**INTRO**

Availablilty zone - one or more discrete data centers

Data centers - each with redundant power, networking and connectivity, housed in seperate facilities

Edge locations - for caching contents. consists of CloudFront, Amazon's Content Delivery Network

**IAM - 중요도 下**

IAM is universial. regions 상관 x

IAM : users / groups / roles / policies(JSON)

Root accounts -> complete admin access

New Users have no permissions when first created

\*\*\*New Users are assigned Access Key ID & Secret Access Keys

-> 얘네는 only for programmatic access (console X)

-> can be viewed only once. 잃어버리면 should be sregenerated

root account는 MFA 활성화하자 - google authenticator app

password rotation policies

**S3 - 중요도 最上**

S3 - object based / 0바이트부터 5tb / unlimited storage / stored in buckets / \*universal namespace

\* db나 os용이 아니야. 그냥 file만

http 200 코드 for successful.

삭제 시 MFA 켤 수 있다.

\*\* FUNDAMENTALS :

key(name of obj) / value(data) / version id (for versioning) / metadata (data about the data) / subresources

\* Read after Write consistency for PUTS

\* Eventual Consistency for overwrite PUTS and DELETES (propagation 좀 필요)

\*\*\* DIFFERENT TIERS

(1) s3 standard(expensive) /

(2) s3IA(infrequent& rapid retrieval) /

(3) RRS == S3 one zone IA (lower cost of s3ia)

(4) s3 intelli(ai) /

(5) s3 glacier (low cost + slow retrieval) /

(6) s3 glacier deep(super slow retrieval)

s3 bucket -> 이름은 global. should be unique

\* bucket policies & access control lists(ACL) 활용하여 접근 설정 가능

항상 log 남기도록 configure 가능

can be moved across region using cross region replication

can change storage class(6 dif) & encryption

\*transfer accerlation: region간 이동 시, edge locations에 먼저 이동시킨 후, edge에서 중심부로, 중심부에서 goal로 가는게 개빠름

\*bucket policies -> entire bucket

\*object -> entire individual files in bucket

\*IAM policies & Group 설정해서 접근 권한 설정 가능

s3 price:

S3 standard -> S3 IA -> S3 Intelligent Tiering (약간의 모니터링비용 있긴 함) ->

onezone ia -> glacier -> glacier deep archive

\* obj가 너무 많으면 intelligent 비용 많아짐. IA와 잘 비교해서 쓰면 될듯.

\* 용감한 사람이면 onezone도 좋아

s3 security

Encryption in Transit -> SSL / TLS

- serverside 3개, client 1개.

-> serverside는 대신 관리해주는 sse-s3와 / aws kms service가 관리해주는 sse-kms /

고객이 알아서 관리하는 SSE-C

-> \* individual lvl 뿐 아니라 bucket level - encrpytion available.

s3 versioning

all versions of an object -> good backup tool

\*cannot be disabled, only suspended

s3 lifecycle

automate moving obj. btw dif. storage tiers

conjunction with versioning -> curr & non-curr 각각 설정 가능

s3 obj lock

\* s3 obj lock for WORM (store once, read many)

-> individual obj & bucket 각각 적용 가능

-> \*governance mode -> special permission 있으면 조작 가능 // compliance mode -> 심지어 root도 조작 불가. (조작 = delete, update)

\* s3 glacier vault lock -> s3 glacier 차원에서 compliance ctrl 쉽게 조작할 수 있게 해줌

-> specify controls like WORM in Vault Lock policy & 그 policy를 lock

s3 performance

\* prefixes -> half name / prefix당 리퀘량 정해져있어서 많이쓰는게 이득

3500 put copy pose del / 5500 get head per sec per prefix.

SSE-KMS encryption은 kms api limit 땜에 리퀘량 limited

\* Multipart upload -> upload 시 여러개로 쪼개서

-> recommended over 100mb / required over 5gb

\* s3 byte-range fetches -> download시 여러개로 쪼개서.

s3 select & glacier select

\* s3에서 sql문 활용하여 곧바로 데이터에 접근할 수 있다.

-> retrieve only a subset of data (only rows || columns)

performance up / cost down

glacier에도 비슷한 기능.

AWS organizations & consolidated billing

aws organization

enable mfa on root

do not deploy anything on paying account. Only for billing

service control policies (SCP) 는 ou / individual 각각 적용 가능

\*

organization에 친구 추가

iam에서 새로운 권한 만들고 링크(a) 복사

새로운 계정에서 role만들어서 그 계정의 iam으로 접속 (root 피하려고 그런듯)

새로운 계정 iam에서 (a) paste하면 role 전환 가능

https://signin.aws.amazon.com/switchrole?roleName=s3-cross-account&account=daeyong2021

https://093599856640.signin.aws.amazon.com/console

s3 bucket across accounts

using bucket policies+ IAM => entire bucket & only programmatic

using bucket ACLs + IAM => indiv. obj & only programmtic

cross-account IAM Role => Programmatic + Console

\*\* cross region replication

\* destination bucket도 versioning 켜져있어야 한다. (both src, dest)

\* replication only affects new objects & deleting version이나 delete markers not replicated

\* src property 바꿔도 dest에 적용되지는 않는다.

s3 transfer acceleration

특정 region의 s3에 직접 업로드하는 것보다 edge location을 거쳐서 가는 것이 보통 더 빠름 = transfer acceleration

AWS Datasync

\* on-premise용. -> large amounts of data / nfs- smb- compatible file sys./ replication hourly, daily, weekly

install DataSync agent.

can be used to replicate EFS to EFS (ex. ec2 -> ec2)

CloudFront: Amazon의 CDN. large file을 위해 내가 직접 edge location의 caching을 이용하는 것

Edge location -> where content will be cached. (seperate to AWS region / AZ)

-> NOT read only. can be wrote (지금까지는 read에만 썼잖아.)

