2주차 교육 세미나

2023. 07. 25
DQA1-3 이윤재

I. 지난주 피드백

II. 리눅스 주요 명령어 및 OS 개요

III. Database 서버 구축과 운영

Ⅰ. 지난주 피드백

- Windows와 리눅스/유닉스 차이
- Socket과 Service
- Dynamic view와 Static View
- Git, SVN의 Commit Hash와 명령어

Windows와 리눅스 차이

다중 사용자 체제

리눅스는 다중 사용자 체제이므로 관리자 권한(root)으로 로그인 하지 않으면 모든 사용자는 보호 모드에서 작동하므로, 윈도우에 비해 보안성이 높다.

소유자 제외 모든 접근 제한. →

moti로 로그인 →

moti0 디렉토리의 소유자가 아니기 때문에 접근 제한 →

```
[root@localhost userHome]# ls -al
total 12
drwxr-xr-x.  4 root  root  4096 Jul 21 11:10 .
dr-xr-xr-x. 18 root  root   240 Jul 21 11:07 ..
drwxr-xr-x.  2 moti  moti  4096 Jul 21 11:10 moti
drwx------.  2 moti0 moti0 4096 Jul 21 11:08 moti0
[root@localhost userHome]# su moti
bash-4.2$ ls
moti  moti0
bash-4.2$ cd moti0
bash: cd: moti0: Permission denied
bash-4.2$ cd moti
bash-4.2$ ls
bash-4.2$ touch hello.txt
bash-4.2$ ls
hello.txt
bash-4.2$ ls -al
total 8
drwxr-xr-x. 2 moti moti 4096 Jul 21 11:13 .
drwxr-xr-x. 4 root root 4096 Jul 21 11:10 ..
-rw-r--r--. 1 moti moti    0 Jul 21 11:13 hello.txt
```

Windows와 리눅스 차이

접근 허가권

디렉토리	소유자			그룹			그 외 사용자		
d	r	w	x	r	w	x	r	w	x

d : directory
r : read
w : write
x : execute

```
[root@localhost userHome]# ls -al
total 12
drwxr-xr-x.  4 root  root  4096 Jul 21 11:10 .
dr-xr-xr-x. 18 root  root   240 Jul 21 11:07 ..
drwxr-xr-x.  2 moti  moti  4096 Jul 21 11:10 moti
drwx------.  2 moti0 moti0 4096 Jul 21 11:08 moti0
[root@localhost userHome]# su moti
bash-4.2$ ls
moti  moti0
bash-4.2$ cd moti0
bash: cd: moti0: Permission denied
bash-4.2$ cd moti
bash-4.2$ ls
bash-4.2$ touch hello.txt
bash-4.2$ ls
hello.txt
bash-4.2$ ls -al
total 8
drwxr-xr-x. 2 moti moti 4096 Jul 21 11:13 .
drwxr-xr-x. 4 root root 4096 Jul 21 11:10 ..
-rw-r--r--. 1 moti moti    0 Jul 21 11:13 hello.txt
```

Dynamic View와 Static View

Dynamic Performance View

- 내부 디스크 구조와 메모리 구조에 대한 데이터를 제공하며 데이터베이스가 오픈되어 사용중인 동안 계속 갱신된다.
- 메모리 구조에 데이터를 저장하고 있고, 인스턴스가 다시 시작하면 사라진다.
- V$로 시작하는 뷰이고, x$로 시작하는 테이블의 정보를 바탕으로 이루어졌다.
- 대표적인 view : v$sysstat, v$log

Dynamic View와 Static View

V$sysstat

- 인스턴스 기동 시 현재까지 누적된 수행 통계치를 시스템 레벨로 확인하고자 할 때 사용하는 뷰
- V$sysstat 값은 누적된 값으로 두 구간 사이의 변화량을 구해 SQL 수행 도중에 내부적으로 어떤 일들이 발생했는지 판명해야 한다.

```
SQL> select name, value
  2  from v$sysstat
  3  where statistic#
  4  in (7, 47, 50, 54, 134, 343, 344, 345, 349, 350);

NAME                                                                  VALUE
---------------------------------------------------------------- ----------
recursive calls                                                       214215
db block gets                                                         160025
consistent gets                                                        92662
physical reads                                                         10629
redo size                                                           18591264
bytes sent via SQL*Net to client                                       55306
bytes received via SQL*Net from client                                 24912
SQL*Net roundtrips to/from client                                        137
sorts (memory)                                                          6293
sorts (disk)                                                               0

10 개의 행이 선택되었습니다.
```

Dynamic View와 Static View

Static Data Dictionary View

- 데이터 딕셔너리에 변경(예를 들어, 테이블 생성)이 이뤄져야만 변하기 때문에 Static View라고 한다.
- 그에 반해 Dynamic performance view는 데이터베이스 활동을 모니터링한다.

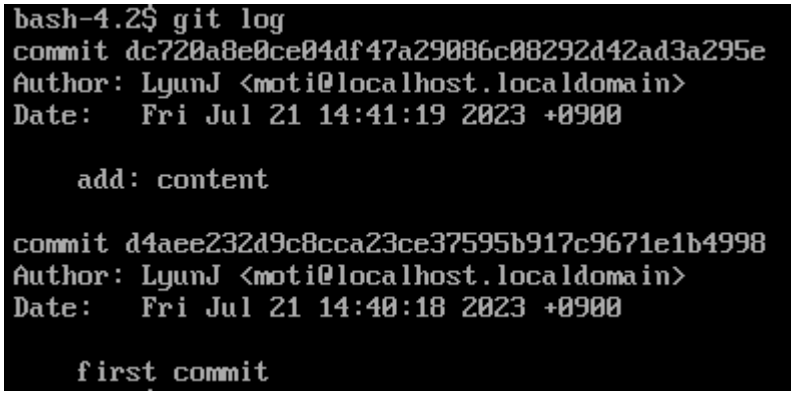
종류:

- ALL_ 뷰는 현재 유저가 접근할 수 있는 모든 스키마에 대한 정보를 보여준다.
- DBA_ 뷰는 데이터베이스 전체의 정보를 보여준다. 이 뷰는 DBA role을 가진 사람이 확인할 수 있다.
- USER_ 뷰는 현재 유저의 스키마에 대한 정보를 보여준다

Git, SVN의 commit hash와 명령어

Git의 commit hash

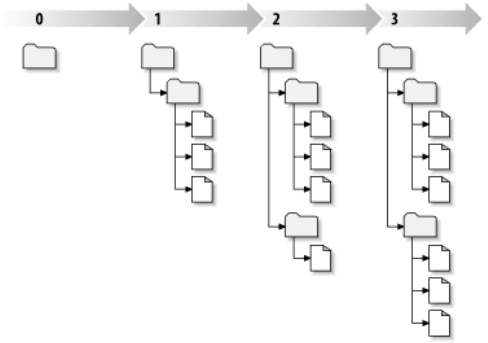
- Commit을 식별하기 위해 SHA1 알고리즘으로 만든 해시 값.
- 정수형 번호를 할당하면 오프라인에서 commit을 할 때 중복 검사를 할 수 없어 해시 값을 활용.
- 부모 commit의 해시 값을 함께 저장함으로써 commit 순서를 기억할 수 있음

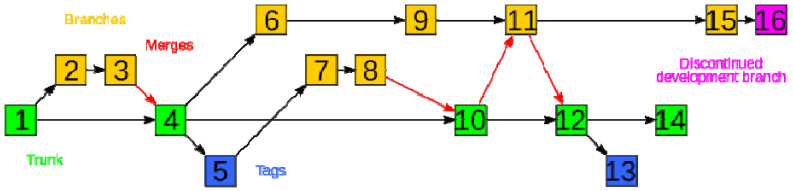


Git, SVN의 commit hash와 명령어

SVN의 revision number

- 레포지토리가 commit을 받아들이면, revision이라고 불리는 새로운 파일시스템 트리의 상태를 만들어낸다.
- 각각의 revision은 이전 revision보다 1이 증가한 자연수이다.
- Revision number는 각각의 파일이 아닌 트리 전체에 할당된다.

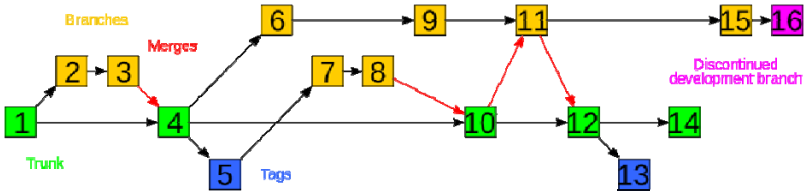




Git, SVN의 commit hash와 명령어

SVN의 용어

- Repository : 프로젝트의 파일 및 변경 정보가 저장되는 장소
- Trunk : 개발 소스를 commit 했을 때 개발 소스가 모이는 곳(메인 개발 소스)
- Branch : Trunk에서 분기된 개발 소스
- Tag : 특정 시점에서 프로젝트의 스냅샷을 찍어두는 것.
    - Branch와 tag는 사실 동일하지만, tag는 관례적으로 더 이상 개발하지 않고 어떤 버전으로 얼려두는 것.



Git, SVN의 commit hash와 명령어

SVN의 명령어

- Checkout[co] : 원격 저장소에서 최신 버전의 소스코드를 최초로 받아오는 것
  - svn **checkout[co]** svn://127.0.0.1/TestRepo1 LocalRepo

- Import : 아무것도 들어있지 않은 원격 저장소에 처음으로 파일을 업로드 할 때 한번만 사용
  - svn **import** sampledir svn://127.0.0.1/TestRepo1/trunk

- Export : 버전 관리 파일들을 뺀 순수한 파일들만 빼내는 것
  - svn **export** svn://127.0.0.1/TestRepo1

- Update[up] : 로컬 저장소에 있는 파일들을 원격 저장소의 최신 버전으로 받아오는 것.
  - svn **update[up]**
  - svn **update[up]** –r 1 (최신 리비전보다 이전으로 되돌리는 것도 가능)

Git, SVN의 commit hash와 명령어

SVN의 명령어

- add : 버전관리 대상으로 파일을 등록하는 것(add 후 commit 해야 적용된다)
  - svn **add** main.c

- commit[ci] : 로컬 저장소의 변경된 내용을 서버로 전송하는 것(revision 수가 증가)
  - svn **commit[ci]** -m "수정사항에 대한 메시지 입력"

- status : 로컬 저장소에서 변경된 이후 아직 저장소로 commit 되지 않은 내용을 확인
  - svn **status[stat, st]** main.c

