



Fine-Tuning Airflow: Parameters You May Not Know About

Ivan Sayapin
Yu Lung Law

3.0

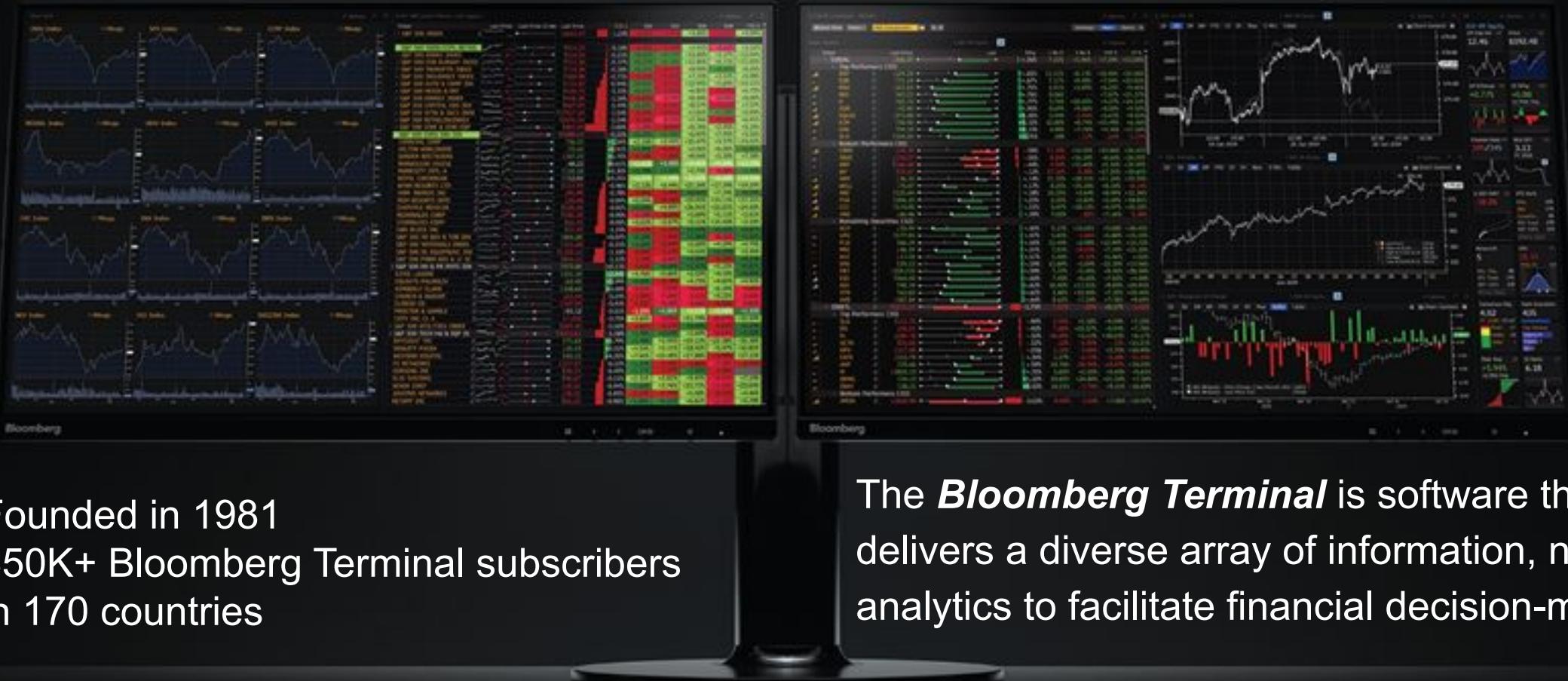
Agenda

- A Brief Introduction to Bloomberg
- Our Team and the Role of Apache Airflow
- Understanding Airflow Configurations
- Case Studies: How Configurations Resolved Four (4) Distinct Production Issues
- Key Insights and Takeaways

TechAtBloomberg.com

© 2025 Bloomberg Finance L.P. All rights reserved.

