

Chariot MQTT Server Configuration

This document describes the configuration options in the Chariot® MQTT Server UI.

The UI can be accessed at the following URL:

```
http://<server-url>:8080
```

When installed on [Linux](#), [Windows](#) or deployed via [Azure Marketplace](#), the default User login credentials are:

```
username: admin  
password: password
```

When deployed via [AWS Marketplace](#), the default User login credentials are:

```
username: admin  
password: EC2 instance ID for example: i-0049ac1e13e558b70
```

The Chariot MQTT Server Web UI provides multiple configuration pages on the left navigation panel.

- GENERAL
 - [Backup](#)
 - [Web Server](#)
 - [Certificates](#)
- ADMINISTRATIVE
 - [User Accounts](#)
 - [User Sources](#)
 - [MQTT Accounts](#)
 - [License](#)
- MQTT
 - [Server Configuration](#)

User Accounts

The User Accounts page allows for the creating, updating and deleting of [Users](#) and [Roles](#)

Users

The Users page allows for the creating, updating and deleting of Chariot Users. A Chariot User may log into the Chariot Web UI and perform different actions based on the Roles assigned to that User. The main Chariot Users page shows a table of existing Chariot Users. Users can be added by clicking the "Add User" button, or they can be edited by clicking on the individual User in the table.

The screenshot shows the Chariot web application interface. The top navigation bar includes the Chariot logo and a user profile icon. A left sidebar contains a menu with categories: GENERAL (Dashboard, Logging, Alerts, Backup, Web Server, Certificates, Diagnostics), ADMINISTRATIVE (User Accounts, User Sources, License), MQTT CLIENT (Client), and MQTT SERVER (Status, Configuration, Credentials). The main content area is titled "User Accounts" and features a "USERS" tab. At the top of the users section is an "ADD USER" button and a search bar with a dropdown menu set to "Search by Username". Below this is a table with columns for Username, Email, First Name, Last Name, and Roles. A single user entry is visible with the username "admin" and role "admin". The table also includes edit and delete icons for each user. At the bottom right of the table, it indicates "Records per page: 20" and "1-1 of 1".

Add User

Clicking the "Add User" button will display a modal form for creating a new Chariot User.

The form contains the following fields:

- **Username**
 - A unique username for the Chariot User (*required*).
- **Password**
 - A secure password for the Chariot User that will be used to log into the Chariot Web UI (*required*).
- **Email**
 - An email address associated with the Chariot User (*optional*).
- **First Name**
 - A first name associated with the Chariot User (*optional*).
- **Last Name**
 - A last name associated with the Chariot User (*optional*).
- **Roles**
 - A list of Chariot Roles associated with the Chariot User.



By default, a User with the Username **admin** and Password **password** will be created

The "Save" button in the bottom right of the form can be clicked to create the User.

Edit User

Clicking on the edit icon on a Chariot User in the table will display a modal form for editing the Chariot User.

Username	Email	First Name	Last Name	Roles	
bsmith	bsmith@fake.com	Bob	Smith	admin	 
admin				admin	Edit User 

Edit User



Username

Password

Roles

Email

First Name

Last Name

Phone Number

Company

Country

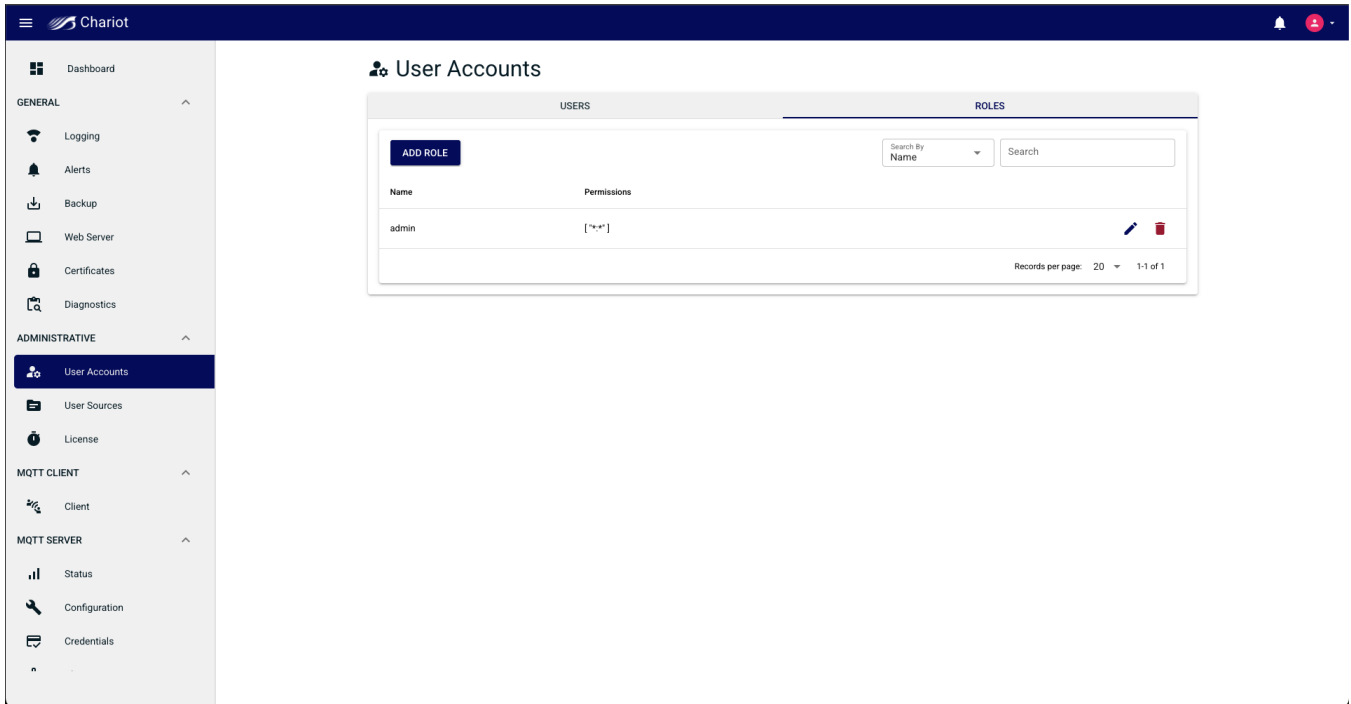
State or Province

SAVE

Leaving the password empty will retain the existing users password.