- log : revision 로그 보기
  - svn log

Ⅱ. 실습 환경 구축

- 가상머신과 가상머신 소프트웨어 개념
- 가상머신 소프트웨어 설치 및 리눅스 설치

가상머신과 가상머신 소프트웨어 개념

가상머신

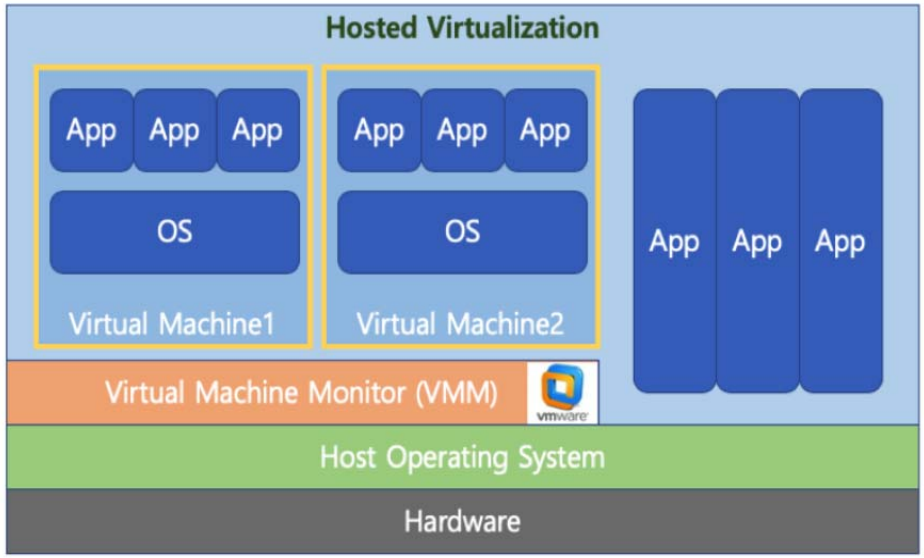
가상머신은 물리적으로 존재하는 컴퓨터가 아닌, 다른 컴퓨터가 만들어내는 가상의 컴퓨터를 말한다
.

- 가상머신을 활용하면 하나의 컴퓨터에서 가상의 컴퓨터를 여러 대 만들 수 있고, 서로 독립적인 공간에서 작업할 수 있다.

가상머신과 가상머신 소프트웨어 개념

가상머신의 구조

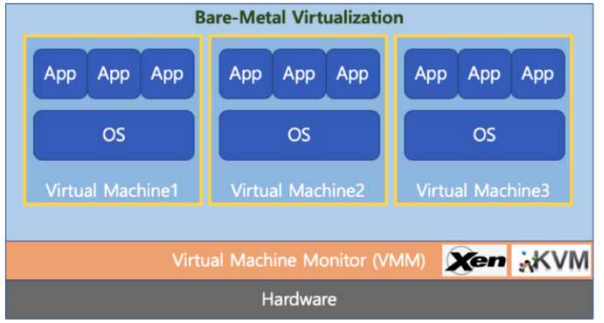
- 호스트의 운영체제 위에 VMM이 설치됨
- VMM은 운영체제 위에서 가상머신의 요청을 운영체제로 보내는 역할을 함.
- VMM은 VM에게 독립적인 환경을 제공하고, VM이 사용하는 자원의 사용 계획을 설정함.

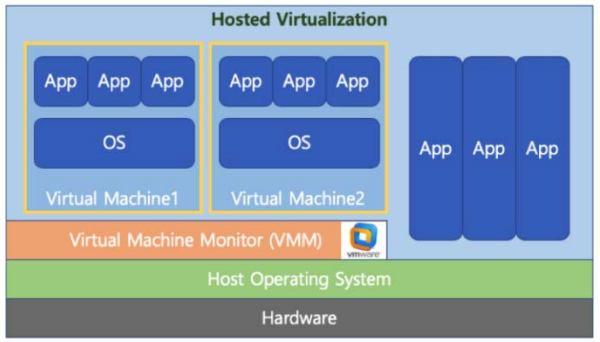


가상머신과 가상머신 소프트웨어 개념

가상머신의 유형

- 하드웨어 위에 바로 VMM이 올라간 타입 (Type1)과 OS위에 VMM이 올라간 타입 (Type2)으로 나뉨
- Type1의 VMM이 하드웨어 통제권을 가져, 거쳐야 하는 것들이 적기 때문에 더 빠를 수 있음
- Type2는 안정성에 문제가 생겨도 Host OS 에는 영향을 주지 않음.



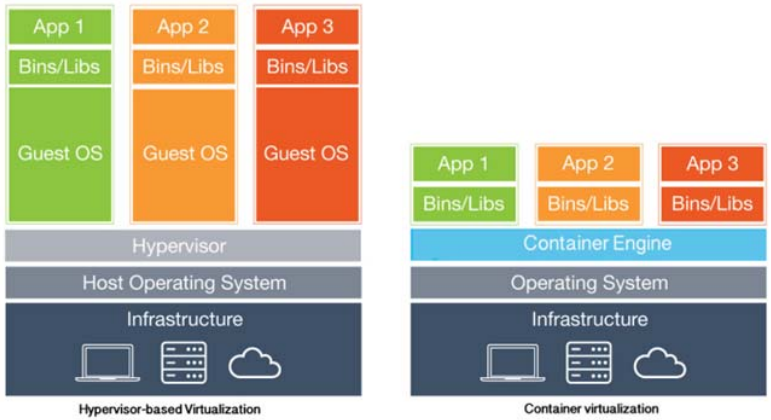


가상머신과 가상머신 소프트웨어 개념

컨테이너

컨테이너는 모듈화되고 격리된 컴퓨팅 공간 또는 컴퓨팅 환경, 다시 말해 어플리케이션을 구동하는 환경을 격리한 공간을 의미

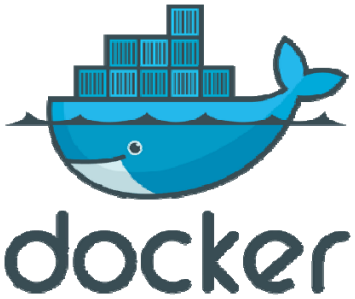
- 가상머신에서 게스트 OS가 필요했던 것과 달리, 컨테이너는 운영체제를 공유하고 어플리케이션 실행에 필요한 모든 파일만을 패키징한 형태.
- 가상머신에 비해 가볍고 빠르게 동작이 가능



가상머신과 가상머신 소프트웨어 개념

가상머신 소프트웨어 종류

- 가상머신
  - Type1
    - Hyper-V
  - Type2
    - Vmware
    - Virtual Box
- 컨테이너
  - 도커



▪ 가상머신 소프트웨어 설치 및 환경 구축

Vmware 설치 및 환경 구축

- 티베로 설치를 위해 티베로 하드웨어 요구사항을 참고하였음

- OS : CentOS7
- 메모리 : 4GB
- 보조 메모리 : 40GB
- 스왑 공간 : 4 GB

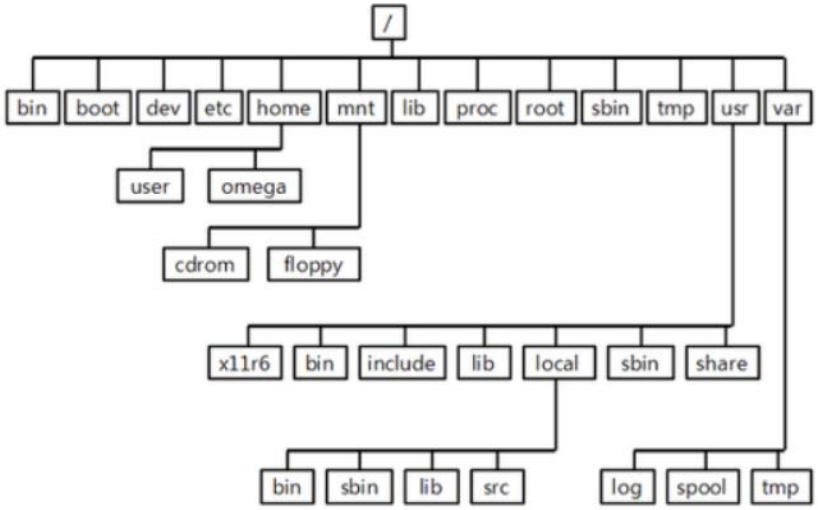
Device	Summary
Memory	4 GB
Processors	2
Hard Disk (SCSI)	40 GB
CD/DVD (IDE)	Using file autoinst.iso
CD/DVD 2 (IDE)	Using file C:₩Users₩tedle₩...
Network Adapter	NAT
USB Controller	Present
Sound Card	Auto detect
Printer	Present
Display	Auto detect

Platform	RAM	Swap Space	/tmp Directory Space	HDD Space (Full / Client Only)
LINUX/x86	2GB	4GB	500MB	2.5GB / 400MB

▪ 가상머신 소프트웨어 설치 및 환경 구축

디렉토리

- 디렉토리란 파일 저장소를 의미하며, 리눅스 디렉토리는 최상위 디렉토리를 기준으로 하위 디렉토리들이 존재하는 계층적 트리 구조로 구성되어 있다.



▪ 가상머신 소프트웨어 설치 및 환경 구축

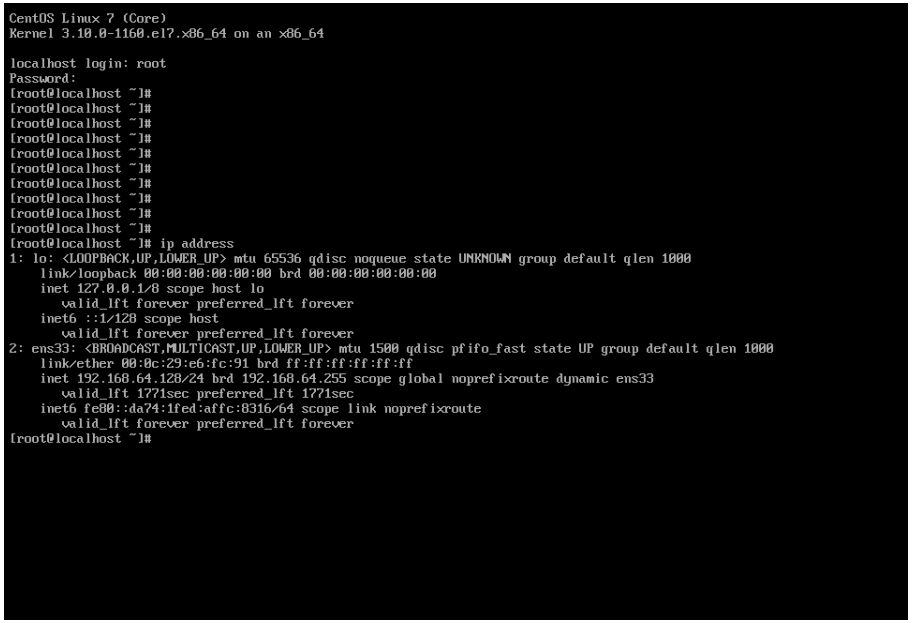
디렉토리 종류

- / : 최상위 디렉토리
- /bin : 기본적인 명령어 바이너리 파일이 저장된 디렉토리
- /boot : 리눅스 부트로더가 존재하는 디렉토리
- /dev : 시스템 디바이스 파일을 저장
- /etc : 시스템 환경 설정 파일과 부팅 관련 스크립트 파일들 저장
- /home : 사용자 계정들의 홈 디렉토리
- /lib : 공유 라이브러리 디렉토리
- /mnt : 파일 시스템을 일시적으로 마운트할 때 사용
- /proc : 시스템 정보 디렉토리이며 커널 기능을 제어
- /root : 시스템 관리자용 홈 디렉토리
- /sbin : 관리자용 시스템 표준 명령 및 시스템 관리와 관련된 실행 명령어
- /tmp : 각종 프로그램이나 프로세스 작업을 할 때 임시로 생성되는 파일 저장
- /usr : 일반 사용자 디렉토리로 사용자 데이터나 애플리케이션 저장
- /var : 가변 자료 저장 디렉토리로 로그 파일이나 메일 데이터 저장
- /lost+found : 결함이 있는 파일에 대한 정보가 저장

