```
# 🔒 Security & Data Protection

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

## Sensitive Data Discovery, Masking, and Compliance

This notebook covers sensitive data discovery, OpenPipeline masking
configuration, security event monitoring, and compliance reporting.

---

## Table of Contents

1. Sensitive Data Discovery
2. OpenPipeline Masking Configuration
3. IP Address Analysis
4. Security Event Monitoring
5. Audit Log Queries
6. Compliance Reporting
7. Masking Best Practices


## Prerequisites

- ✅ Access to a Dynatrace environment with log data
- ✅ Completed OPLOGS-01 through OPLOGS-07
- ✅ Understanding of data privacy requirements



## 1. Sensitive Data Discovery

Before implementing masking, discover what sensitive data exists in your
logs.

![Sensitive Data Classification]
(
ciIHZpZXdCb3g9IjAgMCA4MDAgMzQwIj4KICA8ZGVmcz4KICAgIDxsaW5lYXJHcmFkaWVudCBpZD0
iY3JpdGljYWxHcmFkIiB4MT0iMCUiIHkxPSIwJSIgeDI9IjEwMCUiIHky PSIxMDAlIj4KICAgICAg
PHN0b3Agb2Zmc2V0PSIwJSIgc3R5bGU9InN0b3AtY29sb3I6I2VmNDQ0NDtzdG9wLW9wYWNpdHk6M
SIgLz4KICAgICAgPHN0b3Agb2Zmc2V0PSIxMDAlIiBzdHlsZT0ic3RvcC1jb2xvcjojZGMyNjI2O3
N0b3Atb3BhY2l0eToxIiAvPgogICAgPC9saW5lYXJHcmFkaWVudD4KICAgIDxsaW5lYXJHcmFkaWV
udCBpZD0iaGlnaEdyYWQiIHgxPSIwJSIgeTE9IjAlIiB4Mj0iMTAwJSIgeTI9IjEwMCUiPgogICAg
ICA8c3RvcCBvZmZzZXQ9IjAlIiBzdHlsZT0ic3RvcC1jb2xvcjojZjk3MzE2O3N0b3Atb3BhY2l0e
ToxIiAvPgogICAgICA8c3RvcCBvZmZzZXQ9IjEwMCUiIHN0eWxlPSJzdG9wLWNvbG9yOiNlYTU4MG
M7c3RvcC1vcGFjaXR5OjEiIC8+CiAgICA8L2xpbmVhckdyYWRpZW50PgogICAgPGxpbmVhckdyYWR
pZW50IGlkPSJtZWRpdW1HcmFkIiB4MT0iMCUiIHkxPSIwJSIgeDI9IjEwMDAlIj4K
```

