



[RCE Messageboard's Regroupment](#) > [Forums](#) > [Advanced Reversing and Programming](#) > NTFS MFT Internals

[PDA](#)

View Full Version : [NTFS MFT Internals](#)

WaxfordSqueers

May 9th, 2013, 23:50

I can hear all the Olly users going Arrrrggggh!!!!

I am having severe brain lock. I fired up softice because I want to trace a file creation through kernel mode to see if I can get access to where the system accesses the NTFS MFT structure on disk. When I set a bpx on creatfileA and go to double click my file, createfile fires off on everything....of course. So, I need to use an IF statement in my BPX expression.

If I look at createfileA in kernel32, I see:

```
PUSH DWORD PTR [EBP+hTemplateFile]
PUSH DWORD PTR [EBP+dwFlagsAndAttributes]
PUSH DWORD PTR [EBP+dwCreationDisposition]
PUSH DWORD PTR [EBP+lpSecurityAttributes]
PUSH DWORD PTR [EBP+dwSharedMode]
PUSH DWORD PTR [EBP+dwDesiredAccess]
PUSH DWORD PTR [EAX+04]
CALL _CreateFileW@28
```

[EAX+04] points to the drive, path and filename of the file I want to open.

How can I enter that as a BPX in softice so createfile will only go off when I activate my file?

I am thinking BPX creatfilew IF [EAX+04] = 'c:\(path)\(file)'

I get that flagged as an invalid expression.

Extra questions:

- 1)should createfile look on the disk via the NTFS MFT table to find the file...I think it should.
- 2)which form of createfile do I use? The call that uses the parameters above is createfileW but it has that @28 attached.

WaxfordSqueers

May 10th, 2013, 00:26

Quote:

[Originally Posted by WaxfordSqueers;94668]

I am thinking BPX creatfilew IF [EAX+04] = 'c:\(path)\(file)'

I get that flagged as an invalid expression

Played with it for a while and got past the invalid expression warning.

I used BPX creatfilew EAX+4 == 'drive\path\file' ...ie. no brackets on EAX+4 and == instead of =

It now calls the drive\path\file expression an invalid character constant but it doesn't seem to mind the rest of the expression. I tried it with double quotes and it goes back to invalid expression.

I have double checked the path a dozen times and it is dead on. I have checked the context and creatfilea and creatfilew are both listed under the exp command.

I tried a BPX on readfile and got it to break but I think readfile is too far down the chain for what I want. By the time readfile gets into the act, the file seems to have already been located. I need to get at it as createfile is trying to open it and finding it on the disk.

Kayaker

May 10th, 2013, 02:22

Hey Waxford,

It sounds like you want to do a system break when you double click your file, in which case the first break would be in Explorer. You could set a bp on NtCreateFile.

```
:TABLE ntoskrnl
:bpx _NtCreateFile do "dd esp"
```

Now you can start building the conditional expression.

Code:

```
NTSTATUS NtCreateFile(
    _Out_ PHANDLE FileHandle,
    In ACCESS_MASK DesiredAccess
```

The 3rd parameter of POBJECT_ATTRIBUTES is PUNICODE_STRING ObjectName, so when you get the basic break on NtCreateFile:

```
:dd *(esp+c)
will point to POBJECT_ATTRIBUTES. Dereference that to get PUNICODE_STRING
```

```
:dd (*(esp+c)+8)
then one more dereference to get UNICODE_STRING.Buffer
```

```
:d (*(esp+c)+8)+4)
and we see (for notepad):
```

Code:

```
0023:025D5418 003F005C 005C003F 003A0043 0057005C \.?.?.\C.:.\W.
0023:025D5428 004E0049 004F0044 00530057 0073005C I.N.D.O.W.S.\.s.
0023:025D5438 00730079 00650074 0033006D 005C0032 v s t e m 3 2 \
```

Now you need to get creative, there's a character limit in Softice of ULONGULONG or UQUAD (which explains your problem in your second post; as well I thought that '\ characters need to be double escaped to '\\'), so you can't do the whole string, but you could specify the string offset where "notepad.exe" starts. As you can see in this case it's +0x30 into the buffer, so a working breakpoint could be:

```
:BPX _NtCreateFile IF *((*(esp+c)+8)+4)+30) == 006f006e
```

And we get success:

```
Break due to BP 00: BPX _NtCreateFile IF *((*(esp+c)+8)+4)+30) == 006f006e (ET=3.80 seconds)
```

Sorry, can't answer extra question #1, and #2 is moot but the same principles apply.

WaxfordSqueers

May 10th, 2013, 03:05

Quote:

[Originally Posted by Kayaker;94671]Hey Waxford,

It sounds like you want to do a system break when you double click your file, in which case the first break would be in Explorer. You could set a bp on NtCreateFile.

Hey, Kayaker, thanks for response and good to hear from you. It's scary how much you know about this stuff.

I need time to digest what you wrote but nothing you said is mystifying. It's late and I'll have a go tomorrow.

I did resort to placing my file in the root directory to simplify the path. No good, but I may do that using your suggestion.

I have been reading a lot on the MFT file (Master File Table) in NTFS and it appears Windows reads the MFT initially, when the drive is first accessed, and puts certain parameters from the MFT in memory for later access. I had hoped to read the MFT right from the beginning but I may have to settle for just reading the ROOT attribute, which is essentially the ROOT directory. If I can find that using softice I can then try finding a file further down the B-tree. I know how it's done now but there just isn't much detail out there on how to read an MFT.

NTFS uses a B-tree structure to store files and the first node of the B-tree is in the \$ROOT attribute. I would think the system would have to read the B-tree structure to find any file being opened by Createfile.

It would be nice to find a way to set a breakpoint on an external drive so it would fire when I plugged the drive into the USB port. I'd like to trace it from the time Windows detects the drive being plugged in through to the point where it first reads the MFT to initialize the file system.

I am getting ahead of myself here but maybe I could use your suggestions in some way to get softice to break when the drive is plugged in. Simplifying the process so I did not have to wade through scads of plug & play and other code not essential to my problem is something for me to think about later.

Thanks again for the suggestions.

WaxfordSqueers

May 10th, 2013, 19:39

Quote:

*[Originally Posted by Kayaker;94671]:d *((*(esp+c)+8)+4)*

Kayaker...have followed your examples up to the one listed and I can see the path and filename. However, I am still having grief with _ntcreatefile firing off before I can double-click the file I want to trap. For example, _ntcreatefile was going off on my NIC card (\Device\NetBT_Tcpip), so I disabled it. Then it went off on a pipe (\pipe\gpxhwt.....) belonging to ASP.Net in NET Framework version 2. I have looked into uninstalling Net 2 but it is apparently tied in with versions 3 and 3.5.

In case I am not being clear, I have set a BPX on _NtCreateFile, but when I hit X, then Enter, to get out of softice, softice comes back up again with a BP on _NtCreateFile before I can activate the mouse to start the app I want.

The solution I have in mind is to run softice in VMWare, to get away from all the garbage running in the background.

I don't recall having such issues in the past. If I set a BPX on Createfile, or Readfile, I was always able to get out of softice and activate an app.

Kayaker

May 10th, 2013, 22:20

In theory the IF statement is supposed to prevent premature ebreakulation. Again, the final +0x30 offset is the unicode string offset to the filename itself, in the case of a file in the system root directory. Maybe you're still trying to determine that offset for your own app is I guess what you're getting at.

You could try executing it from /system32 and hope that my derived offset would be the same as yours and write your IF statement the same as mine, just changing the unicode hex bytes to match the first 2 letters of your app.

Or, if it is Explorer that you want to start tracing the file execution from you could use the PID to get the break only in that address context at least. i.e.

```
:BPX _NtCreateFile IF PID == pidofExplorer
```

You may have to hit F5 a few times until your app name comes up but that should eliminate most of the white noise breaks.

That's all I really did to check that initially, under VMWare with not many background processes running I set

```
:BPX _ntcreatefile do "dd *((*(esp+c)+8)+4)"
```

and just kept hitting F5 until I could click my app, then F5 again until I saw the 'notepad' string show in the data window. It just so happened that Explorer was the address context that occurred in.

Maybe try the PID == trick as a start.

Make sense?

WaxfordSqueers

May 10th, 2013, 22:56

Quote:

[Originally Posted by Kayaker;94675]just kept hitting F5 until I could click my app, then F5 again until I saw the 'notepad' string show in the data window. It just so happened that Explorer was the address context that occurred in.

Maybe try the PID == trick as a start.

Make sense?

Makes eminent sense...thanks.

Meantime, I have been using SPYXX and watching the mouse messages sent when I double-click my app. I was hoping to get the process started right after a mouse-click, which has worked for me many times in the past. Then, when the app is processing the mouse message, I can set a BPX on NtCreateFile. I was hoping that if I was in the context of the file process that the NtCreateFile hit would be mine.

Problem so far is that I need WM_LBUTTONDOWN and all I am finding sent for messages is WM_LBUTTONDOWN and WM_LBUTTONUP. The double-click mouse message could be a user-defined message, which complicates matters.

I am also working on my van out front and getting a but frustrated, so maybe I'll put this away for the night.

Thanks for taking the time to reply.

blabberer

May 11th, 2013, 03:39

throw away soft-ice it is old creaky and cranky 😊 (hiding from Kayaker removing myself from linked list)

get windbg its is young creaky and cranky 😊

so you have nt!NtCreateFile (.....,POBJECT_ATTRIBUTES oa,.....) at DWORD PTR SS:[esp+c] lets first make a masm expression out of it

@@masm(poi(@esp+c)) this is a pointer to _OBJECT_ATTRIBUTE so lets make a c++ expression for it

((nt!_OBJECT_ATTRIBUTES *)@@masm(poi(@esp+c)))

it is a structure and ObjectName is Member in this structure and Object Name is a Pointer to another Structure _UNICODE_STRING lets make an expression to this

((nt!_UNICODE_STRING *)((nt!_OBJECT_ATTRIBUTES *)@@masm(poi(@esp+c)))->ObjectName)

Buffer is member in this Structure which is a pointer to a wide character string (wchar_t *)

so lets **express ify** and **c++ ify** and **alias ify** the whole expression so that we can use it as a pattern

as /mu \${/v:instr} @@c++((wchar_t *)(((nt!_UNICODE_STRING *)((nt!_OBJECT_ATTRIBUTES *)@@masm(poi(@esp+c)))->ObjectName)->Buffer)) ;

you can now compare this alias with any wild characterized pattern and / or do whatever you can imagine

lets search for a pattern say *note* as in notepad.exe

.block { r \$t0 = \$spat("\${instr}", "*\${\$arg1}*") ; } ;

it is simple you are looking for a pattern that you will specify in run-time and assigns the result to a pseudo register so you can use this script as a generic script (instead of =====**note** you can use **NETBT** or **waxf** or **squeer**=====

lets compare results and break when we are matching is what is left

lets do it

.if (@\$t0 !=1) {gc} .else { .echo matched \${instr} } ;

whole script below

Code:

```

kd> .block { r $t0 = $spat( "${instr}", "*${$arg1}*" ) ; } ;
.if (@$t0 !=1) {gc} .else { .echo matched ${instr} } ;

```

copy paste to "filepatternSearcher.wds" and put it in a folder where windbg can access it (.wds is a random extension name is supposed to mean windbg script it can be .txt too 😊

now open windbg and set a break on nt!NtCreateFile

Code:

```

kd> bl
0x8056cdc0-0001 (0001) nt!NtCreateFile "$$>a<filepattern.txt note"

```

notice the argument passed note at the end you are all set for breaking on NtCreateFile when the path to NtCreateFile contains the pattern note

if you have a folder called note where your notepad.exe resides and you open badnote.exe in that folder this will break because it is in blind love with a partial word note you may need to correct its near or long vision with proper spectacles and if it breaks too much you can curtail it with /p = _EPROCESS address of process

like

bp /p 0x12345678 nt!NtCreateFile " \$\$>a< filepattern.txt ntdll.dll"

a result from a debug spew enabled build of the above script

Code:

```

\Device\Tcp6
\Device\Tcp6

```

Kayaker

May 11th, 2013, 04:10

"Old, creaky and cranky"

Yep, I resemble that remark fully 😊

I admit, a tool able to pattern match like that is powerful and elegant. Cryptic at first, but elegant. Nice one.

naides

May 11th, 2013, 07:16

"IF statement is supposed to prevent premature ebreakulation"

This statement. . .

This lone statement. . .

Is the gem of what makes this forum a classic of literature!

WaxfordSqueers

May 11th, 2013, 07:41

Quote:

[Originally Posted by blabberer;94677]throw away soft-ice it is old creaky and cranky 😊 (hiding from Kayaker removing myself from linked list)
get windbg its is young creaky and cranky 😊

Thanks Blabberer, I need time to go through what you wrote and it's way too late tonight.

I agree that I need to get into windbg but the learning curve is slowing me down. Besides, softice is working like a charm on my xp box and I have Elenil's hiding apparatus along with sten's old hider.

I tried and tried to use NtCreateFile but it would not settle down. I removed Net Framework, all version, and that cut the garbage dramatically and now I am down to the rasacd (remote service related to Telephony) driver popping up NtCreateFile all the time. I shut off the service...no go. Even with the driver popping up, in between popups, NtCreateFile pops up with nothing there...just question marks in lieu of the filename. I can't get it to stop long enough to double-click my file.

F5 is a shortcut in softice to run the app and I hit F5 over 600 times to see what would happen. The BPX on NtCreateFile went off over 600 times without stopping. Kayaker claims he got it to stop in VM with less overhead. I have it set up on an XP virtual machine but the rust is thick. I have not tried the IF statement yet, which should solve the problem. I just don't understand what is setting off NtCreateFile.

I looked it up and it's a user mode function, so I switched to ZwCreateFile, and bingo, I'm right into kernel-mode and all it's goodies. Found some interesting functions that sound NTFSish, like ZwQueryDirectoryFile, ZwQueryVolumeInformation and CcPpPrefetchMetadata. MFT Metadata is what I am looking for, but I selected Notepad and it has a prefetch shortcut. I have to read up on prefetch versus finding a file that prefetch doesn't know about, but I am getting somewhere. I think prefetch files are kept elsewhere, other than in the MFT structure.

Saw the IRP being assigned, so I must be close to the actual NTFS MFT disk data that is hidden from the user-side OS. I'll get back when I get some rest.

Thanks again.

Kayaker

May 11th, 2013, 17:26

Lo! Naides. I don't know about a gem..., but it might be good on the back of a Woodmann Swag tshirt 🇺🇸

Elenil

May 12th, 2013, 04:46

from what i know things of this get to over filter driver to ntfs.sys ?

WaxfordSqueers

May 12th, 2013, 06:24

Quote:

[Originally Posted by Elenil;94692]from what i know things of this get to over filter driver to ntfs.sys ?

Hey, Elenil...I am just reading about that now but from what I understand filter drivers are related to OEM manufacturers who want to divert and modify code specific to their equipment, in this case a hard drive. It might not hurt to find the filter driver, if it's there, and see if I can set a BP on one of it's functions. There are no drivers supplied with my drive, however, which is an external USB type from Western Digital. They have likely supplied a driver to Microsoft and had it incorporated into Msofs kernel. Don't know.

I am taking some time right now to read about functions I have encountered in ntoskrnl and win32k. In fact, I am reviewing the entire Windows API since I have become quite rusty. I figure it would be easier to trace through the kernel if I recognized functions and knew which ones to jump over and which ones to trace.

I am already fairly comfortable tracing through the kernel and I know that HAL is the interface between the disk and the OS. I would imagine filter drivers would lead to HAL but I don't want to make presumptions right now, at least not till I get an idea where I am going in the code.

WaxfordSqueers

May 13th, 2013, 01:37

Quote:

[Originally Posted by Kayaker;94675]In theory the IF statement is supposed to prevent premature ebreakulation.

Kayaker...found reasonable explanation for why NtCreateFile is failing on me and contributing to premature ebreakulation. If only it would explain the other kind.

Normally, a system service request from user mode requires an Int 2e on earlier Pentiums or a sysenter on more recent models. However, if the caller is already in kernel mode, the int or sysenter is not required. Apparently, drivers are not supposed to access system calls directly, even if they have a proper privilege level. At least, that's my understanding, which is still a bit foggy.

If NtCreateFile is called directly, the kernel notes the previous mode value and sees that the call was initiated in u-mode but that the desired address is in k-mode. It fails the call. I have noted that you were able to get in eventually but I tried several hundred times without NtCreateFile pausing to let me double-click my app. Perhaps the system service interrupt is set up differently in a VM.

The way around that is for a driver to call ZwCreateFile. Apparently the Zw version is not an alias, or a wrapper around NtCreateFile. It does not generate an INT or a Sysenter, rather, it builds a fake interrupt stack then calls KiSystemService directly. In other words, it emulates a CPU interrupt. The handler executes as if the call came from u-mode, but it detects the privilege level the call came from and sets the previous mode to kernel.

NtCreateFile now sees the call came from k-mode and accepts it.

That's my story and I'm sticking to it...for now, at least. 🇺🇸

Keep in mind we are talking about drivers and I don't know how that might apply to the softice driver.

Anyway, the moment I set a BP on ZwCreateFile, there was no more premature ebreakulation.

WaxfordSqueers

May 13th, 2013, 02:36

Quote:

[Originally Posted by blabberer;94677]get windbg its is young creaky and cranky ☺

Still have not attacked the theme of your post, busy reading about kernel processes. However, one of my sources uses LiveKD. My understanding, from well in the past, is that windbg requires a remote processor set to debug mode. That's why LiveKD was produced, to use windbg on one system.

What's your take on that? What kind of setup are you using?

I normally work on a laptop these days, running win 7 but I have a desktop nearby running XP, with all my reversing stuff on it. I think I have windbg on there as well. I have a primitive network setup between the two but I use it primarily for file sharing. Could I setup windbg on the laptop and debug on the desktop. Reason I ask is that the laptop is online and has access to symbols but the desktop does not have access. I could always d/l the symbols if packages are still available.

WaxfordSqueers

May 13th, 2013, 05:19

Quote:

[Originally Posted by WaxfordSqueers;94706]...busy reading about kernel processes.

Sorry...had to take a break from reading and express my frustration. Windows is bloatware of the highest order. I am reading about objects and still trying to figure out why this obfuscation is necessary. I am reading it only because I have come across a reference to a hardware object in the code (I call it a disk drives, but what would I know, being one of those lowly hardware types?).

I imagine the people who came up with this have pants that come 6" above their shoe tops, held up with suspenders, belts, and ropes for good measure. They must hate hardware because they have taken great pains to make it appear as if their obfuscated, virtual reality is somehow running the show while the processor is there as a show piece, like a blond babe with a sugar daddy. Object-oriented logic is the quantum physics of computing. As Feynman claimed, it works, but no one knows why.

I swear, they could cut out 3/4 of windows overhead and it would still work fine, even better. Is it just me who has noticed while tracing through windows code, how many times a piece of code is repeated, like when verifying a string? It's even worse in the kernel, where myriads of bloated code seems to run endlessly in circles. I don't know how many times I have seen windows code go over the same function several times for no apparent reason. I wonder if Windows programmers have taken the time to run through their bloatware at a low level.

I'd be willing to bet they have solved problems where the code has gotten away from them by patching it so it would return to the calling function.

Gripe ends. Sorry Woody, for wasting your disk space. Move this to off-topic, if you like, or just delete it.

Aimless

May 13th, 2013, 08:39

Sorry...had to take a break from reading and express my frustration too!!

But unfortunately, it's linux this time. :P

Last time I tried Linux (Debian, Red Hat, Slackware, Ubuntu, Mint, Puppy, Parsix, Cent and Gentoo), I came away feeling dirtier than when I had used windows. Linux is not really what it's made out to be. It's one of the least robust OSes I've ever worked on. It's much-vaunted stability, is in reality, about as robust as a freshman girl's virginity on Prom Night with her favourite quarterback crush her current date and lots of empty bedrooms available with free condoms strewn around. The only good system was the UNIX one (AIX and System V) and occasionally, Solaris (but only when running on Sparcs). Netware had uptimes in YEARS, not months, but was too exclusive. FreeBSD was okay - strictly when running it as a web server, not otherwise. No doubt Linux developers often advertise it as the best web server, or file server, or print server --- never a good desktop, though.

The ONLY other system I have come across to liking currently is MAC OS X Lion (but the mountain lion version squeals like a cat when it's tail is stepped on)

I've got XP Pro with SP3 running on my many systems and it's NOT had a downtime for the last 8 months (yeah, that's not a typo) and I've yet to see a memory leak or slowdown. I have a Windows 7 at work and I'm throwing everything on it including the kitchen sink and I've YET to re-install or repair it in the last 3 years (yep, not a typo either).

Just thinking, maybe if linux supported ALL hardware and peripherals that Windows does, had all the backward compatibility that windows has, ran on so many types of PCs, Netbooks, Laptops, Tablets that windows does, came across so many malware that windows did, had to cater to so many different regional settings, had to work for so many varied industries in the world, the code would, perhaps, have not be that bloated. Or maybe if it was not backward compatible with so many software tools and older versions of the programs, it would be 'streamlined' too.

I've also seen hardened linux admins get a confused look on their face when asked why this or that hardware is not being recognized. And I'm talking about the latest version of Ubuntu and Mint -- leave alone slack or debby. Compared to which windows can be setup by a newbie too (another reason for bloated code?). And while linux gurus extoll the virtues of working in CLI (not that I mind it, I use it myself - rarely, though ☺), have they ever tried image manipulation in a CLI? Or music editing in a terminal window? Or, dare I say it, even DEBUGGING CODE in a CLI? Perhaps, to SUPPORT the WINDOWS based debugging, the code is bloated. Perhaps, to SUPPORT so many commands in the debugger, the code is bloated. Try running IDA in CLI. And let me know if you don't go goggle-eyed at that. GDB is good. But not THAT good. Not, OLLY good. Not, WINDbg good. Not, IDA Good.

Thankfully, the people who came up with this code in windows wear pants that come 6" above their shoe tops, held up with suspenders, belts, and ropes for good measure -- because the way Linux works, I am sure THEIR developers coded it without a pair, in their underwear, in their parent's house too, while wiping the sauce from their faces, after they had played with their GI Joe figurines, that is (Hey, it could've been spongebob too, for all I know :P) .

MS Paint is not better than Adobe Photoshop, just because its more 'streamlined', is it? I mean, Photoshop is BLOATED for a reason, I think.

Gripe over. Sorry Woody, for wasting your disk space. Move this to off-topic, if you like, or just delete it.

Hey Waxy, love you. Peace, brother.

Have Phun

WaxfordSqueers

May 13th, 2013, 18:58

Quote:

[Originally Posted by Aimless;94710]Last time I tried Linux (Debian, Red Hat, Slackware, Ubuntu, Mint, Puppy, Parsix, Cent and Gentoo), I came away feeling dirtier than when I had used windows.

Hey. Aimless...whazzup?

I am not complaining about Windows as an OS in general. I am venting at the infrastructure, which is based on abstraction so unrealistic that it divorces an average user from the machine. As I said, I am coming from a hardware POV which is where my expertise is based. Windows marketers have passed the system off as a glossy front end and the developers have pushed concepts like wizards. The wizards are so stupid, that if inf files are missing from the INF directory, they are absolutely useless. I have fixed many a problem by hiding an INF file and forcing Windoze to load 'MY' driver. Even at that, it whines at you.

Also, anyone who has tried to use the Windows help system has surely cried out in frustration. They lead you through questions which in my experience end with them telling you they don't have the answer.

I stay away from Linux because of it's command line system, which is archaic. I started out in computers in the late 1970s and the kind of mickey mouse utilities you still find in the Linux command line system were out-dated in the 1980s. Although you can apparently do more with it than DOS, I found DOS easy to use and logical. There's nothing logical to me

about files with no extensions and directories and disk drives regarded as files. And the security is just plain annoying. I don't need it and it is a hassle trying to read and set permissions on everything encountered.

The Linux GUI is not bad so long as you don't have to look under the hood. I preferred KDE to Gnome but I could not see anything offered that I did not have in Windows. I use both Win 7 and XP and put up with the arrogance built into either system whereby Msoft treats the user as a noob who has never run a computer. Being told that I don't have permission to use a file or directory is way beyond cool. I fix it all with my 'take ownership' utility, as I spit. "ef you, Gates".

I have never consider MAC simply because I'd have to change all my apps.

blabberer

May 13th, 2013, 19:39

Quote:

what is your take on this

sysinternals livekd and latest windbg's local kernel debugging both are fake those are not debugging the system live. they create a snapshot and provide various facets about the kernel state.

(yes it is quiet usefull under many circumstances .

I use the facility quiet often

have written a plugin for ollydbg to use the functionality inside ollydbg blah blah blah)

but both of them are not kernel debugging they are similar to

MRI or CAT or Sonograph or Endoscope seeing something without actually seeing it

hearing something without actually hearing and

feeling something without actually feeling them

yes you need two machines and a transport between them for windbg to work if you want to work with real hot iron

you can kinda fake this and perform a single machine kernel debugging if you don't have to work with hot iron

simply get and install one of the virtual (PC / ware / box /) **(hereinafter called as target)** in your laptop **(hereinafter called as host)**

install your favorite OS (preferably xp sp3 to start with) in the target

create a virtual transport using named-pipe between host and the target

open windbg in host and connect it the virtual transport send a break-in from host (ctrl+break) or from target (ctrl+alt+fn+prtscreen)

enjoy uninterrupted kdculation

Quote:

Zw versus Nt

read the articles they both exist for a reason bloat with garter-belt is not one of them

<http://msdn.microsoft.com/en-in/library/windows/hardware/ff565438%28v=vs.85%29.aspx>
<http://www.osronline.com/article.cfm?id=257>

Quote:

should x api be called before y DDI in fist pose

windbg knows ntfs better than me

started a vpc - kd session

asked what windbg knows about ntfs,mft,read,write with

Code:

```
kd> x ntfs!*r*mft*r*
```

```
fc34fc03 Ntfs!NtfsReadMftRecord = <no type information>
```

```
fc2a2545 Ntfs!NtfsDefragMftPriv = <no type information>
```

set a log break on one of them and on nt!NtCreateFile and dumped the stack to examine it

a bit of googling and observation shows that the third argument to this call is MFT_SEGMENT_REFERENCE or FILE_REFERENCE

again a bit of googling says ntfswalk by dmitybyrant can show the mft details downloading and running it and spleunking a little with windbg

it seems i can narrow down mft reads and ntcreatefile

the script as used for logging mft reads and create file as follows

Code:

```
printdec.txt (ntfswalk shows mft# in decimal so
```

```
.printf "mft# = %6d\tFileName = %msu\n" , poi(poi(@esp+c)) , poi(poi(@esp+c)+20)+8 ;
```

```
gc
```

and output on double clicking the waxford.txt file in vpc's Desktop till opening of the file in desktop

Code:

```
mft# = 10535  FileName = NTUSER.DAT.LOG
```

Aimless

May 13th, 2013, 21:45

Waxy...love ya!



Have Phun

WaxfordSqueers

May 14th, 2013, 00:41

Quote:

*[Originally Posted by blabberer;94718]**yes you need two machines and a transport between them for windbg to work if you want to work with real hot iron*

Appreciate all the work Blabs, hope I am not taking you away from something important.

In the old days, it required a serial cable between host and target. I don't know if USB -> USB has functionality for that these days. I have a LAN working between my laptop and desktop via NIC cards but it is unstable. Works for file transfer but it complains a lot and sometimes refuses to work.

I do have a VM setup (VMWare) with windows and I recall trying to start a pipe at one time between the VM as target and windbg on the host. For some reason it never got going. I'll play with it.

