컴퓨터학부 20201841 박세연

<chmod1>

1. 소스코드

**[ssu\_chmod\_1.c]**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/stat.h>

int main(void){

struct stat statbuf;

char \*fname1 = "ssu\_file1";

char \*fname2 = "ssu\_file2";

// stat함수로 fname1의 파일 정보를 statbuf에 저장

if(stat(fname1, &statbuf) < 0)

fprintf(stderr, "stat error %s\n", fname1);

// chmod함수로 접근 권한을 변경

// (statbuf.st\_mode & ~S\_IXGRP) : 현재 접근 권한에서 S\_IXGRP(그룹의 실행)권한을 제거

// | S\_ISUID : set user id 권한을 추가

if (chmod(fname1, (statbuf.st\_mode & ~S\_IXGRP) | S\_ISUID) < 0)

fprintf(stderr, "chmod error %s\n", fname1);

// user의 읽기, 쓰기 권한과 group의 읽기 권한과 other의 읽기, 실행권한을 부여

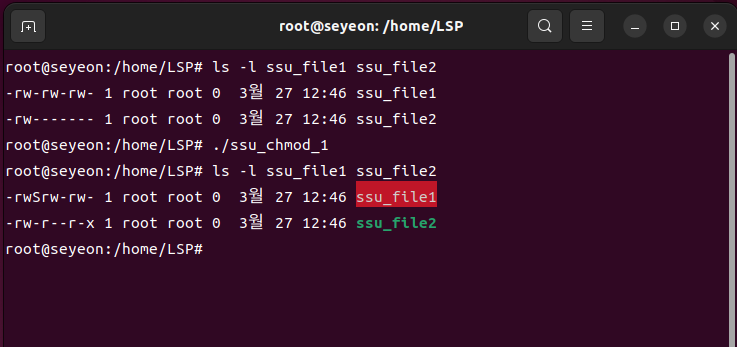
if (chmod(fname2, S\_IRUSR | S\_IWUSR | S\_IRGRP | S\_IROTH | S\_IXOTH) < 0)

fprintf(stderr, "chmod error %s\n", fname2);

exit(0);

}

1. 실행결과



<chmod2>

1. 소스코드

**[ssu\_chmod\_2.c]**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/stat.h>

#define MODE\_EXEC (S\_IXUSR | S\_IXGRP | S\_IXOTH)

// user의 실행권한, group의 실행권한, other의 실행권한

int main(int argc, char \*argv[])

{

struct stat statbuf;

int i;

// 파일명을 인자로 받지 못한 경우 에러 처리

if (argc < 2) {

fprintf(stderr, "usage: %s <file1> <file2> ... <fileN>\n", argv[0]);

exit(1);

}

for (i = 1; i < argc; i++) {

if (stat(argv[i], &statbuf) < 0) { // stat함수로 파일의 정보를 statbuf에 저장

fprintf(stderr, "%s : stat error\n", argv[i]);

continue;

}

statbuf.st\_mode |= MODE\_EXEC;

// 파일의 현재 접근 권한에서 MODE를 추가(위의 주석에 있는 권한들)

statbuf.st\_mode ^= (S\_IXGRP | S\_IXOTH);

// XOR연산자로 반전 시킴.

// group과 other의 실행권한을 다시 제거

if (chmod(argv[i], statbuf.st\_mode) < 0)

fprintf(stderr, "%s : chmod error\n", argv[i]);

else

printf("%s : file permission was changed.\n", argv[i]);

}

exit(0);

}

// /bin/su : 일반 사용자 권한으로 파일 접근 권한을 변경할 수 없음 -> 에러메세지 출력

// /home/user/새로만든디렉터리 : 파일의 권한이 바뀌었음을 출력

2. 실행결과

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

<chown>

1. 소스코드

**[ssu\_chown.c]**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/stat.h>

int main(void) {

struct stat statbuf;

char \*fname = "ssu\_myfile";

int fd;

if ((fd = open(fname, O\_RDWR | O\_CREAT, 0600)) < 0) {

fprintf(stderr, "open error for %s\n", fname);

exit(1);

}

close(fd);

stat(fname, &statbuf); // fname의 파일 정보를 statbuf에 저장

printf("# 1st stat call # UID:%d GID:%d\n", statbuf.st\_uid, statbuf.st\_gid);

//소유자 ID와 그룹 사용자 ID를 출력

// chown함수를 이용해 UID, GID를 변경

if (chown(fname, 501, 300) < 0){

fprintf(stderr, "chown error for %s\n", fname);

exit(1);