ICAgICAgPHN0b3Agb2Zmc2V0PSIwJSIgc3R5bGU9InN0b3AtY29sb3I6I2Y1OWUwYjtzdG9wLW9wY
WNpdHk6MSIgLz4KICAgICAgPHN0b3Agb2Zmc2V0PSIxMDAlIiBzdHlsZT0ic3RvcC1jb2xvcjojZD
k3NzA2O3N0b3Atb3BhY2l0eToxIiAvPgogICAgPC9saW5lYXJHcmFkaWVudD4KICAgIDxsaW5lYXJ
HcmFkaWVudCBpZD0ibG93R3JhZCIgeDE9IjAlIiB5MT0iMCUiIHgyPSIxMDAlIiB5Mj0iMTAwJSI+
CiAgICAgIDxzdG9wIG9mZnNldD0iMCUiIHN0eWxlPSJzdG9wLWNvbG9yOiMzYjgyZjY7c3RvcC1vc
GFjaXR5OjEiIC8+CiAgICAgIDxzdG9wIG9mZnNldD0iMTAwJSIgc3R5bGU9InN0b3AtY29sb3I6Iz
I1NjNlYjtzdG9wLW9wYWNpdHk6MSIgLz4KICAgIDwvbGluZWFyR3JhZGllbnQ+CiAgICA8ZmlsdGV
yIGlkPSJzZW5zU2hhZG93Ij4KICAgICAgPGZlRHJvcFNoYWRvdyBkeD0iMSIgZHk9IjEiIHN0ZERl
dmlhdGlvbj0iMiIgZmxvb2Qtb3BhY2l0eT0iMC4xNSIvPgogICAgPC9maWx0ZXI+CiAgPC9kZWZzP
goKICA8IS0tIEJhY2tncm91bmQgLS0+CiAgPHJlY3Qgd2lkdGg9IjgwMCIgaGVpZ2h0PSIzNDAiIG
ZpbGw9IiNmOGY5ZmEiIHJ4PSIxMCIvPgoKICA8IS0tIFRpdGxlIC0tPgogIDx0ZXh0IHg9IjQwMCI
geT0iMjgiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxOCIgZm9u
dC13ZWlnaHQ9ImJvbGQiIGZpbGw9IiMzMzMiIHRleHQtYW5jaG9yPSJtaWRkbGUiPlNlbnNpdGl2Z
SBEYXRhIENsYXNzaWZpY2F0aW9uIE1hdHJpeDwvdGV4dD4KICA8dGV4dCB4PSI0MDAiIHk9IjQ4Ii
Bmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTEiIGZpbGw9IiM2NjY
iIHRleHQtYW5jaG9yPSJtaWRkbGUiPlBJSS9QSEkvUENJIGRhdGEgdHlwZXMgcmVxdWlyaW5nIG1h
c2tpbmcgaW4gT3BlblBpcGVsaW5lPC90ZXh0PgoKICA8IS0tIENyaXRpY2FsIFJpc2sgLS0+CiAgP
HJlY3QgeD0iMzAiIHk9IjcwIiB3aWR0aD0iMTc1IiBoZWlnaHQ9IjI1MCIgcng9IjEwIiBmaWxsPS
IjZmZmIiBzdHJva2U9IiNlZjQ0NDQiIHN0cm9rZS13aWR0aD0iMiIvPgogIDxyZWN0IHg9IjMwIiB
5PSI3MCIgd2lkdGg9IjE3NSIgaGVpZ2h0PSIzNSIgcng9IjEwIiBmaWxsPSJ1cmwoI2NyaXRpY2Fs
R3JhZCkiLz4KICA8dGV4dCB4PSIxMTciIHk9Ijk1IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc
2VyaWYiIGZvbnQtc2l6ZT0iMTIiIGZvbnQtd2VpZ2h0PSJib2xkIiBmaWxsPSJ3aGl0ZSIgdGV4dC
1hbmNob3I9Im1pZGRsZSI+Q1JJVElDQUw8L3RleHQ+CgogIDx0ZXh0IHg9IjUwIiB5PSIxMzAiIGZ
vbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMSIgZm9udC13ZWlnaHQ9
ImJvbGQiIGZpbGw9IiNkYzI2MjYiPkNyZWRpdCBDYXJkPC90ZXh0PgogIDx0ZXh0IHg9IjUwIiB5P
SIxNDUiIGZvbnQtZmFtaWx5PSJtb25vc3BhY2UiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM2NDc0OG
IiPjQxMTEtMTExMS0xMTExLTExMTE8L3RleHQ+CiAgPHRleHQgeD0iNTAiIHk9IjE2MCIgZm9udC1
mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSIjN2YxZDFkIj7i
hpIgTFVITiBwYXR0ZXJuPC90ZXh0PgoKICA8dGV4dCB4PSI1MCIgeT0iMTg1IiBmb250LWZhbWlse
T0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTEiIGZvbnQtd2VpZ2h0PSJib2xkIiBmaW
xsPSIjZGMyNjI2Ij5TU048L3RleHQ+CiAgPHRleHQgeD0iNTAiIHk9IjIwMCIgZm9udC1mYW1pbHk
9Im1vbm9zcGFjZSIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzY0NzQ4YiI+MTIzLTQ1LTY3ODk8L3Rl
eHQ+CiAgPHRleHQgeD0iNTAiIHk9IjIxNSIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmI
iBmb250LXNpemU9IjEwIiBmaWxsPSIjN2YxZDFkIj7ihpIgTk5OLU5OLU5OTk48L3RleHQ+CgogID
x0ZXh0IHg9IjUwIiB5PSIyNDAiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1
zaXplPSIxMSIgZm9udC13ZWlnaHQ9ImJvbGQiIGZpbGw9IiNkYzI2MjYiPkJhbmsgQWNjb3VudDwv
dGV4dD4KICA8dGV4dCB4PSI1MCIgeT0iMjU1IiBmb250LWZhbWlseT0ibW9ub3NwYWNlIiBmb250L
XNpemU9IjEwIiBmaWxsPSIjNjQ3NDhiIj4wMjEwMDAwMjEvMTIzNDU2Nzg8L3RleHQ+CgogIDx0ZX
h0IHg9IjUwIiB5PSIyODAiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXp
lPSIxMSIgZm9udC13ZWlnaHQ9ImJvbGQiIGZpbGw9IiNkYzI2MjYiPlBISSBNZWRpY2FsPC90ZXh0
PgogIDx0ZXh0IHg9IjUwIiB5PSIyOTUiIGZvbnQtZmFtaWx5PSJtb25vc3BhY2UiIGZvbnQtc2l6Z
T0iMTAiIGZpbGw9IiM2NDc0OGIiPk1STiwgZGlhZ25vc2VzPC90ZXh0PgoKICA8IS0tIEhpZ2ggUm
lzayAtLT4KICA8cmVjdCB4PSIyMjAiIHk9IjcwIiB3aWR0aD0iMTc1IiBoZWlnaHQ9IjI1MCIgcng
9IjEwIiBmaWxsPSIjZmZmIiBzdHJva2U9IiNmOTczMTYiIHN0cm9rZS13aWR0aD0iMiIvPgogIDxy
