GS LABS
GS GROUP

Публичный - L0

Сервис авторизации и проверки доступа

Руководство по установке

Индекс	2004-Shield.Int-IG
Секретность	Публичный - L0
Ревизия	1.0
Статус	Согласован
Подразделение	ДПРСУД
Компания	GS Labs

Содержание

1. Аннотация

Документ предназначен для технических специалистов, занимающихся установкой, настройкой и поддержкой **Сервиса авторизации и проверки доступа** (далее в документе используется условное наименование данного сервиса - **Shield**). Документ рассчитан на инженеров, обладающих специальными навыками и знаниями в области программного обеспечения.

⚠ **Данный документ опубликован исключительно с целью изучения системных требований для установки продукта, а также ознакомления с последовательностью и деталями процесса установки. Реальная установка продукта производится с использованием внутренних репозиториев ООО "Цифра", доступ к которым предоставляется заказчику по запросу.**

1.1. Справочная документация

№	Наименование документа, ссылка	Индекс документа (если есть)
1	Сервис авторизации и проверки доступа. Руководство администратора	2004-Shield.Int-AG
2	Сервис авторизации и проверки доступа. Техническое описание	2004-Shield.Int-TD

2. Минимальные системные требования

Для установки сервиса необходимо наличие не менее 3 серверов с разными именами (hostname): master, worker1, worker2. Общее количество серверов должно быть нечетным.

Серверы должны удовлетворять следующим требованиям:

1. Операционная система ubuntu-20.04-server-amd64 (с установленным пакетом sudo).
2. Центральный процессор с тактовой частотой каждого ядра 2 ГГц (минимум 4 ядра).
3. Объем оперативной памяти 4 ГБ.
4. Наличие дискового пространства не менее 40 Гб .
5. Два интерфейса Ethernet 100 и 1000 Base-T с поддерживаемой пропускной способностью 100 и 1000 Мбит/сек соответственно. Один предназначен для сети поддержки, второй используется для вывода генерируемого транспортного потока.

Установка должна производиться с дополнительного Ubuntu-сервера, не имеющего отношения к будущему кластеру. Требования к объему ресурсов дополнительного сервера отсутствуют.

Корректная работа сервиса гарантируется на версиях ОС Ubuntu 20.04

3. Установка и настройка баз данных

3.1. Подготовка баз данных

Для работы продукта требуется предварительно установить:

- Базу данных Postgresql
- KV хранилище Redis
- KV хранилище Etcd
- Систему сообщений NATS

Помимо их установки, необходимо произвести следующие действия:

1. В базе данных Postgresql создать базы данных и пользователей (с паролями). При этом потребуется обеспечить соответствие этих данных с указываемыми в default.yaml (см. раздел "Развёртывание внутри Kubernetes"). Например, создается база devices, пользователь devices с паролем devices. Аналогично для остальных сервисов.

   ⚠ Описание accounts, devices и domains приведено в документе [2], в гл. "Описание БД" (не является публичной информацией).

2. Требуется выполнить *./shield id set default id-provider:8080* для начального наполнения Etcd под сервис shield.

4. Развёртывание сервиса в кластере kubernetes

Кластер развёртывается по официальной инструкции (https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/high-availability/).

Для установки сервиса в имеющийся настроенный кластер Kubernetes используется процесс CI/CD, настраиваемый с помощью GitLab.

Все действия возможно производить на локальной машине или на любом Ubuntu-сервере с доступом через консоль от имени любого пользователя.

4.1. Требования к окружению

Для развёртывания сервиса нужно соблюсти следующие требования:

1. Развернуть высокодоступный Kubernetes кластер
2. Развернуть высокодоступный PostgreSQL
3. Развернуть кластер NATS с использованием nats-operator
4. Развернуть etcd кластер

4.1.1. Требования к NATS

Для работы продукта необходим кластер NATS, развернутый при помощи nats-operator.
Пример helmfile для установки оператора в кластер:

nats-operator.yaml

```
helmDefaults:
  kubeContext: integration

repositories:
  - name: geek-cookbook
    url: https://geek-cookbook.github.io/charts

releases:
  - name: nats-operator
    namespace: data
    chart: geek-cookbook/nats-operator
    version: 0.1.3
    labels:
      app: nats-operator
      level: infra
    values:
    - cluster:
        auth:
          enabled: false
        metrics:
          enabled: true
```

4.1.2. Требования к etcd

Для работы продукта необходим кластер etcd.
Пример helmfile для для установки etcd в кластер:

etcd.yaml

```
helmDefaults:
  kubeContext: integration

repositories:
  - name: stable
    url: https://kubernetes-charts.storage.googleapis.com/
  - name: bitnami
    url: https://charts.bitnami.com/bitnami

releases:
  - name: etcd
    namespace: data
    chart: bitnami/etcd
    version: 4.8.0
    labels:
      app: etcd
      level: infra
    values:
    - statefulset:
        replicaCount: 3
      auth:
        rbac:
          enabled: false
      fullnameOverride: "etcd"
```

