Contents

[Contents 1](#_Toc518380015)

[IAM 3](#_Toc518380016)

[What does IAM give you? 3](#_Toc518380017)

[Critical terms 3](#_Toc518380018)

[IAM summary 3](#_Toc518380019)

[S3 4](#_Toc518380020)

[S3 basics: 4](#_Toc518380021)

[S3 objects consists of: 4](#_Toc518380022)

[S3-Storage Tiers/Classes: 4](#_Toc518380023)

[S3 charges: 4](#_Toc518380024)

[Cross region replication 4](#_Toc518380025)

[Lifecycle management: 5](#_Toc518380026)

[CloudFront(CDN) 5](#_Toc518380027)

[Security: 5](#_Toc518380028)

[Encryption: 5](#_Toc518380029)

[Storage Gateway 6](#_Toc518380030)

[S3 transfer acceleration 6](#_Toc518380031)

[S3 static websites 6](#_Toc518380032)

[CLI: 6](#_Toc518380033)

[EC2 7](#_Toc518380034)

[EC2 Options 7](#_Toc518380035)

[Instance types 7](#_Toc518380036)

[AMI (Amazon Machine Instances) selecting 8](#_Toc518380037)

[EBS vs Instance Store 8](#_Toc518380038)

[EBS 9](#_Toc518380039)

[EBS volume types 9](#_Toc518380040)

[EBS Volume Snapshots 9](#_Toc518380041)

[Volumes vs Snapshots – Security 10](#_Toc518380042)

[Security groups 10](#_Toc518380043)

[RAID, Volumes & Snapshots 10](#_Toc518380044)

[Windows instances 10](#_Toc518380045)

[Load Balancers 11](#_Toc518380046)

[Cloud Watch 11](#_Toc518380047)

[EC2 Placement groups 11](#_Toc518380048)

[EC2 CLI 12](#_Toc518380049)

[EFS 12](#_Toc518380050)

[LAMBDA 13](#_Toc518380051)

[POLLY PROJECT 14](#_Toc518380052)

[DNS, Route 53 16](#_Toc518380053)

[Route53 Routing policies 16](#_Toc518380054)

[Databases 17](#_Toc518380055)

[Relational databases types OLTP 17](#_Toc518380056)

[Non relational DB 17](#_Toc518380057)

[Data warehousing – OLAP 17](#_Toc518380058)

[Elasticache 17](#_Toc518380059)

# IAM

*Essentially, IAM allows you to manage users and their level of access to the AWS console.*

## What does IAM give you?

* Centralized control of your AWS account
* Shared access to your AWS account
* Granular permissions
* Identity Federation (including Active directory, Facebook, Linkedin etc)
* Multifactor Authentication
* Provide temporarily accessfor users/devices and services where necessary
* Allows you to set up your own password rotation policy
* Integrates with many different WAS services
* Suppoorts PCI DSS Compliance

Critical terms***:***

{“Version”:”2012-10-17”,

”Statement”:

[{“Effect”:”Allow”,

”Action”:”\*”,

”Resource”:”\*”}]

}

* Users – end users (think people)
* Groups – a collection of users under one set of permissions
* Roles – you create roles and can then assign to AWS resources
* Policies – a document that defines one or more permissions

IAM summary***:***

IAM is universal – does not apply to regions

- Root account is the account during the account setup and has admin privileges

- New users have no permissions when first created

- New users assigned Access Key ID and Secret Access keys

- Key ID and Secret Access keys cannot be used for login but for access using API

And CLI. You will view them once, if loose, need to be recreated.

- Always setup MFA (Multifactor authentication) on root acc.

- You can create and customize your own password rotation policies.