-> caching : TTL (time to live)

-> \* clear caching will be charged "Invalidating the cache"

Origin -> origin of the files that CDN will distribute. (s3, ec2, elb, route53 ..)

Distribution -> CDN which consists of a collection of Edge locations

web location -> distribution for websites

RTMP -> for media streaming

CloudFront Signed URLS & Cookies

only for authorized users

\* 1 file = 1 URL / \* 1 cookie = multiple files

EC2가 origin이면 use cloudfront.

s3 signed와의 차이는, s3에 직접 접근 여부. cloudfront 사용하면 OAI를 통해 인가된 사용자만.

snowball

매우 큰 크기 (pb수준)의 데이터 전송을 위해. import to/ export from s3

Storage Gateway

-> s3보다 빠르게 작동. iSCSI protocol. + physical appliances 가능

File Gateway : for flat files, stored directly on s3.

Volume Gateway

Stored Volumes -> entire dataset을 로컬(site)에 저장. & async backed up to s3

Cached Volumes -> Most freq. data만 로컬에 저장 & entire on s3

Gateway Virtual Tape Library

\*\*\* Athena vs Macie

athena - interactive query service, query data in s3 using SQL(usually for log data), serverless

macie - ai to analyze data & identify PII. / something suspicious, criminal logs. / reports & alerting

lab에서 supersuper important -> IAM의 policy -> inline으로 설정한 Deny는 어떠한 설정도 overwrite한다. "Explicit Deny"

"Least privilege"

iam에서 그룹 생성 & 유저 생성. 그룹의 policy 설정. 유저를 그룹에 추가. 유저로 로그인. 권한 제대로 제한되는지 체크

# Q) Power User Access allows Access to all AWS services except the management of groups and users within IAM.

**EC2 - 중요도 上**

EC2

resizable compute capacity & reduce time to boot new server & 설정 변화에 빠르게 대응한다.

On Demand(쓴만큼만 바로바로) / Reserved(1,3년 텀으로 계약) / Spot(남는거 굽신굽신) / Dedicated Hosts(실제 피지컬한거 줌)

EC2 Hypervisor – Xen, Nitro

EC2는 AZ에 생성되는 것이다. == subnet

EC2-lab

yum install httpd (<- 아파치) -> index.html만들어서 /var/www/html/에 넣기

termination protection은 디폴트가 turned off

Root EBS는 delete on termination이 디폴트, 추가 EBS는 설정 X

\*Root와 추가 EBS volume encryption은 이제 생성 단계에서 설정 가능하다

추가 volume -> cold hard가 가장 싼거. Throughput이 data warehouse용

Security Group

All inbound traffic is block by default & all outbound traffic is allowed

rule은 바로 적용된다.

다만 개별 ip나 개별 포트만 block할 순 없다. 하려면 Network Access Control List 사용할 것

ec2에 security group 여러개 가능. 하나의 security group에 여러개 ec2 가능

\* Security Group = Stateful. inbound 설정하면 알아서 outbound 열어준다.

EBS: virtual hard disk

주의) “Block based storage”를 사용한다. (\*S3는 object based)

종류

gp2(general purpose - 대부분) / io1(Provisioned - 최고성능, DB에 사용) / \*st1(throughput optimized - iops낮고 싸서 빅데이터 – 주의, 빅데이터라고 high perf. 아냐) /

sc1(cold hdd - 제일 싸고 느려서 file server) / standard(EBS magnetic - infrequent data. (s3 glacier와 비슷))

\*\* EBS volumes AZ = EC2 instance AZ

EBS snapshot 찍기

Termination Protection은 디폴트로 꺼져있다.

따로 설정 안해줬으면 root만 지우면 다른 ebs는 persists.

Volume exists on EBS / Snapshots exists on S3

\* Snapshots are incremental -> 가장 최신의 snapshot만 유지된다.

Snapshot 찍으려면 인스턴스 멈춰라.

can create AMI from snapshots(아래 lab 내용)

\*주의) registered AMI의 EBS snapshot만 지울 수는 없다. 먼저 AMI부터 지워라.

\*주의) EBS snapshot 생성, 공유 복사 등등 API, CLI, Console에서 다 가능하다.

EBS lab az 변경

ebs 종류는 바로 modifiable

ebs az 변경하기

1. 원하는 ebs에 가서 snapshot 생성 // 2. 해당 snapshot으로 image 생성 // 3. 해당 image를 launch한 뒤 다른 az(subnet)으로 ec2생성

2번 후에 copy img를 다른 region에 하면 다른 region으로도 이전 가능하다.

\*\* AMIs

Instance Store Volumes = "Ephemeral(수명이 짧은) Storage" VS EBS backed instances (요게 우리가 쓰는거)

\* Instance store Volume는 멈출수 없고 host dependent하다. 걔가 fail, 나도 fail.

EBS backed는 멈출 수 있고 데이터 안날라가. instance store과 ebs모두 reboot는 가능

둘다 디폴트로 root volume은 terminate되지만, EBS의 경우 terminate하지 않게 설정 가능하다.

\*\*\* ENI vs ENA vs EFA

[senario] EC2 network workload very high, needs 50 gbs, which is better : multiple ENIs and ENA? → 무조건 ENA. Multiple ENI doesn’t speed up network + ENA is always btr than VF

ENI: basic networking. Separate network at low cost

EN: needs reliable / high throughput, 10~100Gbps

Elastic Fabric Adapter: High Performance Computing이나 ML, OS by-pass.

\*\*\*Encrypted Root Device Volumes & Snapshots

snapshot만들고 → copy snapshot에서 encrypted로 복사 → image 만들고 ami로 인스턴스

\* snapshot of encrypted volumes are encrypted automatically

snapshot can be share only if it’s unencrypted

\*\*\*이제는 creation of ec2단계에서 can be encrypted

Spot instances

90%까지 cost 줄일 수 있다.

