75xx/76xx 라우터 - 분산 기능 구성 및 확인

목차

소개

이 문서는 다음 기능을 이해하고 구성하고 확인하는 데 도움이 됩니다.

- dMLP(Distributed Multilink Point to Point Protocol)
- LFI(Distributed Link Fragmentation and Interleaving)
- 임대 회선을 통한 분산 LFI(dLFIoLL)
- Distributed LFI over Frame Relay(dLFIoFR)
- ATM을 통한 분산 LFI(dLFIoATM)
- dMLP(Distributed MLP)와 dLFIoLL의 차이점

- dMLFR(Distributed Multilink Frame Relay)
- DDR(Distributed Dial on Demand Routing)

사전 요구 사항

요구 사항

이 문서의 독자는 Cisco 7500/7600/6500용 분산 기능에 대해 알고 있어야 합니다.

사용되는 구성 요소

이 문서의 정보는 다음 소프트웨어 및 하드웨어 버전을 기반으로 합니다.

- 모든 Cisco 7500 및 7600 플랫폼**참고:** 이 문서의 정보는 6500 플랫폼에도 적용됩니다.
- 관련 Cisco IOS® 소프트웨어 릴리스(이 표에는 다음이 나열되어 있습니다.

각 지사 및 플랫폼에 대한 분산 기능 지원

기능	PA(Port Adapter)[1] 지원됨	7500 플랫폼		7600 플랫폼	
		주요 Cisco IOS 소프트웨어 릴리스	Cisco IOS 릴리스(중간)	주요 Cisco IOS 소프트웨어 릴리스	Cisco IOS Software 릴리스(중간)
dMLP	Chan-PA PA-4T+ PA-8T	12.0T 12.0S 12.1 12.1T 12.2 12.2T 12.3 12.3T 12.2S 12.1E[2]	12.0(3)T 이상 12.0(9)S 이상	12.2SX 12.1E[2]	—
LFI전체	Chan-PA PA-4T+ PA-8T	12.2T 12.3 12.3T 12.0S	12.2(8)T 이상 12.0(24)S 이상	12.2SX	12.2(17)SXB 이상
dLFIoFR	Chan-PA PA-4T+ PA-8T	12.2T 12.3 12.3T	12.2(4)T3 이상	12.2SX	12.2(17)SXB 이상
dLFIATM	PA-A3 PA-A6	12.2T 12.3 12.3T	12.2(4)T3 이상	12.2SX	12.2(17)SXB 이상
dMLFR	Chan-PA PA-4T+ PA-8T	12.0S 12.3T	12.0(24)S 이상 12.3(4)T 이상	12.2SX	12.2(17)SXB 이상
dMLP	Chan-PA	12.0S 12.2T	12.0(	12.2S	12.2(1

의 QoS	PA-4T+ PA-8T	12.3 12.3T	24)S 이상 12.2(8)T 이상	X	7)SXB 이상
dMLP의 MPLS dLFIoLL의 MPLS	Chan-PA PA-4T+ PA-8T	12.2조 12.3	12.2(15)T11 이상 12.3(5a) 이상	12.2SX	12.2(17)SXB 이상
분산된 DDR	PA-MC-xT1 PA-MC-xE1 PA-MC-xTE1 PA-MCX-xTE1	12.3조	12.3(7)T 이상	—	—

참고: 다음 정보에 유의하십시오.

1. 이러한 PA는 다음과 같은 분산 기능을 지원합니다.CT3IPPA-MC-T3PA-MC-2T3+PA-MC-E3PA-MC-2E1PA-MC-2T1PA-MC-4T1PA-MC-8T1PA-MC-8E1PA-MC-8TE1+PA-MC-STM-1
2. Cisco IOS Software 릴리스 12.1E는 7500 및 7600 플랫폼 모두에서 이러한 기능을 지원합니다.

이 문서의 정보는 특정 랩 환경의 디바이스를 토대로 작성되었습니다. 이 문서에 사용된 모든 디바이스는 초기화된(기본) 컨피그레이션으로 시작되었습니다. 현재 네트워크가 작동 중인 경우, 모든 명령어의 잠재적인 영향을 미리 숙지하시기 바랍니다.

표기 규칙

문서 규칙에 대한 자세한 내용은 Cisco 기술 팁 표기 규칙을 참조하십시오.

분산 기능

이 문서에서는 다음과 같은 기능에 대해 설명합니다.

- 분산 MLP
- 분산 LFI
- 임대 회선을 통한 분산 LFI
- Distributed LFI over Frame Relay
- ATM을 통한 분산 LFI
- dMLP와 dLFIoLL의 차이점
- 분산 MLFR
- 분산 다이얼러
- 분산 기능을 지원하는 플랫폼 및 라인 카드

분산 MLPPP

dMLP(Distributed Multilink Point to Point Protocol) 기능을 사용하면 Cisco 7500 또는 7600 Series 라우터의 라인 카드(VIP, FlexWAN)에 있는 전체 또는 소수 T1/E1 회선을 여러 링크의 대역폭이 결합된 번들로 결합할 수 있습니다. 이렇게 하려면 분산 MLP 번들을 사용합니다. 사용자는 라우터의 번들 수와 번들당 링크 수를 선택할 수 있습니다. 따라서 사용자는 T3 라인을 구입할 필요 없이 단일 T1/E1 라인 이상으로 네트워크 링크의 대역폭을 늘릴 수 있습니다. 비 dMLP에서 모든 패킷은 RP(Route Processor)에 의해 스위칭됩니다. 따라서 이러한 구현은 MLP를 실행하는 몇 개의 T1/E1 회선에서 CPU 사용률이 높으므로 RP의 성능에 영향을 미칩니다. dMLP를 사용하면 데이터 경로가 처리되고 라인 카드 CPU 및 메모리로 제한되므로 라우터에서 처리할 수 있는 총 번들 수가 증가합니다. dMLP를 사용하면 DS0(64Kbps)부터 시작하여 분수 T1/E1을 번들링할 수 있습니다.

분산 LFI

dLFI 기능은 실시간 트래픽에 과도한 지연 없이 저속 프레임 릴레이 및 ATM VC(Virtual Circuit)와 임대 회선에서 실시간 트래픽(예: 음성)과 비실시간 트래픽(예: 데이터)의 전송을 지원합니다.

이 기능은 프레임 릴레이, ATM 및 임대 회선을 통해 MLP(Multilink PPP)를 사용하여 구현됩니다. 이 기능은 대용량 데이터 패킷을 작은 프래그먼트의 시퀀스로 프래그먼트하여 지연 시간에 민감한 실시간 패킷과 비실시간 패킷이 동일한 링크를 공유할 수 있도록 합니다. 그런 다음 프래그먼트가 실시간 패킷과 인터리빙됩니다. 링크의 수신 측에서 프래그먼트가 리어셈블되고 패킷이 재구성됩니다.

dLFI 기능은 음성 등의 분산형 낮은 대기 시간 대기열을 통해 실시간 트래픽을 전송하지만 대역폭 문제가 있는 네트워크에서 자주 유용합니다. 이렇게 하면 시간이 덜 중요한 대규모 데이터 패킷의 전송으로 인해 실시간 트래픽이 지연됩니다. 이러한 네트워크에서 dLFI 기능을 사용하여 대규모 데이터 패킷을 여러 세그먼트로 분해할 수 있습니다. 그런 다음 데이터 패킷의 이러한 세그먼트 간에 실시간 트래픽 패킷을 전송할 수 있습니다. 이 시나리오에서 실시간 트래픽은 낮은 우선 순위의 데이터 패킷이 네트워크를 통과할 때까지 기다리는 동안 긴 지연이 발생하지 않습니다. 데이터 패킷은 링크의 수신 측에서 리어셈블되므로 데이터가 그대로 전달됩니다.

링크의 프래그먼트 크기는 멀티링크 번들의 프래그먼트 지연을 기반으로 계산되며 ppp multilink fragment-delay *n* ***명령으로*** 구성됩니다. 여기서,

```
fragment size = bandwidth × fragment-delay / 8
```

이 프래그먼트 크기는 IP 페이로드만 나타냅니다. 캡슐화 바이트(프래그먼트 크기 = 가중치 - 캡슐화 바이트)는 포함하지 않습니다. "weight" 및 "fragment size"라는 용어는 RP의 **show ppp multilink** 명령 출력에서 볼 수 있습니다. 프래그먼트 지연이 구성되지 않은 경우 최대 프래그먼트 지연 30에 대해 기본 프래그먼트 크기가 계산됩니다.

참고: 대역폭이 다른 링크의 경우 선택한 프래그먼트 크기는 대역폭이 가장 적은 링크를 기반으로 합니다.

임대 회선을 통한 분산 LFI

dLFIoLL 기능은 분산된 링크 조각화 및 인터리빙 기능을 임대 회선으로 확장합니다. 분산 LFI는 멀티링크 그룹 인터페이스**에서 ppp multilink interleave** 명령을 사용하여 구성됩니다. 대역폭이 768Kbps 미만인 다중 링크 인터페이스에서 분산 LFI를 사용하는 것이 좋습니다. 이는 768Kbps보다 큰 대역폭에 대한 1500바이트 패킷의 직렬화 지연이 허용 가능한 지연 제한 내에 있으며 프래그먼트화할 필요가 없기 때문입니다.

Distributed LFI over Frame Relay

dLFIoFR 기능은 MLPoFR(Multilink PPP over Frame Relay) 기능의 확장입니다. MLP는 조각화에 사용됩니다. 이 기능은 프래그먼트화를 지원하고 낮은 레이턴시 대기열 처리를 통해 높은 우선 순위 패킷을 인터레이션할 수 있는 FRF.12와 유사합니다.

연결된 **가상** 액세스 인터페이스에서 인터리빙을 활성화하려면 Virtual-template의 ppp multilink interleave 명령이 필요합니다. 직렬 인터페이스에서 분산 CEF 스위칭을 활성화하는 것 외에도 이 명령은 분산 LFI가 작동하기 위한 전제 조건입니다.

참고: ATM 인터네트워킹에 프레임 릴레이를 사용하지 않는 한 FRF.12에서는 대역폭 사용률이 더 높으므로 dLFIoFR 대신 FRF.12를 사용하는 것이 좋습니다.

ATM을 통한 분산 LFI

dLFIoATM 기능은 MLPoATM(Multilink PPP over ATM) 기능의 확장입니다. MLP는 조각화에 사용됩니다.

연결된 **가상** 액세스 인터페이스에서 인터리빙을 활성화하려면 Virtual-template의 ppp multilink interleave 명령이 필요합니다. 직렬 인터페이스에서 분산 CEF 스위칭을 활성화하는 것 외에도 이 명령은 분산 LFI가 작동하기 위한 전제 조건입니다.

dLFIoATM에서는 ATM 셀에 불필요한 패딩이 발생하지 않도록 패킷을 ATM 셀에 맞게 만드는 프래그먼트 크기를 선택하는 것이 매우 중요합니다. 예를 들어, 선택한 프래그먼트 크기가 124바이트이면 124바이트의 IP 페이로드가 124 + 10(MLP 헤더) + 8(SNAP 헤더) = 142바이트로 최종 전환됩니다. 첫 번째 프래그먼트는 124 + 10 + 2(First Fragment PID Header size) + 8 = 144바이트로 출력됩니다. 즉, 이 패킷은 3개의 ATM 셀을 사용하여 페이로드를 전송하므로, 압축된 셀을 가장 효율적으로 사용합니다.

dMLP와 dLFIoLL의 차이점

dMLP는 전송 측에서 프래그먼트화를 지원하지 않는 반면 dLFIoLL은 이를 지원합니다.

