Week 7.

Efficient use of IP Address and Routing

차례

- DHCP
- NAT
- IPv6
- Introduction to Routing
- Link-state Routing
- Distance Vector Routing

DHCP (Dynamic Host Configuration Protocol)

- 기존 IPv4로는 IP주소가 부족하다.
  - 근데 모든 사람이 동시에 컴퓨터를 켜고 있는게 아니기 때문에 동접자 수만 알아서 IP주소를 할당한다면 기존 IPv4로 충분하다.
- 동시에 접속하는 사람의 수 만큼만 IP주소를 동적으로 할당하자! → DHCP

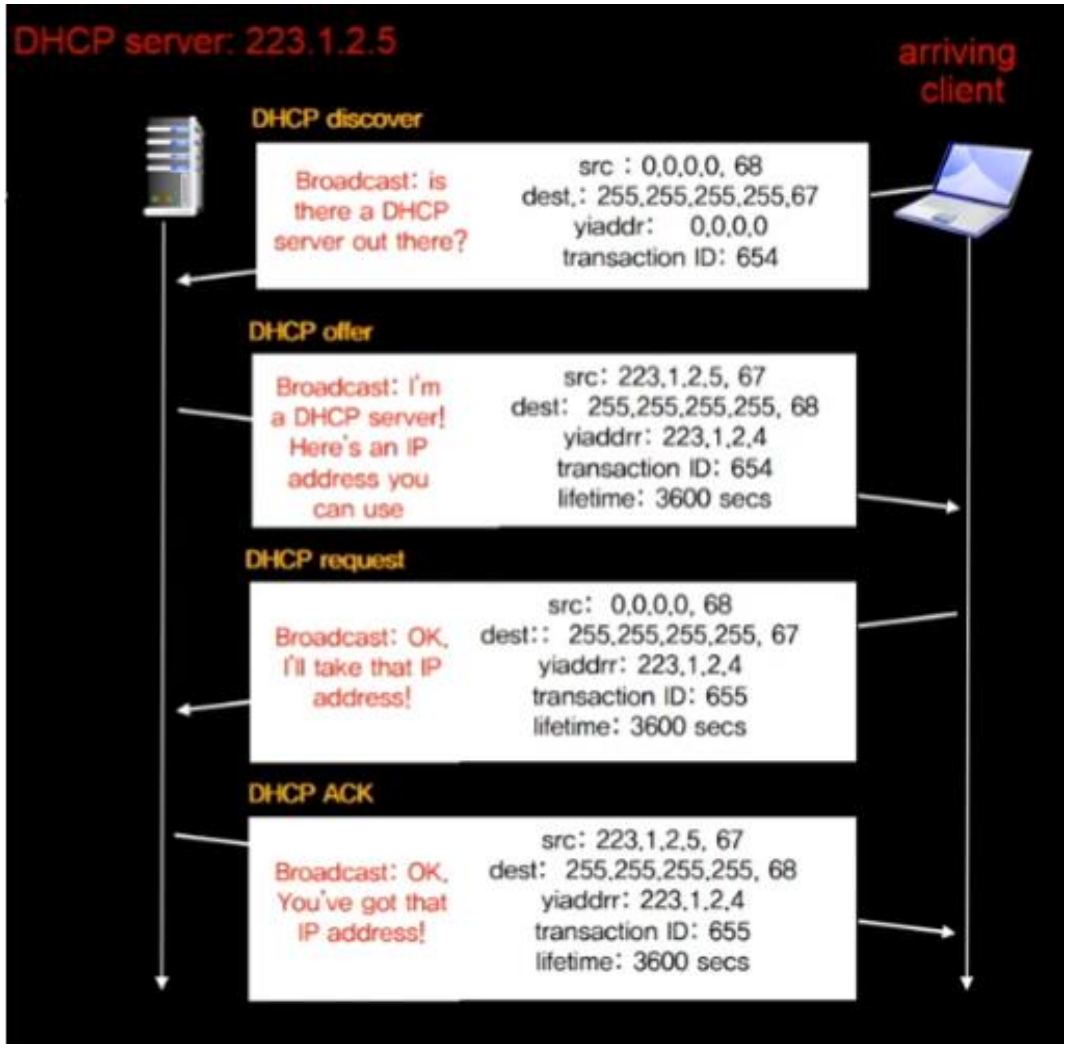
- DHCP는 전송 프로토콜로 UDP를 사용하게 됨.
- DHCP프로토콜은 IP주소와 라우터의 주소, 네트워크 내 사용하는 DNS 서버의 주소를 같이 줌 → Plug&Play 가능.
- 2011년에 IPv4 주소는 고갈 되었음. 그러나 DHCP를 통해 적은 수 의 IP주소를 가지고도 많은 수의 컴퓨터를 지원할 수 있게 됨.
- 일반 네트워크 지식이 없는 사람도 쉽게 인터넷을 사용할 수 있게 해준 것이 DHCP의 가장 큰 역할

