Mastering Your Logging Ninja Skills with LogAnalytics v2

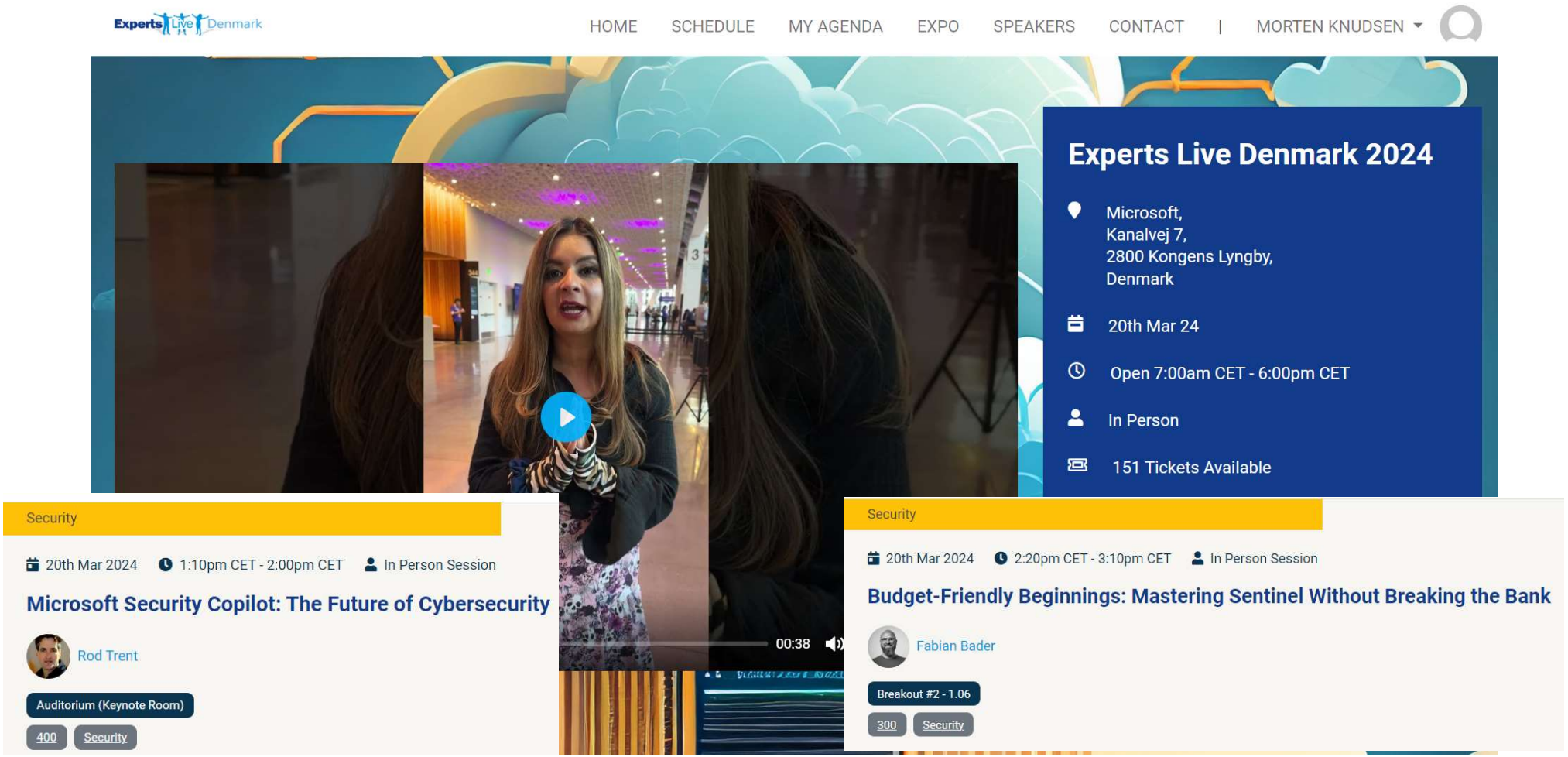
Morten Knudsen
Microsoft MVP, MCT, Cloud & Security Architect

Sponsors

Morten Knudsen

- Security & Cloud Architect – Denmark - freelance
- Focus: Technology – Processes – People - Cost
  - Microsoft MVP Security & Azure Hybrid MVP
  - Microsoft Certified Trainer
- Blog | Speak | Co-founder Experts Live Denmark | Mentorship | Feedback Microsoft product-team

Please join us in Copenhagen - conference.expertslive.dk



Session Objective – hopefully, something for all ☺

Audience - we come from different background? Competency?

Introduction & Inspiration

- Get inspired about Azure Logging capabilities - use-cases

How-to migrate from Azure Logging V1 -> V2?

- MMA → AMA -> End of Support August 2024
- HTTP Data Collector REST API → Azure Log Ingestion Pipeline, Log Ingestion API, Azure LogAnalytics custom tables v1 → v2 – End of Support September 2026
- (new) Data manipulation – Schema/Table Management - Data Transformation

How can log data be used today (just a few samples) ?

Security & Compliance

SIEM (use-cases)

Secure Score (Desired State)

Audit-logs

Operation

Availability

Performance

Troubleshooting

Who?

What

When

Why?

Typically available as standard features from vendors

WHY ingesting custom log data can be of value

Examples of Use Cases

- Are we in control?
  - Bitlocker, Local admins, Windows Update
- Prepare business case
  - Detect local printers – replace with large multi-function printers
- Budget for next year
  - PC replacement - Lenovo / Dell warranty lookup
- SQL Monitoring
  - Troubleshooting & Performance
- Backup monitoring
  - Collection of TSM Spectrum log-files
- Production monitoring
  - Monitoring of Production environment

SERVER KPI STATUS | MANAGED D...
Shared dashboard

+ Create | Upload | Refresh | Full screen | Edit | Manage sharing | Export | Clone | Assign tags | Delete | Feedback

Auto refresh : Off | UTC Time : Past 24 hours

INCOMPLIANT AGENT VMCONNECTION (LAST 24H) | SERVERS...
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
9

INCOMPLIANT AGENT EVENT (LAST 24H) | SERVERS | COUNT
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
6

INCOMPLIANT MDE AGENT NOT_ONBOARDED | SERVERS | CO...
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
1

INCOMPLIANT AGENT HEARTBEAT (LAST 24H) | SERVERS | CO...
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
4

INCOMPLIANT DEFENDER AV | SERVERS | COUNT
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
0

INCOMPLIANT MDE AGENT NO_SENSORDATA (LAST 24H) | SE...
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
The query returned no results.

INCOMPLIANT AGENT SECURITYEVENT (LAST 24H) | SERVERS |...
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
6

LAST GROUP POLICY REFRESH MORE THAN 7 DAYS AGO | SER...
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
0

INCOMPLIANT MDE AGENT INACTIVE_NOT_DECOMISSIONED |...
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
The query returned no results.

DEVICES WITH COLLECTION ISSUES (FIX WMI) | SERVERS | CO...
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
The query returned no results.

NO RESTART MORE THAN 50 DAYS DAY | SERVERS | COUNT
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
6

INCOMPLIANT LAPS | SERVERS | COUNT
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
38

PENDING UPDATES OLDER THAN 40 DAYS | SERVERS | COUNT
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
0

PENDING UPDATES, NO DRV, OLDER THAN 40 DAYS | SERVERS...
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
0

DEVICES WITH NON-STANDARD MEMBERS LOCAL ADMINS G...
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
59

LAST WINDOWS UPDATE MORE THAN 40 DAYS AGO | SERVER...
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
6

INCOMPLIANT WINDOWS FIREWALL | SERVERS | COUNT
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
0

DEVICES IN INVENTOR
log-platform-management-srv...
5

DISCONNECTED ARC STATUS | SERVERS | COUNT
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
49

FAILED ARC ONBOARDING | SERVERS | COUNT
log-platform-management-srvnetworkcloud-p (@ 1/21 9:39 AM)
77

FAILED ARC ONBOARDING | SERVERS | LIST

Search

AD_CN	Arc_Onboarding_Status	A.	AD_DNSHostName	AD_IPv4Address
INVENTOR	Missing	K.	Inventor.FVF.local	192.168.152.62
CKN00GK085	Missing		CKN00GK085.FVF.local	10.81.0.85
CKN05GK036	Missing		CKN05GK036.FVF.local	10.81.5.36
RDS-HOST1	Missing		RDS-Host1.FVF.local	10.81.21.80
MEPLUS	Missing		MEPLUS.FVF.local	10.81.21.83
RPA03	Missing		RPA03.FVF.local	10.81.21.89
SAP-BW-DEV	Missing		SAP-BW-DEV.FVF.local	192.168.152.163
POWERBI-GATEWAY	Missing		PowerBI-Gateway.FVF.local	10.81.21.143
GIS-ENTERPR-3	Missing		GIS-EnterPR-3.FVF.local	10.81.21.147
CNN00GK061	Missing		CNN00GK061.FVF.local	10.81.0.61
CKN05GK125	Missing		CKN05GK125.FVF.local	10.81.5.125

NO RESTART MORE THAN 50 DAYS DAY | SERVERS | LIST

Computer	OsUptime_Days	CollectionTime
POWERBI-GATEWAY	311	1/21/2024, 2:30:06.000 AM
ANSYS	211	1/21/2024, 2:02:40.000 AM
GIS-APPL-2	82	1/21/2024, 2:10:32.000 AM
GIS-CLIENT-2	81	1/21/2024, 2:27:31.000 AM
VEEAM-ONE	74	1/21/2024, 2:24:11.000 AM
GIS-GEOCORTEX-2	52	1/21/2024, 2:02:27.000 AM

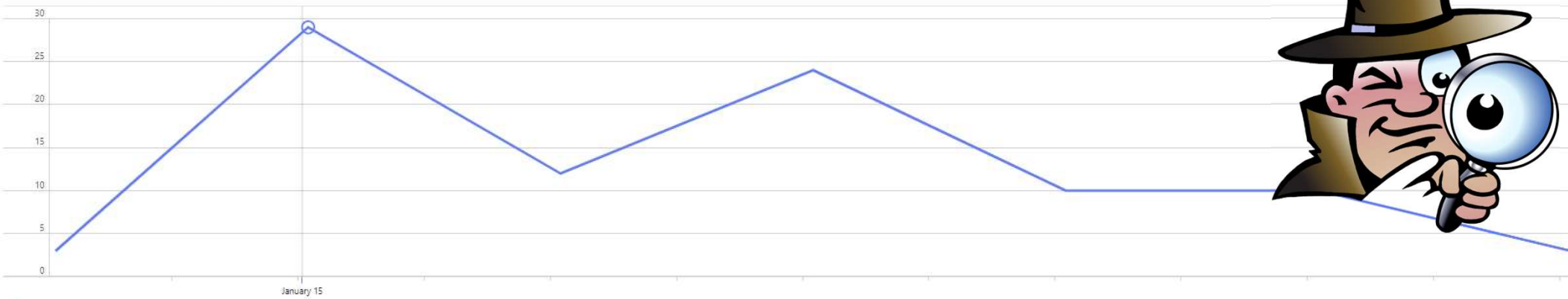
CLIENT KPI STATUS | MANAGED D...
Shared dashboard

Create | Upload | Refresh | Full screen | Edit | Manage sharing | Export | Clone | Assign tags | Delete | Feedback

Auto refresh : **Off** | UTC Time : **Past 24 hours**

UNEXPECTED SHUTDOWNS | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 98	INCOMPLIANT BITLOCKER | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 7	LAST GROUP POLICY REFRESH MORE THAN 7 DAYS AGO | CLIE... log-platform-management-client-p (@ 1/21 9:40 AM) 0	NO RESTART MORE THAN 7 DAYS DAY | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 111
OTHER PRIMARY ANTIVIRUS (SECURITY CENTER) | CLIENTS | C... log-platform-management-client-p (@ 1/21 9:40 AM) 0	DEVICES WITH COLLECTION ISSUES (FIX WMI) | CLIENTS | COU... log-platform-management-client-p (@ 1/21 9:40 AM) 0	INCOMPLIANT DEFENDER AV | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 13	INCOMPLIANT WINDOWS FIREWALL | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 0
PENDING UPDATES OLDER THAN 40 DAYS | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 54	PENDING UPDATES, NO DRV, OLDER THAN 40 DAYS | CLIENTS ... log-platform-management-client-p (@ 1/21 9:40 AM) 4	LAST WINDOWS UPDATE MORE THAN 40 DAYS AGO | CLIENT... log-platform-management-client-p (@ 1/21 9:40 AM) 0	INCOMPLIANT LAPS | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 54
UPDATING O365 OFFICE NOT ENABLED | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 0	INCOMPLIANT UPDATE CHANNEL O365 OFFICE | CLIENTS | CO... log-platform-management-client-p (@ 1/21 9:40 AM) 0	ONACCESSPROTECTION DISABLED DEFENDER AV | CLIENTS | C... log-platform-management-client-p (@ 1/21 9:40 AM) 0	PASSIVE / EDB BLOCK MODE DEFENDER AV | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 0
REALTIME PROTECTION NOT RUNNING DEFENDER AV | CLIEN... log-platform-management-client-p (@ 1/21 9:40 AM) 0	SIGNATURE TOO OLD DEFENDER AV | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 14	AM SERVICE NOT RUNNING DEFENDER AV | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 0	ANTISPYWARE NOT ENABLED DEFENDER AV | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 0
TAMPERPROTECTION NOT ENABLED DEFENDER AV | CLIENTS |... log-platform-management-client-p (@ 1/21 9:40 AM) 1	DEFENDER AV NOT INSTALED OR ENABLED | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 0	ANTISPYWARE NOT ENABLED DEFENDER AV | CLIENTS | COUNT log-platform-management-client-p (@ 1/21 9:40 AM) 0	AM VERSION TOO OLD DEFENDER AV | CLIEN... log-platform-management-client-p (@ 1/21 9:40 AM) 0

UNEXPECTED SHUTDOWNS | CLIENTS | TIME 7 DAYS



total
29

UNEXPECTED SHUTDOWNS | CLIENTS | GROUPED

Search

Group	MachineName	Message	TimeCreated	TimeGenerated	ActivityId	Bookmark	CollectionTime	Computer
FVF-woTYBxbgo2p.FVF.local (1)								
	FVF-woTYBxbgo2p.FVF.local	Den foregående systemlukning kl. 12:39:59 d. 22-12-2023...	12/22/2023, 1:09:36.973 PM	12/23/2023, 10:38:34.182 AM		{}	12/23/2023, 10:38:30.000 AM	FVF-WOTYBXBGO2P
PC1141.FVF.local (1)								
	PC1141.FVF.local	Den foregående systemlukning kl. 10:39:56 d. 17-01-2024...	1/17/2024, 10:50:28.471 AM	1/18/2024, 5:45:31.978 AM		{}	1/18/2024, 5:45:28.000 AM	PC1141
PC2002.FVF.local (2)								
	PC2002.FVF.local	Den foregående systemlukning kl. 14:47:32 d. 05-01-2024...	1/8/2024, 8:04:16.886 AM	1/8/2024, 8:17:37.248 AM		{}	1/8/2024, 8:17:34.000 AM	PC2002
	PC2002.FVF.local	Den foregående systemlukning kl. 14:01:01 d. 22-12-2023...	1/3/2024, 8:41:25.587 AM	1/4/2024, 7:48:43.570 AM		{}	1/4/2024, 7:48:33.000 AM	PC2002
PC2007.FVF.local (1)								
	PC2007.FVF.local	Den foregående systemlukning kl. 16:01:39 d. 17-01-2024...	1/17/2024, 6:21:51.637 PM	1/18/2024, 11:27:07.933 AM		{}	1/18/2024, 11:27:02.000 AM	PC2007
PC2012.FVF.local (1)								
	PC2012.FVF.local	Den foregående systemlukning kl. 18:54:26 d. 11-01-2024...	1/11/2024, 7:06:16.414 PM	1/12/2024, 1:41:51.479 PM		{}	1/12/2024, 1:41:45.000 PM	PC2012

BITLOCKER INCOMPLIANT | CLIENTS | LIST

Search

Computer	UserLoggedOn	CollectionTime	OSDisk_DriveLetter	OSDisk_EncryptionPercentage	OSDisk_VolumeStatus	OSDisk_ProtectionStatus	OSDisk_CapacityGB
FVF-LDMWUSHUHO0		1/19/2024, 3:13:01.000 AM	C:	0	0	0	98.39258
PC0724	FVF\uf	1/9/2024, 8:54:24.000 PM	C:	0	0	0	238.375977
PC0801	FVF\ast	1/19/2024, 8:18:31.000 AM	C:	0	0	0	237.2853
PC0832	FVF\nhn	1/19/2024, 8:08:50.000 AM	C:	0	0	0	237.288422
PC0833		1/18/2024, 6:16:37.000 PM	C:	0	0	0	237.357819
PC0844	FVF\lpo	1/21/2024, 8:47:56.000 AM	C:	0	0	0	237.359955
PC0845	FVF\tl	1/19/2024, 6:56:48.000 AM	C:	0	0	0	237.359955

INCOMPLIANT DEFENDER AV | CLIENTS | TIME 1 MONTH

total (Avg)
2

INCOMPLIANT DEFENDER AV | CLIENTS | LIST

Search

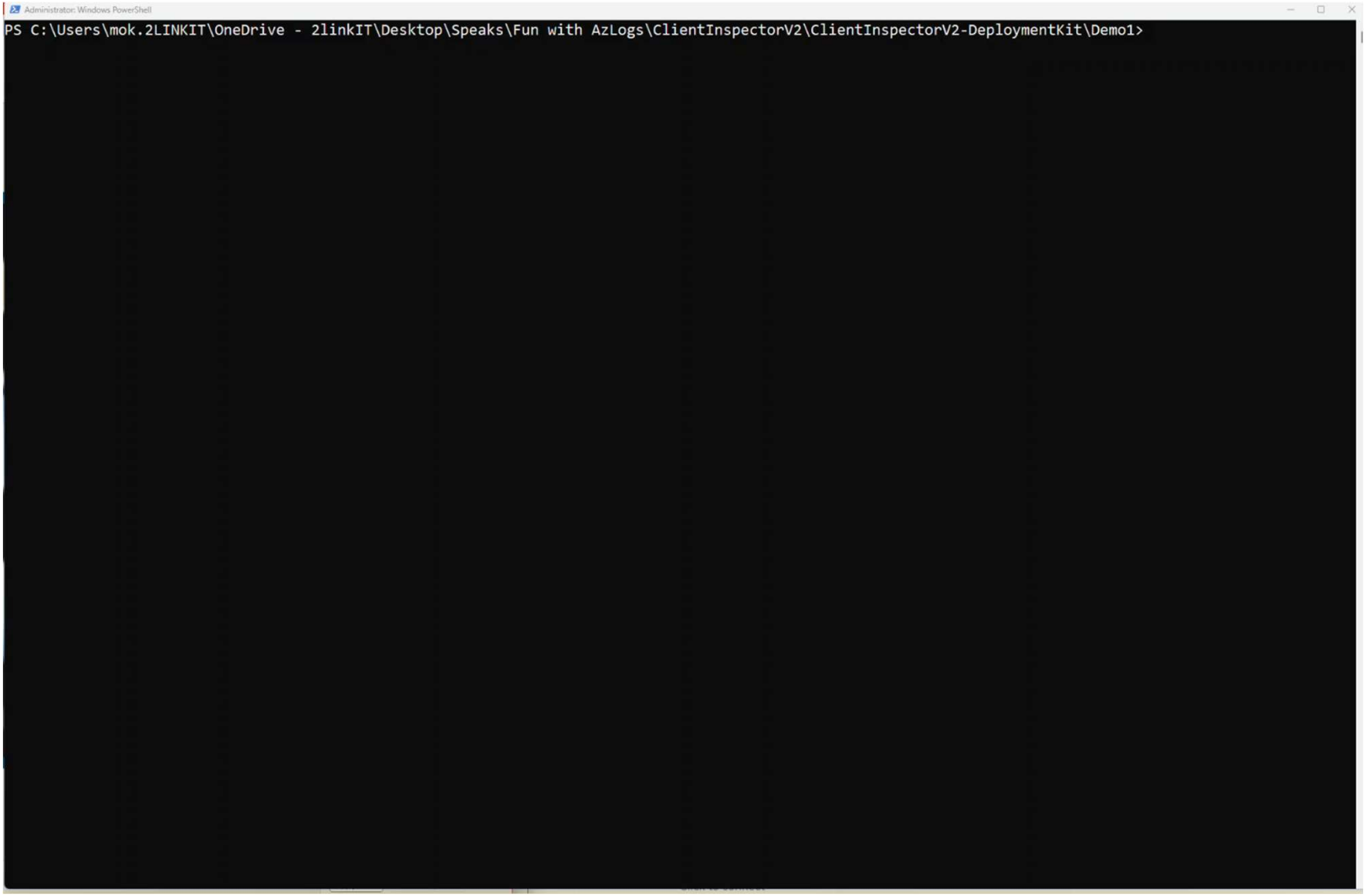
Computer	UserLoggedOn	CollectionTime	MPComputerStatusFound	MPPreferenceFound	AMEngineVersion	AMProductVersion	AMRunningMo...	AMServiceEnabled	AMServiceVersion	AntispywareEnabled	AntispywareSignat
PC1079	FVF\hcf	1/21/2024, 8:14:17.000 AM	true	true	1.1.23080.2005	4.18.23080.2006	Normal	true	4.18.23080.2006	true	
PC2126	FVF\sjo	1/21/2024, 5:59:40.000 AM	true	true	1.1.23090.2007	4.18.23090.2008	Normal	true	4.18.23090.2008	true	
PC0992	FVF\plm	1/19/2024, 10:54:56.000 AM	true	true	1.1.23060.1005	4.18.23050.9	Normal	true	4.18.23050.9	true	
PC2116	FVF\la	1/19/2024, 8:47:13.000 AM	true	true	1.1.23090.2007	4.18.23080.2006	Normal	true	4.18.23080.2006	true	
PC2117	FVF\kn	1/19/2024, 10:16:49.000 AM	true	true	1.1.23100.2009	4.18.23100.2009	Normal	true	4.18.23100.2009	true	
PC0762	FVF\hns	1/19/2024, 9:54:57.000 AM	true	true	1.1.23110.2	4.18.23100.2009	Normal	true	4.18.23100.2009	true	
PC2123	FVF\hec	1/18/2024, 7:24:35.000 AM	true	true	1.1.23100.2009	4.18.23070.1004	Normal	true	4.18.23070.1004	true	
PC2130	FVF\pss	1/19/2024, 6:59:06.000 AM	true	true	1.1.23100.2009	4.18.23100.2009	Normal	true	4.18.23100.2009	true	
PC0931	FVF\ma	1/18/2024, 8:05:46.000 AM	true	true	1.1.23090.2007	4.18.23090.2008	Normal	true	4.18.23090.2008	true	
PC2115	FVF\hjo	1/17/2024, 4:45:12.000 PM	true	true	1.1.23090.2007	4.18.2303.8	Normal	true	4.18.2303.8	true	
PC0993	FVF\an	1/16/2024, 7:15:35.000 AM	true	true	1.1.23110.2	4.18.23110.3	Normal	true	4.18.23110.3	true	

Are We In Control - *Operational & Security Inventory of Clients*

ClientInspector
(clientinspector.net)

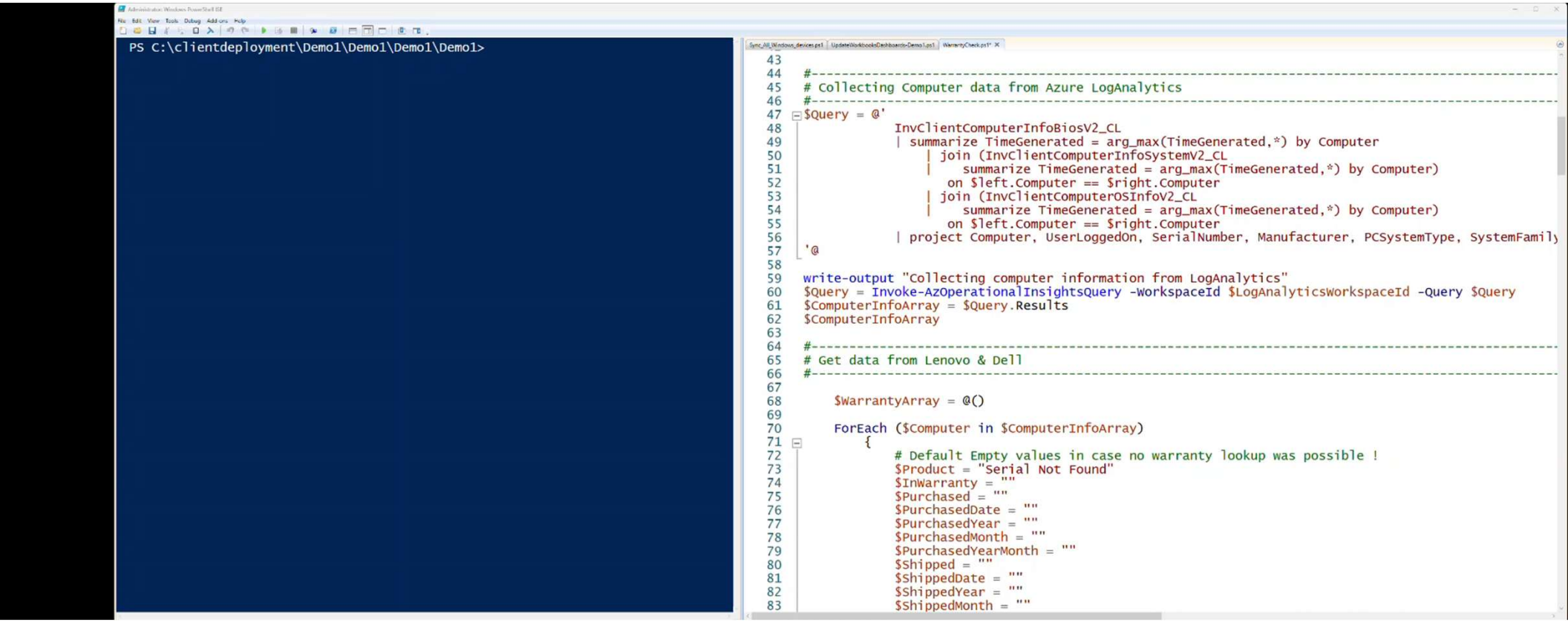
FREE – community edt.