Quote:

[Originally Posted by blabberer;94718]read the articles they both exist for a reason bloat with garter-belt is not one of them

<http://msdn.microsoft.com/en-in/library/windows/hardware/ff565438%28v=vs.85%29.aspx>
<http://www.osronline.com/article.cfm?id=257>

The second article is good and reflects what I read in a book by Russinovich (sysinternals and now microsoft guru). I'll have to read it more closely and poke around in the code. I understand the number passed in EAX is an index into the system service table and the pointer in EDX points to a user stack. Sysenter now makes perfect sense to me. However, if these funcs are identical as claimed by msoft in the first link, and we know how they tend to mislead users by withholding all the information, that does not explain why the softice driver can't deal with NtCreateFile (wont break on a bpx) yet works perfectly with ZwCreateFile.

My reference to bloatware was not to the Nt/Zw names and usage, it was to the whole idea of objects. Why not call a file a file and a disk drive a disk drive? What's the penchant for obfuscating real world objects with virtual creations that are so generalized they fail to make sense? Once again, I think the problem is with msoft trying to keep trade secrets too close to the vest.

I understand the need to virtualize certain facets of programming, to distinguish the real disks and files from their images in memory. But msoft is getting carried away developing a language and implementation that is so broad that someone could learn this jargon and never understand what a real disk drive or file is. It's like at university, where some profs teach math from an unrealistically broad set of definitions that first year students get totally messed up. I had a prof tell our class that one could take the square root of -1 in certain situation. I asked for an example and he just stuttered. I was a marked man from that moment on but I was not about to let some egghead spin bs about math. I know about the usage of the square root of -1 in complex numbers but no one takes the root of -1 there.

Quote:

*[Originally Posted by blabberer;94718]windbg knows ntfs better than me**started a vpc - kd session**asked what windbg knows about ntfs,mft,read,write with*

found this part really interesting but I need to get up to speed on what you are talking about. The ref to Ntfs!NtfsReadMftRecord is interesting. I'll have to sift through ntoskrnl symbols and see if there are other NFT references. Maybe even the HAL symbols.

What I am trying to do is approach the \$MFT file from its entry point and trace through it's front end so I can see where it goes, especially from the \$ROOT file.

Each of the first set of files are metafiles and begin with the header 'FILE'. The very first entry in an MFT file is FILE, like MZ is the first entry in an exe file. From there onward, there are attributes at various offsets, and each attribute begins with a number. It's like a PE header and I want to learn to read it.

Trouble is, as I understand it, once the disk has been accessed, NTFS reads the files and moves info to another location. Also, a file like Notepad, if used regularly, will have a prefetch record and the MFT will likely be bypassed. So I need to catch the NTFS system in the act of initializing the MFT. I want to see what it's initial entry point is into the table, and why.

WaxfordSqueers

May 14th, 2013, 00:47

Quote:

[Originally Posted by Aimless;94720]Waxy...love ya!*Have Phun*

Worked with a guy years ago and he always posed the question, "Having fun yet?"

blabberer

May 14th, 2013, 02:02

the free vmware player is more than sufficient and you can use vmkd a superfast transport in this
 the free vpc is quiet good and the virtual serial interface with namedpipe on com1 is what i normally use
 the free virtualbox has its own followers and iirc source code is available if you are bent on recompiling it with some super duper crap

all of these work very very superbly for kernel debugging in virtual environment without having a second machine / and / or comport (a rarity nowadays)/ null modem cable / firewire / 1394 / debugusb / network (yes real tcp network transport is available in win8 the touch touch feel feel duplo crap ui)

set aside three to four hours and decide to make it work and you wont regret dumping softice and its quirks and limitations.

square root of -1 = i or -i ☺

ntfs!NtfsReadMftRecord is a ntfs symbol you need to look int ntfs.pdb for it not in ntoskrnl symbols

if you were on windbg it would be as simple as .reload /f ntfs.sys and then using x (examine)

in softice you would need the .nms or some such file for ntfs.sys

i am not sure how reading memory with debuggers is going to get you anything from a physical media
my understanding till now regarding recovery is you need some disk utility to read the physical media scanning for patterns
and getting deep from there so since it is offline / static / //// imur nothing is going to move from anywhere to anywhere in a physical disk
or do they move even if you do read only access ?? iam absolutely no good at this game (i make backups and simply zero out dod style fdisk and repartition and reformat and install new

WaxfordSqueers

May 14th, 2013, 04:40

Quote:

*[Originally Posted by blabberer;94724]
square root of -1 = i or -i ☺*

Appreciate the info on vmware, my info is outdated. Forgot that ntfs.sys is a file. Better check it out. It's no problem making an nms for softice but I will look into windbg.

BTW, the derivation of i is $i^2 + 1 = 0$, therefore $i = \sqrt{-1}$. The i means nothing other than in applications like electrical engineering but the root of -1 is never taken. i forms a complex plane with real numbers in which i is imaginary. That imaginary component can be used to represent reactive power in an electrical motor, which represents the reactive power required to maintain the motor's electric fields, for it's magnets. The real component represents real power.

It becomes a big issue with Hydro as more and more magnetic devices come on line from factories. The current through an inductively reactive device is out of phase with the applied voltage, and as more reactive devices come on line, the phase of the current lags the voltage more and more. It's called the power factor and a unity power factor is ideal, meaning the current is in phase with the voltage, which is the case in a purely resistive device. As it lags, the pf drops down below 90%, and Hydro gets worried because it's supplying a lot of power to feed the magnetic fields in reactive devices that is doing no good work.

In factories, they have huge banks of capacitors, through which current leads voltage. By applying the capacitor banks across a motor load, it brings the lagging current closer to unity by offsetting the inductive load.

Quote:

*[Originally Posted by blabberer;94724]
i am not sure how reading memory with debuggers is going to get you anything from a physical media
my understanding till now regarding recovery is you need some disk utility to read the physical media scanning for patterns....*

When you call a file, the windows kernel has to i/f with the physical media to retrieve the data. Since the MFT in NTFS is regarded as one large file, something has to read it. I am not trying to recover data, I am trying to repair a link that has been broken in the MFT. Most of my data is sitting there in files and directories but the link is broken. In essence, I am troubleshooting (reversing) a broken link in the MFT.

MY MO is to find how Windoze reads the disk, especially the MFT, so that I can look up in the real MFT how to access directories and files. I plan to do that by reading a known good NTFS MFT table. If I can repair the broken link, I may be able to put the files system back the way it was before I messed up.

I would imagine HAL, or something, reads the standard 512 bytes from disk, or multiples thereof, and reads it into a buffer. Then something must process the buffer to get the info it needs to access files stored in a binary tree. Otherwise, how would they ever retrieve a file of any kind, or know where to read a directory? The system must know where the MFT pointer is located in the boot record section and it must know where to go in the MFT to get the file attributes.

I have found nothing on the Net that goes into it at such a low-level.

I have a couple of excellent disk editors, both freeware. One of them, Active@ Disk Editor, allows me to write to disk much like Norton Disk Editor. It also has a really good facility (called a template) that allows me to place the cursor on the 'FILE' header of an MFT file, and it will tell me, using tooltips, what each attribute means. It even colours each attribute a different colour. Another template, when placed on the boot sector, tells you what each offset means.

The other disk program (testdisk), recovers files by reading the file signature. So far, it has recovered most of the known files on the disk but I am look for one that was a backup of another computer. I don't think testdisk would have the signature for that file since it was a Microsoft backup. I have since made a good backup of that computer. It does not always recover the original filename, so I have to open each file to see what it is. With exes that can be dangerous, as you know, so I look up the file properties or use a hex editor to read the file info in the resource section.

NTFS uses a numbering system for files that I need to learn how to decode. Small files, under about 1K are kept in the MFT and called 'resident'. However, larger files have only their front end in the MFT and reside outside the MFT and are called non-resident. Each non-resident portion is pointed to by the file number, which lists the offset and length. As it stands, I don't even know where to look for the file number. It's in an attribute but I don't have a template for the \$ROOT file, which is the root directory in NTFS. I thought I might be able to trace into it through Ntoskrnl and HAL, or some other low-level access.

I pulled a bonehead move and used Comodo Backup in clone mode thinking I was creating a disk image. It ran for several seconds before I realized that and I managed to stop it. The mirror MFT is intact and I can replace the damaged MFT portion covered by it but I am not sure yet if too much as been damaged. I wont know until I trace into the MFT to see where the break is located.

I don't know if Comodo did a bit by bit clone, beginning at the the first sector, or whether it has an algorithm whereby it writes the MFT first. The MFT is well into the disk(cluster 0xC0000) and I am hoping the clone program did not alter it too much. It obviously did something but I am not sure as to the extent of the damage yet. Many of the original MFT files are still there...I think. ☹

It's not critical that I recover the data. I am just curious as to whether i can reverse the damage, and it gives me a chance to delve into NTFS and the kernel.

blabberer

May 15th, 2013, 20:03

yep i understand real numbers and complex numbers and power factor have tinkered with an automatic power factor corrector that connects and disconnects capacitors as needed for radyne / elind induction heating furnaces / esab welding transformers

like i said i was wondering what this ntfs is about and opened my harddisk in a simple fopen kinda program and tried looking for FILE0 and FileName

it seems decoding this is not much of a deal
google is strife with info and all it would take is a bit of dedicated hours

my simple program using pure crt lib functions like fopen , fread , ftell , & printf
can open my harddisk of laptop (about 112 gb) and is able to list almost all of the first 2000 files names i asked from it in reasonable minutes

Code:

```
#include "stdafx.h"
#include <windows.h>
```

result as follows

Code:

lets open the harddisk and look at it

FILE0 found at 7a600 file name is CA0834~1.CAT

FILE0 found at 7a600 file name is CA0A5A~1.CAT

WaxfordSqueers

May 16th, 2013, 03:32

Quote:

*[Originally Posted by blabberer;94730]
it seems decoding this is not much of a deal
google is strife with info and all it would take is a bit of dedicated hours*

oooooh...some people's kids. All I have been doing the past week is scouring the net and reading book after book on the NTFS and the Windows kernel. I have forgotten what sunlight looks like.

I appreciate the work you have done on this but your program is far too simple. I can find all the FILE headers using Active@ and I can see files in there as well, but it's not as straight forward finding a file in the MFT and putting a directory back together. The FILE header has nothing to do with the files per se, it is a header for metafiles, which describe the system. There are at least 30 different metafiles, all beginning with the FILE header.

eg. hope the formatting holds

Inode-----Filename-----Description

0-----\$MFT-----Master File Table - An index of every file...pointer to \$MFT start cluster in Master Boot Record.
1-----\$MFTMirr-----A backup copy of the first 4 records of the MFT....ptr to \$MFTMirr start cluster in MBR.
2-----\$LogFile-----Transactional logging file...chkdsk writes to this log and uses it to roll back errors during caching.
3-----\$Volume-----Serial number, creation time, dirty flag....you can see disk volume name here....dirty flag for chkdsk.
4-----\$AttrDef-----Attribute definitions
5-----.(dot)-----Root directory of the disk
6-----\$Bitmap-----Contains volume's cluster map (in-use vs. free)
7-----\$Boot-----Boot record of the volume
8-----\$BadClus-----Lists bad clusters on the volume
9-----\$Quota-----Quota information
10-----\$UpCase-----Table of uppercase characters used for collating

These are the first 10 files (records) in the MFT and they all begin with 'FILE'. To make matters confusing, everything is a file and each file has attributes. File #6 at inode 5 is called the dot file as in '.' of normal directory roots. That's where you need to look for directories and files, but as you might guess, directories are files as well, and they have attributes.

The top node of the B-tree structure used to store files and directories is located in the dot file.

Each one of those files has attributes as follows:

Type-----Name-----
0x10-----STANDARD_INFORMATION-----file creation date, etc.
0x20-----ATTRIBUTE_LIST-----used when there lots of files that wont fit into the MFT
0x30-----FILE_NAME-----stores name of file attribute
0x40-----VOLUME_VERSION-----
0x50-----SECURITY_DESCRIPTOR
0x60-----VOLUME_NAME-----name of volume
0x70-----VOLUME_INFORMATION
0x80-----DATA-----contains information about files data i.e. the actual files.
0x90-----INDEX_ROOT-----root node of the data file/directory B-tree
0xA0-----INDEX_ALLOCATION
0xB0-----BITMAP-----has similarities to the FAT, but very loosely
0xC0-----SYMBOLIC_LINK
0xD0-----EA_INFORMATION
0xE0-----EA
0xF0-----PROPERTY_SET

So, the first table is the actual metafiles and items from the 2nd table are found in whichever metafile needs them for description.

For example, the STANDARD_DEFINITION begins with an 0X10. The data (actual files) section begins with 0x80.

If you want to see how they are laid out, download the free app Active@ Disk Editor and have a look at the templates section. You can find the address of the MFT by using the template for the master boot record. Then look up the MFT and it will begin with 'FILE'. Align the template over the beginning of FILE and you'll get a nice colour coded layout with tooltips that reveal what each part means as you hover the mouse over them.

I understand all of that quite well now, after hours and hours of reading, what I am looking for now is how to interpret entries in the attributes that describe how files are connected from the the root out.

I thought one way would be to trace the system using softice because I have not seen an adequate description of how that is done. In other words, I am breaking new ground, as far as I know. Someone has no doubt done it and I have seen a program that is trying to do it. The code is here:

<http://www.autoitscript.com/forum/topic/94269-mft-access-reading-parsing-the-master-file-table-on-ntfs-fileystems/>

Here's the source of my two tables above:

<http://dubeyko.com/development/FileSystems/NTFS/ntfsdoc.pdf>

and another <http://grayscale-research.org/new/pdfs/NTFS%20forensics.pdf>

WaxfordSqueers

May 16th, 2013, 23:47

Quote:

*[Originally Posted by blabberer;94730]my simple program using pure crt lib functions like fopen , fread , ftell , & printf
can open my harddisk of laptop (about 112 gb) and is able to list almost all of the first 2000 files names i asked from it
in reasonable minutes*

If it interests you, I have found an interesting app at the Code Project, with source code for helper library headers and samples of the code, that claims to parse the MFT.

<http://www.codeproject.com/Articles/81456/An-NTFS-Parser-Lib>

At this link, there is code for an undelete app that uses the headers from above:

http://read.pudn.com/downloads149/sourcecode/windows/system/642135/Undelete/NTFSDrive.cpp___.htm

I saw in another thread that there is a problem with the compiled version of this code, but it seems to be minor. I am trying to find the reference again and I'll forward it if I do.

blabberer

May 17th, 2013, 06:35

ooohh some peoples fathers glad i provoked and glad you replied this thread atleast gathers moss

the autoit script guy has a google code page that has a few more autoit scripts for extracting files as well apart from the mft2csv

i just read the greyscale pdf (couldnt download when i was searching) has code similar to what i wrote 😊 and the unfinished structure is completely same apart from the fact that i have a 0xd0 filling that is AttrData left out pending research

apart from them alex ionescu seems to have decoded ntfs in visual basic when he was probably running in diapers (some peoples children)

ntfsdoc is the mother of all docs it seems by russon & fledel (referenced from lot of pages that has the keyword NTFS)

thomas schwarz ntfsfs forensics ppt mft structure is what i used when i cooked the code above

like i said ntfswalker by dmitrybyrant is a free program that displays all these tables in a bit understandable format right from sequence id 0 to N

btw plain hxd hexeditor does a fine job of showing me the raw layout of hdd which i collated with ntfs walker and the output of my program

i accept it is not as trivial as i made it to be 😊 but if you persist and i persist we may be able to document half the ntfs in this thread

as both of us has got nothing to lose or no money to gain but just pure satisfaction of doing something we like and doing it in the best possible way

have fun

hope you finally unravel how to traverse the ntfs web like a spider getting to anywhere from anywhere and everywhere methodically

by methodically i mean we can probably extract and then use secure erase / wipe to zero the extracted file and the harddisk becomes clutter free

and with the process of extraction and elimination we have less and less data to decode

and one day some day the hard disk will be clean without any files just as plain as a newly fdiskd formatted factory sealed disk 😊

WaxfordSqueers

May 17th, 2013, 07:39

Quote:

[Originally Posted by blabberer;94736]hope you finally unravel how to traverse the ntfs web like a spider getting to anywhere from anywhere and everywhere methodically

I'll answer your post in more detail later. It's 5 am and I am getting burnt. I am supposed to be out exercising, not getting myself bleary eyed traversing the ntfs web. However, it is intriguing.

Made some headway tonight with good old softice and some basic hacking. I have no idea at this point what kind of breakpoint to set or which one. So, I tried SetFilePointer with ReadFile. Then I set a BMSG on LBUTTONDOWNLCLK (don't use that often) with a HWND I found in SPYXX for the file manager I use. SPYXX is kinda neat because you can set the window to spew messages for a particular HWND and it tells you which window spewed them. Only one was spewing the double click message, so I did a BMSG on that one with the handle and softice popped up in my file manager after I double-clicked notepad.

From there I tried F5 (go in softice) with both SetFilePointer and ReadFile set, but I was getting way too many hits on ZwReadFile. So I did it with just SetFilePointer and F5'd till notepad popped up. It took 16 hits, so I reset and broke after 14 hits, then traced. It was not long till I hit NTFS.SYS, which seemed encouraging.

Somewhere along the way I hit FsRtlCopyRead in Ntoskrnl at offset B7E5CE94. The literature says that function does a fast copy read from a cached file to a buffer. A bit later, in the function call, there is a call to _CcGetActiveVACB, where VACB is the virtual address control block. The lit claims that if a file mapped into the VACB has a name, the name gets listed. However, if it has no name, the VACB is caching metadata. Metadata is the magic word in my mind for MFT stuff. Since I just came through NTFS.SYS, it seemed reasonable that ref to metadata meant the MFT.

They showed an example related to the VACB using the kernel debugger as follows:

```
kd> !filecache
```

It returned a list with many MFT records in it.

It's not clear to me at the moment the difference between windbg and kd. Does kd require the same rigmarole as windbg or is it just part of the same package? Or is kd the command line part and windbg the GUI?

It seems that windows reads the MFT records while accessing the drive initially and caches them for future use. That's where I left off tonight. I am hoping that if I dig further into this that I will be able to see how Win and NTFS access the metadata to find a file. I have turned off prefetch but I am not sure if Windoze caches files in the same way it does prefetch data. If it does, I need to find a way to flush it's cache so it starts over. Or, approach it using my other scheme of unplugging the external drive, rebooting the OS, and plugging the drive back in with a breakpoint on an I/O port. Have not tried that before so it would be new ground for me.

In Win 7 you can set the drive offline using the MMC. I don't think there's a way to do that in XP.

Have fun yourself, or phun, as aimless poots it.

Aimless

May 17th, 2013, 08:18

Quote:

[Originally Posted by WaxfordSqueers;94725] ...

I would imagine HAL, or something, reads the standard 512 bytes from disk, or multiples thereof, and reads it into a buffer. Then something must process the buffer to get the info it needs to access files stored in a binary tree. Otherwise, how would they ever retrieve a file of any kind, or know where to read a directory? The system must know where the MFT pointer is located in the boot record section and it must know where to go in the MFT to get the file attributes.

...

My suggestion?

Use Ubuntu in the INSTALLATION AS APP mode using the WUBI installer (link on Ubuntu website itself).

Code:

<https://wiki.ubuntu.com/WubiGuide>

Ubuntu is the only linux installation that I am currently aware of, that has a unique installation mode. The Windows APP mode. Here, you install Ubuntu as an application (just as you install Word, or Office or Comodo). The software (in this case, our OS) installs. All you need is to create a folder where it will be installed.

THEN, Ubuntu simply changes the boot.ini and includes a place to boot itself. Reboot the machine, and it will give you a choice to boot into Ubuntu or Windows.

The main point, however, is that Ubuntu, once booted like this has 2 very important characteristics:

1. It can read NTFS (read, write, execute... you get the idea) including all DRIVES residing in Windows.
2. It uses Windows SYSTEM FILES (HAL, NTOSKRNL, et al) for everything. Which means, if you go into Ubuntu, DELETE these files, and reboot, UBUNTU will NOT WORK. That's because its installed in the APPLICATION mode. To uninstall UBUNTU, simply boot back in Windows and Program->Uninstall. Peachy!

However, Point no. 2 has a deeper meaning.

IF Ubuntu/WUBI uses HAL and NTOSKRNL to read/write/execute everything, THEN it must know how to read/write/execute from it. Comprendre? THIS means, if you download the SOURCE CODE for UBUNTU/WUBI, then you will get an idea, where in the APPLICATION mode, is the HAL/NTOSKRNL et al being used, and HOW (plus, C/C++/ASM code in all its glory!) No need to scour google. No need to read up searchlores.org to polish your searching skills 😊 (heh, hello mods, I'm unable to wget any of the pages from woodmann.com/searchlores just thought you should know -- even with the robots.txt control OFF -- but I digress)

Mayhaps, that could be a slightly easier way to look at what's happening at HAL/NTOSKRNL levels without delving too much into Google. Just another attack vector, if you will.

Have Phun.

WaxfordSqueers

May 17th, 2013, 23:05

Quote:

[Originally Posted by Aimless;94738]My suggestion? Use Ubuntu in the INSTALLATION AS APP mode using the WUBI installer (link on Ubuntu website itself).

Thanks for suggestion. It's a good idea and worth checking out but I am being stretched in several different directions right now and tending to lose focus. Sometimes that is part of reversing, as you know.

Having the source code for the Ubuntu kernel is obviously a plus.

Quote:

[Originally Posted by Aimless;94738]IF Ubuntu/WUBI uses HAL and NTOSKRNL to read/write/execute everything, THEN it must know how to read/write/execute from it.

Not necessarily. Ubuntu knows how to call functions in those windows kernel modules but they are the ones that know how to interpret disk I/O. I could be wrong. For example, in what I know of C++ code, all the calls are high level, unless they build in inline assembly instructions. I would imagine all you're going to see in Ubuntu source is calls to Hal, Ntoskernel, NTFS.sys, etc.

blabberer

May 18th, 2013, 14:04

windbg is a frontend to kd kd can perform better in few circumstances where windbg wont run (like remotely terminally ill platforms with no access to food water or weed)

a quote from windbg doc

Quote:

If the client is unable to send a connection request to the server, but the server is able to send a request to the client, you can use remote debugging through the debugger with a reverse connection by using the clicon parameter.

Remote debugging through remote.exe is used to remotely control a Command Prompt window. It can be used to remotely control KD, CDB, or NTSD. It cannot be used with WinDbg.

oh btw i cleaned up the src i posted few threads above to use a few structures

and it can now find all the file names 498 filenames out of 512 asked for
the remaining few have no attribute 0xffffffff as the first header
and few have attribute offset > 0x50 (ima readin only 0x200 bytes)

you can also stuff these structs back into ntfs.pdb and build an nms of the modified pdb to use with softice

since you brought up cache

one of the places where nt plays with cache is nt!ccmapdata

you can use the struct in the code i post below like this

Code:

```
kd> bp 8056d719 "dt Ntfs!_NTFSMFT (@ecx & 0xffffffff00)"
```

```
breakpoint 0 redefined
```

```
kd> bl
```

the code i cleaned up is below

Code:

```
#include "stdafx.h"
```

here is how to put the structures back into ntfs.pdb

Code:

copy the three defined structures into a file named xxxx.c and declare them copy the xxx.c file to a new folder

```
typedef struct _YYYYY{
```

WaxfordSqueers

May 18th, 2013, 22:24

Quote:

[Originally Posted by blabberer;94741]windbg is a frontend to kd kd can perform better in few circumstances where windbg wont run (like remotely terminally ill platforms with no access to food water or weed)

I have the Debugging Tools for Windows set up on my XP (they have been there for a long time) and when I called up KD it gave me an option to connect with various means, none of which I have immediately available. However, the 'Local' choice brought up LiveKD, so I played with it.

I was able to use !filecache, !ca, !fileobj, etc. to examine the VABC table, also DC to dump some stuff. Barely got started.

The network setup I have between XP and my laptop (Win 7) is not stable. i.e. it works sometimes and sometimes not. I am looking in to a USB - USB network but it means getting the special cable with the chip in it that allows USBs between computers to operate without blowing the USB bus. Apparently it comes with software so you can use it for either straight file copies or for networking.

Quote:

[Originally Posted by blabberer;94741]oh btw i cleaned up the src i posted few threads above to use a few structures

Again, I appreciate the amount of work you have put into this and I have read it with great interest. I have little practical experience programming in C or C++, just enough to make myself dangerous. I think I have Visual C++ 6 somewhere, so I'll have to set it up again.

I was surprised that you have a pdb for NTFS.sys. I looked in my symbols file I have on disk and there is no PDB listed for it. I made one using the IDA2ICE plugin for IDA. I'll have to look closer for a pdb.

I am driving myself a bit wonky through the focus I have put on this stuff the past couple of weeks. I tend to get into something like this to the point where my health suffers due to lack of exercise and sleep issues. I never learn. I am going to have to back off a bit, temporarily.

Meantime, I have made a couple of screen capture of Active@ to show you how they have captured the data extracted by your program.

In capture '\$Boot.jpg', at offset 0x30, you can see the pointer to the \$MFTfile. At 0x38 is the ptr to \$MftMirr, the back up of MFT. Those are cluster offset so they have to be converted to byte offsets. My MFT is at 0xC0000 and my Mirr is at cluster 2.

\$MFT001.jpg shows offset 0xC0000000 and (0xC0000 clusters) x (0x8 sectors/cluster) x (0x200 bytes /sector) = byte offset 0xC0000000

Note that 0xC0000000 begins with FILE. I have placed the NTFS MFT File Record template over the F in File (ie. offset 0 of the MFT) using right-click Set Template Position. You can set bookmarks in Active@ to quickly return to your previous view. Look in the template table and you will see all the file record headers you have listed with your app. Also, you can hover the mouse over the offset and it will display some info from the template table.

Sorry...for some reason the system wont let me upload jpegs. I get a red exclamation mark each time I try. Maybe files are too big (250K) or maybe I need to zip them).

2765

Let me know if they come out OK.

I made some headway tracing the files and directories using the info from Active@ template. If you look down the table further you get much more info. For example, it lists 'data runs' which is a file's actual data stream. You will see various attributes highlighted and if you open the 0x80 attribute, you'll see data runs in some. On one jpeg file I was able to trace it from the an MFT record right to the file, but I have not learned yet how to identify a directory. Apparently directories have an \$I30 header in them, although it's not always aligned with a paragraph boundary like a FILE header is.

I am still working with softice and painting myself into a corner by approaching this from too many angles. :-)

blabberer

May 19th, 2013, 02:42

I have the Debugging Tools for Windows set up on my XP (they have been there for a long time) and when I called up KD it gave me an option to connect with various means, none of which I have immediately available. However, the 'Local' choice brought up LiveKD, so I played with it.

don't give me this again it is absolutely not possible that nothing could be available
i said you **DONT NEED TWO MACHINES**

a connection problem between vm and physical machine cannot exist unless the vm and machine are absolutely fubar

ill put it up once again

- 1) go to ms and download virtual pc 2007 sp1 (free)
- 2)) install virtual pc in physical machine and create a virtual hard disk
- 3) create an iso from xp setup (imgburn)
- 4) mount the iso in virtual pc using capture iso image in cd tab on virtual pc
- 5) fdisk , format and install xpsp3 in vpc 's virtual harddisk
- 6) open boot.ini in c:\ in vpc hard disk and add one more boot entry
- 7) in the virtual pc edit->setting->com1-> setup a named pipe \\.\pipe\debugPipe
- 8)reboot vpc and select the newly added boot entry
- 9)in the physical machine create a shortcut <path>\windbg.exe -k com1 pipe,port=\\.\pipe\debugPipe, resets=0, reconnect
- 10) double click and launch the shortcut (i e windbg waiting to reconnect)
- 11)press ctrl+break a few times in windbg or open the vpc and press ctrl+alt+prtscreen for kd connection to happen
- 12) set up symbol path in myComputer->properties->advanced -> environment variables-> new in physical machine
- 13) if your internet connection is active windbg will automatically fetch the symbols from ms as and when needed

14) that is it nothing more nothing less

[QUOTE]
iam surprised you have ntfs.pdb
[/quote]

i have ntfs pdb for several os gathered over a few years and all of it were fetched by windbg as and when it required

Code:

```
F:\>cd SYMBOLS\ntfs.pdb
```

i ll take a look at the images but there are several utilities that will show them in colored and or several other formats (0x10 /winhex / to mention a few

WaxfordSqueers

May 19th, 2013, 08:15

Quote:

*[Originally Posted by blabberer;94743]don't give me this again it is absolutely not possible that nothing could be available
i said you **DONT NEED TWO MACHINES***

Heh....be nice, after all, I sent you all those purdy peectoores. I hope you don't take me seriously with my offhand comments. I worked for years in construction/industrial environment and I'm used to mouthing off and being mouthed off at. It's all in good fun.