▪ 가상머신 소프트웨어 설치 및 환경 구축

CentOS 설치

- Xwindow를 설치하지 않기 위해서 최소 설치로 진행



설치 완료

Ⅲ. 리눅스 주요 명령어 및 OS 개요

- 리눅스 주요 명령어
- 파이프, 필터, 리디렉션
- 프로세스, 데몬, 서비스, 소켓
- 쉘 스크립트 개념, 실습
- 텔넷, ssh 개념

▪ 리눅스 주요 명령어

ls

디렉터리 파일 리스트 출력

[옵션]
- a : 숨김파일 포함
- l : 접근 허가권부터 파일 크기까지 상세한 정보
- 정규표현식에 해당하는 파일만 리스트 가능

```
[root@localhost bin]# ls -al vm*
-rwxr-xr-x. 1 root root  9802 10월  2  2020 vm-support
-rwxr-xr-x. 1 root root 54056 10월  2  2020 vmhgfs-fuse
-rwxr-xr-x. 1 root root 32232 10월  1  2020 vmstat
-rwxr-xr-x. 1 root root 57248 10월  2  2020 vmtoolsd
-rwxr-xr-x. 1 root root 11336 10월  2  2020 vmware-checkvm
-rwxr-xr-x. 1 root root 11488 10월  2  2020 vmware-hgfsclient
-rwxr-xr-x. 1 root root 19504 10월  2  2020 vmware-namespace-cmd
-rwxr-xr-x. 1 root root 19936 10월  2  2020 vmware-rpctool
-rwxr-xr-x. 1 root root 40664 10월  2  2020 vmware-toolbox-cmd
-rwxr-xr-x. 1 root root 15376 10월  2  2020 vmware-vgauth-cmd
-rwxr-xr-x. 1 root root 11344 10월  2  2020 vmware-xferlogs
```

▪ 리눅스 주요 명령어

rm

파일 또는 디렉터리 삭제

[옵션]

- r : recursive의 약자. 디렉토리와 하위 데이터들을 삭제
- f : 엑세스 권한이 없는 파일도 강제로 삭제

```
[root@localhost seminar]# vi rm_test.txt
[root@localhost seminar]# ls
hello.txt  rm_test.txt
[root@localhost seminar]# rm rm_test.txt
rm: remove 일반 파일 `rm_test.txt'? yes
[root@localhost seminar]# ls
hello.txt
[root@localhost seminar]# mkdir rm_test
[root@localhost seminar]# vi rm_test/rm_test_file.txt
[root@localhost seminar]# ls rm_test/rm_test_file.txt
rm_test/rm_test_file.txt
[root@localhost seminar]# ls rm_test/
rm_test_file.txt
[root@localhost seminar]# rm -rf rm_test/
[root@localhost seminar]# ls
hello.txt
```

▪ 리눅스 주요 명령어

rm

파일 또는 디렉터리 삭제

[옵션]
- r : recursive의 약자. 디렉토리와 하위 데이터들을 삭제
- f : 엑세스 권한이 없는 파일도 강제로 삭제

```
[root@localhost rm_test]# ls
rm_test1.txt  rm_test2.txt  rm_test3.txt
[root@localhost rm_test]# rm rm_test*
rm: remove 일반 빈 파일 `rm_test1.txt'? y
rm: remove 일반 빈 파일 `rm_test2.txt'? y
rm: remove 일반 빈 파일 `rm_test3.txt'? y
[root@localhost rm_test]# ls
[root@localhost rm_test]#
```

▪ 리눅스 주요 명령어

cp

파일 복사
복사 뿐만 아니라 파일 이름을 바꾸거나 디렉토리를 복사할 수 있다.

[옵션]
- r: 디렉토리 하위 데이터까지 복사
- a: 원본 파일의 속성, 링크 정보까지 복사
- f: 복사 위치에 파일이 있으면 덮어 씀

```
[root@localhost seminar]# mkdir cp_test
[root@localhost seminar]# ls
cp_test  hello.txt
[root@localhost seminar]# cp hello.txt ./cp_test/
[root@localhost seminar]# ls cp_test/
hello.txt
[root@localhost seminar]# cp hello.txt ./cp_test/new_hello.txt
[root@localhost seminar]# ls cp_test/
hello.txt  new_hello.txt
```

▪ 리눅스 주요 명령어

cp

파일 복사
복사 뿐만 아니라 파일 이름을 바꾸거나 디렉토리를 복사할 수 있다.

[옵션]
- r: 디렉토리 하위 데이터까지 복사
- a: 원본 파일의 속성, 링크 정보까지 복사
- f: 복사 위치에 파일이 있으면 덮어 씀

```
[root@localhost seminar]# head ./cp_test/hello.txt
hello world!!
```

```
[root@localhost seminar]# head hello_force.txt
hello force!!!!!
```

```
[root@localhost seminar]# cp -f hello
hello.txt        hello_force.txt
[root@localhost seminar]# cp -f hello
hello.txt        hello_force.txt
[root@localhost seminar]# cp -f hello_force.txt cp_test/hello.txt
cp: overwrite `cp_test/hello.txt'? y
[root@localhost seminar]# head cp_test/hello.txt
hello force!!!!!
```

▪ 리눅스 주요 명령어

mkdir

디렉토리 생성

[옵션]
- p : 부모 디렉터리까지 생성

```
[root@localhost ~]# mkdir mkdir_test/test1/test2/test3
mkdir: `mkdir_test/test1/test2/test3' 디렉토리를 만들 수 없습니다 : 그런 파일이나 디렉터리가 없습니다
[root@localhost ~]# mkdir -p mkdir_test/test1/test2/test3
[root@localhost ~]# ls
anaconda-ks.cfg  mkdir_test  tibero7  windowsShared
[root@localhost ~]# cd mkdir_test/test1/test2/test3/
[root@localhost test3]# ls
```

▪ 리눅스 주요 명령어

rmdir

디렉토리 삭제

[옵션]
- p : 부모 디렉터리까지 삭제

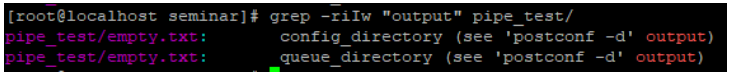
```
[root@localhost mkdir_test]# rmdir -p test1/test2/test3
[root@localhost mkdir_test]# ls
```

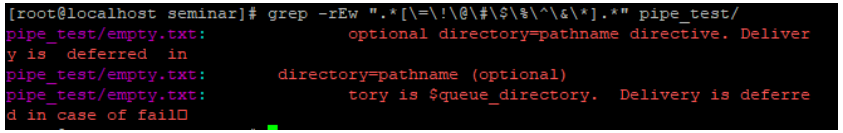
▪ 리눅스 주요 명령어

grep

파일에서 지정한 문자열을 찾는 명령어

[옵션]
- c : 일치하는 행의 번호를 출력
- i : 대소문자를 구분하지 않음
- l : 패턴이 포함된 파일의 이름 출력
- w : 단어와 일치하는 행만 출력
- x : 라인과 일치하는 행만 출력
- r : 하위 디렉토리의 모든 파일에서 검색
- E : 찾을 패턴을 정규 표현식으로 검색





▪ 리눅스 주요 명령어

useradd

새로운 사용자 추가.
이 명령을 실행하면 /etc/passwd, /etc/shadow, /etc/group 파일에 새로운 행이 추가된다.

[옵션]
- u : 사용자 추가하면서 uid값 등록
- p : 암호를 추가할 때 사용
- m : 홈 디렉터리가 없으면 추가
- s : 사용자가 사용할 셸 지정

```
[root@localhost seminar]# useradd moti -p 1234 -m -s /bin/bash
[root@localhost seminar]# ls /home
moti  tibero
```

▪ 리눅스 주요 명령어

usermod

사용자의 속성을 변경

[옵션]
- c : 사용자 comment 변경
- d : 홈 디렉터리 변경
- e : 계정 만료 날짜 설정
- g : 유저 그룹 변경

```
[root@localhost ~]# usermod -c "Hello Moti\!\!" moti
[root@localhost ~]# vi /etc/passwd
```

```
moti:x:1001:1001:Hello Moti\!\!:/home/moti:/bin/bash
~
~
"/etc/passwd" 21L, 948C
```

▪ 리눅스 주요 명령어

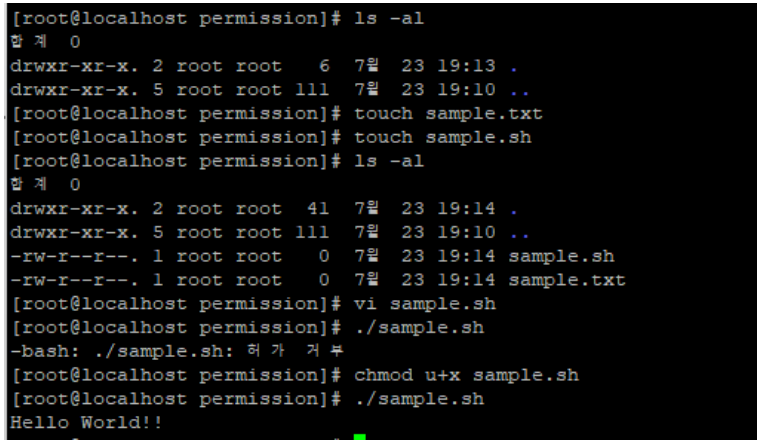
chmod

파일 허가권을 변경하는 명령어
ex) chmod 777 sample.txt

[상대 모드]
현재 허가권을 기준으로 허가권을 추가 또는 삭제하는 명령어.

