Chapter 24

Configuring Link Services and Multilink Interfaces

Multilink Point-to-Point Protocol (MLPPP) enables you to bundle multiple PPP links into a single logical link. *Multilink Frame Relay* (MLFR) enables you to bundle multiple Frame Relay data-link connection identifiers (DLCIs) into a single logical link. MLPPP and MLFR provide service option granularity between low-speed T1 and E1 services and higher-speed T3 and E3 services. You use MLPPP and MLFR to increase bandwidth in smaller, more cost-effective increments. In addition to providing incremental bandwidth, bundling multiple links can add a level of fault tolerance to your dedicated access service, because you can implement bundling across multiple Physical Interface Cards (PICs), protecting against the failure of any single PIC.

The JUNOS software supports four Multilink Protocol (MP)-based services PICs: the Adaptive Services PIC, Multilink Services PIC, Link Services PIC, and Voice Services PIC. For more information about the Adaptive Services PIC, see "Configuring Adaptive Services Interfaces" on page 157. For more information about the Voice Services PIC, see "Configuring Voice Services Interfaces" on page 583.

At the logical unit level, the Multilink Services, Link Services, and Voice Services PICs support the MLPPP and MLFR FRF.15 encapsulation types. At the physical interface level, the Adaptive Services, Link Services, and Voice Services PICs also support the MLFR FRF.16 encapsulation type.

MLPPP is supported on interface types ls-*fpc*/*pic*/*port*, lsq-*fpc*/*pic*/*port*, ml-*fpc*/*pic*/*port*, and vsp-*fpc*/*pic*/*port*.

MLFR Frame Relay Forum (FRF)15 is supported on interface types ml-*fpc*/*pic*/*port*, ls-*fpc*/*pic*/*port*, and vsp-*fpc*/*pic*/*port*. For MLFR FRF.15, multiple permanent virtual circuits (PVCs) are combined into one aggregated virtual circuit (AVC). This provides fragmentation over multiple PVCs on one end and reassembly of the AVC on the other end.

MLFR FRF.16 is supported on the AS PIC interface, lsq-*fpc*/*pic*/*port* and the Link Services PIC interface, ls-*fpc*/*pic*/*port*:*channel*, which denotes a single MLFR FRF.16 bundle. For MLFR FRF.16, multiple links are combined to form one logical link. Packet fragmentation and reassembly occur on a per-VC basis. Each bundle can support multiple VCs. Link Services PICs can support up to 256 DLCIs per MLFR FRF.16 bundle. The physical connections must be E1, T1, channelized DS3-to-DS1, channelized DS3-to-DS0, channelized E1, channelized STM1, or channelized intelligent queuing (IQ) interfaces. When you bundle channelized interfaces using the link services interface, the channelized interfaces require M-series Enhanced FPCs.

The standards for MLPPP, MLFR FRF.15, and MLFR FRF.16 are defined in the following specifications:

RFC 1990, *The PPP Multilink Protocol (MP)*

Frame Relay Forum, FRF.15, *End-to-End Multilink Frame Relay Implementation Agreement*

Frame Relay Forum, FRF.16.1, *Multilink Frame Relay UNI/NNI Implementation Agreement*

To configure multilink and link services logical interface properties, include the following statements:

```
interface-name {
    unit logical-unit-number {
        dlci dlci-identifier;
        drop-timeout milliseconds;
        encapsulation type;
        fragment-threshold bytes;
        interleave-fragments;
        minimum-links number;
        mrru bytes;
        multicast-dlci dlci-identifier;
        short-sequence;
        family family {
            address address {
                destination address;
            }
            bundle interface-name;
            }
        }
    }
}
```

You can include these statements at the following hierarchy levels:

[edit interfaces]

[edit logical-routers *logical-router-name* interfaces]

To configure link services physical interface properties, include the mlfr-uni-nni-bundle-options statement at the [edit interfaces *interface-name*] hierarchy level:

```
[edit interfaces interface-name]
encapsulation type;
mlfr-uni-nni-bundle-options {
    acknowledge-retries number;
    acknowledge-timer milliseconds;
    action-red-differential-delay (disable-tx | remove-link);
    drop-timeout milliseconds;
    fragment-threshold bytes;
    hello-timer milliseconds;
    lmi-type (ansi | itu);
    minimum-links number;
    mrru bytes;
    n391 number;
    n392 number;
    n393 number;
    red-differential-delay milliseconds;
    t391 number;
    t392 number;
    yellow-differential-delay milliseconds;
}
```

This chapter is organized as follows:

For examples of multilink and link services interface configuration, see the following sections:

Examples: Configuring Multilink Interfaces on page 436

Examples: Configuring Link Services Interfaces on page 438

Configuring Logical Interface Properties

You configure multilink and link services interface properties at the logical unit level. For information about default settings for multilink and link services logical interface properties, see “Default Settings for Logical Interfaces” on page 413.

You can configure the following multilink and link services logical interface properties:

For general information about logical unit properties, see “Configuring Logical Interface Properties” on page 93. For general information about family inet properties, see “Configuring Protocol Family and Address Interface Properties” on page 107. For information about multilink and link services properties you configure at the family inet hierarchy level, see “Configuring Bundles” on page 428.

Default Settings for Logical Interfaces

Table 35 lists the default settings for multilink and link services logical interface statements, together with the other permitted values or value ranges.

Table 35: Multilink and Link Services Logical Interface Statements

Option	Default Value	Possible Values
Data-link connection identifier (DLCI)	None	16 through 1022
Drop timeout period	0 milliseconds	0 through 2000 milliseconds
Encapsulation	For multilink interfaces, multilink-ppp For link services interfaces, multilink-frame-relay-end-to-end	multilink-frame-relay-end-to-end multilink-ppp
Fragmentation threshold	0 bytes	128 through 16,320 bytes (*N*x64)
Interleave fragments	disabled	enabled, disabled
Minimum links	1 link	1 through 8 links
MRRU (maximum received reconstructed unit)	1504 bytes	1500 through 4500 bytes
Sequence ID format for ML-PPP	24 bits	12 or 24 bits
Sequence ID format for MLFR FRF.15 and FRF.16	12 bits	12 bits

For statements that apply to link services physical interfaces only, see Table 36 on page 423.

Configuring a Link Services Point-to-Point DLCI

For link services interfaces only, you can configure multiple DLCIs for each MLFR FRF.16 or MLPPP bundle. A channelized interface, such as ls-1/1/1:0, denotes a single MLFR FRF.16 bundle. To configure a DLCI, include the dlci statement:

dlci *dlci-identifier*;

You can include this statement at the following hierarchy levels:

[edit interfaces *interface-name* unit *logical-unit-number*]

[edit logical-routers *logical-router-name* interfaces *interface-name* unit *logical-unit-number*]

The DLCI identifier is a value from 16 through 1022. Numbers 1 through 15 are reserved for future use.