참고: 멀티링크 번들에서 음성 트래픽에 대해 둘 이상의 링크와 함께 사용되는 인터리빙 및 단편화는 번들의 여러 링크를 통해 수신되는 음성 트래픽을 순서대로 수신하도록 보장하지 않습니다. 정확한 음성 순서는 상위 레이어에서 처리됩니다.

분산 MLFR

분산 MLFR 기능에는 Frame Relay Forum Multilink Frame Relay UNI/NNI Implementation Agreement(FRF.16)에 기반한 라인 카드 지원 Cisco 7500 및 7600 Series 라우터의 기능이 도입되었습니다. 분산된 MLFR 기능은 여러 직렬 링크를 단일 대역폭 번들로 집계할 수 있으므로 특정 애플리케이션의 대역폭을 늘릴 수 있는 비용 효율적인 방법을 제공합니다. MLFR는 프레임 릴레이 네트워크의 UNI(User-to-Network Interfaces) 및 NNI(Network-to-Network Interfaces)에서 지원됩니다.

번들은 번들 링크라고 하는 여러 직렬 링크로 구성됩니다. 번들 내의 각 번들 링크는 물리적 인터페이스에 해당합니다. 번들 링크는 프레임 릴레이 데이터 링크 레이어에 표시되지 않으므로 이러한 인터페이스에서 프레임 릴레이 기능을 구성할 수 없습니다. 이러한 링크에 적용할 일반 프레임 릴레이 기능은 번들 인터페이스에서 구성해야 합니다. 번들 링크는 피어 디바이스에 표시됩니다.

분산된 DDR

분산 DDR 기능을 사용하면 다이얼러 인터페이스에서 분산형 스위칭을 수행할 수 있습니다. 이 기능이 없으면 모든 다이얼인 트래픽은 스위칭을 위해 호스트에 펀딩되어야 합니다. 이 기능을 사용하면 제어 패킷만 RP로 전송되지만, 연결이 설정된 후 VIP 자체에서 스위칭 결정이 수행됩니다.

레거시 다이얼러 컨피그레이션 및 다이얼러 프로파일 컨피그레이션은 모두 PPP 캡슐화에서만 지원됩니다. 다이얼러 인터페이스의 MLP도 지원됩니다. QoS는 다이얼러 인터페이스의 분산 스위칭에서 지원되지 않습니다.

분산 기능 사전 요구 사항 및 제한 사항

사전 요구 사항

이러한 모든 분산 기능에 대한 일반적인 사전 요구 사항은 다음과 같습니다.

- dCEF(Distributed Cisco Express Forwarding) 스위칭은 전역적으로 활성화해야 합니다.
- MLP 번들의 일부인 멤버 직렬 인터페이스에서 dCEF 스위칭을 활성화해야 합니다.
- dCEF 스위칭은 dLFIoFR 및 dLFIoATM 인터페이스의 물리적 링크에서 활성화해야 합니다.
- LFIoFR 및 LFIoATM을 배포하려면 인터리브 컨피그레이션이 필요합니다.
- dLFIoFR 및 dLFIoATM 인터페이스에 대한 가상 템플릿 인터페이스에서 필요한 대역폭을 구성합니다.
- RP에서 PPP 디버그가 활성화되면 MLP를 관찰할 수 . RSP(Route Switch Processor)의 메시지로 전달되었습니다. 이 메시지는 혼동되고 원치 않으므로(특히 Cisco CDP(Discovery Protocol) 패킷에 대한 메시지인 경우) 번들의 멤버 링크에서 cdp**를 사용하지 않도록** 구성해야 합니다.
- 번들의 모든 멤버 링크에는 keepalive가 활성화되어 있어야 합니다.

제한 사항

이러한 모든 분산 기능에 대한 일반적인 제한은 다음과 같습니다.

- T1 및 E1 라인은 번들에서 혼합할 수 없습니다.
- 지원되는 최대 차등 지연은 30ms입니다.
- 번들의 모든 행은 동일한 포트 어댑터(PA)에 있어야 합니다.
- 하드웨어 압축은 지원되지 않습니다.
- VIP 또는 FlexWAN CEF는 IP로만 제한됩니다. 다른 모든 프로토콜은 RSP로 전송됩니다.
- dMLP 및 dMLFR의 전송 측에서 프래그먼트화가 지원되지 않습니다.
- 대부분의 이전 대기열 메커니즘은 dLFI에서 지원되지 않습니다. 이러한 메커니즘은 다음과 같습니다.가상 템플릿 인터페이스에서 공정 대기열 생성가상 템플릿 인터페이스에서 임의 탐지 사용자 지정 대기열 지정우선 순위 큐잉
- Modular QoS CLI를 사용하여 트래픽 정책에서 공정 대기열, 임의 탐지(dWRED) 및 우선순위 큐잉을 구성할 수 있습니다.
- dLFIoFR 또는 dLFIoATM을 사용하는 경우 MLP 번들당 하나의 링크만 지원됩니다. dLFIoFR 또는 dLFIoATM을 사용할 때 MLP 번들에 둘 이상의 링크가 사용된 경우 dLFI가 자동으로 비활성화됩니다. 임대 회선을 통해 dLFI를 사용할 경우 MLP 번들에서 dLFI로 둘 이상의 링크를 구성할 수 있습니다.
- dLFIoATM에서는 al5snap 및 aal5mux만 지원됩니다. aal5nlpid 및 aal5ciscop 캡슐화는 지원되지 않습니다.
- VoIP만 지원됩니다. Voice over Frame Relay 및 Voice over ATM은 dLFI 기능에서 지원되지 않

습니다.
- 이 기능 조합을 사용할 경우 CRTP(Compressed Real-Time Protocol) 컨피그레이션은 멀티링크 인터페이스에서 구성할 수 없습니다.LFI가 활성화된 멀티링크 인터페이스멀티링크 번들에 멤버 링크가 두 개 이상 있음멀티링크 인터페이스에서 우선 순위 기능이 있는 QoS 정책이 활성화됨

dMLP 및 dLFI 컨피그레이션에서는 우선순위 패킷이 MLP 헤더 및 시퀀스 번호를 전달하지 않으며, MLP는 모든 멤버 링크에 우선순위 패킷을 배포합니다. 따라서 CRTP로 압축된 패킷은 수신 라우터에서 순서가 초과될 수 있습니다. 이로 인해 CRTP는 패킷 헤더의 압축을 해제하고 CRTP가 패킷을 삭제하도록 강제할 수 없습니다.

권장 사항

번들의 멤버 링크는 대역폭이 동일해야 합니다. 번들에 서로 다른 대역폭 링크를 추가하면 많은 패킷 순서 변경이 발생하므로 전반적인 번들 처리량이 감소합니다.

VIP2-50(8MB SRAM 포함) 이상은 이러한 분산 기능과 함께 사용하는 것이 좋습니다.

번들 수, 링크 및 메모리 요구 사항

Cisco 7500 Series 라우터용 Distributed Multilink Point-to-Point Protocol을 참조하십시오.

7600 SIP 라인 카드의 하드웨어 및 소프트웨어 MLPPP 또는 MLFR

MLP 및 MLFR은 소프트웨어 또는 하드웨어 기반일 수 있습니다. 하드웨어 기반 MLP 또는 MLFR에서 Freedm은 멀티링크 기능을 제공하며 MLP 헤더는 Freedm Chip에 의해 추가됩니다. 소프트웨어 기반 MLP 또는 MLFR에서 SIP 라인 카드 CPU는 멀티링크 기능(VIP 및 FlexWAN 구현과 유사)을 제공합니다.

하드웨어 기반 MLP 또는 MLFR을 실행하기 위한 제한 사항 및 조건입니다.

- 라인 카드당 최대 336개의 번들과 SPA(Security Posture Assessment)(Freedm)당 168개의 번들만 있을 수 있습니다.
- 번들당 최대 12개의 DS1/E1만 있을 수 있습니다.
- 모든 링크는 동일한 SPA(Fredmm)에 속해야 합니다.
- 번들의 모든 링크는 동일한 속도로 작동해야 합니다.
- TX 프래그먼트 크기는 128, 256 또는 512가 될 수 있습니다. CLI 구성 프래그먼트 크기는 지원되는 가장 가까운 프래그먼트 크기에 매핑됩니다.

```
IF (0 < cli_fragment_size - 6 < 256)
configured_fragment_size = 128
Else IF (cli_fragment_size < 512)
configured_fragment_size = 256
Else
configured_fragment_size = 512
```

- RX 프래그먼트 크기는 1~9.6KB입니다.
- Cisco 전용 형식은 지원되지 않습니다(MLFR).

하드웨어 LFI에서 번들에 하나의 링크만 있고 DS1/E1인 경우 Fragmentation and Interleaving은 Freedm에서 수행됩니다.

show ppp multilink의 출력에서는 하드웨어 구현이 실행 중인지 여부를 표시합니다.

```
Multilink1, bundle name is M1
  Bundle up for 00:14:51
  Bundle is Distributed

  0 lost fragments, 0 reordered, 0 unassigned
  0 discarded, 0 lost received, 1/255 load
   Member links: 1 active, 0 inactive (max not set, min not set)
   Se6/1/0/1:0, since 00:14:51, no frags rcvd
  Distributed fragmentation on. Fragment size 512.  Multilink in Hardware.
```

multilink가 소프트웨어 기반인 경우 **show ppp multilink** 출력에는 `Hardware in Multilink in Hardware(`의 멀티링크)가 없습니다.

