CA APIM Gateway - JDBC over SSL/TLS

This article will cover the setup and configuration of using SSL/TLS with the out of the box JDBC assertion available in the CA APIM Gateway 8.x and later.

Before you begin this how to..

This article is for CA APIM systems administrators who wish to enable JDBC connections to a MySQL Server using SSL/TLS v1.2 CIPHERS. This document covers the setup and configuration of a Java Keystore, Truststore and the required certificates. This document does not cover the setup of the MySQL server SSL support. Please see the MySQL Master - Master Replication over SSL/TLS documentation.

Before you begin this how to you should read the useful links at the bottom of the page and have a good understanding of MySQL, openssl and basic linux shell operations.

All of the operations below are performed within the 'privileged' secure shell also known as 'root'

Pay special attention to all commands. They will start with the hostname of which host they should be executed on. Example below is viewing a file on Gateway-01 using the cat command

```
[root@gateway-01 ~]# cat /etc/hosts
```

Server Configuration Details	root CA	Gateway-01	Database-01 (SQL Host)
Hostname	cn.gateway-01.ps.ca.com	gateway-01.ps.ca.com	database-01.ps.ca.com
IP Address	10.7.49.150	10.7.49.150	10.7.48.153
FQDN - CN	cn.gateway-01.ps.ca.com	gateway-01.ps.ca.com	database-01.ps.ca.com
Certificate Files Local Directory		/opt/SecureSpan/mysql-ssl	/opt/SecureSpan/mysql-ssl
Server Private Key		gateway-01.key.pem	database-01.key.pem
Server Public (Client) Cert		gateway-01.cert.pem	database-01.cert.pem

Verify the '/etc/hosts' on Gateway-01 and Database-01

a. Validate that both gateway's are properly configured within your '/etc/hosts' files on both Gateway-01 and Database-01

```
[root@gateway-01 ~]# cat /etc/hosts
[root@database-01 ~]# cat /etc/hosts
```

Your file should look similar to the example below but with your correct hostnames and IP address. Without a correct '/etc/hosts' file you will likely see errors later on during the setup

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4

::1         localhost localhost.localdomain localhost6 localhost6.localdomain6

10.7.49.150     gateway-01.ps.ca.com

10.7.48.153     database-01.ps.ca.com
```

SSL Keystore / CA / root CA Creation / Certificates Steps

1. Create CA key and certificate

- **REQUIRED:** Specify unique FQDN for the local root CA when prompted. In my example below I used 'cn.gateway-01.ps.ca.com'.

```
[root@gateway-01 ~]# mkdir -p /opt/SecureSpan/mysql-ssl

[root@gateway-01 ~]# cd /opt/SecureSpan/mysql-ssl

[root@gateway-01 ~]# openssl genrsa 2048 > /opt/SecureSpan/mysql-ssl/ca-key.pem

[root@gateway-01 ~]# openssl req -new -x509 -nodes -days 1000 -key ca-key.pem >
/opt/SecureSpan/mysql-ssl/ca-cert.pem
```

2. ## Create Server 1 Key and Certificate

a. Create Private Key for Gateway-01

```
[root@gateway-01 ~]# openssl req -newkey rsa:2048 -days 1000 -nodes -keyout servera-key.pem >
/opt/SecureSpan/mysql-ssl/servera-req.pem
```

b. Create Public Cert for Gateway-01

```
[root@gateway-01 ~]# openssl x509 -req -in servera-req.pem -days 1000 -CA ca-cert.pem -CAkey ca-key.pem
-set_serial 01 > /opt/SecureSpan/mysql-ssl/servera-cert.pem
```

3. ## Create Server 2 Key and Certificates

a. Create Private Key for Database-01

```
[root@gateway-01 ~]# openssl req -newkey rsa:2048 -days 1000 -nodes -keyout serverb-key.pem >
/opt/SecureSpan/mysql-ssl/serverb-req.pem
```

b. Create Public Cert for Database-01

```
[root@gateway-01 ~]# openssl x509 -req -in serverb-req.pem -days 1000 -CA ca-cert.pem -CAkey ca-key.pem set_serial 01 > /opt/SecureSpan/mysql-ssl/serverb-cert.pem
```

c. Copy Certificates and Keys to Gateway-01 and Database-01. Fix the perms on the files to allow mysql to access them

```
[root@gateway-01 ]# cd /opt/SecureSpan/mysql-ssl

[root@gateway-01 ]# scp ca-*.pem server*.pem ssgconfig@10.7.48.153:/home/ssgconfig
```

d. Fix the certificate file perms on Gateway-01.

