# URLS:

## Amazon EBS Volume Types and Features:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html#EBSVolumeTypes_piops>

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumes.html#EBSFeatures>

<https://aws.amazon.com/ebs/details/>

## S3 Objects Naming convention for performance

<https://docs.aws.amazon.com/AmazonS3/latest/dev/request-rate-perf-considerations.html>

## Sharing objects with others using pre-signed urls

<https://docs.aws.amazon.com/AmazonS3/latest/dev/ShareObjectPreSignedURL.html>

## Cloud front signed urls / Origin access identity: (Cloud Front OAI)

<https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/private-content-restricting-access-to-s3.html>

## Multi part upload of S3 objects

<https://docs.aws.amazon.com/AmazonS3/latest/dev/mpuoverview.html>

## Reserved instances

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-reserved-instances.html>

## Modifying reserved instances

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ri-modifying.html>

## Health checks by ELB

<https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-healthchecks.html>

## ELB Cross zone balancing

<https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/enable-disable-crosszone-lb.html>

## DIFFERENCE BETWEEN ROLES AND USERS, IAM Roles and Roles vs Resource based policies:

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id.html>

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html>

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_compare-resource-policies.html>

## SAML and STS: Security token service:

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers_saml.html>

## Stateful vs stateless :

<https://www.quora.com/What-is-the-difference-between-security-groups-and-the-network-access-control-list-in-AWS>

## Bastion hosts:

<https://docs.aws.amazon.com/quickstart/latest/linux-bastion/architecture.html>

<https://aws.amazon.com/blogs/security/how-to-record-ssh-sessions-established-through-a-bastion-host/>

## Active Directory:

<https://en.wikipedia.org/wiki/Active_Directory>

## LDAP: Lightweight Directory Access Protocol

<https://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol>

## S3 Bucket Permissions:

<https://aws.amazon.com/articles/amazon-s3-bucket-public-access-considerations/>

## EC2 User data:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/user-data.html#user-data-shell-scripts>

## BASH Programming for shell scripting:

<http://tldp.org/HOWTO/Bash-Prog-Intro-HOWTO.html>

## MAC Address and NIC (Network Interface Card)

<https://whatismyipaddress.com/mac-address>

## Network Interface:

<https://docs.oracle.com/javase/tutorial/networking/nifs/definition.html>

<https://www.lifewire.com/definition-of-nic-817866>

## ENI: Elastic Network Interfaces and best practices:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-eni.html>

## S3 Reduced Redundancy Storage (RRS):

<https://aws.amazon.com/s3/reduced-redundancy/>

## S3 Data Encryption:

<https://docs.aws.amazon.com/AmazonS3/latest/dev/UsingEncryption.html>

## Bucket policies:

<https://docs.aws.amazon.com/AmazonS3/latest/dev/example-bucket-policies.html#example-bucket-policies-use-case-3>

## Security groups in EC2 VPC:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-network-security.html#security-group-connection-tracking>

## S3 Server Access Logging:

<https://docs.aws.amazon.com/AmazonS3/latest/dev/ServerLogs.html>

## Instance Types:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-types.html#instance-networking-storage>

## S3 Access Control

<https://docs.aws.amazon.com/AmazonS3/latest/dev/s3-access-control.html>

## EC2 Windows Device Naming:

<https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/device_naming.html>

## Instance Meta Data

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html>

## EC2 API Tools:

<https://docs.aws.amazon.com/cli/latest/reference/ec2/index.html>

## S3 IAM Policy:

<https://docs.aws.amazon.com/AmazonS3/latest/dev/example-policies-s3.html#iam-policy-ex0>

## Starting and stopping instances:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Stop_Start.html>

## S3 PUT Objects vs S3 POST Objects

<https://docs.aws.amazon.com/AmazonS3/latest/API/RESTObjectPUT.html>

<https://docs.aws.amazon.com/AmazonS3/latest/API/RESTObjectPOST.html>

## S3 Versioning

<https://docs.aws.amazon.com/AmazonS3/latest/dev/Versioning.html>

## EC2 Network and Security:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_Network_and_Security.html>

## Enhanced networking in Linux:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/enhanced-networking.html#supported_instances>

## AMI Virtualization (HVM vs PVM)

<https://cloudacademy.com/blog/aws-ami-hvm-vs-pv-paravirtual-amazon/>

## XEN HYPERVISOR:

<http://www-archive.xenproject.org/products/xenhyp.html>

## Attaching roles to instances: (2nd blog important)

<https://aws.amazon.com/about-aws/whats-new/2017/02/new-attach-an-iam-role-to-your-existing-amazon-ec2-instance/>

[**https://aws.amazon.com/blogs/security/new-attach-an-aws-iam-role-to-an-existing-amazon-ec2-instance-by-using-the-aws-cli/?sc\_channel=sm&sc\_campaign=rolesforrunninginstances&sc\_publisher=tw&sc\_medium=social&sc\_content=read-post&sc\_country=global&sc\_geo=global&sc\_category=ec2&sc\_outcome=launch**](https://aws.amazon.com/blogs/security/new-attach-an-aws-iam-role-to-an-existing-amazon-ec2-instance-by-using-the-aws-cli/?sc_channel=sm&sc_campaign=rolesforrunninginstances&sc_publisher=tw&sc_medium=social&sc_content=read-post&sc_country=global&sc_geo=global&sc_category=ec2&sc_outcome=launch)

## IAM Best practices:

<https://aws.amazon.com/blogs/security/adhere-to-iam-best-practices-in-2016/>

## Object Lifecycle Management:

<https://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html>

## Difference between dedicated hosts and dedicated instances:

<https://aws.amazon.com/ec2/dedicated-hosts/>

<https://aws.amazon.com/ec2/purchasing-options/dedicated-instances/>

## Instance Console Output:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-console.html#instance-console-console-output>

## EBS Encryption:

<https://docs.aws.amazon.com/kms/latest/developerguide/services-ebs.html#ebs-encrypt>

## EC2 Trouble Shooting:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/TroubleshootingInstancesConnecting.html#TroubleshootingInstancesConnectionTimeout>

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/TroubleshootingInstances.html#InitialSteps>

## Sharing EBS Snapshot:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-modifying-snapshot-permissions.html>

## Preserving EBS Volumes on Instance Termination / Instance Termination:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/terminating-instances.html#preserving-volumes-on-termination>

## EC2 Monitoring using EC2 metrics vs CloudWatch:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring_ec2.html>

## CloudWatch Monitoring Scripts for EC2: (PERL Scripts)

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/mon-scripts.html>

## Sending logs, events to Amazon CloudWatch for Windows using cloud watch agent:

<https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/send_logs_to_cwl.html>

## Block Device Mapping:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/block-device-mapping-concepts.html>

## Instance Store Volumes:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/InstanceStorage.html>

## Monitoring EBS volumes with Status Checks:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/TroubleshootingInstances.html#InitialSteps>

## Public Datasets:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-public-data-sets.html>

## EBS PIOPS:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html#EBSVolumeTypes_piops>

## Tag Restrictions:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Using_Tags.html#tag-restrictions>

## Troubleshooting Instance Capacity:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-capacity.html>

## Spot Instances:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/how-spot-instances-work.html>

## EBS Deleting Snapshots, How incremental snapshots work:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-deleting-snapshot.html>

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSSnapshots.html#how_snapshots_work>

## De-registering Linux AMI:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/deregister-ami.html>

## Device Naming:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/device_naming.html>

## Detaching EBS Volume from an Instance:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-detaching-volume.html>

## EBS Volume Pricing:

<https://aws.amazon.com/ebs/pricing/>

## EBS Creating Snapshot:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-creating-snapshot.html>

## Understanding how IAM Works:

<https://docs.aws.amazon.com/IAM/latest/UserGuide/intro-structure.html>

## IAM Permission Types:

<https://docs.aws.amazon.com/IAM/latest/UserGuide/access_controlling.html#TypesPermissions>

## AD Connector:

<https://aws.amazon.com/blogs/security/how-to-connect-your-on-premises-active-directory-to-aws-using-ad-connector/>

## Cloud Trail:

<https://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-concepts.html#cloudtrail-concepts-global-service-events>

## Send message to SQS using SNS:

<https://docs.aws.amazon.com/sns/latest/dg/SendMessageToSQS.html>

## AWS Direct Connect:

<https://docs.aws.amazon.com/directconnect/latest/UserGuide/Welcome.html>

## AWS SQS, Short and Long Polling:

<https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-long-polling.html>

<https://docs.aws.amazon.com/AWSSimpleQueueService/latest/APIReference/API_ReceiveMessage.html>

## AWS SQS Standard and FIFO Queue Types:

<https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/standard-queues.html>

<https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/FIFO-queues.html>

## SQS Throughput:

<https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-throughput-horizontal-scaling-and-batching.html>

## Instance Profile in Cloud Formation:

<https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-resource-iam-instanceprofile.html>

## Web Identity Federation:

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers_oidc.html>

## Direct Connect IAM:

https://docs.aws.amazon.com/directconnect/latest/UserGuide/using\_iam.html

## Amazon Kinesis, Kinesis Data Streams:

<https://aws.amazon.com/kinesis/>

<https://docs.aws.amazon.com/streams/latest/dev/kinesis-kpl-concepts.html#w2ab1c12b7b7c19c11>

## AWS Appstream 2.0:

<https://docs.aws.amazon.com/appstream2/latest/developerguide/what-is-appstream.html>

## AWS Elastic search:

<https://docs.aws.amazon.com/elasticsearch-service/latest/developerguide/what-is-amazon-elasticsearch-service.html>

## Elastic cache and types:

<https://aws.amazon.com/elasticache/>

<https://aws.amazon.com/redis/>

<https://aws.amazon.com/memcached/>

## Storage Gateway:

<https://docs.aws.amazon.com/storagegateway/latest/userguide/WhatIsStorageGateway.html>

<https://docs.aws.amazon.com/storagegateway/latest/userguide/StorageGatewayConcepts.html>

## SWF Domain:

<https://docs.aws.amazon.com/amazonswf/latest/developerguide/swf-dev-domains.html>

## ELB Listener configuration:

<https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-listener-config.html>

## Best Practices for Handling Time-Series Data in DynamoDB:

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/bp-time-series.html>

## Scope of AWS Resources (AZ/Regional/Global)

<http://jayendrapatil.com/aws-global-vs-regional-vs-az-resources/>

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/resources.html>

## Regions and Availability zones:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html>

## Dynamo DB Errors:

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Programming.Errors.html>

## Route 53 Overview:

<https://aws.amazon.com/route53/>

## DNS Resource Record types of Route 53:

<https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/ResourceRecordTypes.html>

