System Programming
숙제 6

인천대학교

시스템 프로그래밍 | 지도교수 박문주

송병준 | 201701562

0.목차

System Programming

숙제 6

송병준
2019년 10월 28일

1. 과제

Computer System, A Programmer's Perspective 3장 연습 문제 풀이
- 3.64
- 3.67
- 3.69

2. 문제 풀이

3.64

A.
3차원으로 확장된 식은 아래와 같이 주어진다.

$$\&A[i][j][k] = x_A + L(S \cdot T \cdot i + T \cdot j + k)$$

테스트를 해보자.

```
TEST(nested_array_test) {
#define R 24
#define S 35
#define T 12
#define ADDR3D(X, Y, Z) (S*T*X + T*Y + Z)

      long A[R][S][T];
      ASSERT(&A[1][2][3] == ((long *)A + ADDR3D(1, 2, 3)));

      return true;
}
```

```
==== Test 1 Started ====
Assertion succeeded: &A[1][2][3] == ((long *)A + ADDR3D(1, 2, 3))
==== Test 1 Succeeded ====
```

잘 된다.

B.

주어진 어셈블리의 마지막 %eax로의 대입문을 통해 $R \cdot S \cdot T \cdot sizeof(long)$이 3640임을 알 수 있다. 이 문제에서 long 형의 크기는 8이므로 $R \cdot S \cdot T$는 455이다.

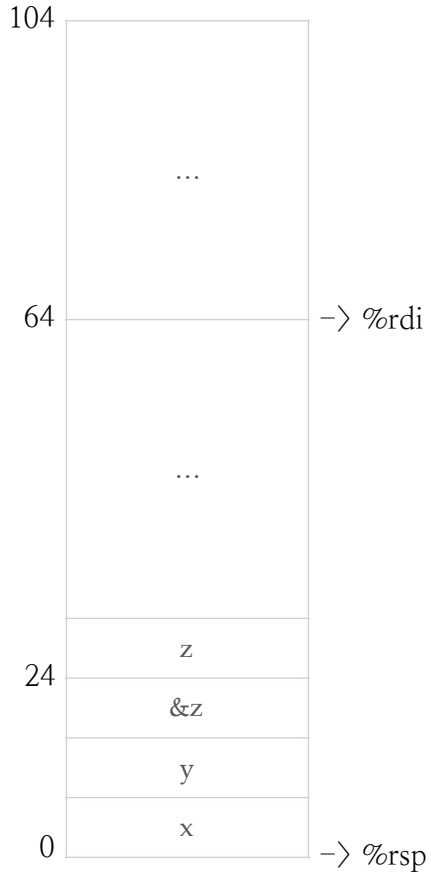
처음 두 줄을 실행한 뒤에 %rax에는 13j의 값이 담긴다. 그 후에 i와 64i, k가 더해져 선형 주소는 (65i + 13j + k) * 8의 값을 가진다.

이때,
$65i + 13j + k = S \cdot T \cdot i + T \cdot j + k$이므로
$S \cdot T = 65$ 이고 $T = 13$이다.
$S = (S \cdot T)/T = 65/13 = 5$이다.
$R = (R \cdot S \cdot T)/(S \cdot T)$이므로 $R = 455/65 = 7$이다.

따라서
$R = 7$
$S = 5$
$T = 13$
이다.

3.67

A.