# S3

*S3 provides developers and IT teams with secure, durable, highly scalable object storage. Amazon S3 is easy to use, with a simple web services interface to store and retrieve any amount of data from anywhere on the web*

## S3 basics:

* Object-based – i.e. allows you to upload files.
* Files can be from 0 bytes to 5TB
* There is unlimited storage
* Files are stored in Buckets
* S3 is universal namespace. Names must be unique globally.
* https://s3-**eu-west2**.amazonaws.com/**karalov**
* When you upload a file to S3, you will receive a HTTP 200 code if the upload was successful
* Read after Write consistency for PUTS of new Objects
* Eventual consistency for overwrite PUTS and DELETES (can take some time to propagate)
* S3 is a simple Key-value store, object based

## S3 objects consists of:

* Key (name of the object)
* Value (data made from bytes)
* Version ID
* Metadata (data about data you storing)
* Sub resources: Access Control Lists, Torrent

## S3-Storage Tiers/Classes:

99.99999999% durability for all classes

1. **S3 Standard**: 99.99% availability, stored redundantly across multiple devices in multiple facilities, and is designed to sustain the loss of 2 facilities concurrently.

2**. S3 – IA**: (Infrequently Acessed): for data that is accessed less frequently, but requires rapid access when needed, Lower fee than S3, but you’re charged retrieval fee. Avail. 99.9%,

3. **S3 One Zone** – IA: a lower cost optionfor infrequently accessed data, but do not require the multiple Availability Zone data residence. Availability 99.5%

4. **Glacie**r: very cheap, but used for archival only. Expedited, standard or bulk. A standard retrieval time takes 3-5 hours.

## S3 charges:

* Storage
* Requests
* Storage management pricing
* Data transfer pricing

Cross region replication**:**

*Cross-region replication is a bucket-level configuration that enables automatic, asynchronous copying of objects across buckets in different AWS Regions.*

* Versioning must me enabled on both the source and destination buckets
* Regions must be unique
* Files in an existing bucket are not replicated automatically. All subsequent updated files will be replicated automatically.
* You cannot replicate to multiple buckets or use daisy chaining
* Delete markers are replicated
* Deleting individual versions or delete markers will be not replicated.

## Lifecycle management:

* Can be used in conjunction with versioning.
* Can be applied to current versions and previous versions.
* Following actions can be done:
* Transition to standard IS storage class (128kb and 30 days after the creation date)
* Archive to Glacier (30 days after IA)
* Permanently delete

## CloudFront(CDN)

Amazon CloudFront is a global content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to your viewers with low latency and high transfer speeds. CloudFront is integrated with AWS – including physical locations that are directly connected to the AWS global infrastructure

* Edge location – location where contant will be cached. This is separate to an AWS region/AZ. Not read only, they are writable
* Origin – origin of all files, that the CDN will distribute. Thi scan be either as S3 bucket, an EC2 instance, Elastic Load balancer or Route53
* Distribution – a collection of Edge locations.
* Web distribution – used for websites
* RTMP – used for media streaming
* Objects are cached for the life of TTL (Time to Live)
* You can clear cached objects, but you will be charged.

## Security:

* All newly created buckets are PRIVATE by default
* You can setup access control to your buckets using:
* Bucket policies
* Access control lists
* S3 buckets can be configured to create access logs which log requests made to the S3 bucket. This can be done to another bucket.

## Encryption:

* In transit (SSL/TLS)
* AT Rest:
* Server side encryption:
  + S3 managed keys – **SSE-S3**
  + AWS key management service, managed keys – **SSE-KMS**
  + With customer provided keys – **SSE-C**
* Client side encryption

## Storage Gateway

*AWS Storage Gateway is a hybrid storage service that enables your on-premises applications to seamlessly use AWS cloud storage. You can use the service for backup and archiving, disaster recovery, cloud bursting, storage tiering, and migration. Your applications connect to the service through a gateway appliance using standard storage protocols, such as NFS and iSCSI.*

* File gateway – for flat files, stored directly on S3
* Volume gateway:
* Stored volumes – entire dataset is stored on site and is asynchronously backed up to S3
* Cached volumes – entire dataset is stored on S3 and the most frequently accessed data is cached on site
* Gateway Virtual tape library (VTL)
* Used for backup and uses popular backup applications like NetBackup, Backup Exec, Veeam etc.

## S3 transfer acceleration

Utilizes the CloudFront edge network to accelerate uploads to S3. Instead of uploading directly to S3 bucket, you can use a distinct URL to upload directly to an edge location which will then transfer that file to S3.

