Operator on ClickHouse Clusters Management Crossing AZs

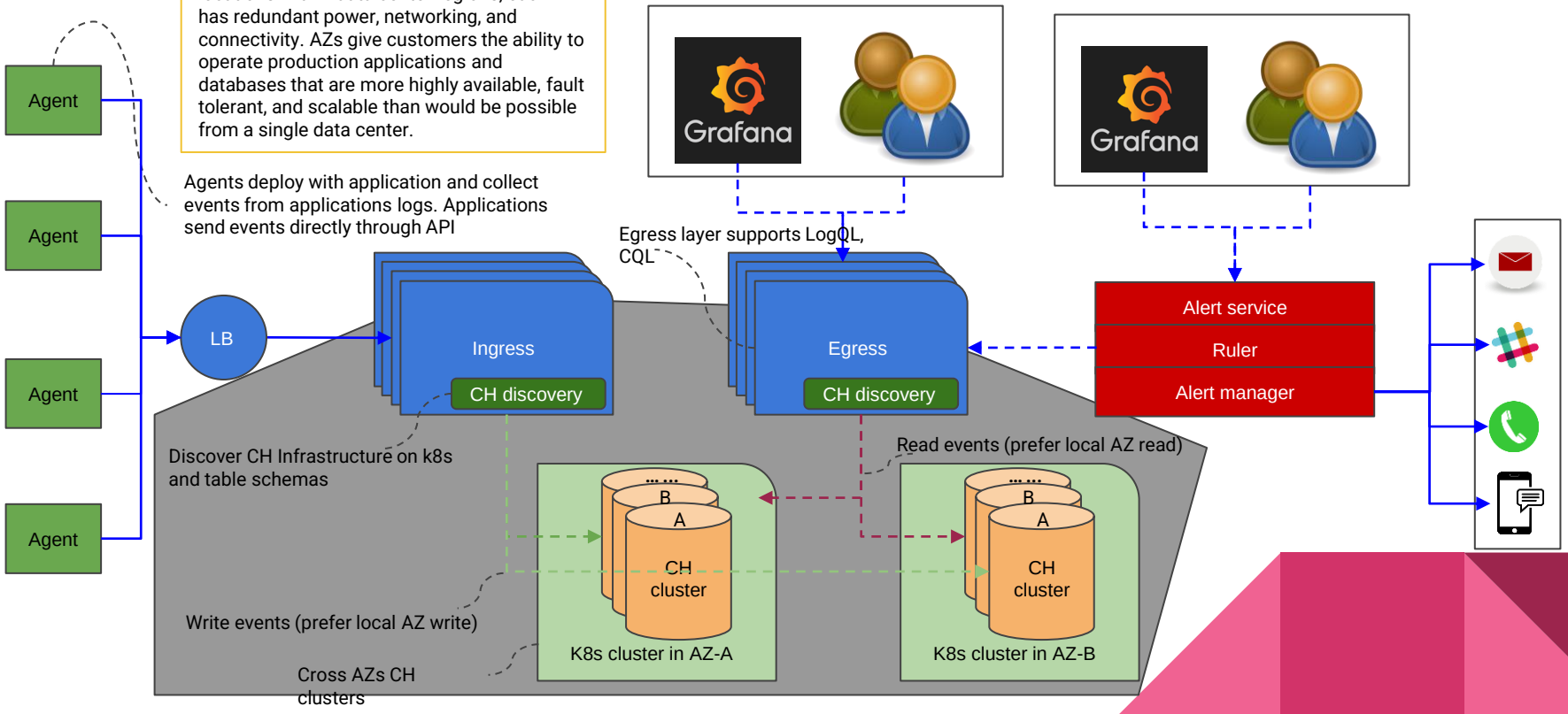
QiGang Zuo
qzuo@ebay.com
2021/06/26

Agenda

- Background
  - Events Monitoring Platform With ClickHouse on Kubernetes
- ClickHouse Clusters Management on Kubernetes via Operator
  - High Availability by Cross AZs
  - Scalable
  - Table Schemas Management and Data Redistribution
- Q&A

Background - Events monitoring Platform With Clickhouse on Kubernetes

Availability zones (AZs) are isolated locations within data center regions, each AZ has redundant power, networking, and connectivity. AZs give customers the ability to operate production applications and databases that are more highly available, fault tolerant, and scalable than would be possible from a single data center.

