DiffServ 모델의 이해와 적용법

이번호에는 DiffServ 모델에 대한 개요와 분류자, 미터, 마커에 대해 이론적인 접근과 더불어 실무에서 QoS를 설정할 때 분류자와 미터가 어떤식으로 구현되는지 살펴본다. 향후 DiffServ 모델의 모듈별로 QoS 설정을 추가해 엔드 투 엔드 QoS를 완성해 나가는 방향으로 강좌를 진행할 것이다.

김화식
zion@onsetel.co.kr 온세통신 시설운영팀 CCIE
s12840@freechal.com NRC QVO 팀 마스터

난 이 도 上 中 下

기본적으로 인터넷은 네트워크 트래픽을 하나의 클래스(Class)로 취급해 동일한 정책을 적용하는 베스트 에포트(Best-effort) 방법을 채택하고 있다. 물론, 초기에 인터넷이 구현됐을 때는 네트워크 트래픽을 하나의 클래스로 처리해도 큰 문제가 되지 않았다. 하지만 현재의 인터넷 상황은 새로운 유형의 트래픽(화상전화, 방송, VoIP 등)이 생겨남에 따라 기존의 베스트 에포트 서비스만으로는 QoS(Quality of Service)를 제공하지 못하게 됐다.

이를 보완하기 위한 방법으로 다양한 QoS 프로토콜이나 표준이 만들어졌는데, 대표적인 QoS 관련 프로토콜과 표준에는 IntServ (integrated Service), DiffServ(Differentiated Service), RSVP, MPLS, IEEE 802.1p/Q 등이 있다. 이들 표준은 그 자체로서 QoS 제공 기능을 갖는 것이 아니며, 다양한 QoS 구현 기술과 이들을 적용하는 방식 혹은 정책들로 이뤄져 있다.

이번호에 소개할 DiffServ 모델은 QoS를 보장하기 위한 방법 중 하나로, IntServ 모델이 확장성이 취약한 것을 극복하기 위해 ISP(Internet Service Provider)들이 제안한 모델이다.

기존의 IntServ는 트래픽 흐름(flow)에 대해 미리 필요한 시간에, 필요한 만큼의 대역폭을 할당받는 방식으로, 그 흐름들에 대해 100% 보장된(guaranteed) 서비스와 컨트롤된 로드(controlled load) 서비스를 제공받았다. 이에 비해 DiffServ는 패킷들을 비슷한 성질을 가진 클래스로 구분해서 중요한 클래스에 대해서는 가중치를 둬 서비스를 제공하는 방법이다. DiffServ는 100% 보장된 서비스를 제공하지는 못하지만, 구현이 쉽고 확장성이 뛰어나기 때문에 현재 인터넷에서 가장 많이 채택하는 방식이다. 이제부터 DiffServ 모델에 대해 보다 자세히 알아보자.

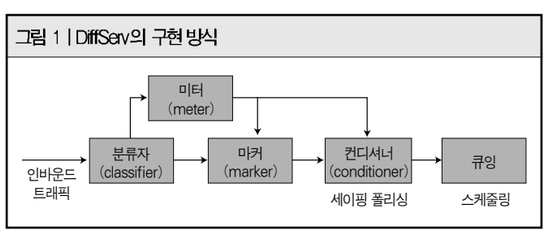
중요한 클래스에 가중치 두는 DiffServ

(그림 1)을 보면 DiffServ가 어떤 방법으로 구현되는지 알 수 있다. 먼저 분류자(Classifier)로 들어오는 트래픽을 다양한 기준에 따라 여러 개의 클래스로 구분하는데, 여기서는 도착한 패킷들이 어떤 클래스 큐에 저장될지 결정한다.

좀 더 구체적으로 이야기를 하면, 입력된 트래픽은 다양한 분류기준(extended ACL)에 의해 플로우 단위로 구분되며, 이와 동시에 플로우가 속하게 될 트래픽 클래스가 결정된다.

트래픽 클래스가 결정된다는 것은, 입력된 트래픽이 실제로 저장될 큐와 서비스(스케줄링)되는 방식이 결정된다는 것을 의미한다. 분류자를 통과한 패킷들은 각 트래픽 플로우에 할당된 미터(meter)에 의해 특성을 측정받는다. 측정된 결과는 사전에 약속한 QoS 트래픽 특성과 비교되며, 그 결과에 따라 마커(marker)에 의해 몇 가지 우선순위로 마킹(marking) 된다.