## S3 static websites

* You can use S3 to host static websites
* Serverless
* Very cheap, scales automatically
* Statis only, cannot host dynamic sites

## CLI:

>aws configure

>aws s3 ls

>aws s3 cp –recursive s3://karalov/myfolder . --region us-west-2

check region:

aws s3api get-bucket-location --bucket <BUCKET>

to have access from ec2 instance to s3 of the same aws account, you should assign role with S3 access to the instance in EC2 instances dashboard

# EC2

Elastic compute cloud is a web service that provides resizable compute capacity in the cloud. Amazon EC2 reduces the time required to obtain and boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change.

## EC2 Options

* On Demand – allows you to pay a fixed rate by the hour (or by second) with no commitment.
* perfect for users that want the low cost and flexibility of Amazon EC2 without any up-front payment or long term commitment
* applications with short term, spiky, or unpredictable workloads that cannot be interrupted
* applications being developed or tested on Amazon EC2 for the first time.
* Reserved – provides you with a capacity reservation, and offer significant discount on the hourly charge for an instance. 1 year to 3 years terms.
* Applications with steady state or predictable usage
* Applications that require reserved capacity
* Users can make up-front payments to reduce their total computing costs even further
* Standard RI – up to 75% off on demand
* Convertible RI – up to 54% off on demand – capability to change the attributes.
* Scheduled RI available to launch within the time window you reserve
* Spot – enables you to bid whatever price you want for instance capacity, providing for even greater savings if your applications have flexible start and end times.
* Applications that have flexible start and end times
* Applications that are only feasible at very low compute prices
* Users with an urgent need for large amounts of additional computing capacity
* Dedicated hosts – physical EC2 server dedicated for your use. Dedicated hosts can help you reduce costs by allowing you to use your existing server-bound software licenses.
* Useful for regulatory requirements that may not support multitenant virtualization
* Great for licensing which does not support multi-tenancy or cloud deployments
* Can be purchased on demand (hourly)
* Can be purchased as a reservation for up to 70% off the on demand price

## Instance types

FIGHT DR MCPX

F – FPGA

I – IOPS

G – Graphics

H – High Disk Throughput

T – cheap general purpose

D – density

R- RAM

M – main choice for general purp. Apps

C – compute

P – Graphics (pics)

X – extreme memory

## Related imageAMI (Amazon Machine Instances) selecting

* Based on region
* Operating system
* Archirecture (32/64bit)
* Launch permissions
* Storage for root device (Root Device Volume)
* Instance store (EPHEMERAAL STORAGE) – cannot be stopped, terminated only
* EBS backed volumes

## EBS vs Instance Store

All AMIs are categorized as either backed by Amazon EBS or backed by instance store

***For EBS Volumes:***

- The root device for an instance launched from the AMI os an Amazon EBS volume created from an Amazon EBS snapshot.

***For Instance Store Volumes:***

- The root device for an instance launched from the AMI is an instance store volume created from a template stored in Amazon S3.

* Instance Store Volumes are sometimes called Ephemeral Storage.
* Instance Store volumes cannot be stopped. If the underlying host fails, you will lose your data.
* EBS backed instances can be stopped. You will not lose the data on this instance if it is stopped,
* You can reboot both, you will not lose your data,
* By default, both Root volumes will be deleted on termination, however with EBS volumes, you can tell AWS to keep the root device volume.

## EBS

Allows to create storage volumes and attach them to EC2 instances. Once attached, you can create a file system on top of these volumes, run a database, or use them in any other way you would use a block device. Amazon EBS volumes are placed in a specific AZ, where they are automatically replicated to protect you from the failure of a single component.

## EBS volume types

* General purpose SSD(GP2)
* General purpose, balances both price and performance.
* Ratio of 3 IOPS per GB with up to 10,000 IOPS and the ability to burst up to 3000 IOPS for extended periods of time for volumes at 3334 GB and above.
* Provisioned IOPS SSD (IO1)
* Designed for I/O intensive applications such as large relational or NoSQL databases.
* Use if you need more than 10,000 IOPS
* Can provision up to 20,000 IOPS per volume
* Throughput optimized HDD (ST1)
* Big data
* Data warehouses
* Log processing
* Cannot be a boot volume
* Cold HDD (SC1)
* Lowest cost storage for infrequently accessed workloads
* File server
* Cannot be a boot volume
* Magnetic (Standard)
* Lowest cost per gig of all EBS volume types that is ***bootable***. MV are ideal for workloads where data is accessed infrequently, and applications where the lowest storage cost is important.

