Общество с ограниченной ответственностью «Зетра»

ОГРН 1237700278155 ИНН 7707488152 КПП 770701001
Юр. адрес: 127473, г. Москва, вн.тер.г. муниципальный округ Тверской, пер 1-й Волконский, д. 15, помещение 1/3
тел.: +7 (909) 909-17-73,
e-mail: Zetrasoft@mail.ru

Функциональные характеристики программного обеспечения

«Система оркестровки контейнеризированных приложений ZETRAKUBER»

(«ZETRAKUBER»)

Москва

2023 г.

ZETRAKUBER - это программное обеспечение для автоматизации развёртывания, масштабирования и управления контейнеризированными приложениями, с расширенными функциями безопасности.

Областью применения Системы является автоматизация развёртывание, масштабирование и координация контейнеризированных приложений в условиях кластера. Системой поддерживаются основные технологий контейнеризации, включая Docker, rkt, а также поддержка технологий аппаратной виртуализации.

ZETRAKUBER реализован на базе программного обеспечения с открытым исходным кодом Kubernetes. ZETRAKUBER обладает следующими основными преимуществами по отношению к стандартному Kubernetes, а именно:

- поддержка отечественных стандартов шифрования (ГОСТ 34.10-2018, ГОСТ 34.11-2018, ГОСТ Р 34.12-2015, ГОСТ Р 34.13-2015).

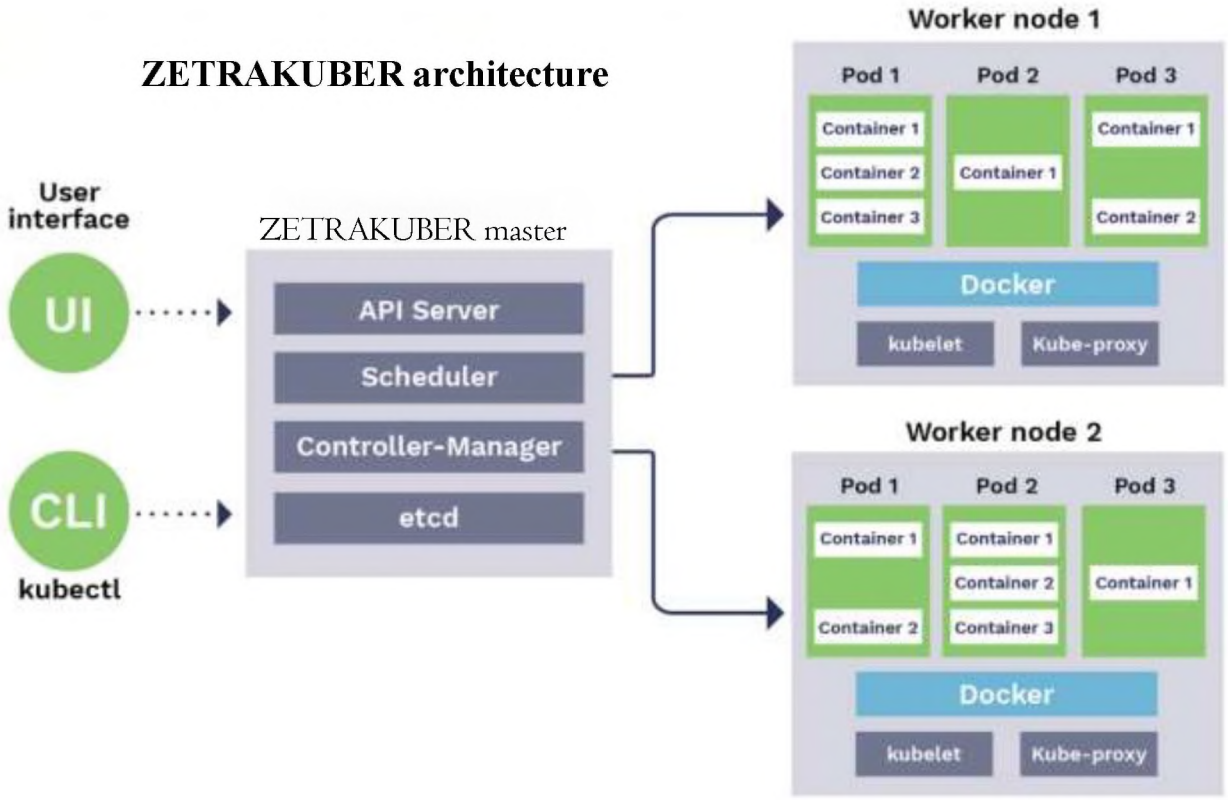
- повышенный уровень надежности (за счет исправления программных ошибок и оптимизации открытого исходного кода);

