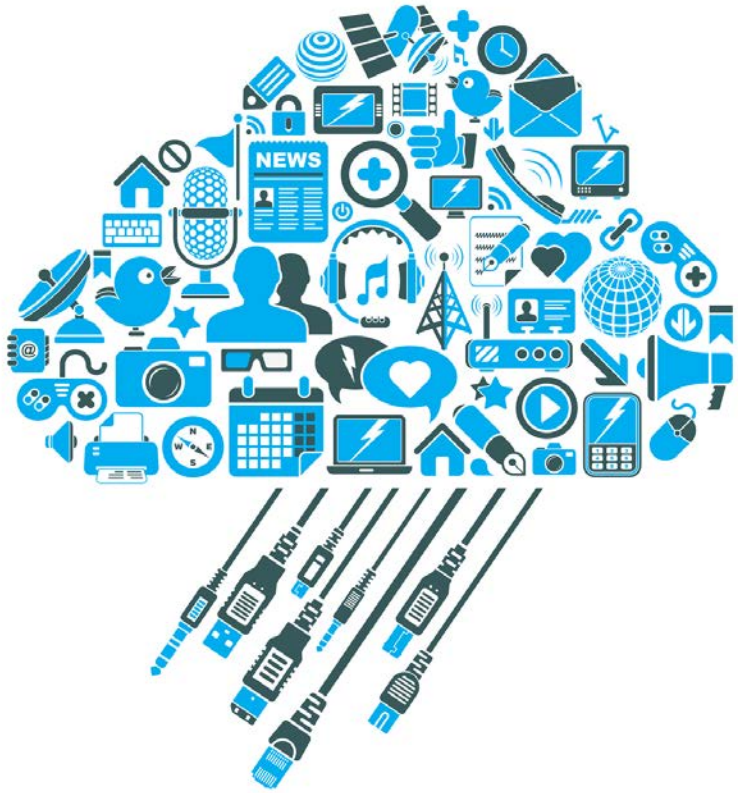
CRITFC AWS CLOUD STORAGE FRAMEWORK

Colleen Roe
April 17, 2019
ITMD Annual Workshop



Why Store Backups in AWS Cloud?

- Needed a secure off site storage location for weekly full backups
- Wanted to use this to take first step into the cloud
  - Economics of the cloud are very favorable (see next slide)
  - Wanted to lessen administrative burden at CRITFC
  - Wanted to gain experience using cloud so we can move entire ITMD “data center” there when equipment needs to be replaced

S3 and Glacier Costs

Example: cost of storing 1 TB of Glacier data for a year

1 TB * $0.004 per GB/month * 12 = $48

Example: cost of retrieving 50 GB of Glacier data (first 50 GB retrieval is free)

50 GB * $0.0 / GB = $0.00

Example: cost of storing 10 GB of S3 data

10 GB * $0.023 / GB * 12 = $27.60

SAN Costs

Example: cost of a new SAN (~5 year life)

~$15,000 -> ~$3000/year + indirect + maint + ….

Added benefit: less of ITMD staff time for admin tasks

S3 and Glacier Pricing

Storage pricing

Region: US West (Oregon)

- $0.004 per GB / Month

Retrieval pricing

Region: US West (Oregon)

Retrieval Time	Data Retrievals
Expedited	$0.03 per GB
Standard	$0.01 per GB
Bulk	$0.0025 per GB

