Akachain Documentation

Release latest

Aug 09, 2019

Contents

akaChain

For a long time, along with the growth of crypto currencies such as Bitcoin or Ethereum, Blockchain technology has rapidly gained attention of enterprises seeking on how to apply it to other domain beyond financial transactions such as distributed storage or Internet of Things. However, even when we have blockchain suitable use case, employing such technology is not a trivial task.

We have seen little success with either public or private blockchain on different domains. The public are raising many questions on the actual outcome of the blockchain technology after so much investment and research efforts. Scholars pointed out one of the many reasons for this slow adaptation:

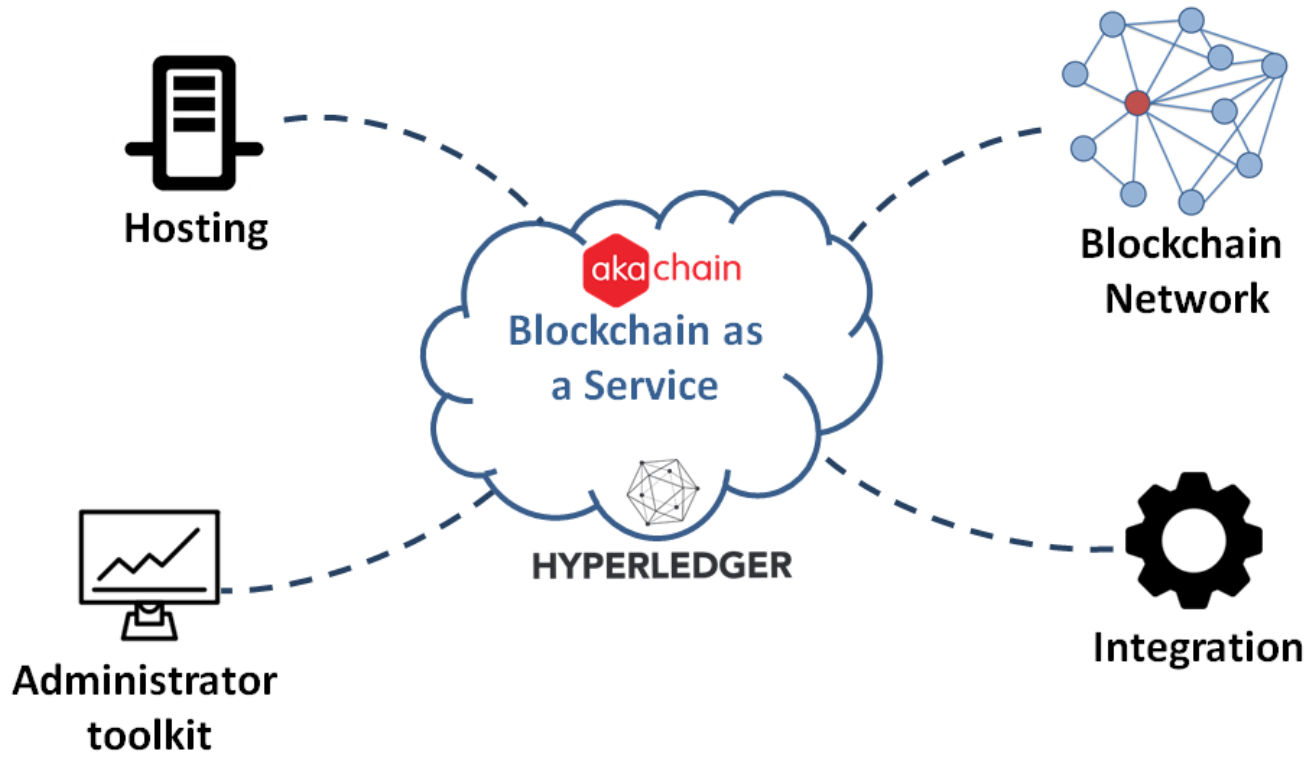
"Open platforms can't win by directly appealing to users on philosophical grounds, or even cost (see Linux on the desktop). Mainstream users have no good reason to directly interact with blockchain technology—or any piece of code—without intermediaries involved . . . ".

Indeed, we have to consider that many of the current blockchain technology platforms are too complex for mainstream users. Even to technology companies, successfully employing the blockchain technology impose great challenges both in term of economy as well as technical issues.

