[Network Monitoring]

정충호

클라우드 운영 전문가 교육 과정

목차

1. 개요

A. 배경

B. 기술 개요

2. 기술 설명

A. SNMP & SNMP Trap

B. Netflow

3. 기술구현

A. SNMP 구현

B. SNMP Trap 구현

C. Netflow 구현

4. 프로젝트 소감

I 개요

1. 배경

1차 프로젝트에서 대규모 네트워크를 구축을 시도했으나 네트워크 규모에 비해서 네트워크에 대한 상태 확인 및 장애 대응에 대한 자세한 내용을 명시하지 못했습니다. 따라서 이번 개인 프로젝트를 통해서 다양한 네트워크 모니터링 기술에 대해서 추가 프로젝트를 진행했습니다.

2. 기술 개요 (네크워크 모니터링 시스템)

네트워크 모니터링 시스템에는 네트워크 및 네트워크 운영과 관련된 여러 측면(예: 트래픽, 대역폭 사용률, 업타임)을 추적할 수 있는 소프트웨어/하드웨어 도구가 포함됩니다. 이러한 시스템은 디바이스, 그리고 네트워크를 구성하거나 네트워크와 관련이 있는 기타 요소를 감지할 수 있으며 상태 업데이트도 제공할 수 있습니다.

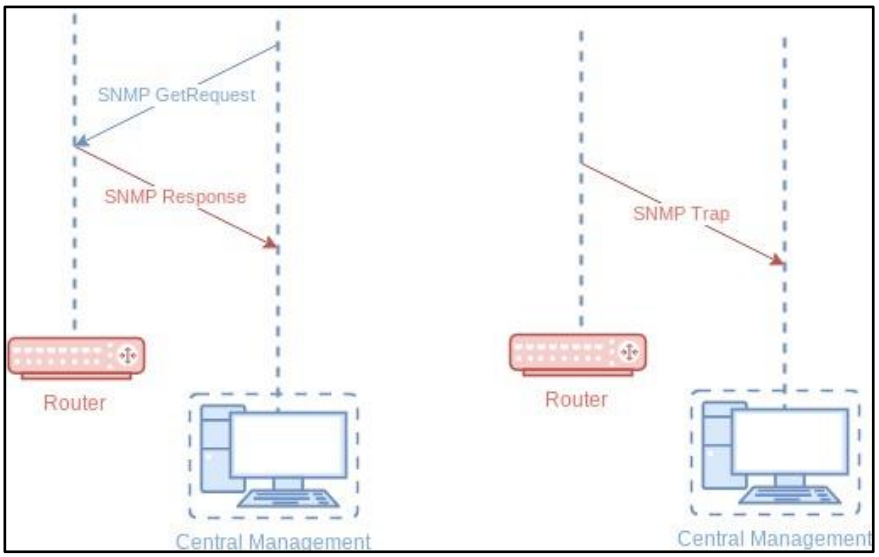
네트워크 관리자는 네트워크 모니터링 시스템을 통해 데이터 흐름을 제한하는 트래픽 병목 현상과 같은 디바이스 또는 연결 오류나 문제를 빠르게 감지할 수 있습니다. 이러한 시스템은 이메일이나 문자로 관리자에게 문제 알림을 전송할 수 있으며, 네트워크 분석을 통해 보고서를 제공할 수 있습니다.

Ⅱ 기술 설명

1. SNMP & SNMP Trap

SNMP (Simple Network Management Protocol)은 네트워크 장비의 모니터링과 관리를 위해 개발된 프로토콜입니다. 네트워크 장비들은 SNMP 에이전트를 통해 관리자에게 자신의 상태 정보를 제공하며, 이 정보는 SNMP 매니저를 통해 수집되어 모니터링 및 분석됩니다. SNMP는 네트워크 장비의 CPU, 메모리, 대역폭 사용량과 같은 성능 지표부터 인터페이스 상태, 장애 알림까지 다양한 정보를 제공하여 네트워크의 상태를 실시간으로 파악할 수 있게 합니다.

