Руководство по установке

Компонента Сервис трассировки (TRAS)

Продукта Platform V Synapse Service Mesh (SSM)

ОГЛАВЛЕНИЕ

Руководство по установке

Термины и определения

Термин/Аббревиатура	Определение
API	Application Programming Interface, программный интерфейс приложения
URL	Uniform Resource Locator, унифицированный адрес ресурса
Трейс	Цепочка единичных операций (span), представляющих часть потока выполнения запроса. Трейс позволяет визуализировать активность запроса при его перемещении по системе и является самостоятельной единицей трассировки
Платформа	Платформа оркестрации приложений с средствами автоматизации и управления на основе политик, например Kubernetes
Platform V Synapse Service Mesh	Программный продукт на базе Istio SE, обеспечивающий возможность создания сервисной сети поверх Платформенной в Kubernetes
Контрольная панель	Проект, где запущены управляющие приложения Synapse Service Mesh (компонент POLM)
Zipkin хранилище	Внешняя система хранения и отображения данных трассировки совместимая с форматом Zipkin
Управление политиками / POLM	Компонент Управление политиками из состава продукта Platform V Synapse Service Mesh

Термин/Аббревиатура	Определение
Граничный прокси / IGEG	Компонент Граничный прокси продукта Platform V Synapse Service Mesh
Сервисный прокси / SVPX	Компонент Сервисный прокси продукта Platform V Synapse Service Mesh
Сервис трассировки / TRAS	Компонент Сервис трассировки продукта Platform V Synapse Service Mesh

Системные требования

Для функционирования компонента TRAS предъявляются требования по наличию следующего программного обеспечения:

Наименование ПО	Версия ПО	Назначение ПО
Kubernetes	1.19 и выше	Платформа оркестрации приложений с средствами автоматизации и управления на основе политик
Zipkin хранилище	Любая	Внешняя система хранения и отображения данных трассировки совместимая с форматом Zipkin

Для возможности получения трейсов компонентом TRAS рекомендуется развернуть следующие компоненты продукта Platform V

Наименование компонента Platform V	Код компонента Platform V	Назначение компонента Platform V
Управление политиками	POLM	Управление политиками из состава продукта Platform V Synapse Service Mesh формирует конфигурации компонент сервисного и граничного прокси. Данный

Наименование компонента Platform V	Код компонента Platform V	Назначение компонента Platform V
		компонент настраивает политики отправки трейсов в сервисном и граничном прокси
Граничный прокси	IGEG	Граничный прокси из состава продукта Platform V Synapse Service Mesh, предназначен для обеспечения управляемого вызова интеграционных сервисов прикладной части в проекте Kubernetes. Отправлет спаны в компонент TRAS
Сервисный прокси	SVPX	Сервисный прокси из состава продукта Platform V Synapse Service Mesh (далее Synapse) предназначен для предоставления базовых интеграционных операций прикладной части интеграционного сценария-сервиса. Сервисный прокси используется для маршрутизации и обеспечения безопасности трафика между приложениями. Отправлет спаны в компонент TRAS

Перед установкой проверьте соблюдение следующих условий:

- Развернутый и настроенный кластер Платформы (Kubernetes 1.19 и выше)
- В кластере создан проект для развертывания контрольной панели Platform V Synapse Service Mesh
- В проекте создана учетная запись с правами на загрузку артефактов (администратор проекта).
- Свободные в проекте ресурсы, по лимитам и реквестам по размеру равные или превышающие зарезервированные в конфигурационных артефактах продукта Platform V Synapse Service Mesh.

Минимальные ресурсы

Минимальные ресурсы, необходимые для компонента TRAS:

Контейнер	CPU Request	Memory Request	CPU Limit	Memory Limit
synapse-tracer	200	200	200	200

Количество задействованных реплик компонента TRAS зависит от объема принимаемых данных.

Одна реплика обрабатывает до 700 сообщений в секунду. (Один трейс может содержать от 2 до 12 сообщений)

Требования к КТС

- **Требования к техническим компонентам**

Требования к КТС — это размер ресурсов CPU, памяти, выделяемых в Kubernetes. Их значение вычисляется для конкретного разворачиваемого инстанса в зависимости от профиля нагрузки.