Inspired by such challenges, we introduce Akachain, a blockchain-as-a-service solution that brings the power of the blockchain to modern businesses. Akachain simplifies the development and managing a private blockchain system by providing automatic deployment solution on different cloud infrastructure, ready-made business application templates and professional support services.

CHAPTER 1

1. Blockchain as a Service



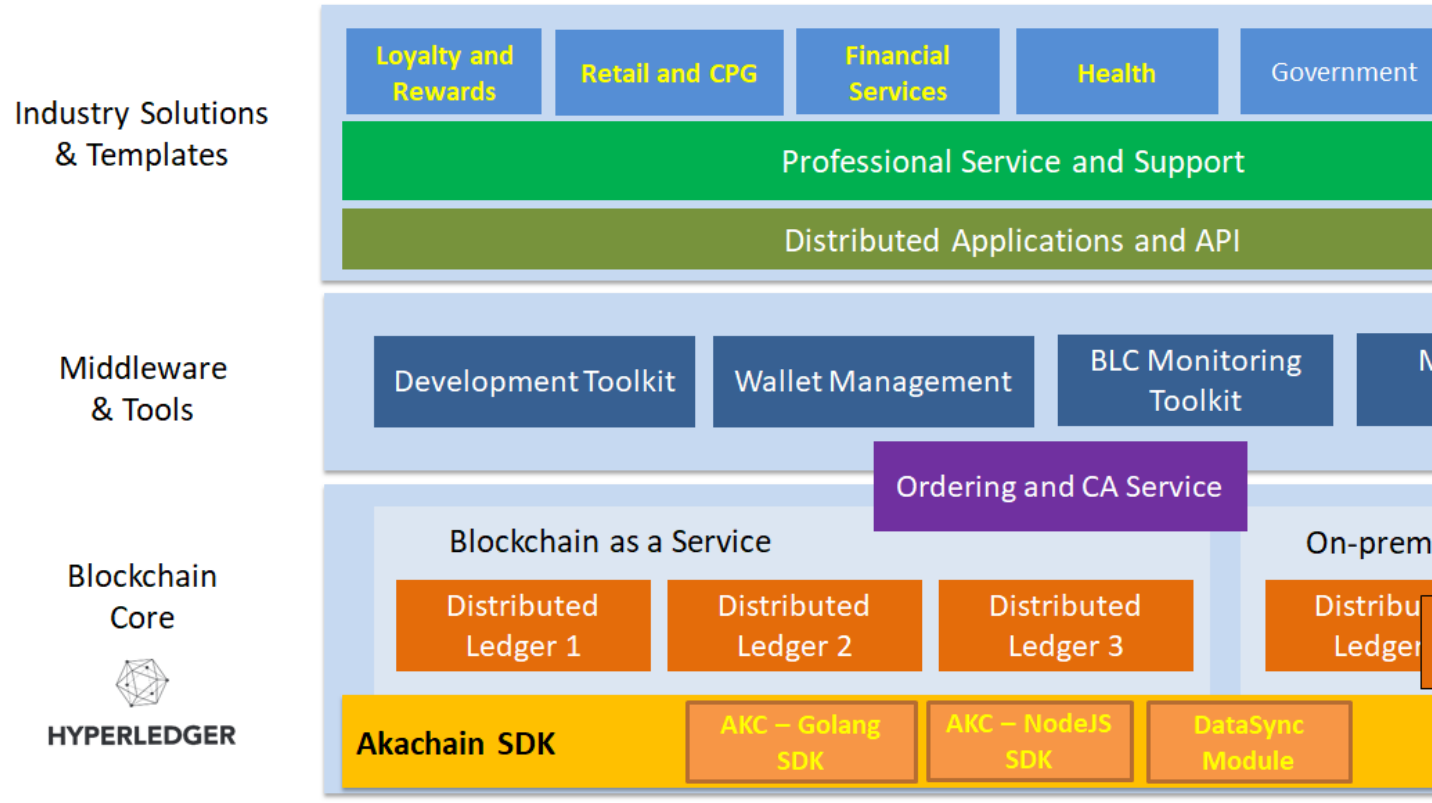
Akachain provides a Blockchain as a Service platform allowing developers quickly build and test their blockchain applications without worrying about the blockchain network layer. In particular, Akachain provides a number of core services:

1. Hosting: using the Akachain platform, a developer can quickly deploy a consortium blockchain network (Hyperledger Fabric) on AWS cloud environment. Other clouds support (Azure or on-premis cloud) is coming in late 2019.

