Oracle® Solaris 11.2의 TCP/IP 네트워크, IPMP 및 IP 터널 관리

부품 번호: E53802
2014년 7월

ORACLE

목차

이 설명서 사용

- **개요** - Oracle Solaris OS(운영 체제)에서 TCP/IP 네트워크, IPMP 및 IP 터널 관리 작업에 대해 설명합니다.
- **대상** - 시스템 관리자
- **필요한 지식** - TCP/IP 네트워크 관리 및 고급 네트워킹 기능(예: IPMP 및 IP 터널)을 비롯한 네트워크 관리와 관련된 고급 경험

제품 설명서 라이브러리

이 제품에 대한 최신 정보 및 알려진 문제는 설명서 라이브러리(http://www.oracle.com/pls/topic/lookup?ctx=E56343)에서 확인할 수 있습니다.

Oracle 지원 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.

피드백

http://www.oracle.com/goto/docfeedback에서 이 설명서에 대한 피드백을 보낼 수 있습니다.

♦ ♦ ♦ 1 장

1 TCP/IP 네트워크 관리

이 장에서는 Oracle Solaris가 설치된 시스템에서 TCP/IP 프로토콜을 관리하는 방법에 대해 설명합니다. 이 장에서 설명하는 작업은 사용자의 사이트에서 TCP/IP 네트워크 즉, IPv4 전용 또는 듀얼 스택 IPv4/IPv6 네트워크가 작동 가능하다고 가정합니다.

네트워크 배치 계획에 대한 자세한 내용은 "Oracle Solaris 11.2의 네트워크 배치 계획 "을 참조하십시오.

네트워크 구성 작업은 "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "를 참조하십시오.

TCP/IP 네트워크 문제 해결에 대한 일반 정보는 "Oracle Solaris 11.2의 네트워크 관리 문제 해결 "을 참조하십시오.

이 장의 내용:

TCP/IP 등록 정보 사용자 정의

`ipadm` 명령을 사용하면 조정 가능 항목이라고도 하는 대부분의 TCP/IP 등록 정보를 구성할 수 있습니다. `ipadm` 명령은 `ndd` 명령을 조정 가능 항목을 설정하는 기본 도구로 대체합니다.

이러한 변경 사항에 대한 자세한 내용은 "Oracle Solaris 10에서 Oracle Solaris 11.2로 전환 "의 "ndd 명령과 ipadm 명령 비교"를 참조하십시오.

TCP/IP 등록 정보는 인터페이스 기반 또는 전역 등록 정보일 수 있으므로, 특정 인터페이스나 영역의 모든 인터페이스에 전역적으로 등록 정보를 적용할 수 있습니다. 전역 등록 정보는 각 비전역 영역에서 다른 값을 사용할 수도 있습니다. 지원되는 프로토콜 등록 정보의 전체 목록은 `ipadm`(1M) 매뉴얼 페이지를 참조하십시오.

일반적으로 TCP/IP 프로토콜의 기본값으로도 네트워크가 작동하는 데 충분합니다. 그러나 이러한 값이 특정 네트워크 토폴로지에 사용하기에 충분하지 않을 경우 다음 세 개의 `ipadm` 하위 명령을 사용하여 등록 정보를 사용자 정의할 수 있습니다.

- `ipadm show-prop -p` *property protocol* - 프로토콜의 등록 정보와 현재 값을 표시합니다. `-p` *property* 옵션을 사용하지 않으면 프로토콜의 모든 등록 정보가 표시됩니다. 프로토콜을 지정하지 않으면 모든 프로토콜의 모든 등록 정보가 표시됩니다.
- `ipadm set-prop -p` *property=value protocol* - 프로토콜 등록 정보에 값을 지정합니다.
  - 프로토콜 등록 정보에 값을 여러 개 지정하려면 다음 구문을 사용합니다.

    `ipadm set-prop [-t] -p` *property=value*`[,...] protocol`
  - 제공된 등록 정보의 값 세트에서 값을 하나 제거하려면 -= 수식자를 사용합니다.

    `ipadm set-prop -p` *property-=value2*
- 다음과 같이 특정 프로토콜 등록 정보를 기본값으로 재설정합니다.

  `ipadm reset-prop -p` *property protocol*

IP 인터페이스 등록 정보 사용자 정의에 대한 자세한 내용은 "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "의 "IP 인터페이스 등록 정보 및 주소 사용자 정의"를 참조하십시오.

IP 주소 등록 정보 사용자 정의에 대한 자세한 내용은 "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "의 "IP 주소 등록 정보 사용자 정의"를 참조하십시오.

전역적으로 패킷 전달 사용

`ipadm set-ifprop` 명령을 사용하여 IP 인터페이스에 대한 전달을 사용으로 설정할 경우 해당 인터페이스에 대해서만 전달이 사용으로 설정되고 다른 모든 인터페이스에 대한 전달은 변경되지 않은 상태로 유지됩니다. 개별 IP 인터페이스 등록 정보에 대한 패킷 전달을 설정하면 시스템의 특정 인터페이스에서 이 기능을 선택적으로 구현할 수 있습니다. 자세한 내용은 "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "의 "패킷 전달 사용"을 참조하십시오.

전체 시스템에 대한 패킷 전달을 사용으로 설정하려면 IP 인터페이스 수에 관계없이 `protocol` 등록 정보를 사용하십시오. `forwarding` 등록 정보는 전달을 관리하는 데 사용되는 전역 IP 등록 정보로, 개별 IP 인터페이스에 대한 전달을 관리하는 데 사용되는 것과 동일한

등록 정보 이름입니다. IPv4 또는 IPv6 프로토콜(또는 둘 다)에 대해 전달을 사용으로 설정할 수 있으므로 각 프로토콜을 개별적으로 관리해야 합니다.

예를 들어 시스템에서 모든 IPv4 및 IPv6 트래픽에 대해 패킷 전달을 다음과 같이 사용으로 설정할 수 있습니다.

```
# ipadm show-prop -p forwarding ip
PROTO   PROPERTY     PERM   CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ipv4    forwarding   rw     off       --           off       on,off
ipv6    forwarding   rw     off       --           off       on,off

# ipadm set-prop -p forwarding=on ipv4
# ipadm set-prop -p forwarding=on ipv6

# ipadm show-prop -p forwarding ip
PROTO   PROPERTY     PERM   CURRENT   PERSISTENT   DEFAULT   POSSIBLE
ipv4    forwarding   rw     on        on           off       on,off
ipv6    forwarding   rw     on        on           off       on,off
```

참고 - IP 인터페이스 또는 프로토콜의 `forwarding` 등록 정보는 배타적이지 않습니다. 인터페이스 및 프로토콜에 대한 등록 정보를 동시에 설정할 수 있습니다. 예를 들어, 프로토콜에서 전역적으로 패킷 전달을 사용으로 설정한 후 시스템에서 각 IP 인터페이스에 대한 패킷 전달을 사용자 정의할 수 있습니다. 따라서 전역적으로 사용으로 설정되었지만, 인터페이스별로 패킷 전달을 선택적으로 관리할 수도 있습니다.

기본 주소 선택 관리

Oracle Solaris에서는 한 인터페이스에서 여러 개의 IP 주소를 사용할 수 있습니다. 예를 들어 IPMP와 같은 기술이 여러 NIC(네트워크 인터페이스 카드)를 사용하여 동일한 IP 링크 계층에 연결할 수 있도록 해줍니다. 이러한 링크는 여러 개의 IP 주소를 사용할 수 있습니다. 또한 IPv6 지원 시스템의 인터페이스에는 적어도 하나의 인터페이스에 대해 링크 로컬 IPv6 주소 하나, IPv6 경로 지정 주소 하나 이상 및 IPv4 주소 하나가 포함됩니다.

시스템에서 트랜잭션이 시작되면 응용 프로그램에서 `getaddrinfo` 소켓을 호출합니다. `getaddrinfo` 소켓은 대상 시스템에서 사용 중인 가능한 주소를 검색합니다. 그러면 커널에서 이 목록의 우선 순위를 정해 패킷에 사용할 최적의 대상을 찾습니다. 이 프로세스를 대상 주소 순서 지정이라고 합니다. 패킷에 대한 최적의 대상 주소가 제공된 경우 Oracle Solaris 커널에서 소스 주소에 적합한 형식을 선택합니다. 이 프로세스를 주소 선택이라고 합니다. 대상 주소 순서 지정에 대한 자세한 내용은 `getaddrinfo`(3SOCKET) 매뉴얼 페이지를 참조하십시오.

IPv4 전용 및 듀얼 스택 IPv4/IPv6 시스템 모두 기본 주소 선택을 수행해야 합니다. 대부분의 경우에는 기본 주소 선택 방식을 변경할 필요가 없습니다. 그러나 IPMP를 지원하거나 6to4 주소 형식을 선호하는 경우 주소 형식의 우선 순위를 변경해야 할 수 있습니다.

/etc/inet/ipaddrsel.conf 구성 파일 설명

/etc/inet/ipaddrsel.conf 파일에는 IPv6 기본 주소 선택 정책 테이블이 포함되어 있습니다. IPv6이 사용 가능한 상태로 Oracle Solaris를 설치하면 이 파일에는 표 1-1. “IPv6 주소 선택 정책 테이블”에 표시된 내용이 포함됩니다.

/etc/inet/ipaddrsel.conf의 내용은 편집할 수 있습니다. 그러나 대부분의 경우 이 파일을 수정하지 않는 것이 좋습니다. 수정이 필요할 경우 IPv6 주소 선택 정책 테이블을 관리하는 방법 [13] 절차를 참조하십시오. ippaddrsel.conf에 대한 자세한 내용은 “IPv6 주소 선택 정책 테이블을 수정하는 이유” [13] and the ipaddrsel.conf(4) 매뉴얼 페이지를 참조하십시오.

ipaddrsel 명령 설명

ipaddrsel 명령을 사용하여 IPv6 기본 주소 선택 정책 테이블을 수정할 수 있습니다.

Oracle Solaris 커널은 IPv6 기본 주소 선택 정책 테이블을 사용하여 IPv6 패킷 헤더에 대한 대상 주소 순서 지정 및 소스 주소 선택을 수행합니다. /etc/inet/ipaddrsel.conf 파일에는 정책 테이블이 포함되어 있습니다.

다음 표는 정책 테이블의 기본 주소 형식 및 우선 순위를 보여줍니다. IPv6 주소 선택에 대한 기술적인 세부 정보는 inet6(7P) 매뉴얼 페이지를 참조하십시오.

표 1-1 IPv6 주소 선택 정책 테이블

접두어	우선 순위	정의
::1/128	50	루프백
::/0	40	기본값
2002::/16	30	6to4
::/96	20	IPv4 호환 가능
::ffff:0:0/96	10	IPv4

이 표에서 IPv6 접두어(::1/128 및 ::/0)가 6to4 주소(2002::/16) 및 IPv4 주소(::/96 및 ::ffff:0:0/96)보다 우선적으로 사용됩니다. 따라서 기본적으로 커널은 다른 IPv6 대상으로 이동하는 패킷에 대해 전역 IPv6 주소의 인터페이스를 선택합니다. IPv4 주소의 인터페이스는 특히 IPv6 대상으로 이동하는 패킷에 대해 낮은 우선 순위를 갖습니다. 선택한 IPv6 소스 주소가 제공될 경우, 커널에서는 대상 주소에 대해 IPv6 형식도 사용됩니다.

IPv6 주소 선택 정책 테이블을 수정하는 이유

대부분의 경우에는 IPv6 기본 주소 선택 정책 테이블을 변경할 필요가 없습니다. 정책 테이블을 관리해야 하는 경우 `ipaddrsel` 명령을 사용하십시오.

다음과 같은 이유로 정책 테이블을 수정할 수 있습니다.

- 시스템 인터페이스가 6to4 터널에 사용되는 경우, 6to4 주소에 더 높은 우선 순위를 지정할 수 있습니다.
- 특정 소스 주소를 특정 대상 주소와의 통신에만 사용하려는 경우, 이 주소를 정책 테이블에 추가하면 됩니다. 그런 다음 `ipadm` 명령을 사용하여 이 주소를 선호 주소로 플래그 지정할 수 있습니다. 자세한 내용은 `ipadm`(1M) 매뉴얼 페이지를 참조하십시오.
- IPv4 주소가 IPv6 주소보다 우선적으로 사용되게 하려는 경우, `::ffff:0:0/96`의 우선 순위를 더 높은 숫자로 변경할 수 있습니다.
- 제거된 주소에 더 높은 우선 순위를 지정해야 하는 경우, 제거된 주소를 정책 테이블에 추가하면 됩니다. 예를 들어 사이트 로컬 주소는 이제 IPv6에서 제거되었습니다. 이러한 주소의 앞에는 `fec0::/10`이 붙습니다. 사이트 로컬 주소에 더 높은 우선 순위를 지정하도록 정책 테이블을 변경할 수 있습니다.

`ipaddrsel` 명령에 대한 자세한 내용은 `ipaddrsel`(1M) 매뉴얼 페이지를 참조하십시오.

▼ IPv6 주소 선택 정책 테이블을 관리하는 방법

다음 절차는 주소 선택 정책 테이블을 수정하는 방법에 대해 설명합니다. IPv6 기본 주소 선택에 대한 개념 정보는 ipaddrsel 명령 설명을 참조하십시오.

주의 - 다음 절차에 제공된 이유 이외에는 IPv6 주소 선택 정책 테이블을 변경하지 마십시오. 그럴 경우 정책 테이블이 잘못 구성되어 네트워크 문제가 발생할 수 있습니다. 또한 이 절차에 설명된 것과 같이, 정책 테이블의 백업 복사본을 반드시 저장하십시오.

1. **관리자로 로그인합니다.**

2. **현재 IPv6 주소 선택 정책 테이블을 검토합니다.**

```
# ipaddrsel
# Prefix                Precedence Label
::1/128                         50 Loopback
::/0                            40 Default
2002::/16                       30 6to4
::/96                           20 IPv4_Compatible
::ffff:0.0.0.0/96               10 IPv4
```

3. **기본 주소 정책 테이블의 백업 복사본을 만듭니다.**

```
# cp /etc/inet/ipaddrsel.conf /etc/inet/ipaddrsel.conf.orig
```

4. **/etc/inet/ipaddrsel.conf 파일에 사용자 정의 내용을 추가합니다.**

   ```
   # pfedit /etc/inet/ipaddrsel.conf
   ```

   /etc/inet/ipaddrsel의 항목에 다음 구문을 사용합니다.

   prefix/prefix-length precedence label [# comment]

   일반적으로 수정할 수 있는 몇 가지 사항은 예 1-1. “기본 Pv6 주소 선택 정책 테이블 수정”을 참조하십시오.

5. **수정된 정책 테이블을 커널로 로드합니다.**

   ```
   # ipaddrsel -f /etc/inet/ipaddrsel.conf
   ```

6. **수정된 정책 테이블에 문제가 있는 경우 기본 IPv6 주소 선택 정책 테이블을 복원합니다.**

   ```
   # ipaddrsel -d
   ```

예 1-1 기본 Pv6 주소 선택 정책 테이블 수정

다음은 정책 테이블에 대해 일반적으로 수정할 수 있는 몇 가지 사항입니다.

- 6to4 주소에 가장 높은 우선 순위를 지정합니다.

  ```
  2002::/16                       50 6to4
  ::1/128                         45 Loopback
  ```

  이제 6to4 주소에 가장 높은 우선 순위인 50이 지정됩니다. 루프백의 경우 우선 순위가 이제 50에서 45로 변경됩니다. 기타 주소 지정 형식은 그대로 유지합니다.
- 특정 대상 주소와의 통신에 사용할 특정 소스 주소를 지정합니다.

  ```
  ::1/128                         50 Loopback
  2001:1111:1111::1/128           40 ClientNet
  2001:2222:2222::/48             40 ClientNet
  ::/0                            40 Default
  ```

  이 특정 항목은 물리적 인터페이스가 한 개뿐인 호스트에 유용합니다. 2001:1111:1111::1/128은 2001:2222:2222::/48 네트워크 내에서 대상에 대해 바운드되는 모든 패킷에 대한 소스 주소로 선호됩니다. 우선 순위 40은 소스 주소 2001:1111:1111::1/128에 대한 우선 순위로, 해당 인터페이스에 대해 구성된 다른 주소 형식보다 높습니다.
- IPv6 주소보다 IPv4 주소를 선호합니다.

  ```
  ::ffff:0.0.0.0/96               60 IPv4
  ::1/128                         50 Loopback
  .
  ```

.

IPv4 형식 `::ffff:0.0.0.0/96`의 우선 순위가 10(기본값)에서 60(테이블의 가장 높은 우선 순위)으로 변경되었습니다.

▼ 현재 세션에 대해서만 IPv6 주소 선택 정책 테이블을 수정하는 방법

`/etc/inet/ipaddrsel.conf` 파일을 편집하면 수정 사항이 시스템 재부트 후에도 지속됩니다. 수정된 정책 테이블이 현재 세션에서만 사용되도록 하려면 다음 절차를 수행하십시오.

1. **관리자로 로그인합니다.**

2. **`/etc/inet/ipaddrsel` 파일의 내용을 *filename*으로 복사합니다. 여기서 *filename*은 사용자가 선택한 파일의 이름을 나타냅니다.**

   ```
   # cp /etc/inet/ipaddrsel filename
   ```

3. **`pfedit` 명령을 사용하여 *filename*의 정책 테이블을 원하는 지정 사항으로 편집합니다.**

4. **수정된 정책 테이블을 커널로 로드합니다.**

   ```
   # ipaddrsel -f filename
   ```

   시스템을 재부트할 때까지 커널에서 새 정책 테이블을 사용합니다.

전송 계층 서비스 관리

Oracle Solaris의 표준 전송 계층 프로토콜은 TCP(전송 제어 프로토콜), SCTP(흐름 제어 전송 프로토콜) 및 UDP(사용자 데이터그램 프로토콜)입니다. 일반적으로 이러한 프로토콜은 개입 없이도 제대로 실행됩니다. 하지만 사이트의 요구 사항에 따라 이러한 전송 계층 프로토콜을 통해 실행되는 서비스를 기록하거나 수정해야 할 수도 있습니다. 이 경우 SMF(서비스 관리 기능) 명령을 사용하여 이러한 서비스에 대한 프로파일을 수정해야 합니다.

`inetd` 데몬은 시스템 부트 시 표준 인터넷 서비스를 시작합니다. 이러한 서비스에는 TCP, SCTP 또는 UDP를 전송 계층 프로토콜로 사용하는 응용 프로그램이 포함됩니다. SMF 명령을 사용하여 기존 인터넷 서비스를 수정하거나 새 서비스를 추가할 수 있습니다.

전송 계층 프로토콜과 관련된 작업은 다음과 같습니다.

- 권한이 부여된 포트 설정
- 트래픽 혼잡 제어 구현
- 모든 수신 TCP 연결 기록

- 전송 계층 프로토콜을 통해 실행되는 서비스 추가, SCTP를 예로 사용
- 액세스 제어를 위해 TCP 래퍼 기능 구성
- TCP 수신 버퍼 크기 변경

자세한 내용은 "Oracle Solaris 11.2의 시스템 서비스 관리 "의 "inetd로 제어되는 서비스 수정" 및 `inetd`(1M) 매뉴얼 페이지를 참조하십시오.

권한이 부여된 포트 설정

TCP, UDP, SCTP 등의 전송 프로토콜에서 권한이 부여된 포트는 기본적으로 1-1023 포트입니다. 권한이 부여된 포트에 바인드하려면 root 권한을 사용하여 프로세스를 실행해야 합니다. 1023 포트 이후의 포트는 기본적으로 권한이 부여되지 않은 포트입니다. ipadm 명령을 사용하여 권한이 부여된 포트의 범위를 확장하거나, 권한이 부여되지 않은 범위의 특정 포트를 권한이 부여된 포트로 표시할 수 있습니다.

권한이 부여된 포트의 범위를 관리하기 위해 다음과 같은 전송 프로토콜 등록 정보를 사용자 정의할 수 있습니다.

`smallest_nonpriv_port` 권한이 부여되지 않은 포트(일반 사용자가 바인드할 수 있는 포트)의 시작 범위를 나타내는 값을 지정합니다. 권한이 부여되지 않은 범위 내의 개별 포트를 권한이 부여된 포트로 설정할 수 있습니다. 등록 정보 값을 표시하려면 `ipadm show-prop` 명령을 사용합니다.

`extra_priv_ports` 권한이 부여된 범위 밖의 포트 중 권한을 부여할 포트를 지정합니다. 제한할 포트를 지정하려면 `ipadm set-prop` 명령을 사용합니다. 이 등록 정보에 여러 값을 지정할 수 있습니다.

예를 들어, TCP 포트 3001 및 3050을 root 역할로만 액세스가 제한된 권한이 부여된 포트로 설정한다고 가정해보십시오. smallest_nonpriv_port 등록 정보는 1024가 권한이 부여되지 않은 포트에 대한 최저 포트 번호임을 나타냅니다. 따라서 지정된 포트 3001 및 3050을 다음과 같이 권한이 부여된 포트로 변경할 수 있습니다.

```
# ipadm show-prop -p smallest_nonpriv_port tcp
PROTO PROPERTY               PERM   CURRENT   PERSISTENT   DEFAULT   POSSIBLE
tcp   smallest_nonpriv_port   rw     1024      --           1024      1024-32768

# ipadm show-prop -p extra_priv_ports tcp
PROTO   PROPERTY          PERM   CURRENT     PERSISTENT   DEFAULT     POSSIBLE
tcp     extra_priv_ports  rw     2049,4045   --           2049,4045   1-65535

# ipadm set-prop -p extra_priv_ports+=3001 tcp
# ipadm set-prop -p extra_priv_ports+=3050 tcp
# ipadm show-prop -p extra_priv_ports tcp
PROTO   PROPERTY          PERM   CURRENT     PERSISTENT   DEFAULT     POSSIBLE
tcp     extra_priv_ports  rw     2049,4045   3001,3050    2049,4045   1-65535
                                 3001,3050
```

권한이 부여된 포트(예: 4045)를 다음과 같이 제거합니다.

```
# ipadm set-prop -p extra_priv_ports-=4045 tcp
# ipadm show-prop -p extra_priv_ports tcp
PROTO    PROPERTY           PERM   CURRENT     PERSISTENT   DEFAULT     POSSIBLE
tcp      extra_priv_ports   rw     2049,3001   3001,3050    2049,4045   1-65535
                                   3050
```

트래픽 혼잡 제어 구현

네트워크 혼잡은 네트워크가 수용할 수 있는 것보다 많은 수의 패킷을 노드가 전송할 때 일반적으로 라우터 버퍼 오버플로우의 형태로 발생합니다. 여러 알고리즘에서 전송 시스템에 대한 제어를 설정하여 트래픽 혼잡을 방지합니다. 지원되는 내장 알고리즘을 설명하는 다음 표에 나와 있는 것과 같이, 이러한 알고리즘은 Oracle Solaris에서 지원되며 OS에 쉽게 추가하거나 직접 플러그인할 수 있습니다.

알고리즘	Oracle Solaris 이름	설명
NewReno	newreno	Oracle Solaris의 기본 알고리즘입니다. 이 제어 방식에는 발신자의 혼잡 윈도우, 느린 시작 및 혼잡 회피가 포함됩니다.
HighSpeed	highspeed	고속 네트워크를 위한 가장 잘 알려지고 가장 단순한 NewReno 수정 버전 중 하나입니다.
CUBIC	cubic	Linux 2.6에서 현재 기본 알고리즘입니다. 혼잡 회피 위상을 선형 윈도우 증가에서 3차 함수로 변경합니다.
Vegas	vegas	실제 패킷 손실을 트리거하지 않고 혼잡을 예측하려고 시도하는 고전적인 지연 기반 알고리즘입니다.

혼잡 제어는 다음과 같은 제어 관련 TCP 등록 정보를 설정하여 사용으로 설정됩니다. 이러한 등록 정보는 TCP용으로 나열되지만 해당 등록 정보로 사용으로 설정되는 제어 방식은 SCTP 트래픽에도 적용됩니다.

`cong_enabled` 시스템에서 현재 작동되는 알고리즘을 쉼표로 구분하여 나타냅니다. 알고리즘을 추가 또는 제거하여 사용하려는 알고리즘만 사용으로 설정할 수 있습니다. 이 등록 정보는 여러 값을 포함할 수 있습니다. 따라서 변경하려는 사항에 따라 += 수식자 또는 -= 수식자를 사용해야 합니다.

