OWASP DEPENDENCY-TRACK

Community Meeting
August 2025

Organizational

- Community meetings are recorded and uploaded to YouTube
- Slides will be published in the `DependencyTrack/community` repository
- Please use the Zoom chat to ask questions during the presentation
- There will be an open Q&A section towards the end

Call for Guest Presentations

- Want to brag about the cool DT setup you've built?
- Want to vent about what needs improvement?
- Want to get input on DT-related designs?
- Want to propose changes?

We'd love to host you here!

Agenda

1. Recent Releases
2. Upcoming Releases
3. Guest Presentation:
   Component Health Integration
4. Q&A

Recent Releases

Recent Releases: v4.13.3

- Released August 4th.
- 10 bugfixes that landed since the v4.13.2 release.
- Base image and dependency updates.
- Noteworthy:
  - Significant speedup of internal vuln analysis for components with CPEs.
  - Improved performance of BOM exports, particularly when component properties are involved.

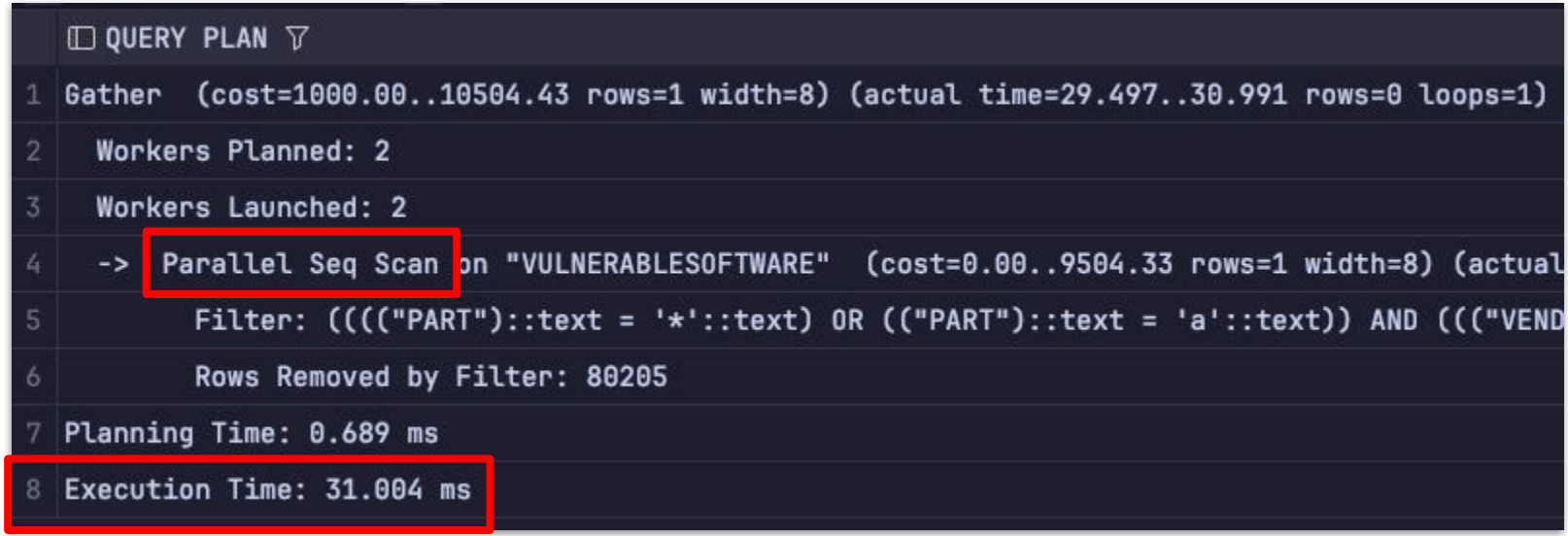
https://docs.dependencytrack.org/changelog/#v4-13-3

v4.13.3: Faster Internal Vuln Analyzer

- cpe:2.3:a:foo:bar:1.0.0:*:*:*:*:*:*:*
- DB query to find potential vulnerability matches:

```
SELECT "ID"
FROM "VULNERABLESOFTWARE"
WHERE
  ("PART" = '*' OR "PART" = 'a')
  AND ("VENDOR" = '*' OR "VENDOR" = 'foo')
  AND ("PRODUCT" = '*' OR "PRODUCT" = 'bar');
```