When you configure point-to-point connections, the MTU sizes on both sides of the connection must be the same.

DLCIs are not supported on multilink interfaces.

Configuring a Link Services Multicast-Capable DLCI

For link services interfaces only, you can configure multiple multicast-capable DLCIs for each MLFR FRF.16 bundle. A channelized interface, such as ls-1/1/1:0, denotes a single MLFR FRF.16 bundle. By default, Frame Relay connections assume unicast traffic. If your Frame Relay switch performs multicast replication, you can configure the link services connection to support multicast traffic by including the multicast-dlci statement:

```
multicast-dlci dlci-identifier;
```

You can include this statement at the following hierarchy levels:

[edit interfaces *interface-name* unit *logical-unit-number*]

[edit logical-routers *logical-router-name* interfaces *interface-name* unit *logical-unit-number*]

The DLCI identifier is a value from 16 through 1022 that defines the Frame Relay DLCI over which the switch expects to receive multicast packets for replication.

You can configure multicast support only on point-to-multipoint link services connections. Multicast-capable DLCIs are not supported on multilink interfaces.

If keepalives are enabled, causing the interface to send Local Management Interface (LMI) messages during idle times, the number of possible DLCI configurations is limited by the maximum transmission unit (MTU) selected for the interface. For more information, see "Configuring Link Services Keepalive Settings on Frame Relay LMI" on page 426.

Configuring a Drop Timeout Period

By default, the drop timeout parameter is disabled. You can configure a drop timeout value to provide a recovery mechanism if individual links in the multilink or link services bundle drop one or more packets. Drop timeout is not a differential delay tolerance setting and does not limit the overall latency. We recommend setting a drop timeout value significantly larger than the expected differential delay across the links; this way, the timeout period elapses when there is actual packet loss, and not under normal jitter conditions. You can configure differential delay tolerance for link services interfaces only. For more information, see "Configuring Link Services Differential Delay" on page 425.

To configure the drop timeout value, include the drop-timeout statement:

```
drop-timeout milliseconds;
```

You can include this statement at the following hierarchy levels:

[edit interfaces *interface-name* unit *logical-unit-number*]

[edit logical-routers *logical-router-name* interfaces *interface-name* unit *logical-unit-number*]

For link services interfaces, you also can configure the drop timeout value at the physical interface level by including the drop-timeout statement at the [edit interfaces *interface-name* mlfr-uni-nni-bundle-options] hierarchy level:

[edit interfaces *interface-name* mlfr-uni-nni-bundle-options]
drop-timeout *milliseconds*;

The drop timeout has a duration from 0 through 2000 milliseconds. Values less than 5 milliseconds are not recommended; a value of 0 disables the timeout.

NOTE: For multilink or link services interfaces, if a packet or fragment encounters an error condition and is destined for a disabled bundle or link, it does not contribute to the dropped packet and frame counts in the per-bundle statistics. The packet is counted under the global error statistics and is not included in the global output bytes and output packet counts. This unusual accounting happens only if the error conditions are generated inside the multilink interface, not if the packet encounters errors on the wire or elsewhere in the network.

Configuring Logical Interface Encapsulation

Multilink and link services interfaces support the following logical interface encapsulation types:

MLPPP

MLFR End-to-End

By default, the logical interface encapsulation type on multilink interfaces is MLPPP. The default logical interface encapsulation type on link services interfaces is MLFR End-to-End. For more information, see "Configuring the Encapsulation on a Logical Interface" on page 105.

You can configure physical interface encapsulation on link services interfaces. For more information, see "Configuring Link Services Physical Interface Encapsulation" on page 423.

To configure multilink or link services encapsulation, include the encapsulation statement:

encapsulation *type*;

You can include this statement at the following hierarchy levels:

[edit interfaces *interface-name* unit *logical-unit-number*]

[edit logical-routers *logical-router-name* interfaces *interface-name* unit *logical-unit-number*]

You must also configure the T1, E1, or DS0 physical interface with the same encapsulation type.

Configuring a Fragmentation Threshold

By default, the fragmentation threshold parameter is disabled. For interfaces with MLPPP encapsulation only, you can configure a fragmentation threshold to set a maximum size for packet payloads transmitted across the individual links within the multilink circuit. The software splits any incoming packet that exceeds the fragmentation threshold into smaller units suitable for the circuit size; it reassembles the fragments at the other end, but does not affect the output traffic stream. The threshold value affects the payload only; it does not affect the MLPPP header.

NOTE: To ensure proper load balancing:

For Link Services MLFR (FRF.15 and FRF.16) interfaces, do not include the fragmentation-threshold statement in the configuration.

For MLPPP interfaces, do not include both the fragmentation-threshold statement and the short-sequence statement in the configuration.

For MLFR (FRF.15 and FRF.16) and MLPPP interfaces, if the MTU of links in a bundle is less than the bundle MTU plus encapsulation overhead, then fragmentation is automatically enabled. You should avoid this situation for MLFR (FRF.15 and FRF.16) interfaces and for MLPPP interfaces on which short-sequencing is enabled.

To configure a fragmentation threshold value, include the fragment-threshold statement:

fragment-threshold *bytes*;

You can include this statement at the following hierarchy levels:

[edit interfaces *interface-name* unit *logical-unit-number*]

[edit logical-routers *logical-router-name* interfaces *interface-name* unit *logical-unit-number*]

For link services interfaces, you also can configure a fragmentation threshold value at the physical interface level by including the fragment-threshold statement at the [edit interfaces *interface-name* mlfr-uni-nni-bundle-options] hierarchy level:

[edit interfaces *interface-name* mlfr-uni-nni-bundle-options]
fragment-threshold *bytes*;

The maximum fragment size can be from 128 through 16,320 bytes. The JUNOS software automatically subdivides packet payloads that exceed this value. Any value you set must be a multiple of 64 bytes (*N*x64). The default value, 0, results in no fragmentation.

Configuring Link Services Delay-Sensitive Packet Interleaving

For link services FRF.15 and MLPPP interfaces only, you can configure link fragment interleaving (LFI). LFI reduces excessive delays of Frame Relay packets by fragmenting long packets into smaller packets and interleaving them with real-time frames. This allows real-time and non-real-time data frames to be carried together on lower-speed links without causing excessive delays to the real-time traffic. When the peer interface receives the smaller fragments, it reassembles the fragments into their original packet. For example, short delay-sensitive packets, such as packetized voice, can race ahead of larger delay-insensitive packets, such as common data packets..

NOTE: All Link Services PICs (4-multilink bundle, 32-multilink bundle, and 128-multilink bundle) support up to 256 link services interfaces with LFI enabled, if those link services interfaces contain only one constituent link each. For the Link Services PIC, multiple-link LFI bundles are simply multilink bundles, so are limited based on the type of PIC (4-multilink bundle, 32-multilink bundle, and 128-multilink bundle).