Bloomberg
Engineering



Founded in 1981
350K+ Bloomberg Terminal subscribers
in 170 countries

The **Bloomberg Terminal** is software that delivers a diverse array of information, news and analytics to facilitate financial decision-making.

TechAtBloomberg.com

© 2025 Bloomberg Finance L.P. All rights reserved.

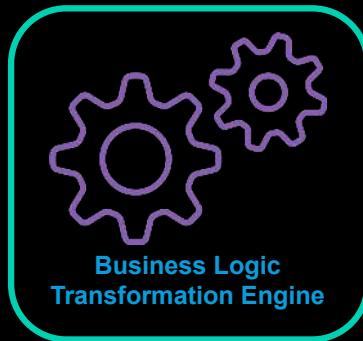
Bloomberg
Engineering

Our Team and the Role of Airflow

- Data Platform Engineering
- Manage & orchestrate data workflows for global financial professionals
- Ingestion of Alternative Data:
 - Pharmaceutical drug pricing and sales
 - Airline flight totals and jet fuel consumption
- Transform click and drag configurations into data pipelines



Raw Data
from Vendor



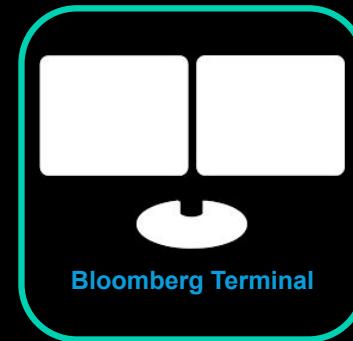
Business Logic
Transformation Engine



Load

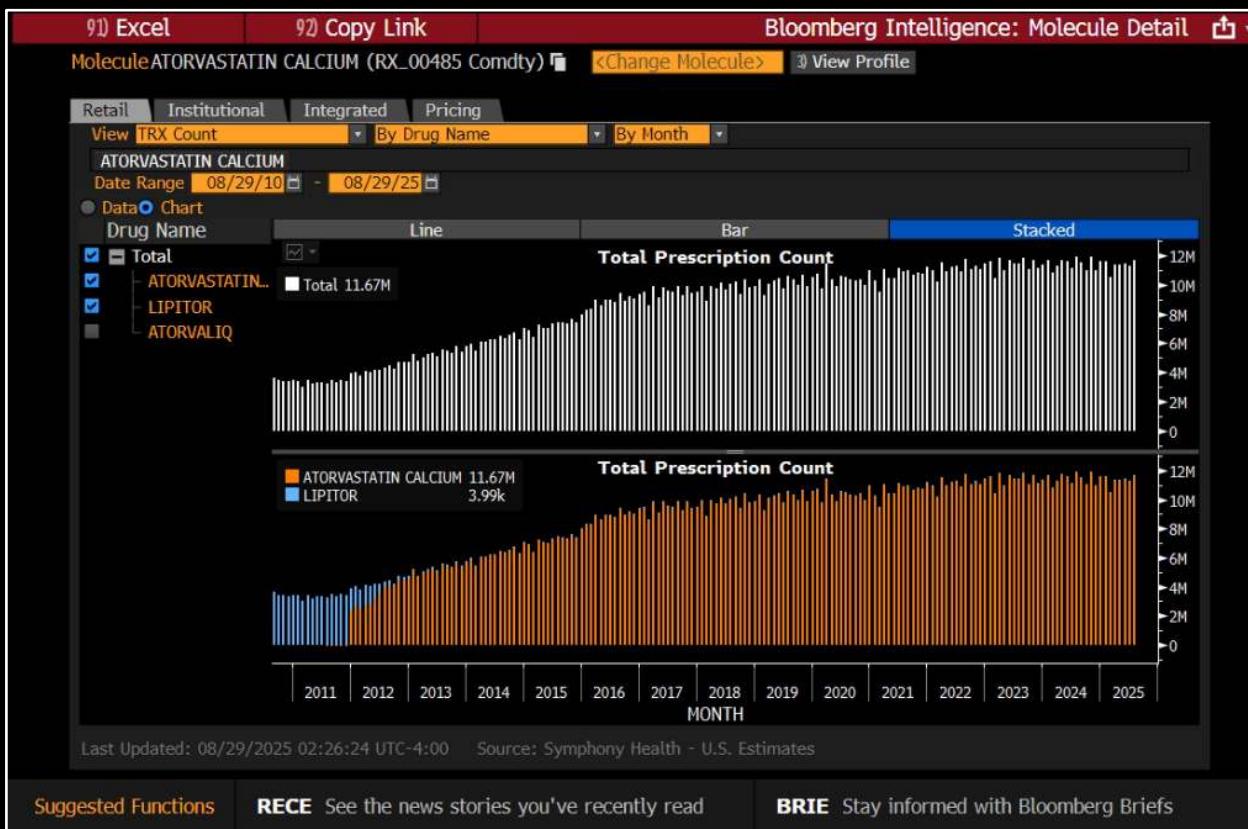


RDBMS



Bloomberg Terminal

Alternative Data On The Terminal



Actions Views Dataset Information Export  Dataset Library: Dataset Views 

BloombergNEF Jet Fuel Weekly Demand Forecast and Number of Scheduled Flights | Shortcut: DSET FLY 

Global Jet Fuel Demand & Flights Jet Fuel Demand Revision

View Jet Fuel Demand (in barrel) By Origin Region

By Week Any Flight Type Date Range 09/09/25 - 02/17/26

Data Chart

Origin Region 02/17/2026 02/10/2026 02/03/2026 01/27/2026 01/20/2026 01/13/2026 01/06/2026 12/30/2025

| Origin Region | 02/17/2026 | 02/10/2026 | 02/03/2026 | 01/27/2026 | 01/20/2026 | 01/13/2026 | 01/06/2026 | 12/30/2025 |
|------------------------------|------------|------------|------------|------------|------------|------------|------------|------------|
| Total | 7,073,728 | 6,969,033 | 6,945,775 | 6,955,744 | 6,977,192 | 7,026,927 | 7,299,812 | 7,279,965 |