ZETRAKUBER включает в себя существенно переработанные компоненты Kubernetes, такие как:

- kubeadm
- kubectl
- kubelet
- kube-apiserver
- kube-controller-manager
- kube-proxy
- kube-scheduler

Вышеперечисленные компоненты поддерживают отечественные стандарты шифрования поддержка отечественных стандартов шифрования (ГОСТ 34.10-2018 - электронная цифровая подпись, ГОСТ 34.11-2018 - хеш-функция, ГОСТ Р 34.12-2015 — симметричные блочные шифры, ГОСТ Р 34.13-2015 — режимы применения симметричных блочных шифров) при установлении соединения и последующим взаимодействии между собой.

ZETRAKUBER master создавалась с учетом наработок в управлении удаленными проектами с масштабными рабочими нагрузками. Поэтому изначально в ней применяются продвинутые технологии, которые давно заменили принципы физического и даже виртуального развертывания систем. Контейнерное построение считается «облегченным» вариантом виртуальных машин.



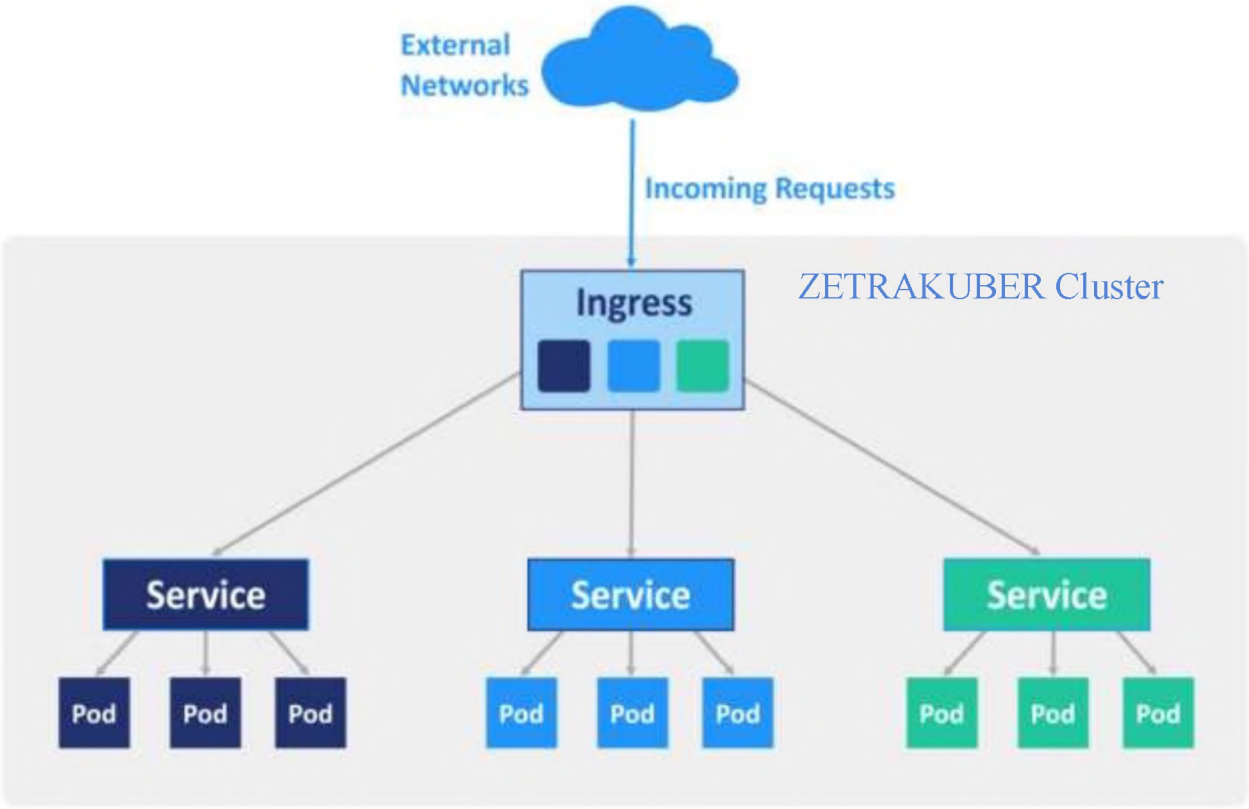
Преимущества:

1. Развертываемые приложения отделены от инфраструктуры для повышения безопасности.
2. Разработка и тестирование работают одинаково на локальном компьютере и в облаке.
3. Контейнеры свободно переносятся между Ubuntu, CoreOS, RHEL и другими системами.
4. Происходит изоляция ресурсов с прогнозированием производительности.

Вместо монолитного стека одной выделенной машины, приложения разбиваются на независимые друг от друга микросервисы с возможностью динамического развертывания и управления. За счет такого подхода ресурсы

используются более грамотно. Приложения наблюдают не только за метриками операционной системы, а еще и другими программами и оборудованием.

Контейнеры упрощают блочную разработку продуктов и их последующую сборку перед запуском. Такой подход дает возможность дорабатывать отдельные модули без влияния на работу остальных функций.



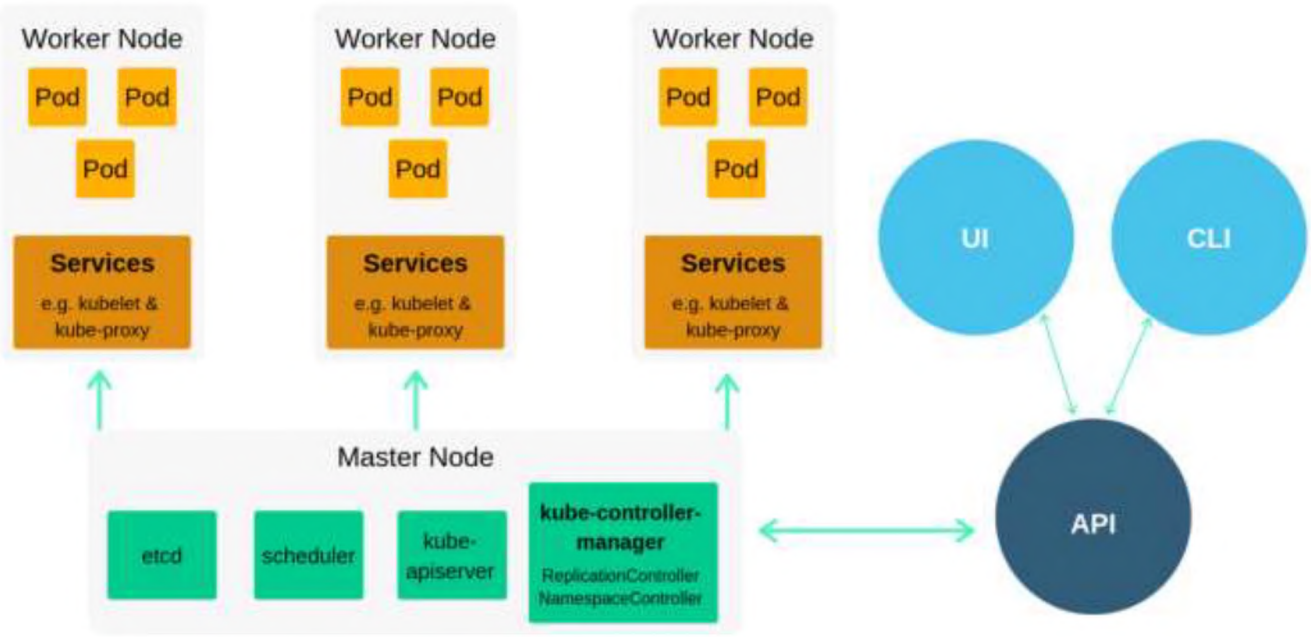
Перечень возможностей:

1. Обнаружение контейнера происходит по имени DNS или IP-адресу.
2. Система самостоятельно балансирует нагрузку и распределяет трафик в сети.
3. Подключение выбранного типа хранилища происходит в автоматическом режиме.
4. Платформа перезапускает отказавшие контейнеры или блокирует к ним доступ.
5. Конфиденциальная информация хранится изолированно от других данных.

Развертывание или откат изменений происходит по заданному сценарию, без участия пользователя. То же относится к резервированию аппаратных ресурсов, количеству процессорных ядер, оперативной памяти (на каждый отдельно взятый контейнер).

Основные компоненты архитектуры

Несмотря на удобство и простоту ZETRAKUBER, перед его использованием, при разработке или развертывании приложений, необходимо разобраться в архитектуре. Например, понять, как соединяются между собой контейнеры (через интерфейс API), и узнать, почему компоненты разделены на две основные группы – Master Node и Worker Node.



Базовые компоненты:

1. Nodes – виртуальная (физическая) машина, на мощностях которой запущены контейнеры.
2. Pods – базовые модули управления приложениями, состоящие из одного или нескольких контейнеров.
3. Volume – ресурс, позволяющий одновременно запускать несколько контейнеров.

4. Kube-Proxy – комплекс из прокси-сервера и модуля балансировки нагрузки, позволяющий маршрутизировать входящий трафик под конкретный контейнер Pods.
5. Kubelet – транслятор статусов Pods на узле и контроллер корректности работы контейнера и образа в целом.

Перечисленные компоненты устанавливаются автоматически при помощи сторонних инструментов или вручную, по отдельности. Они составляют модуль под названием Master Node, где собраны все управляющие и контролирующие функции. Через API они связываются с контейнерами, считывают их показатели, дают команду на запуск, штатную остановку или принудительное закрытие.

Под (pod)

Под — это базовая единица для запуска и управления приложениями: один или несколько контейнеров, которым гарантирован запуск на одном узле, обеспечивается разделение ресурсов и межпроцессное взаимодействие и предоставляется уникальный в пределах кластера IP-адрес.

Последнее позволяет приложениям, развёрнутым на поде, использовать фиксированные и предопределённые номера портов без риска конфликта. Поды могут напрямую управляться с использованием API ZETRAKUBER или управление ими может быть передано контроллеру.

Узел (node)

Отдельная физическая или виртуальная машина, на которой развёрнуты и выполняются контейнеры приложений. Каждый узел в кластере содержит сервисы для запуска приложений в контейнерах (например Docker, а также компоненты, предназначенные для централизованного управления узлом. Узел состоит из следующих компонент:

Kubelet
Container Runtime

Kube-proxy
Один или несколько подов.

Том (volume)

Общий ресурс хранения для совместного использования из контейнеров, развёрнутых в пределах одного пода.

Метки и селекторы

Все объекты управления (узлы, поды, контейнеры) в ZETRAKUBER помечаются метками (label), селекторы меток (label selector) — это запросы, которые позволяют получить ссылку на объекты, соответствующие какой-то из меток. Метки и селекторы — это главный механизм ZETRAKUBER, который

позволяет выбрать, какой из объектов следует использовать для запрашиваемой операции.

