New SectopRAT: Remote access malware utilizes second desktop to control browsers

gdatasoftware.com/blog/2019/11/35548-new-sectoprat-remote-access-malware-utilizes-second-desktop-to-control-browsers

This new remote access malware creates a second desktop that is invisible to the system's user. The threat actor can surf the Internet using the infected machine.

General appearance and obfuscation

SectopRAT is a .NET based remote access malware. The sample[1] was originally found by MalwareHunterTeam and announced in a tweet on 15. November 2019. It was compiled on 13. November 2019. Using the following Yara rule we were able to obtain a second sample[2] that was compiled on 14. November 2019 and submitted a day later to Virustotal.

```
rule SectopRat
{       meta:    author = "Karsten Hahn at G DATA CyberDefense AG"       strings:        $s_1 = "RemoteClient\x00"       $s_2 =
"InitHDesktop\x00"      $s_3 = "InitBrowser\x00"        $s_4 = "EnoghtSpace\x00"        $s_5 = "SPI_SETSCREENSAVEACTIVE\x00"
condition:      all of them and uint16(0) == 0x5A4D
}
```

The first sample[1] is signed by Sectigo RSA Code Signing CA, uses a Flash icon and has the following Version Information.

language ID: 0x0409
code page: 0x04B0
Comments: Idito PleasweN MinIMus Inc.
CompanyName: Nikler
LegalCopyright: Nikler
ProductName: Idito PleasweN MinIMus Inc.
FileVersion: 3.21.0005
ProductVersion: 3.21.0005
InternalName: Burataslop
OriginalFilename: Burataslop.exe

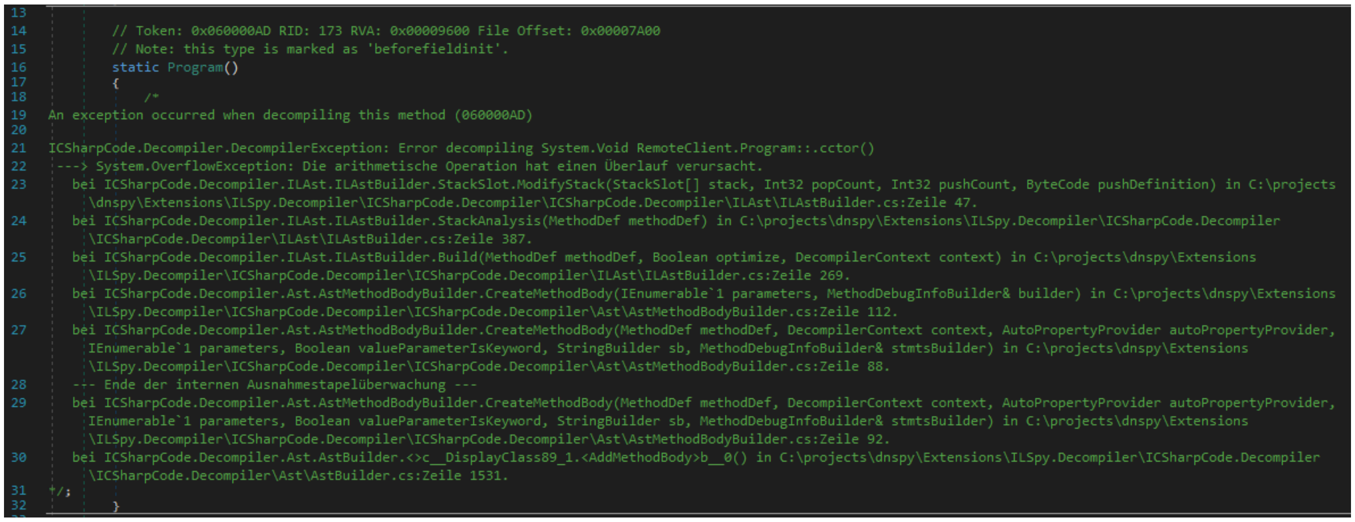
The second sample[2] is not signed and uses an icon that looks like a red floppy disk. The Version Information looks different too but follows a similar pattern of upper and lower case combinations in a jumble of arbitrary words.

language ID: 0x0409
code page: 0x04B0
Comments: errORs KilEfnos INCreASe MY Wife

CompanyName: LAkoRasen Kuscev MeaninG Jow
LegalCopyright: FAW ISir Polaris ComapNY
LegalTrademarks: investORS Leanda MikiRUck
ProductName: Colleti
FileVersion: 4.01.0009
ProductVersion: 4.01.0009
InternalName: Veerfus413
OriginalFilename: Veerfus413.exe

The first section of both samples has arbitrary characters for its name and has write and execute characteristics. The 5th and last section has no name and contains the entry point. The other sections look rather typical.