| Asia | 2,480,046 | 2,481,196 | 2,479,050 | 2,480,201 | 2,482,130 | 2,485,620 | 2,485,049 | 2,481,968 |
| North America | 1,606,460 | 1,537,853 | 1,531,412 | 1,535,557 | 1,543,189 | 1,552,740 | 1,723,613 | 1,733,271 |
| Western Europe | 1,156,234 | 1,123,946 | 1,105,065 | 1,098,607 | 1,101,727 | 1,122,918 | 1,204,915 | 1,178,364 |
| Middle East and North Afr... | 613,716 | 612,814 | 611,826 | 610,937 | 613,139 | 617,789 | 623,889 | 622,072 |
| South and Central America | 427,909 | 425,723 | 430,192 | 435,205 | 436,873 | 437,246 | 447,269 | 449,305 |
| Eastern Europe | 368,445 | 366,623 | 366,368 | 366,775 | 367,527 | 373,506 | 376,954 | 380,399 |
| Oceania | 242,468 | 242,640 | 243,628 | 249,807 | 251,782 | 253,234 | 252,924 | 250,195 |
| Sub-Saharan Africa | 168,258 | 168,036 | 168,073 | 168,521 | 170,661 | 173,578 | 174,887 | 174,112 |
| ***Unspecified*** | 10,192 | 10,202 | 10,161 | 10,134 | 10,164 | 10,296 | 10,312 | 10,279 |

Last Updated: 08/23/2025 03:26:25 UTC-4:00 Source: Bloomberg FLY<GO>

Suggested Functions RECE See the news stories you've recently read INDE Bloomberg Indices Introduction

TechAtBloomberg.com

© 2025 Bloomberg Finance L.P. All rights reserved.

Bloomberg
Engineering

Understanding Airflow Configurations

- Parameters control how Airflow behaves
- Govern:
 - DAG parsing
 - Scheduler heartbeat
 - Database interactions
 - Task queueing & retries
 - Metadata cleanup & observability
- Airflow provides default values, but often need to be customized
- Defined via:
 - Configuration file
 - Environment variables: AIRFLOW_{SECTION}_{KEY}

Case Studies: How Configurations Resolved Four (4) Distinct Production Issues?

