Oracle® Solaris 11.1 네트워크 성능 관리

부품 번호: E36608–02
2013년 9월

ORACLE

130924@25097

목차

머리말

Oracle Solaris 11.1 네트워크 성능 관리를 시작합니다. 이 설명서는 Oracle Solaris 네트워크 구성에 대한 기본적인 내용과 절차를 다루는 **Oracle Solaris 11.1 네트워크 설정** 시리즈의 일부입니다. 이 설명서에서는 Oracle Solaris를 이미 설치했다고 가정합니다. 네트워크를 구성할 준비가 되었거나 네트워크에 필요한 네트워킹 소프트웨어를 구성할 준비가 되어 있어야 합니다.

이 설명서의 대상

이 책은 네트워크에 구성된 Oracle Solaris 실행 시스템의 관리 책임자를 대상으로 작성되었습니다. 이 설명서를 사용하려면 2년 이상의 UNIX 시스템 관리 경력이 있어야 합니다. UNIX 시스템 관리 교육 과정에 참석하는 것도 도움이 될 수 있습니다.

Oracle Support에 액세스

Oracle 고객은 My Oracle Support를 통해 온라인 지원에 액세스할 수 있습니다. 자세한 내용은 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info를 참조하거나, 청각 장애가 있는 경우 http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs를 방문하십시오.

활자체 규약

다음 표는 이 설명서에서 사용되는 활자체 규약에 대해 설명합니다.

표 P–1 활자체 규약

활자체	설명	예
AaBbCc123	명령, 파일, 디렉토리 이름 및 컴퓨터 화면에 출력되는 내용입니다.	.login 파일을 편집하십시오. 모든 파일 목록을 보려면 ls -a 명령을 사용하십시오. machine_name% you have mail.

표 P–1 활자체 규약 *(계속)*

활자체	설명	예
AaBbCc123	사용자가 입력하는 내용으로 컴퓨터 화면의 출력 내용과 대조됩니다.	`machine_name%` **`su`** `Password:`
AaBbCc123	위치 표시자: 실제 이름이나 값으로 바뀝니다.	`rm` *filename* 명령을 사용하여 파일을 제거합니다.
AaBbCc123	설명서 제목, 새 용어, 강조 표시할 용어입니다.	**사용자 설명서**의 6장을 읽으십시오. **캐시**는 로컬로 저장된 복사본입니다. 파일을 저장하면 **안 됩니다**. **주:** 일부 강조된 항목은 온라인에서 굵은체로 나타납니다.

명령 예의 셸 프롬프트

다음 표에는 Oracle Solaris OS에 포함된 셸의 UNIX 시스템 프롬프트 및 수퍼유저 프롬프트가 나와 있습니다. 명령 예에서 셸 프롬프트는 명령을 일반 사용자가 실행해야 하는지 또는 권한 있는 사용자가 실행해야 하는지 나타냅니다.

표 P–2 셸 프롬프트

셸	프롬프트
Bash 셸, Korn 셸 및 Bourne 셸	`$`
수퍼유저용 Bash 셸, Korn 셸 및 Bourne 셸	`#`
C 셸	`machine_name%`
수퍼유저용 C 셸	`machine_name#`

◆ ◆ ◆ 1 장

1 네트워크 성능 관리 소개

이 장에는 네트워크 성능 관리 개요가 나와 있습니다. 이 장에서는 본 설명서의 나머지 부분에 설명되어 있는 기능도 소개합니다. 이러한 기능을 사용하면 네트워크 성능을 향상시킬 수 있습니다.

본 설명서에 있는 구성을 수행하기 전에 기본 네트워크 구성을 완료하여 시스템을 네트워크에 연결해야 합니다. 기본 네트워크 구성에 대한 내용은 **Oracle Solaris 11.1에서 반응적 네트워크 구성을 사용하여 시스템 연결** 및 **Oracle Solaris 11.1에서 고정된 네트워크 구성을 사용하여 시스템 연결**을 참조하십시오.

Oracle Solaris 11 시스템의 네트워킹에 대한 전반적인 개요는 **Oracle Solaris 11 네트워킹 소개**를 참조하십시오.

네트워크 성능을 향상시키는 기능

네트워크 인터페이스를 구성하면 시스템이 네트워크에 연결됩니다. 그러나 Oracle Solaris 11의 다른 기능을 사용하면 네트워크의 전체 성능이 향상됩니다. 네트워크 성능 관리는 이러한 기능과 기술을 사용하여 시스템의 네트워크 트래픽 처리 방식을 조정하는 것입니다. 이러한 기술을 사용하여 구성된 시스템은 보다 효율적으로 네트워크 트래픽을 관리할 수 있어 네트워크의 전체 성능을 향상시키는 데 도움을 줍니다. 이러한 기능은 네트워크 작업의 다양한 영역을 처리합니다. 그러나 다음과 같은 일반적인 이점을 제공할 수도 있습니다.

- 네트워크 연결성 – 링크 통합 및 IPMP 같은 기능은 네트워크에 대한 액세스가 계속 유지되는지 확인합니다.
- 효율성 – 로드 균형 조정 같은 일부 기능을 통해 사용 가능한 리소스에 네트워크 처리 부하를 분산시킬 수 있으므로 효율성이 향상됩니다.
- 네트워크 관리 – VLAN(가상 LAN) 같은 기능을 통해 관리가 원활해집니다. 네트워크를 향상시키는 다른 기능을 사용하면 네트워크 관리도 단순화됩니다.
- 비용 – 이러한 모든 Oracle Solaris 11 기술은 비싼 하드웨어를 추가 구매하지 않아도 네트워크 성능을 향상시킵니다.

본 설명서는 시스템의 네트워크 트래픽 호스트 및 처리 방식을 향상시키는 기능에 대해 설명합니다. 다음 예를 고려해 보십시오.

- 데이터 링크 및 인터페이스를 하나의 단위로 구성할 수 있습니다. 풀링된 리소스를 네트워크 처리 전용으로 사용할 수 있어 네트워크 처리량을 증가시킬 수 있습니다. 자세한 내용은 2 장, "링크 통합 사용" 또는 5 장, "IPMP 소개"를 참조하십시오.
- 근거리 통신망을 가상 부네트워크로 나누어 관리를 단순화할 수 있습니다. 자세한 내용은 3 장, "VLAN 작업"을 참조하십시오.
- 시스템에서 최단 연결 경로를 사용하여 대상으로 패킷을 전송하도록 할 수 있습니다. 자세한 내용은 4 장, "브리징된 네트워크 관리(작업)"를 참조하십시오.
- 다양한 시스템과 구성으로 이루어진 네트워크에서 네트워크의 모든 피어가 서로의 구성에 대해 정보를 제공받도록 설정할 수 있습니다. 따라서 두 피어 모두 지원하는 협상된 구성 세트를 기반으로 네트워크 패킷을 교환할 수 있습니다. 피어 협상은 자동입니다. 따라서 수동 구성이 필요하지 않습니다. 자세한 내용은 7 장, "LLDP를 사용하여 네트워크 연결 정보 교환"을 참조하십시오.

개별 환경에 맞게 본 설명서에 나오는 다른 기술을 사용합니다. 또한 일부 하드웨어 구성에서는 특정 유형의 기능을 사용해야 합니다. 예를 들어 동일한 서브넷에 속하도록 구성된 인터페이스가 시스템에 여러 개 있는 경우 IPMP(IP 다중 경로) 기술을 사용해야 합니다. 따라서 본 설명서에 있는 구성 절차를 모두 수행할 필요는 없습니다. 해당 네트워크 요구 사항에 맞는 기술을 선택하고 배포하십시오.

또한 네트워크를 향상시키는 데 도움이 되는 다른 네트워크 관련 시스템 구성은 Oracle Solaris 11.1 설명서 라이브러리를 참조하십시오. 예를 들어 가상 네트워크, 영역 및 리소스 관리를 사용하면 사용 가능한 네트워크 리소스의 사용을 최소화하면서 여러 일체형 네트워크 성능을 배포할 수 있습니다. 자세한 내용은 **Oracle Solaris 11.1에서 가상 네트워크 사용** 및 **Oracle Solaris 11.1 관리: Oracle Solaris 영역, Oracle Solaris 10 영역 및 리소스 관리**를 참조하십시오.

◆ ◆ ◆ 2 장

2 링크 통합 사용

링크 통합을 사용하여 단일 단위로 관리하는 여러 데이터 링크 리소스를 풀링할 수 있습니다. 여러 데이터 링크 리소스를 결합하고 이러한 리소스를 시스템의 네트워크 작업에 전용으로 사용하도록 하면 시스템의 성능이 크게 향상됩니다. 이 장에서는 링크 통합을 구성하고 유지 관리하는 데 사용하는 절차에 대해 설명합니다.

이 장에서는 다음 내용을 다룹니다.

링크 통합 개요

링크 통합은 **트렁킹**이라고도 하며 하나의 논리 단위로 구성되어 네트워크 트래픽 처리량을 증가시키는 시스템의 여러 인터페이스로 구성되어 있습니다. 다음 그림은 시스템에 구성된 링크 통합의 예를 보여줍니다.

그림 2–1 링크 통합 구성

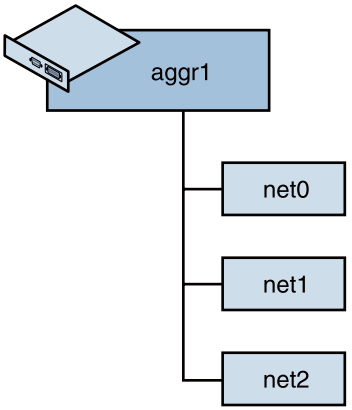


그림 2–1은 기본 데이터 링크 3개 net0 - net2로 구성된 통합 aggr1을 보여줍니다. 이러한 데이터 링크는 통합을 통해 시스템을 순회하는 트래픽에 전용으로 사용됩니다. 기본 링크는 외부 응용프로그램으로부터 숨겨져 있습니다. 대신 논리적 데이터 링크 aggr1에 액세스할 수 있습니다.

링크 통합에는 다음과 같은 기능이 있습니다.

- **대역폭 증가** - 여러 링크의 기능이 하나의 논리 링크로 결합됩니다.
- **자동 페일오버 및 페일백** – 링크 기반 실패 감지를 지원하여 실패 링크의 트래픽을 통합의 다른 작업 링크로 페일오버합니다.
- **관리 향상** – 모든 기본 링크가 단일 단위로 관리됩니다.
- **네트워크 주소 풀의 드레인 감소** - 전체 통합에 IP 주소 한 개를 지정할 수 있습니다.
- **링크 보호** – 통합을 통한 패킷 플로우에 대해 링크 보호를 사용으로 설정하는 데이터 링크 등록 정보를 구성할 수 있습니다.
- **리소스 관리** – 네트워크 리소스에 대한 데이터 링크 등록 정보와 플로우 정의를 사용하여 응용프로그램의 네트워크 리소스 사용을 규제할 수 있습니다. 리소스 관리에 대한 자세한 내용은 **Oracle Solaris 11.1에서 가상 네트워크 사용**의 3 장, “Oracle Solaris의 네트워크 리소스 관리”를 참조하십시오.

주 – 링크 통합은 IPMP(IP 다중 경로)와 유사한 기능을 수행하여 네트워크 성능 및 가용성을 향상시킵니다. 이러한 두 기술에 대한 비교는 부록 B, “링크 통합 및 IPMP: 기능 비교”를 참조하십시오.

Oracle Solaris는 다음 두 가지 유형의 링크 통합을 지원합니다.

- 트렁크 통합
- DLMP(데이터 링크 다중 경로) 통합

이러한 두 가지 유형의 링크 통합에 대한 차이점을 빠르게 보려면 부록 A, “링크 통합 유형: 기능 비교”를 참조하십시오.

다음 절에서는 각 유형의 링크 통합에 대해 자세히 설명합니다.

트렁크 통합

트렁크 통합은 트래픽 부하가 다른 다양한 네트워크에서 유용합니다. 예를 들어 네트워크의 시스템에서 많은 트래픽이 분산된 응용프로그램을 실행하는 경우 트렁크 통합을 해당 응용프로그램의 트래픽에서 전용으로 사용하도록 하여 대역폭 증가에 도움을 줄 수 있습니다. IP 주소 공간이 제한적임에도 불구하고 많은 대역폭이 필요한 사이트의 경우 대량의 인터페이스 통합에 대해 IP 주소가 하나만 필요합니다. 내부 인터페이스의 존재를 숨겨야 하는 사이트의 경우, 통합의 IP 주소가 외부 응용 프로그램으로부터 해당 인터페이스를 숨깁니다.

Oracle Solaris에서는 통합을 만드는 경우 기본적으로 트렁크 통합이 구성됩니다. 일반적으로 링크 통합과 함께 구성된 시스템은 외부 스위치를 사용하여 다른 시스템에 연결하기도 합니다. 다음 그림을 참조하십시오.

그림 2–2 스위치를 사용한 링크 통합

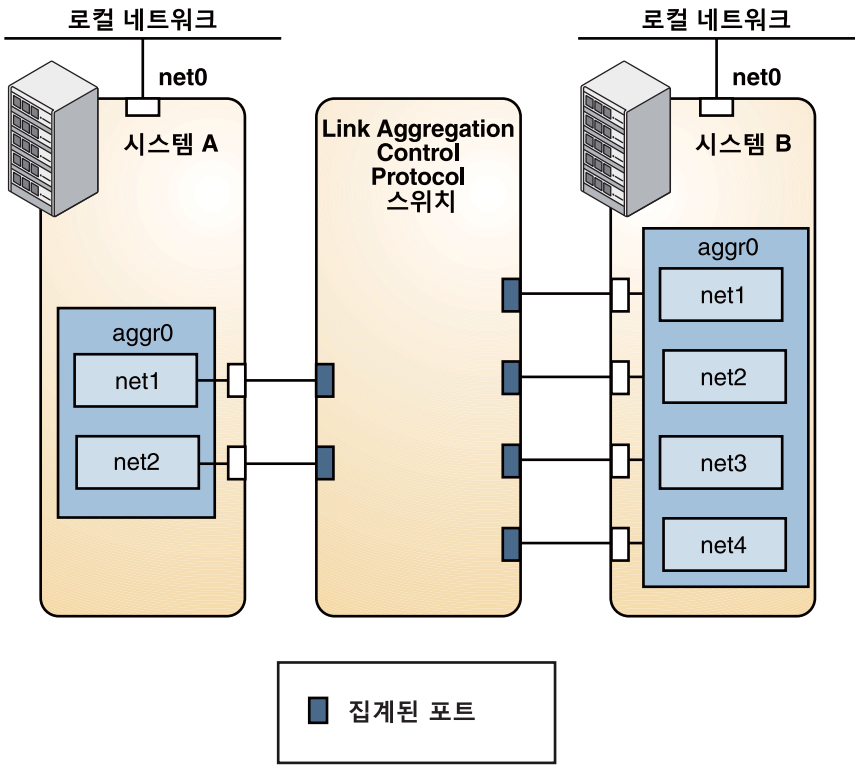


그림 2–2에서는 두 시스템이 포함된 로컬 네트워크를 보여줍니다. 각 시스템에는 하나의 통합이 구성되어 있습니다. 두 시스템은 LACP(링크 통합 제어 프로토콜)가 구성된 스위치에 의해 연결됩니다.

시스템 A에는 net1과 net2 두 인터페이스로 구성된 통합이 있습니다. 이러한 인터페이스는 통합된 포트를 통해 스위치에 연결됩니다. 시스템 B에는 net1에서 net4까지 인터페이스 4개로 구성된 통합이 있습니다. 이러한 인터페이스도 스위치의 통합된 포트에 연결됩니다.

이러한 링크 통합 토폴로지에서는 스위치가 통합 기술을 지원해야 합니다. 따라서 해당 스위치 포트가 시스템의 트래픽을 관리하도록 구성되어 있어야 합니다.

트렁크 통합도 인접(Back-to-Back) 구성을 지원합니다. 스위치를 사용하는 대신 다음 그림에 표시된 것처럼 두 시스템이 서로 직접 연결되어 병렬 통합을 실행합니다.

그림 2–3 인접(Back-to-Back) 링크 통합 구성

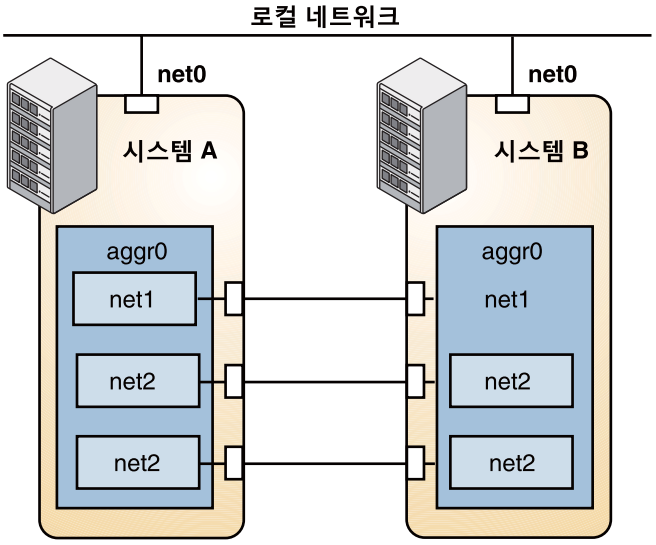


그림 2–3은 시스템 A의 링크 통합 `aggr0`을 보여 줍니다. 해당하는 각각의 기본 데이터 링크 간 연결을 통해 시스템 B의 링크 통합 `aggr0` 과 직접 연결되어 있습니다. 이 경우 시스템 A 및 B는 중복성과 고가용성을 제공하며 두 시스템 간 고속 통신도 제공합니다. 각 시스템에는 로컬 네트워크 내의 트래픽 플로우에 대해 `net0`도 구성되어 있습니다.

인접(Back-to-Back) 링크 통합에 사용되는 가장 일반적인 응용프로그램은 미러링된 데이터베이스 서버 구성입니다. 두 서버를 함께 업데이트해야 하므로 상당한 대역폭, 고속 트래픽 플로우 및 안정성이 필요합니다. 인접(Back-to-Back) 링크 통합의 가장 일반적인 사용은 데이터 센터에서 이루어집니다.

주 – DLMP 통합에서는 인접(Back-to-Back) 구성이 지원되지 않습니다.

다음 절에서는 트렁크 통합에 고유한 다른 기능에 대해 설명합니다. DLMP 통합 생성 시에는 이러한 기능을 구성하지 마십시오.

정책 및 로드 균형 조정

트렁크 통합을 사용하려는 경우 송신 트래픽에 대한 정책 정의를 고려해 보십시오. 이 정책은 사용 가능한 통합 링크에 패킷을 배포하여 로드 균형 조정을 설정하는 방법을 지정할 수 있습니다. 통합 정책에 가능한 계층 지정자 및 해당 중요성은 다음과 같습니다.

- *L2* - 각 패킷의 MAC(L2) 헤더를 해싱하여 송신 링크를 결정합니다.
- *L3* - 각 패킷의 IP(L3) 헤더를 해싱하여 송신 링크를 결정합니다.
- *L4* - 각 패킷의 TCP, UDP 또는 기타 ULP(L4) 헤더를 해싱하여 송신 링크를 결정합니다.

이러한 정책의 모든 조합도 유효합니다. 기본 정책은 L4입니다.

통합 LACP 모드 및 스위치

트렁크 통합 설정에 스위치가 포함되는 경우 스위치의 LACP 지원 여부에 유의해야 합니다. 스위치가 LACP를 지원하는 경우 스위치와 통합에 대해 LACP를 구성해야 합니다. 통합의 LACP는 다음 값 중 하나로 설정될 수 있습니다.

- `off` – 통합의 기본 모드입니다. *LACPDU*라고 하는 LACP 패킷이 생성되지 않습니다.
- `active` – 시스템에서 정기적으로 LACPDU를 생성하며, 생성 간격을 사용자가 지정할 수 있습니다.
- `passive` – 스위치로부터 LACPDU를 받는 경우에만 시스템에서 LACPDU를 생성합니다. 통합과 스위치가 모두 Passive 모드로 구성되어 있는 경우 LACPDU를 교환할 수 없습니다.

데이터 링크 다중 경로 통합

트렁크 통합은 일반적으로 네트워크 설정 요구 사항에 충분합니다. 그러나 트렁크 통합은 스위치를 하나만 사용하는 작업으로 제한됩니다. 따라서 스위치가 시스템 통합의 단일 실패 지점이 됩니다. 통합이 여러 스위치로 분산될 수 있도록 하는 이전 솔루션에는 고유의 단점이 있습니다.

- 스위치에 구현된 솔루션은 판매자별로 다르므로 표준화되어 있지 않습니다. 다른 판매자의 여러 스위치가 사용되는 경우 한 판매자의 솔루션이 다른 판매자의 제품에 적용되지 않을 수도 있습니다.
- 링크 통합을 IPMP(IP 다중 경로)와 결합하는 것은 매우 복잡합니다. 전역 영역과 비전역 영역이 관련되는 네트워크 가상화 컨텍스트에서는 더욱 복잡합니다. 많은 수의 시스템, 영역, NIC, VNIC(가상 NIC) 및 IPMP 그룹이 포함된 시나리오와 같이 구성 규모를 확장하는 경우 복잡성이 높아집니다. 이 솔루션은 시스템의 전역 영역 및 모든 비전역 영역 둘 다에서 구성을 수행해야 합니다.
- 링크 통합 및 IPMP 조합이 구현된 경우에도 이러한 구성에서 링크 보호, 사용자 정의 플로우, 대역폭 등의 링크 등록 정보 사용자 정의 기능 등 링크 계층에서 작업하는 장점을 누릴 수 없습니다.

DLMP 통합은 이러한 단점을 극복합니다. 다음 그림은 DLMP 통합 작동 방식을 보여줍니다.

그림 2–4 DLMP 통합

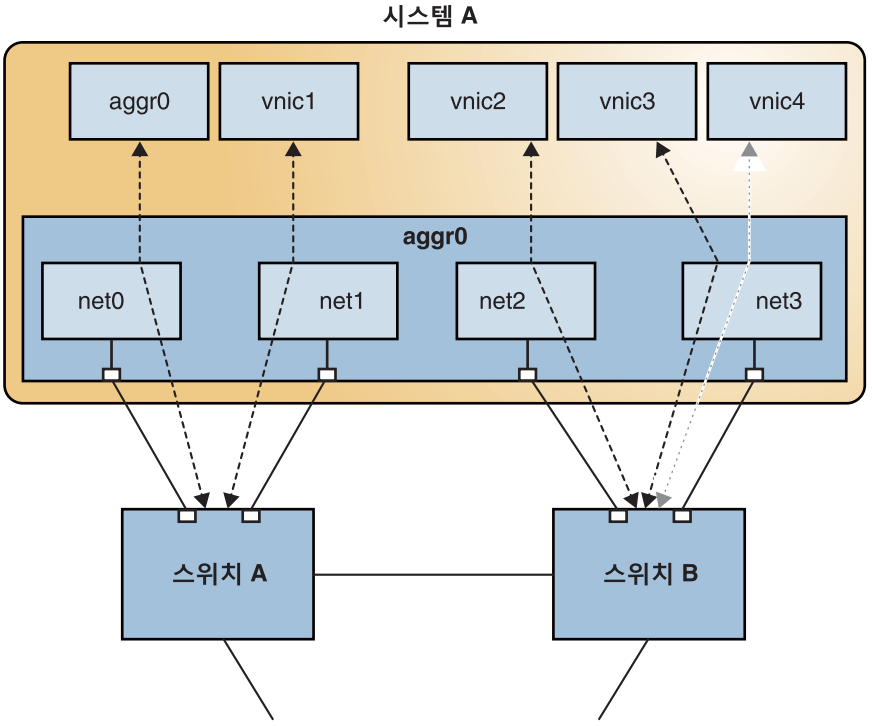


그림 2–4는 링크 통합 aggr0이 포함된 시스템 A를 나타냅니다. 통합은 기본 링크 net0 - net3까지 4개의 기본 링크로 구성되어 있습니다. aggr0 외에, 기본 인스턴트 VNIC도 통합에 대해 구성될 수 있습니다(vnic1 - vnic4). 통합은 스위치 A 및 스위치 B에 연결되어 있으며 이들 스위치는 더 큰 네트워크의 다른 대상 시스템에 연결됩니다.

트렁크 통합에서는 모든 포트가 통합을 통해 모든 구성된 데이터 링크와 연관됩니다. DLMP 통합에서는 포트가 통합의 구성된 데이터 링크와 연관되고 해당 통합을 통해 기본 인터페이스 및 VNIC에 연관됩니다.

VNIC 수가 기본 링크 수를 초과하는 경우 하나의 개별 포트가 여러 데이터 링크와 연관됩니다. 예를 들어, 그림 2–4는 vnic4가 vnic3과 포트를 공유함을 보여줍니다.

마찬가지로, 통합 포트가 실패하면 해당 포트를 사용하는 모든 데이터 링크가 다른 포트로 분배됩니다. 예를 들어 net0이 실패하면 aggr0이 다른 데이터 링크 중 하나와 포트를 공유합니다. 통합 포트에서 사용자에게 투명하고 통합에 연결된 외부 스위치에 독립적인 분배가 발생합니다.

스위치가 실패하면 통합은 다른 스위치를 사용하여 해당 데이터 링크에 연결성을 계속 제공합니다. 따라서 DLMP 통합에서는 여러 스위치를 사용할 수 있습니다.

요약하면 DLMP 통합은 다음 기능을 지원합니다.

- 통합은 여러 스위치로 분산될 수 있습니다.
- 스위치 구성이 필요하지 않으며 스위치에서 수행될 필요도 없습니다.
- 특정 유형에서 지원되는 옵션만 사용하는 경우 `dladm modify-aggr` 명령을 사용하여 트렁크 통합 및 DLMP 통합 사이에서 전환할 수 있습니다.

주 – 트렁크 통합에서 DLMP 통합으로 전환하는 경우 트렁크 통합에 대해 이전에 생성된 스위치 구성을 제거해야 합니다.

링크 통합의 요구 사항

링크 통합 구성은 다음 요구 사항에 따라 제한됩니다.

- 통합으로 구성될 데이터 링크에 대해 구성되어야 하는 IP 인터페이스는 없습니다.
- 통합의 모든 데이터 링크가 동일한 속도 및 전이중 모드로 실행되어야 합니다.
- DLMP 통합의 경우 통합을 다른 시스템의 포트에 연결하려면 스위치가 하나 이상 있어야 합니다. DLMP 통합 구성 시 인접(Back-to-Back) 설정을 사용할 수 없습니다.
- SPARC 기반 시스템에서는 각 데이터 링크에 고유 MAC 주소가 있어야 합니다. 수행 방법은 **Oracle Solaris 11.1에서 고정된 네트워크 구성을 사용하여 시스템 연결**의 "각 인터페이스의 MAC 주소가 고유한지 확인하는 방법"을 참조하십시오.

포트를 통합에 연결하거나 통합에서 분리하려면 IEEE 802.3ad 링크 통합 표준에 정의된 대로 장치에서 링크 상태 알림을 지원해야 합니다. 링크 상태 알림을 지원하지 않는 장치는 `dladm create-aggr` 명령의 `-f` 옵션을 사용해야만 통합할 수 있습니다. 이러한 장치의 경우 링크 상태가 항상 `UP`으로 보고됩니다. `-f` 옵션 사용에 대한 자세한 내용은 23 페이지 "링크 통합을 만드는 방법"을 참조하십시오.

링크 통합 관리

이 절에서는 링크 통합 구성 및 관리에 사용되는 다른 절차를 설명합니다. 이 절차의 일부 단계는 트렁크 통합 및 DLMP 통합 구성에서 공통으로 사용됩니다. 유형의 고유 단계가 표시되어 있습니다.

▼ 링크 통합을 만드는 방법

시작하기 전에

주 – 링크 통합은 동일한 속도로 작동하는 전이중 P2P 연결에서만 작동합니다. 통합의 인터페이스가 이 요구 사항을 준수하는지 확인하십시오.

통합 토폴로지에서 스위치를 사용하고 있으며 트렁크 통합을 생성 중인 경우 스위치에서 다음 작업을 수행해야 합니다.

- 통합으로 사용할 포트를 구성했습니다.
- 스위치가 LACP를 지원하는 경우 Active 모드나 Passive 모드로 LACP를 구성했습니다.

이러한 필수 항목은 통합 DLMP 통합에 적용되지 않습니다.

1 관리자로 전환합니다.

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 통합할 데이터 링크를 결정하려면 데이터 링크 정보를 표시합니다.

```
# dladm show-link
```

3 통합으로 구성할 데이터 링크가 응용프로그램에 의해 열려 있지 않은지 확인합니다.

예를 들어 IP 인터페이스가 데이터 링크를 통해 생성된 경우 IP 인터페이스를 먼저 제거합니다.

a. 링크가 응용프로그램에서 사용되고 있는지 확인하려면 `ipadm show-if` 명령의 출력 결과를 검사합니다.

```
# ipadm show-if
IFNAME       CLASS        STATE     ACTIVE     OVER
lo0          loopback     ok        yes        --
net0         ip           ok        no         --
```

출력에는 데이터 링크 `net0`을 통해 IP 인터페이스가 있음이 표시됩니다.

b. IP 인터페이스를 제거하려면 다음 명령을 입력합니다.

```
# ipadm delete-ip interface
```

여기서 *interface*는 링크를 통해 IP 인터페이스를 지정합니다.

4 다음 명령 중 하나를 실행하여 링크 통합을 만듭니다.

- 트렁크 통합을 만들려면 다음 명령을 실행합니다.

```
# dladm create-aggr [-f] [-P policy]  [-L lacpmode] \
 [-T time] [-u address] -l link1 -l link2 [...] aggr
```

`-f` 통합을 강제로 만듭니다. 링크 상태 알림을 지원하지 않는 장치를 통합하려는 경우 이 옵션을 사용합니다.

`-P` *policy* 통합에 대해 로드 균형 조정 정책을 지정합니다.

`-L` *lacpmode* LACP 모드가 사용되는 경우 해당 모드를 지정합니다. `off`, `active` 또는 `passive`가 될 수 있습니다. 20 페이지 “통합 LACP 모드 및 스위치”를 참조하십시오.

`-T` *time* LACP 시간을 지정합니다.

`-u` *address* 통합에 대해 고정된 유니캐스트 주소를 지정합니다.

`-l` *linkn* 통합하려는 데이터 링크를 지정합니다.

aggr 통합 이름을 지정합니다. 해당 이름은 사용자 정의될 수 있습니다. 이름 지정 규칙은 **Oracle Solaris 11 네트워킹 소개**의 “유효한 링크 이름 규칙”을 참조하십시오.

- DLMP 통합을 만들려면 다음 명령을 실행하십시오.

  # dladm create-aggr -m haonly -l *link1* **-l** *link2* **[...]** *aggr*

  `-l` *linkn* 통합하려는 데이터 링크를 지정합니다.

  aggr 통합의 이름을 지정합니다.

5 (옵션) 방금 만든 통합의 상태를 확인합니다.

```
# dladm show-aggr
```

통합의 상태는 `UP`이어야 합니다.

6 IP 인터페이스 만들기, VNIC 등과 같은 통합 추가 구성을 수행합니다.

예 2–1 트렁크 통합 만들기

이 예에서는 두 개의 기본 데이터 링크 net0 및 net1를 사용하여 링크 통합을 만드는 명령을 보여줍니다. 통합은 또한 LACP 패킷을 전송하도록 구성되어 있습니다. 이 예에서는 기본 데이터 링크를 통해 기존 IP 인터페이스를 제거하는 것으로 시작합니다.

```
# ipadm show-if
IFNAME       CLASS        STATE     ACTIVE     OVER
lo0          loopback     ok        yes        --
net0         ip           ok        no         --
net1         ip           ok        no         --

# ipadm delete-ip net0
# ipadm delete-ip net1
# dladm create-aggr -L active -l net0 -l net1 aggr0

# dladm show-aggr
LINK    MODE      POLICY   ADDRPOLICY     LACPACTIVITY   LACPTIMER
aggr0   standard  L4       auto           on             short
```

예 2–2 DLMP 통합 만들기

이 예에서는 DLMP 통합 생성 방법을 보여줍니다. 통합에는 기본 데이터 링크 3개가 있습니다.

```
# dladm create-aggr -m haonly -l net0 -l net1 -l net2 aggr0
# dladm show-link
LINK      CLASS     MTU     STATE   BRIDGE     OVER
net0      phys      1500    up      --         ----
net1      phys      1500    up      --         ----
net2      phys      1500    up      --         ----
aggr0     aggr      1500    up      --         net0, net1, net2

# dladm show-aggr
LINK    MODE       POLICY  ADDRPOLICY      LACPACTIVITY   LACPTIMER
aggr0   haonly     --      ----            ---            ----
```

▼ 링크 통합 유형 간 전환 방법

트렁크 통합과 DLMP 통합 사이에서 통합 유형을 전환하려면 `dladm modify-aggr` 명령을 사용하여 통합 모드를 수정합니다. 통합 유형을 전환하면 전체 구성이 변경됩니다. 따라서 이 절차는 다른 링크 통합 등록 정보를 간단히 수정하는 것보다 더 포괄적인 방식으로 통합에 영향을 줍니다.

1. **관리자로 전환합니다.**

   자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2. **링크 통합의 현재 유형을 확인합니다.**

   ```
   # dladm show-aggr
   ```

   출력 `MODE` 필드는 통합의 현재 유형을 나타냅니다. `MODE` 값은 트렁크 통합인 경우 **표준**이고 DLMP 통합인 경우 `haonly`입니다.

3. **통합은 DLMP 통합으로 전환합니다.**

   ```
   # dladm modify-aggr -m mode aggr
   ```

   여기서 *mode*는 트렁크 통합으로 전환하는 경우 `standard` DLMP 통합으로 전환하는 경우 `haonly`입니다.

4. **새로운 링크 통합 유형의 요구 사항에 맞게 스위치에 대한 조정을 수행합니다.**

5. **(옵션) 링크 통합 구성을 표시합니다.**

   ```
   # dladm show-aggr
   ```

예 2–3 트렁크 통합에서 DLMP 통합으로의 전환

이 예에서는 트렁크 통합에서 DLMP 통합으로의 통합 변경 방법을 보여줍니다.

```
# dladm show-aggr
LINK    MODE       POLICY  ADDRPOLICY      LACPACTIVITY   LACPTIMER
aggr0   standard   L2      auto            active         short

# dladm modify-aggr -m haonly aggr0
# dladm show-aggr
LINK    MODE       POLICY  ADDRPOLICY      LACPACTIVITY   LACPTIMER
aggr0   haonly     --      ----            -----          ----
```

다음으로 스위치측에서 이전에 트렁크 통합에 적용되었던 이전 스위치 구성이 제거됩니다.

▼ 트렁크 통합 수정 방법

이 절차에서는 트렁크 통합의 선택한 속성만 수정하는 방법을 보여줍니다. DLMP 통합에서는 이러한 속성이 지원되지 않습니다.

1 관리자로 전환합니다.

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 통합 정책을 수정합니다.

 # dladm modify-aggr -P *policy aggr*

policy 19 페이지 "정책 및 로드 균형 조정"에 설명된 대로 L2, L3 및 L4 정책 중 하나 이상을 나타냅니다.

aggr 정책을 수정하려는 통합을 지정합니다.

3 통합의 LACP 모드를 수정합니다.

 # dladm modify-aggr -L *lacpmode* **-T** *time aggr*

-L *lacpmode* 통합을 실행할 LACP 모드를 나타냅니다. 값은 `active`, `passive` 및 `off`입니다.

-T *time* LACP 타이머 값(`short` 또는 `long`)을 나타냅니다.

aggr 정책을 수정하려는 통합을 지정합니다.

예 2–4 트렁크 통합 수정

이 예에서는 링크 통합 `aggr0`의 정책을 L2로 수정한 다음 활성 LACP 모드를 설정하는 방법을 보여줍니다.

```
# dladm modify-aggr -P L2 aggr0
# dladm modify-aggr -L active -T short aggr0
# dladm show-aggr
LINK    MODE       POLICY  ADDRPOLICY      LACPACTIVITY   LACPTIMER
aggr0   standard   L2      auto            active         short
```

▼ 통합에 링크를 추가하는 방법

1 관리자로 전환합니다.

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 추가하려는 링크에 연결된 IP 인터페이스가 없는지 확인합니다.

```
# ipadm delete-ip interface
```

여기서 *interface*는 데이터 링크에 대해 구성된 IP 인터페이스입니다.

3 통합에 링크를 추가합니다.

```
# dladm add-aggr -l link [-l link] [...] aggr
```

여기서 *link*는 통합에 추가하는 데이터 링크를 나타내며 *aggr*은 통합의 이름입니다.

4 (트렁크 통합 전용) 링크 통합에 LACP가 구성되어 있지 않은 경우 추가 데이터 링크에 적합하게 스위치를 다시 구성합니다.

스위치에서 재구성 작업을 수행하려면 스위치 설명서를 참조하십시오.

예 2–5 통합에 링크 추가

이 예에서는 `aggr0` 통합에 링크를 추가하는 방법을 보여줍니다.

```
# dladm show-link
LINK     CLASS     MTU     STATE     BRODGE   OVER
net0     phys      1500    up        --       ----
net1     phys      1500    up        --       ----
aggr0    aggr      1500    up        --       net0, net1
net3     phys      1500    up        --       ----

# ipadm delete-ip net3
# dladm add-aggr -l net3 aggr0
# dladm show-link
LINK     CLASS     MTU     STATE     BRIDGE   OVER
net0     phys      1500    up        --       ----
net1     phys      1500    up        --       ----
aggr0    aggr      1500    up        --       net0, net1, net3
net3     phys      1500    up        --       ----
```

▼ 통합에서 링크를 제거하는 방법

1 **관리자로 전환합니다.**

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 **통합에서 링크를 제거합니다.**

```
# dladm remove-aggr -l link aggr
```

예 2–6 통합에서 링크 제거

이 예에서는 `aggr0` 통합에서 링크를 제거하는 방법을 보여줍니다.

```
dladm show-link
LINK     CLASS     MTU     STATE     OVER
net0     phys      1500    up        --       ----
net1     phys      1500    up        --       ----
aggr0    aggr      1500    up        --       net0, net1, net3
net3     phys      1500    up        --       ----

# dladm remove-aggr -l net3 aggr0
# dladm show-link
LINK     CLASS     MTU     STATE     BRIDGE   OVER
net0     phys      1500    up        --       ----
net1     phys      1500    up        --       ----
aggr0    aggr      1500    up        --       net0, net1
net3     phys      1500    unknown   --       ----
```

▼ 링크 통합 삭제 방법

1 **관리자로 전환합니다.**

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 **링크 통합에 구성된 IP 인터페이스를 삭제합니다.**

```
# ipadm delete-ip IP-aggr
```

여기서 *IP-aggr*은 링크 통합의 IP 인터페이스입니다.

3 **링크 통합을 삭제합니다.**

```
# dladm delete-aggr aggr
```

예 2–7 링크 통합 삭제

이 예에서는 `aggr0` 통합을 삭제하는 방법을 보여줍니다. 삭제는 지속적입니다.

```
# ipadm delete-ip aggr0
# dladm delete-aggr aggr0
```

◆ ◆ ◆ 3 장

3 VLAN 작업

가상 LAN을 사용하여 물리적 네트워크 환경을 추가할 필요 없이 네트워크를 부네트워크로 나눌 수 있습니다. 따라서 부네트워크는 가상이며 동일한 물리적 네트워크 리소스를 사용합니다. VLAN은 네트워크 관리를 용이하게 합니다. 그룹 크기가 작아지면 유지 관리하기가 더 쉽기 때문입니다. VLAN의 기능과 이점은 다음 절에서 더 자세히 설명합니다.

이 장에서는 VLAN(가상 LAN)을 구성하고 유지 관리하는 절차에 대해 설명합니다. 여기에서는 다음 내용을 다룹니다.

VLAN 배포: 개요

*VLAN(가상 LAN)*은 프로토콜 스택의 데이터 링크 계층에서 LAN(Local Area Network)이 세분화된 것입니다. 스위치 기술을 사용하는 LAN에 대해 VLAN을 만들 수 있습니다. VLAN에 사용자 그룹을 지정하면 전체 로컬 네트워크에 대한 네트워크 관리와 보안을 향상시킬 수 있습니다. 동일한 시스템의 인터페이스를 서로 다른 VLAN에 지정할 수도 있습니다.

다음 절에서는 VLAN에 대한 간단한 개요를 제공합니다.

VLAN 사용 시기

다음을 수행해야 하는 경우 로컬 네트워크를 VLAN으로 나누는 것을 고려해 보십시오.

- 작업 그룹의 논리적 분할을 만듭니다.

  예를 들어, 건물 한 층의 모든 호스트가 스위치 기반 로컬 네트워크 한 개에 연결되어 있다고 가정합니다. 한 층의 각 작업 그룹에 대해 별도의 VLAN을 만들 수 있습니다.

- 작업 그룹에 대해 서로 다른 보안 정책을 적용합니다.

  예를 들어 재무 부서와 정보 기술 부서의 보안 요구 사항은 상당히 다릅니다. 두 부서의 시스템이 동일한 로컬 네트워크를 공유하는 경우 각 부서에 대해 별도의 VLAN을 만들 수 있습니다. 그런 다음 VLAN별로 적절한 보안 정책을 적용할 수 있습니다.
- 작업 그룹을 관리 가능한 브로드캐스트 도메인으로 분할합니다.

  VLAN을 사용하면 브로드캐스트 도메인의 크기가 감소하며 네트워크 효율성이 향상됩니다.

VLAN 및 사용자 정의된 이름

VLAN은 일반 또는 사용자 정의된 이름 사용의 이점을 보여줍니다. 이전 릴리스에서는 VLAN이 데이터 링크의 하드웨어 기반 이름과 VLAN ID를 결합해야 하는 PPA(물리적 연결 지점)에 의해 식별되었습니다. Oracle Solaris 11에서는 VLAN을 식별할 보다 의미 있는 이름을 선택할 수 있습니다. 해당 이름은 **Oracle Solaris 11 네트워킹 소개**의 "유효한 링크 이름 규칙"에 있는 데이터 링크 이름 지정 규칙을 준수해야 합니다. 따라서 VLAN에 `sales0` 또는 `marketing1` 같은 이름이 지정될 수 있습니다.

VLAN 이름은 VLAN ID와 함께 작동합니다. 근거리 통신망의 각 VLAN은 VLAN 태그로도 알려져 있는 VLAN ID로 식별됩니다. VLAN ID는 VLAN 구성 중에 지정됩니다. VLAN 지원 스위치를 구성하는 경우 VLAN ID를 각 포트에 지정해야 합니다. 포트의 VLAN ID는 포트에 연결되는 인터페이스에 지정된 VLAN ID와 동일해야 합니다.

사용자 정의된 이름 및 VLAN ID 사용은 다음 절에 나와 있습니다.

VLAN 토폴로지

스위치 LAN 기술을 사용하면 로컬 네트워크의 시스템을 VLAN으로 구성할 수 있습니다. 로컬 네트워크를 VLAN으로 분할하려면 먼저 VLAN 기술을 지원하는 스위치를 구해야 합니다. VLAN 토폴로지 설계에 따라 스위치의 모든 포트가 단일 VLAN이나 여러 VLAN을 서비스하도록 구성할 수 있습니다. 각 스위치 제조업체에 따라 스위치의 포트 구성 절차가 달라집니다.

다음 그림은 3개의 VLAN으로 나누어진 근거리 통신망을 보여줍니다.

그림 3–1 VLAN 세 개가 있는 LAN(Local Area Network)

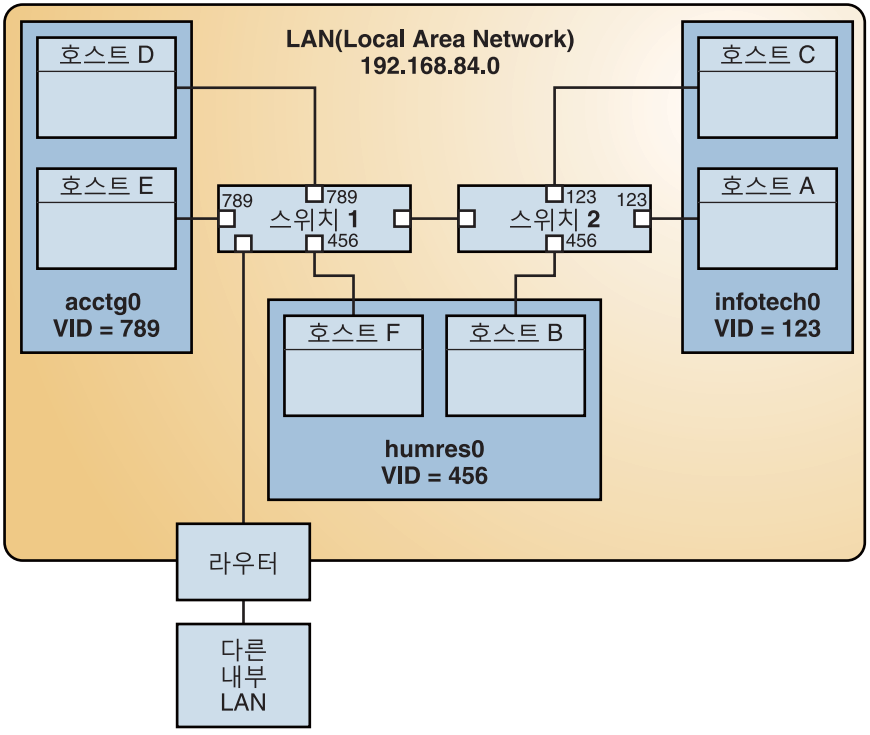


그림 3–1에서 LAN의 서브넷 주소는 `192.168.84.0`입니다. 이 LAN은 3개의 작업 그룹에 해당하는 3개의 VLAN으로 세분화되어 있습니다.

- `acctg0`, VLAN ID `789` – 회계 그룹. 이 그룹은 호스트 D 및 호스트 E를 소유합니다.
- `humres0`, VLAN ID `456` – 인사 그룹. 이 그룹은 호스트 B 및 호스트 F를 소유합니다.
- `infotech0`, VLAN ID `123` – 정보 기술 그룹. 이 그룹은 호스트 A 및 호스트 C를 소유합니다.

그림 3–1의 변형이 그림 3–2에 있습니다. 여기에서는 스위치가 하나만 사용되며 서로 다른 VLAN에 속하는 여러 호스트가 이 하나의 스위치에 연결됩니다.

그림 3–2 VLAN을 사용하는 네트워크의 스위치 구성

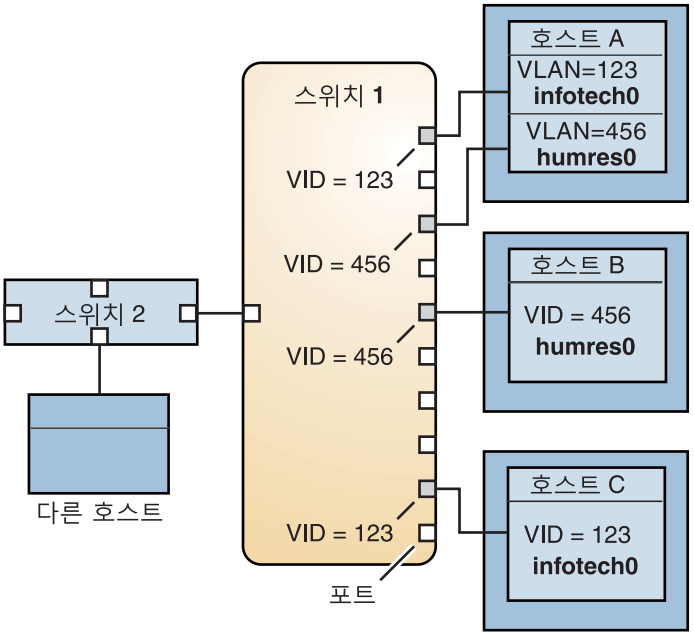


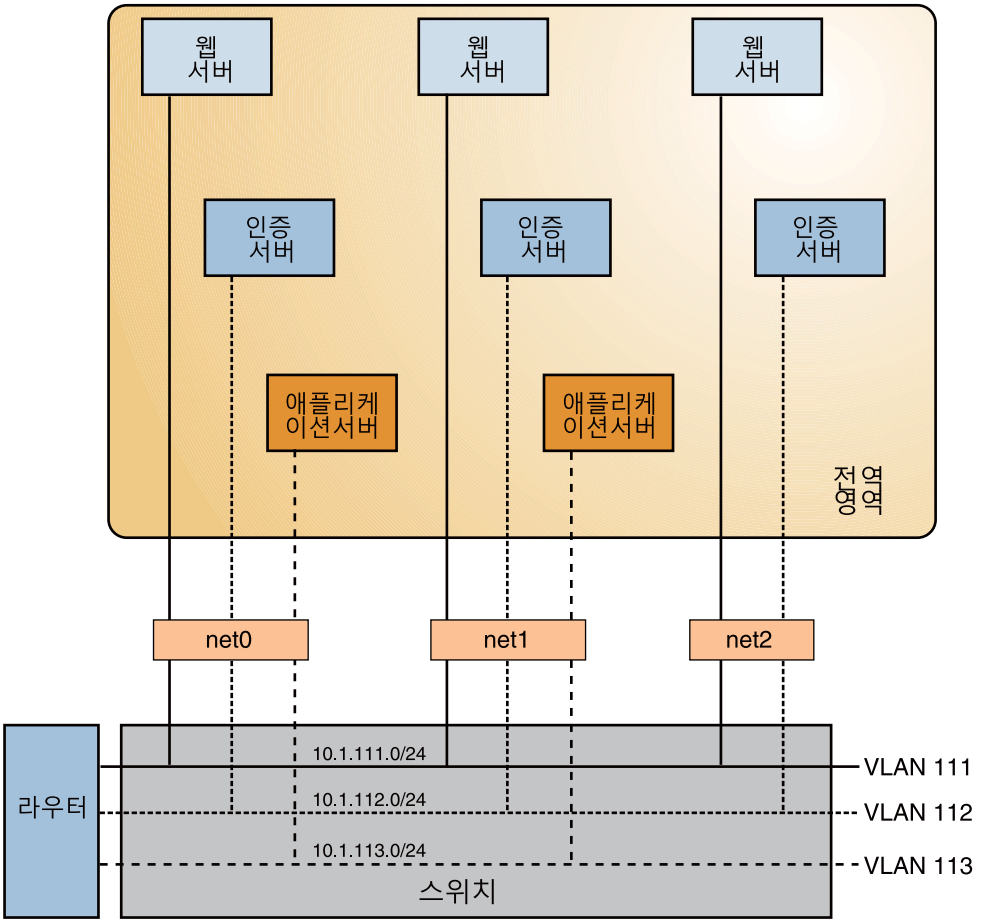
그림 3–2에서는 호스트 A 및 호스트 C가 VLAN ID가 123인 정보 기술 VLAN에 속합니다. 따라서 호스트 A의 인터페이스 중 하나가 VLAN ID 123으로 구성되어 있습니다. 이 인터페이스는 마찬가지로 VLAN ID 123으로 구성된 스위치 1의 포트 1에 연결됩니다. 호스트 B는 VLAN ID가 456인 인사 VLAN의 구성원입니다. 호스트 B의 인터페이스는 VLAN ID 456으로 구성된 스위치 1의 포트 5에 연결됩니다. 마지막으로 호스트 C의 인터페이스는 VLAN ID 123으로 구성되어 있습니다. 이 인터페이스는 스위치 1의 포트 9에 연결됩니다. 포트 9도 VLAN ID 123으로 구성되어 있습니다.

그림 3–2는 단일 호스트가 여러 VLAN에 속할 수 있다는 점도 보여줍니다. 예를 들어 호스트 A에는 호스트 인터페이스를 통해 2개의 VLAN이 구성되어 있습니다. 두번째 VLAN은 VLAN ID 456으로 구성되어 있으며 마찬가지로 VLAN ID 456으로 구성된 포트 3에 연결되어 있습니다. 따라서 호스트 A는 `infotech0` 및 `humres0` VLAN 모두의 구성원입니다.

VLAN 및 영역 사용

VLAN과 Oracle Solaris 영역을 결합하여 스위치 같은 단일 네트워크 장치 내에 여러 가상 네트워크를 구성할 수 있습니다. 3개의 물리적 네트워크 카드 `net0`, `net1` 및 `net2`가 사용된 시스템에 대한 아래 그림을 살펴보겠습니다.

그림 3–3 여러 VLAN이 있는 시스템



VLAN을 사용하지 않을 경우 특정 기능을 수행하는 여러 시스템을 구성하고 이러한 시스템을 별도의 네트워크에 연결할 것입니다. 예를 들어, 웹 서버는 하나의 LAN에 연결되고, 인증 서버는 다른 LAN에 연결되고, 애플리케이션 서버는 세번째 LAN에 연결됩니다. VLAN과 영역을 사용하면 모두 8개 시스템을 축소하여 단일 시스템의 영역으로 구성할 수 있습니다. 그런 다음 VLAN ID를 사용하여 동일한 기능을 수행하는 각 영역 세트에 VLAN을 지정합니다. 그림에 제공된 정보를 다음과 같이 표로 작성할 수 있습니다.

기능	영역 이름	VLAN 이름	VLAN ID	IP 주소	NIC
웹 서버	webzone1	web1	111	10.1.111.0	net0
인증 서버	authzone1	auth1	112	10.1.112.0	net0

기능	영역 이름	VLAN 이름	VLAN ID	IP 주소	NIC
애플리케이션 서버	appzone1	app1	113	10.1.113.0	net0
웹 서버	webzone2	web2	111	10.1.111.0	net1
인증 서버	authzone2	auth2	112	10.1.112.0	net1
애플리케이션 서버	appzone2	app2	113	10.1.113.0	net1
웹 서버	webzone3	web3	111	10.1.111.0	net2
인증 서버	authzone3	auth3	112	10.1.112.0	net2

그림에 표시된 구성을 만들려면 예 3–1을 참조하십시오.

VLAN 관리

이 절에서는 VLAN 구성 및 관리에 사용되는 절차를 다룹니다.

▼ VLAN 구성을 계획하는 방법

1 LAN 토폴로지를 검사하고 VLAN으로 세분화하기에 적합한 위치를 확인합니다.

이러한 토폴로지의 기본 예는 그림 3–1을 참조하십시오.

