```
# OpenPipeline Migration Guide: Part 1

> **Series:** OPMIG | **Notebook:** 1 of 9 | **Created:** December 2025

## Introduction & Why Migrate from Classic to OpenPipeline v2.0

---

## Learning Objectives

By the end of this notebook, you will:

- ✅ Understand what OpenPipeline is and why it replaces Classic log
ingestion
- ✅ Learn the key architectural differences between Classic and OpenPipeline
v2.0
- ✅ **Understand API endpoint compatibility and migration requirements**
- ✅ **Know OpenPipeline limits and constraints**
- ✅ Identify the benefits of migrating (cost, security, performance)
- ✅ **Review real-world migration scenarios**
- ✅ Assess your readiness to migrate

---

---

## What is OpenPipeline?

**OpenPipeline** is Dynatrace's unified data handling solution that
seamlessly ingests and processes data from different sources, at any scale,
and in any format.

> 💡 **Key Insight:** OpenPipeline is NOT just for logs! It's a comprehensive
data processing framework that handles **logs, spans, metrics, events,
business events, security events, and more**.

### OpenPipeline Architecture Overview

![OpenPipeline Architecture]
(
ciIHZpZXdCb3g9IjAgMCA4MDAgMzIwIj4KICA8ZGVmcz4KICAgIDxsaW5lYXJHcmFkaWVudCBpZD0
iaW5nZXN0R3JhZCIgeDE9IjAlIiB5MT0iMCUiHgyPSIxMDAlIiB5Mj0iMTAwJSI+CiAgICAgIDxz
dG9wIG9mZnNldD0iMCUiHN0eWxlPSJzdG9wLWNvbG9yOiMzYjgyZjY7c3RvcC1vcGFjaXR5OjEiI
C8+CiAgICAgIDxzdG9wIG9mZnNldD0iMTAwJSIgc3R5bGU9InN0b3AtY29sb3I6IzI1NjNlYjtzdG
9wLW9wYWNpdHk6MSIgLz4KICAgIDwvbGluZWFyR3JhZGllbnQ+CiAgICA8bGluZWFyR3JhZGllbnQ
gaWQ9InJvdXRlR3JhZCIgeDE9IjAlIiB5MT0iMCUiHgyPSIxMDAlIiB5Mj0iMTAwJSI+CiAgICAg
IDxzdG9wIG9mZnNldD0iMCUiHN0eWxlPSJzdG9wLWNvbG9yOiM4YjVjZjY7c3RvcC1vcGFjaXR5O

```

```
jEiIC8+CiAgICAgIDxzdG9wIG9mZnNldD0iMTAwJSIgc3R5bGU9InN0b3AtY29sb3I6IzdjM2FlZD
tzdG9wLW9wYWNpdHk6MSIgLz4KICAgIDwvbGluZWFyR3JhZGllbnQ+CiAgICA8bGluZWFyR3JhZGl
lbnQgaWQ9InByb2Nlc3NHcmFkIiB4MT0iMCUiIHkxPSIwJSIgeDI9IjEwMCUiIHkyPSIxMDAlIj4K
ICAgICAgPHN0b3Agb2Zmc2V0PSIwJSIgc3R5bGU9InN0b3AtY29sb3I6I2Y1OWUwYjtzdG9wLW9wY
WNpdHk6MSIgLz4KICAgICAgPHN0b3Agb2Zmc2V0PSIxMDAlIiBzdHlsZT0ic3RvcC1jb2xvcjojZD
k3NzA2O3N0b3Atb3BhY2l0eToxIiAvPgogICAgPC9saW5lYXJHcmFkaWVudD4KICAgIDxsaW5lYXJ
HcmFkaWVudCBpZD0iZXh0cmFjdEdyYWQiIHgxPSIwJSIgeTE9IjAlIiB4Mj0iMTAwJSIgeTI9IjEw
MCUiPgogICAgICA8c3RvcCBvZmZzZXQ9IjAlIiBzdHlsZT0ic3RvcC1jb2xvcjojMTBiOTgxO3N0b
3Atb3BhY2l0eToxIiAvPgogICAgICA8c3RvcCBvZmZzZXQ9IjEwMCUiIHN0eWxlPSJzdG9wLWNvbG
9yOiMwNTk2Njk7c3RvcC1vcGFjaXR5OjEiIC8+CiAgICA8L2xpbmVhckdyYWRpZW50PgogICAgPGx
pbmVhckdyYWRpZW50IGlkPSJzdG9yZUdyYWQiIHgxPSIwJSIgeTE9IjAlIiB4Mj0iMTAwJSIgeTI9
IjEwMCUiPgogICAgICA8c3RvcCBvZmZzZXQ9IjAlIiBzdHlsZT0ic3RvcC1jb2xvcjojZWY0NDQ0O
3N0b3Atb3BhY2l0eToxIiAvPgogICAgICA8c3RvcCBvZmZzZXQ9IjEwMCUiIHN0eWxlPSJzdG9wLW
NvbG9yOiNkYzI2MjY7c3RvcC1vcGFjaXR5OjEiIC8+CiAgICA8L2xpbmVhckdyYWRpZW50PgogICA
gPGZpbHRlciBpZD0ib3BTaGFkb3ciPgogICAgICA8ZmVEcm9wU2hhZG93IGR4PSIyIiBkeT0iMiIg
c3RkRGV2aWF0aW9uPSIzIiBmbG9vZC1vcGFjaXR5PSIwLjIiLz4KICAgIDwvZmlsdGVyPgogICAgP
G1hcmtlciBpZD0ib3BBcnJvdyIgbWFya2VyV2lkdGg9IjEwIiBtYXJrZXJIZWlnaHQ9IjciIHJlZl
g9IjkiIHJlZlk9IjMuNSIgb3JpZW50PSJhdXRvIj4KICAgICAgPHBvbHlnb24gcG9pbnRzPSIDA
sIDEwIDMuNSwgMCA3IiBmaWxsPSIjNjQ3NDhiIi8+CiAgICA8L21hcmtlcj4KICA8L2RlZnM+Cgog
IDwhLS0gQmFja2dyb3VuZCAtLT4KICA8cmVjdCB3aWR0aD0iODAwIiBoZWlnaHQ9IjMyMCIgZmlsb
D0iI2Y4ZjlmYSIgcng9IjEwIi8+CgogIDwhLS0gVGl0bGUgLS0+CiAgPHRleHQgeD0iNDAwIiB5PS
IyOCIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjE4IiBmb250LXd
laWdodD0iYm9sZCIgZmlsbD0iIzMzMyIgdGV4dC1hbmNob3I9Im1pZGRsZSI+T3BlblBpcGVsaW5l
IEFyY2hpdGVjdHVyZTwvdGV4dD4KCiAgPCEtLSBTdGFnZSAxOiBJbmdlc3QgLS0+CiAgPHJlY3Qge
D0iMzAiIHk9IjYwIiB3aWR0aD0iMTIwIiBoZWlnaHQ9IjEwMCIgZmlsbD0iIGZpbGw9IiNFZmY2
ZjQiIHN0cm9rZT0iIzA2YjZkNCIgc3Ryb2tlLXdpZHRoPSIyIiByeD0iNSIvPgogIDx0ZXh0IHg9
IjkwIiB5PSIxMDUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxM
iIgZmlsbD0iIzA2NWY0NiIgdGV4dC1hbmNob3I9Im1pZGRsZSI+SW5nZXN0PC90ZXh0PgogIDx0ZX
h0IHg9IjkwIiB5PSIxMjUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXp
lPSIxMCIgZmlsbD0iIzA2NWY0NiIgdGV4dC1hbmNob3I9Im1pZGRsZSI+UmF3IERhdGE8L3RleH
Q+CgogIDwhLS0gU3RhZ2UgMjogUHJvY2VzcyAtLT4KICA8cmVjdCB4PSIxNzAiIHk9IjYwIiB3aWR
0aD0iMTIwIiBoZWlnaHQ9IjEwMCIgZmlsbD0iI2RiZWFmZSIgc3Ryb2tlPSIjM2I4MmY2IiBzdH
Jvbmctd2lkdGg9IjIiIHJ4PSI1Ii8+CiAgPHRleHQgeD0iMjMwIiB5PSIxMDUiIGZvbnQtZmFtaWx
5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMyIgZmlsbD0iIzFlNDBhZiIgdGV4dC
1hbmNob3I9Im1pZGRsZSI+VHJhbnNmb3JtPC90ZXh0PgogIDx0ZXh0IHg9IjIzMCIgeT0iMTI1I
iBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTIiIGZvbnQtd2VpZ2h0PSJib2xkIiBmaWxsPSIjMzM
```