DHCP (Dynamic Host Configuration Protocol)

- 우측 그림은 DHCP의 과정을 나타낸다.
  1. 클라이언트가 서버에게 DHCP 주소가 있는지 DHCP discover 메시지를 보내면
  2. DHCP 서버에서 IP주소가 담긴 DHCP offer를 보낸다.
  3. 클라이언트가 해당 IP주소를 사용하겠다는 DHCP request를 보내면
  4. DHCP서버에서 이를 받아들이고 DHCP ACK 메시지를 보낸다.

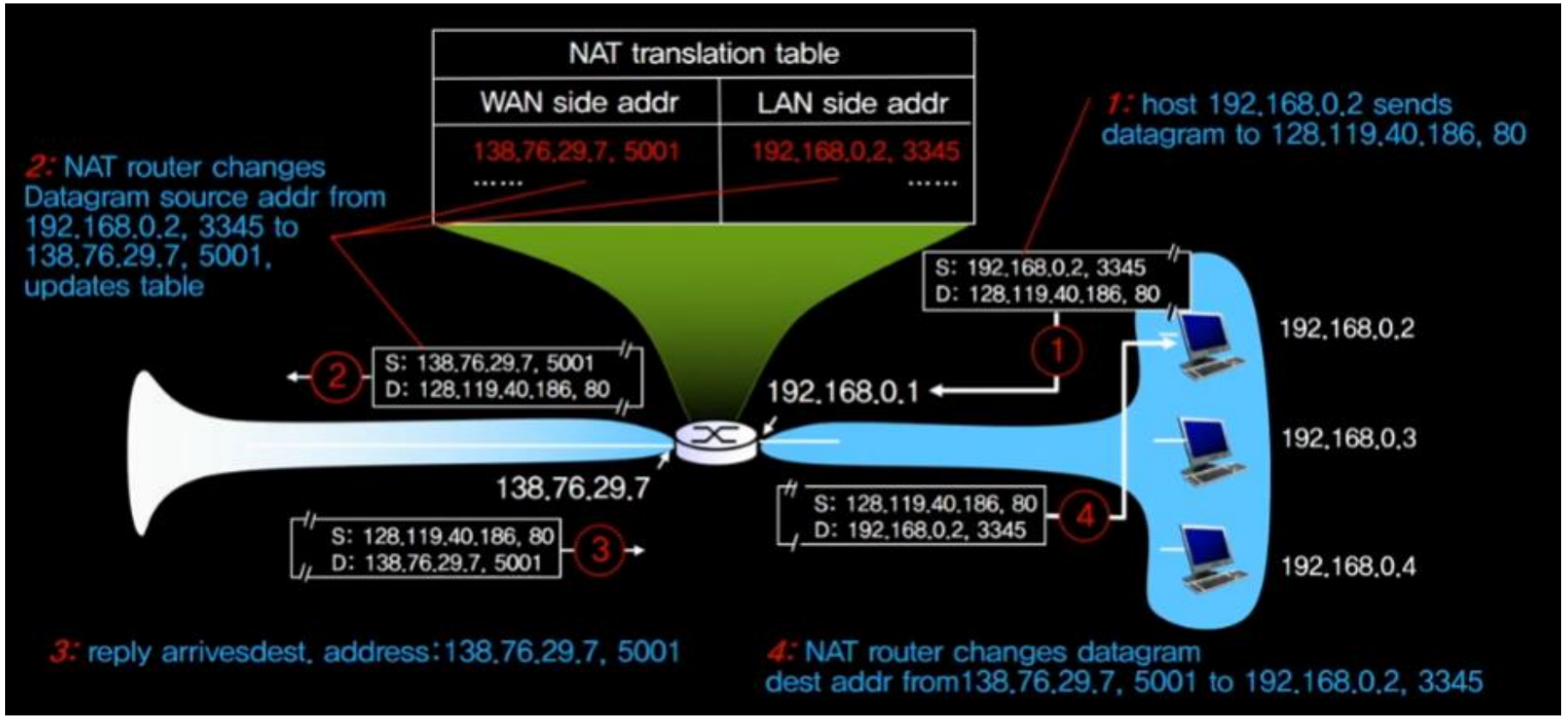
- 주소 할당이 목적이라면 왜 2번과정에서 끝나지 않는가?