I have not forgotten what you said about VMs, I just haven't had time to go through the learning curve. My eyes feel like pee holes in the snow from all the reading I've been doing. They are a very unnatural colour of red, like a vampire in heat. I used Live KD just to have a look at !filecache, etc.

I was only opening files like KD to see what would happen. I got a tabbed menu with different options for connecting to a remote machine and the last tab was for local connections, That's all I was saying. Another problem is that my XP machine is not online hence a difficulty in getting the symbols directly. My XP was configured through the LAN connection to access the Net via my laptop wireless connection but despite all the blethering from Microsoft, setting up a LAN connection between XP and 7 is not that apparent. For example, setting up the XP side of an Internet connection calls for making a disk on XP and using it on the other system. There are no provisions for that on 7 and I had to resort to processes I have since forgotten, like adjusting the XP NIC to my laptop's wireless IP.

It just disappeared on me for no apparent reason and troubleshooting it became a major event.

I am completely comfortable with softice (hey Kayaker...how's it goin?) and learning windbg at this particular moment would hold me up so badly you'd not hear back from me for days, maybe weeks.

The hold up is that I am trying to learn how to get 'direct' disk access via softice. I had no idea what I was dealing with when it came to file caching and that has been a major revelation in itself. Typical Microsoft bs, trying to out-think the user. They should have a radio button prominently displayed to turn off file caching and save us some grief. I know of the function in Device Manager for hard disks where some file caching (write-behind) can be turned off but I think that refers only to one part where the system holds writes back for a while. Both the drives on my XP system are turned off for write-behind. I turned that off for good when I was single-stepping and killed my system during a BSOD. Fortunately, chkdsk recovered the disk. There are times under heavy usage when my Win 7 system just slows to a near-halt, and now I know why, not the lazy-write, but the entire file caching nonsense.

It's so convoluted it's a wonder Windoze works at all. They seem to forget this is a time-slice system where all this stuff has to be done in a fraction of a second. All it takes is an overloaded or confused file caching system to slow things right down, and there's no remedy but a reboot.

Imagine Microsoft logic trying to guess what a user is going to read next, and all the bloatware that goes into doing that. Every time I open File Explorer, it anticipates what I want to do and opens to a directory I have never used. That's partly why I never use the piece of garbage, using a third party file manager that actually allows me to open two panes at the same time. With Explorer, you still have to open another instance and I don't bother. Same with IE Explorer....garbage. I used Firefox, which is twice the browser and free, Even Opera leaves IE in the dust.

Anyway, I now feel far more confident about exploring the MFT via softice, or windbg, if I get it going. Each day I learn a bit more about the spider approach. And I keep you full of life raving at me. 🐸

I have discovered some functions that relate to MFT metadata, like CcPinRead. Of course, those are cache related funcs. There's also IoCallDriver and CcCopyRead, which I encountered the other night while traipsing through the dark code woods. Also, I learned that NtReadFile should be called closer to the MFT read. As I told you, it was breaking all over the place the other night.

I will look at your VM setup tomorrow. I already have VMWare setup with a Windows XP VHD but I don't think it's SP3. Softice did not like SP3 but deroko claims to have it running with SP3 on a VM. He's in the softice genius class, however, like Kayaker.

Quote:

[Originally Posted by blabberer;94743]i ll take a look at the images but there are several utilities that will show them in colored and or several other formats (0x10 /winhex / to mention a few

The point of the images is not the nice colours, it's the correlation between them and the template descriptions. A heck of a lot of work has gone into deciphering the file attributes and it makes life a whole lot easier when trying to make head or tail about what is happening. The template descriptors go into great detail but unfortunately only for files beginning with 'FILE'. There are other that begin with 'INDX' I'd like to get into.

According to the reading I was doing tonight, NTFS uses an LCN - VCN compression and I have seen reference to that before. As you know, an LCN is the cluster distance from the start of volume and the VCN is the offset from the start of file. It also uses record numbers for files that are not all that apparent because they are combined with another number into a 64-bit number. The MSBs of the number represents the number of times the MFT space has been used and the LSBs is the file number.

blabberer

May 19th, 2013, 15:27

Quote:

Another problem is that my XP machine is not online hence a difficulty in getting the symbols directly.

no need for vm to be online is your win7 machine fine ? if yes set xp vm in win7 get symbols in win7 for xp run windbg in win7 kernel debugging xp vm

hey im not forcing you to setup windbg and learn it i understand your reluctance i myself dont use ida much though the whole fscking world loves it it is an opinion from me that is all

Quote:

There are other that begin with 'INDX' I'd like to get into.

here you go atleast how indx looks like in cache 😊

Code:

```
kd> bl
0 e 8056d719 0001 (0001) nt!CcMapData+0xf2 ".if ( @@masm(dwo (@ecx & 0xffffffff)) != 'XDNI' ) {gc}"
```

easy aint it 😊
come on keep me raving

WaxfordSqueers

May 20th, 2013, 00:34

Quote:

[Originally Posted by blabberer;94745]hey im not forcing you to setup windbg and learn it i understand your reluctance i myself dont use ida much though the whole fscking world loves it it is an opinion from me that is all

I have already indicated to you that I want to learn windbg and I have been thinking that long before starting this thread. Right now, my focus is on doing something with my external drive so I can use it again. All I have to do is reformat it and the case is closed. I am taking an interest in NTFS to see if I can repair a broken link in the MFT chain and make my directories and files re-appear.

When the Comodo backup utility began to write to my external drive in a clone process, it only lasted a few seconds, However, it managed to overwrite the existing Boot sector. It may have gone a lot further and overwritten critical files placed near the start of disk, but the drive is not bootable and that should not matter. It's for storage only. I don't know if it got to the MFT but it created a new partition, relegating my data to unallocated space. I have no idea as of yet how much damage it has done to my original root directory, and if it has damaged it, whether I can re-route the existing root directory to another node on the B-tree.

I was pretty naive about NTFS, and too impatient to get results. I should have sat on it till I understand what I was dealing with but got impatient and used a tool to remove the partition created by Comodo. I know now to immediately make an image of the damaged disk and store it away.

I can see directories I recognize in the unallocated data based on file headers but I still don't know how to traverse the B-tree. I think most of it is still intact. I learned last night that once NTFS open a file, it no longer traverses the B-tree. It uses the file record number to find it directly.

I don't know how NTFS implements the B-tree structure of it's directory structure, but if only a few nodes are damaged, can I repair them? First I have to learn how the system gets from the root to the next B-tree node and try to trace it through. Understanding B-tree theory should not be necessary so long as I see a few traversals of the nodes. I can do that with a dry read, using a disk editor like Active@ or I can trace it with softice, or windbg, if I can learn it fast enough.

Now might be a good time to learn windbg as I use softice in parallel. I have no idea whether their drivers will clash, but that's part of the fun in reversing, finding out.

Quote:

[Originally Posted by blabberer;94745]here you go atleast how indx looks like in cache 😊

It looks interesting but can you explain how to find a file using that info? for example:

mftFileReferenceOfParent : 0x50000`00000005 obviously reference to inode #5 on the MFT metadata file structure. That would be the root directory, so we're in the ballpark. The other records you have generated, like \$BadClust, \$Boot, etc, are of no help. So you need to read the \$Root entry and figure out how to get to the first file or directory from there.

Apparently files have metadata that point back to the parent, but that metadata is in a data stream that sits between the root directory and file or directory. I am guessing that the root has a pointer to a stream (data run) but I have been unable to decipher it yet.

Quote:

[Originally Posted by blabberer;94745]easy aint it 😊 come on keep me raving

The raving shot was a humorous reference to your use of block capitals to draw my attention to information for which I had full awareness. I got a grin out of it but wondered at the emotion and frustration driving you.

No...it's not easy. It's easy to print out configuration data but it's far from easy to interpret it and use it to find things.

Show me how to get from inode 5 to the first directory entry, then I'll agree that it's easy.

WaxfordSqueers

May 20th, 2013, 11:31

Quote:

[Originally Posted by blabberer;94743]6) open boot.ini in c:\ in vpc hard disk and add one more boot entry

Another insanely late night.

I tried my vmware setup first but I have d/l'd and installed VM 2007 on my laptop. I was checking to see if I could import/mod a vhd disk from VMWare to VP 2007. Some say you can, but I'm getting ahead of myself.

Why did I try VMWare first, because as George Mallory said when he was asked why he climbed Everest, because it's there? And because from past experience it has more bells and whistles. I had it setup with XP and softice. Better check to see if it has sp3 or not.

Got seriously hung up on a few issues but just got it going. One problem was with the debug statement in boot.ini. From everything I had read, it was supposed to look like

debug" /debugport=com1 /baudrate=115200 at the end of the other long string for debugging. However it would not work. Also, in the VM, for the serial port, the choices made little sense. You had to choose between server/client and The other end is a virtual machine/The other end is an application. I finally settled on client/application.

I was also thrown by the windbg shortcut, about where to put the extra command line stuff. It goes on the end of the target path/filename but you have to observe the quotation marks around path/filename. You put double quotes around the path/filename and tack the rest onto the end, i.e.

"<path>\windbg.exe" -k com pipe,port=\\.\pipe\debugPipe,reset=0,reconnect

If it's not exactly that, windbg whines and whines, giving no clue as to what is wrong.

Your clue about ctrl-break while in the windbg window was fortuitous because someone else suggested tacking /break onto the end of the debug string in boot.ini. All that does is freeze the boot.

Furthermore, I had to add something to the debug line. Instead of:

```
debug" /debugport=com1 /baudrate=115200, I had to add another COM1 as in:
```

```
debug COM1" /debugport=com1 /baudrate=115200
```

I have no idea why, but I saw it on an internet article and it worked.

Quote:

*[Originally Posted by blabberer;94743]12) set up symbol path in myComputer->properties->advanced -> environment variables-> new in physical machine
13) if your internet connection is active windbg will automatically fetch the symbols from ms as and when needed
14) that is it nothing more nothing less*

I've had it for the night. I found my Internet connection working on my XP, thru my laptop, for whatever reason. Windbg already had my local symbol storage and the MDSL address. It might have changed, I'd better check.

windbg loaded and it's sitting at

```
nt!RtlpBreakWithStatusInstruction  
804e3592 cc int 3
```

I entered !filecache and it's off processing that. It's taking its time. livekd had it back pretty quick.

In fact, it never did come back. I gave it a ctrl-break and it returned.

Under breakpoints, I don't see the equivalent of a softice BMSG, where you can enter a handle and a windows message. Is such an animal available on windbg?

Putting it another way, how could I load notepad in the VM and have windbg break early enough in the load procedure? With softice, I set a BMSG hwnd 203 for the LButtonDown message (203) and it broke right away inside my file manager.

WaxfordSqueers

May 21st, 2013, 00:58

Quote:

[Originally Posted by blabberer;94743]2)) install virtual pc

I have a lot to say about Microsoft virtual PC and none of it is good. So, I'll say no more than that I will not be using it further. I just spent about 12 fruitless hours trying to configure it and I must say that it rates among the most non-functional and poorly designed pieces of software I have ever encountered. It's flexibility and ability to adapt to varying user-mode circumstances is essentially nil.

The thing I hate about myself at times is my tendency to beat a dead horse. When it is abundantly clear that a tool is not doing the job, I have a terrible habit of trying to make a silk purse out of a sow's ear.

If there's a redeeming value in this it's that I have learned a lot about how VMs are not supposed to work. ☹

blabberer

May 21st, 2013, 20:25

Quote:

Show me how to get from inode 5 to the first directory entry, then I'll agree that it's easy.

i will show how to extract an arbitrary file from a nonresident datarun from FILE0 signature

2766

WaxfordSqueers

May 21st, 2013, 21:25

Quote:

[Originally Posted by blabberer;94757]i will show how to extract an arbitrary file from a nonresident datarun from FILE0 signature

Hexcellent (digital for excellent) or 48 65 78 65 6C 6C 65 6E 74 00 (zero-terminated).

Appreciate the work.

I did trace one file a few days ago using your method but not from the root directory. It too came from a non-resident file. I would like to tie that into the root, to see where it finds the first directory.

I am theorizing that even if the B-tree has been lopped off near the top that I might be able to splice the root dir to the B-tree using a pointer in the root. However, that might mess up the algorithm used to read the B-tree. The aim is not to repair the system per se, but to enable seeing a directory structure that has been lost due to a breach in the B-tree. As it stands, utilities that do such recoveries normally rename the files they recover. It would be nice just to see them listed on the disk under their proper directories.

The Active@ Disk Editor allows writing to the disk but I know any good hex-editor allows that. I used to do it with Norton but unfortunately it does not work on USB.

In one of my MFTs, I have seen several metafiles listed in the root metafile, like \$MFT, etc, along with other typical root files. I have read that it is possible, using some NTFS utilities to see them in the root directory of a volume. Apparently it used to be possible to see them using a dir command from a DOS prompt.

I am forcing myself out the door for a long walk but I'll have a closer look when I get back. I am also going out of town for a week or so, so if you don't see further progress that's what happened.

BTW...after sleeping for nearly 24 hours, after being up nearly as long, I am not so steamed about Microsoft VM. Nevertheless, I have a VM that works a heck of a lot better, and which is quite flexible. I have also garnered more info on how to setup on a laptop and use the desktop as a client. I did not know that I could simply set a port between 49152 and 65533 and use an Ethernet connection. Live and learn.

Thanks again.

Kayaker

May 22nd, 2013, 00:42

Interesting stuff you guys have gotten into. I've been trying to follow it, reading a bit, downloaded a few of the NTFS directory listing apps and traced them in Softice (yes, still my GOTO Wax) to see how they access the data, etc.

Still wallowing in the shallow end for sure, but I had to say that I was pleasantly surprised to discover one of those NTFS based file listing apps, that I don't think has been mentioned yet, but is blazingly fast at doing file searches. Never again will I use the Windows file search function. Thanks for that enlightenment at least.

NTFS Direct File Find
<http://ndff.hotbox.ru/en/index.html>

Waxford, if you're setting up the VMWare/Softice thing, you probably already know about editing the .vmx file. Also discussed in a few forum threads here.

```
vmmouse.present = FALSE
svga.maxFullscreenRefreshTick = 5
```

http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=965

I recently re-set things up successfully in Win7/x64 with Softice running with XP3 in VMWare. You should be able to do that too. One note that may help - make sure you set the VM to use only 1 processor - no multiple cores, no Hyperthreading.

Also, re the remote Windbg/VMWare setup, at least in a single box host/guest setup, I highly recommend incorporating VirtualKD. Not only is it faster, it does all the configuration for you.

<http://virtualkd.sysprogs.org/>

WaxfordSqueers

May 22nd, 2013, 06:04

Quote:

[Originally Posted by Kayaker;94761]Interesting stuff you guys have gotten into. I've been trying to follow it, reading a bit, downloaded a few of the NTFS directory listing apps and traced them in Softice (yes, still my GOTO Wax) to see how they access the data, etc.

Interesting reply Kayaker....thanks. I am still very comfortable with softice but I have been reading a lot on windbg and it has obvious advantages for listing internal stuff that only microsoft knows about. Of course, I have only used ice minimally and don't know it to the level you do.

What I like about softice is seeing everything on one screen, like registers, data dumps, code, etc. I like the black background, it sucks me into the debugging space.

The breakpoint instructions are surprisingly similar but I have still to find the equivalent in windbg of BMSG, one of my favourites in ice for getting an app to stop on hwns I get from SPYXX.

Quote:

[Originally Posted by Kayaker;94761]I was pleasantly surprised to discover one of those NTFS based file listing apps, that I don't think has been mentioned yet

I'll have to check that out. I was using the search function in Active@ Disk Editor and it is quite fast on record header like FILE but terminally slow on string searches like CBU (extension for a Microsoft backup). I calculated at the rate it was going on a 500 gig drive that it could take me months for it to finish. 😊

Quote:

[Originally Posted by Kayaker;94761]Waxford, if you're setting up the VMWare/Softice thing, you probably already know about editing the .vmx file.

I did know about it but would likely have forgotten. Thanks for reminder. Your suggestion of setting up softice in a VM on win7 is my next project. I had success setting up windbg on an XP with a VM and windbg running on the host, but it sure ain't intuitive. No matter how many good tutes you get on it, or good advice from Blabs, there is always a wrinkle peculiar to your own setup it doesn't like. Of course, softice setup can be like that too.

Quote:

[Originally Posted by Kayaker;94761]I recently re-set things up successfully in Win7/x64 with Softice running with XP3 in VMWare. You should be able to do that too. One note that may help - make sure you set the VM to use only 1 processor - no multiple cores, no Hyperthreading.

Does VMware have to be 64 bit as well? I loaded debugging tools for windows (x86) no problem and I have windbg running on it but doing nothing.

I was checking out the tab under KD where it claims you can use a network to connect. After reading on it, they require certified NIC cards and Win Vista on the client. Funny enough, the host can be XP. Also, when you set up the debug environment in boot.ini, the Debug command is apparently geared to COM ports. I am not sure how they can claim that and use a network.

BTW, while reading on this I noticed there are USB to serial converters available if you are trying to run a host on a laptop to a client on a desktop. You still need a null-modem cable and the one Microsoft describes is one with full handshaking. Cheaper null modem cables come with only the receive/send pairs connected and a ground. Microsoft calls for one with most of the 9 other pins connected, like RTS and CTS. That would make a huge difference if someone just picked up the cheaper version of the cable and it did not work.

Correction: The USB connector has only a transmit and a receive available in it's 4 connector tabs, the other two being +5 volts and ground. So, you have to watch what kind of USB - serial connector you buy. A basic one could give you basic functionality without full handshaking and windbg require the latter.

This converter uses a chip in the adapter to do that.

http://www.ftdichip.com/Support/Documents/DataSheets/Cables/DS_Chipe-X.pdf

Handshaking refers to the communication between peers where they ask each other how the other is doing. RTS means request to send (rather polite) and the other replies CTS = clear to send. There is also DTR = data terminal ready, and so forth. The cheaper null modem cables have none of that and if the software is using it, things jam up. A word to the wise, when buying null modem cables for microsoft communications make sure it states 'full handshaking'. Or, buy the wire, connectors and solder one up.

Sometimes it's just as expensive by the time you get all the gear, depending on where you buy it. Of course, if you buy the 9-pin D-connectors, you can repair them whereas store bought cables are usually molded. It wont be the first time I took a sharp knife to a molded assembly to get at the pins, then taped it back together.

<http://msdn.microsoft.com/en-us/library/windows/hardware/ff556867%28v=vs.85%29.aspx>

see the hyperlink to null-modem cable.

Also,

http://www.lammertbies.nl/comm/info/RS-232_null_modem.html

explains RS-232 cables pretty well (a null modem cable is a form of RS-232).

Thanks for the VirtualKD heads up.

Kayaker

May 22nd, 2013, 20:11

Quote:

Does VMware have to be 64 bit as well?

32bit VMWare works fine.

Quote:

The breakpoint instructions are surprisingly similar but I have still to find the equivalent in windbg of BMSG, one of my favourites in ice for getting an app to stop on hwnds I get from SPYXX.

Fortunately it's not much more involved, simply set a conditional breakpoint on the message handler for the Msg of interest. Spy++ should give you all the information you need including the Msg ID.

Say you want to break on WM_MENUSELECT (0x11f) when a menu item is selected in notepad. Spy++ says the WndProc of the main window is at:

Code:

```
:01003429 _NPWndProc@16 proc near
:01003429
:01003429 Msg = dword ptr 0Ch
```

The Windbg conditional breakpoint would be:

bp 1003431 "j (poi(ebp+0x0c)==0x11f) "; 'gc' "

Who needs old creaky cranky Softice? (oops, did I say that? 🤪)

WaxfordSqueers

May 23rd, 2013, 01:10

I gave you a bit of a bum steer with respect to USB - Serial adapters. The USB connector has only a transmit and a receive available in it's 4 connector tabs, the other two being +5 volts and ground. So, you have to watch what kind of USB - serial connector you buy. A basic one could give you basic functionality without full handshaking and windbg require the latter.

This converter uses a chip in the adapter to do that.

http://www.ftdichip.com/Support/Documents/DataSheets/Cables/DS_Chipi-X.pdf

Quote:

bp 1003431 "j (poi(ebp+0x0c)==0x11f) "; 'gc' "

Just learning to read this language. From what I know, the bp @ 1003431 is subjected to a j operator which is the windbg if/else statement. So the statement seems to be saying: break if the value in (ebp+c) = 0x11f is true. I think your " after the j operator should have a space, should it not, as in ' ', meaning true/false in windbg jargon.

I am not too clear on the poi operator yet. I know it is used to dereference a pointer. In this case, in MASM, (ebp+0x0c) is normally an address normally found with square brackets as in [ebp+c]) but in C++ it is a value. This is what I mean about Microsoft, they go out of their way to be vague. The @@ operator also dereferences pointers so what is the difference between poi and @@? It also appears as if you have to find where the message value is in reference to the EBP base, whereas in softice you simply supply the message number and the window handle.

All in all, in some cases, windbg goes to a lot of trouble to obfuscate a simple issue. As you know, in softice, you give a simple BMSG hwnd msg# and you are done. You can add a conditional statement if you want but you don't have to.

BTW...anyone reading this who is new to softice should realize, at least in my setup, that softice does not accept the message name anymore, it has to be the message number. At one time you could write BMSG hwnd WM_COMMAND, but now you have to write BMSG hwnd 111.

blabberer

May 23rd, 2013, 01:24

oops yep you said it 🤪

windbg notepad

Code:

```
0:000> bu user32!TranslateMessage ".if ((poi(poi(esp+4)+4))==0x201) { .printf \"weeee waxford clicked me ooohh\\n\" };gc\"
0:000> g
ModLoad: 5cb70000-5cb96000 C:\WINDOWS\system32\ShimEng.dll
```

WaxfordSqueers

May 23rd, 2013, 01:35

Quote:

[Originally Posted by blabberer;94768]oops yep you said it 🤪

windbg notepad

Code:

```
weeee waxford clicked me ooohh
weeee waxford clicked me ooohh
weeee waxford clicked me ooohh
```

Sounds like Waxford is a bit of a pervert. 🤪

blabberer

May 23rd, 2013, 01:37

```
poi(R#) == <size> ptr <seg>:[R#] // dword ptr ds:[eax] or [eax]
poi(poi(R#)) = " " //[eax] etc
```

@ is used to tell we are referring a register and not a symbol and cuts down symbol searching time

so always make it a habit to use

? poi(@eax+0x4) so that results emerge faster

"think typo and hitting enter and waiting for eternity till ~1000 symbol files are downloaded /parsed and /negated"

Code:

```
0:001> ? poi(@esx)
Bad register error at '@esx'
0:001> ? poi(esx)
```

WaxfordSqueers

May 23rd, 2013, 01:47

```
bu user32!TranslateMessage ".if ((poi(poi(esp+4)+4))==0x201) { .printf \"weeee waxford clicked me ooohh\\n\" };gc"
```

Would you be kind enough, kind sir, to break this statement down for me?

One poi I can deal with but a poi poi sounds like some kind of Hawaiian meal.

First, you loaded notepad directly with a windbg notepad.

It seems the inner poi dereference esp+c to a value, then I get lost. What does the outer poi do?

You seem to be setting a general break on TranslateMessage with WM_LBUTTONDOWN (WMSG 0x201), and if it's true you print that Waxford is a pervert. So '.if' is equivalent to the j operator, is that right? I have not touched on dot operators yet but it seems you can replace the '.' with { }. Does that mean '.' applies only to true/false and the { } applies to general statements?

WaxfordSqueers

May 23rd, 2013, 01:58

Quote:

```
[Originally Posted by blabberer;94770]poi(R#) == <size> ptr <seg>:[R#] // dword ptr ds:[eax] or [eax]
```

So, poi (eax) is equivalent to [eax], meaning the value at the address pointed to by EAX???

I have never encountered [[eax]]. Is that like finding the value at the EAX address and if it's a ptr, finding the value at that address...like a nested pointer?

For example, in softice, I sometimes see a ptr in EAX and dump it, to find another ptr, which I dump as well. If I find a value at the second pointer address, that would be the same value as a double poi???

I am used to @ referencing a pointer and I understood @@ to mean the value at the address pointed to. I get your point about symbols lookup time, which is invaluable, but I need to read more.

Kayaker

May 23rd, 2013, 02:02

Quote:

```
[Originally Posted by WaxfordSqueers;94767]I think your " after the j operator should have a space, should it not, as in ' ', meaning true/false in windbg jargon.
```

That is actually 2 separate single quotation marks ("), albeit without a space between them, not a double quotation mark (""), it probably just looks that way in the browser. Works as a copy/paste breakpoint in any case.

Quote:

```
All in all, in some cases, windbg goes to a lot of trouble to obfuscate a simple issue. As you know, in softice, you give a simple BMSG hwnd msg# and you are done. You can add a conditional statement if you want but you don't have to.
```

That is very true, it's better in Softice because you don't have to find the WndProc, Softice finds it for you via the BMSG command.

Quote:

```
BTW...anyone reading this who is new to softice should realize, at least in my setup, that softice does not accept the message name anymore, it has to be the message number. At one time you could write BMSG hwnd WM_COMMAND, but now you have to write BMSG hwnd 111.
```

Works for me:

Code:

```
01) BMSG 11010C WM_MENUSELECT
```

```
Break due to BP 01: BMSG 11010C WM_MENUSELECT (ET= 2.18 seconds)
```

WaxfordSqueers

May 23rd, 2013, 02:14

Quote:

[Originally Posted by Kayaker;94773]Works for me:

Code:

```
01) BMSG 11010C WM_MENUSELECT
```

Break due to BP 01: BMSG 11010C WM_MENUSELECT (ET=3 18 seconds)

Weird. It stopped working on my version. I have to look up the message number and enter it in hex. Oddly, once I have entered it in hex, and do a BL, it shows the message name.

Maybe it's just messing with me. Softice is a very intelligent app. 🤖

blabberer

May 23rd, 2013, 02:47

break this statement down

sure

Code:

WaxfordSqueers

May 23rd, 2013, 10:37

Quote:

[Originally Posted by blabberer;94775]TranslateMessage takes one parameter ie a pointer to MSG structure so when you have broken on TranslateMessage esp+4 will point ot MSG structure ie [esp+4] == addrof (lpMsg) would be something like 0x12345678

Appreciate all the info, I am prepping for my trip and may not get a chance to focus on what you are saying.

Glad I am not the only one who is dyslexic. I keep reversing words when I am typing, but you do ot (to) while i tend to do wrok (work). And you may have noticed my i (I), which I can't seem to control no matter how much I focus. Don't know if it's true dyslexia or something peculiar to typing on a keyboard. 😊

I have encountered TranslateMessage a lot while tracing code, usually in the loop where Win processes messages. One thing I have never been able to figure out is how to break out of the message loop at times. Sometimes the code leads through the TranslateMessage area and back into the app, and other times it gets into an endless loop with PeekMessage, etc.

Another area where that happens is in WaitForSingleObject and its brethern, especially in the kernel. I need to get deeper into signalling, mutexes and so on and I have one app that uses it extensively. I don't encounter these problems enough to get focused on them. I have no gotten a feel for what it is up to however since it seems to spawn another thread of itself along the way.

