TCPDUMP 기본 사용법 #01

TCPDUMP 사용법 및 예제

TCPDUMP 란?

명령줄 인터페이스에서 실행하는 네트워크 인터페이스의 패킷을 가로채는 (스니핑) 프로그램이다
TCP/IP 뿐만이 아닌 다른 여러 패킷의 정보를 출력할 수 있다.

기본 사용법

패킷을 수집하여 파일로 저장한 후 Wireshark 등의 패킷 분석 프로그램을 사용한다.
수많은 부가 옵션이 있으나 주로 쓰이는 몇 가지 옵션만 기술하였다.

패킷 수집 패턴 커맨드 (Basic / Expression / Header Flag)

Basic (필수 옵션)

기본적인 옵션값

-i : 네트워크 인터페이스가 여러 개일 경우 지정할 수 있다
-c : 스니핑 할 패킷의 개수를 정할 수 있다.
-v : 패킷 헤더의 정보를 조금 더 자세하게 보여준다 -vv 하나 더 붙이면 더욱 자세해진다
-n : 알려진 도메인 문자열을 IP 로 보이게 해준다. -nn 하나 더 붙이면 Port 도 숫자로 표시한다.
-w : 스니핑 한 패킷을 .pcap 타입 바이너리로 저장한다. -r 옵션으로 TXT 타입으로 로드할 수 있다.

 # 특정 인터페이스(-i) 의 모든 패킷을 100개만(-c) .pcap 파일 형식으로 저장(-w) 하기
해당 형식은 txt 가 아닌 Binary 타입이며 -r 옵션으로 읽거나 wireshark 에서 로드할 수 있다.

```
tcpdump  -i ens33 -vv -c 100 -w ttt.pcap
tcpdump: listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
100 packets captured
100 packets received by filter
```

```
0 packets dropped by kernel
```

 # 특정 인터페이스(-i) 의 모든 패킷을 5개만(-c) DNS/PORT 번역하지 않고 IP 그대로 출력(-nn) 한다.

```
tcpdump  -i br0 -n -c 5
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on br0, link-type EN10MB (Ethernet), capture size 262144 bytes
14:08:14.531606  ARP,  Request  who-has  8.8.8.8  tell 10.10.10.200, length 28
14:08:15.533556  ARP,  Request  who-has  8.8.8.8  tell 10.10.10.200, length 28
14:08:16.535615  ARP,  Request  who-has  8.8.8.8  tell 10.10.10.200, length 28
14:08:18.531679  ARP,  Request  who-has  8.8.8.8  tell 10.10.10.200, length 28
14:08:19.533632  ARP,  Request  who-has  8.8.8.8  tell 10.10.10.200, length 28
5 packets captured
5 packets received by filter
0 packets dropped by kernel
```

 # 더 자세한 패킷 정보를 출력 (-v , -vv) 한다.

```
tcpdump  -i br0 -n -vv -c 5
tcpdump: listening on br0, link-type EN10MB (Ethernet), capture size 262144 bytes
14:10:46.548613 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 8.8.8.8 tell 10.10.10.200, length 28
14:10:47.549640 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 8.8.8.8 tell 10.10.10.200, length 28
14:10:48.551557 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 8.8.8.8 tell 10.10.10.200, length 28
14:10:50.548614 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 8.8.8.8 tell 10.10.10.200, length 28
14:10:51.549621 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has 8.8.8.8 tell 10.10.10.200, length 28
5 packets captured
5 packets received by filter
0 packets dropped by kernel
```

 # 특정 인터페이스에서(-i) 특정 IP 가 포함된 패킷(-nn, grep) 을 보여준다
tcpdump -i ens33 -nn | grep "1.2.3.4"
```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
5 packets captured
6 packets received by filter
0 packets dropped by kernel
14:28:31.524052 IP 1.2.3.4.22 > 5.6.7.8.53707: Flags [P.], seq 4138122029:4138122237, ack 1068364849, win 1432, length 208
14:28:31.525940 IP 5.6.7.8.53707 > 1.2.3.4.22: Flags [.], ack 208, win 509, length 0
```

Expression (조건식)

조건에 부합하는 패킷 정보만을 보여준다. 조건의 가장 큰 구분은 다음과 같다. 조합할 수 있다.

