Cozy Smuggled Into the Box: APT29 Abusing Legitimate Software for Targeted Operations in Europe

cluster25.io/2022/05/13/cozy-smuggled-into-the-box/

May 13, 2022

Cozy Bear (aka **Nobelium**, **APT29**, **The Dukes**) is a well-resourced, highly dedicated and organized cyberespionage group that is believed to work in support of the decision-making process of **Russian** government since at least 2008. Nobelium primarily targets western governments and related **organizations**, with a particular focus on **government**, **diplomat**, **political and think tank** sectors. Recently we analyzed several **spear-phishing** campaigns linked with this adversary that involve the usage of a **side-loaded DLL** through **signed software** (like **Adobe suite)** and **legitimate**

webservices (like **Dropbox)** as communication vector for **Command and Control** (C&C). The misuse of legitimate webservices is in attempt to **evade** the detection from automatic analysis software. Recently, third-party researchers have also reported it used **Trello** and its **REST API** to simulate a first-level Command & Control server. In addition to this evasion attempt, as we are going to discuss later, the side-loaded DLL tries to **unhook** the **windows libraries** loaded in the process memory to evade possible EDRs.

To maximize the chances of success, **Nobelium**,in at least two cases, sent **spear-phishing** emails from spoofed or compromised government addresses. As **initial access** we identified the following **attack vectors**:

1. The first approach involves the distribution of an **IMG file** which, when mounted, contains an **LNK** shortcut and the signed software with the other DLLsand a decoy PDF as hidden files. This attack vector lures the user through a **masquerading** technique by changing the **LNK** file icon to a *folder* icon in order to convince the user to click on it. In fact, once triggered, the **cmd.exe** utility is invoked to **run** the **signed executable** and to start the **side-loading** of the **malicious DLL (i.e. AcroSup.dll).**
2. The second approach involves the usage of the **EnvyScout** dropper that is basically an **HTML** file with an embedded **JavaScript** designed to decode and drop the next-stage payload (**HTML Smuggling**). In fact, once the HTML file is executed, the JavaScript code decodes a **bytes array** and saves the result under an **archive** in the **Download directory**. In this case, the user is responsible to unzip the archive (that contains the signed software, the relative DLL's and the lure PDF) and to run manually the executable to start the chain (even if the JavaScript code contains unused snippet for automating the process).

The **EnvyScout** dropper was used by this threat actor in different campaigns. From mid-January 2022 **Cluster25** internally reported differet **Nobelium**-linkedcampaigns against **European** entities that leveraged fairly complex kill-chain started with **EnvyScout** as well.

INSIGHTS

In the reported case, the signed executable is represented by **WCChromeNativeMessagingHost.exe** from the **Adobe Create PDF** module of the Adobe suite. It's a plugin for **Google Chrome**. Since the malware bundle contains a local copy of **vcruntime140.dll**, once the abused software is executed, the local copy of this Windows library is loaded into the program memory from the **PE** import table. Analyzing the local copy of **vcruntime140.dll**, we noticed that the **PE imports of this library** have been modified: it contains an entry to the **AcroSup.dll** delivered through the malware bundle.

So, this import chain, leads to the **side-loading** of the malicious **AcroSup.dll** and the execution of its **DllMain** export before the execution of the signed **Adobe** executable. To evade possible debuggers the execution of the malicious **AcroSup.dll** starts with a **thread hijacking** by overwriting the thread context of the main thread (updating the **RIP** register) in the signed executable space. To avoid the DLL execution in suspected processes, before the thread context overwriting the malware checks for the process image name is currently matching with the name of the signed expected executable, through the **K32GetProcessImageFileNameA** API.

After that the malware iterates on the loaded **Windows DLLs** through the **K32EnumProcessModules** APIs to unhook each DLL and evade active **EDR**s on the system. Basically, for each loaded DLL, the **.text** section of each of them is freshly mapped to the virtual address of the possible hooked DLL.

From this point the malware enters a **pseudo-infinite** loop where, each second, goes to contact the **Dropbox** service to communicate the victim identifier and receive next-stage payloads.

First of all, the **api.dropbox.com** endpoint is contacted at the **/oauth2/token/** URI through an **HTTP POST** request to receive a **r**efresh token, necessary to contact the **Dropbox** APIs.

For this request the following combination of **API key** and **API secret** are used to represent the Dropbox account used by the threat actor:

- API key: **fm09ogco339u0a9**
- API secret: **scqekoaqqj98sze**

In addition, for all the network-related requests the malware uses a **fixed user-agent**:

> *User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.75 Safari/537.36 AtContent/91.5.2444.45*

If the refresh token is received from the **Dropbox API**s and successfully parsed, the implant proceeds with the integration of some masquerading and persistence techniques. More in details, a new subprocessis created to open the lure PDF document (an empty PDF) contained in the bundle and used to make the user think that he has opened the legit **Adobe Acrobat** application.

