\*208 EFS는 디폴트가 burst throughput 모드인데, 용량 작은경우 효과적이지 않다. 용량 작아도 꾸준한 고사양이 필요하면 Provisioned

209 한 계정의 ec2와 다른 계정의 Redshift를 잇는 가장 cost effective 방법은 VPC sharing

210 일단 high availability라면 autoscaling + ELB + Multi-AZ를 떠올리고 시작하자.

211 UDP = lv3,4계층 = NLB

\*212 CloudTrail이 secure한지 확인한다 → (1) Log 유효한지 CloudTrail log file validation (2) KMS쓰고있는지

213 key-value = DynamoDB // Api Gateway 쓰는데 unpredictable해서 scaling있어야 한다 = Lambda

\*214 VPC에서 IPv6 space가 부족해지면 IPv4를 더 큰 range로 만들어서 launch해야한다. IPv6-only subnet같은건 애초에 불가능하다. IPv4가 디폴트다!

215 on-premise의 RDS를 코드를 바꾸지 않고 AWS로 이전하려면 DMS 밖엔 없지. snowball은 storage

216 특정 personnel에게만 db 접근 가능하게 하려면 IAM도 있겠지만 DB 차원에서 KMS-CMS 쓸수도 있겠지

\*217 인터넷 연결 잘 안되는 Disaster 지역에서 s3에 저장하려면? 그냥 Snowball 들고다니자. Storage gateway는 가능은 한데 비효율적이다.

218 참고) VPC Flow Logs의 target은 S3와 CloudWatch이다. Kinesis(X) CloudTrail(X)

219 SQS는 poll 당한다. 하는게 아니라. SNS는 push 알아서 해준다.

220 리눅스 + HPC = Fsx Lustre

\*221 디도스 = Shield // “DNS is managed by 3rd party” = Route 53 못쓴다

222 SMB + File Server = Fsx

\*223 UDP = NLB // TCP,UDP + high performance = GA // NLB와 Cloudfront는 연동 안되는듯

224 ECS에서 EC2로 가는건 autoscale 여부를 떠나서 NOT cost effective

\*225 MySQL → Aurora MySQL migration은 매우 쉽다 // Caching 추가하는 것은 heavy code change. // Caching은 Replication latency와는 상관없지

226 Static IP address + lowering latency + many Regions = GA

\*227 10TB 비용이 나왔는데 bucket 보면 5TB만 있어. (1) Versioning (2) Multipart 다 안끝나서

228 ec2에서 지표 하나라도 있으면 simple scaling보단 target tracking scaling

229 Elasticache는 DB/S3 앞에 한다 // latency 향상 필요 + S3 + 그외 단서 X = CloudFront

230 virtual private gateway는 DC랑 연결되는 것 같은데.. 확실치 않음. 확실한 NGW 고르자

231 reproduction이나 second backup 아니면 웬만하면 OneZone 고르지 말자

232 Multi-instance + low latency btw insts = Cluster Placement

233 VPC 내에서 resources 연결 + No internet access = VPC endpoint

234 1초 이내 RPO + 1분 이내 RTO + Multi-Region DR + RDB = Aurora Global DB

\*235 Static website + Cost effective = S3 // 최소 99% availability = No versioning. // S3자체가 99는 디폴트로 넘는다.

236 Static website는 S3없이 Cloudfront로 못옮김 // Lambda는 최대 15분이야. 긴 작업은 못한다.

237 ec2에 credential 절대 넣지 말고 대신 IAM Role 붙여주는게 국룰이다.

238 S3내 파일마다 접근 확률 들쑥날쑥이면 intelligent가 제일 낫다.

239 Data Events - S3 object레벨에서의 작업(get, delete ..), Lambda function 기록 // Management Events – configuring security(IAM role), Register Device, Router 등. // 로깅은 CloudTrail거쳐서 하자

\*240 Global 언급 나왔으니, Dynamo DB에서 DAX보다는 global table이 더 시급한 것이다.

241 S3 인터넷 없이 내부적으로 ec2랑 연결하려면 S3 gateway endpoint

242 API gateway + cost-eff. = Lambda // cost-eff storage → Cloudfront+S3 > EC2+RDS+ELB

243 주의. NAT gw 열어도 다른 website나 organization이 볼수 있는건 아냐. 들어오는 인터넷만 오픈.

244 Credential은 언제나 Secrets Manager에서 auto rotation 켜는게 최고다.

245 Standard Reserved - 인스턴스 수는 정해져 있고, 각각 인스턴스의 resources나 config를 사전에.

246 하나의 VPC(network service)에 연결할 accounts (=VPCs)가 많다 = Transit Gateway

\*247 worldwide = CloudFront // language based caching = Accept-Language request header

애매하긴 한데 static website 명시되어있지 않으면 일단 S3는 의심하고 본다.

248 하나의 volume에 여러 ec2가 접근 = EFS

249 Loosely coupled = SQS // SQS는 누가 poll할때까지 남아있는거니까 더 Durable하다. // SQS의 남은 길이로 Scaling하는 case (SNS로는 X)

250 S3 접근 권한 (URL 제한하는 케이스)+ 주로 Cloudfront = OAI

251 File sharing System = EFS, FSX, S3, StorageGateway // SMB = FSX, Storage Gateway(File)

252 On-premise(AWS간에도 가능)에서 Migration + Sync가 지속적으로 필요 = DataSync

253 RDS의 Backup inst.는 serve 불가하다.

\*254 DB에서 inst type을 specify하지 않고자 한다 = Serverless 쓰겠다. Aurora Serverless

255 어디 단에서의 지연인지 확인. EC2와 RDS중 CPU문제없으면 RDS가 못버티는거니 Replica.

