7주차_네트워크 계층 3

상태	진행 중
강의	CS 스터디
작성일	@2024년 5월 15일
자료	https://github.com/IT-Book-Organization/Computer-Networking_A-Top-Down-Approach

5.3 인터넷에서의 AS 내부 라우팅: OSPF

자율 시스템(autonomous system, AS)

- 각 AS는 동일한 관리 제어하에 있는 라우터의 그룹으로 구성된다.
- 자율 시스템은 전 세계적으로 고유한 `AS 번호(autonomous system number, ASN)`으로 식별된다.
- 같은 AS 안에 있는 라우터들은 동일한 라우팅 알고리즘을 사용하고 상대방에 대한 정보를 갖고 있다.
- 자율 시스템 내부에서 동작하는 라우팅 알고리즘을 `AS 내부 라우팅 프로토콜(intra-autonomous system routing protocol)`이라고 한다.

개방형 최단 경로 우선(OSPF) 프로토콜

> 개방형 최단 경로 우선(open shortest path first, OSPF) 라우팅과 IS-IS(Intermediate System to Intermediate System)는 인터넷에서 AS 내부 라우팅에 널리 사용된다.

OSPF는 링크 상태 정보는 플러딩(flooding)하고 다익스트라 최소비용 경로 알고리즘을 사용하는 링크 상태 알고리즘이다.

- OSPF를 이용하여 각 라우터는 전체 AS에 대한 완벽한 토폴로지 지도(그래프)를 얻는다.

- 각 라우터는 자신을 루트 노드로 두고 모든 서브넷에 이르는 최단 경로 트리를 결정하기 위해 혼자서 다익스트라의 최단 경로 알고리즘을 수행한다.
- OSPF를 사용하는 라우터는 자율 시스템 내의 다른 모든 라우터에게 라우팅 정보를 브로드캐스팅한다.
    - 링크 상태가 변경될 때마다
    - 링크 상태가 변경되지 않았더라도 정기적으로 (최소한 30분마다 한 번씩)
- OSPF 메시지에 포함된 상태 정보는 인터넷 프로토콜에 의해 전달되며, 상위 계층 프로토콜 번호로는 OSPF를 의미라는 89를 갖는다.
    - 따라서 OSPF 프로토콜은 신뢰할 수 있는 메시지 전송과 링크 상태의 브로드캐스트와 같은 기능을 스스로 구현해야한다.
    - 또한 OSPF 프로토콜은 링크가 동작하고 있는지 검사하고, OSPF라우터가 네트워크 전반의 링크 상태에 대한 이웃과 라우터의 데이터베이스를 얻을 수 있도록 해야 한다.

OSPF 링크 가중치 설정

링크 상태 라우팅을 설명하면서 우리는 순서를 묵시적으로 아래와 같이 가정했다.

1. 링크 가중치가 설정되고,
2. OSPF같은 라우팅 알고리즘이 수행되며
3. LS 알고리즘에 의해 계산된 라우팅 테이블의 내용에 따라 트래픽이 흐른다.

이를 원인과 결과 방식으로 설명하면,

- 원인 : 링크 가중치가 주어지고
- 결과 : 이에 따라 전체 비용을 최소화하는 라우팅 경로가 결정된다.

실제로는 링크 가중치와 라우팅 경로 간의 원인과 결과 관계는 반대가 될 수도 있다.

네트워크 운영자가 어떤 트래픽 관리 목표를 충족시키는 라우팅 경로를 얻기 위해 링크 가중치를 설정할 수 있다.

즉, 트래픽 흐름에 대한 바람직한 경로가 알려져 있고, OSPF 라우팅 알고리즘이 이 경로대로 구성하게 되도록 OSPF 링크 가중치를 찾아야 한다.

따라서 관리자는 모든 링크 비용을 1로 설정함으로써 최소 홉 라우팅이 이루어지게 하거나,

적은 대역폭을 가진 링크 사용을 억제하기 위해 링크 용량에 반비례하게 링크 가중치를 설정할 수 있다.

OSFP에 구현된 개선사항들

보안

- OSPF 라우터들 간의 정보 교환(e.g., 링크 상태 갱신)을 인증할 수 있으며, 인증을 통해 신뢰할 수 있는 라우터들만이 AS 내부의 OSPF 프로토콜에 참여할 수 있다.
- 원래 라우터 간의 OSPF 패킷은 인증을 하지 않으므로 위조될 수 있다.

복수 동일 비용 경로

- 하나의 목적지에 대해 동일한 비용을 가진 여러 개의 경로가 존재할 때 OSPF는 여러 개의 경로를 사용할 수 있도록 한다.
- 즉, 비용이 동일한 여러 개의 경로가 있을 때 **모든 트래픽을 전달하기 위한 단 하나의 경로를 선택할 필요가 없다.**

유니캐스트와 멀티캐스트 라우팅의 통합 지원

`MOSPF(multicast OSPF)`는 멀티캐스트 라우팅 기능을 제공하기 위해 OSPF를 단순 확장했다.

- 기존의 OSPF 링크 데이터베이스를 사용
- OSPF 링크 상태 브로드캐스트 메커니즘에 새로운 형태의 링크 상태 알림을 추가

단일 AS 내에서의 계층 지원

OSPF의 자율 시스템(AS)는 계층적인 영역으로 구성될 수 있다.

- 각 영역은 자신의 OSPF 링크 상태 라우팅 알고리즘을 수행한다.
- 한 역역 내의 라우터는 **같은 영역 내의 라우터들에게만** 링크 상태를 브로드캐스트한다.
- 각 영역 내에서 하나 혹은 그 이상의 `영역 경계 라우터(area border router)`가 **영역 외부로의 패킷 라우팅을 책임진다.**
- `백본 영역`의 주요 역할은 AS 내 영역 간의 트래픽을 라우팅하는 것이다.

AS 내 영역 간 라우팅을 위해서는,

1. 영역 경계 라우터로 패킷을 라우팅한다. (영역 내 라우팅)
2. 백본을 통과하여 목적지 영역의 영역 경계 라우터로 라우팅한다.
3. 그 후 최종 목적지로 라우팅한다.

5.4 인터넷 서비스 제공업자(ISP) 간의 라우팅: BGP

패킷이 **여러 AS를 통과하도록** 라우팅할 때, (e.g., 한국에 있는 스마트폰이 실리콘 밸리에 있는 데이터 센터로 전송을 할 경우) 자율 시스템 간 라우팅 프로토콜(inter-autonomous system routing protocol)이 필요하다.

통신하는 AS들은 같은 AS 간 라우팅 프로토콜을 수행해야만 한다.

실제로 인터넷의 모든 AS는 경계 게이트웨이 프로토콜(Border GateWay Protocol, BGP)을 사용하며,

이는 거리 벡터 라우팅과 같은 줄기에서 나왔다고 볼 수 있는 **분산형 비동기식 프로토콜**이다.

5.4.1 BGP의 역할

같은 AS 내에 있는 목적지에 대해서는 라우터 포워딩 테이블 엔트리들이 해당 AS의 AS 내부 라우팅 프로토콜에 의해 결정된다.