Think of it as inspiration ☺


Administrator: Windows PowerShell
PS C:\Users\mok.2LINKIT\OneDrive - 2linkIT\Desktop\Speaks\Fun with AzLogs\ClientInspectorV2\ClientInspectorV2-DeploymentKit\Demo1>

PC replacement – Warranty Lookup

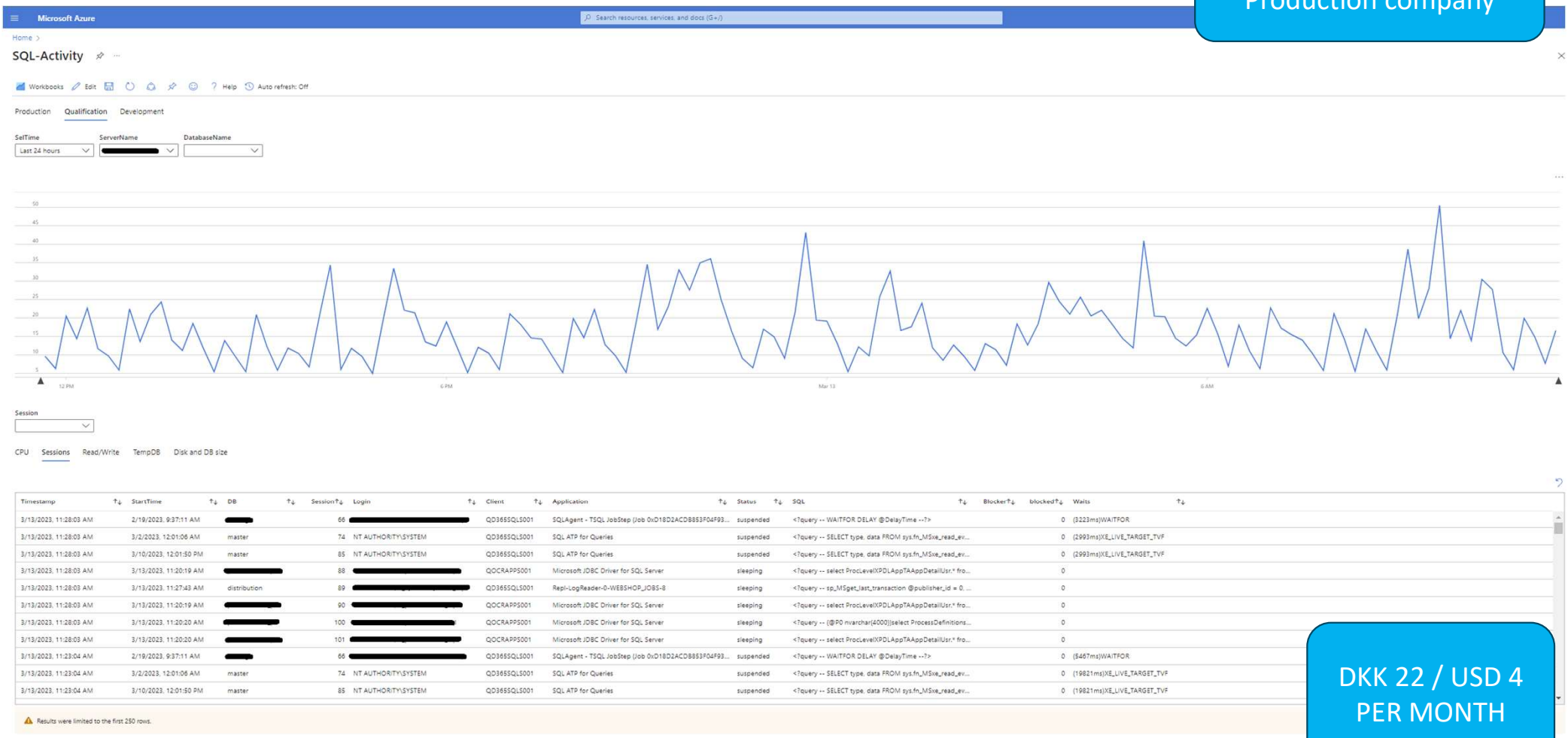
We re-use the inventory data from ClientInspector (serialnumber) to identify computers to replace




```
#-------------------------------------------------------------------------------------------------------------
# Collecting Computer data from Azure LogAnalytics
#-------------------------------------------------------------------------------------------------------------
$Query = @'
            InvClientComputerInfoBiosV2_CL
            | summarize TimeGenerated = arg_max(TimeGenerated,*) by Computer
                | join (InvClientComputerInfoSystemV2_CL
                |    summarize TimeGenerated = arg_max(TimeGenerated,*) by Computer)
                  on $left.Computer == $right.Computer
                | join (InvClientComputerOSInfoV2_CL
                |    summarize TimeGenerated = arg_max(TimeGenerated,*) by Computer)
                  on $left.Computer == $right.Computer
            | project Computer, UserLoggedOn, SerialNumber, Manufacturer, PCSystemType, SystemFamily
'@

write-output "Collecting computer information from LogAnalytics"
$Query = Invoke-AzOperationalInsightsQuery -WorkspaceId $LogAnalyticsWorkspaceId -Query $Query
$ComputerInfoArray = $Query.Results
$ComputerInfoArray

#-------------------------------------------------------------------------------------------------------------
# Get data from Lenovo & Dell
#-------------------------------------------------------------------------------------------------------------

    $WarrantyArray = @()

    ForEach ($Computer in $ComputerInfoArray)
        {
            # Default Empty values in case no warranty lookup was possible !
            $Product = "Serial Not Found"
            $InWarranty = ""
            $Purchased = ""
            $PurchasedDate = ""
            $PurchasedYear = ""
            $PurchasedMonth = ""
            $PurchasedYearMonth = ""
            $Shipped = ""
            $ShippedDate = ""
            $ShippedYear = ""
            $ShippedMonth = ""
```

Monitoring of 1200 SQL databases on 120 SQL-servers

Global Enterprise Danish Production company



DKK 22 / USD 4 PER MONTH

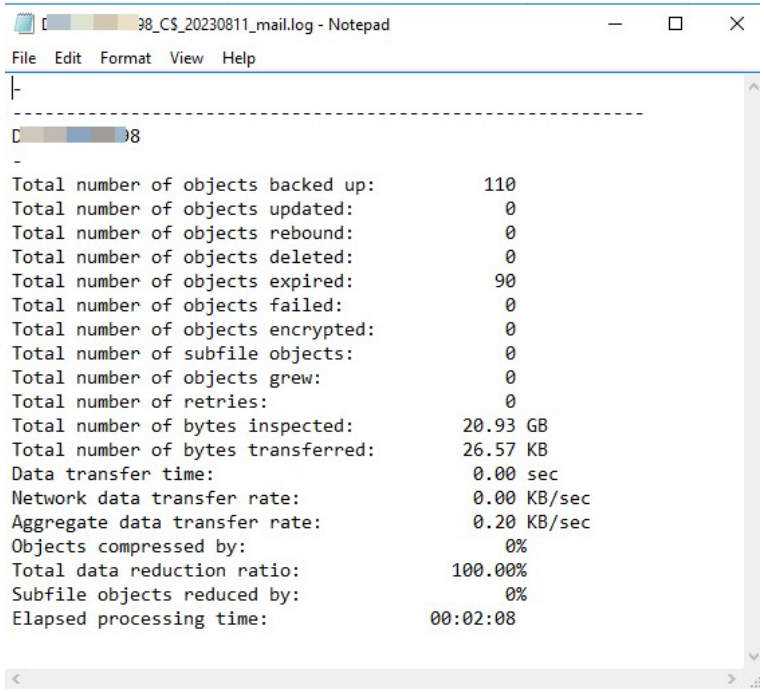
Backup Monitoring of IBM TSM Spectrum

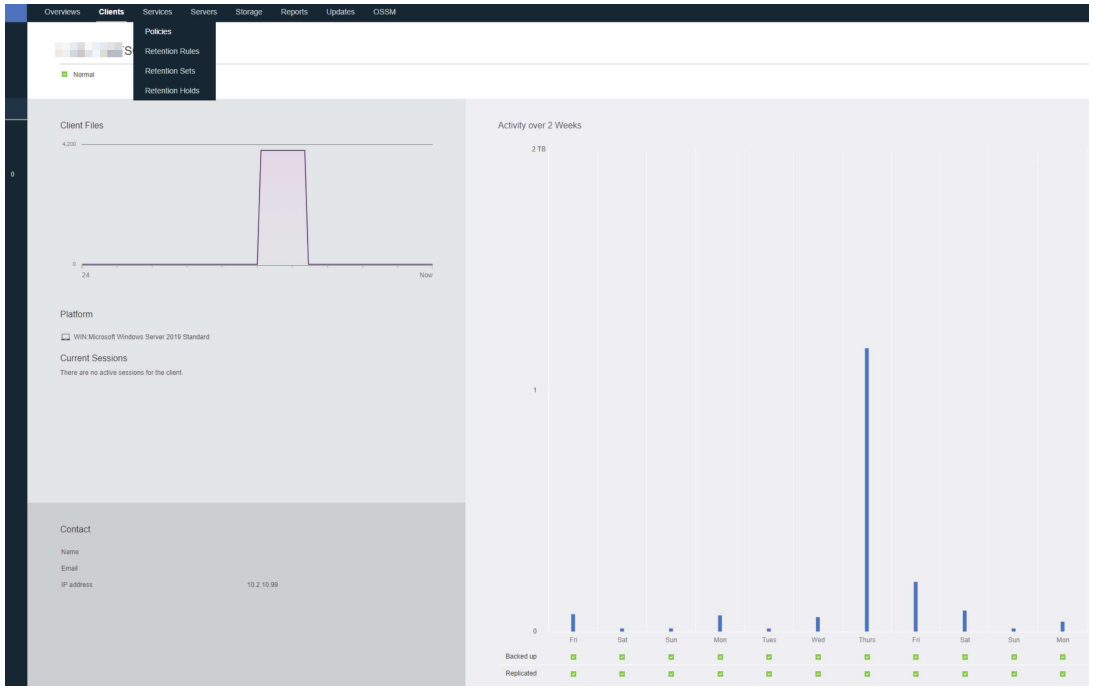
Large Enterprise German Production company

IBM Spectrum Protect aka. Tivoli TSM is used to perform backups especially on file-servers.

The backup agent writes a log file containing various information directly on the machine it runs.

Reporting capabilities of the TSM-Server are limited and do not allow detailed analysis.





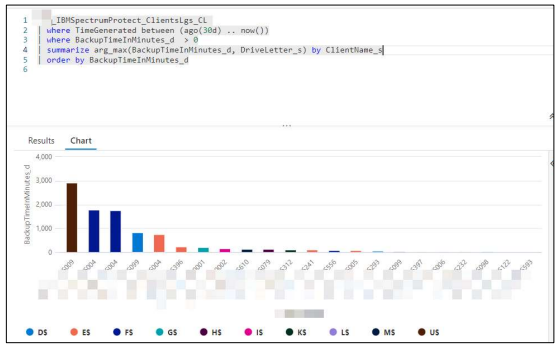
Backup Monitoring of IBM TSM Spectrum (3/3)

Central Log Collection

Large Enterprise German Production company

Building queries for Trending and Azure Monitor for Alerting

After having all information in the Log Analytics Workspace, KQL can be used for detailed analysis.:



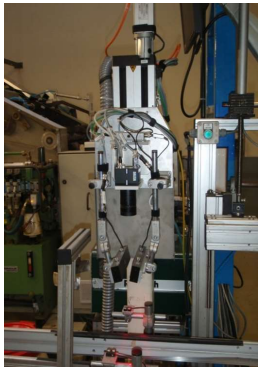
Servers, sorted by time taken for backup

Plotting GBs transferred during backup

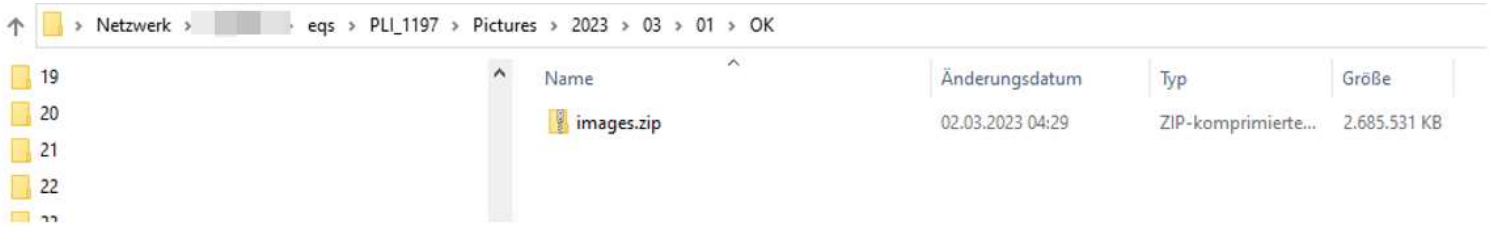
Servers with most failed objects during backup

Quality Inspection for Production

Large Enterprise German Production company



XYZ is an image-processing-system. During one cycle about 10.000 pictures are taken and firstly stored on vendor proprietary hardware. A script runs regularly and copies the pictures to Fileserver on which data is than bundled to control the backup time.



Multiple production lines are equipped with those image-processing-systems. The script is re-used and only passes in different parameters to distinct between the vendor-hardware devices.

Unfortunately, it happens that pictures are missed. – From power outage on the camera, IP connectivity loss to seldom script errors. The script neither offers monitoring capabilities nor the tooling of the image-processing-system does.

Quality Inspection for Production

Building queries for Trending and Azure Monitor for Alerting

Large Enterprise German Production company

Topics we will cover …

- Understanding LogAnalytics V2 vs. V1
- Endpoint log collection using Azure Monitor Agent
- Push & Pull data using Log Ingestion API
  - Challenges & options
  - Automate the transition using AzLogDcrIngestPS
  - Transformations: Data manipulations before & after sending data
- Migration strategies to Log Ingestion API
- How can I get started ?

How many have tried Azure LogAnalytics before ?

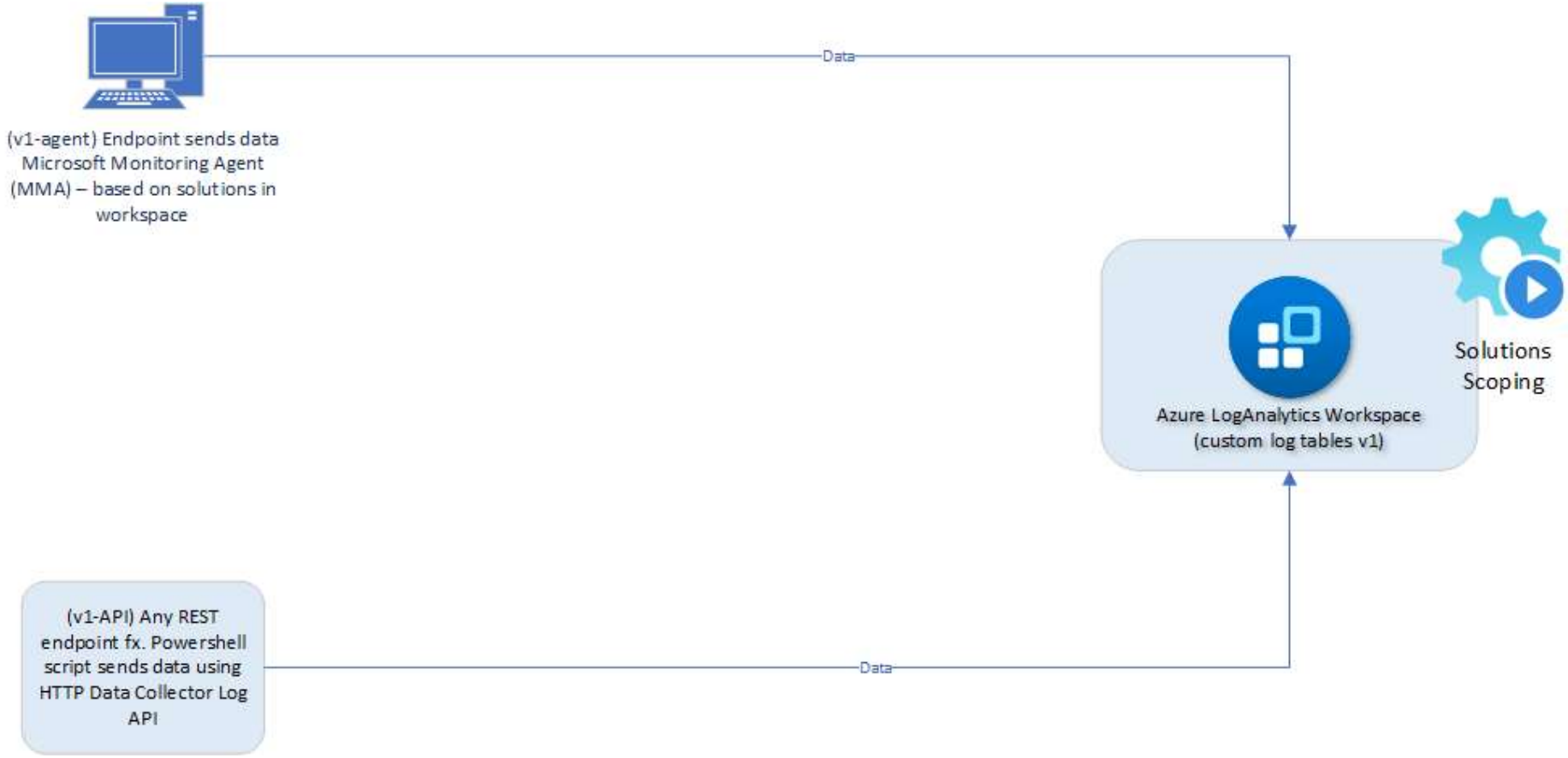
How many have used MMA agent to send data ?
…. AMA agent ?

How many have used HTTP Data Collector API before ?

How many have tried to play with Log Ingestion API ?

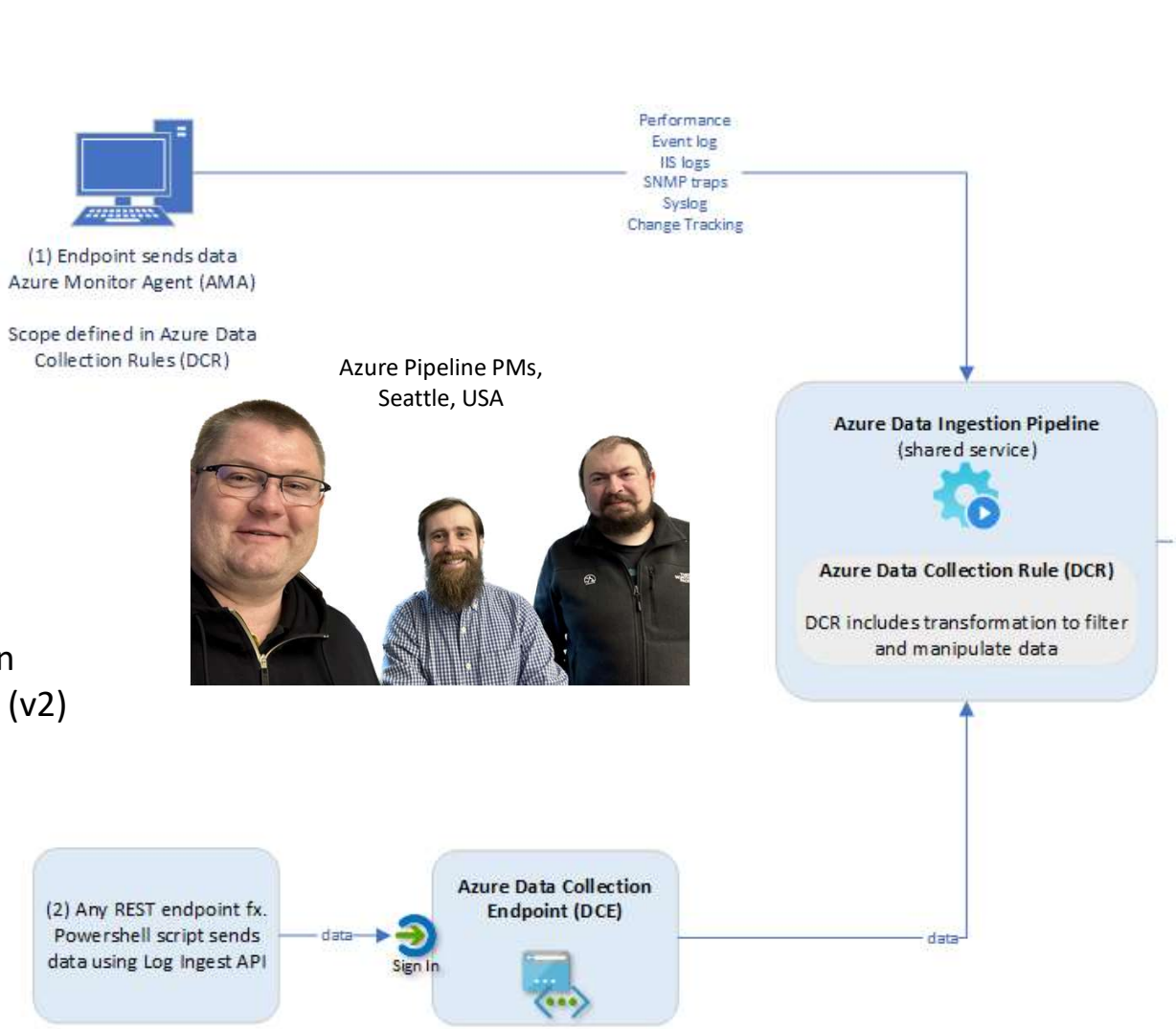
“V1” (legacy)

- LA Solutions
- MMA - Limited collection-capabilities – built- into agent
- Azure Monitor HTTP Data Collector API



“V2”

- Azure Pipeline
- DCR & DCE
- Transformation
- Custom tables (v2)



Azure Pipeline PMs,
Seattle, USA

Azure DCR PMs &
Dev Lead, Seattle, USA

Azure LogAnalytics PMs, Israel

Endpoint-based collection using AMA

Performance metrics & telemetry

Performance Counters (SQL)

Eventlogs

IIS logs

Standard/Platform metrics/Telemetry (AzPaaS)

Custom Logs (Ingestion API)

Windows Fw Logs

Activity logs

Custom metrics/telemetry (custom AzApp)

Platform logs (diagnostics)

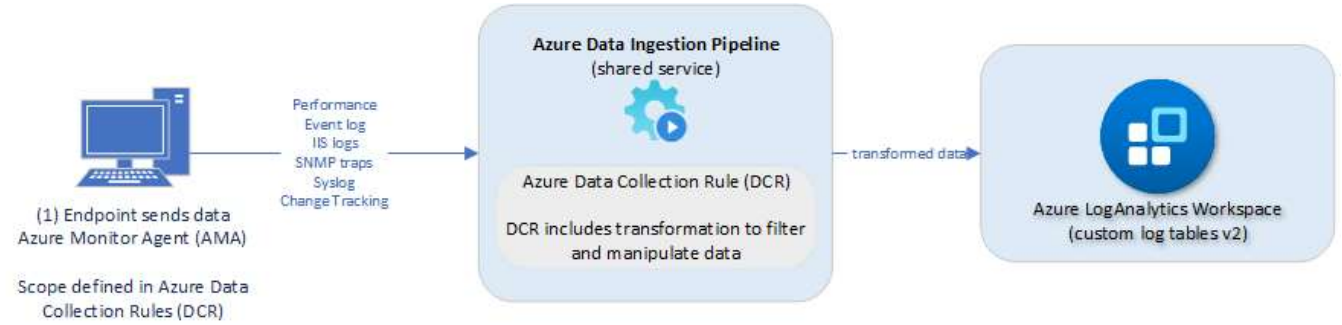
Standard Logs (Ingestion API)

Text logs

Change Tracking

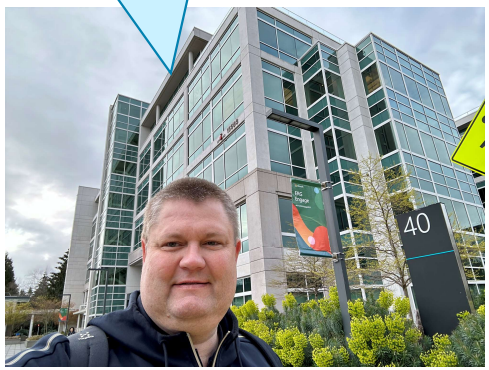
SNMP Traps

Meet the Azure Monitor PMs (Seattle, USA)



- **Transformations -> remove / add / merge data** before being sent into LogAnalytics
  - remove "noice" data/cost optimization, GDPR/compliance
- better data quality of **array data (dynamic array)**
- security is based on **Entra ID RBAC**
- **naming of data columns** are prettier
- Future - couldn't it be REALLY cool, if we could ....☺
  - support to **send to other destinations**
  - **Aggregation of data**
  - **Monitoring of data**

MS HQ - more really cool stuff is 'in the oven being backed' – coming soon ☺



How to get started with Azure Logging v2 on Azure Monitor Agent ??

