EVP Platform V Synapse Streaming Event Processing

EVTP Сервис обработки сообщений

Руководство по эксплуатации

ОГЛАВЛЕНИЕ

Руководство по эксплуатации

Термины и определения

Термин/Аббревиатура	Определение
DSL	Dictionary Specification Language — предметно-ориентированный язык — это язык программирования с более высоким уровнем абстракции, который отражает специфику решаемых с их помощью задач
EVTP	Четырехбуквенный код программного компонента. Streaming Event Processing — сервис обработки событий, из состава программного продукта Platform V Synapse Streaming Event Processing, основанный на технологиях Apache Flink
TaskManager EVTP	Один из двух основных компонентов сервиса EVTP, играет роль исполнителя и предоставляет свои ресурсы в виде потоков для работы отдельных экземпляров операторов
JobManager EVTP	Один из двух основных компонентов сервиса EVTP, является координатором работы кластера, контролирует работу Task Manager и управляет командами и другими процессами

Термин/Аббревиатура	Определение
playbook	Список сценариев для запуска
Параллелизм / parallelism	Количество одновременно запускаемых экземпляров обработки
Nexus	Nexus репозиторий
inventory	Хранилище конфигурационных файлов и настроек
Источник	Поток событий, который в дальнейшем будет подвержен преобразованиям. Источников может быть несколько
Шаги преобразования	Преобразования с потоками событий могут выполняться в несколько шагов

Руководство прикладного разработчика

Системные требования

Java версии 11 и выше.

Подключение и конфигурирование

Загрузите проект универсального потокового обработчика в среду разработки кода. Необходимо импортировать в среду разработки **x.jar**.

Миграция на текущую версию

При переходе на текущую версию программного компонента сохраняется совместимость с предыдущими версиями.

Разработка первого приложения с использованием программного продукта

Основные пользовательские функции описаны в руководстве оператора. В данном разделе описаны дополнительные пользовательские функции, которые могут быть добавлены разработчиками в случае необходимости.

Дополнительные пользовательские функции добавляются в виде классов в пакет `package ru.sbt.cep.flow.event.process.flow.function.dsl`.
За основу берется класс `EmptyUserFunction`:

```
package ru.sbt.cep.flow.event.process.flow.function.dsl
import ru.sbt.cep.mapper.grammar.functions.user.UserDefinedFunction
import ru.sbt.cep.mapper.tree.{TrNode, TrUnitNode}

class EmptyUserFunction(name: String, args: Int = 0) extends
```

```
UserDefinedFunction {
override def getName: String = name
override def validateArgumentsSize(n: Int): Boolean = n >= 0 && n
<= args
override def invoke(args: Array\[TrNode\]): TrNode = TrUnitNode.UNIT
}

object EmptyUserFunction {
def apply(name: String, args: Int = 0): EmptyUserFunction = new
EmptyUserFunction(name, args)
}
```

Необходимо указать:

- имя функции для обращения к ней в DSL в `getName()`;
- количество аргументов в `validateArgumentsSize(int n)`;
- в invoke занести непосредственно код функции.

В самом DSL производится обращение к функции по ее имени.

Пример:

Функция:

```
class ConcatFunction extends UserDefinedFunction with Encryption {

override def getName: String = "Concat"

override def validateArgumentsSize(n: Int): Boolean = n == 2

override def invoke(args: Array\[TrNode\]): TrNode = {

val stringArgs = args.map(TreeWalker.extractString(\_))

val (string1, string2) = (stringArgs(0), stringArgs(1))

new TrTextNode(string1 + string2)

}
```

```
}
```

В DSL:

```
out.result = Concat(IN.name, IN.surname)
```

Использование программного продукта

Основная функциональность продукта — настройка правил преобразования потоков событий.

Дополнительно предоставляется возможность расширения функциональности через добавление пользовательских модулей.

Часто встречающиеся проблемы и пути их устранения

Не выявлено.

Руководство оператора

Доступ к приложению

Не применимо.

Использование приложения оператором

Создание новых конфигураций обработчиков

Конфигурация потокового обработчика событий состоит из одного файла `*.conf`, описывающего ход выполнения обработки событий, и одного или нескольких файлов `*.tr`, описывающих шаги обработки событий.

Формирование файла `*.conf`

Файл `*.conf` обязательно должен содержать следующие секции:

- наименование потока и перечень источников;
- `flow: {`
- `  name: "<наименование>"`
- `  source: [${<наименование источника 1>},${<наименование источника n>}]`
- `}`
- определение источника;
- `<наименование источника>: {`
- `  name: "<описание источника в формате <Домен>_<Топик>_<Поток>>"`
- `  type: "source"`
- `  topic: "<наименование топика>"`
- `  config: ${<ссылка на конфигурацию транспорта из defaults.json>} {`
- `    "consumer": {`
- `      "group.id": "<consumer group, с которой производится подключение>"`
- `    }`
- `  }`
- `  destination: ${<наименование следующего шага обработки>}`
- `}`
- определение выходной точки;
- `<наименование выходной точки>: {`
- `  name: "<описание выходной точки>"`
- `  type: "destination"`
- `  topic: "<наименование топика>"`
- `  config: ${<ссылка на конфигурацию транспорта из defaults.json>}`
- `}`
- определение шагов преобразований (дополнительная секция).
- `<наименование шага преобразования>: {`
- `  name: "<описание шага преобразования>"`
- `  type: "dsl"`
- `  format: {`
- `    input: "<указание входного формата xml/json>"`
- `    output: "<указание выходного формата xml/json>"`
- `  }`
-
- `  dsl: {`

```
      source: "file"
      path: "путь до tr файла, описывающего преобразования"
    }
    destination: ${<наименование следующего шага обработки>}
  }
```

Формирование файла `*.tr`

В файле `*.tr` содержатся необходимые преобразования событий. По умолчанию обработчик не производит никаких действий над потоком, все необходимые действия должны указываться явно.

