

Zebrocy's Multilanguage Malware Salad

SL securelist.com/zebrocys-multilanguage-malware-salad/90680

By GReAT

Zebrocy is Russian speaking APT that presents a strange set of stripes. To keep things simple, there are three things to know about Zebrocy

- Zebrocy is an active sub-group of victim profiling and access specialists
- Zebrocy maintains a lineage back through 2013, sharing malware artefacts and similarities with BlackEnergy
- The past five years of Zebrocy infrastructure, malware set, and targeting have similarities and overlaps with both the Sofacy and BlackEnergy APTs, yet throughout that time it has remained different from both of those groups

We originally described a rare “Zebrocy Delphi payload” in late 2015 private reports. That malware set, activity, and infrastructure has greatly expanded. Its malware set has been coded in a half dozen languages or so. Related activity has gone on for years, spearphished hundreds of government, foreign affairs related, and military related targets, and initially was regarded as a Sofacy subset.

Essentially, in our SAS2019 presentation “Zebrocy's Multilanguage Malware Salad”, we publicly provided for the first time original insights on Zebrocy and its characteristics, based on five years of research and private reports on this group:

- Game is on – a small new Zebrocy spearphish wave with a new Golang downloader variant was sent out the week before SAS2019
- Consistent profiling and process enumeration reporting behavior has been redeveloped and redeployed in Zebrocy backdoors across five+ years
- Multiple bespoke second stage implants perform credential harvesting based on stage one process enumeration
- Copy-and-paste coding tendencies
- Complex overlaps with Sofacy and BlackEnergy/GreyEnergy over five years, suggesting a supportive role as a sub-group
- Initial malware development and deployment lineage with BlackEnergy stretches back to 2013

While Zebrocy has never presented 0day exploits, this group's lineage comes from not-so-humble BlackEnergy and Sofacy beginnings. The group presents an agile and capable malware set, its spearphishing and intrusion activity surged in volume in 2018, and their efficacy requires network defenders' attention.

A Dish before the Main Course

- **RU speaking APT - profiling and access specialists**
- **Different from past Sofacy and BlackEnergy (BE)**
- **Roots and ongoing overlap with Sofacy and BE**



Zebrocy shares data points and crosses lines with other clusters of activity in unique and unexpected ways. Zebrocy initially shared limited infrastructure, targets, and interests with Sofacy. Zebrocy also shared malware code with past BlackEnergy/Sandworm; and targeting, and later very limited infrastructure with more recent BlackEnergy/GreyEnergy. Oddly, Turla deployed spearphish macros almost identical to previous, non-public Zebrocy code in 2018.

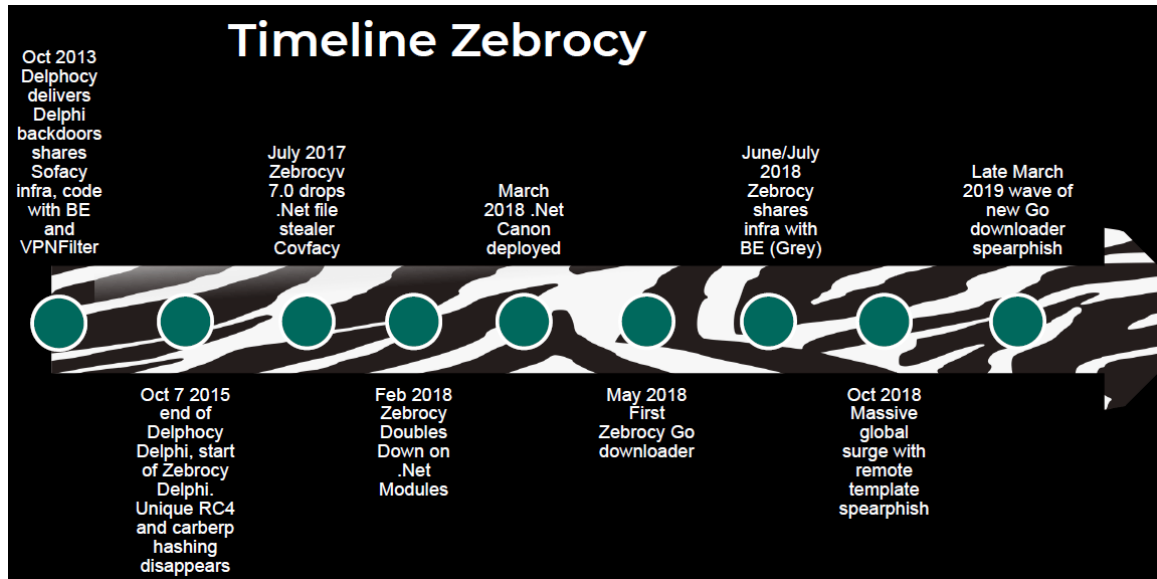
It's fantastic to see some of these same points being repeated publicly by other research teams. A previous claim that Zebrocy distributed Sofacy's XAgent as a second stage implant remains unsubstantiated but now is replaced with findings identical to these following the SAS2019 presentation, so it seems we are all slowly getting on the same page.