ex) chmod u+x sample.txt

Owner			Group			Other		
r	w	x	r	w	x	r	w	x
4	2	1	4	2	1	4	2	1
7			7			7		



▪ 리눅스 주요 명령어

chown

파일 소유권을 변경하는 명령어
ex) chown moti sample.sh

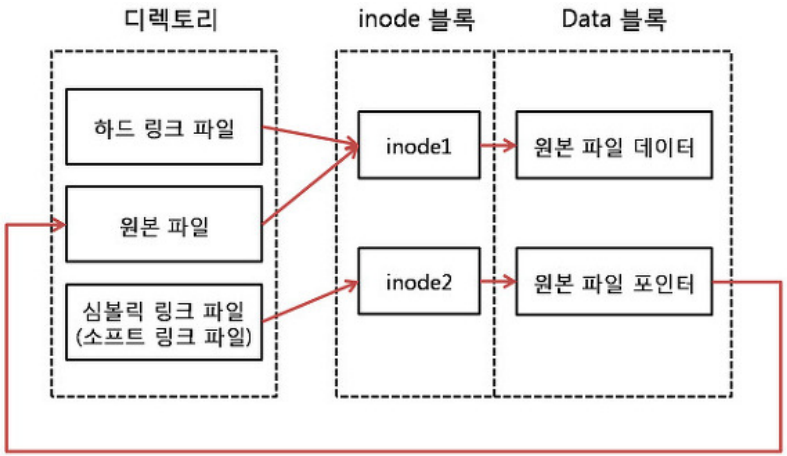
```
[root@localhost permission]# chown moti sample.sh
[tibero@localhost permission]$ su tibero
암 호 :
[tibero@localhost permission]$ ls -al
합 계  4
drwxr-xr-x. 2 root root  41  7월  23 19:14 .
drwxr-xr-x. 5 root root 111  7월  23 19:10 ..
-rwxr--r--. 1 moti root  34  7월  23 19:14 sample.sh
-rw-r--r--. 1 root root   0  7월  23 19:14 sample.txt
[tibero@localhost permission]$ ./sample.sh
bash: ./sample.sh: 허 가  거 부
```

▪ 리눅스 주요 명령어

링크

원본 파일을 가리키는 파일을 생성하는 것.

- 하드 링크 : 원본 파일의 inode를 공유
- 심볼릭 링크 : 생성한 파일의 inode가 가리키는 데이터가 원본 파일의 포인터



▪ 리눅스 주요 명령어

링크

원본 파일을 가리키는 파일을 생성하는 것.

- 하드 링크 : 원본 파일의 inode를 공유
- 심볼릭 링크 : 생성한 파일의 inode가 가리키는 데이터가 원본 파일의 포인터

```
[root@localhost link_test]# touch original.txt
[root@localhost link_test]# ln original.txt hardlink.txt
[root@localhost link_test]# ln -s original.txt softlink.txt
[root@localhost link_test]# ls -ali
합계 0
100665091 drwxr-xr-x. 2 root root  66  7월  23 19:56 .
   502217 drwxr-xr-x. 6 root root 128  7월  23 19:54 ..
100665128 -rw-r--r--. 2 root root   0  7월  23 19:56 hardlink.txt
100665128 -rw-r--r--. 2 root root   0  7월  23 19:56 original.txt
100665129 lrwxrwxrwx. 1 root root  12  7월  23 19:56 softlink.txt -> original.txt
```

```
[root@localhost link_test]# vi original.txt
[root@localhost link_test]# cat original.txt
modified
[root@localhost link_test]# cat hardlink.txt
modified
[root@localhost link_test]# cat softlink.txt
modified
[root@localhost link_test]# vi softlink.txt
[root@localhost link_test]# cat softlink.txt
softlink modified
[root@localhost link_test]# cat hardlink.txt
softlink modified
[root@localhost link_test]# cat
hardlink.txt  original.txt  softlink.txt
[root@localhost link_test]# cat original.txt
softlink modified
```

▪ 리눅스 주요 명령어

링크

원본 파일을 가리키는 파일을 생성하는 것.

- 하드 링크 : 원본 파일의 inode를 공유
- 심볼릭 링크 : 생성한 파일의 inode가 가리키는 데이터가 원본 파일의 포인터

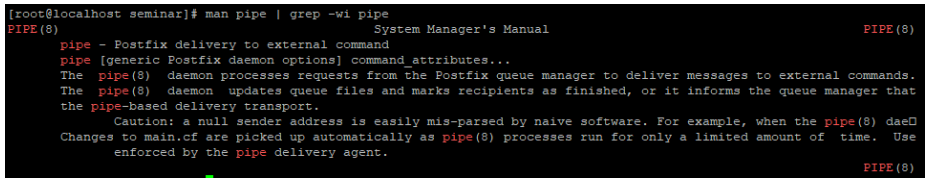
```
[root@localhost link_test]# rm original.txt
rm: remove 일반 파일 `original.txt'? y
[root@localhost link_test]# ls -al
합계 4
drwxr-xr-x. 2 root root  46  7월  23 20:03 .
drwxr-xr-x. 6 root root 128  7월  23 19:54 ..
-rw-r--r--. 1 root root  18  7월  23 20:01 hardlink.txt
lrwxrwxrwx. 1 root root  12  7월  23 19:56 softlink.txt -> original.txt
```

▪ 파이프, 필터, 리디렉션

파이프

파이프란 2개의 프로그램을 연결하는 연결 통로의 의미이다

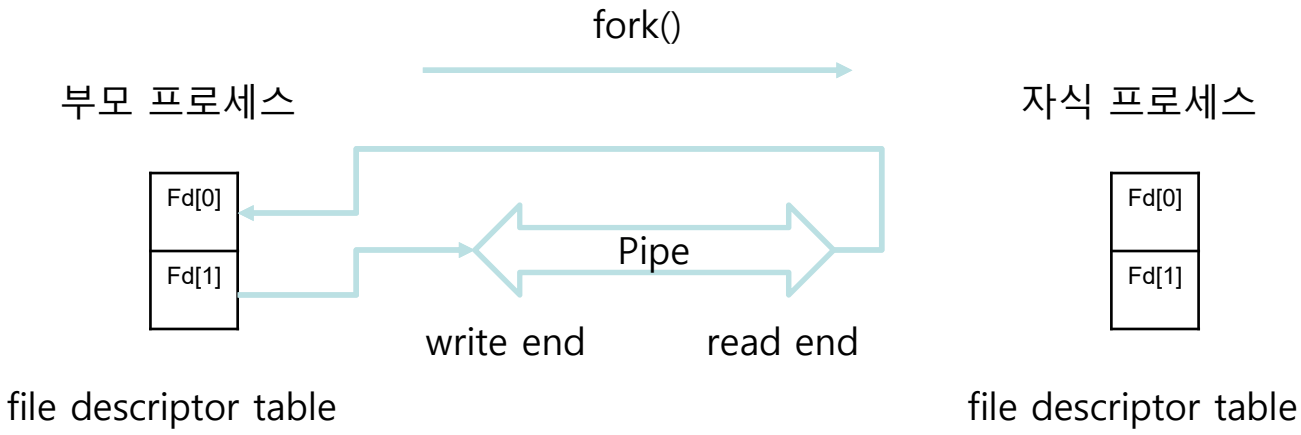
- 터미널 프로세스에서 자식 프로세스를 생성(PID는 같음)하여 부모 프로세스의 결과를 자식 프로세스에서 실행시킬 수 있도록 하는 것.



