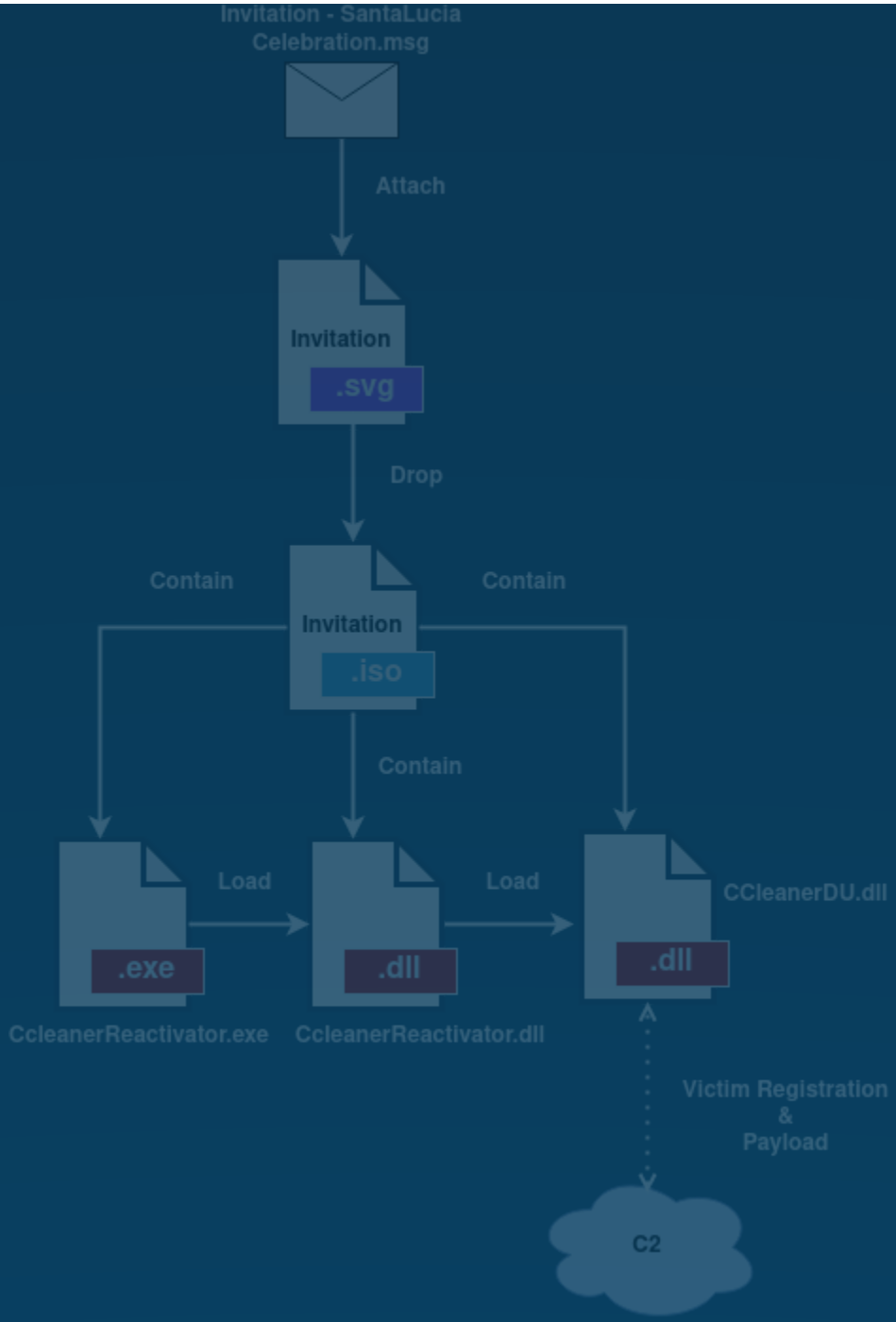
New invitation from APT29 to use CCleaner

July 12, 2023

Last month of May we were talking about the new APT29 campaign that we called "Information". Recently, just a week ago, an unknown actor used similar techniques to APT29. This time APT29 is once again the focus after new techniques were identified in their operations.

