컴퓨터 네트워크

(Term project)

학과:컴퓨터공학과
학번:201312799
성명:김태형
제출 일자:2019/05/31
담당 교수님:서경룡 교수님

<목차>

1. Term Project 개요

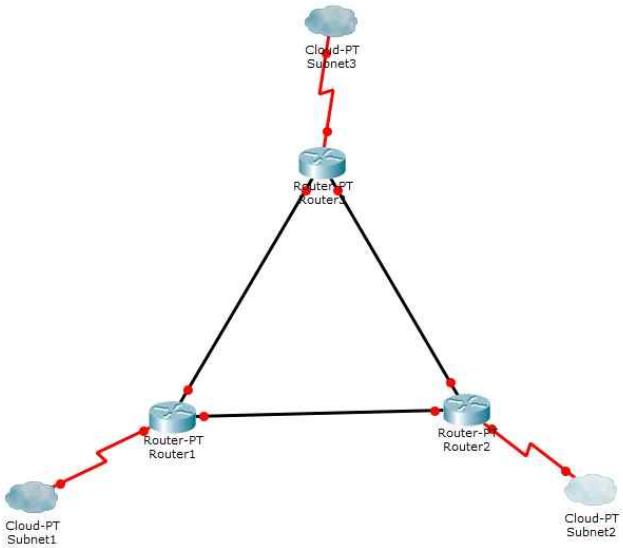
1-1 Term Project 목표

• 라우터로 연결된 소규모 네트워크를 구성하고 인터넷의 각종 서버를 구축하여 네트워크 동작을 확인한다.

• 구현방법은 실제 라우터가 가장 좋지만, 최적이나 현실적으로는 불가능하므로 각종 공개 프로그램을 활용하여 구현한다.

1) 구현 목표

• Packet Tracer를 이용하여 나타낸 간략도



1-2 개념

1) 컴퓨터 네트워크

• 컴퓨터 네트워크[1)]

-노드들이 자원을 공유할 수 있게 하는 디지털 전기통신망.

‣분산되어 있는 컴퓨터를 통신망으로 연결한 것.

2) 서브넷

• Subnet

-네트워크의 논리적인 분할

‣통상적으로,작고 단일한 물리적 네트워크를 말함.

-큰 네트워크가 작은 네트워크로 분할된 단위.

• Subnetting

-네트워크 세분화를 위한 IP 주소의 구성 변경.

---

1) 또는 컴퓨터망

‣IP 주소 체계가 2단계 구분방식인 것을 다시 3단계로 네트워크 세분화.
‣호스트 구분 ID에 할당된 비트들을 추가적으로 네트워크 구분 ID로 사용 가능.

[출처:: https://http://www.ktword.co.kr]

• Subnet Mask
-서브 네트워크를 만들기 위해 AND 비트 연산에 의해 씌우는 마스크.
‣TCP/IP 프로토콜에서 IP 주소체계로 네트워크를 나누는 논리적인 수단.

• 장점
-브로드캐스팅 영역크기를 작게함.
-주소 절약.
-라우팅 정보의 크기 감소.

3) 라우터

• 각기 독립된 네트워크들을 연결해 주는 장치.
-또는, 네트워크를 분할/구분시켜 연결하는 장치.

• 주요 기능
-네트워킹
‣각기 독립적으로 구성된 네트워크들을 연결해 줌.
-패킷 스위칭(Forwarding)
‣한 포트에서 패킷을 받아, 다른 포트로 전송.
-라우팅
‣라우터끼리 상호연결된 복잡한 망에서 경로의 배정 및 제어를 자동으로 수행.
-네트워크의 논리적 구조(Mao)를 습득(Learning)
‣이를 위해 이웃하는 라우터와 지속적으로 라우팅 정로를 교환.
-로드 밸런싱(Load Balancing)
‣라우터로부터 나오는 여러 케이블 선들의 Traffic양을 고루게 분산
-우회 경로
‣링크 중 하나가 고장나면 우회 경로를 구성함.

[출처:: https://http://www.ktword.co.kr]

2. 개발 환경

2.1 가상 머신

1) Virtual Machine

• 가상 머신(Virtual Machine, VM)은 컴퓨팅 환경을 소프트웨어로 구현한 것,
-즉 컴퓨터를 에뮬레이션하는 소프트웨어이다.

• 가상 컴퓨터 는 실제 컴퓨터의 기능을 제공하며 그 구현에는 특수 하드웨어, 소프트웨어 또는 그 조합이 포함될 수도 있다.

• 일반적으로 컨테이너라고 불리며 운영체제에 의해 컴퓨터의 리소스를 분할하여 사용자들에게 지원
-최종 사용자에게 실제 컴퓨터처럼 보이게 한다

2) Virtual Box

• VirtualBox 본래 이노테크가 개발한 뒤 현재는 오라클이 개발 중인 상용, 사유 소프트웨어
-리눅스, OS X, 솔라리스, 윈도를 게스트 윤영 체제로 하는 x86 가상화 소프트웨어

• VMWare 워크스테이션과 마이크로소프트 버추얼 PC와 같은 다른 상용 가상화 소프트웨어와 견주어 볼 때, VirtualBox는 기능이 부족한 편이지만 특별한 기능이 제공된다.
- 원격 데스크톱 프로토콜 (RDP), iSCSI 지원, RDP를 거치는 원격 장치의 USB 지원과 같이 원격으로 가상 컴퓨터를 제어하는 기능이 있다.
- VirtualBox는 인텔의 하드웨어 가상화 VT-x와 AMD의 AMD-V를 지원한다.

3) VMware

• VMware EMC 코퍼레이션의 공식 자회사이며, VMware 워크스테이션 및 VMware Plater를 포함한 x86 호환 컴퓨터를 위한 가상화 소프트웨어를 공급한다.

4) Hyper-V

• Hyper-V 마이크로소프트의 Hyper-V는 x64 시스템을 위한 하이퍼 바이저 기반의 가상화 시스템이다.
-윈도우 서버 가상화라는 이름으로도 알려져 있다.

• Hyper-V는 리눅스 게스트 운영 체제에 대한 기본적인 가상화 지원을 제공한다.
-반 가상화 지원은 리눅스 통합 구성 요소와 SatoriInputVSC 드라이버를 설치함으로써 사용할 수 있다.
-2009년 7월 20일에 마이크로소프트는 리눅스 커널에 들어간 이 세 개의 드라이버들을 GPL로 공개하여 2.6.32 이상의 커널은 자체적으로 Hyper-V 반 가상화 지원을 포함

5) Parallels

• 패러렐즈(Parallels, Inc)는 미국, 독일, 영국, 프랑스, 일본, 중국, 러시아, 우크라이나에 사무소를 둔 사유 가상화 기술 회사

• 페러럴즈 데스크톱 5 기준으로 다양한 32비트 및 64비트 x86 운영 체제의 지원을 포함

6) 비교

구분	Virtual Box	VMware	Hyper-V	Parallels
Host OS	Windows/Linux/Mac/Solaris/FreeBSD	Windows/Linux/Mac	Windows/Linux/FreeBSD	Windows/Linux/Mac
Guest OS	DOS/Windows/Linux/Solaris/FreeBSD	DOS/Windows/Linux/Solaris/FreeBSD	Windows/Linux/	DOS/Windows/Linux/OS2
제공사	Oracle	VMware	Microsoft	Parallels IP Holdings
라이센스	무료(오픈소스)	유료(49~79$)	평가용-무료	쉐어웨어/상용(79~189$)

[참고:https://www.wikipedia.org/]

2.2 설치 및 설정

1) Virtual Box 설치

• 대부분의 OS를 사용 가능하며 게스트 확장으로 편의성이 좋음

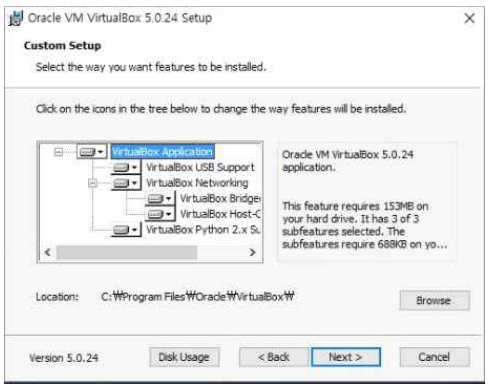
• Oracle이 오픈소스이며 다른 VM보다 설치 용량이 작고 간단함

• Virtual Box를 https://www.virtualbox.org/에서 다운로드.

-Virtual Box 6.0.24는 Ubuntu 18.04 Lts와 오류가 있어서 5.0.24를 설치.

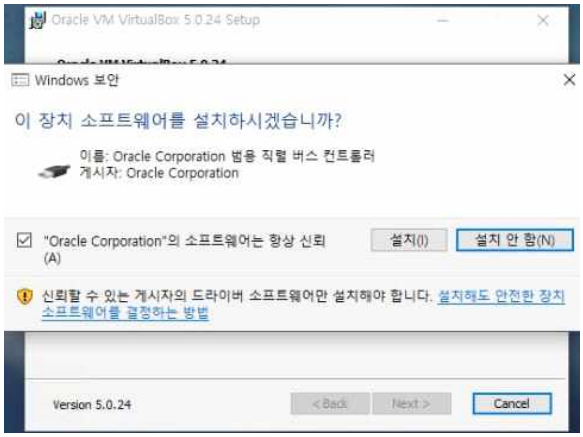
• 설치 과정

›Next>



›Next>

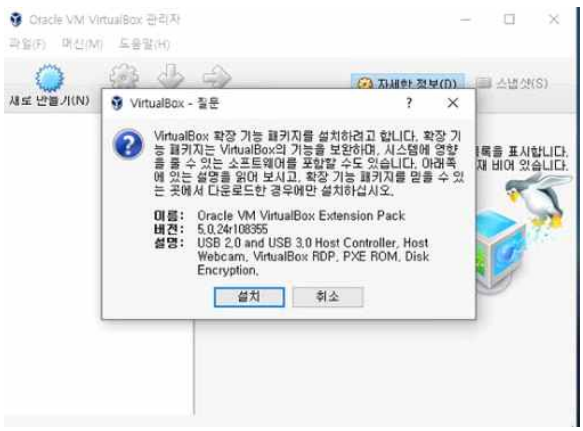
›운영체제와 Virtual Box간
가상 네트워크 구성을 설정하면서
네트워크가 중단될 수 있음.



‣Virtual Box와 Host 운영체제간 하드웨어 리소스 공유를 하기 위한 드라이버 설치.

‣설치완료

• Virtual Box Extnsion Pack을 https://www.virtualbox.org/에서 다운로드.



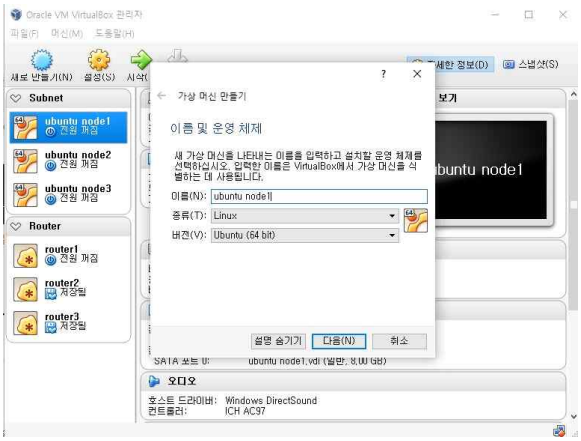
‣Pack을 다운로드 후 확장팩을 실행하면 Virtual Box에서 자동으로 창이 뜸.

• Virtual Box 설치 완료

2) Linux 설치

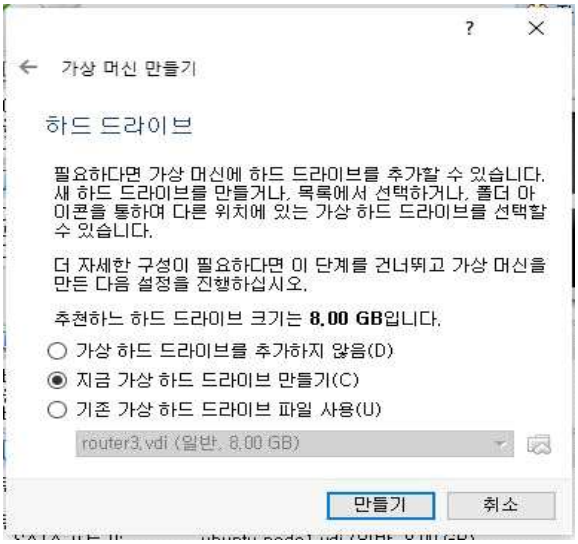
• Ubuntu Linux 18.04.2LTS amd64를 https://www.ubuntu.com/에서 다운로드.

• Virtual Box에 Ubuntu를 설치할 컴퓨터 만들기

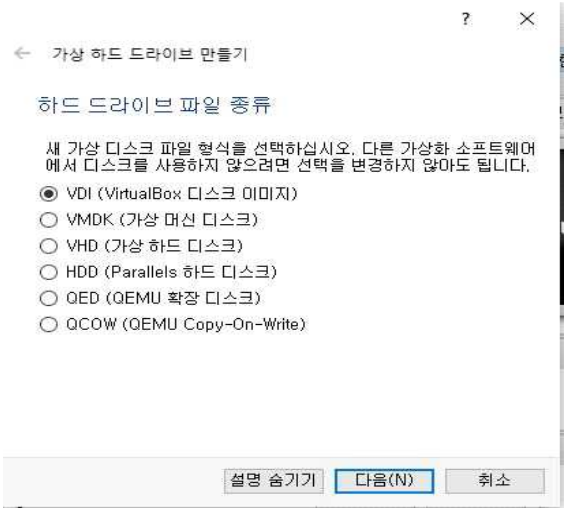


‣새로 만들기를 누르고 종류에 Ubuntu를 적으면 자동으로 Linux로 탭이 바뀐다. 버전은 자신이 사용할 버전을 선택하면 된다.

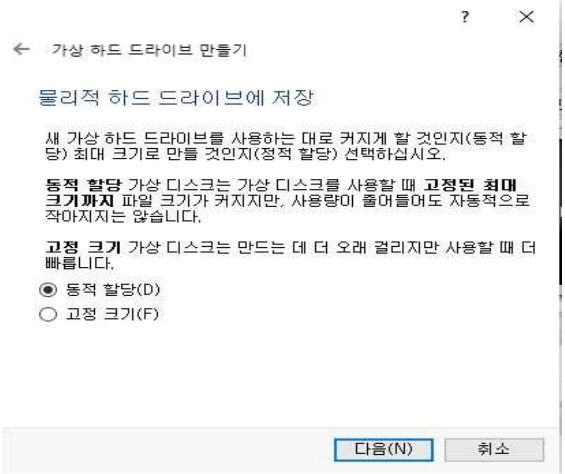
‣Ubuntu Linux는 2GB의 램이 권장사양이므로 2048MB로 설정.



