File Connector
Quick Start

Table of Contents

File Connector

The Solace File connector provides the ability to replicate from source to sink or stream file as events using PubSub+ event brokers and the Solace Event Mesh. The File Connector is based on the new common architecture of the Solace Framework Connectors

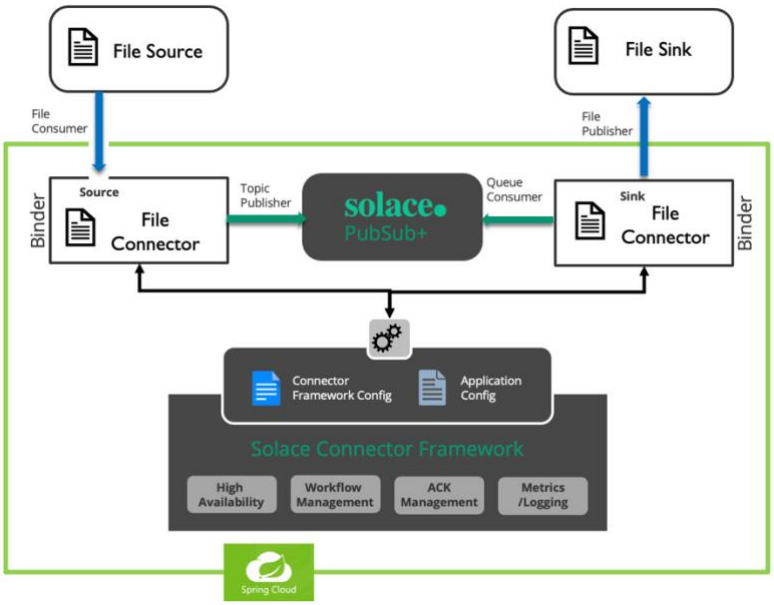
Functionality

Following are some of the high level features of File Connector. Detailed configuration and descriptions can be found in User Guide document.

1. Managed File Transfer of Static, Dynamic and Directories from Source to Sink
2. Stream file as events.
3. Supports local file transfer and remote file transfer via SFTP, FTP and FTPs protocols
4. Checkpoints the state of replication, so that connector resumes from last successful multipart
5. Provides the ability to send state events to Command Control Centre to monitor the state of replication
6. Provides the ability to run the connector at fixed intervals and to exit at specific time
7. Provides the ability to control the amount of bandwidth used by connector to replicate files

Architecture

File Connector is based on the new common architecture of the Solace Framework Connectors



Prerequisites

The following are the prerequisites for the Source and Sink Connectors

Source Connector – Prerequisites

1. Application server to deploy the source connector.
2. Source files location either on the same application server or a remote sftp server.
3. Solace PubSub+ broker instance.

Sink Connector – Prerequisites

1. Application server to deploy the sink connector.
2. Destination location for the files on the same application server.
3. Solace PubSub+ broker instance.

Environment

Latest version of Solace PubSub+ broker.

Java – Open JDK 18 or later

Quick Start Steps

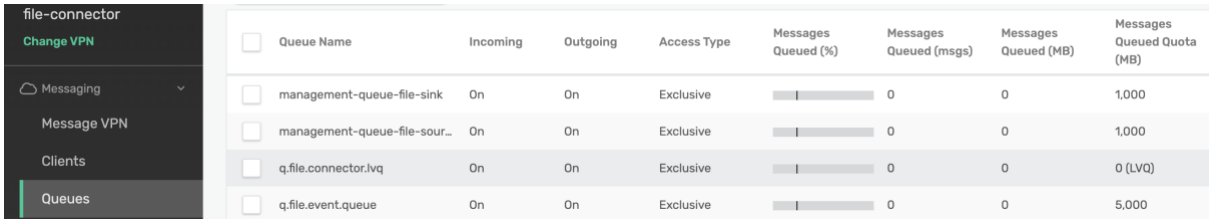
Step 1 – Get Solace PubSub+ Credentials

1. Create a new VPN on the Solace PubSub+ instance or a service if using Solace Cloud.
2. Create User Credentials with the necessary ACL's inside the VPN, separate user credentials for the source and sink connectors, preferably.