2. Blockchain Network: Akachain provides a secure protocol allowing seperated private Fabric network to link together. Thus, it allows specific transactions in a private network can be verified by other private networks as they need.
3. Administrator Toolkit: The Akachain platforms offers a number of adminstration tools that help developers to automate their develop, deployment or monitor their blockchain network and applications.
4. Integration: Akachain provides a number of industry solutions and template services allowing businesses and enterprises quickly integrate blockchain applications with their existing systems.

CHAPTER 2

2. Platform landscape



The Akachain Plaform includes 3 core layers as depicted in the picture above.

2.1 2.1. Blockchain core

At the lowest layer, the Akachain platform runs on top of **Hyperledger Fabric**. Each business runs their own private Distributed Ledger, either hosted on Amazon Web Service or on-premise cloud.

Akachain provides centralized Ordering Service using Apache Kafka and Identity Service using RootCA to reduce the cost of running those individual ledgers. However, it is possible for businesses to run their own Ordering service on their network.

On top of the stock Hyperledger Fabric, Akachain provides several Software Development Kits (SDKs) that accelerate the process of applications development on Akachain:

AKC - Golang SDK: https://github.com/Akachain/akc-go-sdk, The Golang SDK offers a number libraries to help with writing Golang based chaincode on Hyperledger Fabric. Notables lirabries are: Advance Unit Test SDK, High Throughput SDK and High Secure SDK.

AKC - NodeJS SDK: https://github.com/Akachain/akc-node-sdk, The NodeJS SDK provides several libraries to assist writing Decentralized Applications (DApp) using NodeJS on Akachain.

AKC - DataSync Module: DApp template that periodically synchronizes Hyperledger Fabric transactions to an external PostgreSQL database to allow better data analytic functions.

2.2 2.2. Middleware and Tools

We provide a number of pre-built Middleware and tools support developers and network administrators to develop/deploy/monitor blockchain applications and private Fabric network.

Development Toolkit: Take your time and try our development tool. The tool allows developers to write and upload their chaincode directly to Akachain Test Network with 2 peers. It also automatically exposes REST APIs for DApp to call to those deployed chaincodes.

Wallet Management: Akachain accounts are linked to customer public keys. Currently this tool behaves like a normal cryptocurrency wallet. It allow users having Akachain account to access several public services such as Development Toolkit Portal and Network Monitoring tools for our test network.

Blockchain Monitoring Toolkit: Akachain provides a number of monitoring tools such as Network Explorer, Network Log Toolkit (Kibana), Prometheus, Graphana for critical components that work out of the box.

Network Management Toolkit: Akachain provides automated network management tools that support automatic application deployment on private network via CI/CD, automatic Hyperledger Fabric network deployment, chaincode installation/upgrade, etc.

2.3 2.3. Industry Solutions and Templates

Akachain provides a number of ready-made industrial solution templates from multiple domains: Loyalty and Rewards Network, Retail Solution, Financial Services, Health and Insurance and Manufacturing Traceability. They are template solutions that have already been developed and tested on our platform. Thus, they require only a small modification to be ready for production.