하지만 목적지가 AS 외부에 있는 경우, BGP가 필요하다.

BGP에서는 패킷이 CIDER(Classless Inter-Domain Routing) 형식으로 표현된, 주소의 앞쪽 프리픽스(prefix)를 향해 전달된다.

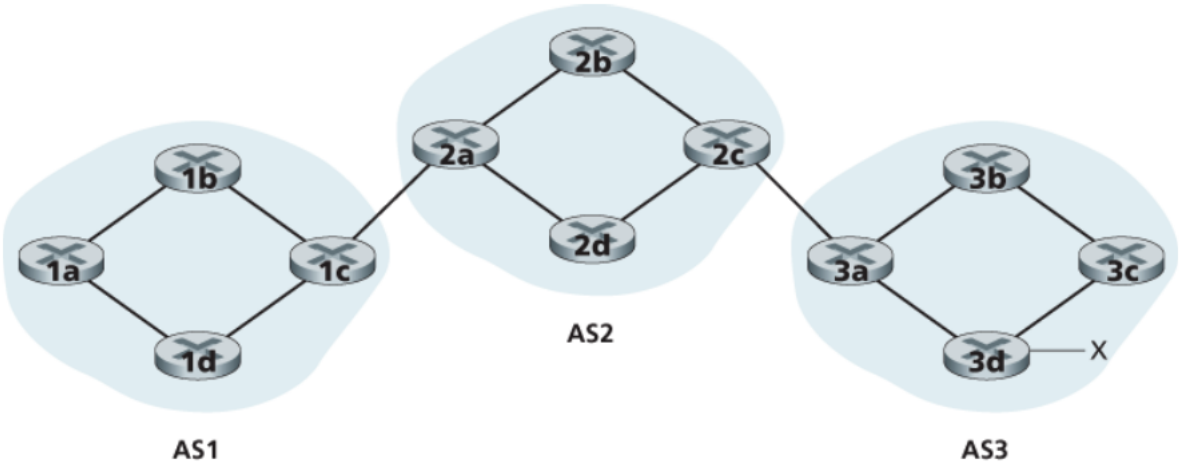
각 프리픽스는 서브넷이나 서브넷의 집합을 나타낸다. (e.g., 139.16.68/22)

라우터의 포워딩 테이블은 (x, I) 같은 형식의 엔트리들을 갖게 된다

- x : 주소 프리픽스 (e.g., 139.16.68/22)
- I : 라우터 인터페이스의 인터페이스 번호

5.4.2 BGP 경로 정보 알리기

아래의 단순한 네트워크는 3개의 자율 시스템 AS1, AS2, AS3 를 가지며, AS3는 주소 프리픽스가 x인 서브넷을 포함한다.



각 AS에서 각각의 라우터들은 게이트웨이 라우터(gateway router) 또는 내부 라우터(internal router)다.

게이트웨이 라우터(gateway router)

- AS의 경계에 있는 라우터
- 다른 AS들에 있는 하나 또는 여러 개의 라우터와 직접 연결된다.
- e.g., AS1의 라우터 1c

내부 라우터(internal router)

- 자신의 AS 내에 있는 호스트 및 라우터와만 연결된다.
- e.g., AS1의 라우터 1a, 1b, 1d

자율 시스템은 서로 메시지를 보내지 않고 **라우터가 보낸다.**

BGP 연결 (BGP connection)

- BGP에서 라우터의 쌍들은 포트 번호가 179이고 반영구적인 TCP 연결을 통해 라우팅 정보를 교환한다.
- 이 TCP 연결을 통해 모든 BGP 메시지가 전송된다.

외부 BGP(external BCP, eBGP) 연결

- **2개의 AS**를 연결하는 BGP 연결

내부 BGP(internal BGP, iBGP) 연결

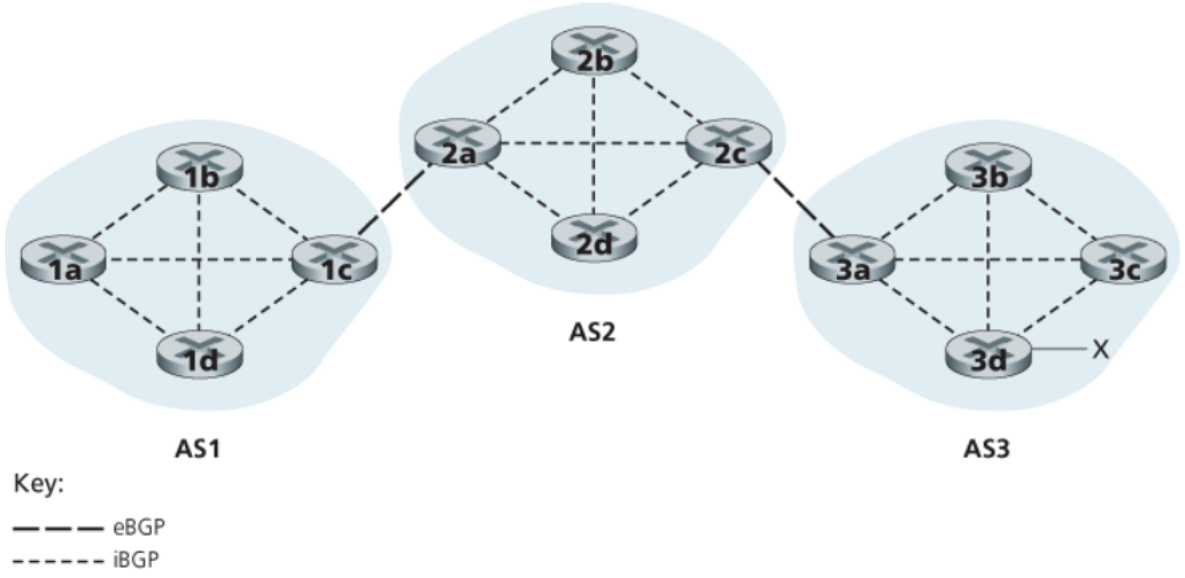
- **같은 AS 내의 라우터**를 연결하는 BGP 연결

보통 각기 다른 AS에 속하는 게이트웨이 라우터들을 직접 연결하는 링크에는 `eBGP 연결`이 존재한다.

각 AS 내부 라우터 간에는 `iBGP 연결`도 존재하며,

아래 그림은 한 AS 내 모든 라우터의 쌍 각각에 대해 하나씩의 BGP 연결을 두는 일반적인 설정 모습을 보인다.

> iBGP 연결은 물리적인 링크와 항상 일치하지는 않는다.



