State-sponsored Attack Groups Capitalise on Russia-Ukraine War for Cyber Espionage

By alexandrag

Published: 2022-03-31 · Archived: 2026-04-07 15:24:19 UTC

Introduction

Geopolitical tensions often make headlines and present a golden opportunity for threat actors to exploit the situation, especially those targeting high-profile victims. In the past month while the Russian invasion of Ukraine was unfolding, Check Point Research (CPR) has observed advanced persistent threat (APT) groups around the world launching new campaigns, or quickly adapting ongoing ones to target victims with spear-phishing emails using the war as a lure. The attackers use decoys ranging from official-looking documents to news articles or even job postings, depending on the targets and region. Many of these lure documents utilize malicious macros or template injection to gain an initial foothold into the targeted organizations, and then launch malware attacks. The use of the conflict as a bait is not limited to a specific region or APT group, it goes from Latin America to the Middle East and to Asia. In this article, CPR will provide an overview of several campaigns by different APT groups using the ongoing Russia-Ukraine war to increase the efficiency of their campaigns. CPR will discuss the victimology of these campaigns; the tactics used, and provide technical analysis of the observed malicious payloads and malware. Below are the campaigns identified and profiled in this article:

APT Name	APT Origin	Targeted Sector	Targeted Countries
El Machete	Spanish-speaking Country	Financial, Governmental	Nicaragua, Venezuela
Lyceum	Iran	Energy	Israel, Saudi Arabia
SideWinder	Possibly India	Unknown	Pakistan

Latin America: El Machete APT

Targets: Financial and governmental sectors

Kaspersky first publicly disclosed El Machete, a Spanish-speaking threat actor that focuses on Latin American's targets, in 2014 with the group's activity dating back to 2010. The group's activities have persisted throughout the years, adopting the practice of using government-themed documents as decoys, as well as using lures related to the current political situation.

In mid-March, El Machete was spotted sending spear-phishing emails to financial organizations in Nicaragua, with an attached Word document titled "**Dark plans of the neo-Nazi regime in Ukraine.**" The document

contained an article written and published by Alexander Khokholikov, the Russian Ambassador to Nicaragua that discussed the Russo-Ukrainian conflict from the perspective of the Kremlin.

Figure 1 – Lure document that contains an article about the Russia-Ukraine conflict, sent by El Machete APT to Nicaraguan financial institutions.

Infection chain

The malicious macro inside the document drops a base64-encoded file named `~djXsfwEFYETE.txt`, uses the built-in certutil.exe to decode it to `~djXsfwEFYETE.vbe`, an encoded VBScript file. The macro then launches the `wscript.exe` to execute the .vbe file, whose primary objective is to execute `msiexec.exe` with a remotely hosted .msi file titled `Adobe.msi`, which masquerades itself as Adobe software.

Figure 2 – Schema of the main components of the infection chain.

The `Adobe.msi` installer initially installs malware-related files to a subfolder in the user's TEMP directory. Later, the malware copies itself from the TEMP directory to a working directory `C:\ProgramData\PD`, which is set as hidden to make sure users do not see it when they open the ProgramData folder in File Explorer. The malware is primarily written in Python, and comes with two different Python interpreters that also masquerade as executables related to Adobe, `AdobeReaderUpdate.exe` and `ReaderSetting.exe`. The malware sets up persistence via a scheduled task that runs every 5 minutes, pretending to be an update task for Adobe Reader named `UpdateAdobeReader`. The task executes the `AdobeReaderUpdate` script, a customized version of the open-source Loki.Rat which has been used by the El Machete APT group in an ongoing campaign since 2020.

C&C communication

The malware does not have a hardcoded C&C server address. Instead, it relies on a file called `license.dll`, which contains a Base64-encoded URL to a BlogSpot webpage. This page seemingly contains security-related content and discuses asymmetric encryption. However, embedded inside the BlogSpot page is another base64 string that contains the encoded C&C URL that the malware will eventually use. To find the relevant URL, the malware knows to search between two hardcoded strings that are 6-7 characters long. They tend to follow the pattern of `/AAAA/` and `*AAAA/`, where the AAAA represents a 4-5-letter string.

*Figure 3 – BlogSpot page used by Adobe.msi. The C&C server is encoded between /noul/ and *noul/.*

This method of retrieving the C&C servers has several advantages. Foremost, it easily allows the attacker to make the initial connection look innocuous by connecting to a subdomain of a known and seemingly benign server (blogspot.com). In addition, the attackers can switch C&C infrastructure very easily without having to redeploy new code to the victims' machines.

The data is submitted to the C&C server in a somewhat obfuscated but consistent JSON format:

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

```
{
"nu8": "<hostname-username>",
"d4": "<tag>",
```

"r88": "<module name/data type>",

"m77": "<file path>.pgp",

"ns32": "<payload>",

"submit": "submit"

}

{ "nu8": "<hostname-username>", "d4": "<tag>", "r88": "<module name/data type>", "m77": "<file path>.pgp", "ns32": "<payload>", "submit": "submit" }

```
{
        "nu8": "<hostname-username>",
        "d4": "<tag>",
        "r88": "<module name/data type>",
        "m77": "<file path>.pgp",
        "ns32": "<payload>",
        "submit": "submit"
}
```

The tag in the d4 field used by the Adobe malware is `Utopiya_Nyusha_Maksim`, which El Machete has used since 2020.