패킷의 수명

Rx 데이터 경로

1. 드라이버에서 받은 패킷.
2. 캡슐화가 선택되었습니다. 다음과 같이 하십시오.기본 캡슐화:dMLP에서 인그레스 인터페이스의 캡슐화 유형은 ET_PPP입니다.dMLFR에서 인그레스 인터페이스의 캡슐화 유형은 ET_FRAME_RELAY입니다.dLFIoLL에서 인그레스 인터페이스의 캡슐화 유형은 ET_PPP입니다.dLFIoFR에서 인그레스 인터페이스의 캡슐화 유형은 ET_FRAME_RELAY입니다 .dLFIoATM에서 인그레스 인터페이스의 캡슐화 유형은 ET_ATM입니다.dDialer에서 캡슐화 유형은 ET_PPP입니다.추가 캡슐화 처리:ET_PPP의 경우 NLPID가 스니핑됩니다.dMLP의 경우 NLPID는 MULTILINK입니다.dLFIoLI의 경우 두 가지 사항을 고려해야 합니다.VoIP 패킷 - MLP 헤더가 없으며 IP를 나타내는 NLPID가 있습니다.Data Packets(데이터 패킷) - NLPID는 MULTILINK입니다.dDialer의 경우 패킷에는 MLP 헤더가 없으며 IP를 나타내는 NLPID가 있습니다.**참고:** 이 경우 dCRTP(Distributed Compressed Real-Time Protocol)를 구성할 수 있습니다. 이 경우 추가 처리를 수행하기 전에 헤더의 압축을 해제합니다.
3. ET_FRAME_RELAY의 경우, 패킷이 수신되는 링크가 dMLFR에 대해 구성된 경우 dMLFR에 대해 패킷이 처리됩니다.
4. dLFIoFR 및 dLFIoATM의 경우 캡슐화 유형은 각각 ET_FRAME_RELAY 및 ET_ATM입니다. PPP 헤더가 있습니다 dLFIoLL과 마찬가지로 PPP 헤더는 패킷이 음성 패킷인지 데이터 패킷인지 나타냅니다. dCRTP가 구성된 경우 추가 처리를 수행하기 전에 헤더가 압축 해제됩니다. 음성 패킷이 즉시 전환됩니다.프래그먼트된 데이터 패킷은 스위칭하기 전에 리어셈블해야 합니다.ET_PPP를 사용하면 PPP 링크 패킷을 받을 수 있습니다. ET_FRAME_RELAY를 사용하면 MLFR 제어 패킷을 받을 수 있습니다. 이러한 모든 제어 패킷은 처리를 위해 RP에 펀딩됩니다.
5. 앞서 설명한 디코딩에 따라 패킷이 필요한 스위칭 유형을 확인합니다. 링크 유형은 패킷이 IP 스위치드 또는 MPLS 스위치드 패킷인지 결정합니다. 그런 다음 패킷이 각 스위칭 기능에 제공됩니다.
6. 분산 기능과 함께 번들링을 사용하면 IP 터보 고속 스위칭 벡터가 도난됩니다. 이 작업은 패킷이 멤버 링크에서 수신되기 때문에 수행됩니다. 그러나 번들에서 수신되도록 처리해야 합니다 .또한 호스트에 펀팅된 제어 패킷을 확인해야 합니다. 주로 dMLFR에는 MLFR 제어 패킷이 *아닌* LMI(Local Management Interface) 패킷이 있습니다. 이러한 경우 dLCI 번호 공간의 다른 부분이 사용됩니다. dLCI가 이 공간에 디코딩될 때마다 패킷은 LMI 패킷으로 인식되기 때문에 호스트에 푸시됩니다.VoIP 패킷(Low Latency Queue에 대기됨)은 MLP 헤더를 추가하지 않고 바로 전환됩니다. 분산된 기능은 프래그먼트된 데이터 패킷이 수신될 때 패킷을 수신하고 리

어셈블할 수 있습니다. 리어셈블리 프로세스는 이후 단원에서 설명합니다.패킷이 태그 스위칭되어야 하는 경우 dMLP에서 태그 스위칭 루틴으로 전달됩니다. 그렇지 않으면 IP 스위치로 전환될 경우 IP 스위칭 루틴으로 전달됩니다.**참고:** 모든 비IP 패킷은 dMLFR에서 호스트에 펀딩됩니다.

7. **IP:** IP 스위칭 기능은 모든 패킷에 공통적입니다. 주로 세 가지 작업을 수행합니다.기능이 구성된 경우 필요한 패킷 처리를 수행합니다. 또한 분산 다이얼러를 사용할 경우 "관심 있는 패킷"이 수신될 때 여기서 유휴 타이머를 업데이트합니다. **유휴 타이머 구성 매개 변수**에 대한 자세한 내용은 **다이얼러 유휴 시간 제한(인터페이스)**, 다이얼러 **빠른 유휴(인터페이스)** 및 다이얼러 프로필 구성을 참조하십시오.75xx 라우터에서 인접성은 이그레스 인터페이스의 `tx_acc_ptr`을 나타냅니다. 이그레스 인터페이스가 가상 액세스 인터페이스인 경우 tx_acc_ptr은 `NULL`. 이 경우 캡슐화를 수정하고 fib hwidb에서 `tx_acc_ptr`을 . dLFIoFR 및 dLFIoATM에서 이 조회 및 캡슐화 수정이 필요합니다. dLFIoLL에서 링크는 멀티링크 번들의 일부로 처리됩니다.**참고:** 패킷의 TTL이 여기서 조정되고 IP 프래그먼트화 검사가 수행됩니다. mci_status 모든 패킷에 `RXTYPE_DODIP`로 설정됩니다.
8. 스위칭 결정을 내리면 패킷이 인터페이스에서 배송될 준비가 됩니다. 인터페이스가 로컬 스위칭을 지원하는지 확인합니다. 그렇게 되면, 빠른 전송을 통해 직접 발송됩니다. 그렇지 않으면 경로 캐시 스위치를 시도합니다.인터페이스에 대해 QoS가 구성된 경우 QoS에서 로컬 스위칭 벡터를 도난당합니다. HQF는 패킷을 큐에 넣고, 마지막으로 인터페이스에서 전송되기 전에 패킷을 추가로 처리합니다. dLFI의 경우 dLFI의 경우 조각화 및 인터리빙이 설정됩니다. QoS는 프래그먼트화 루틴의 호출을 처리하고, LLQ가 구성된 경우 우선 순위 큐에서 대기될 음성 패킷과 프래그먼트된 패킷을 인터리프합니다. 이를 통해 VoIP 패킷은 링크를 통해 대용량 데이터 패킷을 전송하는 데 필요한 지연으로 인해 어려움을 겪지 않습니다.

Tx 데이터 경로

vip_dtq_consumer는 패킷을 가져오고 인터페이스 번호를 가져옵니다. 여기서 idb가 . idb에 해당하는 fastsend 루틴을 다음 합니다.

i) 패스센드

1. dMFR에서 fr_info 구조는 if_index와 fr_info가 일치하는 테이블에서 검색됩니다. 제어 패킷이 방금 전송되었습니다. 프레임 헤더에서 dLCI를 제공합니다. 이를 통해 LMI 패킷인지 데이터 패킷인지 확인할 수 있습니다. 프레임 헤더의 dlci 필드를 dmfr 시퀀스 번호로 덮어씁니다. LMI 및 데이터 패킷에 별도의 시퀀스 번호가 사용됩니다.**참고:** 별도의 dLCI에는 별도의 순차 번호가 사용됩니다.
2. dMLP에서 제어 패킷은 우선순위가 높음으로 설정된 상태로 전송됩니다. 데이터 패킷에서 dCRTP가 구성된 경우 헤더가 압축됩니다. 시퀀싱 정보가 포함된 VIP MLP 헤더가 추가되고 멤버 링크에서 전송됩니다.
3. dLFI에서 HQF는 인터페이스를 통해 전송할 패킷을 인터셉트합니다. 음성 패킷인 경우 음성 패킷은 우선순위 대기열에 배치되고(LLQ가 구성된 경우) MLP 캡슐화 없이 인터페이스에서 전송됩니다. 데이터 패킷을 사용하여 dLFI 프래그먼트화 코드를 호출합니다. 이 코드는 프래그먼트를 QoS 코드로 반환하며, 이 코드는 우선순위 트래픽과 인터리빙되어 음성 트래픽의 지연 요구 사항이 충족됩니다. 또한 dCRTP가 구성된 경우 음성 패킷의 헤더만 압축됩니다. 데이터 패킷 헤더는 그대로 유지됩니다.
4. dDialer에서 패킷이 전송되기 전에 출력 링크의 유휴 타이머를 재설정하기 위해 패킷이 분류됩니다. 여러 링크가 동일한 다이얼러에 바인딩되는 경우 출력 링크를 선택한 후에 이 작업이 수행됩니다. 다이얼러 패킷에 헤더가 추가되지 않습니다. 따라서 다이얼러 인터페이스에서는 패킷의 시퀀싱 및 리어셈블이 지원되지 않습니다.

참고: 여러 링크가 있는 dMLP, dDialer, dMLFR 및 dLFI에서 트래픽이 전달되는 물리적 링크는 링크의 혼잡에 따라 달라집니다. 링크가 혼잡할 경우 다음 링크로 이동합니다. (dMLFR, dMLP without QoS 및 dDialer 기능은 링크에 있는 바이트 수를 기준으로 링크를 선택합니다. 현재 링크가 이미 라운드 로빈을 기준으로 바이트 할당량을 전송한 경우 다음 링크를 선택합니다. 이 할당량은 링크의 frag_bytes에 의해 결정됩니다. 다이얼러 멤버 인터페이스의 경우 frag_bytes는 인터페이스 대역폭의 기본값으로 설정됩니다.)

참고: 이그레스 VIP의 인터페이스에서 HQF 컨피그레이션에서 HQF는 dtq_consumer 벡터를 훔칩니다. 이그레스 VIP에 대한 패킷 DMA는 먼저 HQF 검사를 통과합니다. 이그레스 인터페이스에 QoS가 구성된 경우 HQF가 패킷을 처리하면서 패킷을 인터페이스 밖으로 빠르게 전송합니다.

리어셈블리

일반 dDialer 인터페이스는 리어셈블리 및 시퀀싱을 지원하지 않습니다. 다이얼러 인터페이스에서 이 기능을 활성화하려면 다이얼러 인터페이스를 통한 MLP를 구성해야 합니다. 이 작업을 수행하면 Rx 및 Tx 경로가 dMLP 경로와 동일합니다. 패킷이 수신되면 예상 시퀀스 번호에 대해 시퀀스 번호가 확인됩니다.

- 순차 번호가 일치하는 경우패킷이 조각화되지 않은 패킷인 경우 리어셈블리가 필요하지 않습니다. 추가 스위칭 단계를 진행합니다.패킷이 프래그먼트이면 시작 및 종료 비트를 확인하고 프래그먼트가 수신되는 시간과 함께 패킷을 구성합니다.
- 시퀀스 번호가 일치하지 않는 경우:시퀀스 번호가 예상 시퀀스 번호 창 내에 있는 경우 정렬된 "할당되지 않은 조각 목록"에 해당 시퀀스 번호를 넣습니다. 나중에 예상 시퀀스 번호를 받지 못하면 패킷이 여기에 저장된 경우 이 목록이 선택됩니다.시퀀스 번호가 창에 없으면 이를 무시하고 "수신된 손실 조각"을 보고합니다. 이 패킷을 기다리는 동안 시간 초과가 나중에 발생하면 수신기가 재동기화되고 수신한 다음 패킷으로 다시 시작됩니다.

이러한 모든 경우 올바르게 정렬된 패킷 스트림이 이 인터페이스에서 전송됩니다. 프래그먼트가 수신되면 완전한 패킷이 형성되고 전송됩니다.

