Set Up External Storage Location for Attachment Files

Last Modified on 06/09/2023 5:09 am EDT

V9.2

Overview

Files can be attached to emails or uploaded during form submission. By default, uploaded files are stored as binary records in the Cora SeQuence database. To reduce database size and improve database performance, you can set up a different storage location. The new file location does not impact Flowtime user experience and is transparent to end users.

You can set up the file storage location on-premises or on a cloud provider. Currently, Cora SeQuence supports the following storage locations out-of-the-box: Microsoft Azure Blob Storage, Azure File Storage, FTP, SFTP, and SQL Server.

After you set up the new storage location, any attachment files uploaded to the system are stored in the new location. Existing files remain in the previous storage location.
For details on how to migrate existing files, see this article.

Set up a new default file storage location

You can run PowerShell functions to set the connection to the new file storage location. This is a one-time procedure that distributes the configuration to all the applications deployed on the server, except the Active Directory Synchronization Service (ADSS) application.

Prerequisites

Before you set up the new storage location:

- Set up the new file storage location on-premises or on the cloud, and configure it according to your organization needs.
- Make sure that the Cora SeQuence user has permissions to access the external file storage.

> **IMPORTANT**
> Make sure that the configured location complies with your organization's security requirements for the types of data that can be stored in the new location.
> For more details, see the section on security below.

Procedure

Run the following PowerShell functions on all the servers that host Cora SeQuence applications:

1. To configure the connection to the new file storage location, run this PowerShell function: Add-CoraSeQuenceFilesStorageConfiguration.
2. To set the connection as the default file storage location, run this function: Set-CoraSeQuenceDefaultFilesStorageConfiguration.

All the email and web form attachments uploaded after the change are stored in the new file location. The files' metadata and relative path are stored in the new `tblfiles` table in the Cora SeQuence database.

IMPORTANT

- You can have only one default connection string.
- The system does not validate the connection string. If you enter invalid values, file upload and the Email Listener activity will fail.
- Config files are overwritten when you upgrade to a later version of Cora SeQuence. To avoid losing files that have been uploaded to previous storage locations, after system upgrade, you need to reconfigure all the connection records (including history connection records) in the config files.

For a configuration example, see this article.

Reset the default file connection to the Cora SeQuence database

You can run the Set-CoraSeQuenceDefaultFilesStorageConfiguration function to restore the configuration to the Cora SeQuence SQL database.

1. Run the Set-CoraSeQuenceDefaultFilesStorageConfiguration function, and enter one of the following values for the **-ConfigurationName** parameter: `"Default"` or `"CoraSeQuence.Default"`.

Example:

```
PS C:\>Set-CoraSeQuenceDefaultFilesStorageConfiguration -ConfigurationName "Default"
```

Set up an expression to access the details of an attachment

You can use an expression to access the properties of an attachment, regardless of where it is stored.

Syntax

```
{Form1}.Query("UACT1").Include("AttachmentField")
```

Where:
`AttachmentField` is the name of the attachment field in the query.

Example:

```
{Form1}.Query("UACT1").Include("AttachmentField").Field("AttachmentField").Content
```

Security features and best practices for Azure storage services

Azure storage services provide the following security features:

- Data at rest is encrypted with Storage Service Encryption (SSE).
- Data in transit is encrypted and it is enabled only via HTTPS.

NOTE
Azure storage doesn't natively support HTTPS with custom domains. Associate the custom domain with an Azure CDN endpoint to be able to access blobs using the HTTPS protocol.

- Azure role based access control (RBAC) is used to secure access for managing storage accounts.
- By default, access to blobs does not allow anonymous access.

Security best practices

- Use Managed Service Identities to allow applications to access keys, eliminating the need to add them directly into code or configuration files.
- Generate Shared Access Signature (SAS) when accessing storage through applications.
- Change your storage account custom keys regularly. Add this to your regular credential rotation process (or use Microsoft-managed keys which are being rotated automatically).
    - Keep in mind that any SAS generated with the rotated key will no longer work, so you'll need to generate new SAS for those instances.

> **NOTE**
> For Azure File Storage and Azure Blob Storage, make sure to encrypt the AccountKey attribute in the Cora SeQuence toolkit before passing in the Connection parameter in the Set-CoraSeQuenceDefaultFilesStorageConfiguration function.

- Grant access to storage account administration through RBAC only to the people who specifically need access.
- Always use HTTPS to encrypt data between the client and the Azure storage service.

Want to learn more security requirements and configuration?

Refer to the following Microsoft articles:

- Azure Storage Account Security
- Azure Storage security guide
- Manage anonymous read access to containers and blobs
- Authorize access to Azure blobs and queues using Azure Active Directory

---

Check out these demos on how to set up an external location on Microsoft Azure and migrate existing files to the new location.

- Part 1: Set up the external location
- Part 2: Store files in the external location
- Part 3: Migrate existing files to the external storage location

---