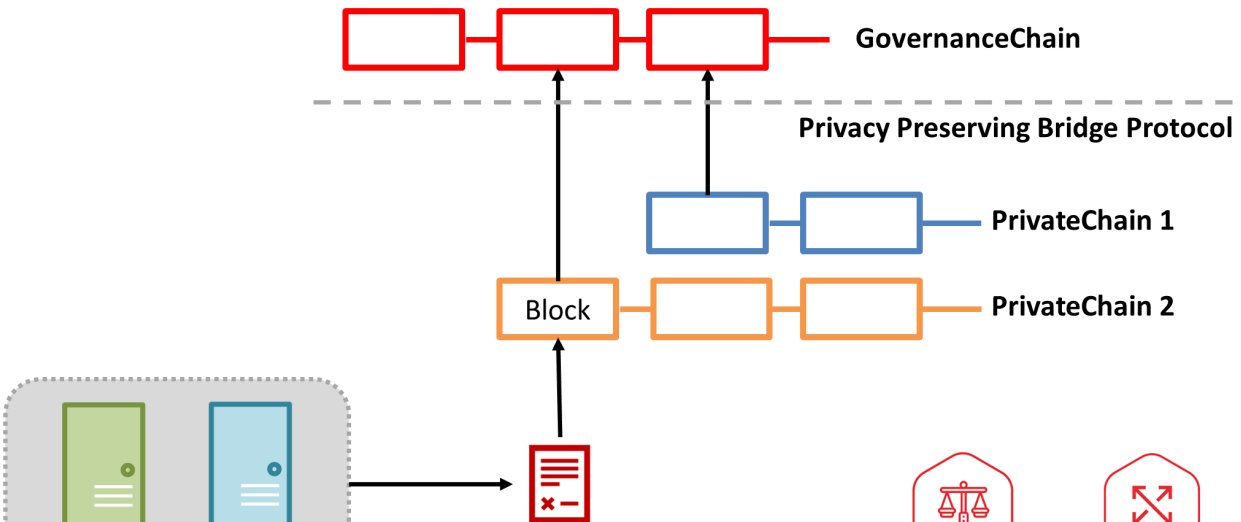
Among our industrial templates, we are proud to present one of the first blockchain-based Loyalty / Rewards Network & E-Payment solution in Southeast Asia: Utop (https://utop.vn/). Utop currently has around 6000 users in Vietnam and 4000 transactions per day. Gift redeemption and payment using Utop is widely accepted at all major merchants in Vietnam.

Note: To get more information about our industrial solutions and templates, please send us an email at: admin@akachain.io

CHAPTER 3

3. High Level Blockchain Architecture

Akachain provides a blockchain network that includes multiple separated consortium blockchains that can be linked together if they need.



On a high level, the Akachain network includes 3 main components:

1. Private Chain: Each consortium of peers forms a private blockchain network. This follows the normal Hyperledger Fabric model. Each private chain is a separated distributed ledger system where peers interact with each

others via private smart contracts.

2. Governance Chain: Akachain provides an intermediate Governance Chain that allows any **Endorsing Peer** from any PrivateChain to join. The Governance Chain is a public permissioned blockchain that stores transaction proof as well as encrypted meta data from other private chains.
3. Privacy Preserving Bridge Protocol: We develop a customized protocol to link private chains and Governance Chain together.

3.1 3.1. Private Chain

3.2 3.2. Governance Chain

3.3 3.3. Bridge Protocol

CHAPTER 4

4. Microservice Architecture

The Akachain (AKC) platform and its applications on top of AKC platform follow a microservices architecture rather than the traditional monolithic software stack. The main reason for this decision is due to the complexity of the system. There are many components that are implemented using different platform/programming languages with different network configurations and development life-cycles.

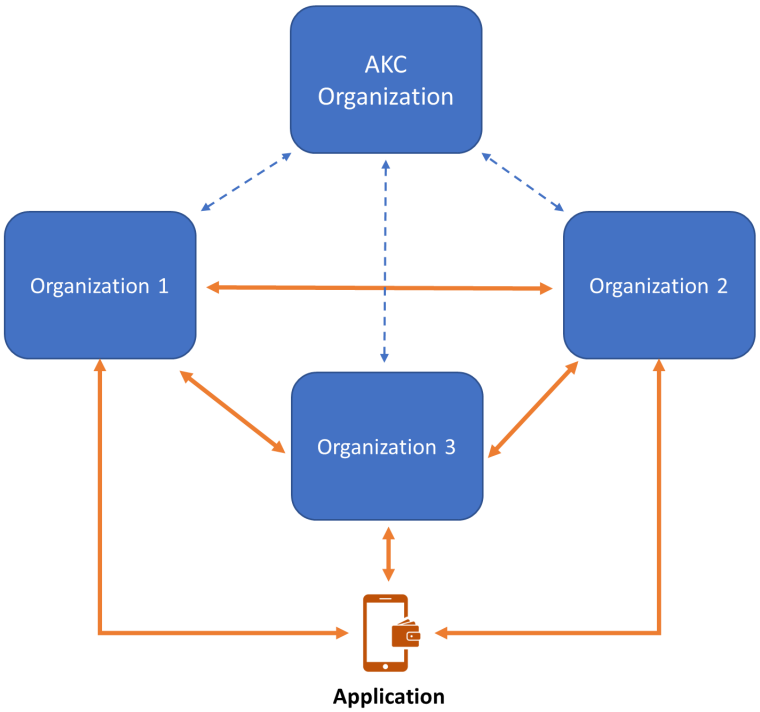
Each microservice is put inside a Docker container and is deployed entirely on AWS. However, it is straightforward to bring those to any other cloud providers or dedicated servers per customer request.

