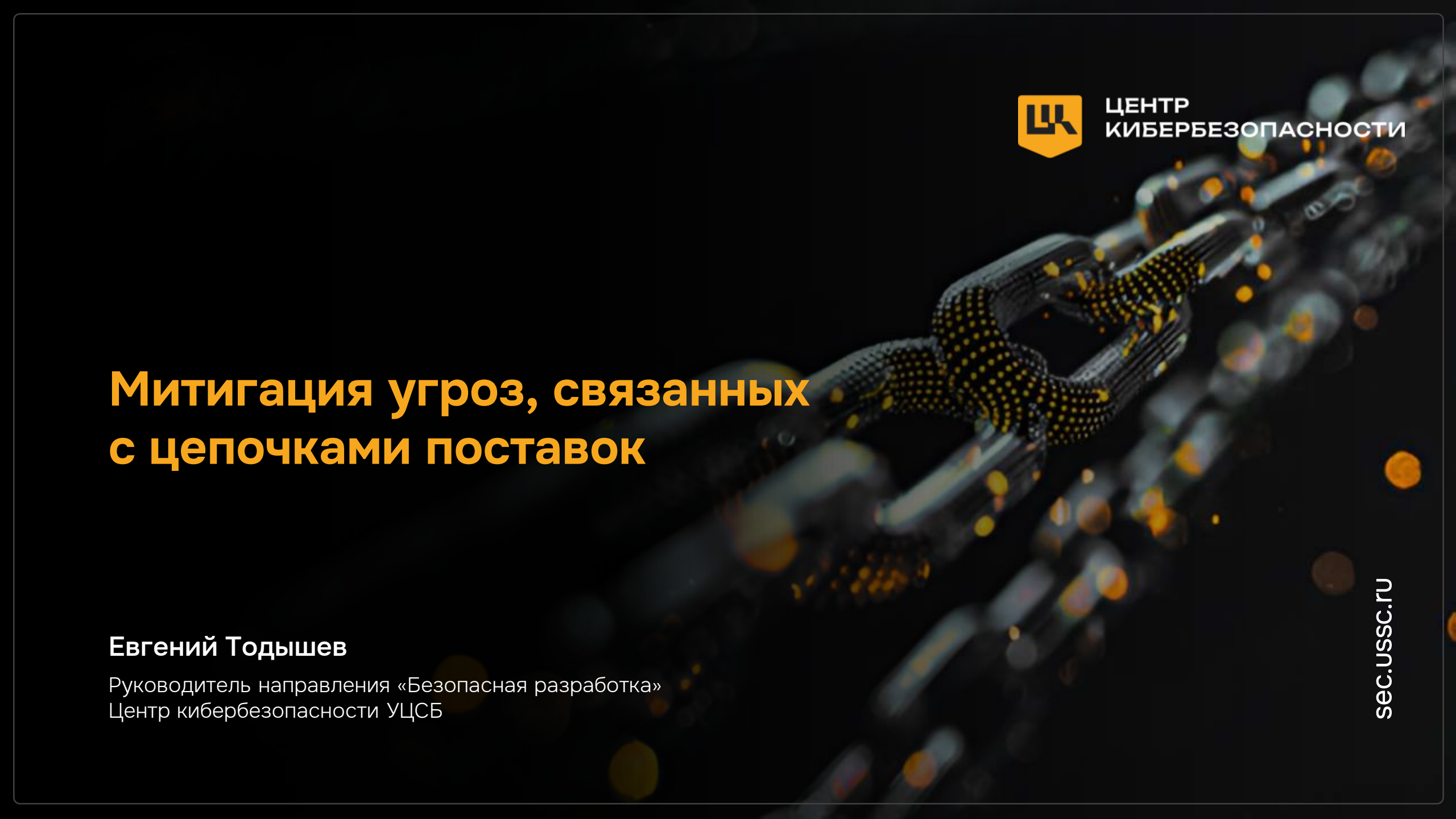


ЦЕНТР КИБЕРБЕЗОПАСНОСТИ

Митигация угроз, связанных с цепочками поставок

Евгений Тодышев

Руководитель направления «Безопасная разработка»
Центр кибербезопасности УЦСБ

sec.ussc.ru

WHOAMI

- Руководитель направления «Безопасная разработка» в Центре кибербезопасности УЦСБ
- Помогаем нашим Заказчикам создавать безопасные приложения
- Обеспечиваем безопасность облаков и микросервисов
- Участвую в создании профильных мероприятий
- CPO Apsafe