## Alias vs CNAME Records:

<https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/resource-record-sets-choosing-alias-non-alias.html>

## Name Server (NS) and Start of Authority (SOA) records:

<https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/SOA-NSrecords.html>

## Promoting Read Replicas to Standby DB Instances:

<https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_ReadRepl.html#USER_ReadRepl.Promote>

## RDS Instance Storage:

<https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_Storage.html>

## Benefits of Auto Scaling:

<https://docs.aws.amazon.com/autoscaling/ec2/userguide/auto-scaling-benefits.html#arch-AutoScalingMultiAZ>

## Read Replicas:

<https://aws.amazon.com/blogs/aws/amazon-rds-announcing-read-replicas/>

## RDS Troubleshooting Tips:

<https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/APITroubleshooting.html>

## RDS SQL Server SSL Support:

<https://aws.amazon.com/blogs/aws/amazon-rds-sql-server-ssl-support/>

## DB Instance Status:

<https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Overview.DBInstance.Status.html>

## Route 53 Routing Policies:

<https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html>

## Rebooting a DB instance:

<https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_RebootInstance.html>

## ELB Sticky Sessions:

<https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-sticky-sessions.html>

## NAT:

<https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-nat.html>

## Comparison of NAT Instances vs NAT Gateways:

<https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-nat-comparison.html>

## Auto Scaling Launch configuration:

<https://docs.aws.amazon.com/autoscaling/ec2/userguide/LaunchConfiguration.html>

## VPC IP Addresses:

<https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-ip-addressing.html#vpc-private-ipv4-addresses>

## Dead Letter Queues in Elastic Beanstalk:

<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features-managing-env-tiers.html#worker-deadletter>

## VPC Sub Nets:

<https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Subnets.html>

## VPC Route Tables:

<https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Route_Tables.html>

## EC2 Burstable Instances:

<https://aws.amazon.com/blogs/aws/low-cost-burstable-ec2-instances/>

## ELB Access Logs:

<https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/access-log-collection.html>

## VPC NACLS:

<https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_ACLs.html>

## VPC VPN:

<https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_VPN.html#Examples>

## VPC VPN Connection Types:

<https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpn-connections.html>

## AWS Direct Connect:

https://docs.aws.amazon.com/directconnect/latest/UserGuide/Welcome.html

## S3 CORS (Cross Origin Resource Sharing):

<https://docs.aws.amazon.com/AmazonS3/latest/dev/cors.html>

## Auto Scaling Default Termination Policy:

<https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-termination.html>

## EBS Optimized Instance:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSOptimized.html>

## Job Observer Pattern: (EC2, Auto scaling and SQS Queue)

<http://en.clouddesignpattern.org/index.php/CDP:Job_Observer_Pattern>

## VPC Security, Comparison of Security Groups and Network ACLs:

<https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Security.html>

## Subnet CIDR blocks Calculator:

<http://www.subnet-calculator.com/cidr.php>

## Cloud Front Origin Settings:

<https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/distribution-web-values-specify.html#DownloadDistValuesOrigin>

## VPC Egress Traffic control, Web Proxy Server:

<https://d0.awsstatic.com/aws-answers/Controlling_VPC_Egress_Traffic.pdf>

## EC2 Cloud Watch Monitoring:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-cloudwatch-new.html>

## VPC Security Groups:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-network-security.html#vpc-security-groups>

## RAID Configuration:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/raid-config.html>

<https://en.wikipedia.org/wiki/RAID>

## Disaster Recovery:

<https://media.amazonwebservices.com/AWS_Disaster_Recovery.pdf>

<http://jayendrapatil.com/aws-disaster-recovery-whitepaper/>

## S3 Access Control:

<https://docs.aws.amazon.com/AmazonS3/latest/dev/how-s3-evaluates-access-control.html>

## Storage Options:

<https://d0.awsstatic.com/whitepapers/Storage/aws-storage-options.pdf>

## EC2 Key Pairs:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html>

## Multiple IP Addresses:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/MultipleIP.html>

## SQS Pricing, Standard Queues vs FIFO:

<https://aws.amazon.com/sqs/pricing/>

## Protecting Data Using Server-Side Encryption with Customer-Provided Encryption Keys (SSE-C):

<https://docs.aws.amazon.com/AmazonS3/latest/dev/ServerSideEncryptionCustomerKeys.html>

## Restoring a DB Instance to a Specified Time:

<https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_PIT.html>

## Access Keys Best Practices:

<https://docs.aws.amazon.com/general/latest/gr/aws-access-keys-best-practices.html>

## Instance Purchasing options:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-purchasing-options.html>

## Instance Status Checks:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring-system-instance-status-check.html>

## ELB Backed Instances:

<https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-backend-instances.html>

## AWS Directory Services:

<http://jayendrapatil.com/aws-directory-services/>

## AWS Import/Export (alias) SnowBall:

<https://aws.amazon.com/snowball/>

## SSD Instance Stores:

<https://aws.amazon.com/blogs/aws/new-high-io-ec2-instance-type-hi14xlarge/>

## Dynamo DB Optimistic Locking:

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/DynamoDBMapper.OptimisticLocking.html>

## Dynamo DB Throughput Capacity:

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/HowItWorks.ProvisionedThroughput.html>

## Spot vs On-demand Instances:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-spot-instances.html>

## Dynamo DB Primary Key, Sort Key, Global Secondary Index:

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GSI.html>

## AWS EMR Best Practices:

<https://d0.awsstatic.com/whitepapers/aws-amazon-emr-best-practices.pdf>

## Dynamo DB Partition Key Design Considerations:

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/bp-partition-key-uniform-load.html>

## Tips for Securing EC2 Instances:

<https://aws.amazon.com/articles/tips-for-securing-your-ec2-instance/>

## Dynamo DB SCAN Considerations:

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/bp-query-scan.html>

## Cloud Front Expiration :

<https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/Expiration.html>

## Sharing EBS Snapshot:

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-modifying-snapshot-permissions.html>

# NOTES:

## Scope of ELB, AMI, EBS and Snapshots:

* ELB AND AMI **cannot span across regions in applications but can span across AZ**. ELB doesn’t have a fixed IP Address.
* When you create an EBS volume in an Availability Zone, it is automatically replicated within that zone to prevent data loss due to failure of any single hardware component. After you create a volume, you can attach it to any EC2 instance in the same Availability Zone**. An EBS volume and the instance to which it attaches must be in the same Availability Zone.**
* Amazon EBS provides the ability to **create snapshots (backups) of any EBS volume** and write a copy of the data in the volume to Amazon S3, **where it is stored redundantly in multiple Availability Zones.** **These snapshots can be used to create multiple new EBS volumes or move volumes across Availability Zones.**

## S3 Object size:

0 bytes to 5 TB

## Ports:

* port 80 http
* port 443 https
* port 22 ssh
* port 8080:
* If you have an own web server running, then it runs on port 80. Port 8080 is a place to host a secondary or alternate web server.
* It is commonly used for proxy and caching. Example: Apache Tomcat, M2MLogger and a Web GUI.
* ICMP for Pings

## Important blogs:

https://aws.amazon.com/blogs/security/the-most-viewed-aws-security-blog-posts-in-2016/

## AD and LDAP:

Active Directory is a database based system that provides authentication, directory, policy, and other services in a Windows environment

LDAP (Lightweight Directory Access Protocol) is an application protocol for querying and modifying items in directory service providers like Active Directory, which supports a form of LDAP.

Short answer: AD is a directory services database, and LDAP is one of the protocols you can use to talk to it. A common use of LDAP is to provide a central place to store usernames and passwords. This allows many different applications and services to connect to the LDAP server to validate users.

## Instance profiles for roles:

If you use the IAM console, the instance profile is managed for you and is mostly transparent to you. However, if you use the AWS CLI or API to create and manage the role and EC2 instance, then you must create the instance profile and assign the role to it as separate steps. Then, when you launch the instance, you must specify the instance profile name instead of the role name.

## Bastion host:

A bastion host is a server whose purpose is to provide access to a private network from an external network, such asthe Internet. Because of its exposure to potential attack, a bastion host must minimize the chances of penetration. For example, you can use a bastion host to mitigate the risk of allowing SSH connections from an external network to the Linux instances launched in a private subnet of your Amazon Virtual Private Cloud (VPC).

## Stateful Security Groups vs Stateless NACLS:

* Security groups are stateful, if you add an inbound rule say for port 80, it is automatically allowed out, and meaning outbound rule for that particular port need not be explicitly added. Instance to Instance communication is controlled using **security groups configured at the instance level**
* NACLs are stateless. You need to provide explicit inbound and outbound rules. Instance to Instance communication is controlled using **NACLS configured at subnet level and associated with a subnet (NOT Associated with an AZ).**Subnet associated with an AZ
* Security groups control inbound and outbound traffic for your instances, and network ACLs control inbound and outbound traffic for your subnets.
* **Network ACLs are stateless; responses to allowed inbound traffic are subject to the rules for outbound traffic (and vice versa).**
* Your VPC automatically comes with a modifiable default network ACL. By default, it allows all inbound and outbound IPv4 traffic and, if applicable, IPv6 traffic.
* You can create a custom network ACL and associate it with a subnet. By default, each custom network ACL denies all inbound and outbound traffic until you add rules.

|  |  |
| --- | --- |
| **Security Group** | **Network ACL** |
| Operates at the instance level | Operates at the subnet level |
| Supports allow rules only | Supports allow rules and deny rules |
| Is stateful: Return traffic is automatically allowed, regardless of any rules | Is stateless: Return traffic must be explicitly allowed by rules |
| We evaluate all rules before deciding whether to allow traffic | We process rules in number order when deciding whether to allow traffic |
| Applies to an instance only if someone specifies the security group when launching the instance, or associates the security group with the instance later on | Automatically applies to all instances in the subnets it's associated with (therefore, you don't have to rely on users to specify the security group) |

## Elastic IP Address:

An Elastic IP address is a static IPv4 address designed for dynamic cloud computing. An Elastic IP address is associated with your AWS account. With an Elastic IP address, you can mask the failure of an instance or software by rapidly remapping the address to another instance in your account.An Elastic IP address is a public IPv4 address, which is reachable from the internet. If your instance does not have a public IPv4 address, you can associate an Elastic IP address with your instance to enable communication with the internet; for example, to connect to your instance from your local computer.