}

stat(fname, &statbuf); // fname의 파일 정보를 statbuf에 다시 저장

printf("# 2nd stat call # UID:%d GID:%d\n", statbuf.st\_uid, statbuf.st\_gid);

// 바뀐 소유자 ID와 그룹 사용자 ID를 출력

if (unlink(fname) < 0) {

fprintf(stderr, "unlink error for %s\n", fname);

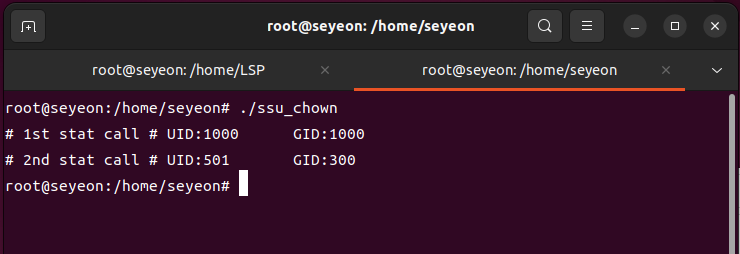
exit(1);

}

exit(0);

}

2. 실행결과



<link>

1. 소스코드

**[ssu\_oslab.c]**

#include <stdio.h>

#include <stdlib.h>

int main(void){

printf("This is oslab file\n");

exit(0);

}

**[ssu\_link.c]**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

int main(int argc, char \*argv[]){

// 인자로 파일을 2개 받아야함

if (argc < 3){

fprintf(stderr, "usage: %s <file1> <file2>\n", argv[0]);

exit(1);

}

// link함수로 argv[1]로 지정한 파일과 같은 파일을

// argv[2]로 지정한 이름으로 새로운 파일을 생성 (동일한 디스크 공간을 가리킴)

if (link(argv[1], argv[2]) == -1) {

fprintf(stderr, "link error for %s\n", argv[1]);

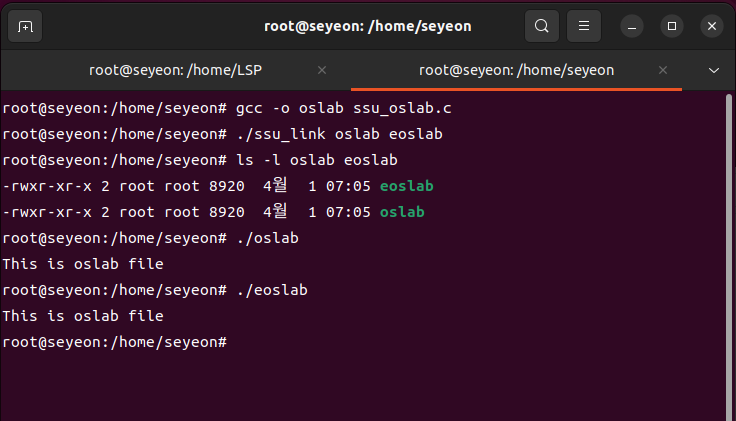
exit(1);

}

exit(0);

}

2. 실행결과



<unlink1>

1. 소스코드

**[ssu\_dump.txt]**

Linux System Programming!

**[ssu\_unlink\_1.c]**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

int main(void){

char \*fname = "ssu\_dump.txt";

if (open(fname, O\_RDWR) < 0) {

fprintf(stderr, "open error for %s\n", fname);

exit(1);

}

// unlink함수로 파일의 링크를 해제

if (unlink(fname) < 0) {

fprintf(stderr, "unlink error for %s\n", fname);

exit(1);