```
[root@gateway-01 ]# cd /opt/SecureSpan/
[root@gateway-01 ]# chown -R mysql:mysql /opt/SecureSpan/mysql-ssl
```

e. Fix the certificate file perms on Database-01. Login to the host and become the root user through the privileged shell. Once you have logged in execute the commands below

```
[root@database-01 ]# cd /opt/SecureSpan/
[root@database-01 ]# cp /home/ssgconfig/*.pem .
[root@database-01 ]# chown -R mysql:mysql /opt/SecureSpan/mysql-ssl
```

4. Update the Java KeyStore on Gateway-01

a. Combine the Public / Private Cert for Gateway-01 as a PKCS12 file

```
[root@gateway-01 ]# /opt/SecureSpan/JDK/jre/bin/keytool -importkeystore -deststorepass keystore -destkeystore keystore -srckeystore servera.p12 -srcstoretype PKCS12 -srcstorepass keystore -alias gateway-01.ps.ca.com
```

```
[root@gateway-01 ]# openssl pkcs12 -export -in servera-cert.pem -inkey servera-key.pem -out servera.p12 -name gateway-01.ps.ca.com -CAfile ca-cert.pem -caname root
```

b. Combine the Public / Private Cert for Database-01 as a PKCS12 file

```
[root@gateway-01 ]# openssl pkcs12 -export -in serverb-cert.pem -inkey serverb-key.pem -out serverb.p12 -name database-01.ps.ca.com -CAfile ca-cert.pem -caname root
```

Once you have combined both gateways cert and key files you will need to copy the files to the 'Database-01' host. Using the same procedure as before we will copy them with 'scp'

```
[root@gateway-01 ]# scp *.p12 ssgconfig@10.7.48.153:/home/ssgconfig
```

Login to the 'Database-01' host as privileged shell user and copy the files to the correct location.

```
[root@database-01 ]# cp /home/ssgconfig/*.p12 /opt/SecureSpan/mysql-ssl
```

c. Import the Public / Private Cert for Gateway-01

```
[root@gateway-01 ]# /opt/SecureSpan/JDK/jre/bin/keytool -importkeystore -deststorepass keystore -destkeystore keystore -srckeystore servera.p12 -srcstoretype PKCS12 -srcstorepass keystore -alias gateway-01.ps.ca.com
```

d. Import the Public / Private Cert for Database-01

```
[root@gateway-01 ]# /opt/SecureSpan/JDK/jre/bin/keytool -importkeystore -deststorepass keystore -destkeystore keystore -srckeystore serverb.p12 -srcstoretype PKCS12 -srcstorepass keystore -alias database-01.ps.ca.com
```

5. Create Java TrustStore on Gateway-01

a. Create the Java TrustStore File

```
[root@gateway-01] # /opt/SecureSpan/JDK/jre/bin/keytool -import file
/opt/SecureSpan/mysql-ssl/cascerts/ca-certs.pem -alias cn.gateway-01.ps.ca.com -keystore TrustStore
```

b. Import the Public Cert for root CA

```
[root@gateway-01] # /opt/SecureSpan/JDK/jre/bin/keytool -import file /opt/SecureSpan/mysql-ssl/cacerts.pem -alias
cn.gateway-01.ps.ca.com -keystore TrustStore
```

c. Import the Public Cert for Gateway-01

```
[root@gateway-01] # /opt/SecureSpan/JDK/jre/bin/keytool -import file /opt/SecureSpan/mysql-ssl/servera-cert.pem
-alias gateway-01.ps.ca.com -keystore TrustStore
```

d. Import the Public Cert for Database-01