Сервис

Сервисом в ZETRAKUBER называют совокупность логически связанных наборов подов и политик доступа к ним. Например, сервис может соответствовать одному из уровней программного обеспечения, разработанного в соответствии с принципами многоуровневой архитектуры программного обеспечения. Набор подов, соответствующий сервису, получается в результате выполнения селектора соответствующей метки. ZETRAKUBER обеспечивает функции обнаружения сервисов и маршрутизации по запросу. В частности, система умеет переназначать необходимые для обращения к сервису IP-адрес и доменное имя сервиса различным подам, входящим в его состав. При этом обеспечивается балансировка нагрузки в стиле Round robin DNS между подами, чьи метки соответствуют сервису, а также корректная работа в том случае, если один из узлов кластера вышел из строя и размещённые на нём поды автоматически были перемещены на другие узлы. По умолчанию сервис доступен внутри управляемого ZETRAKUBER кластера — например, поды бэкенда группируются для обеспечения балансировки нагрузки и в таком виде предоставляются фронтенду. Также кластер может быть настроен и для предоставления доступа к входящим в его состав подам извне как к единому фронтенду.

Контроллер (controller)

Процесс, который управляет состоянием кластера, пытаясь привести его от фактического состояния к желаемому; он делает это, оперируя набором подов, определяемых с помощью селекторов меток и являющихся частью определения контроллера. Выполнение контроллеров обеспечивается компонентом Kubernetes Controller Manager. Один из типов контроллеров, самый известный — это контроллер репликации (Replication Controller), который обеспечивает масштабирование, запуская указанное количество копий пода в кластере. Он

также обеспечивает запуск новых экземпляров пода в том случае, если узел, на котором работает управляемый этим контроллером под, выходит из строя. Другие контроллеры, входящие в основную систему ZETRAKUBER, включают в себя «DaemonSet Controller», который обеспечивает запуск пода на каждой машине (или подмножеством машин), и «Job Controller» для запуска подов, которые выполняются до завершения, например, как часть пакетного задания.

Операторы (operators)

Специализированный вид программного обеспечения ZETRAKUBER, предназначенный для включения в кластер сервисов, сохраняющих своё состояние между выполнениями (stateful), таких как СУБД, системы мониторинга или кэширования. Назначение операторов — предоставить возможность управления stateful-приложениями в кластере ZETRAKUBER прозрачным способом и скрыть подробности их настроек от основного процесса управления кластером ZETRAKUBER.

Архитектура и компоненты

ZETRAKUBER реализует архитектуру «ведущий — ведомый»: выделяется подсистема управления кластером, а часть компонентов управляют индивидуальными, ведомыми узлами (называемых собственно узлами ZETRAKUBER)

Подсистема управления

Подсистема управления обеспечивает коммуникацию и распределение нагрузки внутри кластера; компоненты подсистемы могут выполняться на одном или на нескольких параллельно работающих ведущих узлах, совместно обеспечивающих режим высокой доступности.

Etcd

Компонент подсистемы управления, отвечающий за согласованное хранение конфигурационных данных кластера, в некотором смысле — распределённый эквивалент каталога /etc Unix-систем. Реализован как

легковесная распределённая NoSQL-СУБД класса «ключ — значение»; создан в рамках проекта CoreOS.

Сервер API

Ключевой компонент подсистемы управления, предоставляющий API в стиле REST (с использованием коммуникации в формате JSON поверх HTTP-транспорта), и используемый для организации внешнего и внутреннего доступа к функциям ZETRAKUBER. Сервер API обновляет состояние объектов, хранящееся в etcd, позволяя своим клиентам управлять распределением контейнеров и нагрузки между узлами управляемой системы.