그림 1 | DiffServ의 구현 방식



마킹된 패킷들은 컨디셔너(conditioner)를 거치면서 사전에 약속된 트래픽의 대역폭 특성에 맞도록 조절된다. 컨디셔너는 지연(delay)을 이용해 대역폭을 조절하는 세이핑(Shaping)과 드로퍼(dropper)를 이용해 대역폭을 조절하는 폴리싱(policing)으로 구성돼 있다.

트래픽 컨디셔너는 경우에 따라서 흐름제어(flow control)도 처리할 수 있다. 컨디셔너를 통과한 패킷들은 큐잉(queuing)을 거치며 분류자에서 결정된 자신의 클래스에 맞는 큐에 저장된다. 큐에 저장된 패킷들은 스케줄링 과정을 통해 출력 링크로 보내진다.

분류자의 중요한 역할

분류자는 DiffServ 모델의 처음 과정으로, 들어오는 패킷을 일정한 기준에 따라 여러 개의 클래스로 구분하는 기능을 한다. 하지만 여러 종류로 나눠 복잡하게 처리하려는 의도가 아니라, 여러 종류의 패킷을 제한된 몇 개의 클래스로 분류한다는 개념으로 이해해야 한다.

분류자의 목적은 동일하거나 유사한 특성을 갖는 패킷들을 함께 처리함으로써 QoS의 구조를 단순화하자는데 있다. 즉, 다양한 QoS 특성을 갖는 트래픽들을 각각 처리한다면, 수없이 많은 패킷 처리 기준이 있어야 된다. 이것은 현실적으로 불가능하며, 가능하다 하더라도 노드에 커다란 부하를 주는 원인이 된다. 따라서 제한된 클래스로 패킷을 나눠 처리하는 것이다.

실무에서는 적게는 2개에서, 많게는 8개까지의 클래스를 사용하고 있으며, 가장 일반적인 경우가 4개의 클래스를 사용하는 것이다. 큐잉에 관한 연구를 살펴보면, 큐나 클래스의 개수가 1에서 2로 증가할 때 복잡성에 비해 성능 향상 효과가 가장 크다. 2개에서 4개로 증가할 때도 만족할만한 수준이지만, 4개에서 8개 또는 그 이상으로 증가할 때는 복잡성은 배로 증가하지만 성능의 향상효과는 수% 이내로 증가한다. 때문에 클래스의 수를 무작정 늘리는 것이 좋은 것만은 아니다. 따라서 이번 연재에서도 3개 내지 4개의 클래스를 주로 사용할 것이다.

앞에서도 언급했지만 분류자는 DiffServ 모델에서 가장 중요한 부분이다. 그것은 분류자에서 패킷을 구분해주기 때문이다. 뒤에서 여러 QoS 구현 방법들에 대해 설명을 하겠지만, 그런 모든 방법들도 분류자에서 패킷을 클래스로 나눠줘야지 비로소 적용할 수 있다. 실제 업무에서는 분류자를 구현하기 위해서 ACL(Access-Control Lists)을 많이 사용한다.

예를 들어 ACL 101에 해당하는 패킷은 첫번째 큐(Queue)로 보내고, ACL 102에 해당하는 패킷은 두번째 큐로 보내고자 할 때, 큐잉 작업이 이뤄지기 위해서는 분류자 작업이 선행돼야 한다. 트래픽 세이핑이나 폴리싱도 마찬가지이다.

(표 1)은 확장된 액세스 리스트(extended ACL)를 이용해 분류할 수 있는 필드 값을 나타낸 것이다. 확장된 액세스 리스트를 사용하면 (표 1)의 필드값을 하나 또는 그 이상을 조합해서 세부적인 분류도 가능하다. 클래스를 나누는데 쓰이는 또 다른 방법은 PBR(Policy-Based Routing)을 이용하는 것이다. 이 방법은 ACL을 이용하는 것보다 좀 더 유연한 명령으로, 'match'와 'set'을 이용해 분류자 적용 후, 마킹 모듈까지 적용할 수 있는 명령어다. 이는 (그림 2)와 같은 방식으로 구현될 수 있다.

