Details:

Author: **Yossef Zidan (@yossefzidann)**
Challenge Overview: The challenge goal is to modify the pe header of the Dont_run.bin binary to the right values that is being checked in the PE anatomy executable.

Step 1: Discovery

```
Cmder
C:\Users\joezid\Desktop\PE Anatomy-Medium\Challenge\PE Anatomy
λ file *
Dont_run.bin:   PE32+ executable (console) x86-64, for MS Windows
PE_Anatomy.exe: PE32+ executable (console) x86-64, for MS Windows

C:\Users\joezid\Desktop\PE Anatomy-Medium\Challenge\PE Anatomy
λ
```

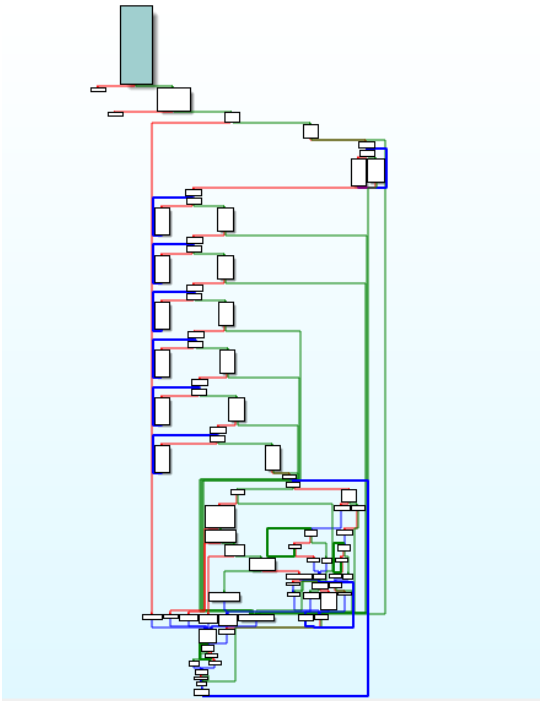
In this challenge, we are given two x64 PE files.

Step 2: Binary Analysis

```
C:\Users\joezid\Desktop\PE Anatomy-Medium\Challenge\PE Anatomy
λ PE_Anatomy.exe
No Flag for you ,Set Your Heart Ablaze to be able to see the flag.

C:\Users\joezid\Desktop\PE Anatomy-Medium\Challenge\PE Anatomy
λ
```

Running the anatomy file we will see the message no flag for you so we can open it in IDA to check it.



As we can see we have a lot of branches so let's check the code.

```
hFile = CreateFileA("Dont_run.bin", 0x80000000, 1u, 0i64, 3u, 0, 0i64);
if ( hFile == -1i64 )
  exit(55);
dwSize = GetFileSize(hFile, 0i64);
lpBuffer = VirtualAlloc(0i64, dwSize, 0x3000u, 4u);
if ( !ReadFile(hFile, lpBuffer, dwSize, &NumberOfBytesRead, 0i64) )
  exit(5555);
```

The program starts by reading the content of the Dont_run.bin file and saving it to the newly allocated memory lpBuffer.

```
if ( lpBuffer->e_magic == 0x5A4D )
{
  v33 = (lpBuffer + lpBuffer->e_lfanew);
```

Then it checks the file that was read if it starts with the magic bytes of the PE file "MZ".

```
  if ( v33->FileHeader.TimeDateStamp != 0x65617379 )
    goto LABEL_81;
  v37 = 0i64;
  v18 = 0;
  while ( v37 != 5 )
  {
    v19 = a12345[v37++] + v18;
    v18 = ((1025 * v19) >> 6) ^ (1025 * v19);
  }
  if ( v33->FileHeader.PointerToSymbolTable != 0x6767 )
    goto LABEL_81;
  v38 = 0i64;
  v20 = 0;
  while ( v38 != 5 )
  {
    v21 = aDdddd[v38++] + v20;
    v20 = ((1025 * v21) >> 6) ^ (1025 * v21);
  }
  if ( v33->FileHeader.NumberOfSymbols != 0x657A )
    goto LABEL_81;
  v39 = 0i64;
  v22 = 0;
  while ( v39 != 5 )
  {
    v23 = aZzzzz[v39++] + v22;
    v22 = ((1025 * v23) >> 6) ^ (1025 * v23);
  }
  if ( v33->FileHeader.Characteristics != 0x6E6F )
    goto LABEL_81;
```