1) Type (패킷의 종류 조건)
- host , net , port

2) Dir (패킷의 방향 조건)
- src, dst , src or dst, src and dst

3) Proto (패킷의 프토토콜 타입 조건)
- ether, fddi, ip, arp, rarp, decnet, lat, sca, moprc, mopdl, tcp, udp

4) 그 외
- 산술 연산자 ('and(&&)', 'or(||)', 'not(!)')
- gateway, broadcast, less, greater

 # 특정 인터페이스(-i) 에서 tcp 패킷이며 Destination Port 가 22인 패킷을 IP 로(-nn) 출력하기
tcpdump -i ens33 tcp dst port 22 -nn
```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size 262144 bytes
16:50:25.964928 IP 211.115.223.215.53707 > 211.239.150.48.22: Flags [.], ack 4194846013, win 511, length 0
```

```
16:50:26.016014 IP 211.115.223.215.53707 > 211.239.150.48.22:
Flags [.], ack 161, win 511, length 0
16:50:26.060903 IP 211.115.223.215.53707 > 211.239.150.48.22:
Flags [.], ack 321, win 510, length 0
```

 # 특정 인터페이스(-i) 에서 tcp 패킷"이며" Destination Port 가 80 "이거나" 443 인 패킷을 IP 로(-nn) 출력하기

```
tcpdump -i ens33 -nn tcp and dst port 80 or dst port 443
16:59:12.840778 IP 211.239.150.48.49514 > 8.8.8.8.443: Flags
[F.], seq 5, ack 9, win 229, options [nop,nop,TS val
3924566459 ecr 3515313672], length 0
16:59:14.863982 IP 211.239.150.48.47286 > 8.8.8.8.80: Flags
[S], seq 873533324, win 29200, options [mss 1460,sackOK,TS val
3924568482 ecr 0,nop,wscale 7], length 0
16:59:15.865581 IP 211.239.150.48.47286 > 8.8.8.8.80: Flags
[S], seq 873533324, win 29200, options [mss 1460,sackOK,TS val
3924569484 ecr 0,nop,wscale 7], length 0
```

Header Flag

패킷 해더의 정의된 플래그 타입을 조건으로 한 결과값을 출력할 수 있다.

 SYN : Client 가 데이터를 보내기 전 Server 에 악수(동기)를 요청하는 가장 첫 단계
이 플래그를 통해 IN/OUT Port 통신을 체크할 수 있어서 자주 활용된다.

 # SYN 패킷만 수집 하기

= TCP Flag(tcp[13]) 중에 SYN (0x02) 인 IP (-nn) 패킷 10개를(-c) txt 타입 result.log 로 저장(>) 하기

```
tcpdump -i ens33 -c 10 "tcp[13] == 0x02" -n -nn > result.log
tcpdump: verbose output suppressed, use -v or -vv for full
protocol decode
listening on ens33, link-type EN10MB (Ethernet), capture size
262144 bytes
10 packets captured
10 packets received by filter
0 packets dropped by kernel
```

 # cat resule.log

```
16:16:15.308872 IP 211.239.150.48.43192 > 61.100.13.230.10051:
Flags [S], seq 3596242163, win 29200, options [mss
1460,sackOK,TS val 3921988927 ecr 0,nop,wscale 7], length 0
16:16:17.048128 IP 121.40.192.14.41265 > 211.239.150.48.4244:
Flags [S], seq 3887030721, win 1024, length 0
```

[S] = SYN Flag
수집한 부분을 확인할 수 있으며
json 정렬이나, 조건식을 통해 다듬을 수도 있다.

```
Unskilled Attackers Pester Real Security Folks
==============================================
                      TCPDUMP FLAGS
Unskilled =  URG  =  (Not Displayed in Flag Field, Displayed
elsewhere)
Attackers =  ACK  =  (Not Displayed in Flag Field, Displayed
elsewhere)
Pester    =  PSH  =  [P] (Push Data)
Real      =  RST  =  [R] (Reset Connection)
Security  =  SYN  =  [S] (Start Connection)
Folks     =  FIN  =  [F] (Finish Connection)
          SYN-ACK =  [S.] (SynAcK Packet)
                     [.] (No Flag Set)
```

 # JSON 을 사용한 덤프파일 정렬 예시

result.log
서버 IP: 211.239.150.48

JSON:

```
[ { ipaddr1: '221.239.150.48',
    ipaddr2: '61.100.13.230',
    type: 'Outbound',
    port: '10051' },
  { ipaddr1: '121.40.192.14',
    ipaddr2: '221.239.150.48',
    type: 'Inbound',
    port: '4244' } ]
```

Table:

ipaddr1	ipaddr2	type	port

```
221.239.150.48  61.100.13.230   Outbound  10051
121.40.192.14   211.239.150.48  Inbound   4244
```

