

Through the Looking Glass CIFS/SMB/SMB2 Telemetry

**Robert Randall
Micron Technology, Inc.**

**Christopher Hertel
ubiqx Consulting, Inc.**

- ❑ The in-box Windows drivers have a lot of telemetry to share.
 - ❑ History of telemetry for drivers.
 - ❑ How do they share it?
 - ❑ How do you listen to it?
 - ❑ What does it mean?
 - ❑ Applying it to things you care about...
 - ❑ The SMB & SMB2 client & server are drivers.
 - ❑ What do they share?

The evolution of driver telemetry

- ❑ The dark times
 - ❑ Ye old printf – this better not surprise you.
 - ❑ Any questions?
 - ❑ Getting the output requires the checked / debug version of a driver that uses DbgPrint
 - ❑ Checked builds cost money; MSDN only?
 - ❑ Getting the output requires the Windows kernel debugger.
 - ❑ Output to the debugger REALLY slows the system.

The Dark Times - DbgPrint

- ❑ The Windows kernel debugger is **the** tool to use when debugging device drivers.
- ❑ Windows kernel debugger is **FREE** and very good.
 - ❑ Comes with the WDK or separate [download](#) (both free)
 - ❑ You need the symbols for your version of Windows for debugging to be useful. Free [download](#)
 - ❑ You need a checked (debug) build of the drivers you are interested in; the ones that interact with your code.
 - ❑ Installing a checked driver onto a target is a manual process that takes some tinkering (easy to get it wrong).

The Dark Times - DbgPrint

- There are over 140 debug print masks available in the kernel. That is a lot...

```
kd> x /n nt!Kd_*
fffff800`01e23d5c nt!Kd_ACPI_Mask = <no type information>
fffff800`01e23ea0 nt!Kd_ALPC_Mask = <no type information>
fffff800`01e23d60 nt!Kd_AMLI_Mask = <no type information>
fffff800`01e23ee8 nt!Kd_APPCOMPAT_Mask = <no type information>
fffff800`01e23da8 nt!Kd_AUTOCHK_Mask = <no type information>
fffff800`01e23dc0 nt!Kd_BURNENG_Mask = <no type information>
fffff800`01e23ef8 nt!Kd_CACHEMGR_Mask = <no type information>
fffff800`01e23d14 nt!Kd_CDAUDIO_Mask = <no type information>
fffff800`01e23d18 nt!Kd_CDROM_Mask = <no type information>
fffff800`01e23f00 nt!Kd_CFR_Mask = <no type information>
fffff800`01e23d1c nt!Kd_CLASSPNP_Mask = <no type information>
...
fffff800`01e23ea4 nt!Kd_WDI_Mask = <no type information>
fffff800`01e23ed4 nt!Kd_WDT_Mask = <no type information>
fffff800`01e23eb8 nt!Kd_WHEA_Mask = <no type information>
fffff800`01e0bfb0 nt!Kd_WIN2000_Mask = <no type information>
fffff800`01e23dbc nt!Kd_WMICORE_Mask = <no type information>
fffff800`01e23d58 nt!Kd_WMILIB_Mask = <no type information>
fffff800`01e23e9c nt!Kd_WOW64_Mask = <no type information>
fffff800`01e23e14 nt!Kd_WSOCKETTRANSPORT_Mask = <no type information>
fffff800`01e23f38 nt!Kd_XSAVE_Mask = <no type information>
```

