Reality - ROOTCON 2020 Easter Egg Hunt

(by @katipuzer0)

Disclaimer: I am new to malware analysis so my approach for this problem is mostly by trial-and-error plus some 'googling'. My target audience for this write-up are beginners so the text might be verbose. I started working on the problem on April 15 but I gave up because I cannot find the flag. I decided to document it today April 17 and luckily I was able to find the flag after repeating the steps! I used ABIWord for this write-up to save time.

In this problem, we are given a MS Excel file(**topsecret.xls**) that is possibly malicious. To the uninformed (like me), the initial thing to do is to open it in MS Excel. Better do this in a VM!

First my working folder for this problem. I used Windows 10 and Kali.

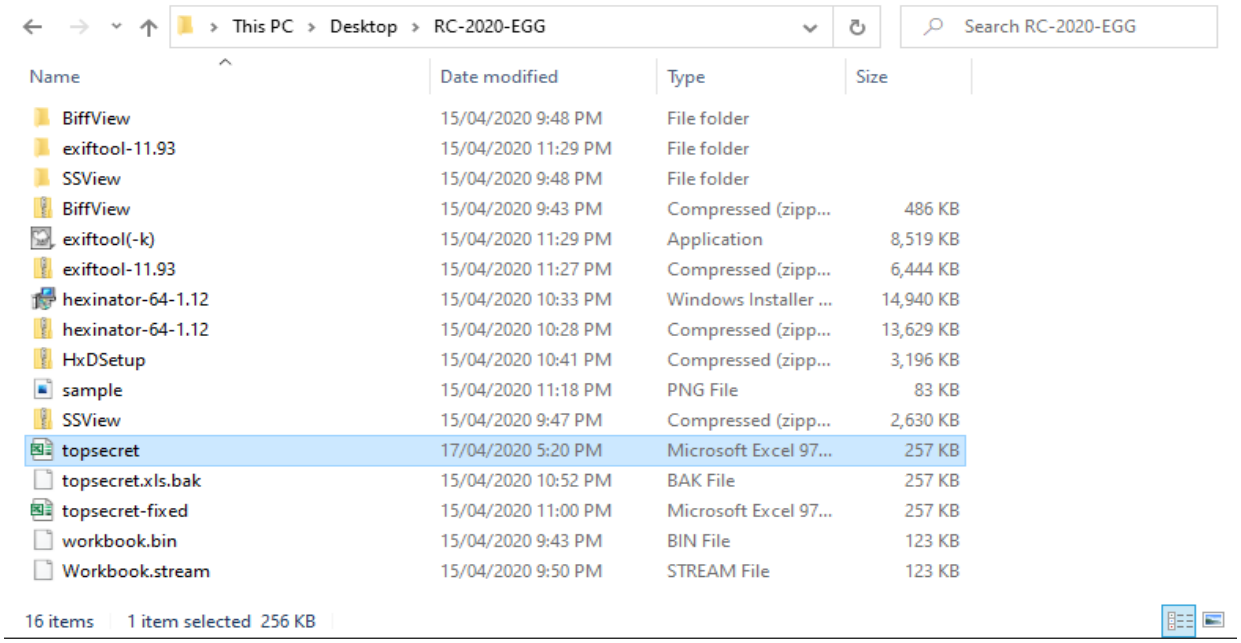

Figure 1.

This is how the spreadsheet looks like before I “enabled content”.

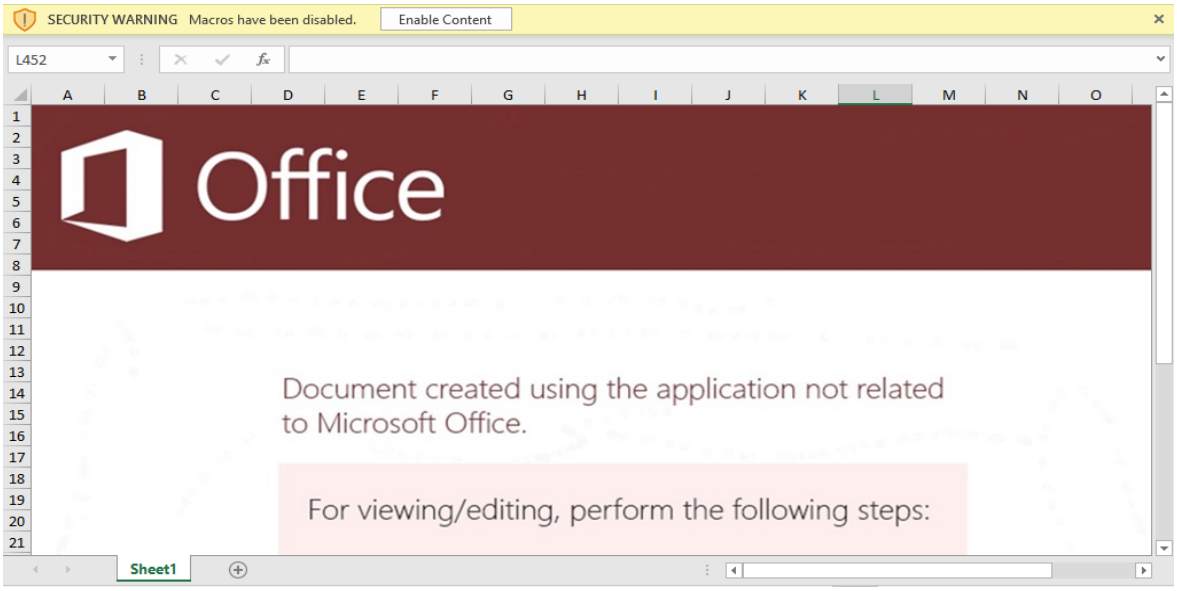

Figure 2.