- **Принципы размещения сервиса на КТС**

Компонент TRAS разворачивается в виде контейнера в отдельном Деплойменте.

- **Специфические технологические решения**

Отсутствуют.

Установка

Состав дистрибутива

Дистрибутив компоненты TRAS содержит следующие файлы:

- Deployment.yaml (synapse-tracer-v2)
- ConfigMap.yaml (tracerconfig)
- Service.yaml (zipkin)
- Service.yaml (jaeger-collector)
- EnvoyFilter.yaml (filter-custom-tag-traice)

Deployment

```
- kind: Deployment
    apiVersion: apps/v1
    metadata:
      name: synapse-tracer-v2
      labels:
        app: synapse-tracer
    spec:
      replicas: <количество реплик>
      selector:
        matchLabels:
          app: synapse-tracer
      template:
        metadata:
          labels:
            app: synapse-tracer
          annotations:
            sidecar.istio.io/inject: 'false'
        spec:
          volumes:
            - name: tracerconfig
```

```
            configMap:
              name: tracerconfig
              defaultMode: 0400
          - name: tengrytrust
            secret:
              secretName: tengrytrust
              defaultMode: 0400
        containers:
          - name: synapse-tracer
            image:  <ссылка на реджестри>rhel7-java-synapse-ci03227168_synapse-tracing-
ci03227168:<версия>
            ports:
              - name: http
                containerPort: 8080
                protocol: TCP
            resources:
              limits:
                cpu: 200m
                memory: 200Mi
              requests:
                cpu: 200m
                memory: 200Mi
            livenessProbe:
              httpGet:
                path: /health
                port: 8080
                scheme: HTTP
              initialDelaySeconds: 30
              timeoutSeconds: 3
              periodSeconds: 10
              successThreshold: 1
              failureThreshold: 3
            readinessProbe:
              httpGet:
                path: /health
                port: 8080
                scheme: HTTP
              initialDelaySeconds: 10
              timeoutSeconds: 3
              periodSeconds: 10
              successThreshold: 1
              failureThreshold: 3
            securityContext:
              readOnlyRootFilesystem: true
            volumeMounts:
              - name: tracerconfig
                readOnly: true
                mountPath: /opt/synapsetracer/configs
              - name: tengrytrust
                readOnly: true
                mountPath: /opt/synapsetracer/certs
            terminationMessagePath: /dev/termination-log
            terminationMessagePolicy: File
            imagePullPolicy: Always
        restartPolicy: Always
        terminationGracePeriodSeconds: 30
        dnsPolicy: ClusterFirst
        securityContext: {}
        affinity:
          podAntiAffinity:
            preferredDuringSchedulingIgnoredDuringExecution:
              - weight: 100
                podAffinityTerm:
                  labelSelector:
                    matchExpressions:
                      - key: app
```

```
                        operator: In
                        values:
                          - synapse-tracer
                    #перечень namespace, в которых может быть установлен сервис трассировки
                    namespaces: ["control-panel1", "control-panel2"]
                    topologyKey: kubernetes.io/hostname
          schedulerName: default-scheduler
```

Configmap

```
- kind: ConfigMap
    apiVersion: v1
    metadata:
      name: tracerconfig
    data:
      tracerconfig.json: >
        {"port":":8080",
        "host":"<URL Zipkin хранилища>",
        "clientTimeout":500,
        "spanBuffer":30,
        "control_panel_tag":"имя namespace в котором устанавливается компонент",
        "cluster_name_tag": "имя кластера платформы",
        "tls_server_uncheck": false,
        "is_timeout_on": true,
        "is_retry_on": true,
        "retry_count": 2,
        "retry_wait_time": 100,
        "log_time": 20,
        "client_max_conn": 30,
        "server_max_request_per_conn": 50}
      tracerconfigfilter.json: >
        {"tags":
         [
           #трейсы в который присутствуют теги с такими значениями, будут отброшены
           {"tag":":authority", "value": "db-service:50003"},
           {"tag":":path", "value": "/metrics"},
           {"tag":":path", "value": "/actuator/health"}
         ]}
```

