ARCHITECTING ON AWS

summary

요약

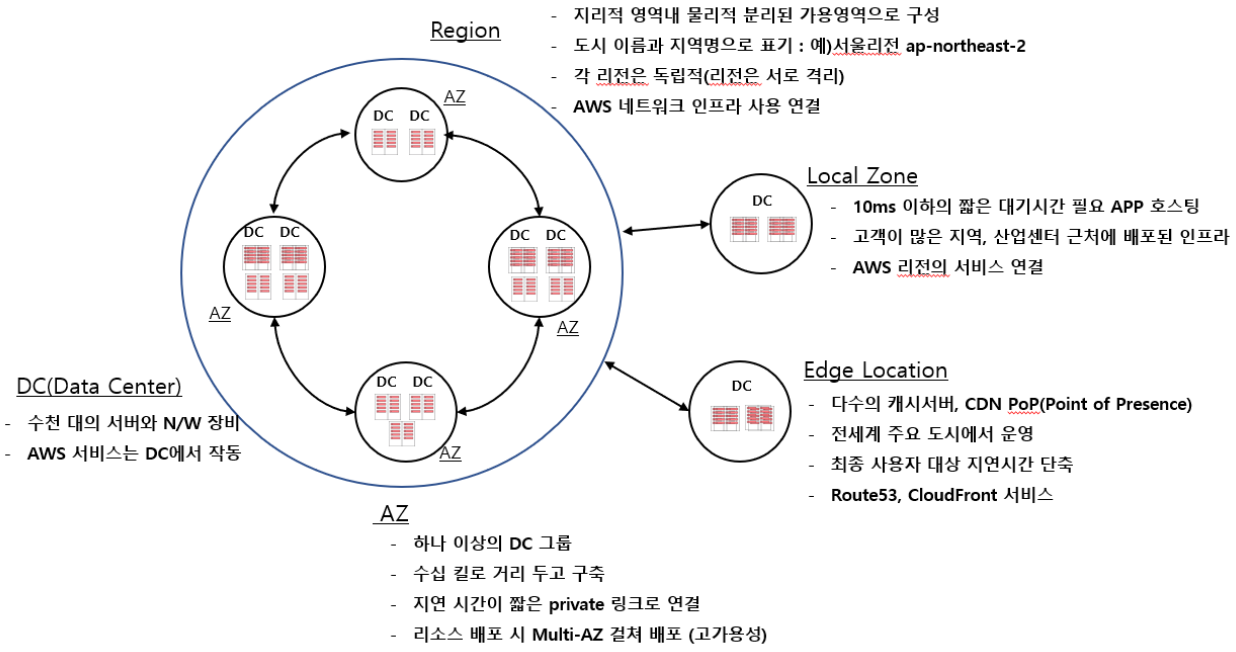
[매력적인 요약문으로 독자의 시선을 끌어 보세요. 일반적으로 요약은 문서의 내용을 간략하게 정리한 것입니다.
내용을 추가하려면 여기를 클릭하고 입력하세요.]

Ar7.9.12
[전자 메일 주소]

목차

Module 01 - 아키텍팅 기본 사항

- **AWS 서비스 : 클라우드 컴퓨팅**
    1. 인터넷
    2. IT 리소스(하드웨어, 소프트웨어)
    3. 온디멘드(요구시 즉시)
    4. 종량 요금제
- AWS 인프라
    1. 데이터 센터를 모아서 가용영역으로, 가용영역 2 개 이상을 모아서 리전으로
    2. 가용영역 표기: 리전의 지역명 표기 끝에 영문자를 추가하여 표기 ex. ap-northeast-2a, ap-northeast-2b...
       내결함성을 갖도록 설계
    3. 리전표기: 도시이름과 지역명으로 표기 ex. 서울리전 or ap-northeast-2
    4. Edge Location: PoP(Point of Presence); 전 세계 주요 도시에서 운영; Route53 및 CloudFront, WAF, Shield 같은 서비스가 지원
    5. Local Zone: Edge location 의 AWS infrastructure(도시형 가용영역); AWS 리전의 서비스와 연결되어 있고 지연 시간이 짧은 애플리케이션 제공을 위해 사용



- **리전 선택의 요인**: 거버넌스(규정준수), 지연시간, 서비스 가용성, 비용
- **AWS Well-Architected: 클라우드 아키텍처를 이해하고, 측정하고 모범 사례를 통해 아키텍처 구축에 직접 점검할 수 있도록 지침 제공**(셀프설문도구)
    1. WAF: 6 개 설계원칙 -보안, 비용최적화, 안정성, 성능효율성, 운영우수성, 지속가능성
    2. AWS Well-Architected Tool: 워크로드를 파악하고 아키텍처 검토, 개선 계획 검토, 문제해결 계획 수립
- **AWS 서비스에 연결**

1. AWS 관리 콘솔: Username/Password, 가시성, 직관적 인터페이스, 사용이 간단
2. AWS CLI: Access key(AK/SK), 반복적인 작업을 수행에 유리(스크립트화를 통해)
3. AWS SDK: Access key(AK/SK), 프로그램 적인 기법

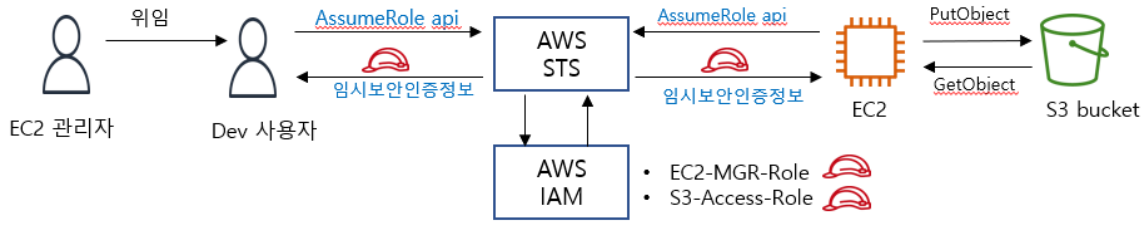
- References
  1. https://aws.amazon.com/ko/about-aws/global-infrastructure/
  2. https://aws.amazon.com/architecture/well-architected/
  3. https://docs.aws.amazon.com/cli/latest/reference/
     (https://docs.aws.amazon.com/cli/latest/reference/ec2/run-instances.html)

Module 02 - 계정보안

- **IAM: 인증과 권한부여**
  1. User, Group, Role 생성관리
  2. AWS 서비스 및 리소스에 대한 액세스 관리; 액세스 제어 분석
  3. 디렉토리 서비스와 통합
- **사용자**

