PSP0201 Week 5 Writeup

Group Name: Undecided

Members

ID	Name	Role
1211101390	Aslamia Najwa Binti Ahmad Khadri	Leader
1211100431	Mohammad Omar Torofder	Member
1211103388	Vishnu Karmegam	Member
1211103092	Farryn Aisha Binti Muhd Firdaus	Member

Day 18 : Reverse Engineering – The Bits of Christmas

Tools used: Kali Linux, Firefox

Solution/walkthrough:

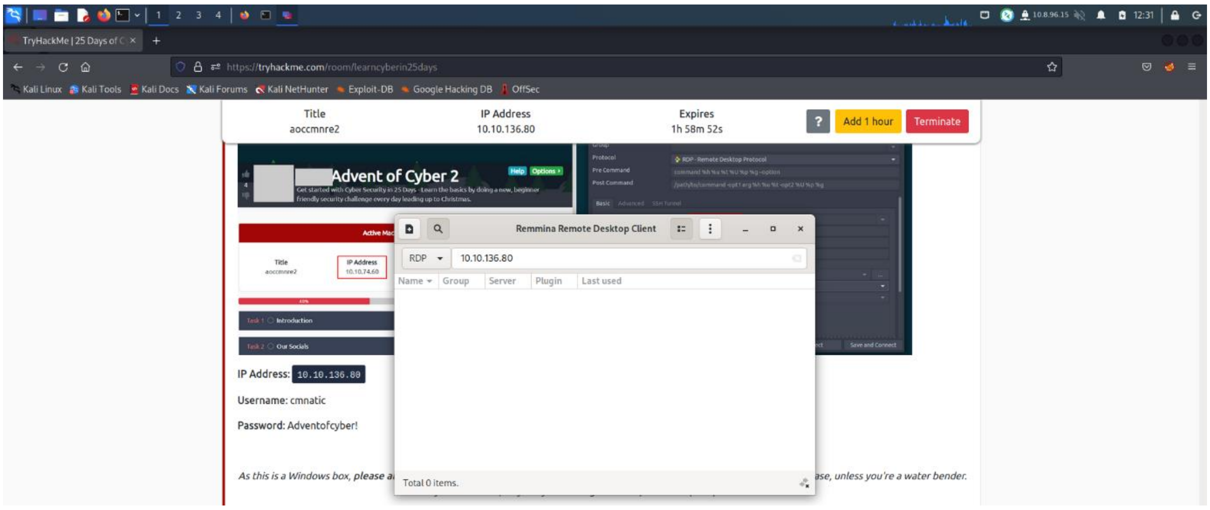
Question 1

To start the task, we first deployed the machine and log in using the Remote Desktop Protocol (RDP). Using Remina, we connected to the instance with the credentials provided.

IP Address: MACHINE_IP

Username: cmnatic

Password: Advertofcyber!



Title: aoccmnre2 | IP Address: 10.10.136.80 | Expires: 1h 58m 52s

Remmina Remote Desktop Client

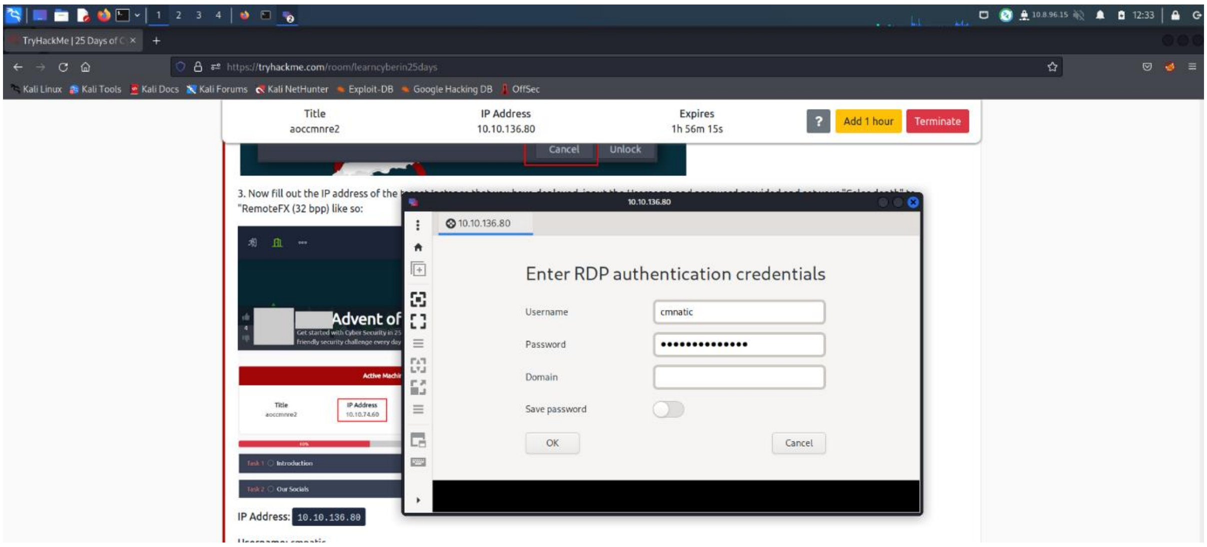
RDP 10.10.136.80

Total 0 items.

IP Address: 10.10.136.80

Username: cmnatic

Password: Adventofcyber!



Title: aoccmnre2 | IP Address: 10.10.136.80 | Expires: 1h 56m 15s

3. Now fill out the IP address of the

"RemoteFX (32 bpp) like so:

