Containers & ECS

ECS Concepts

- Provides a platform to run Docker containers.
- It removes the need to manually manage container hosts.
    - Container Host: A machine that runs a container (e.g., Docker or containerd).
        - containerd: Software that runs a container.
    - No need to:
        - Launch EC2 instances (container hosts)
        - Install Docker
        - Manage networking, scaling, and etc.
        - Instead, AWS manages all of this.
- There're two modes:
    - EC2 Mode: Containers are run on EC2 instance you manage.
    - Fargate Mode: AWS manages the container hosts and you just define containers.
- Clusters are where your containers are run.
    - Created and managed by ECS.
    - You provide:
        - A container image
        - Instructions on how to run it.
- Container images are stored in a container registry such as Docker Hub and ECR (Elastic Container Registry).
    - ECR is AWS's private registry, integrated with IAM and offers scalability and AWS-level security.
- Container Definition
    - Defines a single container.
    - Part of a task definition.
    - Task: A running instance of a task definition.
    - Pointer (to where the container is and its port).
    - Specifies ECS where the container image is stored and which port(s) to expose.
- Task Definition
    - Blueprint for how ECS should run containers.

    - A full application configuration.
    - Contains:
        - One or more container definitions
        - Resources (CPU, memory)
        - Network mode
        - Compatibility (EC2 or Fargate)
        - Task Role (IAM role)
    - Examples:
        - A task could run:
            - A single container (simple app)
            - Multiple containers (e.g., web app + database)
- Task Role
    - Each task can assume an IAM role
    - Role provides temporary credentials to the containers inside the task.
    - Containers use those credentials to access other AWS services.

ECS Tasks vs. Services

- ECS Task
    - A single run of your application based on a task definition.
    - Does not scale or self-recover
    - Good for demos, or dev testing.
- ECS Service
    - A long-running, scalable group of ECS tasks.
    - Resilient.
    - Deploys load balancers in front of tasks.
    - Monitors health and auto-restarts tasks.

ECS Cluster Mode

- EC2 Mode
    - Create a cluster using EC2 instances as container hosts.
    - You're responsible for provisioning, managing, and paying for the EC2 instances.
    - Architecture
        - EC2 instances act as container hosts.
        - ECS cluster is created inside your VPC.

        - Benefits from multiple AZs.
        - Instances are managed using an ASG.
            - ASG controls the number of instances.
            - Horizontal scaling.
    - Billed for the EC2 instances regardless of container usage.
    - EC2-based ECS clusters are not serverless.
        - Serverless: A cloud computing model where you don't provision and manage servers directly and pay for what you use.
    - Containers come from container registries.
    - ECS handles running tasks or services while you manage the underlying host capacity.
- Fargate Mode
    - Serverless.
        - AWS manages the container host infrastructure.
        - Only pay for the container resources used.
    - Architecture
        - Fargate infrastructure is shared among all users (like EC2 is on shared hardware).
    - Tasks and services are still deployed into your VPC.
        - Each task gets an ENI.
        - ENIs are assigned IP addresses inside your VPC.
    - Fargate tasks can be accessed from within the VPC.
        - Also from public, if configured.
- Choosing between EC2 and Fargate
    - EC2
        - When your org. already uses containers.
        - Price-sensitive; potentially lower cost with spot/reserved instances.
    - Fargate
        - Minimal operational overhead.
        - Small, burst or batch workloads.

Elastic Container Registry (ECR)

- Amazon ECR is a managed container image registry service provided by AWS.
    - Public registry for everyone.
    - Private registry for IAM-based access.

- ECR supports enhanced scanning for security, powered by AWS Inspector.
- ECR publishes real-time metrics into Amazon CloudWatch.
    - Authentication attempts
    - Push and pull requests
- All ECR API actions are logged in CloudTrail.
- ECR generates events that can be routed via Amazon EventBridge.
    - Enables event-driven workflows.

Kubernetes 101

- Kubernetes is an open-source system that automates deployment, scaling, and management of containerized applications.
- Cloud agnostic.
- Kubernetes Cluster: A collection of compute resources working together as a single unit.
    - Control Plane
        - Manages the cluster.
        - Handles scheduling, deployment, scaling, and health and management of applications.
        - Components
            - kube-apiserver: Frontend API for the cluster. Everything communicates with the control plane via this component.
            - etcd: A distributed key-value store used as the cluster's database. Stores all cluster state data.
            - kube-scheduler: Monitors pods without assigned nodes and assigns nodes based on resource needs, locality, affinity, etc.
                - Pod: The smallest deployable unit.
                    - Contains one or more containers.
                    - Shared networking and storage.
                    - Commonly, one container per pod.
                    - Ephemeral.
            - cloud-controller-manager: Enables integration with cloud providers.
            - kube-controller-manager: A bundle of controllers (controlling nodes, jobs, endpoints, etc.)
    - Nodes

        - Physical or virtual servers that run containerized applications.
        - containerd: A container runtime
        - The Kublet agent that interacts with the control plane.
        - The Kube Proxy that manages networking rules and traffic routing.
        - Componenets
            - containerd: Software that actually runs containers
            - kubelet: Agent that ensures containers run correctly on nodes. Talks to the control plane.
            - kube-proxy: Manages networking rules and allows communication with pods. Helps implement services.
- Summary
    - Cluster: A full Kubernetes deployment; includes control plane and nodes.
    - Node: A physical/virtual machine that runs containers via pods.
    - Pod: The smallest unit of deployment; contains one or more containers.
    - Service: Provides a stable, abstracted endpoint to access pods.
    - Job: A temporary workload that runs to completion.
    - Ingress: Allows external traffic to access services inside the cluster.
    - Ingress controller: Software that sets up actual access.

Concept	Role in Kubernetes
Cluster	Complete Kubernetes deployment
Control Plane	Orchestrates and manages everything
Node	Where containerized apps actually run
Pod	Basic execution unit – 1+ containers
kubelet	Talks to control plane
kube-proxy	Manages traffic routing
kube-apiserver	Entry point to the control plane
etcd	Stores all cluster state
Job	Runs tasks to completion
Service	Exposes pods using a stable endpoint
Ingress	External access path into the cluster
Persistent Volume	Provides long-term data storage

Elastic Kubernetes Service (EKS)

- AWS's managed implementation of Kubernetes.
- Control plane is fully managed by AWS, automatically scales, and runs across multiple AZs.
- EKS integrates with ECR, ELB, IAM, and VPC.
- You can choose how to run Kubernetes worker nodes:
    - Self-managed nodes
        - Provision EC2 instances manually
    - Managed node groups
        - Uses EC2, but AWS handles lifecycle management
        - AWS manages nodes
    - Fargate (serverless)
        - Similar to ECS Fargate
- Pods are ephemeral by default, but:
    - EKS integrates with AWS storage services for persistent volumes (EBS, EFS, FSx)
- Typical Layout
    - AWS-managed VPC (where the EKS control panel resides)
    - Customer VPC (where worker nodes run via EC2 or Fargate)
    - ENIs (injected into the customer VPC to allow communication between the control plane and worker nodes)