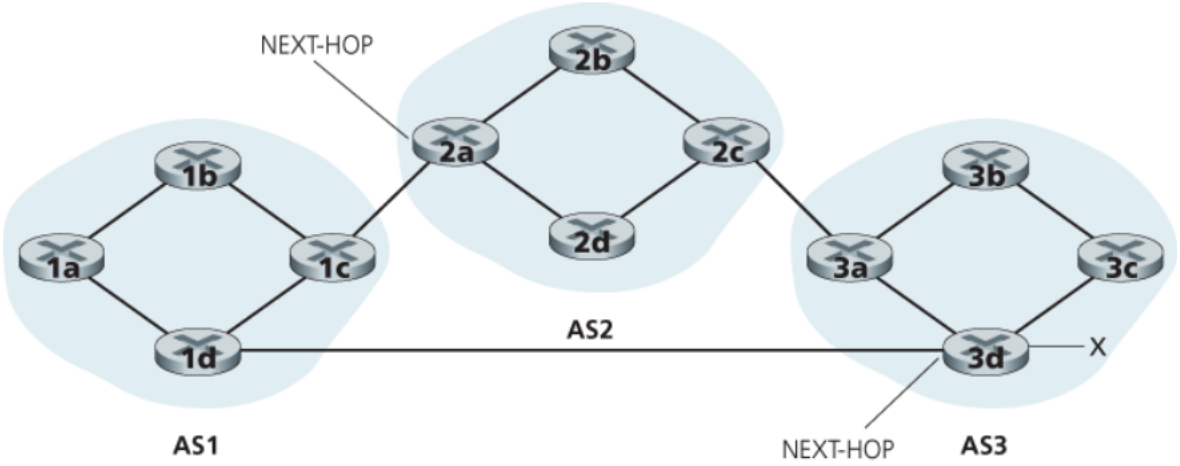
5.4.3 최고의 경로 결정

AS-PATH

- **알림 메시지가 통과하는 AS들의 리스트**를 담는다.
  1. 프리픽스가 어떤 AS에 전달되었을 때
  2. 그 AS는 자신의 ASN을 `AS-PATH` 내 현재 리스트에 추가한다.
- 메시지의 루프를 감지하고 방지하기 위해 활용한다.
  1. 어떤 라우터가 자신의 AS가 경로 리스트에 포함되어 있는 것을 발견하면
  2. 그 알림 메시지를 버린다.

NEXT-HOP

- **AS-PATH가 시작되는 라우터 인터페이스의 IP 주소**



AS1에서 AS2를 통과하여 x로 가는 '`AS2 AS3 x`' 경로의 NEXT-HOP 속성은 **라우터 2a의 왼쪽 인터페이스의 IP 주소**다.

AS1에서 AS2를 우회하여 x로 가는 '`AS3 x`' 경로의 NEXT-HOP 속성은 **라우터 3d의 맨 왼쪽 인터페이스의 IP 주소**다.

NEXT-HOP 속성은 **AS1에 속하지 않는** 라우터의 IP 주소이다.

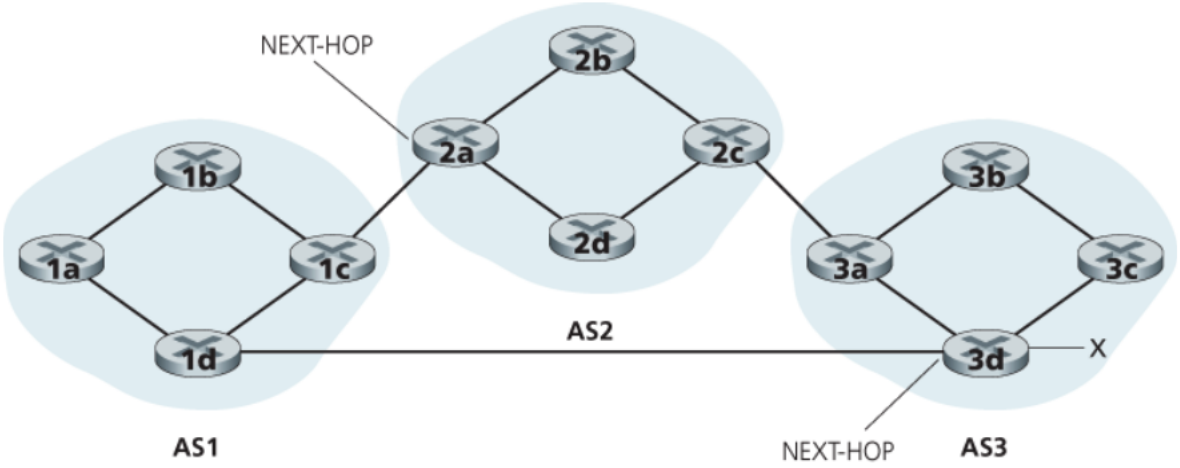
그러나 이 IP 주소를 포함하는 서브넷이 AS1에 **직접적으로 연결된다.**

뜨거운 감자 라우팅(hot potato routing)

> 가능한 모든 경로 중, 경로 각각의 시작점인 **NEXT-HOP 라우터까지의 경로 비용이 최소가 되는 경로**를 선택한다.

1. 여러 게이트웨이를 통해 서브넷 x에 도달할 수 있다는 사실을 AS 간 프로토콜로부터 알게 된다.
2. 각 게이트웨이까지의 최소 비용 경로를 정하기 위해 AS 내부 프로토콜을 통해 얻은 라우팅 정보를 이용한다.
3. **뜨거운 감자 라우팅: 가장 적은 비용의 게이트웨이를 선택한다.**
4. 포워딩 테이블로부터 최소 비용 게이트웨이로의 인터페이스 I를 결정한 후 포워딩 테이블에 `(x, I)`를 추가한다.

> 포워딩 테이블에 AS 외부의 목적지를 추가할 때 AS 간 라우팅 프로토콜(BGP)과 AS 내부 라우팅 프로토콜(e.g., OSPF) **둘 다가 사용된다.**



1. `라우터 1b`는 주소가 `x`로 시작하는 서브넷으로 가는 2개의 BGP 경로를 안다.
2. `NEXT-HOP 라우터 2a와 3d` 각각에 대해 최소 비용을 가진 AS 내부 경로를 찾기 위해 **AS 내부 라우팅 정보를 조사한다.**
3. 이들 최소 비용 경로 중에서도 **가장 적은 비용을 가진 경로를 선택한다.**

   비용을 거쳐가야 하는 링크의 수로 정의하면,

   - 라우터 1b에서 `라우터 2a`까지의 최소 비용 : 2
   - 라우터 1b에서 `라우터 3d`까지의 최소 비용 : 3

   따라서 라우터 2a가 선택된다.
4. 라우터 1b는 자신의 (AS 내부 알고리즘에 의해 설정된) 포워딩 테이블을 관찰하여 라우터 2a로 가기 위한 인터페이스 I를 찾아내고,

   엔트리 `(x, I)`를 **자신의 포워딩 테이블에 추가한다.**

> **기본 아이디어**
>
> 라우터가 목적지까지의 경로 중 **자신의 AS 바깥에 있는 부분에 대한 비용은 신경 쓰지 않고**
>
> 최대한 신속하게(가능한 한 최소의 비용으로) 패킷을 자신의 AS 밖으로 내보내는 것이다.

