Cyber Forensics Quiz2

tags: `class`

1.

```
0000041d <foo>: // stdcall
41d: 55 push %ebp
41e: 89 e5 mov %esp,%ebp
420: e8 0e 00 00 00 call 433 <__x86.get_pc_thunk.ax>
425: 05 db 1b 00 00 add $0x1bdb,%eax
42a: b8 00 04 00 00 mov $0x400,%eax
42f: 5d pop %ebp
430: c2 04 00 ret $0x4

0000041d <foo>: // cdecl
41d: 55 push %ebp
41e: 89 e5 mov %esp,%ebp
420: e8 0c 00 00 00 call 431 <__x86.get_pc_thunk.ax>
425: 05 db 1b 00 00 add $0x1bdb,%eax
42a: b8 00 04 00 00 mov $0x400,%eax
42f: 5d pop %ebp
430: c3 ret
```

In cdecl, the caller has to cleaup the stack, hence c3 is used to return, whereas in stdcall, the callee is responsible for cleaning up the stack, hence the c2 op code.

2.

Assuming the main procesure is compiled with cdecl, the caller(main) will clean up the arguments for calling `foo`, so if we use `foo` defined in libstdcall, which will also clean up the arguments as the callee because it's compiled with stdcall, segmentation fault may occur.

3.

Because the problem lies in the stack pointer, an error is more likely to occur in code block 2 where a function is invoked.