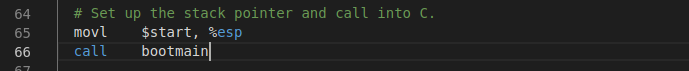
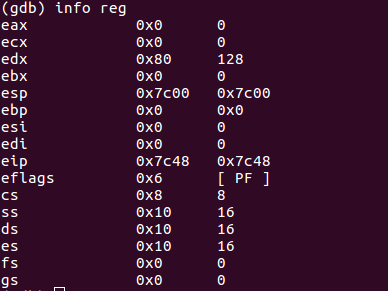
bootasm.S의 어느 부분에서 stack pointer가 초기화 되는 부분?



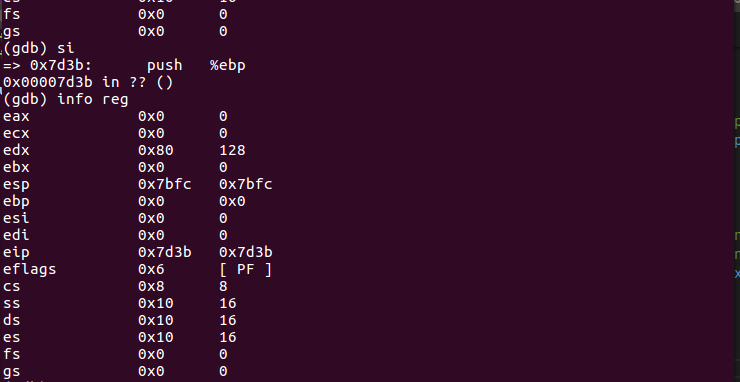


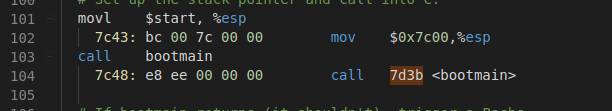
따라가다 보면 bootblock의 시작점 7c00에서 stack pointer가 초기화된다.

Bootmain의 코드를 한 단계씩 수행한다; 현재 stack에는 무엇이 있는가?

