SER Platform V Synapse Event replication

SEDR Сервис междоменной репликации событий

Руководство по установке

ОГЛАВЛЕНИЕ

Руководство по установке

Термины и сокращения

Термин/Аббревиатура	Определение
SEDR	Четырехбуквенный код программного компонента. Event Replication Service — сервис для репликации событий между событийными доменами, в том числе при нахождении доменов в разных сетевых зонах, из состава программного продукта Platform V Synapse Event Replication Service, основанный на технологиях Kafka Connect
EVTD	Четырехбуквенный код программного компонента. Event Transfer Service — сервис передачи событий из состава программного продукта Platform V Synapse Event Transfer Service, основанный на технологиях Apache Kafka
Bitbucket	Веб-сервис для хостинга проектов и их совместной разработки, основанный на системе контроля версий Git
Nexus	Nexus репозиторий
inventory	Хранилище конфигурационных файлов и настроек
Jenkins	Программная система для обеспечения процесса непрерывной интеграции программного обеспечения
Ansible	Программное обеспечение с открытым исходным кодом, которое автоматизирует поставку программного обеспечения, управление конфигурацией и

Термин/Аббревиатура	Определение
	развертывание приложений
Kafka producer	Брокер (или группа брокеров) в Apache Kafka, который отвечает за производство и отправку сообщений остальным брокерам-получателям (broker consumer) в распределенном кластере.
ОС	Операционная система
УЦ	Удостоверяющий центр
DN	Distinguished Name — Уникальное имя сертификата, должно быть уникальным в пределах дерева. В DName описывается содержимое атрибутов в дереве (так называемый путь навигации), требуемое для доступа к конкретной записи ИЛИ базовой (стартовой) записи поиска. DName состоит из серии RDN (Relative Distinguished Names, относительных уникальных имен), определяемых путем перемещения вверх по дереву в направлении его корневой записи (суффикса или базовой записи), и записываемых слева направо
SSH	Сетевой протокол прикладного уровня, позволяющий производить удаленное управление операционной системой и туннелирование TCP-соединений
URL (Uniform Resource Locator)	Унифицированный указатель ресурса
REST	Архитектурный стиль взаимодействия компонентов распределенного приложения в сети

Системные требования

Пререквизиты установки

- Ansible 2.9.
- Узлы SEDR обязательно проверить на возможность подключения на порт 9093 узлов сервисов продукта EVTD с которыми будет производится работа.

При использовании Jenkins (опциональный способ), дополнительно:

- должен быть доступ в Jenkins и созданы необходимые сущности в нем;
- все узлы сервиса SEDR должны быть доступны для вызова со стороны Jenkins;
- должен быть доступ в BitBucket и создан в нем проект для помещения ролей и inventory;
- должен быть доступ в Nexus и туда должны быть помещен дистрибутив SEDR;

При использовании логирования в централизованную систему журналирования (наприме р, в систему журналирования Platform V Monitor):

- необходимо узнать названия топиков централизованной системы журналирования, куда будут отправляться логи SEDR;
- перед установкой (переустановкой) новой версии SEDR заполнить параметры в конфигурационном файле **vars.yml** (подробнее см. документ *Руководство оператора*, раздел *Руководство по системному администрированию*, подраздел «Журналирование»);
- указать настройки отправляемых сообщений в файле **/conf/logback.xml**.

Требования к серверам

SEDR node

- от 2х серверов 4/8/150Гб;
- ОС Linux с версией kernel не ниже 3.10.0-327.

Настройка серверов SEDR

1. Создать пользователя **kafka**, выдать пользователю права sudoedit для создания сервисов и права для управления сервисами:

```
kafka (ALL) NOPASSWD: /bin/systemctl start <service_name>
kafka (ALL) NOPASSWD: /bin/systemctl stop <service_name>
kafka (ALL) NOPASSWD: /bin/systemctl status <service_name>
kafka (ALL) NOPASSWD: /bin/systemctl restart <service_name>
kafka (ALL) NOPASSWD: /bin/systemctl enable <service_name>
kafka (ALL) NOPASSWD: /bin/systemctl disable <service_name>
kafka (ALL) NOPASSWD: /bin/systemctl daemon-reload
```

где <service_name> соответствует имени воркера.

