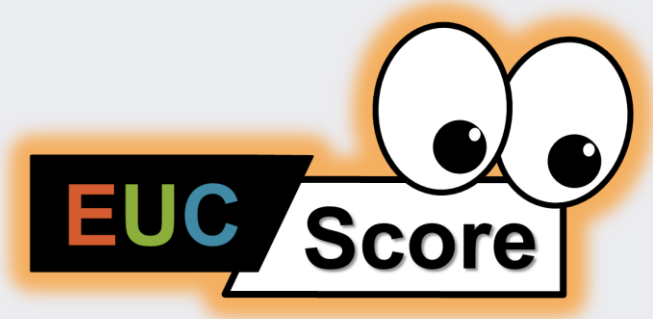


>< AVD TECH FEST

hosted by control UP

Microsoft nerdio citrix IGEL LOGINVSI 10ZiG ezeep by ThinPrint numecent UltraArmor

EUC Score

2025 AVD TECH FEST

hosted by control UP

Mastering Windows Performance Counters for Azure Virtual Desktops and Cloud PCs

Dr. Benny Tritsch | Independent Performance Data Scientist

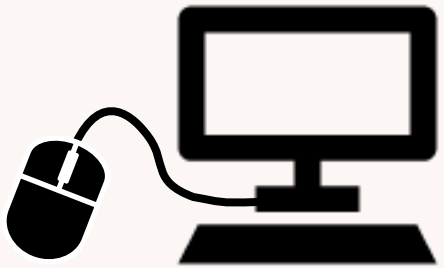
info@eucscore.com | https://drtritsch.com | linkedin.com/in/drtritsch

NVIDIA Advisor NVIDIA Enterprise Platform | omnissa TECH INSIDER | NERDIO VALUED PROFESSIONAL | Microsoft MVP Most Valuable Professional | VIPP Very Important Parallels Professional Parallels

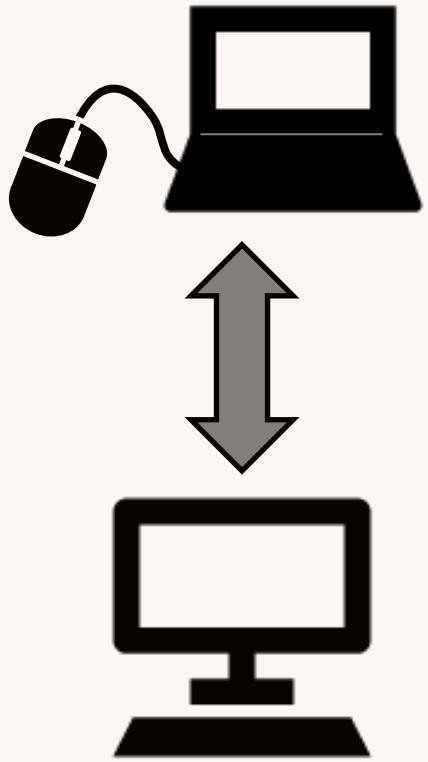
Microsoft | nerdio | citrix | IGEL | LOGINVSI | 10ZiG | ezeep by ThinPrint | numecent | UltrArmor