v4.13.3: Faster Internal Vuln Analyzer



```
QUERY PLAN
Gather  (cost=1000.00..10504.43 rows=1 width=8) (actual time=29.497..30.991 rows=0 loops=1)
  Workers Planned: 2
  Workers Launched: 2
  ->  Parallel Seq Scan on "VULNERABLESOFTWARE"  (cost=0.00..9504.33 rows=1 width=8) (actual
        Filter: ((("PART")::text = '*'::text) OR (("PART")::text = 'a'::text)) AND ((("VEND
        Rows Removed by Filter: 80205
Planning Time: 0.689 ms
Execution Time: 31.004 ms
```

All involved columns are indexed. What gives?

v4.13.3: Faster Internal Vuln Analyzer

```
SELECT "ID" FROM "VULNERABLESOFTWARE" WHERE "PART" = '*' AND "VENDOR" = '*' AND "PRODUCT" = '*'
UNION ALL
SELECT "ID" FROM "VULNERABLESOFTWARE" WHERE "PART" = '*' AND "VENDOR" = '*' AND "PRODUCT" = 'bar'
UNION ALL
SELECT "ID" FROM "VULNERABLESOFTWARE" WHERE "PART" = '*' AND "VENDOR" = 'foo' AND "PRODUCT" = '*'
UNION ALL
SELECT "ID" FROM "VULNERABLESOFTWARE" WHERE "PART" = '*' AND "VENDOR" = 'foo' AND "PRODUCT" = 'bar'
UNION ALL
SELECT "ID" FROM "VULNERABLESOFTWARE" WHERE "PART" = 'a' AND "VENDOR" = '*' AND "PRODUCT" = '*'
UNION ALL
SELECT "ID" FROM "VULNERABLESOFTWARE" WHERE "PART" = 'a' AND "VENDOR" = '*' AND "PRODUCT" = 'bar'
UNION ALL
SELECT "ID" FROM "VULNERABLESOFTWARE" WHERE "PART" = 'a' AND "VENDOR" = 'foo' AND "PRODUCT" = '*'
UNION ALL
SELECT "ID" FROM "VULNERABLESOFTWARE" WHERE "PART" = 'a' AND "VENDOR" = 'foo' AND "PRODUCT" = 'bar';
```

Equivalent “exploded” query without OR

v4.13.3: Faster Internal Vuln Analyzer

```
QUERY PLAN
Append  (cost=0.42..67.66 rows=8 width=8) (actual time=0.059..0.060 rows=0 loops=1)
  ->  Index Scan using "VULNERABLESOFTWARE_CPE_PURL_PARTS_IDX" on "VULNERABLESOFTWARE"  (cost=0.42..8.44 rows=1 width=8) (actu
        Index Cond: ((("PART")::text = '*'::text) AND (("VENDOR")::text = '*'::text) AND (("PRODUCT")::text = '*'::text))
  ->  Index Scan using "VULNERABLESOFTWARE_CPE_PURL_PARTS_IDX" on "VULNERABLESOFTWARE" "VULNERABLESOFTWARE_1"  (cost=0.42..8.4
        Index Cond: ((("PART")::text = '*'::text) AND (("VENDOR")::text = '*'::text) AND (("PRODUCT")::text = 'bar'::text))
  ->  Index Scan using "VULNERABLESOFTWARE_CPE_PURL_PARTS_IDX" on "VULNERABLESOFTWARE" "VULNERABLESOFTWARE_2"  (cost=0.42..8.4
        Index Cond: ((("PART")::text = '*'::text) AND (("VENDOR")::text = 'foo'::text) AND (("PRODUCT")::text = '*'::text))
  ->  Index Scan using "VULNERABLESOFTWARE_CPE_PURL_PARTS_IDX" on "VULNERABLESOFTWARE" "VULNERABLESOFTWARE_3"  (cost=0.42..8.4
        Index Cond: ((("PART")::text = '*'::text) AND (("VENDOR")::text = 'foo'::text) AND (("PRODUCT")::text = 'bar'::text))
  ->  Index Scan using "VULNERABLESOFTWARE_CPE_PURL_PARTS_IDX" on "VULNERABLESOFTWARE" "VULNERABLESOFTWARE_4"  (cost=0.42..8.4
        Index Cond: ((("PART")::text = 'a'::text) AND (("VENDOR")::text = '*'::text) AND (("PRODUCT")::text = '*'::text))
  ->  Index Scan using "VULNERABLESOFTWARE_CPE_PURL_PARTS_IDX" on "VULNERABLESOFTWARE" "VULNERABLESOFTWARE_5"  (cost=0.42..8.4
        Index Cond: ((("PART")::text = 'a'::text) AND (("VENDOR")::text = '*'::text) AND (("PRODUCT")::text = 'bar'::text))
  ->  Index Scan using "VULNERABLESOFTWARE_CPE_PURL_PARTS_IDX" on "VULNERABLESOFTWARE" "VULNERABLESOFTWARE_6"  (cost=0.42..8.4
        Index Cond: ((("PART")::text = 'a'::text) AND (("VENDOR")::text = 'foo'::text) AND (("PRODUCT")::text = '*'::text))
  ->  Index Scan using "VULNERABLESOFTWARE_CPE_PURL_PARTS_IDX" on "VULNERABLESOFTWARE" "VULNERABLESOFTWARE_7"  (cost=0.42..8.4
        Index Cond: ((("PART")::text = 'a'::text) AND (("VENDOR")::text = 'foo'::text) AND (("PRODUCT")::text = 'bar'::text))
Planning Time: 0.253 ms
Execution Time: 0.100 ms
```

