Computers & Security, 16 (1997) 430-438

A History Of Computer Viruses: Three Special Viruses

Harold Joseph Highland, FICS, FACM
Editor-in-Chief Emeritus

There are several computer viruses that we have in executable form but whose existence in the business world cannot be satisfactorily confirmed. Among these are:

- the Macro virus, a spreadsheet virus that attaches itself to a data file,
- the Vienna virus, actually two versions although only one has appeared in print, and
- the Batch virus, written in legal DOS batch commands.

The Macro Virus

In the spring of 1988, following the rash of computer virus reports — The Pakistani Brain, Lehigh's COMMAND.COM, Friday the 13th, April 1st, Alameda Boot — there were persistent reports about a virus that attacked spreadsheets. At that time we were unable to confirm its existence. We decided that it was part of the mass hysteria since so many so-called virus attacks turned out to be human error and/or a bug in a program.

Late in the summer of 1988 we learned from two European computer security specialists that they too had reports about a computer virus that attacked spreadsheets. We had the impression that they knew more than they said. From the details given about how the virus worked, it was more than likely that one or both of these virus hunters had encountered a live version of the virus.

Periodically during the year we had requests from reporters and writers from the States and Europe about what we knew of this particular virus. We noted that we had no information and that we follow a policy of not acknowledging the existence of any virus unless we had a copy in our hands. Over many months we had received copies of most of the viruses rampant at the time. We were aware that any abnormality in operations was immediately considered a possible virus; a conservative approach was therefore needed.

Even when we received telephone calls from security specialists from large organizations making similar inquiries, we maintained our no-knowledge position.

0167-4048/97$17.00 © 1997 Elsevier Science Ltd

Again in speaking with some of the specialists we felt that their questions were based on more information than they were willing to admit having.

We could not envision a spreadsheet virus since it appeared obvious that any virus had to attach itself to an executable program. Obviously by attaching itself, it had to alter the program size. There was the possibility that a 'hole', a stack, in a program might be used to hide; this is what the Lehigh COMMAND.COM virus did. There appeared to be no way in which a computer virus could attach itself to data.

Early in January 1989 we received a telephone call from a systems specialist from a large multinational corporation who had experience in mainframes and microcomputers. He sought our assistance in the company's development of anti-virus protective measures. He was particularly interested in the work we were doing in virus attacks against encrypted programs and data. The request about data seemed strange.

After considerable evasive discussion, he admitted that his organization had heard about the possibility of a spreadsheet virus many months earlier and they did some research in this area. They had created a conceptual model of a spreadsheet virus. From the way he spoke we were certain that his centre had gone beyond the theoretical model. Later in our conversation, when he mentioned that they had found data modified during a test, we were sure that they had either written the virus or discovered it somewhere within their company.

With this knowledge we again contacted several of our earlier sources and were able to ask more specific questions. Undoubtedly some of those with whom we spoke felt that we too knew more than we were willing to disclose. They were therefore willing to talk more freely hoping to exchange information.

- The macro virus, although part of a data file, is **executable code**. However it undoubtedly would not be detectable with most of the conventional anti-virus products now on the market.
- Unless it is highly sophisticated, the macro virus is easily detected by any attentive and alert user.
- Methods of detection are currently available. They require some time overhead but security is not free and unobtrusive.
- Good backup policy is helpful in reducing the risk but is no guarantee. Backup procedures should be enforced.

Creating a Macro Virus

Because we were to speak at association meetings of internal auditors and EDP auditors in March/April of 1989, we wanted a simple spreadsheet virus to use for a demonstration during our talk. We therefore decided to create a laboratory version of this type of virus. We consulted with a number of specialists familiar with spreadsheet languages, especially one of the most popular of the packages, Lotus 123. Because of their sensitive positions they asked not to be identified. Yet at this time we should like to thank them for their invaluable assistance.

We maintained, more or less, a security compartmentalization policy in the writing of the laboratory virus. Some were involved only with the replication code. Others helped in preparing code for destructive action by the virus. Some worked on the 'routines to erase' the macro after it had performed its task. Others worked on a module to have the virus store itself in RAM memory and/or embed itself in other locations in the system.

> Note that a macro virus is **not limited** to Lotus 123. We used that language to prepare a demonstration virus only because of the specialists available to us. In can be written in any spreadsheet language. Similar techniques can be used with any program that permits a user to write his/her own macros. This not only includes spreadsheets but also database programs. One researcher with whom we spoke believed that such viruses might even be created in some of the text editors that permit the writing of macros.

