Tungsten Cluster Master Class

Advanced: Securing Your Cluster with SSL

Matthew Lang

VP of Customer Success, Americas

Topics

In this short course, we will discuss:

- What is SSL?
- Deploying SSL for Cluster communications
- Deploying SSL for Tungsten Connector

Background SSL

What is SSL

- Secure Sockets Layer
- Actually, it's depreciated
  - Modern applications use TLS (Transport Layer Security)
  - TLS is just an updated version of SSL
  - Even though we're using TLS, we still refer to it as SSL
- Provides encryption between client and server
- Provides verification that the server's advertised name is correct
  - Difficult for server to masquerade as another server
  - Can be authenticated by a third party (CA – Certificated Authority)
  - Or, we can have a client "trust" a server

Why Use SSL

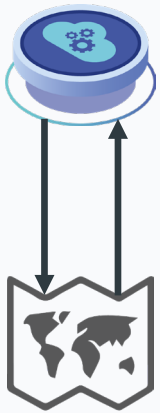
- Encryption helps protect sensitive data
- Many local networks are not controlled by business
  - Cloud
  - Co-location
- Requirement for many organizations
  - PHI (Protected Health Information)
  - PCI (Payment Card Industry) compliance, for protecting credit card data
  - Other compliance
- Prevent “man-in-the-middle” attacks. Properly configured SSL guarantees that the target server cannot be impersonated.

Key Pair

1. Private Key
   - Never shared
   - Used to decrypt
   - Used to “sign”
2. Public Key
   - Shared
   - Used to encrypt
   - Used to “verify”

- A key pair contains cryptographic information to allow encryption between a client and server
- When a key pair also contains information about the server name and organization, it is called a “certificate.”
- A key pair can be issued and “signed” by a Certificate Authority (CA)

SSL Sample with Certificate Authority (CA)



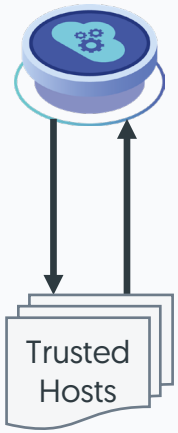
Hi, I want to connect to node1.mydevsite.com

I'm node1.mydevsite.com. Sending signature.

Hold on, let me verify.

Ok, I trust you, sending you encrypted data now....

SSL Sample, Self Signed



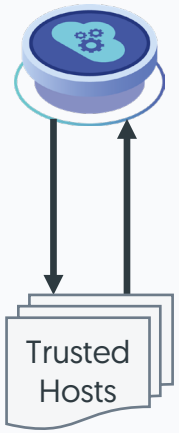
Hi, I want to connect to node1.mydevsite.com

I'm node1.mydevsite.com. Sending signature.

Hold on, let me check if you're in my list of trusted hosts.

Ok, I you're in the list, sending you encrypted data now....

SSL Sample, Host Mismatch



Hi, I want to connect to node1.mydevsite.com

I'm node1.mydevsite.com. Sending signature.

Hold on, let me check if you're in my list of trusted hosts.

Hmm, your signature looks good, but DNS says you're really node1.mydatasteal.com. I'm not going to send you anything

