IBM Financial Transactions Repository
Version 2.0.3

IBM Financial Transactions Repository Installation Guide

Note

Before using this information and the product it supports, read the information in Notices.

Product Information

This document applies to Version 2.0.3 and may also apply to subsequent releases.

Copyright

Contents

Chapter 1. Installing IBM Financial Transactions Repository

Preparing for installation

Before you install and configure IBM Financial Transactions Repository, ensure that the following pre-installation tasks are completed and system requirements are met.

You must ensure that:

- All servers are configured with networking and ping-able.
- All servers are defined in the `/etc/hosts` file. Ensure that you add all of the server names in the `/etc/hosts` for each computer as follows:

  For a small topology that uses 3 servers:

  ```
  x.x.x.x   ambarihostname.domainname          ambarihostname
  x.x.x.x   hadoopmasterhostname.domainname    hadoopmasterhostname
  x.x.x.x   hadoopgatewayhostname.domainname   hadoopgatewayhostname
  ```

  For a medium topology that uses 6 servers:

  ```
  x.x.x.x   ambarihostname.domainname          ambarihostname
  x.x.x.x   hadoopmasterhostname.domainname    hadoopmasterhostname
  x.x.x.x   hadoopsecondaryhostname.domainname hadoopsecondaryhostname
  x.x.x.x   hadoopslave1hostname.domainname    hadoopslave1hostname
  x.x.x.x   hadoopslave2hostname.domainname    hadoopslave2hostname
  x.x.x.x   hadoopgatewayhostname.domainname   hadoopgatewayhostname
  ```

- Ensure that `files` is listed first on the hosts: entry in the Name Service Switch configuration file (`/etc/nswitch.conf`).

  For example, run the following command:

  ```
  cat /etc/nsswitch.conf | grep "hosts"
  ```

  The output should show:

  ```
  hosts: files dns myhostname
  ```

Installing MySQL

You must install MySQL for IBM Financial Transactions Repository.

For a small topology installation, install MySQL on the hadoop.master node computer. For a medium topology installation, install it on the hadoop.secondary node computer.

Procedure

1. Log in to the hadoop.master or the hadoop.secondary computer.
2. Run the following command to create a `mysql.repo` in the `/etc/yum.repos.d` directory.

   ```
   vi /etc/yum.repos.d/mysql.repo
   ```

3. Add the following text to the `mysql.repo` file:

   ```
   [mysql-connectors-community]
   name=MySQL Connectors Community
   baseurl=http://repo.mysql.com/yum/mysql-connectors-community/el/7/$basearch/
   ```

```
enabled=1
gpgcheck=0
gpgkey=file:/etc/pki/rpm-gpg/RPM-GPG-KEY-mysql
[mysql-tools-community]
name=MySQL Tools Community
baseurl=http://repo.mysql.com/yum/mysql-tools-community/el/7/$basearch/
enabled=1
gpgcheck=0
gpgkey=file:/etc/pki/rpm-gpg/RPM-GPG-KEY-mysql

# Enable to use MySQL 5.5
[mysql55-community]
name=MySQL 5.5 Community Server
baseurl=http://repo.mysql.com/yum/mysql-5.5-community/el/7/$basearch/
enabled=0
gpgcheck=0
gpgkey=file:/etc/pki/rpm-gpg/RPM-GPG-KEY-mysql

# Enable to use MySQL 5.6
[mysql56-community]
name=MySQL 5.6 Community Server
baseurl=http://repo.mysql.com/yum/mysql-5.6-community/el/7/$basearch/
enabled=1
gpgcheck=0
gpgkey=file:/etc/pki/rpm-gpg/RPM-GPG-KEY-mysql
```

4. Save and close the file.
5. Ensure that you do not have the mariadb packages installed.

   If you do, you must remove them. You can check to see if you have the mariadb packages installed by running the following command:

   ```
   rpm -qa | grep mariadb
   ```

   If you do have the packages installed, run the following command to remove them:

   ```
   rpm -e --nodeps <package_name>
   ```

6. Run the following commands to install MySQL:

   ```
   yum -y install mysql-community-release-el7-5.noarch
   ```

   ```
   yum -y install mysql-community-common-5.6.39-2.el7.x86_64
   ```

   ```
   yum -y install mysql-community-libs-5.6.39-2.el7.x86_64
   ```

   ```
   yum -y install mysql-community-client-5.6.39-2.el7.x86_64
   ```

   ```
   yum -y install mysql-community-server-5.6.39-2.el7.x86_64
   ```

Downloading the Hadoop media files to the Ambari server

You must install the Hadoop media files to the computer that will host the Ambari server.

Procedure

1. Create a directory for the media files.

   ```
   mkdir -p /opt/ibm/fci/install/media
   ```

2. Go to the directory:

   ```
   cd /opt/ibm/fci/install/media
   ```

3. Run the following commands to download the media files:

   ```
   wget https://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-5.1.46.tar.gz
   ```

   ```
   wget http://public-repo-1.hortonworks.com/HDP/centos7/2.x/updates/2.6.4.0/HDP-2.6.4.0-
   centos7-rpm.tar.gz
   ```

   ```
   wget http://public-repo-1.hortonworks.com/HDP-UTILS-1.1.0.22/repos/centos7/HDP-
   UTILS-1.1.0.22-centos7.tar.gz
   ```

   ```
   wget http://public-repo-1.hortonworks.com/HDP-GPL/centos7/2.x/updates/2.6.4.0/HDP-
   GPL-2.6.4.0-centos7-rpm.tar.gz
   ```

   ```
   wget http://public-repo-1.hortonworks.com/ambari/centos7/2.x/updates/2.6.1.5/ambari-2.6.1.5-
   centos7.tar.gz
   ```

Downloading and running the Financial Crimes Insight Hadoop installer

You must download IBM Financial Crimes Insight with Watson, Private platform 1.0.3 parts. These parts contain the Hadoop installation files that you must use for IBM Financial Transactions Repository.