SNMP Trap은 SNMP 에이전트로부터 SNMP 매니저에게 전송되는 이벤트나 경고 메시지를 말합니다. 장비에서 중요한 상황이나 문제가 발생할 때, SNMP 에이전트는 해당 이벤트를 Trap 메시지로 매니저에게 전송하여 관리자가 빠르게 대응할 수 있도록 도와줍니다. 이를 통해 장비의 문제를 빠르게 감지하고 조치할 수 있어 네트워크의 가용성과 성능을 유지할 수 있습니다.



SNMP Version

1. SNMPv1

   가장 초기에 개발된 버전으로, 보안 기능이 제한적으로 커뮤니티 문자열을 사용하여 관리자가 장비 정보를 조회하고 수정할 수 있습니다. 데이터의 무결성과 인증이 보장되지 않기 때문에 보안에 취약하며 메시지 또한 클리어 텍스트로 전송되어 보안상 취약합니다.

2. SNMPv2c

   SNMPv1의 단점을 보안하고자 개발된 버전으로 커뮤니티 문자열을 사용하여 인증없이 데이터를 전송합니다. 하지만 보안상의 이슈와 취약점은 존재합니다.

3. SNMPv2u

   유저 기반 보안 모델을 도입하여 보안 개선을 시도한 버전입니다. 인증과 암호화를 지원하나 보안상 취약점이 존재합니다.

4. SNMPv3

   시존 취약점을 보완한 버전으로 인증, 암호화, 대아토 무결성을 지원하여 보안 기능을 제공합니다. 암호화된 메시지 전송과 사용자 기반 보안 모델을 도입하여 더욱 안전한 통신을 제공합니다.

   Username(계정)과 Password(비밀번호,커뮤니티 값)을 사용

SNMP Message

1. 관리 요청 메시지((Management Request)

종류	설명
GetRequest	SNMP 관리자가 관리 대상 장비로부터 특정 OID(객체 식별자)에 대한 정보를 조회하기 위한 요청입니다.

GetNextRequest	GetRequest와 유사하지만, 요청한 OID의 다음 OID에 해당하는 정보를 조회하는 요청입니다.
SetRequest	SNMP 관리자가 관리 대상 장비의 OID에 값을 설정하거나 수정하기 위한 요청입니다.
GetBulkRequest	여러 개의 OID 정보를 한 번에 조회하기 위한 요청으로, 대량의 정보를 효율적으로 가져올 수 있습니다.
InformRequest	SNMP 에이전트 간에 관리 정보를 공유하기 위한 요청으로, 에이전트가 수신한 정보에 대해 응답을 보냅니다.
Trap	SNMP 에이전트가 관리자에게 이벤트나 알림을 전달하기 위한 비동기적인 메시지입니다.

2. 관리 응답 메시지(Management Response)

종류	설명
GetResponse	GetRequest나 GetNextRequest에 대한 응답으로, OID에 해당하는 정보를 에이전트가 전송합니다.
SetResponse	SetRequest에 대한 응답으로, 설정한 값을 에이전트가 처리한 결과를 전송합니다.
GetBulkResponse	GetBulkRequest에 대한 응답으로, 여러 개의 OID 정보를 에이전트가 전송합니다.

SNMP Trap

네트워크 장비 운영에 있어서 큰 영향을 미칠 수 있는 중대한 정보 변경 사항에 대해 사용자에게 능동적으로 전송함으로써, 사용자가 장비 모니터링에 있어 놓칠 수 있는 부분을 보완, SNMP와 달리 162번 포트를 독자적으로 사용하여 일반적인 정보 교환과 통로를 따로 두고 있다.

Object ID(OID)