One of the main drawbacks of microservices in general is the complexity in managing all those containers. They are hard to keep track of (especially when we put auto scaling group and backup for each services) and debugging is a nightmare. Our current system in April 2019 runs ~ 100 microservices.

For this reason, we choose Kubernetes (K8S) to improve the complexity of the system. Kubernetes is a portable, extensible open-source platform for managing containerized workloads and services. It allows us to automate the deployment of containerized microservices. This makes it easier to manage all of the components and microservices of Akachain Platform. For AWS environment, we use the implementation of K8S on AWS (Amazon EKS). For on-premise system, we also support deployment on VMware PKS.

4.1 4.1. Enterprise Organization

In the Hyperledger Fabric universe, or more general, in a consortium blockchain network model, each consortium is composed by a group of **organizations**. An organization may have multiple **peers** that simultaneously send transactions created by the organization or receive/validate/endorse transactions from other organizations. The picture below describes the organization structure of a typical consortium blockchain system using Akachain platform.

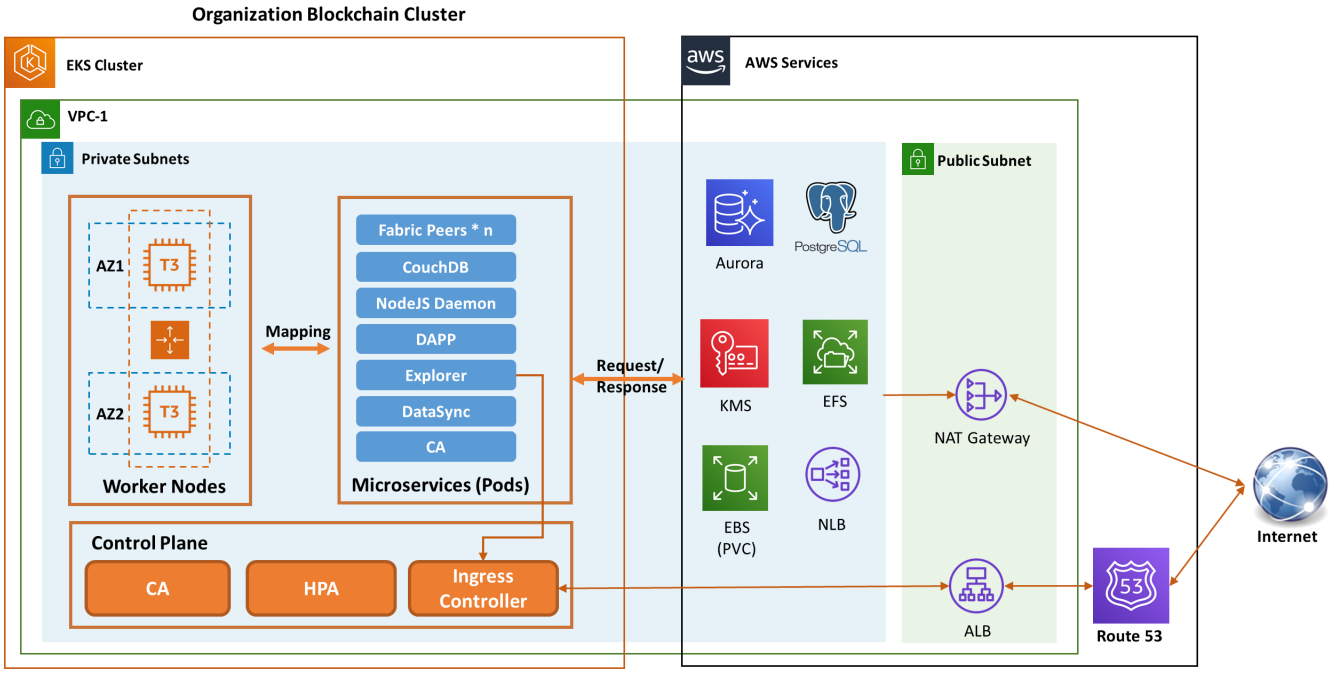


Following sections describe the high level microservice architecture of a typical organization in one of Akachain private consortium. Generally, we provide 2 K8S for each organization:

