


whyask37's blog

mud lecture

[Rude Lecture] 3. Pointer

 why do you ask  
2014. 1. 21. 0:46

add neighbor

The size of the course is as much as I like.

-----

Let's learn pointers.  
In fact, it's easy to poke the pointer.  
By the way, the application can be very interesting.

It's a rough idea  
If you learn how to create a .trg in Lesson 4,  
I think I can write an incredibly long 5 rivers.  
I want to loosen the limit on the number of units in the Round of 5.  
I wonder if there are already example maps out there.

-----

Let's do something fun to figure out what a pointer is.  
Go to page 2 of EUDDB ( <http://farty1billion.dyndns.org/EUDDB/?pg=index&st=2> )  
There is something like this.

|          |            |         |                                     |       |  |
|----------|------------|---------|-------------------------------------|-------|--|
| 00596A18 | 1.1<br>6.1 | Wi<br>n | Virtual Key Array                   | On 25 | An array containing the state of all the virtual key codes ... |
| 005993D4 | 1.1<br>6.1 | Wi<br>n | <b>Pointer</b> to map "STR" section | 4     | On Pointer default: first string                               |
| 0059CC80 | 1.1<br>6.1 | Wi<br>n | Drop Timer                          | 4     | On The amount of time you have to wait to drop someone.        |

Now let's look at the meaning of pointers. The definition of a pointer itself is very simple.

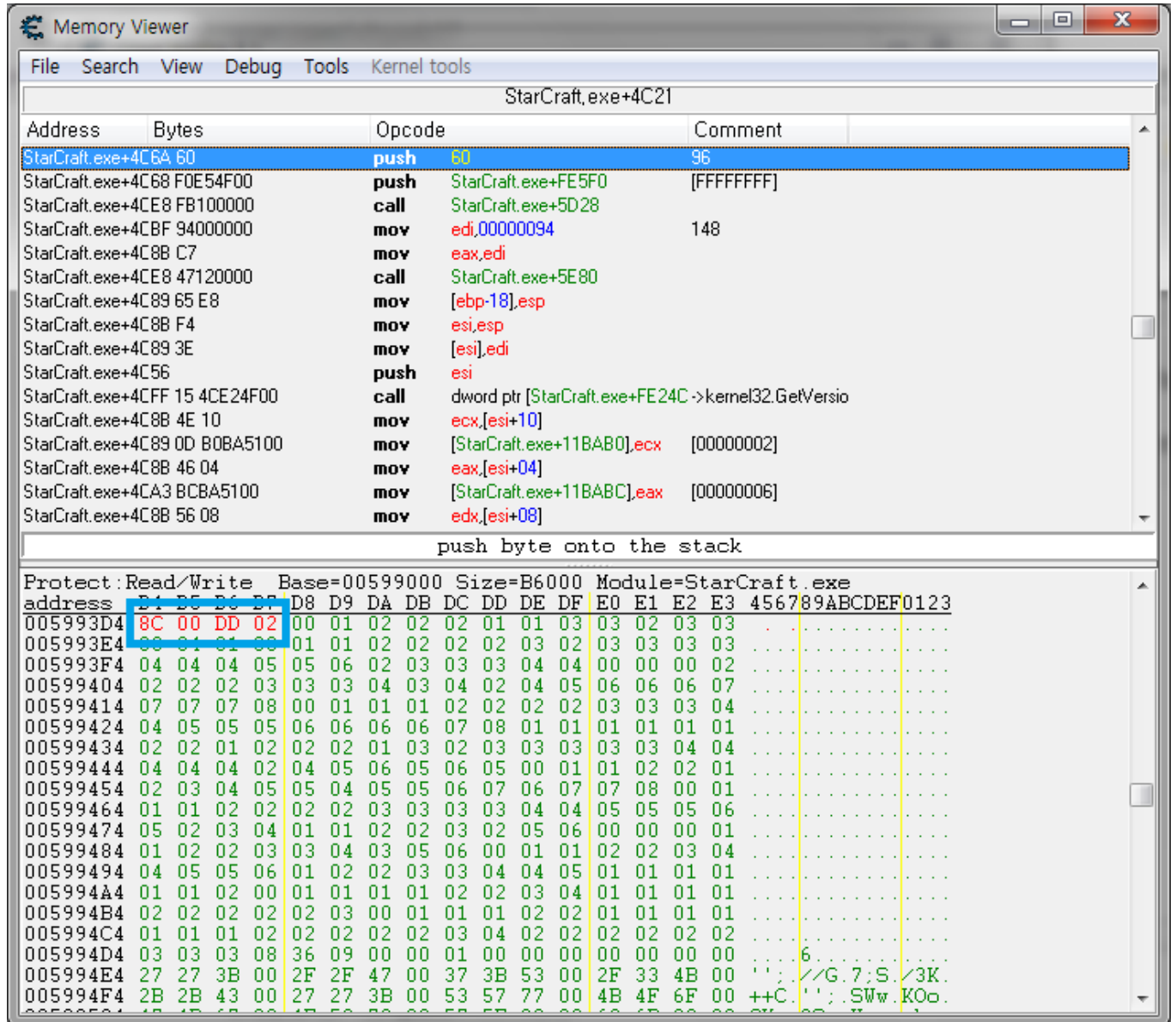
A variable that has an address as a value is called a pointer. (A pointer in a star is 4 bytes in size)