Challenges with SSL

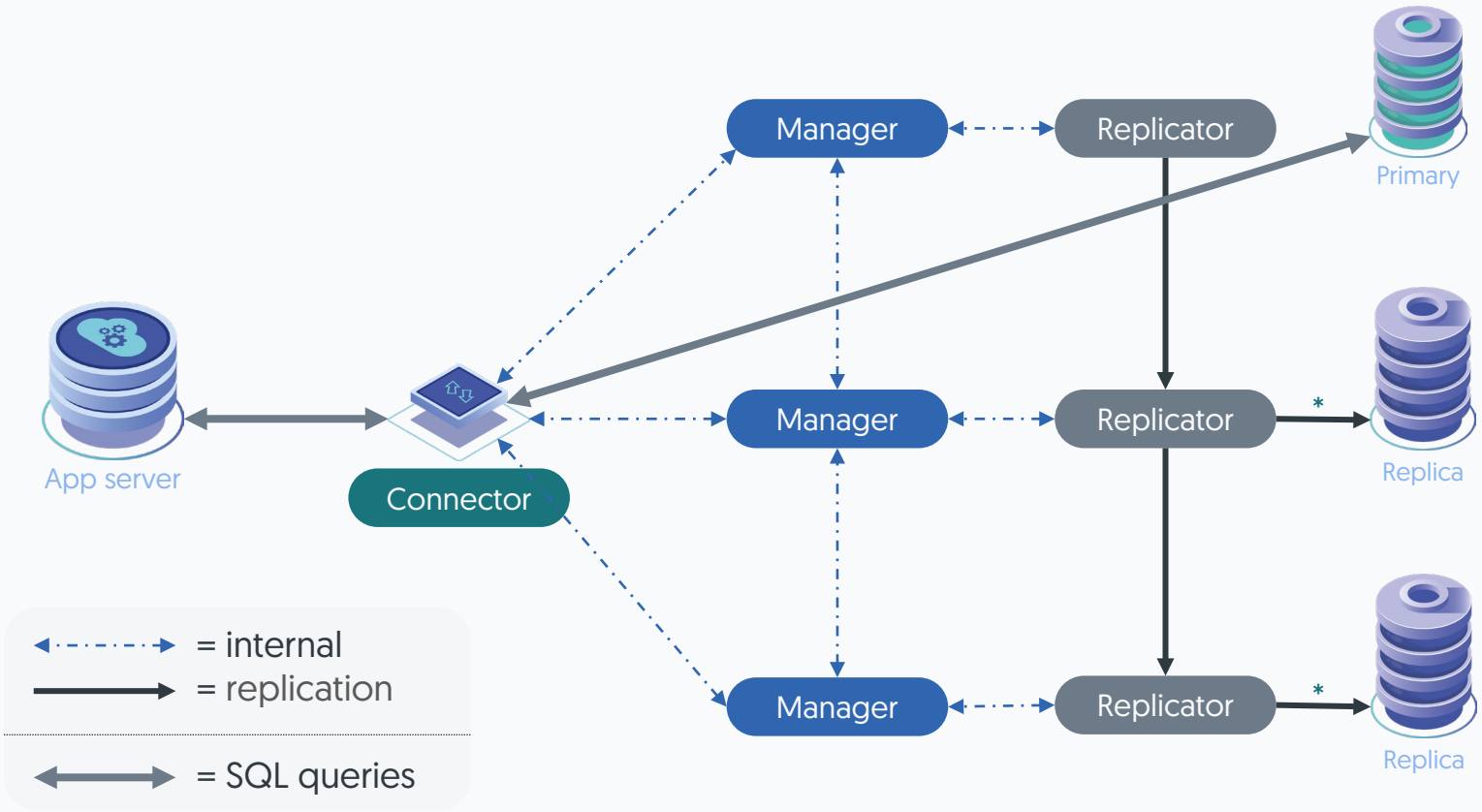
- Supported ciphers
- Supported protocols
- DNS mismatches
  - Incorrectly configured hostnames
  - Aliases
  - Incorrect or incomplete DNS entries
  - Forward/Reverse lookup
- Many different file types for certificates
- Lack of helpful error messages

Using Certificates with Java

- Two repository files: keystore and truststore
  - Keystore is where you store your key pairs, or at a minimum, private keys
  - Truststore is where public keys of trusted sites are stored
- A keystore is normally password protected
- Managed by `keytool`
- To have `host1` connect to (and trust) `host2`, put `host2`'s public key into `host1`'s truststore
- To have many hosts all connect to and trust each other:
  - Create a key pair on `host1`
  - Add private key keystore
  - Add public key to truststore
  - Copy the keystore and truststore to all hosts
  - Now all hosts will be using the same keypair

SSL within Tungsten Cluster

Data Streams



Installing Security for Tungsten Services

- In tungsten.ini: `disable-security-controls=false`
- Enables SSL for manager, replication, and connector communication
- But not for SQL queries (explained later)
- Generate key pair
- Creates keystore and truststore on each host in `/opt/continuent/share`
- Also creates `passwords.store`, contains hash of keystore passwords
- When using staging method, the initial keystore and truststore are copied to all hosts
- When using INI method, additional steps are required

Installing Security for Tungsten Services (INI)

- INI method of installation runs an independent installation script on each host
- Therefore, the key pairs will be different, and the nodes will NOT be able to communicate with each other
- Add the following to tungsten.ini and install:

```
disable-security-controls=false
start-and-report=false
```

- Then on one host, copy the appropriate files to all other nodes:

```
shell> tpm copy --init
```

- This command does the following:

```
shell> for i in `seq 2 6`; do scp /opt/continuent/share/[jpt]* db$i:/opt/continuent/share/; done
shell> for i in `seq 2 6`; do scp /opt/continuent/share/.[jpt]* db$i:/opt/continuent/share/; done
```

- Then start the services with `startall`

Updating an Existing Configuration

- In tungsten.ini: `disable-security-controls=false`
- However, just adding that line won't create certificates needed for SSL
- Must update like this:

```
shell> tools/tpm update --replace-jgroups-certificate --replace-tls-certificate --replace-release
```

- This will create the keypair and certificates
- If INI method, copy the files as explained previously
- Restart all services
- Due to restart, this WILL take the cluster offline for a moment!