분산 기능 구성, 확인 및 디버깅

이 섹션에서는 각 분산 기능을 확인하고 디버깅하는 데 사용할 수 있는 **show** 및 **debug** 명령을 다룹니다.

dMFR 구성 및 확인

MFR 샘플 컨피그레이션

```
interface MFR1
 no ip address

interface MFR1.1 point-to-point
 ip address 181.0.0.2 255.255.0.0
 frame-relay interface-dlci 16
```

참고: MFR 인터페이스는 다른 FR 인터페이스와 유사하므로 대부분의 FR 컨피그레이션을 지원합니다.

```
interface Serial5/0/0/1:0
 no ip address
```

```
 encapsulation frame-relay MFR1
 tx-queue-limit 26

interface Serial5/0/0/2:0
 no ip address
 encapsulation frame-relay MFR1
 tx-queue-limit 26

interface Serial5/0/0/3:0
 no ip address
 encapsulation frame-relay MFR1
```

RP에서 MFR 번들 상태 확인

show frame-relay multilink

```
Bundle: MFR1, State = up, class = A, fragmentation disabled
BID = MFR1
Bundle links:
 Serial5/0/0/3:0, HW state = up, link state = Add_sent, LID = Serial5/0/0/3:0
 Serial5/0/0/2:0, HW state = up, link state = Up, LID = Serial5/0/0/2:0
 Serial5/0/0/1:0, HW state = up, link state = Up, LID = Serial5/0/0/1:0
```

이는 두 인터페이스가 올바르게 추가되었고 한 인터페이스가 아직 MLFR LIP 메시지를 협상하지 않았음을 나타냅니다.

MFR 번들 및 멤버 링크에 대한 자세한 내용을 보려면 다음 명령을 실행하십시오.

show frame-relay multilink mfr1 detailed

```
Bundle: MFR1, State = up, class = A, fragmentation disabled
BID = MFR1
No. of bundle links = 3, Peer's bundle-id = MFR1
Rx buffer size = 36144, Lost frag timeout = 1000
Bundle links:
 Serial5/0/0/3:0, HW state = up, link state = Add_sent, LID = Serial5/0/0/3:0
   Cause code = none, Ack timer = 4, Hello timer = 10,
   Max retry count = 2, Current count = 0,
   Peer LID = , RTT = 0 ms
   Statistics:
   Add_link sent = 35, Add_link rcv'd = 0,
   Add_link ack sent = 0, Add_link ack rcv'd = 0,
   Add_link rej sent = 0, Add_link rej rcv'd = 0,
   Remove_link sent = 0, Remove_link rcv'd = 0,
   Remove_link_ack sent = 0, Remove_link_ack rcv'd = 0,
   Hello sent = 0, Hello rcv'd = 0,
   Hello_ack sent = 0, Hello_ack rcv'd = 0,
   outgoing pak dropped = 0, incoming pak dropped = 0
  Serial5/0/0/2:0, HW state = up, link state = Up, LID = Serial5/0/0/2:0
   Cause code = none, Ack timer = 4, Hello timer = 10,
   Max retry count = 2, Current count = 0,
   Peer LID = Serial6/1/0/2:0, RTT = 32 ms
   Statistics:
   Add_link sent = 0, Add_link rcv'd = 0,
   Add_link ack sent = 0, Add_link ack rcv'd = 0,
   Add_link rej sent = 0, Add_link rej rcv'd = 0,
   Remove_link sent = 0, Remove_link rcv'd = 0,
   Remove_link_ack sent = 0, Remove_link_ack rcv'd = 0,
   Hello sent = 7851, Hello rcv'd = 7856,
```

```
   Hello_ack sent = 7856, Hello_ack rcv'd = 7851,
   outgoing pak dropped = 0, incoming pak dropped = 0
  Serial5/0/0/1:0, HW state = up, link state = Up, LID = Serial5/0/0/1:0
   Cause code = none, Ack timer = 4, Hello timer = 10,
   Max retry count = 2, Current count = 0,
   Peer LID = Serial6/1/0/1:0, RTT = 32 ms
   Statistics:
   Add_link sent = 0, Add_link rcv'd = 0,
   Add_link ack sent = 0, Add_link ack rcv'd = 0,
   Add_link rej sent = 0, Add_link rej rcv'd = 0,
   Remove_link sent = 0, Remove_link rcv'd = 0,
   Remove_link_ack sent = 0, Remove_link_ack rcv'd = 0,
   Hello sent = 7851, Hello rcv'd = 7856,
   Hello_ack sent = 7856, Hello_ack rcv'd = 7851,
   outgoing pak dropped = 0, incoming pak dropped = 0
```

MFR 디버그 명령

이러한 디버그는 링크가 번들에 추가되지 않는 문제를 해결하는 데 유용합니다.

debug frame-relay multilink control

참고: 특정 MFR 인터페이스 또는 Serial 인터페이스가 지정되지 않은 경우 모든 MFR 링크에 대한 디버그가 활성화됩니다. 라우터에 많은 MFR 링크가 있는 경우 이는 큰 부담이 될 수 있습니다.

RP에서 수신된 MFR 패킷을 디버깅하고 MFR 제어 활동을 디버깅하려면 이 디버그가 유용합니다.

debug frame-relay multilink

참고: 트래픽이 많을 경우 CPU가 마비될 수 있습니다.

LC에서 dMLFR 번들 상태 확인

show frame relay multilink

참고: 현재 LC에서는 제공되지 않지만 곧 추가됩니다. 그때까지 show ppp multilink를 사용합니다.

```
Bundle MFR1, 2 members
  bundle 0x62DBDD20, frag_mode 0
  tag vectors 0x604E8004 0x604C3628
  Bundle hwidb vector 0x6019271C
  idb MFR1, vc 0, RSP vc 0
  QoS disabled, fastsend (mlp_fastsend), visible_bandwidth 3072
  board_encap 0x60577554, hw_if_index 0, pak_to_host 0x0
  max_particles 400, mrru 1524, seq_window_size 0x200
  working_pak 0x0, working_pak_cache 0x0
  una_frag_list 0x0, una_frag_end 0x0, null_link 0
  rcved_end_bit 1, is_lost_frag 0, resync_count 0
  timeout 0, timer_start 0, timer_running 0, timer_count 0
  next_xmit_link Serial0/0:1, member 0x3, congestion 0x3
dmlp_orig_pak_to_host 0x603E7030
dmlp_orig_fastsend 0x6035DBC0
bundle_idb->lc_ip_turbo_fs 0x604A7750
```

```
  0 lost fragments, 0 reordered, 0 unassigned
  0 discarded, 0 lost received
  0x0 received sequence, 0x58E sent sequence
 DLCI: 16
  769719 lost fragments, 22338227 reordered,
                                0 unassigned
  27664 discarded, 27664 lost received
  0xF58 received sequence, 0x8DE sent sequence
 timer count 767176
  Member Link: 2 active
   Serial0/0:0, id 0x1, fastsend 0x60191E34, lc_turbo 0x601913AC, PTH 0x603E7030, OOF 0
   Serial0/0:1, id 0x2, fastsend 0x60191E34, lc_turbo 0x601913AC, PTH 0x603E7030, OOF 0
```

dMLP/dLFIoLL 구성 및 확인

멀티링크 PPP 컨피그레이션

```
interface Multilink1
 ip address 151.0.0.2 255.255.0.0
 no cdp enable
 ppp chap hostname M1
 ppp multilink
!
```

Serial 인터페이스의 샘플 컨피그레이션:

```
interface Serial5/0/0/4:0
 no ip address
 encapsulation ppp
 tx-queue-limit 26
 no cdp enable
 ppp chap hostname M1
 ppp multilink group 1
!
interface Serial5/0/0/5:0
 no ip address
 encapsulation ppp
 tx-queue-limit 26
 no cdp enable
 ppp chap hostname M1
 ppp multilink group 1
!
```

참고: ppp chap **hostname M1** 명령은 CHAP 인증이 활성화되었음을 의미하지는 않습니다. 이 명령의 문자열 **M1**은 엔드포인트 판별자 역할을 하며 동일한 두 라우터 간에 둘 이상의 멀티링크 번들이 있을 경우에만 필요합니다. 이러한 경우 번들에 속하는 모든 링크는 동일한 엔드포인트 판별자를 가져야 하며, 다른 번들에 속하는 두 링크는 동일한 엔드포인트 판별자를 가질 수 없습니다.

선택적 구성 매개변수

[아니요] ppp multilink interleave

이렇게 하면 멀티링크 번들에서 인터리빙이 활성화됩니다. 이는 모듈형 QoS CLI와 함께 작동합니다. 우선 순위가 높은 패킷은 MLP 시퀀스 및 헤더를 추가하지 않고 전송되는 반면, 다른 패킷은 프래그먼트화되어 MLP 시퀀스 및 헤더와 함께 전송됩니다.

참고: 둘 이상의 링크로 인터리빙이 활성화된 경우 우선순위가 높은 트래픽이 다시 주문될 수 있습니다. 인터리빙이 활성화되거나 비활성화된 경우, 라인 카드에서 활성화되려면 번들을 재설정해야

합니다.

ppp multilink mrru local *value*

멀티링크의 최대 수신 유닛을 지정합니다. 이 크기까지 패킷은 멀티링크 인터페이스에서 허용됩니다. 여기서는 MLP 헤더가 제외됩니다.

ppp multilink mrru remote *value*

원격 끝에서 지원해야 하는 최소 MRRU를 지정합니다. 원격 엔드 MRRU가 이 값보다 작은 경우 번들 협상이 실패합니다.

ppp multilink fragment delay *seconds*

데이터 프래그먼트로 인해 허용되는 지연 시간(밀리초)을 지정합니다. 즉, delay 값은 허용되는 최대 조각 크기를 계산하는 데 사용됩니다. 분산 구현은 다음과 같은 방식으로 Cisco IOS 구현과 다릅니다.

1. 인터리빙이 활성화되어 있지 않으면 조각화가 수행되지 않습니다.
2. 대역폭이 다른 링크의 경우 선택한 프래그먼트 크기는 최소 대역폭 인터페이스를 기반으로 합니다.

ppp multilink fragment disable

이 명령은 분산 구현에 어떤 기능도 추가하지 않습니다. 조각화는 인터리빙이 활성화된 경우에만 발생합니다. 인터리빙이 활성화된 경우 ppp multilink fragment disable 명령**은** 무시됩니다.