mEoMjU1LDI1NSwyNTUsMC45KSIgdGV4dC1hbmNob3I9Im1pZGRsZSI+U2VsZWN0aW9uPC90ZXh0Pg
ogIDx0ZXh0IHg9IjI0NSIgeT0iMTM4IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZ
vbnQtc2l6ZT0iMTAiIGZpbGw9InJnYmEoMjU1LDI1NSwyNTUsMC45KSIgdGV4dC1hbmNob3I9Im1p
ZGRsZSI+TWF0Y2hpbmc8L3RleHQ+CiAgPHRleHQgeD0iMjQ1IiB5PSIxNTIiIGZvbnQtZmFtaWx5P
SJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0icmdiYSgyNTUsMjU1LDI1NS
wwLjkpIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5Db25kaXRpb25zPC90ZXh0PgoKICA8IS0tIEFycm9
3IDIgLS0+CiAgPHBhdGggZD0iTTMwNSwxMTAgTDMzMCwxMTAiIHN0cm9rZT0iIzY0NzQ4YiIgc3Ry
b2tlLXdpZHRoPSIyIiBmaWxsPSJub25lIiBtYXJrZXItZW5kPSJ1cmwoI29wQXJyb3cpIi8+CgogI
DwhLS0gU3RhZ2UgMzogUHJvY2Vzc luZyAtLT4KICA8cmVjdCB4PSIzNDAiIHk9IjYwIiB3aWR0aD
0iMTIwIiBoZWlnaHQ9IjEwMCIgcng9IjIgIGZpbGw9InJnYmEoMjU1LDI1NSwyNTUsMC45KSIgc3R
yb2tlPSIjYzY0NzQ4YiIgc3Ryb2tlLXdpZHRoPSIyIi8+CiAgPHRleHQgeD0iMzAwIiB5PSIxNDAi
IHRleHQtYW5jaG9yPSJtaWRkbGUiPlRyYW5zZm9ybTwvdGV4dD4KICA8dGV4dCB4PSIzMDAiIHk9I
jE1MiIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPS
JyZ2JhKDI1NSwyNTUsMjU1LDAuOSkiIHRleHQtYW5jaG9yPSJtaWRkbGUiPkRyb3A8L3RleHQ+CgogI
DwhLS0gQXJyb3cgMyAtLT4KICA8cGF0aCBkPSJNNDYwLDEzMCBMNDg1LDEzMCIgc3Ryb2tlLXdp
8L3RleHQ+CgogIDwhLS0gQ3NDhiIiBzdHJva2Utd2lkdGg9IjIiIGZpbGw9Im5vbmUiIG1hcmtlci1lbmQ9I
c3Ryb2tlPSIjNjQ3NDhiIiBzdHJva2Utd2lkdGg9IjIiIGZpbGw9Im5vbmUiIG1hcmtlci1lbmQ9I
nVybCgjb3BBcnJvdykiLz4KICAgPCEtLSBTdGFnZSA0OiBFeHRyYWN0IC0tPgogIDxyZWN0IHg9Ij
Q5NSIgeT0iNjAiIHdpZHRoPSIxMjAiIGhlaWdodD0iMTAwIiByeD0iOCIgZmlsbD0idXJsKCNleHR
yYWN0R3JhZCkiIGZpbHRlcj0idXJsKCNvcFNoYWRvdykiLz4KICA8dGV4dCB4PSIxNTUiIHk9Ijkw
IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTIiIGZvbnQtd2VpZ
2h0PSJib2xkIiBmaWxsPSJyZ2JhKDI1NSwyNTUsMjU1LDAuOSkiIHRleHQtYW5jaG9yPSJtaWRkbGUiPkxvc
8L3RleHQ+CgogIDwhLS0gRVhUUkFDVDwvdGV4dD
4KICA8cGF0aCBkPSJNNDYwLDEzMCBMNDg1LDEzMCIgc3Ryb2tlLXdpZHRoPSIyIiBmaWxsPSJub25lIiBtYXJrZXItZW5kPSJ1cmwoI29wQXJyb3cpIi8+CgogIDwhLS0gU3RhZ2UgNTogU
RvcmUgLS0+CiAgPHJlY3QgeD0iNjUwIiB5PSI2MCIgd2lkdGg9IjEyMCIgaGVpZ2h0PSIxMDAiIHJ
4PSI4IiBmaWxsPSJ1cmwoI3N0b3JlR3JhZCkiIGZpbHRlcj0idXJsKCNvcFNoYWRvdykiLz4KICA8
dGV4dCB4PSI3MDAiIHk9IjkwIiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc
2l6ZT0iMTIiIGZvbnQtd2VpZ2h0PSJib2xkIiBmaWxsPSJyZ2JhKDI1NSwyNTUsMjU1LDAuOSkiIH
RsZSI+U1RPUkU8L3RleHQ+CiAgPHBhdGggZD0iNzEwIiB5PSIxMTAiIGZvbnQtZmFtaWx5PSJBcml
hbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0icmdiYSgyNTUsMjU1LDI1NSwwLjkp
IiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5HcmFpbDwvdGV4dD4KICA8dGV4dCB4PSI3MDAiIHk9IjEyN

CIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSJyZ2
JhKDI1NSwyNTUsMjU1LDAuOSkiIHRleHQtYW5jaG9yPSJtaWRkbGUiPkJ1Y2tldHM8L3RleHQ+CiA
gPHRleHQgeD0iNzEwIiB5PSIxMzgiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9u
dC1zaXplPSIxMCIgZmlsbD0icmdiYSgyNTUsMjU1LDI1NSwwLjkpIiB0ZXh0LWFuY2hvcj0ibWlkZ
GxlIj5SZXRlbnRpb248L3RleHQ+CiAgPHRleHQgeD0iNzEwIiB5PSIxNTIiIGZvbnQtZmFtaWx5PS
JBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0icmdiYSgyNTUsMjU1LDI1NSw
wLjkpIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5Sb3V0aW5nPC90ZXh0PgoKICA8IS0tIEtleSBGZWF0
dXJlcyBTZWN0aW9uIC0tPgogIDxyZWN0IHg9IjMwIiB5PSIxODAiIHdpZHRoPSI3NDAiIGhlaWdod
D0iMTI1IiByeD0iOCIgZmlsbD0iI2ZmZiIgc3Ryb2tlPSIjZTJlOGYwIiBzdHJva2Utd2lkdGg9Ij
IiLz4KICA8dGV4dCB4PSI0MDAiIHk9IjIwNSIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcml
mIiBmb250LXNpemU9IjEyIiBmb250LXdlaWdodD0iYm9sZCIgZmlsbD0iIzMzMyIgdGV4dC1hbmNo
b3I9Im1pZGRsZSI+S2V5IFByaW5jaXBsZXM8L3RleHQ+CgogIDwhLS0gRmVhdHVyZSBib3hlcyAtL
T4KICA8cmVjdCB4PSI1MCIgeT0iMjIwIiB3aWR0aD0iMTYwIiBoZWlnaHQ9IjcwIiByeD0iNiIgZm
lsbD0iI2RiZWFmZSIvPgogIDx0ZXh0IHg9IjEzMCIgeT0iMjQ1IiBmb250LWZhbWlseT0iQXJpYWw
sIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZvbnQtd2VpZ2h0PSJib2xkIiBmaWxsPSIjMWU0
MGFmIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5QcmUtU3RvcmFnZTwvdGV4dD4KICA8dGV4dCB4PSIxM
zAiIHk9IjI2MiIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIi
BmaWxsPSIjMWU0MGFmIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5BbGwgcHJvY2Vzc2luZyBiZWZvcmU
8L3RleHQ+CiAgPHRleHQgeD0iMTMwIiB5PSIyNzYiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1z
ZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzFlNDBhZiIgdGV4dC1hbmNob3I9Im1pZGRsZSI+Z
GF0YSBpcyBwZXJzaXN0ZWQ8L3RleHQ+CgogIDxyZWN0IHg9IjIzMCIgeT0iMjIwIiB3aWR0aD0iMT
YwIiBoZWlnaHQ9IjcwIiByeD0iNiIgZmlsbD0iI2ZlZjNjNyIvPgogIDx0ZXh0IHg9IjMxMCIgeT0
iMjQ1IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZvbnQt
d2VpZ2h0PSJib2xkIiBmaWxsPSIjOTI0MDBlIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5NdWx0aS1Qa
XBlbGluZTwvdGV4dD4KICA8dGV4dCB4PSIzMTAiIHk9IjI2MiIgZm9udC1mYW1pbHk9IkFyaWFsLC
BzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSIjOTI0MDBlIiB0ZXh0LWFuY2hvcj0ibWl
kZGxlIj5VcCB0byA1IHBpcGVsaW5lcyBwZXI8L3RleHQ+CiAgPHRleHQgeD0iMzEwIiB5PSIyNzYi
IGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzkyN
DAwZSIgdGV4dC1hbmNob3I9Im1pZGRsZSI+cmVjb3JkIHNpbXVsdGFuZW91c2x5PC90ZXh0PgoKIC
A8cmVjdCB4PSI0MTAiIHk9IjIyMCIgd2lkdGg9IjE2MCIgaGVpZ2h0PSI3MCIgcng9IjYiIGZpbGw
9IiNkMWZhZTUiLz4KICA8dGV4dCB4PSI0OTAiIHk9IjI0NSIgZm9udC1mYW1pbHk9IkFyaWFsLCBz
YW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmb250LXdlaWdodD0iYm9sZCIgZmlsbD0iIzA0Nzg1N
yIgdGV4dC1hbmNob3I9Im1pZGRsZSI+U2VjdXJpdHkgRmlyc3Q8L3RleHQ+CiAgPHRleHQgeD0iND
kwIiB5PSIyNjIiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCI
gZmlsbD0iIzA0Nzg1NyIgdGV4dC1hbmNob3I9Im1pZGRsZSI+TWFza2luZyBiZWZvcmUgYW55PC90
ZXh0PgogIDx0ZXh0IHg9IjQ5MCIgeT0iMjc2IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2Vya
WYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiMwNDc4NTciIHRleHQtYW5jaG9yPSJtaWRkbGUiPm90aG
VyIHByb2Nlc3Npbmc8L3RleHQ+CgogIDxyZWN0IHg9IjU5MCIgeT0iMjIwIiB3aWR0aD0iMTYwIiB
oZWlnaHQ9IjcwIiByeD0iNiIgZmlsbD0iI2ZjZTdmMyIvPgogIDx0ZXh0IHg9IjY3MCIgeT0iMjQ1
IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZvbnQtd2VpZ
2h0PSJib2xkIiBmaWxsPSIjOWQxNzRkIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5FbnRpdHkgRGV0ZW
N0aW9uPC90ZXh0PgogIDx0ZXh0IHg9IjY3MCIgeT0iMjYyIiBmb250LWZhbWlseT0iQXJpYWwsIHN
hbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM5ZDE3NGQiIHRleHQtYW5jaG9yPSJtaWRk
bGUiPmR0LmVudGl0eS4qIGFkZGVkIGFmdGVyPC90ZXh0PgogIDx0ZXh0IHg9IjY3MCIgeT0iMjc2I
iBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM5ZD
E3NGQiIHRleHQtYW5jaG9yPSJtaWRkbGUiPlByb2Nlc3NpbmcgU3RhZ2U8L3RleHQ+Cjwvc3ZnPgo
=)

```
### Core Capabilities

| Capability | Description |
|------------|-------------|
| **Unified Ingestion** | Single solution for logs, spans, metrics, events,
and business events |
| **Dynamic Routing** | Route data to specific pipelines based on matching
conditions |
| **Real-time Processing** | Transform, enrich, and mask data at ingestion
time |
| **Metric Extraction** | Create metrics from any data source for long-term
analytics |
| **Event Generation** | Generate custom events and business events from
incoming data |
| **Bucket Management** | Control retention and cost with targeted storage
routing |
| **Security & Compliance** | Mask sensitive data before storage |

---

## Classic vs OpenPipeline: Key Differences

Understanding the fundamental differences helps you plan your migration
effectively.

### Feature Comparison

| Feature | Classic Ingestion | OpenPipeline v2.0 |
|---------|-------------------|-------------------|
| **Data Types** | Logs only | Logs, Spans, Metrics, Events, Bizevents |
| **Processing Location** | Post-storage | Pre-storage (at ingestion) |
| **Parsing** | Limited built-in parsers | Full DQL + DPL (Dynatrace Pattern
Language) |
| **Routing** | Basic log sources | Dynamic routing with matching conditions
|
| **Metric Extraction** | Not available | Extract metrics with dimensions |
| **Event Generation** | Not available | Generate events and bizevents |
| **Data Masking** | Post-processing only | At ingestion (before storage) |
| **Cost Control** | Limited | Drop unwanted data before storage |
| **Bucket Routing** | Default only | Route to custom buckets per pipeline |
| **API Endpoints** | `/api/v2/logs/ingest` | Multiple endpoints per data
type |
| **Configuration** | Settings → Logs | Settings → OpenPipeline |

### Processing Stage Comparison
```