In addition, the multilink bundles you configure subtracts from the total of 256 possible LFI-enabled link services interfaces. For example, if a 32-multilink bundle Link Services PIC has 24 multilink bundles configured and active, then you can configure 256 - 24 = 232 LFI-enabled link services interfaces, each with a single constituent link.

The JUNOS software supports end-to-end fragmentation according to the FRF.12 Implementation Agreement standard. Unlike UNI and NNI fragmentation, end-to-end supports fragmentation only at the endpoints. The standard for FRF.12 is defined in the specification FRF.12, *Frame Relay Fragmentation Implementation Agreement*.

By default, packet interleaving is disabled. To enable packet interleaving, include the interleave-fragments statement:

```
interleave-fragments;
```

You can include this statement at the following hierarchy levels:

[edit interfaces *interface-name* unit *logical-unit-number*]

[edit logical-routers *logical-router-name* interfaces *interface-name* unit *logical-unit-number*]

Configuring LFI with DLCI Scheduling

For Link Services and Channelized DS3 IQ PICs, you can configure LFI and DLCI scheduling. For channelized DS3 interfaces, LFI is supported with FRF.15 only, and on M10i and M20 platforms only.

Configuring LFI with DLCI scheduling enables packets entering the Link Services PIC to be fragmented before being transmitted to the Channelized DS3 IQ PIC. Once the fragmented packets enter the Channelized DS3 IQ PIC, they are scheduled at the DLCI level, to allow priority transmission for real-time applications.

For more information about associating a scheduler with a DLCI, see “Associating a Scheduler Map with a DLCI or VLAN” on page 846.

Example: Configuring LFI with DLCI Scheduling

Configure packets entering the Link Services PIC to be fragmented before being transmitted to the Channelized DS3 IQ PIC. Once the fragmented packets enter the Channelized DS3 IQ PIC, they are scheduled at the DLCI level, to allow priority transmission for real-time applications.

```
[edit interfaces]
ls-1/0/0 {
    unit 1 {
        encapsulation multilink-frame-relay-end-to-end;
        interleave-fragments;
        family inet {
            address 192.168.5.2/32 {
                destination 192.168.5.3;
            }
        }
    }
t3-1/0/0:1 {
    per-unit-scheduler;
    unit 0 {
        dlci 16;
        encapsulation multilink-frame-relay-end-to-end;
        family mlfr-end-to-end {
            bundle ls-1/0/0.1;
        }
    }
}

[edit class-of-service]
interfaces {
    t3-1/0/0:1 {
        unit 0 {
            scheduler-map sched-map-logical-0;
            shaping-rate 10m;
        }
        unit 1 {
            scheduler-map sched-map-logical-1;
            shaping-rate 20m;
        }
    }
}
```

```
scheduler-maps {
    sched-map-logical-0 {
      forwarding-class best-effort scheduler sched-best-effort-0;
      forwarding-class assured-forwarding scheduler sched-bronze-0;
      forwarding-class expedited-forwarding scheduler sched-silver-0;
      forwarding-class network-control scheduler sched-gold-0;
    }
sched-map-logical-1 {
      forwarding-class best-effort scheduler sched-best-effort-1;
      forwarding-class assured-forwarding scheduler sched-bronze-1;
      forwarding-class expedited-forwarding scheduler sched-silver-1;
      forwarding-class network-control scheduler sched-gold-1;
}
schedulers {
    sched-best-effort-0 {
      transmit-rate 4m;
    }
    sched-bronze-0 {
      transmit-rate 3m;
    }
    sched-silver-0 {
      transmit-rate 2m;
    }
    sched-gold-0 {
    transmit-rate 1m;
    }
    sched-best-effort-1 {
      transmit-rate 8m;
    }
    sched-bronze-1 {
      transmit-rate 6m;
    }
    sched-silver-1 {
      transmit-rate 4m;
    }
    sched-gold-1 {
    transmit-rate 2m;
    }
}
```

Configuring Minimum Links

You can set the minimum number of links that must be up for the multilink bundle as a whole to be labeled up. By default, only one link must be up for the bundle to be labeled up.

To set the minimum number, include the minimum-links statement:

minimum-links *number*;

You can include this statement at the following hierarchy levels:

[edit interfaces *interface-name* unit *logical-unit-number*]

[edit logical-routers *logical-router-name* interfaces *interface-name* unit *logical-unit-number*]

For link services interfaces, you also can configure the minimum number of links at the physical interface level by including the minimum-links statement at the [edit interfaces *interface-name* mlfr-uni-nni-bundle-options] hierarchy level:

[edit interfaces *interface-name* mlfr-uni-nni-bundle-options]
minimum-links *number*;

The number can be from one through eight. The maximum number of links supported in a bundle is eight. When 8 is specified, all configured links of a bundle must be up.

Configuring MRRU

The maximum received reconstructed unit (MRRU) is similar to a maximum transmission unit (MTU), but applies only to multilink bundles; it is the maximum packet size that the multilink interface can process. By default, the MRRU is set to 1500 bytes; you can configure a different MRRU value if the peer equipment allows. The MRRU includes the original payload plus the 2-byte PPP header, but not the additional MLPPP or MLFR header applied while the individual multilink packets are traversing separate links in the bundle.

To configure a different MRRU value, include the mrru statement:

mrru *bytes*;

You can include this statement at the following hierarchy levels:

[edit interfaces *interface-name* unit *logical-unit-number*]

[edit logical-routers *logical-router-name* interfaces *interface-name* unit *logical-unit-number*]

For link services interfaces, you also can configure a different MRRU at the physical interface level by including the mrru statement at the [edit interfaces *interface-name* mlfr-uni-nni-bundle-options] hierarchy level:

```
[edit interfaces interface-name mlfr-uni-nni-bundle-options]
mrru bytes;
```

The MRRU size can be from 1500 through 4500 bytes.

NOTE: If you set the MRRU on a bundle to a value larger than the MTU of the individual links within it, you must enable a fragmentation threshold for that bundle. Set the threshold to a value no larger than the smallest MTU of any link included in the bundle.

Determine the appropriate MTU size for the bundle by ensuring the MTU size does not exceed the sum of the encapsulation overhead and the MTU sizes for the links in the bundle.

Configuring Sequence Format

For MLPPP, the sequence header format is set to 24 bits by default. You can configure an alternative value of 12 bits, but 24 bits is considered the more robust value for most networks.

To configure a different sequence header value, include the short-sequence statement:

```
short-sequence;
```

You can include this statement at the following hierarchy levels:

[edit interfaces *interface-name* unit *logical-unit-number*]

[edit logical-routers *logical-router-name* interfaces *interface-name* unit *logical-unit-number*]

For MLFR FRF.15, the sequence header format is set to 24 bits by default. This is the only valid option.

Configuring Link Services Physical Interface Properties

You configure link services interface properties at the logical unit and physical interface level.

