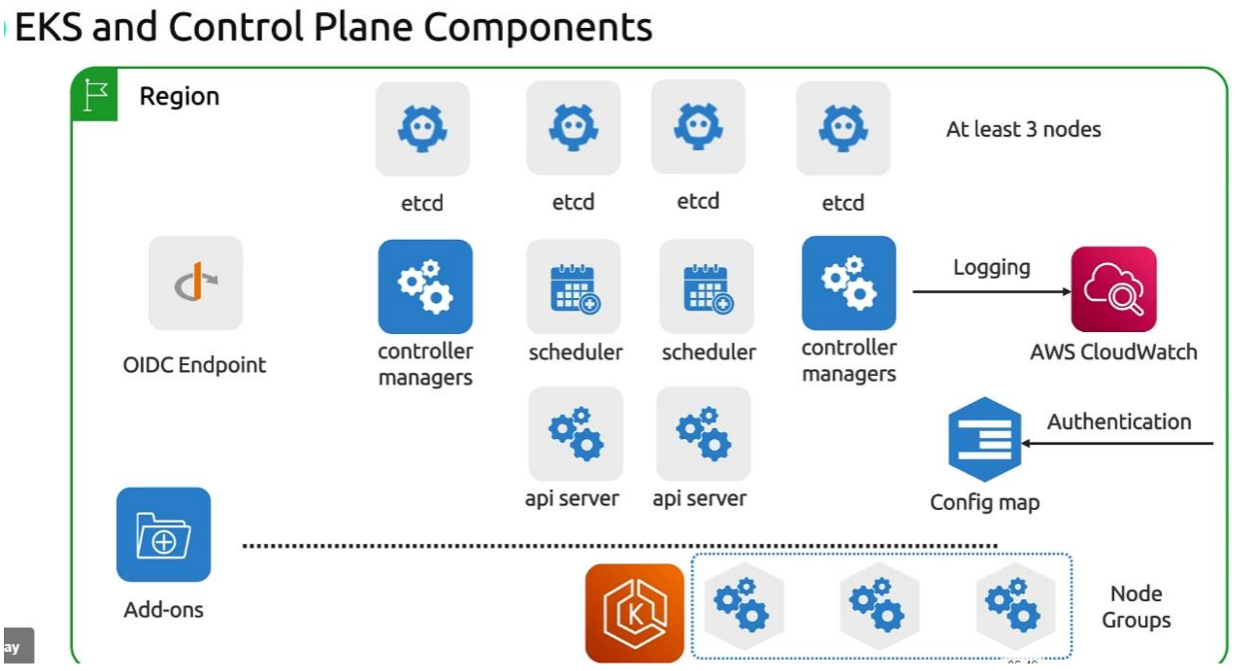
In Amazon EKS (Elastic Kubernetes Service), the control plane is fully managed by AWS and runs in an AWS account, meaning users don't directly manage or access the infrastructure behind it. The control plane includes components like etcd and the Kubernetes API servers. In contrast, the data plane, which consists of worker nodes (typically EC2 instances), runs in the user's AWS account within their VPC. These worker nodes communicate with the control plane over the network using ENIs (Elastic Network Interfaces), which bridge the communication between the control plane in AWS's account and the data plane in the user's VPC. While users interact with the EKS cluster through tools like kubectl and can manage the worker nodes, AWS abstracts and handles the management of the control plane, ensuring its availability and scalability. Networking between the control plane and data plane is secured and handled through cross-account access, but the user retains control over the networking rules and configurations within their own VPC.

Common Use Cases
EC2 — Use some EC2 instances as the control plane and some as the data plane.
KOPS (Kubernetes operations), **Kubespray** — Helps manage VMs as your own Kubernetes cluster.

In an EKS cluster, the user needs to keep the EKS cluster updated and ensure that the VPC and ENIs still exist. Make sure the necessary permissions are in place to allow AWS to manage certain resources in the user's account.

Users can integrate AWS services like RDS, S3, Load Balancers, Route 53, etc., with EKS.



In a distributed etcd setup, crucial for Kubernetes control planes, quorum is required for leader election to ensure high availability and data consistency. At least three etcd nodes are necessary, and this can scale up to five for better fault tolerance, with nodes typically distributed across different Availability Zones (AZs) within a single region. If one AZ goes down, the remaining etcd nodes in other AZs can continue leader election to maintain cluster functionality. **Amazon EKS (Elastic Kubernetes Service) is a regional service**, meaning all components, including the control plane and etcd, reside in the same region, though they can span across multiple AZs for high availability. The etcd component is managed by AWS as part of the control plane, so users don't need to manually configure it. Authentication for accessing the EKS cluster is handled through OIDC (OpenID Connect) endpoints and AWS IAM roles, allowing users to map roles for secure access. Node groups, which impact the data plane, determine where pods are scheduled and are managed via the EKS API. CloudWatch is typically used for logging and monitoring both control and data plane activities, and AWS's new authentication APIs define which IAM roles or principles have access to the cluster.