Procedure

1. Download the following eImage and save the file to a temporary directory.

   Table 1: Hadoop installer download

Name	eImage part number	Downloaded file name
IBM Financial Crimes Insight Core Services Hadoop Installer (14 of 14) 1.0.3 Multiplatform English	CNT2TEN	fci-core-103-14of14.tar

   For more information about the available downloads, see Downloading IBM Surveillance Insight for Financial Services (www.ibm.com/support/docview.wss?uid=swg24042930).
2. Go to the directory where you download the file, and extract the downloaded installation package:

   ```
   tar -xvf fci-core-103-14of14.tar
   ```

   Untarring the file creates a file that is named `fci-hadoop-install-rpm.rpm`.
3. Enter the following command to install the RPM file:

   ```
   rpm -ivh fci-hadoop-install-rpm.rpm
   ```

   This command copies the installation files to the `/opt/ibm/fci/install` directory.
4. Go to the `/opt/ibm/fci/install` directory.
5. Edit the appropriate `hosts.properties` file for your deployment.

   If you are doing a small deployment, edit `fci-hadoop-1.0.3.small.hosts.properties`. If you are doing a medium deployment, edit `fci-hadoop-1.0.3.medium.hosts.properties`.

   Ensure that the host names match what you have in the `/etc/hosts` file.

   Note: Do not change the role name that is used in the `hosts.properties` file. For example, do not change `ambari`, `hadoop.gateway`, etc.
6. Open the `fci-hadoop-1.0.3.properties` file in a text editor.

   a) Change the fcai.environment.size value to 3 for a small deployment or 6 for a medium deployment.

   b) Change the fcai.database.host entry value to the host name of the Ambari server.

c) Change the fcai.logstash.host entry value to the host name of the Ambari server.

d) Save and close the file.

7. Run the following command to create the Hadoop cluster:

```
./fci-hadoop-1.0.3.create.cluster.sh 2>&1 | tee "createcluster.`date +%F_%T`.log"
```

Verify the installation

You can verify the installation of Ambari and Kafka by logging in to the Ambari console and creating a Kafka topic.

Procedure

1. In a web browser, enter the Ambari console URL: `https://`*`ambari.server`*`:8081`
2. Enter the user credentials. The default is `admin` / `admin`.

Enable security and data encryption

Data must be encrypted between the Kafka server and client components.

Securing data in motion for Apache Kafka

You must create a secure key and keystore to be able to encrypt and decrypt messages with Apache Kafka.

Procedure

1. Log on to the computer where Apache Kafka is installed as the root user.
2. Create a key and a keystore for each Kafka broker.

   ```
   keytool -genkey -alias SIKafkaServerSSL -validity 365 -keystore
   SIKafkaServerSSLKeystore.jks -dname "CN=si.ibm.com,O=IBM,OU=IBMAnalytics,L=IN,ST=ON,C=CA" -
   keypass YourKeyPassword
   ```

3. When prompted, enter a password for the key. For example, enter `sifs123`.
4. Export the certificate from the keystore.

   ```
   keytool -certreq -file SIKafkaCert -alias SIKafkaServerSSL -keystore
   SIKafkaServerSSLKeystore.jks
   ```

5. When prompted, enter the password that you used. For example, enter `sifs123`.

   Note: The certificate must be signed by a certificate authority.

6. Generate the certificate authority key.

   ```
   openssl req -new -x509 -keyout ca-key -out ca-cert -days 365
   ```

   Follow the prompts to generate the key.

7. Add the key to the server truststore.

   ```
   keytool -import -file ca-cert -keystore SIKafkaServerSSLKeystore.jks -alias CARoot
   ```

   The truststore is automatically created.

8. Add the key to the server keystore.

   ```
   keytool -import -file ca-cert -keystore SIKafkaServerSSLKeystore.jks -alias CARoot
   ```

9. Sign the certificate:

   ```
   openssl x509 -req -CA ca-cert -CAkey ca-key -in SIKafkaCert -out SIKafkaCertSigned -days
   365 -CAcreateserial -passin pass:YourPassword
   ```

10. Import the signed certificate into the server keystore:

    ```
    keytool -import -file SIKafkaCertSigned -keystore SIKafkaServerSSLKeystore.jks -alias
    SIKafkaServerSSL
    ```

11. Update the `KafkaInstallLocation/config/server.properties` file to include the following text:

    ```
    listeners=SSL://<IP>:<Port>
    advertised.listeners=SSL://<IP>:<Port>
    ssl.keystore.location=/home/SIUser/SIKafkaServerSSLKeystore.jks
    ssl.keystore.password=YourPassword
    ssl.key.password= YourKeyPassword
    ssl.truststore.location=/home/SIUser/SIKafkaServerSSLTruststore.jks
    ssl.truststore.password=YourPassword
    ssl.client.auth=required
    security.inter.broker.protocol=SSL
    ```

    Where *<IP>* is the IP address where Kafka is running and *<Port>* can be any open port number, such as 2182.

12. Copy the `SIKafkaServerSSLKeystore.jks` and `SIKafkaServerSSLTruststore.jks` files to the `/home/streamsadmin/security` directory

    Note: Ensure that the streamsadmin user has access to this file.

Enabling SSL for Kafka

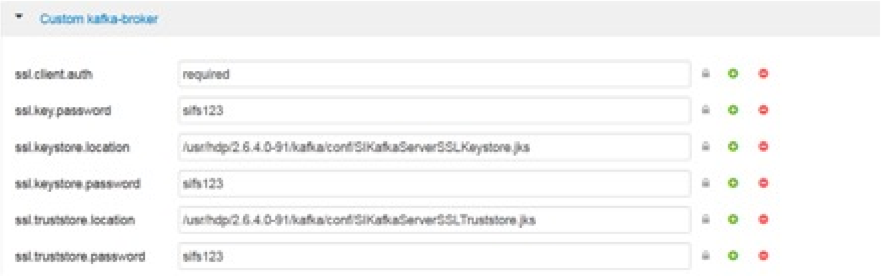
You create the SSL keys for the Kafka server and clients from the Ambari console.

