Cloud Command Center (CCC) API
Admin User Guide

Introduction

The Cloud Command Center (CCC) API is a ReSTful web services API providing centralized provisioning and administrative services with the Secure Messaging Platform.

This document provides a general overview of the CCC API and how to get started using it under some common scenarios. The details of the resources exposed by the API are described in the live documentation generated automatically by the system. The CCC API is deployed on Microsoft's Windows Azure platform, and can be accessed directly via third-party API integrations or via the provided Cloud Command Center user interface: https://ccc.secure-messaging.com.

User access model

The CCC allows for very flexible configurations by supporting a Partner / Reseller / Customer model. The system allows setting up Partner accounts, which in turn can create their own Reseller accounts. Then, under each Reseller, multiple Customer accounts (Portals) can be created.

The Customer accounts can be deployed into multiple datacenters available throughout the world (cloud based) or even in private on-premise deployments. No matter where the actual Customer account is created, all the management operations are carried out from this centralized CCC API.

To better manage these properties, administrative users can be setup to manage a specific partner, reseller, or customer. Some operations in the API are available only to a specific type of users. Complementing this hierarchy, the system supports a list of roles that can be assigned to each user. The role grants the user the capability to execute specific commands. This model allows for very flexible configurations where a partner-level user can be provisioned to have access to all accounts under a given Partner, but only be assigned reporting features without access to provisioning features, for example.

API Reference

API endpoints

There are 2 public endpoints that can be consumed from the client. The first one is a staging environment used for development purposes, and the other is the production environment for live data. The staging environment has a more regular deployment schedule than production and even though efforts are made to keep it as stable as possible, we cannot guarantee that will always be on line and backward compatible.

The URL's for the 2 environments are:

- Staging: https://staging-api.secure-messaging.com/api
- Production: https://api.secure-messaging.com/api

For more information on specific features and resources made available through the API, consult the live documentation for each environment. These links contain the most up to date information about each API call.

JSON documentation

- Staging: https://staging-api.secure-messaging.com/documentation/index.html
- Production: https://api.secure-messaging.com/documentation/index.html
Metadata documentation

- Staging: https://staging-api.secure-messaging.com/api/metadata
- Production: https://api.secure-messaging.com/api/metadata

Authentication

All operations against the API need to be authenticated, except for some specific operations, like the Login action. Two types of authentication are supported: Credentials and Basic.

Credentials authentication

`POST /auth/credentials?userName={username}&password={password}&rememberMe=true`

To authenticate using credentials authentication a POST action is made to the above URL. Once the client is authenticated an ss-id & ss-pid cookie will be set. These cookies need to be included in all subsequent API requests.

Basic authentication

Authorization: Basic dGVzdEBlbWFpbDIuY29tOnRlc3QxMjM=

Note: The key provided above is only an example.

Basic authentication can be used by providing the “username:password” combination encoded as base64. Add an Authorization header to the first request made to the API. If the client is authenticated an ss-id & ss-pid cookie is set which should be included in all subsequent requests.

More information on the Basic access authentication protocol can be found here: http://en.wikipedia.org/wiki/Basic_access_authentication

Getting Started – Microsoft’s .NET implementations

The easiest way to get started programming against the CCC API is to download the Service model NuGet package. The package includes strongly typed classes for all API calls as well as a client to handle the calls and error messages to and from the API. The package can be found on a private hosted NuGet repository. To get access to this private repository, register an account with www.myget.org and send us an email (support@secure-messaging.com) requesting access to the private package repository.

Configure NuGet to look for packages from the package feed.

Run the `nuget sources add` command to add the package feed. Details about the command can be found at NuGet’s command line reference documentation (scroll all the way down to the Sources Command section).

The command line to run is:

```
nuget sources add -Name [name] -Source https://www.myget.org/F/email2partners -UserName [username] -Password [password]
```

<table>
<thead>
<tr>
<th>[name]</th>
<th>The name you’d like to appear in NuGet Package manager for the feed. i.e. Secure Messaging</th>
</tr>
</thead>
<tbody>
<tr>
<td>[username]</td>
<td>The username/email address you use to log into Myget.</td>
</tr>
<tr>
<td>[password]</td>
<td>The password you use to sign in to Myget.</td>
</tr>
</tbody>
</table>

Note: Visual Studio may need to be restarted after running the above command.

Alternatively you can follow the instructions at the following link to add the package source using the Visual Studio dialog. http://docs.nuget.org/docs/start-here/managing-nuget-packages-using-the-dialog
Once NuGet has been configured with the package feed you will be able to view the Secure Messaging packages. Download the SecMsg.API.ServiceModel package from the NuGet feed.

**Code examples**

A sample application that use the ServiceModel package can be found on GitHub at [https://github.com/Email2/SecMsg.API.ClientDemo](https://github.com/Email2/SecMsg.API.ClientDemo)

This application shows how to create and configure an API client and how to make calls to and process responses from the API using C# code.

**Credentials**

Before the ServiceModel client will be allowed to communicate with the CCC API, the client need to authenticate against the API. The C# client will automatically log you in when the Username and Password is set when the client is created (as can be seen in the sample application).