```
# OPMIG-08: Security, Masking & Compliance

> **Series:** OPMIG | **Notebook:** 8 of 9 | **Created:** December 2025

> **OpenPipeline Migration Series** | Notebook 8 of 9
> **Level:** Advanced
> **Estimated Time:** 80 minutes

---

## Learning Objectives

By completing this notebook, you will:

1. Configure masking processors for PII protection
2. Use built-in and custom masking patterns
3. ⭐ **NEW:** Implement GDPR compliance checklist
4. ⭐ **NEW:** Implement HIPAA compliance checklist
5. ⭐ **NEW:** Implement PCI-DSS compliance checklist
6. ⭐ **NEW:** Implement SOC 2 compliance checklist
7. Validate masking effectiveness
8. Design complete security pipelines

---



---

## Security Stage Overview

Masking executes **first** in the Processing stage, ensuring sensitive data
is redacted before any other processing occurs.

![Masking Executes First]
(
ciIHZpZXdCb3g9IjAgMCA4MDAgMzIwIj4KICA8ZGVmcz4KICAgIDxsaW5lYXJHcmFkaWVudCBpZD0
idW5zYWZlR3JhZCIgeDE9IjAlIiB5MT0iMCUiIHgyPSIxMDAlIiB5Mj0iMCUiPgogICAgICA8c3Rv
cCBvZmZzZXQ9IjAlIiBzdHlsZT0ic3RvcC1jb2xvcjojZWY0NDQ0O3N0b3Atb3BhY2l0eToxIiAvP
gogICAgICA8c3RvcCBvZmZzZXQ9IjEwMCUiIHN0eWxlPSJzdG9wLWNvbG9yOiNkYzI2MjY7c3RvcC
1vcGFjaXR5OjEiIC8+CiAgICA8L2xpbmVhckdyYWRpZW50PgogICAgPGxpbmVhckdyYWRpZW50IGl
kPSJzYWZlR3JhZCIgeDE9IjAlIiB5MT0iMCUiIHgyPSIxMDAlIiB5Mj0iMCUiPgogICAgICA8c3Rv
cCBvZmZzZXQ9IjAlIiBzdHlsZT0ic3RvcC1jb2xvcjojMjJjNTVlO3N0b3Atb3BhY2l0eToxIiAvP
gogICAgICA8c3RvcCBvZmZzZXQ9IjEwMCUiIHN0eWxlPSJzdG9wLWNvbG9yOiMxNmEzNGE7c3RvcC
1vcGFjaXR5OjEiIC8+CiAgICA8L2xpbmVhckdyYWRpZW50PgogICAgPGxpbmVhckdyYWRpZW50IGl
kPSJtYXNraW5nR3JhZCIgeDE9IjAlIiB5MT0iMCUiIHgyPSIxMDAlIiB5Mj0iMTAwJSI+CiAgICAg
IDxzdG9wIG9mZnNldD0iMCUiIHN0eWxlPSJzdG9wLWNvbG9yOiM4YjVjZjY7c3RvcC1vcGFjaXR5O
jEiIC8+CiAgICAgIDxzdG9wIG9mZnNldD0iMTAwJSIgc3R5bGU9InN0b3AtY29sb3I6IzdjM2FlZD
```