Преобразования, которые могут быть выполнены над потоком событий:

- копирование заголовков;
- `headers.kafka = Environment.kafka.headers`
- копирование тела сообщения;
- `OUT = IN`
- передача на выход тела сообщения, полученного на вход;
- `return("passthrough")`
- журналирование.
- `INFO(<текст>, <идентификатор события>, <источник события>, <получатель события>)`

Вызов внешних сервисов по http(s) из потоковых обработчиков

Для того, чтобы осуществлять вызов внешних сервисов по http(s) из потокового обработчика, необходимо:

1. Внести изменения в файл ***.conf**

В шагах трансформации и агрегации предусмотрена возможность обращения к сторонним сервисам по rest-api. Для этого необходимо в описании потока в файле ***.conf** добавить параметры:

- `restUrls` - массив ключ-значение, где ключ - это id, по которому будет происходить обращение к функции, значение - соответствующий ему url;
- `restSslProperties` - опциональные настройки ssl; используются в случае, если подключение к сторонним сервисам производится с помощью ssl-соединения (`ssl.keystore.location` `ssl.keystore.password` `ssl.key.password` `ssl.truststore.location` `ssl.truststore.password` `ssl.protocol`)

Пример конфигурации (файл .conf)

```
flow: {
  name: "Event Process test"
  source: [${source}]
  restUrls {
    testGet: "https://localhost:12345/test"
  }
  restSslProperties {
    "ssl.keystore.location" : "/path/to/jks"
    "ssl.keystore.password" : "pass"
    "ssl.truststore.location" : "/path/to/jks"
    "ssl.truststore.password" : "pass"
    "ssl.key.password" : "pass"
  }
}
```

1. Внести изменения в файл ***.tr**

Для вызова внешнего сервиса по http(s) в файле ***.tr** используются функции:

- getRequest("url-id", "allow-redirects", "headers") - для отправки GET запросов;
- postRequest("url-id", "allow-redirects", "headers", "body") - для отправки POST запросов;
- deleteRequest("url-id", "allow-redirects", "headers") - для отправки DELETE запросов;
- putRequest("url-id", "allow-redirects", "headers", "body") - для отправки PUT запросов;

где

- *url-id* - alias url из файла конфигурации ***.conf** (в примере файла конфигурации "testGet");
- *allow-redirects* - параметр REST-запроса, отвечает за перенаправление запроса;
- *headers* - параметр REST-запроса, заголовки ключ-значение;
- *body* - параметр REST-запроса, тело запроса.

Пример запроса

```
curl -X get -H "input-header: test" http://localhost:12345/test
```

По результатам вызова функции будет сформировано сообщение вида:

```
{"statusCode":200,"body":"Test response","headers":{"content-length":"13","date":"Mon, 08 Nov 2021 14:00:23 GMT"}}
```

где *statusCode* - код ответа сервиса, *body* - тело ответа сервиса, *headers* - заголовки ответа сервиса.

Пример использования функции в файле .tr

```
define result = getRequest("testGet", false, in)
out = $result
```

Запуск обработчика

Для запуска обработчика выполните команду на узле, на котором расположен Job Manager: `/opt/Apache/flink/bin/flink run -d -p 1 -C file:///<полный путь до fat Jar универсального обработчика> -c ru.sbt.cep.flow.event.process.Main <полный путь до Jar универсального обработчика> --config "<полный путь до конфигурации обработчика>" --defaults "<полный путь до файла с настройками транспорта>"`.
Для запуска обработчика из сохраненного состояния используйте флаг `-s "<полный путь до сохраненного состояния>"` .

Остановка работы обработчика

1. Выполните команду для получения списка идентификаторов обработчиков на узле, на котором расположен Job Manager: `/opt/Apache/flink/bin/flink list -a`
2. Выполните команду для остановки работы обработчика: `/opt/Apache/flink/bin/flink stop <идентификатор запущенного обработчика>`

Прерывание работы обработчика

1. Выполните команду для получения списка идентификаторов обработчиков на узле, на котором расположен Job Manager: `/opt/Apache/flink/bin/flink list -a`
2. Выполните команду для прерывания работы обработчика: `/opt/Apache/flink/bin/flink cancel <идентификатор запущенного обработчика>`
   При этом не сохранится состояние обработчика.

Часто встречающиеся проблемы и пути их устранения

Часто встречающихся проблем не выявлено.

Ниже приведены рекомендации при создании файла трансформации `*.tr`.

- Все необходимые для передачи параметры должны быть явно указаны в описании трансформации.
- Конструкция `OUT = IN` строит новое выходное дерево события на основе входного. Если требуется передача тела события без изменения, то лучше использовать `return("passthrough")`.
- При использовании сравнений рекомендуется явно производить приведение типов, используя конструкции `nodeToString()`, `toInt()`, `toLong()`, `toDecimal()`, `toDouble()`.

Параметры настройки

Настройки продукта задаются администратором и указаны в документе «Руководстве по установке».

Правила эксплуатации

Дополнительные правила эксплуатации отсутствуют.

Руководство по системному администрированию

Сценарии администрирования

Создание архивных копий компонентов EVTP

Предусловия: Необходимо заполнить параметры в конфигурационном файле **vars.yml**.

- в разделе `flink`:
  - `backup_installdir_to:` — директория создания архивных копий для директории установки EVTP;

- в разделе `zookeeper`:
    - `backup_installdir_to:` — директория создания архивных копий для директории установки Zookeeper.

Ручной способ

На сервере, с которого производилась установка выполнить команду:

```
ansible-playbook -i inventories/<ID>/inventory flink.yml --ask-vault-pass -t backup -l <host>
```

, где `ID` - имя созданного inventory, `host` - доп. параметр для указания конкретных хостов, если требуется выполнение на конкретных хостах. Данной операцией создается архивная копия только для EVTP. Для создания архивной копии только Zookeeper используется аналогичная команда с выбором playbook `zookeeper.yml`.