Indy

May 25th, 2013, 08:25

Too lazy to read that crap. Learn math part. I for example long time did not used the km debugger. Beginning study the architecture. At the kernel of not learning on the narvahi canvas" 🤖

WaxfordSqueers

July 7th, 2013, 01:59

Took off on a trip for a while but I have been attacking the same problem about tracking a file read process to the NTFS or MFT level.

It appears that Notepad makes judicious use of shell32.dll to do it's path work. I would think that would be simple in itself, but it's horribly convoluted the way paths get checked and double checked. As if that's not bad enough, they have an SHItemID list which seems to parallel the MFT attributes, creating a sort of Btree listing that can be broken down into smaller directory/file lists. They have procedures in shell32 and shlwapi to deal with the lists.

Tell the truth, did anyone know shlwapi means shell lightweight API, and works hand in hand with shell32. Has anyone heard of shell32 before. 😊

It gets tricky during tracing when working with file system drivers because, as I recall, the filter drivers get stacked. So, I ended up tracing through a driver from Silicon Image for my CD/DVD writer, then there's one from sr.sys for system restore, then fltmgr.sys itself, while en route to Nt_ReadFile.

I am still encountering the same old problem....the file cache. Notepad is getting loaded in there somehow rather than being read directly from the MFT tables. Even the MFT attributes are in the file cache. It's either that or I have bypassed a piece of code where that is done. Blabberer will be happy to know I have been using Windbg to look at the file cache, and further breaking it down using !ca, !fileobj and db dumps to look at the file remnants to see what is there.

I have deleted the paging file in XP but have not tried tracing since I did. I can't imagine where else they'd hide copies of the file cache.

Next thing is to find an app that will choke the file cache, or purge it. I saw mention of a cache app from sysinternals but have yet to track it down. The Net connection on my desktop is wonky.

There is a possibility that I have gone too far into the shell path/file processing and simply ran past the point where they check the MFT, before checking the cache. Shell32 sure can be tedious to trace through with all the stuff they do to paths. It's unbelievable how many times they check paths and file attributes before finally doing something.

Anyway, I have traced through most of the pertinent parts and I am making some progress, so I have indicated that in this note to let anyone interested know i am still working in the thread.

Kayaker

August 30th, 2013, 23:58

Quote:

[Originally Posted by WaxfordSqueers;94722] The ref to Ntfs!NtfsReadMftRecord is interesting. I'll have to sift through ntoskrnl symbols and see if there are other NTF references. Maybe even the HAL symbols.

What I am trying to do is approach the \$MFT file from its entry point and trace through it's front end so I can see where it goes, especially from the \$ROOT file.

Each of the first set of files are metafiles and begin with the header 'FILE'. The very first entry in an MFT file is FILE, like MZ is the first entry in an exe file. From there onward, there are attributes at various offsets, and each attribute begins with a number. It's like a PE header and I want to learn to read it.

Trouble is, as I understand it, once the disk has been accessed, NTFS reads the files and moves info to another location. Also, a file like Notepad, if used regularly, will have a prefetch record and the MFT will likely be bypassed. So I need to catch the NTFS system in the act of initializing the MFT. I want to see what it's initial entry point is into the table, and why.

Hey Wax,

I was muddling through this a bit again, trying to figure out where you're at tracing the connection from CreateFile -> MFT, or W(hatever)TF you're doing. This may be of no use, but I was playing with yet another NTFS utility, nfi.exe (Windows NT File System (NTFS) File Sector Information Utility), part of this package:

<http://support.microsoft.com/kb/253066/en-us>

Here's a search for notepad.exe

Code:

```
c:\>nfi c:\windows\system32\notepad.exe

NTFS File Sector Information Utility.

Copyright (C) Microsoft Corporation 1999. All rights reserved.
```

What it gives is the MFT # for a given file, i.e. (**file number 15622**)
Using NTFSWalker confirmed that's the actual sequential MFT record for notepad.

I just thought that if you ever get to a point, maybe in NtfsReadMftRecord, you could use the known MFT record number as a breakpoint qualifier for the target file you're chasing. Just a random thought.

Oh, and if you deleted the prefetch for a file, would it go through the initialization steps of accessing the MFT that you're hoping to catch?

blabberer

August 31st, 2013, 01:28

well i was going to post the exact same sentiments but refrained from posting it as i thought you might on to a bigger fish that i couldn't visualize reading your post

if you wish to catch some thing about mft etc you need to sit on the other side of the fence
shell is a muddle and fishes don't thrive there except for an occasional tortoise

ShellExecute finally crosses the um -> km boundary via ntddl!NtCreateProcessEx and that can be verified in few steps like below
you don't have to trace its piddle muddle

lets break on ShellExecuteA and swim along till the net shall we ?

Code:

```
0:000> $ no point using a bulldohhzzer to flatten a mole hill

0:000> bp SHELL32!ShellExecuteA

0:000> g
```

so sift the cosmic blubb for secrets of universe no point looking at all the thugs with piddles trying to muddle you out

Kayaker

August 31st, 2013, 03:05

Interesting journey, though I think you may have been watching too many movies lately there amigo.

W_x might be wanting to watch the whole story, from start to finish, but you could also jump to the ending rather than piddling in the middle.

Break on NtfsReadFileRecord or NtfsReadMftRecord and see what it tells you.

```
/* non-Windbg */
Disassemble ntfs.sys in IDA, with PDB. Rebase to 0
Get offset of NtfsReadFileRecord
In Softice get base address with 'driver ntfs', add offset and set breakpoint
Do stuff
```

Code:

```
00026B5C _NtfsReadFileRecord@28
00026B5C
00026B5C arg_0 = dword ptr 8
```

http://read.pudn.com/downloads171/sourcecode/windows/vxd/794585/ntfs/mftsup.c_.htm

Code:

```
VOID
NtfsReadFileRecord (
    IN PIRP_CONTEXT IrpContext
```

blabberer

August 31st, 2013, 14:15

Quote:

Break on NtfsReadFileRecord or NtfsReadMftRecord and see what it tells you.

post #20 in this thread shows a script which prints out the mft record number in windbg with the second function NtfsReadMftRecord

<http://www.woodmann.com/forum/showthread.php?15188-Softice-and-createfile&p=94718#post94718>

nope no movies those were from books of robert ludlum frederik forsyth genre read ages ago

WaxfordSqueers

August 31st, 2013, 15:03

Quote:

[Originally Posted by Kayaker;95344]I was muddling through this a bit again, trying to figure out where you're at tracing the connection from CreateFile -> MFT

Thanks for that, K., there is some interesting stuff in your reply about record numbers (nfi.exe) and MFT access. Let me give you a summary of what I am trying to accomplish and why I am where I am in the tracing.

1)I stupidly overwrote part of the file system on my external drive, which I use for storage. The original MFT and MftMirr files are still there and the damage seems to have separated the main file archive from the MFT metafiles.

2)I am looking for a way to connect the MFT data to the lost files. The record number supplied in your utility is interesting but finding the reference to that number in the MFT is compounded by the way it is listed. They use an offset from the beginning of the partition and an offset from the beginning of the MFT to find the file record. The offsets are combined in a way that is not easily read and there is not much info on the Net as to how it should be read...only vague references.

Even Microsoft admits the process of tracing the MFT is not trivial.

I am hoping, that if I find the lost files, that record number will be there at the beginning of the lost record. I have an app that does a search through raw data on disk and I may plug in the record number to see what it finds. I am reasoning that the NTFS system relates the MFT with the stored files so it can verify that it has the right file. So far, I have been unable to find raw data with any such reference.

I used another app to retrieve most of the files I lost, but it does it by recognizing file signatures then recovering the file data. It does not use the MFT as far as I can see. Actually, the lost files are not lost, I have them backed up elsewhere. I am doing this as a project to see if I can reconstruct a file/directory system from a damaged MFT.

3)I glommed onto the idea of watching NTFS unravel the mystery for me and I began with a bpx on Createfile. It lead me into NTFS.sys and even HAL via driver filters, which was interesting, but when the file was accessed, it had already been loaded into memory. That is, the MZ header was already there.

That seems to suggest that CreateFile, when called initially, retrieves a handle to a file that has already been found on disk by other processes and loaded into memory. From what I have seen, Createfile does not initiate the process of dealing with the MFT.

4)I decided to back up to the mouse capture when I double-clicked the app and quickly trace through to the CreateFile process. I had no idea how much path parsing and object creation there was in-between, involving shell 32, Shlwapi and Ole32. That's where the PIDs and IDLs come into play. I did see a reference to the MFT in some PIDL theory and that encouraged me.

Since IDLs (Item ID lists) are part of the shell namespace that underlies the file and directory system, it makes sense to me intuitively that the MFT must somehow be tied into that namespace process. That's what I am looking for.

<http://msdn.microsoft.com/en-us/library/windows/desktop/cc144093%28v=vs.85%29.aspx>

I became so immersed in the theory of PIDs and IDLs, with SHITEMID lists that it became a project in itself.

Your suggestion makes eminent sense, to bpx on an MFT function, and I will try that. Something stopped me in the past and I think it had something to do with failing to load symbols in softice from NTFS.sys. The NTFS.sys nms created by symbol loader seemed to be missing those functions and I created an nms using IDA2ICE in IDA, but it did not work for some reason. I'll try it again now that I am loaded in a VM.

Quote:

[Originally Posted by Kayaker;95344]Here's a search for notepad.exe

There is some interesting stuff in here. I'll need to take a closer look to see if I can relate the logical sectors to the offsets stored in the MFT. It might even be interesting to follow NFI.exe with softice or windbg to see where it finds its info.

Quote:

[Originally Posted by Kayaker;95344]I just thought that if you ever get to a point, maybe in NtfsReadMftRecord, you could use the known MFT record number as a breakpoint qualifier for the target file you're chasing. Just a random thought.

Oh, and if you deleted the prefetch for a file, would it go through the initialization steps of accessing the MFT that you're hoping to catch?

Good idea. I deleted the prefetch and turned it off so that Windows would not use shortcuts to bypass the MFT. Maybe my reasoning is awry.

WaxfordSqueers

August 31st, 2013, 17:47

Quote:

[Originally Posted by Kayaker;95346]Break on NtfsReadFileRecord or NtfsReadMftRecord and see what it tells you.

I edited some of your code to keep the post shorter...

Code:

```
00026B5C _NtfsReadFileRecord@28
```

```
00026B5C
```

```
00026B5C arg_0 = dword ptr 8
```

Note the first call 'call _NtfsFindCachedFileRecord@16'

It checks the filecache to see if the file is already cached. I have not done this yet but I am wondering if I can jump over that call then edit memory to make it look as if it did not find a file in the cache. That might force it to read from the MFT.

Anyway...I still have to find that code in relation to my project.

Just rechecked your code and apparently I can force it to read the MFT. I omitted this important step from your code:

```
00026B7D jnz loc_274C7
```

That follows the cache read and a test al,al. If the file is in the cache, it jumps elsewhere, if not, it goes on to read the MFT.

I am debating whether to go for a long walk or carry on. I really need the exercise.

Kayaker

August 31st, 2013, 19:47

I was hoping you'd pick up on that NtfsFindCachedFileRecord call, because it seemed to fit what you were saying about the system checking the file cache first. I got quite a few breaks when I set a bp on the proc, though I didn't check the what/where/why of them.

WaxfordSqueers

September 1st, 2013, 00:37

Quote:

[Originally Posted by Kayaker;95352]I was hoping you'd pick up on that NtfsFindCachedFileRecord call, because it seemed to fit what you were saying about the system checking the file cache first.

I had a fair amount of success once I got things ironed out with ntfs.sys and it's nms file. I must have made a mistake when I loaded ntfs.sys in IDA initially because I could not find any of the references you posted with regard to NtfsReadFileRecord. The nms file created by symbol loader had no symbols at all. I did a fresh load of ntfs.sys in IDA and this time it came up with all the functions. I made an NMS file using IDA2ICE that is nearly a meg long and ice whined about the date stamp. It claimed it is newer than the actual module...no kidding?

I found your function after loading the ntfs nms file into ice but found that you have to enter it exactly as listed. ie. bpx _NtfsReadFileRecord@28. I had already double-clicked the app in the file manager using a hwnd from SPYXX and was sitting in the file manager context in ice. When I set the bpx and hit F5 it broke right away in the code you posted.

I think I was wrong in my last post about bypassing the cache check by redirecting the jump following the test al, al. I think you actually need to take the jump after the cache check otherwise it goes to NtfsReadMFTRecord. Anyway, the test ax, ax did not set off the jump and it filtered through to NtfsReadMFTRecord.

After that there was a call to NtfsMapStream@28 which I need to research. They call any of the MFT data a stream and you can have the default stream, which is unnamed, or you can have several named streams for the same data.

Immediately following is a call to Ntoskrnl!CcMapData which maps a byte range of a cached file to a buffer. That's why I think I took a wrong turn unless it's just checking to see if the file is cached or not.

Here's where the fun begins. When it returns from that call, an address is loaded in ESI that contains one of the metadata sections from the MFT. The data begins with the signature 'FILE' and later in the section, the loaded file name appears. I am presuming the address, which is 23:C3764400 is the actual address on disk of the MFT section. It may, however, be the cache address.

Then disaster struck. My VM input froze again, both mouse and keyboard. It's frustrating. I could access the VM controls bar to shut the VM down but nothing I did would recover the input to windows on the guest. I could not access softice, the windows start button, or anything on the guest desktop. I could access the host OK and the VM tool bar to shut the VM down.

If I hit ctrl-alt-esc, the softice mouse appears briefly then disappears. The cursor blinks but it is stuck in the code window.

I managed to watch ntfs.sys begin to parse the metafile. It took ESI+6, which is 3, did an shl, 9 to get 0x400 then compared it to that value (0x400) passed as a parameter in one of the calls. It checked a couple of other values so I need to go back and find out which metafile is being parsed and what each member means.

Another point of interest. I know from studying the MFT that the attributes beginnng with 'FILE' are 0x400 apart. So I scrolled down to the next one in softice and the file listed was notepad. I have simplified matters this time by creating a directory aaa right under the C:\ directory. Therefore, I know I am in the right directory because the only two files in it are the file I loaded and notepad. I may need to back up to find the root.

I was hoping to trace right through ntfs.sys to see where it returned to. That is, what called it. Unfortunately, the VM input broke down.

WaxfordSqueers

September 2nd, 2013, 23:15

Quote:

[Originally Posted by Kayaker;95344]This may be of no use, but I was playing with yet another NTFS utility, nfi.exe

Kayaker, old boy...nfi was of considerable use. It did not solve my overall problem but an example from it might illustrate the problem I am facing.

I have a directory, C:\aaa and in it I have two files...7z457.exe, which is the free 7zip installer, and notepad.exe. Below is a partial readout from the nfi file which I got using the standard DOS redirection, nfi c: > c:\mft.exe. That gave me a 7 meg file with all the mft entries.

```
File 25744
\aaa
$STANDARD_INFORMATION (resident)
$FILE_NAME (resident)
$INDEX_ROOT $I30 (resident)
```

```
File 25745
\aaa\7z457.exe
$STANDARD_INFORMATION (resident)
$FILE_NAME (resident)
$DATA (nonresident)
logical sectors 13718824-13720511 (0xd15528-0xd15bbf)
```

```
File 25746
\aaa\notepad.exe
$STANDARD_INFORMATION (resident)
$FILE_NAME (resident)
$DATA (nonresident)
logical sectors 13195840-13195975 (0xc95a40-0xc95ac7)
```

You will notice that each file is numbered, with notepad being File 25746. In hex that is 0x6492, a magic number I found while exploring with softice the other night in the NTFS.sys code. That was a good find.

The other thing to note is the logical sector number for notepad, 13195840. There are 512 bytes per sector on my disk so that LSN converts to a byte offset of 6,756,270,080. Using the free Active@ Disk Editor, I went to that offset, and, bingo, there was the MZ header for notepad. I confirmed that by comparing it to the hex readout for notepad. I did the same for the 7zip installer.

OK...here's my problem. How do I relate that location to the MFT file? On the nfi readout, there are 29,617 files all in MFT segments with signature 'FILE'. Those 29,617 file numbers are part pf a binary tree system that relate back to an earlier entry in the MFT.

There are two ways I can do this. One is to decipher Microsoft's Madness, which most people on the Net have failed to do, as far as I can see. Even Microsoft claims it is not trivial to

trace through the MFT. I have yet to find an explanation for the MFT on the Net other than a trivial and generalized presentation.

The other way is the one I am undertaking using reverse engineering. I have identified the point in shell32 where ntfs is called initially. It does not make a lot of sense since it is called from the function `_SHCreateQueryCancelAutoPlayMoniker`.

<http://msdn.microsoft.com/en-us/library/windows/desktop/bb762140%28v=vs.85%29.aspx>

They claim this function is deprecated and that `CreateClassMoniker` should be used:

<http://msdn.microsoft.com/en-us/library/windows/desktop/ms688698%28v=vs.85%29.aspx>

This is further based on `IMoniker`:

<http://msdn.microsoft.com/en-us/library/windows/desktop/ms679705%28v=vs.85%29.aspx>

What any of this has to do with programming, hardware, or anything else is beyond me. This is highly obfuscated Greek that only Microsoft could develop.

Whatever, it leads to `NTFS.sys` and the processing of the MFT file. The fear I have is that the MFT processing is related only to `IMoniker` and that I will have to dig further to find the real deal. I am thinking that because there are references in the initial code in `shell32` to `_SetFileTime` and `__imp__ReadFile`. Why they want to set the file time before doing anything else makes little sense unless they need to set the file access time during that process. But why set the file time before finding the file?

So...I am tracing through from there to find out how Windows finds the file. As I said, there has already been reference to the Notepad file number in the `c:\aaa` directory on my system. That's encouraging but the point I was at before with `NtfsReadFileRecord@28` is too far past the preliminary entry into `ntfs.sys`, so I have to go back and see what it is doing to find file record numbers, etc.

Kayaker

September 2nd, 2013, 23:46

Madness indeed. Maybe you've seen this in one form or another, but the beginning of this file describes the ntfs structure with some details perhaps not written elsewhere. There is mention of the b-tree, how resident vs non-resident attributes are handled, and other things which might be useful, if it's not entirely outdated that is.

http://read.pudn.com/downloads171/sourcecode/windows/vxd/794585/ntfs/ntfs.h_.htm

blabberer

September 3rd, 2013, 07:13

Quote:

the file access time during that process. But why set the file time before finding the file?

`SetFileTime` Needs a Valid `FileHandle` which is fetched through `CreateFile` so it could be an existing file or a new file by Passing `OPEN_EXISTING` or `OPEN_ALWAYS` flags

so it already has found the file

you can always prevent this api from setting the file time by providing appropriate values (0xffffffff) in fact all of the antivirus programs open and scans any file that you want to view before it is presented to you and they appropriately prevent changing file access time

WaxfordSqueers

September 3rd, 2013, 17:59

Quote:

[Originally Posted by Kayaker;95361]Maybe you've seen this in one form or another

Yes...in several different forms. This one is typical of what is out there, although some are better presented than others. They talk about the MFT, describing it's structure, but none I have found seriously attempt to trace a file through it.

Obviously some have traced through it to produce disk editors and the likes, like the Active@ Disk Editor, but even that app is only partially finished. They do a good job of laying out parts of the MFT graphically, colouring each attribute within a file. However, they have stopped short of the files with `INDX` signatures that represent directories. They even run out of steam in larger file headers, marking the parameters as general data. However, they do an excellent job of covering what they have covered.

I have seen one attempt at describing how offsets are described in the MFT so as to find a file that is non resident. The difference between resident and non-resident depends on the size of the file. Only files of slightly less than 1 Kb can be stored inside the MFT. Larger files are referenced in attributes inside the MFT but they have pointers to the file which is located outside the MFT. I have given an example of that below where the 7 zip installer is located outside but referenced inside.

I finally figured it out last night and was able to trace the 7 zip installer and notepad's location from inside an MFT record. It gets complicated because they talk in clusters and I figured out with 512 bytes per sector at 8 sectors per cluster that you multiply the cluster pointer given in the MFT by 0x1000 to get the actual byte offset outside the MFT.

That is, 512d bytes/sector = 0x200 bytes/sector and 0x200 bytes/sector x 8 sectors/cluster = 0x1000 bytes/cluster. Therefore, multiplying the number of clusters in the MFT by 0x1000 gives the byte offset on disk relative to the beginning of the disk partition (LCN). There is also a VCN, which is the offset from the beginning of the MFT. I found another magic number in my exploration with softice, 0x1924400. I had no idea what it was but it suddenly occurred to me that it is a VCN.

The base address of the MFT is at byte offset 0xC0000000 and adding 0x1924400 to that gives 0xC1924400, which is the offset of the 'FILE' signature representing the MFT record for the 7 ZIP file I had in directory `c:\aaa`. That's the record I found while tracing through `NTFS.sys`.

Within that record is reference to a cluster offset of 0x1A2AA5. Multiplying that by 0x1000 gives 0x1A2AA5000 and that is the address of the MZ header for the 7 Zip installer.

This is all a bit loose in my mind at this time and rattles around in there with other loose nuts and bolts. I need to firm up the process of how to find that record referencing the non-resident 7 Zip installer in directory `c:\aaa`. That could be a lot easier said than done because it involves dealing with b-trees with close to 30,000 entries. The record referencing the 7 zip installer is file record number 25745 and you can imagine sifting through more than 25,000 records to trace the b-tree.

At this time, I have no idea where each directory has priority in the b-tree. In other words, does a directory get listed in the b-tree so it is easily found, rather than tracing through thousands of entries to find it?

Another question to consider. If part of the b-tree path is eradicated, can it be re-constructed using what is left? I don't mean the entries that have been lost, I mean can the remainder be spliced to the root in order to see what is left?

Each file record has an identifier, as I described in my last post. I don't know at this point if that is the identifier used by the b-tree to create its nodes.

WaxfordSqueers

September 3rd, 2013, 18:08

Quote:

[Originally Posted by blabberer;95362]SetFileTime Needs a Valid FileHandle which is fetched through CreateFile so it could be an existing file or a new file by Passing OPEN_EXISTING or OPEN_ALWAYS flags

Thanks for the info Blabs. I am going to trace through it again and this time I will pay closer attention to what is PUSHed for `CreateFile`.

Quote:

[Originally Posted by blabberer;95362]so it already has found the file

That's what is confusing me. Windows differentiates between a file as an object and a file as an image on disk. Obviously `CreateFile` returns a handle before `SetFileTime` is called but

is CreateFile returning a handle to an abstract object or to the actual image on disk?

Last time i passed through this section of code, or a similar one, the CreateFile process lead to SetFileTime then to ReadFile. It progressed through the fltmgr to NTFS.sys and found an MZ header. It did not do any of the stuff I have just been through in ntfs.sys like NtfsReadFileRecord, and I presumed that had been done before. When I retrace it from the stack return points I have recorded I'll know more.

WaxfordSqueers

September 4th, 2013, 00:13

Quote:

[Originally Posted by blabberer;95362]so it already has found the file

Just traced through the entire thing again and you are right, is has found the file by the time SetFileTime is called...even before CreatFile is called.

I verified that by using a file that had never been loaded before. When I set a bpx at a location where a previously loaded file had broken, it missed the BP. Obviously a fresh file that has never been loaded into the file cache takes a different code route.

After CreateFile returns a handle, the ntfs code knows about the file number for notepad. It appears in a buffer like a magic number.

Kayaker

September 4th, 2013, 00:21

Might as well add these three NTFS powerpoint docs to the list of reading material. Ntfs, Ntfs Recovery, Ntfs Encryption.

blabberer

September 4th, 2013, 00:36

an article and a python indx parser
http://www.williballenthin.com/forensics/indx/index.html

@k
hey i was thinking where i saw this nfi and had to download it to confirm (its damn old nt4 resource kit)

want some thing glitzy and glazy ?

check out the graphical engine for ntfs analysis (gena package from tzworks)
will list volume wise so simply browse to c:\windows\system32\notepad.exe

or something less glitzy is ntfswalk by dmitry byrant will list by mft ID you can use the MFT id that was generated by nfi to look at this

both will show the LCN
ntfswlak by byrant is decimal
gena is hex display

so for me notepad is

Code:

```
C:\Documents and Settings\Admin\Desktop\oem3sr2\nfi>nfi c:\WINDOWS\system32\note
pad.exe | grep log
logical sectors: 28616744-28616870 (0x1b4a828-0x1b4a8af)
```

see below for cluster run in gena

```
C:\Documents and Settings\Admin\Desktop\oem3sr2\nfi>set /a 0x369505*0x1000 <----- gena
1766871040
C:\Documents and Settings\Admin\Desktop\oem3sr2\nfi>set /a 28616744*512 <----- nfi
1766871040
C:\Documents and Settings\Admin\My Documents\dloads\ntfswalker>set /a 3577093 * 4096 <----- ntfswalker
1766871040
C:\Documents and Settings\Admin\My Documents\dloads\ntfswalker>
```

Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

369505000 4D 5A 90 00 03 00 00 04 00 00 00 FF FF 00 00 MZ.....ÿÿ.. < notepad mz header

2806

gena will show

Path: [root]\WINDOWS\system32:{DA6227CB-326B-4B4D-9A81-04B61F1538DD}\notepad.exe

File Record:
MFT entry: 0x00000f3e [3902]
Seq Num: 0x0f3e [3902]
Type: file
Ref Count: 1

Standard Info Attrib: (ARCHIVE)
modified: 04/14/2008 00:12:30.000
accessed: 09/03/2013 20:26:11.531
mft mod: 09/04/2013 04:50:10.937
created: 01/01/1980 00:00:00.000
secure id: 0x00000000 [0]
quota id: 0x00000000 [0]
Jrnl num: 0x62318388 [1647412104]

Filename Attrib: notepad.exe
parnt mft: 0x00000a37 [2615]
modified: 04/14/2008 00:12:30.000
accessed: 01/20/2010 00:00:00.000
mft mod: 04/14/2008 00:12:30.000
created: 01/01/1980 00:00:00.000

security descr
resident data used: 0x0098

Security Identifiers
owner sid [S-1-5-32-544]
group sid [S-1-5-32-544]

Discretionary Access Control List
access allowed Local System DELETE | READ_CONTROL | WRITE_DAC | WRITE_OWNER | SYNCHRONIZE

access allowed Admins DELETE | READ_CONTROL | WRITE_DAC | WRITE_OWNER | SYNCHRONIZE
access allowed Users READ_CONTROL | SYNCHRONIZE
access allowed Power Users READ_CONTROL | SYNCHRONIZE

Data Attrib: (unnamed)
allocated: 0x000000011000
file size: 0x000000010e00
valid data: 0x000000010e00
cluster runs
0x00369505 -> 0x00369515 <-----

obj id
resident data used: 0x0010
9d30c76e-b92e-11e0-b97f-001f3a5a6fe5

blabberer

September 4th, 2013, 01:10

might then as well add the completed pdf half of which i posted earlier in this thread
2807

WaxfordSqueers

September 4th, 2013, 19:16

Quote:

[Originally Posted by blabberer;95374]might then as well add the completed pdf half of which i posted earlier in this thread
2807

That's good info but the reason I started this thread was to find a way to that file record from the MFT root.

If you look on your PDF at offset 2C (A1922C) you will see the file number 0xD5C5, which is decimal 54,725. How do you wade through those 54,725 MFT files from the MFT root, all with signature 'FILE', to find that record? That's what I am trying to accomplish by tracing through NTFS.sys, hoping to find the path used by Windows to find a file on disk.

On my external disk, there is a break between the MFT and it's metafiles, hence the the non-resident files. I am trying to decipher how the b-tree is initialized and setup to link those records.

The record you have posted for devtes~1.hxi does not indicate the root directory. You need that to associate it with the MFT root directory record.

I have a simplified directory at c:\aaa in which I have 2 files. I have found the record for the directory and each of the two files which are located successively in the MFT. I can trace forward and locate the MZ headers for both files but I cannot trace backward to link them to the MFT root.

That is the problem.

I have been scheming meantime. I thought of creating a new directory through the file manager, capturing the process, and following it to the MFT.