10.10.136.80

Enter RDP authentication credentials

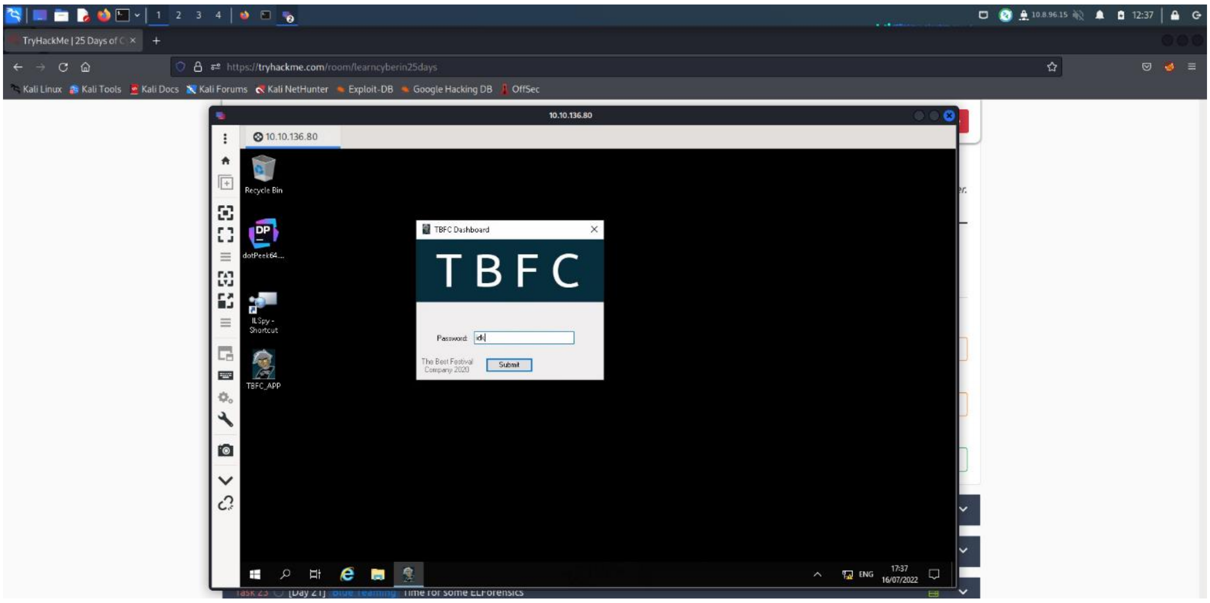
Username: cmnatic

Password: ••••••••••••••

Domain:

IP Address: 10.10.136.80

Once we are logged in, we open the TBFC_APP in the desktop and tried to enter a random password.



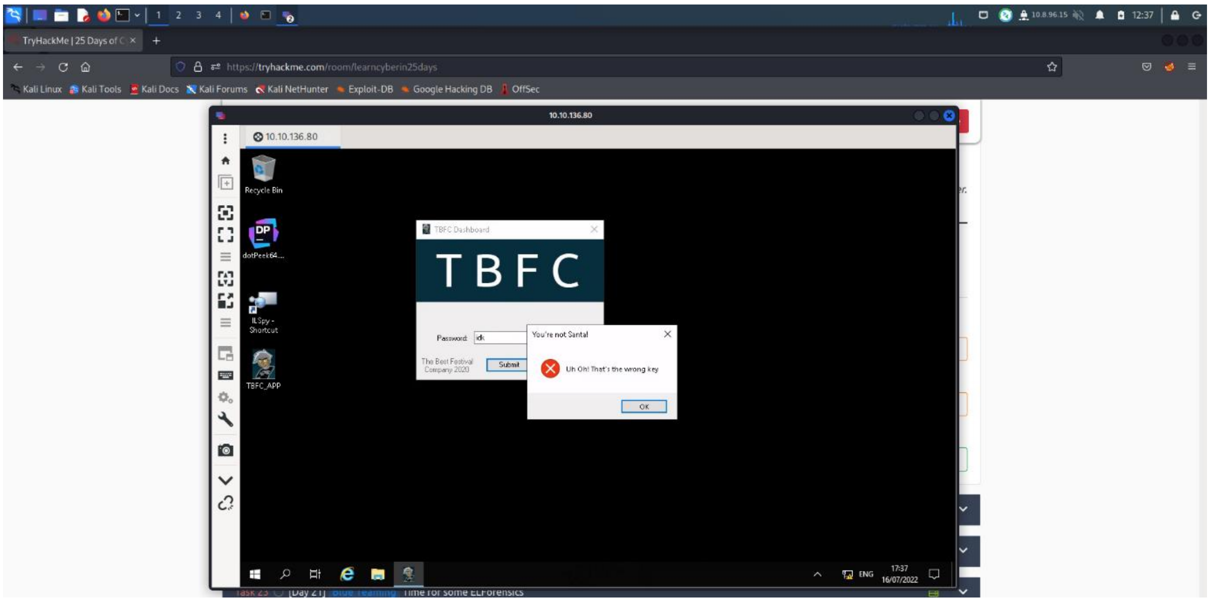
TBFC Dashboard

TBFC

Password: idk

The Best Festival Company 2020

The password that we had entered is wrong, so we received a message pop up. On the casing, it displayed the message “Uh Oh! That’s the wrong key”



TBFC Dashboard

TBFC

Password: idk

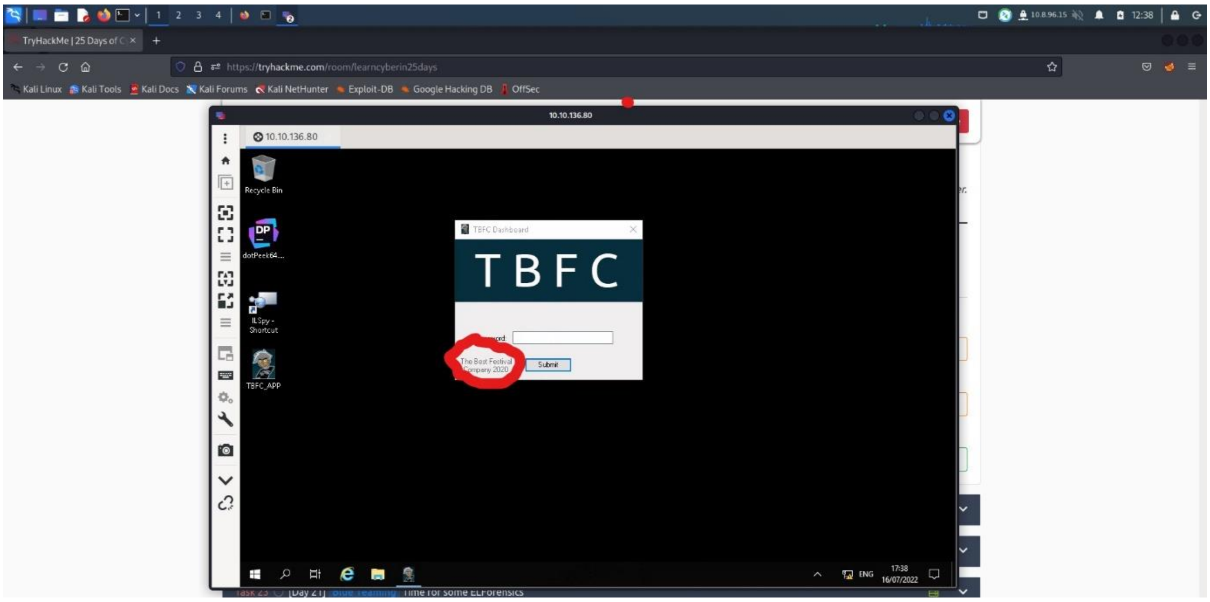
The Best Festival Company 2020

You're not Santa!