For information about default settings for link services physical interface properties, see "Default Settings for Link Services Interfaces" on page 423.

You can configure the following link services physical interface properties:

For descriptions of link services physical interface properties that also can be configured at the logical unit level, see "Configuring Logical Interface Properties" on page 412.

Default Settings for Link Services Interfaces

Table 36 lists the default settings for link services statements, together with the other permitted values or value ranges.

Table 36: Link Services Physical Interface Statements for MLFR FRF.16

Option	Default Value	Possible Values
Action red differential delay	disable-tx	disable-tx, remove-link
Red differential delay	10	1 through 2000
Yellow differential delay	6	1 through 2000
Drop timeout period	0 milliseconds	0 through 2000 milliseconds
Encapsulation	multilink-frame-relay-uni-nni	multilink-frame-relay-uni-nni
Fragmentation threshold	0 bytes	128 through 16,320 bytes (*N*x64)
LMI type	itu	ansi, itu
Minimum links	1 link	1 through 8 links
MRRU (maximum received reconstructed unit)	1504 bytes	1500 through 4500 bytes
n391 (full status polling counter)	6	1 through 255
n392 (LMI error threshold)	3	1 through 10
n393 (LMI monitored event count)	4	1 through 10
t391 (link integrity verify polling timer)	10	5 through 30
t392 (polling verification timer)	15	5 through 30
Sequence ID format for MLFR	12 bits	12 bits

Configuring Link Services Physical Interface Encapsulation

Link services interfaces support the physical interface encapsulation MLFR user-to-network interface (UNI) network-to-network interface (NNI). By default, the physical interface encapsulation on link services interfaces is MLFR UNI NNI. Multilink interfaces do not support physical interface encapsulation.

For more information, see "Configuring the Encapsulation on a Physical Interface" on page 73.

You can configure logical interface encapsulation on multilink and link services interfaces. For more information, see "Configuring Logical Interface Encapsulation" on page 415.

To explicitly configure link services physical interface encapsulation, include the encapsulation statement at the [edit interfaces *interface-name*] hierarchy level:

```
[edit interfaces interface-name]
encapsulation type;
```

You must also configure the T1, E1, or DS0 physical and physical interface with the same encapsulation type.

Configuring Link Services Acknowledgment Timers

For link services interfaces configured with MLFR FRF.16, each link endpoint in a bundle initiates a request for bundle operation with its peer by transmitting an add link message. A hello message notifies the peer endpoint that the local endpoint is up. Both ends of a link generate a hello message periodically, or as configured with the hello timer. A remove link message notifies the peer that the local end management is removing the link from bundle operation. Endpoints respond to add link, remove link, and hello messages by sending acknowledgement messages.

You can configure the maximum period to wait for an add link acknowledgement, hello acknowledgement, or remove link acknowledgement by including the acknowledge-timer statement at the [edit interfaces *interface-name* mlfr-uni-nni-bundle-options] hierarchy level:

```
[edit interfaces interface-name mlfr-uni-nni-bundle-options]
acknowledge-timer milliseconds;
```

The acknowledgement timer can be from 1 through 10 milliseconds. The default is 4 milliseconds.

For link services interfaces, you can configure the number of retransmission attempts to be made for consecutive hello or remove link messages after the expiration of the acknowledgement timer by including the acknowledge-retries statement at the [edit interfaces *interface-name* mlfr-uni-nni-bundle-options] hierarchy level:

```
[edit interfaces interface-name mlfr-uni-nni-bundle-options]
acknowledge-retries number;
```

Acknowledgement retries can be a value from 1 through 5. The default is 2.

You can configure the rate at which hello messages are sent by including the hello-timer statement at the [edit interfaces *interface-name* mlfr-uni-nni-bundle-options] hierarchy level:

```
[edit interfaces interface-name mlfr-uni-nni-bundle-options]
hello-timer milliseconds;
```

A hello message is transmitted after the specified period (in milliseconds) has elapsed. The hello timer can be from 1 through 180 milliseconds; the default is 10 milliseconds. When the hello timer expires, a link endpoint generates an add link message.

Configuring Link Services Differential Delay

For link services interfaces configured with MLFR FRF.16, the differential delay between links in a bundle is measured and warning is given when a link has a substantially greater differential delay than other links in the same bundle. The implementing endpoint can determine if the differential delay is in an acceptable range and decide to remove the link from the bundle, or to stop transmission on the link.

You can configure the yellow differential delay for links in a bundle by including the yellow-differential-delay statement at the [edit interfaces *interface-name* mlfr-uni-nni-bundle-options] hierarchy level:

```
[edit interfaces interface-name mlfr-uni-nni-bundle-options]
yellow-differential-delay milliseconds;
```

The yellow differential delay can be from 3 through 2000 milliseconds. The default is 6 milliseconds.

You can configure the red differential delay for links in a bundle to give warning by including the red-differential-delay statements at the [edit interfaces *interface-name* mlfr-uni-nni-bundle-options] hierarchy level:

```
[edit interfaces interface-name mlfr-uni-nni-bundle-options]
red-differential-delay milliseconds;
```

The red differential delay can be from 5 through 2000 milliseconds. The default is 10 milliseconds.

You can configure the action to be taken when differential delay exceeds the red limit by including the action-red-differential-delay red statements at the [edit interfaces *interface-name* mlfr-uni-nni-bundle-options] hierarchy level:

```
[edit interfaces interface-name mlfr-uni-nni-bundle-options]
action-red-differential-delay (disable-tx | remove-link);
```

The disable-tx option disables transmission on the link. The remove-link option removes the link from the bundle. The default action is disable-tx.

Configuring Link Services Keepalive Settings on Frame Relay LMI

You can tune the keepalive settings on the physical link-services interface. By default, the JUNOS software uses ITU Q.933 Annex A LMIs for FRF.16. To use ITU Annex A LMIs, include the lmi-type ansi statement at the [edit interfaces *interface-name* mlfr-uni-nni-bundle-options] hierarchy level:

```
[edit interfaces interface-name mlfr-uni-nni-bundle-options]
lmi-type ansi;
```

To configure Frame Relay keepalive parameters on a link services interface, include the n391, n392, n393, t391, and t392 statements at the [edit interfaces *interface-name* mlfr-uni-nni-bundle-options] hierarchy level:

```
[edit interfaces interface-name mlfr-uni-nni-bundle-options]
n391 number;
n392 number;
n393 number;
t391 number;
t392 number;
```

You can set the following properties:

n391—Full status polling interval. The data terminal equipment (DTE) sends a status inquiry to the data communication equipment (DCE) at the interval specified by t391. n391 specifies the frequency with which these inquiries expect a full status report; for example, an n391 value of 10 would specify a full status report in response to every tenth inquiry. The intermediate inquiries ask for a keepalive exchange only. The range is from 1 through 255, with a default value of 6.

n392—Error threshold. The number of errors required to bring down the link, within the event count specified by n393. The range is from 1 through 10, with a default value of 3.

n393—Monitored event count. The range is from 1 through 10, with a default value of 4.

t391—Keepalive timer. Interval at which the DTE sends out a keepalive response request to the DCE and updates status, depending on the error threshold value. The range is from 5 through 30 seconds, with a default value of 10 seconds.

t392—Keepalive timer. Period during which the DCE checks for keepalive responses from the DTE and updates status, depending on the DCE error threshold value. The range is from 5 through 30 seconds, with a default value of 15 seconds.