Планировщик (scheduler)

Компонент подсистемы управления, который выбирает, на каком узле должен выполняться конкретный под, опираясь на критерии доступности ресурсов. Планировщик отслеживает использование ресурсов на каждом из узлов, обеспечивая распределение нагрузки так, чтобы она не превышала доступный объём ресурсов. Для этой цели планировщик должен обладать информацией о доступных на каждом из узлов ресурсах, требованиях к ним со стороны управляемых подов, а также различных дополнительных пользовательских ограничениях и политиках, таких как QoS, требования аффинитета и антиаффинитета (affinity — anti-affinity — связки или развязки объектов управления друг с другом), локализации данных. Иными словами, роль планировщика — находить и предоставлять ресурсы в зависимости от запросов, возникающих в связи с загрузкой.

Менеджер контроллеров (controller manager)

Процесс, выполняющий основные контроллеры ZETRAKUBER, такие как DaemonSet Controller и Replication Controller. Контроллеры взаимодействуют с сервером API ZETRAKUBER, создавая, обновляя и удаляя управляемые ими ресурсы (поды, точки входа в сервисы, и другие).

Kubectl

Интерфейс командной строки, наряду с API обеспечивающий управление ресурсами, подконтрольными ZETRAKUBER.

Компоненты узлов

Процедура работы ZETRAKUBER состоит в том, что ресурсы узлов динамически распределяются между выполняемыми на них подами. Каждый узел в кластере содержит ряд типовых компонентов. Сервис для запуска контейнеров обеспечивает функции выполнения контейнеров соответствующего вида (в зависимости от типа используемого контейнерного движка). С точки зрения программной среды ZETRAKUBER, контейнеры инкапсулируются в подах, при этом сами контейнеры являются наиболее низкоуровневыми программными компонентами, с которыми взаимодействует программное обеспечение ZETRAKUBER. Они, в свою очередь, содержат выполняемые приложения, библиотеки и иные необходимые для работы этих приложений ресурсы. Для внешнего мира контейнеры доступны через назначаемый каждому из подов IP-адрес.

1. Kubelet

Kubelet - это главный ZETRAKUBER agent. На каждом узле работает по kubelet. Отвечает за статус выполнения подов на узле — отслеживает, корректно ли выполняется каждый из контейнеров, находясь в рабочем состоянии. Kubelet обеспечивает запуск, остановку и управление контейнерами приложений, организованными в поды. После установки kubelet на узел - он регистрируется в кластере и следит за API сервером чтобы получать задания Функционально Kubelet можно рассматривать как аналог supervisord. Если обнаруживается, что какой-то из подов находится в неверном состоянии, компонент пытается осуществить его повторное развёртывание и перезапуск на узле. Статус самого узла отправляется на подсистему управления каждые несколько секунд в форме диагностических сообщений (heartbeat message). Если мастер-узел, исходя из содержания этих сообщений или их отсутствия,

обнаруживает, что конкретный узел не работает должным образом, процесс подсистемы управления Replication Controller пытается перезапустить необходимые поды на другом узле, находящемся в рабочем состоянии.

2. Container runtime

Container Runtime (CRI - Container Runtime Interface. Например Docker, containerd, CRI-O, katacontainers, gVisor)

3. Kube-proxy

Компонент, являющийся комбинацией сетевого прокси-сервера и балансировщика нагрузки. Реализованные в нём операции сетевого уровня используют абстракцию сервиса. Он отвечает за маршрутизацию входящего трафика на конкретные контейнеры, работающие в пределах пода, расположенного на узле. Маршрутизация обеспечивается на основе IP-адреса и порта входящего запроса.

4. cAdvisor

Агент системы внутреннего мониторинга ZETRAKUBER, собирающий метрики производительности и информацию об использовании контейнерами, работающими в пределах узла, таких ресурсов как время работы центрального процессора, оперативной памяти, нагрузку на файловую и сетевую системы.

5. Поды

В узле может быть несколько подов. Каждый под может содержать в себе несколько контейнеров.