참고 자료

TCP 3 way handshark : https://websecurity.tistory.com/93

[OS] CentOS 7 에서 iptables 로 초기 접근 제어하기

iptables 로 초기 접속 제어 하기

CentOS 7 기본 설치 후, 공인 네트워크 환경에 연결해야 할 때 우선 최소 SSH 접근만 허용하기

(Pre) CentOS 7 에서 firewalld 대신 iptables 사용하기

firewalld 사용이 권장되지만, iptables 에 익숙할 경우 iptables.target 을 service 에 연결하는 패키지를 운영할 수 있다.

```
# firewalld disable
systemctl stop firewalld && systemctl disable firewalld

# firewalld 대신 service 를 통한 운영.
# 설치 후 서비스 재시작하면 /etc/sysconfig/iptables 에 실시간 룰이 저장된다.
yum install iptables-services
service iptables reload
service iptables status

# 룰 저장
service iptables save

# 룰 리로드
```

```
service iptables reload

# 룰 형식으로 텍스트 저장. 반드시 -c ( ALL Rule ) 옵션을 지정해서 저장하여야 방화벽 정책과 ROUTE(NAT) 룰이 같이 저장된다.
iptables-save -c > rules.txt

# 룰 복원
iptables-restore < rules.txt
```

iptables 초기 설정 (모두 차단 후 특정 IP만 허용하기)

```
# 모든 룰 초기화
iptables -F

# lo 인터페이스 모든 ACCEPT
iptables -A INPUT -i lo -j ACCEPT

# 모든 접근 허용할 소스 IP 지정. 포트(SSH) 까지 지정할 경우 -p tcp (-m tcp 생략 가능) --dport 22 를 추가한다
iptables -A INPUT -s 1.2.3.4/32 -m comment --comment "회사" -j ACCEPT

# state 상태를 지정한 ACCEPT.
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

# (선택) Ping request 요청에 대해 호스트 접근 불가 응답 설정. 서버가 존재한다는 힌트를 주는 것이다.
iptables -A INPUT -j REJECT --reject-with icmp-host-prohibited

# (선택) Ping request 요청에 거절 하고 싶다면.
iptables -A INPUT -p icmp --icmp-type echo-request -j REJECT

# (선택) Ping 접근 자체를 막고 싶다면 DROP. 허용하려면 ACCEPT 를 주면 된다.
iptables -A INPUT -p icmp -j DROP

# 그 외 모든 TCP 접근 DROP
iptables -A INPUT -p tcp -j DROP

# 저장
```

```
service iptables save
```

추가 접근 정책 추가

```
# 기본 설정 이후에 -A 로 추가하면 DROP 이후의 Line 에 들어가므로 Line 순서 상 제대로 허용되지 않는다
# -I INPUT [DROP Line] 을 기입하여 DROP 라인 위로 추가되도록 한다.

iptables -nL --line-number
------------------------------------------------------------------------
------------------
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  1.2.3.4                    0.0.0.0/0
/* 회사 */
2    DROP       tcp  --  0.0.0.0/0            0.0.0.0/0
------------------------------------------------------------------------
------------------

# 2번 DROP 정책 위로 추가한다.
iptables -I INPUT 2 -s 5.6.7.8 -j ACCEPT -m comment --comment "
허용 추가"
iptables -nL --line-number
------------------------------------------------------------------------
------------------
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     all  --  1.2.3.4                    0.0.0.0/0
/* 회사 */
2    ACCEPT     all  --  5.6.7.8                    0.0.0.0/0
/* 허용 추가 */
3    DROP       tcp  --  0.0.0.0/0            0.0.0.0/0
------------------------------------------------------------------------
------------------

# 라인 정렬이 어려울 경우 /etc/sysconfig/iptables 파일을 직접 수정 후 reload 가능하다.
```

라인 정책 삭제

```
iptables -D INPUT [Number]
```

원격으로 CentOS 7 에서 내부 Windows 서버에 RDP 연결

CentOS 7 에서 Windows RDP 연결하기

진행 순서

```
# OS
CentOS 7.9 x86_64 minimal
로컬--> XRDP 를 통한 리눅스 GUI 접속 --> 리눅스 GUI 에서 내부 Windows RDP 접속
```

Linux GUI 환경 세팅

```
# GUI GroupInstall
root@localhost ~]# yum groups list | grep -i desktop
   Cinnamon Desktop
   MATE Desktop
   GNOME Desktop
   General Purpose Desktop
   LXQt Desktop
# GNOME 이 제대로 설치되지 않는다면 "Server with GUI" 그룹을 설치한다.
root@localhost ~] yum groupinstall "GNOME Desktop"

# GUI init
[root@localhost ~]# systemctl get-default
multi-user.target
[root@localhost ~]# systemctl set-default graphical.target
```

```
[root@localhost ~]# systemctl get-default
graphical.target
# Reboot 후 자동으로 GUI 모드로 부팅되는지 확인한다.
[root@localhost ~]# reboot
```

