

OWASP DEPENDENCY-TRACK

Community Meeting
May 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

Agenda

1. Overview: v5
2. Functional changes
 - a. Policy
 - b. Access Control
3. Non-functional changes
 - a. Database
 - b. Async Processing
 - c. Data Retention
4. Notes for Contributors
5. What's next

Overview: v5

- Codenamed hyades
- Objectives:
 - Enable Dependency-Track to handle portfolios spanning hundreds of thousands of projects
 - Improve resilience of Dependency-Track, providing more confidence when relying on it in critical workflows
 - Improve deployment and configuration management experience for containerized / cloud native tech stacks
 - Become platform for more supply-chain related risk tracking
- <https://github.com/DependencyTrack/hyades>
- <https://github.com/DependencyTrack/hyades-apiserver>
- <https://github.com/DependencyTrack/hyades-frontend>

FUNCTIONAL CHANGES

A high-contrast silhouette of a person's head and shoulders, facing right, is positioned on the left side of the frame. The person appears to be holding a traditional triple-beam balance scale. The scale is tilted, with the left pan resting lower than the right. The background is a solid, bright yellow.

POLICY

Policy: CEL Expressions

The screenshot shows a policy configuration interface with the following details:

- Policies:** 1 (highlighted)
- License Groups:** 4
- Vulnerability Policies:** 0
- Create Policy** button
- Search** bar
- WARN Demo** status message
- Name:** Demo
- Operator:** Any
- Violation State:** Warn
- Conditions:** Security
- Expression:** "public-facing" in project.tags && vulns.exists(vuln, vuln.severity == 'CRITICAL' && vuln.cvssv3_vector.matches(".*/AV:N/.*"))
- Buttons:** + (add condition), Limit To, Delete Policy

Policy: CEL Expressions

Policies 1 License Groups 4 Vulnerability Policies 0

+ Create Policy Search

WARN Demo

Name * Operator * Violation State *

Demo Any Warn

Conditions

Expression ✓

```
1 "public undefined field 'severi'  
2   vulns View Problem (F8) No quick fixes available  
3     vuln.severi == 'CRITICAL' &&  
4       vuln.cvssv3_vector.matches(".*/AV:N/.")  
5   ]
```

Security ✖

Limit To Delete Policy

The screenshot shows a policy configuration screen. At the top, there are tabs for 'Policies' (1), 'License Groups' (4), and 'Vulnerability Policies' (0). Below the tabs is a search bar and a refresh button. A yellow 'WARN' status badge with the word 'Demo' is visible. The main form has fields for 'Name' (set to 'Demo'), 'Operator' (set to 'Any'), and 'Violation State' (set to 'Warn'). Under the 'Conditions' section, there is an 'Expression' input field containing a CEL expression. The expression is: 1 "public undefined field 'severi' 2 vulns 3 vuln.severi == 'CRITICAL' && 4 vuln.cvssv3_vector.matches(".*/AV:N/.") 5]. A tooltip for 'View Problem (F8)' appears over the 'severi' field, stating 'No quick fixes available'. To the right of the expression input is a 'Security' dropdown with a red '✖' icon. At the bottom right are buttons for 'Limit To' and 'Delete Policy'.

Policy: Behind the scenes

- Engine entirely powered by CEL expressions
- “Legacy” conditions transparently converted to CEL
- Inputs defined in Protobuf
 - Type-safe
 - Easy to ensure backward-compatibility (checked in CI with buf)
- Only data required by expressions is loaded from database
- Compared to solutions like OPA, no networking overhead
- CEL is not a scripting language (that's good)
- Custom functions to provide domain-specific functionality
 - e.g. vers-powered version range checks, dependency graph traversal