Procedure

1. Log in to the Ambari console.

   `https://ambari.server:8081`

   The default credentials are `admin/admin`.
2. Click **Kafka**, and then click the **Configs** tab.
3. Expand **Custom kafka-broker**, and click **Add Property**.
4. Add the properties as follows:

ssl.client.auth	required
ssl.key.password	sifs123
ssl.keystore.location	/usr/hdp/2.6.4.0-91/kafka/conf/SIKafkaServerSSLKeystore.jks
ssl.keystore.password	sifs123
ssl.truststore.location	/usr/hdp/2.6.4.0-91/kafka/conf/SIKafkaServerSSLTruststore.jks
ssl.truststore.password	sifs123

   Figure 1: Kafka properties
5. Do not restart the Kafka services yet.

Enabling Kerberos

You must enable Kerberos for IBM Financial Transactions Repository.

You enable Kerberos on the Ambari server.

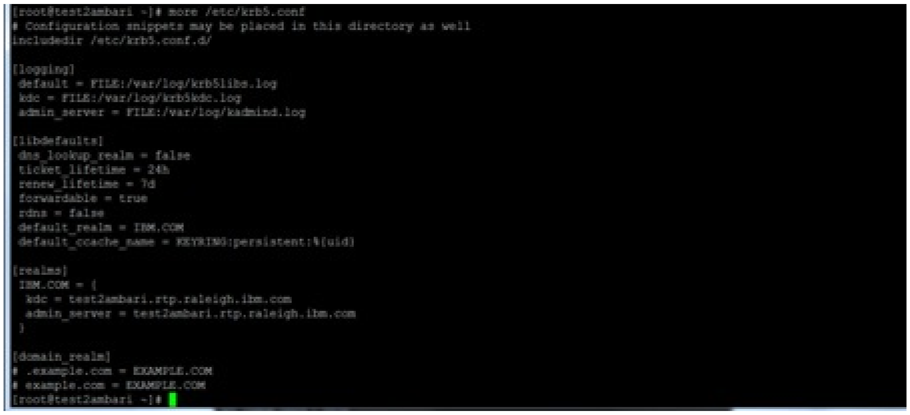
For more information about Kerberos, see the Kerberos documentation (http://web.mit.edu/kerberos/krb5-1.12/doc/index.html).

Procedure

1. Log in to the Ambari server.
2. Enter the following command to install the KDC.

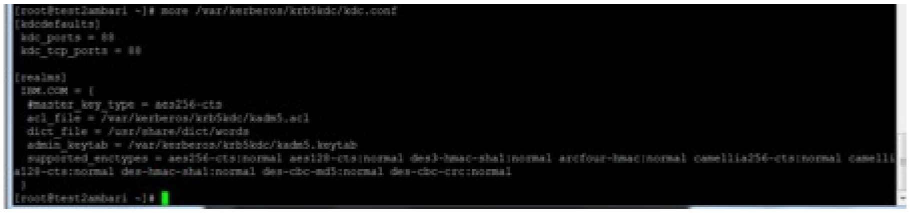
   ```
   yum install krb5-server krb5-libs krb5-workstation - -
   ```

3. Edit both the `/etc/krb5.conf` and `/var/kerberos/krb5kdc/kdc.conf` files for your environment.



```
[root@test2ambari ~]# more /etc/krb5.conf
# Configuration snippets may be placed in this directory as well
includedir /etc/krb5.conf.d/

[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 dns_lookup_realm = false
 ticket_lifetime = 24h
 renew_lifetime = 7d
 forwardable = true
 rdns = false
 default_realm = IBM.COM
 default_ccache_name = KEYRING:persistent:%{uid}

[realms]
 IBM.COM = {
  kdc = test2ambari.rtp.raleigh.ibm.com
  admin_server = test2ambari.rtp.raleigh.ibm.com
 }

[domain_realm]
# .example.com = EXAMPLE.COM
# example.com = EXAMPLE.COM
[root@test2ambari ~]#
```

Figure 2: krb5.conf file



```
[root@test2ambari ~]# more /var/kerberos/krb5kdc/kdc.conf
[kdcdefaults]
 kdc_ports = 88
 kdc_tcp_ports = 88

[realms]
 IBM.COM = {
  #master_key_type = aes256-cts
  acl_file = /var/kerberos/krb5kdc/kadm5.acl
  dict_file = /usr/share/dict/words
  admin_keytab = /var/kerberos/krb5kdc/kadm5.keytab
  supported_enctypes = aes256-cts:normal aes128-cts:normal des3-hmac-sha1:normal arcfour-hmac:normal camellia256-cts:normal camelli
a128-cts:normal des-hmac-sha1:normal des-cbc-md5:normal des-cbc-crc:normal
 }
[root@test2ambari ~]#
```

Figure 3: kdc.conf file

4. Enter the following command to create the KDC database:

   ```
   kdb5_util create -s
   ```

5. Enter the following commands to start the KDC and admin servers:

   ```
   systemctl enable krb5kdc
   ```

   ```
   systemctl enable kadmin
   ```

6. Use the following command to add administrators to the Kerberos database:

   ```
   kadmin.local -q "addprinc admin/admin"
   ```

7. Add users to the `/var/kerberos/krb5kdc/kadm5.acl` file. For more information about using the access control list, see the Kerberos documentation (http://web.mit.edu/kerberos/krb5-1.12/doc/admin/conf_files/kadm5_acl.html).
8. Restart the processes:

   ```
   systemctl restart kadmin
   ```

   ```
   systemctl restart krb5kdc
   ```

9. Verify that the daemons started properly:

   a) Check the messages in the `/etc/krb5.conf` file.