**EBS volume** must be in the same AZ that EC2 instance. In case we need it to be in different zones, we have to create a snapshot, create another volume from it and choose type and Availability Zone.

## EBS Volume Snapshots

* Volumes exist on EBS - Virtual hard disk
* Snapshots exist on S3
* Snapshots are point in time copies of Volumes
* SS are incremental – only the blocks that have changed since the last snapshot are moved to S3
* To create a snapshot for Amazon EBS volumes that serve as root devices, it is better to stop the instance before taking the snapshot, however, you can take SS while th e instance running.
* You can create AMI’s from EBS-backed instances and Snapshots
* You can change EBS volume sizes on the fly, including changing the size and storage type.

## Volumes vs Snapshots – Security

* Snapshots of encrypted volumes are encrypted automatically.
* Volumes restored from encrypted snapshots are encrypted automatically
* You can share snapshots, but only if they are unencrypted – can be shared with other AWS accounts or made public
* To encrypt a boot volume:
* Stop instance
* Take a snapshot of the instance
* Copy snapshot
* Use encryption while copied
* Create image from the snapshot

## Security groups

Security group is a virtual firewall. 1 instance can have multiple security groups.

* All inbound traffic is blocked by default
* All outbound traffic is allowed
* Changes to security group take effect immediately
* You can have any number of EC2 instances within a security group
* Security Groups are STATEFUL – if you create inbound rule allowing traffic in, that traffic is automatically allowed out again.
* You cannot block specific IP addresses using SG, instead use Network Access Control lists.
* You can specify allow rules and not deny rules

## RAID, Volumes & Snapshots

You need to create RAID when you don’t have enough disk IO and you have to combine several EBS volumes in array.

* RAID – Redundant Array of Independent Disks
* RAID 0 – Striped, No Redundancy (one disk filed, all array filed). Good performance
* RAID1 – Mirrored, Redundancy. Disk failed, you can continues working.
* RAID 5 – Good for reads, bad for writes, AWS does not recommend ever putting RAID5 on EBS
* RAID10 – RAID1 + RAID0, stripped & mirrored, good redundancy, good performance.
* To get a snapshot of RAID disks system:
* Stop the application from writing to disk
* Flash all caches to the disk

To accomplish above, do one of the following:

* Freeze file system
* Unmounts RAID array
* Shutdown the associated EC2 instance

## Windows instances

To get a password for “Administrator” user, use Actions -> Get Windows password in instances dashboard

To create a RAID volume on Windows – on disk management tool delete volumes for all available disk drives except of c and by right click choose new <Simple/Spanned/Stripped etc> volume.

## Load Balancers

* Three types of Load Balancers:

- Application Load Balancers (Layer 7)

- Network Load balancers (Layer 4, high performance)

- Classic Load Balancers (Elastic LB)— previous generations.(Usually layer 4, sometime layer 7 with sticky sessions)

* Error 504 means the gateway gas times out. This means that the application not responding within the idle timeout period. It may be Web Server or Database server.
* If you need Ipv4 address of your end user, look for the X-Forwarded-For header.
* Instances monitored by ELB are reported as InService and OutofService
* Health Checks check the instance health by talking to it.
* Have their own DNS name, you are never given an IP addresses.

**Auto Scaling group** – you can setup group of instances with the same content using bootstrap script and set auto scaling (increase/decrease number of instances according to load or other conditions). Load balancer will spread traffic to instances evenly.

## Cloud Watch

* Standard monitoring = 5 minutes
* Detailed monitoring = 1 minute
* Dashboards – creates dashboards to see graphs for metrics of AWS environment
* Alarms – allows you to set alarms that notify you when particular thresholds are hit.
* Events – helps you to respond to state changes in your AWS resources.
* Logs – help to aggregate, monitor and store logs

## EC2 Placement groups

Two types of PG:

* Clustered PG
* Spread PG

The name you specify for a particular placement group must be unique within your AWS account.