`cong_default` 응용 프로그램에서 소켓 옵션에 알고리즘을 명시적으로 적용하지 않은 경우 기본적으로 사용됩니다. `cong_default` 등록 정보의 값은 전역 영역과 비전역 영역에 모두 적용됩니다.

다음 예는 혼잡 제어 알고리즘을 프로토콜에 추가하는 방법을 보여줍니다.

```
# ipadm set-prop -p cong_enabled+=algorithm tcp
```

다음과 같이 알고리즘을 제거합니다.

```
# ipadm set-prop -p cong_enabled-=algorithm tcp
```

다음과 같이 기본 알고리즘을 바꿉니다.

```
# ipadm set-prop -p cong_default=algorithm tcp
```

참고 - 알고리즘을 추가 또는 제거할 때 따라야 할 시퀀스 규칙은 없습니다. 알고리즘을 등록 정보에 추가하기 전 알고리즘을 제거할 수 있습니다. 하지만 `cong_default` 등록 정보는 항상 알고리즘이 정의되어 있어야 합니다.

다음 예는 혼잡 제어 구현 방법을 보여줍니다. 이 예에서 TCP 프로토콜의 기본 알고리즘은 newreno에서 cubic으로 변경됩니다. 그런 후 vegas 알고리즘이 사용으로 설정된 알고리즘 목록에서 제거됩니다.

```
# ipadm show-prop -p extra_priv_ports tcp
PROTO PROPERTY              PERM CURRENT      PERSISTENT   DEFAULT      POSSIBLE
tcp   extra_priv_ports      rw   2049,4045    --           2049,4045    1-65535

# ipadm show-prop -p cong_default,cong_enabled tcp
PROTO PROPERTY              PERM CURRENT      PERSISTENT   DEFAULT      POSSIBLE
tcp   cong_default          rw   newreno      --           newreno      newreno,cubic,
                                                                       highspeed,vegas
tcp   cong_enabled          rw   newreno,cubic, newreno,cubic, newreno  newreno,cubic,
                                 highspeed,   highspeed,                highspeed,vegas
                                 vegas        vegas

# ipadm set-prop -p cong_enabled-=vegas tcp
# ipadm set-prop -p cong_default=cubic tcp

# ipadm show-prop -p cong_default,cong_enabled tcp
PROTO PROPERTY              PERM CURRENT      PERSISTENT   DEFAULT      POSSIBLE
tcp   cong_default          rw   cubic        cubic        newreno      newreno,cubic,
                                                                       highspeed
tcp   cong_enabled          rw   newreno,cubic, newreno,cubic, newreno  newreno,cubic,
                                 highspeed    highspeed                highspeed,vegas
```

모든 수신 TCP 연결의 IP 주소 기록

모든 수신 TCP 연결의 IP 주소는 `daemon.notice` 기능 및 레벨에서 `syslog` 서비스를 통해 기록됩니다. 이 정보는 로컬 `syslog.conf` 파일 설정을 변경하지 않는 한 `/var/adm/messages`에 있습니다. `syslog.conf` 파일 설정을 변경하면 이 정보가 저장되는 위치에 영향을 줄 수 있습니다. 자세한 내용은 `syslog.conf`(4) 매뉴얼 페이지를 참조하십시오.

다음과 같이 `inetd` 데몬에서 관리하는 모든 서비스에 대해 TCP 추적을 `TRUE`(사용)로 설정하십시오.

```
# inetadm -M tcp_trace=TRUE
```

SCTP 프로토콜을 사용하는 서비스 추가

SCTP(흐름 제어 전송 프로토콜) 프로토콜은 TCP와 유사한 방식으로 응용 프로그램 계층 프로토콜에 서비스를 제공합니다. 하지만 SCTP는 둘 중 하나 또는 모두가 멀티홈일 수 있는 두 시스템 간의 통신을 가능하게 합니다. SCTP 연결을 연관이라고 합니다. 연관에서 응용 프로그램은 하나 이상의 메시지 스트림(다중 스트림이라고도 함)으로 전송될 데이터를 구분합니다. SCTP 연결은 IP 주소가 여러 개인 끝점으로 이동할 수 있으므로 전화 기술 응용 프로그램에서 특히 중요합니다. 사이트에서 IP 필터 또는 IPsec를 사용하는 경우 보안상 SCTP의 멀티홈 기능을 고려해야 합니다. 이러한 고려 사항 중 몇 가지는 `sctp`(7P) 매뉴얼 페이지에서 설명됩니다.

▼ SCTP 프로토콜을 사용하는 서비스를 추가하는 방법

기본적으로 SCTP는 Oracle Solaris에 포함되어 있으므로 추가로 구성할 필요가 없습니다. 단, SCTP를 사용하도록 특정 응용 프로그램 계층 서비스를 명시적으로 구성해야 합니다. `echo` 및 `discard`가 이러한 응용 프로그램에 해당합니다. 다음 절차는 SCTP 일대일 스타일 소켓을 사용하는 echo 서비스를 추가하는 방법에 대해 설명합니다. TCP 및 UDP에도 동일한 절차를 사용하여 서비스를 추가할 수 있습니다.

다음 작업은 `inetd` 데몬에서 관리하는 SCTP `inet` 서비스를 SMF 저장소에 추가하는 방법에 대해 설명합니다. 그런 다음 SMF 명령을 사용하여 서비스를 추가하는 방법에 대해서도 설명합니다.

시작하기 전에 다음 절차를 수행하기 전에 서비스에 대한 매니페스트 파일을 만드십시오. 예를 들어 이 절차에서는 `echo` 서비스에 대한 매니페스트(`echo.sctp.xml`)를 사용합니다.

1. **시스템 파일에 대한 쓰기 권한이 있는 사용자 계정으로 로컬 시스템에 로그인합니다.**

2. **`pfedit` 명령을 사용하여 새 서비스에 대한 정의를 `/etc/services` 파일에 추가합니다.**

   `pfedit`(1M) 매뉴얼 페이지를 참조하십시오.
   서비스 정의에 대해 다음 구문을 사용합니다.

   service-name port/protocol aliases

3. **서비스 매니페스트가 저장된 디렉토리로 변경한 다음 서비스 매니페스트를 가져옵니다.**

   ```
   # cd dir-name
   # svccfg import service-manifest-name
   ```

   예를 들어 `service.dir` 디렉토리에 있는 `echo.sctp.xml` 매니페스트를 사용하여 새 SCTP `echo` 서비스를 추가합니다.

   ```
   # cd service.dir
   # svccfg import echo.sctp.xml
   ```

4. **서비스 매니페스트가 추가되었는지 확인합니다.**

```
# svcs FMRI
```

FMRI 인수로 서비스 매니페스트의 FMRI(Fault Managed Resource Identifier)를 사용합니다.

5. **수정해야 할지 여부를 결정할 서비스의 등록 정보를 나열합니다.**

```
# inetadm -l FMRI
```

6. **새 서비스를 사용으로 설정합니다.**

```
# inetadm -e FMRI
```

7. **서비스가 사용으로 설정되었는지 확인합니다.**

예 1-2 SCTP 전송 프로토콜을 사용하는 서비스 추가

다음 예는 사용할 명령 및 echo 서비스가 SCTP를 사용하도록 하는 데 필요한 파일 항목을 보여 줍니다.

```
# cat /etc/services
.
.
echo            7/tcp
echo            7/udp
echo             7/sctp

# cd service.dir

# svccfg import echo.sctp.xml

# svcs network/echo*
STATE          STIME    FMRI
disabled       15:46:44 svc:/network/echo:dgram
disabled       15:46:44 svc:/network/echo:stream
disabled       16:17:00 svc:/network/echo:sctp_stream

# inetadm -l svc:/network/echo:sctp_stream
SCOPE    NAME=VALUE
         name="echo"
         endpoint_type="stream"
         proto="sctp"
         isrpc=FALSE
         wait=FALSE
         exec="/usr/lib/inet/in.echod -s"
         user="root"
default  bind_addr=""
default  bind_fail_max=-1
default  bind_fail_interval=-1
default  max_con_rate=-1
default  max_copies=-1
default  con_rate_offline=-1
```

```
default  failrate_cnt=40
default  failrate_interval=60
default  inherit_env=TRUE
default  tcp_trace=FALSE
default  tcp_wrappers=FALSE

# inetadm -e svc:/network/echo:sctp_stream

# inetadm | grep echo
disabled  disabled       svc:/network/echo:stream
disabled  disabled       svc:/network/echo:dgram
enabled   online         svc:/network/echo:sctp_stream
```

TCP 래퍼 구성

TCP 래퍼는 데몬과 수신 서비스 요청 사이에서 서비스 데몬에 대한 보안 조치를 추가합니다. 또한 연결 시도 성공 및 실패를 기록합니다. TCP 래퍼는 요청 시작 위치에 따라 연결을 허용하거나 거부하여 액세스 제어를 제공할 수도 있습니다. TCP 래퍼를 사용하여 SSH(보안 셸), 텔넷 및 FTP(File Transfer Protocol) 등의 데몬을 보호할 수 있습니다. `sendmail` 응용 프로그램은 “Oracle Solaris 11.2에서의 sendmail 서비스 관리 ”의 “sendmail 버전 8.12의 TCP 래퍼에 대한 지원”에 설명된 대로 TCP 래퍼를 사용할 수 있습니다.

▼ TCP 래퍼를 사용하여 TCP 서비스에 대한 액세스를 제어하는 방법

`tcpd` 프로그램은 *TCP* 래퍼를 구현합니다.

1. **`root` 역할로 전환합니다.**
2. **TCP 래퍼를 사용으로 설정합니다.**

   ```
   # inetadm -M tcp_wrappers=TRUE
   ```

3. **TCP 래퍼 액세스 제어 정책을 구성합니다.**

   지침은 `hosts_access`(3) 매뉴얼 페이지(`/usr/sfw/man` 디렉토리에 있음)를 참조하십시오.

TCP 수신 버퍼 크기 변경

TCP 수신 버퍼의 크기는 TCP 등록 정보 `recv_buf`(기본값 128KB)를 사용하여 설정됩니다. 하지만 응용 프로그램은 사용 가능한 대역폭을 동일하게 사용하지 않습니다. 따라서 연결 대기 시간에 따라 기본 크기를 변경해야 할 수 있습니다. 예를 들어, Oracle Solaris의 보안 셸 기능을 사용하면 데이터 스트림에서 수행되는 추가 체크섬 및 암호화 프로세스로 인해 대역폭 사용에 대한 오버헤드가 발생합니다. 따라서 버퍼 크기를 늘려야 할 수 있습니다. 마찬가

지로, 대량 전송을 수행하는 응용 프로그램이 대역폭을 효율적으로 사용하도록 하려면 동일한 버퍼 크기 조정도 필요합니다.

BDP(대역폭 지연 곱)를 예상하여 사용할 올바른 수신 버퍼 크기를 계산할 수 있습니다. BDP를 계산하려면 사용 가능한 대역폭에 연결 대기 시간 값을 곱하십시오.

연결 대기 시간 값을 가져오려면 `ping -s` *host* 명령을 사용하십시오.

적합한 수신 버퍼 크기는 BDP 값에 가깝습니다. 하지만 대역폭 사용도 다양한 옵션에 따라 달라질 수 있습니다. 공유 기반구조 또는 응용 프로그램 수 및 대역폭 사용을 경합하는 사용자로 인해 예상값이 변경될 수 있습니다.

다음과 같이 버퍼 크기 값을 변경합니다.

```
# ipadm set-prop -p recv_buf=value tcp
```

다음 예는 버퍼 크기를 164KB로 늘리는 방법을 보여줍니다.

```
# ipadm show-prop -p recv_buf tcp
PROTO PROPERTY   PERM CURRENT   PERSISTENT   DEFAULT  POSSIBLE
tcp   recv_buf   rw   128000       --        128000   2048-1048576

# ipadm set-prop -p recv_buf=164000 tcp

# ipadm show-prop -p recv_buf tcp
PROTO PROPERTY   PERM CURRENT   PERSISTENT   DEFAULT  POSSIBLE
tcp   recv_buf   rw   164000    164000       128000   2048-1048576
```

선호 크기는 상황에 따라 달라지므로 버퍼 크기에 대해 선호되는 설정 값은 없습니다. 특정 조건의 각 네트워크에서 서로 다른 BDP 값이 설정되는 다음 예를 고려하십시오.

버퍼 크기의 기본값이 128KB인 일반적인 1Gbps LAN(Local Area Network):	`BDP = 128 MBps * 0.001 s = 128 kB`
대기 시간이 100밀리초인 이론적 1Gbps WAN(Wide Area Network):	`BDP = 128 MBps * 0.1 s = 12.8 MB`
유럽-미국 링크(`uperf`로 측정된 대역폭)	`BDP = 2.6 MBps * 0.175 = 470 kB`

BDP를 계산할 수 없으면 다음 지침을 따르십시오.

- LAN 기반 대량 전송의 경우 버퍼 크기의 기본값(128KB)으로도 충분합니다.
- 대부분의 WAN 배포에서 수신 버퍼 크기는 2MB 범위 내에 있어야 합니다.

주의 - TCP 수신 버퍼 크기를 늘리면 여러 네트워크 응용 프로그램의 메모리 사용 공간이 늘어납니다.

netstat 명령으로 네트워크 상태 모니터링

netstat 명령은 네트워크 상태 및 프로토콜 통계를 표시하는 화면을 생성하는 데 사용됩니다. TCP, SCTP 및 UDP 끝점의 상태를 표 형식으로 표시하고 경로 지정 테이블 및 인터페이스 정보를 표시할 수 있습니다.

netstat 명령은 지정한 명령줄 옵션에 따라 다양한 유형의 네트워크 데이터를 표시합니다. 표시되는 정보는 시스템 관리에 특히 유용합니다. 이 장에서는 보다 일반적으로 수행되는 일부 작업에 사용되는 옵션에 대해 설명합니다. 자세한 내용은 netstat(1M) 매뉴얼 페이지를 참조하십시오.

이 절은 다음 항목으로 구성됩니다.

주소 유형으로 netstat 출력 필터링

netstat 명령을 사용하여 IPv4 및 IPv6 네트워크 상태를 모두 표시할 수 있습니다. /etc/default/inet_type 파일에서 DEFAULT_IP 값을 설정하거나 -f 옵션을 사용하여 표시할 프로토콜 정보를 선택할 수 있습니다. DEFAULT_IP 값을 영구적으로 설정하면 netstat 명령으로 IPv4 정보만 표시됩니다. 이 설정을 대체하려면 명령줄에서 -f 옵션을 사용하십시오. inet_type 파일에 대한 자세한 내용은 inet_type(4) 매뉴얼 페이지를 참조하십시오.

-f 옵션을 사용하여 inet, inet6, unix(내부 통신에 사용되는 Unix 도메인 소켓의 경우) 또는 sdp(Socket Description Protocol) 인수를 지정할 수도 있습니다.

소켓 상태 표시

netstat 명령을 사용하여 로컬 호스트에 있는 소켓의 상태를 확인할 수 있습니다. 일반 사용자가 netstat 명령 옵션을 여러 개 지정할 수 있습니다.

다음과 같이 연결 소켓의 상태를 표시합니다.

```
% netstat
```

다음과 같이 연결되지 않은 리소스 소켓을 포함하여 모든 소켓의 상태를 표시합니다.

```
% netstat -a
```

예 1-3 연결된 소켓 수

netstat 명령의 출력에는 광범위한 통계가 표시됩니다. 다음 예는 출력을 IPv4 소켓으로만 제한하는 방법을 보여줍니다.

```
% netstat -f inet

TCP: IPv4
   Local Address        Remote Address    Swind Send-Q Rwind Recv-Q    State
-------------------- -------------------- ----- ------ ----- ------ -----------
system-1.ssh          remote.38474     128872      0 128872      0 ESTABLISHED
system2.40721         remote.ldap      49232       0 128872      0 ESTABLISHED
```

프로토콜별 통계 표시

netstat -s 옵션은 UDP, TCP, SCTP, ICMP(Internet Control Message Protocol) 및 IP 프로토콜에 대한 프로토콜 통계를 표시합니다.

다음과 같이 프로토콜 상태를 표시합니다.

```
% netstat -s
```

-P 옵션을 사용하여 프로토콜을 기준으로 netstat 명령의 출력을 필터링할 수 있습니다. 이 옵션은 전송 프로토콜로 제한되지 않습니다.

이 옵션을 사용하여 다음 값을 지정할 수 있습니다.

- icmp
- icmpv6
- igmp
- ipv6tcp
- rawip
- sctp
- tcp
- udp

예를 들어 다음과 같이 UDP 프로토콜을 기준으로 netstat 출력을 필터링합니다.

```
% netstat -s -P udp

UDP     udpInDatagrams      =  3704     udpInErrors         =     0
        udpOutDatagrams     =  3703     udpOutErrors        =     0
```

예 1-4 TCP 프로토콜의 상태 표시

다음 예는 -P 옵션을 지정하여 TCP 프로토콜에 대한 결과를 표시하는 방법을 보여줍니다.

```
% netstat -P tcp

TCP: IPv4
Local Address     Remote Address    Swind Send-Q  Rwind Recv-Q      State
----------------- -------------------- ----- ------  ----- ------     -------
lhost-1.login      abc.def.local.Sun.COM.980 49640      0     49640    0 ESTABLISHED
lhost.login        ghi.jkl.local.Sun.COM.1020 49640     1     49640    0 ESTABLISHED
remhost.1014       mno.pqr.remote.Sun.COM.nfsd 49640    0     49640    0 TIME_WAIT

TCP: IPv6
Local Address    Remote Address        Swind Send-Q Rwind Recv-Q   State If
---------------- --------------------- ------ ----- ------ ----------- -----
localhost.38983   localhost.32777       49152      0 49152      0 ESTABLISHED
localhost.32777   localhost.38983       49152      0 49152      0 ESTABLISHED
localhost.38986   localhost.38980       49152      0 49152      0 ESTABLISHED
```

네트워크 인터페이스 상태 표시

netstat 명령의 i 옵션을 사용하여 로컬 시스템에 구성된 네트워크 인터페이스의 상태를 표시할 수 있습니다. 이 옵션을 사용하면 시스템이 각 네트워크에서 전송하고 수신하는 패킷 수를 확인할 수 있습니다.

다음과 같이 네트워크에 있는 인터페이스의 상태를 표시합니다.

```
% netstat -i
```

다음 예는 netstat 명령을 필터링 옵션과 함께 사용하여 특정 인터페이스의 출력을 제한하는 방법을 보여줍니다.

```
% netstat -i -I net0 -f inet
  Name  Mtu  Net/Dest      Address        Ipkts  Ierrs Opkts  Oerrs Collis Queue
  net0  1500 abc.oracle.com abc.oracle.com 231001 0     55856  0     0      0
```

예 1-5 네트워크 인터페이스 상태 표시

다음 예는 호스트 인터페이스를 통해 IPv4 및 IPv6 패킷 플로우의 상태를 보여줍니다. 예를 들어 서버에 대해 표시되는 입력 패킷 수(Ipkts)는 클라이언트를 부트하려고 할 때마다 늘어나지만, 출력 패킷 수(Opkts)는 그대로 유지됩니다. 이 결과에는 서버가 클라이언트에서 보내는 부트 요청 패킷을 인식하고 있는 것으로 표시됩니다. 그러나 서버가 이에 응답하는 방법을 알지 못합니다. 이러한 혼동은 hosts 또는 ethers 데이터베이스의 주소가 잘못되었기 때문일 수 있습니다.

그러나 시간이 경과해도 입력 패킷 수가 일정할 경우 시스템에서는 패킷을 인식하지 못합니다. 이 출력은 다른 유형의 오류(하드웨어 문제)를 보여줍니다.

```
Name  Mtu  Net/Dest      Address        Ipkts  Ierrs Opkts  Oerrs Collis Queue
lo0   8232 loopback      localhost      142    0     142    0     0      0
net0  1500 host58        host58        1106302 0     52419  0     0      0

Name  Mtu  Net/Dest      Address                   Ipkts  Ierrs Opkts  Oerrs Collis
lo0   8252 localhost     localhost                  142    0     142    0     0
net0  1500 fe80::a00:20ff:feb9:4c54/10 fe80::a00:20ff:feb9:4c54 1106305 0 52422 0  0
```

사용자 및 프로세스 정보 표시

netstat 명령에는 새 -u 옵션이 포함됩니다. 이 옵션은 특정 포트를 사용하고 있는 시스템의 사용자 및 프로세스에 대한 정보를 제공합니다. netstat -u 명령을 사용하면 사용자, 프로세스 ID 및 네트워크 끝점을 만든 프로그램 또는 네트워크 끝점을 현재 제어하는 프로그램을 표시할 수 있습니다.

netstat 명령 출력을 보다 잘 정렬하기 위해 -n 옵션을 -u 옵션과 함께 지정할 수 있습니다. 자세한 내용 및 자세한 예는 netstat(1M) 매뉴얼 페이지를 참조하십시오.

알려진 경로의 상태 표시

netstat 명령과 함께 -r 옵션을 사용하여 로컬 호스트의 경로 지정 테이블을 표시할 수 있습니다. 이 표는 호스트에 알려진 모든 경로의 상태를 보여줍니다.

```
% netstat -r
```

예 1-6 netstat 명령으로 경로 지정 테이블 출력 표시

다음 예는 netstat -r 명령의 출력을 보여줍니다.

```
Routing Table: IPv4
Destination           Gateway           Flags  Ref   Use   Interface
-------------------- -------------------- ----- ----- ------ ---------
host15               myhost               U         1  31059  net0
10.0.0.14            myhost               U         1      0  net0
default              distantrouter        UG        1      2  net0
localhost            localhost            UH        42019361  lo0

Routing Table: IPv6
Destination/Mask      Gateway                           Flags Ref  Use   If
--------------------- --------------------------          ----- --- ------ -----
2002:0a00:3010:2::/64 2002:0a00:3010:2:1b2b:3c4c:5e6e:abcd U     1   0      net0:1
fe80::/10             fe80::1a2b:3c4d:5e6f:12a2            U     1   23     net0
ff00::/8              fe80::1a2b:3c4d:5e6f:12a2            U     1   0      net0
default               fe80::1a2b:3c4d:5e6f:12a2            UG    1   0      net0
localhost             localhost                            UH    9   21832  lo0
```

다음 표는 netstat -r 명령의 출력에 표시되는 정보에 대해 설명합니다.

매개변수	설명
Destination	경로의 대상 끝점인 호스트를 지정합니다. IPv6 경로 지정 테이블은 6to4 터널 끝점(2002:0a00:3010:2::/64)의 접두어를 경로 대상 끝점으로 표시합니다.
Destination/Mask	
Gateway	패킷 전송에 사용할 게이트웨이를 지정합니다.
Flags	경로의 현재 상태를 나타냅니다. U 플래그는 경로가 작동 중임을 나타냅니다. G 플래그는 경로가 게이트웨이임을 나타냅니다.
Use	전송된 패킷 수를 표시합니다.
Interface	전송의 소스 끝점인 로컬 호스트의 특정 인터페이스를 나타냅니다.

netstat 명령으로 추가 네트워크 상태 표시

netstat 명령을 다양한 다른 명령줄 옵션과 함께 사용하여 이 장에 설명된 네트워크 상태 이외의 추가 네트워크 상태를 표시할 수 있습니다. 10가지 형식의 명령이 있으며, 각 형식은 네트워킹 부속 시스템의 여러 부분에 대한 서로 다른 통계 테이블을 생성합니다.

추가로 얻을 수 있는 몇 가지 통계는 다음과 같습니다.

netstat -g	멀티캐스트 그룹 멤버쉽을 표시합니다.
netstat -P	net-to-media 테이블: ARP 및 NDP 매핑을 표시합니다.
netstat -m	STREAMS 메모리 통계를 표시합니다.
netstat -M	멀티캐스트 경로 지정 테이블을 표시합니다.
netstat -D	DHCP 임대의 상태를 표시합니다.
netstat -d	대상 캐시 테이블을 표시합니다.

자세한 내용은 netstat(1M) 매뉴얼 페이지를 참조하십시오.

netcat 유틸리티로 TCP 및 UDP 관리 수행

netcat(nc) 유틸리티를 사용하여 TCP 또는 UDP 관리와 연관된 다양한 작업을 수행할 수 있습니다. IPv4 및 IPv6 네트워크 둘 다에 명령을 사용할 수 있습니다.

netcat 유틸리티를 사용하여 다음 작업을 수행할 수 있습니다.

- TCP 연결 열기
- UDP 패킷 전송
- 임의의 TCP 및 UDP 포트 수신
- 포트 스캔 수행

각각의 오류 메시지가 개별적으로 표준 출력에 표시되는 `telnet` 명령과 달리, `nc` 스크립트로 생성된 오류 메시지는 하나의 표준 오류에 결합되므로 훨씬 효율적입니다.