In the meanwhile, all the files involved in the bundle (signed software and relative DLL's) are **copied** under the **%APPDATA%\AcroSup** directory and a new registry key under **HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run** is created to achieve persistence.

The lure PDF document is not copied under the above mentioned directory so this document will not be opened again after a restart of the victim system. To register the victim entry on **Dropbox**, a victim identifier is created through the **hex-encoded** combination of the current usernameand current computer name (e.g., *john::windows10*). The communication of the new victim is then completed through a push of a new **.mp3** file(named *Rock_[VICTIM_ID].mp3*) via the **Dropbox APIs** on **/2/files/upload/** URI.

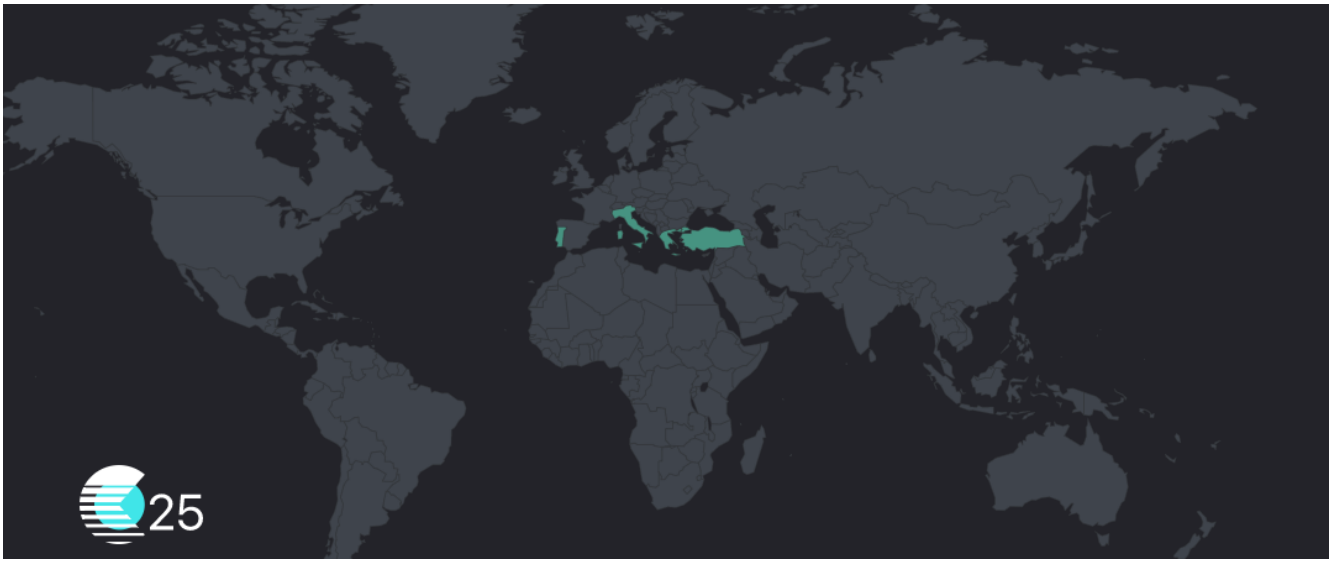
Interestingly, the pushed **.MP3** files identifying the victims, contains always the following string that likely represents the malware family:

> *ME3.99.5UUUUUUUUUU*

Finally, another request is performed to the **/2/files/download/** path of **Dropbox APIs** that tries to download a file named *Rock_VICTIM_ID.mp3.backup*. The response is parsed to determine if a **next-stage** payload exists for the current registered victim. If one does, the malware will allocate a new heap space to store the downloaded payload and will execute it in the executable space by overwriting again the thread context of the main executable.

VICTIMOLOGY

In recent months Cluster25 had evidence of **Cozy Bear**'s campaigns that potentially impacted *at least* **Greece**, **Italy**, **Turkey** and **Portugal** especially in government and foreign affairs sectors.



CONCLUSIONS

NOBELIUM confirms its interest in government and foreign affairs by targeting organizations in **Europe** and possibly other parts of the world. The campaigns and the payloads analyzed over time show a strong focus on operating under the radar and lowering the detection rates. In this regard, even the use of legitimate services such as **Trello** and **DropBox** suggest the adversary's will to operate for a long time within the victim environments remaining undetected. It is possible to foresee that this actor will also try to change TTPs (Technical and Tactical Procedures) in the near future in order to make any mitigation action aimed at its contrast more difficult. In this regard, we provide a wide set of detection rules useful for verifying potential malicious activities attributable to this threat actor (see DETECTION AND THREAT HUNTING SECTION).

