AKKA

Self-Hosted Regions

Overview

Introduction

Akka is a platform for building and running applications designed to deliver guaranteed resilience and achieve up to 99.9999% availability. This level of availability is accomplished by running applications concurrently across multiple regions, with seamless, transparent data replication between regions to ensure consistency and fault tolerance.

Akka self-hosted regions can be deployed in your data centers or within major cloud providers, including AWS, GCP, and Azure. Self-hosted is a deployment model where Akka's application planes operate within your company's control plane ecosystem. Self-Hosted is ideal for companies that have an existing private or hybrid control plane, airgapped environments, far edge controllers with limited resources, or FedRAMP regulated environments.

Self-Hosted regions operate entirely within your Virtual Private Cloud (VPC/VNet), ensuring you maintain custodial ownership of the infrastructure and the environment where Akka applications are executed. Akka Self-Hosted regions are installed, monitored, and updated by your team with cooperation from the Akka team. Akka enters into a cooperation agreement with your organization to advise, instruct, and support you through the lifecycle of the Self-Hosted region's operation. This enables organizations to benefit from Akka's operational expertise while maintaining full control over their Akka environments.

1. **Cloud-agnostic deployment**
    a. Supports major cloud providers: AWS, Azure, Google Cloud Platform (GCP)
    b. Akka services can be deployed across multiple cloud providers
2. **Enhanced security and compliance**
    a. Akka application planes are participants within your organization's ecosystem of control and observability technologies.
    b. Data remains within the client's chosen environment, aiding compliance with regional regulations (e.g., GDPR, HIPAA).
    c. Clients maintain control over their security configurations while benefiting from our platform's built-in security features.
3. **Cost management**
    a. Allows clients to utilize existing cloud contracts and pricing models.
    b. Reduces the need for data transfer and egress fees often associated with external cloud-to-cloud interactions.

AKKA

Table of Contents

Terminology Used Throughout This Document

Term	Description
Akka Application	An application that is built using the Akka SDK. Applications contain APIs, workflows, streaming consumers, timers, and views for querying data. Applications are packed into Docker images and deployed as microservice instances within an Akka operating environment. Akka applications act as their own in-memory, durable database. They take responsibility for persisting their own state. Akka apps also “cluster from within”, creating a runtime cluster with other instances that handle balancing traffic, sharding data, and replicating their data to instances running within another region. Applications can be replicated between regions.
Akka SDK	Software Development Kit with support for programming components, a local debugging console, and a testkit for building, testing, and packing Akka applications.
Akka Application Plane	The runtime environment for hosting Akka applications within one or more regions. The Akka application plane provides compute, storage and I/O to execute Akka apps. It also provides automation to increase or decrease application instance capacity, observability for monitoring and debugging application behavior, and infrastructure management. The application plane is responsible for ensuring an Akka application meets its SLA by managing the Akka application and some of the underlying infrastructure. An Akka application plane depends upon: 1. A Kubernetes cluster. 2. A Postgres-compatible encrypted application data store. 3. A set of Akka operators for elasticity, observability, routing, and service management. 4. Proxies for traffic steering between regions. 5. Physical compute and storage for application execution. 6. An image repository that contains the packed applications available for deployment within that region.
Akka Control Plane	A global coordination point that is responsible for organizations, accounts, billing, and federating multiple Akka regions into a single substrate for which Akka applications can be deployed into. Akka applications that are deployed within a serverless or Bring Your Own Cloud (BYOC) model, leverage Akka’s global control plane that runs at Akka.io and is managed by Akka. Akka applications that are deployed with an Akka Self-Hosted application plane leverage your existing control plane (or equivalent DevOps services) that are assisted by Kubernetes operators and automation provided by Akka to your organization. An unavailable control plane will never impact running Akka applications.
Akka CLI	The Command Line Interface for developers, operators, and Infosec teams to interface with various Akka environments. The CLI provides utilities for deploying and controlling Akka applications. It also provides commands to configure observability, secrets management, service scaling, and account management.

Term	Description
Data persistence	Akka depends upon a durable event store for persisting data that is available in each Akka region, with all data encrypted at rest. The data is stored within a journal, adhering to Akka's event-sourcing architecture. This format is optimized for Akka's internal processing and cannot be directly queried.

Your Control Plane and the Akka Application Plane

Control Plane

The global Akka Control Plane serves as the central coordination system for managing organizational and operational aspects of Akka, such as account setup, role assignments, user access control, key rotation, and access token creation. It is managed by Akka and operates as a globally accessible service.

In the Self-Hosted model, an Akka application plane is installed within your environment and then integrated into your organization's ecosystem of control plane services. Your organization will provide user authentication and access, keys, certificates, certificate rotation, and access tokens.

Akka Application Plane

The application plane runs your workloads. It consists of one or more federated regions that run independently of the control plane and each other.

Every region within the application plane is responsible for pulling the application images, deploying it, scaling your application's instances within a region, and connecting the application to its peers running within other Akka regions.

The application plane contains a Kubernetes orchestration cluster, the persistence store the application stores its data in, and a local container registry where application images are cached and can be sourced from.

The application plane API consists of Kubernetes resources that enable remote governance and management of the application plane.

Responsibilities

In a Self-Hosted model, the responsibilities for system management and maintenance are defined in a cooperation agreement between Akka and your team, covering both initial provisioning and ongoing operational tasks.