Why Performance Counters Matter

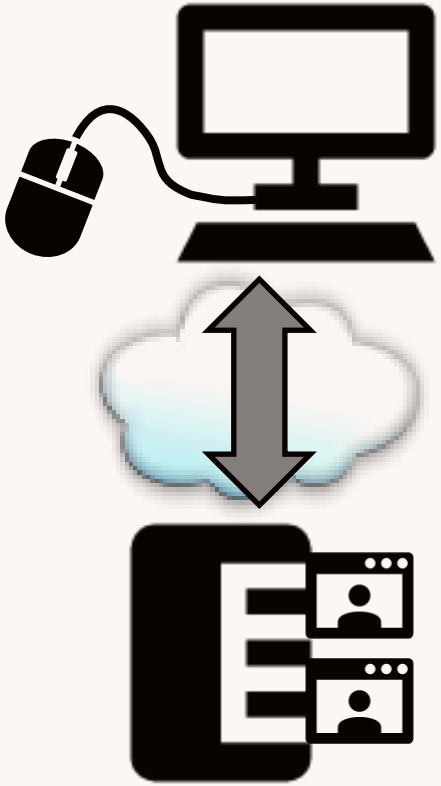
On-Prem Windows



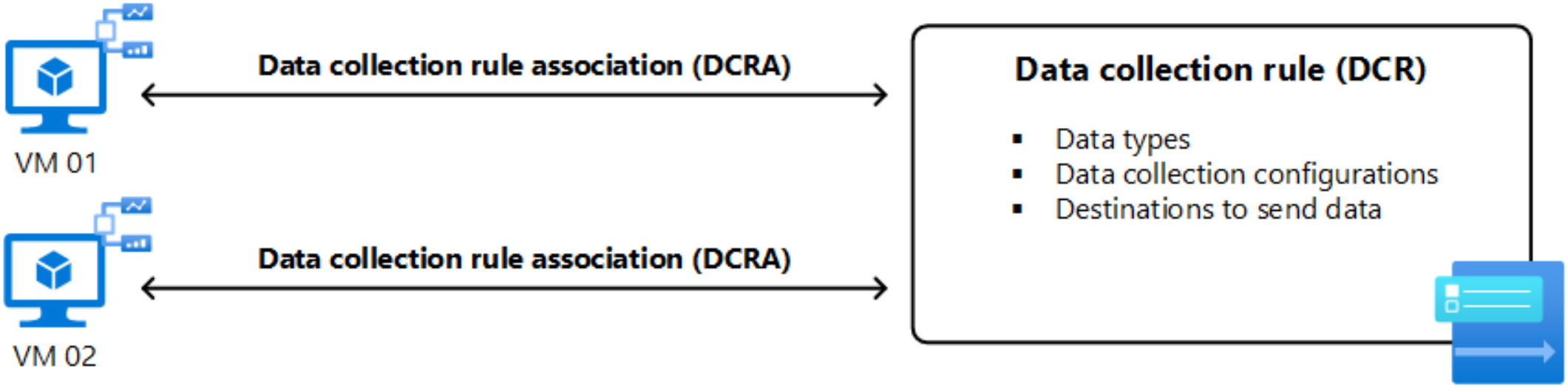
VDI / RDSH on LAN



AVD / Win365



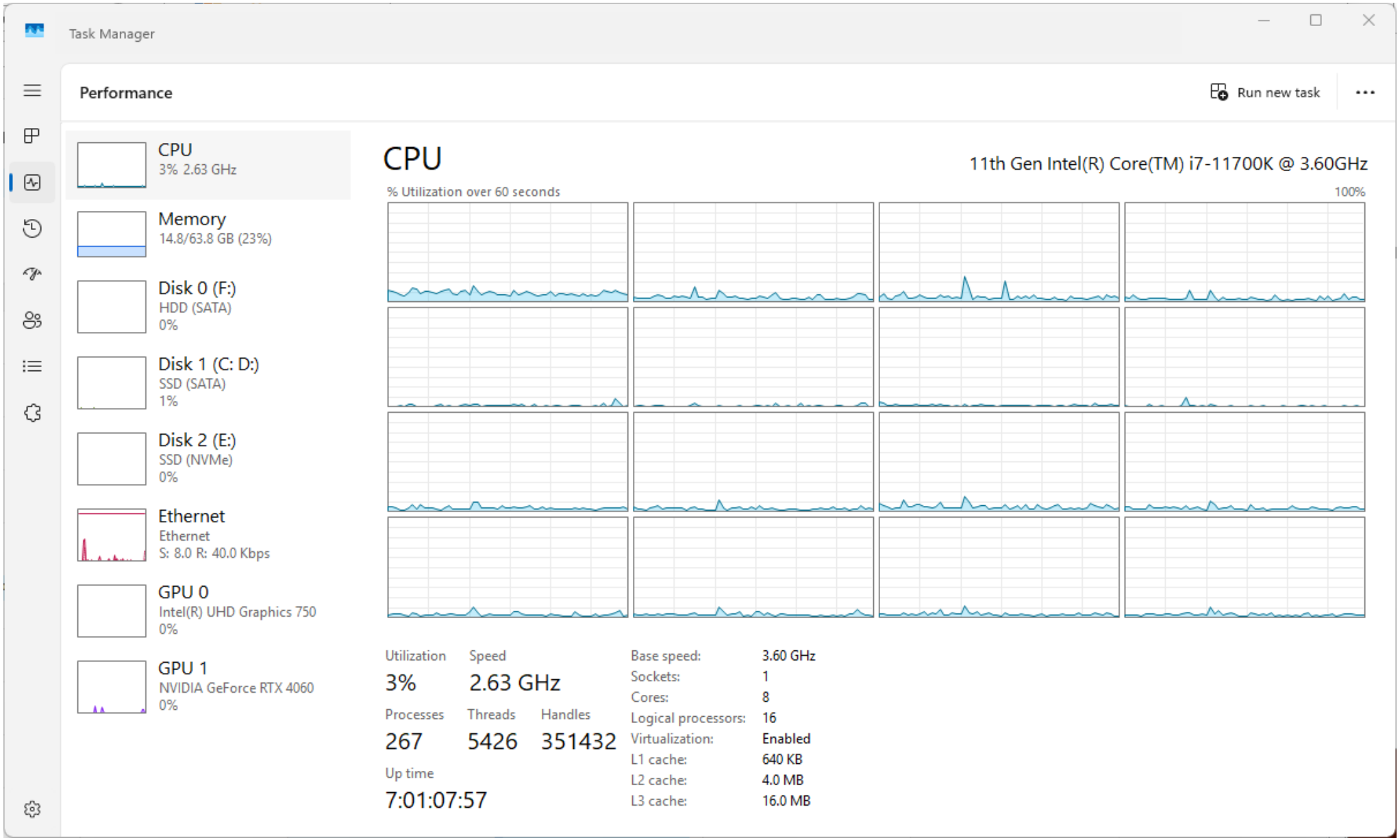
Azure Monitor Agent



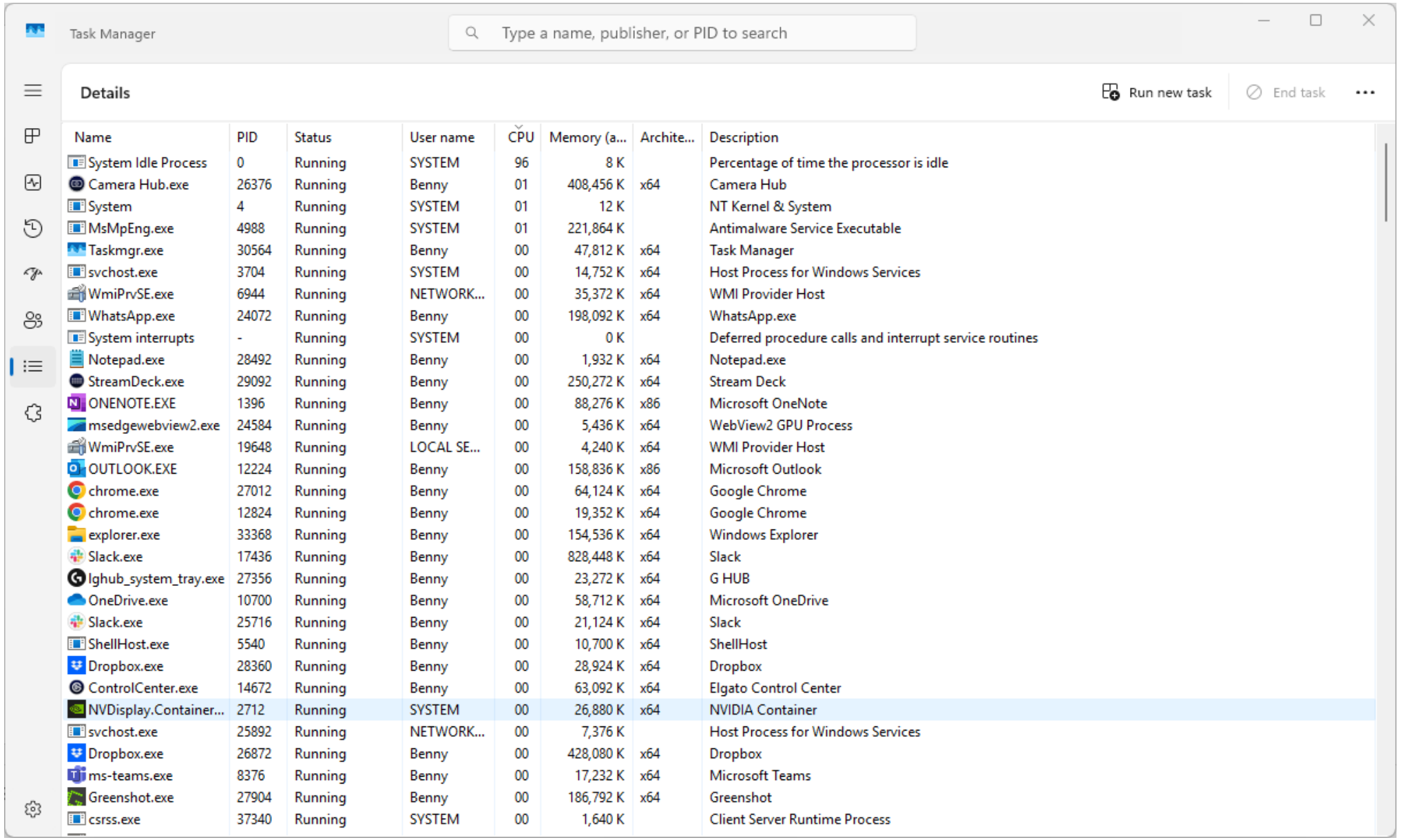
https://learn.microsoft.com/en-us/azure/azure-monitor/agents/azure-monitor-agent-overview