ЦЕНТР КИБЕРБЕЗОПАСНОСТИ

АКТУАЛЬНОСТЬ БЕЗОПАСНОСТИ ЦЕПОЧЕК ПОСТАВКИ

Реальные примеры атак из-за уязвимостей цепочки поставок

09/2021 SushiSwap

Кто: подрядчик с доступом к SCM

Что: отправил коммит, перенаправляющий криптовалюту на свой кошелек

Почему: отсутствовала проверка коммитов вторым лицом

Последствия: украдены 3 миллиона долларов в виде криптовалюты

03/2021 php

Кто: злоумышленник, скомпрометировавший git.php.net

Что: отправил два коммита, позволяющих выполнять произвольный PHP-код, передаваемый в HTTP-заголовке Useragent

Почему: недостаточная защита SCM

Последствия: удаленное выполнение кода

2019 webmin

Кто: злоумышленник, взломавший сервер сборки

Что: изменил процесс сборки ПО

Почему: отсутствие проверки источников кода

Последствия: захват серверов с установленным Webmin

04/2021 Codecov

Кто: злоумышленник, получив ключ от GCS

Что: модифицировал ПО, которое пользователи загружали из GCS

Почему: отсутствие проверки происхождения артефакта и его источника

Последствия: кража паролей и ключей шифрования

Реальные примеры атак из-за уязвимостей цепочки поставок

04/2024

Кто: злоумышленник, став мейнтейнером OSS проекта XZ Utils

Что: модифицировал ПО, которое пользователи загружали из официального репозитория

Почему: отсутствие проверки контроля вносимых изменений, фишинг

Последствия: ряд дистрибутивов Linux были заражены

Прогнозы безопасности цепочек поставок

Основные прогнозы на 2024 год

- Все больше приложений используют ПО с открытым исходным кодом

- Количество атак на цепочки поставок будет только расти

- Процент библиотек с уязвимостями растет
- Так же как и пакетов, содержащих в себе вредоносный функционал

УГРОЗЫ И ИХ МИТИГАЦИЯ

Угрозы цепочек поставок

01. | Исходный код

- Уязвимый код
- Компрометация SCM

Угрозы цепочек поставок

01. | Исходный код

- Уязвимый код
- Компрометация SCM

02. | Сборка

- Изменение кода вне SCM
- Компрометация CD
- Внесение изменений в репозиторий пакетов в обход CI/CD
- Компрометация репозитория пакетов
- Передача плохих пакетов потребителю

Угрозы цепочек поставок

01. | Исходный код

- Уязвимый код
- Компрометация SCM

02. | Сборка

- Изменение кода вне SCM
- Компрометация CD
- Внесение изменений в репозиторий пакетов в обход CI/CD
- Компрометация репозитория пакетов
- Передача плохих пакетов потребителю

03. | Зависимости

- Использование скомпрометированной зависимости

Митигация угроз в процессе разработки

01. Исходный код

- Практики безопасности
- Подписание кода
- Утверждение изменений
- Регулярный анализ
- Аудит

Митигация угроз в процессе разработки

01. Исходный код

- Практики безопасности
- Подписание кода
- Утверждение изменений
- Регулярный анализ
- Аудит

02. Сборка

- Контроль подписи
- Подписания артефактов
- Контроль сборки
- Безопасность CI/CD
- Доверенный репозиторий

Митигация угроз в процессе разработки

01. Исходный код

- Практики безопасности
- Подписание кода
- Утверждение изменений
- Регулярный анализ
- Аудит

02. Сборка

- Контроль подписи
- Подписания артефактов
- Контроль сборки
- Безопасность CI/CD
- Доверенный репозиторий

03. Зависимости

- Доверенный репозиторий
- Подписание зависимостей
- OSA/SCA
- SBoM

Митигация угроз при заказной разработке

01. Исходный код

- Безопасный репозитория
- Анализ кода

Митигация угроз при заказной разработке

01. Исходный код

- Безопасный репозитория
- Анализ кода

02. Сборка

- Сборка у потребителя
- Контроль подписи/хеша
- Оценка изменений

Митигация угроз при заказной разработке

01. Исходный код

- Безопасный репозитория
- Анализ кода

02. Сборка

- Сборка у потребителя
- Контроль подписи/хеша
- Оценка изменений

03. Зависимости

- OSA/SCA
- SBoM

Митигация угроз при использовании стороннего ПО

Убедиться в безопасности не представляется возможным

Оценить выполнение требований ИБ при разработке ПО:

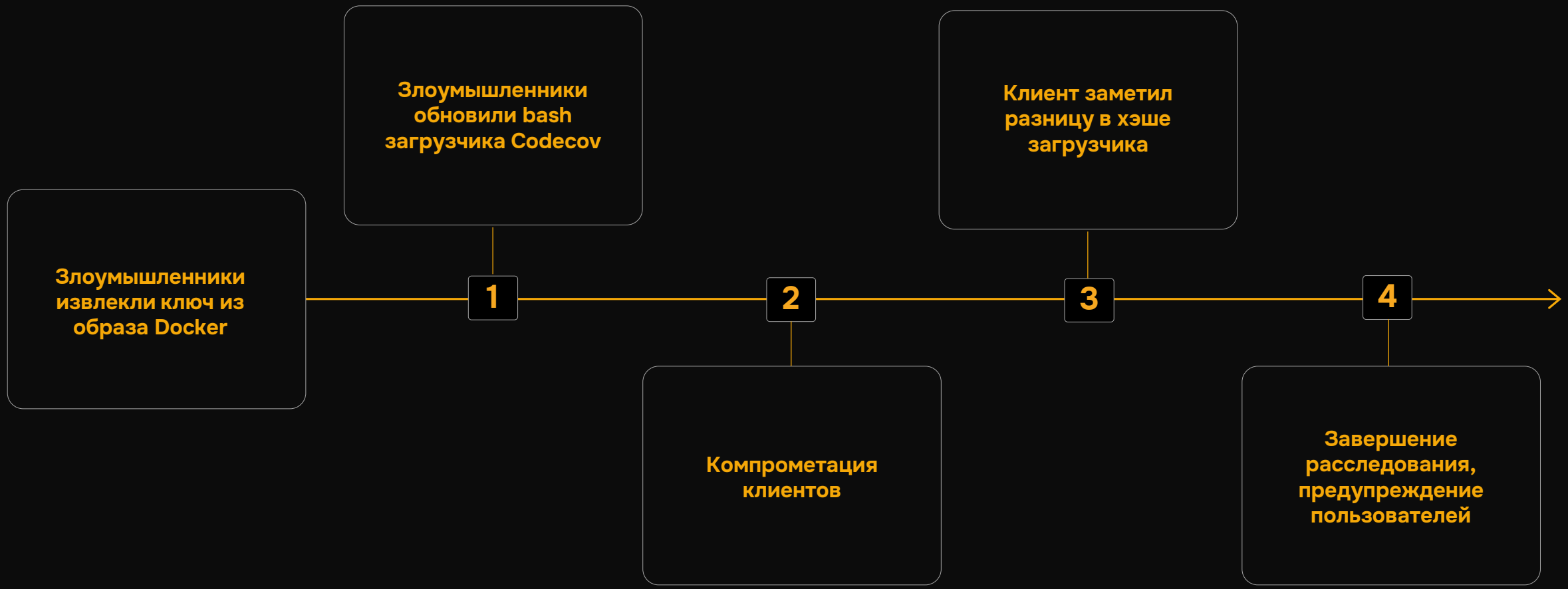
- Наличие практик безопасной разработки
- Подтверждение внедрения практики
- Регулярные проверки безопасности
- Соглашение об оповещении и устранении уязвимостей
- Наличие стороннего заключения
- Построение Zero Trust

Митигация угроз: общие рекомендации

> Общие рекомендации, которые не зависят от типа используемого ПО или процесса его разработки

- **Регулярно** проводить анализ защищенности черным/белом ящиком
- **Анализировать** события безопасности в течение всего процесса разработки ПО
- **Контролировать** удаленный и привилегированный доступ к среде разработки ПО
- **Собирать и анализировать** события безопасности с систем сборки и разработанного ПО
- **Выявлять** нетиповое поведение для ПО и пользователей

Codecov


Злоумышленники извлекли ключ из образа Docker
1
Злоумышленники обновили bash загрузчика Codecov
2
Компрометация клиентов
3
Клиент заметил разницу в хэше загрузчика
4
Завершение расследования, предупреждение пользователей

