Secrets behind the Lazarus's VHD ransomware

seguranca-informatica.pt/secrets-behind-the-lazaruss-vhd-ransomware/

August 17, 2021

Introduction

Data encryption malware is one of the most popular malware families in recent years and targets mass volumes of users and companies around the world. In this article, we will take a deep dive into a new VHD ransomware distributed in the wild by the Lazarus group — the criminals behind the WannaCry incident in 2017.

The VHD ransomware is one of the most dangerous threats today. This is because it tries to change system settings, disable some programs and functions of the target operating systems, and demands payment in cryptocurrency via a ransom note dropped during the ransomware execution. The Lazarus group have been using this ransomware, which uses methods typical of APT attacks but specialized in financial cybercrime.

The activity of the Lazarus Group surged in 2014 and 2015, where custom-tailored malware was used in its cyberattacks. The group has been linked to several major cyberattacks, including the 2014 Sony Pictures hack, several SWIFT banking attacks since 2016, and the WannaCry ransomware infection in May 2017.

Lazarus's VHD ransomware

Researchers from Kaspersky detailed a new VHD ransomware used by the group between March and May 2020. The analyzed samples have been deployed over the network of the target enterprises, brute-forcing the SMB service on every discovered machine and using the

MATA malware framework documented by Kaspersky here and also used by the Lazarus group.

Initially, criminals gain access to the target infrastructures via widespread botnet infections such as Emotet and Trickbot families or by exploiting vulnerable VPN gateways. When criminals have a good understanding of the target's finances and IT processes and network, they escalated their privileges on the compromised systems and installed a backdoor that is part of the MATA malware framework. The criminals leverage the backdoor to control the Active Directory server and use it to deliver VHD ransomware to all systems in the target network within 10 hours using the help of a Python-based loader.

After finishing the infection chain with the ransomware deployment, criminals enter a new phase: negotiation.

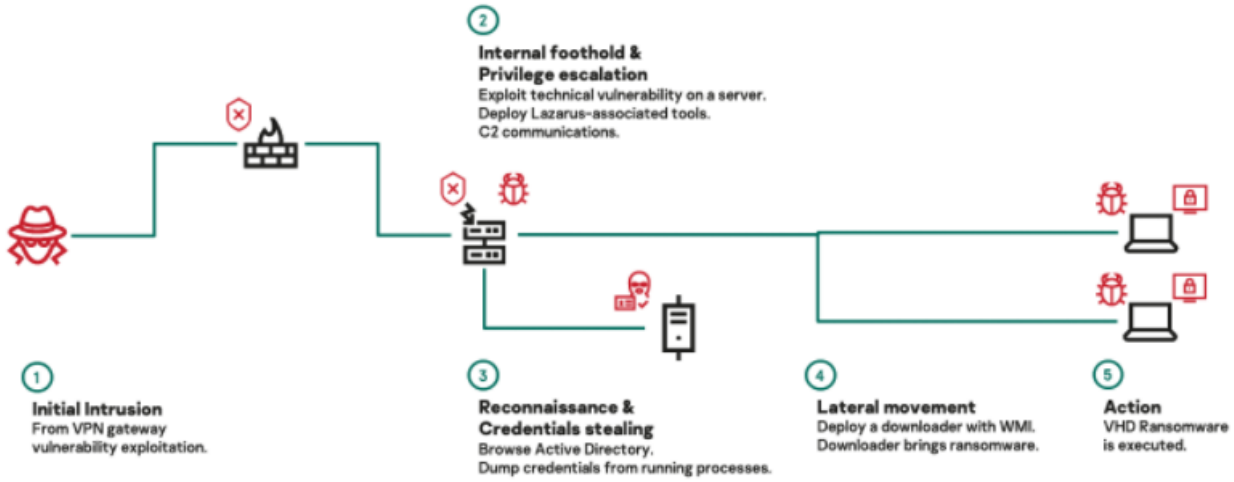


Figure 1: *High-level diagram of the Hakuna MATA and VHD ransomware.*

VHD is a standard ransomware tool that spreads through the drives connected to the target device, encrypts all the files and deletes all System Volume Information folders that prevent the impacted system could be restored. The ransomware binary is protected with VMProtect packer in order to prevent its analysis. The binary sections vmp0, vmp1 are the first signal of this packer.