Step 2 – Create required queues on Solace PubSub+

Inside the VPN/service, create the following queues

Queue Name	Queue Type	Access Type	Required
management-queue-file-sink	Regular	Exclusive	Y
management-queue-file-source	Regular	Exclusive	Y
q.file.connector.lvq	LVQ, Spool size 0 MB	Exclusive	Y
q.file.event.queue	Regular	Exclusive	Y



Step 3 – Add topic subscriptions to the queues

Please note that you can define your own base topic subscriptions in the connector configuration and the queue subscriptions.

The <basic-topic> is the destination topic inside your Source connector configuration

Checkpoint subscriptions are internal to the functioning of the connectors and are depicted as <base-topic>/checkpoint

Add the following topic subscriptions to the queues.

For MFT file types – Static and Dynamic

Queue Name	Topic Subscriptions
q.file.connector.lvq	solace/fc/source/test/checkpoint solace/fc/source/test
q.file.event.queue	solace/fc/source/test

For MFT file types – Directory and Recursive Directory options

Queue Name	Topic Subscriptions
q.file.connector.lvq	solace/fc/source/test/checkpoint
q.file.event.queue	solace/fc/source/test

For File to Events – All File Types without dynamic topic

Queue Name	Topic Subscriptions
q.file.connector.lvq	solace/fc/source/test/checkpoint solace/fc/source/test
q.file.event.queue	solace/fc/source/test

For File to Events – CSV file Types with dynamic topic

Queue Name	Topic Subscriptions
q.file.connector.lvq	solace/fc/source/test/checkpoint solace/fc/source/dynamic/*/*
q.file.event.queue	solace/fc/source/dynamic/*/*

Step 4 – Deploy the Source connector

1. Deploy Connector Package

   Copy and unzip the **pubsubplus-connector-file-2.1.1.zip** package on the source application system.

2. Copy Configuration File

Use the application-source.yml configuration template inside the directory path **pubsubplus-connector-file-2.1.1/samples/config/source/application-source.yml**

For quick setup us, use the **application-source-sample.yml** provided with this Quick Start document.

Create a config/source/ directory path.

Copy and rename the configuration sample as application.yml in a config/source/ directory. Use the same configuration path while running the connector.

3. Add Solace Endpoint and Credentials

Edit the sample configuration file and add/update the Solace connection and credentials

a. Solace Credentials – which includes the host url, vpn, user and password on the configuration level **solace.java**
b. Solace Management Credentials – which includes the host url, vpn, user and password on the configuration level **solace.connector.management.session**
c. Solace Management Queue – on the configuration level **solace.connector.management.queue**

```
solace:
  connector:
    error:
      handle: stop_all
    workflows:
      0:
        enabled: true
        acknowledgment:
          publish-async: true
    management:
      leader-election:
        mode: active_standby  # The connector?~@~Ys leader election mode. (values: standalone, active_active, active_standby)
        fail-over:
          max-attempts: 3  # The maximum number of attempts to perform a fail-over.
          back-off-initial-interval: 1000  # The initial interval (milliseconds) to back-off when retrying a fail-over.
          back-off-max-interval: 10000  # The maximum interval (milliseconds) to back-off when retrying a fail-over.
          back-off-multiplier: 2.0  # The multiplier to apply to the back-off interval between each retry of a fail-over.
      queue: management-queue-file-source  # The management queue name.
      session: # The management session. This has the same interface as that used by `solace.java.*`. For more info: https://github.com/SolaceProducts/solace-spring-bo
ot/tree/master/solace-spring-boot-starters/solace-java-spring-boot-starter#updating-your-application-properties
        host: tcp://SOLACE_IP:55555
        msg-vpn: file-connector
        client-username: default
        client-password: default
  java:
    host: tcp://SOLACE_IP:55555
    msg-vpn: file-connector
    client-username: default
    client-password: default
    connectRetries: -1
    reconnectRetries: -1
```