256 HA 단골 → autoscaling + ALB + multiple AZ 조합 // 새로운 AZ에 subnet 만들고 multiAZ deploy

257 Supplementary storage + Cost effective = S3 One Zone IA

\*258 Near-Real-time = Kinesis DS // Scalable = Lambda // 명확한 fan-out아니면 SNS는 재고하자.

259 Guarantee processing = SQS // Cost-effective + scale-in protection = Spot Inst. // SQS와 EC2 Autoscale 조합이면 큐 길이 보고 정한다. = current # of msg in the queue

\*260 VPC peering / Transit GW 모두 서로의 resrc를 볼 수 있다. // Direct Connect + VPN 조합은 Scalable하고 각자는 링크 안된다. [Datacenter에 DC가 연결되고, DC끝에 VPN들을 붙이는 구조이다]

\*261 Geoproximity routing - resource의 location 따라서 // Multivalue answer – DNS queries 경우

262 HA = MultiAZ-DB + AG EC2 + ELB

263 S3 bucket에 항상 encrypted obj →policy 중 “x-amz-server-side-encryption” header set

264 mry 안잃으려면 Hibernate ec2

\*265 real-time = Kinesis // Orderly = SQS FIFO or Kinesis DS // 이경우 많은 데이터를 SQS로 처리하기도 애매하고, 그 데이터를 굳이 비싼 EFS에 저장할 이유도 없다.

266 DynamoDB Streams는 log 기록 / 24시간 저장하지만, app이 직접 확인해야한다. (Cloud X)

267 AWS Organization 단위에서는 사용자들 Service Control Policy로 권한 설정 가능하다.

268 EC2 recovery를 automate하는 가장 쉬운 방법 = CloudWatch alarm

CloudWatch - 서비스/활동의 상태와 성능을 보고 + trigger기능. What is happening

CloudTrail – 모든 API활동 기록하는 로그 + 진짜 Auditing느낌. Who did what

269 Share + MS Window + storage capacity부족 = Fsx for Windows

270 Lambda가 권한 필요로 함 → EC2에 붙이듯이 IAM role 만들어서 붙여주자. Resource base policy는 다른 account나 service의 접근을 설정할 때 사용한다.

\*271 꼼꼼히 읽자. Disaster에 대비하는건데 autoscale이 있어야 한다. 속도 자체는 cloudformation template가 ec2 여는것보다는 빠를 것이다.

272 EC2간 data transfer시, 같은 Region이어도 inter-AZ면 charge된다. (다른 서비스들이 EC2로 in, out되는건 같은 Region이면 no charge로 추정된다.)

273 Lambda Edge로 파일 압축하면 save money 가능하다. 한번 쓰는건 Caching의미없음에 유의해라.

274 Hostname(DNS)는 하나인데, 여러 url path → AWS로 옮길 때에는 path based routing ALB

275 Kinesis Data Firehose + Analytics 조합. Analytics는 Firehose, DS가 데이터 넣어줘야한다.

276 EC2와 S3의 은밀한 만남 = VPC endpoint

277 SG는 Stateful, Nacl은 less. 가장 limited access라면 ELB→Web→RDS 끼리만 여는게 맞지

278 SQL query + ACID = RDS // Neptune은 graph형 DB

\*279 Availability 높이기 위해, 새로운 S3 bucket 만들고, 그걸 포인팅하는 새로운 CloudFront origin을 만든 담에, 기존의 CloudFront origin과 Group으로 묶어서, 하나를 primary, 나머지를 secondary로 만든다.

Route53을 쓸 수도 있는데, CloudFront HA를 위한 recommended 구조는 이게 더 가까운 듯

\*280 Kinesis DS는 디폴트 24시간 보관. ‘Every other day’마다 S3에 저장하려면 retention 늘려야한다.

\*281 Big data, Real time도 아닌데 Kinesis가 무슨 상관. Category처리는 Fanout의 전형이다.

282 Gateway VPC Endpoint = S3, Dynamo 두개만 // Interface VPC Endpoint

283 루트 유저 보안 → 평소에는 IAM 유지하기 // MFA 켜기

284 private subnet의 ec2를 그냥 open하려면 → 같은 AZ에 public subnet열고 LB에 associate. 이후 LB에 private subnet의 ec2 등록한다.

\*285 Realtime + Scalable = Kinesis DS or Firehose // Data loss 줄인다 = Kinesis Data Analysis보다는 warehouse로 저장기능이 있는 Redshift가 낫다.

286 Aurora Read Replica도 autoscaling 켤 수 있어!

287 Caching은 code-change 큰 작업이다. 심플한 read replica가 db performance향상엔 최고다.

\*288 Fargate는 따로 AZ 설정을 안해줘도 된다. 알아서 span된다. ECS에 Fargate 얹으면 그 수 만큼만 켜면 된다. = Placement나 AZ 고려 아예 필요없다.

\*289 AWS Storage Gateway – Volume 으로 저장한 것은 바로 S3 API로 접근불가. SG IO기능으로만.

290 Organization 내에서 account에 S3 bucket 접근 줄때 + 업무량 고려하면 → ‘aws:PrincipalOrgID’로 global condition key 만들어서 S3 policy에 붙이기. OrgID만으로 접근 제어할 수 있다.

291 S3로 오기 전 이미 encrypt 되어있어야 한다 = client side encryption // KMS와 SSE-C는 공통이지만, S3 Mangaed Keys 방식은 serverside에만 적용가능하다.

\*292 빠른 입출력 필요 + temporary file = Instance store. 인스턴스에 블록 단위 임시 스토리지 제공된다. Instance store는 재부팅 외에는 메모리 날아간다.

293 제일 빠른, milisecond 보장되는 DB는 DynamoDB밖에 없다.