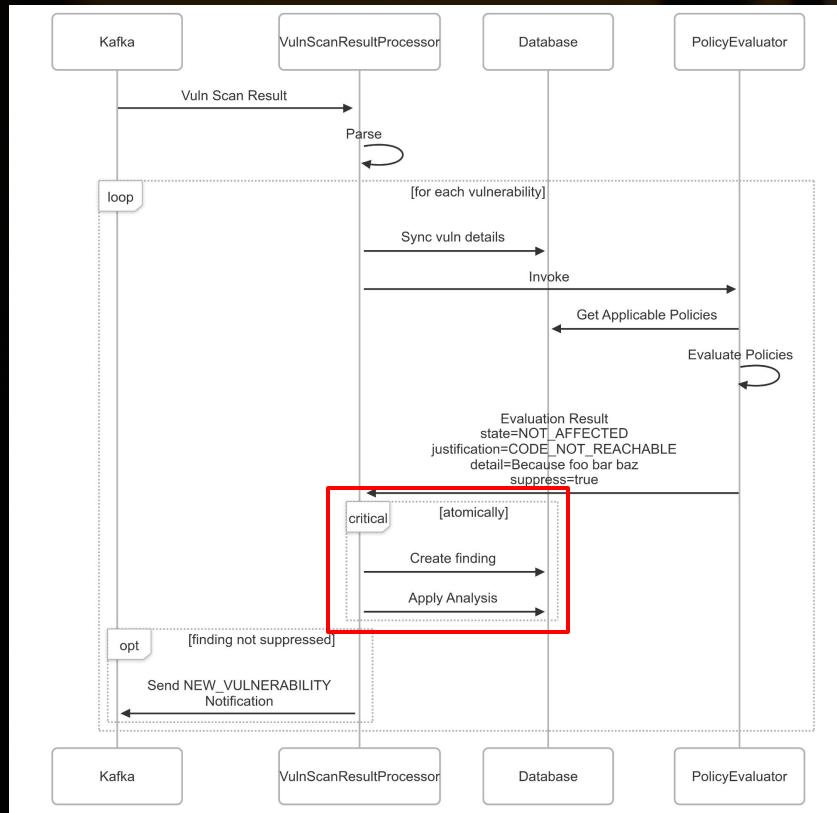
Policy: Automatic Vulnerability Auditing

CEL

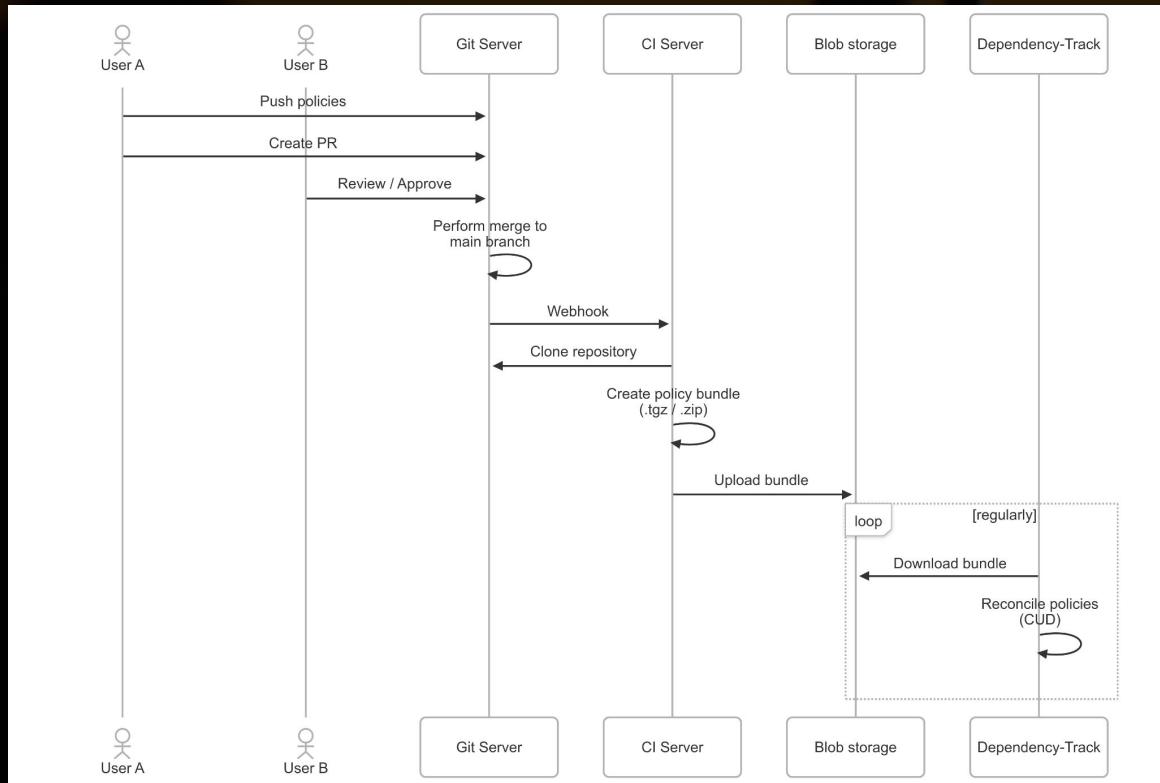
"VEX"

```
apiVersion: v1.0
type: Vulnerability Policy
name: Spring Cloud CVE-2022-41852 Suppression
description: |-
    Suppresses occurrences of CVE-2022-41852 in commons-jxpath,
    when commons-jxpath is introduced through Spring Cloud.
operationMode: APPLY
author: Jane Doe
conditions:
- |-
    vuln.id == "CVE-2022-41852"
    && component.name == "commons-jxpath"
    && component.is_dependency_of(org.dependencytrack.policy.v1.Component{
        group: "org.springframework.cloud",
        version: "vers:maven/>3.1|<3.3"
    })
analysis:
state: NOT_AFFECTED
justification: CODE_NOT_REACHABLE
details: |-
    It was determined that CVE-2022-41852 is not exploitable in Spring Cloud
    components, because the vulnerable code is not used.
suppress: true
```