С помощью Jenkins (опциональный способ)

1. Для создания архивной копии директории установки EVTP необходимо запустить задание Jenkins **SYN_custom** (поставляется в дистрибутиве) с выбором playbook `flink.yml` с тегом *backup*. Результатом выполнения задачи Jenkins является архив с названием *flink_backup.tar.gz*, расположенный в директории, указанной в параметрах конфигурационного файла **vars.yml**.
2. Для создания архивной копии Zookeeper необходимо запустить задание Jenkins **SYN_custom** (поставляется в дистрибутиве) с выбором playbook `zookeeper.yml` с тегом *backup*. Результатом выполнения задачи Jenkins является создание архива *zookeeper_backup.tar.gz*, расположенного в директории, указанной в параметрах конфигурационного файла **vars.yml**.

Созданные архивные копии хранятся в директориях в единичном экземпляре для предотвращения переполнения директории хранения архивов. При попытке создания архивных копий компонентов повторно выполнение задания Jenkins будет прерываться, если в указанной директории уже создан ранее архив. Поэтому необходимо перед созданием нового архива предварительно удалить существующие архивные копии из данных директорий. Для этого необходимо запустить задание Jenkins **SYN_custom** с выбором соответствующего playbook с тегом *backup_remove*.

Теги *backup_remove* и *backup* можно совмещать в одном запуске задания Jenkins для экономии времени.

Восстановление из архивных копий компонентов EVTP

Предусловия: Необходимо заполнить параметры в конфигурационном файле **vars.yml**.

- в разделе `flink`:
    - `backup_installdir_to:` — директория создания архивных копий для директории установки EVTP;

- в разделе `zookeeper`:
    - `backup_installdir_to:` — директория создания архивных копий для директории установки Zookeeper.

Ручной способ

На сервере, с которого производилась установка выполнить команду:

```
ansible-playbook -i inventories/<ID>/inventory flink.yml --ask-vault-pass -t backup_restore -l <host>
```

, где `ID` - имя созданного inventory, `host` - доп. параметр для указания конкретных хостов, если требуется выполнение на конкретных хостах. Данной операцией создается архивная копия для EVTP. Для создания архивной копии Zookeeper используется аналогичная команда с выбором playbook `zookeeper.yml`.

С помощью Jenkins (опциональный способ)

1. Для восстановления EVTP из архивной копии необходимо запустить задание Jenkins **SYN_custom** (поставляется в дистрибутиве) с выбором playbook `flink.yml` с тегом *backup_restore*. Результатом будет являться восстановление директории установки EVTP из архива, помещенного в указанную в файле **vars.yml** директорию.
2. Для восстановления Zookeeper из архивной копии необходимо запустить задание Jenkins **SYN_custom** (поставляется в дистрибутиве) с выбором playbook `zookeeper.yml` с тегом *backup_restore*. Результатом будет являться восстановление директории установки Zookeeper из архива, помещенного в указанную в файле **vars.yml** директорию.

Перезапуск компонентов EVTP

Ручной вариант

Перейдите на узел, на котором будет производиться перезапуск EVTP или одного из компонентов EVTP.

Если требуется перезапусть Job Manager, то выполните действия:

1. Остановите сервис `sudo systemctl stop flink_jobmanager.service`.
2. Остановите Job Manager `opt/Apache/flink/bin/jobmanager.sh stop-all`.
3. Запустите Job Manager `opt/Apache/flink/bin/jobmanager.sh start`.
4. Запустите сервис `sudo systemctl start flink_jobmanager.service`.

Если требуется перезапусть Task Manager, то выполните действия:

1. Остановите сервис `sudo systemctl stop flink_taskmanager.service`.
2. Остановите Job Manager `opt/Apache/flink/bin/taskmanager.sh stop-all`.
3. Запустите Job Manager `opt/Apache/flink/bin/taskmanager.sh start`.
4. Запустите сервис `sudo systemctl start flink_taskmanager.service`.

Если требуется перезапусть Zookeeper, то выполните действия:

1. Остановите сервис `sudo systemctl stop zookeeper.service`.
2. Остановите Zookeeper `opt/Apache/flink/bin/zookeeper.sh stop-all`.
3. Запустите Zookeeper `opt/Apache/flink/bin/zookeeper.sh start`.
4. Запустите сервис `sudo systemctl start zookeeper.service`.

С помощью Jenkins (опциональный способ)

Для перезапуска компонентов с помощью Jenkins используйте задание `SYN_custom`:

- с выбором playbook `flink.yml` с тегами `jobmanager_stop`, `jobmanager_start` (перезапустит JobManager EVTP);
- с выбором playbook `flink.yml` с тегами `taskmanager_stop`, `taskmanager_start` (перезапустит TaskManager EVTP);
- с выбором playbook `zookeeper.yml` с тегами `stop`, `start` (перезапустит Zookeeper).

Настраиваемые параметры:

- `kontur` — выбрать контур, на котором необходимо перезапустить компоненты;
- `only_on_host` - отметить галочками нужные хост(ы) из списка (список соответствует выбранному inventory), если необходимо перезапустить компоненты только для данного хоста(ов) выбранного кластера;
- `install_all_hosts` - выбрать параметр, если необходимо перезапустить все хосты из данного inventory.

Если не выбран ни один из параметров `only_on_host`, `install_all_hosts` выполнение задания Jenkins прервется с ошибкой *Не выбраны хосты*.

Просмотр списка запущенных обработчиков

Ручной вариант

На узле, на котором расположен Job Manager, выполните команду `/opt/Apache/flink/bin/flink list -a`.

С помощью Jenkins (опциональный способ)

Для перезапуска компонентов с помощью задания Jenkins используйте **SYN_custom** с выбором playbook `flink.yml` с тегом `status` (отображает список запущенных обработчиков EVTP).

Настраиваемые параметры:

- `kontur` — выбрать контур, на котором необходимо запущенных обработчиков EVTP;
- `only_on_host` - отметить галочками нужные хост(ы) из списка (список соответствует выбранному inventory), если необходимо перезапустить компоненты только для данного хоста(ов) выбранного кластера;
- `install_all_hosts` - выбрать параметр, если необходимо перезапустить все хосты из данного inventory.

Если не выбран ни один из параметров `only_on_host`, `install_all_hosts` выполнение задания Jenkins прервется с ошибкой *Не выбраны хосты*.