This post details the new techniques observed, in particular:

- SVG Dropper
- DLL used for infection
- C2 behaviour

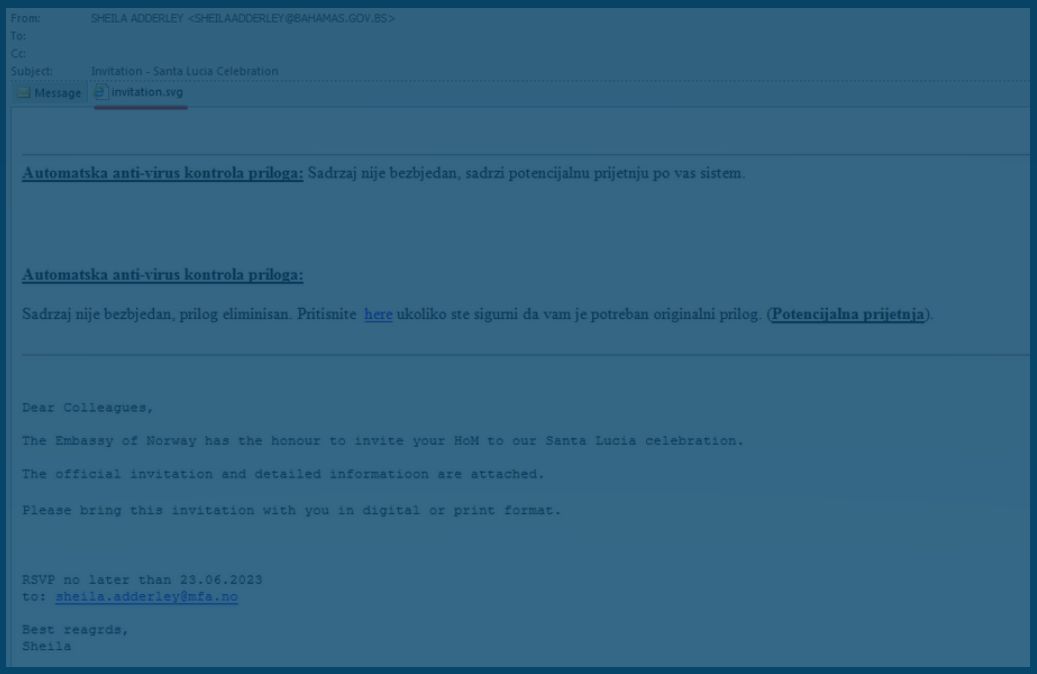


Infection chain

Stage0: SVG Dropper

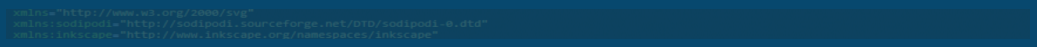
The input vector for this campaign has been the email. The phishing email used by the authors has the subject "**Invitation – Santa Lucia Celebration**". This

seems to impersonate the Norwegian embassy inviting to a celebration. This particular “invitation” is in .svg format.



Phishing Mail (@StopMalvertisin)

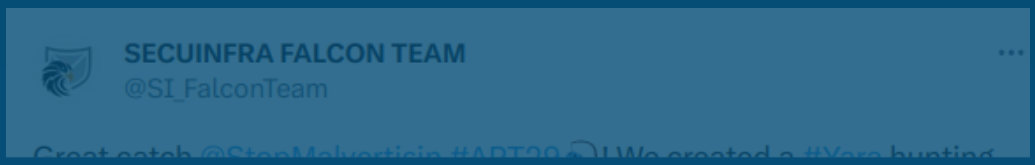
When the file is opened, a script is executed that mounts and downloads a file with .iso extension containing the next stage of infection. In this way, the .svg file functions as an HTML Smuggling that infects the victim dropping the next stage.



