Arenadata™ Hadoop

Версия - v2.0.0

Руководство по работе с Knox Gateway

2019 Arenadata™

Оглавление

1 Knox Gateway. Обзор **3**

2 Архитектура развертывания Knox Gateway **5**

3 Поддерживаемые шлюзом сервисы Hadoop **6**

4 Демонстрационные примеры возможностей шлюза **8**
- 4.1 Допущения . . . 8
- 4.2 Шаги развертывания Knox Gateway в Ambari . . . 9
- 4.3 Шаги развертывания ручной установки Knox Gateway . . . 9

5 Каталоги шлюза **10**

6 Master Secret **11**

7 Ручное перераспределение кластеров **12**
- 7.1 Одновременное перераспределение всех кластеров . . . 12
- 7.2 Перераспределение конкретного кластера . . . 13

8 Запуск и остановка Knox вручную **14**

9 Включение WebSocket **15**

10 Определение топологий кластера **16**

11 Доступ к внутренним сервисам Hadoop **18**

12 Пример определения сервиса **19**

13 Проверка подключения к сервисам **21**
- 13.1 Тестирование WebHDFS . . . 21
- 13.2 Тестирование WebHCat/Templeton . . . 21
- 13.3 Тестирование Oozie . . . 22
- 13.4 Тестирование HBase/Stargate . . . 22
- 13.5 Тестирование HiveServer2 . . . 22

14 Добавление нового сервиса **23**

15 Настройка Knox SSO для Ambari **24**
- 15.1 Пример Knox SSO для Ambari . . . 24

В документе приведены сведения о шлюзе Knox Gateway и основы работы с ним.

Руководство может быть полезено администраторам, программистам, разработчикам и сотрудникам подразделений информационных технологий, осуществляющих внедрение и сопровождение кластера Hadoop.

Important: Контактная информация службы поддержки – e-mail: info@arenadata.io

Глава 1

Knox Gateway. Обзор

Knox Gateway (**Knox**) – отвечает за безопасность периметра, что позволяет предприятиям расширять доступ к **Hadoop** новых пользователей, а также поддерживать соответствие политик безопасности предприятия. **Knox** также упрощает обеспечение безопасности **Hadoop** для пользователей, имеющих доступ к данным кластера и выполняющих на нем свои задания.

Knox интегрируется с системами управления идентификацией и SSO (Single Sign-On, технология единого входа), что позволяет использовать идентификацию систем для доступа к кластерам **Hadoop**.

Шлюз **Knox** обеспечивает безопасность для кластеров **Hadoop** со следующими преимуществами (Рис.1.1.):

- *Упрощенный доступ*: расширение сервисов Hadoop REST/HTTP путем инкапсуляции Kerberos в кластер;
- *Повышенный уровень безопасности*: экспонирование доступа к сервисам Hadoop REST/HTTP, не раскрывая информации о сети, предоставляя SSL из коробки;
- *Централизованное управление*: централизованное обеспечение безопасности REST API, маршрутизация запросов к нескольким кластерам Hadoop;
- *Корпоративная интеграция*: поддержка LDAP, Active Directory, SSO, SAML и других систем аутентификации.

Knox можно использовать как с незащищенными кластерами **Hadoop**, так и с защитой **Kerberos**. В корпоративной среде, использующей кластеры с защитой **Kerberos**, шлюз **Knox** обеспечивает следующие возможности:

- Хорошо интегрируется с корпоративными решениями по управлению идентификационной информацией;
- Защищает сведения о развертывании кластера **Hadoop** (узлы и порты скрыты от конечных пользователей);
- Упрощает ряд сервисов, с которыми должен взаимодействовать клиент.