Azure monitoring was not designed for Windows telemetry data

Task Manager – Performance Tab

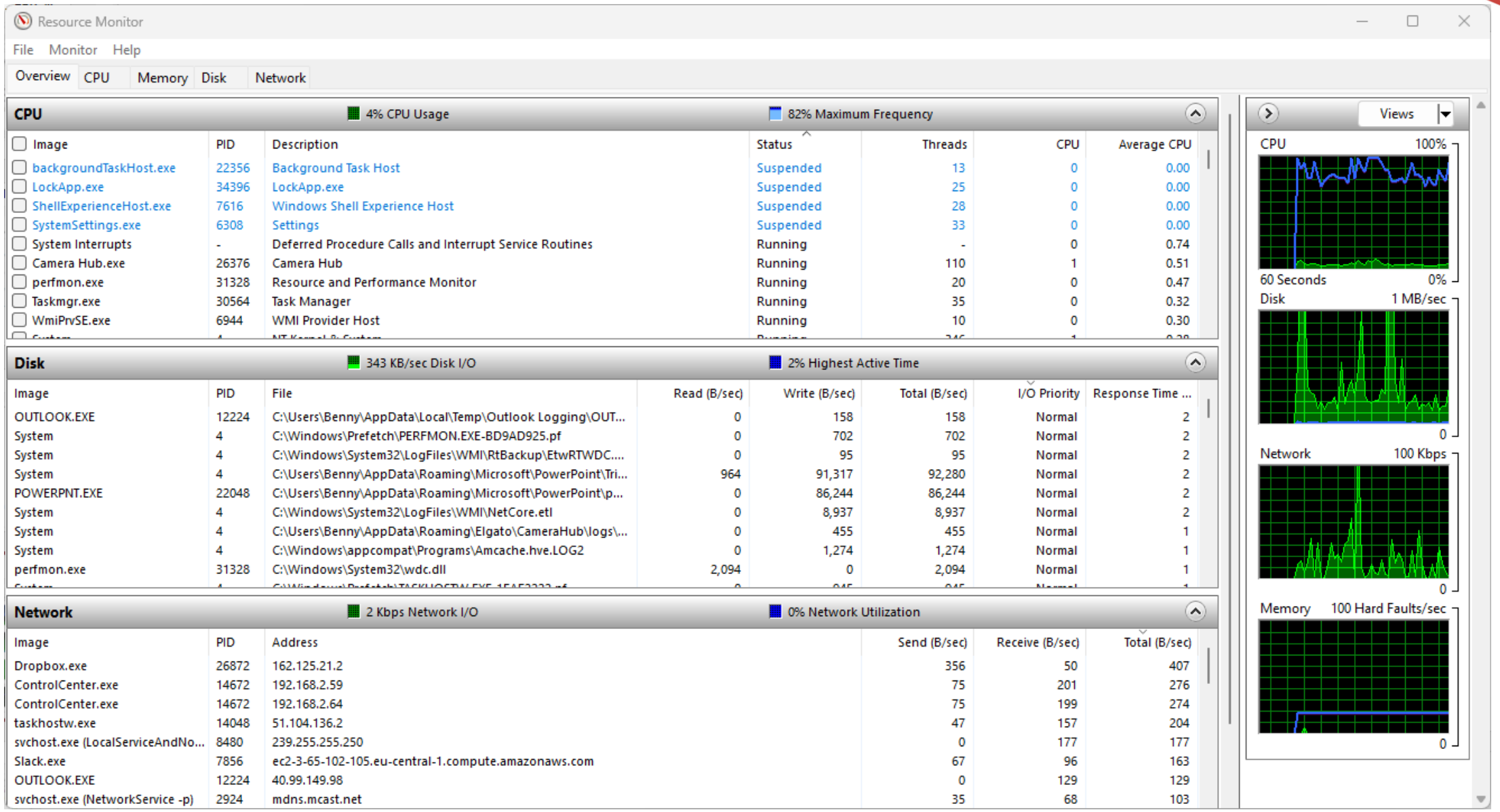


Task Manager – Detail Tab

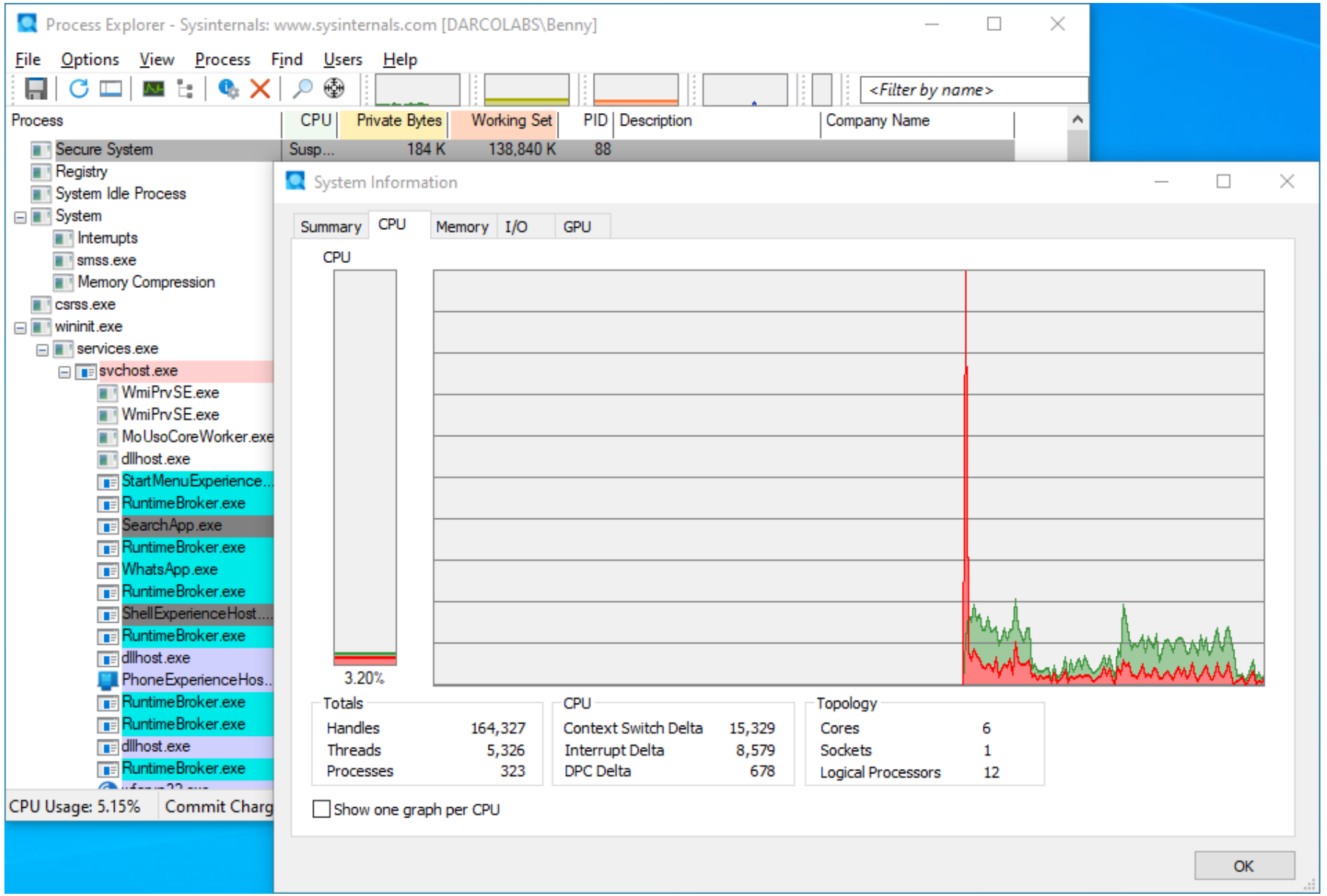


Name	PID	Status	User name	CPU	Memory (a...	Archite...	Description
System Idle Process	0	Running	SYSTEM	96	8 K		Percentage of time the processor is idle
Camera Hub.exe	26376	Running	Benny	01	408,456 K	x64	Camera Hub
System	4	Running	SYSTEM	01	12 K		NT Kernel & System
MsMpEng.exe	4988	Running	SYSTEM	01	221,864 K		Antimalware Service Executable
Taskmgr.exe	30564	Running	Benny	00	47,812 K	x64	Task Manager
svchost.exe	3704	Running	SYSTEM	00	14,752 K	x64	Host Process for Windows Services
WmiPrvSE.exe	6944	Running	NETWORK...	00	35,372 K	x64	WMI Provider Host
WhatsApp.exe	24072	Running	Benny	00	198,092 K	x64	WhatsApp.exe
System interrupts	-	Running	SYSTEM	00	0 K		Deferred procedure calls and interrupt service routines
Notepad.exe	28492	Running	Benny	00	1,932 K	x64	Notepad.exe
StreamDeck.exe	29092	Running	Benny	00	250,272 K	x64	Stream Deck
ONENOTE.EXE	1396	Running	Benny	00	88,276 K	x86	Microsoft OneNote
msedgewebview2.exe	24584	Running	Benny	00	5,436 K	x64	WebView2 GPU Process
WmiPrvSE.exe	19648	Running	LOCAL SE...	00	4,240 K	x64	WMI Provider Host
OUTLOOK.EXE	12224	Running	Benny	00	158,836 K	x86	Microsoft Outlook
chrome.exe	27012	Running	Benny	00	64,124 K	x64	Google Chrome
chrome.exe	12824	Running	Benny	00	19,352 K	x64	Google Chrome
explorer.exe	33368	Running	Benny	00	154,536 K	x64	Windows Explorer
Slack.exe	17436	Running	Benny	00	828,448 K	x64	Slack
lghub_system_tray.exe	27356	Running	Benny	00	23,272 K	x64	G HUB
OneDrive.exe	10700	Running	Benny	00	58,712 K	x64	Microsoft OneDrive
Slack.exe	25716	Running	Benny	00	21,124 K	x64	Slack
ShellHost.exe	5540	Running	Benny	00	10,700 K	x64	ShellHost
Dropbox.exe	28360	Running	Benny	00	28,924 K	x64	Dropbox
ControlCenter.exe	14672	Running	Benny	00	63,092 K	x64	Elgato Control Center
NVDisplay.Container...	2712	Running	SYSTEM	00	26,880 K	x64	NVIDIA Container
svchost.exe	25892	Running	NETWORK...	00	7,376 K		Host Process for Windows Services
Dropbox.exe	26872	Running	Benny	00	428,080 K	x64	Dropbox
ms-teams.exe	8376	Running	Benny	00	17,232 K	x64	Microsoft Teams
Greenshot.exe	27904	Running	Benny	00	186,792 K	x64	Greenshot
csrss.exe	37340	Running	SYSTEM	00	1,640 K		Client Server Runtime Process

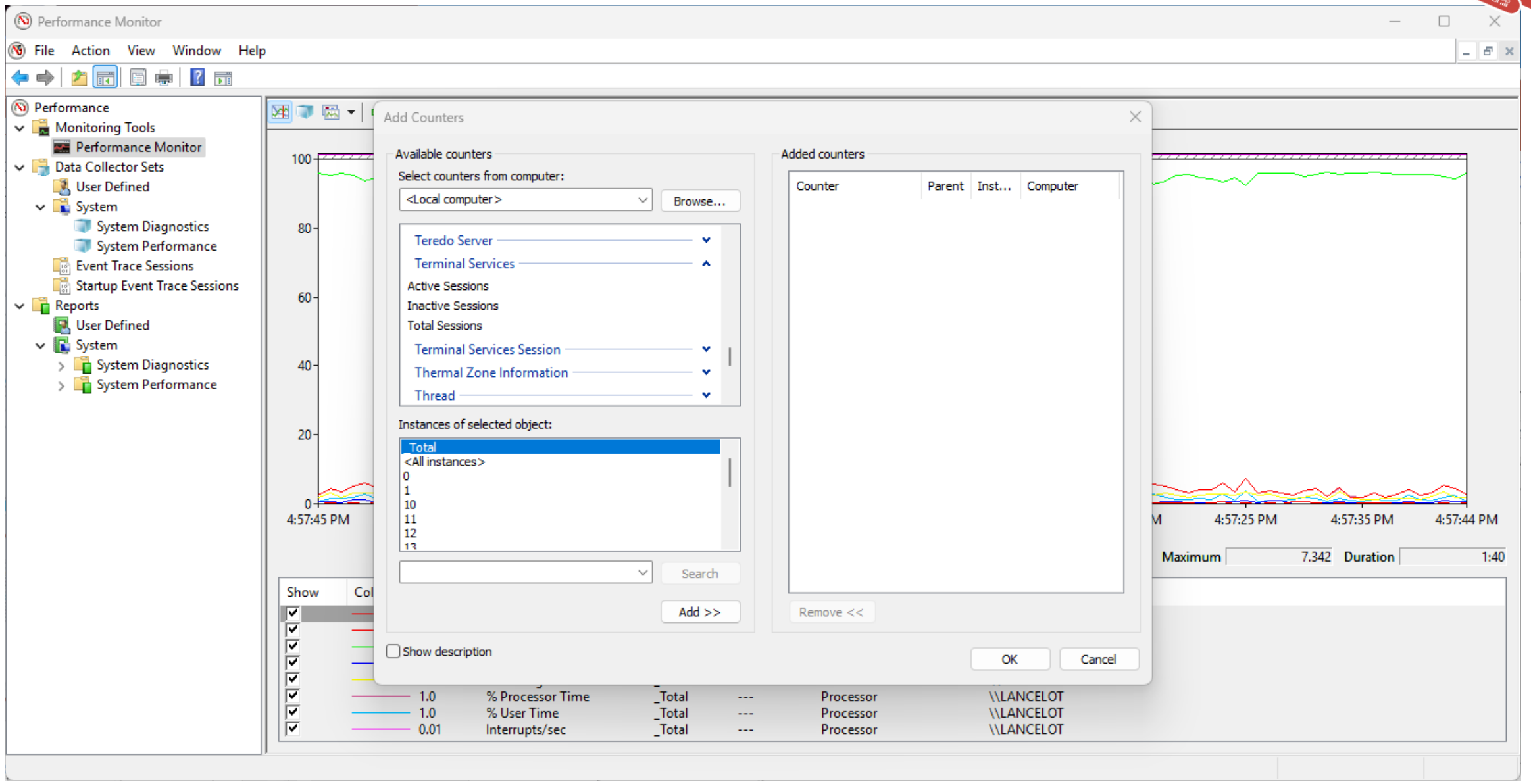
Resource Monitor



Sysinternals Process Explorer



Performance Monitor



Project Hydra | OneDrive Dev Cent... | Screenshot provided by Marcel Meurer | All Bookmarks

Microsoft Azure | Search resources, services, and docs (G+/) | Copilot | mm-admin@ITProClou... ITPROCLOUD.DE

Home > Data collection rules > microsoft-avdi-westeurope

Data collection rules
ITProCloud.de

+ Create | Manage view | ...

Filter for any field...

We are previewing a new Browse experience. Click to switch.

Name

- hydra-itpc-dev
- microsoft-avdi-westeurope

Azure Monitor

Page 1 of 1

microsoft-avdi-westeurope | Data sources
Data collection rule

Search

- Overview
- Activity log
- Access control (IAM)
- Tags
- Settings
  - Locks
- Configuration
  - Data sources
  - Resources
- Automation
  - Policies (Preview)
  - CLI / PS
  - Tasks (preview)
  - Export template
- Security
  - Identity
- Monitoring
  - Alerts
  - Metrics
  - Diagnostic settings
  - Logs
- Help
  - Support +

+ Add | Delete

Filter by name...

Data source
Performance Counters
Windows Event Logs

Add data source

Azure Monitor Agent
Azure Log Analytics
AVD Insights

*Data source | Destination

Select which data source type and the data to collect for your resource(s).

Data source type *: Performance Counters

Choose Basic to enable the collection of performance counters. Choose Custom if you want more control over which performance counters are collected.

None | Basic | Custom

Configure the performance counters to collect, and how often they should be sampled:

Add

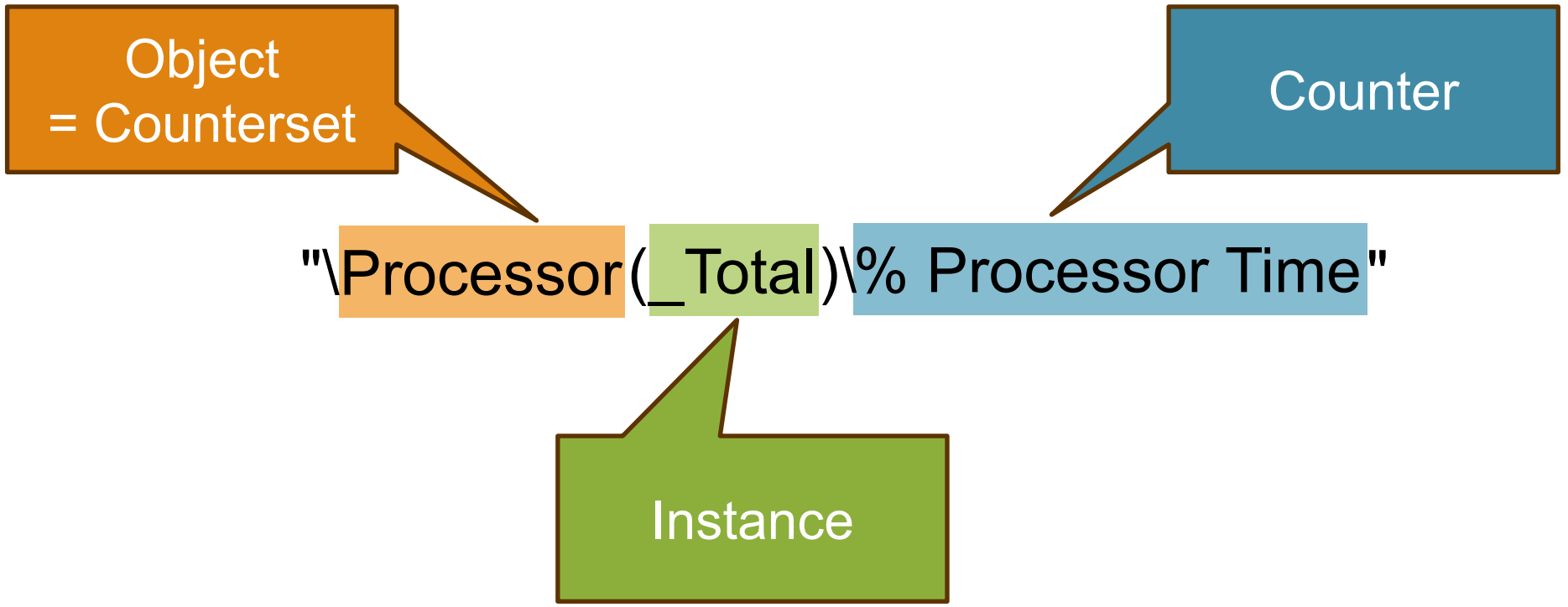
Performance counter	Sample rate (seconds)
\Memory\% Committed Bytes In Use	30
\Memory\Available Bytes	60
\Memory\Committed Bytes	60
\Memory\Cache Bytes	60
\Memory\Pool Paged Bytes	60
\Memory\Pool Nonpaged Bytes	60
\Memory\Pages/sec	30
\Memory\Page Faults/sec	30
\Process(_Total)\Working Set	60
\Process(_Total)\Working Set - Private	60