즉, 뜨거운 감자 라우팅은 오로지 자신의 경로 중에서 **자기 AS 내부 비용만 줄이려는** 이기적인 알고리즘이다.

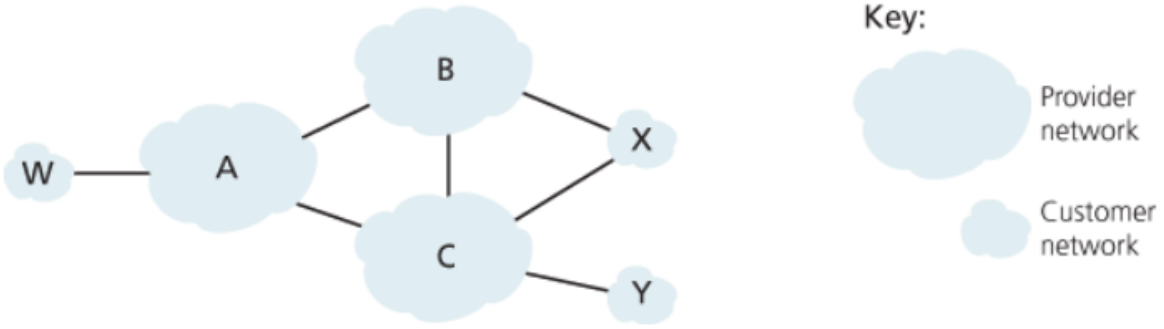
이를 사용하면 한 AS 내 2개의 라우터가 동일한 목적지 주소에 대해 각기 다른 AS 경로를 선택할 수도 있다.

e.g. 위의 예시에서 라우터 1b는 AS2를 통해 서브넷 x로 패킷을 보냈지만, 라우터 1d는 바로 AS3로 보내 서브넷 x에 도달한다.

5.4.5 라우팅 정책

라우터가 목적지까지의 경로를 선택하려고 할 때 AS 라우팅 정책은 최단 AS-PATH나 뜨거운 감자 라우팅 등의 다른 모든 고려사항보다 우선시된다.

- **W, X, Y**는 사용자 접속 ISP
- **A, B, C**는 백본 제공자 네트워크
  - 트래픽을 서로에게 직접 보낸다.
  - 그들의 사용자 네트워크에 완전한 BGP 정보를 제공한다.



왜 AS 간 라우팅과 AS 내부 라우팅 프로토콜이 다를까?

정책

AS 간 라우팅은 정책 이슈가 지배한다. 특정 AS에서 시작된 트래픽이 다른 특정 AS를 통과할 수 없다는 것은 중요할 수 있으며, 특정 AS가 다른 AS들 사이에서 어떤 트래픽을 전달할지 결정할 수 있기를 원하는 것 역시 당연하다.

반면, 하나의 AS 안에서는 **모든 것이 동일한 관리 통제하에 있으므로** 정책 문제는 경로 선택에 그리 중요하지 않다.

확장성

AS 간 라우팅에서 수많은 네트워크로, 또는 네트워크 간 경로 설정을 처리하기 위한 라우팅 알고리즘과 자료 구조의 능력은 매우 중요한 문제다.

반면, 한 AS 내에서는 확장성이 중요하지 않다. 하나의 ISP가 너무 커지면 이를 2개의 AS로 분리하고, 이 새로운 두 AS 사이에서 AS 간 라우팅을 수행할 수 있다.

(OSPF가 하나의 AS를 여러 영역으로 나눔으로써 계층을 만드는 것을 허용함)

성능

AS 간 라우팅은 정책 지향형이므로 사용하는 라우터의 품질(e.g., 성능)은 부수적인 관심사에 지나지 않는다.

단일 AS에서는 이러한 정책의 고려가 중요하지 않으므로, 경로의 성능 수준에 좀 더 초점을 두고 라우팅을 한다.