After enabling the content, the calculator application appeared and a dialog box pops-up stating that the document cannot be opened. From this, we can infer that there is something fishy going on.

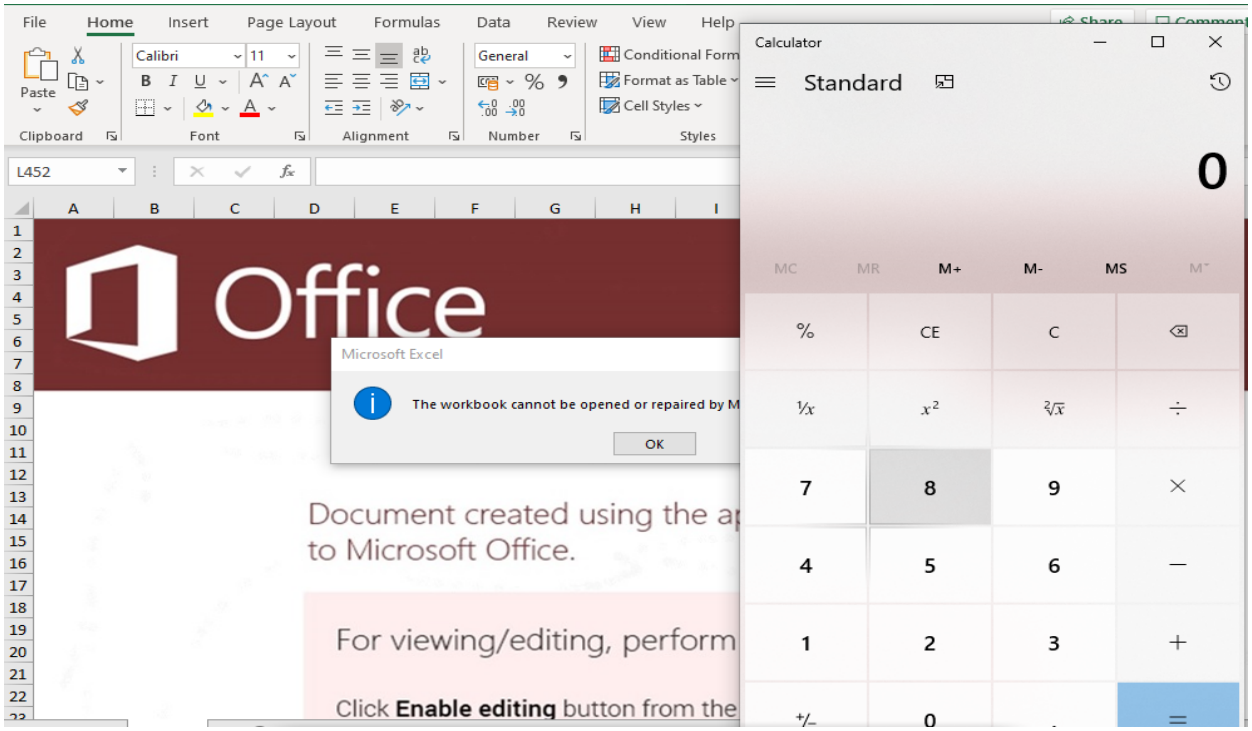

Figure 3.

Time to try some linux OLE tools [1][2]. I admit I have limited knowledge of OLE.

```
 2020-04-17 05:34:11   hagar in ~/CTFs/rc14-egg/reality
o → ole
olebrowse  olefile    olemap     oleobj     olevba
oledir     oleid      olemeta    oletimes   olevba3

 2020-04-17 05:34:11   hagar in ~/CTFs/rc14-egg/reality
o → ole
```

Figure 4.

```
 2020-04-17 05:36:53   hagar in ~/CTFs/rc14-egg/reality
o → oleid topsecret.xls
oleid 0.54 - http://decalage.info/oletools
THIS IS WORK IN PROGRESS - Check updates regularly!
Please report any issue at https://github.com/decalage2/oletools/issues

Filename: topsecret.xls
 Indicator                      Value
 OLE format                     True
 Has SummaryInformation stream  True
 Application name               Microsoft Excel
 Encrypted                      False
 Word Document                  False
 VBA Macros                     False
 Excel Workbook                 True
 PowerPoint Presentation        False
 Visio Drawing                  False
 ObjectPool                     False
 Flash objects                  0
```

