Application Control Framework Deployment Guide

Version 1.7
Marc Henderson
01/14/2025

Revision Summary

Date	Revision History	Comments
01/14/2025	1.0	Updates for ACF 1.7 release

Table of Contents

Prerequisites

- Download and install the latest version of SnowSQL.
- Review and accept the Snowflake Provider and Consumer Terms of Service.
- The provider must determine which cloud regions the app will be available.
- The provider must create an account that serves as the main account for the ACF.
- The provider must create an account in each cloud region their native app will be available in. This account is used to collect events from consumer apps in each region. Events from this account are routed to the native app/ACF account via private data listings.
    - Once this account is created, the provider will set it as the Event account for the cloud region, by executing the following in the Snowflake Organization account, as `ORGADMIN`
        - `CALL SYSTEM$SET_EVENT_SHARING_ACCOUNT_FOR_REGION('<REGION>', 'PUBLIC', '<ACCOUNT_NAME>');`
            - <REGION> can be found by executing `SELECT CURRENT_REGION();`
            - <ACCOUNT_NAME> can be found by executing `SELECT CURRENT_ACCOUNT_NAME();`
        - `CALL SYSTEM$SYSTEM$ENABLE_GLOBAL_DATA_SHARING_FOR_ACCOUNT('<ACCOUNT_NAME>');`
            - <ACCOUNT_NAME> can be found by executing `SELECT CURRENT_ACCOUNT_NAME();`
- The user that will execute the scripts in each account must have either the `ACCOUNTADMIN` role granted or a role that can create roles and manage grants on the account to other roles.

Purpose

The purpose of this guide is to provide deployment instructions, using SnowSQL and/or Snowsight worksheets. The setup notebooks deploy the Application Control Framework (ACF) as is. In the event developers want to make changes to the ACF, they can fork the public repository, make desired modifications, then deploy the updated ACF code.

Consumption Tracking

Comments have been added to key framework objects to track Snowflake credit consumption of applications built using the Application Control Framework. **Please do not remove or modify comments.** Any modifications could result in the ability to properly track consumption.

Deploy Application Control Framework

The ACF is a framework that allows native app providers to build apps that allow providers to manage and understand consumer usage and create custom rules and controls that govern consumer app experience. The native app controls, rules, and consumers can be managed via the ACF's App Control Manager Streamlit UI.

Deployment of the ACF is done in two steps, event(s) account setup and ACF installation. There is an optional third step, to create a demo native app. Each step can be deployed either using the installer notebooks included with the repo or via the setup scripts that must be deployed via SnowSQL or in Snowsight. Each setup script requires a configuration file, which is defined below.

Modifications may be made to either or both parts.

Recommendation: it is easier to use the setup notebooks to deploy the part of the deployment process that is not being modified, rather than deploying via SnowSQL.

Configuration Parameters

Executing the scripts via SnowSQL requires a set of parameters to be passed via a config file. The config file must contain a value for the following parameters, unless otherwise noted:

- **DIR** - the URI where the `application_control_framework` root directory resides on the provider's local machine
    - **NOTE**: The URI formatting differs depending on the provider's local operating system:
        - Linux/MacOS:
            - The path must include the initial forward slash in the path (e.g. `/Users/mhenderson/Documents/GitHub`.
            - If the directory path and/or filename includes special characters, the entire URI must be enclosed in single quotation marks.
        - Windows:
            - You must include the drive and backslash in the path (e.g. `C:\Users\mhenderson\Documents\GitHub`).
            - If the directory path and/or filename includes special characters, the entire file URI must be enclosed in single quotes. Note that the separator character is a forward slash (/) in enclosed URIs (e.g. `'C:\Users\mhenderson\Documents\GitHub Repos'` for a path in Windows that includes a directory named GitHub Repos).
- **ORG_NAME** - the organization provider's Snowflake account belongs (this can be found by executing the `SELECT CURRENT_ORGANIZATION_NAME()` command)
    - **NOTE**: only required for the **config_events**.txt file.
- **ACF_ACCOUNT_NAME** - the name of the provider's Snowflake where the ACF will be deployed (this can be found by executing the `SELECT CURRENT_ACCOUNT_NAME()` command)
    - **NOTE**: only required for the **config_events**.txt file.
- **ACF_ACCOUNT_LOCATOR** - the account identifier for provider's Snowflake account where the ACF will be deployed (this can be found by executing the `SELECT CURRENT_ACCOUNT()` command)
- **EVENT_ACCOUNT_LOCATOR** - the account identifier for provider's Snowflake account where ACF events will be collected. (this can be found by executing the `SELECT CURRENT_ACCOUNT()` command)
    - **NOTE**: only required for the **config_events**.txt file.
- **APP_CODE** - an abbreviated representation for the name of the app (e.g. `SDE` for Sample Data Enrichment)

Recommendation: in the repo's `config` directory, create a config file for each event region and ACF account.

