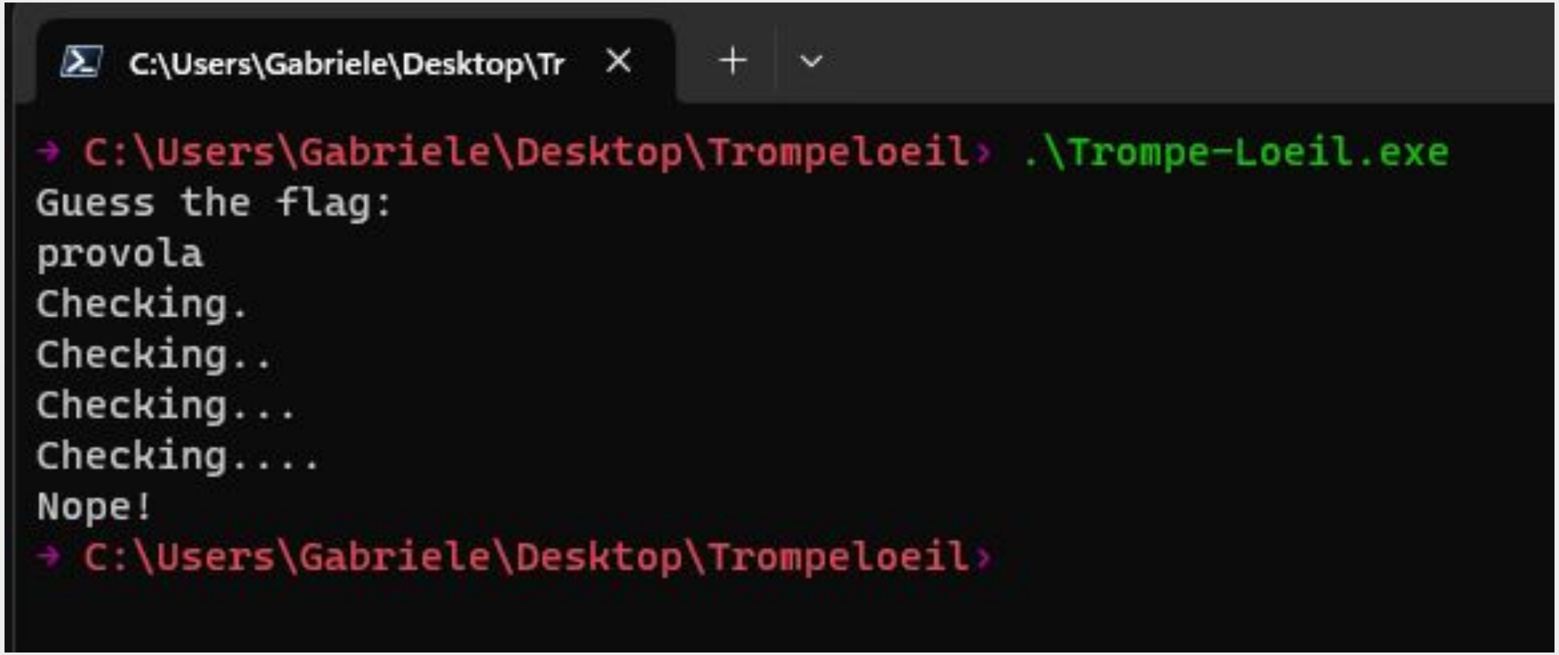
Trompe - L'Oeil

Insomni'hack CTF

Gabriele Digregorio *@lo_no*
Roberto A. Bertolini *@MrIndeciso*
6 July 2024

The Challenge



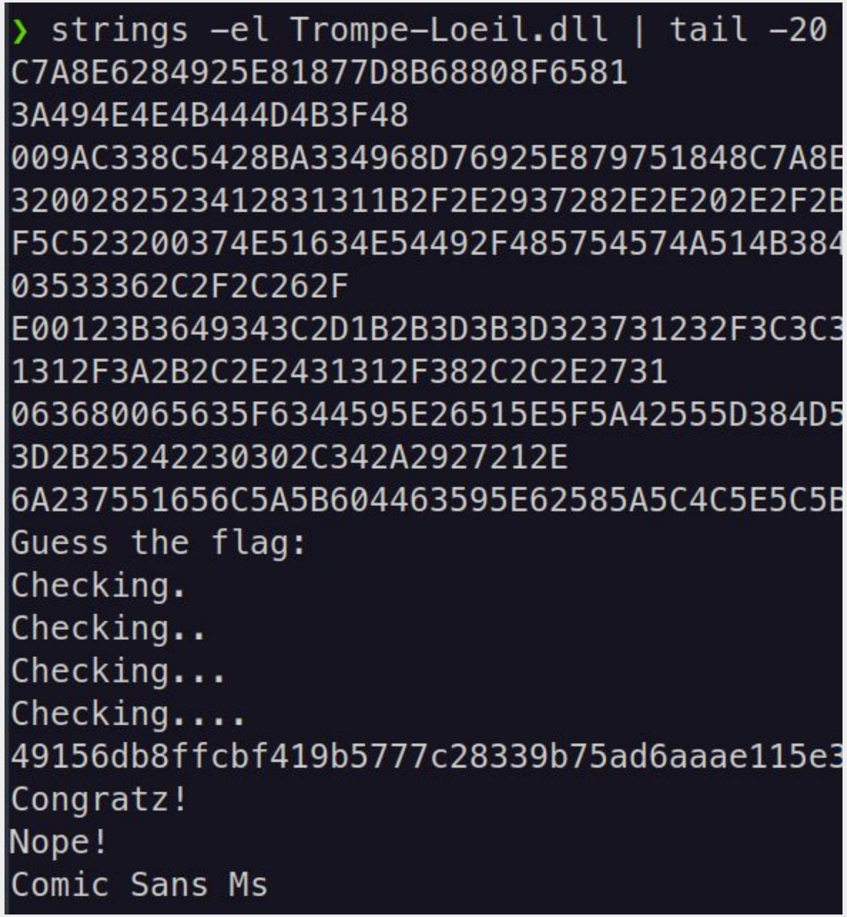
```
→ C:\Users\Gabriele\Desktop\Trompeloeil› .\Trompe-Loeil.exe
Guess the flag:
provola
Checking.
Checking..
Checking...
Checking....
Nope!
→ C:\Users\Gabriele\Desktop\Trompeloeil›
```

It just asks for the flag, **takes a few seconds** to perform the check and prints “Nope!”

Static Analysis

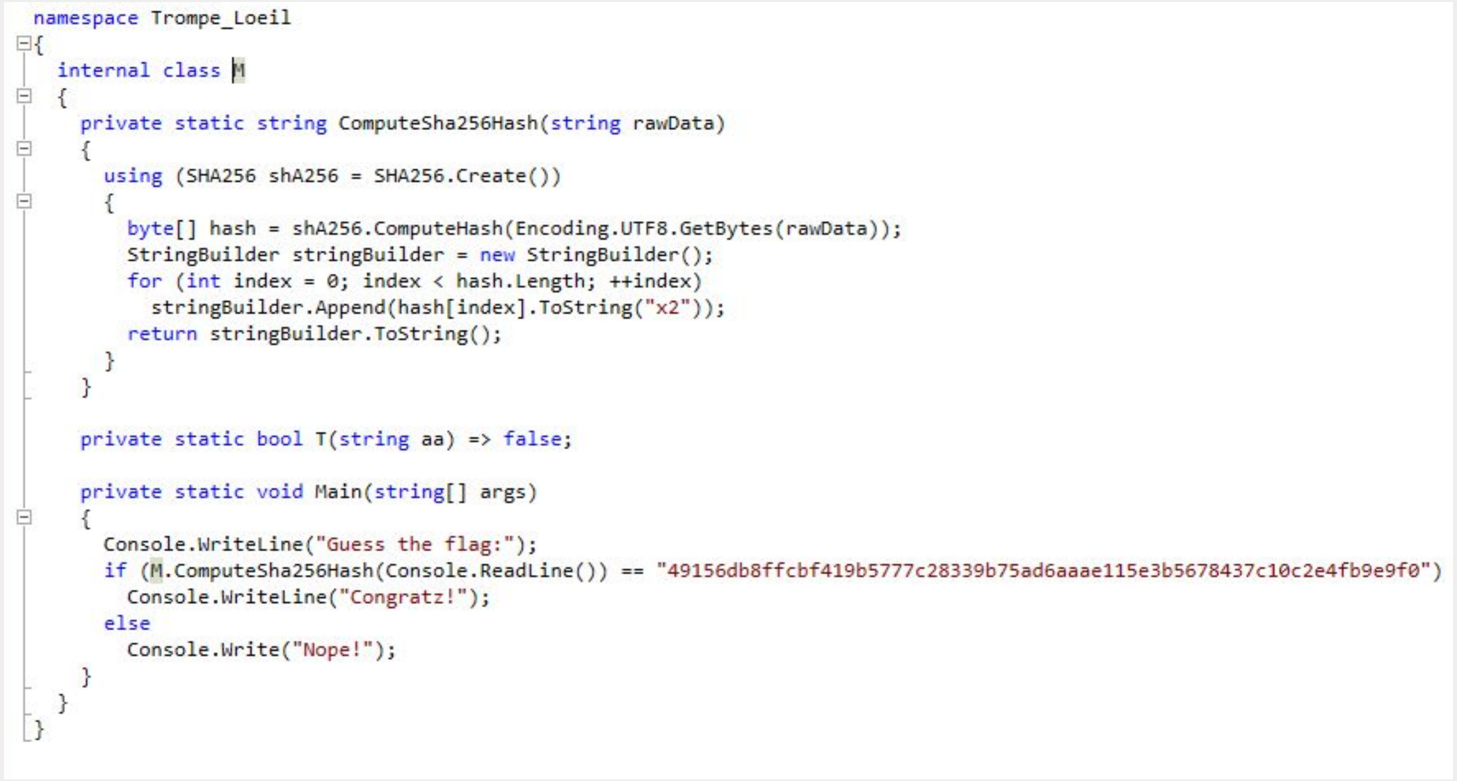
Along with Trompe-Loeil.exe we were given **Trompe-Loeil.dll**, which looked important. We quickly ran ***strings -el*** on it.