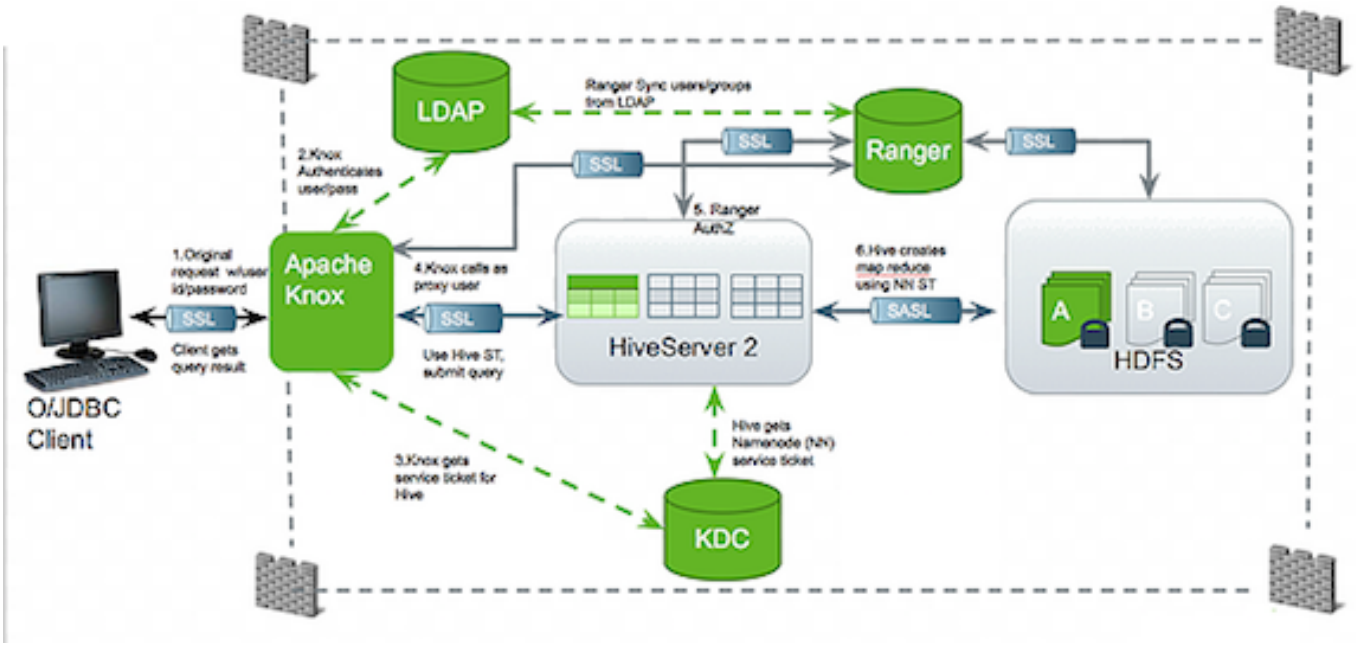


Рис.1.1.: Типичный поток безопасности

Глава 2

Архитектура развертывания Knox Gateway

Получение доступа к кластеру **Hadoop** извне осуществляется через **Knox**, **REST API** либо через интерфейс командной строки **Hadoop**.

Следующая диаграмма показывает, как **Knox Gateway** вписывается в развертывание **Hadoop**, где *NN = NameNode*, *RM = Resource Manager*, *DN = DataNote*, *NM = NodeManager* (Рис.2.1.).

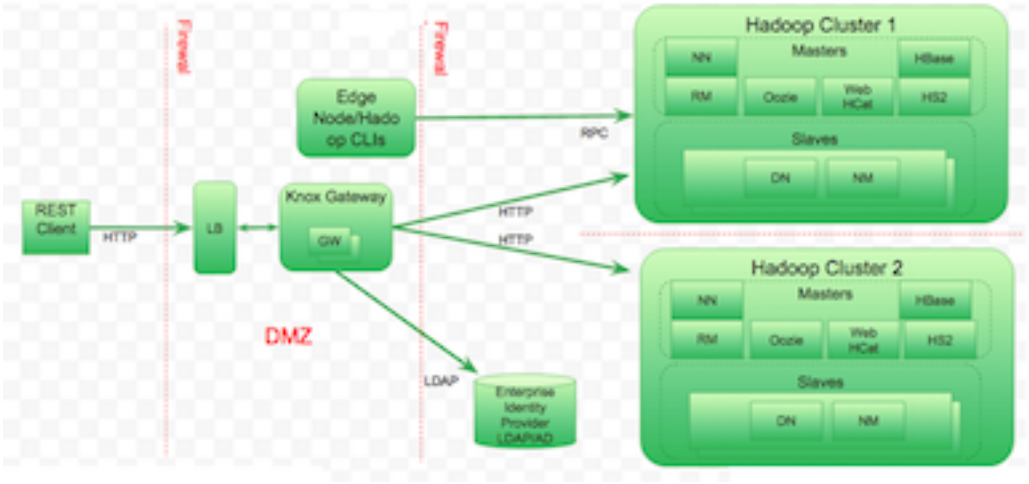


Рис.2.1.: Архитектура развертывания Knox в Hadoop

Глава 3

Поддерживаемые шлюзом сервисы Hadoop

Шлюз **Knox Gateway** поддерживает следующие сервисы **Hadoop** в кластерах как с **Kerberos**, так и без:

- Поддерживаемые API-интерфейсы компонентов (прокси-сервер):
  - YARN
  - WebHDFS
  - WebHCat/Templeton
  - Oozie
  - HBase/Stargate
  - Hive (через WebHCat)
  - Hive (через JDBC)
  - Ambari
  - Atlas
  - Ranger
  - Zeppelin
- Поддерживаемые пользовательские интерфейсы компонентов (прокси-сервер):
  - Ambari UI
  - Atlas
  - Ranger Admin Console
  - Zeppelin

Important: API и визуальные интерфейсы Knox, которые не перечислены в данной главе, рассматриваются как функционал, добавляемый Community проекта

Important: Функционал Community проекта разрабатывается и тестируется, но официально не поддерживается компанией Arenadata. Эти функции исключаются по ряду причин, включая недостаточную надежность или неполное покрытие тестового сценария. Подобные функции не рекомендуется использовать в рабочей среде

Глава 4

Демонстрационные примеры возможностей шлюза

Существует несколько методов развертывания **Knox** в кластере, отличие которых заключается в разных способах установки и настройки шлюза. Примеры в каталоге *{GATEWAY_HOME}/samples* демонстрируют возможности шлюза **Knox** для обеспечения доступа к многочисленным API-интерфейсам из сервисных компонентов кластера **Hadoop**.

В зависимости от того, как именно выполняется установка **Knox**, требуется определенное количество шагов для полной установки и настройки примеров использования.

4.1 Допущения

Приведенные примеры написаны с целью разработки стандартных демонстрационных сред **Hadoop**, развернутых в виде кластера с одним узлом внутри виртуальной машины. В связи с этим сделаны следующие допущения:

- Существует действительный java JDK в пути PATH для выполнения примеров;
- Сервер Knox Demo LDAP запущен на “localhost” с портом “33389” по умолчанию для сервера ApacheDS LDAP;
- В используемом каталоге LDAP есть набор демо-пользователей, которым предоставляется условное обозначение имени пользователя “username” и пароль “password”. Так же примеры имеют вариации с “guest” и “guest-password”;
- Экземпляр Knox Gateway запущен на той же машине, на которой планируется запуск примеров, поэтому используется “localhost” и порт по умолчанию “8443”;
- В каталоге *{GATEWAY_HOME}/conf/topologies* имеется корректно созданная топология *sandbox.xml*, которая настроена так, чтобы указывать на фактический хост и порты запущенных компонентов сервиса.

В подготовленной демонстрационной среде кластера с одним узлом для иллюстрации использования **Knox** необходимо обеспечить:

- Корректно настроенную топологию *sandbox.xml* для развернутых сервисов;
- Сервер LDAP, запущенный с пользователем *guest/guest-password*.

4.2 Шаги развертывания Knox Gateway в Ambari

Инстансы **Knox**, находящиеся под управлением **Ambari**, как правило, не считаются демо-экземплярами. Данные инстансы предназначены для облегчения разработки и тестирования или создания кластеров **Hadoop**. Однако их можно заставить работать с инстансами **Knox** под управлением **Ambari** за несколько шагов:

1. Необходимо иметь доступ через ssh.
2. Запустить сервер Knox Demo LDAP (можно запустить его из Ambari).
3. Файл топологии *default.xml* скопировать в файл *sandbox.xml*.
4. Обязательно использовать фактически существующую Java JRE для запуска примера, на подобии:

```
/usr/jdk64/jdk1.8.0_67/bin/java -jar bin/shell.jar samples/ExampleWebHdfsLs.groovy
```

4.3 Шаги развертывания ручной установки Knox Gateway

Для установленных вручную инстансов **Knox** необходимо выполнить идентичные шаги, как для развертывания инстанса через **Ambari**, за исключением пункта *3*:

1. Необходимо иметь доступ к среде ssh.
2. Запустить сервер Knox Demo LDAP (можно запустить его из Ambari).
3. Изменить хосты и порты в *{GATEWAY_HOME}/conf/topologies/sandbox.xml* для отражения фактического расположения сервиса кластера.
4. Обязательно использовать фактически существующую Java JRE для запуска примера, на подобии:

```
/usr/jdk64/jdk1.8.0_67/bin/java -jar bin/shell.jar samples/ExampleWebHdfsLs.groovy
```

Глава 5

Каталоги шлюза

При установке **Knox Gateway** создаются следующие каталоги:

- *HADOOP_NODE_INSTALL_ROOT*;
- *knox-X.X.X.X.X.X.X-XXXX* – каталог *$gateway*. Например, каталог */usr/lib/knox* содержит файлы, описание которых приведено далее в таблице;
- */var/log/knox* – содержит выходные файлы шлюза **Knox**.