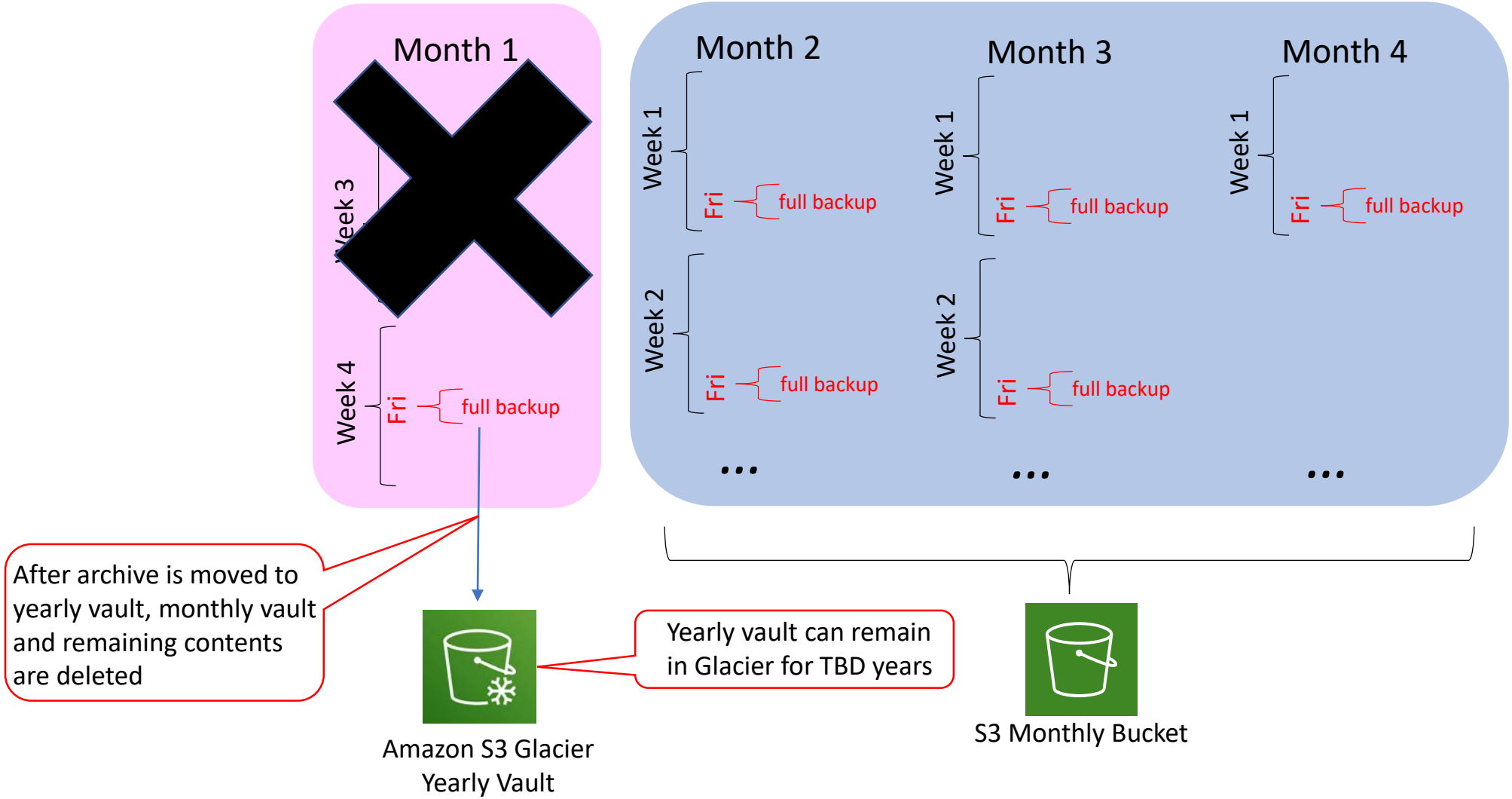
S3 Standard Storage

First 50 TB / Mont	$0.023 per GB

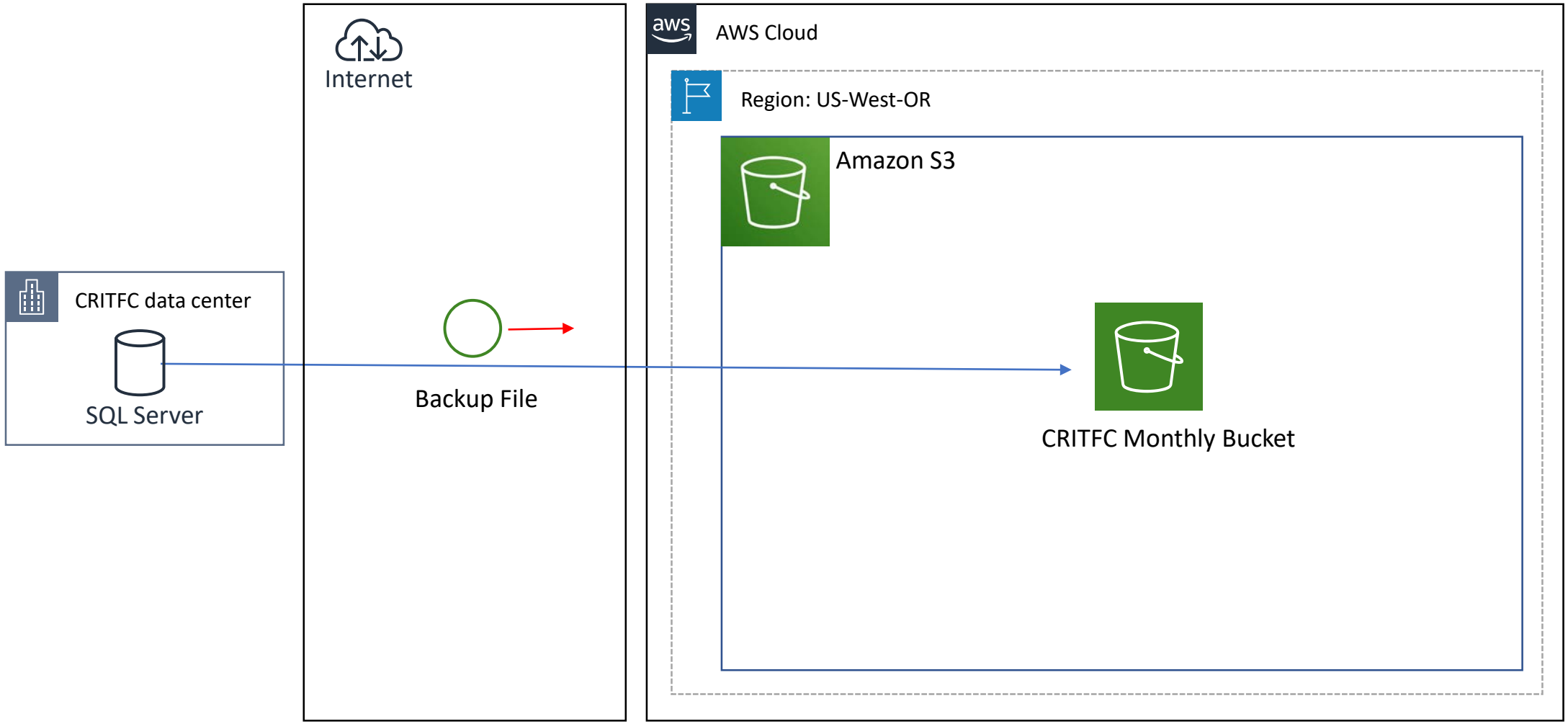
How CRITFC's AWS Backups Will Work

- Friday weekly full DB backups are uploaded into the cloud
- Stored into a S3 bucket for that month
- Backups remain in S3 for three months
- At the end of three months, the newest full backup for the month is archived into a yearly Glacier vault and the others are deleted
- Glacier is a long term archive for files
  - Archived files can be recalled in 24 hours or less
  - This can be expedited at an increased cost
- For all files in S3 and Glacier, AWS automatically:
  - Makes a duplicate copy in another availability zone (geographically remote location)
  - Encrypts the file as it stores it (we can also do a separate encryption)

