# 네트워크 프로토콜 설계 및 실습 과제 5

전기정보공학부 2016-10769 이권형

## 1. Goal of the experiment

각각 두 개의 OnOff application과 PacketSink application을 이용한 topology를 만들고, 이때 각 OnOff application으로부터 packet이 공유하게 되는 channel에 대해 두 application으로부터의 packet이 어떻게 Sink에 도달하는지 관찰한다. 공유되는 channel을 CSMA, P2P로 바꾸어 가며 ns-3로 simulation 이후 capture된 packet을 wireshark로 traffic을 분석한다. 이후 CSMA channel에 대해 하나의 OnOff application에서 Data Rate를 기본 1Mbps에서 0.5, 1, 2 Mbps로 변경시키며 traffic의 변화를 관찰한다.

## 2. Topology & realization

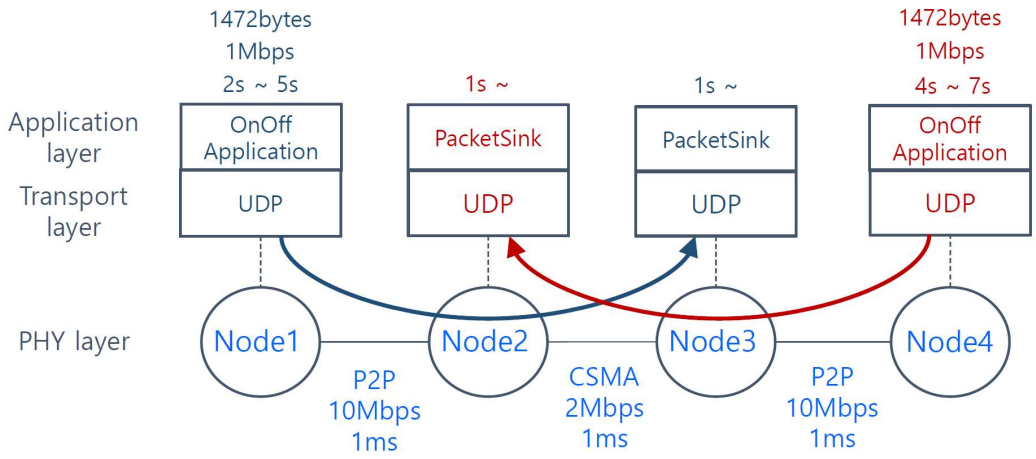


그림1. Simulation Topology

위와 같은 topology에서, 각 OnOff application의 packet들은 Node2와 Node3를 연결하는 channel을 공유하게 된다. 따라서 이 channel에서 동시에 서로 다른 packet을 어떻게 처리하는지 관찰하는 것이 본 실험의 목적이다. Transport layer의 protocol은 모두 UDP를 사용하고, Node2와 Node3를 연결하는 channel을 CSMA, P2P로 바꿔가며 관찰하고 Node4에 설치된 application의 Data Rate을 1Mbps에서 0.5, 1, 2Mbps로 바꿔가며 관찰한다. 본 실험에서는 따라서 Node2와 Node3의 channel만 관찰할 것이다.

이를 구현하기 위한 code에 대한 대략적인 설명은 다음과 같다. 자세한 사항은 code의 주석에 표시되어 있다.

먼저, node를 4개 만든 후 각각의 node를 연결하는 channel을 설정한다. Node1과 Node2를 연결하는 channel과 Node3과 Node4를 연결하는 channel은 PointToPointHelper를 이용하여 생성하고, DataRate와 Delay를 명시된 것과 같이 입력한다. Node2와 Node3를 연결하는 channel의

경우 csma라는 argument를 command line으로 받아 1이면 CsmaHelper를 이용하고, 0이면 PointToPointHelper를 이용한다.

각각의 channel을 devices_1, _2, _3으로 표시되는 container에 설치하고, address를 subnet을 변경해가며 할당한다. 이후 PacketSinkHelper를 이용하여 Node2와 3에 PacketSink Application을 설치하고, OnOffHelper를 이용하여 OnOff application을 Node1과 4에 설치한다. 각 application의 시작과 끝을 정한 뒤 Ipv4GlobalRoutingHelper::PopulateRoutingTables()을 이용해 global routing table을 자동으로 설정한다.