- Organization Blockchain Cluster: contains microservices that directly connect and interact to the blockchain
- Organization Application Cluster: contains microservices that handle client application requests.

We now describe a more detail about each cluster architecture using Amazon EKS. On-premise cluster architecture is nearly identical similar as VMware PKS provides very similar services

4.2 4.2. Organization Blockchain Cluster



Starting with the Blockchain Cluster, each cluster requires at least 7 microservices:

- Fabric Peer: Blockchain peer that handle interaction with the blockchain network. Currently, we suggest each organization runs 3 peers that load balanced to ensure fault-tolerance.
- CouchDB: Database instance storing blockchain ledger data
- NodeJS Daemon: Daemon process that perform various tasks in AKC-NodeJS-SDK to improve the system throughput (high throughput transaction)
- DAPP: Decentralized application that invoke Hyperledger Fabric SDK to interact with smart contracts and peers
- Explorer: Administration tool to monitor information in the blockchain network
- Data Sync: Periodically fetches or subscribes to push event from the Peer to synchronize ledger data to a Postgre SQL instance so that other services can quickly implement rich queries for reporting services.
- CA: Certificate Authority client or intermediate CA server.

Using EKS, those microservices are mapped to a number of worker nodes running T3.Large on multiple AZs. This guarantees the availability of the system.

Over the control plane of EKS, we configure the system to use Cluster Autoscaling (CA) and Horizontal Pod Autoscaling (HPA) to auto scale the system both in term of microservices as well as worker nodes.

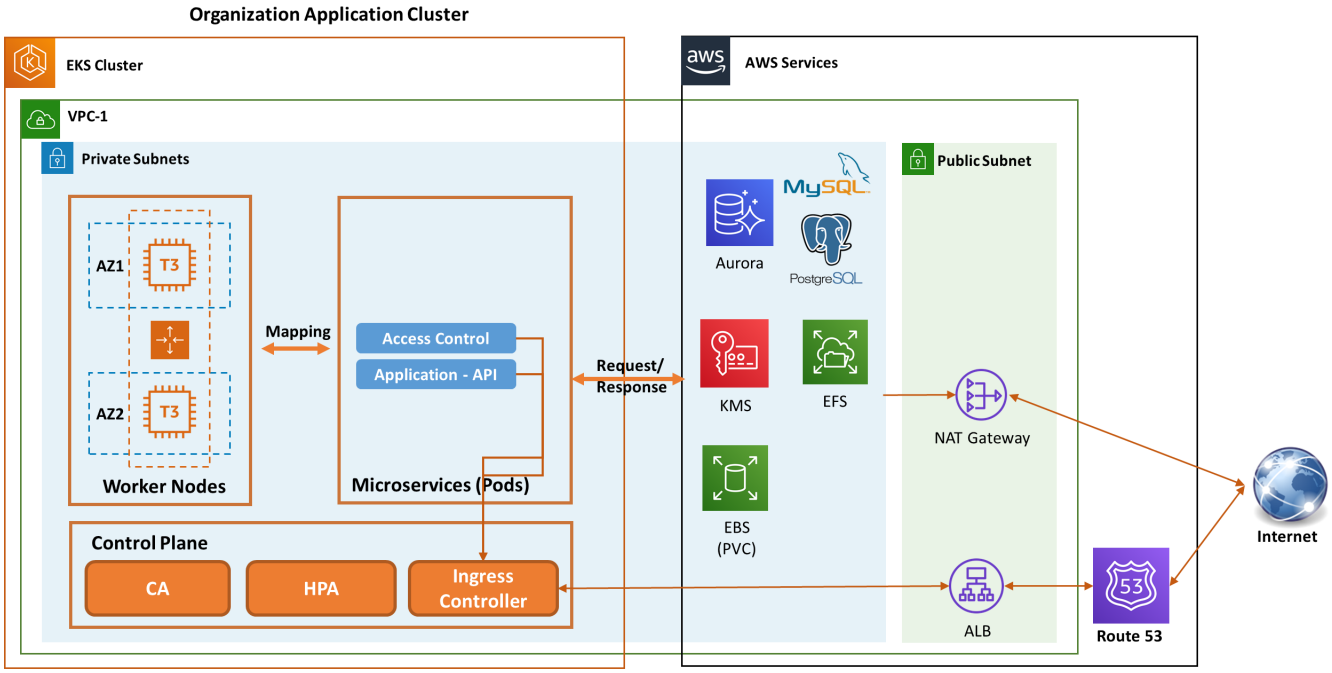
We also use a number of stock AWS services:

- Aurora: PostgreSQL instances that are managed by AWS
- KMS: Key Management Service that we use to store database and disk encryption key. We also plan to use KMS to store various data encryption key that can be used in the application layer later.

- EFS: We store various configuration files on EFS that can be shared among microservices.
- EBS: Microservices in pods managed by EKS can perform a Persistent Volume Claim. This action creates a number persistent instances of EBS and mount them to those pods. Later on, if those pods are replaced, the new pod will automatically mount these persistent EBS volume.
- NLB: For GRPC communication between Peers and the Ordering services over the internet.
- ALB, NAT Gateway, Route53: Allow some services to be accessible from the internet through the ingress controller module of EKS (in this case, the explorer web server)

From the networking point of view. Each cluster is mapped to one VPC. There are multiple private subnets and public subnets accross multiple AZ and only the one that needs to be expose to the internet go into the public subnet.

4.3 4.3. Organization Application Cluster



The Application Cluster follows the same architecture as the blockchain cluster. There is only a slight different in terms of microservices. Overall, we have 2 main group of microservices:

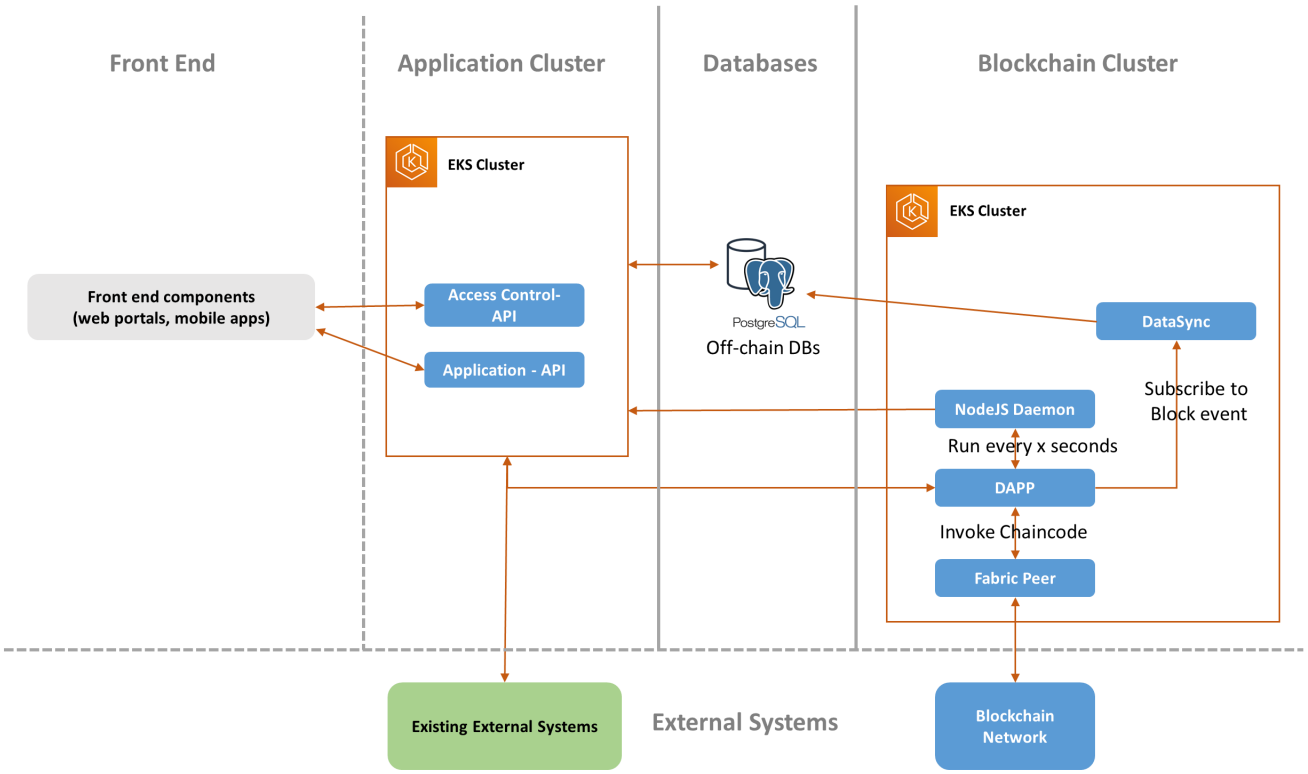
- Access Control : Microservice handles authentication and authorization for front end applications
- Application - API: Everything that serves front end applications