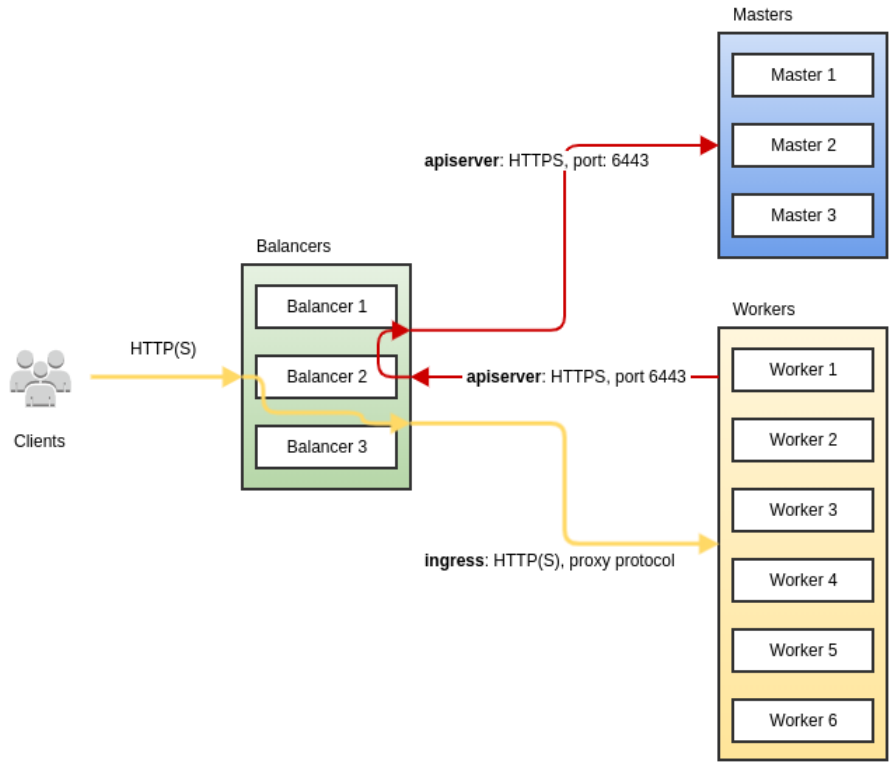
4.1.3. Распределение узлов

Необходимо разделить узлы логически на две роли:

Роль	Компоненты	Пример распределения ресурсов	Требования к количеству узлов	Требования к физическому местоположению	Резервное копирование
Master	• apiserver • control plane • etcd	• 2 CPU cores • 2GB RAM	1. Не менее трёх 2. Нечётное количество	Распределить по физическим машинам. Рекомендуется 1 физическая машина - 1 master нода.	Каждый час - полный snapshot etcd. (Документация https://etcd.io/docs/v3.3.12/op-guide/recovery/)
Worker	• kubelet	• 4 CPU cores • 16GB RAM	1. Не менее двух	Распределить по физическим машинам. Рекомендуется 1 физическая машина - 2 worker-ноды.	Не требуется

4.1.4. Общая схема

Перед Kubernetes кластером необходимо разместить балансировщики на базе haproxy для балансировки как внутреннего трафика kubernetes (worker<->master), так и внешнего (ingress). Балансировщики должны иметь общий Virtual IP адрес, который настраивается через keepalived. Этот адрес должен использоваться как адрес мастера при создании кластера. Таким образом, в случае отказа мастера, трафик с воркеров будет успешно достигать одного из мастеров. Схематически процесс изображен ниже:



4.1.5. Балансировка

Конфигурация haproxy (на тестовом кластере, реальные параметры подбираются исходя из доступных вычислительных мощностей):

- 3 Master узла
- 6 Worker узлов
- 3 балансера

4.1.5.1. Входящий трафик

Во внешнюю сеть должны быть открыты только эти порты:

- 80;
- 443;

В тестовых средах дополнительно может быть открыт порт 6443 для работы с kubernetes кластером.

4.2. Развёртывание внутри Kubernetes

Ссылка на проект для развертывания Shield в kubernetes
предоставляется заказчику по запросу.

4.2.1. Состав репозитория

- docs - документация по каждому Helm Chart, который устанавливается в систему
- templates - конфигурационные шаблоны helmfile для генерации values.yaml файлов для каждого helm chart
- helmfile.yaml - конфиг helmfile
- default.yaml - значения по умолчанию
- versions.gen.yaml - файл, содержащий последние стабильные версии сервисов

4.2.2. Развертывание системы

Чтобы развернуть shield, нужно:

1. Создать отдельный проект в Gitlab
2. Настроить данный проект либо как субмодуль, либо с использованием средств автоматизации (Makefile)
3. В проекте среды создать helmfile.yaml с содержимым:
   helmfiles:
   - path: <путь до субмодуля>/helmfile.yaml values:
   - <путь до субмодуля>/versions.gen.yaml # Загружаем последние стабильные версии сервисов (можно зафиксировать версии, если скопировать файл в свой репозиторий)
   - <путь до субмодуля>/default.yaml # Загружаем значения по умолчанию
   - production.yaml # Применяем собственную конфигурацию

4.2.3. Создание сущности Realm по умолчанию

⚠ Процедура выполняется после успешного развертывания shield в kubernetes.

При развёртывании shield в самом сервисе Shield не будет ни одной сущности Realm, даже по умолчанию.

В связи с этим нужно указать Realm по умолчанию с помощью команды:

` *kubectl -n shield_namespace exec shield-код_конкретного_пода – ./shield id set default id-provider:8080* `

4.2.4. Настройка конфигурации

Информация о конфигурации сервисов приведена в Руководстве администратора.