# VIRUS ANALYSIS 1

## Junkie Memorial?

*Péter Ször*
*Data Fellows*

Clinton Haines did not have time to change for the better. He was an active virus writer and he died a junkie. The VLAD virus writer group have dedicated Memorial.12314 to him.

There has been a healthy development in *Windows* virus writing thus far, but the list of infectors is still not long. Memorial infects DOS COM and EXE files, as well as Win32 PE (Portable Executable) files. The main format of the virus is a *Windows 95* VxD (Virtual Device Driver). It takes an interesting new direction in loading its memory-resident component.

The virus is unencrypted in DOS COM and EXE files, but infected PE files are encrypted with an oligomorphic routine. Moreover, the main virus body (the VxD image) is packed with a simple algorithm. There is already one virus which is encrypted in PE files ( Win95.Mad), but Memorial is certainly the first *Windows 95* virus with oligomorphic properties – thus, we see the first steps towards *Windows 95* polymorphism.

Memorial is also an effective retro-virus. It manipulates the registry to disable several anti-virus programs. Fortunately, the virus has a few serious bugs which can slow its spread. Despite this, Memorial.12413 has been reported to be in the wild in Sweden and Norway.

### Running an Infected COM File

*Windows 95* virus writers appreciate that people are still exchanging more DOS programs than *Windows 95* ones. This is a big problem for viruses; they simply cannot spread very far by infecting only *Windows 95* programs. Memorial addresses this by also targeting DOS executables, which then function as droppers of the main VxD module.

If *Windows* is running when an infected COM file is executed, the virus simply executes the host program. If *Windows* is not running, Memorial makes its 'Are you there?' call – Int 2Fh, AX=0h. If AX=4AB3h is returned, the virus assumes it is already active in memory and passes control to the host. Otherwise, it creates C:\CLINT.VXD with the hidden attribute, and starts to write into it.

Initially, the writing routine looked to me like an anti-heuristic function. That is not its aim, however. The main body of the virus is packed to 7508 bytes and this function is supposed to unpack it. The algorithm is very simple, but effective. VxD files are in LE (Linear Executable) format, and their structure contains many zeros. The full VxD is packed to 7508 bytes, and grows to 12413 bytes after unpacking. When CLINT.VXD is ready, Memorial copies a piece of code from its body into the Interrupt Vector Table at 0:200 and hooks Int 2Fh (Multiplex Interrupt). So, while the virus is not troubled with memory allocation, it is incompatible with some applications.

Aside from answering the 'Are you there?' call, the Int 2Fh handler waits for AX=1605h. This notifies DOS device drivers and TSRs that standard- or 386 enhanced-mode *Windows* is starting.

When Memorial receives the *Windows* initialization notification, it tries to open C:\CLINT.VXD to check for its existence. If this succeeds, it initializes the appropriate data structures to direct *Windows* to load C:\CLINT.VXD. Thus, the virus uses a documented way to ensure its resident part is loaded by *Windows*. This is more elegant than modifying the SYSTEM.INI file, and more successful.

### Running an Infected PE EXE

On executing an infected PE file, the virus decrypts itself. C:\CLINT.VXD is dropped here, too. In the case of DOS infections, the dropper function takes 275 bytes of additional code at the virus entry point. In PE files the dropper is in 32-bit code and is more complex, so this code is longer – 1360 bytes. This function includes Memorial's activation routine and is also supposed to load the VxD.

Memorial works out the entry point, in memory, of the GetModuleHandleA function. It does this with a real hack: searching internal *Windows 95* structures. This makes subsequent PE infection easier. The virus need not add any entries to the Imported Names table, thus removing the need for a complicated patch function.

Next, it calculates the entry point of GetProcAddress with the same trick, before using GetModuleHandleA to get the handle of the KERNEL32 module. By manipulating this handle, Memorial is able to call GetProcAddress to find and save addresses for the CreateFileA, WriteFile, ReadFile, SetFilePointer, CloseHandle, GetLocalTime and LocalAlloc procedures. After this, Memorial is provided with USER32's handle by the same manipulation of GetModuleHandleA, and saves the address of the MessageBox procedure through further use of GetProcAddress.

The virus then calls GetLocalTime to check the date – if it is 10 April the virus activates and displays a message box. Otherwise, it checks whether \\.\CLINT is running. If so, the host program is executed. If not, \\.\C:\CLINT.VXD is created as a normal file using CreateFileA, and if successful, Memorial allocates memory for unpacking the virus body by calling the LocalAlloc procedure. It unpacks the VxD code to this buffer, then writes the resulting 12413 bytes into CLINT.VXD with the WriteFile function, before

closing the file with CloseHandle. The CreateFileA procedure then executes the VxD, and Memorial finally starts the host program.