An auto-execute macro in Lotus 123 is created in the same way as any other macro. It is given a special name: \0 [zero]. It is loaded automatically as soon as a file is retrieved. It is normally used to backup files or to assure the performance of certain operations when other people use the worksheet.

- The original macro virus developed in our laboratory was designed to alter a single value in a specific column each time the worksheet was loaded.
- Also the percentage change was restricted within a small range and that percentage did not vary.
- Later we added a clock-depending variable so that the percentage change could be either added or subtracted from the value being changed.

The virus is a 'kindergarten' type in its simplicity of action. It could easily be made more sophisticated by having it:

- execute only once on any given day,
- attack any cell in the worksheet,
- copy itself into memory or another unrelated program so that it could be activated later,
- be variable in its percentage change and/or,
- erase itself after a single execution.

A Simple Demonstration

As the macro virus is now designed any attentive and capable user can easily spot the virus at work. The original data are shown in Table 1. The values in the first two columns are identical. The values in the remaining columns were included to provide a spreadsheet atmosphere. Only the values in the second column were attacked by this version of the virus. We kept the original data in the first column so that the change would be easily seen.

Running the program once a day caused data corruption. In Table 2 the values in row 1 were changed when the worksheet was called for the first time. The 4,550.00 in the original values column was altered to 4,441.76. There was a decrease of slightly more than 2 percent. A special section within the virus determined the percentage change; another module determined if the

Table 1

	Original Values	Duplicate Values	Other Data	More Data
1	4,550.00	4,550.00	1,793.50	250.00
2	4,500.00	4,500.00	2,613.00	300.00
3	5,120.00	5,120.00	3,750.00	500.00
4	4,600.00	4,600.00	2,850.50	180.00
5	1,100.00	1,100.00	950.00	100.00
6	4,650.00	4,650.00	3,850.00	550.00
7	4,450.00	4,450.00	3,900.00	550.00
8	7,500.00	7,500.00	6,500.00	750.00
9	3,750.00	3,750.00	2,700.00	210.00
10	40,220.00	40,220.00	28,907.00	3,390.00

Table 2

	Original Values	Virused" Values	Other Data	More Data
1	**4,550.00**	**4,441.76**	1,793.50	250.00
2	4,500.00	4,500.00	2,613.00	300.00
3	5,120.00	5,120.00	3,750.00	500.00
4	4,600.00	4,600.00	2,850.50	180.00
5	1,100.00	1,100.00	950.00	100.00
6	4,650.00	4,650.00	3,850.00	550.00
7	4,450.00	4,450.00	3,900.00	550.00
8	7,500.00	7,500.00	6,500.00	750.00
9	3,750.00	3,750.00	2,700.00	210.00
10	40,220.00	40,111.76	28,907.00	3,390.00

Table 3

	Original Values	"Virused" Values	Other Data	More Data
1	4,550.00	4,441.76	1,793.50	250.00
2	**4,500.00**	**4,615.79**	2,613.00	300.00
3	5,120.00	5,120.00	3,750.00	500.00
4	4,600.00	4,600.00	2,850.50	180.00
5	1,100.00	1,100.00	950.00	100.00
6	4,650.00	4,650.00	3,850.00	550.00
7	4,450.00	4,450.00	3,900.00	550.00
8	7,500.00	7,500.00	6,500.00	750.00
9	3,750.00	3,750.00	2,700.00	210.00
10	40,220.00	40,227.55	28,907.00	3,390.00

Table 4

	Original Values	"Virused" Values	Other Data	More Data
1	4,550.00	4,441.76	1,793.50	250.00
2	4,500.00	4,615.79	2,613.00	300.00
3	**5,120.00**	**5,196.60**	3,750.00	500.00
4	4,600.00	4,600.00	2,850.50	180.00
5	1,100.00	1,100.00	950.00	100.00
6	4,650.00	4,650.00	3,850.00	550.00
7	4,450.00	4,450.00	3,900.00	550.00
8	7,500.00	7,500.00	6,500.00	750.00
9	3,750.00	3,750.00	2,700.00	210.00
10	40,220.00	40,304.15	28,907.00	3,390.00

Table 5

	Original Values	"Virused" Values	Other Data	More Data
1	4,550.00	4,441.76	1,793.50	250.00
2	4,500.00	4,615.79	2,613.00	300.00
3	5,120.00	5,196.60	3,750.00	500.00
4	4,600.00	4,586.51	2,850.50	180.00
5	1,100.00	1,100.00	950.00	100.00
6	**4,650.00**	**4,882.74**	3,850.00	550.00
7	4,450.00	4,450.00	3,900.00	550.00
8	7,500.00	7,500.00	6,500.00	750.00
9	3,750.00	3,750.00	2,700.00	210.00
10	40,220.00	40,723.39	28,907.00	3,390.00