VMInsight - Linux - Performance_ServiceMap | Deploy DCR-rule to Azure

Deploy to Azure

Blog:
https://mortenknudsen.net

Topic	Link
Understanding Azure logging capabilities in depth	https://mortenknudsen.net/?p=1433
How to do data transformation using Workspace transformation for legacy upload methods	https://mortenknudsen.net/?p=1436
Understanding the fundamentals of log-collection with Azure Monitor Agent & Azure Data Collection Rules	https://mortenknudsen.net/?p=1438
Understanding Azure Data Collection Endpoint	https://mortenknudsen.net/?p=1442
Collecting Security events using Azure Monitor Agent	https://mortenknudsen.net/?p=1444
Collecting System & Application events using Azure Monitor Agent	https://mortenknudsen.net/?p=1446
Collecting Performance data using Azure Monitor Agent, VMInsights and ServiceMap	https://mortenknudsen.net/?p=1449
Collecting IIS logs using Azure Monitor Agent	https://mortenknudsen.net/?p=1451
Collecting text logs using Azure Monitor Agent	https://mortenknudsen.net/?p=1453
Collecting Syslogs using Azure Monitor Agent	https://mortenknudsen.net/?p=1455
Collecting CEF Syslogs using Azure Monitor Agent	https://mortenknudsen.net/?p=1457
Tutorial – How to make data transformations using Data Collection Rules?	https://mortenknudsen.net/?p=1440

GiveAway Questions 1 & 2

When will Microsoft retire Microsoft Monitoring Agent (MMA) ?

Month + Year

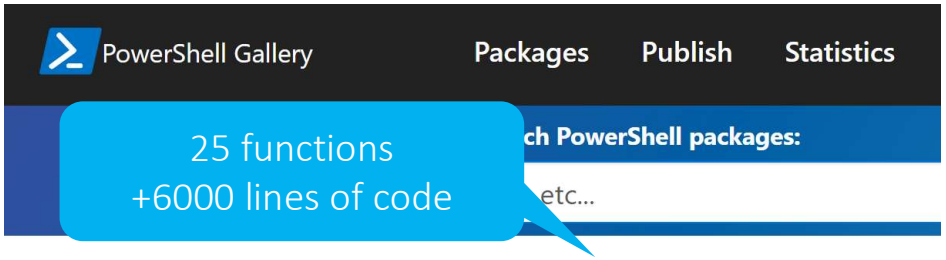
When will Microsoft retire HTTP Data Collector api ?

Come up to me afterwards

Challenges with v2 technologies

Compared with V1 (HTTP Data Collector API)

- Requirements / Dependencies
  - **Creation of DCR + tables** before sending data
  - **Data Collection EndPoint** must exist
  - **Schema** for data must be defined in both DCR and v2 custom table
  - **Azure seconds** - timing / delays – dependencies
- Architecture
  - **Naming conventions** & limitations / Prohibited names
  - Handle new properties in source object (**merge, overwrite**)
  - **Property changes** -> True (string) → True (Boolean) – "internal server 500"
  - **Upload changes** (32 mb -> 1 mb) per JSON (batches, calculations)
  - **DCR limitations** with large schema
  - **Platform** – no-internet access, TLS incompatibility
- **Data manipulations** of source data (filtering, remove)



938,147
Downloads

683,974
Downloads of 1.4.4
View full stats

8/25/2023
Last Published

AzLogDcrIngestPS 1.4.4

This module includes cmdlets to automate all the
LogAnalytics tables and Azure Data Collection Rules
custom logs using Log ingestion API and Azure Data

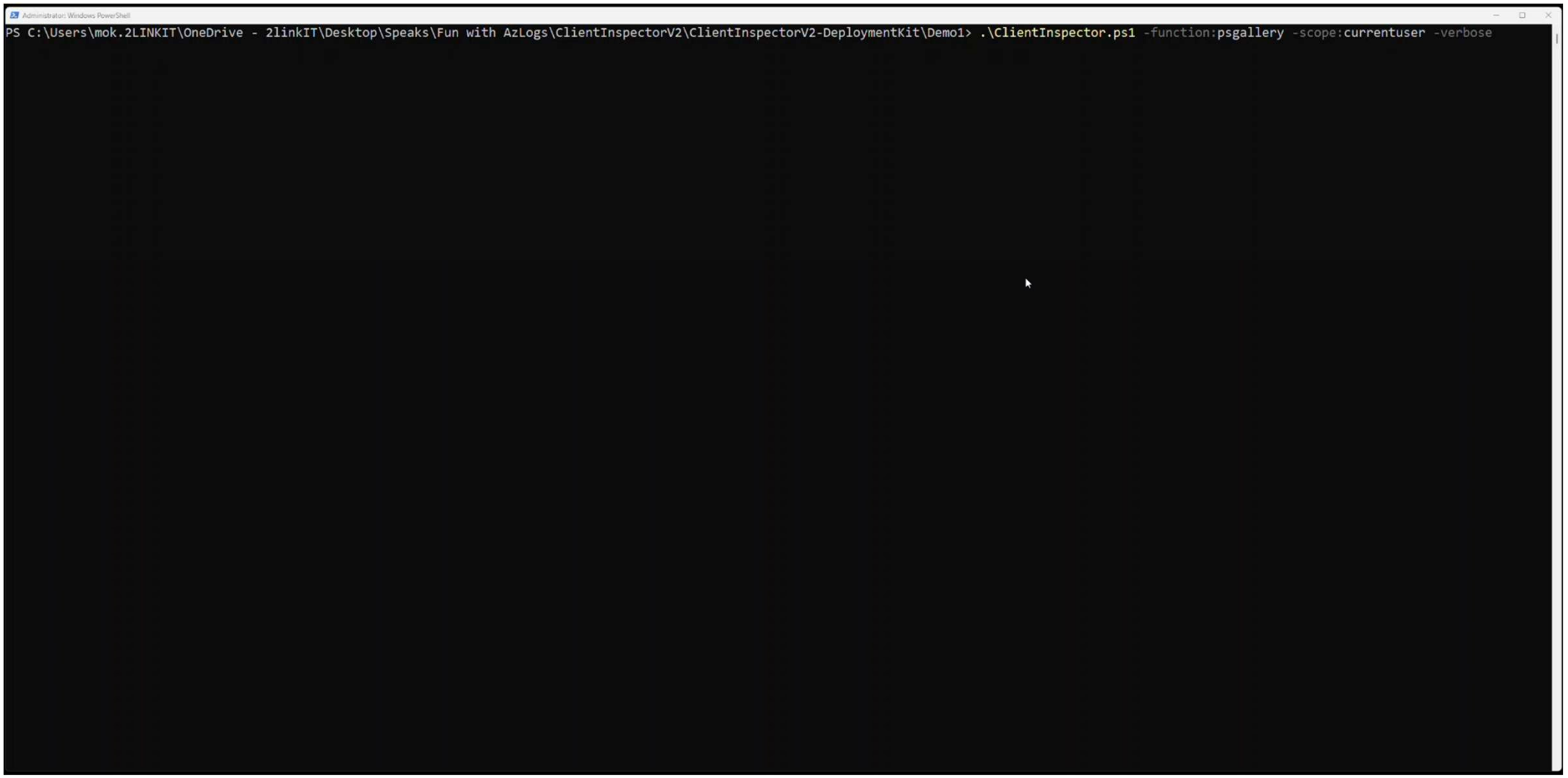
Functions can be used for:
(1) manipulation of source object (fix invalid struc
+ Show more

Minimum PowerShell version
5.1

Installation Options

Install Module | Install PSResource | A
Manual Download

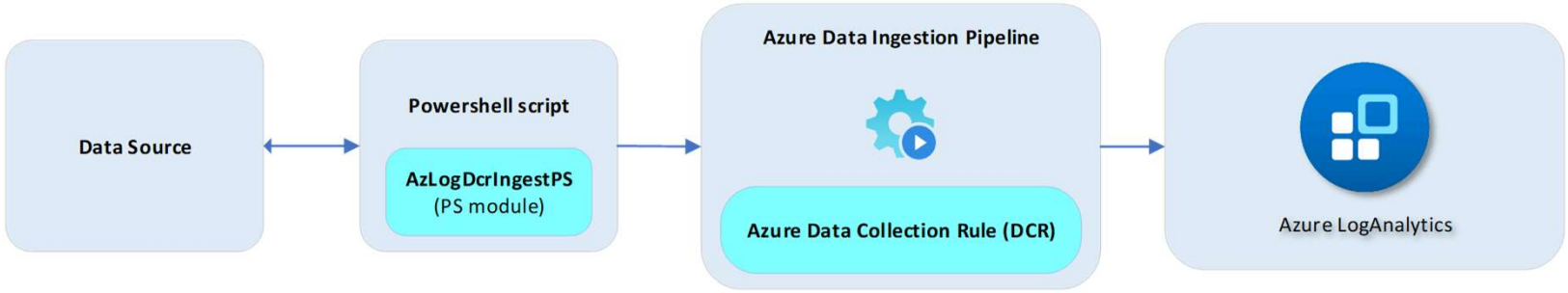
"Wrapper of standard features"


Administrator: Windows PowerShell
PS C:\Users\mok.2LINKIT\OneDrive - 2linkIT\Desktop\Speaks\Fun with AzLogs\ClientInspectorV2\ClientInspectorV2-DeploymentKit\Demo1> .\ClientInspector.ps1 -function:psgallery -scope:currentuser -verbose

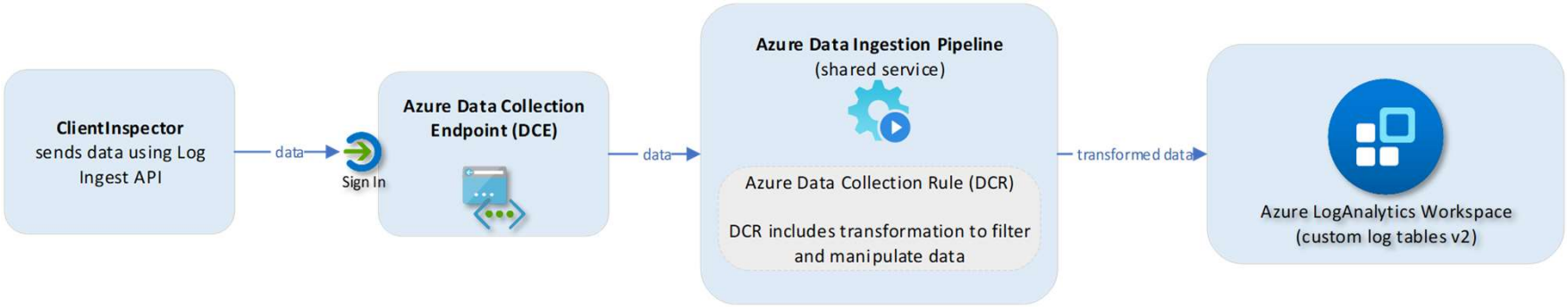
Use-Case | AzLogDcrIngestPS

"AnyConnector"

- **Pull** - Retrieve data from any 3rd party source using PS script (intermediate)



- **Push** - Collect data from endpoint - send data from endpoint (ClientInspector)



Official Microsoft documentation reference

Migrate from the HTTP Data Collector API to the Log Ingestion API to send data to Azure Monitor Logs

Article • 06/19/2023 • 5 contributors

Feedback

In this article

- Advantages of the Log Ingestion API
- Prerequisites
- Create new resources required for the Log ingestion API
- Migrate existing custom tables or create new tables

Show 3 more

The Azure Monitor Log Ingestion API provides more processing power and greater flexibility in ingesting logs and managing tables than the legacy HTTP Data Collector API. This article describes the differences between the Data Collector API and the Log Ingestion API and provides guidance and best practices for migrating to the new Log Ingestion API.

ⓘ Note

As a Microsoft MVP, **Morten Waltorp Knudsen** contributed to and provided material feedback for this article. For an example of how you can automate the setup and ongoing use of the Log Ingestion API, see Morten's publicly available **AzLogDcrIngestPS PowerShell module**.

https://learn.microsoft.com/en-us/azure/azure-monitor/logs/custom-logs-migrate

https://learn.microsoft.com/en-us/azure/azure-monitor/logs/set-up-logs-ingestion-api-prerequisites

Learn / Azure / Azure Monitor /

Set up resources required to send data to Azure Monitor Logs using the Logs Ingestion API

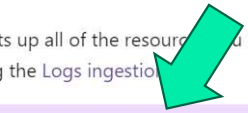
Article • 06/22/2023 • 2 contributors

Feedback

In this article

- Create resources and permissions
- PowerShell script
- Next steps

This article provides a PowerShell script that sets up all of the resour[illegible] need before you can send data to Azure Monitor Logs using the Logs ingestio[illegible]



ⓘ Note

As a Microsoft MVP, **Morten Waltorp Knudsen** contributed to and provided material feedback for this article. For an example of how you can automate the setup and ongoing use of the Log Ingestion API, see Morten's **AzLogDcrIngestPS PowerShell module**.

Create resources and permissions

The script creates these resources, if they don't already exist:

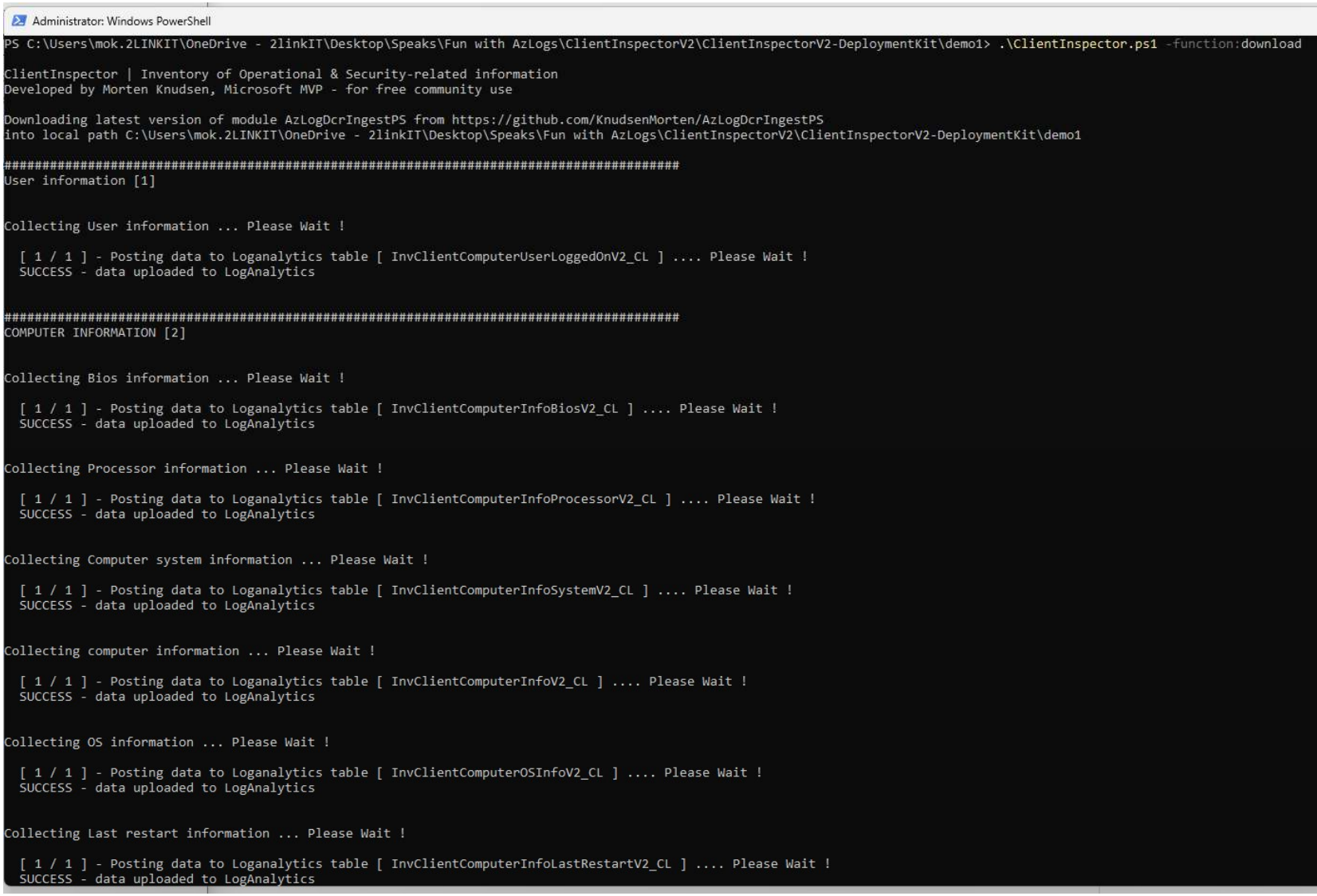
- A Log Analytics workspace and a resource group for the Log Analytics workspace.