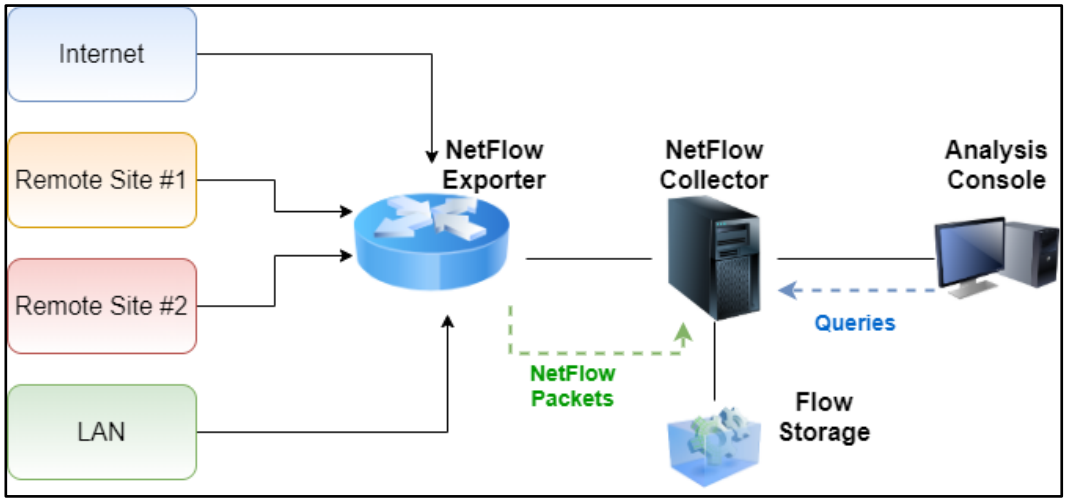
지속성 있는 유무형의 객체를, 전 세계적으로 유일무이하게 식별하기 위한 ID 체계로써 매니저와 에이전트가 메시지를 통해서 주고 받는 정보입니다. 예를 들어 CPU의 평균 5초 / 1분 / 5분 사용률은 각각의 OID에 의해서 제공됩니다.

버전	Cisco IOS Software 릴리스 12.2(3.5) 이상	Cisco IOS Software 릴리스 이후 12.0(3)T 및 12.2(3.5) 이전	12.0(3)T 이전 Cisco IOS Software 릴리스
MIB	CISCO-PROCESS-MIB	CISCO-PROCESS-MIB	이전-CISCO-CPU-MIB
개체	**cpmCPUTotal5minRev** (.1.3.6.1.4.1.9.9.109.1.1.1.1.8)	**cpmCPUTotal5min**(.1.3.6.1.4.1.9.9.109.1.1.1.1.5)	**평균 통화 중5**(.1.3.6.1.4.1.9.2.1.58)
	cpmCPUTotal1minRev (.1.3.6.1.4.1.9.9.109.1.1.1.1.7)	**cpmCPUTotal1min**(.1.3.6.1.4.1.9.9.109.1.1.1.1.4)	**avgBusy1**(.1.3.6.1.4.1.9.2.1.57)
	cpmCPUTotal5secRev (.1.3.6.1.4.1.9.9.109.1.1.1.1.6)	**cpmCPUTotal5sec** (.1.3.6.1.4.1.9.9.109.1.1.1.1.3)	**busyPer**(.1.3.6.1.4.1.9.2.1.56)

2. NetFlow

NetFlow는 네트워크 트래픽의 흐름 데이터를 수집하고 분석하는 기술로, 네트워크 성능 모니터링 및 보안 분석에 중요한 역할을 합니다. NetFlow는 다양한 네트워크 장비에서 지원되며 네트워크의 동작을 파악하고 최적화하는 데에 활용됩니다.

네트워크 장비에서 생성되는 데이터 패킷은 각각 헤더와 페이로드를 가지며, NetFlow는 이러한 패킷의 헤더 정보를 추출하여 데이터베이스에 기록합니다. 이 헤더 정보에는 출발지 IP 주소, 목적지 IP 주소, 포트 번호, 프로토콜 등이 포함됩니다. 이렇게 수집된 데이터는 네트워크의 트래픽 패턴과 트랜드를 이해하고 문제를 식별하는 데에 활용할 수 있습니다.



SNMP의 단점 보완

SNMP는 네트워크 장비에서 흐르는 트래픽의 양을 확인할 수 있으나 트래픽의 정보, 출발지 IP와 목적지 IP, 프로토콜 등을 확인 할 수 없습니다. 이러한 단점을 보완하고자 NetFlow는 다음과 같은 정보를 제공합니다.

[출발지IP, 목적지IP 출발지 Port, 목적지 Port, L3 프로토콜, 라우터 인터페이스]

라우터(스위치)는 이를 Netflow Cache라는 Datebase에 저장 후 이를 토대로 사용자는 트래픽 정보를 확인할 수 있습니다. 이때 UDP/2055를 사용하고 수집한 정보를 토대로 샘플링하여 원격 서버에 이를 압축하여 전송할 수 있습니다.