For the LMI to work properly, you must configure one side of a link services bundle to be a DCE.

Multilink and Link Services Interface Structure

Each Multilink Services or Link Services PIC can support a number of bundles. A bundle can contain up to eight individual links.

For Multilink Services PICs, the links can be T1, E1, or DS0 physical interfaces, and each link is associated with a logical unit number that you configure. For Link Services PICs, the links can be E1, T1, channelized DS3-to-DS1, channelized DS3-to-DS0, channelized E1, channelized STM1interfaces, or channelized IQ interfaces. For MLFR FRF.16 bundles, each link is associated with a channel number that you configure.

You must configure a link before it can join a bundle. Each bundle should consist solely of one type of link; we recommend that you not mix physical interfaces of differing speeds within a bundle.

This section is organized as follows:

Multilink Services PIC and Link Services PIC Capacities

Three versions of Multilink Services and three versions of Link Services PICs are available, as shown in Table 37. The PIC hardware is identical, except for different faceplates that enable you to identify which version you are installing. The software limits the unit numbers and maximum number of physical interfaces you assign to the PIC.

Table 37: Multilink Services PIC Capacities

PIC Capacity	Unit Numbers	Maximum Number of T1 / DS0 Interfaces	Maximum Number of E1 Interfaces
4-bundle PIC	0 through 3	32 links	32 links
32-bundle PIC	0 through 31	256 links	219 links
128-bundle PIC	0 through 127	292 links	219 links

A single PIC can support an aggregate bandwidth of 450 Mbps.

You can configure a larger number of links, but the Multilink Services PIC and Link Services PIC can reliably process only 450 Mbps of traffic. A higher rate of traffic might degrade performance.

Link Services PIC Capabilities

The default number of bundles per Link Services PIC is 16, ranging from *type-fpc/pic/port*:0 to *type-fpc/pic/port*:15.

You can combine MLFR FRF.15, MLFR FRF.16, and MLPPP bundles on a single Link Services PIC.

For an example configuration, see "Configuring a Link Services PIC and Voice Services PIC with a Combination of Bundle Types" on page 442.

To configure the number of bundles on a Link Services PIC, include the mlfr-uni-nni-bundles statement at the [edit chassis fpc *slot-number* pic *pic-number*] hierarchy level:

```
[edit chassis fpc slot-number pic pic-number]
mlfr-uni-nni-bundles number;
```

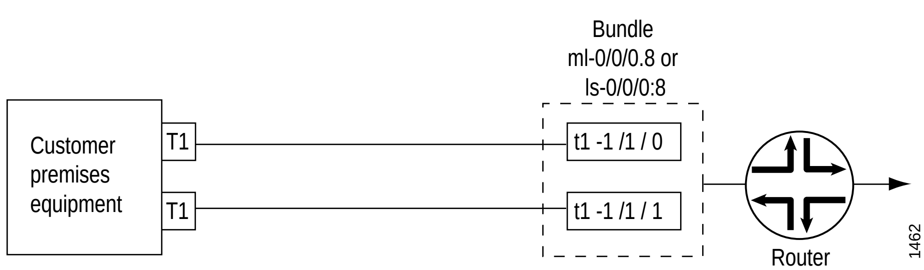
Each Link Services PIC can accommodate a maximum of 256 MLFR UNI NNI bundles. For more information, see the *JUNOS System Basics Configuration Guide*.

A link can associate with one link services bundle only. All Link Services PICs support up to 256 single-link bundles and up to 256 DLCIs. For an example configuration, see "Examples: Configuring Link Services Interfaces" on page 438.

Configuring Bundles

To complete a multilink or link services interface configuration, you need to configure both the physical interface and the multilink or link services bundle. For multilink interfaces, you configure the link bundle on the logical unit. For link services interfaces, you configure the link bundle as a channel (see Figure 28). The physical interface is usually connected to networks capable of supporting MLPPP or MLFR (FRF.15 or FRF.16).

Figure 28: Multilink Interface Configuration



Using the topology in Figure 28 on page 428 as an example, configure a multilink or link services bundle over a T1 connection (for which you have already configured the T1 physical interface) with the following additional configuration statements:

1. To configure a physical T1 link for MLPPP, include the following statements at the [edit interfaces t1-*fpc/pic/port*] hierarchy level:

```
[edit interfaces t1-fpc/pic/port]
unit 0 {
    family mlppp {
        bundle (ml-fpc/pic/port | ls-fpc/pic/port);
    }
}
```

   You do not need to configure an IP address on this link.

   To configure a physical T1 link for MLFR FRF.16, include the following statements at the [edit interfaces t1-*fpc/pic/port*] hierarchy level:

```
[edit interfaces t1-fpc/pic/port]
encapsulation multilink-frame-relay-uni-nni;
unit 0 {
    family mlfr-uni-nni {
        bundle interface-name;
    }
}
```

   You do not need to configure an IP address nor a DLCI on this link.

2. To configure the logical address for the MLPPP, MLFR FRF.15, or MLFR FRF.16 bundle, include the address and destination statements:

```
address address {
    destination address;
}
```

   You can include these statements at the following hierarchy levels:

   [edit interfaces *interface-name* unit *logical-unit-number* family inet]

   [edit logical-routers *logical-router-name* interfaces *interface-name* unit *logical-unit-number* family inet]

   When you add statements such as MRRU to the configuration and commit, the T1 interface becomes part of the multilink bundle.

NOTE: For MLPPP and MLFR (FRF.15 and FRF.16) links, you must specify the subnet address as /32 or /30. Any other subnet designation is treated as a mismatch.

Configuring CoS Components on Link Services PICs

For AS PIC link services IQ interfaces (lsq), JUNOS CoS components are fully supported and are handled normally, as described in “CoS Overview” on page 799. For more information and detailed configuration examples, see “Configuring Link Services IQ Interfaces” on page 449.

For Link Services PIC interfaces (ls), CoS works differently for a link services interface on a J-series Services Services Router than it does for a link services interface on an M-series or T-series platform, as described in the following sections:

Link Services CoS on J-series Services Routers

Unlike M-series and T-series platforms, J-series Services Routers support per-bundle queuing on link services (ls) interfaces. This means that link services interfaces for J-series Services Routers behave the same way as link services IQ interfaces (lsq). (For more information, see “Configuring Link Services IQ Interfaces” on page 449.) There are some exceptions, as follows:

Link services interfaces on J-series Services Routers are internal and are not associated with a physical FPC, PIC, or port. Thus, by convention, link services interfaces are always named ls-0/0/0.*logical*, where the *logical* descriptor is the only variable in the interface name.