   b) Run the following command:

   ```
   netstat -nltp | grep 88
   ```

   The output should be similar to the following:

   ```
   tcp     0     0 0.0.0.0:88     0.0.0.0:*     LISTEN     859185/krb5kdc
   ```

   c) Run the following command:

   ```
   netstat -nltp | grep 464
   ```

   The output should be similar to the following:

   ```
   tcp     0     0 0.0.0.0:464     0.0.0.0:*     LISTEN     859259/kadmind
   ```

   d) Run the following command:

   ```
   netstat -nltp | grep 749
   ```

   The output should be similar to the following:

   ```
   tcp     0     0 0.0.0.0:749     0.0.0.0:*     LISTEN     859259/kadmind
   ```

Enabling Kerberos in Ambari

You must enable Kerberos in the Ambari console.

Procedure

1. Log in to the Ambari console.

   `https://ambari.server:8081`
2. Click **Admin** > **Kerberos**.
3. Click **Enable Kerberos**.
4. On the **Get Started** page, select **Existing MIT KDC**, select all of the check boxes for **Existing MIT KDC**, and click **Next**.
5. Enter values for each of the required fields.

   The **KDC hosts** and **Kadmin host** values are the name of the Ambari server computer.

   The **Admin principal** is value you used in step 6 of "Enabling Kerberos" on page 5. For example, `admin/admin`.
6. Click **Next** to complete the steps in the wizard.

   If there are any errors in the restart services step, you can manually restart the services from the Ambari console. For troubleshooting any errors during these steps, you can check the logs in `/var/log/ambari-server/` and `/var/log/hadoop/hdfs/`.

Creating a user account to run jobs

You must create a user to execute all of the jobs and write the data into the encryption zone.

Procedure

1. Create a non-root user on all servers used in your installation. The user will run Spark. Name the user sifsuser.

For example, use the following commands to create the user:

```
sudo groupadd sifsuser
```

```
sudo useradd -g sifsuser
```

2. As the hdfs user, create a home directory on HDFS for the sifsuser

   For example, use the following command:

```
hdfs dfs -mkdir /user/sifsuser
```

3. Add a principal on the Ambari server and run the `kadmin.local` command as the root user.
4. Run the following commands:

```
addprinc -randkey sifsuser@domain.com
```

```
ktadd -norandkey -k /etc/security/keytabs/sifsuser.keytab sifsuser@domain.COM
```

5. Copy the `sifsuser.keytab` to all Yarn Node Manager nodes.
6. Set the ownership on the new keytab file by using the following commands:

```
chown sifsuser:hadoop /etc/security/keytabs/sifsuser.keytab
```

```
chmod a+r /etc/security/keytabs/sifsuser.keytab
```

7. Initialize a Kerberos ticket by logging in as the sifsuser user and running the following command:

```
kinit –kt /etc/security/keytabs/sifsuser.keytab sifsuser@domain.COM
```

Enabling Hadoop encryption

You must enable encryption for the Hadoop file system (HDFS).

Procedure

1. Set up the permissions in Ranger KMS.
   a) Log in to the Ranger KMS console:

      `https://hadoop.master:6182/index.html`

      The default credentials are `keyadmin/keyadmin`.
   b) In **Service Manager**, click **fcicluster_kms**, and click **Add New Policy**.
   c) In **Policy Name**, enter `sifspolicy`.
   d) In **Key Name**, select `sifshdfskey`.
   e) In the **Allow Conditions** table, enter the following:
      - In **Select User**, select `sifsuser`.
      - In **Permissions**, click **add**, select **Decrypt EEK**, and click the check mark to add the permission.
2. Create an encryption zone for the `sifsuser` directory.
   a) Log in to the Hadoop master node computer as the hdfs user.
   b) Run the following commands:

      If you have not created the `/user/sifsuser` directory, create one now:

```
hdfs dfs -mkdir /user/sifsuser
```

Then, run the following commands:

```
hdfs crypto -createZone -keyName sifshdfskey -path /user/sifsuser
```

```
hdfs dfs -chown sifsuser:hadoop /user/sifsuser
```

3. Create an encryption zone for Hadoop.
    a) Log in to the Hadoop master node computer as the hdfs user.
    b) Run the following commands:

    If you have not created the `/user/hadoop` directory, create one now:

    ```
    hdfs dfs -mkdir /user/hadoop
    ```

    Then, run the following commands:

    ```
    hdfs crypto -createZone -keyName sifshdfskey -path /user/hadoop
    ```

    ```
    hdfs dfs -chown sifsuser:hadoop /user/hadoop
    ```

    For troubleshooting any errors during these steps, you can check the logs in the following locations:

    - `/var/log/hadoop/hdfs/`
    - `/var/log/ambari-server/`
    - `/var/log/ambari-agent/`
    - `/var/lib/ambari-agent/data/`
4. Validate the data transfer from the encrypted zone.
    a) Log in as the sifsuser user.
    b) Enter the following commands:

    ```
    hdfs dfs –put testdata.txt /user/sifsuser/
    ```

    ```
    hdfs dfs -cat /user/sifsuser/testdata.txt
    ```

    This should show decrypted, clear text data.

    c) Enter the following commands:

    ```
    hdfs dfs -cat /.reserved/raw/user/sifsuser/testdata.txt
    ```

    This should show encrypted data.

Enabling wire encryption

You enable wire encryption from the Ambari console.

Procedure

1. Log in to the Ambari console.

    `https://ambari.server:8081`

    The default credentials are `admin/admin`.
2. Click **HDFS**, and then click the **Configs** tab.
3. Click **Advanced**.
4. Expand **Custom core-site**, and click **Add Property**.
    a) In the **Properties** box, enter the following:

    ```
    hadoop.rpc.protection=privacy
    ```

    b) Click **Add**.