표 1 | IP확장된(extended) ACL을 이용해 분류할 수 있는 항목들

필드	설명	
	이름	값
소스 IP 어드레스	소스 IP 어드레스의 범위와 와일드카드 마스크를 이용해 분류한다.	
목적지(destination) IP 어드레스	목적지 IP 어드레스의 범위와 와일드카드 마스크를 이용해 분류한다.	
IP 우선권(Precedence)	IP 우선권의 이름과 값을 이용해 분류한다	
	routine	0
	priority	1
	immediate	2
	flash	3
	flash-override	4
	Critic	5
	internet	6
	network	7
IP DSCP	DSCP(Differentiated Service Code Point)의 이름이나 값을 이용해 분류한다.	
IP ToS	ToS(Type of Service) 필드의 값을 이용해 분류한다	
	normal	0000
	max-relability	1000
	max-throughput	0100
	min-delay	0010
	min-monetary-cost	0001
TCP 포트	소스와 목적지 포트를 이용해 분류한다	
UDP 포트	소스와 목적지 포트를 이용해 분류한다	
ICMP	ICMP 메시지와 코드타입을 이용해 분류한다	
IGMP	IGMP 메시지 타입을 이용해 분류한다	

용 어 설 명

DiffServ
Differential Services의 줄임말. DiffServ는 IETF의 인터서브 작업 그룹이 개발한 모델로, QoS 및 CoS를 보장하기 위한 방법 중 하나다. DiffServ는 사용자와 서비스 공급업체를 구분하는 경계선과 서비스 공급업체들 간의 경계선에서 트래픽을 분류하기 위해 TOS(Type of Service) 필드를 사용하는 방법에 대해 제시하고 있다. 차별화된 서비스 모델은 TOS 필드를 사용한 부가적 서비스를 에서하고 있다. ▲하나는 MIT의 수석 연구원인 데이비드 클락이 정의한 베스트 에포트 서비스보다는 더 나은 보장된 서비스(Assured Service)이며 ▲다른 하나는 로런스 버클리 랩의 연구원인 반 제이콥슨이 정의한 프리미엄 서비스다. 이 두 서비스는 모두 가중치가 부여된 RED(Random Early Detection) 기능의 폐기 우선권(Drop Precedence)과 큐나 VC(Virtual Circuit 또는 Virtual Channel) 선택에 있어서의 지터(Jitter) 우선권의 두 가지 개념을 토대로 하고 있다. DiffServ가 우선 순위를 갖는 패킷을 먼저 전송한다는 점에서 802.1p와 유사하지만, DiffServ는 우선 순위의 타입별로 정책을 정의할 수 있다는 점에서 차이가 난다.

미터와 마커의 이해

미터(Meter)는 장비로 입력돼 분류된 트래픽 플로우를 측정한다. 일반적으로 트래픽 플로우의 입력 속도를 이용해 대역폭을 측정하거나 버스트 정도를 측정하는데, 미터는 미리 약속된 트래픽 프로파일과 입력된 트래픽의 프로파일을 비교함으로써 초과여부를 결정한다.

그림 2 | PBR를 이용한 방식

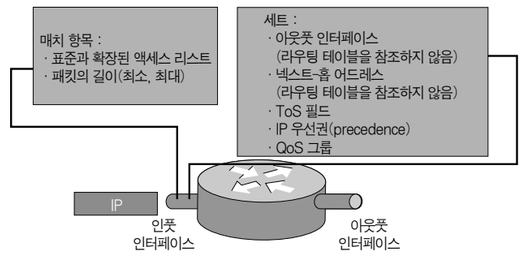


그림 3 | sr-TCM의 동작 방식

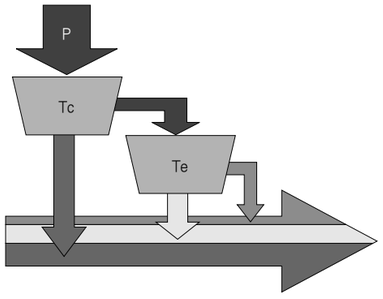
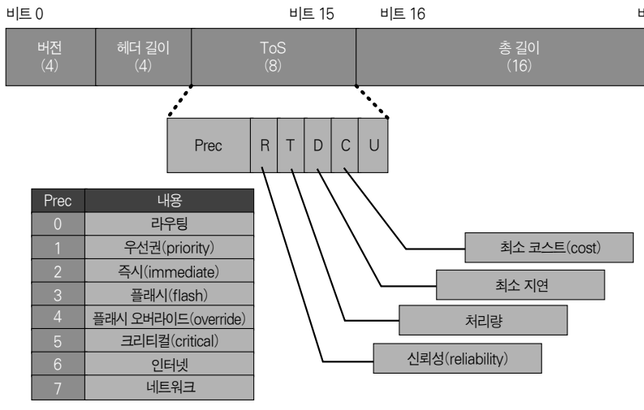


그림 4 | IP데이터그램



그 결과에 따라 마커(Marker)는 적절한 마킹(Marking) 작업을 수행한다.

