SRE практики и инструменты

Site Reliability Engineering, как дисциплина, решает проблемы надежности и доступности сервисов

Длительность курса: 122 академических часа

1 Введение в SRE

1 **Введение в SRE // ДЗ**

Цели занятия:

обсудить историю SRE, связь с DevOps;
рассмотреть варианты взаимодействия;
обсудить подход SRE к управлению сервисами и различные варианты Reliability Engineering: Site, Production, Database.

Домашние задания

1 Настройка окружения

Цель: Нужно подготовить окружение для дальнейшей работы по курсу

VSCode + MARP (либо другая IDE/редактор)
https://code.visualstudio.com/
https://marp.app/

Docker Desktop (либо Docker + minikube)
https://www.docker.com/products/docker-desktop
WSL2 Ubuntu 20.04 (либо другой Linux)
https://docs.microsoft.com/en-us/windows/wsl/install-win10

Git

Vagrant https://www.vagrantup.com/
настроенный libvirt или virtualbox
https://www.virtualbox.org/

Ansible
Molecule https://molecule.readthedocs.io/en/latest/

Terraform https://www.terraform.io/

Дополнительное ПО
https://www.mkdocs.org/
https://diagrams.mingrammer.com/docs/guides/diagram

2 **Основные принципы SRE // ДЗ**

Цели занятия:

разобрать основные приемы работы и обсудить среду эксплуатации с точки зрения SRE.

Краткое содержание:

Типичная для SRE инженерная культура
Цикл Знать - Делать - Обучаться
Измеряем все
Примеры первых метрик
Примеры первых шагов по внедрению и утилиты
Риски внедрения и антипаттерны

Домашние задания

1 PoC GTFO

Цель: Продолжаем подготовку окружения для тестирования SRE-подходов.

Создайте git-репозиторий (github/bitbucket) доступный для клонирования онлайн.

Опишите в Readme https://www.makeareadme.com/ что будет происходить:
- краткое описание проекта
- как добавить новый код
- как продеплоить приложение
Добавьте Readme в репозиторий

Создайте диаграмму для будущего приложения используя https://github.com/mingrammer/diagrams
Добавьте диаграмму и ее код в репозиторий

Создайте локальное окружение для разработки используя Vagrant
Добавьте код локального окружения в репозиторий

Напишите простой деплой для любого веб-приложения, например https://github.com/Zenahr/flask-sqlite3-todo-crud и продеплойте приложение на локальное окружение
Добавьте код в репозиторий

Используйте Mkdocs для оформления документации. Добавьте документацию в репозиторий.

3 **SLI, SLA, SLO и управление рисками // ДЗ**

Цели занятия:

управление рисками - одна из самых важных практик. Обсудим, как SRE оценивает риски, управляет ими и использует лимит времени недоступности сервиса для того, чтобы объективно принимать решения. SLI, SLA, SLO - фундаментальные понятия для SRE. Рассмотрим каждый из этих понятий и определим показатели для сервиса.

Краткое содержание:

SLI;
SLA;
SLO;
error budget.

Домашние задания

1 Сформулировать SLO для тестового окружения

Цель: Цель: закрепить понимание SLI/SLO
Результат: Получить документ, описывающий SLO для тестового окружения из предыдущей домашней работы

Определить SLI для приложения https://github.com/Zenahr/flask-sqlite3-todo-crud
Определить SLI для окружения
Определить SLO для окружения

2 Everything as a Code

1 **GIT 101: совместная работа, CI, вендоринг**

---

2 **Практика управление конфигурацией. Ansible // ДЗ**

Цели занятия:

обсудить управление конфигурацией с точки зрения SRE;
рассмотреть различные подходы.
IaC Ansible Clouds

Домашние задания

1 Дописать роль для исполнения на другом дистрибутиве (.deb/.rpm)

Цель: Ознакомиться со структурой Ansible role
Освоить основные техники работы с кодом для Ansible