The threat actor used ConfuserEx to obfuscate the control flow and add anti-tamper to the .NET assembly. The anti-tamper prevents tools like DnSpy from decompiling the code (see picture below).



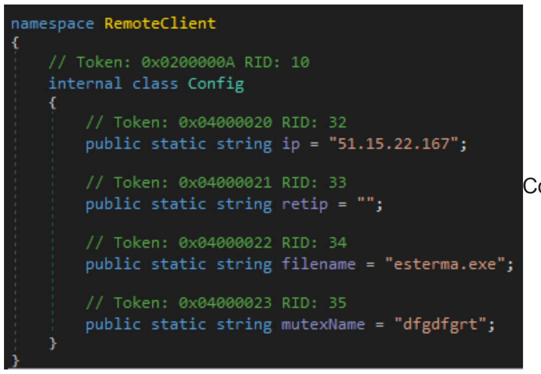
```
        // Token: 0x060000AD RID: 173 RVA: 0x00009600 File Offset: 0x00007A00
        // Note: this type is marked as 'beforefieldinit'.
        static Program()
        {
            /*
An exception occurred when decompiling this method (060000AD)

ICSharpCode.Decompiler.DecompilerException: Error decompiling System.Void RemoteClient.Program::.cctor()
 ---> System.OverflowException: Die arithmetische Operation hat einen Überlauf verursacht.
   bei ICSharpCode.Decompiler.ILAst.ILAstBuilder.StackSlot.ModifyStack(StackSlot[] stack, Int32 popCount, Int32 pushCount, ByteCode pushDefinition) in C:\projects
     \dnspy\Extensions\ILSpy.Decompiler\ICSharpCode.Decompiler\ICSharpCode.Decompiler\ILAst\ILAstBuilder.cs:Zeile 47.
   bei ICSharpCode.Decompiler.ILAst.ILAstBuilder.StackAnalysis(MethodDef methodDef) in C:\projects\dnspy\Extensions\ILSpy.Decompiler\ICSharpCode.Decompiler
     \ICSharpCode.Decompiler\ILAst\ILAstBuilder.cs:Zeile 387.
   bei ICSharpCode.Decompiler.ILAst.ILAstBuilder.Build(MethodDef methodDef, Boolean optimize, DecompilerContext context) in C:\projects\dnspy\Extensions
     \ILSpy.Decompiler\ICSharpCode.Decompiler\ICSharpCode.Decompiler\ILAst\ILAstBuilder.cs:Zeile 269.
   bei ICSharpCode.Decompiler.Ast.AstMethodBodyBuilder.CreateMethodBody(IEnumerable`1 parameters, MethodDebugInfoBuilder& builder) in C:\projects\dnspy\Extensions
     \ILSpy.Decompiler\ICSharpCode.Decompiler\ICSharpCode.Decompiler\Ast\AstMethodBodyBuilder.cs:Zeile 112.
   bei ICSharpCode.Decompiler.Ast.AstMethodBodyBuilder.CreateMethodBody(MethodDef methodDef, DecompilerContext context, AutoPropertyProvider autoPropertyProvider,
     IEnumerable`1 parameters, Boolean valueParameterIsKeyword, StringBuilder sb, MethodDebugInfoBuilder& stmtsBuilder) in C:\projects\dnspy\Extensions
     \ILSpy.Decompiler\ICSharpCode.Decompiler\ICSharpCode.Decompiler\Ast\AstMethodBodyBuilder.cs:Zeile 88.
   --- Ende der internen Ausnahmestapelüberwachung ---
   bei ICSharpCode.Decompiler.Ast.AstMethodBodyBuilder.CreateMethodBody(MethodDef methodDef, DecompilerContext context, AutoPropertyProvider autoPropertyProvider,
     IEnumerable`1 parameters, Boolean valueParameterIsKeyword, StringBuilder sb, MethodDebugInfoBuilder& stmtsBuilder) in C:\projects\dnspy\Extensions
     \ILSpy.Decompiler\ICSharpCode.Decompiler\ICSharpCode.Decompiler\Ast\AstMethodBodyBuilder.cs:Zeile 92.
   bei ICSharpCode.Decompiler.Ast.AstBuilder.<>c__DisplayClass89_1.<AddMethodBody>b__0() in C:\projects\dnspy\Extensions\ILSpy.Decompiler\ICSharpCode.Decompiler
     \ICSharpCode.Decompiler\Ast\AstBuilder.cs:Zeile 1531.
*/;
        }
```