Таблица5.1.: Файлы каталога шлюза

Каталог/Имя файла	Описание
conf/topologies	Содержит общие файлы конфигурации шлюза
bin	Содержит исполняемые скрипты командной оболочки, пакетные файлы и JAR-файлы для клиентов и серверов
deployments	Содержит файлы дескрипторов топологии кластера, определяющие кластеры Hadoop
lib	Содержит JAR-файлы для всех компонентов, составляющих шлюз
dep	Содержит JAR-файлы для всех компонентов, от которых зависит шлюз
ext	Каталог, в котором пользовательские расширения JAR могут быть размещены для расширения функциональности шлюза
samples	Содержит ряд примеров, которые можно использовать для изучения функциональности шлюза
templates	Содержит файлы конфигурации по умолчанию, которые можно скопировать и настроить
README	Предоставляет основные сведения о шлюзе Knox Gateway
ISSUES	Описывает известные проблемы
CHANGES	Перечисляет изменения между версиями
LICENSE	Документирует лицензию, по которой предоставляется программное обеспечение
NOTICE	Документы, необходимые для уведомлений об атрибутах для включенных зависимостей
DISCLAIMER	Документы, подтверждающие, что данный выпуск из проекта, проходящего инкубацию в компании Arenadata

Глава 6

Master Secret

Для запуска **Knox Gateway** необходим Master Secret, защищающий артефакты, используемые инстансом шлюза – keystore, trust stores и хранилища учетных данных.

Important: Важно настроить сохранение в файле *$gateway /data/security/master* и убедиться, что в данном каталоге установлены соответствующие разрешения для среды выполнения

Установка Master Secret осуществляется по команде:

```
cd $gateway bin/knoxcli.cmd create-master
```

При этом появляется предупреждение, указывающее, что сохранение секрета менее безопасно, чем предоставление его при запуске. **Knox** защищает пароль, шифруя его с помощью 128-битного шифрования AES, и, где это возможно, разрешения на доступ к файлам есть только у пользователя *knox*.

Important: Важно убедиться, что каталог безопасности *$gateway/data/security* и его содержимое доступно для чтения и записи только пользователю *knox*. Это самый важный уровень защиты для Master Secret. Нельзя предполагать, что для защиты достаточно только шифрования

В критических ситуациях Master Secret может быть изменен, тогда администратору необходимо повторить все настройки для каждого инстанса шлюза при развертывании. Обновление Master Secret требует не только пересоздания мастера, но и также удаления всех существующих хранилищ ключей и переподготовку сертификатов и учетных данных.

Изменение Master Secret осуществляется по команде:

```
cd $gateway bin/knoxcli.cmd create-master--force
```

При наличии существующего хранилища ключей следует обновить его после выполнения команды.

Глава 7

Ручное перераспределение кластеров

После обновления свойств кластера повторное развертывание кластеров вручную не требуется. Шлюз отслеживает файлы дескрипторов топологии в каталоге *$gateway/conf/topologies* и автоматически перераспределяет кластер при изменении или добавлении дескриптора (соответствующее развертывание находится в *$gateway/data/deployments*).

Однако, после изменения любого из нижеперечисленных свойств или параметров шлюза требуется ручное перераспределение кластеров:

- Настройка времени на хосте шлюза;
- Внедрение или обновление Kerberos;
- Внедрение или обновление SSL-сертификатов;
- Изменение алиаса кластера.