5. Expand **Custom hdfs-site**, and click **Add Property**.

    a) In the **Properties** box, enter the following:

    ```
    dfs.encrypt.data.transfer=true
    ```

    b) Click **Add**.

6. Click **Save**.

Updating the Kerberos configuration for Kafka

Procedure

1. Log in to the Ambari console.

    `https://ambari.server:8081`

    The default credentials are `admin/admin`.
2. Click **Kafka**, and then click the **Configs** tab.
3. Expand **Advanced kafka-broker**.
4. In **security.inter.broker.protocol**, enter SASL_SSL
5. Expand **Kafka Broker**.
6. Change the **listeners** value to SASL_SSL`://`*`hadoop.gateway.domain`*`.com:`6667
7. Click **Save**.

Verifying Kafka

To ensure that your environment is ready to install IBM Financial Transactions Repository, you can verify your Kafka settings.

Procedure

1. Create file in the `/usr/hdp/2.6.4.0-91/kafka/conf` directory that is named `client.ssl.properties`.
2. Add the following contents to `client.ssl.properties`:

    ```
    security.proptcol=SASL_SSL
    ssl.truststore.location=/usr/hdp/2.6.4.0-91/kafka/conf/SIKafkaClientSSLTruststore.jks
    ssl.truststore.password=sifs123
    ssl.keystore.location=/usr/hdp/2.6.4.0-91/kafka/conf/SIKafkaClientSSLKeystore.jks
    ssl.keystore.password=sifs123
    ```

3. In a terminal window, enter the following command to set the JVM parameters:

    ```
    export KAFKA_OPTS="-Djava.security.auth.login.config=/usr/hdp/2.6.4.0-91/kafka/conf/
    kafka_client_jaas_sifs.conf
    ```

4. Go to the directory where Kafka is installed, and run the following commands to run the consumer and producer:

    ```
    ./kafka-console-producer.sh --broker-list hadoop.gateway.domain.com:6667 --topic
    sifs.voice.in --producer.config ../conf/client.ssl.properties --security-protocol SASL_SSL
    ```

    ```
    ./kafka-console-consumer.sh --bootstrap-server hadoop.gateway.domain.com:6667 --topic
    sifs.ecomm.in --new-consumer --consumer.config ../conf/client-ssl.properties --security-
    protocol SASL_SSL
    ```

Integrate WebSphere Liberty with Kafka on a Kerberized cluster

Procedure

1. Edit the `kafka.properties` that is used for WebSphere Liberty to include the following entries:

   ```
   sasl.kerberos.service.name=kafka
   security.protocol=SASL_SSL
   sasl.jaas.config=com.ibm.security.auth.module.Krb5LoginModule required
   useKeytab="/home/liberty-artifacts/hadoop_HDP/kafka.service.keytab"
   credsType="both"
   principal="kafka/hadoop.gateway.domain.com";
   ```

2. Copy the Kafka keytab file to the Liberty instance, and place it in the `useKeytab` location that you used in the `kafka.properties` file.

Integrate Streams with Kafka on a Kerberized cluster

Procedure

1. Create file that is named `sifs-jaas.conf`.
2. Add the following contents to `sifs-jaas.conf`:

   ```
   KafkaClient {
   com.ibm.security.auth.module.Krb5LoginModule required
   useKeytab="/home/streamsadmin/HDPHdfs/kafka.service.keytab"
   credsType="both"
   principal="kafka/hadoop.gateway.domain.com";
   };
   ```

3. Copy the Kafka keytab file to the Liberty instance, and place it in the `useKeytab` location that you used in the `sifs-jaas.conf` file.

Install IBM Financial Transactions Repository

Ensure that you have enabled security for your Hortonworks Data Platform (HDP) environment.

Procedure

1. Update the HDFS configuration.

   a) Log in to the Ambari console.

   `https://ambari.server:8081`

   b) Click **HDFS**, click the **Configs** tab, and then click the **Advanced** tab.

   c) Expand **Custom core-site**, and click **Add Property**.

   d) In the **Properties** box, enter the following:

   ```
   hadoop.proxyuser.sifsuser.hosts=*
   hadoop.proxyuser.sifsuser.groups=*
   ```

   e) Click **Add**, and then click **Save**.

   f) Click **Hive**, click the **Configs** tab, and then click the **Advanced** tab.

   g) Expand **Custom hive-site**, and click **Add Property**.

   h) In the **Properties** box, enter the following:

   ```
   hive.users.in.admin.role=hive,sifsuser
   hadoop.proxyuser.sifsuser.hosts =*
   hadoop.proxyuser.sifsuser.groups=*
   hive.server2.enable.doAs=true
   ```

   i) Click **Add**, and then click **Save**.

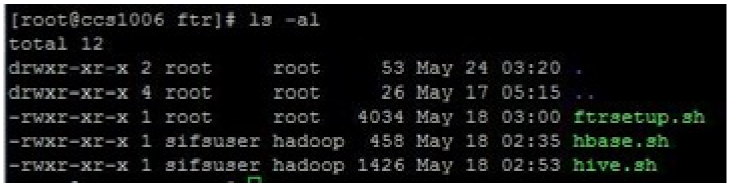
   j) Copy the `hive-site.xml` to the *SPARK_HOME*`/conf` directory on the Spark driver and worker node computers.