tzdG9wLW9wYWNpdHk6MSIgLz4KICAgIDwvbGluZWFyR3JhZGllbnQ+CiAgICA8ZmlsdGVyIGlkPSJ
tb1NoYWRvdyI+CiAgICAgIDxmZURyb3BTaGFkb3cgZHg9IjEiIGR5PSIxIiBzdGREZXZpYXRpb249
IjIiIGZsb29kLW9wYWNpdHk9IjAuMTUiLz4KICAgIDwvZmlsdGVyPgogICAgPG1hcmtlciBpZD0ib
W9BcnJvdyIgbWFya2VyV2lkdGg9IjEwIiBtYXJrZXJIZWlnaHQ9IjciIHJlZlg9IjkiIHJlZlk9Ij
MuNSIgb3JpZW50PSJhdXRvIj4KICAgICAgPHBvbHlnb24gcG9pbnRzPSIwIDAsIDEwIDMuNSwgMCA
3IiBmaWxsPSIjNjQ3NDhiIi8+CiAgICA8L21hcmtlcj4KICA8L2RlZnM+CgogIDwhLS0gQmFja2dy
b3VuZCAtLT4KICA8cmVjdCB3aWR0aD0iODAwIiBoZWlnaHQ9IjMyMCIgZmlsbD0iI2Y4ZjlmYSIgc
ng9IjEwIi8+CgogIDwhLS0gVGl0bGUgLS0+CiAgPHRleHQgeD0iNDAwIiB5PSIyOCIgZm9udC1mYW
1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjE4IiBmb250LXdlaWdodD0iYm9sZCI
gZmlsbD0iIzMzMyIgdGV4dC1hbmNob3I9Im1pZGRsZSI+TWFza2luZyBFeGVjdXRlcyBGaXJzdDwv
dGV4dD4KICA8dGV4dCB4PSI0MDAiIHk9IjQ4IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2Vya
WYiIGZvbnQtc2l6ZT0iMTEiIGZpbGw9IiM2NjYiIHRleHQtYW5jaG9yPSJtaWRkbGUiPlNlbnNpdG
l2ZSBkYXRhIGlzIHByb3RlY3RlZCBiZWZvcmUgYW55IG90aGVyIHByb2Nlc3Npbmc8L3RleHQ+Cgo
gIDwhLS0gSW5wdXQgLS0+CiAgPHJlY3QgeD0iMzAiIHk9Ijc1IiB3aWR0aD0iMTYwIiBoZWlnaHQ9
IjEwMCIgcng9IjgiIGZpbGw9IiNmZWNhY2EiIHN0cm9rZT0iI2VmNDQ0NCIgc3Ryb2tlLXdpZHRoP
SIyIiBmaWx0ZXI9InVybCgjbW9TaGFkb3cpIi8+CiAgPHRleHQgeD0iMTEwIiB5PSIxMDAiIGZvbn
QtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMSIgZm9udC13ZWlnaHQ9ImJ
vbGQiIGZpbGw9IiNkYzI2MjYiIHRleHQtYW5jaG9yPSJtaWRkbGUiPkluY29taW5nIExvZzwvdGV4
dD4KICA8dGV4dCB4PSIxMTAiIHk9IjEyMCIgZm9udC1mYW1pbHk9Im1vbm9zcGFjZSIgZm9udC1za
XplPSIxMCIgZmlsbD0iIzdmMWQxZCIgdGV4dC1hbmNob3I9Im1pZGRsZSI+Y2FyZDogNDExMS0xMT
ExLTExMTEtMTExMTwvdGV4dD4KICA8dGV4dCB4PSIxMTAiIHk9IjEzNSIgZm9udC1mYW1pbHk9Im1
vbm9zcGFjZSIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzdmMWQxZCIgdGV4dC1hbmNob3I9Im1pZGRs
ZSI+ZW1haWw6IHVzZXJAZXhhbXBsZS5jb208L3RleHQ+CiAgPHRleHQgeD0iMTEwIiB5PSIxNTAiI
GZvbnQtZmFtaWx5PSJtb25vc3BhY2UiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM3ZjFkMWQiIHRleH