### Running an Infected DOS EXE

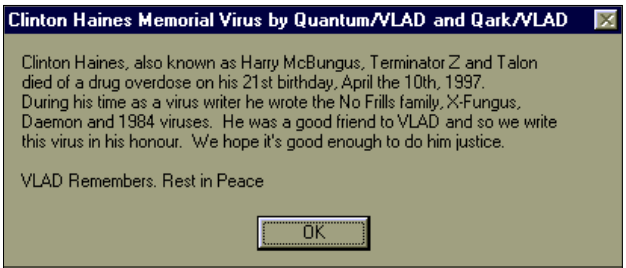
When an infected EXE file is run, the same unpacking code as in COM files takes control from the virus entry point. However, because of a major bug in the DOS EXE infection code, this function writes endlessly to C:\CLINT.VXD. Memorial uses an invalid pointer and a bad virus size parameter during EXE infection. Instead of writing the packed VxD code, it writes the unpacked copy. The dropper code 'extracts' an already unpacked VxD image. The extractor code writes megabytes to C:\CLINT.VXD, until it fills all available hard drive space, but even then, control does not return to DOS. The huge C:\CLINT.VXD can be located by pressing Ctrl-C during this operation or restarting the machine. Thus, Memorial can be classified as 'intended' in DOS EXE files. Fortunately, it is detectable and disinfectible in these cases.

### VxD Initialization (IFS API hook)

CLINT.VXD's message handler waits for four control messages. In response to W32_DEVICEIOCONTROL it returns 00h, as it does not want to communicate with other applications. When a SYS_DYNAMIC_DEVICE_EXIT message appears, Memorial returns 01h to disallow the unload request. In the case of INIT_COMPLETE and SYS_DYNAMIC_DEVICE_INIT messages, the virus executes its initialization procedure which hooks the DOS IFS (Installable File System) API.

Memorial disables many *Windows 95* anti-virus programs by deleting or changing registry settings. Several keys that start the resident or on-access anti-virus programs are deleted under '\System\CurrentControlSet\Services\Vxd'. It also removes similar keys from '\SOFTWARE\Microsoft\Windows\CurrentVersion\Run'. Several keys are also deleted from '\SOFTWARE\McAfee\ScreenScan' and '\SOFTWARE\Cybec\VET Antivirus for Win32\'. *Cybec's VET* is further targeted by setting its 'Scanning\Extension List' value to 'bin, dll, doc, drv, ovl, sys, dot', removing 'com', 'exe' and any user customizations.

The virus then clears the attributes on C:\CLINT.VXD. Subsequently, it opens the file, checks and saves its size, and allocates enough memory for both its unpacked and packed copies. Then it reads itself from the file to the main buffer, before closing the VxD file and deleting it. After this, Memorial packs the VxD code into the second buffer, and returns from its initialization routine.




Memorial's 'tribute' to Clinton Haines

### COM Infection

When Memorial intercepts a file open call, it checks the extension, comparing it first to 'COM'. If there is a match, the virus finds and saves the attributes of the victim file and clears them, allowing infection of read-only, system and/or hidden files. After opening the victim, it reads the first four bytes from it. If offset 03h is 'Z', Memorial assumes the file is already infected. If not, it checks for 'MZ' and 'ZM' markers. If the victim is of COM structure, the size is also checked – it only infects COMs between 7168 and 51200 bytes long. Next, it reads the last five bytes of the program and checks whether they start with 'SN'. If so, the file is not infected. I can see no reason for this other than the inoculation of a certain PC – the virus writer's own?

The 275-byte DOS dropper code is appended to the victim together with the 7508-byte packed virus image. Finally, Memorial changes the first four bytes to a jump to the virus body plus a 'Z' marker and resets the attributes. Thus, in the case of COM infection, the virus is 7783 bytes.

### DOS EXE Infection

If a target starts with 'MZ' or 'ZM', the virus checks the extension against '.EXE' and '.SCR'. On a match, it calls IFSMgr_Get_DOSTime to seed its random number generator. The virus must add a section name to the header of its PE targets, but as it is oligomorphic, it does not want this to be a constant name. Thus it mutates 'CLINTON', using an 8-bit XOR, to a 'garbage' string and calculates a check byte, saving it as the last character of the string. This byte is used as a self-recognition check. The virus mutates the section name during DOS EXE infection only. This makes the mutation rather slow (slow oligomorphism).

Memorial then reads four bytes from offset 3Ch – a pointer to the *Windows* executable header area. If this pointer is 0, the virus assumes that the file is a normal DOS EXE file. If the checksum field of the EXE header is 6666h, or if the IP field is 100h, the virus will not infect.