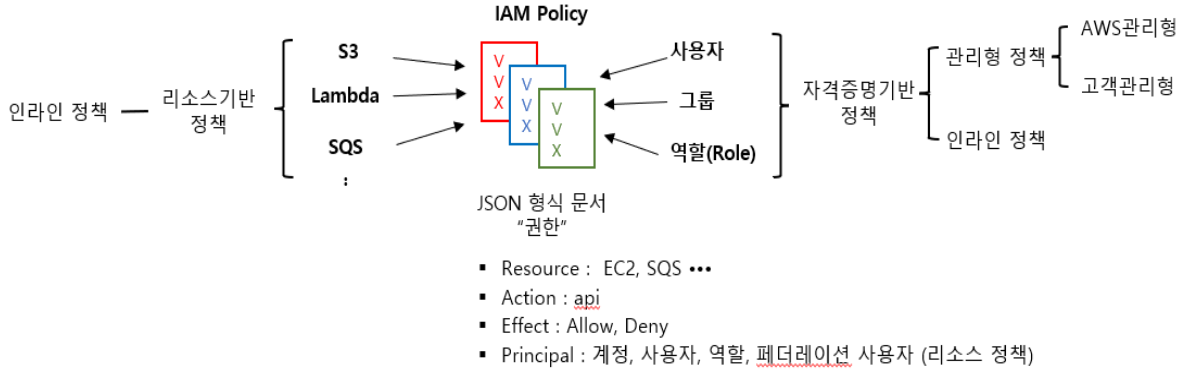
Root 사용자	IAM user
- 무한 권한 보유, - 일상적인 관리업무에 사용하면 안됨 - support plan 변경 - MFA 사용 권장	- 계정 내의 사용자(EC2 인스턴스나 DB 내의 사용자 X - 각 사용자는 자체 보안인증(Credential) 정보가 있고, 권한에 따라 특정 AWS 작업을 수행할 수 있음 - MFA 사용 권장 - 최소 권한 부여 원칙 적용

- **보안주체(Principal): AWS 리소스에 작업을 요청할 수 있는 권한을 가진 entity**
  1. IAM 사용자, IAM 역할, AWS 서비스, 자격증명공급자 또는 페더레이션사용자
- 사용자 보안 인증 정보(Credential)
  1. Root user: Email Address/Password *optional* MFA(Multi Factor Authentication)
  2. IAM user - Console: Username/Password *optional* MFA
  3. IAM user or other - CLI, SDKs: Access key(Access Key ID, Secret Access Key) *optional* MFAAccess key 는 CLI 의 경우 aws configure 로 등록하고, SDK 의 경우 (대개) 지정된 파일에 access key 를 등록
- **IAM Group:** User 에 대한 권한 할당을 상속 형태로 간편화하기 위해 사용
- **IAM Role:** 특정 사용자 또는 서비스에 특정 권한을 위임하고, 보안인증 정보를 (내부적으로 처리되기 때문에) 다른 사용자와 공유하지 않고 수임하여 권한을 가지고 작업을 하는 동안만 유효(Temporary)



1. AWS STS(Security Token Service): Role 수임에 대한 호출에 대해 처리하여 임시 credential 정보를 서비스(임시 access key id/secret access key, token(ttl 포함))
2. Role=모자: Trust Relationship(누가 이 모자를 쓸 수 있는 지의 권한) + Permission(모자를 썼을 때 얻어지는 권한)
3. 이점: 교차 계정 액세스 권한부여, 조직내 IdP 를 통한 권한부여, Application 이 안전하게 API call 을 이용, AWS 서비스가 (사용자 대신) 다른 서비스를 호출 가능

- **IAM Policy:** 하나 이상의 권한(Permission)을 잘 정리한 것으로 권한부여에 사용
    1. IAM 정책 할당 대상: IAM 사용자, IAM 그룹, IAM 역할(role)
    2. IAM 사용자, AWS 리소스는 IAM 역할을 수임 받아 IAM 역할에 할당된 정책내 권한을 갖음
- **Policy 유형**: 크게 권한을 부여하는 정책(Grant permissions)과 사용 가능한 권한의 범위와 제한에 대한 최대 권한 설정(Set maximum permissions)
    1. Grant Permission : iam 자격 증명기반과 IAM 리소스 기반 정책
       Set Maximum Permissions : iam 권한경계, AWS Organization 의 서비스제어정책(SCP)
    2. 정책: JSON 형식 문서(Resource, Action, Effect 필수 정의, 옵션 Condition, Principle 등)
        1. conditions: Optional 하게 사용하여 어떤 조건이 되었을 때 정책이 적용될 수 있도록 함
        2. Principle : 리소스기반 정책에만
    3. Identity-based policy: User, Group, Role
        1. Managed(aws managed, 사용자 생성 managed) vs. In-line
    4. Resource-based policy: S3, SQS 서비스와 같은 리소스에 직접 부여, in-line



- **Policy 권한평가 우선순위**: 1 명시적 거부, 2 명시적 허용, 3 묵시적 거부, 묵시적 허용(존재 X)
- **IAM 권한 경계(permissions boundaries)**: 사용자의 권한 제한(필터 역할, 과도한 허용 방지)
    1. IAM 권한 경계와 IAM 자격증명 기반 정책의 교집합만 허용
- 계정의 페더레이션 인증을 통합하고 서비스 이용에 따른 권한은 Role 을 통해 제공

- **AWS Organizations**: 모든 AWS 계정에 대한 예산, 규정준수, 보안 등을 중앙 집중화 관리(통합 제어 및 관리), OU 단위로 Member 계정을 할당하고 SCP 를 통해 통제; 통합 과금(billing management)
- **SCP(Service Control Policy**): AWS Organizations 서비스에서 계정을 통제하기 위해 사용(Safe guard)
    1. 특정 OU 에서 특정 (비용이 비싼) 서비스를 못 띄우게 하거나 혹은 특정 작업(인스턴스 셧다운)을 못하도록 방지
    2. 내리상속의 특성; IAM 권한과 같이 쓰여 마치 교집합(condition 과 같은 역할)으로 권한 통제
- References