  → 어디에 있을 지 모르는 다른 서버나 호스트에게 해당 IP주소를 사용한다는 것을 알리기 위한 broadcast역할.



NAT (Network Address Translation)

- 한 가구당 컴퓨터 수가 늘어나고 스마트폰이 생기면서 한 가구임에도 별도의 IP 주소를 할당받고 요금을 두 배, 세 배로 내는 경우가 많았는데, 이것을 해결하고자 하는 것이 NAT 다.
- 주소를 하나만 받아서 내부에서 알아서 구별해서 쓰는 것이 NAT 제도



NAT (Network Address Translation)

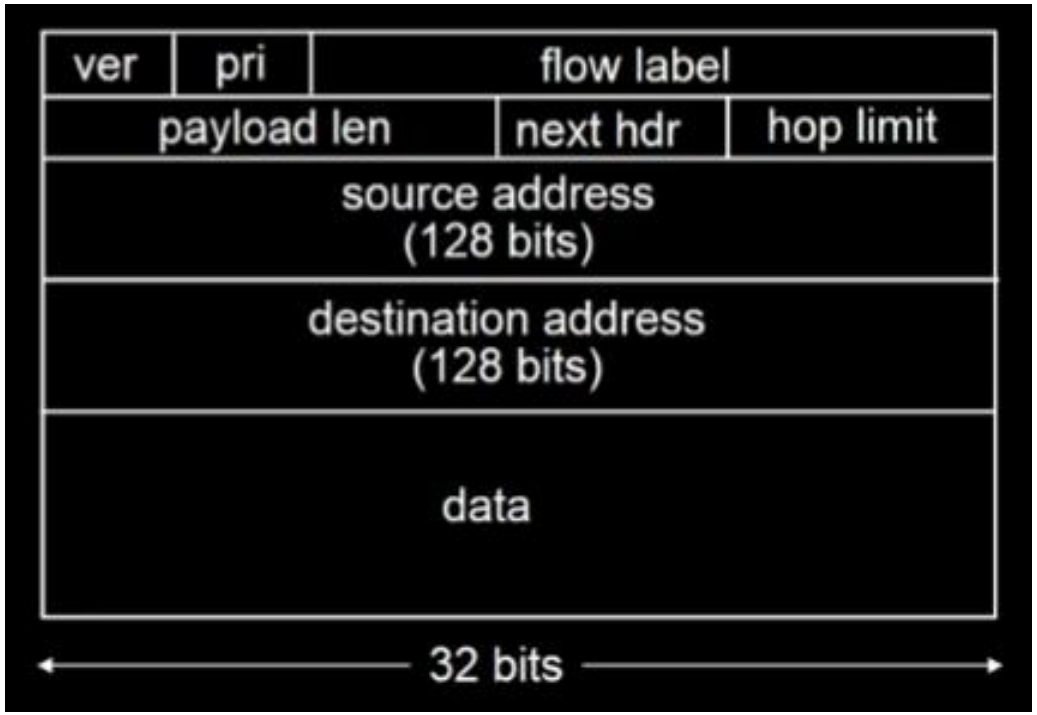
- NAT 제약

  - 어떤 사용자가 웹 호스팅 서버를 두고 싶다고 한다면?
    → 외부 주소는 해당 주소에 접속이 불가능하지만 이를 포트 포워딩으로 해결할 수 있다.

  - 공유기가 외부로부터 주소를 받아서 내부에 특정 포트로 포워딩을 해준다면 NAT problem을 비켜갈 수 있다.

