강의 내용

binaryfile = non textfile \*

ㄴ cat,vi 는 내용을 문자로 해석하기 때문에 binary file 볼 수 없다

ㄴ xxd 는 내용을 확인할 수 있다.

16진수 – 두개의 숫자가 1byte

litle endian / big endian 방식 등이 있습니다. – 컴퓨터는 반대로 표현되는 little endian 방식으로

RIFF header

20 2 audioformat = 1 = 압축안됐다

22 2 numcahnnel

24 4 sample rate = 초당 샘플 수

36 4 subchunk2 – 실제 소리 관련,,

lseek = 파일을 원하는 위치에서부터 읽을 수 있도록 r/w포인터를 움직여주는 함수

fprintf – ‘원하는’ 파일에 넣고, 그 파일을 출력해주는 느낌

&x = x의 주소

소리는 - 데이터를 0 으로 바꿔주면 묵음으로 처리해 줄 수 있습니다.

\*psftp 사용해서 소리파일 받아서 확인해 보기

-숙제 2번에서 문자열은 %s로 숫자는 %x 나 %d 로 출력하기 바랍니다

HW

1) Read swvader03.wav with "xxd". Interpret all fields in the header.

First copy swvader03.wav file from ../../linuxer1 directory into current directory. "**.**" means current directory.

$ cp ../../linuxer1/swvader03.wav **.**

(or cp ../../linuxer2/swvader03.wav **.** in 165.246.38.152)

Look at the file with xxd.

$ xxd swvader03.wav > x

$ vi x

......

The answer should show the byte sequence and the value of each field as follows:

ChunkID: 52 49 46 46 : RIFF

ChunkSize: 34 76 00 00 : 30260

Format : 57 41 56 45 : WAVE

Subchunk1ID : 66 6d 74 20 : fmt

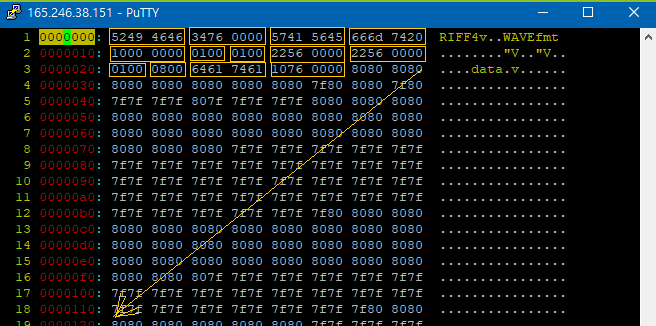
Subchunk1Size: 10 00 00 00 : 16

.............





wav파일이 잘 cp된 것을 확인할 수 있었습니다.



1. ChunkID = 46464952 > 1179011410 > RIFF

2. ChunkSize = 00007634 > 30260

3. Format = 45564157 > 1163280727 > WAVE

4. SubChunk1ID = 20746d66 > 544501094 > fmt  
5. SubChunk1Size = 00000010 > 16

6. AudioFormat = 0001 > 1

7. NumChannels = 0001 > 1

8. SampleRate = 00005622 > 22050

9. ByteRate = 00005622 > 22050

10. BlockAlign = 0001 > 1

11. BitsPerSample = 0008 > 8

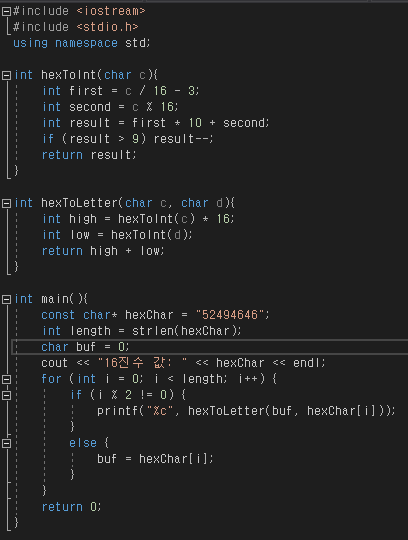
12. SubChunk2ID = 61746164 > 1635017060 > data

13. SubChunk2SIze = 00007610 > 30224

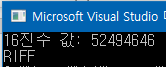
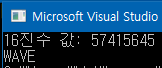
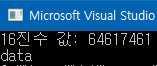
14. .. sound data = .