Persistent storage나 persistnet network workload가 아닌 경우 사용한다.

Spot block 활용해서 threshold 넘어도 바로 꺼지지는 않게

Spot Fleet는 4가지 strategy로 적절한 instance(on-demand도 가능) 배당해준다.

EC2 Hibernate

\*RAM을 preserve. → hibernate 끝내면 reboot가 필요없다. → init이 오래걸리는 경우 쓰면 좋겠지.

Ram >= 150GB

60일이 최대

\*\*On-Demand instances & Reserved instances에만 적용할 수 있다.

\*\*\*CloudWatch (vs CloudTrail): Monitoring performance

AWS 대부분의 활동을 monitor 할 수 있다. + 5분 텀이 디폴트인데, 1분(detailed)으로도 가능

알람 설정 가능 (billing alarm과 비슷한 로직)

\*\*\*Cloudwatch는 performance / Cloudtrail은 기록(API call auditing), “누가”에 초점.

[senario] 누가 ec2를 사용하고 있는지 잡아라! → Cloudtrail / 지금 ec2의 network throughput이나 disk io 살펴봐라! → cloudwatch

Cloudwatch lab

cloudwatch dashboard 새로 생성 가능하다. Global / region 다 가능

Alarms - threshold에 다다랐는지 알람 켜줌

Events – respond to state changes / Logs - 로그들 관리, 확인 가능하다

CLI: command line을 통해 global하게 접근 가능

IAM에서 접근 가능한 user 생성해야하고, 그 access id와 key로 접근한다.

Aws s3 ls, aws s3 mb mybucket 등 다양한 명령어 알아두면 좋을듯! 시험엔 X

IAM Roles

Roles are more secure than access key / secret access key 공개하는 것보다.

Roles are easier to manage - 수천개의 secret key를 업데이트 하는것보다 롤 하나만 바꾸면 끝

Roles are universial = any region

console, cli로 모두 설정 가능

\* EC2 instance에 붙인 IAM Role의 policy는 곧바로 적용된다.

\* Role 없는 ec2에는 stop이든 running이든 바로 적용 가능 / 이미 있는 role replace하려면 무조건 running 상태여야 한다.

Instance Metadata

script 확인하기 → curl <http://169.254.169.254/latest/>{meta-data 나 user-data 등}

EFS: ec2간 파일 공유 시스템

EFS 생성 → ec2 instance A와 B 오픈 → security group에서 inbound NFS 열어주기

→ efs amount → 위치 설정해주면 자동으로 해당 security group내 인스턴스들 sync된다.

Supports NFSv4 protocol

No pre-provisioning = pay as we go

개당 scale up to pb, thousands of NFS connections 지원

region 내 multiple AZ내에 저장 가능하다.

Read After Write Consistency

Amazon FSx

[senario] SMB이거나 Microsoft app이면 FSx / Linux이고 NFS면 EFS

\*\*EFS vs AmazonFSx for Windows vs AmazonFSx for Lustre

EFS: NFS / Linux / resilient storage

AmazonFSx for Windows: SMB / centralized storage for Window apps / Microsoft apps

AmazonFSx for Lustre: Big data / high performance / \*얘는 directly S3에 저장가능하다

\*\*\*EC2 Placement Groups: 성능/안정성 등을 위해 인스턴스들을 그룹화하는 전략!

Clustered placement group: low network 지연 / high network throughput / AZ 하나에만

Spread placement group: individual instances들을 분산시켜 보관 → hardware failure 피해 최소화

Partitioned: multiple instances를 묶어서 분산시켜 보관(dif racks), ex HDFS, HBase, Cassandra

- Spread와 Partitioned는 다른 AZ에 span 가능 but Region은 같아야 한다.

- name은 aws 계정 내 unique해야한다.

- 각 group마다 사용 가능한 instance의 종류가 다르다. (Compute Optimized, Storage …)

- Recommended to make homogeneous instances in clustered

- placement group끼리는 merge 불가하다.

- 이미 있는 instance를 group에 넣을 수 있는데, stop 되어있어야 한다. 아직 aws콘솔에서는 지원X

HPC on AWS: High-Performance-Computing

Data transfer

Snowball, Snowmobile → physically copying

AWS DataSync to store on S3, EFS, Fsx ..

Direct Connect (aws랑 직접 private한 연결 만들어준다.)

Compute & Networking

GPU, CPU optimized

EC2 fleets (Spot Instances or Spot Fleets)

Placement groups (cluster)

\*Enhanced Networking single root I/O virtualization (SR-IOV)

Elastic Network Adapters or VF interface - 항상 ENA가 우세

Elastic Fabric Adapters – os bypass 기능

Storage – Instance attached

EBS – scale up to 64000 IOPS with provisioned one

Instance Store: millions of IOPS, low latency

Storage – Network Storage

Amazon S3 – obj. based storage, NOT a file sys.

Amazon EFS – Scale IOPS on total size

Amazon Fxs for Lustre

Orchestration & Automation

AWS Batch

AWS Parallel Cluster

AWS WAF: Web Application Firewall, http, https 리퀘를 모니터링한다.

[senario] malicious IP address나 sql attacks, cross-site scripting 등등 막으려면 어떻게? WAF로.

+ Network ACL 활용한다.

**DATABASES - 중요도 最上**

종류

RDS(OLTP): 단순한 관계형 db - SQL/MySQL/PostgreSQL/Oracle/Aurora/MariaDB

NoSQL - DynamoDB

\*\*\* Red Shift(OLAP) - Online Analytic Processing. DB의 analysis와 monitoring에 초점

Elasticache: caching solution – Memcached, Redis

Red Shift VS Elasticache - Redshift는 Business Intelli.나 Data warehouse로서의 역할, Elasticache는 기존 DB의 성능을 끌어올리는 역할이다.