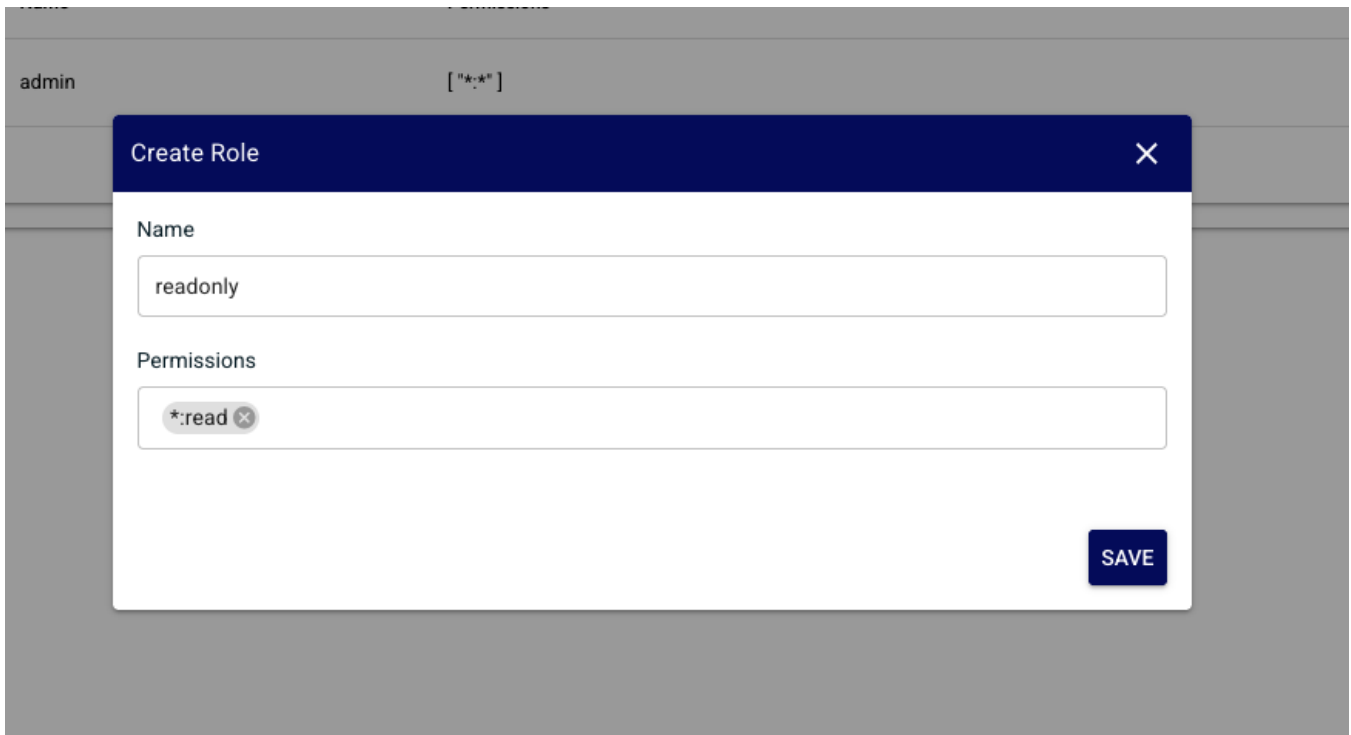
Roles

The Roles page allows for the creating, updating, and deleting of Chariot Roles. A Chariot Role is a collection of permissions that authorized viewing, editing and/or controlling the Chariot MQTT Server. The main Chariot Roles page shows a table of existing Chariot Roles. Roles can be added by clicking the "Add Role" button, or they can be edited by clicking on the individual Role in the table.



Add Role

Clicking the "Add Role" button will display a modal form for creating a new Chariot Role.



The form contains the following fields:

- **Name**
 - A unique name for the Chariot Role.



- **Permissions**

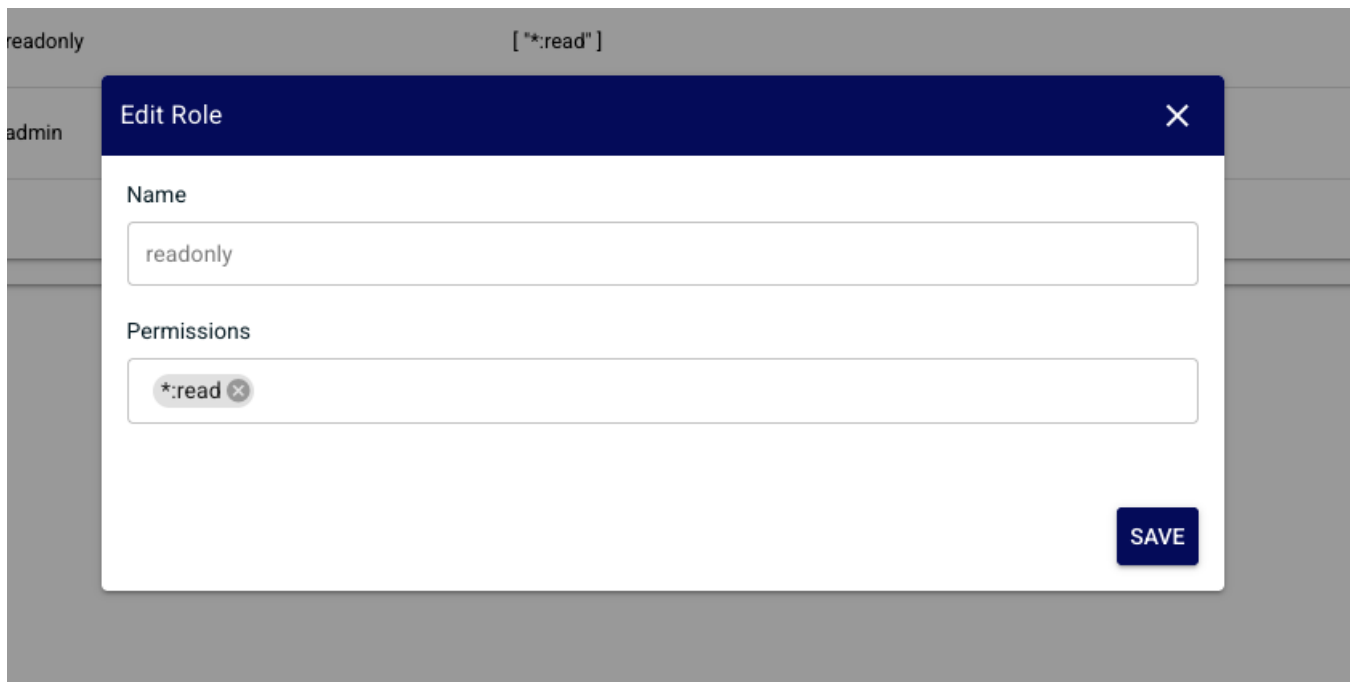
- A list of permissions for the Chariot Role.
- Permissions have the form `<domain>:<action>`
 - The `<domain>` represents a service within Chariot that can be interacted with through the Chariot UI. Some examples are:
 - `user`
 - The User Service for managing Chariot UI users
 - See the Chariot UI under the "Users" tab
 - `role`
 - The Role Service for managing Chariot UI roles
 - See the Chariot UI under the "Roles" tab
 - `mqttuser`
 - The MQTT User Service for managing MQTT Credentials
 - See the Chariot UI under the "MQTT Account Credentials" tab
 - `system`
 - The System Service for configuring Server Name, HTTP, Certificates and Backup/Restore
 - See the Chariot UI under the Web Server, Backup or Certificates pages
 - `server`
 - The MQTT Server Service for configuring and controlling the MQTT server
 - See the Chariot UI under the "MQTT Server Configuration" tab
 - `license`
 - The Licensing Service for licensing the Chariot MQTT Server software
 - See the Chariot UI under the "License" tab
 - The `<action>` represents the scope of this permission for the associated domain. The available actions are:
 - `read`
 - The permission to read/view resources, configurations, and/or data
 - `create`
 - The permission to create new resources and/or configurations
 - `update`
 - The permission to update resources and/or configurations
 - `delete`
 - The permission to delete resources and/or configurations
 - `action`
 - The permission to perform any actions available from a service within Chariot (such as starting/stopping the MQTT server or activating a license)