2 VLAN ID에 대한 번호 지정 체계를 만들고 각 VLAN에 VLAN ID를 지정합니다.

주 – VLAN 번호 지정 체계가 네트워크에 이미 있을 수도 있습니다. 이 경우 기존 VLAN 번호 지정 체계에 VLAN ID를 만들어야 합니다.

---

3 각 시스템에서 특정 VLAN의 구성 요소가 될 인터페이스를 결정합니다.

a. 시스템에 구성된 인터페이스를 확인합니다.

```
# dladm show-link
```

b. **시스템의 각 데이터 링크와 연결할 VLAN ID를 식별합니다.**

c. **VLAN을 만듭니다.**

4 **네트워크 스위치에 대한 인터페이스 연결을 확인합니다.**
각 인터페이스의 VLAN ID와 각 인터페이스가 연결된 스위치 포트를 확인합니다.

5 **연결된 인터페이스와 동일한 VLAN ID로 스위치의 각 포트를 구성합니다.**
구성 지침은 스위치 제조업체의 설명서를 참조하십시오.

▼ VLAN을 구성하는 방법

시작하기 전에 이 절차에서는 영역이 이미 시스템에 생성된 것으로 가정합니다. 영역을 만들고 영역에 인터페이스를 지정하는 단계는 이 절차에서 다루지 않습니다. 영역 구성에 대한 자세한 내용은 **Oracle Solaris 11.1 관리: Oracle Solaris 영역, Oracle Solaris 10 영역 및 리소스 관리**의 17 장, "비전역 영역 계획 및 구성(작업)"을 참조하십시오.

1 **관리자로 전환합니다.**
자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 **시스템에서 사용 중인 링크 유형을 확인합니다.**

```
# dladm show-link
```

3 **데이터 링크에 VLAN 링크를 만듭니다.**

```
# dladm create-vlan -l link -v vid vlan-link
```

link VLAN 인터페이스를 만들 링크를 지정합니다.

vid VLAN ID 번호를 나타냅니다.

vlan-link 관리상 선택한 이름일 수도 있는 VLAN의 이름을 지정합니다.

4 **VLAN 구성을 확인합니다.**

```
# dladm show-vlan
```

5 **VLAN에 IP 인터페이스를 만듭니다.**

```
# ipadm create-ip interface
```

여기서 *interface*는 VLAN 이름을 사용합니다.

6 **IP 주소로 IP 인터페이스를 구성합니다.**

```
# ipadm create-addr -a address interface
```

예 3–1 VLAN 구성

이 예에서는 그림 3–3에 설명된 VLAN 구성을 만드는 방법을 보여줍니다. 이 예는 시스템에 여러 영역을 이미 구성했다고 가정합니다. 영역 구성에 대한 자세한 내용은 **Oracle Solaris 11.1 관리: Oracle Solaris 영역, Oracle Solaris 10 영역 및 리소스 관리**의 제II부, "Oracle Solaris 영역"을 참조하십시오.

관리자는 VLAN을 구성하는 데 사용할 수 있는 링크를 확인하여 시작한 후 특정 링크를 통해 VLAN을 만듭니다.

```
global# dladm show-link
LINK      CLASS     MTU     STATE     BRIDGE     OVER
net0      phys      1500    up        --         --
net1      phys      1500    up        --         --
net2      phys      1500    up        --         --

global# dladm create-vlan -l net0 -v 111 web1
global# dladm create-vlan -l net0 -v 112 auth1
global# dladm create-vlan -l net0 -v 113 app1
global# dladm create-vlan -l net1 -v 111 web2
global# dladm create-vlan -l net1 -v 112 auth2
global# dladm create-vlan -l net1 -v 113 app2
global# dladm create-vlan -l net2 -v 111 web3
global# dladm create-vlan -l net2 -v 112 auth3

global# dladm show-vlan
LINK       VID     OVER        FLAGS
web1       111     net0        ----
auth1      112     net0        ----
app1       113     net0        ----
web2       111     net1        ----
auth2      112     net1        ----
app2       113     net1        ----
web3       111     net2        ----
auth3      113     net2        ----
```

링크 정보가 표시되면 VLAN이 목록에 포함됩니다.

```
global# dladm show-link
LINK       CLASS     MTU      STATE     BRIDGE     OVER
net0       phys      1500     up        --         --
net1       phys      1500     up        --         --
net2       phys      1500     up        --         --
web1       vlan      1500     up        --         net0
auth1      vlan      1500     up        --         net0
app1       vlan      1500     up        --         net0
web2       vlan      1500     up        --         net1
auth2      vlan      1500     up        --         net1
app2       vlan      1500     up        --         net1
web3       vlan      1500     up        --         net2
auth3      vlan      1500     up        --         net2
```

다음으로 관리자는 해당하는 각 영역에 VLAN을 지정합니다. VLAN을 지정하면 다음과 유사한 정보가 각 영역에 표시됩니다.

```
global# zonecfg -z webzone1 info net
net:
     address not specified
     physical: web1

global# zonecfg -z authzone1 info net
net:
     address not specified
     physical: auth1

global# zonecfg -z appzone2 info net
net:
     address not specified
     physical: app2
```

physical 등록 정보의 값은 지정된 영역에 대해 설정된 VLAN을 나타냅니다.

다음으로 관리자는 각 비전역 영역에 로그인하여 IP 주소로 VLAN을 구성합니다.

webzone1:

```
webzone1# ipadm create-ip web1
webzone1# ipadm create-addr -a 10.1.111.0/24 web1
ipadm: web1/v4
```

webzone2:

```
webzone2# ipadm create-ip web2
webzone2# ipadm create-addr -a 10.1.111.0/24 web2
ipadm: web2/v4
```

webzone3:

```
webzone3# ipadm create-ip web3
webzone3# ipadm create-addr -a 10.1.111.0/24 web3
ipadm: web3/v4
```

authzone1:

```
authzone1# ipadm create-ip auth1
authzone1# ipadm create-addr -a 10.1.112.0/24 auth1
ipadm: auth1/v4
```

authzone2:

```
authzone2# ipadm create-ip auth2
autzone2# ipadm create-addr -a 10.1.112.0/24 auth2
ipadm: auth2/v4
```

authzone3:

```
authzone3# ipadm create-ip auth3
authzone3# ipadm create-addr -a 10.1.112.0/24 auth3
ipadm: auth3/v4

appzone1:


appzone1# ipadm create-ip app1
appzone1# ipadm create-addr -a 10.1.113.0/24 app1
ipadm: app1/v4

appzone2:


appzone2# ipadm create-ip app2
appzone2# ipadm create-addr -a 10.1.113.0/24 app2
ipadm: app2/v4
```

VLAN이 모두 IP 주소로 구성되면 구성이 완료됩니다. 3개의 VLAN이 작동하고 있으며 해당 영역의 트래픽을 호스트할 수 있습니다.

▼ 링크 통합에 VLAN을 구성하는 방법

인터페이스에 VLAN을 구성하는 것과 동일한 방식으로 링크 통합에 VLAN을 만들 수도 있습니다. 링크 통합은 2 장, "링크 통합 사용"에 설명되어 있습니다. 이 절에서는 VLAN과 링크 통합의 구성을 결합합니다.

1 시스템에 구성된 링크 통합을 나열합니다.

```
# dladm show-link
```

2 선택한 링크 통합을 통해 만들려는 각 VLAN에 대해 다음 명령을 실행합니다.

```
# dladm create-vlan -l link -v vid vlan-link
```

link VLAN 인터페이스를 만들 링크를 지정합니다. 이 절차에서 링크는 링크 통합을 나타냅니다.

vid VLAN ID 번호를 나타냅니다.

vlan-link 관리상 선택한 이름일 수도 있는 VLAN의 이름을 지정합니다.

3 이전 단계에서 만든 모든 VLAN에 대해 VLAN을 통한 IP 인터페이스를 만듭니다.

```
# ipadm create-ip interface
```

여기서 *interface*는 VLAN 이름을 사용합니다.

4 VLAN의 각 IP 인터페이스에 대해 유효한 IP 주소를 구성합니다.

```
# ipadm create-addr -a address interface
```

예 3–2 링크 통합에 여러 VLAN 구성

이 예에서는 링크 통합에 VLAN 두 개가 구성됩니다. VLAN에는 각각 VLAN ID 193 및 194가 지정됩니다.

```
# dladm show-link
LINK          CLASS     MTU     STATE     BRIDGE     OVER
net0      phys      1500    up        --         ----
net1      phys      1500    up        --         ----
aggr0     aggr      1500    up        --         net0, net1

# dladm create-vlan -l aggr0 -v 193 acctg0
# dladm create-vlan -l aggr0 -v 194 humres0

# ipadm create-ip acctg0
# ipadm create-ip humres0

# ipadm create-addr -a 192.168.10.0/24 acctg0
ipadm: acctg0/v4
# ipadm create-addr -a 192.168.20.0/24 humres0
ipadm: humres0/v4
```

▼ 레거시 장치에서 VLAN을 구성하는 방법

특정 레거시 장치는 프레임 크기로도 알려져 있는 MTU(최대 전송 단위) 크기가 1514바이트인 패킷만 처리합니다. 프레임 크기가 최대 제한을 초과하는 패킷은 삭제됩니다. 이 경우 37 페이지 "VLAN을 구성하는 방법"에 나열된 것과 동일한 절차를 따릅니다. 하지만 VLAN을 만들 때는 `-f` 옵션을 사용하여 VLAN을 강제로 만듭니다.

1. **`-f` 옵션을 사용하여 VLAN을 만듭니다.**

   `# dladm create-vlan -f -l` *link* **`-v`** *vid vlan-link*

   link VLAN 인터페이스를 만들 링크를 지정합니다. 이 절차에서 링크는 레거시 장치를 나타냅니다.

   vid VLAN ID 번호를 나타냅니다.

   vlan-link 관리상 선택한 이름일 수도 있는 VLAN의 이름을 지정합니다.

2. **최대 전송 단위(MTU)에 대해 더 작은 크기를 설정합니다(예: 1496바이트).**

   `# dladm set-linkprop -p default_mtu=1496` *vlan-link*

   MTU\ 값이 작으면 전송 전에 VLAN 헤더를 삽입할 수 있는 공간이 링크 계층에서 허용됩니다.

3 **2단계와 동일한 단계를 수행하여 VLAN에 있는 각 노드의 <literal>MTU</literal> 크기에 대해 더 작은 값을 동일하게 설정합니다.**

링크 등록 정보 값 변경에 대한 자세한 내용은 **Oracle Solaris 11.1에서 고정된 네트워크 구성을 사용하여 시스템 연결**의 "기본 dladm 명령"을 참조하십시오.

VLAN 정보 표시

VLAN은 데이터 링크이므로 `dladm show-link` 명령을 사용하여 VLAN에 대한 정보를 볼 수 있습니다. 그러나 VLAN 특정 정보는`dladm show-vlan` 명령을 사용합니다.

다음 예에서는 이러한 명령을 사용하여 얻은 정보의 유형을 비교합니다. `dladm show-link` 명령이 사용된 첫번째 출력은 VLAN이 아닌 것을 포함하여 시스템의 모든 데이터 링크를 표시합니다. `dladm show-vlan` 명령이 사용된 두번째 출력은 VLAN에만 관련된 데이터 링크 정보의 하위 세트를 표시합니다.

```
# dladm show-link
LINK      CLASS     MTU       STATE     BRIDGE     OVER
net0      phys      1500      up        --         --
net1      phys      1500      up        --         --
net2      phys      1500      up        --         --
web1      vlan      1500      up        --         net0
auth1     vlan      1500      up        --         net0
app1      vlan      1500      up        --         net0
web2      vlan      1500      up        --         net1
auth2     vlan      1500      up        --         net1
app2      vlan      1500      up        --         net1
web3      vlan      1500      up        --         net2
auth3     vlan      1500      up        --         net2

# dladm show-vlan
LINK       VID     OVER      FLAGS
web1       111     net0      ----
auth1      112     net0      ----
app1       113     net0      ----
web2       111     net1      ----
auth2      112     net1      ----
app2       113     net1      ----
web3       111     net2      ----
auth3      113     net2      ----
```

VLAN 수정

`dladm modify-vlan` 명령을 사용하여 다음과 같은 방법으로 VLAN을 수정할 수 있습니다.

- VLAN의 VLAN ID 변경
- 다른 기본 링크로 VLAN 마이그레이션

VLAN의 VLAN ID를 변경하려면 다음 명령 중 하나를 사용합니다.

- `dladm modify-vlan -v` *vid* `- L` *datalink*

  이 명령에서 *vid*는 해당 VLAN에 지정할 새로운 VLAN ID를 지정합니다. *Datalink*는 VLAN이 구성된 기본 링크를 가리킵니다. 이 명령 구문은 해당 데이터 링크에 VLAN이 하나만 있는 경우에 사용할 수 있습니다. 구성된 VLAN이 여러 개인 데이터 링크에서 이 명령을 사용하면 데이터 링크의 VLAN에는 고유 VLAN ID가 있어야 하므로 명령이 실패합니다.

- `dladm modify-vlan -v` *vid vlan*

  하나의 데이터 링크에 VLAN이 여러 개인 경우 고유 VLAN ID를 변경하려면 이 명령을 사용합니다. 데이터 링크의 각 VLAN에는 고유 VLAN ID가 있습니다. 따라서 한 번에 하나씩 VLAN ID를 변경해야 합니다. 그림 3–3에서는 `net0`에 대해 구성된 VLAN ID `web1`, `auth1` 및 `app1`을 변경하려는 것으로 가정합니다. VLAN ID를 변경하려면 다음을 수행합니다.

  ```
  # dladm modify-vlan -v 123 web1
  # dladm modify-vlan -v 456 app1
  # dladm modify-vlan -v 789 auth1
  ```

VLAN을 삭제 및 재구성하지 않고 하나의 기본 데이터 링크에서 다른 기본 데이터 링크로 VLAN을 마이그레이션할 수 있습니다. 기본 링크는 물리적 링크일 수도 있고 링크 통합일 수도 있고 etherstub일 수도 있습니다. etherstubs에 대한 자세한 내용은 **Oracle Solaris 11.1에서 가상 네트워크 사용**의 "네트워크 가상화의 구성 요소"를 참조하십시오.

VLAN을 성공적으로 마이그레이션하기 위해서는 VLAN을 이동할 대상 기본 데이터 링크가 해당 VLAN의 데이터 링크 등록 정보를 수용할 수 있어야 합니다. 해당 등록 정보가 지원되지 않는 경우 마이그레이션이 실패하고 사용자에게 통지됩니다. 마이그레이션이 성공하면 해당 VLAN이 네트워크 연결을 유지하고 있는 경우 해당 VLAN을 사용하는 모든 응용프로그램이 계속 정상적으로 작동합니다.

VLAN 마이그레이션 후 특정 하드웨어 종속 등록 정보가 변경될 수도 있습니다. 예를 들어 VLAN은 항상 해당 기본 데이터 링크와 동일한 MAC 주소를 공유합니다. 따라서 VLAN을 마이그레이션하면 해당 VLAN의 MAC 주소가 대상 데이터 링크의 기본 MAC 주소로 변경됩니다. 영향을 받을 수 있는 다른 등록 정보로는 데이터 링크 상태, 링크 속도, MTU 크기 등이 있습니다. 그러나 응용프로그램은 중단 없이 계속 작동합니다.

주 – 마이그레이션된 VLAN에는 원래 데이터 링크의 하드웨어 레인 통계가 포함되어 있지 않습니다. 대상 데이터 링크에서 VLAN에 사용 가능한 하드웨어 레인이 통계 정보의 새로운 소스가 됩니다. 그러나 `dlstat` 명령에 따라 기본적으로 표시되는 소프트웨어 통계는 그대로 유지됩니다.

VLAN 마이그레이션은 전역으로 또는 선택적으로 수행할 수 있습니다. 전역 마이그레이션은 특정 데이터 링크의 모든 VLAN을 다른 데이터 링크로

마이그레이션하는 것을 의미합니다. 전역 마이그레이션을 수행하려면 소스 데이터 링크와 대상 데이터 링크만 지정하면 됩니다. 다음 예에서는 `ether0`의 모든 VLAN을 `net1`로 이동합니다.

```
# dladm modify-vlan -l net1 -L ether0
```

설명:

- `-L`은 VLAN이 구성된 원래 데이터 링크를 가리킵니다.
- `-l`는 VLAN이 마이그레이션되는 대상 데이터 링크를 가리킵니다.

주 – 소스 데이터 링크 앞에 대상 데이터 링크를 지정해야 합니다.

선택적 VLAN 마이그레이션을 수행하려면 이동하려는 VLAN을 지정합니다. 그림 3–3을 기반으로 하는 다음 예에서는 VLAN이 `net0`에서 `net3`으로 이동됩니다.

```
# dladm modify-vlan -l net3 web1,auth1,app1
```

주 – VLAN을 선택적으로 마이그레이션하는 경우 전역 마이그레이션에만 적용되는 `-L` 옵션은 생략합니다.

마이그레이션을 수행하는 동안 VLAN의 VLAN ID를 변경할 수 있습니다. 그림 3–3을 기초로 하여 다음 예에서는 동시에 여러 VLAN을 마이그레이션하고 VLAN ID를 변경하는 방법을 보여줍니다.

```
# dladm show-vlan
LINK    VID      OVER     FLAGS
web1    111      net0     -----
auth1   112      net0     -----
app1    113      net0     -----

# dladm modify vlan -l net3 -v 123 web1
# dladm modify vlan -l net3 -v 456 auth1
# dladm modify vlan -l net3 -v 789 app1
# dladm show-vlan
LINK    VID      OVER     FLAGS
web1    123      net3     -----
auth1   456      net3     -----
app1    789      net3     -----
```

주 – 병렬 하위 명령 `dladm modify-vnic`는 VLAN으로 구성된 VNIC를 마이그레이션합니다. VLAN을 마이그레이션하는지 또는 VLAN으로 구성된 VNIC를 마이그레이션하는지에 따라 올바른 하위 명령을 사용해야 합니다. `dladm show-vlan` 하위 명령으로 표시된 VLAN에서는 `modify-vlan` 하위 명령을 사용합니다. VLAN ID가 포함된 VNIC를 비롯하여 `dladm show-vnic` 하위 명령으로 표시된 VNIC에서는 `modify-vnic` 하위 명령을 사용합니다. VNIC를 수정하려면 **Oracle Solaris 11.1에서 가상 네트워크 사용**의 "네트워크 가상화의 구성 요소"를 참조하십시오.

VLAN 삭제

시스템의 VLAN 구성을 삭제하려면 `dladm delete-vlan` 명령을 사용합니다.

주 – VLAN을 삭제하려면 먼저 삭제하려는 VLAN의 기존 IP 구성을 삭제해야 합니다. IP 인터페이스가 VLAN에 있으면 VLAN 삭제가 실패합니다.

예 3–3 VLAN 구성 삭제

VLAN 구성을 삭제하려면 다음 예와 유사한 단계를 수행해야 합니다.

```
# dladm show-vlan
LINK      VID     OVER         FLAGS
web1      111     net0      ----
auth1     112     net0      ----
app1      113     net0      ----
web2      111     net1      ----
auth2     112     net1      ----
app2      113     net1      ----
web3      111     net2      ----
auth3     113     net2      ----

# ipadm delete-ip web1
# dladm delete-vlan web1
```

사용 사례: 링크 통합 및 VLAN 구성 결합

이 절의 예에서는 IP 인터페이스가 생성되는 링크 통합 및 VLAN을 사용하는 네트워크 구성 조합 생성 방법을 보여줍니다. 다른 네트워킹 시나리오를 제공하는 문서는 http://www.oracle.com/us/sun/index.htm에 있습니다.

다음 예에서는 NIC 4개를 사용하는 한 시스템이 개별 서브넷 8개의 라우터가 되도록 구성해야 합니다. 이를 위해서 서브넷마다 하나씩 8개의 링크가 구성됩니다. 먼저 NIC 4개에서 모두 링크 통합을 만듭니다. 이러한 태그 미지정된 링크는 기본 경로가 가리키는 네트워크에 대해 태그 미지정된 기본 서브넷이 됩니다.

그런 후에 다른 서브넷의 링크 통합에 VLAN 인터페이스가 구성됩니다. 색상으로 구분된 체계에 따라 서브넷의 이름이 지정됩니다. 마찬가지로 VLAN 이름은 해당 서브넷에 일치하도록 지정됩니다. 최종 구성은 8개 서브넷에 대한 8개 링크로 이루어지며, 한 개는 태그 미지정된 링크이고 7개는 태그 지정된 VLAN 링크입니다. 이 예는 데이터 링크에 IP 인터페이스가 있는지 확인하는 작업으로 시작됩니다. 데이터 링크를 통합으로 결합하려면 먼저 이러한 인터페이스를 삭제해야 합니다.

관리자는 데이터 링크에 구성된 IP 인터페이스를 모두 제거하는 것부터 시작합니다.

```
# ipadm show-if
IFNAME     CLASS      STATE   ACTIVE   OVER
lo0        loopback   ok      yes      --
net0       ip         ok      yes      --
net1       ip         ok      yes      --
net2       ip         ok      yes      --
net3       ip         ok      yes      --

# ipadm delete-ip net0
# ipadm delete-ip net1
# ipadm delete-ip net2
# ipadm delete-ip net3
```

그런 다음 관리자는 링크 통합 `default0`을 만듭니다.

```
# dladm create-aggr -P L2,L3 -l net0 -l net1 -l net2 -l net3 default0

# dladm show-link
LINK        CLASS      MTU  STATE    BRIDGE     OVER
net0        phys      1500  up       --         --
net1        phys      1500  up       --         --
net2        phys      1500  up       --         --
net3        phys      1500  up       --         --
default0    aggr      1500  up       --         net0 net1 net2 net3
```

다음으로 관리자는 `default0`에 대해 VLAN을 만듭니다.

```
# dladm create-vlan -v 2 -l default0 orange0
# dladm create-vlan -v 3 -l default0 green0
# dladm create-vlan -v 4 -l default0 blue0
# dladm create-vlan -v 5 -l default0 white0
# dladm create-vlan -v 6 -l default0 yellow0
# dladm create-vlan -v 7 -l default0 red0
# dladm create-vlan -v 8 -l default0 cyan0

# dladm show-link
LINK        CLASS      MTU  STATE    BRIDGE     OVER
net0        phys      1500  up       --         --
net1        phys      1500  up       --         --
net2        phys      1500  up       --         --
net3        phys      1500  up       --         --
default0    aggr      1500  up       --         net0 net1 net2 net3
orange0     vlan      1500  up       --         default0
green0      vlan      1500  up       --         default0
blue0       vlan      1500  up       --         default0
```

```
white0      vlan      1500  up       --         default0
yellow0     vlan      1500  up       --         default0
red0        vlan      1500  up       --         default0
cyan0       vlan      1500  up       --         default0

# dladm show-vlan
LINK          VID   OVER        FLAGS
orange0         2   default0    -----
green0          3   default0    -----
blue0           4   default0    -----
white0          5   default0    -----
yellow0         6   default0    -----
red0            7   default0    -----
cyan0           8   default0    -----
```

마지막으로 관리자는 VLAN 링크에 대해 IP 인터페이스를 만들고 해당 인터페이스에 IP 주소를 지정합니다.

```
# ipadm create-ip orange0
# ipadm create-ip green0
# ipadm create-ip blue0
# ipadm create-ip white0
# ipadm create-ip yellow0
# ipadm create-ip red0
# ipadm create-ip cyan0

# ipadm create-addr -a address orange0
# ipadm create-addr -a address green0
# ipadm create-addr -a address blue0
# ipadm create-addr -a address white0
# ipadm create-addr -a address yellow0
# ipadm create-addr -a address red0
# ipadm create-addr -a address cyan0
```

◆ ◆ ◆ 4 장

4 브리징된 네트워크 관리(작업)

이 장에서는 브리징된 네트워크 및 브리징된 네트워크 관리 방법에 대해 설명합니다. 다음 내용으로 구성되어 있습니다.

브리징 개요

브릿지는 두 노드 사이의 경로인 개별 네트워크 세그먼트를 연결하는 데 사용됩니다. 브릿지로 연결하면 연결된 네트워크 세그먼트가 단일 네트워크 세그먼트처럼 통신합니다. 브릿징은 네트워킹 스택의 데이터 링크 계층(L2)에 구현됩니다. 브릿지는 패킷 전달 방식을 사용하여 하위 네트워크를 함께 연결합니다.

브리징과 경로 지정은 모두 네트워크의 리소스 위치에 대한 정보를 배포하는 데 사용될 수 있지만 여러 가지 차이점이 있습니다. 경로 지정은 IP 계층(L3)에 구현되고 경로 지정 프로토콜을 사용합니다. 데이터 링크 계층에서는 경로 지정 프로토콜이 사용되지 않습니다. 대신 브릿지에 연결되어 있는 링크에 수신된 네트워크 트래픽을 검사하여 전달된 패킷의 대상을 확인합니다.

패킷이 수신되면 소스 주소가 검사됩니다. 패킷의 소스 주소는 패킷이 전송된 노드를 패킷이 수신된 링크에 연결합니다. 그런 다음 수신된 패킷이 대상 주소와 동일한 주소를 사용하는 경우 브릿지가 링크를 통해 패킷을 해당 주소로 전달합니다.

소스 주소와 연결된 링크는 브리징된 네트워크의 다른 브릿지에 연결된 중간 링크일 수 있습니다. 시간이 경과하면 브리징된 네트워크의 모든 브릿지가 패킷을 지정된 노드로 보내는 링크를 "학습"합니다. 따라서 패킷의 대상 주소를 사용하여 hop 단위 브릿징을 통해 패킷을 최종 대상으로 보냅니다.

로컬 "링크 작동 중지" 알림은 지정된 링크의 모든 노드에 더 이상 연결할 수 없음을 나타냅니다. 이 경우 해당 링크에 대한 패킷 전달이 중지되며 이 링크를 통한 모든 전달 항목이 비워집니다. 이전 전달 항목도 시간이 경과하면 비워집니다. 링크를 복원하면

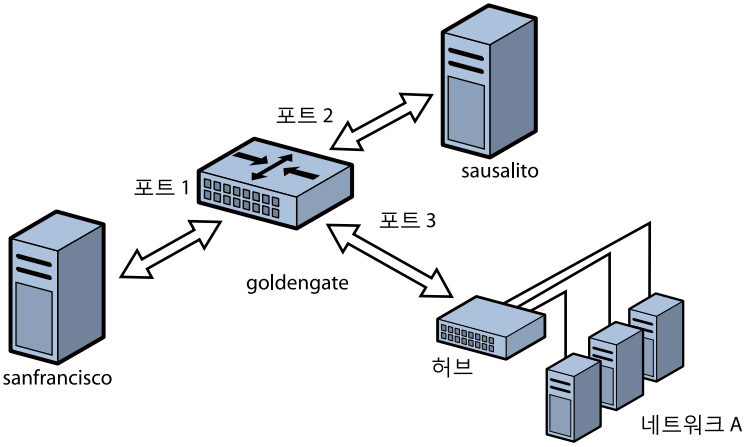
해당 링크를 통해 수신된 패킷이 새 항목으로 간주됩니다. 패킷의 소스 주소를 기반으로 하는 "학습" 프로세스가 다시 시작됩니다. 이 프로세스를 사용하면 주소가 대상 주소로 사용될 때 브릿지가 해당 링크를 통해 패킷을 제대로 전달할 수 있습니다.

패킷을 대상으로 전달하려면 브릿지가 해당 브릿지에 연결된 모든 링크에서 무차별 모드로 수신 대기해야 합니다. 무차별 모드에서 수신 대기하면 브릿지에서 패킷이 전체 회선 속도로 무기한 순환하는 전달 루프가 발생할 수 있습니다. 따라서 브릿징은 STP(Spanning Tree Protocol) 방식을 사용하여 하위 네트워크를 사용할 수 없게 만드는 네트워크 루프를 방지합니다.

브릿지에 STP 및 RSTP(고속 스패닝 트리 프로토콜)를 사용하는 것 외에도, Oracle Solaris에서는 향상된 TRILL(다수 링크의 투명한 상호 연결) 보호를 지원합니다. 기본적으로 STP가 사용되지만 브릿징 명령에 `-P trill` 옵션을 지정하면 TRILL을 사용할 수 있습니다.

브릿지 구성을 사용하면 다양한 노드가 단일 네트워크로 연결되어 네트워크의 다양한 노드 관리가 단순화됩니다. 브릿지를 통해 이러한 세그먼트를 연결하면 모든 노드가 단일 브로드캐스트 네트워크를 공유합니다. 따라서 각 노드는 네트워크 세그먼트에 트래픽을 전달하기 위해 라우터를 사용하는 대신 IP와 같은 네트워크 프로토콜을 사용하여 다른 노드에 연결할 수 있습니다. 브릿지를 사용하지 않는 경우 노드 간의 IP 트래픽 전달을 허용하도록 IP 경로 지정을 구성해야 합니다.

그림 4–1 브리징된 단순 네트워크

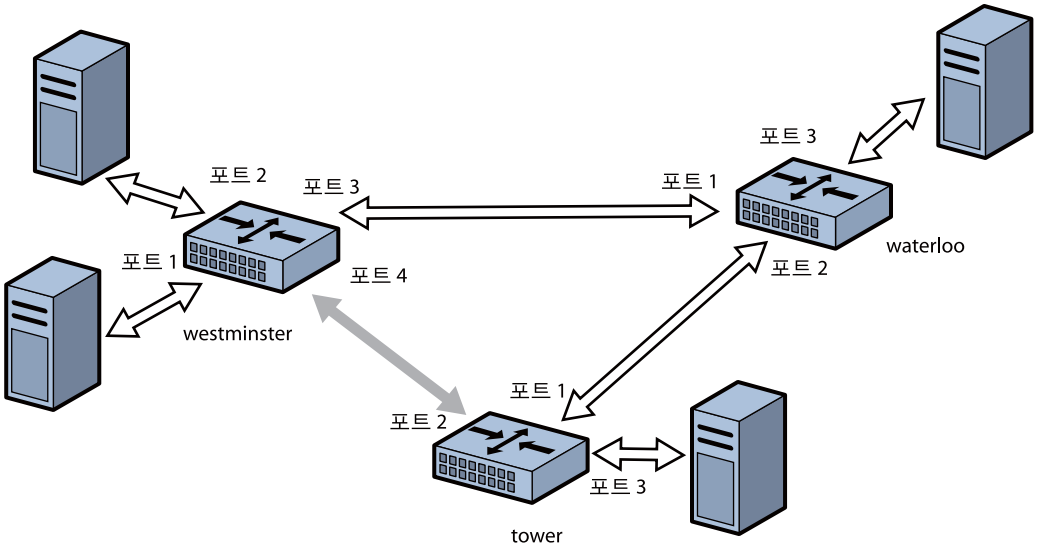


이전 그림에서는 브리징된 단순 네트워크 구성을 보여줍니다. `goldengate` 브릿지는 브리징이 구성되어 있는 Oracle Solaris 시스템입니다. 시스템 `sanfrancisco` 및 `sausalito`는 브릿지에 물리적으로 연결되어 있습니다. 네트워크 A에서 한쪽은 브릿지에, 다른 쪽은 컴퓨터 시스템에 물리적으로 연결된 허브를 사용합니다. 브릿지 포트는 `bge0`, `bge1` 및 `bge2`와 같은 링크입니다.

브리징된 네트워크는 여러 브릿지를 물리적으로 연결하는 링으로 형성될 수 있습니다. 이러한 구성은 네트워크에서 일반적으로 사용됩니다. 이 유형의 구성을 사용하면 오래된 패킷이 링을 무기한 반복하여 네트워크 링크가 포화되는 문제가 발생할 수 있습니다. 이러한 반복 상태로부터 보호하기 위해 Oracle Solaris 브릿지는 STP 및 TRILL 프로토콜을 모두 구현합니다. 대부분의 하드웨어 브릿지는 STP 루프 방지도 구현합니다.

다음 그림에서는 링으로 구성된 브릿징된 네트워크를 보여줍니다. 이 구성에서는 세 개의 브릿지를 보여줍니다. `westminster` 브릿지에는 시스템 2개가 물리적으로 연결되어 있습니다. `waterloo` 브릿지에는 시스템 1개가 물리적으로 연결되어 있습니다. `tower` 브릿지에도 시스템 1개가 물리적으로 연결되어 있습니다. 브릿지는 브릿지 포트를 통해 서로 물리적으로 연결되어 있습니다.

그림 4–2 브리징된 네트워크 링



STP 또는 RSTP를 루프 방지에 사용하는 경우 루프의 연결 중 하나가 패킷을 전달할 수 없게 하여 물리적 루프를 줄여줍니다. 그림 4–2에서는 `westminster` 및 `tower` 브릿지 간의 물리적 링크가 패킷 전달에 사용되지 않음을 보여줍니다.

사용 가능한 물리적 링크를 종료하여 루프 방지를 수행하므로 STP 및 RSTP는 대역폭이 감소합니다.

STP 및 RSTP와 달리 TRILL은 루프 방지를 위해 물리적 링크를 종료하지 않습니다. 대신 TRILL은 네트워크의 각 TRILL 노드에 대해 최단 경로 정보를 계산하고 이 정보를 사용하여 패킷을 개별 대상으로 전달합니다.

따라서 TRILL을 사용하면 시스템의 **모든** 링크를 항상 사용할 수 있습니다. 루프는 IP에서 루프를 처리하는 것과 유사한 방식으로 처리되므로 문제가 되지 않습니다. 즉, TRILL은 필요에 따라 경로를 만들고 전달 hop 제한을 사용하여 일시적 루프 상태에 의한 문제를 방지합니다.

주의 – SPARC 플랫폼에서 `local-mac-address?=false`를 설정하지 **마십시오**. 설정하는 경우 동일한 네트워크의 여러 포트에서 동일한 MAC 주소가 잘못 사용됩니다.

주의 – 가능한 최고 레벨의 성능이 필요한 경우 링크를 브릿지로 구성하지 **마십시오**. 브리징을 사용하려면 기본 인터페이스가 **반드시** 무차별 모드여야 합니다. 이 경우 하드웨어, 드라이버 및 기타 시스템 계층에 있는 많은 중요한 최적화 기능이 사용 안함으로 설정됩니다. 이러한 성능 향상 기능이 사용 안함으로 설정되는 것은 브릿징 방식의 필연적인 결과입니다.

시스템 링크 중 **일부**가 브릿징되지 않아 이러한 제약 조건이 적용되지 않는 시스템에서 브릿지를 사용할 수 있습니다. 이러한 성능 문제는 브릿지의 일부로 구성된 링크에만 영향을 줍니다.

STP에 대한 자세한 내용은 IEEE 802.1D-1998을 참조하십시오. RSTP에 대한 자세한 내용은 IEEE 820.1Q-2004를 참조하십시오. TRILL에 대한 자세한 내용은 Internet Engineering Task Force (IETF) TRILL draft documents (`http://tools.ietf.org/wg/trill`)를 참조하십시오.

링크 등록 정보

다음 링크 등록 정보는 `dladm show-linkprop` 명령을 사용하여 표시할 수 있으며 `dladm set-linkprop` 및 `reset-linkprop` 명령을 사용하여 수정할 수 있습니다.

`default_tag` 링크와 송수신하는 태그 미지정된 패킷의 기본 VLAN(가상 LAN) ID를 정의합니다. 유효한 값은 `0`에서 `4094` 사이입니다. 기본값은 `1`입니다. 비VLAN 및 비VNIC(가상 네트워크 인터페이스 카드) 유형 링크에만 이 등록 정보가 있습니다. 이 값을 `0`으로 설정하면 포트와 주고받는 태그 미지정된 패킷의 전달이 사용 안함으로 설정됩니다. 이것은 MAC 등록 정보입니다.

주 – 또한 브리징 범위 밖에서 이 등록 정보를 사용하여 링크의 IEEE PVID(Port VLAN Identifier)를 지정합니다. `default_tag`가 0이 아닌 경우 기본 링크 자체가 PVID를 자동으로 나타내기 때문에 링크에 동일한 ID를 가진 VLAN을 만들 수 없습니다.

예를 들어, `net0`에서 PVID를 `5`로 설정한 경우 `net0`에 ID가 `5`인 VLAN을 만들 수 없습니다. 이 경우 VLAN `5`를 지정하려면 `net1`을 사용합니다.

`default_tag`를 해당 링크에 생성된 기존 VLAN의 ID와 같도록 설정할 수 없습니다. 예를 들어 다음 명령은 `net0`에 VLAN 22를 만듭니다.

```
# dladm create-vlan -l net0 -v 22 myvlan0
```

이 경우 `default_tag`를 22로 설정할 수 없습니다. 설정하면 `net0`과 `myvlan0`이 모두 동일한 VLAN을 나타내게 됩니다.

`default_tag`를 `0`으로 설정하여 `net0`의 태그 미지정된 패킷이 VLAN과 연결되지 않도록 할 수 있습니다. 이 경우 구성된 브릿지가 해당 패킷을 전달할 수 없습니다.

`forward` 브릿지를 통한 트래픽 전달을 사용 및 사용 안함으로 설정합니다. 이 등록 정보는 VNIC 링크를 제외한 모든 링크에 있습니다. 유효한 값은 `1`(true) 및 `0`(false)입니다. 기본값은 `1`입니다. 트래픽 전달을 사용 안함으로 설정한 경우 링크 인스턴스와 연결된 VLAN이 브릿지를 통해 트래픽을 전달하지 않습니다. 전달을 사용 안함으로 설정하는 것은 일반 브릿지에 대한 "allowed set(허용 설정)"에서 VLAN을 제거하는 것과 같습니다. 즉, 로컬 클라이언트에서 기본 링크로의 VLAN 기반 I/O가 계속되지만 브릿지 기반 전달은 수행되지 않습니다.

`stp` STP와 RSTP를 사용 및 사용 안함으로 설정합니다. 유효한 값은 `1`(true) 및 `0`(false)입니다. 기본값은 `1`이며, STP와 RSTP가 사용으로 설정됩니다. 이 등록 정보를 `0`으로 설정하면 링크에 STP 또는 RSTP가 사용되지 않으며 링크가 항상 전달 모드로 설정됩니다. 전달 모드에서는 BPDU(Bridge Protocol Data Unit) 보호 기능을 사용합니다. 끝 노드에 연결된 P2P 연결을 구성하려면 STP 및 RSTP를 사용 안함으로 설정합니다. 비VLAN 및 비VNIC 유형 링크에만 이 등록 정보가 있습니다.

`stp_cost` 링크를 사용하는 경우 STP 및 RSTP 비용 값을 나타냅니다. 유효한 값은 `1`에서 `65535` 사이입니다. 기본값은 `0`으로, 링크 유형별로 비용이 자동으로 계산됨을 나타냅니다. 다음 값은 여러 링크 유형의 비용을 나타내며, 10메가비트/초의 경우 `100`, 100메가비트/초의 경우 `19`, 1기가비트/초의 경우 `4`, 10기가비트/초의 경우 `2`입니다.

`stp_edge` 포트가 다른 브릿지에 연결되는지 여부를 지정합니다. 유효한 값은 1(true) 및 0(false)입니다. 기본값은 1입니다. 이 등록 정보가 0으로 설정되면 어떠한 유형의 BPDU가 감지되지 않는 경우에도 포트가 다른 브릿지에 연결되는 것으로 데몬에서 가정합니다.

`stp_p2p` 연결 모드 유형을 지정합니다. 유효한 값은 `true`, `false` 및 `auto`입니다. 기본값은 `auto`이며, 지점 간 연결이 자동으로 검색됩니다. 지점간 모드로 강제 설정하려면 `true`를 지정합니다. 일반 다지점 모드로 강제 설정하려면 `false`를 지정합니다.

`stp_priority` STP 및 RSTP 포트 우선 순위 값을 설정합니다. 유효한 값은 0에서 255 사이입니다. 기본값은 128입니다. STP 및 RSTP 포트 우선 순위 값은 해당 값을 PVID로 미리 결정하여 브릿지의 기본 루트 포트를 결정하는 데 사용됩니다. 숫자 값이 작을수록 우선 순위가 더 높습니다.

STP 데몬

`dladm create-bridge` 명령으로 만든 각 브릿지는 동일한 이름을 가진 `svc:/network/bridge`의 SMF(서비스 관리 기능) 인스턴스로 표시됩니다. 각 인스턴스는 STP를 구현하는 `/usr/lib/bridged` 데몬의 복사본을 실행합니다.

예를 들어 다음 명령은 `pontevecchio`라는 브릿지를 만듭니다.

```
# dladm create-bridge pontevecchio
```

시스템은 SMF 서비스 인스턴스 `svc:/network/bridge:pontevecchio`와 관찰 노드 `/dev/net/pontevecchio0`을 만듭니다.

안전을 위해 모든 포트가 기본적으로 표준 STP를 실행합니다. STP와 같이 특정 형태의 브릿징 프로토콜을 실행하지 않는 브릿지는 네트워크에 오래 지속되는 전달 루프를 형성할 수 있습니다. 이더넷은 패킷에 홉 카운트 또는 TTL(트랜지스터-트랜지스터 논리)이 없기 때문에 이러한 루프가 네트워크에 치명적입니다.

특정 포트가 다른 브릿지에 연결되지 않은 경우(예: 호스트 시스템에 대한 직접 지점간 연결이 포트에 있는 경우) 관리상 해당 포트의 STP를 사용 안함으로 설정할 수 있습니다. 브릿지의 모든 포트에서 STP가 사용 안함으로 설정된 경우에도 STP 데몬이 계속 실행됩니다. 데몬은 다음과 같은 이유로 계속 실행됩니다.

- 추가된 새 포트를 처리하기 위해
- BPDU 보호 기능을 구현하기 위해
- 필요한 경우 포트에서 전달을 사용 또는 사용 안함으로 설정하기 위해

포트에서 STP가 사용 안함으로 설정된 경우 `bridged` 데몬이 BPDU를 계속 수신 대기합니다(BPDU 보호). 이 데몬은 `syslog`를 사용하여 오류에 플래그를 지정하고

포트에서 전달을 사용 안함으로 설정하여 잘못된 네트워크 구성을 나타냅니다. 링크가 작동 중지되었다가 다시 작동하거나 수동으로 링크를 제거했다가 다시 추가하면 링크가 다시 사용으로 설정됩니다.

브릿지에 대해 SMF 서비스 인스턴스를 사용 안함으로 설정하면 STP 데몬이 중지될 때 해당 포트에서 브릿지 전달이 중지됩니다. 인스턴스를 다시 시작하면 STP가 초기 상태부터 시작됩니다.

TRILL 데몬

`dladm create-bridge -P trill` 명령을 사용하여 만든 각 브릿지는 동일한 이름을 가진 `svc:/network/bridge` 및 `svc:/network/routing/trill`의 SMF 인스턴스로 표시됩니다. `svc:/network/routing/trill`의 각 인스턴스는 TRILL 프로토콜을 구현하는 `/usr/lib/trilld` 데몬의 복사본을 실행합니다.

예를 들어 다음 명령은 `bridgeofsighs`라는 브릿지를 만듭니다.

```
# dladm create-bridge -P trill bridgeofsighs
```

시스템은 `svc:/network/bridge:bridgeofsighs` 및 `svc:/network/routing/trill:bridgeofsighs`라는 두 개의 SMF 서비스를 만듭니다. 또한 시스템은 `/dev/net/bridgeofsighs0`이라는 관찰 노드를 만듭니다.

브릿지 디버깅

각 브릿지 인스턴스에 **관찰 노드**가 지정됩니다. 이 노드는 `/dev/net/` 디렉토리에 표시되며 브릿지 이름과 후행 `0`을 더한 값으로 이름이 지정됩니다.

관찰 노드는 `snoop` 및 `wireshark` 유틸리티에 사용됩니다. 이 노드는 자동으로 삭제되는 패킷 전송을 제외하고는 표준 이더넷 인터페이스처럼 작동합니다. 관찰 노드 위에 IP를 연결할 수 없으며, 수동 옵션을 사용하지 않으면 바인드 요청(`DL_BIND_REQ`)을 수행할 수 없습니다.

사용할 경우 관찰 노드는 브릿지에서 처리되는 각 패킷의 수정되지 않은 복사본 1개를 사용자가 모니터링 및 디버깅에 사용할 수 있게 합니다. 이 동작은 일반 브릿지의 모니터링 포트와 유사하며 일반 DLPI(데이터 링크 공급자 인터페이스) 무차별 모드 규칙이 동작에 적용됩니다. `pfmod` 명령 또는 `snoop` 및 `wireshark` 유틸리티의 기능을 사용하여 VLAN ID를 기준으로 패킷을 필터링할 수 있습니다.

전달된 패킷은 브릿지에 수신된 데이터를 나타냅니다.

주 – 브리징 프로세스가 VLAN 태그를 추가, 제거 또는 수정하는 경우 `dlstat` 명령으로 표시되는 데이터에 이 프로세스 발생 전 상태가 설명되어 있습니다. 드물긴 하지만 이러한 상황은 여러 링크에서 고유 `default_tag` 값이 사용되는 경우 혼동을 줄 수 있습니다.

브릿징 프로세스가 완료된 후 특정 링크에서 전송 및 수신되는 패킷을 보려면 브릿지 관찰 노드 대신 개별 링크에서 `snoop`를 실행합니다.

`dlstat` 명령을 사용하여 링크의 네트워크 패킷이 네트워크 리소스를 사용하는 방식에 대한 통계를 얻을 수도 있습니다. 자세한 내용은 **Oracle Solaris 11.1에서 가상 네트워크 사용**의 4 장, “Oracle Solaris에서 네트워크 트래픽 및 리소스 사용 모니터링”을 참조하십시오.

브릿지 사용 시의 링크 동작 변경 방식

다음 절에서는 네트워크 구성에 브릿지를 사용하는 경우 링크 동작이 어떻게 변경되는지에 대해 설명합니다.

표준 링크 동작에 대한 자세한 내용은 31 페이지 “VLAN 배포: 개요”를 참조하십시오.

DLPI 동작

다음은 브릿지를 사용으로 설정한 경우 링크 동작의 차이점에 대해 설명합니다.

- 링크 작동(`DL_NOTE_LINK_UP`) 및 링크 작동 중지(`DL_NOTE_LINK_DOWN`) 알림은 통합되어 전달됩니다. 즉, 모든 외부 링크가 링크 작동 중지 상태를 표시하는 경우 MAC 계층을 사용하는 상위 레벨 클라이언트도 링크 작동 중지 이벤트를 표시합니다. 브릿지의 외부 링크가 링크 작동 상태를 표시하는 경우 상위 레벨 클라이언트도 모두 링크 작동 이벤트를 표시합니다.

이 통합 링크 작동 및 링크 작동 중지 보고는 다음과 같은 이유로 수행됩니다.

- 링크 작동 중지 상태가 표시되는 경우 링크의 노드에 더 이상 연결할 수 없습니다. 브릿징 코드에서 다른 링크를 통해 패킷을 보내고 받을 수 있는 경우에는 해당하지 않습니다. 링크의 실제 상태가 필요한 관리 응용프로그램은 기존의 MAC 계층 커널 통계를 사용하여 상태를 표시할 수 있습니다. 이러한 응용프로그램은 하드웨어 상태 정보를 보고하며 전달 패킷에 참여하지 않는다는 점에서 IP 등의 일반 클라이언트와 다릅니다.
- 모든 외부 링크가 작동 중지된 경우 브릿지 자체가 종료된 것처럼 상태가 표시됩니다. 이러한 특수 사례에서는 연결할 수 있는 노드가 없다고 시스템에서 인식합니다. 단점은 모든 인터페이스가 "실제"(가상 아님)이고 모든 링크가 연결 해제된 경우 브릿지를 사용하여 로컬 전용 통신을 사용으로 설정할 수 없다는 것입니다.

- 링크 관련 기능은 모두 일반적으로 생성됩니다. 특수 하드웨어 가속 기능을 지원하는 링크는 클라이언트가 전적으로 실제 출력 링크를 결정하지 않으므로 이러한 기능을 사용할 수 없습니다. 브릿지 전달 기능이 대상 MAC 주소를 기준으로 출력 링크를 선택해야 하며, 이 출력 링크는 브릿지의 모든 링크일 수 있습니다.

브리징된 네트워크에서 VLAN 관리

기본적으로 시스템에 구성된 VLAN은 브릿지 인스턴스의 모든 포트에서 패킷을 전달합니다. `dladm create-vlan` 또는 `dladm create-vnic -v` 명령을 호출하면 기본 링크가 브릿지의 일부인 경우 이 명령이 해당 브릿지 링크에서 지정된 VLAN의 패킷 전달도 사용으로 설정합니다.

링크에서 VLAN을 구성하고 브릿지의 다른 링크와 패킷을 주고 받지 않도록 하려면 `dladm set-linkprop` 명령으로 VLAN의 `forward` 등록 정보를 설정하여 전달을 사용 안함으로 설정해야 합니다.

기본 링크가 브릿지의 일부로 구성된 경우 브릿징에 대해 VLAN을 자동으로 사용으로 설정하려면 `dladm create-vlan` 명령을 사용합니다.

표준 준수 STP에서는 VLAN이 무시됩니다. 브리징 프로토콜은 태그가 없는 BPDU 메시지를 사용하여 루프가 없는 토폴로지 한 개만 계산하고 이 트리 토폴로지를 사용하여 링크를 사용 및 사용 안함으로 설정합니다. 링크가 STP에 의해 자동으로 사용 안함으로 설정될 때 구성된 VLAN의 연결이 해제되지 않도록 네트워크에 프로비전된 중복 링크를 구성해야 합니다. 즉, 브리징된 백본 전체에서 모든 VLAN을 실행하거나 모든 중복 링크를 신중하게 검사해야 합니다.

TRILL 프로토콜은 복잡한 STP 규칙을 따르지 않습니다. 대신 TRILL은 VLAN 태그가 유지되는 패킷을 자동으로 캡슐화하고 네트워크를 통해 전달합니다. 즉, TRILL은 브리징된 단일 네트워크 내에서 동일한 VLAN ID가 재사용된, 격리된 VLAN을 바인딩합니다.

STP와 다른 중요한 차이점은 네트워크의 격리된 섹션에서 VLAN 태그를 재사용하여 4094 제한보다 큰 VLAN 세트를 관리할 수 있다는 것입니다. TRILL을 사용하여 이런 방식으로 네트워크를 관리할 수 없는 경우 공급자 기반 VLAN과 같은 다른 솔루션을 구현할 수 있습니다.

VLAN이 있는 STP 네트워크의 경우 STP가 "잘못된" 링크를 사용 안함으로 설정할 때 VLAN 분할을 방지하도록 페일오버 특성을 구성하기 어려울 수 있습니다. TRILL 모델의 이러한 강력한 장점은 격리된 VLAN의 비교적 적은 기능 손실이라는 장점보다 더 뛰어납니다.

VLAN 동작

브릿지는 허용되는 VLAN 세트와 각 링크의 `default_tag` 등록 정보를 검사하여 패킷 전달을 수행합니다. 일반 프로세스는 다음과 같습니다.

1. **입력 VLAN 결정.** 이 프로세스는 링크의 시스템에서 수신 패킷을 받으면 시작됩니다. 패킷이 수신되면 VLAN 태그가 확인됩니다. 해당 태그가 없거나 태그가 우선 순위 전용(0으로 설정됨)이면 해당 링크에 구성된 `default_tag` 등록 정보 값(0으로 설정되지 않은 경우)이 내부 VLAN 태그로 사용됩니다. 태그가 없거나 0이고 `default_tag`가 0이면 패킷이 무시됩니다. 태그 미지정된 전달은 수행되지 않습니다. 태그가 있고 `default_tag` 값과 동일한 경우 패킷도 무시됩니다. 그렇지 않은 경우 태그가 수신 VLAN 패킷으로 간주됩니다.
2. **링크 멤버쉽 확인.** 입력 VLAN이 이 링크에서 허용되는 VLAN으로 구성되지 않은 경우 패킷이 무시됩니다. 그런 다음 전달이 계산되고 동일한 검사가 출력 링크에 대해 수행됩니다.
3. **태그 업데이트.** VLAN 태그(이 때 0이 아님)가 출력 링크의 `default_tag` 값과 동일한 경우 우선 순위와 상관없이 패킷의 태그가(있는 경우) 제거됩니다. 출력 링크에서 VLAN 태그가 `default_tag` 값과 다른 경우 현재 태그가 없으면 추가되고, 현재 우선 순위를 패킷에 복사하여 출력 패킷에 대한 태그가 설정됩니다.

주 – 전달 시 패킷을 여러 인터페이스로 보내는 경우(브로드캐스트, 멀티캐스트 및 알 수 없는 대상) 출력 링크 검사와 태그 업데이트를 각 출력 링크에 대해 독립적으로 수행해야 합니다. 일부 전송에는 태그가 지정되고 다른 전송에는 태그가 지정되지 않을 수 있습니다.

브릿지 구성 보기

다음 예에서는 브릿지 구성 및 브릿징 서비스에 대한 정보를 확인하는 방법을 보여줍니다.

- 다음 명령을 실행하여 브릿지에 대한 정보를 볼 수 있습니다.

```
# dladm show-bridge
BRIDGE      PROTECT ADDRESS            PRIORITY DESROOT
tonowhere   trill   32768/66:ca:b0:39:31:5d 32768 32768/66:ca:b0:39:31:5d
sanluisrey  stp     32768/ee:2:63:ed:41:94 32768 32768/ee:2:63:ed:41:94
pontoon     trill   32768/56:db:46:be:b9:62 32768 32768/56:db:46:be:b9:62
```

- 다음 명령을 실행하여 브릿지에 대한 TRILL 별명 정보를 볼 수 있습니다.

```
# dladm show-bridge -t tonowhere
NICK FLAGS LINK        NEXTHOP
38628 --   simblue2    56:db:46:be:b9:62
58753 L    --          --
```

브릿지 관리

Oracle Solaris에서는 `dladm` 명령과 SMF 기능을 사용하여 브릿지를 관리합니다. SMF 명령을 사용하면 `svc:/network/bridge` 인스턴스의 FMRI(결함 관리 리소스 식별자)를 통해 브릿지 인스턴스를 사용 및 사용 안함으로 설정하고 모니터할 수 있습니다. `dladm` 명령을 사용하면 브릿지를 만들거나 삭제하고 브릿지에 링크를 지정하거나 제거할 수 있습니다.

브릿지 관리(작업 맵)

다음 표에서는 브릿지 관리에 사용할 수 있는 작업을 보여줍니다.