Действия разработчика

01. Исходный код

- Поиск чувствительной информации
- Утверждение изменений
- Подписание коммитов

02. Сборка

- Контроль подписи
- Подписания артефактов

03. Зависимости

- Подписание артефактов

Подписание артефактов - коммитов

Подписываем коммиты

(Конфигурация хранится в $HOME/.gitconfig)

```
$ git config --global user.signingKey XXXXX
$ git config --global commit.gpgsign true
$ git config --global tap.gpgsign true
```

Проверить коммит можно с помощью команды

> git verify-commit (и verifytag в случае с метками)

Подписание артефактов - коммитов

Подписываем коммиты
(Конфигурация хранится в $HOME/.gitconfig)

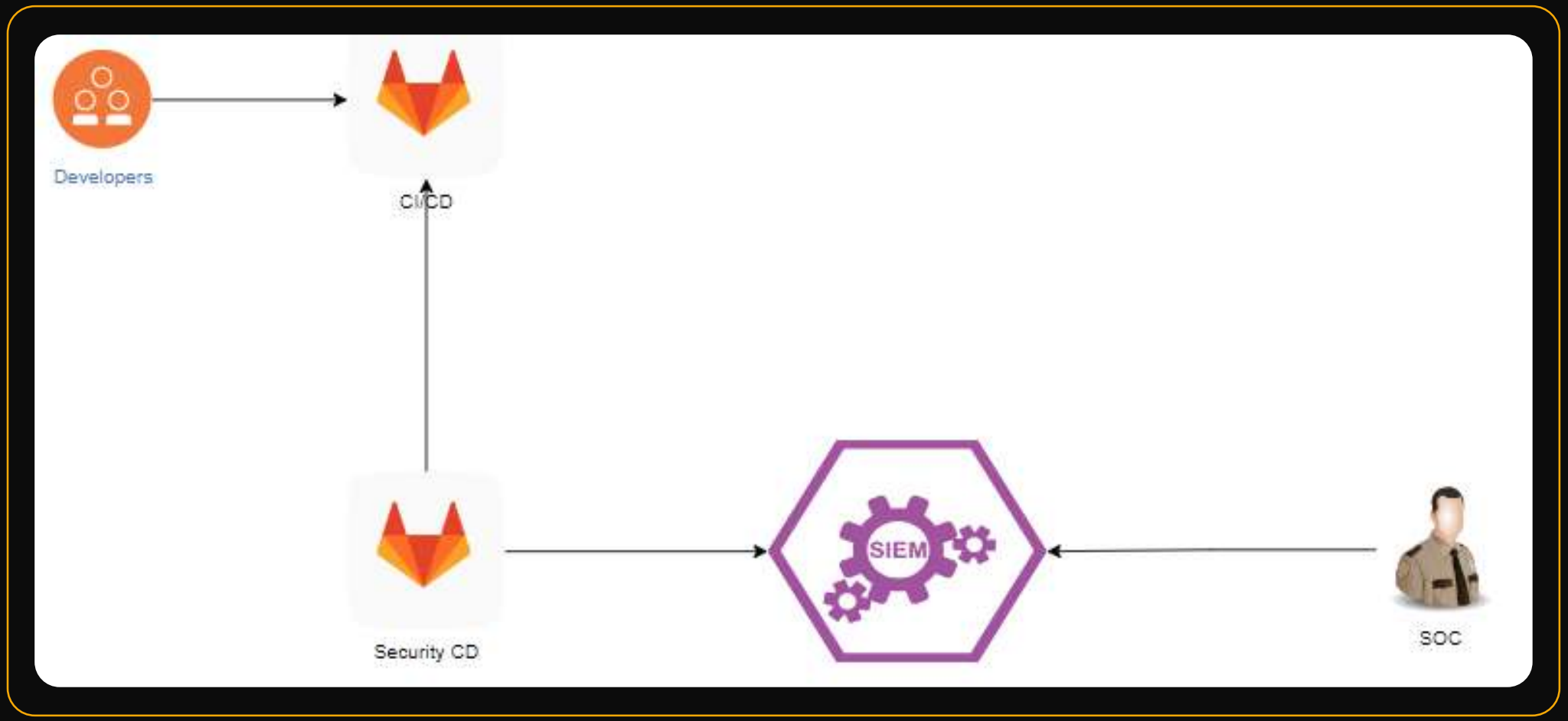
```
$ git config --global user.signingKey XXXXX
$ git config --global commit.gpgsign true
$ git config --global tap.gpgsign true
```