## S3 Bucket access:

Objects are made publicly readable - Object ACL / Bucket policy. Bucket ACLS are for individual AWS accounts. Not recommended to give public access to bucket using Bucket ACL

IAM is not for bucket access permissions but for user/resource control access

## Linux vs Windows :

-> Passing custom script to instances in Autoscaling group : if windows, use EC2config; if Linux, use user data section

## Types of ENI (Elastic Network Interface):

Hot: Attach ENI to an instance which is running

Warm: Attach ENI to an instance which is stopped

Cold: Attach ENI to an instance during launch

## S3 cost considerations:

S3 Cost depends on number of storage requests and Data transfer out rate. Beyond a limit, AWS Support required since total number of requests is to be taken into account. Size of objects doesn’t matter, but number of requests per second /objects is important for performance

## S3 Data encryption (Rest vs transit):

You can protect data in transit by using SSL or by using client-side encryption.

You have the following options of protecting data at rest in Amazon S3:

* Use Server-Side Encryption – You request Amazon S3 to encrypt your object before saving it on disks in its data centers and decrypt it when you download the objects.
* Use Client-Side Encryption – You can encrypt data client-side and upload the encrypted data to Amazon S3. In this case, you manage the encryption process, the encryption keys, and related tools.

## Restricting public objects in S3 bucket from IP Address range:

Use bucket policies. NACLS are used only in VPC while S3 here is public

## Default ENI Termination when EC2 terminates:

ENI automatically created by console – Terminates

ENI created using CLI – Doesn’t Terminate

## HVM vs PVM

Hardware Virtual Machine for enhanced networking as well as performance than Para Virtual Machine. The virtualization type of your instance is determined by the AMI that you use to launch it. Current generation instance types support hardware virtual machine (HVM) only. Some previous generation instance types support paravirtual (PV) and some AWS regions support PV instances. For more information, see Linux AMI Virtualization Types.For best performance, we recommend that you use an HVM AMI. In addition, HVM AMIs are required to take advantage of enhanced networking. HVM virtualization uses hardware-assist technology provided by the AWS platform. With HVM virtualization, the guest VM runs as if it were on a native hardware platform, except that it still uses PV network and storage drivers for improved performance.

## Secondary Indexes:

**They exist on DYNAMODB but not on S3**

## EBS Backed only Instances:

General purpose T2 and Compute Optimized C4

## IAM for EC2:

IAM – Has access to launch/start/stop/terminate/other controls to EC2. But AFTER launch, it cant control access. Now access can be controlled by SSH Keys(Linux), Passwords(Windows) and Security groups (IP, Traffic)

## Placement Groups:

A placement group is a logical grouping of instances within a single Availability zone. Placement group can span across peered VPCs.

Recommended for low latency, high network throughput or both. Two types :

* Cluster—clusters instances into a low-latency group in a single Availability Zone
* Spread—spreads instances across underlying hardware

Spread Placement groups can span multiple Availability zones.

## Hypervisor:

A hypervisor or virtual machine monitor (VMM) is computer software, firmware or hardware that creates and runs virtual machines. A computer on which a hypervisor runs one or more virtual machines is called a host machine, and each virtual machine is called a guest machine.

## Ports for SSH in LINUX and RDP in WINDOWS:

Check your security group rules. You need a security group rule that allows inbound traffic from your public IPv4 address on the proper port. For Linux instances: Verify that there is a rule that allows traffic from your computer to port 22 (SSH).For Windows instances: Verify that there is a rule that allows traffic from your computer to port 3389 (RDP).

## Preventing accidental termination of Instances:

By default, you can terminate your instance using the Amazon EC2 console, command line interface, or API. If you want to prevent your instance from being accidentally terminated using Amazon EC2, you can enable termination protection for the instance. The DisableApiTermination attribute controls whether the instance can be terminated using the console, CLI, or API. By default, termination protection is disabled for your instance. You can set the value of this attribute when you launch the instance, while the instance is running, or while the instance is stopped (for Amazon EBS-backed instances).

The DisableApiTermination attribute does not prevent you from terminating an instance by initiating shutdown from the instance (using an operating system command for system shutdown) when the InstanceInitiatedShutdownBehavior attribute is set.

## What to Do If an Instance Immediately Terminates?

After you launch an instance, we recommend that you check its status to confirm that it goes from the pending state to the running state, not the terminated state. The following are a few reasons why an instance might immediately terminate:

* You've reached your EBS volume limit. For information about the volume limit, see Instance Volume Limits. To submit a request to increase your Amazon EBS volume limit, complete the AWS Support Center Create Case form. For more information, see Amazon EC2 Service Limits.
* An EBS snapshot is corrupt.
* The instance store-backed AMI you used to launch the instance is missing a required part (an image.part.xx file).

## How to copy data from 1 EBS volume (unencrypted) to a new encrypted EBS volume?

* Create a new encrypted volume. Copy data from old EBS volume to new using FILE MANIPULATION TOOLS like RSYNC(LINUX) and ROBOCOPY (WINDOWS) etc.
* (or) Create an encrypted snapshot of the volume, restore the encrypted snapshot to encrypted volume and mount EBS volume

## Instance Store Volumes vs EBS Volumes:

A block device is a storage device that moves data in sequences of bytes or bits (blocks). These devices support random access and generally use buffered I/O. Examples include hard disks, CD-ROM drives, and flash drives. A block device can be physically attached to a computer or accessed remotely as if it were physically attached to the computer. Amazon EC2 supports two types of block devices:

* Instance store volumes (virtual devices whose underlying hardware is physically attached to the host computer for the instance).
* EBS volumes (remote storage devices)

## Public dataset formats:

2 namely EBS Snapshots and S3 Buckets

## EBS Volume types and ratio, snapshots occurrence:

* The three types that are now available include Magnetic, Provisioned IOPS (SSD) and General Purpose (SSD) EBS volumes.
* Ratio is 50:1
* Snapshots occurrence: Asynchronous, Incremental and point in time.
* Snapshot status is Pending(To transfer blocks from EBS to S3) -> EBS Volume can be used/available for read/writes, can be attached/detached

## Tag keys and values:

Tag keys and values are case-sensitive.

Don't use the aws: prefix for either keys or values; it's reserved for AWS use. You can't edit or delete tag keys or values with this prefix. Tags with this prefix do not count against your tags per resource limit. **Specific conditions can be added to IAM policy which allows access to specific tags**

## How Devices Are Made Available in the Operating System

Device names like /dev/sdh and xvdh are used by Amazon EC2 to describe block devices. The block device mapping is used by Amazon EC2 to specify the block devices to attach to an EC2 instance. After a **block device is attached to an instance**, it must be **mounted by the operating system** before you can access the storage device. **When a block device is detached from an instance, it is unmounted by the operating system and you can no longer access the storage device**.

## Root Device Name:

/dev/sda1 (or) /dev/xvda

## AWS Direct Connect:

* Direct Connect allows us to access all AZ’s within a region. It does not involve the internet. Instead it uses a dedicated, private network between the intranet and VPC. Provides **greater bandwidth** than internet based VPN.
* Direct Connections are not redundant. So as to increase fault tolerance, set up a second direct connection, private virtual interface in the same region (or) HARDWARE VPN over the internet.
* Each connection consists of a single dedicated connection between ports on your router and an Amazon router. We recommend establishing a second connection if redundancy is required. When you request multiple ports at the same AWS Direct Connect location, they will be provisioned on redundant Amazon routers. To achieve high availability, we recommend you to have connections at multiple AWS Direct Connect locations. You can refer to this page to learn more about achieving highly available network connectivity.
* IAM Key names are case insensitive for direct connect. aws:CurrentTime is same as AWS:currenttime.
* In an IAM policy, you can specify any or all actions that AWS Direct Connect offers. The action name must include the lowercase prefix directconnect:. For example: directconnect:DescribeConnections, directconnect:CreateConnection, or directconnect:\*

## Cloud HSM (Hardware Security Module):

The AWS CloudHSM service helps you meet corporate, contractual and regulatory compliance requirements for data security by using dedicated Hardware Security Module (HSM) instances within the AWS cloud. AWS and AWS Marketplace partners offer a variety of solutions for protecting sensitive data within the AWS platform, but for some applications and data subject to contractual or regulatory mandates for managing cryptographic keys, additional protection may be necessary. CloudHSM complements existing data protection solutions and allows you to protect your encryption keys within HSMs that are designed and validated to government standards for secure key management. CloudHSM allows you to securely generate, store and manage cryptographic keys used for data encryption in a way that keys are accessible only by you.

Place CloudHSM instances near EC2 Instances decreases network latency, which can improve application performace.

## End Point:

An endpoint is a URL that is the entry point for web service requests. For example, https://cloudtrail.us-west-2.amazonaws.com is the US West (Oregon) regional entry point for the AWS CloudTrail service. Regional endpoints help reduce latency in your applications.

SNS can send notifications to end points such as Email JSON,HTTP, SQS but can’t send to SES

A VPC endpoint enables you to create a private connection between your VPC and another AWS service. When you create an endpoint, you specify the route tables in your VPC that are used by the endpoint. A route is automatically added to each of the route tables with a destination that specifies the prefix list ID of the service (pl-xxxxxxxx), and a target with the endpoint ID (vpce-xxxxxxxx). You cannot explicitly delete or modify the endpoint route, but you can change the route tables that are used by the endpoint.

## STS, IDP:

1) User accesses Identity broker. Identity broker authenticates user from Corporate Directory Store/LDAP Identity store/AD (Enterprise side).

2) Identity Broker (Enterprise) Gets temporary credentials from STS (Security Token Service in AWS). User redirected to AWS Management console and Access AWS APIs (AWS Side)

## SAML, IDP:

1) User accesses Identity broker. Identity broker authenticates user from Corporate Directory Store/LDAP Identity store/AD. After authentication, Identity provider returns SAML assertion to the user/browser interface. (Enterprise side - LDAP needs to be SAML compliant)

2) Client posts the SAML assertion to sign- in-URL (AWS SSO End point) connected to STS. The AWS SSO end point validates and sends redirect to user/browser interface (AWS side)

3) Client directed to AWS Management console

## Operating system privileges on a relational database server:

RDS is a managed service from AWS, so it does not provide operating system privileges. So design EC2 Instances with RDS installed and keeps it in replication configuration for high availability in 2 different availability zones.

## Sharing snapshots:

Volumes can’t be shared. Encrypted snapshots can be shared with specific AWS accounts but can’t be made public. Snapshots with AWS Marketplace product codes can’t be made public too.

## AWS Trusted Advisor:

Online resource to help you reduce cost, increase performance, improves security and fault tolerance.