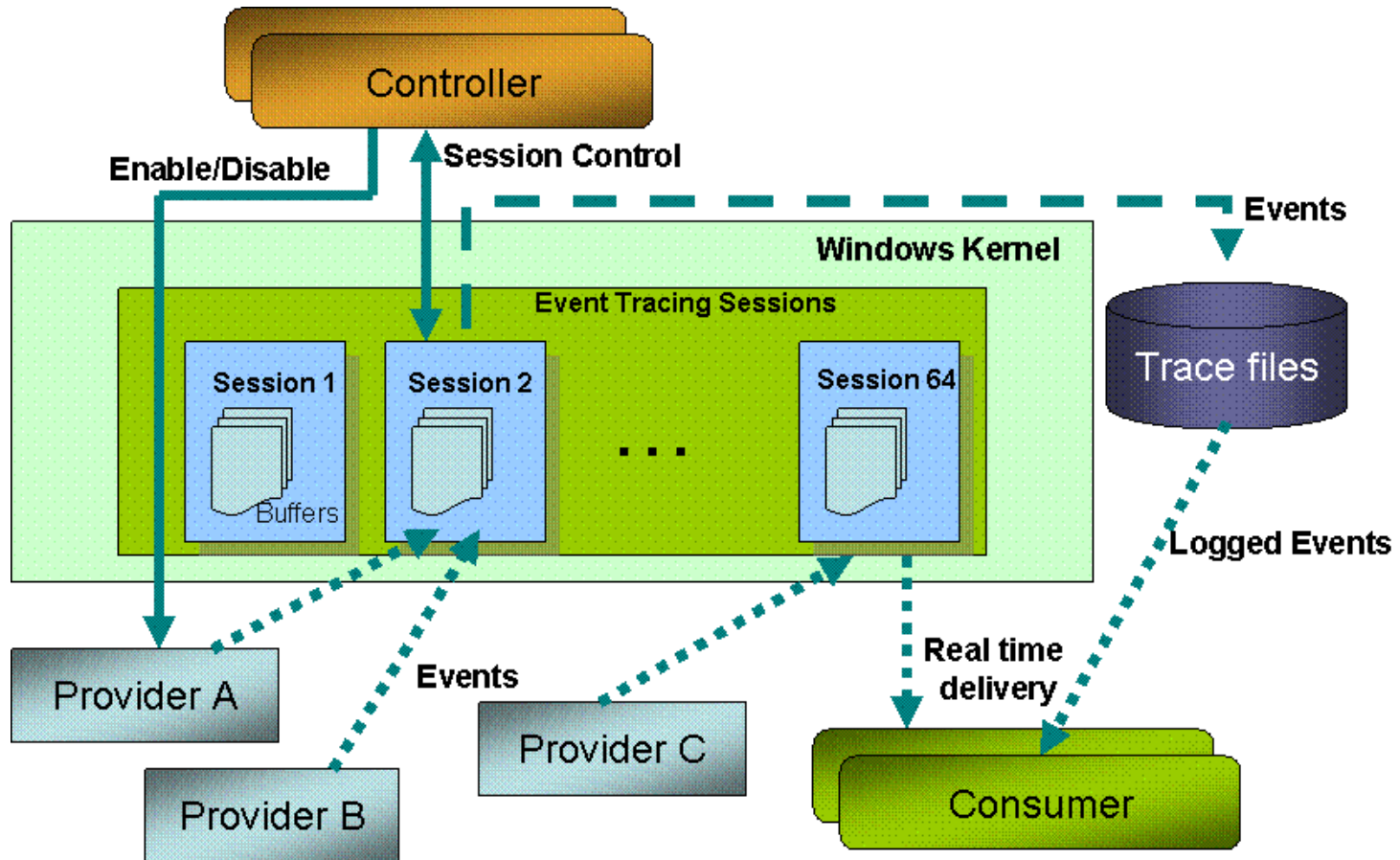
The Dark Times - DbgPrint

- ❑ Several steps to get this working properly
- ❑ Telemetry is very raw
 - ❑ Great learning tool if you already understand kernels and want to know more about what is going on inside of the Windows kernel.
 - ❑ Quite useful for understanding problems with device drivers that service hardware
 - ❑ Many device driver do not use hardware, they are services that run in kernel space like the SMB client and server.

The evolution of driver telemetry

- ❑ The Middle Ages: [WMI Event Tracing](#) (overview [doc](#))
 - ❑ Raw data is separate from printf style output strings.
 - ❑ Compiled into checked / non-debug / released drivers
 - ❑ Very nice data capture; pre-allocated in kernel buffers for capturing raw data, very little impact on speed of code execution.
 - ❑ Windows 2000 +