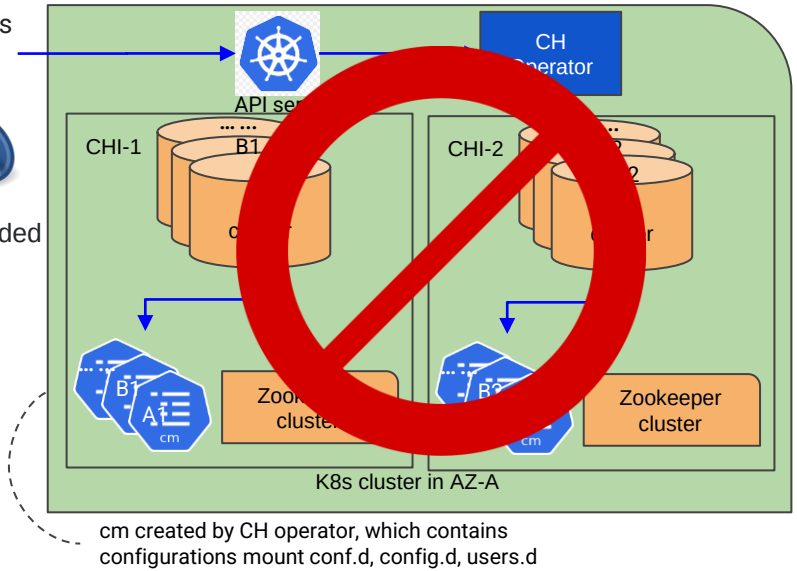


ClickHouse Clusters Management on Kubernetes via Operator

Operators are software extensions to Kubernetes that make use of custom resources to manage applications and their components. In another words, operators are clients of the Kubernetes API that act as controllers for a Custom Resource.

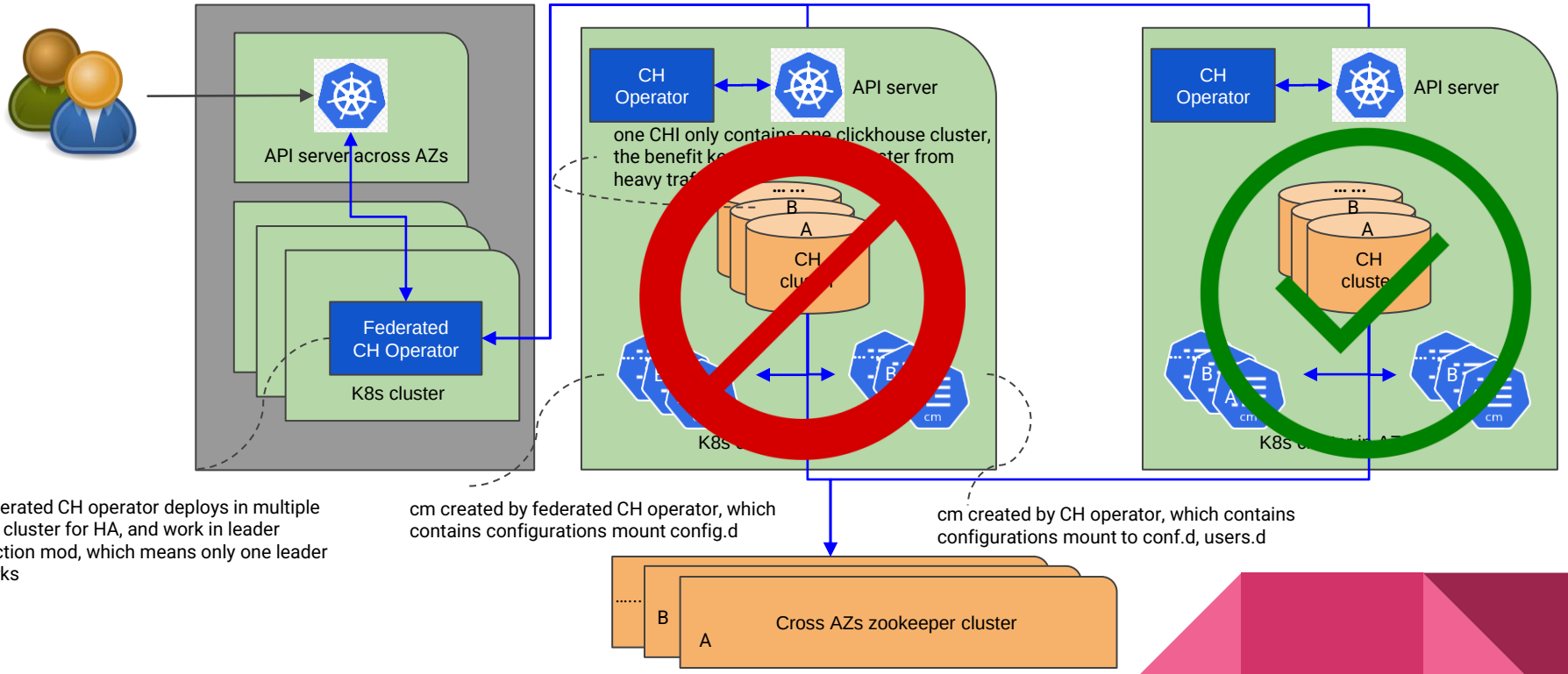
The ClickHouse Operator for Kubernetes currently provides the following:

- Creates ClickHouse clusters based on Custom Resource specification provided
- Customized storage provisioning (VolumeClaim templates)
- Customized pod templates
- Customized service templates for endpoints
- ClickHouse configuration and settings (including Zookeeper integration)
- Flexible templating
- ClickHouse cluster scaling including automatic schema propagation
- ClickHouse version upgrades
- Exporting ClickHouse metrics to Prometheus



https://github.com/Altinity/clickhouse-operator

ClickHouse Clusters Management on Kubernetes via Operator

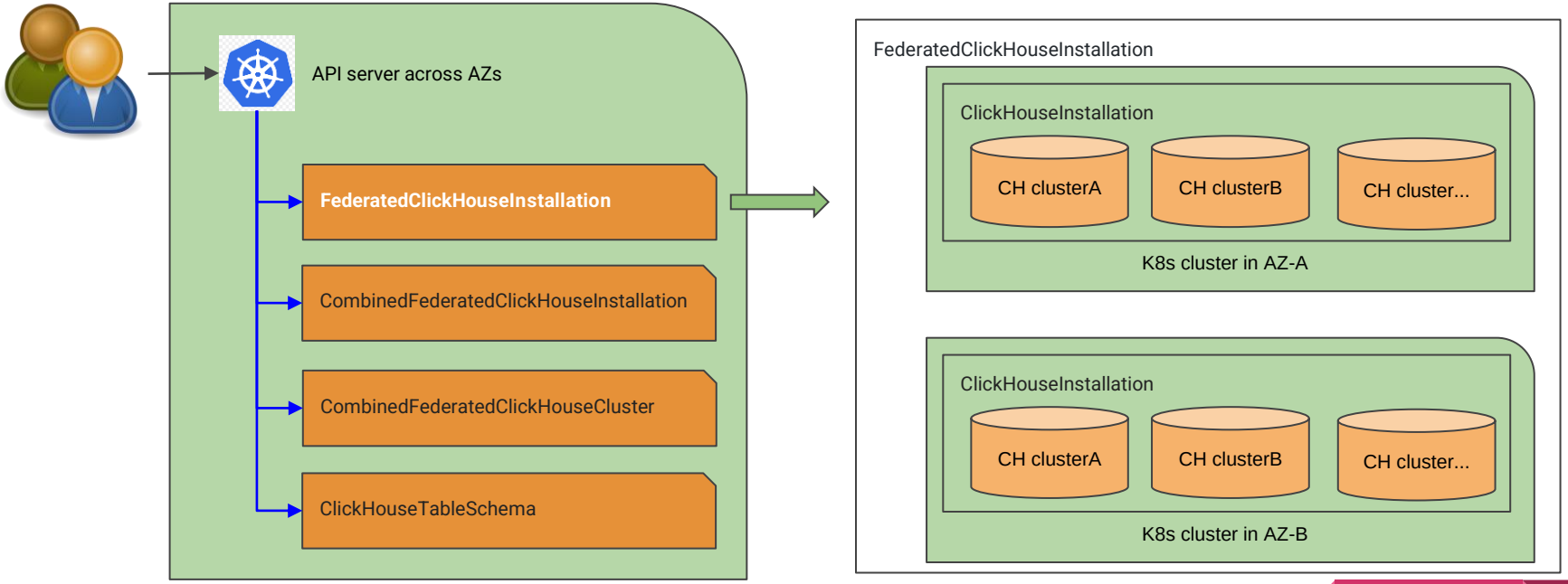


Federated CH operator deploys in multiple k8s cluster for HA, and work in leader election mod, which means only one leader works