일반적으로 ▲미리 약속된 트래픽 프로파일을 만족하는 경우 ▲일정한 범위 내에서 초과하는 경우 ▲일정한 범위를 넘어서 초과하는 경우의 세가지로 구분한다. DiffServ 모델에서는 그린(Green), 옐로우(Yellow), 레드(Red)로 표시하지만, 시스코는 Conform, exceed, violate이라는 용어를 사용해 나타낸다.

(그림 3)은 DiffServ 모델에서 가장 많이 사용하고 있는 sr-TCM(single-rate Three-Color Marker)에 대한 동작 방식을 설명한 것으로, 듀얼 토큰 바스킷 구조를 사용하고 있다.

토큰이 떨어지는 속도로는 CIR(Committed Information Rate)를 사용하며, (그림 3)에서 입력된 패킷의 크기 P가 토큰 카운터 Tc보다 작으면 그린 패킷으로 마크하고, Tc보다 크지만 Te보다 적은 경우 옐로우 패킷으로 마크한다. Te보다도 큰 경우에는 레드 패킷으로 마크된다. 토큰 바스킷에 대한 추가 설명은 다음 시간에 소개할 트래픽 세이핑과 폴리싱 부분에서 자세하게 설명할 것이다.

마킹에 필요한 IP 우선권과 TOS 필드

그럼 이제부터 실제로 마킹하는데 사용되는 요소들에 대해 알아보자. IP 우선권(precedence)은 DiffServ 모델이 사용되기 이전에 QoS를 보장하기 위해 사용한 필드로, IP 헤더에 ToS(Type of Service) 필드의 상위 3비트를 사용해 표시한다. 숫자가 클수록 우선순위가 높음을 표시한다.

0의 값은 우선순위가 제일 낮은 BE(Best-Effort)를 뜻하며, 6과 7은 인터넷용과 네트워크용으로 예약돼 있어 실제로 우선권의 값 중에서 가장 우선순위가 높은 것은 5(Critical)이다.

(그림 4)는 IP 데이터그램(datagram)을 나타낸 것으로, ToS의 8비트 중에 상위 3비트가 IP 우선권(precedence)으로 표기되는 모습을 잘 나타낸다. 우선권 뒤의 4비트는 ToS 필드이며, 각 해당 비트마다 신뢰성, 처리량, 지연, 코스트 등을 나타낸다. 해당 필드에 마킹(1로 표기)되면 높은 신뢰성, 높은 처리량, 낮은 지연, 적은 코스트를 뜻하며, 해당 패킷의 중요도를 나타내는데 사용된다.

DiffServ 모델의 핵심 'DSCP 필드'

이름에서도 알 수 있듯이 DSCP(Differentiated Service Code Point) 필드는 DiffServ 모델의 핵심이라고 할 수 있다. DiffServ 모델이 제안되기 이전에 사용됐던 우선권을 포함하면서 확장한 개념이다. 이는 기존 IP 체계에 큰 변화없이 헤더값에서 우선권과 ToS 필드가 사용하던 비트를 대체하는 방법으로, 기존의 우선권이 트래픽 플로우에 대해 세부적인 컨트롤을 할 수 없었던 한계를 극복했다.

DiffServ 모델에서는 IP 헤더의 ToS 필드에서 상위 6비트를 이용해 DSCP 필드 값으로 사용, 패킷들을 최대 64개의 클래스로 구분했다. 그러나 64개의 DSCP 필드값 중에서 제일 마지막 6번째 비트값이 1인 필드는 실험용이거나 사설용으로 예약돼 있다. 때문에 실제로는 32개의 표준 PHB(Per-Hop Behavior)가 클래스로 사용될 수 있다

(표 2)에서 보듯이 DSCP의 종류는 다음과 같이 4가지로 나눌 수 있다.