‣기존 선택되어 있는 것으로 만들기.



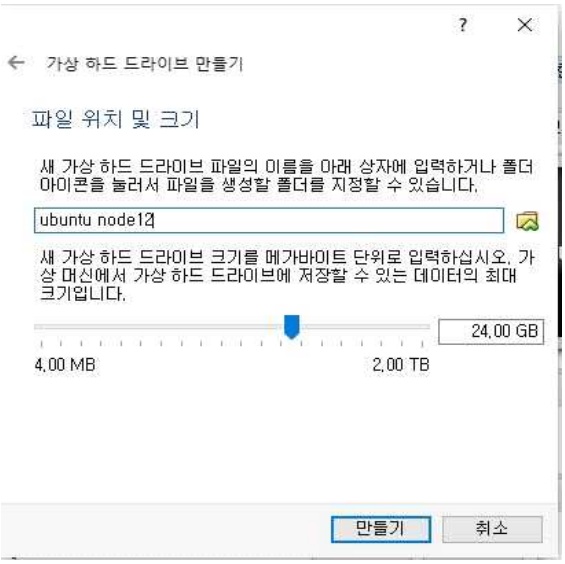
‣다음 선택



‣추가적인 디스크가 필요한데 사용량이 확실하지 않으면 동적 할당 선택.

‣필요한 디스크가 얼만지 정확하게 알고 있다면 고정 크기 사용.

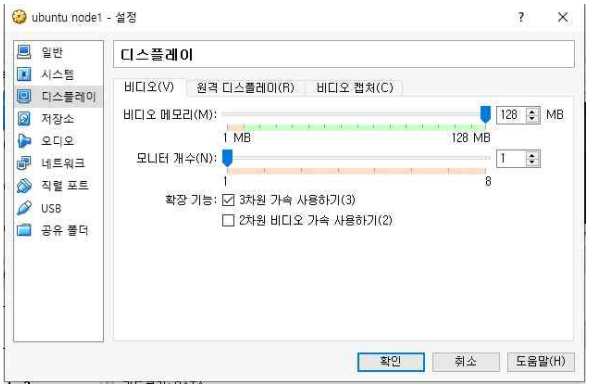
‣속도는 고정 크기가 더 빠름.



‣Ubuntu Linux의 권장 디스크량은 25GB이지만 이 가상 머신은 Term Project 용이기 때문에 조금 부족하게 해도 괜찮음.

‣추가적인 사용이 있거나 디스크가 더 필요하면 자유롭게 설정

• Ubuntu 설치 전 설정 변경



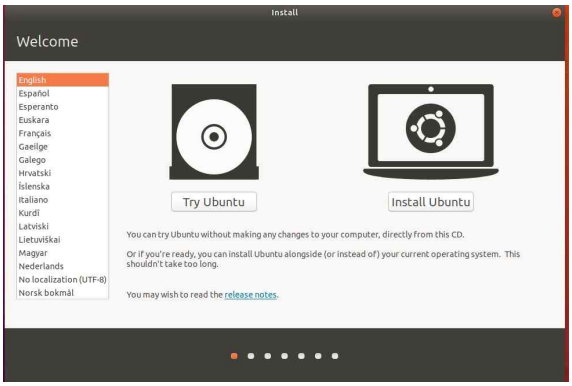
‣디스플레이 탭에서 비디오 메모리를 128MB로 변경
-가상 머신의 그래픽 기능을 위해 사용할 비디오 메모리 크기

‣확장기능에서 3차원 가속 사용하기 체크
-호스트 컴퓨터가 가능하면 체크

‣저장소에서 컨트롤러:IDE에서 CD모양 클릭

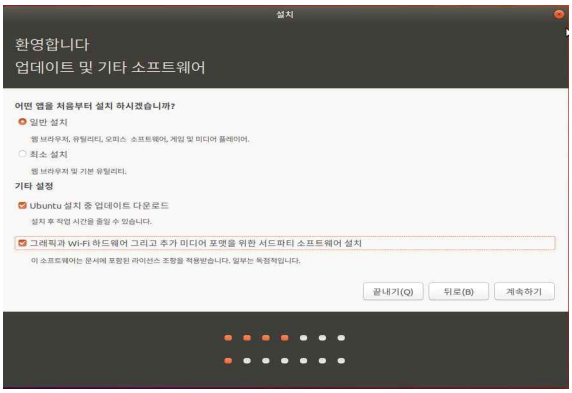
‣디스크 선택하기 누른 후 다운로드한 Ubuntu.iso 선택

• Ubuntu 설치

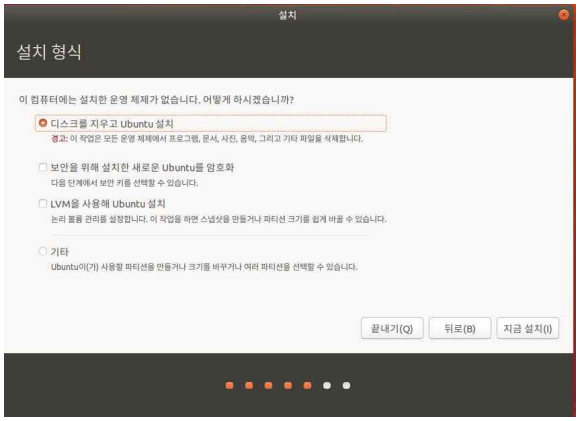


‣가상머신을 실행

‣원하는 언어선택 후 설치

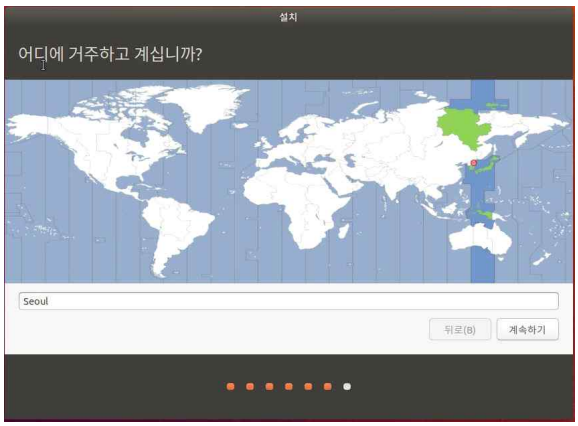


‣일반 설치와 업데이트, 서드파티 소프트웨어 설치



‣디스크를 지우고 설치를 클릭

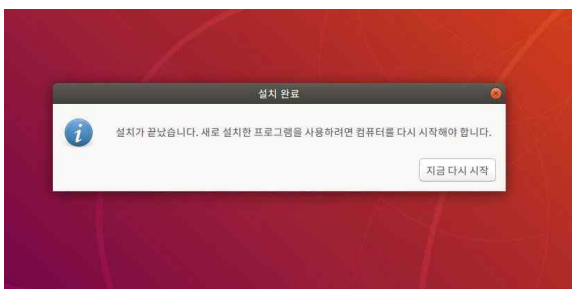
‣듀얼 부팅을 하려면
기타 후 분할 한 디스크 선택



‣거주지 선택

‣이름과 암호 설정

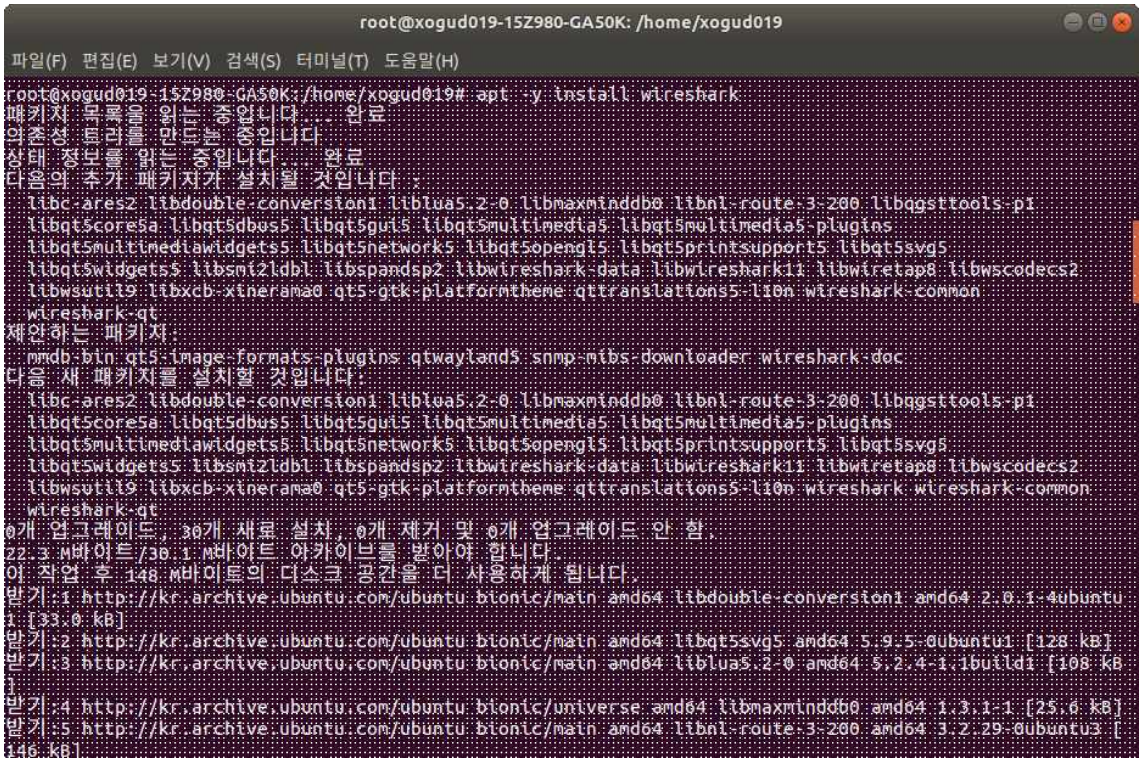
‣여기까지 하면 자동으로 설치가
진행된다.



‣설치 완료 후
지금 다시 시작을 누르면
재부팅 후 Ubuntu가 실행된다.

• Packet 이동 확인을 위해 WireShark를 설치한다.

‣sudo apt-get -y install wireshark

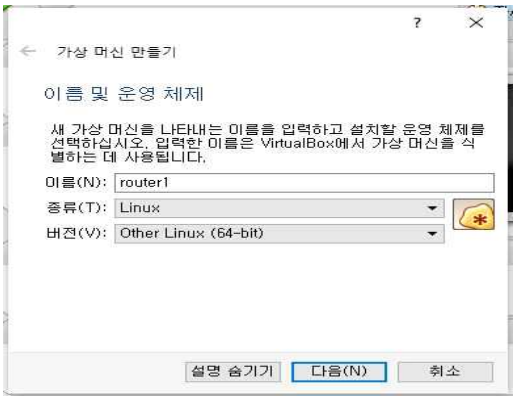


3) Vyos 설치

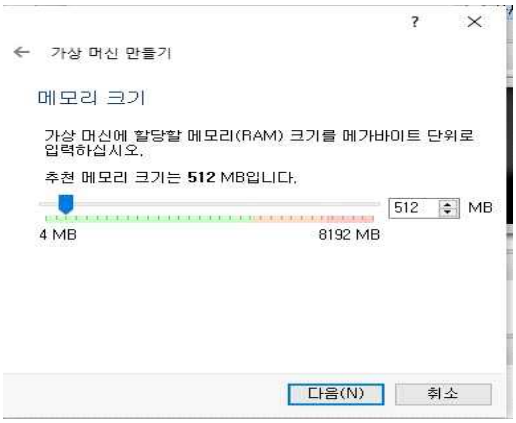
• 현재는 Brocade에 매각된 Vyatta를 기반으로 하는 라우터 소프트웨어 오픈소스
• 리눅스 기반이라 리눅스 명령어를 포함하여 리눅스 시스템을 그대로 활용가능
-GUI가 없어서 편의성은 떨어지지만 성능이 상당히 좋고,
-사용처 또한 엔터프라이즈 급이기 때문에 활용도가 좋다

• Vyos 1.2.0 amd64를 https://vyos.io/에서 다운로드

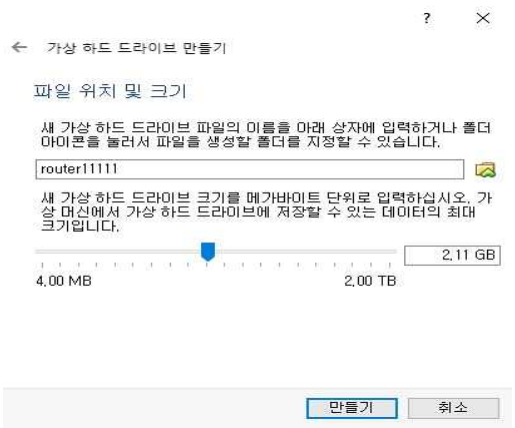
• 설치



‣만들기 선택 후 버전을 Other Linux로 설정



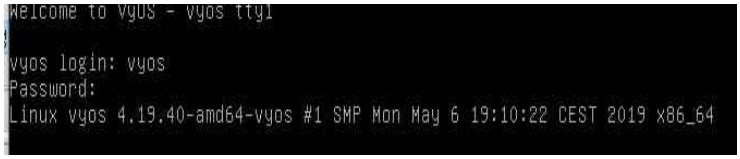
‣Vyos의 권장 Ram은 512MB이므로 설정



‣Vyos의 권장 Disk는 2GB이므로 설정

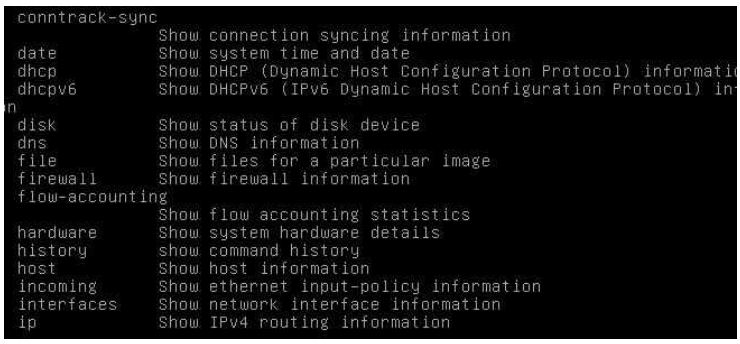
‣이 외에 설정은 Ubuntu Linux 설정과 동일하다.

‣디스크 삽입 후 실행



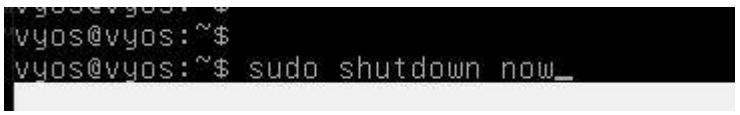
‣로그인 default 값은 vyos/vyos이다.