IPv6

- DHCP, NAT 모두 근본적인 해결책이 아니다. IP주소의 개수를 아예 늘려버리자.
- 바꾸면서 Header도 바꿔버리자! (우측 그림)

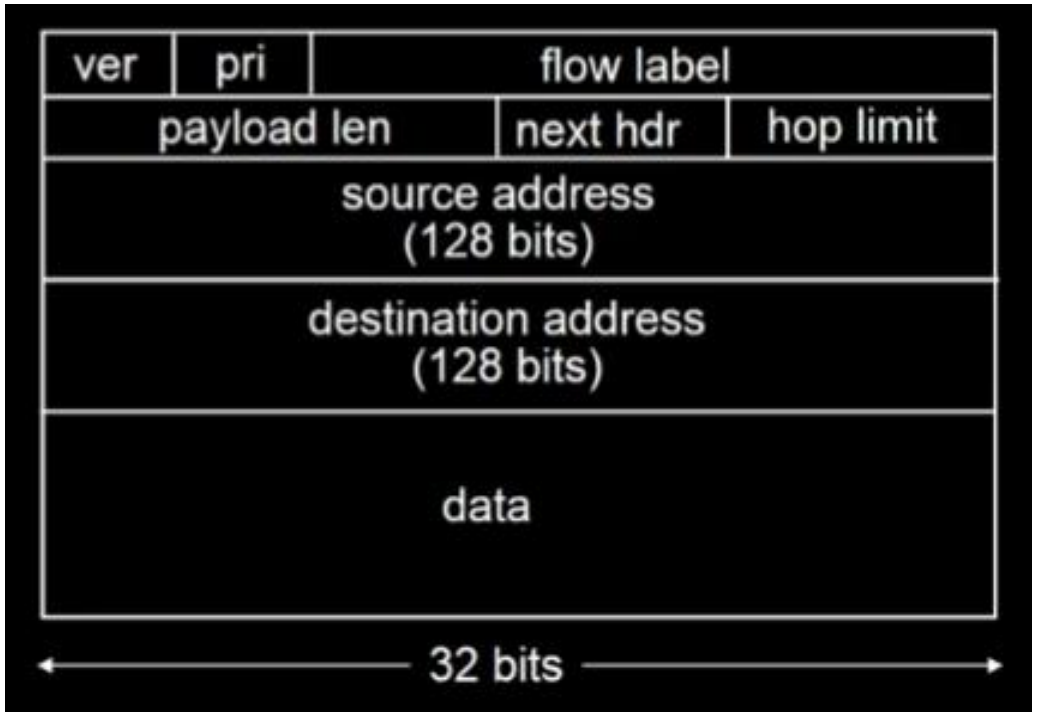


- Src 주소, dest 주소가 128bits(16byte)씩 총 32byte를 차지하고 있고, 8byte가 나머지.
- IPv4는 src, dest 주소가 4byte 씩 총 8byte를 차지하고 있고, 12byte가 나머지.

- IPv6으로 오면서 header에 fragmentation & reassembly가 없어졌다. 전 세계의 네트워크가 많이 균일화 되었기 때문에.
- 그럼에도 fragmentation이 필요하다면? 다시 src에게 알려줘서 애초에 쪼개서 전송하자. (추후 ICMP에서 설명)

IPv6

- Ver : 버전 (4또는 6)
- Pri : priority 우선도가 높은 애들은 buffer overflow가 나도 keep을 한다든지 등등...
- Flow label : IPv4에서는 없던 개념.
  모든 데이터 그램이 개별 취급되던 IPv4와 달리, IPv6에서는 일련된 데이터 그램을 한 덩어리의 flow로 취급할 수 있게 생각함.
- Payload len : payload의 길이
- Next hdr : 상위 레이어에 어떤 프로토콜이 적용되는지, TCP, UDP 등등...
- Hop limit : TTL과 같은 개념.



IPv6

- Checksum이 없어도 되는건가요?

  - Flow label이 생기면서 checksum이 없어짐. → 네트워크 전송 속도가 빨라짐.
  - 과거에는 중간에 구리선 같은 케이블을 쓰면서 에러가 많이 발생하기 때문에 checksum이 필요했다면 지금은 그렇지 않음.
  - 그래서 현재는 에러가 발생한다면 재전송하는 것이 checksum을 살리는 것보다 효율이 좋음. 중간 라우터에 대한 부담도 적음.