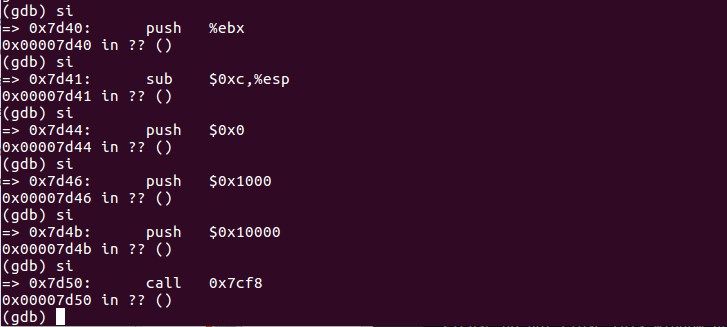


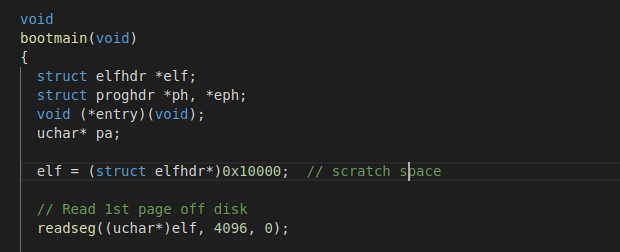
Esp 현재 스택 top >> bootblock



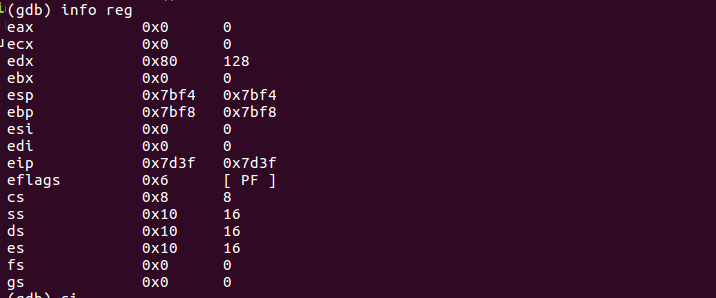


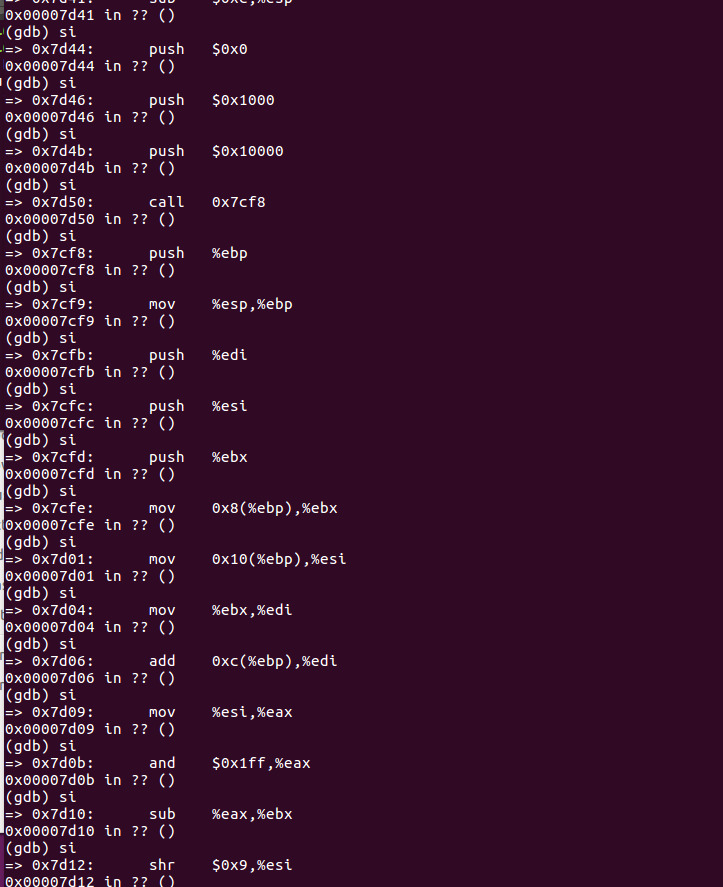
Eip가 bootmain을 call을 가리킴

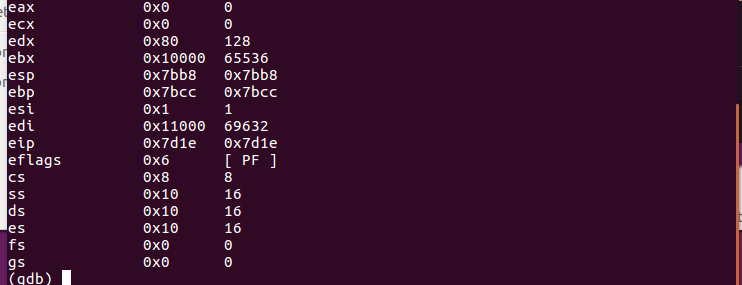




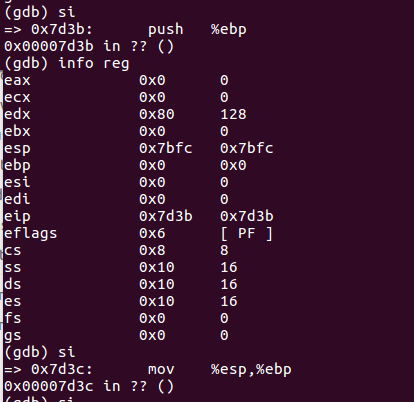
bootmain에서 readseg에 인자를 넣음







Bootmain의 첫 번째 assembly 명령어들이 stack에서 하는 일은 무엇인가?