The "Save" button in the bottom right of the form can be clicked to create the Role.

Edit Role

Clicking on the edit icon on a Chariot Role in the table will display a modal form for editing the Chariot Role.

readonly	[*:read]	 
admin	[*:*]	 



The "Save" button in the bottom right can be clicked to update the Chariot Role once all changes have been made.

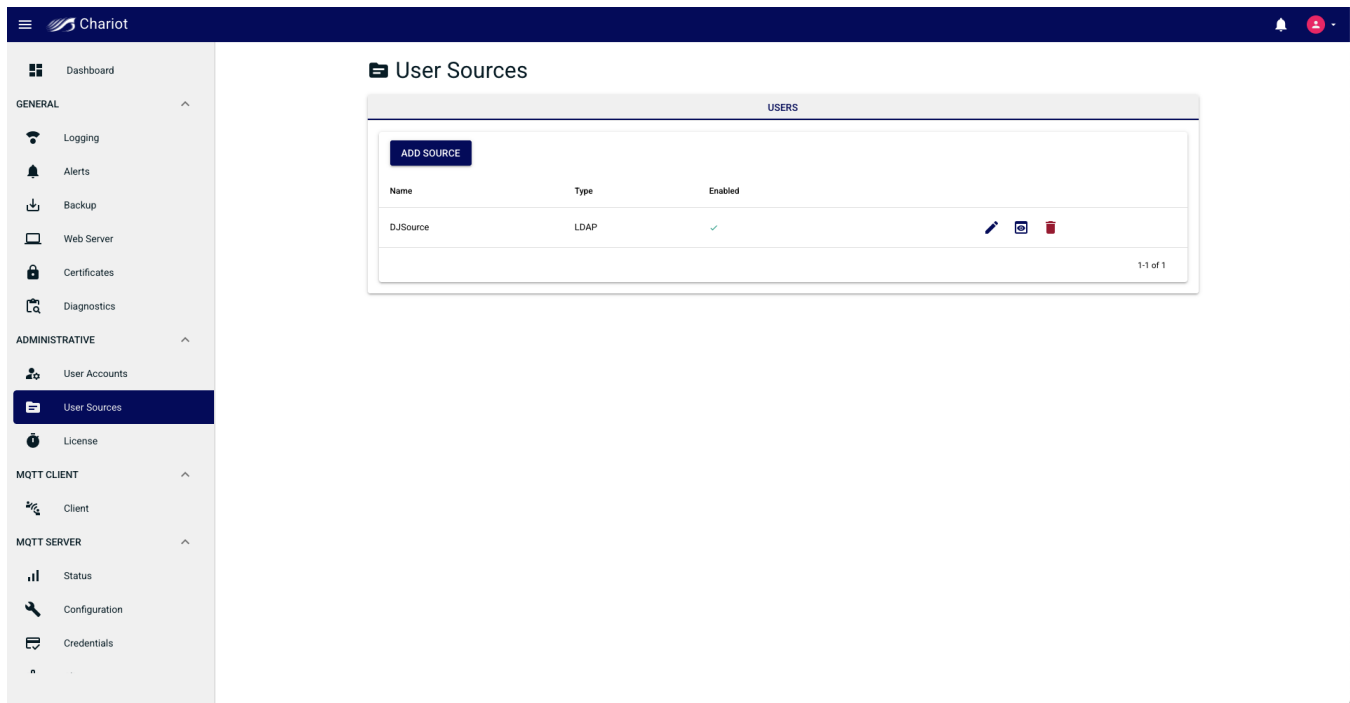
User Sources

The Chariot Security Service can be configured to add an LDAP Realm to use when authenticating and authorizing access via the Chariot UI.