▪ 파이프, 필터, 리디렉션

파이프 프로세스

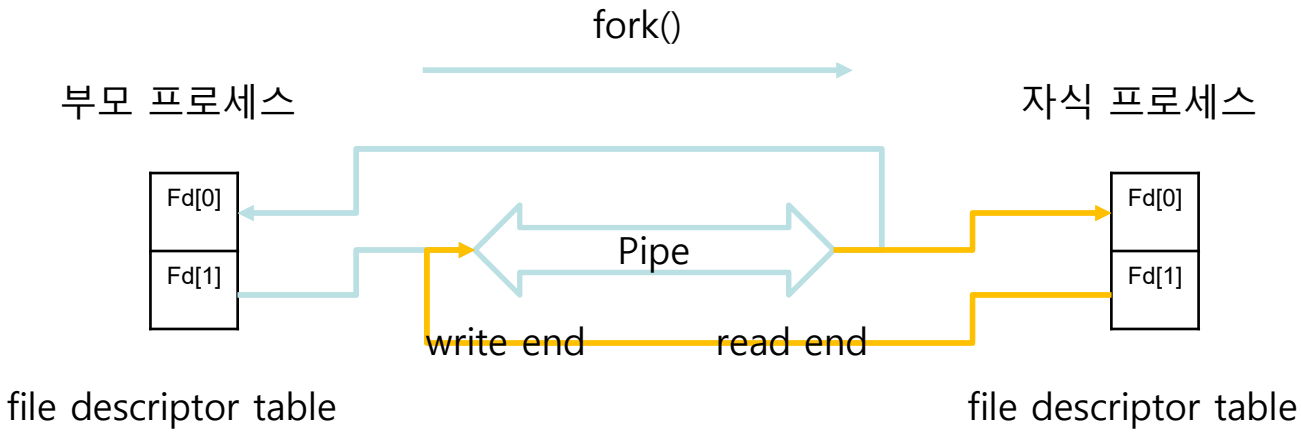
Command : cat file.txt | grep text



▪ 파이프, 필터, 리디렉션

파이프 프로세스

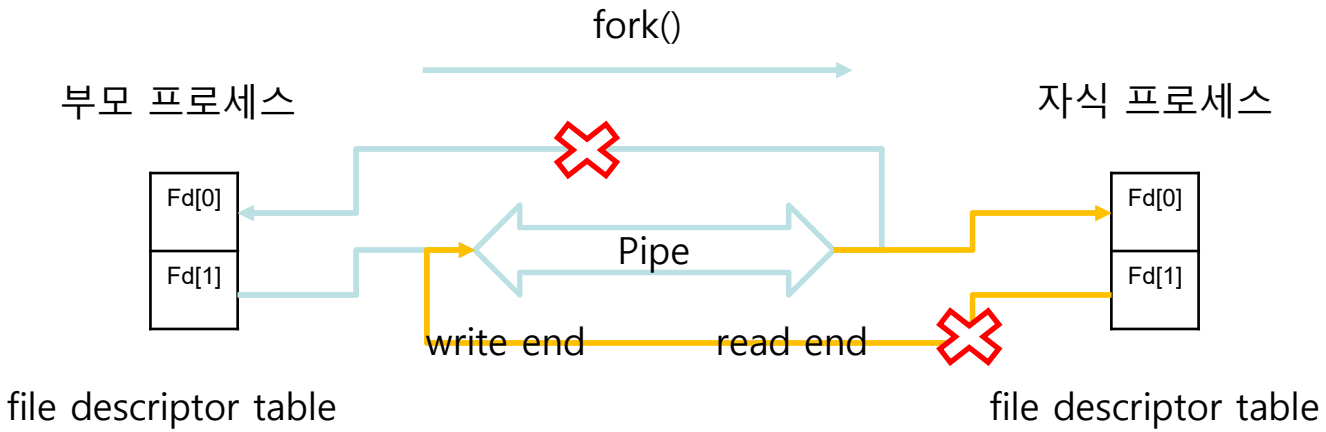
Command : cat file.txt | grep text



▪ 파이프, 필터, 리디렉션

파이프 프로세스

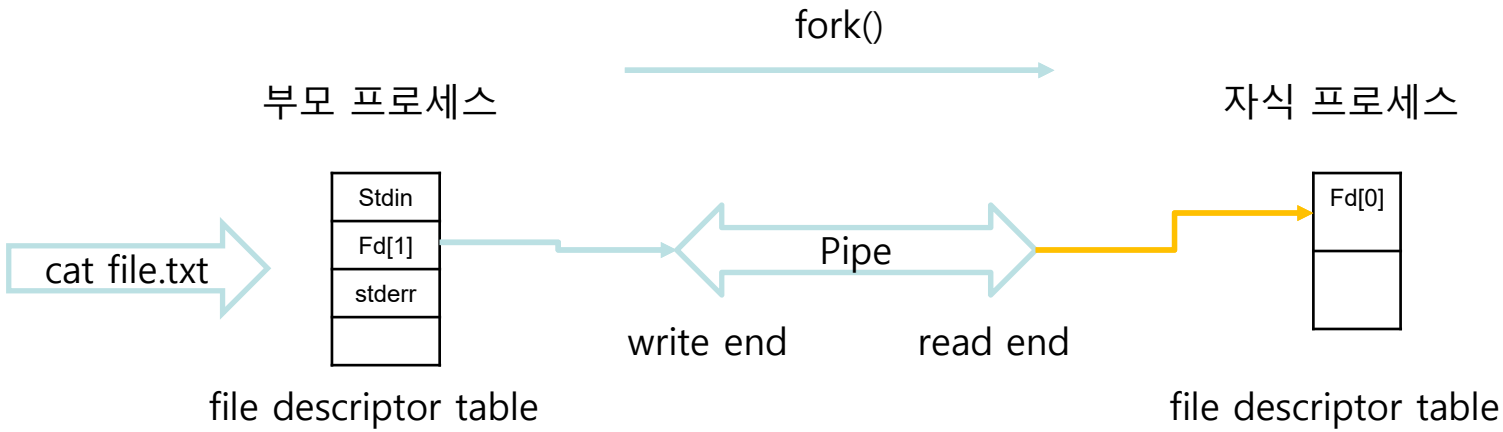
Command : cat file.txt | grep text



▪ 파이프, 필터, 리디렉션

파이프 프로세스

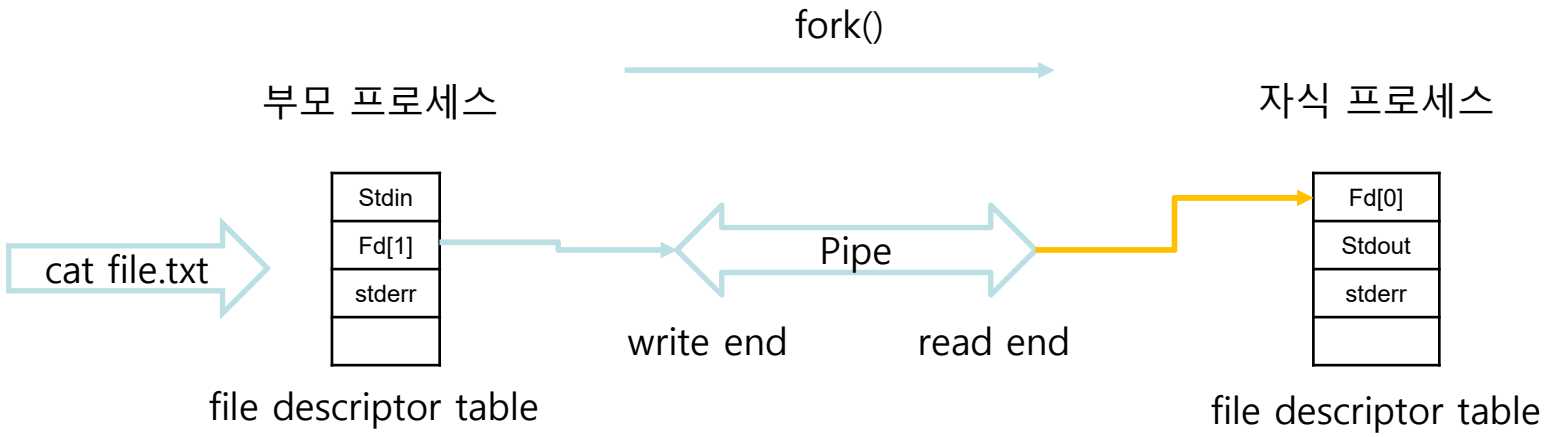
Command : cat file.txt | grep text



▪ 파이프, 필터, 리디렉션

파이프 프로세스

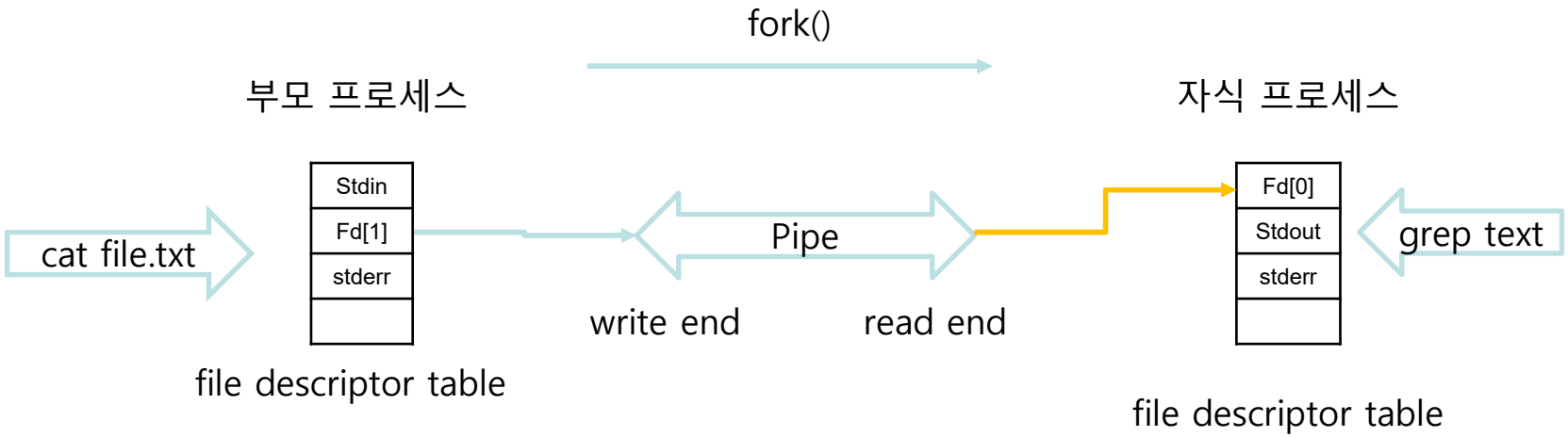
Command : cat file.txt | grep text



▪ 파이프, 필터, 리디렉션

파이프 프로세스

Command : cat file.txt | grep text



▪ 파이프, 필터, 리디렉션

필터

필터란 필요한 것만 걸러주는 명령어다.
grep, tail, wc, sort, awk, sed 명령어 등이 있고,
주로 파이프와 같이 사용된다.

```
[root@localhost seminar]# man pipe | grep pipe
      pipe - Postfix delivery to external command
      pipe [generic Postfix daemon options] command_attributes...
      The  pipe(8)  daemon processes requests from the Postfix queue manager to deliver messages to external commands.
      The  pipe(8)  daemon  updates queue files and marks recipients as finished, or it informs the queue manager that
      the pipe-based delivery transport.
             Caution: a null sender address is easily mis-parsed by naive software. For example, when the pipe(8) dae□
      Changes to main.cf are picked up automatically as pipe(8) processes run for only a limited amount of  time.  Use
             enforced by the pipe delivery agent.
```

```
[root@localhost seminar]# man pipe | wc
    373    2209   18673
```

```
[root@localhost seminar]# man pipe | grep pipe | sed 's/pipe/PIPE/g' | grep PIPE
      PIPE - Postfix delivery to external command
      PIPE [generic Postfix daemon options] command_attributes...
      The  PIPE(8)  daemon processes requests from the Postfix queue manager to deliver messages to external commands.
      The  PIPE(8)  daemon  updates queue files and marks recipients as finished, or it informs the queue manager that
      the PIPE-based delivery transport.
             Caution: a null sender address is easily mis-parsed by naive software. For example, when the PIPE(8) dae□
      Changes to main.cf are picked up automatically as PIPE(8) processes run for only a limited amount of  time.  Use
             enforced by the PIPE delivery agent.
```

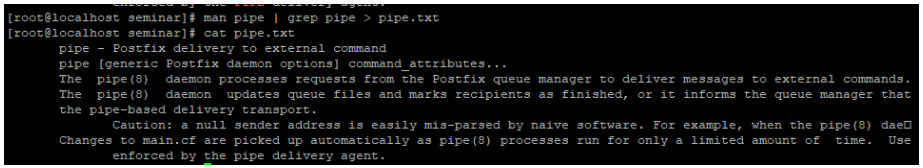
▪ 파이프, 필터, 리디렉션

리디렉션

표준 입출력의 방향을 바꿔준다.
표준 입력은 키보드, 표준 출력은 모니터지만 이를 파일로 처리하고 싶을 때 주로 사용한다.

>, >> : 표준 출력 방향 변경

<, << : 표준 입력 방향 변경



▪ 프로세스, 데몬, 서비스, 소켓

프로세스와 데몬

프로세스는 프로그램이 활성화 된 것이라고 할 수 있다.
그 종류로는 백그라운드 프로세스와 포그라운드 프로세스가 있다.

데몬이라고 부르는 서비스는 서버 프로세스를 말한다.
즉 데몬은 웹 서버, 네임 서버, DB 서버 등의 프로세스를 지칭한다.

프로세스와 데몬의 차이는 프로세스는 필요 시 실행되고 종료된다. 하지만
데몬은 항상 실행되며 클라이언트 요청에 응답을 보낸다.