4. Configure the transfer protocol for source connector

The source can be either local file system or a remote server (SFTP/FTP/FTPs). Only supported for Managed file transfer and not the file to event streaming.

a. For Local file transfer, no special configuration required. Keep the SFTP and FTP settings to disabled

b. For configuring SFTP protocol enable the file.source.sftp_settings block and configure the credentials. SFTP login is supported with either user and password combination or the user and private key combination

Login with user and password

```
sftp_settings:
  enabled: 1
  ip: 127.0.0.1
  port: 22
  user: sftpuser
  password: sftpuser
  strictHostKeyChecking: no
  #privateKeyPath: /home/centos/.ssh/id_rsa
```

Login with user and private key

```
sftp_settings:
  enabled: 1
  ip: 127.0.0.1
  port: 22
  user: sftpuser
  #password: sftpuser
  strictHostKeyChecking: no
  privateKeyPath: /home/centos/.ssh/id_rsa
```

c. For configuring FTP protocol enable the file.source.ftp_settings block and configure the credentials.

```
ftp_settings:
  enabled: 1
  ip: 127.0.0.1
  port: 21
  user: ftpuser
  password: ftpuser
  secured: false
```

d. For configuring FTPs protocol enable the file.source.ftp_settings block and configure the credentials. Also set the value for file.source.ftp_settings.secured to true

```
ftp_settings:
  enabled: 1
  ip: 127.0.0.1
  port: 21
  user: ftpuser
  password: ftpuser
  secured: true
```

5. Configure the File Type value

Define the required **File Type** for Managed File Transfer or the File to Events

a. Configuration level – **file.source.general.file_type**
b. File Types – 1 (Static Files), 2 (Dynamic File), 3 (Directory – First level) and 4 (Directory Recursive – Sub directories included)

```
file:
  source:
    scheduler:
      restart_time_sec: 0
    general:
      adapter_id: SOURCE_CONN_ID1
      file_type: 3
```

6. Configure the Source location

Set the source file/directory location with respect to the file_type value configured

For Static Files

Create a new file static-files.cfg in the samples/config directory and add all the individual file paths that need to be transferred in the below format

ABSOLUTE_FILE_PATH_SOURCE|ABSOLUTE_FILE_PATH_DESTINATION

ABSOLUTE_FILE_PATH_SOURCE = Absolute file path of the file that is required to be streamed on the source system
ABSOLUTE_FILE_PATH_DESTINATION = Absolute file path where the file needs to be created on the destination system.

For example, the entries in the static-files.cfg will appear as

/absolute/file/path/source/file1.txt|/absolute/file/path/destination/file1.txt
/absolute/file/path/source/file2.txt|/absolute/file/path/destination/file2.txt

/absolute/file/path/source/file3.txt|/absolute/file/path/destination/file3.txt

Once static-files.cfg is created, add its absolute file path in the application.yml at configuration level **spring.cloud.stream.bindings.input-0.destination** as shown below

```
spring:
  application:
    name: File Connector(Source)
  cloud:
    stream:
      bindings:
        input-0:
          destination: /home/centos/pubsubplus-connector-file-2.0.0/samples/config/static-files.cfg
          binder: file
```

For Dynamic File

Since dynamic file needs to be continuously watched for changes, only a single file is processed by each workflow and it can be directly configured at configuration level **spring.cloud.stream.bindings.input-0.destination** as shown below

```
spring:
  application:
    name: File Connector(Source)
  cloud:
    stream:
      bindings:
        input-0:
          destination: /home/centos/pubsubplus-connector-file-2.0.0/data/files/dynamic-file.txt
          binder: file
```

For Directory and Directory Recursive

Similar to how the static files are configured, multiple directory locations can be configured in the similar way.