![Classic vs OpenPipeline Processing]
(
ciIHZpZXdCb3g9IjAgMCA4MDAgMzAwIj4KICA8ZGVmcz4KICAgIDxsaW5lYXJHcmFkaWVudCBpZD0
iY2xhc3NpY0dyYWQiIHgxPSIwJSIgeTE9IjAlIiB4Mj0iMTAwJSIgeTI9IjAlIj4KICAgICAgPHN0
b3Agb2Zmc2V0PSIwJSIgc3R5bGU9InN0b3AtY29sb3I6I2Y1OWUwYjtzdG9wLW9wYWNpdHk6MSIgL
z4KICAgICAgPHN0b3Agb2Zmc2V0PSIxMDAlIiBzdHlsZT0ic3RvcC1jb2xvcjojZDk3NzA2O3N0b3
Atb3BhY2l0eToxIiAvPgogICAgPC9saW5lYXJHcmFkaWVudD4KICAgIDxsaW5lYXJHcmFkaWVudCB
pZD0ibW9kZXJuR3JhZCIgeDE9IjAlIiB5MT0iMCUiIHgyPSIxMDAlIiB5Mj0iMCUiPgogICAgICA8
c3RvcCBvZmZzZXQ9IjAlIiBzdHlsZT0ic3RvcC1jb2xvcjojMjJjNTVlO3N0b3Atb3BhY2l0eToxI
iAvPgogICAgICA8c3RvcCBvZmZzZXQ9IjEwMCUiIHN0eWxlPSJzdG9wLWNvbG9yOiMxNmEzNGE7c3
RvcC1vcGFjaXR5OjEiIC8+CiAgICA8L2xpbmVhckdyYWRpZW50PgogICAgPGZpbHRlciBpZD0iY3Z
zU2hhZG93Ij4KICAgICAgPGZlRHJvcFNoYWRvdyBkeD0iMSIgZHk9IjEiIHN0ZERldmlhdGlvbj0i
MiIgZmxvb2Qtb3BhY2l0eT0iMC4xNSIvPgogICAgPC9maWx0ZXI+CiAgICA8bWFya2VyIGlkPSJjd
nNBcnJvdyIgbWFya2VyV2lkdGg9IjEwIiBtYXJrZXJIZWlnaHQ9IjciIHJlZlg9IjkiIHJlZlk9Ij
MuNSIgb3JpZW50PSJhdXRvIj4KICAgICAgPHBvbHlnb24gcG9pbnRzPSIwIDAsIDEwIDMuNSwgMCA
3IiBmaWxsPSIjNjQ3NDhiIi8+CiAgICA8L21hcmtlcj4KICA8L2RlZnM+CgogIDwhLS0gQmFja2dy
b3VuZCAtLT4KICA8cmVjdCB3aWR0aD0iODAwIiBoZWlnaHQ9IjMwMCIgZmlsbD0iI2Y4ZjlmYSIgc
ng9IjEwIi8+CgogIDwhLS0gVGl0bGUgLS0+CiAgPHRleHQgeD0iNDAwIiB5PSIyOCIgZm9udC1mYW
1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjE4IiBmb250LXdlaWdodD0iYm9sZCI
gZmlsbD0iIzMzMyIgdGV4dC1hbmNob3I9Im1pZGRsZSI+Q2xhc3NpYyB2cyBPcGVuUGlwZWxpbmUg
UHJvY2Vzc2luZzwvdGV4dD4KCiAgPCEtLSBDbGFzc2ljIFNlY3Rpb24gLS0+CiAgPHJlY3QgeD0iM
zAiIHk9IjU1IiB3aWR0aD0iNzQwIiBoZWlnaHQ9Ijk1IiByeD0iOCIgZmlsbD0iI2ZmZiIgc3Ryb2
tlPSIjZjU5ZTBiIiBzdHJva2Utd2lkdGg9IjIiLz4KICA8cmVjdCB4PSIzMCIgeT0iNTUiIHdpZHR
oPSI3NDAiIGhlaWdodD0iMjgiIHJ4PSI4IiBmaWxsPSJ1cmwoI2NsYXNzaWNHcmFkKSIvPgogIDx0
ZXh0IHg9IjQwMCIgeT0iNzUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1za
XplPSIxMiIgZm9udC13ZWlnaHQ9ImJvbGQiIGZpbGw9IndoaXRlIiB0ZXh0LWFuY2hvcj0ibWlkZG
xlIj5DTEFTU0lDIElOR0VTVElPTiAoUG9zdC1TdG9yYWdlIFByb2Nlc3NpbmcpPC90ZXh0PgoKICA
8IS0tIENsYXNzaWMgZmxvdyAtLT4KICA8cmVjdCB4PSI2MCIgeT0iMTAwIiB3aWR0aD0iMTAwIiBo
ZWlnaHQ9IjM1IiByeD0iNiIgZmlsbD0iI2ZlZjNjNyIgZmlsdGVyPSJ1cmwoI2N2c1NoYWRvdykiL
z4KICA8dGV4dCB4PSIxMTAiIHk9IjEyMiIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIi
Bmb250LXNpemU9IjEwIiBmb250LXdlaWdodD0iYm9sZCIgZmlsbD0iIzkyNDAwZSIgdGV4dC1hbmN
ob3I9Im1pZGRsZSI+SW5nZXN0PC90ZXh0PgoKICA8cGF0aCBkPSJNMTYwLDExNyBMMjAwLDExNyIg
c3Ryb2tlPSIjNjQ3NDhiIiBzdHJva2Utd2lkdGg9IjIiIGZpbGw9Im5vbmUiIG1hcmtlci1lbmQ9I
nVybCgjY3ZzQXJyb3cpIi8+CgogIDxyZWN0IHg9IjIxMCIgeT0iMTAwIiB3aWR0aD0iMTAwIiBoZW
lnaHQ9IjM1IiByeD0iNiIgZmlsbD0iI2ZlZjNjNyIgZmlsdGVyPSJ1cmwoI2N2c1NoYWRvdykiLz4
KICA8dGV4dCB4PSIyNjAiIHk9IjEyMiIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBm
b250LXNpemU9IjEwIiBmb250LXdlaWdodD0iYm9sZCIgZmlsbD0iIzkyNDAwZSIgdGV4dC1hbmNob
3I9Im1pZGRsZSI+U3RvcmU8L3RleHQ+CgogIDxwYXRoIGQ9Ik0zMTAsMTE3IEwzNTAsMTE3IiBzdH
Jva2U9IiM2NDc0OGIiIHN0cm9rZS13aWR0aD0iMiIgZmlsbD0ibm9uZSIgbWFya2VyLWVuZD0idXJ
sKCNjdnNBcnJvdykiLz4KCiAgPHJlY3QgeD0iMzYwIiB5PSIxMDAiIHdpZHRoPSIxMDAiIGhlaWdo
dD0iMzUiIHJ4PSI2IiBmaWxsPSIjZmVmM2M3IiBmaWx0ZXI9InVybCgjY3ZzU2hhZG93KSIvPgogI
Dx0ZXh0IHg9IjQxMCIgeT0iMTIyIiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbn
Qtc2l6ZT0iMTAiIGZvbnQtd2VpZ2h0PSJib2xkIiBmaWxsPSIjOTI0MDBlIiB0ZXh0LWFuY2hvcj0
ibWlkZGxlIj5RdWVyeTwvdGV4dD4KCiAgPHRleHQgeD0iNTEwIiB5PSIxMjIiIGZvbnQtZmFtaWx5
PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzkyNDAwZSIgdGV4dC1hb
mNob3I9InN0YXJ0Ij4oUGFyc2UgYXQgcXVlcnkgdGltZSk8L3RleHQ+CgogIDwhLS0gTGltaXRhdG
lvbiBpY29ucyAtLT4KICA8dGV4dCB4PSI2NDAiIHk9IjEwNSIgZm9udC1mYW1pbHk9IkFyaWFsLCB
zYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSIjZGMyNjI2Ij7inYwgTm8gcHJlLXByb2Nl

```
c3Npbmc8L3RleHQ+CiAgPHRleHQgeD0iNjQwIiB5PSIxMjAiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc
2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iI2RjMjYyNiI+4p2MIE5vIG1hc2tpbmc8L3
RleHQ+CiAgPHRleHQgeD0iNjQwIiB5PSIxMzUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJ
pZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iI2RjMjYyNiI+4p2MIE5vIGNvc3QgY29udHJvbDwvdGV4
dD4KCiAgPCEtLSBPcGVuUGlwZWxpbmUgU2VjdGlvbiAtLT4KICA8cmVjdCB4PSIyIi8+CiAgPHR
iaB3aWR0aD0iNzQwIiBoZWlnaHQ9IjEyMCIgcng9IjgiIGZpbGw9IiNmZmYiIHN0cm9rZT0iIzIyYz
U1ZSIgc3Ryb2tlLXdpZHRoPSIyIi8+CiAgPHJlY3QgeD0iMzAiIHk9IjE2NSIgd2lkdGg9Ijc0MCI
gaGVpZ2h0PSIyOCIgcng9IjgiIGZpbGw9InVybCgjbW9kZXJuR3JhZCkiLz4KICA8dGV4dCB4PSI0
MDAiIHk9IjE4NSIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjE4
MDAiIHk9IjE4NSIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjE4