Uh Oh! That's the wrong key

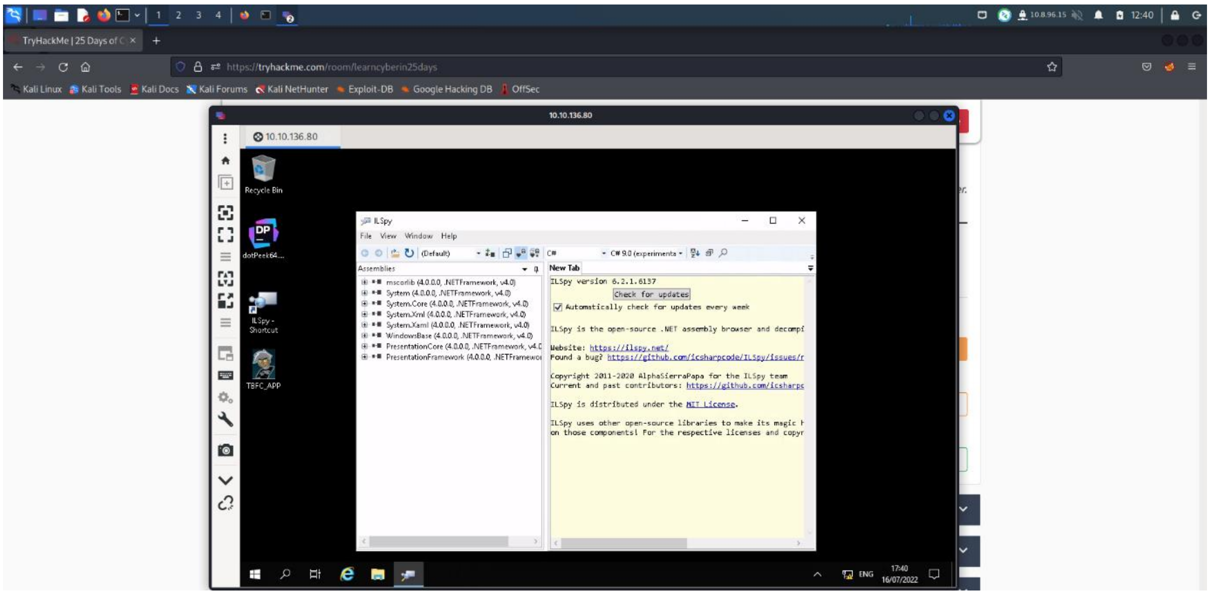
Question 2

TBFC can be seen to stand for “The Best Festival Company”. It is displayed in the casing of the application’s dashboard.



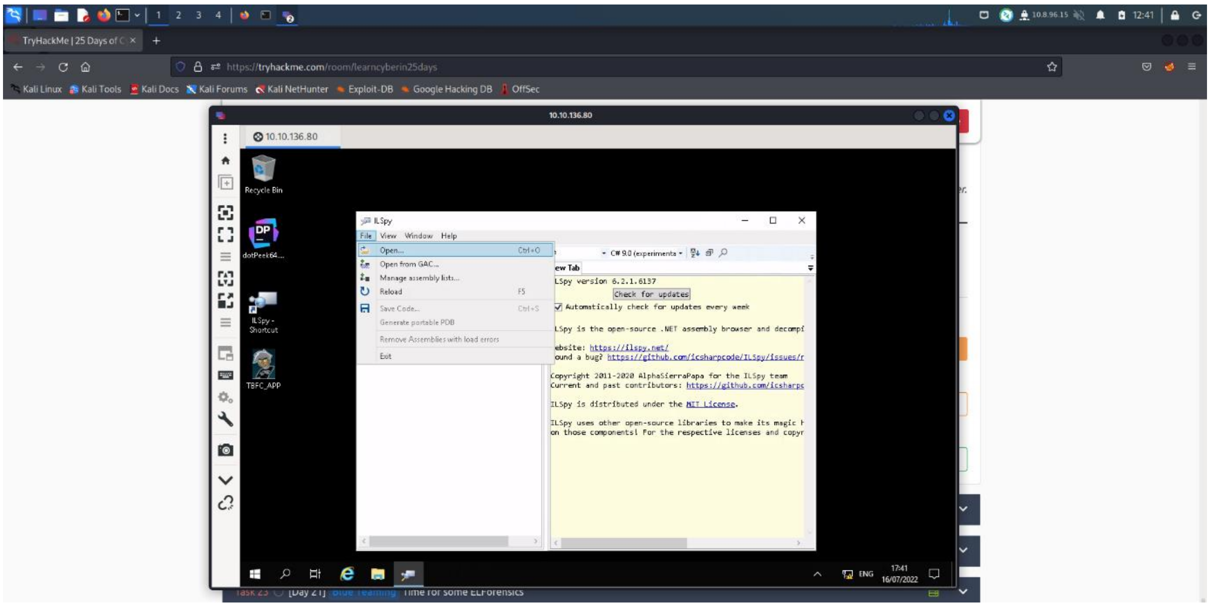
Question 3

We navigate to file and choose the TBFC_APP application in ILSpy to decompile it.