‣로그인 후 install image 입력

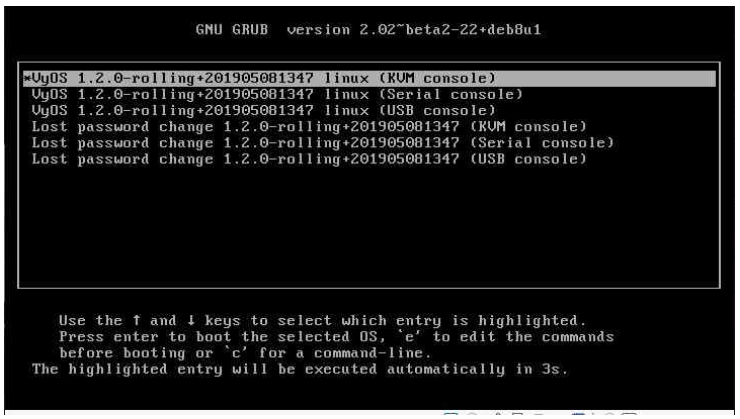


‣설치가 완료되면 다음과 같이 s[Tab], show [Tab], configure를 입력해서 다음과 창이 뜨면 설치 완료

‣설정 변경 시 항상 configure 상태에서 하고 끝난 뒤 commit, save



‣다음 명령어를 이용, 시스템 종료



‣가상 머신 실행 전 아까 삽입한 디스크 제거 후 실행

‣다음과 같은 화면이 나오면 성공

• 총 세 대의 Ubuntu와 세 대의 Vyos를 설치함

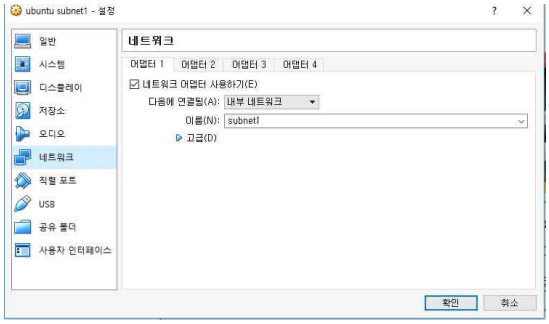
4) 기타설정

• Ubuntu의 네트워크 설정 변경

-네트워크 NAT->내부 네트워크 변경

›내부 네트워크 전환 시 인터넷이 사용 불가

›필요한 Package는 NAT 상태에서 다운로드

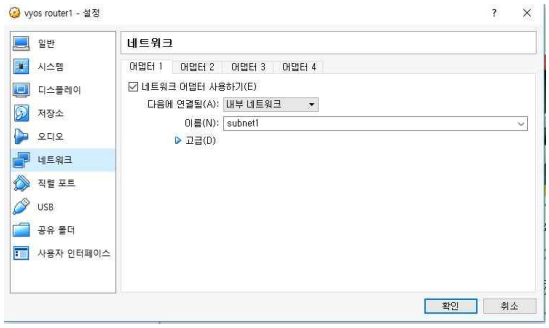


›Ubuntu Node1 설정

›어뎁터 이름은 연결하고자 하는 Router의 어뎁터 이름과 일치해야함

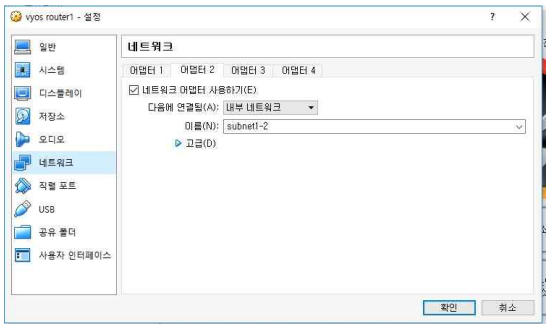
• Vyos 네트워크 설정 변경

-3개의 어뎁터를 설정

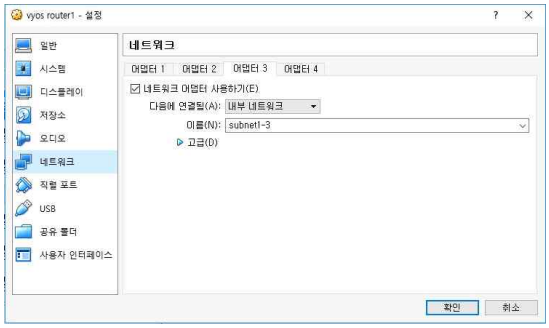


›Router1-adp1

›어뎁터 이름은 연결하는 다른 Node의 어뎁터 명과 일치해야함



›Router1-adp2

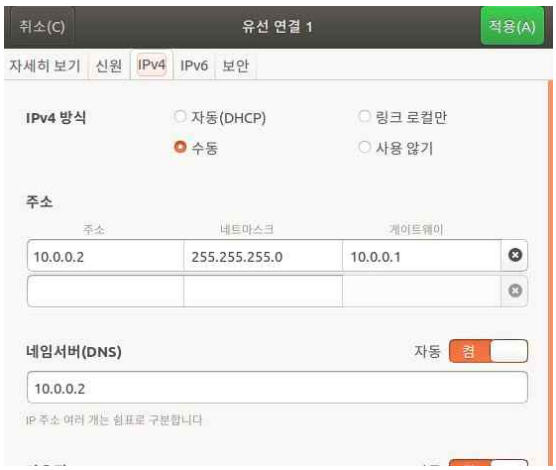


›Router1-adp3

• 남은 2개의 Ubuntu와 2개의 Vyos도 내부 네트워크로 설정 변경함

• Ubuntu IP 할당

-Ubuntu 컴퓨터에 내부 네트워크에서 사용할 IP주소, 마스크 그리고 게이트웨이를 할당해야한다.



‣네트워크 유선 관리에 들어간 후 자동에서 수동으로 변경

‣주소에 IP, 네트마스크에24(혹은 255.255.255.0), 게이트웨이에 연결할 Router 주소 입력

‣네임서버는 자기 자신의 IP 입력

-DNS Server 설치 후 변경

• 남은 2대의 Ubuntu도 설정 변경

• Vyos IP 할당

-Router의 NIC[2]에 IP를 할당해야한다.

```
vyos@vyos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                        S/L  Description
---------        ----------                        ---  -----------
eth0             10.0.0.1/24                       u/u
eth1             100.0.0.1/24                      u/u
eth2             200.0.0.2/24                      u/u
lo               127.0.0.1/8                       u/u
                 ::1/128
```

‣set interfaces ehternet eth" " address " "

-첫 번째 “ ” 안에는 eth 번호 0~2

-두 번째 “ ” 안에는 ‘ip/24'를 입력

‣입력 완료 후 반드시 commit, save

‣show interfaces를 입력해 확인

<Router 1>

• 남은 2대의 Vyos도 설정 변경

```
vyos@vyos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                        S/L  Description
---------        ----------                        ---  -----------
eth0             20.0.0.1/24                       u/u
eth1             200.0.0.1/24                      u/u
eth2             103.0.0.2/24                      u/u
lo               127.0.0.1/8                       u/u
                 ::1/128
```

<Router 1-2>

```
vyos@vyos:~$ show interfaces
Codes: S - State, L - Link, u - Up, D - Down, A - Admin Down
Interface        IP Address                        S/L  Description
---------        ----------                        ---  -----------
eth0             30.0.0.1/24                       u/u
eth1             100.0.0.2/24                      u/u
eth2             103.0.0.1/24                      u/u
lo               127.0.0.1/8                       u/u
                 ::1/128
```

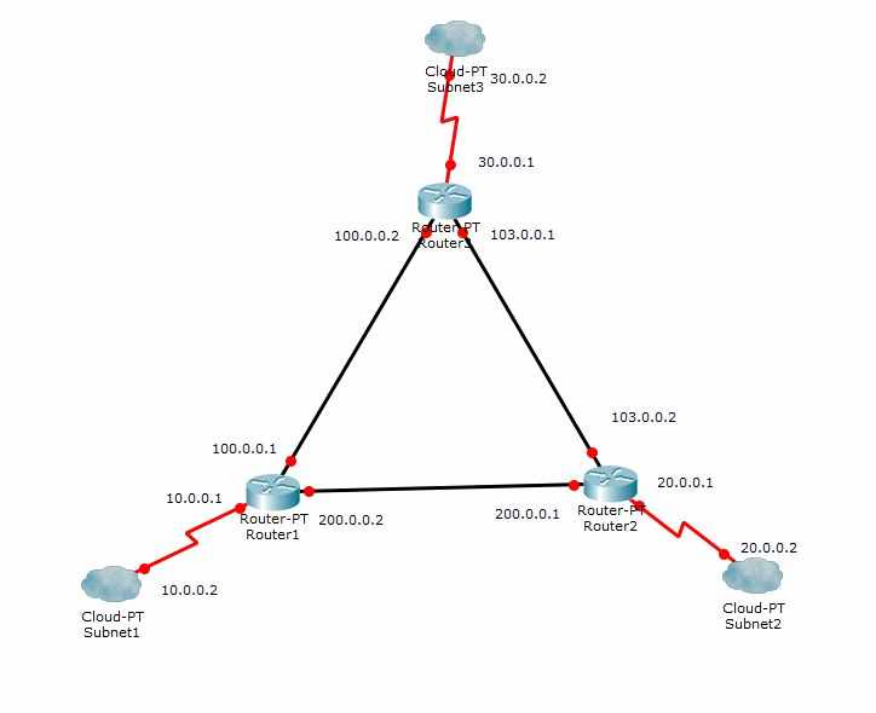
<Router 1-3>

2) Network Interface Card

3. 설계 및 구현

3.1 구성도

• 현재 만들어진 구성



3.2 네트워크 연결

• 라우터와 서브넷간의 연결은 완료됨.

• 라우터 간 연결을 위한 설정을 해야한다.

-Static, RIP, OSPF, BGP를 사용

1)Routing Protocol

• '라우터 간에 라우팅 정보의 교환' 및 '라우팅 테이블'의 유지관리를 동적으로 수행하는 프로토콜

• 구분

-Distance Vertor Routing Protocol

‣RIP, IGRP, EIGRP

-Link State Routing Protocols

‣OSPF, IS-IS

-AS 내부 라우터 간 : IGP

‣RIP, OSPF, IS-IS

-AS 외부 라우터 상호 간 : EGP

‣EGP, BGP, IDRP

2) Static

• 네트워크 관리자가 라우터를 직접 목적지 별로 라우팅을 지정해주는 방식

-주로 외부 네트워크와 연결되는 경로가 하나인 Stub Network에서 많이 사용함.

›set protocols static route " " next-hop " " distance '1'

-첫 번째 “ ” 안에는 목적지의 Route 주소,

-두 번째 “ ” 안에는 거쳐 갈 hop의 주소

›예를 들어 Router 1은

-set protocols static route 20.0.0.0/24 next-hop 200.0.0.1 distance '1'

-set protocols static route 30.0.0.0/24 next-hop 100.0.0.2 distance '1'

• 총 3대의 Vyos Router를 설정해야함

›show protocols 을 통해 확인

```
vyos@vyos:~$ configure
[edit]
vyos@vyos# show protocols
 static {
     route 20.0.0.0/24 {
         next-hop 200.0.0.1 {
             distance 1
         }
     }
     route 30.0.0.0/24 {
         next-hop 100.0.0.2 {
             distance 1
         }
     }
 }
```

<Router 1>

```
vyos@vyos# show protocols
 static {
     route 10.0.0.0/24 {
         next-hop 200.0.0.2 {
             distance 1
         }
     }
     route 30.0.0.0/24 {
         next-hop 103.0.0.1 {
             distance 1
         }
     }
 }
```

<Router 1-2>

```
vyos@vyos:~$ configure
[edit]
vyos@vyos# show protocols
 static {
     route 10.0.0.0/24 {
         next-hop 100.0.0.1 {
             distance 1
         }
     }
     route 20.0.0.0/24 {
         next-hop 103.0.0.2 {
             distance 1
         }
     }
 }
```

<Router 1-3>

• ping 을 통한 확인

‣Node 1 -> Node 2, Node 3

```
node1@node1-VirtualBox:~$ ping 20.0.0.2
PING 20.0.0.2 (20.0.0.2) 56(84) bytes of data.
64 bytes from 20.0.0.2: icmp_seq=1 ttl=62 time=1.09 ms
64 bytes from 20.0.0.2: icmp_seq=2 ttl=62 time=1.49 ms
64 bytes from 20.0.0.2: icmp_seq=3 ttl=62 time=0.892 ms
64 bytes from 20.0.0.2: icmp_seq=4 ttl=62 time=1.07 ms
^C
--- 20.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3009ms
rtt min/avg/max/mdev = 0.892/1.138/1.492/0.219 ms
node1@node1-VirtualBox:~$ ping 30.0.0.2
PING 30.0.0.2 (30.0.0.2) 56(84) bytes of data.
64 bytes from 30.0.0.2: icmp_seq=1 ttl=62 time=1.20 ms
64 bytes from 30.0.0.2: icmp_seq=2 ttl=62 time=1.60 ms
64 bytes from 30.0.0.2: icmp_seq=3 ttl=62 time=1.59 ms
64 bytes from 30.0.0.2: icmp_seq=4 ttl=62 time=1.51 ms
^C
--- 30.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3009ms
rtt min/avg/max/mdev = 1.200/1.478/1.600/0.163 ms
```

• Wireshark 를 통한 확인

‣Node 1 -> Node 2

1	0.000000000	10.0.0.2	20.0.0.2	ICMP	98 Echo (ping) request id=0x0761, seq=1/256, ttl=64
2	0.001133954	20.0.0.2	10.0.0.2	ICMP	98 Echo (ping) reply id=0x0761, seq=1/256, ttl=62
3	1.001934508	10.0.0.2	20.0.0.2	ICMP	98 Echo (ping) request id=0x0761, seq=2/512, ttl=64
4	1.003330347	20.0.0.2	10.0.0.2	ICMP	98 Echo (ping) reply id=0x0761, seq=2/512, ttl=62
5	2.004665244	10.0.0.2	20.0.0.2	ICMP	98 Echo (ping) request id=0x0761, seq=3/768, ttl=64
6	2.006184232	20.0.0.2	10.0.0.2	ICMP	98 Echo (ping) reply id=0x0761, seq=3/768, ttl=62
7	3.007812380	10.0.0.2	20.0.0.2	ICMP	98 Echo (ping) request id=0x0761, seq=4/1024, ttl=64
8	3.010183471	20.0.0.2	10.0.0.2	ICMP	98 Echo (ping) reply id=0x0761, seq=4/1024, ttl=62
9	4.011051149	10.0.0.2	20.0.0.2	ICMP	98 Echo (ping) request id=0x0761, seq=5/1280, ttl=64
10	4.012418898	20.0.0.2	10.0.0.2	ICMP	98 Echo (ping) reply id=0x0761, seq=5/1280, ttl=62
11	5.012777020	10.0.0.2	20.0.0.2	ICMP	98 Echo (ping) request id=0x0761, seq=6/1536, ttl=64
12	5.014318129	20.0.0.2	10.0.0.2	ICMP	98 Echo (ping) reply id=0x0761, seq=6/1536, ttl=62