    https://docs.aws.amazon.com/ko_kr/sdkref/latest/guide/overview.html

Module 03 - 네트워킹 1

- **VPC 는 Private IPv4 주소(default) 또는 IPv6 를 사용할 수 있고 CIDR 표기법 이용**
    1. 단일 AWS 리전에 설정; 워크로드의 논리적 격리를 제공, 보안설정 가능; 사용 가능한 IP 주소의 범위를 설정,
    2. Default VPC: 계정 생성 시 모든 리전에 자동적으로 만들어짐(IGW 포함)
    3. 여러 가용영역에 걸쳐 VPC 배포하면 고가용성을 실현, 각 가용영역에 리소스 배포
- **Subnet: VPC CIDR 블록의 하위집합**, 하나의 AZ 에 위치하고 여러 AZ 에 걸칠 수 없음; 서브넷간 IP 는 중첩 불가
    1. Public/Private 종류가 있는 건 아니고, 인터넷에 노출여부로 지칭
    2. Public subnet 내의 인스턴스는 사설 IP 와 공인 IP 를 갖음. 퍼블릭서브넷의 라우팅테이블에는 인터넷 게이트웨이에 대한 경로가 들어가 있음.
    3.
- **공인(Public) IP** 는 EC2 인스턴스를 론칭할 때 부여하는 Public IP(PIP)와 Elastic IP(EIP) 2 가지
    1. PIP 는 인스턴스 stop/start 후 IP 주소가 변경됨; EIP 는 변경되지 않음
    2. EIP 는 인스턴스 또는 ENI 에 연결할 수 있고, 고정 IP 이므로 서버의 서비스용 IP 로 이용할 수 있고 리전당 5 개로 제한(soft), 다른 인스턴스에 이동 가능
- **Internet Gateway:** 인스턴스와 인터넷간 (양방향) 통신 허용; 서브넷과 연결된 route table 에 등록필요, (내부적으로) 수평확장/중복
- **NAT Gateway**: private subnet 의 private IP 주소를 가진 인스턴스가 outbound 로 인터넷을 이용할 수 있게끔, 인스턴스의 사설 IP 를 NAT gateway 의 EIP 로 NAT 처리(소스 IP 로)하여 트래픽을 처리.
- **Route table:** VPC 를 만들면 main route table 이 자동적으로 만들어지고, (대개) main route table 은 private subnet 이 참조하는 route table 로 이용; public subnet 은 사용자 지정 route table 을 별도로 만들어 association
    1. 모든 route table 에는 default route(*VPC_IP_CIDR local*) 가 등록되는데 삭제할 수 없음

2. Default route: 모든 사용자지정 route table 생성 시 기본적으로 추가, default route 로 인해 VPC 내 모든 인스턴스는 서로간 통신가능

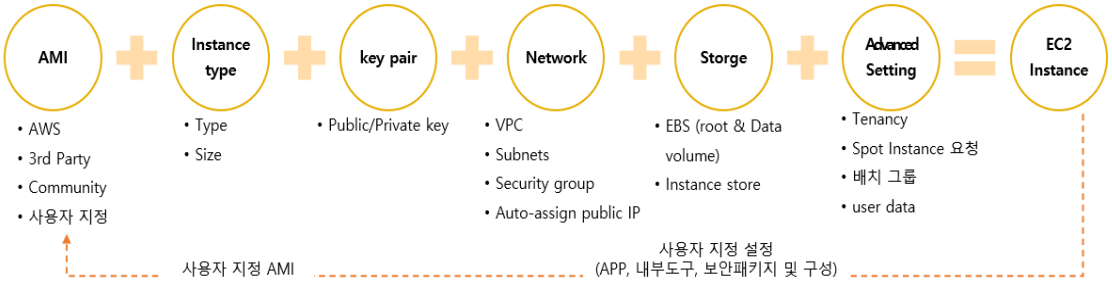
- **ENI(Elastic Network Interface):** 동일 AZ 내에서 리소스간 이동가능; Private, Public, MAC 주소를 유지
- **Network ACL**: 서브넷의 방화벽; 아웃바운드, 인바운드가 오픈 상태로 시작; 낮은 규칙번호부터 적용, 마지막은 * (일반적으로 모든 트래픽 거부 등록); stateless
- **Security Group**: 인스턴스(ENI) 방화벽; 아웃바운드는 열리고 인바운드는 닫힌 상태로 시작; IP 주소, 프로토콜, 포트기반 트래픽제어, stateful
    1. Security Group Chaining: Source 를 다른 SG 로 참조하여 마치 체인의 연결처럼 SG 를 구성하여 더욱 안전한 보안구성
- References

1. NAT 인스턴스 : https://docs.aws.amazon.com/ko_kr/vpc/latest/userguide/VPC_NAT_Instance.html

Module 04 - 컴퓨팅

- **AMI(Amazon Machine Image): 골든이미지**
    1. OS + 알파 : 인스턴스볼륨 템플릿, 시작권한, 볼륨디바이스매핑
    2. 가져오는 위치: AWS 제공(사전구축-Quick start), AWS Marketplace, 자체생성(고객 생성-My AMI), Community AMI

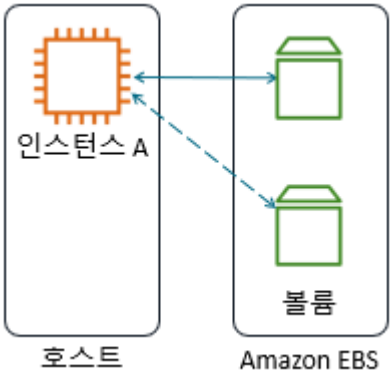
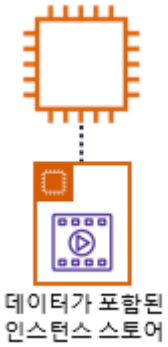
EC2 인스턴스 런치



- EC2 Image Builder: AMI 생성 자동화
- **Instance type: Family-Generation-Attribute.Size ➔ c5n.xlarge**
    1. 범용, 메모리최적화, 스토리지최적화, 컴퓨팅최적화, 엑셀러레이티드 컴퓨팅
    2. 최신 세대 인스턴스가 성능대비 비용효율적(용량은 늘리고 비용은 낮춤)
- EC2 키 페어 : EC2 에 ssh 연결할 때 사용하는 인증키, public key 는 EC2 에 저장, private key 는 사용자에 저장, System Manager 의 session manager 는 키페어 없이 브라우저에서 ec2 연결 지원
- Multi-Tenancy(shared tenancy) vs. Single-Tenancy: **기본적으로 공유테넌시(multi-tenancy)**, 전용인스턴스 및 전용호스트로 single-tenancy
    1. Dedicated instance 의 경우 같은 계정내 일반 인스턴스와 hardware 를 공유 가능
    2. dedicated host 의 경우 socket 확인이 가능해 이를 요구하는 라이선스 환경에 이용
- 배치그룹 : 워크로드의 요구사항에 맞도록 인스턴스간의 거리 선택
    1. 클러스터 배치 그룹: EC2 인스턴스를 서로 인접하게 배치, 고성능컴퓨팅(HPC)

2. 분산형 배치 그룹: 여러 네트워크 세그먼트와 랙에 분산 배치, 장애에 민감한 내결함성이 필요한 워크로드
3. 파티션 배치 그룹 : 지정된 수의 파티션에 균일하게 분산 배치, 상관관계가 있는 하드웨어 장애를 방지, 대규모 분산형 워크로드(하둡, 카산드라 등)

- User data | Instance metadata
  1. User data : 인스턴스가 시작(론치)될 때 자동 구성이 필요한 작업을 실행하기 위한 스크립트
  2. Instance metadata : 실행 중인 인스턴의 구성, 관리 정보 관리, 저장소는 http://169.254.169.254/lastest/meta-data 사용, curl 명령어를 이용 메타
- AWS Compute Optimizer: 기계학습을 사용하여 최적의 인스턴스를 사용할 수 있도록 권고 (지난 14 일 동안의 지표를 기반으로 분석, 권장사항 도출)
- **EBS vs Instance Store(Storage): 데이터 휘발성 여부, snapshot 백업 가능여부**