`netcat` 유틸리티는 새 `-M` 옵션을 지원합니다. 이 옵션은 소켓 SLA(서비스 단계 계약) 등록 정보별로 지정할 수 있습니다. 적합한 등록 정보를 `-M` 옵션과 함께 지정하면 소켓에 대한 MAC 플로우가 만들어집니다. 예를 들어 다음과 같이 `-M` 옵션을 사용할 수 있습니다.

```
% nc -M maxbw=50M host.example.com 7777
% nc -l -M priority=high,inherit=on 2222
```

위 예에 표시된 것과 같이, `-M` 옵션을 사용하여 SLA 등록 정보의 *name=value* 쌍을 쉼표로 구분하여 지정할 수 있습니다.

일부 설치 방법의 경우 `netcat` 소프트웨어 패키지가 기본적으로 설치되지 않습니다. 다음과 같이 패키지가 시스템에 설치되었는지 확인합니다.

```
% pkg list network/netcat
```

패키지가 설치되지 않은 경우 다음과 같이 패키지를 설치합니다.

```
% pfexec pkg install pkg:/network/netcat
```

자세한 내용은 `netcat`(1) 매뉴얼 페이지를 참조하십시오.

네트워크 상태 화면 관리 및 기록

다음 작업은 잘 알려진 네트워킹 명령을 사용하여 네트워크의 상태를 확인하는 방법에 대해 설명합니다.

▼ IP 관련 명령의 화면 출력을 제어하는 방법

IPv4 정보만 표시하거나 IPv4와 IPv6 정보를 모두 표시하도록 `netstat` 명령의 출력을 제어할 수 있습니다.

1. **`/etc/default/inet_type` 파일을 만듭니다.**
2. **다음 항목 중에서 네트워크에 필요한 항목을 `/etc/default/inet_type` 파일에 추가합니다.**
   - **IPv4 정보만 표시하려면 다음 변수를 설정합니다.**

```
DEFAULT_IP=IP_VERSION4
```

- **IPv4 및 IPv6 정보를 모두 표시하려면 다음 변수 중 하나를 설정합니다.**

```
DEFAULT_IP=BOTH

DEFAULT_IP=IP_VERSION6
```

자세한 내용은 inet_type(4) 매뉴얼 페이지를 참조하십시오.

참고 - netstat 명령의 -f 옵션은 inet_type 파일에 설정된 값을 대체합니다.

예 1-7 IPv4 및 IPv6 정보를 표시하도록 출력 제어

다음 예는 inet_type 파일에 DEFAULT_IP=BOTH 또는 DEFAULT_IP=IP_VERSION6 변수를 지정한 경우 출력을 보여줍니다.

```
# ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
net0: flags=100001004843<UP,BROADCAST,RUNNING,MULTICAST,DHCP,IPv4,PHYSRUNNING> mtu 1500 index
 603
        inet 10.46.86.54 netmask ffffff00 broadcast 10.46.8.255
        ether 0:1e:68:49:98:80
lo0: flags=2002000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv6,VIRTUAL> mtu 8252 index 1
        inet6 ::1/128
net0: flags=120002000841<UP,RUNNING,MULTICAST,IPv6,PHYSRUNNING> mtu 1500 index 603
        inet6 fe80::21e:68ff:fe49:9880/10
        ether 0:1e:68:49:98:80
net0:1: flags=120002080841<UP,RUNNING,MULTICAST,ADDRCONF,IPv6,PHYSRUNNING> mtu 1500 index 603
        inet6 2001:b400:414:6059:21e:68ff:fe49:9880/64
```

inet_type 파일에 DEFAULT_IP=IP_VERSION4 변수를 지정할 경우 다음과 같은 출력이 표시되어야 합니다.

```
# ifconfig -a
lo0: flags=2001000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4,VIRTUAL> mtu 8232 index 1
        inet 127.0.0.1 netmask ff000000
net0: flags=100001004843<UP,BROADCAST,RUNNING,MULTICAST,DHCP,IPv4,PHYSRUNNING> mtu 1500 index
 603
        inet 10.46.86.54 netmask ffffff00 broadcast 10.46.8.255
        ether 0:1e:68:49:98:80
```

IPv4 경로 지정 데몬의 작업 기록

IPv4 경로 지정 데몬 routed의 오작동이 의심되는 경우 데몬의 작업을 추적하는 로그를 시작할 수 있습니다. routed 데몬이 시작되면 이 로그에는 모든 패킷 전송이 포함됩니다.

다음과 같이 경로 지정 데몬의 작업을 추적하는 로그 파일을 만듭니다.

```
# /usr/sbin/in.routed /var/log-file-name
```

주의 - 사용량이 많은 네트워크에서는 이 명령이 거의 연속적으로 출력을 생성할 수 있습니다.

다음 예는 "IPv4 경로 지정 데몬의 작업 기록" [29] 절차를 수행할 때 만들어지는 로그의 시작 부분을 보여줍니다.

```
-- 2003/11/18 16:47:00.000000 --
Tracing actions started
RCVBUF=61440
Add interface lo0  #1   127.0.0.1      -->127.0.0.1/32
<UP|LOOPBACK|RUNNING|MULTICAST|IPv4> <PASSIVE>
Add interface net0 #2   10.10.48.112    -->10.10.48.0/25
<UP|BROADCAST|RUNNING|MULTICAST|IPv4>
turn on RIP
Add    10.0.0.0        -->10.10.48.112      metric=0  net0  <NET_SYN>
Add    10.10.48.85/25  -->10.10.48.112      metric=0  net0  <IF|NOPROP>
```

▼ IPv6 Neighbor Discovery 데몬의 작업을 추적하는 방법

IPv6 `in.ndpd` 데몬의 오작동이 의심되는 경우 데몬의 작업을 추적하는 로그를 시작할 수 있습니다. 이 추적은 종료될 때까지 표준 출력에 표시됩니다. `in.ndpd` 데몬이 시작되면 이 추적에는 모든 패킷 전송이 포함됩니다.

1. **`in.ndpd` 데몬의 추적을 시작합니다.**

   ```
   # /usr/lib/inet/in.ndpd -t
   ```

2. **필요한 경우 Ctrl-C를 눌러 추적을 종료합니다.**

예 1-8 `in.ndpd` 데몬 추적

다음 출력은 `in.ndpd` 데몬 추적의 시작 부분을 보여줍니다.

```
# /usr/lib/inet/in.ndpd -t
Nov 18 17:27:28 Sending solicitation to  ff02::2 (16 bytes) on net0
Nov 18 17:27:28         Source LLA: len 6 <08:00:20:b9:4c:54>
Nov 18 17:27:28 Received valid advert from fe80::a00:20ff:fee9:2d27 (88 bytes) on net0
Nov 18 17:27:28         Max hop limit: 0
Nov 18 17:27:28         Managed address configuration: Not set
Nov 18 17:27:28         Other configuration flag: Not set
Nov 18 17:27:28         Router lifetime: 1800
Nov 18 17:27:28         Reachable timer: 0
Nov 18 17:27:28         Reachable retrans timer: 0
Nov 18 17:27:28         Source LLA: len 6 <08:00:20:e9:2d:27>
Nov 18 17:27:28         Prefix: 2001:08db:3c4d:1::/64
Nov 18 17:27:28                 On link flag:Set
```

```
Nov 18 17:27:28                 Auto addrconf flag:Set
Nov 18 17:27:28                 Valid time: 2592000
Nov 18 17:27:28                 Preferred time: 604800
Nov 18 17:27:28         Prefix: 2002:0a00:3010:2::/64
Nov 18 17:27:28                 On link flag:Set
Nov 18 17:27:28                 Auto addrconf flag:Set
Nov 18 17:27:28                 Valid time: 2592000
Nov 18 17:27:28                 Preferred time: 604800
```

ping 명령으로 원격 호스트 확인

ping 명령을 사용하면 시스템이 원격 호스트와 통신할 수 있는지 여부를 확인할 수 있습니다. ping 명령을 실행하면 ICMP 프로토콜에서 지정된 호스트로 데이터그램을 전송하여 응답을 요청합니다. ICMP는 TCP/IP 네트워크에서 오류 처리를 담당하는 프로토콜입니다. ping 명령을 사용할 경우 사용자의 호스트와 지정된 원격 호스트 간에 IP 패킷을 교환할 수 있는지 여부를 확인할 수 있습니다.

다음은 ping 명령의 기본 구문입니다.

/usr/sbin/ping *host* **[***timeout***]**

여기서 *host*는 원격 호스트의 이름입니다. 선택적 *timeout* 인수는 ping 명령이 계속해서 원격 호스트에 연결하려고 시도하는 시간(초)을 나타냅니다. 기본값은 20초입니다. 자세한 내용은 ping(1M) 매뉴얼 페이지를 참조하십시오.

IPv6 지원을 위한 ping 명령 수정 사항

ping 명령은 IPv4 및 IPv6 프로토콜을 모두 사용하여 대상 호스트를 프로브합니다. 이름 서버가 지정된 대상 호스트에 대해 반환하는 주소에 따라 프로토콜 선택이 달라집니다. 기본적으로 이름 서버가 대상 호스트에 대해 IPv6 주소를 반환하는 경우 ping 명령은 IPv6 프로토콜을 사용합니다. 이름 서버가 IPv4 주소만 반환하는 경우 ping 명령은 IPv4 프로토콜을 사용합니다. -A 옵션으로 사용할 프로토콜을 지정하여 이 동작을 대체할 수 있습니다.

자세한 내용은 ping(1M) 매뉴얼 페이지를 참조하십시오.

원격 호스트에 연결할 수 있는지 확인

원격 호스트에 연결할 수 있는지 확인하려면 다음과 같이 ping 명령을 사용합니다.

% **ping** *hostname*

호스트가 ICMP 전송을 허용하는 경우 다음 메시지가 표시됩니다.

hostname is alive

이 메시지는 호스트가 ICMP 요청에 응답했음을 나타냅니다. 그러나 호스트가 작동 중지되었거나 ICMP 패킷을 수신할 수 없는 경우 또는 사용자의 호스트와 원격 호스트 간에 경로 지정 문제가 있는 경우, ping 명령으로부터 다음과 같은 응답이 수신됩니다.

```
no answer from hostname
```

사용자의 호스트와 원격 호스트 간 패킷이 삭제되었는지 확인

패킷이 손실될 경우 삭제된 데이터를 다시 전송하는 데 추가 시간이 걸리므로 네트워크 연결이 느려질 수 있습니다. ping 명령의 -s 옵션을 사용하여 사용자의 호스트와 원격 호스트 간 패킷이 삭제되었는지 확인할 수 있습니다.

```
% ping -s hostname
```

다음 예에서 ping -s *hostname* 명령은 사용자가 인터럽트 문자를 전송하거나 시간 초과가 발생할 때까지 계속해서 패킷을 지정된 호스트로 전송합니다.

```
% ping -s host1.domain8
PING host1.domain8 : 56 data bytes
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=0. time=1.67 ms
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=1. time=1.02 ms
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=2. time=0.986 ms
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=3. time=0.921 ms
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=4. time=1.16 ms
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=5. time=1.00 ms
64 bytes from host1.domain8.COM (172.16.83.64): icmp_seq=5. time=1.980 ms

^C

----host1.domain8  PING Statistics----
7 packets transmitted, 7 packets received, 0% packet loss
round-trip (ms)  min/avg/max/stddev = 0.921/1.11/1.67/0.26
```

패킷 손실 통계는 호스트에서 패킷이 삭제되었는지 여부를 나타냅니다.ping 명령이 패킷 손실을 나타낼 경우, ipadm 및 netstat 명령을 사용하여 네트워크의 상태를 확인하십시오. 자세한 내용은 “Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 ”의 “IP 인터페이스 및 주소 모니터링” 및 “netstat 명령으로 네트워크 상태 모니터링” [23]을 참조하십시오.

traceroute 명령으로 경로 지정 정보 표시

traceroute 명령은 원격 시스템에 대한 IP 패킷의 경로를 추적합니다. traceroute 명령을 사용하면 잘못된 경로 지정 구성 및 경로 지정 경로 오류를 찾을 수 있습니다. 특정 호스트에 연결할 수 없는 경우 traceroute를 사용하여 원격 호스트에 대한 패킷 경로 및 오류가 발생할 수 있는 위치를 확인할 수 있습니다.

traceroute 명령은 대상 호스트에 대한 경로를 따라 전송하는 각 게이트웨이에 대한 라운드 트립 시간도 표시합니다. 이 정보는 두 노드 간의 네트워크 트래픽이 느려지는 위치를 분석하는 데 유용할 수 있습니다.

traceroute 명령에 대한 자세한 내용은 traceroute(1M) 매뉴얼 페이지를 참조하십시오.

이 절은 다음 항목으로 구성됩니다.

IPv6 지원을 위한 traceroute 명령 수정 사항

traceroute 명령을 사용하여 특정 호스트에 대해 IPv4 및 IPv6 경로를 추적할 수 있습니다. 프로토콜 관점에서 traceroute 명령은 ping 명령과 동일한 알고리즘을 사용합니다. 이 동작을 대체하려면 -A 명령줄 옵션을 사용하십시오. 또한 -a 명령줄 옵션을 사용하면 멀티홈 호스트의 각 주소에 대해 개별 경로를 추적할 수 있습니다.

원격 호스트에 대한 경로 찾기

다음과 같이 traceroute 명령을 사용하여 원격 시스템에 대한 경로를 찾을 수 있습니다.

```
% traceroute destination-hostname
```

traceroute 명령의 다음 출력은 로컬 호스트 nearhost에서 원격 시스템 farhost로 전송되는 패킷의 7홉 경로를 보여줍니다. 이 출력에는 패킷이 각 홉을 순회하는 데 걸린 시간도 표시됩니다.

```
% traceroute farhost
traceroute to farhost (172.16.64.39), 30 hops max, 40 byte packets
1  frbldg7c-86 (172.16.86.1)  1.516 ms  1.283 ms  1.362 ms
2  bldg1a-001 (172.16.1.211)  2.277 ms  1.773 ms  2.186 ms
3  bldg4-bldg1 (172.16.4.42)  1.978 ms  1.986 ms  13.996 ms
4  bldg6-bldg4 (172.16.4.49)  2.655 ms  3.042 ms  2.344 ms
5  ferbldg11a-001 (172.16.1.236)  2.636 ms  3.432 ms  3.830 ms
6  frbldg12b-153 (172.16.153.72)  3.452 ms  3.146 ms  2.962 ms
7  farhost (172.16.64.39)  3.430 ms  3.312 ms  3.451 ms
```

모든 경로 추적

로컬 시스템에서 traceroute 명령을 -a 옵션과 함께 사용하여 모든 경로를 추적할 수 있습니다.

```
% traceroute -a host-name
```

다음 예는 이중 스택 호스트에 대해 가능한 모든 경로를 보여줍니다.

```
% traceroute -a v6host
traceroute: Warning: Multiple interfaces found; using 2001:db8:4a3a:1:56:a0:a8 @ net0:2
 traceroute to v6host (2001:db8:4a3b:5:102:a00:fe79:19b0),30 hops max, 60 byte packets
1 v6-rout86 (2001:db8:4a3b:1:56:a00:fe1f:59a1) 35.534 ms 56.998 ms *
2 2001:db8::255:0:c0a8:717 32.659 ms 39.444 ms *
3  farhost (2001:db8:4a3b:2:103:a00:fe9a:ce7b)  401.518 ms  7.143 ms *
4  distant (2001:db8:4a3b:3:100:a00:fe7c:cf35)  113.034 ms  7.949 ms *
5  v6host (2001:db8:4a3b:5:102:a00:fe79:19b0)  66.111 ms *  36.965 ms *

traceroute to v6host  (192.168.10.75),30 hops max,40 byte packets
1  v6-rout86 (172.16.86.1)  4.360 ms  3.452 ms  3.479 ms
2  flrmpj17u (172.16.17.131)  4.062 ms  3.848 ms  3.505 ms
3  farhost (10.0.0.23)  4.773 ms *  4.294 ms
4  distant (192.168.10.104)  5.128 ms  5.362 ms *
5  v6host  (192.168.15.85)  7.298 ms  5.444 ms *
```

TShark 및 Wireshark 분석기로 네트워크 트래픽 분석

*TShark*는 명령줄 네트워크 트래픽 분석기로, 디코딩된 형식의 패킷을 표준 출력에 인쇄하거나 패킷을 파일로 작성하는 방식으로 라이브 네트워크에서 패킷 데이터를 캡처하거나 이전에 저장된 캡처 파일에서 패킷을 읽을 수 있도록 해줍니다. 옵션을 사용하지 않을 경우 TShark는 `tcpdump` 명령과 유사하게 작동하며 동일한 라이브 캡처 파일 형식 `libpcap`을 사용합니다. 또한 TShark는 Wireshark에서 지원하는 것과 동일한 캡처 파일을 감지하고 읽고 쓸 수 있습니다.

*Wireshark*는 네트워크 트래픽을 대화식으로 덤프하고 분석하는 데 사용되는 타사 GUI(그래픽 사용자 인터페이스) 네트워크 프로토콜 분석기입니다. `snoop` 명령과 유사하게, Wireshark를 사용하여 라이브 네트워크 또는 이전에 저장한 캡처 파일에서 패킷 데이터를 찾아볼 수 있습니다. 기본적으로 Wireshark는 파일 캡처에 `libpcap` 형식을 사용하는데, 이 형식은 `tcpdump` 유틸리티 및 기타 유사한 도구에서도 사용합니다. Wireshark를 사용하는 주요 이점은 `libpcap` 형식 이외에도 여러 다른 파일 형식을 읽고 가져올 수 있다는 점입니다.

TShark와 Wireshark 모두 다음을 포함한 여러 고유한 기능을 제공합니다.

- TCP 대화의 모든 패킷을 어셈블하고 해당 대화의 데이터를 ASCII, EBCDIC 또는 16진수 형식으로 표시할 수 있습니다.
- 다른 네트워크 프로토콜 분석기보다 필터링 가능한 필드가 더 많이 포함되어 있습니다.
- 필터를 만드는 데 다른 네트워크 프로토콜 분석기보다 다양한 구문을 사용합니다.

Oracle Solaris 시스템에서 TShark 및 Wireshark를 사용하려면 먼저 소프트웨어 패키지가 설치되어 있는지 확인한 다음 필요한 경우 다음과 같이 설치합니다.

```
# pkg install tshark

# pkg install wireshark
```

자세한 내용은 `tshark`(1) 및 `wireshark`(1) 매뉴얼 페이지를 참조하십시오.

Wireshark 설명서(http://www.wireshark.org/)도 참조하십시오.

snoop 명령으로 패킷 전송 모니터링

snoop 명령을 사용하여 네트워크 트래픽을 모니터링할 수 있습니다. snoop 명령은 네트워크 패킷을 캡처하고 지정한 형식으로 컨텐츠를 표시합니다. 패킷은 수신 즉시 표시하거나 파일에 저장할 수 있습니다. snoop 명령이 중간 파일에 기록할 경우 추적 사용 조건에서 패킷 손실이 발생할 가능성이 거의 없습니다. snoop 명령은 이 파일을 해석하는 데 사용됩니다.

Promiscuous 모드에서 기본 인터페이스에 대한 패킷을 캡처하려면 사용자가 네트워크 관리 권한 프로파일 또는 root 역할을 사용해야 합니다. 요약 양식에서 snoop 명령은 최고 레벨 프로토콜에 해당하는 데이터만 표시합니다. 예를 들어 NFS 패킷은 NFS 정보만 표시합니다. 기본 RPC(원격 프로시저 호출), UDP, IP 및 이더넷 프레임 정보는 표시되지 않지만, 상세 정보 표시 옵션을 선택할 경우 표시될 수도 있습니다.

snoop 명령을 자주 그리고 일관되게 사용하면 정상적인 시스템 동작에 익숙해질 수 있습니다. snoop 명령에 대한 자세한 내용은 snoop(1M) 매뉴얼 페이지를 참조하십시오.

이 절은 다음 항목으로 구성됩니다.

IPv6 지원을 위한 snoop 명령 수정 사항

snoop 명령은 IPv4 및 IPv6 패킷을 모두 캡처할 수 있습니다. 이 명령은 IPv6 헤더, IPv6 확장 헤더, ICMPv6 헤더 및 ND(Neighbor Discovery) 프로토콜 데이터를 표시할 수 있습니다. 기본적으로 snoop 명령은 IPv4 및 IPv6 패킷을 모두 표시합니다. ip 또는 ip6 프로토콜 키워드를 지정하면 IPv4 또는 IPv6 패킷만 표시됩니다. IPv6 필터 옵션을 사용하여 IPv6 패킷만 표시하도록 모든 패킷(IPv4 및 IPv6)을 필터링할 수 있습니다. “IPv6 네트워크 트래픽 모니터링” [38] 및 snoop(1M) 매뉴얼 페이지를 참조하십시오.

▼ 모든 인터페이스의 패킷을 확인하는 방법

1. **시스템에 연결된 인터페이스에 대한 정보를 출력합니다.**

```
# ipadm show-if
```

snoop 명령은 일반적으로 첫번째 비루프백 장치(보통 기본 네트워크 인터페이스)를 사용합니다.

2. **인수 없이 snoop를 입력하여 패킷 캡처를 시작합니다.**

```
# snoop
```

3. **Ctrl-C를 눌러 프로세스를 정지합니다.**

예 1-9 기본 snoop 출력 표시

다음 예는 듀얼 스택 호스트에 대한 기본 snoop 명령 출력을 보여줍니다.

```
# snoop
Using device /dev/net (promiscuous mode)
router5.local.com -> router5.local.com ARP R 10.0.0.13, router5.local.com is
0:10:7b:31:37:80
router5.local.com -> BROADCAST     TFTP Read "network-confg" (octet)
myhost -> DNSserver.local.com      DNS C 192.168.10.10.in-addr.arpa. Internet PTR ?
DNSserver.local.com  myhost        DNS R 192.168.10.10.in-addr.arpa. Internet PTR
niserve2.
.
.
.
fe80::a00:20ff:febb:e09 -> ff02::9 RIPng R (5 destinations)
```

위 출력에서 캡처된 패킷은 DNS 쿼리와 응답 및 로컬 라우터에서 보내는 주기적 ARP(Address Resolution Protocol) 패킷과 in.ripngd 데몬에 대한 IPv6 링크 로컬 주소의 알림을 보여줍니다.

▼ IPMP 그룹의 패킷을 확인하는 방법

Oracle Solaris 10에서 IPMP 그룹의 패킷을 캡처하려는 경우 IPMP 그룹의 각 인터페이스에서 snoop 명령을 수동으로 실행해야 합니다. Oracle Solaris 11 릴리스에서는 snoop 명령을 -I 옵션과 함께 사용하여 IPMP 그룹의 모든 인터페이스에서 패킷을 캡처할 수 있습니다. 출력은 하나의 출력 스트림으로 통합됩니다.

snoop 명령은 일반적으로 첫번째 비루프백 장치(보통 기본 네트워크 인터페이스)를 사용합니다. 그러나 -I 옵션을 사용할 경우 snoop 명령은 ipadm show-if 명령으로 표시되는 것과 같이, IP 인터페이스(/dev/ipnet/*에서 제공함)를 사용합니다. 이 옵션을 사용하여 루프백 트래픽의 스누핑 및 다른 IP 전용 구문을 사용으로 설정할 수도 있습니다. -d 옵션은 데이터링크 인터페이스를 사용하는데, 이 인터페이스는 dladm show-link 명령의 출력에 표시됩니다.

1. **시스템에 연결된 인터페이스에 대한 정보를 출력합니다.**

```
# ipadm show-if
```

2. **지정된 IPMP 그룹에 대한 패킷 캡처를 시작합니다.**

   # **snoop -I** *ipmp-group*

3. **Ctrl-C를 눌러 프로세스를 정지합니다.**

▼ snoop 출력을 파일에 캡처하는 방법

1. **snoop 세션을 파일로 캡처합니다. 예를 들면 다음과 같습니다.**

```
# snoop -o /tmp/cap
Using device /dev/eri (promiscuous mode)
30 snoop: 30 packets captured
```

   위 예에서는 패킷 30이 /tmp/cap 파일에 캡처되었습니다. 이 파일은 디스크 공간이 충분한 모든 디렉토리에 있을 수 있습니다. 캡처된 패킷 수는 명령줄에 표시되는데, Ctrl-C를 누르면 언제든지 중단할 수 있습니다.
   snoop 명령을 사용할 경우 호스트 시스템에서 많은 네트워크 로드가 발생하는데, 이로 인해 결과가 왜곡될 수 있습니다. 실제 결과를 표시하려면 세번째 시스템에서 snoop를 실행하십시오.