The Loki.Rat Backdoor

Each of the Python script files is obfuscated using base64 encoding. However, once decoded from base64, the code is relatively straightforward, only with few minor variable name obfuscation.

Figure 4 – Deobfuscated AdobeReaderUpdate script.

Malware capabilities include:

- Keylogging – The keylogger runs as a separate process and script: the `ReaderSetting.exe` Python interpreter is used to run a separate file called `SearchAdobeReader`.
- Collect credentials stored in Chrome and Firefox browsers.
- Upload and download files.

- Collect information about the files on each drive – collect file names and file sizes for all the files with the extensions from the list: `.doc, .docx, .pdf, .xlsx, .xls, .ppt, .pptx, .jpg, .jpeg, .rar, .zip, .odt, .ott, .odm, .ods, .ots, .odp.` except excluded (system, temp) folders.
- Take screenshots.
- Collect clipboard data.
- Execute commands.

Commands and payloads

The actors first send several commands to understand if the infected machine is interesting enough to proceed: these commands perform screenshots, keylogging, and listing files on the system. If deemed worthwhile, the actors execute a command to download and install another malware, `JavaOracle.msi`, via `msiexec.exe`.

Similar to `Adobe.msi`, `JavaOracle.msi` installs a Python-based malware and uses scheduled tasks for persistence. However, the Python scripts are not based on the Loki.Rat backdoor, although they offer some similar functionality through the modules placed in the directory `Libs\site-packages\Java`. The malware was observed launching multiple Python interpreters in parallel, each one running a different module. The Python executables are disguised as `JavaHosts.exe`, `JavaExt.exe` and `JavaAdd.exe`, and the actors also use these Python "clones" to check if a certain script/module is running, based on the process name. The modules include the following capabilities:

- Download a payload from the C&C server (`GAME` module) – The code implies that the payload is expected to be either a .exe or a .msi file. The payload is written to the directory `C:\ProgramData\ControlD\`, which it sets as a folder with system and hidden attributes.
- Keylogger (`TIME` module) – This is similar to the one that came with the Adobe.msi payload, but it never writes to disk. Instead, it posts the keylogger data directly to the C&C server.
- `BOX` module – This iterates over files in the system and uploads files of interest that are less than 5 MB, encoded as base64. The module first checks connectivity by opening a TCP socket to google.es. If the site is not accessible, the script exits.
- Screenshot (`LIST` module) – The module saves screenshots to `-shopt.png` inside a directory masquerading as Microsoft, namely `%APPDATA%\Microsoft\ControlDesktop\`. It then uploads the screenshot to the C&C server and proceeds to delete all PNG files in this directory. Similar to BOX, it initially checks that it can open a TCP socket to google.ru. If it fails, the script exits.
- Clipboard stealer (`SCAN` module) – Posts the data directly to the C&C server, without writing the data to disk. Before doing so, it checks that it can open a TCP connection to google.ru.

The malware from the `JavaOracle.msi` file seems to be using a new hardcoded tag, `Foo_Fighters_Everlong`. The timing appears to be coincidental, as the payload was first seen a few days before the news that Foo Fighters drummer Taylor Hawkins died.

Figure 5 – JavaOracle code steals the clipboard contents and posts the data to the C&C with a custom tag.

Targets and goals

Although the specific email trap targeted a financial institution in Nicaragua, multiple artifacts suggest that this is part of a larger campaign, which is also targeting government entities in Venezuela. Judging by the activities that the actors perform in the infected networks, the purpose of the whole campaign is deemed a cyberespionage operation, consistent with the previously disclosed activity by the same attack group. This indicates that El Machete APT group continues to operate with slightly changing TTPs, even after researchers published technical descriptions and indicators of compromise for the malware used by the group.

Middle East: Lyceum

Targets: Energy sector

Believed to be active since 2017, Lyceum is an Iranian APT group active in the Middle East and Africa, and is known to target sectors of strategic national importance to carry out cyber espionage. Mid March, an Israeli energy company received an email from the address `inews-reporter@protonmail[.]com` with the subject "Russian war crimes in Ukraine". The email contained a few pictures taken from public media sources and contained a link to an article hosted on the `news-spot[.]live` domain:

Figure 6 – Lure email utilizing the Russia-Ukraine conflict theme, sent by Lyceum group.

The link in the email leads to a document that contains the article “Researchers gather evidence of possible Russian war crimes in Ukraine” published by The Guardian:

Figure 7 – Lure document that contains The Guardian article on possible Russian war crimes in Ukraine.

The same domain hosts a few more malicious documents related to the Russia and Russia-Ukraine conflict, such as a copy of an article by The Atlantic Council from 2020 on Russian nuclear weapons, and a job posting for the "Extraction / Protective Agent" agent in Ukraine:

Figure 8 – Russia-Ukraine war-related decoy documents used by the Lyceum APT group.

Infection chain

The malicious Office document executes a macro code when the document is closed. The macro deobfuscates an executable embedded in the document and saves it to the `%APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup\` directory. By using this method, the payload isn't executed directly by the Office document, but it will run the next time the computer is restarted.

As part of the wider Lyceum campaign, we also observed different executable droppers. These are executables bearing PDF icons, not documents:

Figure 9 – Two variants of Lyceum infection chain: lures related to the RU-UA conflict (top) and to Iran (bottom).

All the executables are written slightly differently but the main idea is the same: first, the dropper extracts a lure PDF file embedded as a resource and opens it, in the background and unnoticed by a victim, the dropper then downloads and executes the payload. We identified three categories of droppers:

- **.NET DNS dropper** – Used to drop the .NET DNS backdoor (discussed later):

Figure 10 – The .NET dropper opens the decoy PDF and downloads the payload.