Showing 11 - 20 of 59 results. < 1 2 3 4 5 >

Save | Next : Destination > | Cancel

Performance Counter Path Syntax

\\ComputerName\ObjectName(ObjectInstance)\ObjectCounter



A *counterset* is a grouping of performance data within a provider
A *counter* is the definition of single piece of performance data
An *instance* is an entity about which performance data is reported

Windows Performance Counters

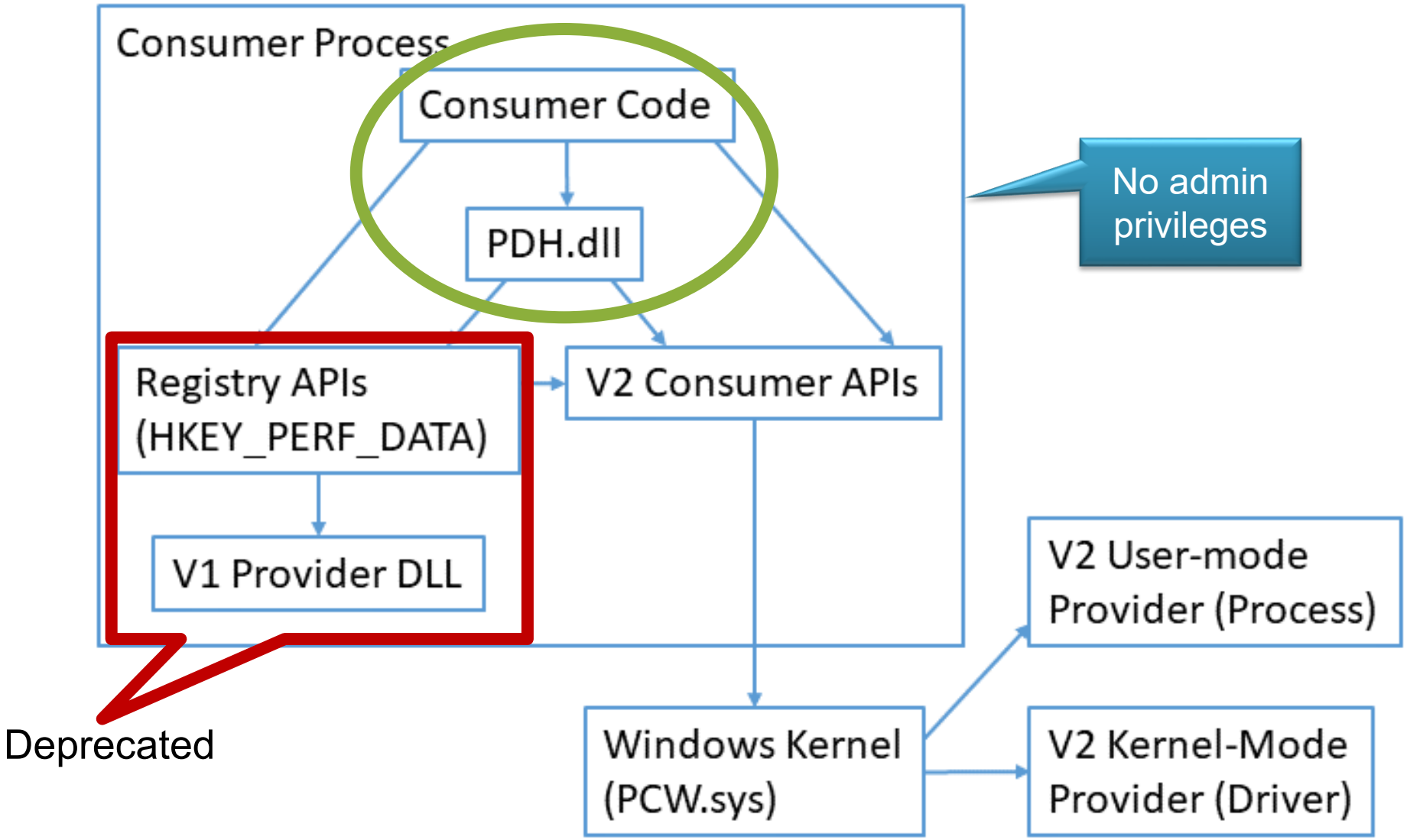
Consistent interface for collecting various kinds of system data

- A **provider** is a software component that generates and publishes performance data
- A **counterset** (or **object**) is a grouping of performance data within a provider
- A **counter** is the definition of single piece of performance data
- An **instance** is an entity about which performance data is reported
- A **counter value** is the value of a single piece of performance counter data
- The **counter type** indicates the type of the counter's raw value and indicates what the counter's raw value represents

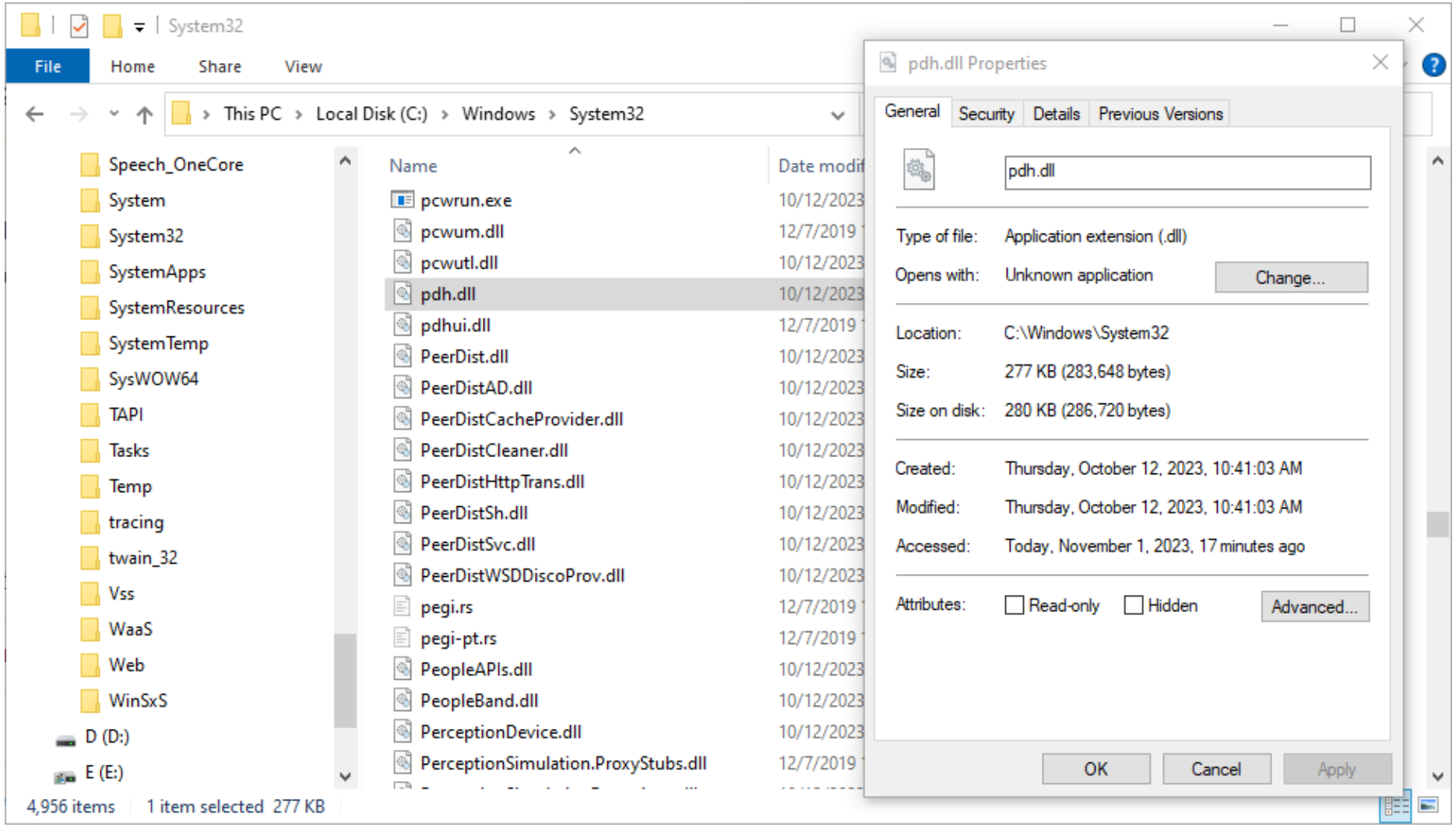
Windows Performance Counters

- **Single-instance countersets** always contain data for exactly one instance
- **Multi-instance countersets** contain data for a variable number of instances
- A **consumer** is a software component that makes use of performance data. It periodically collects and records the data from a provider's counterset:
  - GUI: Task Manager, Resource Monitor, Performance Monitor, and Sysinternals Process Explorer
  - CMD: Typeperf.exe, Logman.exe, and Relog.exe
  - EUC Score: Simload Base Counters, Telemetry Collector, Data Miner

Performance API Architecture



Windows Performance Data Helper DLL – PDH.dll



API: https://learn.microsoft.com/en-us/windows/win32/api/pdh/

Relevant RDSH/AVD/Win365 Perf Counters

\Processor(_Total)\% Processor Time
\System\Processor Queue Length
\System\Context Switches/sec
\System\Processes
\Memory\Available MBytes
\Process(_Total)\Working Set
\PhysicalDisk(_Total)\Disk Read Bytes/sec
\PhysicalDisk(_Total)\Disk Write Bytes/sec
\PhysicalDisk(_Total)\Disk Transfers/sec
\PhysicalDisk(_Total)\Avg. Disk Queue Length