Create a new file directory-paths.cfg in the samples/config directory and add all the individual diretory paths that need to be transferred in the below format

ABSOLUTE_DIR_PATH_SOURCE|ABSOLUTE_DIR_PATH_DESTINATION

ABSOLUTE_DIR_PATH_SOURCE = Absolute path of the directory that is required to be streamed on the source system
ABSOLUTE_DIR_PATH_DESTINATION = Absolute path where the directory needs to be created on the destination system.
For example, the entries in the directory-paths.cfg will appear as

/absolute/dir/path/source/dir1|/absolute/dir/path/destination/dir1
/absolute/dir/path/source/dir2|/absolute/dir/path/destination/dir2

/absolute/dir/path/source/dir3|/absolute/dir/path/destination/dir3

Once directory-paths.cfg is created, add its absolute file path in the application.yml at configuration level **spring.cloud.stream.bindings.input-0.destination** as shown below

```
spring:
  application:
    name: File Connector(Source)
  cloud:
    stream:
      bindings:
        input-0:
          destination: /home/centos/pubsubplus-connector-file-2.0.0/samples/config/directory-paths.cfg
          binder: file
```

7. Configure the Solace Destination Topics

Configure the Solace destination topic <base-topic> in the application.yml file at below configuration levels

a. **spring.cloud.stream.bindings.output-0.destination**
b. **file.source.solace_out.destination**

```
spring:
  application:
    name: File Connector(Source)
  cloud:
    stream:
      bindings:
        input-0:
          destination: /home/centos/pubsubplus-connector-file-2.0.0/samples/config/static-files.cfg
          binder: file
        output-0:
          destination: solace/fc/source/test
          binder: solace


file:
  source:
    solace_out:
      lvq: q.file.connector.lvq
      destination: solace/fc/source/test
```

8. Configure the File to Event Streaming (Optional - if not Managed file transfer)

This part can be ignored if the requirement is to transfer the files.

For reading the files and streaming the file as events following ways are supported

a. Reading CSV files (Configurable column delimiter. Column delimiter will also be referred to as param delimiter in this document and configuration file). These are

the regular csv files with a line separating each event. Set the **file_to_event.fileFormat** to 1

```
file_to_event:
  enabled: 1
  fileFormat: 1
  #eventDelimiter: '|'
  paramDelimiter: ','
  dynamicTopic: solace/fc/source/test/{param-3}/{param-4}/{param-5}/{event_no}/{param-4}
  paramHeaderMap: event_no,name,city,province,country
  #jsonPath: '$.Authors[*].Books[*]'
  #xPath: '/class/student'
```

The configuration paramDelimeter and dynamicTopic can be used to publish events on a dynamic topic. Each column value can be configured in the topic structure as {param-<ColumnNumber>}

If paramHeaderMap is configured, the records will be mapped to the headers provided and the payload will be published as a json. If no paramHeaderMap is configured, the payload will be published as binary data as read from the file.

Also, If paramHeaderMap is configured, the dynamic variables in the dynamicTopic can also be configured with the column name. For example, {city} or {<ColumnName>}

b. Reading CSV files with custom event delimiter(Configurable column delimiter. Column delimiter will also be referred to as param delimiter in this document and configuration file). Set the **file_to_event.fileFormat** to 2. These files don't have a line separating each event but instead the events are separated by custom Delimiter.