- **.NET TCP Dropper** – Drops the .NET HTTP backdoor variant, and adds a scheduled task to run it.
- **Golang Dropper –** Drops the Golang backdoor to the `Startup` folder and the `Public\Downloads` folder. In addition, it drops a PDF file (a report about the Iranian cyber threat, similar to the other droppers) to the `Public\Downloads` folder and executes it. After the PDF report is opened, the dropper finally executes the Golang backdoor from `Public\Downloads` folder.

Figure 11 – Code snippet of the Golang dropper, which drops a Golang backdoor and a PDF report titled "Iranian Cyber Threat".

The dropped files can be downloaded from the internet, or extracted from the dropper itself, depending on the sample.

Payloads

Each dropper bring its own type of payload. We observed the following backdoors deployed:

.NET DNS Backdoor

The .NET DNS backdoor is a modified version of a tool called DnsDig, with code added to form frm1 that uses HeijdenDNS and DnsDig capabilities.

Figure 12 – Original DnsDig tool (left) vs Modified DnsDig (added frm1).

The backdoor uses DNS tunneling to communicate with its C&C server, and is able to download/upload files and execute commands.

.NET TCP Backdoor

The backdoor communicates with the C&C using raw TCP sockets, and it implements its own communication protocol on top of this. Each sample contains a configuration that defines how it should communicate with the C&C, including separator characters, TCP ports and mapping of command types to numbers:

Figure 13 – Configuration snippet of the .NET TCP backdoor.

Although the malware contains a configuration for the C&C communication, it still uses hardcoded values in the code itself, instead of the configuration constants. This indicates that the malware might still be under active development.

The capabilities of this backdoor include:

- Execute commands.

- Take screenshots.
- List files/directories.
- List installed applications.
- Upload/download/execute files.

Golang HTTP Backdoor

The execution of the HTTP backdoor, written in Golang, consists of 3 stages, that occur in a loop:

Stage 1 – Connectivity check. The malware generates a unique ID for the victim, based on the MD5 hash of the username. It then sends an empty HTTP POST request to the URI `/GO/1.php` of the C&C server. If the server responds with OK, the backdoor continues to the next stage.

Stage 2 – Victim registration. In this step, the malware sends basic details of the victim in a POST request to the URI `/GO/2.php`, to register the victim in the attacker's C&C server.

Stage 3 – Commands retrieval and execution. First, the malware sends HTTP POST requests to the URI `/GO/3.php` to get commands for execution. Like the other backdoors we described, the backdoor supports

commands that allow it to download/upload files and execute shell commands.

Figure 14 – Network traffic of the Golang HTTP backdoor, per execution stage

Attribution and victimology

In addition to targets in the Israeli energy sector, when hunting for the files and infrastructure related to this attack, CPR observed some artifacts uploaded to VirusTotal (VT) from Saudi Arabia. Although these artifacts contain traps related to Iran, the other documents found on the relevant infrastructure suggest that the group might have used the baits related to the Russia-Ukraine war in Saudi Arabia as well, and probably in other countries in the region, which is the primary focus of the group's activities.

As well as the clear victimology, other indicators that suggest this activity is from the Lyceum APT group include:

- Use of Heijden.DNS open-source library, which was used by Lyceum in their previous attacks. This time, the actors did not obfuscate the library name but modified a tool named DnsDig that uses Heijden.DNS.
- DNS tunneling technique in the C&C communication widely used in previous Lyceum campaigns.
- Overlaps in the infrastructure, such as known Lyceum C&C servers hosted on the same ASN in the same networks with C&C from this campaign, and use of the same domain registrars such as Namecheap.
- Use of Protonmail email addresses to send the malicious email to their targets or to register the domains.

Judging by the timestamps artifacts found and malicious domains registration, this specific campaign has been running for a few months. The adoption of more relevant lures and constant malware retooling suggests that the Lyceum group will continue to conduct and adjust their espionage operations in the Middle East, despite public disclosures.

South Asia: SideWinder

Targets: Entities in Pakistan

SideWinder is a suspected Indian APT group that strongly focuses on Pakistan and China government organizations. SideWinder's malicious document, which also exploit the Russia-Ukraine conflict, was uploaded to VT in the middle of March. Judging by its content, the intended targets are Pakistani entities; the bait document contains the document of National Institute of Maritime Affairs of Bahria University in Islamabad, and is titled "Focused talk on Russian Ukraine Conflict Impact on Pakistan."

Figure 15 – Decoy document related to Russia-Ukraine war, by Sidewinder APT.

This malicious document uses remote template injection. When it's opened, the document retrieves a remote template from an actor-controlled server. The external template that's downloaded is an RTF file that exploits the CVE-2017-11882 (Equation Editor) vulnerability. When the vulnerability is exploited, it drops and executes 1.a package, that contains obfuscated JavaScript. The Sidewinder campaigns TTPs have not changed in the last few years, so we do not include exact technical details here, as they have been thoroughly described by multiple researchers.

However, it is worth mentioning that a typical SideWinder APT payload is a .NET-based infostealer, originally called “SystemApp.dll”, and is capable of gathering system information, exfiltrating files from the infected machine and executing commands. The infostealer has been used with minor modifications in the group’s espionage campaigns since early 2019.

