Cloudera Data Science Workbench

Enabling TLS/SSL for Cloudera Data Science Workbench

Date published: 2020-02-28
Date modified: 2021-02-25

https://docs.cloudera.com/

Legal Notice

Contents

Enabling TLS/SSL for Cloudera Data Science Workbench

Cloudera Data Science Workbench uses HTTP and WebSockets (WS) to support interactive connections to the Cloudera Data Science Workbench web application. However, these connections are not secure by default. This topic describes how you can use TLS/SSL to enforce secure encrypted connections, using HTTPS and WSS (WebSockets over TLS), to the Cloudera Data Science Workbench web application.

Starting with version 1.6, Cloudera Data Science Workbench defaults to using TLS 1.2. The default cipher suites have also been upgraded to Mozilla's Modern cipher suites.

Cloudera Data Science Workbench can be configured to use a TLS termination proxy to handle incoming connection requests. The termination proxy server will decrypt incoming connection requests and forward them to the Cloudera Data Science Workbench web application. A TLS termination proxy can be internal or external.

Note: If you are using an internal custom Certificate Authority, you must add your CA to the "TLS_ROOTCA" configuration. See Configuring Custom Root CA Certificate for further information.

Internal Termination

An internal termination proxy will be run by Cloudera Data Science Workbench's built-in load balancer, called the ingress controller, on the master host.

The ingress controller is primarily responsible for routing traffic and load balancing between Cloudera Data Science Workbench's web service backend. Once configured, as shown in the instructions that follow, it will start terminating HTTPS traffic as well. The primary advantage of internal termination approach is simplicity.

External Termination

External TLS termination can be provided through a number of different approaches.

Common examples include:

- Load balancers, such as the AWS Elastic Load Balancer
- Modern firewalls
- Reverse web proxies, such as nginx
- VPN appliances supporting TLS/SSL VPN

Organizations that require external termination will often have standardized on single approach for TLS. The primary advantage of this approach is that it allows such organizations to integrate with Cloudera Data Science Workbench without violating their IT department's policies for TLS. For example, with an external termination proxy, Cloudera Data Science Workbench does not need access to the TLS private key.

Load balancers and proxies often require a URL they can ping to validate the status of the web service backend. For instance, you can configure a load balancer to send an HTTP GET request to /internal/load-balancer/health-ping. If the response is 200 (OK), that means the backend is healthy. Note that, as with all communication to the web backend from the load balancer when TLS is terminated externally, this request should be sent over HTTP and not HTTPS.

Note that any terminating load balancer must provide the following header fields so that Cloudera Data Science Workbench can detect the IP address and protocol used by the client:

- X-Forwarded-For (client's IP address),
- X-Forwarded-Proto (client's requested protocol, i.e. HTTPS),
- X-Forwarded-Host (the "Host" header of the client's original request).

See Configuring HTTP Headers for Cloudera Data Science Workbench for more details on how to customize HTTP headers required by Cloudera Data Science Workbench.

Private Key and Certificate Requirements

The TLS certificate issued by your CA must list both, the Cloudera Data Science Workbench, as well as a wildcard for all first-level subdomains.

For example, if the Cloudera Data Science Workbench domain is cdsw.company.com, then the TLS certificate must include both cdsw.company.com and *.cdsw.company.com.

Creating a Certificate Signing Request (CSR) and Key/Certificate Pair

Use the following steps to create a Certificate Signing Request (CSR) to submit to your CA. Then, create a private key/certificate pair that can be used to authenticate incoming communication requests to Cloudera Data Science Workbench.

About this task

Important: Make sure you use openssl, and not keytool, to perform these steps. Keytool does not support a wildcard Subject Alternative Name (SAN) and cannot create flat files.

Procedure

1. Create a cdsw.cnf file and populate it with the required configuration parameters including the SAN field values.

```
vi cdsw.cnf
```

2. Copy and paste the default openssl.cnf from: http://web.mit.edu/crypto/openssl.cnf.
3. Modify the following sections and save the cdsw.cnf file:

```
[ CA_default ]
default_md = sha2

[ req ]
default_bits       = 2048
distinguished_name = req_distinguished_name
req_extensions     = req_ext

[ req_distinguished_name ]
countryName                 = Country Name (2 letter code)
stateOrProvinceName         = State or Province Name (full name)
localityName               = Locality Name (eg, city)
organizationName           = Organization Name (eg, company)
commonName                 = Common Name (e.g. server FQDN or YOUR name)

[ req_ext ]
subjectAltName = @alt_names

[alt_names]
DNS.1   = *.cdsw.company.com
DNS.2   = cdsw.company.com
```

Key points to note:

- The domains set in the DNS.1 and DNS.2 entries above must match the DOMAIN set in cdsw.conf.
- The default_md parameter must be set to sha256 at a minimum. Older hash functions such as SHA1 are deprecated and will be rejected by browsers, either currently or in the very near future.
- The commonName (CN) parameter will be ignored by browsers. You must use Subject Alternative Names.

4. Run the following command to generate the CSR.

```
openssl req -out cert.csr -newkey rsa:2048 -nodes -keyout private.key -c
onfig cdsw.cnf
```

This command generates the private key and the CSR in one step. The -nodes switch disables encryption of the private key (which is not supported by Cloudera Data Science Workbench at this time).