‣Node 1 -> Node 3

20	47.609965162	30.0.0.2	10.0.0.2	ICMP	98 Echo (ping) reply id=0x0766, seq=1/256, ttl=62 (
21	48.612673281	10.0.0.2	30.0.0.2	ICMP	98 Echo (ping) request id=0x0766, seq=2/512, ttl=64 (
22	48.614546371	30.0.0.2	10.0.0.2	ICMP	98 Echo (ping) reply id=0x0766, seq=2/512, ttl=62 (
23	49.613667189	10.0.0.2	30.0.0.2	ICMP	98 Echo (ping) request id=0x0766, seq=3/768, ttl=64 (
24	49.615054352	30.0.0.2	10.0.0.2	ICMP	98 Echo (ping) reply id=0x0766, seq=3/768, ttl=62 (
25	50.615090464	10.0.0.2	30.0.0.2	ICMP	98 Echo (ping) request id=0x0766, seq=4/1024, ttl=64
26	50.616500224	30.0.0.2	10.0.0.2	ICMP	98 Echo (ping) reply id=0x0766, seq=4/1024, ttl=62
27	51.617741938	10.0.0.2	30.0.0.2	ICMP	98 Echo (ping) request id=0x0766, seq=5/1280, ttl=64
28	51.618688502	30.0.0.2	10.0.0.2	ICMP	98 Echo (ping) reply id=0x0766, seq=5/1280, ttl=62
29	52.619574818	10.0.0.2	30.0.0.2	ICMP	98 Echo (ping) request id=0x0766, seq=6/1536, ttl=64
30	52.621393946	30.0.0.2	10.0.0.2	ICMP	98 Echo (ping) reply id=0x0766, seq=6/1536, ttl=62
31	52.634039783	PcsCompu_53:1b:ee	PcsCompu_1a:3f:86	ARP	42 Who has 10.0.0.1? Tell 10.0.0.2

• RIP를 위해 Static Routing을 삭제

‣delete protocols static route

```
vyos@vyos# delete protocols static route

  Nothing to delete (the specified node does not exist)

[edit]
vyos@vyos# commit
[edit]
vyos@vyos# save
Saving configuration to '/config/config.boot'...
Done
[edit]
vyos@vyos# show protocols
 static {
 }
[edit]
```

3) RIP

• DV 라우팅 프로토콜

-거리(hop의 개수)와 (hop의) 방향으로 길을 찾아가는 프로토콜

-Hop Count 만을 보고 경로를 결정

• 30초마다 주기적으로 인접 라우터에게 자신의 정보를 광고한다.

• 동적 라우팅 프로토콜이므로 어떤 최단 경로가 차단되면 우회한다.

‣ `set protocols rip network " "`

-" " 안에 목적지 IP 주소 입력

‣ 예를 들어 Router 1은

-`set protocols rip network 10.0.0.0/24`

-`set protocols rip network 100.0.0.0/24`

-`set protocols rip network 200.0.0.0/24`

‣ `show protocols` 을 통해 확인

```
vyos@vyos# show protocols
 rip {
     network 10.0.0.0/24
     network 100.0.0.0/24
     network 200.0.0.0/24
 }
 static {
 }
```

<Router 1>

```
vyos@vyos# show protocols
 rip {
     network 20.0.0.0/24
     network 200.0.0.0/24
     network 103.0.0.0/24
 }
 static {
 }
```

<Router 1-2>

```
vyos@vyos# show protocols
 rip {
     network 30.0.0.0/24
     network 100.0.0.0/24
     network 103.0.0.0/24
 }
 static {
 }
```

<Router 1-3>

```
vyos@vyos:~$ show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

C>* 10.0.0.0/24 is directly connected, eth0, 00:34:58
R>* 20.0.0.0/24 [120/2] via 200.0.0.1, eth2, 00:01:58
R>* 30.0.0.0/24 [120/2] via 100.0.0.2, eth1, 00:00:35
C>* 100.0.0.0/24 is directly connected, eth1, 00:34:55
R>* 103.0.0.0/24 [120/2] via 200.0.0.1, eth2, 00:01:58
C>* 200.0.0.0/24 is directly connected, eth2, 00:34:57
```

<Router 1 Router Table>

• ping 을 통한 확인

‣Node 2 -> Node 1, Node 3

```
node2@node2-VirtualBox:~$ ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=62 time=1.09 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=62 time=1.58 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=62 time=2.03 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=62 time=1.72 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=62 time=1.73 ms
^C
--- 10.0.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4008ms
rtt min/avg/max/mdev = 1.092/1.634/2.032/0.309 ms
node2@node2-VirtualBox:~$ ping 30.0.0.2
PING 30.0.0.2 (30.0.0.2) 56(84) bytes of data.
64 bytes from 30.0.0.2: icmp_seq=1 ttl=62 time=1.18 ms
64 bytes from 30.0.0.2: icmp_seq=2 ttl=62 time=1.69 ms
64 bytes from 30.0.0.2: icmp_seq=3 ttl=62 time=1.69 ms
64 bytes from 30.0.0.2: icmp_seq=4 ttl=62 time=2.31 ms
64 bytes from 30.0.0.2: icmp_seq=5 ttl=62 time=1.64 ms
^C
--- 30.0.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4008ms
rtt min/avg/max/mdev = 1.185/1.706/2.317/0.361 ms
```

• Wireshark 를 통한 확인

‣Node 2 -> Node 1

25	12.034021501	20.0.0.2	10.0.0.2	ICMP	98	Echo (ping) request id=0x068f, seq=20/5120, tt
26	12.035522444	10.0.0.2	20.0.0.2	ICMP	98	Echo (ping) reply id=0x068f, seq=20/5120, tt
27	13.038637443	20.0.0.2	10.0.0.2	ICMP	98	Echo (ping) request id=0x068f, seq=21/5376, tt
28	13.040162415	10.0.0.2	20.0.0.2	ICMP	98	Echo (ping) reply id=0x068f, seq=21/5376, tt
29	13.943540956	103.0.0.1	224.0.0.9	RIPv2	126	Response
30	13.943869731	103.0.0.1	224.0.0.9	RIPv2	126	Response
31	14.039462649	20.0.0.2	10.0.0.2	ICMP	98	Echo (ping) request id=0x068f, seq=22/5632, tt
32	14.041189420	10.0.0.2	20.0.0.2	ICMP	98	Echo (ping) reply id=0x068f, seq=22/5632, tt
33	14.108291146	20.0.0.1	224.0.0.9	RIPv2	146	Response
34	14.108303850	20.0.0.1	224.0.0.9	RIPv2	146	Response
35	14.108307187	20.0.0.1	224.0.0.9	RIPv2	146	Response
36	15.041097479	20.0.0.2	10.0.0.2	ICMP	98	Echo (ping) request id=0x068f, seq=23/5888, tt

• OSPF를 위해 RIP Routing을 삭제

‣delete protocols RIP

```
vyos@vyos# delet protocols rip
[edit]
vyos@vyos# commit
[edit]
vyos@vyos# save
Saving configuration to '/config/config.boot'...
Done
[edit]
vyos@vyos# show protocols
 static {
 }
```

4) OSPF

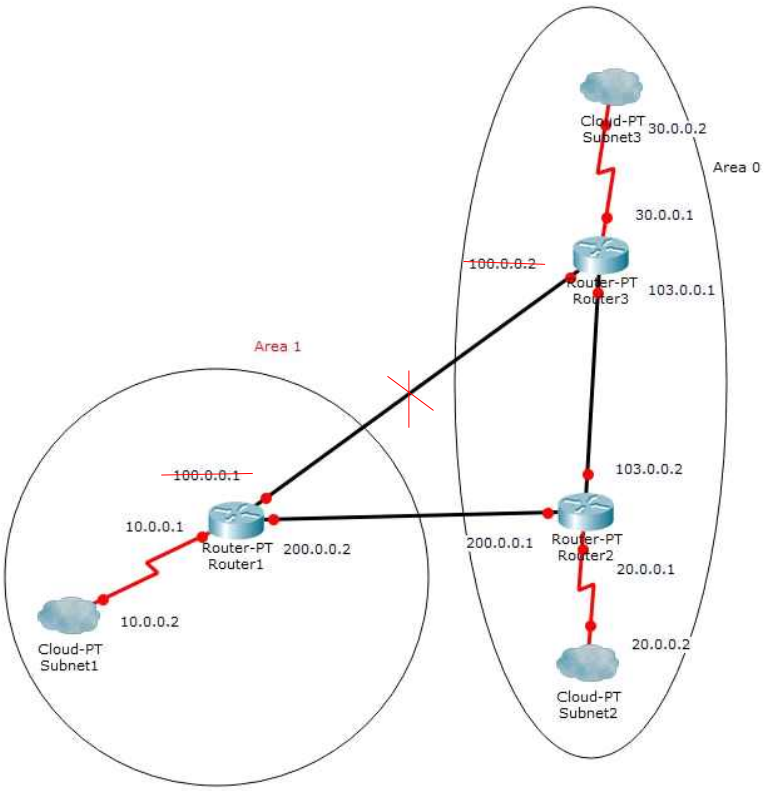
• 계층구조 동적 라우팅 프로토콜로 주로 대규모 네트워크에서 Area를 이용하여 분할 관리

• 많은 요소를 고려하여 경로를 선택
  -RIP보다 효율적인 경로 선택

• 링크에 변화가 생겼을 때만 업데이트

• Backbone Area 1개가 반드시 존재해야함

• 구현하고자 하는 네트워크



‣ABR : Area와 Area의 경계선에서 통신을 가능하도록 하는 라우터
  -적어도 하나의 인터페이스가 Backbone Area에 포함되어야함

‣DR : OSPF에 참여하는 라우터들이 알려주는 LSA를 관리하면서 링크의 상태를 항상 일치시키는 라우터
  -1개의 Area에 1개의 DR이 존재해야함.

• 다음과 같은 명령어로 OSPF Protocols 설정

```
set interfaces loopback lo address 1.1.1.1/32
set protocols ospf area 1 network 192.168.0.0/24
set protocols ospf default-information originate always
set protocols ospf default-information originate metric 10
set protocols ospf default-information originate metric-type 2
set protocols ospf log-adjacency-changes
set protocols ospf parameters router-id 1.1.1.1
set protocols ospf redistribute connected metric-type 2
set protocols ospf redistribute connected route-map CONNECT
set policy route-map CONNECT rule 10 action permit
set policy route-map CONNECT rule 10 match interface lo
```

• Router-id를 이용하여 Area에서 BR의 우선순위를 부여

-ABR이 BR이 되지 않도록 우선순위를 가장 낮게 잡음

-ABR은 2개의 area를 포함

show protocols 을 통해 확인

```
ospf {
    area 0 {
        network 100.0.0.0/24
    }
    area 1 {
        network 10.0.0.0/24
    }
    area 2 {
        network 200.0.0.0/24
    }
    default-information {
        originate {
            always
            metric 10
            metric-type 2
        }
    }
    log-adjacency-changes {
    }
    parameters {
        router-id 1.1.1.1
    }
    redistribute {
        connected {
```

<Router 1>

```
ospf {
    area 0 {
        network 200.0.0.0/24
    }
    area 1 {
        network 20.0.0.0/24
    }
    area 2 {
        network 103.0.0.0/24
    }
    default-information {
        originate {
            always
            metric 10
            metric-type 2
        }
    }
    log-adjacency-changes {
    }
    parameters {
        router-id 2.2.2.2
    }
    redistribute {
        connected {
```

<Router 1-2>

```
ospf {
    area 1 {
        network 80.0.0.0/24
    }
    area 2 {
        network 100.0.0.0/24
        network 103.0.0.0/24
    }
    default-information {
        originate {
            always
            metric 10
            metric-type 2
        }
    }
    log-adjacency-changes {
    }
    parameters {
        router-id 3.3.3.3
    }
    redistribute {
        connected {
            metric-type 2
            route-map CONNECT
:_
```

<Router 1-3>

‣run show ip ospf route를 통해 table 확인

```
vyos@vyos# run sh ip ospf route
============ OSPF network routing table ============
N     10.0.0.0/24             [10] area: 0.0.0.1
                              directly attached to eth0
N IA  20.0.0.0/24             [20] area: 0.0.0.1
                              via 200.0.0.1, eth2
N IA  30.0.0.0/24             [30] area: 0.0.0.1
                              via 200.0.0.1, eth2
N IA  103.0.0.0/24            [20] area: 0.0.0.1
                              via 200.0.0.1, eth2
N     200.0.0.0/24            [10] area: 0.0.0.1
                              directly attached to eth2

============ OSPF router routing table =============
R     2.2.2.2                 [10] area: 0.0.0.1, ABR
                              via 200.0.0.1, eth2
```

<Router 1>

```
vyos@vyos# run sh ip ospf route
============ OSPF network routing table ============
N     10.0.0.0/24             [20] area: 0.0.0.1
                              via 200.0.0.2, eth1
N     20.0.0.0/24             [10] area: 0.0.0.0
                              directly attached to eth0
N     30.0.0.0/24             [20] area: 0.0.0.0
                              via 103.0.0.1, eth2
N     103.0.0.0/24            [10] area: 0.0.0.0
                              directly attached to eth2
N     200.0.0.0/24            [10] area: 0.0.0.1
                              directly attached to eth1
```

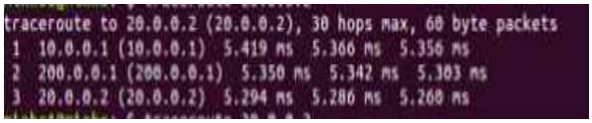
<Router 1-2>

```
vyos@vyos# run sh ip ospf route
============ OSPF network routing table ============
N IA  10.0.0.0/24             [30] area: 0.0.0.0
                              via 103.0.0.2, eth2
N     20.0.0.0/24             [20] area: 0.0.0.0
                              via 103.0.0.2, eth2
N     30.0.0.0/24             [10] area: 0.0.0.0
                              directly attached to eth0
N     103.0.0.0/24            [10] area: 0.0.0.0
                              directly attached to eth2
N IA  200.0.0.0/24            [20] area: 0.0.0.0
                              via 103.0.0.2, eth2

============ OSPF router routing table =============
R     2.2.2.2                 [10] area: 0.0.0.0, ABR
                              via 103.0.0.2, eth2
```