RDS Lab

\*\*\*RDS는 virtual machine에서 굴러간다 → 해당 머신에 access는 불가 (can’t SSH)

patching RDS OS and DB = amazon’s responsibility

\* RDS is NOT Serverless (Aurora Serverless 빼고)

RDS Backup & Multi-AZ and Read Replicas

RDS Backup:

automated backup(defalut)과 database snapshots(manually)로 나뉜다.

Read Replicas: replica를 생성해서 heavy-read workload에 사용하면 좋다.

Can be Multi-AZ/ 성능 향상이 목적 / Backup should be turned on / 다른 region에 가능 /

mysql, postgre, maria, oracle, aurora 가능 / Read Replica깨고 master로 승격가능

MultiAZ: 여러 availability에 저장한다.

Disaster Recovery가 목적이다. 성능 향상과는 무관. / 리부팅 통해 failover를 강제할 수 있다.

Encryption:

mysql, postgre, maria, aurora, sql server 가능 / KMS 방식 / RDS내 자료, 백업, 스냅샷 등 다 encrypt된다.

DynamoDB

SSD storage에 저장된다.

3개의 distinct data centers에 분산저장된다.

Write-read 텀이 1초 후가 보장되는 Eventual Consistent Reads(default), 1초 이내에는 Strongly Consistent Reads

Redshift

Business Intelligence용도이다.

하나의 AZ에서만 사용할 수 있다.

1 day retention period로 최대 35일까지 Backup 가능

3개의 copy(original, computer node replica, backup in S3)를 유지한다.

Async replicate snapshot to s3 in another region for DR

Aurora: Aws가 개발한 DB.

2 copies each in min. 3 az (최소 6개)

Aurora snapshot with other AWS accounts

3 type = Aurora / MySQL / Postgre replicas 만들 수 있지만 automated failover는 Aurora만 가능

automated backup 켜진게 디폴트. 스냅샷 찍을 수 있고 공유도 가능하다.

\*\*\* Aurora Serverless for 심플, 가격 좋고, “Unpredictable” workloads

Elasticache

DB와 Web application의 성능 향상이 목적이다.

[senario] overloaded인데, 어떡하면 좋냐? → read replica / use elasticache

Redis와 Memcached가 내부적으로 사용되고 // Redis는 Multi-AZ, 백업, restore 가능

DMS; Databaes Migration Service

Migrate DB from one src to AWS → src는 on-premise, AWS 서비스, 다른 클라우드 등 가능

Homogenous(같은 DB엔진), Heterogeneous migration 모두 가능

\*Hetero면 AWS Schema Conversion Tool (SCT) 필요하다.

Caching Strategy 4가지

Cloudfront/Api Gateway/Elasticache(Memcached & Redis)/ DynamoDB Accelerator(DAX)

\* EMR; Elastic MapReduce

Big Data processing에 사용된다.

Master / core / task(optional) node로 구성되며, log는 master node에 저장된다.

\*\*\* 5문마다 master node에서 s3로 로그가 저장되는데, “cluster를 만들때만” 설정가능. 사후 X

문제풀이)

MySQL의 IOPS를 늘리고 싶으면 EBS추가 후 RAID 0 설정. CloudFront는 IOPS와 노상관

Single-AZ에서는 DB 백업이나 스냅샷 시 I/O가 느려지거나 중단된다(아주 잠깐).

Mongo DB는 Amazon Document DB와 연동된다. DynamoDB와는 상관없음.

Redshift = Business Intelligence

RDS는 backup window에 정해진 스케쥴대로 / immediately store된다.

RDS API에서 에러 탐지하고 싶으면 error node 참고하기

Reserved RDS는 multi AZ, single AZ 다 가능하다.

DB security groups are used with DB instances that are not in a VPC and on the EC2-Classic platform. When you create a DB security group, you need to specify a destination port number. (X)

RDS with provisioned IOPS storage → “SQL-based” rdb suitable for 고성능 OLTP

Athena(S3 스토리지를 쿼리로 찾는거)는 XML 지원 X

Provisioned IOPS → 뭐가 됐든 고성능, low latency와 sensitive한 작업 그런거.

Single-AZ RDS는 첨에 AZ정할 수 있다.

If I wanted to run a database on an EC2 instance, which of the following storage options would Amazon recommend? → EC2에 올린다는 거니까 EBS다. RDS는 올리는 개념이 아니니까.

MS SQL server 최대치는 100개의 db, 16TB

**Advanced IAM - 중요도 中**

AD vs Non-AD compatible : Amazon Directory (Service)

AD – Connect AWS resources with on-premises AD / SSO활용

AD compatible: Managed Microsoft AD / AD Connector / SimpleAD(얘는 trust X)

→ SimpleAD는 Microsoft AD의 little brother 개념

Non- : Cloud Directory(개발자용, fully managed) / Cognito user pools(유저 cred.관리 SNS로긴)

IAM Policies

ARN - 모든 AWS리소스에는 주소가 붙음

policy는 Effect(deny/allow) // Action // Resource로 나뉜다.

policies 자체는 statement 모음일 뿐이야. 이걸 적용하려면 Role에 attach, 그 Role을 User나 EC2 같은 곳에 주는거지

→ AWS joins all applicable policies

→ Policy의 변경 말고 Role 내에서만 설정 추가하려면 inline policy. 추천은 X

\*\*\*Not Explicitly allowed == Implicitly denied 디폴트가 deny인거지

Explicit deny > everything else

Aws managed (기성품) vs Customer managed

Permission Boundaries: 다른 유저에게 권한을 delegate할때 / 2차 방어막 역할이다.

