Technical Documentation
“Automated S3 Backup & Restore”

1. Project Overview

This document explains the design, implementation, and usage of an automated backup and restore system to Amazon S3 as per the CoreXtech project requirements. The system securely backs up a specified local directory to S3, supports automation, logging, incremental backups, and optional encryption.

2. Architecture and Approach

The solution is built using AWS CLI tools and Bash scripting. The primary components include:
- Source Directory: ~/data
- S3 Bucket: corextech-backup-[yourname]
- Backup Script: Uses aws s3 sync for efficient incremental backups.
- Restore Script: Retrieves the full backup from S3.
- Logging: Tracks each operation's outcome.
- Cron Job: Automates periodic execution.
- Optional Encryption: Server-side encryption using AES256.

3. Backup Flow

1. User triggers the backup script or it runs via cron.
2. Script checks for the source directory.
3. Syncs the content to the S3 bucket using incremental update.
4. Logs are recorded in a user-accessible location.

4. Restore Flow

1. User runs the restore script manually.
2. The script uses aws s3 sync to copy data from S3 to the local restore directory.
3. Log file captures the restoration process.

5. Challenges Faced

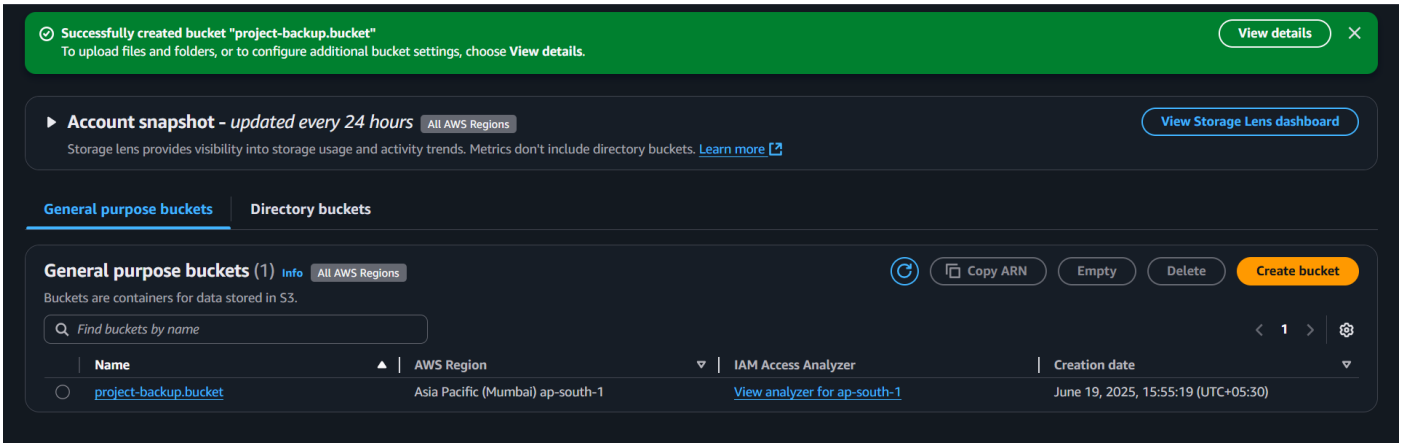
- Permission issues when writing to /var/log – resolved by logging to the user's home directory.
- 'No such file or directory' errors – resolved by creating missing directories.
- Access errors with S3 – resolved through correct configuration using AWS CLI.
- Script execution errors – resolved via permission and path corrections.

6. Assumptions and Trade-offs

- Assumes the AWS CLI is properly configured with access credentials.
- Assumes the backup frequency suits daily backups (cron timing adjustable).
- Encryption is optional and handled by enabling SSE (AES256).
- Backups are optimized for bandwidth using aws s3 sync.

A Basic Outline of Steps

Create S3 bucket in AWS through management console:



Install AWS CLI in local:

```
Ubuntu
anurag16@Anu-Anurag:~$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 63.1M  100 63.1M    0     0  7721k      0  0:00:08  0:00:08 --:--:-- 8110k
anurag16@Anu-Anurag:~$ unzip awscliv2.zip
```

```
anurag16@Anu-Anurag:~$ sudo ./aws/install
You can now run: /usr/local/bin/aws --version
anurag16@Anu-Anurag:~$ aws --version
aws-cli/2.27.39 Python/3.13.4 Linux/6.6.87.1-microsoft-standard-WSL2 exe/x86_64.ubuntu.24
anurag16@Anu-Anurag:~$
```

Test the connection:

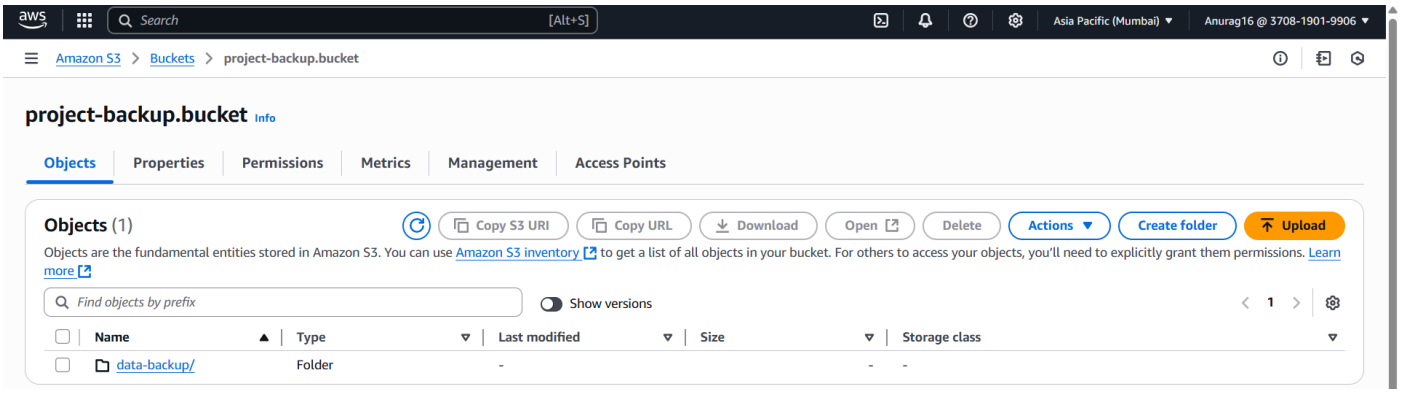
```
anurag16@Anu-Anurag:~$ aws s3 ls
2025-06-20 10:05:43 project-backup.bucket
anurag16@Anu-Anurag:~$
```

Create a backup script and give executable permission:

```
  GNU nano 7.2                                        /root/scripts/s3backup.sh
#!/bin/bash

SOURCE_DIR="/data"
BUCKET_NAME="corextech-backup-username"
S3_PATH="s3://$BUCKET_NAME/data-backup"
LOG_FILE="/var/log/s3_backup.log"

echo "[$(date)] Starting backup of $SOURCE_DIR to $S3_PATH" >> "$LOG_FILE"

aws s3 sync "$SOURCE_DIR" "$S3_PATH" --storage-class STANDARD_IA --sse AES256 >> "$LOG_FILE" 2>&1

if [ $? -eq 0 ]; then
    echo "[$(date)] Backup successful." >> "$LOG_FILE"
else
    echo "[$(date)] Backup FAILED!" >> "$LOG_FILE"
fi
```

```
anurag16@Anu-Anurag:~$ sudo chmod +x /root/scripts/s3backup.sh
```

Manually run the backup script to check if it's working fine:



Create a restore script for retrieving backup from S3 and give executable permission to it:

```
  GNU nano 7.2                                    /home/anurag16/s3restore.sh
#!/bin/bash

TARGET_DIR="$HOME/data"
BUCKET_NAME="project-backup.bucket"
S3_PATH="s3://$BUCKET_NAME/data-backup"
LOG_FILE="$HOME/s3_restore.log"

echo "[$(date)] Starting restore from $S3_PATH to $TARGET_DIR" >> "$LOG_FILE"

aws s3 sync "$S3_PATH" "$TARGET_DIR" >> "$LOG_FILE" 2>&1

if [ $? -eq 0 ]; then
    echo "[$(date)] Restore successful." >> "$LOG_FILE"
else
    echo "[$(date)] Restore FAILED!" >> "$LOG_FILE"
fi
```

```
anurag16@Anu-Anurag:~$ chmod +x ~/s3restore.sh
anurag16@Anu-Anurag:~$
```

Setup cron for backup to run every Monday and Thursday at 2 A.M.:

```
anurag16@Anu-Anurag:~$ crontab -e
no crontab for anurag16 - using an empty one

Select an editor.  To change later, run 'select-editor'.
  1. /bin/nano        <---- easiest
  2. /usr/bin/vim.basic
  3. /usr/bin/vim.tiny
  4. /bin/ed

Choose 1-4 [1]: 1
crontab: installing new crontab
anurag16@Anu-Anurag:~$
```

```
anurag16@Anu-Anurag:~$ crontab -l
0 2 * * 1,4 /home/anurag16/s3backup.sh >> /home/anurag16/html_s3_backup_cron.log 2>&1
```

And Done....