Запуск обработчика

Ручной вариант

На узле, на котором расположен Job Manager, выполните команду:

```
/opt/Apache/flink/bin/flink run -d -p 1 -C file:///<полный путь до fat Jar универсального обработчика> -c ru.sbt.cep.flow.event.process.Main
<полный путь до Jar универсального обработчика> --config "<полный путь до конфигурации обработчика>" --defaults "<полный путь до файла с
настройками транспорта>"
```

Описание файла **defaults.json** находится в документе «Руководство по установке», раздел «Настройка транспорта».

Дополнительно может использоваться параметр `-s "<полный путь до сохраненного состояния>"` для запуска обработчика из сохраненного состояния.

С помощью Jenkins (опциональный способ)

Для запуска компонентов с помощью задания Jenkins используйте в папке *Release* задание Jenkins `flink_universal_job_install` с выбором тега `start_flink_jobs` (запускается EVTP выбранной версии в выбранном кластере).

Настраиваемые параметры:

- `inventory` — кластер, для которого необходимо запустить EVTP;
- `nexus_version` — версия задания Jenkins из nexus;

- `parallelism` — указать параллелизм

Остановка обработчика

Ручной вариант

1. На узле, на котором расположен Job Manager, выполните команду для получения списка идентификаторов обработчиков:
2. `/opt/Apache/flink/bin/flink list -a`
3. Выполните команду для остановки обработчика:
4. `/opt/Apache/flink/bin/flink stop <идентификатор запущенного обработчика>`

С помощью Jenkins (опциональный способ)

Для остановки компонентов с помощью задания Jenkins используйте в папке *Release* задание Jenkins `flink_universal_job_install` с выбором тега `stop_flink_jobs` (остановка EVTP выбранной версии (из nexus) в выбранном кластере).

Настраиваемые параметры:

- `inventory` — выбрать кластер, для которого необходимо остановить EVTP;
- `nexus_version` — выбрать версию задания Jenkins из nexus;
- `parallelism` — указать параллелизм.

Для остановки обработчика необходимо установить параметр `savepointSave`. При этом сохранится состояние обработчика.

Прерывание работы обработчика

Ручной вариант

1. На узле, на котором расположен Job Manager, выполните команду для получения списка идентификаторов обработчиков:
2. `/opt/Apache/flink/bin/flink list -a`
3. Выполните команду для прерывания работы обработчика:
4. `/opt/Apache/flink/bin/flink cancel <идентификатор запущенного обработчика>`
   При этом не сохранится состояние обработчика.

С помощью Jenkins (опциональный способ)

Для остановки компонентов с помощью задания Jenkins используйте в папке *Release* задание Jenkins `flink_universal_job_install` с выбором тега `stop_flink_jobs` (остановит EVTP выбранной версии (из nexus) в выбранном кластере).

Настраиваемые параметры:

- `inventory` — выбрать кластер, для которого необходимо остановить EVTP;
- `nexus_version` — выбрать версию задания Jenkins из nexus;
- `parallelism` — указать параллелизм.
- **Для остановки обработчика необходимо отключить параметр `savepointSave`. При этом не сохранится состояние обработчика.**

Перезапуск обработчиков

Ручной вариант

1. На узле, на котором расположен Job Manager, выполните команду для получения списка идентификаторов обработчиков:
2. `` `/opt/Apache/flink/bin/flink list -a` ``
3. Выполните команду для остановки обработчика:
4. `` `/opt/Apache/flink/bin/flink stop <идентификатор запущенного обработчика>` ``
5. Выполните команду для запуска обработчика:
6. `/opt/Apache/flink/bin/flink run -d -p 1 -C file:///<полный путь до fat Jar универсального обработчика> -c ru.sbt.cep.flow.event.process.Main <полный путь до Jar универсального обработчика> --config "<полный путь до конфигурации обработчика>" --defaults "<полный путь до файла с настройками транспорта>"`

Дополнительно может использоваться параметр `-s "<полный путь до сохраненного состояния>"` для запуска обработчика из сохраненного состояния.

С помощью Jenkins (опциональный способ)

Для перезапуска компонентов с помощью заданий Jenkins используйте в папке *Release* задание Jenkins `flink_universal_job_install` с выбором тегов `stop_flink_jobs`, `start_flink_jobs` (перезапустит EVTP выбранной версии (из nexus) в выбранном кластере).

Настраиваемые параметры:

- `inventory` — выбрать кластер, для которого необходимо перезапустить EVTP;
- `nexus_version` — выбираем версию задания Jenkins из nexus;
- `parallelism` — указать параллелизм.

Использование скриптов автоматизации

Перезапуск компонентов EVTP

Ручной вариант

Перезапуск JobManager EVTP на сервере, с которого производилась установка, выполняется командой:

```
ansible-playbook -i inventories/<ID>/inventory flink.yml --ask-vault-pass -t jobmanager_start,jobmanager_stop
```

Перезапуск TaskManager EVTP на сервере, с которого производилась установка, выполняется командой:

```
ansible-playbook -i inventories/<ID>/inventory flink.yml --ask-vault-pass -t taskmanager_start,taskmanager_stop
```

Перезапуск Zookeeper на сервере, с которого производилась установка, выполняется командой:

```
ansible-playbook -i inventories/<ID>/inventory zookeeper_for_flink.yml --ask-vault-pass -t start,stop
```

Здесь `ID` — имя созданного inventory.
Для указания конкретных узлов используйте параметр `-l <fqdn узла>`.
При использовании Jenkins выберите нужный контур, playbook `flink.yml`/`zookeeper.yml` параметры `start` и `stop`, узлы на которых будет выполняться операция перезапуска компонентов EVTP, или оставьте пустыми для выполнения перезапуска компонентов EVTP на всех узлах.

С помощью Jenkins (опциональный способ)

Для перезапуска компонентов с помощью Jenkins используется задание `SYN_custom`:

- с выбором playbook `flink.yml` с тегами `jobmanager_stop`, `jobmanager_start` (перезапустит JobManager EVTP);
- с выбором playbook `flink.yml` с тегами `taskmanager_stop`, `taskmanager_start` (перезапустит TaskManager EVTP);
- с выбором playbook `zookeeper.yml` с тегами `stop`, `start` (перезапустит Zookeeper).

