hardsecure

CASE STUDY

MALWARE & FORENSIC ANALYSIS

TO A RANSOMWARE ATTACK

IDENTIFICATION OF THE ORIGIN OF THE INCIDENT

Rua Acácio de Paiva nº 16, 1ºdir,
1700-006 Lisbon, Portugal

www.hardsecure.com
geral@hardsecure.com

INDEX

INDEX OF FIGURES

CONTEXT

Hardsecure was contacted by customer X because one of their servers was compromised in a ransomware attack, where the customer sought to determine the origin of the intrusion.

The early stage of artefact collection:

- Customer: "We are no longer able to access the servers, AD accounts aren´t working. We are able to change the admin password and restore access".
- It was detected in one of the machines that had the files encrypted, in our 1st analysis, were found in only one machine with ransomware.
- Only after analyzing the firewall logs they realized an alarm was triggered by it at May 17th at 11:48 p.m, note that the dates of the systems analyzed sometimes change by one hour.
- Although a snapshot of the machine was made for analysis, this snapshot was already made after a "cleaning" of malware of the machine.
- It hadn´t centralized logs, only those located in the firewall.
- Something that intrigued the customer was the fact that this machine had no services exposed to the Internet.

When Hardsecure was contacted, the customer at this point had already sanitized the infrastructure and user accounts.

Hardsecure was given access to the client's infrastructure so that it could start the investigation process.

ANALYSIS

The machine, **10.0.0.20 (Xserver)**, was restored from a snapshot so that Hardsecure could analyze it. After the artefacts of the system were extracted for forensic analysis purposes, after a live analysis of the machine a file, Del. cmd was immediately found and it was at the machine's autoruns and, therefore, made to be executed as soon as the machine started.

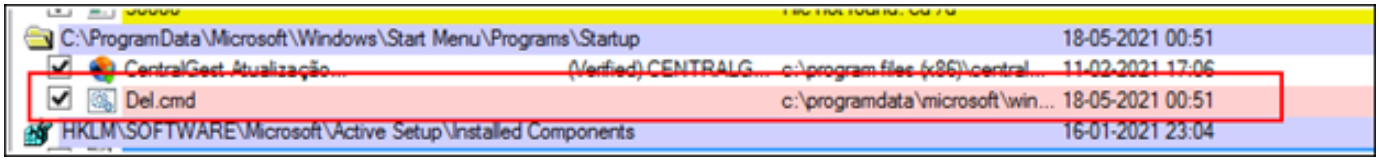

Figure 1 - Machine autoruns

This same file is a script that will eliminate an executable in the path: **C:\Users\Public\Videos\Sys.exe**

During the process of search for artefacts in this folder, other files related to the ransomware were also identified:

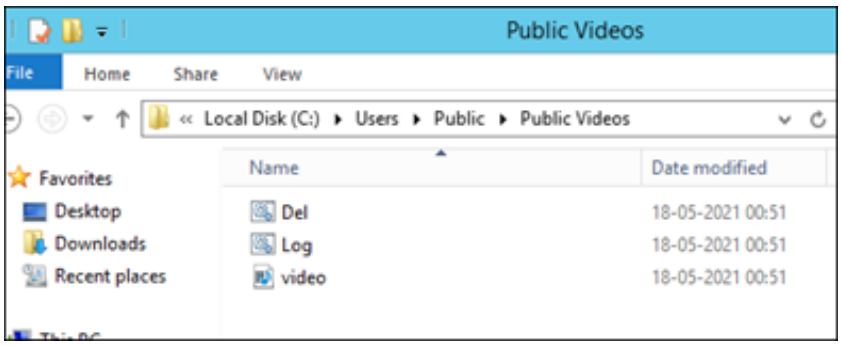

Figure 2 - Files in folder

So here we already had some evidence that there was malicious software running in the machine.

The **SYS.exe** file was no longer in the folder, the log.cmd file is used to erase the machine's logs, the **del.cmd** file is the same one found in the autoruns and the **video.mp4** is a video showing the ransomware in execution and how they manage to encrypt and decrypt the files.