<Router 1-3>

• traceroute 을 통한 확인

‣Node 1 -> Node 2



‣Node 1 -> Node 3

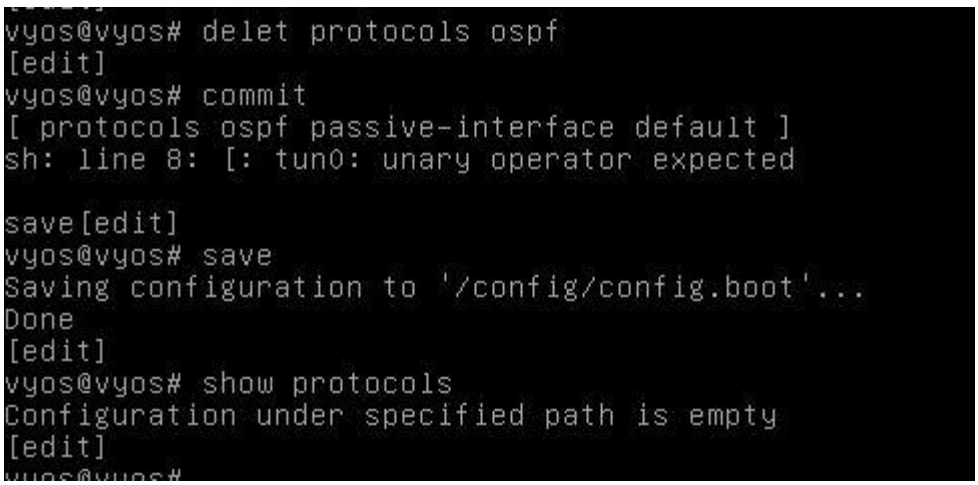
• Wireshark 를 통한 확인

‣Node 2

‣주기적으로 Hello Packet을 받는 것을 확인

• BGP를 위해 OSPF Routing을 삭제

‣delete protocols OSPF



5) BGP

• OSPF의 Area라는 개념과 비슷한 AS라는 개념을 사용해 통신

-서로 다른 AS간의 BGP Session => External BGP

-동일 AS내의 BGP Router간의 BGP Session => Internal BGP

```
set interfaces protocols bgp 65536 neighbor 100.0.0.2 ebgp-multihop '2'
set interfaces protocols bgp 65536 neighbor remote-as 65539
set interfaces protocols bgp 65536 neighbor update-source 100.0.0.1
set interfaces protocols bgp 65536 neighbor 200.0.0.1 ebgp-multihop '2'
set interfaces protocols bgp 65536 neighbor remote-as 65538
set interfaces protocols bgp 65536 neighbor update-source 200.0.0.2
set interfaces protocols bgp 65536 parameters router-id 200.0.0.1
set interfaces protocols bgp 65536 address-family ipv4-unicast network 10.0.0.0/24
```

-'ebgp' 라는 명령어를 통해서 External BGP라고 알림

-'remote-as'를 통해서 neighbor IP가 속해있는 AS가 어디인지 설정

-'set protocols bgp'를 통해서 현재 작업하고 있는 AS가 65536라고 알림.

이 AS를 서로 다르게 설정.

show protocols 을 통해 확인

```
bgp 65536 {
    address-family {
        ipv4-unicast {
            network 10.0.0.0/24 {
            }
        }
    }
    neighbor 100.0.0.2 {
        ebgp-multihop 2
        remote-as 65539
        update-source 100.0.0.1
    }
    neighbor 200.0.0.1 {
        ebgp-multihop 2
        remote-as 65538
        update-source 200.0.0.2
    }
    parameters {
        router-id 200.0.0.1
    }
```

<Router 1>

```
vyos@vyos# show protocols
 bgp 65536 {
     address-family {
         ipv4-unicast {
             network 20.0.0.0/24 {
             }
         }
     }
     neighbor 103.0.0.1 {
         ebgp-multihop 2
         remote-as 65537
         update-source 103.0.0.2
     }
     neighbor 200.0.0.2 {
         ebgp-multihop 2
         remote-as 65538
         update-source 200.0.0.1
     }
     parameters {
         router-id 103.0.0.1
     }
 }
```

<Router 1-2>

```
vyos@vyos# show protocols
 bgp 65536 {
     address-family {
         ipv4-unicast {
             network 30.0.0.0/24 {
             }
         }
     }
     neighbor 100.0.0.1 {
         ebgp-multihop 2
         remote-as 65539
         update-source 100.0.0.2
     }
     neighbor 103.0.0.2 {
         ebgp-multihop 2
         remote-as 65537
         update-source 103.0.0.1
     }
     parameters {
         router-id 100.0.0.1
     }
 }
[edit]
```

<Router 1-3>

• traceroute 을 통한 확인

‣Node 1 -> Node 2

```
traceroute to 20.0.0.2 (20.0.0.2), 30 hops max, 60 byte packets
 1  10.0.0.1 (10.0.0.1)  0.586 ms  0.512 ms  0.499 ms
 2  200.0.0.1 (200.0.0.1)  0.971 ms  0.949 ms  0.941 ms
 3  20.0.0.2 (20.0.0.2)  0.933 ms  0.926 ms  0.919 ms
```

‣Node 1 -> Node 2

```
traceroute to 30.0.0.2 (30.0.0.2), 30 hops max, 60 byte packets
 1  10.0.0.1 (10.0.0.1)  0.254 ms  0.204 ms  0.192 ms
 2  100.0.0.2 (100.0.0.2)  0.513 ms  0.493 ms  0.482 ms
 3  30.0.0.2 (30.0.0.2)  0.871 ms  0.872 ms  0.845 ms
```

• RIP 프로토콜로 다음을 진행하기에 BGP 프로토콜 삭제

‣delete protocols BGP

```
vyos@vyos# delete protocols bgp
[edit]
vyos@vyos# commit
save[edit]
vyos@vyos# save
Saving configuration to '/config/config.boot'...
Done
[edit]
vyos@vyos# show protocols
Configuration under specified path is empty
[edit]
vyos@vyos#
```

5) 고장 Test

• 고장 Test는 RIP Protocol에서 진행함.

• Node 1 에서 Node 3으로 가는 100.0.0.0/24 네트워크를 삭제하였음.

```
vyos@vyos# show protocols
 rip {
     network 10.0.0.0/24
     network 200.0.0.0/24
 }
 static {
 }
[edit]
```

• ping Test

```
node1@node1:~$ ping 30.0.0.2
PING 30.0.0.2 (30.0.0.2) 56(84) bytes of data.
64 bytes from 30.0.0.2: icmp_seq=1 ttl=61 time=1.55 ms
64 bytes from 30.0.0.2: icmp_seq=2 ttl=61 time=1.35 ms
64 bytes from 30.0.0.2: icmp_seq=3 ttl=61 time=2.08 ms
64 bytes from 30.0.0.2: icmp_seq=4 ttl=61 time=2.03 ms
^C
```

‣100.0.0.0/24를 삭제했음에도 잘 전송되는 것을 확인할 수 있다.

• traceroute Test

```
traceroute to 30.0.0.2 (30.0.0.2), 30 hops max, 60 byte packets
 1  _gateway (10.0.0.1)  0.318 ms  0.166 ms  0.094 ms
 2  103.0.0.2 (103.0.0.2)  0.364 ms  0.372 ms  0.471 ms
 3  30.0.0.1 (30.0.0.1)  0.757 ms  0.688 ms  0.918 ms
 4  30.0.0.2 (30.0.0.2)  1.411 ms  1.259 ms  1.188 ms
```

‣새로운 103.0.0.2라는 루트가 생겼음을 확인할 수 있다.

• wireShark 확인

```
     5 1.546456865   10.0.0.1        224.0.0.9       RIPv2     146 Response
     6 1.546986529   10.0.0.1        224.0.0.9       RIPv2     146 Response
     7 1.546990979   10.0.0.1        224.0.0.9       RIPv2     146 Response
```

‣wireShark도 정상적으로 RIPv2를 통해 가는 것을 확인할 수 있다.

3.3 서버

1) Calculation Server&Client

• Socket Programming 과제에서 만든 간단한 Server-Client 모델

-Server는 Client로부터 받은 계산 결과를 연산해서 보내주고,

-Client는 계산 내용을 Server로 보낸다.

[참고사이트:https://beej.us/guide/bgnet/]

• C로 작성하였기 때문에 gcc를 다운로드

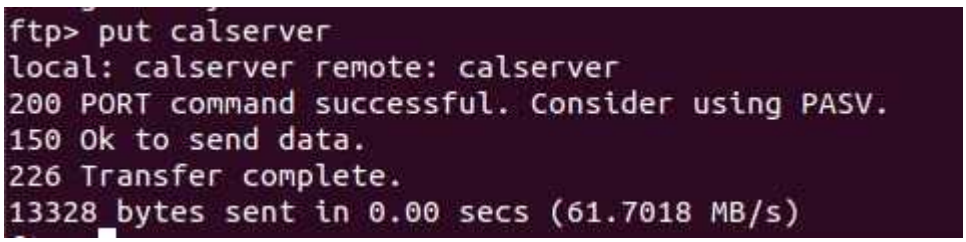
‣sudo apt-get -y gcc

• 네트워크간 통신을 원하기 때문에 내용을 수정해야함.

‣vi calserver

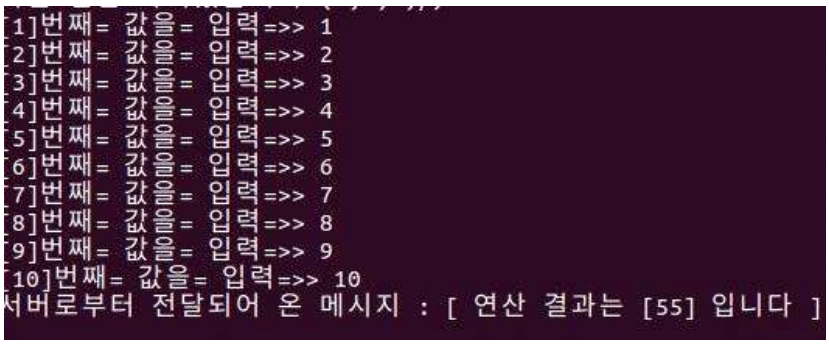
‣vi calclient

• CalServer는 Node 2로 배정할 예정이기 때문에 FTP server를 이용해 파일전송을 한다.



• 동작을 확인한다.

‣cal serv Result



2) Web Server *(Node 1)*

• Web은 인터넷에 연결된 컴퓨터들을 통해 사람들이 정보를 공유할 수 있는 전 세계적인 정보 공간을 뜻한다.

-전자 메일과 같이 인터넷 상에서 동작하는 하나의 서비스

-1993년 이래로 인터넷 구조의 절대적인 위치 차지

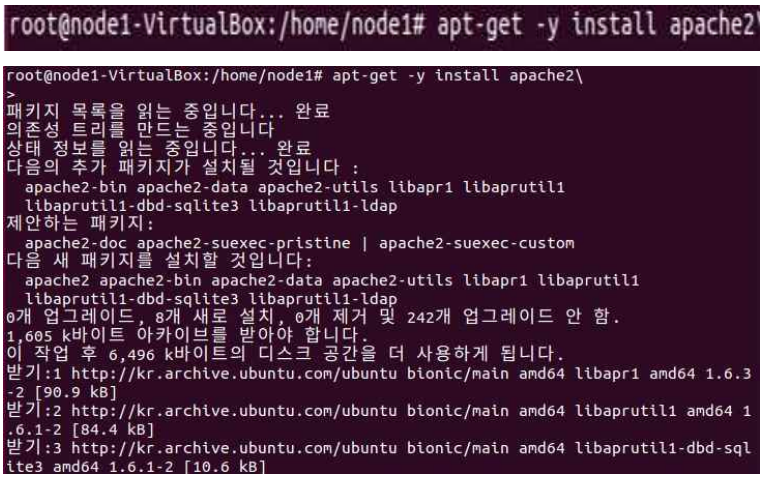
-HTTP Protocol, Hyper Text 형식등을 사용하여 그림과 문자를 교환하는 전송방식을 뜻하기도 함

[참고:이것이 우분투 리눅스다/한빛미디어/우재남 저]

[참고:https://www.wikipedia.org/]

• WEB server를 구현하기 위해 apache2 Package를 설치한다.

‣sudo apt-get -y install apache2



• Page의 내용을 수정

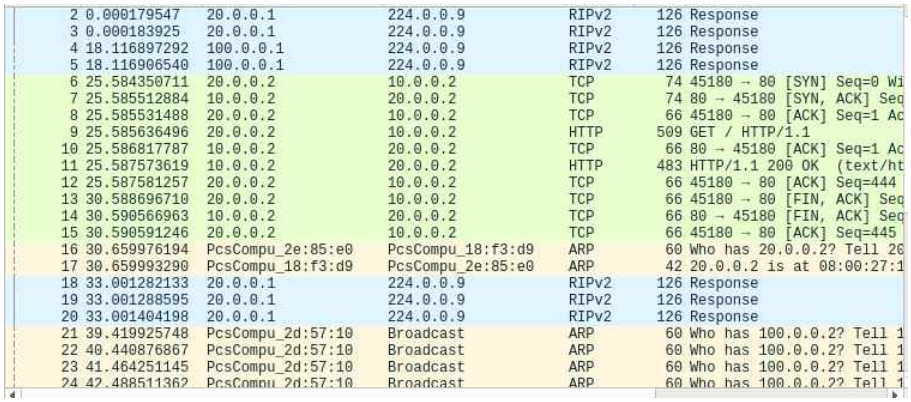
‣vi /var/www/html/index.html

• 내용을 임의로 수정

• Web Page를 확인한다.

