Well Architectured

Wednesday, September 23, 2020 3:13 PM

Intro

- Operational Excellence - does it work/ will it continue to work
  - All operations r code, great documentation (easy with code), small changes are better - iterate, plan for failure, then learn from that failure (and success)
- Security- does what u want and that's it
  - Least privileges, track who did what when, security is everywhere, automate sec tasks, encrypt at rest/transit
- Cost - spend only what you have to
  - Using consumption based pricing (not flat), always measure efficiency, pay AWS to do it (so you don't)
- Reliability- it is consistent and recover quickly
  - Recover automatically, scale horizontally (no vertical!), reduce idle resources, make it automated
- Performance- reduce bottlenecks and waste
  - Let AWS help (scaling), understand the infrastructure (edge), serverless, use new services, user first think

Cloud Best practice

- No guessing capacity needs, automate everything (cons and repeatable), testing at scale (on replica), adapt your architecture as needed, be data driven, practice

Operational Excellence

- Key questions in creating solutions - what is the business goal, what do we need to meet that goal, what compliance restrictions/ requirements, what service dependencies do we have
- Designing the solution - is everything code, is the design observable, are logs/ data useable
- Ready to deploy- are processes consistent, is code well managed, enough testing?, anticipate failure
- Is it healthy - use metrics for each service, show overall health, measure business metrics
- Maintaining - anticipate and plan/ unplanned events, respond with code, automate everything, connect to 3rd party
- How to evolve- what did we learn, test any assumptions, experiment often

Cost operations

- Spend what you have to - consumption based pricing (usually min pricing), measure effectiveness of things, pay AWS so you don't have to manage
- Use appropriate resources- autoscaling, right size (no overkill), use that data, optimize by region, optimize data transfer
- Match supply & demand - demand based, buffer based (SNS/SQS), time based (Kinesis) pricing
- Awareness - stakeholders, some governance model (alarms, budgets), make per team, tagging, tracking lifecycle /usage
- Optimize - utilization with requirements, report findings, evaluate new services for value, use managed services

Reliability

- Think of resource limits- lets not go over, how would we go over, can limits be lifted
- Networking (topology, bandwidth) - vpc networking limits (IP, subnets), resilience, traffic spikes, consistent low latency
- Availability (failure, etc) - can people access it quickly (anywhere), deploy updates easily, withstand outages (4 9's)

Efficiency

- AWS do it - they nasty and they always have cool new services to use
- Reduce latency with AWS edge and network- think of the user not the tech
- Serverless always - default efficient and managed
- Process of efficiency
  - Selection (what to use) - is this the right compute? What data store? Does it complement itself (use the edge)
  - Review (choices hold up) - stored as code (easier to switch), fully automated, can you benchmark, load testing
  - Monitoring then repeat (they still work) - passive (cloudtrail) vs active (events) monitoring, create metrics
    - Generation, Aggregation, real time processing, storage, analytics
- Making tradeoffs - caching, sharding/ partitioning, buffer- fake it until you make it

Security

- Identity & Access - who did what- IAM, least privilege, no root access, review accesses
- Detective Controls - monitoring point - capture (analyze) logs, audit controls and configs, look for odd activity/problems
- Infrastructure protection - AWS - trust boundaries (lanes of commun), NACL, SG, protect hosts, use AWS configs, service level protection (along with VPCs)
- Data protection - classify/ manage data - sensitivity, who can access when, encrypt in transit/rest, backup and test it
- Incident Response- contain and respond to attack - plan to tag resources, adjust permissions, redeploy, what u learn

Security

Wednesday, October 28, 2020 3:17 PM

Sooooo boring

RDS

Wednesday, September 16, 2020 2:04 PM

Overview
- Only have to create database, load data and tune queries
- Aurora isnt free
- Options to configure- DB engine, license model, engine version, instance class, multi az, storage type, backups, monitoring, when is ur maint window
- Engine Types
  - MySQL
  - Maria
  - Postgre
  - Oracle - Enterprise, standard, standard 1, standard 2
  - SQL Server - Express, Web, Standard, Enterprise
  - Aurora - MySQL & PostgreSQL options
- Use Cases
  - Prod - def mutli AZ
  - Dev/ Test - def single AZ, 20 G allocated storage
- License - bring in license for ur engine (extra cost)
- Instance Class - sim to ec2 instance type, determines CPU and mem capacity
  - T2/t3 burstable instances (1-8 vPCU, 1-32 G RAM)
  - M3/M4 gen purpose (2-64, 8-256)- for harder like wordpress
  - R3/r4 mem optimized - HPnetworking
- Multi AZ - redundancy and availability- across Azs, in region, primary and standby nodes, cannot read/write directly to standby, more money
- Storage types
  - All but aurora uses EBS - GP (burst to 3000 IOPS), IO (40K IOPS)
  - Aurora special -
- Pricing - engine, version, license, class, multi az, storage type and allocation
- DB Identifier- unique for customer in region
- Credentials- set during creation, admin rights, don't use (use roles)
- Network - VPC (environment), Subnet (IP ranges), Public access (gives public IP), AZ, VPC SG
- DB name, DB port - how apps access, DB Parameter Group, Option Group (optional functions), Encrypt, Enh Mon
- Backups - 0 to 35 days, has backup windows, keeps tags
- Maintenance - has versions, auto upgrade minor updates (X.Y.Z format), does create downtime for bigger updates

Costs
- Instance Hours - based on region, type, DB Engine, License used or not
- DB Storgae- EBS volumes (if not aurora). Storage type, storage GBs allocated
- Backup Storage - no charge for backups up to 100% of total DB storage
- Data Transfer - outgoing only, per region pricing, includes cross region copying
- Can use tags to keep in order
- Reserved instances to save money

Scaling
- Scale vertically - new host has minimal downtime
  - In multi AZ, secondary instances resize then first resizes (very small downtime)
  - Can choose maint window or immediately (e.g. change instance type or modify rds instance)
  - Single AZ - took 3 mins to vert scale, multi - 1.5 min (14 min total)
- Scale EBS storage- no downtime, reconfiguring though (reconf IOPS on the fly)
- Scale read replicas - no downtime, only some engines, helps read only
- Storage and CPU decoupled (can change storage types of instance)

Multi AZ
- Synchronous higher availability (not perf), for failovers, writes slower
- Improves backups and costs twice as much
- 5 scenarios - AZ outage, when DB fails, DB class changes, software patch, manual failover
- Ex. Change class - provisions bigger then catch up through replication, then switch roles (primary vs sec)

Read Replicas
- 5 uses - scaling (reading from 1 or more), source DB unav, Reporting/ data warehouse, DR (promote), lower latency

Backups
- To ebs volumes, can be done with multi AZ
  - Daily ebs snapshot of secondary host, put in S3 (not accessible- rds EBS so entire host snapshot)
  - Taken with transaction logs from primary (every 5 minutes)
  - Can use backup to repl cross region or cross acct
- Backup window
- Retention period (1-35 days, default 7)
- Manual snaps - cli, api, console, snaps kept until you delete them, used for checkpoints rather than point in time
- Incremental snaps (save what changes), saces money
- Impact - single - brief few min impact but multi az has no (unless SQL server)
- Restoring - creates new DB instance, retains PG and SG, changing storage type will slow down process
  - Auto backups - point in time
  - Takes a while since blocks pulled form S3, full perf takes time to warmup
    - Can fix with higher I/Os

Security
- Network Iso
  - Enabled with VPCs - private subnets, SGs (firewall, NACLs), no public access
  - Use ClassicLink to network with non VPC, DIrectConnect to repl on premise DB, VPC peering to share between
- IAM
  - Management (create, delete, who can see)
  - Users and roles, with MFA
  - Within DB (view, edit)
  - Integrated security with AD for SQL server and IAM with MySQL, Postgres, Aurora
- Encrypt at rest
  - Free, by KMS 256, no impact on performance, encryption replicated (cross region), keeps log, done at volume (not on app), can rotate keys but not unencrypt, 2 tier encryption (master created by customer and instances have data keys)
    - Benefits 2 tier - limit risk, easier to manage fewer master keys, easier encrypt large data, can rotate keys
- SSL for connectivity
  - Communication from RDS to client (such as on EC2)
  - Can enforce SSL via Parameter Groups
  - Encryption at rest not free tier

Monitoring
- 15 -18 metrics, access via rds or cloudwatch console/ cloudwatch APIs
  - CPU (util), Storage Space, Network Throughput, DB Connections, IOPS
- Enhanced Monitoring - host vs hyper visor
  - OS/ host level monitoring
  - 1 sec intervals, default 60
  - 50 more metrics
- Perf insights
  - SQL, free, impact on server, not on db.t2, identifies bottlenecks
- AWS RDS events (notify when RDS management event occurs) - subscribe to when things change
- AWS Config - changes to config
- CloudTrail
- Trusted Dvisor

Aurora
- Not coupled, monolithic (faster, harder to fail, scales easier) - self healing, distributed, elastic
- Aurora better by fixing logging and storage layers, compatible with most
- Cluster - one or more instances and cluster volume, spans AZ
  - Primary instance or aurora replica (same storage volume, only read) - up to 15 replicas, auto failover
- DB Connections - connect through endpoints
  - Cluster endpoint - write ops
  - Reader endpoint - replica connect, for reading, LB support
  - Custom endpoint - LB and can do what u want
  - Instance endpoint - specific to instance
- Global DBs - primary region (read and write), secondary region (within 1 sec, read only), can be promoted
- Uses log records, faster and smaller, storage monitored consistently, at least 4 nodes (of 6) will be up to date
  - Also DB monitoring to make sure DBs don't go down
- Storage up to 64 TB, 11 9s durability, no backup penalty
- Serverless - can only connect from your VPC through endpoints (via NLB)
  - Decouples instances from data storage - no classes in serverless- runs on Aurora Compute Unit- has max mem and CPU
  - Auto scales to 0 potentially, based on CPU and connections, can scale manually, first is a lil slower

S3 1

Thursday, September 17, 2020 3:48 PM

Intro
- Web based, secure, durable, infinite, secure
- Used for backup, archive, secure, fast storage
- Buckets - soft 100 limit, region, have ARN, subresource depends on parent, universal namespace
  - Virtual style URL - bucket name.s3.region.amazonaws.com
  - Path style url - s3- region.amazonaws.com/bucket - will be dep
- Flat files - no directories - can imitate with prefixes (more means faster)
  - Key names cant be longer than 1024 bytes (UTF-8)
- Up to 10 tags, 384 char length for key value
- No locking for S3 - if 2 requests, latest wins
- Cost - amount storage, requests, data transfer (out), transfer accel, management- monitoring, storage class analysis, S3 inventory, and tagging

Classes
- Standard IA - Min billable object size 128 KB and 30 day retention (charged for retrieval/revival per GB)
- OneZone - sim to IA but less availability so lower storage fee
- Glacier - min billable storage time 90 days, durable, cheap
- Deep_archive - min billable 180 days, super cheap, hour retrieval (per GB)
- Intell Tiering - between S3 and IA - no retrieval fee for access IA, 30 day min to move to IA, must be larger than 128 KB to be transitioned (individual level)
- Lifecycle Policies - on buckets
  - Transition rule - when obj moves to another class
  - Expiration rules - when objects expire/deleted

Buckets
- Create folders for faster
- Precreated metadata (content-type) and custom meta (x-amz-meta)
  - Fixed list, x-amz-meta is the only one which can be customized
- Tags good for description, metadata can be accessed from http

Timed URL
- Temp permission on object via signed-URL- bucket owner gen url with security, name, http method, exp time
  - Use script (python)

Monitoring
- Daily Storage Metrics - on by default, one report each day (24 hr intervals), free
  - Has num objects per bucket, size in bytes of bucket, saved for 15 days
- Request and Data Transfer Metrics - optional, 1 min intervals, standard billing rates
  - Get/head/put/delete/list/post requests reported, 400/500 errors, how many bytes uploaded/downloaded, latency for first byte and total, enabled at bucket level for all objects in it, for 15 months
- Filters - by bucket (default), by storage type, by filter ID (prefix, to specific object)
- Cloudwatch is by best effort- completeness and timeliness not guaranteed, but pretty good
- Access logging - track requests for bucket access- include time, requester IP, key name, operation, response codes
  - Disabled by default, no charge, logs storage costs money, best effort basis, periodic delivery to S3
  - Turn on by adding logging config - need target bucket, optional log prefix, permissions
  - Needs write permission to S3 log delivery group- both buckets must be owned by same acct
  - Logs build over time (use lifecycle rules), not auto analysed (athena, elasticasearch), bucket level
  - Auto adds log delivery ACL when created
- CloudTrail Logging
  - Records API calls- track changes, identity, time, parameters, returns
  - Enabled with CloudTrail, captures specific API calls (bucket level, can do object level), in JSON, within 15 min
  - Cloudwatch too- cloudwatch logs (monitor as metrics change), cloudwatch events (i.e. lambda when occur)
    - Query/ filtering capability, filtering
  - Diff from access logging - API calls (bucket) vs access (object)
  - Default S3 encrypt (can do KMS)

Permissions
- Default all private (owner only)- write policies
  - Resource (obj/bucket) (in S3)
    - Access Control Lists- read/write for AWS Accounts **not users**, use XML, list of grants to grantee
      - Created by default and creator has admin rights
      - Basic read write, cant do conditional access or grant denies to other accts
      - Consists of owner CanonicalID, grantee ID, permission - up to 100 grants
      - Permissions: read, write (cant write within object, only write /delete/modify to a bucket), write acp (write ACL), read acp, full control (grantee has all of the above, object cant write still)
      - Can be given to predefined groups (of accts), need to change permission grant ID to group URI (uniform resource id, http)
        - Authenticated users is a group (rep all aws accts), all users is a group (public public), log delivery (write bucket logs)
      - Canned ACLS- predefined ACL to set common permissions, usual **with cli**
      - Bucket perm and object perm indep
      - Used for: access to obj not owned by obj owner, manage access to indv objects, grant permission to S3 log delivery group
    - Bucket Policies - access to IAM users, KSON, fine grained permissions
      - Used for: cross account more than basic read and write
  - User policies - to users, groups, roles managed in IAM, JSON, fine grained perm - cant be given to root or for anonymous
  - Bucket and user policies - C PEARS and C EARS
    - Limit 20 KB, neither by default, both grant to IAM users, deny, condition
    - Policy - Principal (acct or user granted, none for user), effect (deny/allow), action (list permissions), resource (ARN), SID (tag)
      - Principal (only Bucket) - needs ARN or CanonID, "AWS":"arn:aws:iam**:ID:**root" for root, could be user/uname
      - Resource - arn:aws:s3:::bucketname can have /* for all in bucket, can be :::* for everything
    - Conditions - like an if statement, added after resource, ex. StringEquals: { s3:x-amz-acl : [public-read"] } would be ACL = pub read
- Cross acct - acct A gives permissions to acct B, can be given to user in B, common for billing
- If user and bucket same acct, all policies evaluated at same time, deny > allow
  - For Cross, owner must grant access and user must grant access via user policy
  - For both object operations and bucket operations, except owner not needed for cross object operation

Encryption
- Transit - man in the middle attacks
  - Use --sse on CLI to upload to S3 with sse amazon managed keys (need to enable encryption on bucket)
  - Use aws/s3 KMS to use generated AWS managed key, in KMS create key, who administrates, users
  - Cant do customer provided keys on console, **--sse-c - key <32 digit key> --sse-c AES256** is code to upload w/ur key(need for download)
- Rest - hard drives
- Ciphertext - encrypted (as opposed to plain), created via algorithm- usually public, means secret key more important)
- Asymmetric uses 2 keys for encrypt and decrypt
- Use master key to encrypt specific keys (one key in plain text for all others)
- Versioning - MFA delete only enabled by root acct, treated as subresource, delete markers also have versions (stack)
  - Way to restore - use a get and put to put object back at current
  - Versioning only suspended no versions - on susp, ones made with versioning still persist, but new ones will not persist if deleted/over

Replication
- Asynchronous, instant to hours, over SSL
  - Same region- for log aggregation, data sovereignty, between aws accts
  - Cross region - for compliance, latency disaster recovery
- Enabled with repl configuration - specify objects using tags (or all), are exact copies
  - Can change storage class and ownership (owner override), to another acct
- Both source and destination buckets must be version enabled, destination region must be enabled, S3 needs permissions to replicate to dest, aws acct must be able to read objects, object lock must be on in both if in one, cant replicate to mult buckets at once
- Are replicated - objs made after repl enabled, unencrypted objects, encrypted at rest with S3 keys, metdata, tags, lock info, object ACLs
- Not replicated - past objs, objects missing permissions, c-key encrypted objs, KMS keys unless specified, lifecycle actions, replicas, glacier
- Repl time control - repl in seconds/ minutes (99.99% in 15 mins), costs $
- Owner Override - replaces ACL in transit so dest owner owns, done in repl config, must update permissions
- Can enable KMS repl in repl config, need to update S3 permissions
- Deletion Protection - delete markers not repl (added to source), version deletion not repl

S3 2

Monday, September 21, 2020 10:38 AM

Lifecycle Management
- Done in lifecycle configurations using rules (up to 1000), applied to bucket or subset, in XML
- Possible changes- Standard → Standard IA → int → onezone → glacier →deep archive to none
  - Object in standard must be at least 30 days old before move to IA, IA to glacier at last 30 days old
  - Obj smaller than 128kb cant be moved to IA or Int
  - Cannot create lifecycle on MFA
- Glacier - Transitioned objects cant be accessed via Amazon S3 Glacier Service, not available in real time, must be restored, restorations are copies and only available for limited time
  - 40kb added to each for name, index, metadata (increases storage charge)
  - Min storage fees - at least 90/180 days fees
- Version enabled buckets - can do noncurrent version transitions and expirations, can create rules for current versions, non current versions
  - Version suspended - same but can delete current or as usual make noncurrent (with delete marker)

Storage class analysis
- Monitors data access over time and recommends policies
- Up to 1000, can filter by tags, prefixes, buckets, daily, has visuals, can be exported, costs money ($.1 for 1M obj)
- Takes 2 days to populate, 30 days to get real analysis

Events
- Stored as subresource, enabled by notif config based on events and destinations, by default off
- Send to SNS, SQS, Lambda
- Config needs event type, destination, permissions to use dest, filter rules (depending on ur needs)

Performance Opt
- 3500 put/copy/post/delete per second per prefix, 5500 get/head requests per second per prefix
- Benchmarking - think about throughput - need different/ more EC2?
- Min latency - proper region, using transfer acceleration, caching freq access content (CDN), in mem cache (elasticache), for media cache (elemental mediastore), use retries (retry after 503 errors)
- Horizontal scaling for parallelizing - multi part uploading, range based gets, parallelize list calls, query other things like database not S3 (secondary indexes), s3 inventory (put into downloadable csv file)
- Using latest SDK (faster, retries, transfer managers in code, mpu native)
- KMS key limits

CloudFront
- Origin (S3, EC2, ELB,Route 53 endpoint or external), edge locations, distribution (list of edges)
- TTL default 24 hours, can be invalidated but will be charged, supports both static and dynamic (php) content
- 2 types- web distribution and RTMP (media streaming)

Transfer Acceleration
- Region choice- proximity to resources and users
- TA is chargeable, creates diff S3 endpoint URL (old one still exists), only config by bucket owner

CORS
- One domain engage with other domain (between buckets), blocked by default, use CORS configuration to enable
  - Config needs XML rules, needs allowed origin, allowed method, allowedheader (if applicable)
  - Same origin needs no extra things
- Causes fail, not error.html

S3 Action Optimization
- Put- multi part to optimize- can stop/start as wanted, need to optimize part size (too small increases overhead)
  - For choosing part size: 25- 50 MB on higher bandwidth networks, 10 MB for mobile
  - Use CLI, need permissions, default for >8 MB
- Gets- use cloudfront, use endpoints, global, low latency, cache objects, reduce S3 gets (get from elsewhere)
  - Range-based gets - getting in parts, use range http header for specific bytes (can parallelize)
- Lists - parallelize when need sequential list of keys such as multiple commands based on suffixs
  - 2nd index - when sort by metadata, timestamp (can use dbs, Elasticsearch, CloudSearch, ec2)
- Storage inventory - scheduled inventory of objects and metdata, daily/weekly, out to csv, can decide which metadata to include, whether to list prefix objects, mult lists per bucket, can configure notifs, eventual consistent
  - Inventory takes 2 days to create, has manifest files (lots of metdata in JSON), checksum file means done, actual data is in data folder as zips, hold csvs with lots of metdata

Static Web Hosting
- HTML, CSS, Javascript webs, super low cost, high avail, auto scale, unlimited storage, API gateway /lambda for a fully dynami c serverless website
- S3 endpoint changes with website hosting enabled (s3-website in URL,)
  - Public reads only, HTML, can redirect from bucket/ object, supports get and head ops, returns index.html, no SSL
  - API- public and private reads, XML, no redirect, does all bucket operations, has SSL
- Bucket level redirects - can redirect to diff bucket, good for when one bucket better
  - Done with redirect rules, done in XML,
  - Can also redirect requests to another bucket or domain on http or https
- Access Control for objects not owned by user, otherwise bucket policy is best

CloudBerry Backup
- Securely backup machines, cross platform, can be scheduled, encrypted on client or server (default by SSL), can compress data , can configure retention, can use Glacier, can use Transfer Acceleration
- Need Cloudberry software
- Backing Up
  - Need a dest bucket, a user with role that can do things to S3 bucket
  - In cloudberry, choose backup to S3, name, both keys, bucket name, choose what to backup
    - Can use compression, encryption (128/256, password or not), use SSE, transfer accel or IA, retention (default 3 versions), schedule (manual, daily, etc), notification
- Restoring
  - File - create restore plan- save restore plan? - restore latest/point in time, default wont restore Glacier (cus costs) - where to restore
- Great for auditing (has backup/restore history and file history)

Securing S3
- ACL - only 4 permissions, default owner, bucket/objects
- Bucket - much more fine grained, bucket
- User policies - bucket policies but IAM user based
- Logging - log delivery in permissions of target, then properties of source and enable
- Versioning - duh
- CloudTrail - has data events which can make trail logs for buckets, has metric,
- CloudWatch - can filter by metric, make alarm based on metrics
- Replication - cross acct
- Can configure SNS topic with management - events and send it to SQS, SNS, Lambda

SageMaker

Tuesday, June 22, 2021 12:52 PM

- ML vs Deep Learning
  - ML - takes data, uses algorithm to learn (create rules by observing input and output)
  - Deep Learning - takes data (ML), self-trains to better improve algorithm
- Types of ML
  - Supervised - learn by looking at examples - classification, can tell if correct
  - Unsupervised - find patterns in data, grouping/clustering - might be correct
  - Reinforcement - correct actions rewarded (doggo)
- AWS Integration
  - S3, Athena, EMR, Glue, Quicksite, Lake Formation, Redshift
- ML prep - data exploration (ex. flip images), formatting, conversion, encoding, cleaning, normalization, resampling
- Pricing- AMIs free, econ of scale with AWS
- AMIs - TensorFlow, Apache AXNet
- Sagemaker
  - For data visualization, analysis, feature manipulation, training and evaluation
  - Ground Truth - help you label, accurate results, private public human workforce
    - Raw data, human annotations, begins active learning model, annotates some itself then sends back to humans to feed back in (eventually creating training data)
  - Notebooks
    - Based on Jupyter Notebooks- units of instances, etc that can be saved
  - Built in Algorithms
    - Linear, logistic regression, clustering, decoding etc
    - KNN, linear learner, K means (groups), Random Forest (classif) are for gen purpose
      - KNN - regression classification, needs supervision
      - K means - good for unstructured grouping
    - Specific cases- classification, test encode, trend recog
      - XGBoost - estimate costs with custom feature weighting (good for estimating rebuild costs)
  - Deployment
    - Hosted - one at a time, using endpoint
    - Batch Transform - pred on whole dataset, one after another
    - Autopilot - model creation
    - Model monitor - monitors model in prod, tracks drift in pred
  - Production variants
    - In Sagemaker can create a few model variations and run them based on container images (such as ECR docker), save in S3 as model artifact, then test with different weights
      - Sage can handle autoscaling for APIs,
    - Can change endpoint with no downtime
  - At edge
    - IoT greengrass - extends IoT to customer devices, enables local ML (good for no cell service)
  - Neo
    - Compiles model and can run anywhere (MXNet, Tensorflow, PyTorch, etc)
    - Inf1 instances - increase throughput, lower cost per inference
    - A2i -
- AI in AWS
  - Managed services - recog, textract (text from image), polly (text to speech), transcribe, translate, comprehend (key phrases and sentiment analysis), lex (chatbot), forecast (inventory forecast), personalize

Intro

Thursday, August 13, 2020 3:28 PM

Advantages of cloud computing

1. Trade capital expense (pay for comp) for variable expense (pay as u go) - No data centers/ servers, just pay for what you use
2. Benefit from economies of scale - amazon is too awesome
3. No guessing about capacity (will I have enough when I buy?) - can set metrics to be more accurate with capacity needs
4. Increase speed and agility - scales infinitely with demand
5. No spending money on data centers!
6. Go global in minutes - deploy lower latency or better

3 types

1. Iaas - infra as service - consumer manages the server (phys or virtual) as well as OS
    a. Example - EC2
2. Platform as service
    a. Godaddy - don't have to worry about hardware, just put in website code and they handle the hosting (hardware, OS)
3. Software as a service
    a. Gmail - you manage inbox, they handle data centers, servers, network, storage, patching

3 types of deployment

1. Public - AWS, Azure, GCP
2. Hybrid - website public, conf email info on your servers
3. Private (on premise) - you manage it in your datacenter

High level services- AWS Global infr - a lot of crap they do as high level services, and each one has a few services

For CP :

1. Compute services - core - EC2 , lambda
2. Storage services - core - S3
3. Databases - core RDS, DynamoDB (non-relational database)
4. Migration & Transfer
5. Network & Content Delivery - VPC, Route53
6. Security, Identity, Compliance - core
7. AWS Cost management - core

Global Infra

- 24 regions, 72 Availability zones (2019)
- Availability zone (data center(s)) - filled with servers
- Region - geographic area of 2+ availability zones
    - Why choose a region?
        - Data Sovereignty Laws - By law need to have data maybe in a certain country (EU/ federal requirements)
        - Latency - majority end users want it closer (faster)
        - AWS Services - US East 1 (primary region) has first services usually
- NA has 9 - 2 gov, 1 canada , SA has 1, Europe/MiddleEast/Africa has 4, Asia Pacific has 8
- Edge Location - endpoints for AWS - used for caching content (quicker access on 2+ attempt) lots of edges
    - Typically CloudFront is used by consumers to access content in edge location

Login To AWS

Monday, August 17, 2020 11:30 AM

AWS support plans

1. Basic - Customer Service for account, billing, access to AWS Community Forum
2. Dev - Primary Contact - 29/month, 12- 24 response rate, scales based on usage
3. Business - 100/month, 24/7 support, AWS trusted adviser, AWS support API, 1 hour response rate, scales based on usage
4. Enterprise- 15000 a month, technical account manager (TAM), all features of business, 15 min response rate

Go to CloudWatch for Billing alarm - uses SNS topic, can email you

IAM- identity access management - used to manage users within an account

- IAM is not region specific, is global
- Policies are written in JSON - JavaScript Object Notation
- Lots of different GROUP accesses, admin access gives policy to use or do anything
    - Within groups, have users which allow people to login as that user within the group

Access AWS Platform

1. Via Console (in browser)
2. Programmatically (Using command line)
3. Using Software Developer Kit

Root account - email used to set up acct, always has full admin access. For other users, create indv user accounts (with specific accesses)

- Group is place to store users, users inherit group permissions - ex. Devs, System admins, Finances
- To set permissions, need to apply policy (JSON) and referred to as key value pairs

S3

Monday, August 17, 2020 11:30 AM

S3 - Simple Storage Service
- Provides devs and IT with secure, durable, highly scalable object storage
- It's a place to put ur files (flat files, don't change)
- Object based storage (block space is where install OS or database)
- Stored on multiple files in mult facilities

Basics:
- Files 0 B to 5 TB
- Unlimited storage
- Files stored in buckets (folder in cloud)
  - Http200 code if the upload is complete (how to check if upload was successful)
- S3 is a universal namespace (bucket names are unique globally)
  - Ex: se-eu-west-1.amazonaws.com/bucketname
  - Creates DNS or web URL so must be unqiue

S3 Objects are just files
- Have keys (name)
- Value (data and is made of sequence of bytes)
- Version ID
- Metadata (data about data)
- Subresources (access control lists, Torrent)

How does data consistency work
- Read after Write for PUTS of new objects
  - If you read immed after writing a file, viewing is possible
- Eventual consistency for overwrite PUTS and DELETES (takes a lil time)
  - If update existing or delete file then immed read, you may get older version (updating/ deleting takes time to propagate amon g s3)

S3 guarantee
- Availability for S3 platform is 99.99%
- Amazon availability for S3 is 99.9% (file available)
- Amazon file durability for s3 is 99.99999999999% (file being lost)

S3 features
- Tiered Storage -
- Lifecycle Management - can manage which tier it goes
- Versioning - can restore previous
- Encryption - encrypting files
- Security - using Access Control Lists (Indv file basis, can choose who can access) and Bucket Policies (can say which people can access bucket)

S3 storage classes - retrieval fees for all but highest 2
- S3 standard - has the percents above, can handle loss of 2 facilities concurrently
- S3 - IA (infreq access) - for less frequent but rapid access
- S3 One Zone IA - for lower cost infreq access, don't req zone data resilience
- S3 - Intelligent Tiering - optimize costs by auto moving data to most cost efficient access tier without performance impact
- S3 Glacier - secure, durable, low cost storage, slower retrieval time
- S3 Glacier Deep Archive - lowest cost storage, retrieval is 12 hours

S3 charging
- Storage, requests, storage management, data transfer, transfer Acceleration, cross region replication
- Transfer Accel - fast transfer over long distances (edge locations) via optimizes network path
  - Upload to near bucket which transfers to original bucket, rather than them all the way to
- Cross Region
  - Buckets copied from region to region (primary to secondary)