소프트웨어 정의 네트워크(SDN) 제어 평면

SDN 구조의 특징

플로우 기반 포워딩

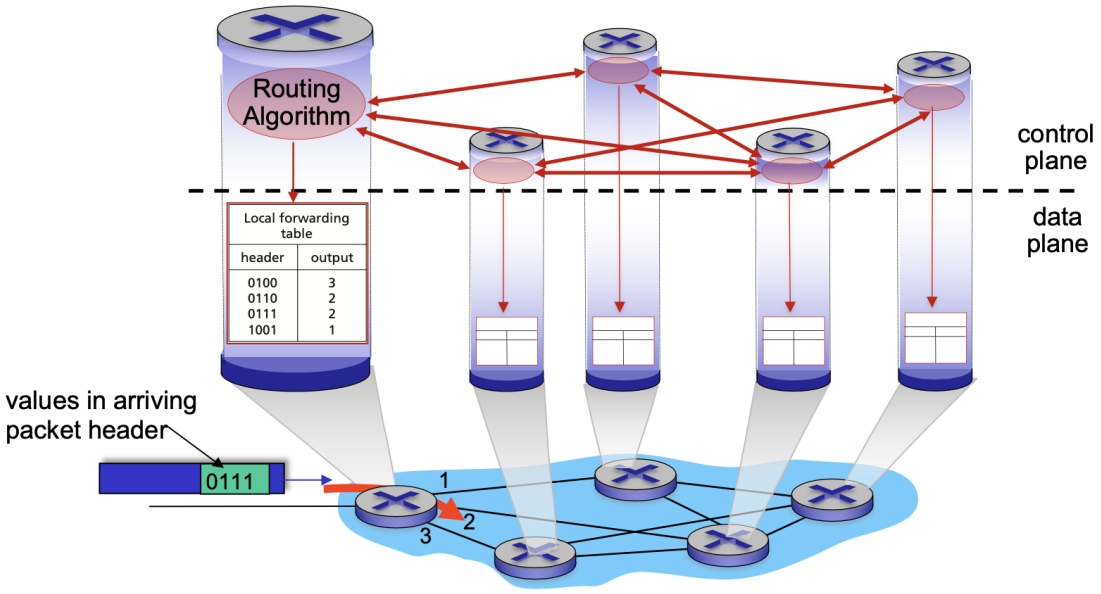
데이터 평면과 제어 평면의 원리

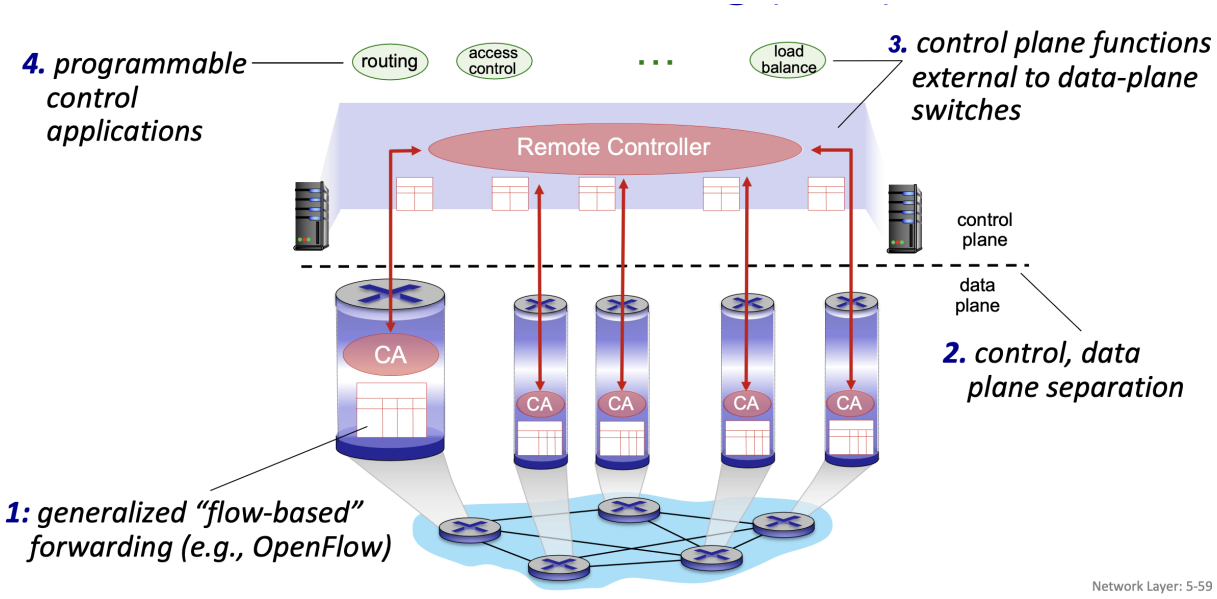
데이터 평면

- 네트워크의 스위치들로 구성된다. (이들은 상대적으로 단순하지만 빠른 장치들)
- 자신들의 플로우 테이블 내용을 기반으로 '매치 플러스 액션'을 수행한다.

제어 평면

- 서버와 스위치들의 플로우 테이블을 결정, 관리하는 소프트웨어로 이루어진다.



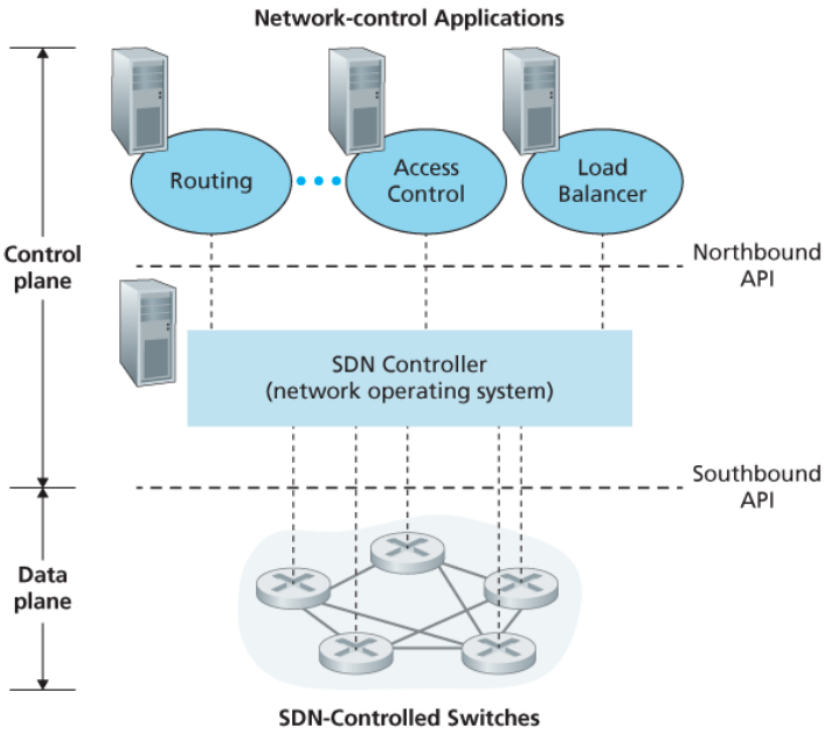


네트워크 제어 기능이 데이터 평면 스위치 내부에 존재

SDN 제어 평면은 소프트웨어로 구현되어 있으며, 네트워크 스위치로부터 멀리 떨어진 별도의 서버에서 수행된다.

제어 평면은 2개의 구성요소로 이루어진다.

1. SDN 컨트롤러(또는 네트워크 운영체제)
2. SDN 네트워크 제어 애플리케이션들의 집합

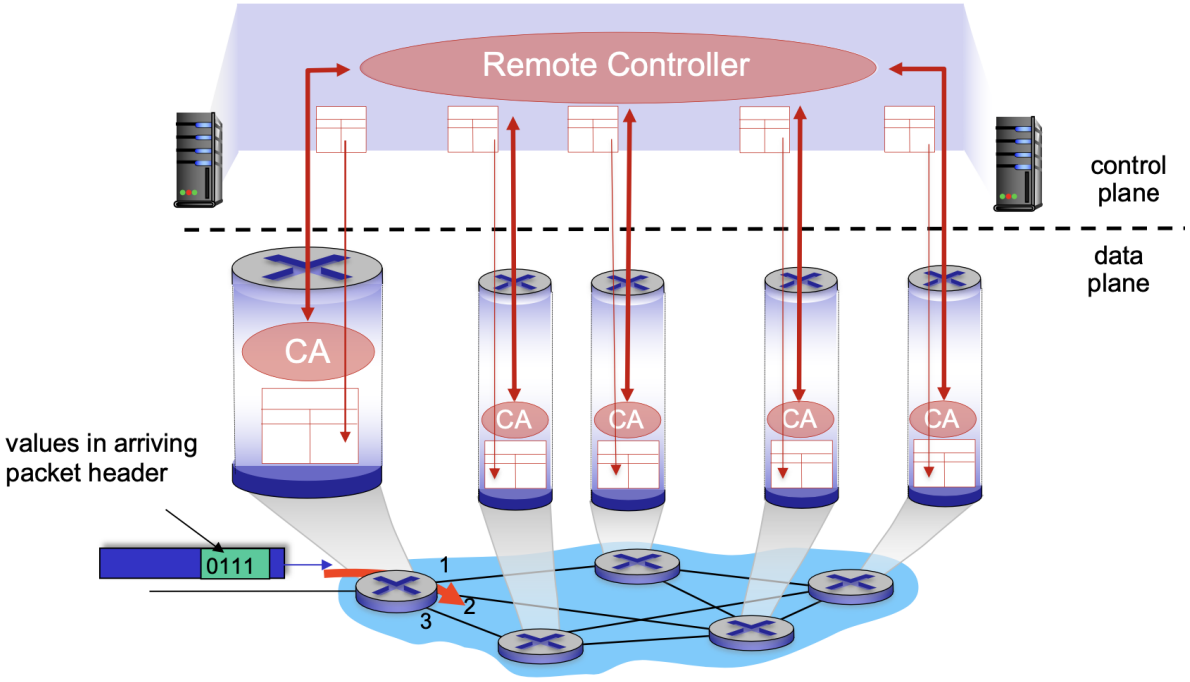


SDN 컨트롤러는

1. 정확한 상태정보(e.g., 원격 링크와 스위치, 호스트들의 상태)를 유지하고,
2. 이 정보를 네트워크 제어 애플리케이션들에 제공하며,
3. 애플리케이션들이 하부 네트워크 장치들을 모니터하고 프로그램하고 제어까지 할 수 있도록 수단을 제공한다.