7.1 Одновременное перераспределение всех кластеров

При внесении изменений на уровне шлюза (например, при реализации протокола Kerberos или SSL) или при изменении системного времени необходимо повторно развернуть все топологии кластера:

1. Для проверки временной метки на развернутых кластерах необходимо ввести: *cd $gatewaydir data/deployments*. Система при этом отображает информацию на подобии следующей:

```
Directory of /usr/lib/knox/data/deployments


.
..
cluster.war.145514f4dc8
myCluster.war.145514f4dc8
sandbox.war.145514f4dc8
```

2. Для повторного развертывания всех кластеров необходимо ввести команду:

   `/bin/knoxcli.cmd redeploy`

3. Для проверки вновь созданных кластеров использовать команду: *cd $gatewaydir data/deployments*. Система при этом отображает информацию, из которой видно, что для каждого кластера создается новый файл с текущей временной меткой:

```
Directory of /usr/lib/knox/data/deployments
```

```
.
..
cluster.war.145514f4dc8
cluster.war.1457241b5dc
myCluster.war.145514f4dc8
myCluster.war.1457241b5dc
sandbox.war.145514f4dc8
sandbox.war.1457241b5dc
```

7.2 Перераспределение конкретного кластера

При внесении изменений, влияющих на один конкретный кластер (например, при изменении алиаса или восстановлении кластера из более раннего файла дескриптора топологии), необходимо перераспределить только данный кластер. Для этого следует выполнить действия:

1. Для проверки временной метки в WAR-файлах топологии развернутого кластера необходимо ввести команду: *cd $gatewaydir data/deployments*. Система при этом отображает информацию на подобии следующей:

```
Directory of /usr/lib/knox/data/deployments


.
..
cluster.war.145514f4dc8
myCluster.war.145514f4dc8
sandbox.war.145514f4dc8
```

2. Для повторного развертывания определенного кластера необходимо ввести команду:

   `cd $gateway bin/knoxcli.cmd redeploy --cluster $cluster_name`

   Где *$cluster_name* – имя дескриптора топологии кластера (без расширения *.xml*). Например, *myCluster*.

3. Для проверки факта развертывания кластера использовать команду: *cd $gatewaydir data/deployments*. Система при этом отображает информацию на подобии следующей:

```
Directory of /usr/lib/knox/data/deployments


.
..
cluster.war.145514f4dc8
myCluster.war.145514f4dc8
myCluster.war.1457241b5dc
sandbox.war.145514f4dc8
```

Должно прослеживаться, что war-файлы кластера не изменились, за исключением обновленного файла для *myCluster* (имеет текущую временную метку).

Глава 8

Запуск и остановка Knox вручную

За исключением изменений в `../ ../conf/topology/*.xml`, изменения общих настроек **Knox Gateway** в *$gateway /conf/gateway-site.xml* загружаются только после перезапуска шлюза.

Остановка **Knox** вручную осуществляется по команде:

```
cd $gateway/bin/gateway.sh stop
```

Данный способ называется “чистым” завершением работы, так как скрипт шлюза очищает все файлы *.out* и *.err* в каталоге журналов.

Первичный или повторный запуск **Knox** после завершения очистки осуществляется по команде:

```
cd $gateway /bin/gateway.sh start
```

Перезапуск **Knox** после “нечистого” отключения осуществляется по команде:

```
cd $gateway/bin/gateway.sh clean /bin/gateway.sh start
```

Данная команда удаляет старые файлы *.out* и *.err* в каталоге журналов.

Глава 9

Включение WebSocket

Включение протокола связи WebSocket для **Knox Gateway** допускает применение прокси-приложений, использующих данное соединения (например, **Zeppelin**).

WebSocket – это протокол связи, позволяющий осуществлять полнодуплексную связь по одному TCP-соединению. **Knox** предоставляет встроенную поддержку WebSocket, но в настоящее время поддерживаются только текстовые сообщения.

По умолчанию в **Knox Gateway** функция WebSocket отключена. Однако для работы сервиса **Zeppelin UI** (`<role>ZEPPELINUI</role>`) данная функциональность должна быть включена; для этого необходимо:

1. В файле */conf/gateway-site.xml* изменить значение параметра *gateway.websocket.feature.enabled* на *true*:

```
<property>
      <name>gateway.websocket.feature.enabled</name>
      <value>true</value>
      <description>Enable/Disable websocket feature.</description>
  </property>
```

2. В файле */conf/{topology}.xml* изменить правило топологии:

```
<service>
      <role>WEBSOCKET</role>
      <url>ws://myhost:9999/ws</url>
  </service>
```

3. Перезапустить шлюз по команде:

```
cd $gateway bin/gateway.sh stop bin/gateway.sh start
```

Глава 10

Определение топологий кластера

Knox Gateway осуществляет поддержку одного или нескольких кластеров **Hadoop**. Каждая конфигурация кластера определяется в файле дескриптора топологии в каталоге шлюза *$gateway/conf/topologies* и разворачивается в соответствующем WAR-файле в каталоге *$gateway/data/deployments*. Данные файлы определяют, как шлюз взаимодействует с каждым кластером **Hadoop**.