ZWN0IHg9IjIyMCIgeT0iNzAiIHdpZHRoPSIxNzUiIGhlaWdodD0iMzUiIHJ4PSIxMCIgZmlsbD0id
XJsKCNoaWdoR3JhZCkiLz4KICA8dGV4dCB4PSIzMDciIHk9Ijk1IiBmb250LWZhbWlseT0iQXJpYW
wsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTIiIGZvbnQtd2VpZ2h0PSJib2xkIiBmaWxsPSJ3aGl
0ZSIgdGV4dC1hbmNob3I9Im1pZGRsZSI+SElHSDwvdGV4dD4KCiAgPHRleHQgeD0iMjQwIiB5PSIx

```
MzAiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMSIgZm9udC13Z
WlnaHQ9ImJvbGQiIGZpbGw9IiNlYTU4MGMiPkVtYWlsIEFkZHJlc3M8L3RleHQ+CiAgPHRleHQgeD
0iMjQwIiB5PSIxNDUiIGZvbnQtZmFtaWx5PSJtb25vc3BhY2UiIGZvbnQtc2l6ZT0iMTAiIGZpbGw
9IiM2NDc0OGIiPnVzZXJAZXhhbXBsZS5jb208L3RleHQ+CiAgPHRleHQgeD0iMjQwIiB5PSIxNjAi
IGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzlhM
zQxMiI+4oaSIEVNQUlMIHBhdHRlcm48L3RleHQ+CgogIDx0ZXh0IHg9IjI0MCIgeT0iMTg1IiBmb2
50LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtd2VpZ2h0PSJ
ib2xkIiBmaWxsPSIjZWE1ODBjIj5QaG9uZSBOdW1iZXI8L3RleHQ+CiAgPHRleHQgeD0iMjQwIiB5
PSIyMDAiIGZvbnQtZmFtaWx5PSJtb25vc3BhY2UiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM2NDc0O
GIiPisxLTU1NS0xMjMtNDU2NzwvdGV4dD4KICA8dGV4dCB4PSIyNDAiIHk9IjIxNSIgZm9udC1mYW
1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSIjOWEzNDEyIj7ihpI
gUEhPTkUgcGF0dGVybjwvdGV4dD4KICAgPHRleHQgeD0iMjQwIiB5PSIyNDAiIGZvbnQtZmFtaWx5
PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMSIgZm9udC13ZWlnaHQ9ImJvbGQiIGZpb
Gw9IiNlYTU4MGMiPkFQSSBLZXlzPC90ZXh0PgogIDx0ZXh0IHg9IjI0MCIgeT0iMjU1IiBmb250LW
ZhbWlseT0ibW9ub3NwYWNlIiBmb250LXNpemU9IjEwIiBmaWxsPSIjNjQ3NDhiIj5za19saXZlX2F
iYzEyMy4uLjwvdGV4dD4KICAgPHRleHQgeD0iMjQwIiB5PSIyODAiIGZvbnQtZmFtaWx5PSJBcmlh
bCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMSIgZm9udC13ZWlnaHQ9ImJvbGQiIGZpbGw9IiNlY
TU4MGMiPlBhc3N3b3JkczwvdGV4dD4KICA8dGV4dCB4PSIyNDAiIHk9IjI5NSIgZm9udC1mYW1pbH
k9Im1vbm9zcGFjZSIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzY0NzQ4YiI+cGFzc3dvcmQ9c2VjcmV
0PC90ZXh0PgoKICA8IS0tIE1lZGl1bSBBcHAgIC0tPgogIDxyZWN0IHg9IjMyMCIgeT0iNDAiIH
ZHRoPSIxNzUiIGhlaWdodD0iMjUwIiByeD0iMTAiIGZpbGw9IiNmZmYiIHN0cm9rZT0iI2Y1OWUwY
iIgc3Ryb2tlLXdpZHRoPSIyIi8+CiAgPHJlY3QgeD0iMzQwIiB5PSI3NSIgd2lkdGg9IjEzNSIgG
VpZ2h0PSIzNSIgcng9IjEwIiBmaWxsPSJ1cmwoI21lZGl1bUdyYWQpIi8+CiAgPHRleHQgeD0iND
3IiB5PSI5NSIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEyIiBm
b250LXdlaWdodD0iYm9sZCIgZmlsbD0id2hpdGUiIHRleHQtYW5jaG9yPSJtaWRkbGUiPk1FRElVT
TwvdGV4dD4KICAgPHRleHQgeD0iNDMwIiB5PSIxMzAiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy
1zZXJpZiIgZm9udC1zaXplPSIxMSIgZm9udC13ZWlnaHQ9ImJvbGQiIGZpbGw9IiNkOTc3MDYiPkl
QIEFkZHJlc3M8L3RleHQ+CiAgPHRleHQgeD0iNDMwIiB5PSIxNDUiIGZvbnQtZmFtaWx5PSJtb25v
c3BhY2UiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM2NDc0OGIiPjE5Mi4xNjguMS4xMDA8L3RleHQ+C
iAgPHRleHQgeD0iNDMwIiB5PSIxNjAiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm
9udC1zaXplPSIxMCIgZmlsbD0iIzkyNDAwZSI+4oaSIElQQUREUiBtYXRjaGVyPC90ZXh0PgogIDx
8dGV4dCB4PSIzMzAiIHk9IjE4NSIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250
LXNpemU9IjEwIiBmaWxsPSIjZExiBmb250LXdlaWdodD0iYm9sZCIgZmlsbD0iI2Q5NzcwNiI+VXNlciBJRDwvdGV4d
D4KICA8dGV4dCB4PSIzMzAiIHk9IjIwMCIgZm9udC1mYW1pbHk9Im1vbm9zcGFjZSIgZm9udC1zaX
plPSIxMCIgZmlsbD0iIzY0NzQ4YiI+dXNlcl9pZD0xMjM0NTwvdGV4dD4KICA8dGV4dCB4PSIzMzA
iIHk9IjIyMCIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBm
aWxsPSIjOWEzNDEyIj7ihpIgUEhPTkUgcGF0dGVybjwvdGV4dD4KICAgPHRleHQgeD0iMzMwIiB5P
SIyMzUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMSIgZm9udC
13ZWlnaHQ9ImJvbGQiIGZpbGw9IiNkOTc3MDYiPlNlc3Npb24gSUQ8L3RleHQ+CiAgPHRleHQgeD0
iMzMwIiB5PSIyNDAiIGZvbnQtZmFtaWx5PSJtb25vc3BhY2UiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9