CB4PSI5MCIgeT0iMjMyIiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT
0iMTAiIGZvbnQtd2VpZ2h0PSJib2xkIiBmaWxsPSIjMDQ3ODU3IiB0ZXh0LWFuY2hvcj0ibWlkZGx
lIj5Jbmdlc3Q8L3RleHQ+CgogIDxwYXRoIGQ9Ik0xMzAsMjI3IEwxNTAsMjI3IiBzdHJva2U9IiM2
NDc0OGIiIHN0cm9rZS13aWR0aD0iMiIgZmlsbD0ibm9uZSIgbWFya2VyLWVuZD0idXJsKCNjdnNBc
nJvdykiLz4KICAgPHJlY3QgeD0iMTYwIiB5PSIyMTAiIHdpZHRoPSI4MCIgaGVpZ2h0PSIzNSIgcn
g9IjYiIGZpbGw9IiNkMWZhZTUiIGZpbHRlcj0idXJsKCNjdnNTaGFkb3cpIi8+CiAgPHRleHQgeD0
iMjAwIiB5PSIyMzIiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIx
MCIgZm9udC13ZWlnaHQ9ImJvbGQiIGZpbGw9IiMwNDc4NTciIHRleHQtYW5jaG9yPSJtaWRkbGUiP
lJvdXRlPC90ZXh0PgoKICA8cGF0aCBkPSJNMjQwLDIyNyBMMjYwLDIyNyIgc3Ryb2tlPSIjNjQ3ND
hiIiBzdHJva2Utd2lkdGg9IjIiIGZpbGw9Im5vbmUiIG1hcmtlci1lbmQ9InVybCgjY3ZzQXJyb3c
pIi8+CogIDxyZWN0IHg9IjI3MCIgeT0iMjEwIiB3aWR0aD0iODAiIGhlaWdodD0iMzUiIHJ4PSI2
IiBmaWxsPSIjZDFmYWU1IiBmaWx0ZXI9InVybCgjY3ZzU2hhZG93KSIvPgogIDx0ZXh0IHg9IjMxM
CIgeT0iMjMyIiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIG
ZvbnQtd2VpZ2h0PSJib2xkIiBmaWxsPSIjMDQ3ODU3IiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5Qcm9
jZXNzPC90ZXh0PgoKICA8cGF0aCBkPSJNMzUwLDIyNyBMMzcwLDIyNyIgc3Ryb2tlPSIjNjQ3NDhi
IiBzdHJva2Utd2lkdGg9IjIiIGZpbGw9Im5vbmUiIG1hcmtlci1lbmQ9InVybCgjY3ZzQXJyb3cpI
i8+CogIDxyZWN0IHg9IjM4MCIgeT0iMjEwIiB3aWR0aD0iODAiIGhlaWdodD0iMzUiIHJ4PSI2Ii
BmaWxsPSIjZDFmYWU1IiBmaWx0ZXI9InVybCgjY3ZzU2hhZG93KSIvPgogIDx0ZXh0IHg9IjQyMCI
geT0iMjMyIiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZv
bnQtd2VpZ2h0PSJib2xkIiBmaWxsPSIjMDQ3ODU3IiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5FeHRyY
WN0PC90ZXh0PgoKICA8cGF0aCBkPSJNNDYwLDIyNyBMNDgwLDIyNyIgc3Ryb2tlPSIjNjQ3NDhiIi
BzdHJva2Utd2lkdGg9IjIiIGZpbGw9Im5vbmUiIG1hcmtlci1lbmQ9InVybCgjY3ZzQXJyb3cpIi8
+CogIDxyZWN0IHg9IjQ5MCIgeT0iMjEwIiB3aWR0aD0iODAiIGhlaWdodD0iMzUiIHJ4PSI2IiBm
aWxsPSIjZDFmYWU1IiBmaWx0ZXI9InVybCgjY3ZzU2hhZG93KSIvPgogIDx0ZXh0IHg9IjUzMCIge
T0iMjMyIiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZvbn
Qtd2VpZ2h0PSJib2xkIiBmaWxsPSIjMDQ3ODU3IiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5TdG9yZTw
vdGV4dD4KCiAgPCEtLSBCZW5lZml0IGljb25zIC0tPgogIDxwYXRoIGQ9Ik U5MCIgeT0iMjE1IiBm
b250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiMxNmEzN
GEiPuKchSBQcmUtc3RvcmFnZSBtYXNraW5nPC90ZXh0PgogIDx0ZXh0IHg9IjUwMCIgeT0iMjE1Ii
Bmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiMxNmE
zNGEiPuKchSBzdG9yYWdlPC90ZXh0PgogIDx0ZXh0IHg9IjY1MCIgeT0iMjE1IiBmb250LWZhbWls
eT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiMxNmEzNGEiPuKchSB
mEzNGEiPuKchSBFeHRyYWN0IG1ldHJpY3M8L3RleHQ+CiAgPHRleHQgeD0iNTkwIiB5PSIyNjAiIG
ZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzE2YTM
0YSI+4pyFIEJ1Y2tldCByb3V0aW5nPC90ZXh0PgoKICA8IS0tIFN1Yi1sYWJlbHMgZm9yIFByb2Nl
```

```
c3MgLS0+CiAgPHRleHQgeD0iMzEwIiB5PSIyNTUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZ
XJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzA0Nzg1NyIgdGV4dC1hbmNob3I9Im1pZGRsZSI+TW
FzaywgUGFyc2U8L3RleHQ+CiAgPHRleHQgeD0iMzEwIiB5PSIyNjUiIGZvbnQtZmFtaWx5PSJBcml
hbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzA0Nzg1NyIgdGV4dC1hbmNob3I9
Im1pZGRsZSI+VHJhbnNmb3JtLCBEcm9wPC90ZXh0Pgo8L3N2Zz4K)



> ⚠️ **Important:** With OpenPipeline, data processing happens **before** storage. This means you can reduce storage costs by dropping unwanted data and masking sensitive information before it's ever written to Grail.

---

## Benefits of Migrating

### 1. 💰 Cost Optimization

**Drop unwanted data before storage:**
- Filter out debug logs, health checks, and noise
- Reduce storage costs by 30-70% in typical deployments
- Route high-volume, low-value data to shorter retention buckets

### 2. 🔐 Security & Compliance

**Mask sensitive data at ingestion:**
- PII (Personal Identifiable Information) never touches storage
- Credit card numbers, SSNs, emails masked before persistence
- Meet GDPR, HIPAA, PCI-DSS requirements

### 3. 📊 Enhanced Analytics

**Extract metrics with dimensions:**
- Create custom metrics from any log pattern
- Build long-term trend dashboards
- Enable business KPI tracking from technical data

### 4. ⚡ Improved Query Performance

**Parse once, query fast:**
- Structured fields extracted at ingestion
- No runtime parsing overhead
- Faster dashboards and alerts

### 5. 🎯 Unified Data Processing

**Single solution for all data types:**
- Consistent processing for logs, spans, metrics, events
```

```
- Centralized configuration management
- Simplified operations and governance

### 6. 🔄 Real-time Enrichment

**Add context at ingestion:**
- Add environment tags (prod, staging, dev)