294 NAT gateway는 인터넷으로 가는거니까 비싸다. Vpc endpoint 뚫어주는게 cost effective

295 Windows + shared file = FSX // access AD = AD domain for autentication

296 global + performance + web = CloudFront // CloudFront는 static, dynamic 모두 커버 가능하다.

\*297 Granuality no more than 2min = 1분 단위 EC2 Detailed Monitoring // Cloudwatch로 분석한다.

298 global = CloudFront

299 EFS → EFS IA로 보낼 수 있는 최소 기간 = 7일

300 Amazon MQ - 메시징 서비스를 Aws로 migration할 때 사용한다. Active/Standby broker + MultiAZ로 HA 설계할 수 있다.

\*301 권한 줄 때 IAM 잘 살펴봐라. application까지 줘서 오답. Presigned URL로 log만 download.

302 security group = stateful. // \* 들어오는 포트는 자기 기준이다. EC2→ MS SQL server라면, DB입장에서 EC2를 1433포트로 열어줘야 한다.

303 CloudFront에 (ec2)origin group을 주고 primary, secondary 주면 알아서 failover 시켜준다. HA용.

304 Historical data + querying = Athena

305 CloudFront + HTTPS + Security + additional layer = Field-level encryption profile

\*306 report를 만드는데 너무 쿼리가 커서 EC2 서버 자체가 잘 안돌아간다 = Decoupling의 전형적인 사례이다. SQS에 넣고 연결성 좋은 람다로 report 만들어주자.

307 Global = cloudfront // cost-effective + static web = S3

308 Fresh(new) data + upload = NOT cloudfront // Slow start configuration = ALB(O) NLB(X)

\*309 Autoscaling에서 EC2 열때, Launch Configuration에서 script 설정은 오직 First launch에만 적용된다. Shutdown 시에도 audit 진행하려면 Autoscaling Lifecycle Hooks에서 설정해야한다.

310 Onpremise+low latency = DC // endure slower traffic + low cost = VPN

\*311 EC2 종류 (EBS 아님) → 범용(t) / 컴퓨팅(c) / 메모리(r,x) / 가속컴퓨팅(p,g) / 스토리지(i,d) // IOPS따질거면 Storage Optimized다.

\*312 Query + real time = Kinesis Daya Analytics. // real time 단서 없었으면 충분히 Athena도 될듯하다. 다만 Athena는 너무 강력히 historical이 use case이긴 하다.

313 RDS 부하걸리면 Read Replica가 국룰이다. 참고) mysql → Aurora mysql 이식은 code change X

314 CORS는 web hosting 시 single origin을 override한다. Cross Region Replication과는 아무 상관없다. // Glacier Deep Archive는 retrieve에 12시간까지 걸린다.

315 파일에 접근하고, 수정하고, 삭제하는 건 모두 CloudTrail의 Data events이다. (mngement 아님 + 얘는 애초에 default로 켜져있다.)

316 Disaster라면 availability 위해 ASG설정한다.

317 API Gateway + traffic varies(autoscalable) = Lambda

\*318 Rest API = API gateway / modernize = serverless = Lambda

319 HDD – 500 IOPS / gp – 3000~16000 / io – 16000~64000 / ioblock – 64000~256,000 // 참고) million 단위 iops는 Instance Store만 지원한다. 3Million iops (3,000,000)

320 copy를 원하고, secure manner이면, encrypted snapshot 만들어서 주자. 복잡하게 생각하지 말자.

321 NFS – EFS or Storage Gateway. // EC2 share 관련이면 EFS, On-premise에서 옮기는거면 SG.주의) S3는 아예 NFS지원 안한다.

322 Frequent read requests = Cache // Read Replica는 100ms 이하일 순 있지만.. // 1ms 이하는 Caching과 DynamoDB정도만 가능하다.

323 safeguard access – vpc endpoint

324 cheap + global + static web = Cloudfront + S3

\*325 Mobile + Login process = Cognito // MFA 설정하려면 SMS-based MFA 켤 수 있다.

326 Scheduled Reserved Inst는 더이상 X

327 dynamic IP = 계속 바뀜 = NACL(X) // inspector는 test용 // GuardDuty는 detection, NOT prevention. // WAF가 가장 prevention에 가깝다.

328 iam policy에서 “bucket-name/\*” 이 bucket 내의 각 원소를 지칭한다.

\*329 public subnet hosting을 가능케 하는 건 NATgw와 ALB. 문제를 꼼꼼히 읽자

\*330 ECS에서 IAM role을 설정하고 싶으면 taskRoleArn으로 설정한다. 단순히 IAM role을 ec2처럼 붙이는 방법은 없는 것으로 보인다. + S3에 접근 가능하게 하려면 S3가 아닌 접근하려는 서비스에 붙여야지.

331 별다른 code change 없이 mysql을 이식한다 = aurora mysql

332 log 기록하는데 igw, vpcpeering, sqs 등이 나올 이유가 없다. Cloudwatch

\*333 GA는 자체적으로 endpoint의 healthiness 여부를 체크한다. 즉, Redirecting traffic to healthy endpoints가 성립한다. ALB를 쓴다면 그걸 endpoint로 추가해서 그 ALB가 전부 건강한지 체크하는 식.

334 인터넷 없이 diff Region의 두 VPC간 통신 원해 → VPC peering. Peering은 cross-Region 가능, VPC endpoint는 한 Region 내에서만 가능하다.

335 API Gateway의 access 보안 = API key // WAF는 당연하고.

\*336 주의) Region 단위에서 DB를 failover하는 방법은 Cross-Region replication을 켜서 Read Replica를 만들고 그걸 master로 승격시킨다. AZ에서는 MultiAZ를 켜거나 Read Replica를 만드는 거지 // standby instance는 MultiAZ의 근거로 보는게 맞을듯 // Multimaster read / write 가 가능한건 DynamoDB, Aurora. // 참고로 Dif. Region에 만드려면 DB 인스턴스는 Encrypted 되어있어야 하고, 애초에 SQL svr는 지원 X

\*337 Low bandwidth + 60TB → 아무래도 snowball이 나을거다. 너무 커서. Storage gw도 가능은 하다.