This feature is available in Chariot v2.4.2 and newer

Review the [LDAP and Microsoft Active Directory for Chariot UI/REST Users](#) for configuration details



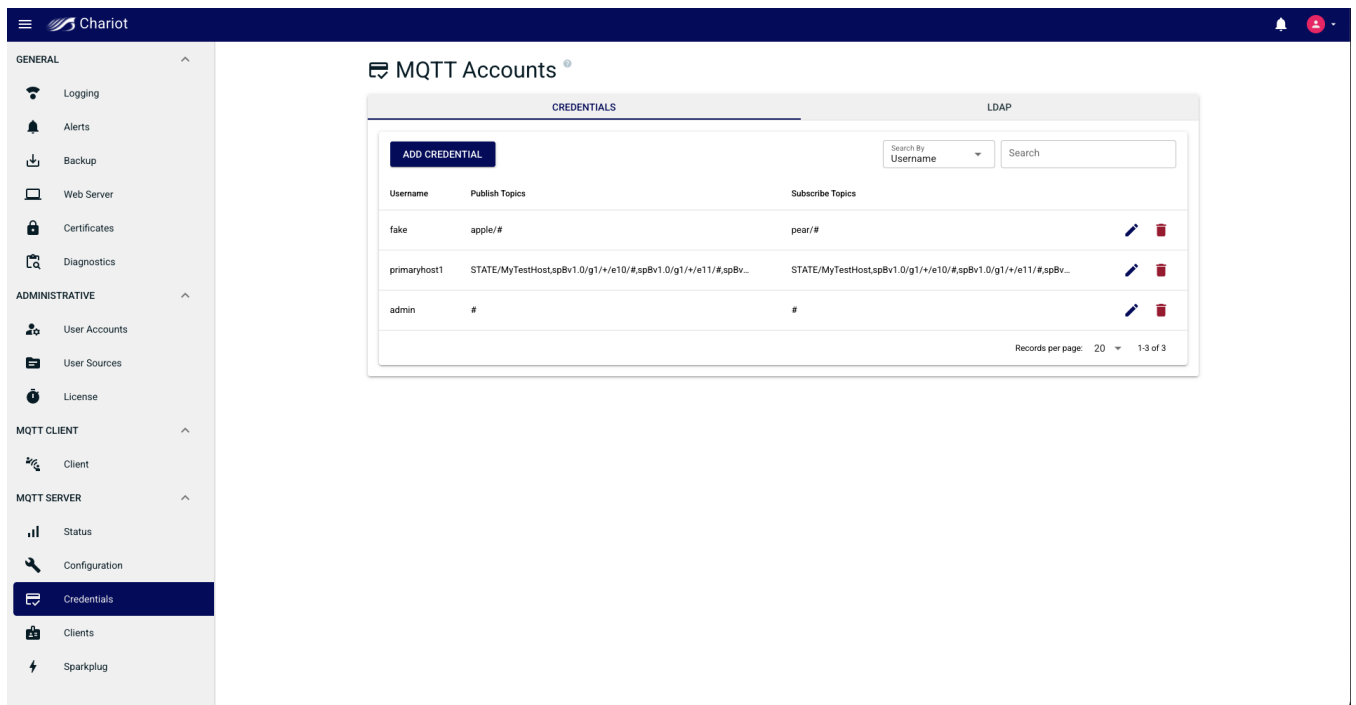
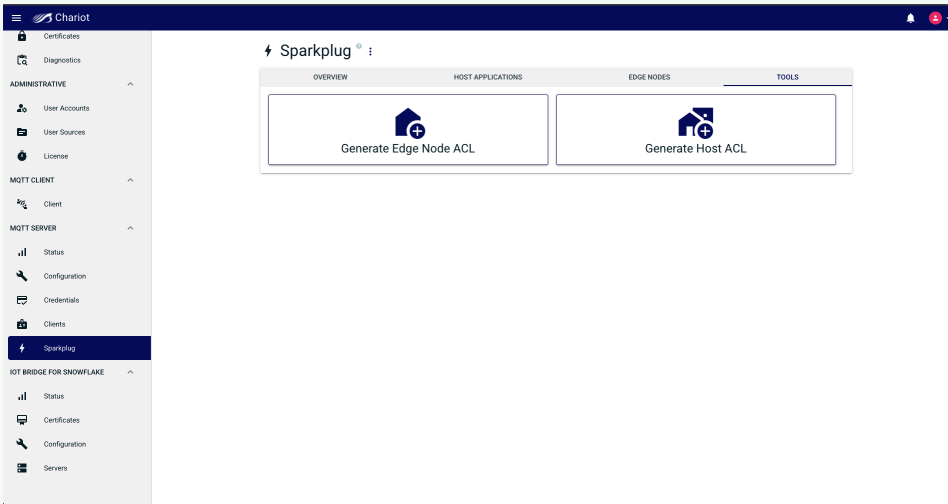
MQTT Accounts

The MQTT Accounts page contains two tabs: Credentials and [LDAP](#)

MQTT Account Credentials are the credentials that clients will use to authenticate when establishing an MQTT connection to the Chariot MQTT Server.

MQTT Account Credentials also contain the access control lists (ACLs) that control which MQTT topics a client can publish and/or subscribe on. MQTT Account Credentials can be added by clicking the "Add Credential" button, or they can be edited by clicking on the individual Credential in the table.

✔ From Chariot release 2.5.0, to avoid manual configuration, the required ACLs can be generated from the **Main Navigation MQTT Sparkplug Tools**. Clicking one of the available options will provide a modal that will help you automatically generate ACLs to be added to the MQTT Account Credentials.



Add Credential

Clicking the "Add Credential" button will display a modal form for creating a new MQTT Credential.

The form contains the following fields:

- **Username**
 - A unique username that an MQTT client will use when connecting.
- **Password**
 - A secure password that an MQTT client will use when connecting.
- **Publish Topics**
 - A list of MQTT topic filters that the client is allowed to publish on (wildcards may be used).
- **Subscribe Topics**
 - A list of MQTT topic filters that the client is allowed to subscribe on (wildcards may be used).

 By default, two default MQTT Credentials are created:

Username **admin** Password **changeme**

Username **anonymous** Password **changeme**

The "Save" button in the bottom right of the form can be clicked to create the Credential.