2. **snoop 출력 캡처 파일을 검사합니다.**

   # **snoop -i** *filename*

예 1-10 snoop 출력 캡처 표시

다음 출력은 snoop -i 명령의 출력으로 수신할 수 있는 캡처를 보여줍니다.

```
# snoop -i /tmp/cap
1   0.00000 fe80::a00:20ff:fee9:2d27 -> fe80::a00:20ff:fecd:4375
ICMPv6 Neighbor advertisement
...
10  0.91493    10.0.0.40 -> (broadcast)  ARP C Who is 10.0.0.40, 10.0.0.40 ?
34  0.43690 nearserver.here.com  -> 224.0.1.1  IP  D=224.0.1.1 S=10.0.0.40 LEN=28,
ID=47453, TO =0x0, TTL=1
35  0.00034  10.0.0.40 -> 224.0.1.1    IP  D=224.0.1.1 S=10.0.0.40 LEN=28, ID=57376,
TOS=0x0, TTL=47
```

▼ IPv4 서버와 클라이언트 간 패킷을 확인하는 방법

1. **클라이언트 또는 서버에 연결된 허브(또는 스위치의 미러링된 포트)와 떨어져 snoop 시스템을 설정합니다.**

세번째 시스템(snoop 시스템)은 방해하는 모든 트래픽을 확인하므로 snoop 추적은 회선에서 실제로 발생한 사항을 반영합니다.

2. **적합한 옵션과 함께 snoop 명령을 입력한 다음 출력을 파일에 저장합니다.**

3. **출력 내용을 검사하고 해석합니다.**
   snoop (http://www.rfc-editor.org/rfc/rfc1761.txt) 캡처 파일에 대한 자세한 내용은 RFC 1761, Snoop Version 2 Packet Capture File Format을 참조하십시오.

IPv6 네트워크 트래픽 모니터링

다음과 같이 snoop 명령을 사용하여 IPv6 패킷을 캡처할 수 있습니다.

```
# snoop ip6
```

다음 예는 노드에서 snoop ip6 명령을 실행할 때 표시될 수 있는 일반 출력을 보여줍니다.

```
# snoop ip6
fe80::a00:20ff:fecd:4374 -> ff02::1:ffe9:2d27 ICMPv6 Neighbor solicitation
fe80::a00:20ff:fee9:2d27 -> fe80::a00:20ff:fecd:4375 ICMPv6 Neighbor
solicitation
fe80::a00:20ff:fee9:2d27 -> fe80::a00:20ff:fecd:4375 ICMPv6 Neighbor
solicitation
fe80::a00:20ff:febb:e09 -> ff02::9      RIPng R (11 destinations)
fe80::a00:20ff:fee9:2d27 -> ff02::1:ffcd:4375 ICMPv6 Neighbor solicitation
```

snoop 명령에 대한 자세한 내용은 snoop(1M) 매뉴얼 페이지를 참조하십시오.

IP 계층 장치를 사용하여 패킷 모니터링

IP 계층 장치는 IP 관찰을 향상하기 위해 Oracle Solaris에서 도입되었습니다. 이 장치는 시스템의 네트워크 인터페이스와 연관된 주소를 사용하는 모든 패킷에 액세스할 수 있습니다. 이 주소에는 비루프백 인터페이스 또는 논리적 인터페이스에서 호스트된 주소 및 로컬 주소가 포함됩니다. IPv4 주소와 IPv6 주소 둘 다의 트래픽을 관찰할 수 있습니다. 따라서 시스템을 대상으로 하는 모든 트래픽을 모니터링할 수 있습니다. 트래픽은 루프백 IP 트래픽, 원격 시스템에서 보내는 패킷, 시스템에서 전송 중인 패킷 또는 전송된 모든 트래픽일 수 있습니다.

IP 계층 장치를 사용하면 Oracle Solaris 전역 영역 관리자가 영역 간 트래픽과 영역 내 트래픽을 모니터링할 수 있습니다. 비전역 영역의 관리자도 해당 영역에서 전송하고 수신한 트래픽을 관찰할 수 있습니다.

IP 계층에 대한 트래픽을 모니터링하려면 snoop 명령을 최신 -I 옵션과 함께 사용하십시오. 이 옵션은 명령이 기본 링크 계층 장치 대신 새 IP 계층 장치를 사용하여 트래픽 데이터를 표시하도록 지정합니다.

▼ IP 계층에서 패킷을 확인하는 방법

1. **(옵션) 필요한 경우 시스템에 연결된 인터페이스에 대한 정보를 출력합니다.**

   ```
   # ipadm show-if
   ```

2. **특정 인터페이스에서 IP 트래픽을 캡처합니다.**

   ```
   # snoop -I interface [-V | -v]
   ```

패킷 확인 방법

다음 예는 모두 이 시스템 구성을 기반으로 합니다.

```
# ipadm show-addr

ADDROBJ           TYPE     STATE        ADDR
lo0/v4            static   ok           127.0.0.1/8
net0/v4           dhcp     ok           10.153.123.225/24
lo0/v6            static   ok           ::1/128
net0/v6           addrconf ok           fe80::214:4fff:2731:b1a9/10
net0/v6           addrconf ok           2001:0db8:212:60bb:214:4fff:2731:b1a9/64
net0/v6           addrconf ok           2001:0db8:56::214:4fff:2731:b1a9/64
```

sandbox 및 toybox라는 두 영역이 다음 IP 주소를 사용한다고 가정합니다.

- sandbox - 172.0.0.3
- toybox - 172.0.0.1

시스템에 있는 서로 다른 인터페이스에서 snoop -I 명령을 사용할 수 있습니다. 표시되는 패킷 정보는 사용자가 전역 영역 관리자인지 아니면 비전역 영역의 관리자인지 여부에 따라 달라집니다.

예 1-11 루프백 인터페이스에 대한 트래픽 관찰

다음 예는 루프백 인터페이스에 대한 snoop 명령의 출력을 보여줍니다.

```
# snoop -I lo0
Using device ipnet/lo0 (promiscuous mode)
localhost -> localhost    ICMP Echo request (ID: 5550 Sequence number: 0)
localhost -> localhost    ICMP Echo reply (ID: 5550 Sequence number: 0)
```

상세 정보 출력을 생성하려면 -v 옵션을 사용하십시오.

```
# snoop -v -I lo0
Using device ipnet/lo0 (promiscuous mode)
IPNET:  ----- IPNET Header -----
IPNET:
IPNET:  Packet 1 arrived at 10:40:33.68506
IPNET:  Packet size = 108 bytes
```

```
IPNET:  dli_version = 1
IPNET:  dli_type = 4
IPNET:  dli_srczone = 0
IPNET:  dli_dstzone = 0
IPNET:
IP:   ----- IP Header -----
IP:
IP:   Version = 4
IP:   Header length = 20 bytes
...
```

IP 계층에서는 패킷 관찰이 지원되므로 새 `ipnet` 헤더가 관찰 중인 패킷의 앞에 표시됩니다. 소스 및 대상 ID가 모두 표시됩니다. ID `0`은 트래픽이 전역 영역에서 생성됨을 나타냅니다.

예 1-12 로컬 영역 내의 `net0` 장치에서 패킷 플로우 관찰

다음 예는 시스템 내의 다른 영역에서 발생하는 트래픽을 보여줍니다. 로컬에서 다른 영역으로 전달되는 패킷을 비롯하여 `net0` IP 주소와 연관된 모든 패킷을 확인할 수 있습니다. 상세 정보 출력을 생성하면 패킷 플로우와 관련된 영역을 확인할 수도 있습니다.

```
# snoop -I net0
Using device ipnet/net0 (promiscuous mode)
toybox -> sandbox TCP D=22 S=62117 Syn Seq=195630514 Len=0 Win=49152 Options=<mss
sandbox -> toybox TCP D=62117 S=22 Syn Ack=195630515 Seq=195794440 Len=0 Win=49152
toybox -> sandbox TCP D=22 S=62117 Ack=195794441 Seq=195630515 Len=0 Win=49152
sandbox -> toybox TCP D=62117 S=22 Push Ack=195630515 Seq=195794441 Len=20 Win=491

# snoop -I net0 -v port 22
IPNET:  ----- IPNET Header -----
IPNET:
IPNET:  Packet 5 arrived at 15:16:50.85262
IPNET:  Packet size = 64 bytes
IPNET:  dli_version = 1
IPNET:  dli_type = 0
IPNET:  dli_srczone = 0
IPNET:  dli_dstzone = 1
IPNET:
IP:   ----- IP Header -----
IP:
IP:   Version = 4
IP:   Header length = 20 bytes
IP:   Type of service = 0x00
IP:         xxx. .... = 0 (precedence)
IP:         ...0 .... = normal delay
IP:         .... 0... = normal throughput
IP:         .... .0.. = normal reliability
IP:         .... ..0. = not ECN capable transport
IP:         .... ...0 = no ECN congestion experienced
IP:   Total length = 40 bytes
IP:   Identification = 22629
IP:   Flags = 0x4
IP:         .1.. .... = do not fragment
IP:         ..0. .... = last fragment
```

```
IP:   Fragment offset = 0 bytes
IP:   Time to live = 64 seconds/hops
IP:   Protocol = 6 (TCP)
IP:   Header checksum = 0000
IP:   Source address = 172.0.0.1, 172.0.0.1
IP:   Destination address = 172.0.0.3, 172.0.0.3
IP:   No options
IP:
TCP:  ----- TCP Header -----
TCP:
TCP:  Source port = 46919
TCP:  Destination port = 22
TCP:  Sequence number = 3295338550
TCP:  Acknowledgement number = 3295417957
TCP:  Data offset = 20 bytes
TCP:  Flags = 0x10
TCP:        0... .... = No ECN congestion window reduced
TCP:        .0.. .... = No ECN echo
TCP:        ..0. .... = No urgent pointer
TCP:        ...1 .... = Acknowledgement
TCP         .... 0... = No push
TCP         .... .0.. = No reset
TCP:        .... ..0. = No Syn
TCP:        .... ...0 = No Fin
TCP:  Window = 49152
TCP:  Checksum = 0x0014
TCP:  Urgent pointer = 0
TCP:  No options
TCP:
```

위 출력에서 ipnet 헤더는 패킷이 전역 영역(ID 0)에서 sandbox(ID 1)로 제공됨을 나타냅니다.

예 1-13 영역 식별을 통한 네트워크 트래픽 관찰

다음 예는 영역 식별을 통해 네트워크 트래픽을 관찰하는 방법을 보여줍니다. 이 방법은 영역이 여러 개 있는 시스템에서 매우 유용합니다. 현재는 영역 ID로만 영역을 식별할 수 있습니다. 영역 이름과 함께 snoop 명령을 사용하는 것은 지원되지 않습니다.

```
# snoop -I hme0 sandboxsnoop -I net0 sandbox
Using device ipnet/hme0 (promiscuous mode)
toybox -> sandbox TCP D=22 S=61658 Syn Seq=374055417 Len=0 Win=49152 Options=<mss
sandbox -> toybox TCP D=61658 S=22 Syn Ack=374055418 Seq=374124525 Len=0 Win=49152
toybox -> sandbox TCP D=22 S=61658 Ack=374124526 Seq=374055418 Len=0 Win=49152
```

ipstat 및 tcpstat 명령으로 네트워크 트래픽 관찰

이 릴리스에서는 서버에서 다양한 유형의 네트워크 트래픽을 관찰하는 두 가지 새 명령인 ipstat 및 tcpstat가 도입되었습니다.

ipstat 명령은 선택한 출력 모드 및 명령 구문에 지정된 정렬 순서를 기준으로 서버의 IP 트래픽에 대한 통계를 수집하고 보고하는 데 사용됩니다. 따라서 소스, 대상, 상위 계층 프로토콜 및 인터페이스에서 통합된 IP 계층의 네트워크 트래픽을 관찰할 수 있습니다. 한 서버와 다른 서버 간 트래픽 양을 관찰하려는 경우 이 명령을 사용하십시오.

tcpstat 명령은 선택한 출력 모드 및 명령 구문에 지정된 정렬 순서를 기준으로 서버의 TCP 및 UDP 트래픽에 대한 통계를 수집하고 보고하는 데 사용됩니다. 따라서 전송 계층에서 특히 TCP 및 UDP에 대한 네트워크 트래픽을 관찰할 수 있습니다. 소스 및 대상 IP 주소 이외에도, 소스 및 대상 TCP 또는 UDP 포트, 트래픽을 송수신하는 프로세스의 PID, 프로세스가 실행 중인 영역의 이름을 관찰할 수 있습니다.

다음과 같은 몇 가지 방법으로 tcpstat 명령을 사용할 수 있습니다.

- 서버에서 발생하는 TCP 및 UDP 트래픽의 최대 소스를 식별합니다.
- 특정 프로세스에 의해 발생하고 있는 트래픽을 검사합니다.
- 특정 영역에서 발생하고 있는 트래픽을 검사합니다.
- 로컬 포트에 바인드될 프로세스를 결정합니다.

참고 - 위 목록은 전체 목록이 아닙니다. tcpstat 명령을 사용할 수 있는 다른 몇 가지 방법이 더 있습니다. 자세한 내용은 tcpstat(1M) 매뉴얼 페이지를 참조하십시오.

ipstat 및 tcpstat 명령을 사용하려면 다음 권한 중 하나가 필요합니다.

- root 역할을 맡습니다.
- dtrace_kernel 권한을 명시적으로 지정합니다.
- 네트워크 관리 또는 네트워크 관찰 권한 프로파일을 지정합니다.

다음 예는 이 두 가지 명령으로 네트워크 트래픽을 관찰하는 데 사용할 수 있는 몇 가지 방법을 보여줍니다. 자세한 내용은 tcpstat(1M) 및 ipstat(1M) 매뉴얼 페이지를 참조하십시오.

다음 예는 -c 옵션과 함께 실행된 경우 ipstat 명령의 출력을 보여줍니다. -c 옵션을 사용하면 이전 보고서를 덮어쓰지 않고 이전 보고서 뒤에 새 보고서를 인쇄합니다. 이 예에서 숫자 3은 데이터 표시 간격을 지정합니다. 이는 명령을 ipstat 3으로 호출하는 경우와 같습니다.

```
# ipstat -c 3
SOURCE                    DEST                      PROTO    INT       BYTES
zucchini                  antares                   TCP      net0      72.0
zucchini                  antares                   SCTP     net0      64.0
antares                   zucchini                  SCTP     net0      56.0
amadeus.foo.example.com   10.6.54.255               UDP      net0      40.0
antares                   zucchini                  TCP      net0      40.0
zucchini                  antares                   UDP      net0      16.0
antares                   zucchini                  UDP      net0      16.0
Total: bytes in: 192.0  bytes out: 112.0
```

이에 비해, 다음 예는 -c 옵션과 함께 사용된 경우 tcpstat 명령의 출력을 보여줍니다.

```
# tcpstat -c 3
ZONE         PID PROTO  SADDR             SPORT DADDR             DPORT   BYTES
global    100680 UDP    antares           62763 agamemnon          1023   76.0
global    100680 UDP    antares             775 agamemnon          1023   38.0
global    100680 UDP    antares             776 agamemnon          1023   37.0
global    100680 UDP    agamemnon          1023 antares           62763   26.0
global    104289 UDP    zucchini          48655 antares            6767   16.0
global    104289 UDP    clytemnestra      51823 antares            6767   16.0
global    104289 UDP    antares            6767 zucchini         48655   16.0
global    104289 UDP    antares            6767 clytemnestra     51823   16.0
global    100680 UDP    agamemnon          1023 antares             776   13.0
global    100680 UDP    agamemnon          1023 antares             775   13.0
global    104288 TCP    zucchini          33547 antares            6868    8.0
global    104288 TCP    clytemnestra      49601 antares            6868    8.0
global    104288 TCP    antares            6868 zucchini         33547    8.0
global    104288 TCP    antares            6868 clytemnestra     49601    8.0
Total: bytes in: 101.0  bytes out: 200.0
```

다음 추가 예는 ipstat 및 tcpstat 명령을 사용하여 네트워크 트래픽을 관찰할 수 있는 다른 방법을 보여줍니다.

예 1-14 ipstat 명령을 사용하여 가장 활동적인 IP 트래픽 플로우 관찰

다음 예는 가장 활동적인 5개의 IP 트래픽 플로우를 보고합니다. -l *nlines* 옵션은 보고서당 출력할 데이터 라인 수를 지정합니다.

```
# ipstat -l 5
SOURCE                    DEST                      PROTO    IFNAME     BYTES
charybdis.foo.example.com  achilles.exampl           UDP      net0        6.6K
eratosthenes.example.com   aeneas.example.c          TCP      tun0        6.1K
achilles.exampl            charybdis.foo.example.com  UDP      net0       964.0
aeneas.example.c           eratosthenes.example.com   TCP      tun0       563.0
odysseus.example.          255.255.255.255           UDP      net0        66.0
Total: bytes in: 12.6K bytes out:  2.2K
```

예 1-15 ipstat 명령을 사용하여 시간 기록 표시

다음 예는 표준 날짜 형식(-d d)의 시간 기록과 함께 최상위 IP 트래픽을 보고합니다. 시간 기록을 초 또는 Unix 시간(-d u)으로 인쇄하도록 지정할 수 있습니다. 간격은 10초로 설정됩니다.

```
# ipstat -d d -c 10
Monday, March 26, 2012 08:34:07 PM EDT
SOURCE                    DEST                      PROTO    IFNAME     BYTES
charybdis.foo.example.com  achilles.exampl           UDP      net0       15.1K
eratosthenes.example.com   aeneas.example.c          TCP      tun0       13.9K
achilles.exampl            charybdis.foo.example.com  UDP      net0        2.4K
aeneas.example.c           eratosthenes.example.com   TCP      tun0        1.5K
odysseus.example.          255.255.255.255           UDP      net0       66.0
cassiopeia.foo.example.com aeneas.example.c          TCP      tun0       29.0
aeneas.example.c           cassiopeia.foo.example.com TCP      tun0       20.0
Total: bytes in: 29.1K bytes out:  3.8K
```

예 1-16 tcpstat 명령을 사용하여 가장 활동적인 트래픽 플로우 관찰

다음 예는 가장 활동적인 5개의 서버 TCP 트래픽 플로우를 보고합니다.

```
# tcpstat -l 5
ZONE            PID PROTO  SADDR            SPORT DADDR            DPORT   BYTES
global        28919 TCP    achilles.exampl  65398 aristotle.exampl   443   33.0
zone1          6940 TCP    ajax.example.com  6868 achilles.exampl  61318    8.0
zone1          6940 TCP    achilles.exampl  61318 ajax.example.com  6868    8.0
global         8350 TCP    ajax.example.com  6868 achilles.exampl  61318    8.0
global         8350 TCP    achilles.exampl  61318 ajax.example.com  6868    8.0
Total: bytes in: 16.0  bytes out: 49.0
```

예 1-17 tcpstat 명령을 사용하여 시간 기록 정보 표시

다음 예에서 tcpstat 명령은 서버 TCP 네트워크 트래픽에 대한 시간 기록 정보를 표준 날짜 형식으로 표시하는 데 사용됩니다.

```
# tcpstat -d d -c 10
Saturday, March 31, 2012 07:48:05 AM EDT
ZONE            PID PROTO  SADDR            SPORT DADDR            DPORT   BYTES
global         2372 TCP    penelope.example 58094 polyphemus.examp    80   37.0
zone1          6940 TCP    ajax.example.com  6868 achilles.exampl  61318    8.0
zone1          6940 TCP    achilles.exampl  61318 ajax.example.com  6868    8.0
global         8350 TCP    ajax.example.com  6868 achilles.exampl  61318    8.0
global         8350 TCP    achilles.exampl  61318 ajax.example.com  6868    8.0
Total: bytes in: 16.0  bytes out: 53.0
```

♦ ♦ ♦ 2 장

2 IPMP 관리 정보

IPMP(IP Network Multipathing)는 여러 IP 인터페이스를 하나의 논리적 인터페이스로 그룹화할 수 있는 계층 3(L3) 기술입니다. 실패 감지, 투명한 액세스 페일오버, 패킷 부하 분산과 같은 기능을 갖추고 있으므로 IPMP는 시스템에서 네트워크가 항상 사용 가능하도록 하여 네트워크 성능을 향상시킵니다.

이 장의 내용:

참고 - 이 장과 3장. IPMP 관리 전체에서 특히 인터페이스라는 용어는 *IP* 인터페이스를 의미합니다. NIC(네트워크 인터페이스 카드)와 같이 용어 정의에서 용어의 다른 사용을 명시적으로 나타내지 않는 경우 이 용어는 항상 IP 계층에 구성된 인터페이스를 가리킵니다.

IPMP의 새로운 기능

이 릴리스에서 새로 추가되었거나 변경된 기능은 다음과 같습니다.

IPMP 구성 변경 사항

이 릴리스에서 IPMP는 Oracle Solaris 10에서와 다르게 작동합니다. 한 가지 중요한 변경 사항은 IP 인터페이스가 이제 가상 IP 인터페이스(예: `ipmp0`)로 그룹화된다는 점입니다. 가상 IP 인터페이스는 모든 데이터 IP 주소를 지원하지만 프로브 기반 오류 검색을 위해 사용되는 테스트 주소는 `net0`과 같은 기본 인터페이스에 지정됩니다. 자세한 내용은 "IPMP 작동 방식" [51]을 참조하십시오.

기존 IPMP 구성에서 새로운 IPMP 모델로 전환할 때는 다음과 같은 일반적인 워크플로우를 참조하십시오.

1. IPMP를 구성하기 전에 `DefaultFixed` 프로파일이 시스템에서 사용으로 설정되었는지 확인합니다. "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "의 "프로파일 사용 및 사용 안함으로 설정"을 참조하십시오.
2. SPARC 기반 시스템에서 MAC 주소가 고유한지 확인합니다. "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "의 "각 인터페이스의 MAC 주소가 고유한지 확인하는 방법"을 참조하십시오.
3. `dladm` 명령을 사용하여 데이터 링크를 구성합니다. IPMP 구성 내에서 동일한 물리적 네트워크 장치를 사용하려면 먼저 각 장치 인스턴스와 연관된 데이터 링크를 식별해야 합니다.

```
# dladm show-phys
LINK              MEDIA                STATE      SPEED  DUPLEX    DEVICE
net1              Ethernet             unknown    0      unknown   bge1
net0              Ethernet             up         1000   full      bge0
net2              Ethernet             unknown    1000   full      e1000g0
net3              Ethernet             unknown    1000   full      e1000g1
```

4. 이전에 IPMP 구성에 `e1000g0` 및 `e1000g1`을 사용한 경우 이제 `net2` 및 `net3`을 사용합니다. 데이터 링크는 물리적 링크는 물론 집계, VLAN, VNIC 등을 기반으로 할 수 있습니다. 자세한 내용은 "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "의 "시스템의 데이터 링크 표시"를 참조하십시오.
5. `ipadm` 명령을 사용하여 다음 작업을 수행합니다.
   - 네트워크 계층 구성
   - IP 인터페이스 만들기
   - IPMP 그룹에 IP 인터페이스 추가
   - IPMP 그룹에 데이터 주소 추가

이 릴리스의 네트워크 관리 명령 변경 사항에 대한 자세한 내용은 "Oracle Solaris 10에서 Oracle Solaris 11.2로 전환 "의 "네트워크 관리 명령 변경 사항"을 참조하십시오.

Oracle Solaris의 IPMP 지원

IPMP에서는 다음과 같은 지원을 제공합니다.

- IPMP를 사용하면 여러 IP 인터페이스를 IPMP 그룹이라는 하나의 그룹으로 구성할 수 있습니다. 여러 기본 IP 인터페이스가 포함된 IPMP 그룹은 전체적으로 하나의 *IPMP* 인터페이스로 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 인터페이스 간 데이터 주소 분배를 처리합니다. IPMP 그룹이 만들어지면 데이터 주소가 주소 풀로 IPMP 인터페이스에 속합니다. 그런 다음 커널이 데이터 주소를 그룹의 기본 활성 인터페이스에 임의로 자동 바인딩합니다.

- 주로 `ipmpstat` 명령을 사용하여 IPMP 그룹에 대한 정보를 가져옵니다. 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` 명령 기능, 사용할 수 있는 옵션 및 각 옵션이 생성하는 출력 결과는 모두 "IPMP 정보 모니터링" [81]에 설명되어 있습니다.
- IPMP 그룹을 식별하기 쉽도록 IPMP 인터페이스에 사용자 정의 이름을 지정할 수 있습니다. "IPMP 그룹 구성" [63]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* 그룹으로 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 분산이라고 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.