Figure 5. **oleid**

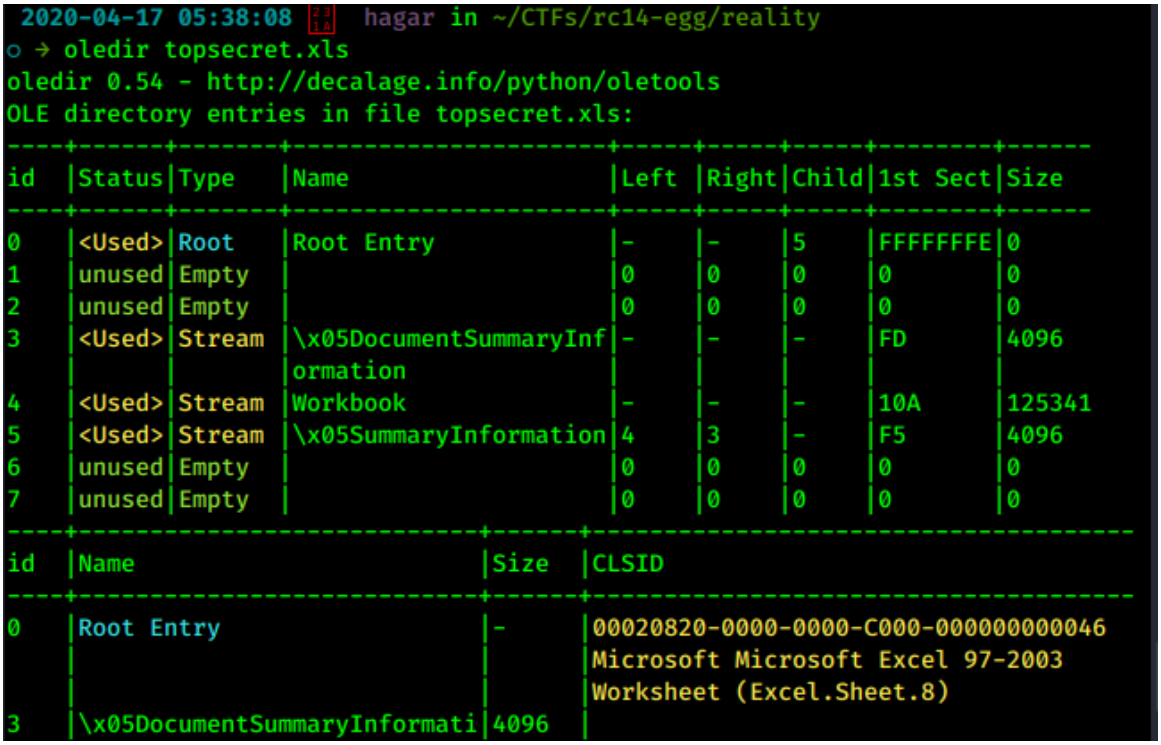

Figure 6. **oledir**

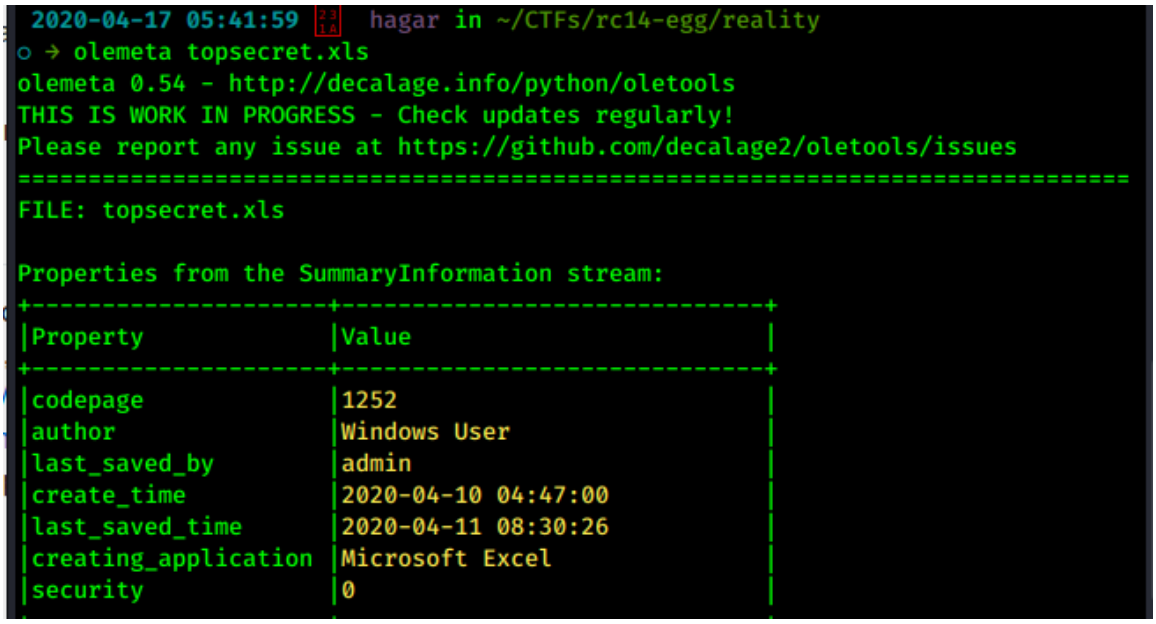

Figure 7. **olemeta**

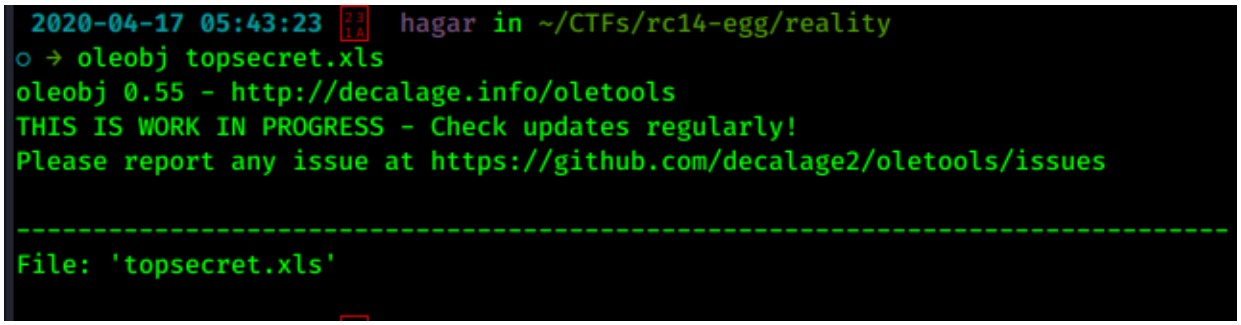

Figure 8. **oleobj**

```
 2020-04-17 05:44:50 23 1A  hagar in ~/CTFs/rc14-egg/reality
○ → oletimes topsecret.xls
oletimes 0.54 - http://decalage.info/python/oletools
THIS IS WORK IN PROGRESS - Check updates regularly!
Please report any issue at https://github.com/decalage2/oletools/issues
================================================================================
FILE: topsecret.xls

+----------------------------+---------------------+---------------+
| Stream/Storage name        | Modification Time   | Creation Time |
+----------------------------+---------------------+---------------+
| Root                       | 1601-01-01 00:05:49 | None          |