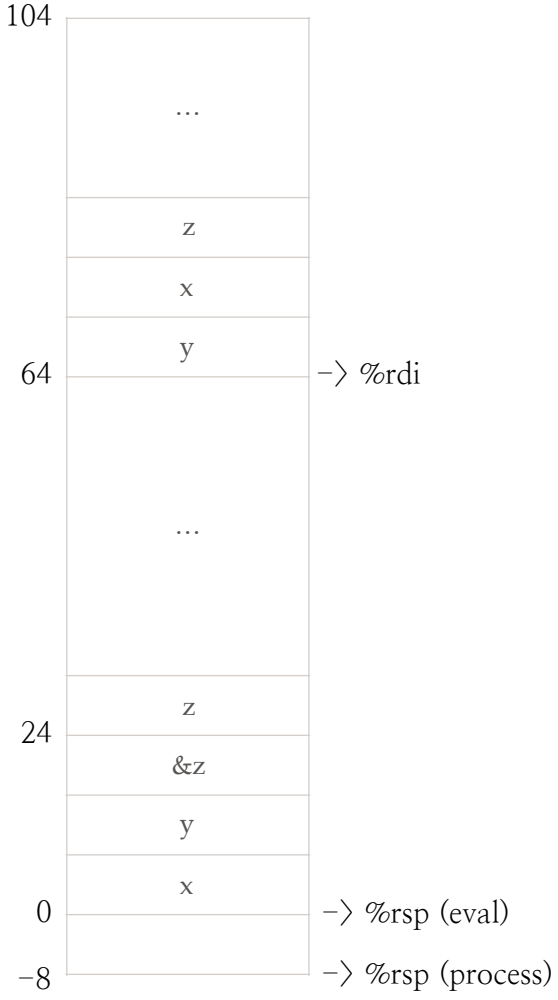
B.

process를 호출할 때에 eval은 구조체를 위해 할당된 스택 공간의 주소를 전달한다.

C.
process는 넘겨받은 인자가 아닌 caller의 스택에 직접 접근하여 인자 s의 원소에 접근한다. 단, process 함수가 시작되는 순간 새로운 프레임이 만들어지고 스택의 8 바이트가 점유되는 것을 감안한다.

D.
process는 인자로 넘겨받은 %rdi를 통해 구조체 r의 필드에 기록하는데, 이때 %rdi는 caller 스택 top으로부터 64번지만큼 떨어진 곳이다. 이곳은 caller가 process를 호출하기 전에 미리 할당한 104바이트의 일부이다.

E.



F.
메모리 관리를 caller가 담당한다. Caller는 callee가 사용할 구조체를 위한 공간까지 스택에 할당한 다음, callee에게 그 공간의 주소를 넘겨준다.

3.69

A.
어셈블리의 첫 두 줄을 통해 a 필드의 크기는 280 바이트라는 것을 유추할 수 있다. CNT를 알기 위해서는 a_struct의 크기를 알아야 한다.

인자로 넘어온 bp로부터 필드인 a 배열의 i번째 원소에 접근하기 위해 $bp + 8 + 40i$의 주소를 사용한다. 이때 i의 계수인 40이 a_struct의 크기이다. 따라서 $CNT = 280/40 = 7$이다.

B.
위에서 $bp + 8 + 40i$가 a 배열의 i번째 원소를 가리킨다고 하였다. 이때 +8은 배열에서 찾아낸 a_struct 내부의 필드를 참조하기 위한 오프셋일 수 있으며, 단지 a 배열이 b_struct 구조체 시작점에서 8바이트 떨어져있기 때문일 수도 있다.

b_struct 구조체 선언은 64비트 환경에서 첫번째 필드와 두번째 필드 사이에 4바이트의 padding을 유발한다. 그렇다면 +8 오프셋은 padding 4 바이트와 첫번째 필드인 4 바이트짜리 first를 넘어 a 배열에 접근하기 위한 것임을 알 수 있다.

위에서 얻은 $bp + 8 + 40i$가 가리키는 값은 별도의 참조 없이 바로 idx의 값으로 쓰인다. 따라서 b_struct의 첫번째 필드는 idx임을 알 수 있다.

그 다음에는 a_struct의 x 필드에 대한 힌트가 등장하는데, 정수 배열인 x 필드에 접근할 때에는 +16의 오프셋이 붙는다. 이중 +8은 앞서 언급한 int 4 바이트와 padding 4 바이트이며, 나머지 +8은 a_struct 내에서 idx 다음에 등장할 x에 접근하기 위한 것이다.

x 배열에 접근할 때에는 8 바이트 단위로 접근한다. 즉 x 배열의 원소는 8바이트 부호형 정수 long이다. 그렇다면 idx는?

a_struct의 크기가 40이 되려면 양의 정수 k에 대해 $size_{idx} = 40 - 8k = 8(5 - k)$이어야 한다. 즉 idx의 크기는 8의 배수이다. long 형이다.

따라서 a_struct의 완전한 선언문은 다음과 같다.

```
typedef struct {
    long idx;
    long x[4];
} a_struct;
```