NetFlow 구성 요소

Sensor : 인터페이스에서 트래픽을 수집

Exporter : 수집한 트래픽을 샘플링하여 전송

Collector : 네트워크 장치가 전송한 데이터를 전달받아 다양한 형태호 사용자에세 제공

Ⅲ 기술 구현

1. SNMP

1.1 네트워크 장비 설정

```
[ACL 설정]
access-list 50 permit 172.20.1.2
## access-lis [정책번호] permit [모니터링 서버 IP] : 모니터링 서버는 허용 설정


access-list 50 deny any
## access-list [정책번호] deny any : 모니터링 서버를 제외한 IP를 차단
```

모니터링 장비를 제외한 장치의 접근을 막기 위해 ACL 정책을 수립

```
snmp-server community com ro 50
## snmp-server community [community 이름] [ro : Read only] [ACL 정책]
# 앞에서 설정한 정책을 통한 모니터링 서버를 제외한 점근 통제 및 커뮤니티 설정
```

SNMP 커뮤니티 설정

1.2 서버 설정

- 서버 설치
  - Ubuntu 20.04 [Server | Desktop]
  - Sudo apt-get install snmp snmpd -y

- 서버 설정

■ Sudo vi /etc/snmp/snmpd.comf

```
# Access Control
com2sec local     localhost          com
com2sec mynetwork 172.20.0.0/24      com

# Group and Access
group MyROGroup v1          local
group MyROGroup v2c         local
group MyROGroup usm         local
group MyROGroup v1          mynetwork
group MyROGroup v2c         mynetwork
group MyROGroup usm         mynetwork

view all    included  .1 80
access MyROGroup ""      any       noauth    exact  all    none   none
```

com2sec [securityName] [source] [community]

securityName : 해당 설정의 이름, Group 설정 시 참조 사용

source : SNMP IP 대역을 설정

community : 해당 대역에 커뮤니티 이름을 명시

group [groupName] [securityModel] [securityName]

groupName : 해당 그룹의 이름으로 사용

securityModel : SNMP의 보안 모델로 v1,v2,usm 이 존재

securityName : 위 설정에 정의한 보안 이름

1.3 확인

위와 같이 설정할 경우 모니터링 서버에서 네트워크 장비로 정보를 요청할 경우 다음과 같이 정보를 받아올 수 있다.

```
server@ubuntu-desktop-monitor:~$ snmpwalk -v2c -c com 172.20.0.17 1.3.6.1.4.1.9.9.109.1.1.1
iso.3.6.1.4.1.9.9.109.1.1.1.1.2.1 = INTEGER: 0
iso.3.6.1.4.1.9.9.109.1.1.1.1.3.1 = Gauge32: 0
iso.3.6.1.4.1.9.9.109.1.1.1.1.4.1 = Gauge32: 0
iso.3.6.1.4.1.9.9.109.1.1.1.1.5.1 = Gauge32: 0
iso.3.6.1.4.1.9.9.109.1.1.1.1.6.1 = Gauge32: 0
iso.3.6.1.4.1.9.9.109.1.1.1.1.7.1 = Gauge32: 0
iso.3.6.1.4.1.9.9.109.1.1.1.1.8.1 = Gauge32: 0
iso.3.6.1.4.1.9.9.109.1.1.1.1.9.1 = Gauge32: 5
iso.3.6.1.4.1.9.9.109.1.1.1.1.10.1 = Gauge32: 0
iso.3.6.1.4.1.9.9.109.1.1.1.1.11.1 = Gauge32: 0
```

하지만 모니터링 장비가 아닌 다른 장비에서 해당 네트워크 장비에 정보를 요청할 경우 다음과 같이 Timeout이 발생한다.

```
server@ubuntu-desktop-monitor:~$ snmpwalk -v2c -c com 172.20.0.17 1.3.6.1.4.1.9.9.109.1.1.1
Timeout: No Response from 172.20.0.17
server@ubuntu-desktop-monitor:~$ ping 172.20.0.17
PING 172.20.0.17 (172.20.0.17) 56(84) bytes of data.
64 bytes from 172.20.0.17: icmp_seq=1 ttl=254 time=16.8 ms
64 bytes from 172.20.0.17: icmp_seq=2 ttl=254 time=28.4 ms
```