338 Nat Inst. 는 Nat GW로 바꾸자. scalable해지고, 더 HA해진다. NatGW는 AZ당 하나씩은 필요.

339 흔한 HA. Aurora는 replica 만들고, ALB앞에 Cloudfront두기

340 Fargate는 MultiAZ같은거 고려하지 말자. 알아서 span하고 failover도 하고.

341 Glacier도 expedited 쓰면 대부분 5분 이내로 들어오지만, MUST retrieved는 아니다.

342 Abstain from infra control = NOT EC2 // Milisec=DBcaching or DynamoDB // HA,Scalable = Lambda + API Gateway

\*343 주의) 이미 CPU의 사용은 낮은 상황. CPU 사용량 따라 ASG 조정하는 것보다, 애초에 작은 메모리 쓰도록 새로운 inst. Launch하는게 낫다. 이 launch config는 only creation 단계에서만 생성 가능하다.

344 \*\*\*\*\*

345 20TB + bandwidth가 15Mbps = Snowball

346 reserved Inst.는 min 1년이다. 일주일동안 예약하고싶으면 ondemand capacity reservation

347 snapshot을 매번 찍어서 copy는 operational overhead가 크다. 여러 Region은 Read Replica

\*348 EFS is NOT for Windows. // tape나 별다른 얘기 없으면 SMB, NFS다 지원하는 File Gw

\*349 Key rotation = KMS+CMS조합 // credential, pw 관련 AUTO-rotation = Secrets Manager

\*350 PrivateLink와 ALB의 직접 연결은 불가능하다. PrivateLink는 NLB와. 다른 계정에서 VPC끼리 연결하고프면 VPC peering or Resource Access Manager를 택한다.

351 public subnet에 bastion host가 있고, 그 뒤 private subnet에 EC2들이 있다. On premise에서 들어오는 network의 external ip를 bastion host에 연결하고, bastion host의 private ip 주소로 ec2 ssh를 연결한다.

352 한 작업에 3개의 microservices가 있다. SNS+SQS의 fanout 전형.

353 특정 기간에만 spike = autoscaling + schedule

\*354 DynanoDB에 write 시에 spike나면 Create table 시 Autoscaling 켤 수 있다! On-demand capacity allocation같은 설정도 있지만 autoscale이 ondemand보다는 cost-effective.

355 Caching→Reading intensive // Read Replica→ Read + Write. 그냥 원본과 똑같이 serve한다.

356 Cloudfront 자체가 global하다.

357 VPC 내부에서 인터넷 연결 필요(반대는 X) → IPv4 = NATgw / IPv6 = egress-only igw

358 MySQL 쓰면서 AutoScale = MySQL-compatible Amazon Aurora Serverless 가능하다.

\*359 microservices를 하나하나 쪼갤때는 SQS생각해라. + Autoscale 은 Scale-out이다!!

360 cold data → S3 // HPC 필요한 hot data → FSx

361 익숙해지기. MySQL-compatible AmazonAurora Multi-AZ / SQL server stand. Edition multiAZ

362 EKS는 Bastion Host or Transit GW로 connection 할 수 있으며, Internet 피하려면 VPN

363 Lambda는 15분!!

\*364 DMS 속도 올리기 = Multi-AZ 끄기 + logging 끄기

365 S3 DR용으로 Cross-Region할때 CORS는 아무상관없다. Versioning은 켜져있어야 한다. 주의) 같은 region 내에서 옮길 때에도 Versioning은 켜져있어야 한다.

366 access 정도 모르면 Intelligent tiering

367 NAT gw보다 VPC endpoint가 싸구나.

368 worldwide + static = s3+ Cloudfront

369 Time-limited access to S3 = Presigned URL

370 Region 지켜져야하면 geolocation routing

\*371 EC2가 외부의 특정 service에 접근해야 할때 = NAT gw

372 S3와 CloudFront 모두 Serverless잖아. Fargate는 사실상 컨테이너용으로만 사용된다.

373 S3는 NFS지원 안한다. Concurrent access + NFS + Scalable = EFS

374 RDS proxy는 RDS작업 시 connection 안전하게, dataloss 등에 대비하기 위한 서비스이다. RDS 점검하는동안 Lambda가 꺼져서 데이터가 손실되는 것은 SQS에 저장하는 식으로 해결한다. proxy와는 다른 층위의 문제이다.

375 Least amount of work = NOT caching

376 주의) SQS가 MicroserviceB에게 보내는데, 4번 실패한 걸 DLQ로 보낸다. DLQ로 보내는 작업은 main SQS 차원에서 진행되고, 4번 visible하다는 게 실패했다는 뜻이다.

377 Spot Fleet = Spot + On-demand. 밤에만 큰 용량 필요하면, 밤만을 위해 reserve하긴 아깝다.

378 FTP는 파일 전송 국룰이지만 암호화 아예 X // Snowball과 DC는 72시간 내에 절대 불가

\*379 DynamoDB 알아서 multiAZ에 span한다.(한 지역 내) // S3도 알아서 multiAZ에 Replicate한다. 그래서 S3는 항상 Cross-Region Replication만 나오는 것. CRR은 DR용이다. // DynamoDB, S3, RDS 이렇게 있으면 가장 HA에 거리가 먼 것은 RDS이다.

380 별다른 단서 없으면 Kinesis DataStream + Kinesis Analytics 조합으로 가자. Firehose는 일단 Redshift랑..

