



AKKA AUTOMATED OPERATIONS

BYOK8s on Azure

Self-contained BYOK8s setup for Azure, with data flows.

DOC.AKKA.IO · AKKA

Run Akka Automated Operations (AAO) on an AKS cluster you provision and operate. This page is the complete BYOK8s-on-Azure path: the shared requirements first, then the Azure specifics.

Once the region is running, do not change any provisioned infrastructure without first informing Akka. It could break the Akka region.

Installation flow

1. **Understand:** review the [technical overview](#) and contact your Akka Success Team for a BYOK8s package.
2. **Capacity planning:** size the environment with your Akka Success Team.
3. **Infrastructure requirements:** agree on DNS subdomain zones, certificate provider, and related decisions.
4. **Infrastructure provisioning:** provision the network, cluster, and database, and set up the egress domain allowlist.
5. **Customer smoke tests:** run the checklist below before handing the cluster to Akka.
6. **Teleport:** install the Akka-provided Teleport Helm chart. Deploy it promptly: the join token has a 3-hour TTL.
7. **Installation:** Akka configures the cluster as an application-plane region and sets up platform observability. You seed database credentials into the namespaces Akka provides.
8. **Akka smoke tests:** Akka attaches the region to the Federation Plane and runs smoke tests.
9. **Region handover:** Akka assigns the region to your organization. A production-labeled region is not fully handed over until you complete the [region-readiness steps](#).

Infrastructure requirements

Requirement	Specification
Kubernetes	Version 1.34 at minimum. Cilium or Calico is mandatory for network-policy enforcement. Configure it as an overlay network only when the cloud-native CNI is unfeasible due to IPAM constraints.
Cluster CIDRs	<code>/17</code> for the pod network and <code>/17</code> for the service network.
Network	Minimum <code>/21</code> CIDR block from your IPAM allocation, subdivided into three database subnets (<code>/27</code> each) and three private subnets (<code>/24</code> each).

Requirement	Specification
Nodes	16 vCPU / 64 GB RAM instances (for example AWS <code>m5.4xlarge</code> , GCP <code>n2d-standard-16</code> , Azure <code>Standard_D16s_v6</code>). Minimum one node. Kubernetes autoscales the node count with demand.
Service Mesh	Akka deploys Linkerd, currently the only supported service mesh.
Load Balancer	Public Layer 4 by default. An internal load balancer is supported, and a subnet or IP can be specified on supported clouds.
DNS and Certificates	Two subdomain DNS zones, one for platform APIs and one for deployed services (for example <code>akka.acme.com</code> , <code>apps.akka.acme.com</code>), plus cert-manager <code>ClusterIssuer</code> objects for automatic certificate generation on both.
PostgreSQL	Version 17 or later, provisioned and configured by you, highly available across zones (RDS on AWS, CloudSQL on GCP, Azure Database for PostgreSQL flexible server).
CSI Driver	The cloud-specific secrets-store CSI (Container Storage Interface) driver provider.
Registry	Platform images can be pulled through your Artifactory, using Akka's container registry as a mirror.

Database credentials

Seed the connection credentials (username, password, host, database name) as Kubernetes Secrets into both the `kalix-system` and `kalix-management-system` namespaces. You create these namespaces if needed, but do not manage them: AAO imports them into its management scope and reconciles them. Akka recommends the [external-secrets operator](#) to mirror credentials from your cloud secret store.

Access

Within a customer-provisioned cluster, Akka uses least-privilege permissions scoped to the Akka-managed namespaces and limits its access to customer-owned Kubernetes resources.

Capacity planning

Database size is driven by total application data operations per second at peak load. Work with your Akka Success Team to determine sizing.

DB Size	Data ops/sec	CPU	Memory (GB)	Storage (GB)	Max Connections
XSmall	1,000	1	4	100	200
Small	3,000	2	16	200	500
Medium	6,000	4	32	400	1,000
Large	12,000	8	64	800	1,500
XLarge	24,000	16	128	1,600	2,000

Egress allowlist and Teleport

Akka reaches the cluster through Teleport, which uses ports 3023, 3024, and 3026 in addition to 443. Teleport is a certificate authority and identity-aware proxy; the non-standard ports segment traffic types, and the security boundary is mutual TLS rather than the port. The full egress hostname and IP allowlist (Teleport, console, Federation Plane, Control Tower, observability, and container registry endpoints) is on the [data flows](#) page.

Customer smoke-testing checklist

- Check connectivity between the database instance and the Kubernetes cluster.
- Ensure the `ClusterIssuer` is ready and in a good state, and verify all tokens it uses.
- In multi-region configurations, ensure traffic flows in both directions between regions.

Azure specifics

Apply these Azure additions to the requirements above.