Review
- S is object based
- 0 B to 5 TB
- Unlimited storage
- Files stored in buckets
- S3 is universal namespace so to use buckets they have to be unique
- Not suitable for OS or databes
- Object uploads have http 200 code
- Key of s3 - key value pairs (key or name and value or data)
- S3 can read new file immed, read update or delete can take a lil time to propagate
- 6 different storage classes (original, infreq, one zone, intelligent, lower cost, lowest cost)

Lab
- Notice s3 is global
- Can create buckets (default private)
- Can add things to bucket with upload (default private but can make public via file permissions and public access or actions-> make public)
- Http 200 code is sent back to browser
- Review
  - Bucket Names share common space (need unique bucket names)
  - View buckets globally but have them in specific regions
  - Can use cross region replication for more dependability
  - Can change storage on demand
  - Can use transfer accel for faster endpoint download
  - Restrict bucket access -
    - bucket policies (restriction access to whole bucket)
    - Object policy - indv file policy
    - IAM policies - to control bucket access via user accounts in group

Create website
- Create bucket
- Add html files to bucket
- Go to static website hosting
  - Will need an html and an error file at least
- Click on link
  - To access, need files to be public - can be done with policy for bucket or make each file publ
- Can handle lots of users (scales automatically)
- Cant connect to database/ dynamic websites on S3
- Review
  - Can make entire bucket public with bucket policy
  - Can use S3 to host static websites (html), dynamic cant
  - S3 scales automatically (such as millions of users)

CloudFront

Monday, August 17, 2020 3:33 PM

Cloudfront - a Content Delivery Network (CDN) or system of distributed servers (network) that deliver webpages and other web content to a user

- Based on geographic location, web page origin, content delivery server
- CDN connects to users across the globe and makes distr faster
  - Edge Locations - where content is created, separate to region/ AZ
  - Origin - origin that CDN will distribute - can be s3 bucket, EC2 instance, Elastic LoadBalancer, Route53
  - Distribution - name given to CDN, consists of list of edge locations it connects
  - Faster after first access since CDN distrib data from origin to edge location, then allows it to be cached
  - How long? - TTL (in second, usually 48 hours)
- Cloudfront can be used for entire website content
  - Request for website content auto routed to nearest edge location so content is delivered with best poss perf
  - 2 distributions
    - Web distr
    - RTMP - used for media streaming

In AWS

- create based on buckets (if want entire bucket then no path, if in bucket use origin path)
- Can set protocols for https
- Can set TTL
- Creates a domain name for your website origin
- To get files, can type into browser <domain name>/<name_of_file_in_bucket>
  - This content is now cached at edge location (should be faster access)

Review

- Edge location - as before, locations where content will be cached
- Origin - where files to be distributed live (can be s3, ec2, etc)
- Distribution- name of CDN and is the list of edge locations that it can send origin files to
- Web distr - used for websites, RTMP is media streaming, phased out
- Edge locations can be written to (can put object on them)
- Objects cached for certain TTL (time to live)
- Need to be disabled then deleted

EC2

Monday, August 17, 2020 4:50 PM

Elastic Compute Cloud - EC2
EC2 - a virtual server(s) in cloud

- Reduced time req to obtain and boot server instances, allows for quick scaling as requirements change
- Used to be TERRIBLE
- Nearly instant

EC2 pricing models

- On demand - pay a fixed rate by hour/second
  - Useful for users who want low cost, flexible without long term/ upfront payment
  - Great for apps with short term, spiky or unpredictable workloads
  - For first time apps
- Reserved - locked into contract, gives capacity reservation, offers discount
  - For steady state or predictable usage
  - Require reserved capacity
  - Upfront costs to reduce overall costs
  - Types - standard (75% off, can't change capacity), Convertible Reserved (54% off, can change attributes of instance for better value, change class of instance), Scheduled reserve instance (allow capacity growth for predictable schedule e.g. everyone starts work at 9 am)
- Spot - bid a price for instance capacity, when it hits your price you can use server, when drops below it loses server
  - Useful for apps with flexible starts and end times
  - For apps that are only feasible at low comp rates (since can set rate with bid)
  - Users with urgent comp needs for large amount capac
- Dedicated Hosts - physical EC2, reduce costs by allowing server bound licenses
  - Useful for reg requirements (e.g gov laws make your servers have to be dedicated)
  - Great for licensing when doesn't support multi tenancy (microsoft, oracle)
  - Can be purchased on demand
  - Can be purchased on reservation (70% off)

EC2 types - have families

- F1 - field programmable Gate array
- I3 - High speed storage
- G3- Graphics intense
- T3- low cost, gen purpose

EBS

- Virtual hard disks used by EC2
- Attaches to EC2, used like a hard disk
- Auto replicated in Availability zone for higher durability
- Types
  - SSD -
    - Gen purpose SSD (GP2) - balance price and perf for variety workloads
    - Provisioned IOPS SSD (IO1) - high perf SSD for mission critical, low latency, high throughput
  - Magnetic
    - ST1- Throughput Optimized HDD - low cost volume for freq access, throughput intense workloads
    - SC1 - Cold HDD - lowest cost HDD, for least freq access
    - Magnetic- prev gen

Review

- EC2 - is virtual server in cloud
- 4 prices
  - On demand - pay a fixed rate per second
  - Reserved - can offer a discount with higher upfront, long contract

    - Spot - bid for how much you want to pay, great for flexible start and end times
        - Amazon charges you for partial hour usage if you terminate, does not if they terminate it (cus the price went above your bid)
    - Dedicated - physical - for reg requirements
- FIGHT DR MC PXZ
- 4 types of EBS
    - SSD GP2 - gen purpose, balance price and performance
    - SSD IO1 - high power
    - Magnetic ST1 - low cost
    - Magnetic SC1 - lowest cost

EC2 Labs

Tuesday, August 18, 2020 12:56 PM

Create EC2

- EC2 - choose template, instance type (fightMcPXZ), instance details, storage, tag, security groups,
- How computers communicate -
  - Linux - SSH (Port 22)
  - Microsoft - Remote Desktop Protocol - RDP (Port 3389)
  - Http (Port 80) - less secure connection
  - Https (Port 443) - more secure
  - Use firewall to restrict specific port communications (per port)
    - 0.0.0.0/0 lets everything in - great for say web server
    - x.y.z.w/32 lets one IP in (32 means 1)
    - Adv and disadv of specific source - more secure but only from this laptop
- Asks you to create key pair- for logging into EC2, has one public and one private
  - Public keys - the padlock on your bike, maybe lots of copies
  - Private key - one key for all your padlocks
    - Save private key
- To view EC2
  - Copy IPv4 public IP - changes every time instance is loaded
  - Go to puttygen, put in your private key from created EC2. Save as ppk
  - Go to Putty, SSH, Auth - load new ppk
    - Back to Session, put in IP address of EC2 instance in saved session and host name
    - Login as ec2-esuer
  - If having access problems, make sure in security groups, in inbound, ssh source is 0.0.0.0/0
- Review
  - EC2 is a compute based server- it's a server
  - Need private ket to connect to EC2
  - Linux uses SSH (22), Microsoft uses RDP(3389), http (80), https (443)
  - Security Groups are virtual firewalls in the cloud - need ports to use
  - Everything from port in (0.0.0.0/0 lets in everything, x.y.z.w/32 lets in one source)
  - Always have multiple EC2 instances and have them in different availability zones

AWS Command Line

- Have Ec2 running, make sure user has programmatic access (must have access key under security groups)
  - Only worked when I was logged into user not root account…?
- Login to putty as usual (make sure have private key in ssh auth, then save public IP) then sudo su to grant root priv
- To use aws from command line, need to either give credentials or use roles
  - Credentials
    - Type aws configure
    - Give user access key then secret access key (both in downloaded csv)
    - us-east-1 as region name
    - No output format (press enter)
    - Can overwrite with same aws configure
  - Roles
    - In IAM, click roles, create role (e.g. s3fullaccess)
    - Go to EC2, actions - add IAM role
    - Go back to SSH, make sure no .aws credentials file (remove it or exit and come back)
    - Try messing with s3 now, works even though no credentials
      - Much safer- still allows admin access to ec2 instance, but not to aws
- Make a bucket - aws s3 mb s3://<nameofbucket> - Format is aws <service>, mb is make bucket, address gives name
- Aws s3 ls - lists buckets
- Can create a file (echo "text" > text.txt will write text to text.txt)
- Then put the file to bucket with aws s3 cp <file name> <s3 bucket address>
  - e.g. aws s3 cp hello.txt s3://kobesbucket
- Cd ~ takes to home directory, where there is .aws folder that has credentials file (risky)

Review

- To interact with aws, use 1) console (browser), 2) CLI , 3) Software Dev Kit
- Roles much more secure, easier to manage (can pick and choose policies)
- Can apply roles any time, no wait
- Roles are universal - no region, similar to users

Web Server

- Need httpd (hibachi?) with **yum install httpd -y** and then start with **service httpd start**
- Web server is up (I think) - to get into root use **cd /var/www/html**
  - Now in root - create webpage with nano index html

Elastic Load Balancer - way to direct traffic to app across servers (EC2s)

- Create Load Balancer in Ec2 service
  - App load balancer - flexible for most web apps, can see into code and make decisions
  - Network load balancer - ultra high performance, static ip addresses
  - Configure LB - give name and usually put it in every AZ- durable if one fails
  - Add ec2 instance to registered targets
  - Go to ec2 and make new instance - want it to be in different region (us east 1b vs 1c)
  - For bootstrap
    - In configure instance, at bottom, goal is we don't have to login to httpd and start
    - **#!/bin/bash** - gets us into interpreter (takes us to root level)
    - **yum update -y**
    - **yum install httpd -y**
    - **service httpd start** - all the above installs updates, htp, then starts httpd
    - **chkconfig on** - if ec2 restarts, restart httpd
    - **cd /var/www/html** - takes us to web server main
    - **echo "<html><body><h1> YO YO YO </h1></body></html>" > index.html**
  - Storage, tags as normal with creating EC2, security group (same as we made before)
  - With Load balancer, can access from web browser the DNS name of LB, not IP address
    - Goes to EC2 instance assoc with ELB
    - Add by going to target groups, selecting ELB, register
      - We can see both index.html pages from each EC2 by refreshing DNS page

Review

- App LB - Layer 7 (flexible, can make decisions based on app)
- Network LB - Extreme performance, static Ip Addresses
- Classic LB - low cost for test and dev
- Keep Ec2 instances in mulp AZ so if a AZ goes down its not terrible

Databases

Tuesday, August 18, 2020 1:53 PM

Database - a spreadsheet

- Relational - each row is a relation with lots of values
  - Each table is a file, has columns, rows
- In AWS - RDS
  - 6 types- SQL Server, Oracle, MySQL, PostgreSQL, Aurora (amazons), Maria DB
  - 2 key features- mult AZ (for disaster recovery). Read replicas (copies of RDS) to help performance by having copies
    - Multi AZ allows for auto switch if one AZ fails
    - Read replicas copy (5 copies) database info to another AZ, so if we only read from (5) copy and write to original, can greatly decrease latency
- Non Relational - a lot more flexibility
  - Collection (table), document (row), field (info)
  - Have key value pairs, which can have key values inside key values (value is list)
  - Columns can vary, wont affect other rows in database
  - Amazons is called Dynamo DB
- Online Trans Processing - gives row of data from database
- Online Analytics Processing - can give large number of records (all the orders from Europe), can get sums, unit cost (i.e. run macros)
- Data Warehousing was invented for OLAP - takes a dumb amount of time
  - To do analytics processing away from database so people aren't using data you're analyzing
  - Usually diff architecture (and infrastructure) than normal data storage
- ElastiCache - web service for easy deploy, scale of in-memory cache in cloud
  - Improves performance of web apps with fast in memory caches not slower disk- based Dbs
  - Actually just caches most common queries and thus makes it a lot faster (uncommon in normal DB)
  - EC supports 2 engines
    - Memcached, Redis

Review

- RDS - Amazons Relational Database (for SQL or OLTP)
  - SQL, MySQL, PostgreSQL, Oracle, Aurora, MariaDB
- For non relational - DynamoDB (no SQL)
- RedShift OLAP - Amazons Data Warehousing (for really complex queries on lots of records for analysis)
- ElastiCache - Memcached, Redis - caching in cloud for faster access
  - For speeding up performance of existing dbs- for frequent, identical queries

Lab

- To create Rds, costs money, make sure multi AZ deploy, has names, needs additional config - database name so wordpress can connect to it
- Needs a port for MySQL traffic -
  - click security groups, edit inbound rules, type is mySQL (should be 3306 port), source is custom, use autofill to put in ecs security group
- Now need an EC2 - create with default but in user details have

```
#!/bin/bash
yum install httpd php php-mysql -y
amazon-linux-extras install -y php7.2
cd /var/www/html
wget https://wordpress.org/wordpress-5.4.1.tar.gz
tar -xzf wordpress-5.4.1.tar.gz
cp -r wordpress/* /var/www/html/
rm -rf wordpress
rm -rf wordpress-5.4.1.tar.gz
chmod -R 755 wp-content
chown -R apache:apache wp-content
service httpd start
chkconfig httpd on
```

- When EC2 running, go to ip address and put in **DB Name, username, password** from when created DB
  - For DB host, need endpoint of RDS so copy from there and past into wordpress config
  - Causes error - need to write config file from ssh
    - Ssh in with public IP of EC2 web server, then sudo su and cd to /var/www/html
    - Nano to write wp-config.php (what is causing error) - copy and paste from wordpress page
  - Now try again. Make username password to be able to login to website
  - Wordpress is writing files to instance
  - After logging in, allows user access to wordpress console
  - Wordpress is connected to EC2 IP address- wont work if we stop EC2 instance
    - Fix with DNS (we can get one from load balancer)
    - After adding instance as target to LB, get DNS and paste into worpress address and site address (save changes)
- Take a snapshot by just going to EC2, actions, create image

Autoscaling

Tuesday, August 18, 2020 3:27 PM

Autoscaling -

- Launch config -
  - Create with image (that already has WordPress config code)
  - Same key, defaults
- Autoscaling groups - use saved launch config
  - Have subnet as all AZs
  - Adv details- yes receive traffic from LB - target group is our target group
  - Config group size lets us choose how many instances will be created and the target policy which has a
    value (is % when need to create a new instance)
  - Now its fault tolerant!

DNS - Domain Name System

- Like a phonebook, computers use names to map to IP Addresses
- Route53- Amazons DNS service
- Register Domain - if buying, have s3 bucket with same name
  - Go to s3, create bucket with same name (needs.com or extension)
  - Make sure bucket can host static website (properties -> turn on static hosting -> index, error html files)
  - Make sure bucket is accessible publicly
    - Use bucket policy to make everything in bucket public (needs resource, others as JSON key pairs)
- Hosted Zones - can configure a DNS
  - Create different record sets
  - Alias makes it so its not www, is just the domain name
    - Has alias target- needs to be bucket name

Elastic Beanstalk

- Provision ec2, security groups, etc easily
- Very low skill way of deploying aws apps to cloud
- Elastic provisions environments, instances, security groups, etc

Cloud Formation

- Aws most powerful tools- turns infrastructure into code to be deployed in mult regions
- Used by creating template that describes AWS resources you want, cloud formation takes care of provisioning
- Consists of Stacks
  - With a few buttons creates Instances, RDSs, S3s, security groups depending on the template

Review

- Route53 is Amazons DNS Service
- Global (no region) with Route53
- Can direct traffic across the world
- Can register domain name with Route 53
- Quickly deploy and manage apps in AWS cloud without worrying about infrastructure
  - Beanstalk auto handles details of capacity, load balance, scaling, health monitoring
  - Beanstalk is more limited in what it can provision and isnt programmable, cloudformation much wider
- Beanstalk and CloudFormation free but they create things that are not free (EC2, RDS)

Architecting for the Cloud

Tuesday, August 18, 2020 4:41 PM

Traditional vs cloud computing
- It assets are now provisioned resources
- Global, Available, Scalable
- Higher level Services (Machine Learning, Ai)
- Built in Security (lots of products)
- Architecting for cost (can easily control costs)
- AWS operations might be better than normally doing it (rearchitecting)

Design Principles:
- Scalability
  - Scaling up - increasing ram or CPU on a specific machine
  - Scaling out- using more virtual machines on say a load balancer
    - Stateless apps - lambda- functions or runs algorithm base don what you say
    - Distr load to nodes- ELB, RDS (read replicas)
    - Stateless Components- easier to scale with stateless things (say cookie of login vs holding as server)
    - Stateful components- don't want to lose info (RDS) - ex. a purchase
    - Implement session affinity- keeping user in ec2 for a session (sticky session- stuck to ec2 during session) - ex. Cookie in browser
    - Distributed Processing - faster processing by dividing between EC2 fleet (100s) rather than 1
    - Implement Distr Processing
- Disposable Resources
  - EC2 instead of physical assets (can terminate quickly) no contracts
  - Instantiating compute resources
    - Bootstrapping - don't have to manually configure each time
    - Golden Images - image of old instance can help deploy future
    - Containers
    - Hybrid - combo containers and eC2
  - Infra as Code - virtually build out
    - CloudFormation - can provision easily a server
- Automation-
  - Serverless Management and Deployment- serverless makes it so u don't have to worry about infrastructure- worry about deployment
  - Infr Manage & Deploy - automated ways to help grow/ maintain app infrastructure (by auto creating more)
    - Elastic Beanstalk
    - EC2 auto Recovery
    - AWS Systems Manager
    - AutoScaling
  - Alarms & events
    - CloudWatch alarms- when bills high for example
    - CloudWatch events- when things uploaded for example
    - Lambda scheduled events
    - WAF security automations - firewall automatic protects
- Loose Coupling
  - Well defined interfaces - using APIs for example
  - Service Discovery - implement Service discovery - allowing AWS to auto find another resource (ex when looking for EC2 when an EC2 fails, with multi AZ we have a backup)
  - Asynchronous integration- increased resilience with less dependent process (parallel vs series circuits)
  - Distributed systems best practices
    - Graceful Failure - ex Error.html page tells info on error
- Services not Servers
  - Managed Services - lambda, s3, route53
  - Serverless Architectures - serverless means no managing servers (less money, less problems)
- Databases (anti patterns is when not to use it)
  - Relational Databases (Aurora)
    - Scalability
    - High Availability (Multi AZ)
    - Don't use when no need for joins or complex queries
  - Non relational or No SQL (Dynamo DB)
    - Scalable
    - High availability
    - Don't use when need for joins and complex transactions or have large binary files (audio, video, image)
  - Data Warehouses (Redshift)
    - Scalable, high availble, not meant for Online Transaction Processing (single row payments)
  - Search
    - Scalable, available
  - Graph
    - Scale, high availability - Amazon Neptune
- Managing increasing volumes of data
  - Data lake- arch approach to store lots of data in a central location while readily available
    - Great place is S3, then can use SQL queries
- Removing Single point of Failure
  - Introduce redundancy ()
  - Mechanism to detect failure
  - Durable Data Storage- if important probably want on multiple s3 in different zones
  - Auto multi-data center resilience
  - Fault iso and traditional horizontal scaling - how to isolate fault and scaling OUT (mult ec2)
  - Sharding - processing data by seperating and spreading
- Optimize for cost
  - Right sizing (EC2)
  - Elasticity (so it can change with more users)
  - Lots of purchasing options
    - Reserved capacity
    - Spots
- Caching
  - App caching - ElastiCache (saving common queries like featured items on Amazon)
  - Edge caching - easier to access on 2nd time as its stored in edge location (CloudFront)
- Security
  - AWS features for defense in depth
  - Shared security responsibility with AWS - your responsible for some, AWS does other
  - Reduced privledge access
  - Security as code (Golden template code) or other template to implement security for many
  - Real-time auditing

CloudWatch vs AWS Config, Systems Manager, Athena, MAcie

Tuesday, August 18, 2020 5:43 PM

Cloudwatch is monitoring service - monitors performance

- Monitors
  - Compute - EC2, autoscaling, ELBs, Route53 Health checks
  - Storage - EBS (hard disk) volumes , cloudfront, EC2 gateway
  - Physical - CPU, Network, Disk, Status Check of Host server
- Can monitor most of AWS
  - EC2 CloudWatch monitors every 5 minutes by default
    - can use detailed monitoring to have 1 minute intervals
- Can create alarms to trigger notifications

AWS Config

- Detailed view of aws resource configurations - how they relate to each other and how they were configured
- Monitors setting of AWS environment
- All about configuration - security groups, etc.

AWS Systems Manager

- Allows manage EC2 at scale - runs commands across all EC2s at once
  - Can work for on premise servers as well
- For fleets or Vms
- A piece of software is installed on each VM
- Run command is used to install, patch, uninstall software
- Integrates with CloudWatch to show you entire fleet (on or off premise info)

Athena

- Interactive query service - enables analysis data in s3 using standard SQL
- Serverless, nothing to provision- pay per query/ per TB scanned
- No set up complex extract/transform/load processes
- Works with s3
- Can be used for
  - S3 log files (ELB logs, s3 access logs),
  - Generate business reports on data stored in s3
  - Analyze AWS costs and usage
  - Run queries on click-stream data

Macie

- Security service uses ML and Natural Language Processing
- Analyzes your s3 to see if it has PII
- Can report and alert about possible PII
- Can also analyze CloudTrail
- Great for PCI-DSS and preventing ID theft

Pricing & Calculators

Wednesday, August 19, 2020 8:40 PM

Pricing 101

- Pay as you go, pay for what you use, pay less when use more, pay least when you reserve capacity
- Capex - Cap expenditure - pay up front- it's a fixed sunk cost
- Opex - Operational expenditure - pay for what u use (utility billing)
- 5 policies
    - Pay as u go
    - Pay less when reserve - longer contract more save
    - Pay less when use more
    - Pay less as AWS grows
    - Customizable Pricing
- Principles of pricing
    - Understand pricing fund- the 3 fund drivers of cost:
        - Compute
        - Storage
        - Data outbound (leaving environ)
    - Start early with cost optim
        - Put cost visibility and control mechs in before scaling
    - Maximize power of flexibility
        - Choose what u want, don't pay for what u don't want/ arent using at the time
    - Use right pricing model for job
        - Review - On Demand, Reserved, Spot, Dedicated
        - Free tier - allows free usage for things
            - Amazon VPC - Virtual data center in cloud
            - Elastic Beanstalk (resources cost money)
            - CloudFormation (same)
            - IAM
            - AutoScaling (EC2 cost moneys)
            - Opsworks (uses things that arent free)
            - Consolidated Billing
- EC2 Pricing
    - Clock hours of server time (based on type is hours or seconds)
    - Instance Type (instance family - FIGHT DR MCPXZ)
    - Price model (of the 4)
    - Num Instances
    - Load Balancing (network > classic)
    - Detailed Monitoring (every 1 minute not 5)
    - Auto Scaling (uses more EC2)
    - Elastic IP Addresses
    - OS and Software (windows > linux)
    - Reserved Instances - reserve capacity, receive discount
        - 1 year - no to all upfront - 36 to 40% off On Demand
        - 3 year - no to all upfront - 56 to 62% off On Demand
- Lambda Pricing- Serverless way to do compute- execution time
    - Request pricing
        - Free Tier: 1 mill per month, .20 per 1 million after
    - Duration Pricing - 400GB-seconds per month free
    - Additional - if lambda function uses other AWS resources
- EBS
    - Volumes (per GB) , Snapshots (per GB), Data Transfer
- S3 -
    - Storage class (standard, IA, 1IA), storage (amount), num requests, data transfer
    - Glacier - storage amount, data retrieval times (longer times, more savings)
- Snowball - PB scale secure data transport into and out of cloud
    - Service fee per job (200 for 50 TB, 250 for 80 TB)
    - Daily charge (first 10 days are free, after is $15 a day)
    - Data transfer into S3 free but out is not
- RDS
    - Clock hours of server time
    - Database characteristics
    - Database purchase type (higher perf costs more)
    - Num DBS
    - Provisioned storage (how big)
    - Additional
    - Requests
    - Deploy time
    - Data transfer
- Dynamo DB
    - Provisioned throughput (write) - .47 per write capacity unit
    - Provisioned throughput (read) - cheaper
    - Indexed data storage- .25 per GB
- CloudFront
    - Traffic distribution
    - Num endpoints
    - Data Transfer (out)

Budgets

- AWS Budgets - ability to set custom budget that alerts if cost/ usage > budget
    - Budget costs before incurred
- AWS Cost Explorer- interface to visualize AWS costs data
    - Budget after incurred

Support Plans

- Review - Basic, Dev, Business, Enterprise
    - Tech support for dev - business hours, via email - one person handles all your stuff
    - Response times for dev - < 24 hours in general, <12 for system fails
    - Tech support for Bus and Enterprise - 24/7 email/chat/phone - anyone can handle your stuff
    - Response times for business- < 24 hours in general, <12 for system impair, impaired prod < 4, prod system down < 1
    - Response times for enterprise- same as business but business critical system down < 15 min
- Level support for TAM - Enterprise

Calculators - help calc costs

- AWS Simple Monthly Calc - hosted in S3 - tells you monthly cost for your AWS environ (use tool)
    - Depreciated - AWS Pricing Calc is a service
    - Can pick service, then pick configurations based on min requirements and will tell you monthly costs
- AWS Total Cost of Ownership - what is costing you to be on premise versus on cloud (comparison tool)
    - Creates reports of estim for num servers and amount storage
    - Customizable with currency, region, premise vs co location, number of VMs on servers, DB or nonDB, mem of servers
    - Methodology
    - Assumptions (can exchange)
    - Takes Server costs, overhead, storage costs, it labor costs, network costs, etc

Tagging, Organization, Consolidated Billing

Thursday, August 20, 2020 5:24 PM

Tags are Key value Pairs attached to AWS resources
- Metadata - data about data
- Tags can be inherited - CloudFormation Stack for example can be propagated
- Resource Groups - making groups of resources based on tags- within AWS Systems Manager
  - Contain info such as region, name, EID, department, etc
  - Can have specific info
    - EC2 - public/ private IP
    - ELB - Port Configs
    - RDS - Database Engine
- In AWS - top left resource groups
  - Tag Editor- can search for resources by region, types, and any tags, then create/edit tags for a resource
    - Global service
  - Create a group - per region
    - Query based, can group your resources by a specific tag or if it has a tag **in the region you are in**
    - So it is only for the region in your top right
    - Can use resource group name and "execute automation"
      - Lots of actions such as StopEC2, just choose a target - resource group - name of RG, parameter…

Review
- Tags r key value pair attached to resources, have metadata, can sometimes be inherited
- Resource groups make it easy to group based on assigned tags (one or more. e.g. region, name, health check)
  - Can use automation (per region) via the AWS Systems Manager
- Tag editor is global service - to discover and add tags to all resources

Organizations- account management service to consolidate multiple accounts into one org that is centrally managed
- 2 features - consolidated billing
  - Paying account, linked account, each has monthly bill but paying account gets all of it meaning it gets saving, still indp accounts (cant access resources of others)
  - Adv -
    - one bill
    - Easier to track changes/ costs
    - Discount
  - Billing works so if have 5 reserved (use 3) and 6 on demand in another, consolidated acct, paying is actually 5 reserved and 4 on demand (can move on demand to reserved if didn't use it all)
  - When monitoring enabled on paying acct, billing for all accts is included (can still have alerts per indv acct)
- All features (full access)- can link accounts where all can interact with each other resources
- Best Practices
  - Enable multi factor authen on root, strong password
  - Paying acct should only be billing (not resources)
  - Use CloudTrail
- Is done with root (as usual) , then under is organizations (with policy), under those are accounts and organizations (with other policies), under those are more accts/ orgs
- Example of economies of scale (more stuff means cheaper rates)
- Default limit 20 linkedaccts
- CloudTrail - similar to CloudWatch (mainly EC2 watching performance) - is done in a bucket
  - Monitors API calls in AWS - essentially a trail of changes and actions in the AWS account (great for audits)
  - Per acct and per region, should be turned on in paying acct
  - Then create bucket policy (in paying acct) that allows multi acct access
  - Then have CloudTrail in other accts and use the bucket in paying acct to log all the other acct actions
    - Can only write, not read for example

Lab- Organizations
- Profile - my organizations - is global
- Can create (main one is full access, other is just consolidate billing)
- Once in org (which has the current acct), can invite or create accts
  - Invite - acct id or email and notes
- Can then create Organizational Units (essentially just names at first)
  - Then add accts into groups
  - Can make policies - by default fullaccess (if full access org not just CB)
  - Create own- name, describe, deny or allow, select service, select action in service
    - This is a statement, can add inf statements to a policy
    - Then can go into OU and attach made policy
- Root acct cant invite other root accts

Lab- QuickStart
- Environments that are made partially by Solutions Architects for lots of things (e.g. setting up Microsoft exchange server)
- Free (like CloudFormation)
- Takes you into AWS, CloudFormation, to a stack with the picked template
- AWS Landing Zone - allows set up multi accts with shared environment (faster setup on more than 1 acct)

Review - QuickStart deploys environments quickly using CloudFormation templates built by experts, Landing Zone - can quickly set up multi account AWS environment based on best practices

Security

Wednesday, August 19, 2020 8:40 PM

Compliance on AWS

- Go to aws.amazom.com/compliance
    - Lots of compliance programs (global and per continent/country)
    - Has some responsibilities - example compliant with Payment Card Standards, but an audit will look at your EC2 security groups also to make sure they are safe
    - Can get copies of reports
- AWS Artifact - comprehensive list of access control docs in aws cloud

Shared Responsibility

- While AWS manages cloud, customer manages in-cloud content
    - Example - ec2 needs updated security
    - AWS respon for region, AZ, edge locations (hardware of global infra) as well as compute, storage, database, networking, AWS services code (patching DynamoDB, making s3 work)
    - Customer resp for
        - client side data/ server side data authentication
        - Server side encryption
        - Networking traffic protection (ex. Use https not http since https encrypted)
        - Os (such as EC2 but not say RDS or S3), network, firewall decisions (using aws services safely)
        - Platform, apps, IAM
        - Customer data
- What things you can do in console (EC2 patch, security group, IAM, patch DB on EC2)
- Encryption is shared (s3 needs to be turned on by you but AWS actually does it)