Sometimes more is… less?

Recent Releases: Hyades v0.6.0

- Released July 31st.
- “Savepoint” before more intrusive changes get merged.
- Started to modularize the code base.
  - Enables build parallelization and caching.
  - Faster local builds, faster test runs in CI.
- Introduced new major REST API version.
  - Using OpenAPI in a “spec first” approach.
  - Tooling and tests in place to ensure consistency, spec compliance, and detect breaking changes.
  - v1 will remain, but endpoints may be deprecated.
- Last release that includes Kafka requirement.

https://dependencytrack.github.io/hyades/latest/getting-started/upgrading/#upgrading-to-060

Upcoming Releases

Upcoming Releases: v4.13.4 / 4.14.0

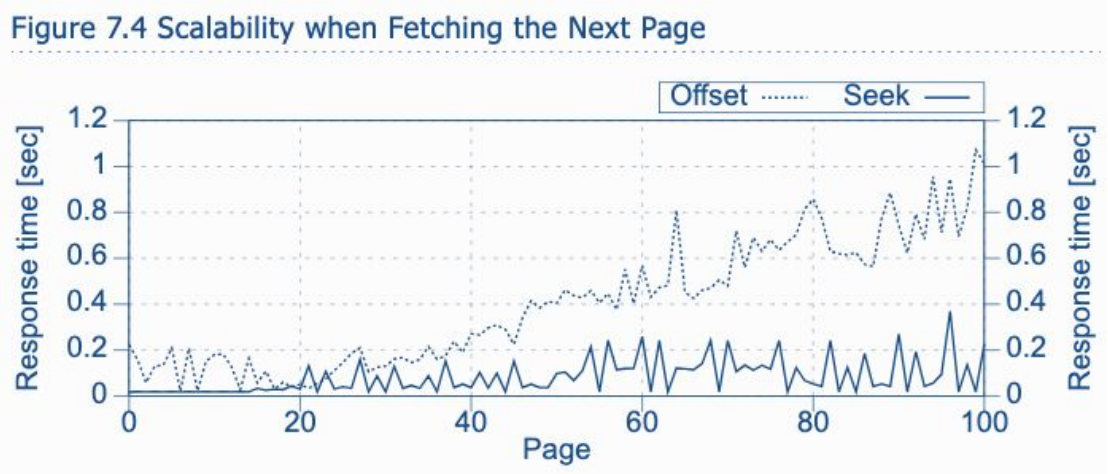
- Support for NVD feed files 2.0.
  - Could deprecate or remove NVD REST API mirroring as data is identical.
- Fix for possible endless recursion in dependency graph view.
- More generally: Get pending PRs reviewed and merged.

Upcoming Releases: Hyades v0.7.0

- Removal of Kafka requirement.
- Introduction of durable execution engine.
- Introduction of plugin mechanism and publishing of plugin API.
- Discontinuation of various services we created before.
- Frontend updates to leverage REST API v2 endpoints.