```
[root@gateway-01] # /opt/SecureSpan/JDK/jre/bin/keytool -import file /opt/SecureSpan/mysql-ssl/serverb-cert.pem
-alias database-01.ps.ca.com -keystore TrustStore
```

6. ## Create Java TrustStore on Database-01

a. Create the Java TrustStore File

```
[root@database-01] # /opt/SecureSpan/JDK/jre/bin/keytool -import file
/opt/SecureSpan/mysql-ssl/cascerts/ca-certs.pem -alias cn.gateway-01.ps.ca.com -keystore TrustStore
```

b. Import the Public Cert for root CA

```
[root@database-01] # /opt/SecureSpan/JDK/jre/bin/keytool -import file /opt/SecureSpan/mysql-ssl/cacerts.pem -alias
cn.gateway-01.ps.ca.com -keystore TrustStore
```

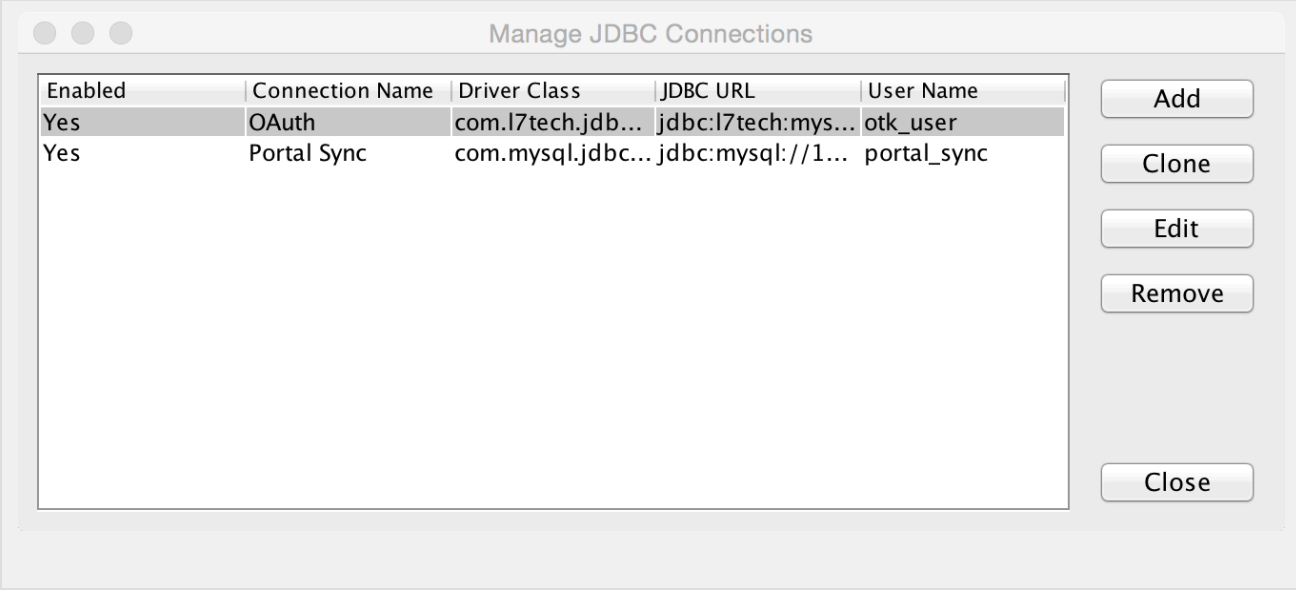
c. Import the Public Cert for Gateway-01