}

printf("File unlinked\n"); // 파일이 해제되었음을 알리는 출력문

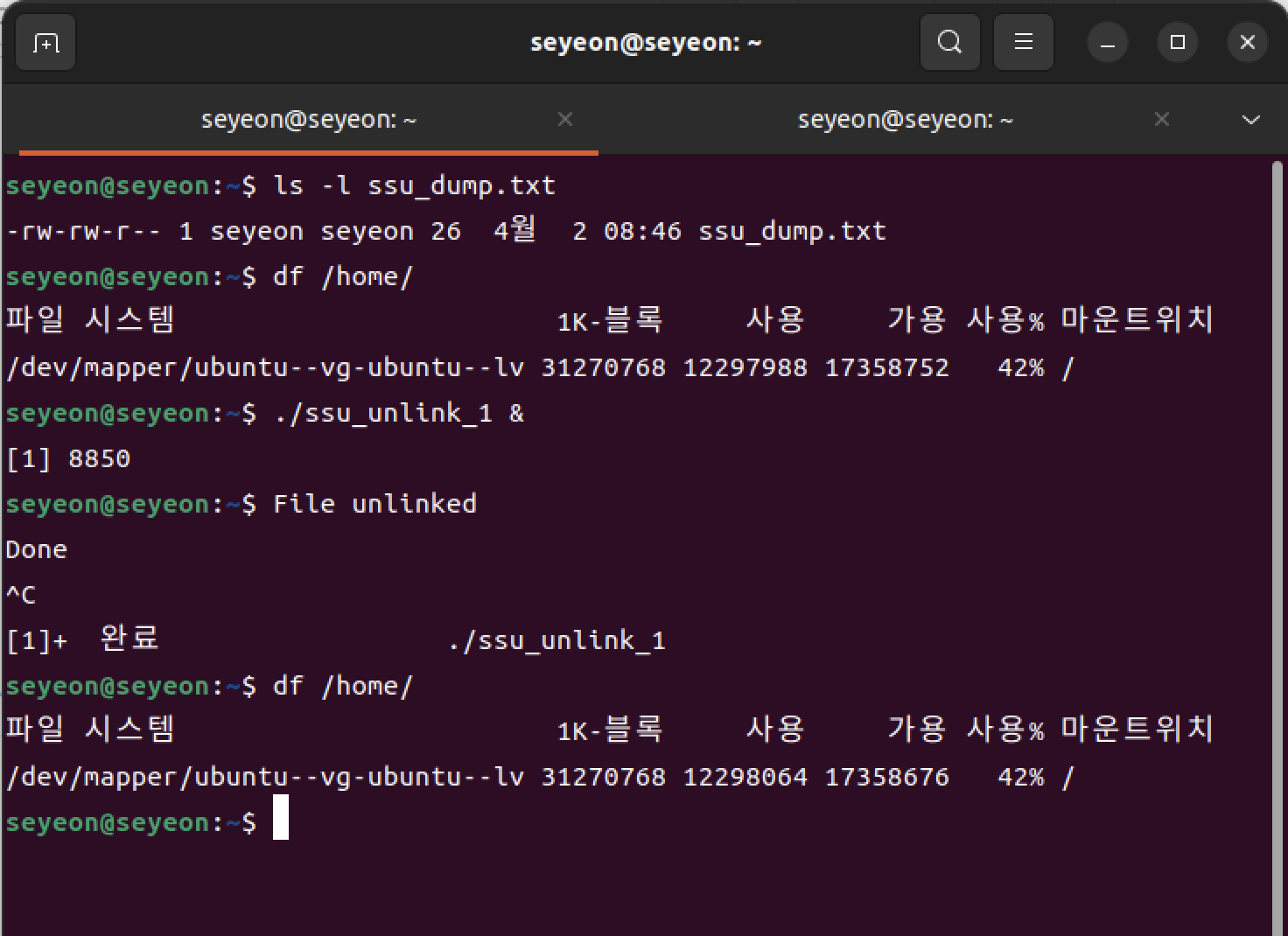
sleep(20);

printf("Done\n");

exit(0);

}

2. 실행결과



<unlink2>

1. 소스코드

**[ssu\_unlink\_2.c]**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

int main(void) {

char buf[64];

char \*fname = "ssu\_tempfile";

int fd;

int length;

if ((fd = open(fname, O\_RDWR | O\_CREAT | O\_TRUNC, 0600)) < 0)

{

fprintf(stderr, "first open error for %s\n", fname);

exit(1);

}

// unlink로 파일의 링크를 해제 -> unlink는 파일을 닫은 후 삭제됨

if (unlink(fname) < 0){

fprintf(stderr, "unlink error for %s\n", fname);

exit(1);

}

if (write(fd, "How are you?", 12) != 12) {

fprintf(stderr, "write error\n");

exit(1);

}

lseek(fd,0,0);

// unlink는 파일을 닫은 후 삭제되기 때문에 read로 읽고 출력함

if ((length = read(fd, buf, sizeof(buf))) < 0) {

fprintf(stderr, "buf read error\n");

exit(1);

}

else

buf[length] = 0;

printf("%s\n", buf);

close(fd);

// 다시 open함수로 삭제했던 파일을 열면 에러 발생

if ((fd = open(fname, O\_RDWR)) < 0) {

fprintf(stderr, "second open error for %s\n", fname);

exit(1);

