ISBN 89-5884-682-8 98560

네트워크 가상화를 위한 Virtual Routing 제공 방법

일자: 2006년 10월 30일
부서: 슈퍼컴퓨팅센터 연구망개발팀
제출자: 홍원택, 김민아, 공정욱
{wthong, petimina, kju}@kisti.re.kr

한국과학기술정보연구원
305-806 대전광역시 유성구 어은동 52번지
TEL (042)869-0676 / FAX (042)869-0679
www.kisti.re.kr

네트워크 가상화를 위한 Virtual Routing 제공 방법

작성자: 홍원택, 김민아, 공정욱

연구망개발팀, 슈퍼컴퓨팅센터, 한국과학기술정보연구원
{wthong, petimina, kju}@kisti.re.kr

최종수정일: 2006년 10월 30일

Abstract

네트워크 가상화는 다수의 가상 네트워크들이 공유된 물리 매체를 기반으로 공존하는 기법이다. 이를 통해 물리적 네트워크 노드들의 추가적인 설치 없이 논리적으로 독립된 네트워크들을 생성할 수 있고, 이러한 점은 비용 절감 측면 및 관리의 유연성을 제공한다. 본 보고서에서는 Juniper M5 라우터를 대상으로 네트워크 가상화와 관련된 logical router 및 virtual router instance 기능을 조사 및 분석하고, 여러 조합에 따른 종단간 성능을 평가한다.

Topics

1. 서론

본 문서는 네트워크 가상화를 위한 virtual routing 제공 방법에 관한 것이다. 문서의 구성은 네트워크 가상화의 개념에 대해 살펴보고, 여러 네트워크 벤더들 중에서 Juniper 장비를 대상으로 virtual routing 메커니즘을 확인한다. 특히, 대상 장비에

서 제공하는 virtual routing 방법들 중에서 logical router와 virtual router instance의 기능을 조사하고, 여러 조합에 따른 망 토폴로지들을 구성한 후 종단간 성능 시험을 통해 각각의 구성을 비교 분석한다. 본 문서에서는 Juniper M5 라우터를 대상으로 실험을 수행하였고, 각각의 망 토폴로지에 따른 설정 파일을 추가적으로 덧붙인다.

2. 네트워크 가상화 개념

네트워크 가상화 개념은 논리적으로 분리된 네트워크를 제공한다는 측면에서 람다 네트워킹의 본래 목적과 맥을 같이 한다고 할 수 있다. 네트워크 가상화의 개념을 살피기에 앞서, 람다 네트워킹의 개념을 먼저 알아본다.

2.1 람다 네트워킹

최근, 독점적인 네트워크 사용의 필요성이 증대되면서 람다 네트워킹 기술이 연구 및 교육 망 분야에서 활발히 사용되는 추세이다. 엄밀한 의미로 본래 통신 분야에서의 람다 네트워킹[1, 2]은 테라 비트 수준의 대역폭을 갖는 광섬유 네트워크를 효과적으로 활용하기 위한 기술로서, 입력 신호를 하나의 파장(람다)에 할당하여 전용 대역폭을 갖는 독립적인 연결을 만들어 결과적으로 하나의 광섬유를 통해 다수의 신호를 동시에 전송할 수 있는 기술이다. 이렇게 확보된 구간에는 전용 회선이 구성되고, 이를 통해 네트워크 상의 일정 자원을 독점할 수 있으며, 라우팅 장비 입출력에 따른 홉(Hop)과 지연(Delay) 현상을 감소시킬 수 있는 장점을 갖고 있다. 보편적으로 람다 네트워킹의 의미는 본래의 정의에서 더욱 확장되어 Layer 1을 포함한 상위 계층으로까지 적용되고 있다.

람다 네트워킹을 통해 종단간 대용량 데이터를 전송하려는 그리드, e-Science 응용 연구자들은 고 성능, 저 지연의 특성을 갖는 독점적인 대역을 제공 받을 수 있다. 일반적 관점에서의 람다 네트워킹의 목적은 특정 목적을 같이 하는 연구 그룹들이 그들 자신의 "lightpath"[3]를 제공 받는 것이다. 이러한 "lightpath"를 통해 연구자들은 고 대역의 응용에 특화된 독점적인 네트워크를 구성할 수 있게 된다. "lightpath"는 일정 대역을 보장하는 종단간 연결로서 다양한 종류의 링크들로 생성될 수 있다. 예를 들어, CWDM 또는 DWDM상의 아날로그 파장, SONET 또는 SDH 회선의 STS 채널, ATM CBR 연결, 기 정의된 대역 또는 QoS를 갖는 MPLS LSP, 패킷 기반 네트워크상에서의 Diffserv "gold" 서비스, IEEE 802.1q tagging을 이용한 Ethernet Vlan 기술들은 각각의 계층에서 "lightpath"의 예가 될 수 있다. 이러한 기술들은 Layer 1에서 Layer 3 사이에 걸쳐 존재하고 있고, 특히 Layer 2와 Layer 3에서의 기술들은 액세스 네트워크 상에서 일반 사용자들을 위해 일정 대역을 보장하는 종단간 연결을 제공하는데 유용하게 사용될 수 있다.

"lightpath" 프로비저닝의 요구가 증대되면서 원하는 시점에 "lightpath"를 동적으로 할당해 주는 서비스에 대한 요구가 증가되었다. 현재 요구 기반 대역 할당 서비스

와 관련된 연구 프로젝트들이 있다. UCLP (User Controlled LightPaths) [4, 5] 프로젝트는 일반 사용자의 관점에서 트래픽 엔지니어링이 가능한 방법을 제공한다. 즉, 일반 사용자는 UCLP 소프트웨어를 이용하여 단일 혹은 복수의 도메인에서 Layer 1에서의 광 패스를 동적으로 프로비저닝할 수 있다. CHEETAH (Circuit-switched High-speed End-to-End Transport ArcHitecture) [6] 프로젝트는 동적인 call-by-call 메커니즘에 사용하여 종단간 고 대역의 연결을 제공한다. 특히, 엄격한 QoS가 요구되는 광범위한 첨단 응용들을 지원하기 위해 단순한 고 대역 제공의 관점보다는 지연/지터(Jitter) 이슈에 좀 더 초점을 둔다. 한편, DRAGON (Dynamic Resource Allocation via GMPLS Optical Networks) [7] 프로젝트는 광 장비에 기반한 코어/에지 네트워크에서 동적인 패스 프로비저닝 서비스를 제공한다. DRAGON은 주로 GMPLS-enabled 네트워크 환경에 적합한 방법으로 응용에 특화된 토폴로지를 생성하는 부가적인 다양한 API들을 제공한다. 위에서 언급된 연구 프로젝트들은 주로 인터넷 환경에서의 서비스 질을 높이고, 네트워크 대역을 조절할 수 있는 방법론을 제공한다.