The Middle Ages – Tracing

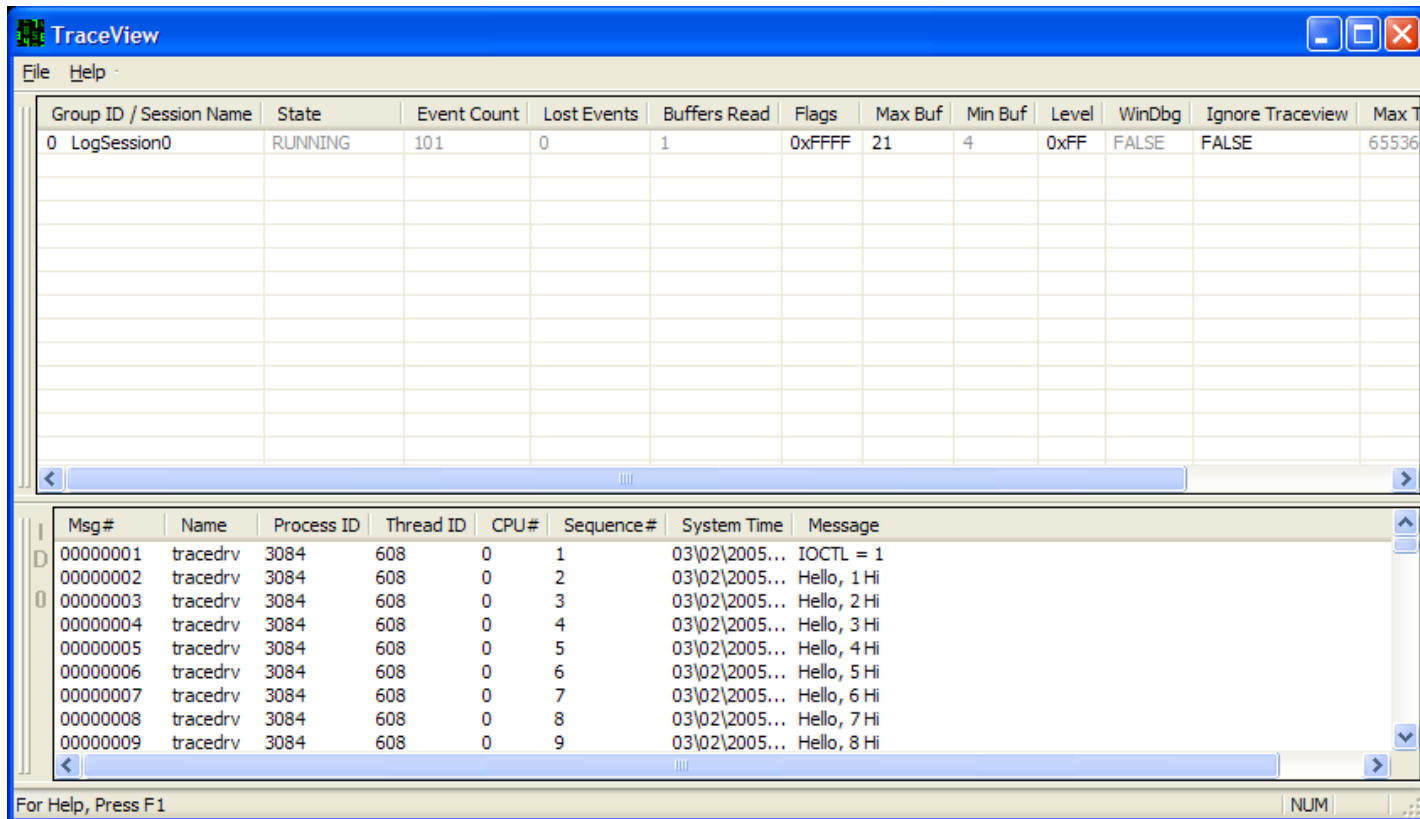


The Middle Ages – Tracing

- ❑ Turn it on/off, trace levels and components
 - ❑ Nearly ZERO COST when turned off.
- ❑ Reading a trace is a two (2) step process
 - ❑ Collect the raw data → Format it into human readable text **BUT** to do this you need the magic decoder rings
 - ❑ The TMF files or the PDB file.
 - ❑ You need **private symbols!**
 - ❑ Oh, and by the way, Microsoft **doesn't share** private symbols and you can't expect them to – very risky and you know why.

The Middle Ages - Tracing

- Use TraceView to start / stop, format & view trace sessions in near-real time.



The Middle Ages - Tracing

- ❑ In-box drivers begin to incorporate tracing.
 - ❑ Good news; more info w/o debugger slow down
 - ❑ Bad news; decoder rings are only available from Microsoft Professional Support Services.
 - ❑ You have to pay to get logs decoded.
 - ❑ Necessary if you're stuck.
 - ❑ But what if you can get un-stuck with just the decoder ring?
 - ❑ You don't need a PSS engineer to tell you how to fix it, you just need a better understanding of the failure condition.