A first course with new additions

When we originally documented a Zebrocy malware incident in late 2015, we noted an Oct 2015 AutoIT downloader and a Delphi backdoor payload. Since then, we have noted a virtual salad of Zebrocy code tossed together, built with a handful of languages, often ripped from various code sharing sites. Zebrocy activity initiates with spearphishing operations delivering various target profilers and downloaders without the use of any 0day exploits. Browser credential theft, keylogging, and Windows credential theft, along with some incidents of file and communications theft, are all on the list of Zebrocy second stage implant specials.

This Zebrocy dish is served before the main course – gaining and maintaining access is not an easy job. And, because the group seems to maintain lineage in both the 0day capable and destructive BlackEnergy/Sandworm APT and the prolific and 0day capable Sofacy APT, this course is very interesting. Let's take a more intelligent perspective on the Zebrocy malware set and activity and its lineage, based on reporting provided to our private report customers covering the past five years.

It's important to note that Zebrocy-related spearphishing activity continued into April 2019, in the week prior to SAS2019. Recent changes to its Go downloader variant make it clear that the Zebrocy malware set is still under active development. And observed



A set of Zebrocy related events best characterize years of the activity and help to carve out the group's own profile, its lineage, malware set, infrastructure, and modus operandi.

- Zebrocy lineage – early Sofacy infrastructure overlap (late 2015/early 2016) for the Zebrocy Delphi backdoor
- Zebrocy lineage – Delphocy Delphi deployment and abrupt conclusion (2013 – late 2015), and start of Zebrocy Delphi timeline (late 2015)
- Zebrocy lineage – shared, unique kernel code between BlackEnergy and Delphocy bootkit (2013 – 2015)
- Zebrocy unique malware set – vintage Delphi programming coupled with unusual and agile development capabilities with new managed languages like Python, C#, and Go all perform screengrab anchor, volume serial number id, systeminfo and process list collection
- Zebrocy ongoing targeting and infrastructure overlap – fairly recent [BlackEnergy/GreyEnergy](#).
- Zebrocy matching spearphish macro artefact overlap – recent Turla

When we first reported on the Zebrocy package in early October 2015, this activity was not quite Sofacy and not quite BlackEnergy. Zebrocy spearphished diplomatic targets with a compiled AutoIT script that downloaded a new Delphi payload. This payload was unusual, as only the Delphocy backdoors and Madi backdoors from 2012 were well known APT related Delphi malware – Delphi programming itself is an unusually older skill. These malware artefacts, and campaign events suggested ties in the malware to BlackEnergy, while overlap in infrastructure suggested ties to Sofacy.