1. Создать разделы на диске:

`/opt/Apache/` - 10Гб, владелец kafka:kafka.

1. Создать JKS хранилище с клиент-серверным сертификатом для брокеров, подписанное УЦ, доверенным для всех клиентов.
2. Создать JKS хранилище с сертификатом для REST-endpoint (kafka-connect-rest.jks).
3. Создать JKS хранилище с сертификатом для конкретного экземпляра продукта (kafka-connect.jks). Для каждого экземпляра может использоваться как единый сертификат, так и разные.
4. Создать сервис:

под пользователем ***kafka*** создаем файл сервиса командой: `sudo /bin/sudoedit /etc/systemd/system/kafka-replicator.service`

```
    [Unit]
    Description=<описание>
    Requires=network.target remote-fs.target
    After=network.target remote-fs.target

    [Service]
    WorkingDirectory=/opt/Apache/kafka
    Type=simple
    User=kafka
    Group=kafka
    ExecStart=/opt/Apache/kafka/bin/connect-distributed-<имя воркера>.sh /opt/Apache/kafka/connect-distributed-<имя воркера>.properties
    LimitNOFILE=512000
    LimitNPROC=512000
    Restart=on-failure
    RestartSec=30


    [Install]
    WantedBy=default.target
```

Создание JKS хранилища и сертификатов

Для создания сертификата используется утилита **keytool.exe** из состава JDK. Для получ ения сертификата нужно:

1. Создать хранилище ключей и сертификатов:

```
keytool -genkey -keyalg RSA -alias Test -keystore [путь и имя файла с хранилищем ключей и сертификатов] -storepass [пароль для хранилища ключей и сертификатов] -validity 1440 -keysize 2048 -dname CN=[по правилам описанным ниже],OU=00ZZ,O=Org,L=Moscow,ST=Moscow,C=RU`
```

Например: `keytool -genkey -keyalg RSA -alias ks -keystore D:\ks.jks -storepass 23101989 -validity 1440 -keysize 2048 -dname CN=00CA0001P.TestProducer.zzzz,OU=00CA,O=Org,L=Moscow,ST=Moscow,C=RU.`

```
keytool -certreq -alias Test -keyalg RSA -file [путь и имя файла с запросом на сертификат] -keystore [путь и имя файла с хранилищем ключей и сертификатов, созданного на шаге 1]
```

1. Создать запрос на сертификат.

Например: `keytool -certreq -alias ks -keyalg RSA -file D:\testProducer.csr -keystore D:\ks.jks`.

1. Отправить запрос на сертификат в УЦ.
2. Импортировать сертификаты в хранилище ключей.
   Полученные от УЦ файл с сертификатом и файлы с корневыми сертификатами н еобходимо импортировать в хранилище ключей и сертификатов при помощи утил иты **keytool.exe**.
   4.1. Первым необходимо импортировать корневой сертификат:

```
keytool -import -alias ks1 -file [путь и имя файла с корневым сертификатом, полученным от УЦ] -keystore [путь и имя файла с хранилищем ключей и сертификатов, созданного на шаге 1]
```

4.2. Вторым необходимо импортировать сертификат УЦ:

```
keytool -import -alias ks2 -file [путь и имя файла с сертификатом УЦ] -keystore [путь и имя файла с хранилищем ключей и сертификатов, созданного на шаге 1]
```

4.3. Последним импортируется TLS-сертификат:

```
keytool -import -alias cmks -file [путь и имя файла с TLS-сертификатом, полученным от УЦ] -keystore [путь и имя файла с хранилищем ключей и сертификатов, созданного на шаге 1]
```

Формирование DN сертификата

CN = 00ZZ0001M.monitoringsystem.segment.contour.AS (пример)

Рекомендуется заполнять следующим образом:

- код ЦА - «00CA»;
- порядковый номер ключа (4 цифры) – до появления централизованной системы управления сертификатами не заполняется;
- тип ключа:
  - P - Producer (Продьюсер)
  - C - Consumer (Консьюмер)
  - S - Support monitoring (Инженер мониторинга)
  - M - Monitoring System (Система мониторинга)
  - A - Access manager (Администратор доступа)
  - E - Maintenance Engineer (Инженер сопровождения)
  - B - Broker (Брокер)
  - I - InfoSec Admin (Администратор безопасности)