TechAtBloomberg.com

© 2025 Bloomberg Finance L.P. All rights reserved.

Bloomberg
Engineering

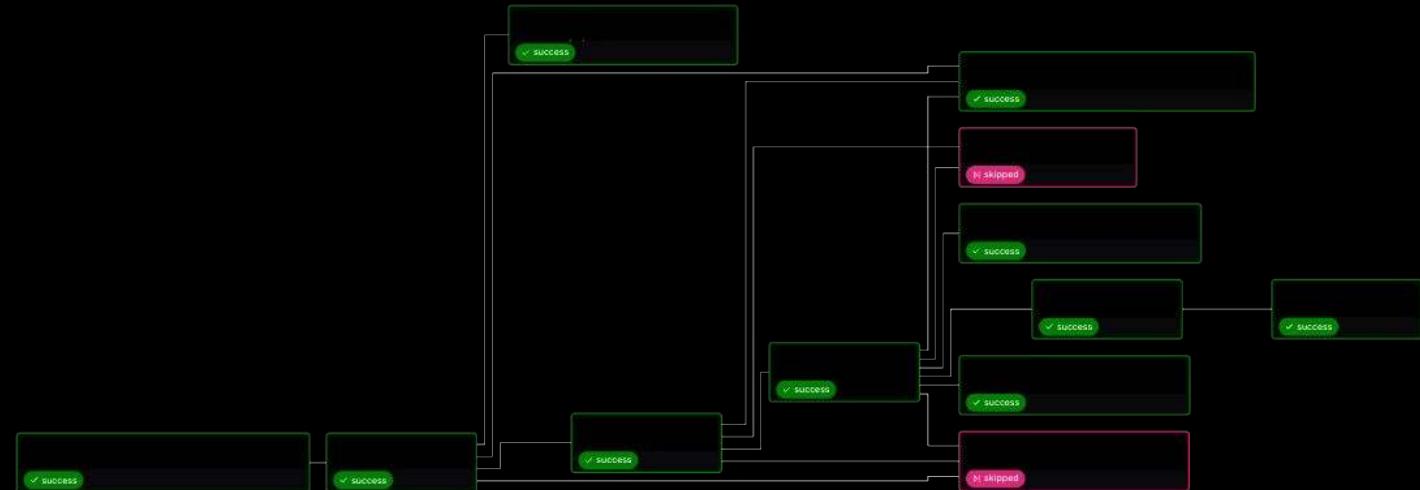
Case 1: Dynamic DAGs Not Parsed

Case 1: Dynamic DAGs Not Parsed

- We have 2,500+ dynamic DAGs
 - Each corresponding to a **distinct** dataset data pipeline
 - Similarity between DAGs, but slightly different based on configs
 - Configs come in **different permutations** and **vary** for each dataset
 - Dynamic DAGs avoid creating the same DAG, same tasks with slight modifications

Case 1: Dynamic DAGs Not Parsed

- `create_dynamic_dags.py` file generates dynamic DAGs at runtime
 - Calls services to:
 - Get a list of tables in our system
 - Obtain the tree of task descendants
 - Creates a dynamic DAG using those relationships
 - Some complex DAGs can have 150+ tasks
 - Process is resource intensive



Case 1: Dynamic DAGs Not Parsed

The screenshot shows the Airflow web interface with the 'Dags' tab selected. The main content area displays a search bar with the query '_ingestion_mc', several status filters (All, Failed, Running, Success), and a dropdown for 'Filter by tag'. Below these, it shows '0 Dags' and two view mode icons (grid and list). A prominent message 'No Dags found.' is displayed in a red-bordered box. The left sidebar contains navigation links for Home, Dags (which is selected and highlighted in blue), Assets, Browse, Admin, and Security.