그림에서의 컨트롤러는 단일 중앙 서버의 형태이지만, 실제로 컨트롤러는 **논리적으로만 중앙 집중 형태**다.

(일반적으로는 협업 능력과 확장성, 높은 이용성을 갖도록 몇 개의 서버에 구현)



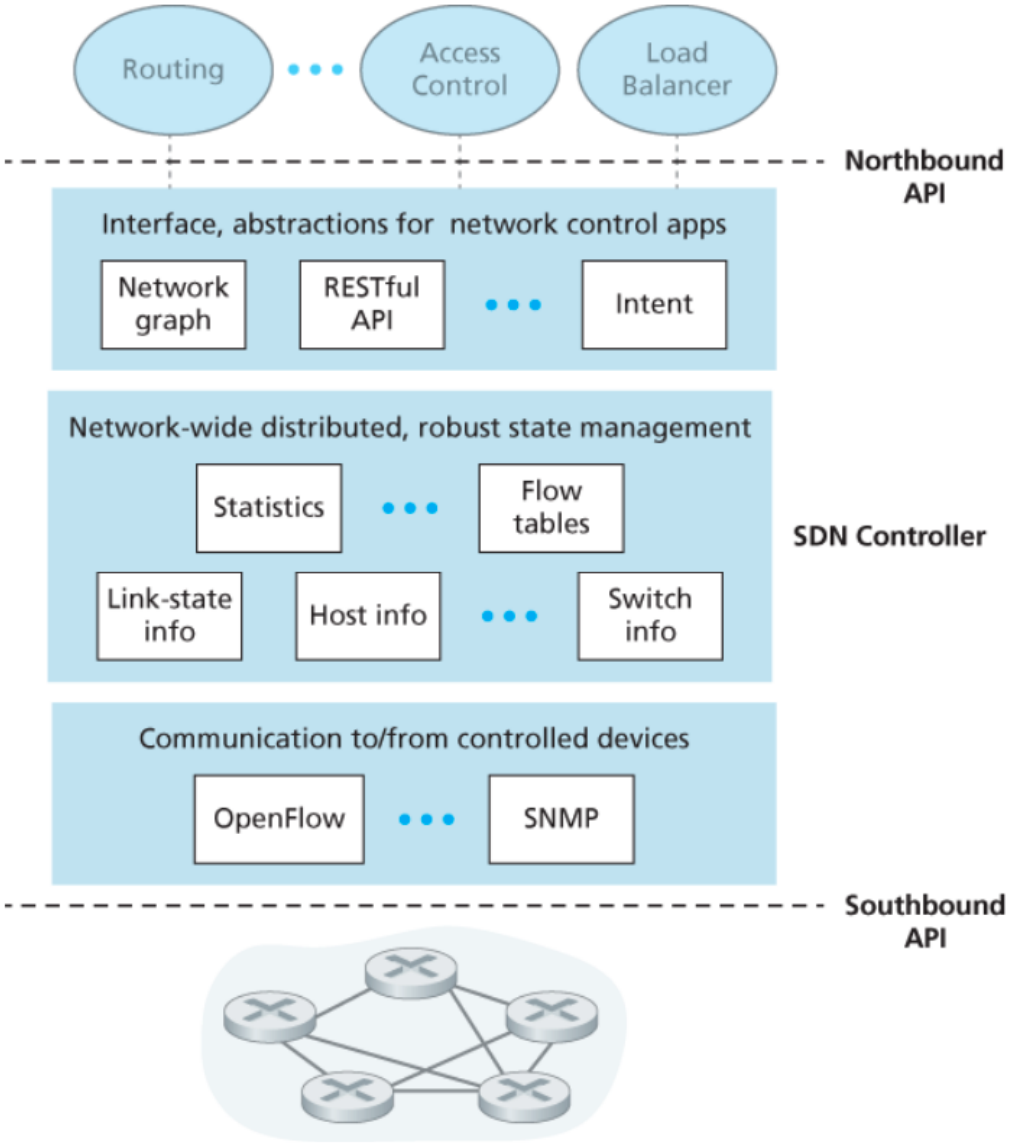
5.5.1 SDN 제어 평면

: SDN 컨트롤러와 SDN 네트워크 제어 애플리케이션

SDN 컨트롤러

컨트롤러의 기능은 크게 3개의 계층으로 구성된다.

1. 네트워크 제어 애플리케이션 계층과의 인터페이스
2. 네트워크 전역 상태 관리 계층
3. 통신 계층



통신 계층: SDN 컨트롤러와 제어받는 네트워크 장치들 사이의 통신

> 💡 제어받는 장치들과의 통신

- SDN 컨트롤러가 원격의 SDN 기능이 가능한 장치들의 동작을 제어하려면 컨트롤러와 그 장치들 사이에 정보를 전달하는 프로토콜이 필요하다.
- 장치는 주변에서 관찰한 이벤트를 컨트롤러에 알려, 네트워크 상태에 대한 최신의 정보를 제공해야 한다.

컨트롤러와 제어받는 장치들 간의 통신은 '사우스바운드(southbound)'라고 알려진 **컨트롤러 인터페이스**를 넘나든다.

이 통신 기능을 제공하는 구체적 프로토콜은 OpenFlow이며, 이는 모두는 아니지만 대부분의 SDN 컨트롤러에 구현되어 있다.

네트워크 전역 상태 관리 계층

> 💡 네트워크 전역에 분산되고 견고한 상태 관리

SDN 제어 평면의 궁극적인 제어 결정을 위해서는

컨트롤러가 네트워크 호스트와 링크, 스위치, 그리고 SDN으로 제어되는 다른 장치들에 대한 최신 정보를 알아야 한다.

제어 평면의 궁극적인 목적은 다양한 제어 장치들의 플로우 테이블을 결정하는 것이므로 컨트롤러도 이 테이블들의 복사본을 유지해야 할 것이다.

스위치의 플로우 테이블이 가지는 카운터들과 같은 이러한 정보 조각들은 모두 SDN 컨트롤러가 유지하는 **네트워크 전역 '상태'**의 예들이다.

네트워크 제어 애플리케이션 계층과의 인터페이스

> 💡 네트워크 제어 애플리케이션들을 위한 인터페이스와 추상화

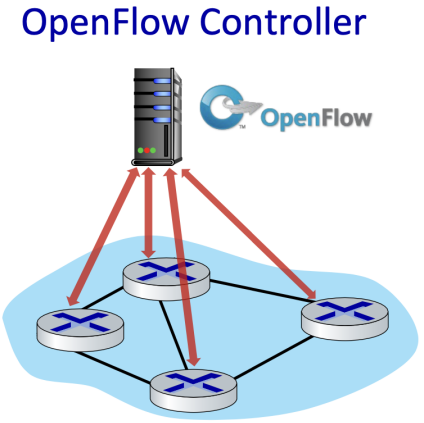
컨트롤러는 '노스바운드(northbound)' 인터페이스를 통해 네트워크 제어 애플리케이션과 상호작용한다.