Core Telemetry / Base Counters

\Terminal Services\Active Sessions
\Terminal Services\Total Sessions
\Processor(_Total)\Interrupts/sec
\Processor(_Total)\% Interrupt Time
\System\File Read Operations/sec
\System\File Write Operations/sec
\Memory\Free System Page Table Entries
\Memory\Page Faults/sec
\Memory\Pages/sec
\Memory\Pool Nonpaged Bytes
\Memory\Pool Paged Bytes

Relevant RDSH/AVD/Win365 Perf Counters

\Terminal Services Session(RDP-Tcp x)\% Processor Time

\Terminal Services Session(RDP-Tcp x)\Working Set

\User Input Delay per Session(Max)\Max Input Delay

\Network Interface(*)\Output Queue Length

\Network Interface(*)\Current Bandwidth

\Network Interface(*)\Packets Received Unicast/sec

\Network Interface(*)\Packets Received Non-Unicast/sec

\Network Interface(*)\Packets Sent Unicast/sec

\Network Interface(*)\Packets Sent Non-Unicast/sec

\Network Interface(*)\Bytes Received/sec

\Network Interface(*)\Bytes Sent/sec

\Network Interface(*)\Bytes Total/sec

No _Total Instance

Relevant RDSH/AVD/Win365 Perf Counters

\RemoteFX Network(*)\Current TCP Bandwidth	TCP Bandwidth detected in bits per second (bps)
\RemoteFX Network(*)\Current TCP RTT	Average TCP round-trip time (RTT) detected in ms
\RemoteFX Network(*)\Current UDP Bandwidth	UDP Bandwidth detected in bits per second (bps)
\RemoteFX Network(*)\Current UDP RTT	Average UDP round-trip time (RTT) detected in ms
\RemoteFX Network(*)\Loss Rate	Loss percentage
\RemoteFX Network(*)\Retransmission Rate	Percentage of packets that have been retransmitted
\RemoteFX Network(*)\TCP Received Rate	Rate in bps at which data is received over TCP
\RemoteFX Network(*)\TCP Sent Rate	Rate in bps at which data is sent over TCP
\RemoteFX Network(*)\UDP Received Rate	Rate in bps at which data is received over UDP
\RemoteFX Network(*)\UDP Sent Rate	Rate in bps at which data is sent over UDP

Relevant RDSH/AVD/Win365 Perf Counters

\RemoteFX Graphics(*)\Graphics Compression ratio	Ratio of bytes encoded to bytes input
\RemoteFX Graphics(*)\Average Encoding Time	Average frame encoding time
\RemoteFX Graphics(*)\Frame Quality	Quality of the output frame
\RemoteFX Graphics(*)\Input Frames/second	Number of sources frames
\RemoteFX Graphics(*)\Output Frames/second	Number of frames sent to the client
\RemoteFX Graphics(*)\Source Frames/second	Number of frames composed by source

\RemoteFX Graphics(*)\Frames Skipped/second – Insufficient Client Resources

\RemoteFX Graphics(*)\Frames Skipped/second – Insufficient Network Resources

\RemoteFX Graphics(*)\Frames Skipped/second – Insufficient Server Resources

Citrix HDX on Windows 365

\ICA Session(Console (UserName))\EDT bandwidth available (bps)

\ICA Session(Console (UserName))\Input Session Bandwidth

\ICA Session(Console (UserName))\Output Session Bandwidth

\ICA Session(Console (UserName))\Input Session Compression

\ICA Session(Console (UserName))\Output Session Compression

\ICA Session(Console (UserName))\Latency - Last Recorded

\ICA Session(Console (UserName))\Latency - Session Average

\ICA Session(Console (UserName))\EDT RTT

\ICA Graphics(<n>)\BitRate - Current

\ICA Graphics(<n>)\Frame Rate - Current FPS

\ICA Graphics(<n>)\Frame Rate - Input FPS

\ICA Graphics(<n>)\Frame Rate - Target FPS

\ICA Graphics(<n>)\Thinwire Average Input FPS

\ICA Graphics(<n>)\Thinwire Average Output FPS

\ICA Graphics(<n>)\Thinwire Bandwidth Estimate (bps)

\ICA Graphics(<n>)\Thinwire Latency Estimate (ms)

Omnissa Blast on Windows 365

\Horizon Blast Session Counters(Session ID: n; (main))\Estimated Bandwidth (Uplink)
\Horizon Blast Session Counters(Session ID: n; (main))\Instantaneous Transmitted Bytes over TCP
\Horizon Blast Session Counters(Session ID: n; (main))\Instantaneous Transmitted Bytes over UDP
\Horizon Blast Session Counters(Session ID: n; (main))\Instantaneous Received Bytes over TCP
\Horizon Blast Session Counters(Session ID: n; (main))\Instantaneous Received Bytes over UDP
\Horizon Blast Session Counters(Session ID: n; (main))\Packet Loss (Uplink)
\Horizon Blast Session Counters(Session ID: n; (main))\Jitter (Uplink)
\Horizon Blast Session Counters(Session ID: n; (main))\RTT
\Horizon Blast Session Counters(Session ID: n; (main))\Automatic Reconnect Count
\Horizon Blast Imaging Counters(Session ID: n; Channel: Imaging; (main))\Frames per second
\Horizon Blast Imaging Counters(Session ID: n; Channel: Imaging; (main))\Dirty frames per second
\Horizon Blast Imaging Counters(Session ID: n; Channel: Imaging; (main))\FBC Rate
\Horizon Blast Imaging Counters(Session ID: n; Channel: Imaging; (main))\Poll Rate
\Horizon Blast Imaging Counters(Session ID: n; Channel: Imaging; (main))\Outbound Bandwidth (Kbps)
\Horizon Blast Imaging Counters(Session ID: n; Channel: Imaging; (main))\Inbound Bandwidth (Kbps)
\Horizon Blast Imaging Counters(Session ID: n; Channel: Imaging; (main))\Out Queueing time (us)

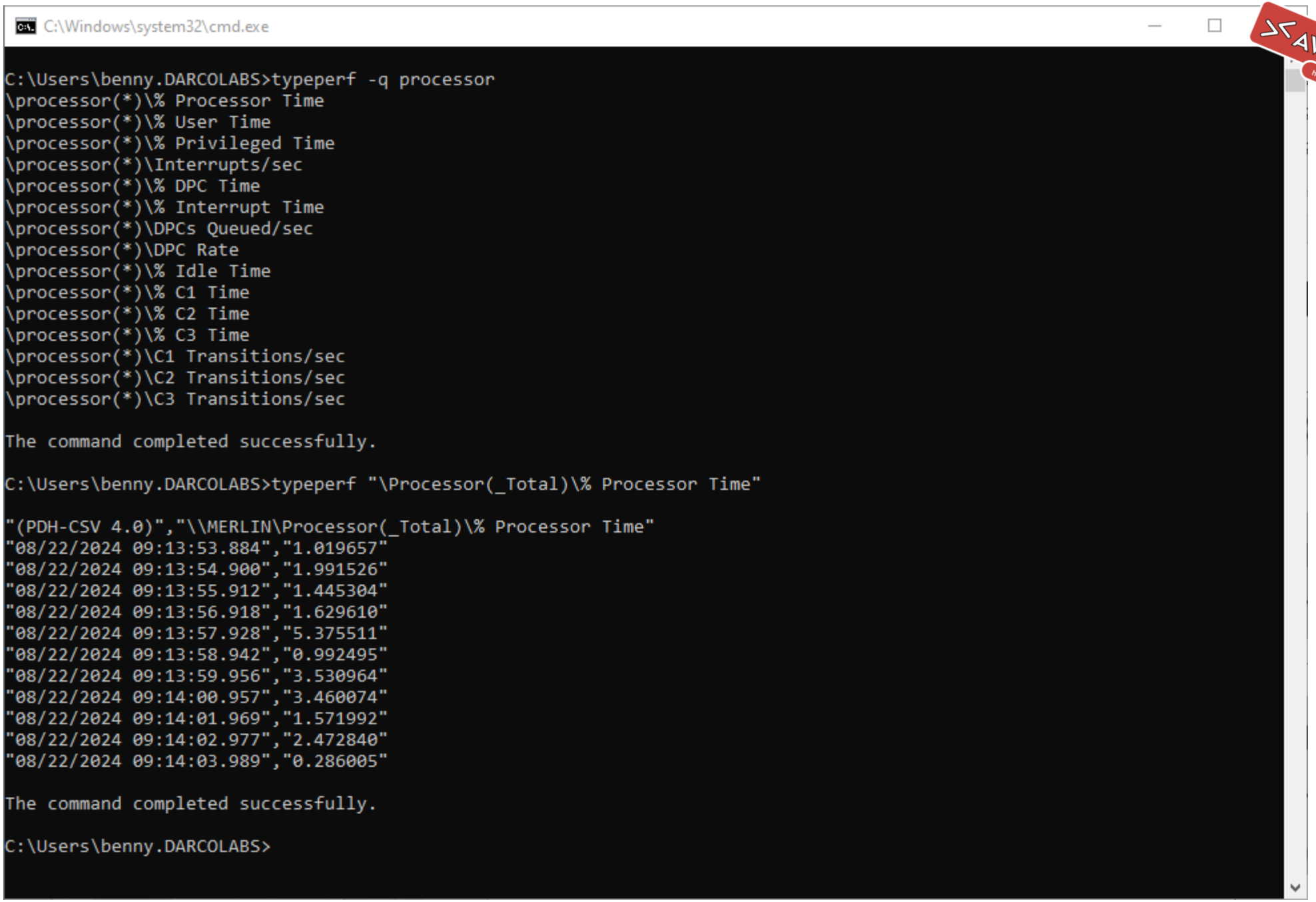
Collecting Performance Metrics

Command Line, Scripting and more…

Command Line Tools

- **Typeperf** writes performance data to the command window or to a log file https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/typeperf
- **Logman** creates and manages Event Trace Session and Performance logs and supports many functions of Performance Monitor from the command line
- **Relog** extracts performance counters from performance counter logs into other formats, such as text-TSV (for tab-delimited text), text-CSV (for comma-delimited text), binary-BIN (BLG), or SQL