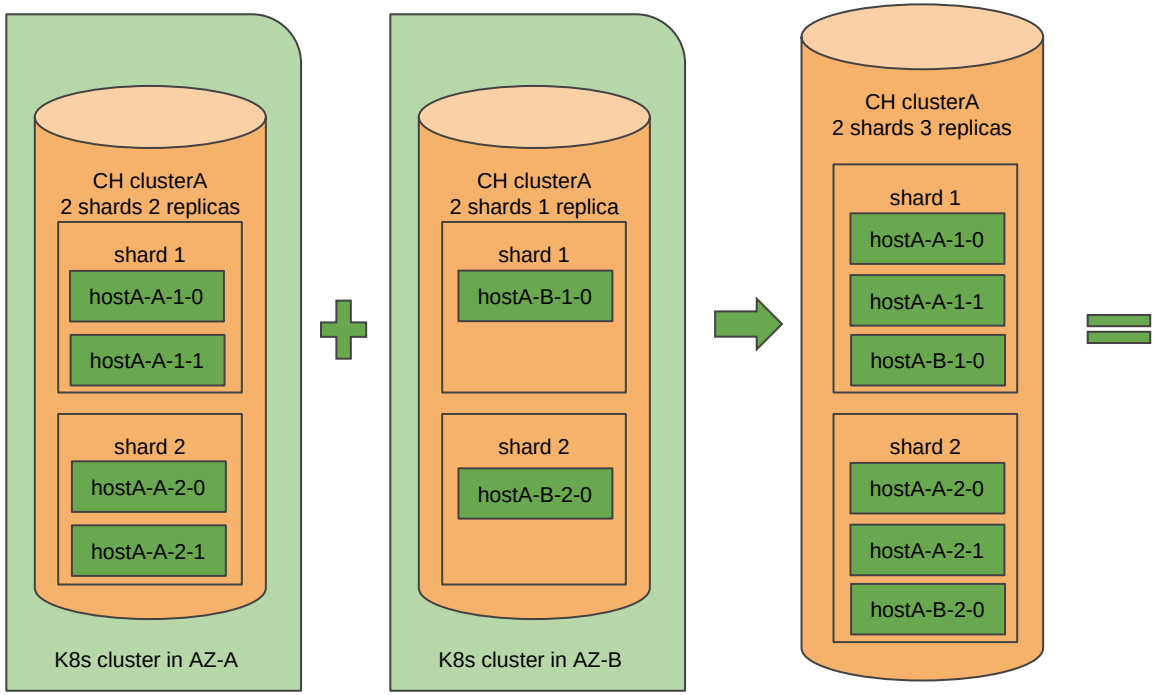
cm created by federated CH operator, which contains configurations mount config.d

cm created by CH operator, which contains configurations mount to conf.d, users.d

ClickHouse Clusters Management on Kubernetes via Operator - High Availability by Cross AZs