▪ 프로세스, 데몬, 서비스, 소켓

프로세스 관련 명령어

ps : 현재 프로세스의 상태를 확인하는 명령어

```
[root@localhost seminar]# ps -ef | grep terminal
root       15777   14804   0 23:36 pts/1    00:00:00 grep --color=auto terminal
```

kill : 프로세스를 강제로 종료하는 명령어

▪ 프로세스, 데몬, 서비스, 소켓

서비스와 소켓

서비스는 평상시에도 늘 가동하는 서버 프로세스며, 소켓은 필요할 때만 작동하는 서버 프로세스다.

▪ 프로세스, 데몬, 서비스, 소켓

서비스

- 시스템과 독자적으로 구동 및 제공되는 프로세스. 웹 서버, DB 서버 등을 예로 들 수 있다.
- 실행 및 종료는 대개 systemctl start/stop/restart 명령으로 사용된다.
- 서비스의 실행 스크립트 파일은 /usr/lib/system/system/ 디렉터리에 '서비스이름.service'라는 이름으로 확인할 수 있다.

```
[root@localhost seminar]# ls /usr/lib/systemd/system/ | head
-.slice
NetworkManager-dispatcher.service
NetworkManager-wait-online.service
NetworkManager.service
auditd.service
autovt@.service
basic.target
basic.target.wants
blk-availability.service
bluetooth.target
```

▪ 프로세스, 데몬, 서비스, 소켓

소켓

- 서비스는 항상 가동되지만 소켓은 외부에서 특정 서비스를 요청할 경우 systemd가 구동시킨다.

- 소켓과 관련된 스크립트 파일은 /usr/lib/systemd/system/ 디렉터리에 '소켓이름.socket'이라는 이름으로 존재한다

```
[root@localhost system]# ls -al | grep socket
-rw-r--r--.  1 root root   102 10월   1  2020 dbus.socket
-r--r--r--.  1 root root   248  4월  28  2021 dm-event.socket
-r--r--r--.  1 root root   241  4월  28  2021 lvm2-lvmetad.socket
-r--r--r--.  1 root root   239  4월  28  2021 lvm2-lvmpolld.socket
-rw-r--r--.  1 root root   138 12월  16  2022 rsyncd.socket
-rw-r--r--.  1 root root   356  9월   1  2022 sockets.target
drwxr-xr-x.  2 root root   189  7월  21 17:13 sockets.target.wants
-rw-r--r--.  1 root root   181 11월  25  2021 sshd.socket
-rw-r--r--.  1 root root  1235  9월   1  2022 syslog.socket
-rw-r--r--.  1 root root   524  9월   1  2022 systemd-initctl.socket
-rw-r--r--.  1 root root   833  9월   1  2022 systemd-journald.socket
-rw-r--r--.  1 root root   528  9월   1  2022 systemd-shutdownd.socket
-rw-r--r--.  1 root root   595  9월   1  2022 systemd-udevd-control.socket
-rw-r--r--.  1 root root   570  9월   1  2022 systemd-udevd-kernel.socket
-rw-r--r--.  1 root root   153 11월  16  2020 telnet.socket
```

▪ 셸 스크립트 개념, 실습

셸 스크립트

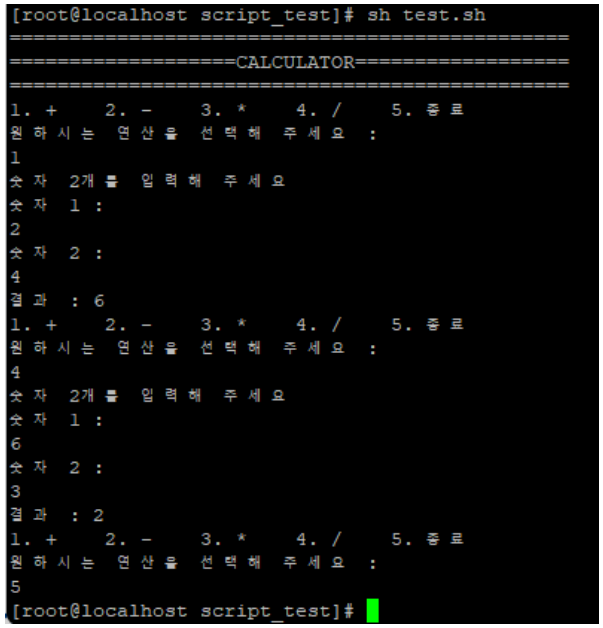
- 셸 스크립트는 C 언어와 유사한 방법으로 프로그래밍할 수 있는 스크립트 언어다.
- 일반적인 프로그래밍 언어와 비슷하게 변수, 반복문, 제어문 등을 사용할 수 있다.
- 셸 명령어를 나열해서 실행되는 방식.

▪ 셸 스크립트 개념, 실습

셸 스크립트

계산기 만들기

- 두 수를 입력 받아 사칙연산 (+, -, *, /)을 수행하는 계산기 만들기
- 종료 메뉴를 선택하거나 메뉴에 없는 값을 입력하면 종료



▪ 셸 스크립트 개념, 실습

셸 스크립트

```
#!/bin/bash

echo "================================================"
echo "==================CALCULATOR===================="
echo "================================================"


calc_option=0

num1=0
num2=0
result=0
getTwoNumbers(){
        echo "숫 자  2개 를  입 력 해  주 세 요 "
        echo "숫 자  1 : "
        read num1
        echo "숫 자  2 : "
        read num2
}
```

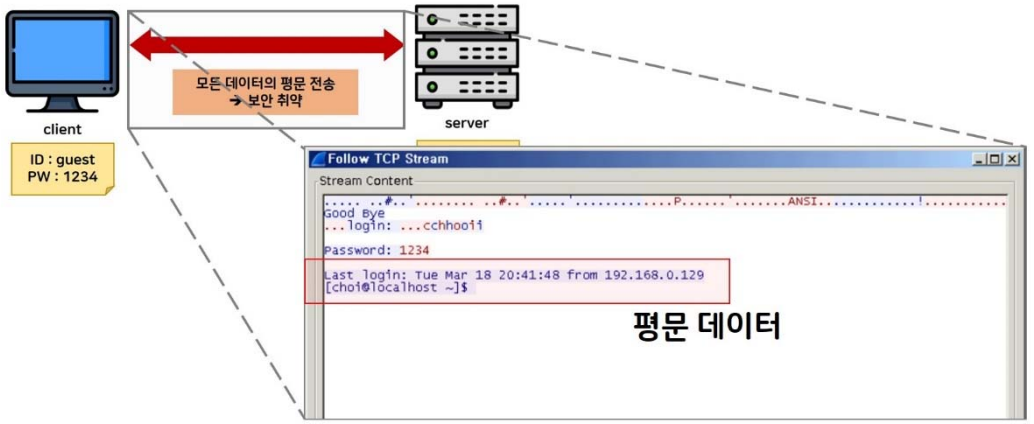
```
while [ 1 ]
do
        echo "1. +      2. -     3. *     4. /     5. 종 료 "
        echo "원 하 시 는  연 산 을  선 택 해  주 세 요  : "
        read calc_option

        if [ "$calc_option" = "1" ]
        then
                getTwoNumbers
                result=`expr $num1 + $num2`
                echo "결 과  : $result"
        elif [ "$calc_option" = "2" ]
        then
                getTwoNumbers
                result=`expr $num1 - $num2`
                echo "결 과  : $result"
        elif [ "$calc_option" = "3" ]
        then
                getTwoNumbers
                result=`expr $num1 \* $num2`
                echo "결 과  : $result"
        elif [ "$calc_option" = "4" ]
        then
                getTwoNumbers
                result=`expr $num1 / $num2`
                echo "결 과  : $result"
        elif [ "$calc_option" = "5" ]
        then
                break
        else
                echo "정 해 진  숫 자  범 위  내 에 서  입 력 해  주 세 요  (1~4)"
                continue
        fi
done
exit 0
```

▪ 텔넷, ssh 개념

텔넷

텔넷은 전통적으로 사용되어 온 원격 접속 방법이다.

- 텔넷에서는 보통 데이터를 암호화하지 않고 평문으로 송수신하기 때문에 패킷 캡처 등에 의해 통신 내용을 도난당할 위험이 있다



▪ 텔넷, ssh 개념

텔넷 접속

방화벽에서 텔넷을 허용해야 다른 컴퓨터에서 텔넷을 통해 접속할 수 있다.

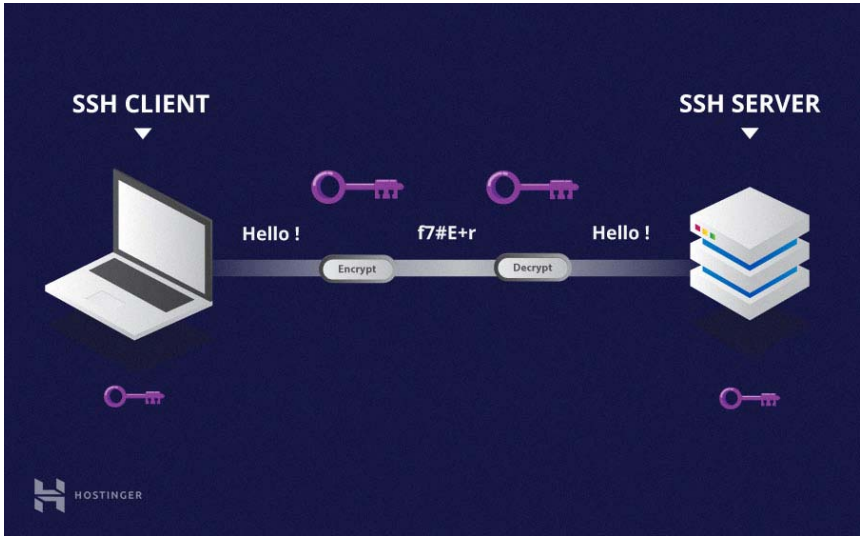
1. 윈도우에서 텔넷 기능 켜기
2. 리눅스에 텔넷 설치 후, 방화벽에 텔넷 추가하기

```
Last login: Mon Jul 24 08:42:27 2023
[root@localhost ~]# firewall-config
-bash: firewall-config: command not found
[root@localhost ~]# firewall-cmd --add-service=telnet
success
[root@localhost ~]# systemctl enable telnet.socket
Created symlink from /etc/systemd/system/sockets.target.wants/telnet.socket to /usr/lib/systemd/
system/telnet.socket.
[root@localhost ~]#
```

▪ 텔넷, ssh 개념

ssh

텔넷과 기능은 비슷하지만, 데이터를 암호화하기 때문에 보안이 강화되었다.