RP에서 dMLP 번들 상태 확인

show ppp multilink

```
Multilink1, bundle name is M1
Endpoint discriminator is M1
Bundle up for 00:09:09, 1/255 load
Receive buffer limit 24000 bytes, frag timeout 1000 ms
Bundle is Distributed
  0/0 fragments/bytes in reassembly list
  0 lost fragments, 0 reordered
  0/0 discarded fragments/bytes, 0 lost received
  0x9 received sequence, 0x0 sent sequence
dLFI statistics:
          DLFI Packets     Pkts In   Chars In   Pkts Out  Chars Out
            Fragmented           0          0          0          0
          UnFragmented           9       3150          0          0
           Reassembled           9       3150          0          0
      Reassembly Drops           0
```

```
   Fragmentation Drops          0
      Out of Seq Frags          0
Member links: 2 active, 0 inactive (max not set, min not set)
  Se5/0/0/4:0, since 00:09:09, 768 weight, 760 frag size
  Se5/0/0/5:0, since 00:09:09, 768 weight, 760 frag size
```

1. 번들이 분산 모드인 경우 **show ppp multilink** 출력에 **표시됩니다**. 그렇지 않으면 어떤 이유로 인해 번들이 배포되지 않습니다.
2. 라인 카드**에서 ppp multilink 인터리브**를 구성하고 활성화하면 **show ppp multilink** 출력에 dLFI .Fragmented(조각화) - 전송 및 수신된 프래그먼트의 수를 나타냅니다.Unfragmented(조각화되지 않음) - 프래그먼트화되지 않고 전송되거나 수신된 패킷의 수를 나타냅니다.Reassembled(리어셈블됨) - 리어셈블된 전체 패킷 수를 나타냅니다. 인터리빙이 활성화되지 않은 경우 출력은 다음과 같습니다.

```
Multilink1, bundle name is M1
  Endpoint discriminator is M1
  Bundle up for 00:00:00, 0/255 load
  Receive buffer limit 24000 bytes, frag timeout 1000 ms
  Bundle is Distributed
    0/0 fragments/bytes in reassembly list
    0 lost fragments, 0 reordered
    0/0 discarded fragments/bytes, 0 lost received
    0x0 received sequence, 0x2 sent sequence
  Member links: 2 active, 0 inactive (max not set, min not set)
    Se5/0/0/5:0, since 00:00:00, 768 weight, 760 frag size
    Se5/0/0/4:0, since 00:00:03, 768 weight, 760 frag size
```

이전 예의 프래그먼트 크기는 760바이트입니다.

LC에서 dMLP 번들 상태 확인

show ppp multilink

```
dmlp_ipc_config_count 24
dmlp_bundle_count 2
dmlp_ipc_fault_count 1
dmlp_il_inst 0x60EE4340, item count 0
0, store 0, hwidb 0x615960E0, bundle 0x622AA060, 0x60579290, 0x6057A29C
1, store 0, hwidb 0x615985C0, bundle 0x622AA060, 0x60579290, 0x6057A29C
2, store 0, hwidb 0x0, bundle 0x0,
3, store 0, hwidb 0x0, bundle 0x0,
4, store 0, hwidb 0x0, bundle 0x0,
5, store 0, hwidb 0x0, bundle 0x0,
6, store 0, hwidb 0x0, bundle 0x0,
7, store 0, hwidb 0x0, bundle 0x0,
8, store 0, hwidb 0x0, bundle 0x0,
9, store 0, hwidb 0x0, bundle 0x0,

Bundle Multilink1, 2 members
  bundle 0x622AA060, frag_mode 0
  tag vectors 0x604E8004 0x604C3628
  Bundle hwidb vector 0x6057B198
  idb Multilink1, vc 4, RSP vc 4
  QoS disabled, fastsend (qos_fastsend), visible_bandwidth 3072
  board_encap 0x60577554, hw_if_index 0, pak_to_host 0x0
  max_particles 400, mrru 1524, seq_window_size 0x8000
  working_pak 0x0, working_pak_cache 0x0
  una_frag_list 0x0, una_frag_end 0x0, null_link 0
```

```
  rcved_end_bit 1, is_lost_frag 1, resync_count 0
  timeout 0, timer_start 0, timer_running 0, timer_count 1
  next_xmit_link Serial0/0:3, member 0x3, congestion 0x3
dmlp_orig_pak_to_host 0x603E7030
dmlp_orig_fastsend 0x6035DBC0
bundle_idb->lc_ip_turbo_fs 0x604A7750
  0 lost fragments, 0 reordered, 0 unassigned
  0 discarded, 0 lost received
  0xC3 received sequence, 0x0 sent sequence
  Member Link: 2 active
   Serial0/0:4, id 0x1, fastsend 0x60579290, lc_turbo 0x6057A29C, PTH 0x60579A18, OOF 0
   Serial0/0:3, id 0x2, fastsend 0x60579290, lc_turbo 0x6057A29C, PTH 0x60579A18, OOF 0
```

dMFR에서는 시퀀스 번호가 dLCI별로 유지되며, 번들의 시퀀스 번호는 LMI dLCI에 사용됩니다.

필드	설명
dmlp_ipc_config_count	멀티링크 또는 MLFR 컨피그레이션에 대해 LC에서 받은 IPC 메시지 수
dmlp_bundle_count	LC의 MLP 및 MLFR 번들 수
dmlp_ipc_fault_count	LC에서 장애가 발생한 구성 메시지 수입니다. 0이어야 합니다. 0이 아니면 문제가 있을 수 있습니다.
	tag_optimal_fs에 대한 idb와 태그 스위칭에 사용되는 idb to ip2tag_optimal_fs 벡터를 나타냅니다.
/	7500 플랫폼에 변경 링크가 있는 경우 보드 캡슐화의 2바이트를 추가하는 데 사용되는 board_encap 벡터를 나타냅니다. 링크에가 인터페이스가 포함된 경우 NULL이어야 합니다.
	리어셈블리 버퍼에 저장할 수 있는 최대 입자 수
mru	MLP 캡슐화를 고려하지 않고 허용되는 패킷의 최대 크기입니다. MLFR 인터페이스에는 해당되지 않습니다.
__	시퀀스 번호의 최대 창 크기
_pak	리어셈블리 중인 현재 pak를 나타냅니다. NULL(없는 경우)
_pak_cache	재어셈블리에 사용되는 고정 팩에 대한 포인터입니다. 이는 번들에서 첫 번째 불완전한 패킷을 수신할 때 할당됩니다.
una_frag_	리어셈블리 대기열의 첫 번째 항목입니다. 항목이 NULL 아니며 변경되지 않으면 타이머가 소프트웨어 문제를 실행하고 있지 않음을 나타냅니다.
una_frag_end	리어셈블리 대기열의 마지막 항

	목
rcved_end_bit	번들에 종료 비트가 수신되었으므로 시작 비트를 찾고 있음을 나타냅니다.
is_lost_frag	프래그먼트가 lost로 선언된 경우 true입니다. 예상 시퀀스의 조각을 받으면 이 값이 지워집니다.
_	수신기가 송신기와 동기화되지 않았으며 마지막으로 수신한 순차 프래그먼트로 시작하여 재동기화해야 하는 횟수를 나타냅니다.
	리어셈블리 시간 초과가 발생했고 리어셈블리 대기열에서 패킷이 처리되고 있음을 나타냅니다.
_	리어셈블리 타이머가 시작된 횟수
_	리어셈블리 타이머가 실행 중인지 여부를 나타냅니다.
_	리어셈블리 타이머가 만료된 횟수를 나타냅니다.
_xmit_link	다음 패킷을 전송할 링크
	존재하는 멤버를 나타내는 비트 필드입니다.
	모든 분기에서 필드가 사용되지 않습니다. 혼잡하지 않은 멤버 링크를 나타냅니다.
dmlp_pak_to_host	RP에 패킷을 푸시하는 데 사용되는 벡터입니다.
dmlp_fastsend	MLP 또는 MLFR이 드라이버의 fastsend를 수정하기 전에 원래 드라이버가 fastsend를 수행합니다.
	손실된 프래그먼트 수(수신자가 이러한 프래그먼트를 받지 못했습니다). 호스트에 업데이트가 전송될 때 주기적으로 지워집니다.
	예상 주문에서 수신된 프래그먼트 수입니다. 호스트에 업데이트가 전송될 때 주기적으로 지워집니다.
	전체 패킷을 만들 수 없어 삭제된 프래그먼트 수
	수신된 프래그먼트의 수. 이는 링크 간 지연이 dMLP 리어셈블리 시간 초과인 30ms보다 큰것임을 나타냅니다.

dLFIoFR 및 dLFIoATM 구성 및 확인

```
class-map voip
 match ip precedence 3

policy-map llq
 class voip
  priority

int virtual-template1
 service-policy output llq
 bandwidth 78
 ppp multilink
 ppp multilink interleave
 ppp multilink fragment-delay 8


int serial5/0/0/6:0
encapsulation frame-relay
frame-relay interface-dlci 16 ppp virtual-template1
!--- Or

int ATM4/0/0
  no ip address
int ATM4/0/0.1 point-to-point
  pvc 5/100
  protocol ppp virtual-template 1
```

RP에서 dLFIoFR/ATM 번들 상태 확인

```
show ppp multilink

Virtual-Access3, bundle name is dLFI
  Endpoint discriminator is dLFI
  Bundle up for 00:01:11, 1/255 load
  Receive buffer limit 12192 bytes, frag timeout 1524 ms
  Bundle is Distributed
    0/0 fragments/bytes in reassembly list
    0 lost fragments, 0 reordered
    0/0 discarded fragments/bytes, 0 lost received
    0x0 received sequence, 0x0 sent sequence
  dLFI statistics:
            DLFI Packets     Pkts In    Chars In    Pkts Out   Chars Out
              Fragmented           0           0           0           0
            UnFragmented           0           0           0           0
             Reassembled           0           0           0           0
        Reassembly Drops           0
     Fragmentation Drops           0
        Out of Seq Frags           0
  Member links: 1 (max not set, min not set)
    Vi2, since 00:01:11, 240 weight, 230 frag size
```

참고: 번들은 가상 템플릿에서 **ppp multilink interleave**가 구성된 경우에만 배포됩니다. 이 명령을 사용하지 않으면 번들이 배포되지 않습니다.