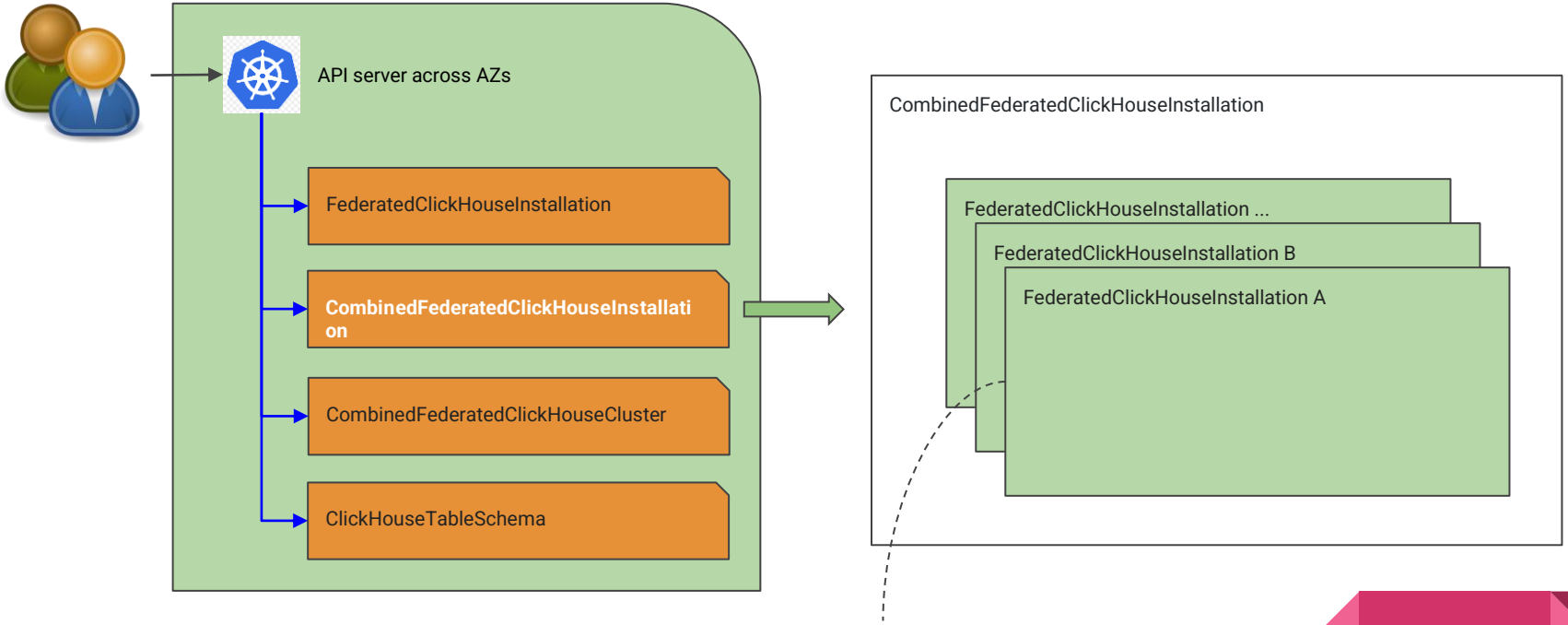


ClickHouse Clusters Management on Kubernetes via Operator - High Availability by Cross AZs



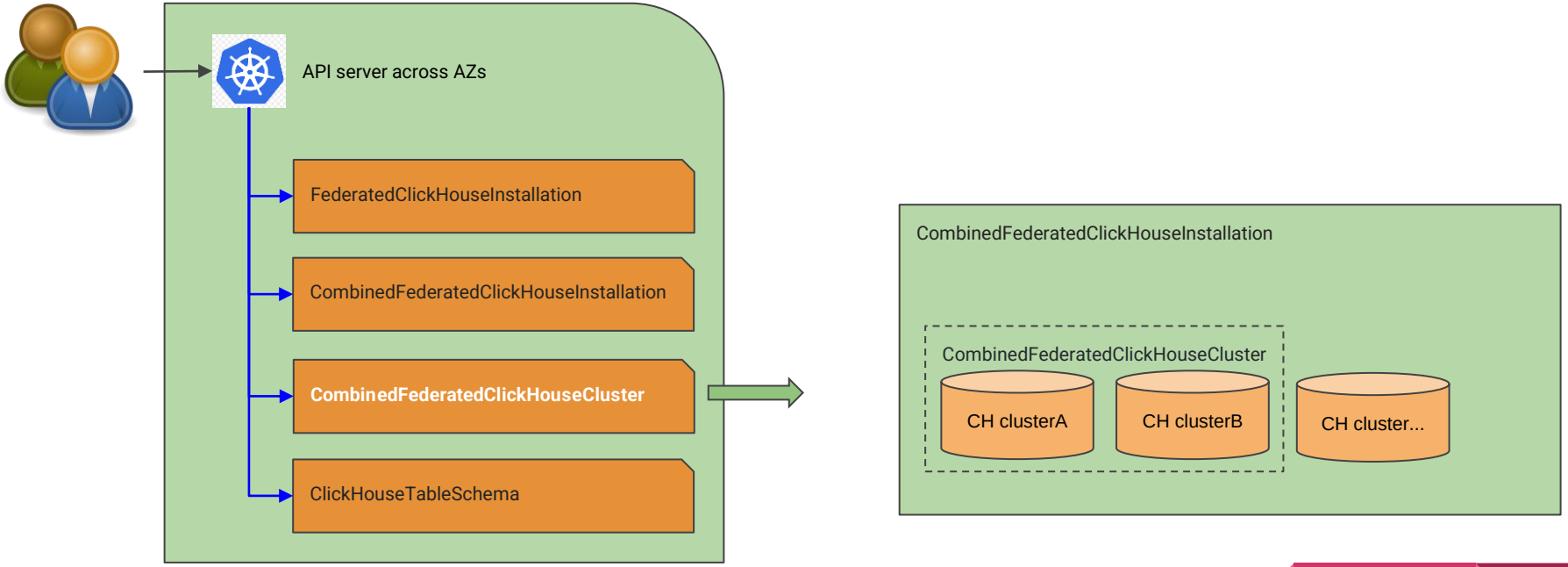

```
<yandex>
  <remote_servers>
    <CLUSTERA>
      <shard>
        <internal_replication>true</internal_replication>
        <replica>
          <host>hostA-A-1-0</host><port>9000</port>
        </replica>
        <replica>
          <host>hostA-A-1-1</host><port>9000</port>
        </replica>
        <replica>
          <host>hostA-B-1-0</host><port>9000</port>
        </replica>
      </shard>
      <shard>
        <internal_replication>true</internal_replication>
        <replica>
          <host>hostA-A-2-0</host><port>9000</port>
        </replica>
        <replica>
          <host>hostA-A-2-1</host><port>9000</port>
        </replica>
        <replica>
          <host>hostA-B-2-0</host><port>9000</port>
        </replica>
      </shard>
    </CLUSTERA>
    <CLUSTERB>...</CLUSTERB>
    <CLUSTER...>...</CLUSTER...>
  </remote_servers>
</yandex>
```

