에어플로우를 argcd에서 sync하고 다시 배포할때, dag에 오류가 있으면 안뜨나?

짧게 답이면: **네. ArgoCD는 DAG 파싱 오류를 모릅니다.**
ArgoCD가 `Sync` 해서 Airflow(웹서버/스케줄러/워커)가 정상 기동만 하면 상태는 Healthy로 보이고, **DAG 파일에 에러가 있으면 그 DAG만 UI에 안 뜨거나 "Import Errors"에만 뜹니다.** 배포 자체는 실패로 안 잡혀요.

어떻게 보이나요?

- **UI**: `Browse → Import Errors`(2.9.2)에서 해당 파일/트레이스백을 확인할 수 있습니다. DAG 목록에는 안 뜨는 게 정상입니다.
- **로그**: 스케줄러 로그에 보통 이렇게 남습니다: *"Failed to import: <dag_file> ... Traceback ..."*
- **CLI**(컨테이너 안):

```bash
bash

# 스케줄러/웹서버 컨테이너에서
airflow dags list      # 안 뜨면 파싱 실패로 의심
airflow dags show <dag_id>  # 파싱 통과해야만 동작
```

왜 이런가요?

- ArgoCD는 쿠버네티스 리소스(Deployment/StatefulSet 등)가 **정상 실행**되는지만 봅니다.
- **DAG 파싱 성공/실패는 애플리케이션 내부 상태**라 ArgoCD Health와는 별개예요.

빠른 디버깅 체크리스트

1. **스케줄러 로그 확인**

```bash
bash

kubectl logs deploy/airflow-scheduler -n <ns> | grep -i "import" -n
```

2. **웹서버 UI → Import Errors** 페이지 확인
3. **의존 라이브러리**: DAG가 import하는 패키지가 **런타임 이미지에 설치**되어 있는지 확인(특히 `KubernetesPodOperator` 사용 시).
4. **경로/마운트**: DAGs 폴더가 올바르게 마운트/동기화되는지(Helm 값에서 `dags` 설정, git-sync/PVC/S3 등).

"배포 전에 막고 싶다"면(추천)

ArgoCD는 모니터이므로, **CI 또는 Sync Hook에서 사전 검증**을 걸어두면 실수 줄일 수 있습니다.

옵션 A. PreSync Hook(Job)으로 문법/임포트 검사

- 간단히 문법만:

```
bash

python -m compileall /opt/airflow/dags
```

- 임포트까지(런타임 이미지 기준):

```
bash

python -c "import pkgutil, sys; sys.path.append('/opt/airflow/dags')"
airflow dags list  # 0이 아니면 실패 처리
```

> 주의: **`airflow dags list`**는 Airflow 환경/커넥션까지 요구할 수 있어요. 그래도 "임포트 에러"는 대부분 걸러집니다.

옵션 B. CI에서 정적 체크

- **`ruff/flake8/pyproject`** 기반 린트 + **`python -m compileall`**로 문법 에러 선제 차단.
- 필요 시 **`pytest`**로 DAG 유효성 테스트(예: **`DagBag().dag_ids`**가 비어있지 않은지).

옵션 C. 헬스체크 강화

- 스케줄러 컨테이너에 사이드카/크론으로 간단한 **self-check 스크립트**를 두고, 최근 파싱 실패가 있으면 **`/health`** 실패로 바꾸는 식도 가능(운영 난이도는 조금 올라갑니다).

원하시면 지금 쓰시는 Helm values/배포 구조(git-sync/PVC/S3 중 무엇인지) 기준으로 **PreSync Hook Job YAML**이랑 간단한 체크 스크립트까지 바로 만들어드리겠습니다.

> 에어플로 도커 이미지를 바꾸었다가 원복했더니 잘되던게 안되는상태야

바뀐 이미지를 "원복"했는데도 갑자기 안 돌아간다면, 보통은 **이미지/태그 캐시, DB 스키마, 의존성, 권한, git-sync** 중 하나가 원인입니다. 아래 순서대로 딱딱 확인해보시죠. (한 번에 다 하지 말고 위에서부터 내려오면 원인 금방 잡힙니다.)

1) 진짜 그 이미지 맞나? (태그 캐시/다이제스트)

동일 태그로 왔다갔다 하면 노드에 캐시가 남아 **옛날 레이어**로 뜨는 경우가 많습니다.

```
bash

# 어떤 이미지가 실제로 뜨는지(다이제스트) 확인
kubectl get pod -n <ns> -l app=airflow-scheduler -o
jsonpath='{.items[0].status.containerStatuses[0].imageID}{"\n"}'
kubectl get pod -n <ns> -l app=airflow-web -o
jsonpath='{.items[0].status.containerStatuses[0].imageID}{"\n"}'
# 디플로이먼트에 선언된 이미지(tag)도 같이 확인
kubectl get deploy airflow-scheduler -n <ns> -o
jsonpath='{.spec.template.spec.containers[0].image}{"\n"}'
```

- **다이제스트가 기대값과 다르면**: 태그가 캐시일 수 있습니다. 임시로 `imagePullPolicy: Always`로 바꾸고 `kubectl rollout restart deploy ...` 하거나, 문제 노드에서 이미지를 비워주세요(`crictl rmi` 등).