2.2 GENI 네트워크 가상화

위에 언급된 프로젝트들과는 달리 미국 NSF (National Science Foundation)에 의해 지원되는 GENI (Global Environment for Network Innovations) [8] 프로젝트는 최근에 시작되어 전통적인 인터넷 구조의 고착화 현상을 해결하고 광, 무선, 센서 네트워크 및 분산 시스템에 기반하여 새로운 네트워크 구조 및 서비스들을 설계하는 것을 목표로 하고 있다. 현재까지 GENI의 목표를 따르는 기존의 프로젝트들이 있어왔다. 우선, PlanetLab[9]은 일종의 글로벌 규모 오버레이 네트워크로서 지역적으로 분산된 일종의 리눅스 PC들을 집합이다. 일반 사용자들은 하나의 서버 계정을 갖게 되면 분산된 PlanetLab의 모든 자원들을 접근할 수 있다. 이러한 PlanetLab은 실제 WAN 환경에서 새로운 글로벌 규모의 다양한 서비스들을 개발하고 테스트하는데 매우 유용하게 사용되고 있다. 이러한 PlanetLab의 접근 방법도 일종의 GENI 프로젝트의 한 범주로서 분산화된 가상화 일환으로 볼 수 있다. 실제로 GENI는 보다 광의의 개념으로 다양한 종류의 종단 노드들을 지원하고, 네트워크 링크 수준에서의 동작을 공개한다. 또한 GENI를 이용하는 첨단 응용 연구자들에게 진입 장벽을 낮추기 위해 여러 종류의 서비스들을 제공한다.

GENI의 여러 연구 영역 중 기존 인터넷 구조의 문제점과 한계를 극복하려고 하는 시도가 Figure 1에서의 GENI 네트워크 가상화 [10]이다. 이를 통해 다수의 가상 네트워크들이 공유된 물리 매체를 기반으로 공존할 수 있다. 서로 다른 가상화된 네트워크들은 각각 여러 개의 종단간 패킷 전달 시스템을 지원할 수 있고, 각각의 가상 링크에서 다른 프로토콜들과 패킷 포맷들을 사용할 수 있다. 가상화된 네트워크들은 virtual router 메커니즘 및 virtual link들에 의해 구현될 수 있다. virtual router 메커니즘에 의한 논리적인 네트워크 노드의 추가를 통해 추가적인 물리적인

네트워크 노드들이 설치될 필요가 없다는 점에서 이러한 접근 방법은 비용 절감 측면 및 관리의 유연성을 제공한다. 일반적으로 라우터를 독립적인 라우팅 태스크를 수행하는 여러 개의 virtual router들로 분리할 수 있다. virtual router들은 각각의 태스크를 수행하므로 라우터의 사용을 극대화시킬 수 있다. 네트워크 가상화와 관련된 용어들은 네트워크 벤더들에 따라 약간의 차이가 존재한다.

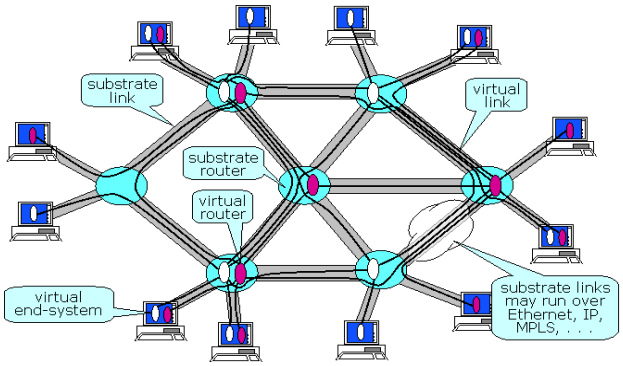

Figure 3 GENI 네트워크 가상화 [10]

3. Juniper 장비에서의 Virtual routing

실제로 네트워크 가상화 개념은 Layer 3에서의 virtual routing 메커니즘에 의해 구현될 수 있고, Cisco와 Juniper와 같은 네트워크 장비 벤더들은 virtual routing 개념을 지원하고 있다. 본 절에서는 Juniper의 M5 라우터를 대상으로 Virutal routing 개념에 대해 알아본다. Juniper 네트워크 장비들은 logical router 및 virtual router instance 들을 통해 virtual routing 메커니즘을 제공한다. 두 용어가 혼재되어 사용되지만, 둘 사이에는 약간의 차이가 존재한다.

Table 1에서 보는 것처럼 logical router [11]는 실제 라우터를 논리적으로 독립된 여러 개의 라우터들로 분리하고, 각각의 logical router는 본래 라우터의 기능들과 속성들을 동일하게 갖는다. 반면, virtual router instance는 소프트웨어적으로 본래 라우터 기능의 일부를 에뮬레이션한다. logical router가 복수개의 라우팅 테이블을 포함하는 반면, virtual router instance는 단지 하나의 라우팅 테이블을 유지할 수 있다. 또한 하나의 logical router는 여러 개의 virtual router instance들을 포함할 수 있다. 물론, virtual router instance는 logical router의 생성 없이 본래의 기본 라우터인 main router내에서도 생성 가능하다. logical router와 virtual router instance 모두 물리적 라우터의 추가적인 설치 없이 네트워크의 분리 및 유연성을 높이는데 유용하게 이용될 수 있다.

Table 1 Logical router와 Virtual router instance 비교

	Logical router	Virtual router instance
Implementation	Partitioning	Emulation at software layer (One of 6 routing instance types)
Functionality	Almost same	Some limitation (ex. MPLS, RSVP, DVMRP)
Maximum #	Up to 16	No limit (in theory)
Routing table #	Multiple	Single
Where to configure	In a physical router	In a main or logical router

네트워크 가상화에 기반한 새로운 네트워크 토폴로지를 설계할 때, logical router 및 virtual router instance의 차이점을 인식하고 고려할 필요가 있다. 각 조합에 따른 다양한 토폴로지 구성이 가능하다. Figure 2는 물리적 라우터에서의 logical router와 virtual router instance의 예를 보여준다. 왼쪽 그림은 logical router를 여러 개 생성하고 물리적 링크를 공유하여 다음 홉에 있는 라우터와 연결하는 예를 보여준다. 세부적으로 virtual router instance들을 포함하고 있는 main router 및 logical router와 virtual router instance들을 포함하지 않는 logical router들이 존재할 수 있다. 반면 오른쪽 그림은 logical router를 여러 개 생성하여 서로 독립된 물리적 링크로 다음 홉에 있는 라우터와 연결하는 예를 보여준다. Table 2에서 언급된 것처럼 물리적인 링크를 독점하는 경우는 가용 대역의 관점에서 최대 대역을 B라 할 경우 종단간 연결의 수와 관계없이 각각 B 대역을 항상 점유할 수 있다. 실제 이러한 네트워크 토폴로지는 네트워크 가상화를 통해 각각의 링크에서 지속적인 최대 대역을 소비하는 네트워크 응용에 적합하다. 반면, 물리적인링크를 공유하여 네트워크를 분리하는 경우는 물리적 인터페이스의 소비 측면에서 경제적이고 IEEE 802.1q tagging 기법을 사용하여 각각 논리적으로 분리된 종단간 연결을 생성할 수 있다. 또한, 최대 대역을 B라 할 경우 종단간 연결의 수 n에 따라 가용 대역이 변화하게 된다. 직관적으로는 논리적 연결 당 B/n의 대역으로 예상되지만, 물리적인 연결을 서로 논리적으로 공유한다는 측면을 고려할 때 각 연결에서의 트래픽 패턴 및 평균적인 성능을 검증해 볼 필요가 있을 것이다. 이러한 네트워크 토폴로지는 B/n 범위 내에서 최대 트래픽이 발생하는 네트워크 응용에 적합할 것으로 보이고, B/n을 초과하는 연결의 경우에는 각 연결에 대한 추가적인 스케줄링이 요구될 것이다.