EBS	Instance store
	
네트워크를 통해 EC2 연결 OS, 애플리케이션 데이터 볼륨(영구적 저장) 동일 가용영역내 인스턴스와 연결 스냅샵 생성->백업/복원, 스냅샷을 다른 리전복사 가능 SSD 기반 볼륨: 범용(gp2,gp3), Provisioned Iops(io1,io2,io2 express) HDD 기반볼륨: 처리량 최적화(st1),콜드 HDD(sc1)	EC2 에 로컬하게 연결된 볼륨 (host 컴퓨터에 물리적으로 연결된 디스크에 위치) 버퍼, 캐시, 임시데이터용(EC2 종료시 데이터손실) HDD,SSD 및 NVMe SSD 유형 지원 스냅샷을 지원하지 않음

- **EBS volume types: SSD 기반 vs. HDD 기반**
  1. SSD 기반: 범용, Provisioned IOPS ➔ OS,애플리케이션, 데이터베이스
  2. HDD 기반: 처리량 최적화, 콜드➔ 스트리밍, 로그처리, OS 볼륨 X

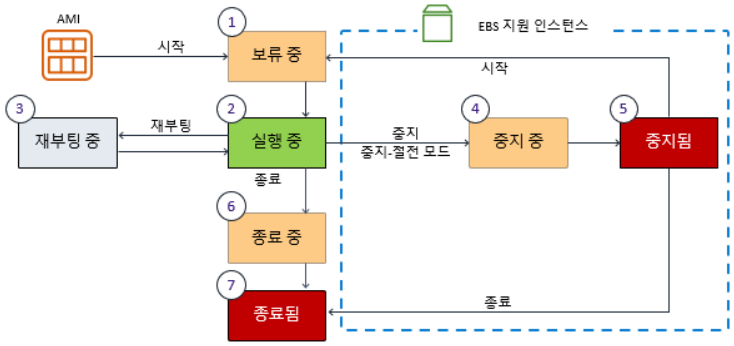
- **EC2 구매옵션: Savings Plans, Spot Instances, On-Demand Instances**
  1. 온디멘드 : 약정없이 초당 또는 시간당 결제, 변화가 심한 워크로드, 임시 사용
  2. Savings plans : 1 년 또는 3 년 약정, 컴퓨팅 리소스에 대해 시간당 금액으로 약정, 컴퓨팅에 대한 유연한 사용이 필요한 경우, Compute savings plans 와 EC2 Instance Savings Plans
  3. 스팟 인스턴스 : 온디멘드 대비 최대 90% 할인금액, 스팟성(임시) 사용, 내결함성, stateless 워크로드에 적합

4. Savings Plans vs Reserved Instances(1 년 또는 3 년 약정, 사용량이 꾸준한 경우 적합)

구분	컴퓨팅 savings plan	EC2 인스턴스 Savings plan	컨버터블 RI	표준 RI
할인혜택 (온디멘드 대비)	최대 66%	최대 72%	최대 54%	최대 72%
인스턴스 사이즈 변경	O	O	O	O
인스턴스 패밀리, OS 변경	O	△ (OS 변경가능)	O	X
리전 변경	O	X	X	X
Fargate,Lambda 지원	O	X	X	X

- **Lambda:** 서버리스 컴퓨팅, 다양한 코드지원, 최대 15 분간 실행, 최대 10GB 메모리 지원(메모리에 비례해서 자동적으로 CPU 할당)
    1. 주로 이벤트소스가 되는 다른 서비스를 통해 호출되어 사용, 시간설정 호출 가능
    2. 람다 생성시 메모리 사이즈를 정의, 실행시간 지정 가능
- References

1. https://docs.aws.amazon.com/ko_kr/AWSEC2/latest/UserGuide/resize-limitations.html
2. **EC2 인스턴스 수명주기**



3. EC2 ssh key pair 분실 시 교체 방법
https://aws.amazon.com/ko/premiumsupport/knowledge-center/ec2-windows-replace-lost-key-pair/

Module 05 - 스토리지

- **S3: 내구성이 뛰어난 객체 스토리지 솔루션; 백업 및 복원, 비즈니스 크리티컬 애플리케이션, 아카이빙 및 규정준수**
    1. **버킷에 객체 저장**; folder 라 쓰고 prefix 라 읽는다; 객체 Key 로써 객체를 인식
    2. **All block public access**: ACL, Bucket policy, CORS 를 하나의 옵션으로 모든 public 액세스를 차단
    3. **버전관리(Versioning):** 삭제와 over-write 상황에서 파일을 보존할 수 있음
    4. **스토리지 클래스: 액세스 빈도에 따라 클래스 선택➔** 비용, 성능
        1. Standard, Standard IA(Infrequent Access), One Zone IA(Infrequent Access), Intelligent-Tiering, Glacier Instant Retrieval, Glacier Flexible Retrieval(Formerly S3 Glacier), Glacier Deep Archive

    2. S3 Glacier Instant Retrieval 에서는 분기당 한 번 데이터에 액세스하는 경우 S3 Standard-Infrequent Access(S3 Standard-IA) 스토리지 클래스를 사용할 때와 비교하면 최대 68%의 스토리지 비용을 절감
    3. S3 Express One Zone : 디렉토리 버킷 사용할 때 단일 가용영역내 데이터를 저장되며, 10 밀리초 미만의 일관된 낮은 latency 를 보장. 고성능 컴퓨팅 및 AI/ML 작업을 위한 스토리지로 사용
5. S3 Glacier(Flexible Retrieval): Archive 를 Vault 에 저장; 장기 보관 데이터에 대한 비용 효율적이고 내구성이 높은 스토리지
    1. S3 GFR: 신속(1~5 분) – 표준(3~5 시간) – 대량(5~12 시간)
6. **S3 수명 주기 정책(Lifecycle)**: 데이터의 수명주기에 따라 스토리지 클래스 전환
7. **저장 시 암호화(SSE(Server-side encryption)**: SSE-S3, SSE-KMS, DSSE-KMS, SSE-C
8. **객체 복제(Replicating):** S3 버킷 전체에 걸쳐 객체를 비동기로 복사; 동일 리전간 복제, 교차 리전간 복제
9. **이벤트 알림(Event Notification):** S3 이벤트 기반의 트리거하여 람다와 상호 동작
10. S3 Transfer Acceleration: 엣지 로케이션을 이용하여 데이터를 빠르게 전송
11. S3 멀티파트 업로드: 여러 개의 파트로 쪼개서 병렬 업로드; 일시중단/재개;콘솔 x ,API O
12. **액세스 포인트:** 대규모 액세스 고유한 DNS 주소로 관리; 각 포인트별로 고유한 권한 및 네트워크 제어