.text	00013505	00001000	00013600
.rdata	000053C0	00015000	00005400
.data	000035F0	0001B000	00001400
.vmp0	003306CF	0001F000	00330800
.vmp1	0026F350	00350000	0026F400
.reloc	000030EC	005C0000	00003200
.rsrc	000001B2	005C4000	00000200

Figure 2: *Binary sections (vmp0 and vmp1 — VMProtect packer).*

As observed in the PorEx diagram below, some parts of the binary were obfuscated, such as the malware config, ransom note details and so on.

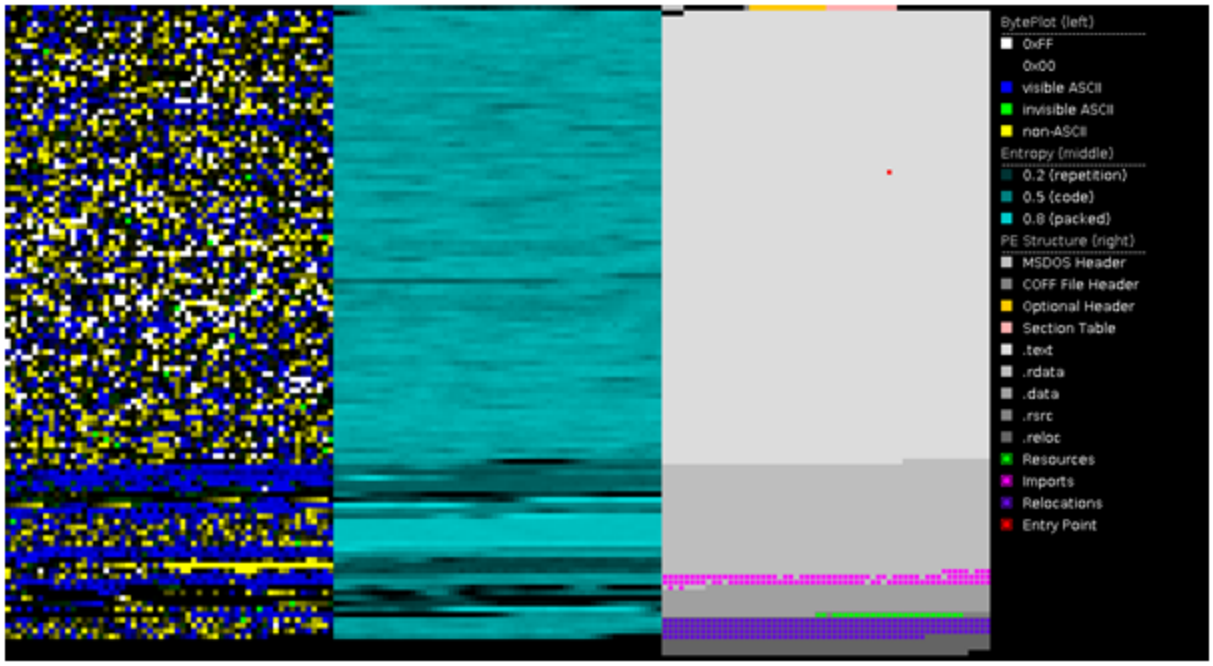


Figure 3: *PortEx diagram — VHD ransomware.*

One interesting detail this threat is related to some services that are suspended to preserve important files from modification such as Microsoft Exchange and SQL server. Next, the services suspended during the ransomware operation are presented.

```
.rdata:00418B0E                 align 10h
.rdata:00418B10 aScStopMicrosof db 'sc stop "Microsoft Exchange Active Directory Toplogy"',0
.rdata:00418B10                                         ; DATA XREF: .text:00405131↑o
.rdata:00418B46                 align 4
.rdata:00418B48 aScStopMicros_0 db 'sc stop "Microsoft Exchange Anti-spam Update"',0
.rdata:00418B48                                         ; DATA XREF: .text:00405139↑o
.rdata:00418B76                 align 4
.rdata:00418B78 aScStopMicros_1 db 'sc stop "Microsoft Exchange Compliance Audit"',0
.rdata:00418B78                                         ; DATA XREF: .text:00405141↑o
.rdata:00418BA6                 align 4
.rdata:00418BA8 aScStopMicros_2 db 'sc stop "Microsoft Exchange Compliance Service"',0
.rdata:00418BA8                                         ; DATA XREF: .text:00405149↑o
.rdata:00418BD8 aScStopMicros_3 db 'sc stop "Microsoft Exchange DAG Management"',0
.rdata:00418BD8                                         ; DATA XREF: .text:00405151↑o
.rdata:00418C04 aScStopMicros_4 db 'sc stop "Microsoft Exchange Diagnostics"',0
.rdata:00418C04                                         ; DATA XREF: .text:00405159↑o
.rdata:00418C2D                 align 10h
.rdata:00418C30 aScStopMicros_5 db 'sc stop "Microsoft Exchange EdgeSync"',0
```

Figure 4: *Services stoped during the ransomware execution.*

Here's a complete list of stoped services during VHD ransomware execution:

```
sc stop "Microsoft Exchange Anti-spam Update"
sc stop "Microsoft Exchange Active Directory Toplogy"
sc stop "Microsoft Exchange Compliance Audit"
sc stop "Microsoft Exchange Compliance Service"
sc stop "Microsoft Exchange DAG Management"
sc stop "Microsoft Exchange Diagnostics"
sc stop "Microsoft Exchange EdgeSync"
sc stop "Microsoft Exchange Frontend Transport"
sc stop "Microsoft Exchange Health Manager"
sc stop "Microsoft Exchange IMAP4"
sc stop "Microsoft Exchange Health Manager Recovery"
sc stop "Microsoft Exchange Information Store"
sc stop "Microsoft Exchange IMAP4 Backend"
sc stop "Microsoft Exchange Mailbox Assistants"
sc stop "Microsoft Exchange Mailbox Transport Delivery"
sc stop "Microsoft Exchange POP3"
sc stop "Microsoft Exchange POP3 Backend"
sc stop "SQL Server Agent (TESTINSTANCE)"
sc stop "SQL Server (TESTINSTANCE)"
```

Some important Windows calls are also executed during the malware operation. Some of them are described in a brief way below to a better understanding of how the malware works.

Calls loaded from kernel32.dll

CreateFileW()	Creates or opens a file or I/O device during malware execution.
FindFirstFileW()	Searches a directory for a file or subdirectory with a name that matches a specific name (or partial name if wildcards are used).
FindNextFileW()	Continues a file search from a previous call to the FindFirstFile, FindFirstFileEx, or FindFirstFileTransacted functions.
GetLogicalDrives()	Retrieves a bitmask representing the currently available disk drives.
IsDebuggerPresent()	Determines whether the calling process is being debugged by a user-mode debugger (used to prevent malware running in virtual machines).
IsProcessorFeaturePresent()	Determines whether the specified processor feature is supported by the current computer (used in anti-debug techniques).
CreateMutexW()	Creates or opens a named or unnamed mutex object (used to prevent further ransomware infections).

Call loaded from shell32.dll

SHEmptyRecycleBinW() Empties the Recycle Bin on the specified drive.

Call loaded from user32.dll

ShowWindow() Sets the specified window's show state. Used to hide the ransomware window during its execution.