작업	설명	수행 방법
구성된 브릿지에 대한 정보를 확인합니다.	`dladm show-bridge` 명령을 사용하여 시스템에 구성된 브릿지에 대한 정보를 확인합니다. 구성된 브릿지, 링크, 통계 및 커널 전달 항목에 대한 정보를 확인할 수 있습니다.	60 페이지 “구성된 브릿지에 대한 정보를 확인하는 방법”
브릿지에 연결된 링크에 대한 구성 정보를 확인합니다.	`dladm show-link` 명령을 사용하여 시스템에 구성된 링크에 대한 정보를 확인합니다. 링크가 브릿지에 연결된 경우 `BRIDGE` 필드의 출력을 참조하십시오.	62 페이지 “브릿지 링크에 대한 구성 정보를 확인하는 방법”

작업	설명	수행 방법
브릿지를 만듭니다.	`dladm create-bridge` 명령을 사용하여 브릿지를 만들고 선택적 링크를 추가합니다. 기본적으로 브릿지는 STP를 사용하여 생성됩니다. 대신 TRILL을 사용하여 브릿지를 만들려면 `dladm create-bridge` 명령줄에 `-P trill`을 추가합니다. 또는 `dladm modify-bridge` 명령을 사용하여 TRILL을 사용으로 설정합니다.	62 페이지 "브릿지를 만드는 방법"
브릿지에 대한 보호 유형을 수정합니다.	`dladm modify-bridge` 명령을 사용하여 브릿지에 대한 보호 유형을 수정합니다.	63 페이지 "브릿지에 대한 보호 유형을 수정하는 방법"
브릿지에 링크를 추가합니다.	`dladm add-bridge` 명령을 사용하여 기존 브릿지에 링크를 하나 이상 추가합니다.	63 페이지 "기존 브릿지에 링크를 하나 이상 추가하는 방법"
브릿지에서 링크를 제거합니다.	`dladm remove-bridge` 명령을 사용하여 브릿지에서 링크를 제거합니다. 모든 링크가 제거될 때까지 브릿지를 삭제할 수 없습니다.	64 페이지 "브릿지에서 링크를 제거하는 방법"
시스템에서 브릿지를 삭제합니다.	`dladm delete-bridge` 명령을 사용하여 시스템에서 브릿지를 삭제합니다.	65 페이지 "시스템에서 브릿지를 삭제하는 방법"

▼ 구성된 브릿지에 대한 정보를 확인하는 방법

이 절차에서는 `dladm show-bridge` 명령에 다양한 옵션을 사용하여 구성된 브릿지에 대한 여러 종류의 정보를 표시하는 방법을 설명합니다.

`dladm show-bridge` 명령 옵션에 대한 자세한 내용은 `dladm(1M)` 매뉴얼 페이지를 참조하십시오.

1 관리자로 전환합니다.

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 특정 브릿지나 구성된 모든 브릿지에 대한 정보를 확인합니다.

- 브릿지 목록을 확인합니다.

```
# dladm show-bridge
```

- 브릿지의 링크 관련 상태를 표시합니다.

  # dladm show-bridge -l *bridge-name*
- 브릿지에 대한 통계를 표시합니다.

  # dladm show-bridge -s *bridge-name*

  주 – 보고된 브릿지 통계의 이름과 정의는 변경될 수 있습니다.

- 브릿지에 대한 링크 관련 통계를 표시합니다.

  # dladm show-bridge -ls *bridge-name*
- 브릿지에 대한 커널 전달 항목을 표시합니다.

  # dladm show-bridge -f *bridge-name*
- 브릿지에 대한 TRILL 정보를 표시합니다.

  # dladm show-bridge -t *bridge-name*

예 4–1 브릿지 정보 보기

다음은 dladm show-bridge 명령에 다양한 옵션을 사용하는 예입니다.

- 다음 명령은 시스템에 구성된 모든 브릿지에 대한 정보를 보여줍니다.

```
# dladm show-bridge
BRIDGE       PROTECT ADDRESS           PRIORITY DESROOT
goldengate   stp     32768/8:0:20:bf:f 32768    8192/0:d0:0:76:14:38
baybridge    stp     32768/8:0:20:e5:8 32768    8192/0:d0:0:76:14:38
```

- 다음 명령은 단일 브릿지 인스턴스 tower에 대한 링크 관련 상태 정보를 보여줍니다. 구성된 등록 정보를 보려면 dladm show-linkprop 명령을 사용합니다.

```
# dladm show-bridge -l tower
LINK         STATE        UPTIME   DESROOT
net0         forwarding   117      8192/0:d0:0:76:14:38
net1         forwarding   117      8192/0:d0:0:76:14:38
```

- 다음 명령은 지정한 브릿지 terabithia에 대한 통계를 보여줍니다.

```
# dladm show-bridge -s terabithia
BRIDGE       DROPS        FORWARDS
terabithia   0            302
```

- 다음 명령은 지정한 브릿지 london의 모든 링크에 대한 통계를 보여줍니다.

```
# dladm show-bridge -ls london
LINK         DROPS     RECV      XMIT
net0         0         360832    31797
net1         0         322311    356852
```

- 다음 명령은 지정한 브릿지 avignon에 대한 커널 전달 항목을 보여줍니다.

```
# dladm show-bridge -f avignon
DEST              AGE     FLAGS  OUTPUT
8:0:20:bc:a7:dc   10.860  --     net0
```

```
8:0:20:bf:f9:69   --      L      net0
8:0:20:c0:20:26   17.420  --     net0
8:0:20:e5:86:11   --      L      net1
```

- 다음 명령은 지정한 브릿지 key에 대한 TRILL 정보를 보여줍니다.

```
# dladm show-bridge -t key
NICK FLAGS LINK        NEXTHOP
38628 --   london      56:db:46:be:b9:62
58753 L    --          --
```

▼ 브릿지 링크에 대한 구성 정보를 확인하는 방법

dladm show-link 출력에는 BRIDGE 필드가 포함됩니다. 링크가 브릿지의 구성원이면 이 필드에 구성원으로 속한 브릿지의 이름이 식별됩니다. 이 필드는 기본적으로 표시됩니다. 브릿지에 속하지 않는 링크의 경우 -p 옵션을 사용하면 필드가 비어 있습니다. 그렇지 않으면 필드에 --가 표시됩니다.

또한 브릿지 관찰 노드는 dladm show-link 출력 결과에 별도의 링크로 표시됩니다. 이 노드의 경우 기존 OVER 필드에 브릿지의 구성원인 링크가 나열됩니다.

1 관리자로 전환합니다.

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 브릿지의 구성원인 링크에 대한 구성 정보를 확인합니다.

```
# dladm show-link [-p]
```

-p 옵션은 구문 분석 가능한 형식으로 출력을 생성합니다.

▼ 브릿지를 만드는 방법

이 절차에서는 기본 프로토콜인 STP를 사용하여 브릿지를 만드는 방법에 대해 설명합니다. 브릿지 만들기 옵션에 대한 자세한 내용은 dladm(1M) 매뉴얼 페이지의 dladm create-bridge 명령 설명을 참조하십시오.

주 – TRILL을 사용하여 브릿지를 만들려면 dladm create-bridge 명령에 -P trill을 추가합니다. 또는 dladm modify-bridge 명령을 사용하여 TRILL을 사용으로 설정합니다.

---

dladm create-bridge 명령은 브릿지 인스턴스를 만들고 선택적으로 하나 이상의 네트워크 링크를 새 브릿지에 지정합니다. 기본적으로 시스템에 브릿지 인스턴스가 없으므로 Oracle Solaris는 기본적으로 네트워크 링크 간 브릿지를 만들지 않습니다.

링크 간 브릿지를 만들려면 브릿지 인스턴스를 하나 이상 만들어야 합니다. 각 브릿지 인스턴스를 별개입니다. 브릿지 간에는 전달 연결이 없으며 링크가 최대 한 개 브릿지의 구성원입니다.

*bridge-name*은 적합한 SMF 서비스 인스턴스 이름이어야 하는 모든 문자열입니다. 이 이름은 이스케이프 시퀀스가 없는 FMRI 구성 요소입니다. 즉, 공백, ASCII 제어 문자 및 다음 문자가 포함될 수 없습니다.

```
; / ? : @ & = + $ , % < > # "
```

SUNW 문자열로 시작하는 모든 이름과 마찬가지로 이름 default는 예약되어 있습니다. 후행 숫자가 있는 이름은 디버깅에 사용되는 관찰 장치 생성을 위해 예약되어 있습니다. 관찰 장치 사용으로 인해 올바른 브릿지 인스턴스의 이름이 올바른 dlpi 이름이 되도록 추가로 제한됩니다. 이름은 알파벳 문자나 밑줄 문자로 시작하고 끝나야 합니다. 이름의 나머지 부분에는 영숫자와 밑줄 문자를 사용할 수 있습니다.

1 **관리자로 전환합니다.**

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 **브릿지를 만듭니다.**

```
# dladm create-bridge [-l link]... bridge-name
```

-l *link* 옵션은 브릿지에 링크를 추가합니다. 지정한 링크를 추가할 수 없는 경우 명령이 실패하고 브릿지가 생성되지 않습니다.

다음 예에서는 net0 및 net1 링크를 연결하여 brooklyn 브릿지를 만드는 방법을 보여줍니다.

```
# dladm create-bridge -l net0 -l net1 brooklyn
```

▼ 브릿지에 대한 보호 유형을 수정하는 방법

이 절차에서는 dladm modify-bridge 명령을 사용하여 보호 유형을 STP에서 TRILL로 또는 TRILL에서 STP로 수정하는 방법을 설명합니다.

- **브릿지에 대한 보호 유형을 수정합니다.**

```
# dladm modify-bridge -P protection-type bridge-name
```

-P *protection-type* 옵션은 사용할 보호 유형을 지정합니다. 기본적으로 보호 유형은 STP(-P stp)입니다. TRILL 보호 유형을 사용하려면 -P trill 옵션을 사용합니다.

다음 예에서는 brooklyn 브릿지에 대한 보호 유형을 기본값인 STP에서 TRILL로 수정하는 방법을 보여줍니다.

```
# dladm modify-bridge -P trill brooklyn
```

▼ 기존 브릿지에 링크를 하나 이상 추가하는 방법

이 절차에서는 브릿지 인스턴스에 링크를 하나 이상 추가하는 방법을 설명합니다.

링크는 최대 한 개 브릿지의 구성원일 수 있습니다. 따라서 한 브릿지 인스턴스에서 다른 브릿지 인스턴스로 링크를 이동하려면 먼저 현재 브릿지에서 링크를 제거한 후 다른 브릿지에 추가해야 합니다.

브릿지에 지정되는 링크는 VLAN, VNIC 또는 터널일 수 없습니다. 통합의 일부로 허용되는 링크 또는 통합 자체인 링크만 브릿지에 지정할 수 있습니다.

동일한 브릿지에 지정되는 링크는 동일한 MTU 값을 가져야 합니다. Oracle Solaris에서는 기존 링크의 MTU 값을 변경할 수 있습니다. 그러나 브릿지를 다시 시작하기 전에 MTU 값이 일치하도록 지정된 링크를 제거하거나 변경할 때까지 브릿지 인스턴스가 유지 관리 상태로 전환됩니다.

브릿지에 지정되는 링크는 802.3 및 802.11 매체를 포함하는 이더넷 유형이어야 합니다.

1 **관리자로 전환합니다.**

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 **기존 브릿지에 새 링크를 추가합니다.**

```
# dladm add-bridge -l new-link bridge-name
```

다음 예에서는 기존 브릿지 `rialto`에 `net2` 링크를 추가하는 방법을 보여줍니다.

```
# dladm add-bridge -l net2 rialto
```

▼ 브릿지에서 링크를 제거하는 방법

이 절차에서는 브릿지 인스턴스에서 링크를 하나 이상 제거하는 방법을 설명합니다. 브릿지를 삭제하려는 경우 이 절차를 사용합니다. 브릿지를 삭제하려면 먼저 해당 링크를 모두 제거해야 합니다.

1 **관리자로 전환합니다.**

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 **브릿지에서 링크를 제거합니다.**

```
# dladm remove-bridge [-l link]... bridge-name
```

다음 예에서는 `charles` 브릿지에서 `net0`, `net1` 및 `net2` 링크를 제거하는 방법을 보여줍니다.

```
# dladm remove-bridge -l net0 -l net1 -l net2 charles
```

▼ 시스템에서 브릿지를 삭제하는 방법

이 절차에서는 브릿지 인스턴스를 삭제하는 방법을 설명합니다. 브릿지를 삭제하려면 먼저 `dladm remove-bridge` 명령을 실행하여 연결된 링크를 모두 비활성화해야 합니다. 64 페이지 "브릿지에서 링크를 제거하는 방법"을 참조하십시오.

1 관리자로 전환합니다.

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 시스템에서 브릿지를 삭제합니다.

```
# dladm delete-bridge bridge-name
```

다음 예에서는 먼저 `coronado` 브릿지에서 `net0`, `net1` 및 `net2` 링크를 제거한 다음 시스템에서 브릿지 자체를 제거하는 방법을 보여줍니다.

```
# dladm remove-bridge -l net0 -l net1 -l net2 coronado
# dladm delete-bridge coronado
```

◆ ◆ ◆ 5 장

5 IPMP 소개

IPMP(IP 네트워크 다중 경로)는 여러 IP 인터페이스를 하나의 논리적 인터페이스로 그룹화할 수 있는 계층 3 기술입니다. 실패 감지, 투명한 액세스 페일오버, 패킷 부하 분산과 같은 기능을 갖추고 있으므로 IPMP는 시스템에서 네트워크가 항상 사용 가능하도록 하여 네트워크 성능을 향상시킵니다.

이 장에서는 다음 내용을 다룹니다.

주 – 이 장과 6 장, “IPMP 관리(작업)”의 IPMP 설명 전체에서 용어 **인터페이스**는 모두 *IP* **인터페이스**를 의미합니다. NIC(네트워크 인터페이스 카드)와 같이 용어 정의에서 용어의 다른 사용을 명시적으로 나타내지 않는 경우 이 용어는 항상 IP 계층에 구성된 인터페이스를 가리킵니다.

Oracle Solaris의 IPMP

Oracle Solaris의 IPMP에는 다음 기능이 포함되어 있습니다.

- IPMP를 사용하면 여러 IP 인터페이스를 IPMP 그룹이라고 부르는 하나의 그룹으로 구성할 수 있습니다. 여러 기본 IP 인터페이스가 포함된 IPMP 그룹은 전체적으로 하나의 *IPMP* **인터페이스**로 표시됩니다. 이 인터페이스는 네트워크 스택의 IP 계층에 있는 다른 인터페이스와 동일하게 처리됩니다. 모든 IP 관리 작업, 경로 지정 테이블, ARP(Address Resolution Protocol) 테이블, 방화벽 규칙 및 기타 IP 관련 절차는 IPMP 인터페이스를 참조하여 IPMP 그룹과 함께 작동합니다.

- 시스템은 기본 활성 인터페이스에서 데이터 주소 분배를 처리합니다. IPMP 그룹이 생성되면 데이터 주소가 주소 풀로 IPMP 인터페이스에 속합니다. 그런 다음 커널이 데이터 주소를 그룹의 기본 활성 인터페이스에 임의로 자동 바인딩합니다.
- `ipmpstat` 도구는 IPMP 그룹에 대한 정보를 가져오는 주요 도구입니다. 이 명령은 그룹의 기본 IP 인터페이스, 테스트 및 데이터 주소, 사용되는 실패 감지 유형, 실패한 인터페이스 등 IPMP 구성의 모든 측면에 대한 정보를 제공합니다. `ipmpstat` 함수, 사용할 수 있는 옵션 및 각 옵션이 생성하는 출력 결과는 모두 101 페이지 "IPMP 정보 모니터링"에서 설명합니다.
- IPMP 그룹을 식별하기 쉽도록 IPMP 인터페이스에 사용자 정의 이름을 지정할 수 있습니다. 85 페이지 "IPMP 그룹 구성"을 참조하십시오.

IPMP 사용의 이점

인터페이스 오류 또는 유지 보수를 위한 인터페이스 오프라인 전환 등 다른 요인으로 인해 인터페이스가 사용 불가능하게 될 수 있습니다. IPMP가 없으면 사용할 수 없는 인터페이스와 연결된 IP 주소를 사용하여 더 이상 시스템에 연결할 수 없습니다. 또한 해당 IP 주소를 사용하는 기존 연결이 손상됩니다.

IPMP를 사용하면 여러 IP 인터페이스를 *IPMP* **그룹**으로 구성할 수 있습니다. 이 그룹은 네트워크 트래픽을 보내거나 받는 데이터 주소가 있는 IP 인터페이스로 작동합니다. 그룹의 기본 인터페이스가 실패할 경우 데이터 주소가 그룹의 나머지 기본 활성 인터페이스에 재배포됩니다. 따라서 인터페이스 실패 후에도 그룹이 네트워크 연결을 유지합니다. IPMP를 사용하면 그룹에 대해 사용할 수 있는 인터페이스가 하나만 있어도 항상 네트워크 연결을 사용할 수 있습니다.

IPMP는 IPMP 그룹의 인터페이스 세트에서 아웃바운드 네트워크 트래픽을 자동으로 분산하여 전체 네트워크 성능을 향상시킵니다. 이 프로세스를 아웃바운드 **부하 분산**이라고 합니다. 또한 시스템은 응용 프로그램에서 해당 IP 소스 주소가 지정되지 않은 패킷에 대해 소스 주소를 선택하여 인바운드 부하 분산을 간접적으로 제어합니다. 그러나 응용프로그램에서 명시적으로 IP 소스 주소를 선택한 경우 시스템에서도 해당 소스 주소가 사용됩니다.

주 – 링크 통합은 IPMP와 유사한 기능을 수행하여 네트워크 성능 및 가용성을 향상시킵니다. 이러한 두 기술에 대한 비교는 부록 B, "링크 통합 및 IPMP: 기능 비교"를 참조하십시오.

IPMP 사용 규칙

IPMP 그룹의 구성은 시스템 구성에 의해 결정됩니다.

IPMP를 사용할 때는 다음 규칙을 따르십시오.

- 동일한 LAN의 여러 IP 인터페이스를 IPMP 그룹으로 구성해야 합니다. 광범위한 의미의 LAN은 해당 노드가 동일한 링크 계층 브로드캐스트 도메인에 속하는 유무선 로컬 네트워크와 VLAN을 포함하는 다양한 로컬 네트워크 구성을 나타냅니다.

  주 – 동일한 링크 계층(L2) 브로드캐스트 도메인의 여러 IPMP 그룹은 지원되지 않습니다. L2 브로드캐스트 도메인은 일반적으로 특정 서브넷에 매핑됩니다. 따라서 서브넷당 IPMP 그룹 한 개만 구성해야 합니다.

- IPMP 그룹의 기본 IP 인터페이스가 여러 LAN에 걸쳐 있지 않아야 합니다.

예를 들어, 세 개의 인터페이스가 있는 시스템이 두 개의 개별 LAN에 연결되어 있다고 가정합니다. 두 개의 IP 인터페이스가 하나의 LAN에 연결되고 남은 단일 IP 인터페이스가 다른 LAN에 연결됩니다. 이 경우 첫번째 규칙에 따라 첫번째 LAN에 연결한 두 개의 IP 인터페이스를 IPMP 그룹으로 구성해야 합니다. 두번째 규칙에 따라 두번째 LAN에 연결한 단일 IP 인터페이스는 해당 IPMP 그룹의 구성원이 될 수 없습니다. 단일 IP 인터페이스의 경우 IPMP 구성이 필요하지 않습니다. 하지만 인터페이스의 가용성을 모니터하기 위해 단일 인터페이스를 IPMP 그룹으로 구성할 수 있습니다. 단일 인터페이스 IPMP 구성은 70 페이지 “IPMP 인터페이스 구성 유형”에서 자세히 설명합니다.

첫번째 LAN에 대한 링크가 IP 인터페이스 세 개로 구성되고 다른 링크는 인터페이스 두 개로 구성된 또 다른 사례를 고려해 보십시오. 이 설치에서는 IPMP 그룹 두 개를 구성해야 합니다. 첫번째 LAN에 연결하는 세 개의 인터페이스가 한 그룹이고, 두번째 LAN에 연결하는 두 개의 인터페이스가 또 다른 그룹입니다.

IPMP 구성 요소

IPMP 소프트웨어 구성 요소는 다음과 같습니다.

- **다중 경로 데몬**, `in.mpathd` - 인터페이스 실패 및 복구를 감지합니다. 기본 인터페이스에 대해 테스트 주소가 구성되어 있을 경우 이 데몬은 링크 기반 실패 감지와 프로브 기반 실패 감지를 모두 수행합니다. 사용되는 실패 감지 방법의 유형에 따라 데몬은 인터페이스에서 해당 플래그를 설정하거나 지워 인터페이스가 실패했는지 또는 복구되었는지를 나타냅니다. 옵션으로, IPMP 그룹에 속하도록 구성되지 않은 인터페이스를 포함하여 모든 인터페이스의 가용성을 모니터하도록 데몬을 구성할 수도 있습니다. 실패 감지에 대한 설명은 77 페이지 “IPMP의 실패 감지”를 참조하십시오.

또한 in.mpathd 데몬은 IPMP 그룹의 활성 인터페이스 지정을 제어합니다. 이 데몬은 IPMP 그룹을 만들 때 구성된 원래 활성 인터페이스 수를 동일하게 유지 관리합니다. 따라서 in.mpathd는 관리자가 구성한 정책에 따라 필요할 경우 기본 인터페이스를 활성화하거나 비활성화합니다. in.mpathd 데몬이 기본 인터페이스의 활성화를 관리하는 방식에 대한 자세한 내용은 71 페이지 "IPMP 작동 방식"을 참조하십시오. 데몬에 대한 자세한 내용은 in.mpathd(1M) 매뉴얼 페이지를 참조하십시오.

- **IP 커널 모듈** - IPMP 그룹에서 사용 가능한 IP 데이터 주소 세트를 그룹에서 사용 가능한 기본 IP 인터페이스 세트에 분배하여 아웃바운드 부하 분산을 관리합니다. 또한 이 모듈은 소스 주소를 선택하여 인바운드 부하 분산을 관리합니다. 모듈의 두 역할은 모두 네트워크 트래픽 성능을 향상시킵니다.
- **IPMP 구성 파일**(/etc/default/mpathd) - 데몬 동작을 정의합니다.

  이 파일을 사용자 정의하려면 다음 매개변수를 설정합니다.

  - 프로브 기반 실패 감지를 실행하는 경우 프로빙할 대상 인터페이스
  - 실패를 감지하기 위해 대상을 프로빙하는 기간
  - 실패 인터페이스 복구 후 해당 인터페이스에 플래그 지정할 상태
  - 모니터할 IP 인터페이스 범위 및 IPMP 그룹에 속하도록 구성되지 않은 시스템의 IP 인터페이스를 포함할 것인지 여부

  구성 파일을 수정하는 절차는 100 페이지 "IPMP 데몬의 동작을 구성하는 방법"을 참조하십시오.
- ipmpstat **명령** - 전체적인 IPMP 상태에 대한 다른 유형의 정보를 제공합니다. 또한 이 도구는 그룹에 대해 구성된 데이터 및 테스트 주소뿐 아니라 각 IPMP 그룹의 기본 IP 인터페이스에 대한 다른 정보도 표시합니다. 이 명령에 대한 자세한 내용은 101 페이지 "IPMP 정보 모니터링" 및 ipmpstat(1M) 매뉴얼 페이지를 참조하십시오.

IPMP 인터페이스 구성 유형

IPMP 구성은 일반적으로 동일한 LAN에 연결된 동일한 시스템에 있는 둘 이상의 물리적 인터페이스로 구성됩니다. 이러한 인터페이스는 다음 구성 중 하나로 IPMP 그룹에 속할 수 있습니다.

- **활성-활성 구성** – 모든 기본 인터페이스의 IPMP 그룹이 활성입니다. **활성 인터페이스**는 현재 IPMP 그룹이 사용할 수 있는 IP 인터페이스입니다.

  주 – 기본적으로 IPMP 그룹에 속하도록 인터페이스를 구성하면 기본 인터페이스가 활성화됩니다.

- **활성-대기 구성** – 관리상 인터페이스가 하나 이상 **대기 인터페이스**로 구성되어 있는 IPMP 그룹입니다. 유휴 상태이긴 하지만 대기 인터페이스가 인터페이스 구성 방식에 따라 인터페이스의 가용성을 추적하기 위해 다중 경로 데몬에 의해 모니터됩니다.

인터페이스에서 링크 실패 알림을 지원하는 경우 링크 기반 실패 감지가 사용됩니다. 인터페이스가 테스트 주소로 구성된 경우 프로브 기반 실패 감지도 사용됩니다. 활성 인터페이스가 실패할 경우 대기 인터페이스가 필요에 따라 자동으로 배포됩니다. IPMP 그룹에 대해 원하는 개수만큼 대기 인터페이스를 구성할 수 있습니다.

단일 인터페이스를 고유한 IPMP 그룹으로 구성할 수도 있습니다. 단일 인터페이스 IPMP 그룹은 여러 인터페이스가 포함된 IPMP 그룹과 동일한 방식으로 동작합니다. 하지만 이 IPMP 구성은 네트워크 트래픽에 고가용성을 제공하지 않습니다. 기본 인터페이스가 실패할 경우 시스템에서 트래픽을 보내거나 받는 기능이 모두 손실됩니다. 단일 인터페이스 IPMP 그룹을 구성하는 목적은 실패 감지를 사용하여 인터페이스의 가용성을 모니터하기 위한 것입니다. 인터페이스에 테스트 주소를 구성하면 다중 경로 데몬에서 프로브 기반 실패 감지를 사용하여 인터페이스를 추적할 수 있습니다.

일반적으로 단일 인터페이스 IPMP 그룹 구성은 Oracle Solaris Cluster 소프트웨어와 같이 광범위한 페일오버 기능을 가진 다른 기술과 함께 사용됩니다. 시스템이 기본 인터페이스 상태를 계속 모니터할 수 있으나 Oracle Solaris Cluster 소프트웨어에서 실패 발생 시 네트워크 가용성을 확인하는 기능을 제공합니다. Oracle Solaris Cluster 소프트웨어에 대한 자세한 내용은 **Oracle Solaris Cluster Concepts Guide**를 참조하십시오.

기본 인터페이스가 제거된 그룹과 같이 기본 인터페이스가 없는 IPMP 그룹도 존재할 수 있습니다. IPMP 그룹은 삭제되지 않지만 이 그룹을 사용하여 트래픽을 보내고 받을 수는 없습니다. 그룹에 대해 기본 인터페이스를 온라인 상태로 전환하면 IPMP 인터페이스의 데이터 주소가 해당 인터페이스에 할당되며 시스템이 네트워크 트래픽을 계속 호스트합니다.

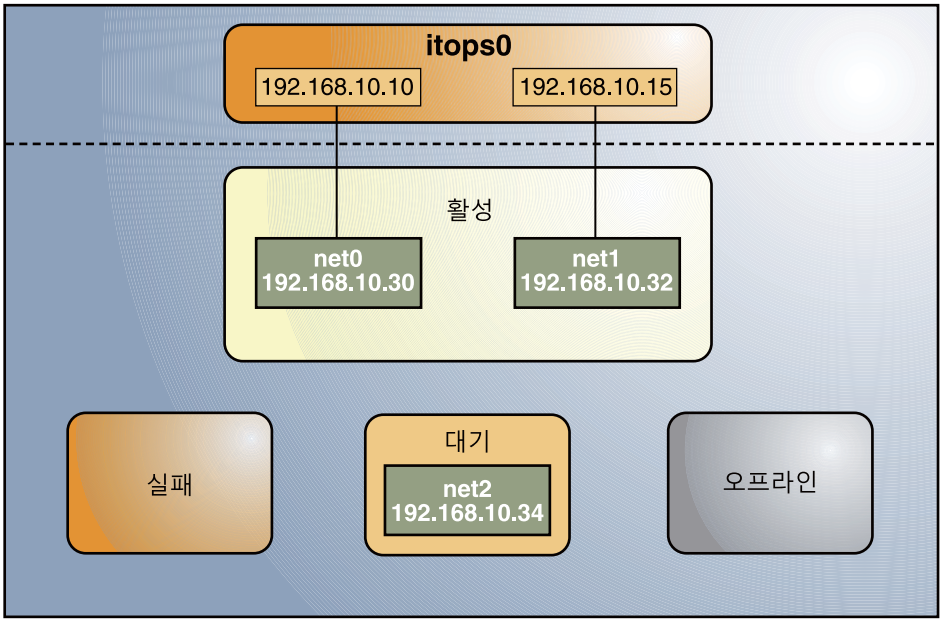
IPMP 작동 방식

IPMP는 IPMP 그룹 생성 시 구성된 원래 활성 및 대기 인터페이스 수를 동일하게 유지하여 네트워크 가용성을 유지 관리합니다.

IPMP 실패 감지는 그룹에서 특정 기본 IP 인터페이스의 가용성을 확인하기 위한 링크 기반, 프로브 기반 또는 둘 다일 수 있습니다. IPMP에서 기본 인터페이스가 실패했음을 확인하면 해당 인터페이스에 failed 플래그가 지정되며 더 이상 사용할 수 없습니다. 실패한 인터페이스와 연결된 데이터 IP 주소가 그룹에서 작동하는 다른 인터페이스에 재배포됩니다. 사용 가능한 경우 활성 인터페이스의 원래 개수를 유지 관리하기 위해 대기 인터페이스도 배포됩니다.

다음 그림에 설명된 것처럼 활성-대기 구성을 사용하는, 인터페이스가 3개인 IPMP 그룹 `itops0`을 고려해 보십시오.

그림 5–1 IPMP 활성-대기 구성



IPMP 그룹 itops0은 다음과 같이 구성됩니다.

- 192.168.10.10과 192.168.10.15라는 두 개의 데이터 주소가 그룹에 지정됩니다.
- 기본 인터페이스 두 개가 활성 인터페이스로 구성되고 net0과 net1이라는 유연한 링크 이름이 지정됩니다.
- 그룹에는 net2라는 유연한 링크 이름을 가진 대기 인터페이스 한 개가 있습니다.
- 프로브 기반 실패 감지가 사용되므로 활성 및 대기 인터페이스가 다음과 같이 테스트 주소로 구성됩니다.
  - net0: 192.168.10.30
  - net1: 192.168.10.32
  - net2: 192.168.10.34

주 – 그림 5–1, 그림 5–2, 그림 5–3 및 그림 5–4의 활성, 오프라인, 대기 및 실패 영역은 물리적 위치가 아닌 기본 인터페이스의 상태만 나타냅니다. 이 IPMP 구현에서 인터페이스 또는 주소의 물리적 이동이나 IP 인터페이스의 전송은 발생하지 않습니다. 이 영역은 실패 또는 복구의 결과로 기본 인터페이스의 상태가 어떻게 변경되는지만 보여줍니다.

ipmpstat 명령을 여러 옵션과 함께 사용하여 기존 IPMP 그룹에 대한 특정 유형의 정보를 표시할 수 있습니다. 추가 예를 보려면 101 페이지 “IPMP 정보 모니터링”을 참조하십시오.

다음 ipmpstat 명령은 그림 5–1에 있는 IPMP 구성에 대한 정보를 표시합니다.

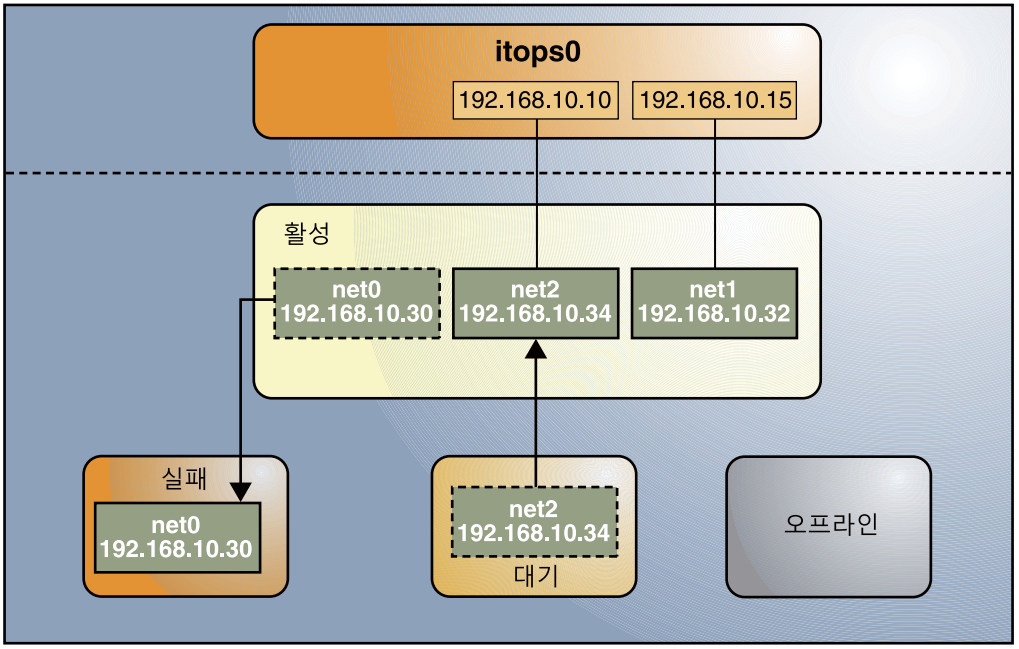
```
# ipmpstat -g
GROUP     GROUPNAME     STATE     FDT       INTERFACES
itops0    itops0        ok        10.00s    net1 net0 (net2)
```

그룹의 기본 인터페이스에 대한 정보를 표시하려면 다음을 입력합니다.

```
# ipmpstat -i
INTERFACE   ACTIVE    GROUP     FLAGS      LINK       PROBE     STATE
net0        yes       itops0    -------    up         ok        ok
net1        yes       itops0    --mb---    up         ok        ok
net2        no        itops0    is-----    up         ok        ok
```

IPMP는 기본 인터페이스 관리를 통해 활성 인터페이스의 원래 개수를 보존하여 네트워크 가용성을 유지 관리합니다. 따라서 net0이 실패할 경우 net2가 배포되어 IPMP 그룹에서 활성 인터페이스 두 개가 유지되도록 합니다. net2 활성화는 다음 그림에 나와 있습니다.

그림 5–2 IPMP의 인터페이스 실패



주 – 그림 5–2에 있는 데이터 주소와 활성 인터페이스 간 일대일 매핑은 설명을 단순화하기 위한 것일 뿐입니다. IP 커널 모듈은 데이터 주소와 인터페이스 간 일대일 관계를 준수할 필요 없이 임의로 데이터 주소를 지정할 수 있습니다.

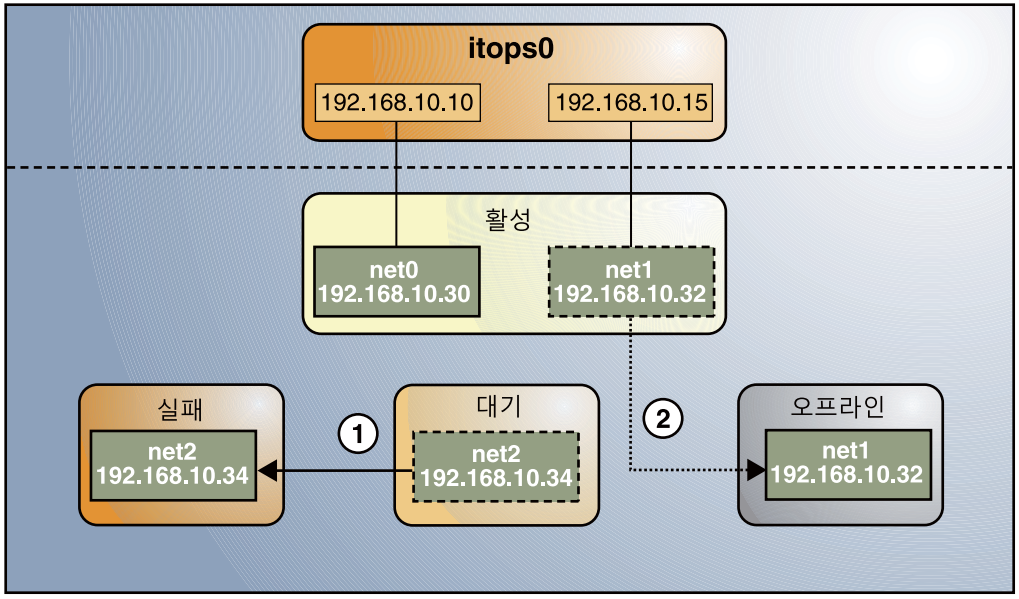
ipmpstat 명령은 그림 5–2의 정보를 다음과 같이 표시합니다.

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP      FLAGS      LINK       PROBE      STATE
net0        no         itops0     -------    up         failed     failed
net1        yes        itops0     --mb---    up         ok         ok
net2        yes        itops0     -s-----    up         ok         ok
```

net0은 복구 후 활성 인터페이스 상태로 돌아갑니다. net2는 원래의 대기 상태로 돌아갑니다.

다른 실패 시나리오는 그림 5–3에 나와 있으며, 여기에서는 대기 인터페이스 net2가 실패합니다(1). 나중에 활성 인터페이스, net1이 관리자에 의해 오프라인으로 전환됩니다(2). 그 결과, IPMP 그룹에서 작동하는 인터페이스는 net0 한 개뿐입니다.

그림 5–3 IPMP의 대기 인터페이스 실패



ipmpstat 명령은 그림 5–3의 정보를 다음과 같이 표시합니다.

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP      FLAGS       LINK        PROBE      STATE
net0        yes        itops0     -------     up          ok         ok
net1        no         itops0     --mb-d-     up          ok         offline
net2        no         itops0     is-----     up          failed     failed
```

이 특정 실패의 경우 인터페이스가 복구된 후의 복구가 다르게 작동합니다. 복구 후의 구성과 비교하여 복구 프로세스는 IPMP 그룹의 원래 활성 인터페이스 수에 따라 다릅니다. 복구 프로세스는 다음 그림에 그래픽으로 표시되어 있습니다.

그림 5–4 IPMP 복구 프로세스

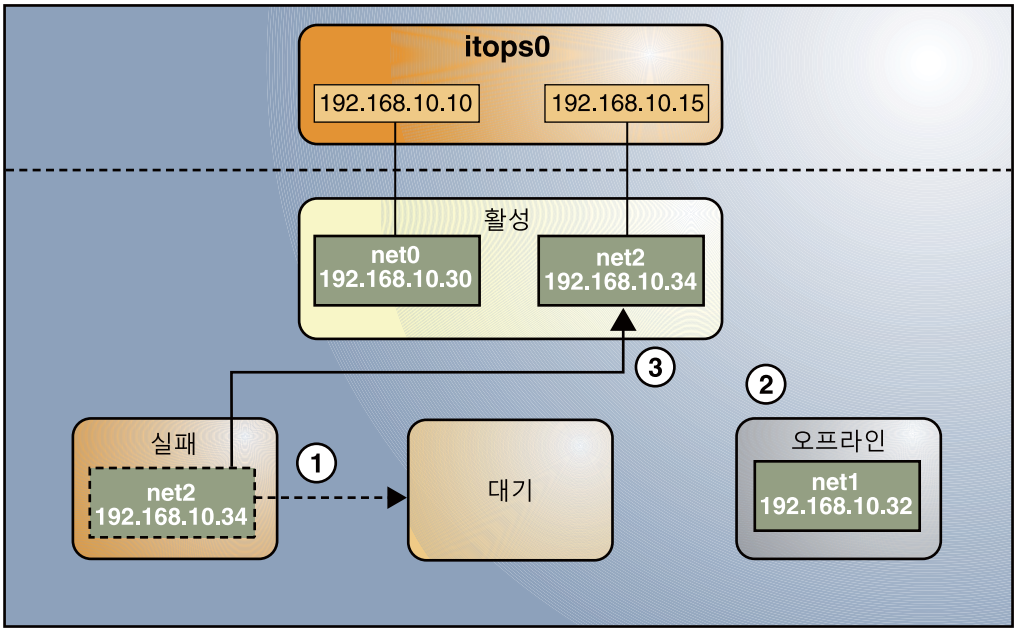


그림 5–4에서는 net2가 복구되면 정상적으로 원래 대기 인터페이스 상태로 돌아갑니다(1). 하지만 net1이 계속 오프라인 상태로 유지되므로 IPMP 그룹이 원래 활성 인터페이스 두 개를 반영하지 않습니다(2). 따라서 IPMP가 net2를 활성 인터페이스로 대신 배포합니다(3).

ipmpstat 명령은 복구 후 IPMP 시나리오를 다음과 같이 표시합니다.

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP      FLAGS       LINK        PROBE      STATE
net0        yes        itops0     -------     up          ok         ok
net1        no         itops0     --mb-d-     up          ok         offline
net2        yes        itops0     -s-----     up          ok         ok
```

실패한 활성 인터페이스가 복구 시 자동으로 활성 상태로 돌아가지 않는 FAILBACK=no 모드로 구성된 활성 인터페이스가 실패에 관련된 경우 유사한 복구 프로세스가 발생합니다. 그림 5–2의 net0이 FAILBACK=no 모드로 구성되었다고 가정합니다. 이 모드에서는 복구된 net0이 원래 활성 인터페이스였어도 대기 인터페이스가 됩니다. net2 인터페이스가 활성 상태로 유지되어 IPMP 그룹의 원래 활성 인터페이스 두 개를 유지합니다.

ipmpstat 명령은 복구 정보를 다음과 같이 표시합니다.

```
# ipmpstat -i
INTERFACE   ACTIVE     GROUP      FLAGS      LINK       PROBE      STATE
net0        no         itops0     i------    up         ok         ok
net1        yes        itops0     --mb---    up         ok         ok
net2        yes        itops0     -s-----    up         ok         ok
```

이 유형의 구성에 대한 자세한 내용은 81 페이지 “FAILBACK=no 모드”를 참조하십시오.

IPMP 주소 지정

IPv4 네트워크와 이중 스택 IPv4 및 IPv6 네트워크에서 모두 IPMP 실패 감지를 구성할 수 있습니다. IPMP를 사용하여 구성된 인터페이스는 다음 절에 설명된 두 가지 유형의 주소를 지원합니다. Oracle Solaris 11부터는 IP 주소가 IPMP 인터페이스**에만** 상주하며, 데이터 주소로 지정됩니다. 반면에 테스트 주소는 기본 인터페이스에 상주합니다.

데이터 주소

데이터 주소는 부트 시 DHCP 서버에 의해 동적으로 또는 ipadm 명령을 사용하여 수동으로 IP 인터페이스에 지정된 일반적인 IPv4 및 IPv6 주소입니다. IPMP 인터페이스에는 데이터 주소가 지정됩니다. 표준 IPv4 패킷 트래픽과 해당하는 경우 IPv6 패킷 트래픽은 **데이터 트래픽**으로 간주됩니다. 데이터 트래픽은 IPMP 인터페이스에 호스트된 데이터 주소를 사용하여 해당 IPMP 인터페이스 또는 그룹의 활성 인터페이스를 통해 전달됩니다.

테스트 주소

테스트 주소는 in.mpathd 데몬에서 프로브 기반 실패 또는 복구 감지를 수행하는 데 사용되는 IPMP 특정 주소입니다. 테스트 주소도 DHCP 서버에 의해 동적으로 또는 ipadm 명령을 사용하여 수동으로 지정될 수 있습니다. IPMP 그룹의 기본 인터페이스에 테스트 주소만 지정합니다. 기본 인터페이스가 실패할 경우 in.mpathd 데몬은 해당 인터페이스의 테스트 주소를 프로브 기반 실패 감지에 계속 사용하여 인터페이스의 후속 복구를 확인합니다.

주 – 프로브 기반 실패 감지를 사용하려는 경우에만 테스트 주소를 구성해야 합니다. 그렇지 않으면 테스트 주소를 사용하지 않고 전이적 프로브에서 실패를 감지하도록 할 수 있습니다. 테스트 주소를 사용하거나 사용하지 않는 프로브 기반 실패 감지에 대한 자세한 내용은 78 페이지 “프로브 기반 실패 감지”를 참조하십시오.

이전 IPMP 구현에서는 특히 인터페이스 실패 시 응용프로그램이 사용하지 않도록 테스트 주소를 `DEPRECATED`로 표시해야 했습니다. 현재 구현에서는 테스트 주소가 기본 인터페이스에 있습니다. 따라서 IPMP를 인식하지 않는 응용 프로그램이 해당 주소를 실수로 사용할 수 없습니다. 하지만 이러한 주소가 가능한 데이터 패킷 소스로 간주되지 않도록 시스템에서 자동으로 `NOFAILOVER` 플래그가 설정된 주소를 `DEPRECATED`로 표시합니다.

IPv4 주소를 서브넷에서 테스트 주소로 사용할 수 있습니다. IPv4 주소는 대부분의 사이트에서 제한된 리소스이므로 경로 지정 불가능한 RFC 1918 개인 주소를 테스트 주소로 사용하는 것이 좋습니다. `in.mpathd` 데몬은 테스트 주소와 동일한 서브넷에 있는 다른 호스트하고만 ICMP 프로브를 교환합니다. RFC 1918 스타일 테스트 주소를 사용하는 경우 해당 RFC 1918 서브넷의 주소를 가진 네트워크에서 다른 시스템(특히 라우터)을 구성해야 합니다. 그러면 `in.mpathd` 데몬이 대상 시스템과 프로브를 성공적으로 교환할 수 있습니다. RFC 1918 개인 주소에 대한 자세한 내용은 RFC 1918, Address Allocation for Private Internets (`http://www.ietf.org/rfc/rfc1918.txt?number=1918`)를 참조하십시오.

유효한 IPv6 테스트 주소는 물리적 인터페이스의 링크-로컬 주소뿐입니다. IPMP 테스트 주소로 사용할 별도의 IPv6 주소가 필요 없습니다. IPv6 링크-로컬 주소는 인터페이스의 MAC(Media Access Control) 주소를 기반으로 합니다. 인터페이스가 부트 시 IPv6 사용으로 설정되거나 `ipadm` 명령을 통해 인터페이스를 수동으로 구성하면 링크-로컬 주소가 자동으로 구성됩니다.

IPMP 그룹의 모든 인터페이스에 IPv4와 IPv6이 모두 연결되어 있는 경우 별도의 IPv4 테스트 주소를 구성할 필요가 없습니다. `in.mpathd` 데몬은 IPv6 링크-로컬 주소를 테스트 주소로 사용할 수 있습니다.

IPMP의 실패 감지

네트워크가 계속해서 트래픽을 보내거나 받을 수 있도록 IPMP는 IPMP 그룹의 기본 IP 인터페이스에서 실패 감지를 수행합니다. 실패한 인터페이스는 복구될 때까지 사용할 수 없습니다. 나머지 활성 인터페이스는 계속 작동하고 기존의 대기 인터페이스가 필요에 따라 배포됩니다.

`in.mpathd` 데몬은 다음 유형의 실패 감지를 처리합니다.

- 프로브 기반 실패 감지. 다음 두 가지 유형이 있습니다.
  - 테스트 주소가 구성되지 않음(전이적 프로브)

  - 테스트 주소가 구성됨
- 링크 기반 실패 감지, NIC 드라이버에서 지원되는 경우

프로브 기반 실패 감지

프로브 기반 실패 감지는 ICMP 프로브를 사용하여 인터페이스 실패 여부를 확인하는 작업으로 구성됩니다. 이 실패 감지 방법의 구현은 테스트 주소의 사용 여부에 따라 달라집니다.

테스트 주소를 사용하는 프로브 기반 실패 감지

이 실패 감지 방법에서는 테스트 주소를 사용하는 ICMP 프로브 메시지를 보내고 받습니다. **프로브 트래픽** 또는 **테스트 트래픽**이라고도 하는 이 메시지는 인터페이스를 통해 동일한 로컬 네트워크에 있는 하나 이상의 대상 시스템으로 전송됩니다. `in.mpathd` 데몬은 프로브 기반 실패 감지가 구성된 모든 인터페이스를 통해 모든 대상을 개별적으로 프로빙합니다. 지정된 인터페이스에 대한 5회 연속 프로브에 대해 응답이 없을 경우 `in.mpathd`는 해당 인터페이스가 실패했다고 간주합니다. 프로빙 속도는 *FDT*(**실패 감지 시간**)에 따라 달라집니다. 실패 감지 시간의 기본값은 10초입니다. 하지만 IPMP 구성 파일에서 FDT를 전환할 수 있습니다. 지침은 100 페이지 “IPMP 데몬의 동작을 구성하는 방법”을 참조하십시오.

프로브 기반 실패 감지를 최적화하려면 `in.mpathd` 데몬에서 프로브를 받을 대상 시스템을 여러 개 설정해야 합니다. 여러 대상 시스템을 사용하면 보고된 실패의 특성을 확인하는 데 도움이 됩니다. 예를 들어, 정의된 유일한 대상 시스템에서 응답이 없을 경우 시스템이 대상 시스템이나 IPMP 그룹의 인터페이스 중 하나에 실패를 표시할 수 있습니다. 반면, 여러 대상 시스템 중에서 한 시스템만 프로브에 응답하지 않는 경우 실패가 IPMP 그룹 자체가 아니라 대상 시스템에서 발생했을 가능성이 큽니다.

`in.mpathd` 데몬은 동적으로 프로빙할 대상 시스템을 결정합니다. 먼저 데몬이 IPMP 그룹의 인터페이스와 연결된 테스트 주소와 동일한 서브넷에 있는 대상 시스템을 경로 지정 테이블에서 검색합니다. 이러한 대상이 있으면 데몬이 프로브 대상으로 사용합니다. 동일한 서브넷에 대상 시스템이 없는 경우 데몬은 멀티캐스트 패킷을 보내 링크에서 인접한 호스트를 프로빙합니다. 모든 호스트 멀티캐스트 주소인 `224.0.0.1`(IPv4) 및 `ff02::1`(IPv6)로 멀티캐스트 패킷이 전송되어 대상 시스템으로 사용할 호스트를 결정합니다. 에코 패킷에 응답하는 처음 5개 호스트가 프로브 대상으로 선택됩니다. 데몬이 멀티캐스트 프로브에 응답한 라우터나 호스트를 찾을 수 없는 경우 프로브 기반 실패를 감지할 수 없습니다. 이 경우 `ipmpstat -i` 명령은 프로브 상태를 `unknown`으로 보고합니다.

호스트 경로를 사용하여 `in.mpathd` 데몬에서 사용할 대상 시스템 목록을 명시적으로 구성할 수 있습니다. 지침은 97 페이지 “프로브 기반 실패 감지 구성”을 참조하십시오.

테스트 주소를 사용하지 않는 프로브 기반 실패 감지

테스트 주소를 사용하지 않을 경우 이 방법은 다음 두 가지 프로브 유형으로 구현됩니다.

- **ICMP 프로브**

  ICMP 프로브는 경로 지정 테이블에 정의된 대상을 프로빙하기 위해 IPMP 그룹의 활성 인터페이스에서 전송합니다. **활성** 인터페이스는 인터페이스의 링크 계층(L2) 주소가 지정된 인바운드 IP 패킷을 받을 수 있는 기본 인터페이스입니다. ICMP 프로브는 데이터 주소를 프로브의 소스 주소로 사용합니다. ICMP 프로브가 대상에 도달하고 대상으로부터 응답을 받으면 활성 인터페이스가 작동합니다.

- **추이적인 프로브**

  추이적인 프로브는 활성 인터페이스를 프로빙하기 위해 IPMP 그룹의 대체 인터페이스에서 전송합니다. 대체 인터페이스는 인바운드 IP 패킷을 받지 않는 기본 인터페이스입니다.

  예를 들어, 기본 인터페이스 4개로 구성된 IPMP 그룹을 고려해 보십시오. 이 그룹은 데이터 주소 한 개로 구성되었지만 테스트 주소가 없습니다. 이 구성에서 아웃바운드 패킷은 모두 기본 인터페이스를 사용할 수 있습니다. 하지만 인바운드 패킷은 데이터 주소가 바인딩된 인터페이스만 받을 수 있습니다. 인바운드 패킷을 받을 수 없는 나머지 기본 인터페이스 세 개가 **대체** 인터페이스입니다.

  대체 인터페이스가 성공적으로 활성 인터페이스에 프로브를 보내고 응답을 받을 수 있으면 활성 인터페이스가 작동하며 프로브를 보낸 대체 인터페이스도 작동하는 것입니다.

주 – Oracle Solaris의 프로브 기반 실패 감지는 테스트 주소를 사용하여 작동됩니다. 테스트 주소가 사용되지 않는 프로브 기반 실패 감지를 선택하려면 추이적인 프로빙을 수동으로 사용 설정해야 합니다. 절차는 99 페이지 “사용할 실패 감지 방법을 선택하는 방법”을 참조하십시오.

그룹 실패

IPMP 그룹의 모든 인터페이스가 동시에 실패하면 **그룹 실패**가 발생합니다. 이 경우 기본 인터페이스를 사용할 수 없습니다. 또한 모든 대상 시스템이 동시에 실패하고 프로브 기반 실패 감지가 사용으로 설정된 경우 `in.mpathd` 데몬이 현재 대상 시스템을 모두 비우고 새 대상 시스템을 프로빙합니다.

테스트 주소가 없는 IPMP 그룹에서는 활성 인터페이스를 프로빙할 수 있는 단일 인터페이스가 프로버로 지정됩니다. 이 지정된 인터페이스에는 `FAILED` 플래그와 `PROBER` 플래그가 모두 설정됩니다. 인터페이스가 복구를 감지하기 위한 대상 프로빙을 계속할 수 있도록 데이터 주소가 이 인터페이스에 바인딩됩니다.

링크 기반 실패 감지

인터페이스가 이 유형의 실패 감지를 지원하는 경우 링크 기반 실패 감지가 항상 사용으로 설정됩니다.

타사 인터페이스가 링크 기반 실패 감지를 지원하는지 확인하려면 `ipmpstat -i` 명령을 사용합니다. 지정된 인터페이스에 대한 출력에서 `LINK` 열이 `unknown` 상태로 표시되는 경우 해당 인터페이스는 링크 기반 실패 감지를 지원하지 않습니다. 장치에 대한 자세한 내용은 제조업체 설명서를 참조하십시오.

링크 기반 실패 감지를 지원하는 네트워크 드라이버는 인터페이스의 링크 상태를 모니터하고 해당 링크 상태가 변경될 경우 네트워킹 부속 시스템에 알려줍니다. 변경 알림을 받으면 네트워킹 부속 시스템이 해당 인터페이스에 대해 `RUNNING` 플래그를 적절하게 설정하거나 지웁니다. `in.mpathd` 데몬이 인터페이스의 `RUNNING` 플래그가 지워진 것을 감지하면 데몬이 즉시 인터페이스 실패를 발생시킵니다.

실패 감지 및 익명 그룹 기능

IPMP는 익명 그룹의 실패 감지를 지원합니다. 기본적으로 IPMP는 IPMP 그룹에 속하는 인터페이스의 상태만 모니터합니다. 하지만 IPMP 그룹에 속하지 않는 인터페이스의 상태도 추적하도록 IPMP 데몬을 구성할 수 있습니다. 따라서 이 인터페이스는 "익명 그룹"의 일부로 간주됩니다. `ipmpstat -g` 명령을 실행하면 익명 그룹이 이중 대시(`--`)로 표시됩니다. 익명 그룹에서 인터페이스의 데이터 주소는 테스트 주소 역할도 수행합니다. 이 인터페이스는 명명된 IPMP 그룹에 속하지 않으므로 해당 주소가 응용 프로그램에 표시됩니다. IPMP 그룹에 속하지 않는 인터페이스 추적을 사용으로 설정하려면 100 페이지 "IPMP 데몬의 동작을 구성하는 방법"을 참조하십시오.

물리적 인터페이스 복구 감지

복구 감지 시간은 실패 감지 시간의 두 배입니다. 실패 감지의 기본 시간은 10초입니다. 이에 따라 복구 감지의 기본 시간은 20초입니다. 실패한 인터페이스에 `RUNNING` 플래그가 다시 표시되고 실패 감지 방법에서 인터페이스가 복구된 것으로 감지되면 `in.mpathd` 데몬이 인터페이스의 `FAILED` 플래그를 지웁니다. 복구된 인터페이스는 관리자가 원래 설정한 활성 인터페이스 수에 따라 재배포됩니다.

기본 인터페이스가 실패하고 프로브 기반 실패 감지가 사용되면 테스트 주소가 구성되지 않은 경우 지정된 프로버를 통해 또는 인터페이스의 테스트 주소를 사용하여 `in.mpathd` 데몬이 프로빙을 계속합니다. 인터페이스 복구에서는 실패한 인터페이스의 원래 구성에 따라 다음과 같이 복구 프로세스가 계속됩니다.

- 실패한 인터페이스가 원래 활성 인터페이스였으면 복구된 인터페이스가 원래 활성 상태로 돌아갑니다. 실패 시 대체 역할을 한 대기 인터페이스는 시스템 관리자가 정의한 개수의 인터페이스가 IPMP 그룹에 대해 활성인 경우 다시 대기 상태로 전환됩니다.

주 – 예외는 복구된 활성 인터페이스가 FAILBACK=no 모드로 구성되어 있는 경우입니다. 자세한 내용은 81 페이지 “FAILBACK=no 모드”를 참조하십시오.

- 실패한 인터페이스가 원래 대기 인터페이스였으면 IPMP 그룹에 원래 활성 인터페이스 수가 반영되는 경우 복구된 인터페이스가 원래 대기 상태로 돌아갑니다. 그렇지 않으면 대기 인터페이스가 활성 인터페이스가 됩니다.

인터페이스 실패 및 복구 중 IPMP가 어떻게 작동하는 지에 대한 그래픽 표현을 보려면 71 페이지 “IPMP 작동 방식”을 참조하십시오.

FAILBACK=no 모드

기본적으로 실패 후 복구된 활성 인터페이스는 자동으로 다시 IPMP 그룹의 활성 인터페이스가 됩니다. 이 동작은 in.mpathd 데몬 구성 파일에 있는 FAILBACK 매개변수 값에 따라 제어됩니다. 하지만 데이터 주소를 복구된 인터페이스로 재매핑할 때 발생하는 사소한 중단조차 일부 관리자에게는 허용되지 않을 수 있습니다. 이러한 관리자는 활성화된 대기 인터페이스가 활성 인터페이스로 계속 작동하도록 할 수 있습니다. IPMP를 사용하면 관리자가 기본 동작을 대체하여 인터페이스가 복구 시 자동으로 활성이 되지 않게 할 수 있습니다. 해당 인터페이스는 FAILBACK=no 모드로 구성해야 합니다. 관련 절차는 100 페이지 “IPMP 데몬의 동작을 구성하는 방법”을 참조하십시오.

FAILBACK=no 모드의 활성 인터페이스가 실패하고 이후에 복구되면 in.mpathd 데몬이 IPMP 구성을 다음과 같이 복원합니다.

- IPMP 그룹이 활성 인터페이스의 원래 구성을 반영하는 경우 데몬이 인터페이스의 INACTIVE 상태를 유지합니다.
- 복구 시 IPMP 구성이 활성 인터페이스에 대한 그룹의 원래 구성을 반영하지 않는 경우 FAILBACK=no 상태에 관계없이 복구된 인터페이스가 활성 인터페이스로 재배포됩니다.

주 – FAILBACK=NO 모드는 전체 IPMP 그룹에 대해 설정됩니다. 인터페이스별 조정 가능 매개변수는 아닙니다.

IPMP 및 동적 재구성

Oracle Solaris DR(동적 재구성) 기능을 사용하면 시스템이 실행되는 동안 인터페이스 같은 시스템 하드웨어를 재구성할 수 있습니다. DR은 이 기능을 지원하는 시스템에서만 사용할 수 있습니다. DR이 지원되는 시스템에서는 IPMP가 RCM(Reconfiguration Coordination Manager) 프레임워크로 통합됩니다. 따라서 NIC를 안전하게 연결, 분리 또는 재연결할 수 있으며 RCM이 시스템 구성 요소의 동적 재구성을 관리합니다. 예를 들어 새 인터페이스를 연결하고 기존 IPMP 그룹에 추가할 수 있습니다. 이러한 인터페이스는 구성된 후 IPMP에서 즉시 사용할 수 있습니다.

모든 NIC 분리 요청이 먼저 검사되어 연결을 유지할 수 있는지 확인합니다. 예를 들어, IPMP 그룹에 없는 NIC는 기본적으로 분리할 수 없습니다. IPMP 그룹에서 작동하는 유일한 인터페이스를 포함하는 NIC도 분리할 수 없습니다. 하지만 시스템 구성 요소를 제거해야 하는 경우 `cfgadm(1M)` 매뉴얼 페이지에 설명된 대로 `cfgadm` 명령의 `-f` 옵션을 사용하여 이 동작을 대체할 수 있습니다.

검사에 성공하면 `in.mpathd` 데몬이 인터페이스에 대해 `OFFLINE` 플래그를 설정합니다. 인터페이스에서 모든 테스트 주소의 구성이 해제됩니다. 그런 다음 NIC가 시스템에서 연결 취소됩니다. 이러한 단계 중 하나라도 실패하거나 동일한 시스템 구성 요소에 있는 다른 하드웨어의 DR이 실패하면 이전 구성이 원래 상태로 복원됩니다. 이 이벤트에 대한 상태 메시지가 표시됩니다. 그렇지 않으면 분리 요청이 성공적으로 완료됩니다. 시스템에서 구성 요소를 제거할 수 있습니다. 기존 연결은 중단되지 않습니다.

주 – NIC를 대체하는 경우 이더넷 같은 카드가 동일한 유형인지 확인하십시오. NIC 교체 후에는 지속 IP 인터페이스 구성이 해당 NIC에 적용됩니다.

◆ ◆ ◆ 6 장

6 IPMP 관리(작업)

이 장에서는 Oracle Solaris 11 릴리스에서 IPMP(IP Network Multipathing)를 사용하여 인터페이스 그룹을 관리하는 작업을 제공합니다.

이 장에서는 다음 내용을 다룹니다.

주 – 이 장의 작업에서는 Oracle Solaris 10 및 이전 릴리스에서 사용된 `ifconfig` 명령을 대신하는 `ipadm` 명령을 사용해서 IPMP를 구성하는 방법에 대해 설명합니다. 두 명령이 서로 매핑되는 방법에 대한 자세한 내용을 보려면 **Oracle Solaris 11.1에서 고정된 네트워크 구성을 사용하여 시스템 연결**의 부록 A, "비교 맵: ifconfig 및 ipadm 명령"을 참조하십시오.

Oracle Solaris 11에는 IPMP에 대한 새로운 개념 모델도 도입되었습니다. 자세한 설명은 71 페이지 "IPMP 작동 방식"을 참조하십시오.

IPMP 배치 중 경로 지정 유지 관리

IPMP 그룹을 구성할 때 IPMP 인터페이스는 기본 인터페이스의 IP 주소를 상속하여 이를 데이터 주소로 사용합니다. 그런 다음 기본 인터페이스에 IP 주소 `0.0.0.0`이 수신됩니다. 따라서 특정 IP 인터페이스를 사용해서 정의된 경로는 이러한 인터페이스가 이후에 IPMP 그룹에 추가된 경우 손실됩니다.

IPMP를 구성할 때 경로 지정이 손실되는 경우는 일반적으로 기본 경로와 관련해서 Oracle Solaris 설치 시에 발생합니다. 설치 중에는 사용자가 시스템에서 인터페이스(예: 기본 인터페이스)를 사용할 기본 경로를 정의해야 합니다. 이후에 기본 경로를 정의한

동일한 인터페이스를 사용해서 IPMP 그룹을 구성합니다. IPMP 구성 후에는 인터페이스 주소가 IPMP 인터페이스로 전송되었기 때문에 시스템이 네트워크 패킷의 경로를 더 이상 지정할 수 없습니다.

IPMP를 사용할 때 기본 경로가 보존되도록 하려면 인터페이스를 지정하지 않고 경로를 정의해야 합니다. 이 방식을 사용하면 IPMP 인터페이스를 포함해서 모든 인터페이스를 경로 지정에 사용할 수 있습니다. 따라서 시스템이 트래픽 경로를 계속 지정할 수 있습니다.

주 – 이 단원에서는 기본 경로가 정의된 예로 기본 인터페이스를 사용합니다. 하지만 경로 지정에 사용되었고 나중에 IPMP 그룹의 일부가 된 인터페이스에는 경로 지정 손실 사례가 적용됩니다.

▼ IPMP를 사용하는 동안 경로를 정의하는 방법

다음 절차에서는 IPMP를 구성할 때 기본 경로를 보존하는 방법을 설명합니다.

1. **콘솔을 사용하여 시스템에 로그인합니다.**
   이 절차를 수행하려면 콘솔을 사용해야 합니다. `ssh` 또는 `telnet` 명령을 사용해서 로그인할 경우, 후속 단계를 수행할 때 연결이 손실됩니다.

2. **(옵션) 경로 지정 테이블에 정의된 경로를 표시합니다.**

   ```
   # netstat -nr
   ```

3. **특정 인터페이스에 바인딩된 경로를 삭제합니다.**

   ```
   # route -p delete default gateway-address -ifp interface
   ```

4. **인터페이스를 지정하지 않고 경로를 추가합니다.**

   ```
   # route -p add default gateway-address
   ```

5. **(옵션) 다시 정의된 경로를 표시합니다.**

   ```
   # netstat -nr
   ```

6. **(옵션) 경로 지정 테이블에서 정보가 변경되지 않은 경우, 경로 지정 서비스를 다시 시작한 후 경로가 올바르게 다시 정의되었는지 경로 지정 테이블에서 정보를 확인합니다.**

   ```
   # svcadm restart routing-setup
   ```

예 6–1 IPMP에 대한 경로 정의

이 예에서는 설치 중에 `net0`에 대한 기본 경로가 정의되었다고 가정합니다.

