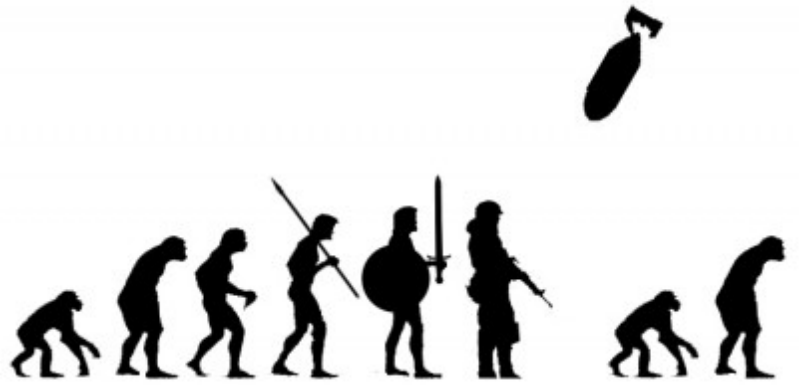
Related Posts

malwaretech.com/2014/05/rovnix-new-evolution.html

MalwareTech May 6, 2014

Rovnix is an advanced VBR (Volume Boot Record) rootkit best known for being the bootkit component of Carberp. The kit operates in kernel mode, uses a custom TCP/IP stack to bypass firewalls, and stores components on a virtual filesystem outside of the partition. Yesterday Microsoft posted an update explaining a new “evolution” to rovnix that had been found.

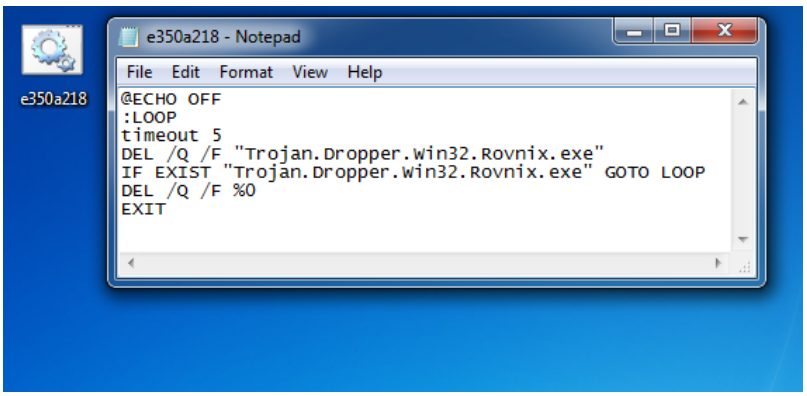


“evolution”

The so-called Evolution

I’m Melting

The first thing i noticed was the file “melts” (delete’s itself once run (well, tries to)), this is done by a lot of malware to prevent future forensics, but how this sample does it is a little less than elegant.



So advanced

The bot drops a non-hidden batch file to the location it's run from (in my case the desktop), the batch file just uses the "DEL" command on an infinite loop, which uses all of the CPU, until the file is deleted. On my test system, the batch file actually fails because the executable locks the file, meaning it can only be deleted once the executable stops (the system reboots), when the system reboots windows stops the batch file before the executable, thus it's never deleted.

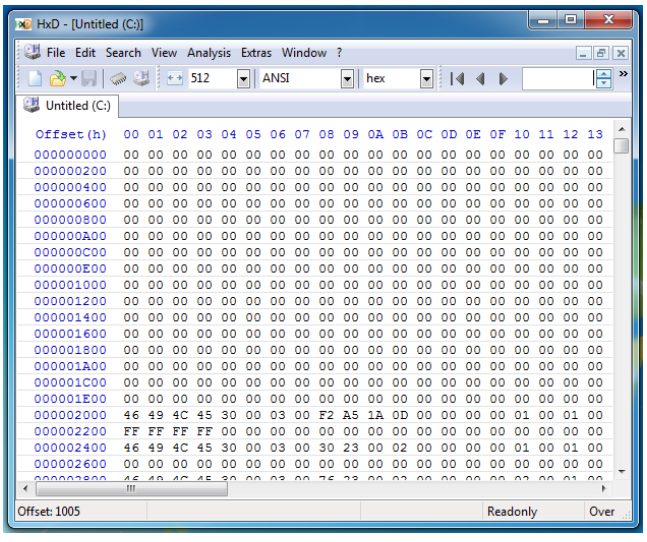
Initial Infection

After executing the packed binary it unpacks itself and continues running, the above batch file is deployed to delete the dropper after run. For stealth reasons, the kit sits idle for an undefined amount of time (I'm yet to find out how this is done), then the system is automatically rebooted. NtShutdownSystem is hooked to receive notifications of shutdowns / reboots, so rebooting the computer will result in intimidate infection and save you a wait. Amusingly the packed dropper doesn't exit until reboot and the delay is long enough to attach a debugger, dump the unpacked code from memory, then move it to another computer.

The entire kit is packed inside the dropper, about 13 files total (32-bit and 64-bit) and during the reboot delay everything exists in one continuous block of memory and once dumped; the components can be split up by byte signature "0x4D 0x5A 0x90" (DOS header).

No VBR

The first thing i noticed after infection is that the first 16 sectors of the disk are blank (where the VBR should be located). To anyone familiar with rovnix this is a common sign of infection, as it uses the kernel driver to hide infected sectors (which is probably just as suspicious as showing them).