각 블록은 의미를 가지고 있는 것을 눈으로 확인할 수 있었습니다.

각 정보를 확인하기 위해, hex to string 코드를 작성하여 값을 구해보았습니다.



사용한 VS코드는 위와 같습니다.

  … 

2) Write a program that reads swvader03.wav and displays the content as above.

..............

char ChunkID[10]; // use char array for text data

int ChunkSize; // use "int" for 4 byte data

char Format[10];

........

short AudioFormat; // use "short" for 2 byte data

........

x=open("./swvader03.wav", ...........);

y=read(x, ChunkID, 4); // read first 4 bytes into ChunkID[]

ChunkID[y]=0; // to print as a string

y=read(x, &ChunkSize, 4); // read next 4 bytes and store at address &ChunkSize

y=read(x, Format, 4); // read "WAVE"

Format[y]=0;

.......

y=read(x, &AudioFormat, 2); // read next 2 bytes and store at address &AudioFormat

..........

printf("ChunkID:%s\n", ChunkID);

printf("ChunkSize:%d\n",ChunkSize);

printf("Format:%s\n",Format);

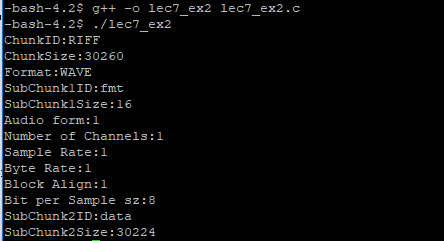
.......

printf("AudioFormat:%d\n", AudioFormat);

.......



전체코드는 위와 같습니다.



1) 번에서 구했던 값과 동일한 것을 확인할 수 있었습니다.

3) Same as 2), but display the content in file sw2-wav.txt. Using "write()" to write into a text file is very hard. Use fopen() and fprintf() for formatted output.

..........

x=open("./swvader03.wav", ...........); // input file

FILE \*fout=fopen("sw2-wav.txt", "w"); // output file

y=read(x, ChunkID, 4); // read "RIFF"

ChunkID[y]=0; // to print as a string

y=read(x, &ChunkSize, 4); // read chunk size

y=read(x, Format, 4); // read "WAVE"

Format[y]=0;

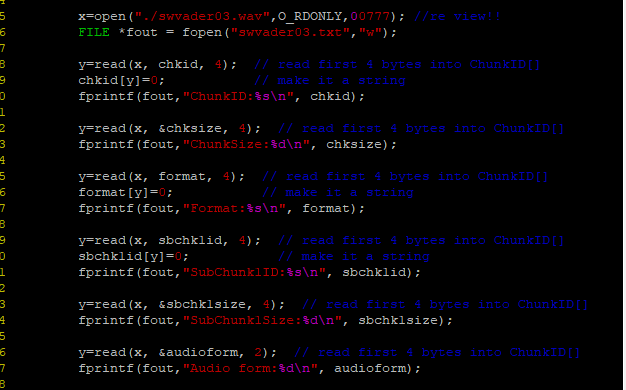
.......

fprintf(fout,"ChunkID:%s\n", ChunkID); // write to sw2-wav.txt

fprintf(fout, "ChunkSize:%d\n",ChunkSize);

fprintf(fout, "Format:%s\n",Format);

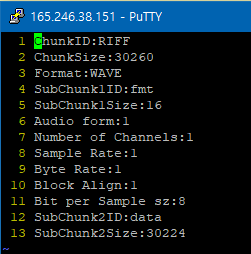
.......



2)번과 코드는 거의 동일하지만, file 과 fprintf를 사용해 주었습니다.







