strcpy( ) 구현

입력 : 한 줄의 문자열
출력 : bss 섹션 데이터에 문자열 저장하기

bss 섹션에 64bit의 크기를 갖는 데이터를 정의했다는 점 빼고는 strlen( ), strcat( )과 크게 다르지 않다.

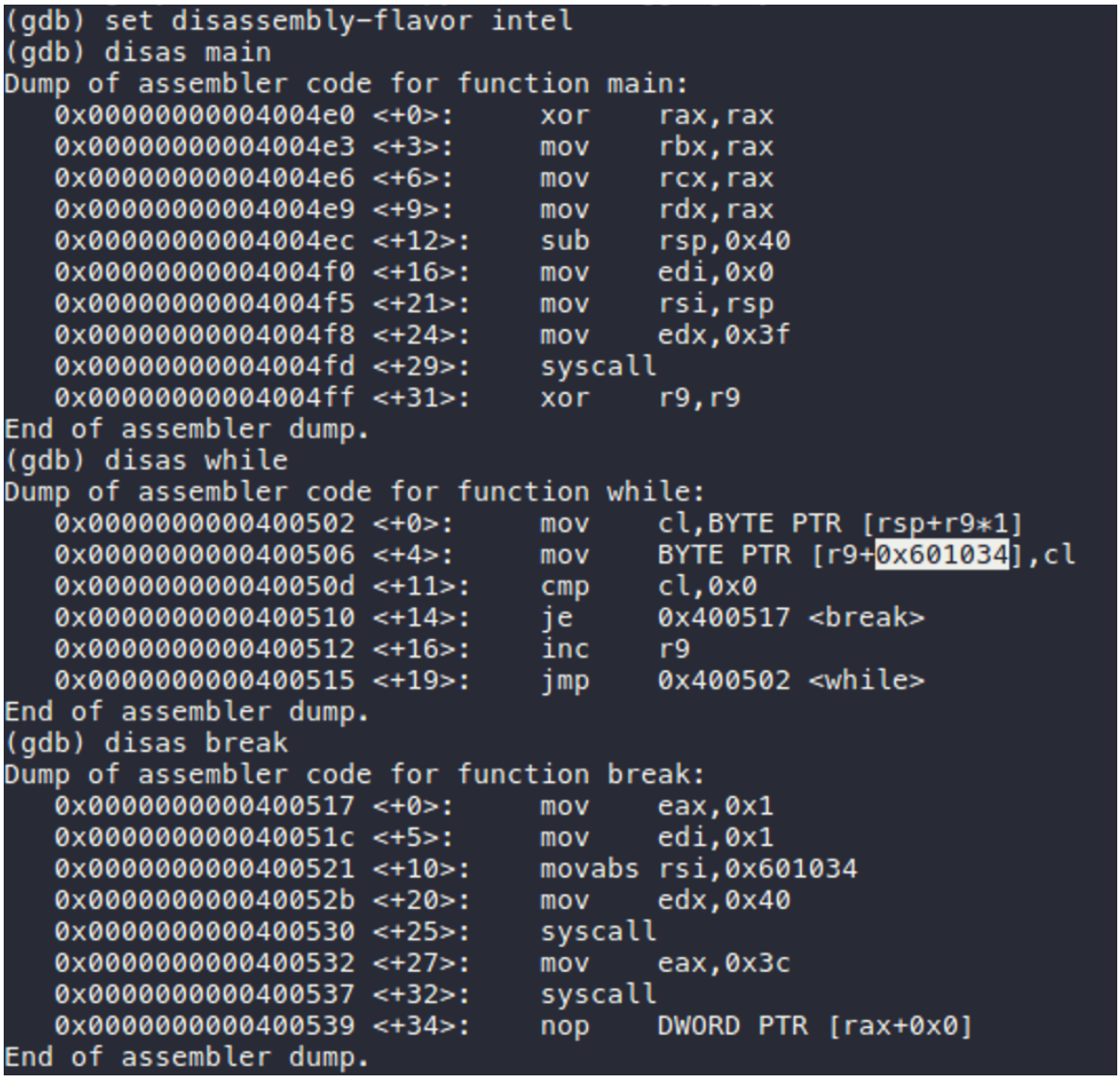
```
;line 22 - 30
xor r9, r9
while:
  mov cl, [rsp + r9]
  mov [str1 + r9], cl
  cmp cl, 0x00
  je break
  inc r9
```

```
    jmp while
break:
```