SAS2018 Claims and Predictions

Last year's SAS2018 "[Masha and these Bears](#)" presentation focused on SPLM/XAgent Sofacy activity, but included mention of [Zebrocy](#) due to targeting overlap and shared interests. It was first to record and predict several items:

The limited set of 2013-2015 Delphocy intrusions in Ukraine and Poland deployed a Delphi backdoor both with and without a bootkit loader. This bootkit loader included a routine that shares the same compiled code with only the BlackEnergy kernel loaders, helping to tie Zebrocy malware to the BlackEnergy malware set.

This unique encryption implementation was shared between BlackEnergy's kernel loader, and Delphocy's bootkit kernel loader code. The appearance of this code overlap coincides with several project events:

- End of Delphocy/BlackEnergy overlapped code use, while BlackEnergy moved forward with other code
- End of Delphocy's user-mode Delphi payload (October 2015)
- Start of Zebrocy's Delphi payload (October 2015)

A particular chunk of kernel mode code for a custom encryption routine was shared across the older Delphocy bootkit and the BlackEnergy malware platform in 2013. While Delphocy replaced this bootkit with a simplified user-mode persistence technique, BlackEnergy malware continued using this code until late 2015. Then, these APTs discontinued both the Delphi-based Delphocy project and the use of this mysterious chunk of code within BlackEnergy malware. Almost immediately, Delphi-based Zebrocy backdoors began to be deployed. Several months later, a Zebrocy backdoor connected back to a domain that was registered by a particular email address. This address had been used to register another Sofacy domain hosted on a well-known Sofacy IP at the time (rammatica[.]com/raveston[.]com).

```
bVar1 = *(byte *) (param_1 + 0x100);
cVar2 = *(char *) ((ulonglong)bVar1 + param_1);
bVar4 = *(char *) (param_1 + 0x101) + cVar2;
cVar3 = *(char *) ((ulonglong)bVar4 + param_1);
*(char *) ((ulonglong)bVar1 + param_1) = cVar3;
*(char *) ((ulonglong)bVar4 + param_1) = cVar2;
*param_2 = *param_2 ^ *(byte *) ((ulonglong)(byte)(cVar2 + cVar3) + param_1);
*(char *) (param_1 + 0x100) = bVar1 + 1;
*(byte *) (param_1 + 0x101) = bVar4;
return;
```

Note that both Delphocy's and BlackEnergy's kernel mode code appropriated unique content in 2013 from the Carberp codebase – hashing, injection, bootkit functionality. Surprisingly, this same unique encryption cipher was seen pasted again into 2018 VPNFilter code as well. Clearly it happens with other malware, but Zebrocy's consistent copy/paste tendency is something not frequently seen in other APT malware with a "best use" date spanning five years or more. Portions of its AutoIT code were copied from code sharing forums and pasted into their own code. This is different from Sofacy's disappeared and exhaustive SPLM/XAgent codebase. It was used for at least six years and was entirely custom-built.

Zebrocy's mix

The Zebrocy malware set is tossed together from a wide set of languages and technologies, including both legitimate and malicious code shared on online forums and sites like Github and Pastebin. This repeated “copy/paste” practice is not frequently seen in Russian speaking APT malware sets, although open source and penetration testing/red teaming malware are frequently used by other groups, like Empire, Responder, BeEF, and Mimikatz. Also unusual, this Zebrocy malware assortment is frequently rebuilt on multiple languages, along with new malware components added to the mix.

- AutoIT
- Delphi
- C#
- PowerShell
- Go
- Python
- Nim

AutoIT component

The group emailed zipped AutoIT attachments to at least a dozen targets in the first wave of Zebrocy spearphishing in early October 2015. The group is fairly successful in convincing recipients to open the attachments; generally around half of recipients attempted to open them. The AutoIT code is a target profiling component and downloader. It appears to have been developed with copy and paste coding talent from AutoIT scripting forums, Github, etc, tossed together from multiple sources. Here are a few:

- [AutoIT scripting forums](#)
- [Github](#)
- [Pastebin](#)

The AutoIT executable contains over 60 functions listed in the script, with 10 of the most interesting functions listed here. The same initial profiling functionality is maintained almost four years later in various newer Go and C# downloader variants, with screengrabs, system info and running process collection built into the startup routine of each one. Unique victim ID's, for example, are based on target systems' volume serial number and collected and reported to its C2 in its initial connectback for ongoing tracking.

