**Student ID: 2016025105**

**Name: 강재훈**

# Project3

## 목차

1. 디자인

1. User accounts

2. File mode

3. Modification of ls

2. 구현

1. Login / Logout

2. Add user / Delete user

3. Owner/others

4. change mode

5. modification of ls

3. 실행 결과

1. project 03 test example.mp4

4. 트러블슈팅

## 1. 디자인

### 1-1. User acoounts

\*\*user accounts\*\*는 xv6에 다수의 유저가 등록될 수 있도록 합니다. 각 유저는 자신의 이름, 비밀번호, 그리고 자신의 디렉토리를 가지고 있습니다. (root 계정은 이름과 비밀번호만) 최초의 유저는 root 1명만 있고 root는 모든 파일에 대해 그 owner와 동일한 권한을 가집니다. 유저의 이름, 비밀번호는 알파벳 대소문자, 숫자로 이루어져있고 1~15글자입니다. 최대 유저 수는 10명입니다. 유저목록은 루트 디렉토리에 1개의 파일로 관리합니다.(userlist) login 프로그램을 만들었고 init 프로세스는 sh 대신 이 프로그램을 실행하게 되며 login 프로그램을 통해 login 한 후 sh를 해당 owner로 실행합니다. 특정 사용자로 로그인 되어있다면 fork나 exec으로 생성된 프로세스도 해당 사용자 권한을 따릅니다. logout은 sh의 cd와 같이 내장명령으로 sh을 종료시키고 wait 중인 login 프로그램으로 돌아갑니다. useradd, userdel 시스템 콜을 통해 user를 추가, 삭제 할 수 있고 이미 있는 유저는 추가가 불가능하며 최대 유저수는 10명까지입니다. 두 시스템콜은 root 유저만이 부를 수 있습니다. 만약 새로운 유저가 만들어졌다면 루트 디렉토리 안에 유저 디렉토리를 만들어주어야 합니다.

따라서 login 프로그램을 만들어 userlist가 없다면 userlist를 만들고 root와 root의 비밀번호를 입력합니다. userlist가 있든 없든 그 후는 동일한대 userlist에서 50byte씩 읽어와 name과 password를 저장합니다. 이후 login 정보가 맞으면 해당 username으로 sh을 실행시켜줍니다. 각 프로세스 마다 username을 가지고 있어야 하므로 proc 구조체 안에 userN을 선언해주어야 합니다. 또한 useradd와 userdel 시스템 콜 안에서 inode 단계로 userlist를 작성 삭제해줍니다.

### 1-2. File mode

\*\*File mode\*\*는 각 디렉토리와 파일들에 권한(read, write, execute)을 부여한 뒤 해당 권한에 맞게 파일을 다룰 수 있게 해주는 것입니다. 파일을 read 하는건 읽는 것, write 하는 건 쓰는 것, execute하는 건 실행 하는 것을 의미하고 디렉토리를 read 하는 건 디렉토리내의 목록을 보는 것, write 하는 건 디렉토리 내의 파일을 생성, 삭제 하는 것, execute 하는 건 디렉토리 내의 파일(디렉토리 안의 디렉토리 포함)에 접근 하는 것을 의미합니다.

따라서 inode, dinode 안에 mode와 owner를 표시 해 줄 수 있는 변수를 선언해주어야 하고 ialloc을 비롯한 여러가지 함수 안에서 적절히 초기화 해주어야 합니다. 그 후 create, open, unlink, chdir, namex, exec 등의 함수 안에서 해당 권한을 확인 한 뒤 위의 설명대로 파일 또는 디렉토리를 사용 할 수 있게 해주어야합니다.

### 1-3. Modification of ls

\*\*Modification of l\*\*는 기존 ls가 출력하는 양식에서 파일인지, 디렉토리인지와 우리가 앞서 추가해준 권한과 owner 를 추가적으로 출력해주는 ls입니다.

## 2. 구현

### 2-1. Login/Logout

login.c는 login 프로그램입니다. 해당 프로그램은 load\_ulist를 통해 userlist가 없다면 만들어주고 root와 root의 비밀번호 1234를 써줍니다. 이후는 userlist가 기존에 있었던 것과 동일하게 userlist로 부터 50byte씩 읽어 username들과 password들을 완성하고 main에서 login하고자 하는 username과 password를 받아 옳다면 로그인 해주고 틀리다면 다시 입력하도록 도와줍니다. logout은 sh의 내장명령으로 구현하였고 getcmd로 받은 내용이 logout이면 sh를 종료하고 wait 중 인 login으로 돌아가게 만들었습니다.