  You probably already have a Log Analytics workspace, in which case, provide the

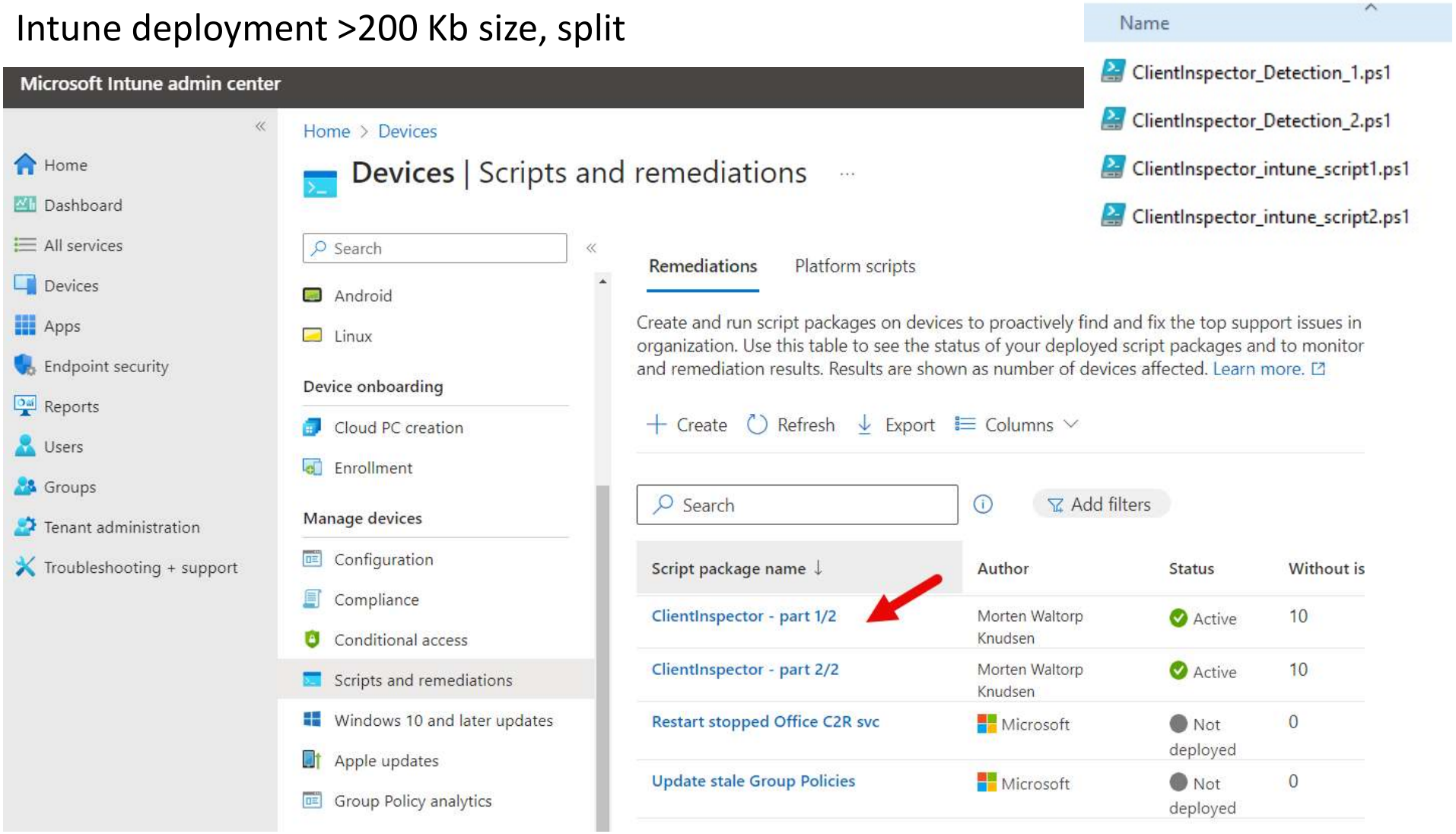
Run with

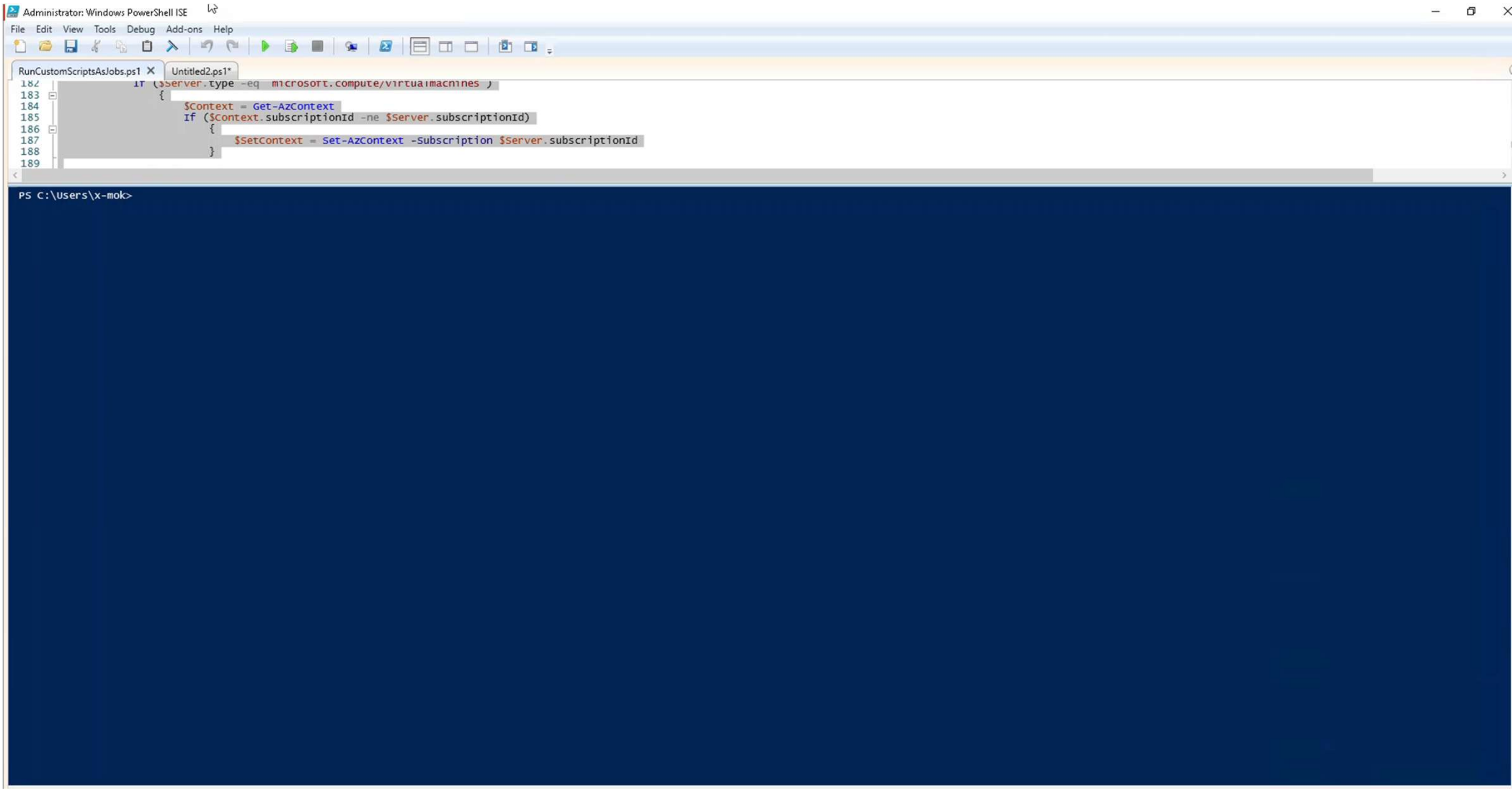
ConfigMgr
Intune
any deployment-tool

Servers:
RemotePS
CustomScript Extension



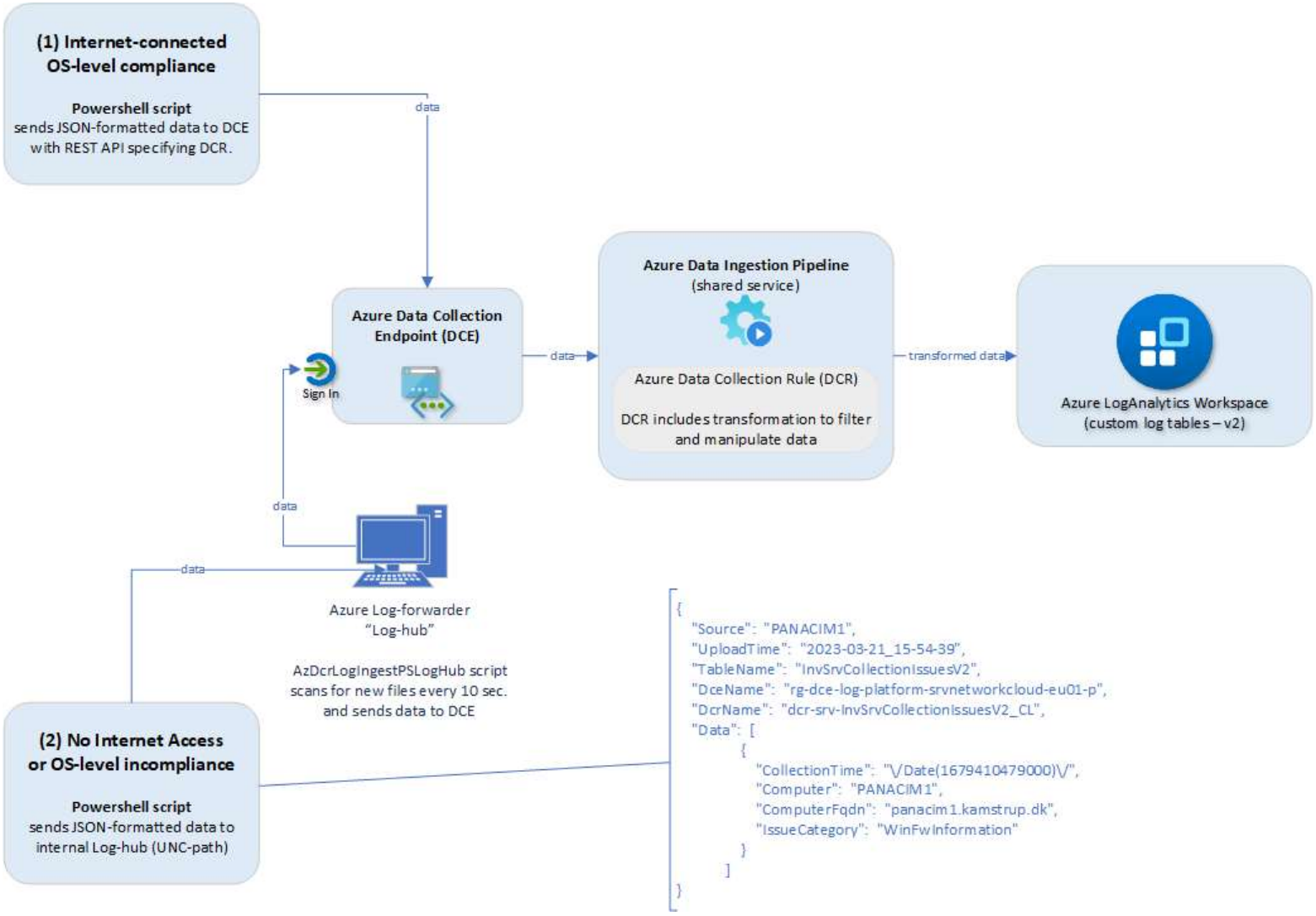
Intune deployment >200 Kb size, split

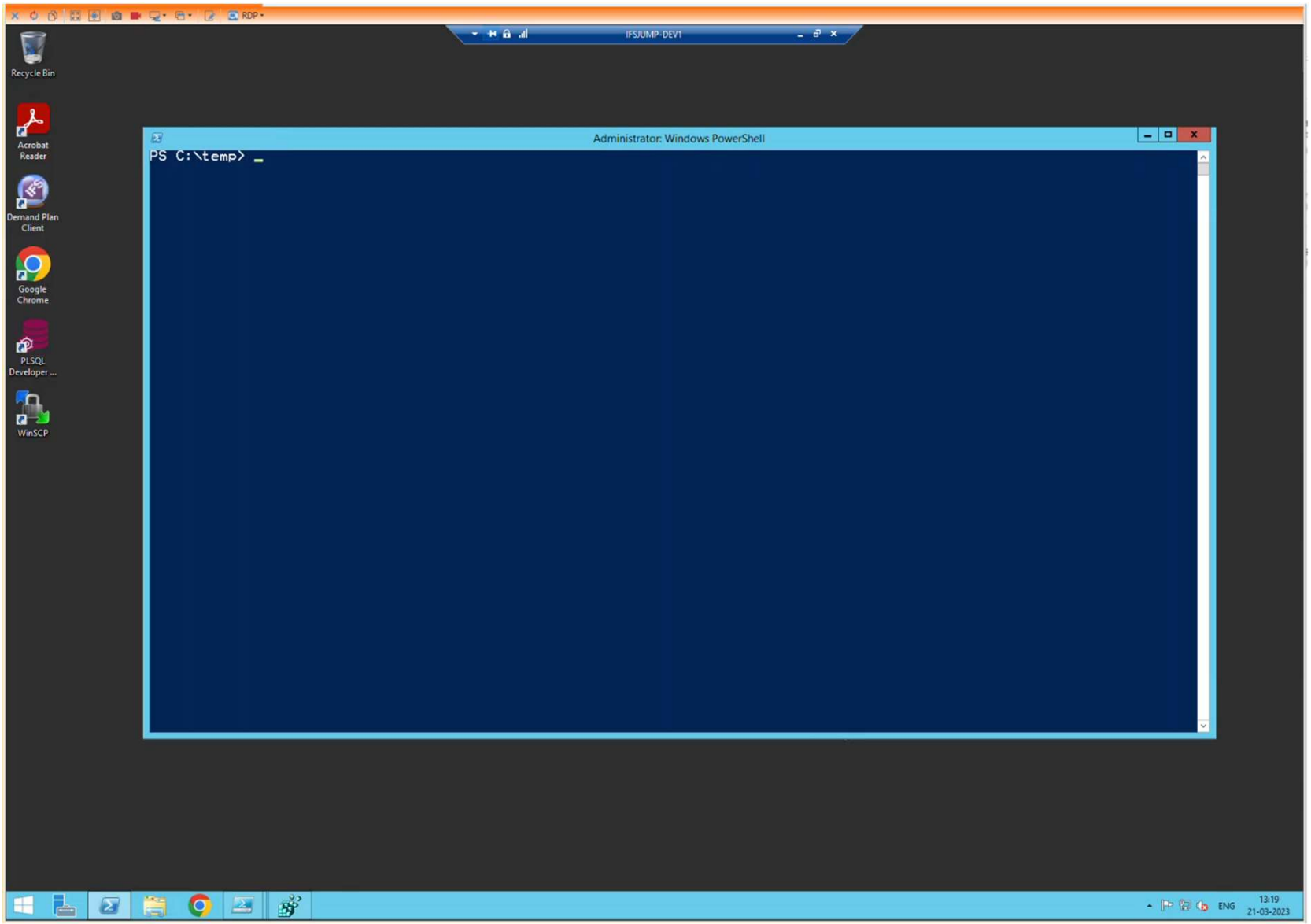



Administrator: Windows PowerShell ISE
File Edit View Tools Debug Add-ons Help
RunCustomScriptsAsJobs.ps1
Untitled2.ps1*
182 if ($server.type -eq 'microsoft.compute/virtualmachines')
183 {
184 $Context = Get-AzContext
185 If ($Context.subscriptionId -ne $Server.subscriptionId)
186 {
187 $SetContext = Set-AzContext -Subscription $Server.subscriptionId
188 }
189
PS C:\Users\x-mok>

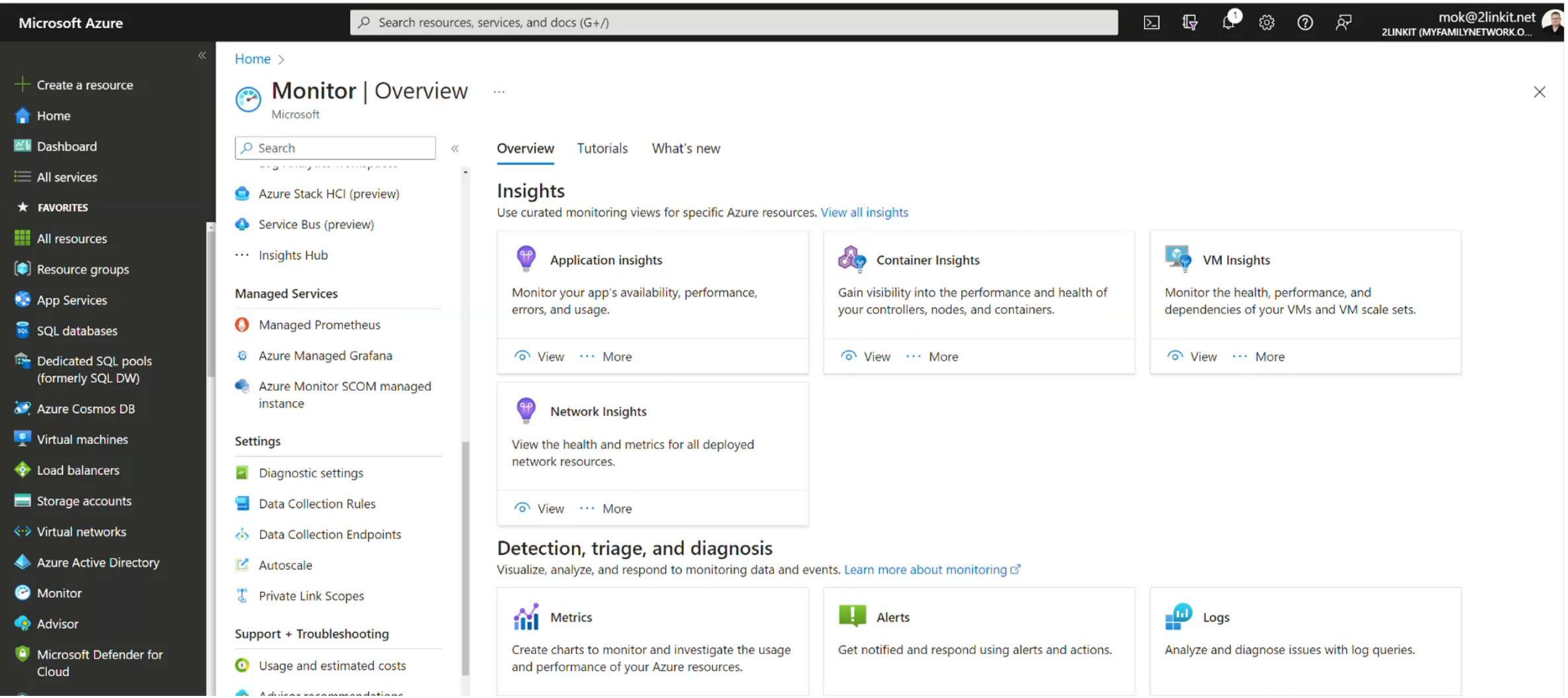
LogHub

Available on Github
(Community - Free)




RDP
IFSJUMP-DEV1
Recycle Bin
Acrobat Reader
Demand Plan Client
Google Chrome
PLSQL Developer ...
WinSCP
Administrator: Windows PowerShell
PS C:\temp> _
ENG
13:19
21-03-2023

Schema – getting “under the hood”



Azure Data Ingestion Pipeline

Data Collection Rule (DCR)

dataCollectionEndpointId
(variable, input, from where)

streamDeclarations
(variable, schema)
column name, description, type

destinations
(variable Loganalytics workspace)

dataFlows

streams = array of streamDeclarations

destinations = array of destinations

outputStream = variable for custom log table in Loganalytics

transformKql = transformations


```
{
  "properties": {
    "immutableId": "dcr-3b8b1ae5e4b24c89b314e8b6e888608f",
    "dataCollectionEndpointId": "/subscriptions/fce4f282-fcc6-43fb-94d8-bf1701b862c3/resourceGroups/rg-dce-log-
management-client-demo1-t/providers/Microsoft.Insights/dataCollectionEndpoints/dce-log-management-client-demo1-t",
    "streamDeclarations": {
      "Custom-InvClientComputerInfoLastRestartV2_CL": {
        "columns": [
          {
            "name": "Computer",
            "type": "string"
          },
          {
            "name": "DaysSinceLastRestart",
            "type": "int"
          },
          {
            "name": "LastRestart",
            "type": "datetime"
          },
          {
            "name": "UserLoggedOn",
            "type": "string"
          }
        ]
      }
    },
    "destinations": {
      "logAnalytics": [
        {
          "workspaceResourceId": "/subscriptions/fce4f282-fcc6-43fb-94d8-bf1701b862c3/resourceGroups/rg-
logworkspaces-client-demo1-t/providers/Microsoft.OperationalInsights/workspaces/log-management-client-demo1-t",
          "workspaceId": "a8c786cb-e8c1-40b8-9eb9-f0ceef8157fe",
          "name": "log-management-client-demo1-t"
        }
      ]
    },
    "dataFlows": [
      {
        "streams": [
          "Custom-InvClientComputerInfoLastRestartV2_CL"
        ],
        "destinations": [
          "log-management-client-demo1-t"
        ],
        "transformKql": "source | extend TimeGenerated = now()",
        "outputStream": "Custom-InvClientComputerInfoLastRestartV2_CL"
      }
    ],
    "provisioningState": "Succeeded"
  }
```

Azure Monitor
(unique resource)

Data Collection Endpoint (DCE)

Properties:

immutableId

configurationAccess
(url for configuration)

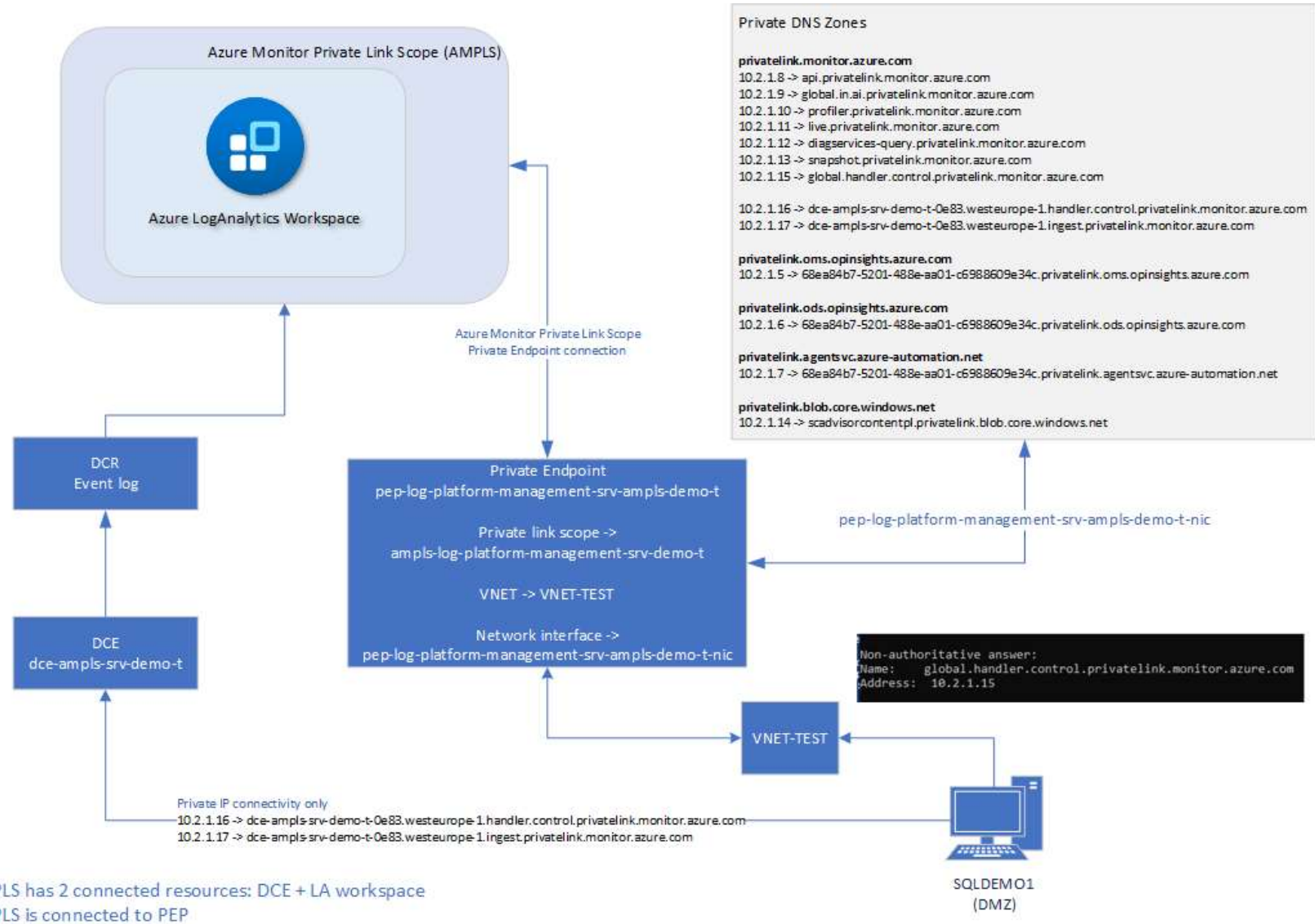
logsIngestion
(url for data upload)

metricsIngestion
(url for monitoring)

networkAcls
publicNetworkAccess
= Enabled/Disabled


```
"properties": {
    "description": "DCE for LogIngest to LogAnalytics log-management-client-demo1-t",
    "immutableId": "dce-df70b659df0a47b99f09860ded66ee06",
    "configurationAccess": {
      "endpoint": "https://dce-log-management-client-demo1-t-2r31.westeurope-1.handler.control.monitor.azure.com"
    },
    "logsIngestion": {
      "endpoint": "https://dce-log-management-client-demo1-t-2r31.westeurope-1.ingest.monitor.azure.com"
    },
    "metricsIngestion": {
      "endpoint": "https://dce-log-management-client-demo1-t-2r31.westeurope-1.metrics.ingest.monitor.azure.com"
    },
    "networkAcls": {
      "publicNetworkAccess": "Enabled"
    },
    "provisioningState": "Succeeded"
  },
  "location": "westeurope",
  "id": "/subscriptions/fce4f282-fcc6-43fb-94d8-bf1701b862c3/resourceGroups/rg-dce-log-management-client-demo1-t/
providers/Microsoft.I
  "name": "dce-log-m
  "type": "Microsoft.I
  "etag": "\"b309c03
  "systemData": {
    "createdBy": "mok@2linkit.net",
    "createdByType": "User",
    "createdAt": "2023-03-19T10:57:58.6132975Z",
    "lastModifiedBy": "mok@2linkit.net",
    "lastModifiedByType": "User",
    "lastModifiedAt": "2023-03-19T10:57:58.6132975Z"
  }
}
```


Possible to optimize networking with
Azure Monitor Private Link Scope (AMPLS)



AMPLS has 2 connected resources: DCE + LA workspace
AMPLS is connected to PEP
AMPLS - Private Link Scope access modes – ingestion = open, query access = open

GiveAway Question #3

Who can mention the 2 main components you use when ingesting data through log ingestion api – besides Azure LogAnalytics custom table (v2) and Azure Pipeline ?

Hints:

They both have a 3-letter acronym - both starting with D

Sending custom data into LA with AzLogDcrIngestPS

Data In (“AnyConnector”)

WMI, CIM, REST API, DBs, CSV, JSON, XML, etc.