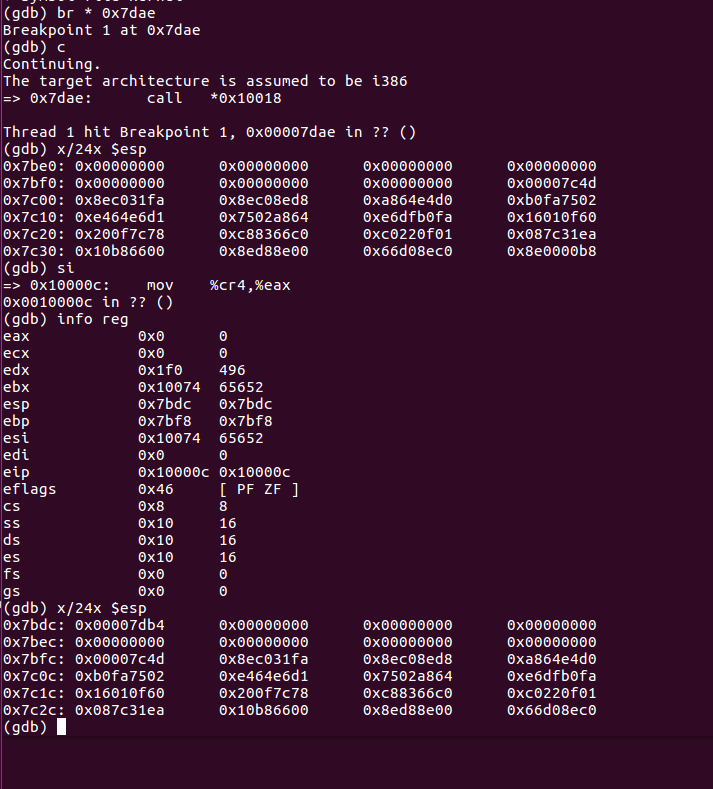


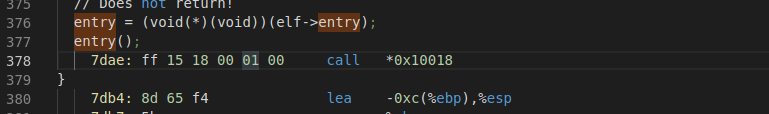
7d3b 즉 bootmain을 실행하면서 첫번째 명령어로 Mov %esp,%ebp

>> Ebp 값을 esp로, 즉 0으로 만들면서 stack pointer를 초기화 한다.

gdb를 (필요하다면 breakpoints를 사용) 통해 추적하고 eip를 0x10000c로

변경하는 코드를 탐색한다. 그 코드가 stack에서 하는 일은 무엇인가?





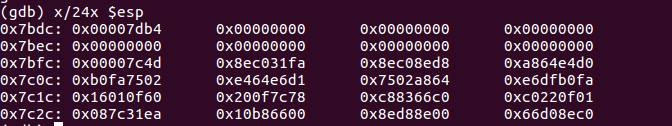
Entry point를 불러낸다. 불러내고 나서 entry 함수 실행 다음으로 갈부분이 7db4인데 그부분의 주소를 스택에 저장한다.

0x00000000 >> 0x00007db4로 바뀌게됨

결론)

gdb에서 x/24x $esp 명령어 통해 출력된 결과와 출력된 스택 영역 데이터의

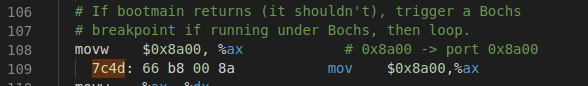
의미를 기술하여 제출



1-2째줄이 실제 스택 영역이다.(8번째까지) 위의 info reg를 참고하면 0x7bdc(esp)에 entry call이후 들어갈 주소가 저장 되어있음. 함수실행이전 ebp (0x00000000)

Si 즉 어셈블리어 명령들 push, sub, mov 거치며 스택에 올라간 레지스터의 메모리 주소들이다.

Bootmain 실행할 당시의 ebp, edi, esi, ebx 값이 저장 되어 있다. 0x00000000



0x7bfc에서의 0x7c4d의 의미 bootmain return시 trigger 하는 것(스택영역 X)