IPMP가 인바운드 및 아웃바운드 부하 분산에 적용하는 정책과 관련하여 다음과 같은 중요한 정보에 유념하십시오.

- 온 링크 IP 주소의 경우, 해당 IP 주소에 연결할 하나의 활성 IP 인터페이스를 IPMP에서 무작위로 선택합니다. 지정된 온 링크 IP 주소에 대한 고유 연결이 여러 개 있는 경우 해당 연결이 모두 동일한 아웃바운드 IP 인터페이스를 사용합니다. 또한 IP 인터페이스가 시간 경과에 따라 변경될 경우 변경 사항이 해당 IP 주소에 대한 모든 연결에 적용됩니다.
- 오프 링크 IP 주소의 경우, 오프 링크 IP 주소에 연결될 때 사용할 온 링크 IP 라우터의 IP 주소에 연결할 하나의 IP 인터페이스를 IPMP에서 무작위로 선택합니다. 이 정책은 지정된 IPMP 그룹의 모든 오프 링크 IP 주소가 하나의 IP 인터페이스를 사용함을 의미합니다.

참고 - 현재 인바운드 및 아웃바운드 부하 분산 정책은 변경될 수 있습니다.

링크 통합은 IPMP와 유사한 기능을 수행하여 네트워크 성능 및 가용성을 향상시킵니다. 이 두 기술을 비교하려면 "Oracle Solaris 11.2의 네트워크 데이터 링크 관리 "의 부록 A, "링크 집계 및 IPMP: 기능 비교"를 참조하십시오.

IPMP 사용 규칙

IPMP 그룹 구성은 특정 시스템 구성에 의해 결정됩니다.

IPMP에 대해 다음 규칙을 따르십시오.

1. 동일한 LAN에 있는 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.

참고 - 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. Oracle에서 제공하는 특정 엔지니어링된 시스템의 경우 이 규칙에 대해 몇 가지 예외 사항이 적용됩니다. 자세한 분류는 Oracle 지원 담당자에게 문의하십시오.

---

2. 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 "IPMP 인터페이스 구성 유형" [50]에 자세히 설명되어 있습니다.

   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.
3. 동일한 그룹의 모든 인터페이스에 동일한 STREAMS 모듈이 동일한 순서로 구성되어 있어야 합니다. IPMP 그룹을 계획할 때 먼저 잠재 IPMP 그룹의 모든 인터페이스에 있는 STREAMS 모듈의 순서를 확인한 다음 각 인터페이스의 모듈을 IPMP 그룹에 대한 표준 순서로 푸시합니다. STREAMS 모듈 목록을 인쇄하려면 `ifconfig` *interface* `modlist` 명령을 사용합니다. 예를 들어, `net0` 인터페이스의 `ifconfig` 출력은 다음과 같습니다.

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

   위 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.

다음 예는 각 인터페이스의 모듈을 IPMP 그룹에 대한 표준 순서로 푸시하는 데 사용할 수 있는 명령을 보여줍니다.

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

IPMP 그룹 계획에 대한 단계별 지침은 IPMP 그룹을 계획하는 방법 [64]을 참조하십시오.

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 모니터링하도록 데몬을 구성할 수도 있습니다. 실패 감지에 대한 설명은 "IPMP의 실패 감지" [57]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.mpathd`는 관리자가 구성한 정책에 따라 필요할 경우 기본 인터페이스를 활성화하거나 비활성화합니다. `in.mpathd` 데몬이 기본 인터페이스의 활성화를 관리하는 방식에 대한 자세한 내용은 "IPMP 작동 방식" [51]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 프로브할 대상 인터페이스
  - 실패를 감지하기 위해 대상을 프로브하는 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 방법은 IPMP 데몬의 동작을 구성하는 방법 [79]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 인터

페이스에 대한 다른 정보도 표시합니다. 이 명령에 대한 자세한 내용은 “IPMP 정보 모니터링” [81] 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 인터페이스는 현재 IPMP 그룹이 사용할 수 있는 IP 인터페이스입니다.

참고 - 기본적으로 IPMP 그룹에 속하도록 인터페이스를 구성하면 기본 인터페이스가 활성화됩니다.

- **활성-대기 구성** - 관리상 인터페이스가 하나 이상 대기 인터페이스로 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 필요한 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 모니터링할 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.

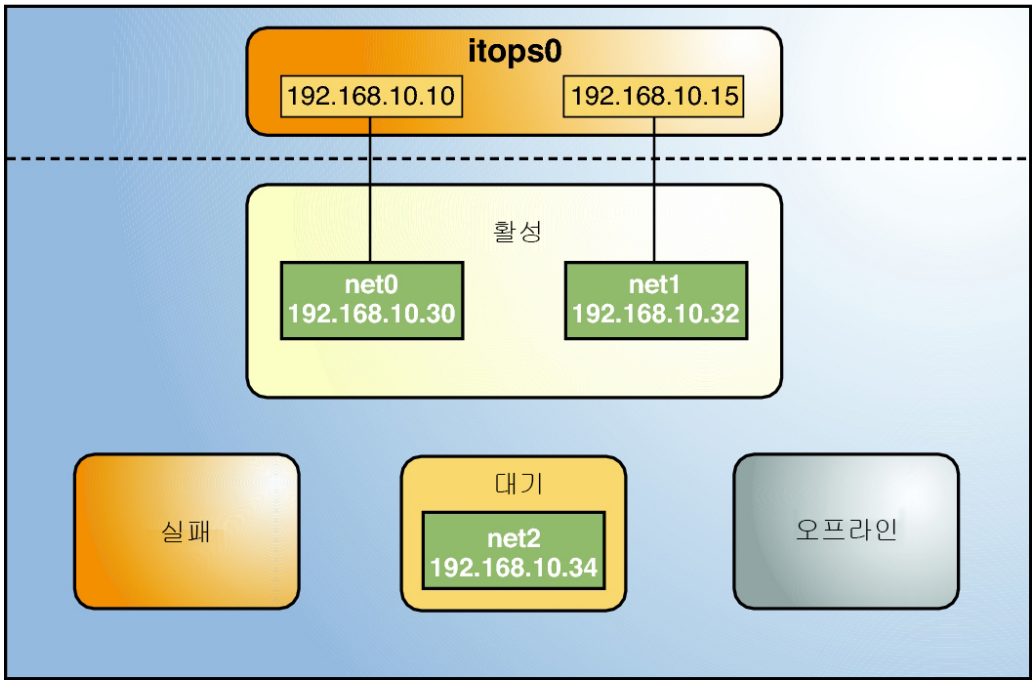
IPMP 작동 방식

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 것과 같이 활성-대기 구성을 사용하는, 인터페이스가 3개인 IPMP 그룹 `itops0`을 고려해 보십시오.

그림 2-1 IPMP 활성-대기 구성



IPMP 그룹 `itops0`은 다음과 같이 구성됩니다.

- `192.168.10.10`과 `192.168.10.15`라는 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

참고 - 그림 2-1. “IPMP 활성-대기 구성”, 그림 2-2. “IPMP의 인터페이스 실패”, 그림 2-3. “IPMP의 대기 인터페이스 실패” 및 그림 2-4. “IPMP 복구 프로세스”의 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 “IPMP 정보 모니터링” [81]을 참조하십시오.

다음 명령은 그림 2-1. “IPMP 활성-대기 구성”에 있는 IPMP 구성에 대한 정보를 표시합니다.

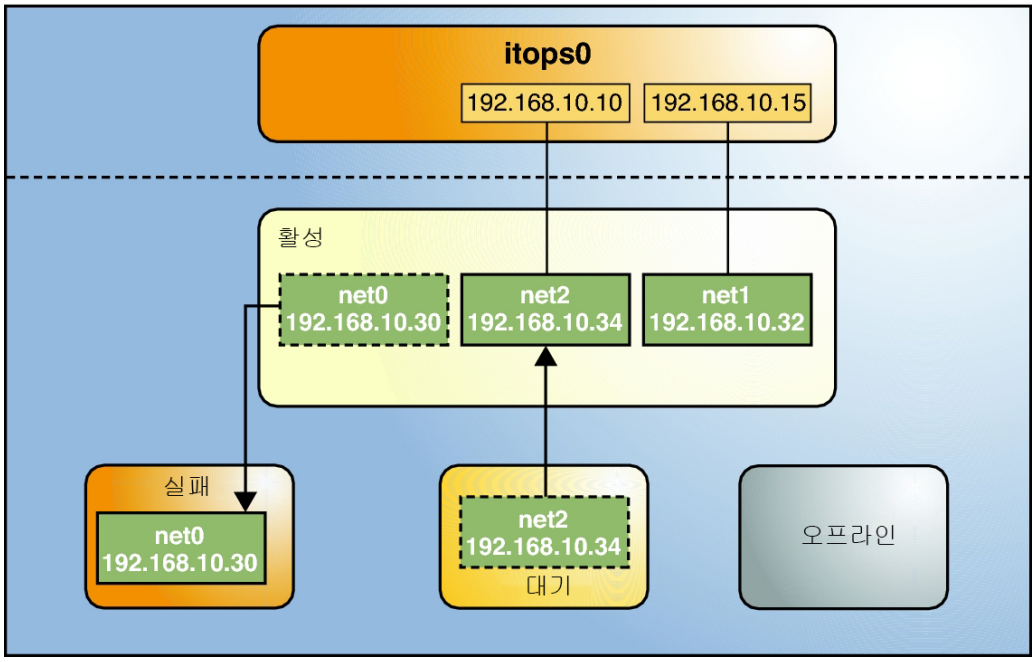
```
# ipmpstat -g
GROUP     GROUPNAME     STATE     FDT       INTERFACES
itops0    itops0        ok        10.00s    net1 net0 (net2)
```

다음과 같이 그룹의 기본 인터페이스에 대한 정보를 표시합니다.

```
# 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 2-2 IPMP의 인터페이스 실패



참고 - 그림 2-2. "IPMP의 인터페이스 실패"에 표시된 데이터 주소와 활성 인터페이스 간 일대일 매핑은 그림을 단순화하기 위한 것일 뿐입니다. IP 커널 모듈은 데이터 주소와 인터페이스 간의 일대일 관계를 준수할 필요 없이 임의로 데이터 주소를 지정할 수 있습니다.

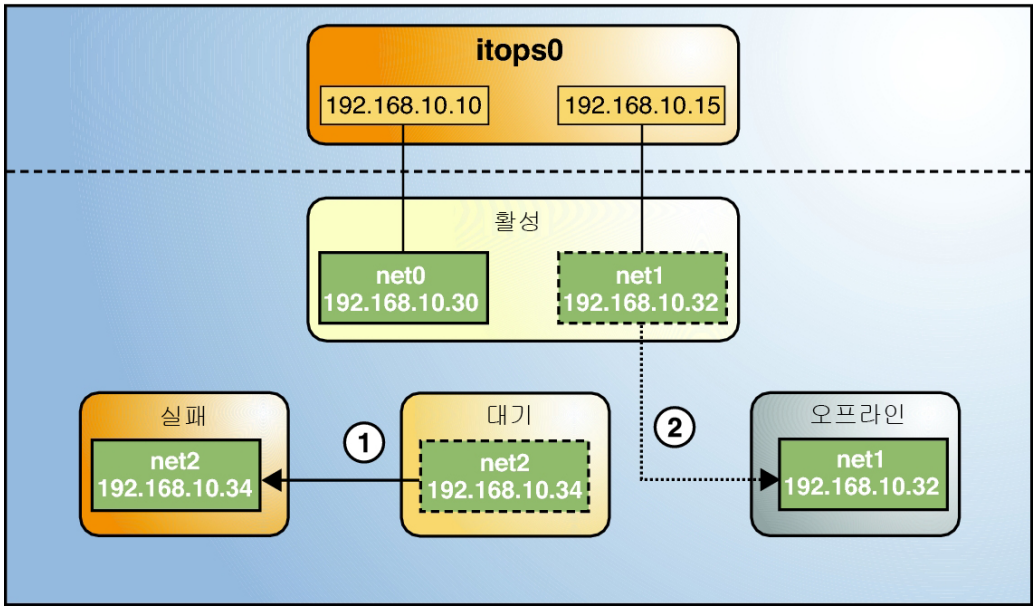
`ipmpstat` 명령은 그림의 정보를 다음과 같이 표시합니다.

```
# 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
```

`net0`은 복구 후 활성 인터페이스 상태로 돌아갑니다. `net2`는 원래의 대기 상태로 돌아갑니다.

다른 실패 시나리오는 그림 2-3. "IPMP의 대기 인터페이스 실패"에 나와 있으며, 여기에서는 대기 인터페이스 `net2`가 실패합니다(1). 나중에 활성 인터페이스, `net1`이 관리자에 의해 오프라인으로 전환됩니다(2). 그 결과, IPMP 그룹에서 작동하는 인터페이스는 `net0` 한 개뿐입니다.

그림 2-3 IPMP의 대기 인터페이스 실패



`ipmpstat` 명령은 그림의 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 이후의 복구 프로세스는 다릅니다. 복구 후의 구성과 비교하여 복구 프로세스는 IPMP 그룹의 원래 활성 인터페이스 수에 따라 다릅니다. 다음 그림은 복구 프로세스를 나타냅니다.

그림 2-4 IPMP 복구 프로세스

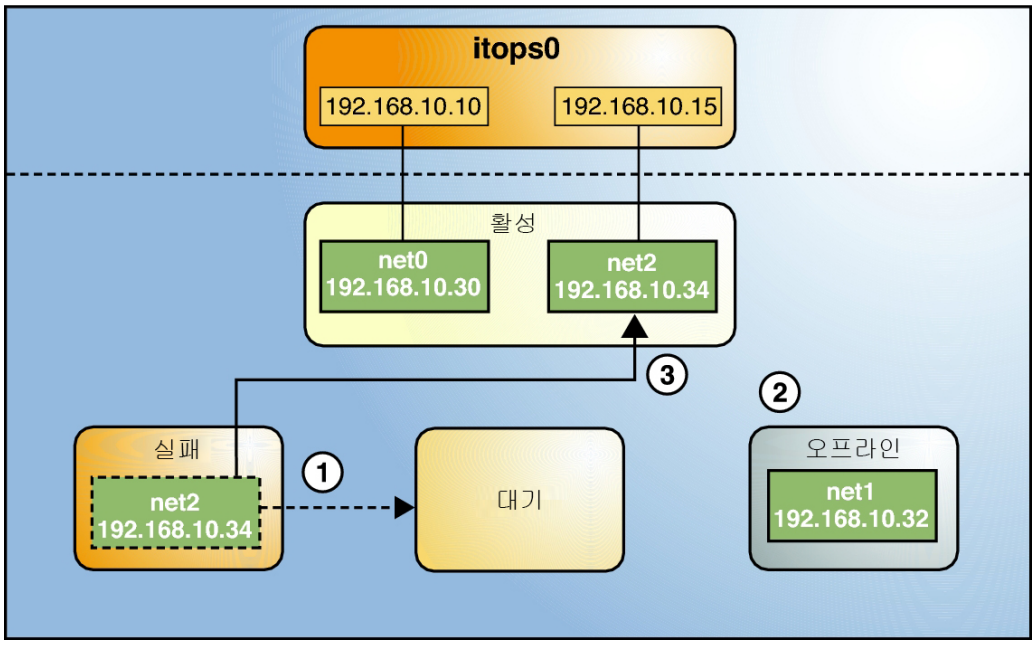


그림 2-4, “IPMP 복구 프로세스”에서는 net2가 복구되면 정상적으로 원래 대기 인터페이스 상태로 돌아갑니다(1). 하지만 net1이 계속 오프라인 상태로 유지되므로 IPMP 그룹이 원래 활성 인터페이스 두 개를 반영하지 않습니다(2). 따라서 IPMP가 net2를 활성 인터페이스로 대신 배포합니다(3).

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 2-2, “IPMP의 인터페이스 실패”의 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 “FAILBACK=no 모드” [61]를 참조하십시오.

IPMP 주소 지정

IPv4 네트워크와 이중 스택 IPv4 및 IPv6 네트워크에서 모두 IPMP 실패 감지를 구성할 수 있습니다. IPMP를 사용하여 구성된 인터페이스는 두 가지 유형의 주소(데이터 주소와 테스트 주소)를 지원합니다. IP 주소는오직 IPMP 인터페이스(그룹)에만 있으며 데이터 주소로 지정됩니다. 테스트 주소는 기본 인터페이스에 있는 IP 주소입니다.

데이터 주소

데이터 주소는 부트 시 DHCP 서버에 의해 동적으로 또는 `ipadm` 명령을 사용하여 수동으로 IP 인터페이스에 지정된 일반적인 IPv4 및 IPv6 주소입니다. 데이터 주소는 오직 IPMP 인터페이스(그룹)에만 지정됩니다. 표준 IPv4 패킷 트래픽과 IPv6 패킷 트래픽(해당하는 경우)은 데이터 트래픽으로 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.

참고 - 프로브 기반 실패 감지를 사용하려는 경우에만 테스트 주소를 구성하십시오. 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 “프로브 기반 실패 감지” [57]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.rfc-editor.org/rfc/rfc1918.txt)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 감지의 두 가지 유형:
  - 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 트래픽이라고도 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 프로브합니다. 지정된 인터페이스에 대한 5회 연속 프로브에 대해 응답이 없을 경우 `in.mpathd` 데몬은 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 IPMP 데몬의 동작을 구성하는 방법 [79]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 프로브할 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 "프로브 기반 실패 감지 구성" [76]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.

- **전이적 프로브**

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

예를 들어, 기본 인터페이스 4개와 데이터 주소 1개로 구성된 IPMP 그룹을 고려해 보십시오. 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.

참고 - Oracle Solaris에서 프로브 기반 실패 감지는 테스트 주소를 사용하여 작동됩니다. 테스트 주소가 사용되지 않는 프로브 기반 실패 감지를 선택하려면 전이적 프로브를 수동으로 사용 설정해야 합니다. 지침은 "실패 감지 방법 선택" [78]을 참조하십시오.

그룹 실패

IPMP 그룹의 모든 인터페이스가 동시에 실패하면 그룹 실패가 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 프로브합니다.

테스트 주소가 없는 IPMP 그룹에서는 활성 인터페이스를 프로브할 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 이러한 인터페이스는 익명 그룹의 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 IPMP 데몬의 동작을 구성하는 방법 [79]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 복구 중 복구 프로세스가 진행되는 방식은 다음과 같이 실패한 인터페이스의 원래 구성 방식에 따라 다릅니다.

- 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.

참고 - 예외는 복구된 활성 인터페이스가 `FAILBACK=no` 모드로 구성되어 있는 경우입니다. 자세한 내용은 “`FAILBACK=no` 모드” [61]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 작동하는지에 대한 그래픽 표현은 “IPMP 작동 방식” [51]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 중단조차도 허용되지 않을 수 있습니다. 이 경우 활성화된 대기 인터페이스가 활성 인터페이스로 계속 작동하도록 설정할 수 있습니다. IPMP를 사용하면 기본 동작을 대체하여 인터페이스가 복구 시 자동으로 활성이 되지 않게 할 수 있습니다. 해당 인터페이스는 FAILBACK=no 모드로 구성해야 합니다. 관련 절차는 IPMP 데몬의 동작을 구성하는 방법 [79]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.

참고 - FAILBACK=NO 모드는 인터페이스별 조정 가능 매개변수로 설정되는 대신 전체 IPMP 그룹에 대해 설정됩니다.

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 실패할 경우 지속 구성만 복원됩니다. 이 경우 다음 오류 메시지가 기록됩니다.

```
"IP: persistent configuration is restored for <ifname>"
```

그렇지 않으면 분리 요청이 성공적으로 완료됩니다. 시스템에서 해당 구성 요소를 제거할 수 있습니다. 이 경우 기존 연결이 중단되지 않습니다.

참고 - NIC를 교체하는 경우 교체 카드가 동일한 유형(예: 이더넷)인지 확인하십시오. NIC 교체 후에는 지속 IP 인터페이스 구성이 해당 NIC에 적용됩니다.

♦ ♦ ♦ 3 장

3 IPMP 관리

이 장에서는 Oracle Solaris 릴리스에서 IPMP로 인터페이스 그룹을 관리하는 방법에 대해 설명합니다. 이 장의 작업은 Oracle Solaris 10에서 IPMP를 구성하는 데 사용되는 `ifconfig` 명령을 대체하는 `ipadm` 명령을 사용하여 IPMP를 구성하는 방법에 대해 설명합니다. 이러한 두 명령을 서로에게 매핑하는 자세한 방법은 "Oracle Solaris 10에서 Oracle Solaris 11.2로 전환 "의 "ifconfig 명령과 ipadm 명령 비교"를 참조하십시오.

IPMP 개념 모델의 변경 사항에 대한 자세한 설명은 "IPMP의 새로운 기능" [45]을 참조하십시오.

이 장의 내용:

IPMP 그룹 구성

다음 정보는 IPMP 그룹을 계획하고 구성하는 방법에 대해 설명합니다. 2장. IPMP 관리 정보의 개요에는 IPMP 그룹을 인터페이스로 구현하는 방법이 설명되어 있습니다. 이 장에서 *IPMP* 그룹 및 *IPMP* 인터페이스는 같은 의미로 사용되는 용어입니다.

이 절은 다음 작업으로 구성됩니다.

▼ IPMP 그룹을 계획하는 방법

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

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.

참고 - 각 서브넷 또는 L2 브로드캐스트 도메인에 대해 IPMP 그룹 한 개만 구성해야 합니다. 자세한 내용은 "IPMP 사용 규칙" [48]을 참조하십시오.

---

1. **요구에 맞는 일반 IPMP 구성을 결정합니다.**
   사용할 IPMP 구성을 결정하는 지침은 이 절차의 작업 요약에 나오는 정보를 참조하십시오.

2. **(SPARC only) 그룹의 각 인터페이스에 고유한 MAC 주소가 있는지 확인합니다.**
   시스템의 각 인터페이스에 대해 고유 MAC 주소를 구성하려면 "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "의 "각 인터페이스의 MAC 주소가 고유한지 확인하는 방법"을 참조하십시오.

3. **IPMP 그룹의 모든 인터페이스에서 동일한 STREAMS 모듈 세트가 구성되고 푸시되는지 확인하십시오.**
   지침 및 사용할 명령 구문은 "IPMP 사용 규칙" [48]을 참조하십시오.

4. **IPMP 그룹의 모든 인터페이스에서 동일한 IP 주소 형식을 사용합니다.**
   IPv4에 대해 한 인터페이스가 구성된 경우 IPMP 그룹의 모든 인터페이스가 IPv4를 사용하도록 구성해야 합니다. 마찬가지로, IPv6 주소를 한 인터페이스에 추가하는 경우 IPv6를 지원하도록 IPMP 그룹의 모든 인터페이스를 구성해야 합니다.

5. **구현하려는 실패 감지 유형을 결정합니다.**
   예를 들어, 프로브 기반 실패 감지를 구현하려는 경우 기본 인터페이스에 테스트 주소를 구성해야 합니다. "IPMP의 실패 감지" [57]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.

참고 - 예를 들어, 시스템에 물리적 인터페이스가 하나뿐인 경우 단일 인터페이스 IPMP 그룹을 구성할 수도 있습니다. "IPMP 인터페이스 구성 유형" [50]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.**

   RFC 1577 (http://www.rfc-editor.org/rfc/rfc1577.txt) 및 RFC 2225 (http://www.rfc-editor.org/rfc/rfc2225.txt)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. "IPMP 인터페이스 구성 유형" [50]을 참조하십시오. 다음 절차에서는 DHCP를 사용하여 활성-대기 IPMP 그룹을 구성하는 방법에 대해 설명합니다.

시작하기 전에 다음 절차를 수행하기 전에 다음을 수행하십시오.

- 잠재 IPMP 그룹에 포함될 IP 인터페이스가 시스템의 네트워크 데이터 링크에서 올바르게 구성되었는지 확인합니다. 절차는 "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "를 참조하십시오. 기본 IP 인터페이스를 만들지 않은 경우에도 IPMP 인터페이스를 만들 수 있습니다. 그러나 기본 IP 인터페이스를 만들지 않은 경우 IPMP 인터페이스의 후속 구성이 실패합니다.
- 또한 SPARC 기반 시스템을 사용하는 경우 인터페이스마다 고유한 MAC 주소를 구성해야 합니다. "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "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 11.2의 DHCP 작업 "을 참조하십시오.

1. **`root` 역할로 전환합니다.**

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. **IPMP 그룹에 대한 테스트 주소가 포함될 기본 IP 인터페이스를 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 관리하도록 지정합니다.**

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

   이전 단계에서는 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 11.2 네트워크 구성 요소의 구성 및 관리 "의 "IPv4 인터페이스를 구성하는 방법"을 참조하십시오.

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

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

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

7. **(옵션) IPMP 그룹의 각 기본 인터페이스에 대해 6단계를 반복합니다.**

예 3-1 DHCP를 사용하여 IPMP 그룹 구성

다음 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.

IPMP 인터페이스가 먼저 만들어집니다.

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

기본 IP 인터페이스가 만들어지고 IPMP 인터페이스에 추가됩니다.

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

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

DHCP 관리 IP 주소가 IPMP 인터페이스에 지정됩니다. 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 DHCP 관리 IP 주소가 IPMP 그룹의 기본 IP 인터페이스에 지정됩니다. 기본 인터페이스에 지정된 IP 주소는 프로브 기반 실패 감지에 사용될 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 net2 인터페이스가 대기 인터페이스로 구성됩니다.

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

▼ 활성-활성 IPMP 그룹을 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 확인하십시오. 지침은 "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "의 "IPv4 인터페이스를 구성하는 방법"을 참조하십시오. 기본 IP 인터페이스가 없는 경우에도 IPMP 인터페이스를 만들 수 있습니다. 하지만 IPMP 인터페이스의 후속 구성이 실패합니다.

또한 SPARC 기반 시스템을 사용하는 경우 각 인터페이스에 고유한 MAC 주소를 구성합니다. "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "의 "각 인터페이스의 MAC 주소가 고유한지 확인하는 방법"을 참조하십시오.

1. **root 역할로 전환합니다.**

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-interface*`1` **[-i** *underinterface*`2` **...]** *ipmp-interface*

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

   참고 - 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.

   참고 - 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 구성하는 방법

다음 절차는 한 인터페이스가 대기 인터페이스로 유지되도록 IPMP 그룹을 구성하는 방법에 대해 설명합니다. 이 인터페이스는 그룹의 활성 인터페이스가 실패하는 경우에만 배포됩니다.

대기 인터페이스에 대한 개요 정보는 "IPMP 인터페이스 구성 유형" [50]을 참조하십시오.

1. **root 역할로 전환합니다.**

2. **IPMP 인터페이스를 만듭니다.**

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

여기서 *ipmp-interface*는 IPMP 인터페이스의 이름을 지정합니다.

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.

참고 - 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 -m ip under-interface
```

예 3-2 활성-대기 IPMP 그룹 구성

다음 예는 활성-대기 IPMP 구성을 만드는 방법을 보여줍니다.