Figure 4 링크 공유 및 독점 기반 네트워크 가상화

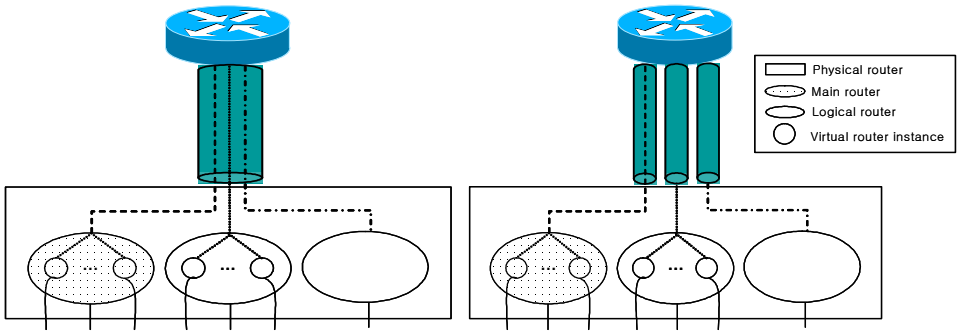

Table 2 Layer 3에서의 네트워크 가상화 방법

	링크 독점	링크 공유		
네트워크 분리방법	다수의 logical router	다수의 logical router, Vlan tagging	Logical router 공유, Vlan tagging	Main router 공유, Vlan tagging
가용 대역 (최대대역:B, 연결수 :n)	B	B/n 추정	B/n 추정	B/n 추정
네트워크 응용 특징	지속적인 최대대역 소비	논리적 연결당 최대 B/n 소비	B/n 범위 내에서 사용자별 트래픽 분리	B/n 범위 내에서 사용자별 트래픽 분리
기타	Logical router 생성 제한	B/n 초과 트래픽 발생시 스케줄링 요구	Virutal router instance 생성	Virutal router instance 생성

Figure 3과 4에서는 Juniper M5에서의 logical router와 virtual router instance의 설정과 관련된 주요 부분을 보여준다.

```
logical-routers {
    lr1 {
        interfaces {
            fe-0/1/0 {
                unit 1 {
                    vlan-id 102;
                    family inet {
                        address 10.21.2.2/24;
                    }
                }
            }
            fe-0/1/2 {
                unit 0 {
                    family inet {
                        address 10.11.3.1/24;
                    }
                }
            }
        }
        routing-options {
            static {
                route 10.11.1.0/24 next-hop 10.21.2.1;
            }
        }
    }
}

interfaces {
    fe-0/1/0 {
        vlan-tagging;
        unit 0 {
            vlan-id 103;
            family inet {
                address 10.31.2.2/24;
            }
        }
    }
...
}
```

Figure 5 Logical router 설정 예

```
interfaces {
    fe-0/1/0 {
        vlan-tagging;
        unit 0 {
            vlan-id 103;
            family inet {
                address 10.31.2.2/24;
            }
        }
        unit 1 {
            vlan-id 102;
            family inet {
                address 10.21.2.2/24;
            }
        }
    ...
    }
...
}

routing-instances {
    blue {
        instance-type virtual-router;
        interface fe-0/1/2.0;
        interface fe-0/1/0.1;
        routing-options {
            static {
                route 10.11.1.0/24 next-hop 10.21.2.1;
            }
        }
    }
...
}
```

Figure 6 Virtual router instance 설정 예

4. Virtual routing에 기반한 망 구성

본 절에서는 logical router 및 virtual router instance를 기반으로 링크를 독점하는 경우와 공유하는 경우의 망 구성을 보이고, 각각의 종단간 성능 실험 환경을 알아본다. 사용한 실험 장비들은 아래와 같다.

Machines: Juniper M5 라우터 2대
JunOS version: 7.2
End Host: Windows XP, Iperf 1.7.0 성능 측정용.

4.1 Logical router 기반 링크 독점

각각의 M5 router에서 logical router를 2개씩 생성한다. 각각의 logical router는 분리된 Physical Link를 이용하여 서로 연결되고, 상호간의 Source/Destination 네트워크를 분리한다. Router 설정은 아래와 같다.

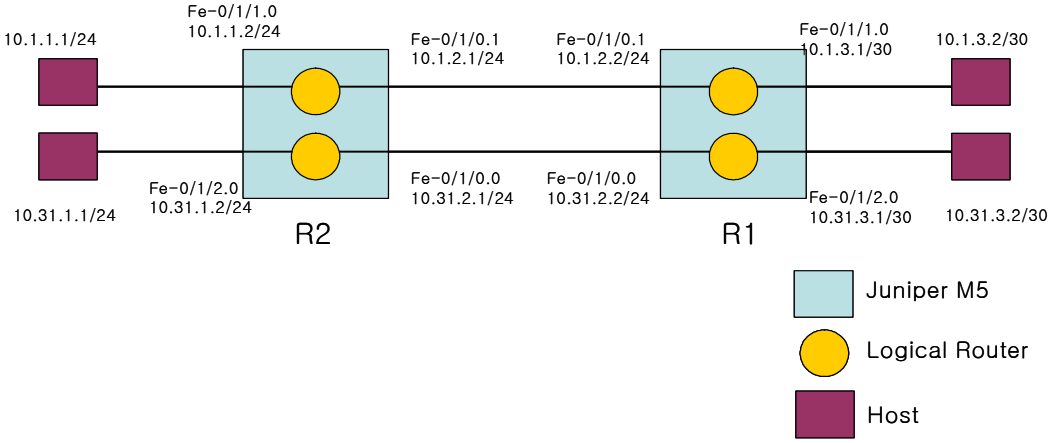

Figure 7 Logical router 기반 링크 독점

```
logical-routers {
    lr2 {
        interfaces {
            fe-0/1/0 {
                unit 0 {
                    family inet {
                        address 10.1.2.1/24;
                    }
                }
            }
            fe-0/1/1 {
                unit 0 {
                    family inet {
                        address 10.1.1.2/24;
                    }
                }
            }
        }
        routing-options {
            static {
                route 10.1.3.0/30 {
                    qualified-next-hop 10.1.2.2;
                }
            }
        }
    }
}
interfaces {
    fe-0/1/2 {
        unit 0 {
            family inet {
                address 10.31.2.1/24;
            }
        }
    }
    fe-0/1/3 {
        unit 0 {
            family inet {
                address 10.31.1.2/24;
            }
        }
    }
}
routing-options {
    static {
        route 10.31.3.0/30 {
            qualified-next-hop 10.31.2.2;
        }
    }
}
```

Figure 8 Logical router 기반 링크 독점 설정

4.2 Logical router 기반 링크 공유

각각의 M5 router에서 logical router를 2개씩 생성한다. 각각의 logical router는 공유된 Physical Link를 이용하여 서로 연결되고, 상호간의 Source/Destination 네트워크를 분리한다. Router 설정은 아래와 같다. 주목해야 할 점은 같은 Physical Link를 공유하기 위해 logical router의 인터페이스에서는 Tagged Vlan ID를 이용하여 L2 VLAN 연결을 유지한다.