- 디폴트 PHB(Best-effort)
- CS(Class Selector) PHB (IP Precedence)
- AF(Assured Forwarding) PHB
- EF(Expedited Forwarding) PHB

이중에서 디폴트 PHB(베스트 에포트)는 가장 낮은 우선순위를 뜻하며, CS(Class Selector) PHB는 클래스를 결정하는 상위 3비트만 표시해 기존의 우선권 값과 서로 호환해서 사용할 수 있다.

AF(Assured Forwarding) PHB는 4개의 클래스가 존재하며, 각 클래스에 대해 3개의 서로 다른 드롭 우선권을 갖게 된다. 드롭 우선권(drop precedence)이란 DSCP 코드값 중 클래스를 결정하는 상위 3비트를 제외한 4, 5번째 비트가 드롭될 가능성의 고/저를 표시하는 것으로, 프레임 릴레이의 DE 비트와 성격이 비슷하지만 더 세밀하게 설정할 수 있다.

EF(Expedited Forwarding) PHB는 DSCP 코드값 중에서 우선순위가 가장 높은 클래스라는 의미이다. EF는 2진수 값으로 '101110'으로 표현하며, DSCP 값을 인식못하는 장비(DiffServ 모델에 적용되지 않는 장비)에서도 IP 우선권(precedence) 값으로 인식(5→Critical)돼 최고의 서비스를 보장받는다.

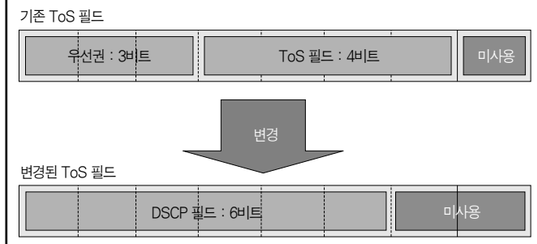
그 외에 마킹용으로 사용하는 다른 필드에는 ▲내부적으로만 사용하는 QoS 그룹 ▲프레임 릴레이 상에서 마킹해 혼잡이 생겼을 때 우선적으로 드롭되게 설정하는 DE(Discard Eligible) 비트 ▲ATM에서 사용하는 CLP(cell Loss Priority) 비트 ▲IEEE 802.1Q나 ISL에서 구현하는 CoS 필드 ▲MPLS에서 QoS 마킹용으로 사용하는 실험적인(experimental) 비트 등이 있다.

실전! 분류자와 마킹 따라하기

지금까지 분류자와 마킹에 대해 자세히 알아봤다. 이제부터는 사례를 통해 이들이 실제 상황에서 어떻게 사용되는지 설명하도록 한다.

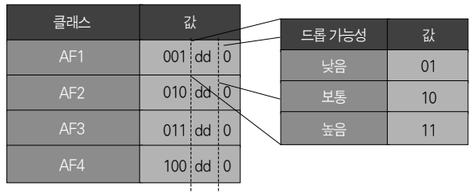
(그림 7)은 네트워크 상에 VoIP 트래픽과 일반 데이터 트래픽이 혼합돼 흐르고 있는 환경이다. 처음하는 실전 훈련이니 가장 쉽게 VoIP 트래픽과 일반 데이터 트래픽을 클래스로 나누고, DSCP 값을 이용

그림 5 | DSCP의 구현



* IP 헤더의 우선권(precedence) 필드와 ToS 필드를 DSCP 필드로 변경함

그림 6 | AF(Assured Forwarding) PHB



* 4개의 표준 클래스(af1, af2, af3, af4)
* DSCP 값의 범위 : 'aaaddd0' 여기서 'aaa'는 클래스 값을 표현하고, 'dd' 드롭 가능성의 높고 낮음을 의미한다.

