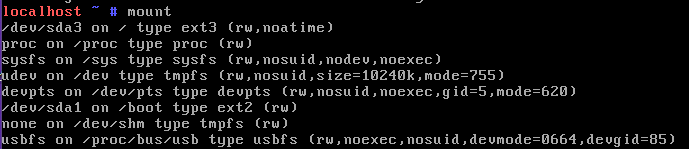
1) Your Gentoo Linux has two disks: /dev/sda3 and /dev/sda1. Which one is the root file system? Where is the mounting point for the other one? Use "mount" command to answer this.



/dev/sda3이 /에 연결돼있기 때문에 /dev/sda3이 root file system이란걸 알 수 있다. 나머지 하나는 /dev/sda1으로 /boot에 연결돼있단걸 알 수 있다.

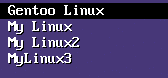
2) Add another entry in /boot/grub/grub.conf as below. This boot selection does not use initrd directive to prevent initramfs loading (initramfs is a temporary in-ram file system used for perfornace improvement).

title=MyLinux3

root (hd0,0)

kernel /boot/bzImage root=/dev/sda3





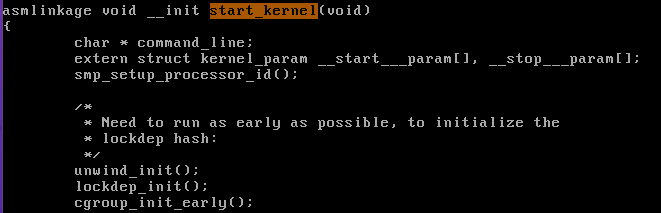
**From now on, use MyLinux3.**

/boot/grub/grub.conf를 고쳐줘서 MyLinux3을 만들어줬다.

3) The kernel calls "mount\_root" to cache the root file system. Starting from "start\_kernel", find out the call chain that leads to "mount\_root".

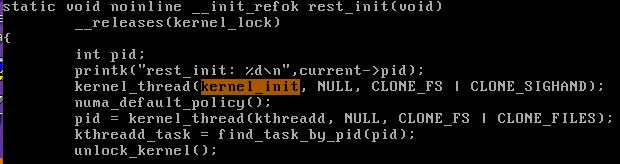
init/main.c

start\_kernel();

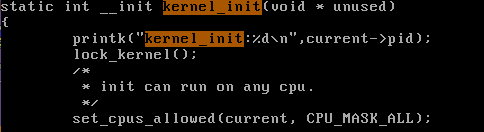


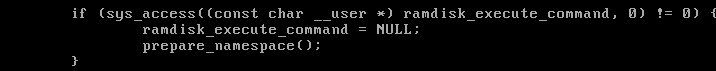


rest\_init()

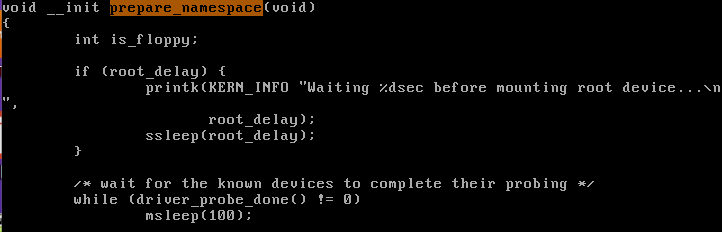


kernel\_init()





prepare\_namespace()





먼저 start\_kernel()이 rest\_init()을 부르고, rest\_init()이 kernel thread를 사용해서 kernel\_init을 만든다. 그러면 kernel\_init()이 prepare\_namespace()를 부르고 prepare\_namespace 안에서 mount\_root()를 부르게 된다.

(start\_kernel -> rest\_init -> make kernel\_init using kernel thread -> prepare\_namespace -> mount\_root)

4) Find the data type for each added variable for super\_block, inode, buffer\_head, and dentry.