ATT&CK MATRIX

TACTIC	TECHNIQUE	DESCRIPTION
Initial Access	T1566.001	Phishing: Spearphishing Attachment
Execution	T1204.001	User Execution: Malicious Link
Execution	T1204.002	User Execution: Malicious File
Execution	T1059.007	Command and Scripting Interpreter: JavaScript
Defense Evasion	T1036	Masquerading
Defense Evasion	T1622	Debugger Evasion
Defense Evasion	T1140	Deobfuscate/Decode Files or Information
Defense Evasion	T1027	Obfuscated Files or Information
Defense Evasion	T1055.003	Process Injection: Thread Execution Hijacking
Defense Evasion	T1553.002	Subvert Trust Controls: Code Signing
Defense Evasion	T1562.001	Impair Defenses: Disable or Modify Tools

Defense Evasion	T1112	Modify Registry
Defense Evasion	T1202	Indirect Command Execution
Defense Evasion	T1497	Virtualization/Sandbox Evasion
Defense Evasion	T1620	Reflective Code Loading
Discovery	T1082	System Information Discovery
Discovery	T1057	Process Discovery
Persistence	T1098	Account Manipulation
Persistence	T1547.001	Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder
Command and Control	T1105	Ingress Tool Transfer
Command and Control	T1071.001	Application Layer Protocol: Web Protocols
Command and Control	T1102	Web Service

INDICATORS OF COMPROMISE

CATEGORY	TYPE	VALUE
PAYLOAD	SHA256	5292c0f5a7ea80124cf7584eacea1881cf2f0814fa13dcc0de56624e215aaba2
PAYLOAD	SHA1	32792827c14075cc3091244425e302b1ebe3259c
PAYLOAD	MD5	2fbccfc5a1b91b2609e3ae92a93ff7cb
PAYLOAD	SHA256	9d063a05280fbce6ff0fd62a877f3fd1e80f227522e16918e6bede2e6ee398de
PAYLOAD	SHA1	05241afa180d70e17647b2d8cbc1660adbe3af88
PAYLOAD	MD5	d86283af2d5888b0ce3ea63eb26f60f7
PAYLOAD	SHA256	4c68c840ae1a034d47900ebdc291116726fd37b3ab0b7e026fad90eaab84d820
PAYLOAD	SHA1	c9a5314eb247c7441a5262a7cd22abbe1fcba7b6
PAYLOAD	MD5	110c4ae194e7b49ed3e3b254d599f7f4
PAYLOAD	SHA256	7f96d59cb02229529b14761f979f710bca500c68cc2b37d80e60e751f809475e
PAYLOAD	SHA1	489c36c9ea3fb90f61209d43efffd8d997a362c6
PAYLOAD	MD5	9ec1fcb11b597941bec03078cccab724
PAYLOAD	SHA256	23a09b74498aea166470ea2b569d42fd661c440f3f3014636879bd012600ed68
PAYLOAD	SHA1	ad33bab4bc6232a6666c2190b3bf9fc2ab2a720a

PAYLOAD	MD5	454f59dc7d3d7f228bbd4ddd4c250ed8
PAYLOAD	SHA256	729fb24b6c18232fc05ccf351edaeaa8a76476ba08cba37b8a93d34f98fa05ed
PAYLOAD	SHA1	900cba1d73ddca31a7bb7b7af5b3b7f1a0bc6fbf
PAYLOAD	MD5	6bc8be27898e1e280e402a7981be55ae

DETECTION AND THREAT HUNTING

SNORT #SSL_DECRYPT_ONLY

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"CLUSTER25 NOBELIUM Registration via Dropbox API"; flow:established,to_server; http.uri; content:"/2/files/upload"; http.header; content:"|22|path|22|"; content:"|22|/Rock_"; fast_pattern; distance:0; content:".mp3|22|"; distance:0; http.host; content:"content.dropboxapi.com"; bsize:22; reference:url,cluster25.io/2022/05/13/cozy-smuggled-into-the-box/; reference:md5,3f400f30415941348af21d515a2fc6a3; classtype:trojan-activity; sid:7704250; rev:1;)
```

SNORT

```
alert http $HOME_NET any -> $EXTERNAL_NET any (msg:"CLUSTER25 NOBELIUM Backdoor Download via Dropbox API"; flow:established,to_server; http.uri; content:"/2/files/download"; http.header; content:"|22|path|22|"; content:"|22|/Rock_"; fast_pattern; distance:0; content:".mp3.backup|22|"; distance:0; http.host; content:"content.dropboxapi.com"; bsize:22; reference:url,cluster25.io/2022/05/13/cozy-smuggled-into-the-box/; reference:md5,3f400f30415941348af21d515a2fc6a3; classtype:trojan-activity; sid:7704251; rev:1;)
```

YARA

