

# Building Anna Kournikova: An Analysis of the VBSWG Worm Kit

*Markus Schmall* 2001-08-02

## Building Anna Kournikova: An Analysis of the VBSWG Worm Kit

by *Markus Schmall*

last updated August 2, 2001

---

The Homepage and the Anna Kournikova worms are two high-profile examples of the VBS/VBSWG@mm family of visual basic script worms. These worms are generated by the VBSWG kit, one of the many virus-generating kits that are easily available on the Internet. These kits make writing a virus a simple, straightforward and unskilled task. Given the prominence of this kit, and its related worms, it would be useful for security and virus professionals to better understand it. With this in mind, this article will analyze the VBSWG kit itself (version 1.50b) and will discuss its functionality in detail. This discussion will also explain the attack points by which heuristic engines can detect all possible generations of the VBS/VBSWG@mm worms.

### How It All Began...

There is a long history of virus creation kits, although the first advanced creations kits for macro/script viruses did not appear until the end of the 1990s. The first virus generation kits for binary MSDOS viruses appeared in the early 90s. In these first years, only very basic creation kits with limited functionality were available. However, in 1998, the infamous virus programmer Vicodines introduced the first W97M/Class virus and the VMPCK kit (version 1.0a - 1.0d), which was used in many cases (VMPCK 1 & 2 families.) The VMPCK kit can be seen as one of the first advanced virus construction kits for Visual Basic for Applications (VBA.)

Later versions of the VMPCK kit were also compatible with Word97 service release 1.0. Already this kit contained functionality to add "noise" data to the created viruses, so that the analysis of these viruses became harder. Additionally the variable names were randomly generated, which made the effort to manually follow the program flow even harder.

Following the trend to release advanced virus creation kits, the CPCK kit for W97M/Class-based macro viruses was released by the same author. This kit contained a number of advanced techniques, including a polymorphic engine, which was very advanced at the time of writing.

### Coming Back to VBSWG...

The VBSWG kit can be seen as the first advanced VBS worm creation kit that is programmed in Visual Basic. It was developed by a programmer named [K]alamar, who is probably located in Argentina and is responsible for quite a few viruses and worms, most of which did not work and, from the technical standpoint, were not advanced. These creations were mostly distributed via the Virii Argentina web site, to which the programmer seems to have some relation.

Once the VBSWG kit program is started, the user works with a simple, standard Windows user interface to generate his or her own malicious code. Worms created with this kit are easily identified by the first line, which always contains the string "Vbs.Worm Created By [K]Alamar".

It should be noted that a detection of malicious code based on a human-readable comment line is not reliable. This human-readable information (comments, text strings and messages) is very often modified to create variants of known malicious codes. This type of modification does not require expert knowledge, so that even beginners are capable of doing this. Therefore the detection of malicious code should be based on important functions within the code (e.g. the replication routine) rather than on information contained within a comment line.

Generally, the generated malicious code can be detected using advanced scan string methods, checksums and heuristics. Code generated by the [K]alamar kit does not contain any state-of-the-art technology that would prevent or hinder detection. However, there are a couple of well-known methods to hide important information from heuristic engines. One possibility is to encrypt critical parts of the malicious code by selecting the "Encrypt the code" option within the VBSWG kit. When this option is selected, the program generates a simple encryption routine for the complete body of the worm and a small function call construction. The body in encrypted form will be parsed to the encryption routine as a very long string parameter and the encryption routine will return a string, which contains the decrypted body (= worm). The returned string will then be executed.

Because of this, simple heuristic engines that look only at the encrypted function parameter and the encryption routine are not able to detect any typical malicious operation. Thus malicious operations like the creation of file system activex object, suspicious registry access functions, etc., are within the encrypted parameter and are therefore not visible. What heuristic engines can detect is the execution of the result of a function whose parameter is obviously not ordinary VBS code (for example, if the relation between capital letters, small letters, spaces and numbers for a typical English text is not correct.)

## Encryption

The kit creates fairly simple encryption routines. To make the manual analysis more complicated, all variable/function names have been generated randomly.

Basically, there exists a For/Next loop depending on the length of the parameter. The encryption routine always reads two bytes, checks for special control characters and performs, if necessary, a simple arithmetic operation to decrypt the string. Finally, the resulting string will be concatenated with the existing string and the next two bytes will be read. The encryption routine itself is clearly written and leaves enough attack points for heuristics:

- the incoming string will be read (at least partly);
- there exists a loop based on the length of the incoming parameter;
- arithmetic operations will be performed on the read information; and,
- changed information will be used to create output string.

To have a look inside the encrypted block, some kind of code emulation would be needed. As mentioned previously, the generated code is suspicious enough to generate a suitable heuristic profile without the need for a code-breaking engine including a sandbox.

## Proliferation/ E-Mail Spreading Routines

There are two available e-mail spreading routines within the VBSWG kit. The first routine (called within the GUI "Outlook Attachment") utilizes Microsoft Outlook, and is comparable to the routines already found within the W97M/Melissa and VBS/Loveletter families. The routine creates an "Outlook. Application" object and parses through all available address lists. If the list is not empty, a new mail will be created and every entry in the list will be added to the recipient's list. At the end of one of each targeted address list, the previously generated mail will be sent. Additionally, the code contains functionality to delete the mail after it is sent in order to minimize signs of its existence, a characteristic has previously been seen in a couple of other worms and viruses.