Thus, we deduced that **the checking logic must be in the .dll file**.



```
❯ strings -el Trompe-Loeil.dll | tail -20
C7A8E6284925E81877D8B68808F6581
3A494E4E4B444D4B3F48
009AC338C5428BA334968D76925E879751848C7A8E
3200282523412831311B2F2E2937282E2E202E2F2B
F5C523200374E51634E54492F485754574A514B384
03533362C2F2C262F
E00123B3649343C2D1B2B3D3B3D323731232F3C3C3
1312F3A2B2C2E2431312F382C2C2E2731
063680065635F6344595E26515E5F5A42555D384D5
3D2B25242230302C342A2927212E
6A237551656C5A5B604463595E62585A5C4C5E5C5B
Guess the flag:
Checking.
Checking..
Checking...
Checking....
49156db8ffcbf419b5777c28339b75ad6aaae115e3
Congratz!
Nope!
Comic Sans Ms
```

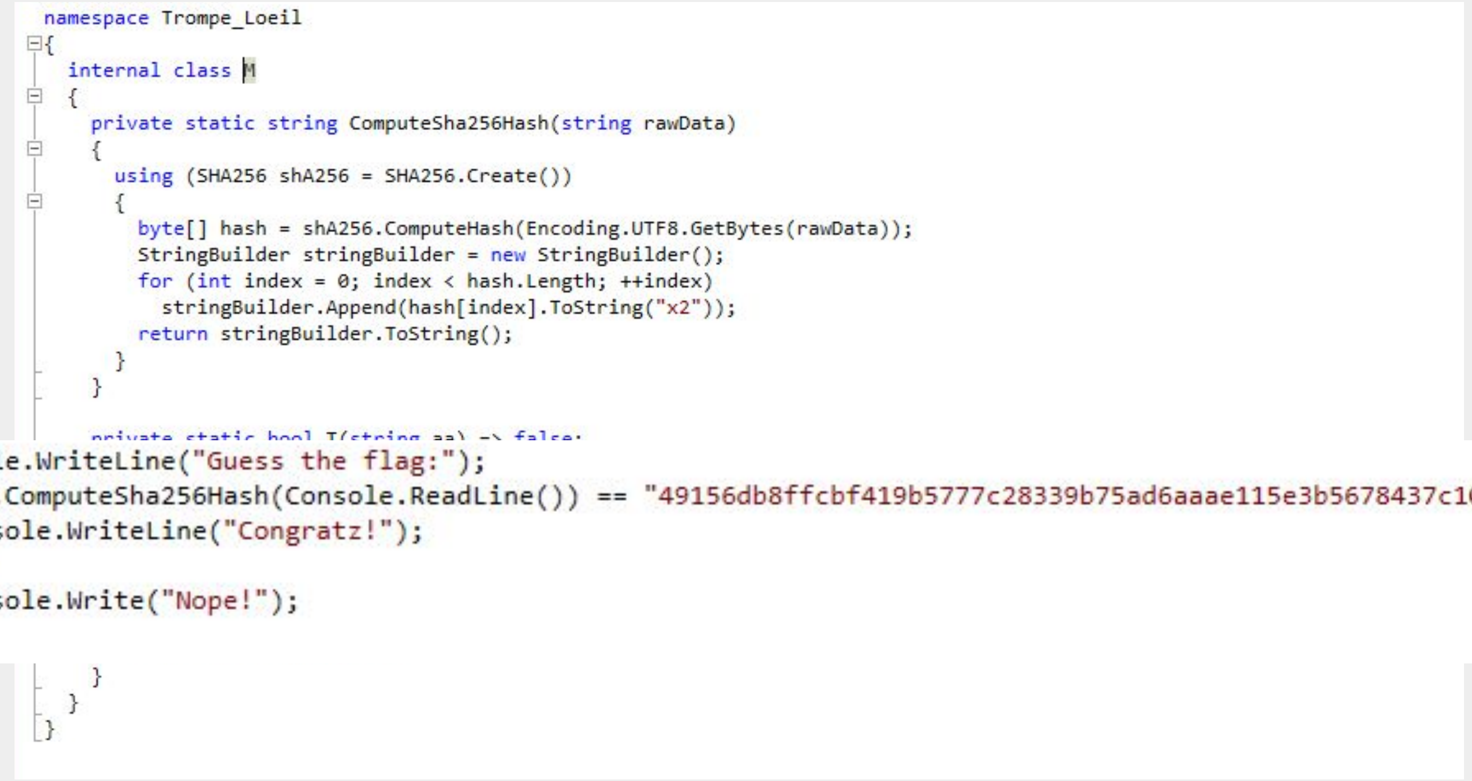
Static Analysis



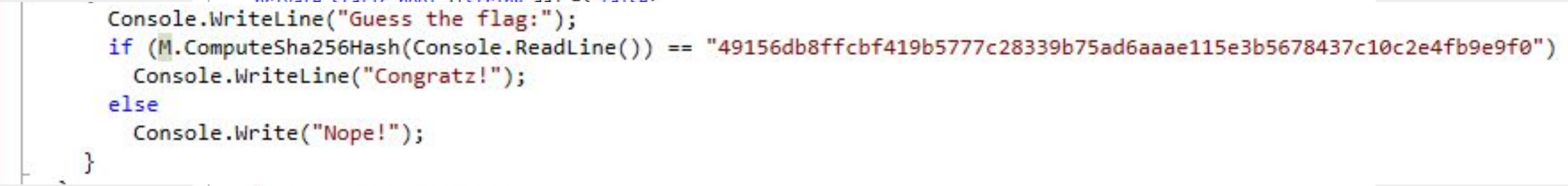
```
namespace Trompe_Loeil
{
  internal class M
  {
    private static string ComputeSha256Hash(string rawData)
    {
      using (SHA256 shA256 = SHA256.Create())
      {
        byte[] hash = shA256.ComputeHash(Encoding.UTF8.GetBytes(rawData));
        StringBuilder stringBuilder = new StringBuilder();
        for (int index = 0; index < hash.Length; ++index)
          stringBuilder.Append(hash[index].ToString("x2"));
        return stringBuilder.ToString();
      }
    }

    private static bool T(string aa) => false;

    private static void Main(string[] args)
    {
      Console.WriteLine("Guess the flag:");
      if (M.ComputeSha256Hash(Console.ReadLine()) == "49156db8ffcbf419b5777c28339b75ad6aaae115e3b5678437c10c2e4fb9e9f0")
        Console.WriteLine("Congratz!");
      else
        Console.Write("Nope!");
    }
  }
}
```

We disassembled the .NET dll using **dotPeek**

Static Analysis



```
namespace Trompe_Loeil
{
  internal class M
  {
    private static string ComputeSha256Hash(string rawData)
    {
      using (SHA256 shA256 = SHA256.Create())
      {
        byte[] hash = shA256.ComputeHash(Encoding.UTF8.GetBytes(rawData));
        StringBuilder stringBuilder = new StringBuilder();
        for (int index = 0; index < hash.Length; ++index)
          stringBuilder.Append(hash[index].ToString("x2"));
        return stringBuilder.ToString();
      }
    }

    private static bool T(string aa) => false;
```

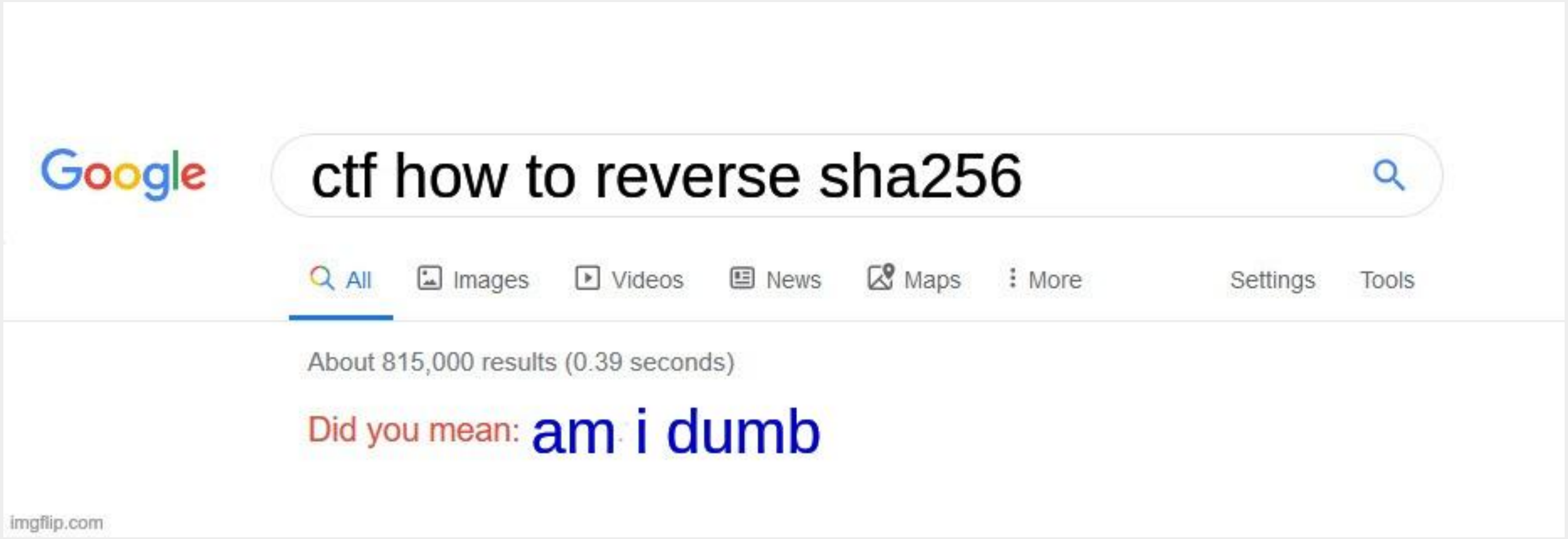


```
    Console.WriteLine("Guess the flag:");
    if (M.ComputeSha256Hash(Console.ReadLine()) == "49156db8ffcbf419b5777c28339b75ad6aaae115e3b5678437c10c2e4fb9e9f0")
      Console.WriteLine("Congratz!");
    else
      Console.Write("Nope!");
  }
```

```
    }
  }
}
```

We disassembled the .NET dll using **dotPeek**

Static Analysis



ctf how to reverse sha256

About 815,000 results (0.39 seconds)