  Для одного бизнес-сервиса допускается сертификат с несколькими ролями. Например сертификат системы, которая является одновременно поставщиком и потребителем событий, должен иметь тип ключа «PC» в DN сертификата.
- логин учетной записи пользователя;
- сетевой сегмент;
- тип (контур) кластера (только для брокера и администрирования, **роли producer и consumer не должны включать данный раздел**);

OU = OrganizationalUnitName

O = Organization

L = LocalityName

ST = StateOrProvinceName

C = CountryName

Установка

Версия дистрибутива

В корневом каталоге дистрибутива находится файл **SEDR-<номер дистрибутива>.info**, в котором содержится информация о версии дистрибутива. При уст

ановке SEDR на сервер данный файл появится в корне директории установки продукта. Содержимое файла **SEDR-<номер дистрибутива>.info**:

```
Info: <Наименование продукта>
Version: <Версия дистрибутива>
Date: <Дата сборки дистрибутива>
```

Например, файл **sedr-D-01.001.00-00.info** имеет следующее содержимое:

```
Info: SEDR Сервис междоменной репликации событий
Version: D-01.001.00-00
Date: 2021-12-23 14:05
```

Ручной способ установки

1. Проверить, что на сервер, с которого будет производиться установка, установлен Ansible и с него доступны все узлы сервиса SEDR.
2. Распаковать дистрибутив и поместить содержимое директории *modules* в *scripts/Ansible*.
3. Все дальнейшие операции производить из директории *scripts/Ansible*.
4. Создать свой инвентори (например, *ID*). Для этого создать директорию *ID* в папке *inventories*.
5. Создать структуру файлов в директорию, созданную на предыдущем шаге, по примеру, указанному в таблице.

Файл конфигурации	**Секция**	**Параметры**	**Описание значения**
`group_vars/all/vars.yml`		`ansible_user:` `ansible_port:`	Пользователь и порт для SSH-соединения ansible
	`replicator:audit_configs`	`audit.reporter.url:`	URL для подключения к REST-endpoint Platform V Audit SE. Если аудит не требуется, то секция replicator:audit_configs не указывается
	`replicator`	`truststore_path:`	Путь до файла с trustStore

Файл конфигурации	Секция	Параметры	Описание значения
	`replicator`	`truststore_password:`	Пароль от trustStore
	`replicator`	`keystore_path:`	Путь до файла с keyStore
	`replicator`	`keystore_password:`	Пароль от keyStore
	`replicator`	`key_password:`	Пароль от ключа в хранилище
	`replicator:rest`		Настройки безопасности для REST-endpoint
	`replicator:workers`	`name`	Наименование экземпляра продукта
	`replicator:workers`	`from_group:`	Наименование группы узлов EVTD с которых производится репликация
	`replicator:workers`	`to_group:`	Наименование группы узлов EVTD в которые производится репликация
	`replicator:workers`	`jmxport:`	Порт для подключения по JMX
	`replicator:workers`	`restport:`	Порт для подключения по

Файл конфигурации	Секция	Параметры	Описание значения
			REST
	replicator:workers	service_name:	Имя сервиса для переподнятия процесса, совпадает с наименованием экземпляра продукта
	replicator:workers	consumer_group:	Consumer group с которой производится подключение к источнику, совпадает с наименованием экземпляра продукта
	replicator:workers	suffix_for_service_topics:	Суффикс для служебных топиков в виде -<наименование экземпляра продукта>
inventory.yml	[kafka_from]		Перечень хостов EVTD с которых производится репликация в виде записей: <fqdn сервера> advertised_host= <fqdn для подключения извне> Пример: [replicator] my-host.org advertised_host= my-host.org

Файл конфигурации	Секция	Параметры	Описание значения
			my-host.org advertised_host= my-host.org my-host.org advertised_host= my-host.org
inventory.yml	[replicator]		Перечень хостов EVTD в которые производится репликация в виде записей: <fqdn сервера> advertised_host= <fqdn для подключения извне> Пример: [replicator] my-host.org advertised_host= my-host.org my-host.org advertised_host= my-host.org my-host.org advertised_host= my-host.org
inventory	[replicator]		Перечень хостов SEDR в виде записей: <fqdn сервера> advertised_host= <fqdn для подключения извне> Пример: [replicator] my-host.org advertised_host= my-host.org my-host.org advertised_host= my-host.org my-host.org

Файл конфигурации	Секция	Параметры	Описание значения
			`advertised_host=my-host.org`
Папка `ssl`			Поместить jks-хранилище сертификата узла SEDR
Папка `ssl_admin`			Поместить jks-хранилище сертификата администратора доступа SEDR

Важно: Все параметры *inventory* аннотированы, приведенные выше параметры являются теми, на которые требуется обратить внимание (стендозависимые). В случае необходимости настройки других параметров рекомендуется читать аннотации.