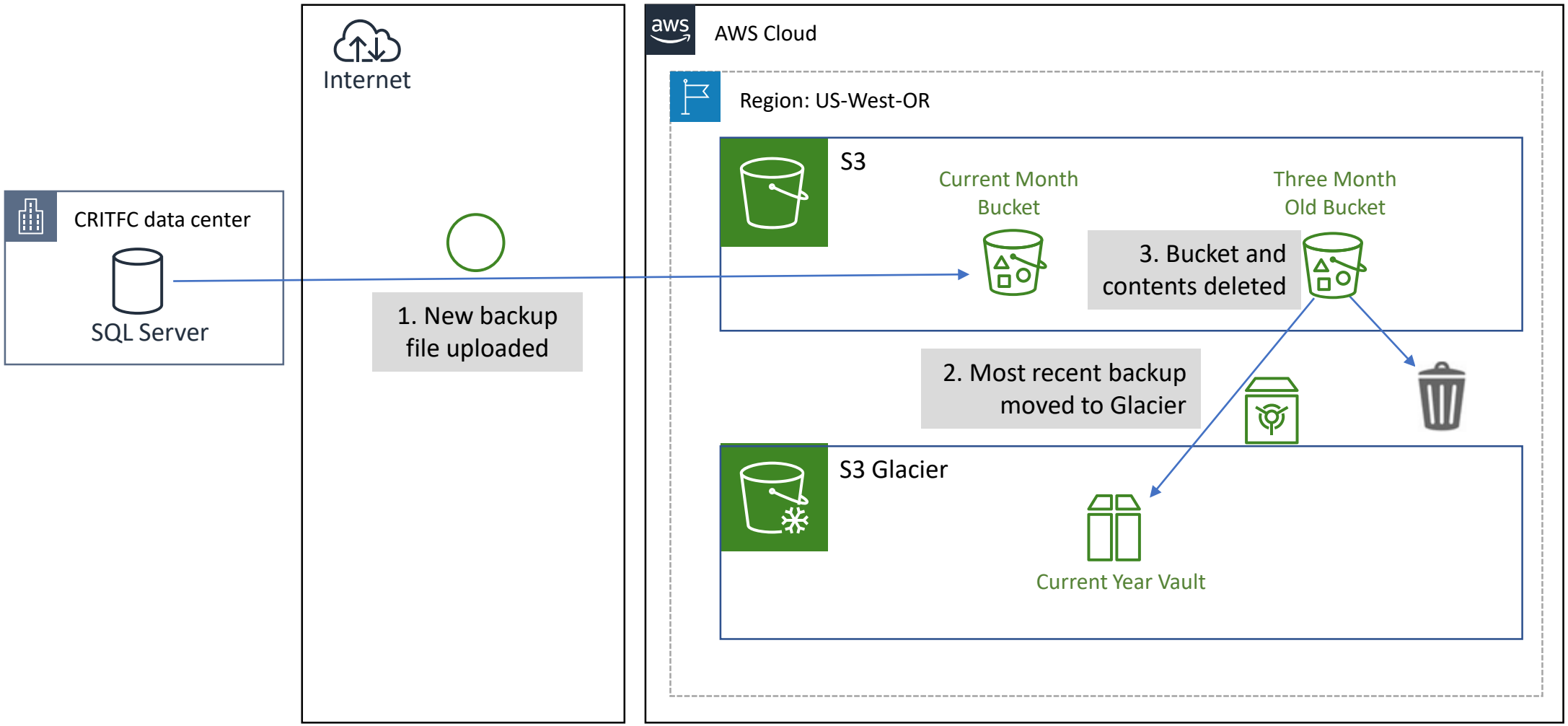
CRITFC Archiving Scheme



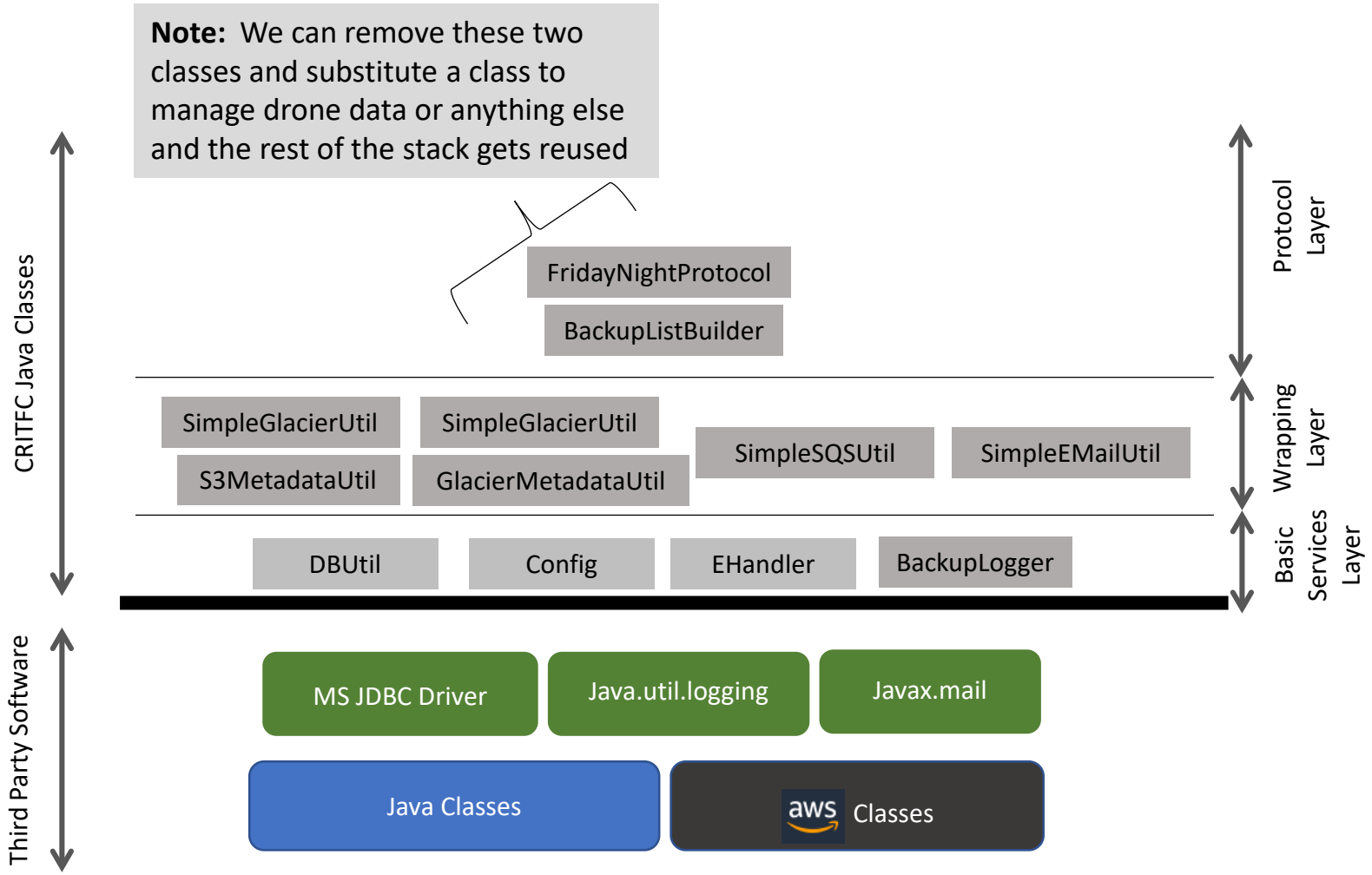
Every Friday Night



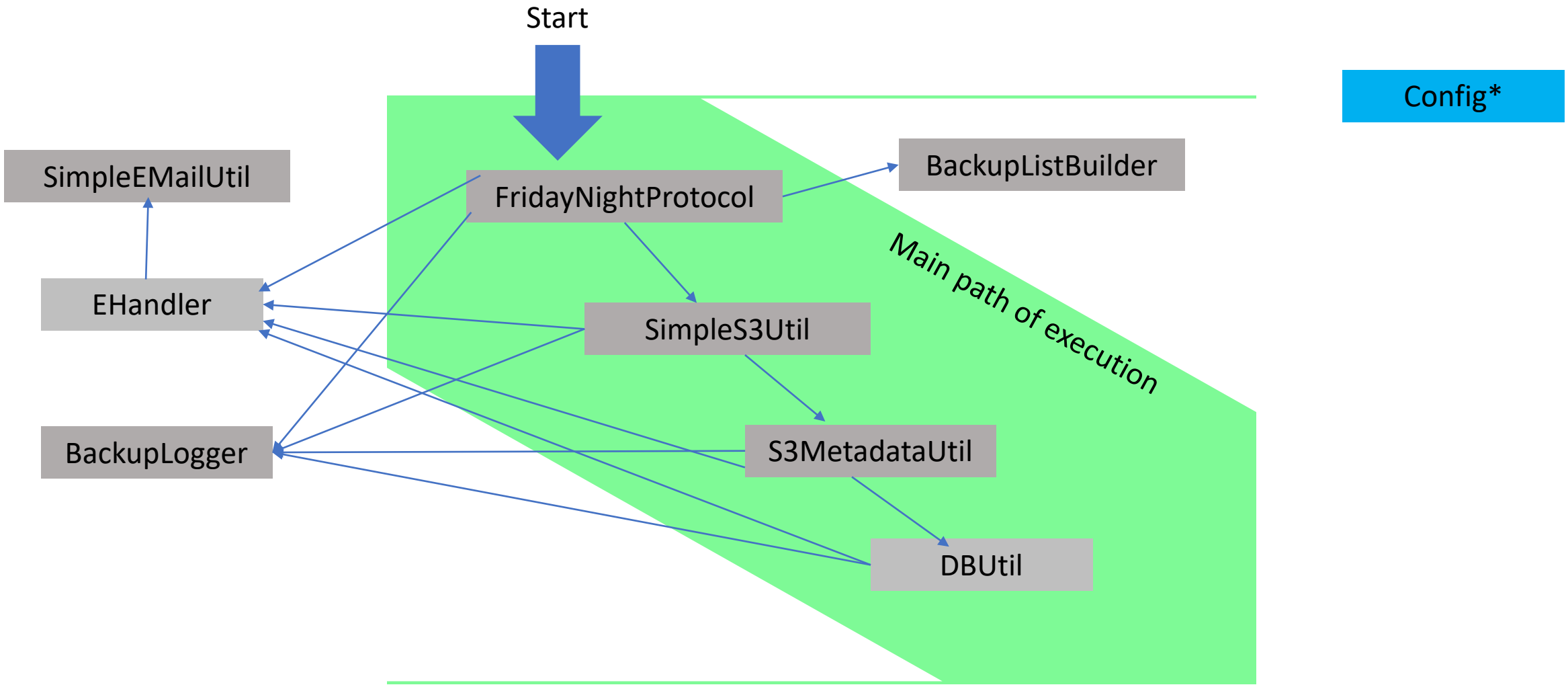
Last Friday of the Month



AWSBackup Software Stack



Calling Tree



* Config's static initializer runs at class load time before the main thread of execution launches

Config.java

- This class holds the configuration information for the application in globals
- Configuration data is held in a properties file (AWSBackup.properties)
- All the classes in the package org.critfcv.awsbackup have visibility to these properties
- The static initializer for this class acquires the path to the file from the environmental variable GLACIER-CONFIG-FILE
- It reads the file and assigns the global variables their configured values

AWS “Clients”

- This class also holds onto the AWS clients (S3, Glacier,SQS) and JDBC statement which are used in the classes
  - Initially these are `NULL`
  - They are lazily initialized when first needed
  - Other classes can reference them as:
    - `Config.S3`
    - `Config.Glacier`
    - `Config.SQS`
    - `Config.stmt`