- **EFS: NFSv4 프로토콜**, 탑재대상(mount target)을 통해 액세스, Windows 인스턴스는 지원하지 않음
    1. IP Address or DNS 로 액세스
    2. 처리량은 Enhanced 모드와 버스팅 모드
    3. 버스트 처리량 모드 :스토리지 용량에 따라 처리량 확장
    4. 프로비저닝 모드 : 스토리지 용량 무관하게 처리량을 프로비저닝 ; Elastic ➔ I/O 가 예측 불가능, 처리량이 자동 관리 / Provisioned ➔ I/O 처리량 요구사항이 예측 가능한 경우, 정의한 처리량 만큼의 성능을 확보
    5. 파일 스토리지가 자동 확장 및 축소
    6. 여러 스토리지 클래스를 지원- Standard, Infrequent Access(Standard, One-zone), Intelligent-Tiering
- **Amazone FSx : 다양한 기능을 제공하는 고성능 파일 시스템**
    1. **FSx for Windows File Server: SMB 프로토콜**을 이용하여 Windows 서버간 파일 공유지원
    2. **FSx for Lustre: 대규모 용량/성능을 지원하는 분산 파일시스템**; S3 버킷과 통합하여 읽기/쓰기 가능
    3. **FSx for NETapp ONTAP, FSX for OpenZFS**
- **AWS Snow Family**: Offline 데이터 마이그레이션 서비스
    1. Snowball: Snowball Edge Storage Optimized 및 Compute Optimized 포함; 페타바이트 규모 데이터 전송; EC2 수행

- **DataSync**: 에이전트를(VMware, KVM, EC2 instance 등) 통해 온프레미스 데이터를 S3 또는 EFS 로 copy.
- **Storage Gateway**: 온프레미스 IT 환경과 AWS 스토리지 인프라 사이 통합 서비스
    1. File Gateway: S3 File Gateway(NFS), FSx File Gateway(SMB)
    2. Volume Gateway: Cached mode, Stored mode; iSCSI
    3. Tape Gateway: VTL(Virtual Tape Library); iSCSI
- References
    1. S3 Transfer Acceleration 속도 측정 데모 사이트 : https://s3-accelerate-speedtest.s3-accelerate.amazonaws.com/en/accelerate-speed-comparsion.html

Module 06 - 데이터베이스 서비스

- (관계형 데이터베이스 vs. 비관계형 데이터베이스) vs. (관리형 데이터베이스 vs. 비관리형 데이터베이스)
- **RDS: HW, OS 및 DB 배포 및 유지관리; 저장 시 및 전송 중 암호화; Multi-AZ, Read Replica; 손쉬운 컴퓨팅 및 스토리지 조정**
    1. **6 개 엔진: Aurora(2 개 인터페이스-PostgreSQL, Mysql 호환), PostgreSQL, MySQL, MariaDB, Oracle, SQL Server**
    2. **Aurora: MySQL & PostgreSQL compatible; 3 개의 가용영역에 각각 2 개의 사본을 유지; 최대 15 개 RR(Read Replica) 구성; Compute Node 와 Storage Node 가 분리**
    3. **Multi-AZ**: Primary - Secondary; HA 환경; 동기식 복제
        1. MySQL, PostgreSQL : 2 개의 read replicas 를 갖는 Multi-AZ Cluster 지원
    4. **Read Replicas**: Read only; 읽기 성능 개선; 비동기식 복제
        1. Aurora –15 개까지, 기타 RDS 5 개까지
    5. Aurora Serverless : 고객 요구사항 기반으로 자동 시작, 미사용시 종료, 사용한만큼 지불, 크기조정시 어플리케이션 영향 없음. (버전 1 은 단일 가용영역, 버전 2 멀티 AZ 및 읽기복제본 지원)
- **DynamoDB: 완전 관리형 NoSQL 데이터베이스; Key-Value 데이터모델**; partition key 로 분할되고 스케일링 되어 일관된 성능을 제공
    1. table, item, attribute
    2. primary key: partition key or composite key(복합키->파티션키+정렬키)
    3. **consistency options**: Eventually consistent vs. Strongly consistent
        1. Eventually consistent : 기본 설정 예) 신문기사 댓글
        2. Strongly consistent : 읽기 작업(getitems 등) 시 ConsistenRead 옵션을 true, 예) 비용과 민감한 업무
    4. **용량관리 모드** : Provisioned mode vs On-demand mode
        1. 성능 기준 : RCU(Read Capacity unit): 1RCU 는 4KB 항목/s ; WCU(Write Capacity Unit) : 1WCU 는 1KB 항목/s;
        2. 읽기 일관성 옵션 :최종일관성 0.5 RCU 사용, 강력한일관성 1 RCU 사용
        3. 온디멘드 모드는 변동성에 대한 예측이 불가능한 경우 사용

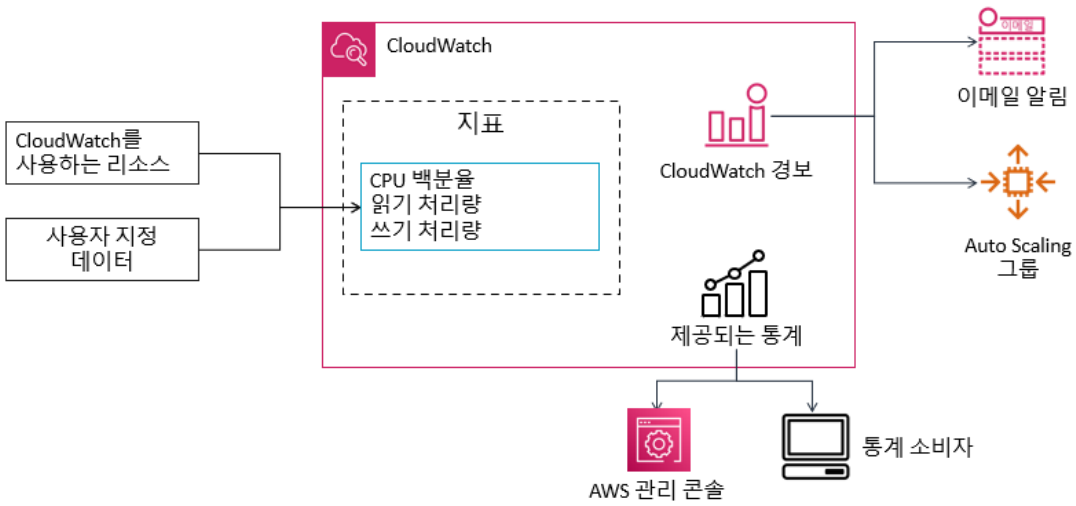
4. Provisioned mode : 트래픽이 예측 가능한 경우 사용, Auto-Scaling 옵션을 적용하면 비용을 효율적으로 관리
5. **Global table**: 리전당 한 개의 복제 테이블, 모든 복제본은 동일한 테이블 이름과 동일한 primary key 스키마