QtYW5jaG9yPSJtaWRkbGUiPnNzbjogMTIzLTQ1LTY3ODk8L3RleHQ+CiAgPHRleHQgeD0iMTEwIiB
5PSIxNjUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZmls
bD0iI2RjMjYyNiIgdGV4dC1hbmNob3I9Im1pZGRsZSI+4pqg77iPIENvbnRhaW5zIFBJSTwvdGV4d
D4KCiAgPHBhdGggZD0iTTE5MCwxMjUgTDIzMCwxMjUiIHN0cm9rZT0iIzY0NzQ4YiIgc3Ryb2tlLX
dpZHRoPSIyIiBmaWxsPSJub25lIiBtYXJrZXItZW5kPSJ1cmwoI21vQXJyb3cpIi8+CgogIDwhLS0
gU3RlcCAxOiBNYXNraW5nIC0tPgogIDxyZWN0IHg9IjI0MCIgeT0iNzAiIHdpZHRoPSIxNTAiIGhl
aWdodD0iMTEwIiByeD0iOCIgZmlsbD0idXJsKCNtYXNraW5nR3JhZCkiIGZpbHRlcj0idXJsKCNtb
1NoYWRvdykiLz4KICA8dGV4dCB4PSIzMTUiIHk9Ijk1IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbn
Mtc2VyaWYiIGZvbnQtc2l6ZT0iMTEiIGZvbnQtd2VpZ2h0PSJib2xkIiBmaWxsPSJ3aGl0ZSIgdGV
4dC1hbmNob3I9Im1pZGRsZSI+MS4gTUFTS0lORzwvdGV4dD4KICA8dGV4dCB4PSIzMTUiIHk9IjEx
NSIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSJyZ
2JhKDI1NSwyNTUsMjU1LDAuOSkiIHRleHQtYW5jaG9yPSJtaWRkbGUiPkNyZWRpdCBDYXJkczwvdG
V4dD4KICA8dGV4dCB4PSIzMTUiIHk9IjEzMCIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcml
mIiBmb250LXNpemU9IjEwIiBmaWxsPSJyZ2JhKDI1NSwyNTUsMjU1LDAuOSkiIHRleHQtYW5jaG9y
PSJtaWRkbGUiPkVtYWlsczwvdGV4dD4KICA8dGV4dCB4PSIzMTUiIHk9IjE0NSIgZm9udC1mYW1pb
Hk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSJyZ2JhKDI1NSwyNTUsMj
U1LDAuOSkiIHRleHQtYW5jaG9yPSJtaWRkbGUiPlNTTjwvdGV4dD4KICA8dGV4dCB4PSIzMTUiIHk
9IjE2NSIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjEwIiBmaWxs
PSJyZ2JhKDI1NSwyNTUsMjU1LDAuOCkiIHRleHQtYW5jaG9yPSJtaWRkbGUiPlJ1bnMgRklSU1QgY
Wx3YXlzPC90ZXh0PgoKICA8cGF0aCBkPSJNMzkwLDEyNSBMNDIwLDEyNSIgc3Ryb2tlPSIjNjQ3ND
hiIiBzdHJva2Utd2lkdGg9IjIiIGZpbGw9Im5vbmUiIG1hcmtlci1lbmQ9InVybCgjbW9BcnJvdyk
iLz4KCiAgPCEtLSBNYXNrZWQgTG9nIC0tPgogIDxyZWN0IHg9IjQzMCIgeT0iNzUiIHdpZHRoPSIx
NjAiIGhlaWdodD0iMTAwIiByeD0iOCIgZmlsbD0iI2QxZmFlNSIgc3Ryb2tlPSIjMjJjNTVlIiBzd
HJva2Utd2lkdGg9IjIiIGZpbHRlcj0idXJsKCNtb1NoYWRvdykiLz4KICA8dGV4dCB4PSI1MTAiIH