Дескриптор представляет собой XML-файл, содержащий:

- *gateway/provider* – параметры конфигурации, установленные шлюзом **Knox** при предоставлении доступа к кластеру **Hadoop**;
- *service* – URL-адреса сервиса **Hadoop**, используемые шлюзом для прокси-связи с внешними клиентами.

Шлюз автоматически перераспределяет кластер при обнаружении нового файла дескриптора топологии или при обнаружении изменений в существующем файле. Далее в таблице представлен обзор провайдеров и сервисов.

Таблица10.1.: Провайдеры и сервисы

Тип	Роль	Описание
gateway/provider	hostmap	Сопоставляет внешние и внутренние имена узла, заменяя внутреннее имя узла сопоставленным внешним именем, когда имя узла включено в ответ из кластера
	authentication	Интегрирует хранилище LDAP для аутентификации внешних запросов, обращающихся к кластеру через шлюз Knox
	federation	Определяет поля аутентификации HTTP-заголовка для SSO и поставщика решений
	identity-assertion	Отвечает за способ подтверждения идентификации подлинности пользователя в сервисе, для которого предназначается запрос. Также отображает внешних пользователей внутреннего кластера, прошедших проверку подлинности, которых шлюз утверждает как пользователей или группу текущей сессии
	authorization	Авторизация на уровне сервиса, ограничивающая доступ к кластеру указанным пользователям, группам и/или IP-адресам
	webappspec	Настраивает плагин безопасности веб-приложений, обеспечивающий защиту от межсайтовой подделки запроса
HA provider	high availability	Синхронизирует все инстансы Knox для использования одних и тех же хранилищ ключей учетных данных топологии
service	$service_name	Связывает сервис Hadoop с внутренним URL-адресом, используемым шлюзом для прокси-запросов от внешних клиентов к внутренним сервисам кластера

Дескрипторы топологии кластера имеют следующий формат XML:

```
<topology>
    <gateway>
        <provider>
            <role></role>
            <name></name>
            <enabled></enabled>
            <param>
                <name></name>
                <value></value>
            </param>
        </provider>
    </gateway>
    <service></service>
</topology>
```

Глава 11

Доступ к внутренним сервисам Hadoop

Настройка доступа к внутреннему сервису **Hadoop** через **Knox Gateway** осуществляется в два шага:

1. Необходимо изменить файл шлюза *$gateway/conf/topologies$cluster-name.xml*, добавив для каждого сервиса **Hadoop** запись, аналогичную следующей:

```
<topology>
    <gateway>
     ...
     </gateway>
     <service>
         <role> $service_name </role>
         <url> $schema://$hostname:$port</url>
     </service>
</topology>
```

Где:

- *$service_name* – название сервиса – *AMBARI*, *AMBARIUI*, *ATLAS*, *HIVE*, *JOBTRACKER*, *NAMENODE*, *OOZIE*, *RANGER*, *RANGERUI*, *RESOURCEMANAGER*, *WEBHBASE*, *WEBHCAT*, *WEBHDFS*, *ZEPPELINUI* или *ZEPPELINWS*;
- *<url>* – полный внутренний URL-адрес кластера для доступа к сервису, включая:
  - *$schema* – протокол сервиса;
  - *$hostname* – разрешенное имя внутреннего узла;
  - *$port* – порт прослушивания сервиса.

2. Сохранить файл. При этом шлюз создает новый WAR-файл с измененной временной меткой в *$gateway/data/deployments*.

Перезапуск сервера **Knox** после внесения изменений в топологию или сервисы кластера **Hadoop** не требуется.