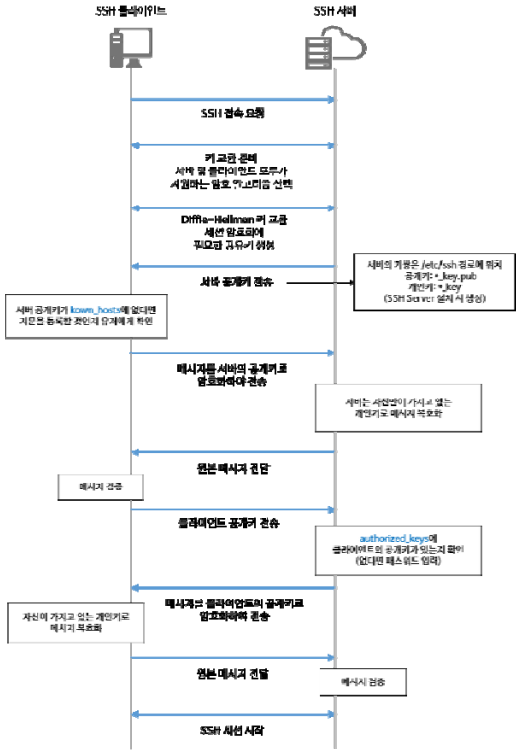


▪ 텔넷, ssh 개념

Ssh의 암호화

비대칭 키 암호화를 통해 보안성을 높였다.

- 서버와 클라이언트가 각자의 공개키를 공유하고, 상대방의 공개 키로 암호화를 하면 각자 가지고 있는 개인 키로 복호화 하는 방식



▪ 텔넷, ssh 개념

Ssh 접속

Ssh 서비스 상태 점검 후 클라이언트에서 ssh 연결.

첫 연결일 경우, 서버의 공개 키가 저장되어 있지 않기 때문에 바로 연결되지 않는다.

```
[root@localhost ~]# systemctl status sshd
● sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled)
   Active: active (running) since 2023-07-24 08:42:18 KST; 2h 16min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
 Main PID: 1073 (sshd)
   CGroup: /system.slice/sshd.service
           └─1073 /usr/sbin/sshd -D

 724 08:42:18 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...
 724 08:42:18 localhost.localdomain sshd[1073]: Server listening on 0.0.0.0 port 22.
 724 08:42:18 localhost.localdomain sshd[1073]: Server listening on :: port 22.
 724 08:42:18 localhost.localdomain systemd[1]: Started OpenSSH server daemon.
 724 08:53:55 localhost.localdomain sshd[1712]: Accepted password for root from 192.168...h2
Hint: Some lines were ellipsized, use -l to show in full.
```

```
C:₩Users₩이윤재>ssh telnet@192.168.80.128
The authenticity of host '192.168.80.128 (192.168.80.128)' can't be established.
ECDSA key fingerprint is SHA256:gTfBZQo8AricRmvNrobNnltsEoFCp+5JROB6KMsi9MM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? _
```

```
C:₩Users₩이윤재>ssh telnet@192.168.80.128
telnet@192.168.80.128's password:
Last login: Mon Jul 24 11:02:06 2023 from 192.168.80.1
[telnet@localhost ~]$ whoami
telnet
[telnet@localhost ~]$ ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0c:29:0b:c5:4c brd ff:ff:ff:ff:ff:ff
    inet 192.168.80.128/24 brd 192.168.80.255 scope global noprefixroute dynamic ens33
       valid_lft 1117sec preferred_lft 1117sec
    inet6 fe80::6cd6:4627:aed2:f646/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

IV. Database 서버 구축과 운영

- Tibero 설치 및 운영

▪ Tibero 설치 및 운영

티베로 설치 준비

티베로 설치 전, 하드웨어 요구사항과 소프트웨어 요구사항을 만족할 수 있도록 준비한다.

Platform	RAM	Swap Space	/tmp Directory Space	HDD Space (Full / Client Only)
LINUX/x86	2GB	4GB	500MB	2.5GB / 400MB

Platform	OS	Compiler	JDK Version
LINUX/x86	Red Hat Enterprise Linux 7 kernel 3.10.0 이상	C99 지원 컴파일러, gcc version 4.8.5 이상	JDK 1.5.17 이상

Platform	Packages
Linux	gcc-3.4.6-11 gcc-c++-3.4.6-11 libgcc-3.4.6-11 libstdc++-3.4.6-11 libstdc++-devel-3.4.6-11 libaio-0.3.105-2 libaio-devel-0.3.105-2 pstack glibc

▪ Tibero 설치 및 운영

티베로 설치 준비(커널 파라미터)

리눅스 시스템의 커널에서 정의된 자원 설정값. 커널 파라미터 값을 변경하면, 프로세스가 런타임 중에도 즉시 반영되어 시스템을 최적화, 튜닝이 가능.

- 커널 파라미터
  - 설정 파일

    ```
    /etc/sysctl.conf
    ```

  - 설정값

커널 파라미터	값
kernel.sem	순서대로 SEMMSL, SEMMNS, SEMOPM, SEMMNI 최소 설정값은 아래와 같다. ◦ SEMMSL : 2 / 권장값 : (Tibero 전체 Thread 수) x 2 ◦ SEMMNS : (Tibero 전체 Thread 수) x 2 ◦ SEMOPM : 2(=SEMMSL) / 권장값 : (Tibero 전체 Thread 수) x 2 ◦ SEMMNI : (Tibero 전체 Thread 수) max값을 높게 설정해도 큰 문제가 없기에 여유롭게 설정하는 것을 권장한다.
kernel.shmall	ceil(shmmax/PAGE_SIZE)
kernel.shmmax	물리적인 메모리의 절반 (byte)
kernel.shmmni	4096
fs.nr_open	nofile 파라미터 이상으로 설정한다.
fs.file-max	(nofile 파라미터) x (WTHR_PROC_CNT + PEP_PROC_CNT) 또는 67108864 으로 설정한다.
fs.aio-max-nr	1048576
net.ipv4.ip_local_port_range	1024 65000
net.core.rmem_default	262144
net.core.wmem_default	262144
net.core.rmem_max	67108864
net.core.wmem_max	67108864

▪ Tibero 설치 및 운영

티베로 설치 준비(커널 파라미터)

파라미터	설명
kernel.sem	커널 세마포어 설정
kernel.shmall	특정 시점에서 시스템에 사용 가능한 공유 메모리의 최대 크기 설정(페이지 단위)
kernel.shmmax	공유 메모리 세그먼트의 최대 크기를 정의함
kernel.shmmni	공유 메모리 세그먼트의 최대 개수를 설정하는데 사용됨
fs.nr_open	프로세스당 file open 개수 설정
fs.file-max	리눅스 커널이 동시에 열 수 있는 file handle의 개수
fs.aio-max-nr	시스템이 서버에서 처리할 수 있는 최대 비동기 I/O 작업 수를 정의한다.
net.ipv4.ip_local_port_range	사용할 수 있는 포트 범위를 정의한다.
net.core.rmem_default	TCP의 기본 수신 소켓 메모리 정의
net.core.wmem_default	TCP의 기본 전송 소켓 메모리 정의
net.core.rmem_max	TCP의 최대 수신 소켓 메모리 정의
net.core.wmem_max	TCP의 최대 전송 소켓 메모리 정의

SEMMSL	배열당 최대 세마포어 수
SEMMNS	시스템 전체 최대 세마포어 수
SEMOPM	세마포어 호출당 최대 operation 수
SEMMNI	최대 배열 수

▪ Tibero 설치 및 운영

티베로 설치 준비

티베로 설치 전, 하드웨어 요구사항과 소프트웨어 요구사항을 만족할 수 있도록 준비한다.

Shell Limits 파라미터

- 설정 파일

```
/etc/security/limits.conf
```

- 설정값

파라미터	설명
nofile	(WTHR_PER_PROC * ((total data files in db) + 15)) + (tbsvr process count + 5) + 100 또는 크게 3,000,000 으로 설정한다.
nproc	MAX_SESSION_COUNT+10000 이상으로 설정한다. ◦ Soft Limit : 65536 ◦ Hard Limit : 65536

[참고]

RHEL 7.2 이상 버전부터는 아래 커널 파라미터 설정이 필요하다.

- 설정 파일

```
/etc/systemd/logind.conf
```

- 설정값

커널 파라미터	값
RemoveIPC	No

▪ Tibero 설치 및 운영

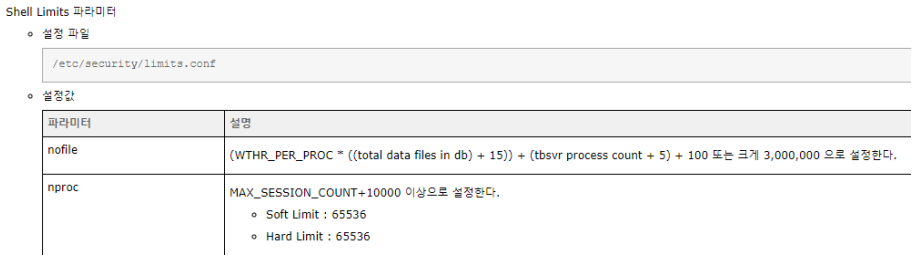
티베로 설치 준비

[Shell Limits 파라미터]

- nofile : 사용자, 그룹이 오픈할 수 있는 최대 파일 개수
- nproc : 사용자, 그룹의 최대 프로세스 개수

RemoveIPC가 yes일 경우, 사용자가 로그아웃을 하였을 때, IPC 자원을 모두 제거한다.

이 기능을 yes로 했을 경우, DB 시스템이 정상 작동하지 않을 가능성이 높아 no로 해주어야한다.



Shell Limits 파라미터

- 설정 파일

```
/etc/security/limits.conf
```

- 설정값

파라미터	설명
nofile	(WTHR_PER_PROC * ((total data files in db) + 15)) + (tbsvr process count + 5) + 100 또는 크게 3,000,000 으로 설정한다.
nproc	MAX_SESSION_COUNT+10000 이상으로 설정한다. ◦ Soft Limit : 65536 ◦ Hard Limit : 65536

[참고]

RHEL 7.2 이상 버전부터는 아래 커널 파라미터 설정이 필요하다.

- 설정 파일

```
/etc/systemd/logind.conf
```

- 설정값

커널 파라미터	값
RemoveIPC	No

▪ Tibero 설치 및 운영

티베로 설치

1. 바이너리 실행 파일과 라이선스 파일을 준비
2. 환경 변수 설정