Conclusion

CPR shared a few examples of APT groups attempting to abuse the interest in the ongoing war between Russia and Ukraine. As some of these campaigns contain previously undisclosed technical details or updated malware, CPR researchers included Yara rules in the Appendix, which can assist with threat hunting for these APT campaigns and the tools they utilize.

Although the attention of the public does not usually linger on a single issue for an extended period, the Russian-Ukrainian war is an obvious exception. This war affects multiple regions around the world and has potentially far-reaching ramifications. As a result, we can expect that APT threat actors will continue to use this crisis to conduct targeted phishing campaigns for espionage purposes.

IOCs

Lyceum APT:

```
13814a190f61b36aff24d6aa1de56fe2
f9fd9e32cb04c4fc93e65f48562ecad3
53542ec51daf61fba2d26fe91b7d701f
d962dd55fde800d972a156f5c63a6243
1a5489147a888c4f5f32e97ffcb01733
9fcad8f97eeae10f7a222eca94cb9a5f
f8c29040122cf892190bcf3665975d2f
a5dbfd729b6fd64a6c4fd77a3e356989
8b01dec07856a67db0e0d849bc84fd9e
23d174e6a0905fd59b2613d5ac106261
a437f997d45bc14e76d0f2482f572a34
ce186cda677f0120cfdb308803b8e8d8
214011a0d57b1d8238532be4f6414f58
8d51fbb90ad5942cd1a5a6534bd9d1d7
6aeca48c9090b301b3fdf9da4382c882
c41ffcbd933039bb6981d05b4c4c673e
e03c7e3e8957ede592de07d3dca247b7
f72768f352994ecce3b9e5109fe93eec
8199f14502e80581000bd5b3bda250ee
d79687676d2d152aec4143c852bdbc4a
2bc2abefc1a721908bc805894b62227d
37a1514a7a5f9b2c6786096129a30721
1c444ebeba24dcba8628b7dfe5fec7c6
85ca334f87667bd7fa0c47ae6149353e
73bddd5f1a0847ae5f5d55e7d9c177f6
```

```
9fb86915db1b7c00f1a4587de4e052de
37fe608983d4b06a5549247f0e16bc11
5916e5189ef0050dfcc3cc19382d08d5
f3b395661cc663c1baad41b439622071
8044dc6078b003698d6e1cbbd22a9ea6
bcb465cc2257e5777bab431690ca5039
news-spot[.]live
news-spot[.]xyz
cyberclub[.]one
science-news[.]live
news-reporter[.]xyz
104.249.26[.]60
85.206.175[.]201
185.243.112[.]136
```

El Machete APT:

```
8e1360cc27e95fc47924d9ba3ef84cb8fa9e142cfd16e1503c5277d0c16ae241
e2c67e495166be1b97134e67b2326e1b800d3d4d8dba4bc61fd3f8eb3a92d612
e3718adaca6eafeba6ff171669210cb55a3b8babf3b78072cc513273b99a7639
ed09da9d48afe918f9c7f72fe4466167e2f127a28a7641ba80d6165e82f48431
b9bf3e9725696331916e32e5936111e1166867b1d2d3ab05e46b9fff8679cf8f
c6c794348d17d40c544487154ca72e8e6199b670f804ee25d7bcd9ff884d67b1
7115580f8235a0bbce61e8af79c3ed5cbe46900912eb0765ccaee82213a9275e
907ccb541d0066d36701310e86e1d2b61448178d1d36f6748af0b3163ca273ac
7ea7cae7dd6353831359179f4834ac4c2e9022659e205ca8506f372aad63f629
bb4b04eff1b5154d23b2636fc55222e4f27c654777f348edee47c920e457835e
ebbcc2075fcb0ba18d43475b8454c51b35bb65e1ed323b657ea7d9651e98074d
da81697353fe3238920a8c2c4cbbf25a298b3e3414f988ece0cf7afb73e3e0a5
4c22116b68732f8fe9e2fb5e56e9ff798f30805f9008e4f7a4be1e1c830162b8
65e48c986d185d156999adc762d7bff84ddbf44851419d66c2985a2ccc2e072d
caac5087528dde6839481133737de12af973080184b2aa0b2eb35af88875adbb
a5f0af1124f7abf06e712a2bfb4f1104ee0df17934302057795933961 7db69b3
ca4182fbaf3f02d9b428f7e851d5a679d6dcfceafabb245cff155b48d9c09307
96b33df5720901b4f2fc6fb810b6eca994fb8b2ff0edc0aa456195a7c9115615
e27f75c4e4e74bff20270ec0f2bd41a4b54c121bcb811451a67c831dba1e4c03
a26751cde843d44506ccece87d6347ede5071703bfd63fb12f8982eae7aaf3dd
e60ea877d008e61cb625b4f8b2d712ce9289892f7e799dbb1030301e2db4b0ac
hxxps://correomindefensagobvemyspace[.]com/kolomenskoye/Adobe.msi
hxxps://solutionconect[.]online/uu2/x3/JavaOracle.msi
hxxps://great-jepsen.51-79-62-98[.]plesk[.]page/MKS/w3/Adobe.msi
hxxps://asymmetricfile.blogspot[.]com
hxxps://postinfomatico.blogspot[.]com
hxxp://31.207.44[.]72:8080
hxxps://Intelligent-archimedes.51-79-62-98[.]plesk[.]page/x3/Uu-3.php
```

Sidewinder APT:

```
f765b0b6e4a34eb95c6f0ddf058bc88d5ef9ec2b11a5f3504d1673f4f69aceca
maritimepakistan.kpt-pk[.]net
kpt-pk[.]net
```