Настраиваемые параметры:

- `kontur` — выбрать контур, на котором необходимо перезапустить компоненты;
- `only_on_host` - отметить галочками нужные хост(ы) из списка (список соответствует выбранному inventory), если необходимо перезапустить компоненты только для данного хоста(ов) выбранного кластера;
- `install_all_hosts` - выбрать параметр, если необходимо перезапустить все хосты из данного inventory.

Если не выбран ни один из параметров `only_on_host`, `install_all_hosts` выполнение задания Jenkins прервется с ошибкой *Не выбраны хосты*.

Установка и перезапуск задач EVTP

Ручной способ

Заполнить в соответствующем *inventory* файл **group_vars/all/vars.yml**:

```
flink:
  jobs:
    my_job: # имя обработчика
      distr: "jobs/event-process-flow.jar"
      jobclass: "ru.sbt.cep.flow.event.process.Main"
      parallelism: 1
      args:
        config: <Относительный путь до файла конфигурации обработчика>
      args_from_flink_presets:
        - job_restart
        - job_checkpoint
```

Для создания сущностей используется команда:

```
ansible-playbook -i inventories/<ID>/inventory flink.yml --ask-vault-pass -t start_flink_jobs
```

где ID - имя созданного inventory.

Для удаления сущностей используем команду:

```
ansible-playbook -i inventories/<ID>/inventory flink.yml --ask-vault-pass -t stop_flink_jobs
```

где ID - имя созданного inventory.

При помощи Jenkins (опциональный способ)

Работа с сущностями ведется посредством конфигурационных дистрибутивов.

Создание конфигурационных дистрибутивов осуществляется с помощью задания Jenkins **flink_config_create**.

Необходимо заполнить параметры:

- *NexusUrl* — url-ссылка на папку, в которую необходимо поместить дистрибутив;
- *NexusCred* — ID креденшнл для публикации дистрибутива в указанной папке Nexus.

При запуске задания Jenkins с указанными параметрами дистрибутив EVTD будет создан в указанной папке.

Для создания сущностей используется задание Jenkins **flink_config_deploy** с выбором соответствующего конфигурационного дистрибутива, playbook **flink.yml** с тегом *start_flink_jobs*.

Для удаления сущностей используется задание Jenkins **flink_config_deploy** с выбором соответствующего конфигурационного дистрибутива, playbook **flink.yml** с тегом *stop_flink_jobs*.

Выполнить перезапуск и быстро восстановить работоспособность задач EVTP можно с помощью задания Jenkins **flink_config_deploy**.

Для перезапуска задач EVTP необходимо выбрать:

- параметр `all_releases_from_conf` - если требуется запустить все задачи EVTP, указанные в настройке inventory (запустятся все задачи EVTP, указанные в файле `flink_releases.conf`);
- в параметре `custom_releases_from_conf` выбрать те версии дистрибутивов, которые необходимо запустить (в данном параметре подгружается список всех версий дистрибутивов из файла `flink_releases.conf` из настроек `inventory`, можно выбрать конкретную версию задачи Apache Flink, работоспособность которой необходимо восстановить).

При работе с одним из этих параметров скачивания дистрибутивов из пространства nexus может не производиться, вследствие чего возможно сократить время работы задания Jenkins (при указании соответствующих настроек).

Предварительно файл `flink_releases.conf` должен быть заполнен всеми дистрибутивами требующихся обработчиков Flink. Порядок строк в данном файле имеет значение. Перезапуск или установка обработчиков выполняется в порядке, указанном в файле, сверху вниз. Первыми будут установлены те обработчики, которые расположены выше.
Шаблон заполнения файла `flink_releases.conf`

Варианты заполнения файла:

- Для скачивания дистрибутива из nexus — «<версия дистрибутива> <параллелизм>» (параметры указываются через пробел)

- Для использования дистрибутива со стенда — «<версия дистрибутива>,<параллелизм>,<версия_обработчика>,<имя_конфигурации>,<имя_задачи>», где:
  - <версия_обработчика> — содержимое файла `job.release` из дистрибутива;
  - <имя_конфигурации> — имя папки внутри `roles/flink/custom_files/mapping` из дистрибутива;
  - <имя_задачи> — поле `flow.name` в файле `roles/flink/custom_files/mapping/<имя_конфигурации>/<имя_конфигурации>.conf`

Примеры заполнения файла:

- D-01.001.00-15 4 — со скачиванием из nexus.
- D-30.002.00-39,1,SP-02.004.03-02,SMV2FCCM,SMV2FCCM — без скачивания из nexus, используются дистрибутивы со стенда.

Порядок строк в данном файле имеет значение. Перезапуск или установка обработчиков осуществляется в порядке, указанном в файле, сверху вниз.

Журналирование

Для EVTP может быть настроена отправка системных журналов в централизованную систему журналирования Platform V Monitor.

Предварительно необходимо узнать названия топиков централизованной системы журналирования Platform V Monitor, куда будут отправляться журналы EVTP.

Перед установкой (переустановкой) новой версии EVTP необходимо в конфигурационном файле `vars.yml` заполнить блок:

```
logback_kafka_appender:
    enable: false # включение механизма отправки логов в Kafka  через logback (true в случае использования механизма отправки логов в Platform V Monitoring)
#    topic_FlinkLogger: <имя топика системы журналирования для отправки логов EVTP> # топик Platform V Monitoring для отправки логов EVTP
#    topic_UniversalJobLogger: <имя топика системы журналирования для логов универсального обработчика> # топик для отправки логов универсального обработчика
```

и блок *logback_kafka_appender.producer_configs:*

```
producer_configs: # настройка Kafka producer
      - "bootstrap.servers=<бутстрапы подключения к централизованной системе журналирования>"
      - "security.protocol=SSL"
      - "ssl.keystore.location=<указать путь до сертификатов подключения к централизованной системе журналирования>"
      - "ssl.keystore.password=<зашифрованный пароль подключения к keystore>"
```

```
    - "ssl.truststore.location=<указать путь до сертификатов подключения к централизованной системе журналирования>"
    - "ssl.truststore.password=<зашифрованный пароль подключения к truststore>"
    - "ssl.endpoint.identification.algorithm="
```

Настройки отправляемых сообщений указываются в файле `/conf/logback.xml`. Отправляемые сообщения имеют формат JSON согласно шалблону, описываемому в файле `logback.xml`.
Для настройки отправки сообщений необходимо заполнить блок аппендера `FlinkLogger` файла `logback.xml`.