For example, data in the file can be

E1,John Doe,NYC,New York,US|E2,Jane Doe,Toronto,Ontario,Canada

```
file_to_event:
  enabled: 1
  fileFormat: 2
  eventDelimiter: '|'
  paramDelimiter: ','
  dynamicTopic: solace/fc/source/test/{param-3}/{param-4}/{param-5}/{event_no}/{param-4}
  paramHeaderMap: event_no,name,city,province,country
  #jsonPath: '$.Authors[*].Books[*]'
  #xPath: '/class/student'
```

The configuration of paramDelimiter, paramHeaderMap and dynamicTopic is done in the same way as mentioned in case a.

c. Reading JSON files. Set the **file_to_event.fileFormat** to 3

```
file_to_event:
  enabled: 1
  fileFormat: 3
  jsonPath: '$.Authors[*].Books[*]'
  #eventDelimiter: '|'
  #paramDelimiter: ','
  #dynamicTopic: solace/fc/source/test/{param-3}/{param-4}/{param-5}/{event_no}/{param-4}
  #paramHeaderMap: event_no,name,city,province,country
  #xPath: '/class/student'
```

d. Reading XML files. Set the **file_to_event.fileFormat** to 4

```
file_to_event:
  enabled: 1
  fileFormat: 4
  xPath: '/class/student'
  #eventDelimiter: '|'
  #paramDelimiter: ','
  #dynamicTopic: solace/fc/source/test/{param-3}/{param-4}/{param-5}/{event_no}/{param-4}
  #paramHeaderMap: event_no,name,city,province,country
  #jsonPath: '$.Authors[*].Books[*]'
```

Please note that dynamic topic is not yet supported for JSON and XML files, this will be supported in the future release.

9. Configure the Directory Wildcard – Blacklist/Whitelist files (Add on feature)

This is an add on feature supported by the File Connectors. Users can either whitelist or blacklist files based on the configured regular expressions. This feature is only applicable if the source location file type is either First level directory transfer or recursive directory i.e. The **file.source.general.file_type** value is either 3 (First Level Directory) or 4 (Recursive Directory). Refer to Step 6 to configure the correct file_type value

There are 3 values for wildcard_type configuration

a. No Wildcard
b. Whitelist – Only allow certain files which matches the regular expressions. Block all other files
c. Blacklist – Only block certain files which matches the regular expressions. Allow all other files

Configuring the wildcard

1. **Default Value 0** – No Blacklist or Whitelist required. All files will be processed normally

```
file:
  source:
    directory_wildcard:
      wildcard_type: 0
```

2. **Value 1** – Whitelist the file names based on the regular expressions. To configure regular expression create a separate file which will contain all the required regular expressions. For example create file name dir_regexps.conf
   The content of this file will have regular expressions on each line as below

```
[centos@ip-172-31-19-31 config]$ cat dir_regexps.conf
^.*\/EXPORT_FILE_.*\.csv$
^.*\/LOG_FILE_.*\.csv$
```

   We can add more regular expression with each new line. A sample config file can be found with this Quick Start guide package for your reference.

Now, the next step is to configure the wildcard_type=1 and wildcard config_path in the main application.yml configuration file as below

```
file:
  source:
    directory_wildcard:
      wildcard_type: 1
      config_path: /home/centos/pubsubplus-connector-file-2.1.1/samples/config/dir_regexps.conf
```

Once configured, the Source File Connector will only process files in the directory which matches the configured regular expressions. The files which do not match any of the regular expressions will be blocked and will not be processed.

3. **Value 2** – Blacklist the file names based on the regular expressions. To configure regular expression create a separate file which will contain all the required regular expressions. For example create file name dir_regexps.conf
   The content of this file will have regular expressions on each line as below

```
[centos@ip-172-31-19-31 config]$ cat dir_regexps.conf
^.*\/EXPORT_FILE_.*\.csv$
^.*\/LOG_FILE_.*\.csv$
```

We can add more regular expression with each new line. A sample config file can be found with this Quick Start guide package for your reference.

Now, the next step is to configure the wildcard_type=2 and wildcard config_path in the main application.yml configuration file as below

```
file:
  source:
    directory_wildcard:
      wildcard_type: 2
      config_path: /home/centos/pubsubplus-connector-file-2.1.1/samples/config/dir_regexps.conf
```

Once configured, the Source File Connector will block all the files in the directory which matches the configured regular expressions. Only the files which do not match any of the regular expressions will be processed as normal.