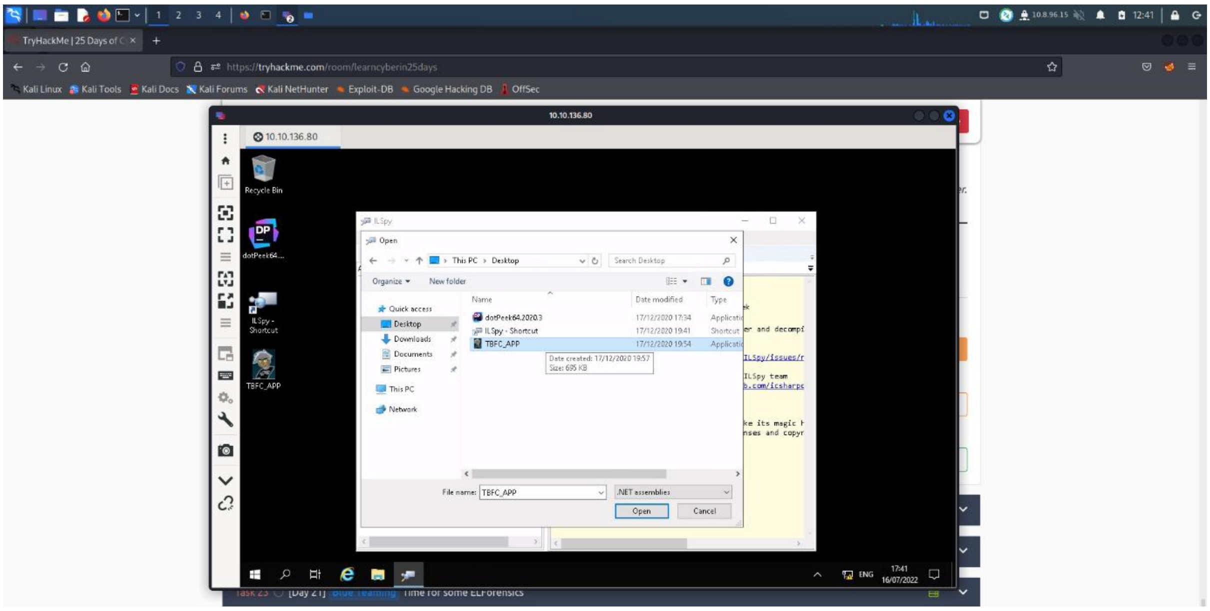


mscorlib (4.0.0.0, .NETFramework, v4.0)
System (4.0.0.0, .NETFramework, v4.0)
System.Core (4.0.0.0, .NETFramework, v4.0)
System.Xml (4.0.0.0, .NETFramework, v4.0)
System.Xaml (4.0.0.0, .NETFramework, v4.0)
WindowsBase (4.0.0.0, .NETFramework, v4.0)
PresentationCore (4.0.0.0, .NETFramework, v4.0
PresentationFramework (4.0.0.0, .NETFramewor

```
ILSpy version 6.2.1.6137
☑ Automatically check for updates every week

ILSpy is the open-source .NET assembly browser and decompi

Website: https://ilspy.net/
Found a bug? https://github.com/icsharpcode/ILSpy/issues/r

Copyright 2011-2020 AlphaSierraPapa for the ILSpy team
Current and past contributors: https://github.com/icsharpc

ILSpy is distributed under the MIT License.

ILSpy uses other open-source libraries to make its magic h
on those components! For the respective licenses and copyr
```



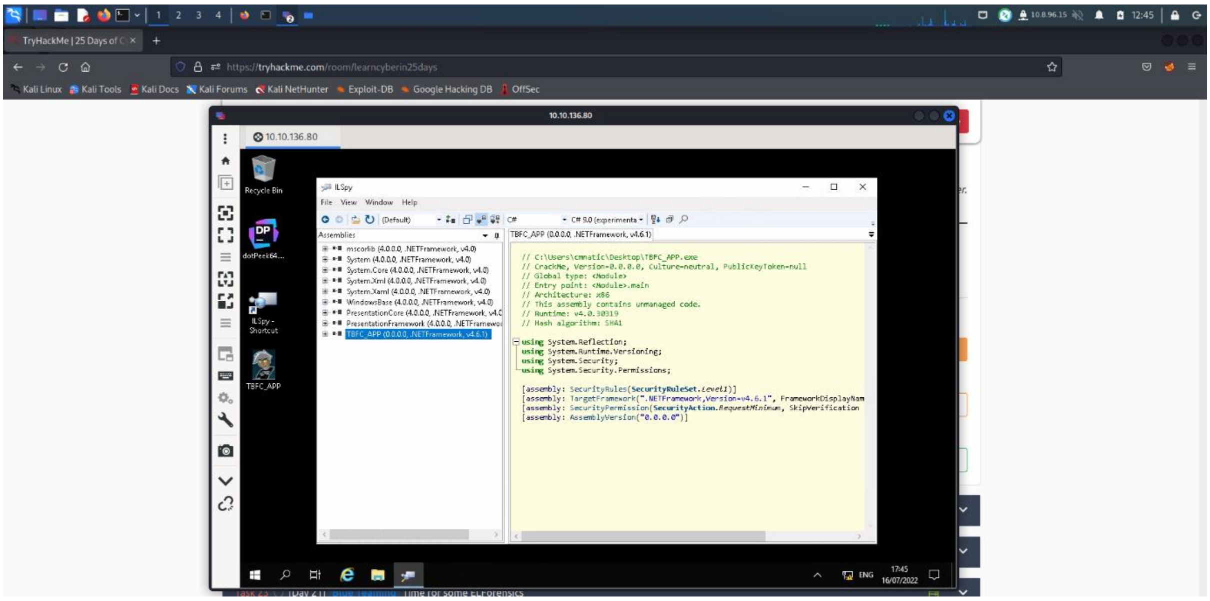
```
LSpy version 6.2.1.6137
                 Check for updates
☑ Automatically check for updates every week

LSpy is the open-source .NET assembly browser and decompi

ebsite: https://ilspy.net/
ound a bug? https://github.com/icsharpcode/ILSpy/issues/r

Copyright 2011-2020 AlphaSierraPapa for the ILSpy team
Current and past contributors: https://github.com/icsharpc

ILSpy is distributed under the MIT License.

ILSpy uses other open-source libraries to make its magic h
on those components! For the respective licenses and copyr
```



Name	Date modified	Type
dotPeek64.2020.3	17/12/2020 17:34	Applicati
ILSpy - Shortcut	17/12/2020 19:41	Shortcut
TBFC_APP	17/12/2020 19:54	Applicati