```
import “pe”
rule APT29_Loader_87221_00001 {
   meta:
      author = “Cluster25”
tlp = “white”
      description = “Detects DLL loader variants used in Nobelium kill-chain”
      hash1 = “6fc54151607a82d5f4fae661ef0b7b0767d325f5935ed6139f8932bc27309202”
      hash2 = “23a09b74498aea166470ea2b569d42fd661c440f3f3014636879bd012600ed68”
   strings:
      $s1 = “%s\\blank.pdf” fullword ascii
      $s2 = “%s\\AcroSup” fullword ascii
      $s3 = “vcruntime140.dll” fullword ascii
      $s4 = “ME3.99.5UUUUUUUUUU” fullword ascii
      $c1 = “Rock” fullword ascii
      $c2 = “.mp3” fullword ascii
      $c3 = “%s.backup” fullword ascii
      $sequence1 = { C7 45 ?? 0B 00 10 00 48 8B CF FF 15 ?? ?? ?? 00 85 C0 74 ?? 48 8D 55 ??
48 89 75 ?? 48 8B CF FF 15 ?? ?? ?? 00 85 C0 74 ?? 48 8B CF FF 15 ?? ?? ?? 00 } // Thread
contect change
      $sequence2 = { 0F B6 0B 4C 8D 05 ?? ?? ?? 00 89 4C 24 ?? 4D 8B CD 49 8B CD BA 04 01 00
00 E8 ?? ?? ?? ?? 48 8D 5B 01 48 83 EF 01 75 ?? } // encoding cycle
      $sequence3 = { 4C 8D 8C 24 ?? 00 00 00 8B 53 ?? 44 8D 40 ?? 48 03 CD 44 89 A4 24 ?? 00
00 00 FF 15 ?? ?? ?? 00 8B 43 ?? 44 8B 43 ?? 4A 8D 14 38 48 8D 0C 28 E8 ?? ?? 00 00 8B 4B ??
4C 8D 8C 24 ?? 00 00 00 8B 53 ?? 48 03 CD 44 8B 84 24 ?? 00 00 00 FF 15 ?? ?? ?? 00 } //DLL
Unhook
      $sequence4 = { 42 0F B6 8C 32 ?? ?? ?? 00 48 83 C2 03 88 0F 48 8D 7F 01 48 83 FA 2D 7C
E7 } // get domain name string
   condition:
      uint16(0) == 0x5a4d and filesize < 200KB
         and pe.imports(“kernel32.dll”, “SetThreadContext”) and pe.imports(“kernel32.dll”,
“ResumeThread”) and pe.imports(“kernel32.dll”, “K32GetModuleFileNameExA”)
         and 3 of ($s*)
         and all of ($c*)
         and 3 of ($sequence*)
}
```

YARA

```
rule APT29_HTMLSmuggling_ZIP_82733_00001 {
meta:
author = “Cluster25”
description = “Rule to detect the EnvyScout HTML smuggling with ZIP payload used in the
APT29/Nobelium APT29 chain”
date = “2022-05-12”
hash = “d5c84cbd7dc70e71f3eb24434a58b2f149d0c39faa7e4157552b60c7dbb53d11”
strings:
$s1 = “new Blob(“
$s2 = “new Uint8Array(“
$s3 = “application/octet-stream”
$t1 = “saveAs(“
$t2 = “download(“
$r1 = { 66 6F 72 28 76 61 72 20 69 20 3D 20 30 78 30 3B 20 69 20 3C 20 64 5B 27 6C 65 6E 67 74
68 27 5D 3B 20 69 2B 2B 29 20 7B 0A 20 20 20 20 64 5B 69 5D 20 3D 20 64 5B 69 5D }
condition: (filesize > 500KB and all of ($s*) and ($t1 or $t2) and $r1)
}
```

SIGMA

```
title: Potential NOBELIUM APT persistence by detection of registry key events (via registry_event)
status: stable
description: This rule detects potential NOBELIUM APT persistence via registry event
author: Cluster25
date: 2022/04/27
references:
– internal research
tags:
– attack.persistence
logsource:
  product: windows
  category: registry_event
detection:
  selection:
    TargetObject|contains:
    – '\Software\Microsoft\Windows\CurrentVersion\Run\'
    Details|endswith:
    – '\AppData\Roaming\AcroSup\Acro.exe'
  condition: selection
falsepositives:
– unknown
level: high
```

Written by: Cluster25