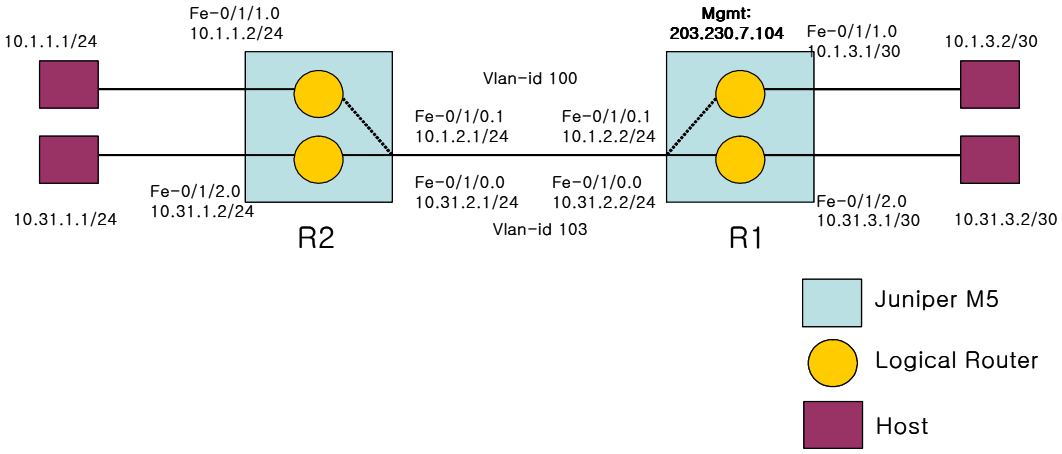

Figure 9 Logical router 기반 링크 공유

```
logical-routers {
    lr2 {
        interfaces {
            fe-0/1/0 {
                unit 1 {
                    vlan-id 100;
                    family inet {
                        address 10.1.2.1/24;
                    }
                }
            }
            fe-0/1/1 {
                unit 0 {
                    family inet {
                        address 10.1.1.2/24;
                    }
                }
            }
        }
        routing-options {
            static {
                route 10.1.3.0/30 {
                    qualified-next-hop 10.1.2.2;
                }
            }
        }
    }
}
interfaces {
    fe-0/1/0 {
        vlan-tagging;
        unit 0 {
            vlan-id 103;
            family inet {
                address 10.31.2.1/24;
            }
        }
    }
    fe-0/1/2 {
        unit 0 {
            family inet {
                address 10.31.1.2/24;
            }
        }
    }
}
routing-options {
    static {
        route 10.31.3.0/30 {
            qualified-next-hop 10.31.2.2;
        }
    }
}
```

Figure 10 Logical router 기반 링크 공유 설정

4.3 Logical router 및 Virtual router instance 조합 기반 링크 공유

본 절에서는 logical router 및 virtual router instance의 조합을 기반으로 네트워크 토폴로지를 구성한다. 물리적 링크를 독점하여 사용하는 경우와 달리 공유된 링크를 사용하여 가상화된 네트워크를 구성한 경우에는 종단간 성능이 보장되는지 확인을 할 필요가 있다. 본 실험의 목적은 100Mpbs 네트워크 환경에서 네트워크 가상화를 시험해 보는 것이다. 실험을 위한 네트워크 토폴로지는 각각 Figure 9, 10, 11과 같다. 일반적인 실험 시나리오는 다음과 같다. 첫째, 공유된 물리적 링크 상에 logical router 및 virtual router instance들을 설정하여 각각의 네트워크 토폴로지를 생성한다. 둘째, 각각의 네트워크 토폴로지에서 양단간 호스트들 사이에 논리적 연결상으로 100초간 1Mbyte의 윈도우 크기를 갖는 TCP트래픽을 동시에 발생시켜 전송한다. 여기서의 논리적인 연결은 각각의 그림에서 볼 수 있듯이 3개의 logical router를 연결한 경우, main router내에서 3개의 virtual router instance들을 연결한 경우, logical router에서 3개의 virtual router instance들을 연결한 경우 등으로 구성된다. 마지막으로 이러한 과정을 10번 반복하여 평균 처리율을 얻는다

