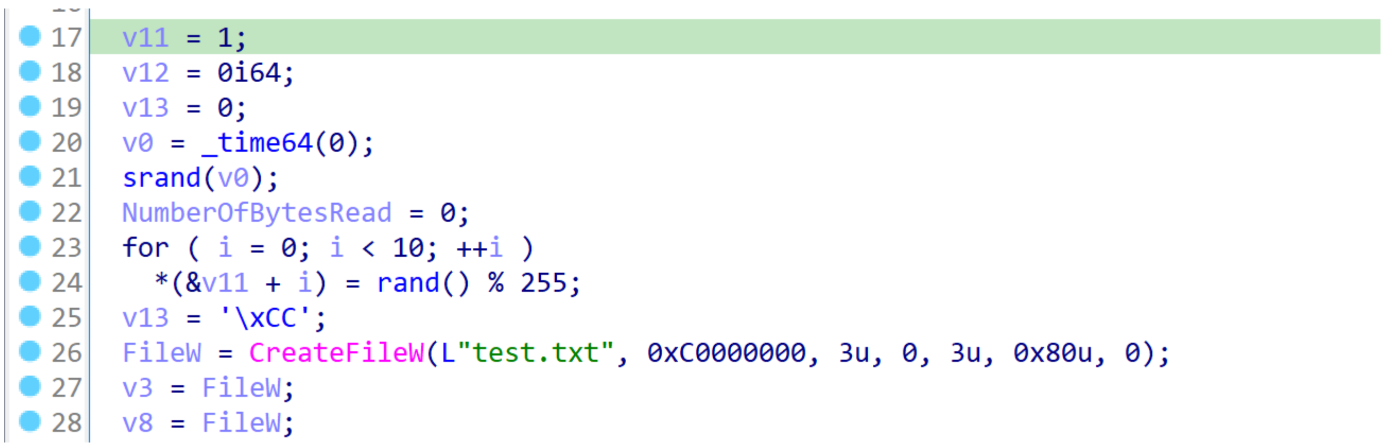
test.exe.dmp

1. exe file analysis

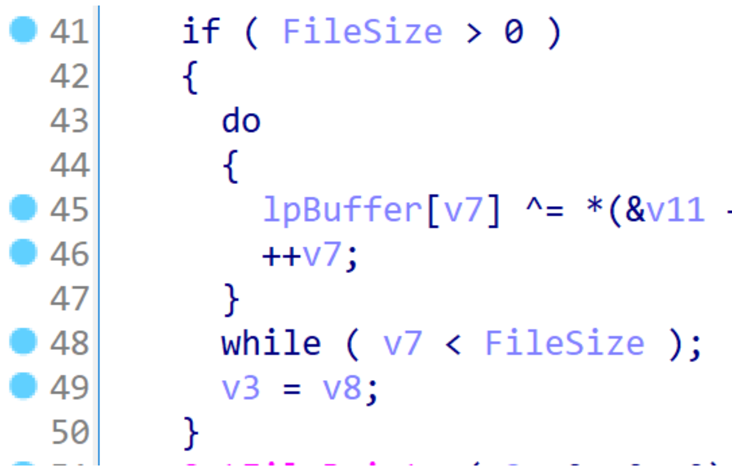


```
v11 = 1;
v12 = 0i64;
v13 = 0;
v0 = _time64(0);
srand(v0);
NumberOfBytesRead = 0;
for ( i = 0; i < 10; ++i )
  *(&v11 + i) = rand() % 255;
v13 = '\xCC';
FileW = CreateFileW(L"test.txt", 0xC0000000, 3u, 0, 3u, 0x80u, 0);
v3 = FileW;
v8 = FileW;
```

First it call `srand()` with current time as seed. Then it generate an array with 10 random values.

The last element is set to `0xCC`

After that it opens `test.txt`, read the content and performs `xor` operation on the read data.