ClickHouse Clusters Management on Kubernetes via Operator - Scalable

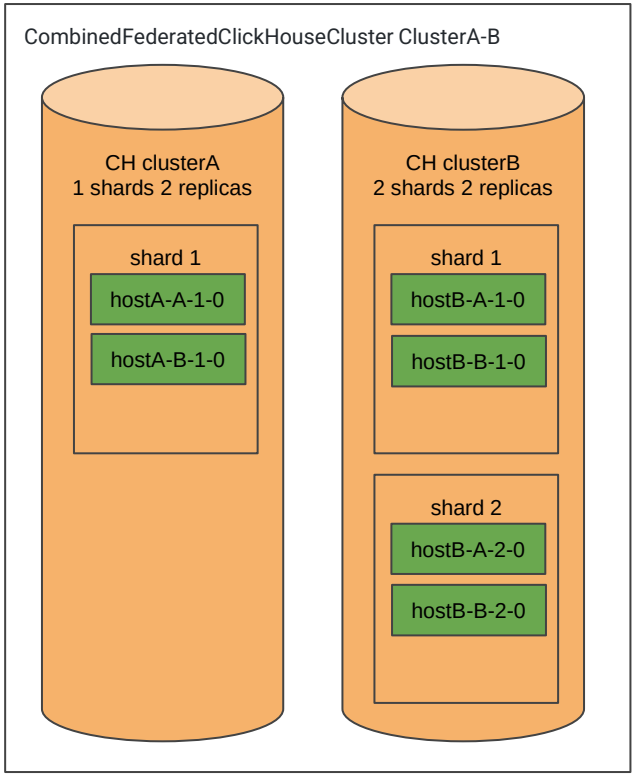


CH servers created by the same CombinedFederatedClickHouseInstallation share same remote_servers configuration

