Practical - 3: AWS Cross-Account Role Configuration

Objective: To enable an IAM User in **Account 1** to assume an IAM Role in **Account 2** to perform actions (like S3 Full Access) in **Account 2**, demonstrating cross-account access using AWS Security Token Service (STS).

Phase 1: Configure IAM User and Permissions in Account 1

Goal: Create an IAM User (`pranav`) that has an inline policy allowing it to assume a role in Account 2. The Account 1 ID is used as **692977928139**.

1. **Create IAM User:**
   - Navigate to the IAM console in **Account 1**.
   - Create a new user named **`pranav`**.
   - Select **"Provide user access to the AWS Management Console"**.
   - Choose a **Custom password** or an auto-generated one.
   - Click **Next** to proceed to permissions.
2. **Set Initial Permissions (None):**
   - On the **Set permissions** step, choose **"Attach policies directly"**.
   - Do **not** attach any managed policies or add the user to a group.
   - Review and **Create user**.
3. **Attach Inline Policy (STS:AssumeRole):**
   - Navigate to the details page for the newly created user `pranav`.
   - Add an **Inline Policy** to the user.
   - Use the **Visual editor** and configure the policy as follows:
     - **Service: STS** (AWS Security Token Service).
     - **Actions:** Select **All STS actions** or specifically **`sts:AssumeRole`**.
     - **Resources:** Specify the **ARN** of the role that will be created in Account 2 (this is configured later, but must be defined here). For now, you can select **All** resources.
     - *Note: While the screenshot shows **All** resources, in a real environment, this should be restricted to the specific Role ARN.*
   - Review and create the policy.

Phase 2: Configure IAM Role in Account 2

Goal: Create a Role (`S3FullAccessForPranav`) in **Account 2** that is trusted by Account 1 and has S3 Full Access permissions. The Account 2 ID is used as **772548858659**.

1. **Create IAM Role:**

    - Navigate to the IAM console in **Account 2**.
    - Click **Create role**.
2. **Select Trusted Entity:**
    - For **Trusted entity type**, choose **AWS account**.
    - Select **Another AWS account**.
    - Enter the **Account ID of Account 1**: **`692977928139`**.
    - *Leave "Require MFA" and "Require external ID" unchecked for this simple practical.*
3. **Attach Permissions Policy:**
    - On the **Add permissions** page , search for and select the AWS managed policy **`AmazonS3FullAccess`**.
4. **Name, Review, and Create:**
    - Set the **Role name** to **`S3FullAccessForPranav`**.
    - Review the details, confirming the Trust Policy allows the `sts:AssumeRole` action for Account 1's ID.
    - Click **Create role**.
5. **Create S3 Bucket (for Validation):**
    - In **Account 2**, navigate to the S3 console.
    - Create a bucket, e.g., **`pranav-paralkar-aws-bucket`** , in a region like Europe (Stockholm) `eu-north-1`. This bucket will be used to test the access later.

Phase 3: Cross-Account Access Validation

Goal: Log in as the IAM User in Account 1 and assume the new role in Account 2.

1. **Log in as IAM User (Account 1):**
    - Use the console sign-in URL for Account 1.
    - Log in using the **Account ID** (`692977928139`), **IAM username** (`pranav`), and password. 2. **Switch Role (Assume Role):**
    - In the AWS Management Console (top-right corner), click on the username/role (which is currently `pranav@...`).
    - Click **Switch Role**.
    - Enter the following details for **Account 2**:
        - **Account ID:** `772548858659`
        - **Role:** `S3FullAccessForPranav`
        - **Display Name:** `S3FullAccessForPranav` (optional color selection, e.g., Yellow)
    - Click **Switch Role**.

2. **Validate S3 Access (Account 2):**
   - After switching, the console context changes to **Account 2** with the assumed role.
   - Navigate to the **S3** console.
   - You should be able to see and access the bucket `pranav-paralkar-aws-bucket` created in Phase 2. This proves the cross-account role assumption was successful and the permissions are working.