## 3. Result and Discussion

1) Compare CSMA vs P2P

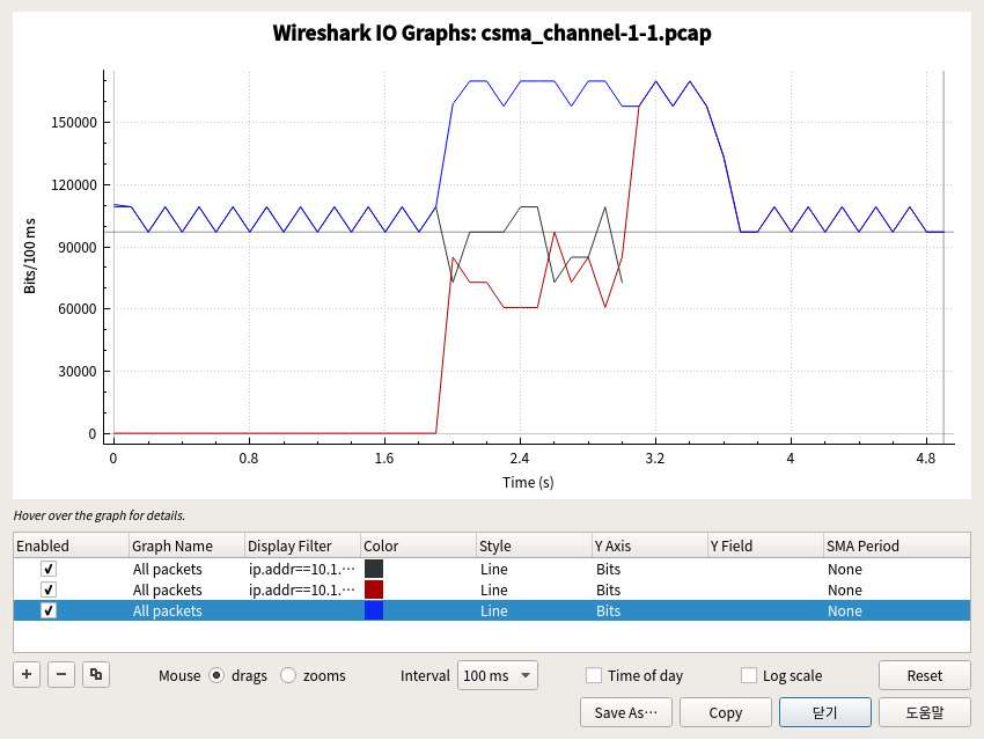


**그림2.** Node2와 Node3의 channel에 대해 CSMA 사용 시 I/O graph

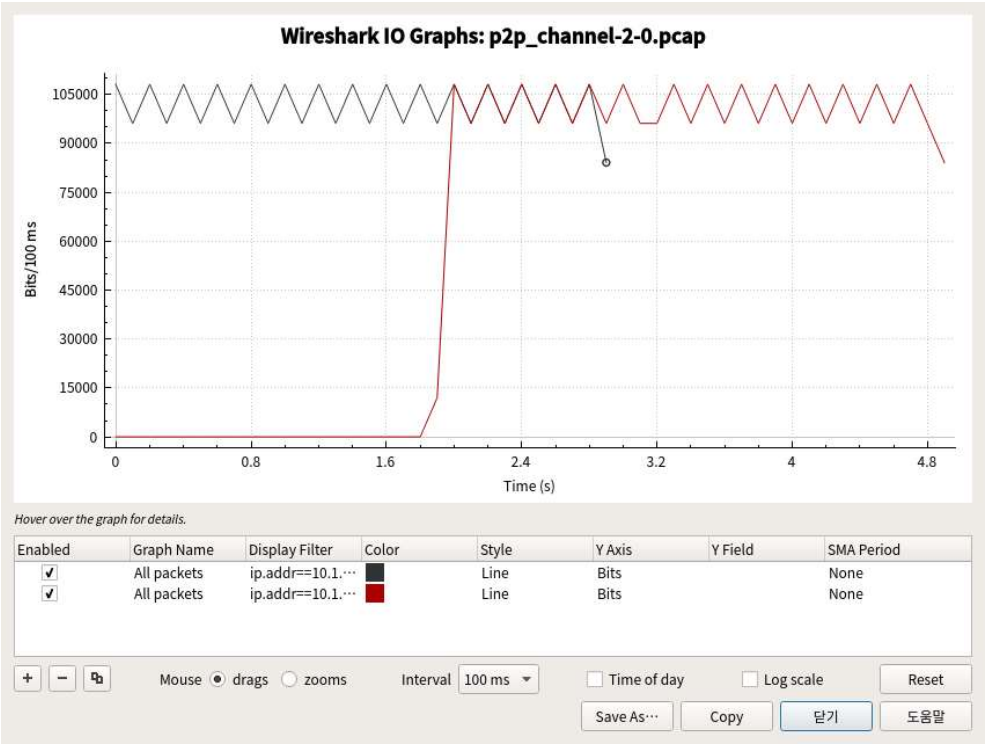


**그림3.** Node2와 Node3의 channel에 대해 P2P 사용 시 I/O graph

그림 2와 그림 3은 각각 CSMA와 P2P를 사용했을 시의 결과이다. 검정색 선은 Node1에서 Node3으로 가는 packet을, 빨간색 선은 Node4에서 Node2로 가는 packet을 나타낸다. 그림2에서 파란색 선은 모든 packet을 나타낸다. Node1에서는 2초부터 5초까지, Node 4에서는 4초에서 7초까지 packet을 전송한다. wireshark는 처음으로 packet이 관찰될 때를 0초로 하여 나타내므로, 실제로는 2 + delay(Node1에서 Node3까지)정도의 offset이 존재한다.

CSMA와 P2P는 packet의 크기가 다르다. CSMA의 경우 1514bytes이고, P2P는 1502bytes이다. 이는 link layer에서의 header크기가 각각 18, 2 bytes인 것으로 인해 생기는 차이이다. CSMA의 경우 MAC address를 사용하여 header의 크기가 더 크다.

CSMA와 P2P 모두 Node1만 전송하는 0~2초에서는 1Mbps정도의 throughput을 보여준다. 삼각파형으로 보이는 것은 wireshark에서 packet이 discrete하게 도착하는 것을 곡선으로 표현하기 때문이다. 이후 다르게 보이는 것은 Node1에서와 Node4에서 동시에 전송하여 channel을 공유하게 되는 2초부터이다.