Step 5 – Deploy the Sink connector

1. Deploy Connector Package

Copy and unzip the **pubsubplus-connector-file-2.1.1.zip** package on the destination application system.

2. Copy Configuration File

Use the application-sink.yml configuration template inside the directory path **pubsubplus-connector-file-2.1.1/samples/config/sink/application-sink.yml**

For quick setup us, use the **application-sink-sample.yml** provided with this Quick Start document.

Create a config/sink/ directory path.

Copy and rename the configuration sample as application.yml in the config/sink/ directory. Use the same configuration path while running the connector.

3. Add Solace Endpoint and Credentials

Edit the sample configuration file and add/update the Solace connection and credentials

d. Solace Credentials – which includes the host url, vpn, user and password on the configuration level **solace.java**
e. Solace Management Credentials – which includes the host url, vpn, user and password on the configuration level **solace.connector.management.session**
f. Solace Management Queue – on the configuration level **solace.connector.management.queue**

```
solace:
  connector:
    error:
      handle: stop_all
    workflows:
      0:
        enabled: true
        acknowledgment:
          publish-async: true
    management:
      leader-election:
        mode: active_standby  # The connector?~@~Ys leader election mode. (values: standalone, active_active, active_standby)
        fail-over:
          max-attempts: 3  # The maximum number of attempts to perform a fail-over.
          back-off-initial-interval: 1000  # The initial interval (milliseconds) to back-off when retrying a fail-over.
          back-off-max-interval: 10000  # The maximum interval (milliseconds) to back-off when retrying a fail-over.
          back-off-multiplier: 2.0  # The multiplier to apply to the back-off interval between each retry of a fail-over.
      queue: management-queue-file-source  # The management queue name.
      session: # The management session. This has the same interface as that used by `solace.java.*`. For more info: https://github.com/SolaceProducts/solace-spring-bo
ot/tree/master/solace-spring-boot-starters/solace-java-spring-boot-starter#updating-your-application-properties
        host: tcp://SOLACE_IP:55555
        msg-vpn: file-connector
        client-username: default
        client-password: default
  java:
    host: tcp://SOLACE_IP:55555
    msg-vpn: file-connector
    client-username: default
    client-password: default
    connectRetries: -1
    reconnectRetries: -1
```

4. Configure the Solace Queue Consumer endpoint

a. Configure the Solace Queue from where the Sink connector will consume the file events/multiparts at the configuration level **spring.cloud.stream.bindings.input-0.destination**

```
spring:
  application:
    name: File Connector(Sink)
  cloud:
    stream:
      bindings:
        input-0:
          destination: q.file.event.queue
          binder: solace
        output-0:
          destination: ./data/output
          binder: file
```

5. Configure the transfer protocol for sink connector

The sink can be either local file system or a remote server (SFTP/FTP/FTPs)

a. For Local file transfer, no special configuration required. Keep the SFTP and FTP settings to disabled
b. For configuring SFTP protocol enable the file.sink.sftp_settings block and configure the credentials. SFTP login is supported with either user and password combination or the user and private key combination

Login with user and password

```
sftp_settings:
  enabled: 1
  ip: 127.0.0.1
  port: 22
  user: sftpuser
  password: sftpuser
  strictHostKeyChecking: no
  #privateKeyPath: /home/centos/.ssh/id_rsa
```

Login with user and private key

```
sftp_settings:
  enabled: 1
  ip: 127.0.0.1
  port: 22
  user: sftpuser
  #password: sftpuser
  strictHostKeyChecking: no
  privateKeyPath: /home/centos/.ssh/id_rsa
```

c. For configuring FTP protocol enable the file.sink.ftp_settings block and configure the credentials.

```
ftp_settings:
  enabled: 1
  ip: 127.0.0.1
  port: 21
  user: ftpuser
  password: ftpuser
  secured: false
```

d. For configuring FTPs protocol enable the file.sink.ftp_settings block and configure the credentials. Also set the value for file.sink.ftp_settings.secured to true

```
ftp_settings:
  enabled: 1
  ip: 127.0.0.1
  port: 21
  user: ftpuser
  password: ftpuser
  secured: true
```