(include/linux/fs.h)

superblock :

s\_list => struct list\_head

s\_dev => dev\_t

s\_type => struct file\_system\_type

s\_op => const struct super\_operations

s\_root => struct dentry

s\_files => struct list\_head

s\_id => char

inode :

i\_list => struct list\_head

i\_ino => unsigned long

i\_rdev => dev\_t

i\_count => atomic\_t

i\_op => const struct inode\_operations

i\_sb => struct super block

i\_pipe => struct pipe\_inode\_info

(include/linux/buffer\_head.h)

buffer\_head :

b\_blocknr => sector\_t

b\_bdev => struct block\_device

b\_size => size\_t

b\_data => char

(include/linux/dcache.h)

d\_inode => struct inode

d\_op => struct dentry\_operations

d\_mounted => int

d\_name => struct qstr

5) Change the kernel such that it displays all superblocks before it calls "mount\_root" and after "mount\_root". Boot with MyLinux3 to see what happens.

To display all superblocks, use below.

void display\_superblocks(){

struct super\_block \*sb;

list\_for\_each\_entry(sb, &super\_blocks, s\_list){

printk("dev name:%s dev maj num:%d dev minor num:%d root ino:%d\n",

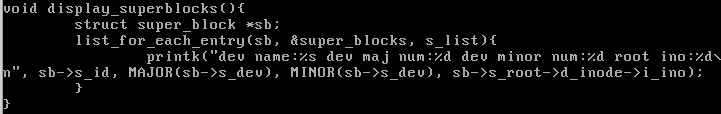
sb->s\_id, MAJOR(sb->s\_dev), MINOR(sb->s\_dev),

sb->s\_root->d\_inode->i\_ino);

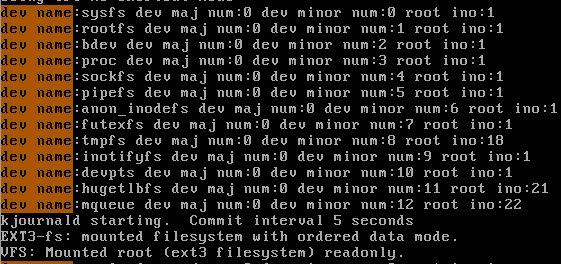
}

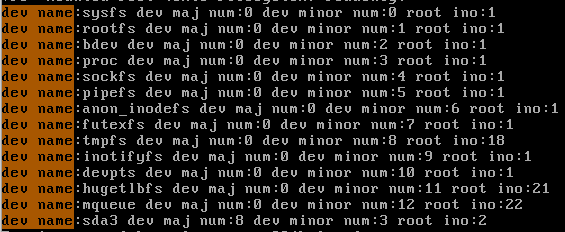
}

init/do\_mounts.c









mount\_root 함수가 불려진 후에는 루트에 실제 파일 시스템인 sda3가 mount되기 때문에 sda3가 새로운 디바이스로 추가된다. (mount\_root 전에는 파일시스템이 없어서 파일에 접근을 못한다.)

6) Change the kernel such that it displays all cached inodes before it calls "mount\_root" and after "mount\_root". Boot with MyLinux3 to see what happens.

To display all cached indoes, use below.

extern struct list\_head inode\_in\_use;

void display\_all\_inodes(){

struct inode \*in;

list\_for\_each\_entry(in, &inode\_in\_use, i\_list){

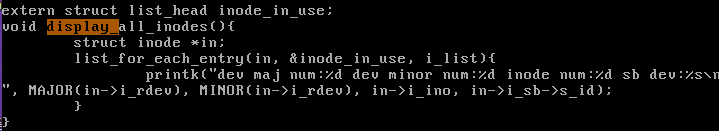
printk("dev maj num:%d dev minor num:%d inode num:%d sb dev:%s\n",

MAJOR(in->i\_rdev), MINOR(in->i\_rdev), in->i\_ino, in->i\_sb->s\_id);