IiM2NDc0OGIiPnNlc3M9YWJjMTIzZGVmPC90ZXh0PgoKICA8dGV4dCB4PSIzMzAiIHk9IjI2NSIgZ
m9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSIjOWEzND
EyIj7ihpIgUEhPTkUgcGF0dGVybjwvdGV4dD4KICAgPHRleHQgeD0iMzMwIiB5PSIyODAiIGZvbnQ
tZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMSIgZm9udC13ZWlnaHQ9Im
JvbGQiIGZpbGw9IiNkOTc3MDYiPlNlc3Npb24gSUQ8L3RleHQ+CiAgPHRleHQgeD0iMzMwIiB5PSI
zMDUiIGZvbnQtZmFtaWx5PSJtb25vc3BhY2UiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM2NDc0OGIi
PmFjY3Q9MDAwMTIzNDU2PC90ZXh0PgogIDx0ZXh0IHg9IjMzMCIgeT0iMzIwIiBmb250LWZhbWlsa
T0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM5MjQwMGUiPuKGkiBQ
eyZWN0IHg9IjIyMCIgeT0iNDAiIHdpZHRoPSIxNzUiIGhlaWdodD0iMjUwIiByeD0iMTAiIGZpbG9
IiNmZmYiIHN0cm9rZT0iIzNiODJmNiIgc3Ryb2tlLXdpZHRoPSIyIi8+CiAgPHJlY3QgeD0iMjQw
3MCIgd2lkdGg9IjEzNSIgaGVpZ2h0PSIzNSIgcng9IjEwIiBmaWxsPSJ1cmwoI21lZGl1bUdyYW
QpIi8+CiAgPHRleHQgeD0iMzA3IiB5PSI5NSIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcml
mIiBmb250LXNpemU9IjEyIiBmb250LXdlaWdodD0iYm9sZCIgZmlsbD0id2hpdGUiIHRleHQtYW5j
```

```
aG9yPSJtaWRkbGUiPkxPVzwvdGV4dD4KCiAgPHRleHQgeD0iNjIwIiB5PSIxMzAiIGZvbnQtZmFta
Wx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMSIgZm9udC13ZWlnaHQ9ImJvbGQiIG
ZpbGw9IiMyNTYzZWIiPlVzZXJuYW1lPC90ZXh0PgogIDx0ZXh0IHg9IjYyMCIgeT0iMTQ1IiBmb25
0LWZhbWlseT0ibW9ub3NwYWNlIiBmb250LXNpemU9IjEwIiBmaWxsPSIjNjQ3NDhiIj5qc21pdGg8
L3RleHQ+CiAgPHRleHQgeD0iNjIwIiB5PSIxNjAiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZ
XJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzFlM2E4YSI+Q29udGV4dC1kZXBlbmRlbnQ8L3RleH
Q+CogIDx0ZXh0IHg9IjYyMCIgeT0iMTg1IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWY
iIGZvbnQtc2l6ZT0iMTEiIGZvbnQtd2VpZ2h0PSJib2xkIiBmaWxsPSIjMjU2M2ViIj5UcmFuc2Fj
dGlvbiBJRDwvdGV4dD4KICA8dGV4dCB4PSI2MjAiIHk9IjIwMCIgZm9udC1mYW1pbHk9Im1vbm9zc
GFjZSIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzY0NzQ4YiI+dHhuX2FiYzEyMzwvdGV4dD4KICAgPH
RleHQgeD0iNjIwIiB5PSIyMjUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1
zaXplPSIxMSIgZm9udC13ZWlnaHQ9ImJvbGQiIGZpbGw9IiMyNTYzZWIiPlJlcXVlc3QgSUQ8L3Rl
eHQ+CiAgPHRleHQgeD0iNjIwIiB5PSIyNDAiIGZvbnQtZmFtaWx5PSJtb25vc3BhY2UiIGZvbnQtc
2l6ZT0iMTAiIGZpbGw9IiM2NDc0OGIiPnJlcS11dWlkLXZhbHVlPC90ZXh0PgoKICA8dGV4dCB4PS
I2MjAiIHk9IjI2NSIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjE
xIiBmb250LXdlaWdodD0iYm9sZCIgZmlsbD0iIzI1NjNlYiI+SG9zdG5hbWVzPC90ZXh0PgogIDx0
ZXh0IHg9IjYyMCIgeT0iMjgwIiBmb250LWZhbWlseT0ibW9ub3NwYWNlIiBmb250LXNpemU9IjEwI
iBmaWxsPSIjNjQ3NDhiIj5zZXJ2ZXIwMS5pbnQ8L3RleHQ+CogIDwhLS0gTGVnZW5kIC0tPgogID
xyZWN0IHg9IjMwIiB5PSIzMjUiHdpZHRoPSI3NDAiGhlaWdodD0iMTAiIHJ4PSIyIiBmaWxsPSI
jZTJlOGYwIi8+CiAgPHJlY3QgeD0iMzAiIHk9IjMyNSIgd2lkdGg9IjE4NSIgaGVpZ2h0PSIxMCIg
cng9IjIiIGZpbGw9InVybCgjY3JpdGljYWxHcmFkKSIvPgogIDx0ZXh0IHg9IjIyNSIgeT0iMzI1I
iB3aWR0aD0iMTg1IiBoZWlnaHQ9IjEyeD0iMiIgZmlsbD0idXJsKCNoaWdoR3JhZCkiLz4KICA8cmVjdCB4PSI0MDAiIHk9IjMyNSIgd2lkdGg9IjE4NSIgaGVpZ2h0PSIxMCIgcng9IjIiIGZpbGw9InVybCgjbWVkaXVtR3JhZCkiLz4KICA8cmVjdCB4PSI1ODUiIHk9IjMyNSIgd2lkdGg9IjE4NSIgaGVpZ2h0PSIxMCIgcng9IjIiIGZpbGw9InVybCgjbG93R3JhZCkiLz4KPC9zdmc+Cg==)



### Common Sensitive Data Patterns

| Data Type | Pattern Example | Risk Level |
|-----------|-----------------|------------|
| Email addresses | `user@domain.com` | Medium |
| Credit cards | `4111-1111-1111-1111` | **Critical** |
| Social Security | `123-45-6789` | **Critical** |
| API keys | `sk_live_abc123...` | **Critical** |
| IP addresses | `192.168.1.100` | Low-Medium |
| Phone numbers | `+1-555-123-4567` | Medium |
| JWT tokens | `eyJhbG...` | High |

