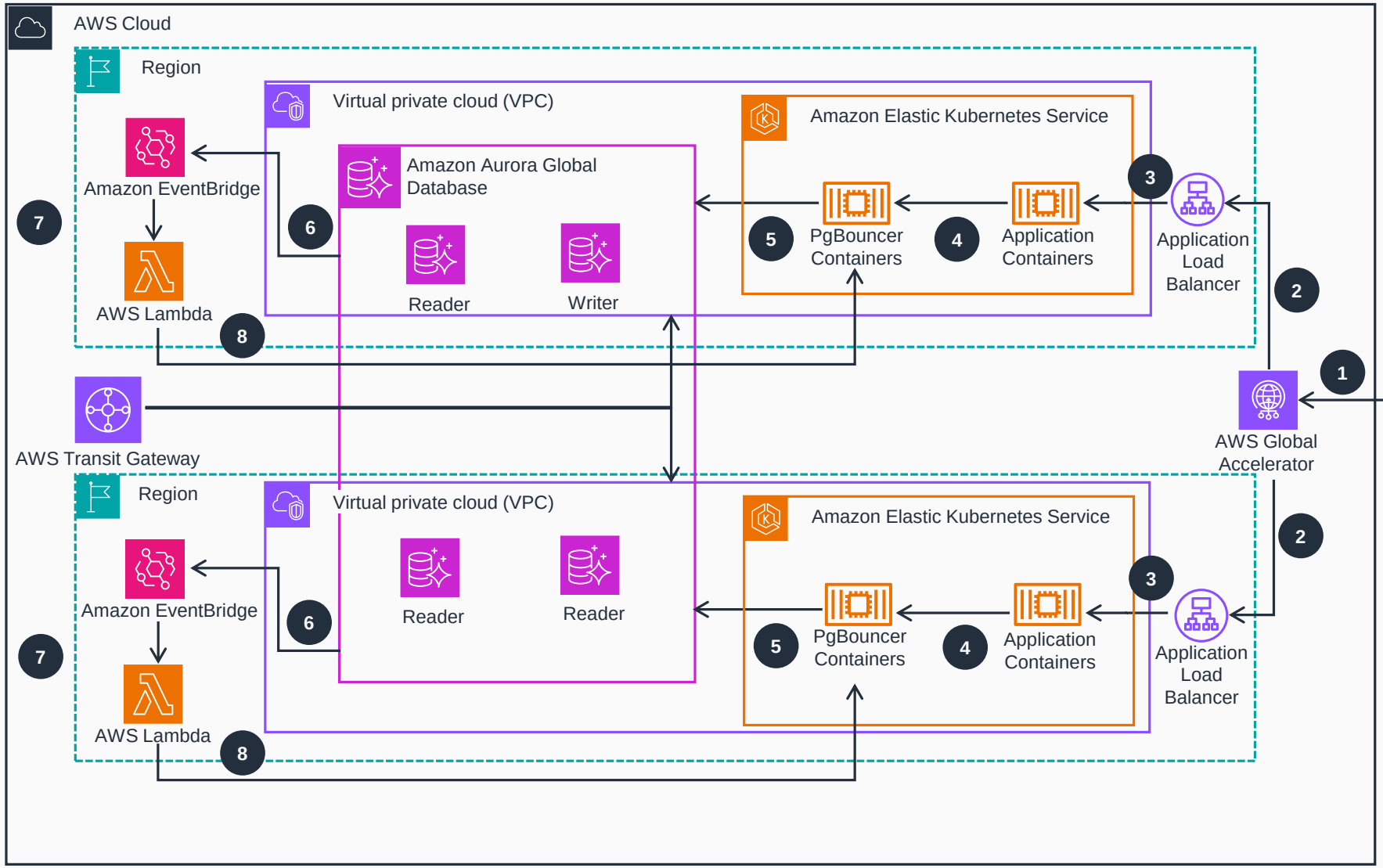
Guidance for Multi-Region Application Scaling Using Amazon Aurora

This architecture diagram shows how to scale applications globally using multi-Region Amazon Elastic Kubernetes Service (Amazon EKS) and Amazon Aurora Global Database.



1. Users connect to the application through **AWS Global Accelerator**, which sends application traffic through the AWS global network infrastructure.
2. **Global Accelerator** routes the connection to the nearest Region's Application Load Balancer.
3. The Application Load Balancer routes the connection to the application Pods on **Amazon Elastic Kubernetes Service (Amazon EKS)**.
4. Setup the PgBouncer Proxy Pods on the same **Amazon EKS** cluster to automatically scale using the Horizontal Pod Autoscaler.
5. Maintain a pool of connections to **Amazon Aurora Global Database** using the PgBouncer Proxy. Divide connections into *writer* and *reader* pools. The *writer* pool connects to the **Amazon Aurora** writer node in the primary Region. The *reader* pool connects to the **Aurora** reader nodes in the same Region as **Amazon EKS**.
6. Generate an event on an **Amazon EventBridge** event bus when **Aurora Global Database** switches over or fails over to the secondary Region.
7. Run an **AWS Lambda** function for the **Aurora Global Database** switchover and failover using an event rule.
8. Synchronize PgBouncer Proxy configuration for the *writer* node in the primary Region of **Aurora Global Database** using **Lambda**.

Reviewed for technical accuracy May 20, 2024

AWS Reference Architecture