```
[root@database-01] # /opt/SecureSpan/JDK/jre/bin/keytool -import file /opt/SecureSpan/mysql-ssl/servera-cert.pem
-alias gateway-01.ps.ca.com -keystore TrustStore
```

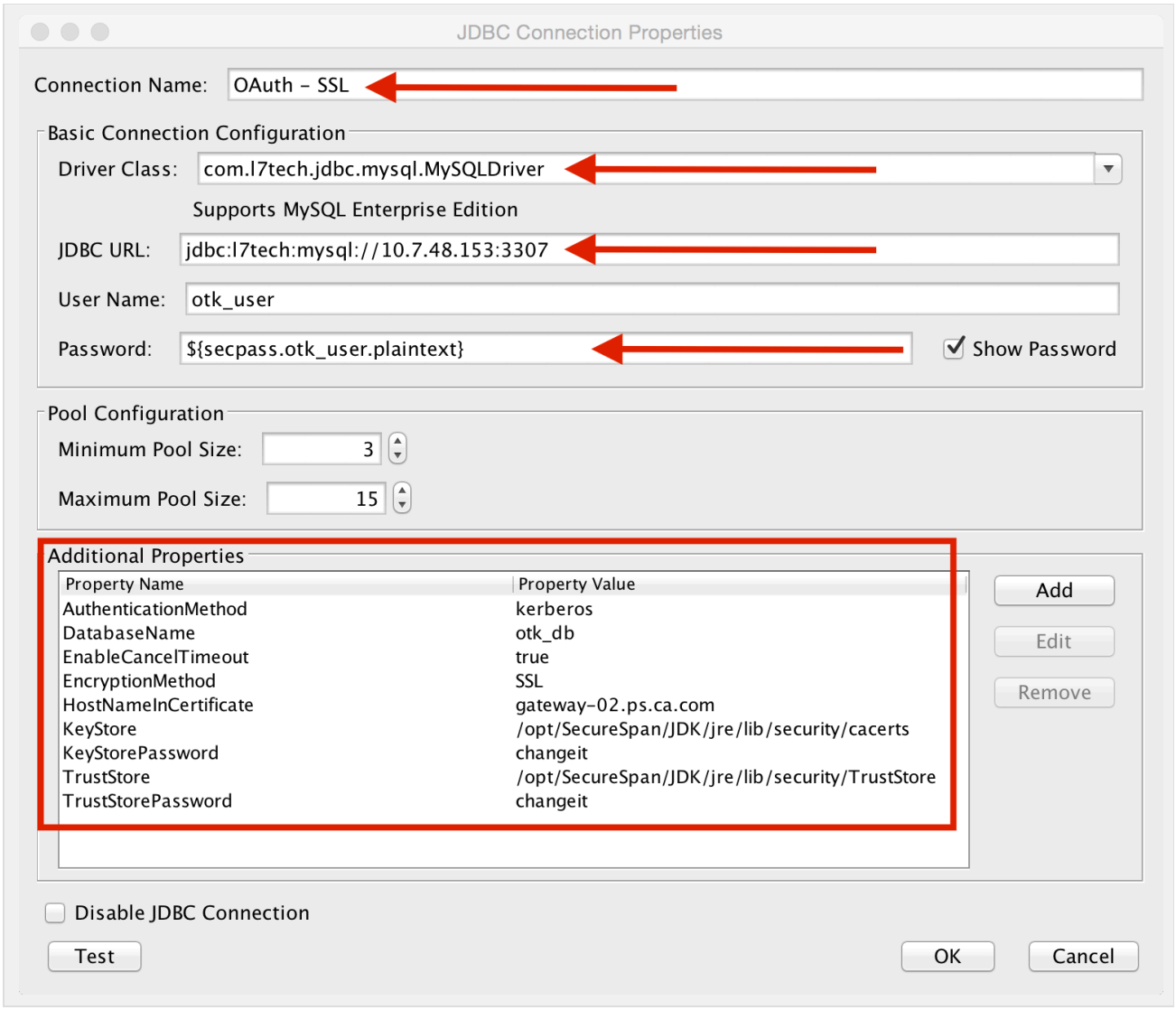
d. Import the Public Cert for Database-01