Проверить коммит можно с помощью команды
> git verify-commit (и verifytag в случае с метками)

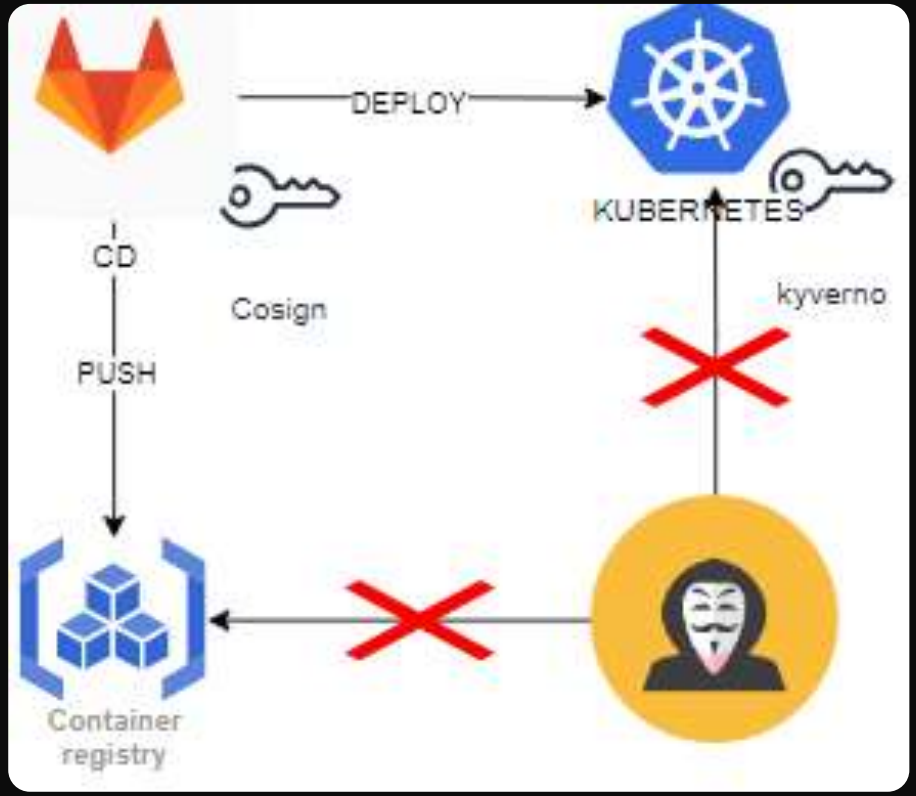
Для проверки можно написать скрипт, в котором:

1. указываем список доверенных сертификатов
2. проверяем все коммиты в репозитории, игнорируя их слияние
3. проверяем, есть ли подпись
4. проверяем, находится ли подпись в списке доверенных

Подписание артефактов - коммитов



Подписание артефактов - образов контейнеров



01. Для подписания используем утилиту Cosign

02. Подписанный образ складываем в репозиторий контейнеров

03. На стороне Kubernetes развернут Kyverno admission controller

04. Политика Kyverno проверяет подпись образа с известным ей публичным ключом

05. На основании политики деплой либо разрешается, либо отклоняется

06. Аудит событий K8s позволяет своевременно реагировать на события запрета деплоя

Действия клиента

Безопасность процесса разработки

- Zero trust
- Правильное хранение секретов
- Оценка мейнтейнеров
- Оценка репозитория
- Анализ кода
- Анализ коммитов

Безопасное хранение и управление секретами

```
{
"encoding": "base64",
"data": "TE9MIEtFSyBDSEVCVVJFSwo=",
"filename": "some.secret",
"mode": "400",
"owner": "root",
"group": "root",
}
```

Пример записи в Vault

Типы секретов

- Логин и пароль
- Токен доступа
- API-ключи
- Ключ сертификатов
- Ключ для подписи