Figure 3 - Del.cmd

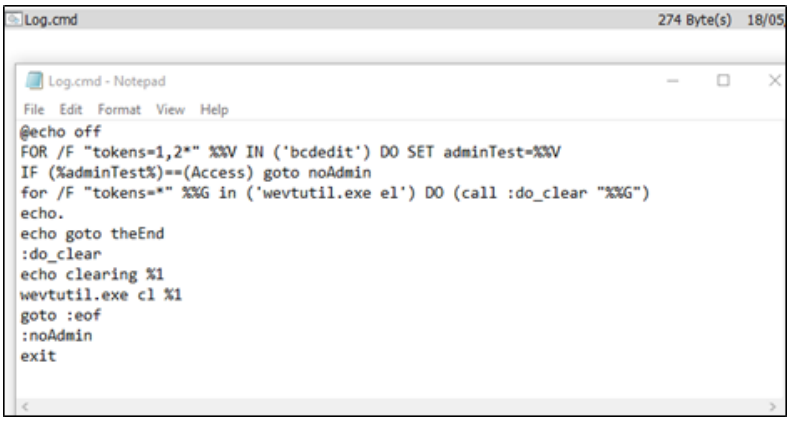

Figure 4 - Log.cmd

Next, having this information, Hardsecure ran the KAPE tool, produced by Eric Zimmerman (@EricRZimmerman), a tool that will extract a vast amount of artefacts from the system for further analysis.

After extracting the artefacts, the tool allows, among other tasks, the creation of a global timeline joining all the system's Event Logs, which allows a better understanding of what is happening in the system at each moment.

It was then in this analysis that two major events were identified:

- May,17, 2021 11:52:03 p.m

» Logs cleared by user **batchaccount**

- May *17, 2021 11:54:15 p.m*

» Remote Desktop Services: Session has been disconnected for user ***DOMAIN\batchaccount*** *address* ***10.0.0.21***

Therefore the logs were deleted at 11:52 p.m, not allowing us to see what happened before, however later, it was identified at 11:54 p.m the logoff of an account and a source IP, in which it is assumed that it was this user who connected and deleted the logs, partially hiding his actions.

Before continuing the investigation to server **10.0.0.21 (WebServer)**, we proceeded to access machine **10.0.0.10 (DCServer)** to extract the artefacts from it, to search for evidence of malicious activity, where we identified this entry in the logs:

- May,*17, 2021 11:16:59 p.m*

» *Remote Desktop Services: Session logon succeeded for account* ***DOMAIN\batchaccount*** *address* ***10.0.0.20.***

By analyzing the event, we verify that the connection was made from the first machine analyzed, the **10.0.0.20 (Xserver)**, but at an earlier stage than the one we have in the logs, as they were deleted, so we move on to the next machine to be analyzed, the **10.0.0.21 (WebServer)**.

Then, again, we extract the artefacts from this machine, **10.0.0.21 (WebServer)**, and since it is a web server, we also extract the IIS logs.

By analyzing the IIS logs by alarm date, it was not possible to isolate any particular request or set of requests.

We then proceeded to analyze the timeline of events and it was in this machine that we found the events closest to the time of the alarm triggered by the firewall, at 10:45 p.m, and it was the sequence of the next events that revealed what happened:

- May,*7, 2021 10:45:52 p.m*

» *ASP.NET application crash report, IIS APPPOOL\\XApp, failed to load file* ***file:///C:/Windows/Temp/1621291550.8233094.dll***

- May, *17, 2021 10:46:53 p.m*

» *FW rule added to exception list, *: *, : RDP Port 3389, Direction: 1, ModifyingApplication: C:\\Windows\\System32\\netsh.exe*

- May *17, 2021 10:48:54 p.m*

» *Remote Desktop Services: Session logon succeeded, account* ***WEBSERVER\Administatr****, address* ***::%16777216***

Now in this sequence, we saw that there was a crash when trying to load a library, shortly after it is created an exception in the firewall to allow all **RDP** traffic and login in the **Administrator** account with an address that seems corrupted/unknown.