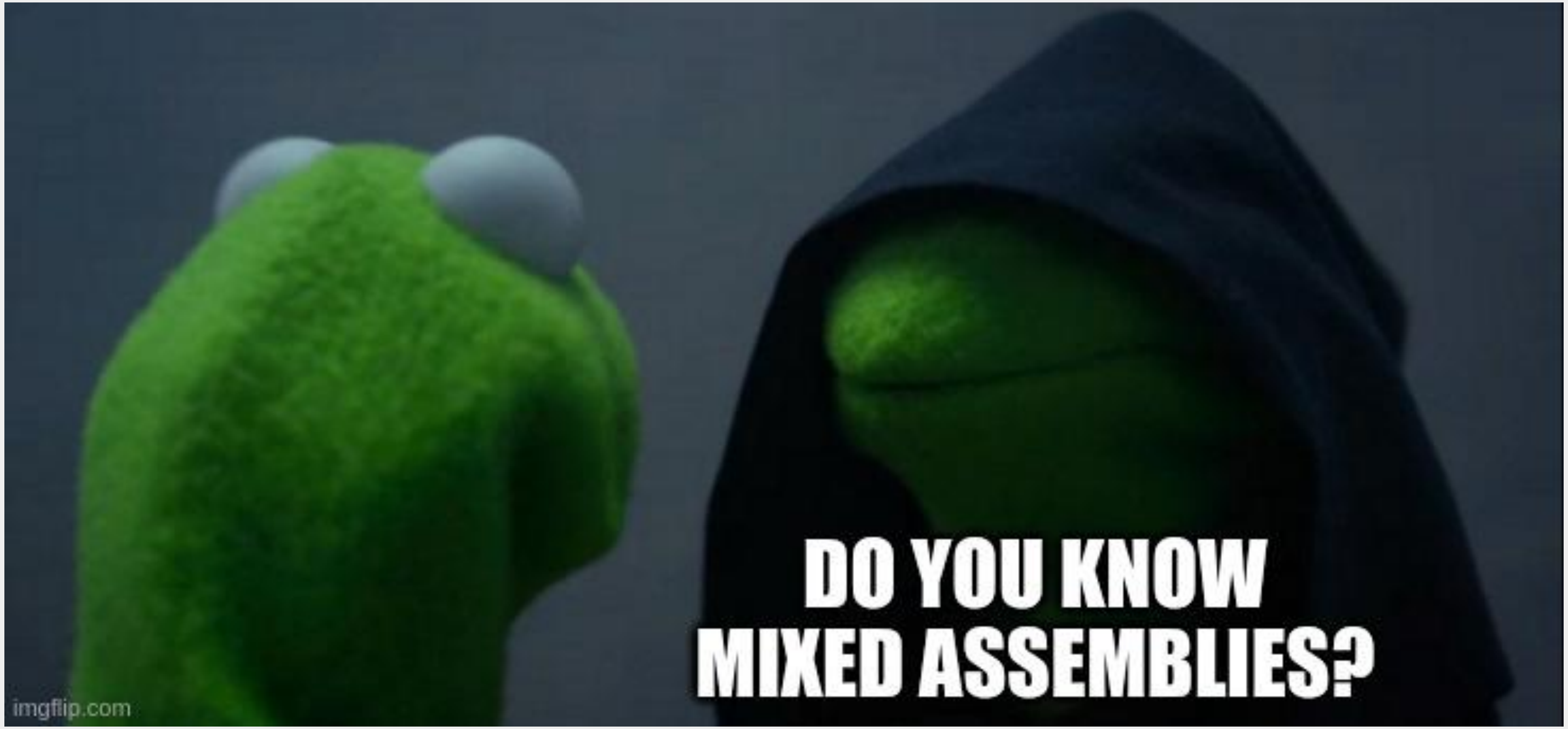
Did you mean: am i dumb

Static Analysis

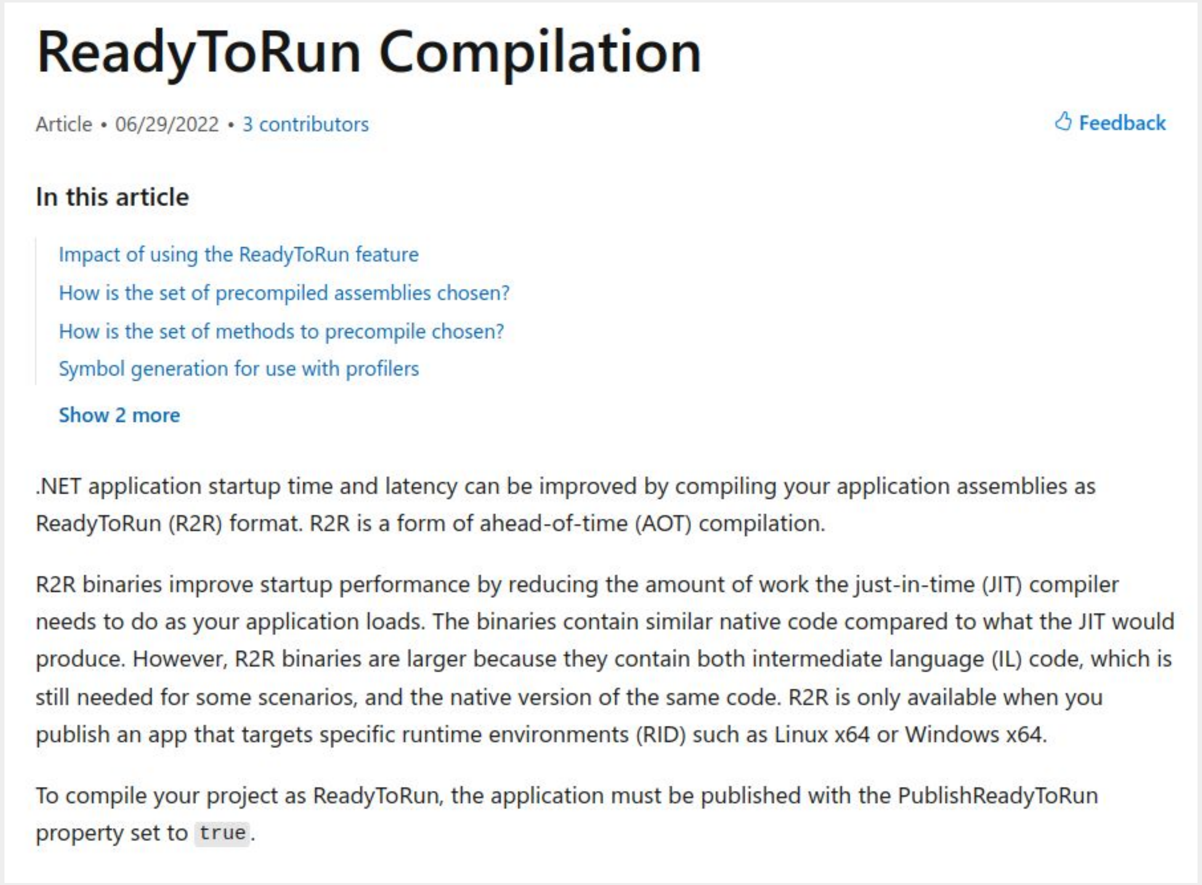
Static Analysis

Brainstorming Time

Guessing Analysis



Guessing Analysis



ReadyToRun Compilation

Article • 06/29/2022 • 3 contributors

Feedback

In this article

Impact of using the ReadyToRun feature
How is the set of precompiled assemblies chosen?
How is the set of methods to precompile chosen?
Symbol generation for use with profilers

Show 2 more

.NET application startup time and latency can be improved by compiling your application assemblies as ReadyToRun (R2R) format. R2R is a form of ahead-of-time (AOT) compilation.

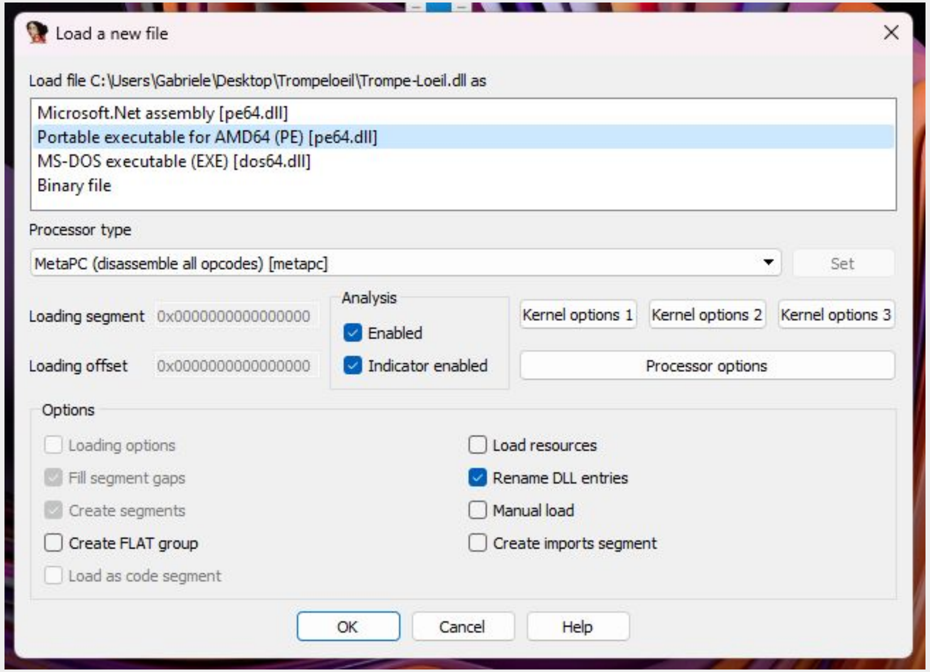
R2R binaries improve startup performance by reducing the amount of work the just-in-time (JIT) compiler needs to do as your application loads. The binaries contain similar native code compared to what the JIT would produce. However, R2R binaries are larger because they contain both intermediate language (IL) code, which is still needed for some scenarios, and the native version of the same code. R2R is only available when you publish an app that targets specific runtime environments (RID) such as Linux x64 or Windows x64.

To compile your project as ReadyToRun, the application must be published with the PublishReadyToRun property set to `true`.

Static Analysis

We opened Trompe-Loeil.dll as a **native PE**, not as a .NET assembly.

We started looking at the native code, to find differences with the bytecode.