В данном блоке стоит обратить внимание на заполнение следующих параметров:

Параметр	Описание	Значение по умолчанию
message	Указать название поля для тела сообщения	message
timestamp	Указать название поля для временной метки	timestamp
level	Уровень логирования сообщения	level
needAddThreadToMessage	Необходимость добавления имени потока в поле с телом сообщения	true
name=host	Имя хоста расположения EVTP	dns-имя или ip

Параметр	Описание	Значение по умолчанию
name=system	Тип сервиса (job-manager или task-manager)	job-manager
topic	Топик записи сообщений логов	flink_log
client.id	Идентификатор продьюсера Kafka, по которому можно однозначно идентифицировать продьюсера Apache Kafka	пусто
bootstrap.servers	Хосты подключения к системе журналирования	host:port,host:port
security.protocol	Тип протокола подключения к системе журналирования	SSL
ssl.endpoint.identification.algorithm		пусто

Параметр	Описание	Значение по умолчанию
appender-ref	Альтернативное место для сообщений журналов	По умолчанию не указывается

Пример файла с заполненными настройками:

```
configuration>
        <appender name="file" class="ch.qos.logback.core.rolling.RollingFileAppender">
                <file>${log.file}</file>
                <rollingPolicy class="ch.qos.logback.core.rolling.SBTTimeBasedRollingPolicy">
                        <!--+  ежедневно создается новый файл логов, старый переносится в архив -->
                        <fileNamePattern>${log.file}-%d{yyyy-MM-dd}.%i.log.zip</fileNamePattern>
                        <timeBasedFileNamingAndTriggeringPolicy class="ch.qos.logback.core.rolling.SizeAndTimeBasedFNATP">
                                <!--+ или когда размер файла логов достигнет размера 50MB -->
                                <maxFileSize>50MB</maxFileSize>
                                </timeBasedFileNamingAndTriggeringPolicy>
                        <maxHistory>3</maxHistory>
                </rollingPolicy>
                <encoder>
                        <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%thread] %-5level %logger{60} %X{sourceThread} - %msg%n</pattern>
        </encoder>
        </appender>

        <!--+ Блок настройки для отправки логов Apache Flink (EVTP) в центральную систему журналирования Platform V Monitor  -->
        <appender name="FlinkLogger" class="com.github.danielwegener.logback.kafka.KafkaAppender">
                <encoder class="com.github.danielwegener.logback.kafka.encoding.LayoutKafkaMessageEncoder">
                        <layout class="ru.sbt.logback.layout.Grok">
                                <message>message</message>
                                <timestamp>timestamp</timestamp>
                                <level>level</level>
                                <needAddThreadToMessage>false</needAddThreadToMessage>
                                <fields>
                                        <![CDATA[
                                                name=host       ENV_VARIABLE=FLINK_IP,HOSTNAME
```

```
                                                name=system      CONST=Common
                                        ]]>
                                </fields>
                        </layout>
                </encoder>
                <deliveryStrategy class="com.github.danielwegener.logback.kafka.delivery.AsynchronousDeliveryStrategy"/>
                <topic>flink_log</topic>
                <producerConfig>client.id=flink_fp_ss_tkles-snaps0001.vm.esrt.cloud.sbrf.ru</producerConfig>
                <producerConfig>bootstrap.servers=tkleq-snaps0001.vm.esrt.cloud.sbrf.ru:9092,tkleq-
snaps0002.vm.esrt.cloud.sbrf.ru:9092</producerConfig>
                <producerConfig>security.protocol=SSL</producerConfig>
                <producerConfig>ssl.keystore.location=/opt/Apache/flink/ssl/flink.jks</producerConfig>
                <producerConfig>ssl.keystore.password=<пароль к keystore></producerConfig>
                <producerConfig>ssl.truststore.location=/opt/Apache/flink/ssl/flink.jks</producerConfig>
                <producerConfig>ssl.truststore.password=<пароль к truststore></producerConfig>
                <producerConfig>ssl.endpoint.identification.algorithm=</producerConfig>
                <!--<appender-ref ref="file"/>-->
        </appender>

        <!--+ Блок настроек для отправки логов Apache Kafka (EVTD) в центральную систему журналирования Platform V Monitor -->
        <appender name="KafkaLogger" class="com.github.danielwegener.logback.kafka.KafkaAppender">
                <encoder class="com.github.danielwegener.logback.kafka.encoding.LayoutKafkaMessageEncoder">
                        <layout class="ru.sbt.logback.layout.Grok">
                                <message>message</message>
                                <timestamp>timestamp</timestamp>
                                <level>level</level>
                                <needAddThreadToMessage>false</needAddThreadToMessage>
                                <fields>
                                        <![CDATA[
                                                name=host        ENV_VARIABLE=FLINK_IP,HOSTNAME
                                                name=system      CONST=Kafka
                                        ]]>
                                </fields>
                        </layout>
                </encoder>
                <deliveryStrategy class="com.github.danielwegener.logback.kafka.delivery.AsynchronousDeliveryStrategy"/>
                <topic>flink_log</topic>
                <producerConfig>client.id=flink_fp_ss_host</producerConfig>
                <producerConfig>bootstrap.servers=host:port,host:port</producerConfig>
                <producerConfig>security.protocol=SSL</producerConfig>
                <producerConfig>ssl.keystore.location=/opt/Apache/flink/ssl/flink.jks</producerConfig>
                <producerConfig>ssl.keystore.password=<пароль к keystore></producerConfig>
```

```
        <producerConfig>ssl.truststore.location=/opt/Apache/flink/ssl/flink.jks</producerConfig>
        <producerConfig>ssl.truststore.password=<пароль к truststore></producerConfig>
        <producerConfig>ssl.endpoint.identification.algorithm=</producerConfig>
        <!--<appender-ref ref="file"/>-->
    </appender>

    <!--+ Блок настроек для отправки логов универсального обработчика (EVTP) в центральную систему журналирования Platform V Monitor --
>
    <appender name="UniversalJobLogger" class="com.github.danielwegener.logback.kafka.KafkaAppender">
        <encoder class="com.github.danielwegener.logback.kafka.encoding.LayoutKafkaMessageEncoder">
            <layout class="ch.qos.logback.classic.PatternLayout">
                <pattern>%msg%n</pattern>
            </layout>
        </encoder>
        <deliveryStrategy class="com.github.danielwegener.logback.kafka.delivery.AsynchronousDeliveryStrategy"/>
        <topic>flink_log</topic>
        <producerConfig>client.id=flink_fp_ss_host</producerConfig>
        <producerConfig>bootstrap.servers=host:port,host:port</producerConfig>
        <producerConfig>security.protocol=SSL</producerConfig>
        <producerConfig>ssl.keystore.location=/opt/Apache/flink/ssl/flink.jks</producerConfig>
        <producerConfig>ssl.keystore.password=<пароль к keystore></producerConfig>
        <producerConfig>ssl.truststore.location=/opt/Apache/flink/ssl/flink.jks</producerConfig>
        <producerConfig>ssl.truststore.password=<пароль к truststore></producerConfig>
        <producerConfig>ssl.endpoint.identification.algorithm=</producerConfig>
        <!--<appender-ref ref="file"/>-->
    </appender>

    <appender name="ASYNC-UniversalJobLogger" class="ch.qos.logback.classic.AsyncAppender">
        <appender-ref ref="UniversalJobLogger"/>
    </appender>
    <appender name="ASYNC-FlinkLogger" class="ch.qos.logback.classic.AsyncAppender">
        <appender-ref ref="FlinkLogger"/>
    </appender>
    <appender name="ASYNC-KafkaLogger" class="ch.qos.logback.classic.AsyncAppender">
        <appender-ref ref="KafkaLogger"/>
    </appender>

    <logger name="ru.sbt" level="DEBUG">
        <appender-ref ref="file"/>
    </logger>

    <logger name="ru.sbt.cep.flow.event.process.flow.function.EventHolderTransformer" level="INFO" additivity="false">
```

```
                <appender-ref ref="file"/>
                <appender-ref ref="ASYNC-UniversalJobLogger"/>
        </logger>

        <logger name="org.apache.flink" level="INFO">
                <appender-ref ref="file"/>
                <appender-ref ref="ASYNC-FlinkLogger"/>
        </logger>

        <logger name="org.apache.flink.streaming.api.functions.sink.TwoPhaseCommitSinkFunction" level="WARN">
                <appender-ref ref="file"/>
                <appender-ref ref="ASYNC-FlinkLogger"/>
        </logger>

        <logger name="org.apache.flink.runtime.checkpoint.CheckpointCoordinator" level="WARN">
                <appender-ref ref="file"/>
                <appender-ref ref="ASYNC-FlinkLogger"/>
        </logger>

        <logger name="akka" level="INFO">
                <appender-ref ref="file"/>
        </logger>

        <logger name="org.apache.kafka" level="INFO">
                <appender-ref ref="file"/>
                <appender-ref ref="ASYNC-KafkaLogger"/>
        </logger>

        <logger name="org.apache.hadoop" level="INFO">
                <appender-ref ref="file"/>
        </logger>

        <logger name="org.apache.zookeeper" level="INFO">
                <appender-ref ref="file"/>
        </logger>

        <logger name="org.apache.flink.shaded.akka.org.jboss.netty.channel.DefaultChannelPipeline" level="ERROR">
                <appender-ref ref="file"/>
                <appender-ref ref="ASYNC-FlinkLogger"/>
        </logger>
</configuration>
```