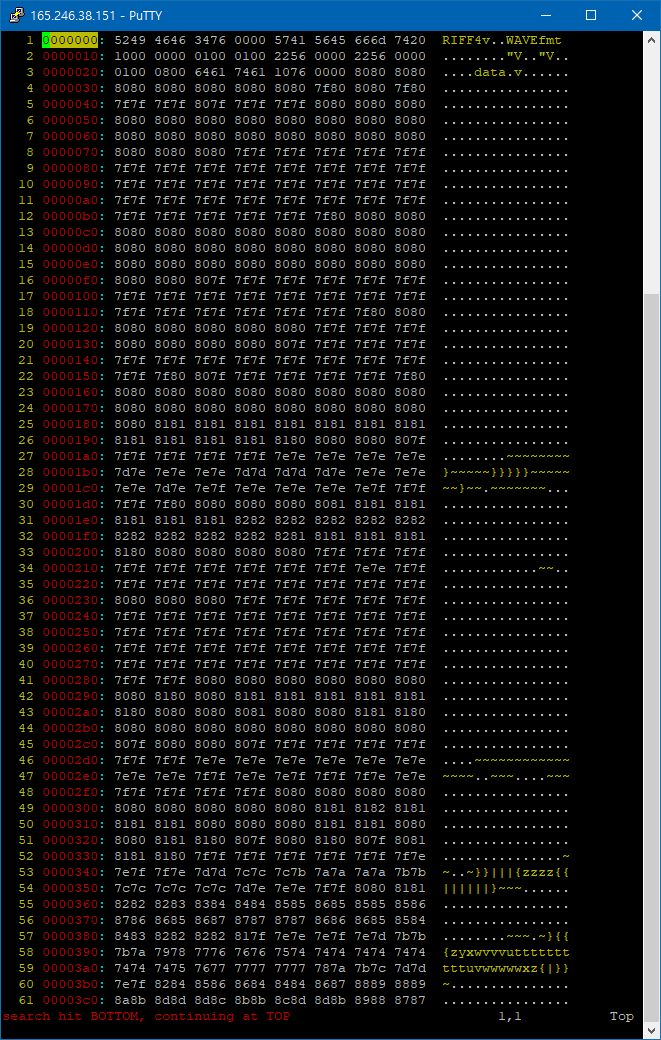
컴파일 후, 제가 데이터를 넣기로 결정한 txt파일의 이름(swvader.03)을 vi 문으로 열어보니

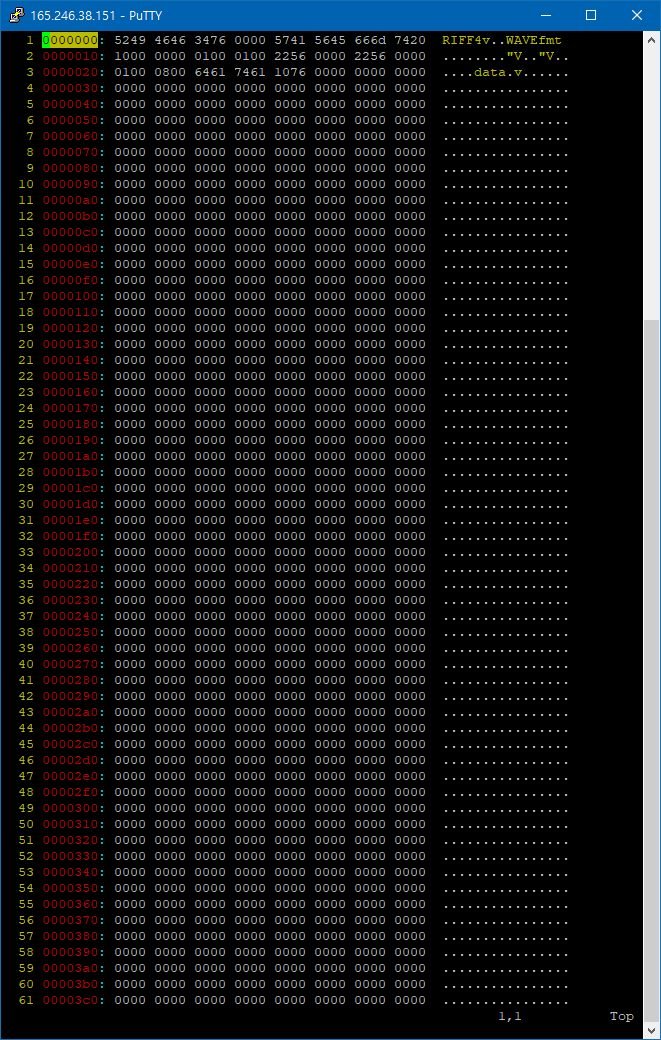
결과는 위와 같았습니다.