## Minimal Time for consistent backups (RAID vs NON-RAID):

Application consistent backups for RAID:

* Stop applications from writing to RAID Array. Flush all caches to disk
* Ensure associated EC2 Instances are not writing to RAID Array by freezing the file system, unmounting the RAID Array, shutting down associated EC2 Instances. After steps to halt I/O is completed, take snapshot of each EBS Volume.
* Wait for snapshots to complete since RAID and Resume the disk I/O for **consistent** backups

Application consistent backups for NON-RAID:

* Stop applications from writing to RAID Array. Flush all caches to disk
* Ensure associated EC2 Instances are not writing to RAID Array by freezing the file system, unmounting the RAID Array, shutting down associated EC2 Instances. After steps to halt I/O is completed, take snapshot of each EBS Volume.
* **Need not wait for snapshots to complete** since NON-RAID (Because time taken will be less and snapshots happen immediately), so Resume the disk I/O for **consistent** backups without waiting for snapshot completion.

## VPC Peering:

* A VPC peering connection is a networking connection between two VPCs that allows you to route traffic between them using private IPv4 addresses. Instances in either VPC can communicate with each other as if they are part of the same network. however, you cannot create a VPC peering connection between VPCs that have overlapping CIDR blocks.
* If 2 VPCs are peered, and there is a direct connection from 1 VPC to an on-premises center, if a second connection needs to be established to the same VPC, it needs to be done to that VPC only as VPC Peering cannot connect from 1 VPC to other.
* To enable the routing of traffic between VPCs in a VPC peering connection, you must add a route to one or more of your VPC route tables that points to the VPC peering connection to access all or part of the CIDR block of the other VPC in the peering connection. Similarly, the owner of the other VPC must add a route to their VPC route table to route traffic back to your VPC.
* For example, you have a VPC peering connection (pcx-1a2b1a2b) between two VPCs, with the following information:
* VPC A: vpc-1111aaaa, CIDR block is 10.0.0.0/16
* VPC B: vpc-2222bbbb, CIDR block is 172.31.0.0/16
* To enable traffic between the VPCs and allow access to the entire IPv4 CIDR block of either VPC, the VPC A route table is configured as follows.

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 10.0.0.0/16 | Local |
| 172.31.0.0/16 | pcx-1a2b1a2b |

* The VPC B route table is configured as follows.

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 172.31.0.0/16 | Local |
| 10.0.0.0/16 | pcx-1a2b1a2b |

* You can optionally set up a connection between your VPC and your corporate or home network. If you have an IPv4 address prefix in your VPC that overlaps with one of your networks' prefixes, any traffic to the network's prefix is dropped. For example, let's say that you have the following:
* A VPC with CIDR block 10.0.0.0/16
* A subnet in that VPC with CIDR block 10.0.1.0/24
* Instances running in that subnet with IP addresses 10.0.1.4 and 10.0.1.5
* On-premises host networks using CIDR blocks 10.0.37.0/24 and 10.1.38.0/24
* When those instances in the VPC try to talk to hosts in the 10.0.37.0/24 address space, the traffic is dropped because 10.0.37.0/24 is part of the larger prefix assigned to the VPC (10.0.0.0/16). The instances can talk to hosts in the 10.1.38.0/24 space because that block isn't part of 10.0.0.0/16.

## Amazon Cognito:

It acts as an identity broker and does much of the federation work for you. If you don't use Amazon Cognito, then you must write code that interacts with a web IdP (Login with Amazon, Facebook, Google, or any other OIDC-compatible IdP) and then calls the AssumeRoleWithWebIdentity API to trade the authentication token you get from those IdPs for AWS temporary security credentials.

## Four levels of Support in AWS:

Basic, Developer, Business, Enterprise

## SNS Topic:

A topic is a communication channel to send messages and subscribe to notifications. It provides an access point for publishers and subscribers to communicate with each other.

## ARN:

* Amazon Resource name is created for a topic.
* arn:aws:sns:us-west-2:111122223333:MyTopic

## Full admin privileges of underlying EC2 Instances provided in:

* EMR
* Elastic Beanstalk

## Multi AZ Failover for RDS Instances:

Failover is automatically handled by Amazon RDS so that you can resume database operations as quickly as possible without administrative intervention. **When failing over, Amazon RDS simply flips the canonical name record (CNAME) for your DB instance to point at the standby, which is in turn promoted to become the new primary.**

Failovers, as defined by the interval between the detection of the failure on the primary and the resumption of transactions on the standby, typically complete within one to two minutes. Failover time can also be affected by whether large uncommitted transactions must be recovered; **the use of adequately large instance types is recommended with Multi-AZ for best results. AWS also recommends the use of Provisioned IOPS with Multi-AZ instances, for fast, predictable, and consistent throughput performance.**

## Multi AZ Deployment for RDS Instances:

When you create or modify your DB instance to run as a Multi-AZ deployment, **Amazon RDS automatically provisions and maintains a synchronous “standby” replica in a different Availability Zone**. **Updates to your DB Instance are synchronously replicated across Availability Zones to the standby in order to keep both in sync and protect your latest database updates against DB instance failure. During certain types of planned maintenance, or in the unlikely event of DB instance failure or Availability Zone failure, Amazon RDS will automatically failover to the standby so that you can resume database writes and reads as soon as the standby is promoted**. **Since the name record for your DB instance remains the same, your application can resume database operation without the need for manual administrative intervention.** With Multi-AZ deployments, replication is transparent: you do not interact directly with the standby, and it cannot be used to serve read traffic.

## Events causing Amazon RDS to initiate a failover to the standby replica:

Amazon RDS detects and automatically recovers from the most common failure scenarios for Multi-AZ deployments so that you can resume database operations as quickly as possible without administrative intervention. Amazon RDS automatically performs a failover in the event of any of the following:

* Loss of availability in primary Availability Zone
* Loss of network connectivity to primary
* Compute unit failure on primary
* Storage failure on primary

Note: When operations such as DB instance scaling or system upgrades like OS patching are initiated for Multi-AZ deployments, for enhanced availability, they are applied first on the standby prior to an automatic failover. As a result, your availability impact is limited only to the time required for automatic failover to complete. **Note that Amazon RDS Multi-AZ deployments do not failover automatically in response to database operations such as long running queries, deadlocks or database corruption errors.**

## Creation of a read replica of another read replica in RDS:

**Amazon Aurora, Amazon RDS for MySQL and MariaDB: You can create a second-tier Read Replica from an existing first-tier Read Replica.** By creating a second-tier Read Replica, you may be able to move some of the replication load from the master database instance to a first-tier Read Replica. Please note that a second-tier Read Replica may lag further behind the master because of additional replication latency introduced as transactions are replicated from the master to the first tier replica and then to the second-tier replica.

**Amazon RDS for PostgreSQL: Read Replicas of Read Replicas are not currently supported.**

## Storage engines for use with RDS for MYSQL Read Replicas:

* **Amazon RDS for MySQL Read Replicas require a transactional storage engine and are only supported for the InnoDB storage engine.**
* **Non-transactional MySQL storage engines such as MyISAM might prevent Read Replicas from working as intended.** However, if you still choose to use MyISAM with Read Replicas, we advise you to watch the Amazon CloudWatch “Replica Lag” metric (available via the AWS Management Console or Amazon CloudWatch APIs) carefully and recreate the Read Replica should it fall behind due to replication errors. The same considerations apply to the use of temporary tables and any other non-transactional engines.

## Data Transfer Rate for replicating data between primary and standby in RDS:

You are **not charged** for the data transfer incurred in replicating data between your primary and standby. Internet data transfer in and out of your DB instance is charged the same as with a standard deployment.

## Read Replica stuck after Multi-AZ failover(Unable to get/apply updates) :

You may find in some cases that your Amazon RDS for MySQL Read Replica(s) aren’t able to receive or apply updates from their source Multi-AZ DB instance after a Multi-AZ failover. **This may be due to MySQL binlog events not being flushed to disk at the time of the failover.** Resolve the issue by **deleting the read replica** and creating a new one replacing it.

## Amazon Kinesis Types:

* Streams – To collect and process large streams of data in real time (Video streams/data streams)
* Firehose – To deliver real time streaming data to destinations like S3,Redshift
* Analytics-To process and analyze streaming data with standard SQL

Note:

**API CALL EVENTS are STREAMS**. Streams act as **buffer and transport mechanisms** for in-order programmatic events, hence useful for replicating API calls across systems.

## AWS Data Pipeline:

* AWS Data Pipeline is a web service that helps you reliably process and move data between different AWS compute and storage services, as well as on-premises data sources, at specified intervals. With AWS Data Pipeline, you can regularly access your data where it’s stored, transform and process it at scale, and efficiently transfer the results to AWS services such as Amazon S3, Amazon RDS, Amazon DynamoDB, and Amazon EMR.
* AWS Data Pipeline helps you easily create complex data processing workloads that are fault tolerant, repeatable, and highly available. You don’t have to worry about ensuring resource availability, managing inter-task dependencies, retrying transient failures or timeouts in individual tasks, or creating a failure notification system. AWS Data Pipeline also allows you to move and process data that was previously locked up in on-premises data silos.

## Amazon Appstream 2.0:

Amazon AppStream 2.0 is a fully managed application streaming service that provides users with instant access to their desktop applications from anywhere. AppStream 2.0 manages the AWS resources required to host and run your applications, scales automatically, and provides access to your users on demand. AppStream 2.0 provides users access to the applications they need on the desktop device of their choice, with a responsive, fluid user experience that is indistinguishable from natively installed applications. There are no files to download and no time-consuming installations.

## Cloud Formation Stack:

* **A stack is a collection of AWS resources that you can manage as a single unit.** In other words, you can create, update, or delete a collection of resources by creating, updating, or deleting stacks. **All the resources in a stack are defined by the stack's AWS CloudFormation template.** A stack, for instance, can include all the resources required to run a web application, such as a web server, a database, and networking rules. If you no longer require that web application, you can simply delete the stack, and all of its related resources are deleted.
* AWS CloudFormation ensures all stack resources are created or deleted as appropriate. Because AWS CloudFormation treats the stack resources as a single unit, they must all be created or deleted successfully for the stack to be created or deleted. If a resource cannot be created, AWS CloudFormation rolls the stack back and automatically deletes any resources that were created. If a resource cannot be deleted, any remaining resources are retained until the stack can be successfully deleted.
* You can work with stacks by using the AWS CloudFormation console, API, or AWS CLI.You are charged for the stack resources for the time they were operating (even if you deleted the stack right away).