FUNC Name	Detail
SCREENCAPTURE	user32.getdesktopwindow -> capture screenshot
SCREENCAPTURE_SAVETOJPG	save screenshot to C:\Users\xxx\AppData\Roaming\Desktop.jpg
FILESEARCH	recursive and selective extension based file search
SCREEN_SI	generates base64 encoded string of screengrab desktop.jpg, systeminfo output, processlist output for POST
SI (SystemInfo)	cmd /c SYSTEMINFO & PROCESSLIST
CMD	cmd /c <anything>
POST	set up persistence with registry run key write, fetch volume serial number, build URL with serial number and SCREEN_SI output, POST to hardcoded server using winhttp.winhttprequest.5.1
UPLOAD	open file, read contents, POST binary string of file contents to hardcoded c2 as form data

C# Zebrocy backdoor

Zebrocy pushed a C# backdoor that maintains much the same functionality as its other assortment of backdoor implementations.

A C# Zebrocy

- C# Zebrocy payload
- August 2018 - GetVolumeInformation variant
- September 2018 - WMIC variant

```

try
{
    Process process = new Process();
    process.StartInfo.UseShellExecute = false;
    process.StartInfo.CreateNoWindow = true;
    process.StartInfo.RedirectStandardOutput = true;
    process.StartInfo.FileName = "systeminfo";
    process.Start();
    text2 += process.StandardOutput.ReadToEnd().Replace("
", "
");
    process.WaitForExit();
}
}

try
{
    Process process2 = new Process();
    process2.StartInfo.UseShellExecute = false;
    process2.StartInfo.CreateNoWindow = true;
    process2.StartInfo.RedirectStandardOutput = true;
    process2.StartInfo.FileName = "tasklist";
    process2.Start();
}
}

```

```

// Token: 0x0600000A RID: 10 RVA: 0x0002180 File Offset: 0x0000380
private string screen()
{
    string result = "";
    try
    {
        Image image = new Bitmap(Screen.PrimaryScreen.Bounds.Width,
Screen.Bounds.Height, PixelFormat.Format32bppArgb);
        Graphics graphics = Graphics.FromImage(image);
        graphics.CopyFromScreen(Screen.PrimaryScreen.Bounds.X, Screen.PrimaryScreen.Bounds.Y, 0,
Screen.Bounds.Size, CopyPixelOperation.SourceCopy);
        MemoryStream memoryStream = new MemoryStream();
        image.Save(memoryStream, ImageFormat.Jpeg);
        result = BitConverter.ToString(memoryStream.ToArray()).Replace("-", string.Empty);
    }
    catch
    {
    }
}

```

Security Analyst Summit 2019 13

Most interesting in this implementation is its consistent collection of screengrab and system information, and a list of running processes. Again, with this first stage backdoor, it is profiling its targets and looking for unexpected sources of credential collection to develop bespoke second stage credential harvesters against. Additionally, Zebrocy wheeled out a [Cannon backdoor](#) built on C# throughout 2018.

Delphi backdoor

The Zebrocy Delphi backdoor has been publicly documented fairly well, but its artefacts may have a more interesting part in the Zebrocy story. Because of the unusual ongoing presence of Delphi backdoors in its malware set, early Zebrocy roots appear to be

planted in Delphocy malware deployed to Ukraine and Poland targets from early 2013 to late 2015. Also interesting is the unique encryption cipher implemented in both the 2013 Delphocy bootkit kernel mode loader and the BlackEnergy kernel modules, along with simple API hashing functions that both shared with Carberp. While Delphocy was deployed for a second version without the bootkit in 2015, the BlackEnergy platform continued to deploy with the shared kernel loader code until late 2015.