- Database Caching
  1. 캐싱전략: AWS 서비스에 내재되어 있는 선택기능은 아니고 애플리케이션 캐시 사용 시 고려사항
     1. **Lazy-loading**: cache-miss 시에 캐시에 저장; 원본 데이터의 업데이트 변화 시 stale 될 수 있음
     2. **Write-through**: 데이터 쓰기 시 캐시에 저장; 불필요한 데이터가 캐시될 수 있음.
     3. **TTL(Time to Live)** 값을 통해 2 가지 전략의 단점을 낮출 수 있다.
     4. **메모리관리(eviction 전략) :** TTL 만료, 사용한지 가장 오래된 데이터, 사용빈도가 가장 낮은 데이터
  2. **ElastiCache: Memcached vs Redis ; in-Memory 엔진, 탁월산 성능과 확장성**
     1. 데이터베이스 캐시로 둘다 적합
     2. Redis : 다양한 데이터 형식(리스트, 집합, 해시, 등), 데이터세트 정렬 및 순위지정, 게시/구독 등 추가 지원 기능이 많음
  3. **DynamoDB Accelerator(DAX)**: DynamoDB 를 위한 완전관리형 고가용성 캐시
     1. 마이크로초 단위의 성능(초당 백만 리퀘스트 처리)
- **Database Migration Service(DMS)**: 한번 또는 지속적 마이그레이션 가능, Snowball Edge 이용가능, 이기종 가능
  1. **이기종 마이그레이션 시 SCT(Schema Conversion Tool)**로 스키만 변환

- References
  1. Aurora Severless 버전 1 과 2 비교
     https://docs.aws.amazon.com/ko_kr/AmazonRDS/latest/AuroraUserGuide/aurora-serverless-v2.upgrade.html#aurora-serverless.comparison

Module 07 - 모니터링 및 스케일링

- **CloudWatch**: 지표수집, 모니터링, 경보를 생성하여 알림, 리소스 용량 변경(auto-scaling), 대시보드



1. 지표(metric): 리소스의 성능 데이터 수집, 저장. Standard metric vs. custom metric; 일정한 시간 간격
2. **CloudWatch Logs**: 애플리케이션 로그를 모아서 저장, 분석하여 지표 및 경보 생성(지표필터)
    1. 로그 그룹 : 보관주기, 동일 유형의 로그의 모임
    2. 로그 스트림: 로그 그룹내에서 인스턴스 ID, 네트워크 ID 등과 같은 기준으로 구분된 로그의 모임
    3. 필터, 액세스제어, 보존기간 설정 가능
    4. Ec2(agent 설치), 외부소스(cloudtrail, VPC flow logs)
3. **경보(alarm)**: 대상 metric 식별 - 경보 생성 - 액션 정의
    1. 경보 상태: OK, ALARM, INSUFFICIENT DATA
    2. 임계값이 지정된 기간을 초과할 때 경보가 발생하며 임계값 아래로 떨어지면 바로 OK 로 전환

- **CloudTrail:** 계정활동을 로깅하고 모니터링, API 기록, 감사 목적
- **VPC Flow Logs**: Elastic 네트워크 인터페이스(ENI)에서 송수신되는 IP 트래픽에 대한 정보를 수집
- **CloudWatch Events 및 EventBridge**: 이벤트 중심의 아키텍처 구현
    1. EventBridge 는 Events 를 확장한 개념; 이벤트 소스가 사용자 지정 이벤트 및 SaaS 애플리케이션으로 확장
- **ELB: 사용자 트래픽을 분산하고, 고가용성 제공, health check**
    1. 리스너 - 규칙 - Target Groups - Targets
    2. 유형: ALB(L7-HTTP,HTTPS), NLB(L4-TCP,UDP), GWLB(L3/L4-IP), CLB(L7, L4-HTTP/HTTPS,TCP, 이전세대)
- **Auto Scaling**: 조건에 따라 AWS 리소스를 시작 또는 종료, 새 인스턴스를 로드 밸런서에 등록
    1. Auto Scaling (EC2, DynamoDB, Aurora 등) vs. EC2 Auto Scaling
    2. **EC2 Auto Scaling:**
        1. 시작 구성 및 시작 템플릿 : AMI, 인스턴스 유형, 보안그룹,역할
        2. Auto Scaling group : VPC, 서브넷, 로드밸런서, min, max, desired capacity

3. Auto Scaling policy : 예약/온디멘드/예측가능 오토스케일링, 스케일링 인/아웃정책

- **Auto Scaling group 용량 조정 방법**
    1. 특정 고정 인스턴스 수 유지(Maintaining a fixed number of instances(Steady State)): 실습 4 참조
    2. Manual scaling(수동)

    (automatic scaling tab 에서 조정 가능
    3. Dynamic scaling(동적) ➔ 클라우드와치 함께 이용됨
        1. 단순조정정책 : 기준을 1 개 이용 (예,클라우드와치 경보 CPU 70%이상이면 추가 인스턴스 3 개 실행)
        2. 단계조정정책 : 동적 조정 기준을 단계별로 설정 가능 (예, CPU 20% 40%일 때 등 단계설정) :
        3. 대상추적조정정책: 기준을 맞추기 위해 min/max 범위에서 인스턴스를 조정
    4. Scheduled scaling(스케쥴)
    5. Predictive scaling(예측): 머신러닝 기반 예측 기반
- Auto Scaling group 에 예약 인스턴스(RI), 스팟 인스턴스 사용으로 비용 최적화

Module 08 - 자동화

- **AWS CloudFormation : AWS 리소스들을 안전하고 반복 가능하도록 구성하는 방식을 JSON 또는 YAML 형식의 템플릿 파일 사용**
    1. Infrastructure as Code(IaC)
    2. Template 를 통해 CFN(CloudFormation)에 의해 만들어진 인프라 스트럭쳐가 곧 stack 이라고 할 수 있음.
        1. Resources 섹션은 mandatory
    3. 드리프트|drift 감지 기능으로 탬플릿을 통해 stack 이 만들어진 이후 사용자 등에 의해 구성이 변경되는 것을 감지하는 기능
    4. Change set 기능: stack 을 업데이트 하기 전에 변경에 대해 미리보기 기능
- AWS CDK 는 익숙한 프로그래밍 언어를 사용하여 클라우드 애플리케이션 리소스를 모델링 및 프로비저닝할 수 있는 오픈 소스 소프트웨어 개발 프레임워크
    1. Python, JavaScript, TypeScript, Java, C# 등
- **AWS Elastic Beanstalk** : 개발자가 기본 인프라에 대해 걱정할 필요 없이 클라우드에 확장 가능한 웹 애플리케이션 및 서비스를 배포하고 유지 관리할 수 있도록 도움
- AWS 솔루션 라이브러리 사이트 : AWS 아키텍트가 승인한 솔루션, 템플릿과 스크립트 제공
- **AWS Systems Manager** : 다수의 EC2 에 대한 소프트웨어 인벤토리 수집, OS 패치 적용, 시스템 이미지 생성, Windows 및 Linux 운영 체제 구성을 자동화
- **Amazon Q : 개발자와 비즈니스 사용자를 위한 생성형 AI 도우미. 자연어 질문에 대한 답변, 콘텐츠 생성, 작업 수행 등 지원하는 서비스**
    1. Q Developer : 개발자, IT 전문가용,