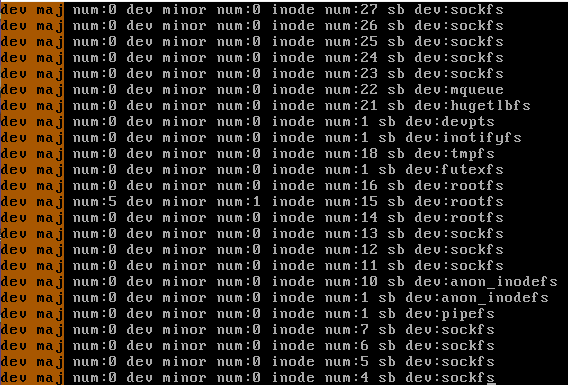
}

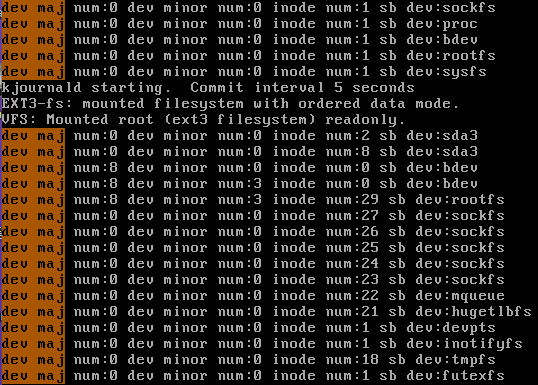
}

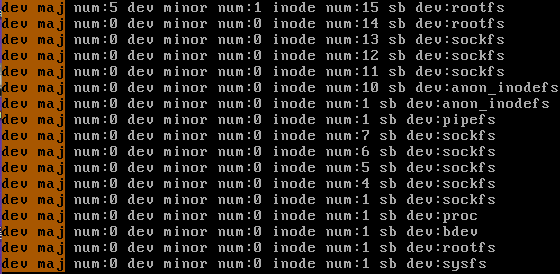
init/do\_mounts.c









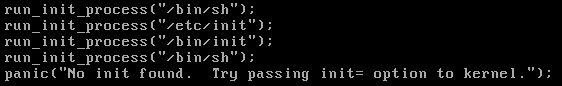


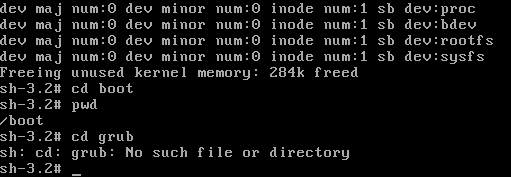
mount\_root가 불려지기 전에는 /dev/sda3가 루트에 연결이 안된 상태이기 때문에 sda3과 관련된 inode들은 출력이 안되지만, mount\_root가 불려진 후에는 /dev/sda3가 루트에 연결이 됐기 때문에 inode 번호가 각각 2, 8, 0, 29번인 sda3, bdev, 그리고 rootfs가 추가된걸 볼 수 있다.

7) The pid=1 process (kernel\_init) eventually execs to /sbin/init with

run\_init\_process("/sbin/init");

by calling kernel\_execve("/sbin/init", ....) in “init/main.c/init\_post()”. Change the kernel such that it execs to /bin/sh. Boot the kernel, and you will find you cannot access /boot/grub/grub.conf. Explain why.





원래는 /sbin/init이 execve되면 kernel\_\_init이 /sbin/init으로 바뀌게 되고 /sbin/init이 /sbin/agetty을 부르게 되는데, 여기에서 초기 프로그램들이 다 실행되고 /dev/sda1이 /dev/sda3의 boot에 연결된다. 그러고 나서 /sbin/agetty는 /bin/login을 부르고 /bin/login은 /bin/sh를 불러서 초기 과정을 전부 다 거친 상태로 /bin/sh에 들어가기 때문에 /boot/grub/grub.conf에 들어갈 수 있게 된다. 하지만 이런 초기과정을 건너뛴채로 바로 /bin/sh에 들어가게 되면, /dev/sda3은 루트에 연결이 됐지만 /dev/sda1은 boot에 연결이 안돼있어서 boot에 연결돼있는 파일시스템은 없기 때문에 inaccessible한 상태가 되는 것이다.