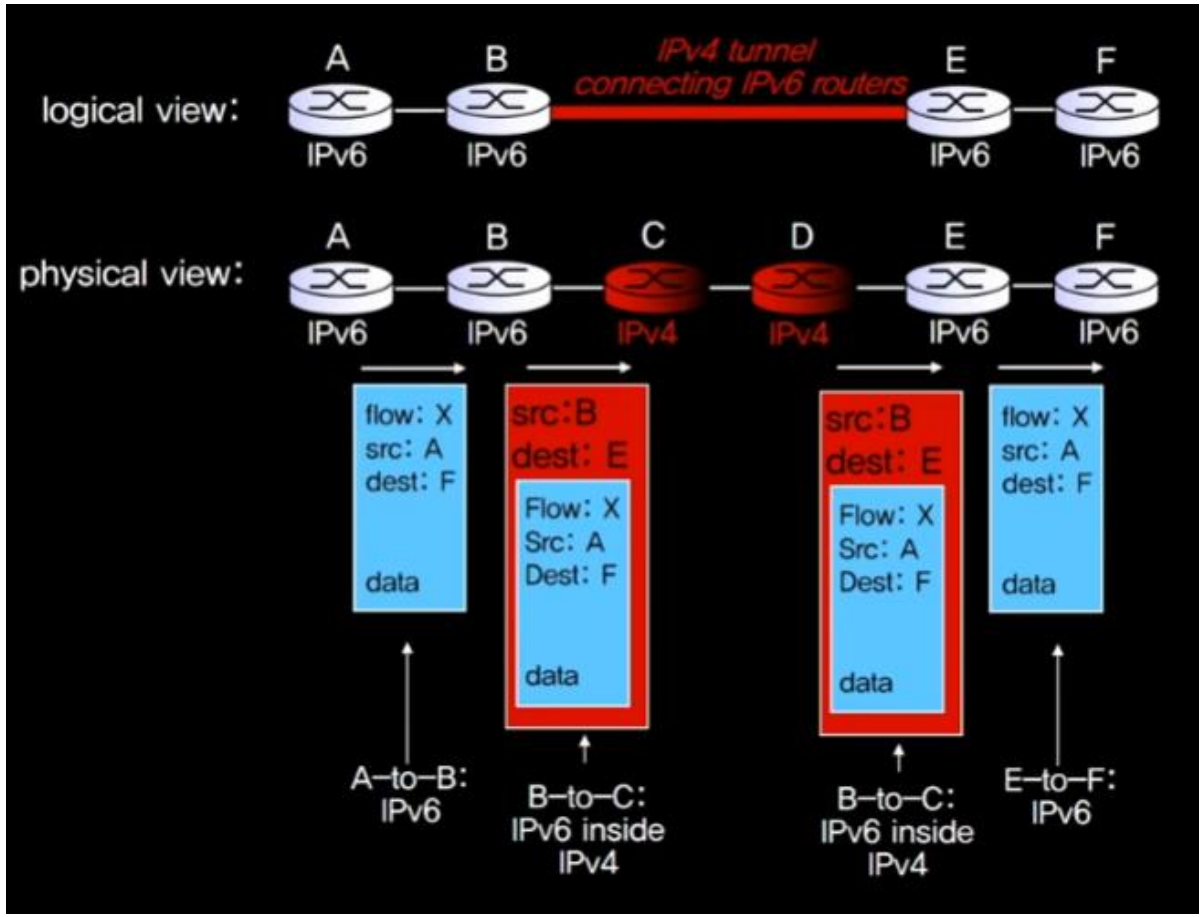
IPv6

- Tunneling

- IPv6에서 IPv4로 들어갈 때, IPv4 헤더를 붙여서 보낸다.

- 그래서 C, D라우터가 보기에는 IPv4 데이터 그램인 것처럼 보이게 된다. IPv6을 사용하는 라우터 사이에 IPv4 터널을 지난다는 의미로 터널링이라고 한다.

- DHCP, NAT 등의 기술이 잘 버텨준 덕에 아직 IPv6의 사용이 많지는 않다.



Introduction to Routing

- 포워딩 테이블을 만드는데 있어 기본적인 기능을 하는 것이 라우팅 알고리즘.

- src~dest까지의 길을 설정해주는 라우팅 알고리즘에 따라 포워딩 테이블이 만들어지고 포워딩 기능을 수행하는 것이 라우터의 기능.

- 기존네트워크에서는 모든 라우터가 이 기능을 수행했다면, SDN에서는 중앙 집중형으로 중앙서버가 한다든지 이런 식으로 진행

- 이런 라우팅을 그래프로 주로 표현하는데, 정점과 간선으로 이루어져 있고, src부터 dest 까지의 경로 비용의 합은 적을수록 좋다.
  → 라우팅 알고리즘이 최소비용의 경로를 찾는다.

Introduction to Routing

● 라우팅 알고리즘의 종류

■ Static vs Dynamic

- **Static** : 각 path, route를 관리자가 결정해줌. 중간에 있는 라우터들을 수동으로 설정해줌.
  중간에 네트워크 상황이 바뀌면 역시 관리자가 라우터에 접속해서 값들을 수동으로 바꿔줌.