| '\x05DocumentSummaryInform | None                | None          |
| ation'                     |                     |               |
| '\x05SummaryInformation'   | None                | None          |
| 'Workbook'                 | None                | None          |
+----------------------------+---------------------+---------------+
```

Figure 9. **oletimes**

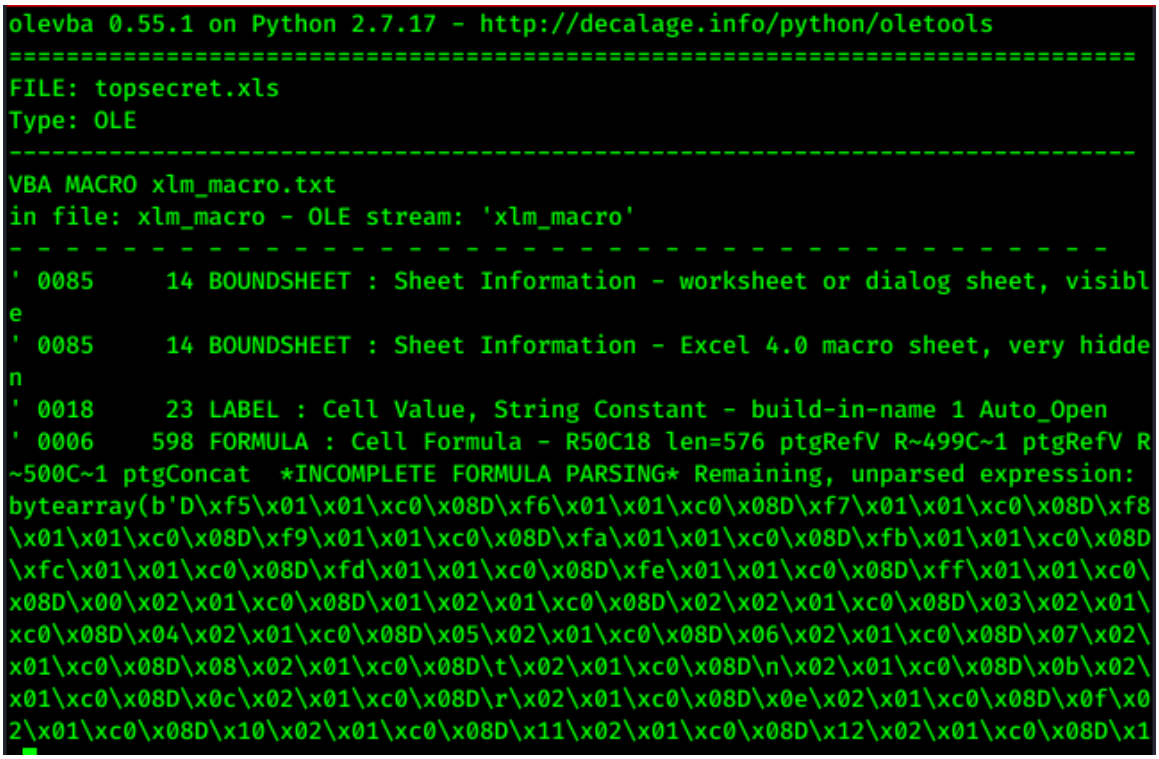

Figure 10. **olevba (1)**

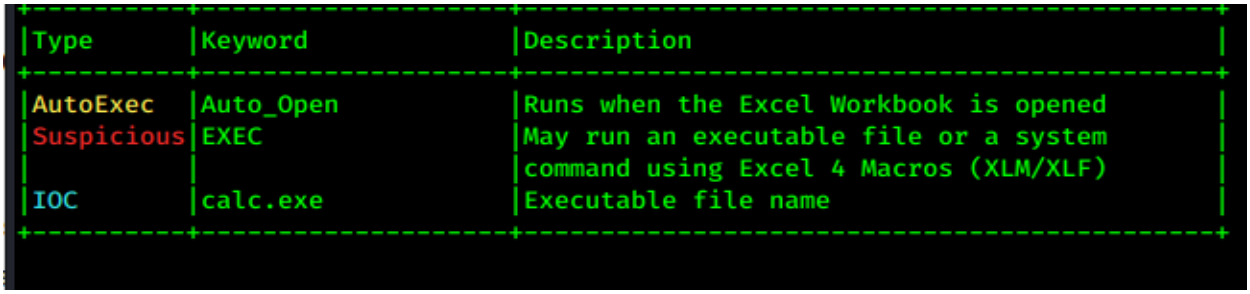

Figure 11. **olevba (2)**

So the document has some code in it that executed **calc.exe**. At this point I have no idea how to proceed so I searched for some online resources about malware in documents and I came across this blog post [3](Read this first!) The excel document given in this problem is very similar to the one discussed in this post.