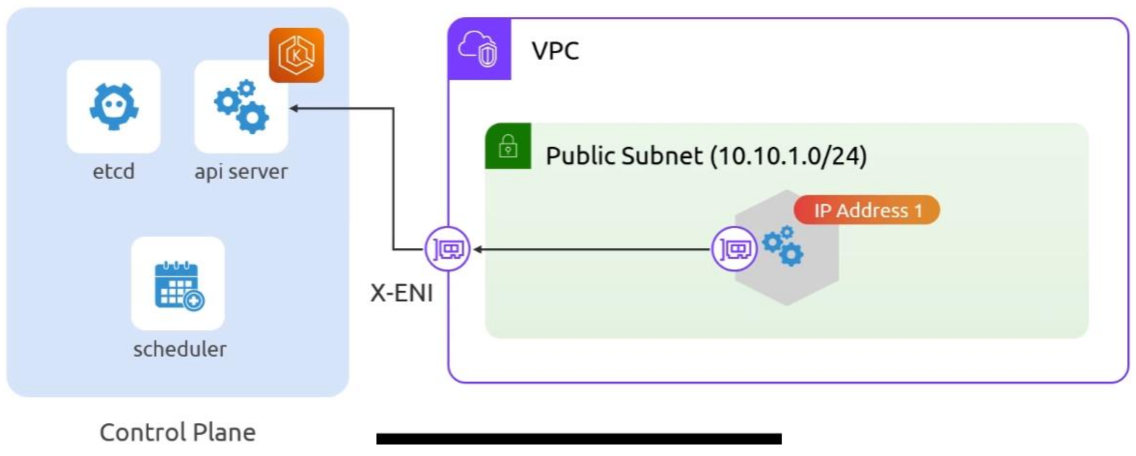
AWS **CloudFormation** is a service that enables infrastructure as code (IaC) to create and manage AWS resources through templates written in YAML or JSON. It can be generated using the **AWS CDK (Cloud Development Kit)**, which allows users to define their cloud infrastructure using familiar programming languages such as TypeScript, Python, or JavaScript. The CDK then synthesizes this code into CloudFormation templates, which are pushed to the CloudFormation API. In addition to CloudFormation, **Terraform** is another popular IaC tool that can be used to provision AWS infrastructure, including EKS clusters. **Amazon** provides **EKS Blueprints**, a set of Terraform modules that simplify EKS cluster creation. Other tools for managing Kubernetes clusters on AWS include **Pulumi**, which allows infrastructure creation with general-purpose programming languages, and **ClusterAPI**, which provides declarative APIs for cluster lifecycle management. Key components of an **EKS cluster** include **EC2 instances** for worker nodes, **VPCs**, **Security Groups**, **Load Balancers**, and **IAM roles** to manage permissions.

Tools for EKS

eksctl,

eksdemo – temporary create eks cluster

AWS IAM Authenticator – a tool to use AWS IAM credentials to authenticate to EKS cluster



In AWS, each Kubernetes node in an EKS cluster must have an IP address to join the cluster, and the network interface (ENI) plays a key role in communication. The EKS control plane runs in Amazon services, outside the user's VPC, while the node communicates with the Kubernetes API server through an ENI in the user's VPC, known as a cross-account ENI (XENI). This XENI is assigned an IP from the user's VPC and handles traffic between the node and the API server. Typically, there is one Kubernetes cluster per VPC, and as the number of pods increases, it consumes more IP addresses. To mitigate this, users can choose a large CIDR block or use overlay network solutions like Calico or Cilium, though these can create bottlenecks due to IP address limitations. Each ENI can have multiple IP addresses, depending on the node size, with some nodes supporting up to 10 IP addresses per ENI. The number of ENIs and IP addresses a node can have is determined by the instance type, with instance size affecting the available throughput and IP capacity.

Authentication

OIDC is one way to determine how things are authenticated within the EKS cluster. Auth config resides inside the cluster and does this mapping around who has access inside the cluster and how those relate to AWS IAM credentials. EKS Auth can allow access into clusters by handling some of the permissions on the control plane side as part of the service.