```

### Creating a Routing Instance with a Bridge Domain

**Step-by-Step Procedure**

In this example, the routing instance is created with three bridge domains: **vlan-100**, **vlan-200**, **vlan-300**. Only the packets with VLAN IDs that match the VLAN ID configured for a bridge domain are forwarded within the bridge domain.

NOTE: The VLAN ID configuration supports a range of VLAN IDs but for brevity the example only shows one VLAN ID configured for each bridge domain.

To configure a routing instance with bridge domains:

1. Create a routing instance named **red**.

```
[edit ]
user@host# set routing-instances red instance-type virtual-switch
```

2. Configure the bridge domains on the routing instance.

```
[edit routing-instances red]
user@host# set bridge-domains vlan-100 vlan-id 100
user@host# set bridge-domains vlan-200 vlan-id 200
user@host# set bridge-domains vlan-1000 vlan-id 1000
```

### Associating Dynamic Profiles with a Routing Instance

**Step-by-Step Procedure**

By associating the dynamic profiles with a routing instance, you can filter the frames on egress and ingress.

To associate dynamic profiles with a routing instance, configure the routing protocol to accept the VLANs. In this example, the dynamic profile **pw-trunk-1** is associated with neighbor 10.255.1.2, the dynamic profile **pw-trunk-2** is associated with neighbor 10.255.1.3, and the dynamic profile **pw-trunk-3** is associated with neighbor 10.255.1.4.

```
[edit routing-instances red]
user@host# set protocols vpls neighbor 10.255.1.2 associate-profile pw-trunk-1
user@host# set protocols vpls neighbor 10.255.1.3 associate-profile pw-trunk-2
user@host# set protocols vpls neighbor 10.255.1.4 associate-profile pw-trunk-3
```

## Verification

### Verifying the Trunk Interface

**Purpose** Make sure that the settings for the trunk interface are correct.

**Action** From operational mode, enter the following commands:

```
user@host> show dynamic-profiles

  pw-trunk-1 {
    interfaces {
      "$junos-interface-ifd-name" {
        unit "$junos-underlying-interface-unit" {
          family-bridge {
          interface-mode trunk;
          vlan-id-list 100-300;
          }
        }
      }
    }
  }
  pw-trunk-2 {
    interfaces {
      "$junos-interface-ifd-name" {
        unit "$junos-underlying-interface-unit" {
          family-bridge {
          interface-mode trunk;
          vlan-id-list 301-600;
          }
        }
      }
    }
  }
  pw-trunk-3 {
    interfaces {
      "$junos-interface-ifd-name" {
        unit "$junos-underlying-interface-unit" {
          family-bridge {
          interface-mode trunk;
          vlan-id-list 601-1000;
          }
        }
      }
    }
  }
```

```
user@host> show routing-instances

  red {
    instance-type virtual-switch;
      protocols {
      vpls {
        neighbor 10.255.1.2 {
          associate-profile pw-trunk-1;
        }
        neighbor 10.255.1.3 {
          associate-profile pw-trunk-2;
        }
        neighbor 10.255.1.4 {
          associate-profile pw-trunk-3;
        }
      }
    }
```

```
    bridge-domains {
      vlan-100 {
        vlan-id 100;
      }
      vlan-1000 {
        vlan-id 1000;
      }
      vlan-200 {
        vlan-id 200;
      }
    }
  }
```

**Meaning** On the ingress, the pseudowire receives frames with the VLAN ID 100 to 1000. All other frames are dropped. On the egress, the pseudowire sends frames with VLAN ID 100-300 to a particular neighbor, 301-600 to the second neighbor, and 601-1000 to the third neighbor. All other frames are dropped.

**Related Documentation**

- Dynamic Profiles Overview on page 1
- Dynamic Profiles for VPLS Pseudowires on page 3
- Use Cases for Dynamic Profiles for VPLS Pseudowires on page 4
- Example: Configuring the Router to Push an Extra VLAN Tag onto Pseudowire Traffic Using Dynamic Profiles on page 5