Service Zipkin

```
- kind: Service
    apiVersion: v1
    metadata:
      name: zipkin
      labels:
        app: synapse-tracer
    spec:
      ports:
        - name: http
          port: 8080
          protocol: TCP
          targetPort: 8080
        - name: http-zipkin
          protocol: TCP
          port: 9411
          targetPort: 8080
      selector:
        app: synapse-tracer
```

Service Jaeger-collector

```
- kind: Service
    apiVersion: v1
```

```
    metadata:
      name: jaeger-collector
      labels:
        app: synapse-tracer
    spec:
      ports:
        - name: http
          port: 8080
          protocol: TCP
          targetPort: 8080
        - name: http-zipkin
          protocol: TCP
          port: 9411
          targetPort: 8080
      selector:
        app: synapse-tracer
```

EnvoyFilter

```
- kind: EnvoyFilter
    apiVersion: networking.istio.io/v1alpha3
    metadata:
      name: filter-custom-tag-traice
    spec:
      configPatches:
        - applyTo: NETWORK_FILTER
          match:
            context: ANY
            listener:
              filterChain:
                filter:
                  name: envoy.http_connection_manager
          patch:
            operation: MERGE
            value:
              typed_config:
                '@type': >-

type.googleapis.com/envoy.config.filter.network.http_connection_manager.v2.HttpConnectionManage
r
                tracing:
                  request_headers_for_tags:
                    - from
                    - host
                    - ':authority'
                    - ':path'
                    - ':method'
                    - ':scheme'
                    - content-type
                    - http_host
                    - grpc-accept-encoding
                    - x-forwarded-proto
                    - x-request-id
                    - x-envoy-decorator-operation
                    - x-forwarded-for
                    - x-envoy-internal
                    - x-b3-traceid
                    - x-b3-spanid
                    - x-b3-parentspanid
                    - x-b3-sampled
                    - x-synapse-status-code
                    - x-synapse-messageid
                    - x-synapse-corellationid
                    - x-synapse-rquid
                    - x-synapse-rqtm
```

```
    - x-synapse-spname
    - x-synapse-scname
    - x-synapse-operationname
    - x-synapse-from-pod-name
    - x-synapse-serviceversion
    - x-synapse-custom
```

Сертификат в дистрибутиве для установки TRAS не содержится и его необходимо выпустить самостоятельно. См. Раздел "Управление ключами и сертификатами" в "Руководстве по безопасности".

Для установки компонента выполните следующие действия:

1. Создайте директорию установки.

Шаг	Действия	Описание
Создайте директорию для установки.	Создайте папку. *Пример: synapse_tracer*.	
Разархивируйте файлы.	Распакуйте в созданную папку архив с конфигурационными артефактами.	

1. Подключитесь к проекту.

Для подключения через веб-интерфейс Платформы:

Шаг	Действия	Описание
Авторизуйтесь в веб-консоли Платформы	Перейдите по ссылке URL в веб-консоли нужного кластера Платформы. Введите в окне ввода учетных данных логин и пароль.	
Перейдите в проект	Выберите пункт меню **Home/Projects** и выберите из списка проект.	

Для подключения через консоль Платформы:

Шаг	Действия	Описание

Шаг	Действия	Описание
Войдите в консольного клиента платформы.	В окне командной строки в приглашении введите команды: kubectl config set-credentials /<host-alias-без-точек>: --token= kubectl config set-cluster <host-alias-без-точек>: --insecure-skip-tls-verify=true --server=https://: kubectl config set-context /<host-alias-без-точек>:/ --user=/<host-alias-без-точек>: --namespace=maximov-test --cluster=<host-alias-без-точек>: kubectl config use-context /<host-alias-без-точек>:/	

1. Установите трейсер.