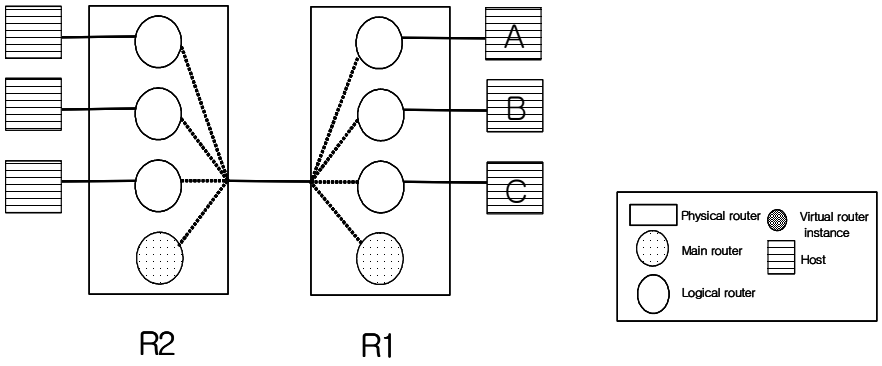

Figure 11 3개의 Logical router를 이용한 네트워크 토폴로지

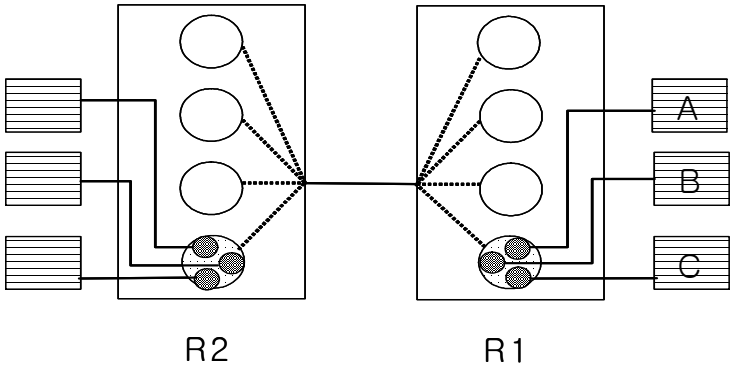

Figure 12 Main router 내의 3개의 virtual router instance를 이용한 토폴로지

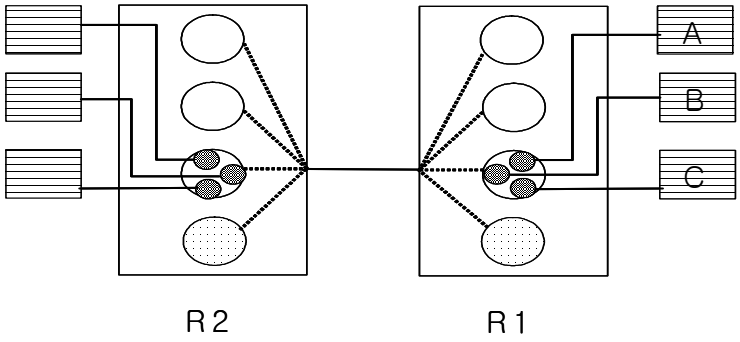

Figure 13 Logical router 내의 3개의 virtual router instance를 이용한 토폴로지

5. 종단간 성능 실험 및 결과분석

4절에서 구성된 토폴로지를 기반으로 TCP 트래픽을 발생시켜 평균 처리율을 시험하였다. Figure 12, 13에서처럼 logical router를 기반으로 링크를 독점한 경우 각 연결은 약 100Mbps에 육박하는 처리율을 보였고, 링크를 공유한 경우에는 양단간 약 50Mbps의 처리율을 보였다. 그리고, TCP 트래픽이 logical router와 virtual router instance의 조합에 따라 가상화된 네트워크에서 어떠한 영향을 받는지 확인하기 위한 실험의 결과는 Figure 14, 15, 16에서 보여준다. 각 토폴로지마다 3개의 종단간 연결상에 TCP 트래픽이 동시에 전송되고 하나의 물리적 링크를 공유하므로 직관적으로 볼 때, 전체 대역을 균일하게 나눠 전송될 것으로 기대된다. 평균 처리율은 약 32 Mbps로 3가지로 조합된 네트워크 토폴로지와 상관없이 거의 동일하다. 약간의 처리율에 있어서의 차이는 발생하기는 하지만 전체 성능에 크게 영향을 줄 정도는 아니다. 이러한 결과는 Layer3에서의logical router와 virtual router instance의 조합에 기반하여 가상화된 네트워크가 물리적 링크를 공유하여 다양한 토폴로지로 네트워크를 분리하는데 유용하게 사용될 수 있음을 의미한다. 다만, 각각의 네트워크 토폴로지에 따라 3개의 종단간 연결상의 TCP 트래픽이 균일하게 분배되지 않는 경우가 종종 발생한다. 최대 20Mbps 보다 낮은 처리율의 경우 5~12번 정도가 발생하는 것으로 각각의 그래프에서 관찰할 수 있고, 이러한 결과는 반복된 10번의 시도 중에 비슷한 패턴으로 나타났다. 특히, virtual router instance를 통한 가상화 보다 logical router를 기반으로 하는 경우 빈도 수가 잦은 것으로 관찰되었다. 이러한 부분은 물리적 라우터에서 logical router를 생성하는 최대 개수를 제한하는 부분과도 연관이 있을 것이라고 생각된다. 또한, 평균 처리율이 일반적으로 만족할만한 수준이라 하더라도 서로 편차가 큰 처리율이 자주 발생할 경우, 서로 독점적인 대역을 보장하기 위해 각 연결 당 추가적인 QoS 메커니즘이나 연결들 간의 스케줄링이 필요하다는 것을 시사한다.