Linux 서버에 원격 접속 설정

```
# XRDP Install.
[root@localhost ~]# yum install epel-release
[root@localhost ~]# yum install xrdp
[root@localhost ~]# systemctl enable xrdp && systemctl start
xrdp

# selinux disable 및 iptables -F or  방화벽 정책에 tcp/3389 를 추가
해준다.
```

rdesktop 설치

```
# 컴파일러와 openssl-devel 이 패키지 컴파일 선행조건이다.
 yum -y install gcc openssl-devel
                                                          wget
https://github.com/rdesktop/rdesktop/releases/download/v1.8.6/
rdesktop-1.8.6.tar.gz
 tar xvzf rdesktop-1.8.6.tar.gz
 cd rdesktop-1.8.6/
 ./configure --disable-credssp --disable-smartcard
 make
 make install
```

Check

```
# 리눅스 서버에 RDP 로 접속 후, rdesktop -u [User] [ip]  입력한다.
root@localhost ~]# rdesktop -u administrator 10.10.10.5
Autoselected keyboard map en-us
Connection established using SSL.
WARNING: Remote desktop does not support colour depth 24;
falling back to 16
```

CentOS 7 설치 후 할 일 – 필수 패키지 및 권장 패키지 설치 (2)

CentOS 7 패키지 설치

CentOS 7 기본 설정 후 서버 관리 및 운영을 도와주는 몇 가지 패키지들을 설치해주어야 한다.

해당 포스트에서는 필수 패키지 / 권장 패키지로 나누어 설명하도록 하겠다.

1. 패키지 업데이트

커널 및 각종 패키지들이 업데이트 되며 서버의 구성환경을 고려하여 진행한다.

```
## 패키지명을 지정하지 않을경우 설치되어있는 모든 패키지를 업데이트 한다.
yum update [패키지명]
## 설치된 패키지 중 업데이트된 패키지 목록 확인
yum list updates
```

2. 필수 패키지

일반적으로 설치해야하는 패키지들과 서버 모니터링을 위한 패키지로 구분하여 설명한다.

일반 패키지

epel-release : 각종 패키지의 최신 버전을 사용하기 위해 설치
해당 패키지 설치후 다른 패키지 설치시에 최신 버전으로 설치되므로 disable 설정 후 enablerepo 옵션 사용

vi /etc/yum.repos.d/epel.repo
enabled=1 을 0으로

[epel]
...
enabled=0
...

yum repolist
yum --enablerepo=epel install [패키지명]

net-tools : 각종 네트워크 명령어들을 관리하는 패키지로 IP 확인을 위한 ifconfig, 네트워크 연결 상태 확인을 위한 netstat 둘을 포함한 명령어를 사용하기 위해 설치한다.

<ifconfig>
ifconfig -a
ifconfig [interface] up
ifconfig [interface] down

<netstat>
netstat -nap
netstat -an | grep [Port]
netstat -nlpt

unzip : zip으로 압축된 파일을 풀기 위해 사용한다.

unzip [file_name].zip
unzip -l [file_name].zip
unzip -t [file_name].zip

wget : 웹 서버로부터 콘텐츠를 가져오는 명령어로 HTTP, HTTPS, FTP 프로토콜클 지원한다.

```
wget -O [파일 이름] [URL 주소]
wget --no-check-certificate [URL 주소]
```

curl : 커맨드 라인용 데이터 전송 도구로 HTTP, FTP등 다양한 프로토콜을 지원한다.
Web 파일 다운받을 경우 wget 권장

```
curl -o [파일 이름] [URL 주소]
curl -T [파일 이름] [서버 IP]
curl -L [URL 주소]
```

chrony : NTP를 구현한 Server/Client로 ntpd를 대체하며 서버의 시간을 동기화한다.

```
## 동기화 시킬 NTP 서버의 IP 추가
vi /etc/chrony.conf

server [NTP server IP] iburst <iburst = 동기화 시간을 단축 시켜주는 옵션>

systemctl restart chronyd
chronyc sources
```

gcc, gcc-c++ : 각각 C언어, C++언어 컴파일러로 서버에 개발환경을 구축하기 위해 설치한다.

openssl-devel : openssl을 설치하기 위한 의존성 패키지로 openssl을 사용할 애플리케이션 개발용 파일이다.

서버 모니터링 관련 패키지

htop : 리눅스에서 사용중인 리소스등을 확인 (윈도우의 작업 관리자와 비슷한

역할을 함)

프로세스

PID
USER
PR
NI
VIRT
RES
SHR
S
%CPU
%MEM
TIMR+
COMMAND

iftop : 네트워크 인터페이스의 트래픽 모니터링

```
iftop -i eno1
iftop -f "dst port 22"
```

dstat : 실시간으로 디스크별 I/O 사용량 체크 및 부하 유발을 주는 프로세스를 쉽게 확인 가능하다.

```
dstat -tcdml
```

sysstat : 리눅스 성능 측정 도구 패키지로 sar, iostat 등을 사용하기 위해 설치

<iostat> : CPU와 디스크의 처리량, 대기열 길이등을 실시간으로 모니터링 한다.

```
iostat -d 3
iostat -c 3
```