WaxfordSqueers

September 4th, 2013, 20:00

Quote:

[Originally Posted by blabberer;95374]might then as well add the completed pdf half of which i posted earlier in this thread
2807

I went back and found your original post at

<http://www.woodmann.com/forum/showthread.php?15188-Softice-and-createfile/page3>mid page.

You may think I have been ignoring you or that I am terminally stupid, but I have been going in so many direction on this problem that I don't know which way is up at times. Between trying to learn windbg and trying to setup softice on a VM in which I could not set BPs. then having to re-install, I have forgotten much of what has already been posted. On top of that, I took a week's vacation and rebuilt two computers, spending hours trying to get a new motherboard to recognize the existing disks.

A couple of posts ago, I discovered on my own what you had already written. Sorry about that. However, it gets me not much closer to my initial reason for this thread and that's to find a way to repair a broken MFT record.

We now know how to get from a file record for a specific file to the file's MZ header. The MFT record at Inode 5, I think it is, is called the dot file record because it represents the root directory. I am going to explore in there to see if it gives a clue as to how the first sub-directory can be accessed and to find a semblance to a b-tree linkage.

That's what I really want to do, take an existing MFT that has been truncated and rebuild it...if possible. Obviously, files are missing and I want to see if it's possible to splice the tree back together so windows will read the rest. There are scads of files on the disk that I have recovered using another app, but that app finds them using known file signatures. I would like to rebuild the directory structure, placing the files back in their original directories.

WaxfordSqueers

September 4th, 2013, 20:04

Quote:

[Originally Posted by Kayaker;95372]Might as well add these three NTFS powerpoint docs to the list of reading material. *Ntfs, Ntfs Recovery, Ntfs Encryption.*

K., here are two docs that I posted earlier:

<http://dubeyko.com/development/FileSystems/NTFS/ntfsdoc.pdf>

<http://grayscale-research.org/new/pdfs/NTFS%20forensics.pdf>

WaxfordSqueers

September 5th, 2013, 00:21

Quote:

[Originally Posted by WaxfordSqueers;95378]K., here are two docs that I posted earlier:]

Here's an interesting ntfs-related project that may interest the advanced programming abilities of Kayaker and Blabs.

<http://www.codeproject.com/Articles/81456/An-NTFS-Parser-Lib>

blabberer

September 5th, 2013, 01:00

Quote:

You may think I have been ignoring you or that I am terminally stupid,

why should i judge you by looking from my peep hole ? no

i am still not sure what exactly you are aiming at

i posted the pdf and i posted how to extract a file from a file0 signature

the only link to a parent from that structure is the parent directory entry
you would need to follow the trail from there only

read the post again which you linked get a hang of the structures and look at offset **a192b0** you will see 0x60ed this is the parent of devtext.xxxxx

Code:

```
Filename Attrib: devtest_g.hxi
parnt mft: 0x000060ed [24813]
```

now if you follow this parent you will see it has a parent

Code:

```
Path: [root]\WinDDK\7600.16385.1\help
```

File Record:

```
MFT entry: 0x000060ed [24813]
```

which again has a parent

Code:

```
Path: [root]\WinDDK\7600.16385.1
```

File Record:

```
MFT entry: 0x000060ca [24778]
```

again a parent but this time it is the root

Code:

```
Path: [root]\WinDDK
```

File Record:

```
MFT entry: 0x000060c9 [24777]
```

it points to itself as parent (i am you you are me them are us and they are we)

so it is all intertwined

so if you find the parent directory you would need to find its cluster runs and carve out each file
so no file that points to a parent directory is left

now you need to climb one step up since this is a directory you would need to carve all the directories first and follow its subdirectories till you reach a file entry and so on

the thing is like a single celled ameoba splitting itself into two at regular intervals and the children ameoba splitting itself into two again at regular intervals

the 2^x

simplyfying greatly leaving alone corner cases optimizations gotchas special cases

each directory can have sub directories and every directory has got only one great greatest grandest father the inode 5 also called dot directory

it is not possible to find the ameoba that first split into two when you are looking at a culture some hours after you left a single ameoba in the petri dish you would need to stop the grandchildren from proliferating and do a dna sequencing sort them into hierachy and label each of them to come anywhere nearer the first single celled organism

WaxfordSqueers

September 5th, 2013, 01:46

Quote:

[Originally Posted by blabberer;95380]i am still not sure what exactly you are aiming at

You have hit on a lot of it in this reply. I need to look into it more. The b-tree/ameoba comparison is apt. I have done similar stuff studying digital theory in electronics with Karnagh maps. Not quite the same but tracing through digital logic can be like tracing a tree structure.

Right now I am immersed in an interesting find. Kayaker posted some code the other day with calls to `_NtfsReadFileRecord` and `_NtfsReadMftRecord`. Sandwiched in between is a call to `_NtfsFindCachedFileRecord`, which looks to see if the file is in the file cache. When it return it does a test `al, al`, which has always been 0 when I passed through and that always lead to `_NtfsReadMftRecord`, which always had the file record number.

This time, instead of following the normal trail, I forced a 1 into `al` and altered the `Z` flag so a jump was forced. I had no idea where it was going. It spat at me and complained by sending me through an error function which complained further of corrupted files and setting 'dirty tags', presumably for `chkdsk` to handle.

I was expecting a program termination with an error message box but instead, it sent me back through `NtfsReadFileRecord`. Bingo...I hit the holy grail...the memory location of the \$MFT record. Now I am going through routines that check the attributes in each FILE record and that could get lengthy if it checks all of the 18 or so base metafiles.

Anyway, it is revealing a few things such as offset number 4 in each attribute being the attribute length. It compares that to 0x400 which is the size of each FILE record. So far, we have checked the \$VOLUME record and I have no idea where this is heading but it's interesting.

I have read that the MFT is loaded into memory during the volume initialization at boot time. I am just hoping it will reload the file I am using from the disk.

blabberer

September 6th, 2013, 06:06

Quote:

Sandwiched in between is a call to _NtfsFindCachedFileRecord,

if you get bored from tracing and would want to get an overview of all the calls that xyz apis make directly you can ask windbg to tell you with

uf /c <symbol / addr/ @eip>

for example ntfsReadFile call all these ddi's directly

uf /c Ntfs!NtfsReadFileRecord

Code:

Ntfs!NtfsReadFileRecord (bae93b5c)

Ntfs!NtfsFsdClose+0x3b9 (bae9248d):

call to Ntfs!FSH_apis (bae6d300)

these are not sequential calls but will be called based on results of earlier calls for example RaiseStatus may be called from failure path or never called at all

any way you wrote earlier that you would want to trace through file creation

then try Ntfs!NtfsCreateNewFile:

set process specific breakpoint first

one way to curtail spurious breaks could be to

open a command prompt
and type copy con blah.txt

!process 0 0 cmd.exe

bp /p <EPROCESS of cmd.exe FROM above command > ntfs!NtfsCreatenewFile
g

now if you start writing to the file thsi will break in proper context

btw this ddi takes 20 parameters 😊 should be fun to unravel it have FHUN

a call stack

Code:

kd> kb

ChildEBP RetAddr Args to Child

804f4628 bae05b22 ff00718 81502e48 81502f48 Ntfs!NtfsCreateNewFile

the third parameter to ZwCreateFile is OBJECT_ATTRIBUTES check it

kd> !obja 0013f2f0

Obja +0013f2f0 at 0013f2f0:

Name is blah.txt

OBJ_CASE_INSENSITIVE

the first param is an undocumented IRP_CONTEXT structure whose 0x24 param is a pointer to IRP Structure which contains a file object

kd> dt -r nt!_IRP ta..or->Ft* @@masm(poi(poi(@esp+4)+24))

+0x040 Tail :

+0x000 Overlay :

+0x024 OriginalFileObject :

+0x01c FinalStatus : 0n0

+0x030 FileName : _UNICODE_STRING "\\Documents and Settings\\admin\\blah.txt"

Code:

kd> !irp @@masm(poi(poi(@esp+4)+24)) 1

Irp is active with 9 stacks 9 is current (= 0x81502f48)

it is an IRP_MJ_CREATE [0,0]

the MajorNames are in an array whose 0th element foo[0] ==

kd> dpa nt!IrpMajorNames L1

8069c118 8069c3dc "IRP_MJ_CREATE"

sizeof(IRP_CONTEXT) == 0x44 dwords (see NtfsInitializeMFT -> ZeroMemory(&...,0,sizeof(...)) Constant) and it will contain a pointer to the MFT record of current directory structure in my case the admin directory

Code:

```
kd> .load domdbg
dom WinDBG extension v0.3 loaded
kd> !process 0 "FILE0" -c "dps edi i44"
```

Code:

```
kd> dt Ntfs!_NTFSMFT c7716000
+0x000 MAGIC           : 0x454c4946
+0x004 UpdateSeqOffset : 0x30
```

WaxfordSqueers

September 6th, 2013, 17:58

Quote:

[Originally Posted by blabberer;95387]if you get bored from tracing and would want to get an overview of all the calls that xyz apis make directly you can ask windbg

Thanks for reply and I will check it later, I have to go out. I just want to say that I tried windbg the other night in the VM with !filecache and it complained that it could not find any vacbs, I think it was. I tried it in debug mode (adding /debug in boot.ini) as well with the same results.

I also tried ntfswalk and it has reported scads of directories and files in the unallocated region. Oddly, I cannot find any of the 'FILE' records with pertinent material. The main system records are there up to about file 16 but that is followed by up to 0x255 seemingly empty records with the 'FILE' signature. It's like something has written over the existing files.

I was reading last night that the files may still be there. Apparently files are never deleted completely, just like the FAT system, they are just marked as not there by the \$BitMap attribute. I need to explore that attribute to see if turning bits (clusters) on and off does anything. For all I know, all my information is there and just hidden.

WaxfordSqueers

September 10th, 2013, 01:55

Quote:

*[Originally Posted by blabberer;95380]i am still not sure what exactly you are aiming atread the post again which you linked get a hang of the structures and look at offset **a192b0** you will see 0x60ed this is the parent of devtext.xxxxx*

I have done a lot more reading and I am getting a better understanding of the MFT structure.

In your reference above, you mention the parent directory 0x60ed. That is actually the record number of the file record.

Here's how I understand it. The MFT begins with record \$MFT which is file record 0, or inode 0. The \$MFT record is followed by 16 system file records and user file records begin at about the 24th record. So, all file records begin with the signature 'FILE'. In your case file record 0x60ed is the decimal 24813 record from the beginning of the MFT table, where \$MFT is file 0.

At inode 5, the 0x5 file record, is the 'dot' or '.' file record, which represents the root directory of the directory/sub-directory system. The b-tree lives in this record which is regarded as another file in NTFS terminology. Or, at least, node 0 of the b-tree is here and most of the rest is non-resident and pointed to in this record.

To summarize that, file record # 5, which is inode 5, begins with signature 'FILE', and is regarded as just another file in the MFT structure. However, the function of that file is to layout the directory/sub-directory index, which may contain other files, and whose files use the signature INDXX.

Here is a copy from my system:

Code:

```
Offset | 0 1 2 3 4 5 6 7 - 8 9 A B C D E F | ASCII
-----
00C0001400 | 46 49 4C 45 30 00 03 00 CF 0A 00 0C 00 00 00 00 | FILE0.....
00C0001410 | 05 00 01 00 38 00 03 00 38 02 00 00 04 00 00 00 | 8 8
```

Offset 000 = file signature = FILE
 Offset 004 = offset to update sequence = 0x30
 Offset 006 = update sequence size in words = 3
 Offset 008 = \$LogFile sequence number (LSN) = C000ACF = 201,329,359
 Offset 010 = Sequence number = 5
 Offset 012 = Hard Link count = 1
 Offset 014 = offset to the first attribute = 0x38 *****
 Offset 016 = Flags = 3 00 (in use and a directory)
 Offset 018 = Real size of the file record = 0x238 = 568 bytes
 Offset 01C = Allocated size of the file record = 0x0400 = 1024 bytes
 Offset 020 = Base File record = 0 (\$MFT)
 Offset 028 = Next attribute ID = 0x19 = 25
 Offset 02C = ID of this record = 5 *****
 Offset 030 = Update Sequence Number = 12 00
 Offset 032 = Update Sequence Array = 05 00 00 00

Offset 014 told us the first attribute was located at offset 0x38

Offset 038 = Attribute \$10 = \$STANDARD_INFORMATION (contains file time/date etc.)

Inside attribute \$10 it tells us the offset to the next attribute is 0x18
 Also, it tells us the length of the \$10 attribute is 0x60, including header.
 Adding 0x60 to Offset 038 gives us Offset 098

Offset 098 = Attribute \$30 = \$FILE_NAME, which tells us the file name of record 5 is 0x2e, or . (ie. dot)

That attributes size is also 0x60, so adding that to Offset 096 gives us Offset 0F8

Offset 0F8 = attribute \$90 = \$INDEX_ROOT

Bingo!! Here is the root directory attribute. If the directory index is really small, it can all be described here. If not, pointers are required to a non-resident index described by the \$0A attribute = \$INDEX_ALLOCATION.

The word 'allocation' gives us a hint. On a volume, space is either allocated (in use) or not. Another attribute \$B0 - \$BITMAP, keeps track of allocated and unallocated space by marking a cluster in a table as allocated or not allocated. So, \$INDEX_ROOT, \$INDEX_ALLOCATION and \$BITMAP work together to keep track of directory/sub-directory indexing.

In the dot directory (file record 5), if the directory structure is small, there is no need for the \$ALLOCATION_INDEX attribute because everything is resident within the 1024 bytes allocated to file record 5.

Getting back to:

Offset 098 = \$90 attribute (\$INDEX_ROOT), it tells us that the name of the directory is at 0x18 = \$I30. All directories are called \$I30. It also tells us that the directory data is at 0x20. There you will find 4 x 64 bit time/date structures, file created, file modified, record changed and last access time.

Here's where it gets tricky for me. The next attribute at Offset 150 is the \$INDEX_ALLOCATION attribute, which describes non-resident directory info:

Offset 150 = \$A0
 Offset 154 = Length including header = 0x80
 Offset 158 = Non-resident flag = 1 (means data that follows is not in the MFT).
 Offset 159 = Name length = 4 (name is \$I30...another directory)
 Offset 15A = Name offset = 0x40 (\$I30 @ offset 190)
 Offset 15C = Flags = 00 00 = (not compressed, not encrypted, not sparse)
 Offset 15D = Attribute ID = 24 (???)

Now for the good stuff:

Offset 160 = First VCN = 0
 Offset 168 = Last VCN = 0 (not sure what this means yet)
 Offset 170 = Data Runs Offset = 0

This could be referring to \$INDEX_ROOT which has no data runs offset or VCN offset. (ie. it's right here)

Fast forward to:

Offset 198 = \$INDEX_ALLOCATION = Data run

At 198, you see the sequence 11 01 2C, meaning size = 0x11, cluster count = 01 and first cluster is at 0x2C. I am still not sure what size refers to but the cluster count is the number of times a standard sized cluster can be divided into the file length I have verified that with another situation.

So, my C: directory should be at cluster offset 2C from the beginning of the partition. Let's check:

Sure enough, at offset 2C000, there is an INDX signature.

I won't print out the whole thing, just the header:

Code:

```
Offset | 0 1 2 3 4 5 6 7 - 8 9 A B C D E F | ASCII
```

```
000002C000 1 48 4E 44 58 28 00 09 00 91 0A 00 0C 00 00 00 00 1 INDX/
```

You can see the \$AttrDef file which is an MFT file listed in the C:\ directory but hidden. Also, later in the record, you can see all the other MFT system files like \$Mft, \$MftMirr, etc., plus the Recycle bin.

At this time, I don't have a structure for the INDX files and I am working on that. Your reference to parent and child interactions applies to the nodes in the b-tree which is made up of \$I30 directories and INDX files.

WaxfordSqueers

September 10th, 2013, 05:11

Quote:

[Originally Posted by WaxfordSqueers;95401]I have done a lot more reading....

I am now proceeding to make myself dangerous. You know how directories like System Volume Information are inaccessible? Not anymore, I just turned off the file access flags that make it system and hidden and now it is fully accessible.

On my system SVI is file record 38 (0x26) and it has three attributes that control the permissions. There is one \$10 (\$STANDARD_INFORMATION) and 2 x \$30 (\$FILE_NAME). There are two \$30s I think because one is for DOS names.

Anyway, in \$10, at Offset 0x70 (from beginning of file record at FILE signature), there is a file permissions string. It was 06 00 00 00 which marks it for system and hidden. Changing it to 20 00 00 00 makes it an archive.

In both \$30s, at offset 0x50 from the beginning of the attribute (from the 0x30 signature) there is a flag string. It is normally set at 06 00 00 10, which marks it as system, hidden and a directory. The 10 at the end holds the directory bit while the 6 holds the hidden and system bits. Changing both \$30 attributes to 20 00 00 10 marks them as archives with a directory.

When I went to the file manager and double-clicked System Volume Information, I got a message box telling me I did not have file access and that I'd have to consent to get it. I hit OK, or whatever it was, and bingo, SVI was fully accessible.

As I said, could be dangerous. I presume the same can be done to make the \$MFT, \$MftMirr, and other hidden system files visible.

I wonder if it works making rootkits and other hidden malware visible? One would think a rootkit has to register itself in the MFT, otherwise how would it operate within an NTFS system? Same with hidden directories as might be revealed by rootkitrevealer or Gmer. Just change the file/directory attributes and have a look inside.

Same with registry keys I suppose, since the registry is just another file in the MFT. I have seen Windbg used to access the registry with a plugin. That would be more elegant than searching through it via the MFT but it's nice to know you could possibly access hidden registry keys via the MFT.

blabberer

September 10th, 2013, 15:44

Quote:

At this time, I don't have a structure for the INDX files and I am working on that.

INDX is structured like below it is similar to other attributes variable length structure consisting of a size and offset to information

Code:

```
kd> bl
0 e bae942fe 0001 (0001) Ntfs!NtfsCheckIndexBuffer "da poi(@esp+8);dt Ntfs!_NTATTR_STANDARD_INDEX_HEADER poi(@esp+8); dt Ntfs!_NTATTR_IN
```

Quote:

I have seen Windbg used to access the registry with a plugin.

do you have a name ? or a link ? i know windbg can natively parse through the registry files in raw format but haven't seen this plugin yet

Code:

```
kd> !reg findkcb \registry\machine\software\microsoft\windows\currentversion
```

WaxfordSqueers

September 10th, 2013, 19:13

Quote:

[Originally Posted by blabberer;95404]INDX is structured like below it is similar to other attributes variable length structure consisting of a size and offset to information

Thanks for info, I'll check it out.

Quote:

[Originally Posted by blabberer;95404]do you have a name ? or a link ? i know windbg can natively parse through the registry files in raw format but haven't seen this plugin yet

I should not have called it a plugin, it's a script.

Actually, after all my goofing around, the script is right here at RCE (credit to Lionel Hauenens for script):

http://www.woodmann.com/collaborative/tools/index.php/WinDbg_Script

I got the lead from this paper on hunting rootkits with windbg:

<http://www.reconstructor.org/papers/Hunting%20rootkits%20with%20Windbg.pdf>

It's got some good stuff on registry searches. Although the script is not related to a registry search per se, I seem to remember using it in a related manner.

Sorry if my faulty memory has mislead you.

WaxfordSqueers

September 10th, 2013, 22:57

Quote:

[Originally Posted by blabberer;95404]

Code:

```
kd> bl
0 e bae942fe 0001 (0001) Ntfs!NtfsCheckIndexBuffer "da poi(@esp+8);dt Ntfs!_NTATTR_STANDARD_INDEX_HEADER poi(@esp+8); dt Ntfs!_NTATTR_INDEX_RECORD_ENTR
```

I am still trying to get my head around what you are doing here. You wrote a similar input line in an earlier post with the 0001 (0001).

Is this statement meant to be kd>bl 0 e bae942fe or is it the whole line?

questions:

1)where did you get bae942fe from?

2)Where are you getting the values like Ntfs!NtfsCheckIndexBuffer?

Is this from the SDK or DDK?

Elenil

September 11th, 2013, 02:45

windbg command maybe

thats the advantage of a newer debugger compared to softice it also shows local variables but thats almost all what is the advantage , maybe it can find the call positions for jmps/calls but that isnt useful when file is crypted

however there was some project what tried to fix this

<http://www.woodmann.com/collaborative/tools/index.php/IDA2SICE>

but it doesnt give a command to iceext what would be useful

it make it easy to re-construct things what make it very simple to understand for lerned programmer , the same count for idas "create pseudocode" function what shows a c++ like code
instead of having the understanding how this really works

Kayaker

September 11th, 2013, 04:43

I'll see if I can babelfish that.

>bl is list breakpoints

The second line is the breakpoint set, address is derived from the symbol for Ntfs!NtfsCheckIndexBuffer from the standard ntfs pdb file
>bp Ntfs!NtfsCheckIndexBuffer would be the basic breakpoint

The fancy blabberese breakpoint annotations include 'dt Ntfs!_NTATTR_*' portions which are typeinfo additions added to the standard pdb file. Blabberer mentioned in an earlier post about having a secret stash of ntfs pdb's

Code:

```
| \---addtypeinfotntfs.pdb
|      addtypeinfot.bat
|      ntfs.c
```

His procedure I believe is outlined here

<http://www.woodmann.com/forum/showthread.php?10295-Mysteries-of-win32k-amp-GDI&p=72632&viewfull=1#post72632>

or

<http://www.woodmann.com/forum/entry.php?227-How-To-Add-TypeInfo-So-That-Dt-Commands-Work-Properly-In-Windbg>

/remove fish from ear

blabberer

September 11th, 2013, 06:57

bl = list breakpoints

in windbg you can set 4 types of breakpoints

bp = permanent breakpoint to set this you need an address like 0xbadd00d or a resolved symbol

bu = unresolved breakpoint this bp can be set in advance and doesnt matter if the load address changes due to aslr etc

the break will be set correctly on the address but having many of them can slow down windbg because windbg will load each modules symbols too as and when a new module is loaded and will try to resolve the symbol

ba = access breakpoint or data brekpoints or hardware breakpoints (read write execute byte word dword)

bm = memory breakpoints (not much usefull for reversing as you need private pdb with full typeinfo)

breakpoints can be disabled and enabled

bl will list all the breakpoints with their current status

Quote:

where did you get bae942fe from?

windbg resolved it for me

Quote:

Where are you getting the values like Ntfs!NtfsCheckIndexBuffer?

if you have the nms for ntfs.sys

typing **exp ntfs!*che*ind*** in softice should get it for you too

Code:

```
kd> .reload /f ntfs.sys
kd> x ntfs!*che*ind*
```

below are some steps that you can try reproducing

Code:

```
kd> bl
```

here is an unresolved breakpoint notice the u

Code:

```
kd> bc *  
kd> bu NtfsCheckIndexbuffer  
kd> bl
```

blabberer

September 11th, 2013, 07:25

ah the secret stash isnt that much of a secret

all you need is one xxx.c file winddk build environnement and a pdb to add your own type info

Code:

```
\>dir /b
```

the c file contains

Code:

```
\>type ntfs.c
```

the pdb is from ms symbol server which gets modified

WaxfordSqueers

September 12th, 2013, 02:48

Quote:

[Originally Posted by Kayaker;95409]I'll see if I can babelfish that.

I'll address this to both kayaker and Blabs and try to kill two babelfishes with one stone. 🙏

I have read quite a bit on windbg and understand a preliminary amount about their use of breakpoints. In fact I did a lookup of _ntfscheckindexbuffer@8 in softice and found a similar address (0xF89932FE).

Thanks to you and blabs for elaborating on the usage. I could not figure out where the address came from and you guys have explained it adequately. The lights have come on...thanks.

I am currently tracing through ntfs.sys using _ntfscheckindexbuffer@8 as a breakpoint and trying to equate blabberer's structure to what I am seeing. It's not perfectly clear yet and I am seeing offsets that extend beyond what blabs listed.

When I first used the BP it went off before I could exit ice and click on a file to load. However, an F5 lead me back to the file manager and I am now back in the code related to my file after the BP fired again. Meantime I looked for an INDX structure that was more comprehensive and that lead to a python parsing program. I am looking at reloading python after swearing I would never use it again. 😊

WaxfordSqueers

September 12th, 2013, 02:53

Quote:

[Originally Posted by Elenil;95407]<http://www.woodmann.com/collaborative/tools/index.php/IDA2SICE>

I use ida2ice to make nms files when I can't get a good one from Msoft's PDB file. However, the IDA structure gets annoying at times when they refer to register values using their in-house terminology. Also, they tend to replace useful values with names they have created such as structxxx.

WaxfordSqueers

September 12th, 2013, 09:29

Release the spinlock, Hal. 🙏🙏🙏

blabberer

September 12th, 2013, 10:45

Quote:

Release the spinlock, Hal.

what was that ?

Code:

```
lkd> x hal!*rel*spin*  
806d2868 hal!KeReleaseQueuedSpinLock = <no type information>  
806d870c hal!KeReleaseSpinLock = <no type information>
```


Kayaker

September 12th, 2013, 11:57

Oh Oh. I've seen those warning signs before, Waxford needs a reboot!

WaxfordSqueers

September 19th, 2013, 03:03

Quote:

[Originally Posted by WaxfordSqueers;95401]

Now for the good stuff:

*Offset 160 = First VCN = 0
Offset 168 = Last VCN = 0 (not sure what this means yet)
Offset 170 = Data Runs Offset = 0*

This could be referring to \$INDEX_ROOT which has no data runs offset or VCN offset. (ie. it's right here)

Fast forward to:

Offset 198 = \$INDEX_ALLOCATION = Data run

At 198, you see the sequence 11 01 2C, meaning size = 0x11, cluster count = 01 and first cluster is at 0x2C. I am still not sure what size refers to but the cluster count is the number of times a standard sized cluster can be divided into the file length I have verified that with another situation.

This is an excerpt from a post I made recently and is on page 6 of this thread.

The reference above to VCN = 0 was not clear to me at the time. A VCN is an offset into a file from an LCN and in this case it is zero because the reference is to a root index. The data run offset is 0 as well because everything is resident in this MFT-resident root directory and data runs are data streams outside the MFT.

The interesting part is the reference to the sequence at offset 198 which is a data run referring to data outside the MFT.

The data run 11 01 2C is a very simply data run pointing to one offset. It is interpreted as follows:

11 is in format xy where y refers to the following bytes. If y = 1, there is one byte following xy to indicate the size in clusters of the non-resident stream. If y = 2 there are two bytes following and if 3, there are 3 bytes following, etc. The case where y = 0 is a special case used for sparse files, which are files with long runs of zeros that can be omitted from the file by indicating how many zeros are in the run.

The x position refers to the number of bytes following the size parameter pointed to by y and refers to the size of the offset byte field.

Therefore 11 means there is 1 byte following the 11 for size (= 01) and 1 byte following the size for location (= 2C).

In this case, 11 01 2C means the size byte is 1 cluster and the offset (LCN) from the beginning of the partition is 0x2C (in clusters).

Here's a more complicated data run:

31 38 73 25 34 32 14 01 E5 11 02 31 42 AA 00 03 00

*****The thing to note is that each file offset is added to the previous one in the data run.

This data run might represent a fragmented file and it can be handled by breaking it into parts based on what was described above:

31 38 73 25 34.....38 = cluster length at LCN1 = 0x342573

32 14 01 E5 11 02 0x0114 = cluster length at 0x0211E5...add to previous LCN1 = 0x211E5 + 0x342573 = LCN2 = 0x363758

31 42 AA 00 03.....0x42 = cluster length at 0x0300AA.....add to previous LCN2 = 0x0300AA + 0x363758 = LCN3 = 0x393802

00 = end of run

Therefore this data run represents 3 fragmented non-resident files at LCNs 0x342573, 0x363758 and 0x393802. All LCNs are in clusters and with standard parameters with 512 bytes/sector and 8 sectors/cluster the physical offset can be found by adding 000 to the cluster address. That is, LCN1 is found at byte offset 0x342573000

In the first example, with cluster offset = 0x2C, the byte offset is 0x2C000. That address is from my system and indicates the directory of c:\ and has a file signature of INDX. NTFS refers to directories as INDX or \$I30.