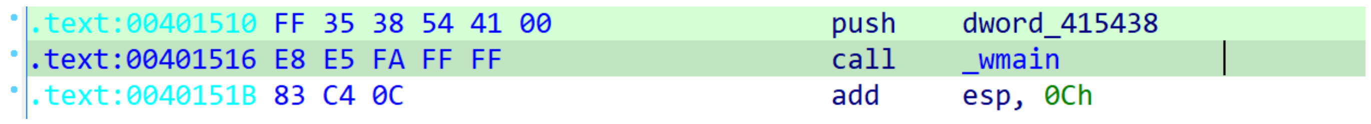
```
    if ( FileSize > 0 )
    {
      do
      {
        lpBuffer[v7] ^= *(&v11 + v7 % 0xAu);
        ++v7;
      }
      while ( v7 < FileSize );
      v3 = v8;
    }
```

2. dmp file analysis

Our goal is to find the array contains 10 random values generated above

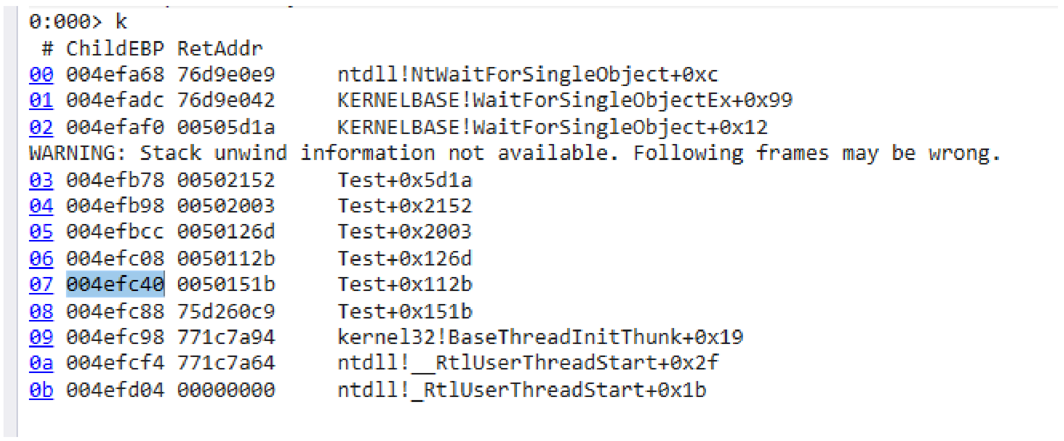
That array is placed on stack, so we have to locate the stack frame of this function.

In ida, our function is `wmain()` and after calling this function, the program return to `0x0040151B`



```
.text:00401510 FF 35 38 54 41 00                push    dword_415438
.text:00401516 E8 E5 FA FF FF                   call    _wmain
.text:0040151B 83 C4 0C                         add     esp, 0Ch
```