<sar> : 과거 모니터링 정보를 확인 /var/log/sa 폴더에 기록된 sa파일들을 조회한다.

```
sar -u
sar -r
sar -dp
sar -n DEV
```

lsof : 시스템에서 열려있는 파일에 대한 정보를 출력

lsof -u [사용자 이름]
lsof -i
lsof -c [서비스 이름]

psmisc : proc 파일 시스템을 사용하는 유틸리티 패키지로 fuser, killall, pstree등을 사용

<fuser> : 특정 파일시스템을 umount 할 때 어떠한 프로세스 때문에 진행이 안될 경우, 그 프로세스를 kill 하기 위해 사용
fuser -v [디렉터리]
fuser -ck [디렉터리]
<killall> : 특정 프로세스를 한 번에 죽이기 위해 사용
killall -i [프로세스]
killall -v [프로세스]
killall -w [프로세스]

<pstree> : 프로세스의 상태를 Tree 구조로 출력하여 확인
pstree -anp

3. 권장 패키지

mlocate : 파일을 검색하기 위해 find 명령어 대신 데이터베이스 기반으로 파일을 검색하기 때문에 빠른 검색을 위해 사용한다.

updatedb
locate [파일 이름]
locate -n [숫자] [파일 이름]

ncat : 서버와 클라이언트가 소켓 통신을 하기위해 사용한다.

<서버>
ncat -l [Port]
ncat -lk [Port]

<클라이언트>
ncat [Server IP] [Port]

whois : 도메인 소유자나 IP 주소 위치 등 이외 다른 정보를 확인할 때 사용한다.

whois [도메인 주소]
whois [IP 주소]

cloud-utils-growpart : LVM 구성이 아닌 상태에서 root 파티션 용량을 추가하기 위해 사용한다.

growpart [디스크 명] [파티션 번호]
resize2fs [파티션 경로]

tcping : 클라이언트에서 서버의 TCP 포트를 ping 유틸리티로 체크하기 위해 사용한다.

tcping [Server IP] [Port]

About OOM Killer ?

본 글은 Kernel 5.4.0-104-generic 을 기반으로 작성하였습니다.

OOM(Out Of Memory) 이란 ?

Linux의 swap 메모리와 물리 메모리를 모두 사용 중 일 때, 프로세스가 추가적인 물리 메모리 할당을 요청 하여도 할당이 불가능 한 상태를 의미한다.
주로 Over Commit 상태일 때 해당 현상이 동반된다.

OOM(Out Of Memory) Killer 란?

Linux 의 메모리 부족으로 Out Of Memory가 발생 하였을 때, OOM Score 를 기준으로 점수가 높은 프로세스를 Kill 하여 메모리 확보를 하는 Linux Kernel 기능이다.
OOM Killer 로 인한 프로세스 강제 종료 시 /var/log/ 내부의 시스템 로그에서 확인 할 수 있다.

oom_killer 로 인한 프로세스 종료 Log

```
$ cat /var/log/syslog | grep oom
 Mar  7 19:14:00 zabbix-node01 kernel: [1132818.054201] ib_log_writer invoked oom-killer: gfp_mask=0x100cca(GFP_HIGHUSER_MOVABLE), order=0, oom_score_adj=0
```

OOM Score 점수 산정 및 OOM Killer 구동의 직접적인 영향을 주는 요인을 알아보자.

1. 프로세스에 할당 된 메모리 양 + 프로세스로 부터 fork() 된 자식 프로세스의 메모리 양
2. 프로세스의 동작 중 인 시간
3. 프로세스가 하드웨어에 직접 접근하거나, root계정 (super user) 에 의해 실행 된 프로세스
4. 프로세스의 nice 값이 1 이상일 경우 Score 점수 2배 증가
5. /proc/[PID]/oom_score_adj 값 (유저가 직접 변경 가능)
6. /proc/[PID]/oom_adj 값 (유저가 직접 변경 가능)

프로세스의 OOM Score를 확인하는 방법은 다음과 같다.

```
# oom_score 확인 방법
$ cat /proc/890081/oom_score
1048
```

890081은 PID 이며, 현재 OOM Score는 1048이다.

oom_adj / oom_score_adj 에 대한 부연 설명

종료되면 안되는 주요 프로세스가 OOM Score 가 높아 OOM Killer에 의한 종료가 예상 될 때에는, oom_adj /oom_score_adj 값의 조절을 통하여, OOM Score 조절이 가능하다.

oom_adj 의 범위는 -17 ~ 15 이며, -17로 지정 시 OOM Killer Disable과 같은 의미를 가진다.

oom_score_adj의 범위는 -1000 ~ 1000 이며, 현재 OOM Score 에서 oom_score_adj 값을 가감한다.(oom_scoer_adj 의 -1000은 oom_adj 의 -17과 같다.)

oom_adj / oom_score_adj 의 값을 변경 시 변경 한 값을 참조하여 oom_score의 재 산정이 이루어진다.