Step 6 – Run the Sink connector

On the destination application system, run the sink connector using the below command

```
sh bin/start.sh --config ./config/sink/
```

Step 7 – Run the Source connector

On the source application system, run the source connector using the below command

```
sh bin/start.sh --config ./config/source/
```

Step 8 – Validate the transferred files

On the destination application system, validate if the files or directories have been transferred successfully once the Source and Sink connectors have completed processing.

Test Cases

Functional

Replicating Static Files

Test case 1: Replicating Static Files (File Type: 1)

Configure a workflow to read data from a file location and publish to a topic on Solace. Input Channel destination will contain the path of a file which contains a list of static files to be replicated.

Sample configuration to simulate the test case:

```
spring:
  application:
    name: File Connector(Source)
  cloud:
    stream:
      bindings:
        input-0:
          destination: /home/centos/pubsubplus-connector-file-2.0.0/samples/config/static-files.cfg
          binder: file
```

Replicating Dynamic Files

Test case 2: Replicating Dynamic Files (File Type: 2)

Configure a workflow to read data from a file location and publish to a topic on Solace. Input Channel destination will contain the path of a file which contains a single path location to the dynamic file. Example of what a dynamic file can be is a log file, file that is getting continuously appended.

```
spring:
  application:
    name: File Connector(Source)
  cloud:
    stream:
      bindings:
        input-0:
          destination: /home/centos/pubsubplus-connector-file-2.0.0/data/files/dynamic-file.txt
          binder: file
```

Replicating Directories

Test case 3: Replicating First level files in a Directory. Files in the Sub directories will be excluded (File Type: 3)

Configure a workflow to read data from a file location and publish to a topic on Solace. Input Channel destination will contain the path of a file which contains a list of paths for the directories that need to be replicated.

```
spring:
  application:
    name: File Connector(Source)
  cloud:
    stream:
      bindings:
        input-0:
          destination: /home/centos/pubsubplus-connector-file-2.0.0/samples/config/directory-paths.cfg
          binder: file
```

Replicating Directories – Sub directories included

Test case 4: Replicating a complete Directory. All files and sub directories included (File Type: 4)

Configure a workflow to read data from a file location and publish to a topic on Solace. Input Channel destination will contain the path of a file which contains a list of paths for the directories that need to be replicated.

```
spring:
  application:
    name: File Connector(Source)
  cloud:
    stream:
      bindings:
        input-0:
          destination: /home/centos/pubsubplus-connector-file-2.0.0/samples/config/directory-paths.cfg
          binder: file
```

Replicating Directories over SFTP

Test case 5: Replicating a complete Directory from a remote location over SFTP protocol. All files and sub directories included (File Type: 4 and SFTP Settings Enabled)

Configure a workflow to read data from a file location and publish to a topic on Solace. Input Channel destination will contain the path of a file which contains a list of paths for the directories that need to be replicated. SFTP Settings will be enabled with the details of the SFTP server and authentication credentials.

```
file:
  source:
    scheduler:
      restart_time_sec: 0
    general:
      adapter_id: SOURCE_CONN_ID1
      file_type: 4
      state_backup_path: backup_state.txt
      copy_file_mode_permissions: 0
      max_file_transfer_size: 999000
      max_files_allowed: 99999
      max_bandwidth_limit: 0
      clear_state_on_eod: 1
      eod_time_sec: -1
    sftp_settings:
      enabled: 0
      ip: XXX.XXX.XXX.XXX
      port: 22
      user: sftpuser
      password: password
```