Для установки через веб-интерфейс Платформы:

Шаг	Действия	Описание
Загрузите секрет с ключами	В правом верхнем углу окна проекта нажмите иконку с изображением значка + Import YAML. Перенесите, используя механизм drag and drop, файл с секретом из директории, содержащей конфигурационные артефакты. В открывшемся окне редактирования нажмите Create.	Артефакт, содержащий файлы ключей и сертификатов в формате pem
Загрузите артефакт, содержащий файлы ключей и сертификатов в формате pem.	В правом верхнем углу окна проекта нажмите иконку с изображением значка + Import YAML. Перенесите, используя механизм drag and drop, файл с секретом из директории, содержащей конфигурационные артефакты. В открывшемся окне редактирования нажмите Create.	Артефакт, содержащий Deployment.yaml
Загрузите артефакты из файла Configmap.yaml	В правом верхнем углу окна проекта нажмите иконку с изображением значка **+ Import YAML**. Перенесите, используя механизм drag and drop, файл с config map из директории, содержащей конфигурационные артефакты.	Артефакт, содержащий Configmap.yaml.

Шаг	Действия	Описание
	В открывшемся окне редактирования нажмите **Create**.	
Загрузите артефакты из файла Service-zipkin.yaml	В правом верхнем углу окна проекта нажмите иконку с изображением значка **+ Import YAML**. Перенесите, используя механизм drag and drop, файл с Service из директории, содержащей конфигурационные артефакты. В открывшемся окне редактирования нажмите **Create**.	Артефакт, содержащий Service-zipkin.yaml.
Загрузите артефакты из файла Service-jaeger-collector yaml	В правом верхнем углу окна проекта нажмите иконку с изображением значка **+ Import YAML**. Перенесите, используя механизм drag and drop, файл с Service из директории, содержащей конфигурационные артефакты. В открывшемся окне редактирования нажмите **Create**.	Артефакт, содержащий Service-jaeger-collector.yaml
Загрузите артефакты из файла EnoyFilter.yaml	В правом верхнем углу окна проекта нажмите иконку с изображением значка **+ Import YAML**. Перенесите, используя механизм drag and drop, файл с EnvoyFilter из директории, содержащей конфигурационные артефакты. В открывшемся окне редактирования нажмите **Create**.	Артефакт, содержащий EnvoyFilter.yaml

Обновление

Для обновления выполните следующие шаги:

1. Удалите действующую конфигурацию и загрузите новую версию.
2. Создайте директорию установки.

Шаг	Действия
Создайте директорию для	Создайте папку. *Пример: synapse_tracer*.

Шаг	Действия
установки.	
Разархивируйте файлы.	Распакуйте в созданную папку архив с конфигурационными артефактами новой версии трейсера.

1. Подключитесь к проекту.

Для подключения через веб-интерфейс Платформы:

Шаг	Действия
Авторизуйтесь в веб-консоли Платформы.	Перейдите по ссылке URL в веб-консоли кластера Платформы. В окне ввода учетных данных введите логин и пароль.
Перейдите в проект.	Выберите пункт меню **Projects** и выберите проект из списка.

Для подключения через консоль Платформы:

Шаг	Действия
Войдите в консольного клиента платформы	В окне командной строки в приглашении введите команды: kubectl config set-credentials /<host-alias-без-точек>: --token= kubectl config set-cluster <host-alias-без-точек>: --insecure-skip-tls-verify=true --server=https://: kubectl config set-context /<host-alias-без-точек>:/ --user=/<host-alias-без-точек>: --namespace=maximov-test --cluster=<host-alias-без-точек>: kubectl config use-context /<host-alias-без-точек>:/

1. Остановите старую версию.

Для остановки через веб-интерфейс Платформы:

Шаг	Действие
Остановка Пода	Выберите в меню **Workload/Deployments**. На странице найдите Deployment. При необходимости воспользуйтесь поиском по имени. Пройдите по ссылке в наименовании на вкладку **Detail**. Стрелкой вниз (**Decrease the pod count**) уменьшите количества Подов до 0.

Для остановки через консоль Платформы:

Шаг	Действие
Остановка	В консоли выполните команду: `kubectl scale --replicas=0 deployment/<имя Деплоймента>`

1. Сохраните артефакты действующей версии.