```
[root@database-01] # /opt/SecureSpan/JDK/jre/bin/keytool -import file /opt/SecureSpan/mysql-ssl/serverb-cert.pem
-alias database-01.ps.ca.com -keystore TrustStore
```

7. ## Clone your OTK database connection



8. **Modify the JDBC URL to support SSL**

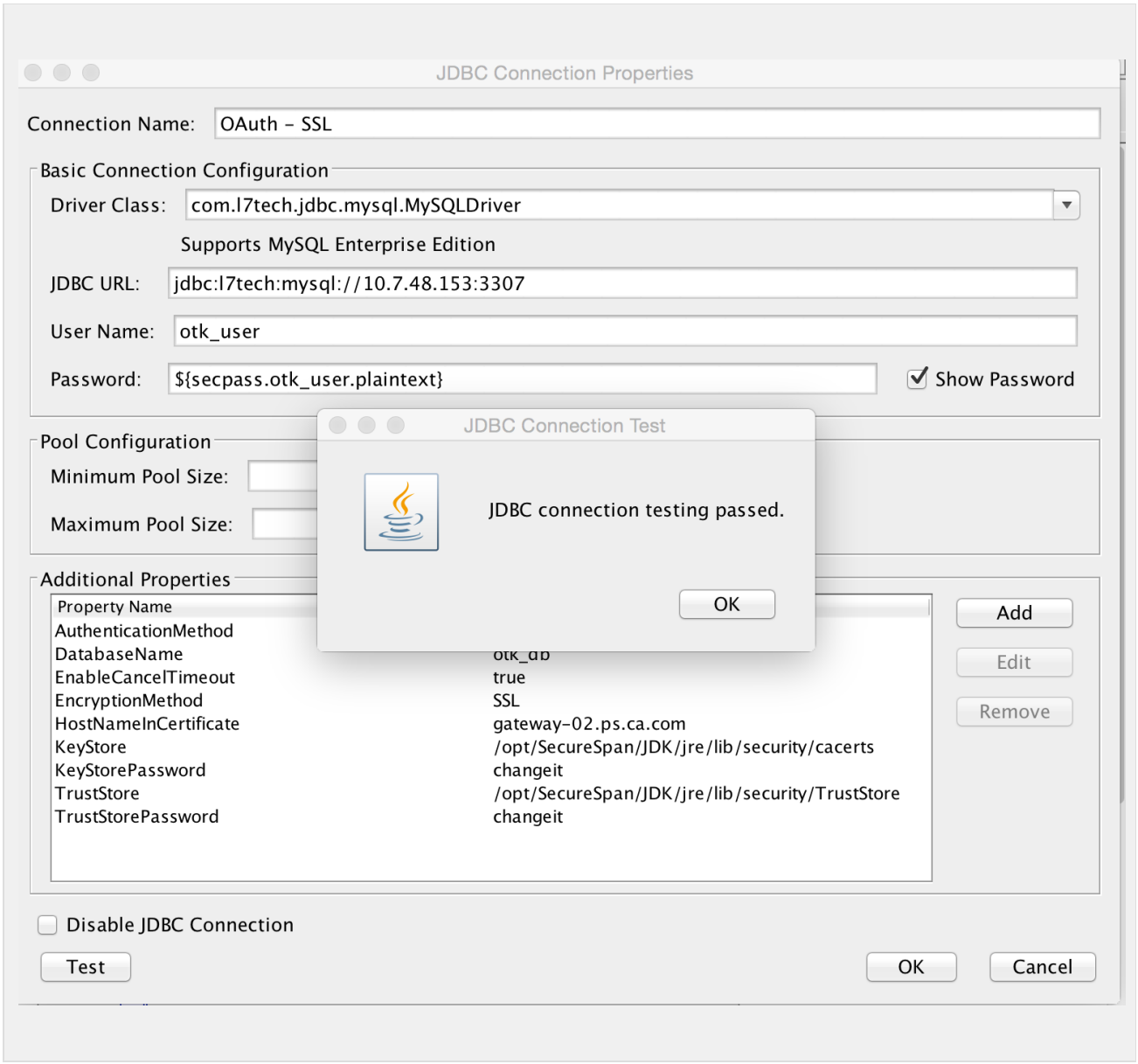


Connection Properties			
Connection Name:		OAuth SSL	
Driver Class		com.l7tech.jdbc.mysql.MySQL Driver	Select the correct driver from the drop down list to support MySQL Enterprise Edition. If you are using a different server select the appropriate driver. MySQL Community Edition does not support this feature.
JDBC URL		jdbc:l7tech:mysql://10.7.48.153:3307;	Note the addition of 'l7tech' after the jdbc section and the changes to port 3307. You can connect with SSL on either port 3306 or 3307.
User Name		otk_user	Username for the database you wish to connect to

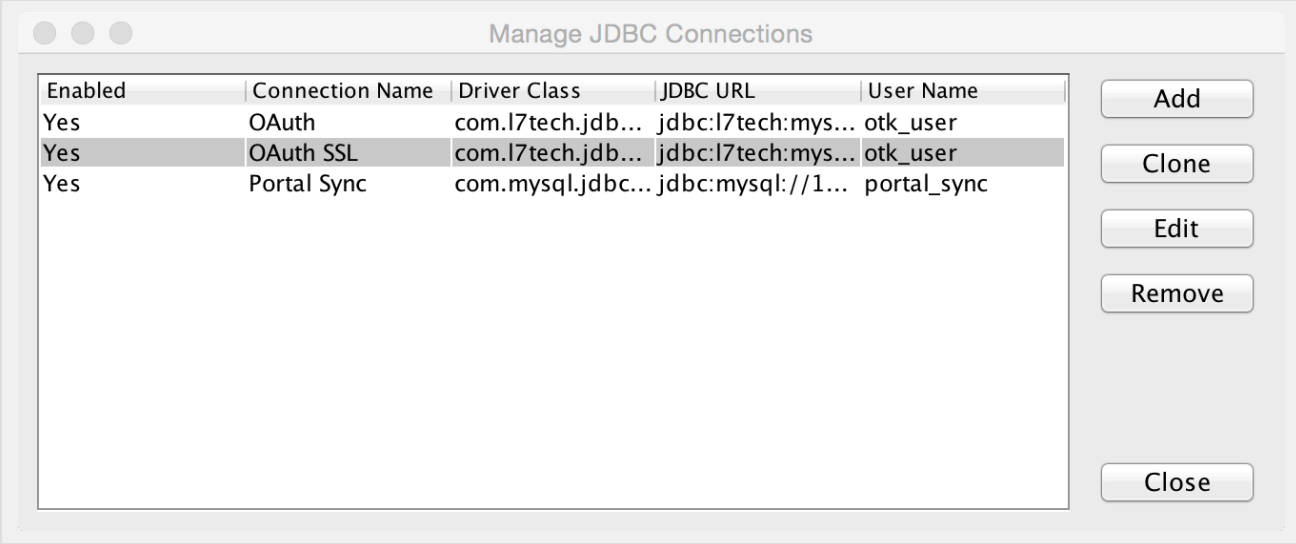
Password		${secpass.otk_user.plaintext}	User password is stored in the 'Managed Password's allowing a secure reference