2. Run the following commands as the hbase user on the HBase master server node computer:

   ```
   Hbase shell
   ```

   ```
   grant ‘RWCA’, ‘sifsuser’
   ```

3. Copy the following JAR files to the `/opt/ibm/ftr/lib` directory on all HDP nodes.
   - `json-1.8.jar`
   - `kafka-clients-0.10.1.2.6.4.0-91.jar`
4. Go to the `/opt/ibm/ftr` directory.
5. Ensure that the scripts in this directory have the following permissions:

   ```
   [root@ccs1006 ftr]# ls -al
   total 12
   drwxr-xr-x 2 root     root     53 May 24 03:20 .
   drwxr-xr-x 4 root     root     26 May 17 05:15 ..
   -rwxr-xr-x 1 root     root   4034 May 18 03:00 ftrsetup.sh
   -rwxr-xr-x 1 sifsuser hadoop  458 May 18 02:35 hbase.sh
   -rwxr-xr-x 1 sifsuser hadoop 1426 May 18 02:53 hive.sh
   ```

   Figure 4: Script permissions

6. Run the following command:

   ```
   ./ftrsetup.sh
   ```

   The artifacts are downloaded from Artifactory and extracted. The JAR files are copied to the relevant directories, and the Hbase and Hive tables are created.

7. Run the following commands to set up DAL and configure the Hive and HBase tables. Run the commands as the root user.

   ```
   mkdir /home/sifsuser/ConfFiles
   ```

   ```
   cp /usr/hdp/2.6.4.0-91/kafka/conf/kafka.client.keystore.jks /home/sifsuser/ConfFiles/
   ```

   ```
   cp /usr/hdp/2.6.4.0-91/kafka/conf/kafka.client.truststore.jks /home/sifsuser/ConfFiles/
   ```

   ```
   cp /usr/hdp/2.6.4.0-91/kafka/conf/kafka_jaas.conf /home/sifsuser/ConfFiles/
   ```

   ```
   chown -R sifsuser:hadoop /home/sifsuser/ConfFiles
   ```

8. Configure the `ExtractConfig.jar` file.

   a) Create a temporary directory:

   ```
   mkdir temp
   ```

   b) Extract `ExtractConfig.jar` to the `temp` directory:

   ```
   unzip ExtractConfig.jar -d ./temp
   ```

   c) In the extracted files, open the `resource/config.properties` file to set the REST API URL. Change the *<Kube_manager_IP>* value to match your environment:

   ```
   API_URL= https://<Kube_manager_IP>:3001/config
   ```

   d) Compress the file to create new `ExtractConfig.jar` file:

   ```
   zip -r ExtractConfig.jar
   ```

9. Create Hive external tables by using beeline:

```
su - sifsuser
```

```
>beeline
```

```
beeline>!connect <JDBC hiveserver2 address> "" "";
```

```
beeline>create database if not exists ccsdb;
```

```
beeline>CREATE EXTERNAL TABLE ccsdb.tblCCSAudit(key string, streamId string,
activityTimestamp string, Component string, description string, fileIndicator string,
fileName string, system string,fileSize String,notification String,numberOfParsingErrors
String,numberOfRecords String,originalFileName String,processingStage String,status
String,tradingDate String) STORED BY 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES ("hbase.columns.mapping" =
"auditData:streamId,auditData:activityTimestamp,auditData:component,auditData:description,au
ditData:fileIndicator,auditData:fileName,
auditData:system,auditData:fileSize,auditData:notification,auditData:numberOfParsingErrors,a
uditData:numberOfRecords,auditData:originalFileName,auditData:processingStage,auditData:stat
us,auditData:tradingDate") TBLPROPERTIES("hbase.table.name" = "tblCCSAudit");
```

```
beeline>CREATE EXTERNAL TABLE ccsdb.tblFileVersion(key string, StreamId string,
Ingestion_Timestamp string, Version string, fileCreatedDate string, fileHandler string,
fileName string, system string) STORED BY
'org.apache.hadoop.hive.hbase.HBaseStorageHandler' WITH SERDEPROPERTIES
("hbase.columns.mapping" =
"metaData:streamId,metaData:activityTimestamp,metaData:version,metaData:tradingDate,metaData
:fileHandler,metaData:fileName, metaData:system") TBLPROPERTIES("hbase.table.name" =
"tblFileVersion");
```

```
beeline>create external table ccsdb.fr_data_spark(rowkey string, msgType string, streamId
string ,ClOrdID string, Symbol string, TransactTime string, OrdType string, OrderQty
int,Price double, Side string, PartyID string) stored by
'org.apache.hadoop.hive.hbase.HBaseStorageHandler'with serdeproperties
("hbase.columns.mapping"=":key,messageHeader:MsgType,metaData:streamId,
messageBody:ClOrdID, messageBody:Symbol,messageBody:TransactTime, messageBody:OrdType,
messageBody:OrderQty, messageBody:Price, messageBody:Side, messageBody:PartyID")
TBLPROPERTIES("hbase.table.name" = "tblHRData");
```

10. Edit the `/opt/ibm/ftr/config/configuration.properties` file to include the environment details:

Table 2: configuration.properties values

Property	**Location or value**
SERVER_ADDRESS	Ambari > Services > HDFS > Config (search for the following property and copy the value) > fs.defaultFS
HADOOP_USER_NAME	Hadoop
HIVE_ADDRESS	Ambari > Services > Hive > HiveServer2 JDBC URL
HDFS_SERVER	Ambari > Services > HDFS > Config (search for the following property and copy the value) > fs.defaultFS
HBASE_ZOOKEEPER_QUORUM_VALUE	Ambari > Services > HBASE > Config (search for the following property and copy the value) > hbase.zookeeper.quorum

Table 2: configuration.properties values (continued)

Property	**Location or value**
HBASE_ZOOKEEPER_CLIENT_VALUE (Port)	Ambari > Services > HBASE > Config (search for the following property and copy the value) > hbase.zookeeper.property.clientPort
BOOTSTRAP_SERVERS_CONFIG_PRODUCER	Ambari > Services > Kafka > Config (search for the following property and copy the value) > *<Kafka Host name : Port Id>*. The port number is 6667 for an HDP environment.
BOOTSTRAP_SERVERS_CONFIG_CONSUMER	Ambari > Services > Kafka > Config (search for the following property and copy the value) > *<Kafka Host name : Port Id>*. The port number is 6667 for an HDP environment.
SSL_TRUSTSTORE_LOCATION_CONFIG	*<Include SSL truststore .jks file path here>*
SSL_TRUSTSTORE_PASSWORD_CONFIG	*<Trustore password>*
SSL_KEYSTORE_LOCATION_CONFIG	*<Include SSL keystore .jks file path here>*
SSL_KEYSTORE_PASSWORD_CONFIG	*<Keystore password>*
SSL_KEY_PASSWORD_CONFIG	*<Key password>*
SASL_KERBEROS_SERVICE_NAME	kafka

