**AWS Cloud Practitioners Essentials -**

1- Overview

2- What is cloud computing - Use and manage a scalable online data center

3 Scalability - resize resource’s as needed

4- Elasticity - Create scale and shut down new resource’s as needed

5 auto scale and elastic load balancing allow you to have this whole process automated

6 fault tolerance- the built in redundancy that helps you run even when you have errors

7 Your data is always owned by you - in AWS

8 Using AWS services-

**AWS Management Console - UI - Navigation and usability**

Aws services tools

AWS resource groups help cluster these services

Resource groups are user specific but can be shared

you can also tag the resource’s in groups for searching

It make time for your feature updates to take effect

AWS CLI - control from command line- flexibility to create scripts

make commands like to any remotely running EC2 instance

AWS SDK’s - Software development kits to use AWS with - the ability to use AWS in existing application’s

the SDK allows you to manage you infrastructure as code

All of these perform their actions by accessing AWS API's

**AWS EC2**

What is EC2 Elastic compute Cloud - amazon server instance tools pay as you go

How to build and configure a ec2 instance

login AWS

choose region

launch ec2 wiz

select AMI-

The OS of the server or image if you have one the server will use

select instance type

hardware selection - like t2 micro or 64 core hard drive

config network

how many instances running - 1 -to- 100000 max

setting for cloud and sub net - Note default is good here unless client or boss says otherwise

default auto assign settings which give us a DSC2 address- everything else usually good as well

Configure storage

set the gigs of storage and the type of volume

you can have multiple volumes configured at once

Make sure you set volumes to be deleted automatically if you choose at any point to terminate the instance

then add the TAG - name what you want

Security rules- SSH connectivity- or simple web coactivity

Save SSH key to computer

Last Review and launch your new EC2 instance

Configure key pairs

by creating a new key pair you get the info on how to connect to your services

Launch and connect

Hit launch to launch - once pending changes to running we can connect with the instance

highlight and copy public DNS -

launch a tool called Putty that you have downloaded on to your computer -

Host name = ec2-user@ ( paste in the DNS )

Then use Putty gen + where you saved your SSH key from before to make a ppk file

set this new ppk file as the SSH security key in putty and you should be able to log in with no problems

**AWS EBS Volumes**

EBS - Volumes to store your eC2 instances

pay as you go -

take snapshots of volumes at times for backup

full encryption options for data transit

EBS volumes can change the type need to hold all the EC2 instances

using EBS volumes

Volumes are located on the side bar when you are accessing EC2

Volumes must be in the same availability zones as their EC2 instances

Volume Config

General purpose SSD is best Volume type as it only changes for GB used

Snapshot ID section lets you restore a snapshot to a volume

after that just click create Volume

actions attach volume - and then specify instance attaching and device

from there you can go to your instance copy the SSH commands and run it in your terminal

on mac the command lsblk -will let you see all the instances you have running in this volume

by attaching and mounting a specific instance with root permissions you can begin cd’ing in and start accessing or creating files

You can also select and detach any volume at any time -

it will still be available once you have detached it

you can also tag your volumes

tags on AWS resources let you examine billing based on those tags

**AWS S3 -**

the data in S3 is not associated with ant specific server it’s all over the world

you can retrieve data any time anywhere - several terabytes in size

by default none of your data is shared publicly

you can also encrypt data in transit

we take an object like a video - and give it a key

backups are automatic when you make a bucket

S3 will scale data size and availability of requests

Access - you can access the data from the console the AWS CLI or AWS sdk's

you can also access your data through a rest service endpoint

this is an http request / (region specific endpoint)/ Object Key

S3 bucket names must be unique-

Use cases -

1 storing application assets - like server logs

2 for static web hosting s3 can serve up the content of your files

3 cross region replication makes it great a database backup tool

4 Staging area for Big data - you can use Red shift EMR amazon Athena or import export large data with snowball

5 It may be the most useful single tool in AWS

How to create a new S3 bucket

go to s3 section of AWS

hit create bucket

set name- must be DNS compliant

set region-make sure matches any EC2 instance that might be trying to access the data

and this is all we need to get started -

Interacting with the S3 bucket in UI

click on your target bucket

once you hit upload you can easily drag and drop files in the bucket

Interacting with the S3 bucket in CLI

target a file

use the (s3 copy command) to copy an object that lives in your s3 and give it a key to add it to that s3 instance

you can also sync a folder with the s3 bucket and the CLI will take those files and check to see if they exist in the bucket

if they don’t it will upload them - this is the best way to ensure no duplicate objects are in your s3 buckets