Every time a file is encrypted, it is renamed. The extension "**.vhd**" is appended to the end of the file name.

```
.text:00404BF7 loc_404BF7:                             ; CODE XREF: .text:00404A73↑j
.text:00404BF7                                         ; .text:00404A87↑j ...
.text:00404BF7                 mov     edx, [ebp-4B84h]
.text:00404BFD                 mov     eax, [ebp-4B80h]
.text:00404C03                 push    edx
.text:00404C04                 push    eax
.text:00404C05                 lea     edi, [ebp-120h]
.text:00404C0B                 mov     esi, ebx
.text:00404C0D                 call    sub_405AD0
.text:00404C12                 push    104h
.text:00404C17                 push    offset unk_41CF20
.text:00404C1C                 lea     ecx, [ebp-328h]
.text:00404C22                 push    104h
.text:00404C27                 push    ecx
.text:00404C28                 call    sub_4076F3
.text:00404C2D                 push    104h
.text:00404C32                 push    offset a_vhd    ; ".vhd"
.text:00404C37                 lea     edx, [ebp-328h]
.text:00404C3D                 push    104h
.text:00404C42                 push    edx
.text:00404C43                 call    sub_407B1F
.text:00404C48                 add     esp, 28h
.text:00404C4B                 push    ebx
.text:00404C4C                 call    sub_734F29
```

Figure 5: *Ransomware extension (.vhd) is appended to the file name whenever a file is encrypted.*

The ransom note is also dropped into the desktop folder "***HowToDecrypt.txt***" with the instructions for paying the ransom and recovering the damaged files.