- **Dynamic** : 라우터들끼리 정보를 주고 받은 다음, 거기에 따라서 길을 설정.
  같은 src, 같은 dest일지라도 네트워크 상황에 따라서 route, path가 변할 수 있음.

■ Global vs Decentralized

- **Global** : 전체 네트워크 topology를 완벽하게 알고 있고, 각 네트워크에 있는 모든 link cost를 알고 있는 상태에서 경로를 결정하는 경우
  (=link state algorithm)

- **Decentralized** : 전체 topology를 아는 것이 아니라 자기와 직접 연결 되어 있는 이웃들만 알고,
  다른 node들에 대해서는 '그 node까지의 총 비용' 과 '그 비용으로 보내기 위해 어떤 node를 다음 node로 골라야 하는지' 정도만 안다.
  (=distance vector)

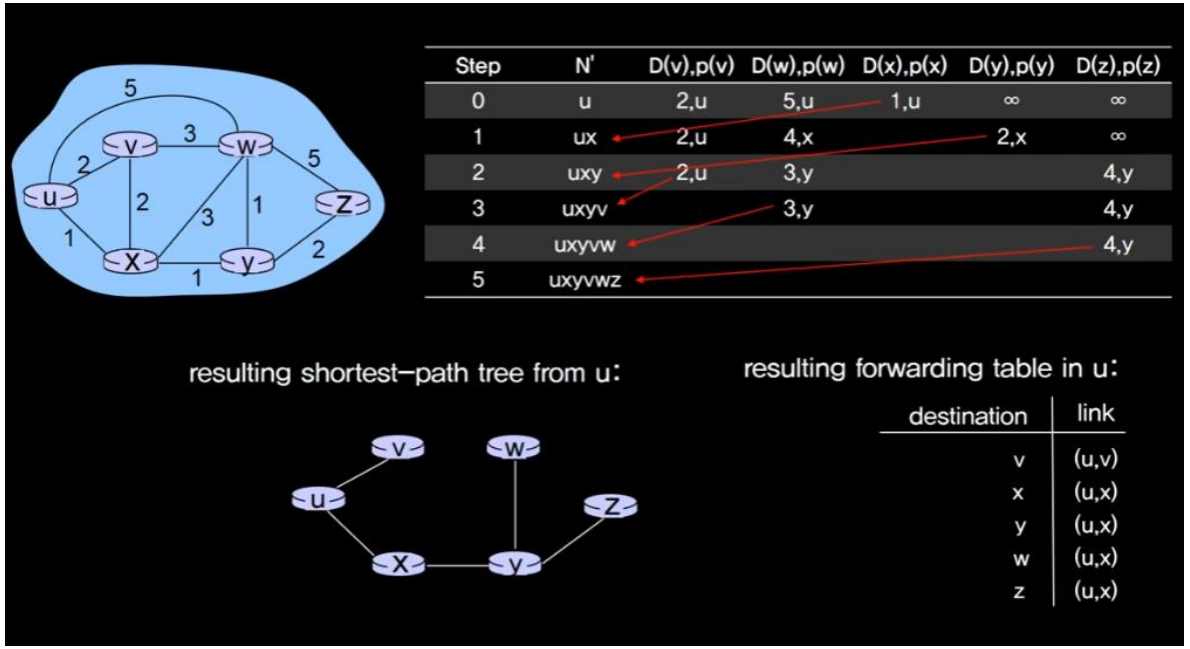
Link-state Routing

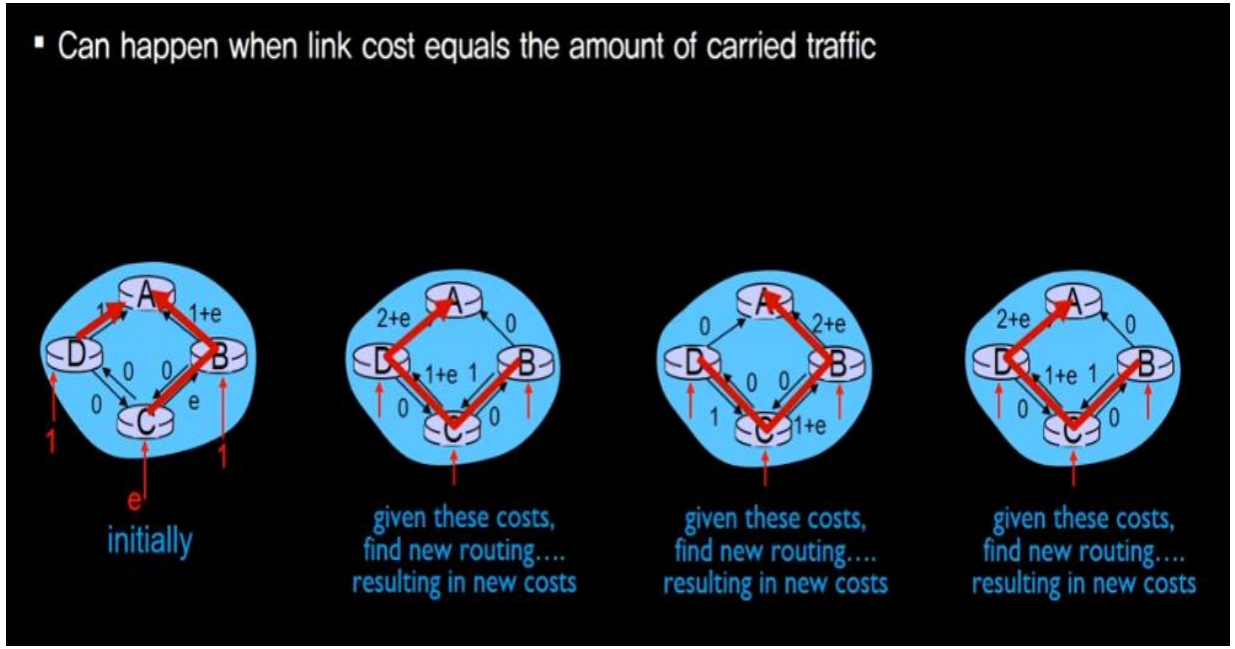
- 전체 topology 정보를 알고 그 상황에서 길을 설정하는 것.