1. IDE, 명령줄에서 주석이나 기존 코드 패턴을 기반으로 한 코드 제안
2. 복잡한 코드 설명, 보안 취약점 검사 및 수정 등 지원
3. 아키텍쳐, 비용최적화 및 문제 진당 등
4. 소프트웨어 개발 수명주기 전반에 걸친 사용자 지원

2. **Q Business : 비즈니스 사용자용.**
    1. 회사의 정책, 제품, 비즈니스 결과 등 회사의 데이터 리포지토리와 정보시스템을 검색하여 질문에 대한 답변, 콘텐츠 생성, 작업 수행

Module 09 - 컨테이너

- 모놀리식 인프라, 모놀리식 애플리케이션 ➔ 마이크로서비스
    1. 모놀리식 인프라를 ELB, SQS, SNS 서비스를 이용 loosely coupling 환경으로 개선
    2. 모놀리식 애플리케이션(모든 기능을 단일 서비스로) ➔ 마이크로 서비스(기능별 서비스 분리, 개발언어, 플랫폼, 인프라 독립적)
- **컨테이너** : 서버에 설치된 운영 체제를 공유하며 리소스가 격리된 프로세스 형태로 실행되므로 환경에 상관없이 빠르고 안정적이며 일관된 배포를 보장
- **가상화 및 추상화 수준: 베어메탈 서버➔가상머신➔컨테이너**
    1. 가상머신은 격리되어 있지만 동일한 OS 및 바이너리/라이브러리를 공유하지 않음
    2. 컨테이너는 격리되어 있지만 OS 를 공유하고 필요한 경우 바이너리/라이브러리를 공유
- **관리형 컨테이너 서비스: ECS vs. EKS**
    1. ECR(Elastic Container Registry)에 컨테이너 이미지를 등록하여 중앙에서 서비스 및 관리
    2. EC2 or Fargate(서버,클러스터 관리 필요가 없는 컨테이너-서버리스형 컨테이너)를 이용하여 컨테이너를 운영
    3. ECS : 도커 컨테이너를 지원하는 컨테이너 관리 서비스
    4. EKS : 온프레미스에서 실행되는 쿠버네티스 배포와 완벽히 호환성 유지
    5. ECR(Elastic Container Registry)에 컨테이너 이미지를 등록하여 중앙에서 서비스 및 관리

Module 10 - 네트워킹 2

- **VPC Endpoints** : VPC 내 EC2 인스턴스가 AWS Service 들(S3,DynamoDB 등)을 사용시 인터넷을 통하지 않고 리전 내부적으로 Private 하게 연결
    1. **Gateway Endpoint**: S3, DynamoDB 만 지원, 라우팅테이블에 엔트리가 들어감(전용게이트웨이로 볼 수 있음)
    2. **Interface Endpoint**: AWS PrivateLink 를 지원하는 서비스에 연결, 같은 subnet 에 ENI 를 만들어 지원되는 서비스로 향하는 트래픽의 진입점 역할, 시큐리티그룹과 같은 보안설정 가능
       S3 는 gateway endpoint 와 Interface endpoint 를 모두 가능
- **VPC Peering** : VPC 와 VPC 간에 1:1 연결 , 리전에 걸쳐 연결가능, AWS Account 간 연결 가능
    1. VPC Peering 대상 VPC 의 IP 가 중복될 수 없음

    2. 두 VPC 간 라우팅 테이블 업데이트 필요,
- **TGW(Transit Gateway): Hub-spoke 방식으로 VPC 간 연결**, 온프레미스와 VPC 간 연결(VPN)
    1. Hub 역할
    2. TGW's route table 을 이용하여 연결/격리
    3. TGW 리전간 피어링은 모든 트래픽을 암호화
- **Site-to-Site VPN**: 인터넷을 통한 온프레미스와 VPC 연결, 최대 1.25Gbps
    1. VGW(Virtual Private Gateway)와 고객게이트웨이디바이스(온프레미스)
    ※ VPN 연결 옵션 : site-to-site VPN, EC2 인스턴스에 VPN 구성, Transit gateway
- **Direct Connect(DX): 전용선 서비스**; DX 로케이션에서 연결, 최대 100Gbps (리전에 따라 다를 수 있음),
- **VGW(Virtual Private Gateway)** : 가상 프라이빗 게이트웨이, Amazon VPC 와 다른 네트워크 사이에 프라이빗 연결(VPN)
- 온프레미스와 AWS VPC 연결은 VPN , DX (Direct Connect) 이용
- References
    1. **AWS PrivateLink 솔루션**: PrivateLink 를 지원하는 AWS 서비스에 대해 Interface Endpoint 를 통해 퍼블릭 인터넷을 타지 않고 내부적으로 직접 연결
        1. AWS 파트너가 호스트하는 서비스 및 Marketplace 에서 제공되는 지원 솔루션에도 연결 가능,
        2. 온프레미스 데이터센터내 애플리케이션(DX,VPN 연결 환경에서)에서 VPC 인터페이스 앤드포인트를 통해 S3 버킷 연결 가능
        3. 단방향 액세스, 단일 서비스에만 액세스(연결 대상이 여러 개인 경우 각각 생성), 서로 다른 VPC 인 경우 주소가 겹쳐도 OK

Module 11 - 서버리스

- **서버리스: 프로비저닝하거나 관리할 인프라가 없음, 자동으로 크기조정, 종량제 요금, 내장된 보안-고가용성 컴퓨팅**
- **API Gateway** : 애플리케이션의 "현관" 역할을 하는 API 를 생성 , 최대 수십만 건의 동시 API 호출을 처리
    1. 백앤드 애플리케이션과 함께 이용 (람다함수, Ec2 의 퍼블릭 엔드포인트, 기타 AWS 서비스 등)
    2. HTTP, REST API, WebSocket API 지원
- **Amazon Simple Queue Service (Amazon SQS)** : 완전 관리형 메시지 대기열 서비스, 메세지가 처리및 삭제 될 때까지 저장됨.
    1. **대기열은 표준(Standard) 대기열과 FIFO 대기열**을 제공함.
        1. 표준 : 최소 1 회 보장(2 번이상 처리될 수 있음), 순서비지정
        2. FIFO : 한번만, 정확한 순서대로, 제한된 초당 Api (기본 300api/s)
    2. **배달 못한 편지 대기열 (DLQ-Dead Letter Queue)** 를 지원하여 처리되지 못한 메세지를 별도로 구분하여 문제 파악 가능

3. **가시성 제한 시간** : SQS 에 저장된 메세지를 수신하는 경우 다른 수신자가 메제지를 보지 못하도록 설정하는 시간(중복처리방지), 기본 30 초, 최대 12 시간까지 설정가능
4. **긴(long) vs 짧은(short) 폴링**
    1. 긴폴링: 기능을 통해서 메세지가 큐에 들어올때 까지 긴 시간동안 기다림
    2. 짧은폴링에 비해 적은 폴링은로 비용을 아낄 수 있음.