- Enrich with business context
- Standardize field names across sources

---

## OpenPipeline Configuration Scopes

OpenPipeline supports multiple **configuration scopes** - each handling a
different data type:

| Configuration Scope | Data Type | Use Case |
|---------------------|-----------|----------|
| **Logs** | Log records | Application logs, system logs, audit logs |
| **Spans** | Distributed traces | Span processing, trace enrichment |
| **Metrics** | Time-series data | Metric ingestion and transformation |
| **Events** | Platform events | Generic events, Davis events, SDLC events |
| **Business Events** | Business analytics | User journeys, transactions,
conversions |
| **Security Events** | Security data | Vulnerability findings, compliance
events |
| **System Events** | Infrastructure | System-level events and alerts |

### Ingest Sources per Scope

Each scope supports different ingest sources:

**Logs:**
- OneAgent log ingestion
- Generic log API (`/api/v2/logs/ingest`)
- OTLP logs
- Fluent integrations

**Spans:**
- OneAgent distributed tracing
- OTLP spans
- OpenTelemetry collectors

**Metrics:**
- OneAgent metrics
- OTLP metrics
- Metric ingestion API
```

```
> 💡 **Tip:** Access OpenPipeline configuration at: **Settings → Process and contextualize → OpenPipeline**

---

## API Migration: Classic → OpenPipeline v2.0

### API Endpoint Compatibility

Good news: **The API endpoint remains the same!**

| Endpoint | Classic | OpenPipeline | Notes |
|----------|---------|--------------|-------|
| **Logs** | `/api/v2/logs/ingest` | `/api/v2/logs/ingest` | ✅ No change required |
| **OTLP Logs** | Not available | `/otlp/v1/logs` | ✅ New endpoint |
| **Spans** | `/api/v2/otlp/v1/traces` | `/api/v2/otlp/v1/traces` | ✅ No change required |

> 💡 **Migration Tip:** You don't need to update your API calls! OpenPipeline automatically receives data sent to `/api/v2/logs/ingest`. The difference is in **how** the data is processed after ingestion.

### Ingestion Methods Comparison

| Method | Classic | OpenPipeline | `dt.openpipeline.source` Value |
|--------|---------|--------------|-------------------------------|
| **OneAgent** | ✅ Supported | ✅ Supported | `oneagent` |
| **Generic Log API** | ✅ Supported | ✅ Supported | `generic` |
| **OTLP Protocol** | ⚠️ Limited | ✅ Full support | `otlp` |
| **Fluent Bit** | ✅ Via API | ✅ Via API | `generic` |
| **Fluentd** | ✅ Via API | ✅ Via API | `generic` |
| **Logstash** | ✅ Via API | ✅ Via API | `generic` |
| **Vector** | ✅ Via API | ✅ Via API | `generic` |

### Required Token Permissions

| Scope | Classic | OpenPipeline | Notes |
|-------|---------|--------------|-------|
| `logs.ingest` | ✅ Required | ✅ Required | Same permission |
| `metrics.ingest` | N/A | ⚠️ Optional | Only if using metric extraction |
| `events.ingest` | N/A | ⚠️ Optional | Only if using event extraction |

### Migration Strategy for API Clients

| Scenario | Action Required |
|----------|-----------------|
```

```
| **Using OneAgent** | ✅ No code changes | OpenPipeline handles
automatically |
| **Using `/api/v2/logs/ingest`** | ✅ No code changes | Same endpoint works
|
| **Using custom log shippers** | ⚠️ Optional | Add `log.source` field for
routing |
| **Want to leverage new features** | ⚠️ Configure pipelines | Create
OpenPipeline config in UI |

---

## Real-World Migration Scenarios

### Scenario 1: E-Commerce Platform

**Challenge:**
- 50M logs/day (70% debug logs)
- Payment logs contain credit card numbers
- Need to track order metrics from logs

**OpenPipeline Solution:**
1. **Drop debug logs** → Reduce volume by 70%
2. **Mask credit cards** → PCI-DSS compliance
3. **Extract payment metrics** → `order.amount`, `order.count`
4. **Route to tiered buckets** → High-value logs = 90 days, others = 7 days

**Result:** 70% cost savings, PCI compliance, new business metrics

### Scenario 2: Healthcare SaaS

**Challenge:**
- HIPAA compliance required
- Logs contain patient IDs, MRNs, SSNs
- Need audit trail for 7 years

**OpenPipeline Solution:**
1. **Mask all PHI fields** → Patient IDs, SSNs, MRNs
2. **Route audit logs** → Dedicated bucket with 2555-day retention
3. **Extract security events** → Failed auth attempts, data access
4. **Drop health checks** → Reduce noise

**Result:** HIPAA compliance, 7-year audit retention, 40% cost savings

### Scenario 3: FinTech Startup (ELK Migration)

**Challenge:**
- Migrating from ELK stack to Dynatrace
- Custom log formats (not JSON)
```