**Clustered placement group**

CPG is a grouping of instances within a *single AZ*. They are recommended for applications that need low network latency, high network throughput or both.

**Spread PG**

SPG is a group of instances that are each placed on distinct underlying hardware. They are recommended for applications that have a small number of critical instances that should be kept separate from each other. Spread placement group can be placed in multiple Availability Zones.

* Only certain instances can be launched in to a clustered PG.(Compute optimized, GPU, Memory optimized, storage optimized)
* AWS recommend homogenous instances within placement groups.
* You can’t merge placement groups
* You can’t move an existing instance into a placement group. You can create an AMI from your existing instance, and then launch a new instance from the AMI into a placement group.

## EC2 CLI

>aws ec2 help

>aws ec2 describe-instances

>aws ec2 terminate-instances –instance-ids i-023nhjfdsauje

EC2 METADATA

>curl http://169.254.169.254/latest/meta-data/

ami-id

ami-launch-index

ami-manifest-path

block-device-mapping/

hostname

iam/

instance-action

instance-id

instance-type

local-hostname

local-ipv4

mac

metrics/

network/

placement/

profile

public-hostname

public-ipv4

public-keys/

reservation-id

security-groups

services

>curl http://169.254.169.254/latest/meta-data/public-ipv4

18.191.226.229

>curl http://169.254.169.254/latest/user-data/ - init bootstrap bash script that was added during instance creation.

## EFS

*Amazon Elastic File System is a file storage service for Amazon EC2 instances. Amazon EFS is easy to use and provides a simple interface that allows you to create and configure file systems quickly and easily. With Amazon EFS, storage capacity is elastic, growing and shrinking automatically as you add and remove files, so your applications have the storage they need, when they need it.*

You can create several EC2 instances and mount to the same EFS via NFS (Network FS). They should be in the same region, but may be in different AZ. Should have the same security group.

# LAMBDA

AWS Lambda is a compute service where you can upload your code and create a Lambda function. AWS Lambda takes care of provisioning and managing the servers that you use to run the code. You don’t have to worry about operating systems, patching, scaling, etc. You can use Lambda in the following ways:

* As an event-driven compute service where AWS Lambda runs your code in response to events. These events could be changes to data in an Amazon S3 bucket or an Amazon DynamoDB tables
* As a compute service to run your code in response to HTTP requests using Amazon API Gateway or API calls made using AWS SDKs.

**Lambda Features:**

* No Servers, continuous scaling, cheap
* Scales out(not up) automatically
* Functions are independent, 1 event = 1 function
* Serverless
* Can trigger other lambda functions, 1 event can = x functions if functions trigger other functions
* Architectures can get extremely complicated, AWS X-ray allows you to debug what is happening.
* Lambda can do things globally, you can use it to back up S3 buckets to other S3 buckets etc



# C:\python\py-conspect\aws_polly\schema.pngPOLLY PROJECT



* Create two S3 buckets:

1. karalovlambda.com (for webpage, apply JSON bucket policy bucketpolicypermissions.json)

2. karalovmp3 (for storing created mp3 files)

* Create DynamoDB table “new\_topics”, partition key – “id”
* Create SNS topic “new\_posts”
* Create new IAM role, first create new policy for Lambda service myPollyPolicy, using JSON lambdapolicy.json, then create new Role MyPollyRole , attach policy MyPollyPolicy
* Create 3 Lambda functions.

1. PostReader\_NewPosts (Python, use sec. role created before - MyPollyRole)

Code: newposts.py

Add environment variables TABLE\_NAME and SNS\_TOPIC (below lambda code)

2. PostReader\_ConvertToAudio

Code: convertoaudio.py

Role: MyPollyRole

Variables: BUCKET\_NAME(mp3 bucket, karalovmp3), TABLE\_NAME(new\_topics)

Trigger: SNS , topic new\_posts

3. PostReader\_GetPostFromDB

Code: getposts.py

Variables; TABLE\_NAME(new\_topics)