I installed the necessary tools(Figure 1): *BiffView*, *SSView*, *HxD*. My key takeaway from [3] is that there is a hidden sheet in the workbook. We need to be able to access this sheet and see its contents. Observe that in Figure 2, there is only one sheet. However, in the **olevba** output there are two "**BOUNDSHEET**" entries with one marked as "**very hidden**".

I opened the document in BiffView++.

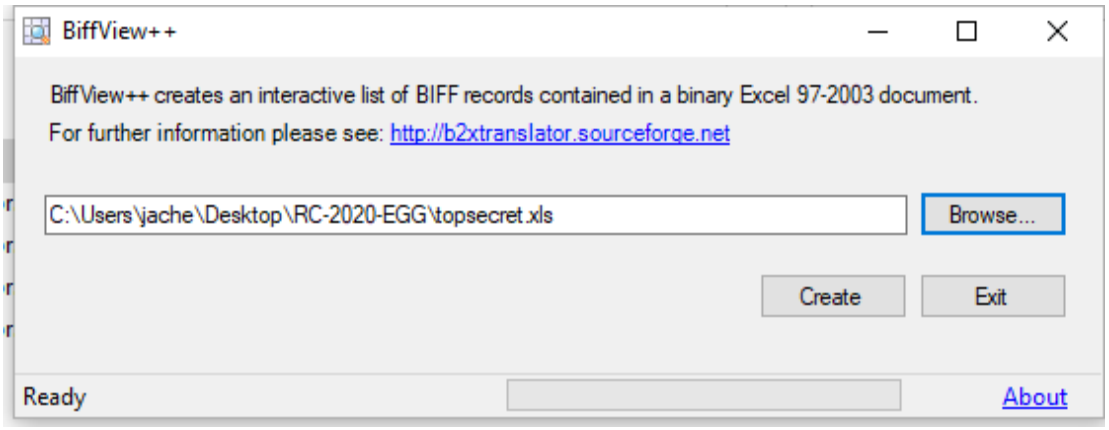

Figure 12.

```
BIFF USESELFS (160h)          2    00 00
BIFF BOUNDSHEET (85h)         14   29 5B 01 00 00 00 06 00   53 68 65 65 74 31
BIFF BOUNDSHEET (85h)         14   A3 5D 01 00 02 01 06 00   53 68 65 65 74 32
BIFF MTRSETTINGS (89Ah)       24   9A 08 00 00 00 00 00 00   00 00 00 00 01 00 00 00
                                   00 00 00 00 04 00 00 00
```

Figure 13.

The one in green is the hidden sheet because its *hsState* is *0x2*, its offset is at *0x00015DA3* based on the discussions in [3]. We open the document in HxD and searched for the offset values.

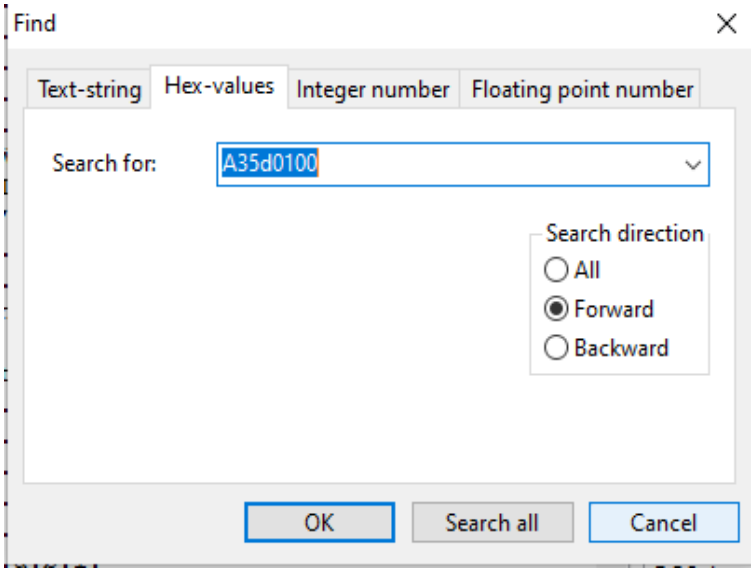

Figure 14.

topsecret.xls

Offset (h)	00	01	02	03	04	05	06	07	Decoded text
000243B0	69	00	67	00	68	00	74	00	i.g.h.t.
000243B8	31	00	36	00	60	01	02	00	1.6.`...
000243C0	00	00	85	00	0E	00	29	5B	..…...)[
000243C8	01	00	00	00	06	00	53	68	......Sh
000243D0	65	65	74	31	85	00	0E	00	eetl…...
000243D8	A3	5D	01	00	02	01	06	00	£].....
000243E0	53	68	65	65	74	32	9A	08	Sheet2š.
000243E8	18	00	9A	08	00	00	00	00	..š.....

Figure 15.