It does not seem to be merely coincidental that the very last of the BlackEnergy kernel mode components sharing this code with the Delphocy-related kernel loaders wilted at the very end of 2015 along with the Delphocy campaigns altogether. This came after the Delphocy project delivered a second round of Delphi variants without the bootkit loader. Meanwhile, Zebrocy Delphi variants grew from the initial row of Zebrocy Delphi implants in October 2015. While it is a loose connection, it seems more than just coincidental to observe unusual Delphi coding end with one activity set and begin with another activity set.


Go

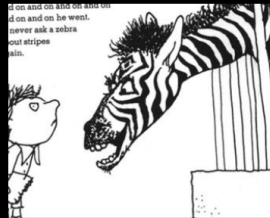
The group's Go backdoors continued to be modified and deployed into late March 2019. These backdoors also include a large amount of code included from external sources. They also include the screengrab and system information collection, along with running process enumeration in order to profile targets and further inform their second stage credential harvesting efforts. Zebrocy Go backdoors have been sent out in waves over the past year, maintaining a variety of project strings seen below. Vitaly Kremez often has concise, [interesting tweets](#) highlighting some of the changes in this Go downloader code, and [deeper analysis](#) on related Zebrocy malware.

Fresh Zebrocy

Multiple Go variants over time

- March 21, 2019 - C:/!Dev/Spite/main.go
- Dec 20, 2018 - C:/!Project/C1/ProjectC1Dec/main.go
- Dec 04, 2018 - C:/!Project/C2/Project3_I_HEX/main.go
- Nov 26, 2018 -
C:/Work/prog/Windows_autoIT_WMVare_works/GoLand/Project3_I_HEX/main.go
- Jun 18, 2018 - C:/!=/GoLand/Project3_I/main.go
- May 10, 2018 - C:/!=/GoLand/Project1_HEX/Project2.go

 Security Analyst Summit 2019



"A Light in the Attic", Shel Silverstein 8

A second stage

These findings were particularly interesting in the light of past claims about SPLM/XAgent being the second stage of choice for Zebrocy, for which there was a lot of monitoring on our part, but never any data support. Some guesses were made about

why that was, perhaps Zebrocy downloaders were all mitigated prior to attempting to download further stages? But never any answers.

Instead, we arrived at the answers ourselves. In order to account for unexpected software installations at victim systems, no matter which language, each first stage backdoor implementation collects a “system information” listing, screengrab, and enumerates running processes. This malware behavior was included in Zebrocy backdoors from the very first backdoor that we reported on, and continued into 2019 with the latest rounds of Go backdoors. After collected information is POSTed to the C2, a long delay ensues. Eventually, target systems may receive a custom built second stage implant to retrieve credentials from those unexpected software sources. More unusual software packages included little-known customized Chromium builds like CentBrowser and 7Star from Asian studios. In some cases, malware password stealers are deployed to address more common software.

Recency - Second Stage Implants

- **Initial process listing POST – what creds did we miss?**
- **Browser credentials**
 - **CentBrowser, 7Star, Chromium, Opera, Yandex, Chrome**
- **Email client**
 - **Outlook**
- **Keystrokes and files**
 - **C# removable drive file content stealer**
 - **C# keylogger**

In addition, Zebrocy file content stealers and keyloggers coded in C# were detected at targets in 2017 and 2018. Some of this code and their build id value format was reviewed in the SAS2018 “Masha and these Bears” presentation.

Served cold

Zebrocy version 2.2 called back to a domain sharing Whois and hosting resources with Sofacy in early 2016, and later versions used naming and URL constructs very similar to BlackEnergy resources. And since then, just like BlackEnergy, mostly all of the Zebrocy C2 used no domain registrations. Communications directly to the host over IPv4 with no domain resolution are common behavior for the group’s malware. However, every now and then, Zebrocy malware calls back to servers located by hardcoded domain names.

Its ongoing activity demonstrates a long game commitment to gaining access to targeted networks. And as we predicted at SAS2018 and SAS2019, this latest new Nim coding adds to the growing list of languages for this malware set. We will see more from Zebrocy into 2019 on government and military related organizations.