Debugging DAG Processor Logs

In the logs, the value *50* appears in multiple places

| DAG File Processing Stats | | | | | | | |
|---------------------------|------------------------|-------|------------------|--------|----------|---------------|---------------------|
| Bundle | File Path | PID | Current Duration | # DAGs | # Errors | Last Duration | Last Run At |
| dags-folder | create_dynamic_dags.py | 35115 | 13.05s | 0 | 1 | 50.01s | 2025-08-22T15:03:10 |


```
dag-processor-1 | 2025-08-22T15:19:17.768+0000 {manager.py:1018} ERROR - Processor for DagFileInfo(rel_path=PosixPath('create_dynamic_dags.py'), bundle_name='dags-folder', bundle_path=PosixPath('/airflow-docker/home/dags'), bundle_version=None) with PID 77574 started 50 ago killing it.
```

[dag_processor] dag_file_processor_timeout

dag_file_processor_timeout

How long before timing out a DagFileProcessor, which processes a dag file 

Type:

integer

Default:

50

Environment Variable:

AIRFLOW__DAG_PROCESSOR__DAG_FILE_PROCESSOR_TIMEOUT

Increase Timeout

```
[dag_processor]
```

```
dag_file_processor_timeout = 500
```



| DAG File Processing Stats | | | | | | |
|---------------------------|------------------------|-------|------------------|--------|----------|---------------|
| Bundle | File Path | PID | Current Duration | # DAGs | # Errors | Last Duration |
| dags-folder | create_dynamic_dags.py | 18045 | 66.96s | 2964 | 0 | 78.33s |

The screenshot shows the Airflow web interface. On the left is a sidebar with icons for Home, Dags (selected), Assets, Browse, and Admin. The main area has tabs for Dags, Runs, and Task Instances, with Dags selected. It includes a search bar and filters for Failed, Running, Success, and a dropdown for task instances. A prominent green box highlights the number "2,964 Dags". Below this, two DAG entries are listed: "_ingestion_mc" and "_unload_mc". Each entry shows its schedule, latest run, and next run.

| DAG ID | Description | Schedule | Latest Run | Next Run |
|---------------|-----------------------|----------|------------|----------|
| _ingestion_mc | dynamic, ingestion_mc | | | |
| _unload_mc | unload_mc, dynamic | | | |

Case 1: Solution

- Increasing [dag_processor] dag_file_processor_timeout allowed us to:
 - Quickly fix the issue
 - Find the right value for the current processing times
 - Buy time to think about long term solutions
 - Cleaning up unused datasets
 - Splitting dynamic DAGs into multiple files
 - Dedicating more compute resources to the dag processor

Case 2: DAG Parsing Delay

Case 2: DAG Parsing Delay

| dag-processor-1 DAG File Processing Stats | | PID | Current Duration | # DAGs | # Errors | Last Duration | Last Run At |
|---|------------------------------------|-----|------------------|--------|----------|---------------|---------------------|
| dag-processor-1 | Bundle File Path | | | | | | |
| dag-processor-1 | dags-folder create_dynamic_dags.py | | | 2964 | 0 | 70.12s | 2025-09-08T20:29:50 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 1.83s | 2025-09-08T20:29:00 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.78s | 2025-09-08T20:30:58 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.74s | 2025-09-08T20:31:02 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.72s | 2025-09-08T20:31:01 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.71s | 2025-09-08T20:30:11 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.69s | 2025-09-08T20:30:59 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.68s | 2025-09-08T20:30:12 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.68s | 2025-09-08T20:31:32 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.67s | 2025-09-08T20:31:31 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.65s | 2025-09-08T20:29:10 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.63s | 2025-09-08T20:30:12 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.62s | 2025-09-08T20:31:29 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.59s | 2025-09-08T20:29:00 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.58s | 2025-09-08T20:31:01 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.48s | 2025-09-08T20:30:59 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.46s | 2025-09-08T20:31:03 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.45s | 2025-09-08T20:31:30 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.43s | 2025-09-08T20:31:03 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.42s | 2025-09-08T20:31:00 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.39s | 2025-09-08T20:30:57 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.39s | 2025-09-08T20:29:10 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.35s | 2025-09-08T20:31:04 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.34s | 2025-09-08T20:30:57 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.32s | 2025-09-08T20:31:04 |
| dag-processor-1 | dags-folder [REDACTED].py | | | 1 | 0 | 0.16s | 2025-09-08T20:29:01 |

[dag_processor] parsing_processes

parsing_processes

The DAG processor can run multiple processes in parallel to parse dags. This defines how many processes will run.