From here if we ssh into an EC2 instance we can see if it has been given IAM role to manage buckets what buckets its managing

using ls for macs onto that instance will allow you to see all the files in that bucket

from here you can use copy commands to bring the files back from s3 over to your main computer or ec2 instance

you can use the sync commands in reverse as well -

Back in the UI

when accessing a file in s3 you have options

you can modify properties and permissions you can also see attributes of the file

**AWS Global Infrastructure-**

Regions - locations hosting availability zones

these determine optimizing latency and reducing costs

you could deploy major resource’s in one location then deploy tools to interact with it in another

if one zone goes down others can pick up the request-

you should deploy your content across different locations whenever possible

edge locations are locations with quick access to high population areas

**Amazon VPC virtual private cloud**

this allows you to create a virtual private network within the aws cloud

vpc allows you complete control over the security and configurations - such as routing rules-

numerous services work directly with Amazon VPC

amazon ec2's are deployed into your amazon VPC-

amazon RDS or regional database service deploy directly into VPC -others work similarly to this

Amazon VPC's live within regions and can span across multiple AZ - availability zones

Subnets -

these allow the VPC to span multiple AZ's but fewer is recommended to make the service as simple as possible

Route tables control traffic between subnets and the internet- all sub nets have access to each other’s

subnets are either public or private - this means access to the internet or no access

Building a VPC

go to VPC section of AWS

select region

select name of VPC

set the IP address

next build a subnet - name it

specify subnet locations

continue to create all subnets-

add in an internet gateway to the subnet's you want to be public

name the gateway

**AWS Security Groups**

act as built in fire walls

control access to instances

you can have different security groups with different levels of access

security group will be on the left of your service

clicking on it will allow you to adjust security group settings on your security groups or create new ones

Creating a security group- Name - description - Rules

default - inbound traffic in denied all outbound traffic is allowed -you can edit this

The best practice is to figure out what traffic is required for your service and only allow it

you can also set - type - protocol - port rage - Source - Description

by setting sources to all 0 you allow all ip address’s

by adding multiple rules you can designate between things like http and https requests security settings

**AWS Application Load Balancer**

replaces the classic load balancer with new features

Supported Protocols- Cloud watch metrics - Access Logs - Health Checks

additional features like path and host based routing and native ipv6 support included

Application load balancer allows you route different requests to the same ec2 instance or other service target

you set up rules to target what you want

ALB -terms

Listeners - listen for requests

target -is the target of the requests

Target Groups- hold multiple targets for requests

Creating an Application load balancer

go to EC2 in aws services

go to a running instance

copy the ip address of the instance

if they are sites you can check to see if they are listening correctly with the ip address plus domain name

next go to load balancers on the left side

click create load balancer

you can then choose application load balancer or the classic load balancer

set name

set scheme- internet facing

set ip address

set listeners- set them to the ports you want to listen for

than choose the availability zone

then set a key

then configure security settings

then configure security group

then configure routing - for target groups

next we can set the heath check settings - every 10secs is good

next we register the targets

then add to register

then we get to the review page - where we see everything we have just configured

after that we can just hit create

with two targets you must create a target group

click create target group

set name

set request type

set health check

use view and edit rules to change things like forwarding address targets

next we can check to see if it is working by copying the DNS name and checking the http request in the browser

to adjust the listeners go to the listeners tab

**AWS Auto Scaling**

checks the available EC2 instances to manage your workload

cloud watch checks values but only Auto Scaling can add more to your network

you can set the conditions for auto scaling for the days you need more performance versus less

scaling out is adding more scaling in is adding less

1 create launch configuration

2 create an auto scaling group - min/max instances

3 policy- do you scale on a time schedule or based on use

4 dynamic auto scaling can be set to go off on cloud watch alarms

5 you should have alarms to both add and remove instances

Let’s Build a demo

open EC2

choose auto scaling groups

choose ec2 conditions

go to launch instance conditions

choose instance type

give launch condition a name

choose an existing key pair

choose properties for auto scaling group

specify instances

specify subnet deployment

set configure scaling policy’s to adjust capacity of group

set min max instance values

set target tracking policy - like cpu usage 60%

hit crate auto scaling group

go to the instance tab you will see pending your min instances already being generated

if you go back to the auto scaling group you can modify these min max instances based on launch config