* Create API PostReaderAPI with two methods

1. Method POST – attached to lambda function PostReader\_NewPosts

2. Method GET – attached to lambda PostReader\_GetPostFromDB

* In API dashboard, Actions - > enable CORS to enable cross servers API calls
* In the GET method parameters (click on GET Method request on the method diagram)
* Add URL query string parameters: postId

\* In the GET Integration Request parameters (click on GET Integration Request on the method diagram)

- Add Body mapping templates:

+ When are no templates defined

+ Add mapping template

+ Content type: application/json

+ Click on generated link “application/json”

Paste code from file mappings.json

* Deploy API (Return to resources, press on “/” from actions choose deploy API)
* Copy to clipboard generated API URL
* In the code package, update API\_ENDPOINT variable in the file scripts.js with the API URL
* In AWS console open S3 service, upload following files to the bucket that is defined for static web page (karalovlambda.com):
* scripts.js
* index.html
* styles.css

# DNS, Route 53

* Elastic Load Balancers do not have pre-defined IPv4 addresses, you resolve to them using a DNS name.
* Understand the difference between an Alias Record and a CNAME
* Given a choice, always choose an Alias Record over a CNAME (Alias is free of charge).

## Route53 Routing policies

* ***simple***

default routing policy when you create a new record set, This is the most commonly used when you have a single resource that performs a given function for your domain, for example, one web server that server content http://cocktails-r.us website

* ***weighted***

Let you split your traffic based on different weights assigned. For example, you can set 10% traffic to go to US-EAST and 90% to go to EU-WEST-1

* ***latency***

Latency based routing allows you to route your traffic based on the lowest network latency for your end user (ie which region will give them the fastest response time). To use latency-based routing you create a latency resource record set for the Amazon EC2 (or ELB) resource in each region that hosts your website, When Amazon Route 53 receives a query for your site, it selects the latency resource record set for the region that gives the user the lowest latency. Route 53 then responds with the value associated with that resource record set.

* ***failover***

Failover routing policies are used when you want to create an active/passive set up. For example you may want your primary site to be in EU-WEST-2 and your secondary DR site in SOUTHEAST-2. Route 53 will monitor the health of your primary site using a health check. A health check monitors the health of your end points.

* ***geolocation***

Geolocation routing lets you choose where your traffic will be sent based on the geographic location of your users ( ie the location from which DNS queries originate). For example, you might want all queries from Europe to be routed to a fleet of EC2 instances that are specifically configured for your European customers. These servers may have the local language of your European customers and all prices are displayed in Euros.

# Databases

## Relational databases types OLTP

Tables similar to excel tables. Cons – rows must have same number of fields

Instance-db account-table-row-field

* SQL Server
* Oracle
* MySQL Server
* PostgreSQL
* Aurora
* Maria DB

## Non relational DB

* Dynamo DB

NR Database structure:

* + Collection (account)
  + Document (table)
  + Key-value pairs (data)
* Dynamo DB – JSON format

## Data warehousing – OLAP

* RedShift

Used for business intelligence. Tools like Cognos, Jaspersoft, SQL Server reporting services, Oracle hyperion, SAP NetWeaver. Used to pull in very large and complex data sets. Usually used by management to do queries on data. (Such as current performance vs targets etc)

OLTP vs OLAP

Online Transaction Processing differs from Online Analytics Processing in terms of the types of queries you will run.

***OLTP example:***

* Order number 2120121

Pulls a row of data such as Name, Date, Address to deliver to, and delivery status.

***OLAP example:***

Net profit for EMEA and Pacific for the Digital radio product.

* Pulls in large numbers of records
* Sum of radios sold in EMEA.
* Sum of radios sold in pacific.
* Unit cost of radio in each region.
* Sales price of each radio.
* Sales price – unit cost.

Data warehousing databases use different type of architecture both from a database perspective and infrastructure layer.