k9IjEwMCIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLXNlcmlmIiBmb250LXNpemU9IjExIiBmb25
0LXdlaWdodD0iYm9sZCIgZmlsbD0iIzA0Nzg1NyIgdGV4dC1hbmNob3I9Im1pZGRsZSI+TWFza2Vk
IExvZzwvdGV4dD4KICA8dGV4dCB4PSI1MTAiIHk9IjEyMCIgZm9udC1mYW1pbHk9Im1vbm9zcGFjZ
SIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzA0Nzg1NyIgdGV4dC1hbmNob3I9Im1pZGRsZSI+Y2FyZD
ogW0NDX1JFREFDVEVEXTwvdGV4dD4KICA8dGV4dCB4PSI1MTAiIHk9IjEzNSIgZm9udC1mYW1pbHk
9Im1vbm9zcGFjZSIgZm9udC1zaXplPSIxMCIgZmlsbD0iIzA0Nzg1NyIgdGV4dC1hbmNob3I9Im1p
ZGRsZSI+ZW1haWw6IFtFTUFJTF9SRURBQ1RFRF08L3RleHQ+CiAgPHRleHQgeD0iNTEwIiB5PSIxN
TAiIGZvbnQtZmFtaWx5PSJtb25vc3BhY2UiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiMwNDc4NTciIH
RleHQtYW5jaG9yPSJtaWRkbGUiPnNzbjogW1NTTl9SRURBQ1RFRF08L3RleHQ+CiAgPHRleHQgeD0
iNTEwIiB5PSIxNjUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIx
MCIgZmlsbD0iIzA0Nzg1NyIgdGV4dC1hbmNob3I9Im1pZGRsZSI+4pyFIFBJSSBQcm90ZWN0ZWQ8L
3RleHQ+CgogIDxwYXRoIGQ9Ik01OTAsMTI1IEw2MjAsMTI1IiBzdHJva2U9IiM2NDc0OGIiIHN0cm
9rZS13aWR0aD0iMiIgZmlsbD0ibm9uZSIgbWFya2VyLWVuZD0idXJsKCNtb0Fycm93KSIvPgoKICA
8IS0tIFJlbWFpbmluZyBTdGFnZXMgLS0+CiAgPHJlY3QgeD0iNjMwIiB5PSI3MCIgd2lkdGg9IjE0
MCIgaGVpZ2h0PSIxMTAiIHJ4PSI4IiBmaWxsPSIjZmZmIiBzdHJva2U9IiNlMmU4ZjAiIHN0cm9rZ
S13aWR0aD0iMiIvPgogIDx0ZXh0IHg9IjcwMCIgeT0iOTUiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2
Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZm9udC13ZWlnaHQ9ImJvbGQiIGZpbGw9IiMzMzMiIHR
leHQtYW5jaG9yPSJtaWRkbGUiPjItNi4gUmVtYWluaW5nnPC90ZXh0PgogIDx0ZXh0IHg9IjcwMCIg
eT0iMTE1IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpb
Gw9IiM2NjYiIHRleHQtYW5jaG9yPSJtaWRkbGUiPkRyb3AgKHNhZmUpPC90ZXh0PgogIDx0ZXh0IH
g9IjcwMCIgeT0iMTMwIiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0
iMTAiIGZpbGw9IiM2NjYiIHRleHQtYW5jaG9yPSJtaWRkbGUiPlByb2Nlc3MgKHNhZmUpPC90ZXh0
PgogIDx0ZXh0IHg9IjcwMCIgeT0iMTQ1IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiI
GZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM2NjYiIHRleHQtYW5jaG9yPSJtaWRkbGUiPkV4dHJhY3QgKH
NhZmUpPC90ZXh0PgogIDx0ZXh0IHg9IjcwMCIgeT0iMTYwIiBmb250LWZhbWlseT0iQXJpYWwsIHN
hbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM2NjYiIHRleHQtYW5jaG9yPSJtaWRkbGUi
PlN0b3JlIChzYWZlKTwvdGV4dD4KCiAgPCEtLSBDb21wbGhbmNlIFNlY3Rpb24gLS0+CiAgPHJlY
3QgeD0iMzAiIHk9IjIwMCIgd2lkdGg9Ijc0MCIgaGVpZ2h0PSIxMDUiIHJ4PSI4IiBmaWxsPSIjZm
ZmIiBzdHJva2U9IiNlMmU4ZjAiIHN0cm9rZS13aWR0aD0iMiIvPgogIDx0ZXh0IHg9IjQwMCIgeT0
iMjI1IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTIiIGZvbnQt
d2VpZ2h0PSJib2xkIiBmaWxsPSIjMzMzIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5Db21wbGhbmNlI
EJlbmVmaXRzPC90ZXh0PgoKICA8cmVjdCB4PSI1MCIgeT0iMjQwIiB3aWR0aD0iMTYwIiBoZWlnaH
Q9IjUwIiByeD0iNiIgZmlsbD0iI2ZlY2FjYSIvPgogIDx0ZXh0IHg9IjEzMCIgeT0iMjYwIiBmb25
0LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZvbnQtd2VpZ2h0PSJi
b2xkIiBmaWxsPSIjZGMyNjI2IiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5QQ0ktRFNTPC90ZXh0PgogI
Dx0ZXh0IHg9IjEzMCIgeT0iMjc4IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbn
Qtc2l6ZT0iMTAiIGZpbGw9IiM3ZjFkMWQiIHRleHQtYW5jaG9yPSJtaWRkbGUiPkNhcmQgbnVtYmV
ycyBuZXZlciBzdG9yZWQ8L3RleHQ+CgogIDxyZWN0IHg9IjIzMCIgeT0iMjQwIiB3aWR0aD0iMTYw