Vulnerability Policy: Evaluation



Vulnerability Policy: Lifecycle



Vulnerability Policy: VCS Management

hyades-vuln-policy-examples Public

Sponsor Edit Pins Unwatch 2

main 2 Branches 0 Tags

nscurso Merge pull request #1 from DependencyT... 898d291 · 8 months ago 7 Commits

File	Commit Message	Time
.github	Bump actions/checkout from 4.1.7 to 4.2.0	8 months ago
LICENSE	Initial commit	11 months ago
README.md	Initial commit	11 months ago
cve-2022-41852.yaml	Fix missing operationMode	11 months ago
vuln-policy.schema.json	Add schema and validation workflow	11 months ago

<https://github.com/DependencyTrack/hyades-vuln-policy-examples>

Vulnerability Policy: UI

The screenshot displays a user interface for managing vulnerability policies. At the top, there are navigation tabs: 'Policies' (0), 'License Groups' (4), and 'Vulnerability Policies' (1). Below the tabs are buttons for 'Bundle Info' and 'Sync Bundle'. A search bar and a refresh/collapse icon are also present.

The main area features a table with the following columns: Name, Author, Valid From, Valid Until, Created, and Operation Mode. One row is visible:

Name	Author	Valid From	Valid Until	Created	Operation Mode
Spring Cloud CVE-2022-41852 Suppression	Jane Doe	-	-	6 May 2025	APPLY

Below the table, the 'Conditions' section contains the following Groovy script:

```
1 vuln.id == "CVE-2022-41852"
2 && component.name == "commons-jxpath"
3 && component.is_dependency_of(org.dependencytrack.policy.v1.Component{
4     group: "org.springframework.cloud",
5     version: "vers:maven/>3.1|<3.3"
```

The 'Description' section states: "Suppresses occurrences of CVE-2022-41852 in commons-jxpath, when commons-jxpath is introduced through Spring Cloud."

The 'Ratings' section includes a 'Method' field with the value "Method".

The 'Analysis' section shows a 'State' field with the value "NOT_AFFECTED" and a checked checkbox labeled "Suppressed".

Vulnerability Policy: UI

Bundle Info

General

Bundle URL
<https://github.com/DependencyTrack/hyades-vuln-policy-examples/archive/refs/heads/main.zip>

Bundle Hash
046dc1c92f4c9126067401920388494c680f0e59d0eef833f4753d70c9f2ce0d

Created
6 May 2025

Last Synced On
6 May 2025 at 20:26:31

[Management](#)

[Close](#)

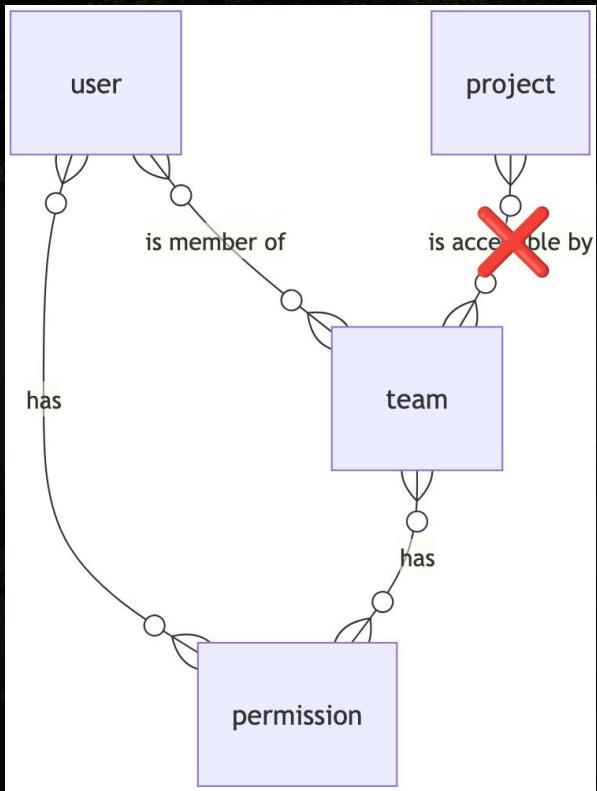
Policy: To Be Done

- CEL should be first choice for conditions
 - Editor to be the central UI element
 - Legacy conditions offered as “templates”
- Vulnerability policies should be manageable via UI
- Improve UX for authoring expressions, editor is still a bit janky
- Offer autocomplete