4) swvader03.wav contains a sentence, “Yes, my master”. Write a program that modifies the file such that it contains only “master”. Move the file read pointer to the start of the actual sound data with lseek() and write 0 for half of the sound data, since "Yes, my" and "master" take about half of the sound data each. It will be better that you copy swvader03.wav to sw2.wav and modify sw2.wav. When you modified the file, you need to download it to your PC using psftp (look at Section 7 for the explanation for psftp).

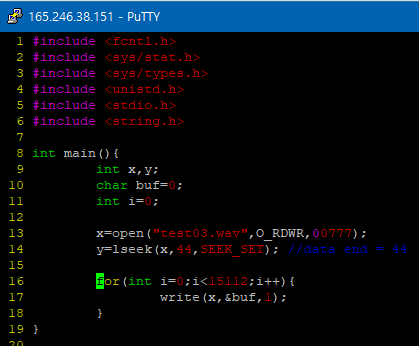


원본 파일을 망칠까봐 형식만 같은 다른 파일에 복사를 해주어서 진행하도록 하겠습니다.





0으로 값이 변경되었습니다.

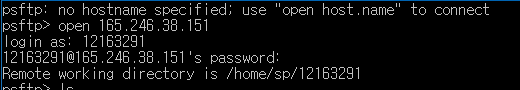


사용한 코드는 위와 같습니다.

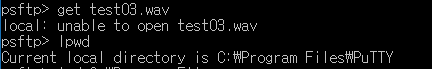
lseek를 통해, 첫 위치를 data가 들어있는 공간만큼 -44byte 뒤로 옮겨주고

그때부터, 중간.

소리가 들어있는 파일의 크기는 전체30224 이므로, 30224/2 = 15112 까지를 0으로 채워주었습니다.



파일을 받기위해 psftp를 실행해 주었습니다.



get 함수를 사용하여 파일을 받아오려 하였습니다.

잘되지않아, 구글링하여 검색해보니 local 주소가 잘못되었나 싶어

lpwd 로 로컬 주소를 확인하고,



lcd를 통해 바탕화면으로 루트를 바꿔주었습니다.

의미는 없는 것 같았습니다만,,





putty에서

chmod 커맨드를 이용해 permission범위를 넓혀 주었습니다.

그후  이던 허용범위가 모두 허용으로 변경된 것을 확인할 수 있었습니다.



get + 절대경로를 입력하니

파일이 잘 다운받아졌습니다.. – 이것이 문제였던 것 같습니다.

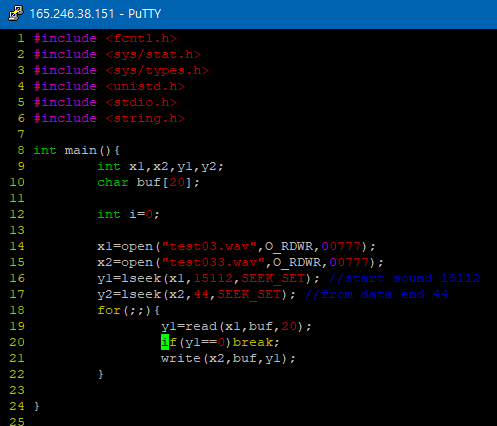


원본파일도 다운받아 차이를 확인하였습니다.

원본인 swvader03.wav 파일과달리

test03.wav 파일은 들어보니 master만 들리는 것이 코드가 잘 실행된 것 같았습니다.

5) Write a program that modifies the wav file such that it contains "master" twice. That is, when you play this file you should here "master master".



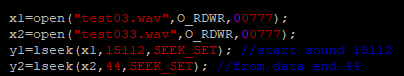
코드는 위와 같습니다.



이미 앞부분이 비워져있는 (사실 상관없습니다. 어짜피 덧붙여이는과정이기 때문에, 원본에 해도 결과는 같습니다.)

test03.wav 파일의 복사본을 생성합니다.

그 후,

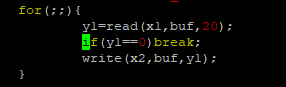


각 파일의 시작점을 정해줍니다.