2. SNMP Trap

2.1 네트워크 장비 설정

```
snmp-server host 172.20.1.2 version 2c com
## SNMP Trap을 보낼 서버 및 버전을 지정
snmp-server enable traps snmp linkdown linkup
## 인터페이스의 상태가 변경시 Trap을 전송하도록 설정
snmp-server source-interface traps GigabitEthernet0/0
## SNMP Trap을 전송할 인터페이스를 설정(해당 Interface의 IP를 물고 나감)
```

snmp-server enable traps 설정을 통해서 다양한 이벤트에 대해 Tarp 전송을 설정할 수 있다.

Ex)

bgp: BGP (Border Gateway Protocol) 트랩을 활성화합니다.
ospf: OSPF (Open Shortest Path First) 트랩을 활성화합니다.
isis: IS-IS 프로토콜에 대한 트랩을 활성화합니다.
config: 구성 변경에 대한 트랩을 활성화합니다.

envmon: 환경 모니터링 트랩을 활성화합니다 (예: 전원 공급 문제, 팬 상태, 온도).

2.2 서버 설정

- 서버 설치

  - Ubuntu 20.04 [Server | Desktop]
  - Sudo apt-get install snmptrapd -y

- 서버 설정

  - Sudo vi /etc/snmp/snmpdtrap.comf

```
authCommunity log,execute,net com
```

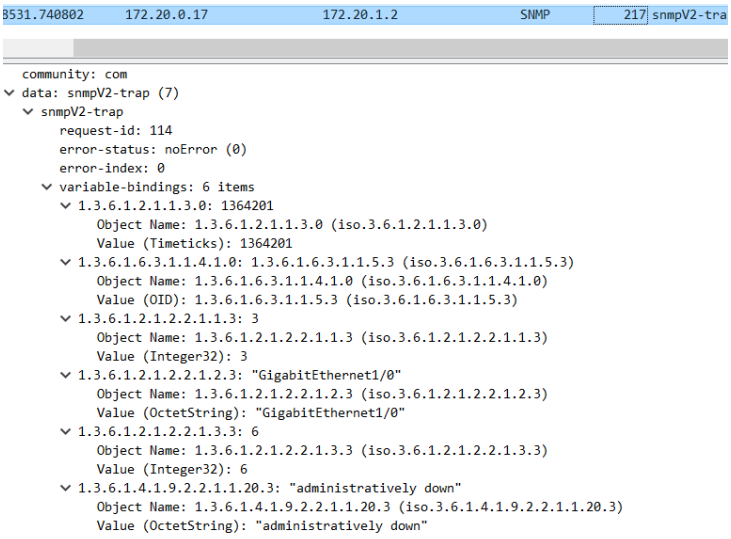
  - Sudo vi /etc/default/snmptrapd

```
TRAPDRUN=yes
```