\*381 in-memory DB = Memcached or Redis // HA + replication = Redis. // Redis와 Mem 모두 캐싱 뿐 아니라 in-mry DB로 사용할 수 있다. (NoSQL의 일종)

\*382 VPC에서 리소스간 Least privilege → security group id로 security group rule 정하는거. Nest 시키는 방식.

383 Private Virtual Interface(VIF)는 DC 필요로한다. VPC끼리 연결은 그냥 peering이면 충분하다.

384 참고) RDS replica 만들면 나중에 master로 승격시켜서 failover 대처 가능하다.

385 Window, AD외에 파일 저장에 별다른 단서 없으면 Fsx windows + AD 조합이 무난하다. (백업기능 O)

386 Not over Internet = VPC endpoint

387 나라마다 다른 distribution right을 지킨다 + 그에 따라서 라우팅 = Geolocation Routing의 use case // Distribution right 위해서 WAF로 접속을 막는 것은 좀 무모..

\*388 meta-data – hostname, events, security groups, iam 등등은 169.254.169.254/latest/meta-data/{iam}/info ~

389 쉬운 문제는 쉽게 생각하자. Root account를 secure하려면 password 강하고 MFA켜야지

390 \*\*\*\*\*

391 Credential → Secrets Manager. // KMS 로 첨에 encrypted key 생성하고 그걸 rotation

\*392 Monolith는 Micro service와 대립되는 개념으로, 하나로 모든게 묶여있어 scale out이 불가능하다. // SQS로 decouple하고 Lambda의 async. 특성으로 bottleneck 줄일 수 있을 듯.

393 S3 + CloudFront

394 “highest feasible Iops” = EC2 instance store. // EBS 따위가 비빌 수 없는 IOPS. 최대 12TB까지 지원한다.

395 주의) Storage Gateway를 통해서 S3에 올라간 파일은 EC2가 바로 볼 수 없다. S3의 EBS snapshot을 만들어서 볼 수 있다.

\*Storage Gateway 정리\*

Storage Gateway는 [S3 – S3 storage class] [S3 – EBS] [S3 – Tape archive] [FSX] 등의 길이 있다.

Storage Gateway에서 EBS/ EFS 에 바로 저장할 수 없다.

Storage Gateway에서 S3에 저장한 파일을 바로 EC2에서 볼 수 없다. EBS snapshot 형태로 만든 뒤 접근 가능하다.

예외로, Volume Gateway 방식에서 lifecycle 거치지 않고 곧바로 Glacier Deep Archive로 넣을 수 있다.

396 MySQL DB 그대로 multiAZ만 켜면 된다.

\*397 AWS Config는 Cloudtrail과 거의 유사한 듯. 로그와 audit에서 Cloudtrail 없으면 Config 고르자. 여기서도 “documenting any modification”// AWS Trusted Advisor는 인프라 최적화에 훨씬 포커스이다.

\*정리\*

CloudTrail-감시,로깅,누가했는지 // Config-감시,config 변화 있었는지, 수많은 managed rules 지원 (ex. SG 포트 체크)

// Cloudwatch-무엇이 발생하는지(보안과는 크게 관련 X) // Trusted Advisor-어디서 (비용)최적화 가능할지

398 Losing user request = SQS 필요 // DynamoDB에 Write한다 = DAX 크게 필요 X. 얘는 Read 위한 것이다.

399 Cost-effective + 용량 TB단위 + Deadline있고 + bandwidth 100Mbps = Snowball

400 Operation cost 고려 = serverless = Aurora + ECS Fargate 조합

401 ELB endpoint을 지역별로(ELB는 within region) 만들고 GA가 그걸 pointing하게 설정한다.

402 기간동안만 접근= pre signed URL // CloudFront의 presigned은 Max 없고, S3의 presigned는 Max 7days

403 EC2에서 돌리는 쿼리 해결하고자 한다. → DB를 RDS로 옮기고 Read Replica 생성하기

404 새로운 Region에서 EC2 replicate할때, EBS는 따로 고려 안해도 된다. AMI 스냅샷에 이미 EBS가 반영된다.

405 processing의 전통은 Batch이지만 // 여러 step이 강조되고, S3, API GW, CloudWatch 등으로 trigger 가능하면 StepFunction이 더 큰 차원이다.

406 Config는 Security group public open 여부 확인하는 managed rules 있다. “Restriced-ssh Config managed Rule”

407 EFS는 마지막 접근 시점을 기준으로 lifecycle (to IA) 결정한다. 2년 된 파일을 IA로 일괄적으로 못옮긴다. (마지막 접근한지 2년이 되어야 옮길 수 있음) // S3는 저장 기간만 보니까 가능하다.

\*408 S3 Key 관리하기는 귀찮은데, regulate는 필요하다 = KMS

\* S3 Key정리 \*

SSE-S3 - AWS가 data key와 master key 다 관리한다

SSE-KMS - AWS가 data key 관리하고, 내가 master key 관리한다. = Key 내용 자체는 AWS가 관리하지만, 누가 key에 접근하고, 어떤 policy를 만들고는 내가 결정하겠다!

SSE-C – 내가 둘다 관리한다.

\*409 각 Region에 사용하는 services가 퍼져있고, 대신 다 Tagged 되어있다. 한번에 labeled(tagged) resources들을 확인하는 방법은 → AWS Resource Groups Tag Editor. Global하게 tag를 관리할 수 있다.

410 Linux + HPC + accessible to all EC2 instances + fast file system = Fsx Lustre

411 Disk 없으므로 serverless. 10GB의 ML 모델을 Lambda에 올리긴 힘들어보인다. + Cost effective = ECS

412 DynamoDB 추적하는 DynamoDB Stream으로 SNS trigger하는 케이스

413 vpc peering 많으면 transit gateway

414 iam policy에서 action : “ds:Delete” = directory service를 Delete하는 action이다.

\*415 vCPU = thread. CPU 관련된 내용은 EC2 launch 단계에서 설정할 수 있다.