```
# netstat -nr
Routing Table: IPv4
 Destination     Gateway       Flags    Ref     Use       Interface
```

```
-------------  ------------  --------  -----  -----------  --------
default        10.153.125.1     UG      107    176682262   net0
10.153.125.0   10.153.125.222   U        22    137738792   net0

# route -p delete default 10.153.125.1 -ifp net0
# route -p add default 10.153.125.1

# netstat -nr
Routing Table: IPv4
 Destination     Gateway       Flags    Ref     Use       Interface
-------------  ------------  --------  -----  -----------  --------
default        10.153.125.1     UG      107    176682262
10.153.125.0   10.153.125.222   U        22    137738792   net0
```

IPMP 그룹 구성

이 절에서는 IPMP 그룹 계획 및 구성을 위한 절차를 제공합니다. 5 장, “IPMP 소개”의 개요에는 IPMP 그룹을 인터페이스로 구현하는 방법이 설명되어 있습니다. 따라서 이 장에서 *IPMP* **그룹** 및 *IPMP* **인터페이스**란 용어는 같은 의미로 사용됩니다.

▼ IPMP 그룹을 계획하는 방법

다음 절차에는 IPMP 그룹을 구성하기 전에 필요한 계획 작업 및 수집할 정보가 포함되어 있습니다. 작업을 순서대로 수행할 필요는 없습니다.

주 – 각 서브넷 또는 L2 브로드캐스트 도메인에 대해 IPMP 그룹 한 개만 구성해야 합니다. 자세한 내용은 69 페이지 “IPMP 사용 규칙”을 참조하십시오.

---

1 요구에 맞는 일반 IPMP 구성을 결정합니다.

IPMP 구성은 네트워크에서 시스템에 호스트된 트래픽 유형을 처리하는 데 필요한 사항에 따라 달라집니다. IPMP는 아웃바운드 네트워크 패킷을 IPMP 그룹의 인터페이스에 분산시키므로 네트워크 처리량이 개선됩니다. 하지만 지정된 TCP 연결에 대해 인바운드 트래픽은 잘못된 순서로 패킷을 처리하는 위험을 최소화하기 위해 대체로 하나의 물리적 경로만 따릅니다.

따라서 네트워크에서 많은 아웃바운드 트래픽을 처리하는 경우 IPMP 그룹에 다수의 인터페이스를 구성하면 네트워크 성능이 향상될 수 있습니다. 대신 시스템에서 많은 인바운드 트래픽을 호스트하는 경우 그룹에 포함된 인터페이스 수가 많아도 반드시 트래픽 부하 분산에 의해 성능이 향상되는 것은 아닙니다. 하지만 기본 인터페이스가 많으면 인터페이스 실패 시 네트워크 가용성을 보장하는 데 도움이 됩니다.

2 **그룹의 각 인터페이스에 고유한 MAC 주소가 있는지 확인합니다.**

시스템의 각 인터페이스에 대해 고유한 MAC 주소를 구성하려면 **Oracle Solaris 11.1에서 고정된 네트워크 구성을 사용하여 시스템 연결**의 "각 인터페이스의 MAC 주소가 고유한지 확인하는 방법"을 참조하십시오.

3 **IPMP 그룹의 모든 인터페이스에서 동일한 STREAMS 모듈 세트가 구성되고 푸시되는지 확인하십시오.**

동일한 그룹의 모든 인터페이스에 동일한 STREAMS 모듈이 동일한 순서로 구성되어 있어야 합니다.

a. **잠재 IPMP 그룹의 모든 인터페이스에서 STREAMS 모듈의 순서를 확인합니다.**

`ifconfig` *interface* `modlist` 명령을 사용하여 STREAMS 모듈 목록을 인쇄할 수 있습니다. 예를 들어, `net0` 인터페이스의 `ifconfig` 출력은 다음과 같습니다.

```
# ifconfig net0 modlist
    0 arp
    1 ip
    2 e1000g
```

출력 결과와 같이 인터페이스는 대체로 IP 모듈 바로 아래에 네트워크 드라이브로 존재합니다. 이러한 인터페이스는 추가 구성이 필요하지 않습니다.

하지만 특정 기술은 IP 모듈과 네트워크 드라이버 간에 STREAMS 모듈로 푸시됩니다. STREAMS 모듈이 Stateful인 경우 그룹의 모든 인터페이스에 동일한 모듈을 푸시해도 페일오버 시 예기치 않은 동작이 발생할 수 있습니다. 하지만 IPMP 그룹의 모든 인터페이스에 모듈을 동일한 순서로 푸시하는 경우 Stateless STREAMS 모듈을 사용할 수 있습니다.

b. **IPMP 그룹의 표준 순서대로 각 인터페이스의 모듈을 푸시합니다.**

예를 들면 다음과 같습니다.

```
# ifconfig net0 modinsert vpnmod@3
```

4 **IPMP 그룹의 모든 인터페이스에서 동일한 IP 주소 형식을 사용합니다.**

IPv4에 대해 하나의 인터페이스가 구성된 경우 IPMP 그룹의 모든 인터페이스를 IPv4에 대해 구성해야 합니다. 예를 들어 IPv6 주소를 하나의 인터페이스에 추가하는 경우 IPMP 그룹의 모든 인터페이스가 IPv6 지원에 적합하게 구성되어야 합니다.

5 **구현하려는 실패 감지 유형을 결정합니다.**

예를 들어, 프로브 기반 실패 감지를 구현하려는 경우 기본 인터페이스에 테스트 주소를 구성해야 합니다. 관련 정보는 77 페이지 "IPMP의 실패 감지"를 참조하십시오.

6 **IPMP 그룹의 모든 인터페이스가 동일한 로컬 네트워크에 연결되어 있는지 확인합니다.**

예를 들어, 동일한 IP 서브넷의 이더넷 스위치를 IPMP 그룹으로 구성할 수 있습니다. 모든 개수의 인터페이스를 IPMP 그룹으로 구성할 수 있습니다.

주 – 예를 들어, 시스템에 물리적 인터페이스가 하나뿐인 경우 단일 인터페이스 IPMP 그룹을 구성할 수도 있습니다. 관련 정보는 70 페이지 "IPMP 인터페이스 구성 유형"을 참조하십시오.

7 **IPMP 그룹에 서로 다른 네트워크 매체 유형의 인터페이스가 포함되지 않도록 합니다.**

함께 그룹화된 인터페이스는 인터페이스 유형이 동일해야 합니다. 예를 들어, 이더넷 및 토큰 링 인터페이스를 IPMP 그룹으로 결합할 수 없습니다. 또 다른 예로 토큰 버스 인터페이스와 ATM(비동기식 전송 모드) 인터페이스를 동일한 IPMP 그룹에 결합할 수 없습니다.

8 **ATM 인터페이스가 있는 IPMP의 경우 LAN 에뮬레이션 모드로 ATM 인터페이스를 구성합니다.**

IETF RFC 1577 (`http://datatracker.ietf.org/doc/rfc1577/`) 및 IETF RFC 2225 (`http://datatracker.ietf.org/doc/rfc2225/`)에 정의된 ATM을 통한 기존 IP 기술을 사용하는 인터페이스에서는 IPMP가 지원되지 않습니다.

▼ DHCP를 사용하는 IPMP 그룹 구성 방법

활성-활성 인터페이스나 활성-대기 인터페이스를 사용하여 다중 인터페이스 IPMP 그룹을 구성할 수 있습니다. 관련 정보는 70 페이지 "IPMP 인터페이스 구성 유형"을 참조하십시오. 다음 절차에서는 DHCP를 사용하여 활성-대기 IPMP 그룹을 구성하는 방법에 대해 설명합니다.

시작하기 전에 잠재 IPMP 그룹에 포함될 IP 인터페이스가 시스템의 네트워크 데이터 링크에서 올바르게 구성되었는지 확인합니다. 링크 및 IP 인터페이스를 구성하는 절차는 **Oracle Solaris 11.1에서 고정된 네트워크 구성을 사용하여 시스템 연결**의 "IP 인터페이스를 구성하는 방법"을 참조하십시오. 기본 IP 인터페이스가 아직 생성되지 않은 경우에도 IPMP 인터페이스를 만들 수 있습니다. 그러나 기본 IP 인터페이스를 만들지 않는 경우 IPMP 인터페이스의 후속 구성이 실패합니다.

또한 SPARC 기반 시스템을 사용하는 경우 각 인터페이스에 고유한 MAC 주소를 구성합니다. 절차는 **Oracle Solaris 11.1에서 고정된 네트워크 구성을 사용하여 시스템 연결**의 "각 인터페이스의 MAC 주소가 고유한지 확인하는 방법"을 참조하십시오.

마지막으로, DHCP를 사용하는 경우 기본 인터페이스에 무기한 임대가 있는지 확인합니다. 그렇지 않으면 IPMP 그룹 실패 시 테스트 주소가 만료되고 `in.mpathd` 데몬이 프로브 기반 실패 감지를 사용 안함으로 설정하여 링크 기반 실패 감지가 사용됩니다. 링크 기반 실패 감지에서 인터페이스가 작동 중으로 검색되면 인터페이스가 복구되었다고 데몬이 잘못 보고할 수 있습니다. DHCP 구성에 대한 자세한 내용은 **Oracle Solaris 11.1의 DHCP 작업**을 참조하십시오.

1. **관리자로 전환합니다.**

   자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2. **IPMP 인터페이스를 만듭니다.**

   ```
   # ipadm create-ipmp ipmp-interface
   ```

   여기서 *ipmp-interface*는 IPMP 인터페이스의 이름을 지정합니다. IPMP 인터페이스에 의미 있는 이름을 지정할 수 있습니다. 모든 IP 인터페이스와 마찬가지로 이름은 문자열과 숫자로 구성됩니다(예: `ipmp0`).

3. **아직 없는 경우 기본 IP 인터페이스를 만듭니다.**

   ```
   # ipadm create-ip under-interface
   ```

   여기서 *under-interface*는 IPMP 그룹에 추가할 IP 인터페이스를 나타냅니다.

4. **테스트 주소가 포함될 기본 IP 인터페이스를 IPMP 그룹에 추가합니다.**

   ```
   # ipadm add-ipmp -i under-interface1 [-i under-interface2 ...] ipmp-interface
   ```

   시스템에서 사용 가능한 개수만큼 IP 인터페이스를 IPMP 그룹에 추가할 수 있습니다.

5. **DHCP가 IPMP 인터페이스의 데이터 주소를 구성하고 관리하게 합니다.**

   ```
   # ipadm create-addr -T dhcp ipmp-interface
   ```

   5단계에서는 DHCP 서버가 제공하는 주소를 주소 객체와 연결합니다. 주소 객체는 *interface*/*address-type* 형식을 사용하여 IP 주소를 고유하게 식별합니다(예: `ipmp0/v4`). 주소 객체에 대한 자세한 내용은 **Oracle Solaris 11.1에서 고정된 네트워크 구성을 사용하여 시스템 연결**의 "IP 인터페이스를 구성하는 방법"을 참조하십시오.

6. **테스트 주소와 함께 프로브 기반 실패 감지를 사용하는 경우 DHCP에서 기본 인터페이스의 테스트 주소를 관리하도록 합니다.**

   IPMP 그룹의 각 기본 인터페이스에 대해 다음 명령을 실행합니다.

   ```
   # ipadm create-addr -T dhcp under-interface
   ```

   6단계에서 자동으로 만든 주소 객체는 *under-interface*/*address-type* 형식을 사용합니다(예: `net0/v4`).

예 6–2 DHCP를 사용하여 IPMP 그룹 구성

이 예에서는 DHCP를 사용하여 활성-대기 IPMP 그룹을 구성하는 방법을 보여주며 다음 시나리오를 기반으로 합니다.

- 3가지 기본 인터페이스 `net0`, `net1` 및 `net2`가 IPMP 그룹으로 구성됩니다.
- IPMP 인터페이스 `ipmp0`은 IPMP 그룹과 동일한 이름을 공유합니다.
- `net2`는 지정된 대기 인터페이스입니다.
- 모든 기본 인터페이스에 테스트 주소가 지정됩니다.

먼저 관리자는 IPMP 인터페이스를 만듭니다.

```
# ipadm create-ipmp ipmp0
```

다음으로 관리자는 기본 IP 인터페이스를 만들고 이를 IPMP 인터페이스에 추가합니다.

```
# ipadm create-ip net0
# ipadm create-ip net1
# ipadm create-ip net2

# ipadm add-ipmp -i net0 -i net1 -i net2 ipmp0
```

다음으로 관리자는 DHCP 관리 IP 주소를 IPMP 인터페이스에 지정합니다. IPMP 인터페이스에 지정된 IP 주소는 데이터 주소입니다. 이 예에서는 IPMP 인터페이스에 2개의 데이터 주소가 있습니다.

```
# ipadm create-addr -T dhcp ipmp0
ipadm: ipmp0/v4
# ipadm create-addr -T dhcp ipmp0
ipadm: ipmp0/v4a
```

다음으로 관리자는 DHCP 관리 IP 주소를 IPMP 그룹의 기본 IP 인터페이스에 지정합니다. 기본 인터페이스에 지정된 IP 주소는 프로브 기반 실패 감지에 사용되는 테스트 주소입니다.

```
# ipadm create-addr -T dhcp net0
ipadm: net0/v4
# ipadm create-addr -T dhcp net1
ipadm: net1/v4
# ipadm create-addr -T dhcp net2
ipadm net2/v4
```

마지막으로 관리자는 net2가 대기 인터페이스가 되도록 구성합니다.

```
# ipadm set-ifprop -p standby=on net2
```

▼ 활성-활성 IPMP 그룹을 수동으로 구성하는 방법

다음 절차에서는 활성-활성 IPMP 그룹을 수동으로 구성하는 방법에 대해 설명합니다. 이 절차에서 1-4단계는 링크 기반의 활성-활성 IPMP 그룹을 구성하는 방법에 대해 설명합니다. 5단계에서는 링크 기반 구성을 프로브 기반으로 만드는 방법에 대해 설명합니다.

시작하기 전에 잠재 IPMP 그룹에 포함될 IP 인터페이스가 시스템의 네트워크 데이터 링크에서 올바르게 구성되었는지 확인합니다. 링크 및 IP 인터페이스를 구성하는 절차는 **Oracle Solaris 11.1에서 고정된 네트워크 구성을 사용하여 시스템 연결**의 "IP 인터페이스를 구성하는 방법"을 참조하십시오. 기본 IP 인터페이스가 없는 경우에도 IPMP 인터페이스를 만들 수 있습니다. 하지만 이 IPMP 인터페이스의 후속 구성이 실패합니다.

또한 SPARC 기반 시스템을 사용하는 경우 각 인터페이스에 고유한 MAC 주소를 구성합니다. 절차는 **Oracle Solaris 11.1에서 고정된 네트워크 구성을 사용하여 시스템 연결**의 "각 인터페이스의 MAC 주소가 고유한지 확인하는 방법"을 참조하십시오.

1 **관리자로 전환합니다.**

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 **IPMP 인터페이스를 만듭니다.**

```
# ipadm create-ipmp ipmp-interface
```

여기서 *ipmp-interface*는 IPMP 인터페이스의 이름을 지정합니다. IPMP 인터페이스에 의미 있는 이름을 지정할 수 있습니다. 모든 IP 인터페이스와 마찬가지로 이름은 문자열과 숫자로 구성됩니다(예: `ipmp0`).

3 **기본 IP 인터페이스를 그룹에 추가합니다.**

```
# ipadm add-ipmp -i under-interface1 [-i underinterface2 ...] ipmp-interface
```

여기서 *under-interface*는 IPMP 그룹의 기본 인터페이스를 나타냅니다. 시스템에서 사용 가능한 개수만큼 IP 인터페이스를 추가할 수 있습니다.

주 – 이중 스택 환경에서 인터페이스의 IPv4 인스턴스를 특정 그룹 아래에 배치하면 IPv6 인스턴스도 동일한 그룹 아래에 자동으로 배치됩니다.

---

4 **IPMP 인터페이스에 데이터 주소를 추가합니다.**

```
# ipadm create-addr -a address ipmp-interface
```

여기서 *address*는 CIDR 표기법을 사용할 수 있습니다.

주 – IPMP 그룹 이름의 DNS 주소 또는 IP 주소만 필요합니다.

---

5 **테스트 주소와 함께 프로브 기반 실패 감지를 사용하는 경우 기본 인터페이스에서 테스트 주소를 추가합니다.**

```
# ipadm create-addr -a address under-interface
```

여기서 *address*는 CIDR 표기법을 사용할 수 있습니다. IPMP 그룹의 모든 테스트 IP 주소는 단일 IP 서브넷에 속해야 하므로 동일한 네트워크 접두어를 사용합니다.

▼ 활성-대기 IPMP 그룹을 수동으로 구성하는 방법

대기 인터페이스에 대한 내용은 70 페이지 "IPMP 인터페이스 구성 유형"을 참조하십시오. 다음 절차에서는 인터페이스 한 개가 대기 상태로 유지되는 IPMP 그룹을 구성하는 방법에 대해 설명합니다. 이 인터페이스는 그룹의 활성 인터페이스가 실패하는 경우에만 배포됩니다.

1 **관리자로 전환합니다.**

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 **IPMP 인터페이스를 만듭니다.**

```
# ipadm create-ipmp ipmp-interface
```

여기서 *ipmp-interface*는 IPMP 인터페이스의 이름을 지정합니다. IPMP 인터페이스에 의미 있는 이름을 지정할 수 있습니다. 모든 IP 인터페이스와 마찬가지로 이름은 문자열과 숫자로 구성됩니다(예: `ipmp0`).

3 **기본 IP 인터페이스를 그룹에 추가합니다.**

```
# ipadm add-ipmp -i under-interface1 [-i underinterface2 ...] ipmp-interface
```

여기서 *under-interface*는 IPMP 그룹의 기본 인터페이스를 나타냅니다. 시스템에서 사용 가능한 개수만큼 IP 인터페이스를 추가할 수 있습니다.

주 – 이중 스택 환경에서 인터페이스의 IPv4 인스턴스를 특정 IPMP 그룹 아래에 배치하면 IPv6 인스턴스도 동일한 그룹 아래에 자동으로 배치됩니다.

---

4 **IPMP 인터페이스에 데이터 주소를 추가합니다.**

```
# ipadm create-addr -a address ipmp-interface
```

여기서 *address*는 CIDR 표기법을 사용할 수 있습니다.

5 **테스트 주소와 함께 프로브 기반 실패 감지를 사용하는 경우 기본 인터페이스에서 테스트 주소를 추가합니다.**

```
# ipadm create-addr -a address under-interface
```

여기서 *address*는 CIDR 표기법을 사용할 수 있습니다. IPMP 그룹의 모든 테스트 IP 주소는 단일 IP 서브넷에 속해야 하므로 동일한 네트워크 접두어를 사용합니다.

6 **기본 인터페이스 중 하나를 대기 인터페이스로 구성합니다.**

```
# ipadm set-ifprop -p standby=on under-interface
```

예 6–3 활성-대기 IPMP 그룹 구성

이 예에서는 활성-대기 IPMP 구성을 수동으로 만드는 방법을 보여줍니다.

먼저 관리자는 IPMP 인터페이스를 만듭니다.

```
# ipadm create-ipmp ipmp0
```

다음으로 관리자는 기본 IP 인터페이스를 만들고 이를 IPMP 인터페이스에 추가합니다.

```
# ipadm create-ip net0
# ipadm create-ip net1
# ipadm create-ip net2

# ipadm add-ipmp -i net0 -i net1 -i net2 ipmp0
```

다음으로 관리자는 IP 주소를 IPMP 인터페이스에 지정합니다. IPMP 인터페이스에 지정된 IP 주소는 데이터 주소입니다. 이 예에서는 IPMP 인터페이스에 2개의 데이터 주소가 있습니다.

```
# ipadm create-addr -a 192.168.10.10/24 ipmp0
ipadm: ipmp0/v4
# ipadm create-addr -a 192.168.10.15/24 ipmp0
ipadm: ipmp0/v4a
```

이 예에서 IP 주소에는 10진수 숫자로 표시되는 `prefixlen` 등록 정보가 포함됩니다. IP 주소에서 `prefixlen` 부분은 주소 중 IPv4 넷마스크 또는 IPv6 접두어를 구성하는 가장 왼쪽의 연속된 숫자를 지정합니다. 남은 하위 비트 숫자들은 주소의 호스트 부분을 정의합니다. `prefixlen` 등록 정보가 주소의 텍스트 표현으로 변환되면 네트워크 부분에 사용되는 비트 위치를 나타내는 1과 호스트 파트에 대한 0이 주소에 포함됩니다. 이 등록 정보는 `dhcp` 주소 객체 유형에서 지원되지 않습니다. 자세한 내용은 `ipadm(1M)` 매뉴얼 페이지를 참조하십시오.

다음으로 관리자는 IP 주소를 IPMP 그룹의 기본 IP 인터페이스에 지정합니다. 기본 인터페이스에 지정된 IP 주소는 프로브 기반 실패 감지에 사용되는 테스트 주소입니다.

```
# ipadm create-addr -a 192.168.10.30/24 net0
ipadm: net0/v4
# ipadm create-addr -a 192.168.10.32/24 net1
ipadm: net1/v4
# ipadm create-addr -a 192.168.10.34/24 net2
ipadm: net2/v4
```

마지막으로 관리자는 `net2`가 대기 인터페이스가 되도록 구성합니다.

```
# ipadm set-ifprop -p standby=on net2
```

관리자는 `ipmpstat` 명령을 사용하여 IPMP 구성을 볼 수 있습니다.

```
# ipmpstat -g
GROUP     GROUPNAME   STATE      FDT        INTERFACES
ipmp0     ipmp0       ok         10.00s     net0 net1 (net2)

# ipmpstat -t
INTERFACE  MODE     TESTADDR        TARGETS
net0       routes   192.168.10.30   192.168.10.1
net1       routes   192.168.10.32   192.168.10.1
net2       routes   192.168.10.34   192.168.10.5
```

IPMP 유지 관리

이 절에는 시스템에서 만든 IPMP 그룹을 유지 관리하기 위한 절차가 들어 있습니다.

▼ IPMP 그룹에 인터페이스를 추가하는 방법

시작하기 전에 그룹에 추가하는 인터페이스가 모든 요구 사항을 충족시키는지 확인합니다. 요구 사항 목록은 85 페이지 “IPMP 그룹을 계획하는 방법”을 참조하십시오.

1. **관리자로 전환합니다.**

   자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 “지정된 관리 권한을 사용하는 방법”을 참조하십시오.

2. **기본 IP 인터페이스가 없는 경우 인터페이스를 만듭니다.**

   ```
   # ipadm create-ip under-interface
   ```

3. **IPMP 그룹에 IP 인터페이스를 추가합니다.**

   ```
   # ipadm add-ipmp -i under-interface ipmp-interface
   ```

   여기서 *ipmp-interface*는 기본 인터페이스를 추가하려는 IPMP 그룹을 가리킵니다.

예 6–4 IPMP 그룹에 인터페이스 추가

다음 예에서는 인터페이스 `net4`를 IPMP 그룹 `ipmp0`에 추가합니다.

```
# ipadm create-ip net4
# ipadm add-ipmp -i net4 ipmp0
# ipmpstat -g
GROUP    GROUPNAME   STATE      FDT       INTERFACES
ipmp0    ipmp0       ok         10.00s    net0 net1 net4
```

▼ IPMP 그룹에서 인터페이스를 제거하는 방법

1. **관리자로 전환합니다.**

   자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 “지정된 관리 권한을 사용하는 방법”을 참조하십시오.

2 **IPMP 그룹에서 인터페이스를 하나 이상 제거합니다.**

```
# ipadm remove-ipmp -i under-interface[ -i under-interface ...] ipmp-interface
```

여기서 *under-interface*는 IPMP 그룹에서 제거할 IP 인터페이스를 가리키고 *ipmp-interface*는 기본 인터페이스를 제거할 IPMP 그룹을 가리킵니다.

단일 명령으로 기본 인터페이스를 필요한 개수만큼 제거할 수 있습니다. 기본 인터페이스를 모두 제거해도 IPMP 인터페이스가 삭제되지는 않습니다. 대신 빈 IPMP 인터페이스 또는 그룹으로 존재합니다.

예 6–5 IPMP 그룹에서 인터페이스 제거

다음 예에서는 인터페이스 `net4`를 IPMP 그룹 `ipmp0`에서 제거합니다.

```
# ipadm remove-ipmp net4 ipmp0
# ipmpstat -g
GROUP   GROUPNAME   STATE      FDT       INTERFACES
ipmp0   ipmp0       ok         10.00s    net0 net1
```

▼ IP 주소 추가 방법

IP 주소를 추가하려면 `ipadm create-addr` 하위 명령을 사용합니다. IPMP에서는 IP 주소가 데이터 주소일 수도 있고 테스트 주소일 수도 있습니다. 데이터 주소는 IPMP 인터페이스에 추가됩니다. 테스트 주소는 IPMP 인터페이스의 기본 인터페이스에 추가됩니다. 다음 절차에서는 테스트 주소 또는 데이터 주소인 IP 주소를 추가하는 방법에 대해 설명합니다.

1 **관리자로 전환합니다.**

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 **IPMP 그룹에 데이터 주소를 추가하려면 다음 명령을 입력합니다.**

```
# ipadm create-addr -a address ipmp-interface
```

방금 만든 IP 주소에 주소 객체가 자동으로 지정됩니다. 주소 객체는 IP 주소의 고유 식별자입니다. 주소 객체 이름에는 이름 지정 규약 *interface/ random-string*이 사용됩니다. 따라서 데이터 주소의 주소 객체는 해당 이름에 IPMP 인터페이스가 포함됩니다.

3 **IPMP 그룹의 기본 인터페이스에 테스트 주소를 추가하려면 다음 명령을 입력합니다.**

```
# ipadm create-addr -a address under-interface
```

방금 만든 IP 주소에 주소 객체가 자동으로 지정됩니다. 주소 객체는 IP 주소의 고유 식별자입니다. 주소 객체 이름에는 이름 지정 규약 *interface/ random-string*이 사용됩니다. 따라서 테스트 주소의 주소 객체는 해당 이름에 기본 인터페이스가 포함됩니다.

▼ IP 주소 삭제 방법

IP 주소를 제거하려면 `ipadm delete-addr` 하위 명령을 사용합니다. IPMP에서 데이터 주소는 IPMP 인터페이스에서 호스트하고 테스트 주소는 기본 인터페이스에서 호스트합니다. 다음 절차에서는 데이터 주소 또는 테스트 주소인 IP 주소를 제거하는 방법을 보여줍니다.

1 관리자로 전환합니다.

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 제거할 주소를 결정합니다.

- 데이터 주소 목록을 표시하려면 다음 명령을 입력합니다.

  ```
  # ipadm show-addr ipmp-interface
  ```

- 테스트 주소 목록을 표시하려면 다음 명령을 입력합니다.

  ```
  # ipadm show-addr
  ```

  테스트 주소는 주소가 구성된 기본 인터페이스가 해당 이름에 포함되어 있는 주소 객체에 의해 식별됩니다.

3 IPMP 그룹에서 데이터 주소를 제거하려면 다음 명령을 입력합니다.

```
# ipadm delete-addr addrobj
```

여기서 *addrobj*에 IPMP 인터페이스 이름이 포함되어 있어야 합니다. 입력한 주소 객체에 IPMP 인터페이스 이름이 포함되어 있지 않으면 삭제될 주소가 데이터 주소가 아닙니다.

4 IPMP 그룹에서 테스트 주소를 제거하려면 다음 명령을 입력합니다.

```
# ipadm delete-addr addrobj
```

여기서 올바른 테스트 주소를 삭제하려면 *addrobj*에 올바른 기본 인터페이스 이름이 포함되어 있어야 합니다.

예 6–6 인터페이스에서 테스트 주소 제거

다음 예에서는 활성-대기 IPMP 그룹 `ipmp0`의 구성을 예 6–3에서 사용합니다. 이 예에서는 테스트 주소를 기본 인터페이스 `net1`에서 제거합니다.

```
# ipadm show-addr net1
ADDROBJ           TYPE     STATE     ADDR
net1/v4           static   ok        192.168.10.30

# ipadm delete-addr net1/v4
```

▼ 하나의 IPMP 그룹에서 다른 IPMP 그룹으로 인터페이스를 이동하는 방법

인터페이스가 기존 IPMP 그룹에 속하는 경우 새 IPMP 그룹에 인터페이스를 배치할 수 있습니다. 현재 IPMP 그룹에서 인터페이스를 제거할 필요는 없습니다. 새 그룹에 인터페이스를 배치하면 기존 IPMP 그룹에서 해당 인터페이스가 자동으로 제거됩니다.

1 **관리자로 전환합니다.**

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 **인터페이스를 새 IPMP 그룹으로 이동합니다.**

```
# ipadm add-ipmp -i under-interface ipmp-interface
```

여기서 *under-interface*는 이동할 기본 인터페이스를 가리키고 *ipmp-interface*는 기본 인터페이스를 이동할 IPMP 인터페이스를 가리킵니다.

예 6–7 다른 IPMP 그룹으로 인터페이스 이동

이 예에서는 IPMP 그룹의 기본 인터페이스가 `net0`, `net1` 및 `net2`입니다. 다음 명령은 `net0` 인터페이스를 IPMP 그룹 `ipmp0`에서 제거한 후 `net0`을 IPMP 그룹 `cs-link1`에 지정합니다.

```
# ipadm add-ipmp -i net0 ca-link1
```

▼ IPMP 그룹을 삭제하는 방법

특정 IPMP 그룹이 더 이상 필요하지 않은 경우 이 절차를 사용합니다.

1 **관리자로 전환합니다.**

자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2 **삭제할 IPMP 그룹 및 기본 IP 인터페이스를 식별합니다.**

```
# ipmpstat -g
```

3 **현재 IPMP 그룹에 속하는 IP 인터페이스를 모두 제거합니다.**

```
# ipadm remove-ipmp -i under-interface[, -i under-interface, ...] ipmp-interface
```

여기서 *under-interface*는 제거할 기본 인터페이스를 가리키고 *ipmp-interface*는 기본 인터페이스를 제거할 IPMP 인터페이스를 가리킵니다.

주 – IPMP 인터페이스를 성공적으로 삭제하려면 IPMP 그룹에 속한 IP 인터페이스가 없어야 합니다.

4 **IPMP 인터페이스를 삭제합니다.**

```
# ipadm delete-ipmp ipmp-interface
```

IPMP 인터페이스를 삭제하면 해당 인터페이스와 연결된 모든 IP 주소가 시스템에서 삭제됩니다.

예 6–8 IPMP 인터페이스 삭제

다음 예에서는 기본 IP 인터페이스가 `net0` 및 `net1`인 인터페이스 `ipmp0`을 삭제합니다.

```
# ipmpstat -g
GROUP    GROUPNAME   STATE      FDT        INTERFACES
ipmp0    ipmp0       ok         10.00s     net0 net1

# ipadm remove-ipmp -i net0 -i net1 ipmp0

# ipadm delete-ipmp ipmp0
```

프로브 기반 실패 감지 구성

프로브 기반 실패 감지의 경우 78 페이지 “프로브 기반 실패 감지”에 설명된 대로 대상 시스템을 사용해야 합니다. 프로브 기반 실패 감지의 대상을 식별할 때 `in.mpathd` 데몬은 라우터 대상 모드나 멀티캐스트 대상 모드의 두 가지 모드로 작동합니다. 라우터 대상 모드에서는 데몬이 경로 지정 테이블에 정의된 대상을 프로빙합니다. 대상을 정의하지 않은 경우 데몬이 멀티캐스트 대상 모드로 작동합니다. 이 모드에서는 멀티캐스트 패킷이 전송되어 LAN에서 인접한 호스트를 프로빙합니다.

프로빙할 `in.mpathd` 데몬에 대해 대상 시스템을 설정해야 할 수도 있습니다. 일부 IPMP 그룹의 경우 기본 라우터를 대상으로 사용해도 됩니다. 하지만 프로브 기반 실패 감지에 대해 특정 대상을 구성해야 하는 IPMP 그룹도 있습니다. 대상을 지정하려면 경로 지정 테이블의 호스트 경로를 프로브 대상으로 설정합니다. 경로 지정 테이블에 구성된 호스트 경로는 기본 라우터 앞에 나열됩니다. IPMP는 명시적으로 정의된 호스트 경로를 대상 선택으로 사용합니다. 따라서 기본 라우터를 사용하는 대신 호스트 경로를 설정하여 특정 프로브 대상을 구성해야 합니다.

경로 지정 테이블에 호스트 경로를 설정하려면 `route` 명령을 사용합니다. 이 명령과 함께 `-p` 옵션을 사용하여 지속 경로를 추가할 수 있습니다. 예를 들어, `route -p add`는 시스템을 재부트한 후에도 경로 지정 테이블에 유지될 경로를 추가합니다. 따라서 `-p` 옵션을 사용하면 각 시스템을 시작할 때마다 경로를 다시 만드는 특수 스크립트 없이도 지속 경로를 추가할 수 있습니다. 프로브 기반 실패 감지를 최적으로 사용하려면 프로브를 받을 대상을 여러 개 설정해야 합니다.

99 페이지 “프로브 기반 실패 감지의 대상 시스템을 수동으로 지정하는 방법”
절차에서는 프로브 기반 실패 감지를 위해 대상에 지속 경로를 추가하는 정확한 구문을 보여줍니다. route 명령의 옵션에 대한 자세한 내용은 route(1M) 매뉴얼 페이지를 참조하십시오.

이 절에서는 다음 내용을 다룹니다.

프로브 기반 실패 감지를 위한 대상 선택 요구 사항

적합한 대상이 될 수 있는 네트워크의 호스트를 평가하는 경우 다음 기준 목록을 따르십시오.

- 잠재 대상이 사용 가능하고 실행되고 있는지 확인합니다. 해당 IP 주소 목록을 만듭니다.
- 대상 인터페이스가 구성 중인 IPMP 그룹과 동일한 네트워크에 있는지 확인합니다.
- 대상 시스템의 넷마스크 및 브로드캐스트 주소가 IPMP 그룹의 주소와 동일해야 합니다.
- 대상 호스트가 프로브 기반 실패 감지를 사용하는 인터페이스의 ICMP 요청에 대답할 수 있어야 합니다.

프로브 기반 실패 감지 구성(작업 맵)

다음 작업 맵에는 IPMP 그룹의 프로브 기반 실패 감지를 구성하는 절차가 나열되어 있습니다.

작업	설명	수행 방법
프로브 기반 실패 감지 작동 방식을 선택합니다.	프로브 기반 실패 감지에서 추이적인 프로브를 사용해야 하는지 테스트 주소를 사용해야 하는지 결정합니다.	99 페이지 “사용할 실패 감지 방법을 선택하는 방법”
프로브 기반 실패 감지에 사용될 대상 시스템을 선택합니다.	프로브 기반 실패 감지에서 IPMP 그룹의 기본 인터페이스 상태 테스트에 사용할 시스템의 IP 주소를 지정합니다.	99 페이지 “프로브 기반 실패 감지의 대상 시스템을 수동으로 지정하는 방법”
복구된 인터페이스가 IPMP 그룹에서 재배치되는 방식을 선택합니다.	복구된 IP 인터페이스가 IPMP 그룹에서 자동으로 다시 활성화되어야 하는지 비활성화된 상태로 남아 있어야 하는지 여부를 결정합니다.	100 페이지 “IPMP 데몬의 동작을 구성하는 방법”

▼ 사용할 실패 감지 방법을 선택하는 방법

기본적으로 프로브 기반 실패 감지는 테스트 주소를 사용하여 작동합니다. NIC 드라이버가 지원하는 경우 링크 기반 실패 감지도 자동으로 사용으로 설정됩니다.

이 방법이 NIC 드라이버에서 지원되는 경우 링크 기반 실패 감지를 사용 안함으로 설정할 수 없습니다. 하지만 구현할 프로브 기반 실패 감지 유형을 선택할 수 있습니다.

시작하기 전에 프로브 대상이 98 페이지 "프로브 기반 실패 감지를 위한 대상 선택 요구 사항"에 나열된 요구 사항을 충족하는지 확인합니다.

1 전이적 프로브만 사용하려면 다음 단계를 수행합니다.

a. 적절한 SMF 명령을 사용하여 IPMP 등록 정보 transitive-probing을 설정합니다.

```
# svccfg -s svc:/network/ipmp setprop config/transitive-probing=true
# svcadm refresh  svc:/network/ipmp:default
```

이 등록 정보 설정에 대한 자세한 내용은 `in.mpathd(1M)` 매뉴얼 페이지를 참조하십시오.

b. IPMP 그룹에 대해 구성된 기존 테스트 주소를 모두 제거합니다.

```
# ipadm delete-addr address addrobj
```

여기서 *addrobj*는 테스트 주소를 호스트하는 기본 인터페이스를 참조해야 합니다.

2 테스트 주소를 사용하여 오류를 프로빙하려면 다음 단계를 수행합니다.

a. 필요한 경우 전이적 프로브를 해제합니다.

```
# svccfg -s svc:/network/ipmp setprop config/transitive-probing=false
# svcadm refresh  svc:/network/ipmp:default
```

b. IPMP 그룹의 기본 인터페이스에 테스트 주소를 지정합니다.

```
# ipadm create-addr -a address under-interface
```

여기서 *address*에는 CIDR 표기법을 사용할 수 있으며 *under-interface*는 IPMP 그룹의 기본 인터페이스입니다.

▼ 프로브 기반 실패 감지의 대상 시스템을 수동으로 지정하는 방법

이 절차에서는 테스트 주소를 사용하여 프로브 기반 실패 감지를 구현하는 경우 특정 대상을 추가하는 방법에 대해 설명합니다.

시작하기 전에 프로브 대상이 98 페이지 "프로브 기반 실패 감지를 위한 대상 선택 요구 사항"에 나열된 요구 사항을 충족하는지 확인합니다.

1. **프로브 기반 실패 감지를 구성 중인 시스템에 사용자 계정으로 로그인합니다.**

2. **프로브 기반 실패 감지에서 대상으로 사용할 특정 호스트에 경로를 추가합니다.**

   ```
   $ route -p add -host destination-IP gateway-IP -static
   ```

   여기서 *destination-IP* 및 *gateway-IP*는 대상으로 사용할 호스트의 IPv4 주소입니다. 예를 들어, IPMP 그룹 `ipmp0`의 인터페이스와 동일한 서브넷에 있는 대상 시스템 `192.168.10.137`을 지정하려면 다음을 입력합니다.

   ```
   $ route -p add -host 192.168.10.137 192.168.10.137 -static
   ```

   이 새로운 경로는 시스템이 다시 시작될 때마다 자동으로 구성됩니다. 프로브 기반 실패 감지의 대상 시스템에 대해 임시 경로만 정의하려는 경우 `-p` 옵션을 사용하지 마십시오.

3. **네트워크에서 대상 시스템으로 사용할 추가 호스트에 경로를 추가합니다.**

▼ IPMP 데몬의 동작을 구성하는 방법

IPMP 그룹에 대해 시스템 차원의 다음 매개변수를 구성하려면 IPMP 구성 파일 `/etc/default/mpathd`를 사용합니다.

- `FAILURE_DETECTION_TIME`
- `FAILBACK`
- `TRACK_INTERFACES_ONLY_WITH_GROUPS`

1. **관리자로 전환합니다.**

   자세한 내용은 **Oracle Solaris 11.1 관리: 보안 서비스**의 "지정된 관리 권한을 사용하는 방법"을 참조하십시오.

2. **`/etc/default/mpathd` 파일을 편집합니다.**

   세 매개변수 중 하나 이상의 기본값을 변경합니다.

   a. **`FAILURE_DETECTION_TIME` 매개변수의 새 값을 입력합니다.**

      ```
      FAILURE_DETECTION_TIME=n
      ```

      여기서 *n*은 ICMP 프로브에서 인터페이스 실패가 발생했는지 여부를 감지하는 데 걸리는 시간(초)입니다. 기본값은 10초입니다.

   b. **`FAILBACK` 매개변수의 새 값을 입력합니다.**

      ```
      FAILBACK=[yes | no]
      ```

      - yes – yes 값은 IPMP 페일백 동작의 기본값입니다. 실패한 인터페이스의 복구가 감지되면 네트워크 액세스가 80 페이지 "물리적 인터페이스 복구 감지"에 설명된 대로 복구된 인터페이스로 페일백됩니다.

- no – no 값은 데이터 트래픽이 복구된 인터페이스로 돌아가지 않음을 나타냅니다. 실패한 인터페이스가 복구된 것으로 감지되면 INACTIVE 플래그가 해당 인터페이스에 설정됩니다. 이 플래그는 인터페이스가 현재 데이터 트래픽에 사용되지 않음을 나타냅니다. 프로브 트래픽에는 계속 인터페이스를 사용할 수 있습니다.

  예를 들어, IPMP 그룹 ipmp0은 두 인터페이스 net0과 net1로 구성되어 있다고 가정합니다. /etc/default/mpathd 파일에서 FAILBACK=no 매개변수가 설정되었습니다. net0이 실패하면 FAILED 플래그가 지정되고 사용할 수 없게 됩니다. 복구 후에는 인터페이스에 INACTIVE 플래그가 지정되며 FAILBACK=no 값으로 인해 인터페이스가 사용할 수 없는 상태로 유지됩니다.

  net1은 실패하고 net0만 INACTIVE 상태인 경우 net0의 INACTIVE 플래그가 지워지고 인터페이스가 사용 가능하게 됩니다. IPMP 그룹에도 INACTIVE 상태의 다른 인터페이스가 있는 경우 net1이 실패하면 이러한 INACTIVE 인터페이스 중 하나(net0일 필요는 없음)가 지워지고 사용할 수 있게 됩니다.

c. TRACK_INTERFACES_ONLY_WITH_GROUPS 매개변수의 새 값을 입력합니다.

```
TRACK_INTERFACES_ONLY_WITH_GROUPS=[yes | no]
```

주 – 이 매개변수와 익명 그룹 기능에 대한 자세한 내용은 80 페이지 “실패 감지 및 익명 그룹 기능”을 참조하십시오.

- yes – yes 값은 IPMP 동작의 기본값입니다. 이 값을 사용하면 IPMP가 IPMP 그룹으로 구성되지 않은 네트워크 인터페이스를 무시합니다.
- no – no 값은 IPMP 그룹으로 구성되었는지 여부에 관계없이 **모든** 네트워크 인터페이스에 대해 실패 및 복구 감지를 설정합니다. 하지만 IPMP 그룹으로 구성되지 않은 인터페이스에서 실패 또는 복구가 감지될 경우 해당 인터페이스의 네트워킹 기능을 유지 관리하기 위해 IPMP에서 아무 작업도 트리거되지 않습니다. 따라서 no 값은 오류 보고에만 유용하며 직접적으로 네트워크 가용성을 향상시키지는 않습니다.

3 in.mpathd 데몬을 다시 시작합니다.