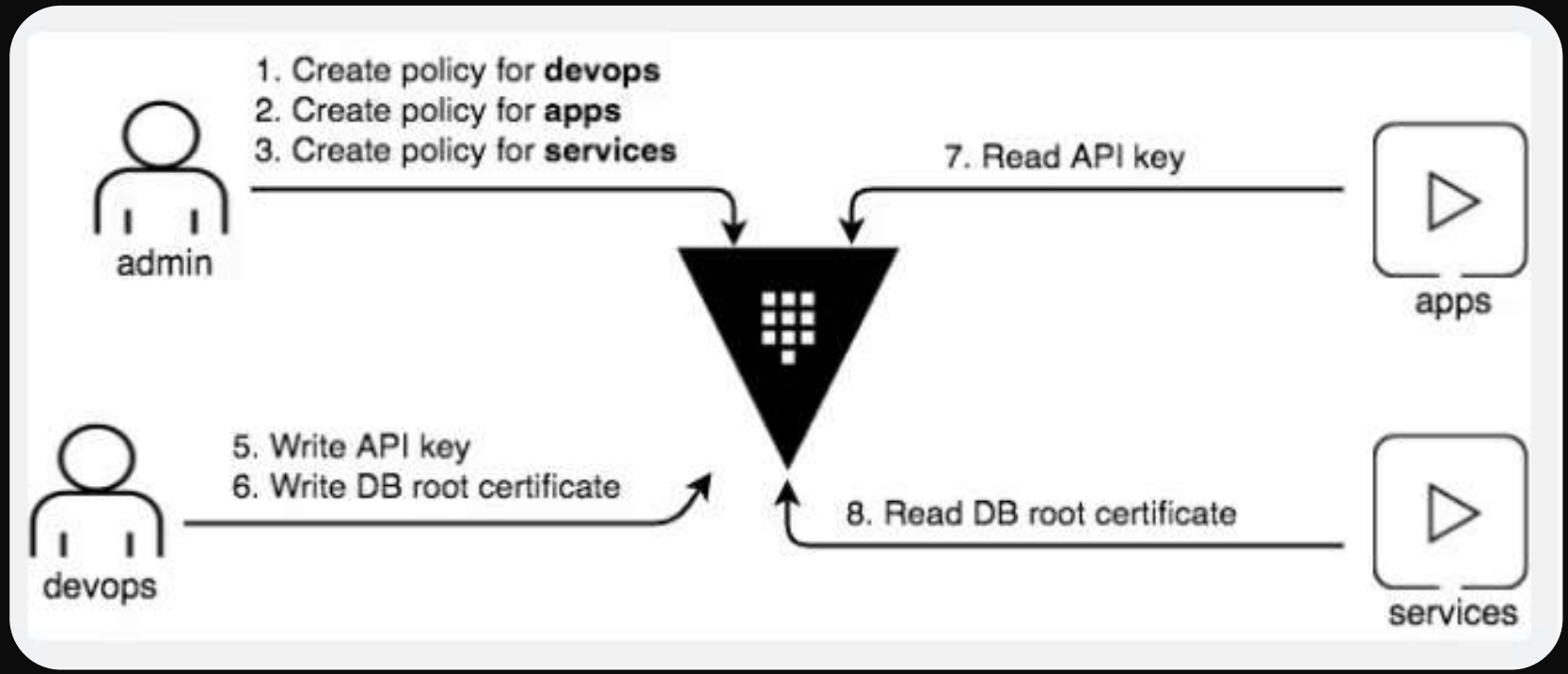
Использование Secret manager

ПОЗВОЛЯЕТ

- Управлять секретами как для пользователей, так и для приложений или сервисов
- Проводить автоматическую ротацию секретов
- Использовать собственный центр сертификации
- Применять одноразовые пароли и динамические секреты
- Управлять доступом к секретам
- Использовать собственные расширения, которые будут встроены в Vault