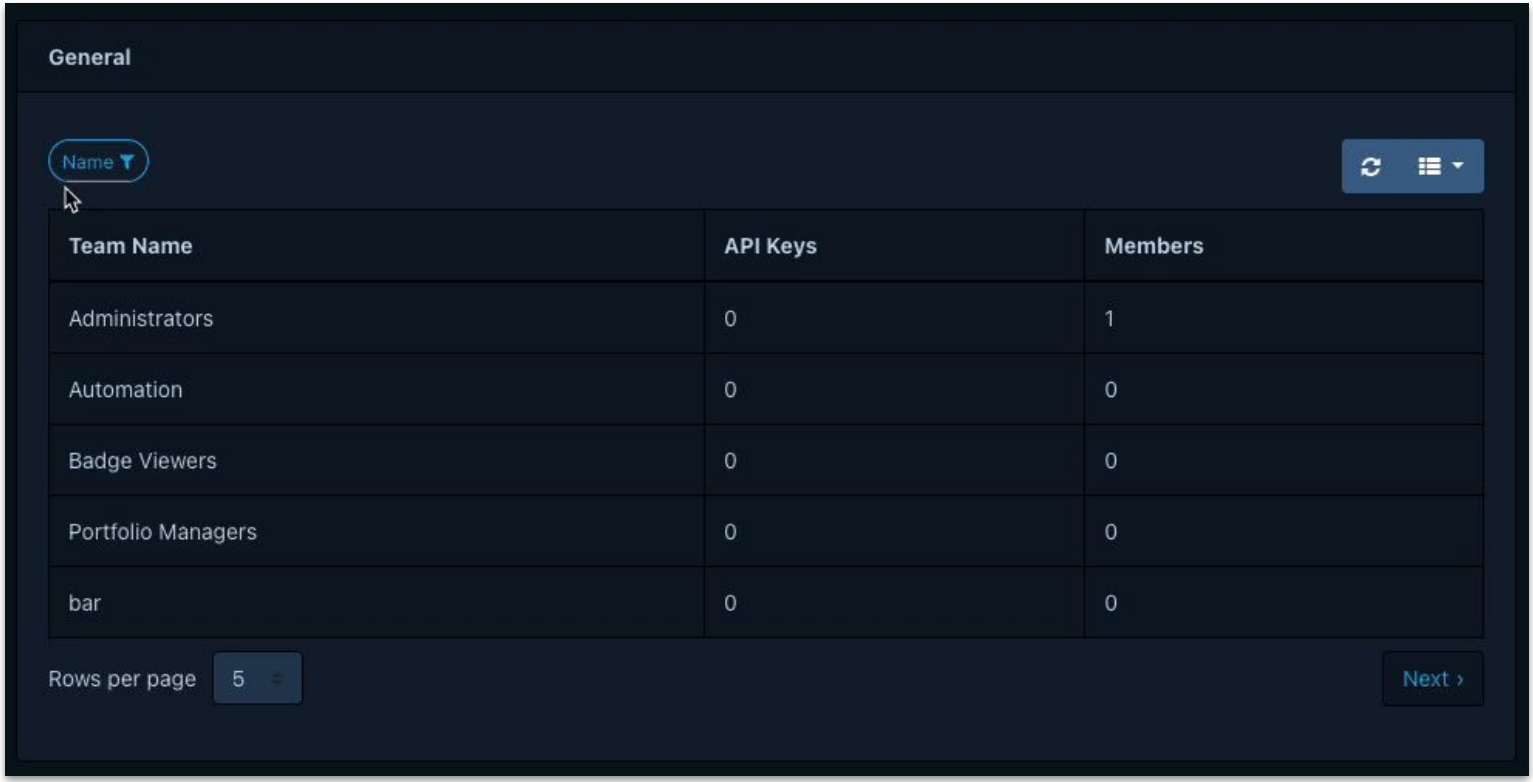
Upcoming Releases: Hyades v0.7.0

- Pagination mechanism in API v2 changes from OFFSET to SEEK.
  - a.k.a. keyset pagination, token pagination, …
  - Navigation using *next* and *previous* links vs. `offset` and `limit`.
  - Better suited for large data sets.



https://use-the-index-luke.com/sql/partial-results/fetch-next-page

Upcoming Releases: Hyades v0.7.0



GUEST PRESENTATION: COMPONENT HEALTH INTEGRATION

Q&A