These tools are very powerful, but not easy to use


C:\Windows\system32\cmd.exe
C:\Users\benny.DARCOLABS>typeperf -q processor
\processor(*)\% Processor Time
\processor(*)\% User Time
\processor(*)\% Privileged Time
\processor(*)\Interrupts/sec
\processor(*)\% DPC Time
\processor(*)\% Interrupt Time
\processor(*)\DPCs Queued/sec
\processor(*)\DPC Rate
\processor(*)\% Idle Time
\processor(*)\% C1 Time
\processor(*)\% C2 Time
\processor(*)\% C3 Time
\processor(*)\C1 Transitions/sec
\processor(*)\C2 Transitions/sec
\processor(*)\C3 Transitions/sec
The command completed successfully.
C:\Users\benny.DARCOLABS>typeperf "\Processor(_Total)\% Processor Time"
"(PDH-CSV 4.0)","\\MERLIN\Processor(_Total)\% Processor Time"
"08/22/2024 09:13:53.884","1.019657"
"08/22/2024 09:13:54.900","1.991526"
"08/22/2024 09:13:55.912","1.445304"
"08/22/2024 09:13:56.918","1.629610"
"08/22/2024 09:13:57.928","5.375511"
"08/22/2024 09:13:58.942","0.992495"
"08/22/2024 09:13:59.956","3.530964"
"08/22/2024 09:14:00.957","3.460074"
"08/22/2024 09:14:01.969","1.571992"
"08/22/2024 09:14:02.977","2.472840"
"08/22/2024 09:14:03.989","0.286005"
The command completed successfully.
C:\Users\benny.DARCOLABS>

PowerShell

Performance Counters

- Get-Counter -ListSet "Processor"
- (Get-Counter -ListSet "Processor").Paths
- (Get-Counter -ListSet "Processor").PathsWithInstances
- Get-Counter -Counter "\Processor(_Total)\% Processor Time" -SampleInterval 2 -MaxSamples 3
- $CounterList = "\Processor(_Total)\% Processor Time", "\System\Processor Queue Length", "\Memory\Available MBytes", "\Process(_Total)\Working Set", "\PhysicalDisk(_Total)\Disk Read Bytes/sec", "\PhysicalDisk(_Total)\Disk Write Bytes/sec", "\PhysicalDisk(_Total)\Disk Transfers/sec", "\PhysicalDisk(_Total)\Current Disk Queue Length", "\System\Context Switches/sec", "\System\Processes“
- Get-Counter -Counter $CounterList -SampleInterval 1 -MaxSamples 20
- https://learn.microsoft.com/en-us/powershell/module/microsoft.powershell.diagnostics/get-counter

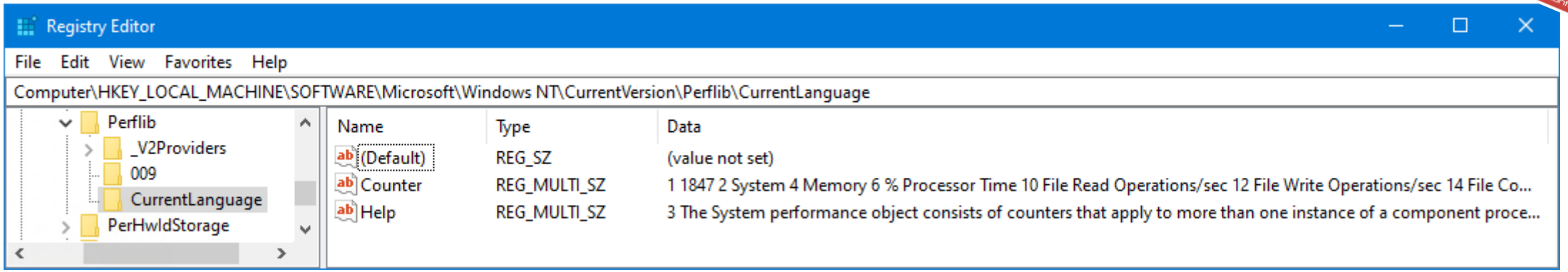
Dealing With Localized Counter Names

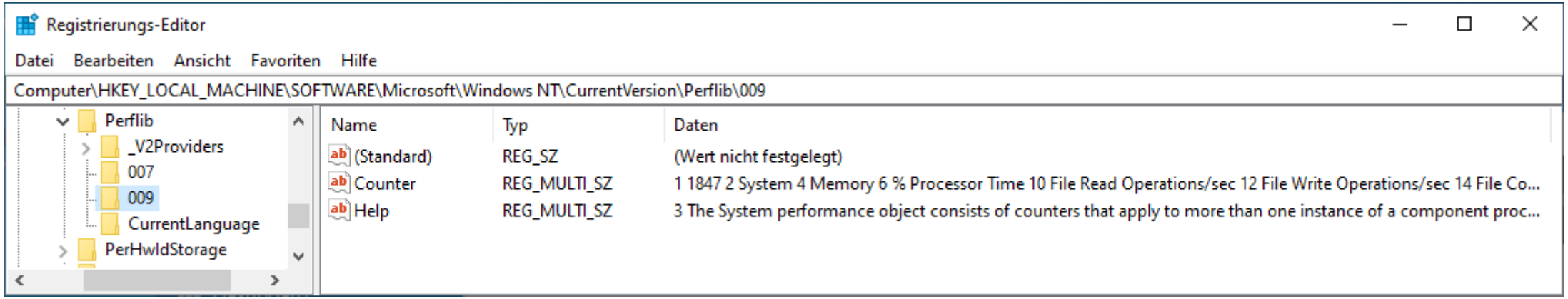
- The most severe limitation of Get-Counter are the localized counter names
- There are two API functions you can use to convert localized counter names to id numbers and vice versa
  - Get-PerformanceCounterId takes a localized performance counter name and translates it to a language-agnostic id number
  - Get-PerformanceCounterLocalName does the opposite and translates the id number to the appropriate local name

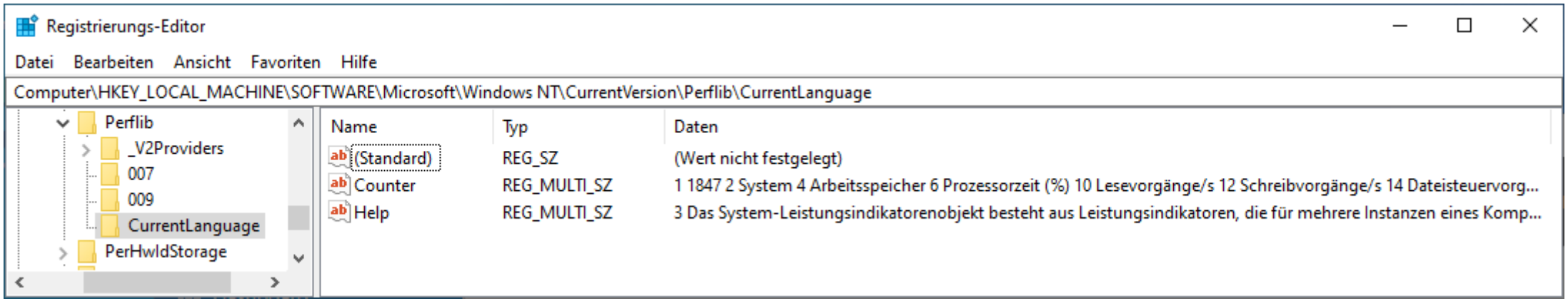
https://powershell.one/tricks/performance/performance-counters

https://powershellmagazine.com/2013/07/19/querying-performance-counters-from-powershell/

Localized Perf Counter Names



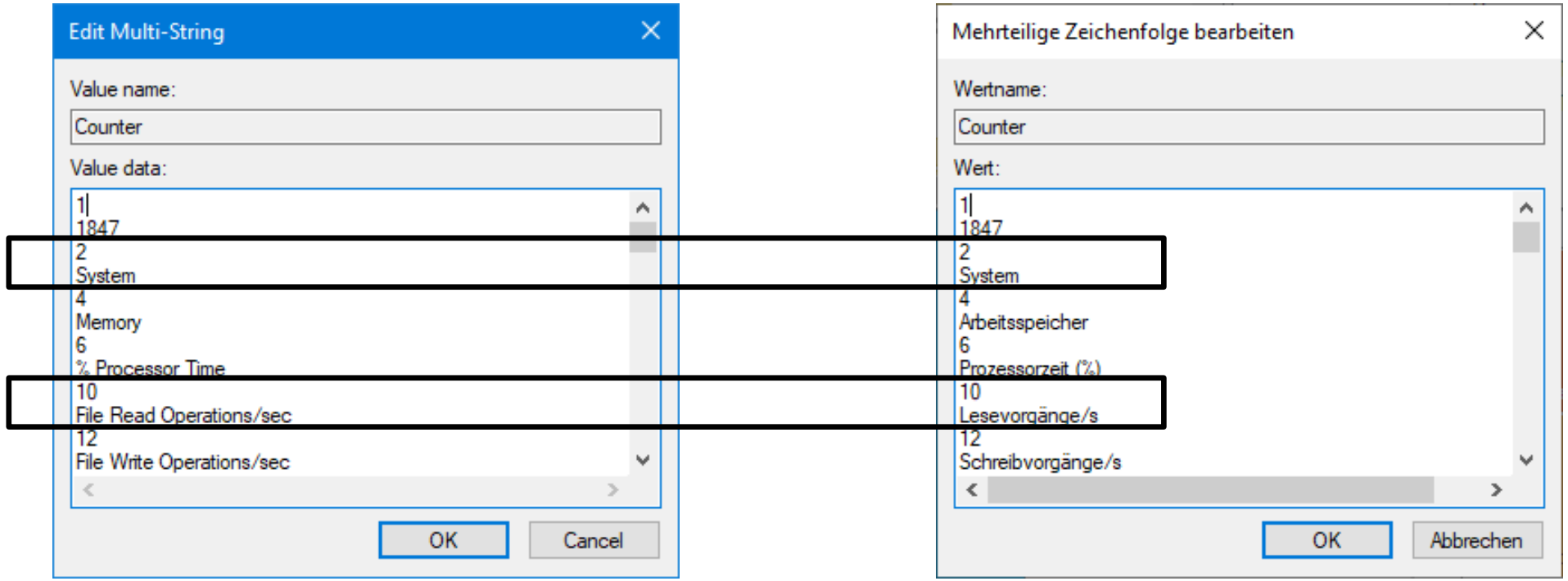




Localized Perf Counter Names

Perflib\CurrentLanguage\Counter
on a system with English (United States) language

Perflib\CurrentLanguage\Counter
on a system with German language



Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Perflib\CurrentLanguage

Get-PerformanceCounterId


```powershell
function Get-PerformanceCounterId
{
    param(
        [Parameter(Mandatory)]
        [string] $Name,
        $ComputerName = $env:COMPUTERNAME
    )

    $code = '[DllImport("pdh.dll", SetLastError=true, CharSet=CharSet.Unicode)]public static extern
UInt32   PdhLookupPerfIndexByName(string szMachineName, string szNameBuffer, ref uint
dwNameIndex);'
    $type = Add-Type -MemberDefinition $code -PassThru -Name PerfCounter2 -Namespace Utility

    [UInt32]$Index = 0
    if ($type::PdhLookupPerfIndexByName($ComputerName, $Name, [Ref]$Index) -eq 0)
    {
      $index
    }
    else
    {
      throw "Cannot find '$Name' on '$ComputerName'."
    }
}
```