Выберете одну из ролей ниже или используйте любую другую роль:

https://github.com/kevit/ansible-role-monica
https://github.com/kevit/ansible-role-dedcon
https://github.com/kevit/ansible-role-funkwhale

Форкните код
Допишите необходимый для исполнения на другом дистрибутиве код (например на Centos/Rhel/AWS linux)

---

3 **Практика управление конфигурацией. Terraform // ДЗ**

Домашние задания

1 написать правило OPA проверяющее terraform plan

3 Экосистема Kubernetes

1 **Linux 101: cgroups/namespaces/network/containers**

2 **Экосистема Kubernetes**

3 **Практика управление конфигурацией. Helm**

4 **QA-сессия**

Цели занятия:

синхронизироваться по вопросам учёбы.

Краткое содержание:

обратная связь;
ответы на вопросы.

4 Автоматизация (Toil elimination)

1 **Автоматизация: CLI, Data transformation // ДЗ**

Цели занятия:

рассмотреть подход SRE к автоматизации, а также примеры ее реализации — как успешные, так и неудачные.

Краткое содержание:

Shell/JQ.

Домашние задания

1 Распарсить вывод API и отдать его в формате Prometheus

Цель: Использовать базовую автоматизацию сбора данных и их конвертацию в нужный формат, по возможности используя утилиты из лекции

Формат экспортера можно найти тут https://github.com/prometheus/docs/blob/master/content/docs/instrumenting/exposition_formats.md#text-format-example

Варианты:

1. Приложение https://github.com/Zenahr/flask-sqlite3-todo-crud/blob/master/app.py отвечает на три запроса:
/ GET
/add POST
/update POST
Проверьте код ответа приложения, время ответа либо иные важдые для ваших SLI на данные запросы и отдайте эту информацию в формате Prometheus

2. Получите ответ от API http://open-notify.org/Open-Notify-API/ISS-Location-Now/ и сконвертируйте его в формат Prometheus

2 **Автоматизация: Runbook Automation (Jenkins/AWX/Rundeck) // ДЗ**

Цели занятия:

лямбды/ifft

Домашние задания

1 Напишите workflow для n8n

Цель: Освоить на практике работу с workflow
Использовать навыки работы с структурированным текстом

Нужно проверить что информация в CSV и JSON идентична. Для этого создать n8n workflow сравнивающий два источника данных

URL:
http://xmlcalendar.ru/data/ru/2022/calendar.csv
http://xmlcalendar.ru/data/ru/2022/calendar.json

Добавить workflow в ваш git репо и опубликовать в Readme ссылку по которой можно импортировать workflow в n8n

---

3 **Автоматизация: Low-Code/No-Code**

5 Мониторинг и практика оповещений

1 **Задачи мониторинга и алертинга**

2 **Observability**

3 **Dashboard as a Code**

4 **QA-сессия**

6 Практики SRE

1 **Непрерывная поставка и управление изменениями // ДЗ**

Цели занятия:

обсудить, как обеспечить уверенность в стабильности и качестве выпускаемого продукта с помощью непрерывной поставки.

Домашние задания

1 В репозитории на гитлаб/битбакет/github actions сделать автоматический прогон тестов на каждый коммит

Цель: Цель дз - самоястоятельно настроить процесс CI/CD.

Для репозитория, созданного во втором домашнем задании, (или любого форкнутого проекта с гитхаба) необходимо настроить процесс CI/CD.

Используя инструменты CI/CD, например, https://github.com/marketplace/circleci https://about.gitlab.com/stages-devops-lifecycle/continuous-integration/ настроить автоматическую запуск пайплайна по комиту (у обоих этих инструментов есть бесплатный пробный период, или можно поднять локальный гитлаб в докере, например, https://docs.gitlab.com/ee/install/docker.html).