## Amazon Elastic Search:

* Amazon Elasticsearch Service (Amazon ES) is a managed service that makes it easy to deploy, operate, and scale Elasticsearch clusters in the AWS Cloud. Elasticsearch is a popular open-source search and analytics engine for use cases such as log analytics, real-time application monitoring, and clickstream analytics. With Amazon ES, you get direct access to the Elasticsearch APIs so that existing code and applications work seamlessly with the service.
* Amazon ES provisions all the resources for your Elasticsearch cluster and launches the cluster. It also automatically detects and replaces failed Elasticsearch nodes, reducing the overhead associated with self-managed infrastructures. You can scale your cluster with a single API call or a few clicks in the console.
* To get started using the service, you create an Amazon ES domain. An Amazon ES domain is an Elasticsearch cluster in the AWS Cloud that has the compute and storage resources that you specify. For example, you can specify the number of instances, instance types, and storage options.

## Cloud Trail Features:

* It is enabled globally
* It is enabled by default
* It is enabled on a per region basis. Can be applied to all regions in the trail configuration page
* Logs can be delivered to a single s3 bucket for aggregation in any region
* It is enabled for all the supported AWS services

## Elastic Cache, 2 Types of open source in-memory engines:

Web service that makes it easy to deploy, operate and scale in-memory data store or cache in the cloud.

* Redis
* Memcached

**Cache Nodes size and number of cache nodes are important**

## AWS SWF: (Simple Work Flow)

Amazon Simple Workflow Service (SWF) is a web service that makes it easy to coordinate work across distributed application components. Amazon SWF enables applications for a range of use cases, including media processing, web application back-ends, business process workflows, and analytics pipelines, to be designed as a coordination of tasks. Tasks represent invocations of various processing steps in an application which can be performed by executable code, web service calls, human actions, and scripts.

The coordination of tasks involves managing execution dependencies, scheduling, and concurrency in accordance with the logical flow of the application. With Amazon SWF, developers get full control over implementing processing steps and coordinating the tasks that drive them, without worrying about underlying complexities such as tracking their progress and keeping their state. Amazon SWF also provides the AWS Flow Framework to help developers use asynchronous programming in the development of their applications. By using Amazon SWF, developers benefit from ease of programming and have the ability to improve their applications’ resource usage, latencies, and throughput.

**SWF Task and Workflow execution can last up to 1 year,** can also include tasks to be performed by on-premises server and humans.

## Sticky sessions in ELB:

When you use HTTP/HTTPS, you can enable sticky sessions on your load balancer. A sticky session binds a user's session to a specific back-end instance. This ensures that all requests coming from the user during the session are sent to the same back-end instance. **ELB is stateful and uses cookie-based session.**

## ELB Listener configuration notes:

Before you start using Elastic Load Balancing, you must configure one or more listeners for your Classic Load Balancer. **A listener is a process that checks for connection requests. It is configured with a protocol and a port for front-end (client to load balancer) connections, and a protocol and a port for back-end (load balancer to back-end instance) connections**.

Elastic Load Balancing supports the following protocols:

* HTTP
* HTTPS (secure HTTP)
* TCP
* SSL (secure TCP)

The HTTPS protocol uses the SSL protocol to establish secure connections over the HTTP layer. You can also use the SSL protocol to establish secure connections over the TCP layer.

**If the front-end connection uses TCP or SSL, then your back-end connections can use either TCP or SSL. If the front-end connection uses HTTP or HTTPS, then your back-end connections can use either HTTP or HTTPS.**

## For HA, AWS services to be implemented in multiple AZ:

* EC2
* ELB

Note:

**DynamoDB, SNS, S3 -> AWS Managed services that are scalable and HA by themselves**

## Services that natively encrypt data at rest within a region:

* Storage Gateway
* Glacier

## IAM Accessibility:

Accessible from all interfaces like Console, CLI, Query API and SDK

## Consolidated Billing Account Types:

* Paying/Payer Account
* Linked Account

## Dynamo DB Features:

* NOSQL Database, Low latency, fully managed cloud DB
* Document and key value store models
* Durable, Scalable, HA, High uptime Data Store : Synchronous replication across 3AZs within a region
* Useful for Real time public Tabulation, web sessions management, JSON documents storage, metadata storage of S3 objects
* **HASH Key determines the partition, hence affects performance**
* **Atomic updates are fast and in-place. Numeric attribute in a row can be incremented or decremented using a single API call. Sets, lists or maps can be atomically added or removed.**
* Dynamo DB Tables can be created with same name as IAM user name
* IAM Group level Policy rule can be defined to grant access based on the DynamoDB ARN using a variable
* Throttling prevents your application from consuming too many capacity units. When a request is throttled, it fails with an HTTP 400 code (Bad Request) and a ProvisionedThroughputExceededException.
* **Eventual READ CONSISTENCY is provided by default.** Can be made strongly consistent too

Note:

S3 suited for BLOB Storage

## Storage Gateway volumes Accessibility:

* NOT accessible using S3 APIs
* Accessible from AWS Storage volumes (we can take a gateway snapshot too)

## Amazon Workspaces:

Amazon WorkSpaces is a managed, secure **cloud desktop service**. You can use Amazon WorkSpaces to provision either Windows or Linux desktops in just a few minutes and quickly scale to provide thousands of desktops to workers across the globe. You can pay either monthly or hourly, just for the WorkSpaces you launch, which helps you save money when compared to traditional desktops and on-premises VDI solutions. Amazon WorkSpaces helps you eliminate the complexity in managing hardware inventory, OS versions and patches, and Virtual Desktop Infrastructure (VDI), which helps simplify your desktop delivery strategy. **With Amazon WorkSpaces, your users get a fast, responsive desktop of their choice that they can access anywhere, anytime, from any supported device.**

## RDS Instance Maintenance Window:

The Amazon RDS maintenance window is your opportunity to control when DB instance modifications (such as scaling DB instance class) and software patching occur, in the event they are requested or required. If a maintenance event is scheduled for a given week, it will be initiated and completed at some point during the maintenance window you identify. Maintenance windows are 30 minutes in duration.

The only maintenance events that require Amazon RDS to take your DB instance offline are scale compute operations (which generally take only a few minutes from start-to-finish) or required software patching. Required patching is automatically scheduled only for patches that are security and durability related. Such patching occurs infrequently (typically once every few months) and should seldom require more than a fraction of your maintenance window. If you do not specify a preferred weekly maintenance window when creating your DB instance, a 30 minute default value is assigned. If you wish to modify when maintenance is performed on your behalf, you can do so by modifying your DB instance in the AWS Management Console, the ModifyDBInstance API or the modify-db-instance command. Each of your DB instances can have different preferred maintenance windows, if you so choose.Running your DB instance as a Multi-AZ deployment can further reduce the impact of a maintenance event.

## RDS Oracle Licensing Models:

* You can run Amazon RDS for Oracle under two different licensing models – “**License Included**” and “**Bring-Your-Own-License (BYOL**)”.
* In the "License Included" service model, you do not need separately purchased Oracle licenses; the Oracle Database software has been licensed by AWS. "License Included" pricing starts at $0.04 per hour, inclusive of software, underlying hardware resources, and Amazon RDS management capabilities.
* If you already own Oracle Database licenses, you can use the "BYOL" model to run Oracle databases on Amazon RDS, with rates starting at $0.025 per hour. The “BYOL” model is designed for customers who prefer to use existing Oracle database licenses or purchase new licenses directly from Oracle.

## AWS Responsibilities for security measures:

* Distributed Denial Of Service (DDoS) Attacks :

AWS API endpoints are hosted on large, Internet-scale, world- class infrastructure that benefits from the same engineering expertise that has built Amazon into the world’s largest online retailer. Proprietary DDoS mitigation techniques are used. Additionally, AWS’ networks are multi- homed across a number of providers to achieve Internet access diversity.

* Man in the Middle (MITM) Attacks :

All of the AWS APIs are available via SSL protected endpoints which provide server authentication. Amazon EC2 AMIs automatically generate new SSH host certificates on first boot and log them to the instance’s console. You can then use the secure APIs to call the console and

access the host certificates before logging into the instance for the first time. We

encourage you to use SSL for all of your interactions with AWS.

* IP Spoofing:

Amazon EC2 instances cannot send spoofed network traffic. The AWS-controlled, host-based firewall infrastructure will not permit an instance to send traffic with a source IP or MAC address other than its own.

* Port Scanning:

Unauthorized port scans by Amazon EC2 customers are a violation of the AWS Acceptable Use Policy. Violations of the AWS Acceptable Use Policy are taken seriously, and every reported violation is investigated. Customers can report suspected abuse via the contacts available on our website at: http://aws.amazon.com/contact-us/report-abuse/. When unauthorized port scanning is detected by AWS, it is stopped and blocked. Port scans of Amazon EC2 instances are generally ineffective because, by default, all inbound ports on Amazon EC2 instances are closed and are only opened by you. Your strict management of security groups can further mitigate the threat of port scans. If you configure the security group to allow traffic from any source to a specific port, then that specific port will be vulnerable to a port scan. In these cases, you must use appropriate security measures to protect listening services that may be essential to their application from being discovered by an unauthorized port scan. For example, a web server must clearly have port 80 (HTTP) open to the world, and the administrator of this server is responsible for the security of the HTTP server software, such as Apache. You may request permission to conduct vulnerability scans as required to meet your specific compliance requirements. These scans must be limited to your own instances and must not violate the AWS Acceptable Use Policy.

* Packet sniffing by other tenants:

It is not possible for a virtual instance running in promiscuous mode to receive or “sniff” traffic that is intended for a different virtual instance. While you can place your interfaces into promiscuous mode, the hypervisor will not deliver any traffic to them that is not addressed to them. Even two virtual instances that are owned by the same customer located on the same physical host cannot listen to each other’s traffic. Attacks such as ARP cache poisoning do not work within Amazon EC2 and Amazon VPC. While Amazon EC2 does provide ample protection against one customer inadvertently or maliciously attempting to view another’s data, as a standard practice you should encrypt sensitive traffic.

## Domain Name Service(DNS):

DNS is a globally distributed service that translates human readable names like www.example.com into the numeric IP addresses like 192.0.2.1 that computers use to connect to each other. The Internet’s DNS system works much like a phone book by managing the mapping between names and numbers. For DNS, the names are domain names (www.example.com) that are easy for people to remember and the numbers are IP addresses (192.0.2.1) that specify the location of computers on the Internet. DNS servers translate requests for names into IP addresses, controlling which server an end user will reach when they type a domain name into their web browser. These requests are called "queries."