LC에서 dLFIoFR/ATM 번들 상태 확인

dLFI가 LC에서 제대로 작동하는지 확인하려면 다음 명령을 실행합니다.

show hqf interface

```
Interface Number 6 (type 22) Serial0/0:5

   blt (0x62D622E8, index 0, hwidb->fast_if_number=35) layer PHYSICAL
   scheduling policy: FIFO
   classification policy: NONE
   drop policy: TAIL
   blt flags: 0x0

   qsize 0 txcount 3 drops 0 qdrops 0 nobuffers 0
   aggregate limit 16 individual limit 4 availbuffers 16
   weight 1 perc 0.00 ready 1 shape_ready 1 wfq_clitype 0
   visible_bw 64 allocated_bw 64 qlimit_tuned 0 vc_encap 2
   quantum 1500 credit 0 backpressure_policy 0 nothingoncalQ 1

     next layer HQFLAYER_FRAMEDLCI_IFC (max entries 1024)

     blt (0x62D620E8, index 0, hwidb->fast_if_number=35) layer FRAMEDLCI_IFC
     scheduling policy: FIFO
     classification policy: NONE
     drop policy: TAIL
     blt flags: 0x0

     qsize 0 txcount 1 drops 0 qdrops 0 nobuffers 0
     aggregate limit 16 individual limit 4 availbuffers 16
     weight 1 perc 0.00 ready 1 shape_ready 1 wfq_clitype 0
     visible_bw 64 allocated_bw 64 qlimit_tuned 0 vc_encap 2
     quantum 1500 credit 0 backpressure_policy 0 nothingoncalQ 1

     blt (0x62D621E8, index 16, hwidb->fast_if_number=35) layer FRAMEDLCI_IFC
     scheduling policy: WFQ
     classification policy: PRIORITY_BASED
     drop policy: TAIL
     frag policy: root
     blt flags: 0x0

     qsize 0 txcount 2 drops 0 qdrops 0 nobuffers 0
     aggregate limit 16 individual limit 4 availbuffers 16
     weight 1 perc 0.00 ready 1 shape_ready 1 wfq_clitype 0
     visible_bw 64 allocated_bw 64 qlimit_tuned 0 vc_encap 2
     quantum 240 credit 0 backpressure_policy 0 nothingoncalQ 1

       next layer HQFLAYER_PRIORITY (max entries 256)

       blt (0x62D61FE8, index 0, hwidb->fast_if_number=35) layer PRIORITY
       scheduling policy: FIFO
       classification policy: NONE
       drop policy: TAIL
       frag policy: leaf
       blt flags: 0x0

       qsize 0 txcount 0 drops 0 qdrops 0 nobuffers 0
       aggregate limit 8 individual limit 2 availbuffers 8
       weight 0 perc 0.99 ready 1 shape_ready 1 wfq_clitype 0
       visible_bw 32 allocated_bw 32 qlimit_tuned 0 vc_encap 2
       quantum 240 credit 0 backpressure_policy 0 nothingoncalQ 1

       blt (0x62D61CE8, index 1, hwidb->fast_if_number=35) layer PRIORITY
```

```
scheduling policy: FIFO
classification policy: NONE
drop policy: TAIL
blt flags: 0x0
Priority Conditioning enabled
qsize 0 txcount 0 drops 0 qdrops 0 nobuffers 0
aggregate limit 0 individual limit 0 availbuffers 0
weight 1 perc 0.00 ready 1 shape_ready 1 wfq_clitype 0
visible_bw 0 allocated_bw 0 qlimit_tuned 0 vc_encap 2
quantum 240 credit 0 backpressure_policy 0 nothingoncalQ 1

PRIORITY: bandwidth 32 (50%)
          last 0 tokens 1500 token_limit 1500

blt (0x62D61EE8, index 255, hwidb->fast_if_number=35) layer PRIORITY
scheduling policy: WFQ
classification policy: CLASS_BASED
drop policy: TAIL
frag policy: MLPPP (1)
  frag size: 240, vc encap: 0, handle: 0x612E1320
blt flags: 0x0

qsize 0 txcount 2 drops 0 qdrops 0 nobuffers 0
aggregate limit 8 individual limit 2 availbuffers 8
weight 1 perc 0.01 ready 1 shape_ready 1 wfq_clitype 0
visible_bw 32 allocated_bw 32 qlimit_tuned 0 vc_encap 2
quantum 1 credit 0 backpressure_policy 0 nothingoncalQ 1

    next layer HQFLAYER_CLASS_HIER0 (max entries 256)

    blt (0x62D61DE8, index 0, hwidb->fast_if_number=35) layer CLASS_HIER0
    scheduling policy: FIFO
    classification policy: NONE
    drop policy: TAIL
    frag policy: leaf
    blt flags: 0x0

    qsize 0 txcount 2 drops 0 qdrops 0 nobuffers 0
    aggregate limit 8 individual limit 2 availbuffers 8
    weight 1 perc 50.00 ready 1 shape_ready 1 wfq_clitype 0
    visible_bw 32 allocated_bw 32 qlimit_tuned 0 vc_encap 2
    quantum 240 credit 0 backpressure_policy 0 nothingoncalQ 1
```

우선 순위 레이어와 WFQ 레이어가 있어야 합니다. 조각화는 WFQ 리프 레이어 블릿에서 수행됩니다.

dDDR 구성 및 확인

분산 DDR은 다이얼러 인터페이스에 **분산된** 전역 컨피그레이션 및 **ip route-cache**에 **분산된 ip cef**를 활성화하면 활성화됩니다.

```
!--- Global configuration that enables distributed CEF switching. ip cef distributed --- Enable
distributed switching on the dialer interface (on the D-channel interface). int serial 3/1/0:23
ip route-cache distributed !--- Or, enable it on the dialer interface. int Dialer1 ip route-
cache distributed
```

분산 DDR에 대한 다른 특별한 구성은 없습니다. 일반 DDR 구성을 따릅니다.

분산형 Dial on Demand 라우팅 확인

```
BOX2002# show isdn status

Global ISDN Switchtype = primary-net5
ISDN Serial3/1/0:23 interface
--- Network side configuration. dsl 0, interface ISDN Switchtype = primary-net5 Layer 1 Status:
ACTIVE Layer 2 Status: TEI = 0, Ces = 1, SAPI = 0, State = MULTIPLE_FRAME_ESTABLISHED

The ISDN status should be MULTIPLE_FRAME_ESTABLISHED. This means that the physical layer is
ready for ISDN connectivity. Layer 3 Status: 0 Active Layer 3 Call(s) Active dsl 0 CCBs = 0 The
Free Channel Mask: 0x807FFFFF Number of L2 Discards = 0, L2 Session ID = 6 EDGE# show dialer

Serial6/0:0 - dialer type = ISDN
Idle timer (120 secs), Fast idle timer (20 secs)
Wait for carrier (30 secs), Re-enable (15 secs)
Dialer state is data link layer up
Time until disconnect 119 secs
Current call connected never
Connected to 54321

Serial6/0:1 - dialer type = ISDN
Idle timer (120 secs), Fast idle timer (20 secs)
Wait for carrier (30 secs), Re-enable (15 secs)
Dialer state is idle
```

다이얼러 은 사용된 다이얼러의 유형을 알려줍니다. ISDN은 레거시 다이얼러 컨피그레이션을 의미하며 PROFILE 프로파일 컨피그레이션을 의미합니다. 다이얼러 는 다이얼러의 현재 상태를 나타냅니다. 연결되지 않은 다이얼러 인터페이스의 상태가 상태가 됩니다. Idle 는 흥미로운 트래픽이 표시될 때마다 재설정됩니다. 이 타이머가 만료되면 즉시 인터페이스의 연결이 끊깁니다. 는 구성 가능한 매개변수입니다. 자세한 내용은 다이얼러 프로파일을 사용하여 피어 투 피어 DDR 구성을 참조하십시오.

```
show ppp multilink
!--- From LC for dialer profile. dmlp_ipc_config_count 2 dmlp_bundle_count 1 dmlp_il_inst
0x60EE4340, item count 0 0, store 0, hwidb 0x0, bundle 0x0, 1, store 0, hwidb 0x0, bundle 0x0,
2, store 0, hwidb 0x0, bundle 0x0, 3, store 0, hwidb 0x0, bundle 0x0, 4, store 0, hwidb 0x0,
bundle 0x0, 5, store 0, hwidb 0x0, bundle 0x0, 6, store 0, hwidb 0x0, bundle 0x0, 7, store 0,
hwidb 0x0, bundle 0x0, 8, store 0, hwidb 0x0, bundle 0x0, 9, store 0, hwidb 0x0, bundle 0x0,
Bundle Dialer1, 1 member bundle 0x62677220, frag_mode 0 tag vectors 0x604E8004 0x604C3628 Bundle
hwidb vector 0x0 idb Dialer1, vc 22, RSP vc 22 QoS disabled, fastsend (mlp_fastsend),
visible_bandwidth 56 board_encap 0x60577554, hw_if_index 0, pak_to_host 0x0 max_particles 200,
mrru 1524, seq_window_size 0x8000 working_pak 0x0, working_pak_cache 0x0 una_frag_list 0x0,
una_frag_end 0x0, null_link 0 rcved_end_bit 1, is_lost_frag 0, resync_count 0 timeout 0,
timer_start 0, timer_running 0, timer_count 0 next_xmit_link Serial1/0:22, member 0x1,
congestion 0x1 dmlp_orig_pak_to_host 0x603E7030 dmlp_orig_fastsend 0x60381298 bundle_idb-
>lc_ip_turbo_fs 0x604A7750 0 lost fragments, 0 reordered, 0 unassigned 0 discarded, 0 lost
received 0x0 received sequence, 0x0 sent sequence Member Link: 1 active Serial1/0:22, id 0x1,
fastsend 0x60579290, lc_turbo 0x6057A29C, PTH 0x60579A18, OOF 0
```

표시된 변수는 dMLP와 동일합니다.

dMLP 및 dDDR 디버깅

RP에서 사용할 수 있는 디버그

dDDR

debug dialer [events | packets | forwarding | map]

통화 설정 등의 제어 경로 기능을 디버깅하려면 이 명령을 실행합니다. 자세한 내용은 디버그 다이얼러 이벤트를 참조하십시오.

debug ip cef dialer

CEF 관련 다이얼러 이벤트를 디버깅하려면 이 명령을 실행합니다. 자세한 내용은 다이얼러 CEF를 참조하십시오.

LC에서 제공되는 디버깅

dMLP

제어 경로 디버깅: **멀티링크 이벤트 디버그**

데이터 경로 디버깅: **멀티링크 조각 디버깅**

데이터 경로 및 제어 경로 오류 디버깅: **디버그 멀티링크 오류**

SIP 라인 카드에서 dMLP 디버깅

CI를 기반으로 패킷 덤핑: 데이터 패킷 및 제어 패킷은 제어 ci 및 시퀀스 ci를 기반으로 라인 카드에 덤프할 수 있습니다.

hw-module *subslot_num* 덤프 *ci CI-NUM* [rx|tx] *num_packets_to_dump* 테스트

CI는 다음과 같은 방법으로 얻을 수 있습니다.

```
!--- Issue show controller serial interface  for CTE1.

SIP-200-6# show controller serial 6/0/0:0

SPA 6/0 base address 0xB8000000 efc 1

Interface Serial6/0/0:0 is administratively down
Type 0xD Map 0x7FFFFFFF, Subrate 0xFF, mapped 0x1, maxmtu 0x5DC
Mtu 1500, max_buffer_size 1524, max_pak_size 1608 enc 84
ROM rev: 0, FW OS rev: 0x00000000 Firmware rev: 0x00000000
  idb=0x42663A30, pa=0x427BF6E0, vip_fci_type=0, port_per_spa=0
  SPA port type is set
  Host SPI4 in sync
  SPA=0x427BF6E0 status=00010407, host=00000101, fpga=0x427EDF98
  cmd_head=113, cmd_tail=113, ev_head=184, ev_tail=184
  ev_dropped=0, cmd_dropped=0
!--- Start Link Record Information.  tag 0, id 0, anyphy 0, anyphy_flags 3, state 0
 crc 0, idle 0, subrate 0, invert 0, priority 0
 encap hdlc
 corrupt_ci 65535, transparent_ci 1
!--- End Link Record Information. Interface Serial6/0/0:0 is administratively down Channel
```

```
Stats: in_throttle=0, throttled=0, unthrottled=0, started=1 rx_packets=0, rx_bytes=0,
rx_frame_aborts=0, rx_crc_errors=0 rx_giants=0, rx_non_aligned_frames=0, rx_runts=0,
rx_overruns=0 tx_packets=0, tx_bytes=0, tx_frame_aborts=0 is_congested=0, mapped=1, is_isdn_d=0,
tx_limited=1 fast_if_number=15, fastsend=0x403339E4 map=0x7FFFFFFF, turbo_vector_name=Copperhead
to Draco switching lc_ip_turbo_fs=403A9EEC, lc_ip_mdfs=403A9EEC
```

CT3의 경우 **show interface serial *CT3_interface_name***의 출력에서 가져올 수 있는 `vc num`을 받아야 합니다.

