



김현수 (hscornelia) 수정

팔로우 중인 프로필 전체 프로필 보기

[돌소리](#)
[알립니다](#)
[구인구직란](#)
[18학번 모임](#)
[버그 제보 및 기능 제안](#)
[김현수](#)
[족보](#)

개인정보 처리방침

[2016-2] 시스템프로그래밍(에거 교수님) 중간고사 족보

5년 전



정모두하리

실제 문제를 보면 전부 영어로 되어 있지만, 그 표현이 기억이 나질 않아서 한글로 작성합니다. 그리고 기억이 잘 안나는 부분이 꽤 있어서, 틀렸거나 추가해야 할 부분 지적해주시는 거 환영합니다.

필기 (90 min, 90점 만점, 답은 영어도 쓸 수 있고 한글도 쓸 수 있음)

1. 객관식 16문제 (개당0.5점, 총 8점)

[2015-2] 기출과 거의 같음(cheating sheet만 잘 써가도 대부분은 맞을 수 있을 것이다.)

2. Linking and Loading(18점)

a) ELF에 포함되는 파일 형식 3가지를 쓰시오.(4점)

- Executable Object File(따로 형식 지정자 없음), Relocatable Object File(.o), Shared Object File(.so)

b) ELF의 fullname과 의미를 쓰시오.(6점)

- Executable and Linkable Format 의 약자, Standard binary format for object files

c) ELF format을 구성하고 있는 section 중 5가지를 쓰고 각 section에 무엇이 있는지도 쓰시오.(8점)

- Elf header

Word size, byte ordering, file type (.o, exec, .so), machine type, etc.

- Segment header table

Page size, virtual addresses memory segments(sections), segment sizes.

- .text section

Code

- .rodata section

Read only data: jump tables, ...

- .data section

Initialized global variables

- .bss section

Uninitialized global variables

“Block Started by Symbol”

“Better Save Space”

Has section header but occupies no space

- .symtabsection

Symbol table

Procedure and static variable names

Section names and locations

- .rel.text section

Relocation info for .text section

Addresses of instructions that will need to be modified in the executable

Instructions for modifying.

- .rel.data section

Relocation info for .data section

Addresses of pointer data that will need to be modified in the merged executable

- .debug section

Info for symbolic debugging (gcc-g)

- Section header table

Offsets and sizes of each section

이 중에서 5가지 골라 쓰면 됨.

3. Exceptional Control Flow - Signal Handling(6점)

```
pid_t pid;
```

```
void f(void)
```

```
{
    printf("{A}\n");
}

void g(void)
{
    printf("{B}\n");
}

void h(int sig)
{
    printf("{C}\n");
    kill(pid, SIGUSR1);
}

void i(int sig)
{
    printf("{D}\n");
    exit (0);
}

void main() {
    signal(SIGUSR1, h);
    atexit(g);

    printf("{E}\n");

    if ((pid = fork()) == 0) {
        signal(SIGUSR1, i);
        kill(getppid(), SIGUSR1);
        while (1) {};
    } else {
        atexit(f);
        waitpid(pid, NULL, 0);
    }
    printf("{F}\n");
}
```

위 소스코드의 실행결과를 쓰시오.(사실상 2015-2 기출에서 1,2,3,4,5,6 -> A,B,C,D,E,F로 바꾼거)

- ECDAFBA(모르겠으면 직접 실행해보자.)

4. Symbol Resolution & Relocation(20점)

보기로 foo.c와 bar.c라는 소스코드가 주어져 있으며, symbol이 여러개 정의되어 있었다. (foo, bar, foobar 등,

이름이 헛갈려서 기억이 제대로 안남)

a) Symbol table 표 채우기(8점)

채워야 할 정보는 symbol 이름, symbol type, weak/strong, section, Remarks(symbol 에 대하여 추가적으로 쓰고 싶은 것을 쓰라 하셨다.)

b) Symbol Resolution이 일어난 정보의 표 채우기(8점)

총 8개의 Symbol Resolution이 있었던거 같다.

채워야 할 정보는 Symbol Resolution이 일어난 코드상 위치, Symbol Resolution으로 인해 바뀐 Symbol Reference의 위치, 그로 인해 바뀐 변수의 type(?)이었나

c) 표가 하나 주어졌다. 위 소스코드의 ELF중 일부를 보여주고 Symbol Resolution이 일어난 곳에 Instruction에서 그 symbol의 주소와 관련된 부분에 빈칸을 뚫고 거기 들어갈 내용을 맞추는 거였다.(4점)

- 총 2개가 주어졌는데, 하나는 Absolute address를 사용했고, 다른 하나는 PC-relative address를 사용함.(하나는 직접 주소값 쓰면 되고, 나머지 하나는 PC register와 offset으로 표현하면 된듯)

5. Linking and Loading(8점)

one.c two.c three.c four.c라는 소스 코드가 주어짐

one.c

```
int a[1000000]={0};
```

```
int main()
```

```
{
    a[0] = 1;
}
```

two.c

```
int a[1000000];
```

```
int main()
```

```
{
    a[0] = 2;
}
```

three.c

```
int main()
```

```
{
    int a[1000000]={1};
}
```

```
four.c
int main()
{
    int *A=calloc(1000000,sizeof(int));
}
```

a) 각 소스코드를 실행시키면 a는 어디에 저장되어 있는가?(4점)

one -> .data (global initialized)
 two -> .bss (global uninitialized)
 three -> stack (local)
 four -> heap (malloc,calloc,realloc)

b) 각 소스코드로 부터 만들어진 ELF파일의 크기와 프로그램 실행속도를 비교하고 그 이유를 서술하시오.(크기 나 속도가 비슷한 경우 ~를 써도 됨)(4점)