Для сохранения через веб-интерфейс Платформы

Шаг	Действия
Сохраните Деплоймент	В меню выберите пункт **Workload/Deployments**. На странице найдите **Deployment**. При необходимости воспользуйтесь поиском по имени. Пройдите по ссылке в наименовании на вкладку **Detail** → **YAML**. Нажмите **Download**. Файл с наименованием `deployment-<имя Деплоймента>.yaml` загрузится на рабочее место.
Сохраните конфигурацию.	В меню выберите пункт **Workload/Config Maps**. На странице найдите артефакт. При необходимости воспользуйтесь поиском по имени. Пройдите по ссылке в наименовании на вкладку **Detail** → **YAML**. Нажмите **Download**. Файл с наименованием `configmaps-<имя config map>.yaml` загрузится на рабочее место.
Сохраните сервисы	В меню выберите пункт **Networking/Services**. На странице найдите артефакт. При необходимости воспользуйтесь поиском по имени. Пройдите по ссылке в наименовании на вкладку **Detail** → **YAML**. Нажмите **Download**. Файл с наименованием `<имя service>.yaml` загрузится на рабочее место.
Сохраните envoyFilter	В меню выберите пункт **Home/Search**. На странице в поле "Resourses" выберете EnvoyFilter. Выберете необходимый артефакт из предложенных. При необходимости

Шаг	Действия
	воспользуйтесь поиском по имени. Пройдите по ссылке в наименовании на вкладку **Detail** → **YAML**. Нажмите **Download**. Файл с наименованием `<имя envoyFilter>.yaml` загрузится на рабочее место.

Для сохранения через консоль Kubernetes:

Шаг	Действия	Описание
Сохраните Деплоймент.	В консоли выполните команду: `kubectl get -o yaml deployment/<имя Деплоймента> > <путь к файлу>.yaml`	
Сохраните конфигурацию.	В консоли выполните команду: `kubectl get -o yaml configmaps/<имя config map> > <путь к файлу>.yaml`	
Сохраните сервисы	В консоли выполните команду: `kubectl get -o yaml services/<имя service> > <путь к файлу>.yaml`	
Сохраните envoyFilter	В консоли выполните команду: `kubectl get -o yaml envoyFilters/<имя envoyFilter> > <путь к файлу>.yaml`	

1. Загрузите новую версию.

Для загрузки через веб-интерфейс Платформы:

Шаг	Действия	Описание
Загрузите артефакты из файла Deployment.yaml.	В правом верхнем углу окна проекта нажмите иконку с изображением значка + Import YAML. Перенесите, используя механизм drag and drop, файл с секретом из директории, содержащей конфигурационные артефакты. В открывшемся окне редактирования нажмите Create.	Артефакт, содержащий Deployment.yml.

Шаг	Действия	Описание
Загрузите артефакты из файла Configmap.yaml.	В правом верхнем углу окна проекта нажмите иконку с изображением значка **+ Import YAML**. Перенесите, используя механизм drag and drop, файл с секретом из директории, содержащей конфигурационные артефакты. В открывшемся окне редактирования нажмите **Create**.	Артефакт, содержащий Configmap.yaml.
Загрузите артефакты из файла Service-zipkin.yaml.	В правом верхнем углу окна проекта нажмите иконку с изображением значка **+ Import YAML**. Перенесите, используя механизм drag and drop, файл с секретом из директории, содержащей конфигурационные артефакты. В открывшемся окне редактирования нажмите **Create**.	Артефакт, содержащий Service-zipkin.yml.
Загрузите артефакты из файла Service-jaeger-collector.yaml.	В правом верхнем углу окна проекта нажмите иконку с изображением значка **+ Import YAML**. Перенесите, используя механизм drag and drop, файл с секретом из директории, содержащей конфигурационные артефакты. В открывшемся окне редактирования нажмите **Create**.	Артефакт, содержащий Service-jaeger-collector.yml.
Загрузите артефакты из файла EnoyFilter.yaml.	В правом верхнем углу окна проекта нажмите иконку с изображением значка **+ Import YAML**. Перенесите, используя механизм drag and drop, файл с секретом из директории, содержащей конфигурационные артефакты. В открывшемся окне редактирования нажмите **Create**.	Артефакт, содержащий EnvoyFilter.yml.