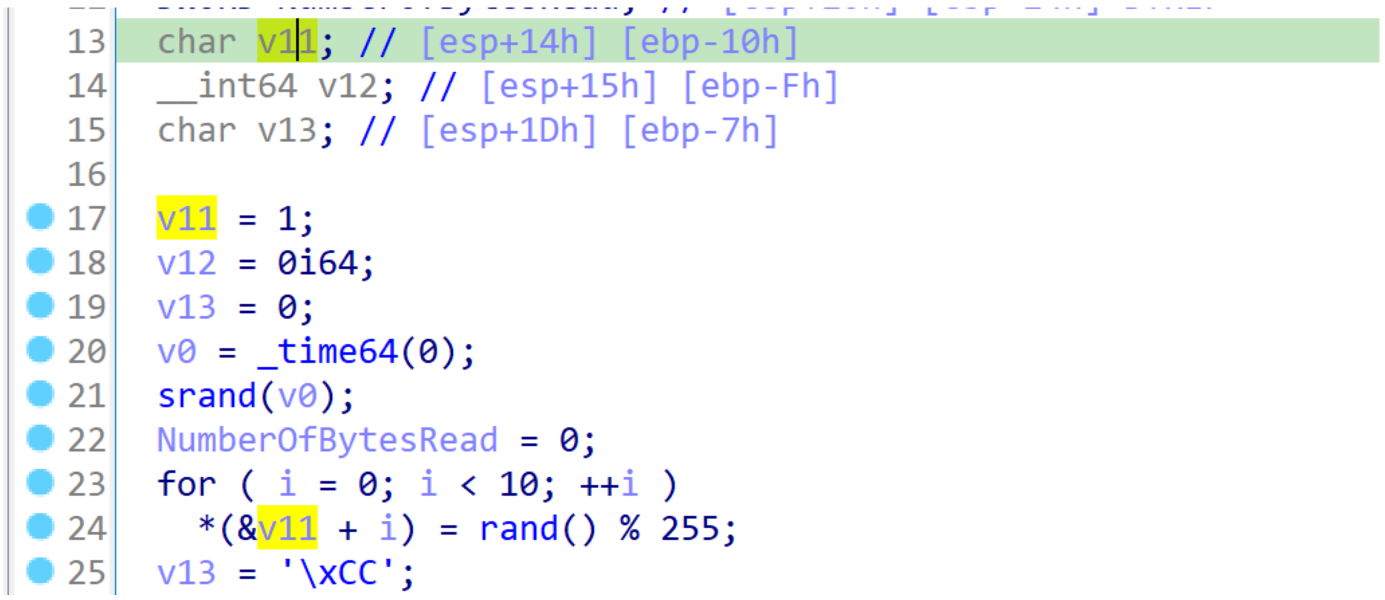
On windbg commandline, press `k` to view stack frame



```
0:000> k
 # ChildEBP RetAddr
00 004efa68 76d9e0e9     ntdll!NtWaitForSingleObject+0xc
01 004efadc 76d9e042     KERNELBASE!WaitForSingleObjectEx+0x99
02 004efaf0 00505d1a     KERNELBASE!WaitForSingleObject+0x12
WARNING: Stack unwind information not available. Following frames may be wrong.
03 004efb78 00502152     Test+0x5d1a
04 004efb98 00502003     Test+0x2152
05 004efbcc 0050126d     Test+0x2003
06 004efc08 0050112b     Test+0x126d
07 004efc40 0050151b     Test+0x112b
08 004efc88 75d260c9     Test+0x151b
09 004efc98 771c7a94     kernel32!BaseThreadInitThunk+0x19
0a 004efcf4 771c7a64     ntdll!__RtlUserThreadStart+0x2f
0b 004efd04 00000000     ntdll!_RtlUserThreadStart+0x1b
```

Here we can see `0x004efc40` is the address which contain `ebp` of `wmain()`.

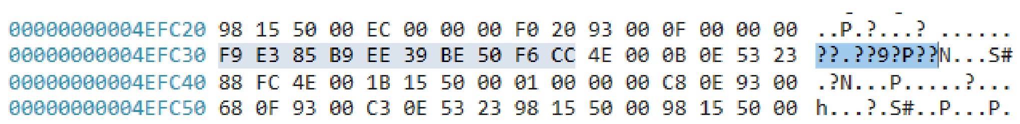
In ida we can see the array we need is located at `ebp - 0x10` which is `0x004efc40 - 0x10 = 0x004efc30` in memory



```
  char v11; // [esp+14h] [ebp-10h]
  __int64 v12; // [esp+15h] [ebp-Fh]
  char v13; // [esp+1Dh] [ebp-7h]

  v11 = 1;
  v12 = 0i64;
  v13 = 0;
  v0 = _time64(0);
  srand(v0);
  NumberOfBytesRead = 0;
  for ( i = 0; i < 10; ++i )
    *(&v11 + i) = rand() % 255;
  v13 = '\xCC';
```

In windbg press `alt+5` to open memory view, go to `0x004efc30` we can see the array with `0xCC` at the end



```
00000000004EFC20 98 15 50 00 EC 00 00 00 F0 20 93 00 0F 00 00 00  ..P.?...? ......
00000000004EFC30 F9 E3 85 B9 EE 39 BE 50 F6 CC 4E 00 0B 0E 53 23  ??.??9?P??N...S#
00000000004EFC40 88 FC 4E 00 1B 15 50 00 01 00 00 00 C8 0E 93 00  .?N...P.....?...
00000000004EFC50 68 0F 93 00 C3 0E 53 23 98 15 50 00 98 15 50 00  h...?.S#..P...P.
```

`F9 E3 85 B9 EE 39 BE 50 F6 CC`

3. test.txt decryption

The original content of `test.txt` is

```
hi i am hainh45
```