5. Use the CSR and private key generated in the previous step to request a certificate from the CA. If you have access to your organization's internal CA or PKI, use the following command to request the certificate. If you do not have access, or are using a third-party/commercial CA, use your organization's respective internal process to submit the request.

```
openssl x509 -req -days 365 -in cert.csr -CA ca.crt -CAkey ca.key -CAcre
ateserial -out <your_tls_cert>.crt -sha256 -extfile cdsw.cnf -extensions r
eq_ext
```

6. Run the following command to verify that the certificate issued by the CA lists both the required domains, cdsw .company.com and *.cdsw.company.com, under X509v3 Subject Alternative Name.

```
openssl x509 -in <your_tls_cert>.crt -noout -text
```

You should also verify that a valid hash function is being used to create the certificate. For SHA-256, the value under Signature Algorithm will be sha256WithRSAEncryption.

Configuring Internal Termination

Depending on your deployment (CSD or RPM), use one of the following sets of instructions to configure internal termination.

CSD Deployments

Provides steps to configure internal termination in a CDS deployment.

Procedure

1. Log in to the Cloudera Manager Admin Console.
2. Navigate to the CDSW service and click Configuration.
3. Search for the following properties and configure as required.
   - Enable TLS - When enabled, this property enforces HTTPS and WSS connections. The server will now redirect any HTTP request to HTTPS and generate URLs with the appropriate protocol.
   - TLS Key for Internal Termination - Set to the path of the TLS private key.
   - TLS Certificate for Internal Termination - Set to the path of the TLS certificate.

     Certificates and keys must be in PEM format.
4. Click Save Changes.
5. Restart the CDSW service.

RPM Deployments

Provides steps to configure internal termination in a RPM deployment.

Procedure

Configure the following properties in cdsw.conf (on all Cloudera Data Science Workbench hosts).

- TLS_ENABLE - When set to true, this property enforces HTTPS and WSS connections. The server will now redirect any HTTP request to HTTPS and generate URLs with the appropriate protocol.

- TLS_KEY - Set to the path of the TLS private key.
- TLS_CERT - Set to the path of the TLS certificate.

  Certificates and keys must be in PEM format.

You can configure these properties either as part of the installation process or after the fact. If you make any changes to cdsw.conf after installation is complete, make sure to restart the master and worker hosts as needed.

Configuring External Termination

Depending on your deployment (CSD or RPM), use one of the following sets of instructions to configure external termination.

CSD Deployments

Provides steps to configure external termination in a CDS deployment.

Procedure

1. Log in to the Cloudera Manager Admin Console.
2. Navigate to the CDSW service and click Configuration.
3. Search for the following properties and configure as required.
   - Enable TLS - When enabled, this property enforces HTTPS and WSS connections. The server will now redirect any HTTP request to HTTPS and generate URLs with the appropriate protocol.

     The TLS Key for Internal Termination and TLS Certificate for Internal Termination properties must be left blank.
4. Click Save Changes.
5. Restart the CDSW service.

RPM Deployments

Provides steps to configure external termination in a RPM deployment.

Procedure

Configure the following property in cdsw.conf (on all Cloudera Data Science Workbench hosts).

- TLS_ENABLE - When set to true, this property enforces HTTPS and WSS connections. The server will now redirect any HTTP request to HTTPS and generate URLs with the appropriate protocol.

  The TLS_KEY and TLS_CERT properties must be left blank.

You can configure this property either as part of the installation process or after the fact. If you make any changes to cdsw.conf after installation is complete, make sure to restart the master and worker hosts as needed.

Configuring Custom Root CA Certificate

If your organization uses its own custom Certificate Authority, CDSW engines will not be able to automatically recognise the custom CA's root certificate. This topic describes how to add your internal root CA certificate to CDSW so that it is inserted into the engine's root certificate store every time a session (or any workload) is launched. This will allow processes inside the engine to communicate securely with the ingress controller.

About this task

Note: You can also make CDSW aware of your internal root CA by pointing the path to your custom root CA certificate using the property TLS_ROOTCA in your CDSW configuration. See Configuring cdsw.conf properties. This ensures that when CDSW starts, the db-migrate pod will copy the contents of your rootCA to CDSW web UI Admin Root CA Configuration which can then be inserted into the engine's root certificate store every time a session (or any workload) is launched.

Procedure

1. Log in to CDSW as a site administrator.
2. Go to Admin Security.
3. Under the Root CA Configuration section, paste in the contents of your organization's internal root CA certificate file.

   The contents of the certificate should remain in .CRT format. For example:

   ```
   ---BEGIN CERTIFICATE---
   xxxxxxxxxxxxxxxxxxx
   xxxxxxxxxxxxxxxxxxx
   ...
   ---END CERTIFICATE---
   ```

   The contents of this field are then inserted into the engine's root certificate store every time a session (or any workload) is launched. This allows processes inside the engine to communicate with the ingress controller.
4. Click Update.

What to do next

Restart any existing sessions and re-build any existing models to ensure that the newly launched engines pick up this change.