Multiple Storage Accounts

TLDR;

Databricks recommends that Storage accounts are segmented into sensitive and non-sensistive accounts. This is to segregate the sensitive data from all the other accounts and provide a clear boundary between those storage locations. Additionally Azure Infrastructure Encryption should be enabled on the sensitive accounts in order to provide a 2nd layer of protection in case of a breach.

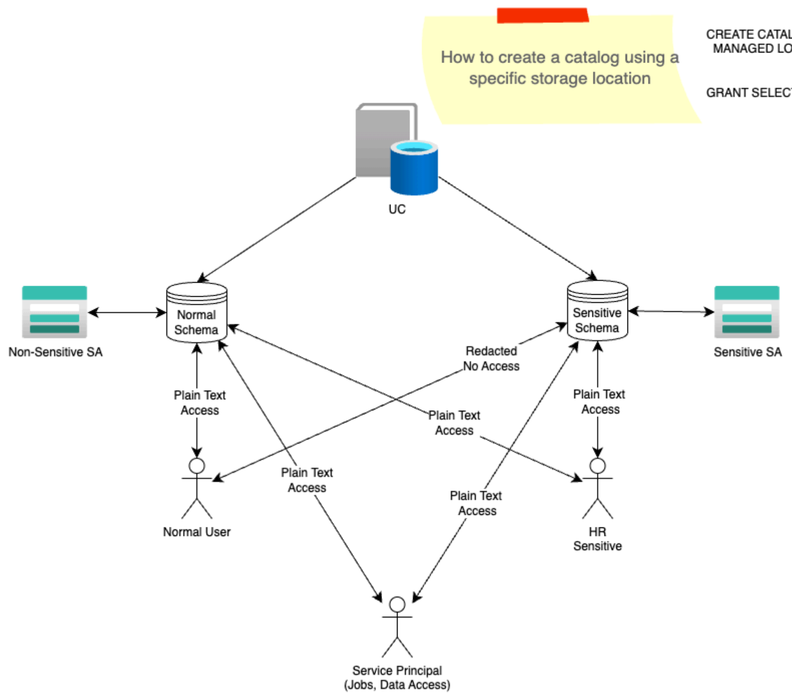
1. Create a new Storage Account(Infrastructure Encryption can not be enabled on existing accounts)
2. Enable Encryption on the new SA
3. Map the storage account to the Metastore
4. Implement Column Masking to redact sensitive information

Reasoning

Physical Separation of Sensitive Data

The first and foremost reason for this is to have a completely different security model around sensitive data if needed. The 2nd reason is because you may not want to encrypt everything in the storage account. Using a separate SA for this purpose we are able to just encrypt the data that we want to encrypt, not everything. There is a performance hit with using infrastructure encryption, but it is small due to MS Azure using Hardware Acceleration to perform the AES-256 encryption calculations.

1. Separate Accounts
    a. Physically separate the data
2. Establish Security Protocol
    a. Assign rights and privileges to service accounts that need to access this data
    b. Assign rights and privileges to user groups(HR, IT, etc) as needed for data access
3. Encryption at rest is only on the secure SA.



```
CREATE CATALOG IF NOT EXISTS mycatalog
  MANAGED LOCATION 'abfss://mycontainer@<myaccount.dfs.core.windows.net//depts/finance';


GRANT SELECT ON mycatalog TO `finance-team`;
```

The above diagram illustrates a simple set up illustrating the needs of Three different Actors.

1. Normal User
   a. Needs to be able to view non-sensitive information
   b. User does not have access to the sensitive information
      i. To redact and allow the user to view some of the information use a Masking policy with a group.
      ii. Redact information based on the group membership.
2. HR Sensitive
   a. This user is able to view sensitive information that is secured by the HR group privilege
   b. This user may not have access to everything in the table
      i. Tables can be redacted by applying the appropriate masking policy
   c. This user is able to access non-sensitive data as well.
3. Service Principals(The accounts that run the jobs)
   a. Service Principals are the 'User' that a job will run as.
      i. These accounts should not be associated with a real user
   b. The Service Principal must be able to read and write to the protected SA as part of its job execution.
      i. Failure to do so will result in the job processing the redacted information and not the real information.

How to configure

This list is assuming a fresh install. I like to give that so that the full process can be seen. The relevant sections are item 4 and higher.

To create a catalog with a specific storage account for Azure in Databricks, follow these steps:

1. **Create an Access Connector for Azure Databricks**:

    - Log in to the Azure Portal as a Contributor or Owner of a resource group.
    - Click on "+ Create" or "Create a new resource".
    - Search for "Access Connector for Azure Databricks" and select it.
    - Click "Create".
    - On the Basics tab, fill in the required fields:
        - **Subscription**: Select the Azure subscription.
        - **Resource group**: Select the resource group.
        - **Name**: Enter a name for the connector.
        - **Region**: Select the same region as the storage account.
    - Click "Review + create" and then "Create".
    - Once the deployment is complete, note the Resource ID of the access connector.

2. **Grant the Managed Identity Access to the Storage Account**:

    - Go to your Azure Data Lake Storage Gen2 account in the Azure Portal.
    - Navigate to "Access Control (IAM)" and click "+ Add" to select "Add role assignment".
    - Assign the "Storage Blob Data Contributor" role to the managed identity associated with the access connector.

3. **Create a Unity Catalog Metastore**:

    - Log in to the Azure Databricks account console as an account admin.
    - Navigate to "Catalog" and click "Create Metastore".
    - Fill in the required fields:
        - **Name**: Enter a name for the metastore.
        - **Region**: Select the region where the metastore will be deployed.
        - **ADLS Gen 2 path**: Enter the path to the storage container (e.g., `abfss://<container>@<storage-account>.dfs.core.windows.net`).
        - **Access Connector ID**: Enter the Resource ID of the access connector.
        - **Managed Identity ID** (optional): If using a user-assigned managed identity, enter its Resource ID.
    - Select the workspaces to link to the metastore.

4. **Create a Storage Credential**:

- Use the Databricks CLI to create a storage credential:

```
Unset
databricks storage-credentials create --json '{

  "name": "<credential-name>",

  "azure_managed_identity": {

    "access_connector_id": "<access-connector-id>",

    "managed_identity_id": "<managed-identity-id>"

  }

}' --profile <profile-name>
```

- Note the storage credential ID from the response.

5. **Create an External Location**:

- Use the Databricks CLI to create an external location:

```
Unset
databricks external-locations create --json '{

  "name": "<external-location-name>",

  "url": "abfss://<container>@<storage-account>.dfs.core.windows.net",

  "credential_name": "<credential-name>"

}' --profile <profile-name>
```

6. **Create the Catalog**:

- Use the Databricks CLI to create the catalog:

```
Unset
databricks catalogs create --json '{

  "name": "<catalog-name>",

  "storage_root": "abfss://<container>@<storage-account>.dfs.core.windows.net",

  "comment": "Catalog managed by Databricks"
}' --profile <profile-name>
```

By following these steps, you will have successfully created a catalog with a specific storage account for Azure in Databricks.

Databricks support can assist you with this setup if you wish. Reach out to the account team for more information.