Load a new file

Load file C:\Users\Gabriele\Desktop\Trompeloeil\Trompe-Loeil.dll as

- Microsoft.Net assembly [pe64.dll]
- [✓ Portable executable for AMD64 (PE) [pe64.dll]]
- MS-DOS executable (EXE) [dos64.dll]
- Binary file

Processor type: MetaPC (disassemble all opcodes) [metapc]

Loading segment: 0x0000000000000000

Loading offset: 0x0000000000000000

Analysis
- ✓ Enabled
- ✓ Indicator enabled

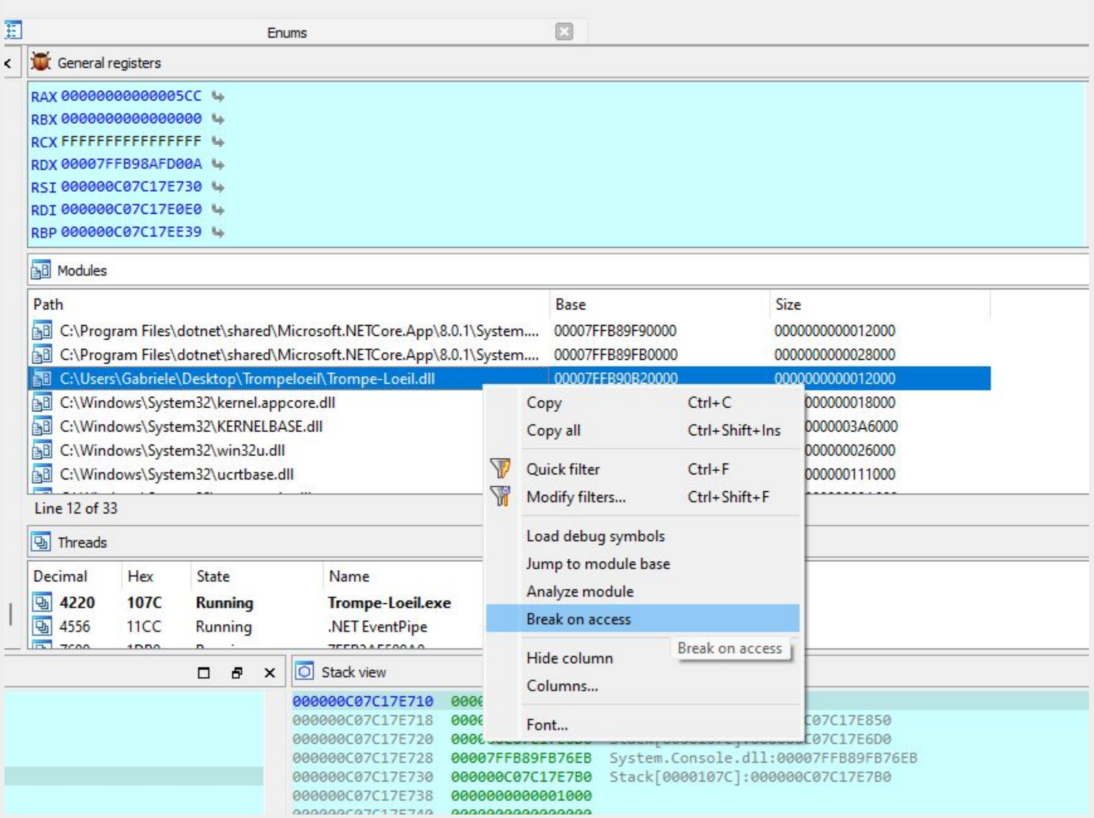
Options
- ☐ Loading options
- ✓ Fill segment gaps
- ✓ Create segments
- ☐ Create FLAT group
- ☐ Load as code segment
- ☐ Load resources
- ✓ Rename DLL entries
- ☐ Manual load
- ☐ Create imports segment

Dynamic Analysis

Dynamic Analysis

We didn’t have any symbols in the native code.

Using IDA Pro’s Windows Debugger, we set a **breakpoint on the first access** to the .dll.



```
RAX 00000000000005CC
RBX 0000000000000000
RCX FFFFFFFFFFFFFFFF
RDX 00007FFB98AFD00A
RSI 000000C07C17E730
RDI 000000C07C17E0E0
RBP 000000C07C17EE39
```

Path	Base	Size
C:\Program Files\dotnet\shared\Microsoft.NETCore.App\8.0.1\System....	00007FFB89F90000	0000000000012000
C:\Program Files\dotnet\shared\Microsoft.NETCore.App\8.0.1\System....	00007FFB89FB0000	0000000000028000
C:\Users\Gabriele\Desktop\Trompeloeil\Trompe-Loeil.dll	00007FFB90B20000	0000000000012000
C:\Windows\System32\kernel.appcore.dll		000000018000
C:\Windows\System32\KERNELBASE.dll		0000003A6000
C:\Windows\System32\win32u.dll		000000026000
C:\Windows\System32\ucrtbase.dll		000000111000

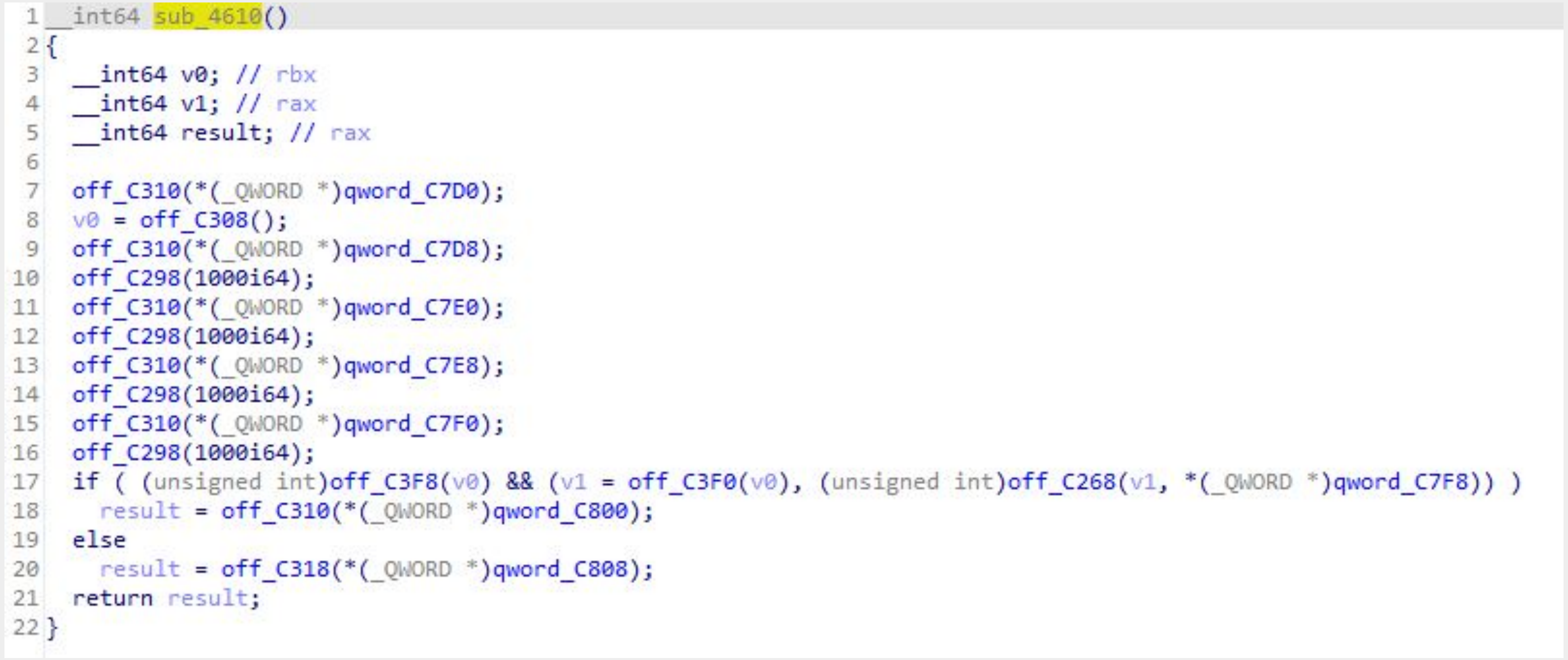
Line 12 of 33

Decimal	Hex	State	Name
4220	**107C**	**Running**	**Trompe-Loeil.exe**
4556	11CC	Running	.NET EventPipe

```
000000C07C17E710  0000
000000C07C17E718  0000                                    C07C17E850
000000C07C17E720  0000                                    C07C17E6D0
000000C07C17E728  00007FFB89FB76EB  System.Console.dll:00007FFB89FB76EB
000000C07C17E730  000000C07C17E7B0  Stack[0000107C]:000000C07C17E7B0
000000C07C17E738  0000000000001000
```

Static Analysis (Again)

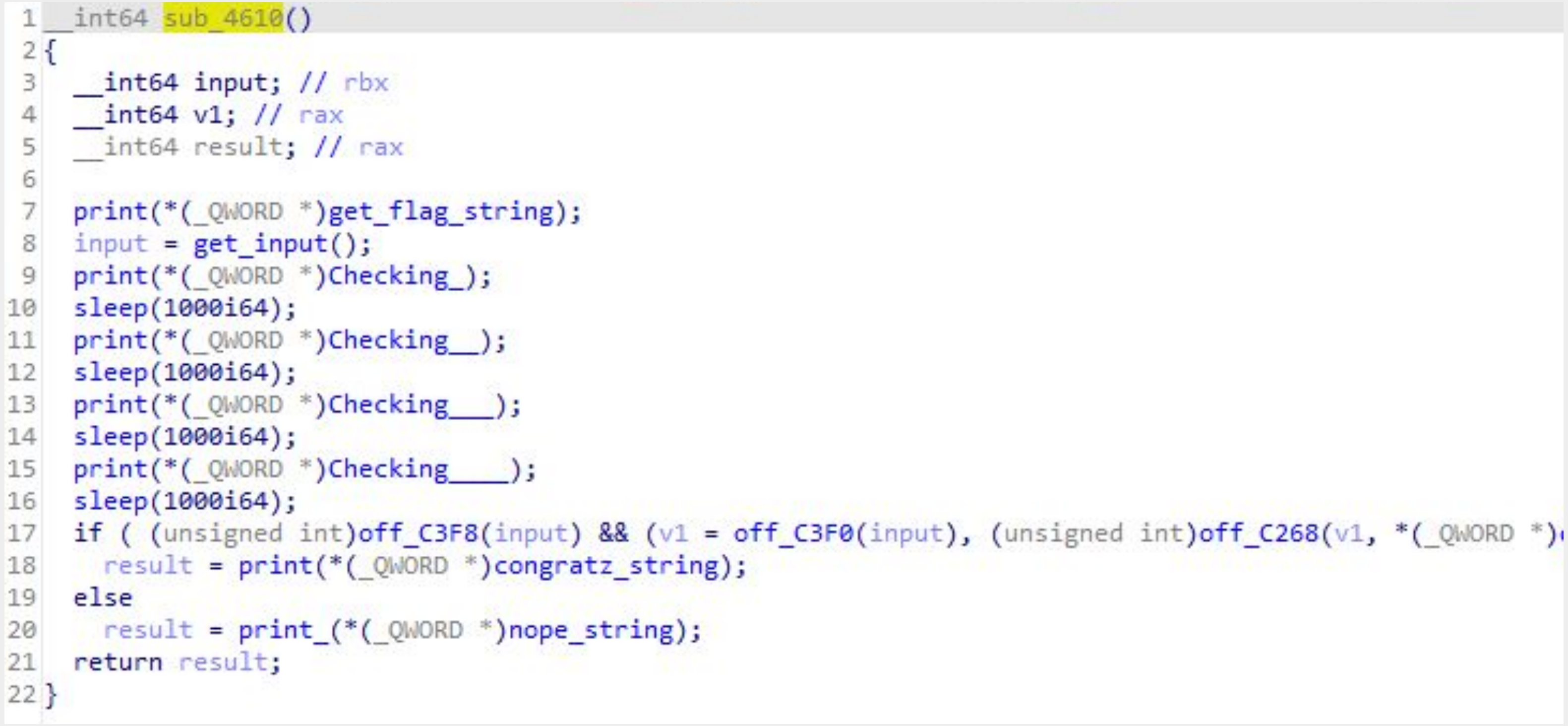
The function at offset 0x4610 appeared as following:



```
__int64 sub_4610()
{
  __int64 v0; // rbx
  __int64 v1; // rax
  __int64 result; // rax

  off_C310(*(_QWORD *)qword_C7D0);
  v0 = off_C308();
  off_C310(*(_QWORD *)qword_C7D8);
  off_C298(1000i64);
  off_C310(*(_QWORD *)qword_C7E0);
  off_C298(1000i64);
  off_C310(*(_QWORD *)qword_C7E8);
  off_C298(1000i64);
  off_C310(*(_QWORD *)qword_C7F0);
  off_C298(1000i64);
  if ( (unsigned int)off_C3F8(v0) && (v1 = off_C3F0(v0), (unsigned int)off_C268(v1, *(_QWORD *)qword_C7F8)) )
    result = off_C310(*(_QWORD *)qword_C800);
  else
    result = off_C318(*(_QWORD *)qword_C808);
  return result;
}
```

