**AVDs -> AWS S3**

**1: Real-Time Streaming (Azure Event Hub → AWS Kinesis Firehose → S3)**

**Process Overview:**

1. Azure Monitor collects logs from 700 AVDs.
2. Logs are streamed to Azure Event Hub.
3. AWS Lambda or Kafka Connect fetches logs from Azure Event Hub.
4. Kinesis Firehose ingests and processes logs.
5. Logs are stored in Amazon S3.

**Flowchart**

Azure Virtual Desktops🡪 Azure Monitor (Log Analytics)🡪Azure Event Hub (Log Stream)🡪AWS Lambda or MSK (Fetch logs)🡪Kinesis Firehose (Data Processing)🡪Amazon S3 (Log Storage)

**2: Batch Processing (Azure Blob → AWS DataSync → S3)**

**Process Overview:**

1. Azure Monitor collects logs from AVDs.
2. Logs are stored in Azure Blob Storage.
3. AWS DataSync periodically transfers logs to S3 (every 60 minutes).

**Flowchart**

Azure Virtual Desktops🡪 Azure Monitor (Log Analytics)🡪Azure Blob Storage (Log Storage)🡪AWS DataSync (Sync every 5 min)🡪Amazon S3 (Log Storage)

Great! Here’s how we can implement this **PoC on a single Azure Virtual Desktop (VDI)**.

**🛠️ Step-by-Step Implementation for PoC**

**📌 Step 1: Set Up an Azure Virtual Desktop (VDI)**

1. **Provision an AVD (if not already set up)**:
   * Go to **Azure Portal** → **Azure Virtual Desktop**.
   * Deploy a **Windows 10/11 VM** with RDP access.
   * Ensure the VM has a public IP or is accessible via an internal network.
2. **Enable Log Collection on AVD**:
   * Open **Event Viewer** (eventvwr.msc).
   * Navigate to **Windows Logs → Application/System**.
   * Export some logs manually for testing:
   * Get-WinEvent -LogName Application | Export-Csv C:\temp\test-log.csv
   * Ensure logs are being generated.

**📌 Step 2: Configure Azure Blob Storage**

1. **Create an Azure Storage Account**:
   * Go to **Azure Portal** → **Storage Accounts**.
   * Click **Create a new Storage Account**.
   * Choose **Standard (General-Purpose V2)**.
   * Enable **Blob Storage** and select the appropriate region.
2. **Create a Blob Container for Logs**:
   * Open **Storage Account** → **Containers**.
   * Click **+ New Container** → Name it **avd-logs-poc**.
   * Set **Public Access Level** to **Private**.
3. **Upload Test Log File to Blob Storage**:
   * Use Azure CLI or the portal to upload logs:
   * az storage blob upload --account-name <storage\_account> --container-name avd-logs-poc --name test-log.csv --file C:\temp\test-log.csv

**📌 Step 3: Set Up AWS DataSync**

1. **Launch AWS DataSync Agent**:
   * Go to **AWS DataSync Console**.
   * Click **Create Agent**.
   * Deploy the agent as an **EC2 instance** (t3.medium) in the **same region as S3**.
2. **Create an Azure Blob Storage Source in AWS DataSync**:
   * Click **Create Task**.
   * Choose **Azure Blob Storage** as the **Source**.
   * Enter **Storage Account Name** and **Access Key**.
   * Provide the Blob Storage URL:
   * https://<storage\_account>.blob.core.windows.net/avd-logs-poc
   * Click **Next**.
3. **Configure Amazon S3 as the Destination**:
   * Choose **Amazon S3** as the destination.
   * Select **Create New S3 Bucket** (or use an existing one).
   * Name it **avd-logs-poc-bucket**.
4. **Set Sync Frequency**:
   * Configure **Sync Interval** to **5 minutes**.
   * Set **Transfer Mode** to **Incremental**.
   * Click **Create Task** and **Start Sync**.

**📌 Step 4: Verify Logs in Amazon S3**

1. Wait for **5 minutes** and check if the **test-log.csv** appears in S3:
   * Go to **AWS S3 Console**.
   * Open the bucket **avd-logs-poc-bucket**.
   * Verify that the logs have been synced.

**Log Migration from Azure Blob Storage to AWS S3 Using AWS DataSync**

**1. Introduction**

This document outlines the steps taken to migrate logs from **Azure Blob Storage** to **AWS S3** using **AWS DataSync**. It details the architecture, setup, challenges encountered, and their resolutions.

**2. Project Overview**

* **Source**: Azure Blob Storage
* **Destination**: AWS S3
* **Migration Method**: AWS DataSync
* **Use Case**: Near real-time or scheduled batch transfer of logs

**3. Architecture Flow**

1. **Azure Blob Storage** stores logs generated from Azure Virtual Desktops (AVDs).
2. **AWS DataSync** reads logs from Azure Blob Storage.
3. **AWS S3 Bucket** serves as the final storage for logs in AWS.
4. **AWS CloudWatch & SNS** (Optional) monitor and send alerts for transfer failures.

**4. Steps for Implementation**

**Step 1: Setup Azure Blob Storage**

1. Create an **Azure Storage Account**.
2. Create a **Blob Container** inside the storage account.
3. Upload sample log files to the container.
4. Generate an **SAS Token** (Shared Access Signature) with **Read (r) and List (l) permissions**.
5. Copy the SAS Token for later use in AWS DataSync.

**Step 2: Configure AWS DataSync**

1. Open **AWS DataSync Console**.
2. Click **Create Task** → Select **Azure Blob Storage** as the source.
3. Enter **Azure Storage Account Name & SAS Token**.
4. Choose **AWS S3** as the destination.
5. Set up the **S3 bucket and prefix** for organized storage.
6. Choose **transfer settings**:
   * Frequency: Every 5 minutes (or batch processing as needed)
   * Overwrite behavior: Replace existing files
7. Create and **Start Task**.

**Step 3: Verify Data Transfer**

1. Check **AWS S3** to confirm files appear in the correct destination

**5. Challenges & Resolutions**

**Challenge 1: Insufficient System Memory Error**

* **Error Message**: *"Insufficient system memory. Minimum required is 4GiB; recommended is 16GiB"*
* **Resolution**:
  + Increased Azure VM size running DataSync to **Standard\_D4s\_v3**.
  + Reduced memory usage by stopping unnecessary background processes.

**Challenge 2: Unable to List Azure Blobs on the Volume Root**

* **Error Message**: *"Unable to list Azure Blobs on the volume root"*
* **Resolution**:
  + Assigned **"Storage Blob Data Reader" IAM role** to DataSync identity.
  + Verified **Azure Storage Account & Container Name**.
  + Regenerated a new **SAS Token** with rl permissions.
  + Checked **network connectivity** between AWS and Azure.

**Challenge 3: Quota Exceeded Error**

* **Error Message**: *"QuotaExceeded: Operation could not be completed as it results in exceeding approved standardDASv5Family Cores quota."*
* **Resolution**:
  + Increased Azure **VM quota** through the portal.
  + Selected an available **region** with enough capacity.
  + Used a **different VM type** with lower quota usage.

By implementing **AWS DataSync**, we successfully migrated logs from **Azure Blob Storage to AWS S3** with a scalable and automated approach. Challenges were resolved by optimizing memory, correcting permissions, and improving network connectivity.