Appendix – YARA rules

Plain text

Copy to clipboard

Open code in new window

EnlighterJS 3 Syntax Highlighter

rule lyceum_dotnet_dns_backdoor

{

meta:

author = "CPR"

hash1 = "8199f14502e80581000bd5b3bda250ee"

hash2 = "d79687676d2d152aec4143c852bdbc4a"

hash3 = "bcb465cc2257e5777bab431690ca5039"

hash4 = "2bc2abefc1a721908bc805894b62227d"

hash5 = "37a1514a7a5f9b2c6786096129a30721"

strings:

$log1 = "MSG SIZE rcvd" wide

$log2 = "Empty output" wide

$log3 = "Big Output. lines: " wide

$com1 = "Enddd" wide

$com2 = "uploaddd" wide

$com3 = "downloaddd" wide

$dga = "trailers.apple.com" wide

```
$replace1 = "BackSlashh" wide

$replace2 = "QuotationMarkk" wide

$re_pattern = "60\\s+IN\\s+TXT" wide

$func1 = "comRun"

$func2 = "PlaceDot"

$func3 = "sendAns"

$heijden1 = "Heijden.DNS"

$heijden2 = "DnsHeijden"

condition:

uint16(0)==0x5a4d and (all of ($log*) or all of ($com*) or all of ($replace*) or all of ($func*) or (any of ($heijden*) and $re_pattern and $dga))

}

rule lyceum_dotnet_http_backdoor

{

meta:

author = "CPR"

hash1 = "1c444ebeba24dcba8628b7dfe5fec7c6"

hash2 = "85ca334f87667bd7fa0c47ae6149353e"

hash3 = "73bddd5f1a0847ae5f5d55e7d9c177f6"

hash4 = "9fb86915db1b7c00f1a4587de4e052de"

hash5 = "37fe608983d4b06a5549247f0e16bc11"

hash6 = "5916e5189ef0050dfcc3cc19382d08d5"

strings:

$class1 = "Funcss"

$class2 = "Constantss"

$class3 = "Reqss"

$class4 = "Screenss"
```

$class5 = "Shll"

$class6 = "test_A1"

$class7 = "Uploadss"

$class8 = "WebDL"

$cnc_uri1 = "/upload" wide

$cnc_uri2 = "/screenshot" wide

$cnc_pattern_hex1 = {43 6f 6e 74 65 6e 74 2d 44 69 73 70 6f 73 69 74 69 6f 6e 3a 20 66 6f 72 6d 2d 64 61 74 61 3b 20 6e 61 6d 65 3d 22 7b 30 7d 22 0d 0a 0d 0a}

$cnc_pattern_hex2 = {6d 75 6c 74 69 70 61 72 74 2f 66 6f 72 6d 2d 64 61 74 61 3b 20 62 6f 75 6e 64 61 72 79 3d 7b 30 7d}

$cnc_pattern_hex3 = {43 6f 6e 74 65 6e 74 2d 44 69 73 70 6f 73 69 74 69 6f 6e 3a 20 66 6f 72 6d 2d 64 61 74 61 3b 20 6e 61 6d 65 3d 22 7b 30 7d 22 3b 20 66 69 6c 65 6e 61 6d 65 3d 22 7b 31 7d 22 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a 20 7b 32 7d 0d 0a 0d 0a}

$constant1 = "FILE_DIR_SEPARATOR"

$constant2 = "APPS_PARAMS_SEPARATOR"

$constant3 = "TYPE_SENDTOKEN"

$constant4 = "TYPE_DATA1"

$constant5 = "TYPE_SEND_RESPONSE_IN_SOCKET"

$constant6 = "TYPE_FILES_LIST"

$constant7 = "TYPE_FILES_DELETE"

$constant8 = "TYPE_FILES_RUN"

$constant9 = "TYPE_FILES_UPLOAD_TO_SERVER"

$constant10 = "TYPE_FILES_DELETE_FOLDER"

$constant11 = "TYPE_FILES_CREATE_FOLDER"

$constant12 = "TYPE_FILES_DOWNLOAD_URL"

$constant13 = "TYPE_OPEN_CMD"

$constant14 = "TYPE_CMD_RES"

$constant15 = "TYPE_CLOSE_CMD"

```
$constant16 = "TYPE_CMD_REQ"

$constant17 = "TYPE_INSTALLED_APPS"

$constant18 = "TYPE_SCREENSHOT"

$constant19 = "_RG_APP_NAME_"

$constant20 = "_RG_APP_VERSION_"

$constant21 = "_RG_APP_DATE_"

$constant22 = "_RG_APP_PUB_"

$constant23 = "_RG_APP_SEP_"

$constant24 = "_SC_EXT_"

condition:

uint16(0)==0x5a4d and (4 of ($class*) or 4 of ($cnc_*) or 4 of ($constant*))

}

rule lyceum_golang_backdoor

{

meta:

author = "CPR"

hash1 = "a437f997d45bc14e76d0f2482f572a34"

hash2 = "23d174e6a0905fd59b2613d5ac106261"

hash3 = "bcb465cc2257e5777bab431690ca5039"

strings:

$func1 = "main.Ase256"

$func2 = "main.DecryptAse256"

$func3 = "main.IsServerUp"

$func4 = "main.register"

$func5 = "main.commandforrun"

$func6 = "main.UPLOAD"
```

$func7 = "main.commandforanswer"