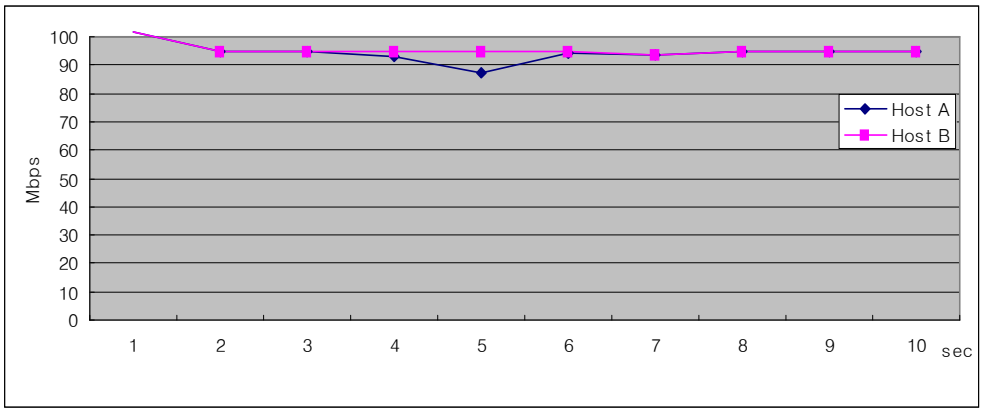

Figure 14 Logical router 기반 링크 독점 TCP 트래픽 처리율

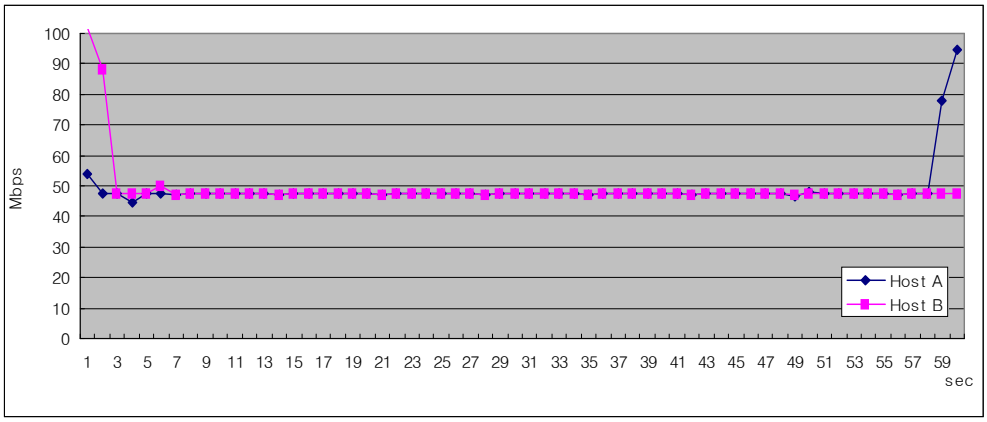

Figure 15 Logical router 기반 링크 공유 TCP 트래픽 처리율

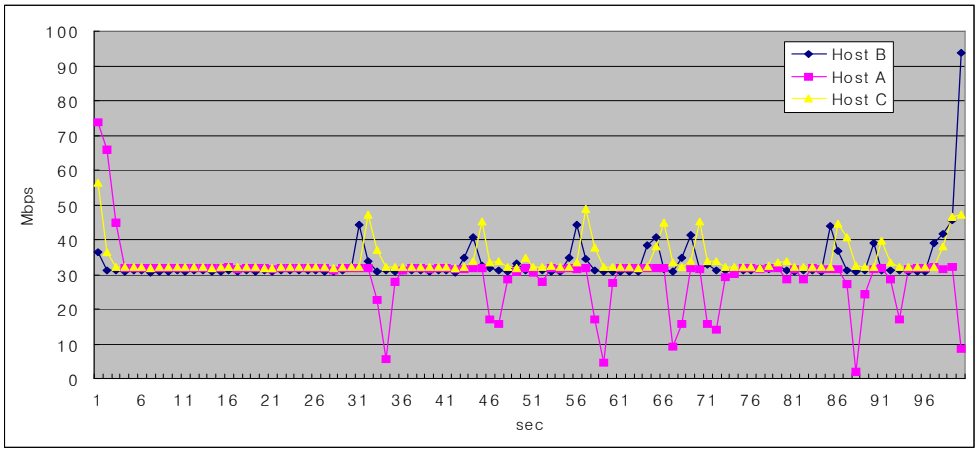

Figure 16 Figure 9에서 100초 동안의 TCP 트래픽 처리율

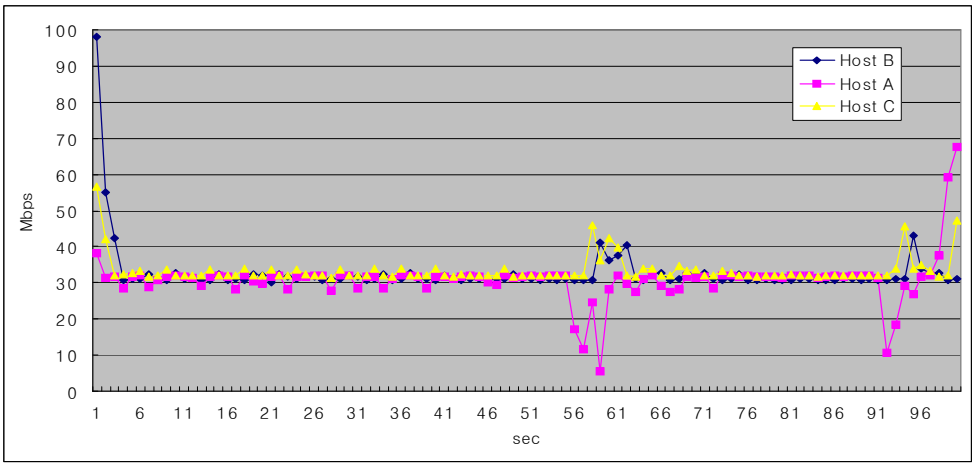

Figure 17 Figure 10에서 100초 동안의 TCP 트래픽 처리율

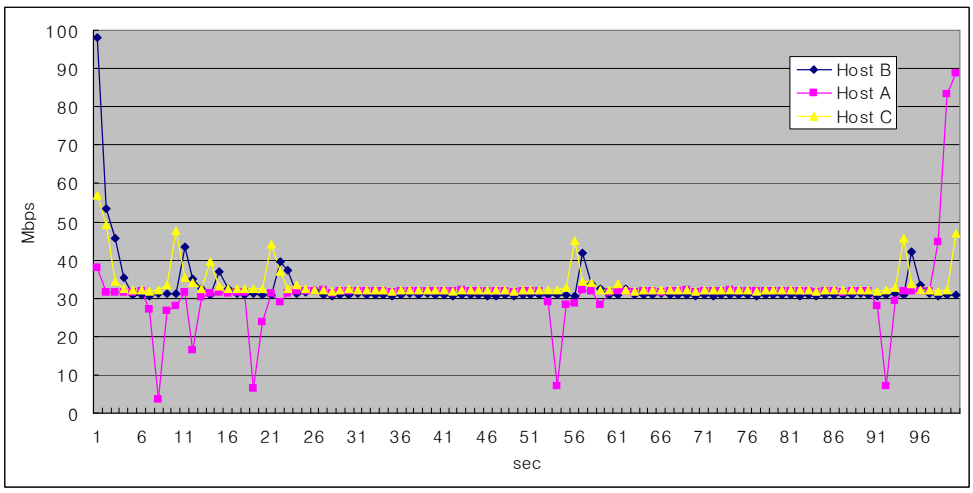

Figure 18 Figure 11에서 100초 동안의 TCP 트래픽 처리율

6. 결론

본 문서에서는 Juniper M5 라우터를 이용하여 virtual routing 제공 방법에 관해 조사 분석하였다. 대상 장비에서 지원하는 virtual routing 방법에는 크게 logical router와 virtual router instance를 이용하는 방법이 있다. Logical router는 라우터를 논리적으로 독립된 여러 개의 라우터로 분리하는 방법이고, virtual router instance는 보다 작은 범위의 개념으로 라우터의 기능을 일부를 에뮬레이션하는 방법이다. Logical router와 virtual router instance 모두 물리적 라우터의 추가적인 설치 없이 네트워크의 분리 및 유연성을 높이는데 유용하게 이용될 수 있다.

이러한 두 가지 기능을 실제 네트워크에 적용하는 방법론에는 라우터의 물리적 인터페이스를 전용으로 이용하는 방법과 공유하여 이용하는 방법이 있다. 전용으로 이용할 경우 각각의 종단간 트래픽 처리율은 인터페이스의 최대 성능을 충족시켰고, 공유할 경우는 논리적인 연결의 수에 따라 비교적 균일하게 분배되었다. 다만,

물리적 링크를 공유할 경우에는 약간의 편차가 발생하기도 하는데, 이러한 부분에 대해서는 네트워크 성능에 민감한 응용을 적용할 경우, 추가적인 트래픽 엔지니어링이 요구되는 부분이다. 또한, 본 실험은 최대 100Mbps의 링크를 대상으로 실험이 수행되었는데, 보다 확장을 위해서는 최소 1G 이상의 링크를 대상으로 실험이 이뤄져야 할 것으로 생각되고, 추가적으로 TCP/UDP 트래픽의 전송 패턴을 관찰할 필요성이 있을 것으로 생각된다.

Figure 17에서처럼 virtual routing 기법은 네트워크 장비의 규모 및 성능에 따라 Core network과 Access network에서 모두 사용될 수 있을 것이다. Layer 1 기반의 람다 네트워크에서 IP 멀티 캐스트 및 라우팅과 같은 Layer 3 기능이 요구되는 경우, virtual routing 기법은 Core network에서 다양하게 응용될 수 있을 것이다. 종단의 네트워크의 응용에 따라 물리적 인터페이스를 독점 혹은 공유하여 사용할 수 있을 것이고, 하나의 라우터에서 여러 virtual router(혹은 logical router)들을 생성함으로써, 여러 그룹별로 논리적으로 독립된 개별 네트워크를 생성할 수 있을 것이다.

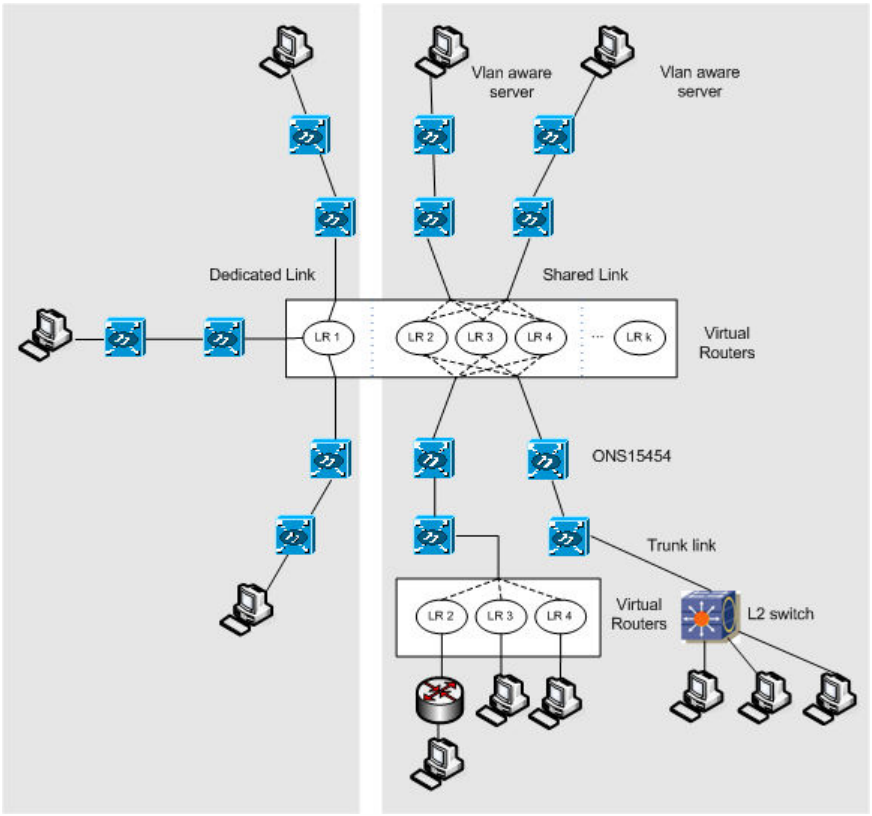

Figure 19 Virtual routing 적용 예

7. 참고문헌

[1] Steven Wallace, "Lambda Networking", http://www.anml.iu.edu/PDF/Lambda_Networking.pdf, 2002.