AWS WAF - Web App Firewall and Shield

- WAF - service helps you protect apps from common web exploits that can affect availability, compromise security, consume excessive resources
    - Layer 1 (hardware) to layer 7 (software) - this is layer 7
- Shield - Managed Distributed Denial of Service (DDos) protection service to safeguard web apps
    - Always on, protects against too much traffic
    - Standard Tier - basic
    - Advanced Tier - cost protection - reimburse route53, CloudFront, ELB DDOS attacks - 3k a month
- WAF protects against web app hacker attacks while Shield protects against DDOS (too much traffic)

AWS Inspector/ Trusted Advisor/ CloudTrail

- Inspector (region) - auto security assessment service - helps improve security and compliance of apps
    - Looks for vulnerabilities or deviations from best practices- gives list of findings with severities
    - Installed on EC2 instances
- Trusted Advisor (Global) - helps reduce cost, increase performance, improve security
    - Real time guidance- on Cost Optimization, Performance, Security, Fault Tolerance
    - Like Inspector but for everything
    - Core Checks and Recommendations - free
    - Full Trusted Advisor - Business and Enterprise
- CloudTrail monitors API calls (records AWS environment actions) - saved into S3, which user did what, source IP addresses, when did it

Quiz Notes- CloudGuru

Wednesday, August 19, 2020 8:41 PM

Cloud Terms Quiz

- Objects stored in S3 are stored in multiple servers in mult facilities
- Route53 policies
  - Failover Routing - routes data to 2nd resource if first is unhealthy
  - Latency-Based Routing - routes data to resources with better performance
- Not valid CloudFormation template sections - Options is not on of 9
- LightSail is Platform as a Service

Billing

- Reserved instances for long term, predictable usage patterns
- Consolidated Billing is feature of AWS organizations
  - Configure to see one bill for all AWS accounts
  - Standalone accts may be combined for reduced overall bill
  - Acct charges can be tracked individually
- AZs designed to be secure from failures
- Access Control Lists - manage access to buckets and objects - defines which accts or groups can access and the type of that access

Security

- HIPAA is for secure storage of medical records
  - PCI DSS is for secure payments
- AWS Trusted advisor - cost optimization, fault tolerance, security
- AWS shield
  - Regular
  - Advance - app level monitoring

Overall

- Aurora is AWS' managed database service that is up to 5X faster than a traditional MySQL DB
- AWS Database Migration Service - to migrate existing DB to AWS
- Multi Region deployment is best for globally available app
- Lambda is AWS Function as a Service - runs code without provisioning any servers
- AWS Storage Gateway - used for attaching infrastructure in a data center to infrastructure in AWS Storage (for S3 or other storage) - mountable file system
  - Elastic File System - mountable file storage for EC2
  - Elastic Block Store - block level storage service for EC2
- S3 is global - publicly accessible anywhere in the world
- AWS EMR- web service to make it easy to process large amount of data
- All accounts receive billing support

Quiz Notes 2

Saturday, August 22, 2020 11:44 AM

Practice Tests (acloudguru & udemy*2)

- Services
  - Abuse team will handle malicious activity, Security team is for service security (AWS side)
  - App Discovery Service - plan app migration to the cloud
  - API Gateway - service to create, publish, monitor APIs at scale
  - Athena - analyzes S3 bucket for data analytics
  - Autoscaling - respond to failure by adding more instances (horiz), perf at lowest cost, **doesn't cost anything**
  - Cloud9 - IDE for code
  - Cloud Adoption Framework - help plan adopt cloud
  - CloudEndure - migration to cloud from your machine
  - CloudFront - costs from data transfer out, traffic distribution, number of requests,
    - provides security and content distribution, accelerate static website content delivery, live video streaming, security, custom delivery
  - CloudFormation - have to use templates to host apps and configure services, can create RDS instances
  - CloudWatch - to monitor your apps and check their performance- can detect behavior, set alarms, **visualize logs, collect log files**, troubleshoot
  - Cognito - user sign ins to apps for lots of peoples
  - Config - monitor and assess AWS resource configs, will say how a resource changed, can audit, resource track & analyze security
  - Directory Service - manage users, groups, computers
  - DynamoDB is suited for storing key value pairs , low latency, auto scales to meet throughput capacity, **SERVERLESS, performance at scale, highly available,** RDS (not aurora) doesn't scale, for complex, for transactions and fixed columns
    - Determine appropriate DB technology - number of read/writes per second and nature queries
  - EC2 - IaaS since it is server (hardware) as a service, leaving u to configure, manage, doesn't scale auto (without autoscale)
    - Pricing from buying option, instance type (RAM, number of cores), AMI, storage capacity, region
    - Add ons: **Elastic** IPs, LoadBalancers
    - D**edicated HOSTS better for BYOL** -most scenarios
  - Elastic Beanstalk - no configure servers, deploy and scale web apps
  - Elastic Block Store - **block-level** storage volumes for use with EC2 - primary storage for RDS, good for high read/write performance
    - Cost - volume type, in/out per second, snapshots, data transfer
  - Elastic Container Service - compute to run containerized apps
  - Elastic File System - simple **file-level** storage, great for letting lots of computers access instances
  - Elastic MapReduce - launches clusters (no need for customer node provision, infrastr, etc.) for big data processing- aws responsible
  - Elastic Network Interface - networking add on to EC2 in VPC
  - Elastic Transcoder - video transcoding
  - Hardware Security Module - create and use own encryption keys
  - Instance Store - **temp block-level** storage for EC2, good for caches, buffers, temp content
  - Internet Gateway - horizontally scaled, redundant, highly available VPC component allows VPC to communicate to internet- provides target in VPC for internet traffic & performs Network address translation
  - KMS - key management service - create and control encrypted keys
  - Lambda - don't manage or run servers, can upload apps, scales automatically, price by num requests to function, compute time cons
  - LightSail - launch and manage Virt Private Server
  - Load balancers - respond to failure- for performance
    - App for balancing/distributing **http** and **https** traffic
    - Network for extreme performance (TCP and TLS)
    - Ensure only healthy targets receive traffic
  - Machine Image - AMI - contains software config details for template creating insts
  - Marketplace - performs periodic security checks on products, provides flexible pricing options, quick launch
  - Migration Acceleration Program - uses Professional Services and Partners to assist organizations in migrating to cloud and lower IT costs with better productivity
  - Network ACL - access control list - layer of security to VPC as firewall in and out of subnets, by customer
  - OpWorks- to create AWS resources, deploy automatically
  - Partner Network (APN) Consulting Partners - pro services firms to help customers
  - Personal Health Dashboard - health of AWS services, alert when your resources impacted - Service Health Dashboard is for global AWS Services in all regions
  - PinPoint - marketing communications service- email, SMS, push, voice
  - Service Catalog- simplifies organizing and governing commonly deployed IT services
  - Support Concierge - Enterprise plan, billing & acct experts - 24/7, guidance & best practices, access to specialists for inquiries, training on cost reporting, assistance with service limits, doing bulk purchases
  - S3**- object-level** intelligence tier is good for unpredictable access patterns and saving money, great for media storage as well, scales automatically
    - Glacier - expedited, standard, bulk are 3 types of retrieval (5 min, 5 hours, 12 hours)
  - SES - Simple Email service - email message platform for businesses and devs (people)
  - SQS - Simple Queue Service - hosted queue to integrate and decouple distributed software systems, good for messaging between components of an app
  - SWF - Simple Workflow Service - easy coordinate work across app components
  - Transit Gateway - interconnect all accts to all VPCs and on premise network
  - VPC - can isolate resources and network configs, creates a virtual network in AWS
  - VPN - tunnel your network to aws, doesn't create
  - X-Ray - tracks how your apps performing, troubleshoot
- Cloud Principles
  - Adv over traditional data centers
    - Higher availability - no single points of failure
    - Distributed infrastructure - all the AZ and regions
    - On Demand infra
    - Cost savings - no data centers
  - Availability - no fault tolerance, easy for user to retrieve
  - Coupling as AWS Design principle - low coupling is better
  - Dedicated EC2 are for BringYourOwnLicense - ex federal regs or software licenses
  - Deployment models - cloud, hybrid, on premises
  - Elasticity - to acquire resources as you need them, serverless enable increased agility and lower total cost
  - Federation - single sign on
  - Fault Tolerance
    - DynamoDB, S3 built with multi AZ (for higher availability)
    - EBS and Redshift are in single AZ (EBS replicated)
  - Hybrid Cloud
    - AWS Direct Connect - uses private network connections from premise to AWS - can reduce costs, increase bandwidth
    - AWS VPN - internet based connection from premise to VPC
    - AWS Storage Gateway - to use cloud storage with on premise app
  - Operational Excellence - ability to run and monitor systems to deliver business value and improve processes, procedures continually
  - Penetration testing - can be done without author from AWS for EC2, RDS, CloudFront, Aurora, API, Lambda, LightSail, EBean
  - Reliability - of 5 pillars well architected framework- needs to recover
  - S3 cannot be scaled manually, unlimited storage but objs do have size limit (5 TB)
  - Services -
    - DaaS - Desktop as a Service - to provision different desktops
    - IaaS - Infr as Service - Ec2 - you can configure and manage server
    - PaaS - Elastic Beanstalk, no need to manage underlying infrastructure (hardware and OS, can focus on apps)
    - SaaS - complete product - gmail - end user app
  - Vertical Scaling - increasing size and computing power of single instance
- Plans
  - Business - 24/7 support , Dev has support for business hours only
  - Enterprise - has Infrastructure Event Management (Business has to pay fee)
  - For more EC2 instances, contact AWS and request service limit increase
- Cost
  - Cost Explorer - details on costs for org accts
  - Cost Allocation Tags - track AWS costs on detailed level
  - CloudWatch does performance, not cost
  - Linux EC2 are billed at one second increments with min bill of a minute (1 sec and 61 secs are same price)
  - Non Linux - charged rounding up for the hour (1 sec and 59 mins same)
  - EC2 pricing depends on type (of the 4 plans), AMI, family type (DR MCPXZ), region, data in/out, storage capacity
- Security
  - Core checks - Security groups, s3 bucket permissions, MFA on root acct
  - Inspector - monitors apps to improve security
  - GuardDuty - threat detection service that monitors for malicious activity to protect AWS accts & workloads
  - Have to encrypt EBS at rest and create EBS snapshots to keep Ebs volumes safe - AWS will update EBS (hardware)
  - Security groups control traffic
  - At rest S3, use permissions, versioning, replication (better read), backup (for fails), encryption (server and client side) to protect data
  - Want to enable traceability to monitor actions in environments
- Shared Responsibility
  - AWS
    - Patches host OS (customer does EC2 patch), hardware
    - Configure infrastructure (hardware run AWS), customer does server side encryption
    - AWS does physical and environmental controls
  - Customer
    - Encrypting data on their EBS (on client or server side)
    - VPC and Elastic Compute Cloud are all customer
    - Protecting data confidentiality at rest and in transit
  - Shared
    - Shared controls - apply to both infra layer and customer layers
    - Configuration management- AWS does underlying host and infrastructure, customer does guest OS, DB, apps
    - Patch management depending on which service (customer does Ec2)
- IAM
  - Access Keys - for access from command line interface (username and password), root also has
  - IAM Role - a distributable role from root acct to users (on acct), and doesn't have long term credentials
  - IAM Groups - collection uses treated as unit, does permissions well, AWS Org does billing, who access which buckets, security, compliance
  - IAM Policies attached to Identities - the IAM resource object used to identify and group (users, group,roles)
  - Entities - IAM resource objects used for AWS authentication (IAM users, IAM roles)
  - S3 Lifecycle Policy - to change S3 storage class
  - SSL security deployments - (IAM and Access Control Management)
  - Tagging strategy - good for tracking aws spending across multiple resources, and to quickly identify resources for a project
  - Bucket policy - IAM policy to grant accts or IAM users permissions

Summary

Tuesday, August 18, 2020 4:13 PM

Compute Services (8)- EC2, ECR, ECS, EKS, Lambda, LightSail, Batch, Elastic Beanstalk

Cloud Terms

6 adv of cloud

- Trade capital for variable expense
- Massive econ of scales
- No guessing on capacity
- Increased speed and agility (versus physical setup)
- No data centers (don't compete with amazon)
- Go global in minutes (to anywhere very easily)

3 types of cloud computing

- Iaas- EC2 - u get a server
- Paas - Elastic Beanstalk - they handle servers, u get OS
- Saas - u get inbox

3 types of cloud deployments

- Public cloud - AWS, Azure, GCP
- Private - on premise, u manage (open stack, VM ware usually)
- Hybrid - mix

Physical things

- Region - 2+ Availability Zones
- Azs- 2+ data centers
- Data Center - lots of servers
- Edge Location - endpoints for AWS, cache content (usually CloudFront, Amazon's Content Delivery Network (CDN))
  - Can also write (not just read) to edge locations
  - Objects cached for a certain time (TTL - Time to Live)
- The right region
  - Data sovereignty laws (federal regulations)
  - Latency to users (closer is less latency)
  - Services (some regions have more - US East 1)

Different Support Packages

- Basic - free
- Dev - 29/month, scales on usage
- Business - 100/month, scales on usage
- Enterprise - 15k/month, scales on usage, get a TAM

Billing alerts/ alarms

- Will tell you if you spend x amount

IAM - identity access management

- GLOBAL (don't specify region for user roles)
- Create users or groups- user or group can be accessed in any reg
- Root acct is email to set up AWS, has full admin access, should be protected with MFA, better practice to create users who can use root AWS but cant do anything
- A group is a place to store users - users inherit group permissions (e.g. devs, system admins, human resources, finances)
  - Permissions made with policies - policies consist of JavaScript Object Notation code - referred to as key value pairs (e.g. "name" : "A Cloud")

Ways to Log in to AWS

- Console (browser)
- Programmatically (command line)
- SDK

S3 - simple storage service

- Is Object Based
- 0 B to 5 TB
- Uses buckets (folders) to hold files
- Unlimited storage
- S3 is universal access- can be accessed from anywhere though lives in region- thus name must be unique globally
  - https://s3-eu-west-1.amazonaws.com/acloudguru
- Not suitable to install Database of OS code on (dynamic code)
- Successful uploads will generate HTTP 200 status code
- Key fundamentals
  - Key - name of object
  - Value - the data (sequence of bytes)
- Read after write has immediate consistency
- Read after puts of deletes has eventual consistency (takes time to update or delete)
- Can replicate buckets - cross region replication
- S3 Transfer Acceleration - upload to endpoints not bucket, faster to use Amazons network then straight to bucket
- Types
  - Standard - 11 9s avail, stored in mult facilites, can sustain 2 facilities lost
  - IA (infreq access) - for less accessed data but rapid access, cheaper than s3 but retrieval fee
  - One Zone IA - low cost IA but no mult AZ
  - Intelligent Tiering - uses ML to move around tiers
  - Glacier - minutes to hours
  - Glacier Deep Archive - 12 hours
- Bucket policies
- Can create static websites (ex html, not db/ wordpress for example)
- Scales automatically to meet our usage demands
- CloudFront - request file first time, saved on edge location, next request is faster
  - Origin - CDN distributes from this (either S3, EC2, ELB, Route53)
  - Distr - name, consists of collection of Edge Locations
  - Web Distr - typically for websites
  - RTMP - used for media streaming (we used web)

EC2 - server instance that can resize capacity quickly as requirements change

- Models
  - On Demand - pay by hour
  - Reserved - discount for higher upfront and longer contract
  - Spot - Can set bid and if higher stops working
    - If you terminate, you do get issued
    - Would use when don't need app on always (badge verif overnight when cost is low)
  - Dedicated Hosts - physical EC2, can reduce costs with your software licenses or regulatory issues
- Types
  - FIGHT DR MC PXZ
- EBS
  - SSD
    - GP2 - Gen purpose - normal
    - Provisioned IOPS - high performance
  - Magnetic
    - ST1 - throughput optimized - low cost for freq access
    - SC1 - Cold - lowest cost for less freq access
- Is compute based service, not serverless since it is a server…
- Private key to access
- Ports
  - Linux - SSH - 22
  - Microsoft - RDP - 3389
  - Http - 80
  - Https - 443
- Firewalls - rules per port
  - 0.0.0.0/0 - everything
  - x.y.z.w/32 - only one IP addess
  - Security groups - virtual firewalls
- Always design for failure - have at most 1 EC2 per AZ
- Accessing EC2s is easiest done with roles rather than passing out individual key ids and secret access keys

Billing

- Based on pay as go, pay use, pay less with more, pay less for reserve
- Cap Expenditure - upfront, fixed, sunk
- Opex - operational - variable based on usage
- 4 kinds of EC2 - On Demand, Reserved, Spot (for flexible on/off app), dedicated (can use ur software)
- Free services - VPC, IAM, Consolidated Billing, Elastic Beanstalk, CloudFormation, Opworks (last 4 use costly resources)
- Budgets (before costs) while cost explorer (after cost analysis)
- 4 plans - Basic, Dev (29), Business (100), Enterprise (15k, TAM)
  - Much better response times for Enterprise
- Tags - key value pairs on AWS resources, basically metadata, can be inherited sometimes
- Resource Groups - easy group resources using one or more tags
  - Examples: region, name, health checks
  - Can automate to do action on all in group (using Systems Manager) by the specific tag
- Tag editor - global service, can discover resources and add/edit their tags
- Organizations - has root then has org units under root, and each OU can have accts
  - Orgs and accts can have policies
  - Best practices - enable MultiFactor Auth on root, have strong password, usually only pay with root acct
  - 20 linked accts by default
  - Billing alerts - with monitoring, billing data for all accts can be paid by root (still have info on each acct)
- CloudTrail - per acct and per region - records actions of all accts
  - Can consolidate action logs into an S3 bucket by creating bucket in root with cloudtrail on, enable multi acct access, having indv accts enable cloudtrail and add to bucket
  - Essentially aggregates into paying acct
- Consolidated billing - allows volume discounts, unused reserve instances can be applied to group
- Quickstart- deploy environments quickly using CloudFormation templates (built by experts)
- Landing Zone - deploy multiple accts, one environment, based on best practices
- AWS Pricing Calc- how much will all these resources cost?
- AWS Total Cost of Ownership- how much to have physical servers vs using cloud

Security

- Artifact - retrieve Compliance reports (documents on if AWS complies with specific things)
- Shared security model
  - AWS- hardware/ global infra (AZ,Region) and the software of that (RDS, MySQL on it), underlying systems (how to run s3)
  - Consumer - EC2 patching, patching apps on EC2 (MySQL if not on RDS), S3 security (policies)
- WAF- Web APP firewall - stops hackers
- AWS Shield - stops DDOS - cant spam u
- AWS Inspector - inspects EC2, reports on vulnerabilities
- AWS Trusted Advisor - does Cost opt, security, as a general helper to your AWS acct
- AWS CloudTrail - recording AWS actions, who did what when
- AWS Athena - interactive queries on your S3 based on SQL, serverless
- AWS Macie - uses AI to identify PII and CloudTrail logs to prevent ID theft

        - Done immediately- are global, product to f IAM
- Load balances
    - Network Load Balancer - extreme performance/ static IPs
    - App Load Balancer - layer 7 (intelligent, see into traffic)
    - Classic LB - low cost for test and dev

RDS - relational database - SQL/OLTP (online trans process)
- SQL, Aurora, MySQL, PostgreSQL, MariaDB, Oracle
- Non relational - DynamoDB - No SQL
- Red Shift OLAP - Online Analytics Processing - data warehousing
    - Business intelligence or data warehouse
- Elasticache - very freq queries, memcached and Redis
    - For better performance on most freq
- Graph Databases - Scalability , high availability (Neptune)
- 2 key features
    - Multi AZ (Disaster recovery)
    - Read Replicas - for better performance

Autoscaling - can create multiple EC2s and they will scale as needed
DNS - Domain Name System - works like phonebook of names to Ips
- Route53 - how to buy domains
    - Global, similar to IAM and S3
    - Can direct traffic all over world

Elastic Beanstalk - upload code for app and AWS handles capacity provisioning, load balancing, scaling, application health monitoring
- Limited in cant be programmed and not all resources can be used

CloudFormation - setup resources and AWS will use template to create and configure any resources you need
- Free but resources not
- Basically has to be programmed

CloudWatch - all about performance
- Used for monitoring performance - can create alarms which trigger notif
- Can monitor most of AWS as well as your apps
- CloudWatch with EC2 monitors every 5 minutes by default, can be changed to 1 minute with detailed monitoring

Systems Manager
- Can be used to manage fleets of EC2 similarly to how we manage 1
- Piece of software is installed on each VM
- Can be either on AWS or on premise
- Run command to install/patch/uninstall
- Integrates with cloudwatch to give dashboard of entire ec2 fleet

Global AWS services
- IAM - users/group/policy are global, not specific to region
- Route53 - global domain name (doesn't change per region)
- CloudFront - CDN connects edges so has to be global
- SNS
- SES

Global Use but are regional
- S3 - can be accessed anywhere but bucket lives in one region

Services on premise
- Snowball - gigantic disk (delivered to headquarters, send it back to amazon) - 80 TB
- Snowball Edge - also has CPU (can deploy lambda function on premise)
    - Helps bring AWS to you
- Storage Gateway - on premise all time Snowball - for caching files and repl to S3, have it locally
- IoT Greengrass - IoT connects devices to cloud
- CodeDeploy - to EC2 or premise Web Servers deploy apps
- Opsworks - uses Schiff to deploy app code to EC2 or on premise web servers
- Code Deploy and OpsWorks can deploy apps

Intro

Wednesday, August 26, 2020 10:55 AM

- Important Concepts
    - Which AWS Services are Serverless or not?
    - Difference between Read Replicas and Multi-AZ deployment
    - Versioning and MFA Delete
    - Instance stores
    - Three types of EC2 Placement groups
    - ENI vs. ENA vs. EFA
    - Stateless (NACLs) vs. stateful (security groups)
    - SQS and how it is different from SWF/SNS
    - Components of a VPC and how they work together
    - Different DNS routing policies
    - How ELBs work generally

CP Duplicate Training

Monday, March 16, 2020 11:17 AM

If you completed CLF-C01, you can skip the following videos:

- **AWS - 10,000 Foot Overview**
- **How to Sign Up to AWS**
- **IAM 101**
- **IAM Lab**
- **Create A Billing Alarm**
- **S3 101**
- **Let's Create an S3 Bucket**
- **AWS Organizations (watch Exam Tips)**
- **CloudFront Overview**
- **Snowball Overview**
- **Snowball Lab**
- **Athena vs. Macie**
- **EC2 101**
- **Let's Get Our Hands Dirty with EC2 Part 1 (can watch at high speed)**
- **EBS 101**
- **CloudWatch 101**
- **Using IAM Roles with EC2**
- **Databases 101**
- **Let's Create an RDS Instance**
- **Route53 - Register a Domain Name Lab**
- **CloudFormation**

If you completed the S3 Masterclass section, you can also skip these videos:

- **S3 Pricing Tiers**
- **S3 Security and Encryption**
- **S3 Version Control**
- **S3 Lifecycle Management and Glacier**
- **Cross Region Replication**
- **S3 Transfer Acceleration**

IAM 1 & 2

Wednesday, August 26, 2020 10:56 AM

IAM Review

- Manage users, level of access
- Offers centralized control, shared access to one act, permissions for each, identity federation (using other logins like fb), MFA, temp access when necessary, allows password rotation policy, integrates with other AWS services, supports PCI DSS (credit card) compliance
- IAM things
    - Users - end users (people, employees)
    - Groups - collection of users (have group permissions)
    - Policies - made of documents (JSON)
    - Roles - assign to aws resources, other accts, corporate directory, web identity
- IAM is universal
- Root acct is god mode, new users have no permissions - get access ID, secret access key but no permissions
- Can create password rotation policies
- Billing alarm - in CloudWatch, alarms -> billing, create alarm
    - Creates SNS topic which sends notification to email about too high of charges

Org and Consolidated Billing

- One bill per acct, easy track charges and costs, volume pricing discount
- Enable MFA on root, paying acct should only be for billing, strong passwords, enable Service Control Policies on Org units or users
- Role - to ec2 or aws acct has permissions policies
    - Has role link - can use link to switch an accts role to this newly created role
    - Essentially assumes the role via the link

IAM Policies- **need to be attached**

- ARN - Amazon Resource Name uniquely identify any AWS resource
    - Partition (aws vs aws china), service, region, account_id, end with res type/resource name/qualif(e.g. *)
- Identity policy - attached to IAM user, group, role (permissions)
- Resource Policy - to resource, access and actions
- JSON- Version- date, statement (API requests)- within, Sid- descrip, effect (allow/deny), action (service and action) then resource - has ARN
- Can create policy easy then attach to role to be given to resource or user
- Can make inline policies - for just one role
- Explicit deny overrides, permissions implicitly denied
- Permission Boundaries - delegating administration to other users, prevent priv escalating or unnec broad perm
    - For devs and setting max that a person can have

AWS Resource Access Manager

- Allowing resource sharing between accts- EC2, aurora, codebuild, licensemanager, EC2 image builder, etc
- Subnets between accts are only reads, cannot edit (except tags)- have to accept invitation in RAM of receipt
    - Can create clone and mess with

AWS Single Sign On

- For centrally managing access to AWS accts or business apps (slack, office, aws, google)
- Use Active Directory to get AD Trust, and with that user can do app things, aws things, SAML things
    - Sec Assertion Markup Language- for logging in based on sessions in other apps
- Recorded in CloudTrail

AWS Directory Service- family managed services (connect AWS with on premises)

- Use aws with corporate credentials, single sign on to any EC2 joined
- Active Directory - on premises, hierarchical DB for users, groups, computer.
    - Uses group policies, LDAP, DNS, Kerberos, NTLM authentication
    - Highly available and managed service takes care of overhead
- AWS Managed Microsoft AD
    - Has domain controllers running on windows servers, reachable by apps in your VPC, exclusive
    - More DCs for higher avail and performance
    - Can extend AD to on premises using AD Trust
    - AWS inc harge of Multi AZ deployment, patch monitor and recover DC, instance update, snapshot restore
    - Customer in charge of user, groups, group policies, can use Standard AD tools, scaling DCs, trusts, certificates, Federation
- Simple AD- standalone directory for basic AD features
    - Small (<500 users), large (<5000)
    - Makes easier to manage EC2, good for Linux workloads, no trusts (ability join on premise ADs)
- AD Connector
    - Using on premise AD with AWS (e.g. ec2), avoids caching info in cloud
- Cloud Directory - store for devs, multiple hierarchies with lots of objects, use for org charts, course catalogs, device registries
- Amazon Cognito User Pools - managed user directory for SaaS apps, used with social media
- AD Compatible
    - Managed Microsoft AD, AD Connector, Simple AD - connect to AWS with AD credentials
    - Cloud Directory and Cognito User Pools are not (good for no AD but need hierarchical info)

S3

Wednesday, August 26, 2020 12:41 PM

S3 Review
- Secure, durable, high scalable object level storage (objects r flat files)
- Spread across facilities
- Stored in buckets (folders)
- Universal namespace - must be global unique for unique web address - http 200 code if upload successful
- Objects
Key - name	Value - data	Version id - versioning
Metadata - data about data (i.e. finance dep)	Sub resources - access control lists	Sub resources - torrents
- Data consistency - read after write consistency, eventual for updates and deletes
- Features -
tiered storage	lifecycle management - can move things after some time
Encryption	Versioning
MFA delete	AccessControlLists - instance level & Bucket Policies - bucket level
- Billing- Storage, requests, type pricing, data transfer, transfer acceleration, cross region replication (6)
  - Can change bucket type pretty easily
  - All s3 have volume savings (50 TB)
  - Intel tier does standard and IA
- Transfer accel - secure, fast transfer of files via endpoints
- OS installation is for block based storage
- Buckets have to be possibly public or not, then files within have to be public or not
- ACL - can be used to block all the way to the object stored
- Bucket policy - used to the bucket access rules

Security
- ACL- object level seceurity, bucket policies- can create access logs

Encryption
- In transit - between ur computer and server - SSL/TLS when using http
- At rest
  - Server side - S3 managed keys - SSE, KMS. Customer provided
  - Client Side - you do it then upload to s3

Versioning
- Store all versions (all writes), cant disable only stop, integrates with lifecycle rules, works with MFA delete
- In properties - new files are auto private, have to be made public
- Increases bucket size very quickly
- To restore, delete the delete marker
- Does have MFA delete capability
- Great backup tool

Lifecycle Management
- In Management - can be done on specific resources via tags or general buckets
- Can set when move files to other s3 bucket types after x days (e.g. move to IA after 30 days)
- Can control for current versions or previous versions
- Then can set when to permanently delete (e.g. after 425 days from creation/ becoming a prev version)
- Automates moving objects between different storage tiers

Object Lock and Glacier Vault Lock
- To store objects using write once, read many - helps prevent objects from being deleted or modified
- Object Locks can be used to meet regulatory reqs. or add extra protection, on bucket or instance, 2 flavors:
- Governance mode - users cant overwrite or delete object version, or alter lock settings (unless u give special permissions)
- Compliance mode - protected object cant be overwritten or deleted (even by root user) for a retention period
- Retention period - protects object for fixed amount of time
- Legal hold - like retention period, but can be indefinite (can be removed by anyone w/permission)
- Glacier lock is for compliance controls on glacier, once locked policy cannot be changed

Performance
- S3 prefixes - the middle parts between a bucket and object (all the folders)
- Performance is about prefixes - by using more prefixes, you get better perf
- KMS - hit KMS limits eventually - there is KMS quota
  - When uploading and downloading files, KMS API calls GenerateDatakey and Decrypt
  - Regions have specific quotas (5500 requests per second, 10k, etc)
- Multipart uploads - increase efficiency by splitting objects into parts (rec over 100 MB and required over 5 G)
- Multipart downloads - byte range fetches - parallelize download by specifying byte ranges - can also be used to only download specific parts of a file