$func8 = "main.GetMD5Hash"

$func9 = "main.get_uid"

$func10 = "main.commandrun"

$func11 = "main.download"

$func12 = "main.postFile"

$func13 = "main.sendAns"

$func14 = "main.comRun"

$cnc_uri1 = "/GO/1.php"

$cnc_uri2 = "/GO/2.php"

$cnc_uri3 = "/GO/3.php"

$auth_token = "auth_token=\"XXXXXXX\""

$log1 = "client registred"

$log2 = "no command"

$log3 = "can not create file"

$log4 = "errorGettingUserName"

$log5 = "New record created successfully"

$log6 = "SERVER_IS_DOWN"

$dga = "trailers.apple.com."

condition:

uint16(0)==0x5a4d and ((10 of ($func*) or any of ($cnc_uri*) or $auth_token or 3 of ($log*)) or ($dga and 4 of them))

}

rule ElMachete_doc

{

meta:

author = "CPR"

hash1 = "8E1360CC27E95FC47924D9BA3EF84CB8FA9E142CFD16E1503C5277D0C16AE241"

strings:

$s1 = "You want to continue with the Document" ascii

$s2 = "certutil -decode" ascii

$s3 = /C:\\ProgramData\\.{1,20}\.txt/

$s4 = /C:\\ProgramData\\.{1,20}\.vbe/

condition:

uint16be(0) == 0xD0CF and 2 of ($s*)

}

rule ElMachete_msi

{

meta:

author = "CPR"

hash1 = "ED09DA9D48AFE918F9C7F72FE4466167E2F127A28A7641BA80D6165E82F48431"

strings:

$s1 = "MSI Wrapper (8.0.26.0)"

$s2 = "Windows Installer XML Toolset (3.11.0.1701)"

$s3 = "\\Lib\\site-packages\\PIL\\"

$s4 = "\\Lib\\site-packages\\pyHook\\"

$s5 = "\\Lib\\site-packages\\requests\\"

$s6 = "\\Lib\\site-packages\\win32com\\"

$s7 = "\\Lib\\site-packages\\Crypto\\"

condition:

4 of them

}

rule lyceum_dotnet_dns_backdoor { meta: author = "CPR" hash1 = "8199f14502e80581000bd5b3bda250ee" hash2 = "d79687676d2d152aec4143c852bdbc4a" hash3 = "bcb465cc2257e5777bab431690ca5039" hash4 =

"2bc2abefc1a721908bc805894b62227d" hash5 = "37a1514a7a5f9b2c6786096129a30721" strings: $log1 = "MSG SIZE rcvd" wide $log2 = "Empty output" wide $log3 = "Big Output. lines: " wide $com1 = "Enddd" wide $com2 = "uploaddd" wide $com3 = "downloaddd" wide $dga = "trailers.apple.com" wide $replace1 = "BackSlashh" wide $replace2 = "QuotationMarkk" wide $re_pattern = "60\\s+IN\\s+TXT" wide $func1 = "comRun" $func2 = "PlaceDot" $func3 = "sendAns" $heijden1 = "Heijden.DNS" $heijden2 = "DnsHeijden" condition: uint16(0)==0x5a4d and (all of ($log*) or all of ($com*) or all of ($replace*) or all of ($func*) or (any of ($heijden*) and $re_pattern and $dga)) } rule lyceum_dotnet_http_backdoor { meta: author = "CPR" hash1 = "1c444ebeba24dcba8628b7dfe5fec7c6" hash2 = "85ca334f87667bd7fa0c47ae6149353e" hash3 = "73bddd5f1a0847ae5f5d55e7d9c177f6" hash4 = "9fb86915db1b7c00f1a4587de4e052de" hash5 = "37fe608983d4b06a5549247f0e16bc11" hash6 = "5916e5189ef0050dfcc3cc19382d08d5" strings: $class1 = "Funcss" $class2 = "Constantss" $class3 = "Reqss" $class4 = "Screenss" $class5 = "Shll" $class6 = "test_A1" $class7 = "Uploadss" $class8 = "WebDL" $cnc_uri1 = "/upload" wide $cnc_uri2 = "/screenshot" wide $cnc_pattern_hex1 = {43 6f 6e 74 65 6e 74 2d 44 69 73 70 6f 73 69 74 69 6f 6e 3a 20 66 6f 72 6d 2d 64 61 74 61 3b 20 6e 61 6d 65 3d 22 7b 30 7d 22 0d 0a 0d 0a} $cnc_pattern_hex2 = {6d 75 6c 74 69 70 61 72 74 2f 66 6f 72 6d 2d 64 61 74 61 3b 20 62 6f 75 6e 64 61 72 79 3d 7b 30 7d} $cnc_pattern_hex3 = {43 6f 6e 74 65 6e 74 2d 44 69 73 70 6f 73 69 74 69 6f 6e 3a 20 66 6f 72 6d 2d 64 61 74 61 3b 20 6e 61 6d 65 3d 22 7b 30 7d 22 3b 20 66 69 6c 65 6e 61 6d 65 3d 22 7b 31 7d 22 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a 20 7b 32 7d 0d 0a 0d 0a} $constant1 = "FILE_DIR_SEPARATOR" $constant2 = "APPS_PARAMS_SEPARATOR" $constant3 = "TYPE_SENDTOKEN" $constant4 = "TYPE_DATA1" $constant5 = "TYPE_SEND_RESPONSE_IN_SOCKET" $constant6 = "TYPE_FILES_LIST" $constant7 = "TYPE_FILES_DELETE" $constant8 = "TYPE_FILES_RUN" $constant9 = "TYPE_FILES_UPLOAD_TO_SERVER" $constant10 = "TYPE_FILES_DELETE_FOLDER" $constant11 = "TYPE_FILES_CREATE_FOLDER" $constant12 = "TYPE_FILES_DOWNLOAD_URL" $constant13 = "TYPE_OPEN_CMD" $constant14 = "TYPE_CMD_RES" $constant15 = "TYPE_CLOSE_CMD" $constant16 = "TYPE_CMD_REQ" $constant17 = "TYPE_INSTALLED_APPS" $constant18 = "TYPE_SCREENSHOT" $constant19 = "_RG_APP_NAME_" $constant20 = "_RG_APP_VERSION_" $constant21 = "_RG_APP_DATE_" $constant22 = "_RG_APP_PUB_" $constant23 = "_RG_APP_SEP_" $constant24 = "_SC_EXT_" condition: uint16(0)==0x5a4d and (4 of ($class*) or 4 of ($cnc_*) or 4 of ($constant*)) } rule lyceum_golang_backdoor { meta: author = "CPR" hash1 = "a437f997d45bc14e76d0f2482f572a34" hash2 = "23d174e6a0905fd59b2613d5ac106261" hash3 = "bcb465cc2257e5777bab431690ca5039" strings: $func1 = "main.Ase256" $func2 = "main.DecryptAse256" $func3 = "main.IsServerUp" $func4 = "main.register" $func5 = "main.commandforrun" $func6 = "main.UPLOAD" $func7 = "main.commandforanswer" $func8 = "main.GetMD5Hash" $func9 = "main.get_uid" $func10 = "main.commandrun" $func11 = "main.download" $func12 = "main.postFile" $func13 = "main.sendAns" $func14 = "main.comRun" $cnc_uri1 = "/GO/1.php" $cnc_uri2 = "/GO/2.php" $cnc_uri3 = "/GO/3.php" $auth_token = "auth_token=\"XXXXXXX\"" $log1 = "client registred" $log2 = "no command" $log3 = "can not create file" $log4 = "errorGettingUserName" $log5 = "New record created successfully" $log6 = "SERVER_IS_DOWN" $dga = "trailers.apple.com." condition: uint16(0)==0x5a4d and ((10 of ($func*) or any of ($cnc_uri*) or $auth_token or 3 of ($log*)) or ($dga and 4 of them)) } rule ElMachete_doc { meta: author = "CPR" hash1 = "8E1360CC27E95FC47924D9BA3EF84CB8FA9E142CFD16E1503C5277D0C16AE241" strings: $s1 = "You want to continue with the Document" ascii $s2 = "certutil -decode" ascii $s3 = /C:\\ProgramData\\.{1,20}\.txt/ $s4 = /C:\\ProgramData\\.{1,20}\.vbe/ condition: uint16be(0) == 0xD0CF and 2 of ($s*) } rule ElMachete_msi { meta: author = "CPR" hash1 =