Topic Examples:

- #
 - Allows publish or subscribe on all topics
- STATE/123
 - Allows publish or subscribe on the specific topic "STATE/123"
- spBv1.0/#
 - Allows publish or subscribe on all topics that start with "spBv1.0/", such as "spBv1.0/g1/DDATA/e1/d1"

Examples of MQTT Account Credentials









Chariot

MQTT Accounts

CREDENTIALS LDAP

ADD CREDENTIAL





Search By Username Search

Username	Publish Topics	Subscribe Topics	
lastcred	test/pub/#	test/pub/#	 
fake	apple/#	pear/#	 
primaryhost1	STATE/MyTestHostspBv1.0/g1/+e10/#spBv1.0/g1/+e11/#spBv...	STATE/MyTestHostspBv1.0/g1/+e10/#spBv1.0/g1/+e11/#spBv...	 
admin	#	#	 

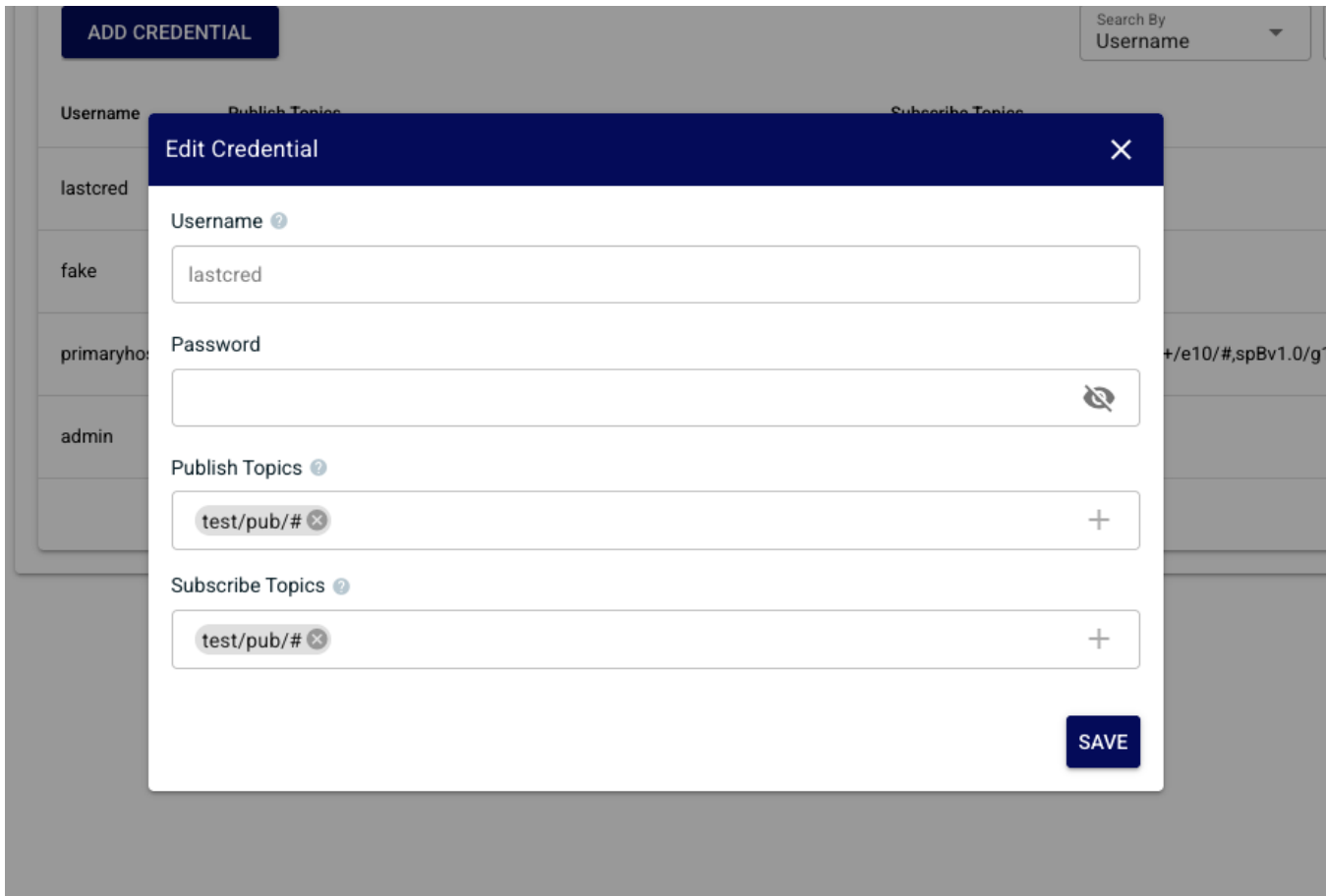
Records per page: 20 1-4 of 4

Edit Credential

Clicking on the edit icon on a MQTT Account Credential in the table will display a modal form for editing the Credential.

Username	Publish Topics	Subscribe Topics	
lastcred	test/pub/#	test/pub/#	 
fake	apple/#	pear/#	 

Edit MQTT Credentials



The "Save" button in the bottom right can be clicked to update the Credential once all changes have been made.

LDAP

Chariot can be configured to use an LDAP server for MQTT client authentication and authorization instead of Chariot's MQTT Account Credentials.



This feature is available in Chariot v2.4.2 and newer

Review the [LDAP for MQTT Clients](#) for configuration details

The screenshot displays the 'MQTT Accounts' configuration interface. A green banner at the top indicates a 'Connected' status. The 'LDAP' tab is selected, showing the following configuration details:

- Source Type: LDAP
- Enabled:
- Host: localhost
- Port: 389
- Use TLS:
- System Username: cn=chariot
- System Password: (empty)
- Base DN: ou=credentials,dc=cirruslink,dc=com
- Username Attribute: uid
- Sub Topic Attribute Name: cls-subTopicFilter

Server Configuration

The Server Configuration page contains two tabs: [Configuration](#) and [Bridging](#).

Configuration

The configuration tab is a simple form used to configure the MQTT Server.

The screenshot displays the 'MQTT Server Configuration' interface. The 'CONFIGURATION' tab is selected, showing the following configuration details:

- Enable Non-secure:
- Non-secure Port: 1883
- Enable Secure:
- Secure Port: 8889
- Enable WebSocket:
- WebSocket Port: 8090
- Enable Secure WebSocket:
- Secure WebSocket Port: 8091
- Bind Address: 0.0.0.0
- Allow Anonymous:
- Anonymous MQTT Credentials: None Selected

The form contains the following fields:

- **Enable Non-secure**
 - Whether to enable non-secure client connections over plain TCP.