```
# pkill -HUP in.mpathd
```

새 매개변수 값이 적용되도록 데몬이 다시 시작됩니다.

IPMP 정보 모니터링

이 절의 예에서는 ipmpstat 명령을 사용하여 시스템에서 IPMP 그룹의 다른 측면을 모니터링할 수 있도록 설정합니다. IPMP 그룹 전체나 해당 기본 IP 인터페이스의 상태를 관찰할 수 있습니다. 또한 IPMP 그룹에 대한 데이터 및 테스트 주소 구성을 확인할 수 있습니다. ipmpstat 명령을 사용하여 실패 감지에 대한 정보를 가져올 수도 있습니다. ipmpstat 명령 및 해당 옵션에 대한 자세한 내용은 ipmpstat(1M) 매뉴얼 페이지를 참조하십시오.

ipmpstat 명령을 사용하는 경우 기본적으로 80개 열에 들어가는 가장 의미 있는 필드가 표시됩니다. ipmpstat -p 구문의 경우를 제외하고 ipmpstat 명령과 함께 사용하는 옵션과 관련된 모든 필드가 출력에 표시됩니다.

기본적으로 호스트 이름이 있을 경우 숫자 IP 주소 대신 호스트 이름이 출력에 표시됩니다. 출력에 숫자 IP 주소를 나열하려면 -n 옵션을 다른 옵션과 함께 사용하여 특정 IPMP 그룹 정보를 표시합니다.

주 – 달리 명시되지 않은 경우 다음 예에서는 ipmpstat 명령을 사용하는 데 시스템 관리자 권한이 필요하지 않습니다.

다음 옵션을 ipmpstat 명령과 함께 사용하여 표시할 정보를 결정합니다.

- -g는 시스템의 IPMP 그룹에 대한 정보를 표시합니다. 예 6–9를 참조하십시오.
- -a는 IPMP 그룹에 대해 구성된 데이터 주소를 표시합니다. 예 6–10을 참조하십시오.
- -i는 IPMP 구성과 관련된 IP 인터페이스에 대한 정보를 표시합니다. 예 6–11을 참조하십시오.
- -t는 실패 감지에 사용되는 대상 시스템에 대한 정보를 표시합니다. 이 옵션은 IPMP 그룹에서 사용되는 테스트 주소도 표시합니다. 예 6–12를 참조하십시오.
- -p는 실패 감지에 사용되는 프로브에 대한 정보를 표시합니다. 예 6–13을 참조하십시오.

다음 예는 ipmpstat 명령을 사용하여 가져올 수 있는 시스템의 IPMP 구성에 대한 정보를 표시합니다.

예 6–9 IPMP 그룹 정보 가져오기

-g 옵션은 기본 인터페이스 상태를 비롯하여 시스템에 있는 여러 IPMP 그룹의 상태를 나열합니다. 특정 그룹에 대해 프로브 기반 실패 감지가 사용으로 설정된 경우 이 명령에 해당 그룹에 대한 실패 감지 시간도 포함됩니다.

```
$ ipmpstat -g
GROUP    GROUPNAME   STATE      FDT        INTERFACES
ipmp0    ipmp0       ok         10.00s     net0 net1
acctg1   acctg1      failed     --         [net3 net4]
field2   field2      degraded   20.00s     net2 net5 (net7) [net6]
```

출력 필드는 다음 정보를 제공합니다.

GROUP — IPMP 인터페이스 이름을 지정합니다. 익명 그룹의 경우 이 필드가 비어 있습니다. 익명 그룹에 대한 자세한 내용은 in.mpathd(1M) 매뉴얼 페이지를 참조하십시오.

GROUPNAME — IPMP 그룹의 이름을 지정합니다. 익명 그룹의 경우 이 필드가 비어 있습니다.

STATE — IPMP 그룹의 현재 상태를 나타내며 다음 중 하나일 수 있습니다.

예 6–9 IPMP 그룹 정보 가져오기 *(계속)*

- `ok`는 IPMP 그룹의 모든 기본 인터페이스를 사용할 수 있음을 나타냅니다.
- `degraded`는 그룹의 일부 기본 인터페이스를 사용할 수 없음을 나타냅니다.
- `failed`는 그룹의 모든 인터페이스를 사용할 수 없음을 나타냅니다.

`FDT` 실패 감지가 사용으로 설정된 경우 실패 감지 시간을 지정합니다. 실패 감지가 사용 안함으로 설정된 경우 이 필드가 비어 있습니다.

`INTERFACES` IPMP 그룹에 속하는 기본 인터페이스를 지정합니다. 이 필드에는 활성 인터페이스, 비활성 인터페이스 및 사용할 수 없는 인터페이스가 차례로 나열됩니다. 인터페이스 상태는 나열된 방식으로 표시됩니다.

- *interface*(괄호 또는 대괄호 없음)는 활성 인터페이스를 나타냅니다. 활성 인터페이스는 시스템에서 데이터 트래픽을 보내거나 받는 데 사용합니다.
- *(interface)*(괄호 있음)는 작동하지만 비활성인 인터페이스를 나타냅니다. 인터페이스가 관리 정책에 정의된 대로 사용되고 있지 않습니다.
- *[interface]*(대괄호 있음)는 인터페이스가 실패했거나 오프라인 상태이므로 인터페이스를 사용할 수 없음을 나타냅니다.

예 6–10 IPMP 데이터 주소 정보 가져오기

`-a` 옵션은 데이터 주소 및 각 주소가 속한 IPMP 그룹을 표시합니다. 표시되는 정보에는 `ipadm [up-addr/down-addr]` 명령으로 주소가 토글되었는지 여부에 따라 사용할 수 있는 주소도 포함됩니다. 주소를 사용할 수 있는 인바운드 또는 아웃바운드 인터페이스를 결정할 수도 있습니다.

```
$ ipmpstat -an
ADDRESS         STATE    GROUP      INBOUND     OUTBOUND
192.168.10.10   up       ipmp0        net0    net0 net1
192.168.10.15   up       ipmp0        net1    net0 net1
192.0.0.100     up       acctg1       --         --
192.0.0.101     up       acctg1       --         --
128.0.0.100     up       field2       net2    net2 net7
128.0.0.101     up       field2       net7    net2 net7
128.0.0.102     down     field2       --         --
```

출력 필드는 다음 정보를 제공합니다.

`ADDRESS` `-n` 옵션을 `-a` 옵션과 함께 사용하는 경우 호스트 이름 또는 데이터 주소를 지정합니다.

`STATE` IPMP 인터페이스의 주소가 `up`(사용 가능) 또는 `down`(사용 불가능) 상태인지를 나타냅니다.

예 6–10 IPMP 데이터 주소 정보 가져오기 *(계속)*

GROUP 특정 데이터 주소를 호스트하는 IPMP 인터페이스를 지정합니다. 일반적으로 Oracle Solaris에서는 IPMP 그룹 이름이 IPMP 인터페이스입니다.

INBOUND 지정된 주소에 대한 패킷을 받는 인터페이스를 식별합니다. 외부 이벤트에 따라 필드 정보가 변경될 수도 있습니다. 예를 들어, 데이터 주소가 작동 중지되었거나 IPMP 그룹에 활성 IP 인터페이스가 남아 있지 않은 경우 이 필드가 비어 있습니다. 빈 필드는 시스템이 지정된 주소로 전송된 IP 패킷을 허용하지 않음을 나타냅니다.

OUTBOUND 지정된 주소를 소스 주소로 사용하는 패킷을 보내는 인터페이스를 식별합니다. INBOUND 필드와 마찬가지로 OUTBOUND 정보도 외부 이벤트에 따라 변경될 수 있습니다. 빈 필드는 시스템이 지정된 소스 주소로 패킷을 보내지 않음을 나타냅니다. 필드가 비어 있는 것은 주소가 작동 중지되었거나 그룹에 활성 IP 인터페이스가 남아 있지 않기 때문일 수 있습니다.

예 6–11 IPMP 그룹의 기본 IP 인터페이스에 대한 정보 가져오기

-i 옵션은 IPMP 그룹의 기본 IP 인터페이스에 대한 정보를 표시합니다.

```
$ ipmpstat -i
INTERFACE   ACTIVE   GROUP      FLAGS      LINK       PROBE      STATE
net0        yes      ipmp0      --mb---    up         ok         ok
net1        yes      ipmp0      -------    up         disabled   ok
net3        no       acctg1     -------    unknown    disabled   offline
net4        no       acctg1     is-----    down       unknown    failed
net2        yes      field2     --mb---    unknown    ok         ok
net6        no       field2     -i-----    up         ok         ok
net5        no       filed2     -------    up         failed     failed
net7        yes      field2     --mb---    up         ok         ok
```

출력 필드는 다음 정보를 제공합니다.

INTERFACE 각 IPMP 그룹의 각 기본 인터페이스를 지정합니다.

ACTIVE 인터페이스 작동 여부 및 사용 여부(yes 또는 no)를 나타냅니다.

GROUP IPMP 인터페이스 이름을 지정합니다. 익명 그룹의 경우 이 필드가 비어 있습니다. 익명 그룹에 대한 자세한 내용은 in.mpathd(1M) 매뉴얼 페이지를 참조하십시오.

FLAGS 각 기본 인터페이스의 상태를 나타내며, 다음 상태 중 하나 또는 모든 조합일 수 있습니다.

- i는 해당 인터페이스에 대해 INACTIVE 플래그가 설정되어 있음을 나타냅니다. 따라서 해당 인터페이스는 데이터 트래픽 송수신에 사용되지 않습니다.

예 6–11 IPMP 그룹의 기본 IP 인터페이스에 대한 정보 가져오기 *(계속)*

- s는 인터페이스가 대기 인터페이스로 구성되었음을 나타냅니다.
- m은 시스템에서 IPMP 그룹에 대한 IPv4 멀티캐스트 트래픽을 보내고 받는 데 해당 인터페이스를 지정했음을 나타냅니다.
- b는 시스템에서 IPMP 그룹에 대한 브로드캐스트 트래픽을 받는 데 해당 인터페이스를 지정했음을 나타냅니다.
- M은 시스템에서 IPMP 그룹에 대한 IPv6 멀티캐스트 트래픽을 보내고 받는 데 해당 인터페이스를 지정했음을 나타냅니다.
- d는 인터페이스가 작동 중지되었으므로 사용할 수 없음을 나타냅니다.
- h는 인터페이스가 다른 인터페이스와 중복 물리적 하드웨어 주소를 공유하며 오프라인 상태로 전환되었음을 나타냅니다. h 플래그는 인터페이스를 사용할 수 없음을 나타냅니다.

LINK 링크 기반 실패 감지의 상태를 나타내며 다음 상태 중 하나입니다.

- up 또는 down은 링크의 사용 가능 여부를 나타냅니다.
- unknown은 링크가 up 또는 down인지에 대한 알림을 드라이버가 지원하지 않으므로 링크 상태 변경을 감지하지 못함을 나타냅니다.

PROBE 테스트 주소로 구성된 인터페이스에 대한 프로브 기반 실패 감지 상태를 다음과 같이 지정합니다.

- ok는 프로브가 작동하며 활성 상태임을 나타냅니다.
- failed는 프로브 기반 실패 감지에서 인터페이스가 작동하지 않는 것이 감지되었음을 나타냅니다.
- unknown은 적합한 프로브 대상을 찾을 수 없으므로 프로브를 보낼 수 없음을 나타냅니다.
- disabled는 인터페이스에 IPMP 테스트 주소가 구성되어 있지 않음을 나타냅니다. 따라서 프로브 기반 실패 감지가 사용 안함으로 설정됩니다.

STATE 인터페이스의 전체 상태를 다음과 같이 지정합니다.

- ok는 인터페이스가 온라인 상태이며 실패 감지 방법의 구성에 따라 정상적으로 작동하고 있음을 나타냅니다.
- failed는 인터페이스의 링크가 작동 중지되었거나 프로브 감지에서 인터페이스가 트래픽을 보내거나 받을 수 없음이 확인되어 인터페이스가 작동하지 않음을 나타냅니다.
- offline은 인터페이스를 사용할 수 없음을 나타냅니다. 일반적으로 인터페이스는 다음과 같은 상황에서 오프라인으로 전환됩니다.
  - 인터페이스를 테스트하고 있습니다.
  - 동적 재구성을 수행하고 있습니다.

예 6–11 IPMP 그룹의 기본 IP 인터페이스에 대한 정보 가져오기 *(계속)*

- 인터페이스가 다른 인터페이스와 중복 하드웨어 주소를 공유합니다.

- unknown은 프로브 기반 실패 감지를 위한 프로브 대상을 찾을 수 없어서 IPMP 인터페이스의 상태를 확인할 수 없음을 나타냅니다.

예 6–12 IPMP 프로브 대상 정보 가져오기

- t 옵션은 IPMP 그룹의 각 IP 인터페이스와 연관된 프로브 대상을 식별합니다. 첫번째 출력은 프로브 기반 실패 감지에 필요한 테스트 주소를 사용하는 IPMP 구성의 예입니다.

```
$ ipmpstat -nt
INTERFACE   MODE          TESTADDR        TARGETS
net0        routes        192.168.85.30   192.168.85.1 192.168.85.3
net1        disabled      --              --
net3        disabled      --              --
net4        routes        192.1.2.200     192.1.2.1
net2        multicast     128.9.0.200     128.0.0.1 128.0.0.2
net6        multicast     128.9.0.201     128.0.0.2 128.0.0.1
net5        multicast     128.9.0.202     128.0.0.1 128.0.0.2
net7        multicast     128.9.0.203     128.0.0.1 128.0.0.2
```

두번째 출력은 테스트 주소 없이 추이적인 프로빙 또는 프로브 기반 실패 감지를 사용하는 IPMP 구성의 예입니다.

```
$ ipmpstat -nt
INTERFACE   MODE         TESTADDR         TARGETS
net3        transitive   <net1>           <net1> <net2> <net3>
net2        transitive   <net1>           <net1> <net2> <net3>
net1        routes       172.16.30.100    172.16.30.1
```

출력 필드는 다음 정보를 제공합니다.

INTERFACE IPMP 그룹의 각 기본 인터페이스를 지정합니다.

MODE 프로브 대상을 가져오는 방법을 지정합니다.

- routes는 시스템 경로 지정 테이블이 프로브 대상을 찾는 데 사용됨을 나타냅니다.
- mcast는 멀티캐스트 ICMP 프로브가 대상을 찾는 데 사용됨을 나타냅니다.
- disabled는 인터페이스에 대해 프로브 기반 실패 감지가 사용 안함으로 설정되었음을 나타냅니다.
- transitive는 두번째 예와 같이 추이적인 프로빙이 실패 감지에 사용됨을 나타냅니다. 추이적인 프로브와 테스트 주소를 동시에 사용하는 경우 프로브 기반 실패 감지를 구현할 수 없습니다. 테스트 주소를 사용하지 않으려는 경우 추이적인 프로빙을 사용으로 설정해야

예 6–12 IPMP 프로브 대상 정보 가져오기 *(계속)*

합니다. 전이적 프로브를 사용하지 않으려는 경우 테스트 주소를 구성해야 합니다. 개요는 78 페이지 “프로브 기반 실패 감지”를 참조하십시오.

TESTADDR -n 옵션을 -t 옵션과 함께 사용하는 경우 프로브를 보내고 받기 위해 인터페이스에 지정하는 IP 주소 또는 호스트 이름을 지정합니다.

전이적 프로브를 사용하는 경우 인터페이스 이름은 데이터를 받는 데 사용되지 않는 기본 IP 인터페이스를 나타냅니다. 또한 이 이름은 지정된 인터페이스의 소스 주소로 추이적인 테스트 프로브가 전송되고 있음을 나타냅니다. 데이터를 받는 활성 기본 IP 인터페이스의 경우 표시되는 IP 주소는 송신 ICMP 프로브의 소스 주소를 나타냅니다.

주 – IP 인터페이스가 IPv4 및 IPv6 테스트 주소로 구성된 경우 프로브 대상 정보가 각 테스트 주소에 대해 별도로 표시됩니다.

TARGETS 현재 프로브 대상을 공백으로 구분된 목록으로 나열합니다. 프로브 대상은 호스트 이름 또는 IP 주소로 표시됩니다. -n 옵션이 -t 옵션과 함께 사용되는 경우 IP 주소가 표시됩니다.

예 6–13 IPMP 프로브 관찰

- p 옵션을 사용하여 진행 중인 프로브를 관찰할 수 있습니다. 이 옵션을 ipmpstat 명령과 함께 사용하는 경우 Ctrl-C를 눌러 명령을 종료할 때까지 시스템의 프로브 작업에 대한 정보가 계속 표시됩니다. 이 명령을 실행하려면 기본 관리자 권한이 있어야 합니다.

첫번째 출력은 프로브 기반 실패 감지에 필요한 테스트 주소를 사용하는 IPMP 구성의 예입니다.

```
# ipmpstat -pn
TIME     INTERFACE   PROBE     NETRTT    RTT       RTTAVG    TARGET
0.11s    net0        589       0.51ms    0.76ms    0.76ms    192.168.85.1
0.17s    net4        612       --        --        --        192.1.2.1
0.25s    net2        602       0.61ms    1.10ms    1.10ms    128.0.0.1
0.26s    net6        602       --        --        --        128.0.0.2
0.25s    net5        601       0.62ms    1.20ms    1.00ms    128.0.0.1
0.26s    net7        603       0.79ms    1.11ms    1.10ms    128.0.0.1
1.66s    net4        613       --        --        --        192.1.2.1
1.70s    net0        603       0.63ms    1.10ms    1.10ms    192.168.85.3
^C
```

두번째 출력은 테스트 주소 없이 추이적인 프로빙 또는 프로브 기반 실패 감지를 사용하는 IPMP 구성의 예입니다.

예 6–13 IPMP 프로브 관찰 *(계속)*

```
# ipmpstat -pn
TIME     INTERFACE   PROBE     NETRTT   RTT      RTTAVG     TARGET
1.39S    net4        t28       1.05ms   1.06ms   1.15ms     <net1>
1.39s    net1        i29       1.00ms   1.42ms   1.48ms     172.16.30.1
^C
```

출력 필드는 다음 정보를 제공합니다.

TIME ipmpstat 명령이 실행된 시간을 기준으로 프로브가 전송된 시간을 지정합니다. ipmpstat를 시작하기 전에 프로브를 시작한 경우 명령이 실행된 시간을 기준으로 시간이 음수 값으로 표시됩니다.

INTERFACE 프로브가 전송되는 인터페이스를 지정합니다.

PROBE 프로브를 나타내는 식별자를 지정합니다. 전이적 프로브가 실패 감지에 사용되는 경우 식별자 앞에 t(전이적 프로브) 또는 i(ICMP 프로브)가 추가됩니다.

NETRTT 프로브의 총 네트워크 라운드 트립 시간을 지정하며 밀리초 단위로 측정됩니다. NETRTT는 IP 모듈이 프로브를 보내는 순간과 IP 모듈이 대상으로부터 ack 패킷을 받는 순간 사이의 시간을 나타냅니다. in.mpathd 데몬이 프로브가 손실되었음을 확인하면 필드가 비워집니다.

RTT 프로브의 총 라운드 트립 시간을 지정하며 밀리초 단위로 측정됩니다. RTT는 in.mpathd 데몬이 프로브를 보내는 코드를 실행하는 순간과 데몬이 대상의 ack 패킷 처리를 완료하는 순간 사이의 시간을 나타냅니다. 데몬이 프로브가 손실되었음을 확인하면 필드가 비워집니다. NETRTT에 없는 RTT에서 발생하는 스파이크는 로컬 시스템이 과부하되었음을 나타낼 수 있습니다.

RTTAVG 로컬 시스템과 대상 간의 인터페이스에서 프로브의 평균 라운드 트립 시간을 지정합니다. 평균 라운드 트립 시간은 느린 대상을 식별하는 데 도움이 됩니다. 데이터가 부족하여 평균을 계산할 수 없는 경우 이 필드가 비워집니다.

TARGET -n 옵션을 -p 옵션과 함께 사용하는 경우 프로브가 전송되는 대상 주소 또는 호스트 이름을 지정합니다.

ipmpstat 명령 출력 사용자 정의

ipmpstat 명령의 -o 옵션을 사용하여 출력을 사용자 정의할 수 있습니다. 기본 옵션에서 일반적으로 표시하는 총 필드 외에 표시할 특정 필드를 선택하려면 앞에 나온 다른 ipmpstat 옵션과 함께 이 옵션을 사용합니다.

예를 들어 -g 옵션은 다음 정보를 제공합니다.

- IPMP 그룹
- IPMP 그룹 이름
- 그룹 상태
- 실패 감지 시간
- IPMP 그룹의 기본 인터페이스

시스템에 있는 IPMP 그룹의 상태만 표시하려 한다고 가정하겠습니다. 다음 예에 표시된 것처럼 `-o` 및 `-g` 옵션을 결합하여 필드 `groupname` 및 `state`를 지정합니다.

```
$ ipmpstat -g -o groupname,state
GROUPNAME  STATE
ipmp0      ok
accgt1     failed
field2     degraded
```

특정 유형의 정보에 대해 `ipmpstat` 명령의 모든 필드를 표시하려면 구문에 `-o all`을 포함하십시오. 예를 들어 그룹 정보를 제공하는 모든 필드를 나열하려면 `ipmpstat -g -o all`을 입력합니다.

스크립트에서 ipmpstat 명령 사용

`-o` 옵션은 스크립트에서 또는 명령어 별명을 사용하여 명령을 실행하는 경우 특히 시스템에서 분석 가능한 출력을 생성하려는 경우 유용합니다.

시스템에서 분석 가능한 정보를 생성하려면 표시하려는 특정 필드가 포함되도록 다른 기본 `ipmpstat` 옵션 중 하나와 `-P` 및 `-o` 옵션을 결합합니다. 시스템에서 분석 가능한 출력과 일반 출력의 차이점은 다음과 같습니다.

- 열 헤더가 생략됩니다.
- 필드가 콜론(`:`)으로 구분됩니다.
- 빈 값이 포함된 필드가 이중 대시(`--`)로 채워지는 대신 비어 있습니다.
- 여러 필드가 요청되는 경우 필드에 리터럴 콜론(`:`) 또는 백슬래시(`\`)가 포함되어 있으면 이러한 문자 앞에 백슬래시(`\`)를 추가하여 이스케이프하거나 제외할 수 있습니다.

`ipmpstat -P` 구문을 올바르게 사용하려면 다음 규칙을 관찰합니다.

- `-o` *option field*를 `-P` 옵션과 함께 사용합니다. 여러 개의 옵션 필드는 쉼표로 구분합니다.
- `-o all`을 `-P` 옵션과 함께 사용하지 않습니다.

이러한 규칙을 하나라도 무시하면 `ipmpstat -P`가 실패합니다.

다음 예는 `-P` 옵션을 사용하는 경우의 정보 형식을 보여줍니다.

```
$ ipmpstat -P -o -g groupname,fdt,interfaces
ipmp0:10.00s:net0 net1
acctg1::[net3 net4]
field2:20.00s:net2 net7 (net5) [net6]
```

그룹 이름, 실패 감지 시간 및 기본 인터페이스는 그룹 정보 필드입니다. 따라서 `-o -g` 옵션을 `-P` 옵션과 함께 사용합니다.

`-P` 옵션은 스크립트에서 사용하기 위한 것입니다. 다음 예는 스크립트에서 ipmpstat 명령이 실행되는 방법을 보여줍니다. 이 스크립트는 특정 IPMP 그룹의 실패 감지 시간을 표시합니다.

```
getfdt() {
         ipmpstat -gP -o group,fdt | while IFS=: read group fdt; do
             [[ "$group" = "$1" ]] && { echo "$fdt"; return; }
         done
     }
```

◆ ◆ ◆ 7 장

7 LLDP를 사용하여 네트워크 연결 정보 교환

근거리 통신망에서는 시스템 및 스위치 같은 개별 구성 요소가 격리되어 구성되지 않습니다. 네트워크 트래픽을 효율적으로 호스트하려면 네트워크의 시스템 구성이 서로 조화를 이루어야 합니다. 따라서 패킷 교환은 구성 요소 기능, 링크 등록 정보 구성, 대역폭 제한 등에 따라 발생합니다. 각 시스템과 스위치, 다른 구성 요소를 수동으로 구성하여 이러한 구성 요소의 호환성을 확보할 수 있습니다. 그러나 이 방법은 위험하며 여러 시스템에서 여러 관리자가 독립적으로 작업하는 경우 특히 구성이 잘못되기 쉽습니다. 보다 효과적인 대안은 시스템이 개별 구성 정보를 피어 시스템으로 전송할 수 있는 기술을 사용하는 것입니다. 이러한 기술을 사용하면 위험이 감소되고 네트워크 관리가 용이해집니다. Oracle Solaris에서는 이러한 목적을 위해 LLDP(Link Layer Discovery Protocol)를 지원합니다.

이 장에서는 LLDP를 사용하여 로컬 네트워크 전체에서 시스템이 시스템 및 네트워크 연결 정보를 교환할 수 있게 하는 방법에 대해 설명합니다. 다음 항목을 다룹니다.

Oracle Solaris의 LLDP 개요

LLDP는 토폴로지 검색을 위해 LAN을 통해 정보를 공개합니다. 이 프로토콜을 사용할 경우 시스템이 네트워크의 다른 시스템에 연결 및 관리 정보를 알릴 수 있습니다. 이 정보에는 시스템 기능, 관리 주소 및 기타 네트워크 작업 관련 정보가 포함될 수 있습니다. 또한 이 프로토콜을 사용하면 동일한 시스템이 동일한 로컬 네트워크에 있는 다른 시스템에 대한 유사한 정보를 받을 수 있습니다.

Oracle Solaris에서는 LLDP가 DCBX(데이터 센터 브리징 교환 프로토콜) TLV 단위 교환에도 사용됩니다. DCBX는 PFC(우선 순위 기반 플로우 제어) 및 ETS(향상된 전송

선택) 같은 DCB 기능에 대한 구성 정보를 제공합니다. DCB에 대한 자세한 내용은 8 장, “Oracle Solaris의 데이터 센터 브리징 기능을 사용하는 작업”을 참조하십시오.

LLDP를 사용할 경우 시스템 관리자가 특히 VLAN(가상 LAN), 링크 통합 등을 포함하는 복잡한 네트워크에서 결함이 있는 시스템 구성을 쉽게 감지할 수 있습니다. 서버, 스위치, 기타 네트워크 구성 장치 간 물리적 연결을 추적할 필요 없이 토폴로지에 대한 정보를 쉽게 얻을 수 있습니다.

LLDP 구현의 구성 요소

LLDP는 다음 구성 요소로 구현됩니다.

- LLDP 기능을 사용으로 설정하려면 LLDP 패키지를 설치해야 합니다. 이 패키지는 LLDP 데몬, 명령줄 유틸리티, 서비스 매니페스트 및 스크립트와 LLDP 작동에 필요한 기타 구성 요소를 제공합니다.
- `lldp` 서비스는 `svcadm` 명령에 의해 사용으로 설정됩니다. 이 서비스는 LLDP 데몬을 관리하며 데몬 시작, 중지, 다시 시작 또는 새로 고침을 담당합니다. 이 서비스는 LLDP 패키지를 설치하면 자동으로 사용 설정됩니다.
- `lldpadm` 명령은 개별 링크에서 LLDP를 관리하며, LLDP의 작동 모드를 구성하고, 전송될 TLV(Type-Length-Value) 단위를 지정하고, DCBX TLV 단위를 구성하는 데 사용됩니다. 특히, 이 명령은 에이전트별 LLDP 등록 정보와 전역 LLDP 등록 정보를 설정하는 데 사용됩니다. `lldpadm` 명령의 일반 하위 명령은 `dladm` 및 `ipadm` 명령의 하위 명령과 유사합니다.
  - `lldpadm set-*`는 수행할 작업을 지정하며, 이 작업에서 지정한 LLDP 등록 정보에 대해 하나 이상의 값이 설정됩니다.
  - `lldpadm show-*`는 지정한 LLDP 등록 정보에 대해 설정되는 값을 표시합니다.
  - `lldpadm reset-*`는 지정한 LLDP 등록 정보의 구성을 기본값으로 재설정합니다.

  이러한 하위 명령의 사용은 이후 절에서 설명합니다. `lldpadm` 명령에 대한 자세한 내용은 `lldpadm(1M)` 매뉴얼 페이지를 참조하십시오.
- LLDP 라이브러리(`liblldp.so`)는 링크의 LLDP 정보를 검색하고 LLDP 패킷을 구문 분석하고 기타 기능을 수행하는 데 사용할 수 있는 API를 제공합니다.
- LLDP 에이전트는 LLDP가 사용으로 설정된 네트워크 인터페이스 카드와 연결된 LLDP 인스턴스입니다. LLDP 에이전트는 연결된 NIC의 LLDP 동작을 제어합니다. LLDP 에이전트는 NIC 또는 물리적 링크에서만 구성될 수 있으며 해당 링크의 포트를 사용하여 정보를 공개합니다. 따라서 이러한 LLDP 설명서에서는 LLDP 에이전트 이름, 사용으로 설정된 물리적 링크 및 포트가 동일합니다.
- LLDP 데몬(`lldpd`)은 시스템에서 LLDP 에이전트를 관리합니다. 또한 SNMP(Simple Network Management Protocol)를 통해 시스템에 수신된 LLDP 정보를 검색하는 SNMP의 데몬인 `snmpd`와 상호 작용합니다. 또한 이 데몬은 LLDP `sysevents` 정보를 게시하고 LLDP 라이브러리의 질의에 응답합니다.

LLDP 에이전트의 정보 소스

LLDP 에이전트는 LLDPDU(LLDP 데이터 단위)를 전송 및 수신합니다. 이 에이전트는 해당 LLDPDU에 포함된 정보를 관리하고 두 가지 유형의 데이터 저장소에 저장합니다.

- **로컬 MIB(Management Information Base) 또는 로컬 MIB** - 이 데이터 저장소에는 LLDP 에이전트가 사용으로 설정된 시스템의 특정 링크에 대한 네트워크 정보가 있습니다. 로컬 MIB에는 공통 정보와 고유 정보가 모두 포함됩니다. 예를 들어, 섀시 ID는 시스템의 모든 LLDP 에이전트에서 공유되는 공통 정보입니다. 그러나 포트 ID는 시스템의 데이터 링크마다 다릅니다. 따라서 각 에이전트가 해당 로컬 MIB를 관리합니다.
- **원격 MIB** - 이 데이터 저장소의 정보는 피어 호스트의 LLDP 에이전트로부터 수신됩니다.

LLDP 에이전트 작동 모드

LLDP 에이전트의 작동 모드는 다음과 같습니다.

- 전송 전용(`txonly`): 이 모드에서는 LLDP 에이전트가 수신 LLDPDU를 처리하지 않습니다. 따라서 원격 MIB가 비어 있습니다.
- 수신 전용(`rxonly`): 이 모드에서는 에이전트가 수신 LLDPDU만 처리하고 정보를 원격 MIB에 저장합니다. 그러나 로컬 MIB의 정보는 전송되지 않습니다.
- 전송 및 수신(`both`): 이 모드에서는 에이전트가 로컬 정보를 전송하고 수신 LLDPDU를 처리하므로 로컬 및 원격 MIB를 모두 유지 관리합니다. 기본 링크에서 DCB 기능을 지원하는 경우 지원되는 DCB 기능에 대해 DCBX TLV 단위가 자동으로 사용 설정됩니다.
- 사용 안함(`disable`): 이 모드에서는 에이전트가 없습니다.

LLDP의 SMF 등록 정보

SMFS(서비스 관리 기능) 등록 정보 `auto-enable-agents`는 시스템에서 LLDP가 사용으로 설정되는 방식을 제어합니다. 이 등록 정보를 사용하여 모든 물리적 링크에서 전역으로 또는 한 번에 하나의 물리적 링크에서만 LLDP를 사용으로 설정하도록 선택할 수 있습니다. 등록 정보 값은 세 가지 가능한 값 중 하나가 될 수 있습니다.

- `yes`는 SMF 등록 정보의 기본값입니다. 이 값을 사용하면 포트에 이전 LLDP 구성이 없는 경우 Rx 및 Tx 모드의 모든 포트에서 전역으로 LLDP가 사용으로 설정됩니다. 포트에 구성이 있는 경우 해당 포트의 구성이 유지됩니다. 예를 들어 포트가 이전에 Rx 모드 전용의 LLDP로 구성된 경우 LLDP 서비스는 에이전트가 Rx 및 Tx 모드로 실행되도록 전환하지 않습니다. 해당 포트의 LLDP는 계속 Rx 모드입니다.

- force를 사용하면 모든 포트에서 Rx 및 Tx 모드의 LLDP가 사용으로 설정되고 모든 포트에서 기존 LLDP 구성을 대체합니다. 예를 들어 포트의 이전 LLDP가 Rx 모드로만 실행되도록 구성된 경우 LLDP 에이전트가 기본 LLDP 모드인 Rx 및 Tx 모드 모두로 실행되도록 전환됩니다.
- no를 사용하면 기존 LLDP 구성이 있는 포트를 제외한 모든 포트에서 LLDP 자동 사용 설정이 사용 안함으로 설정됩니다. 포트의 기존 LLDP 구성은 유지됩니다.

auto-enable-agents 등록 정보를 사용자 정의할 때마다 새 값이 적용되도록 하려면 LLDP SMF 서비스를 다시 시작해야 합니다.

LLDP 에이전트 공개 정보

LLDP 에이전트는 시스템 및 연결 정보를 LLDP 패킷이나 LLDPDU로 전송합니다. 이러한 패킷에는 TLV(Type-Length-Value) 형식이 개별적으로 지정된 정보 단위가 포함됩니다. 따라서 정보 단위를 *TLV* **단위**라고도 합니다. 특정 TLV 단위는 필수이며 LLDP를 사용으로 설정한 경우 기본적으로 LLDP 패킷에 포함됩니다. lldpadm 명령을 사용하여 이러한 단위를 제외할 수 없습니다. 필수 TLV 단위는 다음과 같습니다.

- 섀시 ID
- 포트 ID
- TTL(활성 시간)
- PDU의 끝

섀시 ID는 hostid 명령으로 생성되는 정보와 동일합니다. 포트 ID는 물리적 NIC의 MAC 주소입니다. 링크 수에 따라 단일 시스템에서 여러 LLDP 에이전트를 사용으로 설정할 수 있습니다. 섀시 ID 및 포트 ID 결합은 에이전트를 고유하게 식별하며 시스템의 다른 에이전트와 구분합니다.

선택적 TLV 단위를 LLDP 패킷에 추가할 수 있습니다. 이러한 선택적 TLV 단위는 공급업체가 알릴 공급업체 관련 TLV 단위를 삽입할 수 있는 수단입니다. TLV 단위는 개별 OUI(Organization Unique Identifier)로 식별되며 이러한 OUI가 IEEE 802.1 사양인지 또는 IEEE 802.3 사양인지에 따라 입력됩니다. 이러한 선택적 TLV 단위 전송을 사용 또는 사용 안함으로 설정하도록 LLDP 에이전트 등록 정보를 구성할 수 있습니다.

다음 표에서는 각 TLV 유형이나 그룹, 해당 등록 정보 이름, 각 등록 정보에 대한 TLV 단위 및 해당 설명을 보여줍니다. 이러한 등록 정보 중 하나를 구성하여 LLDP를 사용으로 설정한 경우 패킷에 포함될 TLV 단위를 지정합니다.

표 7–1 LLDP 에이전트에 대해 사용으로 설정할 수 있는 TLV 단위

TLV 유형	등록 정보 이름	TLV 단위	설명
기본 관리	basic-tlv	sysname, portdesc, syscapab, sysdesc, mgmtaddr	알릴 시스템 이름, 포트 설명, 시스템 기능, 시스템 설명 및 관리 주소를 지정합니다.

표 7–1 LLDP 에이전트에 대해 사용으로 설정할 수 있는 TLV 단위 *(계속)*

TLV 유형	등록 정보 이름	TLV 단위	설명
802.1 OUI	dot1-tlv	vlanname, pvid, linkaggr, pfc, appln, evb, etscfg	VLAN 이름, 포트 VLAN ID, 링크 통합, 우선 순위 기반 플로우 제어의 TLV 단위, 응용프로그램, 향상된 전송 선택 및 에지 가상 브리징 같은 정보를 공개하도록 지정합니다.
802.3 OUI	dot3-tlv	max-framesize	알릴 최대 프레임 크기를 지정합니다.
Oracle 관련 OUI(0x0003BA로 정의됨)	virt-tlv	vnic	가상 네트워크가 구성된 경우 알릴 VNIC를 지정합니다.

TLV 단위 및 해당 등록 정보

각 TLV 단위에는 특정 값으로 구성할 수 있는 등록 정보가 있습니다. TLV 단위가 LLDP 에이전트의 등록 정보로 설정된 경우 TLV 단위는 네트워크에서 지정한 값으로만 알려집니다. 예를 들어, 시스템의 기능을 공개하는 TLV 값 syscapab를 고려해 보십시오. 이러한 기능은 라우터, 브릿지, 반복기, 전화 및 기타 장치에 대한 지원을 포함할 수 있습니다. 하지만 라우터 및 브릿지와 같이 실제로 특정 시스템에서 지원되는 기능만 알려지도록 syscapab를 설정할 수 있습니다.

TLV 단위 구성 절차는 **전역** *TLV* **단위** 또는 **에이전트별** *TLV* **단위**를 구성하는지에 따라 달라집니다.

전역 TLV 단위는 시스템의 모든 LLDP 에이전트에 적용됩니다. 다음 표에서는 전역 TLV 값과 가능한 해당 구성을 표시합니다.

표 7–2 전역 TLV 단위 및 해당 등록 정보

TLV 이름	TLV 등록 정보 이름	가능한 등록 정보 값	값 설명
syscapab	supported	other, repeater, bridge, wlan-ap, router, telephone, docsis-cd, station, cvlan, sylvan, tpmr	시스템에서 지원되는 주요 기능을 나타냅니다. 기본값은 router, station 및 bridge입니다.
	enabled	supported에 대해 나열되는 값의 하위 세트입니다.	시스템에서 사용으로 설정된 기능을 나타냅니다.

표 7–2 전역 TLV 단위 및 해당 등록 정보 (계속)

TLV 이름	TLV 등록 정보 이름	가능한 등록 정보 값	값 설명
mgmtaddr	ipaddr	ipv4 또는 ipv6	로컬 LLDP 에이전트와 연결될 IP 주소의 유형을 지정합니다. 이 주소는 상위 계층 엔티티에 도달하는 데 사용되며 네트워크 관리에 의한 검색을 지원합니다. 한 개의 유형만 지정할 수 있습니다.

전역 값을 가질 수 없는 TLV 단위는 LLDP 에이전트 레벨에서 관리됩니다. 에이전트별 TLV 단위를 사용할 경우 제공한 값은 특정 LLDP 에이전트에서 TLV 단위를 전송할 수 있을 때 사용됩니다.

다음 표에서는 LLDP 에이전트에 대한 TLV 값과 가능한 해당 구성을 표시합니다.

표 7–3 에이전트별 TLV 단위 및 해당 등록 정보

TLV 이름	TLV 등록 정보 이름	가능한 등록 정보 값	값 설명
pfc	willing	on, off	LLDP 에이전트에서 우선 순위 기반 플로우 제어에 관한 원격 시스템의 구성 정보를 승인 또는 거부하도록 설정합니다.
appln	apt	응용 프로그램 우선 순위 테이블에 정의된 정보에서 값을 가져옵니다.	응용 프로그램 우선 순위 테이블을 구성합니다. 이 표에는 응용 프로그램의 TLV 단위 및 해당 우선 순위 목록이 포함되어 있습니다. 응용 프로그램은 id/selector 쌍으로 식별됩니다. 테이블 내용은 다음 형식을 사용합니다. id/selector/priority
etscfg	willing	on, off	LLDP 에이전트에서 향상된 전송 선택에 관한 원격 시스템의 구성 정보를 승인 또는 거부하도록 설정합니다.

에이전트별 TLV 단위에 대한 자세한 내용은 8 장, “Oracle Solaris의 데이터 센터 브리징 기능을 사용하는 작업”을 참조하십시오.

시스템에서 LLDP 사용으로 설정

다음 절차에서는 네트워크의 다른 호스트 또는 피어와 시스템 정보를 교환하도록 LLDP를 구성하는 방법을 설명합니다.

▼ LLDP 배포 방법

다음 절차는 시스템에서 LLDP를 사용하여 시스템 기능 공개를 시작하는 방법에 대해 설명합니다. LLDP 패키지 설치를 완료하면 기본적으로 LLDP는 사용으로 설정되어 사용될 준비가 완료됩니다. 기본 LLDP 구성에 만족하는 경우 대부분의 단계가 선택 사항입니다.

시작하기 전에 LLDP를 사용하려면 LLDP 패키지를 설치해야 합니다. 패키지를 설치하려면 다음 명령을 입력합니다.

```
# pkg install lldp
```

1 LLDP 서비스가 시작되었는지 확인합니다.

```
# svcs lldp
STATE          STIME    FMRI
online         Jul_10   svc:/network/lldp:default
```

LLDP 서비스를 사용 안함으로 설정한 경우 다음 명령을 사용하여 서비스를 시작하십시오.

```
# svcadm enable svc:/network/lldp:default
```

2 다음 단계 중 하나를 수행합니다.

- LLDP 서비스를 시스템에서 전역으로 사용 설정하려면 LLDP 에이전트가 공개하는 TLV 단위를 지정합니다.

  # lldpadm set-agentprop -p *property*=*value* *agent*

  여기서 *agent*는 LLDP 에이전트이며 해당 에이전트가 사용되는 물리적 링크에 의해 식별됩니다. 따라서 `net0`에서 LLDP가 사용으로 설정된 경우 에이전트는 `net0`입니다.

주 – lldpadm 하위 명령을 실행하는 경우 축약형을 사용할 수 있습니다. 예를 들어, lldpadm set-agentprop 대신 lldpadm set-ap를 입력할 수 있습니다. 하위 명령 및 약어 형태는 lldpadm(1M) 매뉴얼 페이지를 참조하십시오.

LLDP 에이전트 등록 정보에 대한 설명은 114 페이지 "LLDP 에이전트 공개 정보"를 참조하십시오.

LLDP 에이전트 등록 정보 목록을 보려면 lldpadm show-agentprop 를 입력하십시오. 또는 표 7–1을 참조하십시오.

수행 방법은 120 페이지 "에이전트의 LLDP 패킷에 대해 TLV 단위를 지정하는 방법"을 참조하십시오.

- LLDP 서비스를 선택한 포트에서만 사용으로 설정하려면 다음 단계를 수행합니다.

  a. SMF auto-enable-agents 등록 정보를 no로 변경합니다.

     ```
     # svccfg -s svc:/network/lldp:default setprop lldp/auto-enable-agents = "no"
     ```

     이 SMF 등록 정보는 시스템에서 LLDP가 사용으로 설정되는 방법을 결정합니다. 가능한 3가지 값은 yes, force 및 no입니다. 기본적으로 이 등록 정보는 yes로 설정됩니다. 이러한 값과 이러한 값을 선택한 결과로 야기되는 LLDP 에이전트 동작에 대한 설명은 113 페이지 "LLDP의 SMF 등록 정보"를 참조하십시오.

  b. LLDP 서비스를 다시 시작합니다.

     ```
     # svcadm restart svc:/network/lldp:default
     ```

  c. 선택한 포트 또는 링크에서 LLDP 에이전트를 사용으로 설정합니다.

     ```
     # lldpadm set-agentprop -p mode=value agent
     ```

     여기서 *agent*는 LLDP 에이전트이며 해당 에이전트가 사용되는 물리적 링크에 의해 식별됩니다. 따라서 net0에서 LLDP를 사용으로 설정한 경우 에이전트는 net0입니다.

     등록 정보 mode는 LLDP 에이전트의 작동 모드를 나타내는 4가지 가능한 값 tx, rx, both 및 disable 중 하나로 설정할 수 있습니다. 이러한 값에 대한 설명은 113 페이지 "LLDP 에이전트 작동 모드"를 참조하십시오.

  d. LLDP 에이전트가 공개하는 TLV 단위를 지정합니다.

     ```
     # lldpadm set-agentprop -p property=value agent
     ```

     LLDP 에이전트 등록 정보에 대한 설명은 114 페이지 "LLDP 에이전트 공개 정보"를 참조하십시오.

     mode 등록 정보 외에, LLDP 에이전트의 다른 등록 정보 목록을 보려면 lldpadm show-agentprop를 입력하십시오. 또는 표 7–1을 참조하십시오.

     수행 방법은 120 페이지 "에이전트의 LLDP 패킷에 대해 TLV 단위를 지정하는 방법"을 참조하십시오.

3 **필요한 경우 전역 TLV 단위를 사용자 정의합니다.**

```
# lldpadm set-tlvprop -p property=value global-tlv
```

여기서 *property*는 전역 TLV 단위 등록 정보를 가리킵니다.

전역 TLV 단위에 대한 설명은 115 페이지 "TLV 단위 및 해당 등록 정보"를 참조하십시오.

전역 TLV 목록을 보려면 `lldpadm show-tlvprop`를 입력하십시오. 또는 표 7–2를 참조하십시오.

수행 방법은 121 페이지 "TLV 값을 정의하는 방법"을 참조하십시오.

4 **필요한 경우 에이전트별 TLV 단위를 사용자 정의합니다.**

```
# lldpadm set-agenttlvprop -p property=value -a agent per-agent-tlv
```

여기서 *property*는 에이전트별 TLV 단위 등록 정보를 가리킵니다.

에이전트별 TLV 단위에 대한 설명은 115 페이지 "TLV 단위 및 해당 등록 정보"를 참조하십시오.

에이전트별 TLV 목록을 보려면 `lldpadm show-tlvprop`를 입력하십시오. 또는 표 7–2를 참조하십시오.

수행 방법은 121 페이지 "TLV 값을 정의하는 방법"을 참조하십시오.

예 7–1 auto-enable-agents SMF 등록 정보 사용자 정의

다음 예에서는 SMF 등록 정보 값을 변경하는 경우 LLDP를 사용으로 설정하는 다른 방법을 보여줍니다. 포트가 4개인 시스템에서 다음과 같이 2개 포트의 LLDP를 구성한다고 가정합니다.

- `net0`: Rx 및 Tx 모드
- `net1`: Rx 전용
- `net2` 및 `net3`: 없음

SMF 등록 정보가 기본값 `yes`로 설정되면 `net2` 및 `net3`에서 LLDP가 자동으로 사용 설정됩니다. LLDP 구성은 다음과 같이 표시됩니다.

```
# lldpadm show-agentprop -p mode
AGENT    PROPERTY    PERM    VALUE    DEFAULT    POSSIBLE
net0     mode        rw      both     disable    txonly,rxonly,both,
                                                 disable
net1     mode        rw      rxonly   disable    txonly,rxonly,both,
                                                 disable
net2     mode        rw      both     disable    txonly,rxonly,both,
                                                 disable
net3     mode        rw      both     disable    txonly,rxonly,both,
                                                 disable
```

SMF 등록 정보를 `no`로 전환하면 서비스 다시 시작 시 구성이 변경됩니다.

```
# svccfg -s svc:/network/lldp:default setprop lldp/auto-enable-agents = "no"
# svcadm restart svc:/network/lldp:default
# lldpadm show-agentprop -p mode
AGENT    PROPERTY   PERM   VALUE    DEFAULT   POSSIBLE
net0     mode       rw     both     disable   txonly,rxonly,both,
                                               disable
net1     mode       rw     rxonly   disable   txonly,rxonly,both,
                                               disable
net2     mode       rw     disable  disable   txonly,rxonly,both,
                                               disable
net3     mode       rw     disable  disable   txonly,rxonly,both,
                                               disable
```

샘플 출력에서 해당 LLDP 모드가 이전에 자동으로 사용 설정된 net2 및 net3이 이제 사용 안함으로 플래그 지정됩니다. 그러나 LLDP 에이전트가 이전에 구성되었던 net0 및 net1에서는 변화가 없습니다.

예 7–2 여러 데이터 링크에서 LLDP 사용

이 예에서는 LLDP를 선택적으로 사용으로 설정하는 방법을 보여줍니다. 시스템에는 2개의 데이터 링크, net0 및 net1이 있습니다. net0에서는 에이전트가 LLDP 패킷을 전송하고 수신하도록 하려고 합니다. net1에서는 에이전트가 LLDP 패킷을 전송만 하도록 하려고 합니다. 이 경우 다음 명령을 입력합니다.

```
# svccfg -s svc:/network/lldp:default setprop lldp/auto-enable-agents = "no"
# svcadm restart svc:/network/lldp:default
# lldpadm set-agentprop -p mode=both net0
# lldpadm set-agentprop -p mode=txonly net1
```

▼ 에이전트의 LLDP 패킷에 대해 TLV 단위를 지정하는 방법

이 절차에서는 에이전트가 전송하는 LLDP 패킷에 공개되는 TLV 단위를 지정하는 방법에 대해 설명합니다. TLV 단위를 지정하려면 lldpadm set-agentprop 하위 명령을 사용합니다.

1 필요한 경우 추가할 TLV 단위를 포함할 수 있는 LLDP 에이전트 등록 정보를 식별합니다.

이 하위 명령은 각 등록 정보에 대해 이미 설정된 TLV 단위도 표시합니다.

```
# lldpadm show-agentprop agent
```

이 등록 정보를 지정하지 않을 경우 이 하위 명령은 모든 LLDP 에이전트 등록 정보와 해당 TLV 값을 표시합니다.

2 등록 정보에 TLV 단위를 추가합니다.

```
# lldpadm set-agentprop -p property[+|-]=value[,...] agent
```

+|- 수식자는 여러 값을 허용하는 등록 정보에 사용됩니다. 이러한 수식자를 사용하여 목록에서 값을 추가(+)하거나 제거(-)할 수 있습니다. 수식자를 사용하지 않으면 설정한 값이 이전에 등록 정보에 대해 정의된 모든 값을 대체합니다.

3 **(옵션) 등록 정보의 새 값을 표시합니다.**

```
# lldpadm show-agentprop -p property agent
```

예 7–3 LLDP 패킷에 선택적 TLV 단위 추가

이 예에서는 LLDP 에이전트 `net0`이 해당 LLDP 패킷의 VLAN 정보를 공개하도록 이미 구성되었습니다. 이 알림에 시스템 기능, 링크 통합 및 네트워크 가상화 정보도 포함하려고 합니다. 하지만 패킷에서 VLAN 설명은 제거하려고 합니다.

```
# lldpadm show-agentprop net0
AGENT    PROPERTY    PERM    VALUE           DEFAULT   POSSIBLE
net0     mode        rw      both            disable   txonly,rxonly,both,
                                                       disable
net0     basic-tlv   rw      sysname,         none     none,portdesc,
                             sysdesc                   sysname,sysdesc,
                                                       syscapab,mgmtaddr,
                                                       all
net0     dot1-tlv    rw      vlanname,        none     none,vlanname,pvid,
                             pvid,pfc                  linkaggr,pfc,appln,
                                                       evb,etscfg,all
net0     dot3-tlv    rw      max-framesize    none     none, max-framesize,
                                                       all
net0     virt-tlv    rw      none             none     none,vnic,all

# lldpadm set-agentprop -p basic-tlv+=syscapab,dot1-tlv+=linkaggr,virt-tlv=vnic net0
# lldpadm set-agentprop -p dot1-tlv-=vlanname net0
# lldpadm show-agentprop -p net0
AGENT    PROPERTY    PERM    VALUE           DEFAULT   POSSIBLE
net0     mode        rw      both            disable   txonly,rxonly,both,
                                                       disable
net0     basic-tlv   rw      sysname,         none     none,portdesc,
                             sysdesc,                  sysname,sysdesc,
                             syscapab                  syscapab,mgmtaddr,
                                                       all
net0     dot1-tlv    rw      pvid,            none     none,vlanname,pvid,
                             linkaggr                  linkaggr,pfc,appln,
                                                       evb,etscfg,all
net0     dot3-tlv    rw      max-framesize    none     none, max-framesize,
                                                       all
net0     virt-tlv    rw      vnic             none     none,vnic,all
```

▼ TLV 값을 정의하는 방법

이 절차에서는 특정 TLV 단위에 값을 제공하는 방법을 보여줍니다. 다음 하위 명령 중 하나를 사용합니다.

- 전역 TLV 단위를 구성하려면 `lldpadm set-tlvprop`를 사용합니다.

- 에이전트별 TLV 단위를 구성하려면 `lldpadm set-agenttlvprop`를 사용합니다.

1 전역 TLV 단위를 구성하는지 에이전트별 단위를 구성하는지에 따라 다음 단계 중 하나를 수행합니다.

- 전역 TLV 단위를 구성하려면 공개할 값이 포함되도록 적절한 TLV 등록 정보를 설정합니다.

  ```
  # lldpadm set-tlvprop -p tlv-property=value[,value,value,...] tlv-name
  ```

  여기서 *tlv-name*은 전역 TLV 단위 이름이며 *tlv-property*는 해당 TLV 단위 등록 정보입니다. 하나의 등록 정보에 여러 값을 지정할 수 있습니다. 참조는 표 7–2를 참조하십시오.

- 에이전트별 TLV 단위를 구성하려면 에이전트가 공개하도록 할 값이 포함되도록 LLDP 에이전트의 적절한 TLV 등록 정보를 구성합니다.

  ```
  # lldpadm set-agenttlvprop -p tlv-property[+|-]=value[,value,value,...] -a agent tlv-name
  ```

  여기서 *tlv-name*은 에이전트 TLV 단위 이름이며 *tlv-property*는 해당 TLV 단위 등록 정보입니다. 하나의 등록 정보에 여러 값을 지정할 수 있습니다. 참조는 표 7–3을 참조하십시오.

2 (옵션) 다음 단계 중 하나를 수행하여 방금 구성한 TLV 등록 정보 값을 표시합니다.

- 전역 TLV 등록 정보 값을 표시하려면 다음 명령을 사용합니다.

  ```
  # lldpadm show-tlvprop
  ```

- 에이전트 TLV 등록 정보 값을 표시하려면 다음 명령을 사용합니다.

  ```
  # lldpadm show-agenttlvprop
  ```

예 7–4 시스템의 기능 및 관리 IP 주소 지정

이 예에서는 다음 두 가지 목적을 달성합니다.

- LLDP 패킷에 알릴 시스템 기능에 대한 특정 정보를 제공합니다. 이를 위해서는 `syscapab` TLV 단위의 `supported` 및 `enabled` 등록 정보를 모두 구성해야 합니다.
- 알림에 사용되는 관리 IP 주소를 제공합니다.