.svg content

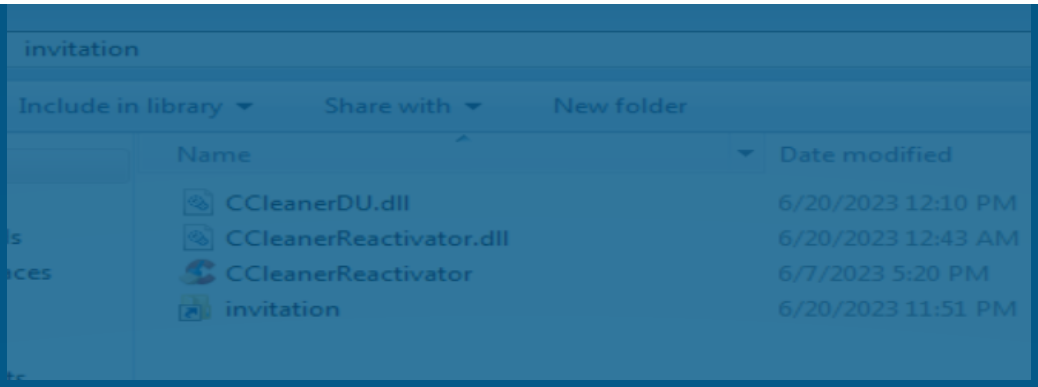
This technique had already been seen before as the user @SI_FalconTeam indicates, in a test sample dropping “Process Explorer”, they also include a Yara rule to detect this type of technique.

The use of this type of file as a dropper is a novelty in APT29 TTPs, so it is interesting to keep an eye on this type of attachments to hunt for future campaigns.



SVG "test" Sample

Once the file is opened, an ISO (**invitation.iso**) will be downloaded with a similar content to the one we have observed in other APT29 campaigns.



"Invitation.iso" content

The file used during this analysis is the following:

File	Sha256
Invitation.iso	AF1922C665E9BE6B29A5E3D0D3AC5916AE1FC74AC2FE9931E5273F3C4043F395

This particular **Invitation.iso** file contains the following files.

File	Sha256	Stage
Invitation.lnk	A8AE10B43CBF4E3344E0184B33A699B19A29866BC1E41201ACE1A995E8CA3149	Stage1
CCleanerReactivator.exe	59E5B2A7A3903E4FB9A23174B655ADB75EB490625DDB126EF29446E47DE4099F	Stage1
CCleanerReactivator.dll	7FC9E830756E23AA4B050F4CEAEB2A83CD71CFC0145392A0BC03037AF373066B	Stage1

CCleanerDU.dll	D7BDA5E39327FE12B0C1F42C8E27787F177A352F8EEBAFBE35D3E790724ECEFF	Stage2

Stage1: Loader

The first file that catches attention is **invitation.lnk**, which, despite having the icon of a folder, is a shortcut that launches the following command:

```
%windir%/system32/cmd.exe /q /c "robocopy . C:\Windows\Tasks /NODCOPY
/NFL /NDL /NJH /NJS /NC /NS /NP > nul & start C:\Windows\Tasks\CCleane
rReactivator.exe > nul"
```

This command makes use of **Robocopy** to copy all files to the "C:\Windows\Tasks" folder and then run **CCleanerReactivator.exe**.

The **CCleanerReactivator.exe** binary is signed and undetected in VirusTotal. It is a software to free up computer space that can be downloaded legitimately.

"CCleanerReactivator.exe" detections in VirusTotal

The malicious activity will therefore be found in the **CCleanerReactivator.dll** and **CCleanerDU.dll** libraries, which will be loaded by the executable using the **DLL Side-Load technique**.
In the Imports of **CCleanerReactivator.exe** we can see how it loads only the library **CCleanerReactivator.dll**.

"CCleanerReactivator.dll" imports

When looking at the ***AutoReactivatorSDK::RunProgram*** function of **CCleanerReactivator.dll** we can see that it only loads the other library **CCleanerDU.dll**, specifically the ***FreeInterface*** function.
So **CCleanerReactivator.dll** only acts as a bridge and **CCleanerDU.dll** library is the one that will contain the malicious code in its *FreeInterface* function.

"AutoReactivatorSDK::RunProgram" loading"CCleanerDU.dll".

Stage2: CCleanerDu.dll

The first thing we find in the ***FreeInstance*** function of **CCleanerDu.dll** is that it tries to load the **wininet.dll** library.
To do this, it reserves memory by directly using calls to **NtAllocateVirtualMemory** and **NtProtectVirtualMemory**. It then loads the library using the **LdrLoad** function of NTDLL.dll.