what do you mean by this

## whyask37's blog

guys call anything.

3. And if you look at address 005993D4 through Memory View, it looks like this.



It says 02DD008C. (Refer to Lesson 1: read in reverse order on Intel CPUs)

4. Let's read this address through the memory view. Then it comes out like this.

## whyask37's blog

| Address                        | Bytes | Opcode  | Comment    |
|--------------------------------|-------|---|------------|
| StarCraft.exe+4C6A 60          |       | push 60   | 96         |
| StarCraft.exe+4C68 F0E54F00    |       | push StarCraft.exe+FE5F0                                | [FFFFFFFF] |
| StarCraft.exe+4CE8 FB100000    |       | call StarCraft.exe+5D28                                 |            |
| StarCraft.exe+4CBF 94000000    |       | mov edi,00000094  | 148        |
| StarCraft.exe+4C8B C7          |       | mov eax,edi   |            |
| StarCraft.exe+4CE8 47120000    |       | call StarCraft.exe+5E80                                 |            |
| StarCraft.exe+4C89 65 E8       |       | mov [ebp-18],esp  |            |
| StarCraft.exe+4C8B F4          |       | mov esi,esp   |            |
| StarCraft.exe+4C89 3E          |       | mov [esi],edi   |            |
| StarCraft.exe+4C56             |       | push esi  |            |
| StarCraft.exe+4CFF 15 4CE24F00 |       | call dword ptr [StarCraft.exe+FE24C->kernel32.GetVersio |            |
| StarCraft.exe+4C8B 4E 10       |       | mov ecx,[esi+10]  |            |
| StarCraft.exe+4C89 0D B0BA5100 |       | mov [StarCraft.exe+11BAB0],ecx                          | [00000002] |
| StarCraft.exe+4C8B 46 04       |       | mov eax,[esi+04]  |            |
| StarCraft.exe+4CA3 BCBA5100    |       | mov [StarCraft.exe+11BABC],eax                          | [00000006] |
| StarCraft.exe+4C8B 56 08       |       | mov edx,[esi+08]  |            |

push byte onto the stack

| Protect:Execute/Read/Write | Base=02DD0000  | Size=1000    |
|----------------------------|--|--------------|
| address                    | 8C 8D 8E 8F 90 91 92 93 94 95 96 97 98 99 9A 9B CDEF0123456789AB |              |
| 02DD008C                   | 00 04 03 08 0C 08 15 08 1E 08 2A 08 7D 08 7F 08                  | .....*..}..! |
| 02DD009C                   | 81 08 83 08 85 08 30 09 33 09 02 08 02 08 02 08                  | ...0.3.....  |
| 02DD00AC                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD00BC                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD00CC                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD00DC                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD00EC                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD00FC                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD010C                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD011C                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD012C                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD013C                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD014C                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD015C                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD016C                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD017C                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD018C                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD019C                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |
| 02DD01AC                   | 02 08 02 08 02 08 02 08 02 08 02 08 02 08 02 08                  | .....        |

I won't deal with it right now

00 04 03 08 0C 08 ... it's been so long

This is the **STR section** of scenario.chk .