Date created: 17/12/2020 19:57
Size: 695 KB

File name: TBFC_APP
.NET assemblies

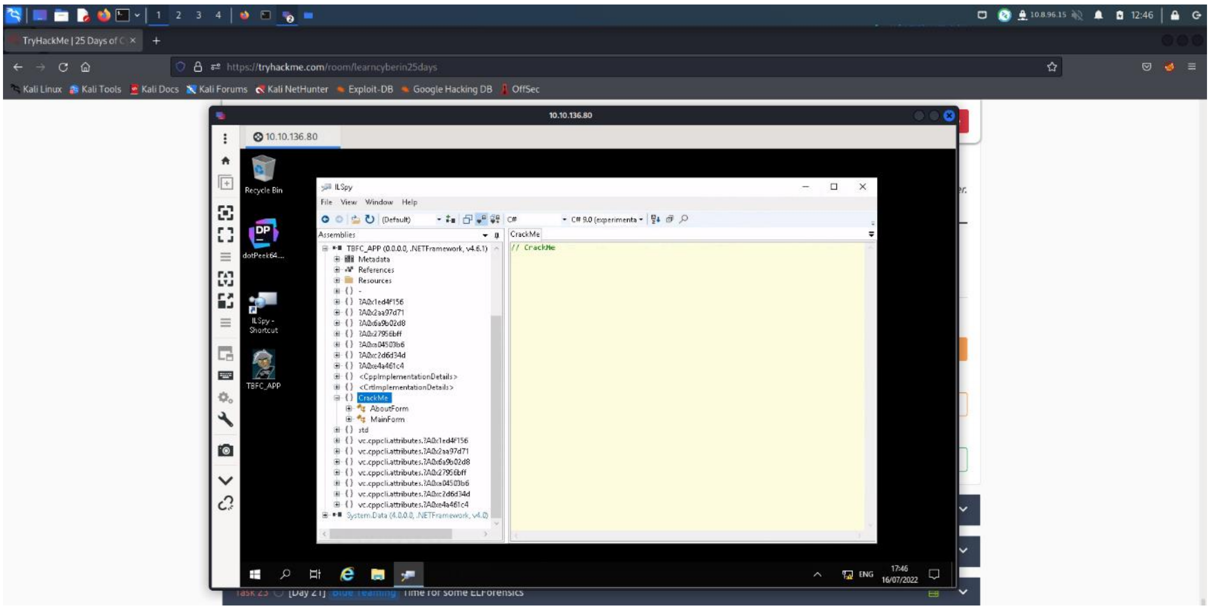


mscorlib (4.0.0.0, .NETFramework, v4.0)
System (4.0.0.0, .NETFramework, v4.0)
System.Core (4.0.0.0, .NETFramework, v4.0)
System.Xml (4.0.0.0, .NETFramework, v4.0)
System.Xaml (4.0.0.0, .NETFramework, v4.0)
WindowsBase (4.0.0.0, .NETFramework, v4.0)
PresentationCore (4.0.0.0, .NETFramework, v4.0
PresentationFramework (4.0.0.0, .NETFramewo
TBFC_APP (0.0.0.0, .NETFramework, v4.6.1)

TBFC_APP (0.0.0.0, .NETFramework, v4.6.1)

```
// C:\Users\cmnatic\Desktop\TBFC_APP.exe
// CrackMe, Version=0.0.0.0, Culture=neutral, PublicKeyToken=null
// Global type: <Module>
// Entry point: <Module>.main
// Architecture: x86
// This assembly contains unmanaged code.
// Runtime: v4.0.30319
// Hash algorithm: SHA1

using System.Reflection;
using System.Runtime.Versioning;
using System.Security;
using System.Security.Permissions;

[assembly: SecurityRules(SecurityRuleSet.Level1)]
[assembly: TargetFramework(".NETFramework,Version=v4.6.1", FrameworkDisplayNam
[assembly: SecurityPermission(SecurityAction.RequestMinimum, SkipVerification
[assembly: AssemblyVersion("0.0.0.0")]
```

Within it, we found a module that stood out to us which is the module “CrackMe”.

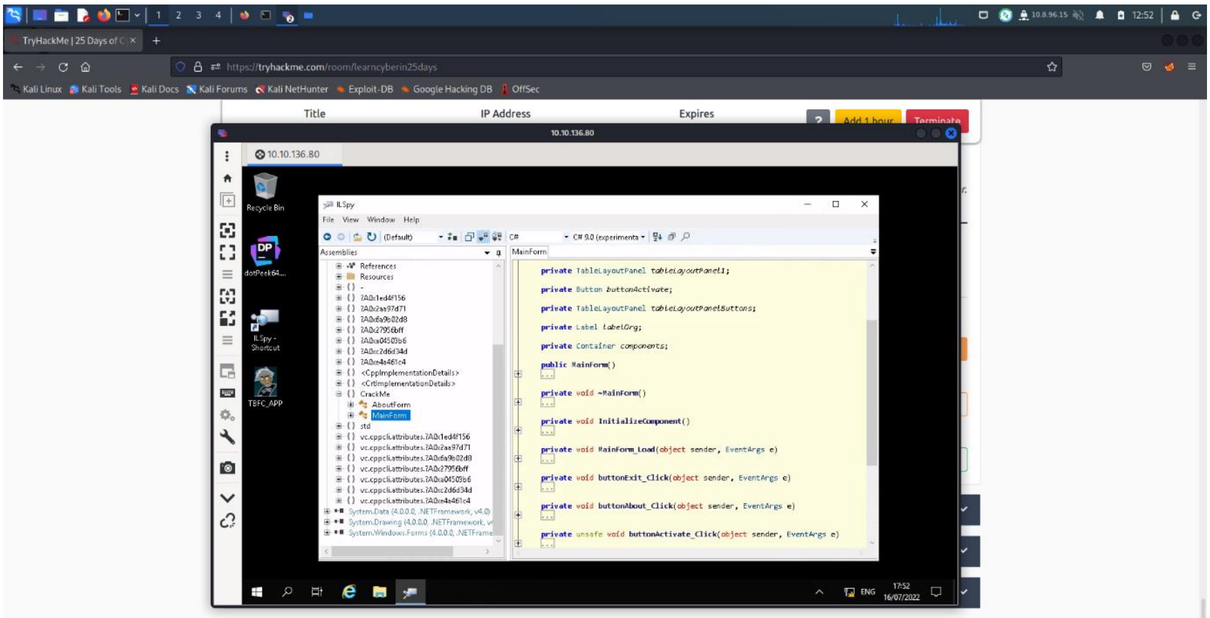


```
TBFC_APP (0.0.0.0, .NETFramework, v4.6.1)
    Metadata
    References
    Resources
    { } -
    { } ?A0x1ed4f156
    { } ?A0x2aa97d71
    { } ?A0x6a9b02d8
    { } ?A0x27956bff
    { } ?A0xa04503b6
    { } ?A0xc2d6d34d
    { } ?A0xe4a461c4
    { } <CppImplementationDetails>
    { } <CrtImplementationDetails>
    { } CrackMe
        AboutForm
        MainForm
    { } std
    { } vc.cppcli.attributes.?A0x1ed4f156
    { } vc.cppcli.attributes.?A0x2aa97d71
    { } vc.cppcli.attributes.?A0x6a9b02d8
    { } vc.cppcli.attributes.?A0x27956bff
    { } vc.cppcli.attributes.?A0xa04503b6
    { } vc.cppcli.attributes.?A0xc2d6d34d
    { } vc.cppcli.attributes.?A0xe4a461c4
System.Data (4.0.0.0, .NETFramework, v4.0)
```

```
// CrackMe
```