ACCESS CONTROL

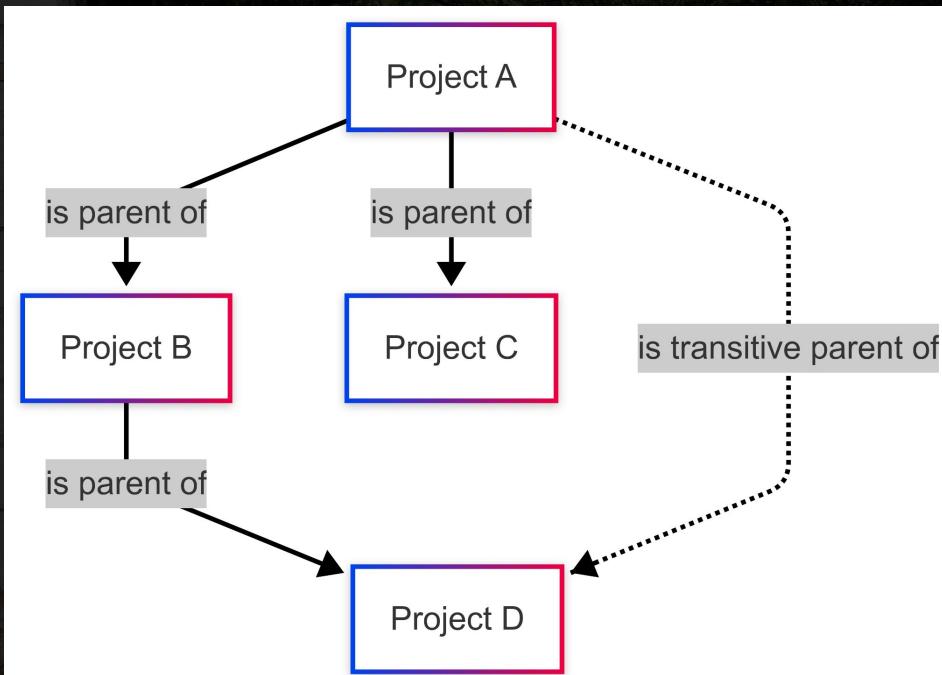
Portfolio Access Control: Refresher



Portfolio Access Control: Closing Gaps

- Feature still in preview in v4, with known gaps
- Goal: production-readiness by v5
- Most known gaps already closed
 - TODO: Filtering of portfolio metrics based on project access
- Inheritance now supported

Portfolio Access Control: Inheritance



Portfolio Access Control: Status

hyades/#1645

*Bring the "Portfolio Access Control" feature
into a final, production-ready state*

