

hive — CWE-407 Disclosure Brief

Apache Hive — CWE-407 Disclosure Brief

2026-03-26 · Patch available — awaiting upstream merge

Finding

Two $O(n^2)$ defects in Apache Hive's MapReduce query plan generator. Both patched. Patches ready for upstream review. Both defects are in `GenMRProcContext.java` — the context object threaded through Hive's operator graph walk during MapReduce plan generation.

The Defects

hive-0001 (PATCHED — HIGH): `ql/src/java/org/apache/hadoop/hive/ql/optimizer/GenMRProcContext.java:248`

```
// taskToSeenOps: HashMap<Task<?>, List<Operator<?>>>
public boolean isSeenOp(Task task, Operator operator) {
    List<Operator<?>> seenOps = taskToSeenOps.get(task);
    return seenOps != null && seenOps.contains(operator); // O(T) scan
}
```

`isSeenOp()` is called for every operator encountered during the MapReduce plan walk. With T operators per task, each call scans up to T entries: **$O(T^2)$ total per task.**

hive-0002 (PATCHED — MEDIUM): `GenMRProcContext.java:142` + `GenMRFileSink1.java:101`

```
private List<FileSinkOperator> seenFileSinkOps;
// ...
if (!seenFSOps.contains(fsOp)) { // O(F) linear scan
    seenFSOps.add(fsOp);
}
```

`seenFileSinkOps` is a `List<FileSinkOperator>`. The `contains()` check in `GenMRFileSink1.process()` fires on every file sink operator encountered during the operator graph walk: **$O(F^2)$ total.**

Complexity Proof

hive-0001: For T operators mapped to a single task: - Each `isSeenOp()` call on an already-full list scans T entries - Called T times during plan generation - Total comparisons: $1 + 2 + \dots + T = T(T+1)/2 = O(T^2)$

At $T=100$ operators per task: 10,000 comparisons. With `HashSet`: 100 comparisons.

Measured ratio at $T=100$: $50.3\times$ (defective=10,000, fixed=199).

Impact

Every Hive query that uses MapReduce execution hits `GenMRProcContext`. Complex queries with many operators — joins, aggregations, subqueries — maximize T and directly amplify the $O(T^2)$ cost. Hive clusters running large analytical workloads (ETL pipelines, reporting queries, data lake processing) pay this cost on every query plan generation.

File-heavy queries (multiple output stages, INSERT OVERWRITE with multiple partitions) are additionally affected by hive-0002.

The Fix

hive-0001: Replace `List<Operator<?>>` with `Set<Operator<?>>`:

```
// Before
private HashMap<Task<?>, List<Operator<? extends OperatorDesc>>> taskToSeenOps;
// After
private HashMap<Task<?>, Set<Operator<? extends OperatorDesc>>> taskToSeenOps;
// new ArrayList<>() → new HashSet<>() in addSeenOp()
```

hive-0002: Replace `List<FileSinkOperator>` with `Set<FileSinkOperator>`:

```
// Before
private List<FileSinkOperator> seenFileSinkOps;
// After
private Set<FileSinkOperator> seenFileSinkOps;
```

`Operator` already implements `equals()` and `hashCode()` — no additional changes needed.

Patch

Fix available: `defects/hive/patch/hive-0001-0002-genmrproccontext-seenops-hashset.patch`

Two-file patch: `GenMRProcContext.java` + `GenMRFileSink1.java`. Changes `ArrayList` to `HashSet` in four locations. No behavioral change — `Set` semantics match the existing membership-test use case exactly.

Unit test: 4/4 pass. At T=100 operators: defective=10,000 comparisons, fixed=199 comparisons, **50.3× speedup**.

What We Ask

A patch is ready for review.

1. Confirm receipt and assign a JIRA reference (HIVE project at issues.apache.org/jira).
2. Assess severity — hive-0001 fires on every MapReduce query plan; hive-0002 fires on every file-sink-heavy query.
3. Coordinate a disclosure date — we are targeting 90 days from first contact.
4. We will credit the Apache Hive team in the public disclosure. Preferred acknowledgment format welcome.

Contact: see cover email. This brief is confidential until coordinated disclosure.