Data Manipulate

Strip data

Convert data

Add relevant data (UserLoggedOn, Computer, CollectionTime)

Table/DCR/Schema/ Transformation management

SchemaMode = Overwrite

SchemaMode = Merge

Create/update DCR based on table schema

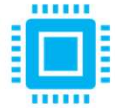
Validation

Auto-fix

Data Upload

Auto-size

BatchAmount



Support functions

Security (token REST)

List DCRs/DCEs

Maintenance (delete)

Transformations (GET, UPDATE, RESET)

Structure 1/5: Variables

naming - where to send the data

```
$TableName                                  = "v2CustomTable" + $DemoNumber

$TenantId                                   = "f0fa27a0-8e7c-4f63-9a77-xxxxxxxxxxxx"
$LogIngestAppId                             = "8cd71693-be69-421c-axxxxxxxxxxxxxx"
$LogIngestAppSecret                         = "Bxxxxxxxxxxxxxxxx"

$DceName                                    = "dce-log-loganalyticsv2-migration-demo"
$AzDcrResourceGroup                         = "rg-dcr-log-loganalyticsv2-migration-demo"

$LogAnalyticsWorkspaceResourceId            = "…/providers/Microsoft.OperationalInsights/workspaces/log-loganalyticsv2-demo"

$AzDcrSetLogIngestApiAppPermissionsDcrLevel = $false
$AzDcrLogIngestServicePrincipalObjectId     = "2f259c5a-503f-4c36-909d-xxxxxxxx"

$AzLogDcrTableCreateFromReferenceMachine    = @()
$AzLogDcrTableCreateFromAnyMachine          = $true
$DcrName                                    = "dcr-demo-" + $TableName + "_CL"

$Verbose                                    = $true
```

```
#--------------------------------------------------------------------------
# Variables
#--------------------------------------------------------------------------

    $TableName  = 'InvClientDefenderAvV2'   # must not contain _CL
    $DcrName    = "dcr-" + $AzDcrPrefix + "-" + $TableName + "_CL"
```

Structure 2/5: Data Collection

```
#-------------------------------------------------------------------------
# Collecting data (in)
#-------------------------------------------------------------------------

    Write-Output ""
    Write-Output "Collecting Bios information ... Please Wait !"

    $DataVariable = Get-CimInstance -ClassName Win32_BIOS

"
```

```
#-------------------------------------------------------------------------------------------
# Collecting data (in)
#-------------------------------------------------------------------------------------------

    Write-Output ""
    Write-Output "Collecting Microsoft Defender Antivirus information ... Please Wait !"

    $MPComputerStatus = Get-MpComputerStatus
    $MPPreference = Get-MpPreference
```

```
AMEngineVersion                 : 1.1.23050.3
AMProductVersion                : 4.18.23050.3
AMRunningMode                   : Normal
AMServiceEnabled                : True
AMServiceVersion                : 4.18.23050.3
AntispywareEnabled              : True
AntispywareSignatureAge         : 0
AntispywareSignatureLastUpdated : 11-06-2023 06:18:21
AntispywareSignatureVersion     : 1.391.1111.0
AntivirusEnabled                : True
AntivirusSignatureAge           : 0
AntivirusSignatureLastUpdated   : 11-06-2023 06:18:21
AntivirusSignatureVersion       : 1.391.1111.0
BehaviorMonitorEnabled          : True
CimClass                        :
ROOT/Microsoft/Windows/Defender:MSFT_MpComputerStatus
CimInstanceProperties           : {AMEngineVersion,
AMProductVersion, AMRunningMode, AMServiceEnabled...}
CimSystemProperties             :
Microsoft.Management.Infrastructure.CimSystemProperties
CollectionTime                  : 11-06-2023 13:02:42
Computer                        : STRV-MOK-DT-02
ComputerFqdn                    : STRV-MOK-DT-02.WORKGROUP
ComputerID                      : A39E177B-9D86-A3AE-93EC-
A9FA04AC9191
ComputerState                   : 0
DefenderSignaturesOutOfDate     : False
DeviceControlDefaultEnforcement : Default Allow
DeviceControlPoliciesLastUpdated : 27-03-2023 00:34:19
DeviceControlState              : Disabled
FullScanAge                     : 83
FullScanEndTime                 : 20-03-2023 01:35:18
FullScanOverdue                 : False
FullScanRequired                : False
FullScanSignatureVersion        : 1.385.490.0
FullScanStartTime               : 19-03-2023 22:31:53
IoavProtectionEnabled           : True
IsTamperProtected               : True
IsVirtualMachine                : False
LastFullScanSource              : 2
LastQuickScanSource             : 2
NISEnabled                      : True
NISEngineVersion                : 1.1.23050.3
NISSignatureAge                 : 0
NISSignatureLastUpdated         : 11-06-2023 06:18:21
NISSignatureVersion             : 1.391.1111.0
OnAccessProtectionEnabled       : True
ProductStatus                   : 524288
PSComputerName                  :
QuickScanAge                    : 0
QuickScanEndTime                : 11-06-2023 07:16:51
QuickScanOverdue                : False
QuickScanSignatureVersion       : 1.391.1075.0
QuickScanStartTime              : 11-06-2023 03:52:21
RealTimeProtectionEnabled       : True
RealTimeScanDirection           : 0
RebootRequired                  : False
SmartAppControlExpiration       :
SmartAppControlState            : Off
TamperProtectionSource          : Intune
TDTMode                         : N/A
TDTSiloType                     : N/A
TDTStatus                       : N/A
TDTTelemetry                    : N/A
TroubleShootingDailyMaxQuota    : 480
TroubleShootingDailyQuotaLeft   : 480
TroubleShootingEndTime          : INFINITE
TroubleShootingExpirationLeft   : INFINITE
TroubleShootingMode             : Disabled
TroubleShootingModeSource       : Service
TroubleShootingQuotaResetTime   : N/A
TroubleShootingStartTime        : N/A
UserLoggedOn                    : 2LINKIT\mok
```

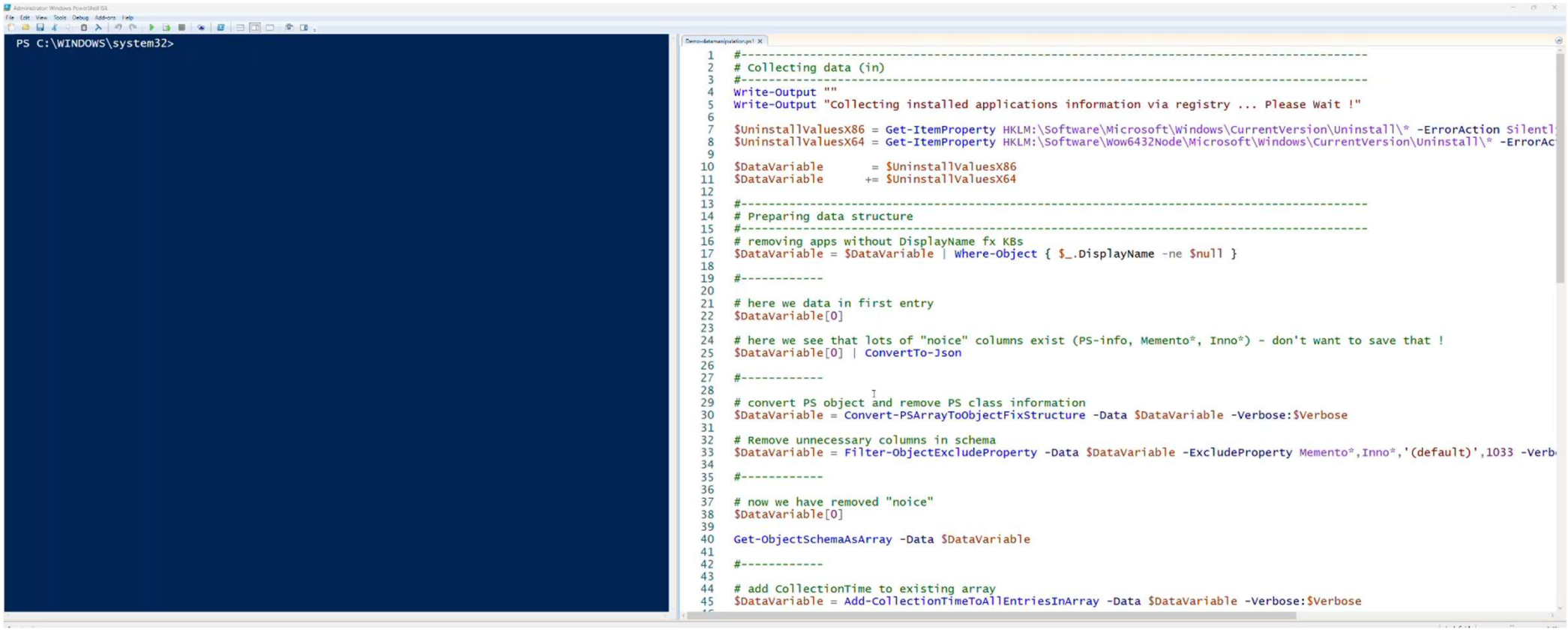
Structure 3/5: Data Manipulation

ensure data is in correct format and any "noise" was removed and relevant information has been added

```powershell
# removing apps without DisplayName fx KBs
$DataVariable = $DataVariable | Where-Object { $_.DisplayName -ne $null }

# convert PS object and remove PS class information
$DataVariable = Convert-PSArrayToObjectFixStructure -Data $DataVariable -Verbose:$Verbose

# add CollectionTime to existing array
$DataVariable = Add-CollectionTimeToAllEntriesInArray -Data $DataVariable -Verbose:$Verbose

# add Computer, ComputerFqdn & UserLoggedOn info to existing array
$DataVariable = Add-ColumnDataToAllEntriesInArray -Data $DataVariable -Column1Name Computer -Column1Data $Env:ComputerName -Column2Name ComputerFqdn -Column2

# Get insight about the schema structure of an object BEFORE changes. Command is only needed to verify columns in schema
# $SchemaBefore = Get-ObjectSchemaAsArray -Data $DataVariable

# Remove unnecessary columns in schema
$DataVariable = Filter-ObjectExcludeProperty -Data $DataVariable -ExcludeProperty Memento*,Inno*,'(default)',1033 -Verbose:$Verbose

# Validating/fixing schema data structure of source data
$DataVariable = ValidateFix-AzLogAnalyticsTableSchemaColumnNames -Data $DataVariable -Verbose:$Verbose

# Aligning data structure with schema (requirement for DCR)
$DataVariable = Build-DataArrayToAlignWithSchema -Data $DataVariable -Verbose:$Verbose
```

```
VERBOSE: Getting Data Collection Rules from Azure Resource Graph .... Please Wait !
VERBOSE: POST with -1-byte payload
VERBOSE: received 1468-byte response of content type application/json; charset=utf-8
VERBOSE: POST with -1-byte payload
VERBOSE: received 11713-byte response of content type application/json; charset=utf-8
VERBOSE:   Adding CollectionTime to all entries in array .... please wait !
VERBOSE:   Adding columns to all entries in array .... please wait !
VERBOSE:   Validating schema structure of source data ... Please Wait !
VERBOSE:   SUCCESS - No issues found in schema structure
VERBOSE:   Aligning source object structure with schema ... Please Wait !
```

Data Manipulation demo



Structure 4/5: Table & DCR management

```
CheckCreateUpdate-TableDcr-Structure -AzLogWorkspaceResourceId $LogAnalyticsWorkspaceResourceId -SchemaMode Merge `
                                     -AzAppId $LogIngestAppId -AzAppSecret $LogIngestAppSecret -TenantId $TenantId -Verbose:$Verbose `
                                     -DceName $DceName -DcrName $DcrName -DcrResourceGroup $AzDcrResourceGroup -TableName $TableName `
                                     -LogIngestServicePricipleObjectId $AzDcrLogIngestServicePrincipalObjectId `
                                     -Data $DataVariable -AzDcrSetLogIngestApiAppPermissionsDcrLevel $AzDcrSetLogIngestApiAppPermissionsDcrLevel `
                                     -AzLogDcrTableCreateFromAnyMachine $AzLogDcrTableCreateFromAnyMachine `
                                     -AzLogDcrTableCreateFromReferenceMachine $AzLogDcrTableCreateFromReferenceMachine
```

```
VERBOSE:   Checking LogAnalytics table and Data Collection Rule configuration .... Please Wait !
VERBOSE: POST with -1-byte payload
VERBOSE: received 1468-byte response of content type application/json; charset=utf-8
VERBOSE: GET with 0-byte payload
VERBOSE:   LogAnalytics table wasn't found !
VERBOSE:   DCR was not found [ dcr-demo-V2CustomTable5778_CL ]
VERBOSE: GET with 0-byte payload
VERBOSE: POST with -1-byte payload
VERBOSE: received 1468-byte response of content type application/json; charset=utf-8
VERBOSE: GET with 0-byte payload
VERBOSE:
VERBOSE: Trying to update existing LogAnalytics table schema for table [ V2CustomTable5778_CL ] in
VERBOSE: /subscriptions/fce4f282-fcc6-43fb-94d8-bf1701b862c3/resourceGroups/rg-loganalyticsv2demo/providers/Microsoft.OperationalInsights/workspaces/log-loganalyticsv2-migration-demo
VERBOSE: PUT with -1-byte payload
VERBOSE: received 8133-byte response of content type application/json; charset=utf-8
VERBOSE: POST with -1-byte payload
VERBOSE: received 1468-byte response of content type application/json; charset=utf-8
VERBOSE: Found required DCE info using Azure Resource Graph
VERBOSE:
VERBOSE: GET with 0-byte payload
VERBOSE: received 893-byte response of content type application/json; charset=utf-8
VERBOSE: Found required LogAnalytics info
VERBOSE:
VERBOSE: GET with 0-byte payload
VERBOSE:
VERBOSE: Creating/updating DCR [ dcr-demo-V2CustomTable5778_CL ] with limited payload
VERBOSE: /subscriptions/fce4f282-fcc6-43fb-94d8-bf1701b862c3/resourceGroups/rg-dcr-log-loganalyticsv2-migration-demo/providers/microsoft.insights/dataCollectionRules/dcr-demo-V2CustomTable5778_CL
VERBOSE: PUT with -1-byte payload
VERBOSE: received 2089-byte response of content type application/json; charset=utf-8
VERBOSE:
VERBOSE: Updating DCR [ dcr-demo-V2CustomTable5778_CL ] with full payload
VERBOSE: /subscriptions/fce4f282-fcc6-43fb-94d8-bf1701b862c3/resourceGroups/rg-dcr-log-loganalyticsv2-migration-demo/providers/microsoft.insights/dataCollectionRules/dcr-demo-V2CustomTable5778_CL
VERBOSE: PUT with -1-byte payload
VERBOSE: received 4805-byte response of content type application/json; charset=utf-8
VERBOSE:
VERBOSE: Waiting 10 sec to let Azure sync up so DCR rule can be retrieved from Azure Resource Graph
```

Structure 5/5: Upload data using Log Ingestion API

```
Post-AzLogAnalyticsLogIngestCustomLogDcrDce-Output -DceName $DceName `
                                                   -DcrName $DcrName `
                                                   -Data $DataVariable `
                                                   -TableName $TableName `
                                                   -AzAppId $LogIngestAppId `
                                                   -AzAppSecret $LogIngestAppSecret `
                                                   -TenantId $TenantId `
                                                   -Verbose:$Verbose
```

Automatic calculation of size of JSON

```
VERBOSE: Getting Data Collection Rules from Azure Resource Graph .... Please Wait !
VERBOSE: POST with -1-byte payload
VERBOSE: received 1468-byte response of content type application/json; charset=utf-8
VERBOSE: POST with -1-byte payload
VERBOSE: received 11713-byte response of content type application/json; charset=utf-8
VERBOSE: POST with -1-byte payload
VERBOSE: received 1468-byte response of content type application/json; charset=utf-8
VERBOSE: POST with -1-byte payload
VERBOSE: received 1302-byte response of content type application/json; charset=utf-8
```

Upload - mitigation of POST error 513 (dataset too large)

Upload data using DCR / DCE / Log Ingestion API

```
Post-AzLogAnalyticsLogIngestCustomLogDcrDce-Output -DceName $DceName `
                                                   -DcrName $DcrName `
                                                   -Data $DataVariable `
                                                   -TableName $TableName `
                                                   -AzAppId $LogIngestAppId `
                                                   -AzAppSecret $LogIngestAppSecret `
                                                   -TenantId $TenantId `
                                                   -Verbose:$Verbose `
                                                   -BatchAmount <number of rows>
```

Multiple records in data-set with different size – force specific batch-amounts per POST

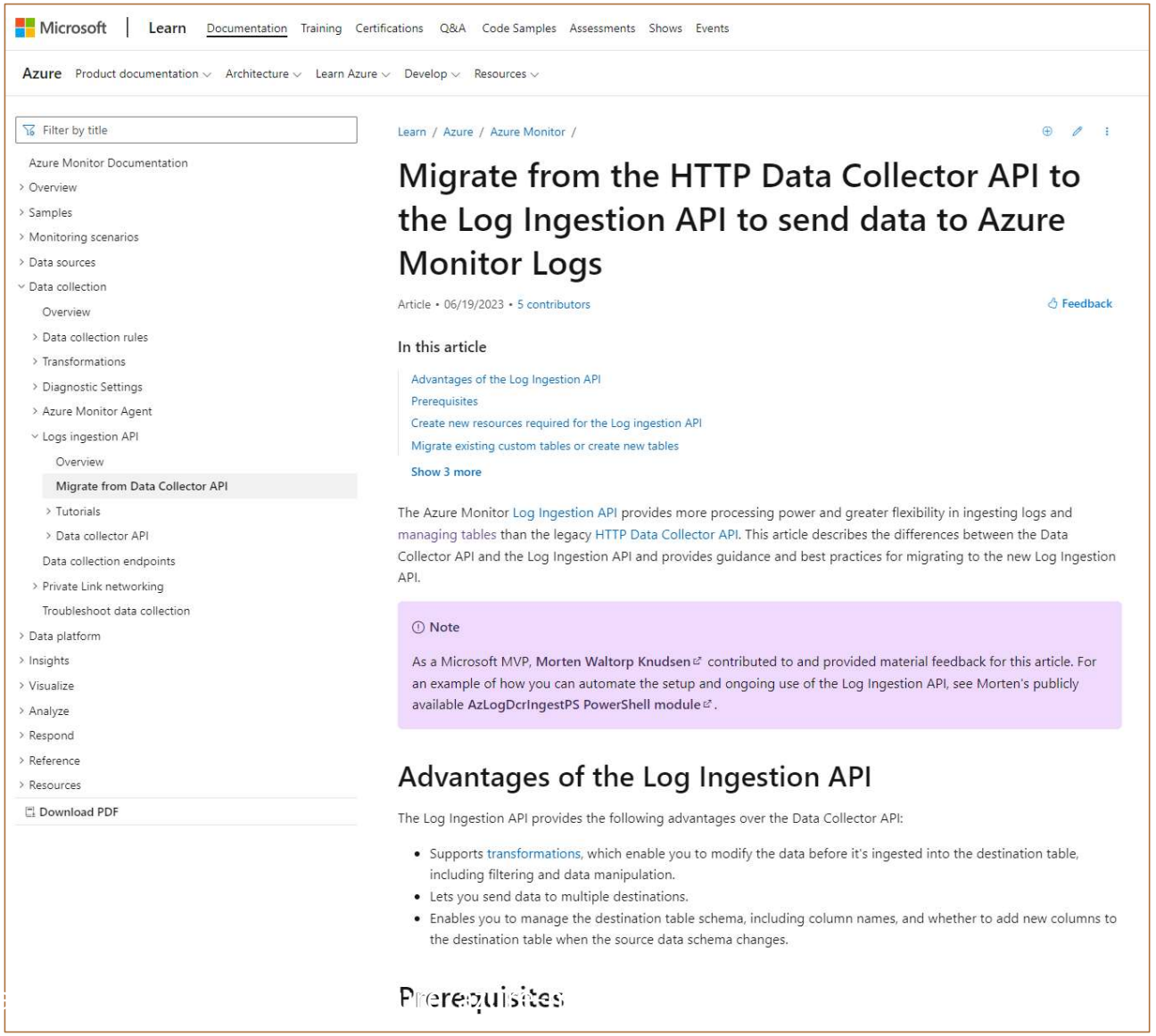
Limit	Value
Maximum size of API call	1 MB for both compressed and uncompressed data.
Maximum data/minute per DCR	2 GB for both compressed and uncompressed data. Retry after the duration listed in the Retry-After header in the response.
Maximum requests/minute per DCR	12,000. Retry after the duration listed in the Retry-After header in the response.

Migration Scenarios from v1 to v2

- Side-by-Side
  - New table with nice naming convention
  - Any integrations, dashboards, etc. can be adjusted before “turn-key”
  - Migration can be done ‘per table’
  - Old data will exist until retention period runs out

- Migrate existing table to DataCollectionRules-format
  - Re-use existing table name
  - Naming – 2 options
    - Add new naming convention to columns + make transformation into old columns
    - Re-use old naming
  - “Big bang” per table (not possible to revert back)
  - After migration of table
    - Sending using HTTP Data Collector with SAME schema is allowed – data will be accepted.
    - No schema changes using old HTTP Data Collector method
    - Schema-changes can only be made using DCR-based method

Migration guide



How to get started with Azure Logging v2 ?

Blog:
https://mortenknudsen.net

Blog by Morten Knudsen about Microsoft Security, Azure, M365 & Automation

Home | Blog Posts | MyGithub | MyPresentations | Meeting Microsoft | Meeting Tech Peers | About | Morten | Contact

Master Azure Logging in depth

02/04/2023 by Morten Knudsen

I am really **passioned** about the **logging capabilities** in **M365 Defender** and **Azure** with the power **to bring data back from clients, servers, cloud and 3rd party systems** – and getting **cool valuable information out of the data** – besides of course for **security hunting**.

As outlined in the last section of this intro/overview, I have prepared a series of blog posts to **master Azure logging in depth**. I will be releasing **19 blog-posts** as shown below.

You will learn about **Azure Log Ingestion Pipeline**, **Azure Data Collection Rules**, **Data Collection Endpoints**, **Azure LogAnalytics custom table (v2)**, **Azure Monitor Agent**, **Azure Monitor Private Link Scope**, **migration from v1 to v2**, etc.

To get you started, I have also prepared templates & scripts in my AzureLogLibrary (github) – with focus on the new capabilities.

Download deployment-script & ARM-templates to deploy all data collections

Understanding Azure logging, DCRs, DCEs, AMA & transformations in depth

- Understanding Azure logging capabilities in depth
- How to do data transformation using Workspace transformation for legacy upload methods
- Understanding the fundamentals of log-collection with Azure Monitor Agent & Azure Data Collection Rules
- Understanding Azure Data Collection Endpoint

Collecting data using Azure Monitor Agent (AMA)

- Collecting Security events using Azure Monitor Agent
- Collecting System & Application events using Azure Monitor Agent
- Collecting Performance data using Azure Monitor Agent, VMInsights and ServiceMap
- Collecting IIS logs using Azure Monitor Agent
- Collecting text logs using Azure Monitor Agent
- Collecting Syslogs using Azure Monitor Agent
- Collecting CEF Syslogs using Azure Monitor Agent

Transformations of data

- Tutorial – How to make data transformations using Data Collection Rules?

My Powershell module AzLogIngestPS for Azure Log Ingestion API

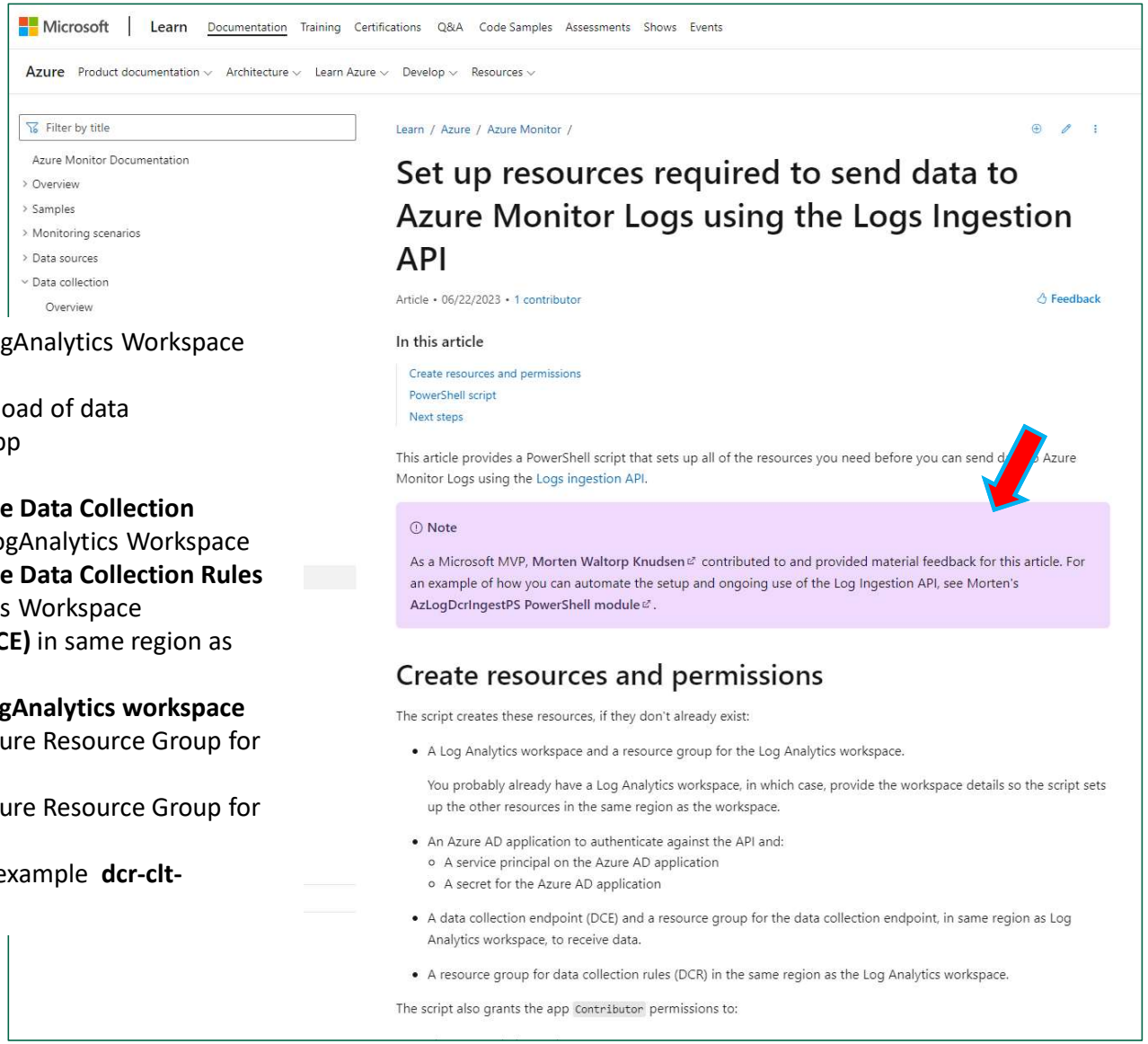
- AzLogDcrIngestPS – your helper to send data via Azure Pipeline, Azure Log Ingestion API & Azure Data Collection Rules into Azure LogAnalytics table
- AzLogDcrIngestPS – how to do data manipulation before sending data via Azure Pipeline, Log ingestion API & Azure Data Collection Rules into Azure LogAnalytics ?
- AzLogDcrInstPS – how to do transformation of your data using Azure Data Collection Rules into Azure LogAnalytics?
- AzLogDcrIngestPS – tips & tricks sending data via Azure Pipeline, Azure Log Ingestion API, Azure Data Collections into Azure LogAnalytics
- Introducing "Log-hub" – sending your data through intermediate hub into Azure Pipeline, Log Ingestion API, Azure LogAnalytics (github)

ClientInspector – showcase for AzLogIngestPS and Azure Log Ingestion

- ClientInspector – a cool showcase to demonstrate Log ingestion API, Azure Log Ingestion Pipeline, Azure Data Collection Rules and my new Powershell module AzLogDcrIngestPS
- ClientInspector-DeploymentKit – How can I get started with ClientInspector so I can play with Azure Pipeline, Azure Data Collection Rules and Azure LogAnalytics (Gibhub)?

