**Transit Gateway:** acts as a hub to connect VPC’s and on prem networks. Can attach to TGW’S in other accounts, VPN’s. Direct Connect gateways and 3rd party appliances.

**Transit Peering**: Like VPC peering but you connect all VPC’s to one Transit gateway and do not have to worry about creating numerous VPC peering connections.

Can connect VPC’s to on prem networks. All traffic is encrypted. Build applications spanning multiple VPC’s. Deploy applications without updating large route tables.

Better for a large number of VPC’s, UP TO 5,000.

**NACL** – Subnet level, rules evaluated in order low to high, allow or explicitly deny traffic.

**Security group** – instance level, allow rules only – all traffic is denied by default. Can have more than one SG per EC2 instance

**IAM Best Practices:**

* Lock away your AWS account root user access keys
* Do not share access keys
* Rotate credentials
* Remove any unnecessary credentials
* Configure a strong password policy for users
* Use MFA

**Service Control Policies (SCP):**

**Limit** and **control** the maximum available permissions in the accounts of an AWS organisation.

**IAM Policies** – allow or deny a request for an action/ operation on resources

**Session Policy** – temporary credentials using AWS STS to access AWS resources. Allows users to assume a role and perform actions for a temporary period. A session policy is an inline policy.

**Nat Gateway** – allows an instance in a private subnet to connect to the internet. Has a private and public IP address. Created in a public subnet – associated to the main route table.

Use a private route table to point instances in your private subnet to the NAT gateway and the NAT gateway will forward it out to the internet/internet gateway. New way to do things. Manged by AWS.

**Nat Instance** – is an instance (with a special AMI) that can be used to allow an instance in a private subnet to connect to the internet via the NAT instance ID. Add the Nat Instance ID to the private subnets route table. You have to manage the OS for your instance. Old way of doing things.

**Migration Services:**

**AWS Server Migration Service –** Migrate on prem servers to EC2 instances

**AWS Database Migration Service –** Migrate on prem database to RDS

**AWS DataSync –** Migrate on prem file servers to EFS

**AWS Migration Hub –** one unified console to observe migrations and the above migration tools.

**AWS KMS**

Create and manage encryption keys.

Customer Master Keys are key that can be used by the user to encrypt data in AWS. (Created in AWS or imported into AWS)

AWS Managed CMK are used by AWS to encrypt data. AWS manage the keys.

**AWS Config**

Evaluated configuration of your AWS account and resources against desired configurations.

**AWS WAF**

Web application firewall, create rules to filter web traffic based on IP addresses. Can attach to a load balancer which is attached to EC2 instances, or API gateway etc…

**AD Connector –** connect onsite AD to AD Connector via VPN or Direct Connect with your AD credentials and you can authenticate to services in AWS i.e., Amazon Workspaces,

**VPC Endpoints**: Keep communications within the AWS network.

**Gateway Endpoint:** VPC level – Used as a gateway for EC2 instances to connect to AWS services without being exposed to the internet. Only S3, DynamoDB.

**Interface Endpoint:** Used as a gateway for EC2 instances to connect to AWS services without being exposed to the internet. All other services.

**AWS Client VPN** – connect your client PC to an AWS VPC via VPN connection. Must use external VPN provider.

**AWS Site-to-site VPN** – on premise data centre connection to AWS and have a private network established. Managed IPSec VPN.

Uses a **Virtual Private Gateway (VGW)** deployed on the AWS side and a **customer gateway** on the on prem side.

**Direct Connect** – Physically Connect to AWS using a private network link. Better for bandwidth and latency. Very expensive.

Connect on prem site physically to local AWS Direct Connect Location (via customer cage). Customer cage is connected to AWS cage and AWS cage is connected to AWS via physical fibre connection – 1GB/s – 10GB/s.

**CloudFront** – data in a CloudFront origin (S3 or EC2) will be pushed (cached) out to an Edge Location. Edge locations are distributed all over the world.

Allows for better latency and higher bandwidth from you service to the end user. Good for web distributions.

**Lambda**

Serverless functions. Upload code without managing servers. Good for short running tasks.

Developer uploads code to Lambda to execute, a trigger such as an event from the Console, CLI, API or SDK causes the Lambda function to execute. Pay for execution time (15 mins max).

Example, uploading an object to an S3 bucket can trigger a Lambda function that grabs meta-data for the S3 object. You can use a blueprint function, or write one from scratch.

7 LANGUAGES: Python, Ruby, Java, Powershell, Go, Node.js , C#

**SQS – Simple Queue Service** (queuing system for messages sent between microservices and serverless applications)

Standard Queue – best effort ordering for messages, message is sent at least once, sometimes more than once

FIFO Queue – First in first out, messages come once, no duplicates

**Virtual NIC’s:**

ENI – basic adapter to assign IP addresses to an instance. Use when there are no high-performance requirements. Use with all instance types.

ENA – For enhanced network performance, higher bandwidth and latency. Only supported instance types.

EFA (Elastic Fabric Adapter) – for high performance computing (AI & ML). All instance types.

ALB – supports instances, IP addresses, containers, and Lambda targets. Uses HTTP protocols

NLB – Routes based on IP, high performance and low latency. TCP and UDP protocols

CLB – older load balancer.

Gateway Load Balancer – used for virtual network appliances

**Amazon EventBridge (CloudWatch events)**

When an AWS service performs an event, an EventBridge rule is triggered (based on configured rules) and the information is forwarded on to complete an action

E.g., EC2 termination event > EventBridge rule > target is SNS to send a notification

**AWS Elastic Beanstalk:** Deploy web servers

Upload your code and Beanstalk will handle the deployment, load balancing and autoscaling.

You retain full control over the resources powering your application (NOT SERVERLESS).

**API Gateway (serverless)**

Deploy and manage API’s with AWS. API’s act as a ‘front door’ for applications to access data and functionality from backend services. RESTful API’s are used with HTTP calls.

Cache calls to improve latency.

**ECS**

Run docker containers and microservices on either serverless (Fargate) or on EC2 instances. Scale and maintain the microservices independently.

ECS runs in tasks – which are running docker containers. Tasks are defined in a task definition.

Has its own container registry for docker images: **Amazon Elastic Container Registry** ECR

**EKS**

Implementation of Kubernetes on AWS. Can run on EC2 or Fargarte.

**Continuous Integration:**

Developers **pushing code** to a **repository** (GitHub, GitLab, CodeCommit), the code is then built and put through a **series** of **automated tests** (unit testing, validation testing, format testing) using tools such as **Jenkins, GitHub Actions, CodePipeline** etc.