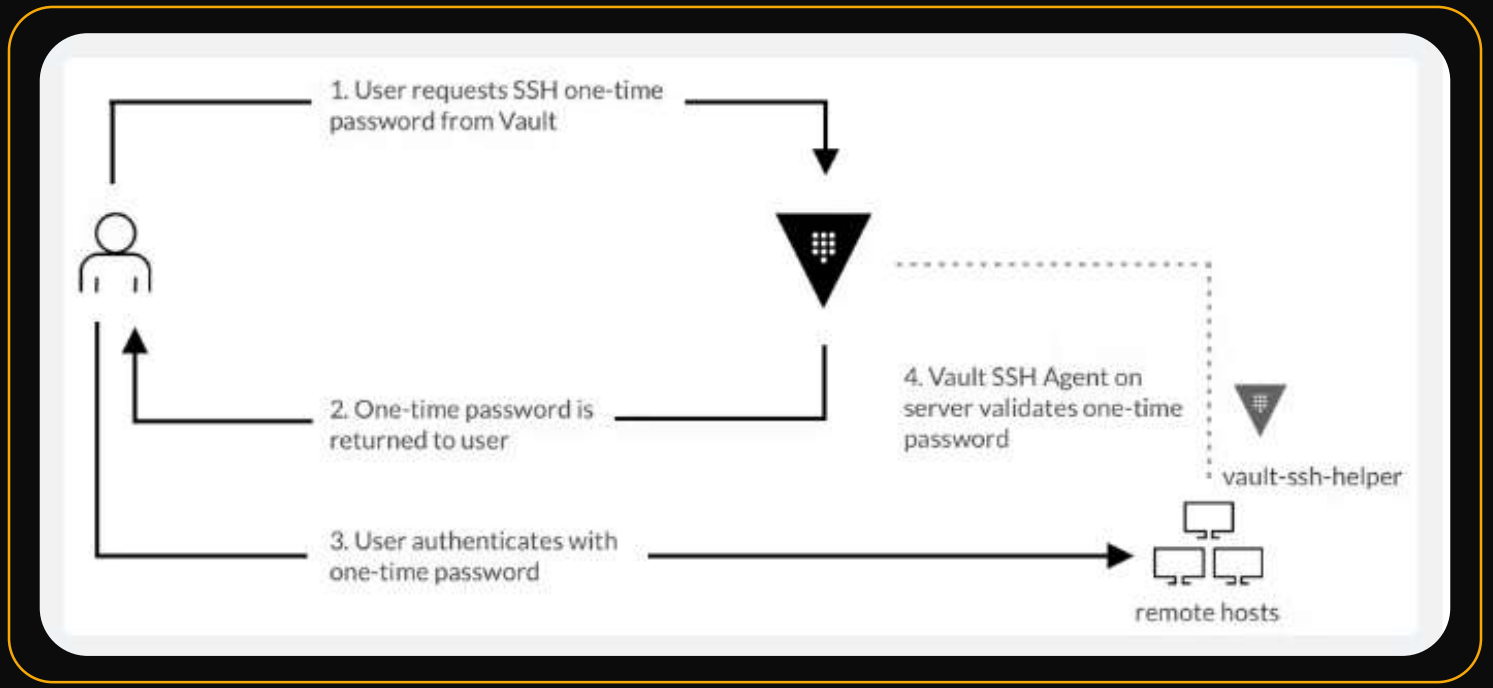
Vault – механизм использования

- Администратор создает структуру хранения ключей и политики доступа к ним
- Задачи по создание и управлению секретов можно делегировать владельцам систем
  Все остальные могут использовать хранимые секреты при наличии доступа



Vault – одноразовые SSH-ключи

- На сервере Vault создается роль с типом ключа OTP и политика доступа к нему
- Пользователь аутентифицируется на сервер Vault и получает OTP
- Полученный OTP используется для подключения к удаленному хосту Удаленный хост идет на сервер Vault, проверяется валидность OTP
- Если OTP подтвержден, то он удаляется из базы Vault



Проектирование Zero Trust архитектуры

01.

СЕГМЕНТАЦИЯ

- Разделяем разработчиков, devops, support, etc по разным сетевым сегментам
- Разделяем тестовый контур от продуктового
- Ограничиваем доступ к тестовым средам из Интернета

Проектирование Zero Trust архитектуры

01. СЕГМЕНТАЦИЯ

- Разделяем разработчиков, devops, support, etcпо разным сетевым сегментам
- Разделяем тестовый контур от продуктового
- Ограничиваем доступ к тестовым средам из Интернета

02. ПРАВА ДОСТУПА

- Предоставляем только необходимые права в зависимости от задач
- По возможности ограничиваем привилегированный доступ к продуктовой среде разработчикам
- Зависимости не должны использоваться напрямую из Интернета

Проектирование Zero Trust архитектуры

01. СЕГМЕНТАЦИЯ

- Разделяем разработчиков, devops, support, etcпо разным сетевым сегментам
- Разделяем тестовый контур от продуктового
- Ограничиваем доступ к тестовым средам из Интернета