AKKA

	You	Akka
Account	N/A. An Akka cloud account is not required.	N/A.
Configuration	Provide an understanding of your Kubernetes operating environment so that Akka can mimic the environment for quality assurance.	Advise on various Kubernetes configuration-related issues and to maintain a quality assurance environment to ensure compatibility with your environment.
Bootstrapping	Review default networking, compute, and persistence configuration of Akka with the Akka team. Infrastructure provisioning and validation. Infrastructure configuration. Create IAM roles for your human users. Integrate security services for audit logging.	Preparation review. Installation sequencing and support.
Maintenance	Updates to the control plane and application plane with security patches and upgrades. Regular (weekly) cadence of coordination meetings to discuss version updates, maintenance, and security.	Notifications and updates on best practices that Akka implements within our control plane with guidance for your organization to follow. Notifications and updates to new versions of Akka's CLI, application plane, and SDK. Regular (weekly) cadence of coordination meetings to discuss version updates, maintenance, and security. Daily standup and remote check-in.
Data Encryption	Establish secrets provided through a Key Management Store. Rotation of secrets and keys within the control plane and your application planes. Create a root certificate for your region group and intermediate certificates for your regions.	N/A
Monitoring	Export application metrics, logs and traces to your observability tools. Monitoring and availability of the Akka control plane and every application plane.	N/A
Backup and Recovery	Infrastructure and persistence backups.	

	You	Akka
Your Akka App Elasticity	Dynamically add / remove additional compute and persistence to accommodate real-time traffic against your performance SLA targets.	

Communications

Communication between your control plane and Akka application planes, as well as inter-region communication between the Akka application planes, is designed with security, reliability, and transparency in mind.

Command and Control

Command and control communication between the Control Plane and Application Plane is facilitated through APIs secured by TLS and token-based authentication.

The Akka application plane is configurable through its API of Kubernetes resources and exposes the Execution API for access to the application services it delivers. These APIs are secured using TLS certificates that will need to be provisioned by a globally trusted Certificate Authority (CA). All communication with these APIs requires bearer tokens passed in the HTTP Authorization header for authentication.

Inter-Region Communication Within the Application Plane

Applications running across multiple Akka regions communicate securely using mutual TLS (mTLS). Each client and service is issued unique certificates which are issued by your Key Management Store, and communication is only established if both sides authenticate each other using these certificates. If authentication fails, communication is blocked.

This transparent and automated mechanism relies on a multi-region ingress service embedded in each region. It ensures seamless communication and eliminates the need for manual configuration

Security and Compliance

Akka adheres to a wide range of compliance standards to ensure the security and integrity of our products. Comprehensive information on the company's compliance and certifications is available in the Akka trust center.

Data Protection and Encryption

All application data is stored within an encrypted, durable event store which cannot be externally queried. Application data at rest is encrypted using the store's default mechanism. Encryption keys can be provided through your Key Management Store.

Additionally, we offer a further layer of encryption for:

- Secrets that are synced to regional execution clusters from the control plane through the management-api.
- Secrets associated with authentication, such as TOTP secrets and OpenID refresh tokens.
- Secrets provided by the customer through environment variables or your KMS.

We provide regular and automated key rotations for data at rest, using Data Encryption Keys (DEK) and Key Encryption Keys (KEK).

DEKs encrypt data and are stored with it, encrypted by a KEK. KEKs are used to encrypt DEKs. KEKs are stored separately as Kubernetes secrets and can be rotated periodically.

When a KEK is rotated, all associated DEKs are re-encrypted without re-encrypting the underlying data, ensuring efficient and seamless re-encryption.

Monitoring and Management

Akka uses Groundcover (groundcover.com) internally for monitoring, observability, and alerting for infrastructure and core platform services. Customers must configure observability and monitoring by exporting application metrics, logs, and traces to their monitoring tools.

Your control plane must manage the underlying compute nodes and database sizing. Compute nodes should be elastic and can grow and shrink with the number of deployed services. The database storage should autoscale and the sizing of the Postgres database should be planned based upon the intended amount of requests per second and data operations that your region is expected to handle.

Logging and Audit Trails

Akka integrates with all security services, whether cloud-based or onprem.

Infrastructure Resource Provisioning

Infrastructure resources within Akka application planes will need to be managed by your control plane to ensure optimal performance and scalability of the application plane. The provisioning and scaling of compute nodes and database resources are detailed below.

Compute Nodes

Akka services are scaled horizontally between a minimum and maximum instance count based on CPU usage. This scaling activity determines the demand for compute nodes in the region.

The number of compute nodes can scale dynamically to meet the workload demands of the region, ensuring sufficient resources for service instances.

The maximum number of compute nodes is configured based on the expected workload and the number of service instances required for the region.

If resource utilization approaches this configured maximum, your team will need to increase the limit to ensure continued performance without disruption.

Database Resources

Database storage should be configured to automatically scale to accommodate increases in data volume without requiring manual intervention.

If additional database compute resources (e.g., CPU or memory) are needed beyond the initial configuration, you can resize the database to handle the increased workload.

This approach ensures the infrastructure remains flexible and responsive to workload changes while maintaining high availability.

Backups and Disaster Recovery

Kubernetes resources and databases should be backed up and are available for recovery. Akka provides data exporting capabilities that allow exporting of your data to external storage for additional processing or long term storage.

Support and Maintenance

The Akka team continuously applies security patches and software upgrades to our software. We provide notifications on when the CLI, SDK, and application planes have been updated with guidance on how to roll out updates to your own environments. You will be responsible for maintenance and infrastructure upgrades necessary for the platform to remain secure and performant. While there are always exceptions, we recommend that your environment does not fall more than 3 months behind the latest Akka release.

If any issues are identified, we have a customer support portal where issues can be created according to our Customer Support Policy. Issues are worked through based on severity and impact.