```python
// Search for potential email addresses in logs
fetch logs, from: now() - 1h
| filter contains(content, "@")
| filter NOT contains(content, "@dynatrace")
| filter NOT contains(content, "@example")
| fieldsAdd content_preview = substring(content, from: 0, to: 150)
| summarize {count = count()}, by: {content_preview, k8s.namespace.name}
```
```

```
| sort count desc
| limit 20
```

```python
// Search for potential API keys or tokens
fetch logs, from: now() - 1h
| filter contains(content, "key=")
        OR contains(content, "token=")
        OR contains(content, "api_key")
        OR contains(content, "apikey")
        OR contains(content, "secret")
| fieldsAdd content_preview = substring(content, from: 0, to: 120)
| summarize {count = count()}, by: {content_preview, k8s.namespace.name}
| sort count desc
| limit 20
```

```python
// Search for potential JWT tokens
fetch logs, from: now() - 1h
| filter contains(content, "eyJ")  // JWT typically starts with eyJ
| fieldsAdd content_preview = substring(content, from: 0, to: 100)
| summarize {count = count()}, by: {content_preview, k8s.namespace.name}
| sort count desc
| limit 15
```

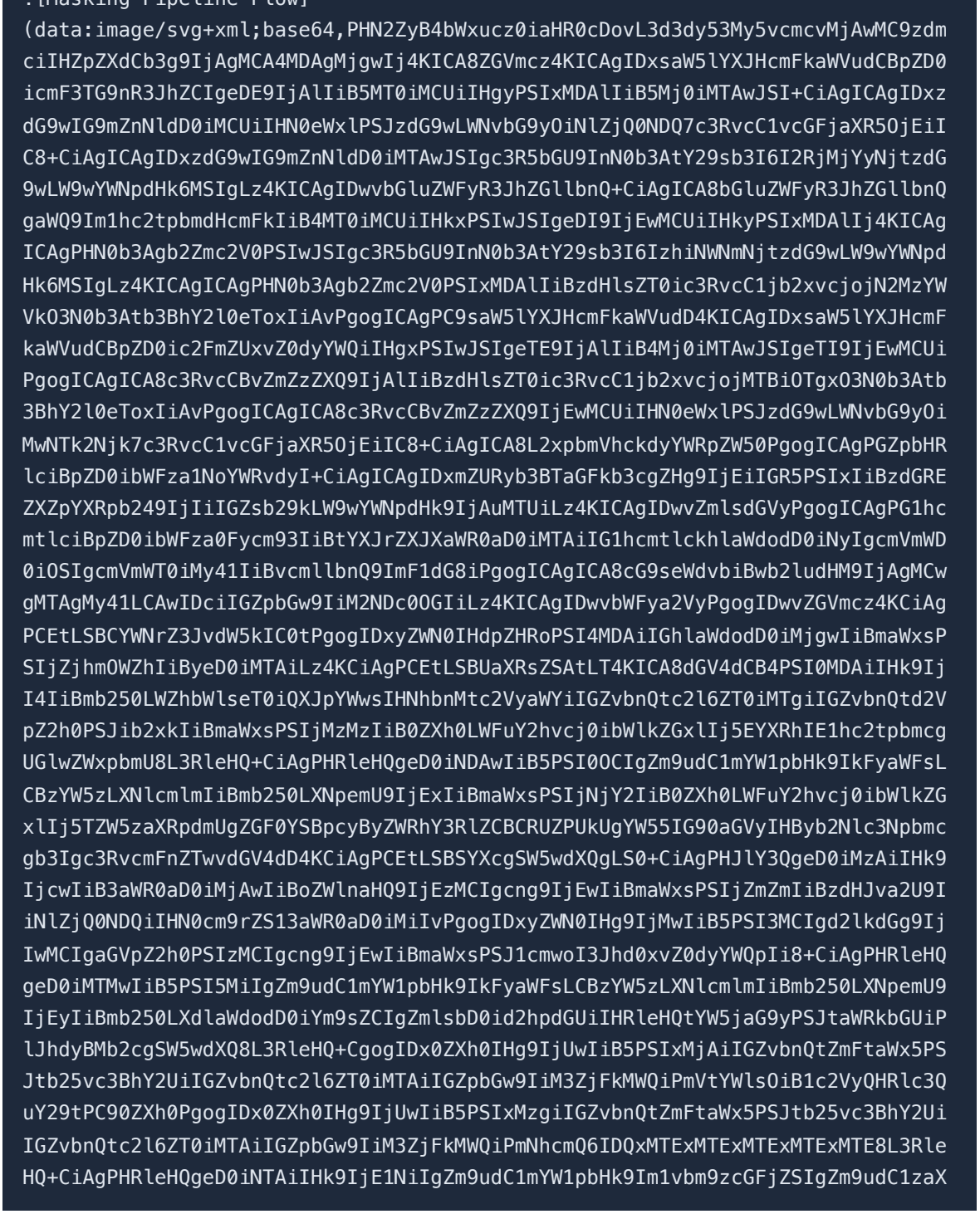
```python
// Search for potential credit card patterns (16 digits)
fetch logs, from: now() - 1h
| filter contains(content, "card")
        OR contains(content, "payment")
        OR contains(content, "visa")
        OR contains(content, "mastercard")
| fieldsAdd content_preview = substring(content, from: 0, to: 120)
| summarize {count = count()}, by: {content_preview}
| sort count desc
| limit 15
```

```python
// Search for password-related entries (should never be logged!)
fetch logs, from: now() - 1h
| filter contains(content, "password")
        OR contains(content, "passwd")
        OR contains(content, "pwd=")
| fieldsAdd content_preview = substring(content, from: 0, to: 100)
```

```
| summarize {count = count()}, by: {content_preview, k8s.namespace.name}
| sort count desc
| limit 20
```