ConfuserEx adds anti-tamper that leads to decompilation errors

Configuration and persistence

The assembly has a *RemoteClient.Config* class containing four public static variables for the configuration: *ip, retip, filename, mutexName*. The ip is the Command and Control (C&C) server IP. The *retip* variable is for setting a new C&C IP that the server can override by using the "set IP" command. The configuration is the same for both SectopRAT samples.



```
namespace RemoteClient
{
    // Token: 0x0200000A RID: 10
    internal class Config
    {
        // Token: 0x04000020 RID: 32
        public static string ip = "51.15.22.167";

        // Token: 0x04000021 RID: 33
        public static string retip = "";

        // Token: 0x04000022 RID: 34
        public static string filename = "esterma.exe";

        // Token: 0x04000023 RID: 35
        public static string mutexName = "dfgdfgrt";
    }
}
```

Config class in SectopRAT sample

The variables *filename* and *mutexName* are set but not used in the code. Instead it uses a hardcoded filename, *spoolsvc.exe*, and copies itself to *%LOCALAPPDATA%\Microsoft\*

Persistence is achieved by adding *spoolsvc.exe* to the RUN key in the registry.

```
HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\SFddg --> %LOCALAPPDATA%\Microsoft\spoolsvc.exe
```

Note that the legitimate Print Spooler Service system file by Microsoft is named *spoolsv.exe*

Command & control features

SectopRAT command and control operates with packet types which are distinguished by a certain byte value in every packet.

Byte value	Packet type

Byte value	Packet type
1	Start stream
2	Stop stream
3	Handle mouse event
4	Handle keyboard event
5	Init browser
6	Disconnect
7	Get codec info (not implemented)
8	Send computer info
9	Set IP

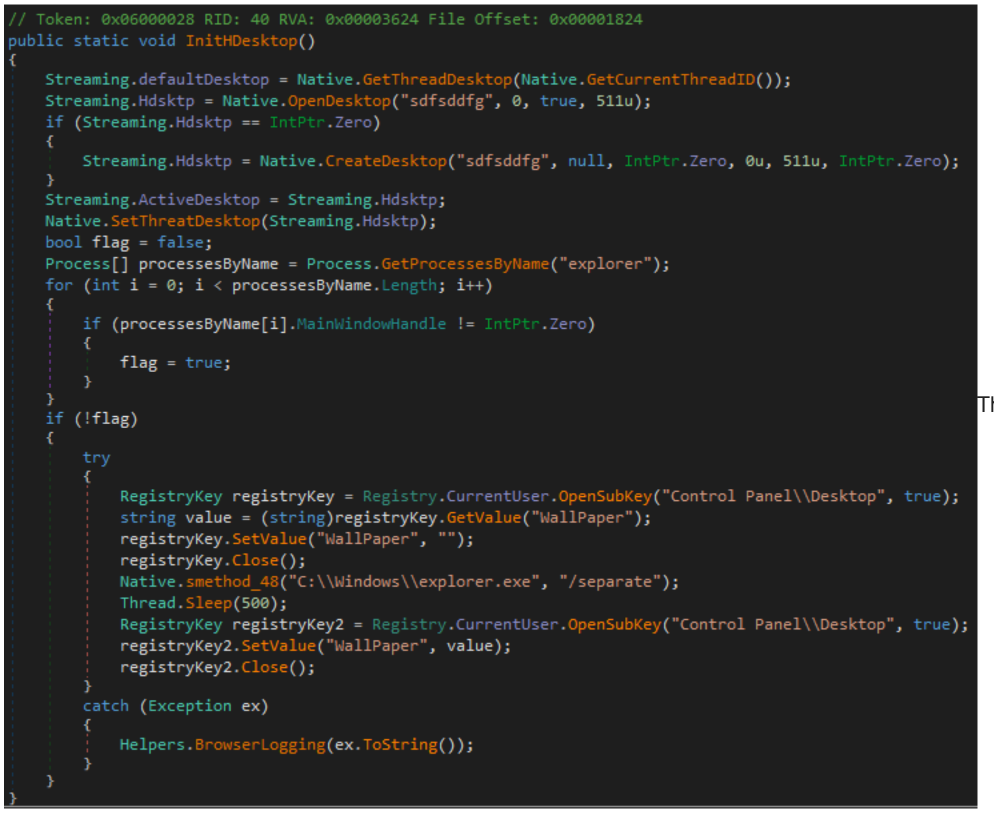
The "Start stream" packet will either stream the current desktop or create another desktop using the hardcoded desktop name "sdfsddfg". The second desktop is not visible to the person who sits in front of the infected computer. The threat actor however can use "Init browser" to surf the Internet via the second desktop on the infected system.

