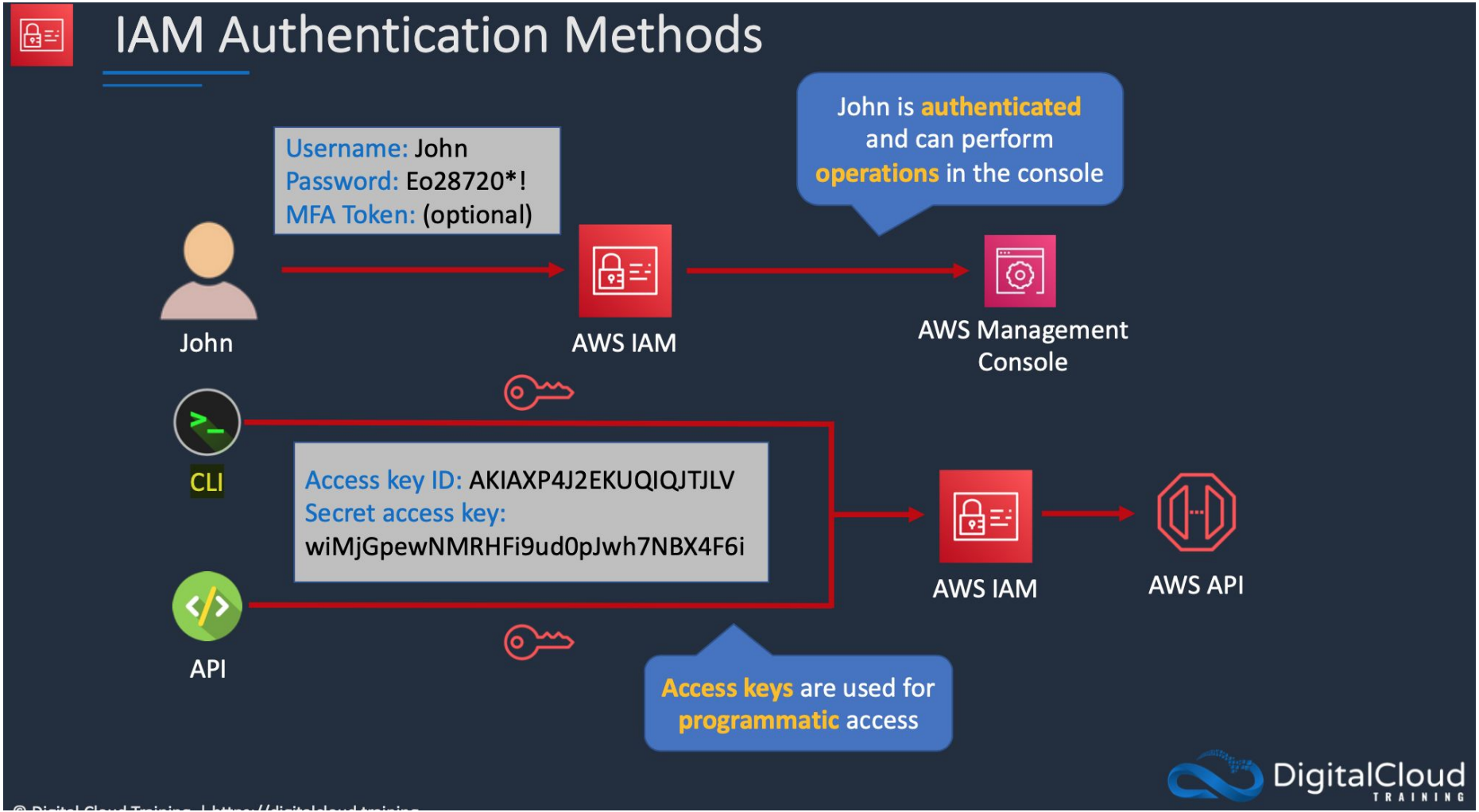
IAM Authentication Methods

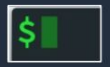


S3 Presigned URLs

https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html

AWS S3 CLI command to generate a presigned URL

```
aws s3 presign s3://dct-data-bucket/cool_image.jpeg
```



```
https://dct-data-bucket.s3.ap-southeast-
2.amazonaws.com/cool_image.jpeg?X-Amz-Algorithm=AWS4-HMAC-
SHA256&X-Amz-Credential=AKIA3KSVPHP6MAHNW5YH%2F20200909%2Fap-
southeast-2%2Fs3%2Faws4_request&X-Amz-Date=20200909T053538Z&X-
Amz-Expires=3600&X-Amz-SignedHeaders=host&X-Amz-
Signature=8b74653beee371da07a73dfdb4ff6883742383afa528aecd5c95
c326c97764db
```