먼저 IPMP 인터페이스가 만들어집니다.

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

기본 IP 인터페이스가 만들어지고 IPMP 인터페이스에 추가됩니다.

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

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

다음으로 IP 주소가 IPMP 인터페이스에 지정됩니다. 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 예의 IP 주소에는 10진수로 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 IP 주소가 IPMP 그룹의 기본 IP 인터페이스에 지정됩니다. 기본 인터페이스에 지정된 IP 주소는 프로브 기반 실패 감지에 사용될 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 net2 인터페이스가 대기 인터페이스로 구성됩니다.

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

관리자는 ipmpstat 명령을 사용하여 IPMP 구성을 확인할 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 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 0.0.0.0이 수신됩니다. 따라서 특정 IP 인터페이스를 사용하여 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 시스템에 있는 인터페이스(예: 기본 인터페이스)를 사용할 기본 경로를 정의해야 합니다. 이후에 기본 경로를 정의한 동일한 인터페이스를 사

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

IPMP를 사용하는 동안 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.

참고 - 다음 작업에서는 기본 경로가 지정된 예로 기본 인터페이스를 사용합니다. 하지만 경로 지정에 사용되었고 나중에 IPMP 그룹의 일부가 된 인터페이스에는 이 유형의 경로 지정 손실 사례가 적용됩니다.

▼ IPMP 사용 중 기본 경로를 보존하는 방법

다음 절차는 IPMP를 구성할 때 기본 경로를 보존하는 방법에 대해 설명합니다.

1. **콘솔을 사용하여 시스템에 로그인합니다.**

   이 절차를 수행하려면 콘솔을 사용해야 합니다. `ssh` 또는 `telnet` 명령을 사용하여 로그인할 경우, 후속 단계를 수행할 때 연결이 끊깁니다.

2. **(옵션) 경로 지정 테이블에 현재 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) 경로 지정 테이블에 재정의된 경로를 표시합니다.**

   ```
   # netstat -nr
   ```

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

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

예 3-3 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 관리

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

- IPMP 그룹에 인터페이스를 추가하는 방법 [72]
- IPMP 그룹에서 인터페이스를 제거하는 방법 [73]
- IPMP 그룹에 IP 주소를 추가하는 방법 [73]
- IPMP 그룹에서 IP 주소를 삭제하는 방법 [74]
- 하나의 IPMP 그룹에서 다른 IPMP 그룹으로 인터페이스를 이동하는 방법 [75]
- IPMP 그룹을 삭제하는 방법 [76]

▼ IPMP 그룹에 인터페이스를 추가하는 방법

시작하기 전에 그룹에 추가한 인터페이스가 필요한 모든 요구 사항을 충족하는지 확인하십시오. 요구 사항 목록은 IPMP 그룹을 계획하는 방법 [64]을 참조하십시오.

1. **root 역할로 전환합니다.**

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 3-4 IPMP 그룹에 인터페이스 추가

다음 예는 net4 인터페이스를 IPMP 그룹 ipmp0에 추가하는 방법을 보여줍니다.

```
# 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. **root 역할로 전환합니다.**
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 3-5 IPMP 그룹에서 인터페이스 제거

다음 예는 net4 인터페이스를 IPMP 그룹 ipmp0에서 제거하는 방법을 보여줍니다.

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

▼ IPMP 그룹에 IP 주소를 추가하는 방법

IPMP 그룹에 IP 주소를 추가하려면 ipadm create-addr 하위 명령을 사용하십시오. IPMP 구성의 경우 IP 주소가 데이터 주소일 수도 있고 테스트 주소일 수도 있습니다. 데이터 주소는 IPMP 인터페이스에 추가되지만, 테스트 주소는 IPMP 인터페이스의 기본 인터페이스에 추가됩니다. 다음 절차에서는 테스트 주소 또는 데이터 주소인 IP 주소를 추가하는 방법에 대해 설명합니다.

1. **root 역할로 전환합니다.**

2. **IPMP 그룹에 IP 주소를 추가합니다.**

   - **다음과 같이 IPMP 그룹에 데이터 주소를 추가합니다.**

     ```
     # 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.

   - **다음과 같이 IPMP 그룹의 기본 인터페이스에 테스트 주소를 추가합니다.**

     ```
     # 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.

▼ IPMP 그룹에서 IP 주소를 삭제하는 방법

IPMP 그룹에서 IP 주소를 삭제하려면 `ipadm delete-addr` 하위 명령을 사용하십시오. IPMP 구성의 경우 데이터 주소는 IPMP 인터페이스에서 호스트되고 테스트 주소는 기본 인터페이스에서 호스트됩니다. 다음 절차에서는 데이터 주소 또는 테스트 주소인 IP 주소를 제거하는 방법을 보여줍니다.

1. **`root` 역할로 전환합니다.**

2. **제거할 IP 주소를 결정합니다.**

   - **다음과 같이 데이터 주소 목록을 표시합니다.**

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

   - **다음과 같이 테스트 주소 목록을 표시합니다.**

     ```
     # ipadm show-addr
     ```

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

3. **IPMP 그룹에서 IP 주소를 제거합니다.**

   - **다음과 같이 데이터 주소를 제거합니다.**

     ```
     # 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.

- **다음과 같이 테스트 주소를 제거합니다.**

  # **ipadm delete-addr** *addrobj*

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

예 3-6 인터페이스에서 테스트 주소 제거

다음 예에서는 예 3-2. "활성-대기 IPMP 그룹 구성"에 표시된 활성-대기 IPMP 그룹 `ipmp0`의 구성을 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. **`root` 역할로 전환합니다.**

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 3-7 다른 IPMP 그룹으로 인터페이스 이동

다음 예에서 IPMP 그룹의 기본 인터페이스는 `net0`, `net1` 및 `net2`입니다. 이 예는 `net0` 인터페이스를 IPMP 그룹 `ipmp0`에서 제거한 다음 `net0`을 IPMP 그룹 `cs-link1`에 배치하는 방법을 보여줍니다.

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

▼ IPMP 그룹을 삭제하는 방법

특정 IPMP 그룹이 더 이상 필요하지 않은 경우 다음 절차를 사용하십시오.

1. **root 역할로 전환합니다.**

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.

   참고 - 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 주소도 시스템에서 삭제됩니다.

예 3-8 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

이 절은 다음 항목으로 구성됩니다.

- “프로브 기반 실패 감지 정보” [77]
- “프로브 기반 실패 감지를 위한 대상 선택 요구 사항” [77]
- “실패 감지 방법 선택” [78]

- 프로브 기반 실패 감지의 대상 시스템을 수동으로 지정하는 방법 [78]
- IPMP 데몬의 동작을 구성하는 방법 [79]

프로브 기반 실패 감지 정보

프로브 기반 실패 감지를 사용하려면 "프로브 기반 실패 감지" [57]에 설명된 대로 대상 시스템을 사용해야 합니다. 프로브 기반 실패 감지의 대상을 식별할 때 `in.mpathd` 데몬은 라우터 대상 모드 또는 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.

`route` 명령이 IPv4 및 IPv6 경로 모두에서 작동합니다. 이때 기본값은 IPv4 경로입니다. `route` 명령 바로 뒤에 `-inet6` 옵션을 사용하면 작업이 IPv6 경로에서 수행됩니다.

프로브 기반 실패 감지의 대상 시스템을 수동으로 지정하는 방법 [78] 절차에서는 프로브 기반 실패 감지를 위해 대상에 지속 경로를 추가하는 데 사용할 정확한 구문을 보여줍니다. `route` 명령과 함께 사용할 수 있는 옵션에 대한 자세한 내용은 `route`(1M) 매뉴얼 페이지 및 "IPMP 배치 중 경로 지정 유지 관리" [70]를 참조하십시오.

프로브 기반 실패 감지를 위한 대상 선택 요구 사항

적합한 대상으로 사용할 네트워크 호스트를 결정하려면 다음 요구 사항을 참조하십시오.

- 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.

실패 감지 방법 선택

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.

프로브 기반 감지 방법을 선택하기 전에 프로브 대상이 "프로브 기반 실패 감지를 위한 대상 선택 요구 사항" [77]에 나열된 요구 사항을 충족하는지 확인하십시오.

전이적 프로브만 사용하려면 다음을 수행하십시오.

1. SMF 명령을 사용하여 IPMP 등록 정보 `transitive-probing`을 사용으로 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.
2. IPMP 그룹에 대해 구성된 기존 테스트 주소를 모두 제거합니다.

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

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

테스트 주소를 사용하여 오류를 프로브하려면 다음을 수행하십시오.

1. 필요한 경우 SMF 명령을 사용하여 전이적 프로브를 사용 안함으로 설정합니다.

   ```
   # svccfg -s svc:/network/ipmp setprop config/transitive-probing=false
   # svcadm refresh  svc:/network/ipmp:default
   ```
2. 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

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

시작하기 전에 프로브 대상이 "프로브 기반 실패 감지를 위한 대상 선택 요구 사항" [77]에 나열된 요구 사항을 충족하는지 확인하십시오.

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 구성 파일 `/etc/default/mpathd`를 사용하여 IPMP 그룹에 대해 시스템 차원의 다음 매개변수를 구성할 수 있습니다.

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

1. **`root` 역할로 전환합니다.**

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

   ```
   # pfedit /etc/default/mpathd
   ```

   지침은 `pfedit`(1M) 매뉴얼 페이지를 참조하십시오.

   다음 세 매개변수 중 하나 이상의 기본값을 변경합니다.

   - 다음과 같이 `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.
   - 다음과 같이 `FAILBACK` 매개변수의 새 값을 입력합니다.

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

yes — IPMP의 페일백 동작에 대한 기본값입니다. 실패한 인터페이스의 복구가 감지되면 네트워크 액세스가 “물리적 인터페이스 복구 감지” [60]에 설명된 대로 복구된 인터페이스로 페일백됩니다.

no — 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.

- 다음과 같이 TRACK_INTERFACES_ONLY_WITH_GROUPS 매개변수의 새 값을 입력합니다.

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

yes — IPMP 동작에 대한 기본값입니다. 이 값을 사용하면 IPMP가 IPMP 그룹으로 구성되지 않은 네트워크 인터페이스를 무시합니다.

no — 네트워크 인터페이스가 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.

이 매개변수와 익명 그룹 기능에 대한 자세한 내용은 “실패 감지 및 익명 그룹 기능” [60]을 참조하십시오.

3. **in.mpathd 데몬을 다시 시작합니다.**

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

새 매개변수 값이 적용되도록 데몬이 다시 시작됩니다.

IPMP 정보 모니터링

다음 예는 ipmpstat 명령을 사용하여 시스템에 있는 IPMP 그룹의 여러 측면을 모니터링하는 방법을 보여줍니다. IPMP 그룹 전체나 해당 기본 IP 인터페이스의 상태를 관찰할 수 있습니다. 또한 IPMP 그룹에 대한 데이터 및 테스트 주소 구성을 확인할 수 있습니다. 또한 같은 명령을 사용하여 실패 감지에 대한 정보를 가져올 수도 있습니다. 자세한 내용은 ipmpstat(1M) 매뉴얼 페이지를 참조하십시오.

ipmpstat 명령을 사용하는 경우 기본적으로 80개 열에 들어가는 가장 의미 있는 필드가 표시됩니다. ipmpstat 명령이 -p 옵션과 함께 사용된 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.

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

---

ipmpstat 명령을 다음 옵션과 함께 사용하여 원하는 정보를 표시할 수 있습니다.

-g 시스템의 IPMP 그룹에 대한 정보를 표시합니다. 예 3-9. “IPMP 그룹 정보 가져오기”를 참조하십시오.

-a IPMP 그룹에 대해 구성된 데이터 주소를 표시합니다. 예 3-10. “IPMP 데이터 주소 정보 가져오기”을 참조하십시오.

-i IPMP 구성과 관련된 IP 인터페이스에 대한 정보를 표시합니다. 예 3-11. “IPMP 그룹의 기본 IP 인터페이스에 대한 정보 가져오기”을 참조하십시오.

-t 실패 감지에 사용되는 대상 시스템에 대한 정보를 표시합니다. 이 옵션은 IPMP 그룹에서 사용되는 테스트 주소도 표시합니다. 예 3-12. “IPMP 프로브 대상 정보 가져오기”를 참조하십시오.

-p 실패 감지에 사용되는 프로브에 대한 정보를 표시합니다. 예 3-13. “IPMP 프로브 관찰”을 참조하십시오.

다음 추가 예는 ipmpstat 명령을 사용하여 시스템의 IPMP 구성에 대한 정보를 표시하는 방법을 보여줍니다.

예 3-9 IPMP 그룹 정보 가져오기

-g 옵션은 기본 인터페이스 상태를 비롯하여 시스템에 있는 여러 IPMP 그룹의 상태를 표시합니다. 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.
  - ok - IPMP 그룹의 모든 기본 인터페이스를 사용할 수 있음을 나타냅니다.
  - degraded - 그룹의 일부 기본 인터페이스를 사용할 수 없음을 나타냅니다.
  - failed - 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 표시됩니다. 인터페이스 상태는 표시된 방식으로 표시됩니다.
  - *interface*(괄호 또는 대괄호 없음) - 활성 인터페이스를 나타냅니다. 활성 인터페이스는 시스템에서 데이터 트래픽을 보내거나 받는 데 사용합니다.
  - *(interface)*(괄호 있음) - 작동하지만 비활성인 인터페이스를 나타냅니다. 인터페이스가 관리 정책에 정의된 대로 사용되고 있지 않습니다.
  - *[interface]*(대괄호 있음) - 인터페이스가 실패했거나 오프라인 상태이므로 인터페이스를 사용할 수 없음을 나타냅니다.

예 3-10 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        --          --
192.168.10.31   up       field2        net2    net2 net7
192.168.10.32   up       field2        net7    net2 net7
192.168.10.33   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) 상태인지 여부를 나타냅니다.

`GROUP` 특정 데이터 주소를 호스트하는 IPMP 인터페이스를 지정합니다. 보통 Oracle Solaris에서 IPMP 그룹의 이름은 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 3-11 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.

- b - 시스템에서 IPMP 그룹에 대한 브로드캐스트 트래픽을 받는 데 해당 인터페이스를 지정했음을 나타냅니다.
- d - 인터페이스가 작동 중지되었으므로 사용할 수 없음을 나타냅니다.
- h - 인터페이스가 다른 인터페이스와 중복 물리적 하드웨어 주소를 공유하며 오프라인 상태로 전환되었음을 나타냅니다. h 플래그는 인터페이스를 사용할 수 없음을 나타냅니다.
- i - 해당 인터페이스에 대해 INACTIVE 플래그가 설정되어 있음을 나타냅니다. 따라서 해당 인터페이스는 데이터 트래픽 송수신에 사용되지 않습니다.
- m - 시스템에서 IPMP 그룹에 대한 IPv4 멀티캐스트 트래픽을 보내고 받는 데 해당 인터페이스를 지정했음을 나타냅니다.
- M - 시스템에서 IPMP 그룹에 대한 IPv6 멀티캐스트 트래픽을 보내고 받는 데 해당 인터페이스를 지정했음을 나타냅니다.
- s - 인터페이스가 대기 인터페이스로 구성되었음을 나타냅니다.

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

- up 또는 down - 링크의 사용 가능 여부를 나타냅니다.
- unknown - 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.

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

STATE — 인터페이스의 전체 상태를 다음과 같이 지정합니다.

- ok - 인터페이스가 온라인 상태이며 실패 감지 방법의 구성에 따라 정상적으로 작동하고 있음을 나타냅니다.

- `failed` - 인터페이스의 링크가 작동 중지되었거나 프로브 감지에서 인터페이스가 트래픽을 보내거나 받을 수 없음이 확인되어 인터페이스가 작동하지 않음을 나타냅니다.
- `offline` - 인터페이스를 사용할 수 없음을 나타냅니다. 일반적으로 인터페이스는 다음과 같은 상황에서 오프라인으로 전환됩니다.
  - 인터페이스를 테스트하고 있습니다.
  - 동적 재구성을 수행하고 있습니다.
  - 인터페이스가 다른 인터페이스와 중복 하드웨어 주소를 공유합니다.
- `unknown` - 프로브 기반 실패 감지를 위한 프로브 대상을 찾을 수 없어서 IPMP 인터페이스의 상태를 확인할 수 없음을 나타냅니다.

예 3-12 IPMP 프로브 대상 정보 가져오기

`-t` 옵션은 IPMP 그룹의 각 IP 인터페이스와 연관된 프로브 대상을 식별합니다. 다음 예의 출력은 프로브 기반 실패 감지에 테스트 주소를 사용하는 IPMP 구성을 보여줍니다.

```
% 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     192.168.10.200   192.168.10.1 192.168.10.2
net6        multicast     192.168.10.201   192.168.10.2 192.168.10.1
net5        multicast     192.168.10.202   192.168.10.1 192.168.10.2
net7        multicast     192.168.10.203   192.168.10.1 192.168.10.2
```

다음 출력은 테스트 주소 없이 전이적 프로브 또는 프로브 기반 실패 감지를 사용하는 IPMP 구성을 보여줍니다.

```
% 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.

- `routes` - 시스템 경로 지정 테이블이 프로브 대상을 찾는 데 사용됨을 나타냅니다.
- `mcast` - 멀티캐스트 ICMP 프로브가 대상을 찾는 데 사용됨을 나타냅니다.
- `disabled` - 인터페이스에 대해 프로브 기반 실패 감지가 사용 안함으로 설정되었음을 나타냅니다.

- `transitive` - 두번째 예와 같이 전이적 프로브가 실패 감지에 사용됨을 나타냅니다. 전이적 프로브와 테스트 주소를 동시에 사용하는 경우 프로브 기반 실패 감지를 구현할 수 없습니다. 테스트 주소를 사용하지 않으려는 경우 전이적 프로브를 사용으로 설정해야 합니다. 전이적 프로브를 사용하지 않으려는 경우 테스트 주소를 구성해야 합니다. 개요는 "프로브 기반 실패 감지" [57]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 전이적 테스트 프로브가 전송되고 있음을 나타냅니다. 데이터를 받는 활성 기본 IP 인터페이스의 경우 표시되는 IP 주소는 송신 ICMP 프로브의 소스 주소를 나타냅니다.

참고 - 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 3-13 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. `root` 역할로 전환하거나 이 명령을 실행할 수 있는 적합한 권한이 있어야 합니다.

다음은 프로브 기반 실패 감지에 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     192.168.10.1
0.26s   net6        602       --       --       --         192.168.10.2
0.25s   net5        601       0.62ms   1.20ms   1.00ms     192.168.10.1
0.26s   net7        603       0.79ms   1.11ms   1.10ms     192.168.10.1
1.66s   net4        613       --       --       --         192.1.2.1
1.70s   net0        603       0.63ms   1.10ms   1.10ms     192.168.85.3
^C
```

다음 출력은 테스트 주소 없이 전이적 프로브 또는 프로브 기반 실패 감지를 사용하는 IPMP 구성의 예입니다.

```
# 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 — 호스트 이름을 지정합니다. 또는 -n 옵션을 -p 옵션과 함께 사용하는 경우 프로브가 전송되는 대상 주소를 지정합니다.

ipmpstat 명령 출력 사용자 정의

-o 옵션을 사용하여 ipmpstat 명령의 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 state를 지정합니다.

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

특정 유형의 정보에 대해 ipmpstat 명령의 모든 필드를 표시하려면 -o 옵션을 all 인수와 함께 사용하십시오.

스크립트에서 ipmpstat 명령 사용

-o 옵션은 스크립트를 통해 또는 명령 별칭을 사용하여 ipmpstat 명령을 실행하는 경우, 특히 시스템에서 구문 분석할 수 있는 출력을 생성하려는 경우에 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 요청될 때 필드에 리터럴 콜론(:) 또는 백슬래시(\)가 포함된 경우 이러한 문자 앞에 백슬래시(\)를 추가하여 이스케이프 처리하거나 제외시킬 수 있습니다.

ipmpstat -P 명령을 올바르게 사용하려면 다음 규칙을 따르십시오.

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

주의 - 이러한 규칙을 하나라도 무시하면 ipmpstat -P가 실패합니다.

다음 예는 -P 옵션 사용 시 올바른 구문을 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 및 -g 옵션을 -P 옵션과 함께 사용하십시오.