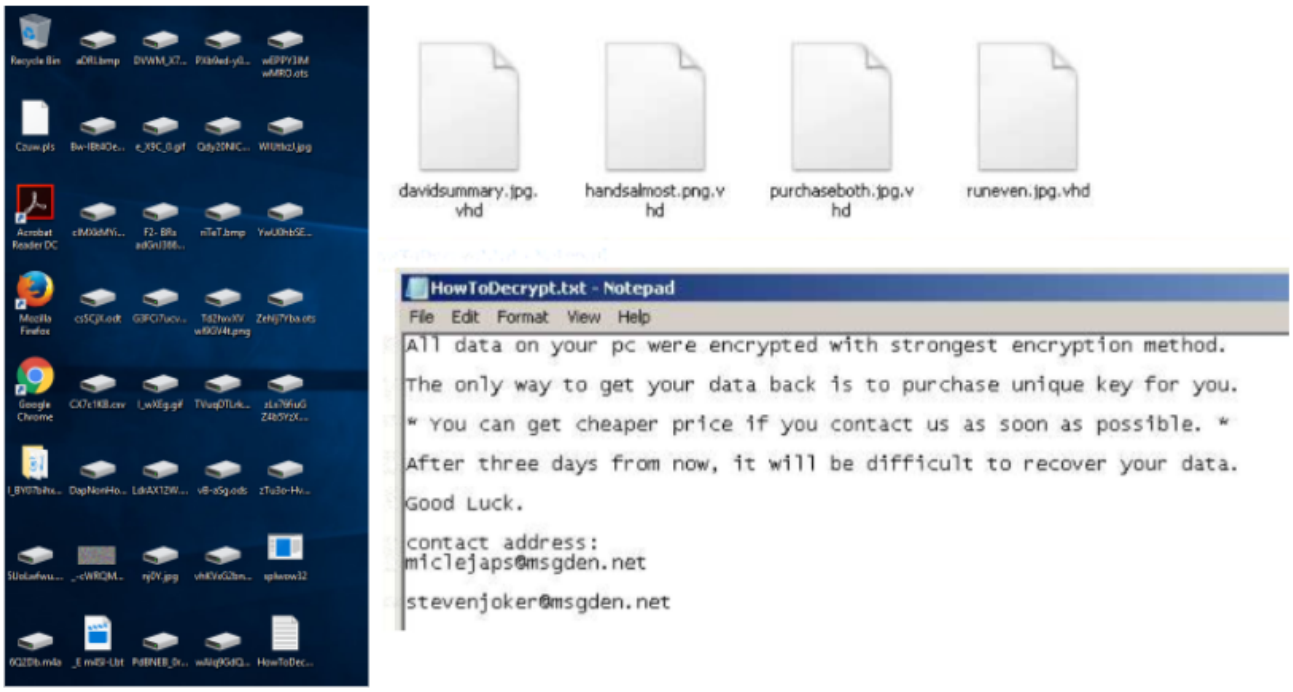
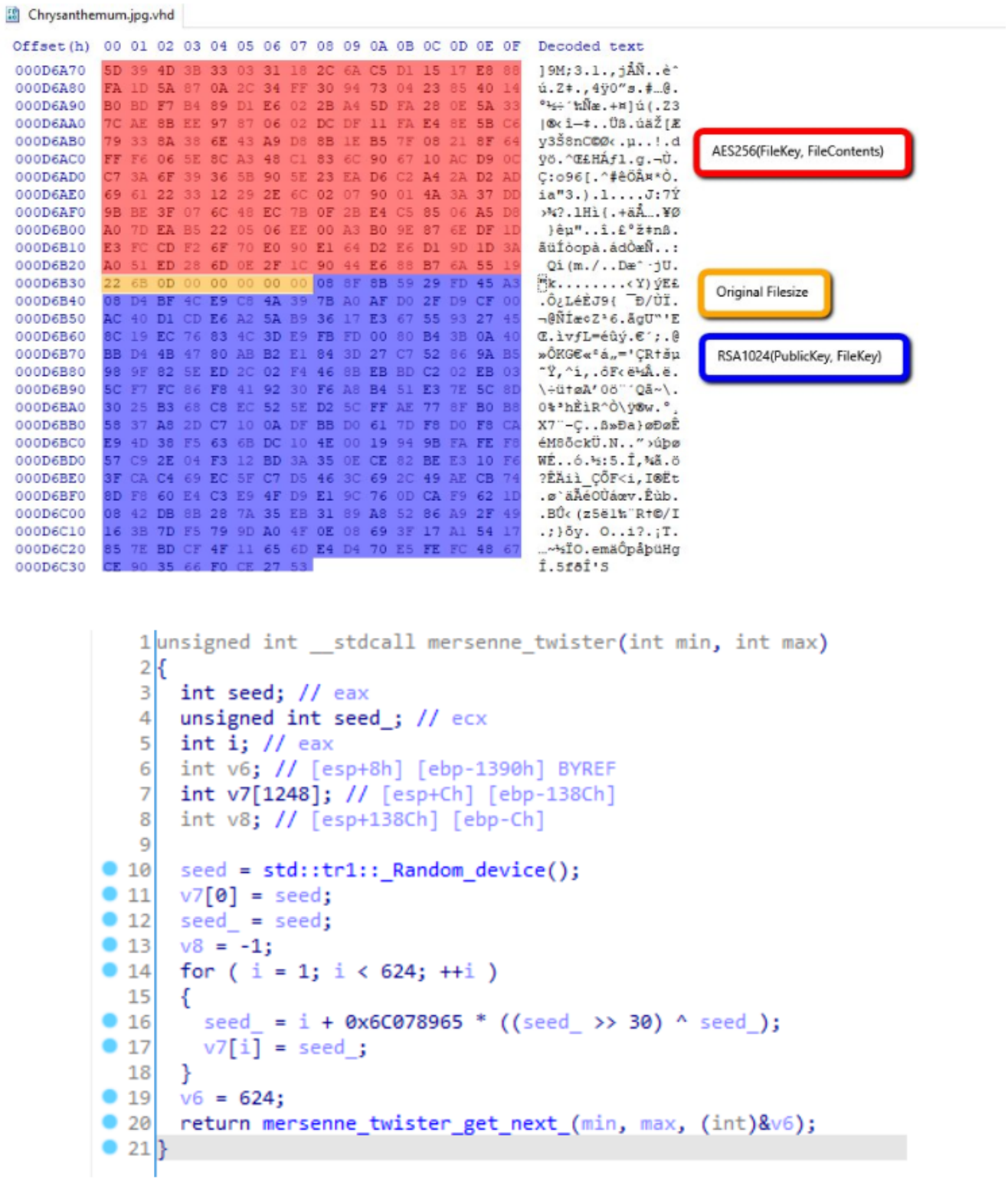