이제 SPA 콘솔에서 CI 정보를 얻을 수 있습니다. 먼저 spa_redirect rp ct3_freedm336 명령을 사용하여 SPA 콘솔 명령**의 출력을 RP로** 리디렉션합니다.

spa**_ct3_test freedm show linkrec *vc* 명령**은 필요한 CI 정보를 표시합니다.

dMFR

제어 경로 디버깅: **debug dmfr 이벤트**

데이터 경로 디버깅: **dmfr 패킷 디버그**

데이터 경로 및 제어 경로 오류 디버깅: **dmfr 디버그 오류**

CI를 기반으로 패킷 덤핑: dMLP를 참조하십시오.

LFI

제어 경로 디버깅: **debug dlfi 이벤트**

데이터 경로 디버깅: **디버그 dlfi 프래그먼트**

데이터 경로 및 제어 경로 오류 디버깅: **디버그 dll 오류**

dDDR

특수한 디버깅 명령은 없습니다. dMLP 디버깅을 사용해야 합니다.

dLFIoLL의 경우 dMLP 및 dLFI 디버그를 모두 사용해야 할 수 있습니다. 이러한 디버그는 조건적이지 않으므로 모든 번들에 대해 트리거됩니다.

자주 묻는 질문(FAQ)

1. **dMLP란?**dMLP는 Distributed Multilink PPP의 약어입니다(RFC1990 참조). 이 기능은 Cisco 7500 Series 및 7600 Series와 같은 분산 플랫폼에서 지원됩니다. dMLP를 사용하면 T1/E1 라인을 Cisco 7500 Series 라우터의 VIP에 결합하거나 7600 Series 라우터의 FlexWAN에 결합할 수 있습니다. 이 번들은 여러 T1/E1 라인의 대역폭을 결합한 것입니다. 이를 통해 고객은 T3/E3 라인 구입 없이 T1/E1 이상으로 대역폭을 늘릴 수 있습니다.
2. **dMLP에서 "분산"이란?**"분산"이란 패킷 스위칭이 RSP가 아니라 VIP에 의해 수행됨을 의미합니다. 왜? RSP 스위칭 기능은 다소 제한적이며 더 많은 중요한 작업을 수행해야 합니다. 패킷을 스위칭할 수 있는 VIP는 RSP에서 이 활동을 오프로드합니다. RSP 기반 Cisco IOS는 여전히 링크를 관리합니다. 번들 생성 및 해체 작업은 RSP에서 수행합니다. 또한 모든 PPP 제어 패킷(LCP, 인증 및 NCP)의 처리를 포함하여 PPP 제어 평면 처리는 RSP에서 계속 수행됩니다. 그러나 번들이 설정되면 온보드 CPU를 통해 스위칭하기 위해 MLP 패킷의 처리가 VIP로

전환됩니다. VIP의 dMLP 엔진은 조각화, 인터리빙, 캡슐화, 여러 링크 간 로드 밸런싱, 인바운드 프래그먼트의 정렬 및 리어셈블리 등 모든 MLP 절차를 처리합니다. 7500 시스템에서 VIP가 수행하는 기능은 7600 기반 시스템의 Flexwan/Enhanced-FlexWAN에서 수행됩니다.

3. **번들이 배포되었는지 여부를 어떻게 알 수 있습니까?**라우터 콘솔에서 **show ppp multilink** 명령을 실행합니다.

```
Router# show ppp multilink

Multilink1, bundle name is udho2
  Bundle up for 00:22:46
  Bundle is Distributed
  174466 lost fragments, 95613607 reordered, 129 unassigned
  37803885 discarded, 37803879 lost received, 208/255 load
  0x4D987C received sequence, 0x9A7504 sent sequence
  Member links: 28 active, 0 inactive (max not set, min not set)
    Se11/1/0/27:0, since 00:22:46, no frags rcvd
    Se11/1/0/25:0, since 00:22:46, no frags rcvd
!--- Output suppressed.
```

4. **RSP16 또는 SUP720으로 업그레이드할 경우 dMLP 성능이 향상됩니까?**아니요. dMLP(또는 모든 분산 기능)의 스위칭 성능은 해당 VIP 또는 FlexWAN에 따라 다릅니다. 예를 들어 VIP6-80의 성능은 VIP2-50의 성능보다 우수합니다.
5. **이 기능에 사용할 수 있는 PA는 무엇입니까?**PA-MC-T3PA-MC-2T3+PA-MC-E3PA-MC-2E1PA-MC-2T1PA-MC-4T1PA-MC-8T1PA-MC-8E1PA-MC-STM-1PA-MC-8TE1+PA-4T+PA-8TCT3IP-50(7500만 해당)
6. **단일 번들에서 구성할 수 있는 링크 수는 몇 개입니까?**이 대답에는 여러 가지 측면이 있습니다. 기본 병목 현상은 라인 카드의 CPU 전력(VIP/FlexWAN/Enhanced-FlexWAN2)입니다. 하드 제한은 번들당 56개의 링크이지만, CPU 전원 또는 제한된 버퍼로 인해 이러한 수를 구성할 수 없거나 트래픽 스위칭이 너무 많습니다. 이 수치는 이 지침을 기반으로 합니다(VIP/FlexWAN/Enhanced-FlexWAN2의 CPU 및 메모리 기준).VIP2-50(4MB SRAM 포함) 최대 T1s = 12VIP2-50(8MB SRAM 포함) 최대 T1s = 16VIP4-80 최대 T1s = 40VIP6-80 최대 T1s = 40FlexWAN 최대 T1s = 곧 업데이트됨Enhanced-FlexWAN 최대 E1s = 베이당 21개의 E1s(라인 카드당 총 42개의 E1s)
7. **각각 3개의 T1로 3개의 번들을 구성하거나 9개의 T1로 1개의 번들을 구성하면 성능이 변경됩니까?**랩 테스트에서 입증된 것처럼 성능에는 변화가 없습니다. 그러나 단일 번들에 T1이 많은 경우(예: 단일 번들의 경우 24개 또는 28개의 T1이라고 함), 버퍼 부족 문제가 있습니다. 단일 번들에 8개 이상의 멤버 링크(T1/E1)를 포함하지 않는 것이 좋습니다.
8. **번들의 대역폭은 어떻게 결정됩니까?**번들의 대역폭을 구성하지 않습니다. 모든 멤버 링크의 총 대역폭입니다. 번들에 4개의 T1이 있는 경우 번들의 대역폭은 6.144Mbps입니다.
9. **어떤 것이 더 나을까요? CEF 로드 밸런싱 또는 dMLP?**이에 대한 간단한 답은 없습니다. 어떤 것이 더 좋은지 여러분의 필요에 따라 결정하세요.**MLP의 장점:**CEF 로드 밸런싱은 IP 트래픽에만 적용됩니다. MLP는 번들을 통해 전송되는 모든 트래픽의 균형을 유지합니다.MLP는 패킷의 순서를 유지합니다. IP 자체도 주문 변경에 대해 허용되므로 이는 상관없습니다. 실제로, MLP를 피하기 위해 순서를 유지하는 데 드는 추가 비용이 발생할 수 있습니다. IP는 데이터 그램을 주문하지 않을 수 있는 네트워크를 위한 것이며, IP를 사용하는 모든 것은 재주문을 처리할 수 있어야 합니다. 그러나 이러한 사실에도 불구하고, 현실은 재정립이 여전히 실질적인 문제를 제기할 수 있다는 것이다.MLP는 피어 시스템에 대한 단일 논리적 연결을 제공합니다.QoS는 멀티링크 번들에서 지원됩니다.MLP는 사용자가 현재 요구 사항에 따라 멤버 링크를 추가하거나 제거할 수 있으므로 동적 대역폭 기능을 제공합니다.MLP는 더 많은 수의 링크를 번들링할 수 있는 반면, CEF 로드 밸런싱은 6개의 병렬 IP 경로로 제한됩니다.플로우별 CEF 로드 밸런싱은 지정된 플로우의 최대 대역폭을 하나의 T1로 제한합니다. 예를 들어 음성 게이트웨이를 사용하는 고객은 소스와 대상이 같은 많은 통화를 가질 수 있으므로 하나의 경로만 사용할 수 있습니다.**MLP의 단점:**MLP는 각 패킷 또는 프레임에 추가 오버헤드를 추가합니다

.MLP는 CPU를 많이 사용합니다. dMLP는 라인 카드 CPU를 많이 사용합니다.