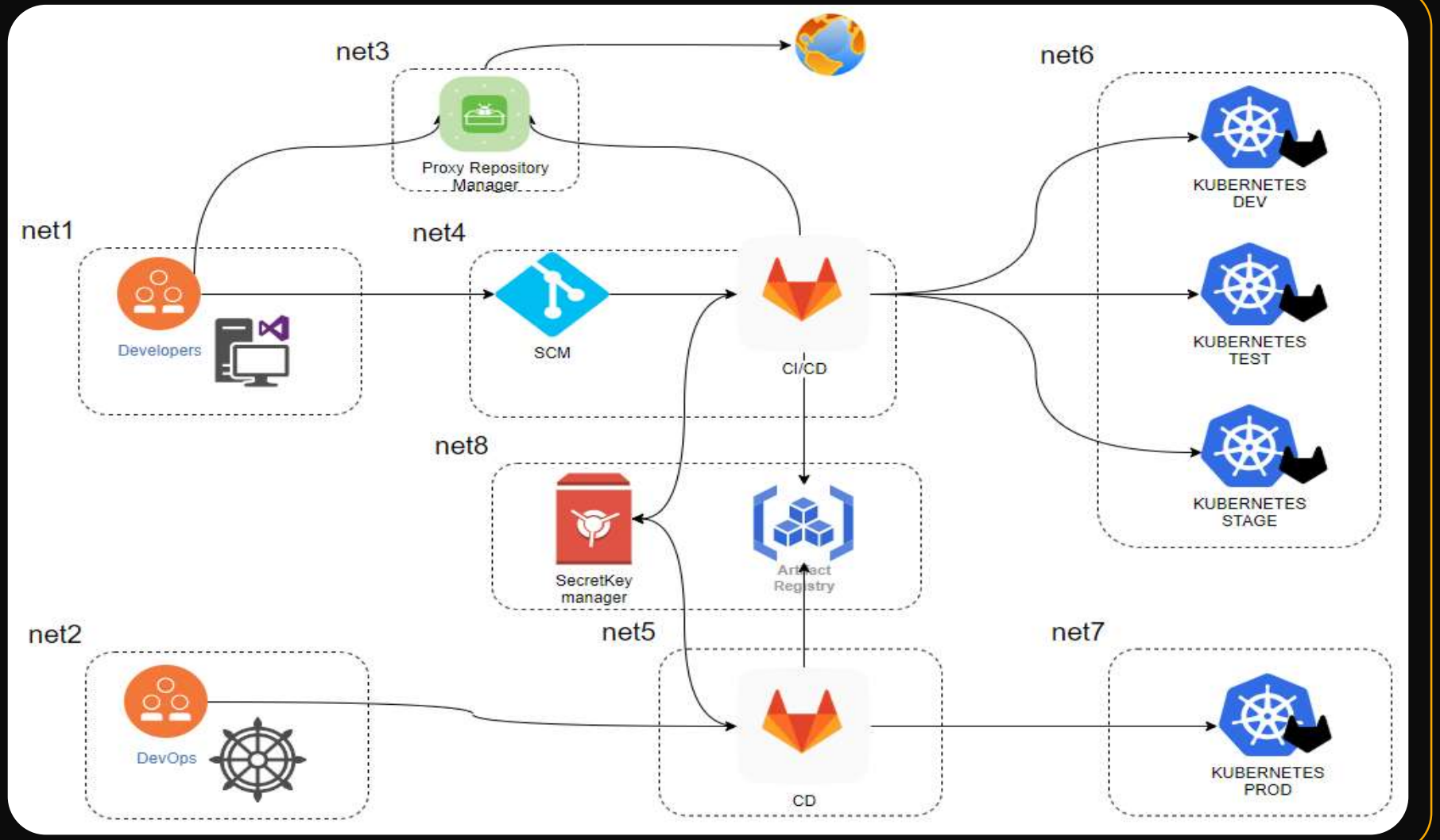
02. ПРАВА ДОСТУПА

- Предоставляем только необходимые права в зависимости от задач
- По возможности ограничиваем привилегированный доступ к продуктовой среде разработчикам
- Зависимости не должны использоваться напрямую из Интернета

03. ДАННЫЕ

- Не допускаем хранение реальных данных в тестовом контуре

Проектирование Zero Trust архитектуры



ЦЕНТР КИБЕРБЕЗОПАСНОСТИ

Евгений Тодышев

etodishev@ussc.ru

+7 (950) 555-68-90

@mr.appsec

sec.ussc.ru

cybersec@ussc.ru