Step 1: Event Account Setup

The first step is to set up event account(s) for each region the native app will be deployed. The scripts create an event table (if one has not already been created), streams, tasks, and data listings to share consumer native app events from the event account to the main ACF account.

NOTE: Repeat this step for each event account created.

Prerequisites

The event account must have been created for each region the native app will be deployed, along with setting each account as the event account for the region. Please see the Prerequisites section for more information.

Script Execution

If not using the `01_event_account_setup` notebook, these scripts can be deployed either via SnowSQL or either in Snowflake Worksheets or an Integrated Development Environment (IDE) of choice.

NOTE: `<REGION>` = the cloud/region where the event account is being configured.

SnowSQL:

The event account scripts can be executed via SnowSQL, by calling the `setup.sql` script:

Execution: `./event_acct/setup/setup.sh config/config_events_<REGION>.txt`

non-SnowSQL

The ACF includes a python utility, `param_sub.py`, to replace parameter macros in the code with the values set in the config_events.txt file. The python script requires at least Python 2.7 and works with Python 3.

Execution: `python3 util/param_sub.py ../config/config_events_<REGION>.txt ../event_acct/scripts/ ../event_acct/scripts/`

Once complete, the following scripts are to be executed in order (either in a Snowflake worksheet or IDE of choice):

- `event_acct/scripts/01_account_setup_sub.sql`

- event_acct/scripts/02_create_events_st_procedure_sub.sql
- event_acct/scripts/03_create_remove_app_events_procedure
- event_acct/scripts/04_create_remove_all_events_procedure

Supporting Events for Multiple Apps in the Same Event Account

The event account setup supports the ability to have multiple ACF-created apps' events located in the same event account. This avoids the provider having to create an account per application per region.

In the event the provider wants to leverage an existing event account to stream and share events to the main ACF account, the provider can execute the following command in the event account:

```
USE ROLE ACCOUNTADMIN;
USE WAREHOUSE <APP_CODE>_EVENTS_WH;
CALL EVENTS.EVENTS.EVENTS_DT(TO_ARRAY('<APP_CODES>'));
```

NOTES:

- <APP_CODES> = a comma-separated string of app code(s) to create streams/tasks for.
- While multiple app's events can reside in the same event account, the events must be shared to the same main ACF account. The ACF supports multiple deployments, for different applications, in the same account.

Step 2: ACF Account Setup

The second step is to install the ACF to the "main" account. This account is different from an event account and will be where the ACF will reside and be managed.

The ACF includes all of the objects required to manage ACF functionality. In addition, this script also deploys the App Control Manager, the Streamlit UI that is used to perform such actions as define custom controls and rules, create app package versions, onboard consumers, etc.

Prerequisites

Recommendation: create event accounts for each region the native app will be deployed in before install the ACF. The ACF can be deployed without the event accounts created. Please see the Step 1: Event Account Setup section for more information.

Script Execution

If not using the `02_acf_installer` notebook, the ACF scripts can only be deployed via SnowSQL due to the requirement to use the `PUT` command to place files onto a stage.

SnowSQL:

The framework scripts can be executed via SnowSQL, by calling the `setup.sql` script:

Execution: `./acf_acct/setup/setup.sh config/config_acf.txt`

Add an Event Account after ACF Deployment

In the event one or more new event accounts are created after deploying ACF, the following commands should be executed to stream the events from each new event account to the ACF account.

```
USE ROLE P_<APP_CODE>_ACF_ADMIN;
USE WAREHOUSE P_<APP_CODE>_ACF_WH;
CALL P_<APP_CODE>_ACF_DB.EVENTS.CREATE_EVENTS_DB_FROM_LISTING();
CALL P_<APP_CODE>_ACF_DB.EVENTS.STREAM_TO_EVENT_MASTER((SELECT * FROM
TABLE(RESULT_SCAN(LAST_QUERY_ID(-1)))));
```

NOTES:

- `<APP_CODES>` = a comma-separated string of app code(s) to create streams/tasks for.
- The `CREATE_EVENTS_DB_FROM_LISTING` stored procedure creates a database from each event account listing shared to the ACF account.
- The `STREAM_TO_EVENT_MASTER` stored procedure creates a stream and task to stream the events to the `EVENT_MASTER` table.

Step 3: Demo App Setup (Optional)

The ACF also comes with a set of scripts that creates objects in the provider's dev environment to build a demo data enrichment app. The demo app includes sample source graph data, a function that can be shared with the consumer, and a stored procedure that enriches consumer data with data from the source graph.

This step is optional, but could prove valuable to providers wanting to practice building native apps using the ACF, before building their own app.

Script Execution

If not using the `03_create_demo_native_app` notebook, these scripts can only be deployed via SnowSQL due to the requirement to use the `PUT` command to place files onto a stage.

SnowSQL:

The framework scripts can be executed via SnowSQL, by calling the `setup.sql` script:

Execution: `./acf_acct/demo_app/setup/setup.sh config/config_acf.txt`

Objects Created

Refer to the **Application Control Framework - Detailed Design** document for a list of objects created.