Table 6

	Original Values	"Virused" Values	Other Data	More Data
1	4,550.00	4,441.76	1,793.50	250.00
2	4,500.00	4,615.79	2,613.00	300.00
3	5,120.00	5,196.60	3,750.00	500.00
4	4,600.00	4,586.51	2,850.50	180.00
5	1,100.00	1,100.00	950.00	100.00
6	4,650.00	4,882.74	3,850.00	550.00
7	**4,450.00**	**4,698.14**	3,900.00	550.00
8	7,500.00	7,500.00	6,500.00	750.00
9	3,750.00	3,750.00	2,700.00	210.00
10	40,220.00	40,771.53	28,907.00	3,390.00

change would be negative or positive. This latter module was time dependent. Calling the worksheet before 9:05 AM or after 5:00 PM would result in a negative change.

When the program was run the second day, the original value of 4,500.00 in line 2 was modified to 4,615.79, a change of about 2.5 percent as shown in Table 3. The original total of the values changed from 40,220.00 to 40,227.55, a change of about 0.01 per-

cent. The increase in the second value was offset by the decrease in the changed first value. Yet the data in two cells were corrupted.

Table 4 shows the change that took place on the third day. The original value of 5,120.00 in line 3 was altered by the virus to 5,196.60. Of course, the total was likewise modified by the spreadsheet program.

By the end of the first week, the virus had made five changes in the data. Note that the value in line 5 of 1,100 had not been altered because the virus was designed to attack values within a specific range. The total of the column had been increased by 503.39 during that period.

When the worksheet was called the next week the original value in line 7 of 4,450.00 was altered to 4,698.14 as shown in Table 6. Had we continued using the worksheet, the test virus would not have altered any other values in the column because they are out of range. The virus code could have been written so that once it found no more data to change in the column, it would have erased itself.

A more universal virus could be written to attack any cell and modify it by a varying percentage. That modification could be negative or positive set not by time but by odd or even dates, day of the week or any other algorithm.

How to Detect a Macro Virus

Many individuals and organizations exchange templates, especially the complicated ones. Some even download such templates from bulletin boards. Again it is the case of a trusted source. However in these days of possible viruses we are unwilling to accept any disk, program, utility or programming aid even when purchased from a reputable vendor in its original sealed package.

Because of this virus threat it will be necessary to evaluate existing templates and particularly any new ones that might be obtained. Each worksheet will have to be examined for the macros it contains. The Advanced Edition of The Norton Utilities offers a fast way to find the range labels and macros.

It is best to use a test microcomputer, a two floppy disk machine. If it is not available, a hard disk computer can be used with precautions:

- First, a full backup of the system should be available.
- Second, if two floppy disk drives are available on the machine, boot the system with a write-protected disk in drive A with a CONFIG.SYS file that does not include drive C, the hard disk.
- Third, if a second floppy drive is not available, it is essential to write-protect the hard disk.
- Fourth, after the template has been examined, the system should be immediately shut down. Do not reboot the system for about five minutes.

Here is the quick and simple test method using the Norton Utilities:

[1] Make a copy of the template or worksheet to a non-bootable floppy disk and place that disk in drive B if using a two floppy disk system.

[2] Place a write-protected copy of The Norton Utilities in drive A or with a hard disk, make certain that The Norton Utilities are in the path of the hard disk, C.

[3] With the system at the A or C drive, enter: NU B: at the system prompt.

[4] Step through the menus — "Explore disk," "Choose item," "File."

[5] The template should be the only file shown in the list on the screen; press the Enter key.

[6] At the next menu, select "Edit/Display item." "A View of a Template" in the accompanying box contains part of the screen showing the range labels and macros used with the worksheet.