• Node 2에서 Node 1의 Web에 접속

• Wireshark 를 통한 확인



2	0.000179547	20.0.0.1	224.0.0.9	RIPv2	126 Response
3	0.000183925	20.0.0.1	224.0.0.9	RIPv2	126 Response
4	18.116897292	100.0.0.1	224.0.0.9	RIPv2	126 Response
5	18.116906540	100.0.0.1	224.0.0.9	RIPv2	126 Response
6	25.584350711	20.0.0.2	10.0.0.2	TCP	74 45180 → 80 [SYN] Seq=0 Wi
7	25.585512884	10.0.0.2	20.0.0.2	TCP	74 80 → 45180 [SYN, ACK] Seq
8	25.585531488	20.0.0.2	10.0.0.2	TCP	66 45180 → 80 [ACK] Seq=1 Ac
9	25.585636496	20.0.0.2	10.0.0.2	HTTP	509 GET / HTTP/1.1
10	25.586817787	10.0.0.2	20.0.0.2	TCP	66 80 → 45180 [ACK] Seq=1 Ac
11	25.587573619	10.0.0.2	20.0.0.2	HTTP	483 HTTP/1.1 200 OK (text/ht
12	25.587581257	20.0.0.2	10.0.0.2	TCP	66 45180 → 80 [ACK] Seq=444
13	30.588696710	20.0.0.2	10.0.0.2	TCP	66 45180 → 80 [FIN, ACK] Seq
14	30.590566963	10.0.0.2	20.0.0.2	TCP	66 80 → 45180 [FIN, ACK] Seq
15	30.590591246	20.0.0.2	10.0.0.2	TCP	66 45180 → 80 [ACK] Seq=445
16	30.659976194	PcsCompu_2e:85:e0	PcsCompu_18:f3:d9	ARP	60 Who has 20.0.0.2? Tell 20
17	30.659993290	PcsCompu_18:f3:d9	PcsCompu_2e:85:e0	ARP	42 20.0.0.2 is at 08:00:27:1
18	33.001282133	20.0.0.1	224.0.0.9	RIPv2	126 Response
19	33.001288595	20.0.0.1	224.0.0.9	RIPv2	126 Response
20	33.001404198	20.0.0.1	224.0.0.9	RIPv2	126 Response
21	39.419925748	PcsCompu_2d:57:10	Broadcast	ARP	60 Who has 100.0.0.2? Tell 1
22	40.440876867	PcsCompu_2d:57:10	Broadcast	ARP	60 Who has 100.0.0.2? Tell 1
23	41.464251145	PcsCompu_2d:57:10	Broadcast	ARP	60 Who has 100.0.0.2? Tell 1
24	42.488511362	PcsCompu_2d:57:10	Broadcast	ARP	60 Who has 100.0.0.2? Tell 1

3) FTP Server *(Node 2)*

• 파일 전송 프로토콜은 TCP/IP 프로토콜을 가지고 Server와 Client 사이의 파일 전송을 하기 위한 프로토콜

-TCP/IP의 응용계층에 속하며,

-역사는 오래되었지만 지금도 인터넷에서 자주 사용됨

[참고:이것이 우분투 리눅스다/한빛미디어/우재남 저]
[참고:https://www.wikipedia.org/]

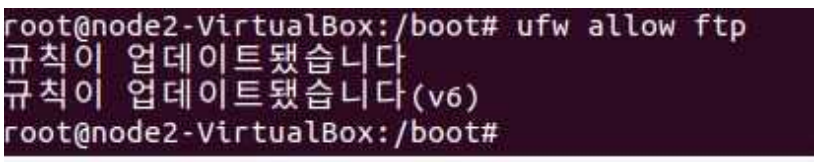
• FTP Server를 구현하기 위해 vsftpd Package를 설치한다.

‣sudo apt-get -y install vsftpd


```
root@node2-VirtualBox:/home/node2# apt-get -y install vsftpd
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음 새 패키지를 설치할 것입니다:
  vsftpd
0개 업그레이드, 1개 새로 설치, 0개 제거 및 293개 업그레이드 안 함.
115 kB바이트 아카이브를 받아야 합니다.
이 작업 후 334 kB바이트의 디스크 공간을 더 사용하게 됩니다.
받기:1 http://kr.archive.ubuntu.com/ubuntu bionic/main amd64 vsftp
amd64 3.0.3-9build1 [115 kB]
내려받기 115 kB바이트, 소요시간 1초 (156 kB바이트/초)
패키지를 미리 설정하는 중입니다...
Selecting previously unselected package vsftpd.
(데이터베이스 읽는중 ...현재 162430개의 파일과 디렉터리가 설치되어
있습니다.)
Preparing to unpack .../vsftpd_3.0.3-9build1_amd64.deb ...
Unpacking vsftpd (3.0.3-9build1) ...
Processing triggers for ureadahead (0.100.0-20) ...
vsftpd (3.0.3-9build1) 설정하는 중입니다 ...
Created symlink /etc/systemd/system/multi-user.target.wants/vsftpd
ervice →/lib/systemd/system/vsftpd.service.
Processing triggers for systemd (237-3ubuntu10.12) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Processing triggers for ureadahead (0.100.0-20) ...
```

• 다른 Node로부터 접속을 받기 위해 FTP Server를 허용한다.

‣ufw allow ftp



• vsftpd file의 설정을 한다.

‣vi /etc/vsftpd.conf

```
# and IPv4 clients. It is not necessary to listen on *both* IPv4 and IPv6
# sockets. If you want that (perhaps because you want to listen on specific
# addresses) then you must run two copies of vsftpd with two configuration
# files.
listen_ipv6=YES
#
# Allow anonymous FTP? (Disabled by default).
anonymous_enable=YES
#
# Uncomment this to allow local users to log in.
local_enable=YES
#
# Uncomment this to enable any form of FTP write command.
write_enable=YES
#
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftpd's)
#local_umask=022
#
# Uncomment this to allow the anonymous FTP user to upload files. This only
# has an effect if the above global write enable is activated. Also, you will
# obviously need to create a directory writable by the FTP user.
anon_upload_enable=YES
#
# Uncomment this if you want the anonymous FTP user to be able to create
# new directories.
anon_mkdir_write_enable=YES
#
# Activate directory messages - messages given to remote users when they
# go into a certain directory.
dirmessage_enable=YES
#
# If enabled, vsftpd will display directory listings with the time
# in  your  local  time  zone.  The default is to display GMT. The
# times returned by the MDTM FTP command are also affected by this
# option.
use_localtime=YES
#
```

‣25 = no->yes 비계정 접속허용

‣31 = # delete 업로드 가능

• Node 1에서 Node 2의 FTP에 접속

‣Connect

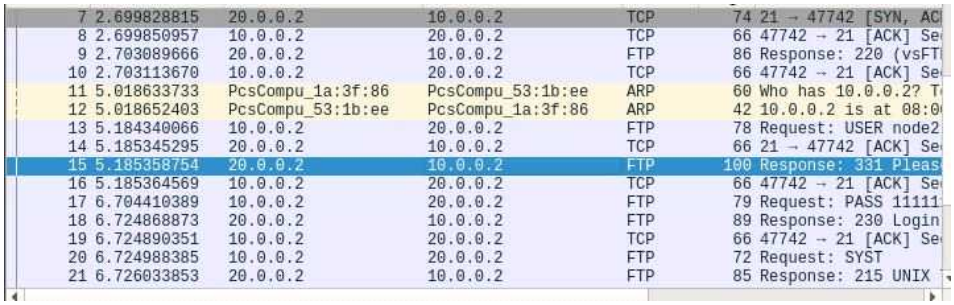
```
root@node1:/home/node1# ftp 20.0.0.2
Connected to 20.0.0.2.
220 (vsFTPd 3.0.3)
Name (20.0.0.2:node1): node2
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
```

‣Get

```
ftp> get hello.txt
local: hello.txt remote: hello.txt
200 PORT command successful. Consider using PASV.
150 Opening BINARY mode data connection for hello.txt (13 b
ytes).
226 Transfer complete.
13 bytes received in 0.01 secs (2.2821 kB/s)
```

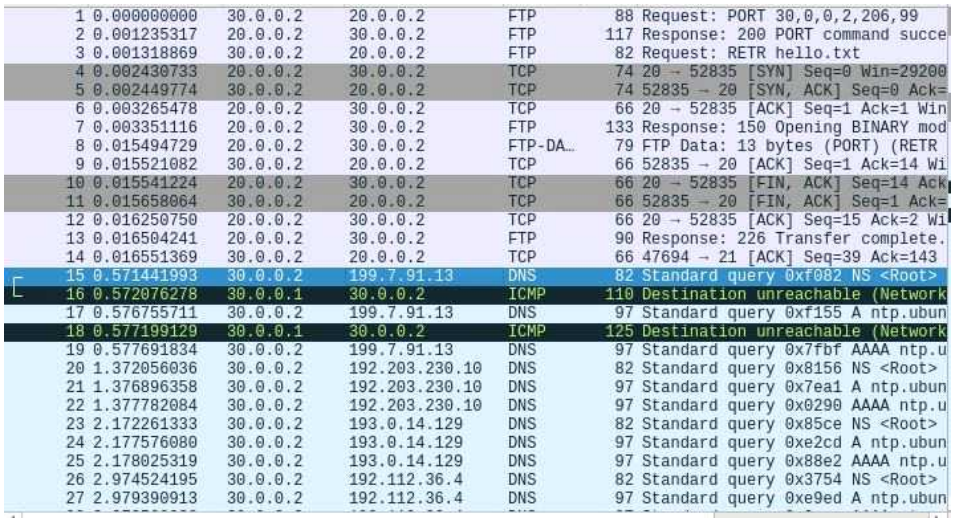
• Wireshark 를 통한 확인

‣Node 1 -> Node 2



7	2.699828815	20.0.0.2	10.0.0.2	TCP	74 21 → 47742 [SYN, AC
8	2.699850957	10.0.0.2	20.0.0.2	TCP	66 47742 → 21 [ACK] Se
9	2.703089666	20.0.0.2	10.0.0.2	FTP	86 Response: 220 (vsFT
10	2.703113670	10.0.0.2	20.0.0.2	TCP	66 47742 → 21 [ACK] Se
11	5.018633733	PcsCompu_1a:3f:86	PcsCompu_53:1b:ee	ARP	60 Who has 10.0.0.2? T
12	5.018652403	PcsCompu_53:1b:ee	PcsCompu_1a:3f:86	ARP	42 10.0.0.2 is at 08:0
13	5.184340066	10.0.0.2	20.0.0.2	FTP	78 Request: USER node2
14	5.185345295	20.0.0.2	10.0.0.2	TCP	66 21 → 47742 [ACK] Se
15	5.185358754	20.0.0.2	10.0.0.2	FTP	100 Response: 331 Pleas
16	5.185364569	10.0.0.2	20.0.0.2	TCP	66 47742 → 21 [ACK] Se
17	6.704410389	10.0.0.2	20.0.0.2	FTP	79 Request: PASS 11111
18	6.724868873	20.0.0.2	10.0.0.2	FTP	89 Response: 230 Login
19	6.724890351	10.0.0.2	20.0.0.2	TCP	66 47742 → 21 [ACK] Se
20	6.724988385	10.0.0.2	20.0.0.2	FTP	72 Request: SYST
21	6.726033853	20.0.0.2	10.0.0.2	FTP	85 Response: 215 UNIX

‣Node 2 -> Node 3



1	0.000000000	30.0.0.2	20.0.0.2	FTP	88 Request: PORT 30,0,0,2,206,99
2	0.001235317	20.0.0.2	30.0.0.2	FTP	117 Response: 200 PORT command succe
3	0.001318869	30.0.0.2	20.0.0.2	FTP	82 Request: RETR hello.txt
4	0.002430733	20.0.0.2	30.0.0.2	TCP	74 20 → 52835 [SYN] Seq=0 Win=29200
5	0.002449774	30.0.0.2	20.0.0.2	TCP	74 52835 → 20 [SYN, ACK] Seq=0 Ack=
6	0.003265478	20.0.0.2	30.0.0.2	TCP	66 20 → 52835 [ACK] Seq=1 Ack=1 Win
7	0.003351116	20.0.0.2	30.0.0.2	FTP	133 Response: 150 Opening BINARY mod
8	0.015494729	20.0.0.2	30.0.0.2	FTP-DA…	79 FTP Data: 13 bytes (PORT) (RETR
9	0.015521082	30.0.0.2	20.0.0.2	TCP	66 52835 → 20 [ACK] Seq=1 Ack=14 Wi
10	0.015541224	20.0.0.2	30.0.0.2	TCP	66 20 → 52835 [FIN, ACK] Seq=14 Ack
11	0.015658064	30.0.0.2	20.0.0.2	TCP	66 52835 → 20 [FIN, ACK] Seq=1 Ack=
12	0.016250750	20.0.0.2	30.0.0.2	TCP	66 20 → 52835 [ACK] Seq=15 Ack=2 Wi
13	0.016504241	20.0.0.2	30.0.0.2	FTP	90 Response: 226 Transfer complete.
14	0.016551369	30.0.0.2	20.0.0.2	TCP	66 47694 → 21 [ACK] Seq=39 Ack=143
15	0.571441993	30.0.0.2	199.7.91.13	DNS	82 Standard query 0xf082 NS <Root>
16	0.572076278	30.0.0.1	30.0.0.2	ICMP	110 Destination unreachable (Network
17	0.576755711	30.0.0.2	199.7.91.13	DNS	97 Standard query 0xf155 A ntp.ubun
18	0.577199129	30.0.0.1	30.0.0.2	ICMP	125 Destination unreachable (Network
19	0.577691834	30.0.0.2	199.7.91.13	DNS	97 Standard query 0x7fbf AAAA ntp.u
20	1.372056036	30.0.0.2	192.203.230.10	DNS	82 Standard query 0x8156 NS <Root>
21	1.376896358	30.0.0.2	192.203.230.10	DNS	97 Standard query 0x7ea1 A ntp.ubun
22	1.377782084	30.0.0.2	192.203.230.10	DNS	97 Standard query 0x0290 AAAA ntp.u
23	2.172261333	30.0.0.2	193.0.14.129	DNS	82 Standard query 0x85ce NS <Root>
24	2.177576080	30.0.0.2	193.0.14.129	DNS	97 Standard query 0xe2cd A ntp.ubun
25	2.178025319	30.0.0.2	193.0.14.129	DNS	97 Standard query 0x88e2 AAAA ntp.u
26	2.974524195	30.0.0.2	192.112.36.4	DNS	82 Standard query 0x3754 NS <Root>
27	2.979390913	30.0.0.2	192.112.36.4	DNS	97 Standard query 0xe9ed A ntp.ubun

4) DNS Server *(Node 3)*

• 도메인 네임 시스템[3]은 호스트의 도메인 이름을 호스트의 네트워크 주소로 바꾸거나, 그 반대의 변환을 수행할 수 있도록 하기 위해 개발

-특정 컴퓨터의 주소를 찾기 위해, 사람이 이해하기 쉬운 도메인 이름을

-숫자로 된 식별 번호로 변환해준다.

• 주 컴퓨터 도메인 이름을 IP 주소로 변환하고 라우팅 정보를 제공하는 분산형 데이터베이스 시스템

[참고:이것이 우분투 리눅스다/한빛미디어/우재남 저]

[참고:https://www.wikipedia.org/]

• DNS Server를 구현하기 위해 bind9 Package를 설치한다.

‣sudo apt-get -y install bind9 bind9utills

```
root@node3-VirtualBox:/home/node3# apt-get -y install bind9 bind9utils
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음의 추가 패키지가 설치될 것입니다 :
  bind9-host dnsutils libbind9-160 libdns1100 libirs160 libisc169 libisccc160
  libisccfg160 liblwres160 python3-ply
제안하는 패키지:
  bind9-doc resolvconf rblcheck python-ply-doc
다음 새 패키지를 설치할 것입니다:
  bind9 bind9utils python3-ply
다음 패키지를 업그레이드할 것입니다:
  bind9-host dnsutils libbind9-160 libdns1100 libirs160 libisc169 libisccc160
  libisccfg160 liblwres160
9개 업그레이드, 3개 새로 설치, 0개 제거 및 110개 업그레이드 안 함.
659 kB바이트/2,209 kB바이트 아카이브를 받아야 합니다.
이 작업 후 3,561 kB바이트의 디스크 공간을 더 사용하게 됩니다.
받기:1 http://kr.archive.ubuntu.com/ubuntu bionic/main amd64 python3-ply all 3.1
1-1 [46.6 kB]
받기:2 http://security.ubuntu.com/ubuntu bionic-security/main amd64 bind9utils a
md64 1:9.11.3+dfsg-1ubuntu1.7 [216 kB]
```

• DNS file을 설정한다.(step. 2)

‣vi /etc/bind/named.conf.local

```
zone "xogud.com"{
        type master;
        file "/etc/bind/db.xogud.com";
};

zone "0.30.in-addr.arpa"{
        type master;
        file "/etc/bind/ip-xogud.com";
};

zone "example.com"{
        type master;
        file "/etc/bind/db.example.com";
};

zone "0.10.in-addr.arpa"{
        type master;
        file "/etc/bind/ip-exmaple.com";
};

zone "example.org"{
        type master;
        file "/etc/bind/db.example.org";
};

zone "0.20.in-addr.arpa"{
        type master;
        file "/etc/bind/ip-example.org";
};
```

3) DNS

• DNS file을 설정한다.(step. 3)

‣vi /etc/hosts

```
127.0.0.1       localhost
30.0.0.2        node3-VirtualBox
30.0.0.2 xogud.com
```

‣Node 1과 Node 2의 호스트 네임에 해당하는 네임을 할당한다.