10. **두 라우터 간에 여러 번들을 구성하려면 어떻게 해야 합니까?**Multilink는 피어의 이름 및 엔드 포인트 판별자를 기반으로 링크가 조인할 번들을 결정합니다. 두 시스템 간에 서로 다른 번들을 여러 개 생성하기 위해 표준 방법은 일부 링크가 서로 다르게 식별되도록 하는 것입니다. 권장되는 방법은 ppp chap hostname name *name* ***명령***을 사용하는 것입니다.
11. **다른 PA의 구성원 링크를 가질 수 있습니까?**아니요. dMLP를 실행하려면 지원되지 않습니다. 그러나 다른 PA에서 멤버 링크가 추가되면 RSP에 컨트롤이 부여되고 더 이상 dMLP가 아닙니다. MLP는 여전히 작동하고 있지만 dMLP의 이점은 사라졌습니다.
12. **두 베이에서 멤버 링크를 혼합할 수 있습니까?**아니요. dMLP를 실행하려면 지원되지 않습니다. 그러나 다른 PA에서 멤버 링크가 추가되면 RSP에 컨트롤이 지정되고 더 이상 dMLP가 아닙니다. MLP는 여전히 작동하고 있지만 dMLP의 이점은 사라졌습니다.
13. **여러 VIP 또는 FlexWAN에 멤버 링크를 제공할 수 있습니까?**아니요. dMLP를 실행하려면 지원되지 않습니다. 그러나 다른 PA에서 멤버 링크가 추가되면 RSP에 컨트롤이 부여되고 더 이상 dMLP가 아닙니다. MLP는 여전히 작동하고 있지만 dMLP의 이점은 사라졌습니다.
14. **단일 PA에서 여러 포트에 멤버 링크를 가질 수 있습니까?**(예: PA-MC-2T3+의 각 CT3 포트에서 멤버 링크 1개)예. 동일한 PA에서 나온 한 문제가 없습니다.
15. **T3 또는 E3 포트를 번들로 구성할 수 있습니까?**아니요. 7500/VIP, 7600/FlexWAN 및 7600/FlexWAN2의 경우 dMLP에서는 DS0, n*DS0, T1 및 E1 속도만 허용됩니다.**참고:** 분산 MLPPP는 T1/E1 또는 하위 T1/E1 속도로 구성된 멤버 링크에만 지원됩니다. Channelized STM-1/T3/T1 인터페이스는 T1/E1 또는 서브레이트 T1/E1 속도에서 dMLPPP를 지원합니다. 일반 채널 T3/E3 이상의 인터페이스 속도로 구성된 멤버 링크에는 분산 MLPPP가 지원되지 않습니다.
16. **"재주문된" 프래그먼트란 무엇입니까?**수신된 프래그먼트 또는 패킷이 예상 시퀀스 번호와 일치하지 않으면 순서가 카운터가 증가합니다. 다양한 패킷 크기의 경우 이 문제가 발생할 수 있습니다. 고정 크기 패킷의 경우, PA 드라이버가 하나의 링크에서 수신한 패킷을 처리하고 라운드 로빈(패킷을 전송하는 동안 dMLP에서 수행되는 것처럼) 기반으로 진행되지 않기 때문에 이러한 문제가 발생할 수도 있습니다. 재주문한다고 해서 패킷 손실을 의미하지는 않습니다.
17. **"분실" 프래그먼트는 무엇입니까?**프래그먼트나 패킷이 주문에서 수신되고 모든 링크에서 순서가 잘못된 프래그먼트나 패킷이 수신되는 것을 알 때마다 카운터가 증가합니다. 또 다른 경우는 주문 외 프래그먼트가 목록에 저장되고 이 프래그먼트가 한도(VIP의 SRAM 및 번들에 대해 할당된 것에 따라 결정됨)에 도달하면 손실된 프래그먼트 카운터가 증가하며 목록의 다음 시퀀스 번호가 처리됩니다.
18. **dMLP는 손실된 부분을 어떻게 탐지합니까?**시퀀스 번호: 시퀀스 번호가 N인 프래그먼트가 도착하기를 기다리고 있고 모든 링크에서 시퀀스 번호가 N보다 높은 프래그먼트를 수신하는 경우, 동일한 링크에서 더 높은 번호의 프래그먼트들 뒤에 합법적으로 도착할 수 없기 때문에 프래그먼트 N이 손실되어야 한다는 것을 알고 있습니다.시간 초과: 조각을 너무 오래 기다리면, 결국 그것을 상실이라고 선언하고 계속 진행합니다.리어셈블리 버퍼 오버플로: 프래그먼트 N이 도착하기를 기다리는 동안 다른 프래그먼트(시퀀스 번호가 N보다 높음)가 일부 링크에 도달하는 경우 프래그먼트 N이 나타날 때까지 리어셈블리 버퍼에 해당 프래그먼트를 파크해야 합니다. 버퍼링할 수 있는 양에 제한이 있습니다. 버퍼가 오버플로되면 프래그먼트 N을 손실됨으로 다시 선언하고 버퍼에 있는 모든 것으로 처리를 재개합니다.
19. **"상실된 수신됨"이란 무엇입니까?**수신된 프래그먼트나 패킷이 손실된 이유는 두 가지가 있습니다.수신된 프래그먼트 또는 패킷이 예상 시퀀스 범위 창을 벗어나는 경우, 수신된 것으로 표시하여 패킷이 삭제됩니다.수신된 프래그먼트 또는 패킷이 예상 시퀀스 범위 창 내에 있지만 패킷 헤더를 이 패킷의 상위를 재지정하기 위해 할당할 수 없는 경우 패킷이 삭제되고 수신된 것으로 표시됩니다.
20. **dMLP에서 암호화를 지원합니까?**아니요.

21. **PFC 헤더 압축을 지원합니까?**아니요. 분산 경로는 아닙니다. 압축된 헤더 프레임 또는 패킷을 수신하는 경우 비분산 모드로 되돌아가므로 원거리 라우터는 PFC 헤더 압축을 구성하지 않는 것이 좋습니다. dMLP를 계속 실행하려면 양쪽 끝에서 PFC 헤더 압축을 비활성화해야 합니다.
22. **dMLP에서는 소프트웨어 압축이 지원됩니까?**아니요. 소프트웨어 압축은 분산 경로에서 작동하지 않습니다.
23. **전송 측에서 조각화가 지원됩니까?**바닐라 dMLP는 아니고요 Vanilla dMLP로 프래그먼트를 수신하는 데에는 문제가 없지만 전송 측면에서 프래그먼트화가 발생하지 않습니다. 전송 측면 조각화는 dMLP 인터페이스에서 **ppp multilink interleave**가 구성된 경우 지원됩니다.
24. **MLP 번들의 멤버 링크를 ping할 수 있습니까?**아니요. 멤버 링크에서 IP 주소를 구성할 수 없습니다.
25. **링크 MTU 및 MLP 프래그먼트 크기에 대한 종속성이 있습니까?**아니요. MTU 크기는 다른 프레임과 마찬가지로 MLP 프래그먼트가 직렬 링크의 MTU 크기를 초과할 수 없다는 명백한 제한 외에는 MLP 프래그먼트 크기와 아무런 관련이 없습니다.
26. **단일 라우터 쌍 간에 2개의 MLP 번들을 구성할 수 있습니까?**네, 가능합니다. 그러나 이로 인해 로드 밸런싱이 저하될 수 있습니다. 테스트 베드에서 두 개 이상의 라우터를 사용하여 시뮬레이션하는 데 유용할 수 있지만 실제 가치는 거의 없습니다.공통 피어로 이동하는 모든 링크는 동일한 번들에 배치되어야 합니다. 즉, 번들은 특정 피어로 이동하는 링크 집합입니다."피어"는 LCP 및 인증 단계에서 제공하는 사용자 이름 및 엔드포인트 판별자 값으로 식별됩니다. 두 라우터 간에 여러 번들을 생성하려는 경우 각 라우터를 해당 라우터에 대한 단일 피어보다 더 많은 피어로 마스커레이드(masquerade)하려고 하는 것입니다. 그들은 자신을 적절하게 식별해야 한다.
27. **멤버 링크에 다른 대기열 알고리즘이 있을 수 있습니까?**번들과 관련된 모든 대기열 메커니즘은 멤버 링크 수준이 아니라 번들 수준에서 적용해야 합니다. 그러나 큐 알고리즘을 구성하더라도 패킷이 번들에서 전환되는 방식에는 영향을 미치지 않습니다.
28. **Cisco 7500에서 dMLP가 활성화된 경우 멀티링크 번들의 멤버 링크에 대해 tx-quque-limit가 기본값으로 26으로 설정된 이유는 무엇입니까?**대역폭 T1/E1의 모든 Serial 인터페이스의 경우 tx-queue-limit은 약 4 또는 5입니다. T1s/E1을 멀티링크에서 함께 번들링할 경우 번들의 대역폭이 증가합니다. 전환은 MLP 인터페이스의 대역폭을 기반으로 이루어지므로 멤버 링크의 tx-queue-limit을 늘려야 합니다. 기본 링크라고 하는 멤버 링크 중 하나만 스위칭에 사용되므로 tx-queue-limit을 늘려야 합니다.또한 이 값은 테스트 후 이 값으로 튜닝한 후 선택한 경험적인 값입니다. 일반적으로 구축에는 번들에 4~6개의 T1/E1 링크가 있습니다. 값이 26이면 6~8개의 T1/E1 링크를 완벽하게 수용할 수 있으므로 이 값이 선택되었습니다.
29. **dMLP 구현에서 차등 지연과 그 가치는 무엇입니까?**dMLP는 30ms의 차등 지연을 지원합니다. 즉, 한 번에 T에서 프래그먼트를 수신하고 순서가 틀린 경우(시퀀스 번호 100이 필요하지만 101을 수신함) T+30ms까지 시퀀스 번호 100을 수신하지 않으면 100은 손실됨으로 선언되고 101에서 처리를 시작할 수 있는 경우 그렇게 합니다. 101로 시작할 수 없는 경우(중간 프래그먼트인 경우), 시작 프래그먼트(예: 104)가 있는 프래그먼트를 찾아 거기서 시작합니다.
30. **패킷이 7500에서 멀티링크가 있는 IP 레벨에서 프래그먼트화될 경우 어떻게 됩니까?**패킷이 IP 레벨에서 프래그먼트된 경우 중간 홉에서 리어셈블하지 않고 전송되지만 대상 라우터에서 리어셈블됩니다.
31. **패킷이 7500의 MLP 레벨에서 프래그먼트화될 경우 어떻게 됩니까?**패킷이 MLP 레벨에서 프래그먼트되고 리어셈블된 패킷이 MRRU보다 크면 멀티링크에서 패킷이 삭제됩니다. 전송 측 조각화는 dMLP에서만 dLFI에서 지원됩니다. packet_size가 frag_size보다 크고 MRRU보다 작은 경우에만 패킷이 MLP 레벨에서 프래그먼트됩니다. MRRU보다 많은 패킷이 전송되고 IP 레벨에서 프래그먼트되지 않은 경우, 패킷이 MRRU보다 많으므로 다른 끝은 MLP 레벨에서 프래그먼트되지 않은 모든 패킷 크기를 삭제합니다.

32. **MRRU는 어떻게 계산됩니까?**MRRU는 다음 환경 설정에 따라 계산됩니다.들어오는 새 멤버 링크의 경우 멤버 링크에 구성된 MRRU에 따라 LCP 레벨에서 MRRU가 다시 협상됩니다 .ppp multilink mrru **interface** 명령을 사용하여 링크 인터페이스에 구성된 값입니다.구성되지 않은 경우 상위 인터페이스에서 ppp multilink mrru 명령의 컨피그레이션에서 상속된 값입니다.두 값이 모두 있는 경우 링크 인터페이스 값이 우선합니다.기본 MRRU 1524입니다.

디버그 개선 사항

이러한 개선 사항은 향후 제공될 예정입니다. 계획이 아직 완료되지 않았습니다.

- LC**에서 debug frame-relay multilink** 명령을 활성화합니다.
- 인터페이스당 현재 디버그 CLI 및 지정된 패킷 수를 개선합니다.
- dDDR의 경우 QoS 기능이 아직 지원되지 않습니다. 이는 적절한 비즈니스 사례에서만 수행할 수 있습니다.

관련 정보

- 다이얼러 CEF
- 다이얼러 프로파일을 사용하여 피어 투 피어 DDR 구성
- MPLS—멀티링크 PPP 지원
- Cisco 7500 Series 라우터용 Distributed Multilink Point-to-Point Protocol
- 분산 멀티링크 프레임 릴레이(FRF.16)
- 임대 회선을 통한 분산 링크 조각화 및 인터리빙
- QoS(Quality of Service)를 통한 VoIP 링크(LLQ/IP RTP 우선순위, LFI, cRTP)
- 문제 해결 TechNotes - Cisco 7500 Series 라우터
- 라우터 제품 지원 페이지 - Cisco Systems
- 기술 지원 및 문서 – Cisco Systems