A View of a Template

```
Cluster 2, Sectors 12-13                                File offset 0, hex 0
000D0800 03000300 0D080003 0004000D 64002000 00000000 . . . . . .d. .....
00000000 00000000 00000000 00000000 00000000 00000000 ........................
00000000 47001900 434F554E 54000000 00000000 00000000 ....G. .COUNT..........
10000500 10000500 00470019 00454E44 00000000 00000000  . . . ..G. .END........
00000000 00060017 00060017 00004700 19004E45 58540000 ..... . . . ..G. .NEXT..
00000000 00000000 00000E00 11000E00 11000047 00190054 ..... . . . ..G. .T
45535400 00000000 00000000 0000000F 0009000F 00090000 EST..... . .    ..
47001900 5C300000 00000000 00000000 00000000 0E001000 G. .\0..... . .
0E001000 00470019 005C5600 00000000 00000000 00000000  . ..G. .\V.....
000E0010 000E0010 00001800 190000FF FF0000FF FF0000FF . . . . .. . ..........
```

You now have the list of macros with the template to analyze. Still using a copy of the worksheet on the disk, load Lotus 123 and retrieve the file. At this point:

[1] Examine the worksheet to ascertain if there are any hidden columns, rows or cells. If there are any, unhide them for this examination of the template; you are using a copy.

[2] Find and evaluate each macro that you have on the list you obtained by using The Norton Utilities.

[3] Take a printout of the entire template. It will include all of the macros. Taking a printout of the worksheet with hidden cells, columns and/or rows is not sufficient since the hidden portions will not be printed.

[4] Since this copy of the template will not be used again on your system, it is not necessary to hide the cells, columns and/or rows.

[5] At this stage shut off the microcomputer. With many systems it is necessary to wait about five minutes to be certain that the system is safe to use again.

It should be obvious that the procedure is not complex, but it is also obvious that evaluating a macro is not a job for an amateur or a data entry clerk. It is necessary to assign a capable individual to do this evaluation.

Once a worksheet has been accepted there is no guarantee that some employee might not take off on his/her own to modify a macro. Therefore, the periodic use of The Norton Utilities to view of the range labels and macros would enable any auditor and/or security specialist to determine whether or not anyone has tampered with the worksheet. How often the review should be made depends on backup procedures and practices used by an organization as well as the critical nature of the data stored in the template.

Other Macro Viruses

When we had a working macro virus we spoke with several researchers and security specialists to alert them about this type of virus. A security specialist, experienced in Dbase, called with a few days to report that he had prepared a simple virus that altered database files. He also had another version that exited from the program and executed a DOS del *.* for any floppy disk on the system.

A virus researcher with whom we spoke earlier in the year called in April 1989. He informed us about his development of a virus using the macro instructions available in a popular text editor. His macro for print instructions included perfectly legal code and erased all text files on the disk.

The Vienna Virus

The Vienna Virus was reportedly found in Europe late in 1987 or early in 1988. We have not been able to confirm its existence nor find anyone who would admit seeing it in the real world.

Some time very early in 1988 an associate from Europe sent us a copy of a disk that contained the purported Vienna virus. He reported that the code came from someone associated with the Chaos Computer Club in Hamburg, Federal Republic of Germany. Because the code "did not work properly" an associate of his modified the program to get it to run. Because at that time we had no reports of the virus attacking any organization in Europe and had our hands full with the Pakastani/Brain, Lehigh and a set of Israeli viruses, we set the disk aside for detailed analysis at a future date.

About one month later another European friend mailed us a copy of the original German version of Ralf Burger's "Computer Viruses: A High-Tech Disease" that had been printed by Data Becker, GmbH of Dusseldorf. In that volume there was a write-up of the Vienna virus along with a pseudo-flowchart in assembly code. Burger noted that the code came from someone in Austria [definitely not our friend]. The Burger Vienna virus supposedly resets the seconds field of a file's time entries to 62 after it has infected the program. Attacking only .COM files in a defined path, it destroys the first five bytes of the infected program. Further Burger added that both he and Bernard Fix independently analyzed the code and that Mr. Fix prepared the flowchart that appears in the book.

Because we had not written in assembly language for several years, we did not attempt to compile the Burger program. However from the descriptive text in the book about the virus, we were aware that the book version was not identical to the one we received earlier on a disk.

Later in 1988 after the Burger book had been translated into English and published by Abacus of Grand Rapids, MI, we learned from Bill Kenny that he had assembled the Vienna virus from the coding in the book. He complained that Burger had been sloppy and/or careless in his coding and that the program would not work as published. He too found it necessary to make slight changes in the code.

The Kenny Version

The Kenny version of the Vienna virus according to Burger attaches itself in the first three bytes of a target program, placing a JMP [jump] instruction there to the virus which is placed at the end of the original program. During the infection the virus checks the current time:

- if the seconds are a multiple of eight, it then writes five 'garbage' bytes to the start of the file. The virus destroys the file instead of infecting it.
- if the seconds are not a multiple of eight, the first three bytes of the infected program are restored and the program is executed.
- a technical explanation of how this version works is included below.