```
# llpdadm set-tlvprop -p supported=bridge,router,repeater syscapab
# llpdadm set-tlvprop -p enabled=router syscapab
# llpdadm set-tlvprop -p ipaddr=192.168.1.2 mgmtaddr
# llpdadm show-tlvprop
TLVNAME    PROPERTY   PERM  VALUE        DEFAULT         POSSIBLE
syscapab   supported  rw    bridge,      bridge,router,  other,router,
                            router,      station         repeater,bridge,
                            repeater                     wlan-ap,telephone,
                                                         docis-cd,station,
                                                         cvlan,svlan,tpmr
syscapab   enabled    rw    router       none            bridge,router,
                                                         repeater
mgmtaddr   ipaddr     rw    192.162.1.2  none            --
```

LLDP 사용 안함으로 설정

개별 포트에서 선택적으로 LLDP를 사용 안함으로 설정하려면 다음 명령 중 하나를 사용합니다.

- `lldpadm set-agentprop -p mode=disable` *agent*

  여기서 *agent*는 LLDP 에이전트이며 해당 에이전트가 사용되는 물리적 링크에 의해 식별됩니다. 따라서 `net0`에서 LLDP를 사용으로 설정한 경우 에이전트는 `net0`입니다. 이 명령은 에이전트 모드를 변경하여 LLDP를 사용 안함으로 설정합니다.

- `lladpadm reset-agentprop`

  이 명령에서는 `mode` 등록 정보 값을 설정하지 않습니다. 이 명령은 포트의 LLDP 구성을 제거하여 LLDP를 사용 안함으로 설정합니다.

주의 – 하위 명령 `lldpadm reset–agentprop`는 LLDP 구성을 포트에서 완전히 제거합니다. `no`로 설정된 `auto-enable-agents`가 `yes`로 다시 전환되면 해당 포트의 에이전트 모드가 단순히 사용 안함으로 설정되었을 경우와 다르게 LLDP가 작동합니다.

모든 시스템 인터페이스에서 전역으로 LLDP를 사용 안함으로 설정하려면 다음 단계를 수행합니다.

1. SMF LLDP 등록 정보를 `no`로 변경합니다.

   ```
   # svccfg -s svc:/network/lldp:default setprop lldp/auto-enable-agents = "no"
   ```

2. LLDP 서비스를 다시 시작합니다.

   ```
   # svcadm restart svc:/network/lldp:default
   ```

3. 이전 LLDP 구성이 유지되는 각 포트에서 LLDP를 사용 안함으로 설정합니다.

   ```
   # lldpadm set-agentprop -p mode=disable agent
   ```

LLDP 에이전트 모니터링

`lldpadm show-agent` 하위 명령은 LLDP 에이전트가 알리는 전체 정보를 표시합니다. 지정된 시스템을 기준으로 알림은 네트워크의 나머지 부분으로 전송되는 로컬 시스템 정보일 수 있습니다. 또는 알림이 동일한 네트워크의 다른 시스템에서 시스템에 수신되는 정보일 수 있습니다.

이 절에서는 다음 두 절차에 대해 설명합니다.

- 124 페이지 “알림을 표시하는 방법”
- 125 페이지 “LLDP 통계를 표시하는 방법”

▼ 알림을 표시하는 방법

이 절차에서는 LLDP 에이전트가 알리는 정보를 표시하는 방법을 보여줍니다. 정보는 로컬 또는 원격일 수 있습니다. **로컬** 정보는 로컬 시스템에서 제공됩니다. **원격** 정보는 네트워크의 다른 시스템에서 제공되며 로컬 시스템에 수신됩니다.

- **`lldpadm show-agent` 하위 명령에 적절한 옵션을 사용하여 원하는 정보를 표시합니다.**
  - **LLDP 에이전트가 알리는 로컬 정보를 표시하려면 다음 명령을 입력합니다.**

    `# lldpadm show-agent -l` *agent*
  - **LLDP 에이전트에 수신되는 원격 정보를 표시하려면 다음 명령을 입력합니다.**

    `# lldpadm show-agent -r` *agent*
  - **로컬 또는 원격 정보를 자세히 표시하려면 다음 명령을 입력합니다.**

    `# lldpadm show-agent -[l|r]v` *agent*

예 7–5 알리는 LLDP 에이전트 정보 가져오기

다음 예에서는 LLDP 에이전트가 로컬 또는 원격으로 알리는 정보를 표시하는 방법을 보여줍니다. 기본적으로 이 정보는 간결한 형태로 표시됩니다. `-v` 옵션을 사용하면 상세 정보 표시 또는 자세한 정보를 가져올 수 있습니다.

```
# lldpadm show-agent -l net0
AGENT   CHASSISID   PORTID
net0    004bb87f    00:14:4f:01:77:5d

# lldpadm show-agent -lv net0
                    Agent: net0
         Chassis ID Subtype: Local(7)
            Port ID Subtype: MacAddress(3)
                    Port ID: 00:14:4f:01:77:5d
              Port Description: net0
                  Time to Live: 81 (seconds)
                   System Name: hosta.example.com
         System Description: SunOS 5.11 dcb-clone-x-01-19-11 i86pc
     Supported Capabilities: bridge,router
       Enabled Capabilities: router
         Management Address: 192.168.1.2
         Maximum Frame Size: 3000
               Port VLAN ID: --
               VLAN Name/ID: vlan25/25
        VNIC PortID/VLAN ID: 02:08:20:72:71:31
    Aggregation Information: Capable, Not Aggregated
                 PFC Willing: --
                     PFC Cap: --
                     PFC MBC: --
                  PFC Enable: --
                 PFC Pending: --
Application(s) (ID/Sel/Pri): --
    Information Valid Until: 117 (seconds)
```

```
# lldpdadm show-agent -r net0
AGENT   SYSNAME   CHASSISID   PORTID
net0    hostb     0083b390    00:14:4f:01:59:ab

# llpdadm show-agent -rv net0
                    Agent: net0
         Chassis ID Subtype: Local(7)
            Port ID Subtype: MacAddress(3)
                    Port ID: 00:14:4f:01:59:ab
          Port Description: net0
              Time to Live: 121 (seconds)
               System Name: hostb.example.com
         System Description: SunOS 5.11 dcb-clone-x-01-19-11 i86pc
     Supported Capabilities: bridge,router
       Enabled Capabilities: router
         Management Address: 192.168.1.3
         Maximum Frame Size: 3000
               Port VLAN ID: --
               VLAN Name/ID: vlan25/25
        VNIC PortID/VLAN ID: 02:08:20:72:71:31
    Aggregation Information: Capable, Not Aggregated
                PFC Willing: --
                    PFC Cap: --
                    PFC MBC: --
                 PFC Enable: --
Application(s) (ID/Sel/Pri): --
    Information Valid Until: 117 (seconds)
```

▼ LLDP 통계를 표시하는 방법

LLDP 통계를 표시하여 로컬 시스템이나 원격 시스템이 알리는 LLDP 패킷에 대한 정보를 가져올 수 있습니다. 이 통계는 LLDP 패킷 전송 및 수신과 관련된 중요한 이벤트를 나타냅니다.

1 LLDP 패킷 전송 및 수신에 대한 모든 통계를 표시하려면 다음 명령을 사용합니다.

```
# lldpadm show-agent -s agent
```

2 선택한 통계 정보를 표시하려면 -o 옵션을 사용합니다.

```
# lldpadm show-agent -s -o field[,field,...]agent
```

여기서 *field*는 `show-agent -s` 명령의 출력 결과에 있는 모든 필드 이름을 나타냅니다.

예 7–6 LLDP 패킷 통계 표시

이 예에서는 LLDP 패킷 알림에 대한 정보를 표시하는 방법을 보여줍니다.

```
# lldpadm show-agent -s net0
AGENT IFRAMES IEER IDISCARD OFRAMES OLENERR TLVDISCARD TLVUNRECOG AGEOUT
net0        9    0        0      14       0          4          5      0
```

명령 출력 결과에서 다음 정보를 제공합니다.

- AGENT는 LLDP 에이전트가 사용으로 설정된 데이터 링크와 동일한 LLDP 에이전트의 이름을 지정합니다.
- IFRAMES, IEER 및 IDISCARD는 수신 중인 패킷, 오류가 있는 수신 패킷 및 삭제된 수신 패킷에 대한 정보를 표시합니다.
- OFRAMES 및 OLENERR은 송신 패킷 및 길이 오류가 있는 패킷을 나타냅니다.
- TLVDISCARD 및 TLVUNRECOG는 삭제된 TLV 단위와 인식할 수 없는 TLV 단위에 대한 정보를 표시합니다.
- AGEOUT은 시간 초과된 패킷을 나타냅니다.

이 예에서는 표준을 준수하지 않아서 시스템에 수신된 9개 프레임 중 5개 TLV 단위를 인식할 수 없음을 나타냅니다. 또한 이 예에서는 로컬 시스템에서 네트워크로 14개 프레임이 전송되었음을 보여줍니다.

◆ ◆ ◆ 8 장

8 Oracle Solaris의 데이터 센터 브리징 기능을 사용하는 작업

7 장, “LLDP를 사용하여 네트워크 연결 정보 교환”에 설명된 LLDP와 마찬가지로, DCB(데이터 센터 브리징)에서도 네트워크에 있는 피어와의 정보 교환이 포함됩니다. 해당 정보는 데이터 센터와 같이 특히 트래픽이 심한 환경의 네트워크 패킷 무결성에 영향을 주는 구성을 참조합니다. DCB를 사용하면 이러한 센터의 고속 트래픽과 관련된 구성 요소 구성을 조정하여 효율적으로 네트워크 트래픽을 교환할 수 있습니다.

이 장에서는 다음 내용을 다룹니다.

DCB(데이터 센터 브리징) 개요

데이터 센터 브리징은 특히 네트워크 트래픽 볼륨 및 전송률이 높은 환경에서 트래픽을 관리하기 위한 기존 이더넷 네트워크 기능을 향상시키는 기능 세트입니다. 광 섬유 채널은 이 유형의 트래픽을 호스트하는 데 전용으로 사용할 수 있습니다. 그러나 전용 링크를 사용하여 광 섬유 채널 트래픽만 서비스할 경우 비용이 많이 들 수 있습니다. 따라서 FCoE(이더넷을 통한 광 섬유 채널) 트래픽이 보다 일반적으로 사용됩니다. DCB 기능은 이더넷 네트워크 순회 중 패킷 손실에 대한 광 섬유 채널의 민감도를 해결합니다.

DCB를 사용하면 피어에서 우선 순위 기반으로 트래픽을 구별할 수 있습니다. 우선 순위를 구별하여 호스트 간 혼잡의 경우 호스트에서 우선 순위가 높은 트래픽의 패킷 무결성이 보존되도록 할 수 있습니다. DCBX(DCB 교환 프로토콜)를 사용하면 통신 호스트에서 고속 네트워크 트래픽에 영향을 주는 구성 정보를 교환할 수 있습니다. 그러면 피어에서 공통 구성에 대해 협상하여 지속적인 트래픽 플로우를 유지하고 우선 순위가 높은 패킷의 패킷 손실을 예방합니다.

Oracle Solaris에서는 LLDP를 사용하여 DCBX TLV 단위를 교환합니다. 기본 NIC에서 DCB를 지원하는 경우 PFC(우선 순위 기반 플로우 제어) 및 ETS(향상된 전송 선택) DCB 기능을 네트워크의 피어 호스트와 공유할 수 있습니다.

- PFC(우선 순위 기반 플로우 제어)에서는 CoS(정의된 서비스 클래스) 우선 순위로 패킷의 트래픽 플로우를 일시 중지하는 방식을 구현하여 패킷 손실을 예방합니다. 129 페이지 "우선 순위 기반 플로우 제어"를 참조하십시오. CoS에 대한 자세한 내용은 `dladm(1M)` 매뉴얼 페이지의 `cos` 링크 등록 정보 설명을 참조하십시오.
- ETS(향상된 전송 선택)를 사용하여 정의된 CoS 우선 순위를 기반으로 패킷 사이에서 대역폭을 공유할 수 있습니다. 134 페이지 "향상된 전송 선택"을 참조하십시오.

LLDP를 사용하여 교환하는 모든 시스템 정보와 마찬가지로 호스트에는 2가지 유형의 DCB 정보인 로컬 DCB 정보와 원격 DCB 정보가 있습니다. PFC 기능이 유효하도록 하려면 호스트에 있는 PFC에 대한 이 2가지 유형의 DCB 정보가 대칭이어야 합니다. 일반적으로 피어로부터 수신하는 DCB 정보를 로컬 호스트에서 일치시킬 수 있어야 합니다. DCB가 사용되는 Oracle Solaris 시스템에서는 DCB 정보를 피어와 동기화할 수 있는 이 기능도 사용됩니다.

주 – 물리적 NIC에서 DCB를 지원하는 경우에만 Oracle Solaris 11 시스템에서 DCB 기능을 사용할 수 있습니다. 또한 해당 카드가 DCB 모드로 실행되도록 구성되어야 합니다.

▼ DCBX를 사용으로 설정하는 방법

LLDP를 사용하는 경우 DCBX 지원이 자동으로 사용됩니다. 이 절차에서는 특정 자동 프로세스가 실패하는 경우 대안이 될 수 있는 수동 단계를 제공합니다. 이 절차에서는 `net0`에 단계가 구현되는 것으로 가정합니다.

1 LLDP 패키지를 설치합니다.

```
# pkg install lldp
```

2 LLDP 서비스가 실행 중인지 확인합니다.

```
# svcs lldp
```

LLDP 서비스를 사용 안함으로 설정한 경우 다음 명령을 사용하여 서비스를 시작하십시오.

```
# svcadm enable svc:/network/lldp:default
```

3 LLDP 에이전트가 Rx 및 Tx 모드로 실행 중인지 확인합니다.

```
# lldpadm show-agentprop -p mode net0
```

LLDP 에이전트가 이러한 두 가지 모드 모두에서 사용되지 않는 경우 다음을 입력하십시오.

```
# lldpadm set-agentprop -p mode=both net0
```

자세한 내용은 113 페이지 "LLDP의 SMF 등록 정보"를 참조하십시오.

LLDP 에이전트의 다른 가능한 구성은 117 페이지 “시스템에서 LLDP 사용으로 설정”을 참조하십시오.

4 **기본 NIC가 DCB를 지원하는지 확인합니다.**

```
# dladm show-linkprop -p ntcs net0
```

등록 정보 값이 제로(0)보다 크면 NIC가 DCB를 지원함을 나타냅니다.

우선 순위 기반 플로우 제어

PFC는 IEEE 802.1p CoS 값을 포함하도록 표준 PAUSE 프레임을 확장합니다. PFC를 사용하면 PAUSE 프레임을 송신할 때 링크의 모든 트래픽을 정지시키지 않고 PFC 프레임에서 사용되는 CoS 값에 대해서만 트래픽을 일시 중지합니다. 트래픽을 일시 중지해야 하는 우선 순위를 사용으로 설정한 경우 PFC 프레임을 송신합니다. 송신 호스트는 해당 우선 순위의 트래픽을 중지하지만 다른 사용하지 않는 우선 순위의 트래픽은 영향을 받지 않습니다. PFC 프레임에 지정된 시간 간격 후 또는 송신 호스트가 다른 PFC 프레임을 수신한 후 해당 패킷에 대한 전송이 계속됩니다. 우선 순위 기반 일시 중지는 해당 우선 순위의 패킷이 삭제되지 않도록 합니다. 우선 순위가 정의되지 않은 패킷의 경우 PAUSE 프레임을 송신하지 않습니다. 따라서 트래픽은 플로우를 계속하고 트래픽 혼잡 중 패킷을 삭제할 수 있습니다.

우선 순위는 `pfcmap` 데이터 링크 등록 정보에 8비트 마스크(0–7)로 표시됩니다. 최저 비트는 우선 순위 0을 나타내고 최고 비트는 우선 순위 7을 나타냅니다. 이 마스크의 각 비트는 해당 우선 순위의 PFC가 사용되는지 여부를 나타냅니다. 기본적으로 `pfcmap`은 1111111로 설정되며 이는 모든 우선 순위의 PFC가 사용됨을 의미합니다. 링크를 통해 전송되는 모든 패킷은 수신 호스트의 혼잡이 가중되는 경우 송신 호스트로 PFC 프레임을 송신합니다.

PFC 관련 데이터 링크 등록 정보

`pfcmap` 등록 정보 외에도 우선 순위 정의 및 매핑에 대한 정보를 제공하는 등록 정보는 다음과 같습니다.

- `pfcmap-lcl-effective`는 로컬 호스트에서 작동 PFC 매핑을 참조합니다. 이 등록 정보에는 읽기 전용 권한이 있습니다. 이 등록 정보에는 `pfcmap` 등록 정보 값 또는 `pfcmap-rmt-effective` 등록 정보 값이 반영될 수 있습니다.
- `pfcmap-rmt-effective`는 원격 피어에서 작동 PFC 매핑을 참조합니다. 이 등록 정보에도 읽기 전용 권한이 있습니다.

PFC 프레임을 올바르게 송신하려면 통신 호스트에 대칭 DCB 구성 정보가 있어야 합니다. Oracle Solaris 11 시스템은 자동으로 해당 PFC 구성이 원격 피어의 PFC 구성과 일치하도록 조정할 수 있습니다.

나열된 두 등록 정보는 피어 간 PFC 정보가 동기화되었는지 여부를 간접적으로 나타냅니다. 로컬 및 원격 피어에서 일치하는 PFC 정보가 포함되어 있는 데이터 링크에서는 `pfcmap`에 대해 설정된 값에 상관없이 `pfcmap-lcl-effective` 및 `pfcmap-rmt-effective`의 값이 동일합니다. 동기화 기능을 로컬 호스트에서 사용 안함으로 설정한 경우 `pfcmap-lcl-effective`에 로컬 호스트의 `pfcmap` 등록 정보 값이 반영됩니다.

이러한 등록 정보 구성에서 제공하는 PFC 정보의 예는 131 페이지 "PFC 구성 정보 가져오기"를 참조하십시오.

우선 순위 기반 플로우 제어 TLV 단위

PFC TLV 단위는 피어 호스트에서 수신하는 정보와 관련하여 호스트 동작을 제어합니다. TLV 단위에는 구성 가능한 등록 정보가 `willing` 하나만 있습니다. 기본적으로 이 등록 정보는 `on`으로 설정되어 있어 로컬 호스트에서 해당 PFC 우선 순위 정의를 원격 피어의 PFC 정의와 동기화할 수 있습니다. 다음과 같이 이 등록 정보를 `off`로 전환하여 특정 에이전트의 자동 정보 동기화를 방지할 수 있습니다.

```
# lldpadm set-agenttlvprop -p willling=off -a agent pfc
```

여기서 *agent*는 해당 에이전트가 사용되는 데이터 링크에 의해 식별됩니다.

▼ DCB의 우선 순위 기반 플로우 제어 사용자 정의 방법

대부분의 경우 PFC 기본 구성인 on을 사용하면 충분합니다. LLDP를 사용하는 경우 이 구성은 자동으로 설정됩니다. 그러나 PFC 구성 시 사용할 수 있는 다른 옵션을 제공하기 위해 이 절차에는 PFC 구성을 위한 수동 단계가 나와 있습니다. 이 절차에서는 자동 구성이 없는 것으로 가정합니다. 단계를 이해하는 데 도움이 되도록 모든 구성은 `net0`에서 수행됩니다.

1 **DCBX가 사용되는지 확인합니다.**

128 페이지 "DCBX를 사용으로 설정하는 방법"을 참조하십시오.

2 **(옵션) 사용하려는 DCB 기능을 사용자 정의합니다.**

기본적으로 PFC, ETS 및 EVB(에지 가상 브리징)가 사용됩니다. PFC만 사용하려는 것으로 가정하겠습니다. 그러면 LLDP 에이전트의 `dot1-tlv` 등록 정보에서 다른 두 값을 제거해야 합니다. `dot1-tlv`의 가능한 값 목록은 표 7–3을 참조하십시오.

```
# lldpadm set-agenttlvprop -p dot1-tlv-=etscfg,evb net0
```

3 **데이터 링크의 `flowctrl` 등록 정보가 `pfc`로 설정되어 있는지 확인합니다.**

```
# dladm show-linkprop -p flowctrl net0
```

등록 정보의 값 목록에 pfc가 없는 경우 다음 명령을 실행합니다.

```
# dladm set-linkprop -p flowctrl=pfc net0
```

4 **기본값 11111111을 사용하지 않으려면 pfcmap 등록 정보를 적절하게 설정합니다.**
예를 들어 CoS 우선 순위 6에서만 우선 순위를 사용하려면 다음 명령을 입력하십시오.

```
# dladm set-linkprop -p pfcmap=01000000 net0
```

5 **호스트에서 해당 PFC 정보를 원격 피어의 PFC 정보와 동기화할 수 있는지 확인합니다.**

```
# lldpadm show-agenttlvprop -p willing -a net0 pfc
```

PFC TLV 등록 정보 willing이 off로 설정되어 있는 경우 다음 명령을 실행하십시오.

```
# lldpadm set-agenttlvprop -p willing=on -a net0 pfc
```

PFC 구성 정보 가져오기

이 절에는 LLDP 및 DCB 구성 후의 PFC 관련 정보에 대한 여러 예가 들어 있습니다.

다음 명령은 PFC와 관련된 정보를 표시합니다.

- `dladm show-linkprop -p pfcmap,pfc-lcl-effective,pfc-rmt-effective` *datalink*
  이 명령은 우선 순위 정의 및 데이터 링크의 유효한 PFC 매핑을 표시합니다.
- `dladm show-phys -D pfc` *datalink*
  이 명령은 NIC에서 사용되는 우선 순위와 관련된 물리적 링크에 대한 PFC 정보를 표시합니다.
- `lldpadm show-agenttlvprop -a` *agent* `pfc`
  여기서 *agent*는 LLDP가 사용되는 데이터 링크에 의해 식별됩니다. 따라서 LLDP 에이전트의 이름은 데이터 링크의 이름과 동일합니다. 이 명령은 해당 PFC 매핑을 피어와 동기화하는 호스트의 기능을 제어하는 PFC TLV 등록 정보를 표시합니다.
- `lldpadm show-agent -lv -o "PFC Pending"` *agent*
  이 명령은 로컬 호스트와 피어의 PFC 매핑 정보 불일치를 경고합니다.

다음 예는 앞에 나열된 명령으로 표시되는 정보의 유형을 보여줍니다.

예 8–1 PFC 관련 데이터 링크 등록 정보 표시

이 예는 우선 순위 기반 플로우 제어와 관련된 데이터 링크 등록 정보의 상태를 표시하는 방법을 보여줍니다.

```
# dladm show-linkprop -p pfcmap,pfc-lcl-effective,pfc-rmt-effective net0
LINK    PROPERTY               PERM   VALUE      DEFAULT    POSSIBLE
net0    pfcmap                 rw     11111111   11111111   00000000-11111111
net0    pfcmap-lcl-effective   r-     11111111   --         --
net0    pfcmap-rmt-effective   r-     01000000   --         --
```

예 8–1 PFC 관련 데이터 링크 등록 정보 표시 *(계속)*

출력은 로컬 호스트의 PFC 매핑에 8개의 우선 순위가 모두 사용되는 기본값이 있음을 나타냅니다. `pfcmap-lcl-effective` 및 `pfcmap-rmt-effective`의 값이 불일치하는 것은 로컬 호스트에서 해당 PFC 정보가 원격 피어와 동기화되지 않았음을 나타냅니다. 이러한 불일치는 동기화가 해제되도록 전환할 수 있는 등록 정보 때문일 수 있습니다. 또는 피어가 PFC TLV 단위를 네트워크로 보내지 않고 있는 것입니다. 다음 명령을 입력하여 이 구성을 확인할 수 있습니다.

예 8–2 PFC 정보를 동기화하는 로컬 호스트 기능 표시

이 예는 피어의 PFC 구성에 맞게 조정하는 호스트 기능의 현재 상태를 표시하는 방법을 보여줍니다.

```
# lldpadm show-agenttlvprop -a net0 pfc
AGENT   TLVNAME   PROPERTY   PERM   VALUE   DEFAULT   POSSIBLE
net0    pfc       willing    rw     off     on        on,off
```

동기화를 사용하려면 다음 명령을 실행하십시오.

```
# lldpadm set-agenttlvprop -p willing=on -a net0 pfc
```

```
# dladm show-linkprop -p pfcmap,pfc-lcl-effective,pfc-rmt-effective net0
LINK   PROPERTY               PERM   VALUE      DEFAULT    POSSIBLE
net0   pfcmap                 rw     11111111   11111111   00000000-11111111
net0   pfcmap-lcl-effective    r-     01000000   --         --
net0   pfcmap-rmt-effective    r-     01000000   --         --
```

두번째 출력에서는 로컬 호스트가 고유 PFC 매핑(11111111)을 삭제했습니다. 대신, 호스트에서 피어와 동기화하여 해당 유효 PFC 매핑이 피어의 PFC 매핑과 동일해졌습니다. 이러한 값의 통합을 통해 호스트는 PFC PAUSE 프레임을 성공적으로 교환할 수 있습니다.

예 8–3 호스트 및 피어 간 PFC 정보 대칭 확인

이 예는 실제 실행 시간에 PFC 정보가 호스트 및 피어에서 동기화되었는지 여부 또는 불일치가 발생했는지 여부를 확인하는 방법을 보여줍니다.

```
# lldpadm show-agent -lv -o "PFC Pending" net0
PFC Pending: True
```

호스트 및 피어 사이의 PFC 정보가 통합되지 않은 경우 `PFC Pending`은 `True` 상태가 됩니다. 불일치가 해결되면 `PFC Pending`의 상태가 `False`로 돌아옵니다.

에이전트에서 공개한 정보를 모두 표시하려면 `lldpadm show-agent` 명령의 `verbose` 옵션을 사용하십시오.

```
# lldpadm show-agent -v agent
```

예 8–4 CoS 우선 순위 정의 표시

이 예는 `pfcmap` 등록 정보의 값에 따라 특정 데이터 링크의 현재 CoS 우선 순위 정의를 표시하는 방법을 보여줍니다. 예를 들어 `pfcmap`이 `01000000`으로 구성되었다고 가정하겠습니다. 물리적 링크에 해당하는 우선 순위 매핑을 표시하려면 다음과 같이 수행합니다.

```
# dladm show-phys -D pfc net0
LINK        COS   PFC   PFC_EFFECT   CLIENTS
ixgbe0      0     YES           NO   net0,vnic1
            1     YES          YES   vnic2
            2     YES           NO   vnic3
            3     YES           NO   vnic4
            4     YES           NO   vnic5
            5     YES           NO   vnic6
            6     YES           NO   vnic7
            7     YES           NO   vnic8
```

물리적 링크 `net0`의 경우 데이터 링크를 통해 구성된 모든 VNIC 클라이언트에 대해 우선 순위를 사용합니다. 그러나 로컬 호스트는 `PFC_EFFECT` 필드 값에 표시된 것처럼 해당 PFC 매핑을 피어의 PFC 매핑에 맞게 조정합니다. 우선 순위는 CoS `0` 및 `2-7`에서 사용 안함으로 설정됩니다. 따라서 리소스 가용성과 상관없이 `vnic2`를 제외한 모든 VNIC에서 트래픽을 위해 PFC 프레임이 교환되지 않습니다. 이러한 구성에서는 `vnic2`를 제외한 모든 VNIC에서 경유하는 트래픽에 대해 패킷 삭제를 허용합니다. `vnic2`의 트래픽에서는 트래픽 혼잡이 발생하는 경우 해당 클라이언트의 패킷 손실을 방지하기 위해 PFC PAUSE 프레임을 송신합니다.

응용프로그램 TLV 단위

응용프로그램 TLV 단위에는 호스트의 응용프로그램에 사용되는 우선 순위에 대한 정보가 있습니다. 우선 순위는 응용프로그램 우선 순위 테이블에 정의되어 있습니다. 테이블의 각 항목에는 응용프로그램 이름과 해당 응용프로그램에 지정된 우선 순위가 포함되어 있습니다. 응용프로그램 TLV는 이 테이블을 사용하여 다른 호스트에 응용프로그램 우선 순위 정보를 전송합니다.

테이블 항목은 다음 형식을 사용합니다.

protocol-id/*selector*/ *priority*

protocol-id/*selector* 쌍은 응용프로그램을 나타냅니다. *Priority*에는 0 - 7 중 해당 응용프로그램의 우선 순위를 나타내는 값이 포함되어 있습니다.

응용프로그램 우선 순위에 대한 정보를 다른 호스트와 교환하려면 다음과 같이 응용프로그램 TLV를 설정합니다.

```
# lldpadm set-agenttlvprop -p property=value -a agent appln
```

예를 들어 FCoE 트래픽의 경우 프로토콜 ID는 `0x8906`이고 선택기 ID는 `1`입니다. 이 응용프로그램에 지정된 우선 순위는 `4`라고 가정하겠습니다. 응용프로그램 TLV를 설정하는 데 사용되는 매개변수가 나열된 표 7–3에 따라 다음 명령을 입력하십시오.

```
# lldpadm set-agenttlvprop -p apt=8906/1/4 -a net0 appln
# lldpadm show-agenttlvprop -a net0 appln
AGENT    TLVNAME    PROPERTY    PERM    VALUE       DEFAULT    POSSIBLE
net0     appln      apt         rw      8906/1/4    --         --
```

향상된 전송 선택

ETS는 DCB 우선 순위에 따라 NIC에서 응용프로그램에 대역폭을 할당할 수 있는 DCB 기능입니다. DCB 우선 순위는 3비트 우선 순위 필드가 포함된 VLAN 헤더입니다. 우선 순위 필드 값은 네트워크의 이더넷 패킷과 다릅니다. DCB는 802.1p 우선 순위라고도 불리는 우선 순위 값을 사용하여 PFC 구성 및 링크 대역폭 같은 다른 DCB 등록 정보와 트래픽을 연결합니다. 해당 우선 순위 값에 따라 패킷에 할당할 특정 대역폭을 설정하도록 DCB를 구성합니다.

ETS를 사용하려면 NIC가 DCB를 지원하고 DCB 모드로 실행되어야 합니다.

ETS 관련 데이터 링크 등록 정보

PFC 정보를 참조하는 데이터 링크 등록 정보는 패킷에 대해 정의된 CoS 우선 순위에 따라 패킷 손실을 방지하는 데 적용됩니다. ETS 정보를 참조하는 등록 정보는 동일한 CoS 우선 순위에 따라 패킷에 공유 대역폭을 할당하는 데 적용됩니다. ETS는 다음 데이터 링크 등록 정보에서 구성합니다.

- `cos`는 데이터 링크의 서비스 클래스를 지정합니다. 등록 정보는 이더넷 우선 순위를 나타냅니다. 0 - 7 범위의 등록 정보 값이 데이터 링크의 아웃바운드 패킷에 적용됩니다. 해당 값은 아웃바운드 패킷의 VLAN 태그에 설정됩니다. 이 등록 정보가 물리적 링크 자체에 설정된 경우 해당 링크의 기본 클라이언트 트래픽에만 우선 순위가 적용됩니다. 이 우선 순위는 VNIC 같은 다른 보조 클라이언트에는 설정되지 않습니다. NIC가 DCB 모드로 실행 중이거나 링크가 VLAN인 경우 기본적으로 `cos`는 `0`으로 설정됩니다.
- `etsbw-lcl`은 데이터 링크의 TX측에 할당되는 ETS 대역폭을 나타냅니다. 기본 물리적 NIC에 DCB 기능이 있고 ETS가 지원되는 경우에만 이 등록 정보를 구성할 수 있습니다. 보조 데이터 링크 또는 클라이언트에 할당하려는 기본 NIC의 총 대역폭 비율을 지정하여 값을 설정합니다. 링크의 `cos`가 제로(`0`)로 설정되지 않은 경우 이 등록 정보를 설정할 수 있습니다.

주 – 통합으로 구성된 DCB 모드의 물리적 링크에서는 현재 ETS가 지원되지 않습니다.

`etsbw-lcl`에 정의된 대역폭 비율은 보조 클라이언트 전용으로 예약된 비율이 아닙니다. 할당된 대역폭을 사용하지 않으면 유사하게 구성된 다른 클라이언트에서 사용할 수 있습니다. 또한 대역폭 할당은 호스트 트래픽의 전송측에만 강제 적용됩니다.

이전 목록의 등록 정보 외에도 로컬 호스트와 해당 피어 사이에서 교환되는 대역폭 데이터에 대한 정보를 제공하는 읽기 전용 등록 정보는 다음과 같습니다.

- `etsbw-lcl-advice`는 권장 대역폭 점유율을 지정합니다. 데이터 링크의 이러한 권장 대역폭을 원격 피어에서 로컬 호스트로 송신합니다.
- `etsbw-lcl-effective`는 로컬 호스트의 데이터 링크에 구현된 실제 대역폭 점유율을 참조합니다. 이 등록 정보에는 `etsbw-lcl` 등록 정보 값 또는 `etsbw-lcl-advice` 등록 정보 값이 반영될 수 있습니다.
- `estbw-rmt-effective`는 원격 피어에 구성된 대역폭 점유율을 참조합니다.

특정 우선 순위 패킷에 해당하는 대역폭이 사용되도록 하려면 통신 호스트 사이의 대칭 또는 동기화된 ETS 정보를 사용하는 것이 좋습니다. 특히 `etsbw-lcl-advice` 값에 맞게 해당 대역폭 점유율을 조정하는 로컬 시스템 기능을 사용하는 것이 바람직합니다. Oracle Solaris 11 시스템은 자동으로 해당 ETS 구성이 원격 피어의 ETS 구성과 일치하도록 조정할 수 있습니다.

`estbw-lcl-effective` 등록 정보는 ETS 정보를 피어와 일치시키는 로컬 호스트 기능이 사용되는지 여부를 간접적으로 표시합니다. 등록 정보 값이 `etsbw-lcl-advice` 값과 일치하는 경우 해당 기능이 사용되는 것입니다. 그렇지 않으면 `etsbw-lcl-effective` 및 `etsbw-lcl` 등록 정보 값이 동일합니다.

향상된 전송 선택 TLV 단위

ETS TLV 단위 `etscfg`는 피어 호스트에서 수신하는 정보와 관련하여 호스트 동작을 제어합니다. TLV 단위에는 구성 가능한 등록 정보가 `willing` 하나만 있습니다. 기본적으로 이 등록 정보는 `on`으로 설정되어 있어 로컬 호스트에서 해당 ETS 구성을 원격 피어의 ETS 구성과 동기화할 수 있습니다. 특정 에이전트의 정보 동기화를 방지해야 하는 경우 다음과 같이 `willing` 등록 정보를 `off`로 설정하십시오.

```
# lldpadm set-agenttlvprop -p willing=off -a agent etscfg
```

여기서 *agent*는 해당 에이전트가 사용되는 데이터 링크에 의해 식별됩니다.

▼ DCB의 향상된 전송 선택 사용자 정의 방법

대부분의 경우 시스템의 ETS 기본 구성을 사용하면 충분합니다. LLDP가 사용되고 DCB가 기본 링크에서 지원되며 기본 링크가 DCB 모드로 실행 중인 경우 이 구성이 자동으로 설정됩니다. 그러나 ETS 구성 시 사용할 수 있는 다른 옵션을 제공하기 위해 이 절차에는 ETS 구성을 위한 수동 단계가 나와 있습니다. 이 단계에서는 자동 구성이 없으며 가상 클라이언트 `vnic1`에서 구성이 수행되는 것으로 가정합니다. LLDP 에이전트인 `net0`을 통해 가상 클라이언트가 구성됩니다.

1 DCBX가 사용되는지 확인합니다.

128 페이지 “DCBX를 사용으로 설정하는 방법”을 참조하십시오.

2 **(옵션) 사용하려는 DCB 기능을 사용자 정의합니다.**

기본적으로 PFC, ETS 및 EVB(에지 가상 브리징)가 사용됩니다. EVB를 사용 안함으로 설정하는 것으로 가정하겠습니다. 그러면 LLDP 에이전트의 `dot1-tlv` 등록 정보에서 다른 두 가지를 제거해야 합니다.

```
# lldpadm set-agenttlvprop -p dot1-tlv-=evb net0
```

3 **CoS 우선 순위 정의를 VNIC로 설정합니다.**

```
# dladm set-linkprop -p cos=value vnic1
```

4 **물리적 링크의 총 대역폭과 공유되는 VNIC 대역폭을 설정합니다.**

```
# dladm set-linkprop -p etsbw-lcl=value vnic1
```

`etsbw-lcl` 등록 정보에 지정하는 값은 기본 링크의 총 대역폭 용량 비율을 나타냅니다. 클라이언트에 지정하는 모든 할당된 대역폭 값의 합계가 100%를 초과하지 않아야 합니다.

5 **호스트에서 해당 ETS 정보를 원격 피어의 ETS 정보와 동기화할 수 있는지 확인합니다.**

```
# lldpadm show-agenttlvprop -p willing -a net0 etscfg
```

`willing` 등록 정보가 `off`로 설정되어 있는 경우 다음 명령을 실행하십시오.

```
# lldpadm set-agenttlvprop -p willing=on -a net0 etscfg
```

ETS 구성 정보 가져오기

이 절에는 LLDP 및 DCB 구성 후의 ETS 구성 관련 정보에 대한 여러 예가 들어 있습니다.

다음 명령은 ETS 구성에 대한 정보를 표시합니다.

- `dladm show-linkprop -p etsbw-lcl,etsbw-advise,etsbw-lcl-effective,etsbw-rmt-effective` *datalink*

  이 명령은 대역폭 할당 정의 및 데이터 링크에 구현된 유효 할당을 표시합니다.

- `dladm show-phys -D ets` *datalink*

  이 명령은 링크의 대역폭 할당 및 분배와 관련된 물리적 링크의 ETS 구성을 표시합니다.

- `lldpadm show-agenttlvprop -a` *agent* `etscfg`

  여기서 *agent*는 LLDP가 사용되는 데이터 링크에 의해 식별됩니다. 이 명령은 ETS 정보를 피어와 동기화하는 호스트의 기능을 제어하는 ETS TLV 등록 정보를 표시합니다.

다음 예는 나열된 명령으로 표시되는 정보의 유형을 보여줍니다.

예 8–5 ETS 관련 데이터 링크 등록 정보 표시

이 예는 향상된 전송 선택과 관련된 데이터 링크 등록 정보의 상태를 표시하는 방법을 보여줍니다.

```
# dladm show-linkprop -p cos,etsbw-lcl,etsbw-lcl-advise, \
etsbw-lcl-effective,etsbw-rmt-effective vnic1
LINK    PROPERTY              PERM  VALUE      DEFAULT    POSSIBLE
vnic1   cos                   rw    2          0          0-7
vnic1   etsbw-lcl             rw    20         0          --
vnic1   etsbw-lcl-advise      r-    --         --         --
vnic1   etsbw-lcl-effective   r-    --         --         --
vnic1   etsbw-rmt-effective   r-    --         --         --
```

출력은 vnic1의 대역폭 점유율이 해당 물리적 링크에 사용 가능한 총 대역폭의 20%로 구성되었음을 보여줍니다. cos 등록 정보로 표시되는 VNIC 802.1p 우선 순위는 2로 설정되어 있습니다.

예 8–6 ETS 정보를 동기화하는 로컬 호스트 기능 표시

이 예는 피어의 ETS 구성에 맞게 조정하는 로컬 호스트 기능의 현재 상태를 표시하는 방법을 보여줍니다.

```
# lldpadm show-agenttlvprop -a net0 etscfg
AGENT   TLVNAME   PROPERTY   PERM   VALUE   DEFAULT   POSSIBLE
net0    etscfg    willing    rw     off     on        on,off
```

동기화를 사용하려면 다음 명령을 실행하십시오.

```
# lldpadm set-agenttlvprop -p willing=on -a net0 etscfg

# dladm show-linkprop -p etsbw-lcl,etsbw-lcl-advise, \
etsbw-lcl-effective,etsbw-rmt-effective vnic0
LINK    PROPERTY              PERM  VALUE      DEFAULT    POSSIBLE
vnic1   cos                   rw    2          0          0-7
vnic1   etsbw-lcl             rw    20         0          --
vnic1   etsbw-lcl-advise      r-    15         --         --
vnic1   etsbw-lcl-effective   r-    15         --         --
vnic1   etsbw-rmt-effective   r-    25         --         --
```

vnic1의 estbw-lcl이 20%로 설정된 경우에도 VNIC 유효 대역폭 점유율은 피어에서 수신한 권장 대역폭과 일치하는 15%입니다. etscfg TLV 단위의 willing 등록 정보를 설정으로 전환했으므로 조정이 발생합니다.

다음 예는 물리적 링크의 우선 순위 매핑을 보여줍니다.

```
# dladm show-phys -D ets net0
LINK        COS   ETSBW   ETSBW_EFFECT   CLIENTS
ixgbe0      0     20      20             <default,mcast>,net0
            1     15      15             vnic2
            2     20      20             vnic1
            3     30      30             vnic5
            4     15      15             vnic3
```

```
5     0       0             vnic4
6     0       0             vnic6
7     0       0             vnic7
```

이 예에서는 다른 VNIC가 고유의 해당 cos 값으로 설정됩니다. 이전 출력에 따라 vnic1의 cos 등록 정보가 2로 설정됩니다. ETSBW 필드에 있는 vnic1 클라이언트의 유효 대역폭 점유율은 15%로, ETSBW_EFFECT 필드에 표시된 피어에서 수신한 권장 값과 일치합니다. 이 예에서는 대역폭의 최대 점유율이 vnic5에 할당되었음도 보여줍니다. vnic4, vnic6 및 vnic7에 0%가 할당되었어도 클라이언트에 대역폭 점유율이 없음을 나타내는 것은 아닙니다. 다른 클라이언트가 해당하는 할당된 대역폭을 사용하는 경우 이러한 클라이언트는 대역폭을 수신하지 않습니다.

◆ ◆ ◆ 9 장

9 Oracle Solaris의 에지 가상 브리징

이 장에서는 EVB(에지 가상 브리징) 기능에 대해 설명합니다. EVB는 또한 다음 장에 설명된 정보 교환 기능을 확장합니다.

- 7 장, “LLDP를 사용하여 네트워크 연결 정보 교환”
- 8 장, “Oracle Solaris의 데이터 센터 브리징 기능을 사용하는 작업”

EVB를 사용하여 시스템의 가상 링크와 관련된 정보를 교환할 수 있습니다. 다음 항목을 다룹니다.

에지 가상 브리징 개요

에지 가상 브리징은 호스트가 외부 스위치와 가상 링크 정보를 교환하기 위해 사용하는 변화하는 IEEE 표준입니다. EVB를 사용하면 DCB 기능이 제공하는 물리적 링크 대역폭 점유율 또는 우선 순위 정의 이상의 더 많은 가상 링크 구성에 대한 정보를 네트워크에서 공개할 수 있습니다.

일반적으로 EVB는 다음 작업에 사용할 수 있습니다.

- 외부 브릿지 포트의 반사 중계 사용. 140 페이지 “반사 중계 기능”을 참조하십시오.
- 브릿지의 가상 포트 구성 자동화. 140 페이지 “브릿지의 가상 포트 구성 자동화”를 참조하십시오.

EVB 방식을 이해하려면 EVB와 함께 사용되는 다음 용어에 유의하십시오.

- **스테이션**은 시스템 또는 호스트를 가리킵니다.
- **브릿지**는 스테이션에 연결된 외부 스위치를 가리킵니다.
- **VSI(가상 스테이션 인스턴스)**는 스테이션에 구성된 VNIC를 가리킵니다.
- **VM(가상 머신)**은 시스템의 영역과 Oracle VM VirtualBox, 기타 소프트웨어 구현 머신을 가리키는 일반 용어입니다.

다음 절에서 EVB 기능에 대해 자세히 설명합니다.

반사 중계 기능

네트워크 가상화를 사용하면 하나의 물리적 NIC에서 여러 VNIC(가상 네트워크 인터페이스 카드)를 사용하여 스테이션을 구성할 수 있습니다. VNIC는 스테이션의 가상 머신에 지정됩니다. 이 설정에서는 스테이션을 유지해야 하는 패킷을 사용하지 않고 가상 머신 사이에서 통신이 이루어질 수 있습니다. 대신, 패킷은 물리적 링크의 가상 스위치에 의해 하나의 VM에서 다른 VM으로 경로가 지정됩니다. VNIC, 가상 스위치 및 가상 머신이 포함된 시스템 구성에 대한 설명은 **Oracle Solaris 11.1에서 가상 네트워크 사용**의 "네트워크 가상화의 구성 요소"를 참조하십시오.

어떤 경우에는 VM 간 내부 통신에 외부 브릿지를 사용해야 할 수도 있습니다. 예를 들어 외부 브릿지에 구성된 액세스 제어 목록(ACL)에 내부 통신이 종속되어야 할 수 있습니다. 따라서 시작 VM의 패킷이 외부 브릿지에 대한 포트를 통해 스테이션을 종료합니다. 그런 다음 해당 패킷은 다시 브릿지에서 수신 VM의 스테이션으로 송신됩니다.

기본적으로 브릿지는 패킷을 수신한 해당 포트에서 패킷을 송신할 수 없습니다. 따라서 외부 브릿지를 사용하는 VM 간 통신의 경우 브릿지에 반사 중계 기능이 있어야 합니다. 브릿지는 이 기능을 사용하여 송신 VM에서 패킷을 수신한 해당 링크의 수신 VM으로 다시 패킷을 중계할 수 있습니다.

EVB는 새로운 구성별 LLDP TLV 단위를 정의하여 네트워크 피어에 반사 중계 기능에 대한 정보를 제공하고 EVB TLV 단위는 해당 정보의 매개체 역할을 합니다. 정보 교환은 반사 중계가 브릿지에서 지원되는 경우 우선 반사 중계를 사용으로 설정하도록 브릿지에 요청하는 스테이션으로 구성됩니다. 브릿지는 해당 기능이 지원되는 경우 기능을 사용으로 설정하고 해당 기능을 요청한 호스트에 이를 알립니다. 브릿지에서 반사 중계를 지원하지 않는 경우 사용 안함 상태가 스테이션으로 다시 송신됩니다. 이 경우 가상 머신 간 통신에서 간단히 물리적 링크의 가상 스위치를 사용합니다.

브릿지의 가상 포트 구성 자동화

LLDP 및 DCBX를 사용하면 스테이션이 가장 가까운 브릿지와 구성 정보를 교환할 수 있습니다. 이러한 교환을 통해 브릿지는 예를 들어, 스테이션에서 트래픽 클래스에 대해 정의된 우선 순위를 감지할 수 있습니다. 이러한 정보를 기반으로 해당 802.1p 우선 순위 값에 따라 패킷을 처리하도록 브릿지가 자동 구성됩니다.

정보 교환 및 자동 구성을 사용하지 않을 경우 스테이션과 별도로 트래픽 클래스 우선 순위에 대한 스테이션 구성을 미러링하도록 브릿지를 수동 구성해야 합니다. 수동 구성에서는 브릿지를 잘못 구성할 위험이 있으며 스테이션 및 브릿지 간 불일치가 발생할 수 있습니다.

EVB를 사용하면 스테이션의 VSI에 대한 정보가 포함되도록 교환 방식이 브릿지로 확장됩니다. 이런 식으로 VNIC 구성이 브릿지로 확장될 수 있습니다. 예를 들어 특정

대역폭 제한을 사용하여 VNIC를 구성했다고 가정하겠습니다. EVB를 사용하면 브릿지에서 해당 VNIC를 대상으로 하는 패킷에 대해 대역폭 제한을 강제 적용할 수 있습니다.

VSI 정보 교환의 EVB 구성 요소

다음 EVB 구성 요소를 사용하여 스테이션에서 VSI 정보를 브릿지에 공개할 수 있습니다.

- **VSI 프로파일**은 특정 VNIC에 대해 구성된 링크 등록 정보로 구성됩니다. 따라서 스테이션은 구성된 VNIC 수만큼 VSI 프로파일을 보유할 수 있습니다.
- VSI 식별자는 VSI 프로파일을 고유하게 식별하는 **VSI 유형 ID** 및 **VSI 버전 ID** 쌍으로 구성됩니다.
- **VSI 관리자**는 특정 VNIC 등록 정보 세트로 VSI 유형 ID-VSI 버전 ID 식별자를 매핑하여 스테이션에 있는 여러 VSI 프로파일을 관리합니다.
- **VSI 관리자 ID**는 특정 VSI 유형 ID - VSI 버전 쌍과 관련된 VSI 관리자를 식별합니다. VSI 관리자 ID는 IPv6 주소로 표시됩니다.

결합된 VSI 관리자 ID, VSI 유형 ID 및 VSI 버전은 특정 VNIC의 등록 정보 세트를 식별하는 튜플을 구성합니다.

VSI 정보는 VDP(VSI 검색 및 구성 프로토콜)를 사용하여 교환되며 VDP TLV 단위는 해당 정보의 매개체 역할을 합니다. 브릿지는 스테이션에서 VDP TLV 단위를 수신합니다. 그런 다음 브릿지는 TLV 단위에 포함된 튜플을 사용하여 VSI와 연결된 등록 정보 세트를 가져옵니다. 브릿지는 VSI 프로파일 또는 유형 등록 정보를 가져온 후 해당 VSI 패킷에 등록 정보 구성을 적용할 수 있습니다.

VSI 정보를 브릿지에 공개하려면 다음 요구 사항이 충족되어야 합니다.

- 스테이션은 VSI 검색 프로토콜 요청을 송신하는 경우 사용할 VSI 관리자 ID를 알고 있어야 합니다.
- 스테이션에서 송신한 VSI 관리자 ID를 브릿지에서 인식하고 지원해야 합니다.

Oracle Solaris의 EVB 지원

VSI 프로파일 정의 예를 들어, 프로파일에 포함되어야 하는 특정 등록 정보를 위해 현재 정의된 표준은 없습니다. 또한 VSI 유형 정의는 일반적으로 판매자별로 다른 VSI 관리자 ID와 밀접하게 연결되어 있습니다.

Oracle Solaris는 3바이트 인코딩, `oracle_v1`을 사용하여 VSI 관리자를 정의합니다. 이 VSI 관리자는 다음 데이터 링크 등록 정보를 지원합니다.

- 대역폭 제한
- 기본 링크의 링크 속도
- 트래픽 클래스

- VNIC의 MTU(최대 전송 단위)

`oracle_v1` 인코딩은 다음과 같이 정의됩니다.

비트	등록 정보
0-4	**링크 대역폭 제한**
	00000-10100: 5%씩 증가하는 0-100%의 링크 속도
	나머지: 예약됨
5–7	**링크 속도**
	000 - 알 수 없음
	001 - 10Mbps
	010 - 100Mbps
	011 - 1Gbps
	100 - 10Gbps
	101 - 40Gbps
	110 - 100Gbps
	111 - 예약됨
8–12	**예약됨**
13-15	**트래픽 클래스(0-7)**
16-17	**링크 MTU**
	00 - 1500바이트
	01 - 9000바이트
	10 - 사용자 정의
	11 - 예약됨

3바이트 인코딩은 Oracle Solaris의 VSI 유형 ID로 직접 사용됩니다.

따라서 Oracle Solaris에서는 브릿지에 공개된 튜플이 Oracle VSI 관리자와 결합된 VSI 유형 ID-VSI 버전 ID 쌍입니다. VSI 정보 교환 방식은 141 페이지 “VSI 정보 교환의 EVB 구성 요소”에 설명된 프로세스와 동일한 프로세스를 따릅니다. 브릿지는 Oracle VSI 관리자를 인식하도록 구성됩니다. 그런 다음 브릿지는 Oracle VSI 관리자 ID 및 결합된 VSI 유형 ID-VSI 버전 ID를 사용하여 VSI 프로파일과 연결된 등록 정보 세트를 가져옵니다. 브릿지는 등록 정보를 가져온 후 해당 VNIC 패킷에 등록 정보 구성을 적용할 수 있습니다.

Oracle 구성별 OUI TLV 단위는 VSI 관리자 ID TLV 전송 후에 송신됩니다. OUI TLV는 사용하는 VSI 관리자 ID에 사용된 인코딩이 있는 경우 해당 인코딩을 표시합니다. 브릿지에서 Oracle 정의 VSI 관리자 ID를 인식하면 브릿지가 요청 스테이션에 응답하는 경우 해당 TLV 단위를 포함시킵니다. 브릿지의 응답에 Oracle 특정 TLV 단위가 없으면 해당 스위치에서 Oracle VSI 관리자가 인식되지도 않고 지원되지도 않음을 나타냅니다.

EVB 관련 데이터 링크 등록 정보

다음은 EVB와 관련된 구성 가능한 데이터 링크 등록 정보 목록입니다.

- `vsi-mgrid`는 물리적 링크 또는 VNIC에 대해 설정되는 VSI 관리자 ID를 지정합니다. Oracle Solaris에서는 이 등록 정보가 기본 VSI 관리자 ID인 `ORACLE_VSIMGR_V1`과 연결됩니다.

  IPv6 주소를 사용하려는 경우 VSI 유형 ID 및 VSI 버전 ID도 정의해야 합니다. 그렇지 않으면 Oracle Solaris에서 튜플을 인식하지 않습니다. 또한 VSI 유형 ID-VSI 버전 ID/VSI 관리자 ID 튜플에 해당하는 데이터 링크 등록 정보도 수동으로 구성해야 합니다.

  EVB 사용 시 기본 Oracle VSI 관리자 ID를 사용해야 할 수도 있습니다. 이런 식으로 Oracle VSI 관리자는 스테이션 VSI 프로파일에 대해 VSI 유형 ID 및 VSI 버전 ID를 자동으로 생성할 수 있습니다.
- `vsi-mgrid-enc`는 VSI 관리자 ID와 연결된 인코딩을 표시합니다. 기본적으로 이 등록 정보는 `oracle_v1`로 설정됩니다. `oracle_v1`을 VSI 관리자 ID와 연결하지 않으려면 이 등록 정보를 `none`으로 설정하십시오.
- `vsi-typeid`는 VSI 유형 ID를 지정합니다. VSI 유형 ID는 VSI 프로파일과 연결되는 VSI 버전 ID와 쌍을 이룹니다. `vsi-mgrid` 및 `vsi-mgrid-enc`의 기본값을 사용하는 경우 3바이트 값이 자동으로 생성됩니다. 그렇지 않은 경우 이 등록 정보의 값을 명시적으로 지정해야 합니다.
- `vsi-vers`는 VSI 버전 ID를 지정합니다. VSI 버전 ID는 VSI 프로파일과 연결되는 VSI 유형 ID와 쌍을 이룹니다. `vsi-mgrid` 및 `vsi-mgrid-enc`의 기본값을 사용하는 경우 1바이트 값이 자동으로 생성됩니다. 그렇지 않은 경우 이 등록 정보의 값을 명시적으로 지정해야 합니다.

주 – 이러한 모든 등록 정보는 모든 VNIC에서 수동으로 구성할 수 있습니다. 그러나 물리적 링크에서는 `vsi-mgrid` 및 `vsi-mgrid-enc` 등록 정보만 구성할 수 있습니다.

이전 목록의 등록 정보 외에도 시스템에서 작동하는 실제 EVB 구성에 대한 정보를 제공하는 읽기 전용 등록 정보는 다음과 같습니다.

- `vsi-mgrid-effective`는 가상 링크 또는 VNIC의 VSI 관리자 ID를 지정합니다.
- `vsi-mgrid-enc-effective`는 가상 링크 또는 VNIC에 사용되고 VSI 관리자 ID의 기초가 되는 VSI 관리자 ID 인코딩을 참조합니다.

- `vsi-typeid-effective`는 가상 링크 또는 VNIC의 유효한 VSI 유형 ID를 지정합니다.
- `vsi-vers-effective`는 링크의 유효한 VSI 버전을 지정합니다.

스테이션에서 EVB 사용

스테이션에서 EVB를 사용하려면 EVB 패키지를 설치해야 합니다. 다음 명령을 입력합니다.