2. OpenPipeline Masking Configuration

OpenPipeline provides built-in masking processors to protect sensitive data at ingestion time.



plPSIxMCIgZmlsbD0iIzdmMWQxZCI+c3NuOiAxMjMtNDUtNjc4OTwvdGV4dD4KICA8dGV4dCB4PSI
1MCIgeT0iMTc0IiBmb250LWZhbWlseT0ibW9ub3NwYWNlIiBmb250LXNpemU9IjEwIiBmaWxsPSIj
N2YxZDFkIj5pcDogMTkyLjE2OC4xLjEwMDwvdGV4dD4KICA8dGV4dCB4PSI1MCIgeT0iMTkyIiBmb
250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiNkYzI2Mj
YiIGZvbnQtd2VpZ2h0PSJib2xkIj5Db250YWlucyBQSUkhPC90ZXh0PgoKICA8IS0tIEFycm93IDE
gLS0+CiAgPHBhdGggZD0iTTIzMCwxMzUgTDI2MCwxMzUiIHN0cm9rZT0iIzY0NzQ4YiIgc3Ryb2tl
LXdpZHRoPSIyIiBmaWxsPSJub25lIiBtYXJrZXItZW5kPSJ1cmwoI21hc2tBcnJvdykiLz4KCiAgP
CEtLSBNYXNraW5nIFByb2Nlc3NvciAtLT4KICA8cmVjdCB4PSIyNzAiIHk9IjcwIiB3aWR0aD0iMj
YwIiBoZWlnaHQ9IjEzMCIgcng9IjEwIiBmaWxsPSIjZmZmIiBzdHJva2U9IiM4YjVjZjYiIHN0cm9
rZS13aWR0aD0iMiIvPgogIDxyZWN0IHg9IjI3MCIgeT0iNzAiIHdpZHRoPSIyNjAiIGhlaWdodD0i
MzAiIHJ4PSIxMCIgZmlsbD0idXJsKCNtYXNraW5nR3JhZCkiLz4KICA8dGV4dCB4PSI0MDAiIHk9I
jkyIiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTIiIGZvbnQtd2
VpZ2h0PSJib2xkIiBmaWxsPSJ3aGl0ZSIgdGV4dC1hbmNob3I9Im1pZGRsZSI+TWFza2luZyBQcm9
jZXNzb3IgKEZpcnN0IFN0YWdlKTwvdGV4dD4KCiAgPHRleHQgeD0iMjkwIiB5PSIxMjAiIGZvbnQt
ZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZm9udC13ZWlnaHQ9ImJvb
GQiIGZpbGw9IiM2ZDI4ZDkiPlBhdHRlcm4gTWF0Y2hpbmc6PC90ZXh0PgogIDx0ZXh0IHg9IjI5MC
IgeT0iMTM4IiBmb250LWZhbWlseT0ibW9ub3NwYWNlIiBmb250LXNpemU9IjEwIiBmaWxsPSIjNjQ
3NDhiIj5FTUFJTCDihpIgW0VNQUlMX1JFREFDVEVEXTwvdGV4dD4KICA8dGV4dCB4PSIyOTAiIHk9
IjE1NiIgZm9udC1mYW1pbHk9Im1vbm9zcGFjZSIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzY0NzQ4Y
iI+TFVITiDihpIgW0NDX01BU0tFRF08L3RleHQ+CiAgPHRleHQgeD0iMjkwIiB5PSIxNzQiIGZvbn
QtZmFtaWx5PSJtb25vc3BhY2UiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM2NDc0OGIiPlNTTiDihpI
gWFhYLVhYLVhYWFg8L3RleHQ+CiAgPHRleHQgeD0iMjkwIiB5PSIxOTIiIGZvbnQtZmFtaWx5PSJt
b25vc3BhY2UiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM2NDc0OGIiPklQQUREUiDihpIgW0lQX0hJR
ERFTl08L3RleHQ+CgogIDwhLS0gQXJyb3cgMiAtLT4KICA8cGF0aCBkPSJNNTMwLDEzNSBMNTYwLD
EzNSIgc3Ryb2tlPSIjNjQ3NDhiIiBzdHJva2Utd2lkdGg9IjIiIGZpbGw9Im5vbmUiIG1hcmtlci1
lbmQ9InVybCgjbWFza0Fycm93KSIvPgoKICA8IS0tIFNhZmUgT3V0cHV0IC0tPgogIDxyZWN0IHg9
IjU3MCIgeT0iNzAiIHdpZHRoPSIyMDAiIGhlaWdodD0iMTMwIiByeD0iMTAiIGZpbGw9IiNmZmYiI
HN0cm9rZT0iIzEwYjk4MSIgc3Ryb2tlLXdpZHRoPSIyIi8+CiAgPHJlY3QgeD0iNTcwIiB5PSI3MC
Igd2lkdGg9IjIwMCIgaGVpZ2h0PSIzMCIgcng9IjEwIiBmaWxsPSJ1cmwoI3NhZmVMb2dHcmFkKSI
vPgogIDx0ZXh0IHg9IjY3MCIgeT0iOTIiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIg
Zm9udC1zaXplPSIxMiIgZm9udC13ZWlnaHQ9ImJvbGQiIGZpbGw9IndoaXRlIiB0ZXh0LWFuY2hvc
j0ibWlkZGxlIj5TYWZlIGZvciBTdG9yYWdlPC90ZXh0PgoKICA8dGV4dCB4PSI1OTAiIHk9IjEyMC
IgZm9udC1mYW1pbHk9Im1vbm9zcGFjZSIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzA0Nzg1NyI+ZW1
haWw6IFtFTUFJTF9SRURBQ1RFRF08L3RleHQ+CiAgPHRleHQgeD0iNTkwIiB5PSIxMzgiIGZvbnQt
ZmFtaWx5PSJtb25vc3BhY2UiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiMwNDc4NTciPmNhcmQ6IFtDQ
19NQVNLRURdPC90ZXh0PgogIDx0ZXh0IHg9IjU5MCIgeT0iMTU2IiBmb250LWZhbWlseT0ibW9ub3
NwYWNlIiBmb250LXNpemU9IjEwIiBmaWxsPSIjMDQ3ODU3Ij5zc246IFhYWC1YWC1YWFhYPC90ZXh
0PgogIDx0ZXh0IHg9IjU5MCIgeT0iMTc0IiBmb250LWZhbWlseT0ibW9ub3NwYWNlIiBmb250LXNp
emU9IjEwIiBmaWxsPSIjMDQ3ODU3Ij5pcDogW0lQX0hJRERFTl08L3RleHQ+CiAgPHRleHQgeD0iN
TkwIiB5PSIxOTIiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMC
IgZmlsbD0iIzA1OTY2OSIgZm9udC13ZWlnaHQ9ImJvbGQiPlBJSSBQcm90ZWN0ZWQ8L3RleHQ+Cgo
gIDwhLS0gQmVuZWZpdHMgU2VjdGlvbiAtLT4KICA8cmVjdCB4PSIzMCIgeT0iMjE1IiB3aWR0aD0i
NzQwIiBoZWlnaHQ9IjUwIiByeD0iMTAiIGZpbGw9IiNmZmYiIHN0cm9rZT0iI2UyZThmMCIgc3Ryb
2tlLXdpZHRoPSIyIi8+CiAgPHRleHQgeD0iNDAwIiB5PSIyNDAiIGZvbnQtZmFtaWx5PSJBcmlhbC
wgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMiIgZm9udC13ZWlnaHQ9ImJvbGQiIGZpbGw9IiMzMzM
iIHRleHQtYW5jaG9yPSJtaWRkbGUiPktleSBCZW5lZml0czwvdGV4dD4KCiAgPHJlY3QgeD0iNTAi
IHk9IjI0NyIgd2lkdGg9IjE0NSIgaGVpZ2h0PSIxMiIgcng9IjIiIGZpbGw9IiNkMWZhZTUiLz4KI
CA8dGV4dCB4PSIxMjIiIHk9IjI1NyIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb2