프로세스의 OOM Score 확인 및 oom_score_adj 를 통한 OOM Score 조절은 아래 예를 참고하자.

```
# 현재 oom_score 확인
$ cat /proc/890081/oom_score
1048
```

890081은 PID 이며, 현재 OOM Score는 1048이다.

```
# oom_score_adj 를 통한 OOM Score 조절
$ echo -1000 > /proc/890081/oom_score_adj

$ cat /proc/890081/oom_score_adj
-1000
# oom_score_adj 값에 가장 낮은 값인 -1000 으로 설정

# 설정 후 oom_score / oom_adj 확인
$ cat /proc/890081/oom_score
0
$ cat /proc/890081/oom_adj
-17
# oom_score_adj 설정 후 확인 결과, oom_adj 는 -17 (OOM Killer Disable)
# oom_score는 1048 -> 0 으로 감소한 것을 확인할 수 있다.
```

추가적으로, overcommit에 대한 설정을 할 수 있다. over commit의 설정 방법은 다음과 같다.

over commit 설정

```
$ cat /etc/sysctl.conf | grep overcommit_memory
vm.overcommit_memory = 1 # 0~2 의 설정값을 가진다.
# 0 = Heuristic overcommit. Default 설정 값이며, 휴리스틱하게(유동적이게) 커널 자체적으로 over commit 비율을 조정한다.
# 1 = 항상 over commit을 허용한다. OOM Killer가 동작하지 않으나, 프로세스의 메모리할당에 실패 할 수 있다.
# 2 = vm.overcommit_ratio 의 값을 참조하여 over commit 비율을 조정한다.

# 사용 가능한 물리 메모리의 백분율
$ cat /etc/sysctl.conf | grep overcommit_ratio
vm.overcommit_ratio = 90
# 해당 설정은 vm.overcommit_memory가 2일때 의미를 가진다.
# 90%의 물리메모리 사용 + swap 메모리 사용을 초과하게 되면 OOM Killer가 동작한다.

# 설정 적용
$ systemctl -w
```

그렇다면, 현재 commit 되고있는 메모리와 허용가능한 over commit 메모리양은 어떻게 확인할까?
답은 /proc/meminfo 과 sysstat 패키지로 조회할 수 있다. 하기 예를 참고하자.

```
# commit 정보 확인
$ cat /proc/meminfo | grep Commit
CommitLimit:    30229156 kB
# vm.overcommit_memory 의 설정이 2 일 때 vm.overcommit_ratio 의 설정을 기반으로 commit 가능 한 메모리 양
Committed_AS:   64267776 kB
# 현재 commit 되고 있는 메모리 양
```

sysstat 패키지를 이용한 현재 commit 양 확인

```
$ apt install -y sysstat
# sysstat 패키지 설치

# sar 명령어를 이용한 메모리 상태 조회
$ sar -r 1
Linux 5.4.0-104-generic (zabbix-node02)         04/14/2022      _x86_64_        (16 CPU)

09:25:17 AM kbmemfree   kbavail kbmemused  %memused kbbuffers kbcached  kbcommit   %commit  kbactive   kbinact   kbdirty
09:25:18 AM    208884   2446760  20804112     85.73    990436   1441416  64262056    196.79  19538260   3304352      1508
# %commit 부분이 현재 commit된 메모리 양을 백분율로 표기해주는 부분이다.
# 현재 100% 이상 commit 되고 있으므로, over commit 상태이다.
# sar 명령어는 시스템 자원(CPU, Memory, I/O) 사용량을 수집, 레포트 하는 명령어다.
# -r 옵션은 Memory 사용량을 조회하는 옵션이며, 뒤의 1은 1초 마다 수집한다는 의미이다.
Linux Memory Commit / Memory Over Commit 이란?
Memory Commit 이란
프로세스가 커널에게 메모리 공간을 요청 하여 가상 메모리 영역 주소를 할당 받은 상태

이해를 위해 아래 예를 참고 하자.

A 프로세스가 커널에게 메모리 할당을 요청 하였다고 가정 했을 때,
커널은 이에 해당하는 가상 메모리 주소를 A 프로세스에게 할당한다.
결과적으로 A프로세스는 커널에게 메모리 영역을 할당 받았다고 생각하지만, 실제론 물리 메모리에 할당 받지 않은 상태다.
이를 Memory Commit (메모리 커밋) 이라고 한다.
그렇다면, 왜 바로 물리 메모리 영역에 프로세스를 할당하지 않고, 가상 영역에 할당을 할까?
```

물리 메모리 영역에 프로세스를 바로 할당하게 되면 몇가지 문제점이 있기 때문이다. 프로세스의 메모리 주소 할당을 바로 물리 메모리 주소로 할당 하면 생길 수 있는 문제점은 다음과 같다.