Pre-req Environment for LogAnalytics (v2)

1. create **Azure Resource Group** for Azure LogAnalytics Workspace
2. create **Azure LogAnalytics Workspace**
3. create **Azure App registration** used for upload of data
4. create **Azure service principal** on Azure App
5. create needed **secret** on Azure app
6. create the **Azure Resource Group** for **Azure Data Collection Endpoint (DCE)** in same region as Azure LogAnalytics Workspace
7. create the **Azure Resource Group** for **Azure Data Collection Rules (DCR)** in same region as Azure LogAnalytics Workspace
8. create **Azure Data Collection Endpoint (DCE)** in same region as Azure LogAnalytics Workspace
9. **delegate permissions** for Azure App on **LogAnalytics workspace**
10. **delegate permissions** for Azure App on Azure Resource Group for **Azure Data Collection Rules (DCR)**
11. **delegate permissions** for Azure App on Azure Resource Group for **Azure Data Collection Endpoints (DCE)**
12. define **naming convention for DCRs** – for example **dcr-clt-InvClientBitlockerInfoV2_CL**



Microsoft | Learn Documentation Training Certifications Q&A Code Samples Assessments Shows Events

Azure Product documentation Architecture Learn Azure Develop Resources

Filter by title

Azure Monitor Documentation
Overview
Samples
Monitoring scenarios
Data sources
Data collection
Overview

Learn / Azure / Azure Monitor /

Set up resources required to send data to Azure Monitor Logs using the Logs Ingestion API

Article • 06/22/2023 • 1 contributor

Feedback

In this article

Create resources and permissions
PowerShell script
Next steps

This article provides a PowerShell script that sets up all of the resources you need before you can send d... Azure Monitor Logs using the Logs ingestion API.

Note

As a Microsoft MVP, **Morten Waltorp Knudsen** contributed to and provided material feedback for this article. For an example of how you can automate the setup and ongoing use of the Log Ingestion API, see Morten's **AzLogDcrIngestPS PowerShell module**.

Create resources and permissions

The script creates these resources, if they don't already exist:

- A Log Analytics workspace and a resource group for the Log Analytics workspace.

  You probably already have a Log Analytics workspace, in which case, provide the workspace details so the script sets up the other resources in the same region as the workspace.

- An Azure AD application to authenticate against the API and:
  - A service principal on the Azure AD application
  - A secret for the Azure AD application
- A data collection endpoint (DCE) and a resource group for the data collection endpoint, in same region as Log Analytics workspace, to receive data.
- A resource group for data collection rules (DCR) in the same region as the Log Analytics workspace.

The script also grants the app `Contributor` permissions to:

AzLogDcrIngestPS - Detailed Documentation
https://github.com/KnudsenMorten/AzLogDcrIngestPS

Videos

I have provided 4 demos for you to try, but if you want to see it first using video, check out these videos:

Video 3m 19s - Running ClientInspector using commandline (normal mode)
Video 1m 40s - Automatic creation of 2 tables & DCRs (verbose mode)
Video 1m 37s - Automatic creation of 2 tables & DCRs (normal mode)
Video 1m 34s - See schema of DCR and table)
Video 2m 19s - Data manipulation
Video 1m 58s - Kusto queries against data
Video 3m 01s - Dashboards
Video 0m 48s - Sample usage of data - lookup against Lenovo warranty db
Video 7m 25s - Deployment via ClientInspector DeploymentKit

Download latest version

You can download latest version of AzLogDcrIngestPS here - or install from Powershell Gallery:

Install AzLogDcringestPS from Powershell Gallery

```
install-module AzLogDcrIngestPS
```

Download AzLogDcringestPS module from this Github repository

Quick links for more information

How to get started in your own environment (demo)
Background for building this Powershell module
Deep-dive about Azure Data Collection Rules (DCRs)
Deep-dive about Log Ingestion API
Architecture, Schema & Networking
Security
Source data - what data can I use ?
Example of how to use the functions
How can I modify the schema of LogAnalytics table & Data Collection Rule, when the source object schema changes ?
How to enable verbose-mode & get more help ?
Integration of AzLogDcrIngest in your scripts
Function synopsis
Detailed - Data Manipulation
Detailed - Table/DCR/Schema/Transformation management
Detailed - Data Out (upload to Azure LogAnalytics) Detailed - Support functions (security)
Contact me

Download this presentation

More information in PPT

Thank You

Connect with me on LinkedIn

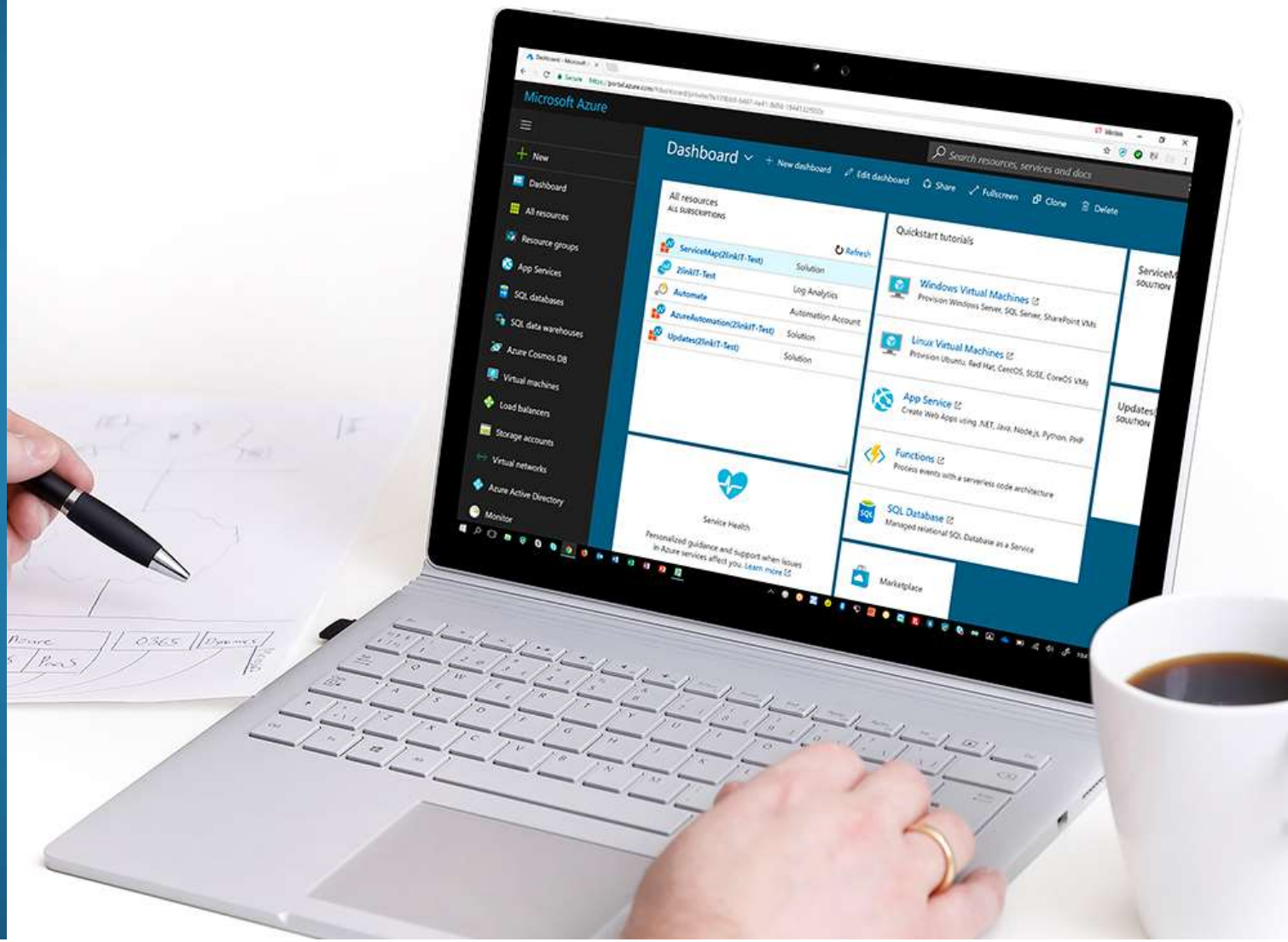
Linkedin:
https://www.linkedin.com/in/mortenwaltorpknudsen

X/Twitter:
https://twitter.com/knudsenmortendk

Github:
https://github.com/KnudsenMorten

More deep-dive information, if people are interested

(not part of main presentation)



Side-by-Side Migration using AzLogDcrIngestPS

Video
(8:30)

Migrate existing table to DataCollectionRules-format

Video
(6:00)

Migrate existing table to DataCollectionRules-format

(Step 1/5) - Migration of existing table to V2-format (DCR-based)

```powershell
$Headers = Get-AzAccessTokenManagement -AzAppId $LogIngestAppId `
                                       -AzAppSecret $LogIngestAppSecret `
                                       -TenantId $TenantId `
                                       -Verbose:$Verbose

# Get existing LA table info
$TableUrl = "https://management.azure.com" + $LogAnalyticsWorkspaceResourceId + "/tables?api-version=2021-12-01-preview"
$tbl = invoke-restmethod -UseBasicParsing -Uri $TableUrl -Method GET -Headers $Headers
$res = $tbl.value.properties | Where-Object { $_.schema.name -like "*$($DemoNumber)*" }
$res.schema
$res.schema.columns

# Migrate table
$Uri         = "https://management.azure.com" + $LogAnalyticsWorkspaceResourceId + "/tables/$($TableName)_CL/migrate?api-version=2021-12-01-preview"
$Response    = invoke-webrequest -UseBasicParsing -Method POST -Uri $Uri -Headers $Headers


# Get existing LA table info
$TableUrl = "https://management.azure.com" + $LogAnalyticsWorkspaceResourceId + "/tables?api-version=2021-12-01-preview"
$tbl = invoke-restmethod -UseBasicParsing -Uri $TableUrl -Method GET -Headers $Headers
$res = $tbl.value.properties | Where-Object { $_.schema.name -like "*$($DemoNumber)*" }
$res.schema
$res.schema.columns
```

```
tableSubType            : Classic
name                    : V1CustomTable2997_CL
tableType               : CustomLog
```

```
tableSubType            : DataCollectionRuleBased
name                    : V1CustomTable2997_CL
tableType               : CustomLog
```

Migrate existing table to DataCollectionRules-format

(Step 2/5) - Add new table structure/naming

```powershell
#----------------------------------------
# (M1) Data to Upload (sample)
#----------------------------------------

$DataVariable = @()

$item = New-Object PSObject
$item | Add-Member -type NoteProperty -Name 'Computer' -Value $Env:ComputerName
$item | Add-Member -type NoteProperty -Name 'ColumnString' -Value 'StringText'
$item | Add-Member -type NoteProperty -Name 'ColumnDate' -Value (Get-date)
$item | Add-Member -type NoteProperty -Name 'ColumnNumber' -Value (Get-random -max 10000)
#$item | Add-Member -type NoteProperty -Name 'ColumnStringExtra' -Value "Extra"

$DataVariable += $item


#-----------------------------------------------------------------------------------------
# (M2) Get info about DCEs & DCRs
#-----------------------------------------------------------------------------------------

# building global variable with all DCEs, which can be viewed by Log Ingestion app
$global:AzDceDetails = Get-AzDceListAll -AzAppId $LogIngestAppId -AzAppSecret $LogIngestAppSecret -TenantId $TenantId -Verbose:$Verbose

# building global variable with all DCRs, which can be viewed by Log Ingestion app
$global:AzDcrDetails = Get-AzDcrListAll -AzAppId $LogIngestAppId -AzAppSecret $LogIngestAppSecret -TenantId $TenantId -Verbose:$Verbose
```

Migrate existing table to DataCollectionRules-format

(Step 3/5) - Add new table structure/naming

```
#--------------------------------------
# (M3) Data Manipulation
#--------------------------------------

# add CollectionTime to existing array
$DataVariable = Add-CollectionTimeToAllEntriesInArray -Data $DataVariable -Verbose:$Verbose

# add Computer, ComputerFqdn & UserLoggedOn info to existing array
$DataVariable = Add-ColumnDataToAllEntriesInArray -Data $DataVariable -Column1Name Computer -Column1Data $Env:ComputerName -Column2Name
ComputerFqdn -Column2Data $DnsName -Column3Name UserLoggedOn -Column3Data $UserLoggedOn -Verbose:$Verbose

# Validating/fixing schema data structure of source data
$DataVariable = ValidateFix-AzLogAnalyticsTableSchemaColumnNames -Data $DataVariable -Verbose:$Verbose

# Aligning data structure with schema (requirement for DCR)
$DataVariable = Build-DataArrayToAlignWithSchema -Data $DataVariable -Verbose:$Verbose
```

Migrate existing table to DataCollectionRules-format

(Step 4/5) - Add new table structure/naming

```
#-------------------------------------------------------------------------------------------
# (M4) Create/Update Schema for LogAnalytics Table & Data Collection Rule schema
#-------------------------------------------------------------------------------------------

CheckCreateUpdate-TableDcr-Structure -AzLogWorkspaceResourceId $LogAnalyticsWorkspaceResourceId -SchemaMode Migrate `
                                     -AzAppId $LogIngestAppId -AzAppSecret $LogIngestAppSecret -TenantId $TenantId -Verbose:$Verbose `
                                     -DceName $DceName -DcrName $DcrName -DcrResourceGroup $AzDcrResourceGroup `
                                     -TableName $TableName -Data $DataVariable `
                                     -LogIngestServicePricipleObjectId $AzDcrLogIngestServicePrincipalObjectId `
                                     -AzDcrSetLogIngestApiAppPermissionsDcrLevel $AzDcrSetLogIngestApiAppPermissionsDcrLevel `
                                     -AzLogDcrTableCreateFromAnyMachine $AzLogDcrTableCreateFromAnyMachine `
                                     -AzLogDcrTableCreateFromReferenceMachine $AzLogDcrTableCreateFromReferenceMachine
```

LogAnalytics table schema will be extended with any new data (merge)
DCR will be created with schema based on (updated) LogAnalytics table

Table + DCR is now ready to accept new data !

Migrate existing table to DataCollectionRules-format

(Step 5/5) - Transformation

```
$transformKql = "source | extend TimeGenerated = now(), ColumnDate_value_t = ColumnDate, ColumnString_s = ColumnString"

$DcrResourceId = ($global:AzDcrDetails | Where-Object { $_.name -eq $DcrName }).id
Update-AzDataCollectionRuleTransformKql -DcrResourceId $DcrResourceId -transformKql $transformKql -Verbose:$Verbose

Get-AzDataCollectionRuleTransformKql -DcrResourceId $DcrResourceId
```

Migrate existing table to DataCollectionRules-format

(Step 1/5) - Migration of existing table to V2-format (DCR-based)

```powershell
$Headers = Get-AzAccessTokenManagement -AzAppId $LogIngestAppId `
                                       -AzAppSecret $LogIngestAppSecret `
                                       -TenantId $TenantId `
                                       -Verbose:$Verbose

# Get existing LA table info
$TableUrl = "https://management.azure.com" + $LogAnalyticsWorkspaceResourceId + "/tables?api-version=2021-12-01-preview"
$tbl = invoke-restmethod -UseBasicParsing -Uri $TableUrl -Method GET -Headers $Headers
$res = $tbl.value.properties | Where-Object { $_.schema.name -like "*$($DemoNumber)*" }
$res.schema
$res.schema.columns

# Migrate table
$Uri         = "https://management.azure.com" + $LogAnalyticsWorkspaceResourceId + "/tables/$($TableName)_CL/migrate?api-version=2021-12-01-preview"
$Response    = invoke-webrequest -UseBasicParsing -Method POST -Uri $Uri -Headers $Headers


# Get existing LA table info
$TableUrl = "https://management.azure.com" + $LogAnalyticsWorkspaceResourceId + "/tables?api-version=2021-12-01-preview"
$tbl = invoke-restmethod -UseBasicParsing -Uri $TableUrl -Method GET -Headers $Headers
$res = $tbl.value.properties | Where-Object { $_.schema.name -like "*$($DemoNumber)*" }
$res.schema
$res.schema.columns
```

```
tableSubType            : Classic
name                    : V1CustomTable2997_CL
tableType               : CustomLog
```

```
tableSubType            : DataCollectionRuleBased
name                    : V1CustomTable2997_CL
tableType               : CustomLog
```

Migrate existing table to DataCollectionRules-format

(Step 2/5) - Add new table structure/naming

```powershell
#----------------------------------------
# (M1) Data to Upload (sample)
#----------------------------------------

$DataVariable = @()

$item = New-Object PSObject
$item | Add-Member -type NoteProperty -Name 'Computer' -Value $Env:ComputerName
$item | Add-Member -type NoteProperty -Name 'ColumnString' -Value 'StringText'
$item | Add-Member -type NoteProperty -Name 'ColumnDate' -Value (Get-date)
$item | Add-Member -type NoteProperty -Name 'ColumnNumber' -Value (Get-random -max 10000)
#$item | Add-Member -type NoteProperty -Name 'ColumnStringExtra' -Value "Extra"

$DataVariable += $item


#-----------------------------------------------------------------------------------------
# (M2) Get info about DCEs & DCRs
#-----------------------------------------------------------------------------------------

# building global variable with all DCEs, which can be viewed by Log Ingestion app
$global:AzDceDetails = Get-AzDceListAll -AzAppId $LogIngestAppId -AzAppSecret $LogIngestAppSecret -TenantId $TenantId -Verbose:$Verbose

# building global variable with all DCRs, which can be viewed by Log Ingestion app
$global:AzDcrDetails = Get-AzDcrListAll -AzAppId $LogIngestAppId -AzAppSecret $LogIngestAppSecret -TenantId $TenantId -Verbose:$Verbose
```

Migrate existing table to DataCollectionRules-format

(Step 3/5) - Add new table structure/naming

```
#--------------------------------------
# (M3) Data Manipulation
#--------------------------------------

# add CollectionTime to existing array
$DataVariable = Add-CollectionTimeToAllEntriesInArray -Data $DataVariable -Verbose:$Verbose

# add Computer, ComputerFqdn & UserLoggedOn info to existing array
$DataVariable = Add-ColumnDataToAllEntriesInArray -Data $DataVariable -Column1Name Computer -Column1Data $Env:ComputerName -Column2Name
ComputerFqdn -Column2Data $DnsName -Column3Name UserLoggedOn -Column3Data $UserLoggedOn -Verbose:$Verbose

# Validating/fixing schema data structure of source data
$DataVariable = ValidateFix-AzLogAnalyticsTableSchemaColumnNames -Data $DataVariable -Verbose:$Verbose

# Aligning data structure with schema (requirement for DCR)
$DataVariable = Build-DataArrayToAlignWithSchema -Data $DataVariable -Verbose:$Verbose
```

Migrate existing table to DataCollectionRules-format

(Step 4/5) - Add new table structure/naming

```
#-------------------------------------------------------------------------------------------
# (M4) Create/Update Schema for LogAnalytics Table & Data Collection Rule schema
#-------------------------------------------------------------------------------------------

CheckCreateUpdate-TableDcr-Structure -AzLogWorkspaceResourceId $LogAnalyticsWorkspaceResourceId -SchemaMode Migrate `
                                     -AzAppId $LogIngestAppId -AzAppSecret $LogIngestAppSecret -TenantId $TenantId -Verbose:$Verbose `
                                     -DceName $DceName -DcrName $DcrName -DcrResourceGroup $AzDcrResourceGroup `
                                     -TableName $TableName -Data $DataVariable `
                                     -LogIngestServicePricipleObjectId $AzDcrLogIngestServicePrincipalObjectId `
                                     -AzDcrSetLogIngestApiAppPermissionsDcrLevel $AzDcrSetLogIngestApiAppPermissionsDcrLevel `
                                     -AzLogDcrTableCreateFromAnyMachine $AzLogDcrTableCreateFromAnyMachine `
                                     -AzLogDcrTableCreateFromReferenceMachine $AzLogDcrTableCreateFromReferenceMachine
```

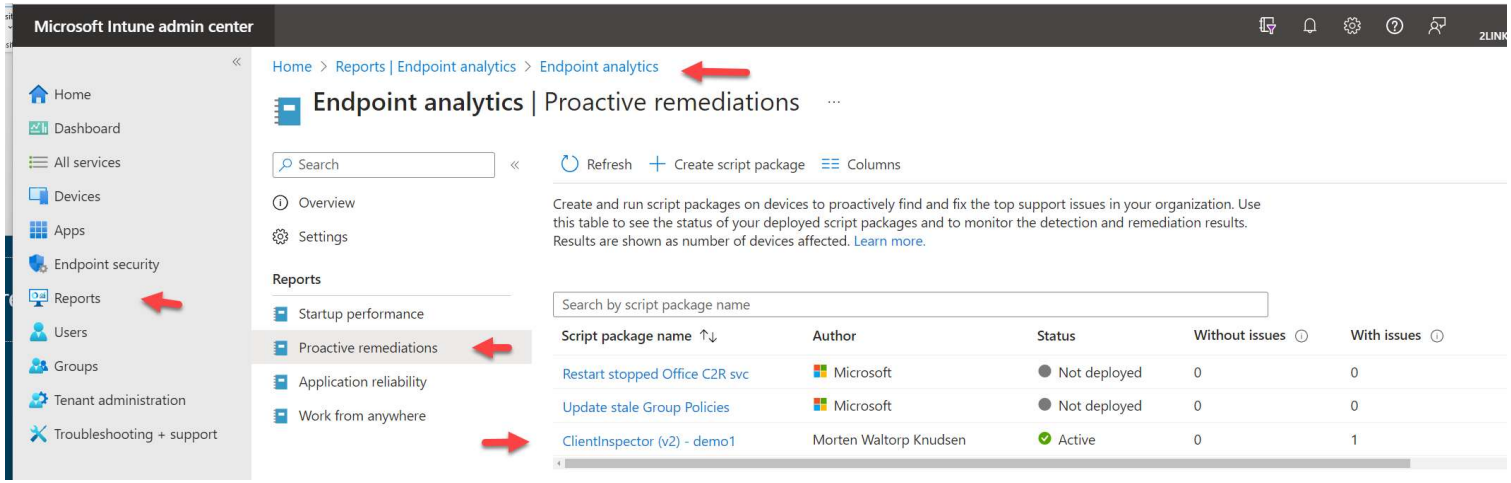
LogAnalytics table schema will be extended with any new data (merge)
DCR will be created with schema based on (updated) LogAnalytics table

Table + DCR is now ready to accept new data !