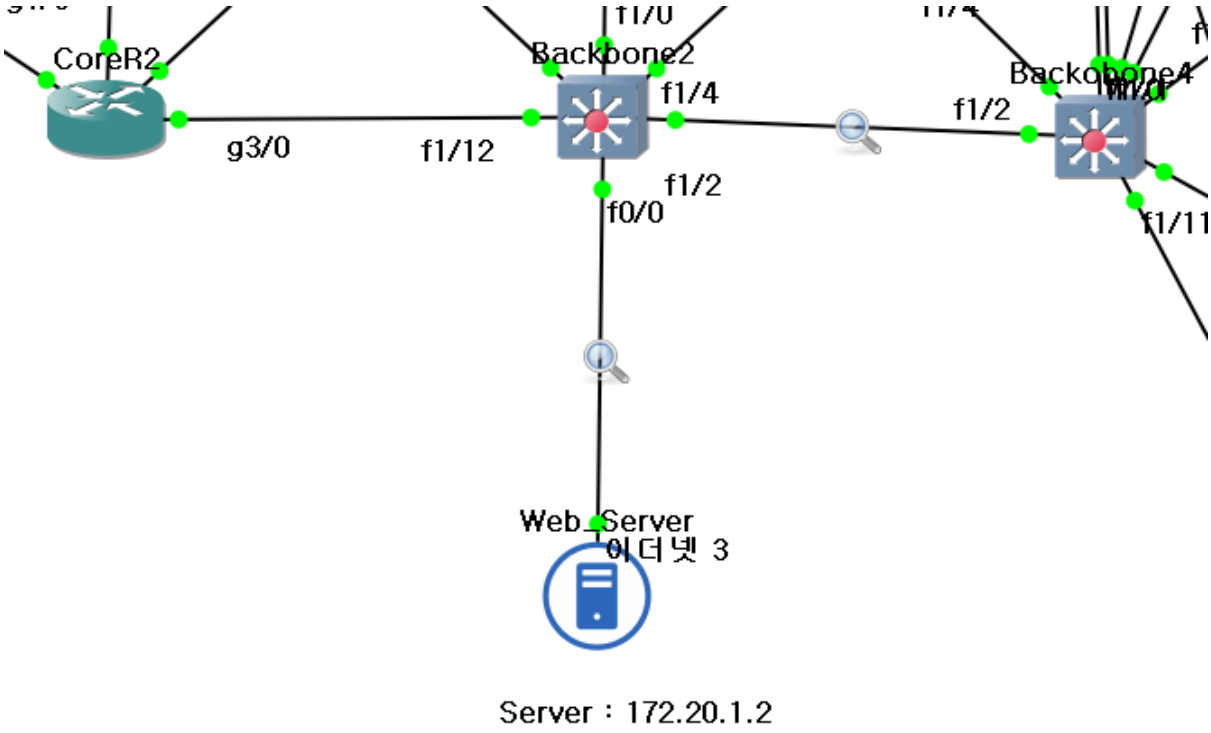
설정 후 서비스 재시작 : sudo systemctl restart snmptrapd

2.3 확인

위에서 설정한 LinkDown, LinkUp 설정을 통해서 해당 네트워크 장비에 인터페이스에 문제가 발생할 경우 해당 정보를 Trap으로 전송합니다.



3. NetFlow



다음과 같은 망 구성에 Backbone2에 NetFlow를 설정해 F1/2로 들어오고 나가는 트래픽을 Server로 보내는 시나리오로 진행하겠습니다.

1. Backbone2 설정

어떤 정보를 수집할 것인지에 대한 설정인 Record를 생성

```
flow record NTA
description config for NTA
match ipv4 tos
match ipv4 protocol
match ipv4 source address
match ipv4 destination address
match transport source-port
match transport destination-port
```

```
collect counter packets long
collect counter bytes long
collect interface output
collect interface input
collect transport tcp flags
```

match

트래픽의 Type of Service (ToS), 프로토콜, 소스 및 대상 IP 주소, 그리고 소스 및 대상 포트를 기반으로 트래픽 플로우를 구분하도록 설정되어 있습니다.

collect

각 플로우의 패킷 수, 바이트 수, 출력 및 입력 인터페이스, 그리고 TCP 플래그 정보를 수집하도록 설정되어 있습니다.

수집한 정보와 해당 데이터의 전송 방식을 지정

```
flow exporter NTA
description Exporter for NTA
destination 172.20.1.2 ## 서버 IP
source FastEthernet0/0 ## 전송할때 사용할 인터페이스
transport udp 2055 ## NetFlow의 전송 포트 번호
template data timeout 60 ## 60초 주기전송
```