→ prevent 과도한 permission & privilege escalation & Control maximum permission

ex) 개발자들이 Lambda function용 role을 만들 때/ EC2 인스턴스용 role / Ad-hoc Users

RAM: multi account strategy에서 resource sharing을 위해

App Mesh, Aurora, CodeBuild, EC2, EC2 ImageBuilder, License Manager, Route53, ResrcGrps

SSO; AWS Single Sign-On

[senario] 이미 존재하는 Gsuit나 office365같은 corporate resource로 AWS를 관리하고 싶어: SSO

여러 외부 서비스들과 함께 같은 identity로 관리

\* SAML나오면 무조건 SSO

**Route53 - 중요도 上**

Route53

Buy domain names directly with AWS

최대 3일 걸린다.

Route53 Routing policies

Health Checks: 상태 확인. fail시 DNS에서 제거되며, 알람도 가능하다.

Simple Routing: 레코드(주소)는 하나인데, 거기 연결할 IP address는 여러개일때.

이때 route53은 랜덤하게 하나 알려준다.

Weighted Routing: Simple과 같지만, 랜덤이 아니라 정해진 weight대로 보내준다.

Latency-based Routing: Lower latency를 찾아서 보내준다.

Failover Routing: active와 passive로 나눠서, active의 failure detection시 passive로 보내준다.

Geolocation Routing: latency와 상관없이 가까운 location으로 보내준다.

Geoproximity Routing: bias를 활용해 geographic location과 관련한 모든 트래픽을 직접 커스터마이징한다. "Traffic Flow"를 활용한다.

Multivalue Answer Routing: Simple + Health Check 기능 (unhealthy하면 failover)

문제풀이)

Domain name은 최대 20개까지, but AWS support로 늘릴 수 있다.

Multivalue Answer Routing은 최대 8개 ip까지. resiliency를 생각한다면 simple보다는 얘가 나음

Route53은 Amazon의 DNS service이다. 이름을 줄 수 있고, 해석할 때 연결해주는 역할이다.

**VPC - 중요도 最上**

vpc intro

vpc: logical datacenter in AWS

VPC = IGW + Route Tables + Network ACL + Subnets + Security Groups

\*\*\* 1 subnet = 1 AZ

Security Group은 stateful. Network ACL은 stateless(in과 outbound 모두 직접 열어줘야 한다.)

\*\*\* No Transitive Peering

1 igw on 1 vpc

VPC : logical datacenter in AWS

VPC 처음 만들면 Route Table + NACL + default Security Group 기본조합

Subnet이나 igw는 디폴트 아니다.

account마다 AZ는 randomized

하나의 Subnet안에 무조건 5개의 ip는 reserve되어있다.

1 igw per VPC / Security group은 VPC 간 적용 안된다. VPC 내부에서만

NAT Instance

Nat instance 생성 시 Src/Dest 중지하기

Nat instance는 public subnet 내에 위치한다.

Private subnet의 route table에서 Nat instance를 열어줘야 한다.(0.0.0.0/0)

Nat instance traffic은 그 instance에 달렸다.

→ 마찬가지로 Autoscaling Groups/ multiple subnets in dif. Az/ script failover등 가능

Security group내에 있어야 한다. (public subnet의 ec2와 같은 그룹)

Nat Gateway

Nat Gateway는 Security Group과 별개다. 전혀 관계없어!

NOT span AZ

Scale automatically & No patch & No security groups & 자동으로 ip address할당

여전히 Route table은 열어줘야 한다.

AZ내에 하나만 있는 Nat Gateway가 fail하면 망하니까 AZ independent 방향으로 짜라

→ 하나의 Nat Gateway로는 high availability가 보장되지 않는다.

NACL

VPC 생성하면 default NACL있다. → 모든 in, outbound 허용하고 있다.

Custom NACL은 처음에 모든 in, outbound 금지된다.

subnet은 default NACL로 자동 연결된다.

이렇게 특정 network를 막는 것은 Security Group (X) Nacl

Nacl에 여러개 subnet을 연결할 수 있지만, 하나의 subnet에는 하나의 Nacl만.

Nacl 규칙은 무조건 작은 순서 우선이다.

Nacl은 in과 out이 따로 놀며, 각자 설정할 수 있다. → stateless

ELB

ELB는 pre-defined IPv4 없다. 직접 DNS name을 줘야 한다.

Alias Record <-> CNAME / 시험에서는 무조건 둘중엔 Alias를 택하라

Alias는 CNAME과 다르게 Zone Apex로 사용될 수 있다. / Alias는 여러 계정에서 쓸 수 있다.

CNAME은 naked domain으로 사용될 수 없다.

SOA record / NS redords / A records/ CNAMES / MX records /PTR records

ELB demo

\*\*\* ELB 생성 시 최소 두개 이상의 서브넷이 있어야 한다.

VPC FlowLogs: CloudWatch랑 연동해서, 모든 트래픽 로그를 감시한다.

VPC peer끼리만 설정 가능, Across accounts는 불가능하다. (같은 계정이면 여러개 가능)

tag할 수 있다.

Configuration는 사후 설정, 변경 불가하다.

다 감시할 수 있는 건 아니다. Windows instance for amazon windows license/ metadata 확인용 (169.254.~) / DHCP / reserved IP address for default VPC router 등은 로그 안남는다.

\* your own DNS server 쓰면 완벽히 로그 남는다.

Bastions: SSH나 RDP로 직접 private에 접근하고싶을때.

Nat Gateway와 Nat instance로 private subnet의 EC2 연결 가능

Bastion은 안전+보안과 함께 EC2 인스턴스를 administer할 수 있는 기능이다.(SSH, RDP)

Nat Gateway를 Bastion host로 사용할 수 없다.