S3 Select
- Apps retrieve only subset of data from object using simple SQL - can drastic increase performance
- Glacier Select - same by letting SQL on Glacier (gets data in rows or columns, save money on data transfer)

Share s3 bucket across accts
- Bucket Policies & IAM (applies to whole bucket) - programmatic access only
- Bucket ACLS & IAM (individual objects) - programmatic access only
- Cross Acct IAM roles- programmatic and console access

Cross region replication
- Requires versioning on source and destination buckets
- Can replicate entire bucket or based on tags/ prefixes (folders)
- Doesn't replicate existing, only future
- Deletes (delete marker or actual object) don't replicate across buckets

S3 Services

Wednesday, August 26, 2020 2:00 PM

Transfer Acceleration
- Use cloudfront to accelerate uploads - url is used to upload to edge location then that goes to S3 bucket using aws network

DataSync
- Allows move large datasets to cloud - usually for on premise
- Encrypts and data security- NFS and SMB compatible file systems
- Used with DataSync agent installation
- Can be used to replicate EFS to EFS, usually on premise
- Replication can be done hourly, daily, weekly

CloudFront
- Content Delivery Network - for getting content to users faster (by not accessing to original bucket)
- With edge locations (cached timed to live in seconds) (are not just read only) can invalidate cached data, but it will cost money
- Distribution - name
- Can be used for entire website (dynamic/static)
- Web distribution and RTMP (adobe and media) types
- Can add WAFs
- Has domain name, in bucket is domain name/nucketname
- Can invalidate distribution (removes things from edge locations)
- Signed URLs or Signed Cookies
  - Can restrict access to CloudFront (e.g. for paying fees)
  - CloudFront Signed URL - for having 1 file, 1 file = 1 URL
    - Origin is EC2, can use ELB etc
    - Key pair is acct wide and managed by root user
    - Can use cloudfront cache
    - Can filter by date, path, etc
  - S3 signed URL - for S3 and indiv file
    - Doesn't use CloudFront
    - Issues a request to make user the IAM user who created the presigned url
    - Limited lifetime
  - Signed Cookie - 1 cookie = multiple files
  - With both attach a policy- can include URL expiration (how long its valid), IP ranges (actual IPs), trusted signers (which aws accts can create signed urls)
  - Origin access identity can be used to only allow access from cloudfront to bucket

Snowball- import/export to s3
- Petabyte scale data transport solution for secure and cheap transfer
- Snowball edge - bigger and better - ex airlines
  - 100TB with storage and compute capabilities (temp storage, support local workloads in remote/ offline)
  - Like a portable AWS
- Snowmobile - exabyte - 100 PB scale - it's a truck

Storage Gateway
- Connects on premise software to AWS - cost effective way to replicate your data to AWS
- Can be a VM on your hosts
- File Gateway - NFS and SMB- for flat files
  - Can be moved to S3 and versioning, lifecycle management, cross region repl can apply
- Volume Gateway - iSCSI - Storing snapshots (EBS) and Virtual HDD - for HDD or VolumeDD, both dataset on S3
  - Stored volumes - lets you store primary data locally- entire data set **on site,** creates backups, asynchronous
  - Cached Volumes - not entire data set, most frequent used data **on site**
- Tape Gateway - virtual tape cartridges
  - For archives (tapes) or backups

Athena
- Query service on S3 using SQL, serverless (pay per query/scan)
- Query log files, make business reports, on click stream data

Macie
- AI service on S3 to find sensitive info
- Orevent ID depth, good for PCI-DSS, alerts

EC2

Thursday, August 27, 2020 11:17 AM

Review
- Resizable reduce capacity
- Reduces time to obtain and boot new servers- quickly scale up and down
- Pricing models-
  - On demand- unpredictable workload, low cost, flexible, no upfront, contracts
  - Reserved (discount for contract)- steady state/predictable usage
    - Standard - up to 75% off
    - Convertible - up to 54% off, can change attributes of RI if RI >= value
    - Scheduled - time windows
  - Spot- flexible start and end, very low, large needs for low cost, charged for hour if you cancel
  - Dedicated- licensing, regulatory, can be purchased hourly or reserved
- Pricing factors -
- EC2 when created has private (your key) and public key (padlock) to access
- Can connect with connect button by ssh ing in- creates browser ssh session
  - Can get chrome ssh extension (secure shell app) - need IP address as host name, username is ec2-user
  - In command prompt, go to where private key is, create public key with
    - **ssh-keygen -y -f <nameOfKey> > <newnameofFile>**
    - Should have .pub now
    - Use **ren <nameprivatekey>.pem <nameprivatekey>** to get rid of extension, very important
  - Import both files (identity should be nameofprivatekey)
  - Install apache - **yum install httpd -y** make it a web server
  - takes to bucket directory **cd /var/www/html**
  - To start httpd service - **service httpd start**
  - Start httpd service if instance reboots - **chkconfig on**
- Can add storage when creating EC2 instance- storage volumes
  - Termination protection is off by default (another protect against accidental termination)
  - Root - can pick size, volume type, IOPS (in out per second), encryption , default terminated with instance
  - EBS - can pick from more volume types (Throughput HDD for data warehouse, cold cheaper), size
  - Both root and additional storage volumes can be encrypted
- Security group
  - Changes to inbound rules are immediate
  - 0.0.0.0/0 IPv4, ::/0 IPv6
  - When creating inbound rules, identical outbound rules are created- stateful (NACLs are stateless)
  - Only whitelist, cant blacklist (everything is blocked by default) inbound traffic
  - All outbound traffic is allowed by default
  - Can also add more one security group - allows more ports, can add inf instances to security group

EBS
- Elastic Block Store - virtual HDD in cloud, block storage, durability and security
- Types (mostly decreasing IOPS)
  - Gen purpose (gp2) - SSD with good price and performance - for most work, default root for EC2
  - Provisioned IOPS (io1) - high performance SSD for mission critical apps - for databases
  - Throughput Optimized (st1)- low cost HDD for freq access, throughput intense - for data warehouses
  - Cold HDD (sc1) - lowest cost HDD for less freq access - file servers
  - Magnetic (standard) - old HDD - workloads with infreq access
- In same AZ as EC2 instances
- Can change size and volume types of EBS volumes - may take some time
- Can change EBS AZ by creating snapshot of EBS, create image from that snapshot (hardware assisted virtualization for more flexibility when creating new EC2), then create EC2 from AMI, and in config change subnet
  - Or copy AMI to move across regions (and when making EC2, in a new AZ)
- Snapshots and AMI cost money
- Snapshots - incremental (copy over changes since last snapshot), should stop EC2 before snapshot of root, exist in S3, can take while running, can create AMIs from snapshots, volumes always in same AZ as EC2, can change type & size
- Cant delete EBS volume if root of an AMI (AMI cant exist without)

EBS vs Instance Store
- Select AMI on Region, OS, Arch, launch permission, storage for root - Instance store, EBS backed volume
- EBS - root device for instance from AMI is EBS volume created from Amazon EBS snapshot, can
- Instance Store - root device from AMI is instance store volume created from template in S3
  - Done by choosing AMI as instance store
  - Cannot be stopped (just terminated and rebooted)
  - Ephemeral storage - if not healthy and stop, **lose data** (not persistent)
  - If EC2 deleted, root volume is gone (can choose not to delete with EBS)
- Both can be rebooted and wont lose data

ENI vs ENA vs EFA
- ENI - Elastic Network Interface- Virtual Network Card - for basic networking
  - Primary private IPv4, can get one or more from range on your VPC
  - One Elastic IP per private IPv4, one public, one or more Ipv6
  - Use for management network, network and security appliances in VPC, can create dual -homed instances with different subnets, low budget high availability solution
- ENA - Enhanced Networking - for enhanced networking
  - Uses single root I/O virtualization for high performance and lower CPU virtualization
  - Higher bandwidth, higher packets per second, EC2 has to be able to use it
  - Enabled via **Elastic Network Adapter-** supports network speeds up to 100 Gbps for supported instance types
  - Enabled via Virtual Function interface - up to 10 Gbps- usually used on older
- EFA -Elastic Fabric Adapter - attach to EC2 to accelerate High Performance Computing & ML
  - Lower and more consistent latency and higher throughput, can use OS -bypass - enables HPC (linux)

Encrypting Root Volumes
- For unencrypted volumes (can encrypt upon creation) - go into volumes and actions -> create snapshot
- With snapshot, copy snapshot but click encrypt copy
- With encrypted copy, create image from it
- Now encrypted image, can make an EC2 from it
- Volume launched from encrypted AMI must be encrypted
- Can share snapshots only when unencrypted

Spot
- Up to 90% discount, for flexible fault tolerant apps
- Decide on max price - provisioned as long as spot price is below ur max price (2 minutes to decide if you want it to die)
- Spot block - to stop your spot instances from being terminated, can be set for 1 to 6 hours, on a regional basis
- Useful for big data and analytics, web services, containerized workloads, high perf computing, media rendering, CI/CD & testing
  - Not good for persistent workloads, critical jobs, dbs
- How to terminate spot instances - create with request (max price, num instances, one time or persistent, valid until)
- Spot Fleets - Collection of instances or On Demand instances
  - Attempts to match target capacity with price constraint
  - Has pools - ec2 instance types, OS, AZs - and strategy implements pools to do the above
    - Capacity optimize - pool with optimal capacity for number of instances running
    - Lowest price - pool with lowest price
    - Diversified - distributed across all
    - InstancePoolstoUseCount - across specific pools with lowest price
  - Stops launching instances when price or capacity met

EC2 Hibernate
- When start EC2 - OS boots, data script runs (bootstrap), app starts, preserves in mem ram on EBS
- Saves contents from RAM to EBS root volume (when reboot, much faster- same instance ID)
- Great for long running instance
- To use - actions - stop hibernate
- Must be less than 150 GB, cant hibernate for more than 60 days, for On Demand and Reserves

CloudWatch
- Monitors Compute (EC2, Autoscaling, ELBs, Route 53 health checks), Storage & CD (EBS Volumes, Storage Gateway, CloudFront)
- Metrics on CPU, Network, Disk, Status Check, every 5 minutes by default (1 min with detailed monitoring, not free)
- In AWS - CloudWatch - EC2 - per Instance - choose metric (e.g. cpu) , name , description, when >= value then set alarm (SNS)
- Can create dashboards for regions/ global, also logs and events (responses)

CloudTrail
- Recording actions and API calls, can get source IPs, when occurred

EC2 Cont

Thursday, August 27, 2020 5:20 PM

Roles
- In IAM, for services, accts - for the role, can use service and has your chosen policy
- To attach role to ec3 - in ec2 -> Actions -> instance setting -> replace IAM role then select role
- Can ssh into EC2 like normal, but before couldn't access anything in aws without role - now that access role is on ec2, our key can access aws (within the applied role)
- Far more secure than access keys, easier to manage, can be assigned with console or CLI, universal

Bootstrap
- Advanced details in config as text
```
#!/bin/bash
yum update -y
yum install httpd -y
service httpd start
chkconfig httpd on
cd /var/www/html
echo "<html><h1>Hello Cloud Gurus Welcome To My Webpage</h1></html>" > index.html
aws s3 mb s3://YOURBUCKETNAMEHERE
aws s3 cp index.html s3://YOURBUCKETNAMEHERE
```
- First line always necessary - shabang!
- Updates, install apache, start apache (web server), have apache reboot if instance reboots, change to home directory, create stuff, make bucket, move index to bucket

Instance Metadata
- Can ssh into ec2 then use **curl http://169.254.169.254/latest/user-data** for user data (bootstrap code)
- Can put that into a .txt
- Can ssh into ec2 then use **curl http://169.254.169.254/latest/meta-data** for meta data
- Can get local-ipv4, public-ipv4 within meta-data by going into those folders

EFS
- Elastic File System - for mounting file systems on EC2 instances, supports Network File System v4, for **LINUX**
- Pay for storage you use (no preprevision), can support thousands connections, stored across multiple AZs
- **yum install -y amazon -efs-utils** to have EFS work on these instances
- Is elastic, can have lifecycle management (move to EFS infreq access),
- If we want to interact with web server - put default security group on EFS and diff security group for web servers
- On EFS security group, add inbound rule for NFS (TCP protocol, 2049 port), port is web server security group
- To mount - **mount -t efs -o tls fs-9816b269:/ efs** with code online, efs replaced with /var/www/html
- If EFS mount code ran on more than one server, can see shared files, edits on files replicated

FSx for windows
- It's a windows file server - great with Microsoft apps, runs Windows Server Message Block, supports AD users, ACLs, security groups, Distributed File System (**SMB or DFS based**)
- EFS is managed NAS filer for EC2 based on NFS (NFSv4, Linux)

FSx for Lustre
- Fully managed file system optimized for compute intense workloads - **large data sets and huge IOPS, HPC**

EC2 Placement Groups
- Clustered - **grouping** of EC2 within AZ (closely), recommended apps need **low network latency or high throughput**
- Spread Placement - on **distinct** underlying hardware- for apps that have a **small number of critical instances,** up to 7
- Partitioned - like spread but partitions have 1+ instances - no partitions on same hardware rack- **HDFS, HBase, Cassandra**
- Cluster is in same AZ where spread/ partition can be in diff AZ same region, name must be unique within AWS acct
- Only instances on PG - Compute Optimized, GPU, Mem Optimized, Storage Optimized (rec same instances in PG)
- Cant merge PGs, to move instance into PG must be stopped and via the CLI or SDK

HPC
- Data Transfer
  - Use Snowball, Snowmobile
  - Use DataSync to store on S3, EFS, push data from AWS from on premise
  - Direct Connect - dedicated network connection from premise to AWS
- Compute and Network
  - Use EC2, EC2 fleets/spot fleets along with PGs - have to be GPU or CPU optimized
  - Enhanced Network
    - Uses single root I/O virtualization for high perf capabilities and lower CPU- when you want better network perf
    - ENA - most of the time, up to 100Gps
    - Virtual Function - up to 10Gps, legacy
  - EFA- lower and consistent latency - faster, OsBypass, Linux
- Storage
  - Instance attached
    - EBS - up to 64k IOPS
    - Instance Store - millions IOPS with low latency
  - Network
    - S3 - Distributed object based- not file system
    - EFS - scales IOPS, use provisioned IOPS, file system
    - FSx for Lustre - HPC optimized file system, millions IOPS, backed by S3
- Orchestration/ Automation
  - Batch - to run batch computing jobs, supports multi node parallel job
  - Parallel Cluster- deploy and manage HPC clusters, has VPC and subnets

WAF
- Firewall monitor http/https request - app level so sees data in app
- Works in 3 ways - count all requests except specified, block all requests except specified, count requests match properties
- Define conditions with IP addresses, country requests, strings in requests, length, SQL code (injection), scripts (cross site scripts)

HyperVisor
- Virtual machine monitors for running Oss simultaneously on one computer - Underlying for EC2 - Xen and Nitro

Databases

Thursday, August 27, 2020 5:21 PM

Review
- Relational (MySQL) - fixed columns - mySQL on wordpress, Aurora RDS and OLTP
  - Key features - Multi AZ (availability- fault tolerant) and read replicas (performance, not fault tolerant, 5 copies)
- NonRelational (no SQL) - Key value (JSON)
- Data Warehousing - Business Intelligence - redshift (OLAP)
  - OLTP - transaction processing - single orders
  - OLAP - analytics - lots of data
- Elasticache - deploy, scale in-memory cache in cloud - Memcached and Redis

RDS
- Can set type, template (how powerful),identifier, credentials, DB instance type, storage, connectivity settings, VPC security group (travels as MySQL/type), DB name, auto backups, maintenance time (time per week for changes)
  - RDS runs on VMs, cant access but exist, **not serverless** (except Aurora)
- Along with RDS, make EC2 instances for dynamic website (make sure EC2 open to RDS via inbound rules open to MySQL security group)
- Connectivity endpoint necessary for wordpress (database host)
- Need to write in ec2 config.php (use nano after ssh in)
- Backups
  - Automated
    - Recover anytime during retention period (set up to 35 days), takes full daily snapshot, and store transaction logs throughout day - recovery gets most recent daily back up and apply transaction logs (you can do point in time recovery down to seconds)
    - Enabled by default, backup data stored in S3 and free storage up to size of DB
    - May increase latency during backups
  - Snapshots - user initiated, stored after \RDS deleted, restored is different RDS instance
  - Multi AZ- for availability - moves over if failed
  - Read Replicas- for performance - used for scaling, must have auto backup on to use, up to 5 copies, each replica has own endpoint, can have multi AZ, can have replica in different region, can be prom to own DB
- Encryption- on all RDS, done using KMS- once RDS instance encrypted so is data at rest, backups, read repl, snapshots
- On Console - shows role (master vs replica)
  - Actions - take snapshot, create aurora read replica
  - Modify - turn on multi AZ (can hit on performance so usually do during maintenance)
    - Actions - reboot with failover to change AZ
  - Read Replicas - need backups turned on (in modify - retention period >1) then actions create read repl (not SQL server)
    - Can put in any region, encrypted, multi AZ, instance name - can be promoted to master (becomes own)
  - Move to aurora -

DynamoDB
- Super-fast dB at any scale
- Stored on SSD, spread on 3 different data center, eventual consistent reads (read after 1 sec), strong consistent reads (returns within 1 second)
- DynamoDB Accelerator - DAX
  - Fully managed, highly available in memory cache, 10x performance - faster than milliseconds, Compatible with usual Dynamo DB usage
  - Normally cache on side, have to write it yourself, can only do reads
  - DAX can do high read and writes, has high availability
- Transactions - ex financial and fulfilling orders
  - Allows us to deal with multiple tables, does 2 read/writes (prepare and commit)
- On Demand
  - Pay per request, easier balance cost and performance, no min, no read/write (provisioned does), though higher pay per request (better for new things)
- Backup and restore
  - Full backups at any time, retained until delete, same region as source table, no impact on table performance
- Point Time Recovery - any point in 35 days, not by default, incremental update
- Streams - time order sequence of item level (first in first out), for 24 hours, inserts/deletes/updates - made of shards
  - Can use for cross region replication and relationships among tables, combine with lambda
- Global tables - multi master, multi region- based on streams, auto, for high availability and disaster recovery, very fast
  - In created dynamo DB table, global tables -> streams enabled (change log) -> add region (destination)
- DMS - Relational database on premise, rds, ec2 - DMS takes source db to target db (all sources + dynamo), no slowdown source
- Security - encryption at rest with KMS, can access DynamoDB with site to site VPN, Direct Connect, IAM policies and roles, fine grained (per row/object), VPC endpoints (IPs)

Redshift- for business intelligence
- Petabyte scale data warehouse (database and infrastructure diff for warehour)
- Single node - 160G
- MultiNode - has leader node (manages client/ receives queries) with compute nodes (do the work)
- Uses compression based on column (much easier) and Massively Parallel Processing (scale out)
- Backups by default (1 day retention), max is 35, always has 3 copies(original, replica, s3), can replicate snapshots to S3 in another region for disaster recovery
- Price - compute node hours (billed for 1 unit per node per hour), only compute nodes, for backups and data transfer (in VPS)
- Security - encrypt in transit with SSL, encrypt at rest with AES, redshift takes care of keys (yours with HSM, or theirs with KMS)
- Availability - only available in 1 zone, can restore snapshots to new Azs

Aurora
- RDS amazon - MySQL and PostgreSQL compatible, up to 5 times better than MySQL, storage scales in 10 GB increments to 64 TB, compute resources scale up to 32 CPUs and 244 GB mem, 2 copies of data in 3 AZ
- Scaling - storage is auto healing (looks for errors auto, repairs auto)
- Read Replicas- 3 types (aurora- 15, MySQL - 5, Postgre - 1, on top of DB replicas
  - Aurora better, faster, more, in region (MySQL is cross), **auto fail over**, less support
- Backups - always enabled, do not impact perf, can snapshot Aurora no impact, can share snaps
- Serverless - on demand, autoscaling config for Aurora - for unpredictable, intermittent, infreq workloads (pay per requests/ invocation so cheaper)
- Migrate to aurora - actions - create aurora read replica (in new AZ or not) . Go to write node - promote to standalone. Could have taken snapshot then created in aurora

ElastiCache
- Memcached vs Redis
  - Memcached- simple, scale horizontally (more instances), multi thread performance
  - Redis - has simple cache, horiz, adv data types, sort, publishing capab, multi AZ, backup & restore
- Elasticache improves performance (can also use Read replicas), Redis has backups and Multi AZ

DMS
- Way of migrating DBs (cloud or premise)
- Most basically - server in Cloud that replicates stuff from source endpoint to target endpoint, you can create target tables manually or use AWS schema conversion tool to help
- Types - homogenous (Oracle to Oracle) and hetero (MySQL to Aurora)
  - Sources - on premises, azure, RDS, S3
  - Targets - on premises, RDS, Redshift, Dynamo, S3, Document DB, ElasticSearch, Kinesis
- How it works (hetero only)- Schema Conversion Tool

Caching Strategies- more caching better perf, balances latency and accuracy
- Cloudfront
- API Gateway
- ElastiCache- Memcached and Redis
- DyanmoDB Accelerator - caches in DB

EMR - Elastic Map Reduce
- Cloud big data for processing vast amounts of data (less than half cost premis)- uses the cluster
- Each instance in cluster is node, has master node (manager), then core (runs tasks and stores data), then task (only tasks no storage) which r optional
- All communicate with each other
- Log data in master- can be archived in S3 so log files persist after termination (5 minute int) done at creation

Route 53

Thursday, August 27, 2020 5:22 PM

Review

- IPv4 going towards IPv6 (everything that connects to internet gets an IP)
- Top level domains (com,gov,edu,co.uk) and is the last word in the domain name (2nd level is 2nd last)
  - Controlled by int assigned numbers authority- domain registrars
  - Domain registrars make sure no duplicates- registered with InterNIC, Db is WhoIS
  - Default 50
  - Start of Authority (SOA)- info on name of server, admin of zone, current version of data file, num secs for Time to Live on resource records (default 48 hours- so 48 hours to move website IP)
    - Has DNS records- A - stands for Address- phonebook
    - CName - canonical name - send to different domain name (which will have an IP)- alias
    - Alias Record - Cname but can be used for naked domain name (no www), better for AWS services, can be used in zone APEX (Cname cant)
  - Name Server records - used by top level domain servers to direct traffic to DNS server (user -> Top Level Domain Server -> NS Record -> SOA)
- ELBs do not have predefined IPv4 address- cant use with A record (use with CNAME)
- MX record (mail), PTR (reverse A)

Policies

- Go to route 53, dns, record set and give it name, type, put IP address of EC2 or whatever target in value, set TTL (saves IP address)
  - Health checks (need to create -choose what to monitor (usually endpoint), IP, host name (domain name), path (usually index)) and checks if can successfully get to path - health checks need to be associated with records
- Remember restart EC2 means different IP address
- Simple Routing - only 1 record with 1+ IP addresses, if 2+ returns all in random order (1st is accepted)
- Weighted Routing- split traffic based on assigned weights- can use health checks
- Latency Based- route based on lowest network latency, needs regions
- Failover Rooting- when create active (primary), and passive (secondary), uses health check on primary
- Geolocation - choose where based on user location - e.g. language, country or continent specific
- GeoProximity - based on user and resource location
  - Needs traffic flow- has traffic policies - DNS types to rule with very complex rules and bias - expands or shrinks size of user region routed to resource
- MultiValue Answer - simple routing but can check health on each IP, auto fail over, returns multiple values, can be random but better resilliency

VPCs

Thursday, August 27, 2020 5:22 PM

Overview
- Virtual data center- logically isolated section of cloud - own network gateways, route tables, subnets, IP range
  - Configured for public and private subnets, lots of security
  - Hardware Virtual Private Network - between corporate datacenter and VPC
- How it works
  - VPC in region, access VPC through internet gateway or virtual private gateway
  - Gateways go to router, routers go to route tables, route tables go to network ACLS (like security groups but stateless- can do individual allow and deny rules)
  - NACLs go to a Subnet - instances and security groups
    - Private subnet cant access internet (if wanted to access from internet, go to public subnet then ssh)
    - Subnet IP ranges - 10 prefix, 172.16 prefix, 192.168 prefix
    - Subnet has to be in 1 AZ
- How to use
  - Launch instances in subnet, assign custom IP address ranges, configure route tables between subnets, create internet gateway and attach to VPC, more security, security groups, create Subnet ACLs
- Default VPC - user friendly, immediately deploy instances, all subnets have route to inter, each EC2 has public, private IP
- VPC Peering - connect VPC using direct network route using private IP, can connect to diff acct or same, between regions - star configuration - 1 central VPC and 4 outside
  - Not Transitive peering - when have to peer from out to central to out to talk to other VPC - means have to make connection from out to out

Create VPC - just need name and IP range
- When created, makes route table, Network ACL, security group
- Can create subnets - name, which VPC, AZ - private by default, 5 IP addresses used for aws things (first 4 and last)
- Can create public IPv4s for subnets in actions
- Need gateways - internet gateway, create with name then attach to VPC - one gateway per VPC
- Route table - for subnets to talk together, by default subnets on main route table, want this private and other table publ
  - Make public route table - name and VPC, after created edit routes (to make it public) - add rule to add 0.0.0.0/0 destination and the target will be internet gateway- add subnet to public route in subnet assoc
- To add EC2- when creating, configure on VPC network (not default) then put in public/private subnet
  - Security groups don't span VPCs
  - Ssh into public EC2 like normal
- To access private EC2, need inbound rule
  - Create security group - inbound rule to allow our public security group to access the private security group- source is web SG, do this for ICMP, HTTP, HTTPS, SSH, Aurora
  - Assign private EC2 to open SG with actions -> network -> change SG
  - SSH into public EC2 (should be able to access private ip cus new inbound rule) -
    - use **ping <privateIP>** to see whats up
    - Not sure how to get in… just use bastion

Nat Instances & Nat Gateways
- Network Address Translations - to allow our subnets to access internet without being public
- Nat Gateway- better (instance is an EC2)
  - Connects Private subnet to public subnet to internet gateway
  - Autoscale, no need to patch, no SG, auto public IP, remember update route tables
  - All in one AZ, more redundant than EC2
- Nat Instance
  - Need to disable source and dest checks since NAT acts as a gate not an actual source or dest
  - Now need to connect- VPC - main route to security group in route table - add 0.0.0.0 rule to Nat instance (now can go out to internet)
  - Must be in public subnet, have a
  - Amount of traffic NAT can handle depends on instance size
  - Can use autoscaling groups, multiple subnets

NACLs
- Security for allow/deny rules - done in ID increments 100
- Created VPCs deny everything by default, rules read in number order (so allows read before denies)
- Default ACL- VPC auto comes with, allows all traffic, has new subnets
- Need to add inbound and outbound rules manually
- Ephemeral - says port range, to continue talking after accessing with another port, for accessing internet in private subnet
- Subnet can be associated with one ACL at a time (one ACL many subnets)
- NACLs evaluated before security groups
- Stateless-need to make inbound in and outbound rules

Custom VPCs with ELBs
- Can create LB- usually app load balancer - can be ipv4 or 6, etc
- Need at least 2 public subnets (to load balance)

VPC Flow Logs
- Capture traffic in and out of your VPC - CloudWatch
- VPC level, subnet level, Network Interface Level
- In VPC- actions- create flow log
  - Type - accepter and rejected
  - Send to s3, CloudWatch, etc - needs new log group in CloudWatch
  - Make IAM role - for flowlog
- Flow logs for Peer VPCs must be in same acct
- Can't change configuration, can tag though
- Not all traffic monitored- not for DNS server, traffic from windows instance, from 169.254.169.254 (metadata), DHCP traffic

Bastion Host
- Special purpose computer to defend against attacks from **SSH**, whereas NAT is for internet traffic rules
- Cannot use Nat gateway as Bastion Host

Direct Connect
- Dedicated network connection from premises to AWS- better connection
- For high throughput of stable, reliable security
- To set up - create virtual interface, go to vpc console, then VPN conenctions, create customer gateway then virtual private gateway - attach to desired VPC, select VPN connections and create new VPN connect, select VPG and CG and set up newly created VPN on C G

Global Accelerator
- Speeds up with edge locations, amazon network
- Static IP addresses - receives users - GA gives u 2 but can bring your own
- Accelerator - directs traffic to optimal endpoint using AWS network - imp avail. and performance, each accel has 1+ listeners
  - DNS name - each GA has DNS, points to static IP address
- Network Zone - services static IP for accelerator from unique subnet - like AZ for networks - fault tolerant (2 IPv4 addresses)
- Listeners - processes inbound connects from clients to GAs, based on port and protocol (GA takes TC and UDP) - listeners have 1+ endpoints associated, traffic forwarded to endpoints in one of groups
  - Endpoint groups - 1+ endpoints, region specific, has target percentages for each endpoint (traffic dial)
- Endpoint - ELM, EC2, Elastic IP, can be internal or public, each can have weights
- To make, just needs name, type (ipv4 or 6), listeners (ports, protocol - TCP and UDP), add endpoint groups (regions), add endpoints and weights
- To delete- disable then delete

VPC Endpoint
- Enables private connect VPC to AWS over Amazon network- don't need internet
- Endpoints are Virtual Devices - horizontally scaled, redundant, highly available VPC components
- Interface Endpoints - elastic network interface with private IP address to allow use of AWS services (gateway to services)
- Gateway Endpoints - same for S3 and dynamoDB
- Need to have role that has S3FullAccess
- EC2 in private subnet with role
- In VPC, endpoints - create endpoint - needs a service - for which Gateway - using what root table - choose policy
  - Have to specify region if ssh in

PrivateLink- best for scale
- Connect VPCs (hundreds) - no internet or peer cus security and lots of peering relationships
- PrivateLink - best way to expose, requires Network Load Balancer on service VPC and Elastic Network Interface on customer VPC