복사할 내용인 x1은 시작점을 15112(중간지점 – master의 시작점)

복사될 내용인 x2는 시작점이 44(데이터 끝점) 입니다.

그후,

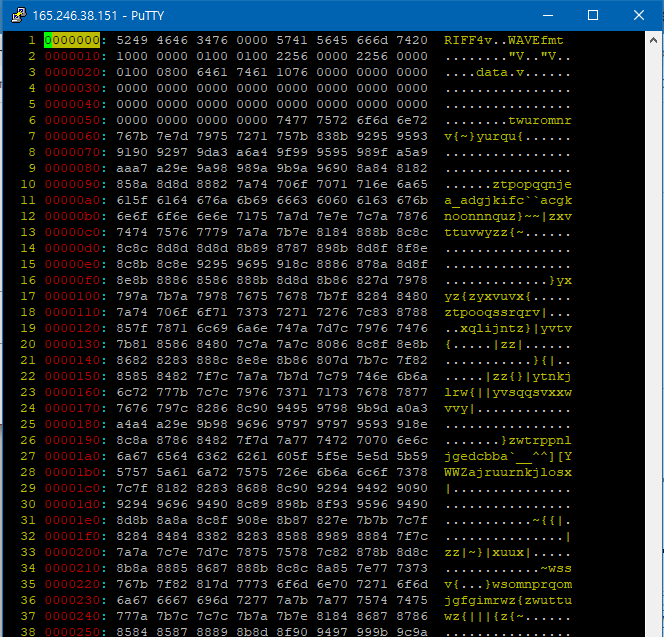


x1파일이 빌때까지 x2파일에 15112이후의 내용을 복붙해줍니다.

파일길이가 30224… 인것을 알고있기 때문에 master master 의 출력이 가능하게 됩니다.



로 test033.wav의 data 직후 부분에 소리 데이터 값이 있는지 확인하여 보았습니다.



data부분 이후로 어떠한 소리 값이 들어온 것을 확인할 수 있었습니다.



psftp로 파일을 받아 확인하여보니, 결과가 잘 출력된 것을 확인할 수 있었습니다.

6) Use gdb (-g + gdb) to debug the error in following code.

#include<fcntl.h>

#include<sys/stat.h>

#include<sys/types.h>

#include<unistd.h>

#include<stdio.h>

int main(){

char chunkID[10];

int chunkSize;

char format[10];

short AudioFormat;

short NumChannel;

int SampleRate;

int ByteRate;

short BlockAlign;

short BitsPerSample;

char data[20];

int x,y;

x = open("./swvader03.wav", O\_RDONLY, 00777);

x = read(x, chunkID, 4);

chunkID[y] = 0;

y = read(x, &chunkSize, 4);

y = read(x, format, 4);

format[y] = 0;

printf("chunkID : %s ", chunkID);

printf("chunkSize : %d ", chunkSize);

printf("format : %s ", format);

printf("\n");

y = read(x, chunkID, 4);

chunkID[y] = 0;

y = read(x, &chunkSize, 4);

y = read(x, &AudioFormat, 2);

y = read(x, &NumChannel, 2);

y = read(x, &SampleRate, 4);

y = read(x, &ByteRate, 4);

y = read(x, &BlockAlign, 2);

y = read(x, &BitsPerSample, 2);

printf("chunkID : %s ", chunkID);

printf("chunkSize : %d ", chunkSize);

printf("AudioFormat : %d ", AudioFormat);

printf("NumChannel : %d ", NumChannel);

printf("ByteRate : %d ", ByteRate);

printf("BlockAlign : %d ", BlockAlign);

printf("BitsPerSample : %d", BitsPerSample);

printf("\n");

y = read(x, chunkID, 4);

chunkID[y] = 0;

y = read(x, &chunkSize, 4);

printf("chunkID : %s ",chunkID);

printf("chunkSize : %d", chunkSize);

printf("\n");

return 0;

}

$ gcc -g -o ex2 ex2.c ==> compile with -g to use gdb

$ gdb ex2

b main

r

     x=open("swvader03.wav",...);

n                             ==> run "x=open(...)"

     x=read(x, chunkID, 4);     ==> next statement to debug

p x                            ==> print x to see the result of "x=open(...)"