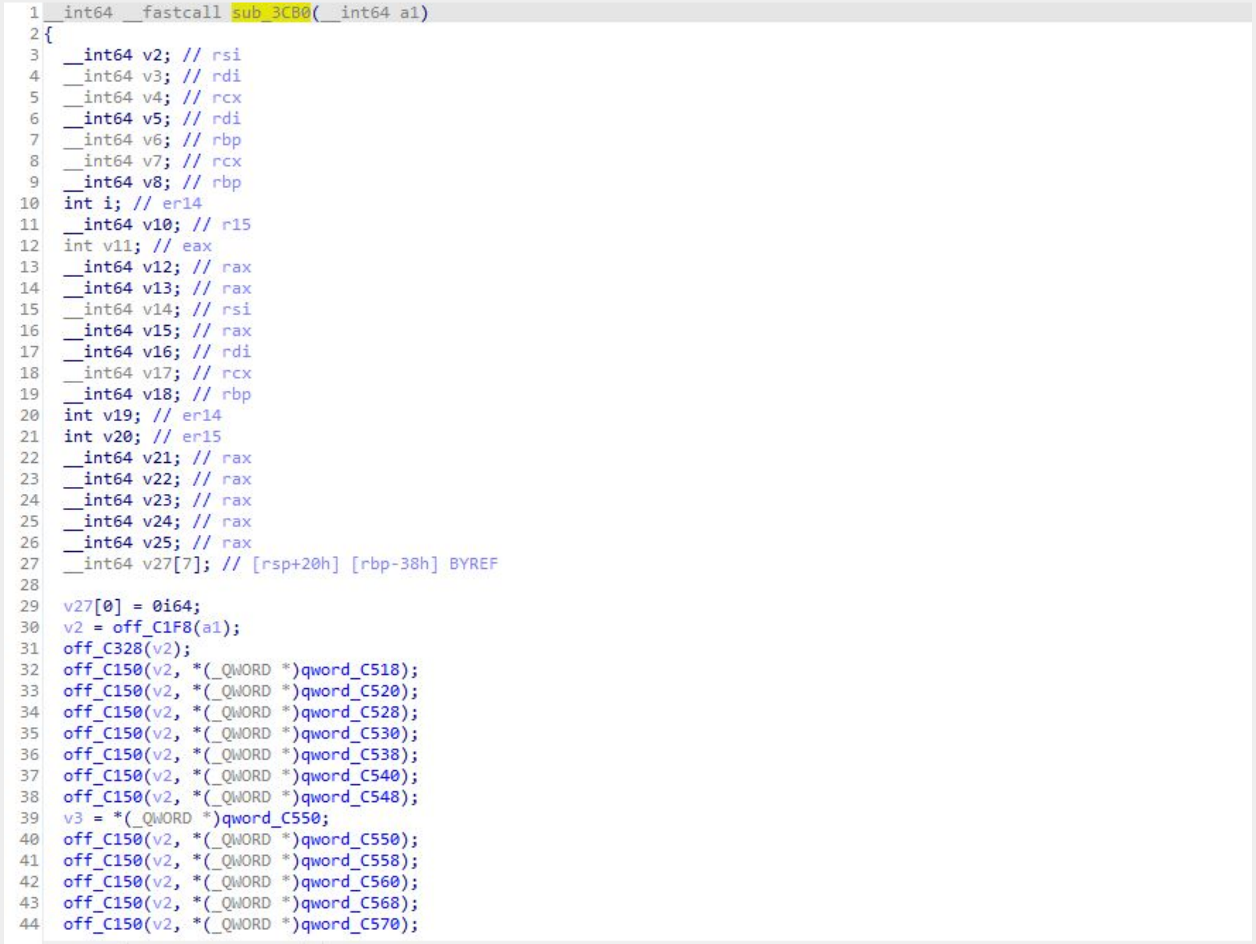
Static Analysis (Again)

The function at offset 0x4610 appeared as following:



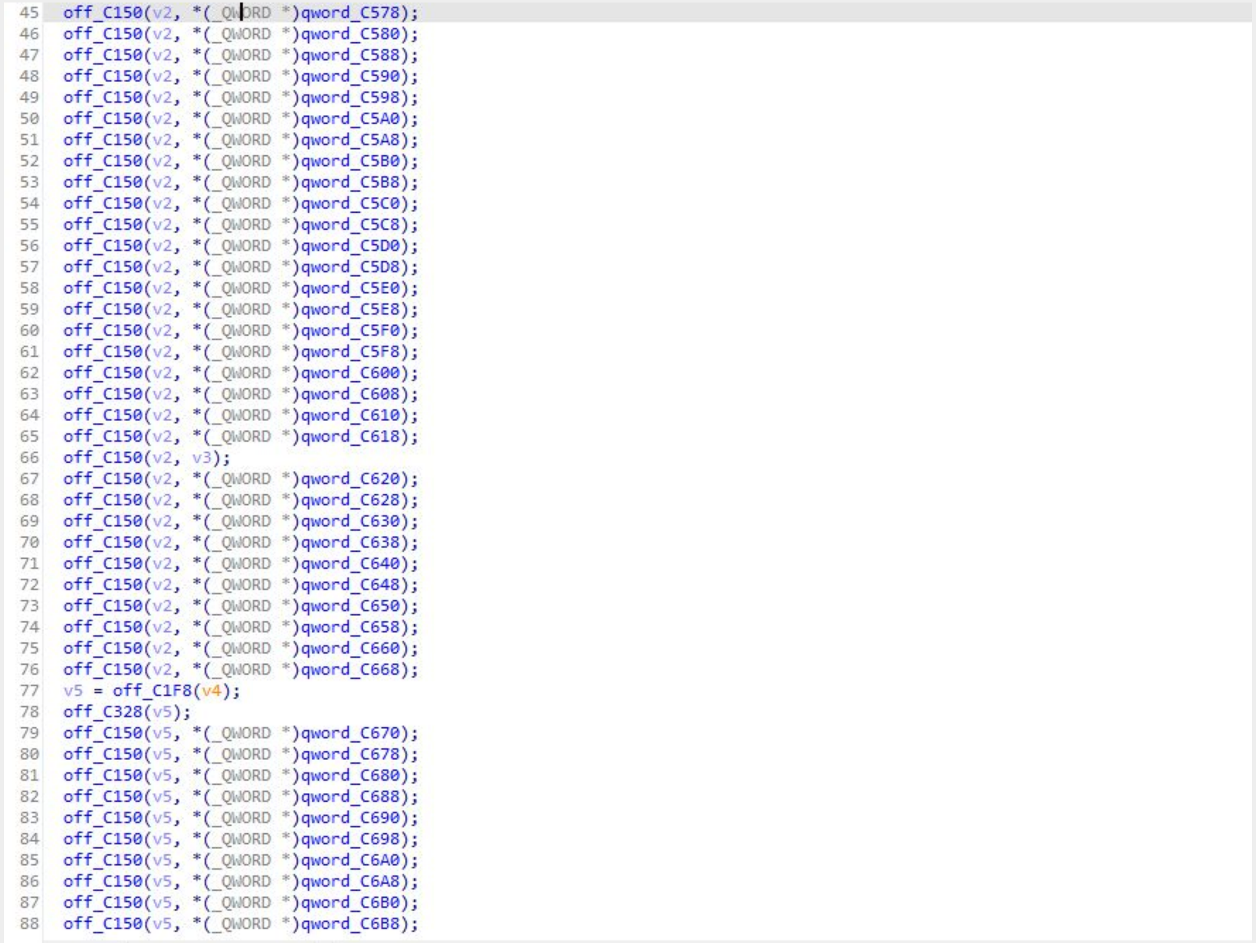
```
__int64 sub_4610()
{
  __int64 input; // rbx
  __int64 v1; // rax
  __int64 result; // rax

  print(*(_QWORD *)get_flag_string);
  input = get_input();
  print(*(_QWORD *)Checking_);
  sleep(1000i64);
  print(*(_QWORD *)Checking__);
  sleep(1000i64);
  print(*(_QWORD *)Checking___);
  sleep(1000i64);
  print(*(_QWORD *)Checking____);
  sleep(1000i64);
  if ( (unsigned int)off_C3F8(input) && (v1 = off_C3F0(input), (unsigned int)off_C268(v1, *(_QWORD *)
    result = print(*(_QWORD *)congratz_string);
  else
    result = print_(*(_QWORD *)nope_string);
  return result;
}
```