**Amazon Route 53**

DNS service or Domain name service to route end users to endpoints

how does it work - user puts in address redirects to route 53 sends back info on a target address they need to go

you an create hosted zones to help route 53 send your users back to the right address

Let’s create a DNS service

go to services on the left choose hosted zones

hit create hosted zone

set the domain name

set type - like public hosted zone

create a new record set - choose sub domain

choose routing policy

then hit create

Now you can see your instance resolving names to your desired instance target address

Route 53 supports a wide variety of DNS resolution options -

works with ipv4 and ipv6 instances

**AWS RDS- (Relational database service)**

running your own relational database has many factors - making in cost time and money

AWS provides a service that automatically handles all the major admin tasks of your db.

with RDS you can just worry about your data

scaling data backup and maintenance are all covered by amazon

what is a RD instance- a individual db. that you can manage scale and control all from aws services

you can run it using the VPC service-

you can run it on its own -

when you configure it you can choose the availability zone used

when you create an amazon RDS a standby back up copy is created as well all interactions are synced to the standby copy

its recommended to run your RDS in multiple AZ's

Amazon RDS supports the creation of read replicas - for my SQL - Maria DB and others

using read replicas you can also scale out beyond the capacity of a single database instance

read replicas can be promoted to be the master db. instance

amazon RDS is perfect for most web application’s, standalone db's, e commerce sites as well as online games

the service can be managed with the AWS UI, CLI, or any API

**AWS Lambda**

aws lambda is a compute service that lets you run code without provisioning or managing services

scales and mages you code

no servers to manage - Continuous Scaling - Sub second Metering

aws lambda runs your code on a highly available compete infrastructure - this provides all administration

capacity provisioning and auto scaling are also handled

also aws lambda supports a variety of programing langue’s like python and C#

best use is to run code in response to events

you can respond to s3 changes or even http requests

you can also build server-less application’s that respond to aws lambda functions

auto deploy with aws code pipeline and aws code deploy

To use -

1 Configure Lambda environment -

2 Upload code

3 Watch it run

config Lambda environment

In the lambda section you can see all lambda functions you have created

choose entry type- Runtime language - and the code handler

set environment vars - for encryption - set tags - choose an execution role- set memory execution types

then set the trigger policy such as cloud watch to trigger your code

next the monitoring page will show you when your function has run

you can also use aws lambda combined aws kinesis to process real time steaming data - and much much more

you can also build aws lambda to handle your extract transform and load pipelines

you can also use aws lambda to build the backend for your IOT devices

you can also use to run the entire backend of your site when combined with other aws services

AWS lambda is thought of as the connective tissue between all other aws services

**AWS Elastic Beanstalk**

AWS beanstalk is a platform as a service

so you can place your code over to launch your application quickly

choose your instance type choose your data base choose your auto scaling levels

update your application Access server logs Enable HTTPS load balancer

EB supports a large range of platforms as well as code languages

this is the bean stalk stack

1 your code-

2 Application service

3 HTTP service- Operating system

4 Language interpreter

5 Host

create application - upload version - Launch Environment - Mange Environment

Update your application as easily as you deploy it

Use case - launch python website

go to elastic beanstalk in the dashboard

create a new application- and then name it - give it a description

choose the environment

set up your environment configuration

than set up your environment’s base config including - tier - platform an application code

advanced config can let you change this like software instances and load balancing

then hit create- and the log will show you all the pieces of your application being generated

after your done you have a dashboard that shows you your whole application hosting

even the url for your site if that is what you were creating

**AWS Cloud Watch**

Monitoring service to monitor resources in real time

Collecting and tracking metrics, motoring cloud resources and set alarms to notify you or take actions

Cloud watch logs can keep track of major changes that don’t need immediate alarms