표 2 | IP우선권과 DSCP의 비교

구분	이진수 값	이름	구분	이진수 값	이름
Precedence 0	**000**	routine	DSCP 0	**000**000	Best-effort
Precedence 1	**001**	priority	DSCP 8	**001**000	CS1
Precedence 2	**010**	immediate	DSCP 16	**010**000	CS2
Precedence 3	**011**	flash	DSCP 24	**011**000	CS3
Precedence 4	**100**	flash override	DSCP 32	**100**000	CS4
Precedence 5	**101**	critical	DSCP 40	**101**000	CS5
Precedence 6	**110**	internet	DSCP 48	**110**000	CS6
Precedence 7	**111**	network	DSCP 56	**111**000	CS7
			DSCP 10	**001**010	AF11
			DSCP 12	**001**100	AF12
			DSCP 14	**001**110	AF13
			DSCP 18	**010**010	AF21
			DSCP 20	**010**100	AF22
			DSCP 22	**010**110	AF23
			DSCP 26	**011**010	AF31
			DSCP 28	**011**100	AF32
CS = Class Selector			DSCP 30	**011**010	AF33
AF = Assured Forwarding			DSCP 34	**100**010	AF41
EF = Expedited Forwarding			DSCP 36	**100**100	AF42
			DSCP 38	**100**110	AF43
			DSCP 46	**101**110	EF

해 마킹하는 실습을 해보자. 연습문제 1은 (그림 7)과 같은 환경에서 수행해야 하는 조건들이다.

연습문제 1

1. VoIP 트래픽은 DSCP 값을 EF(Experdited Forwarding) PHB로 마킹하라
2. 일반 데이터 트래픽은 DSCP 값을 디폴트(베스트-에포트) PHB로 마킹하라
3. CB(Class-Based) 마킹을 사용해 작성하라
4. 라우터 3에 설정하고, 설정한 내용을 확인하라

해결

연습문제 1에 대한 CB(Class-Based) 마킹 설정은 다음과 같다.

```
R3# conf t
Enter configuration commands, one per line. End with CNTL/Z.

R3(config)#ip cef
R3(config)#class-map voip-rtp          → 클래스 맵의 이름을 voip-rtp라고 정의
R3(config-cmap)#match ip rtp 16384 16383 → VoIP 트래픽을 구분하는 과정
R3(config-cmap)#policy-map voip-and-be → 폴리시 맵의 이름을 voip-and-be라고 정의
R3(config-pmap)#class voip-rtp         → voip-rtp라는 클래스 맵을 적용
R3(config-pmap-c)#set ip DSCP EF       → VoIP 트래픽에 대해 DSCP 값을 EF로 설정
R3(config-pmap-c)#class class-default  → 나머지 트래픽들을 표시함
R3(config-pmap-c)#set ip dscp default  → 나머지 트래픽들에 대해 DSCP 값을 디폴트로 설정
R3(config-pmap-c)#interface e 0/0
R3(config-if)#service-policy input voip-and-be → 해당 인터페이스에 폴리시 맵을 적용
R3(config-if)#end
R3#
```

CB(Class-Based) 마킹 설정 확인 내용은 다음과 같다.

```
R3#show running-config
Building configuretion
!Portions removed to save space
ip cef
!
class-map match-all voip-rtp
  match ip rtp 16384 16383
!
policy-map voip-and-be
  class voip-rtp
    set ip dscp 46
  class class-default
    set ip dscp 0
!
interface Fastethernet0/0
ip address 192.168.3.253 255.255.255.0
service-policy input voip-and-be
```

생각보다 설정은 그리 어렵지 않다. 결국 QoS라는 것이 생각만큼 그렇게 거창한 것이 아니라는 것을 느꼈을 것이다. 이번에는 동일한 환경이지만 (그림 8)처럼 중급 정도의 설정을 할 것이다. 어렵게 생각하지 말고 차근히 따라하면 분류자와 마킹을 확실하게 익힐 수 있을 것이다.

그림 7 | VoIP와 데이터 트래픽이 혼합된 네트워크 구성

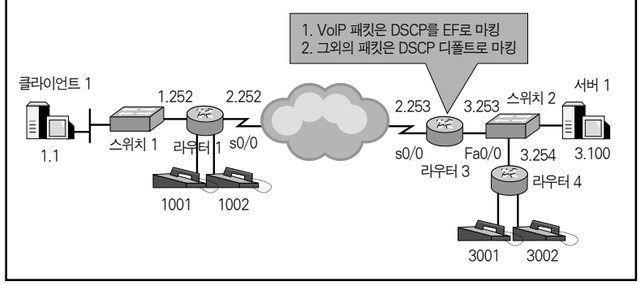
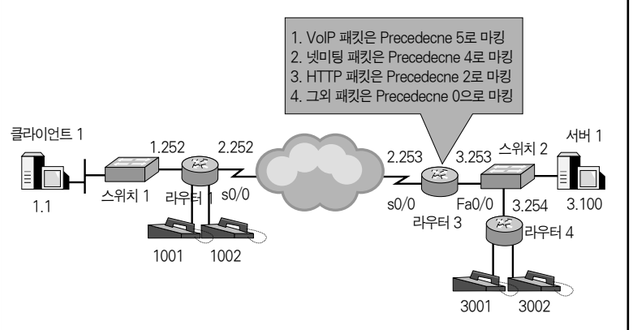