Painful Analysis



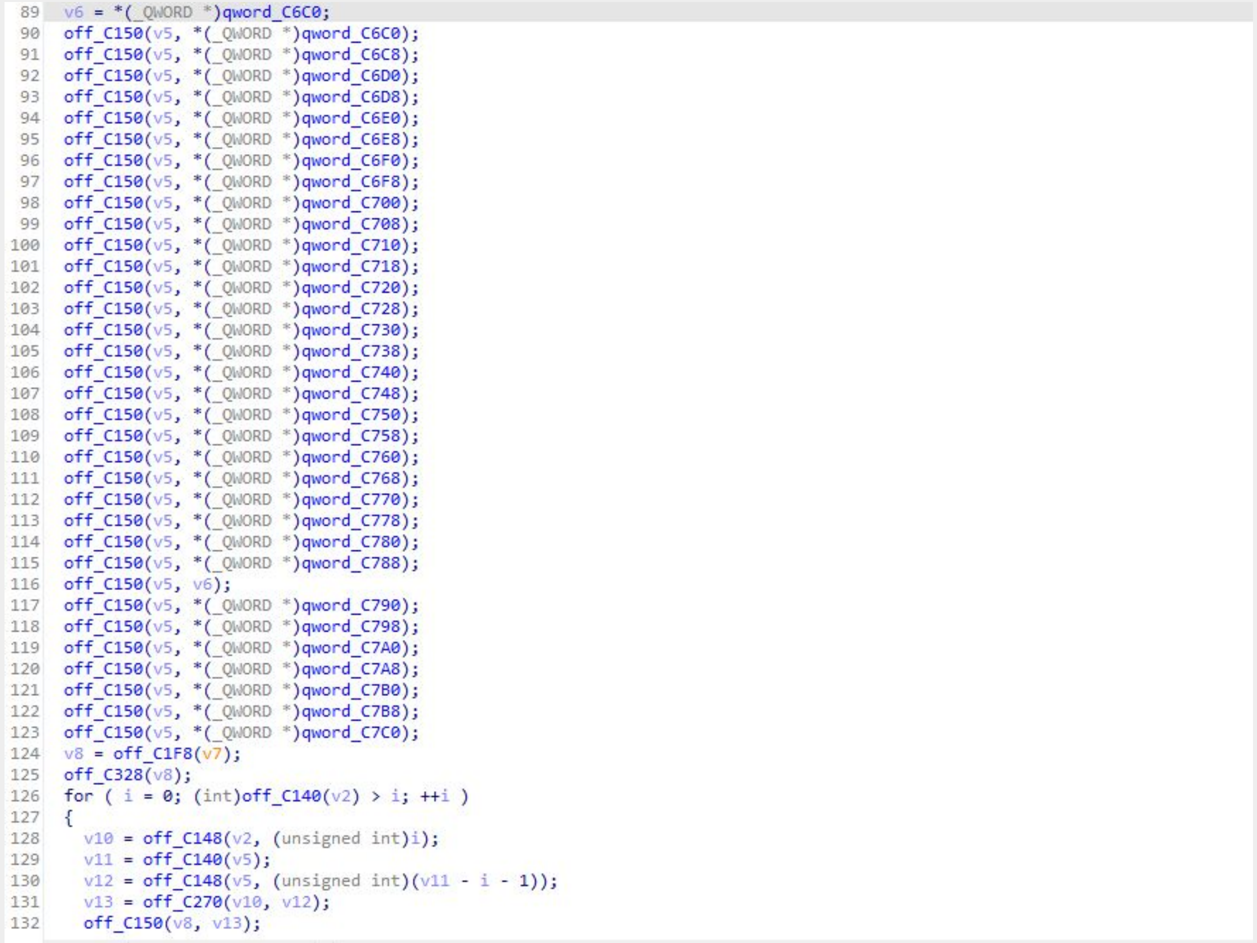
```
__int64 __fastcall sub_3CB0(__int64 a1)
{
  __int64 v2; // rsi
  __int64 v3; // rdi
  __int64 v4; // rcx
  __int64 v5; // rdi
  __int64 v6; // rbp
  __int64 v7; // rcx
  __int64 v8; // rbp
  int i; // er14
  __int64 v10; // r15
  int v11; // eax
  __int64 v12; // rax
  __int64 v13; // rax
  __int64 v14; // rsi
  __int64 v15; // rax
  __int64 v16; // rdi
  __int64 v17; // rcx
  __int64 v18; // rbp
  int v19; // er14
  int v20; // er15
  __int64 v21; // rax
  __int64 v22; // rax
  __int64 v23; // rax
  __int64 v24; // rax
  __int64 v25; // rax
  __int64 v27[7]; // [rsp+20h] [rbp-38h] BYREF

  v27[0] = 0i64;
  v2 = off_C1F8(a1);
  off_C328(v2);
  off_C150(v2, *(_QWORD *)qword_C518);
  off_C150(v2, *(_QWORD *)qword_C520);
  off_C150(v2, *(_QWORD *)qword_C528);
  off_C150(v2, *(_QWORD *)qword_C530);
  off_C150(v2, *(_QWORD *)qword_C538);
  off_C150(v2, *(_QWORD *)qword_C540);
  off_C150(v2, *(_QWORD *)qword_C548);
  v3 = *(_QWORD *)qword_C550;
  off_C150(v2, *(_QWORD *)qword_C550);
  off_C150(v2, *(_QWORD *)qword_C558);
  off_C150(v2, *(_QWORD *)qword_C560);
  off_C150(v2, *(_QWORD *)qword_C568);
  off_C150(v2, *(_QWORD *)qword_C570);
```

Painful Analysis



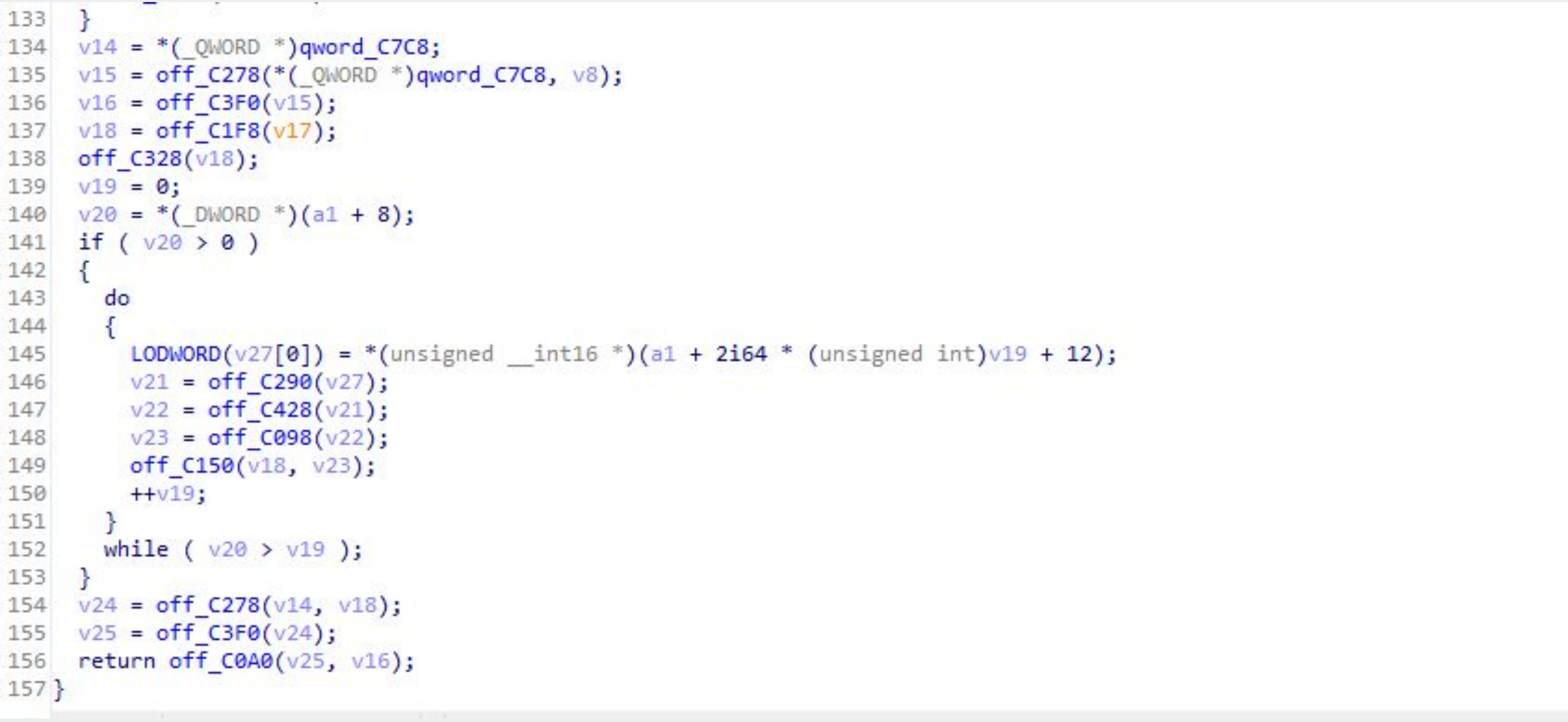
```
off_C150(v2, *(_QWORD *)qword_C578);
off_C150(v2, *(_QWORD *)qword_C580);
off_C150(v2, *(_QWORD *)qword_C588);
off_C150(v2, *(_QWORD *)qword_C590);
off_C150(v2, *(_QWORD *)qword_C598);
off_C150(v2, *(_QWORD *)qword_C5A0);
off_C150(v2, *(_QWORD *)qword_C5A8);
off_C150(v2, *(_QWORD *)qword_C5B0);
off_C150(v2, *(_QWORD *)qword_C5B8);
off_C150(v2, *(_QWORD *)qword_C5C0);
off_C150(v2, *(_QWORD *)qword_C5C8);
off_C150(v2, *(_QWORD *)qword_C5D0);
off_C150(v2, *(_QWORD *)qword_C5D8);
off_C150(v2, *(_QWORD *)qword_C5E0);
off_C150(v2, *(_QWORD *)qword_C5E8);
off_C150(v2, *(_QWORD *)qword_C5F0);
off_C150(v2, *(_QWORD *)qword_C5F8);
off_C150(v2, *(_QWORD *)qword_C600);
off_C150(v2, *(_QWORD *)qword_C608);
off_C150(v2, *(_QWORD *)qword_C610);
off_C150(v2, *(_QWORD *)qword_C618);
off_C150(v2, v3);
off_C150(v2, *(_QWORD *)qword_C620);
off_C150(v2, *(_QWORD *)qword_C628);
off_C150(v2, *(_QWORD *)qword_C630);
off_C150(v2, *(_QWORD *)qword_C638);
off_C150(v2, *(_QWORD *)qword_C640);
off_C150(v2, *(_QWORD *)qword_C648);
off_C150(v2, *(_QWORD *)qword_C650);
off_C150(v2, *(_QWORD *)qword_C658);
off_C150(v2, *(_QWORD *)qword_C660);
off_C150(v2, *(_QWORD *)qword_C668);
v5 = off_C1F8(v4);
off_C328(v5);
off_C150(v5, *(_QWORD *)qword_C670);
off_C150(v5, *(_QWORD *)qword_C678);
off_C150(v5, *(_QWORD *)qword_C680);
off_C150(v5, *(_QWORD *)qword_C688);
off_C150(v5, *(_QWORD *)qword_C690);
off_C150(v5, *(_QWORD *)qword_C698);
off_C150(v5, *(_QWORD *)qword_C6A0);
off_C150(v5, *(_QWORD *)qword_C6A8);
off_C150(v5, *(_QWORD *)qword_C6B0);
off_C150(v5, *(_QWORD *)qword_C6B8);
```

