Configuring Transport Layer Security (TLS)

AtScale can be configured to support Transport Layer Security (TLS) between AtScale and your client applications. TLS is the successor to Secure Sockets Layer (SSL).

To enable TLS, you must obtain an X.509 certificate for the AtScale host issued by a trusted certificate authority (CA). You then copy it to all AtScale hosts prior to configuration. This certificate should contain the public portion of the certificate, plus any intermediate CA certificates and root CA certificates needed to build the chain to the root CA. The private portion of the certificate (the private key) should be in a separate file.

When AtScale is configured in secure mode, all external client applications must use secure protocols when communicating with the AtScale services. This includes:

- ODBC and JDBC connections from your BI clients to the AtScale engine (SQL).
- XMLA connections from your BI clients to the AtScale engine (MDX).
- Browser connections to the AtScale Design Center, and AtScale Administration Console web applications.
- Connections to LDAP directory services.

Note: To change AtScale ports from the default, see Configuring Ports.

Using A Self-Signed Certificate

In case you want to use a self-signed certificate:

1. Log in to the AtScale system as the AtScale user
2. Create a cert directory:

```
mkdir /opt/atscale/certs
```

3. Make that directory the working directory:

```
cd /opt/atscale/certs
```

4. Optionally, check the OpenSSL version: `openssl version`

   You should obtain a response like this one:

```
OpenSSL 1.0.2k-fips  26 Jan 2017
```

5. Create a configuration file openssl.conf with the following content, and set the appropriate hosts, organizations, and addresses:

```
[req]
default_bits          = 2048
default_md            = sha256
encrypt_key           = no
prompt                = no
distinguished_name    = subject
req_extensions        = req_ext
x509_extensions       = x509_ext

[ subject ]
C                     = US
ST                    = Florid
L                     = Orlando
O                     = MyOrg
OU                    = MyOrg Office
emailAddress          = name@mycompany.com
CN                    = test-server.mycompany.com

[ req_ext ]
subjectKeyIdentifier   = hash
basicConstraints       = CA:FALSE
keyUsage               = digitalSignature, keyEncipherment
extendedKeyUsage       = serverAuth, clientAuth
subjectAltName         = @alternate_names
nsComment              = "Self-Signed SSL Certificate"

[ x509_ext ]
subjectKeyIdentifier   = hash
authorityKeyIdentifier = keyid,issuer
basicConstraints       = CA:FALSE
keyUsage               = digitalSignature, keyEncipherment
extendedKeyUsage       = serverAuth, clientAuth
subjectAltName         = @alternate_names
nsComment              = "Self-Signed SSL Certificate"

[ alternate_names ]
DNS.1                  = localhost
DNS.2                  = test-server.mycompany.com
IP.1                   = 127.0.0.1
```

6. Generate the CRT and key:

```
openssl req -config openssl.conf -new -sha256 -newkey rsa:2048 -nodes -keyout private.key -x509 -days 825 -out
certificate.crt
```

7. Create a PEM:

```
openssl x509 -in certificate.crt -out cert.pem
```

8. Open the /opt/atscale/conf/atscale.yaml file with a text editor, and edit the following settings to match your server:

```
loadbalancer_dns_name: "test-server.mycompany.com"
tls:
  enabled: True
  certificate: "/opt/atscale/certs/cert.pem"
  key: "/opt/atscale/certs/private.key"
hosts:
  - dnsname: test-server.mycompany.com
```

9. Apply the settings by running the `configurator.sh` tool as described on the last step in the procedure below.

Procedure

1. Copy the X.509-formatted server certificate to the AtScale host(s), for example: `/path/to/server/certificate.pem`
2. Copy the certificate's private key file to the AtScale host(s), for example: `/path/to/server/key.pem`

   It is recommended - but not required - to simply put the certificate and key files in `/opt/atscale/conf`

3. Update the tls properties in `/opt/atscale/conf/atscale.yaml`. For example:

```
tls:
    enabled: true
    certificate: "/path/to/server/certificate.pem"
    key: "/path/to/server/key.pem"
```

4. Execute configurator.sh with the `--apply` option to apply the new configuration.

```
sudo su - atscale
cd /opt/atscale/versions/<package_version>
./bin/configurator.sh --apply
```

   In case the system is configured to use a different user than `atscale`, you should use this user in the 'sudo' command above. For more information, see Install Stand-Alone AtScale.

Keeping Certificates Imported In Truststore

One of the results from running the `configurator.sh` tool is that the information about certificates in the truststore is deleted. This means you would need to import the certificates again. To avoid importing certificates repeatedly, you can set the system to preserve them in the following way:

1. Put the certificates in a dedicated directory.

   The default directory is: /opt/atscale/data/security/crt

2. Start editing the `/opt/atscale/conf/atscale.yaml` file.
3. In the `tls` section, add the `custom_truststore` option, and use `path` to specify the directory from step 1:

```
tls:
    enabled: true
    certificate: "/path/to/server/certificate.pem"
    key: "/path/to/server/key.pem"
    custom_crt:
        path: "/path/to/certificates"
```

4. Save the changes.
5. Execute configurator.sh with the `--apply` option as described above.