Getting "wininet.dll"

In case the library has been loaded correctly, it will start a function which we have named *C2_comm*. This Function will take care of the communication with the C2, for which it will load the following **wininet.dll** functions necessary to establish a connection:

- InternetOpenA
- InternetConnectA
- HttpOpenRequestA
- HttpSendRequestA
- InternetReadFile
- InternetCloseHandle

After this, it will try to mount the request correctly. The responsible function is one we have named ***CreateRequest*** , which does the following:

1. It obtains the UserName and the ComputerName through calls to **GetUserNameA** and **GetComputerNameExA**. With these values and a series of modular operations it will extract a 4-digit number that will identify the victim.

Create Victim ID

2. The code goes on to list all the running processes, using **CreateToolhelp32Snapshot**, **Process32First** and **Process32Next**. This information will be buffered together with the UserName and ComputerName as follows.

Exfiltration buffer

3. Next, mount the path to which the connection will be made, which follows the following format:

```
search/s.php?i=1&id=APOX8NWOV4{4_DIGITS_VICTIM_ID}
```

4. With the request created, it will perform a PUT registering the victim in the C2 **kefas[.]id**.

PUT Request

5. The last step is to check the server response, which will be successful if it receives "**KKEE**".

Check "KKEE" response

At the end of the ***CreateRequest*** function, it makes another connection and if successful performs a GET of the next stage of infection. This payload starts again with "KKEE", which it checks to see if the communication was successful. If successful it returns the payload (without the "KKEE"), otherwise it suspends execution by calling **NtDelayExecution**.

GET Request

Finally, it reserves memory again with **NtAllocateVirtualMemory** and **NtProtectVirtualMemory** and creates an execution thread with **CreateFiber** that will be in charge of launching the execution of the next stage. A fiber is a much lighter execution unit than a thread since it is not managed by the CPU but by the program itself.

CreateFiber function

C2 Communications

It is interesting to note that communication with C2 has changed significantly since previous campaigns. Previously, registration with C2 was done with a POST of an encrypted JSON with the UserName and ComputerName.

In this new iteration, victim IDs in C2 have been simplified to 4 digits. In addition, the next stage (shellcode) will be downloaded from C2 directly, instead of loading it locally.

IOCs

File	Sha256
Invitation – Santa Lucia Celebration.msg	966E070A52DE1C51976F6EA1FC48EC77F6B89F4BF5E5007650755E9CD0D73281
Invitation.svg	4875A9C4AF3044DB281C5DC02E5386C77F331E3B92E5AE79FF9961D8CD1F7C4F
Invitation.iso	AF1922C665E9BE6B29A5E3D0D3AC5916AE1FC74AC2FE9931E5273F3C4043F395
Invitation.lnk	A8AE10B43CBF4E3344E0184B33A699B19A29866BC1E41201ACE1A995E8CA3149
CCleanerReactivator.exe	59E5B2A7A3903E4FB9A23174B655ADB75EB490625DDB126EF29446E47DE4099F
CCleanerDU.dll	D7BDA5E39327FE12B0C1F42C8E27787F177A352F8EEBAFBE35D3E790724ECEFF
CCleanerReactivator.dll	7FC9E830756E23AA4B050F4CEAEB2A83CD71CFC0145392A0BC03037AF3

	73066B

C2
hxxps://kefas[.]id/search/s.php

Er1c_C

Your email address will not be published. Required fields are marked *

Comment *

Enter your comment here...

Name *

Email *

I hereby declare to have read and accepted the legal notice and the privacy policy. *

POST COMMENT

Related

New tricks of APT29 – update on the CERT.PL report
May 25, 2023

Tags: APT29

Another cyber espionage campaign in the Russia-Ukrainian ongoing cyber attacks
March 24, 2022

Tags: Cyberthreat, cyberwar, maldocs, quasarRAT, Russia, ukraine

Very very lazy Lazyscripter's scripts: double compromise in a single obfuscation
March 09, 2022

Tags: APT, h-worm, hworm, lazyscripter, obfuscation, phishing, threat, trojan