Excel 4.0 Macros [Let's Defend – Write-up]

One of the employees has received a suspicious document attached in the email. When the e-mail flow is examined, it is seen that there is a suspicious Office file. Employees forward the email to the security team for analysis.

When L1 analysts scan the suspicious file with several different scanning tools, they see that it does not contain VBA macros. Since the file format is similar to phishing, they forwarded the suspicious Office file to you for detailed analysis.

** Since the 2nd payload download addresses are closed, the 2nd payload is in the zip. Please start your analysis from the Office file.

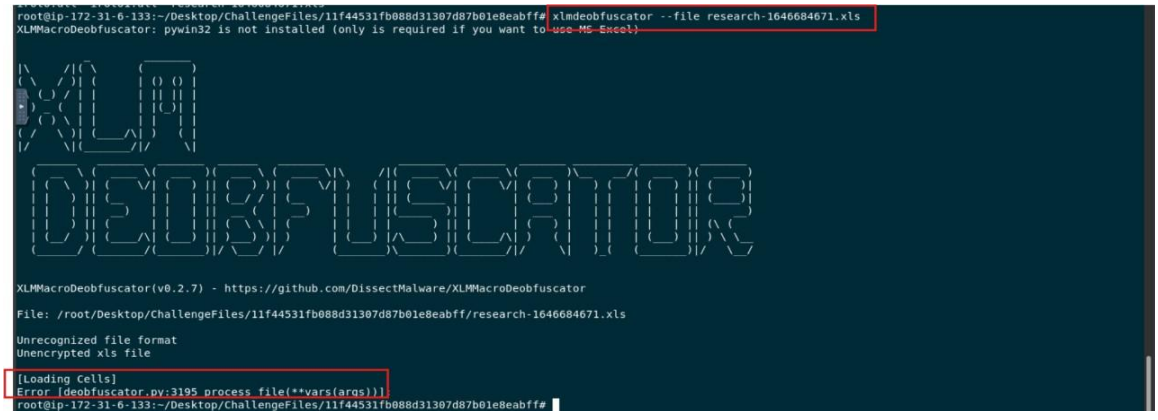
| Start investigation

It is important to note that olevba is specifically designed to scan VBA streams, meaning it will not detect XLM-based macros. These types of macros are often concealed within spreadsheets using deceptive cell names like Auto_Open or other renamed cells intended to trigger execution.

To handle these cases, you need a specialized tool called **XLMMacroDeobfuscator**.

Installation & Execution Guide
Installation: If the tool is missing, use the following command to force an install: pip install XLMMacroDeobfuscator --force

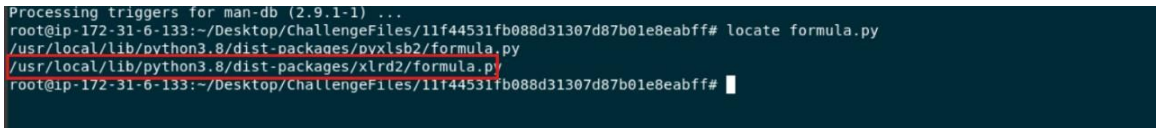
Deobfuscation: To analyze a specific Excel file (e.g., research-1646684671.xls), run: **xlmdeobfuscator --file research-1646684671.xls**



Fixing Assertion Error in deobfuscator.py

If you encounter an error at deobfuscator.py:3195 while processing a file, it is likely due to a version mismatch in the formula parser. To resolve this, you need to manually patch the formula.py file within the package.

Step 1: Locate the formula.py file



Editing formula.py using Vim

To apply the fix, follow these steps to locate and modify the function within the xlrd2 library:

1. Open the file with Root Privileges

Run the following command to open the specific file in the Vim editor:

sudo vim /usr/local/lib/python3.8/dist-packages/xlrd2/formula.py

2. Search for the dump_formula Function

Once the file is open in Vim, use the search feature to jump to the correct line:

Start Search: Press / then type dump_formula and hit Enter.
Navigate Matches: * Press n to jump to the next match.
Press **N** to go back to the previous match.

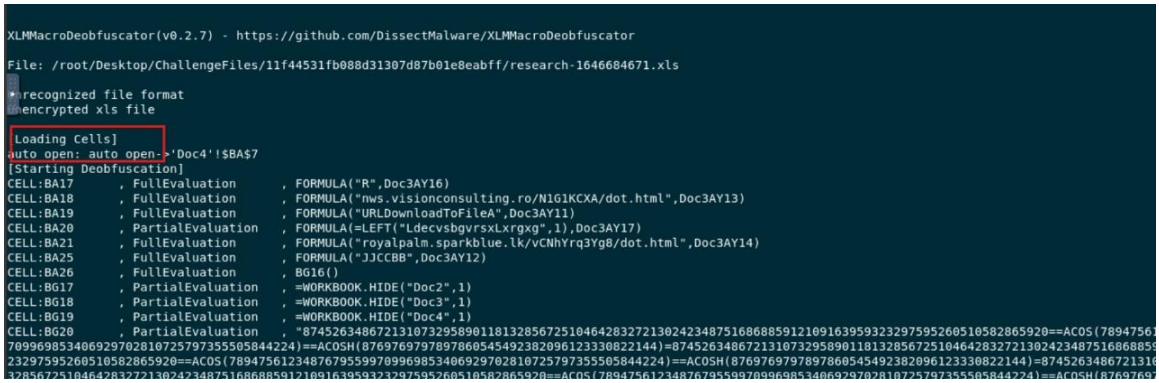
3. Modify the Line

After locating assert bv >= 80, press i to enter Insert Mode, change the value to 70, then press Esc followed by :wq to save and exit