Type:

integer

Default:

2

Environment Variable:

AIRFLOW__DAG_PROCESSOR__PARSING_PROCESSES

[dag_processor] min_file_process_interval

min_file_process_interval

Number of seconds after which a DAG file is parsed. The DAG file is parsed every [dag_processor] `min_file_process_interval` number of seconds. Updates to DAGs are reflected after this interval. Keeping this number low will increase CPU usage.

Type:

integer

Default:

30

Environment Variable:

`AIRFLOW__DAG_PROCESSOR__MIN_FILE_PROCESS_INTERVAL`

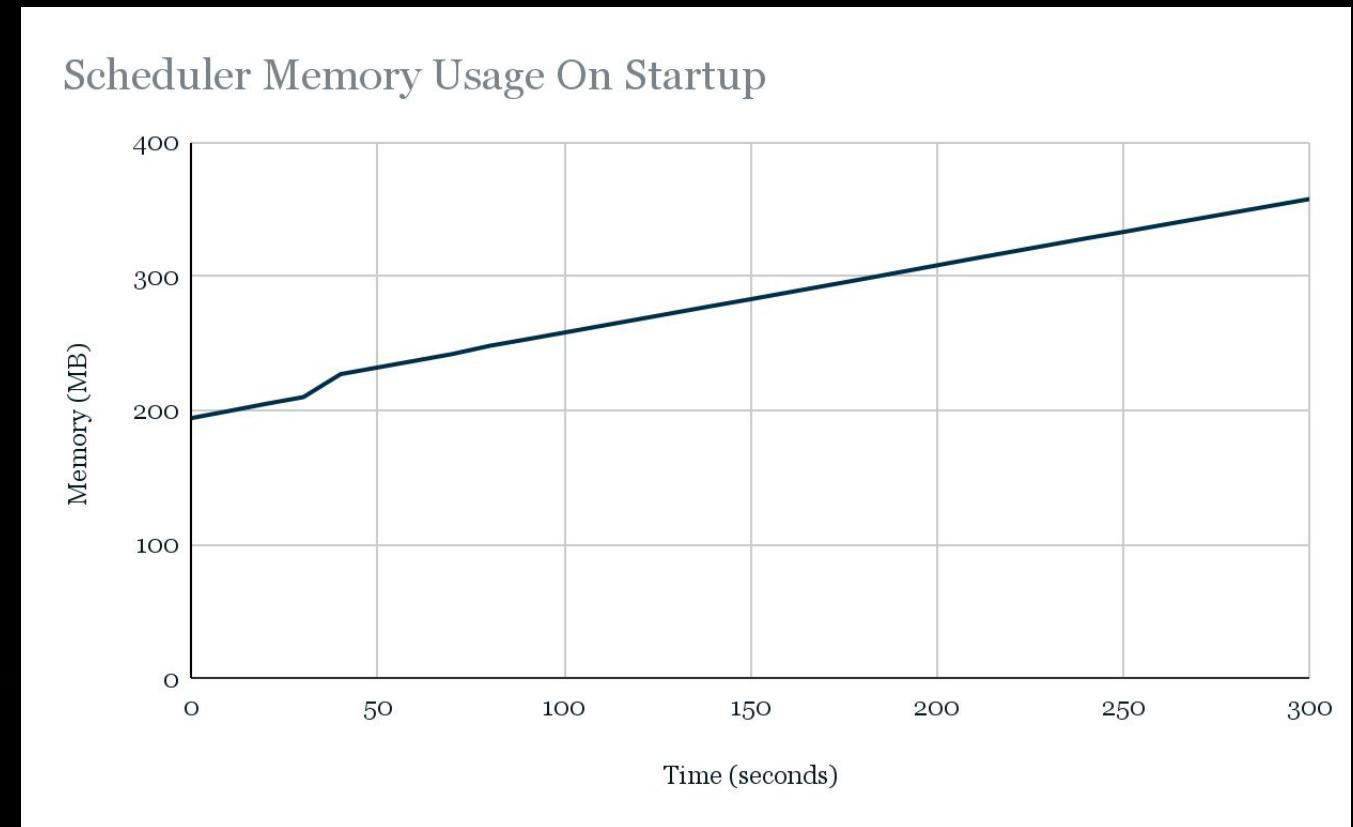
Case 2: Solution

- Tweaking `parsing_processes` and `min_file_process_interval` allowed us to:
 - Get faster DAG processing runs each iteration
 - Get DAG processor to run more frequently
 - Get close to “live” updates to DAG structures
 - Change no code
- Be wary of the CPU usage trade-off

Case 3: Scheduler Memory Leak

Case 3: Scheduler Memory Leak

- Scheduler was using an increasing amount of memory
- This was eventually causing it to get killed and restarted
- Symptoms of a memory leak



Scheduler RSS (Resident Set Size)

```
docker compose exec -it scheduler sh -lc \  
'while true; do ps -o pid,rss,command -p $(pgrep -f "scheduler"); \  
sleep 10; done'
```

| PID | RSS | COMMAND |
|-----|--------|---|
| 16 | 198828 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |
| PID | RSS | COMMAND |
| 16 | 204204 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |
| PID | RSS | COMMAND |
| 16 | 209708 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |
| PID | RSS | COMMAND |
| 16 | 214828 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |
| PID | RSS | COMMAND |
| 16 | 232376 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |
| PID | RSS | COMMAND |
| 16 | 237496 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |
| PID | RSS | COMMAND |
| 16 | 242616 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |



[scheduler] enable_tracemalloc

enable_tracemalloc

Added in version 3.0.0.