Использование vault для шифрования паролей

При хранении сенситивной информации в Git ее рекомендуется шифровать. Для этого можно использовать утилиту ansible-vault (идет в комплекте с пакетом ansible).

Для шифрования пароля следует выполнить команду на сервере, с которого производится развертывание SEDR:

```
ansible-vault encrypt_string -n jks_password 'ENCRYPT_STRING',
```

где `ENCRYPT_STRING` - строка, которую необходимо зашифровать, а `jks_password` - имя переменной.

При запросе вводим пароль для шифрования, а на выходе получаем то, что нужно занести в inventory:

```
jks_password: !vault |
  $ANSIBLE_VAULT;1.1;AES256
  30323632346331616266363234303338663965366539343535353133626165316564633237626536
  39323338313537393561353764633233633261333333338340a33633862383730393739353831393
  37626531383432366662303466363761616566393638306564623661323133356133613863313032
  39666535316436316 60a666136623361613863643137396663653363316139316566393366653838
  3039
```

Аналогично, возможно шифрование файлов:

```
ansible-vault encrypt <имя файла>
```

Ручное шифрование паролей в конфигурационных файлах

Для шифрования паролей используется утилита **password-encrypt-cli-1.3.jar** в составе дистрибутива SEDR и пакет java, установленный на сервере.

Для шифрования вызывается команда: `java -jar password-encrypt-cli-1.3.jar --key <ключ> --password <пароль>`

В результате выполнения команды будет выведен зашифрованный пароль:

```
Encrypted password: <зашифрованный пароль>.
```

Отказоустойчивый Kafka producer

Отказоустойчивый Kafka
producer обеспечивает возможность записи данных в топик резервного кластера, в случа е недоступности основного. При восстановлении доступности, запись возвращается на ос новной кластер.

Для подключения отказоустойчивого Kafka
producer необходимо перед развертыванием SEDR в конфигурационном файле **vars.yml** добавить блок *fallback_connectors*, где необходимо указать список всех коннекторов, для которых настраивается отказоустойчивый Kafka producer:

```
 replicator:
    fallback_connectors:
    - xxx-connector
    - yyy-connector
```

Также в файл **vars.yml** дополнительно необходимо добавить параметры для воркера, вк лючающего в себя эти коннекторы:

```
replicator:
  workers:
     - name: <имя воркера>
        additional_producer_props:
        - actor.system.shutdown.delay.ms=300000
        - retry.delay.ms=300000
        - clients=main,fallback
        - strategy=fallback
        - fallback.bootstrap.servers=host1:9093,host2:9093
```

Здесь необходимо задать *fallback.bootstrap.servers* -
список хостов подключения резервного кластера (*fallback*) в случае отказа главного (*mai n*). Параметр *actor.system.shutdown.delay.ms* -
определяет таймаут, после которого среда останавливается, если все продьюсеры оста новились; параметр *retry.delay.ms* -
определяет задержку в мс, после которой неисправный продьюсер вернется в список ак тивных продьюсеров.

В случае старта коннектора с указанными параметрами в файле **vars.yml** в конфигураци ю коннектора будет записана переменная *"ha.producer.enabled"* со значением *true*.

Создание служебных топиков

Создать служебные топики:

connect-configs-<наименование экземпляра продукта> с настройками:

```
partitions: 1
cleanup.policy=compact
```

connect-offsets-<наименование экземпляра продукта> с настройками:

```
partitions: 25
cleanup.policy=compact
```

connect-status-<наименование экземпляра продукта> с настройками:

```
partitions: 1
cleanup.policy=compact
```

audit-buffer-connect - при использовании буфера отправки событий аудита

Выдать права на чтение и запись для DN сертификата экземпляра продукта к этим топи кам.