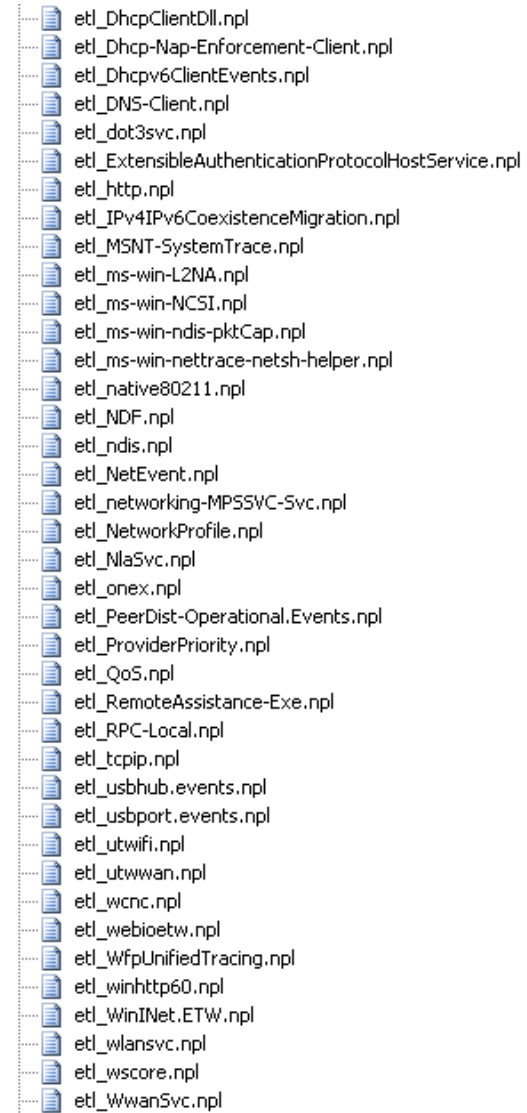
The evolution of driver telemetry

- ❑ The Renaissance: Event Tracing for Windows (ETW)
- ❑ Released with Vista
- ❑ Extends WMI Event Tracing
 - ❑ Decouples raw data → human readable conversion from the symbols.
 - ❑ A raw data → human readable conversion schema may be created and PUBLISHED.

- ❑ Magic Decoder rings go public!
 - ❑ Schema is XML and may contain a lot of event details and context.
 - ❑ Many decoders are built-in and are supported from Performance Monitor and Event Viewer
 - ❑ Microsoft publishes decoder rings.

The Renaissance - ETW

- ❑ NetMon has parsers created from schemata
 - ❑ Parsers are in the public domain and on [CodePlex](#)
 - ❑ Oh this is lovely...



A screenshot of a file explorer window displaying a list of ETW parser files. The files are listed in a vertical column, each with a small icon to its left. The list includes:

- etl_DhcpClientDll.npl
- etl_Dhcp-Nap-Enforcement-Client.npl
- etl_Dhcpv6ClientEvents.npl
- etl_DNS-Client.npl
- etl_dot3svc.npl
- etl_ExtensibleAuthenticationProtocolHostService.npl
- etl_http.npl
- etl_IPv4IPv6CoexistenceMigration.npl
- etl_MSNT-SystemTrace.npl
- etl_ms-win-L2NA.npl
- etl_ms-win-NCSI.npl
- etl_ms-win-ndis-pktCap.npl
- etl_ms-win-nettrace-netsh-helper.npl
- etl_native80211.npl
- etl_NDF.npl
- etl_ndis.npl
- etl_NetEvent.npl
- etl_networking-MPSSVC-Svc.npl
- etl_NetworkProfile.npl
- etl_NlaSvc.npl
- etl_onex.npl
- etl_PeerDist-Operational.Events.npl
- etl_ProviderPriority.npl
- etl_QoS.npl
- etl_RemoteAssistance-Exe.npl
- etl_RPC-Local.npl
- etl_tcpip.npl
- etl_usbhub.events.npl
- etl_usbport.events.npl
- etl_utwifi.npl
- etl_utwwan.npl
- etl_wcnc.npl
- etl_webioetw.npl
- etl_WfpUnifiedTracing.npl
- etl_winhttp60.npl
- etl_WinINet.ETW.npl
- etl_wlansvc.npl
- etl_wscore.npl
- etl_WwanSvc.npl

The Renaissance - ETW

□ The renaissance Event Tracing for Windows (ETW)

The screenshot displays the Microsoft Network Monitor 3.4 interface. The main window shows a list of network conversations, with a selected conversation (Frame 4) expanded to show details. The details pane is split into two sections: Frame Summary and Frame Details.