- **Amazon Simple Notification Service (Amazon SNS)** : 손쉽게 알림 기능을 설정, 전송할 수 있는 웹 서비스, "게시-구독"(pub-sub) 메시징 패러다임을 따르며 "푸시" 메커니즘을 사용하여 클라이언트에 알림을 전달
    1. **SNS 구독 유형** : Email, HTTP/HTTPS , SMS , SQS , Lambda 함수
    2. 1: N 배포, 푸시
    3. 팬아웃 : SNS => 여러 SQS 대기열
- **Kinesis Data Steams**: 분석을 위해 데이터 스트림을 수집 및 저장
    1. 스트림을 생성하고 샤드 수를 지정; 스트림의 총 용량은 샤드 용량의 합계
    2. 샤드당 1MB/s 쓰기, 2MB/s 읽기, 1000 개레코드/s
- **Kinesis Firehose**: 데이터 스트림을 AWS 데이터 스토어에 로드
    1. Redshift, S3, OpenSearch Service, HTTP 엔드포인트, 서드 파티 서비스 공급자
- **AWS Step Functions**: 서비스 구성요소(Lambda Functions, SQS, SNS 등)을 오케스트레이션 하여 State Machine 을 만들 수 있으며, 시각적 워크플로를 사용한 마이크로 서비스 조정
    1. state=Type, Resource(Lambda function, 컨테이너, DynamoDB...),Parameters, Next..

Module 12 - 엣지 서비스

- **AWS Edge**
    1. AWS 엣지 로케이션 - CloudFront, AWS WAF 및 AWS Shield 가 여기에서 사용되는 서비스
    2. Outposts - Outposts 는 온프레미스 또는 VPN 에 위치
    3. AWS Local Zones – 인구가 많은 산업센터 근처에 있는 AWS 클라우드 확장, 도시형 가용영역
    4. AWS snow 패밀리 : 엣지에서 오프라인 스토리지를 제공,

    ※ AWS Wavelength - 5G 사업자는 엣지 로케이션으로 AWS Wavelength 를 사용
    .
- **Route 53 :** 도메인이름을 IP 주소로 확인, 도메인 이름 등록 또는이전
    1. **Public hosted zone vs. Private hosted zone**
    2. Routing policies : 단순(라운드로빈), 장애조치, 지리적위치, 지리적 근접성, 대기시간 기반, 다중값응답, 가중치
    3. 하이브리드 클라우드용 Route 53 Resolver
        1. 온프레미스 DNS 리졸버, AWS route53 리졸버(인바운드엔드포인, 아웃바운드엔드포인트)

- **Amazon CloudFront :** 웹 사이트, API, 동영상 콘텐츠 또는 기타 웹 자산의 전송을 가속화하는 글로벌 CDN (Contenct Delivery Network) Service
    1. CloudFront 는 사용자에게 전달되는 다양한 Content 에 대한 Cache 서비스
    2. 콘텐츠 만료 방법 : Time To Live(TTL) , 객체 이름 변경, 객체 무효화(Invalidation)
- **Shield Service** : DDoS 공격(distributed denial of service (DDoS) attack) 보호
    1. Shield Standard: 3 계층과 4 계층에 대한 DDos 공격에 대해 보호, 모든 고객들 추가 비용없이 자동보호
        1. 인프라 계층 공격: UDP reflection attack, SYN floods
    2. Shield Advanced : 애플리케이션 계층 공격(Slowloris DDos attack, HTTP flood attck(Cache-busting attacks, WordPress XML-RPC floods), DNS floods, SSL negotiation attacks 에 대해 보호
- **AWS WAF**: 애플리케이션 계층 트래픽을 검사하고 필터를 적용(HTTP 및 HTTPS)
    1. 보호대상: CloudFront, ALB, API Gateway, AppSync
    2. 7 계층 HTTP 헤더와 본문, URI 문자열 및 SQL 인젝션 및 크로스사이트스크립트 공격에 대한 방어
    3. ACL 규칙 구문을 사용하여 트래픽 제어
        1. 공격방지, 트래픽 필터링, 패턴일치, 논리적연산
- AWS Firewall Manager: Organization 환경에서 AWS WAF 및 Amazon VPC 보안 그룹의 운영 및 유지 관리 태스크를 단순화
    1. 계정 및 애플리케이션 전반에 걸친 규칙 관리를 단순화
    2. 자동으로 새 계정 검색 및 규정 미준수 이벤트 수정
    3. AWS Marketplace 에서 AWS WAF 규칙을 배포
    4. 모든 계정에서 신속하게 공격에 대응
- **Outposts:** AWS Cloud 를 온프레미스 데이터 센터로 확장; 짧은 지연 시간, 로컬 운영 요구사항 충족
    1. 용량을 주문하기 전에 사이트가 Outposts 요구 사항을 충족하는지 확인 필요
    2. Enterprise Support plan 필요
    3. AWS 리소스 지원: VPC, ALB, EC2, EBS, S3, RDS, ElastiCache, ECS, EKS 등

Module 13 - 백업 및 복구

- **RTO & RPO**
    1. RTO(Recovery Time Objectives) : 재해/장애 발생시 어플리케션 중단이 발생하는 최대 시간.(서비스를 복원하기까지 걸리는 시간)
    2. RPO(Recovery Point Obejctive): 시간 단위로 측정한 허용 가능한 데이터 손실량
- **재해 복구에 필수적인 AWS 서비스**
    1. 스토리지 : S3(교차리전복제), S3 Glacier, EBS(스냅샷, 리전/계정간 스냅샷복제), snowball, Datasync
    2. 컴퓨팅 : AMI, ECS 컨테이너 이미지

3. 네트워킹 : route53(장애조치라우팅), ELB(장애인스턴스 격리), VPC, DX(온프레미스에 대한 DR 로 AWS)
4. 데이터베이스 : RDS (리전에 스냅샷 저장, 읽기전용복제본-다중 AZ, 자동 백업), DynamoDB(글로벌테이블,특정시점복구)
5. 배포 오케스트레이션 : CloudFormation, (CLI,SDK 활동)자동화 스크립트

- **AWS Backup: AWS Backup 은 완전관리형 백업 서비스**로, AWS 서비스 전체에 걸쳐 데이터 백업을 쉽게 중앙 집중화하고 자동화
    1. EBS, EC2, EFS, Storage Gateway, DynamoDB, Aurora, RDS, S3 bucket, Neptune DB, DocumentDB, 4 FSx series,
    2. backup plan 으로 중앙 백업 수행 및 관리
- **재해 복구 전략(재해 복구 시나리오) 사례**
    1. 백업 및 복원
    2. 파일럿 라이트
    3. 완전 동작 저용량 스탠바이
    4. 다중사이트 액티브-액티브