Глава 12

Пример определения сервиса

При настройке каждого сервиса кластера, который требуется экспонировать, важно тщательно определять их внутренние имена хостов и порты. В следующем примере используются порты по умолчанию и поддерживаемые имена сервисов.

```
<service>
    <role>AMBARI</role>
    <url>http://ambari-host:8080</url>
</service>

<service>
    <role>AMBARIUI</role>
    <url>http://ambari-host:8080</url>
</service>

<service>
    <role>ATLAS</role>
    <url>http://atlas-host:8443</url>
</service>

<service>
    <role>HIVE</role>
    <url>http://hive-host:10001/cliservice</url>
</service>

<service>
    <role>JOBTRACKER</role>
     <url>rpc://jobtracker-host:8050</url>
</service>

<service
    <role>NAMENODE</role>
     <url>hdfs://namenode-host:8020</url>
</service>

<service>
    <role>OOZIE</role>
    <url>http://oozie-host:11000/oozie</url>
</service>

<service>
    <role>RANGER</role>
```

```
    <url>http://ranger-host:6080</url>
</service>

<service>
    <role>RANGERUI</role>
    <url>http://ranger-host:6080</url>
</service>

<service>
    <role>RESOURCEMANAGER</role>
    <url>http://hive-host:8088/ws</url>
</service>

<service>
    <role>WEBHBASE</role>
    <url>http://webhbase-host:60080</url>
</service>

<service>
    <role>WEBHCAT</role>
    <url>http://webcat-host:50111/templeton</url>
</service>

<service>
    <role>WEBHDFS</role>
    <url>http://webhdfs-host:50070/webhdfs</url>
</service>

<service>
    <role>ZEPPELINUI</role>
    <url>http://zeppelin-host:9995</url>
</service>

<service>
    <role>ZEPPELINWS</role>
    <url>http://zeppelin-host:9995/ws</url>
</service>
```

Глава 13

Проверка подключения к сервисам

Приведенные далее команды используются для проверки подключения между узлом шлюза и сервисами **Hadoop**, а так же для проверки подключения внешнего клиента к сервисам **Hadoop** через шлюз.

Important: При сбое связи между узлом шлюза и внутренним сервисом Hadoop, необходимо использовать telnet для подключения к порту сервиса, чтобы убедиться, что шлюз имеет доступ к узлу кластера. При этом необходимо использовать имя хоста и порты, указанные в определении сервиса

13.1 Тестирование WebHDFS

Тестирование **WebHDFS** путем получения пути к домашнему каталогу:

- На хосте шлюза используется команда:

```
curl http://$webhdfs-host:50070/webhdfs/v1?op=GETHOMEDIRECTORY
```

- На внешнем клиенте используется команда:

```
curl https://$gateway-host:$gateway_port/$gateway/$cluster_name/$webhdfs_service_name/v1?
↪op=GETHOMEDIRECTORY
```

При этом хост шлюза и внешний клиент реагируют отображением информации:

```
{"Path":"/user/gopher"}
```

13.2 Тестирование WebHCat/Templeton

Тестирование **WebHCat**/**Templeton** путем получения номера версии:

- На хосте шлюза используется команда:

```
curl http://$webhdfs-host:50111/templeton/v1/version
```

- На внешнем клиенте используется команда:

```
curl https://$gateway-host:$gateway_port/$gateway/$cluster_name/$webhcat_service_name/v1/version
```

При этом хост шлюза и внешний клиент реагируют отображением информации:

```
{"supportedVersions":["v1"],"version":"v1"}
```

13.3 Тестирование Oozie

Тестирование **Oozie** путем получения номера версии:

- На хосте шлюза используется команда:

```
curl http://$oozie-host:11000/oozie/v1/admin/build-version
```

- На внешнем клиенте используется команда:

```
curl https://$gateway-host:$gateway_port/$gateway/$cluster_name/$oozie_service_name/v1/admin/build-
↪version
```

При этом хост шлюза и внешний клиент реагируют отображением информации:

```
{"buildVersion":"4.0.0.2.1.1.0-302"}
```

13.4 Тестирование HBase/Stargate

Тестирование **HBase**/**Stargate** путем получения номера версии:

- На хосте шлюза используется команда:

```
curl http://$hbase-host:17000/version
```

- На внешнем клиенте используется команда:

```
curl http://$hbase-host:17000/version
```

При этом хост шлюза и внешний клиент реагируют отображением информации:

```
rest 0.0.2 JVM: Oracle Corporation 1.7.0_51-24.45-b08 OS: Linux 3.8.0-29-generic amd64␣
↪Server:jetty/6.1.26 Jersey:1.8
```

13.5 Тестирование HiveServer2

Тестирование **HiveServer2** посредством URL-адресов:

- На хосте шлюза используется команда:

```
curl http://$hive-host:10001/cliservice
```

- На внешнем клиенте используется команда:

```
curl https://$gateway-host:$gateway_port/$gateway/$cluster_name/$hive_service_name/cliservice
```

При этом хост шлюза и внешний клиент возвращают ошибку проверки подлинности, которую можно игнорировать.

Глава 14

Добавление нового сервиса

В шлюзе **Knox Gateway** новые сервисы и их дополнения реализуются как расширение существующей функциональности.