Direct Connect

directly connects your data center to AWS. AWS와의 통신을 강화하는 것 정도로 이해하자.

useful for high throughput workloads (lots of network traffic) Or stable, reliable connection

[senario] vpn 연결하는데 traffic이 너무 커서 자꾸 멈추면? Direct Connect을 사용해라

\*\*\* Direct Connect steps

PUBLIC Virtual Interface 생성하기 → Customer Gateway 생성하기 → Virtual Private Gateway생성하기 → Virtual Private Gateway를 목표 VPC에 붙이기 → 새로운 VPN Connection 만들기 → Virtual Private Gateway와 Customer Gateway 고르기 → VPN 완성되면 customer gateway나 firewall에서 VPN설정해주기

Global Accelerator:

내가 제공하는 서비스의 local, global user에게 high availability와 performance 위해 사용한다.

두 개의 static ip address가 assign된다. (can bring your own too)

Traffic dial을 이용해 traffic을 control할 수 있다. → endpoint group에서 가능

VPC Endpoint

이전에는 private subnet에서 NAT gateway로 보낸다음에 밖으로 끄집어내는거였는데,

이제는 private에서 바로 VPC gateway를 통해 밖으로 끄집어낸다. “powered by PrivateLink”

Privately connect your VPC to supported AWS services

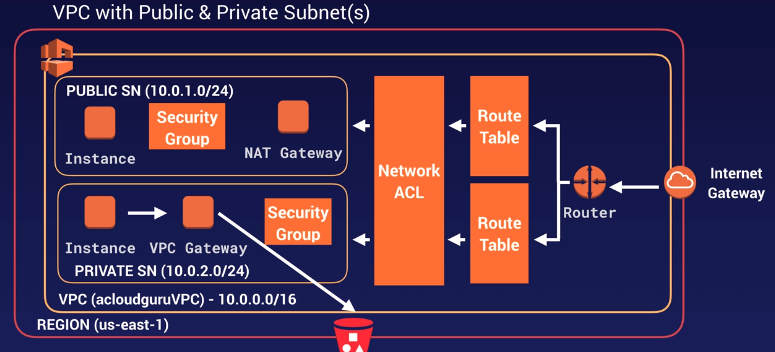
VPC endpoint services는 VPN이나 NAT, Direct Connection등의 과정 없이 가능하다.

\*\*\* Public address 필요 없다. 다 Amazon network 내부적으로 진행되거든.

VPC endpoints are virtual device이고, scalable, redundant, highly available하다.

VPC endpoint 종류 : Interface Endpoints(무지 많음) / Gateway Endpoints(S3, DynamoDB)

VPC diagram



AWS PrivateLink

여러 vpc에 연결해야하는데, 매번 peer로 하기 귀찮음 + 오픈하기는 보안상 안되는 경우

PrivateLink → 수많은 vpc끼리 private하게 연결할 수 있다.

주는 입장 - Network Load Balancer / 받는 입장 – ENI

외엔 아무것도 필요없다. (IGW, NAT, Route table …)

Transit Gateway

thousands of VPC와 transitive peering 가능하게 한다 + on-premise data centers

hub – and – spoke model에서 작동

region기반이지만 multiple region 설정할 수 있다.

Resource Access Manager 활용하면 다양한 accounts에서 공유 활용 가능하다.

Route table로 VPC간 talk를 제한할 수 있다.

Direct Connect + VPN connection과 함께 사용된다.

\* IP multicast 지원된다.

[senario] VPN, DC connection 엄청나고, VPC peering 무지 많고, 심지어 IP multicast 지원하면.

\* 골자는 Network topology를 simplify하기 위해서 사용되는 것이다.

AWS VPN CloudHub Diagram

여러 Site가 있고, 각각 AWS와 VPN으로 연결되어있을때, Site끼리 연결 가능하게 한다.

Hub-and-Spoke / low cost and easy manage / internet으로 진행되지만 encrypted

[senario] 여러 vpn을 쉽고 간단하게 관리하고싶다 → cloudhub diagram

AWS Network Costs

Use Private IP over Public IP to save cost. → AWS backbone network 써서 더 싸다.

가장 싸게 하려면 모든 ec2를 same AZ에 넣고 private ip address써라. \* Failure 대비는 못한다.

참고) inter region도 additional cost 있으니까 region도 굳이 안넘는게 좋지

\* VPN 이해못했어

문제풀이)

Custom vpc는 모든 az/ subnet 디폴트로 허용하는 룰 있다.

Security assessments나 penetration test는 서비스 따라 AWS alert 먼저 해야할수도, 아닐수도 있다.

“Network Zone”은 global accelerator에서 static ip 주는 역할한다.

VPC에서, instance는 중단 / terminate 후 재부팅 시 public ip address가 re-assign된다.

Subnet does NOT span multiple AZ

Global Accelerator component – Listeners / Endpoint groups / Static IP address

VPC Flow Logs는 VPC, Subnet, Network Interface Lvl에서 생성 가능하다.

Direct Connect는 improved security에도 기여한다.

VPC끼리 peer는 cross account에서도 가능하다.

Egress-only IGW는 VPC의 IPv6 기반 트래픽을 위한 것. VPC로 들어오는 IPv6는 막는다.

한 Region에는 한 계정당 5개의 VPC를 만들 수 있다.

**HA Architecture - 중요도 上**

\*\*\* Elastic Load Balancers: network load를 돕는 service

종류:

Application Load Balancers → layer 7에서 intelligent하게 작동

Network Load Balancers → layer 4, Extremely high performance에서

Classic Load Balancers → 가장 싸고 기본

504 Error는 Gateway time out. 즉, Application 차원에서 문제 생겼단 이야기다.

X-Forwarded-For를 보면 LB 때매 가려져서 보지 못한 user ip 볼 수 있다.

FAQ 꼭 읽어볼 것

\*\*\* ELB Lab

we’re NEVER given IP address for an ELB (DNS만 준다.)

Instance는 InService나 OutofService 상태로 ELB에 모니터링된다.

Health Check는 단순히 ping 튕겨보는 것으로 체크한다.

Advanced LB theory

Sticky Session

Users to stick to the SAME ec2 instance.