IiBoZWlnaHQ9IjUwIiByeD0iNiIgZmlsbD0iI2ZlZjNjNyIvPgogIDx0ZXh0IHg9IjMxMCIgeT0iM
jYwIiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZvbnQtd2
VpZ2h0PSJib2xkIiBmaWxsPSIjOTI0MDBlIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5HRFBSPC90ZXh
0PgogIDx0ZXh0IHg9IjMxMCIgeT0iMjc4IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYi
IGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiM3ODM1MGYiIHRleHQtYW5jaG9yPSJtaWRkbGUiPkVtYWlsL
0lQIGFkZHJlc3NlcyBtYXNrZWQ8L3RleHQ+CgogIDxyZWN0IHg9IjQxMCIgeT0iMjQwIiB3aWR0aD
0iMTYwIiBoZWlnaHQ9IjUwIiByeD0iNiIgZmlsbD0iI2RiZWFmZSIvPgogIDx0ZXh0IHg9IjQ5MCI
geT0iMjYwIiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2VyaWYiIGZvbnQtc2l6ZT0iMTAiIGZv
bnQtd2VpZ2h0PSJib2xkIiBmaWxsPSIjMWU0MGFmIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5ISVBBQ
TwvdGV4dD4KICA8dGV4dCB4PSI0OTAiIHk9IjI3OCIgZm9udC1mYW1pbHk9IkFyaWFsLCBzYW5zLX

```
NlcmlmIiBmb250LXNpemU9IjEwIiBmaWxsPSIjMWUzYTVmIiB0ZXh0LWFuY2hvcj0ibWlkZGxlIj5
QSEkgbmV2ZXIgZXhwb3NlZDwvdGV4dD4KCiAgPHJlY3QgeD0iNTkwIiB5PSIyNDAiIHdpZHRoPSIx
NjAiIGhlaWdodD0iNTAiIHJ4PSI2IiBmaWxsPSIjZDFmYWU1Ii8+CiAgPHRleHQgeD0iNjcwIiB5P
SIyNjAiIGZvbnQtZmFtaWx5PSJBcmlhbCwgc2Fucy1zZXJpZiIgZm9udC1zaXplPSIxMCIgZm9udC
13ZWlnaHQ9ImJvbGQiIGZpbGw9IiMwNDc4NTciIHRleHQtYW5jaG9yPSJtaWRkbGUiPlNPQyAyPC9
0ZXh0PgogIDx0ZXh0IHg9IjY3MCIgeT0iMjc4IiBmb250LWZhbWlseT0iQXJpYWwsIHNhbnMtc2Vy
aWYiIGZvbnQtc2l6ZT0iMTAiIGZpbGw9IiMwNjRlM2IiHRleHQtYW5jaG9yPSJtaWRkbGUiPlNlY
3JldHMgcHJvdGVjdGVkPC90ZXh0Pgo8L3N2Zz4K)



### Why Masking First?

| Order | Reason |
|-------|--------|
| **Before parsing** | Sensitive data never extracted to fields |
| **Before routing** | Can't route based on unmasked PII |
| **Before storage** | Compliance-safe from the start |
| **Before extraction** | Metrics/events don't contain PII |

---

## Understanding Masking

### What Masking Does

Masking replaces sensitive patterns with redacted values:

**Before:**
```
User email: john.doe@example.com, card: 4111-1111-1111-1111
```

**After:**
```
User email: [EMAIL_REDACTED], card: [CC_REDACTED]
```

### Masking Processor Configuration

| Field | Description |
|-------|-------------|
| **Name** | Descriptive name for the mask |
| **Matching Condition** | When to apply masking |
| **Fields** | Which fields to mask (or all) |
| **Pattern** | DPL pattern to match sensitive data |
| **Replacement** | Text to replace matches with |
```

````
### Masking vs. Dropping

| Action | Use When |
|--------|----------|
| **Mask** | Need the record, just hide sensitive parts |
| **Drop** | Entire record is not needed |

---

## Built-in Masking Patterns

OpenPipeline includes pre-built matchers for common sensitive data.

### Credit Card Numbers

**DPL Pattern:**
```
CREDITCARD
```

**Matches:**
- `4111111111111111`
- `4111-1111-1111-1111`
- `4111 1111 1111 1111`

**Masking Processor:**
```
Name: Mask Credit Cards
Pattern: CREDITCARD:cc
Replacement: [CC_REDACTED]
Fields: content, message
```

### Email Addresses

**DPL Pattern:**
```
EMAIL
```

**Matches:**
- `user@example.com`
- `first.last@company.org`

**Masking Processor:**
```
Name: Mask Emails
Pattern: EMAIL:email
````
````

````
Replacement: [EMAIL_REDACTED]
Fields: content, message
```