2) 업/다운그레이드 후 DB 스키마 충돌

최근에 3.0.x → 2.x로 되돌린 적이 있다면, **메타데이터 DB 스키마**가 안 맞아 웹/스케줄러가 조용히 죽습니다.

```
bash


# 컨테이너 안에서
airflow version
airflow db version
airflow db check-migrations   # (2.7+)
# 에러나면 현재 이미지 버전에 맞춰 마이그레이션
airflow db upgrade   # 또는 필요 시 downgrade 경로 검토
```

로그 패턴: `alembic` 관련 에러, “table/column does not exist”, “revision mismatch”.

3) 파이썬 패키지 불일치(프로바이더/requirements 누락)

이미지를 바꿨다가 복귀하면, `requirements.txt`가 **빌드에 반영 안 됐거나**, 다른 버전으로 깔려 **DAG import**가 깨질 수 있습니다.

```
bash


# 컨테이너 안
pip freeze | grep -E 'apache-airflow|providers|pyspark'
airflow providers list | head -n 30
# DAG 임포트 검증
python - <<'PY'
from airflow.models import DagBag
d=DagBag()
print("import_errors:", len(d.import_errors))
print("errors:", d.import_errors)
print("dags:", d.dag_ids[:10])
PY
```

- `ModuleNotFoundError`, `ImportError` 나오면 **그 패키지**를 이미지에 다시 포함하거나, KPO로 실행되는 태스크 이미지에도 동일하게 설치해야 합니다.

4) 권한/UID 변경으로 DAG/Logs 폴더 접근 실패

베이스 이미지가 바뀌며 `uid/gid`가 달라지면 **/opt/airflow/dags, /opt/airflow/logs**에 `Permission denied`.

```
bash


# 스케줄러/웹서버 로그에 Permission denied 여부 확인
kubectl logs deploy/airflow-scheduler -n <ns> | grep -i "permission"
# Pod 안에서 실제 권한
id
ls -ld /opt/airflow /opt/airflow/dags /opt/airflow/logs
```

- 해결: initContainer로 `chown -R <uid>:<gid> /opt/airflow/{dags,logs}` 또는 `fsGroup/runAsUser` 일치.
- Helm values의 `uid`, `gid`, `extraInitContainers`가 예전이랑 달라졌는지도 봐주세요.

5) git-sync/공유볼륨 경로가 어긋남

이미지 교체로 **DAG 경로 변수**가 달라지면 스케줄러가 비어있는 폴더를 본 채로 뜹니다.

- 확인 포인트:
  - **`AIRFLOW__CORE__DAGS_FOLDER`**
  - git-sync의 **`GIT_SYNC_ROOT`**, **`GIT_SYNC_DEST`**, 공유 마운트 경로(예: **`/dags`** vs **`/opt/airflow/dags`**)

```bash
bash

env | grep -E 'AIRFLOW__CORE__DAGS_FOLDER|AIRFLOW_HOME|GIT_SYNC'
```

- 웹 UI의 **Browse → Import Errors**도 같이 체크.

6) ArgoCD/Helm 값 드리프트

원복 과정에서 values가 미묘하게 달라졌을 수 있습니다.

```bash
bash

# 현재 배포된 값
helm -n <ns> get values airflow -a
# ArgoCD에서 실제 쿠버네티스와의 차이
argocd app diff <app-name>
argocd app history <app-name>
```

- **`env`**, **`extraEnv`**, **`image`**, **`imagePullPolicy`**, **`securityContext`**, **`volumes/volumeMounts`** 항목이 예전과 1글자라도 다르면 의외의 증상 나옵니다.

7) KPO만 실패한다면(웹/DAG는 보이는데 태스크 실패)

- KPO의 **task pod 이미지**가 교체되었거나 요구 패키지 미설치.
- 해당 task pod 로그를 직접 확인:

```bash
bash

kubectl logs -n <ns> pod/<kpo-task-pod> -c base # 또는 main 컨테이너
```

흔한 로그 시그니처 → 바로 쓰는 처방전

- **`ImportError/ModuleNotFoundError` (DAG 안에서)** → 이미지에 패키지 재설치, **`airflow providers list`** 확인.
- **`alembic/migration` 에러** → **`airflow db upgrade`**(또는 정확한 버전으로 재배포), 버전 역행이면 호환 표 확인.
- **`Permission denied`** → **`fsGroup/runAsUser/chown init`** 재적용.
- **DAG가 0개**인데 에러 없음 → **`AIRFLOW__CORE__DAGS_FOLDER`**/git-sync 경로 어긋남.

- **ArgoCD는 Healthy인데 UI에 DAG가 없거나 Import Errors만 증가** → 어플리케이션 내부 오류(의존성/경로)로 판단.

바로 실행해볼 "3분 점검" 패키지

1. 다이제스트 일치 확인 → **`imagePullPolicy: Always`** 임시 적용, **`rollout restart`**
2. 컨테이너 진입:

```
bash


airflow version && airflow db check-migrations || true
python - <<'PY'
from airflow.models import DagBag
d=DagBag(); print("err:", d.import_errors); print("dags:", len(d.dag_ids))
PY
```

3. **`Permission denied`**/경로/요구 패키지 로그 스캔

원하시면 지금 쓰시는 **Helm values.yaml**(민감정보 가리고)랑, 스케줄러/웹서버 **최근 200줄 로그**만 붙여주세요. 그걸로 어디서 끊겼는지 정확히 짚어드릴게요. 두 가지만 확인해서 보내주시면 됩니다:

1. **`kubectl get pods -n <ns> -o wide | grep airflow`** 결과
2. **`airflow db version && airflow version`** 출력