CSMA의 경우 2초에서 4초까지 각 Node의 packet에 대해 불안정한 throughput을 보인다. 그 합은 2Mbps정도로 일정한 편이지만, 대체로 Node1에서의 throughput이 높게 나오면서도 Node2의 packet이 더 높은 순간도 관찰된다. 또한 이 때의 전체 packet throughput은 1.7Mbps정도로 channel을 모두 이용하지 못하는 것을 확인할 수 있다. CSMA는 collision을 피하기 위해 모든 Node에서 packet을 listen하다가, collision이 발생하였을 때 jamming signal을 통해 packet송신을 중단하고, 각 Node에서 random한 시간 이후 전송을 다시 시작한다. 이를 반복하다 보니 collision에 의해 channel에 할당된 throughput을 모두 활용하지 못하게 된다.

또한 Node1에서 전송을 중지한 이후 Node4에서의 전송된 packet의 traffic이 갑자기 증가하는 것을 확인할 수 있다. 이 값은 1Mbps이상으로 Node4에서 전송하는 traffic보다 많은 양인데, 이는 queueing에 의한 것으로 생각된다. collision에 의해 전송되지 못한 packet을 queue에 저장하여 collision이 발생하지 않을 때 한꺼번에 보내는 것이다.

P2P의 경우 Time Division Multiplexing을 사용하는데, channel의 사용 time을 일정한 간격으로 잘라 각 Node에 할당하는 것이다. 따라서 weight의 차이가 없다고 가정하면, channel을 공평하게 사용하게 된다. 이로 인해 channel의 throughput을 모두 사용할 수 있고, 현재 각 application에서 전송하는 traffic이 1Mbps이기 때문에 간섭 받지 않고 channel을 반으로 나누어 사용하는 것이다.