ClickHouse Clusters Management on Kubernetes via Operator - Scalable

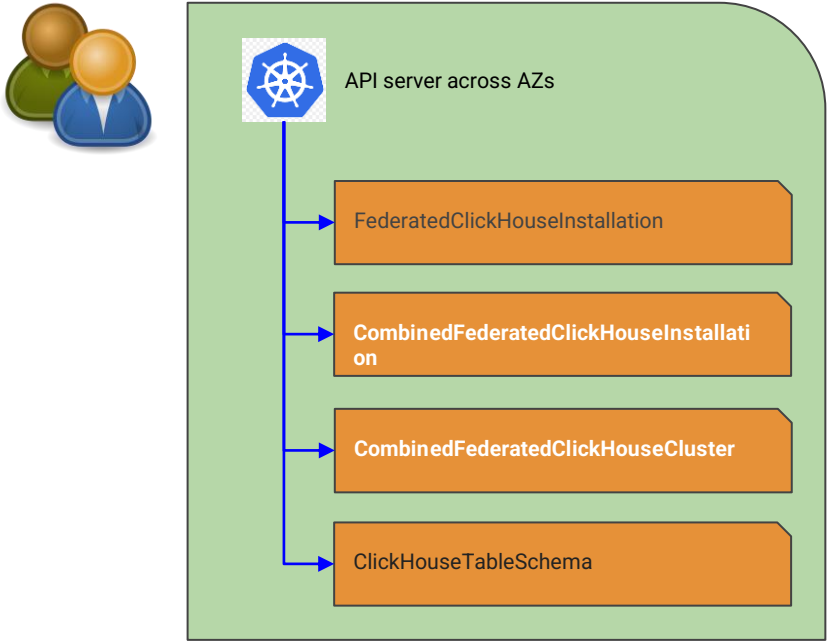


ClickHouse Clusters Management on Kubernetes via Operator - Scalable




```
<yandex>
  <remote_servers>
    <CLUSTERA>
      <shard>
        <internal_replication>true</internal_replication>
        <replica><host>hostA-A-1-0</host><port>9000</port></replica>
        <replica><host>hostA-B-1-0</host><port>9000</port></replica>
      </shard>
    </CLUSTERA>
    <CLUSTERB>
      <shard>
        <internal_replication>true</internal_replication>
        <replica><host>hostB-A-1-0</host><port>9000</port></replica>
        <replica><host>hostB-B-1-0</host><port>9000</port></replica>
      </shard>
      <shard>
        <internal_replication>true</internal_replication>
        <replica><host>hostB-A-2-0</host><port>9000</port></replica>
        <replica><host>hostB-B-2-0</host><port>9000</port></replica>
      </shard>
    </CLUSTERB>
    <CLUSTERA-B>
      <shard>
        <internal_replication>true</internal_replication>
        <replica><host>hostA-A-1-0</host><port>9000</port></replica>
        <replica><host>hostA-B-1-0</host><port>9000</port></replica>
      </shard>
      <shard>
        <internal_replication>true</internal_replication>
        <replica><host>hostB-A-1-0</host><port>9000</port></replica>
        <replica><host>hostB-B-1-0</host><port>9000</port></replica>
      </shard>
      <shard>
        <internal_replication>true</internal_replication>
        <replica><host>hostB-A-2-0</host><port>9000</port></replica>
        <replica><host>hostB-B-2-0</host><port>9000</port></replica>
      </shard>
    </CLUSTERA-B>
    <CLUSTER...>...</CLUSTER...>
  </remote_servers>
</yandex>
```

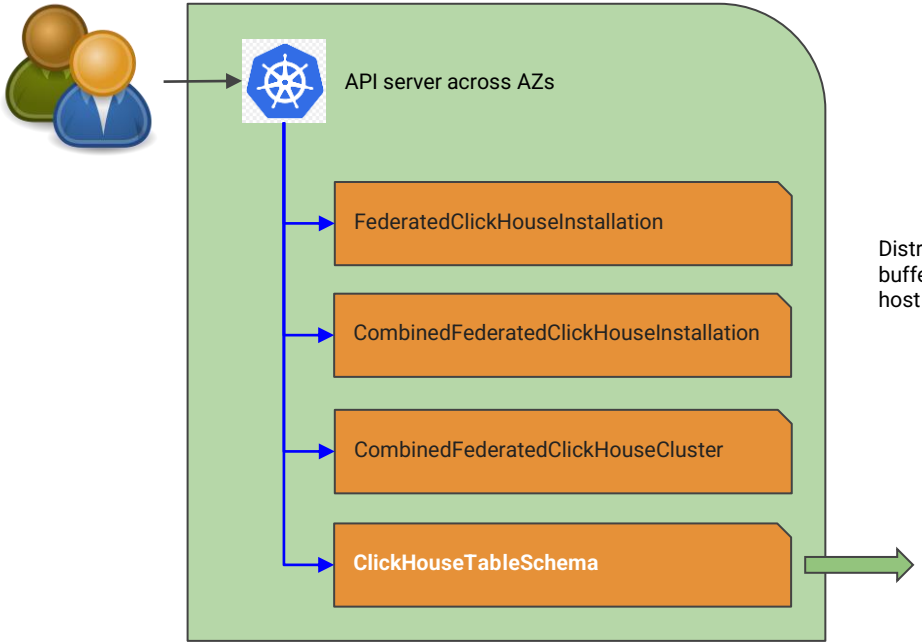
ClickHouse Clusters Management on Kubernetes via Operator - Scalable