Запуск установки

Запустить установку командой:

```
ansible-playbook -i inventories/<ID>/inventory replicator.yml --ask-vault-pass
```
Где `ID` - имя недавно созданного inventory.

Установка будет производиться на все хосты из inventory. Для ограничения списка узлов используем команду:

```
ansible-playbook -i inventories/<ID>/inventory replicator.yml --ask-vault-pass -l <узлы через запятую без пробелов>
```

Установка с помощью Jenkins

1. Распаковать дистрибутив и поместить содержимое папки *scripts* в BitBucket.
2. Создать и настроить инвентори (см. раздел *Ручной способ установки* выше) и поместить изменения в BitBucket.
3. В Jenkins создать Jenkins Pipeline с получением скриптов развертывания из BitBucket.

- Pipepline script from SCM
    - SCM - GIT;
    - repository url - ссылка на репозиторий куда поместили скрипты. Выбираем или добавляем учетные данные для доступа к BitBucket;
    - Script_path - относительный путь **SYN_custom.groovy**. Убедитесь, что не стоит галочка **Lightweight checkout**.

1. Сохранить получившийся Jenkins Pipeline и запустить его.
2. Проверить, что после запуска подгрузились дополнительные параметры.

3. При необходимости, поменять для параметров значения по умолчанию. Например, изменить имя используемых *credentials*.

Запуск установки

При запуске задания Jenkins по установке в параметрах выбирать нужный инвентори, pla ybook `replicator.yml`, а в поле `customURL` указать ссылку на дистрибутив.

Обновление

Общий подход

Поузловое обновление кластера через установку новой версии дистрибутива:

1. Проверить что в inventory в **vars.yml** установлено *cleanData = false*.
2. Произвести установку
    - При ручной установке - при помощи команды:
3. `ansible-playbook -i inventories/<ID>/inventory replicator.yml --ask-vault-pass -l <узел для обновления>`
    - При использовании Jenkins - запустить задание Jenkins по установке с параметрами:
        - выбрать нужный контур;
        - playbook - *replicator.yml*;
        - *customURL* - указать ссылку на дистрибутив;
        - *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 з адания Jenkins прервется с ошибкой *Не выбраны хосты*.
4. Убедиться, что в лог-файлах узла присутствует запись о старте сервера. Для этого на узле выполнить `cat /opt/Apache/kafka/logs/kafka-connect-<наименование экземпляра продукта>.log | grep "Finished starting connectors and tasks"` и убедиться, что вхождение строки есть.
5. Перейти к следующему узлу.

Проверка работоспособности

Скрипты установки автоматически по завершению проверяют корректность и успешност ь проведенных действий. При возникновении ошибки при ручной установке: обработка с криптом остановится, в консоль будет выведен текст ошибки. При возникновении ошибк и при автоматической установке: Jenkins Build завершиться с ошибкой, Console Output будет содержать сообщение об ошибке.

Откат

Общий подход

Поузловое обновление кластера SEDR производится через установку старой версии дис трибутива. Подробно описано в данном руководстве в разделе «Установка».

При наличии созданного ранее бэкапа, можно восстановиться из него, запустив установк у с тегом *backup_restore*.

Часто встречающиеся проблемы и пути их устранения

Не выявлено.

Чек-лист валидации установки

- убедиться, что в лог-файлах узла присутствует запись о старте сервера. Для этого на узле выполнить `cat /opt/Apache/kafka/logs/kafka-connect-<наименование экземпляра продукта>.log | grep "Finished starting connectors and tasks"` и убедиться, что вхождение строки есть;
- запросить список обработчиков: `curl -v --cert ./<наименование экземпляра продукта>.pem --key ./<наименование экземпляра продукта>.pem -k https://localhost:8090/connectors` и убедится что запрос выполнился без ошибок;
- проверить работу сервисов. Для этого требуется зайти на каждый из серверов и выполнить команду: `systemctl list-unit-files | grep -e <наименование экземпляра продукта>`. Проверить содержимое конфигурации сервиса командой `cat /etc/systemd/system/<наименование экземпляра продукта>.service`. В частности, проверить что исполняемые команды в параметрах ExecStart и ExecStop ссылаются на существующие файлы.