Queue 2 is reserved for voice traffic (LFI) on J-series Services Routers, while all other queues perform fragmentation. For FRF.15 on J-series Services Routers, the constituent links bundled on the link services interfaces (ls) require a single scheduler with 25 percent transmission rates and buffer sizes for queues 0 through 3. You should assign this scheduler to each constituent link-in other words, E1 interfaces and T1 interfaces. However, you can configure customized scheduler maps for each associated ls-0/0/0.*logical* interface.

Link services IQ interfaces (lsq) on M-series platforms do not support FRF.15; instead they support end-to-end FRF.12, which is the same as FRF.15 except each bundle has only one link. For FRF.12, link services IQ interfaces (lsq) on M-series platforms do not require equal-size transmission rates and buffer sizes.

For FRF.16 and MLPPP, link services interfaces (ls) on the J-series Services Router work the same as link services IQ interfaces (lsq) on M-series platforms.

Link Services CoS on M-series and T-series Platforms

For Link Services PIC interfaces (ls) on M-series and T-series platforms, queue 0 is the only queue that you should configure to receive fragmented packets. You should configure all other queues to be higher-priority queues.

Table 38 summarizes how CoS queues work on link services (ls) interfaces.

Table 38: Link Services CoS Queues

Supported Bundling Type	Queue 0	Higher-Priority Queues
Hash-based load balancing	No	Yes
MLFR FRF.15	Yes	No
MLFR FRF.16	Yes	No
MLPPP	Yes	No

For M-series and T-series platforms, CoS on link services (ls) interfaces works as follows:

On all platforms, the Link Services PIC currently supports up to four queues: 0, 1, 2, and 3.

Queue 0 uses MLFR FRF.15, MLFR FRF.16, or MLPPP to bundle packets.

Higher-priority queues (1, 2, and 3) use hash-based load balancing to bundle packets. IP and MPLS header information is included in the hash.

For the MLFR FRF.16 protocol, only queue 0 works. If you configure a bundled interface to use MLFR FRF.16 with queue 0, then you must ensure the classifier does not send any traffic to queues 1, 2, and 3 on that interface.

To carry high-priority traffic correctly on MLFR FRF.16 interfaces, you must configure an output firewall filter that forces all traffic into queue 0 on the ls-*fpc*/*pic*/*port*.*channel* interface.

MLFR FRF.15 and MLPPP interfaces support CoS through the use of packet interleaving. The MLFR FRF.16 standard does not support packet interleaving, so all packets destined for an FRF.16 PVC interface must egress from the same queue.

For constituent link interfaces of Link Services PICs, you can configure standard scheduler maps.

For input packets and fragments received from constituent links, you can use regular input firewall filters and standard CoS classifiers on the link services interface.

For packets that pass through a link services interface and are destined for a constituent link interface, all traffic using queue 0 is fragmented. Traffic using higher-priority queues (1, 2, and 3) is not fragmented.

For MLFR FRF.15 and MLPPP, routing protocol packets smaller than 128 bytes are sent to queue 3; routing protocol packets that exceed 128 bytes are sent to queue 0 and fragmented accordingly. For MLFR FRF.16, queue 0 is used for all packet sizes.

You must configure output firewall classification for egress traffic on the link services interface, not directly on the constituent link interface directly.

Inverse multiplexing for ATM (IMA) is not supported on link services interfaces.

For more information, see "Configuring Link Services Delay-Sensitive Packet Interleaving" on page 417 and the *JUNOS Policy Framework Configuration Guide*.

Example: Configuring CoS Components on Link Services PICs

Configure CoS on a link services interface and its bundle link interfaces.

Packets that do not match the firewall filters are sent to a queue that performs load-balanced by sending fragments to all bundle links.

Packets that do match the firewall filters are sent to a queue that does not support packet fragmentation and reassembly; instead, this traffic is load balanced by sending each packet flow to a different bundle link. Each packet that matches a firewall filter is subjected to a hash on the IP source address and the IP destination address to determine the packet flow to which each packet belongs.

When you configure the MLPPP encapsulation type or the multilink FRF.15 Frame Relay end-to-end encapsulation type, routing protocol packets smaller than 128 bytes are sent to the network-control queue on the bundle link interface. This keeps routing protocols operating normally, even when low-speed links are congested by regular packets.

```
[edit interfaces]
ls-7/0/0 {
    unit 0 {
        encapsulation multilink-ppp;
        interleave-fragments;
        family inet {
            filter {
                output lfi_ls_filter;
            }
            address 10.54.0.2/32 {
                destination 10.54.0.1;
            }
        }
    }
}
ge-7/2/0 {
    unit 0 {
        family inet {
            address 192.168.1.1/24;
        }
    }
}
```

```
ce1-7/3/6 {
    no-partition interface-type e1;
}
e1-7/3/6 {
    encapsulation ppp;
    unit 0 {
        family mlppp {
            bundle ls-7/0/0.0;
        }
    }
}
ce1-7/3/7 {
    no-partition interface-type e1;
}
e1-7/3/7 {
    encapsulation ppp;
    unit 0 {
        family mlppp {
            bundle ls-7/0/0.0;
        }
    }
}

[edit class-of-service]
classifiers {
    dscp dscp_default {
        import default;
    }
    inet-precedence inet-precedence_default {
        import default;
    }
}
code-point-aliases {
    dscp {
        af11 001010;
        af12 001100;
        af13 001110;
        af21 010010;
        af22 010100;
        af23 010110;
        af31 011010;
        af32 011100;
        af33 011110;
        af41 100010;
        af42 100100;
        af43 100110;
        be 000000;
        cs1 001000;
        cs2 010000;
        cs3 011000;
        cs4 100000;
        cs5 101000;
        cs6 110000;
        cs7 111000;
        ef 101110;
    }
```

```
        inet-precedence {
            af11 001;
            af21 010;
            af31 011;
            af41 100;
            be 000;
            cs6 110;
            cs7 111;
            ef 101;
            nc1 110;
            nc2 111;
        }
    }
    forwarding-classes {
        queue 0 be;
        queue 1 ef;
        queue 2 af;
        queue 3 nc;
    }
    interfaces {
        ge-7/2/0 {
            scheduler-map sched-map;
            unit 0 {
                classifiers {
                    dscp dscp_default;
                }
            }
        }
        e1-7/3/6 {
            scheduler-map sched-map;
        }
        e1-7/3/7 {
            scheduler-map sched-map;
        }
        ls-7/0/0 {
            unit 0 {
                classifiers {
                    inet-precedence inet-precedence_default;
                }
            }
        }
    }
    scheduler-maps {
        sched-map {
            forwarding-class af scheduler af-scheduler;
            forwarding-class be scheduler be-scheduler;
            forwarding-class ef scheduler ef-scheduler;
            forwarding-class nc scheduler nc-scheduler;
        }
    }
```

```
schedulers {
    af-scheduler {
        transmit-rate percent 25;
        buffer-size percent 25;
    }
    be-scheduler {
        transmit-rate percent 25;
        buffer-size percent 25;
    }
    ef-scheduler {
        transmit-rate percent 25;
        buffer-size percent 25;
    }
    nc-scheduler {
        transmit-rate percent 25;
        buffer-size percent 25;
    }
}

[edit firewall]
filter lfi_ls_filter {
    term term0 {
        from {
            address {
                192.168.1.3/32;
            }
            precedence 5;
        }
        then {
            count count-192-2-1-3;
            forwarding-class af;
            accept;
        }
    }
    term default {
        then {
            log;
            forwarding-class best effort;
            accept;
        }
    }
}
```