416 Scaling 필요없다 = DynamoDB나 Aurora, Replica 할 이유 없다. // Milisecond = caching. // Redis > Memch.

417 latency 기준 gp2는 millisec. io1은 submilli. 가격에 대한 단서가 나와야 정확한 문제일듯.

418 가장 secure은 최대한 접근 적게 하는 것. CIDR range로 주면 그 range 내의 다른 리소스도 접근 가능하다.

\*419 그럴듯한 답을 고르자! Oracle을 Aurora로 옮기면 당연히 더 좋지만, 이미 Oracle 기능을 쓰고 있으므로 Autoscaling만 켜는 게 더 현실적이다.

420 Onpremise + DC + Data replication(Migration) = DataSync // DataSync의 endpoint를 public service endpoint로 잡거나, VPC endpoint로 잡을 수 있다. VPC endpoint의 경우 public internet을 지나지 않아 더 secure하다. // Automated + Expedited도 Storage GW보다 DataSync인 이유이다.

421 DB의 Schema 모른다 + scalable하다 = DynamoDB with on-demand enabled

422 명확한 Fanout이 아님 + Decoupling = SQS

423 caching = S3나 DB에 접근 횟수 줄이니까 당연히 비용 절감 효과 있다.

424 EC2끼리 low latency + VPC관련X = Placement group 中 Clustering

425 UDP = NLB / non-realtional data =DynamoDB

426 규율 어기고 IAM policy를 단 사람 추적한다 = IAM permission은 대표적인 Config rule의 관리 대상이다.

427 HPC에 autoscale 쓰일 순 있지만 당연히 cluster placement가 더 HPC에 적합하다.

428 50TB + 2 weeks = Snowball

429 S3 storage 옮기기 위해 “처음으로”할 것은 analytics로 어떤 bucket들이 옮겨질 수 있는지 확인하는 것. 맥락을 잘 보자.

\*430 첨보는내용) Cloudfront는 serve할 Region을 제한함으로써 가격 줄일 수 있다.

431 HA + HTTPS + Security 케이스 → ALB on public subnet + EC2 on private subnet + CloudFront to ALB

432 AWS Config는 Resource 중 Tagging을 추적하는 기능도 있다. Resource Tag Editor가 베스트겠지만.

433 \*\*\*\*\* FT와 HA 엄밀하게 구분하는게 포인트인 문제인지 잘 모르겠다..

\*434 그림 참고. 한 vpc 내에서 [NLB + PrivateLink 조합]이 onpremise – vpn - ECS를 private ip로 연결할 수 있다.

435 key 관리는 KMS

436 Deep archive가 아닌 그냥 Glacier은 여전히 storage gw에서 s3로 옮긴 뒤 lifecycle 조정을 통해 보관해야한다.

\*437 WAF는 enable시키는 대상이지, CloudFront가 모든 리퀘를 WAF로 넣는 구조가 아니다. // OAI는 리퀘스트가 S3로 흘러들어가지 않고 무조건 CF 거치도록 만드는 효과가 있다. 따라서 CF + OAI + WAF 조합.

438 scale up 할 시간을 안다 = Schedule!

\*439 일단 key면 KMS를 거쳐야한다. KMS-CMK를 Secrets Manager에서 rotate할 순 있지만, Secrets Manager- CMK는 애초에 성립하지 않는 말이다.

440 on-premise Migration + frequent backup = DataSync

\*441 Create replica보다 매일 snapshot 찍는게 더 cost effective하다. Replica가 싼 작업이 아니구나..

\*442 Placement Group은 multiAZ 안된다.

443 NFS = Storage Gateway - File Gateway (SMB + NFS)

\* 파일시스템 정리 \*

EFS – NFS 지원. Concurrency가 주요 단서. Autoscaling 지원. 리눅스만 가능

File Gateway – NFS, SMB 모두 지원

Fsx Windows – SMB

Fsx Lustre – S3 DataLake용 / HPC / Container용 / POSIX-compliant

Fsx ONTAP – NFS, SMB, iSCSI 모두.

444 좋은 문제! HA를 위해 VPC에 AZ다르게 해서 pub와 private subnet 두개씩 만들고, pub에는 ALB를 둔다. 그중 EC2는 internet access되어야 하니까 NGW도 두고. 그림 참고.

445 EBS snapshot에서 recover할 때 무조건 encrypted 되게 하려면 → enable EBS encryption (개별 EBS단위, Region 단위 설정 가능하다.)

446 IAM user의 access key도 AWS Config에서 확인 가능하다. 삭제하고프면 EventBridge가 remove하는 Lambda 콜해서. 중간 매개체로 Trigger하는건 대개 EventBridge(=CloudWatch Events)가 담당하는 듯 하다.

\*447 Container = ECS or EKS // Opensource 가능 + “Cloud Agnostic” = EKS

448 S3에 담긴게 process 되어야 하면 → object-based event notification으로 람다가 process하도록 호출한다. 단순히 S3차원에서 람다 콜하면 되는 걸 굳이 Kinesis DS나 SNS를 쓸 필요 없다.

449 AutoScaling group does NOT span Region. 한 지역 내에서만.

\*450 여러 Region의 많은 ALB들의 IP를 firewall에 넣고싶다. 어케 이 ip를 구하나 → GA는 Static IP를 가진다는 점을 이용한다. GA를 각 Region마다 endpoint 뚫고 여기에 각 지역의 ALB들을 연결해주는 것!!

451 ACID + SQL + Regional Service outage 대비 = Aurora(rds) Global Database

452 Unique data = NOT caching // upload & download latency 고려 = S3 Transfer Acceleration // 확실한 단서 있을 때에는 S3 web같이 애매하게 맞는건 지나쳐도 될 것 같다.