### IP Addresses

**DPL Pattern:**
```
IPADDR
IPV4ADDR
IPV6ADDR
```

**Masking Processor:**
```
Name: Mask IP Addresses
Pattern: IPADDR:ip
Replacement: [IP_REDACTED]
Fields: content, client_ip, remote_addr
```

### Phone Numbers (Custom Pattern)

**DPL Pattern:**
```
'(' INT{3} ')' SPACE? INT{3} '-' INT{4}
```

**Matches:**
- `(555) 123-4567`
- `(555)123-4567`

---

## Custom Masking with replacePattern

The `replacePattern` function enables custom masking using DPL patterns.

### Syntax

```dql
fieldsAdd content = replacePattern(content, "PATTERN", replacement:
"REPLACEMENT")
```

### Example: Mask SSN

**Pattern:** SSN format `123-45-6789`
````
````

```dql
fieldsAdd content = replacePattern(content, "INT{3} '-' INT{2} '-' INT{4}",
replacement: "[SSN_REDACTED]")
```

Example: Mask API Keys

Pattern: Keys starting with `key_` followed by alphanumeric

```dql
fieldsAdd content = replacePattern(content, "'key_' WORD", replacement: "
[API_KEY_REDACTED]")
```

Example: Mask Bearer Tokens

Pattern: Bearer tokens in Authorization headers

```dql
fieldsAdd content = replacePattern(content, "'Bearer ' NSPACE", replacement:
"Bearer [TOKEN_REDACTED]")
```

Example: Mask Passwords in URLs

Pattern: Password parameter in query strings

```dql
fieldsAdd content = replacePattern(content, "'password=' LD:pwd ('&'|EOL)",
replacement: "password=[REDACTED]&")
```

Chaining Multiple Masks

```dql
// Apply multiple masks in sequence
fieldsAdd content = replacePattern(content, "CREDITCARD", replacement: "
[CC_REDACTED]")
| fieldsAdd content = replacePattern(content, "EMAIL", replacement: "
[EMAIL_REDACTED]")
| fieldsAdd content = replacePattern(content, "IPADDR", replacement: "
[IP_REDACTED]")
```

Compliance Patterns

````
### PCI-DSS Compliance

**Requirements:**
- Mask all Primary Account Numbers (PANs)
- Mask CVV/CVC codes
- Mask cardholder names when combined with PANs

**Pipeline Configuration:**

```dql
// Mask credit card numbers
fieldsAdd content = replacePattern(content, "CREDITCARD", replacement: "
[PAN_REDACTED]")
// Mask CVV (3-4 digits after 'cvv=' or 'cvc=')
| fieldsAdd content = replacePattern(content, "('cvv='|'cvc='|'CVV='|'CVC=')
INT{3,4}", replacement: "cvv=[REDACTED]")
```

### GDPR Compliance

**Requirements:**
- Mask personal identifiers (emails, phone numbers)
- Mask IP addresses (considered PII in EU)
- Mask names when identifiable

**Pipeline Configuration:**