[2] Freek Dijkstra and Cees de Laat, "Optical Exchanges", Proceedings of GridNets2004, 2004.

[3] Bill St. Arnaud, "UCLP Roadmap: Web Services Workflow for Connecting Research Instruments and Sensors to Networks", http://www.canarie.ca/canet4/uclp/UCLP_Roadmap.doc, 2004.

[4] Jing Wu, Michel Savoie, Hanxi Zhang, Scott Campbell, Gregor v. Bochmann and Bill St. Arnaud, "Customer-Managed End-to-End Lightpath Provisioning", International Journal of Network Management. Vol. 15, No. 5, pp.349-362, 2005.

[5] Jing Wu, Hanxi Zhang, Scott Campbell, Michel Savoie, Gregor v. Bochmann and Bill St. Arnaud, "A Grid Oriented Lightpath Provisioning System", Proceedings of the 47th IEEE Global Telecommunications Conference (Globecom'04), Workshop on High Performance Global Grid Networks, pp.395-399, 2004.

[6] X. Zheng, M. Veeraraghavan, N. S. V. Rao, Q. Wu, and M. Zhu, "CHEETAH: Circuit-switched High-speed End-to-End Transport ArcHitecture testbed", IEEE Communication Magazine, Vol. 43, Issue 8, pp.s11-s17, 2005.

[7] Tom Lehman, Jerry Sobieski and Bijan Jabbari, "DRAGON: A Framework for Service Provisioning in Heterogenous Grid Networks", IEEE Communications Magazine, Vol. 44, Issue 3, 2006.

[8] GENI Project, "http://www.geni.net/"

[9] Larry Peterson, Tom Anderson, David Culler, and Timothy Roscoe, "A Blueprint for Introducing Disruptive Technology into the Internet", Proceedings of ACM HotNets-I Workshop, 2002

[10] Jon Turnet, "Virtualizing the Net - a strategy for network de-ossification", Keynote presentation at Hot Interconnects, 2004.

[11] Matt Kolon, "Intelligent logical router service", http://www.juniper.net/solutions/literature/white_papers/200097.pdf, 2004.

8. 부록 (M5라우터 설정파일)

본 절에서는 실험에서 사용된 토폴로지들의 설정 파일들을 제공한다.

- Figure 9의 설정 예

```
version "7.2I0 [builder]";
system {
    host-name R1;
    login {
        user admin {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password  "$1$14WDhCTy$dvDih3j.7iDX7VbJQp8p1."; ##
SECRET-DATA
            }
        }
    }
    services {
        ftp;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
    }
}
```

```
logical-routers {
    lr1 {
        interfaces {
            fe-0/1/0 {
                unit 1 {
                    vlan-id 102;
                    family inet {
                        address 10.21.2.2/24;
                    }
                }
            }
            fe-0/1/2 {
                unit 0 {
                    family inet {
                        address 10.11.3.1/24;
                    }
                }
            }
        }
        routing-options {
            static {
                route 10.11.1.0/24 next-hop 10.21.2.1;
            }
        }
    }
    lr2 {
        interfaces {
            fe-0/1/0 {
                unit 2 {
                    vlan-id 101;
                    family inet {
                        address 10.11.2.2/24;
                    }
                }
            }
```

```
            fe-0/1/3 {
                unit 0 {
                    family inet {
                        address 10.1.3.1/24;
                    }
                }
            }
        }
        routing-options {
            static {
                route 10.1.1.0/24 next-hop 10.11.2.1;
            }
        }
    }
    lr3 {
        interfaces {
            fe-0/1/0 {
                unit 3 {
                    vlan-id 100;
                    family inet {
                        address 10.1.2.2/24;
                    }
                }
            }
            fe-0/1/1 {
                unit 0 {
                    family inet {
                        address 10.31.3.1/24;
                    }
                }
            }
        }
        routing-options {
            static {
                route 10.31.1.0/24 next-hop 10.1.2.1;
            }
        }
    }
}
```

```
interfaces {
    fe-0/1/0 {
        vlan-tagging;
        unit 0 {
            vlan-id 103;
            family inet {
                address 10.31.2.2/24;
            }
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 203.230.7.104/27;
            }
        }
    }
}
routing-options {
    static {
        route 0.0.0.0/0 {
            qualified-next-hop 203.230.7.97;
        }
    }
}
```

- Figure 10의 설정 예

```
version "7.2I0 [builder]";
system {
    host-name R1;
    login {
        user admin {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password  "$1$14WDhCTy$dvDih3j.7iDX7VbJQp8p1."; ##
SECRET-DATA
            }
        }
    }
    services {
        ftp;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
    }
}
```

```
interfaces {
    fe-0/1/0 {
        vlan-tagging;
        unit 0 {
            vlan-id 103;
            family inet {
                address 10.31.2.2/24;
            }
        }
        unit 1 {
            vlan-id 102;
            family inet {
                address 10.21.2.2/24;
            }
        }
        unit 2 {
            vlan-id 101;
            family inet {
                address 10.11.2.2/24;
            }
        }
        unit 3 {
            vlan-id 100;
            family inet {
                address 10.1.2.2/24;
            }
        }
    }
    fe-0/1/1 {
        unit 0 {
            family inet {
                address 10.31.3.1/24;
            }
        }
    }
```

```
    fe-0/1/2 {
        unit 0 {
            family inet {
                address 10.11.3.1/24;
            }
        }
    }
    fe-0/1/3 {
        unit 0 {
            family inet {
                address 10.1.3.1/24;
            }
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 203.230.7.104/27;
            }
        }
    }
}
routing-options {
    static {
        route 0.0.0.0/0 {
            qualified-next-hop 203.230.7.97;
        }
        inactive: route 10.31.1.0/24 {
            qualified-next-hop 10.31.2.1;
        }
    }
}
```

```
routing-instances {
    blue {
        instance-type virtual-router;
        interface fe-0/1/2.0;
        interface fe-0/1/0.1;
        routing-options {
            static {
                route 10.11.1.0/24 next-hop 10.21.2.1;
            }
        }
    }
    red {
        instance-type virtual-router;
        interface fe-0/1/3.0;
        interface fe-0/1/0.2;
        routing-options {
            static {
                route 10.1.1.0/24 next-hop 10.11.2.1;
            }
        }
    }
    yellow {
        instance-type virtual-router;
        interface fe-0/1/1.0;
        interface fe-0/1/0.3;
        routing-options {
            static {
                route 10.31.1.0/24 next-hop 10.1.2.1;
            }
        }
    }
}
```