- Need APM + logs correlation

OpenPipeline Solution:
1. **Parse custom log formats** → DPL patterns for structured extraction
2. **Extract request IDs** → Correlate with traces
3. **Create SLI metrics** → Error rate, latency percentiles
4. **Unified observability** → Logs + APM in one platform

Result: ELK replacement, APM correlation, unified observability

Scenario 4: Global Retailer (Multi-Region)

Challenge:
- Multi-region deployment (US, EU, APAC)
- GDPR compliance for EU customers
- 100+ microservices

OpenPipeline Solution:
1. **Environment-based routing** → prod/staging/dev to different buckets
2. **Mask PII for EU** → Email, IP addresses for EU region logs
3. **Service-level pipelines** → Dedicated processing per critical service
4. **Metric extraction** → Service-level SLIs

Result: Multi-region compliance, per-service observability, 50% cost reduction

---

Understanding Your Current State

Before migrating, you need to understand your current log ingestion landscape. The following queries help you assess what you're working with.

```python
// Identify your current log sources and volume
// This shows which sources are sending the most logs
fetch logs, from: now() - 7d
| summarize {log_count = count()}, by: {log.source}
| sort log_count desc
| limit 25
```

```python
// Check which logs are already processed by OpenPipeline vs Classic
// This helps identify migration progress
fetch logs, from: now() - 24h
```

```
| fieldsAdd pipeline_type = if(isNotNull(dt.openpipeline.pipelines),
"OpenPipeline",
                            else: if(isNotNull(dt.openpipeline.source),
"OpenPipeline",
                            else: "Classic"))
| summarize {log_count = count()}, by: {pipeline_type}
| sort log_count desc
```

```python
// Analyze log volume by OpenPipeline source
// Identify the ingestion methods being used
fetch logs, from: now() - 24h
| summarize {log_count = count()}, by: {dt.openpipeline.source}
| sort log_count desc
```

```python
// Check current bucket distribution
// Understand where your logs are being stored
fetch logs, from: now() - 24h
| summarize {log_count = count()}, by: {dt.system.bucket}
| sort log_count desc
```

```python
// Identify which pipelines are processing your logs
// Shows custom pipelines already configured
fetch logs, from: now() - 24h
| filter isNotNull(dt.openpipeline.pipelines)
| summarize {log_count = count()}, by: {dt.openpipeline.pipelines}
| sort log_count desc
```