Transit Gateway
- Simplify networks - hub for all your VPCs (thousands) with transitive peering and on premises data centers
- Works on hub and spoke model, on regional basis but goes across regions
- Can use route tables to limit, Direct connect/ other VPN connections, supports IP multicast…

VPN CloudHub
- Connect all users/sites (each with VPN connection) together
- Hub and spoke model,Low cost, over internet
- Uses customer gateway and CloudHub is encrypted

Network Costs
- Free traffic into VPC, same AZ using private IP - but gives single points of failure since in same AZ
- Costs for traversing internet (public IP), across regions

Configure VPN over Direct Connect
- Dedicated connection not encrypted becomes encrypted (using CG with VPN)
- (already created Direct Connection) choose connection, add Virtual interface - public, name, acct, VLAN & 2 router peer IP (supplied by AWS support), BGP ASN, public IP of VPN firewall
- VPC - Customer Gateway - name, static/dynamic routing IP (aws direct connect customer gateway IP)
- Create virtual private gateway there (name)
- Attach this created VPG to your VPC, go to VPN connections- create new using name, selected VPG, selected CG, choose static IP addresses

HA

Thursday, August 27, 2020 5:22 PM

ELB
- Device to balance network load
- App Load balancer - suited for HTTP and HTTPS, app level and intelligent, can see html
- Network Load Balancer- TCP and extreme performance, at connection level
- Classic - legacy- layer 7 some, http/https, x-forwarded and sticky sessions- for real basic
    - If stops, 504 error - timeout - can be at web server or DB level
- X-forwarded - user goes to ELB goes to EC2, X-forwarded for gives the users IP address

Lab
- Need EC2 instances in different Azs
- LB created within ec2, choose type, name and VPC choose (can choose subnets), choose traffic, security group, health check - tests health checks by response timeout, threshold for healthy and unhealth, interval between tests
    - Can do cross zone LB (what we want usually), can make some metrics with classic
    - Gives DNS for ELB (not IP), auto failover
- Target group - group of EC2 for users based on their IPs - also has health checks (for file)- add targets to it
- ALB - specify AZs, can configure target groups - much more complex rules to ELBs

Advanced
- Sticky sessions - bind a user's session to one EC2 instance - good for saving info (can be done with ALB but only saves to target group not EC2)- can disable sticky sessions to allow other EC2s to be used
- Cross Zone LB - when LBs can send traffic to diff AZ - great for when unequal instances in AZs
- Path Patterns - based on url path, can forward requests to diff AZ/ diff instances (app load balancer)

AutoScaling
- Groups - logical component (put ec2 in)
- Config templates- each group uses templates to create new instances
- Scaling options- ways to scale (based on condition - dynamic or schedule)
    - Maintain at all times - spec num instances always- done with periodic health checks
    - Scale manually - specify desired capacity
    - Schedule - auto as function date and time
    - On demand - most popular - scaling policies control num instances - response to conditions
    - Predictive scaling - with AWS auto scaling combines predictive and reactive
- Needs launch config (in EC2) with role, SG, then create ASG - which LC, network, subnet (can create LB), add policies (type, num instances to scale between)
    - If terminate instances, will auto create to stay at min desired
- HA architecture- Plan for failure- lots of redundancy, health checks, auto failover- multi AZs, scaling up/vertical is increase size, different S3 (single AZ sucks)
    - In CLI - **aws s3 cp --recursive /var/www/html/wp-content/uploads s3://<bucketname>** recursive copy
    - To .htaccess get Domain name for cloudfront and put in rewrite rule of the .htaccess file
    - Also change in httpd - in /etc/httpd/conf go to httpd.conf and change line from AllowOverride none to all
    - Lastly need to update bucket policy

Beanstalk
- For beginners - under compute, give name, platform (php, etc), sample app or code, creates security groups, buckets, etc
- Can add EC2, LBs, capacity configure- u simply upload app and EB does provisioning, load balancing, scaling, health monitor

Bastion Hosts
- Make HA - use 2 bastions with Network Load Balancer or use autoscaling group
- Network Load Balancer with Bastion
- Autoscaling group will provision bastion if first fails- but its slower than first cus has to detect failure and provision

On Premises Strats
- DMS - database migr service - migrate from primary env to AWS
    - Supports homogenous migr and heterogenous
- SMS - incremental repl of on premise to AWS - for backup, multi-site
- AWS App Discovery - helps customers plan migr service - download to server and will build map- downloads encrypted, use to make TCO
    - With migration hub tracks progress
- VM Export/Import - existing apps to EC2, used to create DR strategy or export VMs to on premise
- Amazon Linux 2 as ISO - works with lots of VM providers

Apps

Thursday, August 27, 2020 5:22 PM

SQS
- Oldest service, web service access to message queue between app components (ex. From bucket to lambda, sends job to SQS and EC2 fleet pulls from SQS for jobs)
  - Wont lose data if EC2 goes down, 256 kb of text soft limit
  - Allows decouple components of app so they run independently
- Standard- default, nearly unlimited, possible out of order, message delivered possibly twice
- FIFO - compliments, exactly once processing, first in first out - limited to 300 transactions per second (slower)
- Pull based- have to have EC2 pulling
- In queue 1 min to 14 days, default 4 days
- Visibility timeout - a job is in queue but inv when being processed, timeout makes it visible again so new ec2 can try - this allows for duplicates - max of 12 hours
- Long polling - retrieve messages from SQS only if there are messages (short returns immediately), saves money

Simple WorkFlow Service
- Web service to coordinate work across app components with human beings and processors
- Retention up to 1 year, task oriented API (not message based), ensures task assigned once, keeps track all events in app (SQS u need to implement)
- Domain- collection related workflows
- Actors - workflow starters (app that initiates), deciders (control flow of activity), activity workers (carry out activity)

SNS
- Web service to set up, operate, send notifs from cloud (push notifications, SMS, email, http endpoint)
- Group recipients by topic (people subscribe to topic), can handle diff OS subscribers in a topic
- All messages stored redundantly across multiple AZs
- Inst push based delivery, pay as u go, simple APIs, flexible, easy console interface

Elastic Transcoder
- Media transcoder in cloud - from media source to popular output formats
- Pay based on minutes and resolution

API Gateway
- Publish, maintain, manage APIs at scale - acts as front door for apps to access data, directs traffic (usually to Lambda)
- Can expose https endpoints, **serverlessly** connect to Lambda and DynamoDB, distribute API calls to targets, low cost, **scales** effortlessly, track usage by API key, protect against floods (throttle), connect to CloudWatch, have mult versions
- Define API, define resources (select HTTP methods, set security, set target, set requests and transformations)
- API Caching- improves latency for TTL, saves responses to users
- Same origin policy - one page can talk with other page only if pages have same domain names, enforced by web browsers - can hurt for aws diff domain names want info
  - Cors- cross origin resource sharing- allows restricted resources to be accessed between diff domain names
  - Works with **browser** making URL call, server says if can access or cant (if error cannot be read, need to enable CORS on API Gateway)

Kinesis
- Streaming data - kb data continuously sent in from thousands data sources- ex. Purchase from online stores, stock prices, , game data, social network data
- Kinesis- platform for your streaming data
  - Streams- data producers stream to kinesis (24hr to 7 day retention), stored in shards (buckets), ec2 instances (consumers) do analysis on these, store that analysis (redshift, s3 etc)
    - Shards - 5 transactions or 2MB per second read, 1k or 1MB per second writes
    - Data capacity depends on num shards
  - Firehouse - producers give info, analyzed (no store) maybe with lambda, stored
  - Analytics - works with both - analyzes data on fly and saves it

Web Identity Federation/ Cognito
- Gives users access to AWS by using web identity provider - such as Amazon Cognito
- Cognito - signup or in to apps, access for guest users, acts as identity broker between app and web ID (facebook, google, etc), synchronizes on multiple devises, rec for mobile apps for AWS
  - After facebook/google sign on, provides temporary credentials which map to IAM role allowing access to certain resources (doesn't store credentials)
  - User pools - user directories (can sign in directly), successful gives JSON Web Token (user based)
  - Identity Pools - temp AWS credentials to access AWS services (resource based)
  - Sync - push sync (SNS) to change user info across all devices

Event Processing Pattern
- Event driven architecture - such as the Pub/Sub model (SNS) where publishers create messages to be received by all subscribers
- Dead - Letter -Queue - used by SNS - failed delivery, SQS - message over maxRecieveCount (ignored), Lambda- exec error (tries 2)
- Fanout Pattern - used to make more available publish to sqs connection (**use SNS** between), then add more queues
- S3 event notifications- can make it so S3 sends message to SQS/SNS/Lambda for processing, and can add rules on it (only send png files), can enable versioning for better delivery
  - Object created, removed, restored (glacier), replication

Security

Thursday, August 27, 2020 5:22 PM

Overview
- We can use NACLs (block IP), host based firewall (software), ALB (connects SGs) , NLB (visible client IP all throughout), WAF (common injections, public web), Cloudfront can use WAF (CF doesn't share client IP)

KMS
- Regional service, manages keys, ideal for S3, DB passwords, up to 4 KB in size, pay per API call, connects to CloudTrail, FIPS 140-2 Level 2 is us gov cert (show handle tampering),
- CMKs
    - AWS Managed - AWS uses
    - AWS Owned - AWS service owns and manages across many accts
    - Customer Managed - allows rotation, u control key policies
- Symmetric - same key for encry and decryp , AAES 256, never leaves AWS unencrypted, AWS services use this, gen data keys, data keys, import own key material
- Asymmetric - math related, use RSA and Elliptic Curve Cryptography, download public key to use outside AWS
- To move keys, have to unencrypt, move to other region, encrypt with CMK of new region
- **Aws kms create-alias --target-key-id <target key id> --alias-name alias/<alias>**
- Data Encrypted Key- for more than 4KB of data to encrypt
- Use envelope encryption for faster, bigger encryption

CloudHSM
- Dedicated Hardware Security Modules - when own keys need validated control - FIPS Level 3,
- Manage your own keys, single tenant, no access to AWS managed component, runs within VPC, multi AZ, No AWS APIs - PKCS#11, Java Crypto Extensions, Microsoft CryptoNG

Systems Manager Parameter Store
- Parameter Store - component Systems Manager - to fix the problem of leakable keys
    - Secure serverless storage for configuration and secrets (passwords, license codes, api codes, etc)
    - Can be plaintext or encrypted
    - Can store parameters in hierarchies or track versions, set TTL
        - Can build tree structures - data retrieval in groups (up to 15 levels) by path
- How to create lambda function using Parameter store in AWS:
    - Create lambda function
        - First need execution role - role with JSON policy allows permissions for lambda function
        - Lambda boto3 is AWS SDK for python, needs name, type, execution role - sometimes has environmental variables (e.g. ENV has name environment)
    - Need to encrypt parameter values with CMK - KMS
        - Has admin (person giving) and user (role)
    - Go to parameter store to encrypt values
        - If want these retrieved in lambda function, put in path (e.g. prod/acg/…)
        - Only SecureString is encrypted parameter
    - Now test lambda - create test event with name, can leave code as default

Secrets Manager
- Similar to Parameter Store - param store comes with no addition cost (per 10000 API calls), secrets manager auto rotates secrets (can apply to RDS), can use lambda to rotate, can generate random secrets, can be shared accts

AWS Shield
- Against DDoS attacks,
- Shield standard - auto enabled, no cost, common layer 3 and 4 attacks (UDP packets floods or TCP), reflection attacks (UDP with spoofed origins)
- Shield Advanced - 3k a month per AWS org, enhanced protection for EC2, ELB, CloudFront, Global Accelerator, Route 53 - 24/7 access to DDoS Response Team, DDoS cost protection (insurance)

WAF
- Monitors HTTP requests to CloudFront, ALB, API Gateway
- Control content access- configure filtering rules based on IP address, URLs, Query injection
    - Returns http 403 forbidden code
    - Can allow all but some, block all but some, count requests that match properties - such as origin IP, origin country, request size, values in URLs, strings in regexpression patterns, block SQL injection, XSS attacks
- Firewall manager - configure and manage firewall rules across AWS org
    - Deploy WAF rules (ALB, API Gateway, CloudFront)
    - AWS Shield Advanced - ALB, ELB Classic, Elastic IPs, Cloudfront
    - Configure SGs for EC2 and ENIs

Serverless

Thursday, August 27, 2020 5:23 PM

Lambda
- The ultimate abstraction - aws does datacenters, hardware, assembly code, high level language, OS, app layer/API
- Use as event driven compute (e.g. change s3 bucket or dynamo table) or compute service (http request using API)
- Separate calls are **independent** (scales with more lambdas, out)
- Traditional - require on VMs and OS, serverless - uses API gateway on lambda (scales very well)
- Supports node, java, python, java, c#, go, powershell
- Priced on num requests and duration
- No servers/viruses/crashing, super cheap and scales well
- X-ray allows debug lambda, can do things globally

First Serverless website
- Create function- needs name, type, exec role (microservices for DynamoDB webserver)
  - Now actually code function
  - Triggers
API Gateway	Alexa	AWS IoT	ALB	CloudWatch Events/Logs	CodeCommit
Cognito	Dynamo	Kinesis	S3	SNS	SQS
  - Add API Gateway trigger (serverless) - auto creates method taking in client traffic and returning, type any
    - Actions- create method - type (get, set etc.) , yes lambda proxy - GET allows retrieve
    - Then deploy API
    - Too see result of lambda function, got to GET and invoke URL
  - Make S3 bucket to host website, hold index and html files, and can write in your index.html an html.open("GET", <URL>) to retrieve info on your lambda created page

Alexa Skill
- Skills made of service (e.g. lambda) and interface (name, schema, type)
- Create bucket (make public)
- Polly - pick voice, write message, synthesize to S3 (save as mp3)- should be saved in there now
- Lambda - alexa triggers in certain regions (N Va, Sydney)
  - Serverless Application Repo - AWS published free apps - nodejs fact skill
- In Amazon Developer - in alexa developer console - create skill - name and default, fact skill
  - Change name in invocation
  - Endpoint - put in default region the ARN of the lambda function
  - Intents - GetNewFactIntent - sample utterances to ask alexa "ask cloud facts… "

Server App Model
- Cloudformation extension for serverless - define functions, APIs, tables- supports anything cloudformation does
- Can run locally with docker, can package and deploy using CodeDeploy
- Made in templates- applies properties to all functions, creates resources, output
  - Might need to choose runtime, sam deploy guided - name, region, role, authorize

ECS
- Container - package contains app, libraries, runtime, tools to run app
  - Has engines like Docker, isolated, less overhead, faster start, portable, consistent environment
- ECS manages containers, creates clusters to manage fleets or fargate instances
  - Schedules containers based on rules for CPU mem optimization, deploy update, rollback, free, has VPCs, SGs, ELB, EBS, Cloudtrail and CloudWatch
- Cluster- collection of ECS (EC2 or Fargate)
  - Service - allows task def to be scaled by adding tasks (has max and min)
    - Task Definition - defines apps (can contain mult containers)
      - Container Def - indiv containers task will use, controls CPU and mem alloc
      - Task- single run copy defined by task def (one working copy)
- Registry - storage for container images (Elastic Container Registry or Docker Hub)
- Fargate - serverless compute engines, per app service, uses ECS and EKS, isolated, secure
  - Choose EC2 for strict compliance, broader customization, GPUs
- EKS - Kubernetes - open source deploy containerized apps - same toolset in cloud and on premises
  - Containers grouped in pods (kinda like task)
  - Good for kubernetes, want to migrate to AWS
- ECR- for container images, works with ECS, EKS, highly available, has IAM, pay for storage and data transfer
- ECS and ELB - supports all ELB, by EC2 and Fargate
  - ALB allows Dynamic host port mapping (multi p tasks per container service), path based routing, priority rules (allow multi services run on same ALB), reccomended
- ECS security - instance roles - all tasks on EC2 get permission and task role - permissions per task
- Auto creates subnets, vpcs, ecd2 resources (in cluster, task def, service), cloudwatch log groups,
  - Uses cloudformation, Fargate - gets public and private IP
  - Auto creates tasks to desired num tasks- uses container image to create new instances
  - Use LB so you can have one DNS for all

Summary

Wednesday, August 26, 2020 5:51 PM

IAM

- IAM things- Users, Roles (on resources), Groups (have permissions), Policies (in JSON)
- IAM is universal
- Root acct - acct started, email - has complete admin access, make IAM
- New users have no permissions (cant access aws), have access key &secret access key when created (appear once)
- Power User Access - all AWS services but cannot manage groups and users within IAM
- Directory Service - Active Directory (hierarchical DB), Connecting AWS to on premises AD, AWS Managed Microsoft AD (run AD from AWS), AD trust (AD from AWS to on premises), can sign on from SSO to EC2
    - Simple AD- no trusts, needs AD Connector- on premises AS to AWS, Cloud Directory - devs to work with cloud heirarchies and cognito- social media login
- ARN, structure JSON Policy - effect/action/resource, Identity vs resource policy, deny > allow, aws managed policies unable edit, permissions boundary- max permissions for an identity
- RAM- resource sharing between accts, only so many, SSO - central manage access with existing identities and SAML

S3

- Universal namespace, object based, 0 to 5 TB for files, unlimited storage, 100 buckets by default
- Not good for OS or db, when upload HTTP 200 code, use multipart upload for big files
- All new buckets private by default - have bucket policies (for folder) and ACLs (for specific files)
    - Can be configured to create access logs (log requests sent to another bucket possibly in another acct)
- Fundamentals- Key (name), Value (data), Version ID, Metadata, Sub-resources (ACLs and Torrent
- Remember eventual consistency for updates and deletes but immediate with creates
- Classes
    - Standard, IA - cheaper, for old things, One Zone IA - lowest availability, cheaper than intelligent, Intelligent - for unpredictable scaling, Glacier, Glacier Archive
- Encryption in transit - SSL and TLS (over HTTPS)
- Encryption at Rest
    - Server Side - 3 types - S3 managed, AWS managed KMS, Customer provided
    - Client Side- you do it all then upload to server
- Orgs best practices
    - MFA on root, strong complex passwords, paying acct should only be for billing (no resources), use Service Control Policies on Org unit of individual accounts
- Sharing buckets
    - Bucket Policies (& IAM) - programmatic access
    - Bucket ACLs (& IAM) - programmatic access
    - IAM roles - programmatic access and console access (switches into role from user acct)
- Cross region replication - must have versioning, files in existing bucket not copied auto, delete markers and deletes are not replicated, can be done in same region
- Lifecycle Policy - auto move files between tiers - can be done on current and prev versions
- Edge Locations
    - Transfer acceleration - upload faster
    - Cloudfront - distribute content to be cached, has origin and list of Els, ditribution is name of CDN
        - Web Distribution - for websites , RTMP for Media Streaming
        - ELs can be written to, cached items have TTL, can be cleared but costs money - done by invalidating
- Snowball - big disk for import/export to S3
- Storage Gateway
    - File Gateway - for flat files, store directly on S3
    - Volume Gateway- **Stored**- entire dataset on site and S3, **Cached**- dataset on S3 and frequent data is on site
    - Gateway Virtual Tape Library- for backup tapes
- Athena - interactive query service using SQL for data on S3
- Macie - interactive query service using SQL and AI to find PII, has alerts

EC2

- 4 types - spot terminated not charged unless u terminate
- EBS - virtual HDD on cloud, terminate protection off by default, root terminated with instance, can be encrypted
    - 5 types (2 SSD and 3 HDD), gen purpose, IOPS (DBs), throughput opt (data warehouse), cold (file servers)
    - Snapshots exist on S3, instant copy of volume, incremental - encrypted automatically, can only share if unencrypted
    - Volumes in same EC2 as EC2 instance, can change size/type on fly
    - Can create AMIs from either
    - Migration - take snapshot, create AMI, copy AMI launch in new AZ, or copy into new region then launch
    - For unencrypted root, snapshot then copy as encrypted then create AMI from encrypted snap, launch from AMI
- Security Groups - all inbound traffic blocked by default, outbound is allowed, changes immediate
    - Stateful (open port, inbound and outbound), cant block specific Ips
- Instance Store- ephemeral storage - cannot be stopped (if you do, lose data), can be rebooted though
- Enhanced Networking
    - ENI - Elastic Network Interface - basic networking
    - ENA - Enhanced Network - better performance 1 Gps or 100 Gps
    - EFA - Elastic Fabric Adapter - HPC or ML or OS by-pass
- CloudWatch - performance, detailed monitor for 1 min, create alarms/dashboard, events (respond to state changes), logs
- Roles - much more secure, easier to manage, assign to EC2 after created, universal
- Metadata and user data using curl and address of EC2
- EFS - NFSv4, scale to petabytes, lots of concurrent NFS connections, data stored within regions- for Linux
- FSx for Windows - storage for Microsoft apps
- FSx for Lustre - HPC, financial modeling, uses S3
- Placement Groups - Cluster (in AZ, low latency), Spread (in region),Partition (HDFS,Hbase, Cassandra), only certain types, cant merge, cant move existing instance into PG, rec same
- WAF and ACLs- block Ips

DBs

- RDS - OLTP in 6 flavors, DynamoDB (NoSQL), Redshift (OLAP), backup effects latency
- Elasticache - for better performance, Memcached (simple) & Redis (better)
- RDS on VMs, cant ssh, cant access OS since Amazon in charge, is SERVERLESS (aurora serverless is not), DynamoDB is less
- Backups - automated and RDS Snapshots
- Read Replicas - can be Multi AZ, for perf, must have backups, can be different regions, all 5 but SQL server, can be master (sep)
- Reboot RDS to force a failover from one AZ, Multi AZ is for DR
- Encryption at rest for all, done using KMS, once encrypted so are snapshots, replicas
- Dynamo - stored on SSD, spread across 3 AZs, eventual consistency by default (1 sec), strongly consist (<1 sec)
- Redshift - for business intell, only in 1 AZ currently, enabled 1 day retention backups (can be 35), 3 copies, can repl snaps to S3
- Aurora - 2 copies in 3 AZ, can share snaps, 3 types replicas (Aurora has auto failover), backups on by default, serverless for infr.
- Elasticache - increase DB/ Web App perf, Redis has multi AZ & backups

DNS

- ELBs don't have IPv4 addresses- need DNS name
- Alias vs Cname (Alias can do naked stuff), alias better
- Record types - SOA, A, CName, NS, MX, PTR
- Policies - Simple, Weighted, Latency, Failover, Geolo, GeoProx ,Multiple Answer
    - Health checks - on a record set - removed if fails

VPCs

- VPC - logical data center in AWS, has Virtual Private Gateways, Route tables, NACLs, Subnets, Security Groups, no transitive peering
    - 1 Subnet is in 1 AZ, Security groups are stateful (have to do in and outbound), NACLs are Stateless
    - VPC gets default Route Table, NACL, SG no default subnets of internet gateway
    - Amazon reserves 5 IP within subnets, 1 internet gateway per VPC, SGs don't span VPCs
- Nat Instances - when creating disable source/ dest check, must be in public subnet and route from private subnet to NAT, always behind a SG- traffic support depends on size, can have Autoscaling, multiple subnets, auto failover (script)
- NAT gateway - instance connects to gateway (not behind SG), no need to patch, has auto public IP, redundant in AZ
    - NAT gateway alone has bad availability, need more for better
- Network ACLs - default network ACL allows all in and out, new ACLs deny all in and out by default, new subnets on default ACL, can block IPs with ACLs not SGs - work on Subnet level
    - ACLs have many subnets but subnet has one ACL
    - Contain a numbered list of rules, evaluated in order
    - Have sep in and out rules- stateless
- ELBs- Min 2 public subnets to use
- Flow Logs - cannot have flow logs across accounts, cannot reconfigure, can tag- not all traffic monitored (DNS contact, Winds Instance traffic, traffic for metadata, DHCP traffic, to reserved IP router)
- Bastion - can ssh in to private subnet from internet to public subnet to private
    - Bastion for ssh, NAT for provide internet traffic, cant use NAT as bastion
- Direct Connect - dedicated network to AWS - high throughput, stable, reliable
    - Steps - create VI, create customer gateway, virtual private gateway, attach VPG to desired VPC, select VPN connections and create new one, select VPG, CG then setup VPN on customer gateway
- Global Accelerator - create accelerators to improve performance - uses AWS edge and backbone network, give 2 static IP addresses, has endpoint groups with traffic dials or weights
- VPC endpoint - connect to AWS without internet, on AWS network- 2 types - interface (lots), gateway (S3, DynamoDB)
- PrivateLink - peering VPCs to lots and lots of VPCS, requires NLB on service VPC and ENI on customer VPC
- Transit Gateway - simplifies network topology - works on hub and spoke model, allows transitive peering between VPCs, works on regional basis but can be on multi regions, can go across Accts with Resource Access Manager, can use route tables, works with Direct Connect and IP multicast
- VPN Cloudhub - to simplify VPN connections from site to site, all traffic over internet but encrypted
- Network Costs - use private IPs not public, group EC2 in same AZ

HA

- 3 LBs - app, network, classic, 504 for timeout, is it web server or db?
- Classic - IPv4 of user? Use X forwarded
- ELB are in service or out of service, have health checks, LBs have their own DNS names (no IPs)
    - Sticky sessions - keep on EC2, cross zone LB - load balance across Azs, Path patterns - direct to EC2 per URL
- Cloud Formation - scripting cloud environment, QuickStart bunch of CloudFormation templates already built in
- EB - provision your app without worrying about details, not as powerful
- Bastion hosts - either have Network Load Balancer or AutoScale
- DMS, SMS, ADS, VM Import, Download Linux 2

Apps

- SQS - way to decouple, pull based, 256 kb insize, 1 min to 14 days default 4 days, standard and fifo (has order and 1 send)
    - Visibility time out - increase for longer jobs and duplicate messages, up to 12 hours, long polling saves money
- SWF - retention period up to 14 days, messages last up to 1 year, task oriented API **(humans)** no duplicates, tracks tasks
    - Actors - flow starters (initiate), deciders (what work to do), activity workers (do work)
- SNS- instant, push based, simple, easy, cheap, flexible
- Elastic Transcoder - transcodes media
- API Gateway - front door, can cache, low cost, auto scale, log results, prevent attacks, enable CORS for XS commun (clients)
- Kinesis - streams (data pers- shards), firehouse (in real time), analytics (helps analyze)
- Cognito - web identity providers federation, gives IAM roles for credentials, user pool (user based), identity (resource based)

Serverless

- S3, API Gateway, DynamoDB, Lambda
- Serverless- API Gateway, Lambda, S3 - no EC2 and NLB
- Lambda scales out, independent functions, can trigger other lambda functions, X-Ray to debug serverless app, can do things globally (e.g. S3)
- Triggers -
-

API Gateway	Alexa	AWS IoT	ALB	CloudWatch Events/Logs	CodeCommit
Cognito	Dynamo	Kinesis	Lex	S3	SNS/ SQS

Quiz Notes

Wednesday, September 2, 2020 11:39 AM

AutoScaling
- vCPU based on demand instance limit is diff per region
- Metric backlog per instance to indicate, launch templates are used with dedicated hosts, oldest config deleted
- Has cooldowns (few minutes)
- Step Scaling - scaling metrics and thresholds trigger scaling
- Target Tracking - based on specific metric
- Simple - based on single scaling adjustment

Databases- multi AZ
- RDS
  - DB auth only for **mySQL and PostSQL**
  - Slower latency during backups of RDS
  - Implement multi AZ for high availability & min downtime- synchronous data repl
    - With multi az cant read/write to other instance, only there for failure
  - Stored on EBS, IOPS storage better in out
  - Increase IOPS with increased volumes
  - SQL doesn't have multi az failover
  - Don't use myISAM
  - Normal Monitoring- DB connections, freeable memory, cpu util
  - Enhanced monitoring - get metrics CPU%, mem%, in processes RDS child processes, RDS process, OS processes
- Dynamo
  - Chargeable DynamoDB for storage of data and read/write
  - Autoscale not by default, API Gateway with Lambda are nasty
  - Db Stream - ?????
  - DAX- great for games, in mem cache, lowish cost
  - Limit 400KB for value and name
  - Data stored on SSD
  - Better for stateless
  - Use EMR to import/export
- Redshift
  - Spectrum - queries against exabytes
  - Cluster - has no flow logs by default, keeps track of data in redshift
  - Standard vpc routing - SGs, NACLs, VPC endpoints, policies, internet gateways, DNS
  - For cluster, can use enhanced vpc routing for copy/upload
  - Audit logging - info on connection like user activities and queries
- Elasticache
  - Redis - AUTH command, improve data security with password (involves --transit and encryption-enabled params)
  - Milliseconds, not microseconds- DAX tho
  - Distributed session
- Aurora - cluster DB instances,
  - Custom endpoint - load balancing DB connections
  - Cluster/instance endpoint - for connecting to DB
  - Failover - (no replica on) create new instance in same AZ, then new AW, (replica on) new cname

EC2
- AWS Run Command - at scale config
- EBS snapshots don't affect performance
  - EBS volume can persist after instance, can modify volume type, size, IOPS without service interup, auto repl in AZ, 3 9s SLA, AES 256, can only be on one instance in same AZ, stored in S3
  - Snapshots auto encrypted, all data between instance and volume is encrypted
- On stop/start, can change underlying host and instance store devices lost, no charge for 1 eIP
- Billed for hibernate, not billed for pending
- Bastion host vs nat gateway, use key pairs to log in (not access keys)
- Hot/Cold/Warm attach for Elastic Network Interface when running/launched/stopped
- Reserved - Auto convert from reserved to on demand after expires, terminated reserves still billed
- Failovers take a couple minutes
- Implement Multi AZ
- LBs
  - App - path based routing, host based, request based, http, URLs, lambda, containers
  - Classic - sticky sessions
  - In one region
- Key Pairs are region specific
- PGs
  - Reduce latency, high throughput
  - Spread PG - only 7 per AZ
  - Insuff capacity error - just restart PG and try again, might move hardware too
- Health based on memory usage
- Spread on AZs for better performance
- Soft limit for **each region** (20 res)
- AMI copy doesn't copy launch permissions, tags, S3 bucket permissions
- For EC2 IP Address that doesn't change, launch EC2 in VPC