Localized Perf Counter Names


```
PS F:\> Get-Counter -Counter "\System\File Read Operations/sec"

Timestamp                 CounterSamples
---------                 --------------
7/30/2020 2:24:42 PM      \\host1\system\file read operations/sec :
                          359.52894517603



PS F:\> Get-Counter -Counter "\2\10"

Timestamp                 CounterSamples
---------                 --------------
7/30/2020 2:24:51 PM      \\host1\2\10 :
                          341.618616976008
```

- Unfortunately, many “localized” counter IDs do not work with Get-Counter
- And the decimal separator – period (.) or a comma (,) – depends on the locale…
- …as does the 1000 separator (expect big fun with CSV files)

Language-Agnostic Counters

Simload.ini

```
[Telemetry]
Name1=CPU|%
Counter1=\Processor(_Total)\% Processor Time
Name2=CPU Queue Length
Counter2=\System\Processor Queue Length
Name3=Memory Available|MBytes
Counter3=\Memory\Available MBytes
Name4=Working Set|Bytes
Counter4=\Process(_Total)\Working Set
Name5=Disk Reads|Bytes/sec
Counter5=\PhysicalDisk(_Total)\Disk Read Bytes/sec
Name6=Disk Writes|Bytes/sec
Counter6=\PhysicalDisk(_Total)\Disk Write Bytes/sec
Name7=Disk IOPS
Counter7=\PhysicalDisk(_Total)\Disk Transfers/sec
Name8=Disk Avg. Queue Length
Counter8=\PhysicalDisk(_Total)\Avg. Disk Queue Length
Name9=Context Switches/sec
Counter9=\System\Context Switches/sec
Name10=Processes
Counter10=\System\Processes
```

AutoIt Code

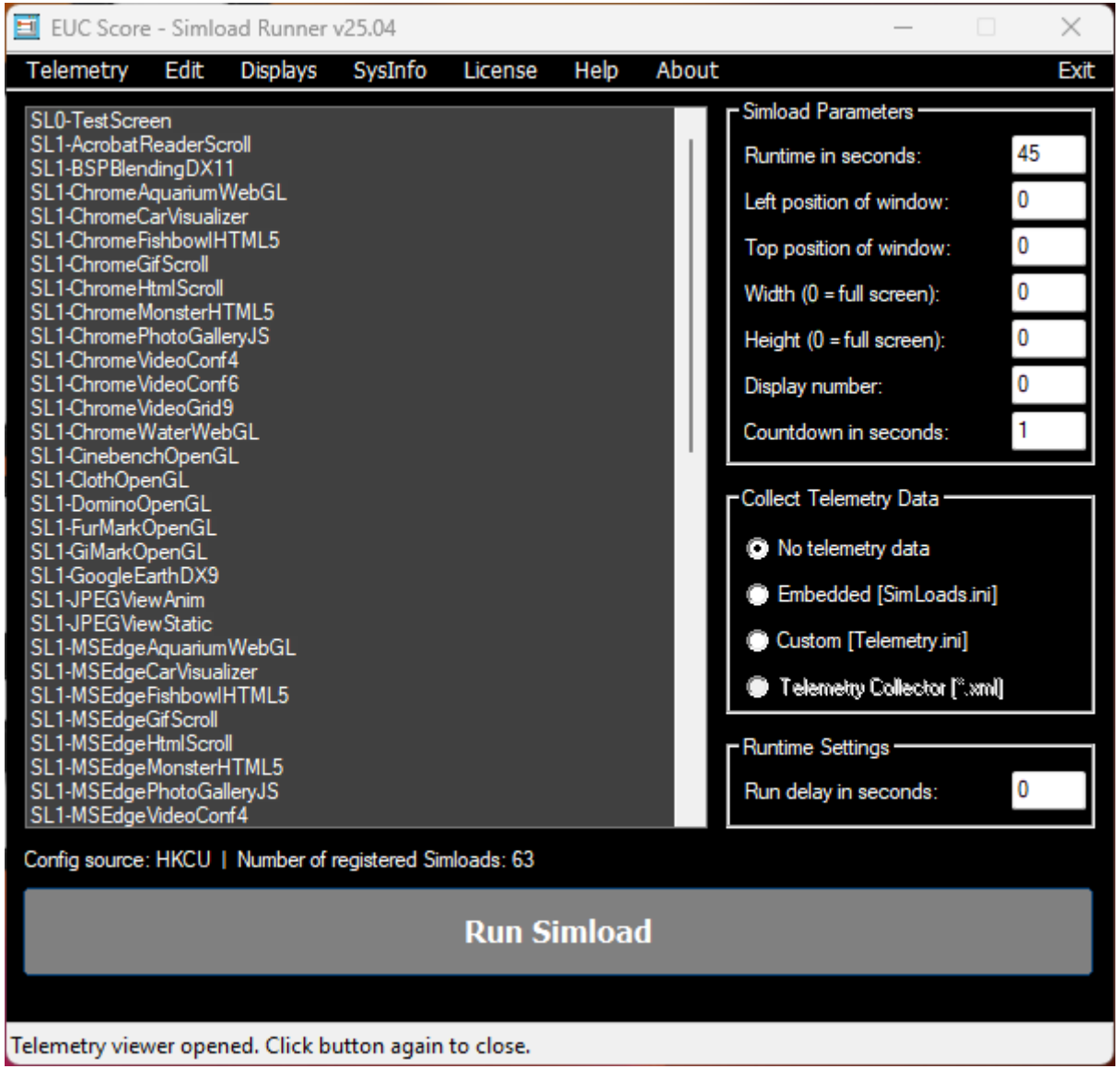
```
Name1=CPU|%
Counter1=:238\6\(_Total)
Name2=CPU Queue Length
Counter2=:2\44\
Name3=Memory Available|MBytes
Counter3=:4\1382\
Name4=Working Set|Bytes
Counter4=:230\180\(_Total)
Name5=Disk Reads|Bytes/sec
Counter5=:234\220\(_Total)
Name6=Disk Writes|Bytes/sec
Counter6=:234\222\(_Total)
Name7=Disk IOPS
Counter7=:234\212\(_Total)
Name8=Disk Avg. Queue Length
Counter8=:234\1400\(_Total)
Name9=Context Switches/sec
Counter9=:2\146\
Name10=Processes
Counter10=:2\248\
```

Get-Counter / TypePerf

```
CPU|%
-Counter "\238(_Total)\6"
CPU Queue Length
-Counter "\2\44"
Memory Available
-Counter "\4\1382"
Working Set
-Counter "\230(_Total)\180"
Disk Reads
-Counter "\234(_Total)\220"
Disk Writes
-Counter "\234(_Total)\222"
Disk IOPS
-Counter "\234(_Total)\212"
Disk Avg. Queue Length
-Counter "\234(_Total)\1400"
Context Switches/sec
-Counter "\2\146"
Processes
-Counter "\2\248"
```

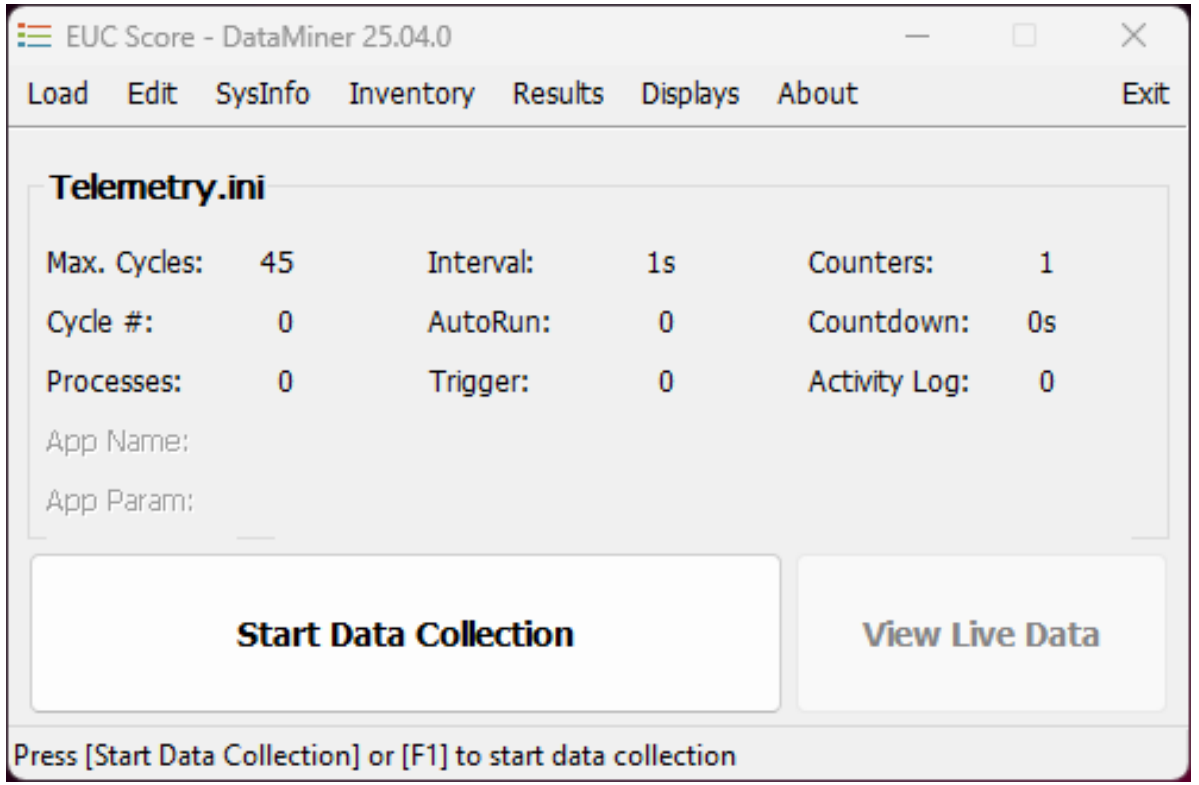
EUC Score Simload Base Counters

Free Download: EUC Score Base Package
https://eucscore.com/freeware



EUC Score Data Miner

- Data Miner integrates several supplementary tools
  - **Collects performance counters independently of Simloads**
  - Writes system information to a text file in the results folder
  - Writes multiple CSV files with inventory data in the results folder
  - Collects process information
  - Launches applications
- Data Miner can be used stand-alone by copy & paste deployment



Data Miner runs both with a GUI and from the command line

Free Data Miner Download: https://eucscore.com/freeware | Docs: https://eucscore.com/docs/tools.html