## Elasticache

- in memory caching

ElastiCache supports two open-source in-memory caching engines:

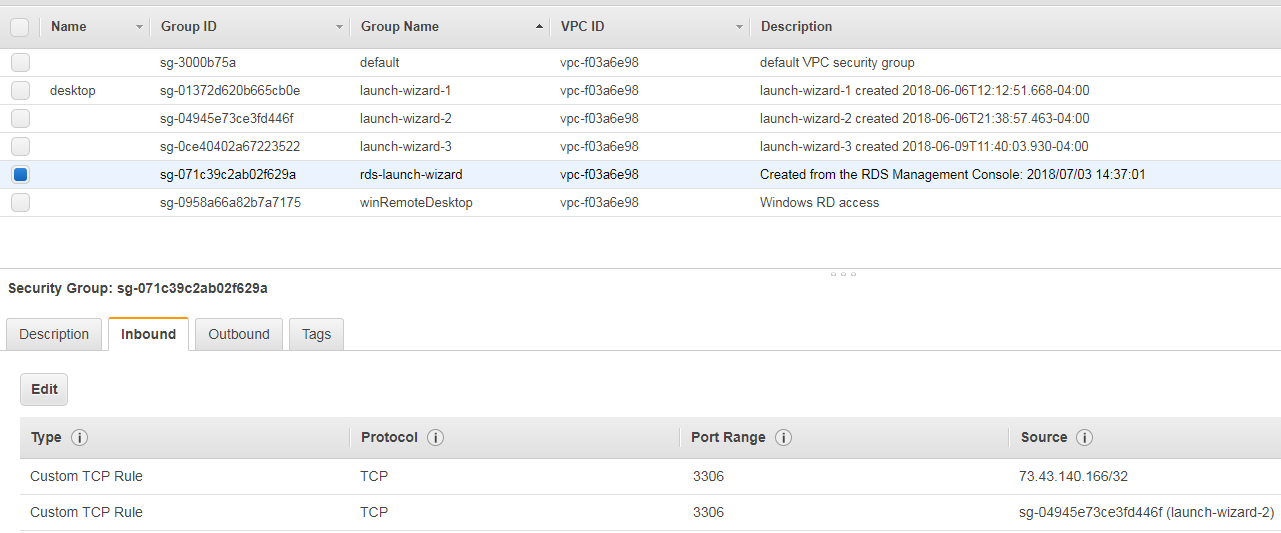
* Memcached
* Redis

EC is a web service that makes it easy to deploy, operate, and scale an in-memory cache in the cloud. The service improves the performance of web applications by allowing you to retrieve information from fast, managed, in-memory caches, instead of relying entirely on slower disk-based databases.

## Create MySQL DB

To access to MySQL from the webserver located on EC2 instance – security should be updated

In security groups, choose auto created rds security group, press edit, add new rule – choose mySql/Aurora type, for source choose security group configured for EC2 instance with web access.



Automated backups

* There are two different types of backups for AWS: Automated and Snapshots
* Automated backups allow you to recover your database to any point in time within a “retention period”. The retention period can be between one and 35 days. Automatic backups will take a full daily snapshot and will also store transaction logs throughout the day. When you do a recovery, AWS will first choose the most recent daily back up and then apply transaction logs relevant to that day. This allows you to do a point in time recovery down to a second, within the retention period.
* Automatic backups are enabled by default. The backup data is stored in S3 and you get free storage space equal to the size of your database. So, if you have an RDS instance of 10 gigabytes you will get 10Gb worth of storage.
* Backups are taken within a defined window. During the backup window storage I/O may be suspended while your data is being backed up and you might experience elevated latency.

## Snapshots

DB snapshots are done manually (i.e. they are user-initiated). They are stored even after you delete the original RDS instance, unlike automated backups

## Restoring backups

Whenever you restore either an Automatic backup or a manual Snapshot, the restored version of the database will be a new RDS instance with a new DNS endpoint.

## Encryption

Encryption at rest is supported for MySQL, Oracle, SQL Server, PostgreSQL, MariaDB and Aurora. Encryption is done using AWS Key Management Service (KMS). Once your RDS instance is encrypted, the data stored at rest in the underlying storage is encrypted, as are its automated backups, read replicas and snapshots.

At the present time, encrypting as existing database, you must first create a snapshot, make a copy of the snapshot and encrypt the copy.

## Multi-AZ

Multi AZ allows you to have an exact copy of your production database in another Availability Zone. AWS handles the replication for you, so when your production database is written to, this will automatically be synchronized to the stand by database.