## Amazon Route 53

Amazon Route 53 provides highly available and scalable Domain Name System (DNS), domain name registration, and health-checking web services. It is designed to give developers and businesses an extremely reliable and cost effective way to route end users to Internet applications by translating names like example.com into the numeric IP addresses, such as 192.0.2.1, that computers use to connect to each other. You can combine your DNS with health-checking services to route traffic to healthy endpoints or to independently monitor and/or alarm on endpoints. You can also purchase and manage domain names such as example.com and automatically configure DNS settings for your domains. Route 53 effectively connects user requests to infrastructure running in AWS – such as Amazon EC2 instances, Elastic Load Balancing load balancers, or Amazon S3 buckets – and can also be used to route users to infrastructure outside of AWS.

If you already have a domain name:

* Use the AWS Management Console or the CreateHostedZone API to create a hosted zone that can store DNS records for your domain. Upon creating the hosted zone, you receive **four Route53 name servers across four different Top-Level Domains (TLDs)** to help ensure a high level of availability.
* Additionally, you can transfer your domain name to Route 53’s management via either the AWS Management Console or the API.

If you don't already have a domain name:

* Use the AWS Management Console or the API to register your new domain name.
* Route 53 automatically creates a hosted zone that stores DNS records for your domain. **You also receive four Route 53 name servers across four different Top-Level Domains (TLDs) to help ensure a high level of availability.**

## Difference between domain and hosted zone:

* A domain is a general DNS concept. Domain names are easily recognizable names for numerically addressed Internet resources. For example, amazon.com is a domain.
* A hosted zone is an Amazon Route 53 concept. A hosted zone is analogous to a traditional DNS zone file; it represents a collection of records that can be managed together, belonging to a single parent domain name. All resource record sets within a hosted zone must have the hosted zone’s domain name as a suffix. For example, the amazon.com hosted zone may contain records named www.amazon.com, and www.aws.amazon.com, but not a record named [www.amazon.ca](http://www.amazon.ca).

## DNS Record types supported by Route 53:

Amazon Route 53 currently supports the following DNS record types:

* A (address record)
* AAAA (IPv6 address record)
* CNAME (canonical name record)
* CAA (certification authority authorization)
* MX (mail exchange record)
* NAPTR (name authority pointer record)
* NS (name server record)
* PTR (pointer record)
* SOA (start of authority record)
* SPF (sender policy framework)
* SRV (service locator)
* TXT (text record)
* Additionally, Amazon Route 53 offers ‘Alias’ records (an Amazon Route 53-specific virtual record). **Alias records are used to map resource record sets in your hosted zone to Amazon Elastic Load Balancing load balancers, Amazon CloudFront distributions, AWS Elastic Beanstalk environments, or Amazon S3 buckets that are configured as websites**.
* **Alias records work like a CNAME record in that you can map one DNS name (example.com) to another ‘target’ DNS name (elb1234.elb.amazonaws.com). They differ from a CNAME record in that they are not visible to resolvers. Resolvers only see the A record and the resulting IP address of the target record.**

## Top Level Domains offered by Route 53:

Route 53 offers a wide selection of both generic Top Level Domains (“gTLDs”: for example, .com and .net) and country-code Top Level Domains (“ccTLDs”: for example, .de and .fr). For the complete list, please see the Route 53 Domain Registration Price List.

## Types of backup ups offered by RDS:

* **Amazon RDS provides two different methods for backing up and restoring your DB instance(s) automated backups and database snapshots (DB Snapshots).**
* The automated backup feature of Amazon RDS enables point-in-time recovery of your DB instance. When automated backups are turned on for your DB Instance, Amazon RDS automatically performs a full daily snapshot of your data (during your preferred backup window) and captures transaction logs (as updates to your DB Instance are made). When you initiate a point-in-time recovery, transaction logs are applied to the most appropriate daily backup in order to restore your DB instance to the specific time you requested. **Amazon RDS retains backups of a DB Instance for a limited, user-specified period of time called the retention period, which by default is 7 days but can be set to up to 35 days.** You can initiate a point-in-time restore and specify any second during your retention period, up to the Latest Restorable Time. You can use the DescribeDBInstances API to return the latest restorable time for you DB instance, which is typically within the last five minutes. Alternatively, you can find the Latest Restorable Time for a DB instance by selecting it in the AWS Management Console and looking in the “Description” tab in the lower panel of the Console.
* **DB Snapshots are user-initiated and enable you to back up your DB instance in a known state as frequently as you wish**, and then restore to that specific state at any time. DB Snapshots can be created with the AWS Management Console, CreateDBSnapshot API, or create-db-snapshot command and are kept until you explicitly delete them.
* The snapshots which Amazon RDS performs for enabling automated backups are available to you for copying (using the AWS console or the copy-db-snapshot command) or for the snapshot restore functionality. You can identify them using the "automated" Snapshot Type. In addition, you can identify the time at which the snapshot has been taken by viewing the "Snapshot Created Time" field. Alternatively, the identifier of the "automated" snapshots also contains the time (in UTC) at which the snapshot has been taken.
* Please note: **When you perform a restore operation to a point in time or from a DB Snapshot, a new DB Instance is created with a new endpoint (the old DB Instance can be deleted if so desired). This is done to enable you to create multiple DB Instances from a specific DB Snapshot or point in time.**
* Do I need to enable backups for my DB Instance or is it done automatically?

By default, Amazon RDS enables automated backups of your DB Instance with a 7 day retention period. Free backup storage is limited to the size of your provisioned database and only applies to active DB Instances. For example, if you have 100 GB of provisioned database storage over the month, we will provide 100 GB-months of backup storage at no additional charge. If you would like to extend your backup retention period beyond one day, you can do so using the CreateDBInstance API (when creating a new DB Instance) or ModifyDBInstance API (for an existing DB Instance). You can use these APIs to change the RetentionPeriod parameter from 1 to the desired number of days.

* What is a backup window and why do I need it? Is my database available during the backup window?

The preferred backup window is the user-defined period of time during which your DB Instance is backed up. Amazon RDS uses these periodic data backups in conjunction with your transaction logs to enable you to restore your DB Instance to any second during your retention period, up to the LatestRestorableTime (typically up to the last few minutes). **During the backup window, storage I/O may be briefly suspended while the backup process initializes (typically under a few seconds) and you may experience a brief period of elevated latency. There is no I/O suspension for Multi-AZ DB deployments, since the backup is taken from the standby.**

* What happens to my backups and DB snapshots if I delete my DB instance?
* When you delete a DB instance, you can create a final DB snapshot upon deletion; if you do, you can use this DB snapshot to restore the deleted DB instance at a later date. Amazon RDS retains this final user-created DB snapshot along with all other manually created DB snapshots after the DB instance is deleted.
* **Automated backups are deleted when the DB instance is deleted. Only manually created DB Snapshots are retained after the DB Instance is deleted.**

## RDS Pricing:

Data transferred between RDS and EC2 Instances in the same AZ is priced as below:

* Using Private IP Address – Free
* Using Public/Elastic IP Address – 0.01$ Per Gb

## Starting and stopping RDS instances:

* Amazon RDS for MySQL, MariaDB, PostgreSQL, Oracle and SQL Server now allows you to easily stop and start your database instances. This makes it easy and affordable to use databases for development and test purposes, where the database is not required to be running all of the time.
* **While your database instance is stopped, you are charged for provisioned storage, manual snapshots and automated backup storage within your specified retention window, but not for database instance hours**. While a database instance is stopped, Amazon RDS does not delete any of your automatic backups or transaction logs. This means you can do a point-in-time restore to any point within your specified automated backup retention window, even after an instance is started. Starting an instance restores it to the same configuration as it had when stopped, including its endpoint, DB parameter group, security group, and option group membership.
* **You can stop an instance for up to 7 days at a time. After 7 days, it will be automatically started.** **The stop/start feature is available for database instances running in a Single-AZ deployment which are not part of a Read Replica (both source and replica) configuration**

## Encrypt connections between my application and my DB Instance using SSL:

* **Yes, this option is currently supported for the MySQL, MariaDB, SQL Server, PostgreSQL, and Oracle engines. Amazon RDS generates an SSL certificate for each DB Instance.**
* **Once an encrypted connection is established, data transferred between the DB Instance and your application will be encrypted during transfer.**
* While SSL offers security benefits, be aware that SSL encryption is a compute-intensive operation and will increase the latency of your database connection. SSL support within Amazon RDS is for encrypting the connection between your application and your DB Instance; it should not be relied on for authenticating the DB Instance itself.

## Handling Auto Scaling that scales up and down multiple times in the same hour:

* Scaling is happening frequently either because Cloud Watch alarms that are configured to cause auto scaling to go up and down fast (OR)
* Cool down timers are small due to which auto scaling activity is triggered before the new instance gets a chance to handle traffic
* So we need to modify the Cloud Watch Alarm or the cool down timer

## Apply immediately parameter when modifying a DB Instance:

* When you modify a DB instance, you can apply the changes immediately. To apply changes immediately, you select the Apply Immediately option in the AWS Management Console, you use the --apply-immediately parameter when calling the AWS CLI, or you set the ApplyImmediately parameter to true when using the Amazon RDS API.
* If you don't choose to apply changes immediately, the changes are put into the pending modifications queue. During the next maintenance window, any pending changes in the queue are applied. If you choose to apply changes immediately, your new changes and any changes in the pending modifications queue are applied.
* If any of the pending modifications require downtime, choosing apply immediately can cause unexpected downtime. Changes to some database settings are applied immediately, even if you choose to defer your changes. To see how the different database settings interact with the apply immediately setting; see the settings for your specific database engine.

## Subnets characteristics:

* Each subnet maps to 1 AZ (resides with an AZ) and cannot span across AZs
* By default, all subnets can route to each other (Private or Public)
* Subnet Types : Public, Private and VPN
* If a subnet's traffic is routed to an internet gateway, the subnet is known as a public subnet.
* If a subnet doesn't have a route to the internet gateway, the subnet is known as a private subnet.
* If a subnet doesn't have a route to the internet gateway, but has its traffic routed to a virtual private gateway for a VPN connection, the subnet is known as a VPN-only subnet.
* Each subnet in your VPC must be associated with a route table; the table controls the routing for the subnet. A subnet can only be associated with one route table at a time, but you can associate multiple subnets with the same route table.

## Route Table characteristics:

* When you create a VPC, it automatically has a main route table.
* Your VPC can have route tables other than the default table. One way to protect your VPC is to leave the main route table in its original default state (with only the local route), and explicitly associate each new subnet you create with one of the custom route tables you've created. This ensures that you explicitly control how each subnet routes outbound traffic.