- **Non-secure Port**
 - The port that the MQTT Server will listen on for non-secure connections.
- **Enable Secure**
 - Whether to enable Secure client connections over SSL/TLS.
- **Secure Port**
 - The port that the MQTT Server will listen on for secure connections.
- **Enable WebSocket**
 - Whether to enable non-secure client connections over WebSockets.
- **WebSocket Port**
 - The port that the MQTT Server will listen on for non-secure WebSocket connections.
- **Enable Secure WebSocket**
 - Whether to enable secure client connections over WebSockets.
- **Secure WebSocket Port**
 - The port that the MQTT Server will listen on for secure WebSocket connections.
- **Bind Address**
 - The address that the MQTT Server will listen on for MQTT connections.
- **Allow Anonymous**
 - Whether to allow anonymous connections (no username and password).
- **Anonymous MQTT Credentials**
 - The MQTT Credentials to use for anonymous client's ACLs

Advanced Options >

- **Enforce Unique LWT Topics (added in release 2.7.0)**
 - Whether to enforce a connecting client's LWT topic is not currently registered with another client.
 - Any connection client with an LWT matching any of the "Unique Topic Filters" will have their connection attempt rejected if there is an existing active client with the same LWT topic connected to the server.
- **Unique LWT Topic Filters (add in release 2.7.0)**
 - Unique LWT topic filters to use.
 - Support the # wildcard ie. MyLWT/#

The "Update" button in the bottom right can be clicked to update the MQTT Server configuration once all changes have been made. An update will restart the MQTT server.

Bridging



Warning

Note MQTT bridging is not compatible with Sparkplug. Bridging should only be used for non-Sparkplug related MQTT messages.

A Bridge Client allows you to connect Chariot to another MQTT Server to share messages between the two servers. The shared messages can be restricted by topic filters to only share a subset of messages flowing through either of the servers.

A common usage is to connect edge MQTT brokers to a central or remote broker.

When enabled on Chariot, the Bridge Client becomes an MQTT client to the other MQTT Server. This client publishes messages, flowing through Chariot, to the other broker. It also subscribes on topics with the other broker to receive messages and deliver them to subscribed clients connected to Chariot.



Bridging is supported in release 2.3.0 forward

Chariot

MQTT Server Configuration

CONFIGURATION BRIDGING

ADD BRIDGE CLIENT

Name	Address	Port	Client ID	Enabled
b1	localhost	1883	bridgeClientID	

Records per page: 20 1-1 of 1

To edit an existing bridge client, select the client from the UI. To add a new bridge client click "Add Bridge Client".

Name	Address	Port	Client ID	Enabled
b1	localhost	1883	bridgeClientID	

Records per page: [Edit Bridge Client](#)

The form contains the following fields:

- Name** (text input): b1
- Enabled** (toggle switch): Off
- Username** (text input): test
- Password** (password input): [Empty]
- Address** (text input): localhost
- Port** (text input): 1883
- Use TLS** (toggle switch): Off
- Client ID** (text input): bridgeClientID
- Keep Alive** (text input): 30
- Clean Session** (toggle switch): On
- Allow Retained** (toggle switch): On
- Try Private** (toggle switch): On

A **SAVE** button is located at the bottom right of the form.

The form contains the following fields:

- **Name**
 - A unique name for the bridge connection.
- **Enabled**
 - Sets the enable state of the bridge connection.
- **Username**
 - Optional username for connecting to the server.
- **Address**
 - The server address.
- **Port**
 - The server port.
- **Use TLS**
 - Whether the bridge connection will be using SSL/TLS.



By default Chariot comes with an empty truststore file clientcerts.jks which overrides the JVM cacerts truststore. If TLS is enabled, the following lines will need to be removed from the <Chariot_install_directory>/conf/com.cirruslink.chariot.system config file:

- trustStoreFile="security/clientcerts.jks"
- trustStorePassword="secretpassword"

If the signer of the SSL/TLS cert installed on the remote MQTT Server is an external, commercial CA (e.g., DigiCert) the bridge client should successfully connect over TLS once you update the config file and restart Chariot.

If the signer of the SSL/TLS cert installed on the remote MQTT Server is an internal, non-commercial, you will need to add your Root CA cert to the JVMs cacerts trust store and restart Chariot.

- **Client ID**
 - The MQTT client ID to use for connecting to the server.
- **Keep Alive**
 - The MQTT keep alive time in seconds.
- **Clean Session**
 - Whether to connect with a clean session.
- **Allow Retained**
 - Whether the retained message flag is allowed on messages published to the server.
- **Try Private**
 - Whether the client should attempt to indicate to the server that it is a bridge client.
 - This feature helps to detect message loopback, but is only supported by some MQTT servers and may need to be disabled in order to connect.
- **Sub Topics**
 - The topics filters that will be subscribed on by the bridge client.
 - The topic filters are of the form <topic>:<qos> or just <topic>.
- **Pub Topics**
 - The topics filters that will be published on by the bridge client. These are used to restrict/filter which messages (flowing through Chariot) are published to the remote server,
 - The topic filters are of the form <topic>:<qos> or just <topic>.

The "Save" button in the bottom right can be clicked to update the MQTT Server Bridging configuration once all changes have been made

License

The License page allows for uploading, activating, and/or deactivating a Chariot License.



A detailed tutorial for licensing Chariot can be found here [Licensing Procedure](#).



If you have deployed Chariot through AWS Marketplace or Azure Marketplace, then no additional steps are required - your license is already installed and activated.

The screenshot shows the Chariot License page with the following sections:

- License Details:** Status: ACTIVE. License Key: ENF9Y-M3N56-ZXJYP-CEX7C-N8WUD. Activation Date: April 3, 2026 8:41 PM. Activation Period: 0. License Expiration: N/A. Support Expiration: January 1, 2027 6:00 AM. Buttons: REACTIVATE, DEACTIVATE.
- License Features:**

Feature	License Status
MQTT Server	●
IoT Snowflake	●
- Meter Details:** Running: x. Last Updated: 2026/04/03 09:23:52. Billed Hours: 35. Last Stopped: N/A. Billed Months: April. Status: N/A. Error: None. Toggle View: ON.

Upload License

A Chariot License is represented by a text file which contains a license key. The file can either be dragged/dropped into the "Upload License" form, or the license text can be pasted directly into the form.