IAM
- Owner is person created AWS acct
- Ssh private key 0777 is unprotected private key
- IAM user to see permissions, need to create credentials (role assumes creds)
- To switch ownership - do cross acct copy or change ACL (hard to manage) so put policy on bucket and user
- DB Authentication - token instead of password, 15 min lifetime, central manager, specific to Ec2
- Parameter Store - cross region but same acct - holds parameters, KMS, used by ecs (needs IAM role)

Network
- A records go to alias names (DNS names)
- API Gateway enables restful APIs, WebSocket APIs for serverless, pay for calls and amount transferred out
- Egress only Internet Gateway- hor scale, only in from internet **for IPv6**
- Port 3389 is Remote Desktop - TCP and UDP
- Route Origin Authorization - allows map IP range to AWS
- Signed Cookies - for access to appl, signed URLs - access to files and RMTP
- Throttle - API gateway can have throttle limits which handle bursts of traffic (temp much better throughput)
- Active - Active failover is for failover and want resources to be active most of time, set up weighted routing too
- Multicast - 1 to many distrib, works with virtual overlay network
- AWS to site VPC- need Internet routable IP address on customer gateway

S3- 3500 adds, 5500 retrieves
- Can make events for S3 Obj Created and S3 obj Removed:Delete
- Encryption types - SSE S3, C, KMS, client library
  - SSE- server side with AES 256
- Encrypt SSL connection by downloading amazon **RDS root CA** cert, import to server
- Event notif destin - send to lambda or SQS, SNS
- Multipart Upload - better throughput, quick recovery, pause/resume uploads, begin upload without full file
- Storage Gateway- mounted file system on S3
- Gateway- Cached is low latency (amount increases with size local disk), File Gateway does same
- Glacier - provisioned capacity - ensures capacity for expedited retrievals is available
  - Glacier Select - SQL on glacier
- Immed cons for puts, eventual for updates and cons- if getting old data probs from parallel requests
- S3 select needs bucket name and object key
- Single put up to 5 GB, multipart up to 5 TB
- Must wait 30 days to transition object classes to IA
- Server Access Logging- get allll the info

Security
- SGs - instance level, allows not denies, sep in and out rules, need SGs and NACLs to be working, all outbound traffic allowed by default
- NACLs - has uneditable * rule, allows denies
- S3, EFS, EBS have native encryption at rest, SSL is for encryption in transit
- For SSL across domains use ALB, bind their certificates - Server name identification chooses which cert for user
  - SSL on multiple domains - gen cert and associate with web distr, enable support for SNI (only ALB)
- Can use Subj Alternative Name for each domain but still have to reauthenticate

Services
- Amazon Connect - cloud call center
- Cloudformation - uses YAML and JSON, version control, pick services, creation policy to test health before launching (cfn - signal for success), DependsOn (one resource after another), UpdatePolicy (existing)
- CloudFront - use Lambda@edge and origin failover (no 504 errors), use versioning to control what is returned, **no Dynamo**
- CloudTrail - default has **encrypted** log files (in S3)
- CloudWatch - doesn't have mem utilization, works with EC2 manipulation but not Route53, can manipulate EC2
- CodeDeploy - deploys app to EC2, lambda, EC2 based on configuration type (how to update)
  - With Lambda - Canary (2 increments, some traffic then all), Linear (equal incr), All-at-once, Blue Green (2 prod env)
- Cognito - temporary AWS credentials use Cognito ID (auth for access)
- Config- password policy and all resource configs
- Data Lifecycle Manager - auto create EBS snapshots, delete outdated, retain
- DataSync- from on premise to AWS Storage, very fast, for larger data sets
- Directory Service - use AWS with directories,
  - Simple AD is subset- manage groups, create policies, secure connect,
  - Service AD Connector - good for connecting existing active directories to aws, use with IAM roles
- Disaster Rec (Architecture)- Pilot light (min version always running),
- EFS is for EC2 **not S3**, good for app storage, scalable, high throughput POSIX, not high powered
- Elasticache - runs pub/sub, in memory, sorted sets, not rds
- EMR- for log analysis, can access OS
- FSx - file system
  - Lustre - high perf, parallel file system, multiple networks, max performance, works with S3
  - Windows - windows system, SMB protocol, NTFS, active directory
- Glue - Expand Load Transform data
- Green/blue deploy - doesn't work with codecommit and is more expensive than in place upgrade
- HSM - gen and encrypt keys
- Import/Export - physical storage to AWS
- IoT- connect devices easily to cloud
- MQ - set up your message broker
- Opworks- Chef, Puppet, creates stacks
- Proactive Cycle Scaling - predict/ respond for fixed intervals
- RAM- cross acct resource access (with 1 acct just use tags/ policies)
- Snowball Edge - snowball with storage, compute
- Storage Gateway - for small data and caching low latency
- SQS- Wait time seconds- ec2 waits for more message to come in, multi AZ, no priority
- Step Functions - automate serverless workflows for recurring tasks
- STS - sec token service, temp cred to AWS (auth for use), used by SSO, not for RDS
- Transit Gateway- connect vpc and in premise network in central hub
- VPN- uses internet, highly avail, IPSec, TLS, on prem to cloud
- WorkDocs - like google docs

VPCs - flow logs
- Active Directory - microsoft, has a federation service, SAML 2.0,
- Public subnet doesn't nec make public instances, by default user VPC doesn't auto assign public IPv4
- Public subnets have to have internet gateways
- Instances in subnets in VPC can communicate across Azs
- IPs don't change on restart
- Flow logs can be for VPC, subnets, network interface
- Egress- only - to allow IPv6 traffic within VPC to go to internet and deny internet based traffic to connect to VPC
- Nats are resizable
- Can change dedicated to default hosting with AWS CLI, SDK, API
- Subnets associated with main route table (has CIDR block)
- VPC Endpoints - private connect VPC to AWS without internet
- VPN Cloudhub - provide secure communication between remote sites
- Cant have transitive peering or edge to edge routing via gateway () , or overlapping CIDR blocks

Review

Monday, September 28, 2020 2:08 PM

Field Programmable Gate Arrays - hcp - parallel
X Forwarded For - ELB forwards ip address to ec2
Port 3306 - mySQL

Serverless

Tuesday, September 29, 2020 1:09 PM

Lambda

- Priced on gb-seconds (duration)
- Versioning
  - 2 types of ARN - Qualified ARN - with version suffix and Unqualified ARN (no version suffix)
  - Aliases- points to version ARN (easier to change mapping than change working version)
  - Qualified uses $Latest, versions are immutable, can split traffic with alias (not with $latest)
- Step Functions- visualize and test serverless apps, good for visualizing, uses cloudformation
  - Auto trigger/track, retries when errors, for faster debugging
  - Using functions as steps in a process
- Concurrent Executions Limit
  - Limit 1000 executions per region per acct per second - error of tooManyRequestsException, 429 status
  - Reserved concurrency - set num executions always available (acts as a limit)
- VPCs
  - Enabling lambda to access VPC resources, not by default
  - Need allow function to connect to private subnet: private subnet ID, SG id and config, then lambda uses ENIs with ur cidr range to connect
  - Aws lambda update-function-configuration --function-name myfunc --vpc-config SubnetID=___,SecurityGroupIds=___
  - In console- author from scratch, go to network, choose VPC, choose subnets, SGs (make sure lambda has role)

X Ray

- Collects data about requests to app, response, calls from app to aws resources, microservices, web APIs
- Has SDK, sends JSON to Xray Daemon, Xray API, creates visual
- Sdk - has interceptors to trace HTTP requests, client handlers for AWS SDK Clients, HTTP client for HTTP web services
- Integrates ELB, Lambda, EC2, API Gateway, Beanstalk
- Uses Java, GO , Node, Ruby, Python, .Net
- Can work on EC2, ECS, your data center, Lambda- AWS SDK is on system, Xray Daemon is on system, sends to queue
- Configures app to send data - on premises and EB download on instances, separate when docker
- Annotations can be added- key value pairs

API

- Website connects other websites, in JSON, comes back to user, shows in JSON
- RESTful - Representational State Transfer - uses JSON
- SOAP - simple object access protocol, XML
- API Gateway - use/manage APIs at scale
  - Makes API using https endpoints - talks to websites & gets info using variables
  - Serverlessly connect to services (Lambda, DynamoDB)
  - Scales, cheap, track and control usage, throttle request (pretect against DDOS), connect to CloudWatch, version control
  - Configure with API, define URL paths, for each resource - set security, choose target
  - API caching- cache endpoint response (reduce latency), has TTL
  - Same origin policy - no cross site scripting, to fix - Cross Origin Resource Sharing
- Import APIs - Swagger v2, can create new API or update
- Throttling - limits steady state to 10000 requests per second
  - Max concurrent is 5000 (can increase for more charges)
  - 429 if above either (throttled)
- Can configure gateway as a SOAP webservice passthrough

Summary

- Lambda- scales out (great for parallel), indp, serverless (Dynamo, S3), can trigger more lambda, use X ray to debug and track, global
  - Triggers - API Gateway,
  - VC - $latest, qualified uses latest (ARN), can split traffic using aliases, versions are immutable, cant split with $latest
  - Step functions - visualize, trigger and track steps, log steps (debug)
  - X Ray - SDK has interceptors (trace HTTP requests), client handlers (AWS SDK services), Http client (http web services)
    - Versions Node, .net, Python, Ruby, Java, Go
    - Integrates with ELB, Lambda, EC2, EB, API Gateway, SNS, SQS
- API Gateway - caching capabilities (most common), low cost, scales, throttle, can log to CloudWatch, use CORS for javascript

DynamoDB

Wednesday, September 30, 2020 3:42 PM

Review
- SSD, spread across 3 Azs, 2 consistency models (Eventual Consistent and Strongly Consistent)
  - Eventual consistent- within a second (default)
  - Strongly consistent - slower but should still work
- JSON, HTML, XML
- Uses primary key - how to store/retrieve data/ identify uniquely
  - Partition key - unique attribute (ID) used in hash function
  - Composite key - combo partition and sort key (ID plus Time for example)
- IAM manages access, can create roles for temp access keys, can use IAM conditions in policies
  - Dynamodb:leadingKeys to allow users to access their items only (partition key = access key)

Index
- Fast queries on SQL (still on Dynamo DB)
  - Local Secondary - created with table, not modified, same partition key as original table, diff sort key
  - Global Secondary- more flexible, create whenever, diff partition key as original, diff sort key

Scan vs Query API
- Query- find entries
  - ProjectionExpression param for only certain attributes of an entry, searches primary key
  - Results sorted by sort key (ascending order, reverse with ScanIndexForward to false), by default they are eventually consistent
- Scan- every item in the table then refine
  - Default returns all attributes, same projectionexpression param
- Q more efficient, scan takes longer, more work- use smaller page size, helps not throttling
- Parallel scans - scan processes partitions of 1 MB at a time, can be done when table is free

Provisioned Throughput- for predictable apps
- Measured in Capacity Units- need to specify Read CU and Write C
  - Write CU = 1 KB write per second, Read CU = Strong Cons Read of 4 KB or 2 Even Cons Read of 4 KB per s
  - Table with 5 Read CU and 5 Write CU can do 5 4KB Strong Cons reads = 20 KB per second or 40KB Eventual Consistent reads, 5 1 KB Writes = 5 KB per second
    - If more needed, then price goes up
  - Does one item at a time so we need to round up size / 4 kb (strong cons) = num CU - divide by 2 if Eventual
  - For write, divide by 1 KB not 4 KB but same process
- ProvisionedThroughputExceededException
  - With SDK, will auto retry until requests succeed, if not reduce request frequency or use exp backoff
  - Exp backoff - auto retries with longer waits between them
    - Aplies to S3, CloudFormation, SES etc

On Demand Capacity
- Charged for reading/writing/ storing- no requirements for on demand, instant scale up and down, pay per request
- On demand - unpredictable, pay per use, spiky short lived peaks

DAX
- Up to 10x performance, clustered in mem cache for dynamDB- microseconds for millions of requests- bursts
- Write through caching service - written to cache and back end store (same time)- points to cluster not table
- Eventual consistent on cache miss (1 sec), can save money on CU in theory
- Not good for strong consistent reads, write intensive (small benefit), small read, don't need microsecond time

Elasticache
- Good for improving read-heavy workloads, freq access data, compute heavy workloads, for I/O intense queries
- Memcached- multithreaded, not multi AZ, object cached
- Redis- key value store, more complex data, master/slave repl and Multi AZ
- Strategies
  - Lazy loading - load to cache when necessary- only requested is cached, node failure isnt fatal, cache miss penalty, stale data (from last cache update) - use TTL
  - Write-through- adds/ updates cache when new data written - never stale, write penalty with update, node fail means data gone until new write, wasted resources if most of the data isnt read

Dynamo Transactions
- Support mission critical, ACID (atomic- all or nothing, consistent - valid, iso- no dep, durable - stays done) transaction
- Read or write multiple items on multiple tables as all or nothing operation, checks for prerequisite before

Dynamo TTL
- Expiry time for data- expired items marked for deletion, deletes within 48 hours- in EPOCH/UNIX/POSIX time
  - Good for removing old/irrelevant data and reduce costs
  - Can be omitted from queries and scans

DynamoDB Streams
- Time ordered sequence of item modifications (insert, update, delete)- record in log, encrypted at rest for 24 hours
- For auditing, good for triggering events- near real time
- Use dedicated endpoint (doesn't slow down table), primary key recorded at least, can store before/after images
- Can see contents of actions, good for lambda

Summary
- Dynamo is low latency NoSQL with tables items and attributes- supports document/ key value models in XML, HTML, JSON
- Max item size is 400 KB, soft limit 20 Global indexes per table
- Has primary keys either partition key or partition + sort (composite key)
- Strong consistent (within 1 sec), eventual consistent (default)
- Access using IAM policies, can use dynamodb:LeadingKeys to allow users to only see a users info
- Indexes - for fast queries - local secondary - created at table creation time, same partition key as table, diff sort key while global secondary can be created at any time, diff partition key, diff sort key
- Scan - all the data then subset vs Query - pick some data, use ProjectionExpression to refine results or filter (in console)
  - Sorted in ascending order (ScanIndexForward only for queries), use smaller page size, isolate scan, parallel scans
- 1 Write CU per second = 1 KB write per second, 1 RCU = 1 4KB strong or 2 4KB eventual reads
  - On calcs divide by size of CU (1 for write, 4 for read), round up, multiply by number wanted per second
- DAX in mem caching, for eventual consistent reads only, point at dax cluster, great for read intensive
- Elasticache - sits in between, lazy loading (only cache when requested) and write through (everything sits there), perf vs penalty
  - Possible wasted data in cache (write through), stale data (lazy loading)

Deployment

Monday, September 28, 2020 4:33 PM

Intro
- CI - code commit, connecting source control
- Cdelivery - everything auto except deploy, codebuiild + codedeploy
- Cdeployment - auto everything, codepipeline

CodeCommit
- Source Control - has code, binary, libraries, track code changes, has version history
- Like github, super easy to use and use branches in console

Code Deploy
- Automated deployment from ec2, lambda, on premises
- In place - app is on each instance and new release is installed - rolling update
    - Capacity goes down while some out of service
    - App versions are called revisions
    - Roll back - need to do another rolling update to redeploy to old
    - Lambda not supported
    - First time deployments
- Blue/Green - instances are replaced where blue is old and green is new
    - Set load balancer to old environment for a roll back (if not terminated)
    - No capacity reduction
    - Pay for 2 environments until terminate
- AppSpec File
    - Config file for parameters in codedeploy deployment
    - For EC2- has version, OS (Linux or windows), Files (config or app files), hooks (lifecycle event scripts to run)
        - Script examples - unzip files, run tests, load balance registration
        - **Yml only** , in zip make sure folder system is correct (usually config, source, scripts folders)
        - Appspec.yml must be in root directory of Revision (deployment change)
    - For Lambda - YML and JSON
    - In place - Lifecycle event hooks - run scripts and order of run (run order)
        - Phase 1 - de register instances from LB
            - BeforeBlockTraffic - run before de register from LB
            - BlockTraffic - de register
            - AfterBlockTraffic - after deregistered from LB
        - 2 - deploy app
            - ApplicationStop- graceful stop app
            - DownloadBundle- CodeDeploy copies app revision files to temp location
            - BeforeInstall- pre install scripts (e.g. backups, decryption)
            - Install - revision to final location
            - AfterInstall- maybe file permissions
            - AppStart- start services stopped
            - ValidateService - run tests
        - 3 - reregister instances with LB
            - BeforeAllowTraffic
            - AllowTraffic - register instances with LB
            - AfterAllowTraffic - after on LB

CodePipeline
- Ci/CD service - configure to auto build, test, deploy - fast, consistent, less mistakes, end to end
- Integrates with Codecommit/ build/ deploy, github, jenkins, EB, CloudFormation, Lambda, ECS
- Workflow is defined, new code appears, code is built, tested, deployed

ECS- container orchestration service
- Apps are created using independent stateless containers
- Docker for Linux and Windows for Windows
- Clusters of VMs (EC2, more control) or use Fargate (Serverless)
- Elastic Container Registry - store images, ECS uses, store either

Docker
- Docker commands
    - Docker build -t myimage - build based on image
    - Docker tag tagname myimage - apply alias to image
    - Docker push image - push image to ECR repo where it can be edited
- **Buildspec**, can be edited in console, build logs in console to view

CloudFormation
- YAML or JSON, define resources from template, uploads to CloudFormation from S3 (delete after resources)
- Version control and peer review, free to use , good for rollbacks and delete stack
- Resources is mandatory in template
    - Metadata, parameters- allowed values prod/test, for env , conditions unnecessary
    - Mappings- custom mappings such as AMI to be used in a region
    - Transform- include snippets from outside code
    - Resources- what to deploy
    - Outputs
- When typing AMIs, remember they are region specific (can only use amis in your region)
- Nested Stacks- stacks create more stacks- just need a type (CloudFormation stack) and **template URL**- s3 url

Serverless App Model
- Extension to CloudFormation for serverlessresources , has its own CLI
- Sam package - takes YAML template and creates sam template, puts in S3
- Sam deploy - creates stack, capabilities create IAM to use SAM
- Both commands take parameters to use template file to do things
- Has transform from serverless
- Handler- specify Lambda Function entry endpoint

IAM, KMS, CloudWatch

Thursday, October 1, 2020 4:49 PM

Review- IAM
- Web Identify Federation- sign in to AWS with Amazon, Fb, Google etc
- Cognito - provides Web fed, acts as identity broker, good for multiple devices, social media sign in
  - Maps to role, allows to use AWS or sign in/up/guest access
  - User pools - directories used to manage sign up/in for apps, can sign in to user pool
  - Identity pools- create unique identities for users, authenticate themselves with identity providers, temp access to AWS
  - Push synchronization - to push updates and synch user data, SNS is used to silently push to mult devices

Policies for IAM
- Managed - created and administered by AWS, common use cases based on job function, cant change
- Customer Managed- you create and administer in your acct, can base off Managed, recc for specificity
- Inline- embedded within the user/group/rule- 1:1 relationship, policy is subresource, useful for no giving permission to others

STS
- AssumeRoleWithIdentity- api provided by STS
- Returns temp sec credentials for users authenticated with mobile or web app or web ID provider to AWS
- STS for non mobile, Cognito for mobile
- Returns AssumeRoleUser ARN identifier with temp credentials and temp credentials (1 hour default)

Cross Account Access
- Manage resources in other accounts/ share resources to another acct - do with switching roles
- Need to create role for other aws acct (need acct ID) then attach policies to role, then in receiving acct, create policy to allow access to other acct's role (can be inline), then switch role
  - To switch role- need acct ID of giver that created role, which role

Policy Simulator
- Can test IAM policies
- Shows user policies (in the acct), then can pick a service and action and see if AWS would let the user do that

Summary
- Web Federation - authenticate with web identity (Google, FB, Amazon) for temp cred to AWS role
- Cognito- identity broker for app and web ID provider- sign up/in/guest, across multiple devices, aws recommended for web ID federation
  - User pools- manage user sign up/in/web Identity providers
  - Push synch to silent push an update to user data
- Policies- Managed - AWS, Customer Managed- editable, Inline- to one user/group/role

Review- KMS
- Works with S3, RDS, DynamoDB, Lambda (encryption), EBS, EFS, CloudTrail (who uses keys), Dev tools, etc
- CMK - customer master key - encrypt/decrypt up to 4 KB (for encrypt data keys) - envelope encryption
  - Has alias (name), creation date, description, key state (deleting, enabled, disabled), key material (KMS or customer provided), stay inside KMS (no exporting)
  - To set up - need to create alias and description, choose material (aws vs customer), add admin permissions (users and roles who can give key through KMS), set key usage permissions
  - Can use cmk to generate, encrypt, decrypt data keys
- KMS API calls
  - aws kms encrypt --key-id YOURKEYIDHERE --plaintext fileb://secret.txt --output text --query CiphertextBlob | base64 --decode > encryptedsecret.txt
    - Turns to base64 encrypted binary data file
  - aws kms decrypt --ciphertext-blob fileb://encryptedsecret.txt --output text --query Plaintext | base64 --decode > decryptedsecret.txt
    - Decrypt, output to plain text (decrypted with base 64)
  - aws kms re-encrypt --destination-key-id YOURKEYIDHERE --ciphertext-blob fileb://encryptedsecret.txt | base64 > newencryption.txt
    - Reencypt (usually done with different cmk)
  - aws kms enable-key-rotation --key-id YOURKEYIDHERE
  - aws kms get-key-rotation-status --key-id YOURKEYIDHERE
  - aws kms generate-data-key --key-id YOURKEYIDHERE --key-spec AES_256

Envelope
- Encrypt using CMK- encrypted data key on data > 4kb, then key is stored with data (all encrypted)
  - This way only transfer data key over network, faster

Summary
- Keys are regional
- KMS is multi tenant (CloudHSM is)

Review- Cloud
- Monitor compute, autoscaling, health checks, ELBs, DB and analytics, Elasticache, Redshift, SNS, SQS, Opsworks, etc
- EC2 by default- host level metrics - CPU, Network, Disk, Status Check at 5 minute intervals
  - Custom- RAM utilization, Mem utilization, storage left, user load/data
  - Custom- min granularity is 1 minute
- Retrieve with GetMetricStatistics, stored indefinitely by default, can change, can get form terminated EC2, ELB
- Can create alarms, events (SNS, lambda function, etc)
- Can be used on premise, need to download agent

Metrics
- Need IAM permissions for cloudwatch in created role (e.g. for Ec2)

Cloudtrail
- Monitor API calls, for auditing, resource & IP address info

Congif
- Record state of aws env and notify on changes (i.e. what were security groups)

Other Services

Thursday, October 1, 2020 12:11 PM

SQS
- Pull based system (fleet to queue), message sup to 256 KB of text
- Standard vs FIFO (messages delivered once, order preserved, 300 transactions per second)
- Message life 1 minute to 14 days, default is 4 days
- Visibility timeout - reader pick up and invisible
- Long polling- returns response when there is message or timeout reached
- Delay Queues - postpone delivery of new messages (0- 900 seconds)
  - Standard- doesn't affect existing, FIFO - recrusively affects
  - Use with large distributed apps to allow for updates
- With large messages (256KB to 2 GB)
  - Use S3 to store, SQS extended Client Library for Java to manage messages and AWS SDK for java (API)
  - With Client library, can choose which messages in S3 based on size, send messages, manipulate S3

SNS
- Scalable, app message delivery, subscription based
- Can trigger SQS, Http endpoint, Lambda functions, email, SMS
- Stored redundantly in multi AZ
- Push based (no polling) simple APIs, flexible delivery, inexpensive, simple

SES
- For marketing/ notifications, purchase confirmations, can trigger lambda or SNS, incoming and outgoing email

Kinesis
- Streaming, KB worth
- Streams- shards, more means more compute, stay 24 hours to 7 days, 5 trans per second read, 1000 records p second for write, send to fleet, send to storage or EMR, Dynamo, S3
  - Shards - 5 read trans per second, up to 2 MB per second, 1000 write per second, up to 1 MB
  - Kinesis Client Libraries runs on Consumer instances, tracks shards and can discover new shards
    - For every shard, is a record processor (process data), manages them, load balances, equal per inst
    - One instance per shard generally, can handle more record processors, don't have more than shards
- Firehouse- real time, computes with lambda, send to S3/redshift/Elasticsearch
- Analytics - for analyzing in Kinesis, send to storage

CLI Pagination
- Control num items in the output when runnign CLI command, by default 1000 (page size)
- Run **aws s3api list-objects my_bucket** for a bucket of 2500 would make 3 calls
- Might get **time out** errors, can change with **--page-size 100** for example or **--max-items 100**
- Still retrieves them all, just uses more api calls instead of less asking for more

Beanstalk
- Throw code in zip and run- can do Java, .net, PHP, Node, Python, Ruby, GO, Docker
- Saves versions in S3
- Fastest way to build, auto scales, can select EC2 instances, can manage updates, monitor app health in dashboard, integrate with cloudWatch and Xray
- Scripting (formation) vs GUI
- EB Deployment strategy
  - All at once - deploys to all instances simult, downtime while all update, not ideal for mission critical, roll back if update fails with another all at once
  - Rolling - deploys new version in batches, env capacity decreases but no downtime, not ideal for perf sens
  - Rolling with additional batch- launches more instances, deploys new version in batches,
  - Immutable- fresh group in autoscaling group are put into existing, old are terminated - preferred for mission critical OS
- Customize EB with elastic beanstalk config file (install packages, create linux users/groups, run shell commands, in YAML or JSON, must have .congif in folder .ebextensions in top level directory of app)
- With RDS
  - For dev/test deploy (not great for prod since couples EB and RDS) - inside beanstalk
  - For prod- launch rds outside for more flexibility, types, decouple - outside beanstalk, more work
    - Need new security group for EC2 autoscaling group and connection string from your servers to RDS
- Docker
  - Self contained, hold all info to run, beanstalk handles capacity provisioning, LB, scaling, health
  - Single Docker on EC2 or mult on EC2 cluster of ECS, deployed as zip file and easy to update
  - Upload from S3 or local, can use CodeCommit but need Elastic Beanstalk CLI

Parameter Store
- Save confidential info, in EC2, uses KMS key - store as String (plain), String List, secure String, have names

Summary
- SQS - pull based, standard (best effort), FIFO (order and 1 message), visibility timeout 30 seconds (max 12 hours), long polling (max 20 sec) to save money only returns with message or timeout
- SNS- scalable push notifications, SMS, Email, SQS, HTTP, subscribe
- SES - email only, in and out email, not subscribe, only need email
- Kinesis- Streams (video for secure stream, data for custom apps in real time), firehouse (near real time analytics with business analytics), can configure lambda to do things before sending along
- Beanstalk - scale and deploy web app, JAVA, Python, PHP, Ruby, Go, Docker, .Net, Node, can provision yourself and can update and can do health checks
  - All at once (change it all), Rolling (change some), Rolling with additional (add some), immutable (add them all)
  - Customize with .config files, JSON or YAML, save .ebxtensions folder in top lvl directory
  - Launch RDS for dev/test, launch RDS outside for decouple but change SG and connection

Other Notes

Friday, October 2, 2020 10:56 AM

Deployment
- CloudFormation
  - **Cfn-init** installs packages and stops/starts services on EC2
  - Stack termination protection- prevent accidental deletion
  - Deletion Policy to retain to not destroy resources on stack deletion
  - Lambda place python code - put in ZipFile parameter
  - StackSets for multiple aws accts
  - Change Sets - preview how changes to stack impact resources
  - Function- attach config info for lambda, LayerVersion, API- API gateway config
- SAM- uses CloudFormation
  - Package- same as CloudFormation- zips code to S3
  - Publish- pushes to serverless app repo
  - Deploy- same as CF, deploys packaged file
- CodeDeploy- compatible with EC2, ECS (EC2 and Fargate), Lambda, On premises
  - Can also do blue/green for basically immutable (has DNS change)
    - For Lambda, EC2/On Premises, ECS, CloudFormation
  - In place deployment
    - For EC2/on Premises
  - Supports in place to on premises and blue/green to ECS
  - Lambda cannot do in place and
  - _ percent _minutes for lambda, can use with every_minutes, HalfAtATime for ec2
  - CodeBuild- buildspec.yml to run a build
- CodePipeline- can add manual approval action using SNS topic
- EB - All at once, Rolling, Rolling w/batch, Immutable?
  - Upgrade simply with zip files in the console
  - EB supports Docker, Ruby, Go, Java, Tomcat, .NET, PHP, Python, Node
    - Supports app servers Tomcat, Passenger, Puma and also does Docker
  - Cron.yml for periodic, env.yml for environment, on CLI use eb deploy
- Spot instances for stateless web services, image rendering, big data, parallel computations
- CodeCommit- need Https Git credentials and public SSH keys