This is the response; the URL expires after 1 hour

NAME

https://git-scm.com/docs/git-branch

git-branch - List, create, or delete branches

SYNOPSIS

```
git branch [--color[=<when>] | --no-color] [--show-current]
        [-v [--abbrev=<n> | --no-abbrev]]
        [--column[=<options>] | --no-column] [--sort=<key>]
        [--merged [<commit>]] [--no-merged [<commit>]]
        [--contains [<commit>]] [--no-contains [<commit>]]
        [--points-at <object>] [--format=<format>]
        [(-r | --remotes) | (-a | --all)]
        [--list] [<pattern>…]
git branch [--track[=(direct|inherit)] | --no-track] [-f]
        [--recurse-submodules] <branchname> [<start-point>]
git branch (--set-upstream-to=<upstream> | -u <upstream>) [<branchname>]
git branch --unset-upstream [<branchname>]
git branch (-m | -M) [<oldbranch>] <newbranch>
git branch (-c | -C) [<oldbranch>] <newbranch>
git branch (-d | -D) [-r] <branchname>…
git branch --edit-description [<branchname>]
```

Special bash parameter	Meaning
$!	$! bash script parameter is used to reference the process ID of the most recently executed command in background.
$$	$$ is used to reference the process ID of bash shell itself
$#	$# is quite a special bash parameter and it expands to a number of positional parameters in decimal.
$0	$0 bash parameter is used to reference the name of the shell or shell script. so you can use this if you want to print the name of shell script.
$-	$- (dollar hyphen) bash parameter is used to get current option flags specified during the invocation, by the set built-in command or set by the bash shell itself. Though this bash parameter is rarely used.
$?	$0 is one of the most used bash parameters and used to get the exit status of the most recently executed command in the foreground. By using this you can check whether your bash script is completed successfully or not.
$_	$_ (dollar underscore) is another special bash parameter and used to reference the absolute file name of the shell or bash script which is being executed as specified in the argument list. This

Special bash character	Meaning
#	# is used to comment a single line in bash script
$$	$$ is used to reference process id of any command or bash script
$0	$0 is used to get the name of the command in a bash script.
$name	$name will print the value of variable “name” defined in the script.
$n	$n will print the value of nth argument provided to bash script (n ranges from 0 to 9) e.g. $1 will print first argument.
>	> is used to redirect output
>>	>> can be used to Append to file
<	< will redirect input

Uptime is a command that returns information about how long your system has been running together with the **current time**, **number of users with running sessions**, and **the system load averages** for the past **1**, **5**, and **15** minutes. It can also filter the information displayed at once depending on your specified options.

It will display an output similar to:

```
09:10:18 up 106 days, 32 min, 2 users, load average: 0.22, 0.41, 0.32
```

Check Linux Server Uptime

You can filter uptime's result to show only the running time of the system with the command:

```
# uptime -p

up 58 minutes
```

Check Linux Server Starting Time

Using option `-s` will display the date/time since when the system has been running.

```
# uptime -s

2019-05-31 11:49:17
```

1. Search any line that contains the word in filename on Linux:

   ```
   grep 'word' filename
   ```

2. Perform a case-insensitive search for the word ‘bar’ in Linux and Unix:

   ```
   grep -i 'bar' file1
   ```

3. Look for all files in the current directory and in all of its subdirectories in Linux for the word ‘httpd’:

   ```
   grep -R 'httpd' .
   ```

4. Search and display the total number of times that the string ‘nixcraft’ appears in a file named frontpage.md:

   ```
   grep -c 'nixcraft' frontpage.md
   ```

Amazon Simple Storage Service (S3)

S3 Bucket

A **bucket** is a container for objects

Object

Object

Object

An **object** is a file you upload

You can store millions of **objects** in a **bucket**

Accessing objects in a **bucket**:

```
https://bucket.s3.aws-region.amazonaws.com/key
https://s3.aws-region.amazonaws.com/bucket/key
```

The **HTTP protocol** is used with a **REST API** (e.g. GET, PUT, POST, SELECT, DELETE)

DigitalCloud TRAINING

Amazon Simple Storage Service (S3)