- ELF크기: one > three ~ four > two (이유는 one(.data)는 저 배열이 통째로 들어가고, two(.bss)는 section header는 있는데 크기를 안 먹는다고 했고, three, four는 Linker에서 Library 함수를 추가해줘서 그런듯)

- 실행속도: one > three ~ four > two or one > three ~ four ~ two (이유는 아마 프로그램 크기가 크면 Loading 속도가 느려지니까 one은 그럴거 같고, three, four, two는 뭐 때문이었는지 기억이 안난다(ㅈㅅ).)

(저 1000000이라는 숫자는 변수를 넣어놓고 #define을 썼던거 같다.)

6. Virtual Memory Management + Exceptional Control Flow (20점)

CPU가 page 참조를 위해 MMU에게 요청을 했더니 알고보니 그 page는 physical memory에 없고 하드디스크 내에 있었다(Page Fault). 이때, 일어나는 모든 일을 최대한 자세하게 적으시오(Page fault가 일어나는 과정부터 Page fault Handling이 끝날 때까지 모든 과정을 전부 적으면 됨).

- 답은 너무 길어서 생략. 이번 필기 시험중에서 가장 답이 길었다. 한 페이지 통째로 답을 쓴거 같음.

7. Dynamic Memory Allocation(10점)

a) Memory Allocator의 Free List 구현 방식중 Implicit Free List와 Explicit Free List의 차이점을 설명하시오. (2점)

- Implicit Free List는 Block size만을 가지고 List를 형성하므로 모든 Block이 List에 포함되지만, Explicit은 Free block 내부의 payload중 일부를 다음 free block과 이전 free block의 주소를 담는데 사용하므로, 실제 free인 block만으로 List가 만들어진다.

b) Implicit Free List와 Explicit Free List의 장단점을 비교하시오.(4점)

- 장점: Implicit Free List는 구현이 단순하고, Explicit Free List는 성능(처리속도)가 빠르다.

- 단점: Implicit Free List는 성능이 느리고, Explicit Free List는 구현이 복잡함(이게 맞을려나?)

c) Garbage Collector 중 Mark and Sweep 방식을 설명하시오.(4점)

- Mark: 주어진 root block에서 접근이 가능한 모든 block에 mark-bit을 set한다.

- Sweep: List 전체를 훑으면서 mark-bit이 set되지 않은 block은 모두 free 시킨다.

실기 (90 min, 100점 만점)

1. Stack Buffer Overflow(buflab 문제)

한 프로그램과 그것의 소스코드가 주어졌다. 이 프로그램은 문자열 입력을 받아서 그 문자열이 secret password와 일치하면 good_password(정확히 기억이 안남)이라는 함수를 부르고, 그렇지 않으면 bad_password를 불렀던 것 같다.

a) 이 프로그램은 문자열을 받을 때 취약점이 있다. 이를 잘 이용해서 적절한 문자열을 입력하여 Buffer Overflow를 일으켜 password가 일치하지 않아도 good_password에 접근하게 만들어라.(25점)

b) a)에서 사용한 문자열은 good_password에 접근한 뒤 Segmentation Fault를 일으킨다. 이번에는 Segmentation Fault 없이 정상적으로 종료하도록 문자열을 입력하라.(15점)

c) secret password는 무엇인가?(10점)

2. Library Interpositioning(Linklab 문제)

한국식 로또(6/45) 프로그램의 소스코드와 학생들이 Library Interpositioning에 사용할 빈 소스코드가 주어졌다. 또한, 환경변수를 읽어오는 함수도 힌트로 같이 주어져서 그대로 활용할 수 있었다

a) Library Interpositioning을 사용하여 원래 랜덤한 결과가 나와야 하는 로또 결과를 자신이 정해진 숫자만 나오게 만들으시오(20점)

b) 환경변수를 설정해 둔 경우에만 그 값을 씨드로 하는 srand, 그렇지 않은 경우엔 유저가 넘긴 시드 값으로 srand 함수를 구현하는 문제(20점)

c) printf 함수를 가로채서 대문자는 소문자로, 소문자는 대문자로 바꾸는 함수를 만들라는 문제(10점)

p.s. 이 과목은 기출이 있어도 노답인듯(특히 실기... 1-A 10분 내외로 풀고 1-B에서 80분 다씀;;;). 그리고 더 늦으면 기억이 진짜 안 날 것 같아서 빨리 기억나는 대로 적었다. (2016-11-08 작성)

p.p.s. 시험 성적 분포가 아직도 나오지 않았다.(10/27에 봤음)(나오면 올리겠습니다). 또, 시험 보신 분들은 아시겠지만, 이 과목은 두 그룹으로 나뉘어서 시험치는데 한 그룹은 필기 먼저보고, 실기를 나중에 보고, 다른 그룹은 그 반대로 한다. 또, 두 그룹은 시험중에는 만날 수 없음.

소감: I am Eggered!

추천(0)


▶

시스템프로그래밍 ×

태그추가

└

7개 더 보기



박찬양

4년 전

감사합니다 ㅎㅎ

추천(0)

답글

확인