453 Automated = NOT snowball // backup 중 local에서 파일에 접근하고싶다 = Storage Gateway – stored volume gateway. 얘가 local에 전부 store하면서 보내는 방식이다.

454

\* Storage Gateway와 DataSync \*

Storage Gateway는 좀 더 hybrid한 성격을 갖는다 – Migrated data에 대해 On-premise에서의 low-latency access 가능

DataSync는 Migration과 Sync에 특화되었다. - 옮기는 속도나 효율은 당연히 SG보다 낫다. 다만 이후 On-premise에서 migrated data에 접근하는 용도로는 쓰이지 않는다.

455 \*\*\*\*\*Region별로 있는 ALB를 failover하려면.. Region간 Route53에서 health check 켜면 된다.

\*456 Account 끼리 접근하려면, 접근 될 account의 IAM Role - Trust policy에서 접근 할 account를 열어주자.

457 여러 ec2에서 공유 = EFS or Fsx // 이전 서버가 Windows 기반 = NOT EFS(얜 smb X)

458 \*\*\*\*\*

459 HPC + concurrent access = Lustre

460 Reduce management cost with maintaining = Cost 자체를 줄이는게 아니라 overhead를 줄이는 것이다. → q비슷한 가격대라면 이왕이면 EC2 안쓰는게 낫다.

\*461 IAM은 다른 계정에 직접적으로 권한을 못주지만 // SQS policy는 다른 계정이 바로 SQS 접근하도록.

462 차단하고픈 IP가 있고 WAF 켜져있으면 WAF에 IP match condition 추가하면 된다. // CloudFront는 subnet 안에 있는게 아니기 때문에 NACL 적용 안된다 // \* 주의 \*) ALB 뒤에 있는 EC2 instances에 NACL을 설정하는 것은 옳지 않다. ALB 뒤에 있는 EC2에는 ALB의 ip가 전달될테니까. 차단하고자 하는 IP는 ALB 전에 처리되어야 한다.

463 인터넷에 open 되진 않아도, 일방적으로 3rd party로부터 업데이트 받고싶은 private subnet 친구에겐 NGW

464 ECS와 DB를 두고 ALB를 scale할 이유가 없다. // Fargate가 나온 시점부터 ECS의 AZ는 신경쓰지 말자.

465 SG에서는 Deny 못해 // IP 막으려면 당근 Inbound를 막아야지 // NACL에서는 낮은 순서 룰이 우선순위!

466 Linux + shared file + No HPC = EFS

\*467 Aurora의 Multi-AZ standby는 read-replica와 사실상 동일한 역할이다. 다른 RDS의 standby는 serve 하지 못하지만, Aurora의 standby는 servable하다. // 이미 multiAZ deployed면 따로 read-replica를 create할 필요 없다.

468 S3도, RDS도, 직접 event가 있어서 notification 보낼 수 있으면 굳이 Lambda를 한번 더 call할 필요 없다.

469 포인트 잘 잡자. HA만 언급되어있으므로 subnet 같은 distractor는 볼 필요도 없다.

470 PrivateLink는 VPC endpoint를 포함(제공)하는 개념

\*471 Client들은 whilelist IP에만 접근 가능하다 → 고정된 IP가 필요하다 = Static, Elastic IP // ALB는 고정 IP 설정 못한다. // NLB는 고정 IP 설정 가능하다. // \*주의\* ALB에 고정 IP 하려면 ALB앞에 Lambda로 NLB를 연결하던지, 아니면 GA를 붙이던지. (이게 450번 문제 내용)

\*\*\*\*472 \*\*\*\*\* 일단은 더 직접적인 VPC gateway endpoint가 PrivateLink보다 나을 것 같다. (S3와 Dyno만 가능)

473 SQS에 다른 account 권한 줄때 SQS policy가 가장 효과적 // Cross account role과 trust document로도 가능한데, 다만 Least privilege에 주의하자.

474 EC2간 성능 = cluster placement

475 예측가능하면 무조건 scheuled

476 NLB + GA 조합. Static ip 쓰고 UDP/TCP 지원하고. // 참고) CF와 GA 모두 디도스 방어용 Shield와 결합.

477 TCP 트래픽만 따로 빼오고 싶다 = NLB로 보내준다.

\*478 좋은문제) Trusted S3 buckets의 ARN만 허용하는 S3 gateway endpoint를 만듦으로써 Trusted buckets만 traffic 허용 // Endpoint를 허용하는 bucket poliy를 모든 Trusted bucket에 붙이는 방법도 가능하지만 매우 비효율적이다.

\*\*\*\*479 rolling update deploy mode 다시 \*\*내일 여기서부터 다시

480 physical DB를 global 유지하면서 옮기려면 Aurora global db / Read Replica / DynamoDB global.

\*\*\*\*481 Route53 activae-passive failover?

482 Kinesis Firehose에서 직접 DynamoDB로 데이터 못넣는다. 람다가 있어야한다. + 저장/process 순서 유의하자

483 Bucket 차원에서 여러 account의 접근 허용하려면 S3 bucket policy 바꾸는게 가장 쉬운 방법이다.

484 EC2 간 번갈아 접근하면 당연히 ebs가 다르니까 다른 정보가 나온다. EFS로 합쳐준다.

485 Millions of transactions per second = Instance Store. 이거 말고는 Million 단위 나올 수 없다.

486 웹 / SQS기반 middle / DB 중 processing 이 오래걸린다 = Middle이 가장 의심된다. // SQS 깊이에 따라 EC2 autoscaling 기준 정하는건 단골문제.

487 S3 lifecycle transition

488 Shield Advanced = Shield + WAF. 보안쪽으로는 최강이다.

489 크게크게 보자. Eliminate need for managing instances, clusters = Serverless.