#### 2-2. Add user / Delete user

useradd와 userdel 시스템콜로 구현하였습니다. 두 함수 본체는 file.c에 있고 랩퍼 함수는 sysfile.c에 있습니다. 먼저 현재 실행 중인 user가 root인지를 확인하고 아니면 -1를 반환합니다. 그 후 userlist를 읽으며 자체적으로 username들과 password들을 저장합니다. 이후 useradd는 기존에 list에 있는 이름인지, 총 유저의 수가 10명인지를 확인하고 두 조건이 모두 맞지 않다면 새로운 유저를 추가해줍니다. 새로운 유저를 추가하며 유저이름의 디렉토리가 없을 경우 디렉토리를 만들어줍니다. 만들어주는 함수는 기존 create 함수를 변형한 mkuserdir입니다. userdel은 기존에 있는 user인지, root 유저를 지우려고 하는 것이 아닌지를 확인한 후 user를 지워줍니다.

#### 2-3. Owner/others

proc구조체 안에 userN을 만들어주어 현재 실행되고 있는 user를 저장하게 합니다.

그리고 inode구조체와 dinode구조체에 mode와 owner 변수를 추가해 해당 inode의 mode와 owner를 저장할 수 있게 해줍니다. 이때 dinode의 크기는 512의 변수가 되어야 하므로 NDIRECT를 15로 변경하여 총 128byte가 되게 해줍니다. 이후 ialloc 함수를 통해 처음 만들어지는 모든 파일에 other write를 뺀 모든 권한을 주고 owner는 root로 지정해줍니다. 이후 create, open, unlink, chdir, namex, exec 등에서 새로 만들어지는 파일들에 대해 적절한 권한을 주기도 하고 권한을 체크하여 해당 파일과 디렉토리에 대한 접근을 허용하기도, 거부하기도 합니다. create는 sysfile.c의 create 함수 안에서 조건을 확인하고, open은 sysfile.c의 sys\_open에서, unlink는 sysfile.c의 sys\_unlink에서, chdir은 sysfile.c의 sys\_chdir에서, namex는 fs.c의 namex 함수에서, exec은 exec.c의 exec에서 권한이 맞는지 틀리는지를 검사합니다.

키포인트는 정확한 타이밍의 ilock 이었습니다. ilock을 정확한 타이밍에 해주지 않고 내가 원할때 lock하고 unlock 하면 생각했던대로 구현되지 않고 inode에 다른 값이 들어가 있는 것을 확인하였습니다.

#### 2-4. Change mode

chmod 함수와 랩퍼 함수는 sysfile.c에 있습니다. chmod 함수는 namei를 통해 해당 path의 inode를 가져오고 ilock, iupdate등의 함수를 이용해 inode와 dinode에 변경된 mode 값을 저장해줍니다. 해당 함수는 root 유저만 사용 할 수 있습니다.

