数组

一维数组定义

```
类型标识符  数组名[元素个数]

// 定义
int n[5];

// 初始化
int n[5] = { 0 };            // 全部初始化为0
int n[5] = { 1 };            // 第一个元素为1，其他全部为0
int n[5] = { 1, 2, 3 };      // 前三个元素为1、2、3，其他全部为0
```

元素个数是个编译期间可以确定的值 —— 常量、常量表达式（C99支持变长数组）
针对于编译期间可以确定的值 —— 常量、常量表达式，编译器可以进行**常量折叠**的优化

```
int main(void)
{
    int n = 6 * 4 - 9 / 2;  // 常量折叠优化后等同于 int n = 20;
    return 0;
}
```

在编译期就自动将其求值，而不给其分配存储空间

连续性和一致性

- 存储空间连续
- 存储的类型一致

下标访问

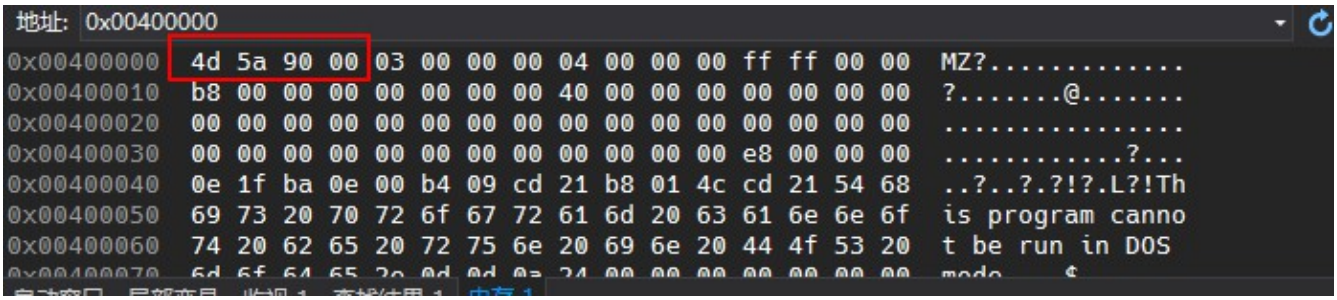
```
// []内是整型常量或者变量
int index = 3;
n[3] = 1;
n[index] = 1;

// 另类不建议用
3[n] = 1;
index[n] = 1;
```

下标访问内部的计算规则：`(int)array_address + sizeof(type) * index`
比如访问**0x00400000**内存的值：

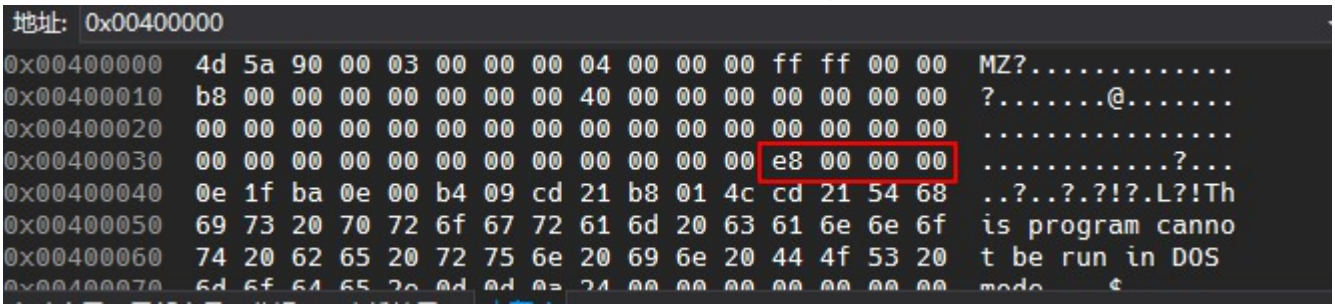
```
int array[ARRAY_SIZE];
array[(0x00400000 - (int)array) / sizeof(int)];
```

定位程序的入口点的地址

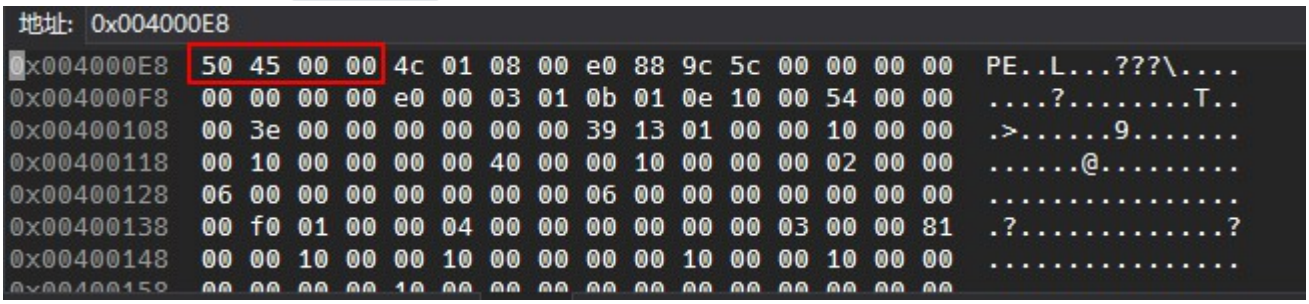
环境使用Visual Studio 2017，关闭随机地址，所以程序加载基址为 0x00400000



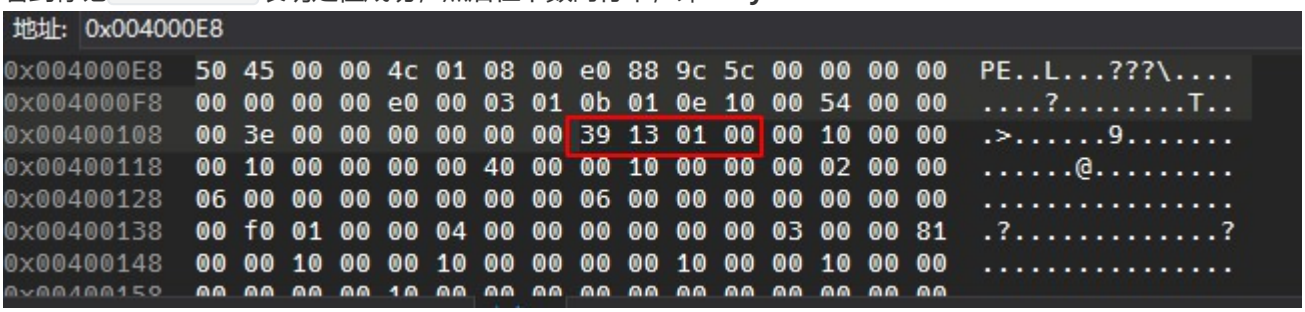
看到标记 0x00905a4d 表明基址定位是正确的，然后从基址这里下数**60Byte**



取这里的值在加上基址得 0x004000e8，在转到这个地址上



看到标记 0x00004550 表明定位成功，然后往下数两行半，即**40Byte**



得到的 0x00011339 在加上基址 0x00400000 得 0x00411339 就是第一条指令处

然后在汇编里转到这个地址验证

这样就定位了程序的入口点

其他

寻路算法

A*，在寻路过程中加入决策值，决策从那边走到终点的距离最近（不一定最优）

算法类函数设计

UI和逻辑分离

伪随机函数srand和rand

经典代码：`srand((unisgned int)time(NULL))`
以时间作为随机种子，时间在增加所以产生的伪随机数可以被预测下一次的趋势——是增加的