After fixing the error i run this again;

xlmdeobfuscator — file research-1646684671.xls

Q1: Attackers use a function to make the malicious VBA macros they have prepared run when the document is opened. What do attackers change the cell name to to make Excel 4.0 macros work to provide the same functionality?

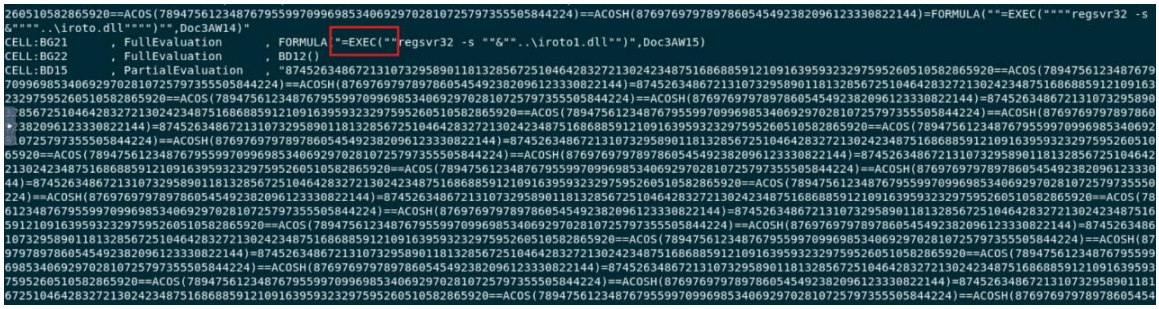


Answer: auto_open

Q2: What is the address of the first cell where Excel 4.0 macros will run in the malicious Office document you are analyzing? (Example: {doc1!ab3})

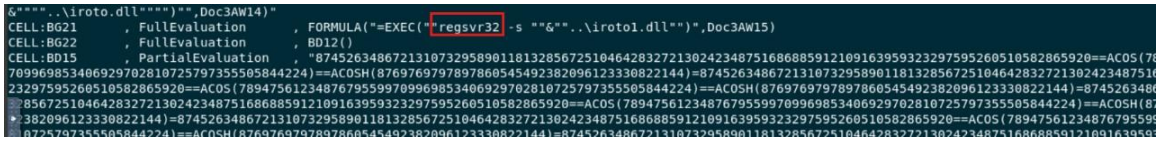
Answer: doc4!ba7

Q3: Which function is used to start a process in the operating system in the document you are analyzing?



Answer: exec

Q4: Which LOLBAS tool was used in the Excel 4.0 macros you analyzed? (Format: {xxxx.exe})

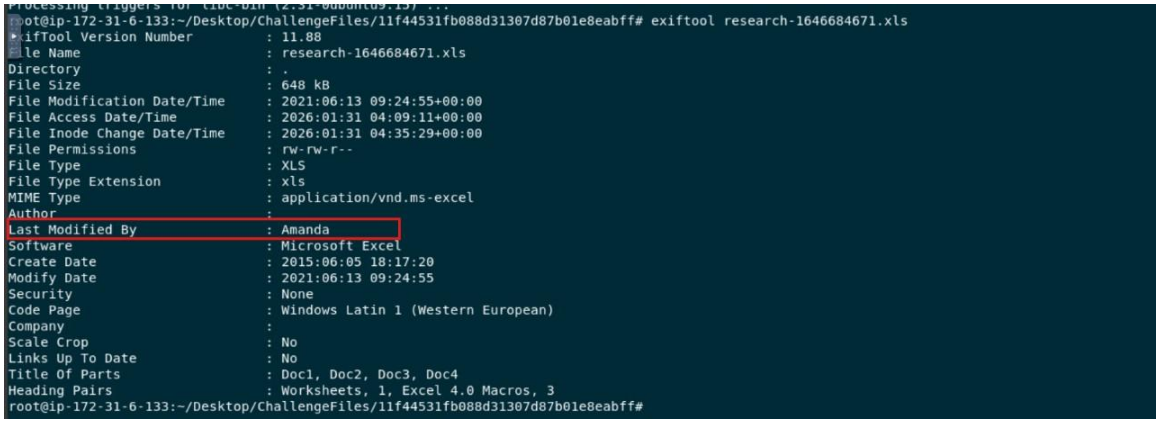


Answer: regsvr32.exe

Q5: What is the name of the registered DLL?

Answer: iroto.dll

Q6: What is the username that made the last change to the malicious document?



Answer: amanda

Summary

While there are always deeper layers to explore when analyzing macro behavior and execution flows, this walkthrough has covered the essential steps for effective detection and deobfuscation.