Painful Analysis



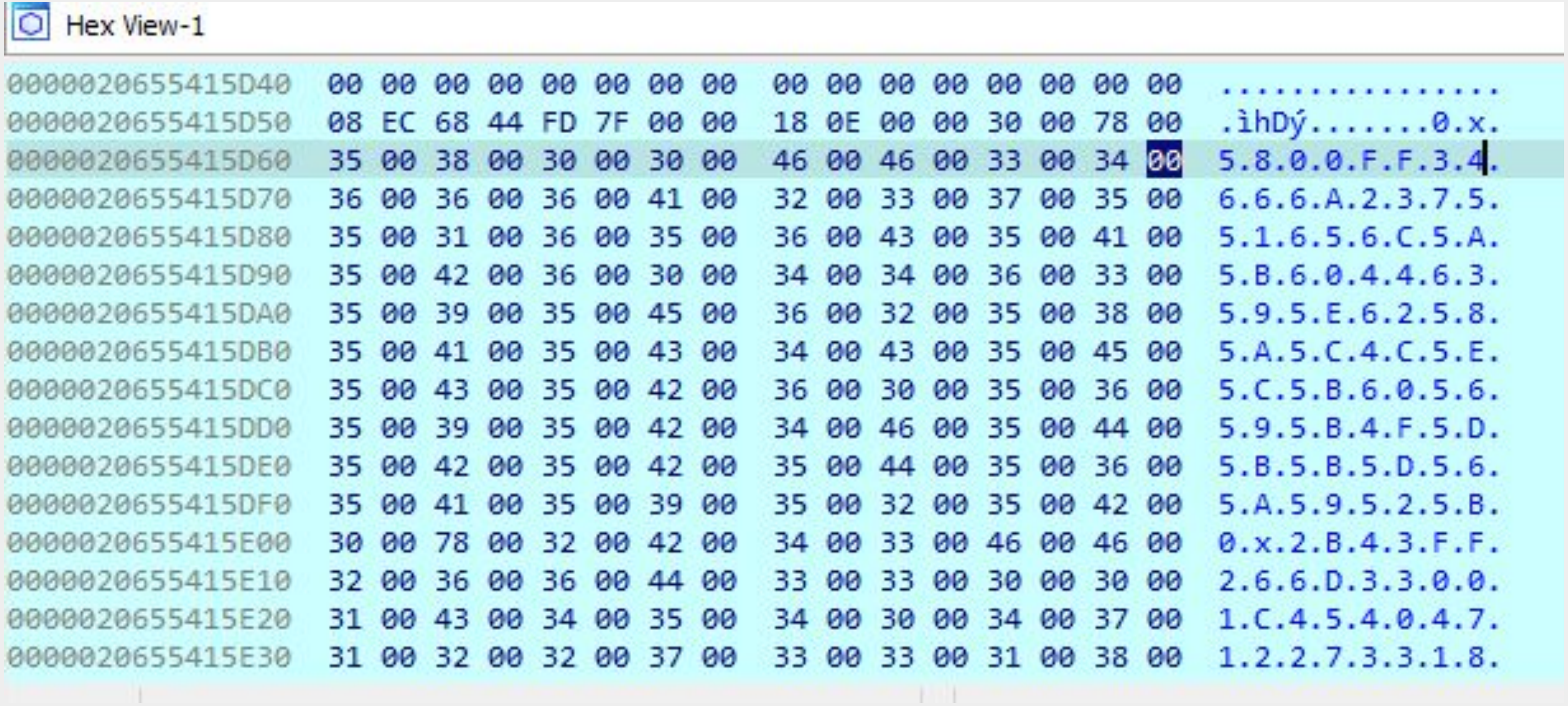
```
v6 = *(_QWORD *)qword_C6C0;
off_C150(v5, *(_QWORD *)qword_C6C0);
off_C150(v5, *(_QWORD *)qword_C6C8);
off_C150(v5, *(_QWORD *)qword_C6D0);
off_C150(v5, *(_QWORD *)qword_C6D8);
off_C150(v5, *(_QWORD *)qword_C6E0);
off_C150(v5, *(_QWORD *)qword_C6E8);
off_C150(v5, *(_QWORD *)qword_C6F0);
off_C150(v5, *(_QWORD *)qword_C6F8);
off_C150(v5, *(_QWORD *)qword_C700);
off_C150(v5, *(_QWORD *)qword_C708);
off_C150(v5, *(_QWORD *)qword_C710);
off_C150(v5, *(_QWORD *)qword_C718);
off_C150(v5, *(_QWORD *)qword_C720);
off_C150(v5, *(_QWORD *)qword_C728);
off_C150(v5, *(_QWORD *)qword_C730);
off_C150(v5, *(_QWORD *)qword_C738);
off_C150(v5, *(_QWORD *)qword_C740);
off_C150(v5, *(_QWORD *)qword_C748);
off_C150(v5, *(_QWORD *)qword_C750);
off_C150(v5, *(_QWORD *)qword_C758);
off_C150(v5, *(_QWORD *)qword_C760);
off_C150(v5, *(_QWORD *)qword_C768);
off_C150(v5, *(_QWORD *)qword_C770);
off_C150(v5, *(_QWORD *)qword_C778);
off_C150(v5, *(_QWORD *)qword_C780);
off_C150(v5, *(_QWORD *)qword_C788);
off_C150(v5, v6);
off_C150(v5, *(_QWORD *)qword_C790);
off_C150(v5, *(_QWORD *)qword_C798);
off_C150(v5, *(_QWORD *)qword_C7A0);
off_C150(v5, *(_QWORD *)qword_C7A8);
off_C150(v5, *(_QWORD *)qword_C7B0);
off_C150(v5, *(_QWORD *)qword_C7B8);
off_C150(v5, *(_QWORD *)qword_C7C0);
v8 = off_C1F8(v7);
off_C328(v8);
for ( i = 0; (int)off_C140(v2) > i; ++i )
{
  v10 = off_C148(v2, (unsigned int)i);
  v11 = off_C140(v5);
  v12 = off_C148(v5, (unsigned int)(v11 - i - 1));
  v13 = off_C270(v10, v12);
  off_C150(v8, v13);
```

Painful Analysis



```
  }
  v14 = *(_QWORD *)qword_C7C8;
  v15 = off_C278(*(_QWORD *)qword_C7C8, v8);
  v16 = off_C3F0(v15);
  v18 = off_C1F8(v17);
  off_C328(v18);
  v19 = 0;
  v20 = *(_DWORD *)(a1 + 8);
  if ( v20 > 0 )
  {
    do
    {
      LODWORD(v27[0]) = *(unsigned __int16 *)(a1 + 2i64 * (unsigned int)v19 + 12);
      v21 = off_C290(v27);
      v22 = off_C428(v21);
      v23 = off_C098(v22);
      off_C150(v18, v23);
      ++v19;
    }
    while ( v20 > v19 );
  }
  v24 = off_C278(v14, v18);
  v25 = off_C3F0(v24);
  return off_C0A0(v25, v16);
}
```