WaxfordSqueers

September 19th, 2013, 07:17

I have been trying to understand the MFT through a combined reverse engineering and theoretical approach. I posted this thread because I was using softice to do the reverse engineering part. The reversing has been a lot tougher than anticipated and I think that is because I still tend to understand file systems with the old DOS structure in mind.

Unfortunately, for me, Microsoft has applied object-oriented theory to file systems. Based on the lack of practical solutions on the Net to problems like NTFS systems I think a lot of people on the Net can talk good theory when it comes to object-oriented systems but they have no idea what is going on at a hardware level or at the low level system level where Hal, NTFS.sys and Ntfskrnl reside.

I have made some notes and I would appreciate some comments. Specifically, I am trying to find where shell32 and ole32 begin communicating with the file driver system. In an earlier reply from Blabberer he commented that the system already knew where the file was located and that seems to be the problem. I can't seem to access the process early enough to catch where the initial call to the file system drivers is made.

Here are the notes, taken from a ppt presentation called ntfs_mod. However, this is more a summary for me since I have researched and understood the following process cerebrally. I am having trouble translating it to tracing code.

-NTFS implements files as objects.

-an application creates or accesses a file by means of object handles

-by the time an I/O request reaches NTFS, the Windows NT object manager and security system have verified the calling process authority.

-the I/O manager has also transformed the file handle to a pointer to a file object.

-NTFS uses the info in the file object to access a file on disk

-the file object represents a single call to the 'open-file' service. It points to a stream control block (SCB) for the file attribute the caller is trying to R/W

-the SCB represents individual file attributes and contains info about how to find specific attributes within a file

-all SCBs for a file point to a common data structure (File Control Block)

-the FCB contains a pointer to the file's record in the MFT.

Here are the stumbling points.

-starting with the last point above, how does the FCB have a pointer to the file's record. Is an FCB created for each file? If so, it must be connected to the namespace process I mention next. Maybe that's the missing link, the system knows from which namespace item is selected where to find the FCB.

-Windows has a file structure system (namespace) that parallels the directory/sub-directory a user encounters. It is handled by shell32.dll. At the same time, shell32 works with ole32.dll to set up an object whenever a file in its namespace is executed.

-somewhere in that process of parsing the path for the file to be executed, the object interface is formed by ole32. It's not too clear to me what happens next but somehow the file system manager gets into the act.

-as I understand it, the object is not the file but a structure which represents the file. If I put a bpx on CreateFile, I get a hit, and Createfile returns a handle. However, that process leads to a place in NTFS code where the file location is already known. I can tell that because it accesses the file's MZ header and verification of the header reveals it to be the file I

am trying to load.

-I would think that the initial handle returned by CreateFile should be used as above by the I/O manager which changes it from a handle to a pointer to a file object. In other words, the file location should not be known yet, but it is. As it says in my notes above, the SCB contains specific info on how to find file attributes within a file.

-it just occurred to me that the handle returned by CreateFile is a file handle and not the object handle referred to earlier. If CreateFile is not used to retrieve object handle, what is?

-I am thinking Windows does far more when it loads the MFT into memory at boot time than what is obvious. I have theorized, perhaps wrongly, that the file is found on disk through fairly straightforward means along the lines of the DOS system. I am beginning to think it is far more convoluted than that. For one, if the file is accessed frequently it is loaded at least partly into the file cache. I have noticed while tracing that NTFS.sys queries the file cache, presumably checking to see if the file is in the cache.

Woodmann

September 19th, 2013, 21:30

I wish I could offer some insight but, I see the whole thing as a linear process which it clearly is not. I think that windows does this on purpose so to make this endeavor as difficult, if not impossible as it can.

My opinion is, because the way windows stores files in fragmented sections, it has been beyond my skill level to decipher.
The calls go in directions I just cant comprehend.

I expect things in windows to be linux like and they are not even remotely close.

I so wish I could understand how windows does things.
I wonder how much more difficult it is in win8 ?

I want the old 8 bit days. Ugghfhhhh Age has been harsh to me.

Woodmann

WaxfordSqueers

September 20th, 2013, 01:31

Quote:

[Originally Posted by Woodmann;95446]I wish I could offer some insight but, I see the whole thing as a linear process which it clearly is not. I think that windows does this on purpose so to make this endeavor as difficult, if not impossible as it can.

Good to hear from you, Woody. I agree 100% but I'm not so sure it is a calculated obfuscation. I think they have over-complicated things by over-thinking the problem.

If you read the book on C++ by Bjarne Stroustrup, the guy who created C++, he gives really good examples of why the object-oriented side is required. However, having created the concepts, Stroustrup in on top of it and explains it all very clearly.

That's not the case with 95% of the other authors I have read on object theory. They talk around examples, they don't explain them. For example, with the concept of a class in C++, I have seen author after author talk around the subject, giving juvenile and cockamamey examples of a class without stating specifically what it is. In Stroustrup's book he comes right and states that a class is a user-defined type. In fact, he points out that C++ was based on giving more flexibility to programmers hence the definition of a type that was not as well-defined as char, int, etc. It gave programmers the power to be more creative.

Microsoft have taken that creativity too literally, although if you read a lot on the NTFS system, it becomes clearer as to why they have gone that route. It's basically a good idea but it gets far too complex and out of control. For example, the MFT, the central record keeping database of NTFS files, can only grow until a theoretical point where there is no more space between the MFT and the user-based file system. Microsoft has not built in the ability to shrink it, even after scads of files have been deleted. Part of the reason, as far as I can see is that each file has a file number and they have not built in the capacity to re-sort the numbers in such a manner as to downsize them. Slots for files at a specific number can be re-used but due to problems with hard links, etc., NTFS wont relinquish that space until certain conditions are met.

Disk utilities like Diskeeper will defrag the MFT but that is done in DOS mode after a reboot. Even so, I have an MFT on my C: drive that is now split into two sections and even after an MFT defrag it is still split in two. The split happens after the MFT becomes so large that it has to make part of itself non-resident as another MFT. It seems the MFT is subject to the laws of entropy and can only become more defragmented as time goes by, presumably to the point where the file system chokes to death.

Quote:

*[Originally Posted by Woodmann;95446]My opinion is, because the way windows stores files in fragmented sections, it has been beyond my skill level to decipher.
The calls go in directions I just cant comprehend.*

That's what I am dealing with now. I have traced the creation of a file instance from the file manager well into shell32 and ole32 but I am tracing blindly. As I go, I can make breakpoints so I can return to the same point but there are thousands and thousands of lines of code that seem to do nothing but parse paths. I have created a very short path in C:\aaa but even at that, the tracing path leads through shell32, shlwapi, ole32, and occasionally into the bowels of ring 0 for apparently unknown reasons.

One thing that really annoys me is the number of times the Windows system checks and rechecks things like paths and file names. They are anal about errors and I can imagine walking into the Microsoft programmers area and seeing men walking around with several belts holding up their pants along with several pairs of suspenders, just in case the belts should fail. You can get seriously immersed in nesting levels for function calls in the shell32/ole32 process.

Quote:

[Originally Posted by Woodmann;95446]I expect things in windows to be linux like and they are not even remotely close.

My hangup is thinking in DOS mode, or even at the assembler level. My background is in hardware and I refuse to immerse myself in a fantasy world that obfuscates the hardware level, or the assembler level. I am continually trying to interpret object theory into real life.

No matter what object-oriented people call it, an object is code. Why not make that clear? In the old days we called them sub-routines. Stroustrup explains that in large software systems there is a necessity to objectify the process and I understand that. I also think that someone should try to relate the objects to the real world.

Any object in a Windows system can be tracked down to a data structure. The concept of objects is far too general and that has crossed over to NTFS where everything is a file.

Even Microsoft gets confused in their morass of double-speak. On their site they explain the MFT by laying out the first few file records in the MFT. File number 0 is called MFT but it has a \$ sign in front of it to indicate that it is a system file record, not a user-mode record. \$Mft is record 0 in the MFT. I caught someone on Microsoft trying to explain that \$MFT is the MFT, that all files on the system were contained in it.

That's double-talk. All files on a system are NOT in \$MFT, in fact there is very little in \$MFT other than a reference to the overall MFT. There is absolutely nothing in \$MFT about other file records. They also tell us that the \$MftMirr, which is file record 2 in the MFT, can help reconstruct a damaged MFT. %MftMirr is a copy of record 0 to 3 in the MFT, so how can it help reconstruct anything unless those 4 records are damaged?

I have suspected that the 3rd record, \$LogFile has something to do with reconstructing a damaged MFT and Microsoft is keeping that from us. The LogFile is supposed to be for recovering from a crash. Apparently the NTFS writes transaction (any operation on a file) to the LogFile, which can grow to 50 megs, or so. As it gets to it's limit, it starts deleting entries near the beginning of the log. If the system crashes, NTFS has enough data in the log to recreate the system.

Quote:

[Originally Posted by Woodmann;95446]I so wish I could understand how windows does things.

At least you have encouraged me to keep looking at the code in shell32 and ole32 to see if I can find the missing link. I know basically that Windows shows a directory structure in file explorer that is realized as a series of structures in a namespace. As I trace through shell32 I can see the system processing those structures, which contain the path and filename broken into a structure with offsets listed to show where each record in the structure ends.

A bit later, ole32 gets into the act and shlwapi, which is a light-weight shell. I am just not clear on the objects being initiated in ole32. Ole contains interfaces that can be called by applications, then it becomes a matter of pointers being issued for the interfaces. After that, the object creation process begins.

I also know what the back end looks like. I have seen handles issued, IRPs issued, and I have traced through scads of NTFS.sys. I have yet to reach the point where the disk is accessed but I just read last night that ftdisk.sys is supposed to handle file I/O. I thought it would be Hal. It seems to me that I was into ftdisk code at one time but thought I'd gotten lost.

What I'd like to find is where the connection between the file manager and the object creation process lies. That's no guarantee of success, however, since one needs a virgin file that has not been used recently in the current NTFS system. If it has been used, it gets cached, so the data for the file comes from a memory cache, not the disk.

Quote:

[Originally Posted by Woodmann;95446]I want the old 8 bit days. Ugghfhhhh Age has been harsh to me.Woodmann

I concur, I try not to dwell on it. ☹

BTW...did you ever find Splaj?

Kayaker

September 20th, 2013, 02:40

Quote:

*[Originally Posted by WaxfordSqueers;95447]
Disk utilities like Diskeeper will defrag the MFT but that is done in DOS mode after a reboot. Even so, I have an MFT on my C: drive that is now split into two sections and even after an MFT defrag it is still split in two. The split happens after the MFT becomes so large that it has to make part of itself non-resident as another MFT. It seems the MFT is subject to the laws of entropy and can only become more defragmented as time goes by, presumably to the point where the file system chokes to death.*

I was going to ask you actually, since you've become familiar with parsing the MFT in raw format, whether you notice any difference if you defrag the MFT, or if that might even make the convoluted path you're trying to follow for your specific test files any easier if by chance their entries happen to be fragmented across MFT sections.

To that end I tried using contig from Sysinternals, which is supposed to be able to defrag the hidden metadata files. Unfortunately I can't get it to work, I get the same error Failed to open C:\\$Mft::\$BITMAP: as reported here:

http://forum.sysinternals.com/contig-mft-interpreting-output_topic26668.html

But in any case, in response to your comment about the MFT still being split in 2 even after defrag, that in itself might be normal as explained in the 2nd post in this thread:

http://forum.sysinternals.com/defragmenting-mft-with-contig_topic25001.html

For the record, my MFT as reported by the default Windows defrag analyzer has 2 fragments as well, is 77% in use, and is 61Mb in size. I think I read somewhere that the MFT will split when it gets over 200Mb, is that correct?

WaxfordSqueers

September 20th, 2013, 05:19

Quote:


[Originally Posted by Kayaker;95448].... whether you notice any difference if you defrag the MFT, or if that might even make the convoluted path you're trying to follow for your specific test files any easier if by chance their entries happen to be fragmented across MFT sections.

Thanks for links Kayaker. I don't see how at this point an mft defrag would help. The MFT lives as a database with it's origin at a specific address. Microsoft claims it has no permanent address so it can be moved in the case of a bad cluster. What I am trying to do is find where the MFT is called to go find a file on disk. You supplied an NTFS function earlier, I think it was _NtfsCheckFile Record, or something like that. I did explore that but all it was doing was testing the file record in the MFT for veracity. It seemed to return to shell32 without finding the file. I presume the link I seek came before that or after that.

On both my systems, with 512 bytes/sector and 8 sectors/cluster, (laptop win7 and desktop XP) the MFT is located at cluster offset 0xC0000. On my systems that is byte offset 0XC0000000. It's likely the same on yours if you have the same cluster factor and 512 bytes/sector, and you can see a file signature of FILE at that address. If you look down the file a short ways, you will see a reference to \$MFT. That is the \$MFT file record which is a reference to MFT itself. However, many people, apparently including Russinovich, the author of Contig, are using it in an odd manner.

I have made an error here but I will leave it and correct it later. My error shows how easy it is to get confused in the MFT

That may sound arrogant but consider the definitions supplied by Microsoft. The first 16 file records in the MFT are system records and that's why they have the \$ sign in front of them. The very first record is \$MFT and it is record 0. When Russinovich writes C:\\$MFT::\$BITMAP, he is suggesting that \$BITMAP is contained in \$MFT and it is not. \$BITMAP is a record in the overall MFT but it is not contained in \$MFT.

I have confused \$Bitmap the file record with \$Bitmap the attribute...explained below**

An interesting aside. Microsoft describes the MFT as a database with rows of file records and columns of attributes. That does make sense since it begins at file record 0 in the first row and each record has a header with attributes numbered from 0x10 (STANDARD_INFORMATION) upwards of 0xB0 in columns. You could visualize the columns as file header, attribute 1, attribute 2, etc. Not every file has the same attributes, however, so some columns would have no data in them. The data (header and attributes) in columns are telling you things about the file record in that row. Every file on the system is in its own row and the rows are numbered as file record numbers.

In the root directory, normally C:\, the metafiles beginning with \$ are listed in the C:\ directory but their attributes are hidden and system. The attribs cannot be changed using attrib from a DOS prompt. At this point, I don't know if the \$MFT is a reference to the entire MFT or just to record 0 in the MFT. Along with \$MFT can be found \$MftMirr and several other system MFT metafiles all beginning with \$.

Problem is, none of them are in the C:\ directory even though they are listed there. As I said above, the MFT database is located at 0xC0000000 on my system and my C:\ directory is at 0x2C000. It seems the references to the metafiles are hard links just as the recycle bin is a virtual file pointed at in C:\. Anyway, at 0xC0000000, is the \$MFT file record with it's file header and attributes, the rest of the file records, including \$MftMirr follow with each of the following 16 records being system records. Here are the system records in the MFT, all rows of the database:

```
0 = $Mft - the record with information about the MFT only. It contains a $Bitmap attribute ($B0) hence $Mft::$Bitmap.
1 = $MftMirr
2 = $LogFile
3 = $Volume
4 = $AttrDef
5 = no name...referred to as the dot record, or '.'. It is the directory system root.
6 = $Bitmap - note that $Bitmap is a separate file and not contained in $Mft <<<-----
7 = $Boot - the actual boot file is at the beginning of the partition with a copy at the end of the partition.
8 = $BadClus
9 = $Secure
10 = $Upcase
11 = $Extend
12 - 15 - reserved
```

The description given on my reference for \$Mft claims it contains one base file record for each file and folder on the NTFS volume. Horsefeathers!! If you look in \$MFT there is nothing there but a reference to itself. There is nary a mention of any other file.

It has a standard header with 4 attributes \$10, \$30, \$80 and \$A0. \$10 is standard information. \$30 is \$File_Name which tells you it's name is \$MFT. \$80 is more interesting, it is \$Data and is presumably a file's data. It gives the first and last VCN and tells you the size of the MFT. However, the data reference is to this \$MFT record only and to no other file on the disk.

\$80 has two data runs. The first tells you the MFT begins at cluster 0xC0000, which is also listed in the NTFS boot record at the start of the partition. The second data run is presumably a pointer to the other half of my split MFT. When I go to that address, sure enough, there is a FILE signature and the continuation of the MFT.

\$B0 = the \$Bitmap 'attribute', not the \$Bitmap 'file record'. Confusing??? So I am wrong, there is a \$MFT::\$Bitmap reference but the \$Mft is a file record reference and the \$Bitmap reference is to an attribute. I interpret C:\\$Mft::\$Bitmap as meaning the \$Bitmap attribute in the \$Mft file record.

The \$BITMAP record is apparently a record of available clusters on the system.

Anyway, it has two data runs in the attribute and data runs must be non-resident, or external to the MFT. The first is at 0xBFFFF and extends for 1 cluster, which is 0x1000. That means that portion of the bitmap extends from one cluster below the MFT to the MFT at 0xC0000. It is full of FFs, an indication of bit fields for each cluster on the drive.

The second \$Bitmap data run is at 0xDCA85 and extends for 6 clusters (0x6000). That bit field is far more sparse than the first one as might be expected since it represents a region on disk that has more free space.

To summarize, the name \$MFT seems to be used incorrectly by many people. \$Mft does not mean MFT. \$Mft is one record, the first, in the MFT database and its sole purpose is to give information about the MFT such as its name, its time/date information (when it was changed, created, etc.), its security attributes, its location, its size and the clusters it occupies. \$MFT tells you nothing about any other file on the system. That information comes from the user file records beginning after the system \$-records and referred to in the index records in the dot file record at file record 0x5.

That seems to be how you locate a file in the MFT. You trace the b-tree with its root node in the 'dot' file record 0x5. The b-tree is a sorted index based on the directory/file name, and once you have the path, you can find the file reference in the index. At that point, there is apparently a pointer to the associated file record.

After having explored the MFT to a decent depth and seeing its complexity I don't think I'd want to run something like contig on my main system. I tried that once with a boot disk from Comodo written in Linux and it completely screwed my system, installing a Linux folder while writing over my directories including Program Files.

I think part of the problem I am having now is based on what Comodo backup did on my external drive. Ultimately, it was a bonehead error on my part, mistaking a clone operation for an image operation. Comodo started cloning to my external disk without a warning that all data would be lost. That would have clued me into my error. Although it only wrote for a few seconds before I aborted it, the damage was done. It seems to have written over my external drive's MFT but why it would begin in mid disk with an MFT is not clear. I would think a sector by sector clone would work from sector 1 outward.

WaxfordSqueers

September 20th, 2013, 19:13

I may be going too far with my long-winded explanation so I don't mind someone saying something. I notice someone changed the title of the thread which is a good idea.

I still plan to attack this problem using reverse engineering and next time I will lay out an outline of my code tracing methodology so someone can follow along and duplicate the process if they like. I have taken a break from code tracing to try understanding the MFT structure better. It's a steep learning curve and passing on what I have learned for peer review is tough to do with terse replies. NTFS and the MFT give up their secrets reluctantly.

In my last windy reply to kayaker I was actually addressing the issues in the links he provided but forgot to say so. In one link there was an error message from contig that claimed it could not find \$Bitmap. That's likely because the \$Bitmap file is damaged, or the \$MFT file is damaged, or even the MFT itself. Just a guess but in the MFT verification routines I have traced the process of finding files and attributes in the MFT records is pretty foolproof. They ID the FILE signature of \$MFT then do the rest using offsets included in the file header or the attribute itself.

I won't pack the tracing reply with code but I will lay out key functions in shell32, etc., so someone can bp on them to save a lot of tracing time should they want to follow along. I am too bagged to do it right now.

If the thread is getting too long don't be afraid to say something. I am not easily offended.

Also, if Blabs wonders why I am not using windbg more it's because it won't work in the VM right now. It took me a long time to debug my softice install on the VM and I don't want to go through that right now with windbg.

Woodmann

September 20th, 2013, 22:34

It's not long winded. We are trying to understand how MFT does what it does.

Some zen sounds good about now. 🙏

Woodmann

Kayaker

September 20th, 2013, 23:09

No worries Waxford, your detailing of your findings is mucho appreciated, it's nice to discuss something new for a change. Yes, it's certainly not an easy subject matter and there's definitely a lot to absorb in how the NTFS is laid out. Let alone the undocumented code paths you're trying to follow. I changed the title to hopefully draw others into the thread.

Thanks for the clarification on why you think Contig might not be working. Yeah, in the first link I posted it didn't work, but in the second it apparently did, as expected. Interestingly, I tried running chkdsk and it did find an error in the MFT Bitmap:

Code:

```
CHKDSK discovered free space marked as allocated in the master file table (MFT) bitmap.
```

```
Correcting errors in the Volume Bitmap.
```

```
Windows found problems with the file system
```

After running chkdsk /f several times, all errors disappeared. However, Contig still gives the Failed to open C:\\$Mft::\$BITMAP: error. To be fair though, this is a VM I'm trying this on.

blabberer

September 21st, 2013, 19:36

i didnt have time to verify or reproduce or check the things so i cant reply any queries
i will just post generic thoughts on certain highlights

mft is was always 2 fragments as far as i have seen

if i have a severely fragmented mft i run fsutil usn deletejournal /n <drive> and recreate it (use query and save the max and alloc numbers first)
this normally brings back mft to 2 fragments

i normally run some boot defraggers (ultradefrag from source forge for example) windows defrag yells that it needs 15% commision to work i cant even afford 1 % sometimes on my real machine

at the moment my mft is 93% full with 2 fragments and the fragmented file is always \$bitmap

Code:

```
Master File Table (MFT) fragmentation
```

```
Total MFT size = 308 MB
```

as far as i have seen defragging BITMAP is kinda impossible as it has details about each and every sector including a reference to itself

i still dont believe shell and ole ole plays significant role in the process the object is created in R0 with ObCreateObjName & sisters and the handle is created with obpCreateObHandle and brothers does shell and ole send some undocumented DeviceIoControl to ntfs.sys

NtCreateFile -> IoCreateFile -> IoCreateFile->ObopenObjectByName -> lookup which goes on into obinsertDirectoryEntry etc loops is how a new file is created

Kayaker

September 23rd, 2013, 00:20

I was trying to think of a way to break into ntfs.sys code which would lead to the MFT being accessed, relatively directly. So I used both OSR's IrpTracker and APSoft's IrpTrace to log the ntfs IRPs used during file creation, opening or saving.

Needless to say there was way too much activity, so I needed something very specific to reduce the noise. I decided to focus on the creation of a hard link (not a shortcut) to a file, thinking that that would be added to the MFT for the file.

The specific IRP to be logged is IRP_MJ_SET_INFORMATION, and the specific InformationClass to look for is FileLinkInformation. This is the IRP used to create an NTFS hard link to an existing file, which can be done with the CreateHardLink() API.

<http://msdn.microsoft.com/en-us/library/windows/hardware/ff549366%28v=vs.85%29.aspx>

I used fsutil to create the hard link (fsutil hardlink create NewFilename ExistingFilename)

http://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/fsutil_hardlink.mspx?mfr=true

Here is the result from IrpTrace for IRP_MJ_SET_INFORMATION / FileLinkInformation, when I created a hard link called 'alink' for the file C:\a.txt

>fsutil hardlink create alink a.txt

Code:

```
>> IRP_MJ_SET_INFORMATION Drv:Ntfs, Dev:#822AA020, IRP:81BDF5B8, Proc:fsutil.exe, Thr:618
      Control: E0 (SL_INVOKE_ON_CANCEL | SL_INVOKE_ON_SUCCESS | SL_INVOKE_ON_ERROR)
      FileObject: 82312368
```

If we go back to the definition of IRP_MJ_SET_INFORMATION we see that the Hex Dump above is the FILE_LINK_INFORMATION structure contained in Irp->AssociatedIrp.SystemBuffer

Code:

```
typedef struct _FILE_LINK_INFORMATION {
    BOOLEAN ReplaceIfExists;
    HANDLE RootDirectory;
```

There is one more bit of info given by IrpTracker that I didn't show, which may be the return from the hook in Ntfs, and it points to the _NtfsCompleteRequest function.

So what we've got now is an ntfs function to break on (_NtfsCompleteRequest) and a conditional parameter to test for (Irp->AssociatedIrp.SystemBuffer ==> FILE_LINK_INFORMATION.FileName). Tracing back from that should lead to the calling code.

Or, more directly, do a general break on the ntfs main Dispatch routine for IRP_MJ_SET_INFORMATION where *IrpSp->Parameters.SetFile.FileInformationClass* == FileLinkInformation. The IRP would have been issued via a fsutil CreateHardLink() call.

EDIT: This is found simply with Softice.

'Driver ntfs' gives offset of IRP_MJ_SET_INFORMATION, referencing with IDA points to

.text:00003ABB ; __stdcall NtfsFsdSetInformation

Or, all the IRP Major Functions are clearly listed in the INIT DriverEntry section with symbols anyway, so you can get it all from IDA alone.

My thinking then is that one might be able to trace from the breakpoint to the writing of the hardlink to the MFT.

EDIT: Function names of interest encountered:

NtfsUpdateScbFromFileObject, NtfsLookupInFileRecord, NtfsChangeAttributeValue, etc.....
so, yeah.

Here are a couple of links that support what should happen in theory at least:

"when you create a new hardlink for a file, ntfs will create a new filename(\$30) attribute in mft."

<http://www.flexhex.com/docs/articles/hard-links.phtml>

<http://www.osronline.com/showthread.cfm?link=228355>

Ultimately I don't know if this would provide an answer to whatever the current question is, it was just of a way of trying to get "into" the MFT.

blabberer

September 24th, 2013, 04:23

Quote:

Or, all the IRP Major Functions are clearly listed in the INIT DriverEntry section with symbols anyway, so you can get it all from IDA alone.

well you can get it for free with windbg alone 😊

Code:

```
lkd> .foreach /p 1 /ps 100 (place {!object \Ntfs} ) {dt nt!_device_object -y -a Driv->maj place }
+0x008 DriverObject :
+0x038 MajorFunction :
```

avast hook for fsdsetinfo call

Code:

```
lkd> uf 0xa952aa60
```

```
aswSP+0x26a60:
```

```
0a952aa60 8b ff mov edi,edi
```

hows dat fer braggin my 1337 windbag skillzzzzjj

edit

oops i hear @w going will ya wanna tell me what tis dat

it takes the DeviceObject pointer from Global NameSpace (\)

or you can find the driver object directly from \FileSystem\Ntfs also

Device object contains a pointer to DriverObject which lists the resolved address for all IRP_MJ's in an array

the pS ps crap is to cut off junk the work is done by dt -a switch to expand the array -y switch to input partial names

Kayaker

September 24th, 2013, 04:37

Lol, I knew that was coming.

So, Windbag master, now that you mention it, I'm curious if Avast also hooks ZwSetInformationFile, or instead it's doing it at the lower level IRP_MJ_SET_INFORMATION stage only, to presumably make it harder to anti-anti?

blabberer

September 24th, 2013, 05:07

you mean avast leaves anything unhooked ? nah impossible it hooks every imaginable function it hooks even if the function doesn't exist

```
lkd> r $t0 = 0 ; .foreach (plcae { !grep -i -e "asw" -c "dpS nt!KiServiceTable !?0x11c" } ) {r$t0 = @$t0+1 ; ? @$t0
Evaluate expression: 112 = 00000070
```

so it is hooking 0x70 out of 0x11c functions

thread hijacked 😊 split it will ya if someone posts kav hooks more than avast mcafee never hooks but overwrites the os

some error i hard counted and didnt see 112 hooks but 50 odd hooks still and this script matches it with !chkimg output

```
lkd> .shell -ci "dpS nt!KiServiceTable !?0x11c" grep -i asw | wc -l
57
.shell: Process exited
```

Code:

```
lkd> .shell -ci "dpS nt!KiServiceTable !?0x11c" grep -i asw | wc -l
```

```
57
```

deepzero

September 24th, 2013, 09:57

woah, can oyu explain how your lkd-kung-fu works?