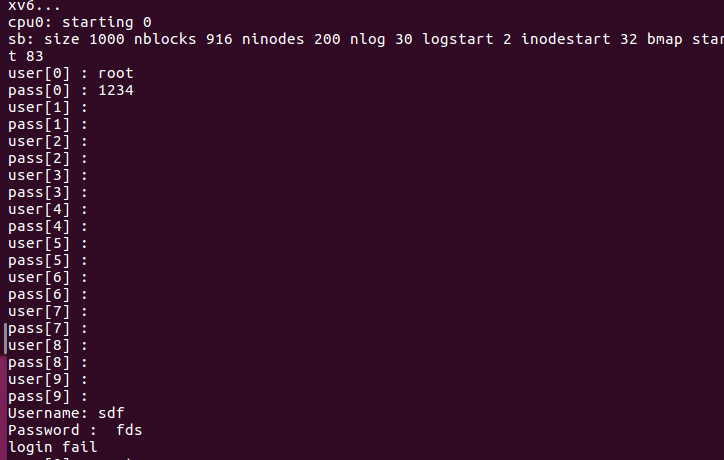
#### 2-5. Modification of ls

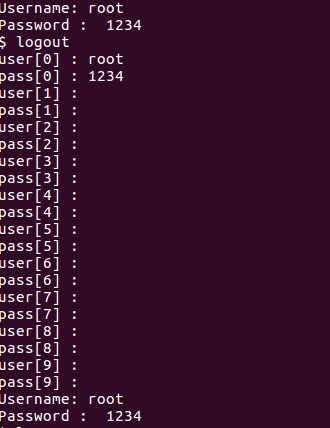
변경된 ls는 파일인지, 디렉토리인지 구별해주고 해당 목표물의 권한을 순차적으로 출력해줍니다. fstat으로 가져온 정보를 토대로 type을 체크하고 각 권한마다 bit and 연산을 한 후 내용을 완성해 출력해줍니다.

----

## 3. 실행결과

project03 test example.mp4를 따라해본 결과입니다.