1. A 프로세스가 메모리를 할당받고 사용하지 않을 수 있다.
2. 이미 사용중인 메모리 주소값으로 접근 할 수 있는 위험성이 있다.
 - 예로, 이미 커널이 사용 중 인 물리 메모리 주소가 반환될 경우 매우 위험
3. 메모리 단편화(Fragmentation)가 발생한다.
 - 메모리 단편화 : RAM 에서 메모리 공간이 작은 공간으로 나뉘어 사용가능한 메모리가 존재하지만 할당이 불가능한 상태
 - 메모리 단편화 발생 사유 및 해결방법에 대한 자세한 내용은 다음에 다루도록 한다.

Memory Over Commit이란
말 그대로, 리눅스의 실제 물리 Memory보다 Over(초과) 하여 Commit 된 상태를 의미한다.
결론적으로 over commit이 발생한다고 하더라도, 시스템에 문제가 생기진 않는다.
하지만, over commit의 양이 지속적으로 증가한다면 oom killer로 중요 프로세스가 갑자기 종료될 수 있다.

그러니 시스템의 설정과 프로세스가 요구하는 메모리를 잘 파악하고 메모리 누수가 없는지 주기적인 점검이 필요하다.

CentOS 7에서 APM 최신 버전 yum으로 설치하기

고정적인 버전 관리가 필요하지 않거나 간단한 설치가 필요할 경우 yum 을 이용할 수 있습니다.

Image OS : CentOS 7.6.1810 Minimal

공통사항
추가 패키지를 설치하기 위해 우선 확장 레포지토리를 설치해 줍니다
```
yum install -y epel-release
```

Apache 2.4.52
아파치는 보통 빌드를 통해 버전 관리용 RPMs 패키지를 만듭니다.
우선 외부 레포지토리(codeit)를 사용합니다.

```
cd /etc/yum.repos.d/ && wget https://repo.codeit.guru/codeit.el`rpm -q --qf "%{VERSION}" $(rpm -q --whatprovides redhat-release)`.repo

# 사용가능 버전 확인
yum info httpd

Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
 * base: mirror.kakao.com
 * epel: hk.mirrors.thegigabit.com
 * extras: mirror.kakao.com
 * remi-safe: mirror.bebout.net
 * updates: mirror.navercorp.com
Installed Packages
Name        : httpd
Arch        : x86_64
Version     : 2.4.52
Release     : 1.codeit.el7
Size        : 4.3 M
Repo        : installed
From repo   : CodeIT
Summary     : Apache HTTP Server
URL         : https://httpd.apache.org/
License     : ASL 2.0
Description : The Apache HTTP Server is a powerful, efficient, and extensible
            : web server.

# Apache 설치
yum --enablerepo=CodeIT install httpd mod_ssl

# PHP 7.4 설치를 위한 Remi Repository 를 설치합니다.
yum install https://rpms.remirepo.net/enterprise/remi-release-7.rpm

# 전체 PHP 버전 리스트를 확인 후 7.4 버전 설치
yum repolist all | grep -i php
yum --enablerepo=remi-php74 install php  php-opcache php-gd php-mysql php-xml

# MariaDB 10.3 설치를 위한 공식 레포지토리 설정
cat << EOF | tee /etc/yum.repos.d/MariaDB.repo
```

```
[mariadb]
name = MariaDB
baseurl = http://yum.mariadb.org/10.3/centos7-amd64
gpgkey=https://yum.mariadb.org/RPM-GPG-KEY-MariaDB
gpgcheck=1
enabled=0
EOF


# MariaDB 10.3 설치
yum -y install --enablerepo=mariadb MariaDB-server MariaDB-client MariaDB-backup


# APM 버전 확인
[root@localhost ~]# php -v
PHP 7.4.28 (cli) (built: Feb 15 2022 13:23:10) ( NTS )
Copyright (c) The PHP Group
Zend Engine v3.4.0, Copyright (c) Zend Technologies
     with Zend OPcache v7.4.28, Copyright (c), by Zend Technologies
[root@localhost ~]# mysql --version
mysql  Ver 15.1 Distrib 10.3.34-MariaDB, for Linux (x86_64) using readline 5.1
[root@localhost ~]# httpd -v
Server version: Apache/2.4.52 (codeit)
Server built:   Dec 20 2021 11:29:54
```

CentOS 7 설치 후 할 일 – 파티셔닝 및 설정 (1)

CentOS 7 설치 후 할 일

실제로 서비스를 제공하는 서버라면 몇 가지 기본적인 설정, 패키지, 보안에 신경을 써줘야 한다.

