**Step 2: Databricks workspace + storage layout (secure access to ADLS)**

Below is a focused plan + Terraform you can apply right away. Outcome: a Databricks workspace (VNet-injected, no public IPs), NAT egress, ADLS **raw/silver/gold** containers, and RBAC so clusters can read/write via private endpoints you already created.

**What we’ll create now**

1. **Network for Databricks**

* Two subnets in your hub VNet: snet-dbx-private, snet-dbx-public
* **NAT Gateway** for egress (clusters have **no public IP**)

1. **Databricks workspace**

* VNet injection into the above subnets
* **No Public IP** (secure cluster connectivity)
* System-assigned managed identity

1. **ADLS containers** (HNS): raw, silver, gold, quarantine, logs
2. **RBAC**: grant the Databricks workspace identity **Storage Blob Data Contributor** on the storage account

**Terraform (append to your /infra)**

Assumes you already applied Step 1 (RGs, VNet, PE subnet, Storage, Key Vault). Adjust names if you changed them earlier.

**databricks.tf**

# -------- Subnets for Databricks (in existing VNet) --------

resource "azurerm\_subnet" "dbx\_public" {

name = "snet-dbx-public"

resource\_group\_name = azurerm\_resource\_group.net.name

virtual\_network\_name = azurerm\_virtual\_network.hub.name

address\_prefixes = ["10.10.2.0/24"]

}

resource "azurerm\_subnet" "dbx\_private" {

name = "snet-dbx-private"

resource\_group\_name = azurerm\_resource\_group.net.name

virtual\_network\_name = azurerm\_virtual\_network.hub.name

address\_prefixes = ["10.10.3.0/24"]

}

# -------- NAT for egress (no public IPs on clusters) --------

resource "azurerm\_public\_ip" "nat" {

name = "pip-aml-nat"

location = azurerm\_resource\_group.net.location

resource\_group\_name = azurerm\_resource\_group.net.name

allocation\_method = "Static"

sku = "Standard"

tags = local.tags

}

resource "azurerm\_nat\_gateway" "nat" {

name = "ngw-aml"

location = azurerm\_resource\_group.net.location

resource\_group\_name = azurerm\_resource\_group.net.name

sku\_name = "Standard"

tags = local.tags

}

resource "azurerm\_nat\_gateway\_public\_ip\_association" "nat\_assoc" {

nat\_gateway\_id = azurerm\_nat\_gateway.nat.id

public\_ip\_address\_id = azurerm\_public\_ip.nat.id

}

resource "azurerm\_subnet\_nat\_gateway\_association" "dbx\_private\_nat" {

subnet\_id = azurerm\_subnet.dbx\_private.id

nat\_gateway\_id = azurerm\_nat\_gateway.nat.id

}

resource "azurerm\_subnet\_nat\_gateway\_association" "dbx\_public\_nat" {

subnet\_id = azurerm\_subnet.dbx\_public.id

nat\_gateway\_id = azurerm\_nat\_gateway.nat.id

}

# -------- Databricks Workspace (VNet-injected, no public IP) --------

resource "azurerm\_resource\_group" "dbx" {

name = "rg-aml-dbx"

location = local.location

tags = local.tags

}

resource "azurerm\_databricks\_workspace" "ws" {

name = "dbw-aml-${var.env}"

resource\_group\_name = azurerm\_resource\_group.dbx.name

location = local.location

sku = "premium"

managed\_resource\_group\_id = null

custom\_parameters {

no\_public\_ip = true

public\_subnet\_name = azurerm\_subnet.dbx\_public.name

private\_subnet\_name = azurerm\_subnet.dbx\_private.name

virtual\_network\_id = azurerm\_virtual\_network.hub.id

}

public\_network\_access\_enabled = false

identity {

type = "SystemAssigned"

}

tags = local.tags

}

# -------- ADLS containers for medallion --------

resource "azurerm\_storage\_container" "raw" {

name = "raw"

storage\_account\_name = azurerm\_storage\_account.adls.name

container\_access\_type = "private"

}

resource "azurerm\_storage\_container" "silver" {

name = "silver"

storage\_account\_name = azurerm\_storage\_account.adls.name

container\_access\_type = "private"

}

resource "azurerm\_storage\_container" "gold" {

name = "gold"

storage\_account\_name = azurerm\_storage\_account.adls.name

container\_access\_type = "private"

}

resource "azurerm\_storage\_container" "quarantine" {

name = "quarantine"

storage\_account\_name = azurerm\_storage\_account.adls.name

container\_access\_type = "private"

}

resource "azurerm\_storage\_container" "logs" {

name = "logs"

storage\_account\_name = azurerm\_storage\_account.adls.name

container\_access\_type = "private"

}

# -------- RBAC: let Databricks workspace MI access ADLS --------

data "azurerm\_client\_config" "current" {}

resource "azurerm\_role\_assignment" "dbx\_sa\_blob\_contrib" {

scope = azurerm\_storage\_account.adls.id

role\_definition\_name = "Storage Blob Data Contributor"

principal\_id = azurerm\_databricks\_workspace.ws.identity[0].principal\_id

}

**Apply**

terraform init

terraform apply -auto-approve \

-var="tenant\_id=$(az account show --query tenantId -o tsv)" \

-var="sa\_name=<your-same-storage-from-step1>"

**Post-provision: quick checks**

1. **Private resolution**  
   From a VM in the VNet, nslookup <sa>.blob.core.windows.net → should resolve to privatelink.blob… private IP.
2. **Databricks workspace**

* Open the workspace URL (from Azure portal).
* Create a **cluster**:
  + Policy: default is fine.
  + Enable “No public IP” (should be enforced).
  + Attach to workspace.

1. **ADLS access from a Notebook (ABFS direct)**  
   Use OAuth passthrough with the workspace **Managed Identity** (recommended) or set Spark configs at cluster level if using a Service Principal. For MI-based access on Unity Catalog you’ll typically define a **storage credential** + **external location** (best practice). If you’re not on UC yet, a quick smoke test:

# In a Databricks notebook (Python), just try to list the raw container:

raw\_path = "abfss://raw@<yourstorage>.dfs.core.windows.net/"

display(dbutils.fs.ls(raw\_path))

(If permission denied, confirm the RBAC role assignment above has propagated—can take a couple of minutes.)

**What’s next (Step 3 preview)**

* **Unity Catalog** setup: metastore, storage credential (Key Vault-backed), external locations pointing to raw/silver/gold.
* **Ingestion bootstrap**: ADF → Databricks Jobs for first sources (transactions, parties, sanctions).
* **Folder standards** under each container, e.g.:
* raw/source=<system>/ingestion\_date=YYYY-MM-DD/
* silver/domain=<entity>/yyyy=MM=dd/

gold/mart=<subject>/