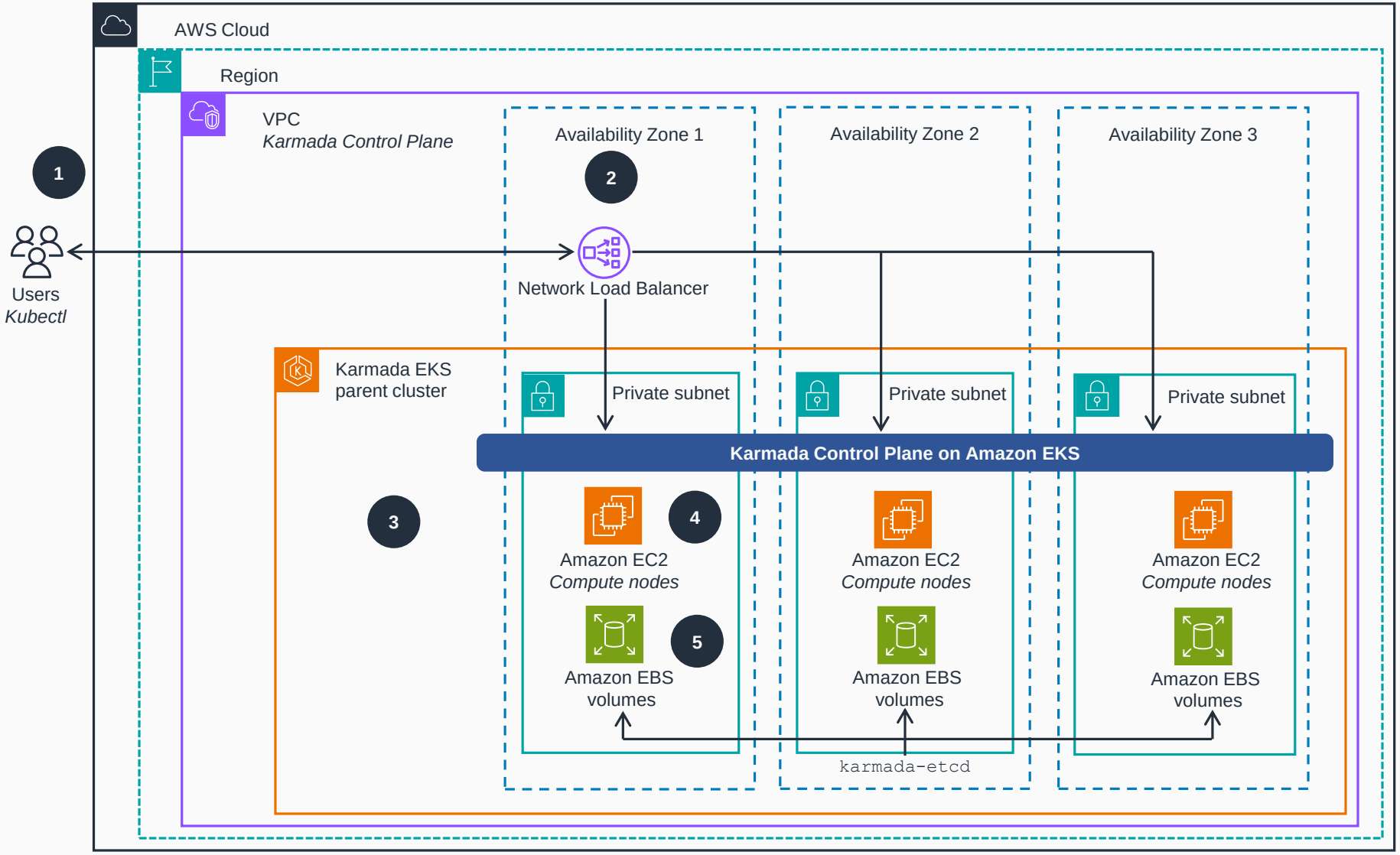
Guidance for Multi-Cluster Application Management with Karmada and Amazon EKS

This architecture diagram shows how to deploy a Karmada Control Plane on an Amazon Elastic Kubernetes Service (Amazon EKS) parent cluster. The next diagram shows how an application is deployed to the Karmada-managed EKS clusters.



1. The user interacts with the Aggregated Kubernetes API Endpoint (part of the Karmada Control Plane) using kubectl, a Kubernetes command line interface (CLI) tool, and a Network Load Balancer as the endpoint.

2. The Network Load Balancer provides SSL termination and acts as a proxy for Karmada services running on the **Amazon Elastic Kubernetes Service** (Amazon EKS) parent cluster.

3. The Karmada Control Plane exposes the Karmada API through its API server. In addition, it exposes the Kubernetes API, which receives calls for Kubernetes and Karmada management tasks.

4. Karmada runs several components on the **Amazon EKS** compute nodes. To keep records of API objects and states, the Karmada API server uses its own etcd database deployment.