-P 옵션은 스크립트에 사용하기 위한 것입니다. 다음 예는 스크립트에서 ipmpstat 명령이 실행되는 방식을 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
}
```

♦ ♦ ♦ 4 장

4 IP 터널 관리 정보

이 장에서는 Oracle Solaris에서 IP 터널 관리에 대한 개요를 제공합니다. 작업 관련 정보는 5장. IP 터널 관리를 참조하십시오.

이 장의 내용:

IP 터널 관리의 새로운 기능

IP 터널 관리는 Oracle Solaris 11에서 네트워크 데이터 링크 관리에 사용되는 새로운 모델과 일치하도록 수정되었습니다. 이 릴리스에서는 `dladm` 명령을 사용하여 IP 터널을 만들어 구성하십시오. 터널은 이 릴리스에서 지원되는 다른 데이터 링크 기능을 사용할 수도 있습니다. 예를 들어 관리상 선택한 이름이 지원되므로 터널에 의미 있는 이름을 지정할 수 있습니다. 자세한 내용은 `dladm`(1M) 매뉴얼 페이지를 참조하십시오.

IP 터널 기능 요약

IP 터널(이 책에서는 간단하게 터널이라고도 함)은 중간 네트워크에서 도메인의 프로토콜을 지원하지 않을 경우 도메인 간에 데이터 패킷을 전송할 수 있도록 해줍니다. 예를 들어 IPv6 네트워크에는 대부분의 네트워크가 IPv4 프로토콜을 사용하는 환경의 경계를 벗어나 통신할 수 있는 방법이 필요합니다. 이 통신은 터널을 통해 가능해집니다. IP 터널은 IP를 사용하여 연결할 수 있는 두 노드 간에 가상 링크를 제공합니다. 따라서 이 링크를 사용하면 IPv4 네트워크를 통해 IPv6 패킷을 전송할 수 있으므로 두 IPv6 사이트 간 IPv6 통신이 가능해집니다.

터널 유형

터널링은 다른 패킷 내에서 IP 패킷을 캡슐화하는 것입니다. 캡슐화는 패킷의 프로토콜을 지원하지 않는 중간 네트워크를 통해 패킷이 대상에 도달할 수 있도록 해줍니다. 터널은 사용되는 패킷 캡슐화의 유형에 따라 달라집니다.

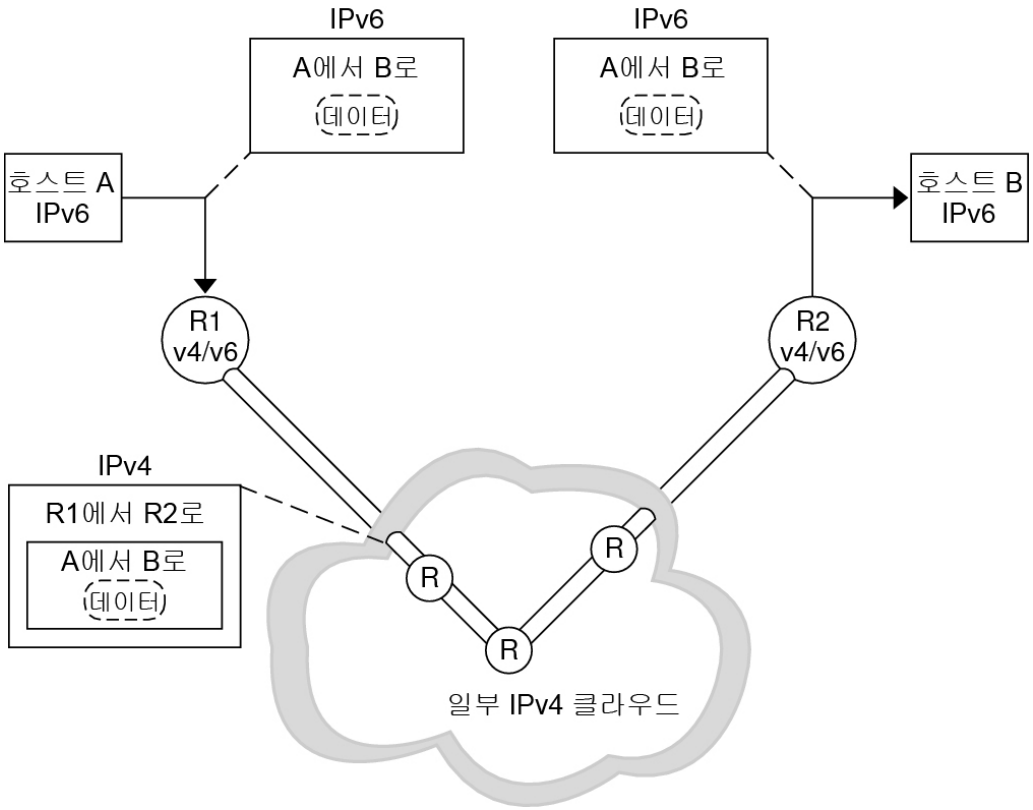
Oracle Solaris에서 지원되는 터널의 유형은 다음과 같습니다.

- **IPv4 터널** - IPv4 패킷은 IPv4 헤더에 캡슐화되고 미리 구성된 유니캐스트 IPv4 대상으로 전송됩니다. 터널을 경유하는 패킷을 보다 명확하게 하기 위해 IPv4 터널을 *IPv4 over IPv4* 터널 또는 *IPv6 over IPv4* 터널이라고도 합니다.
- **6to4 터널** - IPv6 패킷은 IPv4 헤더에 캡슐화되고 패킷당 기준에 따라 자동으로 결정된 IPv4 대상에 전송됩니다. 이때 *6to4* 프로토콜에 정의된 알고리즘을 기준으로 결정됩니다.
- **IPv6 터널** - IPv6 패킷은 IPv4 헤더에 캡슐화되고 패킷당 기준에 따라 자동으로 결정된 IPv4 대상에 전송됩니다. 이때 6to4 프로토콜에 정의된 알고리즘을 기준으로 결정됩니다.

결합된 IPv6 및 IPv4 네트워크 환경에서의 터널

IPv6 도메인이 있는 대부분의 사이트에서는 IPv6 배치의 초기 단계 중 IPv4 네트워크를 순회하여 다른 IPv6 도메인과 통신해야 할 수 있습니다. 다음 그림은 IPv4 라우터를 경유하는 두 IPv6 호스트 간 터널링 방식(그림에서 "R"로 표시됨)을 보여줍니다.

그림 4-1 IPv6 터널링 방식



위 그림에서 터널은 두 개의 라우터로 구성되는데, 이 라우터는 IPv4 네트워크를 경유하여 두 라우터 간에 가상 지점 간 링크를 갖도록 구성되어 있습니다.

IPv6 패킷은 IPv4 패킷 내에서 캡슐화됩니다. IPv6 네트워크의 경계 라우터는 다양한 IPv4 네트워크를 경유하여 대상 IPv6 네트워크의 경계 라우터에 도달하는 지점 간 터널을 설정합니다. 패킷은 터널을 경유하여 대상 경계 라우터로 전송되며, 여기서 패킷이 캡슐화 해제됩니다. 그러면 라우터가 개별 IPv6 패킷을 대상 노드로 전달합니다.

6to4 터널

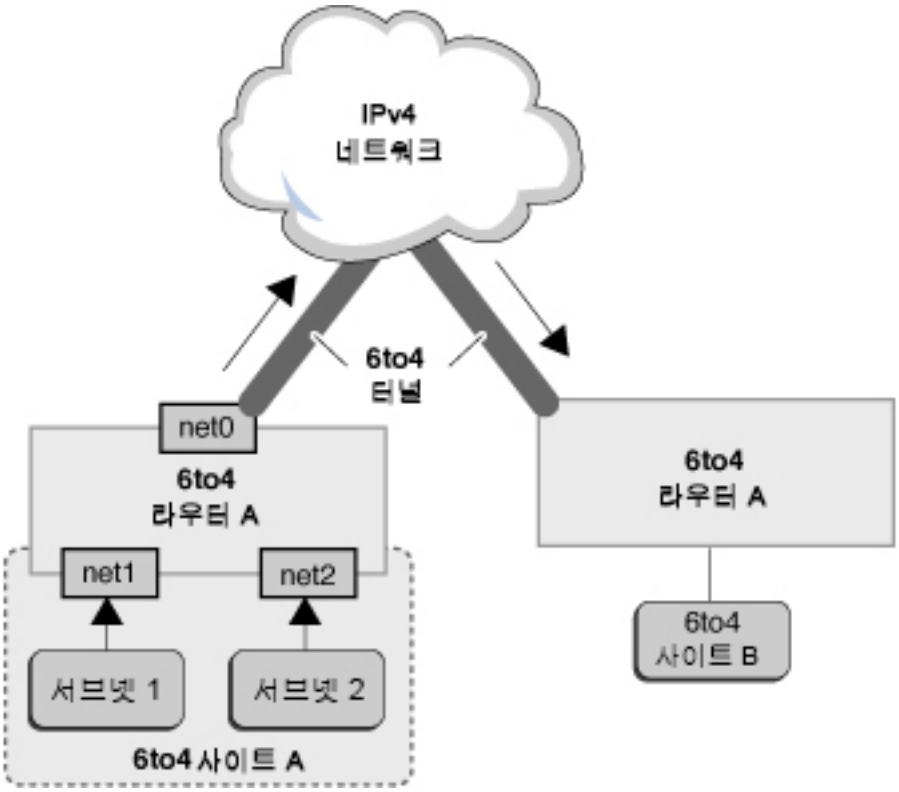
Oracle Solaris는 주소 지정을 IPv4에서 IPv6으로 전환하는 데 선호하는 중간 방식으로 6to4 터널을 제공합니다. *6to4* 터널은 격리된 IPv6 사이트가 IPv6을 지원하지 않는 IPv4 네트워크를 경유하여 자동 터널을 넘어 통신할 수 있도록 해줍니다. 6to4 터널을 사용하려면 먼

저 IPv6 네트워크의 경계 라우터를 6to4 자동 터널의 한 끝점으로 구성해야 합니다. 그러면 6to4 라우터가 다른 6to4 사이트 또는 필요한 경우 고유 IPv6, 비6to4 사이트에 대한 터널에 참여할 수 있습니다.

6to4 터널 토폴로지

6to4 터널은 모든 위치에서 모든 6to4 사이트에 대한 IPv6 연결을 제공합니다. 마찬가지로, 터널이 릴레이 라우터로 전달되도록 구성된 경우 터널은 또한 고유 IPv6 인터넷을 비롯한 모든 IPv6 사이트에 대한 링크로도 작동합니다. 다음 그림은 6to4 터널이 6to4 사이트 간에 이러한 연결을 제공하는 방식을 보여줍니다.

그림 4-2 6to4 사이트 간 터널



이 그림은 격리된 두 6to4 네트워크인 사이트 A 및 사이트 B를 보여줍니다. 각 사이트는 IPv4 네트워크에 대한 외부 연결을 포함하는 라우터를 구성했습니다. IPv4 네트워크를 경유하는 6to4 터널은 6to4 사이트를 연결합니다.

IPv6 사이트가 6to4 사이트가 되려면 먼저 6to4 지원을 위해 적어도 하나의 라우터 인터페이스를 구성해야 합니다. 이 인터페이스는 IPv4 네트워크에 대한 외부 연결을 제공해야 합니다. 위 그림에서 라우터 A의 인터페이스인 `net0`은 사이트 A를 IPv4 네트워크에 연결해 줍니다. `net0`에 구성한 주소는 전역적으로 고유해야 합니다. 라우터에서 6to4를 지원하도록 터널 인터페이스를 구성하려면 먼저 IPv4 주소로 `net0` 인터페이스를 구성해야 합니다.

그림에서 6to4 사이트 A는 두 개의 서브넷으로 구성되며, 두 서브넷은 라우터 A의 `net1` 및 `net2` 인터페이스에 연결되어 있습니다. 사이트 A의 서브넷에 있는 모든 IPv6 호스트는 라우터 A로부터 알림을 수신하면 6to4 파생 주소를 사용하도록 자동 재구성됩니다.

사이트 B는 또 다른 분리된 6to4 사이트입니다. 사이트 A에서 보내는 트래픽을 올바르게 수신하려면 사이트 B의 경계 라우터가 6to4를 지원하도록 구성되어야 합니다. 그렇지 않으면 라우터가 사이트 A로부터 수신하는 패킷이 인식되지 않고 삭제됩니다.

`6to4relay` 명령

6to4 터널링을 사용하면 격리된 6to4 사이트 간에 통신할 수 있습니다. 그러나 원시 비6to4 IPv6 사이트를 포함하는 패킷을 전송하려면 6to4 라우터가 6to4 릴레이 라우터를 사용하여 터널을 설정해야 합니다. 그러면 *6to4* 릴레이 라우터가 6to4 패킷을 IPv6 네트워크 및 원시 IPv6 사이트로 전송합니다. 6to4 지원 사이트가 원시 IPv6 사이트와 데이터를 교환해야 하는 경우 `6to4relay` 명령을 사용하여 해당 터널을 사용으로 설정하십시오.

참고 - 릴레이 라우터 사용은 보안되지 않으므로 Oracle Solaris에서는 기본적으로 릴레이 라우터가 사용 안함으로 설정되어 있습니다. 이 시나리오를 배치하기 전에 6to4 릴레이 라우터에 대한 터널 생성과 관련된 문제를 주의 깊게 고려하십시오. 자세한 내용은 "6to4 릴레이 라우터에 대한 터널을 사용으로 설정하기 위한 고려 사항" [96]을 참조하십시오. 6to4 릴레이 라우터 지원을 사용하려는 경우 IP 터널을 만들고 구성하는 방법 [102]에서 관련 절차를 참조하십시오.

자세한 내용은 `6to4`(7M) 매뉴얼 페이지를 참조하십시오.

6to4 터널을 경유하는 패킷 플로우

이 절에서는 6to4 사이트의 호스트에서 원격 6to4 사이트의 호스트로의 패킷 플로우에 대해 설명합니다. 이 시나리오는 그림 4-2, "6to4 사이트 간 터널"에 표시된 토폴로지를 사용합니다. 또한 이 시나리오는 6to4 라우터와 6to4 호스트가 이미 구성되어 있다고 가정합니다.

패킷 플로우는 다음과 같습니다.

1. 6to4 사이트 A의 서브넷 1에 있는 호스트가 6to4 사이트 B에 있는 호스트를 대상으로 지정하는 전송을 보냅니다. 각 패킷 헤더에는 6to4 파생 소스 주소와 6to4 파생 대상 주소가 있습니다.
2. 사이트 A의 라우터가 IPv4 헤더 내에서 각 6to4 패킷을 캡슐화합니다. 이 프로세스에서 라우터는 캡슐화 헤더의 IPv4 대상 주소를 사이트 B의 라우터 주소로 설정합니다. 터널

인터페이스를 경유하는 각 IPv6 패킷의 IPv6 대상 주소에는 IPv4 대상 주소도 포함되어 있습니다. 따라서 라우터가 캡슐화 헤더에 설정된 IPv4 대상 주소를 확인할 수 있습니다. 그런 다음 라우터는 표준 IPv4 경로 지정 프로시저를 사용하여 IPv4 네트워크를 통해 패킷을 전달합니다.

3. 패킷이 거쳐 가는 IPv4 라우터는 전달 시 패킷의 IPv4 대상 주소를 사용합니다. 이 주소는 라우터 B에 있는 인터페이스의 전역적으로 고유한 IPv4 주소이며, 6to4 의사 인터페이스로도 사용됩니다.
4. 사이트 A의 패킷이 라우터 B에 도달하여 IPv4 헤더에서 IPv6 패킷이 캡슐화 해제됩니다.
5. 그런 다음 라우터 B가 IPv6 패킷의 대상 주소를 사용하여 패킷을 사이트 B의 수신자 호스트로 전달합니다.

6to4 릴레이 라우터에 대한 터널을 사용으로 설정하기 위한 고려 사항

6to4 릴레이 라우터는 원시 IPv6, 비6to4 네트워크와 통신해야 하는 6to4 라우터에서 터널 끝점으로 사용됩니다. 릴레이 라우터는 기본적으로 6to4 사이트와 원시 IPv6 사이트를 연결해 줍니다. 이 솔루션은 안전하지 않으므로 기본적으로 Oracle Solaris에서는 6to4 릴레이 라우터 지원이 사용으로 설정되어 있지 않습니다. 그러나 사이트에 이러한 터널이 필요할 경우 `6to4relay` 명령을 사용하여 다음 그림에 표시된 것과 같이 터널링을 사용으로 설정할 수 있습니다.

그림 4-3 6to4 사이트와 6to4 릴레이 라우터 간 터널

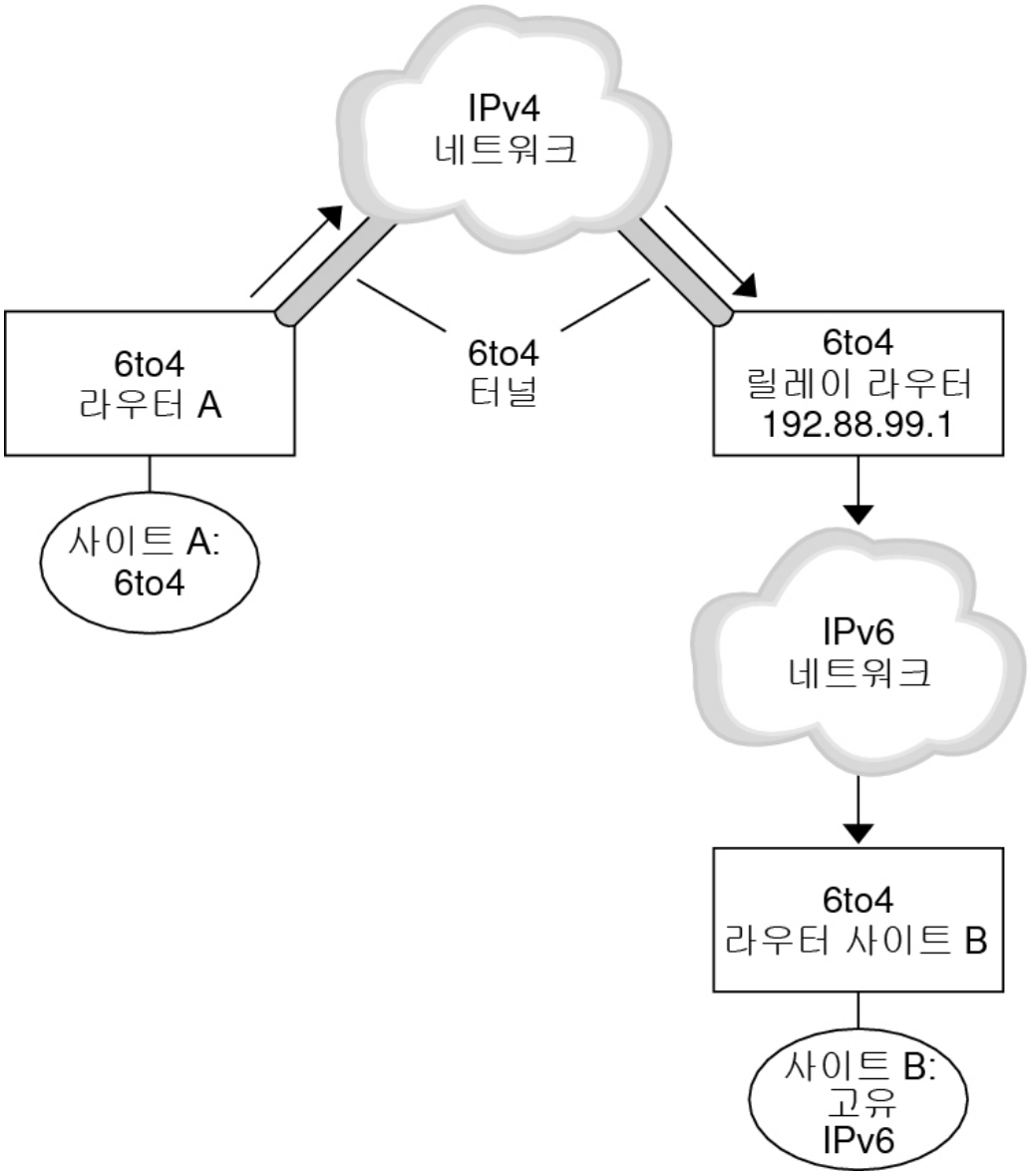


그림 4-3. “6to4 사이트와 6to4 릴레이 라우터 간 터널”에서 6to4 사이트 A는 고유 IPv6 사이트 B에 있는 노드와 통신해야 합니다. 이 그림은 사이트 A에서 IPv4 네트워크를 경유하여 6to4 터널에 도달하는 트래픽 경로를 보여줍니다. 터널의 끝점은 6to4 라우터 A와 6to4 릴

레이 라우터입니다. 6to4 릴레이 라우터를 넘어가면 IPv6 사이트 B가 연결되어 있는 IPv6 네트워크입니다.

6to4 사이트와 원시 IPv6 사이트 간 패킷 플로우

이 절에서는 6to4 사이트에서 고유 IPv6 사이트로의 패킷 플로우에 대해 설명합니다. 이 시나리오는 그림 4-3. “6to4 사이트와 6to4 릴레이 라우터 간 터널”에 표시된 토폴로지를 사용합니다.

패킷 플로우는 다음과 같습니다.

1. 6to4 사이트 A에 있는 호스트가 원시 IPv6 사이트 B에 있는 호스트를 대상으로 지정하는 전송을 보냅니다. 각 패킷 헤더에는 6to4 파생 주소가 소스 주소로 포함되어 있습니다. 대상 주소는 표준 IPv6 주소입니다.
2. 사이트 A의 6to4 라우터가 IPv4 헤더 내에서 각 패킷을 캡슐화합니다. 이 헤더에는 6to4 릴레이 라우터의 IPv4 주소가 대상으로 포함되어 있습니다. 6to4 라우터는 표준 IPv4 경로 지정 프로시저를 사용하여 IPv4 네트워크를 통해 패킷을 전달합니다. 패킷이 거쳐 가는 IPv4 라우터는 패킷을 6to4 릴레이 라우터로 전달합니다.
3. 사이트 A와 물리적으로 가장 가까운 애니캐스트 6to4 릴레이 라우터가 `192.88.99.1` 애니캐스트 그룹에 전송되는 패킷을 검색합니다.

참고 - 6to4 릴레이 라우터 애니캐스트 그룹의 일부인 6to4 릴레이 라우터의 IP 주소는 `192.88.99.1`입니다. 이 애니캐스트 주소는 6to4 릴레이 라우터의 기본 주소입니다. 특정 6to4 릴레이 라우터를 사용해야 하는 경우 기본 주소를 대체하고 해당 라우터의 IPv4 주소를 지정할 수 있습니다.

---

4. 릴레이 라우터가 6to4 패킷에서 IPv4 헤더를 캡슐화 해제하여 원시 IPv6 대상 주소를 표시합니다.
5. 이제 패킷 라우터가 IPv6 전용 패킷을 IPv6 네트워크로 전송합니다. 이 네트워크에서 패킷이 사이트 B의 라우터에 의해 검색됩니다. 라우터가 패킷을 대상 IPv6 노드로 전달합니다.

IP 터널 배치 정보

IP 터널을 제대로 배치하려면 두 가지 기본 작업을 수행해야 합니다. 먼저 터널 링크를 만든 다음 터널을 경유하는 IP 인터페이스를 구성하십시오. 다음은 터널 및 해당 IP 인터페이스를 만들기 위한 요구 사항입니다.

IP 터널을 만들기 위한 요구 사항

IP 터널을 성공적으로 만들려면 다음 요구 사항을 충족해야 합니다.

- 리터럴 IP 주소 대신 호스트 이름을 사용할 경우 해당 이름이 터널 유형과 호환되는 유효한 IP 주소로 분석되어야 합니다.
- 만드는 IPv4 또는 IPv6 터널이 구성된 다른 터널과 동일한 터널 소스 주소 및 터널 대상 주소를 공유해서는 안됩니다.
- 만드는 IPv4 또는 IPv6 터널이 기존 6to4 터널과 동일한 터널 소스 주소를 공유해서는 안됩니다.
- 6to4 터널을 만드는 경우, 터널이 구성된 다른 터널과 동일한 터널 소스 주소를 공유해서는 안됩니다.

자세한 내용은 “Oracle Solaris 11.2의 네트워크 배치 계획 ”의 “네트워크에서 터널 사용 계획”을 참조하십시오.

IP 터널 및 IP 인터페이스 요구 사항

각 터널 유형에는 터널을 경유하도록 구성한 IP 인터페이스에 대한 특정 IP 주소 요구 사항이 있습니다. 이러한 요구 사항은 다음 표에 요약되어 있습니다.

표 4-1 터널 및 IP 인터페이스 요구 사항

터널 유형	터널을 통해 허용되는 IP 인터페이스	IP 인터페이스 요구 사항
IPv4 터널	IPv4 인터페이스	로컬 및 원격 주소를 수동으로 지정해야 합니다.
	IPv6 인터페이스	`ipadm create-addr -T addrconf` 명령을 실행하면 로컬 및 원격 링크 로컬 주소가 자동으로 설정됩니다. `ipadm`(1M) 매뉴얼 페이지를 참조하십시오.
IPv6 터널	IPv4 인터페이스	로컬 및 원격 주소를 수동으로 지정해야 합니다.
	IPv6 인터페이스	`ipadm create-addr -T addrconf` 명령을 실행하면 로컬 및 원격 링크 로컬 주소가 자동으로 설정됩니다. `ipadm`(1M) 매뉴얼 페이지를 참조하십시오.
6to4 터널	IPv6 인터페이스만	`ipadm create-ip` 명령을 실행하면 기본 IPv6 주소가 자동으로 설정됩니다. `ipadm`(1M) 매뉴얼 페이지를 참조하십시오.

IPv6 또는 IPv4 터널을 경유하는 IPv6에 대해 자동으로 설정되는 링크 로컬 주소는 로컬 및 원격 `interface-id`를 `ipadm create-addr -T addrconf` 명령과 함께 지정하여 대체할 수 있습니다.

♦ ♦ ♦ 5 장

5 IP 터널 관리

이 장에서는 Oracle Solaris에서 IP 터널 관리 작업에 대해 설명합니다. Oracle Solaris에서 IP 터널 관리에 대한 개요는 4장. IP 터널 관리 정보를 참조하십시오.

이 장의 내용:

Oracle Solaris에서 IP 터널 관리

이 Oracle Solaris 릴리스에서 터널 관리는 IP 인터페이스 구성과 구분됩니다. IP 터널의 데이터 링크 측면은 `dladm` 명령을 사용하여 관리하고, 구성의 IP 측면(IP 터널에 대한 구성 포함)은 `ipadm` 명령을 사용하여 관리하십시오.

다음 `dladm` 하위 명령은 IP 터널을 구성하는 데 사용됩니다.

- `create-iptun`
- `modify-iptun`
- `show-iptun`
- `delete-iptun`
- `set-linkprop`

자세한 내용은 `dladm`(1M) 매뉴얼 페이지를 참조하십시오.

참고 - IP 터널 관리는 IPsec 구성과 밀접한 관련이 있습니다. 예를 들어 IPsec VPN(Virtual Private Network)은 IP 터널링의 주요 용도 중 하나입니다. Oracle Solaris의 네트워크 보안에 대한 자세한 내용은 “Oracle Solaris 11.2의 네트워크 보안 ”의 6 장, “IP Security Architecture 정보”를 참조하십시오. IPsec을 구성하려면 “Oracle Solaris 11.2의 네트워크 보안 ”의 7 장, “IPsec 구성”을 참조하십시오.

IP 터널 관리

이 절은 다음 항목으로 구성됩니다.

▼ IP 터널을 만들고 구성하는 방법

1. **`root` 역할로 전환합니다.**

2. **터널을 만듭니다.**