Serverless
- DyanmoDB
  - Atomic Counters- to count things
  - Batch Operations - easy decreasing overhead and do multiple
  - Conditional Writes/Deletes- only write to table if conditions
  - Session Data - better than in memory cus doesn't scale well, can set with TTL
  - TransactWriteItems - operation to group multi transactions (all or none)
  - BatchWriteItem - groups multi transactions but isnt atomic
  - Kinesis Adapter to consume Dynamo Streams (real time analytics)
  - Scan with filter expression for non PK, **query filters on PK**, projection for attributes returned, limit for less returned
  - Streams analyzes last 24 hours, StreamViewtype parameter can set new/new_and_old images to be streamed
    - Time ordered sequence of item level mods in a table
  - Projection Expressions- what attributes returned with get/**query**/scan
  - **ReturnConsumedCapacity** - to return num WCU by any operations
    - Total- all, Indexes- subtotals and secondary indexes by operation, none - default
  - Global tables- last writer wins with concurrent requests
  - Global (eventual consistency, have CU from index) vs Local (both const, cant make/del after table) indexes
    - Make sure global secondary index provision CU >= base table, use on non-key attributes
    - Use local when need to switch sort key, up to 5 per table
  - Locking Strategy- opt with DynamoDBVersionAttribute with java SDK
    - Optimistic - client side is same as Dynamo, don't use with Global tables if don't want overwrites
    - Pessimistic locking is exclusive lock so no one else can start modifying
- Lambda- events - API Gateway, IoT, Alexa, App LB, CloudWatch, CloudFront, CodeCommit, Cognito, Dynamo, CloudWatch, Kinesis, MSK, S3, SNS, SQS
  - 50 ms to 29 sec integration timeout, 1000 default limit in region
  - 50 MB upload deploy package limit, 250 MB unzipped limit
  - Timeout 3 seconds to 15 minutes
  - Authorizers- API gateway feature, takes in client request and returns policy, also cross acct lambda auth
    - Token based - client identity as JSON Web Token - custom auth scheme for both, **Oauth SAML**
    - Request parameter based - client identity, string params, variables
  - Aliases- use with traffic shifting to gradually increase traffic
  - ARN - unqualified points to latest, qualified can point to any version
  - BasicExecutionsRole is for cloudwatch
  - Creates ENIs to connect to VPCs, increase in mem triggers increase in CPU available
  - Custom integration- proxy + how incoming request data is mapped and response is mapped
  - **Custom Runtime** - need file named bootstrap, runs functions **set-up code**
    - Can use to run different languages (C++,) & created within a layer
  - Execution Context- Reuse existing conditions, temp runtime env initializes dependencies, /tmp folder
  - Improve performance by including only necessary libraries, establishing DB connection in Lambda env
  - InvalidParameterValueException- gave roll that Lambda cannot assume
  - Layers- allows include packages without putting in deployment package, pull in code and **dependencies**, max 5
  - Proxy Integration - for integrating with API gateway, just set method to POST, ARN of lambda func, IAM role
  - Invoke API uses requestresponse (synch) or event (async) or dryrun (test) to run lambda functs
  - Synchronous invocation- API gateway, ALB, Alexa, Cognito, Cloudfront, Kinesis Firehouse
    - Asycnh- S3, SNS, SES, Cloudformation, CloudWatch, CodeCommit, Config
    - Poll Based - Kinesis, SQS, DB streams
  - Unreserved account concurrency can't be less than 100
  - Use CloudWatch events to schedule Lambda trigger every 24 hours
  - VPC - need NAT gateway and route to connect to internet if inside, needs perm to get in VPC
- Errors
  - Info Errors (100), Success response (200), Redirect (300), Client error (400), Server error (500)
  - 403 access forbidden, 404 not found, 408 request timeout, 429 too many requests
  - 502 invalid gateway, 503 service unavailable, 504 Gateway timeout

Security
- Encrypt KMS only up to 4KB, use generate-data-key to create key and envelope encrypt for files > 4KB
- End to end encryption between EC2 and ELB - add secure listener to ELB, config instances to listen to it, terminate HTTPS connections to EC2 instances (don't need them anymore)
- Delete root access keys- don't use the root acct for managing resources
- For CodeCommit access, generate SSH key and associate as public SSH keky, generate HTTPS Git credentials
- GetSessionToken- for use with MFA code, returns access key, secret access key, security token - to use MFA on API calls
- Store credentials (e.g. EC2) in AWS CLI profile- temp, rotate, secure - instance can only have one profile at a time so if u want to add permissions to instance, must change role
  - Instance profile is container for iam role, used to pass info to EC2 instance
- Use GenerateDataKey to get plain text data encryption key- erase later from mem and store it as encrypted
- With S3 KMS, set amz-server-side-encryption to aws:kms and will use default key, with customer have to set algo, customer key, encryption customer key MD5
- When uploading multipart you need decrypt permission, encrypt, ReEncrypt,GenerateDataKey, Describe key (encrypt nec for all, decrypt nec for multipart upload)
- SSL/TLS certificates= Certificate manager or IAM certificate store
- Cant use IAM roles with on premises (use access keys)
- Key pairs are for SSH

Services
- API Gateway - only exposes HTTPS endpoints, API calls and cache freq API calls, has stages as versions
  - Invalidate cache with cache-control: max-age =0, require authorization when creating so that not all go down at once
  - Allow the action execute-api:InvalidateCache from resource
  - Cloudwatch tracks cachehitcount (hits frontend), cachemisscount (hits backend-)
  - Stage Variables - pass config parameters to function through **API** (need to map it)
  - Integrate API with API Gateway with HTTP Proxy (set method and URI) or Custom (do more)
- APPSync - create flexible API from one or more data sources, local data access, multiple users
- Athena- for fancy SQL, real stuff and costs more- use S3 select for simpler things
- AutoScaling- can be EBS backed or instance store backed, EBS can be stopped and started, configure with policy
- CloudFront- not for DB caching (use Elasticache), end to end SSL with View Protocol Policy and Origin Protocol Policy, use origin failove r to with origin group and 2 origins- secify what to do on certain http code (such as HTTP 504)
- CloudHSM- hardware security modules, for cryptographic things
- CloudWatch - to track mem on ec2 (detailed, can also do swap, disk util), Xray tracks requests
  - Namespace for grouping monitorizations together
  - Period is how often data, eval period is how many periods to study, datapoints to alarm is how many points to create alarm
  - Standard Resolution (1 min intervals) or High Resolution (1 sec)
  - Logs agent for mem metrics on EC2
- CodeGuru Reviewer - automates reviews of code in repo, in CodeCommit
- CodeStar- automate CI/CD and has project dashboard for day to day activities
- Cognito
  - User pools- can have MFA, identity pool pulls from user pool to create access, use MFA with user pool to protect identities
    - Use for sign up/sign in while identity pools is for guest users
  - Events- use with Sync to change profile, preferences, game state
  - Sync - cross device syncing, caches data locally but can push sync to update other devices
  - Can use to track usage across different devices (e.g. netflix videos limit devices)
- Elasticache - good for storing session data for scalable web app- in mem key value store (Redis is better, memcached not HA)
  - Memcached for simple, running large nodes with multi cores, scale in and out/remove nodes, cache objects (DB)
- ELB - can set desired capacity to manually change # instances, cant restart and ELB
- EB- handles containers, resource provision, load balance, auto scale, monitor, uses XrayWriteOnlyAccess to upload to Xray
  - Cant configure Lambda, can configure EC2, CloudWatch, ALB
- ECS- doesn't handle ^, still have to enable, configure but does handle containers
  - Task placement strategies - spread - place evenly (high available), binpack- least sufficient (less instances), random- willy nilly that work and still honors constraints
  - Fine grained control for custom app arch
  - Container instance roles don't work on Fargate (only EC2)
  - Cluster Queries - expressions to group containers by attribute
  - Separate X Ray Docker image than other
  - Task Definition- port mapping, , service scheduler (run tasks manually), container instance (runs ECS agent)
- EC2- user data for config tasks, meta data for IP addresses, name
- Glacier - standard is 3-5 hours, Exped it 1-5 minutes, Bulk is 5-12 hours
- Kinesis- retries from producer or consumer- resolve with embedded primary key within the record (task/job), 1-7 day ret.
- Parameter Store - storing access strings, store AMI IDs to be used with CloudFormation, free
- RDS- enhanced monitoring for CPU bandwidth, mem consumed by each DB
  - Transparent Data Encryption - auto encrypt before written to storage and decrypt when read
  - Slow query log, general log, error log can all be exported to
- S3 - CORS - allowed origin, allowed method (get, post, put, delete, head), allowedheader, can create endpoints for VPC, use multipart for >100MB
- Secrets Manager- parameter store but can rotate and costs money
- SNS- can be made to customize to different endpoint types
- SQS- can set deduplication IDs so duplicate messages will not be delivered for an interval (5 min default)
- Step Functions- serverless workflow orchestration, takes in jSON
  - InputPath (filter input), Parameters (pass key values), ResultPath(return input/state result), OutputPath (filter output)
- SWF - task coordinator
  -

markers - record events/ custom info	Timers- notify decider when time has elapsed
Signals- inject info into workflow	Tags - filter listing of executions

- Xray - send data in segments- segments has resource name, details about request, details about work done
  - Segment Documents- JSON, up to 64 KB, sent to Xray with PutTraceSegments or to daemon (subsegment upload best)
  - Subsegment has granular info on downstream calls, timing (AWS, HTTP API, SQL)
    - Fields: namespace (AWS for AWS SDK calls, remote for downstream calls), annotations (key-value for searches/filter), metadata (other info not for filtering)
  - Inferred Segments- generated, GetBatchSummaries lists IDs & annotations, BatchGetTraces gets list of traces
  - Filter expressions to find traces related to certain paths/users
  - Annotations- key-value pairs to index filter expressions- regular will trace request of application, subsegment section will trace requests to resources/HTTP APIs/SQL Databases, good for filtering
  - Can search segment fields via console of using GetTraceSummaries API
  - Sampling rules tell Xray SDK how many requests to record for set of criteria
  - UDP port 2000 (in task definition)
  - Enable xray with Xray-daemon.config file in .ebextensions folder in source code (only in EB, not EC2)
  - Env Varaibles - Trace ID - tracing header, Context Missing - what to do if Xray records data, demon address (IP address:port)

Monitoring

Thursday, October 29, 2020 11:22 AM

Cloudwatch

- Host level metrics - CPU, Network, Disk, Status Check - not RAM
- Default every 5 minutes, detailed monitoring is 1 minute intervals
- Can retrieve data from terminated EC2 or ELB, logs default stored indefinitely
- Alarms - check EC2 CPU, ELB Latency, charges on AWS- trigger events
- Custom metrics- min granularity is 1 min
- Custom Dashboards- Global, shows per region

EBS

- SSD only can be boot volumes, 10000 IOPS, large DBs for provisioned, big data for throughput optimized
- Burst up to 3000 IOPS
- Degraded/ Severely degraded is a warning
- EBS volume can increase size, type, IOPS without attaching

ELB

- Access logs- latency, time, server response code, disabled by default - can store EC2 data after deletion
- Request tracing - for ALB only, track Http requests from clients to targets
- Cloudtrail logs, Access logs, Request Tracing, Cloudwatch metrics 4 ways to monitor LBs

Elasticache

- Memcached- multithreaded, memached_connections_overhead, increase if >50 mb
- Redis- not multi meaning all nodes share CPU, no swapUsage metric- uses reserved-memory
- SwapUsage mem reserved on Disk for more RAM, should = RAM
- Evictions- memcached has no rec- choose threshold- scale up or scale out with Redis- scale out (read replicas)
- Concurrent connections- choose threshold, spike either means lots of connections or connections need fixing

Grouping- Resource groups- group by 1+ tags, can be used to automate tasks

Cost Explorer

- Cost explorer- see past 13 months, forecast 3 months, see cost and usage, get recs for reserved instances,
- Allocation Tags- activate and become tags to be used in cost explorer

Config

- See config details at any time, for auditing/security/ resource tracking, per region, saved in s3 bucket
- Config items (attributes of resource), snapshots (collection of configs), stream, history (resource over time)
- Compliance checks with periodic checks, configuration changes or managed rules (not custom)

EC2

Friday, October 30, 2020 10:42 AM

Intro

- Hibernate (don't lose EC2 RAM) vs stop (do lose)
- Can set launch groups (only launch if all launch)

Launch Issues

- Instance limit exceeded- default limit 20 by default, per region basis
- InsufficientInstanceCapactiy- aws doesn't have capacity, could get reserved too

EBS & IOPS

- What happens when exceed IOPS, queueing at first then slow down
    - For gp2 can increase size (to 16000 max) or switch to PIOPS

Bastion- connect to EC2 vs Nat (connect to internet), ssh/ RDP and only in to instances

ELB

- ALB- layer 7 , NLB- super fast TCP layer 4
- Can pre-warm ELB to test if it can handle lots of load (10x ish)
- Static IP addresses- NLB doesn't change IP when workload changes (can put ALB behind NLB)
- 200 - success
- 400 - client side
    - 400- Bad request, 401 access user denied, 403- completely forbidden, 460- client closed before LB respond (increase timeout), 463- LB received XForwardedFor with >30 IP addresses
- 500 - server side
    - 500 internal server error, 502- bad gateway, 503 - service unavailable (no targets), 504- gateway timeout (problem web server/db), 561- unauthorized- error code with identity provider
- Cloudwatch
    - For ELB and backend instances, at 60 second intervals by default
    - BackendConnectionErrors- unsuccessful connects
    - HealthyHostCount, UnhealthyHostCount
    - HttpCodeBackend_200,300,400,500
    - Latency- how long to respond, RequestCount- num completed over an interval (1 or 5 min) , SurgeQueueLength- num pending requests (max is 1024), spillovercount (num requests rejected)

Systems Manager

- AWS infrastructure
- Run Command- security patching, patching installs, shell scripts
    - Stop/start/run patch, attach ebs, create snapshots, run shell scripts
- Can organize inventory by env, app, can do on premises too

Placement Groups

- Minimize hardware failure impact, low latency, high throughput
- Cluster (all in AZ) for low latency, partition (logical segments in diff rack) for large distributed networks, spread (each instance on sep rack) for small num critical instances

High Availability

Friday, October 30, 2020 1:58 PM

Elasticity- short term, on demand (increase num ec2)
Scalability- long term, infrastructure (increase size ec2)

- Aurora is awesome (serverless too)

RDS

- Copy in other AZ (multi AZ), read replica (read only copy of DB)
- Snapshots on primary
- RR can be multi AZ too, built off multi AZ DBs
- Replica lag, asynchronous, can be in different region
- Encryption- take a snap, copy the snap to same/diff region and encrypt that copy, then restore
- Sharing snaps between accts -
    - Can share with AES 256
    - Need **custom** KMS key, create RDS snapshot using CMK
    - Cant share TDE (transparent data encryption) snapshots, cant share as public
- Versioning- can see config details in console or describe-db-instances --region
- Aurora
    - HA RDS, managed, 6 copies on 3 Azs, self healing, scans continuously
    - Starts with 10G and scales by 10, lots of read replicas (standard 15 or MySQL 15)
    - 100% CPU utilization- scale up (size) if writes and scale out (num read replicas) if read
    - Serverless- per second basis, on demand
    - Encrypted by default, writer and reader nodes make up aurora cluster
    - Failover priority- lower num means goes toward lower
    - Create reader node in console actions, cant disable encryption
    - Make sure set up multi az for RR in new region

Maintenance Windows

- Document DB, Dynamo DAX, Elasticache, Neptune RDS, Redshift
- Not EC2 (must do manually), Lambda, QLDB (quantum DB)

Elasticache

- In mem cache for most freq DB queries
- Memcached, redis (multi AZ)
- Cachehitratios- ratios served from CDN not origin is ratio
    - More requests from CDN, better
    - Max by longer cache TTL, cache by cookie values, request specified headers, Query String parameters (cons variables r better means more caches), use HTTP, remove accept encoding header
- Can also Pub/Sub, Sorted Sets, In mem data store

Troubleshoot Autoscaling

- Make sure Key pairs (logins), SGs, Autoscaling group created, make sure autoscaling config, instance type, AZ, EBS device mapping, autoscaling service permissions, instance store AMI when using EBS are all correct

Storage

Monday, November 2, 2020 2:29 PM

S3
- Charges- per Gb, requests (get, put copy etc), storage management (inventory, analytics, tags), taking data out, transfer acceleration (uses cloudfront)
- Https 2-- for API or CLI (not console)
- Server access logging (can configure to move to glacier)
- MFA delete- with versioning to prevent accidental or mal deletes
  - Enforces code MFA from device for deleting and suspending versioning
- Encryption- enforce with policy (bucket)
  - In transit SSL, TLS and at rest SSE (SSE-S3, KMS, C)
  - KMS has envelope encryption, audit trail while S3 doesn't

EC2 storage types
- Instance - 10 GB max, all volumes deleted with instance, data gone after terminated
- EBS- 1 to 2 TB, root is terminated by default when instance terminated (can change only at creation)
- Encrypt on provision, volumes for ec2 in same AZ
- Change sizes on the fly, should stop EBS then snap but don't have to
- Can create AMI from both images and snaps
- Snaps encrypted automatically and so are volumes created from encrypted snaps
  - Cant share encrypted snaps, can share with others or public
- Downtime- Most encryption only at creation
  - EFS (new & migrate), RDS (new & migrate), EBS (snap and encrypt)
  - S3 can encrypt whenever

KSM & CloudHSM
- Store and manage cryptographic keys- HSM to protect confidentiality of keys
- KMS- multi tenant managed, free tier, sym
- HSM- dedicated, not free, within VPC, FIPS lvl 3, sym or asym

AMIs
- Template, launch permissions, block device mapping to specific additional EBS volumes at launch time
- Defaults, custom (from existing AMI, customize, create custom AMI image by registering)
  - Per region basis (registered and use)
- Sharing- either private, public, or sellable (marketplace)
  - When sharing, creator pays for storing AMI (s3) and has control
  - Can copy AMI by receiving read permissions for storage (EBS or S3), and if copy u become owner
  - Cant copy encrypted AMI- copy and re-encrpyt with your own key (need owners snap and key)
  - Cant copy billingProducts (Windows, Redhat, AMIs from marketplace) cus they have subscription fees

Snowballs
- For region transfer
- Use for many TB or PB data, Don't want expensive upgrades for one time data transfer, frequent backlogs of data, physically isolated with no internet
- Edge- 100TB device, does S3 compatible endpoint, runs lambda, for edge computing, need client

Storage Gateway
- On premises software which connects to AWS
- File Gateway- NFS/ SMB - Network File system- files stored as S3 objects, using a mount point, backed by S3 (ACLs, Bucket Policy, HA, replication)
- Volume Gateway- cloud backed using iscsi protocol
  - Stored Volumes- store data locally and backup to Aws - low latency and need own storage infrastructure, asynch backups (EBS snaps in S3)
  - Cached Volumes- store data on S3 and bring locally- better for freq access data
- Tape Gateway- Virtual Tape Library- in glacier

Athena
- Interactive query service on S3 using standard SQL- Serverless, pay per query/ per TB scanned
- Can query log files in S3, ELB logs, S3 access logs, can generate business reports on data in S3, run queries on click-stream data, analyze cost and usage reports

EFS
- Can be accessed by mult servers in VPC and other
- Managed etwork File System, NFS protocol, scalable, multiple EC2 can access (cant do this with EBS), has lifecycle management (IA class) encryption at rest and in transit
- Encryption at rest, only at provisioning, if after must be new system,
- NFS 2049 port

Security & Compliance

Tuesday, November 3, 2020 4:52 PM

Compliance
- PCI - security for transactions
  - Firewall configuration, Doesn't use vendor supplied defaults passwords or other parameters
  - Protect data, encrypt transmission across open networks (SSL)
  - Protect malware and update security software, develop secure systems
  - Restrict access to need to know, identify and access system components
  - Restrict physical access, track and monitor access
  - Regularly test security, maintain a policy
- ISO - security management system
- HIPAA - health insurance and medical records
- Fedramp- fed security
- Sas (auting), Soc (acct), FISMA (gov), FIPS (crytpo)

DDOS- denial of service
- NTP Amplification/ Reflection- sends a separate server your IP to give you a big packet
- L7 app attacks (flood get requests), Slowloris (lots of open connections)
- Mitigate - minimize attack area (use ELB), be ready to scale (autoscaling), safeguard exposed resources, learn normal behavior
- ELB, CloudFront, CloudWatch, Autoscaling, Route53, WAFs
- Shield - free, on ELB, CloudFront, Route 53 - protects against SYN/UDP floods, reflection attacks
  - Advanced- more on ELB, CloudFront etc for larger attacks, dedicated team - $3k a month
  - Always on, flow based monitoring, protects against fees from attacks

Marketplace security products
- Kali Linux - penetration testing
- Permission required for pen testing (some pen testing don't need permissions for)
- Firewalls, WAFs, antivirus, security monitoring, all at different rates

MFA- Can enable MFA with CLI or console, on root or user accts, can do with STS service
- Can get credential report from IAM show who uses MFA
- PassRole - for sharing role to another acct, to aws service to assign temp permissions

STS
- For granting temp access to AWS
  - From federation (active directory, SAML, SSO without being a user in IAM)
  - Federation with mobile
  - Cross acct access

Logging
- CloudTrail, Config, CloudWatch Logs, VPC Flow Logs
- Use IAM to restrict unauthorized access, S3 buckets, MFA
- Log changes using Config (set up rules) and CloudTrail
  - Don't let modifications with IAM, S3 policies, CloudTrail encryption, validation

WAF
- Http and HTTP requests, configure what IP addresses, parameters
- Allows all but specified, block all except specified, or count matched conditions
  - Can count strings, ip addresses, length or requests, presence, SQL code, scripts
- Integrates with ALB, CloudFront, API Gateway

AWS Hypervisors - Virtual Machine Monitor creates and runs VMs
- EC2 on Xen, can have guest OS on
  - ParaVirtualization (part virtual) - ring based, iso by layers
  - Hardware VM (fully virtualized)- rec
- Isolation- physical, firewall, security groups, Virtual interface, hypervisors
- Hypervisor access is just admins with business needs, MFA
- Guest access to ec2 user respons
- Mem scrubbing- EBS auto resets storage blocks after customer usage

Dedicated
- Instances- physical isolated at hardware level, may share with your acct, charge per instance
- Hosts- physical server distinction, additional control how hosts placed on server, therefore can use same licenses when same server being used, charge per host, more visibility
- Both have dedicated hardware

Run Command
- To automate common admin tasks, applying updates, joining domains (Windows) at scale
- commands can be applied to groups or manual selection
- SSM agent needs to be installed,
- Commands can be issued using AWS, CLI, Windows Powershell, SDKs, Systems Manager API
- Can be done on premises or EC2

Parameter Store
- Store credentials as String, String List, Secure String (encrypted with KMS)
- Use with EC2, CloudFormation, Lambda, EC2 Run Command

Pre-signed URLs
- Can do via CLI with presign parameter, then creates access key ID, default 1 hour TTL
- Typically with SDK, --expires-in then seconds

S3 Restrict access
- IP address restricting with bucket policy, in condition - aws:SourceIp
- Can blacklist or whitelist

Config
- Can add rules (managed or custom) such as no public read access, no public write access, etc

Inspector vs Trusted
- Inspector - automatic assess for security, on EC2
  - Create a target (instance), install agent on target, create template, run, review findings
  - Needs tags
- Trusted Advisor - paid, security, cost opti, performance,
  - Basic core checks and recs
  - Full - business or enterprise customer

Service Limit- Use trusted advisor (performance)

Shared Responsibility
- AWS- infrastructure, some compute/storage/networking, DBs
- Customer - encryption, IAM, system configuration, network protections, customer data
  - Services- Ec2, EBS, Autoscaling, VPC
    - AMIs, OS, data in transit/rest, data stores, keys, policies/configs
- Container services - RDS, EMR, EB where Amazon does OS but you do everything else
- Abstracted services- S3, Glacier, DynamoDB, SQS, SES where Amazon manages everything but policies

Other Security
- SGs- stateful - open ports auto allow outbound ports (ingress = egress)
- Can use Athena to query cloudtrail logs
- AWS Artifact- downloads for security docs for regulators/ auditors
- CloudHSM - more control on key, symmetric & asymmetric, single tenancy, more money
- Instant S3 encryption, migrate encryption with Dynamo, RDS, EFS, EBS

CloudTrail
- What is logged- metdata about API calls, identity of caller, time of API call, source IP, request parameters, response elements returned
- Sent to S3, can manage with lifecycle, delivered every 5 minutes with up to 15 min delay, can aggregate logs across regions/ accounts
- Can create read/write events to be listened to, can set prefix or not, encrypted, sns notifications, log validation
- Use SHA 256 hashing (digest files every hour, uses private key), IAM policies, bucket policies, use IAM Group, MFA delete, li fecycle,

Encryption
- EFS, EBS, S3 allow native encryption at rest

Networking

Tuesday, November 3, 2020 4:53 PM

VPCs
- Cidr ranges (10/16, 172/16, 192/16), internet gateway (connect from internet, 1 per VPC), Virtual private gateway (connect from VPN), routers, tables, NACLs, subnets (with ranges), instances (with SGs- can span subnets or AZs)
- Default VPC- immediately deploy instances, all subnets have route out to internet, each EC2 instance has public and private IP address
- VPC Peering- connecting VPCs via direct network route using private IPs, can peer in same or different AWS acts, no transitive peering
- 1 subnet = 1 AZ, max is /16 an min is /28

NAT
- Disable source/destination check, NAT must be in public (put in a SG), must be a route out of private subnet
- Can use autoscaling groups, multiple subnets in diff AZs and script to automate failover
- Increase instance size if bottlenecking
- Gateway- autoscale up to 10G, not associated with SGs, auto assign public IP, far more secure

NACLs and SGs
- NACLs- many subnet to 1 ACL,
    - Rules have # and processed in order, allow/deny rules
    - Cannot span VPCs, default closed off, can block IPs
    - Ephemeral Port - short lived IP for the client - allow for outbound rules
    - Stateless- inbound doesn't necessarily equal outbound
- Default NACL is completely open (custom default closed)

VPC Endpoints
- Interface endpoint - ENI on EC2 as entry point
- Gateway endpoint - highly available to lots of services
- To go over private networks

VPC Flow Logs
- At 3 levels - VPC, Subnet, Network Interface (big to small)
- Filter (what to log), role for flow log creation
- Can stream to lambda or S3
- Cant change after creation, cant use with VPCs that are peered to yours (must be in same acct) doesn't log Amazon DNS server, Windows instances, metadata, DHCP

CIDR calcs
- 65,539 for /16, 4096 for /20 , 256 for /24 16 for /28 and always take 5

Direct Connect Gateway- like a direct connect extender- BGP
- Basically a NAT gateway, aws backbone, to connect your site in a new region (don't make another direct connect)

DNS
- Top levels - .com/.edu/.gov then 2nd level r .co.uk/.gov.uk etc
- Domain registrars- organize domain names and ensure no duplicates
- Start of Authority Record - name of server that supplied data, admin of zone, current version, default TTL,
- Name server record - record by top level domain to content DNS server
- TTL- lower means easier to change
- CNAMes- one domain to another, but cant do zone apex (acloud.com, needs www.acloud.com)
- ELBs do not have predefined ipv4, uses DNS name

Routing
- Simple routing- one record to 1+ IP addresses
- weighted routing, latency based, failover routing (choose what to monitor, where to fail to), geolocation, geoproximity, multivalue (multiple record sets, simple with health checks)

Automation & Others

Tuesday, November 3, 2020 4:53 PM

CloudFormation- YAML or JSON
- Parameters- input cutom values
- Conditions - based on env
- Resources - mandatory
- Mappings - like sending a region to an AMI
- Transforms - reference S3 resources (maybe to be used as Lambda code)
- All or none deployment (can disable in console or disablerollback to true)

EB- full control but little interaction- updates, monitoring, metrics, health checks
- Java, PHP, .NET, Go, Docker, Node
- Tomcat, Passenger, Puma, IIs

OpsWorks - server config with Puppet or Chef
- Uses managed instances, enables management with code, works with existing

Service Catalog
- Catalog of approved AWS products for your users
- Has admin control (IAM), personalized portal, centralizes, cloudformation templates and portfolio to group them (regional)

S3 Bucket Policies and WildCards
- Uses * to mean everything in, make sure object actions have * and bucket actions (create, post, delete) don't have

CloudFront
- Status codes - 400- bad request, 403- access denied (public), 404- not exists
- 502- Bad gateway (cloudfront cant connect), 503- service unavailable (performance), 504- gateway timeout (traffic)

Direct Connect with mult accounts
- Better than internet, can access VPCs in different accounts (up to 10 VPCs), must be in same org

InterRegion VPC Peering
- Can peer between regions, send peering request and accept then update routing tables (never uses internet)

HTTPS and SSL certificates - TSL- transport layer security
- Put in Certificate Store and Certificate Manager - new or existing, allows to provision, manage deploy certs
- Free of charge and renew automatically

CloudFormation
- Use Stack Policy to protect resources from unintentional updates
- Use IAM, don't hit service limits
- Failed rollbacks- Update_rollback_failed - maybe cus cloudformation manual changes, stack no longer exists,

Quiz Notes

Tuesday, November 3, 2020 4:53 PM

AWS Networks

Tuesday, October 27, 2020

Basic Network Design

- Up to 5 VPCs by default, Cannot span regions
- Default vpc set up for one subnet per availability zone, also other things (IG, sec groups, ACLs)
- Each VPC has Name, CIDR block, Tenancy (default or dedicated hardware) - cant be changed after (except name)
- Num ipv4 is 2^(32-num), AWS reserves 5, default is 172.31.0.0/16
  - Don't overlap IP addresses with other VPCs, on premises networks and think of how many subnets (1 per AZ), how many tiers in environment, don't use default vpc if communicating
  - Can choose between /16 and /28 for ipv4 and can use ipv6 with fixed size /56 (cant choose)
- To create VPC- choose IPv4, choose tenancy, optional choose ipv6 CIDR (dual-stack mode config routing for each)
  - IPv4- can choose between 16 and 28 CIDR block, can choose values, difference public and private addresses
  - IPv6 is fixed /56 (huge), AWS gives you, no diff public/private address
- Subnets- specified CIDR range within VPC- segment of VPC in one AZ, can be public/private/ VPN only
  - Specify target AW, determine CIDR (smallest is /28 and biggest is /16), 5 reserved - first 4 and last
  - .0 for network address, .1 for vpc router, .2 for DNS, .3 for AWS, .255 for network broadcast
- VPC router- where routing decisions begin , highly available, by default allows communication
- Route table- has entries with dest (CIDR, IPs) and targets (gateways, ENIs, instance, and local)
  - Destination is either defined CIDR or prefix lists (dest is VPC endpoints), Target is everything else
  - First rule: Destination is CIDR, Target is Local - most specific route, all resources in VPC can talk to each other
- ENI- elastic network interface - 1 AZ forever
  - Always in VPC with specific subnet, at least one SG, primary private internal IPv4 addresses, Mac address (unique), auto assigned external IP (cant outlive ENI, made at creation), Elastic IP (can outlive ENI), source/dest check
  - Can have mult ENIs for one instance, usually max is 2, NIC teaming not supported (more ENIs better band)
- Internet Gateway
  - Translates in between public and private, 1 per VPC, feeds to route tables (1 per subnet) and forwards any traffic to public instances
  - Need **public IPs** to use with IG and need route tables to connect to IG
    - For EC2 or instances without public Ips, can use Nat instances with it
  - Elastic IPs (public and static) vs Dynamic external IP (cant disassociate address from ec2)
  - Dual Homed - Can launch ec2 with 2 ENIs on 2 subnets (same AZ) - more flexibility and better ENI security, e.g. one is for internet and one is for data center
  - Virtual Private Gateway - for ingress/egress, data centers, customer gateways
- NACLs - parking garage security- watches the subnet, 1 NACL per subnet, process traffic before gets to instance, applied to all resource behind
  - Stateless- return traffic must be explicitly allowed, allow and denies, eval in order
  - All deny immutable rule
  - Ephemeral Ports- short lived, unique to OS, for clients interacting with subnets
- SG - car security, object based filters, scoped to VPC, default SG, will only protect in or out, not within, attached to ENIs not instances resources can have multiple SG
  - Stateful - return traffic is auto allowed, no order, only allows - matters who initiates
  - Only outbound rule? Can inbound if ec2 starts, only inbound? Can outbound if data starts
  - Can reference group of instances or subnets behind a SG and give access to SG (web access)