- You can store any type of file in S3
- Files can be anywhere from 0 bytes to 5 TB
- There is unlimited storage available
- S3 is a universal namespace so **bucket names** must be **unique globally**
- However, you create your buckets within a **REGION**
- It is a best practice to create buckets in regions that are physically closest to your users to reduce latency
- There is no hierarchy for objects within a bucket
- Delivers strong read-after-write consistency

Buckets, Folders, and Objects

- Folders **can** be created within folders
- Buckets **cannot** be created within other buckets
- An objects consists of:
  - Key (the name of the object)
  - Version ID
  - Value (actual data)
  - Metadata
  - Subresources
  - Access control information

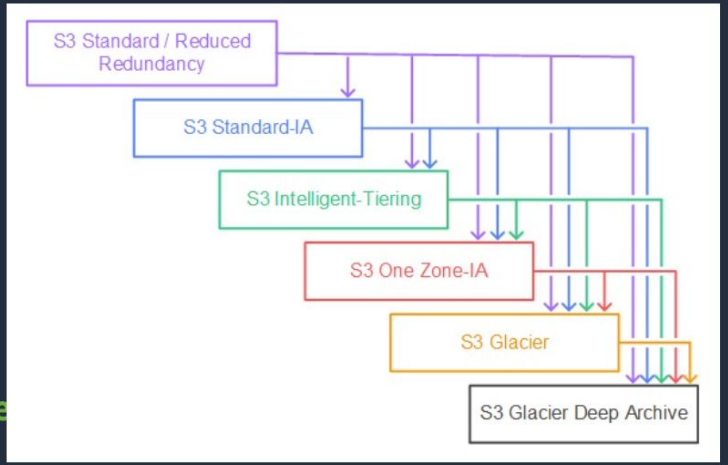
S3 Storage Classes

	S3 Standard	S3 Intelligent Tiering	S3 Standard-IA	S3 One Zone-IA	S3 Glacier	S3 Glacier Deep Archive
Designed for durability	99.999999999%	99.999999999%	99.999999999%	99.999999999%	99.999999999%	99.999999999%
Designed for availability	99.99%	99.9%	99.9%	99.5%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99.9%	99.9%
Availability Zones	≥3	≥3	≥3	1	≥3	≥3
Minimum capacity charge per object	N/A	N/A	128KB	128KB	40KB	40KB
Minimum storage duration charge	N/A	30 days	30 days	30 days	90 days	180 days
Retrieval fee	N/A	N/A	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved
First byte latency	milliseconds	milliseconds	milliseconds	milliseconds	select minutes or hours	select hours
Storage type	Object	Object	Object	Object	Object	Object
Lifecycle transitions	Yes	Yes	Yes	Yes	Yes	Yes

S3 LM: Supported Transitions

You can transition from the following:

- The **S3 Standard** storage class to any other storage class
- Any storage class to the **S3 Glacier** or **S3 Glacier Deep** Archive storage classes
- The **S3 Standard-IA** storage class to the **S3 Intelligent-Tiering** or **S3 One Zone-IA** storage classes
- The **S3 Intelligent-Tiering** storage class to the **S3 One Zone-IA** storage class
- The **S3 Glacier** storage class to the **S3 Glacier Deep Archive** storage class



Architecture Patterns – Amazon S3

Requirement	Solution
Company is concerned about accidental deletion of Amazon S3 objects	Enable S3 versioning
Data stored in S3 is frequently accessed for 30 days then is rarely accessed but must be immediately retrievable	Use a lifecycle policy to transition objects from S3 standard to S3 Standard-IA after 30 days
A backup of S3 objects within a specific folder in a bucket must be replicated to another Region	Configure cross-region replication and specify the folder name as a prefix

Architecture Patterns – Amazon S3

Requirement	Solution
Previous versions of objects in a versioning-enabled S3 bucket must be stored long term at the lowest cost	Create a lifecycle rule that transitions previous versions to S3 Glacier Deep Archive
A company wishes to manage all encryption of S3 objects through their application with their own encryption keys	Use client-side encryption with client managed keys
Unencrypted objects in an Amazon S3 bucket must be encrypted	Re-upload the objects and specify the encryption an encryption key

Amazon EBS

- EBS volume data persists **independently** of the life of the instance
- EBS volumes do not need to be attached to an instance
- You can attach multiple EBS volumes to an instance
- You can use multi-attach to attach a volume to multiple instances but with some constraints
- EBS volumes must be in the **same AZ** as the instances they are attached to
- Root EBS volumes **are deleted** on termination by default
- Extra non-boot volumes **are not deleted** on termination by default