```
# 📋 OpenPipeline Fundamentals

> **Series:** OPLOGS | **Notebook:** 1 of 8 | **Created:** December 2025

## Understanding the Unified Data Ingestion Framework

This notebook introduces OpenPipeline, Dynatrace's unified data processing
framework for logs, traces, metrics, and events.

---

## Table of Contents

1. What is OpenPipeline?
2. OpenPipeline Architecture
3. Exploring Your OpenPipeline Data
4. Key OpenPipeline Fields
5. Data Sources Explained
6. Pipeline Stages Overview
7. Environment Summary


## Prerequisites

- ✅ Access to a Dynatrace environment with log data
- ✅ DQL query permissions (viewer role minimum)
- ✅ Basic understanding of log management concepts



## 1. What is OpenPipeline?

**OpenPipeline** is Dynatrace's unified data ingestion and processing
framework that replaces classic log ingestion. It provides:

- **Unified Processing**: Single framework for logs, metrics, traces, and
business events
- **Real-time Transformation**: Parse, enrich, mask, and route data at
ingestion
- **Grail Storage**: Direct integration with Dynatrace's data lakehouse
- **Flexible Routing**: Send data to different buckets with custom retention
- **Cost Control**: Drop unnecessary data before storage

### OpenPipeline vs Classic Log Ingestion

| Feature | Classic Logs | OpenPipeline v2.0 |
|---------|--------------|-------------------|
```

Data Processing	Post-ingestion	At ingestion time
Storage	Log Storage v1	Grail Data Lakehouse
Query Language	Limited	Full DQL Support
Retention	Global	Per-bucket configurable
Data Masking	Limited	Full regex support
Parsing	Basic	DPL (Dynatrace Pattern Language)
Custom Routing	No	Yes, by content/source