SASL_JAAS_CONFIG	com.sun.security.auth.module.Krb5LoginModule required useKeyTab=true
	keyTab="*<include kafka service keytab>* "
	principal="*<include principal for kafka service keytab>*"
	useTicketCache=true
	renewTicket=true
	serviceName="kafka";

SPARK_HOME	*<Include spark2 path here>*. For example, for an HDP environment: `usr/hdp/2.6.4.0-91/spark2`
KEYTAB	*<include kafka service keytab>*
PRINCIPAL	*<include principal for kafka service keytab>*
JAAS_CONFIG	-Djava.security.auth.login.config=*<include path for kafka JAAS file here>*
CONFIG_API	https://*<kubernetes master hostname>*:3001/SIFSServices/ftr/config/

11. Change to the sifuser user.

```
su sifsuser
```

12. Run the following commands:

    ```
    nohup java -jar /opt/ibm/ftr/jar/Kafka-2.0.3-SNAPSHOT-jar-with-dependencies.jar > /home/
    sifsuser/dataingestion_log.$(date --iso).out &
    ```

    ```
    nohup park-submit --master local --deploy-mode client --driver-memory 10G --executor-memory
    1G --num-executors 5 --executor-cores 3 --conf spark.driver.extraClassPath="/usr/hdp/
    2.6.4.0-91/hive/lib/*:/usr/hdp/2.6.4.0-91/hbase/lib/*" --conf
    "spark.driver.extraJavaOptions=-Djava.security.auth.login.config=/home/sifsuser/ConfFiles/
    sifs-jaas.conf" --files /usr/hdp/2.6.4.0-91/hbase/conf/hbase-site.xml --keytab /etc/
    security/keytabs/sifsuser.keytab --principal sifsuser@IBM.COM --class
    com.ibm.ccs.kafka.notification.SendNotification /opt/ibm/ftr/jar/NotificationComp-2.0.3-
    SNAPSHOT.jar &
    ```

Starting the HBase REST Server

The HBase REST server must be manually started as a background service.

Procedure

Use the following command to start the HBase REST server:

```
<hbase installation path>/bin/hbase-daemon.sh start rest -p <port> --infoport  <infoPort>
```

where:

- *<port>* is the service port number
- *<infoPort>* is the port for the web UI with information about the service

For example,

```
/usr/hdp/2.6.4.0-91/hbase/bin/hbase-daemon.sh.start rest -p 17001 --infoport  17000
```

The port value must match the HBase Rest server port that is configured in the FTR-API’s environment file for the HBASEADDRESS key.

Configure the HBase REST API for SSL

You must enable SSL for the HBase REST API.

Procedure

1. Add the following properties to `hbase-site.xml` by using the Ambari console:

   ```
   hbase.rest.kerberos.principal=hbase/_HOST@<Real name>
   hbase.rest.keytab.file=/etc/security/keytabs/hbase.service.keytab
   hbase.rest.port=17001
   ```

2. Run the following command in the HBase Master server to create a keystore for HBase:

   ```
   keytool -genkey -alias hbase -keyalg RSA -keysize 1024 -keystore hbase.jks
   ```

3. Export the certificate:

   ```
   su -l hbase -c "keytool -exportcert -alias hbase -file certificate.cert -keystore hbase.jks"
   ```

4. Add the following properties to the `hbase-site.xml` configuration file on each node in your HBase cluster by using the Ambari console:

   ```
   hbase.rest.ssl.enabled=true
   hbase.rest.ssl.keystore.store=/path/to/keystore
   hbase.rest.ssl.keystore.password=keystore password
   hbase.rest.ssl.keystore.keypassword=key password
   ```

5. Restart all of the HBase nodes in the cluster by using Ambari.
6. Restart the HBase REST server.

Deploying the Docker images through Helm

Before you begin

- The IBM Financial Crimes Insight platform must be installed and configured.
- The IBM Surveillance Insight for Financial Services is installed and configured.
- The Hortonworks Data Platform (HDP) is set up.
- The Kubernetes cluster is set up.

Procedure

1. Tag the 3 images (API node, backend, and Streams) with a 2.0.3 tag by running the following commands:

   ```
   docker tag <ftr image name>:1.0.3  <ftr image name>:2.0.3
   ```

   You must change the name of one image, `ftr-cat-node-api`, to `ftr-cat-node-ftr` by using the following command:

   ```
   docker tag fcidev-si-docker-registry.fss.ibm.com:5000/ibmcom/fci-cat-node-api:1.0.3 fcidev-
   si-docker-registry.fss.ibm.com:5000/ibmcom/fci-cat-node-ftr:2.0.3
   ```

2. Use the following commands to purge the `cats-policy` release:

   ```
   helm ls
   ```

   If the cats proxy charts are shown, delete them using the following command:

   ```
   helm del –purge cats-proxy
   ```

3. Run the following command to deploy Helm:

   ```
   helm install --set "managerIPAddresses={10.65.6.40}" \
   --set "forwards.3001.serviceReleaseName=cats" \
   --set "forwards.3001.serviceName=ftr-nodejs" \
   --set "forwards.3001.servicePort=3001" \
   --set "forwards.3003.serviceReleaseName=cats" \
   --set "forwards.3003.serviceName=ftr-apinodejs" \
   --set "forwards.3003.servicePort=3003" \
   /fcimedia/ftr/archives/fci-charts-1.0.3/charts/nginx-ingress-controller-1.0.3.tgz
   ```

   If the Kubernetes system variables are not defined in `values.yaml` file, you can define them by using the following commands:

   ```
   set global.managerFQDN=$(hostname -f) --set global.nfsServer=$(hostname -
   f),global.dockerRepository=" "
   ```

   This Helm install command would deploy both the Node images. However, for Streams, the persistent volume must be initialized.