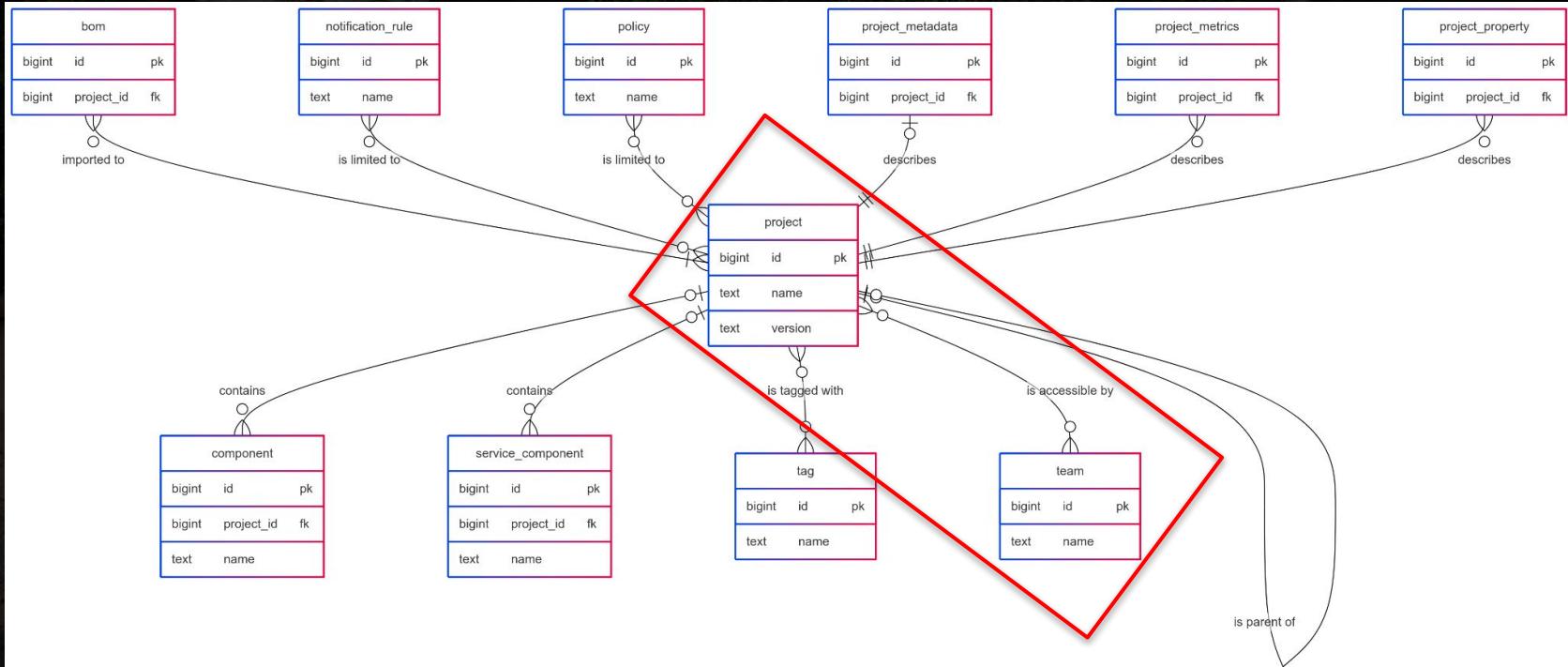


<https://github.com/DependencyTrack/hyades/issues/1645>

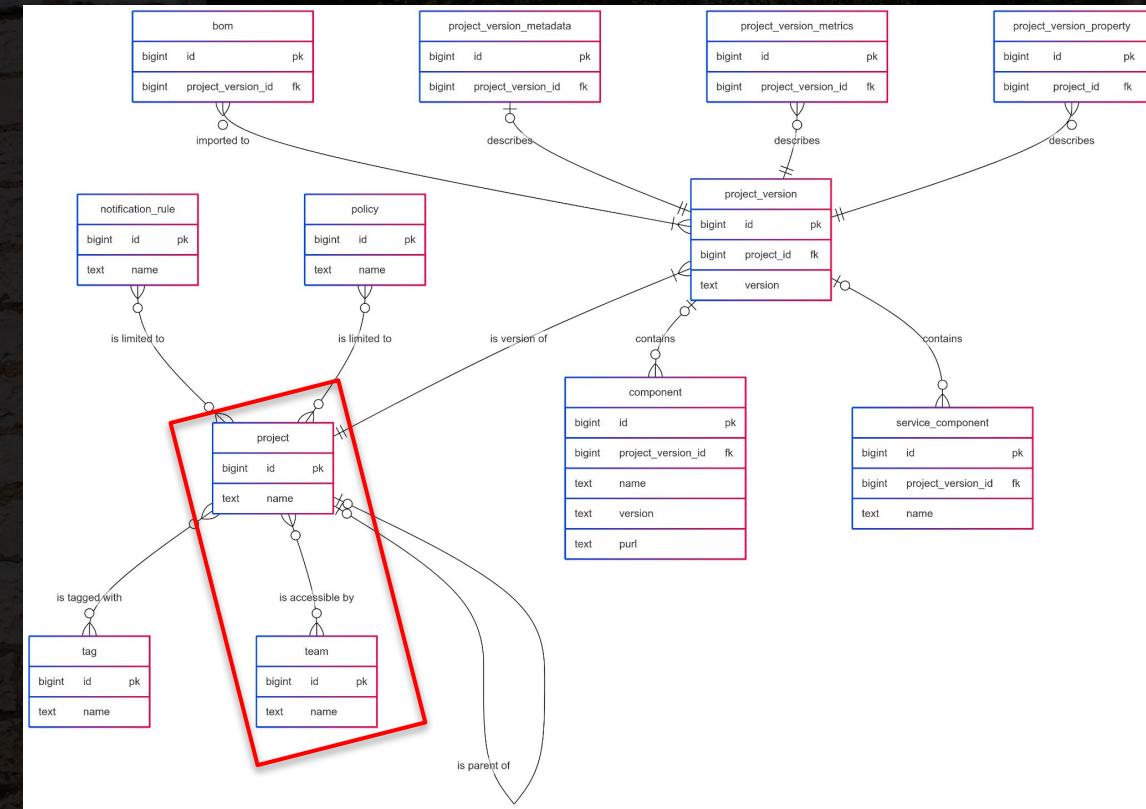
Portfolio Access Control: Granularity

- Current model:
 - Coarse permissions, e.g. PORTFOLIO_MANAGEMENT
 - Global permissions that apply to all accessible projects
 - Teams must be granted access to individual projects
- Vision:
 - Permissions should be more fine-grained
 - It should be possible to scope permissions to specific projects
 - Project access should be assignable on a per-user basis
 - Project access should apply to all versions of a project
- Challenge: Making it perform well
 - Enforcing for individual projects is easy
 - Bulk operations like “list all accessible projects” is not

Project Model



Project Model: Revamped?



Project Model: Revamped?