2. OpenPipeline Architecture

![OpenPipeline Architecture]
(
ciIHZpZXdCb3g9IjAgMCA4MDAgMzQwIj4KICA8ZGVmcz4KICAgIDxsaW5lYXJHcmFkaWVudCBpZD0
ic291cmNlR3JhZCIgeDE9IjAlIiB5MT0iMCUiIHgyPSIxMDAlIiB5Mj0iMTAwJSI+CiAgICAgIDxz
dG9wIG9mZnNldD0iMCUiIHN0eWxlPSJzdG9wLWNvbG9yOiMzYjgyZjY7c3RvcC1vcGFjaXR5OjEiI
C8+CiAgICAgIDxzdG9wIG9mZnNldD0iMTAwJSIgc3R5bGU9InN0b3AtY29sb3I6IzI1NjNlYjtzdG
9wLW9wYWNpdHk6MSIgLz4KICAgIDwvbGluZWFyR3JhZGllbnQ+CiAgICA8bGluZWFyR3JhZGllbnQ
gaWQ9InJvdXRlR3JhZCIgeDE9IjAlIiB5MT0iMCUiIHgyPSIxMDAlIiB5Mj0iMTAwJSI+CiAgICAg
IDxzdG9wIG9mZnNldD0iMCUiIHN0eWxlPSJzdG9wLWNvbG9yOiM4YjVjZjY7c3RvcC1vcGFjaXR5O
jEiIC8+CiAgICAgIDxzdG9wIG9mZnNldD0iMTAwJSIgc3R5bGU9InN0b3AtY29sb3I6IzdjM2FlZD
tzdG9wLW9wYWNpdHk6MSIgLz4KICAgIDwvbGluZWFyR3JhZGllbnQ+CiAgICA8bGluZWFyR3JhZGl
lbnQgaWQ9Im1hc2tHcmFkIiB4MT0iMCUiIHkxPSIwJSIgeDI9IjEwMCUiIHkyPSIxMDAlIj4KICAg
ICAgPHN0b3Agb2Zmc2V0PSIwJSIgc3R5bGU9InN0b3AtY29sb3I6I2VmNDQ0NDtzdG9wLW9wYWNpd
Hk6MSIgLz4KICAgICAgPHN0b3Agb2Zmc2V0PSIxMDAlIiBzdHlsZT0ic3RvcC1jb2xvcjojZGMyNj
I2O3N0b3Atb3BhY2l0eToxIiAvPgogICAgPC9saW5lYXJHcmFkaWVudD4KICAgIDxsaW5lYXJHcmF
kaWVudCBpZD0iZmlsdGVyR3JhZCIgeDE9IjAlIiB5MT0iMCUiIHgyPSIxMDAlIiB5Mj0iMTAwJSI+
CiAgICAgIDxzdG9wIG9mZnNldD0iMCUiIHN0eWxlPSJzdG9wLWNvbG9yOiNmNTllMGI7c3RvcC1vc
GFjaXR5OjEiIC8+CiAgICAgIDxzdG9wIG9mZnNldD0iMTAwJSIgc3R5bGU9InN0b3AtY29sb3I6I2
Q5NzcwNjtzdG9wLW9wYWNpdHk6MSIgLz4KICAgIDwvbGluZWFyR3JhZGllbnQ+CiAgICA8bGluZWF
yR3JhZGllbnQgaWQ9InByb2Nlc3NHcmFkIiB4MT0iMCUiIHkxPSIwJSIgeDI9IjEwMCUiIHkyPSIx
MDAlIj4KICAgICAgPHN0b3Agb2Zmc2V0PSIwJSIgc3R5bGU9InN0b3AtY29sb3I6IzEwYjk4MTtzd
G9wLW9wYWNpdHk6MSIgLz4KICAgICAgPHN0b3Agb2Zmc2V0PSIxMDAlIiBzdHlsZT0ic3RvcC1jb2
xvcjojMDU5NjY5O3N0b3Atb3BhY2l0eToxIiAvPgogICAgPC9saW5lYXJHcmFkaWVudD4KICAgIDx
saW5lYXJHcmFkaWVudCBpZD0iZXh0cmFjdEdyYWQiIHgxPSIwJSIgeTE9IjAlIiB4Mj0iMTAwJSIg
eTI9IjEwMCUiPgogICAgICA8c3RvcCBvZmZzZXQ9IjAlIiBzdHlsZT0ic3RvcC1jb2xvcjojMDZiN
mQ0O3N0b3Atb3BhY2l0eToxIiAvPgogICAgICA8c3RvcCBvZmZzZXQ9IjEwMCUiIHN0eWxlPSJzdG
9wLWNvbG9yOiMwODkxYjI7c3RvcC1vcGFjaXR5OjEiIC8+CiAgICA8L2xpbmVhckdyYWRpZW50Pgo
gICAgPGxpbmVhckdyYWRpZW50IGlkPSJzdG9yZUdyYWQiIHgxPSIwJSIgeTE9IjAlIiB4Mj0iMTAw
JSIgeTI9IjEwMCUiPgogICAgICA8c3RvcCBvZmZzZXQ9IjAlIiBzdHlsZT0ic3RvcC1jb2xvcjojZ
WM0ODk5O3N0b3Atb3BhY2l0eToxIiAvPgogICAgICA8c3RvcCBvZmZzZXQ9IjEwMCUiIHN0eWxlPS
JzdG9wLWNvbG9yOiNkYjI3Nzc7c3RvcC1vcGFjaXR5OjEiIC8+CiAgICA8L2xpbmVhckdyYWRpZW5
0PgogICAgPGZpbHRlciBpZD0ib3BTaGFkb3ciPgogICAgICA8ZmVEcm9wU2hhZG93IGR4PSIxIiBk
eT0iMSIgc3RkRGV2aWF0aW9uPSIyIiBmbG9vZC1vcGFjaXR5PSIwLjE1Ii8+CiAgICA8L2ZpbHRlc
j4KICAgIDxtYXJrZXIgaWQ9Im9wQXJyb3ciIG1hcmtlcldpZHRoPSIxMCIgbWFya2VySGVpZ2h0PS
I3IiByZWZYPSI5IiByZWZZPSIzLjUiIG9yaWVudD0iYXV0byI+CiAgICAgIDxwb2x5Z29uIHBvaW5
0cz0iMCAwLCAxMCAzLjUsIDAgNyIgZmlsbD0iIzY0NzQ4YiIvPgogICAgPC9tYXJrZXI+CiAgPC9k
ZWZzPgoKICA8IS0tIEJhY2tncm91bmQgLS0+CiAgPHJlY3Qgd2lkdGg9IjgwMCIgaGVpZ2h0PSIzN
DAiIGZpbGw9IiNmOGY5ZmEiIHJ4PSIxMCIvPgoKICA8IS0tIFRpdGxlIC0tPgogIDx0ZXh0IHg9Ij