В пайплан должны входить
- сборка проекта (желательно собирать приложение в Docker-контейнер, пригодится в следующем дз. Имеет смысл в качестве базового образа брать Linux alpine https://hub.docker.com/_/alpine/, тк это минимизированный образ, в котором нет ничего лишнего. За счет выбора базового имаджа можно существенно сократить время сборки).
- запуск тестов (достаточно наличия одного, самого простого теста. Самое главное - чтобы для запуска тестов была сделана отдельная джоба, которую можно запустить без сборки проекта)

2 **Управление релизами // ДЗ**

Цели занятия:

рассмотреть практику управления релизами и координацию процесса;
разработать чек-лист для запуска;
обсудить приемы надежных релизов и роль SRE инженера.

Домашние задания

1. отбранчеваться от мастер-репо , написать свой кусок кода, создать PR и вмержить обратно

---

3 **Тестирование надежности систем // ДЗ**

Цели занятия:

рассмотреть виды тестирования ПО, процессы и инструменты;
обсудить Chaos Engineering и проведение учений.

Домашние задания

1. протестировать тестовый сервис из двух микросервисов

---

4 **Управление нагрузкой предотвращения перегрузок и сбоев // ДЗ**

Цели занятия:

обсудить балансировку нагрузки на уровне фронтенда и датацентра;
рассмотреть политики балансировки нагрузки.

Домашние задания

1. отбранчеваться от мастер-репо , написать свой кусок кода, создать PR и вмержить обратно

5	**Практика on-call и жизненный цикл SRE команды**	**Цели занятия:** погрузиться в жизнь дежурного инженера; поговорить про организацию и культуру дежурств; разобрать реализации на практике.
6	**Практика постмортемов**	**Цели занятия:** поговорить про философию постмортемов; рассмотреть хороший и плохой постмортем; понять, с чего начать внедрение культуры постмортема; разобрать примеры, инструменты и шаблоны.
7	**Практика диагностики и решения проблем**	**Цели занятия:** рассмотреть примеры анализа реальных ситуаций и изучить инструментарий.
8	**Customer Reliability Engineering**	

7 Поиск работы

1 **Как подсветить навыки в резюме. Шаблон резюме**

2 **Этапы собеседования: чего ждать? Live-собеседование с поведенческими вопросами**

3 **QA-сессия**

8 Проектная работа

1 **Выбор темы и организация проектной работы**

Цели занятия:

выбрать и обсудить тему проектной работы;
спланировать работу над проектом;
ознакомиться с регламентом работы над проектом.

Краткое содержание:

правила работы над проектом и специфика проведения итоговой защиты;
требования к результату проекта и итоговой документации.

---

2 **Консультация по проектам и домашним заданиям - промежуточная приемка**

Цели занятия:

получить ответы на вопросы по проекту, ДЗ и по курсу.

Краткое содержание:

вопросы по улучшению и оптимизации работы над проектом;
затруднения при выполнении ДЗ;
вопросы по программе.

---

3 **Защита проектных работ**

Цели занятия:

защитить проект и получить рекомендации экспертов.

Краткое содержание:

презентация проектов перед комиссией;
вопросы и комментарии по проектам.

Домашние задания

1 Итоговый проект

Цель: Вы первый SRE в компании, которая построена вокруг блога, продающего вино.

yandex.cloud балансер nginx wordpress mysql memcached nfs

очередь для обработки заказов и ml по логам формирующий ленту контента пользователю.

Внедряем:

Первый этап
Запустить DevOps CI/CD для вашего кода
Automate infrastructure scaling
Как вы будете соблюдать правило 50/50 (стратегия)
Проведите SRE тренинг для других команд (pdf презентация)

Второй этап
Определите SLA,SLO,SLI. Чем вы руководствовались, с кем вы общались в компании?
Измерьте поведение вашей системы на соответствие SL*. Как вы этого добьетесь?
Определите Error-Budgeet (velocity vs quality)
Определите OKR для SRE

Третий этап
Внедрите мониторинг
Golden signals: latency, saturation, traffic, errors
В процессе релиза упал memcached. Выпустите Blameles postmortem
Исправьте архитектуру чтобы этого больше не происходило

Четвертый этап
Проверьте что падение memcached больше не ломает систему. проведите эксперимент
Handle improvement discussion
Response scenarios based on proactive monitoring