```
export TB_HOME=/home/tibero/Tibero/tibero7
export TB_SID=tibero
export LD_LIBRARY_PATH=$TB_HOME/lib:$TB_HOME/client/lib
export PATH=$PATH:$TB_HOME/bin:$TB_HOME/client/bin
```

- $TB_HOME : 티베로가 설치된 홈 디렉터리
- $TB_SID : 한 머신에서 Tibero의 인스턴스를 여러 개로 운영할 때 필요한 서비스 ID이다.
- $LD_LIBRARY_PATH : 티베로를 사용할 때 필요한 공유 라이브러리가 위치한 경로이다.
- $PATH : 파일 시스템을 통해 특정 파일에 접근하기 위한 디렉터리 경로를 설정

▪ Tibero 설치 및 운영

티베로 설치

3. /home/tibero/Tibero 디렉터리에서 압축 해제 후 라이선스 파일을 $TB_HOME/license 디렉터리에 복사
4. $TB_HOME/config 디렉터리에서 gen_tip.sh 스크립트 파일을 실행한다.

- 환경 파일(.tip)과 tdbsn.tbr 파일 생성

```
[tibero@localhost tibero7]$ ./config/gen_tip.sh
Using TB_SID "tibero"
There's already /home/tibero/Tibero/tibero7/config/tibero.tip!!  Nothing has changed!!
Already exists /home/tibero/Tibero/tibero7/config/psm_commands!! Nothing has changed
There's already /home/tibero/Tibero/tibero7/client/config/tbdsn.tbr!!
Client config file already contains SID "tibero":
    please check if port number equals 8629...
> #------------------------------------------------
> # /home/tibero/Tibero/tibero7/client/config/tbdsn.tbr
> # Network Configuration File.
> # Generated by gen_tip.sh at Fri Jul 21 18:02:28 KST 2023
> tibero=(
>     (INSTANCE=(HOST=localhost)
>               (PORT=8629)
>               (DB_NAME=tibero)
>     )
> )
Running client/config/gen_esql_cfg.sh
Done.
```

▪ Tibero 설치 및 운영

티베로 설치

[tibero.tip]
티베로 환경 설정 파일

- DB_NAME : 데이터베이스 이름
- LISTENER_PORT : DB 연결 포트
- CONTROL_FILES : 컨트롤 파일 위치
- MAX_SESSION_COUNT : 최대 접속 세션 수
- TOTAL_SHM_SIZE : 공유 메모리 사이즈
- MEMORY_TARGET : 메모리 총량

▪ Tibero 설치 및 운영

티베로 설치

[tbdsn.tbr]
클라이언트가 티베로의 데이터베이스에 접속하기 위해 필요한 정보를 가지고 있는 환경설정 파일이다

- HOST : 서버의 IP 주소
- PORT : 서버의 포트 번호
- DB_NAME : 데이터베이스의 이름
- SID : 클라이언트에서 서버를 식별하기 위한 고유한 이름

```
#------------------------------------------------
# /home/tibero/Tibero/tibero7/client/config/tbdsn.tbr
# Network Configuration File.
# Generated by gen_tip.sh at Fri Jul 21 18:02:28 KST 2023
tibero=(
    (INSTANCE=(HOST=localhost)
              (PORT=8629)
              (DB_NAME=tibero)
    )
)
```

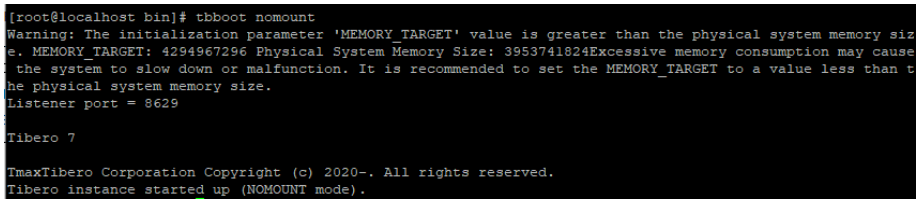
▪ Tibero 설치 및 운영

티베로 설치

5. Tibero 서버를 NOMOUNT 모드로 기동한다

시작 단계 : nomount -> mount -> open
종료 단계 : open -> mount -> nomount

nomount : 인스턴스 생성단계, 파라미터 파일 읽기
mount : 컨트롤 파일 읽기, 데이터베이스 복구
open : 데이터 파일과 로그 파일을 읽어 일관성을 확인한 후 해당 파일들을 열어 실제 데이터베이스를 사용할 수 있도록 함.



▪ Tibero 설치 및 운영

티베로 설치

6. tbSQL 유틸리티를 이용하여 데이터베이스에 접속한다.
7. CREATE DATABASE문을 이용하여 원하는 데이터베이스를 생성한다.

```
SQL> create database "tibero"
  user sys identified by tibero
  maxinstances 8
  maxdatafiles 100
  character set MSWIN949
  national character set UTF16
  logfile
    group 1 'log001.log' size 50M,
    group 2 'log002.log' size 50M,
    group 3 'log003.log' size 50M
  maxloggroups 255
  maxlogmembers 8
  noarchivelog
    datafile 'system001.dtf' size 100M autoextend on next 10M maxsize unlimited
    default temporary tablespace TEMP
      tempfile 'temp001.dtf' size 100M autoextend on next 10M maxsize unlimited
      extent management local autoallocate
    undo tablespace UNDO
      datafile 'undo001.dtf' size 200M autoextend on next 10M maxsize unlimited
      extent management local autoallocate
    SYSSUB
      datafile 'syssub001.dtf' size 10M autoextend on next 10M maxsize unlimited
    default tablespace USR
      datafile 'usr001.dtf' size 100M autoextend on next 10M maxsize unlimited
      extent management local autoallocate;
Database created.
```

▪ Tibero 설치 및 운영

티베로 설치

[create database]

- maxinstances : 데이터베이스를 동시에 마운트하고 열 수 있는 인스턴스의 최대 개수.
- national character set : 유니코드를 지원하지 않는 Character set을 가진 데이터베이스에서 유니코드를 지원하기 위해 부가적으로 설정할 수 있는 character set이다.
- maxlogmembers : 로그 그룹 내의 로그 파일의 최대 개수를 제한한다.

```
SQL> create database "tibero"
  user sys identified by tibero
  maxinstances 8
  maxdatafiles 100
  character set MSWIN949
  national character set UTF16
  logfile
    group 1 'log001.log' size 50M,
    group 2 'log002.log' size 50M,
    group 3 'log003.log' size 50M
  maxloggroups 255
  maxlogmembers 8
  noarchivelog
    datafile 'system001.dtf' size 100M autoextend on next 10M maxsize unlimited
    default temporary tablespace TEMP
      tempfile 'temp001.dtf' size 100M autoextend on next 10M maxsize unlimited
      extent management local autoallocate
    undo tablespace UNDO
      datafile 'undo001.dtf' size 200M autoextend on next 10M maxsize unlimited
      extent management local autoallocate
    SYSSUB
      datafile 'syssub001.dtf' size 10M autoextend on next 10M maxsize unlimited
    default tablespace USR
      datafile 'usr001.dtf' size 100M autoextend on next 10M maxsize unlimited
      extent management local autoallocate;
Database created.
```

▪ Tibero 설치 및 운영

티베로 설치

[create database]

- ARCHIVELOG MODE : redo log의 데이터 복구를 위한 백업 모드
- NOARCHIVELOG MODE : 기존 방식대로 redo log 사용

```
SQL> create database "tibero"
  user sys identified by tibero
  maxinstances 8
  maxdatafiles 100
  character set MSWIN949
  national character set UTF16
  logfile
    group 1 'log001.log' size 50M,
    group 2 'log002.log' size 50M,
    group 3 'log003.log' size 50M
  maxloggroups 255
  maxlogmembers 8
  noarchivelog
    datafile 'system001.dtf' size 100M autoextend on next 10M maxsize unlimited
    default temporary tablespace TEMP
      tempfile 'temp001.dtf' size 100M autoextend on next 10M maxsize unlimited
      extent management local autoallocate
    undo tablespace UNDO
      datafile 'undo001.dtf' size 200M autoextend on next 10M maxsize unlimited
      extent management local autoallocate
    SYSSUB
      datafile 'syssub001.dtf' size 10M autoextend on next 10M maxsize unlimited
    default tablespace USR
      datafile 'usr001.dtf' size 100M autoextend on next 10M maxsize unlimited
      extent management local autoallocate;
Database created.
```

▪ Tibero 설치 및 운영

티베로 설치

8. tbboot 명령어로 Tibero를 다시 기동한다
9. $TB_HOME/scripts 디렉터리에서 system.sh 셸을 실행한다.
10. 티베로 설치 완료

```
tibero@Tibero:~/Tibero/tibero7/scripts$ system.sh
Enter SYS password:

Enter SYSCAT password:

Creating the role DBA...
create default system users & roles?(Y/N):

Creating system users & roles...
Creating virtual tables(1)...
Creating virtual tables(2)...
Granting public access to _VT_DUAL...
Creating the system generated sequences...
Creating internal dynamic performance views...
Creating outline table...
Creating system package specifications:
    Running /home/tibero/Tibero/tibero7/scripts/pkg/pkg_standard.sql...
    Running /home/tibero/Tibero/tibero7/scripts/pkg/pkg_dbms_output.sql...
    Running /home/tibero/Tibero/tibero7/scripts/pkg/pkg_dbms_lob.sql...
    Running /home/tibero/Tibero/tibero7/scripts/pkg/pkg_dbms_utility.sql...
    ......중간 생략......

Creating spatial meta tables and views ...
Creating internal system jobs...
Creating internal system notice queue ...
Done.
For details, check /home/tibero7/tibero7/instance/tibero/log/system_init.log.
```

▪ Tibero 설치 및 운영

티베로 설치

[system.sh]
티베로를 시작하기 전, 유저 생성이나 역할 부여 등 기본적인 설정을 해주는 스크립트

- 시스템 유저 생성 및 권한 부여
- 데이터베이스 객체의 통계 정보 수집을 위한 job 스케쥴링
- TPR 관련 테이블의 생성 여부

```
tibero@Tibero:~/Tibero/tibero7/scripts$ system.sh
Enter SYS password:

Enter SYSCAT password:

Creating the role DBA...
create default system users & roles?(Y/N):

Creating system users & roles...
Creating virtual tables(1)...
Creating virtual tables(2)...
Granting public access to _VT_DUAL...
Creating the system generated sequences...
Creating internal dynamic performance views...
Creating outline table...
Creating system package specifications:
    Running /home/tibero/Tibero/tibero7/scripts/pkg/pkg_standard.sql...
    Running /home/tibero/Tibero/tibero7/scripts/pkg/pkg_dbms_output.sql...
    Running /home/tibero/Tibero/tibero7/scripts/pkg/pkg_dbms_lob.sql...
    Running /home/tibero/Tibero/tibero7/scripts/pkg/pkg_dbms_utility.sql...
    ......중간 생략......

Creating spatial meta tables and views ...
Creating internal system jobs...
Creating internal system notice queue ...
Done.
For details, check /home/tibero7/tibero7/instance/tibero/log/system_init.log.
```

Q & A

감사합니다