(If you've studied Manual Unfl, you've probably heard of it.)

I sort it out.

005993D4 has a value of 02DD008C.

At address 02DD008C, the STR section is stored.

Therefore, 005993D4 has the address where the STR section is stored as a value.

In this case, 005993D4 is called a pointer to the STR section.

So I write like this.

**Pointer to mem "STR" section**

## whyask37's blog

So how do you use this?

Well, I'll use it someday.

-----

It seems that the course is short.

I feel like it's too little if I do this

A very good example that clearly shows why pointers are used.

Let's talk about doubly linked lists.

(Welcome to Hellgate)

(If you have dealt with structural offsets, you may already know this.

저는 구조오프셋 안다뤘었으니까 그냥 할께요)

-----

이제부터는 C언어 그냥 안다고 가정하고 팍팍 갈께요.

포인터 전까지는 C언어 쉽습니다. 사실 포인터도 쉬운데

포인터 단원 들어가면서 갑자기 프로그램에서 '오류 보고 보냄'같은게 엄청 뜨기 때문에 포인터가 어려워보이는거지

포인터는 진짜 위에 써있는 저 정의가 전부입니다.

-----

이중 연결 리스트는 스타에서 유닛, 트리거, 스프라이트 등등 온갖걸 관리하는데 쓰입니다.

실전처럼 강의를 하도록 하죠.

전설의 DoA님의 Unitnode structure를 봅시다. 모두 이미 아실법한 곳입니다.

구조오프셋 다룰 때 굉장히 많이 쓰이지요.

<http://farty1billion.dyndns.org/EUDDDB/?pg=ref&a=unitnode>

맨 처음에

CUNIT STRUCT +0x0 - CUNIT\* Previous +0x4 - CUNIT\* Next

라 나와있네요.

CUNIT\*는 CUNIT형 자료형에 대한 포인터라 쓰입니다 (\*가 여기서의 포인터라는 의미)

## whyask37's blog

CUnit \*prev, \*next, 나 쓰겠습니다.

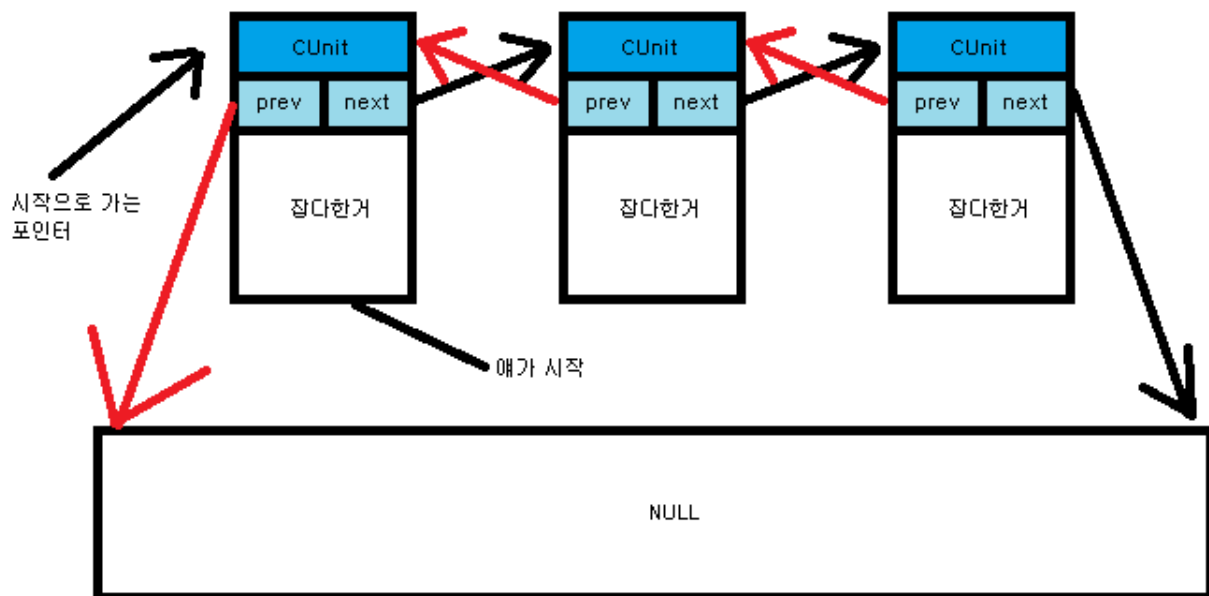
(prev, next라는 CUnit에 대한 포인터가 있어요)