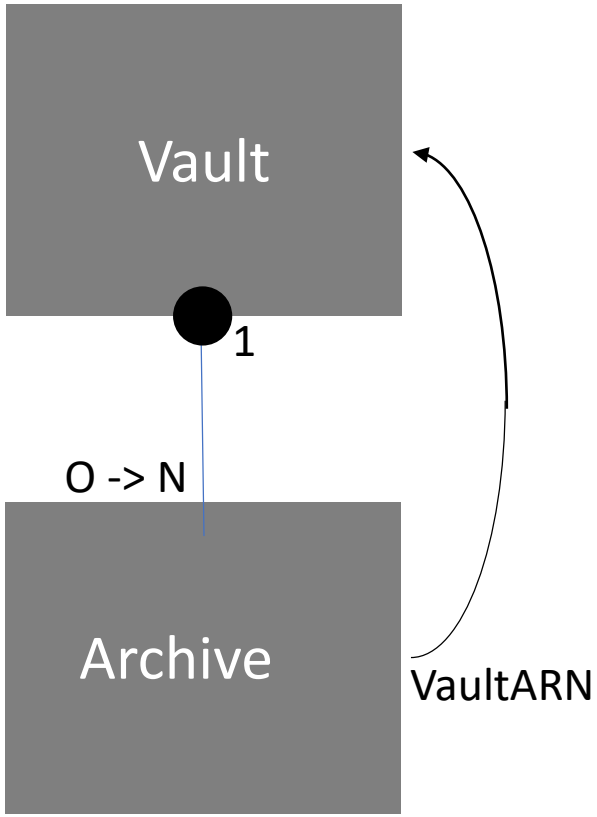
Why Build a Metadata Subsystem? I

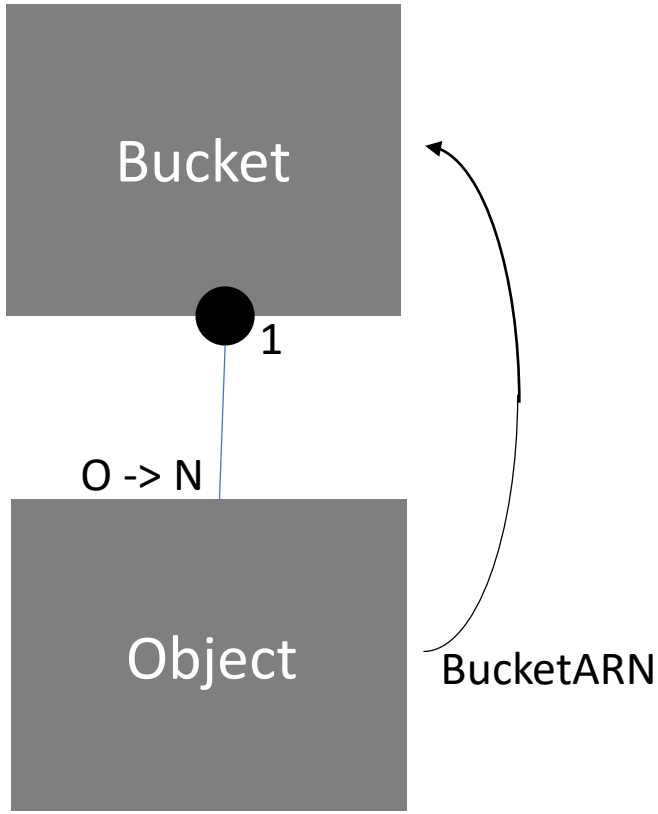
- S3 and Glacier are not ‘file systems’ in the conventional sense of the phrase
- Data is *not* organized in a series of directories and subdirectories
- S3 is basically a hashtable store
  - Hashtables have very quick access even when they grow very large
  - But you don’t recall an item from S3 by a filename, rather you use the key used to place it in the table
  - We need to track the correlation between filenames and keys – i.e., the metadata

Why Build a Metadata Subsystem? II

- Glacier is an archiving system
  - As such, archives are not always online and immediately visible
  - An archive may need to be resuscitated from an off line data store
  - As a result, it can take hours to retrieve information
    - Example: on a fairly small Glacier vault, it took 5 hours to return an inventory list
  - By keeping track of the data that goes in and out of a vault or bucket, the inventory operation can be reduced to the few seconds needed to query metadata in the database
- And there are other reasons for metadata
  - Report generation on storage use
  - Inventory reporting
  - etc.

AWSBackup Metadata Data Model





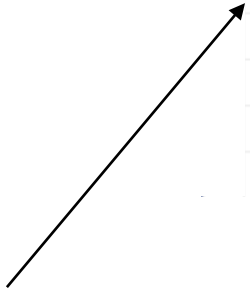
Metadata Schema: Vault and Archive

Vault

Column Name	Data Type	Allow Nulls
VaultName	varchar(50)	☐
ARN	varchar(100)	☐
CreateDate	varchar(50)	☐
Comment	varchar(MAX)	☑
CreatedBy	varchar(50)	☐
Year	int	☑
Month	int	☑
	nchar(10)	☑