Whether to enable memory allocation tracing in the scheduler. If enabled, Airflow will start tracing memory allocation and log the top 10 memory usages at the error level upon receiving the signal SIGUSR1. This is an expensive operation and generally should not be used except for debugging purposes.

Type:

Boolean

Default:

False

Environment Variable:

AIRFLOW__SCHEDULER__ENABLE_TRACEMALLOC

[scheduler] enable_tracemalloc

enable_tracemalloc

Added in version 3.0.0.

Whether to enable memory allocation tracing in the scheduler. If enabled, Airflow will start tracing memory allocation and log the top 10 memory usages at the error level upon receiving the signal SIGUSR1. **This is an expensive operation and generally should not be used except for debugging purposes.**

Type:

Boolean

Default:

False

Environment Variable:

AIRFLOW__SCHEDULER__ENABLE_TRACEMALLOC

Tracemalloc Report

```
docker compose exec -it scheduler sh -c \
'kill -SIGUSR1 $(pgrep -f "airflow scheduler")'
```

```
scheduler-1 | 2025-09-10T19:49:26.114+0000 {scheduler_job_runner.py:307} ERROR - scheduler memory
usage:
scheduler-1 | Top 10
scheduler-1 | /airflow-docker/home/plugins/leaky_plugin.py:7: size=25.0 MiB, count=51, average=502 KiB
scheduler-1 | <frozen importlib._bootstrap_external>:757: size=2521 KiB, count=22861, average=113 B
scheduler-1 | /opt/bb/lib/python3.12/site-packages/sqlalchemy/sql/schema.py:1730: size=351 KiB,
count=455, average=790 B
scheduler-1 | /opt/bb/lib/python3.12/site-packages/gunicorn/config.py:247: size=217 KiB, count=644,
average=346 B
scheduler-1 | /opt/bb/lib/python3.12/site-packages/sqlalchemy/util/langhelpers.py:1185: size=180 KiB,
count=789, average=233 B
scheduler-1 | <frozen abc>:106: size=123 KiB, count=445, average=284 B
scheduler-1 | <frozen abc>:123: size=122 KiB, count=1612, average=77 B
scheduler-1 | /opt/bb/lib/python3.12/site-packages/sqlalchemy/sql/annotation.py:166: size=75.1 KiB,
count=118, average=651 B
```