```
# pkg install evb
```

패키지 설치 후 자동으로 사용으로 설정되는 기본 EVB 구성을 승인해야 할 수도 있습니다. EVB 구성은 EVB를 사용하는 데 필요한 Oracle VSI 관리자 사용을 기반으로 합니다. 기본 EVB 구성을 승인하면 스테이션에서 바로 VSI 정보를 스테이션에 구성된 VNIC에 대한 브릿지와 교환할 수 있습니다.

다음 예는 물리적 링크의 EVB 관련 등록 정보를 보여줍니다.

```
# dladm show-linkprop -p vsi-mgrid,vsi-mgrid-enc
LINK     PROPERTY            PERM VALUE        DEFAULT      POSSIBLE
net4     vsi-mgrid           rw   --           ::           --
net4     vsi-mgrid-enc       rw   --           oracle_v1    none,oracle_v1
```

출력에는 Oracle Solaris 11의 EVB 기본 구성이 표시됩니다. Oracle VSI 관리자는 `oracle_v1` 인코딩을 사용하여 Oracle VSI 관리자에서 인식되고 지원되는 VSI 및 해당 데이터 링크 등록 정보를 관리합니다.

기본 구성을 사용하지 않으려면 인코딩을 `none`으로 변경하십시오.

```
# dladm set-linkprop -p vsi-mgrid-enc=none net4
```

그런 다음 VSI 관리자 ID로 사용할 IPv6 주소를 수동으로 제공하고 VSI 유형 ID, 모든 다른 EVB 관련 구성 요소 및 해당 등록 정보를 정의해야 합니다.

다음 예는 VSI 또는 VNIC의 EVB 관련 등록 정보를 보여줍니다.

```
# dladm show-linkprop vnic0
LINK     PROPERTY                PERM VALUE      DEFAULT  POSSIBLE
...
vnic0    vsi-typeid              rw   --         --       --
vnic0    vsi-typeid-effective    r-   65684      --       --
vnic0    vsi-vers                rw   --         --       --
vnic0    vsi-vers-effective      r-   0          --       --
vnic0    vsi-mgrid               rw   --         --       --
vnic0    vsi-mgrid-effective     r-   ::         --       --
vnic0    vsi-mgrid-enc-effective r-   oracle_v1  --       --
...
```

출력에는 Oracle VSI 관리자 사용을 기반으로 하는 값이 표시됩니다. 해당 VSI 관리자 ID의 VSI 유효 인코딩은 `oracle_v1`입니다. 유형 ID `65684`는 자동으로 생성되며 `vnic0`에 대해 유효합니다.

다음 예는 스테이션에 EVB가 사용으로 설정된 경우 물리적 이더넷 링크의 VDP 상태에 대한 정보를 보여줍니다. 단일 링크에 대한 정보만 표시하려면 명령에 해당 링크를 지정하십시오. 그렇지 않은 경우 모든 이더넷 링크에 대한 VDP 정보가 표시됩니다.

```
# dladm show-ether -P vdb
VSI     LINK   VSIID             VSI-TYPEID VSI-STATE   CMD-PENDING
vnic0   net4   2:8:20:2c:ed:f3   65684/0    TIMEDOUT    NONE
vnic1   net4   2:8:20:df:73:77   65684/0    TIMEDOUT    NONE
```

출력에는 net4 링크에 대해 VSI 2개가 구성되어 있음이 표시되어 있습니다. 해당 특정 VSI ID는 해당 MAC 주소를 참조합니다. vsi-mgrid의 기본값에 따라 두 VSI에 동일한 VSI 유형 ID, 65684가 있습니다.

발신 또는 수신 VDP 패킷에 대한 통계를 가져오려면 다음 명령을 사용하십시오.

```
# dlstat show-ether -P vdb
```

◆ ◆ ◆ 10장

10 통합 로드 밸런서(개요)

이 장에서는 Oracle Solaris의 기능, ILB(통합 로드 밸런서)에 대해 설명합니다. ILB는 SPARC 및 x86 기반 시스템에 설치된 Oracle Solaris에 대한 계층 3 및 계층 4 로드 균형 조정 기능을 제공합니다. ILB는 클라이언트의 수신 요청을 가로채서 로드 균형 조정 규칙을 기반으로 요청을 처리할 백엔드 서버를 결정한 다음 선택한 서버로 요청을 전달합니다. ILB는 선택적 상태 검사를 수행하고 선택한 서버가 수신 요청을 처리할 수 있는지 여부를 확인할 수 있도록 로드 균형 조정 알고리즘에 대한 데이터를 제공합니다. ILB는 위와 같은 기능을 수행하여 서버에 지정된 작업 부하를 여러 서버로 분산합니다. 이를 통해 안정성이 향상되고 응답 시간이 최소화되며 서버의 성능이 전반적으로 개선될 수 있습니다.

다음 내용으로 구성되어 있습니다.

ILB 기능

ILB의 주요 기능은 다음과 같습니다.

- IPv4와 IPv6에 대한 Stateless DSR(Direct Server Return) 및 NAT(네트워크 주소 변환) 작동 모드를 지원합니다.
- CLI(명령줄 인터페이스)를 통해 ILB 관리를 수행할 수 있습니다.
- 상태 검사를 통한 서버 모니터링 기능을 제공합니다.

다음 목록에서는 ILB의 추가 기능에 대해 설명합니다.

- **클라이언트의 VIP(가상 IP) 주소 핑 가능** – ILB는 클라이언트의 가상 서비스 IP 주소에 대한 ICMP(인터넷 제어 메시지 프로토콜) 에코 요청에 응답할 수 있습니다. ILB는 DSR 및 NAT 작동 모드에 이 기능을 제공합니다.
- **서비스 인터럽트 없이 서버 그룹에서 서버 추가 및 제거 가능** – 기존에 설정된 백엔드 서버와의 연결에 대한 인터럽트 없이 동적으로 서버 그룹에서 서버를 추가 및 제거할 수 있습니다. ILB는 NAT 작동 모드에 이 기능을 제공합니다.
- **세션 지속성(고착성) 구성 가능** – 많은 응용프로그램에서 동일한 클라이언트로부터 시작된 일련의 연결이나 패킷 또는 둘 모두가 동일한 백엔드 서버로 전송되어야 합니다. `ilbadm create-rule` 하위 명령에서 `-p` 옵션을 사용하고 `pmask`를 지정하여 가상 서비스에 대한 세션 지속성(소스 주소 지속성)을 구성할 수 있습니다. 자세한 내용은 177 페이지 "ILB 규칙 생성 방법"을 참조하십시오. 지속 매핑이 생성되면 클라이언트의 소스 IP 주소와 일치하는 가상 서비스를 대상으로 하는 연결이나 패킷 또는 둘 모두에 대한 후속 요청이 동일한 백엔드 서버로 전달됩니다. CIDR(Classless Inter-Domain Routing) 표기법의 접두어 길이는 `0-32`의 값(IPv4의 경우) 및 `0-128`의 값(IPv6의 경우)입니다. 세션 지속성에 대한 지원은 DSR 및 NAT 작동 모드에 대해 제공됩니다.
- **연결 드레이닝 수행 가능** – ILB는 NAT 기반 가상 서비스의 서버에 대해서만 이 기능을 지원합니다. 이 기능은 새 연결이 사용 안함으로 설정된 서버로 전송되지 않도록 합니다. 이 기능은 활성 연결 또는 세션에 장애를 일으키지 않고 서버를 종료하는 데 유용합니다. 서버에 대한 기존 연결은 계속 작동합니다. 해당 서버에 대한 모든 연결이 종료되면 유지 관리할 서버를 종료할 수 있습니다. 서버가 요청을 처리할 준비가 되면 로드 밸런서가 새 연결을 전달할 수 있도록 서버가 사용으로 설정됩니다. 이 기능을 사용하면 활성 연결 또는 세션에 장애를 일으키지 않고 유지 관리할 서버를 종료할 수 있습니다.
- **TCP 및 UDP 포트의 로드 균형 조정 가능** – ILB는 각 포트에 대해 명시적 규칙을 설정할 필요 없이 여러 서버 세트에 있는 지정된 IP 주소의 모든 포트에 대해 로드 균형을 조정할 수 있습니다. ILB는 DSR 및 NAT 작동 모드에 이 기능을 제공합니다.
- **동일한 서버 그룹 내 가상 서비스에 대해 독립 포트 지정 가능** – 이 기능을 사용하면 NAT 작동 모드인 동일한 서버 그룹 내에 있는 여러 서버에 대해 서로 다른 대상 포트를 지정할 수 있습니다.
- **간단한 포트 범위의 로드 균형 조정 가능** – ILB는 지정된 서버 그룹의 VIP에서 포트 범위의 로드 균형을 조정할 수 있습니다. 다양한 백엔드 서버의 동일한 VIP에서 서로 다른 포트 범위의 로드 균형을 조정하여 편리하게 IP 주소를 절약할 수 있습니다. 또한 NAT 모드에 대해 세션 지속성이 사용으로 설정되어 있는 경우 ILB는 범위 내 여러 포트의 동일한 클라이언트 IP 주소에서 온 요청을 동일한 백엔드 서버로 전송합니다.
- **포트 범위 이동 및 축소 가능** – 포트 범위 이동 및 축소는 로드 균형 조정 규칙의 서버 포트 범위에 따라 결정됩니다. 따라서 서버 포트 범위가 VIP 포트 범위와 다른 경우 자동으로 포트 이동이 구현됩니다. 서버 포트 범위가 단일 포트인 경우 포트 축소가 구현됩니다. 이러한 기능은 NAT 작동 모드에 대해 제공됩니다.

ILB 구성 요소

ILB의 세 가지 주요 구성 요소는 다음과 같습니다.

- `ilbadm` CLI – 이 명령줄 인터페이스를 사용하여 로드 균형 조정 규칙을 구성하고 선택적 상태 검사를 수행하고 통계를 볼 수 있습니다.
- `libilb` 구성 라이브러리 – `ilbadm` 및 타사 응용프로그램에서 ILB 관리를 위해 `libilb`에 구현된 기능을 사용할 수 있습니다.
- `ilbd` 데몬 – 이 데몬은 다음 작업을 수행합니다.
  - 재부트 및 패키지 업데이트에서 지속 구성을 관리합니다.
  - 구성 정보를 처리하고 실행을 위해 ILB 커널 모듈로 전송하여 ILB 커널 모듈에 대한 순차적 액세스를 제공합니다.
  - 로드 분배가 제대로 조정되도록 상태 검사를 수행하고 결과를 ILB 커널 모듈로 전송합니다.

ILB 작동 모드

IPv4 및 IPv6의 경우 ILB는 단일 각 및 이중 각 토폴로지에서 Stateless DSR(Direct Server Return) 및 NAT(네트워크 주소 변환) 작동 모드를 지원합니다.

- Stateless DSR 토폴로지
- NAT 모드(Full-NAT 및 Half-NAT) 토폴로지

DSR(Direct Server Return) 토폴로지

DSR 모드에서 ILB는 백엔드 서버에 대한 수신 요청의 로드 균형을 조정하지만 서버에서 클라이언트로의 반환 트래픽이 로드 균형 조정을 무시할 수 있도록 합니다. 그러나 ILB가 백엔드 서버에 대한 라우터로 사용되도록 설정할 수도 있습니다. 이 경우 백엔드 서버에서 클라이언트로의 응답은 ILB를 실행 중인 시스템을 통해 경로 지정됩니다. DSR의 현재 ILB 구현에서는 TCP 연결 추적을 제공하지 않습니다(Stateless임을 의미함). Stateless DSR을 사용할 경우 ILB는 기본 통계를 제외하고는 처리된 패킷에 대한 상태 정보를 저장하지 않습니다. ILB가 이 모드에서 상태를 저장하지 않으므로 성능은 일반적인 IP 전달 성능과 유사합니다. 이 모드는 비연결 프로토콜에 가장 적합합니다.

이점:

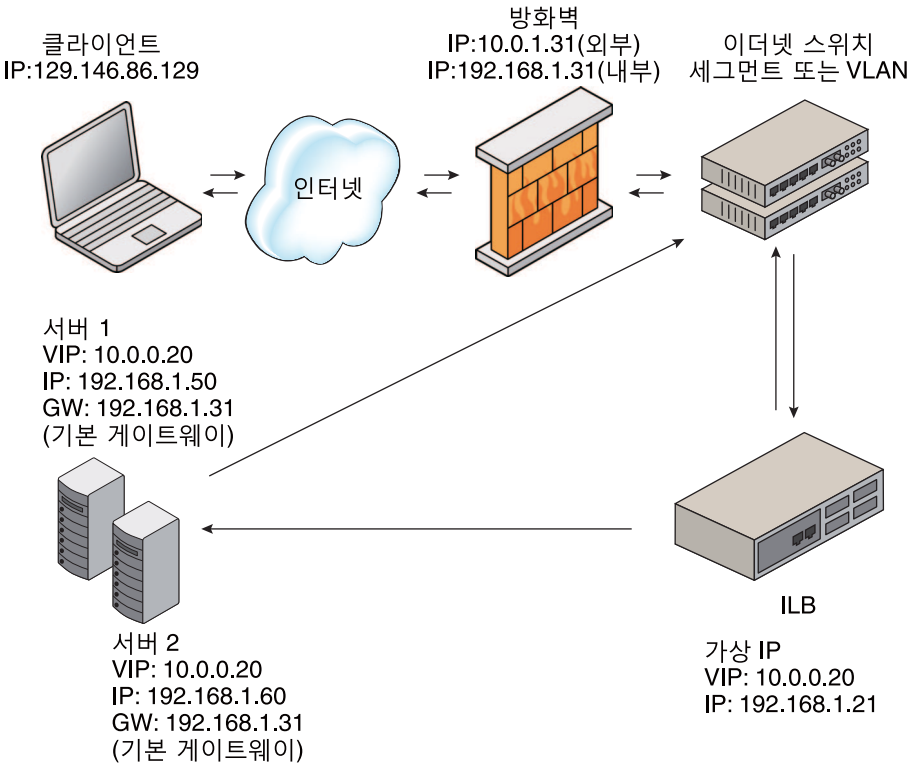
- 패킷의 대상 MAC 주소만 변경되고 서버가 직접 클라이언트에 응답하므로 NAT보다 성능이 뛰어납니다
- 서버 및 클라이언트 간 투명성이 완전합니다. 서버가 클라이언트 IP 주소에서 직접 연결을 확인하고 기본 게이트웨이를 통해 클라이언트에 응답합니다.

단점:

- 백엔드 서버가 고유 IP 주소(상태 검사용)와 가상 IP 주소(로드 균형이 조정된 트래픽용)에 모두 응답해야 합니다.
- 로드 밸런서가 연결 상태를 유지 관리하지 않으므로(Stateless임을 의미함) 서버를 추가하거나 제거하면 연결 장애가 발생합니다.

다음 그림은 DSR 토폴로지를 사용한 ILB 구현을 보여줍니다.

그림 10–1 DSR(Direct Server Return) 토폴로지



이 그림에서 두 백엔드 서버 모두 동일한 서브넷(192.168.1.0/24)에 ILB 상자로 있습니다. 서버는 라우터에도 연결되어 있으므로 ILB 상자에서 전달한 요청을 받은 후 클라이언트에 다시 직접 응답할 수 있습니다.

NAT(Network Address Translator) 토폴로지

ILB는 로드 균형 조정 기능에 대해서만 독립형 모드로 NAT를 사용합니다. 이 모드에서 ILB는 헤더 정보를 재작성하고 수신 및 송신 트래픽을 처리합니다. ILB는 Half-NAT 및 Full-NAT 모드 모두로 작동합니다. 그러나 Full-NAT는 소스 IP 주소도 재작성하여 모든

연결이 로드 밸런서에서 시작되는 것으로 서버에 표시되도록 합니다. NAT는 TCP 연결 추적을 제공합니다(Stateful임을 의미함). NAT 모드는 추가 보안을 제공하므로 HTTP(Hypertext Transfer Protocol) 또는 SSL(Secure Sockets Layer) 트래픽에 가장 적합합니다.

이점:

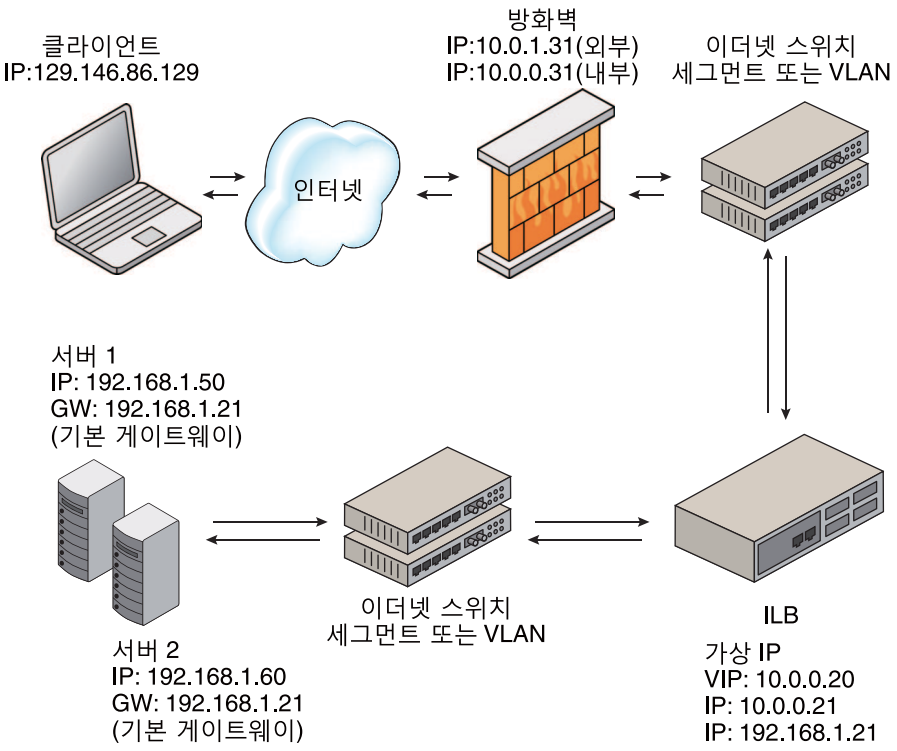
- 기본 게이트웨이가 로드 밸런서를 가리키도록 변경하여 모든 백엔드 서버에서 작동합니다.
- 로드 밸런서가 연결 상태를 유지 관리하므로 연결 장애 없이 서버를 추가하거나 제거할 수 있습니다.

단점:

- 처리 시 IP 헤더가 조작되고 서버가 로드 밸런서로 응답을 보내므로 DSR에 비해 성능이 떨어집니다.
- 모든 백엔드 서버에서 로드 밸런서를 기본 게이트웨이로 사용해야 합니다.

일반적인 NAT 토폴로지 구현은 아래 그림에 표시되어 있습니다.

그림 10–2 NAT(Network Address Translation) 토폴로지



이 경우 VIP에 대한 모든 요청이 ILB 상자를 경유하여 백엔드 서버로 전달됩니다. NAT의 경우 백엔드 서버의 모든 응답이 ILB 상자를 경유합니다.

주의 – ILB에서 구현된 NAT 코드 경로는 Oracle Solaris의 IP 필터 기능에서 구현된 코드 경로와 다릅니다. 이러한 코드 경로를 동시에 사용하지 **마십시오.**

Half-NAT 로드 균형 조정 토폴로지

Half-NAT 모드의 ILB 작동에서는 ILB가 패킷의 헤더에 대상 IP 주소만 재작성합니다. Half-NAT 구현을 사용하는 경우 서버가 상주하는 서브넷과 동일한 서브넷의 서비스 VIP(가상 IP) 주소에 연결할 수 없습니다. 다음 표는 클라이언트와 ILB 사이 그리고 ILB와 백엔드 서버 사이를 이동하는 패킷의 IP 주소를 보여줍니다.

표 10–1 서버 및 클라이언트가 다른 네트워크에 있는 경우 Half-NAT 구현의 요청 플로우와 응답 플로우

	요청 플로우	소스 IP 주소	대상 IP 주소
1.	클라이언트 –> ILB	클라이언트	ILB의 VIP

표 10–1 서버 및 클라이언트가 다른 네트워크에 있는 경우 Half-NAT 구현의 요청 플로우와 응답 플로우 *(계속)*

2.	ILB –> 서버	클라이언트	서버
	응답 플로우		
3.	서버 –> ILB	서버	클라이언트
4.	ILB –> 클라이언트	ILB의 VIP	클라이언트

클라이언트 시스템을 서버와 동일한 네트워크에 연결하면 의도한 서버가 클라이언트에 직접 응답합니다. 4번째 단계가 발생하지 않습니다. 따라서 클라이언트에 대한 서버 응답의 소스 IP 주소가 잘못되었습니다. 클라이언트가 로드 밸런서로 연결 요청을 전송하면 의도한 서버에서 응답이 발생합니다. 그러면 클라이언트의 IP 스택이 정확하게 모든 응답을 삭제합니다.

이 경우 요청 플로우와 응답 플로우는 다음 표에 나타난 대로 진행됩니다.

표 10–2 서버 및 클라이언트가 동일한 네트워크에 있는 경우 Half-NAT 구현의 요청 플로우와 응답 플로우

	요청 플로우	소스 IP 주소	대상 IP 주소
1.	클라이언트 –> ILB	클라이언트	ILB의 VIP
2.	ILB –> 서버	클라이언트	서버
	응답 플로우		
3.	서버 –> 클라이언트	서버	클라이언트

Full-NAT 로드 균형 조정 토폴로지

Full-NAT 구현에서는 트래픽이 로드 밸런서를 양방향으로 통과하도록 소스 및 대상 IP 주소가 재작성됩니다. Full-NAT 토폴로지에서는 서버가 상주하는 서브넷과 동일한 서브넷에서 VIP에 연결할 수 있습니다.

다음 표는 Full-NAT 토폴로지를 사용하여 클라이언트와 ILB 사이 그리고 ILB와 백엔드 서버 사이를 이동하는 패킷의 IP 주소를 보여줍니다. ILB 상자를 사용하는 특별한 기본 경로가 서버에서 필요하지 않습니다. 그러나 Full-NAT 토폴로지에서는 관리자가 ILB에서 백엔드 서버와 통신하는 데 소스 주소로 사용할 하나의 IP 주소 또는 IP 주소 범위를 마련해 두어야 합니다. 사용되는 주소가 서브넷 C에 속한다고 가정하겠습니다. 이 시나리오에서는 ILB가 프록시로 작동합니다.

표 10–3 Full-NAT 구현의 요청 플로우와 응답 플로우

	요청 플로우	소스 IP 주소	대상 IP 주소
1.	클라이언트 –> ILB	클라이언트	ILB의 VIP

표 10–3 Full-NAT 구현의 요청 플로우와 응답 플로우 (계속)

2.	ILB –> 서버	로드 밸런서의 인터페이스 주소(서브넷 C)	서버
	응답 플로우		
3.	서버 –> ILB	서버	ILB의 인터페이스 주소(서브넷 C)
4.	ILB –> 클라이언트	ILB의 VIP	클라이언트

ILB 작동 방식

이 절에서는 ILB 작동, 클라이언트에서 VIP로의 요청 처리 방식, 백엔드 서버로의 요청 전달 방식 및 응답 처리 방식에 대해 설명합니다.

클라이언트-서버 패킷 처리:

1. ILB에서 클라이언트가 VIP 주소로 보낸 수신 요청을 받아 로드 균형 조정 규칙과 요청을 일치시킵니다.
2. ILB에서 일치하는 로드 균형 조정 규칙을 찾을 경우 로드 균형 조정 알고리즘을 사용하여 작동 모드에 따라 요청을 백엔드 서버로 전달합니다.
   - DSR 모드에서는 ILB가 수신 요청의 MAC 헤더를 선택된 백엔드 서버의 MAC 헤더로 바꿉니다.
   - Half-NAT 모드에서는 ILB가 수신 요청의 대상 IP 주소 및 전송 프로토콜 포트 번호를 선택된 백엔드 서버의 대상 IP 주소 및 전송 프로토콜 포트 번호로 바꿉니다.
   - Full-NAT 모드에서는 ILB가 수신 요청의 소스 IP 주소 및 전송 프로토콜 포트 번호를 로드 균형 조정 규칙의 NAT 소스 주소로 바꿉니다. ILB는 또한 수신 요청의 대상 IP 주소 및 전송 프로토콜 포트 번호를 선택된 백엔드 서버의 대상 IP 주소 및 전송 프로토콜 포트 번호로 바꿉니다.
3. ILB는 수정된 수신 요청을 선택된 백엔드 서버로 전달합니다.

서버-클라이언트 패킷 처리:

1. 백엔드 서버가 클라이언트의 수신 요청에 대한 응답으로 ILB에 응답을 보냅니다.
2. 백엔드 서버로부터 응답을 받은 후 ILB의 작업은 작동 모드를 기반으로 합니다.
   - DSR 모드에서는 백엔드 서버의 응답이 ILB를 무시하고 클라이언트로 직접 전송됩니다. 하지만 ILB가 백엔드 서버의 라우터로도 사용되는 경우에는 백엔드 서버가 클라이언트로 보낸 응답이 ILB를 실행하는 시스템을 통해 경로 지정됩니다.

- Half-NAT 모드 및 Full-NAT 모드에서는 ILB가 백엔드 서버의 응답을 수신 요청과 일치시키고 변경된 IP 주소 및 전송 프로토콜 포트 번호를 원래 수신 요청의 IP 주소 및 전송 프로토콜 포트 번호로 바꿉니다. 그런 다음 ILB는 응답을 클라이언트로 전달합니다.

ILB 알고리즘

ILB 알고리즘은 트래픽 분배를 제어하고 다양한 로드 분배 및 서버 선택 특성을 제공합니다. ILB는 두 가지 작동 모드에 대해 다음 알고리즘을 제공합니다.

- 라운드 로빈 – 라운드 로빈 알고리즘에서는 로드 밸런서가 회전을 기준으로 서버 그룹에 요청을 지정합니다. 서버에 요청이 지정되면 목록 끝으로 서버가 이동됩니다.
- *src IP* 해시 – 소스 IP 해시 메소드에서는 로드 밸런서가 수신 요청의 소스 IP 주소에 대한 해시 값을 기반으로 서버를 선택합니다.
- *src-IP, port* 해시 – 소스 IP, 포트 해시 메소드에서는 로드 밸런서가 수신 요청의 소스 IP 주소 및 소스 포트에 대한 해시 값을 기반으로 서버를 선택합니다.
- *src-IP, VIP* 해시 – 소스 IP, VIP 해시 메소드에서는 로드 밸런서가 수신 요청의 소스 IP 주소 및 대상 IP 주소에 대한 해시 값을 기반으로 서버를 선택합니다.

서비스 관리 기능

ILB는 SMF(서비스 관리 기능) 서비스 `svc:/network/loadbalancer/ilb:default`를 통해 관리됩니다. SMF 개요는 **Oracle Solaris 11.1에서 서비스 및 결함 관리**의 1 장, "서비스 관리(개요)"를 참조하십시오. SMF와 관련된 단계별 절차는 **Oracle Solaris 11.1에서 서비스 및 결함 관리**의 2 장, "서비스 관리(작업)"를 참조하십시오.

ILB 명령줄 인터페이스

ILB 명령줄 인터페이스는 `/usr/sbin/ilbadm` 디렉토리에 있습니다. CLI에는 로드 균형 조정 규칙, 서버 그룹 및 상태 검사를 구성하기 위한 하위 명령이 포함되어 있습니다. 통계를 표시하고 구성 상세 정보를 보기 위한 하위 명령도 포함되어 있습니다. 하위 명령은 다음 두 가지 범주로 구분될 수 있습니다.

- **구성 하위 명령** – 이러한 하위 명령을 통해 다음 작업을 수행할 수 있습니다.
  - 로드 균형 조정 규칙 생성 및 삭제
  - 로드 균형 조정 규칙 사용 및 사용 안함으로 설정
  - 서버 그룹 생성 및 삭제
  - 서버 그룹에서 서버 추가 및 제거
  - 백엔드 서버를 사용 및 사용 안함으로 설정
  - 로드 균형 조정 규칙 내 서버 그룹에 대한 서버 상태 검사 생성 및 삭제

주 – 구성 하위 명령을 관리하려면 권한이 필요합니다. 이러한 권한은 Oracle Solaris의 RBAC(역할 기반 액세스 제어)를 통해 얻을 수 있습니다. 적합한 역할을 생성하여 사용자에게 지정하려면 **Oracle Solaris 11.1 관리: 보안 서비스**의 "RBAC 초기 구성(작업 맵)"을 참조하십시오.

- **보기 하위 명령** – 이러한 하위 명령을 통해 다음 작업을 수행할 수 있습니다.
  - 구성된 로드 균형 조정 규칙, 서버 그룹 및 상태 검사 보기
  - 패킷 전달 통계 보기
  - NAT 연결 테이블 보기
  - 상태 검사 결과 보기
  - 세션 지속성 매핑 테이블 보기

주 – 보기 하위 명령을 관리하는 데는 권한이 필요하지 않습니다.

`ilbadm` 하위 명령 목록은 156 페이지 "ILB 명령 및 하위 명령"을 참조하십시오. `ilbadm` 하위 명령에 대한 자세한 내용은 `ilbadm`(1M) 매뉴얼 페이지를 참조하십시오.

ILB 명령 및 하위 명령

`ilbadm` 및 하위 명령을 사용하여 로드 균형 조정 규칙을 조작할 수 있습니다. `ilbadm` 하위 명령에 대한 자세한 내용은 `ilbadm`(1M) 매뉴얼 페이지를 참조하십시오.

표 10–4 로드 균형 조정 규칙 조작에 사용되는 ILB 하위 명령

ILB 하위 명령	설명
`ilbadm create-rule`	지정된 특성으로 `rule name`을 만듭니다.
`ilbadm show-rule`	지정된 규칙의 특성 또는 모든 규칙(지정된 규칙이 없을 경우)을 표시합니다.
`ilbadm delete-rule`	`rule name`과 관련된 모든 정보를 제거합니다. `rule name`이 존재하지 않을 경우 이 하위 명령이 실패합니다.
`ilbadm enable-rule`	명명된 규칙 또는 모든 규칙(지정된 이름이 없을 경우)을 사용으로 설정합니다.
`ilbadm disable-rule`	명명된 규칙 또는 모든 규칙(지정된 이름이 없을 경우)을 사용 안함으로 설정합니다.
`ilbadm show-statistics`	ILB 통계를 표시합니다. 예를 들어, 이 하위 명령에 `-t`를 사용할 경우 모든 헤더와 함께 시간 기록이 포함됩니다.

표 10–4 로드 균형 조정 규칙 조작에 사용되는 ILB 하위 명령 *(계속)*

ILB 하위 명령	설명
`ilbadm show-hc-result`	지정된 규칙 이름 `rule-name`과 연관된 서버에 대한 상태 검사 결과를 표시합니다. `rule-name`이 지정되지 않은 경우 모든 규칙에 대한 서버 상태 검사 결과가 표시됩니다.
`ilbadm show-nat`	NAT 테이블 정보를 표시합니다.
`ilbadm create-servergroup`	서버가 하나 이상 있는 서버 그룹을 만듭니다. `ilbadm add-server`를 사용하여 서버를 더 추가할 수 있습니다.
`ilbadm delete-servergroup`	서버 그룹을 삭제합니다.
`ilbadm show-servergroup`	특정 서버 그룹 또는 모든 서버 그룹(지정된 서버 그룹이 없을 경우)을 나열합니다.
`ilbadm enable-server`	사용 안함으로 설정된 서버를 사용으로 설정합니다.
`ilbadm disable-server`	지정된 서버를 사용 안함으로 설정합니다.
`ilbadm add-server`	지정된 서버를 서버 그룹에 추가합니다.
`ilbadm show-server`	명명된 규칙과 연관된 서버 또는 모든 서버(규칙 이름이 지정되지 않은 경우)를 표시합니다.
`ilbadm remove-server`	서버 그룹에서 서버를 하나 이상 제거합니다.
`ilbadm create-healthcheck`	규칙 설정에 사용할 수 있는 상태 검사 정보를 설정합니다.
`ilbadm show-healthcheck`	구성된 상태 검사에 대한 자세한 정보를 표시합니다.
`ilbadm delete-healthcheck`	상태 검사 정보를 삭제합니다.
`ilbadm show-persist`	세션 지속성 매핑 테이블을 표시합니다.
`ilbadm export-config` *filename*	`ilbadm import`를 사용하여 가져오기 작업에 적합한 형식으로 기존 ILB 구성 파일을 내보냅니다. *filename*이 지정되지 않은 경우 `ilbadm export`가 `stdout`에 기록합니다.
`ilbadm import-config -p` *filename*	파일을 가져오고 기존 ILB 구성을 가져온 파일의 내용으로 바꿉니다. *filename*이 지정되지 않은 경우 `ilbadm import`가 `stdin`에서 읽습니다.

◆ ◆ ◆ 11장

11 통합 로드 밸런서 구성

이 장에서는 ILB(통합 로드 밸런서) 설치에 대해 설명하고 단순 ILB 구성 설정에 대한 예를 제공합니다. 다음 내용으로 구성되어 있습니다.

ILB 설치

ILB는 커널과 userland의 두 부분으로 구성됩니다. 커널 부분은 Oracle Solaris 11 설치의 일부로 자동으로 설치됩니다. ILB의 userland 부분을 얻으려면 사용자가 `pkg install ilb` 명령을 사용하여 수동으로 `ilb` 패키지를 설치해야 합니다.

ILB 사용

이 절에서는 ILB를 사용으로 설정하는 데 사용되는 절차를 설명합니다.

▼ ILB를 사용으로 설정하는 방법

시작하기 전에 시스템의 RBAC(역할 기반 액세스 제어) 속성 파일에 다음 항목이 있는지 확인합니다. 항목이 없는 경우 수동으로 추가하십시오.

- 파일 이름: `/etc/security/auth_attr`
  - `solaris.network.ilb.config:::Network ILB Configuration::help=NetworkILBconf.html`

    - `solaris.network.ilb.enable:::Network ILB Enable Configuration::help=NetworkILBenable.html`
    - `solaris.smf.manage.ilb:::Manage Integrated Load Balancer Service States::help=SmfILBStates.html`
- 파일 이름: `/etc/security/prof_attr`
    - `Network ILB:::Manage ILB configuration via ilbadm:auths=solaris.network.ilb.config,solaris.network.ilb.enable;help=RtNetILB.htm`
    - 파일의 네트워크 관리 항목에 `solaris.smf.manage.ilb`가 있어야 합니다.
- 파일 이름: `/etc/user_attr`
    - `daemon::::auths=solaris.smf.manage.ilb,solaris.smf.modify.application`

ILB 구성 하위 명령에 필요한 사용자 권한 부여를 설정해야 합니다. 156 페이지 “ILB 명령 및 하위 명령”에 나열된 ILB 구성 하위 명령을 실행하려면 `solaris.network.ilb.config` RBAC 권한 부여가 있어야 합니다.

- 기존 사용자에 권한 부여를 지정하려면 **Oracle Solaris 11.1 관리: 보안 서비스**의 9 장, “역할 기반 액세스 제어 사용(작업)”을 참조하십시오.
- 또한 시스템에 새 사용자 계정을 만들 때 권한 부여를 제공할 수 있습니다.

  다음 예에서는 그룹 ID `10`, 사용자 ID `1210` 그리고 시스템에서 ILB를 관리하기 위한 권한 부여를 사용하여 `ilbadm` 사용자를 만듭니다.

  ```
  # useradd -g 10 -u 1210 -A solaris.network.ilb.config ilbadmin
  ```

  `useradd` 명령은 `/etc/passwd`, `/etc/shadow` 및 `/etc/user_attr` 파일에 새 사용자를 추가합니다. `-A` 옵션을 사용하면 사용자에게 권한을 지정합니다.

1 ILB 관리 권한 프로파일이 포함된 역할 또는 슈퍼 유저로 로그인합니다.

사용자가 만든 역할에 ILB 관리 권한 프로파일을 지정할 수 있습니다. 역할을 만들어 사용자에게 지정하려면 **Oracle Solaris 11.1 관리: 보안 서비스**의 “RBAC 초기 구성(작업 맵)”을 참조하십시오.

2 IPv4, IPv6 또는 둘 모두에서 적절한 전달 서비스를 사용으로 설정합니다.

성공하는 경우 이 명령은 출력을 생성하지 않습니다.

```
# ipadm set-prop -p forwarding=on ipv4
# ipadm set-prop -p forwarding=on ipv6
```

3 ILB 서비스를 사용으로 설정합니다.

```
# svcadm enable ilb
```

4 ILB 서비스가 사용으로 설정되었는지 확인합니다.

```
# svcs ilb
```

ILB 구성

이 절에서는 Half-NAT 토폴로지를 사용하여 두 서버 사이의 트래픽 로드 균형을 조정하기 위한 ILB 설정 단계에 대해 설명합니다. 149 페이지 “ILB 작동 모드”의 NAT 토폴로지 구현을 참조하십시오.

▼ ILB 구성 방법

1 ILB 관리 권한 프로파일이 포함된 역할 또는 슈퍼 유저로 로그인합니다.

사용자가 만든 역할에 ILB 관리 권한 프로파일을 지정할 수 있습니다. 역할을 만들어 사용자에게 지정하려면 **Oracle Solaris 11.1 관리: 보안 서비스**의 “RBAC 초기 구성(작업 맵)”을 참조하십시오.

2 백엔드 서버를 설정합니다.

백엔드 서버는 ILB를 이 시나리오의 기본 라우터로 사용하도록 설정됩니다. 이 작업은 두 서버 모두에서 다음 명령을 실행하여 수행할 수 있습니다.

```
# route add -p default 192.168.1.21
```

이 명령을 실행하면 두 서버 모두에서 서버 응용프로그램이 시작됩니다. 5000 포트에서 수신하는 TCP 응용프로그램이라고 가정합니다.

3 ILB의 서버 그룹을 설정합니다.

서버는 `192.168.1.50` 및 `192.169.1.60` 2개가 있습니다. 이러한 두 서버로 구성된 서버 그룹, `srvgrp1`은 다음 명령을 입력하여 만들 수 있습니다.

```
# ilbadm create-sg -s servers=192.168.1.50,192.168.1.60 srvgrp1
```

4 간단한 상태 검사, `hc-srvgrp1` 설정은 다음 명령을 입력하여 만들 수 있습니다.

간단한 TCP 레벨 상태 검사를 사용하여 서버 응용프로그램에 접근할 수 있는지 감지합니다. 이러한 검사는 60초마다 수행됩니다. 서버가 양호한지 확인하려면 최대 3번을, 각 시도마다 최대 3초 동안 대기하면서 수행합니다. 3번의 시도 모두 실패하는 경우 서버를 `dead`로 표시합니다.

```
# ilbadm create-hc -h hc-test=tcp,hc-timeout=3, \
hc-count=3,hc-inerval=60 hc-srvgrp1
```

5 다음 명령을 입력하여 ILB 규칙을 설정합니다.

이 규칙에서는 지속성(32비트 마스크 포함)이 사용됩니다. 그리고 로드 밸런스 알고리즘은 `round robin`입니다. 서버 그룹 `srvgrp1`이 사용되며 사용되는 상태 검사 방식은 `hc-srvgrp1`입니다. 규칙은 다음 명령을 입력하여 만들 수 있습니다.

```
# ilbadm create-rule -e -p -i vip=10.0.2.20,port=5000 -m \
 lbalg=rr,type=half-nat,pmask=32 \
-h hc-name=hc-srvgrp1 -o servergroup=srvgrp1 rule1_rr
```

ILB 사용 안함으로 설정

다음 섹션에서는 ILB를 사용 안함으로 설정하는 절차를 설명합니다.

▼ ILB를 사용 안함으로 설정하는 방법

1 **ILB 관리 권한 프로파일이 포함된 역할 또는 슈퍼 유저로 로그인합니다.**

사용자가 만든 역할에 ILB 관리 권한 프로파일을 지정할 수 있습니다. 역할을 만들어 사용자에게 지정하려면 **Oracle Solaris 11.1 관리: 보안 서비스**의 "RBAC 초기 구성(작업 맵)"을 참조하십시오.

2 **ILB 서비스를 사용 안함으로 설정합니다.**

```
# svcadm disable ilb
```

3 **ILB 서비스가 사용 안함으로 설정되었는지 확인합니다.**

```
# svcs ilb
```

구성 가져오기 및 내보내기

`ilbadm export` 하위 명령은 현재 ILB 구성을 사용자 지정 파일로 내보냅니다. 그런 다음 이 정보를 `ilbadm import` 하위 명령의 입력으로 사용할 수 있습니다.

별도로 구성을 유지하라고 지시하지 않는 한, `ilbadm import` 하위 명령은 기존 구성을 가져오기 전에 삭제합니다. 파일 이름을 생략하면 명령은 `stdin`에서 읽거나 `stdout`에 씁니다.

ILB 구성을 내보내려면 `export-config` 명령을 사용합니다. 다음 예는 `import` 하위 명령을 사용하여 가져오기에 적합한 형식으로 현재 구성을 `/var/tmp/ilb_config` 파일로 내보냅니다.

```
# ilbadm export-config /var/tmp/ilb_config
```

ILB 구성을 가져오려면 `import-config` 명령을 사용합니다. 다음 예는 `/var/tmp/ilb_config` 파일의 내용을 읽고 기존 구성을 대체합니다.

```
# ilbadm import-config /var/tmp/ilb_config
```

고가용성을 위한 ILB 구성(능동-수동 모드 전용)

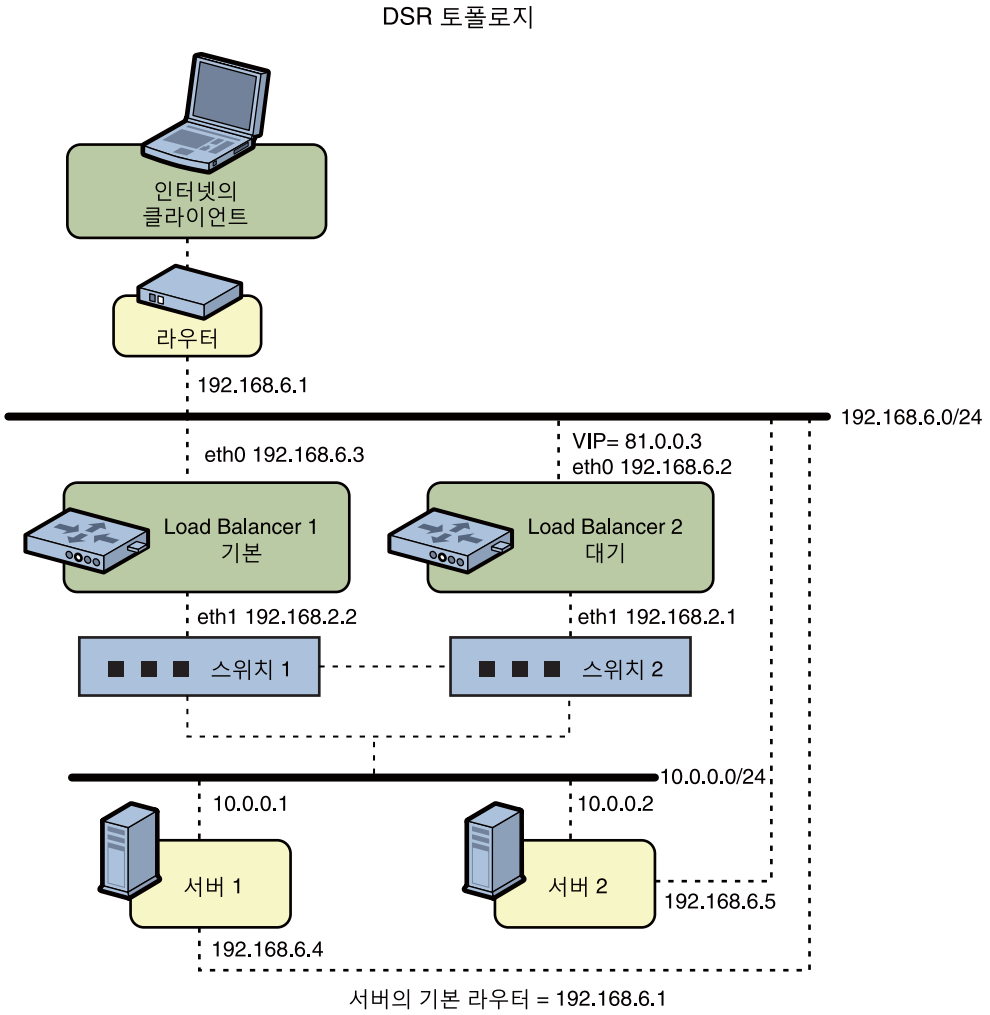
이 절에서는 DSR, Half-NAT 토폴로지를 사용한 ILB의 고가용성 구성에 대해 설명합니다.

DSR 토폴로지를 사용하여 고가용성 ILB 구성

이 절에서는 DSR 토폴로지를 사용하여 HA(고가용성)를 구현하도록 ILB 연결을 설정하는 방법을 설명합니다. 기본 로드 밸런서와 대기 로드 밸런서, 2개의 로드 밸런서를 설정해야 합니다. 기본 로드 밸런서가 실패하면 대기 로드 밸런서가 기본 로드 밸런서의 역할을 맡습니다.

다음 그림은 HA를 구현하도록 ILB 연결을 구성하기 위한 DSR 토폴로지를 보여줍니다.

그림 11–1 DSR 토폴로지를 사용하여 HA ILB 구성



Load Balancer의 모든 VIP는 서브넷 192.168.6.0/24를 대상으로 하는 인터페이스에 구성됩니다.

▼ DSR 토폴로지를 사용하여 고가용성을 구현하도록 ILB를 구성하는 방법

1 **ILB 관리 권한 프로파일이 포함된 역할 또는 슈퍼 유저로 로그인합니다.**

사용자가 만든 역할에 ILB 관리 권한 프로파일을 지정할 수 있습니다. 역할을 만들어 사용자에게 지정하려면 **Oracle Solaris 11.1 관리: 보안 서비스**의 "RBAC 초기 구성(작업 맵)"을 참조하십시오.

2 **기본 및 대기 로드 밸런서를 모두 구성합니다.**

```
# ilbadm create-servergroup -s server=10.0.0.1,10.0.0.2 sg1
# ilbadm create-rule -i vip=81.0.0.3,port=9001 \
-m lbalg=hash-ip-port,type=DSR -o servergroup=sg1 rule1
```

3 **모든 서버의 `lo0` 인터페이스에 VIP가 구성되었는지 확인합니다.**

```
Server1# ipadm create-addr -d -a 81.0.0.3/24 lo0
Server2# ipadm create-addr -d -a 81.0.0.3/24 lo0
```

4 **로드 밸런서 1이 기본 로드 밸런서로 작동하도록 구성합니다.**

```
LB1# dladm create-vnic -m vrrp -V 1 -A inet -l eth0 vnic1
LB1# vrrpadm create-router -V 1 -A inet -l eth0 -p 255 vrrp1
LB1# ipadm create-addr -d -a 81.0.0.3/24 vnic1
```

5 **로드 밸런서 2가 대기 로드 밸런서로 작동하도록 구성합니다.**

```
LB2# dladm create-vnic -m vrrp -V 1 -A inet -l eth0 vnic1
LB2# vrrpadm create-router -V 1 -A inet -l eth0 -p 100 vrrp1
LB2# ipadm create-addr -d -a 81.0.0.3/24 vincl
```

앞의 구성은 다음 실패 시나리오에 대한 보호를 제공합니다.

- 로드 밸런서 1이 실패하면 로드 밸런서 2가 기본 로드 밸런서가 됩니다. 로드 밸런서 2가 VIP `81.0.0.3`에 대한 주소 해석을 인수하고 클라이언트의 모든 패킷을 대상 IP 주소 `81.0.0.3`으로 처리합니다.

  로드 밸런서 1을 복구하면 로드 밸런서 2가 대기 모드로 돌아갑니다.
- 로드 밸런서 1의 인터페이스가 하나 또는 둘 모두 실패하는 경우 로드 밸런서 2가 기본 로드 밸런서 역할을 인수합니다. 로드 밸런서 2가 VIP `81.0.0.3`에 대한 주소 해석을 인수하고 클라이언트의 모든 패킷을 대상 IP 주소 `81.0.0.3`으로 처리합니다.

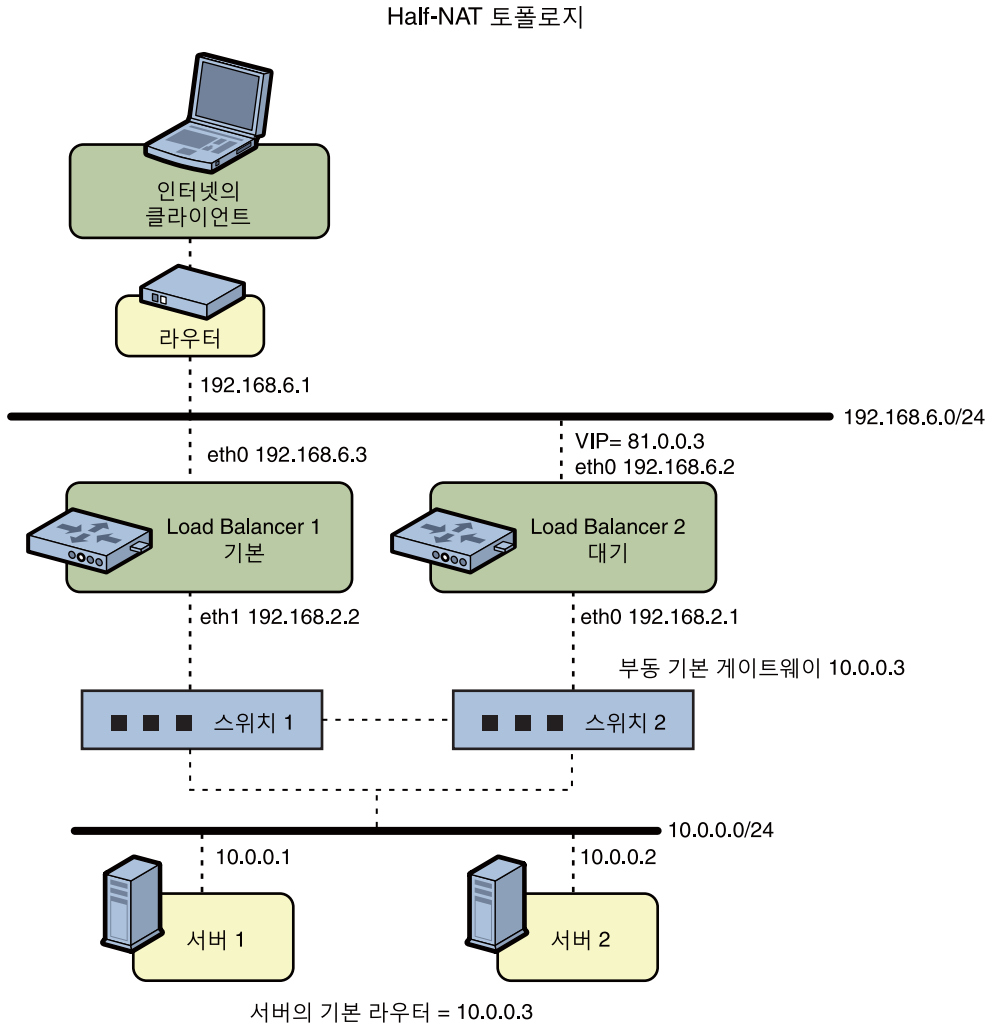
  로드 밸런서 1의 인터페이스가 둘 모두 양호해지면 로드 밸런서 2가 대기 모드로 돌아갑니다.

Half-NAT 토폴로지를 사용하여 고가용성 ILB 구성

이 절에서는 Half-NAT 토폴로지를 사용하여 HA(고가용성)를 구현하도록 ILB 연결을 설정하는 방법을 설명합니다. 기본 로드 밸런서와 대기 로드 밸런서, 2개의 로드 밸런서를 설정해야 합니다. 기본 로드 밸런서가 실패하면 대기 로드 밸런서가 기본 로드 밸런서의 역할을 맡습니다.

다음 그림은 HA를 구현하도록 ILB 연결을 구성하기 위한 Half-NAT 토폴로지를 보여줍니다.

그림 11–2 Half-NAT 토폴로지를 사용하여 HA ILB 구성



Load Balancer의 모든 VIP는 서브넷 192.168.6.0/24를 대상으로 하는 인터페이스에 구성됩니다.

▼ Half-NAT 토폴로지를 사용하여 고가용성을 구현하도록 ILB를 구성하는 방법

1 ILB 관리 권한 프로파일이 포함된 역할 또는 슈퍼 유저로 로그인합니다.

사용자가 만든 역할에 ILB 관리 권한 프로파일을 지정할 수 있습니다. 역할을 만들어 사용자에게 지정하려면 **Oracle Solaris 11.1 관리: 보안 서비스**의 "RBAC 초기 구성(작업 맵)"을 참조하십시오.

2 기본 및 대기 로드 밸런서를 모두 구성합니다.