Migrate existing table to DataCollectionRules-format

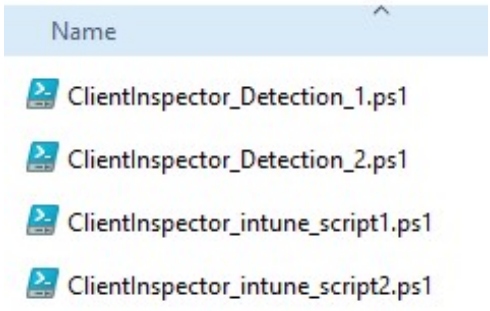
(Step 5/5) - Transformation

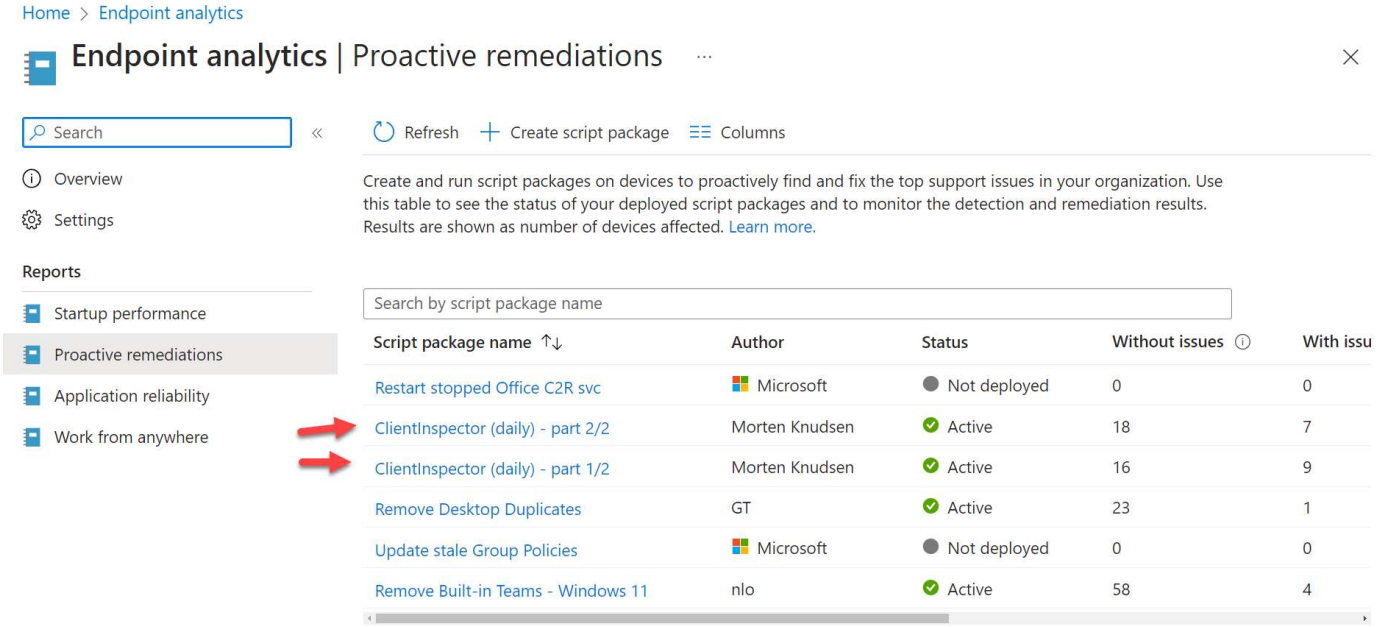
```
$transformKql = "source | extend TimeGenerated = now(), ColumnDate_value_t = ColumnDate, ColumnString_s = ColumnString"

$DcrResourceId = ($global:AzDcrDetails | Where-Object { $_.name -eq $DcrName }).id
Update-AzDataCollectionRuleTransformKql -DcrResourceId $DcrResourceId -transformKql $transformKql -Verbose:$Verbose

Get-AzDataCollectionRuleTransformKql -DcrResourceId $DcrResourceId
```



Intune deployment (<200 Kb in size)

Intune deployment >200 Kb size, split





ClientInspector - What is being collected ?

- User Logged On to Client
- Computer information
  - bios, processor, hardware info, Windows OS info, OS information, last restart
- Installed applications
  - both using WMI and registry
- Antivirus Security Center from Windows
  - default antivirus, state, configuration
- Microsoft Defender Antivirus
  - all settings including ASR, exclusions, realtime protection, etc
- Office - version, update channel config, SKUs
- VPN client - version, product
- LAPS – version
- Admin By Request (3rd party) – version
- Bitlocker - configuration
- Windows Update
  - last result (when), windows update source information (where), pending updates, last installations (what)
- Eventlog
  - look for specific events including logon events, blue screens, etc.
- Network adapters - configuration, installed adapters
- IP information for all adapters
- Local administrators group membership
- Windows firewall
  - settings for all 3 modes
- Group Policy - last refresh
- TPM information
  - relevant to detect machines with/without TPM

```
PS  get-command -module AzLogDcrIngestPS

CommandType     Name                                               Version    Source
-----------     ----                                               -------    ------
Function        Add-CollectionTimeToAllEntriesInArray              1.1.17     AzLogDcrIngestPS
Function        Add-ColumnDataToAllEntriesInArray                  1.1.17     AzLogDcrIngestPS
Function        Build-DataArrayToAlignWithSchema                   1.1.17     AzLogDcrIngestPS
Function        CheckCreateUpdate-TableDcr-Structure               1.1.17     AzLogDcrIngestPS
Function        Convert-CimArrayToObjectFixStructure               1.1.17     AzLogDcrIngestPS
Function        Convert-PSArrayToObjectFixStructure                1.1.17     AzLogDcrIngestPS
Function        CreateUpdate-AzDataCollectionRuleLogIngestCusto... 1.1.17     AzLogDcrIngestPS
Function        CreateUpdate-AzLogAnalyticsCustomLogTableDcr       1.1.17     AzLogDcrIngestPS
Function        Delete-AzDataCollectionRules                       1.1.17     AzLogDcrIngestPS
Function        Delete-AzLogAnalyticsCustomLogTables               1.1.17     AzLogDcrIngestPS
Function        Filter-ObjectExcludeProperty                       1.1.17     AzLogDcrIngestPS
Function        Get-AzAccessTokenManagement                        1.1.17     AzLogDcrIngestPS
Function        Get-AzDceListAll                                   1.1.17     AzLogDcrIngestPS
Function        Get-AzDcrDceDetails                                1.1.17     AzLogDcrIngestPS
Function        Get-AzDcrListAll                                   1.1.17     AzLogDcrIngestPS
Function        Get-AzLogAnalyticsTableAzDataCollectionRuleStatus  1.1.17     AzLogDcrIngestPS
Function        Get-ObjectSchemaAsArray                            1.1.17     AzLogDcrIngestPS
Function        Get-ObjectSchemaAsHash                             1.1.17     AzLogDcrIngestPS
Function        Post-AzLogAnalyticsLogIngestCustomLogDcrDce        1.1.17     AzLogDcrIngestPS
Function        Post-AzLogAnalyticsLogIngestCustomLogDcrDce-Output 1.1.17     AzLogDcrIngestPS
Function        Update-AzDataCollectionRuleDceEndpoint             1.1.17     AzLogDcrIngestPS
Function        Update-AzDataCollectionRuleResetTransformKqlDef... 1.1.17     AzLogDcrIngestPS
Function        Update-AzDataCollectionRuleTransformKql            1.1.17     AzLogDcrIngestPS
Function        ValidateFix-AzLogAnalyticsTableSchemaColumnNames   1.1.17     AzLogDcrIngestPS
```

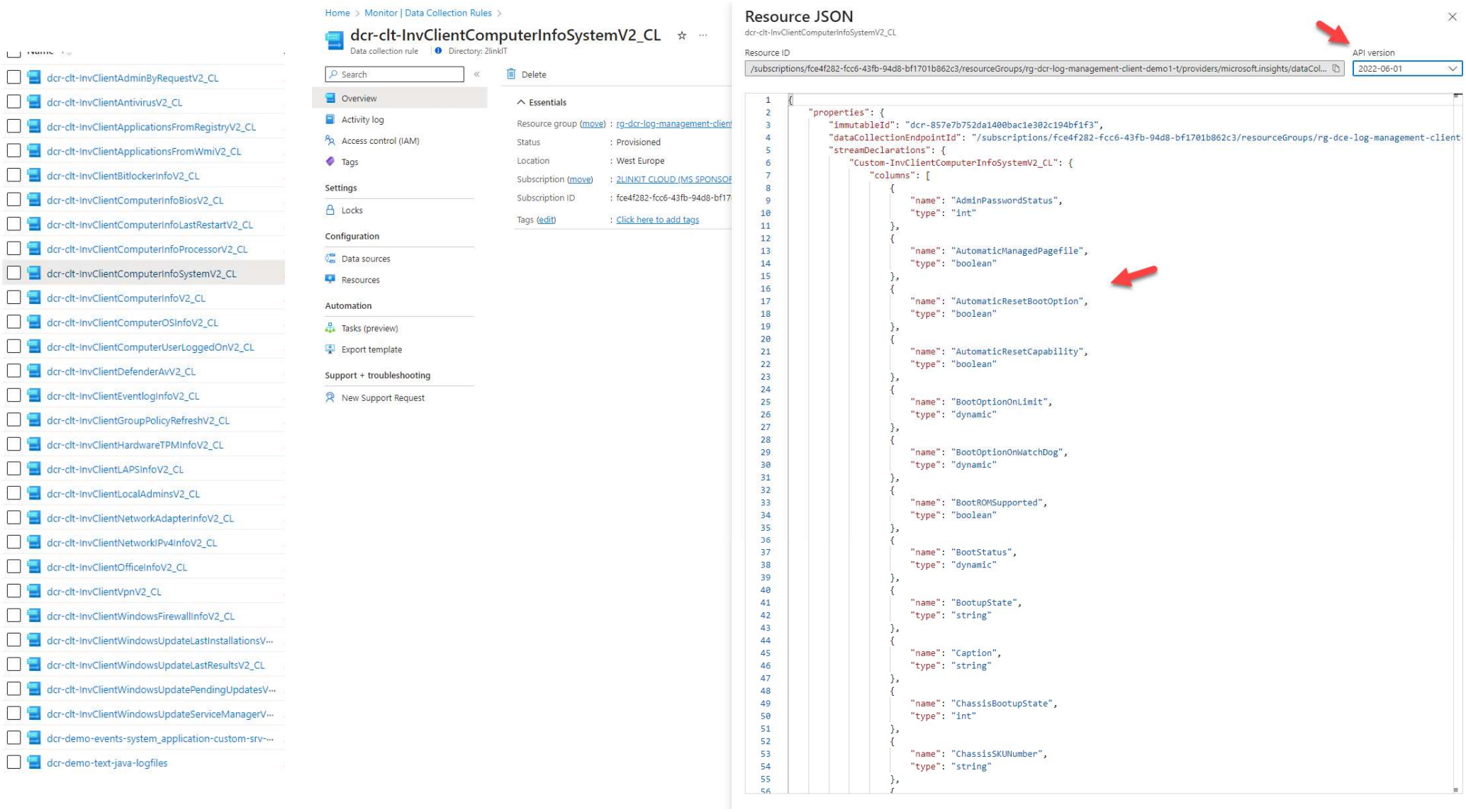
```
get-help Add-CollectionTimeToAllEntriesInArray -full

NAME
    Add-CollectionTimeToAllEntriesInArray

SYNOPSIS
    Add property CollectionTime (based on current time) to all entries on the object


SYNTAX
    Add-CollectionTimeToAllEntriesInArray [-Data] <Array> [<CommonParameters>]


DESCRIPTION
    Gives capability to do proper searching in queries to find latest set of records with same collection tim
    Time Generated cannot be used when you are sending data in batches, as TimeGenerated will change
    An example where this is important is a complete list of applications for a computer. We want all applica
    show up when queriying for the latest data


PARAMETERS
    -Data <Array>
        Object to modify

        Required?                    true
        Position?                    1
        Default value
        Accept pipeline input?       false
        Accept wildcard characters?  false

    <CommonParameters>
        This cmdlet supports the common parameters: Verbose, Debug,
        ErrorAction, ErrorVariable, WarningAction, WarningVariable,
        OutBuffer, PipelineVariable, and OutVariable. For more information, see
        about_CommonParameters (https:/go.microsoft.com/fwlink/?LinkID=113216).

INPUTS
    None. You cannot pipe objects


OUTPUTS
    Updated object with CollectionTime


    -------------------------- EXAMPLE 1 --------------------------

    PS C:\>#-------------------------------------------------------------------------------------------

    # Variables
    #-------------------------------------------------------------------------------------------
    $Verbose                  = $true  # $true or $false

    #-------------------------------------------------------------------------------------------
    # Collecting data (in)
    #-------------------------------------------------------------------------------------------
    $DNSName                  = (Get-CimInstance win32_computersystem).DNSHostName +"." + (Get-CimInstance w
    $ComputerName             = (Get-CimInstance win32_computersystem).DNSHostName
```

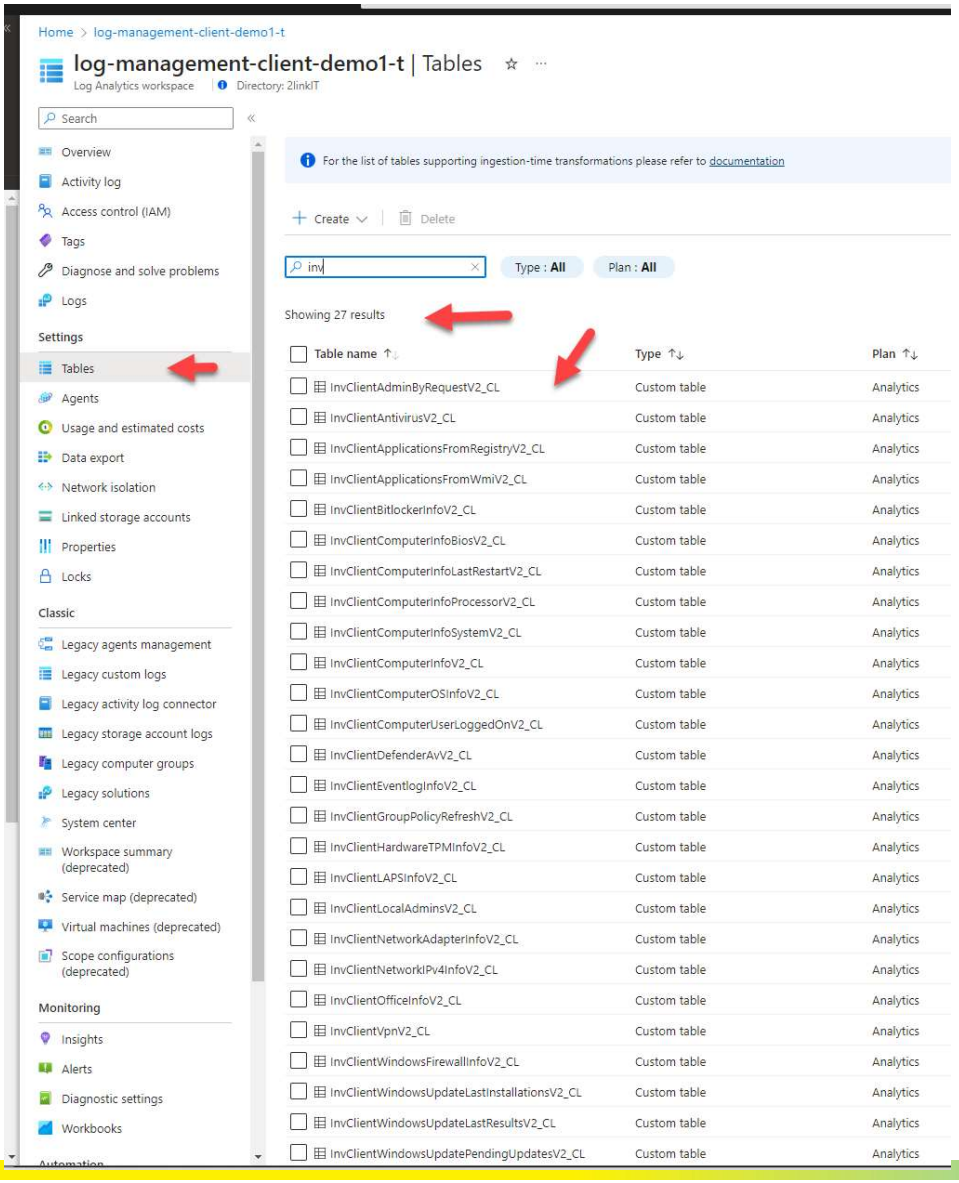
```
.\ClientInspector.ps1 -verbose:$false -function:download

ClientInspector | Inventory of Operational & Security-related information
Developed by Morten Knudsen, Microsoft MVP - for free community use

Downloading latest version of module AzLogDcrIngestPS from https://github.com/KnudsenMorten/CientInspectorV2
into local path D:\scripts\ClientInspectorV2
```

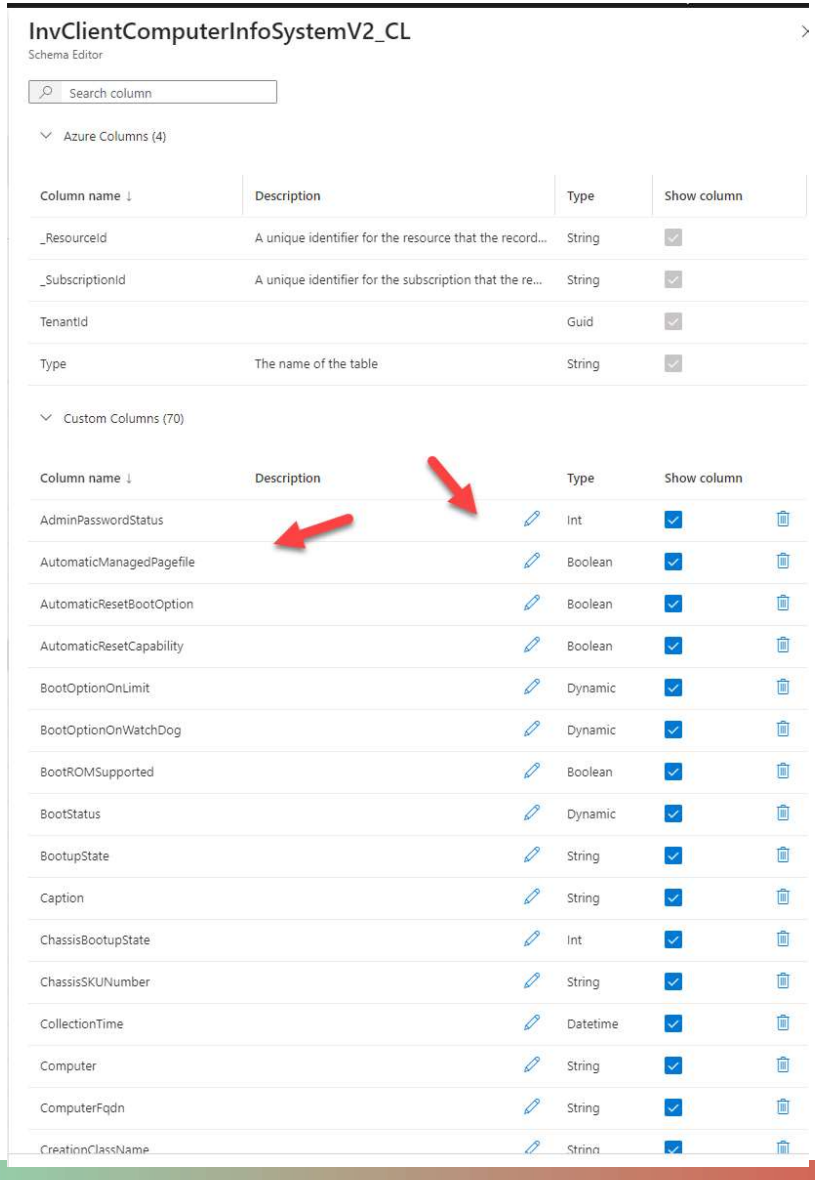
```
.\ClientInspector.ps1 -verbose:$false -function:localpath

ClientInspector | Inventory of Operational & Security-related information
Developed by Morten Knudsen, Microsoft MVP - for free community use

Using AzLogDcrIngestPS module from local path D:\scripts\ClientInspectorV2
```

```
.\ClientInspector.ps1 -verbose:$false -function:PSGallery -scope:CurrentUser

ClientInspector | Inventory of Operational & Security-related information
Developed by Morten Knudsen, Microsoft MVP - for free community use

Powershell module was not found !
Installing in scope currentuser .... Please Wait !
```

```
.\ClientInspector.ps1 -verbose:$false -function:PsGallery -scope:currentuser

ClientInspector | Inventory of Operational & Security-related information
Developed by Morten Knudsen, Microsoft MVP - for free community use

Checking latest version at PsGallery for AzLogDcrIngestPS module
OK - Running latest version
```