Examples: Configuring Multilink Interfaces

These examples show only the multilink part of the configuration. To see the T1 configuration options, see “Configuring T1 Interfaces” on page 545.

Configuring an MLPPP Interface

```
[edit interfaces]
ml-1/0/0 {
    unit 1 {
        fragment-threshold 128;
        family inet {
            address 192.168.5.1/32 {
                destination 192.168.200.200;
            }
        }
    }
    unit 10 {
        family inet {
            address 172.16.1.3/32 {
                destination 172.16.1.2;
            }
        }
    }
}
t1-5/1/0 {
    unit 0 {
        family mlppp {
            bundle ml-1/0/0.1;
        }
    }
}
t1-5/1/1 {
    unit 0 {
        family mlppp {
            bundle ml-1/0/0.1;
        }
    }
}
t1-5/1/2 {
    unit 0 {
        family mlppp {
            bundle ml-1/0/0.1;
        }
    }
}
```

Configuring an MLFR FRF.15 Interface

```
[edit interfaces]
ml-1/0/0 {
    unit 1 {
        encapsulation multilink-frame-relay-end-to-end;
        family inet {
            address 192.168.5.2/32 {
                destination 192.168.5.3;
            }
        }
    }
    unit 10 {
        encapsulation multilink-frame-relay-end-to-end;
        family inet {
            address 172.16.1.3/32 {
                destination 172.16.1.2;
            }
        }
    }
}
t1-5/1/0 {
        unit 0 {
            dlci 16;
            encapsulation multilink-frame-relay-end-to-end;
            family mlfr-end-to-end {
                bundle ml-1/0/0.1;
        }
    }
}
t1-5/1/1 {
    unit 0 {
        dlci 17;
        encapsulation multilink-frame-relay-end-to-end;
        family mlfr-end-to-end {
            bundle ml-1/0/0.10;
        }
    }
}
t1-5/1/2 {
    unit 0 {
        dlci 26;
        encapsulation multilink-frame-relay-end-to-end;
        family mlfr-end-to-end {
            bundle ml-1/0/0.10;
        }
    }
}
```

Examples: Configuring Link Services Interfaces

This example shows only the link services part of the configuration. To see the T1 configuration options, see “Configuring T1 Interfaces” on page 545.

The examples in this section show the following configurations:

- Configuring a Link Services Interface with Two Links on page 438, as listed in Table 39
- Configuring a Link Services PIC with MLPPP on page 440
- Configuring a Link Services PIC with MLFR FRF.15 on page 440
- Configuring a Link Services PIC with MLFR FRF.16 on page 441
- Configuring a Link Services PIC and Voice Services PIC with a Combination of Bundle Types on page 442

Table 39: Link Services Bundle

Router A	Router B
t1-0/1/0 (ls-1/1/0:3)	t1-0/3/0 (ls-0/0/0:10)
t1-0/1/1 (ls-1/1/0:3)	t1-0/3/1 (ls-0/0/0:10)

This configuration initiates the MLFR UNI NNI protocol between Router A and Router B and logically connects link services bundles ls-1/1/0.3 and ls-0/0/0.10.

For the LMI to work properly, you must configure one routing platform to be a DCE.

Configuring a Link Services Interface with Two Links

On Router A

```
[edit interfaces]
ls-1/1/0:3 {
    dce;
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
        dlci 16;
        family inet {
            address 10.3.3.1/32 {
                destination 10.3.3.2;
            }
        }
    }
}
```

```
t1-0/1/0 {
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
        family mlfr-uni-nni {
            bundle ls-1/1/0:3;
        }
    }
}

t1-0/1/1 {
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
        family mlfr-uni-nni {
            bundle ls-1/1/0:3;
        }
    }
}
```

On Router B

```
[edit interfaces]
ls-0/0/0:10 {
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
        dlci 16;
        family inet {
            address 10.3.3.2/32 {
                destination 10.3.3.1;
            }
        }
    }
}

t1-0/3/0 {
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
        family mlfr-uni-nni {
            bundle ls-0/0/0:10;
        }
    }
}

t1-0/3/1 {
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
        family mlfr-uni-nni {
            bundle ls-0/0/0:10;
        }
    }
}
```

Configuring a Link Services PIC with MLPPP

```
[edit interfaces]
t1-0/0/0 {
    encapsulation ppp;
    unit 0 {
        family mlppp {
            bundle ls-0/3/0.0;
        }
    }
}
t1-0/0/1 {
    encapsulation ppp;
    unit 0 {
        family mlppp {
            bundle ls-0/3/0.0;
        }
    }
}
ls-0/3/0 {
    unit 0 {
        encapsulation multilink-ppp;
        family inet {
            address 10.16.1.2/32 {
                destination 10.16.1.1;
            }
        }
        family iso;
        family inet6 {
            address 8016::1:2/126;
        }
    }
}
```

Configuring a Link Services PIC with MLFR FRF.15

```
[edit interfaces]
t1-0/0/0 {
    encapsulation frame-relay;
    unit 0 {
        dlci 16;
        family mlfr-end-to-end {
            bundle ls-0/3/0.0;
        }
    }
}
t1-0/0/1 {
    encapsulation frame-relay;
    unit 0 {
        dlci 16;
        family mlfr-end-to-end {
            bundle ls-0/3/0.0;
        }
    }
}
```

```
ls-0/3/0 {
    unit 0 {
        encapsulation multilink-frame-relay-end-to-end;
        family inet {
            address 10.16.1.2/32 {
                destination 10.16.1.1;
            }
        }
        family iso;
        family inet6 {
            address 8016::1:2/126;
        }
    }
}
```

Configuring a Link Services PIC with MLFR FRF.16

```
[edit chassis]
fpc 1 {
    pic 2 {
        mlfr-uni-nni-bundles 5;
    }
}

[edit interfaces]
t1-0/0/0 {
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
        family mlfr-uni-nni {
            bundle ls-1/2/0:0;
        }
    }
}
t1-0/0/1 {
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
        family mlfr-uni-nni {
            bundle ls-1/2/0:0;
        }
    }
}
ls-1/2/0:0 {
    dce;
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
        dlci 26;
        family inet {
            address 10.26.1.1/32 {
                destination 10.26.1.2;
            }
        }
    }
}
```