## CIDR Blocks: (Classless Inter-Domain Routing)

* Block Mask of **/28** is the smallest range supported
* A single CIDR Block can be assigned to a VPC
* The allowed block size is between a /16 netmask (65,536 IP addresses) and /28 netmask (16 IP addresses)
* For example, if you create a VPC with CIDR block 10.0.0.0/24, it supports 256 IP addresses. You can break this CIDR block into two subnets, each supporting 128 IP addresses. One subnet uses CIDR block 10.0.0.0/25 (for addresses 10.0.0.0 - 10.0.0.127) and the other uses CIDR block 10.0.0.128/25 (for addresses 10.0.0.128 - 10.0.0.255).
* The first four IP addresses and the last IP address in each subnet CIDR block are not available for you to use, and cannot be assigned to an instance. For example, in a subnet with CIDR block 10.0.0.0/24, the following five IP addresses are reserved:
* 10.0.0.0: Network address.
* 10.0.0.1: Reserved by AWS for the VPC router.
* 10.0.0.2: Reserved by AWS. The IP address of the DNS server is always the base of the VPC network range plus two; however, we also reserve the base of each subnet range plus two. For VPCs with multiple CIDR blocks, the IP address of the DNS server is located in the primary CIDR. For more information, see Amazon DNS Server.
* 10.0.0.3: Reserved by AWS for future use.
* 10.0.0.255: Network broadcast address. We do not support broadcast in a VPC, therefore we reserve this address.
* When you add or remove a CIDR block, it can go through various states: associating | associated | disassociating | disassociated | failing | failed. The CIDR block is ready for you to use when it's in the associated state.

## VPC Public vs Private IP Addresses:

* IP addresses enable resources in your VPC to communicate with each other, and with resources over the Internet. Amazon EC2 and Amazon VPC support the IPv4 and IPv6 addressing protocols.
* By default, Amazon EC2 and Amazon VPC use the IPv4 addressing protocol. When you create a VPC, you must assign it an IPv4 CIDR block (a range of private IPv4 addresses). **Private IPv4 addresses are not reachable over the Internet. To connect to your instance over the Internet, or to enable communication between your instances and other AWS services that have public endpoints, you can assign a globally-unique public IPv4 address to your instance.**
* You can optionally associate an IPv6 CIDR block with your VPC and subnets, and assign IPv6 addresses from that block to the resources in your VPC. **IPv6 addresses are public and reachable over the Internet.**
* To ensure that your instances can communicate with the Internet, you must also attach an Internet gateway to your VPC.
* When you launch an instance into a VPC, a primary private IP address from the IPv4 address range of the subnet is assigned to the default network interface (eth0) of the instance.
* When you launch an instance into a subnet that has the public attribute enabled, a public IP address is assigned to the primary network interface (eth0) that's created for the instance. A public IP address is mapped to the primary private IP address through network address translation (NAT).
* An instance receives an Amazon-provided private DNS hostname that corresponds to its private IPv4 address, and if applicable, a public DNS hostname that corresponds to its public IPv4 or Elastic IP address.
* A public IP address is assigned from Amazon's pool of public IP addresses; it's not associated with your account. When a public IP address is disassociated from your instance, it's released back into the pool, and is no longer available for you to use. You cannot manually associate or disassociate a public IP address. Instead, in certain cases, we release the public IP address from your instance, or assign it a new one.
* **If you require a persistent public IP address allocated to your account that can be assigned to and removed from instances as you require, use an Elastic IP address instead.**
* A private IP address remains associated with the network interface when the instance is stopped and restarted, and is released when the instance is terminated.
* You can assign additional private IP addresses, known as secondary private IP addresses, to instances that are running in a VPC. Unlike a primary private IP address, you can reassign a secondary private IP address from one network interface to another.

## SES:

SES can’t be integrated with Cloud Watch while SNS can.

## Visibility Time Out for SQS:

Visibility timeout is important because if it set to a low value, the message will get visible even before the message was deleted, causing it to be read twice

**Consumer must delete the message after receiving and processing it. SQS does not automatically delete messages.**

## Virtual Gateway for VPN connection:

* An internet gateway enables communication over the internet, and a virtual private network (VPN) connection enables communication with your corporate network.
* A virtual private gateway is the VPN concentrator on the Amazon side of the VPN connection. You create a virtual private gateway and attach it to the VPC from which you want to create the VPN connection. A customer gateway is a physical device or software application on your side of the VPN connection.
* VPN Routing Options
* When you create a VPN connection, you must do the following:
* Specify the type of routing that you plan to use (static or dynamic)
* Update the route table for your subnet
* Static and Dynamic Routing
* The type of routing that you select can depend on the make and model of your VPN devices. If your VPN device supports **Border Gateway Protocol (BGP), specify dynamic routing** when you configure your VPN connection. If your device does not support BGP, specify static routing.
* We recommend that you use BGP-capable devices, when available, because the BGP protocol offers robust liveness detection checks that can assist failover to the second VPN tunnel if the first tunnel goes down. Devices that don't support BGP may also perform health checks to assist failover to the second tunnel when needed.
* If you have overlapping routes within a VPN connection and longest prefix match cannot be applied, then **we prioritize the routes as follows in the VPN connection, from most preferred to least preferred:**
* **BGP propagated routes from an AWS Direct Connect connection**
* **Manually added static routes for a VPN connection**
* **BGP propagated routes from a VPN connection**
* In this example, your route table has a static route to an internet gateway (that you added manually), and a propagated route to a virtual private gateway. Both routes have a destination of 172.31.0.0/24. In this case, all traffic destined for 172.31.0.0/24 is routed to the internet gateway — it is a static route and therefore takes priority over the propagated route.

|  |  |
| --- | --- |
| **Destination** | **Target** |
| 10.0.0.0/16 | Local |
| 172.31.0.0/24 | vgw-1a2b3c4d (propagated) |
| 172.31.0.0/24 | igw-11aa22bb |

* You use a VPN connection to connect your remote network to a VPC. Each VPN connection has two tunnels, with each tunnel using a unique virtual private gateway public IP address. It is important to configure both tunnels for redundancy. When one tunnel becomes unavailable (for example, down for maintenance), network traffic is automatically routed to the available tunnel for that specific VPN connection.
* To protect against a loss of connectivity in case your customer gateway becomes unavailable, you can set up a second VPN connection to your VPC and virtual private gateway by using a second customer gateway. By using redundant VPN connections and customer gateways, you can perform maintenance on one of your customer gateways while traffic continues to flow over the second customer gateway's VPN connection. To establish redundant VPN connections and customer gateways on your remote network, you need to set up a second VPN connection. The customer gateway IP address for the second VPN connection must be publicly accessible.
* **Dynamically routed VPN connections use the Border Gateway Protocol (BGP) to exchange routing information between your customer gateways and the virtual private gateways.** Statically routed VPN connections require you to enter static routes for the remote network on your side of the customer gateway. BGP-advertised and statically entered route information allow gateways on both sides to determine which tunnels are available and reroute traffic if a failure occurs. We recommend that you configure your network to use the routing information provided by BGP (if available) to select an available path. The exact configuration depends on the architecture of your network.

## Elastic Beanstalk:

AWS Elastic Beanstalk makes it even easier for developers to quickly deploy and manage applications in the AWS Cloud. Developers simply upload their application, and Elastic Beanstalk automatically handles the deployment details of capacity provisioning, load balancing, auto-scaling, and application health monitoring. Those who want to deploy and manage their applications within minutes in the AWS Cloud. You don’t need experience with cloud computing to get started. AWS Elastic Beanstalk supports Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker web applications.

## ENI for failover:

An elastic network interface (referred to as a network interface in this documentation) is a logical networking component in a VPC that represents a virtual network card.

A network interface can include the following attributes:

* A primary private IPv4 address from the IPv4 address range of your VPC
* One or more secondary private IPv4 addresses from the IPv4 address range of your VPC
* One Elastic IP address (IPv4) per private IPv4 address
* One public IPv4 address
* One or more IPv6 addresses
* One or more security groups
* A MAC address
* A source/destination check flag
* A description
* You can create and configure network interfaces in your account and attach them to instances in your VPC.
* Every instance in a VPC has a default network interface, called the primary network interface (eth0). **You cannot detach a primary network interface from an instance.**
* To ensure failover capabilities, consider using a secondary IPV4 address (attached to the primary ENI) for incoming traffic on a network interface. In the event of instance failure, you can move the interface and / or the secondary private IPV4 address to the standby instance.
* In a VPC, all subnets have a modifiable attribute that determines whether network interfaces created in that subnet (and therefore instances launched into that subnet) are assigned a public IPv4 address. When you launch an instance, the IP address is assigned to the primary network interface (eth0) that's created.
* **Security groups are associated with network interfaces.** **Changing an instance's security groups changes the security groups associated with the primary network interface (eth0).**
* **Incoming traffic is allowed based on the private IP addresses of the instances that are**

## RAID 0 and Striping:

RAID 0 consists of striping, but no mirroring or parity. Compared to a spanned volume, the capacity of a RAID 0 volume is the same; it is sum of the capacities of the disks in the set. But because striping distributes the contents of each file among all disks in the set, the failure of any disk causes all files, the entire RAID 0 volume, to be lost. A broken spanned volume at least preserves the files on the unfailing disks. **The benefit of RAID 0 is that the throughput of read and write operations to any file is multiplied by the number of disks because, unlike spanned volumes, reads and writes are done concurrently,[11] and the cost is complete vulnerability to drive failures.**

In computer data storage, data striping is the technique of segmenting logically sequential data, such as a file, so that consecutive segments are stored on different physical storage devices.

Striping is useful when a processing device requests data more quickly than a single storage device can provide it. By spreading segments across multiple devices which can be accessed concurrently, total data throughput is increased. It is also a useful method for balancing I/O load across an array of disks. Striping is used across disk drives in redundant array of independent disks (RAID) storage, network interface controllers, different computers in clustered file systems and grid-oriented storage, and RAM in some systems. **Advantages of striping include performance and throughput**

**Because different segments of data are kept on different storage devices, the failure of one device causes the corruption of the full data sequence. In effect, the failure rate of the array of storage devices is equal to the sum of the failure rate of each storage device. This disadvantage of striping can be overcome by the storage of redundant information, such as parity, for the purpose of error correction.** In such a system, the disadvantage is overcome at the cost of requiring extra storage.

In some RAID configurations, such as RAID 0, failure of a single member drive of the RAID array causes all stored data to be lost. In other RAID configurations, such as a RAID 5 that contains distributed parity and provides redundancy, if one member drive fails the data can be restored using the other drives in the array.