События системного журнала

Программный компонент EVTP сохраняет информацию о происходящих событиях в файлы:

- `$kafka_logs_dir/*-standalonesession-*.log` — содержит события Job Manager. Подключение и отключение Task Manager, события смены лидера в кластере, события в работе обработчиков.
- `$kafka_logs_dir/*-taskexecutor-*.log` — содержит события Task Manager. События подключения к Job Manager-лидеру, события в работе обработчиков, журналы самих обработчиков.

События мониторинга

Мониторинг выполняется:

- самостоятельно администратором EVTP через JMX-порт;
- с помощью системы мониторинга Mayak компонента Platform V Synapse Event-domain management (EDM). Настройка подключения описана в документации по мониторингу. Подробное описание использования: *Руководство пользователя* документации *Система мониторинга Mayak*.

Список отслеживаемых метрик:

Область действия метрики	**Метрика**	**Триггер**	**Единица измерения**	**MBEAN Name**

Область действия метрики	Метрика	Триггер	Единица измерения	MBEAN Name
JobManager	Загрузка CPU	Если максимальное значение больше 80% - предупреждение, 90% - ошибка	от 0 до 1	org.apache.flink.jobmanager.Status.JVM.CPU.Load:host=tkles-snaps0001.vm.esrt.cloud.sbrf.ru
JobManager	Время использования CPU	должно быть > 0	мс	org.apache.flink.jobmanager.Status.JVM.CPU.Time:host=tkles-snaps0001.vm.esrt.cloud.sbrf.ru
JobManager	Использование MemoryHe	Если максимальное	байт	org.apache.flink.jobmanager.Status.JVM.Memory.Heap.Used:host=tkles-snaps0001.vm.esrt.cloud.sbrf.ru

Область действия метрики	Метрика	Триггер	Единица измерения	MBEAN Name
	ap	значение больше 80% - предупреждение, 90% - ошибка		
JobManager	Кол-во памяти, доступное JVM	Должно быть не меньше 100 mb	байт	org.apache.flink.jobmanager.Status.JVM.Memory.Heap.Committed:host=tkles-snaps0001.vm.esrt.cloud.sbrf.ru
JobManager	Максимальное кол-во памяти, доступное JVM	Должно быть не больше 10Gb	байт	org.apache.flink.jobmanager.Status.JVM.Memory.Heap.Max:host=tkles-snaps0001.vm.esrt.cloud.sbrf.ru