2) Variate DataRate of Node4 Application

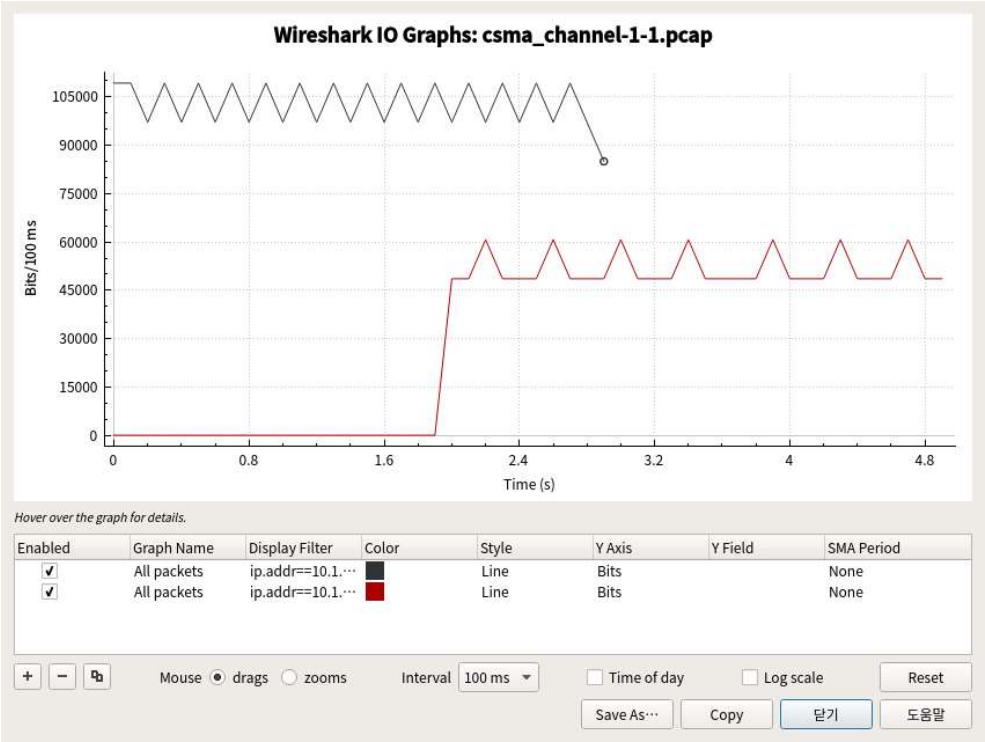


**그림4.** Node4의 Data Rate을 0.5Mbps로 설정한 결과

그림 4는 0.5Mbps로 설정했을 때의 결과이다. 1)에서의 CSMA와는 달리 각 Node에서 보내는 packet이 일정한 throughput을 보인다. 즉, channel에서 packet의 간섭이 없고 collision이 발생하지 않는 것이다. 즉 channel의 throughput보다 낮은 사용량을 가질 때는 CSMA에서도 P2P와 비슷한 양상을 보이는 것을 확인할 수 있다.


**그림5.** Node4의 Data Rate을 1Mbps로 설정한 결과

위 결과는 1)에서의 CSMA와 같이 나타난다. 최대 throughput이 1.7Mbps를 넘지 못하는 것을 확인할 수 있고, 이는 collision에 의한 것임을 알 수 있다. packet의 queueing이 발생하는데, 이 때 queue의 size에 따라 packet loss가 발생할 수 있다. 이를 계산하기 위해, 대략적인 throughput과 시간을 계산하여 총 전송된 data를 구하면 다음과 같다.

$$2sec * 1Mbps + (3.4sec - 2sec) * 1.7Mbps + (5sec - 3.4sec) * 1Mbps = 5.98Mb$$

이는 대략 6Mb정도로, Node1과 Node2에서 전송하는 1Mbps를 각각 3초간 전송한 data와 같게 나온다. 따라서 이 경우 packet loss는 발생하지 않았음을 알 수 있다.