[senario] 한 유저가 많은 리퀘를 보내는데 계속 하나의 ec2로만 간다. 나머지는 아예 안온다.

어케 해결할꺼냐? Sticky Session을 끄면 된다. /

[senario] 로컬의 파일을 특정 instance에 저장하고싶어. 계속 연결이 바뀌면 안되잖아.

그럼 Sticky Session을 켜서 한쪽으로 유지되게 한다.

Cross Zone

Load balance across multiple AZ

[senario] 100%의 리퀘가 갈때, 한쪽에만 집중되고 나머지에는 아예 안왔다.

Cross Zone Load Balancing을 켜서 다른쪽에 가는 걸 cross해서 받아올 수 있다.

Path patterns

리퀘의 URL 패턴따라 로드밸런싱해준다.

[senario] 한 ec2는 그냥 url전용, 한 ec2는 media 전용이면,

normal-pattern은 그냥 url쪽으로, media-pattern은 media 전용으로 나눠서 보낼 수 있다.

\*\*\* Auto Scaling

Groups(logical component – ec2 넣는) / Configuration Templates(config) / Scaling Options

Scaling Options

Maintain current instance lvl: 항상 정해진 수가 돌아가도록(periodic health chk)

Scale Manually: 수를 정해서 직접 알려주기

Scale based on a schedule: 스케줄 정해서 scale up, down. Predictable할때

Scale based on demand: 각종 파라미터로 설정. ex. Cpu utilization 유지되도록

Use predictive scaling: 미리 예측해서, “across multiple services”

(이제는 config 설정보다 기설정된 template 적용이 더 추천되는듯)

\*\*\* Plan For Failure; HA design

[senario] mininum 6개의 instance있어야 하고, high availability 보장하려면?

3AZ with 3 inst여야 한다. 3AZ with 2 inst는 하나라도 fail하면 minimum 못채우니까.

Design for Failure & Cost effective를 생각해라

Muliple AZ와 Region을 최대한 사용하는게 베스트다.

\*\*\* RDS의 Multiple AZ는 DR용도, Read Replicas는 performance 용도 구분해라.

Scaling out은 autoscaling 이용해서 ec2 instance 늘리고 줄이는것 /

Scaling up은 하나의 ec2 instance 내부적으로 사이즈를 늘리는 것 (micro → large)

\*\*\* S3도 나오기 딱 좋다. Class 잘 생각해라.

LAB

RDS Reboot함으로써 한 AZ에서 다른 AZ로 강제 failover 가능하다.

CloudFormation

Completely Scripting your cloud environment (ec2뿐 아니라 각종 서비스들을…!)

Quick Start를 통해 전문가들이 만들어놓은 complex environment templete를 갖다쓴다.

Elastic Beanstalk

CloudFormation의 초보 버전. Quickly deploy and manage. Infra관련한 것 없게.

생성 후에 instance 직접 config / autoscale할 수 있다.

Capacity, load balancing, scaling, health monitor까지 다 알아서 해준다.

HA with Bastion Hosts

(1) Two hosts in two separate AZ. Use Network Load Balancer(4계층) with static IP addresses

→ Fail 시 다른 host로 옮겨간다. (Application Load Balancer는 사용 불가, 7계층이니까.)

(2) One host in one AZ + Auto scaling group with fixed EIP → Host fail 시 Auto Scaling group 이 새로운 EC2만들고, 너가 user data script로 새로운 host에 기존 EIP를 assign해주면 된다.

→ Cheapest지만 잠시 downtime존재한다. = not 100% fault tolerant

On-Premises Service with AWS

Database Migration Service (DMS)

Server Migration Service (SMS)

AWS Application Discovery Service

VM Import / Export

Download Amazon Linux 2 as an ISO

문제풀이)

Placement Group 중 Spread group에 only specific 한 것은? 한 AZ당 최대 7개까지만!

Unpredictable event에 대해 매우 scalable하고, cost가 낮으면서도 0.1% loss까지만

→ standard IA. 특정 기간에만 늘고 나머지는 infrequent하니까 IA이고 / 0.1이니 onezone은 X

끊김없이 잘 돌아가는게 Availability라면, 디자인대로 잘 돌아가는게 Reliability이다.

\*\*\* RDS를 런칭하기 전, 나중에 size를 바꿀 것을 대비하고자 한다. System outage도 최소화.

→ Multiple-AZ(AZ failover) / Aurora(higher availability & perf.).

→ Read Replica는 availability 보장과는 크게 상관없다. Perf. 상승이지.

RRS(S3 Onezone IA)는 이제 deprecated이다. AWS에서 비추한다.

그냥 RDB를 만들어서 read node로 쓸 순 없다. Failover용으로만 가능. Read Replica를 만들어라.

SPOF; Single Point Of Failure. 작동하지 않으면 전체가 중단되는 하나의 요소.

SPOF를 줄이고 bad deploy 위험을 줄이려면?

→ ELB마다 다양한 Target Group과 Auto Scaling Group을 둔다.

→ Route53의 health check를 ELB와 combination으로 쓴다. (Regionally / AZ level도)

→ Classic Load Balancer가 load over several AZ하도록 한다.

\*\*

ALB는 traffic for dif. targets를 distinguish할 수 있고, target group, condition, priority등에 따라 트래픽을 distribute할 수 있다.

**Applications 중요도 上**

SQS: 메세지 전송 큐

SQS is pull based (EC2가 queue로부터 pull해야한다.)

메세지는 256KB 이하 / 보관기간은 디폴트 4일, 최대 14일

standard queue는 빠르고 무식, FIFO queue는 느리지만 순서와 no duplication. (=only once)

\*\*\* Visibility timeout

ex. 메세지를 보내. Ec2 inst가 그걸 받아. 그 순간부터 processing 끝날때까지는 invisible하다.