Verifying SSL

- From `cctrl`

```
$ cctrl
Tungsten Clustering 6.1.6 build 6
nyc: session established, encryption=true, authentication=true
[LOGICAL] /nyc > ls

DATASOURCES:
+----------------------------------------------------------------------------+
|db1(master:ONLINE, progress=0, THL latency=1.067)                           |
|STATUS [OK] [2020/08/31 05:47:33 PM UTC][SSL]                               |
+----------------------------------------------------------------------------+
```

- `trepctl status`

```
masterConnectUri     : thls://localhost:/
masterListenUri      : thls://db1:2112/
```

SSL for Connector

MySQL Setup

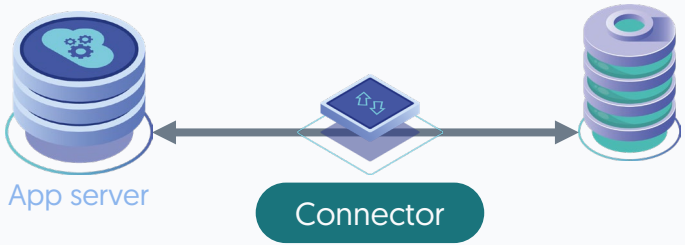
- Most newer version of MySQL come with SSL enabled and certs created
- You can easily create certificates with `mysql_ssl_rsa_setup` (MySQL 5.7+)
- Check datadir for certs
- Copy certs from one node to all others and restart MySQL. All nodes need the same certs.
- Verify SSL connectivity:

```
$ mysql -uapp_user -psecret -h 127.0.0.1 --ssl-ca=ca.pem
mysql> status
--------------
mysql  Ver 14.14 Distrib 5.7.31, for Linux (x86_64) using  EditLine wrapper

Connection id:        41617
Current database:
Current user:               app_user@db1
SSL:                              Cipher in use is ECDHE-RSA-AES128-GCM-SHA256
Current pager:        stdout
```

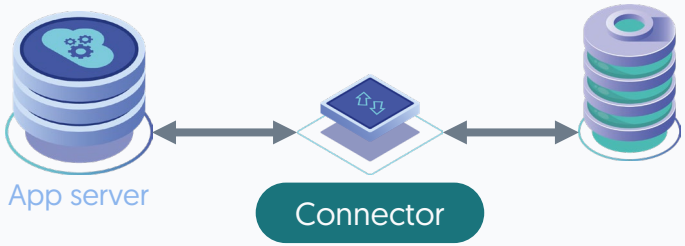
SSL in Bridge Mode

- In Bridge Mode, the connector simply routes traffic to the database
- App is connected directory to MySQL
- Therefore, as long as SSL is setup on the MySQL servers and the app, there is no other configuration necessary in Tungsten.
- All traffic between app and database is encrypted!



SSL Using Proxy Mode

- In proxy mode, connector terminates connection from client, and establishes new connection to MySQL
- Two possible paths for SSL:
  - From App server to Connector
  - From Connector to Database
- If Connector is installed on App server, you need SSL from Connector to MySQL
- Otherwise, you need SSL for BOTH App Server to Connector, and Connector to MySQL



Configure SSL from Connector to MySQL

- Make sure all MySQL databases are using the same certificates. Check permissions!
- Convert MySQL certs into ”pkcs12” format. This combines the key pair into a single encrypted file:

```
openssl pkcs12 -export -inkey client-key.pem -in client-cert.pem -out client-cert.p12 -passout
pass:secret
```

- Now create a keystore for the connector that contains the certificate from above:

```
keytool -importkeystore -srckeystore client-cert.p12 -srcstoretype PKCS12 \
     -destkeystore tungsten_connector_keystore.jks -deststorepass secret -srcstorepass secret
```

- Also import the CA certificate into the keystore:

```
keytool -import -alias mysqlServerCACert -file ca.pem -keystore tungsten_connector_keystore.jks \
     -storepass secret -noprompt
```

- Finally, import the signed CA certificate into truststore:

```
keytool -import -alias mysqlServerCACert -file ca.pem -keystore tungsten_connector_truststore.ts \
     -storepass secret -noprompt
```

Options to Enable SSL to MySQL

```
# enable SSL from the connector to the DB
connector-ssl=true
java-connector-keystore-password=secret
java-connector-truststore-password=secret
java-connector-truststore-path=/home/tungsten/tungsten_connector_truststore.ts
java-connector-keystore-path=/home/tungsten/tungsten_connector_keystore.jks
```