In the event of planned database maintenance, DB instance failure, or an AZ failure, Amazon RDS will automatically failover to the standby so that database operations can resume quickly without administrative intervention.

## Read Replica

Read replicas allow you to have a read-only copy of your production database. This is achieved by using Asynchronous replication from the primary RDS instance to the read replica. You use read replicas primarily for very read-heavy database workloads.

* Used for scaling, not for DR
* Must have automatic backups turned on in order to deploy a read replica
* You can have up to 5 read replica copies of any database.
* You can have read replicas of read replicas (but watch out for latency)
* Each read replica will have its own DNS end point.
* You can have read replicas that have Multi-AZ.
* You can create read replicas of Multi-AZ source databases.
* Read replicas can be promoted to be their own databases. This breaks the replication.
* You can have a read replica in a second region.

## DynamoDB

Amazon DynamoDB is a fast and flexible NoSQL database service for all applications, that need consistent, single-digit millisecond latency at any scale. It is a fully managed database and supports both document and key-value data models. Its flexible data model and reliable performance make it a great fit for mobile, web, gaming, ad-tech, IoT and many other applications.

* Stored on SSD storage
* Spread across 3 geographically distinct data centers
* Eventual consistent reads (default)
  + Consistency across all copied of data is usually reached within a second. Repeating a read after a short time should return the updated data. (Best Read Performance)
* Strongly consistent reads
  + A strongly consistent read returns a result that reflects all writes that received a successful response prior to the read.

## Redshift

Amazon redshift is a fast and powerful, fully managed petabyte scale Data Warehouse service in the cloud. Customers can start small for just $0.25 per hour with no commitments or upfront costs and scale to a petabyte or more for $1,000 per terabyte per year, less than a tenth of most other data warehousing solutions

* Single node (160Gb)
* Multi-Node
  + Leader node(manages client connections and receives queries)
  + Compute node – store data and perform queries and computations. Up to 128 compute nodes.
* Currently available in 1 AZ

***Columnar Data Storage:***

Instead of storing data as a series of rows, Amazon Redshift organizes the data by column. Unlike row based systems which are ideal for transactions processing, column-based systems are ideal for data warehousing and analytics, where queries often involve aggregates performed over large data sets. Since only the columns involved in the queries are processed and columnar data is stored sequentially on the storage media, column-based systems require far fewer I/Os, greatly improving query performance.

* Better compression for data
* Less space required than RDS (no indexes, no mater. Views etc)
* Massively Parallel processing (MPP) – automatically distributes all data and queries across all nodes.

***Redshift security***

* Encrypted in transit using SSL
* Encrypted at rest using AES-256 encryption
* By default RedShift takes care of key management.
  + Manage your own keys through HSM
  + AWS Key Management Service

## Elasticache

EC is a web service that makes it easy to deploy, operate, and scale an in-memory cache in the cloud. The service improves the performance of web applications by allowing you to retrieve information from fast, managed, in-memory caches, instead of relying entirely on slower disk-based databases.

* Memcached
  + Widely adopted memory object caching system. ElastiCache is protocol compliant with Memcached environments will work seamlessly with the service.
* Redis
  + A popular open-source in-memory key-value store that supports data structures such as sorted sets and lists. ElastiCache supports Master/Slave replication and Multi-AZ which can be used to achieve cross AZ redundancy

## Aurora

Amazon Aurora is a MySQL-compatible, relational database engine that combines the speed and availability of high-end commercial databases with the simplicity and cost-effectiveness of open source databases. Amazon Aurora provides up to five times better performance than MySQL at a price point one tenth that of a commercial database while similar performance and availability.

* Scaling:
  + starts with 10Gb, increments by 10Gb to 64Gb (storage autoscaling).
  + Compute resources can scale up to 32vCPUs and 244Gb of memory.
  + 2 copies of your data in each AZ, with minimum of 3 AZs. Total 6 copies of the data.
  + Designed to transparently handle the loss of up to 2 copies of data without affecting database write availability and up to three copies without affecting read availability.
  + Auto healing – errors automatically detected and fixed.
* Aurora replicas
  + 2 types of replicas availables
  + Aurora replicas (up to 15)
  + MySQL Read replicas (up to 5)