Question 4

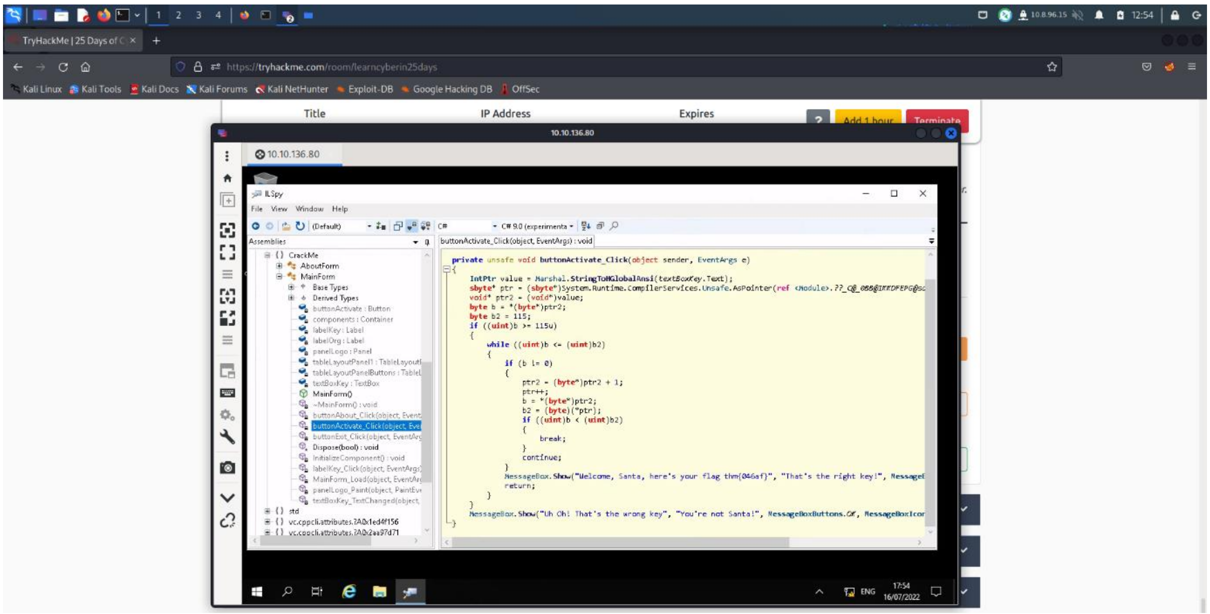
Within the “CrackMe” module, there are two forms which are “AboutForm” and “MainForm”. We checked both of the forms to find the information that we are looking for. After some observation, we found the information that we needed within “MainForm”.



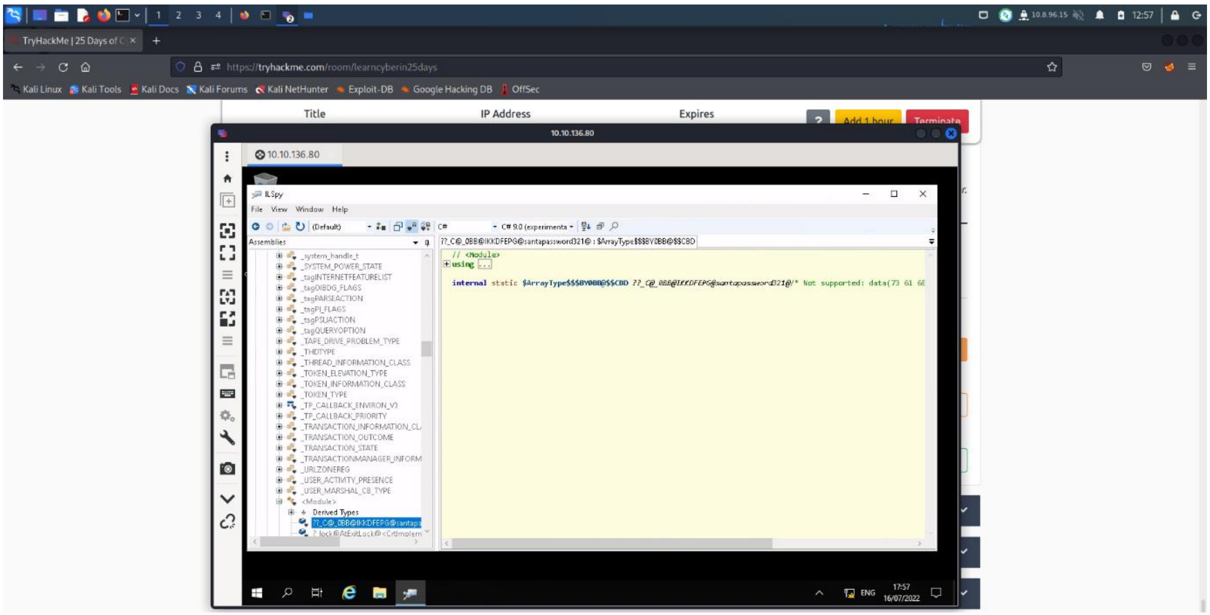
```
private TableLayoutPanel tableLayoutPanel1;

private Button buttonActivate;

private TableLayoutPanel tableLayoutPanelButtons;

private Label labelOrg;

private Container components;

public MainForm()
...

private void ~MainForm()
...

private void InitializeComponent()
...

private void MainForm_Load(object sender, EventArgs e)
...

private void buttonExit_Click(object sender, EventArgs e)
...

private void buttonAbout_Click(object sender, EventArgs e)
...

private unsafe void buttonActivate_Click(object sender, EventArgs e)
...
```

Question 5

After reading through the form, we found that the method “buttonActivate_Click” contain the information that we are seeking.