```
# ilbadm create servergroup -s server=10.0.0.1,10.0.0.2 sg1
# ilbadm create-rule -ep -i vip=81.0.0.3,port=9001-9006,protocol=udp \
-m lbalg=roundrobin,type=HALF-NAT,pmask=24 \
-h hc-name=hc1,hc-port=9006 \
-t conn-drain=70,nat-timeout=70,persist-timeout=70 -o servergroup=sg1 rule1
```

3 로드 밸런서 1이 기본 로드 밸런서로 작동하도록 구성합니다.

```
LB1# dladm create-vnic -m vrrp -V 1 -A inet -l eth0 vnic1
LB1# ipadm create-addr -d -a 81.0.0.3/24 vnic1
LB1# vrrpadm create-router -V 1 -A inet -l eth0 -p 255 vrrp1
LB1# dladm create-vnic -m vrrp -V 2 -A inet -l eth1 vnic2
LB1# ipadm create-addr -d -a 10.0.0.3/24 vnic2
LB1# vrrpadm create-router -V 2 -A inet -l eth1 -p 255 vrrp2
```

4 로드 밸런서 2가 대기 로드 밸런서로 작동하도록 구성합니다.

```
LB2# dladm create-vnic -m vrrp -V 1 -A inet -l eth0 vnic1
LB2# ipadm create-addr -d -a 81.0.0.3/24 vnic1
LB2# vrrpadm create-router -V 1 -A inet -l eth0 -p 100 vrrp1
LB2# dladm create-vnic -m vrrp -V 2 -A inet -l eth1 vnic2
LB2# ipadm create-addr -d -a 10.0.0.3/24 vnic2
LB2# vrrpadm create-router -V 2 -A inet -l eth1 -p 100 vrrp2
```

5 부동 기본 게이트웨이의 IP 주소를 두 서버에 추가합니다.

```
# route add net 192.168.6.0/24 10.0.0.3
```

앞의 구성은 다음 실패 시나리오에 대한 보호를 제공합니다.

- 로드 밸런서 1이 실패하면 로드 밸런서 2가 기본 로드 밸런서가 됩니다. 로드 밸런서 2가 VIP 81.0.0.3에 대한 주소 해석을 인수하고 클라이언트의 모든 패킷을 대상 IP 주소 81.0.0.3으로 처리합니다. 로드 밸런서 2도 부동 게이트웨이 주소 10.0.0.3으로 송신한 모든 패킷을 처리합니다.

  로드 밸런서 1을 복구하면 로드 밸런서 2가 대기 모드로 돌아갑니다.
- 로드 밸런서 1의 인터페이스가 하나 또는 둘 모두 실패하는 경우 로드 밸런서 2가 기본 로드 밸런서 역할을 인수합니다. 로드 밸런서 2가 VIP 81.0.0.3에 대한 주소 해석을 인수하고 클라이언트의 모든 패킷을 대상 IP 주소 81.0.0.3으로 처리합니다. 로드 밸런서 2도 부동 게이트웨이 주소 10.0.0.3으로 송신한 모든 패킷을 처리합니다.

  로드 밸런서 1의 인터페이스가 둘 모두 양호해지면 로드 밸런서 2가 대기 모드로 돌아갑니다.

주 – 현재 ILB 구현은 기본 및 대기 로드 밸런서를 동기화하지 않습니다. 기본 로드 밸런서가 실패하고 대기 로드 밸런서가 역할을 인수하는 경우 기존 연결이 실패합니다. 그러나 동기화 없는 HA는 기본 로드 밸런서가 실패한 상황에서도 여전히 유용합니다.

◆ ◆ ◆ 12장

12 통합 로드 밸런서 관리

이 장에서는 서버 그룹 생성 또는 제거 같은 ILB 서버 그룹을 관리하고 서버 그룹에서 서버를 추가, 삭제하거나 다시 사용으로 설정하거나 사용 안함으로 설정하는 것 같은 백엔드 서버를 관리하고 규칙을 생성 및 삭제하고 통계를 표시하는 절차를 설명합니다.

다음 내용으로 구성되어 있습니다.

ILB 서버 그룹 관리

이 절에서는 `ilbadm` 명령을 사용하여 ILB 서버 그룹을 생성, 삭제 및 나열하는 방법에 대해 설명합니다.

▼ ILB 서버 그룹 생성 방법

1 만들려는 서버 그룹의 이름을 선택합니다.

2 서버 그룹에 포함될 서버를 선택합니다.

서버는 호스트 이름 또는 IP 주소 및 선택적 포트로 지정할 수 있습니다.

3 서버 그룹을 만듭니다.

```
# ilbadm create-servergroup -s servers= \
server1,server2,server3  servergroup
```

예 12–1 ILB 서버 그룹 만들기

다음 예에서는 세 개의 서버로 구성된 webgroup이라는 서버 그룹을 만듭니다.

```
# ilbadm create-servergroup -s servers=webserv1,webserv2,webserv3 webgroup
```

▼ ILB 서버 그룹 삭제 방법

1 터미널 창에서 show-servergroup 하위 명령을 입력하여 특정 서버 그룹 또는 모든 서버 그룹에 대한 정보를 얻습니다.

```
# ilbadm show-servergroup -o all
```

다음 샘플 명령은 모든 서버 그룹에 대한 세부 정보를 나열합니다.

sgname	serverID	minport	maxport	IP_address
specgroup	_specgroup.0	7001	7001	199.199.68.18
specgroup	_specgroup.1	7001	7001	199.199.68.19
test123	_test123.0	7002	7002	199.199.67.18
test123	_test123.1	7002	7002	199.199.67.19

위의 표에는 2개의 서버 그룹, specgroup 및 test123이 표시되어 있습니다. specgroup에는 2개의 서버, 199.199.68.18 및 199.199.68.19가 포함되어 있고 서버는 포트 7001을 사용합니다. 마찬가지로, test123에도 2개의 서버, 199.199.67.18 및 199.199.67.19가 포함되어 있습니다. 이 서버는 포트 7002를 사용합니다.

2 삭제하려는 서버 그룹을 선택합니다.

서버 그룹을 활성 규칙에서 사용하고 있지 않아야 합니다. 그렇지 않을 경우 삭제가 실패합니다.

3 터미널 창에서 다음 명령을 사용하여 서버 그룹을 삭제합니다.

```
# ilbadm delete-servergroup servergroup
```

예 12–2 ILB 서버 그룹 삭제

다음 예에서는 webgroup이라는 서버 그룹을 제거합니다.

```
# ilbadm delete-servergroup webgroup
```

ILB의 백엔드 서버 관리

이 절에서는 ilbadm 명령을 사용하여 서버 그룹 내에서 하나 이상의 백엔드 서버를 추가, 제거, 사용으로 설정 그리고 사용 안함으로 설정하는 방법에 대해 설명합니다.

▼ ILB 서버 그룹에 백엔드 서버를 추가하는 방법

- **서버 그룹에 백엔드 서버를 추가합니다.**

  서버 사양은 호스트 이름 또는 IP 주소를 포함해야 하고, 선택적 포트 또는 포트 범위를 포함할 수도 있습니다. 서버 그룹 내에서 IP 주소가 동일한 서버 항목은 허용되지 않습니다.

  ```
  # ilbadm add-server -s server=192.168.89.1,192.168.89.2 ftpgroup
  # ilbadm add-server -s server=[2001:7::feed:6]:8080 sgrp
  ```

  -e 옵션은 서버 그룹에 추가된 서버를 사용으로 설정합니다.

주 – IPv6 주소는 대괄호로 묶어야 합니다.

예 12–3 ILB 서버 그룹에 백엔드 서버 추가

다음 예에서는 서버 그룹 ftpgroup 및 sgrp에 백엔드 서버를 추가하고 서버를 사용으로 설정합니다.

```
# ilbadm add-server -e -s \
server=192.168.89.1,192.168.89.2 ftpgroup
# ilbadm add-server -e -s server=[2001:7::feed:6]:8080 sgrp
```

▼ ILB 서버 그룹에서 백엔드 서버를 제거하는 방법

1. **서버 그룹에서 백엔드 서버를 제거하려면 다음 단계를 수행합니다.**

   a. **서버 그룹에서 제거할 서버의 서버 ID를 식별합니다.**

   서버 ID는 서버가 서버 그룹에 추가될 때 시스템에 지정되는 IP 주소의 고유 이름입니다. show-servergroup -o all 하위 명령의 출력에서 얻을 수 있습니다.

   b. **서버를 제거합니다.**

   ```
   # ilbadm remove-server -s server=serverID servergroup
   ```

2. **모든 서버 그룹에서 백엔드 서버를 제거하려면 다음 단계를 수행합니다.**

   a. **제거할 서버의 IP 주소 및 호스트 이름을 식별합니다.**

   b. **ilbadm show-servergroup -o all 명령의 출력을 사용하여 서버를 포함하는 서버 그룹을 식별합니다.**

c. 각 서버 그룹에 대해 다음 하위 명령을 실행하여 서버 그룹에서 서버를 제거합니다.

예 12–4 ILB 서버 그룹에서 백엔드 서버 제거

다음 예에서는 서버 그룹 `sg1`에서 서버 ID가 `_sg1.2`인 서버를 제거합니다.

```
# ilbadm remove-server -s server=_sg1.2 sg1
```

다음 사항에 유의하십시오.

- NAT 또는 Half-NAT 규칙에서 서버를 사용 중인 경우 제거 전에 `disable-server` 하위 명령을 사용하여 서버를 사용 안함으로 설정합니다. 자세한 내용은 172 페이지 "ILB 서버 그룹에서 백엔드 서버를 다시 사용 또는 사용 안함으로 설정하는 방법"을 참조하십시오. 서버가 사용 안함으로 설정되면 연결 드레이닝 상태로 진입합니다. 모든 연결이 드레인된 후에 `remove-server` 하위 명령을 사용하여 서버를 제거할 수 있습니다. `disable-server` 명령을 실행한 후에 정기적으로 NAT 테이블을 검사하여(`show-nat` 명령 사용) 문제의 서버에 여전히 연결이 있는지 확인합니다. 모든 연결이 드레인된 후에(`show-nat` 명령 출력에 서버가 표시되지 않음) `remove-server` 명령을 사용하여 서버를 제거할 수 있습니다.
- `conn-drain` 시간 초과 값이 설정된 경우 시간 초과 기간 종결 시 연결 드레이닝 상태가 완료됩니다. `conn-drain` 시간 초과의 기본값은 `0`이며, 이는 연결이 적절하게 종료될 때까지 계속 기다립니다.

▼ ILB 서버 그룹에서 백엔드 서버를 다시 사용 또는 사용 안함으로 설정하는 방법

1 다시 사용 또는 사용 안함으로 설정할 백엔드 서버의 IP 주소, 호스트 이름 또는 서버 ID를 식별합니다.

IP 주소나 호스트 이름이 지정된 경우 연관된 모든 규칙에 대해 서버가 다시 사용 또는 사용 안함으로 설정됩니다. 서버 ID가 지정된 경우 서버 ID와 연관된 특정 규칙에 대해 서버가 다시 사용 또는 사용 안함으로 설정됩니다.

주 – 서버가 다중 서버 그룹에 속할 경우 여러 서버 ID를 가질 수 있습니다.

---

2 백엔드 서버를 다시 사용 또는 사용 안함으로 설정합니다.

```
# ilbadm enable-server webservergroup.1
# ilbadm disable-server webservergroup.1
```

예 12–5 ILB 서버 그룹에서 백엔드 서버를 다시 사용 및 사용 안함으로 설정하는 방법

다음 예에서는 서버 ID가 `websg.1`인 서버를 사용으로 설정했다가 사용 안함으로 설정합니다.

```
# ilbadm enable-server websg.1
# ilbadm disable-server websg.1
```

ILB의 상태 검사 관리

ILB는 다음과 같이 사용자가 선택할 수 있는 선택적 유형의 서버 상태 검사를 제공합니다.

- 내장 핑 프로브
- 내장 TCP 프로브
- 내장 UDP 프로브
- 상태 검사로 실행할 수 있는 사용자 제공 사용자 정의 테스트

기본적으로 ILB는 상태 검사를 수행하지 않습니다. 로드 균형 조정 규칙을 만들 때 각 서버 그룹에 대해 상태 검사를 지정할 수 있습니다. 로드 균형 조정 규칙당 하나의 상태 검사만 구성할 수 있습니다. 가상 서비스가 사용으로 설정된 동안, 사용으로 설정된 가상 서비스와 연관된 서버 그룹에서 상태 검사가 자동으로 시작되고 정기적으로 반복됩니다. 가상 서비스를 사용 안함으로 설정하면 즉시 상태 검사가 중지됩니다. 가상 서비스를 다시 사용으로 설정하면 이전 상태 검사 상태는 보존되지 않습니다.

상태 검사를 실행하기 위해 TCP, UDP 또는 사용자 정의 테스트 프로브를 지정할 때 ILB는 기본적으로 핑 프로브를 보내 지정된 TCP, UDP 또는 사용자 정의 테스트 프로브를 서버로 보내기 전에 서버에 연결할 수 있는지 확인합니다. 핑 프로브는 서버 상태를 모니터하는 방법입니다. 핑 프로브가 실패하면 해당 서버가 사용 안함으로 설정되고 상태 검사 상태가 `unreachable`이 됩니다. 핑 프로브는 성공하지만 TCP, UDP 또는 사용자 정의 테스트 프로브가 실패하면 서버가 사용 안함으로 설정되고 상태 검사 상태는 `dead`가 됩니다.

주 –

- 기본 핑 프로브를 사용 안함으로 설정할 수 있습니다.
- UDP 프로브의 경우 기본 핑 프로브를 사용 안함으로 설정할 수 없습니다. 따라서 UDP 상태 검사의 경우 핑 프로브는 항상 기본 프로브입니다.

상태 검사 생성

다음 예에서는 2개의 상태 검사 객체, `hc1` 및 `hc-myscript`가 생성됩니다. 첫번째 상태 검사는 내장 TCP 프로브를 사용합니다. 두번째 상태 검사는 사용자 정의 테스트인 `/var/tmp/my-script`를 사용합니다.

```
# ilbadm create-healthcheck \
-h hc-timeout=3,hc-count=2,hc-interval=8,hc-test=tcp hc1
# ilbadm create-healthcheck -h hc-timeout=3, \
hc-count=2,hc-interval=8,hc-test=/var/tmp/my-script hc-myscript
```

각 인수에 대한 설명은 다음과 같습니다.

`hc-timeout` 상태 검사가 완료되지 않은 경우 실패한 것으로 간주되는 시간 초과를 지정합니다.

`hc-count` `hc-test` 상태 검사 실행 시도 횟수를 지정합니다.

`hc-interval` 연속 상태 검사 사이의 간격을 지정합니다. 동기화를 피하기 위해 실제 간격은 `0.5 * hc-interval` 및 `1.5 * hc-interval` 사이에서 무작위로 설정됩니다.

`hc-test` 상태 검사 유형을 지정합니다.

주 – `hc-test`의 포트 사양은 `create-rule` 하위 명령의 `hc-port` 키워드로 지정됩니다. 자세한 내용은 `ilbadm(1M)` 매뉴얼 페이지를 참조하십시오.

사용자 제공 테스트 세부 정보

사용자 제공 사용자 정의 테스트는 다음 기준을 충족해야 합니다.

- 테스트는 이진일 수도 있고 스크립트일 수도 있습니다.
- 테스트는 시스템의 어디에든 상주할 수 있으며 `create-healthcheck` 하위 명령을 사용하는 경우 절대 경로를 지정해야 합니다.

  `create-rule` 하위 명령에서 상태 검사 사양의 일부로 테스트를 지정하는 경우(예: `/var/tmp/my-script`) 다음과 같이 `ilbd` 데몬이 프로세스를 포크하고 테스트를 실행합니다.

  ```
  /var/tmp/my-script $1 $2 $3 $4 $5
  ```

  각 인수에 대한 설명은 다음과 같습니다.

  `$1` VIP(리터럴 IPv4 또는 IPv6 주소)

  `$2` 서버 IP(리터럴 IPv4 또는 IPv6 주소)

  `$3` 프로토콜(UDP, TCP를 문자열로)

  `$4` 숫자 포트 범위(`hc-port`에 대한 사용자 지정 값)

  `$5` 테스트에서 실패를 반환하기 전에 대기해야 하는 최대 시간(초) 지정된 시간을 넘어 테스트를 실행하면 테스트가 중지될 수 있고 실패로 간주됩니다. 이 값은 사용자 정의 값으로 `hc-timeout`에 지정됩니다.

사용자 제공 테스트 `my-script`는 모든 인수를 사용할 수도 있고 아닐 수도 있지만, **반드시** 다음 중 하나를 반환해야 합니다.

- 마이크로초 단위의 RTT(라운드 트립 시간)
- 테스트에서 RTT를 계산하지 않는 경우 0

- 실패한 경우 -1

기본적으로 상태 검사 테스트는 PRIV_PROC_FORK, RIV_PROC_EXEC 및 RIV_NET_ICMPACCESS 권한으로 실행됩니다.

더 광범위한 권한 세트가 필요한 경우 테스트에 setuid를 구현해야 합니다. 권한에 대한 자세한 내용은 privileges(5) 매뉴얼 페이지를 참조하십시오.

상태 검사 표시

ilbadm list-healthcheck 하위 명령을 사용하여 구성된 상태 검사에 대한 세부 정보를 얻을 수 있습니다.

```
# ilbadm list-healthcheck
```

다음 샘플 출력에는 2가지로 구성된 상태 검사가 나열됩니다.

NAME	TIMEOUT	COUNT	INTERVAL	DEF_PING	TEST
hc1	3	2	8	Y	tcp
hc2	3	2	8	N	/var/usr-script

상태 검사 결과 표시

ilbadm list-hc-result 하위 명령을 사용하여 상태 검사 결과를 얻을 수 있습니다. 규칙이나 상태 검사가 지정되지 않은 경우 모든 상태 검사가 나열됩니다.

다음 예에는 rule1이라는 규칙과 연관된 상태 검사 결과가 표시되어 있습니다.

```
# ilbadm show-hc-result rule1
```

RULENAME	HCNAME	SERVERID	STATUS	FAIL	LAST	NEXT	RTT
rule1	hc1	_sg1:0	dead	10	11:01:19	11:01:27	941
rule1	hc1	_sg1:1	alive	0	11:01:20	11:01:34	1111

표의 LAST 열에는 서버에서 상태 검사가 수행된 시간이 표시됩니다. NEXT 열에는 서버에서 다음 상태 검사가 수행될 시간이 표시됩니다.

상태 검사 삭제

다음 예에서는 *hc1*이라는 상태 검사를 삭제합니다.

```
# ilbadm delete-healthcheck hc1
```

ILB 규칙 관리

ILB에서는 가상 서버가 로드 균형 조정 규칙으로 표시되며 다음 매개변수로 정의됩니다.

- 가상 IP 주소
- 전송 프로토콜: TCP 또는 UDP
- 포트 번호(또는 포트 범위)
- 로드 균형 조정 알고리즘
- 로드 균형 조정 모드의 유형(DSR, Full-NAT 또는 Half-NAT)
- 일련의 백엔드 서버로 구성된 서버 그룹
- 서버 그룹의 각 서버에 대해 실행할 수 있는 선택적 서버 상태 검사
- 상태 검사에 사용할 선택적 포트

주 – 특정 포트 또는 `ilbd` 데몬이 서버 포트 범위에서 임의적으로 선택하는 포트에 대해 건전성 검사를 지정할 수 있습니다.

- 가상 서비스를 표시할 규칙 이름

이 절에서는 `ilbadm` 명령을 사용하여 로드 균형 조정 규칙을 생성, 삭제 및 나열하는 방법에 대해 설명합니다.

ILB 규칙 나열

규칙의 구성 세부 정보를 나열하려면 `ilbadm show-rule` 하위 명령을 사용합니다. 규칙 이름이 지정되지 않으면 모든 규칙에 대한 정보가 제공됩니다.

```
# ilbadm show-rule
```

다음은 샘플 명령 출력입니다.

RULENAME	STATUS	LBALG	TYPE	PROTOCOL	VIP	PORT
rule-http	E	hash-ip-port	HALF-NAT	TCP	10.0.0.1	80

RULENAME	STATUS	LBALG	TYPE	PROTOCOL	VIP	PORT
rule-dns	D	hash-ip	DSR	UDP	10.0.0.1	53
rule-abc	D	roundrobin	NAT	TCP	2003::1	1024
rule-xyz	E	hash-ip-vip	NAT	TCP	2003::1	2048-2050

▼ ILB 규칙 생성 방법

1 적절한 백엔드 서버를 포함하는 서버 그룹을 만듭니다.

```
# ilbadm create-servergroup -s server=server1:port-range1,server2:port-range2  sg1
```

2 서버 상태 검사를 규칙과 연관시키려면 상태 검사를 만듭니다.

```
# ilbadm create-healthcheck -h hc-test=protocol, \
hc-timeout=value1,hc-count=value2 \
,hc-interval=value3 hc1
```

3 규칙과 연관될 VIP, 포트 및 선택적 프로토콜을 식별합니다.

`-i` 옵션을 사용하여 지정됩니다.

4 DSR, Half-NAT, Full-NAT 중에서 사용할 작업을 선택합니다.

NAT가 선택된 경우 `proxy-src` 주소로 사용될 IP 주소 범위를 지정해야 합니다. Full-NAT 토폴로지의 경우 IP 주소 10개로 범위가 제한됩니다.

5 사용될 로드 균형 조정 알고리즘을 선택합니다.

4단계 및 5단계의 매개변수는 `-m` 옵션에서 지정될 수 있습니다. 자세한 내용은 155 페이지 “ILB 알고리즘”을 참조하십시오.

6 다른 선택적 기능을 선택합니다.

자세한 내용은 `ilbadm(1M)` 매뉴얼 페이지에서 세부 정보를 참조하십시오.

7 규칙 이름을 선택합니다.

8 규칙을 만들고 사용으로 설정합니다.

각 옵션에 대한 자세한 내용은 `ilbadm(1M)` 매뉴얼 페이지를 참조하십시오.

```
# ilbadm create-rule -e -i vip=ipaddr,port=port,protocol=protocol \
-m lbalg=lb-algorithm,type=topology-type,proxy-src=ipaddr1-ipaddr2, \
pmask=value4 -h hc-name=hc1 \
-o servergroup=sg1  rule1
```

다음 예는 상태 검사가 사용된 Full-NAT 규칙 생성 단계를 보여줍니다.

예 12–6 상태 검사 세션 지속성이 사용된 Full-NAT 규칙 만들기

이 예에서는 hc1이라는 상태 검사와 sg1이라는 서버 그룹을 만듭니다. 서버 그룹은 두 개의 서버로 구성되고 각각 포트 범위를 가지고 있습니다. 마지막 명령은 rule1이라는 규칙을 만들어 사용으로 설정하고 서버 그룹 및 상태 검사에 규칙을 연관시킵니다. 이 규칙이 Full-NAT 작동 모드를 구현합니다. 서버 그룹 및 상태 검사 생성은 규칙 생성보다 선행되어야 합니다.

```
# ilbadm create-healthcheck -h hc-test=tcp,hc-timeout=2, \
hc-count=3,hc-interval=10 hc1
# ilbadm create-servergroup -s server=60.0.0.10:6000-6009,60.0.0.11:7000-7009 sg1
# ilbadm create-rule -e -i vip=81.0.0.10,port=5000-5009, \
protocol=tcp -m lbalg=rr,type=NAT, \
proxy-src=60.0.0.101-60.0.0.104,persist=24 \
-h hc-name=hc1 -o servergroup=sg1 rule1
```

Half-NAT 또는 Full-NAT 규칙 생성 시 connection-drain 시간 초과 값을 지정하십시오. conn-drain 시간 초과의 기본값은 0이며, 이는 연결이 적절하게 종료될 때까지 계속 기다립니다.

ILB 규칙 삭제

규칙을 삭제하려면 ilbadm delete-rule 하위 명령을 사용합니다. 모든 규칙을 삭제하려면 -a 옵션을 사용합니다. 다음 예에서는 rule1 규칙을 삭제합니다.

```
# ilbadm delete-rule rule1
```

ILB 통계 표시

이 절에서는 ilbadm 명령을 사용하여 서버 통계 또는 규칙 통계 인쇄와 같은 정보를 얻을 수 있는 방법에 대해 설명합니다. NAT 테이블 정보 및 세션 지속성 매핑 테이블을 표시할 수도 있습니다.

통계 정보 얻기

ilbadm show-statistics 하위 명령을 사용하여 로드 분배 세부 정보를 봅니다. 다음 예에서는 show-statistics 하위 명령 사용법을 보여줍니다.

```
# ilbadm show-statistics
 PKT_P   BYTES_P   PKT_U   BYTES_U   PKT_D   BYTES_D
   9       636       0         0       0         0
```

PKT_P 처리된 패킷

BYTES_P 처리된 바이트

PKT_U 처리되지 않은 패킷

BYTES_U 처리되지 않은 바이트

PKT_D 삭제된 패킷

BYTES_D 삭제된 바이트

NAT 연결 테이블 표시

ilbadm show-nat 하위 명령을 사용하여 NAT 연결 테이블을 표시합니다. 이 명령의 연속적 실행에서 요소의 상대적 위치에 대해 어떤 가정도 없어야 합니다. 예를 들어, 특히 혼잡한 시스템에서 {{ ilbadm show-nat 10}}을 두 번 실행한다고 해서 똑같은 10개 항목이 두 번 표시된다고 보장할 수 없습니다. 개수 값이 지정되지 않은 경우 전체 NAT 연결 테이블이 표시됩니다.

예 12–7 NAT 연결 테이블 항목

다음 예에서는 NAT 연결 테이블에서 5개 항목이 표시됩니다.

```
# ilbadm show-nat 5
UDP: 124.106.235.150.53688 > 85.0.0.1.1024 >>> 82.0.0.39.4127 > 82.0.0.56.1024
UDP: 71.159.95.31.61528 > 85.0.0.1.1024 >>> 82.0.0.39.4146 > 82.0.0.55.1024
UDP: 9.213.106.54.19787 > 85.0.0.1.1024 >>> 82.0.0.40.4114 > 82.0.0.55.1024
UDP: 118.148.25.17.26676 > 85.0.0.1.1024 >>> 82.0.0.40.4112 > 82.0.0.56.1024
UDP: 69.219.132.153.56132 > 85.0.0.1.1024 >>> 82.0.0.39.4134 > 82.0.0.55.1024
```

항목의 형식은 다음과 같습니다.

```
T: IP1 > IP2 >>> IP3 > IP4
```

T 해당 항목에서 사용된 전송 프로토콜

IP1 클라이언트 IP 주소 및 포트

IP2 VIP 및 포트

IP3 Half-NAT 모드인 경우 클라이언트 IP 주소 및 포트

Full-NAT 모드인 경우 클라이언트 IP 주소 및 포트

IP4 백엔드 서버 IP 주소 및 포트

세션 지속성 매핑 테이블 표시

ilbadm show-persist 하위 명령을 사용하여 세션 지속성 매핑 테이블을 표시합니다.

예 12–8 세션 지속성 매핑 테이블 항목

다음 예에서는 세션 지속성 매핑 테이블에서 5개 항목이 표시됩니다.

```
# ilbadm show-persist 5
rule2: 124.106.235.150 --> 82.0.0.56
rule3: 71.159.95.31 --> 82.0.0.55
rule3: 9.213.106.54 --> 82.0.0.55
rule1: 118.148.25.17 --> 82.0.0.56
rule2: 69.219.132.153 --> 82.0.0.55
```

항목의 형식은 다음과 같습니다.

```
R: IP1 --> IP2
```

R 이 지속성 항목이 연결된 규칙

IP1 클라이언트 IP 주소

IP2 백엔드 서버 IP 주소

◆ ◆ ◆ 13장

13 가상 라우터 중복 프로토콜(개요)

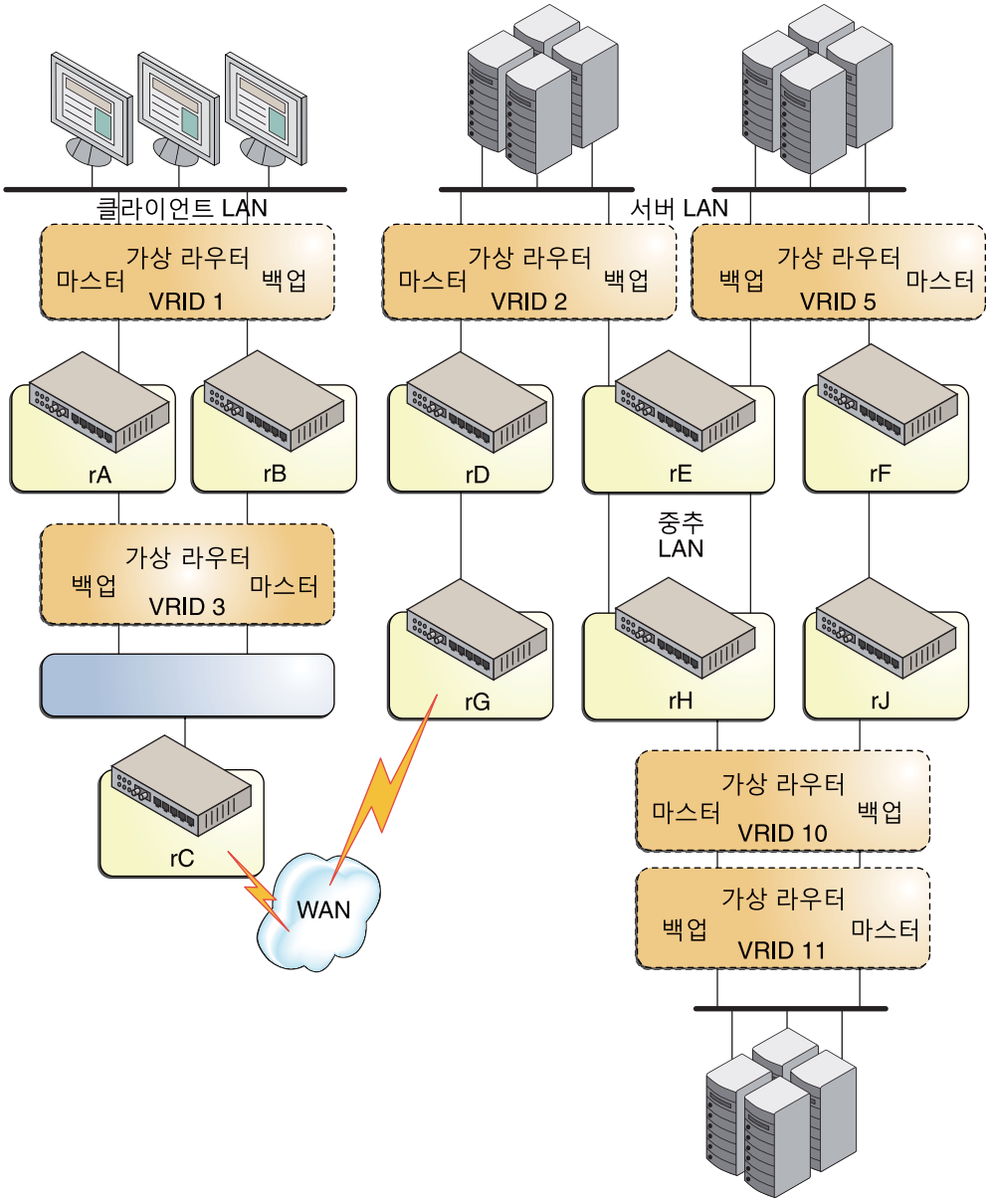
VRRP(Virtual Router Redundancy Protocol)는 IPv4 및 IPv6용 VRRP(Virtual Router Redundancy Protocol) 버전 3에 지정된 인터넷 표준 프로토콜입니다. Oracle Solaris에서는 VRRP가 지원되어 고가용성을 제공합니다. Oracle Solaris는 VRRP 서비스를 구성 및 관리하는 관리 도구를 제공합니다.

다음 내용으로 구성되어 있습니다.

VRRP 작동 방식

다음 그림은 VRRP 작동 방식을 보여줍니다. 그림 뒤에 오는 텍스트는 그림에 사용된 VRRP 구성 요소에 대한 설명입니다.

그림 13–1 VRRP 작업



이전 그림에서는 VRRP가 다음 구성 요소를 기다렸다가 이 지역에 있습니다.

- 라우터 rA는 가상 라우터 VRID 1의 마스터 라우터이며 VRID 3의 백업 라우터입니다. 라우터 rA는 VRID 1의 VIP로 주소가 지정된 패킷의 경로 지정을 처리하고 VRID 3의 경로 지정 역할을 맡을 준비가 되어 있습니다.
- 라우터 rB는 가상 라우터 VRID 3의 마스터 라우터이며 VRID 1의 백업 라우터입니다. 라우터 rB는 VRID 3의 VIP로 주소가 지정된 패킷의 경로 지정을 처리하고 VRID 1의 경로 지정 역할을 맡을 준비가 되어 있습니다.
- 라우터 rC에는 VRRP 기능이 없으나 VRID 3의 VIP를 사용하여 클라이언트 LAN 서브넷에 접속합니다.
- 라우터 rD는 VRID 2의 마스터 라우터입니다. 라우터 rF는 VRID 5의 마스터 라우터입니다. 라우터 rE는 이러한 두 VRID의 백업 라우터입니다. rD 또는 rF가 실패하면 rE가 해당 VRID의 마스터 라우터가 됩니다. rD 및 rF 둘 다 동시에 실패할 수 있습니다. VRRP 라우터가 VRID의 마스터 라우터라고 해서 다른 VRID의 마스터 라우터가 되는 데 배제되는 것은 아닙니다.
- 라우터 rG는 중추 LAN의 WAN(원거리 통신망) 게이트웨이입니다. 중추에 연결된 라우터는 모두 OSPF(Open Shortest Path First) 같은 동적 경로 지정 프로토콜을 사용하여 WAN의 라우터와 경로 지정 정보를 공유합니다. 라우터 rC에서 클라이언트 LAN 서브넷에 대한 경로가 VRID 3의 VIP를 경유한다고 공개하는 경우에도 VRRP는 이 작업에 관련되지 않습니다.
- 라우터 rH는 VRID 10의 마스터 라우터이며 VRID 11의 백업 라우터입니다. 마찬가지로 라우터 rJ는 VRID 11의 마스터 라우터이며 VRID 10의 백업 라우터입니다. 이러한 VRRP 로드 공유 구성은 여러 VRID가 하나의 라우터 인터페이스에 있을 수 있음을 나타냅니다.

VRRP는 네트워크의 모든 시스템에 대해 거의 전체 경로 지정 중복성을 제공하는 네트워크 디자인의 일부로 사용될 수 있습니다.

근거리 통신망에서 VRRP 사용

LAN(근거리 통신망) 같은 네트워크를 설정하는 경우 고가용성 서비스를 제공하는 것이 매우 중요합니다. 네트워크 안정성을 높이는 한 가지 방법은 네트워크의 중요한 구성 요소에 대해 백업을 제공하는 것입니다. 라우터, 스위치, 링크 같은 구성 요소를 네트워크에 추가하면 장애 시에도 서비스 연속성을 보장합니다. 네트워크 끝점에서 중복성을 제공하는 것은 VRRP로 쉽게 수행할 수 있는 중요한 작업입니다. VRRP를 사용하여 라우터에 대해 실패 복구를 제공하면 LAN에 가상 라우터가 도입될 수 있습니다.

VRRP는 가상 라우터의 책임을 LAN 내에 있는 VRRP 라우터 중 하나에 동적으로 지정하는 선택 프로토콜입니다. VRRP는 LAN에 정적으로 구성된 라우터에 하나 이상의 백업 라우터를 제공합니다.

마스터 라우터라고 불리는 VRRP 라우터는 IPv4 또는 IPv6 주소나 가상 라우터와 연결된 주소를 제어합니다. 가상 라우터는 마스터 라우터의 IP 주소로 전송된 패킷을 전달합니다.

선택 프로세스는 이러한 IP 주소로 전송된 패킷을 전달하는 동안 동적 페일오버를 제공합니다. VRRP는 정적 기본 경로 지정 환경에 내재되어 있는 단일 오류 지점을 제거합니다.

Oracle Solaris에서 VRRP 기능을 사용하면 모든 단말 호스트에서 동적 경로 지정 또는 라우터 검색 프로토콜을 구성할 필요 없이 경로 지정 프로세스에 대해 보다 가용성이 높은 기본 경로를 제공할 수 있습니다.

VRRP 라우터

VRRP는 각 VRRP 라우터에서 실행되며 라우터의 상태를 관리합니다. 한 호스트에 여러 개의 VRRP 라우터가 구성될 수 있으며, 각 VRRP 라우터는 서로 다른 가상 라우터에 속합니다.

VRRP 라우터의 속성은 다음과 같습니다.

- **라우터 이름** – 시스템 전체 고유 식별자
- **VRID(가상 라우터 ID)** – 지정된 네트워크 세그먼트에서 가상 라우터를 식별하는 데 사용되는 고유한 숫자입니다. VRID는 LAN 내에서 가상 라우터를 식별합니다.
- **기본 IP 주소** – VRRP 공개의 소스 IP 주소입니다.
- **VRIP(가상 IP 주소)** – 다른 호스트가 네트워크 서비스를 가져오는 VRID와 연결된 IP 주소입니다. VRIP는 VRID에 속한 VRRP 인스턴스에 의해 관리됩니다.
- **VRRP 매개변수** – 우선 순위, 공개 간격, 선취 모드 및 승인 모드를 포함합니다.
- **VRRP 상태 정보 및 통계**

VRRP 하위 명령 관리

다음 절에는 `vrrpadm` 하위 명령이 요약되어 있습니다. 자세한 내용은 `vrrpadm(1M)` 매뉴얼 페이지를 참조하십시오. `vrrpadm show-router` 하위 명령을 제외하고는 모든 하위 명령의 결과가 보존됩니다. 예를 들어 `vrrpadm create-router`로 생성된 VRRP 라우터는 재부트 이후에도 보존됩니다.

VRRP VNIC 생성

시스템의 물리적 네트워크 어댑터를 기반으로 구성되는 의사 네트워크 인터페이스로, NIC(네트워크 인터페이스 카드)라고도 합니다. 물리적 인터페이스에는 VNIC가 여러 개 있을 수 있습니다. VNIC는 네트워크 가상화의 필수 구성 요소입니다. 자세한 내용은 **Oracle Solaris 11.1에서 가상 네트워크 사용**을 참조하십시오.

기존 dladm create-vnic 하위 명령이 확장되어 VRRP VNIC를 만들 수 있습니다. 구문은 다음과 같습니다.

```
# dladm create-vnic [-t] [-R root-dir] [-l link] [-m vrrp -V VRID -A \
{inet | inet6}] [-v vlan-id] [-p prop=value[,...]] vnic-link
```

새 VNIC 주소 유형인 vrrp가 도입되었습니다. 이러한 새 VNIC 주소 유형으로 VRID 및 주소 그룹을 지정해야 합니다.

그 결과 잘 알려진 가상 라우터 MAC 주소를 사용하는 VNIC가 생성됩니다.

라우터 만들기

vrrpadm create-router 하위 명령은 지정된 VRID 및 주소 그룹을 다른 지정된 매개변수와 함께 사용하여 VRRP 라우터를 만듭니다. 각 VRRP 라우터의 경우 특수 VRRP VNIC를 만들어야 하며 VNIC는 dladm create-vnic 명령을 사용하여 만들 수 있습니다. 자세한 내용은 vrrpadm(1M) 매뉴얼 페이지를 참조하십시오. 구문은 다음과 같습니다.

```
# vrrpadm create-router -V vrid -l link -A {inet | inet6} \
[-p priority] [-i adv-interval] [-o flags] router-name
```

-o 옵션은 VRRP 라우터의 선취 모드 및 승인 모드를 구성하는 데 사용됩니다. 값은 preempt, un_preempt, accept, no_accept일 수 있습니다. 기본적으로 두 모드가 모두 true로 설정되어 있습니다.

*router-name*은 이 VRRP 라우터의 고유 식별자로 사용되며, 다른 vrrpadm 하위 명령에 사용됩니다. 라우터 이름에 허용되는 문자는 영숫자(a-z, A-Z, 0-9) 및 밑줄('_')입니다. 라우터 이름의 최대 길이는 31자입니다.

라우터 사용으로 설정

사용 안함으로 설정된 VRRP 라우터는 enable-router 하위 명령을 사용하여 다시 사용으로 설정할 수 있습니다. 라우터를 사용으로 설정하는 경우 VRRP 라우터를 만들 때 경유하는 기본 데이터 링크(vrrpadm create-router를 사용하여 라우터를 만들 때 -l 옵션으로 지정) 및 라우터의 VRRP VNIC가 존재해야 합니다. 그렇지 않으면 사용으로 설정 작업이 실패합니다. 구문은 다음과 같습니다.

```
# vrrpadm enable-router router-name
```

라우터 수정

vrrpadm modify-router 하위 명령은 지정된 VRRP 라우터의 구성을 변경합니다. 구문은 다음과 같습니다.

```
# vrrpadm modify-router [-p priority] [-i adv-interval] \
[-o flags] router-name
```

라우터 구성 표시

vrrpadm show-router 하위 명령은 지정된 VRRP 라우터의 구성 및 상태를 표시합니다. 자세한 정보는 vrrpadm(1M) 매뉴얼 페이지를 참조하십시오. 구문은 다음과 같습니다.

```
# vrrpadm show-router [-P | -x] [-p] [-o field[,...]] [router-name]
```

다음은 vrrpadm show-router 출력 예입니다.

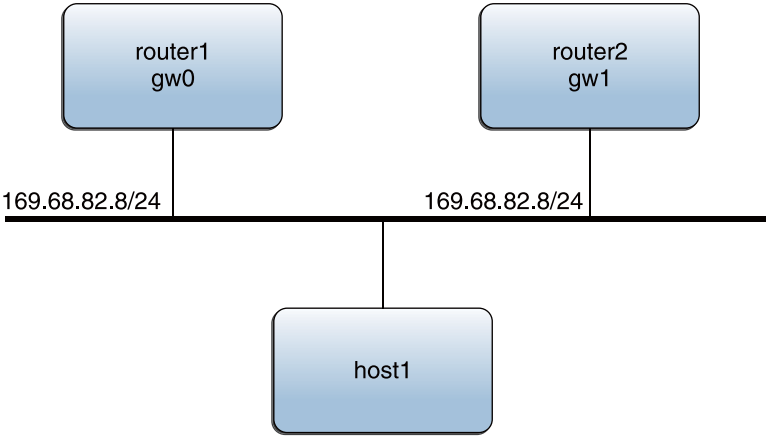
```
# vrrpadm show-router vrrp1
NAME VRID LINK AF PRIO ADV_INTV MODE STATE VNIC
vrrp1 1 bge1 IPv4 100 1000 e-pa- BACK vnic1

# vrrpadm show-router -x vrrp1
NAME STATE PRV_STAT STAT_LAST VNIC PRIMARY_IP VIRTUAL_IPS
vrrp1 BACK MAST 1m17s vnic1 10.0.0.100 10.0.0.1

# vrrpadm show-router -P vrrp1
NAME PEER P_PRIO P_INTV P_ADV_LAST M_DOWN_INTV
vrrp1 10.0.0.123 120 1000 0.313s 3609
```

예 13–1 VRRP 구성 예

다음 그림은 일반 VRRP 구성을 보여줍니다.



이 예에서 IP 주소 169.68.82.8은 host1의 기본 게이트웨이로 구성됩니다. 이 IP 주소는 두 개의 VRRP 라우터(router1 및 router2)로 구성된 가상 라우터가 보호하는 가상 IP 주소입니다. 한 번에 두 라우터 중 하나만 마스터 라우터로 사용되어 가상 라우터의 책임을 맡으며 host1에서 보내는 패킷을 전달합니다.

예 13–1 VRRP 구성 예 *(계속)*

가상 라우터의 VRID는 12라고 가정합니다. 다음 예는 router1 및 router2에서 위의 VRRP 구성을 구성하는 데 사용되는 명령을 보여줍니다. router1은 가상 IP 주소 169.68.82.8의 소유자이며, 우선 순위는 기본값(255)입니다. router2는 우선 순위가 100인 대기 라우터입니다.

```
router1:
# dladm create-vnic -m vrrp -V 12 -A inet -l gw0 vnic1
# vrrpadm create-router -V 12 -A inet -l gw0 vrrp1
# ipadm create-addr -d -a 169.68.82.8/24 vnic1/router1
# ipadm create-addr -d -a 169.68.82.100/24 gw0/router1
# vrrpadm show-router -x vrrp1
NAME STATE PRV_STAT STAT_LAST VNIC PRIMARY_IP VIRTUAL_IPS
vrrp1 MAST BACK 1m17s vnic1 169.68.82.100 169.68.82.8
router2:
# dladm create-vnic -m vrrp -V 12 -A inet -l gw1 vnic1
# vrrpadm create-router -V 12 -A inet -l gw1 -p 100 vrrp1
# ipadm create-addr -d -a 169.68.82.8/24 vnic1/router2
# ipadm create-addr -d -a 169.68.82.101/24 gw1/router2
# vrrpadm show-router -x vrrp1
NAME STATE PRV_STAT STAT_LAST VNIC PRIMARY_IP VIRTUAL_IPS
vrrp1 BACK INIT 2m32s vnic1 169.68.82.101 169.68.82.8
```

router1의 구성을 예로 사용할 경우 gw0을 통해 하나 이상의 IP 주소를 구성해야 합니다. 다음 예에서 router1의 이 IP 주소는 기본 IP 주소로, VRRP 공개 패킷을 전송하는 데 사용됩니다.

```
# vrrpadm show-router -x vrrp1
NAME STATE PRV_STAT STAT_LAST VNIC PRIMARY_IP VIRTUAL_IPS
vrrp1 MAST BACK 1m17s vnic1 169.68.82.100 169.68.82.8
```

라우터 사용 안함으로 설정

VRRP 라우터가 사용으로 설정할 때까지 작동하지 않습니다. 기본적으로 VRRP 라우터는 처음 생성될 때 사용으로 설정됩니다. 그러나 일시적으로 VRRP 라우터를 사용 안함으로 설정하여 구성을 변경한 후 라우터를 다시 사용으로 설정하는 것이 유용할 때도 있습니다. 구문은 다음과 같습니다.

```
# vrrpadm disable-router router-name
```

라우터 삭제

vrrpadm delete-router 하위 명령은 지정된 VRRP 라우터를 삭제합니다. 구문은 다음과 같습니다.

```
# vrrpadm delete-router router-name
```

VRRP 보안 고려 사항

VRRP 서비스를 구성하려면 `solaris.network.vrrp` 권한 부여가 필요합니다. 읽기 전용 작업인 `vrrpadm showrouter`의 경우에는 이 권한 부여가 필요하지 않습니다.

`solaris.network.vrrp` 권한 부여는 네트워크 관리 프로파일의 일부입니다.

VRRP 제한 사항

이 절에서는 VRRP 제한 사항에 대해 설명합니다.

배타적 IP 영역 지원

각 배타적 IP 영역에서는 VRRP 라우터가 해당 영역에서 생성되면 VRRP 서비스 `svc:/network/vrrp/default`가 자동으로 사용으로 설정됩니다. VRRP 서비스는 이러한 특정 영역에 대한 VRRP 라우터를 관리합니다.

그러나 다음과 같은 이유로 배타적 IP 영역에 대한 지원은 제한적입니다.

- 비전역 영역 내에서는 VNIC를 만들 수 없습니다. 그러므로 먼저 전역 영역에서 VRRP VNIC를 만들어야 합니다. 그런 다음 VRRP 라우터가 상주하는 비전역 영역에 VNIC를 지정해야 합니다. 그러면 VRRP 라우터를 `vrrpadm` 명령을 사용하여 비전역 영역에서 만들고 시작할 수 있습니다.
- 단일 Oracle Solaris 시스템에서 동일한 가상 라우터에 참가할 두 개의 VRRP 라우터를 서로 다른 영역에 만들 수 없습니다. Oracle Solaris에서 동일한 MAC 주소를 사용하는 두 개의 VNIC를 만들 수 없기 때문입니다.

다른 네트워크 기능과의 상호 작업

VRRP 서비스는 IPMP(IP 네트워크 다중 경로) 인터페이스에서 작동할 수 없습니다. VRRP에는 특정 VRRP MAC 주소가 필요하지만 IPMP는 IP 층에서 완전하게 작동합니다.

또한 VRRP 가상 IP 주소는 정적으로만 구성될 수 있으며, IP 주소에 대한 두 개의 기존 자동 구성 도구인 `in.ndpd`(IPv6 자동 구성의 경우) 및 `dhcpagent`(DHCP 구성의 경우)로 자동 구성할 수 없습니다. 마스터 및 백업 VRRP 라우터(VNIC)가 동일한 MAC 주소를 공유하기 때문에 `in.ndpd`와 `dhcpagent`가 혼동될 수 있습니다. 결과적으로 예기치 못한 결과가 발생할 수 있습니다. 따라서 IPv6 자동 구성 및 DHCP 구성은 VRRP VNIC를 통해 지원되지 않습니다. VRRP VNIC를 통해 IPv6 자동 구성 또는 DHCP를 구성하면 자동으로 구성된 IP 주소를 가져오려는 시도가 실패하고 자동 구성 작업도 실패합니다.

◆ ◆ ◆ 부 록 A

링크 통합 유형: 기능 비교

트렁크 통합 및 데이터 링크 다중 경로(DLMP) 통합 모두 네트워크 성능 향상을 위해 거의 동일한 기능을 지원합니다. 그러나 차이도 존재합니다. 다음 표는 Oracle Solaris에서 제공하는 2가지 유형의 링크 통합에 대한 일반적인 비교 내용을 보여줍니다.

기능	트렁크 통합	DLMP 통합
링크 기반 실패 감지	지원됨	지원됨
LACP(링크 통합 제어 프로토콜)	지원됨	지원되지 않음
대기 인터페이스 사용	지원되지 않음	지원됨
여러 스위치 확장	공급업체 독점 솔루션을 사용하는 경우에만 지원됨	지원됨
스위치 구성	필수	필요하지 않음
로드 균형 조정 정책	지원됨	적용할 수 없음
모든 통합 포트에서 부하 분산	지원됨	제한됨[1]
사용자 정의 리소스 관리 플로우	지원됨	지원됨
링크 보호	지원됨	지원됨
인접(Back-to-Back) 병렬 구성	지원됨	지원되지 않음[2]

[1]통합을 사용하는 경우 해당 VNIC가 모든 포트에 분산됩니다. 그러나 개별 VNIC는 여러 포트에 부하를 분산할 수 없습니다.

[2]DLMP 통합에서는 항상 중간 스위치를 사용하여 패킷을 다른 대상 시스템으로 보내야 합니다. 그러나 DLMP 통합을 사용하는 경우 링크 통합에 사용할 스위치는 구성하지 마십시오.

◆ ◆ ◆ 부 록 B

링크 통합 및 IPMP: 기능 비교

IPMP 및 링크 통합은 네트워크 성능을 향상시키고 네트워크 가용성을 유지 관리하는 서로 다른 기술입니다.

다음 표에서는 링크 통합과 IPMP의 일반적인 비교 내용을 보여줍니다.

기능	IPMP	링크 통합
네트워크 기술 유형	계층 3(IP 계층)	계층 2(링크 계층)
구성 도구	`ipadm`	`dladm`
링크 기반 실패 감지	지원됨	지원됨
프로브 기반 실패 감지	ICMP 기반으로, 중간에 있는 계층 2 스위치의 여러 레벨에서 테스트 주소와 동일한 IP 서브넷에 정의된 모든 시스템을 대상으로 함	LACP 기반으로, 직접 피어 호스트 또는 스위치를 대상으로 함 DLMP 통합에서는 지원되지 않음
대기 인터페이스 사용	지원됨	트렁크 통합에서는 지원되지 않음 DLMP 통합에서 지원됨
여러 스위치 확장	지원됨	트렁크 통합에서는 지원되지 않음, 일부 공급업체는 여러 스위치로 확장하기 위해 상호 운용 불가능한 독점 솔루션 제공. DLMP 통합에서 지원됨
하드웨어 지원	필요하지 않음	트렁크 통합에 필요. 예를 들어, Oracle Solaris 시스템에서 트렁크 통합을 사용하려면 스위치의 해당 포트도 통합해야 함. DLMP 통합에는 필요하지 않음

기능	IPMP	링크 통합
링크 계층 요구 사항	브로드캐스트 가능	이더넷 관련
드라이버 프레임워크 요구 사항	없음	GLDv3 프레임워크를 사용해야 함
부하 분산 지원	지원됨, 커널에 의해 제어됩니다. 인바운드 부하 분산은 소스 주소 선택의 간접적인 영향을 받음.	트렁크 통합에서만 지원되며 관리자가 `dladm` 명령으로 제어함, 인바운드 부하 분산 지원됨. DLMP 통합에서는 통합을 통한 인터페이스별 부하 분산이 지원되지 않음.
VNIC 통합 시 지원 레벨	빈약한 지원	최고 지원
사용자 정의 리소스 관리 플로우	지원되지 않음	지원됨
링크 보호	지원되지 않음	지원됨
프로토콜 요구 사항	없음	없음

링크 통합에서는 표준 모드로 통합을 구성하는 여러 링크에 수신 트래픽이 분산됩니다. 따라서 통합에 링크를 추가하기 위해 더 많은 NIC를 설치하면 네트워킹 성능이 향상됩니다.

DLMP 통합은 여러 스위치에 확장되어 있습니다. 통합은 계층 2 기술로 다른 Oracle Solaris 가상화 기술과 잘 통합됩니다.

IPMP 트래픽은 사용 가능한 활성 인터페이스에 바인딩된 경우 IPMP 인터페이스의 데이터 주소를 사용합니다. 예를 들어, 모든 데이터 트래픽이 두 개의 IP 주소 간에만 전달되지만 반드시 동일한 연결을 사용하지는 않는 경우 IP 주소 두 개만 사용할 수 있기 때문에 NIC를 더 추가해도 IPMP에서 성능이 향상되지 않습니다.

트렁크 통합 및 IPMP는 서로를 보완하는 기술이므로 함께 배포하면 네트워크 성능과 가용성을 더욱 향상시킬 수 있습니다.

색인

L

M