파티셔닝 및 설정, 필수 및 권장 패키지 설치, 서버 운영을 위한 보안 설정(하드닝) 세가지로 분류해 작성한다.

파티셔닝

LVM : Default로 설정되어 있으며 스토리지 확장 시 유연하게 변경 가능, 관리 난이도가 높아 일반적으로 **표준 파티션** 사용 권장

표준 파티션 : 디스크 성능이 좋아짐에 따라 속도 면에서 **LVM**과 크게 차이가 없음, xfs 형식 사용하여 이전보다 파티션 조절 수월함

1. 네트워크 설정

(OS 설치 과정에서 네트워크 기설정되어 있는 경우 해당 내용 생략)

```
# ip addr
# vi /etc/sysconfig/network-scripts/ifcfg-eth0

. . .
BOOTPROTO=none
. . .
IPV6INIT=no
IPV6_AUTOCONF=no
IPV6_DEFROUTE=no
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
. . .
ONBOOT=yes                                  # 부팅/네트워크 재시작 시  자동으로 인터페이스 활성화를 위해  yes  설정
IPV6_PRIVACY=no
IPADDR=192.168.122.243
NETMASK=255.255.255.0
GATEWAY=192.168.122.1
DNS1=8.8.8.8
DNS2=8.8.4.4
```

```
# systemctl restart network
# ip addr
eth0 영역 IP 확인



# ping -c 4 google.com                # 네트워크 상태 확인
--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3094ms
rtt min/avg/max/mdev = 82.692/83.010/83.554/0.492 ms
```

2. 바이오스 타이머 조정

CentOS 7 부터 timedatectl 명령어로 시간, 타임존 설정을 할 수 있다.

```
# timedatectl

      RTC time : 하드웨어 시간
      NTP enabled : NTP 활성화 여부
      NTP synchronized : NTP 동기화 여부
      RTC in local TZ : RTC 와 Time zone 시간 동기화 여부

# timedatectl list-timezones | grep -i Asia*

# timedatectl set-timezone Asia/Seoul

# timedatectl
```

3. Hostname 설정

CentOS 7 설치 후 기본적으로 hostname이 localhost.localdomain 으로 설정되어있으므로 변경해주어야 함

```
# hostnamectl
Static hostname: localhost.localdomain
```

```
# hostnamectl set-hostname newhostname

# hostnamectl
Static hostname: newhostname
```

4. SELinux 비활성화

SELinux는 기본적으로 활성화 되어있으나, 해당 메뉴얼에서는 disabled(=미사용) 설정한다.

```
# vi /etc/sysconfig/selinux

. . .
SELINUX=disabled     . . .

# shutdown -r now

# getenforce
Disabled
```

5. root 계정 원격 접근 불가 설정

보안을 위해 root로의 원격 접속은 거부하고 타 계정을 사용하여 su 명령어를 통해 사용하도록 한다.

```
# ps -ef | grep sshd
# systemctl enable sshd

# vi /etc/ssh/sshd_config
. . .
PermitRootLogin=no
. . .

# systemctl restart sshd
```

6. 자동 로그아웃 설정

보안을 위해 일정 시간 응답이 없으면 자동으로 로그아웃 하도록 설정한다.

```
# vi /etc/profile.d/timeout.sh
TMOUT=600
export TMOUT

chmod +x /etc/profile.d/timeout.sh

# source /etc/profile
# echo $TMOUT
600
```

7. 히스토리 포맷 설정

명령어의 내역을 출력하는 명령어인 history의 날짜 및 시간을 확인하기 위해 설정한다.

```
# vi /etc/profile.d/history.sh

HISTTIMEFORMAT="%F %T -- "
export HISTTIMEFORMAT


# chmod 644 /etc/profile.d/history.sh
# source /etc/profile.d/history.sh

# hisotry
999  2022-04-06 14:50:10 -- vi /etc/profile.d/history.sh
1000  2022-04-06 14:50:19 -- chmod 644 /etc/profile.d/history.sh
1001  2022-04-06 14:50:28 -- source /etc/profile.d/history.sh
1002  2022-04-06 14:50:30 -- history
```

8. 문자셋 설정

설치시 언어와 문자셋을 잘못 선택했을 경우에 설정한다.

```
# localectl
  System Locale: LANG=en_US.UTF-8
      VC Keymap: us
     X11 Layout: us

# localectl list-locales | grep -i kr
ko_KR
ko_KR.euckr
ko_KR.utf8

# localectl set-locale LANG=ko_KR.UTF-8
# localectl set-keymap kr
# localectl set-x11-keymap kr

# localectl
   System Locale: LANG=ko_KR.UTF-8
       VC Keymap: kr
      X11 Layout: kr
```

다음 장에선 필수적으로 설치해야할 패키지 및 권장 패키지들에 대해서 설명한다.