이 API는 네트워크 제어 애플리케이션이 상태 관리 계층 내의 네트워크 상태 정보와 플로우 테이블을 읽고 쓸 수 있도록 해준다.

💡 SDN 컨트롤러는 외부에서 볼 때 '*논리적으로 중앙 집중된*', 잘 짜여진 하나의 서비스로 보일 수 있지만,
이 서비스들과 상태 정보를 보관하기 위한 데이터베이스는 장애 허용성(fault tolerance)과 높은 가용성, 또는 다른 성능상의 이유로 실제로는
분산된 서버의 집합에 구현된다.
근래의 컨트롤러는 논리적으로는 중앙 집중 형태이나 물리적으로는 분리된 컨트롤러 플랫폼 구조이다.
이런 구조는 제어되는 장치와 네트워크 제어 애플리케이션에게 늘어나는 장치 수에 따라 확장 가능한 서비스와 높은 가용성을 제공한다.

5.5.2 OpenFlow 프로토콜

- OpenFlow 프로토콜은 SDN 컨트롤러와 SDN으로 제어되는 스위치 또는 OpenFlow API를 구현하는 다른 장치와의 사이에서 동작한다.
- OpenFlow 프로토콜은 TCP상에서 디폴트 포트 번호 6653을 가지고 동작한다.



SDN으로 제어되는 **스위치에서 컨트롤러로 전달되는** 주요 메시지는 다음과 같다.

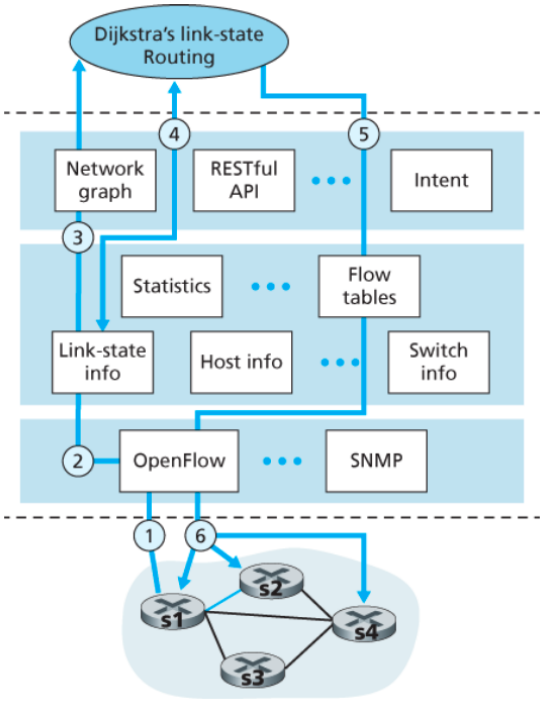
- 플로우 제거 : 이 메시지는 컨트롤러에게 어떤 플로우 테이블 엔트리가 시간이 만료되었거나 **상태 수정** 메시지를 수신한 결과로 삭제되었음을 알린다.
- 포트 상태 : 이 메시지는 스위치가 컨트롤러에게 포트의 상태 변화를 알리기 위해 사용된다.
- 패킷 전달
  - 4.4절에서 스위치 포트에 도착한 패킷 중에서 플로우 테이블의 어떤 엔트리와도 일치하지 않는 패킷은 처리를 위해 컨트롤러에게 전달된다고 했다.
  - 어떤 엔트리와 일치한 패킷 중에서도 일부는 그에 대한 작업을 수행하기 위해 컨트롤러에게 보내지기도 한다.이 메시지는 그러한 패킷을 컨트롤러에게 보내기 위해 사용한다.

5.5.3 데이터 평면과 제어 평면의 상호작용: 예제

아래 그림은 SDN의 제어를 받는 스위치와 SDN 컨트롤러 간의 상호작용에 대한 것이다.

- 여기서는 다익스트라 알고리즘이 최단 경로를 결정하기 위해 사용되는데,

  다익스트라 알고리즘은 **패킷 스위치 외부에서** 별도의 애플리케이션으로 수행된다.
- 패킷 스위치들이 **링크 갱신 정보를** 서로 간이 아닌 **SDN 컨트롤러에게 전송한다.**



1. **스위치 s2와의 링크 단절을 감지한 s1**은 OpenFlow의 포트 상태 메시지를 사용하여 링크 상태의 변화를 SDN 컨트롤러에게 알린다.
2. 링크 상태 변화를 알리는 OpenFlow 메시지를 받은 SDN 컨트롤러는 링크 상태 관리자에게 알리고, 링크 상태 관리자는 링크 상태 데이터베이스를 갱신한다.
3. 다익스트라 링크 상태 라우팅을 담당하는 네트워크 제어 애플리케이션은 링크 상태의 변화가 있을 경우 알려달라고 이전에 등록해두었다.

   이 애플리케이션이 링크 상태의 변화에 대한 알림을 받게 된다.
4. 링크 상태 라우팅 애플리케이션이 링크 상태 관리자에게 요청하여 **갱신된 링크 상태**를 가져온다.
   - 이 작업은 상태 관리 계층에 있는 다른 구성 요소의 도움이 필요할 수도 있다.
   - **그 후 새로운 최소 비용 경로를 계산한다.**
5. 링크 상태 라우팅 애플리케이션은 갱신되어야 할 플로우 테이블을 결정하는 플로우 테이블 관리자와 접촉한다.
6. 플로우 테이블 관리자는 OpenFlow 프로토콜을 사용하여 **링크 상태 변화에 영향을 받는 스위치들의 플로우 테이블을 갱신한다.**
   - 이 예에서는 s1, s2, s4가 이에 해당한다.
   - s1 : 이제부터 s2를 목적지로 하는 패킷을 s4로 보낸다.
   - s2 : 이제부터 s1로부터의 패킷을 중간 스위치 s4를 통해 받는다.
   - s4 : s1에서 s2로 가는 패킷을 전달해야 한다.

> 💡 컨트롤러가 플로우 테이블을 마음대로 변경할 수 있기 때문에 단순히 애플리케이션 제어 소프트웨어를 바꿈으로써 원하는 어떤 형태의 포워딩 방식도 구현할 수 있다.