```
# dladm create-iptun [-t] -T type -a [local|remote]=addr,... tunnel-link
```

`-t` — 임시 터널을 만듭니다. 기본적으로 이 명령은 영구 터널을 만듭니다.

터널을 경유하는 영구 IP 인터페이스를 구성하려는 경우 `-t` 옵션을 사용하지 말고 영구 터널을 만들어야 합니다.

`-T` *type* — 만들려는 터널의 유형을 지정합니다. 이 인수는 모든 터널 유형을 만드는 데 필수입니다.

`-a [local|remote]=`*address,...* — 로컬 주소 및 원격 터널 주소에 해당하는 리터럴 IP 주소 또는 호스트 이름을 지정합니다. 주소는 유효해야 하며 이미 시스템에 생성되어 있어야 합니다. 터널의 유형에 따라 주소를 한 개만 지정하거나, 로컬 및 원격 주소를 모두 지정합니다. 로컬 및 원격 주소를 모두 지정하는 경우 주소를 쉼표로 구분해야 합니다.

- `IPv4` 터널이 작동하려면 로컬 및 원격 IPv4 주소가 필요합니다.
- `IPv6` 터널이 작동하려면 로컬 및 원격 IPv6 주소가 필요합니다.
- `6to4` 터널이 작동하려면 로컬 IPv4 주소가 필요합니다.

참고 - 영구 IP 터널 데이터 링크 구성에 호스트 이름을 주소로 사용하는 경우 호스트 이름은 구성 저장소에 저장됩니다. 이후에 시스템을 부트할 때 이름이 터널을 만든 당시에 사용했던 IP 주소와 다른 IP 주소로 분석되는 경우 터널이 새 구성을 사용하게 됩니다.

tunnel-link IP 터널 링크를 지정합니다. 이 릴리스에서는 네트워크 링크 관리에서 의미 있는 이름이 지원되므로 터널 이름이 더 이상 만들려는 터널의 유형으로 제한되지 않습니다. 대신 관리상 선택한 이름을 터널에 지정할 수 있습니다. 터널 이름은 문자열과 PPA(Physical Point of Attachment) 번호로 구성됩니다(예: *mytunnel0*). 의미 있는 이름 지정을 제어하는 규칙은 "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "의 "유효한 링크 이름 규칙"을 참조하십시오.

3. **(옵션) 홉 한계 또는 캡슐화 한계에 대한 값을 설정합니다.**

```
# dladm set-linkprop -p [hoplimit=value] [encaplimit=value] tunnel-link
```

`hoplimit` IPv6 경유 터널링에 대한 터널 인터페이스의 홉 한계를 지정합니다. *hoplimit*는 IPv4 경유 터널의 IPv4 TTL(time to live) 필드에 해당합니다.

`encaplimit` 패킷에 허용되는 중첩 터널링의 레벨 수를 지정합니다. 이 옵션은 IPv6 터널에만 적용됩니다.

`hoplimit` 및 `encaplimit` 등록 정보에 대해 설정한 값은 허용되는 범위 내에 있어야 합니다. `hoplimit` 및 `encaplimit` 등록 정보는 터널 링크 등록 정보입니다. 따라서 이러한 등록 정보는 다른 링크 등록 정보와 동일한 `dladm` 하위 명령으로 관리됩니다. 사용하는 하위 명령은 `dladm set-linkprop`, `dladm reset-linkprop` 및 `dladm show-linkprop`입니다.

4. **터널을 경유하는 IP 인터페이스를 만듭니다.**

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

여기서 *tunnel-interface*는 터널 링크와 동일한 이름을 사용합니다.

5. **터널 인터페이스에 로컬 및 원격 IP 주소를 지정합니다.**

```
# ipadm create-addr [-t] -a local=address,remote=address interface
```

여기서 *interface*는 터널 인터페이스를 지정합니다.

자세한 내용은 `ipadm`(1M) 매뉴얼 페이지 및 "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "를 참조하십시오.

6. **(옵션) 터널 IP 인터페이스 구성의 상태를 확인합니다.**

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

예 5-1 IPv4 터널을 경유하는 IPv6 인터페이스 만들기

다음 예는 지속적인 IPv6 over IPv4 터널을 만드는 방법을 보여줍니다.

```
# dladm create-iptun -T ipv4 -a local=192.0.2.23,remote=203.0.113.14 private0
# dladm set-linkprop -p hoplimit=200 private0
# ipadm create-ip private0
# ipadm create-addr -T addrconf private0
private0/v6
# ipadm show-addr private0/
ADDROBJ           TYPE     STATE        ADDR
private0/v6       addrconf ok fe80::c000:217->fe80::cb00:710e
```

대체 주소를 추가하려면 동일한 구문을 사용합니다. 예를 들어 다음과 같이 전역 주소를 추가할 수 있습니다.

```
# ipadm create-addr -a local=2001:db8:4728::1,remote=2001:db8:4728::2 private0
private0/v6a
# ipadm show-addr private0/
ADDROBJ           TYPE     STATE        ADDR
private0/v6       addrconf ok fe80::c000:217->fe80::cb00:710e
private0/v6a      static   ok 2001:db8:4728::1->2001:db8:4728::2
```

IPv6 주소의 `2001:db8` 접두어는 설명서 예제에 특별히 사용되는 특수 IPv6 접두어입니다.

예 5-2 IPv4 터널을 경유하는 IPv4 인터페이스 만들기

다음 예는 지속적인 IPv4 over IPv4 터널을 만드는 방법을 보여줍니다.

```
# dladm create-iptun -T ipv4 -a local=192.0.2.23,remote=203.0.113.14 vpn0
# ipadm create-ip vpn0
# ipadm create-addr -a local=10.0.0.1,remote=10.0.0.2 vpn0
vpn0/v4
# ipadm show-addr vpn0/
ADDROBJ           TYPE     STATE        ADDR
vpn0/v4           static   ok 10.0.0.1->10.0.0.2
```

이 터널을 경유하는 패킷에 보안 연결을 제공하도록 IPsec 정책을 추가로 구성할 수 있습니다. 자세한 내용은 "Oracle Solaris 11.2의 네트워크 보안 "의 7 장, "IPsec 구성"을 참조하십시오.

예 5-3 IPv6 터널을 경유하는 IPv6 인터페이스 만들기

다음 예는 지속적인 IPv6 over IPv6 터널을 만드는 방법을 보여줍니다.

```
# dladm create-iptun -T ipv6 -a local=2001:db8:feed::1234,remote=2001:db8:beef::4321 tun0
# ipadm create-ip tun0
# ipadm create-addr -T addrconf tun0
tun0/v6
# ipadm show-addr tun0/
ADDROBJ           TYPE     STATE        ADDR
tun0/v6           addrconf ok fe80::1234->fe80::4321
```

전역 주소 또는 대체 로컬 및 원격 주소 등의 주소를 추가하려면 다음과 같이 `ipadm` 명령을 사용하십시오.

```
# ipadm create-addr -a local=2001:db8:cafe::1,remote=2001:db8:cafe::2 tun0
```

```
tun0/v6a
# ipadm show-addr tun0/
ADDROBJ           TYPE     STATE        ADDR
tun0/v6           addrconf ok fe80::1234->fe80::4321
tun0/v6a          static   ok 2001:db8:cafe::1->2001:db8:cafe::2
```

▼ 6to4 터널을 구성하는 방법

6to4 터널을 구성하는 경우 6to4 라우터를 네트워크의 6to4 사이트에 있는 노드에 대한 IPv6 라우터로 사용해야 합니다. 따라서 6to4 라우터를 구성할 때 물리적 인터페이스에서 해당 라우터를 IPv6 라우터로도 구성해야 합니다. Oracle Solaris 호스트를 라우터로 구성하는 자세한 방법은 "Oracle Solaris 11.2 시스템을 라우터 또는 로드 밸런서로 구성 "의 "IPv6 라우터 구성"을 참조하십시오.

1. **6to4 터널을 만듭니다.**

   ```
   # dladm create-iptun -T 6to4 -a local=address tunnel-link
   ```

   -a local=*address* — 터널 로컬 주소를 지정합니다. 이 주소가 시스템에 이미 존재해야 유효한 주소입니다.

   tunnel-link — IP 터널 링크를 지정합니다. 네트워크 링크 관리에서 의미 있는 이름이 지원되는 경우, 터널 이름이 더 이상 만들려는 터널의 유형으로 제한되지 않습니다. 대신 관리상 선택한 이름을 터널에 지정할 수 있습니다. 터널 이름은 문자열과 PPA 번호로 구성됩니다(예: *mytunnel0*). 자세한 내용은 "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "의 "유효한 링크 이름 규칙"을 참조하십시오.

2. **터널 IP 인터페이스를 만듭니다.**

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

   여기서 *tunnel-interface*는 터널 링크와 동일한 이름을 사용합니다.

3. **(옵션) 터널용 대체 IPv6 주소를 추가합니다.**

4. **`/etc/inet/ndpd.conf` 파일을 편집합니다.**

   ```
   # pfedit /etc/inet/ndpd.conf
   ```

5. **다음 두 라인을 파일에 추가하여 6to4 경로 지정을 알립니다.**

   ```
   if subnet-interface AdvSendAdvertisements 1
   IPv6-address subnet-interface
   ```

   여기서 첫번째 라인은 알림을 수신하는 서브넷을 지정하며, *subnet-interface*는 서브넷이 연결된 링크를 나타냅니다. 두번째 행의 IPv6 주소에는 6to4 터널의 IPv6 주소에 사용되는 6to4 접두어 `2000`이 지정됩니다.

ndpd.conf 파일에 대한 자세한 내용은 ndpd.conf(4) 매뉴얼 페이지를 참조하십시오.

6. **IPv6 전달을 사용으로 설정합니다.**

```
# ipadm set-prop -p forwarding=on ipv6
```

7. **다음 옵션 중 하나를 선택합니다.**

   - **라우터를 재부트합니다.**
   - **/etc/inet/in.ndpd 데몬에 대해 sighup을 실행하여 라우터 알림 전송을 시작합니다.**
     6to4 접두어를 수신할 각 서브넷의 IPv6 노드가 새 6to4 파생 주소로 자동 구성됩니다.

8. **6to4 사이트에서 사용되는 이름 서비스에 노드의 새 6to4 파생 주소를 추가합니다.**
   지침은 "Oracle Solaris 11.2 네트워크 구성 요소의 구성 및 관리 "의 4 장, "Oracle Solaris 클라이언트에서 이름 지정 및 디렉토리 서비스 관리"를 참조하십시오.

예 5-4 6to4 터널 만들기

다음 예는 6to4 터널을 만드는 방법을 보여줍니다. IPv6 인터페이스만 6to4 터널을 경유하도록 구성할 수 있습니다. 이 예에서 서브넷 인터페이스는 /etc/inet/ndpd.conf가 참조하는 net0입니다.

```
# dladm create-iptun -T 6to4 -a local=192.0.2.23 tun0
# ipadm create-ip tun0
# ipadm show-addr
ADDROBJ           TYPE     STATE        ADDR
lo0/v4            static   ok           127.0.0.1/8
net0/v4           dhcp     ok           192.0.2.23/24
lo0/v6            static   ok           ::1/128
tun0/v6           static   ok           2002:c000:217::1/16

# ipadm create-addr -T addrconf net0
net0/v6
# ipadm create-addr -a 2002:c000:217:cafe::1 net0
net0/v6a
# ipadm show-addr
ADDROBJ           TYPE     STATE        ADDR
lo0/v4            static   ok           127.0.0.1/8
net0/v4           dhcp     ok           192.0.2.23/24
lo0/v6            static   ok           ::1/128
net0/v6           addrconf ok     fe80::214:4fff:fef9:b1a9/10
net0/v6a          static   ok           2002:c000:217:cafe::1/64
tun0/v6           static   ok           2002:c000:217::1/16

# vi /etc/inet/ndpd.conf
if net0 AdvSendAdvertisements on
prefix 2002:c000:217:cafe::0/64 net0
```

```
# ipadm set-prop -p forwarding=on ipv6
```

6to4 터널에 대한 IPv6 주소 접두어는 2002입니다.

▼ 6to4 릴레이 라우터에 대한 6to4 터널을 사용으로 설정하는 방법

주의 - 주요 보안 문제로 인해 6to4 릴레이 라우터 지원은 기본적으로 Oracle Solaris에서 사용 안함으로 설정되어 있습니다. 자세한 내용은 "Oracle Solaris 11.2의 네트워크 관리 문제 해결 "의 "6to4 릴레이 라우터로 터널링 시 발생하는 보안 문제"를 참조하십시오.

시작하기 전에 6to4 릴레이 라우터에 대한 6to4 터널을 사용으로 설정하기 전에 다음 작업을 수행하십시오.

- 사이트에서 6to4 라우터를 구성합니다. IP 터널을 만들고 구성하는 방법 [102]을 참조하십시오.
- 6to4 릴레이 라우터에 대한 터널링 관련 보안 문제를 검토합니다.

1. **다음 방법 중 하나를 사용하여 6to4 릴레이 라우터에 대한 터널을 사용으로 설정합니다.**

   - **애니캐스트 6to4 릴레이 라우터에 대한 터널을 사용으로 설정합니다.**

     ```
     # 6to4relay -e
     ```

     -e 옵션은 6to4 라우터와 애니캐스트 6to4 릴레이 라우터 간에 터널을 설정합니다. 애니캐스트 6to4 릴레이 라우터는 잘 알려진 IPv4 주소 192.88.99.1을 사용합니다. 사용자의 사이트와 물리적으로 가장 가까운 애니캐스트 릴레이 라우터가 6to4 터널의 끝점이 됩니다. 이 릴레이 라우터는 6to4 사이트와 원시 IPv6 사이트 간 패킷 전달을 처리합니다.

     자세한 내용은 RFC 3068, "An Anycast Prefix for 6to4 Relay Routers" (http://www.rfc-editor.org/rfc/rfc3068.txt)를 참조하십시오.

   - **특정 6to4 릴레이 라우터에 대한 터널을 사용으로 설정합니다.**

     ```
     # 6to4relay -e -a relay-router-address
     ```

     -a 옵션은 특정 라우터 주소가 뒤에 이어짐을 나타냅니다. *relay-router-address*는 터널을 사용으로 설정할 특정 6to4 릴레이 라우터의 IPv4 주소로 바꿉니다.

   6to4 릴레이 라우터에 대한 터널은 6to4 터널 의사 인터페이스를 제거할 때까지 활성 상태로 유지됩니다.

2. **터널이 더 이상 필요하지 않을 경우 6to4 릴레이 라우터에 대한 터널을 삭제합니다.**

   ```
   # 6to4relay -d
   ```

3. **(옵션) 6to4 릴레이 라우터에 대한 터널이 재부트 후에도 보존되도록 합니다.**
   6to4 라우터가 재부트될 때마다 사이트에서 6to4 릴레이 라우터에 대한 터널을 원래 상태로 복원해야 하는 이유가 있을 수 있습니다. 이 시나리오를 지원하려면 다음을 수행해야 합니다.

   a. **`/etc/default/inetinit` 파일을 편집합니다.**

      ```
      # pfedit /etc/default/inetinit
      ```

      수정할 라인은 파일의 맨 끝에 있습니다.

   b. **`ACCEPT6TO4RELAY=NO` 라인의 `NO` 값을 `YES`로 변경합니다.**

   c. **(옵션) 재부트 후에도 보존되는 특정 6to4 릴레이 라우터에 대한 터널을 만듭니다.**
      `RELAY6TO4ADDR` 매개변수에 대해 `192.88.99.1` 주소를 사용하려는 6to4 릴레이 라우터의 IPv4 주소로 변경합니다.

예 5-5 6to4 릴레이 라우터 지원에 대한 상태 정보 가져오기

`/usr/bin/6to4relay` 명령을 사용하여 6to4 릴레이 라우터에 대한 지원을 사용으로 설정할지 여부를 확인합니다. 다음 예는 6to4 릴레이 라우터에 대한 지원이 사용 안함으로 설정된 경우(Oracle Solaris의 기본값)의 출력을 보여줍니다.

```
# 6to4relay
6to4relay: 6to4 Relay Router communication support is disabled.
```

6to4 릴레이 라우터에 대한 지원이 사용으로 설정된 경우 다음과 같은 출력이 표시됩니다.

```
# 6to4relay
6to4relay: 6to4 Relay Router communication support is enabled.
IPv4 remote address of Relay Router=192.88.99.1
```

IP 터널 구성 수정

다음 명령 구문을 사용하여 터널 구성을 변경할 수 있습니다.

```
# dladm modify-iptun -a [local|remote]=addr,... tunnel-link
```

기존 터널의 유형은 수정할 수 없습니다. 따라서 `-T` *type* 옵션은 이 명령에 사용할 수 없습니다. 수정 가능한 터널 매개변수는 다음과 같습니다.

`-a [local|remote]=`*address,...*
: 로컬 주소 및 원격 터널 주소에 해당하는 리터럴 IP 주소 또는 호스트 이름을 지정합니다. 터널의 유형에 따라 주소를 한 개만 지정하거나, 로컬 및 원격 주소를 모두 지정합니다. 로컬 및 원격 주소를 모두 지정하는 경우 주소를 쉼표로 구분해야 합니다.

- `IPv4` 터널이 작동하려면 로컬 및 원격 IPv4 주소가 필요합니다.
- `IPv6` 터널이 작동하려면 로컬 및 원격 IPv6 주소가 필요합니다.

- 6to4 터널이 작동하려면 로컬 IPv4 주소가 필요합니다.

  영구 IP 터널 데이터 링크 구성에 호스트 이름을 주소로 사용하는 경우 호스트 이름은 구성 저장소에 저장됩니다. 이후에 시스템을 부트할 때 이름이 터널을 만든 당시에 사용했던 IP 주소와 다른 IP 주소로 분석되는 경우 터널이 새 구성을 사용하게 됩니다.

  터널의 로컬 및 원격 주소를 변경하는 경우 해당 주소가 수정하려는 터널의 유형과 일치하는지 확인합니다.

- 터널 링크의 이름을 변경하려면 다음과 같이 modify-iptun 명령 대신 dladm rename-link 명령을 사용합니다.

  # **dladm rename-link** *old-tunnel-link new-tunnel-link*

- hoplimit 또는 encaplimit 등의 터널 등록 정보를 변경하려면 modify-iptun 명령 대신 dladm set-linkprop 명령을 사용합니다.

예 5-6 터널의 주소 및 등록 정보 수정

다음 예는 두 개의 절차로 구성됩니다. 먼저 IPv4 터널 vpn0의 로컬 및 원격 주소가 일시적으로 변경됩니다. 나중에 시스템을 재부트하면 터널이 원래 주소를 사용하도록 복원됩니다. 두 번째 명령은 vpn0의 hoplimit를 60으로 변경하는 방법을 보여줍니다.

```
# dladm modify-iptun -t -a local=10.8.48.149,remote=192.168.2.3 vpn0

# dladm set-linkprop -p hoplimit=60 vpn0
```

IP 터널 구성 표시

다음 명령 구문을 사용하여 IP 터널 구성을 표시할 수 있습니다.

dladm show-iptun [-p] -o *fields* **[***tunnel-link***]**

-p — 시스템에서 분석 가능한 형식으로 정보를 표시합니다. 이 인수는 선택적입니다.

-o *fields* — 특정 터널 정보를 표시하는 선택한 필드를 표시합니다.

tunnel-link — 표시할 구성 정보를 포함하는 터널을 지정합니다. 이 인수는 선택적입니다. 터널 이름을 생략하면 시스템에 있는 모든 터널에 대한 정보가 표시됩니다.

예 5-7 모든 터널에 대한 정보 표시

다음 예에서는 한 개의 터널만 시스템에 존재합니다.

```
# dladm show-iptun
```

```
LINK     TYPE     FLAGS     LOCAL          REMOTE
tun0     6to4     --       192.168.35.10   --
vpn0     ipv4     --       10.8.48.149     192.168.2.3
```

예 5-8 시스템에서 분석 가능한 형식으로 선택한 필드 표시

다음 예에서는 터널 정보가 포함된 특정 필드만 표시됩니다.

```
# dladm show-iptun -p -o link,type,local
tun0:6to4:192.168.35.10
vpn0:ipv4:10.8.48.149
```

IP 터널 등록 정보 표시

다음 명령 구문을 사용하여 터널 링크의 등록 정보를 표시할 수 있습니다.

 # dladm show-linkprop [-c] [-o *fields***] [***tunnel-link***]**

-c 시스템에서 분석 가능한 형식으로 정보를 표시합니다. 이 인수는 선택적입니다.

-o *fields* 링크 등록 정보에 대한 특정 정보를 제공하는 선택한 필드를 표시합니다.

tunnel-link 표시할 등록 정보를 사용하는 터널을 지정합니다. 이 인수는 선택적입니다. 터널 이름을 생략하면 시스템에 있는 모든 터널에 대한 정보가 표시됩니다.

예 5-9 터널 등록 정보 표시

다음 예는 터널의 링크 등록 정보를 모두 표시하는 방법을 보여줍니다.

```
# dladm show-linkprop iptun0
LINK     PROPERTY        PERM VALUE        EFFECTIVE    DEFAULT   POSSIBLE
iptun0   autopush        rw   --           --           --        --
iptun0   zone            rw   --           --           --        --
iptun0   state           r-   up           up           up        up,down
iptun0   mtu             rw   1480         1480         1480      1280-1480
iptun0   maxbw           rw   --           --           --        --
iptun0   cpus            rw   --           --           --        --
iptun0   rxfanout        rw   --           8            8         --
iptun0   pool            rw   --           --           --        --
iptun0   priority        rw   medium       medium       medium    low,medium,
                                                                  high
iptun0   hoplimit        rw   64           64           64        1-255
iptun0   protection      rw   --           --           --        mac-nospoof,
                                                                  restricted,
                                                                  ip-nospoof,
```

```
                                                                    dhcp-nospoof
iptun0   allowed-ips     rw   --          --          --        --
iptun0   allowed-dhcp-cids rw --          --          --        --
iptun0   rxrings         rw   --          --          --        --
iptun0   txrings         rw   --          --          --        --
iptun0   txringsavail    r-   0           0           --        --
iptun0   rxringsavail    r-   0           0           --        --
iptun0   rxhwclntavail   r-   0           0           --        --
iptun0   txhwclntavail   r-   0           0           --        --
```

▼ IP 터널을 삭제하는 방법

1. **인터페이스의 유형에 따라 터널을 통해 구성된 IP 인터페이스를 제거합니다.**

   # ipadm delete-ip *tunnel-link*

   참고 - 터널을 성공적으로 삭제하기 위해서는 터널에 설정된 기존 IP 인터페이스가 없어야 합니다.

2. **IP 터널을 삭제합니다.**

   # dladm delete-iptun *tunnel-link*

   이 명령의 유일한 옵션은 터널을 일시적으로 삭제하는 -t입니다. 시스템을 재부트하면 터널이 복원됩니다.

예 5-10 IPv6 인터페이스로 구성된 IPv6 터널 삭제

다음 예에서는 영구 터널이 영구적으로 삭제됩니다.

```
# ipadm delete-ip ip6.tun0
# dladm delete-iptun ip6.tun0
```

색인

ㅍ

ㅎ