The first crash when loading the library was not immediately obvious, and so we started by investigating if the firewall rule existed, and indeed it was active. After that we analyzed the **Administatr** account, so similar in name to a legitimate administrator account, and found that it was created according to the registry entries at 10:46 p.m, shortly before the first login, with the unknown address:

CreatedOn	LastLoginTime	LastPasswordChange	LastIncorrectPasswo	ExpiresOn	UserName	FullName	Password	Groups
17/05/2021 22:46:52	17/05/2021 23:10:46	17/05/2021 22:46:52			Administatr			Administrators, Users

Figure 5 - Registry creation date of the Administatr account

In the Desktop folder of this account were found traces of the use of **mimikatz**. In addition to a port scanner, **mimikatz** allows the extraction of passwords stored on the machine or in its memory, and the port scanner allows finding open ports on other machines, and it was in the results of **mimikatz** that we found the **batch account** with the password in clear, this was the account that we identified in the logs of the first analyzed machine logging off and deleting the system logs.

file:///C:/Users/Administatr/Desktop/KPortScan%203.0/results.txt	17/05/2021 23:52:56	1
file:///C:/Users/Administatr/Desktop/KPortScan%203.0[sentorion].zip	17/05/2021 23:52:15	1
file:///C:/Users/Administatr/Desktop/mimikatz_trunk.zip	17/05/2021 23:49:50	1

Figure 6 - Traces of mimikatz and port scanner, in the log of accessed files

HiveTy	Description	Category	KeyPath	ValueName	ValueT	ValueData	ValueData2
NtUser	UserAssist	Program Execution	CsiTool-CreateHive-	Zvpebfbsg.Nhgb1	(plugin)	Microsoft.AutoGenerated.{Unmapped GUID: 923DD477-5846-686	Last executed:
NtUser	UserAssist	Program Execution	CsiTool-CreateHive-	Zvpebfbsg.Jvaqbj	(plugin)	Microsoft.Windows.Explorer	Last executed:
NtUser	UserAssist	Program Execution	CsiTool-CreateHive-	P:\Hfref\Nqzvavf	(plugin)	C:\Users\Administatr\Desktop\x64\mimikatz.exe	Last executed: 2021-05-17 22:49:58.3700000
NtUser	UserAssist	Program Execution	CsiTool-CreateHive-	{1NP14R77-02R7-	(plugin)	{System32}\wuauclt.exe	Last executed:
NtUser	UserAssist	Program Execution	CsiTool-CreateHive-	P:\Hfref\Nqzvavf	(plugin)	C:\Users\Administatr\Desktop\KPortScan 3.0\KPortScan3.exe	Last executed: 2021-05-17 22:52:22.6760000

Figure 7 - Traces of mimikatz and port scanner execution in the registry

We assume then that it was from this machine and with the credentials obtained by **mimikatz** that they gained access to the other machines.

After collecting this artifact, we went looking for what was executed which allowed the creation of a local account in the **Administrators** group and the creation of the exception in the firewall. By analyzing the log about the crash when loading the **.dll** library, we went to see if the **.dll** file was present in the folder where the application tried to load it, and as soon as we accessed the **C:\Windows\Temp** folder, the antivirus it immediately detected the file **1621291550.8233094.dll** as **malicious**.

At this point, we can assume two things, that the attacker managed to **upload a file** and managed to **execute code** on the system, now the question is, how?

At first, it was assumed that it could be one or two vulnerabilities in the web application and that for that, a pentest would have to be done on it to try to find them.

Later we had a surprise, and this is where the importance of keeping track of everything that is being analyzed during an investigation is highlighted, even when it seems unrelated to the rest because when reading a news article, which was about the most used vulnerabilities, we see a reference to a vulnerability in **Telerik**, the CVE-2019-18935. **Telerik** is a solution that allows accelerated creation of applications for several operating systems, and it was then that we found that **WebServer** also used **Telerik**.

Looking in more detail at the vulnerability, CVE-2019-18935, we see that there are two vulnerabilities associated with it, one **file upload** and one **remote code execution**, well, precisely what we had supposed needed to happen to validate the sequence of events.