490 Organization 차원에서 Account에 권한 제어 = Service Control Policy

491 단순히 디도스 막는거면 WAF 차원에서 Rate-based rules면 충분하다. Shield도 쓸 수 있다.

492 Little human intervention = ECS + Fargate // highly accessible = MultiAz

493 목표가 HA인데, SingleAZ EC2에 굳이 MultiAZ ALB를 둘 이유가 없다. EC2가 속한 AZ가 fail하면 끝이니까. 그냥 평범한 MultiAZ Autoscaling문제.

494 Cryptocurrency-related assaults = 제대로 된 지능적 공격(디도스 수준 X) → GuardDuty // Inspector는 테스트용.

\*495 RedShift는 Storage Service가 아니다. Current Environment를 그대로 옮기고 싶다 = Storage Gateway

496 Continue using current license + per-socket per-core = HOST // predictable = Reserved

\*\*\*\*\*497

498 iam policy DB의 경우 ~:table/{DB name} 이런 식으로 특정 DB의 table 접근 권한 생성 가능하다.

499 Single AZ RDS의 SPOF 없애기 위해 RDS Multi-AZ deploy 켜자. Snapshot 찍어서 Multi-AZ 버전으로 새로 restore도 가능한데 이 경우 downtime 생기겠지. 처음 방법의 수작업 버전에 불과하다.

500 NACL stateless SG stateful. // SSL만 허용한다 = HTTPS만 허용한다 = 443만 열어준다 // public subnet에 웹 올리고 private에 DB 올리고 SG와 NACL 포트 문제는 정형화되어있다.

501 TCP → NLB // UDP → NLB // 맨 앞에서 GA가 이 NLB들로 연결시켜준다. // GA + NLB 조합

502 특정 bucket의 “내용”을 read, write하는 권한이래도, 해당 bucket을 List하는 권한 있어야 accessible하다. → ex) s3:ListBucket for s3::{bucketname} // s3 GetPutDeleteObj for s3::{bucketname}/\*

503 NLB → EC2의 보안 강화하려면 NLB의 TLS listener 기능 켠다. Server certificate 추가한다.

504 S3 key rotation + user와 admin의 duty 분리 → KMS + CMK 조합

505 Windows + AD domain + high lvl of throughput & Iops = FSX windows

NLB + Privateink로 ECS private하게 연결하기

https://docs.aws.amazon.com/prescriptive-guidance/latest/patterns/access-container-applications-privately-on-amazon-ecs-by-using-aws-privatelink-and-a-network-load-balancer.html

https://d1.awsstatic.com/cloud-storage/product-page-diagram\_AWS-Storage-Gateway\_HIW%402x.6df96d96cdbaa61ed3ce935262431aabcfb9e52d.png

GA냐 Route53이냐

https://www.quora.com/Is-the-AWS-multi-region-load-balancing-configuration-possible

https://aws.amazon.com/backup/?whats-new-cards.sort-by=item.additionalFields.postDateTime&whats-new-cards.sort-order=desc

https://docs.aws.amazon.com/IAM/latest/UserGuide/reference\_policies\_examples\_aws\_deny-requested-region.html

there is other new question about single point of failure. structure consist of two vpcs. vpc peering between them. the first linked with two direct connect to outside the second with single vpn two one application in on-prim. the solution I think is two add vpn. remember that vpc peering doesn't have single point of failure.

https://www.linkedin.com/pulse/high-availability-vs-fault-tolerance-jon-bonso

Q. NAT GW가 인터넷 말고 그냥 resrc끼리 연결할때도 쓰이나?

꼭 인터넷 아니어도 VPC 외부 서비스면 된다. 용도는 private subne → 외부는 되지만 그 반대는 안된다.

Q. VPC endpoint는 인터넷에 연결 가능해?

아니 그냥 services에만. Gateway Endpoint는 S3, Dynano

Q. Gateway LB endpoint와 Interface Endpoint는?

한 vpc 내에서 private, public subnet 끼리는 그냥 security group으로 열어주면 끝

private이 internet이나 다른 연결 원하면 NAT gw

private이 다른 resource (subnet 밖의)와 연결되려면 endpoint

Clarify할 개념

~~VPN의 역할과 NGW와의 차이~~

~~Privatelink \*2~~

~~bastion host~~

~~data anal vs firehose~~

~~virtual private gateway 22~~

~~private virtual interface~~

~~elastcache와 s3가 같이 쓰일 수 있는가?~~

~~Throttling~~

~~DynamoDB secondary idx 대체 뭔데~~

~~service control policy - Organization의 account 권한 제어한다.~~

~~RAID 다시~~

~~EFS vs Fsx~~

~~Dedicated Hosts~~

~~Macie~~

~~storage gateway VS Datasync~~

~~SaaS~~

~~Resource Access Manager 역할 다시~~

~~GuardDuty VS Shield VS WAF VS Shield advanced (488 참고)~~

~~stateless two-tier application. stateless가 여기서 무슨뜻..?~~

~~host 다시. Instance와 달리 소켓 제공한다는 것 같은데.(496)~~

~~cloudfront 가 nlb와 못쓰이는건지, 그냥 잘 안쓰이는건지 확실히 알아놓기~~

~~datalake~~

~~VPC flowlog~~

~~Storage Gw is encrypted in transit? Snowball?~~

~~VISIBILITY TIMEOUT (SQS)~~

~~redshift spectrum, athena, quicksight, emr notebook, kibana~~

~~Q. TLS termination on ELB ALB?~~

<https://aws.amazon.com/storagegateway/volume/>

~~AWS managed CMK와 Customer managed CMK. 전자가 성립해?~~

~~DNS A Record CNAME + 455 (Route 53 활용문제)~~

~~479 rolling update deploy mode - ECS~~

~~route 53 active passive~~

S3 policy(lifecycle) 누구보다 완벽하게!!