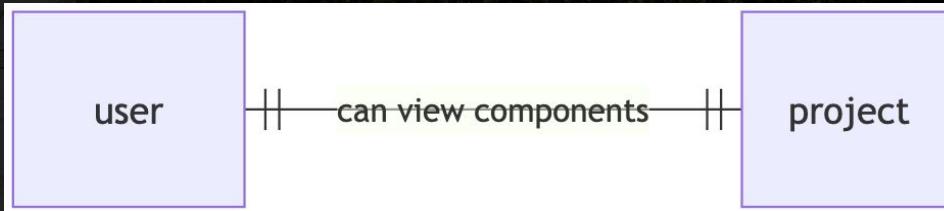
hyades/#1736

*Decouple the concept of project from
project versions*



<https://github.com/DependencyTrack/hyades/issues/1736>

Project-Level Permissions



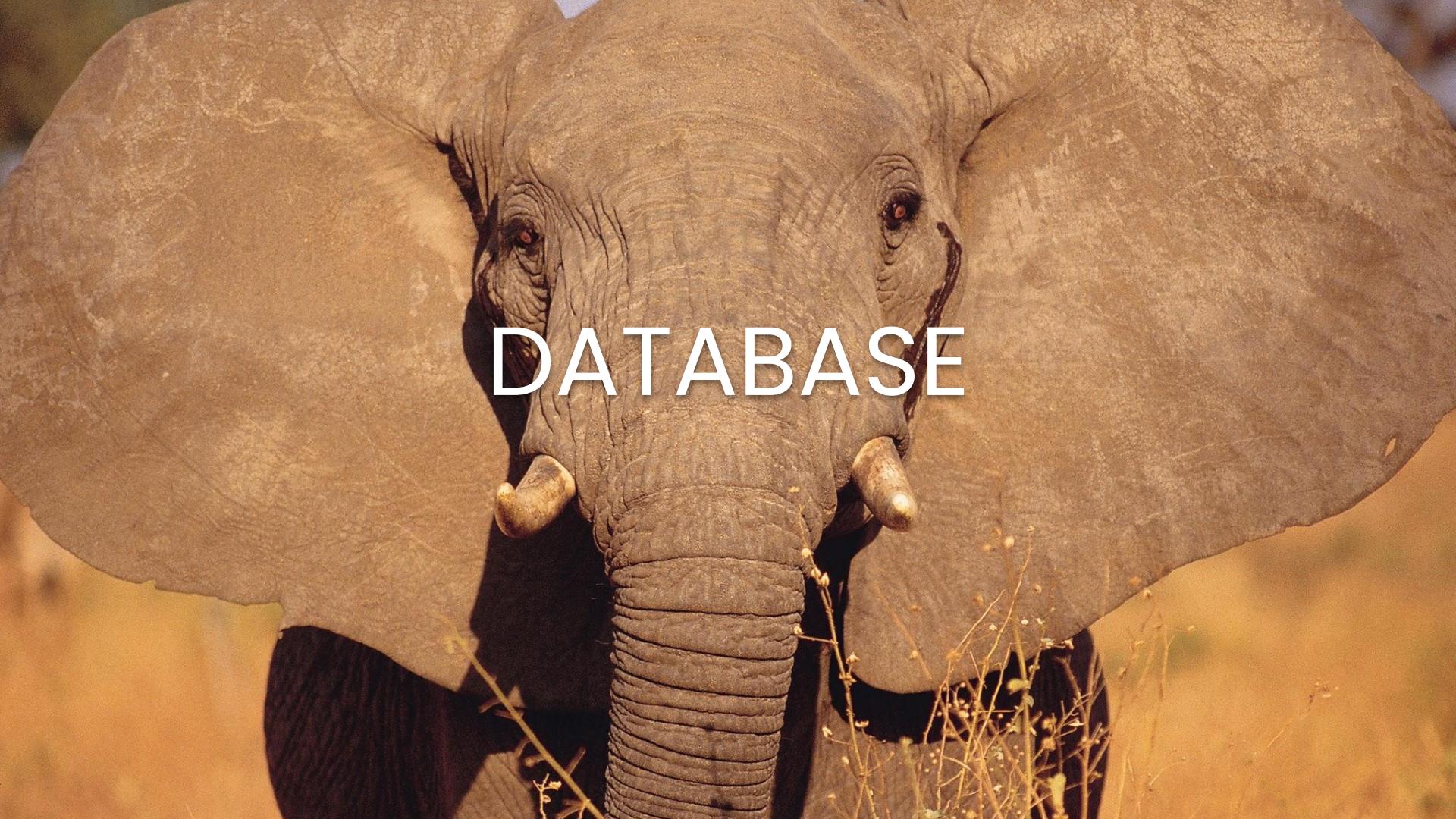
Project-Level Permissions

hyades/#1632
Project-level RBAC



<https://github.com/DependencyTrack/hyades/issues/1632>

NON-FUNCTIONAL CHANGES



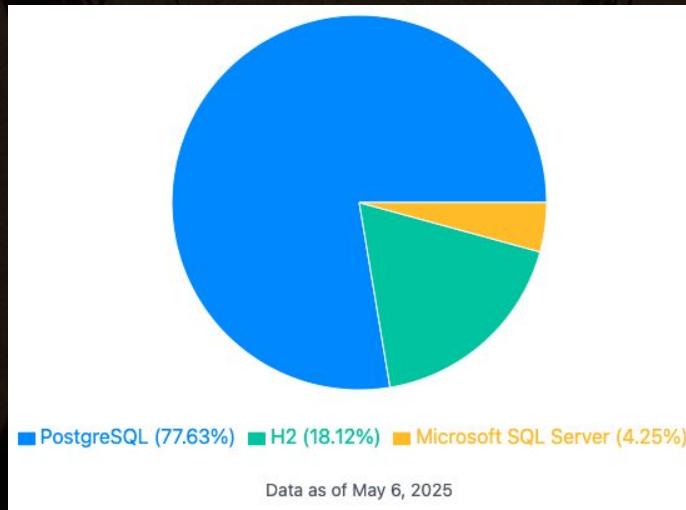
DATABASE

Database

- All-in on Postgres
- Lowest supported version as of now is **13**
 - Might raise to 14 before GA
- Support for H2, MySQL, and MSSQL dropped
 - Drastic reduction in testing effort
 - More opportunities for optimization
- More reliable, changelog-based migrations (Liquibase)
- Ability to use separate set of credentials for migrations
- All tests run against lowest supported version
- Various schema inconsistencies resolved