50LXNpemU9IjEwIiBmaWxsPSIjMDQ3ODU3IiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5HRFBSIENvbXB
saWFudDwvdGV4dD4KCiAgPHJlY3QgeD0iMjEwIiB5PSIyNDciIHdpZHRoPSIxNDUiIGhlaWdodD0i
MTIiIHJ4PSIyIiBmaWxsPSIjZGJlYWZlIi8+CiAgPHRleHQgeD0iMjgyIiB5PSIyNTciIGZvbnQtZ
mFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzFlNDBhZiIgdG
V4dC1hbmNob3I9Im1pZGRsZSI+SElQQUEgUmVhZHk8L3RleHQ+CgogIDxyZWN0IHg9IjM3MCIgeT0
iMjQ3IiB3aWR0aD0iMTQ1IiBoZWlnaHQ9IjEyIiByeD0iMiIgZmlsbD0iI2ZlZjNjNyIvPgogIDx0
ZXh0IHg9IjM4MiIgeT0iMjU3IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc
2l6ZT0iMTAiIGZpbGw9IiM5MjQwMGUiIHRleHQtYW5jaG9yPSJtaWRkbGUiPlBDSS1EU1MgU2FmZT
wvdGV4dD4KCiAgPHJlY3QgeD0iNTMwIiB5PSIyNDciIHdpZHRoPSIyMjAiIGhlaWdodD0iMTIiIHJ
4PSIyIiBmaWxsPSIjZWRlOWZlIi8+CiAgPHRleHQgeD0iNjQwIiB5PSIyNTciIGZvbnQtZmFtaWx5
PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzZkMjhkOSIgdGV4dC1hb
mNob3I9Im1pZGRsZSI+TmV2ZXIgU3RvcmVkIFVubWFza2VkPC90ZXh0Pgo8L3N2Zz4K)

Masking Processor Types

Processor	Use Case
Mask value	Replace with fixed pattern (e.g., `***MASKED***`)
Hash value	One-way hash for correlation without exposure
Remove field	Completely remove the field
Pattern mask	Partial masking (e.g., `****1234`)

Configuration Path

1. Settings → Log processing → OpenPipeline
2. Select or create pipeline
3. Add "Mask value" or "Hash value" processor
4. Configure field and pattern

DPL Patterns for Masking

```
# Email pattern
[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}

# Credit card pattern
\b(?:\d[ -]*?){13,16}\b

# SSN pattern (US)
\b\d{3}-\d{2}-\d{4}\b

# IP address pattern
\b(?:\d{1,3}\.){3}\d{1,3}\b
```

```python
```

```
// Verify masking is working (look for masked patterns)
fetch logs, from: now() - 1h
| filter contains(content, "MASKED")
        OR contains(content, "***")
        OR contains(content, "[REDACTED]")
| fieldsAdd content_preview = substring(content, from: 0, to: 120)
| summarize {count = count()}, by: {content_preview}
| sort count desc
| limit 15
```

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

3. IP Address Analysis

IP addresses may require masking depending on your compliance requirements.

```python
// Find logs containing IP addresses
fetch logs, from: now() - 1h
| parse content, "IPADDR:ip_found"
| filter isNotNull(ip_found)
| summarize {count = count()}, by: {ip_found, k8s.namespace.name}
| sort count desc
| limit 20
```

```python
// Classify IP addresses (internal vs external)
fetch logs, from: now() - 1h
| parse content, "IPADDR:ip_found"
| filter isNotNull(ip_found)
| fieldsAdd ip_type = if(startsWith(ip_found, "10."), "RFC1918-10",
                    else: if(startsWith(ip_found, "192.168."), "RFC1918-
192",
                    else: if(startsWith(ip_found, "172."), "RFC1918-172",
                    else: if(startsWith(ip_found, "127."), "LOOPBACK",
                    else: "EXTERNAL"))))
| summarize {count = count()}, by: {ip_type}
| sort count desc
```

4. Security Event Monitoring

````
Monitor logs for security-relevant events.

```python
// Authentication-related logs
fetch logs, from: now() - 1h
| filter contains(content, "login")
        OR contains(content, "auth")
        OR contains(content, "signin")
        OR contains(content, "logout")
| fieldsAdd content_preview = substring(content, from: 0, to: 100)
| summarize {count = count()}, by: {content_preview, k8s.namespace.name}
| sort count desc
| limit 20
```

```python
// Failed authentication attempts
fetch logs, from: now() - 1h
| filter (contains(content, "failed") OR contains(content, "denied") OR
contains(content, "unauthorized"))
        AND (contains(content, "login") OR contains(content, "auth") OR
contains(content, "access"))
| fieldsAdd content_preview = substring(content, from: 0, to: 120)
| summarize {count = count()}, by: {content_preview, k8s.namespace.name}
| sort count desc
| limit 20
```

```python
// Security-related errors over time
fetch logs, from: now() - 24h
| filter loglevel == "ERROR"
| filter contains(content, "security")
        OR contains(content, "unauthorized")
        OR contains(content, "forbidden")
        OR contains(content, "denied")
| makeTimeseries {security_errors = count()}, interval: 30m
```