Figure 6: *Damage files and ransom note dropped into the desktop folder.*

The ransomware is written in C++ and the files are encrypted with a combination of AES-256 in ECB mode and RSA-2048. The ransomware uses Mersenne Twister as a source of randomness, but unfortunately for the victims, the RNG is reseeded every time new data is consumed. The combination of ECB and AES is not semantically secure, which means the patterns of the original clear data are preserved upon encryption.

VHD implements a mechanism to resume operations if the encryption process is interrupted. For instance, for files larger than 16MB, the ransomware stores the current cryptographic materials on the hard drive, in cleartext. This information is not deleted securely afterwards, which implies there may be a chance to recover some of the damaged files.




```
1  unsigned int __stdcall mersenne_twister(int min, int max)
2  {
3    int seed; // eax
4    unsigned int seed_; // ecx
5    int i; // eax
6    int v6; // [esp+8h] [ebp-1390h] BYREF
7    int v7[1248]; // [esp+Ch] [ebp-138Ch]
8    int v8; // [esp+138Ch] [ebp-Ch]
9
10   seed = std::tr1::_Random_device();
11   v7[0] = seed;
12   seed_ = seed;
13   v8 = -1;
14   for ( i = 1; i < 624; ++i )
15   {
16     seed_ = i + 0x6C078965 * ((seed_ >> 30) ^ seed_);
17     v7[i] = seed_;
18   }
19   v6 = 624;
20   return mersenne_twister_get_next_(min, max, (int)&v6);
21 }
```

Figure 7: *Details about cryptographic algorithms and Mersenne Twister block with the RNG seed.*

A strategy often used by malware developers is the usage of mutex entries to prevent further infections when a target device has already been infected. VHD ransomware creates during its execution the mutex ***"AEEAEE SET"***, as observed below.

```
"\Sessions\1\BaseNamedObjects\AEEAEE SET"
"AEEAEE SET"
```

The malware's plan is not very complex, as presented by the MITRE ATT&CK matrix below. Ony some functions to hide its presence and calls to scan the target device and drives and also to access the system files are used by the malicious binary.

Preventing against ransomware incidents

There is not a perfect formula to fight ransomware incidents, but some measures can be enumerated to prevent some situations:

- Provide social engineering training and explain to employees how following simple rules helps to avoid ransomware incidents.
- Ensure that software, applications and systems are up to date.
- Use endpoint protection solutions to prevent malicious infections.
- Use vulnerability management and monitoring systems to identify potential unpatched flaws and to detect incidents in real time.
- Use canary files to detect ransomware early.
- Perform cybersecurity audits and mitigate any weaknesses discovered in order to prevent attacks in the wild, both from the external and internal perspective.

The article was initially published by Pedro Tavares on resources.infosecinstitute.com.

Pedro Tavares

Pedro Tavares is a professional in the field of information security working as an Ethical Hacker/Pentester, Malware Researcher and also a Security Evangelist. He is also a founding member at CSIRT.UBI and Editor-in-Chief of the security computer blog seguranca-informatica.pt.

In recent years he has invested in the field of information security, exploring and analyzing a wide range of topics, such as pentesting (Kali Linux), malware, exploitation, hacking, IoT and security in Active Directory networks. He is also Freelance Writer (Infosec. Resources Institute and Cyber Defense Magazine) and developer of the 0xSI_f33d – a feed that compiles phishing and malware campaigns targeting Portuguese citizens.

Read more here.