We change the value from *0x2* (very hidden) to *0x0* (visible) then save the document.

topsecret.xls

Offset (h)	00	01	02	03	04	05	06	07	Decoded text
000243B0	69	00	67	00	68	00	74	00	i.g.h.t.
000243B8	31	00	36	00	60	01	02	00	1.6.`...
000243C0	00	00	85	00	0E	00	29	5B	..…...)[
000243C8	01	00	00	00	06	00	53	68	......Sh
000243D0	65	65	74	31	85	00	0E	00	eetl…...
000243D8	A3	5D	01	00	00	01	06	00	£].....
000243E0	53	68	65	65	74	32	9A	08	Sheet2š.
000243E8	18	00	9A	08	00	00	00	00	..š.....

Figure 16.

Yay! We can now see Sheet2.

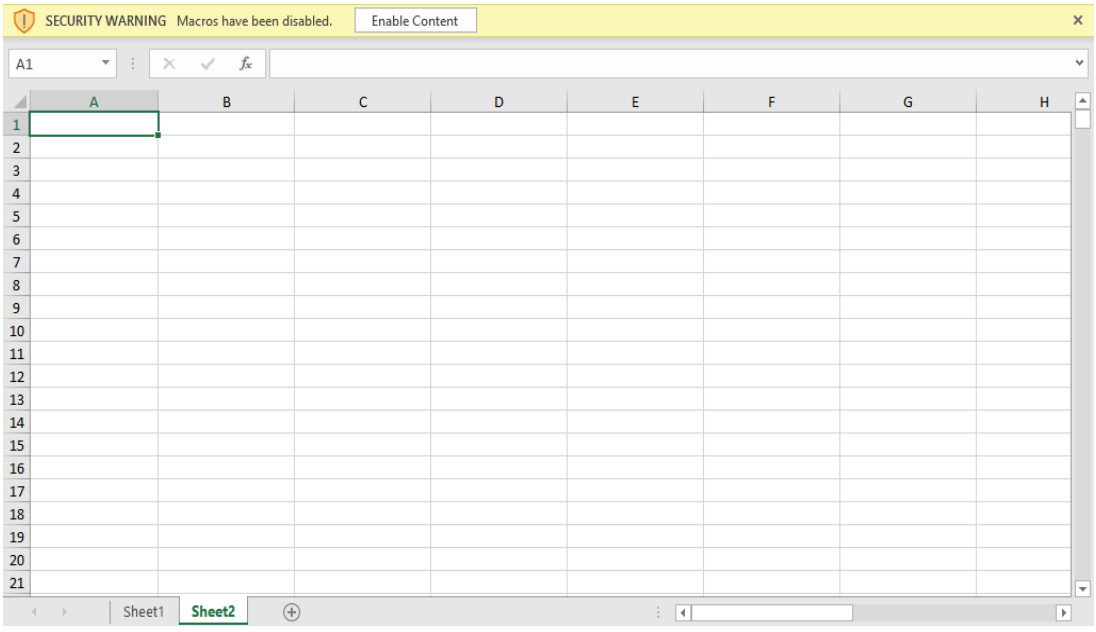

Figure 17.

Going back to the output of olevba (Figure 10), we go to cell **R50C18** and here's what we get.

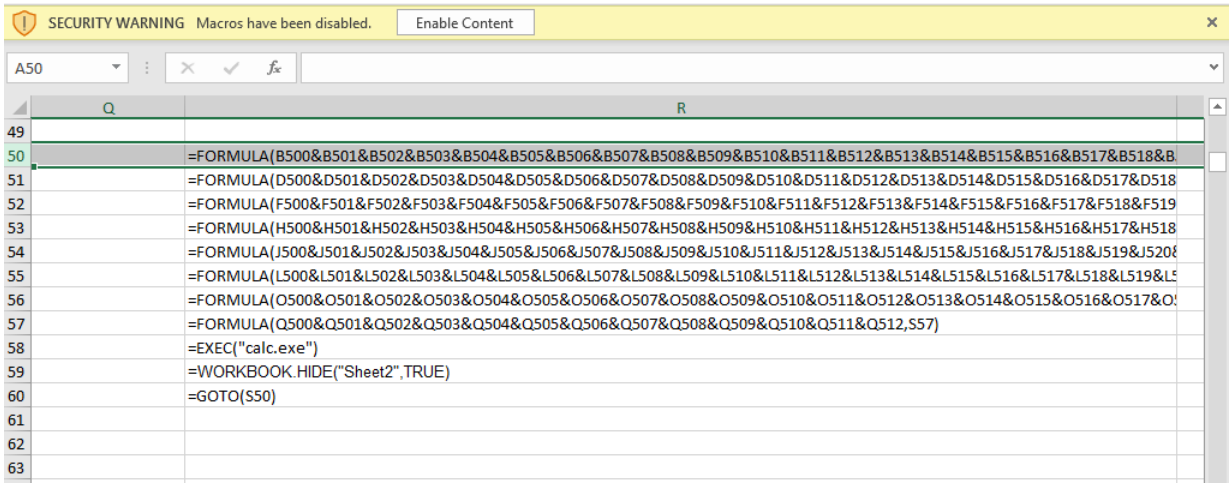

Figure 18.