Archive

Column Name	Data Type	Allow Nulls
Archive	varchar(50)	☐
ArchiveId	varchar(MAX)	☐
CreateDate	varchar(50)	☐
UpLoadedBy	varchar(50)	☑
Description	varchar(MAX)	☐
VaultARN	varchar(150)	☐
DeleteDate	varchar(50)	☑
DeleteReason	varchar(150)	☑
DeletedBy	varchar(50)	☑
Comment	varchar(250)	☑



Foreign key into Vault table

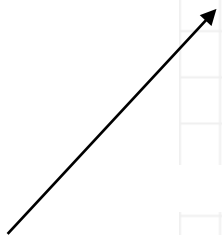
Metadata Schema: Bucket and Object

Bucket

Column Name	Data Type	Allow Nulls
BucketID	varchar(50)	☐
ARN	varchar(150)	☐
CreationDate	varchar(50)	☐
CreatedBy	varchar(50)	☑
DeletionDate	varchar(50)	☑
DeletedBy	varchar(50)	☑
Comment	varchar(MAX)	☑
Year	int	☐
Month	int	☐

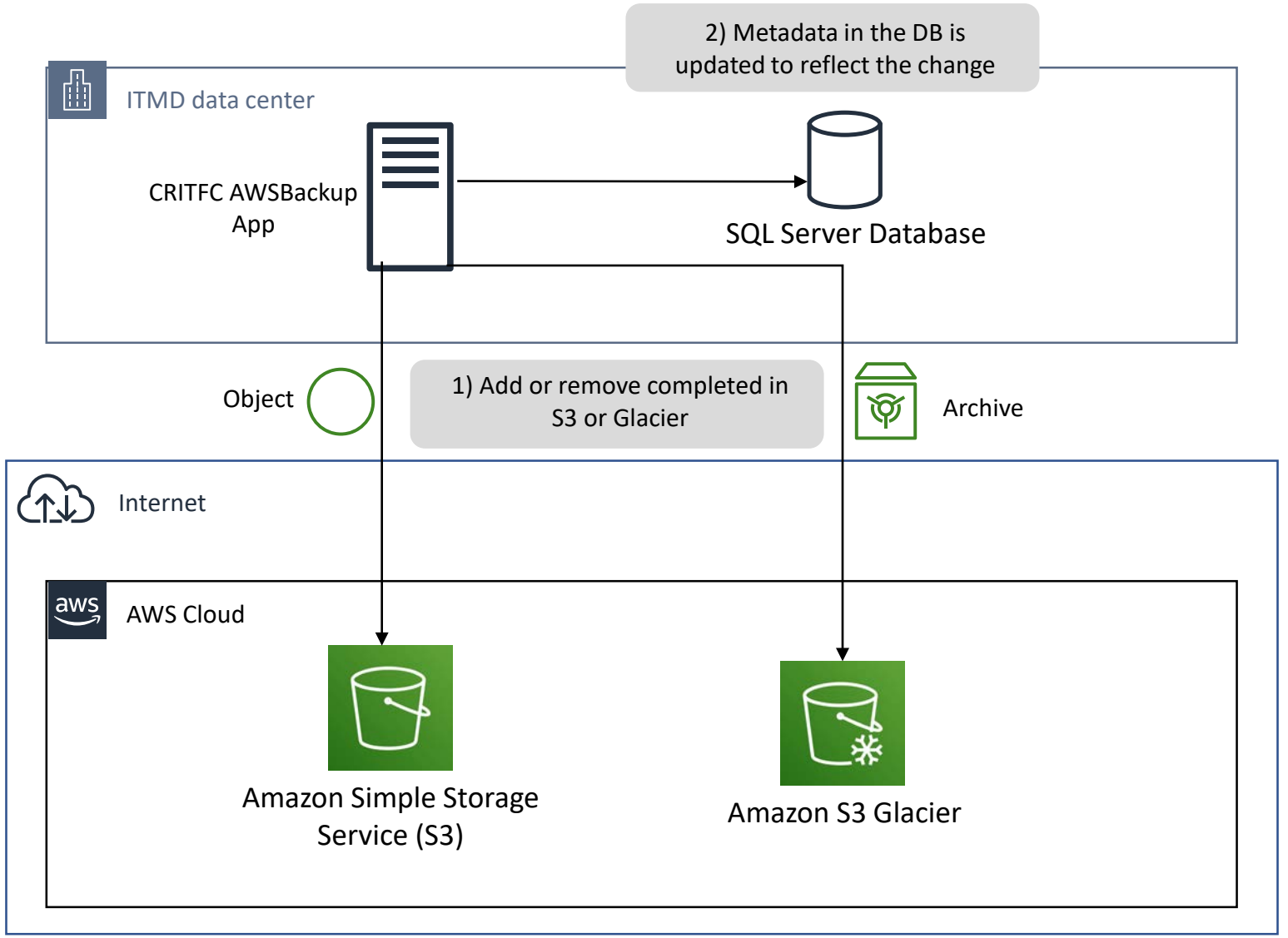
Object

Column Name	Data Type	Allow Nulls
ObjectKey	varchar(50)	☐
FileName	varchar(50)	☐
CreationDate	varchar(50)	☐
CreatedBy	varchar(50)	☐
BucketARN	varchar(150)	☐
DeletionDate	varchar(50)	☑
DeletedBy	varchar(50)	☑
Comment	varchar(MAX)	☑