Inside each group, we can have multiple different microservices such as authentication, user management, token management, etc. They can be bundled together into one pod or splited up as we see fit.

A slight note here is that the Application Cluster uses the same VPC as the Blockchain Cluster. The main reasons is to allow some microservices to call each other without going through the NAT gateway as well as to share some AWS service together.

We also use MySQL in some microservices in the application cluster

4.4 4.4. Interaction between clusters



From here, we can present how those different clusters and microservices work together in a multi tier architecture

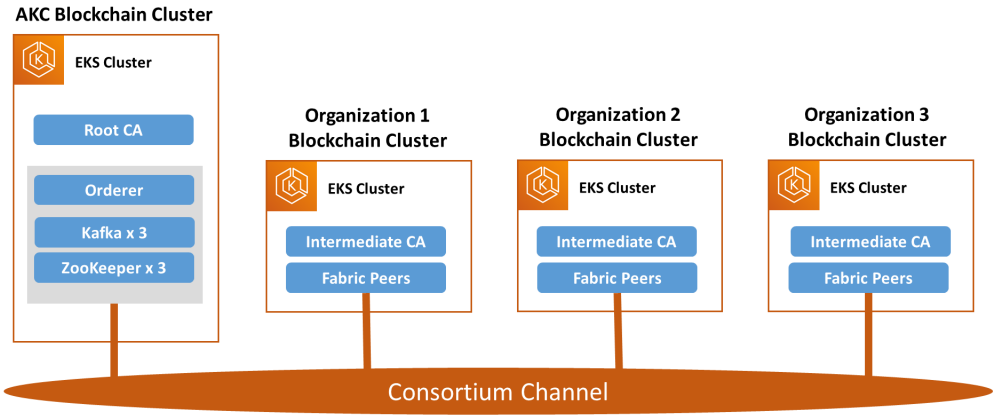
In the Front-end tier, we can have Web Portals and Mobile applications that are used directly by consumers.

The backend services for those applications resides in the Application cluster. They connect to 2 databases MySQL and PostgreSQL managed by Amazon Aurora. We also see here a link from the Application - API to potential external systems such as on premise existing systems either through the internet or a VPN connection.

The Application API also has a direct connection to the DAPP microservice in the Blockchain Cluster. The DAPP will relay request from the application cluster and create appropriate transaction through chaincode/smart contract deployed on Fabric peers to interact with the blockchain network.

The Data Sync service in the blockchain cluster subscribes to various blockchain events and pushes data to the PostgreSQL which later on is consumed by APIs in the Application Cluster. This way, we have a very reponsive cache that serves complex queries without worrying the peer is overloaded.

4.5. Interaction among organizations



By joinning a blockchain channel, all peers in the blockchain clusters can install Chaincodes and start communicating with each others.

Peers communicate directly with each other using a gossiping protocol. They also connect to the Ordering Service (Kafka + zoo keeper) managed by Akachain Blockchain Cluster to submit transactions and fetch confirmed blocks. This part follow the standard Hyperledger Fabric communication model.

Note: If you have questions that are not addressed by this documentation, please send us an email at: admin@akachain.io

---