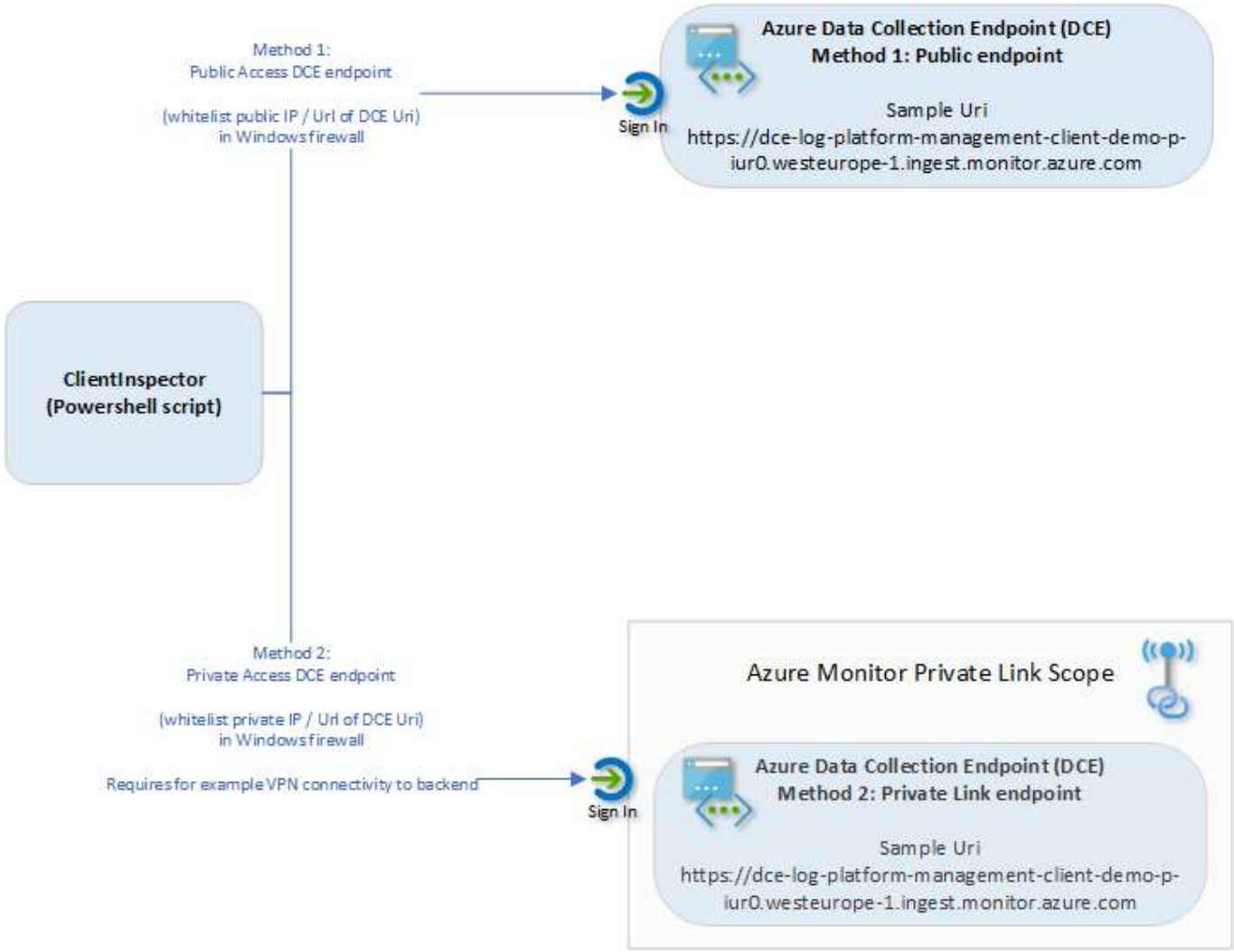

dcr-clt-InvClientAdminByRequestV2_CL
dcr-clt-InvClientAntivirusV2_CL
dcr-clt-InvClientApplicationsFromRegistryV2_CL
dcr-clt-InvClientApplicationsFromWmiV2_CL
dcr-clt-InvClientBitlockerInfoV2_CL
dcr-clt-InvClientComputerInfoBiosV2_CL
dcr-clt-InvClientComputerInfoLastRestartV2_CL
dcr-clt-InvClientComputerInfoProcessorV2_CL
dcr-clt-InvClientComputerInfoSystemV2_CL
dcr-clt-InvClientComputerInfoV2_CL
dcr-clt-InvClientComputerOSInfoV2_CL
dcr-clt-InvClientComputerUserLoggedOnV2_CL
dcr-clt-InvClientDefenderAvV2_CL
dcr-clt-InvClientEventlogInfoV2_CL
dcr-clt-InvClientGroupPolicyRefreshV2_CL
dcr-clt-InvClientHardwareTPMInfoV2_CL
dcr-clt-InvClientLAPSInfoV2_CL
dcr-clt-InvClientLocalAdminsV2_CL
dcr-clt-InvClientNetworkAdapterInfoV2_CL
dcr-clt-InvClientNetworkIPv4InfoV2_CL
dcr-clt-InvClientOfficeInfoV2_CL
dcr-clt-InvClientVpnV2_CL
dcr-clt-InvClientWindowsFirewallInfoV2_CL
dcr-clt-InvClientWindowsUpdateLastInstallationsV...
dcr-clt-InvClientWindowsUpdateLastResultsV2_CL
dcr-clt-InvClientWindowsUpdatePendingUpdatesV...
dcr-clt-InvClientWindowsUpdateServiceManagerV...
dcr-demo-events-system_application-custom-srv-...
dcr-demo-text-java-logfiles
Home > Monitor | Data Collection Rules >
dcr-clt-InvClientComputerInfoSystemV2_CL
Data collection rule
Directory: 2linkIT
Search
Delete
Overview
Activity log
Access control (IAM)
Tags
Settings
Locks
Configuration
Data sources
Resources
Automation
Tasks (preview)
Export template
Support + troubleshooting
New Support Request
Essentials
Resource group (move) : rg-dcr-log-management-client
Status : Provisioned
Location : West Europe
Subscription (move) : 2LINKIT CLOUD (MS SPONSOR
Subscription ID : fce4f282-fcc6-43fb-94d8-bf17
Tags (edit) : Click here to add tags
Resource JSON
dcr-clt-InvClientComputerInfoSystemV2_CL
Resource ID
/subscriptions/fce4f282-fcc6-43fb-94d8-bf1701b862c3/resourceGroups/rg-dcr-log-management-client-demo1-t/providers/microsoft.insights/dataCol...
API version
2022-06-01
{
"properties": {
"immutableId": "dcr-857e7b752da1400bac1e302c194bf1f3",
"dataCollectionEndpointId": "/subscriptions/fce4f282-fcc6-43fb-94d8-bf1701b862c3/resourceGroups/rg-dce-log-management-client
"streamDeclarations": {
"Custom-InvClientComputerInfoSystemV2_CL": {
"columns": [
{
"name": "AdminPasswordStatus",
"type": "int"
},
{
"name": "AutomaticManagedPagefile",
"type": "boolean"
},
{
"name": "AutomaticResetBootOption",
"type": "boolean"
},
{
"name": "AutomaticResetCapability",
"type": "boolean"
},
{
"name": "BootOptionOnLimit",
"type": "dynamic"
},
{
"name": "BootOptionOnWatchDog",
"type": "dynamic"
},
{
"name": "BootROMSupported",
"type": "boolean"
},
{
"name": "BootStatus",
"type": "dynamic"
},
{
"name": "BootupState",
"type": "string"
},
{
"name": "Caption",
"type": "string"
},
{
"name": "ChassisBootupState",
"type": "int"
},
{
"name": "ChassisSKUNumber",
"type": "string"
},

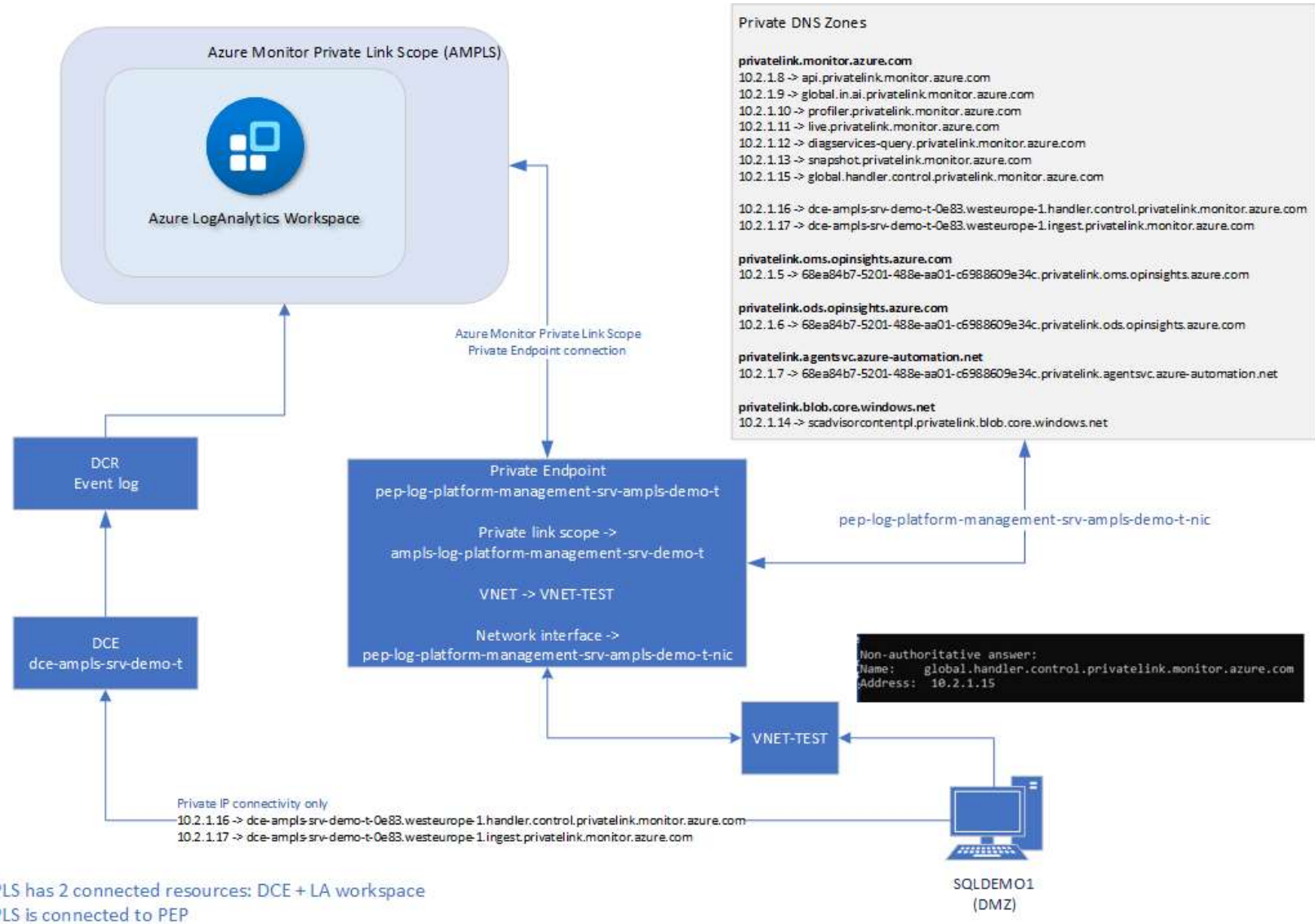

Home > log-management-client-demo1-t
log-management-client-demo1-t | Tables
Log Analytics workspace
Directory: 2linkIT
Search
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
Logs
Settings
Tables
Agents
Usage and estimated costs
Data export
Network isolation
Linked storage accounts
Properties
Locks
Classic
Legacy agents management
Legacy custom logs
Legacy activity log connector
Legacy storage account logs
Legacy computer groups
Legacy solutions
System center
Workspace summary (deprecated)
Service map (deprecated)
Virtual machines (deprecated)
Scope configurations (deprecated)
Monitoring
Insights
Alerts
Diagnostic settings
Workbooks
For the list of tables supporting ingestion-time transformations please refer to documentation
Create
Delete
inv
Type : All
Plan : All
Showing 27 results
Table name
Type
Plan
InvClientAdminByRequestV2_CL Custom table Analytics
InvClientAntivirusV2_CL Custom table Analytics
InvClientApplicationsFromRegistryV2_CL Custom table Analytics
InvClientApplicationsFromWmiV2_CL Custom table Analytics
InvClientBitlockerInfoV2_CL Custom table Analytics
InvClientComputerInfoBiosV2_CL Custom table Analytics
InvClientComputerInfoLastRestartV2_CL Custom table Analytics
InvClientComputerInfoProcessorV2_CL Custom table Analytics
InvClientComputerInfoSystemV2_CL Custom table Analytics
InvClientComputerInfoV2_CL Custom table Analytics
InvClientComputerOSInfoV2_CL Custom table Analytics
InvClientComputerUserLoggedOnV2_CL Custom table Analytics
InvClientDefenderAvV2_CL Custom table Analytics
InvClientEventlogInfoV2_CL Custom table Analytics
InvClientGroupPolicyRefreshV2_CL Custom table Analytics
InvClientHardwareTPMInfoV2_CL Custom table Analytics
InvClientLAPSInfoV2_CL Custom table Analytics
InvClientLocalAdminsV2_CL Custom table Analytics
InvClientNetworkAdapterInfoV2_CL Custom table Analytics
InvClientNetworkIPv4InfoV2_CL Custom table Analytics
InvClientOfficeInfoV2_CL Custom table Analytics
InvClientVpnV2_CL Custom table Analytics
InvClientWindowsFirewallInfoV2_CL Custom table Analytics
InvClientWindowsUpdateLastInstallationsV2_CL Custom table Analytics
InvClientWindowsUpdateLastResultsV2_CL Custom table Analytics
InvClientWindowsUpdatePendingUpdatesV2_CL Custom table Analytics


InvClientComputerInfoSystemV2_CL
Schema Editor
Search column
Azure Columns (4)
Column name
Description
Type
Show column
_ResourceId A unique identifier for the resource that the record... String
_SubscriptionId A unique identifier for the subscription that the re... String
TenantId Guid
Type The name of the table String
Custom Columns (70)
Column name
Description
Type
Show column
AdminPasswordStatus Int
AutomaticManagedPagefile Boolean
AutomaticResetBootOption Boolean
AutomaticResetCapability Boolean
BootOptionOnLimit Dynamic
BootOptionOnWatchDog Dynamic
BootROMSupported Boolean
BootStatus Dynamic
BootupState String
Caption String
ChassisBootupState Int
ChassisSKUNumber String
CollectionTime Datetime
Computer String
ComputerFqdn String
CreationClassName String

Networking

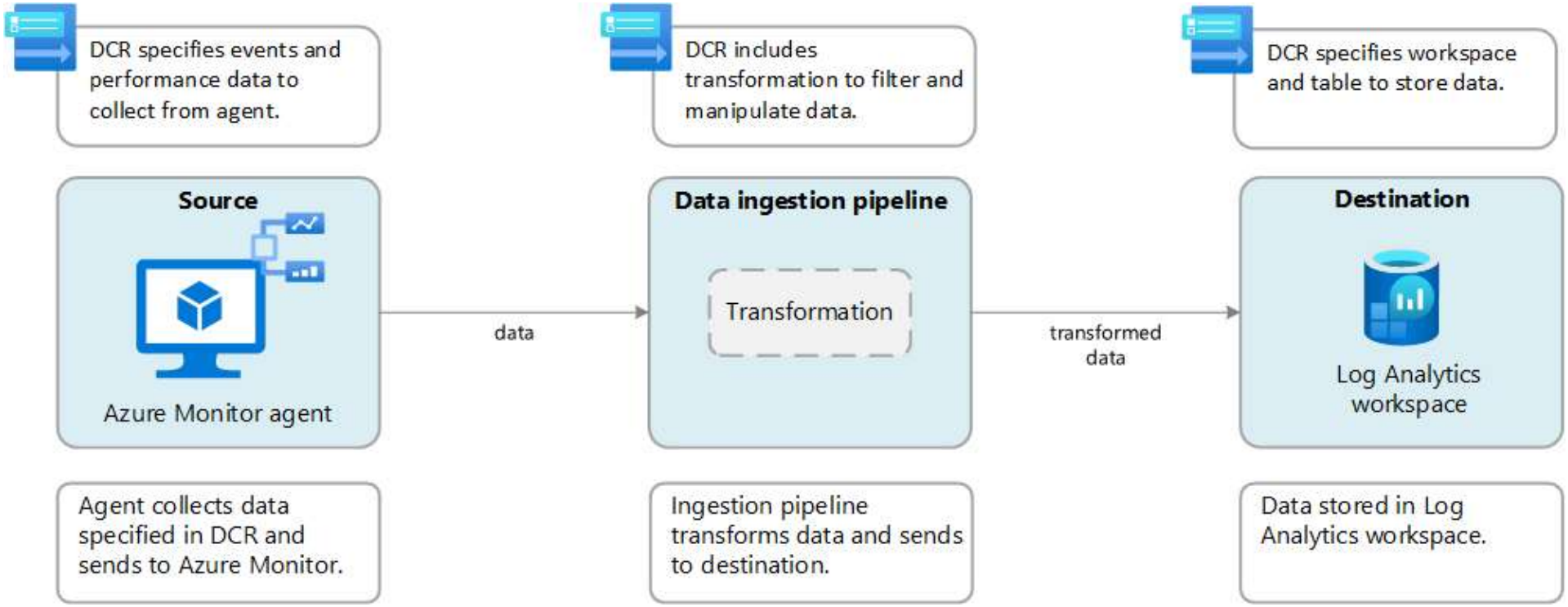
Client endpoint (REST api)
sends data to DCE



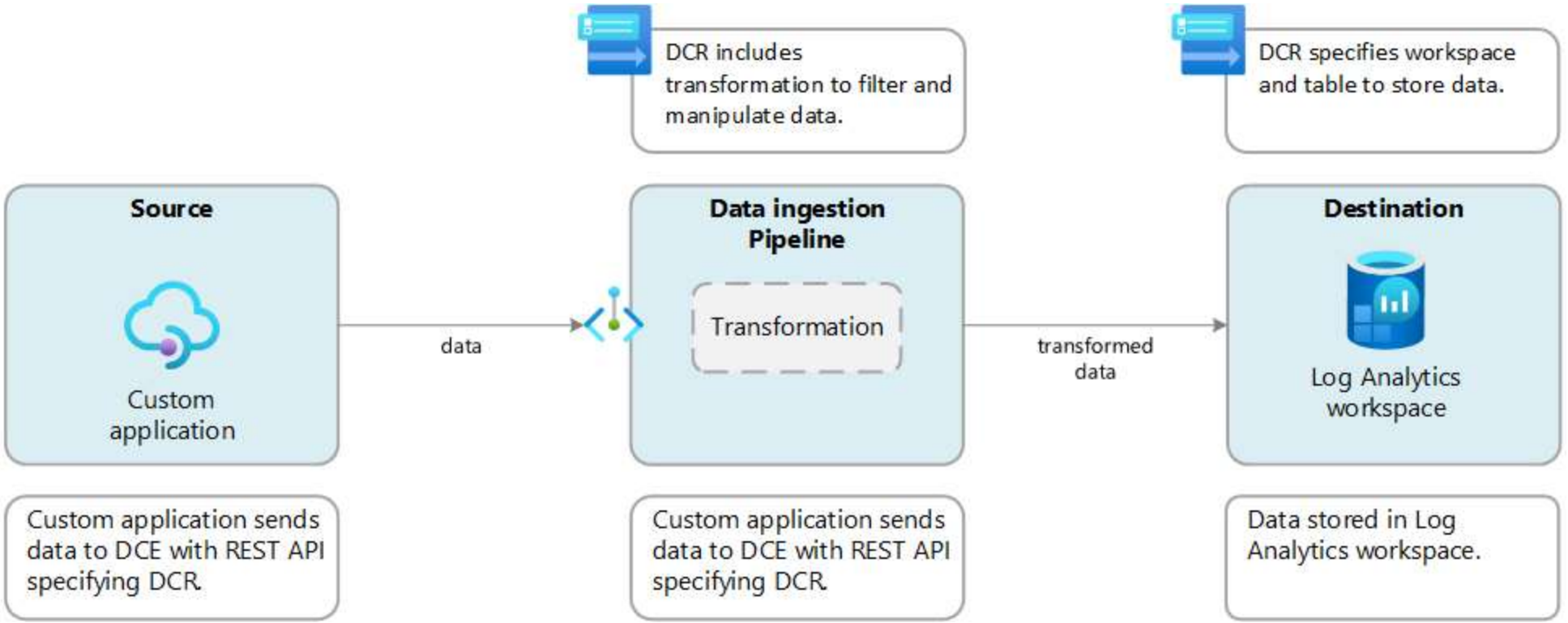


AMPLS has 2 connected resources: DCE + LA workspace
AMPLS is connected to PEP
AMPLS - Private Link Scope access modes – ingestion = open, query access = open

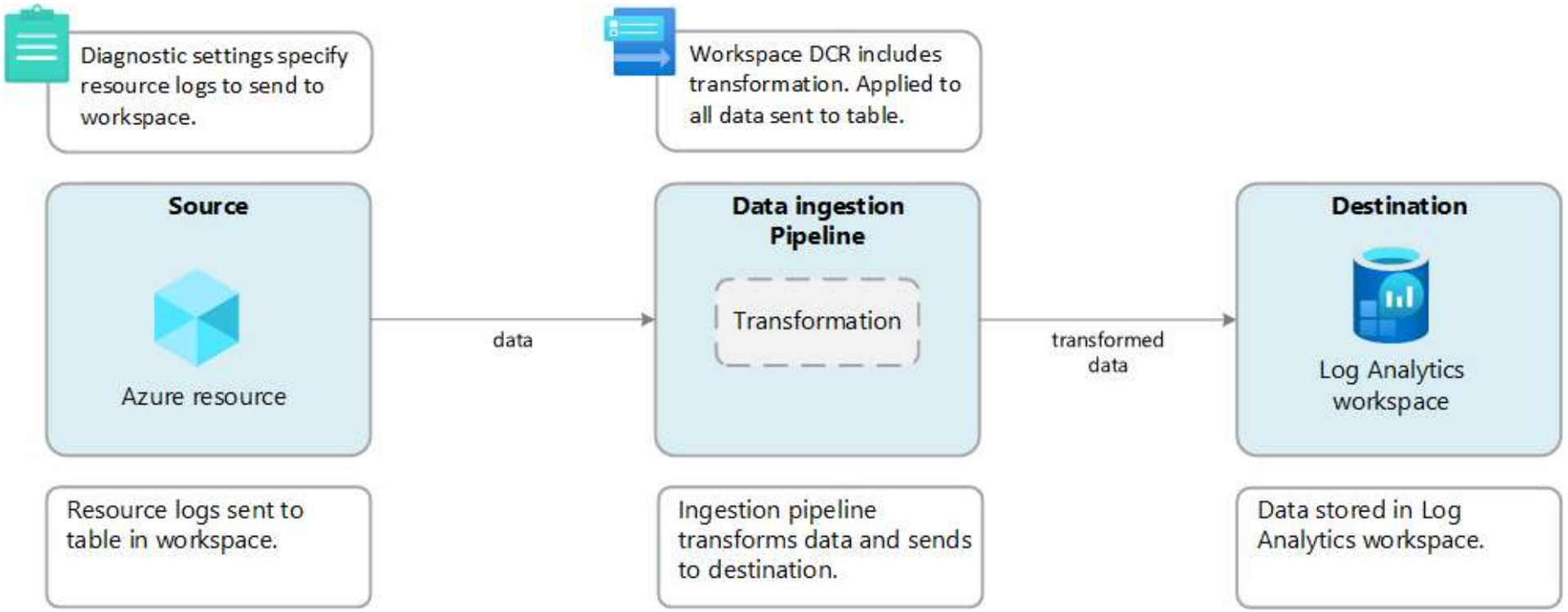
Transformation using AMA & DCR



Transformation using Log ingestion API, DCR & DCE



Workspace Transformation DCR



Collection Source	Technologies needed	Resource Association	Target (today)
Performance Eventlog Syslog	AMA / DCR	Required (DCR)	Standard table (LogAnalytics)
Text logs IIS logs Windows Firewall logs (preview)	AMA / DCR / DCE	Required (DCR)	Custom Log table (LogAnalytics)
SNMP traps	Linux with SNMP trap receiver -and- AMA (syslog file) / DCR - or - AMA (syslog stream) / DCR	Required (DCR)	(LogAnalytics)
Change Tracking (legacy	Change Tracking extension (FIM) / DCR	Required (DCR)	Standard table (LogAnalytics)
Custom logs (Log Ingest API)	DCR / DCE	N/A	Custom Log table (LogAnalytics)
Standard logs (Log Ingest API) (*)	DCR / DCE	N/A	Standard table (LogAnalytics)
Standard/Platform Metrics/Telemetry (Azure Paas)	DCR (built-in, non-manageable, hidden)	N/A	Azure Monitor Metrics
Custom Metrics/Telemetry (custom app)	AMA / DCR -or – Azure Diagnostics extension -or- Azure Monitor REST API -or- Linux: InfluxData Telegraf agent (Linux) + Azure Monitor output plugin	Required (DCR)	Azure Monitor Metrics
Platform Logs (diagnostics per resource) • AllMetrics (don't send to LA – you already have data via Azure Monitor Metrics) • Resource logs (alllogs, audit)	Azure Policy (Diagnostics) DCR (built-in, non-manageable, hidden)	Required (Policy)	Standard table (LogAnalytics)
Activity logs (audit per subscription)	Azure Policy (Diagnostics) DCR (built-in, non-manageable, hidden)		Standard table (LogAnalytics)

Collection Source	Technologies needed	Flow
Performance Eventlog Syslog	AMA / DCR	AMA -> DCR Ingestion Pipeline (backend) → LA
Text logs IIS logs Windows Firewall logs (preview)	AMA / DCR / DCE	AMA -> DCE -> DCR Ingestion Pipeline (backend) → LA
SNMP traps	Linux with SNMP trap receiver -and- AMA (syslog file) / DCR - or - AMA (syslog stream) / DCR	AMA-> DCR Ingestion Pipeline (backend) → LA
Change Tracking (legacy	Change Tracking extension (FIM) / DCR	FIM -> DCR Ingestion Pipeline (backend) → LA
Custom logs (Log Ingest API)	DCR / DCE	REST EndPoint -> DCE -> DCR Ingestion Pipeline (backend) → LA
Standard logs (Log Ingest API) (*)	DCR / DCE	REST EndPoint -> DCE -> DCR Ingestion Pipeline (backend) → LA
Platform (Standard) Metrics/Telemetry (Azure Paas)	DCR (built-in, non-manageable)	Azure Resource -> DCR Ingestion Pipeline (backend) -> Azure Monitor Metrics
Custom Metrics/Telemetry (custom app)	AMA / DCR -or – Azure Diagnostics extension -or- Azure Monitor REST API -or- Linux: InfluxData Telegraf agent (Linux) + Azure Monitor output plugin	AMA -> DCR Ingestion Pipeline (backend) → LA AzDiag Ext -> LA REST EndPoint -> DCE -> DCR Ingestion Pipeline (backend) → LA InfuxData Telegraf -> Azure Monitor output plugin -> LA
Platform Logs (diagnostics per resource) • AllMetrics • Resource logs (alllogs, audit)	Azure Policy (Diagnostics) DCR	Azure Resource -> DCR -> LA
Activity logs (audit per subscription)	Azure Policy (Diagnostics) DCR	Azure Resource -> DCR -> LA