Ermahgerd

Stability

The kit appears to run ok on Windows XP 32-bit, but on Windows 7 64-bit it causes the PC to BSOD about every 20 minutes (Sometimes the system can even get stuck in an infinite BSOD loop).

Anti Reversing

I can't tell if I'm going crazy or the anti reversing protection is what I think it is. The driver appears to check for a wide variety of reversing tools (vbox, vmaware, wireshark, ollydbg, ida, lordpe, etc), then disregards the results and exits the thread. My tests appear to confirm this as I've infected a VM with multiple blacklisted tools running and the malware still continues the infection.

```
DWORD Antis()
{
    int Return = 0;

    if ( IsVirtualMachine() )
        Return = 1;

    if ( IsDebuggerRunning() )
        Return |= 2;

    if ( AreREToolsRunning() )
        Return |= 4;

    return Return;
}
```

The return value will be non-zero if any blacklisted tools are found

```
DWORD WINAPI AntisThread(LPVOID lpParam)
{
    Antis();
    return PsTerminateSystemThread(NULL);
}
```

Disregard everything?

I've checked through the ASM multiple time, but can't seem to find anything that would result in the bot being any wiser about the environment after the execution of this thread.

Virtual File System

Remember the old rovnix filesystem? It used raw disk access to store components outside of the filesystem in non-allocated disk space, making it near impossible for the AV to find or remove? Well that has been "upgraded". The virtual filesystem is now stored inside a file in "C:system32", free for the AV to delete at any point (Coder couldn't figure how to access virtual filesystem from usermode?). The file-name ends in ".bin" and is a 64-bit hex value generated using the time stamp counter ("rdtsc" instruction), all files are encrypted with RC6.

```
if ( FileExtension )
{
    TimeStampCounter = RDTSC();
    FileExtensionPtr = FileExtension;

    do
    {
        WideChar = *(wchar_t *)FileExtensionPtr;
        FileExtensionPtr += sizeof(wchar_t);
    }
    while ( WideChar );

    AllocSize = 2 * ((FileExtensionPtr - (FileExtension + sizeof(wchar_t))) / sizeof(wchar_t)) + 78;
    OutputBuffer = ExAllocatePool(PagedPool, AllocSize + 8);

    if ( OutputBuffer )
    {
        *((DWORD *)&OutputBuffer[4]) = (char *)&OutputBuffer[8]; //OutputBuffer[4] = &OutputBuffer[8]
        *((WORD *)&OutputBuffer[2]) = AllocSize;                 //Store size of buffer OutputBuffer[2]

        //Copy size of string to OutputBuffer[0] and string to OutputBuffer[8]
        *(WORD *)&OutputBuffer[0] = 2 * swprintf(OutputBuffer[4],
            L"\\SystemRoot\\system32\\%I64x.%s",
            TimeStamp,
            FileExtension);

        if ( *(WORD *)OutputBuffer )
        {
            result = OutputBuffer;
        }
```

Example File-name Generation

Because the file system is now vulnerable due to it being saved as a standard windows file, the coders have added a high level (and fairly useless) kernel mode rootkit to the driver. The rootkit places SSDT hooks on the following functions:

- NtQueryDirectoryFile – Hides the file.
- NtCreateFile – Prevents file from being opened by all except bot.
- NtOpenFile – Same as above.
- NtClose – No idea.
- NtSetInformationFile – Prevents rename / delete.
- NtDeleteFile – Prevents delete.
- NtQueryInformationFile – Hides the file.

Additionally the rootkit also hooks some registry functions to hide keys, I can’t see any sane reason why.

```
typedef struct HOOK_ARRAY
{
    CHAR *FunctionName;
    LPVOID *HookProcedure;
    LPVOID *OriginalProcedure;
    DWORD SSDTOrdinal;
};

char __stdcall HookFunctions(HOOK_ARRAY *Hooks)
{
    BOOL Result;
    int i;

    if ( Hooks )
    {
        for ( i = 0; Hooks->FunctionName != NULL; ++i )
        {
            Hooks->OriginalProcedure = NULL;
            Hooks->OriginalProcedure = NULL;

            if ( !HookFunction(Hooks->FunctionName, Hooks->HookProcedure, &Hooks->OriginalProcedure) )
            {
                return FALSE;
            }
        }

        ResolveSSDTOrdinals(Hooks);
        result = TRUE;
    }else{
        result = FALSE;
    }
    return result;
}
```

Each hook entry is an array of 16 bytes (4 double words)

Although this won't protect against antiviruses, it may stop usermode malware and beginner security researchers from tampering with the filesystem.

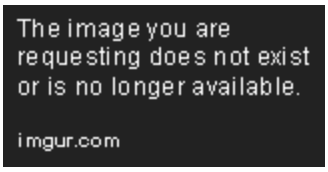
Usermode Component

This entire bootkit + kernel mode rootkit all serves to protect a small trojan which appears to run as a service and do nothing other than log keys and ammyy id's to a command and control server. The MD5 has is: 5e5f3ced234c6f7c91457a875cf4a570.

Conclusion

This isn't the work of common scriptkiddies, it's likely the coder has a moderate knowledge of kernel mode programming; however, the code is not experience with malware (using SSDT hooks and filters in a bootkit, having to move the virtual filesystem into a real file to access it from usermode, using batch file to delete melt exe). This clearly isn't an "evolution" of rovnix as Microsoft claim, it's just some random coders trying to make the bootkit compatible with their bot.

Thanks to Poopsmith for bringing the sample to my attention and Xylitol for retrieving it for me.



Finally, something to reverse.