QwMCIgeT0iMjgiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxOCI
gZm9udC13ZWlnaHQ9ImJvbGQiIGZpbGw9IiMzMzMiIHRleHQtYW5jaG9yPSJtaWRkbGUiPk9wZW5Q
aXBlbGluZSBMb2cgUHJvY2Vzc2luZyBBcmNoaXRlY3R1cmU8L3RleHQ+CgogIDwhLS0gRGF0YSBTb
3VyY2VzIFJvdyAtLT4KICA8dGV4dCB4PSIxMDAiIHk9IjYwIiBmb250LWZhbWlseT0iQXJpYWwsIH
NhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTEiIGZvbnQtd2VpZ2h0PSJib2xkIiBmaWxsPSIjMzMzIiB
0ZXh0LWFuY2hvcj0ibWlkZGxlIj5EYXRhIFNvdXJjZXM8L3RleHQ+CgogIDxyZWN0IHg9IjMwIiB5
PSI3MCIgd2lkdGg9IjYwIiBoZWlnaHQ9IjQwIiByeD0iNiIgZmlsbD0idXJsKCNzb3VyY2VHcmFkK
SIgZmlsdGVyPSJ1cmwoI29wU2hhZG93KSIvPgogIDx0ZXh0IHg9IjYwIiB5PSI5NSIgZm9udC1mYW
1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSJ3aGl0ZSIgdGV4dC1
hbmNob3I9Im1pZGRsZSI+T25lQWdlbnQ8L3RleHQ+CgogIDxyZWN0IHg9IjEwMCIgeT0iNzAiIHdp
ZHRoPSI2MCIgaGVpZ2h0PSI0MCIgcng9IjYiIGZpbGw9InVybCgjc291cmNlR3JhZCkiIGZpbHRlc
j0idXJsKCNvcFNoYWRvdykiLz4KICA8dGV4dCB4PSIxMzAiIHk9Ijk1IiBmb250LWZhbWlseT0iQX
JpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IndoaXRlIiB0ZXh0LWFuY2hvcj0
ibWlkZGxlIj5Mb2cgQVBJPC90ZXh0PgoKICA8cmVjdCB4PSIzMCIgeT0iMTE1IiB3aWR0aD0iNjAi
IGhlaWdodD0iNDAiIHJ4PSI2IiBmaWxsPSJ1cmwoI3NvdXJjZUdyYWQpIiBmaWx0ZXI9InVybCgjb
3BTaGFkb3cpIi8+CiAgPHRleHQgeD0iNjAiIHk9IjE0MCIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW
5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSJ3aGl0ZSIgdGV4dC1hbmNob3I9Im1pZGRsZSI
+T1RMUDwvdGV4dD4KCiAgPHJlY3QgeD0iMTAwIiB5PSIxMTUiIHdpZHRoPSI2MCIgaGVpZ2h0PSI0
MCIgcng9IjYiIGZpbGw9InVybCgjc291cmNlR3JhZCkiIGZpbHRlcj0idXJsKCNvcFNoYWRvdykiL
z4KICA8dGV4dCB4PSIxMzAiIHk9IjE0MCIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIi
Bmb250LXNpemU9IjEwIiBmaWxsPSJ3aGl0ZSIgdGV4dC1hbmNob3I9Im1pZGRsZSI+R2VuZXJpYzw
vdGV4dD4KCiAgPCEtLSBBcnJvdyBmcm9tIHNvdXJjZXMgLS0+CiAgPHBhdGggZD0iTTE2NSwxMTIg
TDE5NSwxMTIiIHN0cm9rZT0iIzY0NzQ4YiIgc3Ryb2tlLXdpZHRoPSIyIiBmaWxsPSJub25lIiBtY
XJrZXItZW5kPSJ1cmwoI29wQXJyb3cpIi8+CgogIDwhLS0gUHJvY2Vzc2luZyBTdGdnZXMgLS0+Ci