Trial License Active ● 1 hours, 49 minutes, 42 seconds

License

The screenshot shows the License activation form with the following elements:

- Progress Bar:** Getting Started (checked), Activation Method (checked), Additional Details (checked), Activation (active), Complete (unchecked).
- Activation Step:** Drag and Drop or Manually Enter a License Key.
- License Key Input:** ENF9Y-M3N56-ZXJYP-CEX7C-N8WUD.
- Buttons:** BACK, NEXT.

✓ The **Activate** toggle (in the Upload License form) can be used to auto-activate the license immediately after it is uploaded. This is a convenience to prevent the need to click **Activate License** after uploading a license.

Online Activation

Online Activation allows the Chariot MQTT Server to remotely connect to the Chariot Licensing server for activation and deactivation of Chariot Licenses. When the "Online Activation" is selected, the License page appears with the following options:

License

Getting Started

Activation Method

Online Offline Floating

BACK NEXT

Additional Details

Activation

Complete

- **Activate License**
 - Activates an uploaded Chariot License by remotely accessing the Chariot Licensing Server.
- **Deactivate License**
 - Deactivates an uploaded & active Chariot License by remotely accessing the Chariot Licensing Server.

Offline Activation

Offline Activation requires the user to generate an activation request and provide it to Cirrus Link Solutions in order to obtain an Activation Code. The Activation Code can then be uploaded into the Chariot MQTT Server to activate the previously uploaded Chariot License. When the "Offline Activation" option is selected, the License page appears with the following options:

License

Getting Started

Activation Method

Online Offline Floating

BACK NEXT

Additional Details

Activation

Complete

- **Generate Activation Code**
 - Generates an Activation Request Code.
- **Deactivate and Generate Code**
 - Deactivates an active Chariot License and generates a Deactivation Request Code.
- **Upload Activation Code**
 - Uploads an Activation Code that was obtained using the Generate Activation Code.

Web Server

The Web Server page uses the following form to configure the web server settings:

Chariot

Web Server

Server Name

Enable HTTP

HTTP Port

Enable HTTPS

HTTPS Port


SAVE


The form contains the following fields:

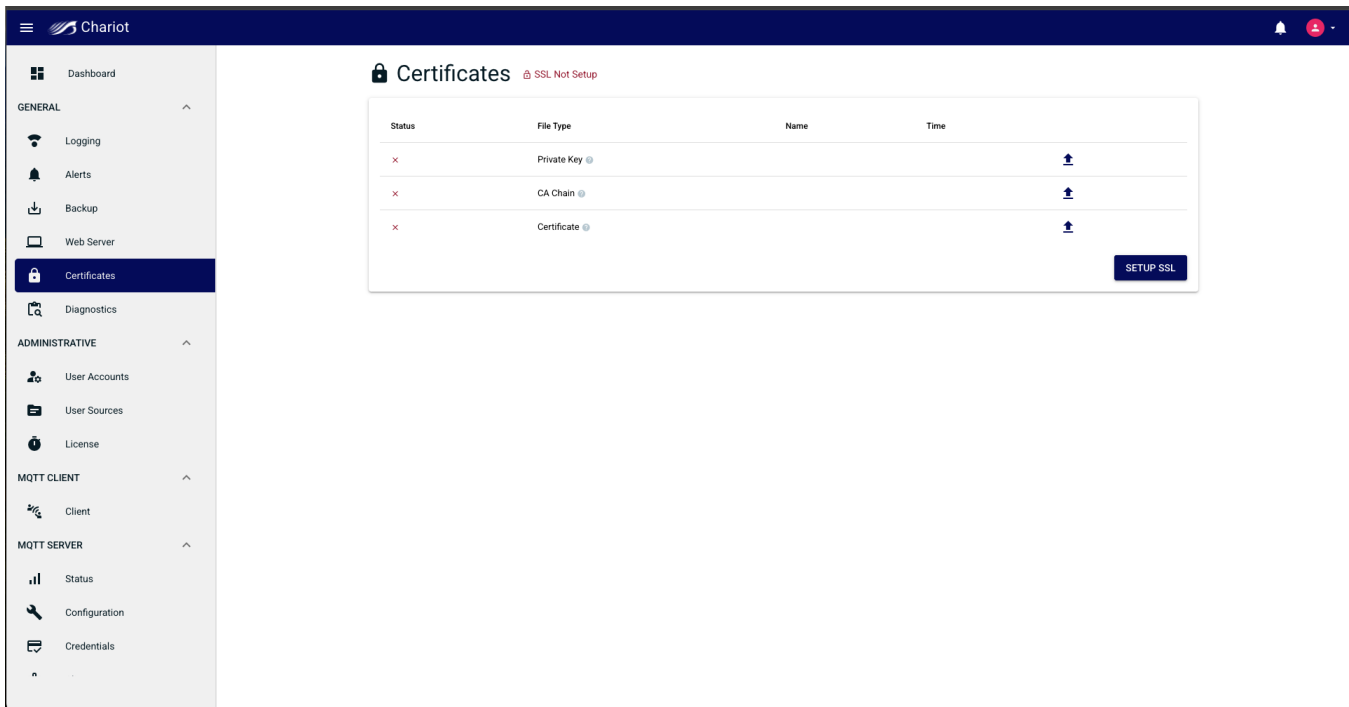
- **Server Name**
 - An optional name for this Chariot installation.
- **Enable HTTP**
 - Whether to enable non-secure HTTP connections to the Chariot Web UI.
- **HTTP Port**
 - The port that the Chariot MQTT Server will use for HTTP.
- **Enable HTTPS**
 - Whether to enable Secure HTTPS connections to the Chariot Web UI.
- **HTTPS Port**
 - The port that the Chariot MQTT Server will use for HTTPS.

Certificates

The certificates page provides the means to upload the necessary certificates and keys for setting up SSL/TLS.

 A detailed tutorial for setting up SSL/TLS can be found here: [Securing Chariot® MQTT Server](#).

 Self-signed certificates should not be used in a production environment on a public network.