Frame Summary - [Conversation Filter]

Frame Number	Time Date Local Adjusted	Time Offset	UT Process Name	Source	Destination	Protocol Name	Description
2	7:47:43 PM 9/18/2011	66.5462437 (4424)				BranchCach...	BranchCacheEvent_MicrosoftWindowsBranchCacheClientEventProvider:ClientOpenConten
3	7:47:43 PM 9/18/2011	66.5498383 (4424)				BranchCach...	BranchCacheEvent_MicrosoftWindowsBranchCacheClientEventProvider:ClientStreamRead
4	7:47:43 PM 9/18/2011	66.8561851 (4424)				BranchCach...	BranchCacheEvent_MicrosoftWindowsBranchCacheClientEventProvider:ClientStreamRead
5	7:47:48 PM 9/18/2011	71.9405439 (4424)				BranchCach...	BranchCacheEvent_MicrosoftWindowsBranchCacheClientEventProvider:ClientCloseConten

Frame Details

```
ProcessId: 4424, ProcessName:  
TimeStamp: 09/19/2011, 00:47:43.608715 UTC  
ProviderId: {E837619C-A2A8-4689-833F-47B48EBD2442}  
Descriptor:  
ProcessorTime: 0 (0x0)  
ActivityId: {014A6A3D-1178-1148-005C-9A0800CC000C}  
ETLProvider:  
BufferContext:  
ProcessorNumber: 0 (0x0)  
Alignment: 8 (0x8)  
LoggerId: 21 (0x15)  
ExtendedDataCount: 0 (0x0)  
UserDataLength: 4 (0x4)  
Reassembled: 0 (0x0)  
MicrosoftWindowsBranchCacheClientEventProvider: ClientStreamReadComplete  
ClientStreamReadCompleteEvent: ClientStreamReadComplete  
Error Code: PEERDIST_ERROR_MISSING_DATA  
Code: {00000000000000000000000000000000} 0x00000FD4 - PEERD  
Facility: {00000000000000000000000000000000} WIN32  
X: {00000000000000000000000000000000} Reserved  
N: {00000000000000000000000000000000} Not NTSTATUS  
C: {00000000000000000000000000000000} Microsoft-defined  
R: {00000000000000000000000000000000} Reserved  
S: {00000000000000000000000000000000} Failure
```

Hex Details

Decode As	Width	Prot Off: 0 (0x00)	Frame Off: 89 (0x59)
0000	54	00 00 00 20 00 00 00 00 DC 11 T...	...
000A	00	00 48 11 00 00 71 83 DD BD ..H...q Y%	...
0014	65	76 CC 01 9C 61 37 E8 A8 A2 evl. a7è'c	...
001E	89	46 83 3F 47 B4 8E BD 24 42 F ?G' %\$B	...
0028	7B	27 00 10 04 00 00 00 00 00 00 00 00	{'.....
0032	00	00 00 00 00 80 00 00 00 00 00 00 00
003C	00	00 00 0C 00 CC 00 08 9Ai..
0046	5C	00 48 11 78 11 3D 6A 4A 01 \.H.x.=jU.
0050	00	08 15 00 00 00 04 00 00 D4D.
005A	0F	07 80

- ❑ The bad news...
 - ❑ The decoder rings for SMB, SMB2, & SMB 2.2 are not part of the NetMon parsers (yet).
 - ❑ **CALL TO ACTION.**
 - ❑ Talk to your favorite Microsoft person at the PlugFest and encourage them to add these protocols to the family of available parsers.
 - ❑ **WHY?**
 - ❑ The more you know the less likely your are to create an incompatible implementation.
 - ❑ Better 3rd party implementations → lower support costs for Microsoft.

Wrap up

- Q&A
- A quick show and tell time permitting...