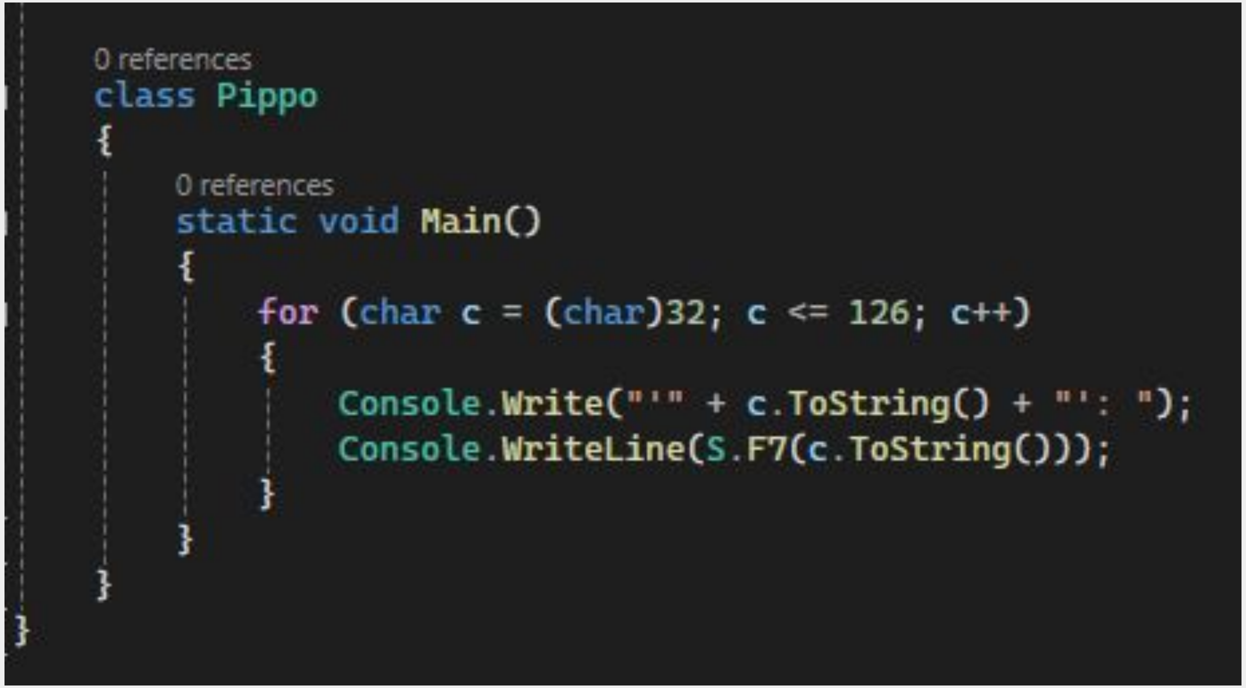
Serious Analysis


Hex View-1

```
000002065541SD40  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
000002065541SD50  08 EC 68 44 FD 7F 00 00  18 0E 00 00 30 00 78 00  .ìhDý.......0.x.
000002065541SD60  35 00 38 00 30 00 30 00  46 00 46 00 33 00 34 00  5.8.0.0.F.F.3.4.
000002065541SD70  36 00 36 00 36 00 41 00  32 00 33 00 37 00 35 00  6.6.6.A.2.3.7.5.
000002065541SD80  35 00 31 00 36 00 35 00  36 00 43 00 35 00 41 00  5.1.6.5.6.C.5.A.
000002065541SD90  35 00 42 00 36 00 30 00  34 00 34 00 36 00 33 00  5.B.6.0.4.4.6.3.
000002065541SDA0  35 00 39 00 35 00 45 00  36 00 32 00 35 00 38 00  5.9.5.E.6.2.5.8.
000002065541SDB0  35 00 41 00 35 00 43 00  34 00 43 00 35 00 45 00  5.A.5.C.4.C.5.E.
000002065541SDC0  35 00 43 00 35 00 42 00  36 00 30 00 35 00 36 00  5.C.5.B.6.0.5.6.
000002065541SDD0  35 00 39 00 35 00 42 00  34 00 46 00 35 00 44 00  5.9.5.B.4.F.5.D.
000002065541SDE0  35 00 42 00 35 00 42 00  35 00 44 00 35 00 36 00  5.B.5.B.5.D.5.6.
000002065541SDF0  35 00 41 00 35 00 39 00  35 00 32 00 35 00 42 00  5.A.5.9.5.2.5.B.
000002065541SE00  30 00 78 00 32 00 42 00  34 00 33 00 46 00 46 00  0.x.2.B.4.3.F.F.
000002065541SE10  32 00 36 00 36 00 44 00  33 00 33 00 30 00 30 00  2.6.6.D.3.3.0.0.
000002065541SE20  31 00 43 00 34 00 35 00  34 00 30 00 34 00 37 00  1.C.4.5.4.0.4.7.
000002065541SE30  31 00 32 00 32 00 37 00  33 00 33 00 31 00 38 00  1.2.2.7.3.3.1.8.
```

Guessing Time (Again)

Bruteforce Time

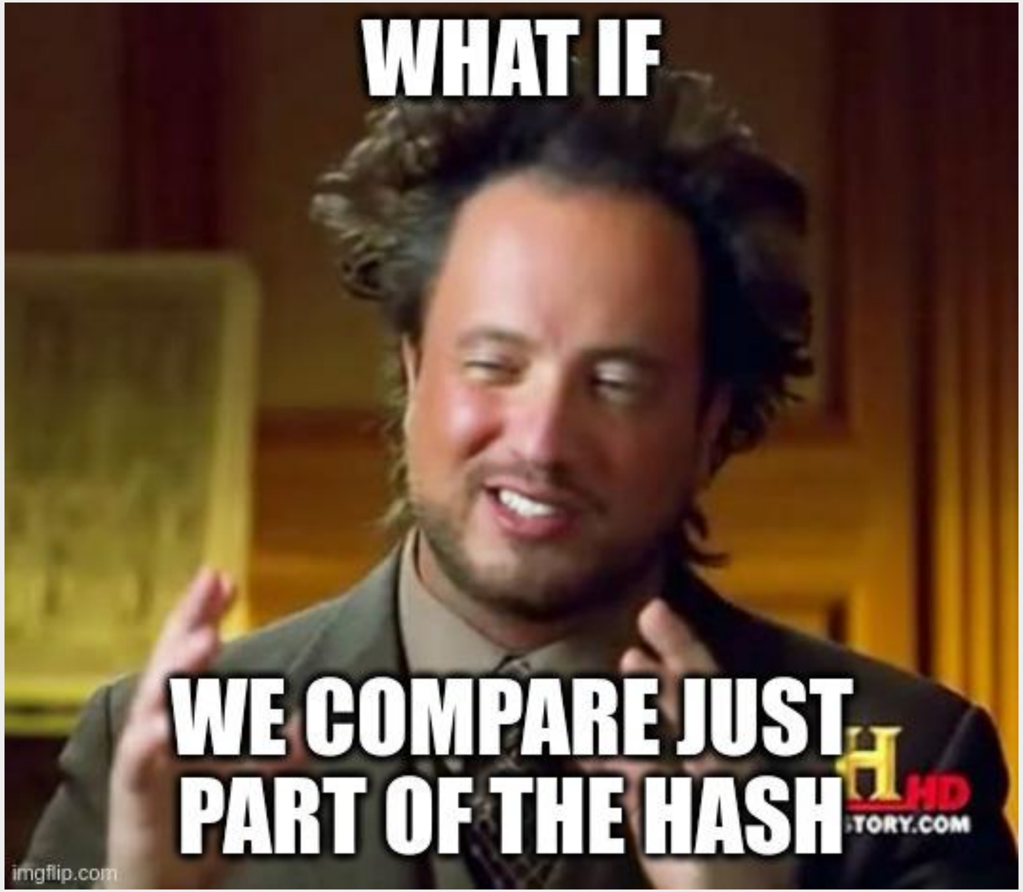


```
0 references
class Pippo
{
    0 references
    static void Main()
    {
        for (char c = (char)32; c <= 126; c++)
        {
            Console.Write("'" + c.ToString() + "': ");
            Console.WriteLine(S.F7(c.ToString()));
        }
    }
}
}
```

Bruteforce Time

brute the hash for all chars
compare with the hashes we found
no matches
no matches
imgflip.com

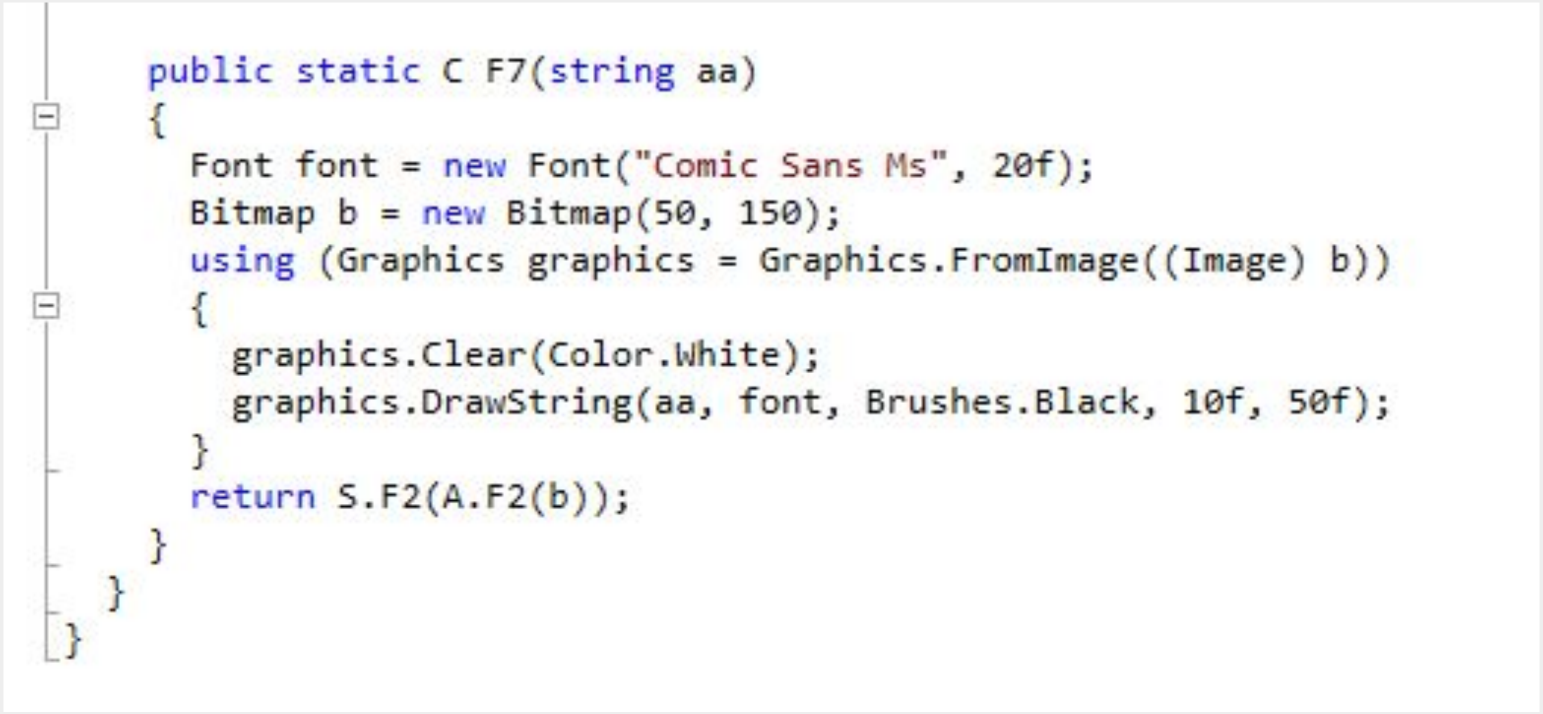
Guessing Time



Flag Time

INS{Re4dy_2_Run_M4ster_4r3_S0_R34dy_2_Fl4g!}

Serious Analysis



```
    public static C F7(string aa)
    {
      Font font = new Font("Comic Sans Ms", 20f);
      Bitmap b = new Bitmap(50, 150);
      using (Graphics graphics = Graphics.FromImage((Image) b))
      {
        graphics.Clear(Color.White);
        graphics.DrawString(aa, font, Brushes.Black, 10f, 50f);
      }
      return S.F2(A.F2(b));
    }
  }
}
```

CYBER CHALLENGE.IT

POLITECNICO MILANO 1863

Thanks for your attention

info@towerofhanoi.it

TOWER OF HANOI