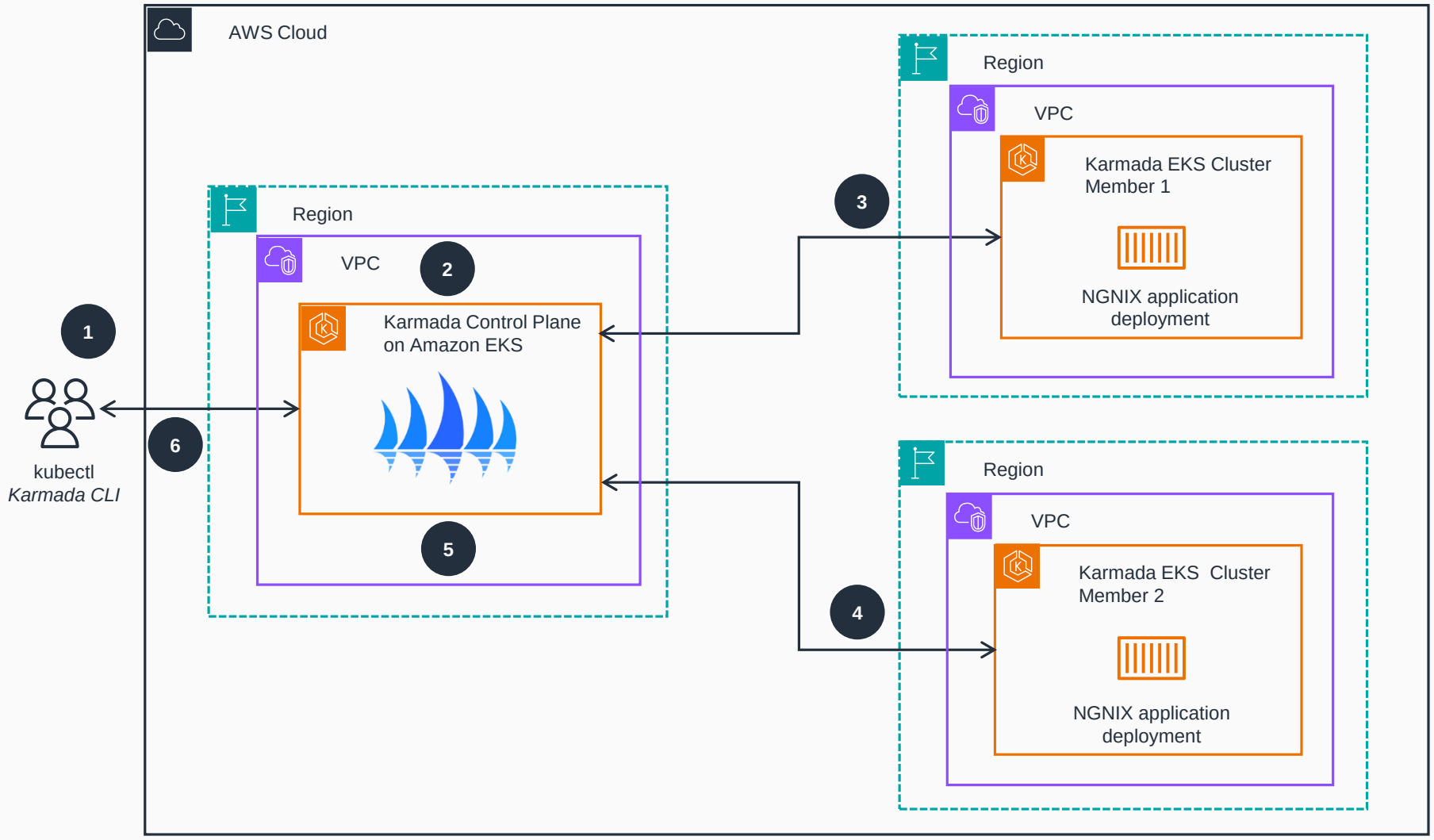
5. The Karmada *etcd* database uses Amazon EBS volumes attached to the **Amazon EKS** compute node and **Amazon Elastic Compute Cloud** (Amazon EC2) instances to maintain its state and consistency. All cluster state changes and updates are stored in persistent **Amazon Elastic Block Store** (Amazon EBS) volumes across all **Amazon EC2** compute nodes that host *etcd* pods.

Reviewed for technical accuracy June 10, 2024

AWS Reference Architecture

Guidance for Multi-Cluster Application Management with Karmada and Amazon EKS

This diagram shows how an application is deployed to the Karmada-managed EKS clusters.



1. The user interacts with the Karmada API server (part of the Karmada **Amazon EKS** Control Plane) using the *kubectl* CLI with the Karmada Installation of CLI Tools. The user sends commands directed at multiple clusters. For example, deploying an NGINX application with an equal load across two **Amazon EKS** clusters in different Regions.

2. The Karmada **Amazon EKS** Control Plane maintains the status and state of all **Amazon EKS** cluster members. Upon receiving the user request, it interprets the requirements and instructs member clusters accordingly. For example, it will deploy and run an NGINX deployment in each member cluster.

3. The Karmada **Amazon EKS** cluster Member 1 receives instructions from the Karmada Control Plane to deploy and run an NGINX container application.

4. The Karmada **Amazon EKS** cluster Member 2 receives instructions from the Karmada Control Plane to deploy and run an NGINX container application.

5. The Karmada **Amazon EKS** Control Plane (parent cluster) checks the application's deployment status on Member Clusters 1 and 2 and updates the state in its *etcd* database.

6. The user validates the status of the multi-cluster application deployment communicating with the Karmada **Amazon EKS** Control Plane through the Karmada CLI commands `*kubectl*.`

Reviewed for technical accuracy June 10, 2024

AWS Reference Architecture