Migration Readiness Assessment

Use these queries to assess your migration readiness and identify areas that need attention.

```python
// Check parsing coverage - how many logs have structured data?
// Low coverage indicates need for parsing pipelines
fetch logs, from: now() - 24h
| fieldsAdd has_structured_data = isNotNull(loglevel) OR isNotNull(status)
| summarize {
    total = count(),
```

```
    structured = countIf(has_structured_data),
    unstructured = countIf(NOT has_structured_data)
  }
| fieldsAdd coverage_pct = round((toDouble(structured) / toDouble(total)) *
100, decimals: 1)
```

```python
// Identify logs that could be dropped to save costs
// Debug logs and health checks are common candidates
fetch logs, from: now() - 24h
| summarize {
    total = count(),
    debug_logs = countIf(loglevel == "DEBUG" OR status == "DEBUG"),
    info_logs = countIf(loglevel == "INFO" OR status == "INFO"),
    health_checks = countIf(contains(toString(content), "health") OR
contains(toString(content), "heartbeat")),
    metrics_endpoints = countIf(contains(toString(content), "/metrics") OR
contains(toString(content), "/prometheus"))
  }
| fieldsAdd droppable = debug_logs + health_checks + metrics_endpoints
| fieldsAdd potential_savings_pct = round((toDouble(droppable) /
toDouble(total)) * 100, decimals: 1)
```

```python
// Find logs with potential PII that needs masking
// Look for common patterns that might contain sensitive data
fetch logs, from: now() - 1h
| filter contains(toString(content), "email")
    OR contains(toString(content), "password")
    OR contains(toString(content), "ssn")
    OR contains(toString(content), "credit")
    OR contains(toString(content), "@")
| summarize {potentially_sensitive = count()}, by: {log.source}
| sort potentially_sensitive desc
| limit 20
```

```python
// Analyze log level distribution
// Helps identify noise reduction opportunities
fetch logs, from: now() - 24h
| summarize {log_count = count()}, by: {loglevel}
| sort log_count desc
```

```python
```

```
// Check log volume trends over time
// Understand your ingestion patterns
fetch logs, from: now() - 7d
| makeTimeseries {log_count = count()}, interval: 1h
```

Migration Readiness Checklist

Based on your assessment queries, complete this checklist:

Discovery Phase
- [ ] Identified all log sources (`log.source` values)
- [ ] Documented current log volume by source
- [ ] Identified which logs are already on OpenPipeline
- [ ] Analyzed current bucket usage

Planning Phase
- [ ] Identified logs that can be dropped (debug, health checks)
- [ ] Identified logs requiring parsing (unstructured content)
- [ ] Identified logs with sensitive data requiring masking
- [ ] Planned bucket strategy (retention periods, cost tiers)

Configuration Phase
- [ ] Created custom pipelines for each use case
- [ ] Configured dynamic routing rules
- [ ] Set up parsing processors (DQL/DPL)
- [ ] Configured masking for sensitive data
- [ ] Set up metric extraction where needed
- [ ] Configured bucket routing

Validation Phase
- [ ] Tested pipelines with sample data
- [ ] Verified parsing produces expected fields
- [ ] Confirmed masking works correctly
- [ ] Validated metrics are being extracted
- [ ] Checked data appears in correct buckets

OpenPipeline Limits & Constraints

Before migrating, understand these key limits:

Data Size Limits

| Limit | Value | What Happens if Exceeded |

```
|-------|-------|------------------------|
| **Max record size (after processing)** | 16 MB | Record is **dropped** |
| **Working memory per record** | 16 MB | Processing fails, record dropped |
| **Log attribute size** | 32 KB | Attribute is **truncated** |
| **Max field name length** | 255 characters | Field creation fails |
| **Max string field length** | 4 KB | Content truncated |

### Processing Limits

| Limit | Value | Impact |
|-------|-------|--------|
| **Max extractions per record** | 5 pipelines | Record processed by max 5 pipelines |
| **Max processors per pipeline** | 50 processors | Cannot add more processors |
| **Max DQL commands per processor** | 10 commands | Split into multiple processors |
| **Max parse operations per processor** | 100 patterns | Create multiple parse processors |
| **Processing timeout** | 30 seconds | Record dropped if processing exceeds |

### Timestamp Constraints

| Data Type | Timestamp Range | Records Outside Range |
|-----------|-----------------|----------------------|
| **Logs** | 24 hours past to 10 min future | **Dropped** |
| **Spans** | 2 hours past | **Dropped** |
| **Events** | 24 hours past to 10 min future | **Dropped** |

> ⚠️ **Important:** Always send data with recent timestamps. Historical data imports require special considerations.

### Routing & Pipeline Limits

| Limit | Value | Notes |
|-------|-------|-------|
| **Max custom pipelines** | 100 pipelines | Per environment |
| **Max dynamic routes** | 100 routes | Per configuration scope |
| **Max conditions per route** | 10 conditions | Use AND/OR to combine |

### Field Restrictions

**Read-Only Fields** (Cannot be modified in pipelines):
- `dt.ingest.*` - Ingestion metadata
- `dt.openpipeline.*` - Pipeline processing metadata
- `dt.retain.*` - Retention information
- `dt.system.*` - System metadata (including bucket)
```

```
**Entity Fields** (Added **after** Processing stage):
- `dt.entity.service`
- `dt.entity.host`
- `dt.entity.process_group`
- `dt.entity.kubernetes_cluster`

> 💡 **Design Tip:** Entity fields are NOT available during routing or processing. They're added automatically by Dynatrace before the Extraction stage.

---

---

## Next Steps

Now that you understand OpenPipeline and have assessed your current state, continue with the migration series:

| Notebook | Focus Area |
|----------|------------|
| **OPMIG-02** | OpenPipeline Architecture & Key Concepts |
| **OPMIG-03** | Migration Assessment & Planning |
| **OPMIG-04** | Pipeline Configuration Fundamentals |
| **OPMIG-05** | Routing & Bucket Management |
| **OPMIG-06** | Processing, Parsing & Transformation |
| **OPMIG-07** | Metric & Event Extraction |
| **OPMIG-08** | Security, Masking & Compliance |
| **OPMIG-09** | Troubleshooting & Validation |

---

## References

- [OpenPipeline Documentation](https://docs.dynatrace.com/docs/discover-dynatrace/platform/openpipeline)
- [OpenPipeline Limits](https://docs.dynatrace.com/docs/discover-dynatrace/platform/openpipeline/reference/limits)
- [Processing Examples](https://docs.dynatrace.com/docs/discover-dynatrace/platform/openpipeline/use-cases/processing-examples)
- [Log Processing Tutorial](https://docs.dynatrace.com/docs/discover-dynatrace/platform/openpipeline/use-cases/tutorial-log-processing-pipeline)
- [Ingest API Reference](https://docs.dynatrace.com/docs/discover-dynatrace/platform/openpipeline/reference/api-ingestion-reference)

---
```

```
*Last Updated: December 12, 2025*
```