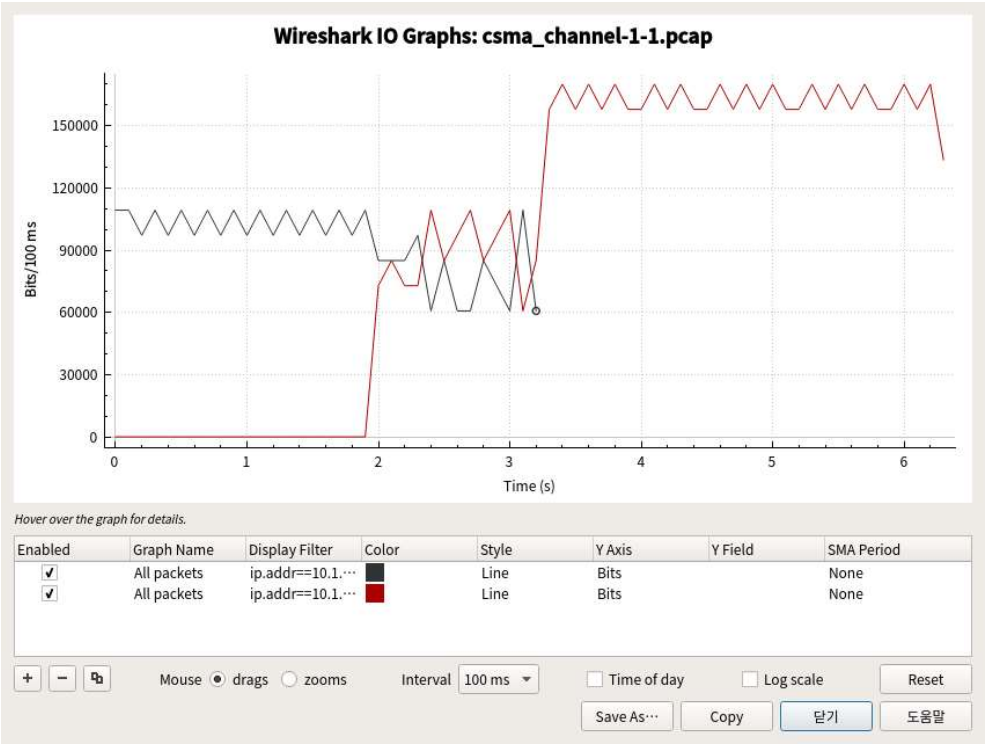


**그림6.** Node4의 Data Rate을 2Mbps로 설정한 결과

그림 6은 Data Rate를 2Mbps로 설정한 결과이다. 다른 결과들과는 다르게 전송시간이 6.2초까지 관찰되며, 이는 collision이 과도하게 발생하여 queueing된 packet을 전송하기 위한 것임으로 보인다. 두 Node에서 동시에 packet을 보내는 구간에서 여전히 1.7Mbps정도의 throughput을 보이고, 그 구간이 달라졌음을 확인할 수 있다. 1Mbps에서는 2초에서 3초정도까지 traffic이 겹쳤지만, 2Mbps에서는 2초에서 3.2초정도까지 traffic이 겹치게 관찰된다. 이는 역시 collision에 의해 Node1의 전송이 끝나는 시점이 0.2초 늦어진 것으로 생각된다. Node4에 의한 packet이 queue에 빠르게 쌓임에 따라 Node3에서 이를 감당하지 못하고 더 자주 packet을 송신하도록 하고, 이로 인해 Node4에서의 packet의 throughput이 더 높게 관찰된다.

또한 총 전송된 data의 양을 계산하면 다음과 같다.

$$2\text{sec} * 1\text{Mbps} + (6.2\text{sec} - 2\text{sec}) * 1.7\text{Mbps} = 9.14\text{Mb}$$

이는 대략 9Mb로 생각할 수 있는데, Node1에서 3Mb, Node4에서 $2 * 3 = 6$Mb를 전송한 것을 합한 9Mb와 같다. 따라서 이 경우에도 packet loss는 발생하지 않았음을 알 수 있다. Node4에서 전송한 packet은 collision이 발생하는 동안 모두 queueing되었다가, 이후 1.7Mbps의 일정한 throughput을 통해 모두 전송되었다.

collision이 발생하지 않음에도 불구하고 channel의 최대 throughput이 1.7Mbps로 유지되는 이유는 CSMA의 protocol에 의한 것이다. CSMA에서는 collision이 발생하지 않더라도 propagation delay $\tau$와 normalization factor D에 대해 $\frac{1}{1+\frac{\tau}{D}}$배 만큼의 channel만 이용할 수 있는데, 이 때 D는 packet size를 throughput으로 나눈 것으로, 6ms정도이다. 따라서

channel의 throughput 2Mbps에 대해 $\frac{1}{1+\frac{1}{6}} * 2 = 1.7Mbps$ 정도의 maximum throughput을 계산할 수 있다. 이는 관찰한 것과 같은 수준의 throughput으로, CSMA에서 channel의 utility가 제한되어 있음을 확인할 수 있다.

## 4. 참고문헌

- Kurose, J. F. and K. W. Ross (2012). Computer Networking: A Top-Down Approach (6th Edition).

- ns3 network 1주차 강의 자료.