- 라우터들끼리 알고 있는 정보를 다른 라우터와 교환해야 함.
- 이 때 쓰이는 것이 LSA(Link-State Advertisement) 메시지 혹은 LSP (Link-State Packet) 이라고 한다.

- LSA 구성요소
  - Neighbor node information : 나와 연결되어 있는 node 들에 대한 정보
    - 라우터가 변경되거나 추가 될 때
    - 링크 활성화/비활성화
    - 네트워크 상태 정보가 바뀌었을 때
    - 그 외 기타 등등 주기적으로 생성된다

Link-state Routing

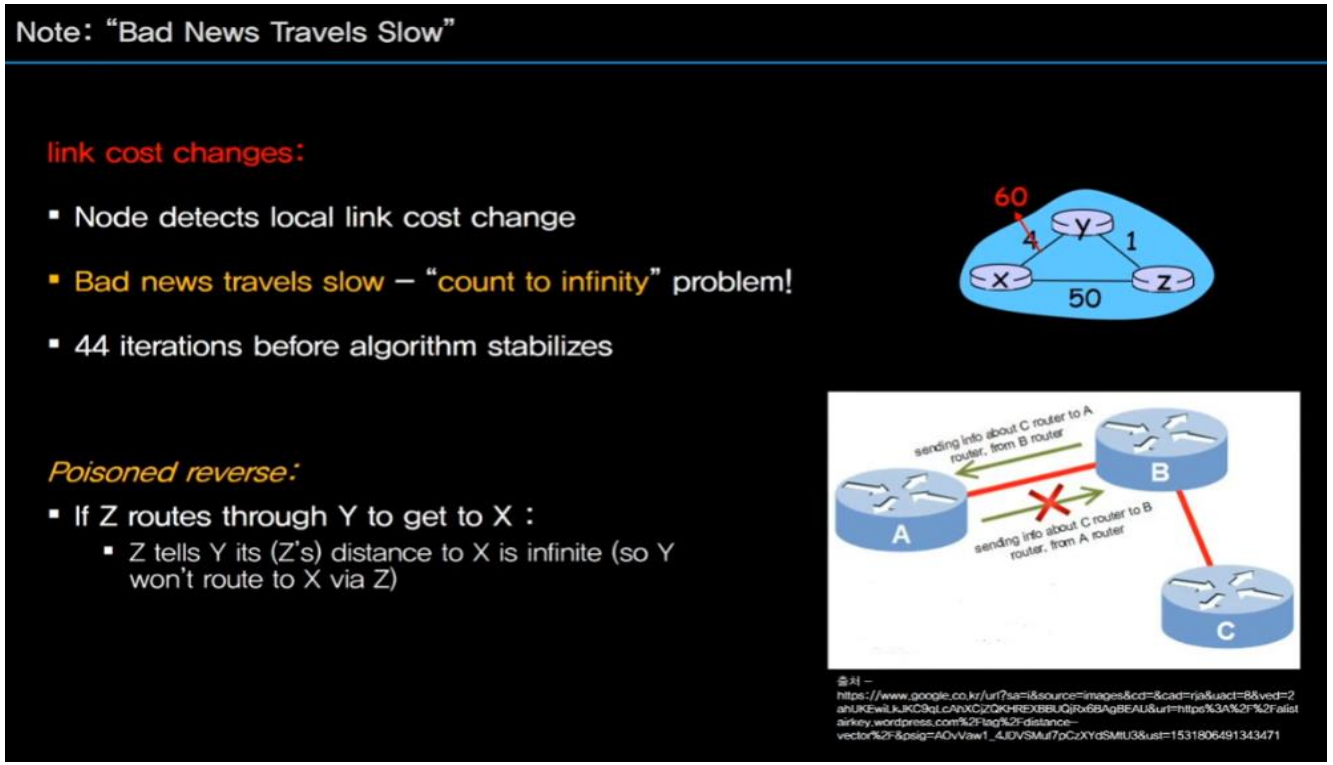
- 만들어진 LSA메시지는 해당 네트워크의 모든 라우터에게 전송이 되고, 그 LSA 메시지를 다 모은 뒤에 각각의 라우터들은 네트워크의 전체 topology를 구성함.
- 그 후 최소 비용 경로 계산 (Dijkstra). 라우터 개수만큼의 반복이 필요한 알고리즘.
- Dijkstra 알고리즘을 사용할 때 Oscillation problem 이 발생할 수 있다.
- Link cost를 실시간으로 반영하는 경우에 오른쪽 아래 그림과 같은 상황이 발생할 수 있다.