AgPHJlY3QgeD0iMjA1IiB5PSI3NSIgd2lkdGg9Ijc1IiBoZWlnaHQ9Ijc1IiByeD0iOCIgZmlsbD0
idXJsKCNyb3V0ZUdyYWQpIiBmaWx0ZXI9InVybCgjb3BTaGFkb3cpIi8+CiAgPHRleHQgeD0iMjQy
IiB5PSIxMDAiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMSIgZ
m9udC13ZWlnaHQ9ImJvbGQiIGZpbGw9IndoaXRlIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5ST1VURT
wvdGV4dD4KICA8dGV4dCB4PSIyNDIiIHk9IjExOCIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXN
lcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSJyZ2JhKDI1NSwyNTUsMjU1LDAuOSkiIHRleHQtYW5j
aG9yPSJtaWRkbGUiPlBpcGVsaW5lPC90ZXh0PgogIDx0ZXh0IHg9IjI0MiIgeT0iMTMzIiBmb250L
WZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9InJnYmEoMjU1LD
I1NSwyNTUsMC45KSIgdGV4dC1hbmNob3I9Im1pZGRsZSI+U2VsZWN0aW9uPC90ZXh0PgoKICA8cGF
0aCBkPSJNMjgwLDExMiBMMzAwLDExMiIgc3Ryb2tlPSIjNjQ3NDhiIiBzdHJva2Utd2lkdGg9IjIi
IGZpbGw9Im5vbmUiIG1hcmtlci1lbmQ9InVybCgjb3BBcnJvdykiLz4KCiAgPHJlY3QgeD0iMzEwI
iB5PSI3NSIgd2lkdGg9Ijc1IiBoZWlnaHQ9Ijc1IiByeD0iOCIgZmlsbD0idXJsKCNtYXNrR3JhZC
kiIGZpbHRlcj0idXJsKCNvcFNoYWRvdykiLz4KICA8dGV4dCB4PSIzNDciIHk9IjEwMCIgZm9udC1
mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjExIiBmb250LXdlaWdodD0iYm9s
ZCIgZmlsbD0id2hpdGUiIHRleHQtYW5jaG9yPSJtaWRkbGUiPk1BU0s8L3RleHQ+CiAgPHRleHQge
D0iMzQ3IiB5PSIxMTgiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPS
IxMCIgZmlsbD0icmdiYSgyNTUsMjU1LDI1NSwwLjkpIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5QSUk
vUEhJPC90ZXh0PgogIDx0ZXh0IHg9IjM0NyIgeT0iMTMzIiBmb250LWZhbWlseT0iQXJpYWwsIHNh
bnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9InJnYmEoMjU1LDI1NSwyNTUsMC45KSIgdGV4d
C1hbmNob3I9Im1pZGRsZSI+UmVkYWN0aW9uPC90ZXh0PgoKICA8cGF0aCBkPSJNMzg1LDExMiBMND
A1LDExMiIgc3Ryb2tlPSIjNjQ3NDhiIiBzdHJva2Utd2lkdGg9IjIiIGZpbGw9Im5vbmUiIG1hcmt
lci1lbmQ9InVybCgjb3BBcnJvdykiLz4KCiAgPHJlY3QgeD0iNDE1IiB5PSI3NSIgd2lkdGg9Ijc1
IiBoZWlnaHQ9Ijc1IiByeD0iOCIgZmlsbD0idXJsKCNmaWx0ZXJHcmFkKSIgZmlsdGVyPSJ1cmwoI
29wU2hhZG93KSIvPgogIDx0ZXh0IHg9IjQ1MiIgeT0iMTAwIiBmb250LWZhbWlseT0iQXJpYWwsIH

NhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTEiIGZvbnQtd2VpZ2h0PSJib2xkIiBmaWxsPSJ3aGl0ZSI
gdGV4dC1hbmNob3I9Im1pZGRsZSI+RklMVEVSPC90ZXh0PgogIDx0ZXh0IHg9IjQ1MiIgeT0iMTE4
IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9InJnY
mEoMjU1LDI1NSwyNTUsMC45KSIgdGV4dC1hbmNob3I9Im1pZGRsZSI+RHJvcDwvdGV4dD4KICA8dG
V4dCB4PSI0NTIiIHk9IjEzMyIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXN
pemU9IjEwIiBmaWxsPSJyZ2JhKDI1NSwyNTUsMjU1LDAuOSkiIHRleHQtYW5jaG9yPSJtaWRkbGUi
PlVud2FudGVkPC90ZXh0PgoKICA8cGF0aCBkPSJNNDkwLDExMiBMNTEwLDExMiIgc3Ryb2tlPSIjN
jQ3NDhiIiBzdHJva2Utd2lkdGg9IjIiIGZpbGw9Im5vbmUiIG1hcmtlci1lbmQ9InVybCgjb3BBcn
JvdykiLz4KCiAgPHJlY3QgeD0iNTIwIiB5PSI3NSIgd2lkdGg9Ijc1IiBoZWlnaHQ9Ijc1IiByeD0
iOCIgZmlsbD0idXJsKCNwcm9jZXNzR3JhZCkiIGZpbHRlcj0idXJsKCNvcFNoYWRvdykiLz4KICA8
dGV4dCB4PSI1NTciIHk9IjEwMCIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250L
XNpemU9IjExIiBmb250LXdlaWdodD0iYm9sZCIgZmlsbD0id2hpdGUiIHRleHQtYW5jaG9yPSJtaW
RkbGUiPlBST0NFU1M8L3RleHQ+CiAgPHRleHQgeD0iNTU3IiB5PSIxMTgiIGZvbnQtZmFtaWx5PSJ
BcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0icmdiYSgyNTUsMjU1LDI1NSww
LjkpIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5QYXJzZTwvdGV4dD4KICA8dGV4dCB4PSI1NTciIHk9I
jEzMyIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPS
JyZ2JhKDI1NSwyNTUsMjU1LDAuOSkiIHRleHQtYW5jaG9yPSJtaWRkbGUiPlRyYW5zZm9ybTwvdGV
4dD4KCiAgPHBhdGggZD0iTTU5NSwxMTIgTDYxNSwxMTIiIHN0cm9rZT0iIzY0NzQ4YiIgc3Ryb2tl
LXdpZHRoPSIyIiBmaWxsPSJub25lIiBtYXJrZXItZW5kPSJ1cmwoI29wQXJyb3cpIi8+CgogIDxyZ
WN0IHg9IjYyNSIgeT0iNzUiIHdpZHRoPSI3NSIgaGVpZ2h0PSI3NSIgcng9IjgiIGZpbGw9InVybC
gjZXh0cmFjdEdyYWQpIiBmaWx0ZXI9InVybCgjb3BTaGFkb3cpIi8+CiAgPHRleHQgeD0iNjYyIiB
5PSIxMDAiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMSIgZm9u
dC13ZWlnaHQ9ImJvbGQiIGZpbGw9IndoaXRlIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5FWFRSQUNUP
C90ZXh0PgogIDx0ZXh0IHg9IjY2MiIgeT0iMTE4IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2
VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9InJnYmEoMjU1LDI1NSwyNTUsMC45KSIgdGV4dC1hbmN
ob3I9Im1pZGRsZSI+TWV0cmljczwvdGV4dD4KICA8dGV4dCB4PSI2NjIiIHk9IjEzMyIgZm9udC1m
YW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSJyZ2JhKDI1NSwyN
TUsMjU1LDAuOSkiIHRleHQtYW5jaG9yPSJtaWRkbGUiPkV2ZW50czwvdGV4dD4KCiAgPHBhdGggZD
0iTTcwMCwxMTIgTDcyMCwxMTIiIHN0cm9rZT0iIzY0NzQ4YiIgc3Ryb2tlLXdpZHRoPSIyIiBmaWx
sPSJub25lIiBtYXJrZXItZW5kPSJ1cmwoI29wQXJyb3cpIi8+CgogIDxyZWN0IHg9IjczMCIgeT0i
NzUiIHdpZHRoPSI1MCIgaGVpZ2h0PSI3NSIgcng9IjgiIGZpbGw9InVybCgjc3RvcmVHcmFkKSIgZ
mlsdGVyPSJ1cmwoI29wU2hhZG93KSIvPgogIDx0ZXh0IHg9Ijc1NSIgeT0iMTA1IiBmb250LWZhbW
lseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTEiIGZvbnQtd2VpZ2h0PSJib2xkIiB
maWxsPSJ3aGl0ZSIgdGV4dC1hbmNob3I9Im1pZGRsZSI+U1RPUkU8L3RleHQ+CiAgPHRleHQgeD0i
NzU1IiB5PSIxMjUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxM
CIgZmlsbD0icmdiYSgyNTUsMjU1LDI1NSwwLjkpIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5HcmFpbD
wvdGV4dD4KCiAgPCEtLSBLZXkgRmVhdHVyZXMgU2VjdGlvbiAtLT4KICA8cmVjdCB4PSIzMCIgeT0
iMTcwIiB3aWR0aD0iNzQwIiBoZWlnaHQ9IjE1NSIgcng9IjEwIiBmaWxsPSIjZmZmIiBzdHJva2U9
IiNlMmU4ZjAiIHN0cm9rZS13aWR0aD0iMiIvPgogIDx0ZXh0IHg9IjQwMCIgeT0iMTk1IiBmb250L
WZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTIiIGZvbnQtd2VpZ2h0PSJib2
xkIiBmaWxsPSIjMzMzIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5LZXkgQ2FwYWJpbGl0aWVzPC90ZXh
0PgoKICA8IS0tIEZlYXR1cmUgYm94ZXMgLS0+CiAgPHJlY3QgeD0iNTAiIHk9IjIxMCIgd2lkdGg9
IjE1NSIgaGVpZ2h0PSIxMDAiIHJ4PSI2IiBmaWxsPSIjZmVmMmYyIi8+CiAgPHRleHQgeD0iMTI3I
iB5PSIyMzUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMSIgZm
9udC13ZWlnaHQ9ImJvbGQiIGZpbGw9IiNkYzI2MjYiIHRleHQtYW5jaG9yPSJtaWRkbGUiPlNlY3V
yaXR5IEZpcnN0PC90ZXh0PgogIDx0ZXh0IHg9IjEyNyIgeT0iMjU1IiBmb250LWZhbWlseT0iQXJp
YWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM3ZjFkMWQiIHRleHQtYW5jaG9yP
SJtaWRkbGUiPk1hc2tpbmcgcnVucyBCRUZPUkU8L3RleHQ+CiAgPHRleHQgeD0iMTI3IiB5PSIyNz

AiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzd
mMWQxZCIgdGV4dC1hbmNob3I9Im1pZGRsZSI+YW55IG90aGVyIHByb2Nlc3Npbmc8L3RleHQ+CiAg
PHRleHQgeD0iMTI3IiB5PSIyOTUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9ud
C1zaXplPSIxMCIgZmlsbD0iIzdmMWQxZCIgdGV4dC1hbmNob3I9Im1pZGRsZSI+UElJIG5ldmVyIH
N0b3JlZDwvdGV4dD4KCiAgPHJlY3QgeD0iMjI1IiB5PSIyMTAiIHdpZHRoPSIxNTUiIGhlaWdodD0
iMTAwIiByeD0iNiIgZmlsbD0iI2ZlZjNjNyIvPgogIDx0ZXh0IHg9IjMwMiIgeT0iMjM1IiBmb250
LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTEiIGZvbnQtd2VpZ2h0PSJib
2xkIiBmaWxsPSIjOTI0MDBlIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5Db3N0IENvbnRyb2w8L3RleH
Q+CiAgPHRleHQgeD0iMzAyIiB5PSIyNTUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiI
gZm9udC1zaXplPSIxMCIgZmlsbD0iIzc4MzUwZiIgdGV4dC1hbmNob3I9Im1pZGRsZSI+RHJvcCB1
bndhbnRlZCBsb2dzPC90ZXh0PgogIDx0ZXh0IHg9IjMwMiIgeT0iMjcwIiBmb250LWZhbWlseT0iQ
XJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM3ODM1MGYiIHRleHQtYW5jaG
9yPSJtaWRkbGUiPkJFRk9SRSBzdG9yYWdlPC90ZXh0PgogIDx0ZXh0IHg9IjMwMiIgeT0iMjk1IiB
mb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM3ODM1
MGYiIHRleHQtYW5jaG9yPSJtaWRkbGUiPlNhdmUgMzAtNzAlIEREVXM8L3RleHQ+CgogIDxyZWN0I
Hg9IjQwMCIgeT0iMjEwIiB3aWR0aD0iMTU1IiBoZWlnaHQ9IjEwMCIgcng9IjYiIGZpbGw9IiNkMW
ZhZTUiLz4KICA8dGV4dCB4PSI0NzciIHk9IjIzNSIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXN
lcmlmIiBmb250LXNpemU9IjExIiBmb250LXdlaWdodD0iYm9sZCIgZmlsbD0iIzA0Nzg1NyIgdGV4
dC1hbmNob3I9Im1pZGRsZSI+VmFsdWUgQ3JlYXRpb248L3RleHQ+CiAgPHRleHQgeD0iNDc3IiB5P
SIyNTUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD
0iIzA2NGUzYiIgdGV4dC1hbmNob3I9Im1pZGRsZSI+RXh0cmFjdCBtZXRyaWNzPC90ZXh0PgogIDx
0ZXh0IHg9IjQ3NyIgeT0iMjcwIiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQt
c2l6ZT0iMTAiIGZpbGw9IiMwNjRlM2IiIHRleHQtYW5jaG9yPSJtaWRkbGUiPkdlbmVyYXRlIGV2Z
W50czwvdGV4dD4KICA8dGV4dCB4PSI0NzciIHk9IjI5NSIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW
5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSIjMDY0ZTNiIiB0ZXh0LWFuY2hvcj0ibWlkZGx
lIj5CdXNpbmVzcyBhbmFseXRpY3M8L3RleHQ+CgogIDxyZWN0IHg9IjU3NSIgeT0iMjEwIiB3aWR0
aD0iMTc1IiBoZWlnaHQ9IjEwMCIgcng9IjYiIGZpbGw9IiNkYmVhZmUiLz4KICA8dGV4dCB4PSI2N
jIiIHk9IjIzNSIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjExIi
Bmb250LXdlaWdodD0iYm9sZCIgZmlsbD0iIzFlNDBhZiIgdGV4dC1hbmNob3I9Im1pZGRsZSI+QnV
ja2V0IEdvdmVybmFuY2U8L3RleHQ+CiAgPHRleHQgeD0iNjYyIiB5PSIyNTUiIGZvbnQtZmFtaWx5
PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzFlM2E4YSIgdGV4dC1hb
mNob3I9Im1pZGRsZSI+Um91dGUgdG8gYnVja2V0czwvdGV4dD4KICA8dGV4dCB4PSI2NjIiIHk9Ij
I3MCIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSI
jMWUzYThhIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5UaWVyZWQgcmV0ZW50aW9uPC90ZXh0PgogIDx0
ZXh0IHg9IjY2MiIgeT0iMjk1IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc
2l6ZT0iMTAiIGZpbGw9IiMxZTNhOGEiIHRleHQtYW5jaG9yPSJtaWRkbGUiPlRlYW0tYmFzZWQgYW
NjZXNzPC90ZXh0Pgo8L3N2Zz4K)