$1=7                          ==> swvader03.wav file is now file no 7

n                              ==> run "x=read(x, chunkID, 4)"

   chunkID[y]=0               ==> next statement to debug

p chunkID       ==> print chunkID to see the result of "x=read(x, chunkID, 4)"

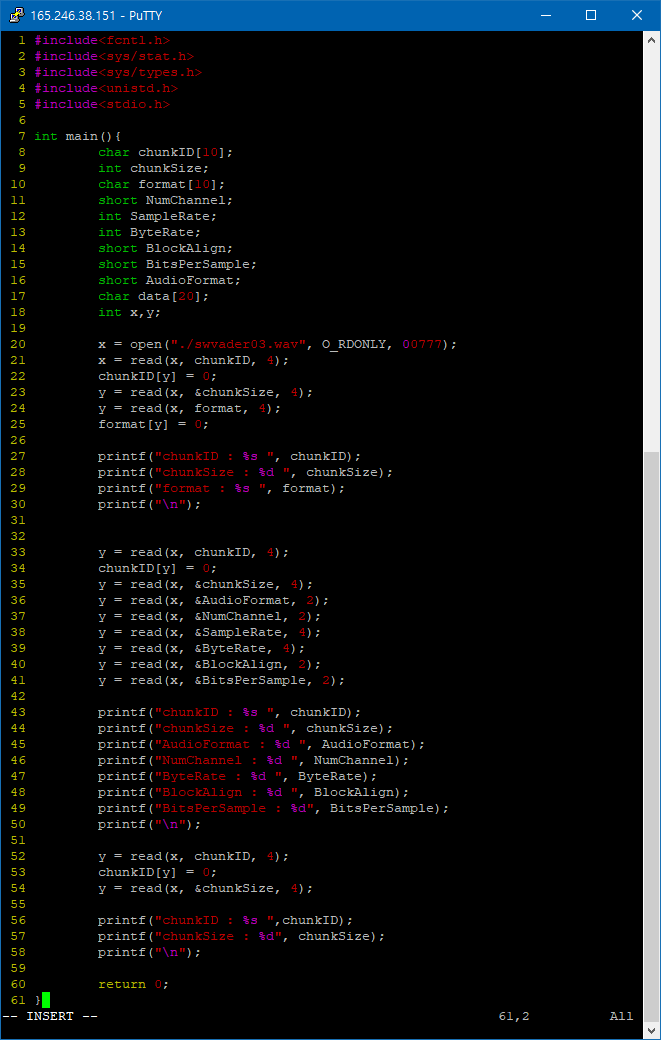
$5="RIFF\000..."              ==> we have RIFF in chunkID

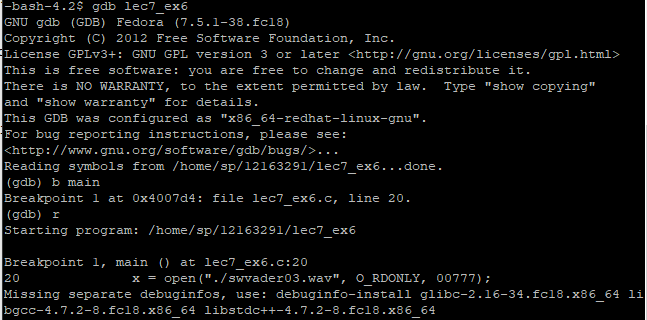
n                             ==> run "chunkID[y]=0"

   y=read(x, ...);              ==> next statement to debug

p chunkID                    ==> check chunkID again after "chunkID[y]=0"

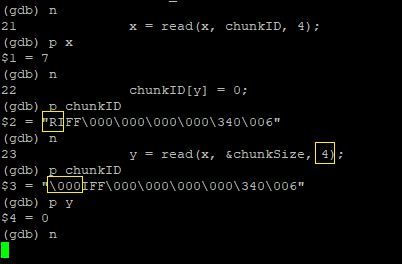
......................





-g 옵션 컴파일 + gdb 커맨드를 통해

디버깅을 실행해 보았습니다.



y=read 를 통해 x 에 4바이트 만큼 chunksize를 read한 이후, 첫부분이

R -> \000으로 변경 되고, 프로그램이 진행이되지 않는 것을 확인할 수 있었습니다