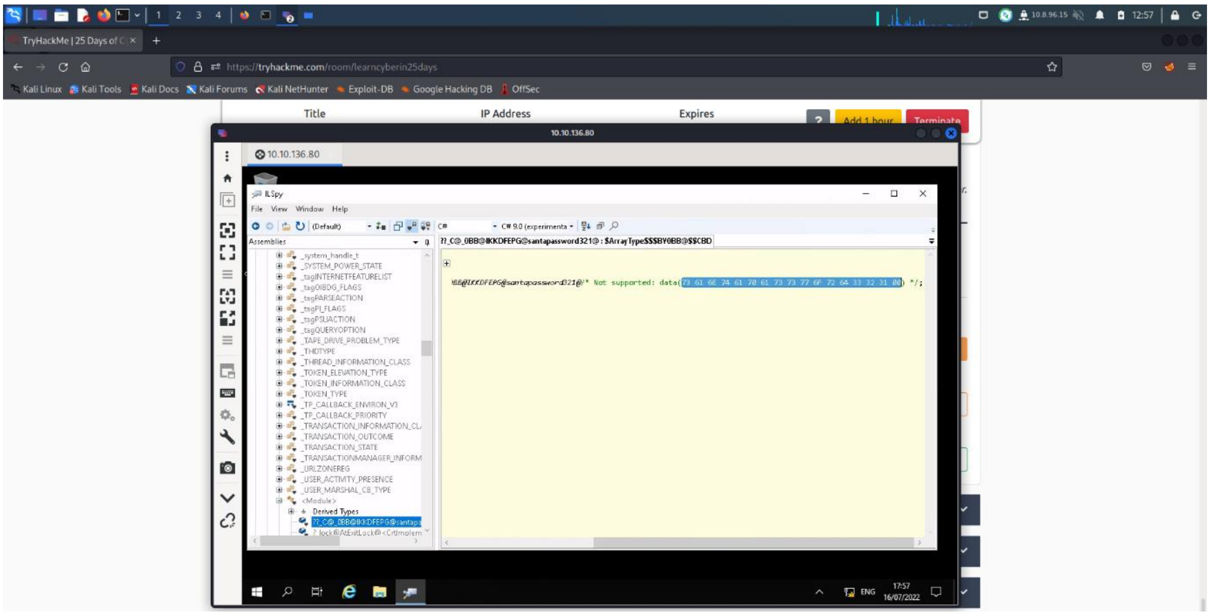


```
private unsafe void buttonActivate_Click(object sender, EventArgs e)
{
    IntPtr value = Marshal.StringToHGlobalAnsi(textBoxKey.Text);
    sbyte* ptr = (sbyte*)System.Runtime.CompilerServices.Unsafe.AsPointer(ref <Module>.??_C@_0BB@IKKDFEPG@sc
    void* ptr2 = (void*)value;
    byte b = *(byte*)ptr2;
    byte b2 = 115;
    if ((uint)b >= 115u)
    {
        while ((uint)b <= (uint)b2)
        {
            if (b != 0)
            {
                ptr2 = (byte*)ptr2 + 1;
                ptr++;
                b = *(byte*)ptr2;
                b2 = (byte)(*ptr);
                if ((uint)b < (uint)b2)
                {
                    break;
                }
                continue;
            }
            MessageBox.Show("Welcome, Santa, here's your flag thm{046af}", "That's the right key!", MessageE
            return;
        }
    }
    MessageBox.Show("Uh Oh! That's the wrong key", "You're not Santa!", MessageBoxButtons.OK, MessageBoxIcon
}
```



```
??_C@_0BB@IKKDFEPG@santapassword321@ : $ArrayType$$$BY0BB@$$CBD

// <Module>
using ...

internal static $ArrayType$$$BY0BB@$$CBD ??_C@_0BB@IKKDFEPG@santapassword321@/* Not supported: data(73 61 6E
```

Within it, we found a hexadecimal code that might be the credentials to Santa’s account.

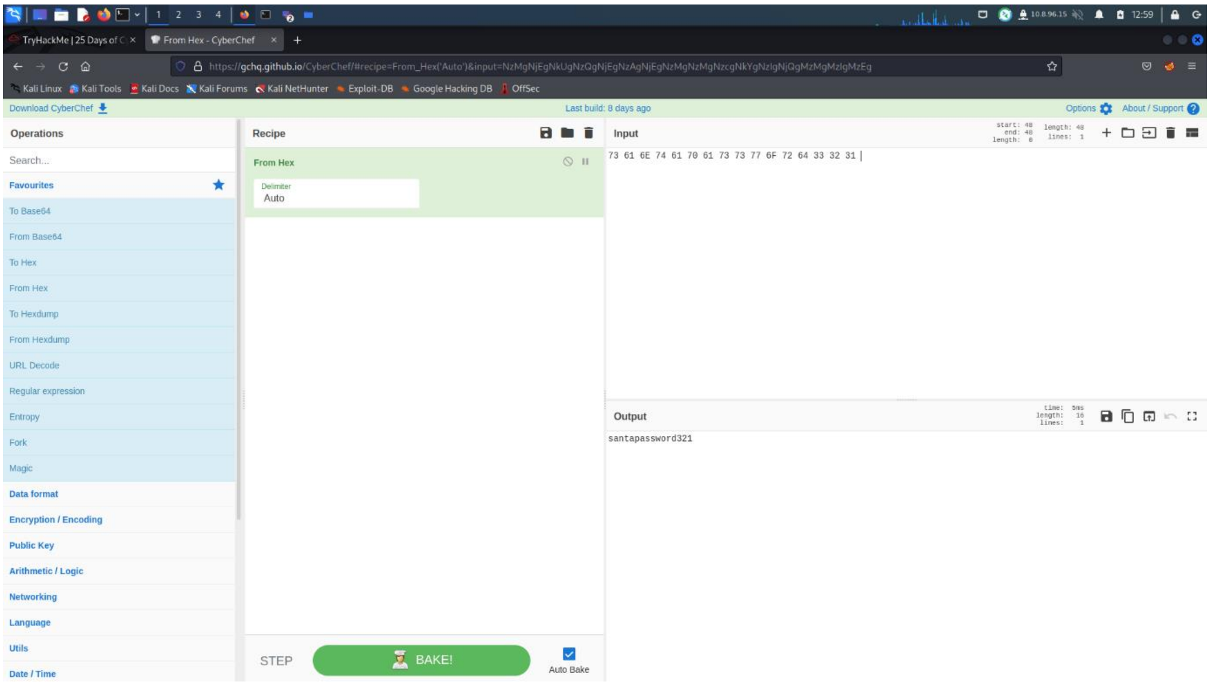


```
??_C@_0BB@IKKDFEPG@santapassword321@ : $ArrayType$$$BY0BB@$$CBD
```

```
0BB@IKKDFEPG@santapassword321@/* Not supported: data(73 61 6E 74 61 70 61 73 73 77 6F 72 64 33 32 31 00) */;
```

Question 6

We headed to CyberChef to decode the hexadecimal code into readable output. Once we decode the hexadecimal code, we found Santa's password which is "santapasssword321".