Then we have four checks here for some info in the PE file header the time stamp have to be 0x65617379 , Pointer to symbol table have to be 0x6767 ,Number of symbols have to be 0x657A and final the characteristics have to be 0x6e6f

```
if ( v33->OptionalHeader.Magic != 0x9999 )
  goto LABEL_81;
v41 = 0i64;
v26 = 0;
while ( v41 != 5 )
{
  v27 = aWqwqw[v41++] + v26;
  v26 = ((1025 * v27) >> 6) ^ (1025 * v27);
}
if ( v33->OptionalHeader.AddressOfEntryPoint != 0x6969 )
  goto LABEL_81;
v42 = 0i64;
v28 = 0;
while ( v42 != 5 )
{
  v29 = aAscwg[v42++] + v28;
  v28 = ((1025 * v29) >> 6) ^ (1025 * v29);
}
v32 = 0;
if ( v33->OptionalHeader.ImageBase != 0x4142434461626364i64 )
  goto LABEL_81;
```

Then we have three checks against the values in the optional header Magic have to be 0x9999 , Address of Entry point have to be 0x6969 and the Image Base Adress have to be 0x4142434461626364

```
for ( i = 0; i < 3; ++i )
{
  v34 = (&lpBuffer[4].e_cparhdr + 40 * i + lpBuffer->e_lfanew);
  if ( i )
  {
    if ( i == 1 )
    {
      v8 = "bbbbb";
      v9 = v34 - "bbbbb";
      while ( 1 )
      {
        v10 = *v8;
        if ( *v8 != v8[v9] )
          break;
        ++v8;
        if ( !v10 )
        {
          v11 = 0;
          goto LABEL_53;
        }
      }
      v11 = v10 < v8[v9] ? -1 : 1;
L_53:
      if ( v11 )
        v32 = 1;
    }
  }
  else
  {
    v4 = "aaaaa";
    v5 = v34 - "aaaaa";
```

And finally we have some checks against the sections as the first section name have to be “aaaaa” , the second one “bbbbb” and the third one have to be “joezid”.

```
phProv = 0i64;
phHash = 0i64;
pdwDataLen = 0;
strcpy(&v50, "0123456789abcdef");
if ( CryptAcquireContextW(&phProv, 0i64, 0i64, 1u, 0xF0000000) )
{
  if ( CryptCreateHash(phProv, 0x8003u, 0i64, 0, &phHash) )
  {
    if ( CryptHashData(phHash, lpBuffer, 0x400u, 0) )
    {
      pdwDataLen = 16;
      if ( CryptGetHashParam(phHash, 2u, &pbData, &pdwDataLen, 0) )
      {
        sub_140001060("Your Flag is : ASCWG{");
        for ( j = 0; j < pdwDataLen; ++j )
          sub_140001060("%c%c");
        sub_140001060("}\n");
      }
      else
      {
        GetLastError();
        sub_140001060("CryptGetHashParam failed: %d\n");
      }
      CryptDestroyHash(phHash);
      CryptReleaseContext(phProv, 0);
      result = 1;
    }
    else
    {
      CryptReleaseContext(phProv, 0);
      CryptDestroyHash(phHash);
```

And if we passed all the checks the executable will calculate the md5 hash of the first 1024 bytes of the PE file Dont_run and add to the flag format.

```
File Header:
Machine: 0x1337
Time Stamp: 0x65617379
Ptr to symbol table: 0x6767
no of symbols: 0x657a
charactristics: 0x6e6f

Optional Header:
Magic: 0x9999
Entrypoint: 0x6969
Image Base: 0x4142434461626364

Sections:

.text  ---> aaaaa
.data ---> bbbbb
.rdata ---> joezid
```

So to get the flag we have to modify the Headers of the Dont_run binary and we can do that using PEbear.

```
C:\Users\joezid\Desktop\PE Anatomy-Medium\Challenge\PE Anatomy
λ PE_Anatomy.exe
Your Flag is : ASCWG{3f386b365b184f27674c4d1d1bdd2a50}
```

After patching the binary we will get the flag.

Flag: **ASCWG{3f386b365b184f27674c4d1d1bdd2a50}**