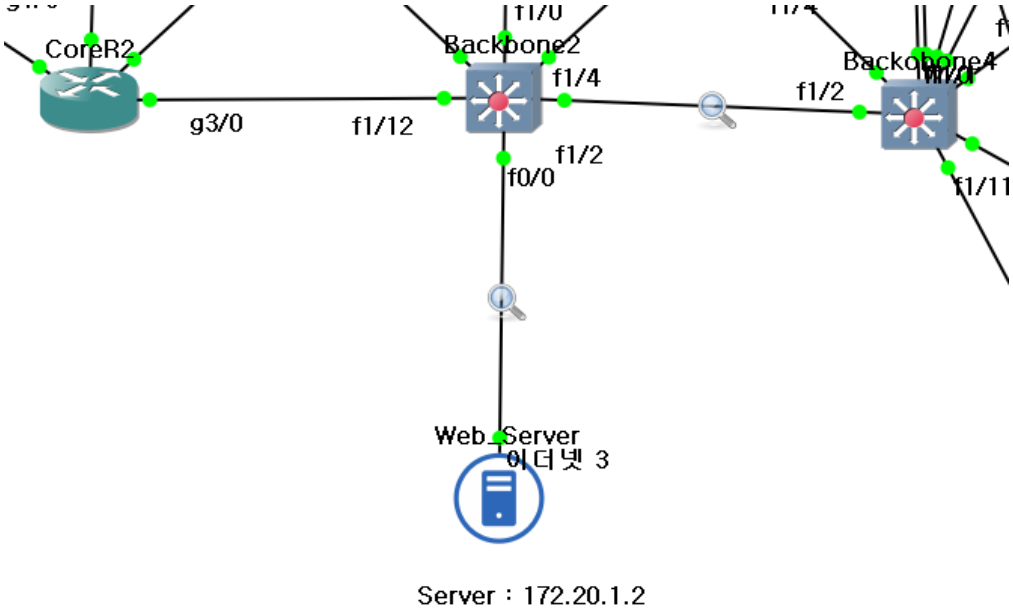
기록 방식과 정보 전달 방식 연결

```
flow monitor NTA
record NTA
exporter NTA
cache timeout active 60
```

데이터를 수집할 인터페이스에 적용

```
int fastEthernet 1/2
ip flow monitor NTA input
ip flow monitor NTA output
```