3. Exploring Your OpenPipeline Data

Let's start by discovering what data sources and pipelines are active in your environment.

```python
// Discover data sources feeding OpenPipeline
fetch logs, from: now() - 1h
```

```
| summarize {log_count = count()}, by: {dt.openpipeline.source}
| sort log_count desc
```

```python
// See which pipelines are processing your logs
fetch logs, from: now() - 1h
| summarize {log_count = count()}, by: {dt.openpipeline.pipelines}
| sort log_count desc
```

```python
// Check storage bucket distribution
fetch logs, from: now() - 1h
| summarize {log_count = count()}, by: {dt.system.bucket}
| sort log_count desc
```

4. Key OpenPipeline Fields

OpenPipeline adds metadata fields to every log record:

Pipeline Metadata

Field	Description	Example
`dt.openpipeline.source`	How the log was ingested	`oneagent`, `/api/v2/logs/ingest`, `/api/v2/otlp/v1/logs`
`dt.openpipeline.pipelines`	Pipeline(s) that processed the log	`["logs:pipeline_Default_Pipeline_2798"]`
`dt.system.bucket`	Storage bucket name	`default_logs`, `custom_logs`

Core Log Fields

Field	Description
`timestamp`	When the log was generated
`content`	The log message body
`loglevel`	Log severity (ERROR, WARN, INFO, DEBUG, NONE)
`status`	Status string (alternative to loglevel)
`log.source`	Source identifier (e.g., "Container Output")
`log.iostream`	Stream type (stdout, stderr)

```python
// View a sample log record with all key fields
fetch logs, from: now() - 1h
| fields timestamp, content, loglevel, status, log.source, log.iostream,
         dt.openpipeline.source, dt.openpipeline.pipelines, dt.system.bucket
```

```
| limit 5
```

```python
// Analyze log levels in your environment
fetch logs, from: now() - 1h
| summarize {count = count()}, by: {loglevel}
| sort count desc
```

5. Data Sources Explained

OneAgent (`oneagent`)
Logs collected automatically by Dynatrace OneAgent from:
- Container stdout/stderr
- Process log files
- System logs

Log Ingest API (`/api/v2/logs/ingest`)
Logs sent directly via the Dynatrace API:
- Custom application logs
- Third-party integrations
- Cloud provider logs (AWS, Azure, GCP)

OTLP (`/api/v2/otlp/v1/logs`)
OpenTelemetry Protocol logs:
- OpenTelemetry Collector
- Fluent Bit with OTLP output
- Custom OTLP exporters

```python
// Compare volume by data source
fetch logs, from: now() - 24h
| summarize {
    log_count = count(),
    unique_hosts = countDistinct(dt.entity.host)
  }, by: {dt.openpipeline.source}
| sort log_count desc
```

```python
// Logs per hour by source (trend analysis)
fetch logs, from: now() - 24h
| makeTimeseries {log_count = count()}, by: {dt.openpipeline.source},
interval: 1h
```

6. Pipeline Stages Overview

````
OpenPipeline processes data through ordered stages:

### Stage 1: Routing
- Matches incoming data to the appropriate pipeline
- Based on source, content, or metadata

### Stage 2: Masking (Security)
- Redacts sensitive data BEFORE processing
- Protects PII, credentials, secrets
- Applied early for security compliance

### Stage 3: Filtering
- Drops unwanted records
- Reduces storage costs
- Removes noise (debug logs, health checks)

### Stage 4: Processing
- Parses structured data from content
- Adds enrichment fields
- Transforms and normalizes data

### Stage 5: Extraction
- Creates metrics from log data
- Generates events and business events

### Stage 6: Storage
- Routes to appropriate Grail bucket
- Applies retention policies

## 7. Environment Summary

Let's get a complete picture of your OpenPipeline environment:

```python
// Complete environment summary
fetch logs, from: now() - 1h
| summarize {
    total_logs = count(),
    unique_sources = countDistinct(dt.openpipeline.source),
    unique_buckets = countDistinct(dt.system.bucket),
    unique_hosts = countDistinct(dt.entity.host),
    error_count = countIf(loglevel == "ERROR" OR status == "ERROR"),
    warn_count = countIf(loglevel == "WARN" OR status == "WARN")
  }
```

```python
````
````

```
// Top log sources by entity
fetch logs, from: now() - 1h
| filter isNotNull(dt.entity.host)
| summarize {log_count = count()}, by: {host.name, log.source}
| sort log_count desc
| limit 15
```

📝 Summary

In this notebook, you learned:

✅ **What OpenPipeline is** - Dynatrace's unified data processing framework
✅ **Architecture** - Data flow from sources through processing to Grail
✅ **Key fields** - `dt.openpipeline.source`, `dt.openpipeline.pipelines`, `dt.system.bucket`
✅ **Data sources** - OneAgent, Log Ingest API, OTLP
✅ **Pipeline stages** - Routing, Masking, Filtering, Processing, Extraction, Storage

---

➡️ Next Steps

Continue to **OPLOGS-02: Migration Guide** to learn how to migrate from classic log ingestion to OpenPipeline v2.0.

📚 References

- [OpenPipeline Documentation](https://docs.dynatrace.com/docs/discover-dynatrace/platform/openpipeline)
- [Grail Data Lakehouse](https://docs.dynatrace.com/docs/platform/grail)
- [DQL Reference](https://docs.dynatrace.com/docs/platform/grail/dynatrace-query-language)