```dql
// Mask emails
fieldsAdd content = replacePattern(content, "EMAIL", replacement: "
[PII_REDACTED]")
// Mask IP addresses
| fieldsAdd content = replacePattern(content, "IPADDR", replacement: "
[IP_REDACTED]")
// Mask user IDs
| fieldsAdd content = replacePattern(content, "'userId=' LD:uid",
replacement: "userId=[USER_REDACTED]")
```

### HIPAA Compliance

**Requirements:**
- Mask Protected Health Information (PHI)
- Mask patient identifiers
- Mask medical record numbers

**Pipeline Configuration:**
````

```dql
// Mask patient IDs
fieldsAdd content = replacePattern(content, "'patientId=' LD:pid",
replacement: "patientId=[PHI_REDACTED]")
// Mask MRNs
| fieldsAdd content = replacePattern(content, "'mrn=' LD:mrn", replacement:
"mrn=[PHI_REDACTED]")
// Mask SSNs
| fieldsAdd content = replacePattern(content, "INT{3} '-' INT{2} '-' INT{4}",
replacement: "[SSN_REDACTED]")
```

SOC 2 Compliance

Requirements:
- Mask credentials and secrets
- Mask authentication tokens
- Audit access to sensitive data

Pipeline Configuration:

```dql
// Mask passwords
fieldsAdd content = replacePattern(content, "('password='|'pwd='|'passwd=')
LD:pwd", replacement: "password=[REDACTED]")
// Mask API keys
| fieldsAdd content = replacePattern(content, "
('api_key='|'apiKey='|'API_KEY=') LD:key", replacement: "api_key=[REDACTED]")
// Mask tokens
| fieldsAdd content = replacePattern(content, "'Bearer ' NSPACE",
replacement: "Bearer [REDACTED]")
```

Field Removal for Security

Sometimes it's better to remove entire fields rather than mask values.

Using fieldsRemove

```dql
// Remove sensitive fields entirely
fieldsRemove password, secret, api_key, token, authorization
```

When to Remove vs. Mask

````
| Scenario | Action |
|----------|--------|
| Field always contains sensitive data | Remove |
| Only some values are sensitive | Mask |
| Need to know field existed | Mask with placeholder |
| Compliance requires no trace | Remove |

### Conditional Field Removal

Apply removal only when field contains sensitive patterns:

```dql
// Remove field only if it contains a pattern
fieldsAdd authorization = if(contains(authorization, "Bearer"), null, else:
authorization)
```

---

## Validating Masking

After configuring masking, verify it's working correctly.

> ⚠️ **Important:** Test with sample data before deploying to production.

```python
// Check for any remaining credit card patterns in stored logs
// If masking works, this should return 0 results
fetch logs, from: now() - 24h
| filter matchesPhrase(content, "4111")
   OR matchesPhrase(content, "5500")
   OR matchesPhrase(content, "3782")
| limit 10
```

```python
// Check for any remaining email patterns
// Look for @ symbol with surrounding text
fetch logs, from: now() - 24h
| filter matchesPhrase(content, "@gmail.com")
   OR matchesPhrase(content, "@yahoo.com")
   OR matchesPhrase(content, "@example.com")
| limit 10
```

```python
// Verify redaction placeholders are present
````
````

```
// This confirms masking is actively working
fetch logs, from: now() - 24h
| filter contains(content, "[REDACTED]")
   OR contains(content, "[CC_REDACTED]")
   OR contains(content, "[EMAIL_REDACTED]")
   OR contains(content, "[PII_REDACTED]")
| summarize {masked_count = count()}, by: {dt.openpipeline.pipelines}
| sort masked_count desc
```

```python
// Sample masked logs to verify format
fetch logs, from: now() - 24h
| filter contains(content, "REDACTED")
| fields timestamp, content
| limit 20
```

```python
// Audit: Count masked records by pipeline and type
fetch logs, from: now() - 24h
| fieldsAdd has_cc_mask = contains(content, "[CC_REDACTED]"),
           has_email_mask = contains(content, "[EMAIL_REDACTED]"),
           has_ip_mask = contains(content, "[IP_REDACTED]"),
           has_pii_mask = contains(content, "[PII_REDACTED]")
| summarize {
    total = count(),
    cc_masked = countIf(has_cc_mask),
    email_masked = countIf(has_email_mask),
    ip_masked = countIf(has_ip_mask),
    pii_masked = countIf(has_pii_mask)
  }, by: {dt.openpipeline.pipelines}
```

```python
// Check for potential SSN patterns that might be missed
// Pattern: ###-##-#### where # is a digit
fetch logs, from: now() - 24h
| filter matchesPhrase(content, "-")
| filter NOT contains(content, "[SSN_REDACTED]")
| fields content
| limit 50
```