- **Resource providers:** register the resource providers and enable EncryptionAtHost as described on the [BYOC on Azure page](#).
- **Database:** use Azure Database for PostgreSQL flexible server.
- **Secrets:** use the Azure secrets-store CSI driver provider.

See also

- [BYOC on Azure](#): the Azure resource inventory and identity model
- [BYOK8s production readiness](#)
- [Data flows](#)

This page documents the individual network flows between your environment and the Akka platform. Each flow lists the data carried, its source and destination, the connectivity used, and how it is authenticated. "Your ..." denotes a resource in your own environment.

Connectivity terms used below:

- **Peering:** a private network path between your environment and the Akka region.
- **Firewall, Federation Plane allowlisted:** a customer-initiated outbound flow to the Federation Plane over the internet, from an allowlisted egress path.
- **Akka NAT gateway:** a region-initiated egress flow. Akka can work with you to route these through your firewall instead.
- **Pre-authorized IP ranges:** Federation-Plane-to-region flows over the internet. A private-connectivity option is available.

Your end users

Flow	From	To	Connectivity	Auth
End user accessing an Akka service from outside your environment	End user	Akka Region	Public (through your firewall)	Your keys, secrets, and tokens
End user accessing an Akka service from inside your environment	End user	Akka Region	Private (peering)	Your keys, secrets, and tokens

Your deployed Akka services

Flow	From	To	Connectivity	Auth
Access to your private internal systems	Your service	Your APIs	Private (peering)	Your keys, secrets, and tokens
Pushing observability metrics for your services to your observability	Akka Region	Your observability	Private (peering)	Your keys, secrets, and tokens

Your operators, admins, and developers

Flow	From	To	Connectivity	Auth
Akka CLI or console authentication initiation	CLI or console	Federation Plane	Public (firewall, Federation Plane allowlisted)	Akka authentication
OpenID authentication	Console	Your IAM	Public	OpenID
Add a region to a project	CLI	Federation Plane	Public (firewall, Federation Plane allowlisted)	Akka authenticated, valid access token
Initiate multi-region failover	CLI	Akka Region	Private (peering)	Akka authenticated, valid access token
List services and projects	CLI	Akka Region	Private (peering)	Akka authenticated, valid access token
Create a project	CLI	Federation Plane	Public (firewall, Federation Plane allowlisted)	Akka authenticated, valid access token
Add a user, role, or permission	CLI	Federation Plane	Public (firewall, Federation Plane allowlisted)	Akka authenticated, valid access token
Inspect service components in a region	CLI	Akka Region	Private (peering)	Akka authenticated, valid access token
Deploy a service: initiate deployment	CI/CD or developer	Akka Region	Private (peering)	Akka authenticated, valid access token
Deploy a service: region pulls the image	Akka Region	Your image registry	Private (peering)	Akka authenticated, valid access token

Akka SRE

Flow	From	To	Connectivity	Auth
Troubleshoot a region (cloud console or kubectl)	Your VDI, when needed	Akka Region	Private (your remote desktop)	Your cloud credentials, granted through Teleport

Akka Region

Flow	From	To	Connectivity	Auth
Region observability metadata to the Federation Plane	Akka Region	Federation Plane	Public (Akka NAT gateway)	Kubernetes RBAC and Federation Plane observability token
Region requests the public key set	Akka Region	Federation Plane	Public (Akka NAT gateway)	Internal Akka JWT authentication
Certificate expiration monitoring (can be disabled)	Akka Region	Akka SRE	Public (Akka NAT gateway)	Kubernetes RBAC
Replication to another Akka Region	Akka Region	Akka Region	Private (peering)	Mutual TLS and ACLs

Akka Federation Plane

All flows below use the bootstrap credential you create and share with Akka.

Flow	From	To	Connectivity
Bootstrap a new Akka Region	Federation Plane	Akka Region	Public (pre-authorized IP ranges)
Install a new version of Akka into a region	Federation Plane	Akka Region	Public (pre-authorized IP ranges)

Flow	From	To	Connectivity
Maintain infrastructure (drift detection)	Federation Plane	Akka Region	Public (pre-authorized IP ranges)
Upgrade infrastructure (Kubernetes, database)	Federation Plane	Akka Region	Public (pre-authorized IP ranges)
Multi-region root CA management for cross-region mTLS	Federation Plane	Akka Region	Public (pre-authorized IP ranges)
Synchronize project, user, role, organization, and region metadata	Federation Plane	Akka Region	Public (pre-authorized IP ranges)

Private connectivity by cloud

Under BYOC, Akka can set up a private-connectivity option per cloud to keep region-to-internal traffic off the public internet: Transit Gateway on AWS, VPC Peering on GCP, and VNet Peering with Azure Firewall on Azure. See the [BYOC on AWS](#), [BYOC on Azure](#), and [BYOC on GCP](#) pages.