If we go to B500 we get the following:

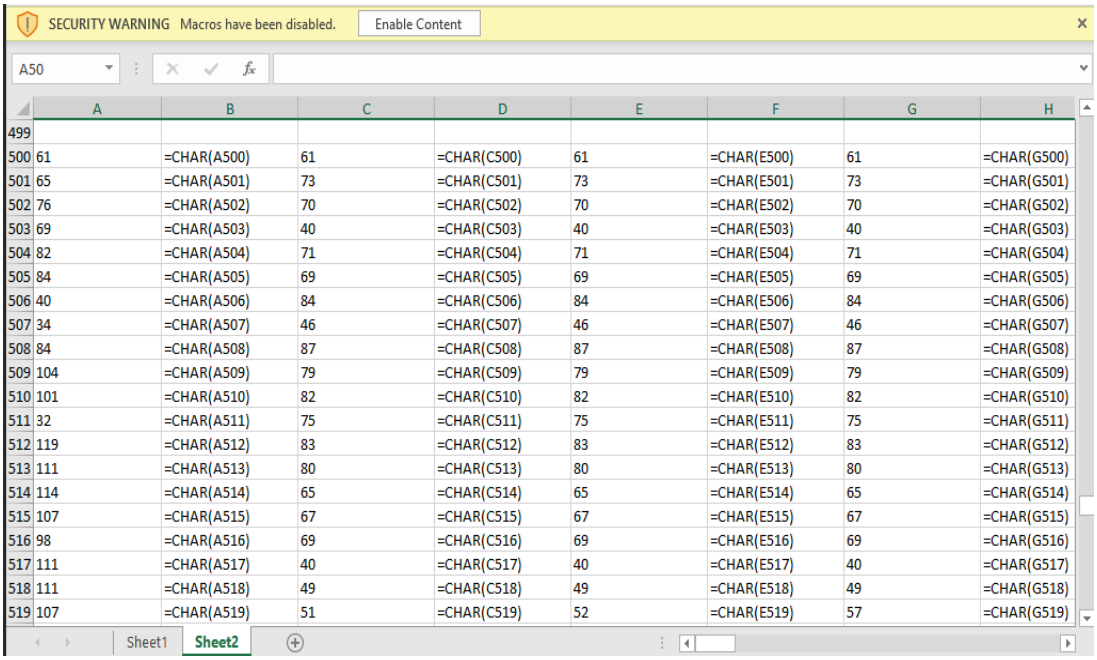

Figure 19.

In order to execute these code snippets, I created a new document and copy-pasted the cell values.

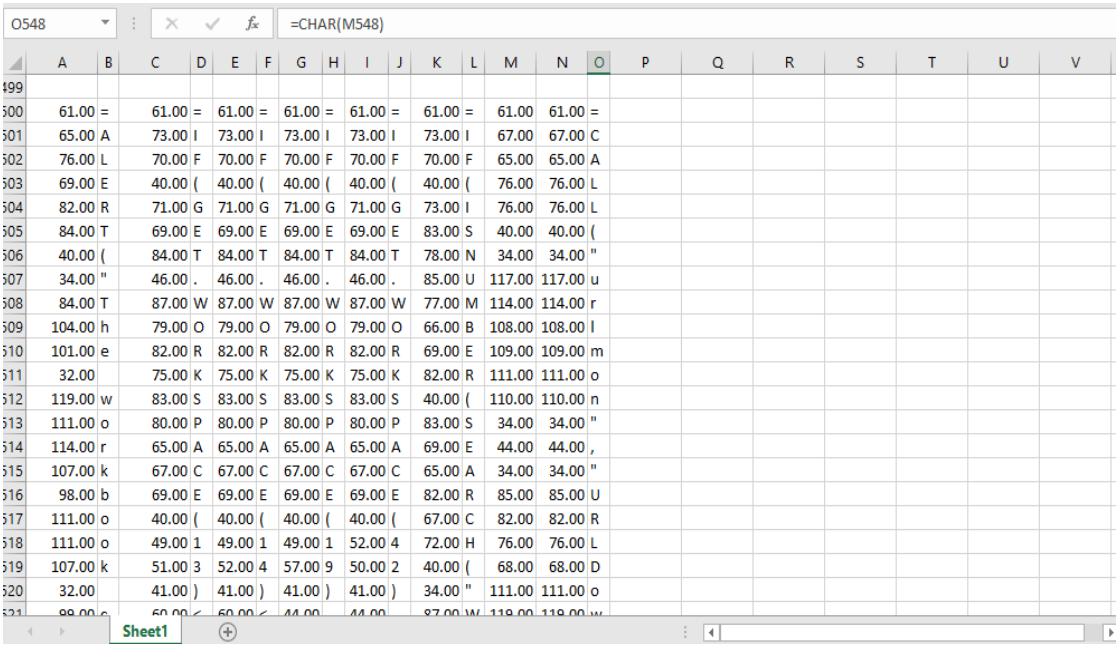

Figure 20.

Examining **column O**, **<u>it is a code fragment(vertically organized) that downloads something from rootcon.net</u>** so I extracted the URL parameter and pasted it in the browser. Originally I saved it as is (text file) but it was actually a binary file.

(Note: If this is a real malware doc, my machine could have been compromised already since I opened it in my main machine without an AV product! Anyways..)