Case 3: Solution

```
docker compose exec -it scheduler sh -c \  
'kill -SIGUSR1 $(pgrep -f "airflow scheduler")'
```

```
scheduler-1 | 2025-09-10T20:34:04.807+0000 {scheduler_job_runner.py:307} ERROR - scheduler  
memory usage:  
scheduler-1 |   Top 10  
scheduler-1 |   <frozen importlib._bootstrap_external>:757: size=2521 KiB, count=22860,  
average=113 B  
scheduler-1 |   /opt/bb/lib/python3.12/site-packages/sqlalchemy/sql/schema.py:1730: size=357  
KiB, count=461, average=794 B  
scheduler-1 |   /opt/bb/lib/python3.12/site-packages/gunicorn/config.py:247: size=217 KiB,  
count=644, average=346 B  
scheduler-1 |   /opt/bb/lib/python3.12/site-packages/sqlalchemy/util/langhelpers.py:1185:  
size=197 KiB, count=868, average=233 B  
scheduler-1 |   <frozen abc>:106: size=123 KiB, count=445, average=284 B
```

Case 3: Solution

```
docker compose exec -it scheduler sh -lc \  
'while true; do ps -o pid,rss,command -p $(pgrep -f "scheduler"); \  
sleep 10; done'
```

| PID | RSS | COMMAND |
|-----|--------|---|
| 20 | 207968 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |
| PID | RSS | COMMAND |
| 20 | 207968 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |
| PID | RSS | COMMAND |
| 20 | 207968 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |
| PID | RSS | COMMAND |
| 20 | 207968 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |
| PID | RSS | COMMAND |
| 20 | 207968 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |
| PID | RSS | COMMAND |
| 20 | 207968 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |
| PID | RSS | COMMAND |
| 20 | 207968 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |
| PID | RSS | COMMAND |
| 20 | 207968 | /opt/bb/bin/python3 /opt/bb/bin/airflow scheduler |

Case 4: Observability

TechAtBloomberg.com

© 2025 Bloomberg Finance L.P. All rights reserved.

Bloomberg
Engineering

Case 4: Observability

- With many components running, we need to make sure we have observability into components
- Ways to identify issues early for previous problems encountered:
 - No DAGs parsed
 - High DAG parsing run times
- Health of scheduler, DagRun, Celery, etc.

[metrics] statsd_on

```
pip install 'apache-airflow[statsd]'
```

statsd_on

Added in version 2.0.0.

Enables sending metrics to StatsD.

Type:

Boolean

Default:

False

Environment Variable:

AIRFLOW__METRICS__STATSD_ON

[metrics] metrics_allow_list

metrics_allow_list

Added in version 2.6.0.

Configure an allow list (comma separated regex patterns to match) to send only certain metrics.

Type:

string

Default:

"

Environment Variable:

AIRFLOW__METRICS__METRICS_ALLOW_LIST

Example:

"scheduler,executor,dagrun,pool,triggerer,celery"

Case 4: Solution

```
[metrics]
metrics_allow_list = dag_processing,dagrun
```

- `dag_processing.last_run.seconds_ago.<dag_file>`
 - Alert if it's been too long since last run
- `dag_processing.last_duration.<dag_file>`
 - Alert if DAG files are taking too long to process
- `dagrun.<dag_id>.first_task_scheduling_delay`
 - Alert if there is a large delay for first task runs for DAGs of interest

Takeaways

- Small config tweaks: big impact on scale, stability and observability
 - [dag_processor] dag_file_processor_timeout
 - [dag_processor] parsing_processes
 - [dag_processor] min_file_process_interval
 - [scheduler] enable_tracemalloc ←
 - [metrics] statsd_on
 - [metrics] metrics_allow_list
- Test with different values of configurations in lower environments
- Temporarily changing configs can buy time to look into long term solutions
- Read the documentation and keep up to date with latest configurations

Engineering

Bloomberg

Thank you!

Questions?

Learn more: <https://TechAtBloomberg.com>

We're hiring: <https://www.bloomberg.com/careers>

TechAtBloomberg.com

© 2025 Bloomberg Finance L.P. All rights reserved.