```python
// Access pattern anomalies - hourly access distribution
fetch logs, from: now() - 24h
| filter contains(content, "access") OR contains(content, "request")
| fieldsAdd hour_bucket = bin(timestamp, 1h)
| summarize {access_count = count()}, by: {hour_bucket, k8s.namespace.name}
| sort hour_bucket asc
```
````

5. Audit Log Queries

Track administrative and configuration changes.

```python
// Administrative action logs
fetch logs, from: now() - 24h
| filter contains(content, "created")
        OR contains(content, "deleted")
        OR contains(content, "updated")
        OR contains(content, "modified")
| filter contains(content, "user")
        OR contains(content, "admin")
        OR contains(content, "config")
| fieldsAdd content_preview = substring(content, from: 0, to: 100)
| summarize {
    count = count(),
    first_seen = min(timestamp),
    last_seen = max(timestamp)
  }, by: {content_preview, k8s.namespace.name}
| sort count desc
| limit 20
```

```python
// Configuration change events
fetch logs, from: now() - 24h
| filter contains(content, "config")
        OR contains(content, "setting")
        OR contains(content, "environment")
| filter contains(content, "change")
        OR contains(content, "update")
        OR contains(content, "reload")
| fieldsAdd content_preview = substring(content, from: 0, to: 120)
| summarize {count = count()}, by: {content_preview, k8s.namespace.name}
| sort count desc
| limit 15
```

```python
// Permission-related events
fetch logs, from: now() - 24h
| filter contains(content, "permission")
        OR contains(content, "role")
        OR contains(content, "privilege")
        OR contains(content, "rbac")
| fieldsAdd content_preview = substring(content, from: 0, to: 100)
```

```
| summarize {count = count()}, by: {content_preview, k8s.namespace.name}
| sort count desc
| limit 15
```

6. Compliance Reporting

```python
// Data access summary
fetch logs, from: now() - 24h
| filter contains(content, "access") OR contains(content, "read") OR
contains(content, "view")
| summarize {
    total_accesses = count(),
    unique_sources = countDistinct(dt.openpipeline.source)
  }, by: {k8s.namespace.name}
| sort total_accesses desc
| limit 15
```

```python
// Log retention verification
fetch logs, from: now() - 7d
| summarize {
    earliest = min(timestamp),
    latest = max(timestamp),
    total_logs = count()
  }, by: {dt.system.bucket}
| fieldsAdd retention_days = (latest - earliest) / 86400000000000
```

```python
// Sensitive field exposure audit
fetch logs, from: now() - 1h
| summarize {
    total = count(),
    with_email_pattern = countIf(contains(content, "@")),
    with_key_pattern = countIf(contains(content, "key=") OR contains(content,
"token=")),
    with_password_ref = countIf(contains(content, "password") OR
contains(content, "passwd"))
  }
| fieldsAdd email_exposure_pct = round((with_email_pattern * 100.0) / total,
decimals: 2)
| fieldsAdd key_exposure_pct = round((with_key_pattern * 100.0) / total,
decimals: 2)
| fieldsAdd password_exposure_pct = round((with_password_ref * 100.0) /
total, decimals: 2)
```

```

## 7. Masking Best Practices

### Do's ✅

1. **Mask at ingestion** - Use OpenPipeline processors to mask before storage
2. **Use hashing** - For fields that need correlation but not visibility
3. **Audit regularly** - Run discovery queries to find new sensitive data patterns
4. **Document patterns** - Keep a registry of masked data types and patterns
5. **Test in staging** - Verify masking works before production deployment

### Don'ts ❌

1. **Don't log passwords** - Ever, even "for debugging"
2. **Don't log full credit cards** - Maximum last 4 digits
3. **Don't rely on application masking alone** - OpenPipeline is your safety net
4. **Don't skip internal data** - Internal IP/usernames may still be sensitive
5. **Don't forget API keys** - Rotate if accidentally logged

### OpenPipeline Masking Configuration Example

```yaml
# OpenPipeline processor configuration
processors:
  - name: mask-emails
    type: mask
    field: content
    pattern: "[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,}"
    replacement: "[EMAIL-MASKED]"

  - name: mask-api-keys
    type: mask
    field: content
    pattern: "(api_key|apikey|key)=[A-Za-z0-9]{20,}"
    replacement: "$1=[KEY-MASKED]"
```

```python
// Security health summary
fetch logs, from: now() - 1h
| summarize {
    total_logs = count(),
    security_events = countIf(contains(content, "security") OR
contains(content, "auth")),
```
```

```
    failed_events = countIf(contains(content, "failed") OR contains(content,
"denied")),
    potential_pii = countIf(contains(content, "@") OR contains(content,
"password"))
  }
| fieldsAdd security_event_rate = round((security_events * 100.0) /
total_logs, decimals: 2)
| fieldsAdd failure_rate = round((failed_events * 100.0) / total_logs,
decimals: 2)
| fieldsAdd pii_risk_rate = round((potential_pii * 100.0) / total_logs,
decimals: 2)
```

📝 Summary

In this notebook, you learned:

✅ **Sensitive data discovery** - Finding PII, tokens, and credentials
✅ **OpenPipeline masking** - Configuration and patterns
✅ **IP address analysis** - Classification and masking needs
✅ **Security monitoring** - Auth events, failures, anomalies
✅ **Audit logging** - Admin actions, config changes
✅ **Compliance reporting** - Data access and exposure audits

➡️ Next Steps

Continue to **OPLOGS-07: Buckets & Cost Optimization** for storage management.

📚 References

- [OpenPipeline Data Masking](https://docs.dynatrace.com/docs/platform/openpipeline/use-cases/log-processing/mask-sensitive-data)
- [Log Security Best Practices](https://docs.dynatrace.com/docs/observe-and-explore/logs/log-management-and-analytics/lma-security)
- [GDPR and Compliance](https://docs.dynatrace.com/docs/manage/data-privacy-and-security)