Configuring a Link Services PIC and Voice Services PIC with a Combination of Bundle Types

```
[edit chassis]
fpc 1 {
    pic 3 {
        mlfr-uni-nni-bundles 4;
    }
}

[edit interfaces]
t1-0/2/0:0 {
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
        family mlfr-uni-nni {
            bundle ls-1/3/0:0;
        }
    }
}
t1-0/2/0:1 {
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
        family mlfr-uni-nni {
            bundle ls-1/3/0:0;
        }
    }
}
t1-0/2/0:5 {
    unit 0 {
        family mlppp {
            bundle ls-1/3/0.2;
        }
    }
}
t1-0/2/0:6 {
    unit 0 {
        family mlppp {
            bundle ls-1/3/0.2;
        }
    }
}
t1-0/2/0:7 {
    encapsulation frame-relay;
    unit 0 {
        dlci 20;
        family mlfr-end-to-end {
            bundle ls-1/3/0.1;
        }
    }
}
t1-0/2/0:8 {
    encapsulation frame-relay;
    unit 0 {
        dlci 20;
        family mlfr-end-to-end {
            bundle ls-1/3/0.1;
        }
    }
}
```

```
t1-0/2/0:10 {
    no-keepalives;
    encapsulation ppp;
    unit 0 {
        family mlppp {
            bundle vsp-1/1/0.0;
        }
    }
}
t3-1/0/0 {
    no-keepalives;
    encapsulation ppp;
    unit 0 {
        family mlppp {
            bundle vsp-1/1/0.2;
        }
    }
}
vsp-1/1/0 {
    unit 0 {
        encapsulation multilink-ppp;
        compression {
            rtp {
                f-max-period 100;
                queues [ q1 q2 ];
                port minimum 2000 maximum 6000;
            }
        }
        family inet {
            address 10.5.5.5/24;
        }
    }
    unit 1 {
        encapsulation multilink-ppp;
        compression {
            rtp {
                port minimum 2000 maximum 6000;
            }
        }
        family inet {
            address 10.6.6.1/24;
        }
    }
    unit 2 {
        encapsulation multilink-ppp;
        compression {
            rtp {
                port minimum 2000 maximum 6000;
            }
        }
        family inet {
            address 10.9.9.1/24;
        }
    }
}
```

```
t1-1/2/0 {
    no-keepalives;
    unit 0 {
        family mlppp {
            bundle vsp-1/1/0.1;
        }
    }
}
ls-1/3/0 {
    unit 1 {
        encapsulation multilink-frame-relay-end-to-end;
        family inet {
            address 10.1.4.1/24;
        }
    }
    unit 2 {
        encapsulation multilink-ppp;
        family inet {
            address 10.7.4.1/24;
        }
    }
}
ls-1/3/0:0 {
    encapsulation multilink-frame-relay-uni-nni;
    mlfr-uni-nni-bundle-options {
        debug-flags 15;
    }
    unit 0 {
        dlci 20;
        family inet {
            address 10.5.4.1/24;
        }
    }
}

[edit routing-options]
static {
    route 10.12.12.0/24 next-hop 10.1.1.9;
}
```

On Router B

```
[edit chassis]
fpc 1 {
    pic 3 {
      mlfr-uni-nni-bundles 4;
    }
}

[edit interfaces]
ge-0/0/0 {
    unit 0 {
      family inet {
        address 10.1.1.1/24;
      }
    }
}
so-0/1/1 {
    encapsulation ppp;
    unit 0 {
      family inet {
        address 10.7.7.7/24;
      }
    }
}
t1-0/2/0:0 {
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
      family mlfr-uni-nni {
        bundle ls-1/3/0:0;
      }
    }
}
t1-0/2/0:1 {
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
      family mlfr-uni-nni {
        bundle ls-1/3/0:0;
      }
    }
}
t1-0/2/0:5 {
    no-keepalives;
    unit 0 {
      family mlppp {
        bundle ls-1/3/0.2;
      }
    }
}
t1-0/2/0:6 {
    no-keepalives;
    unit 0 {
      family mlppp {
        bundle ls-1/3/0.2;
      }
    }
}
```

```
t1-0/2/0:7 {
    dce;
    encapsulation frame-relay;
    unit 0 {
        dlci 20;
        family mlfr-end-to-end {
            bundle ls-1/3/0.1;
        }
    }
}
t1-0/2/0:8 {
    dce;
    encapsulation frame-relay;
    unit 0 {
        dlci 20;
        family mlfr-end-to-end {
            bundle ls-1/3/0.1;
        }
    }
}
t1-0/2/0:10 {
    no-keepalives;
    encapsulation ppp;
    unit 0 {
        family mlppp {
            bundle vsp-1/1/0.0;
        }
    }
}
t3-0/3/0 {
    no-keepalives;
    encapsulation ppp;
    unit 0 {
        family mlppp {
            bundle vsp-1/1/0.2;
        }
    }
}
ge-1/0/0 {
    unit 0 {
        family inet {
            address 10.2.2.1/24;
        }
    }
}
```

```
vsp-1/1/0 {
    unit 0 {
        compression {
            rtp {
                port minimum 2000 maximum 6000;
            }
        }
        family inet {
            address 10.5.5.1/24;
        }
    }
    unit 1 {
        encapsulation multilink-ppp;
        compression {
            rtp {
                port minimum 16384 maximum 20102;
            }
        }
        family inet {
            address 10.3.4.1/24;
        }
    }
    unit 2 {
        encapsulation multilink-ppp;
        compression {
            rtp {
                port minimum 2000 maximum 6000;
            }
        }
        family inet {
            address 10.9.9.9/24;
        }
    }
}
t1-1/2/2 {
    no-keepalives;
    unit 0 {
        family mlppp {
            bundle ls-1/3/0.1;
        }
    }
}
t1-1/2/3 {
    no-keepalives;
    unit 0 {
        family mlppp {
            bundle vsp-1/1/0.1;
        }
    }
}
```

```
ls-1/3/0 {
    unit 1 {
        encapsulation multilink-frame-relay-end-to-end;
        family inet {
            address 10.1.4.4/24;
        }
        family iso;
    }
    unit 2 {
        encapsulation multilink-ppp;
        family inet {
            address 10.7.4.4/24;
        }
    }
}
ls-1/3/0:0 {
    dce;
    encapsulation multilink-frame-relay-uni-nni;
    unit 0 {
        dlci 20;
        family inet {
            address 10.5.4.4/24;
        }
    }
}

[edit routing-options]
static {
    route 10.12.12.0/24 next-hop 10.3.4.4;
}
```