Additional Properties			
		DatabaseName	otk_db
		AuthenticationMethod	kerberos
		EncryptionMethod	SSL
		HostNameInCertificate	database-01.ps.ca.com
		KeyStore	/opt/SecureSpan/JDK/jre/lib/security/cacerts
		KeyStorePassword	changeit
		TrustStore	/opt/SecureSpan/JDK/jre/lib/security/TrustStore
		TrustStorePassword	changeit

9. ## Test your SSL enabled JDBC connection

Once you have populated the Connection Properties for your JDBC data source will then be able to test the connection using the 'Test' button located in the bottom
left of the menu. The test button will attempt a TLS/SSL enabled connection to the SQL server you selected. If successful you should see the screen below. If you see an
error message displayed take note of the message and review the steps again to validate the file permissions are correct for all of your certificates, keystone and truststore.



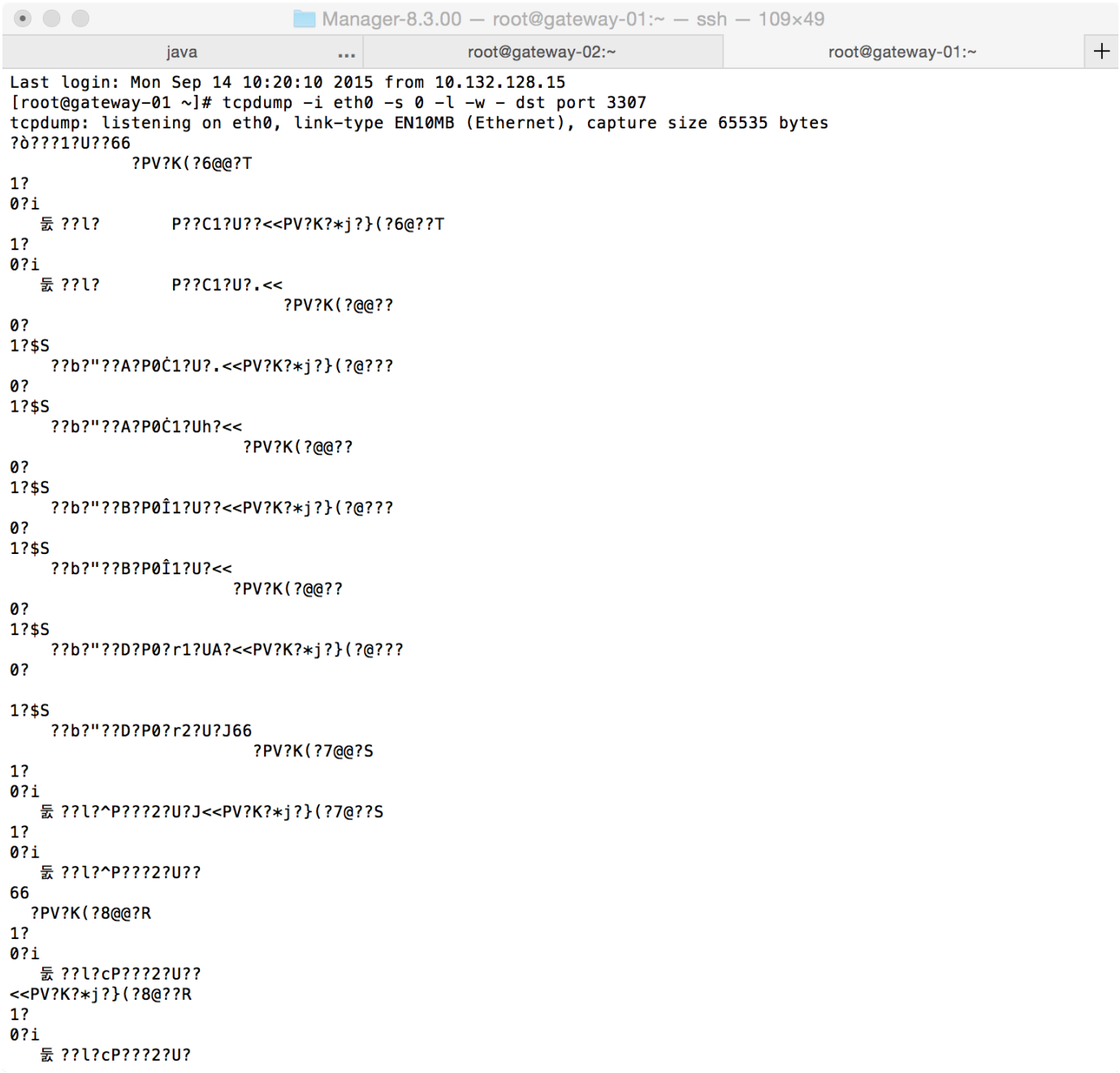
Below is a cloned instance



10. ## Validate SSL is in use with 'tcpdump' (optional)

One method to validate the connections between your gateway and the MySQL host is to use the tcpdump package which is available on most Linux distributions. You can see an example
command below along with the encrypted output.

```
[root@database-01 ~]# tcpdump -i eth0 -s 0 -l -w - src port 3307
```



You now have a JDBC connection with TLS/SSL enabled. End of document.

JDBC Whitepaper: http://www.informationweek.com/pdf_whitepapers/approved/1338916974_Security_Tutorial.pdf

CA APIM JDBC Connection Properties: https://wiki.ca.com/display/GATEWAY84/JDBC%20Connection%20Properties

Java Keytool: http://docs.oracle.com/javase/6/docs/technotes/tools/solaris/keytool.html

Creating Truststore and Keystore: https://docs.oracle.com/cd/E19509-01/820-3503/6nf1il6er/index.html

Related articles

- CA APIM Gateway - JDBC SSL with Failover
- CA APIM Gateway - JDBC over SSL/TLS
- MySQL Master - Master Replication over TLS/SSL - DRAFT READY TO REVIEW