Для загрузки через консоль Платформы:

Шаг	Действия	Описание
Загрузите артефакт Платформы Deployment	В консоли выполните команду: `kubectl apply -f Deployment.yml`	Артефакт, содержащий Deployment.yml.
Загрузите артефакт	В консоли выполните	Артефакт, содержащий

Шаг	Действия	Описание
Платформы Configmap	команду: `kubectl apply -f Configmap.yml`	Configmap.yaml.
Загрузите артефакт Платформы Service Zipkin	В консоли выполните команду: `kubectl apply -f Service-zipkin.yml`	Артефакт, содержащий Service-zipkin.yml.
Загрузите артефакт Платформы Service Jaeger-collector	В консоли выполните команду: `kubectl apply -f Service-jaeger-collector.yml`	Артефакт, содержащий Service-jaeger-collector.yml.
Загрузите артефакт Платформы EnvoyFilter	В консоли выполните команду: `kubectl apply -f EnvoyFilter.yml`	Артефакт, содержащий EnvoyFilter.yml.

Проверка работоспособности

Чеклист:

1. Проверка корректности установки:
   а) В меню выберите пункт **Workload/Deployments**.
   б) На вкладке **Pods** найдите поды с именем Synapse-tracer-v2-*, убедитесь что количество реплик соответствует указанному в переменных и статус подов имеет значение "Running". По умолчанию должен подняться 1 под. с) На вкладке YAML, по структуре Spec/template/spec/containers/image, убедитесь что ссылка на имидж соответствует указанной в файле Deployment.yaml.
2. Проверка корректности работы:
   а) На вкладке **Pods** выберите один из подов и кликните правой кнопкой мыши.
   б) На открывшейся странице выберите вкладку **Logs** , убедитесь что отсутствуют ошибки в консольном выводе.
   в) Перейдите на вкладку Terminal, в открывшейся консоли введите комманду: curl localhost:9192/loglevel -d "debug"
   г) Вернитесь на вкладку Logs - в консольном выводе должна появиться информация о принимаемых и отправляемых трейсах
3. Проверка корректности отображения трейсов
   а) Перейдите в вебинтерфейс Zipkin хранилища
   б) Произведите поиск по тегу synapseTracer и значению synapseTracer*
   в) Убедитесь в наличии трейсов с таким тегом

Откат

1. Произведите действия описанные в пункте 4 раздела Обновление
2. Произведите действия описанные в пункте 6 раздела Обновление с файлами полученными в результате выполнения пункта 5 раздела Обновление.

Часто встречающиеся проблемы и пути их устранения

Проблема	Решение
В логах подов встречаются ошибки типа: 2022-05-20T10:03:54.718Z warn non 200 response from collector: no healthy upstream	Убедитесь, что эндпойнт Zipkin хранилища, настроен и готов к работе, убедитесь в наличии физического доступа до эндпойнта из кластера Платформы
Не стартуют поды	Проверьте события во вкладке Event и в разделах Status/Yaml на страницах Deployment ReplicaSet и Pod. Устраните ошибки согласно описанию найденых ошибок
Отсутствует информация о трейсах во вкладке Log в режиме debug	Убедитесь, что в проектах подключенных к контрольной панели Synapse Service Mesh есть трафик Убедитесь, что функционал трейсинга включен на уровне контрольной панели Synapse Service Mesh

Чеклист валидации установки

1. Создан deployment с именем - synapse-tracer-v2
2. Создан service с именем zipkin
3. Создан service с именем jaeger-collector
4. Создан configMap с именем tracer-config
5. Создан envoyFilter с именем filter-custom-tag-traice
6. Создан secret с именем tengritrust
7. Secret с именем tengritrust содержит файлы сертификатов под именами - client.key, client.cer, issuerootcatracecert.pem