Теоретическое введение

В данной практической работе необходимо будет работать со множеством сервисов. В теоретическом введение будет дана краткая информация о них.

Kraken - это расширяемый, декларативный, высокопроизводительный API-шлюз с открытым исходным кодом.

Его основная функциональность заключается в создании API, который действует как агрегатор многих микросервисов в конечные точки, автоматически выполняя тяжелую работу: агрегирование, преобразование, фильтрацию, декодирование, дросселирование, аутентификацию и многое другое.

Конфигурацию для него можно писать самостоятельно в формате JSON или же использовать онлайн конфигуратор KrakenD Designer.

Kafka Apache — распределенная система обмена сообщениями между серверными приложениями в режиме реального времени. Благодаря высокой пропускной способности, масштабируемости и надежности применяется в компаниях, работающих с большими объемами данных. Написана на языках Java и Scala. Kafka — связующее звено между отдельными функциональными модулями большой системы. Например, с ее помощью можно подписать микросервис на другие компоненты для регулярного получения обновлений.

Управление приложениями — сложный аспект Kubernetes. Helm значительно упрощает его, предоставляя единый метод упаковки программного обеспечения, поддерживающий контроль версий. Helm устанавливает пакеты (называются Чартами в Helm) для Kubernetes и управляет ими, как это делают yum и apt. Чарты содержат все определения ресурсов, необходимые для запуска приложения, инструмента или службы внутри кластера Kubernetes.

Jaeger – это программное обеспечение, которое вы можете использовать для отслеживания и устранения проблем во взаимосвязанных программных компонентах, называемых микросервисами. Несколько микросервисов взаимодействуют между собой с целью выполнения единой программной функции. Разработчики используют Jaeger для визуализации цепочки событий при взаимодействии микросервисов, чтобы изолировать проблему, когда что-то

идет не так, как требуется. Jaeger также называется трассировкой Jaeger, потому что отслеживает, или трассирует, путь запроса через серию взаимодействий между микросервисами.

Полезные ссылки

K8S для начинающих. Первая часть — Текст: электронный [сайт]. — URL: https://habr.com/ru/post/589415/

KrakenD Designer | KrakenD — Текст: электронный [сайт]. — URL: https://designer.krakend.io/

Operator for Kubernetes | Jaeger — Текст: электронный [сайт]. — URL: https://www.jaegertracing.io/docs/1.24/operator/

Kafka 3.3 Documentation | Kafka — Текст: электронный [сайт]. — URL: https://kafka.apache.org/documentation/

Helm Docs | Helm — Текст: электронный [сайт]. — URL: https://helm.sh/docs/

KrakenD Documentation | KrakenD — Текст: электронный [сайт]. — URL: https://www.krakend.io/docs/overview/

Практическая часть

Разработать Backend или часть Backend системы в соответствии с выбранной для ВКР темой с использованием Spring/Spring Boot. Темы в группах не должны повторяться.

Система должна иметь сервис авторизации с хранением токенов в RedisDB и минимум 2 сервиса, которые должны общаться между собой с помощью шины сообщений с использованием Kafka. Хранение данных должно производиться с помощью СУБД PostgreSQL.

В качестве API-шлюза к сервисам должен выступать KrakenD.

Необходимо логировать все обращения к сервисам с помощью Graylog. В логах должны указываться HTTP-метод, URL, IP-адрес отправителя.

Также необходимо собирать метрики с баз данных с помощью связки Prometheus + Grafana, а все транзакции подвергать трассировке с помощью Jaeger.

В случае невозможности реализации системы с использованием предложенного стека технологий необходимо обосновать это и предложить решение на собственном стеке.

Для каждого сервиса необходимы:

— Установка в Minikube при помощи Helm-чартов;

— Настройка лимитов;

— Настройка Liveness, Readiness and Startup Probes;

— Настройка перезапуска при падении;

— Настройка использования ресурсов пода на 60% и использование Autoscaling.

Для СУБД:

— Использование persistent volume;

— Конфигурация макс количества соединений;

— Создание пользователя для считывания данных мониторингом;

— Настройка мониторинга пода с БД при помощи Prometheus и Grafana;

— Должен соблюдаться паттерн “1 БД на 1 сервис”.

В отчете вам необходимо отобразить все этапы конфигурации системы, показать конфигурационные файлы Minikibe, Helm Charts, KrakenD, отобразить работу системы с учетом авторизации, показать процесс в сервисах Kafka, Jaeger, Prometheus, Grafana, Graylog, отобразить сохранение данных в persistent volume при отключении пода с БД.

Вопросы к практической работе

1. Назовите плюсы и минусы использования helm. Какие форматы конфигурационных файлов он поддерживает? Какие существуют аналоги?
2. Почему для микросервисного взаимодействия используется шина сообщений? Какие аналоги для микросервисного взаимодействия существуют? Каковы их преимущества и недостатки?
3. Для чего используется persistent volume в БД? Какие преимущества и недостатки у паттерна “1 БД на 1 сервис”?
4. Почему для чтения данных из БД системами мониторинга создается отдельный пользователь? К чему может привести использование аккаунта с лишними правами
5. Какой функционал предоставляет KrakenD? Какие существуют аналоги?

Критерии оценки

За выполнение данной практической работы можно максимально получить 2 балла.

Критерии на выставление 2 баллов:

— Соблюдены общие требования выполнения практических работ, представленные в документе “Требования к выполнению практических работ”.

— В отчете отображены все этапы конфигурации системы

— Показаны конфигурации и Helm Charts в Minikube

— Показана работоспособность системы с учетом авторизации

— Показано сохранение данных в persistent volume при отключении пода с БД

— Показана работоспособность Kafka, Jaeger, Prometheus, Grafana, Graylog на примере отправки запросов

— Показана конфигурация KrakenD

— Сделан отчет с описанием и скриншотами выполненных заданий

— Дан полный и развернутый ответ на все вопросы преподавателя, как по вопросам к практике, так и по дополнительным вопросам к выполненному заданию.

Критерии на выставление 1 балла:

— Соблюдены общие требования выполнения практических работ, представленные в документе “Требования к выполнению практических работ”.

— В отчете отображены все этапы конфигурации системы

— Показаны конфигурации и Helm Charts в Minikube

— Показана работоспособность системы с учетом авторизации

— Показано сохранение данных в persistent volume при отключении пода с БД

— Не показана работоспособность Kafka, Jaeger, Prometheus, Grafana, Graylog на примере отправки запросов

— Показана конфигурация KrakenD

— Сделан отчет с описанием и скриншотами выполненных заданий

— Дан полный и развернутый ответ на все вопросы преподавателя, как по вопросам к практике, так и по дополнительным вопросам к выполненному заданию.

Критерии на выставление 0 баллов:

— Не соблюдены общие требования выполнения практических работ, представленные в документе “Требования к выполнению практических работ”.

— В отчете не отображены все этапы конфигурации системы

— Не показаны конфигурации и Helm Charts в Minikube

— Не показана работоспособность системы с учетом авторизации

— Не показано сохранение данных в persistent volume при отключении пода с БД

— Не показана работоспособность Kafka, Jaeger, Prometheus, Grafana, Graylog на примере отправки запросов

— Не показана конфигурация KrakenD

— Сделан отчет с описанием и скриншотами выполненных заданий.

— Студент не смог ответить ни на вопросы к практической работе, ни на вопросы к ходу выполнения работы.