Область действия метрики	Метрика	Триггер	Единица измерения	MBEAN Name
TaskManager	Кол-во зарегистрированных таскменеджеров	не должно меняться	шт	org.apache.flink.jobmanager.numRegisteredTaskManagers:host=tkles-snaps0001.vm.esrt.cloud.sbrf.ru
TaskManager	Кол-во запущенных служб по кластеру	не должно уменьшаться	шт	org.apache.flink.jobmanager.numRunningJobs:host=tkles-snaps0001.vm.esrt.cloud.sbrf.ru
TaskManager	Кол-во свободных слотов	шт	org.apache.flink.jobmanager.taskSlotsAvailable:host=tkles-snaps0001.vm.esrt.cloud.sbrf.ru	
TaskM	Всего	не меньше	шт	org.apache.flink.jobmanager.taskSlotsTotal:host=tkles-

Область действия метрики	Метрика	Триггер	Единица измерения	MBEAN Name
anager	слотов	50		snaps0001.vm.esrt.cloud.sbrf.ru
TaskManager	Загрузка CPU таскменеджером	Если максимальное значение больше 80% - предупреждение, 90% - ошибка	байт	org.apache.flink.taskmanager.Status.JVM.CPU.Load:host=tkles-snaps0001,tm_id=104af92d2483545
TaskManager	Время использования CPU таскменеджером	Должно быть больше > 0	мс	org.apache.flink.taskmanager.Status.JVM.CPU.Time:host=tkles-snaps0001,tm_id=104af92d2483545043b193c1805f02e0

Область действия метрики	Метрика	Триггер	Единица измерения	MBEAN Name
TaskManager	Используемая память	Если максимальное значение больше 80% - предупреждение, 90% - ошибка	байт	org.apache.flink.taskmanager.Status.JVM.Memory.Heap.Used:host=tkles-snaps0001,tm_id=104af92d2483545043b193c1805f02e0
TaskManager	Кол-во памяти, гарантированно доступной	Должно быть не больше 10Gb	байт	org.apache.flink.taskmanager.Status.JVM.Memory.Heap.Committed:host=tkles-snaps0001,tm_id=104af92d2483545043b193c1805f02e0

Область действия метрики	Метрика	Триггер	Единица измерения	MBEAN Name
TaskManager	Максимальное кол-во памяти, которое может быть использовано	Должно быть не больше 10Gb	байт	org.apache.flink.taskmanager.Status.JVM.Memory.Heap.Max:host=tkles-snaps0001,tm_id=104af92d2483545043b193c1805f02e0
Job	Сколько времени занимает перезапуск службы, или как долго длится текущий перезапус	не больше 10 сек	мс	org.apache.flink.jobmanager.job.restartingTime:host=tkles-snaps0001.vm.esrt.cloud.sbrf.ru, job_name=AppTest,job_id=effdc9639a698236fadf1091fc132fca

Область действия метрики	Метрика	Триггер	Единица измерения	MBEAN Name
	к			
Job	Время работы службы	должно быть больше -1	мс	org.apache.flink.jobmanager.job.uptime:host=tkles-snaps0001.vm.esrt.cloud.sbrf.ru, job_name=ALPHA.COMMON.ESBSM.XFERSYNC, job_id=2b005e66b405382d85b0c5b0d0bfaef8
Job	Время недоступности службы	должно быть равно 0	мс	org.apache.flink.jobmanager.job.downtime:host=tkles-snaps0001.vm.esrt.cloud.sbrf.ru,job_name=ALPHA.COMMON.ESBSM.XFERSYNC,job_id=2b005e66b405382d85b0c5b0d0bfaef8
Job	Кол-во рестартов службы	не больше 5 раз	шт	org.apache.flink.jobmanager.job.numRestarts:host=tkles-snaps0001.vm.esrt.cloud.sbrf.ru,job_name=ALPHA.COMMON.ESBSM.XFERSYNC,job_id=2b005e66b405382d85b0c5b0d0bfaef8

Область действия метрики	Метрика	Триггер	Единица измерения	MBEAN Name
Job	Кол-во неуспешных проверок	не должно увеличиваться	шт	org.apache.flink.jobmanager.job.numberOfFailedCheckpoints:host=tkles-snaps0001.vm.esrt.cloud.sbrf.ru,job_name=ALPHA.COMMON.ESBSM.XFERSYNC,job_id=2b005e66b405382d85b0
Job	Кол-во входных сообщений в очереди	не должно увеличиваться	шт	Shuffle.Netty.Input.Buffers inputQueueLength
Job	Кол-во выходных сообщений в очереди	не должно увеличиваться	шт	Shuffle.Netty.Output.Buffers outputQueueLength
Task DSL	Количество	должно увеличив	шт	input Пример 0.event-flow-transformation-transformationStepName.input

Область действия метрики	Метрика	Триггер	Единица измерения	MBEAN Name
metrics	входящих событий	аться		для трансформации Пример 0.event-flow-aggregation-transformationStepName.input для агрегации
Task DSL metrics	Количество исходящих событий в основной поток обработки	должно увеличиваться	шт	output Пример 0.event-flow-transformation-transformationStepName.output для трансформации Пример 0.event-flow-aggregation-transformationStepName.output для агрегации
Task DSL metrics	Количество отфильтрованных событий	должно увеличиваться	шт	reject Пример 0.event-flow-transformation-transformationStepName.reject для трансформации Пример 0.event-flow-aggregation-transformationStepName.reject для агрегации

Область действия метрики	Метрика	Триггер	Единица измерения	MBEAN Name
Task DSL metrics	Количество событий обработанных с ошибкой	не должно быть	шт	error Пример 0.event-flow-transformation-transformationStepName.error для трансформации Пример 0.event-flow-aggregation-transformationStepName.error для агрегации

Часто встречающиеся проблемы и пути их устранения

Не выявлено.