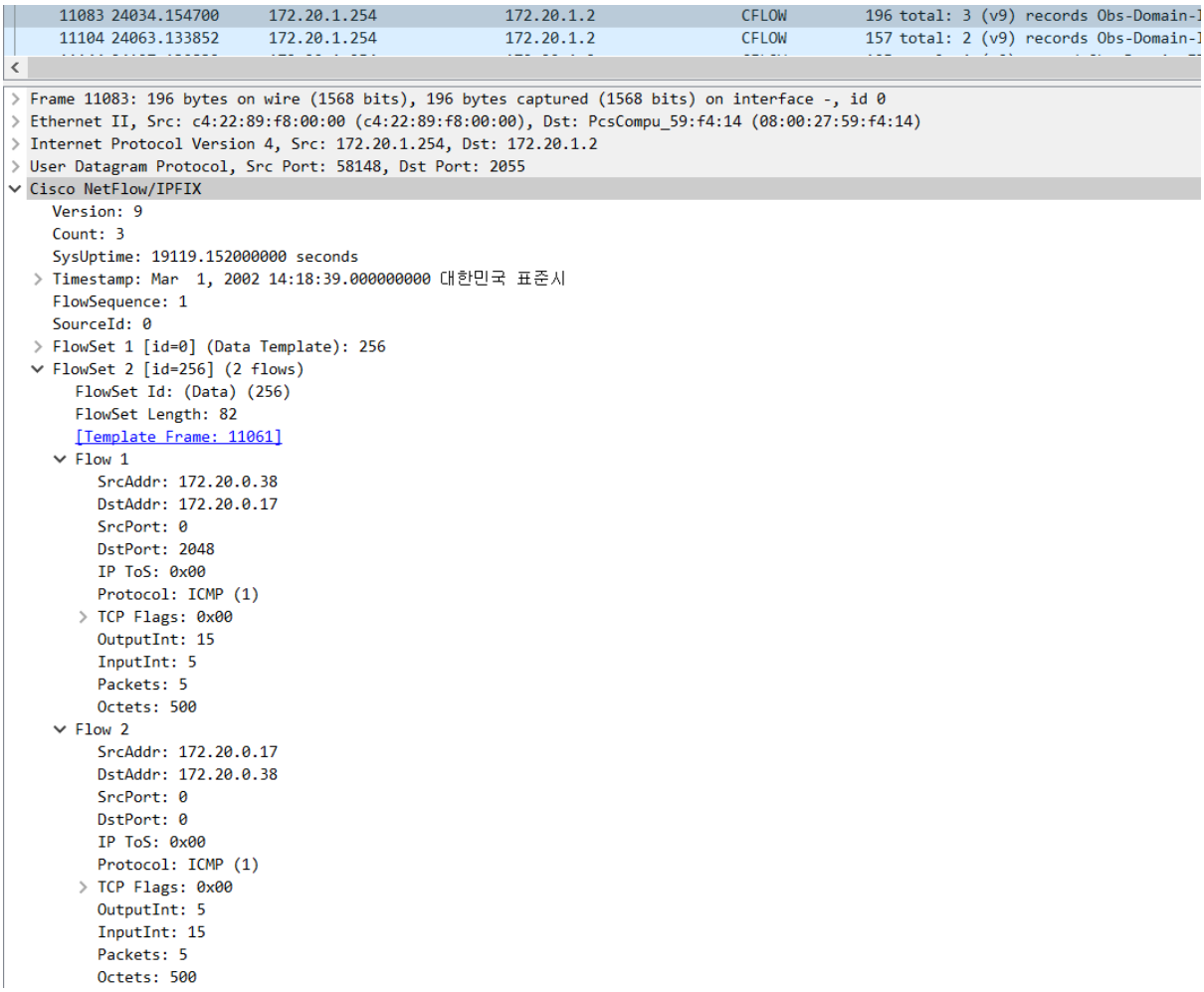
2. 확인



Server : 172.20.1.2

해당 시나리오에서 BackBone4에서 CoreR2로 Ping통신 시 BackBone2의 F1/2를 지나가니 해당 정보에 대한 트래픽을 기록하고 서버에 전송을

확인한다.



ICMP 패킷의 출발지 정보와 도착지 정보를 포함한 패킷을 172.20.1.2서버로 정상적으로 전달하는 것을 확인

3. 서버 설정

Netflow수집 서버를 통해서 해당 정보를 확인 할 수 있다.

서버 설치 : sudo apt install nfdump

확인 : nfdump -r /var/cache/nfdump/nfcapd.<timestamp>

```
server@ubuntu-desktop-monitor:/var/cache/nfdump$ nfdump -r nfcapd.202308170202
Date first seen          Event  XEvent Proto      Src IP Addr:Port          Dst IP Addr:Port     X-Src IP Addr:Port        X-Dst IP
 Addr:Port    In Byte Out Byte
1970-01-01 09:00:00.000 INVALID  Ignore ICMP       172.20.0.38:0     ->      172.20.0.17:8.0           0.0.0.0:0     ->          0
.0.0.0:0          500        0
1970-01-01 09:00:00.000 INVALID  Ignore ICMP       172.20.0.17:0     ->      172.20.0.38:0.0           0.0.0.0:0     ->          0
.0.0.0:0          500        0
Summary: total flows: 2, total bytes: 1000, total packets: 10, avg bps: 0, avg pps: 0, avg bpp: 0
Time window: 2023-08-17 02:02:29 - 2023-08-17 02:07:29
Total flows processed: 2, Blocks skipped: 0, Bytes read: 228
Sys: 0.000s flows/second: 2190.6     Wall: 0.000s flows/second: 29411.8
```

위에서 전송한 Ping통신에 대한 정보를 저장하고 확인할 수 있다.

IV 프로젝트 소감

1차 프로젝트를 진행하면서 모니터링 부분의 중요도가 낮아 많은 시간을 할애하지 못했습니다. 이와 관련된 생각으로 추가 프로젝트를 진행하게 되었습니다. 1차 프로젝트에서 SNMP와 SNMP Trap에 많은 오류가 있음을 인지하게 되었고, 이를 더 효율적으로 설계하는 방법을 학습할 수 있었습니다. 또한, NetFlow를 사용하여 네트워크 장비의 트래픽을 수집하고 분석하는 방법을 알게 되었는데, 이는 매우 유익한 경험이었습니다. 그러나, SNMP 모니터링 도구의 연결과 SNMP Trap 로그 저장 부분에서 완성도가 부족했던 점이 아쉬웠습니다. 이러한 경험을 통해 많은 것을 배울 수 있었습니다.

-끝!-