- Figure 11의 설정 예

```
version "7.2I0 [builder]";
system {
    host-name R1;
    login {
        user admin {
            uid 2000;
            class superuser;
            authentication {
                encrypted-password  "$1$14WDhCTy$dvDih3j.7iDX7VbJQp8p1."; ##
SECRET-DATA
            }
        }
    }
    services {
        ftp;
        telnet;
    }
    syslog {
        user * {
            any emergency;
        }
        file messages {
            any notice;
            authorization info;
        }
    }
}
```

```
logical-routers {
    lr1 {
        interfaces {
            fe-0/1/0 {
                unit 1 {
                    vlan-id 102;
                    family inet {
                        address 10.21.2.2/24;
                    }
                }
                unit 2 {
                    vlan-id 101;
                    family inet {
                        address 10.11.2.2/24;
                    }
                }
                unit 3 {
                    vlan-id 100;
                    family inet {
                        address 10.1.2.2/24;
                    }
                }
            }
            fe-0/1/1 {
                unit 0 {
                    family inet {
                        address 10.31.3.1/24;
                    }
                }
            }
            fe-0/1/2 {
                unit 0 {
                    family inet {
                        address 10.11.3.1/24;
                    }
                }
            }
```

```
            fe-0/1/3 {
                unit 0 {
                    family inet {
                        address 10.1.3.1/24;
                    }
                }
            }
        }
        routing-instances {
            blue {
                instance-type virtual-router;
                interface fe-0/1/2.0;
                interface fe-0/1/0.1;
                routing-options {
                    static {
                        route 10.11.1.0/24 next-hop 10.21.2.1;
                    }
                }
            }
            red {
                instance-type virtual-router;
                interface fe-0/1/3.0;
                interface fe-0/1/0.2;
                routing-options {
                    static {
                        route 10.1.1.0/24 next-hop 10.11.2.1;
                    }
                }
            }
            yellow {
                instance-type virtual-router;
                interface fe-0/1/1.0;
                interface fe-0/1/0.3;
                routing-options {
                    static {
                        route 10.31.1.0/24 next-hop 10.1.2.1;
                    }
                }
            }
        }
    }
}
```

```
interfaces {
    fe-0/1/0 {
        vlan-tagging;
        unit 0 {
            vlan-id 103;
            family inet {
                address 10.31.2.2/24;
            }
        }
    }
    fxp0 {
        unit 0 {
            family inet {
                address 203.230.7.104/27;
            }
        }
    }
}
routing-options {
    static {
        route 0.0.0.0/0 {
            qualified-next-hop 203.230.7.97;
        }
        inactive: route 10.31.1.0/24 {
            qualified-next-hop 10.31.2.1;
        }
    }
}
```

9. 부록 (PC라우터 설정방법)

9.1 Objective

이 절에서는 Linux Virtual Router를 Virtual Routing and Forwarding (VRF) for Linux를 이용하여 설치하고 사용하는 방법에 대하여 기술한다.

9.2 Scope

2.1 Linux Virtual Router를 설치한 하드웨어 구성
2.2 소프트웨어 구성
2.3 소프트웨어 설치방법
2.4 Linux Virtual Router 사용 방법

9.3 하드웨어 구성

3.1 CPU: Intel 2 X Xeon 2.8 GHz
3.2 NIC: Intel Pro/1000MT (or 1GbE Onboard)
3.3 RAM: 1GB
3.4 Board: *Intel 7505VB2 Server Board

* 참고: 1Gbps 이상의 성능을 위하여는 다음과 같은 하드웨어 구성에 대한 고려를 해야한다.

1. PCI bus speed: 64bits 66MHz or 100MHz PCI bus
2. Bus bottleneck: 여러 PCI slot들이 하나의 bus를 공유하도록 만들어진 보드가 있다. 이 경우 공유되는 bus에서 병목현상이 발생하여 충분한 성능을 내지 못할 수 도 있다.

9.4 필요한 소프트웨어 (http://sourceforge.net/projects/linux-vrf)

4.1 kernel-2.6.8-1-521vrf.src.rpm (vrf patched linux kernel)
4.2 iproute-2.4.7-14vrf.src.rpm
4.3 quagga 0.98.5 (routing software)

9.5 소프트웨어 설치방법 (순서대로 실행)

5.1 linux-vrf 소스코드

A. spec 파일들을 생성

```
#rpm -ihv kernel-2.6.8-1-521vrf.src.rpm
#rpm -ihv iproute-2.4.7-14vrf.src.rpm
#rpm -ihv linux-vrf.spec
```

* 위 명령은 kernel-2.6.sepc, iproute.spec, and linux-vrf.sepc을 /usr/src/redhat/SPECS 디렉토리에 생성한다.

B. 소스트리를 생성

```
#rpmbuild -bp --target=i686 kernel-2.6.spec
#rpmbuild -bp iproute.spec
#rpmbuild -bp linux-vrf.spec
```

* 위 명령의 실행결과 /usr/src/redhat/BUILD 디렉토리 하위에 아래와 같은 디렉토리가 생성된 것을 확인할 수 있다.

- iproute2
- kernel-2.6.8
- linux-vrf

C. linux-vrf 커널 컴파일 - /usr/src/redhat/BUILD/kernel-2.6.7/linux-2.6.8

* 커널 컴파일 방법에 대하여는 여기서 자세히 다루지는 않겠다.

1) 아래와 같은 커널 옵션이 선택되어있어야 한다.

```
Networking support --->
   Networking options --->
      <M> 802.1Q VLAN support

Networking support --->
   Networking options --->
      QoS and/or fair queueing --->
         [*] QoS and/or fair queueing
         ...
```

2) 커널 이미지를 생성하고 생성된 커널 이미지로 부팅한다.

5.2 iproute2 utility 생성 -/usr/src/redhat/BUILD/iproute2

A. rtnetlink.h 파일을 커널소스트리에서 /usr/include/linux 디렉토리로 복사한다.

```
#cp /usr/src/linux/include/rtnetlink.h /usr/include/linux
```

B. iproute2를 컴파일한다.

```
#make
```

C. /sbin 디렉토리로 ip 실행파일을 복사한다.
#cp /usr/src/redhat/BUILD/iproute2/ip /sbin

* 아래와 같이 실행하여 linux-vrf를 위한 ip utility가 잘 설치되었음을 확인할 수 있다.

```
# ip vrf show."
vrf 0
```

5.3 linux-vrf utils 생성 - /usr/src/redhat/BUILD/linux-vrf/utils

A. compile
#make

B. chvrf utilty를 /sbin 디렉토리로 복사
#cp /usr/src/redhat/BUILD/linux-vrf/utils/chvrf /sbin

9.6 Linux Virtual Router 사용방법

6.1 virtual router instance 생성과 삭제

```
ip vrf add 1
ip vrf remove 1
```

* This create/delete a virtual router instance 1
You can see virtual router instances by invoking "ip vrf show"

6.2 interface를 virtual router instance에 할당

```
ip link set eth0 vrf 1
```

* eth0 interface를 virtual router instance 1에 할당한다.
virtual router instance들에 할당된 interface들을 “ip link show" 명령으로 확인할 수 있다.

* interface를 virtual rotuer instance에 할당한 후에 interface의 ipv4 address를 설정해주어야 한다.

6.3 interface 설정하기

```
ip addr add 192.168.1.1/24 dev eth0
```

6.4 virtual router instance에 route 설정하기

ip route add 10.0.0.0/8 via 192.168.1.2 vrf 1

6.5 특정 virtual router instance에 프로세스 실행시키기

chvrf 1 ping 192.168.1.2

* virtual router instance 1에서 ping 프로세스가 실행된다.

6.6 an virtual interface 생성과 삭제

A. 생성: vconfig add eth0 2

B. 삭제: vconfig rem eth0.2

* 한 interface는 한 virtual router instance에만 할당되어질 수 있다. 하나의 physical interface를 여러 virtual router instance가 공유할 수 있도록 하기 위해서는 virtual interface를 생성하여 각기 다른 virtual router instance에 할당해 줄 수 있다.

9.7 참조

[1] http://sourceforge.net/projects/linux-vrf

[2] http://linux-vrf.sourceforge.net/