Distance Vector Routing

- Bellman-ford 알고리즘 기반.
- Link-state algorithm은 advertisement message를 네트워크 전체로 브로드캐스팅.
- Distance vector algorithm은 자신의 이웃들하고만 메시지를 교환하고, 이웃이 나한테 전달 해준 정보가 바뀌었다면 그것을 기반으로 자신의 distance vector를 갱신함.
- 장점 : good news는 빨리 전달됨.
- 단점 : bad news가 천천히 전달됨.
- 이를 해결하고자 poisoned reverse가 도입됨.
  그러나 완벽한 해결책이 아님.



Link-state Routing vs Distance Vector Routing

Distance Vector	Link state
전체 라우팅 테이블을 보내야함.	바뀐 정보만 보내면 됨.
Slow convergence	Fast convergence
직접 연결된 이웃들하고만 교환	모든 라우터에게 다 전달.
파악 불가능	전체 네트워크 topology 파악 가능.
Count-to-infinity problem 발생할 수 있음.	Loop가 발생하지 않고 Dijkstra 알고리즘으로 최단 경로 찾음.
간단한 configuration	복잡한 configuration
RIP (Routing Information Protocol)	OSPF (Open Shortest Path First)

keyword

- 동적 호스트 설정 프로토콜(Dynamic Host Configuration Protocol, DHCP): 네트워크 서버가 네트워크에 접속한 호스트에게 자동적으로 IP 주소 및 구성 정보를 할당하고 관리하는 규칙

- 네트워크 주소 번역(Network Address Translation, NAT): 로컬 네트워크 내의 여러 개의 디바이스가 하나의 공식(public) IP로 외부통신을 하는 과정에서 로컬 사설(private) IP 주소와 공식 IP 주소를 매핑하는 방법

- IPv6 터널링(tunneling): IPv6의 데이터그램을 IPv4 체계의 라우터에서도 전달하기 위한 방법으로 해당 IPv6 데이터그램을 IPv4의 페이로드에 담아서 보냄

- 링크 상태 라우팅(link-state routing): 각 라우터가 상태 정보 메시지를 통해 네트워크 전체 토폴로지(topology)를 구성하고 이를 바탕으로 최단 경로의 포워딩 테이블을 만드는 방식

- 거리 벡터 라우팅(distance vector routing): 각 라우터가 주기적으로 이웃 라우터와 라우팅 정보를 주고받음으로써 다른 노드에 이르는 거리 비용을 갱신하는 방식

감사합니다