4. Start the Node application:

   a) Run the following Kubernetes commands:

   ```
   kubectl get pods
   ```

   ```
   kubectl exec <pod ID> bash
   ```

b) Open the `.env` file by using the following command:

```
vi .env
```

c) Change the API node image address:

```
https://localhost:<port of api node image>
```

d) Restart the application:

```
pm2 restart app
```

e) Open `apinodejs` image pod and change the value of HBASEADDRESS variable in `/opt/codebase/.env` file to the current HBase server.

f) Restart the index:

```
pm2 restart index
```

5. Initialize the volume for Streams:

a) Go to the following path on the Kubernetes manager: `/fcimedia/ftr/cats-install-kit-2.0.3/helm`

b) Run the following command:

```
initialize-pv -p $(kubectl get pod -l app=<ftr-streams>,release=cats -o
jsonpath='{.items[*].metadata.name}') -i init-pv -t <location of streams data tar>
```

This step deploys the Streams image.

c) Copy the Hadoop configuration directories (`hadoop` and `hadoop-hdfs`) from the Hadoop installation to the `config` directory in the volume.

d) In the volume point, an `ingest` directory exists. Use this directory to copy the new trade data files for ingestion from the various streams.

6. Start the Streams application:

a) Identify all of the containers that are deployed in this cluster:

```
kubectl get pods
```

b) Get the container name of the FTR Streams container from the output.

c) Run the following command to access the container:

```
kubectl exec -it <pod name> bash
```

d) Update the following configuration:

1) In the `data/sifs-jaas.conf` file, update the Kafka Kerberos principal:

```
principal="kafka/hdp1secondary.fss.ibm.com";
```

2) In `data/producer.properties`, update the Kafka server details:

```
bootstrap.servers==hdp1secondary.fss.ibm.com:6667
```

3) In `data/application.properties`, update the HDFS and Configuration Server locations:

```
hdfs_url=hdfs://hdp1master.fss.ibm.com:8020/
config_server=https://fcidev-si-kmaster.fss.ibm.com:3001/SIFSServices/ftr
```

e) Go to the application directory and start the streams application:

```
cd /opt/codebase
```

```
./app.sh
```

f) Log in as the streamsadmin user, and verify that the IBM FTR Streams jobs are running in the container:

```
streamtool lsjobs -d StreamsDomain -i SIInstance Instance: SIInstance
```

The results appear as:

```
Id State   Healthy User                   Date
Name                                                                    Group
77 Running yes     streamsadmin 2018-07-05T13:46:08+0000
application::CCSIngestFileRenaming_v1_77 default
78 Running yes     streamsadmin 2018-07-05T13:46:27+0000
application::CCSIngestFileRenaming_v1_78 default
```

Use SLM tags to track licensing

Software License Metric (SLM) tag files provide metrics for all applicable licensing options for a product.

The license metrics and the values are exposed in the resource utilization metric (RUM) tagging for data collection and reporting into ILMT. The metrics are based on the licensing types. For example, if your license is based on the number of concurrent users, then that is a metric that the product self-reports.

The SLM files allow the customer to answer the basic question: how much of the license do I need to have?

A product “logs” the consumption of particular license metrics at specified intervals. And the metric data can be fetched from the endpoint by the ILMT/IEM agent.

IBM Financial Transactions Repository defines the APP_NAME and APP_PERSISTENT_ID as constants in the entire SLM tag, and they are set as global static variables.

Function: HbaseRecordCount()

This function connects to the HBase service on the server and checks whether a connection exists. The function then returns the count of the records in the Hbase table. A time frame can be added to filter the records based on the number of days. You can do this by including the time stamps of the start and end time in scan.setTimeRange (start,end). The tablename, the column family, and the column that we want to refer must be defined. A constant field is used on which to base the number of counted records.

Function: generateSLM()

This function generates the SLM tag by providing the relevant parameters to the IBM logger package. The parameters include the instance path, which points to the path where the software is installed, and the location where the log files are maintained. Then, the metric under consideration (the count of records in the table) is provided with the start and end time stamps. These values point out to the validity of this entry in the tag. The header in an SLM tag remains the same and the changed metric entries are added to the tag, so the metrics are updated as needed. The time stamps should not overlap for different metric entries.

Function: main()

This function takes the input from the user about which table to refer to for the data in the Hbase service. Also, the instance path where the logs are generated is defined. To get data from a different table, the filename variable must be changed here.

Notices

This information was developed for products and services offered worldwide.

This material may be available from IBM® in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. This document may describe products, services, or features that are not included in the Program or license entitlement that you have purchased.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Software Group
Attention: Licensing
3755 Riverside Dr.
Ottawa, ON
K1V 1B7
Canada

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

IBM Surveillance Insight® for Financial Services includes Brat (v 1.3) from the following source and licensed under the following agreement:

- http://weaver.nlplab.org/~brat/releases/brat-v1.3_Crunchy_Frog.tar.gz
- https://creativecommons.org/licenses/by-sa/3.0/legalcode

IBM Surveillance Insight for Financial Services includes spaCy Models (v 1.2.0) from the following source and licensed under the following agreement:

- https://github.com/explosion/spacy-models (en_core_web_sm 1.2.0)
- https://creativecommons.org/licenses/by-sa/3.0/legalcode

Trademarks

IBM, the IBM logo and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at " Copyright and trademark information " at www.ibm.com/legal/copytrade.shtml.

IBM®