AutoIt – _PDH_PerformanceCounters

```
Func _PDH_GetCounterList($sCounterWildcardPath,$bReturnAsString=False)
    Local $aRet,$stExpandedPathList
    Local $hPDHDLL,$iBufSize,$sCounterList,$aCounterList[1]=[0]

    If Not IsString($sCounterWildcardPath) Then Return SetError(1,0,$aCounterList)

    ; Unlike other functions, getting a counter list doesn't require initialization,
    ;   though it doesn't hurt (especially if Disable Performance Counters is set)
    If Not $_PDH_bInit Then
        $hPDHDLL="pdh.dll"
    Else
        $hPDHDLL=$_PDH_hDLLHandle
    EndIf

    _PDH_DebugWrite("_PDH_GetCounterList() call, $sCounterWildcardPath='" & $sCounterWildcardPath & _
        "', PDH DLL 'handle' (or just 'pdh.dll'):" & $hPDHDLL)

    ; Non-localized string? Create localized string and add it.
    If StringLeft($sCounterWildcardPath,1)=':' Then
        $sCounterWildcardPath=__PDH_LocalizeCounter($sCounterWildcardPath)
        If @error Then Return SetError(@error,0,"")
        _PDH_DebugWrite("Localized *wildcard* counter (from non-localized string):"&$sCounterWildcardPath)
    EndIf

    ; 1st call to PdhExpandWildCardPathW - get required buffer size
    $aRet=DllCall($hPDHDLL,"long","PdhExpandWildCardPathW","ptr",ChrW(0), _
        "wstr",$sCounterWildcardPath,"ptr",ChrW(0),"dword*",$iBufSize,"dword",0)
    If @error Then Return SetError(2,@error,$aCounterList)  ; DLL Call error
```

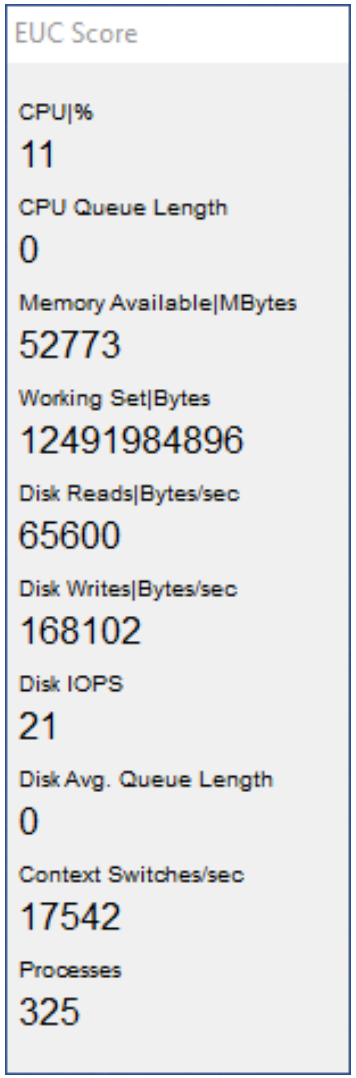
CSV Result Files

TimeStamp|1000	CPU|%	CPU Queue Length	Memory Available|MBytes	Working Set|Bytes	Disk Reads|Bytes/sec	Disk Writes|Bytes/sec	Disk IOPS	Disk Queue Length	Context Switches/sec	Processes
2024.08.08 10:30:36.730	0	1	48828	1.721609E+10	0	0	0	0	0	350
2024.08.08 10:30:37.802	16.0371	0	48808	1.72669E+10	1589410	623872.5	96.00307	0	41422.38	351
2024.08.08 10:30:38.823	16.24802	0	48806	1.728885E+10	2385041	0	19.6916	0	55233.54	352
2024.08.08 10:30:39.823	12.55208	0	48804	1.728939E+10	1083131	354221.5	30.81559	0	24862.56	352
2024.08.08 10:30:40.834	14.79185	0	48810	1.728841E+10	0	158669.5	26.8188	0	25856.44	352
2024.08.08 10:30:41.850	13.87708	0	48808	1.729144E+10	1009577	204725.7	46.0234	0	32820.31	352
2024.08.08 10:30:42.861	8.800934	1	48804	1.728834E+10	0	89528.84	6.960476	0	29901.39	352
2024.08.08 10:30:43.859	13.36615	0	48805	1.728833E+10	0	159098.4	26.89124	0	34092.06	352
2024.08.08 10:30:44.886	17.26958	0	48796	1.729805E+10	0	124779.1	24.56684	0	44967.18	352
2024.08.08 10:30:45.888	15.71634	0	48800	1.729505E+10	0	162253	24.73013	0	41759.28	352
2024.08.08 10:30:46.913	14.22636	0	48800	1.729274E+10	0	622944	21.60948	0	34674.31	352
2024.08.08 10:30:47.895	9.426932	0	48798	1.729646E+10	0	421994.3	42.42533	0	32910.01	352
2024.08.08 10:30:48.913	9.435274	0	48801	1.729297E+10	0	181588.1	34.47602	0	32947.26	352
2024.08.08 10:30:49.922	12.37798	0	48800	1.732194E+10	0	232905.4	36.62416	0	32353.19	353
2024.08.08 10:30:50.965	14.22228	0	48790	1.732325E+10	94723.96	146031	19.42027	0	32691.35	353
2024.08.08 10:30:51.940	13.66384	0	48792	1.732352E+10	0	182360	30.35652	0	34137.27	353
2024.08.08 10:30:52.945	12.97976	0	48784	1.733555E+10	0	216170.4	31.86481	0	33511.35	353
2024.08.08 10:30:53.964	8.67326	0	48781	1.733713E+10	0	81109.59	14.8483	0	29580.76	353
2024.08.08 10:30:54.988	9.81427	0	48784	1.733138E+10	0	63610.1	5.824227	0	32242.77	353
2024.08.08 10:30:55.981	7.289457	0	48784	1.733362E+10	0	140021.6	16.08899	0	31924.67	353

What is missing?

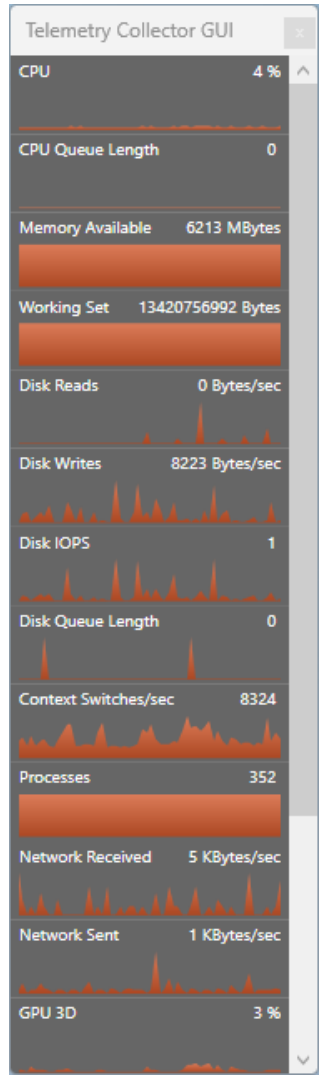
- GPU Performance Counters
  - 3D (rendering)
  - Video Decode
  - Video Encode (NVIDIA) / Video Processing (AMD, INTEL)
  - Dedicated Memory (Frame Buffer)
- Network Performance Counters
  - \Network Interface(_Total)\Bytes Received/sec
  - \Network Interface(_Total)\Bytes Sent/sec
- User Session ID – Auto-Detect

Beyond Standard Performance Counters



Core Telemetry / Base Counters

Avatar Telemetry Collector



Perf Counters in EUC Score Telemetry Collector

Components	Performance Counters
CPU	\Processor(_Total)\% Processor Time \System\Processor Queue Length
Memory	\Memory\Available Mbytes \Process(_Total)\Working Set
Storage	\PhysicalDisk(_Total)\Disk Read Bytes/sec \PhysicalDisk(_Total)\Disk Write Bytes/sec \PhysicalDisk(_Total)\Disk Transfers/sec (IOPS) \PhysicalDisk(_Total)\Current Disk Queue Length
System	\System\Context Switches/sec \System\Processes
Network	TC::network received(_Total) TC::network sent(_Total)
GPU	TC::GPU load(_Total)\3D TC::GPU load(_Total)\Video Decode TC::GPU load(_Total)\Video Processing TC::GPU frame buffer(_Total)

Telemetry Collector – TelemetryDataConfig.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<TelemetryDataConfig>
  <RefreshRate>1000</RefreshRate>
  <CounterDefinitions>
    <!-- Standard Counters -->
    <Counter Name="CPU" Unit="%">
                        <CategoryName>Processor</CategoryName>
                        <CounterName>% Processor Time</CounterName>
                        <InstanceName>_Total</InstanceName>
    </Counter>
    <Counter Name="CPU Queue Length">
                        <CategoryName>System</CategoryName>
                        <CounterName>Processor Queue Length</CounterName>
    </Counter>
    <Counter Name="Memory Available" Unit="MBytes">
                        <CategoryName>Memory</CategoryName>
                        <CounterName>Available Mbytes</CounterName>
    </Counter>
  </CounterDefinitions>
</TelemetryDataConfig>
```

C++ Code Base
Single-instance counters plus additional metrics

TC-specific Metrics

```
<Counter Name="Network Received" Unit="KBytes/sec">
        <CategoryName>TC::network received</CategoryName>
        <InstanceName>_Total</InstanceName>
</Counter>
<Counter Name="Network Sent" Unit="KBytes/sec">
        <CategoryName>TC::network sent</CategoryName>
        <InstanceName>_Total</InstanceName>
</Counter>
<Counter Name="GPU 3D" Unit="%">
        <CategoryName>TC::GPU load</CategoryName>
        <CounterName>3D</CounterName>
        <InstanceName>_Total</InstanceName>
</Counter>
<Counter Name="GPU Video Decode" Unit="%">
        <CategoryName>TC::GPU load</CategoryName>
        <CounterName>Video Decode</CounterName>
        <InstanceName>_Total</InstanceName>
</Counter>
```

```
<Counter Name="GPU Video Processing" Unit="%">
        <CategoryName>TC::GPU load</CategoryName>
        <CounterName>Video Processing</CounterName>
        <InstanceName>_Total</InstanceName>
</Counter>
<Counter Name="GPU Memory" Unit="MBytes">
        <CategoryName>TC::GPU frame buffer</CategoryName>
        <InstanceName>_Total</InstanceName>
</Counter>
<Counter Name="Session CPU" Unit="%">
        <CategoryName>Terminal Services Session</CategoryName>
        <CounterName>% Processor Time</CounterName>
        <InstanceName>TC::current RDP session</InstanceName>
</Counter>
```

Call to Action

If you want to learn more about EUC Score, send me an email

info@eucscore.com

EUC Score

https://eucscore.com
https://eucscore.com/results

NOTE: The EUC Score toolset is free for community benchmarking tests when the results are made freely available to the public

Thank You

Benny Tritsch | info@eucscore.com