The "Init browser" packet has support for running Chrome, Firefox or Internet Explorer. It will change browser configuration, use start parameters and modify registry settings to disable security and make the browsers faster. E.g. for Chrome it disables sandboxes, the graphics cache and graphics options like 3d-apis, flash-3d, gpu-rasterization, gpu-vsync. The browser paths are hardcoded and don't use any environmental variables, which limits compatibility of the RAT.

The "Send computer info" packet will send the operating system name, the username, an id that is based on a randomly created path name and a hash value that is based on hardware information like the processor name, the number of cores and RAM.

"Get codec info" is not implemented yet. The client does not do anything.

"Set IP" makes it possible to change the IP of the C&C server.



```
// Token: 0x06000028 RID: 40 RVA: 0x00003624 File Offset: 0x00001824
public static void InitHDesktop()
{
    Streaming.defaultDesktop = Native.GetThreadDesktop(Native.GetCurrentThreadID());
    Streaming.Hdsktp = Native.OpenDesktop("sdfsddfg", 0, true, 511u);
    if (Streaming.Hdsktp == IntPtr.Zero)
    {
        Streaming.Hdsktp = Native.CreateDesktop("sdfsddfg", null, IntPtr.Zero, 0u, 511u, IntPtr.Zero);
    }
    Streaming.ActiveDesktop = Streaming.Hdsktp;
    Native.SetThreatDesktop(Streaming.Hdsktp);
    bool flag = false;
    Process[] processesByName = Process.GetProcessesByName("explorer");
    for (int i = 0; i < processesByName.Length; i++)
    {
        if (processesByName[i].MainWindowHandle != IntPtr.Zero)
        {
            flag = true;
        }
    }
    if (!flag)
    {
        try
        {
            RegistryKey registryKey = Registry.CurrentUser.OpenSubKey("Control Panel\\Desktop", true);
            string value = (string)registryKey.GetValue("WallPaper");
            registryKey.SetValue("WallPaper", "");
            registryKey.Close();
            Native.smethod_48("C:\\Windows\\explorer.exe", "/separate");
            Thread.Sleep(500);
            RegistryKey registryKey2 = Registry.CurrentUser.OpenSubKey("Control Panel\\Desktop", true);
            registryKey2.SetValue("WallPaper", value);
            registryKey2.Close();
        }
        catch (Exception ex)
        {
            Helpers.BrowserLogging(ex.ToString());
        }
    }
}
```

This code creates another desktop on the infected system.

Unfinished but we will probably see more

SectopRAT is used in the wild but still looks unfinished and in parts hastily done. Some of the class names and also the name of the second desktop look like they were produced while trying to type arbitrarily on the keyboard because the keys are right next to each other and repeated by finger motion.

Despite obvious flaws like using hardcoded paths without environmental variables to access system files, the RAT's architecture, the use of a second desktop and changes in browser configuration files and parameters show some internal knowledge that is far from a greenhorn. It is quite possible that the first samples in the wild are merely for testing. We expect to see new versions with additional features in the future.

Indicators of Compromise

Description	Filenames	SHA256	URL
[1] SectopRAT	Burataslop.exe blad.exe	b1e3b5de12f785c45d5ea3fc64412ce640a42652b4749cf73911029041468e3a	hxxp://45.142.213.230/blad.
Deobfuscated SectopRAT [1]		4409d2170aa9989c6a8dd32b617c51a7c3e328b3c86410813c016691b2bd7774	
[2] SectopRAT	Veerfus413.exe bssd.exe	d5a3d47e1945e9d83a74a96f02a0751abd00078ee62e6d3a546a050e0db10d93	hxxp://45.142.213.230/bssd

Karsten Hahn
Malware Analyst

- You are here:
- Blog (EN)
- Remote access malware utilizes second desktop to control browsers