I am guessing you somehow check in which module a given address lies, and if it contains "asw" it's an avast module? Really dont see that in the script, though...

blabberer

September 24th, 2013, 16:21

Quote:

```
woah, can oyu explain how your lkd-kung-fu works?
```

which form you want explained snake in monkeys shadow or cats paw ?

i guess ill smoke pot and trip the chair see if you can dodge

dpS = display data by de-referencing a pointer and resolving the symbol associated with the de-referenced pointer

how it works

lets find the address of nt!KiServiceTable

```
lkd> x nt!KiServiceTable
80501ba4 nt!KiServiceTable = <no type information> assuming you open this memory address in a hexeditor you will see
```

```
lkd> db nt!KiServiceTable !10
80501ba4 ba 99 59 80 36 6e 5e 80-94 a6 5e 80 68 6e 5e 80 ..Y.6n^...^..hn^.
```

that address is pointing to 0x805999ba

```
lkd> dd nt!KiServiceTable !1
80501ba4 805999ba
```

now if you open this memory address in hexeditor you will see

```
lkd> db 805999ba
805999ba 68 9c 00 00 00 00 68 40 91-4d 80 e8 67 e5 f9 ff 64 h....h@.M....d
```

wouldn't make much sense in hexeditor so you would have to resort to disassembling and seeing if this is code or you can guess this is code because 68 is opcode for push you see two 68 and one e8 for call push 9c / push 804d9140 call negative offset (fff9e567) so if this is code does it have name ? symbol ??

so you would need to do list nearest symbol ln

```
lkd> ln 805999ba
(805999ba) nt!NtAcceptConnectPort | (8059a0a8) nt!NtCompleteConnectPort
```

Exact matches:
nt!NtAcceptConnectPort = <no type information>

dpS does all this for you and will spit **nt!NtAcceptConnectPort** if you provide it the first address / symbol

```
lkd> dpS 80501ba4 l1
805999ba nt!NtAcceptConnectPort
```

```
lkd> dpS nt!KiServiceTable l1
805999ba nt!NtAcceptConnectPort
```

so when you do dpS on a range of Address each line of output will have the module name symbol name

now on an unhooked machine service table entries should all point to address in ntos/ntkern
 if the address points to an address that does not lie in ntos/ntkern then it must be hooked

and dpS dutifully would resolve the module name atleast + a bogus / exported nearest symbol like

someshitmodule!somecrapfunction+0xb0b15bad

like this

```
lkd> dpS nt!KiServiceTable+40 l3
8060bae6 nt!NtAllocateUids
a95115fa aswSP+0xd5fa
805a5a70 nt!NtAreMappedFilesTheSame
```

so service table entry 0x11 is hooked by one aswmodule

on an unhooked machine that would NtAllocateVirtualMemory

Code:

<http://j00ru.vexillium.org/ntapi/>

```
NtAllocateUids          0x0010    0x0010    0x0010    0x0010
NtAllocateVirtualMemory 0x0011    0x0011    0x0011    0x0011
```

what is the module information for aswxxxx

Code:

```
lkd> lm f a a95115fa
start end module name
```

so what is the script doing

Woodmann

September 24th, 2013, 22:13

Crane style ? 🙄

Woodmann

Sorry, I just had to.

blabberer

September 25th, 2013, 00:42

crane style ? oh possible i just edited the original posts output
 cranes can have big beek so it can reach farther
 it can now indicate the service function index on each hooked function

the number in front of the output is 1 based index service function entry is 0 based index so you need to minus 1 from each number to get the unhooked function

cranes can wait for long too till suitable fish comes by

Code:

```
lkd> .for (r $t0 =0; @$t0 < 0x11c*4; r $t0 = @$t0+4) { .if ( poi(nt!KiServiceTable+@$t0) > 806cf980) { .printf "Service index 0x%x is hooked by 0x%x %"
```

blabberer

September 25th, 2013, 12:17

@ k avast is hooking all these functions
 (i got bored looking up jooru metasploit etc for function names setup a clean vm and got the unaltered service function names from it to a txt file for comparison and tweaked the script above to use that file)

the tweaked script as follows

Code:

```
r $t1 =-1 ;foreach /ps 1 ( place { dpS nt!KiServiceTable l?0x11c } ) { r $t0 = place; r $t1 = @$t1 + 1; .if( @$t0 > 0x806cf980) { .printf "Service Function %"
```

the result as follows

Code:

```
lkd> $$>a< .\scripts\print_hooked_ssdt_functions.txt
Service Function 00000009 is hooked by a945d610 aswSnx+0x18610 (a945d610)
Service Function 00000017 is hooked by a95115fa aswSP+0xd5fa (a95115fa)
```

WaxfordSqueers

February 4th, 2014, 08:14

Quote:

[Originally Posted by blabberer;95454]i still dont believe shell and ole ole plays significant role in the process

Sorry I haven't been around for a while. I have intended to find out more about NTFS but matters beyond my control have intervened. Hopefully I'll get back to it sometime.

I have a healthy 500 gig drive tied up because I want to use it to inspect the damage. I have decided the NTFS file system cannot be recovered so I'm going to reformat the drive. The reason I think it can't be recovered is because most of the files that make up the MFT have been overwritten. The index files are all there, or seem to be.

About shell and ole. The reason I think they are involved is that I have traced right through to NTFS.sys from Windows Explorer and another file manager. Using softice, I have traced from Explorer through shell32 to ntfs.sys and back to shell32. I can see references to shell32 and ntfs.sys in the stack with return addresses from ntfs.sys right back to shell32.

As you know, Explorer is based on a namespace system that uses its own file record structures, called ITEMID structures. Each file in an NTFS system is identified in Explorer with one structure per file. Those structures have to lead to the NTFS system somehow and shell32 is the shell manager that handles those structures.

I could be wrong but it makes sense to me that shell32 calls into the NTFS structure somehow. A structure in shell 32 is called early on which contains information about the file being loaded. The structure tells the system whether the file is an executable, or whatever type of file it is. From there, the code goes deep into shell32, ole32, shlwapi with seemingly endless parsing of the file ITEMID structure.

There may be some reference somewhere in that structure to the MFT.

I have set breakpoints while in Explorer to break before shell32, then traced into shell32. I have also traced to just after the Explorer.exe entry point in U32 then set a breakpoint for a function in ntfs.sys.

If I look at the stack along the way I can see shell32 listed as a return address.

Unfortunately, due to the depth of the nesting, and the seemingly endless checks on file structure after file structure in shell32, I have lost my way. I can set a breakpoint on a file and get to ntfs.sys no problem, it's just that when I get there the file seems to already have been found.

If I have some spare time and a clear enough mind, I may just try it with more tenacity sometime. There are times when I get lost that I take shortcuts to get back to a point in the code and I think I am jumping over important code that calls into ntfs before the disk is accessed.

Sometimes I get lost in win32.sys which is a major pain to get out of. Other times I get caught in waitforsingleobject, etc., and at other times I get caught in the code of other apps. As Kayaker pointed out, in a VM, it's easy to get caught in the VM code.

Ntfs.sys is surprisingly easy to trace through once you're in it, but there are times when it returns all the way back to an address in shell32 then all the way back to ntfs.sys.

WaxfordSqueers

February 5th, 2014, 05:39

Quote:

[Originally Posted by WaxfordSqueers;96094] I can see references to shell32 and ntfs.sys in the stack with return addresses from ntfs.sys right back to shell32.

Here's a handwritten copy of the stack I made while tracing a while back. It's not complete with regard to all return addresses. Those addresses shown are return addresses of the function in the following line.

I put a BP on _NtfsReadFileRecord@28 and when it broke, this was most of the stack:

```
F73109E2 Ntfs!_NtfsReadFileRecord@28
Ntfs!_FindFirstIndexEntry@6
Ntfs!_loc_3954A
Ntfs!_loc_396D4
Ntfs!_loc_382E8
804E37F7 Ntfs!_loc_38248
Ntoskrnl!IoCallDriver
F73A306B Ntoskrnl!IoCallDriver
fltmgr
8057216E Ntoskrnl!IoCallDriver
804DE7Ec Ntoskrnl!NtQueryDirectoryFile
Ntoskrnl!ZwYieldExecution
Ntdll!KiFastSystemCallReturn
K32!FindFirstFile
77F742F9 K32!GetLongPathNameW
7E1EA64D Shlwapi!.text + 132F9
7E1EA56D urlmon!.text + 9646
urlmon
urlmon
urlmon
urlmon
urlmon
77F74185 urlmon!.text + 2AA3
77F740EF Shlwapi!.text + 13185
7CA04C37 Shlwapi!.text + 130EF
7CA04173 Shell32!CRegFolder::GetOverlayIndex
7CA040FA Shell32!CInfoTip::AddRef
7CA03071 Shell32!CShellCmdFileIcon::'vtable'
7CA02FCE Shell32!CDefView::FindItem
7CA02F6A Shell32!CFSFolder::HandlerCreateInstance
7CA0F71B Shell32!CDragImages::Show
7CA0F5A7 Shell32!CShellLink::_ShortRetTimeout
Shell32!CShellLink::_ShortRetTimeout
Shell32!CShellLink::_UpdateIconFrom.....
7CAA5D63 Shell32!CShellLink::_UpdateIconFrom.....
7CAAAAFc Shell32!CDUIView::Release
```


The functions shown are generalized areas in the DLLs and don't necessarily indicate the exact function being executed. It's plain to see, however, that Shell32 leads to NTFS.sys.

I am still not sure where in the file location process this takes place. It seems to me the file I am looking for was found before this code appeared.

Note that the last call came from shlwapi which is the Shell lite version. Then there's a call into kernel32 to GetLongPathNameW and FindFirstFile.

This could be a search for the file date/time info required at a later time, after the file has been located.

It should be noted that I traced this code initially from Windows file explorer by setting a BP on a window in Explorer and double-clicking the file I wanted to load. The BP breaks in the Explorer process then runs through shell32, shlwapi and ole32.

I have a stack trace somewhere of the beginning of that process if it interests anyone. The initial stack info is concerned more with identifying the window clicked in Explorer and checking lparam and wparam in the message.

WaxfordSqueers

March 15th, 2014, 09:00

Quote:

[Originally Posted by WaxfordSqueers;96101]Here's a handwritten copy of the stack I made while tracing a while back.

Have not done anything important but made a small discovery related to the MFT.

The hard drive on my laptop had been going since 2009 with all sorts of file activity. As a result, the MFT had become split into 2 sections with the two dataruns sequences clearly indicated in the \$MFT file near the end of the \$80 attribute. I have since upgraded the hard drive to a larger drive and cloned the old OS to the larger drive, complete with split MFT.

I tried to defrag the MFT using Contig from Sysinternals and an older version of Diskeeper, which I bought years ago. Contig claimed the MFT was defragmented but it left it in two pieces. Same with Diskeeper, even though it has the ability to defrag the MFT during a reboot.

I had read that Raxco Perfectdisk had the ability to defrag the MFT as well so I d/led the 30 day trial for version 13. The grid of the disk it supplies has the MFT highlighted in yellow. I let it analyze the disk and it found the disk defragged adequately. However, when I ran a defrag for the heck of it, Perfectdisk did an optimization that took more than an hour.

During the optimization, I watched it move the squares marked as MFT. However I knew my MFT was located at cluster 0xC0000 and after the defrag it was still there. The surprising thing was that SMFT now showed the MFT as being in one piece. It appears as if Perfectdisk did actually affect it significantly, although \$MFT was in exactly the same cluster space at 0xC0000.

I am not pushing Perfectdisk but I am thinking of buying it based on the fact it did what it claimed it would do and it provides a grid one can watch as it works. I have just begun the 30 day trial so I'll keep an eye on it to see if there is any instability, or interference with the OS or other apps.

WaxfordSqueers

March 15th, 2014, 09:08

Quote:

[Originally Posted by Kayaker;95468]Lol, I knew that was coming.

So, Windbag master, now that you mention it, I'm curious if Avast also hooks ZwSetInformationFile, or instead it's doing it at the lower level IRP_MJ_SET_INFORMATION stage only, to presumably make it harder to anti-anti?

I know this comment is old but it just so happens I was messing with gmer the other day and it shows everything that is hooked under the 'Rootkit' tab. I'm sure you have more sophisticated ways of finding hooks but gmer seems to do a good job.

At least, it shows all the hooks if you leave everything checked to the right of the main screen.

WaxfordSqueers

March 15th, 2014, 09:16

Quote:

[Originally Posted by blabberer;95469]so it is hooking 0x70 out of 0x11c functions

That's the problem with most antivirus apps, they follow you around like a rent-a-cop tailing a teenager in a department store and interfere everywhere they can. It's a small wonder any modern OS operates at all.

When I was tracing the NTFS on a VM I found it easier to unload the antivirus and stay offline. But then I had to worry about getting caught up in the VM code. Or getting shanghied by a WaitForSingleObject type trap.

I still don't know whether to trace through such an event or try jumping over it.

WaxfordSqueers

March 15th, 2014, 10:22

Quote:

[Originally Posted by blabberer;95454]i still dont believe shell and ole ole plays significant role in the process the object is created in R0 with ObCreateObjName & sisters and the handle is created with obpCreateObHandle and brothers does shell and ole send some undocumented DeviceIoControl to ntfs.sys

I know this is getting dated Blabs but I was thinking of opening up my investigation again, albeit temporarily, and I was reviewing the structures I know I'll encounter if I BPX on a file in explorer. Came across this concise but dated explanation of the process:

http://netez.com/2xExplorer/shellFAQ/bg_shell.html

The author explains that parsing a file path had to be extended due to the inclusion of objects that were not file objects, like printers, URLs, and virtual objects like the Desktop, Recycle Bin, etc. For that reason, different objects were given the ability to be opened by conventional means in a file manager. The means of opening them, executing them, printing them, etc., are referred to as verbs, which are defined in the registry. Also, you will find verbs as a parameter in ShellExecuteEx, the main function for processing the paths. As you trace through the code, you encounter Urlmon, which is related to processing URLs.

Of course, when you try to trace through this stuff, it is not going to process just the stuff you want, like opening an executable file. It seems to check and test for every conceivable object that can be opened by a file manager.

In the article, they stick to the convention of referring to an Item ID List, an IDL, as a PIDL, which is totally screwed. Even Microsoft does it. A PIDL is a POINTER to an IDL and a pointer is an address, not a list. To me, it is exceedingly sloppy to use the acronym PIDL in reference to an IDL, especially when you have already claimed that an item ID list is an IDL. A PIDL should be the location at which the IDL is found, not the list itself.

Although Shell and Ole may not play direct roles in the process, you have to get through their code unless you know where to go on the other side. I have tried jumping over the code to reach functions in NTFS.sys only to find the file has already been opened.

I am guessing that somewhere in an IDL, there is a reference to the MFT as an object. In fact, I seem to recall such a reference. Perhaps the MFT is being accessed early in the process as a file object, via Shell and Ole long before NTFS.sys comes into it.

Also, as I have pointed out in the past, there are intervening processes, depending on whether the file has already been moved to a cache. That includes the MFT or parts of it. During initial access to the disk volume, the MFT is moved to a filecache. Of course, it doesn't seem to move the entire MFT to a cache, so even if you end up in a cache, eventually it should go back to disk in order to refill the cache.

To complicate matters even more, Windows has different levels of caching, like WriteBehind caches. It also has search indexing. NTFS also writes to logfiles that are used by Chkdsk to rebuild the MFT should a power fail or crash happen before data from the WriteBehind cache can be written to disk. That's why I turn off that feature on a hard drive, especially when I am tracing.

I have already experienced a BSOD while tracing after which my system would not boot. However, running Chkdsk restored the system.

It's little wonder that mere mortals can make head or tail of the NTFS, especially in the deep code woods.

blabberer

March 18th, 2014, 02:11

@waxf
wasnt around so couldnt reply early tx for the link let me check it.

and if you want a collaborative trial setup a vm with say xp-sp3 and hook up windbg to it.
that way we can both go through the same path and see the same things albeit with different eyes. and others may join in too because of reproducibility in thier end at leisure

suppose you want to trace file creation through ntfs of a file name willy.billy.magicext

we can start from fopen(willybilly.magicext,"wb" or rather oldschoolish copy con blah_blah.foo ctrl+z or go with the youth of today by
'loo'sing shitstem.dognasties

WaxfordSqueers

March 18th, 2014, 19:52

Quote:

[Originally Posted by blabberer;96241]and if you want a collaborative trial setup a vm with say xp-sp3 and hook up windbg to it.

Just so I don't have to re-invent the wheel:

- 1)am I hooking windbg up so it runs in XP3 or in the VM?
- 2)If I get a pipe set up between XP3 and VM, how am I initiating the app...say notepad? Do I start the app in windbg?

I think it's better to use an exe file because other apps like a txt file will require opening up an intermediate app to run it.

I had the basics of windbg down a year ago but i have not tried it as of late and rust has set in.

We could start with a BP on fopen but I'm pretty sure the app will be open by then.

WaxfordSqueers

March 18th, 2014, 21:09

Quote:

[Originally Posted by blabberer;96241]... hook up windbg to it.

another question...I have configured the VM as \\.\pipe\com_1, This end is server, The other end is an application (should that be a virtual machine??) and Yield CPU on poll.

I set the boot.ini file up in the VM version of XP with "Debug" /fastdetect /debugport=com1 /baudrate=115200

When I boot into the VM now using the Debug selection in the boot menu, the mouse is acting up. It stays as an hour glass then disappears.

Have a feeling the mouse is using com1, even though it is a USB mouse. I wonder if the boot.ini should be changed to debugport=com_1 or the VM setting should be changed to \\.\pipe\com1

Kayaker

March 19th, 2014, 05:25

Do yourself a favor W and use VirtualKD to set it all up. I just reset everything up on my x64 system -> XP VMWare and it took like 2 seconds.

I forget how to start a user app remotely though, this did nothing

```
>.create c:\windows\system32\notepad.exe
Create will proceed with next execution
> g
```

blabberer

March 19th, 2014, 14:39

boot . ini in vm os /debug /debugport=com1 /baudrate=XXXX

virtual maching settings

remove the printer service which normally hogs com port 1 in vmware iirc (atleast i remember vmware player 4.01 hogging serial port 1)
add serial port using named pipe leave all the defaults iirc \\.\pipe\com_1 select this end is server / other end is application

i dont know what diffrence it makes by ticking yield cpu (it is not ticked by default and i leave it at default)

open windbg in host with

windbg.exe -k com_ipe,port=\\.\pipe\com_1,reset=0,reconnect

you should have a valid connection (i dont remember any problems with mouses disappearing probably the clock struck one and hickory dickory dock ran down

WaxfordSqueers

March 20th, 2014, 09:21

Quote:

[Originally Posted by Kayaker;96248]Do yourself a favor W and use VirtualKD to set it all up.

Thanks for tip Kayaker. After spending multi hours trying the conventional approach, and getting nowhere, I tried Virtual KD. No better luck. I'm beginning to feel like Eagle-eyed Fleagle, walking around with a perpetual black cloud over my head. That might be a step up from Fearless Fosdick, who usually had a cannonball hole in his chest, but it didn't seem to bother him.

I got everything set up ok but when my vm boots, it freezes immediately rather than at the windows icon they suggest. Here's the virtualKD log:

```
*****
*VirtualKD patcher DLL successfully loaded. Patching the GuestRPC mechanism...*
```

Searching patch database for information about current executable...

No information found.

Waiting for VMWare to initialize (5954 ms more to wait)

Analyzing VMWARE-VMX executable...

Building list of EXE sections... 11687K of data found.

Scanning for RPC command name strings...

Finished scanning. Found 41 strings.

Searching for string references...

Found 27 string references.

Found 1 structures resemblant to RPC dispatcher table.

(address 00B0EBE0, matched pointers: 27)

Analyzing potential RPC dispatcher tables...

Potential RPC table analyzis complete. Found 1 candidates.

(address 00B0E640, entries: 74, free entries: 7)

Using RPC dispatcher table at 0x00B0E640 (74 entries)

Waiting for RPC table to be initialized by VMWare...

RPC table initialized. Patching it...

Successfully patched entry #1

VMWare reset monitor activated...

and here's the windbg log:

Microsoft (R) Windows Debugger Version 6.11.0001.404 X86

Copyright (c) Microsoft Corporation. All rights reserved.

Opened \\.\pipe\kd_Windows_XP_Professional

Waiting to reconnect...

Connected to Windows XP 2600 x86 compatible target at (Thu Mar 20 04:59:36.562 2014 (GMT-8)), ptr64 FALSE

Kernel Debugger connection established. (Initial Breakpoint requested)

Symbol search path is: f:\winxp\symbols;srv*f:\winxp\symbols*http://msdl.microsoft.com/download/symbols

Executable search path is:

Windows XP Kernel Version 2600 UP Free x86 compatible

Built by: 2600.xpsp.080413-2111

Machine Name:

Kernel base = 0x804d7000 PsLoadedModuleList = 0x8055b1c0

System Uptime: not available

Break instruction exception - code 80000003 (first chance)

* *

* You are seeing this message because you pressed either *

* CTRL+C (if you run kd.exe) or, *

* CTRL+BREAK (if you run WinDBG), *

* on your debugger machine's keyboard. *

* *

* THIS IS NOT A BUG OR A SYSTEM CRASH *

* *

* If you did not intend to break into the debugger, press the "g" key, then *

* press the "Enter" key now. This message might immediately reappear. If it *

* does, press "g" and "Enter" again. *

* *

nt!RtlpBreakWithStatusInstruction:

804e3592 cc int 3

I did not press CTRL -C or CTRL + Break.

I ran the app mentioned in the instructions inside the VM and they indicated everything was ok. Then I shut the VM down and booted the Virtual machine Monitor from the host. I tried various ways of starting the vm again but none of them worked. I can see a 'yes' under OS and Debugger but the vm just freezes.

WaxfordSqueers

March 20th, 2014, 09:37

Quote:

*[Originally Posted by blabberer;96249]virtual maching settings
remove the printer service which normally hogs com port 1 in wmware iirc (atleast i remember vmware player 4.01 hogging serial port 1)
add serial port using named pipe leave all the defaults iirc \\.\pipe\com_1 select this end is server / other end is application*

thanks for help, blabs. I removed the printer service, no dice. If the player was an issue, I don't think the vm would boot with a regular start up never mind debug. I'll keep it in mind.

The thing that really bugs me is setting com1 (serial0) on a pipe then setting the boot.ini file to com1 for debugport. If windbg is using com1 for a pipe how does windoze in the vm use it? I tried setting it to com2 and I even configured the vm for 4 serial ports but no luck. I say that because com1 on the vm is supposed to be mapped to the com1 irq/mem address of the host.

Somehow the mouse is getting onto com1 even though it's on a USB port on the host machine. It looks to me like an old-style IRQ conflict but I don't see how the mouse is getting onto irq 4 where com1 is located.

Mind you, the vm insists on loading both the PS/2 mouse and the HID variety via USB. It wont let me uninstal the ps/2 driver.

Part of the problem may be that my Intel mobo only supplies one com port.

Kayaker

March 20th, 2014, 09:59

Hmm, it *sounds* like you were 100% successful. That's exactly what I get. Now you should be under Windbg control but need to press F5 (g) to allow the VM to continue loading. Soon you should be able to interact with the VM. Back in Windbg, issue Debug/Break and the VM will "freeze" again but you can use Windbg commands, i.e. !process 0 0 should give a list of all VM processes running.

WaxfordSqueers

March 20th, 2014, 13:19

Quote:

[Originally Posted by Kayaker;96252]Now you should be under Windbg control but need to press F5 (g)....Back in Windbg, issue Debug/Break and the VM will "freeze" again but you can use Windbg commands, i.e. !process 0 0 should give a list of all VM processes running.

Sheer genius, kayaker...as usual. No one mentioned F5 and I'm too stupid to remember what blabs told me about it.

Did the F5 and the vm continued loading. Waited till it was fully booted and windbg changed from busy. Did a cntl - break but did not notice anything. When I did the !process 0 0 it gave me the list.

The vm just sits there, however, with an hour glass where the mouse cursor is. If I move it, the cursor disappears.

I was at the same point with the old method so maybe all I had to do was boot windbg and follow the same procedure. I'll try it.

I had the notion that I needed to use the vm. I presume I go into windbg and issue an attach to process but I don't have notepad running in the vm and there's no way to start it till I sort out the mouse issue.

Need to spend more time with it.

blabberer

March 20th, 2014, 14:15

f5 = g == go same from good old dos days old debug.com uses g = go

Code:

```
C:\WINDOWS\system32>debug command.com
```

windbg printed that for you when it broke and asked you to do it 😊 if you didnt intend to break didnt it ?

anyway now that you are into windbg after you pressed f5 or typed g and the vm booted up fully

if you intend to break again press ctrl+break

when broken when you want to run the target again type g and hit enter or press f5

also note what you are now into is kernel debugging not your usual user mode debugging

you do not attach to process from here (only remote debugging will let you either create or attach to process and that is a different beast

i noticed kayaker doing .create

no .create or .attach wont work in this setup

(this needs -premove option with debugging tools running on both machines one as a server and other as a client or a dbgsrv.exe or remote.exe in commandline)

in this setup you need to set breakpoints and start the process in vm for windbg to break on process creation

for example

do ctrl+break in windbg

type bp nt!NtCreateProcessEx ; g and hit enter

now in vm click notepad.exe and windbg should break again

WaxfordSqueers

March 20th, 2014, 16:24

Quote:

[Originally Posted by blabberer;96255]windbg printed that for you when it broke and asked you to do it 😊 if you didnt intend to break didnt it ?

mighty decent of it, I'd say.

Quote:

[Originally Posted by blabberer;96255]now in vm click notepad.exe and windbg should break again

I can't get into vm and I have to find out why. I had the same issue with softice if I tried to start in debug mode. Don't ask why I wanted to do that...just curious.

I have several things to investigate. For one, my bios can be updated and there may be things in there that are affecting the overall operation. Right now, I am trying to foolproof the bios update using a USB thumb drive but that's another experience by itself.

blabberer

March 20th, 2014, 19:28

Quote:

I had the same issue with softice if I tried to start in debug mode.

do i infer that you have softice installed in the vm ? and you have connected teh vm to windbg too ?

if so ctrl+break will be caught by softice

you either need a neat and clean vm or you have to muck with softice **i3here off**

WaxfordSqueers

March 20th, 2014, 20:20

Quote:

[Originally Posted by blabberer;96258]do i infer that you have softice installed in the vm

I don't run softice from boot, I start it once I get to the desktop. I double-click it to run it and it goes through it's initialization and loads all it's crap. It's driver is loaded and that may be creating problems. Kayaker would know more.

I don't think it would be trapping anything but I could clone my present system and remove SI. Driverstudio is another matter. I have never used it and have no idea what its drivers are up to.

My problems are occurring before windbg is started. Before I loaded the app Kayaker suggested I was trying to launch kernel debug mode and that's when the mouse started to freeze. I thought it might be due to the statements required in the vmx file to let SI run but defeating them did nothing.

I think my problem is a conflict between the mouse and com1 but I will definitely look into the SI matter.

Thanks.

WaxfordSqueers

March 22nd, 2014, 08:20

Quote:

[Originally Posted by blabberer;96258]do i infer that you have softice installed in the vm ?

Cloned the VM amd completely removed SI. Still having mouse trouble.

Mouse seems OK till desktop appears, then within a few secs an hourglass replaces arrow. Any movement of cursor causes it to disappear.

Noted that when I press g in windbg, I get the following:

```
kd> g
ERROR: DavReadRegistryValues/RegQueryValueExW(4). WStatus = 5
ERROR: DavReadRegistryValues/RegQueryValueExW(5). WStatus = 5
ERROR: DavReadRegistryValues/RegQueryValueExW(6). WStatus = 5
Single step exception - code 80000004 (first chance)
First chance exceptions are reported before any exception handling.
This exception may be expected and handled.
001b:006cdad2 8d55b8 lea edx,[ebp-48h]
```

It seems to be indicating an exception and I wonder if it is related.