Cloud watch metrics include CPU use – then are sent the AWS management system

These notifications can trigger an auto scaling event -

These events can also trigger AWS lambda functions- you can even direct specific records to a Kinesis live stream

Metrics - alarms –events - logs

Metrics – data about the performance system – many of these reports come free with things like EC2 instances – you can also publish your own application metrics – or load all metrics into your account

Alarms – watch a single metric – and it can be set to watch for gradual changes over time- it invokes actions for sustained state changes only – can trigger events – take actions or activate lambda code

Actions – Scale out Scale in Stop restart or shut down

Amazon cloud watch events- simple rules route events to functions or streams you can also schedule events to trigger at certain times.

Example – detect and revoke IAM access – API call records event and if the user is unauthorized and then use a lambda function to revoke access

Cloud watch logs – logs can keep track of everything cloud watch monitor’s for instance all info on who did what when – you can also store and monitor your data and then save all this to S3 and review and take further action latter

Cloud watch dashboards – this is a customizable homepage within cloud watch that can monitor all your currently active CloudWatch recourse’s in a single view – you can set custom views of your metrics –

Short demo example-

Go to cloud watch

Go to cloud watch dashboard –

Create a new cloud watch event rule

Set the event source of the event

Create a simple lambda function that is going to log the state change of an EC2 instance

This function should run whenever you launch an EC2 instance

Add a target and then choose than function you just created

Next set up a cloud watch alarm for an EC2 instance

Go to EC2/instances/alarms /cloud watch alarm

Then hit create cloud watch alarm

Set notification call it my topic with a recipient email address –

Set the action we are going to take – ie- bases metric on cpu utilization

Then set the alarm

Then add a custom dash board name it – add in line stack area – Numbers- and text to have all these ways of viewing your data

**AWS cloud front**

What is cloud front to deliver content to your users – with edge locations around the world

This allows local content caching around the world to reduce latency

It is a CDN content delivery network

It Is deeply integrated with other AWS services such as S3

The easiest way to start is to go to console and hit create distribution

First select delivery method – web – or – RTMP

Next specify 1 origin and one behavior- origin = s3 or any endpoint- behavior what you want to happen with that origin

Example use s3 bucket as origin for a group of images

Next set origin settings such as http port and click create distribution

Use cases- static assets caching – API acceleration- and anything really that involves caching data and serving data

Example – when severing a web app that is hosted in Singapore to users in England you can create a cloud front instance that tracks the ip address of your web service as an endpoint – give it a alternate domain name – and then run the service

This can help you as content is still served from the cache even if your original web server crash’s

**AWS Cloud Formation**

What is cloud formation – simplifies the task of creating groups of the things that power your services

This is a resource management service

In the UI console – CLI or SDK we can run Cloud formation to – automate resource management

1 – cloud formation is a fully managed service so there is no infrastructure to support

2- cloud formation can create update and delete instances – in sets known as stacks to optimism performance

3 Cloud formation uses template files to provision resources that are then placed in a stack

We use with cloud formation with templates and those then create stacks they are text files written in json or yaml format

The template uses specific formatting constructs but all these rules can be set in the console as well

A template can accept parameters and conditions to allow it to build more than just one type of environment

What happens to all these templates – they serve as infrastructure as code – they are like source code for applications

2 requirements for running cloud front – 1 working template files- 2 permissions to all things called in the template

-use case example – we will cloud formation designer–

Go to cloud formation in the management console

Click create new stack

Design template is what we select

We come to the interface for the cloud formation designer

Go to EC2 – VPC and drag it into the screen

This is how we build it out – now add vpc property’s – name – cyder bock –

Now we have the basics to create one resource

Now we will say we want to create this stack

Template is auto add to an s3 bucket – now we name the stack we just made

Last we can add permissions to the stack we made then we set the create state

Back in cloud formation overview we can now see our new stack running and if we go to VPC we can see the VPC our stack has created for us –

**AWS Guide to a Well Designed System-**

5 pillars – Security – Reliability – Efficiency – Cost – Operational excellence

1 – Security-

IAM – authentication

Detective controls

Infrastructure protection

Data Protection – backups and replication

Incident response – respond and investigate issues

1st security design principle – apply security at all layers – within and between your resources