Nat - Network Address Translation

- Public IP to sent out packets to internet, for private instances
- Source/Dest checks must be off (NAT instance and ENIs) since drops traffic if source isnt Network Interface
- From private instance to route table to VPC router to Nat back to VPC router to IG to Internet
- Instance problems - not elastic, hard to transition when failure
  - Gateway- AWS managed service, manages NAT translations, scale to 45 GBps
  - cost depends on traffic, cannot associate with SG (yes Subnet) but associate resources behind NG with SG

VPC Endpoints

- Access AWS from VPC, all traffic is on VPC network , by default all access to all S3/ Dynamo
- Regional, VPC DNS req, default endpoint policy is **open**, use SGs, can have mult VPC endpoints with diff policies
- Gateway - S3 and Dynamo, can use with route tables
  - In route table, dest prefix list service (e.g. pl-id s3) target is vpc-endpoint id (e.g. vpce-xyz)
- Interface - AWS PrivateLink, uses ENIs with private Ips,

VPC Peering

- Connectivity between VPCs, for separating resources/ envs
  - Solves compute costs, high availability, operational costs, performance, security, DNS name complications if u used VPN (EC2) software endpoints instead
- No transitive peering, cant have overlapping VPC CIDR ranges
- In route tables each will have dest other CIDR and target peer connect ID (pcx-129348)
- SGs - better practice than before cus we can access (safely) resources in another VPC
- DNS - option to resolve public DNS to private VPC address - if we didn't lots of confusing route table stuff, so enable this option in console (2 options for one VPC to 2nd and 2nd to 1st)

VPC Flow Logs

- For diagnosing issues, captures metadata (not packet data), flow logs attached to ENIs (LB, EC2 etc), subnet, or VPCs
- Can get accepted, rejected, all traffic (or seperate), cant modify after creation, several minute delay (up to 15)
- In cloudwatch (need IAM role, each log group is flow log and ENI is log stream) or S3 (custom logging)
- Reading- version, account id, interface id, source IP (internal primary private IP assoc with ENI), dest IP (same cus logging is at N layer), source Port, dest port, protocol (0-255), packets, bytes, start time, end time, accept, status
  - No data, skip data, ok are status options
- Diagnosing issues
  - Ingress when source IP is not internal VPC,
    - Amazon IP are 10.0.0.0/8, 172.16.0.0/12, 192.168/16
  - Protocol # - 6 is TCP, 1 is ICMP
  - Port # - 22 is SSH, 3389 is Windows RDP, 443 is HTTPS, 80 is HTTP
- Not captured - AWS DNS, license activation, metadata from 169.254.169.254, traffic to/ from that address, aws communication, ENI and NLB traffic
- Can go in to diagnose VPC connection, possibly outbound NCL permissions if not allowing to leave on Ephemeral ports (maybe only port 80 for example)

Network Performance

- HPC - clusters for high perf
  - Faster - same AZ (PGs), bigger instance types, network throughput higher (enhanced networking)
  - Instance types- Elastic Network Adapter (100 GBps), Virtual Function Interface (10 GBps, much less instances)
  - Enhanced Networking
    - Maximum transmission unit is bytes size of the largest packet that can be passed
    - Ethernet Frames - most common is v2 frame format, 1500 MTU, typical internet
    - Jumbo Frames - 9001 MTU, many current sizes while all EC2 can do 1500 (jumbos go back to 1500 when outside aws environment)

Exam tips

- Need ipv4 and optionally ipv6 with VPC- can choose ipv4 but aws assigns ipv6
- Aws reserves first 4 and last of CIDR - implicit router is one
- Main route and custom route tables (main by default, open)
  - Local route and implicit deny by default
- ENIs - primary IPv4, at least 1 SG, can have public ipv4, SGs, private ipv6
  - Can add ENIs but cant remove primary ENI
  - Confined to 1 AZ (and instance), NIC teaming is nono (not better bandwidth)
- Elastic Ips
  - 1:1 relation with private IPv4, own until released, charged for elastic Ips (unless on running instance)
- IG - traffic translator for things in private subnet
- SG - can have multiple, total of 5 to one ENIs, only allow and implicit deny, stateful
- NACLs - firewall for subnets, both allow and deny rules, stateless (explicit rules)
- NAT - disable source/dest check (or else packets dropped), assoc with AZ, add NATs to route tables
- VPC endpoints - gateway - for S3 and Dynamo for when you don't wanna go on internet
- VPC peering - no transitive, better than running software VPN, routes in route tables
- VPC flow logs - metadata, diagnosing, delay (up to 15), gets primary Ips (private not public ENI)
- Network Performance - bigger instances, PGs, Enhanced Networking and Jumbo Frames (in AWS)

DNS and Applications

Monday, November 16, 2020 10:32 AM

DNS
- Enable DNS Resolution and Enable DNS Hostnames
- Records
  - A/AAAA- maps host to IP, CNAME - alias, NS - top level to direct traffic , MX - mail, TXT - text, SOA - start of authority, Pointer- reverse A record, CAA- Certificate Authority Authorization, NAPTR - name authority pointer, SPF - send policy framework, SRV - service locator
  - Alias for AWS constructs
- Requests go to root name server, top level (.com etc) then Domain level (amazon.com etc) then return ip and done
  - DNS resolution- your computer checks locally, forwards request, then waits for response

Route 53 - highly scalable DNS service
- Server Zone file - to direct traffic to your DNS record/entry
- Health Checks- for failovers, can monitor endpoints, health checks, cloudwatch alarms
- Private Hosted Zones - routed in VPC (VPC is on the line between public, private)
  - Instances in VPC might have public IP and private IP (cant be accessed from public except by VPC peering)
  - Any VPCs in PHZ can communicate (Console - same acct, CLI- any accts)
- Split DNS - in both zones with same name, more flexibility with resolution,**overlapping namespaces**

VPC DHCP- dynamic host configuration protocol, one per VPC
- Auto assigns things to devices - basically client asks server for configs and possibly confirmed, everyone acknowledges
- Everyone needs new Ips after a while (TTL)
- Requests made on UDP port 67(server) / 68 (client), VPC can have one DHCP option set at a time
  - DHCP option is [code (type of gift)] [num octets to store][vendor data to store]
    - Common options - 1 - subnet mask (IP), 3 - router, 4 - time server, 6- DNS servers, 7 - logging server
- Reserved IP - .0 - network, .1 VPC router, .2 Reserved for DNS, .3 for future, .255- broadcast address

LBs
- Support HTTP, HTTPS, TCP, TLS to EC2, Have static DNS names (easier to configure), supports health checks
  - Classic - Layer 4 and 7
  - ALB - layer 7, single point of contact, based on **requests, path based routing**, containerized, health of service
  - Network - layer 4, TCP connect (vs ALB HTTPS), scales better, support ^ stuff, target groups, can't do sticky, **UDP**
- All can do X forwarded for but Network doesn't need it
- NLB does webSockets (with stickies)
- Subnets must be public, must be /27 or larger, must have at least 8 available IP addresses
- **Name.region.elb.amazon.com** created for DNS A record
- Cross zone lone balancing - no need to have equal instances in Azs - always enabled (not for NLB)
- Listeners - waits for connection request - has a protocol and port number (can be done for back/front)
- Targets - for app and Network
  - Destinations for traffic (targets/ groups) - ec2 (to private Ips, can be autoscaling group), IP addresses (AWS 10/100/172/192 or subnets of VPC of target group), lambda functions
- Health checks
  - Initial - registering target, Healthy, Unhealthy, Unused - not registered with target group/not in a listener/AZ not enabled, Draining - draining connections
- SSL/TLS
  - Offloading
  - SNI - server name indicator or have multiple certificates per listener - cant do with classic
  - ACM - manage and rotate certificates for SSL (X.509)
  - SSL/TLS termination encryption done at ELB level not EC2 (not end to end, if u want u have to do yourself)
- How to migrate - make new LB, with target group,

CloudFront
- CDN service, regional edge locations (bigger)
- DNS - randomid-afhaskf.cloudfront.net then use a CNAME to make it prettier
- Log files can be enabled, can choose regions, only 1 geoconstriction
- Behaviors - route requests to cache/not cache
  - Default - expression path (*), origin, Http/https/redirect http to https, methods (get, head), field level encryption, cache mech based on header, cache ttl based on headers, TTL, forward cookies, forward strings, signed URLs, lambda at edge
  - For sectioned S3 bucket need multp behaviors
- Signed URLs - same as S3, use from IP addresses at certain times, signed cookies for parts of website/ groups files, origin access identities - S3 origin only this user can use
- HTTPS in cloudFront - free certificate or bring own
  - 502 for bad/ expired/ self-signed certificate
  - With ELB need to use ACM, if not need trusted cert provider

Endpoints
- Privatelink - same region, connect through aws network, consumer VPC gets from provider VPC
  - Uses - connect shared services, connect business partners, more granular access between apps in **different VPCs,** customers in **same AZ as LB** can use, consumer access marketplace/cloud services
  - Has IAM roles, users, aws accts, access to resources, CIDR can overlap, only so many NLBs is cap, one way communication
  - VPC peering doesn't, resources to resources, CIDR cant overlap, 125 peered VPCs, 2 way communication

Lambda
- Can run from customer VPC, your VPC to keep it private
- Have shared ENIs per AZ for lambda to use, SGs have a limit for inbound rules for lambda access, lambda needs NAT to use internet
- @edge - use for A/B testing and inspect cookies, test header requests, generate HTTP responses, modify headers all before hitting cloudFront

Workspaces
- In 2+ Azs (2+ subnets), AWS has VPC using directory service (to manage users for workspaces), captures pc over ip data, which connects to auth gateway and actual workspaces infra on AWS VPC too
- Directory service reserves 1 IP, subnet size limits, **required**
- Workspaces CIDRs - 172.31/16, 192.168/16, 198.19/16
- Each workspace has network interface in one of subnets (2+ needed), on prem or directory service, SG with ENI
- AWS Managed Microsoft AD - easilt managed, simple AD - 500 small or 5000 large, AD Connector - redirect between directories

AppStream 2.0
- Stream an app without using hardware, they just send u pixels - like IDE or web browser or fancy web app , great for gpu intense
- Need html 5 compatible browser, at least 1 subnet to run in, dedicated VMs for each new user,
- For enterprise apps (centralized update/ manage apps), design/ engineering (gpu intense tools), online training/trials (don't have to download app)
- In VPC in AZ, should be multi AZ

Tips
- enableDNSHostnames will auto assign (public) dns hostnames to ec2 instances
- CNAME vs ALIAS
  - ALIAS routes traffic to AWS (CloudFront, Elastic Bean Stalk, S3 buckets, ELB)
  - CNAMES not for zone-apex domains, for sub-domains (www)- treat as redirect
- LB health checks - HTTP, HTTPS for ALB, that plus TCP for NLB
- Can do blue/green with multiple records and weight gradually to green
- TCP for end to end encryption (NLB or CLB)
- To deliver different content to diff devices - Host based routing with ALB or User-Agent with lambda@edge w/Cloudfront

Hybrids

Friday, November 20, 2020 4:13 PM

Gateways

- Private gateway - connect **on prem to VPC**, managed, act as router from other cloud to VPC, one VPG per VPC
  - Works with site to site or DX connection
- ASN- autonomous system numbers- part of VPG config
  - VPG need private ASNs, default is 64512 in all regions, needed for BGP route learning, VPG cant be edited
- Route learning- static - manual put in, limited means of making preferences
  - Dynamic - share knowledge via routing protocol, lots of controls
  - Route learning for hybrid - site to site can do either but DirectConnect only dynamic
  - Route propagation - if VPG learns route, can add to route table
  - Most specific route (longest CIDR prefix) is preferred when conflict on table, but local VPC route is preferred when overlapping routes are getting propagated, static routes preferred to matching (e.g. same CIDR but diff targets)
    - Other - Direct > static > BGP VPN
  - BGP - Border Gateway Protocol- exterior protocol between systems, works between Autonomous Systems
    - network routes (prefixes) shared between **manual** config peers, no prefixes auto shared, selects best prefix of multiple paths, TCP 179
    - Interior Gateway Protocol - BGP learns from this to make decisions
    - Shares route availability with other subnetworks- BGP peers between and can BGP endpoints within network
      - eBGP (between AS), iBGP (in AS but BGP), IGP (in AS not BGP)
    - Used by Direct Connect, BGP only sees peer connections
    - Prefixes- which road to take from router to network, picks best only- **every** possible route to internet
      - Network (going to), Next Hop (next router), Metric, LocPref, Weight, Path (to dest, either I or list AS#)
        - Starts with first connection- def is Network IP, 0.0.0.0, 0, 100, 32768 (this router is directly connected to end network), I (internal) for a BGP connection defining its own network
        - Then can learn from other BGP connections - say there is another on this AS, that entry would be say Network IP (same), Router 2 (hop is no longer local), 0 , 100, 0 (learned), I (still internal)
      - Prefix Pref - Highest weight, local pref, shortest AS path, Lowest Metric
      - Routers can only learn from BGP best prefixes (only from BGP connected to their network)
    - VPG auto shared with all BGP it works with
    - Can manipulate BGP prefixes since it cant see network quality
      - Change weight of preferred to higher than def (>32768), weights are local only (unique to each router)
      - Local pref is for when have lots of BGPs in one AS and to set preference for which local router - diff from weight cus shared with iBGP peers
      - Weight is checked first (so change weight then local pref)
      - For aws things- cant connect to BGP table (since managed service) - so change path prepending
        - Make AS path longer for the bad router (so it wont be chosen by VPG), can be forwarded so…
        - Metric/ Multi Exit Discriminator - increase metric for worse route
    - Cloudhub- VPG using BGP to share info

- Quiz notes
  - AWS VPN Client by default only allows traffic through AWS (can be set to do both AWS and local w/split -tunnel)
  - Site to site Client uses Site to site tunnels (2 endpoints), supports ipsec and ipv4
  - OpenVpn is client vpn (not site to site)

VPN

- Isolated VPN tunnel networks - site-to-site connects endpoints between offices **(IPv4, IPSec tunnel)**, or client to site
- IPSec pre config- needs identity of other endpoint, shared auth method, security policies, what traffic that creates tunnel
  - Process-
    - Server detects traffic
    - Internet key exchange- **asymmetric** - phase 1 (Main mode) - VPN endpoints secure space, negotiate security policy (encryption mechs, types, etc.), gen encryption key, perform key exchange (Diffie-Hellman - multi part key decryption using shared part of key pair and pre shared string), encrypted authentication (of router)
    - Phase 2 - **symmetric -** Ipsec - quick mode, don't need to reauth, detect traffic, fresh keys, creates 2 one way tunnels
    - Tunnel created then terminated, eventual timeout and need to be recreated
  - Security Association- relationship where both parties share security settings
- Policy Based - admin config policies/rulesets to set security, one IPsec assoc per matched rule set
  - If sec policy of phase 2 tunnel doesn't match VPN, they would have to create another phase 2 tunnel (3 total)
- Route based - traffic must target dest network, one IPSec for all traffic
- Tunnels established by traffic from on prem to AWS (not from AWS) **customer side**
- AWS VPN tunnels can only support a **single pair of IPSec assoc** (no normal policy based)
- Customer Gateways
  - Site to site VPN setup- config VPG (AWS), config Customer Gateway Device in customer location, Configure CGW (customer location), configure VPN connection, VPC route tables (static/dynamic), VPN on CGW
  - VPG - **2 endpoints in 2 AZs** for CGW to attach to - active, passive so 1 at a time
  - CGW Device reqs - support IKE v1/v1, support IPSec, accessible by static ipv4, Deed Peer detection, BGP support optional
    - Must have ports on VPN open **UDP 500**, **IP Protocol 50** and if using NAT- Traversal- **UDP 4500** too
  - CGW config - name tag, dynamic or static routing (need ASN of CGD if dynamic), CGD public IP (or NAT IP), optional assign ACM certificate for IKE (if not auto gen by AWS)
  - VPN connection config - name, VPG ID, existing CGW ID or create new (default dynamic routing), uses dynamic/static, tunnel options - all can be changed later
    - Tunnel - IP CIDRs (size /30 - 2) and pre shared keys for tunnels (rem 2), only for dynamic, algos for encrypt, IKE versions, lifetime, etc, can view VPN connection rates
  - VPN config on CGW- download config from console, choose vendor/platform/software, define connection to both VPG endpoints (outside - internet GCW VPG, inside - BGP tunnel for both 169.254)
- Lab
  - **Need to do this later…**
- VPG and VPN limitations
  - VPG- not VPC transitive (cant reach VPC peer from VPG), single endpoint used at a time
  - VPN- throughput capped at 1.25 Gbps over all connections, each VPN tunnel supports only 1 SA pair, only IPSec/ IPv4
- Other VPN Options
  - Software VPN- EC2 with IG, VPN software installed (AMIs in AWS marketplace), good for org requirements, advanced
    - Fixes all 7 problems, source dest check needs to be disabled, downside is unmanaged
    - Other uses for VPN in VPC vs aws traffic - point to point traffic isolation, multicasting
  - Client VPN server- openVPN service, AWS managed, creates VPN endpoint (free) for individual client systems, client authentication via Active Directory or private certificates (ACM), controlled access to a VPC
    - To allow client to access VPC and control access to internet
    - Route controls, SG control, Authorization rules (CIDR range, access to)
    - Split tunneling - client and VPN route tables r useable, for less strict client to VPC connections
- VPN monitoring/ optim
  - Cloudwatch (does it work?) - TunnelState, TunnelDataIn, TunnelDataOut, Network metrics, Egress/Ingress, authentication failures, packets in/out, VPC flow logs, EC2 log streams
  - Faster- throughput by AWS for VPN, use EC2 VPN bigger sizes/types, enhanced network (if works), parallel EC2
  - Fixing- Route tables, Sgs, NACLs, Authentication, CG Devices, OpenVPN client config
    - Permissions - IAM or resource level, Infrastructure (such as CG Device)
  - Cost - aws managed higher avail, cheaper, Client VPN more expensive but no infra, EC2 half cheap, full config but so much work, AWS cheapest but STS and need Client Device

Direct Connect & Hybrid DNS

Tuesday, December 1, 2020 5:32 PM

Intro
- VPN is public infra, out of internet data rates, limited to 1.25 Gbps
- DX- always on, dedicated infrastructure, max speed 10 Gb, mult connections can be made
- Hybrid DNS- private DNS on cloud network

DX
- Has local connect locations like edge locations, traffic is isolated on VLANs, uses this connection to reach AWS for a region
- **On prem** Config reqs- single-mode fiber, disable auto-neg on all ports, 802.1Q VLAN must be supported, **BGP auth**
- Dedicated Connection - customer managed hardware (at DX location)
  - 1 Gbps or **10 Gbps connection** to region, 10 dedicated connections per region (soft), can be done on Console, CLI, API, need (Name, Dx Location, port speed, Direct Connect Partner), can only modify tags
  - Can use mult up to **40 Gbps**
  - Download Letter of Auth, valid for 90 days
  - Direct Connect Location cross-connects your hardware, must be single fiber and use weird bases (10Gbase)
- Hosted - **aws partner** manages hardware
  - Support wider range of bandwidths, at certain locations though, they have their own APIs, faster established

VIFs
- DX connections r layer 2 (need layer 3 for IP), so we need VIF
- Private VIF- connect to private aws, **can attach to 1 VPG** or single Direct Connect Gateway (up to 10 VPCs diff regions)
- Transit VIF- DX through Transit Gateway
- Public VIF- connect to all public services in public regions without internet
- Each VIF runs on own VLAN
- Lots of congif stuff, only tags modified after created, router config file
- Up to 50 public/private VIFs, 1 transit VIF, **hosted** only supports single VIF
- Hosted VIF- DX connection owned by AWS acct can be used by another owner, any VIF type
- BGP
  - All VIFs must be connected to BGP peer, IPv4 and 6 supported, MD5 auth,
  - Max prefixes advertise to Aws - 100 for private and 1000 for public
    - If using VIF all prefixes advertised, if using DX gateways specified prefixes must be advertised (by cidr or prefix)
  - Public Vifs - all prefixes advertised
  - BGP communities- tag/label, routers can be config to handle in/out prefixes based on community
    - Made of ASN and org value, "no_export" comm on things says cant export given communities
    - 7224:8100 - same region as DX, 7224:8200 - same continent as DX, No value - routes to global services
    - Customer specifies which prefix advertised to AWS in public VIF prop, use BGP comm to say where AWS can send customer prefixes (local- 9100, continent- 9200, all public - 9300, no_export, no val = 9300)

VLAN
- Virtual LANs - L2 readable way to identify traffic belonging to L3 network, to isolate traffic in switched networks, each VLAN represents diff IP subnet, must be assigned own VLAN ID
- Puts another layer around 802 on top of customer frame
- Id is customer defined, must be unique, should be consistent
- Ports
  - With single VLAN is access/ untagged port, traffic entering through VLAN port can only exit through VLAN port (many ports per VLAN), can extend across devices, ports with 2+ VLANs are tagged/trunk port
  - Untagged- standard ethernet frame VLAN id is from other port, tagged uses 802 ethernet, VLAN id is in tag
  - Hosted so one VIF but want multi VLAN- can make multi DX connections, aggregated VLANs (tags with more tags)

Lag- Link Aggregation Group
- Distributed to member links- more throughput, more resiliency,
- Must be same bandwidth and DX loc
- VIFs can attach to LAG instead of DX, add LAG primary port to VLAN

Direct Connect Gateways
- Fglobal services faacilitate DX connectivity, bridge between VIFs and AWS, don't work with public, free
- Up to 10 VPG and up to 30 private VIFs for a gateway
- Global service, assoc VPGs to DCG to allow for cheaper access to other regions
- VPCs cant have overlapping IP ranges, only one VIF to a VPC at a time, cant send traffic from one assoc VPC/ Vif to another, can be used by VGWs in different aws acct, VPG needs a VPC, VIF and VGW can only be on a single gateway
- Config with name and ASN, attach to VIF, associate with VPG of VPC

Well Architectured Direct Connect
- Resiliency
  - Multiple DX connections - mult at single loc, mult at diff, up to 4 DX in a LAG
  - Failover to vpn (can be automated with AMIs)
- Failure Detection-
  - routing protocols not quick - BGP peer timeout default is 270 seconds before detection
  - Bidirectional Forwarding Detection- detect link failures in ms, informs assoc protocols then they recalc
    - Auto enabled on VIFs, must be configured at customer devices (how often to test, how many tests)
- BGP communities - use with mult DX connections to control prefix pref (conf load balancing, failover) Private and Transit VIFs
  - Comm values let us only use high pref prefixes
- Performance
  - Lags- up to 40 Gbps (4*10)
  - Jumbo frames- private VIFs sets MTU from 1500 to 9001 - more data in less frames (enabling/disabling lags 30 secs)
    - Support must be enabled end to end, nested VLANs can cause jumbo (extra data from customer data), only dynam
- Sec
  - DX traffic not encrypted, you can use VPN tunnels (public VIF is site to site, private is EC2 hosted)
- Cost - Port hours (flat rate) and Data out (out of AWS region)
- Monitoring - connection, in/out Bps, Packets per second, CRCerrorcount, quality of fiber, **no VIF CW metrics**

Hybrid DNS
- Support private resolution over hybrid networks, - can use AWS resources in prem DNS zones, on prem resources can use VPC
- Route 53 resolver- provides default DNS **in VPC** (VPC + 2 IP address), part of EC2
  - When receives query - resolve at route 53 private zones, VPC DNS domain, public DNS in order
- Route 53 private hosted zones must be assoc with VPCs
- Before use EC2 instances to DNS resolve - limit 1024 queries per second, high availability is annoying (DNS cant load balance)
- Resolver endpoints - ip accessible endpoints to resolver - 2 to 8 ENIs - up to 10k queries
  - Endpoints in single VPC, can be used by others in region
  - All ENIs in one SG - group cannot be changes, ENI is either in or out, and has AZ scope, can use dynamic of customer IP (subresource), pricing per ENI per hour
    - Outbound traffic - forward rules send requests to ipv4 of on prem DNS, system rules send to route 53 resolver
      - Def system rules- private hosted zones, VPC domain names, public reserved domain names
      - Forward over system preferred and more specific DNS name pref

Transitive Networking

Thursday, December 3, 2020 7:29 PM

Intro
- When traffic between networks goes through additional
- Nodes r individual networks (router/switch or entire networks)
- Why isnt this in Aws- network packets outside VPC but have Network Interface Card with VPC as destination or dropped
- VPG cant access from on prem. VPC endpoints work for interface VPC endpoints from on prem, cant reach gateway endpoints (dynamo or s3)

VPC Sharing
- Needs aws org, subnets shared via aws resources, only owner acct can manage
- Transit VPC- EC2 hosted VPN solution in VPC, needs an internet gateway, for each spoke VPC ec2 VPN works as transitive, can make more for higher avail **central VPC**
    - Good to have dynamic routing so VPCs can learn about good prefixes got network, route is over VPN

Transit VPCs
- Hybrid (on prem) - connect VPNs as spoke,
- VPGs only go in, don't send out info (not useful for advertising in transitive)
- Direct Connect
    - Public VIF is for AWS public services
    - Private VIF- single VPG (single VPC), basically a VPN so can only go in
        - Can connect to Direct Connect Gateway (up to 10 VPG in public region)
        - To add DX to transitive network, don't attach it **(floating VPG) -** now can forward traffic
            - Has private VIF
            - If DX to spoke, could add private VIF to VPG of spokes and DCG can connect 2+ spokes
- Max MTU on these connections - to improve performance
    - If MTU too big, will fragment unless Do not fragment flag is on
    - 1500 B is default, Public VIF, VPC Peering, internet
    - 9001 for VPC, Dx Private VIF, 8500 for DX Transit VIF (carries extra header)

Transit Gateway
- Transit hub for many VPCs, mult route tables, attach to VPCs, VPN Customer Gateway (becomes gateway), DX Gateway
- Can peer with Transit Gateways (in other regions), all properties optional, uses attachments to connect to ^
    - Hourly rate per region, must be attached to subnet in AZ, choose either static or dynamic
- Creation- all properties optional, only tags can be changed, 5 TG per region
- VPN attachments to Transit require customer gateway - VPN attachment creates VPN connection (in VPC service)
- DC Gateways- reqs a transit VIF - DCG and TG must be different ASNs
- Route tables- up to 20, defines attachments, dest CIDr range and TGW ID
    - Each object attached is associated to single route table
    - Can propagate CIDR to TG route tables, VPCs propagate local CIDR (not customer), while CG, DCG, VPN propagate customer prefixes
    - Static- only one default, can blacklist, for peered TGWs
    - Routes from VPCs to TGW networks must be manually added but VPN CG is learned from BGP
    - Prefixes from DCG are specified when attaching to Transit
- Has Cloudwatch for bytes/packets, has flow logs within VPCs
- Charged per partial hour (owner of each VPC/TG/DX), and charge per GB processed by TGW (not peering)
- Routing
    - Propagations- auto add prefixes to route table
    - Associations - use route table to determine traffic dest
    - If we don't want communic- isolated VPCs with restricted attachments, make new route table and remove attachment from TG main route table first- still need to attach to main (so main can talk to it)
        - Also add VPCs to talk to
    - Static > propogated and VPC > DX > VPN for route conflicts
- Size limits
    - VPC peering - 50 VPC peering connects, 2 VPCs, routes manually added
    - Endpoint - 55000 simultaneous connections
    - Site2Site- per region- 5 VPG and 10 VPN per VPG
    - Transit VPCs- VPN platform limits, VPC route table 50 routes non prop, 100 prop, 100 dynamic
    - DX Gateway - 30 Private VIFs and 10 VPG, 30 Transit VIFs and 3 Transit TG
    - Transit Gateway - 5000 attachments, 20 DX Gateways, 50 peering attachmnets (can attach TGws)
        - 20 route tables and 10,000 static routes

Security

Monday, December 7, 2020 1:25 PM

Shield

- ffff

Questions

Monday, June 14, 2021 5:38 PM

- Performance
  - Use Enhanced Networking on instances to make them faster (LoadB for using multiple)
  - 1500 MTUs between regions, 9001 inside
- Security
  - In transit encryption - use HTTPS (TLS) with aws:SecureTransport condition for S3 buckets to guarantee encryption during transit
  - S3 - buckets have bucket policies
  - Cant use CIDR if isnt in RFC 1918 (10.0, 172.16, 192.168) https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Subnets.html#add-cidr-block-restrictions
- VPC
  - NAT gateways aren't used with VPC peering