문자열을 한 문자씩 cl 레지스터에 받아와 그걸 그대로 str1에 저장한다. NULL이 나올 때까지 반복.

str1에 입력된 문자열이 잘 들어갔는지 확인하기 위해, gdb로 디버깅 해 보았다.

```
$ nasm -f elf64 strcpy.asm
$ gcc -g -o strcpy strcpy.o
$ gdb strcpy
```



작성했던 코드들을 확인할 수 있다. 또한, while 안에서 내가 str1으로 정의했던 메모리 위치 0×601034를 확인 할 수 있었다.

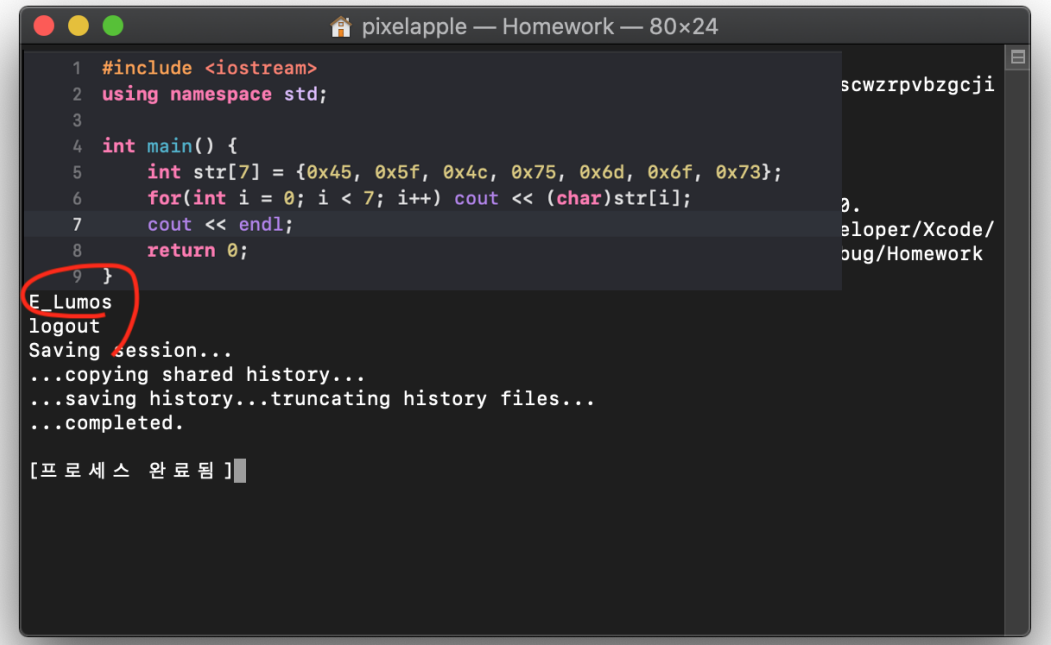
```
(gdb) b break
Breakpoint 1 at 0x400517
(gdb) r
Starting program: /home/e_lumos/Desktop/Assembly/strcpy_f/strcpy
E_Lumos

Breakpoint 1, 0x0000000000400517 in break ()
```

따라서 문자열 복사가 완료되는 시점인 break에 breakpoint를 잡고 실행시켜, 문자열이 모두 str1에 들어갈 수 있게 실행시켰다.

```
(gdb) x/7bx 0x601034
0x601034:       0x45    0x5f    0x4c    0x75    0x6d    0x6f    0x73
```

내가 넣어준 문자는 E_Lumos로 7바이트. str1의 메모리 0×601034의 값을 1 byte씩 7 byte만큼, 16진수로 출력시켜 보았다.

이 값을 순서대로 아스키 코드로 바꾸면,



E_Lumos가 됨을 확인할 수 있다.