The subject and body information for the created e-mails can be entered within the GUI from VBWSG. It is not possible to add a number of possible subjects, which could be selected randomly at runtime. Finally, after all the mail operations have been performed, the generated code sets a marker within the registry. If this marker is set, the main body of the malicious

code does not call the mail replication routine. Similar checks have been found within other mass mailing viruses like W97M/Melissa.

The code of the mail replication routine is programmed fairly simply and contains no obvious errors. For heuristic engines there are a couple of easy identification marks:

- creation of an "Outlook.Application" object;
- creation of a mail item;
- looping through address lists / address entry lists ;
- attaching a static file; and,
- adding numerous entries to the recipients list.

The second e-mail routine hasn't been seen that often. This function is generally comparable to the first function with the exception that an HTML mail is sent. The parsing code for address lists and address entry lists is identical. This routine additionally reads the own code line by line and stores the concatenated result in a variable. An HTML message with embedded script code will then be generated. This script contains the complete malicious body and a simple install routine, which saves the generated code in a specified directory and starts this code afterwards. The embedded code also checks whether or not the creation of activex objects is possible. If not, a warning message will be created. The same heuristic points that were mentioned for the first mail routine are also valid for this routine.

Additionally it is possible for the heuristic engine to check the HTML page for embedded script code. At this point heuristic engines could detect the following additional points:

- creation of a "scripting.filesystem" object;
- calculation of a dedicated directory; and,
- creation of a file in a directory and afterwards start of this file.

## **mIrc**

Following the trend to feature a set of replication methods, the VBSWG kit also offers the possibility of using the popular mIrc program as replication platform. The generated code is straightforward and tests for three typical locations of the mIrc program: in the program files drawer, "c:\mirc" and "c:\mirc32". If one of these drawers is found and the typical mirc.ini file exists within the drawer, the spreading routine continues. The routine itself generates a new

mIrc configuration file, which includes commands to send the malicious code to other people.

Afterwards, again, a simple marker is set, which indicates that the ini file for the IRC client was successfully generated. This marker prevents the worm from running the same code twice.

Similar functionality has been seen in various mIrc worms lately. These similarities can easily be detected by modern heuristic engines.

Beside the mIrc package, the pIrc program is also a popular IRC client and the VBSWG kit offers the possibility to create a replication routine for this platform. This routine is nearly identical to the routine written for mIrc and should be able to be detected by heuristic engines without any problem.

## Visual Basics Script

So far the VBSWG kit offers quite a lot of functionality to spread its code based on email/IRC clients. Additionally, the kit is able to generate code to spread the worm on all accessible drives by overwriting all found files with the extensions .vbs (ordinary Visual Basic Script) and .vbe (encrypted Visual Basic Script). The files will be overwritten and are not restorable. Speaking of the latter file type, the kit is not able to generate code that is encrypted based on the Microsoft VBS encryption routines. Looking at the quality of the generated code, some lines can be seen as obsolete. Besides that, it is written in a straightforward manner and leaves enough attack points for heuristics, such as:

- enumerating files in a drawer;
- enumerating subfolders in a drawer;
- scanning file extensions; and,
- copying own code over enumerated files.

## Anti-Deletion

The VBSWG kit also contains a function called "anti-deletion". Basically, this function checks within a loop to see if the file from which the worm was started (accessible via the scripting.filesystem activex object) still exists. If not, the file will be recreated based on previously-read content. This routine is written using old style "poll" techniques, but the style is again straightforward.

## Payloads

The kit also offers two "traditional" payloads, called "Crash system" and "Crash system2". The first payload tries to allocate a lot of memory within a recursive loop and performs string operation, which shall the system make run out of memory. Similar operations have been seen a few times in the macro virus field. Also the second payload has been seen a couple of times, mainly in the Javascript/VBS field. Within an infinite loop new instances of the notepad application will be started.

The next feature that the VBSWG kit supports is called "Download file" and is comparable to the file download feature found within the initial VBS/Loveletter.A variant. The generated worm first checks to see if the standard download directory for the Internet Explorer (IE) has been set. If not, it will be expected that the download directory is located at the root directory of harddrive "c:". Next it looks for the file to be downloaded in certain locations (the IE download directory and an additional directory). If the file is not found in one of the two locations, then the IE start page is set to the URL, which points to the file. As a result, at the next IE start, the file will be downloaded or directly opened (depending on the user interaction). If the file is found within the IE download directory, then it will be started within a special folder and the IE start page will be set to a blank page. This function is also programmed in a clean, straightforward style. This routine also offers heuristic engines quite good attack points, although no obvious replication routine can be found.

## Conclusion

Looking at the features available within the VBSWG kit (both versions 1.50b and 2 beta), it is obviously one of the most sophisticated virus/worm creation kits available today. No other kit generates working malicious code and offers comparable complex functionality. With that in mind, it is good to see, that most AV vendors have generic/heuristic solutions build into their products that cover all possible variants generated from these kits.

*Markus Schmall works at T-Mobil in the IT Security department and is also writing his PHD on advanced detection methods for malicious code at the VTC/Uni Hamburg. Markus Schmall can be reached at [markus@mschmall.de](mailto:markus@mschmall.de).*

[Privacy Statement](#)

Copyright 2006, SecurityFocus