Recipe: From Hex

Delimiter: Auto

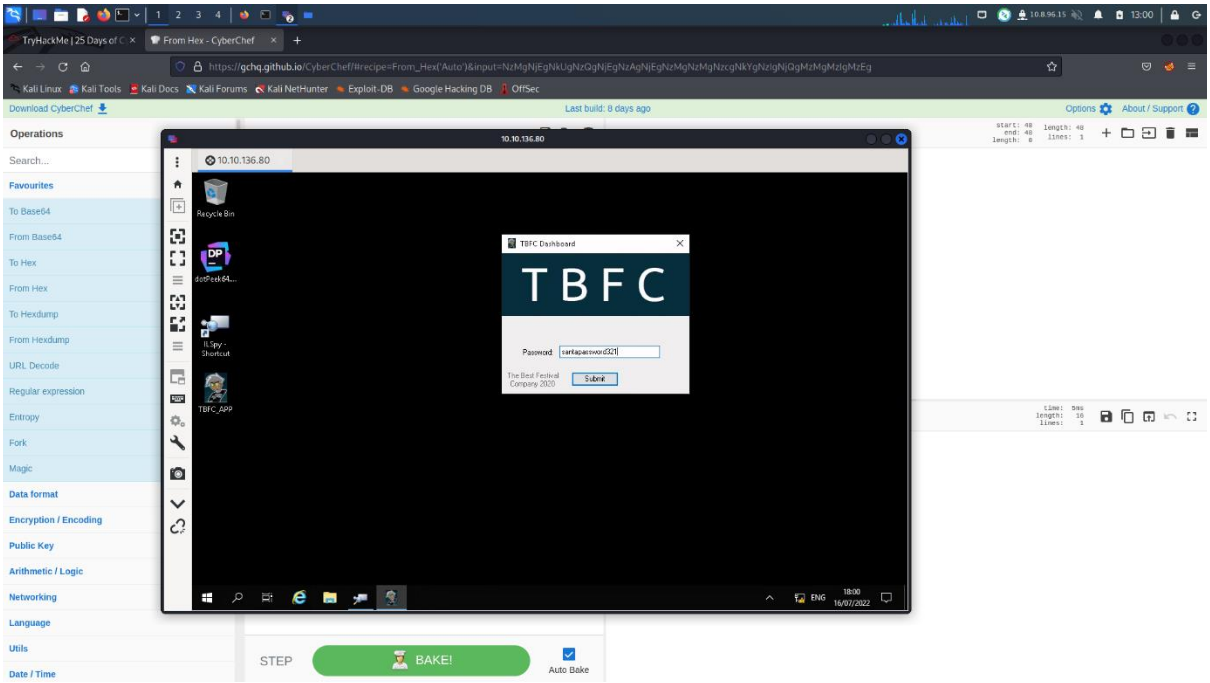
Input

```
73 61 6E 74 61 70 61 73 73 77 6F 72 64 33 32 31
```

Output

```
santapassword321
```

Now, having the credentials to access the TBFC application, we enter the password we have retrieved earlier.

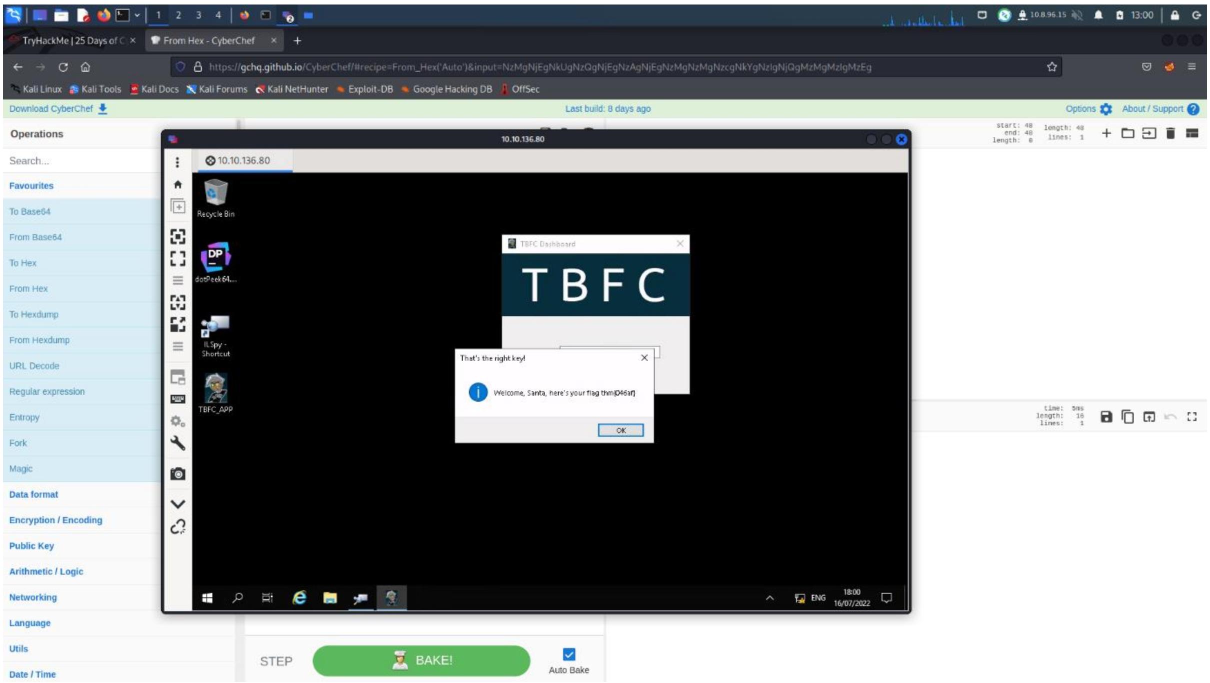


Password: santapassword321

The Best Festival Company 2020

Question 7

With the password we obtained, we logged into the application and voila! We found our flag.



That's the right key!

Welcome, Santa, here's your flag thm{046af}

Thought Process/Methodology:

In order to start the task, we first deployed the machine and log in using the Remote Desktop Protocol (RDP). We downloaded Remmina and with it, we connected to the instance with the credentials that was provided in the room. The IP address can be obtained upon launching Attackbox or connecting to Open VPN when using our own machine. The username is cmnatic and the password is Advertofcyber!.

Once we entered the credentials, we logged in. In the desktop, we launch TBFC_APP application and entered a random password. Of course, the password that we had entered was wrong. Hence, we received a message pop up. On the casing of the message, there were a message that reads “Uh Oh! That’s the wrong key”. Then, we decided to look around the application dashboard to get more information. On the casing of the application’s dashboard, we found what TBFC stands for which is “The Best Festival Company”. To gather even more information about the TBFC_APP application, we headed to ILSpy to decompile the application. In ILSpy, we navigate to file and open the TBFC_APP application. Within it, we found a module that immediately catches our attention which is the module “CrackMe”. We further inspect the module and discover two forms with it. Realizing that one of the forms could be holding information that we needed, we read through both of it. After some observation, we found the information that we needed within “MainForm”. In the “MainForm”, we stumbled upon the method “buttonActivate_Click” which contained the information that we needed which is a credentials that will gives us access to the application. In the method, there was a hexadecimal code. We deduced that this hexadecimal code is the key to finding the credentials. We browsed to Cyberchef to decode the hexadecimal code into a readable output and found what we believed to be Santa’s password. With the password that we have retrieved which is “santapassword321”, we headed to the TBFC application and entered the password into the provided space. Once we logged into the application, we found our flag.