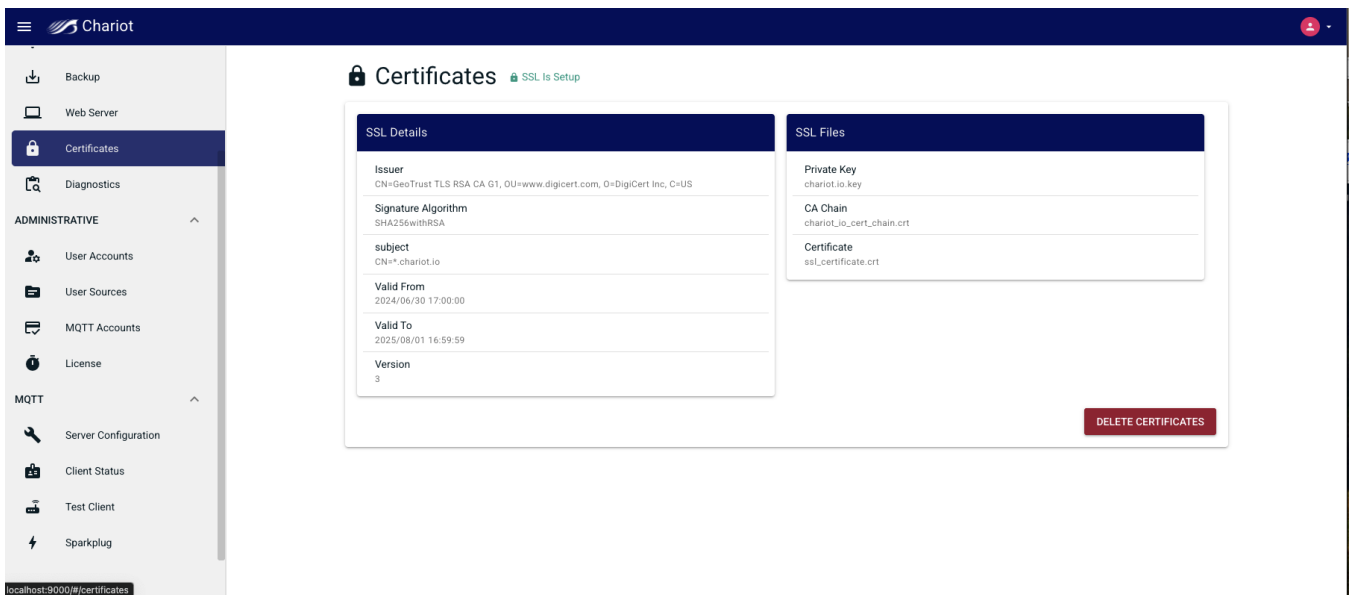


The Certificates page contains the following fields:

- **Private Key**
 - An RSA private key of type PKCS1 in PEM format that was used to generate the certificate signing request for the server Certificate
 - Review [How to identify my Private Key type](#) to verify the type
- **CA Chain**
 - An X.509 public root CA (Certificate Authority) certificate and any/all public intermediate CA certificates between the root and the CA that issued the certificate in PEM format. If there are no intermediate CAs, then the chain is made up of only the public root CA certificate.
- **Certificate**
 - An X.509 server certificate in PEM format.

Once the three files are uploaded, the "Setup SSL" button can be clicked to finish the setup.

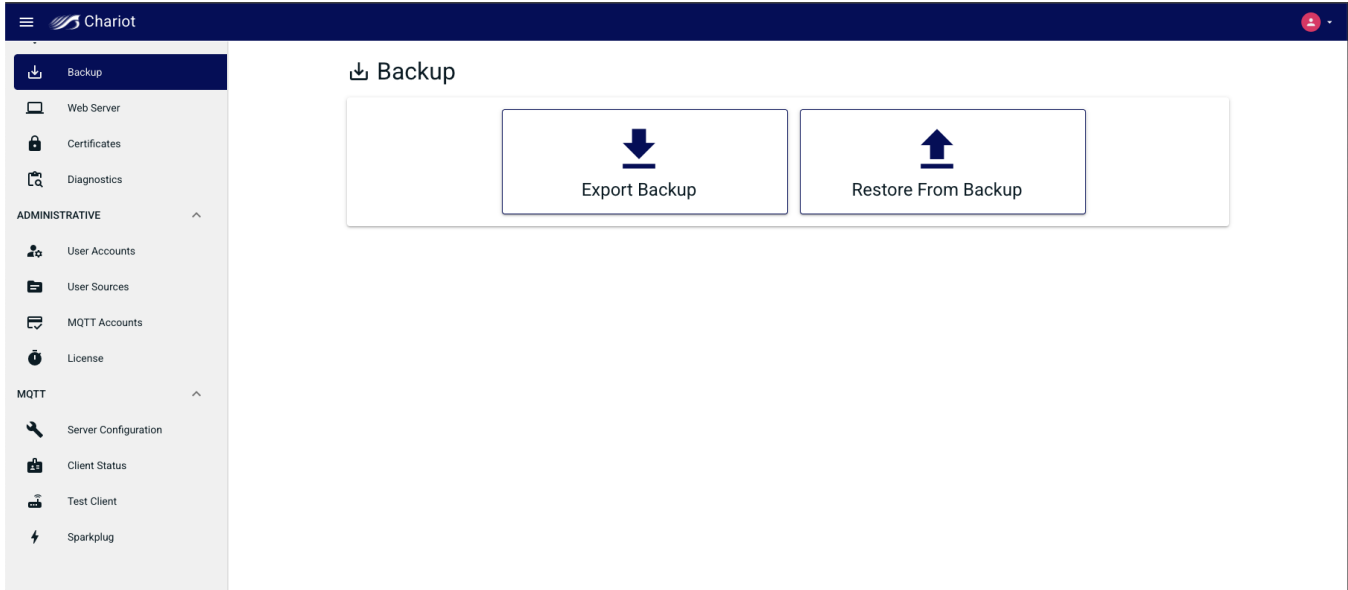
If successful, the certificate information will now be displayed on the Certificates page:



The "Delete Certificates" button can be used to remove the certificate and key from the Chariot MQTT Server and revert to a "SSL Not Setup" state.

Backup

This page allows for the backup and restore of Chariot configuration data.



- **Export Backup**
 - This button downloads a backup.zip file that contains the current configuration data of the Chariot instance
- **Restore From Backup**
 - This button restore configuration data from backup.zip file.



The backup does not include any SSL certificates that may have been loaded. If configured for SSL, make sure to also copy the ssl.jks keystore from <OLD_chariot_install_folder>/security directory and save the ssl.jks keystore and exported backup.tgz file in a safe, known location