• DNS file을 설정한다.(step. 4)

‣vi /etc/bind/db.xogud.com

```
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     xogud.com.      root.xogud.com. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      xogud.com.
@       IN      A       30.0.0.2
@       IN      AAAA    ::1
WWW     IN      A       30.0.0.2
~
~
```

‣위에 나머지 2개도 설정

• DNS file을 설정한다.(step. 5)

‣vi /etc/bind/ip-xogud.com

```
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     xogud.com.      root.xogud.com. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      xogud.com
2       PTR             xogud.com
2       PTR             www.xogud.com
~
```

‣위에 나머지 2개도 설정

• DNS file을 확인한다.(step. 6)

‣named-checkzone xogud.com /etc/bind/db.xogud.com

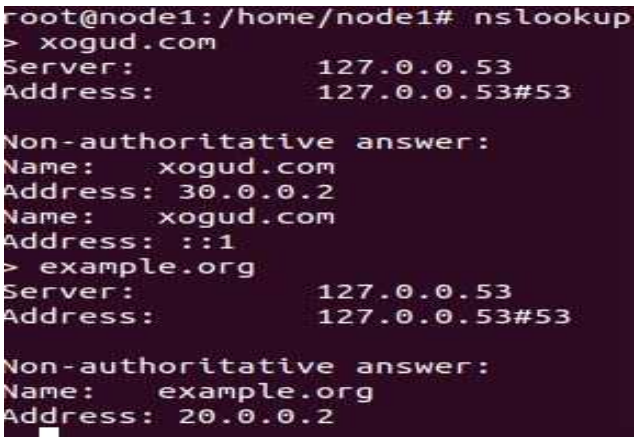
```
root@node3-VirtualBox:/etc/bind# named-checkzone xogud.com /etc/bind/db.xogud.com
zone xogud.com/IN: loaded serial 2
OK
```

‣local로 지정한 시리얼이 ok가 나오면 성공

• IP setting에서 DNS Server를 할당한다.

• Node 1과 Node 2도 마찬가지로 네임서버를 변경해준다.

• nslookup을 통해 확인



• Web Server 접속을 통한 확인

• FTP Server 접속을 통한 확인

```
root@node3-VirtualBox:/etc/bind# ftp example.org
Connected to example.org.
220 (vsFTPd 3.0.3)
Name (example.org:node3): node2
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

• WireShark를 통해 확인

497	0.099389617	30.0.0.2	10.0.0.2	TCP	74 53 → 41704 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len
498	0.099401237	10.0.0.2	30.0.0.2	TCP	66 41704 → 53 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TS
499	0.099470483	10.0.0.2	30.0.0.2	DNS	100 Standard query 0x2854 A ntp.ubuntu.com
500	0.099522862	30.0.0.2	10.0.0.2	TCP	74 53 → 41706 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len
501	0.099531574	10.0.0.2	30.0.0.2	TCP	66 41706 → 53 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TS
502	0.099603858	10.0.0.2	30.0.0.2	DNS	100 Standard query 0x3734 AAAA ntp.ubuntu.com
503	0.100110569	30.0.0.2	10.0.0.2	TCP	66 53 → 41704 [ACK] Seq=1 Ack=35 Win=29056 Len=0 T
504	0.100116870	30.0.0.2	10.0.0.2	TCP	66 53 → 41706 [ACK] Seq=1 Ack=35 Win=29056 Len=0 T
505	0.100506832	30.0.0.2	10.0.0.2	DNS	100 Standard query response 0x2854 Refused A ntp.ut
506	0.100513426	10.0.0.2	30.0.0.2	TCP	66 41704 → 53 [ACK] Seq=35 Ack=35 Win=29312 Len=0
507	0.100595568	10.0.0.2	30.0.0.2	TCP	66 41704 → 53 [FIN, ACK] Seq=35 Ack=35 Win=29312 L
508	0.100710563	10.0.0.2	30.0.0.2	TCP	74 41708 → 53 [SYN] Seq=0 Win=29200 Len=0 MSS=1466

5) Mail Server *(Node 1)*

• 전자 우편은 인터넷의 발단으로 거슬러 올라감

-인터넷이 만들어지는 데에 없어서는 안 되는 도구

‣전자 우편 전송 프로토콜[4]을 사용하고 TCP Port Numbet 25를 사용

‣모든 문자가 ASCII로 되어야 한다고 규정

-HTML과 다름

-이 때문에 문자 표현에 8비트 이상의 코드를 사용하는 언어나 첨부파일과 자주 사용되는 각종 바이너리는 마임(MIME)이라고 불리는 방식으로 7비트로 변환되어 전송

[참고:이것이 우분투 리눅스다/한빛미디어/우재남 저]

[참고:https://www.wikipedia.org/]

• Mail Server를 구현하기 위해 sendmail을 설치한다.(step .1)

‣`sudo apt-get -y install sendmail`

```
root@node1-VirtualBox:/home/node1/바탕화면# apt-get -y install sendmail
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음의 추가 패키지가 설치될 것입니다 :
  liblockfile-bin liblockfile1 libsigsegv2 lockfile-progs m4 make procmail
  sendmail-base sendmail-bin sendmail-cf sensible-mda
제안하는 패키지:
  m4-doc make-doc sendmail-doc rmail logcheck resolvconf sasl2-bin
다음 새 패키지를 설치할 것입니다:
  liblockfile-bin liblockfile1 libsigsegv2 lockfile-progs m4 make procmail
  sendmail sendmail-base sendmail-bin sendmail-cf sensible-mda
0개 업그레이드, 12개 새로 설치, 0개 제거 및 234개 업그레이드 안 함.
1,265 kB바이트 아카이브를 받아야 합니다.
이 작업 후 5,229 kB바이트의 디스크 공간을 더 사용하게 됩니다.
받기:1 http://kr.archive.ubuntu.com/ubuntu bionic/main amd64 libsigsegv2 amd64 2
.12-1 [14.7 kB]
받기:2 http://kr.archive.ubuntu.com/ubuntu bionic/main amd64 m4 amd64 1.4.18-1 [
197 kB]
받기:3 http://kr.archive.ubuntu.com/ubuntu bionic/main amd64 make amd64 4.1-9.1u
buntu1 [154 kB]
받기:4 http://kr.archive.ubuntu.com/ubuntu bionic/main amd64 liblockfile-bin amd
```

• Mail Server를 구현하기 위해 pop3를 설치한다.(step .2)

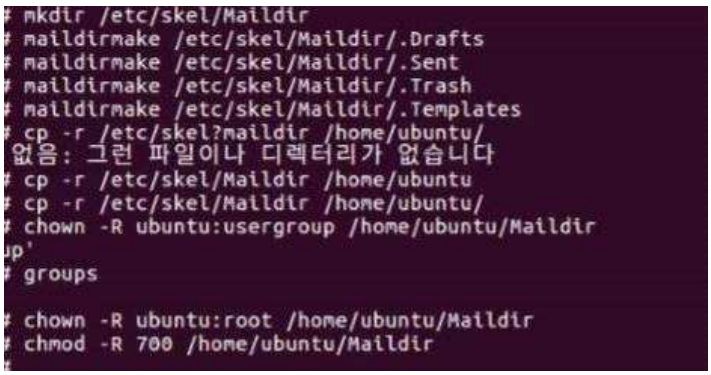
‣`sudo apt-get -y dovecat-pop3d`

```
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음의 추가 패키지가 설치될 것입니다 :
  dovecot-core
제안하는 패키지:
  dovecot-gssapi dovecot-imapd dovecot-ldap dovecot-lmtpd dovecot-managesieved dovecot-mysql dovecot-pgsql dovecot-sieve
  dovecot-solr dovecot-sqlite ntp
다음 새 패키지를 설치할 것입니다:
  dovecot-core dovecot-pop3d
0개 업그레이드, 2개 새로 설치, 0개 제거 및 0개 업그레이드 안 함.
2,772 kB바이트 아카이브를 받아야 합니다.
이 작업 후 8,791 kB바이트의 디스크 공간을 더 사용하게 됩니다.
받기:1 http://kr.archive.ubuntu.com/ubuntu bionic-updates/main amd64 dovecot-core amd64 1:2.2.33.2-1ubuntu4.3 [2,739 kB]
받기:2 http://kr.archive.ubuntu.com/ubuntu bionic-updates/main amd64 dovecot-pop3d amd64 1:2.2.33.2-1ubuntu4.3 [32.2 kB]
내려받기 2,772 kB바이트, 소요시간 2초 (1,294 kB바이트/초)
Selecting previously unselected package dovecot-core.
(데이터베이스 읽는중 ...현재 165646개의 파일과 디렉터리가 설치되어 있습니다.)
Preparing to unpack .../dovecot-core_1%3a2.2.33.2-1ubuntu4.3_amd64.deb ...
Unpacking dovecot-core (1:2.2.33.2-1ubuntu4.3) ...
Selecting previously unselected package dovecot-pop3d.
Preparing to unpack .../dovecot-pop3d_1%3a2.2.33.2-1ubuntu4.3_amd64.deb ...
Unpacking dovecot-pop3d (1:2.2.33.2-1ubuntu4.3) ...
dovecot-core (1:2.2.33.2-1ubuntu4.3) 설정하는 중입니다 ...

Creating config file /etc/dovecot/dovecot.conf with new version

Creating config file /etc/dovecot/dovecot-dict-auth.conf.ext with new version

Creating config file /etc/dovecot/dovecot-dict-sql.conf.ext with new version

Creating config file /etc/dovecot/dovecot-sql.conf.ext with new version

Creating config file /etc/dovecot/conf.d/10-auth.conf with new version

Creating config file /etc/dovecot/conf.d/10-director.conf with new version

Creating config file /etc/dovecot/conf.d/10-logging.conf with new version

Creating config file /etc/dovecot/conf.d/10-mail.conf with new version
```

4) SMTP

• Mail Server를 설정한다.(step .3)

• Mail Server를 설정한다.(step .4)



• telnet을 이용하여 메일 전송 확인

6) DHCP Server *(Node 2)*

• 동적 호스트 구성 프로토콜[5]은 호스트 IP 구성 관리를 단순화 하는 IP표준

-DHCP 표준에서는 DHCP Server를 사용,

-IP주소 및 관련된 기타 구성 세부 정보를 네트워크의 DHCP사용 Client에게 동적으로 할당

‣할당 후 더 이상 사용하지 않으면 회수

[참고:이것이 우분투 리눅스다/한빛미디어/우재남 저]

[참고:https://www.wikipedia.org/]

• DHCP Server를 구현하기 위해 isc-dhcp-server Package를 설치한다.

‣sudo apt-get -y install isc-dhcp-server

```
root@node2-VirtualBox:/home/node2# apt-get -y install isc-dhcp-server
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음의 추가 패키지가 설치될 것입니다 :
  libirs-export160 libisccfg-export160
제안하는 패키지:
  isc-dhcp-server-ldap policycoreutils
다음 새 패키지를 설치할 것입니다:
  isc-dhcp-server libirs-export160 libisccfg-export160
0개 업그레이드, 3개 새로 설치, 0개 제거 및 293개 업그레이드 안 함.
509 kB바이트 아카이브를 받아야 합니다.
이 작업 후 1,791 kB바이트의 디스크 공간을 더 사용하게 됩니다.
받기:1 http://kr.archive.ubuntu.com/ubuntu bionic-updates/main amd64 libisccfg-export160 amd64 1
:9.11.3+dfsg-1ubuntu1.7 [45.5 kB]
받기:2 http://kr.archive.ubuntu.com/ubuntu bionic-updates/main amd64 libirs-export160 amd64 1:9.
11.3+dfsg-1ubuntu1.7 [18.4 kB]
받기:3 http://kr.archive.ubuntu.com/ubuntu bionic-updates/main amd64 isc-dhcp-server amd64 4.3.5
-3ubuntu7.1 [446 kB]
내려받기 509 kB바이트, 소요시간 1초 (556 kB바이트/초)
패키지를 미리 설정하는 중입니다...
Selecting previously unselected package libisccfg-export160.
(데이터베이스 읽는중 ...현재 162487개의 파일과 디렉터리가 설치되어 있습니다.)
Preparing to unpack .../libisccfg-export160_1%3a9.11.3+dfsg-1ubuntu1.7_amd64.deb ...
Unpacking libisccfg-export160 (1:9.11.3+dfsg-1ubuntu1.7) ...
Selecting previously unselected package libirs-export160.
Preparing to unpack .../libirs-export160_1%3a9.11.3+dfsg-1ubuntu1.7_amd64.deb ...
Unpacking libirs-export160 (1:9.11.3+dfsg-1ubuntu1.7) ...
Selecting previously unselected package isc-dhcp-server.
Preparing to unpack .../isc-dhcp-server_4.3.5-3ubuntu7.1_amd64.deb ...
Unpacking isc-dhcp-server (4.3.5-3ubuntu7.1) ...
Processing triggers for ureadahead (0.100.0-20) ...
libisccfg-export160 (1:9.11.3+dfsg-1ubuntu1.7) 설정하는 중입니다 ...
Processing triggers for libc-bin (2.27-3ubuntu1) ...
Processing triggers for systemd (237-3ubuntu10.12) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
```

• DHCP Server를 구현하기 위해 isc-dhcp-server file을 설정

‣vi /etc/default/isc-dhcp-server

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

auto enp0s3
iface enp0s3 inet static
        address 20.0.0.2
        netmask 255.255.255.0
        up service isc-dhcp-server restart
```