}

else {

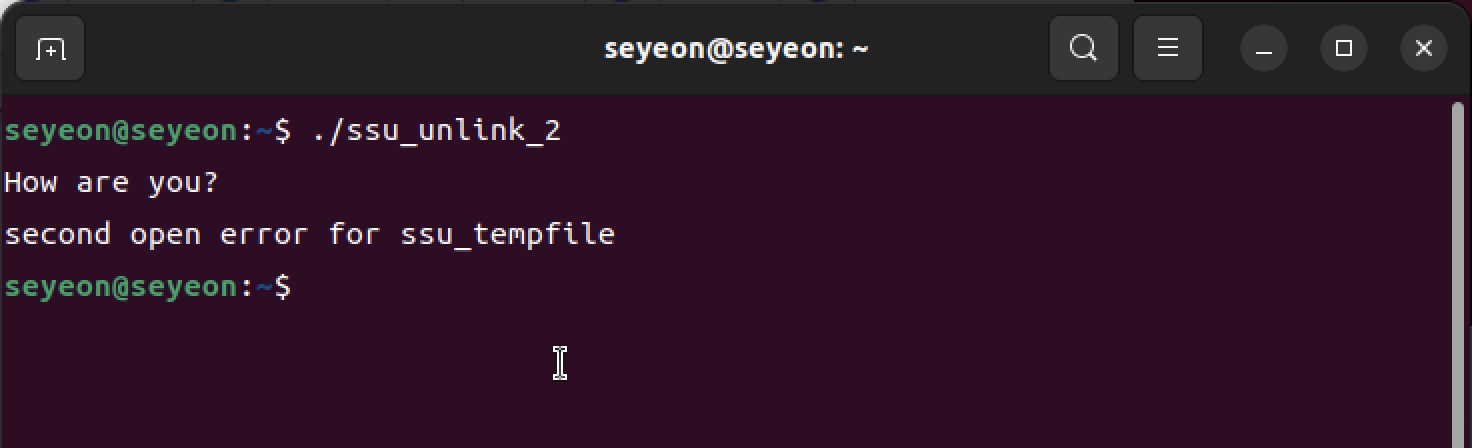
close(fd);

}

exit(0);

}

2. 실행결과



<remove>

1. 소스코드

**[ssu\_remove.c]**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

int main(int argc, char \*argv[]){

// 변경하고자하는 파일, 새이름을 인자로 받아야 함

if(argc != 3){

fprintf(stderr, "usage: %s <oldname> <newname>\n", argv[0]);

exit(1);

}

// link로 파일을 생성

if (link(argv[1], argv[2]) < 0) {

fprintf(stderr, "link error\n");

exit(1);

}

else {

printf("step1 passed.\n"); // 에러발생안하면 출력

}

// remove함수로 oldname에 지정한 파일을 unlink함

// 파일의 해당하는 디스크 공간은 해제되지 않고

// link함수로 생성한 파일이 그 디스크 공간을 가리킴

if (remove(argv[1]) < 0)

{

fprintf(stderr, "remove error\n");

remove(argv[2]);

exit(1);

}

else{

printf("step2 passed.\n"); // 에러발생안하면 출력

}

printf("Success!\n");

exit(0);

}

2. 실행결과

텍스트, 스크린샷, 멀티미디어, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

<rename>

1. 소스코드

**[ssu\_rename.c]**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

int main(int argc, char \*argv[]){

int fd;

// 인자로 파일 이름 2개를 받아야함

if (argc != 3){

fprintf(stderr, "usage: %s <oldname> <newname>\n", argv[0]);

exit(1);

}

// oldname에 지정한 파일을 open함

if ((fd = open(argv[1], O\_RDONLY)) < 0) {

fprintf(stderr, "first open error for %s\n", argv[1]);

exit(1);

}

else

close(fd);

// rename함수를 이용하여 파일의 이름을 newname에 지정한 이름으로 변경

if (rename(argv[1], argv[2]) < 0) {

fprintf(stderr, "rename error\n");

exit(1);

}

// 이름이 newname으로 변경되었기 때문에, oldname으로 open하면 에러 발생

if ((fd = open(argv[1], O\_RDONLY)) < 0)

printf("second open error for %s\n", argv[1]);

else{

fprintf(stderr, "it's very strange!\n");

exit(1);

}

printf("Everything is good!\n");

exit(0);

}

2. 실행결과

텍스트, 스크린샷, 멀티미디어, 소프트웨어이(가) 표시된 사진

자동 생성된 설명