5.6 인터넷 제어 메시지 프로토콜(ICMP)

- 인터넷 제어 메시지 프로토콜(Internet Control Message Protocol, ICMP)은 호스트와 라우터가 서로 간에 네트워크 계층 정보를 주고받기 위해 사용된다.
- ICMP는 종종 IP의 한 부분으로 간주되지만, ICMP 메시지가 IP 데이터그램에 담겨 전송되므로 **구조적으로는 IP 바로 위에 있다.** 즉, ICMP 메시지도 **IP 페이로드로 전송되며,**
- 호스트가 상위 계층 프로토콜이 ICMP라고 표시된(상위 계층 프로토콜 번호가 1번인) IP 데이터그램을 받으면 ICMP로 내용을 역다중화한다.
- ICMP 메시지는 타입(type)과 코드(code) 필드가 있고, ICMP 메시지의 발생 원인이 된 IP 데이터그램의 헤더와 첫 8바이트를 갖는다.
- 이는 송신자가 오류를 발생시킨 패킷을 알 수 있도록 하기 위해서이다.

Traceroute 프로그램

아래의 방식으로 출발지 호스트는 자신과 목적지 호스트 사이에 있는 라우터들의 수와 정체, 그리고 두 호스트 간의 왕복 시간을 알게 된다.

1. **출발지와 목적지 사이의 라우터 이름과 주소를 알아내기 위해**

   출발지의 Traceroute는 일련의 `IP 데이터그램`을 목적지에 보낸다.

   - 각각의 데이터그램은 UDP 포트 번호를 가진 **UDP 세그먼트**를 운반한다.
   - TTL 값은 첫 번째 데이터그램이 1, 두 번째는 2, 세 번째는 3, 이런 식이다.

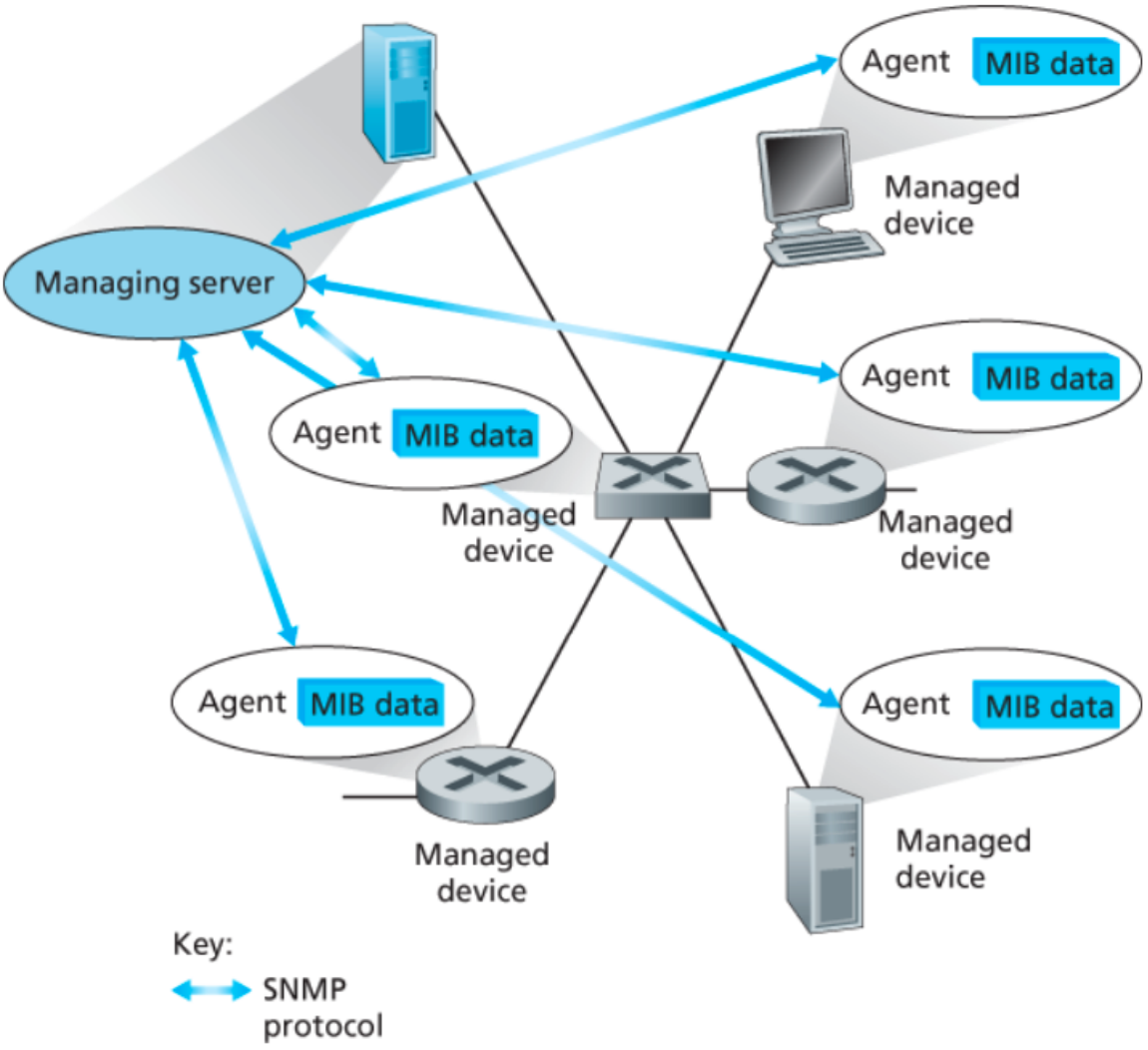
2. 출발지는 각 데이터그램에 대해 타이머를 작동시킨다.
   - n번째 데이터그램이 n번째 라우터에 도착하면 해당 라우터는 데이터그램의 TTL이 방금 만료되었음을 알게 된다.
3. IP 프로토콜 규칙에 따라 라우터는 데이터그램을 폐기하고 `ICMP 경고 메시지(타입 11, 코드 0)`를 출발지에 보낸다.
   - 이 경고 메시지는 라우터의 이름과 IP 주소를 포함한다.

4. 이 ICMP 메시지가 출발지에 도착하면, 출발지는
   (1) 타이머로부터 왕복 시간(round-trip time, RTT),
   (2) ICMP 메시지로부터 n번째 라우터의 주소와 이름을 획득한다.

5.7 네트워크 관리와 SNMP, NETCONF/YANG

네트워크 관리란 무엇인가?

> 네트워크 관리는 적정한 비용으로 실시간, 운용 성능, 서비스 품질 등의 요구사항을 만족시키기 위해 네트워크와 구성요소 자원을 감시, 테스트, 폴링, 설정, 분석, 평가, 제어하는 하드웨어, 소프트웨어, 인간 요소 등을 배치하고, 통합, 조정하는 것이다.



MIB(Management information Base)

SNMP

> SNMPv3(Simple Network Management Protocol version 3)는 관리 서버와 그 관리 서버를 대표하여 실행되고 있는 에이전트 사이에서 네트워크 관리 제어 및 정보 메시지를 전달하기 위해 사용된다.

1. SNMP 관리 서버는 에이전트에게 요청을 송신하고
    - 일반적으로 요청은 피관리 장치과 관련된 MIB 객체 값들을 질의(검색) 또는 수정(설정)하기 위해 이용
2. 이를 받은 SNMP 에이전트는 이를 수행한 후 요청에 대한 응답을 보낸다.

마무리

결국 알아서 찾아봐라...;;