WaxfordSqueers

March 22nd, 2014, 17:02

Quote:

[Originally Posted by blabberer;96258]you either need a neat and clean vm or you have to muck with softice **i3here off**

Came across a fresh XP vmx file I forgot I had. Loaded Virtual KD and things seems to have cleared up, even though it's loaded with SI.

I'm currently sitting with an XP desktop with a live mouse and windbg seems to be talking to it. I don't have a clue how to proceed and I'm too tired to figure it out at the moment. The windbg window says nt!RtlpBreakWithStatusInstruction followed by 804e3592 cc int 3. That's because I issued a ctrl - break.

I tried tlist | grep explorer, as you suggested in a past thread, but it gets snarky right off, claiming ' ^pass count must be preceeded by whitespace error in 'tlist | grep explorer'.

I described my SI procedure to you. I use SPYXX to find a hwnd in explorer.exe that represents the window holding the notepad.exe reference in explorer. Then I 'bmsg hwnd 0x201' in SI which sets a BP on the hwnd if it's window is double-clicked, 0x201 being the wmsg for double-click.

SI breaks in explorer code then returns to u32 code. From the explorer code I can BP on anything I want so long as I know the context and the function, provided the functions is known to SI. A good bp is the shellexecutex function described in that link I posted. It has a structure that describes what is being done to the file being loaded.

What I can't figure out at this time is how to begin.

I am proposing a means of attack but you were suggesting going straight to fopen(). I think that's a waste of time but I'm willing to try it. I think you'll find the file is already open.

By definition fopen() "Opens the file whose name is specified in the parameter filename and associates it with a stream that can be identified in future operations by the FILE pointer returned". Createfile() does the same thing. The trick is to find the stream and that search seems to take place long before createfile or fopen comes into play.

What did you have in mind? You're going to be way ahead of me till I get up to speed but I'd like to try getting the hang of your modus operandi as it applies to windbg.

blabberer

March 22nd, 2014, 17:07

when windbg shows that what does the prompt look like ?

does it show kd
or is it grayed out and busy running ?

if it is kd type g and press enter

that exception seems to show an address from usermode 1b

blabberer

March 22nd, 2014, 18:19

here are three text files that shows the round trip for creation of a txt file named willy.txt that does not exist prior

modus operandi

start run -> type notepad willy.txt and press ok
since the file doesnt exist notepad will ask you whether you want to create it new
you want to create it but before that you want to trace it so

set a breakpoint on nt!IoCreateFile hit g

now in vm say yes to the msgbox

if there are spurious breaks in IoCreateFile without notepad.exe in stack ignore them

you should see a break where stack should look like this

Code:

```
kd> bl
0 e 8056ae1c 0001 (0001) nt!IoCreateFile
```

further tracing should get you to into sr!SrCreate which checks for ADS (advanced data stream) and calls the lower / upper driver

Code:

```
kd> kb
ChildEBP RetAddr  Args to Child
fb3b7a24 80578616 ff458da8 ff2f1dd0 fb3b7c04 nt!IoFCallDriver+0x2d
```

here you enter ntfs

Code:

```

nt!IoPfdDriver+0x2d:
804e0115 ff548638      call     dword ptr [esi+eax*4+38h] ds:0023:80f7a068=Ntfs!NtfsFsdCreate (fc20be01)

```

here is the relevant stack

Code:

```

Ntfs!NtfsCommonCreate+0x172:
fc209cd8 8b4830      mov     ecx, dword ptr [eax+30h] ds:0023:ff323bf0=00380012

```

commoncreate should call

call Ntfs!NtfsCreateNewFile

Code:

```

kd> kb
ChildEBP RetAddr  Args to Child

```

the stack on this call

this function takes 20 arguments

Code:

```

kd> .fnent @eip
Exact matches:

```

full stack on this call

Code:

```

kd> dd esp
fb7af684 fc20eb32 80fdae00 ff3d68f8 ff3d6a40
fb7af694 e101f9f0 e1895f58 005e005e e1076ed0

```

some args to this function are

Code:

```

kd> du e1076ed0
e1076ed0  "\\Documents and Settings\Administ"
e1076ef0  "c:\program files\windows\system32\cmd.exe"

```

this should allocate the mft record

fc219d70 e8e2ecffff call Ntfs!NtfsAllocateMftRecord (fc218a57)

Code:

```

kd> kb
ChildEBP RetAddr  Args to Child

```

further up

Code:

```
kd> kb
```

```
ChildEBP RetAddr  Args to Child
```

further up should lead to MftReadFileInRecord () MftCheckAttributeRecord finally allot and return back

Code:

```
eax=81027cc0 ebx=00000000 ecx=00000000 edx=80fdae0 esi=81027100 edi=80fdae0
eip=fc718ad5 esp=fb7af434 ebp=fb7af4a0 iopl=0         ry up pi op pz ac pe pf
```

the full set of function call grepped and sorted for a round trip should be like this

Code:

```
8056ae23 e8e8d0fcff  call  nt!_SEH_prolog (80537f10)
8056ae4e e82da0dfff  call  nt!ExAllocatePoolWithTag (80544e80)
```

here are the files

WaxfordSqueers

March 23rd, 2014, 20:56

Quote:

```
[Originally Posted by blabberer;96266]set a breakpoint on nt!IopCreateFile hit g ...now in vm say yes to the msgbox
```

Frustrating stuff. There's something about windbg that is not user friendly or obvious. I did not have near this kind of problem using and learning SI.

Of course, I am using virtualKD, which I am just about to scrap. Sorry Kayaker.

I have gotten to the point where I am looking at the notepad window with the message box saying it cannot find the file...do i want to create it. When I set the BP in windbg, I had to type bp nt!IopCreateFile in the kd> box then I hit enter...which you forgot to mention.

Immediately, it gave me Breakpoint 0 hit....nt!IopCreateFile. How did it go off on a BP when I took no action? Now I can't press the OK button on the notepad messagebox because the cursor has become a permanent hourglass.

I have seen the same thing in SI when I set a BP, which went off before I could activate an OK button. There's no reason it should go off on CreateFile, however. That's the reason I scrapped a direct BP on CreateFile in SI and used my indirect method of getting into the code using message BP on a hwnd.

I just think this entire method of a debugger requiring a pipe, or a serial port is the heights of hoakiness. I can understand it for certain drivers for a display, or whatever, but this anal crap about requiring it in general is so typical of microsoft.

Kayaker

March 23rd, 2014, 22:06

Context, context, context.

Get the current context when IopCreateFile breaks indiscriminately like that and you'll probably find it's vmtools.

```
kd> !process -1 0
```

Instead set a context specific breakpoint with the /p Eprocess switch

```
!process 0 0
```

```
PROCESS 82306b48 Image: notepad.exe
```

Here I set the bp by changing the context and using the \$proc pseudo register

```
kd> .process 82306b48
```

```
Implicit process is now 82306b48
```

```
kd> bp /p @$proc nt!IopCreateFile
```

```
kd> bl
```

```
0 e 8057c084 0001 (0001) nt!IopCreateFile
```

```
Match process data 82306b48
```

```
kd> g
```

```
Breakpoint 0 hit
```

```
nt!IopCreateFile:
```

```
8057c084 6a3c push 3Ch
```

```
kd> !process -1 0
```

```
PROCESS 82306b48 Image: notepad.exe
```

Following blabberers footsteps:

Code:

```
kd> kb
```

```
ChildEBP RetAddr  Args to Child
```

```
b2324c94 8057c31d 0007fe0c c0100080 0007fdac nt!IopCreateFile
```

```
b2324c94 8057c360 0007fe0c c0100080 0007fdac nt!IopCreateFile
```

I don't care if you don't use VirtualKD. But "45x Faster Windows Kernel debugging with Virtual Machines." is good enough for me. 😊

blabberer

March 24th, 2014, 01:19

if you scrap virtual kd you will be sitting there for hours waiting for windbg to return back from a single step with registers

pipe debugging turns of register dumping by default and will still be too slow for a saints patience.

instead of cribbing about wife however obtuse she is try to pretend to live with her atleast your dreams will be sweet.

i wrote ignore spurious breaks (they are not spurious perse but spurious for your context)

trash what you are doing sit back relaxed with a coffee unlearn all the shit you have learnt till now and start from scratch.

do you agree doing things serially is far far slower than processing parallely expecially when the queue is very long (millions or billions of requests per time frame)

do you agree in parallel processing doing things asynchronously is faster and profitable overall than synchronous operation ?? (requests that are sent by queen elizabeth are processed first request by waxford with a recommendation from angela merkel gets processed earlier than request from plain citizen waxford) ??

queen elizabeths request comes directly to the manager while citizen waxford has to deal with the grouchy security first and the wax smile pasted angry lady next just to get a form let alone transact

IopCreatefile is a processor it can process only one request at any given time (it doesnt care if the request came from queen elizabeth or if the impatient waxford is requesting it it knows it got a valid request and will process it and wait for next

the manager / the security / the receptionist / the others / all in tandem decide which request to send first to IopCreateFile()

and they all know the request is a winning request that travelled across the channel and reached france and there can be no request for the next 13 months. or all request for the next 13 months will be simply rejected however important they are.

the request sent by notepad is very low priority request

there can always be several requests of higher priority that can preempt the request by notepad.exe the request by notepad will be processed when there are no request to process that have a higher priority than notepad.

that is how windows works and that is how windbg works

you need to be in the context / context / context / context of what you doing for you to transact.

yes there can be several breaks before your notpad gets created and several breaks after your notepad leaves IopCreateFile into unknown uncharted territory

you need to be able to deal with them all

once you get in the right stack frame you need to disable all the breaks

you should have at the most ZERO breaks or at the MOSTEST ONE one shot break further up your chain.

if you leave the break on IopCreatfile as it is and merrily step away

you will break thousands of time when you press p (single step)

because a single step command to reach from windbg in host to the right agent in the targets kernel that will tell the processor to step forward one step in the waxford/notepads.open,context thread may take anything from several nanoseconds to several minutes depending on umpteen factors.

and the kernel isnt going to sit idle waiting for the limping notepad from wax to come and enjoy its holiday in miami beach.

it will and shall be processing trillions (well a bit exaggerated but millions for sure) requests and some of them may be queens notepad with secret encryted messages that are sent to a router that credits billions of pounds into an unnumbered cayman island bank and yes queen will get her billion pounds earlier than your request for 10 pound even though you applied to get your pound years earlier than queen.

thats life and that how you have to pretend to live whether you like it or not whether you love it or not whether you are sorry about it or not.

WaxfordSqueers

March 24th, 2014, 11:39

Quote:

[Originally Posted by Kayaker;96268]Context, context, context.

*Get the current context when IopCreateFile breaks indiscriminately like that and you'll probably find it's vmtools.
kd> !process -1 0*

To put things in a different context, you probably know me well enough by now to know that I was venting due to a bad hair experience.

The problem is, K., that my mouse froze with an hourglass, making both windbg and my vm unresponsive. That's what I was p/o'd about. I have spent hour after hour troubleshooting this stuff and I am not a newbie to com ports, IRQs, etc.

I still don't understand why the VM serial port is set to a pipe and boot.ini is set to com1. I have seen some people suggest it should be set to com2. I have not spent enough time reasoning this out. windbg is obviously using a feature in the PCI bus controller to route data serially through a port (pipe) which the VM connects to on the other end. But the OS in the VM needs to access that data from the VM and it is using com1.

So, how does the VM connect to windbg on serial port 0 via a pipe, then connect to the VM OS on com1? VM serial port 0 should be com1 in normal circumstances. In fact, that's the option supplied if it is not configured as a pipe.

I spent the better part of yesterday bringing my XP SP3 updates up to date. Then I updated my mobo BIOS. I wanted to eliminate any problems the OS might be contributing. I unloaded DS3.2 with SI and I am running with a bare bones VM.

It's seems that virtualKD only adds an entry to the vmx file and adds a line to boot.ini in the VM OS. It just occurred to me that I might still have the mods to the vmx config file set to the settings required by SI. I'll need to look at that. SI used vmouse.present statement and I may be having issues with the mouse being confused. Nothing worse than a confused mouse.

Last night when I posted my vent, I was beyond caring. virtual KD had just started behaving erratically for no known reason.

Quote:

[Originally Posted by Kayaker;96268]Instead set a context specific breakpoint with the /p Eprocess switch

I appreciate this advice but I need to get the setup stable first. SI did the same thing when I set a BP on createfile but it did not freeze up. I could get back in with a ctrl-D and change the BP. Can't do that here. Once that hourglass cursor appears there's no way in or out.

Quote:

[Originally Posted by Kayaker;96268]I don't care if you don't use VirtualKD. But "45x Faster Windows Kernel debugging with Virtual Machines." is good enough for me. 😊

Hopefully you did not take my comment as a shot. I was apologizing because you recommended it. A night's sleep puts a different slant on things.

WaxfordSqueers

March 24th, 2014, 13:57

Quote:

[Originally Posted by blabberer;96269]if you scrap virtual kd you will be sitting there for hours waiting for windbg to return back from a single step with registers

I'm saying this with humour, so you wont lose it, but you are in fact admitting that vmware is a slow piece of crap. 🙄🙄

Only Microsoft could have devised a debugger like this but we are stuck with it since Numega decided to dissolve. So, let's make the best of it, right?

Quote:

[Originally Posted by blabberer;96269]pipe debugging turns of register dumping by default and will still be too slow for a saints patience.

I was reading up on pipes just to refresh rusted parts of my brain. They are implemented as FIFO buffers in the software rendition of serial communication. In normal serial hardware, FIFO buffers are used to store serial bits before sending. Interrupts are used by peripherals like serial ports to tell the CPU when service is required. The CPU controls the data flow using an RTS line, which it turns negative to tell the comm equipment to take a break.

VMs implement pipes as software buffers. So windbg is communicating with a buffer set up by the VM. Or. if the VM wants to set it's serial0 (com1) port to use the com1 port on the hosts hardware, it can do that. Unfortunately, I think windbg requires a null modem connection which I don't think a VM can supply, hence the use of pipes.

Stop me anytime you disagree.

Quote:

[Originally Posted by blabberer;96269]instead of cribbing about wife however obtuse she is try to pretend to live with her atleast your dreams will be sweet.

many wives specialize in cribbing so why should I not crib when said wife does not perform in an ideal manner? After all, everything SHOULD be perfect, should it not?

Quote:

*[Originally Posted by blabberer;96269]i wrote ignore spurious breaks (they are not spurious perse but spurious for your context)
trash what you are doing sit back relaxed with a coffee unlearn all the shit you have learnt till now and start from scratch.*

Ah...so you've done this before? 🙄

Quote:

[Originally Posted by blabberer;96269]IopCreatefile is a processor it can process only one request at any given time

yes...I am aware of the time-slice processing of computers. I am aware that IopCreateFile has other things to do than wait for me. However, in SI, I could figure other ways around that whereas in this new learning curve I am somewhat handcuffed, especially when the mouse freezes.

Quote:

[Originally Posted by blabberer;96269]you need to be in the context / context / context / context of what you doing for you to transact.

I have an intimate relationship with contexts from having plied my trade with SI. SI dealt with it differently. If something else had priority, SI would just break, allowing you access to the command line. Sometimes you could just hit go till it broke in the code you wanted. Other times there were too many intervening breaks.

The windbg/vm arrangement causes a conflict with the mouse and no one on the Net seems to have an answer for that.

Quote:

[Originally Posted by blabberer;96269]if you leave the break on IopCreatfile as it is and merrily step away you will break thousands of time when you press p (single step)

familiar with that from SI. I was religious about disabling the breakpoint that got me there, especially if it was a general breakpoint that could break on system activity. Many a time I have sworn at myself after hitting go, realizing I had forgotten to disable a break, getting stuck in the middle of foreign code. Sometimes i could step out of it but just as often that triggered the app.

You are preaching to the converted. I am from the Kayaker school of contexts.

WaxfordSqueers

March 24th, 2014, 15:32

Quote:

[Originally Posted by blabberer;96269]trash what you are doing sit back relaxed with a coffee unlearn all the shit you have learnt till now and start from scratch.

OK...that's done. I reset the entire system from scratch and followed your earlier instructions.

I get to my VM desktop and windbg tells me the debuggee is running. I go into the debugge, hit run/notepad willy15.text I hit enter and get a notepad window with the message box claiming Cannot find the willy15.txt file.

I get the mouse out of the vm window, go to windbg/debug/break and windebug allows me to enter bp nt!IopCreateFile.

I hit F5 and go back to the vm window so I can press OK in the message box window but i can't because the mouse cursor is an hour glass.

There is no rhyme, reason, or logic for that. There is no explanation anywhere on the Net for that because it is not supposed to do that. And no one has any frigging idea why it is doing that, only Gates and whoever wrote vmware.

I played around with it using ctrl-Alt and got rid of the hourglass. Now I have a cursor but when I press it nothing happens on the OK button in the message box.

It's obvious that the cursor I am seeing is the host cursor. When I press ctrl-Alt, it now disappears.

I am going back to read what Kayaker said about contexts.

WaxfordSqueers

March 24th, 2014, 16:10

Quote:

```
[Originally Posted by Kayaker;96268]kd> !process -1 0  
  
Instead set a context specific breakpoint with the /p Eprocess switch  
  
!process 0 0  
  
PROCESS 82306b48 Image: notepad.exe
```

Hokay...sometimes I'm a bit thick, but no thicker than two short planks.

Followed your recipe and had success. The hourglass appeared but I managed to force it off using ctrl-Alt and a couple of mouse presses. Notepad took it and I was able to get back to windbg and follow the rest.

Right now, I am a bit bagged and I am going to sign off and do this later. Thanks to you and blabs for bearing with me.

Although this method works for creating a file, I'd like to find a way to attack it from explorer by double-clicking notepad with a bp set on IopCreateFile. I notice it is just a sub-function of ZwCreateFile and I fear it will lead to the same problem, where notepad is already open.

Let's not be negative though, eh?

```
kd> .process 84ce0c18  
Implicit process is now 84ce0c18  
WARNING: .cache forcedecodeuser is not enabled  
kd> bp /p @$proc nt!IopCreateFile  
kd> bl  
0 e 8056ca21 0001 (0001) nt!IopCreateFile  
Match process data 84ce0c18  
  
kd> g  
watchdog!WdUpdateRecoveryState: Recovery enabled.  
Breakpoint 0 hit  
nt!IopCreateFile:  
8056ca21 6a3c push 3Ch  
kd> !process -1 0  
PROCESS 84ce0c18 SessionId: 0 Cid: 0788 Peb: 7ffde000 ParentCid: 0398  
DirBase: 0e1a3000 ObjectTable: e188d818 HandleCount: 27.  
Image: notepad.exe  
  
kd> kb  
ChildEBP RetAddr Args to Child  
f5489c94 8056ccba 0007fe0c c0100080 0007fdac nt!IopCreateFile  
f5489cf0 8056cdf0 0007fe0c c0100080 0007fdac nt!IoCreateFile+0x8e  
f5489d30 804de7ec 0007fe0c c0100080 0007fdac nt!NtCreateFile+0x30  
f5489d30 7c90e4f4 0007fe0c c0100080 0007fdac nt!KiFastCallEntry+0xf8  
0007fd68 7c90d09c 7c8109a6 0007fe0c c0100080 ntdll!KiFastSystemCallRet  
0007fd6c 7c8109a6 0007fe0c c0100080 0007fdac ntdll!ZwCreateFile+0xc  
WARNING: Frame IP not in any known module. Following frames may be wrong.  
0007fe04 01004a61 0100a900 c0000000 00000003 0x7c8109a6  
0007ffd0 8054b6b8 0007ffc8 84db1020 ffffffff 0x1004a61  
00080008 00000000 0000000c4 00000000 00000020 nt!ExFreePoolWithTag+0x676
```

WaxfordSqueers

March 24th, 2014, 19:03

Quote:

```
[Originally Posted by Kayaker;96268]kd> .process 82306b48
```

Could you have used kd> .context 82306b48 here?

Also, with !process -1 0 or !process 0 0, how do you know which numbers to use in there?

According to msoft the parameters are /s = session and /m = module but in one example it looks like this:

```
!process [/s Session] [/m Module] 0 Flags ImageName
```

There's a zero sitting right in the middle of it with no explanation.

msoft sure are fun when they explain things. As I told you before, they use \$mft with reference to the entire mft when they have defined the \$ value as applying to the first 16 system metafiles, which means \$mft is the very first file in the mft. Also, they use pidl as reference to a structure after claiming the p is a pointer to an idl structure.

Must be a hoot working at Redmond when Gates comes back to work and can't load the win 8.1 upgrade. I would love to have eaves dropped on that conversation when Gates called in the CEO to explain it.

WaxfordSqueers

March 24th, 2014, 19:09

Quote:

```
[Originally Posted by Kayaker;96268]Context, context, context.
```

Heck, Kayaker, you're closing in on 4000 posts and blabs, a self-proclaimed blabberer only has 1388. I just cleared 700 and I have no idea how I got that many. Wherever did you find the time and energy, especially for all those detailed explanations to the likes of me?

Congrats.

Where has the time gone since fravia and greythorne, not to mention Ork?

blabberer

March 25th, 2014, 01:40

Quote:

There's a zero sitting right in the middle of it with no explanation.

doc

Code:

If Process is 0 and ImageName is omitted, the debugger displays information about all active processes.

flags 0

doc saya

Code:

If Flags is 0, only a minimal amount of information is displayed

a comparison between default output that is plain **!process** without any arguments and an explicit command that reveals the same information viz **!process 0 3 windbg.exe** in windows xp

Code:

```
!kd> !process exe 0 3 windbg
```

GetPointerFromAddress: unable to read from 90558a54

!processs takes an _EPROCESS or Cid as input whereeas .context takes the PageDirectoryBase As input not an _EPROCESS or Cid

Code:

```
!kd> dt nt!_EPROCESS -ya Pc->Dir @$proc
```

```
+0x000_Pcb .
```

WaxfordSqueers

March 25th, 2014, 19:07

Quote:

[Originally Posted by blabberer;96277]doc

Thanks for explanation on process. Just want to verify something.

In SI, an expression with square brackets as in [exp] is interpreted as the value in eax.

Is poi the equivalent of that?. ie. does poi (eax) mean the value at eax?

I saw this expression for evaluating a mouse press:

```
bp [address] "j (poi(ebp+c))==0x202 "';gc"
```

Which breaks on windows message WM_LBUTTONDOWN. It seems to be saying that the bp should break if ebp+c = 0x202.

I seem to remember you giving an example where something similar was used in GetMessage, or some function in the winmain message loop. It's too bad windbg doesn't have the equivalent of the SI bmsg breakpoint where I could have used the expression above as:

```
bp hwnd "j (poi(ebp+c))==0x202 "';gc"
```

The problem I see with breaking in the message loop is having to trace through scads of code to ID the button pressed. I'd have to do it anyway but it's easier if it breaks in explorer as opposed to winmain.

Also, all that's required in SI is providing the hwnd obtained from SPYXX and the wmsg #.

Kayaker

March 25th, 2014, 20:04

<http://reverseengineering.stackexchange.com/questions/3288/how-can-i-set-a-breakpoint-for-a-button-click/3295#3295>

WaxfordSqueers

March 25th, 2014, 21:26

Quote:

[Originally Posted by Kayaker;96282]<http://reverseengineering.stackexchange.com/questions/3288/how-can-i-set-a-breakpoint-for-a-button-click/3295#3295>

Thanks for that Kayaker.

I was just investigating another possible method that I got from this suggestion:

What you need is to get the window procedure address....

A simple solution is to retrieve it manually with Spy++. Once you have it, go to that address and put a conditional breakpoint....

What I did was load explorer and open it to a directory in which I could install notepad. I then opened SPYXX and found the entry for explorer under "directory" ExploreWClass. Under that heading is another called SHELLDLL_DefView with the hwnd in front of it.

Under SHELLDLL_DefView is another entry called "FolderView" SysListView32. FolderView is the window in Explorer where the folders are found.

If you highlight that line, "FolderView" SysListView32, right-click and select Properties, you will find the Window Proc address to be 77420C9D and it is found in comctl32.dll for version 6.0.2600.6028.

Note: the comctl32 version is found in Windows\winsxs, not in windows\system32.

That winproc in comctl32 is the beginning of a function in IDA that leads to CALL GetWindowLongW. The first push before that call is the hwnd and the previous push is the index.

Theoretically, in my particular version of Explorer, I should be able to BP on GetWindowLongW with the hwnd found in SPYXX as a condition.

I realize that the winproc shown in SPYXX is not the address of the call to GetWindowLongW, which comes shortly after at 77420CC5. However, the push for the hwnd is a push eax and that is loaded by a mov eax, [ebp+8] a few steps before.

It might be possible to use the SPYXX winproc address for the "FolderView" SysListView32 explorer window with the condition that [ebp+8] = hwnd of same. ebp does not seem to change from the winproc address until GetWindowLongW is called.

```
.text:77420C9D mov edi, edi
.text:77420C9F push ebp
.text:77420CA0 mov ebp, esp
.text:77420CA2 sub esp, 148h
.text:77420CA8 mov eax, dword_774623E0
.text:77420CAD push ebx
.text:77420CAE mov ebx, [ebp+10h]
.text:77420CB1 push esi
.text:77420CB2 push edi
.text:77420CB3 mov edi, [ebp+14h]
.text:77420CB6 mov [ebp-4], eax
.text:77420CB9 mov eax, [ebp+8]
.text:77420CBC push 0
.text:77420CBE push eax
.text:77420CBF mov [ebp-108h], eax
.text:77420CC5 call ds:GetWindowLongW
```

blabberer

March 26th, 2014, 02:39

a round of applause to kayaker for revealing his secret of success.

```
yes [esp] in sice == poi(@esp)
so [[[ebp+4]]] == poi(poi(poi(@ebp+4)))
```

use of @ denotes what follows is a register and not a symbol and cuts down unneeded symbol loading time poi(esp) should working equally well albeit slower than poi(@esp)

here is a refresher course on semantics take time to practice 😊

Code:

```
for %i in %* do (
    echo %i
)
```

lets play with expressions and get to know our poi on double square brackets

Code:

```
masm expression evaluation
0:000> 2 var1
```

WaxfordSqueers

March 26th, 2014, 13:51

Quote:

[Originally Posted by blabberer;96284]
yes [esp] in sice == poi(@esp)
so [[[ebp+4]]] == poi(poi(poi(@ebp+4)))

Thanks for tute...mighty handy. The thing that scares me is that I can follow you, even the C++ stuff. I'll have to dust off my C++ compiler.

I noticed the use of 'n' rather than 'X'. What's with that? eg. var3 = 0n4228356

I'm going to do what you suggest and practice some of this but it does help a lot when someone familiar with it explains it. I have done a considerable amount of reading on windbg and I recall the explanation of masn expressions as opposed to c++ expressions and @@ versus poi.

It still doesn't make complete sense to me. I guess I have been using the masn expressions in SI but I don't understand the need to bring c++ expressions into the equation.

In SI, I know that 0x7717DDEE is a value and that [0x7717DDE] is a value at the address 0x7717DDEE. I even understand nested values using pointers. I've had that clearly in my mind for a long time and I can read code in SI quickly. Now I have to contend with c++ expressions and I am not getting why. I'll go back and read more on the subject but thanks for taking the time to write down all this stuff. I'm sure anyone reading it will benefit from it as well, even if just for a review.

WaxfordSqueers

March 26th, 2014, 15:03

Quote:

[Originally Posted by WaxfordSqueers;96285]I'll go back and read more on the subject

Answered some of the questions for myself.

The 'n' as in 0n77171177 means decimal.

Whenever I have used SI it has been without source code. I've had no need for evaluating C++ expressions. I can see why the debugger needs that ability.

Powered by vBulletin® Version 4.2.2 Copyright © 2014 vBulletin Solutions, Inc. All rights reserved.