 잘못된 유저네임 패스워드 입력시 로그인 실패

 root 유저로 로그인

텍스트이(가) 표시된 사진

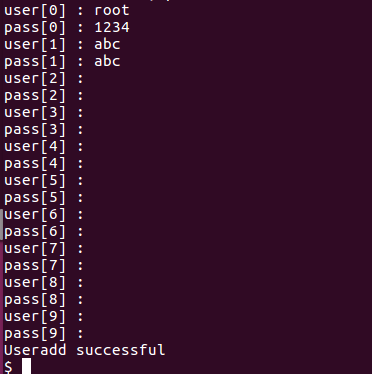
자동 생성된 설명 ls 실행

검은색, 사진, 빨간색, 어두운이(가) 표시된 사진

자동 생성된 설명 ls 실행2, userlist 생성된 것 확인

음식이(가) 표시된 사진

자동 생성된 설명 useradd로 abc유저 생성

 useradd successful

키보드, 컴퓨터이(가) 표시된 사진

자동 생성된 설명 logout하고 이전에 만든 abc로 로그인

텍스트이(가) 표시된 사진

자동 생성된 설명 ls 실행으로 abc 디렉토리 생성 확인

스크린샷이(가) 표시된 사진

자동 생성된 설명

재실행시에도 userlist 유지 되는것 확인

컴퓨터, 키보드, 전화이(가) 표시된 사진

자동 생성된 설명 abc로 로그인

오렌지, 테이블, 쥐고있는, 전화이(가) 표시된 사진

자동 생성된 설명 root가 아닌 abc에서 add 했기 때문에 fail

키보드, 컴퓨터, 방이(가) 표시된 사진

자동 생성된 설명 생성되지 않은 123으로 로그인 할 수 없음

사진, 검은색, 빨간색, 표지판이(가) 표시된 사진

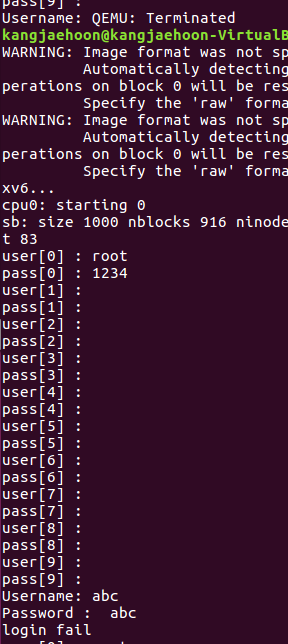
자동 생성된 설명root로 로그인 후 abc 삭제

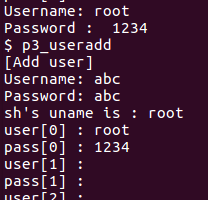
컴퓨터, 키보드이(가) 표시된 사진

자동 생성된 설명 abc 삭제 완료

키보드, 컴퓨터이(가) 표시된 사진

자동 생성된 설명 abc 삭제 후 abc로 로그인 불가

xv6 종료 후 삭제된 abc로 로그인 하려 할 시 실패



사진, 음식, 플레이어, 남자이(가) 표시된 사진

자동 생성된 설명 root로 들어가서 다시 abc 만들어주기

키보드, 컴퓨터이(가) 표시된 사진

자동 생성된 설명 root를 지우려 할 시 실패

텍스트, 테이블이(가) 표시된 사진

자동 생성된 설명 sdf 파일에 sdf라 쓰기, sdf 실행은 불가

사진, 표지판, 쥐고있는, 사람들이(가) 표시된 사진

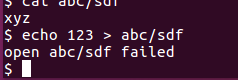
자동 생성된 설명 abc폴더 안에 sdf 만들고 xyz 내용쓰고 확인

키보드이(가) 표시된 사진

자동 생성된 설명 abc로 로그인하고 다시 해보기

표지판, 방이(가) 표시된 사진

자동 생성된 설명 abc로 열기 가능

 abc유저로 쓰기는 불가능

사진, 오렌지, 표지판, 사람들이(가) 표시된 사진

자동 생성된 설명

abc유저로 abc디렉토리에 파일 만들 순 있지만 root 디렉토리엔 불가능

키보드이(가) 표시된 사진

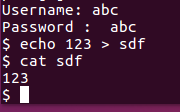
자동 생성된 설명 다시 root유저로 로그인

텍스트, 테이블이(가) 표시된 사진

자동 생성된 설명 권한 변경 성공, ls로 확인 가능

키보드, 컴퓨터이(가) 표시된 사진

자동 생성된 설명 abc유저로 다시 로그인

 sdf에 123 쓰고 cat으로 내용 확인

사진, 테이블, 컴퓨터, 표지판이(가) 표시된 사진

자동 생성된 설명 abc유저로 abc 폴더 권한 변경 후 접근 시 실패

사진, 오렌지, 닫기, 화면이(가) 표시된 사진

자동 생성된 설명 abc유저로 abc 폴더 권한 한번 더 변경시 목록조차 볼수 없음

## 4. 트러블슈팅

### 4-1. login

평소 c언어에서 사용하는 fseek, fopen 등의 함수로 완성하였는데 컴파일해보니 해당 함수가 없다고 하여 놀랐습니다. 급하게 조교님께 문의하였고 실습시간에 나온 여러가지 함수를 사용하면 된다고 하셔서 그 함수들을 활용하여 완성하였습니다. 고정적인 크기를 읽고 고정적인 크기를 쓰는것이 좋다고 하셔서 처음에는 파일 전체를 읽어와 문장을 파싱해 쓰는 방식으로 구현했지만 이후 기능들을 구현하다보니 한계를 느꼈고 조교님께서 말씀하신 의미를 깨달아 지금과 같이 구현하였습니다.

### 4-2. logout

기login과 마찬가지로 프로그램을 구현해야 하는줄 알았지만 조교님께서 ls보다는 cd에 가깝다는 조언을 해주셔서 sh.c를 분석한 후 내장 명령으로 구현하였습니다.

### 4-3. useradd

useradd 함수 안에서 유저의 디렉토리를 만들어 줄 때 기존에 존재하던 create함수를 사용했었습니다. 이후 ls를 구현하고 나니 이렇게 구현하면 owner가 해당 유저가 아닌 root로 들어가는 것을 보고 create함수를 변형한 mkuserdir함수를 만들었습니다.

### 4-4. filemode

피디에프 후반부에 있는 가이드라인을 읽기 전까지는 막막했습니다. 어떤 방식으로 접근해야 하는지, 어느 부분을 고쳐줘야 하는지 몰랐기 때문입니다. 실습 수업을 반복해서 듣고 조교님이 써주신 가이드라인을 천천히 이해하며 진행하였습니다. 그렇게 구현한 코드가 완벽하다고 생각하였는데 자꾸 불러온 inode의 값이 제가 원하던 값이 아닌 것을 확인하고 한참을 헤맸습니다. 원인은 ilock을 소홀히 한점, 엉뚱한 곳에 한 점에 있었고 원인을 발견하고 난 뒤 코드를 전반적으로 수정하여 지금의 코드를 만들었습니다.

### 4-5. chmod

구현 전반적인 과정에는 어려움이 없었으나 ilock을 소홀히 하여 생기는 문제에서 시간을 한참 허비했습니다.

### 4-6. userlist

마지막으로 userlist에 대해 cat userlist를 실행했을 경우 처음 root 유저만 들어 있을때는 문제 없이 출력되지만 이후 user가 추가되고 삭제된 후에 cat userlist를 실행하면 쓰레기 값이 출력되는 것을 확인하였습니다. 하지만 프로그램이 돌아가는 전반적인 부분에는 전혀 문제가 없었고 적절한 시기마다 userlist를 읽어와 쉘에 출력해주어 userlist를 확인하는 방법을 택했습니다.