Foreign key to Bucket table

Metadata Subsystem



```
public static void createVault(String vName, String IAM, String comment) {

    if(Config.VAULT_SIM)
        System.out.println("SimpleGlacierUtil.createVault >> attempted to create vault " + vName);
    else {
        try {
            getGlacier();

            vName = Config.VaultNameBase + vName;
            CreateVaultRequest cvr = new CreateVaultRequest(Config.GlacierIAM, vName);
            cvr.setAccountId(Config.AWSAccount);
            CreateVaultResult cvrs = Config.glacier.createVault(cvr);

            if(Config.DEBUG)
                System.out.println("SimpleGlacierUtil.creastValut >> create vault result: " + cvrs.toString());

            // get vault description and parse ARN out of it
            String summary = vaultSummary(vName);
            StringTokenizer st = new StringTokenizer(summary, ",");
            String token = st.nextToken();
            int start = token.indexOf("arn:");
            String ARN = token.substring(start, token.length() -1);

            if(Config.DEBUG)
                System.out.println("Vault ARN is: " + ARN);

            GlacierBackupMetadata.Vault abm = new GlacierBackupMetadata.Vault();
            abm.vaultName = vName;
            abm.ARN = ARN;
            abm.createDate = (new Date(System.currentTimeMillis())).toString();
            abm.comment = comment;
            abm.createdBy = IAM;

            GlacierBackupMetadata.recordVault(abm);

        }
        catch(Exception ex) {
            eh.handleErrorAndExit(ex, "SimpleGlacierUtil.createVault >> error during valult creation operation");
        }
    }
```

Using AWS Classes Directly

Example from SimpleGlacierUtil:

1 line of code call on Util
VS
41 lines of code in the Util

Next Steps

- Framework has been written but there has been no time to do the needed testing yet
  - Need to write test cases and verify functionality
  - Then need to put it into production and validate results for a couple of months
- When testing is done, the framework will be available to everyone
  - I plan to move it to GitHub (if CRITFC okays this)
  - If anyone wants a copy of the code just to look through, I can arrange that
- Next there will be another protocol added to allow the Habitat folks to up and download drone files

Facts to Know About AWS

- If you want to work with the framework you will have to have an account with AWS
  - button to get account is on the AWS console page (URL below)
  - you can get a completely free account for one year
  - you are (a little) limited on what resources you can use without paying
  - it's a good way to learn the technology
- You will also need to become familiar with navigating and using the AWS console: https://aws.amazon.com/console/
- The framework is written in Java so you need to install Java and Eclipse (both free and easy to install)
- AWS has lots and lots of high quality documentation and tutorials
- Make sure you read about and use the latest release of the software
  - The first version of the software is still supported and available
  - This is because long time customers have deployed systems with it

AWS Management Console

AWS services

Find Services
You can enter names, keywords or acronyms.

Example: Relational Database Service, database, RDS

Recently visited services

- Support
- S3
- EC2
- S3 Glacier
- Cloud9

All services

Build a solution

Get started with simple wizards and automated workflows.

Launch a virtual machine
With EC2
2-3 minutes

Build a web app
With Elastic Beanstalk
6 minutes

Build using virtual servers
With Lightsail
1-2 minutes

Connect an IoT device
With AWS IoT
5 minutes

Start a development project
With CodeStar
5 minutes

Register a domain
With Route 53
3 minutes

Deploy a serverless microservice
With Lambda, API Gateway
2 minutes

Create a backend for your mobile app
With Mobile Hub

Access resources on the go

Access the Management Console using the AWS Console Mobile App. Learn more

Explore AWS

Open Distro for Elasticsearch

A 100% open-source, community driven distribution of Elasticsearch with enterprise-grade security and alerting features Learn more

Run Serverless Containers with AWS Fargate

AWS Fargate runs and scales your containers without having to manage servers or clusters. Learn more

Visit AWS around the world at a Summit

AWS Global Summits bring the cloud computing community together to connect, collaborate, and learn about AWS. Learn more

Amazon RDS

Set up, operate, and scale your relational database in the cloud. Learn more

The View from 50,000’

Amazon S3 Glacier

Amazon Simple Storage Service (S3)

Object

Vault

Bucket with objects

Archive

Bucket

Disk

Generic database

Internet