5) DHCP

• DHCP Server를 구현하기 위해 dhcpd.conf file을 설정

‣vi /etc/dhcp/dhcpd.conf

```
# option definitions common to all supported networks...
option domain-name "example.org";
option domain-name-servers ns1.example.org, ns2.example.org;

default-lease-time 600;
max-lease-time 7200;
```

```
# A slightly different configuration for an internal subnet.
subnet 20.0.0.0 netmask 255.255.255.0{
  range 20.0.0.5 20.0.0.254;
  option domain-name-servers ns1.internal.example.org;
  option domain-name "internal.example.org";
  option subnet-mask 255.255.255.0;
  option routers 20.0.0.1;
  option broadcast-address 20.0.0.255;
  default-lease-time 600;
  max-lease-time 7200;
}
```

```
# set.
host fantasia {
  hardware ethernet 08:00:07:26:c0:a5;
  fixed-address fantasia.example.com;
}
```

• DHCP Server실행 후 동작을 확인

‣sudo systemctl start isc-dhcp-server.service

‣sudo systemctl status isc-dhcp-server.service

```
● isc-dhcp-server.service - ISC DHCP IPv4 server
   Loaded: loaded (/lib/systemd/system/isc-dhcp-server.service; enabled; vendor
   Active: active (running) since Sat 2017-05-06 23:14:21 PDT; 2min 41s ago
     Docs: man:dhcpd(8)
 Main PID: 2511 (dhcpd)
   CGroup: /system.slice/isc-dhcp-server.service
           └─2511 dhcpd -user dhcpd -group dhcpd -f -4 -pf /run/dhcp-server/dhcp
```

7) NFS Server *(Node 3)*

• 네트워크 파일 시스템[6]은 1984년 썬 마이크로 시스템즈가 개발한 프로토콜
-클라이언트 컴퓨터의 사용자가 네트워크상의 파일을
직접 연결된 스토리지에 접근하는 방식과 비슷한 방식으로 접근하도록 도와줌

[참고:이것이 우분투 리눅스다/한빛미디어/우재남 저]
[참고:https://www.wikipedia.org/]

• NFS Server를 구현하기 위해 nfs-common nfs-kernel-server을 설치한다.(step .1)

‣sudo apt-get -y install nfs-common nfs-kernel-server

```
root@node3-VirtualBox:/usr/lib# apt-get -y install nfs-common nfs-kernel-server
rpcbind
패키지 목록을 읽는 중입니다... 완료
의존성 트리를 만드는 중입니다
상태 정보를 읽는 중입니다... 완료
다음의 추가 패키지가 설치될 것입니다 :
  keyutils libnfsidmap2 libtirpc1
제안하는 패키지:
  open-iscsi watchdog
다음 새 패키지를 설치할 것입니다:
  keyutils libnfsidmap2 libtirpc1 nfs-common nfs-kernel-server rpcbind
0개 업그레이드, 6개 새로 설치, 0개 제거 및 0개 업그레이드 안 함.
490 kB바이트 아카이브를 받아야 합니다.
이 작업 후 1,700 kB바이트의 디스크 공간을 더 사용하게 됩니다.
받기:1 http://kr.archive.ubuntu.com/ubuntu bionic/main amd64 keyutils amd64 1.5.
9-9.2ubuntu2 [47.9 kB]
받기:2 http://kr.archive.ubuntu.com/ubuntu bionic/main amd64 libnfsidmap2 amd64
0.25-5.1 [27.2 kB]
받기:3 http://kr.archive.ubuntu.com/ubuntu bionic/main amd64 rpcbind amd64 0.2.3
-0.6 [40.6 kB]
받기:4 http://kr.archive.ubuntu.com/ubuntu bionic/main amd64 nfs-common amd64 1:
1.3.4-2.1ubuntu5 [205 kB]
받기:5 http://kr.archive.ubuntu.com/ubuntu bionic/main amd64 nfs-kernel-server a
md64 1:1.3.4-2.1ubuntu5 [94.0 kB]
```

• NFS Server를 사용하기 위해 공유 폴더를 만든다. (step .3)

‣mkdir /etc/home/node3/myshare

```
root@node3-VirtualBox:/home/node3# mount -t nfs 30.0.0.2:/share myshare

sdasd
asdasd
^C
root@node3-VirtualBox:/home/node3#
root@node3-VirtualBox:/home/node3# ls -l myshare/
합계 0
root@node3-VirtualBox:/home/node3# cd myshare
root@node3-VirtualBox:/home/node3/myshare# ls
root@node3-VirtualBox:/home/node3/myshare# cat > test.txt.
asdasdas
^C
```

• NFS Server를 사용하기 위해 설정을 변경한다. (step. 3)

```
# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes       hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subt
ree_check)
#
# Example for NFSv4:
# /srv/nfs4        gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#
/share 10.0.0.2.*(rw,sync)
/share 20.0.0.2.*(rw,sync)
~
```

‣허용하는 IP추가

• /share 폴더를 공유하는 노드 IP를 확인한다. (step. 4)

‣exportfs -v

6) NFS

```
root@node3-VirtualBox:/# exportfs -v
/share          10.0.0.2.*(rw,wdelay,root_squash,no_subtree_check,sec=sys,rw,sec
ure,root_squash,no_all_squash)
/share          20.0.0.2.*(rw,wdelay,root_squash,no_subtree_check,sec=sys,rw,sec
ure,root_squash,no_all_squash)
root@node3-VirtualBox:/#
```

• NFS Server를 실행 후 동작 확인

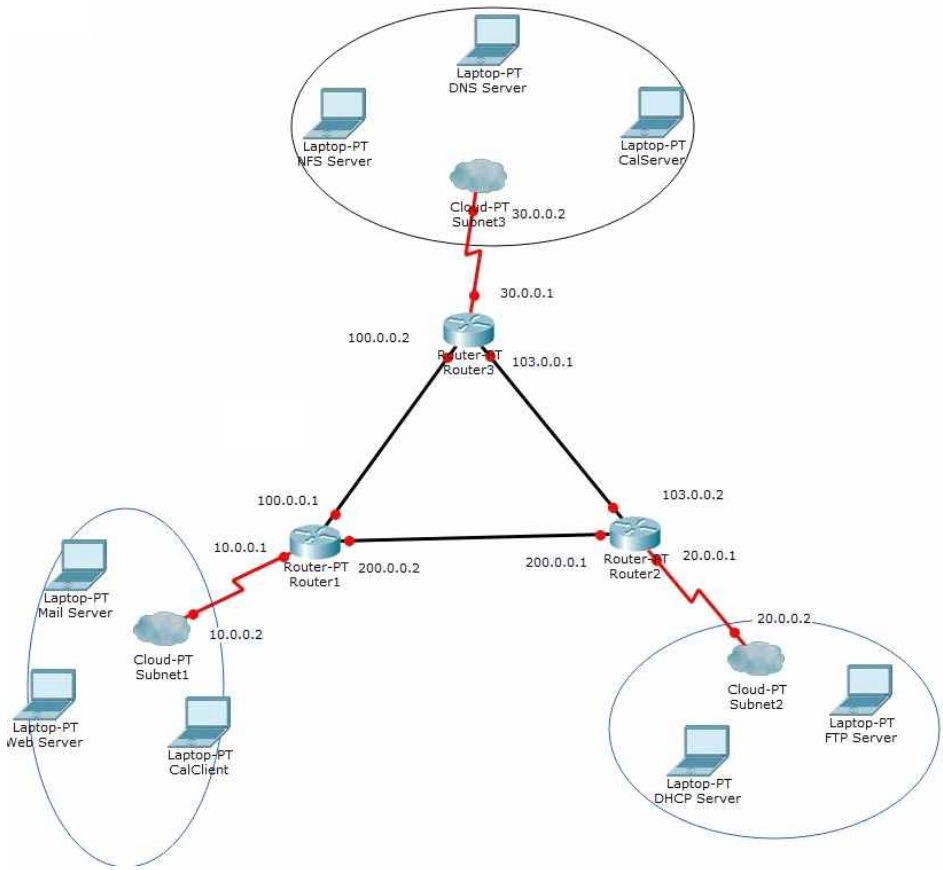
- ‣sudo systemctl start nfs-server
- ‣sudo systemctl status start nfs-server

```
● nfs-server.service - NFS server and services
   Loaded: loaded (/lib/systemd/system/nfs-server.service; enabled; vendor pr
   Active: active (exited) since Sun 2017-05-14 04:08:51 PDT; 23min ago
 Main PID: 6294 (code=exited, status=0/SUCCESS)
   CGroup: /system.slice/nfs-server.service

May 14 04:08:50 ubuntu systemd[1]: Starting NFS server and services...
May 14 04:08:50 ubuntu exportfs[6291]: exportfs: can't open /etc/exports for
May 14 04:08:51 ubuntu systemd[1]: Started NFS server and services.
```

4. 결과

4.1 네트워크 구성도

• 최종적으로 다음과 같은 3개의 라우터로 연결된 소규모 네트워크가 구성되었다.



• 라우터 간의 연결은 동적라우팅을 통해 테이블을 생성하고 경로를 구성했다.
이에 따른 알고리즘으로 RIP, OSPF, BGP의 연결을 확인했고 최종적으로 RIP를 사용하였다.

• Subnet 1(Node 1)은 Mail Server, Web Server, CalClient를 포함하고 있으며.
10.0.0.2/24(A class) IP를 할당받았고 Router 1의 adapter 0번 10.0.0.1을 GateWay로 사용한다.

• Subnet 2(Node 2)은 FTP Server, DHCP Server를 포함하고 있으며,
20.0.0.2/24(A class) IP를 할당받았고 Router 2의 adapter 0번 20.0.0.1을 GateWay로 사용한다.

• Subnet 3(Node 3)은 DNS Server, NFS Server, CalServer를 포함하고 있으며
30.0.0.2/24(A class) IP를 할당받았고 Router 3의 adapter 0번 30.0.0.1을 GateWay로 사용한다.

4.2 정리

이번 과제를 하면서 컴퓨터 네트워크 강의를 들으면서 막연히 느꼈던 네트워크 개념을 조금 더 다 잡게 된 거 같습니다. 강의만 들었을 때는 머리 속에 붕 떠다니는 느낌으로 뭔가 확실하게 자리를 잡지 못하였습니다. 그러나 이 Term Project를 하면서 IP주소를 나누는 Class나 NetMask, Subnetting, Router의 역할이나 Routing등 조금 더 제대로 알게 된 것 같습니다.

과제 중 힘들었던 것은 첫 번째로 과제에 대한 이해 부족이였습니다. 단순히 Router3개 Subnet3개를 사용하는 것을 가상 컴퓨터 없이 하나의 컴퓨터로 구현을 하려고 했습니다. 일주일간 책이나 인터넷을 보면서 공부하면서도 하나의 컴퓨터로 구현해야겠다는 아집에 갇혀 일주일이란 시간을 낭비하고서 혹시나 하는 마음으로 가상 머신을 설치한 뒤 6대의 컴퓨터를 만든 후 세 대는 Ubuntu, 세 대는 Vyos를 설치하고 진행을 했었는데 일주일 동안 끙끙 앓던 게 한번에 해결이 돼서 허탈하면서도 기뻤습니다. 그래도 일주일간에 노력이 전혀 쓸모가 없었던 것은 아니였습니다. 제가 무엇이 잘못된지도 모른채 이것저것 찾다보니 조금 더 깊이 공부 할 기회가 되었고 추가로 Ubuntu linux에대한 지식이 부족한 것인거 같아 교재도 새로 사서 짧게나마 공부를 하면서 하여서 여러모로 실력향상에 조금이라도 도움이 된 시간이였던 것 같습니다.

두 번째는 BGP Routing Protocol이였습니다. 라우팅 알고리즘에 대해서 제대로 숙지도 못하였고 당연히 어떤 방식으로 작용이 되고 실행이 되는지 전혀 감이 없어서 그저 Vyos 유저 가이드를 따라만 했고 당연히 동작이 되지 않았습니다. 그래서 개념부터 다시 해야겠다고 생각을 했고 여러 사이트와 책을 참고하면서 다시 공부를 하였습니다. 라우터가 어떠한 역활을 하고 무엇이고 인터페이스의 개념과 Class 개념, Area개념 또한 라우터 간 연결에 대해서 다시 공부하고 다시 해보자 라는 결심이 서서 다시 시도하게 되었습니다. 그 결과 성공하였고 어떠한 방식으로 동작을 하는지 WireShark를 통해 보았고 조금 더 라우팅 알고리즘에 대해 알게 된 느낌입니다. 특히 WireShark가 많은 도움이 되었습니다. 처음 썼을 때는 너무 난해해 보여서 이게 뭔가 싶었는데 후에 네트워크를 연결하고 이것 저것 WireShark를 통해 보게 되니 Packet의 이동과 프로토콜이 어떻게 쓰이는 지 보면서 많은 공부가 되었습니다.

서버를 구현하면서 인터넷상에서 그저 사용만 했던 것들이 어떻게 동작하는지 알게 되고 또 스스로가 만든다는 게 신기하였습니다. 특히 DHCP Server를 만들면서 처음에 제대로 동작을 하지 않아서 여러 사이트들을 찾고 책도 보면서 공부하였고 왜 작동이 안되었었는지 왜 인터페이스가 응답을 안하는 지 알게 되고 그저 원리도 모르고 사용만 했던 것이 공부를 함으로써 직접 구현을 하여 동작하는 것을 눈으로 본다는 게 의미가 깊었습니다. 또 웹이나 FTP등 여러 서버들이 생각한 것처럼 구현되고 동작되었다는게 기쁘기 그지 없습니다.

이번 Term Project 덕분에 컴퓨터 네트워크라는 강의에 대해서 조금 더 깊은 생각을 할 수 있게 되었고, 직접 네트워크를 구성해본다는 것이 얼마나 도움이 되는지 절실히 알게 되었습니다. 그저 책으로만 본 것을 구현한다는 것이 막상 시작할려고 할 때는 큰 어려움이였지만 시작하고 공부함으로써 구현을 해냈다는 것이 뿌듯하고 과제를 잘 수행한 것 같습니다.

4.3 자가 테스트

컴퓨터 네트워크 term project 자가평가표

항목	세부내용	평가 (1~5점)	비고
1. 개발환경	Packet Tracer	2	Packet Tracer를 구성도를 나타내는 용으로만 사용함
	Virtual Box	4	
	Ubuntu Linux	3	Windows에 익숙하여서 조금 더 전문적으로 다루지 못함
	Vyos	4	
2. 네트워크 구성	3개의 Router	5	
	Static Routing	5	
	RIP Routing	5	
	OSPF Routing	4	
	BGP Routing	4	
	고장 Test	3	RIP만 실험을 해서 다른 프로토콜을 시험해보지 않음
	기타		
3. 서버 동작	DNS server	5	
	DHCP server	3	추가적으로 eth를 새로 할당해야하는 것을 몰라서 많은 시간을 소요함
	Web Server	5	
	Mail Server	4	
	NFS Server	2	공유 폴더 설정까진 하였지만 다른 Node에서 접속을 시도해보지 않음
	FTP Server	5	
	Calculator Server&Client	4	
4. 프로토콜 이해	WireShark test	4	전반적인 흐름 등을 파악하는데 사용하였지만 조금더 구체적으로 공부하질 못함
5. 종합평가	보고서 작성 등	4	