

# Lab 14: Accessing ADLS Using Managed Identity (Most Secure - No Secrets Needed)

Author: Dr. Sandeep Kumar Sharma

---



## Why This Lab Matters

This is the **most secure and recommended** method for production environments using Azure Databricks because: - **No secrets** (no SAS tokens, no access keys, no client secrets) - Auth is handled by **Azure AD automatically** - Full support for enterprise security (RBAC, Conditional Access, auditing) - No risk of key leakage in notebooks or logs

This lab explains everything **step-by-step**, assuming your participants are beginners.

---

## Learning Objective

By the end of this lab, you will be able to: - Understand what a **Managed Identity** is - Enable Managed Identity for an Azure Databricks workspace - Assign correct RBAC roles to allow Databricks to access ADLS - Read files from ADLS using `abfss://` with **zero credentials** - Troubleshoot RBAC & identity access issues

---

## Learning Outcome

You will learn how Databricks automatically authenticates to ADLS using: - The **Databricks Managed Identity** (at workspace level) - Azure Active Directory token flow - Spark config without needing any keys

This becomes your most clean, secure, scalable method.

---

---

## Part A — Concept: What Is a Managed Identity? (Beginner Explanation)

---

A **Managed Identity** is an identity (just like a service principal) created by Azure **automatically** for a resource.

In simple terms:

Azure gives your Databricks workspace a special ID so it can log in to other Azure services **without storing passwords or secrets**.

There are two types: - **System-assigned** (recommended): created automatically for the Databricks workspace - **User-assigned** (advanced): reusable across resources

In this lab we will use **system-assigned managed identity**.

---

---

## Part B — Enable Managed Identity on Databricks (Azure Portal)

---

Follow these steps carefully:

### Step B1 — Open Databricks in Azure Portal

1. Go to [Azure Portal](#)
  2. Search **Azure Databricks**
  3. Open your Databricks workspace
-

## **Step B2 — Go to Managed Identity Section**

On the left blade: - Go to **Identity** → **System Assigned** - Turn **Status = ON** - Click **Save**

Azure automatically creates a managed identity with a unique Object ID.

---

## **Step B3 — Copy the Managed Identity Object ID**

You will need it for RBAC assignment.

Example:

Managed Identity Object ID: 7c1b9d2a-xxxx-xxxx-xxxx-xxxxxxxxxxxx

---

# **Part C — Grant Storage Permissions (RBAC)**

---

Your Databricks workspace now has an identity — but it still needs **permissions** to access ADLS.

## **Step C1 — Open your Storage Account**

Go to **Azure Portal** → **Storage Accounts** → <your-storage-account>.

## **Step C2 — Assign RBAC Role**

1. Go to **Access Control (IAM)**
2. Click **Add** → **Add role assignment**
3. Choose role:
4. **Storage Blob Data Contributor** (read/write)
5. OR **Storage Blob Data Reader** (read-only)
6. Click **Next**
7. Select **Managed Identity**
8. Choose your Databricks workspace
9. Assign role

Now Databricks can access ADLS **without secrets**.

---

---

## Part D — Read ADLS Using Managed Identity (Notebook Code)

---

For **Managed Identity**, Databricks uses `DefaultAzureCredential` (built into compute cluster).

### **IMPORTANT**

**NO spark.conf secrets are required.**

**NO client IDs needed.**

**NO access keys.**

You just provide the path, and Databricks handles identity.

---

### **Step D1 — Define Storage Path**

```
storage_account_name = "yourstorageaccount"
container_name = "input"

full_adls_path = f"abfss://{{container_name}}
@{storage_account_name}.dfs.core.windows.net/"
```

---

### **Step D2 — Read the CSV File**

```
file_path = full_adls_path + "employee.csv"

df = spark.read.csv(file_path, header=True, inferSchema=True)
display(df)
```

If permissions are correct — this will **immediately work**.

---

## Step D3 — List Files

```
display(dbutils.fs.ls(full_adls_path))
```

---

---

---

---

---

## Part E — Write Back to ADLS (No Secrets Needed)

```
output_path = full_adls_path + "processed/employee_output"  
df.write.mode("overwrite").parquet(output_path)
```

---

---

---

---

---

## Part F — How Does Authentication Work Internally? (Simple Explanation)

Databricks cluster internally calls Azure AD using:

```
Managed Identity → Azure AD Token → ADLS
```

---

No passwords.  
No secrets.  
No certificates.  
No SAS tokens.

Azure manages everything automatically.

---

---

## Part G — Troubleshooting Section

---

### Error: AuthorizationPermissionMismatch

- Solution: Assign **Storage Blob Data Contributor** at
  - Storage account level (recommended), or
  - Container level (minimum requirement)
- 

### Error: ResourceNotFound

- Check file path
  - Confirm container exists
  - Confirm file exists using Azure Storage Explorer
- 

### Error: AuthenticationFailed

- Ensure Managed Identity is **enabled**
  - Ensure cluster is **restarted** after enabling identity
- 

---

## Part H — When Should You Use Managed Identity?

---

Use Managed Identity when:

- You want **zero secrets** (best practice)
- You need **fully automated authentication**
- You work in enterprise-grade production
- You want **RBAC-based access** rather than key-based access

This is considered the **#1 best practice** for Databricks + ADLS.

---

---

## End of Lab 14

---

You have successfully learned: - How to enable Managed Identity for Databricks - How to assign Storage RBAC roles - How to read/write ADLS using `abfss://` with NO secrets - Why this is the most secure method in production

If you want, next I can create:  **Lab 15 – Comparing All ADLS Authentication Methods (Key, SAS, SPN, MI)**

or

 **Lab 15 – Full ETL Pipeline Using ADLS + Databricks + Managed Identity**