Tungsten Connection Status

```
mysql> tungsten connection status;
+-------------------------------------------------------------------------------+
| Message                                                                       |
+-------------------------------------------------------------------------------+
| db1@east(master:ONLINE) STATUS(OK), QOS=RW_STRICT SSL.IN=false SSL.OUT=true |
+-------------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

- `SSL.IN=false` because we have not yet configured application to connector SSL
- `SSL.OUT=true` is the connector the database
- If connectors are installed app servers, this is the desired configuration

Configure App to Connector SSL

- Use this if connectors are not installed on app servers
- From the previous step, we've already added MySQL certificates into the connector's keystore.
- So the connector can accept SSL connections when the app simply uses certificate:

```
$ mysql -u app_user -psecret -h 127.0.0.1 --ssl-ca=/home/tungsten/ca.pem

mysql> tungsten connection status;
+-------------------------------------------------------------------------------+
| Message                                                                       |
+-------------------------------------------------------------------------------+
| db1@east(master:ONLINE) STATUS(OK), QOS=RW_STRICT SSL.IN=true SSL.OUT=true    |
+-------------------------------------------------------------------------------+
1 row in set (0.00 sec)
```

- Notice now SSL.IN=true

Verifying SSL with tcpdump

- Connector, without SSL: `mysql> show databases;`
- On primary datasource: `sudo tcpdump -A port 13306`

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
13:29:50.187112 IP trainingdb2.46712 > trainingdb1.13306: Flags [P.], seq 1027118425:1027118444, ack
4197839122, win 261, options [nop,nop,TS val 1647648844 ecr 2428897262], length 19
E..Gx.@.....
../
....x3.=8.Y.5.............
b5 L.........show databases
13:29:50.187397 IP trainingdb1.13306 > trainingdb2.46712: Flags [P.], seq 1:207, ack 19, win 210,
options [nop,nop,TS val 2428965324 ecr 1647648844], length 206
E.....@...x.
...
../3..x.5..=8.l...........
....b5
L.....K....def.information_schema.SCHEMATA.SCHEMATA.Database.SCHEMA_NAME.!..................".....inf
ormation_schema
...     east_load.....hr.....mysql.....performance_schema...       .sys...
.tungsten_alpha.......".
```

Verifying SSL with tcpdump

- Connector, with SSL: `mysql> show databases;`
- On primary datasource: `sudo tcpdump -A port 13306`

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
13:31:54.495055 IP trainingdb3.45998 > trainingdb1.13306: Flags [P.], seq 3526192735:3526192836, ack 2999926415, win
252, options [nop,nop,TS val 3592467640 ecr 1866164827], length 101
E...<.@...#.
..n
.....3..-n_..>.....3......
. ..o;j[....`<X {I....H..dm{4;./. ..s.{g..)...gg.xfU.y.......G|......B..{w..W....|[GX.`R.).e.`(A|.sd.. /.....
13:31:54.495415 IP trainingdb1.13306 > trainingdb3.45998: Flags [P.], seq 1:278, ack 101, win 227, options [nop,nop,TS
val 1866691921 ecr 3592467640], length 277
E..I.6@....,
...
..n3.....>..-n............
oCuQ. ..........3..M......>...u......0.....;$t....n..Pi.....3..0...;@PM'tp2.{..Y...A..0.8@...>.{.P\
x.e.J.ig..~.....<.V..x....6..mE....g.#....s{../..DW..........
        ./.3.....L..&B.QC.......`......x.2IG._......r+#cU!...a..q.......    .......Zv............ouM..'.......j)6..p..O
..X..b..UU....
13:31:54.495717 IP trainingdb3.45998 > trainingdb1.13306: Flags [.], ack 278, win 261, options [nop,nop,TS val
3592467641 ecr 1866691921], length 0
E..4<.@...#.
```

Other SSL Options

Other SSL Options

- Later versions of MySQL and Tungsten Clustering make SSL deployment simple
- However, you can override certification creation and create your own:
  - JGroups certificates, used for cluster communications
  - Certificates for replication SSL
  - Use a signed certificate from a Certificate Authority instead of self-signed certificates
  - tpm cert can handle all of these scenarios and assist with Certificate Rotation
- Provision SSL for replication only

Summary

What we have learnt today

- Why deploy SSL?
  - Data protection in flight
  - Encryption in flight
  - Server Authentication
- Deploying Cluster SSL using `tpm`
- Deploying SSL for the connector and connecting to MySQL
- Verifying SSL encryption

Thank you for listening

continuent.com

Matthew Lang

VP of Customer Success, Americas