제대로 time안에 끝내면 queue에서 삭제된다. 그런데 정해진 Visibility timeout이 expire할때까지 처리 못하면, 메세지가 다시 visible해지고 다른 ec2 inst가 가져가서 작업한다. 이 경우 같은 메세지가 두 번 deliver되는 결과가 생길 수 있다.

[senario] 같은 message를 두번 받았다. 왜 그런가? (1) Visibility timeout이 너무 짧게 설정되었다. → Visibility timeout을 늘려준다. 최대 12시간까지 가능하다.

SQS는 최소 한번은 process되는 것이 보장된다.

\*\*\* long polling

ec2는 queue가 비어있더라도 계속 pulling을 하고 charged된다. Long polling은 queue가 비어있으면, 뭔가 들어올때까지 답을 주지 않아서 ec2가 무지성으로 pull 할수 없게 만든다.

\*\*\* “Decoupling” = SQS

SWF vs SQS

SWF: human interaction이 있는 경우. ex. Amazon warehouse

SQS 14days / messaged oriented API / handle duplicated Msg / need application lvl tracking

SWF 1year / task oriented API / Task never duplicated / all tasks events are kept track

SWF Workflow Starters(첨에 명령 오는) / Deciders(실제 일 관리하는) / Activity workers(일꾼)

SNS: alert, notification 등을 보내주는 서비스

instantaneous, push-based (no polling)

Simple API, eash integration / Flexible msg delivery over multiple protocols

Cheap, pay-as-you-go / click interface 간단하다.

\* multiple AZ에 있어서 availability 걱정마라

\*\* SNS, SQS → 둘다 Msg service지만, SNS=Push, SQS=Pull

\* 대충 email, 알림받는 느낌이면 SNS / ec2 inst. 차원에서 info 다루는 느낌이면 SQS이다.

Elastic Transcoder: media transcoder.

Convert media files from original src format to diff. Formats

\*\*\* API Gateway: Door to your AWS environment

Caching capabilities (for perf.) / low cost & auto-scales / throttle to prevent atk / log to CloudWatch / Enable CORS / CORS is enforced by the client

Expose HTTPS to define Restful API

\* Same Origin Policy: 동일한 출처가 아닌 두 출처에 대해 리소스의 상호작용을 제한한다.

→ Cross-Site Scripting을 막기 위한 것이다.

\* BUT aws의 모든 서비스는 다른 도메인 이름을 가지니까 오히려 방해가 된다.

→ CORS를 켠다. Cross-Origin Resource Sharing. 다른 도메인에게 outside에서 상호작용 가능

[senario] Remote resource에서 접근이 막혔다. → CORS를 켜라

Kinesis

Streaming Data: 수많은 data source로부터 지속적으로 오는 데이터들

Kinesis Streams: Streaming Data를 Shard로 나눠서 보관 & 일주일 24시간 Persistent

Kinesis Firehose: 그냥 연결. Shard 없고, NOT persistent.

Kinesis Analytics: 위 두개를 on the fly에서 분석해주는 서비스

Cognito: AWS이 지원하는 Web Identity Provider 인증 서비스 / Identity Broker

User가 소셜로그인 → Authentication token을 User Pool에 넘김 → User Pool은 JWT를 Identity Pool에 넘김 → Identity Pool은 AWS IAM role 역할의 Credential을 최종적으로 줌 → AWS 접속!

User Pool → user-based. Registration, authentication, account recovery …

Identity Pool → Granting IAM / Authorising access to AWS

Synchronization 기능이 훌륭하다!

Event Processing Patterns

Pub/Sub model: 여러 publisher에서 나온 Msg들을 SNS Topic에 모아서 각각 Subscriber한테 보내주는 모델이다. Pub은 Sub 정보를 몰라도 되고, vice versa

DLQ: SNS에서 Fail한 Msg는 SQS로 보내지고, SQS max 차면 DLQ로 보내진다.

\* 람다 function이 최소 두번씩 더 시도하니까, DLQ는 완전히 실패한 것들이라고 볼 수 있다.

Fanout pattern: Publisher에서 SQS로 바로 보내지 않고 SNS Topic 거쳐서. Reliability위해서.

S3 Event Notifications: Customize한 기준으로, S3 bucket에서의 error들을 보고해준다.

\* 얘의 Notification들은 SQS, SNS, Lambda가 받게된다.

**Security - 중요도 下**

CloudHSM; hardware security module

Regulatory Complicance requirement이거나FIPS 140-2 lvl 3면 CloudHSM. (KMS = lvl 2)

manage your own keys (AWS는 접근권한 없어) / NO AWS APIs

Single tenant + Multi-AZ cluster

Java Cryptography Extensions (JCE) / Microsoft CryptoNG (CNG)

key 잃어버리면 irretrievable

Systems Manager Parameter store

parameter store: SSM(AWS Systems Manager)의 일부

Severless하게 configuration이나 민감한 정보를 저장한다.

→ + hierarchy도 있고, encrypted도 되고

Secrets Manager

SMM과 비슷하다.

SMM보다 더 비싸다 (per 10,000 API calls)

SMM과 다른 점: Automatically rotate secrets / Generate random secrets 가능하다.

**Severless - 중요도 中**

Lambda: Serverless

Architecture 자체가 굉장히 scalable하니까 bottleneck 문제 해결 가능하다.

[senario] ~~상황에 high availability + cost saving → Severless

Scale out automatically (NOT scale-up)

Lambda functions are independent. 1 event = 1 func.

Aurora, S3, API gateway, Lambda가 serverless. 그외는 server

Lambda func는 다른 Lambda func를 callable

AWS X-ray를 통해 debugging 가능하다.

Global하게 가능하다.

SAM; Serverless Application Model

CloudFormation extension optimized for Serverless apps

Funcs, apis, tables등을 정의할 수 있다.

CloudFormation이 지원하는 것 전부 지원한다.

Local하게 돌릴 수 있다.