2nd only allow permeations as needed

3rd Enable traceability like with logs

4th set up automated security as much as possible

2 pillar Reliability –

In order to be reliable your systems need a strong foundation

In a cloud environment you can take advantage of automation though monitoring

1 Test recovery procedures – 2 Enable Auto Recovery- 3 Scale Horizontally – 1 large bad – multiple small good 4 Never Guess Capacity- Always monitor and scale – 5 Manage change and Automation

3 pillar Performance-

1 Selection - choose the best tools 2 review make sure you are still using the best tools

3 Automate monitoring with things like cloud watch 4 Tradeoffs – always get Space to buy time

1 Democratize advanced tech 2 go global in minutes 3 Go Server less 4 Experiment as much as you can

5 mechanical sympathy – align tools to goals

4 Cost Optimization

1 Use cost effective resources 2 match supply and demand 3 increase expenditure awareness 4 Optimize over time

1 Adopt a consumption model – pay for what you use 2 measure efficacy make sure you know your performance 3 Don’t spend money on physical data infrastructure 4 analyze attribute expenditure

5 use managed services to reduce the cost of owner ship-

5 Operational excellence pillar

1 manage and automate changes 2Respond to events 3 define your standards

**Fault Tolerance –**

The ability to remain operational even if something fails

High Availability

Make sure down time is minimized as much as possible automatically

Multiple servers – Availability Zones – in regions multiple regions as well as fault tolerance tools

1 Elastic load balancer’s-

Distribute traffic among your systems – notify you if high server traffic

2Elastic IP address

These allow for better load balancing –

These are IP address designed for flexible cloud computing

High availability is ensured as sites are still up if they fail

3 Amazon route 53

This DNS service allows for the highest availability possible for your applications

4 Auto Scaling

This allows for you to auto generate new resources on demand

5 Amazon Cloud Watch

Gathers metrics on all systems has tools to report and trigger evets to help with availability

Fault tolerance serves

1 Amazon SQS simple que service

A highly reliable distributed messaging system – your que is always available

2 Amazon S3

Highly flexible and secure data storage tools

3 Amazon RDS

Set up operate and scale relational databases – enhance the reliability of your db services

Web hosting with AWS-

Fast easy low cost – easy deploy and maintain-

AWS can host - Company website – scotia media apps and more –

You can handle usage peaks very easily with AWS with on demand provisioning

Based on traffic trends you can have 5 servers in the afternoon and only one in the morning and at night

You can also spin up testing environments only when you need them

Security in the AWS cloud –

Resilient security – high safeguards – rapid innovation – all part of standard security

IAM identify and access management and encryption are set up by default on all services

Cloud based Governance –

More oversight

Security control

Central Automation

You own your own compliance and security programs on AWS

AWS also provides monitoring and logging tools as well as built in firewalls for private networks

Also DDos mitigation tech for all your web needs

Deployment tools – Inventory tools – Encryption features – these are all also standard

Flexible key management options allow AWS to help with key storage or have you manage the keys entirely

In addition aws supply’s SDK’s to work with all security encryption services

Access management control

IAM- SSO and more tools are all included in AWS services as well as log aggregation

Alert notifications are also available to help you spot issues before they boil over

Anti malware and other tools are also available to add on

AWS Responsibility Model-

Who is responsible for security – (YOU and AWS)

1 User Data- YOU

2 Application- YOU

3 Guest OS- EC2 the dividing line of AWS and you once you have a EC2 keys only you have the data access

4 Hypervisor – Zen based and handled by AWS

5 Network – AWS elements like the VPC

6 Physical – AWS data severs

IAM model –

Identify and Access Management –

1 Users – in this case users are permeant named operators human or machine – stay until forced rotation

2 Group - a collection of Users

3 Role – NOT your permissions – but are your authentication procedures – Operator the credentials with a role are temporary – everything in AWS is an API – Role is simply the authentication

4 Policy document – lists the specific API that I am whitelisting for specific resources – and the conditions of access- you could also have an black list or explicit deny – if there is something you never want you can set it up like this so access in always denied – this also helps with authentication data loss