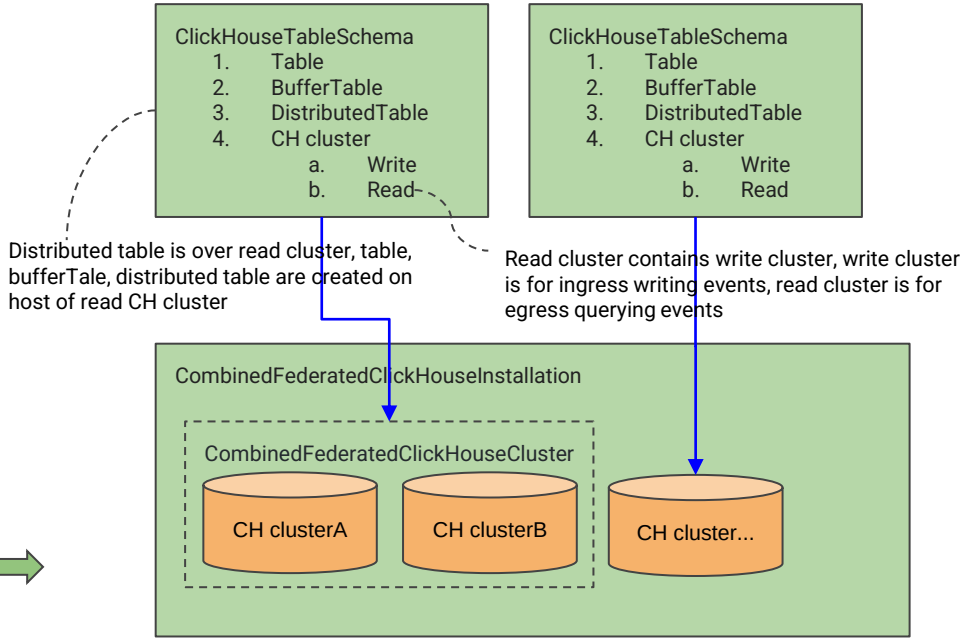


1. With CombinedFederatedClickHouseInstallation, we can combined clickhouse clusters together
2. With CombinedFederatedClickHouseCluster, we can combine any existing CH cluster together to create a new CH cluster

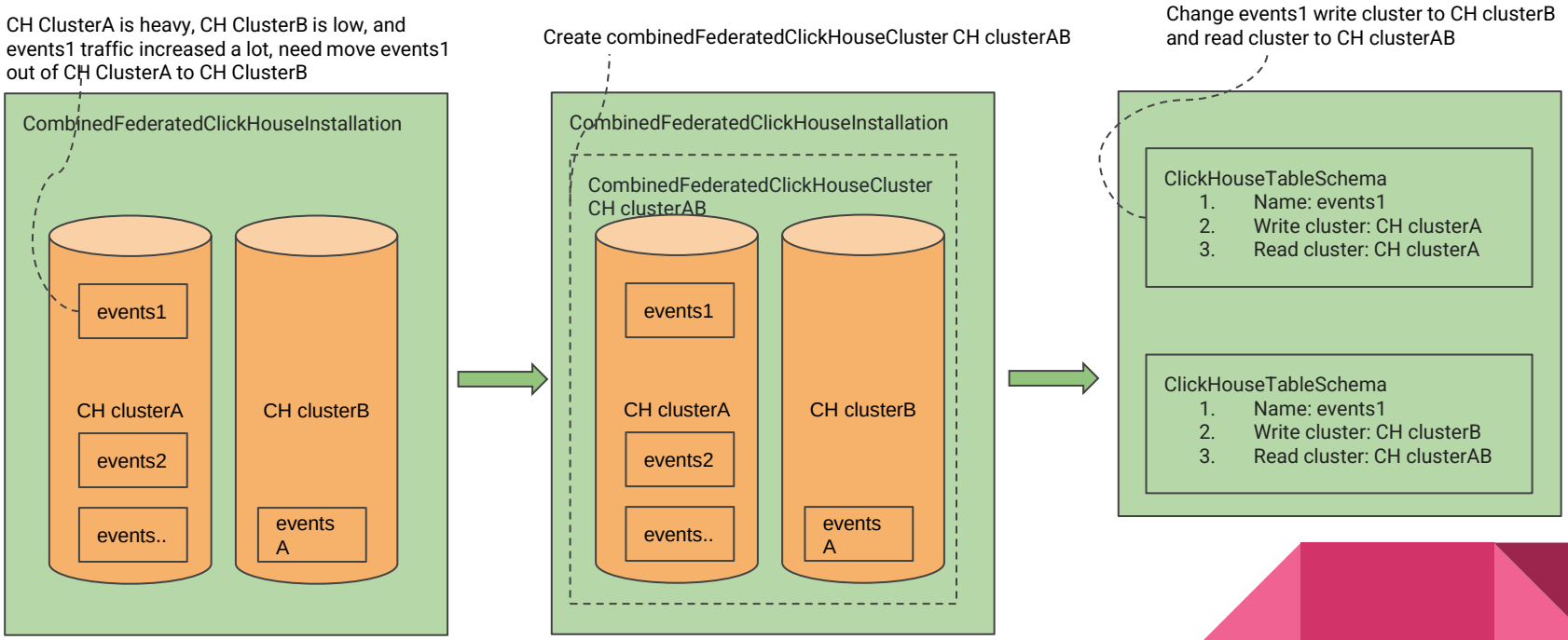
With above two features, we can create CH clusters and combine them to meet requirements, like scale clusters, data migration.

ClickHouse Clusters Management on Kubernetes via Operator - Table Schemas Management and Data Redistribution





ClickHouse Clusters Management on Kubernetes via Operator - Table Schemas Management and Data Redistribution



Q&A

Thank you for your attention