It increases the size of EXE files to the next paragraph boundary and adds the VxD dropper code (275 bytes) to the end of the file. Then comes the virus' biggest bug: it writes the unpacked VxD to the end of the file (12413 bytes). Thus, the virus size is 12688 bytes where the victim is a DOS EXE file. Finally, it modifies the executable's header to point to the virus entry point.

### PE Infection

If the double word at the 3Ch offset is not zero, Memorial checks for the PE signature, before studying the file's read-only attribute. Then it reads the last section name from the

Image Header structure and calculates the check byte on the section name. If the checksum matches, Memorial does not infect. Otherwise, it modifies the PE header by adding the new virus section to it. Since Memorial is oligomorphic, it first mutates its 46-byte long decryptor (see below) for the beginning of its body. Then it encrypts the PE version of the VxD dropper (1360 bytes), adding the decryptor then the dropper code to the end of the victim (1406 bytes). Finally, it encrypts the packed VxD image and adds it to the end of the file (7508 bytes), prior to restoring the victim's file attributes and closing it.

### Oligomorphic Engine

This engine is simple but effective – the basic decryptor consists of eleven 'blocks' of interchangeable code. There are a total of 96 possible combinations, complicating the detection of the virus in PE files.

### Conclusion

Memorial is a complex virus. Its discovery suggests that polymorphic *Windows 95* viruses will appear in the near future. One wonders whether polymorphic mutation engines for *Windows 95* viruses will be far behind. It is time to implement new scanning engines for these strange beasts.

### Memorial

Aliases: Win95.Memorial.12413

Type: Windows 95 PE, DOS COM, EXE infector. Uses VxD to 'stay resident'.

Self-recognition in Files: Offset 03h of COM files is 'Z', DOS EXE files have 6666h marker in the EXE header checksum field. In PE files, tests a checksum byte on the last entry in the section name table (see text).

Self-recognition in Memory: Cannot infect system memory twice under *Windows 95*. Int 2Fh, AX=0 returns AX=4AB3h under DOS.

Hex Pattern in COM, EXE, VxD Files and in Memory:

```
0BC0 7504 B8B3 4ACF 3D05 1674
05EA ???? ???? 9C2E FFIE 2D00
```

Hex Pattern in PE Files: Not possible.

Intercepts: Int 2Fh, AX=1605h (Windows Initialization Notification), ISF API hook OpenFile (IFSFN_OPEN).

Payload: Displays a message box on 10 April.

Removal: Under clean system conditions, recover infected files from a backup or replace from original sources.

# VIRUS ANALYSIS 2

## Search for a Heart of Stone

*Snorre Fagerland*
*University of Bergen, Norway*

StoneHeart.1490 contains some unusual tricks aswellas being quite picky about which files it will infect. It bears a strong similarity to the Nostardamus family, and could even be said to be a Nostardamus variant. It appears to have been written by the same author – Eternal Maverick of the Russian virus-authoring group 'Stealth Group World Wide' (where do they get these names from?).

### Entry

On entry, the virus code first checks that the active PSP points to its own data segment. If not, it starts overwriting the current data or code segment with its own segment address in a loop, finally corrupting running code to a miserable end. This piece of code will probably never be run – it is an anti-heuristic trick, designed to foil scanners that rely heavily on heuristics, in particular *Dr Web*. Next it calls the DOS GetDateTime function (Int 21h, AH=2Ah), with BX=454Dh ('EM') as an installation check. If already resident, this call will return BX=4D45h ('ME').

### Memory-residency – Part One

The virus makes itself memory-resident by shrinking the current memory block by 5E0h bytes. It does not create any new MCBs – this can cause memory blocks to drop out of the MCB chain and DOS to react badly (does 'Memory allocation error – system halted' ring a bell?). However, most of the time it will load as the last program in the MCB chain and set up a segment at the top of DOS memory. It then copies 1490 bytes into the new host segment.

At this stage StoneHeart performs a destructive action; it seeks out and truncates files matching C:\??????C?.???, D:\??????D?.??? and so on, going down the whole drive chain. This is aimed at *ADINF* checksum tables, but could affect other files. While this happens the DOS critical error handler (Int 24h) is disabled. Apart from being truncated, these files also have the system attribute set.

### Night of the Undead

After its truncation spree, StoneHeart does something unusual, which I have only seen in the Nostardamus family. It goes 'zombie'. It does the digital version of dying, to be dug up later. This is an anti-monitoring trick, for most activity monitors keep very close track of memory and interrupt status during the execution of a program. If something happens after the program has terminated, the monitors often miss it.