"ED09DA9D48AFE918F9C7F72FE4466167E2F127A28A7641BA80D6165E82F48431" strings: $s1 = "MSI Wrapper (8.0.26.0)" $s2 = "Windows Installer XML Toolset (3.11.0.1701)" $s3 = "\\Lib\\site-packages\\PIL\\" $s4 = "\\Lib\\site-packages\\pyHook\\" $s5 = "\\Lib\\site-packages\\requests\\" $s6 = "\\Lib\\site-packages\\win32com\\" $s7 = "\\Lib\\site-packages\\Crypto\\" condition: 4 of them }

```
rule lyceum_dotnet_dns_backdoor
{
    meta:
        author = "CPR"
        hash1 = "8199f14502e80581000bd5b3bda250ee"
        hash2 = "d79687676d2d152aec4143c852bdbc4a"
        hash3 = "bcb465cc2257e5777bab431690ca5039"
        hash4 = "2bc2abefc1a721908bc805894b62227d"
        hash5 = "37a1514a7a5f9b2c6786096129a30721"
    strings:
        $log1 = "MSG SIZE rcvd" wide
        $log2 = "Empty output" wide
        $log3 = "Big Output. lines: " wide
        $com1 = "Enddd" wide
        $com2 = "uploaddd" wide
        $com3 = "downloaddd" wide
        $dga = "trailers.apple.com" wide
        $replace1 = "BackSlashh" wide
        $replace2 = "QuotationMarkk" wide
        $re_pattern = "60\\s+IN\\s+TXT" wide
        $func1 = "comRun"
        $func2 = "PlaceDot"
        $func3 = "sendAns"
        $heijden1 = "Heijden.DNS"
        $heijden2 = "DnsHeijden"
    condition:
        uint16(0)==0x5a4d and (all of ($log*) or all of ($com*) or all of ($replace*) or all of ($fu
}

rule lyceum_dotnet_http_backdoor
{
    meta:
        author = "CPR"
        hash1 = "1c444ebeba24dcba8628b7dfe5fec7c6"
        hash2 = "85ca334f87667bd7fa0c47ae6149353e"
        hash3 = "73bddd5f1a0847ae5f5d55e7d9c177f6"
        hash4 = "9fb86915db1b7c00f1a4587de4e052de"
        hash5 = "37fe608983d4b06a5549247f0e16bc11"
        hash6 = "5916e5189ef0050dfcc3cc19382d08d5"
    strings:
        $class1 = "Funcss"
```

```
        $class2 = "Constantss"
        $class3 = "Reqss"
        $class4 = "Screenss"
        $class5 = "Shll"
        $class6 = "test_A1"
        $class7 = "Uploadss"
        $class8 = "WebDL"
        $cnc_uri1 = "/upload" wide
        $cnc_uri2 = "/screenshot" wide
        $cnc_pattern_hex1 = {43 6f 6e 74 65 6e 74 2d 44 69 73 70 6f 73 69 74 69 6f 6e 3a 20 66 6f 72
        $cnc_pattern_hex2 = {6d 75 6c 74 69 70 61 72 74 2f 66 6f 72 6d 2d 64 61 74 61 3b 20 62 6f 75
        $cnc_pattern_hex3 = {43 6f 6e 74 65 6e 74 2d 44 69 73 70 6f 73 69 74 69 6f 6e 3a 20 66 6f 72
        $constant1 = "FILE_DIR_SEPARATOR"
        $constant2 = "APPS_PARAMS_SEPARATOR"
        $constant3 = "TYPE_SENDTOKEN"
        $constant4 = "TYPE_DATA1"
        $constant5 = "TYPE_SEND_RESPONSE_IN_SOCKET"
        $constant6 = "TYPE_FILES_LIST"
        $constant7 = "TYPE_FILES_DELETE"
        $constant8 = "TYPE_FILES_RUN"
        $constant9 = "TYPE_FILES_UPLOAD_TO_SERVER"
        $constant10 = "TYPE_FILES_DELETE_FOLDER"
        $constant11 = "TYPE_FILES_CREATE_FOLDER"
        $constant12 = "TYPE_FILES_DOWNLOAD_URL"
        $constant13 = "TYPE_OPEN_CMD"
        $constant14 = "TYPE_CMD_RES"
        $constant15 = "TYPE_CLOSE_CMD"
        $constant16 = "TYPE_CMD_REQ"
        $constant17 = "TYPE_INSTALLED_APPS"
        $constant18 = "TYPE_SCREENSHOT"
        $constant19 = "_RG_APP_NAME_"
        $constant20 = "_RG_APP_VERSION_"
        $constant21 = "_RG_APP_DATE_"
        $constant22 = "_RG_APP_PUB_"
        $constant23 = "_RG_APP_SEP_"
        $constant24 = "_SC_EXT_"
    condition:
        uint16(0)==0x5a4d and (4 of ($class*) or 4 of ($cnc_*) or 4 of ($constant*))
}

rule lyceum_golang_backdoor
{
    meta:
        author = "CPR"
        hash1 = "a437f997d45bc14e76d0f2482f572a34"
        hash2 = "23d174e6a0905fd59b2613d5ac106261"
        hash3 = "bcb465cc2257e5777bab431690ca5039"
```

```
    strings:
        $func1 = "main.Ase256"
        $func2 = "main.DecryptAse256"
        $func3 = "main.IsServerUp"
        $func4 = "main.register"
        $func5 = "main.commandforrun"
        $func6 = "main.UPLOAD"
        $func7 = "main.commandforanswer"
        $func8 = "main.GetMD5Hash"
        $func9 = "main.get_uid"
        $func10 = "main.commandrun"
        $func11 = "main.download"
        $func12 = "main.postFile"
        $func13 = "main.sendAns"
        $func14 = "main.comRun"
        $cnc_uri1 = "/GO/1.php"
        $cnc_uri2 = "/GO/2.php"
        $cnc_uri3 = "/GO/3.php"
        $auth_token = "auth_token=\"XXXXXXX\""
        $log1 = "client registred"
        $log2 = "no command"
        $log3 = "can not create file"
        $log4 = "errorGettingUserName"
        $log5 = "New record created successfully"
        $log6 = "SERVER_IS_DOWN"
        $dga = "trailers.apple.com."
    condition:
        uint16(0)==0x5a4d and ((10 of ($func*) or any of ($cnc_uri*) or $auth_token or 3 of ($log*))
}

rule ElMachete_doc
{
    meta:
        author = "CPR"
        hash1 = "8E1360CC27E95FC47924D9BA3EF84CB8FA9E142CFD16E1503C5277D0C16AE241"
    strings:
        $s1 = "You want to continue with the Document" ascii
        $s2 = "certutil -decode" ascii
        $s3 = /C:\\ProgramData\\.{1,20}\.txt/
        $s4 = /C:\\ProgramData\\.{1,20}\.vbe/
    condition:
        uint16be(0) == 0xD0CF and 2 of ($s*)
}

rule ElMachete_msi
{
    meta:
```

```
        author = "CPR"
        hash1 = "ED09DA9D48AFE918F9C7F72FE4466167E2F127A28A7641BA80D6165E82F48431"
    strings:
        $s1 = "MSI Wrapper (8.0.26.0)"
        $s2 = "Windows Installer XML Toolset (3.11.0.1701)"
        $s3 = "\\Lib\\site-packages\\PIL\\"
        $s4 = "\\Lib\\site-packages\\pyHook\\"
        $s5 = "\\Lib\\site-packages\\requests\\"
        $s6 = "\\Lib\\site-packages\\win32com\\"
        $s7 = "\\Lib\\site-packages\\Crypto\\"
    condition:
        4 of them
}
```

Source: https://research.checkpoint.com/2022/state-sponsored-attack-groups-capitalise-on-russia-ukraine-war-for-cyber-espionage/