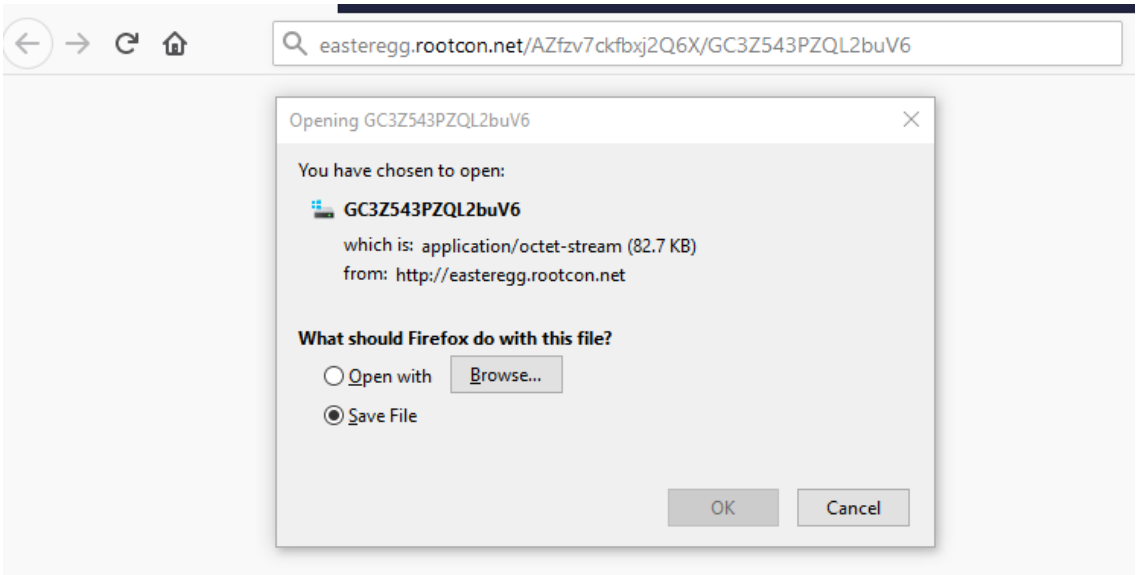

Figure 21.

I tried saving it as .png and here is what I got.

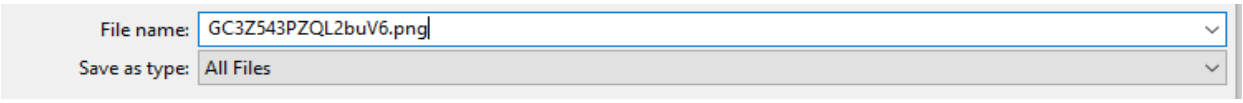

Figure 22.

Figure 23.

I noticed that the column M is used as input to the code in column O. What about column N? It seems unused. So I copied column N to column M and a different url is accessed.

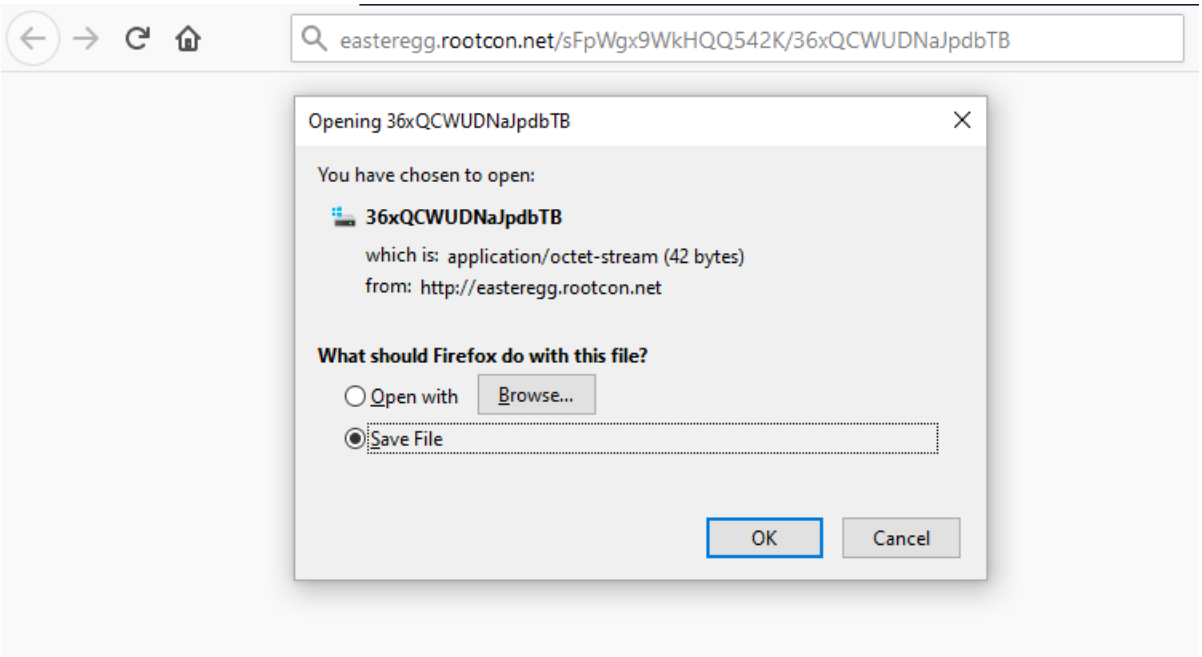

Figure 24.

And there goes the flag!

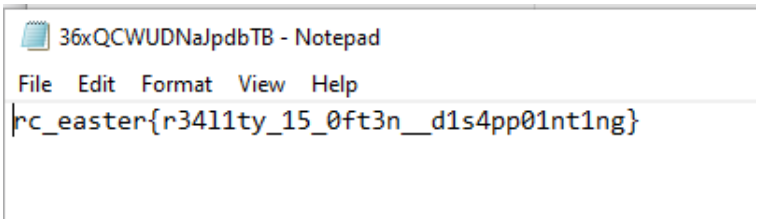

Figure 25.

References:
[1] https://github.com/decalage2/oletools
[2] https://blog.didierstevens.com/programs/oledump-py/
[3] https://inquest.net/blog/2019/01/29/Carving-Sneaky-XLM-Files