그림 8 | VoIP와 데이터 트래픽이 혼합된 네트워크 구성



연습문제 2

1. VoIP 트래픽은 우선권(precedence) 값을 5로 마킹하라
2. 서버 1에서 클라이언트 1로 가는 넷미팅 비디오와 음성 패킷은 우선권(precedence) 값을 4로 마킹하라
3. 모든 HTTP 패킷들은 우선권(precedence) 값을 2로 마킹하라
4. 일반 데이터 트래픽은 우선권(precedence) 값을 0으로 마킹하라
5. PBR(Policy-Based Routing)를 사용해 작성하라

해결

연습문제 2에 대한 답으로, PBR 설정은 다음과 같다.

```
ip route-chache policy
!
ip access-list extended VoIP-ACL  → Named ACL의 이름을 VoIP-ACL이라 정의
  permit udp any range 16384 32768 any range 16384 32768
                                  → VoIP 트래픽을 구분하는 과정
!
ip access-list extended NetMeet-ACL  → Named ACL의 이름을 NetMeet-ACL이라 정의
  permit udp host 192.168.1.100 range 16384 32768 192.168.3.0 0.0.0.255 range 16384 32768
!
ip access-list extended http-acl  → Named ACL의 이름을 http-acl이라 정의
  permit tcp any eq www any       → 목적지 포트가 TCP 80인 패킷을 구분
  permit tcp any any eq www       → 소스 포트가 TCP 80인 패킷을 구분
!
interface fastethernet 0/0        → 해당 인터페이스에 폴리시 맵을 적용
  ip policy route-map voip-routemap
!
rotue-map voip-routemap permit 10 → 라우트맵을 이용해 10번부터 순차적으로 적용됨
  match ip-address NetMeet-ACL    → 먼저 넷미팅 패킷을 구분(작은 범위 먼저 적용 원칙)
    set ip precedence 4           → 넷미팅 패킷에 대해 precedence 값을 4로 마킹
!
rotue-map voip-routemap permit 20 → 라우트맵 10번을 지난 패킷이 20번에 적용된다
  match ip-address VoIP-ACL       → VoIP 패킷을 구분
    set ip precedence 5           → VoIP 패킷에 대해 precedence 값을 5로 마킹
!
rotue-map voip-routemap permit 30
  match ip-address http-acl       → HTTP 패킷을 구분
    set ip precedence 2           → HTTP 패킷에 대해 precedence 값을 2로 마킹
!
rotue-map voip-routemap permit 40
    set ip precedence 0           → 나머지 패킷들에 대해 precedence 값을 0으로 마킹
!
end
```

PBR 설정 확인은 다음과 같다.

```
R3#show ip policy
Interface              Route map
Fastethernet0/0      voip-routemap

R3#show route-map
route-map voip-routemap, permit, sequence 10
  Match clauses:
    ip address  (access-lists): NetMeet-ACL
  Set clauses:
    ip precedence flash-override
  Policy routing matches: 3 packets, 222 bytes
route-map voip-routemap, permit, sequence 20
  Match clauses:
    ip address  (access-lists): VoIP-ACL
  Set clauses:
    ip precedence Critical
  Policy routing matches: 14501 packets, 1080266 bytes
route-map voip-routemap, permit, sequence 30
  Match clauses:
    ip address  (access-lists): http-acl
  Set clauses:
    ip precedence immediate
  Policy routing matches: 834 packets, 1007171 bytes
route-map voip-routemap, permit, sequence 40
  Match clauses:
  Set clauses:
    ip precedence routine
  Policy routing matches: 8132 packets, 11263313 bytes
```

이번 시간에는 DiffServ 모델에 대한 이론과 더불어 실제 환경에서 그것이 어떻게 적용되는지 다각도로 살펴봤다. 다음 시간에서는 네트워크 대역폭 조절에 막대한 영향을 끼치고 있으며, 현재 가장 이슈가 되고 있는 트래픽 세이핑과 폴리싱에 대해 알아보자. NET