Knox Gateway поддерживает декларативный способ легкого “подключения” нового сервиса к шлюзу, используя следующие два файла:

- *service.xml* – файл, содержащий предоставляемые сервисом маршруты (пути) и правила перезаписи для привязки этих путей;
- *rewrite.xml* – файл, содержащий правила перезаписи для сервиса.

Important: Файл *service.xml* является обязательным, *rewrite.xml* – опциональным

Глава 15

Настройка Knox SSO для Ambari

В разделе описывается, как настроить **Ambari** на использование в шлюзе **Knox** технологии единого входа для аутентифицированного пользователя (SSO, Single Sign-on). При такой конфигурации не прошедшие проверку подлинности пользователи, которые пытаются получить доступ к **Ambari**, перенаправляются на страницу авторизации.

Настройка **Knox SSO** для **Ranger** выполняется следующими действиями:

1. Выполнить вход в систему под пользователем *root*.
2. Выполнить команду:

```
ambari-server setup-sso
```

3. На выпадающий запрос ответить *y*.
4. Ввести URL-адрес:

```
https://<hostname>:8443/gateway/knoxsso/api/v1/websso
```

5. Выполнить следующую команду CLI для экспорта сертификата **Knox**:

```
JAVA_HOME/bin/keytool -export -alias gateway-identity -rfc -file <cert.pem> -keystore /usr/lib/
↪knox-server/data/security/keystores/gateway.jks
```

При появлении запроса ввести пароль мастера **Knox**. Обратить внимание на место сохранения файла *cert.pem*.

6. При появлении запроса на настройку дополнительных свойств ответить *n*.
7. Оставить поля *JWT Cookie name (hadoop-jwt)* и *JWT audiences list* пустыми. Запрос возвращает успешное завершение *Ambari Server 'setup-sso' completed successfully.*
8. Перезапустить **Ambari Server**: `ambari-server restart`.

15.1 Пример Knox SSO для Ambari

```
ambari-server setup-sso
Setting up SSO authentication properties...
Do you want to configure SSO authentication [y/n] (y)?y
Provider URL [URL] (http://example.com):https://c6402.ambari.apache.org:8443/gateway/knoxsso/api/
↪v1/websso
Public Certificate pem (empty) (empty line to finish input):
```

```
MIICYTCCAcqgAwIBAgIIHd3j94bX9IMwDQYJKoZIhvcNAQEFBQAwczELMAkGA1UEBhMCVVMxDTAL
BgNVBAgTBFRlc3QxDTALBgNVBAcTBFRlc3QxDzANBgNVBAoTBkhhZG9vcDENMAsGA1UECxMEVGVz
dDEmMCQGA1UEAxMda25veHNzby1za29uZXJ1LTItMi5ub3ZhbG9jYWwwHhcNMTYwMzAxMTEzMTQ0
WhcNMTcwMzAxMTEzMTQ0WjBzMQswCQYDVQQGEwJVUzENMAsGA1UECBMEVGVzdDENMAsGA1UEBxME
VGVzdDEPMA0GA1UEChMGSGFkb29wMQ0wCwYDVQQLEwRUZXN0MSYwJAYDVQQDEx1rbm94c3NvLXNr
b25lcnUtMi0yLm5vdmFsb2NhbDCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAlVOJtd8zmzVZ
UZRbqxXvK9MV5OYIOWTX9/FMthwr99eClHp3JdZ1x3utYr9nwdZ6fjZaUIihzu8a8SGoipbW2ZVU
TShGZ/5VKtu96YcSAoB3VTyc3WWRDGERRs7aKAlEqnURDkQz7KRs2tvItJpBBjrTXZpHKFTOecL4
hCkaalUCAwEAATANBgkqhkiG9w0BAQUFAAOBgQAqvPfl4fivozd+4QI4ZBohFHHvf1z4Y7+DxlY7
iNAnjnau4W3wgwTt6CQ1B9fSx3zVTlhu2PfDJwvumBbuKuth/M+KXpG28AbKIojrL2Odlv+cftrJ
YeJC6Qjee+5Pf2P9G2wd9fahWF+aQpr5OYlMZSU+VMiTO2a2FSAXvOdjvA==

Do you want to configure advanced properties [y/n] (n) ?y
JWT Cookie name (hadoop-jwt):
JWT audiences list (comma-separated), empty for any ():
Ambari Server 'setup-sso' completed successfully.

ambari-server restart
```