Testing Masking in DPL Architect

```
Before deploying masking rules, test them:

1. Open **DPL Architect** in Dynatrace
2. Paste sample log with sensitive data
3. Test your DPL pattern matches correctly
4. Verify replacement produces expected output

### Test Cases to Include

| Test Case | Input | Expected Output |
|-----------|-------|-----------------|
| Credit card | `4111-1111-1111-1111` | `[CC_REDACTED]` |
| Credit card no dashes | `4111111111111111` | `[CC_REDACTED]` |
| Email | `user@example.com` | `[EMAIL_REDACTED]` |
| IPv4 | `192.168.1.1` | `[IP_REDACTED]` |
| IPv6 | `2001:db8::1` | `[IP_REDACTED]` |
| Mixed content | Multiple patterns | All redacted |

---

## Best Practices

### Masking Configuration

| Practice | Reason |
|----------|--------|
| Mask before routing | Routing rules shouldn't see PII |
| Use descriptive placeholders | Aids troubleshooting |
| Test patterns thoroughly | Avoid over/under masking |
| Document masking rules | Compliance audits |

### Pattern Design

| Practice | Reason |
|----------|--------|
| Be specific | Avoid false positives |
| Handle variations | Same data, different formats |
| Use built-in matchers | Pre-tested, reliable |
| Chain patterns | Cover all sensitive data |

### Compliance

| Practice | Reason |
|----------|--------|
| Document all masking | Audit trail |
| Regular pattern review | New data sources |
| Test with real samples | Ensure effectiveness |
| Monitor for leaks | Continuous validation |
```

````
### Performance

| Practice | Reason |
|----------|--------|
| Apply to specific fields | Faster than all fields |
| Use matching conditions | Skip unneeded records |
| Order patterns by frequency | Most common first |

---

## Complete Security Pipeline Example

### Pipeline: `payment-logs-secure`

**Masking Processor 1: Credit Cards**
```
Name: Mask Credit Cards
Pattern: CREDITCARD
Replacement: [CC_REDACTED]
Fields: content, card_number
Matching: (all records)
```

**Masking Processor 2: CVV Codes**
```dql
fieldsAdd content = replacePattern(content, "('cvv='|'cvc=') INT{3,4}",
replacement: "cvv=[REDACTED]")
```

**Masking Processor 3: Emails**
```
Name: Mask Emails
Pattern: EMAIL
Replacement: [EMAIL_REDACTED]
Fields: content, customer_email
Matching: (all records)
```

**Masking Processor 4: IP Addresses**
```
Name: Mask IPs
Pattern: IPADDR
Replacement: [IP_REDACTED]
Fields: content, client_ip, remote_addr
Matching: (all records)
```
````

Masking Processor 5: Auth Tokens
```dql
fieldsAdd content = replacePattern(content, "'Bearer ' NSPACE", replacement:
"Bearer [TOKEN_REDACTED]")
| fieldsRemove authorization, auth_token
```

Pipeline Verification Query

```python
// Verify complete masking for payment-logs pipeline
fetch logs, from: now() - 1h
| filter dt.openpipeline.pipelines == "payment-logs-secure"
| summarize {
    total = count(),
    with_cc = countIf(contains(content, "[CC_REDACTED]")),
    with_email = countIf(contains(content, "[EMAIL_REDACTED]")),
    with_ip = countIf(contains(content, "[IP_REDACTED]")),
    with_token = countIf(contains(content, "[TOKEN_REDACTED]"))
  }
```

Next Steps

Now that security is configured, complete your migration:

Notebook	Focus Area
OPMIG-09	Troubleshooting & Validation

References

- [OpenPipeline Masking](https://docs.dynatrace.com/docs/discover-dynatrace/platform/openpipeline/use-cases/mask-sensitive-data)
- [DPL replacePattern](https://docs.dynatrace.com/docs/discover-dynatrace/platform/grail/dynatrace-pattern-language/dpl-architecture)
- [Data Privacy in Dynatrace](https://docs.dynatrace.com/docs/manage/data-privacy-and-security)
- [Compliance Best Practices](https://docs.dynatrace.com/docs/manage/data-privacy-and-security/data-privacy)

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