## Default Visibility Timeout of message in SQS Queue:

30 seconds. Each SQS message which is 64KB is billed as 1 request. A single request can have from 1 to 10 messages, up to a maximum total payload of 256 KB.

## ELB Diverting Traffic pattern:

ELB sends traffic to instances when healthy. Once detected as unhealthy, stops sending traffic to the instances. After detecting that the instance is healthy again, it resumes routing traffic again to that instance. **ELB Performs health check on all the registered instances, irrespective of whether the instance is healthy or unhealthy**

## ECM: (Enterprise Content Management)

OmniDocs is an Enterprise Content Management system for creating, capturing, managing, delivering, and archiving large volumes of documents and content. OmniDocs handles scanned document images, electronic documents, emails, and electronic data output from other applications with equal efficiency and ease.

## Import Key Pair Feature:

Key pairs have regional level scope. Imports the public key from an RSA key pair that you created with a third-party tool. Compare this with create-key-pair , in which AWS creates the key pair and gives the keys to you (AWS keeps a copy of the public key). **With ImportKeyPair, you create the key pair and give AWS just the public key. The private key is never transferred between you and AWS.**

You can now import your own RSA keypair (or the public half, to be precise) for use with your Amazon EC2 instances. Why would you want to do this? Here are a couple of reasons:

* Trust – By importing your own keypair you can ensure that you have complete control over your keys.
* Security -You can be confident that your private key has never been transmitted over the wire.
* **Management of Multiple Regions – You can use the same public key across multiple AWS Regions.**

You can upload RSA keys (which can be 1024, 2048, or 4096 bits long) in a variety of formats including OpenSSH public key format, Base64 encoded DER format, or the SSH public key file format specified in RFC 4716. The ssh-keygen tool (part of the standard OpenSSH installation) is a handy way to create keys.

## Second Network Interface:

When you add a second network interface, the system can no longer auto-assign a public IPv4 address. **You will not be able to connect to the instance over IPv4 unless you assign an Elastic IP address to the primary network interface (eth0).** You can assign the Elastic IP address after you complete the Launch wizard.

## Configuring the Operating System on Your Instance to Recognize the Secondary Private IPv4 Address

After you assign a secondary private IPv4 address to your instance, you need to configure the operating system on your instance to recognize the secondary private IP address.

If you are using Amazon Linux, the **ec2-net-utils package** can take care of this step for you**. It configures additional network interfaces that you attach while the instance is running, refreshes secondary IPv4 addresses during DHCP lease renewal, and updates the related routing rules.** You can immediately refresh the list of interfaces by using the command **sudo service network restart and then view the up-to-date list using ip addr li.** If you require manual control over your network configuration, you can remove the ec2-net-utils package

## Provisioned IOPS:

The maximum ratio of provisioned IOPS to requested volume size (in GiB) is 50:1. For example, a 100 GiB volume can be provisioned with up to 5,000 IOPS. Any volume 640 GiB in size or greater allows provisioning up to the 32,000 IOPS maximum (50 × 640 GiB = 32,000).

## Edge Locations:

Edge locations serve content for CloudFront and Route53. CloudFront is a CDN (Content Delivery Network) while Route53 is a DNS service. Requests going to one of these services are routed to the nearest edge location automatically. This helps to achieve low latency.

## Storing session data externally for making the application stateless:

Session data can be stored in either of the following:

* Elastic Cache
* Dynamo DB
* RDS

## IAM Login Ids, alias and groups:

* For multiple users of the same account, it is **NOT** possible to have the same login id
* Account alias must also be unique across all AWS products. Your AWS account can have only one alias. It must contain only digits, lowercase letters, and hyphens.
* Group ID can also be used to login
* 1 MFA device can be enabled per AWS root account or IAM user. Device can be used by specified user only

## Network Throughput / Network Bandwidth and Instance size:

A benchmark can be used to measure throughput. **In data transmission, network throughput is the amount of data moved successfully from one place to another in a given time period,** and typically measured in bits per second (bps), as in megabits per second (Mbps) or gigabits per second (Gbps).

The **instance size** determines the network throughput or bandwidth associated with the instance.

This example applies only to Amazon EC2 instances in the same VPC. Here are some factors that can affect Amazon EC2 network performance:

* The **physical proximity of EC2 instances**. Instances in the same Availability Zone are geographically closest to each other, while instances in different Availability Zones in the same region, instances in different regions on the same continent, and instances in different regions on different continents are progressively geographically farther away from one another.
* EC2 instance **maximum transmission unit (MTU).** The default interface configuration for EC2 instances uses jumbo frames if they are one of the instance sizes listed in Jumbo Frames (9001 MTU).
* **The size of your EC2 instance.** Larger instance sizes for an instance type typically provide better network performance than smaller instance sizes of the same type.
* EC2 **enhanced networking support** for Linux, with the exception of T2 and M3 instance types.
* EC2 **high performance computing (HPC) support using placement groups**. HPC provides full-bisection bandwidth and low latency, with support for up to 25-gigabit network speeds depending on the instance type.

## KPL:

Kinesis Producer Library.

**Data records are accessible only for a period of 24 hours from the time they are added to a stream. You can extend to 7 days by enabling extended data retention.**

## DNS Failovers to a Static S3 website:

Route 53 helps for DNS Failovers to a Static S3 website as DNS Failovers and S3 static hosting pairs well as a back up site.

## Cloud Hub:

If you have more than one remote network (for example, multiple branch offices), you can create multiple AWS managed VPN connections via your virtual private gateway to enable communication between these networks.

## Internet Gateways

An internet gateway is a horizontally scaled, redundant, and highly available VPC component that allows communication between instances in your VPC and the internet. It therefore imposes no availability risks or bandwidth constraints on your network traffic.

An internet gateway serves two purposes: to provide a target in your VPC route tables for internet-routable traffic, and to perform network address translation (NAT) for instances that have been assigned public IPv4 addresses.

An internet gateway supports IPv4 and IPv6 traffic.

## Optimistic vs Pessimistic concurrency:

|  |  |
| --- | --- |
| **Optimistic Concurrency approach** | **Pessimistic Concurrency approach** |
| * Allow concurrency conflict happens and if it happens, we react on it in some manner      * Best solution when concurrency possibility is rather low      * Doesn’t lock records – to ensure record wasn’t changed in time between select & submit operations, it checks row version      * Simple in designing and programming      * Suits best when database has a lot of records and not too many (relatively) users | * Protects system from concurrency conflict so it will not happen      * Best solution when there is a lot of updates and concurrency possibility is high      * Locks records so record selected for update will not be updated meantime by another user      * More complex in designing and managing the programming part (deadlocks’ risk)      * Suits well when we have a table with relatively small amount of records but a lot of update operations . Often transaction rollback would be an ‘effort waste’ [1] |

## Instance Limits:

You are limited to running up to a total of **20** On-Demand instances across the instance family, purchasing **20** Reserved Instances, and requesting Spot Instances per your dynamic Spot limit per region. New AWS accounts may start with limits that are lower than the limits described here.

## Instance Billing:

Billing commences when Amazon EC2 initiates the boot sequence of an AMI instance. Billing ends when the instance terminates, which could occur through a web services command, by running "shutdown -h", or through instance failure. When you stop an instance, we shut it down but don't charge hourly usage for a stopped instance, or data transfer fees, but we do charge for the storage for any Amazon EBS volumes

## Scanning a Global Secondary Index

You can use the Scan operation to retrieve all of the data from a global secondary index. You must provide the base table name and the index name in the request. **With a Scan, DynamoDB reads all of the data in the index and returns it to the application.** You can also request that only some of the data be returned, and that the remaining data should be discarded. To do this, use the FilterExpression parameter of the Scan operation.

## Local vs Global Secondary Index:

* Local Secondary Index is needed when partition key is the same as primary. **Cant** be modified once created
* Global Secondary Index is needed when partition key is different from the primary. **Can** be modified once created

## S3 Log Object Key Format

Amazon S3 uses the following object key format for the log objects it uploads in the target bucket:

* **TargetPrefixYYYY-mm-DD-HH-MM-SS-UniqueString**
* In the key, YYYY, mm, DD, HH, MM, and SS are the digits of the year, month, day, hour, minute, and seconds (respectively) when the log file was delivered.
* A log file delivered at a specific time can contain records written at any point before that time. There is no way to know whether all log records for a certain time interval have been delivered or not.
* The UniqueString component of the key is there to prevent overwriting of files. It has no meaning, and log processing software should ignore it.

## Securing EC2 Instances

* Once you've launched an EC2 instance, you can connect to it just like any other remote server. To keep communications to your EC2 instances as secure as possible, you should use secure and encrypted protocols such as Secure Shell (SSH) to access your instance instead of Telnet, because Telnet transmits information in cleartext over the network.
* Additional security can be provided by using secure authentication methods. We recommend using public-key authentication instead of passwords to remotely log in to your instances with SSH. Passwords are vulnerable to a variety of simple attacks, including dictionary and brute-force attacks.
* If you used the Amazon Linux AMI to launch your instance, the password login feature is already disabled by default and you must use a public/private key pair to SSH into the instance. You can use the AWS Management Console to create the key pair or you can use a third-party tool like ssh-keygen (a tool provided with the standard OpenSSH installation) and import the public key to EC2.
* If you're using a third-party AMI or one you've created, the AMI may not have the password authentication option disabled by default. You should take a moment to verify and disable that option.

## Pre-Warming the Load Balancer

Amazon ELB is able to handle the vast majority of use cases for our customers without requiring "pre-warming" (configuring the load balancer to have the appropriate level of capacity based on expected traffic). In certain scenarios, such as when flash traffic is expected, or in the case where a load test cannot be configured to gradually increase traffic, we recommend that you contact us to have your load balancer "pre-warmed". We will then configure the load balancer to have the appropriate level of capacity based on the traffic that you expect. We will need to know the start and end dates of your tests or expected flash traffic, the expected request rate per second and the total size of the typical request/response that you will be testing.

# To be discussed:

* Cloudfront
* docker
* ecs
* vpc
* RAID
* synchronous vs asynchronus
* public datasets
* cloud hsm
* Amazon EC2 || Amazon S3 || Amazon VPC || Amazon Route 53 || Amazon RDS || Amazon SQS
* SSL
* ec2 iNSTANCE CONSOLE
* difference between detach and unmount
* rto - recovery time objective
* REST VS SOAP in s3
* ssh vs rdp - tbd
* https://poshsecurity.com/blog/2015/6/22/why-remoting-vs-ssh-vs-rdp-shouldnt-be-a-thing
* key namespace in s3
* tcp vs udp
* https://aws.amazon.com/certification/certification-prep/