Every API action is recorded in cloud trail whether approved or denied

**Amazon Inspector**

A fully automated AWS threat detector

Access vulnerability’s – and deviations from best practices

AWS gives detailed report on suggestions once generated

You are responsible for the general security of your applications

Other features – identifies issues before and after deployment-

It is Agent based and API focused

It can be used as a integral part of the deployment process and proactively identify vulnerability’s

AWS continuously access the environment and makes AWS inspector better all the time

This steam lines the process of keeping up with AWS best practices

It also simplifies enforcement of security policies

Aws management console can access the Inspector services as well as CLI and SDK’s that can speed up the process

Findings from AWS inspector include security vulnerabilities as well as ways to fix these issues

**AWS Shield**

A managed disturbed denial of service or Ddos protection that will minimize application down time and latency

What are ddos attacks – a deliberate attempt to make your application unavailable to users

A slow Loras attack or spam flooding are all form of dos attack

A ddos attack is when you have a bunch of bot computers and the goal is to knock an application offline

Sometimes the attack is based off of how he application interacts with other sources

Customers use waf or web application firewall’s to protect against these attacks

An attack will require a complex response if you handle it on your own but shield can help

AWS Shield standard-

1 automatic protection with no additional charge

2 uses techniques like quick detection and inline mitigation

AWS Shield advanced –

2 gives you access to your own private ddos response team – additional charge

24/7 access to DRT team and apply mitigations on your behalf

You can also set up black lists that make is so certain sources can no longer route traffic through your systems

AWS Shield can auto scale up its services at any time to help you and will not change you for the additional resources used to achieve a protected system

Wrap up AWS shield is great for protecting your hosted AWS web applications

**AWS Security Compliance**

Both parties have a responsibility to maintain the security of their environments

Although customers don’t communicate everything to AWS AWS will culminate information about security and control parameters to the users

Aws compliance practices are all designed to make sure AWS is always complete with best security practices

AWS employs governance frameworks to focus on risk management and make sure your information and applications are secure

External threat assessments are performed at regular intervals for AWS by third party sources

There are also customer threat assessment scans to manage their individual threat assessment

Customer must handle their IT - Review – deign – identify – verify

**Pricing in AWS Cloud**

1 you only pay for what you consume

2 you pay as you go – pay less as you use more – and can pay even less if you compartmentalize and order full sets of infrastructure over time periods

3 with reserved instances you can save 70% - you can get a discount based on how much you pay up front

4 The more you use the less you pay and data going in is always free

5 custom pricing is also available as well as free ec2 for a year micro instance

6 with multiple accounts you can still benefit from tier discounts

7 you pay for

Compute – Storage – and Data Transfer Out

No change for data transfer in-between in the same region services or any data transfer in

Basic pricing for EC2 – basic web interface set up on amazons proven compute environment –

Clock hours of server time – this is what you are charged for – as well as resource allocation within that EC2 instance

Auto scaling comes at no extra change

S3 – simple storage instances- Cost is defined by storage class – standard storage – versus substandard storage - + size and requests to the data – charges are incurred whenever data is transferred out of the service –

EBS – elastic block store - Three storage type – general – provisioned – magnetic

Provisioning data blocks and b=making snapshots of the data – it is charged per gigabyte per month

Amazon RDS – regional database service

You are changed based on clocked hour of server time –

You are changed based on the capacity of the databases you choose

When you use on demand database services you pay for the time the database is used but only then with reserved databases you pay for a time where the data based is fully reserved to you with a discount for paying up front

You can also be changed based on deployment type of your data base

Amazon cloud front – distribute content anywhere fast

You are charged based on traffic distribution through the edge location that the content is served on

**AWS trusted Advisor**

Aws TA is a tool that gives best practices of resource management –

It tracks – Security – Cost Optimization –Performance – Fault Tolerance

It is designed to save customers money and help you from spending money on things you don’t need

It is all about eliminating waste

It can work with things like cloud watch and AWS lambda to automate regulation of your amazon services

**AWS Support**

You can get –

Protective Guidance

Best Practices

Account assistance

Any time as part of AWS continued support plan for your services –

You can also get a dedicated rep or TAM technical account manager to help manage and guide you with all your services should you need it

You can also select from support plan options for full managed support of your services

**END of AWS cloud essentials overview**