We searched for public exploits for this **CVE** and when analyzing an exploit, we checked which URL is used:

```
"\n" +
"Decrypt a plaintext:       -d ciphertext\n" +
"Decrypt rauPostData:       -D rauPostData\n" +
"Encrypt a plaintext:       -e plaintext\n" +
"Gen rauPostData:           -E TempTargetFolder Version\n" +
"Gen rauPostData (quiet):   -Q TempTargetFolder Version\n" +
"Version in HTTP response: -v url\n" +
"Generate a POST payload:   -p TempTargetFolder Version c:\\\\folder\\\\filename\n" +
"Upload a payload:          -P TempTargetFolder Version c:\\\\folder\\\\filename url\n\n"
"Example URL:               http://target/Telerik.Web.UI.WebResource.axd?type=rau"
```

Figure 8 - URL used to exploit the vulnerability

Relating this to the error logs when the application crashed while loading the malicious library, we can see that the same URL was used:

Figure 9 - URL used during crash

With this information, we then went to the IIS logs to confirm if there were any requests at this point, and we confirmed that the vulnerability was used, precisely at the time of the web application crash:

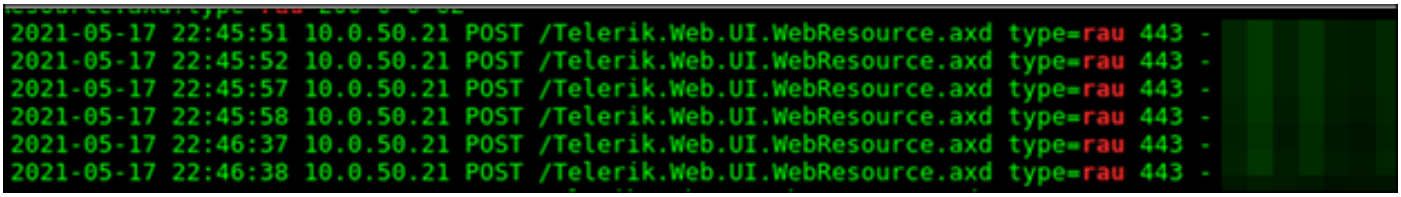

Figure 10 - HTTP requests exploiting vulnerability

We also found that there is a **Metasploit** module for this vulnerability. **Metasploit** is an offensive tool that, among other things, facilitates vulnerability exploitation, and if such was used, we can assume that the attacker used a **meterpreter** reverse shell. The **meterpreter** is a reverse shell that has many features to help an attacker after accessing the system and used its port-forwarding feature to create a tunnel that allowed him to connect by **RDP** to the machine and hence we have the creation of a rule in the firewall to allow **RDP** access, and that first login in the **Administatr** account with a source that is not an IP address, because it is coming from the machine to itself through the tunnel.

Desta forma, foi detetada a origem da intrusão, o uso de uma *framework* desatualizada e com *exploits* públicos disponíveis.

CONCLUSION

During the malware & forensics analysis process, it's critical to make notes about what you're looking at for later, after everything has been seen at least once, we can start to correlate information.

The forensic analysis involves a methodology for collecting data, analyzing them, creating assumptions, confirming the existence or non-existence of evidence to confirm the assumption, repeating the entire process.

The purpose of this analysis was not to know what had been affected or extracted, as the client had already cleaned up its infrastructure, and therefore contaminated the evidence, but rather to discover and understand the source of the problem.

We found that the incident did not start on the machine infected with ransomware, but rather, this was the culmination of an attack on the infrastructure that started by exploiting vulnerabilities in an outdated framework that was installed on a machine exposed to the Internet.

It is important to discover and understand what happened in a case of intrusion and compromise of the infrastructure so that the same flaw cannot be used again by attackers, the fact that we discovered this flaw in the customer's **Telerik**, allowed the customer to update the system and close the flaw, blocking new attackers from entering its infrastructure. If the infrastructure had only been sanitized, which is always advisable, it would be a matter of time before it would happen again, as frequent requests to the vulnerable URL were verified in the logs at the time of the investigation, not in a context of exploiting the flaw, but possibly just a verification by a third party to keep a record of vulnerable machines/infrastructures for later use.