그러니까

```
struct CUnit {
    struct CUnit *prev, *next;
    기타잡다한것
};
```

이런 구조로 되어 있지요.

그림으로 나타내면 다음과 같습니다.



그림판 만세

대충 이렇게 next는 다음 유닛을 가르키는 포인터고  
prev는 이전 유닛을 가르키는 포인터예요.

NULL이라고 이상한게 보일텐데

NULL은 어떻게 어떻게 해서 하여튼간에 **접근할 수 없는 메모리**예요.

CUnit은 NULL에 생길 수가 없어요.

처음 유닛의 prev는 NULL을 가르키고요

마지막 유닛의 next도 NULL을 가르킵니다.

거꾸로 말하면

## whyask37's blog

- 
- 그러니까 next가 NULL인 CUnit은 맨 마지막 유닛이 되는거고
  - prev가 NULL인 CUnit은 맨 처음 유닛이 되는겁니다.

대충 이렇게 이중 연결 리스트입니다.

(prev 없이 그냥 next만 가지고 있는 리스트도 있는데 그걸 단일 연결 리스트라 해요.)

이러면 맨 처음 원소가 어디 있는지만 알고 있으면 (= 맨 처음 원소에 대한 포인터를 가지고 있으면) 모든 유닛들을 접근할 수 있겠죠.

(아 물론 7번째 유닛이 뭐냐라는 질문에 대해서는 쉽게 답하기 힘들지만요. '7번째 유닛이 뭐냐'같은걸 풀려면 배열을 써야합니다.)

사실 이것보다 약간 더 복잡할것같은 한데

그건 나중에 알아볼게요. 아직 CUnit쪽은 저도 공부를 더 해야 하네요.

스타맵 예제 없습니다.

읽어주셔서 감사합니다.

-----

생각해본 내용인데

EUD를 써서 매우 재미있는 일들을 할 수 있을것같아요.

강좌의 단기 목표를 3개 잡겠습니다.

1. memcpy를 스타 트리거로 구현하기. (1, 2, 3강의 결합)
2. 유닛 제한수 5000개로 늘리기
3. 맵에서 이미지 불러와서 스프라이트로 쓰기

2, 1, 3 순서대로 할것같네요.

#IT·컴퓨터

0

0

## whyask37's blog

### 왜물어

whyask37님의 블로그입니다.

이웃추가

#### this blog **mud lecture** Category article

##### [Rock lecture] 5. Pointer example - Reducing unit limit

2014. 1. 23.

0

##### [Rock Lecture] 4. TRG file format

2014. 1. 21.

3

##### [Rude Lecture] 3. Pointer

2014. 1. 21.

0

##### [Rude Lecture] 2. Substitution between death, plus

2014. 1. 19.

One

##### [Lecture] 1. Brief summary of EPD

2014. 1. 19.

0



#### 이 블로그 인기글

0

## whyask37's blog

---

11

---

### 5. SFmpq (ShadowFlare's MPQ Library) 와 예제

2013. 9. 11.

1

---

### [별강의] 13. 트리거 프로그래밍 - TRIG-MRGN 루프

2014. 2. 24.

0

---

### 4. scenario.chk

2013. 9. 10.

0

---

### [별강의] 2. 데스 사이의 대입, 더하기

2014. 1. 19.

1

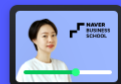
---



---

[back to top](#)

블로그 글쓰기부터 브랜딩까지  
전문가 무료 특강 듣고! 페이도 받고!



[View in PC version](#)

---

0