How the Vienna Virus Operates

The Vienna virus is a .COM file infector, which attaches itself in the first three bytes of the target program by placing a JMP instruction there. The remainder of the virus goes at the end of the original program.

The virus recognition signature is the seconds in the time field of the directory entry. Seconds are stored as seconds/2, with legal values of 0 to 1DH [0 to 29 corresponding to 0 to 58 seconds]. The virus sets a value of 1FH, 31, corresponding to 62 seconds in the time field.

Target programs are searched for at the start of execution of the infected program. The search consists of looking through the current directory for an uninfected .COM program. If that fails, all directories in the PATH are searched for an uninfected .COM file. If this too fails, the virus gives up trying to infect another program.

During an infection, a check is made on the current time:

- If the seconds is a multiple of 8 [i.e. 0, 8, 16, 24, 32,40, 48, 56], then five 'garbage' bytes are written to the start of the file. This action effectively destroys the file instead of infecting it.

- If the seconds is not a multiple of 8, the first three bytes of the infected program are restored, and the program is executed.

Known Bugs

- The virus assumes that a PATH can be found in the environment. If no PATH exists, it can do a lot of searching before it finds 'random' data that matches its criteria for the end-of-path marker.

- When entering the infected program, AX is set to 0 rather than to the drive validity information passed by DOS. This will affect programs like FORMAT.COM, which use it. - **Bill Kenny**

Two Versions Compared

Both versions of the Vienna virus are .COM file infectors. Both place a 'jump' instruction in the first three bytes of a target file. In both the remainder of the virus code is attached at the end of that file. Both versions of the virus attack programs in the current directory as well as any of those listed in a defined PATH.

[1] In the version that closely follows that in this article, the virus makes a check of the current time when a program is called:

- If the seconds are a multiple of 8, five 'garbage' bytes are written to the start of the .COM file, effectively destroying it.

- If the seconds are not a multiple of 8, the first three bytes of an infected program are replaced and the program executed.

[2] In the earlier disk version we received from Europe, the virus attack depends on the time-and-date stamp of the program that is called:

- If the year modulus 4 equals zero and the month is less than 8, the virus trashes the pro gram by filling the first five bytes with garbage.

- If the year and month tests do not meet the trashing requirement, the infected program is executed after the first three bytes are replaced.

Both versions are relatively the same size. One used a time modulus to set its trigger; the other the time-and-date stamp of the program. When we telephoned Europe to speak with our friend, who cannot be acknowledged, he told us that he too had two versions now. Which one was the Burger 'original' he did not know. He had not saved the original copy of the 'source code' and his associate no longer worked for the organization. He believed that the original code came from a photocopy of a book or possibly from *Datenscheuder*, the Hamburg Computer Chaos Club newsletter.

The Batch Virus

It is possible to prepare a virus in legal batch language that is executable under DOS. This Batch virus can be incorporated as part of an AUTOEXEC.BAT file so that it is executed when the floppy disk is used to boot the system. One of the versions we have copies itself to other .BAT files on the disk. When any one of these is invoked, the virus is triggered. Another version of this virus copies itself to the hard disk's AUTOEXEC.BAT file. Later when the hard disk is booted the virus is triggered.

The replication portion of the virus is the more complex portion. To write it one must be well versed in the batch control language. The version we use in our lectures contains only 300 bytes and is 11 lines long. Three of the lines include "echo off" and two "routine labels." Thus the actual executable code contains eight lines of code or only 271 bytes.

Some 10 bytes of this code consists of the action code as a single line. When demonstrating this program we confine its action to a floppy disk. The action code, del *.com, deletes all the .COM programs on the floppy disk. It would be simple using either DOS's text editor, EDLIN, or any other text editor to alter this line to read "del c:*.COM" or "del c:*.EXE"so that everything on the hard disk is erased. Entering "del c:*.*" would alert the user with the "are you sure?" display on the screen.

The potential danger of this virus can be demonstrated easily. Normally when one receives a new disk he/she obtains a directory of the disk by using the DOS DIR command. Consider the case when the following directory is shown, (see below).

In most cases the individual would use the README.BAT file to read the lengthy manual by entering README followed by a carriage return. This would release the batch virus immediately. The virus could also have been included within an INSTALL.BAT file.

COMMAND	COM	README	BAT	MANUAL	1	MANUAL	2	MANUAL	3		
MANUAL	4	MANUAL	5	MANUAL	6	MANUAL	7	MANUAL	8		
VARAO	EXE	VAROA	OVL	VAARA	EXE	VAARA	OVL	ELOTON	COM		