Database: Migration

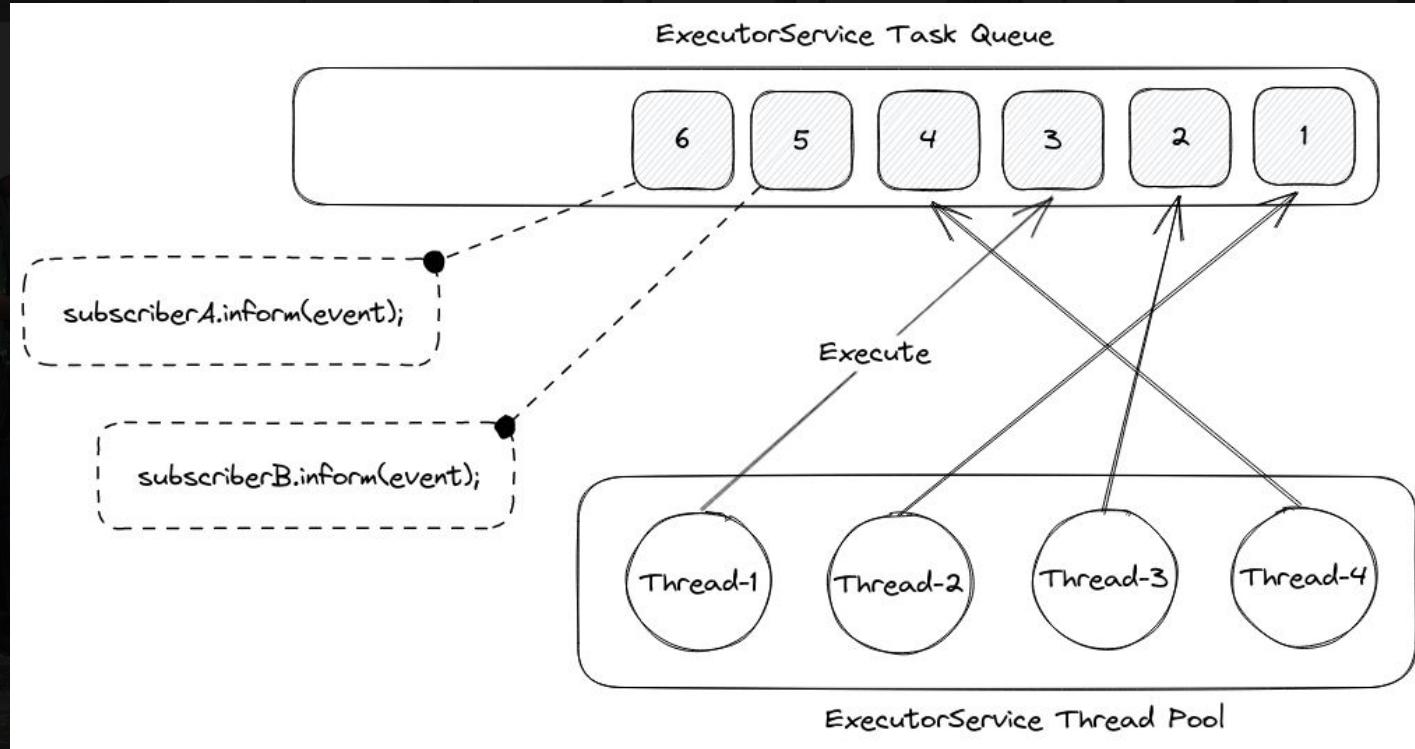
- Tooling for v4 → v5 Upgrade will be provided
- When deploying new v4 instances, choosing Postgres is advisable
- Majority of v4.13+ users already use Postgres



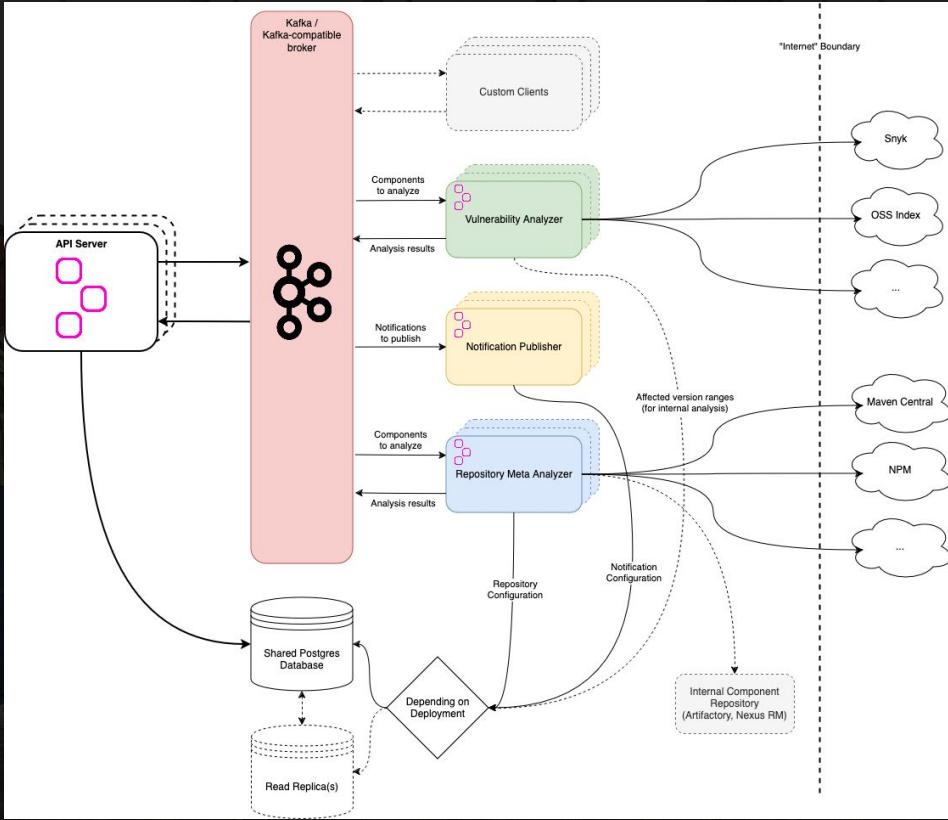
ASYNC PROCESSING



Async Processing: Starting Point



Async Processing: Kafka as Backbone



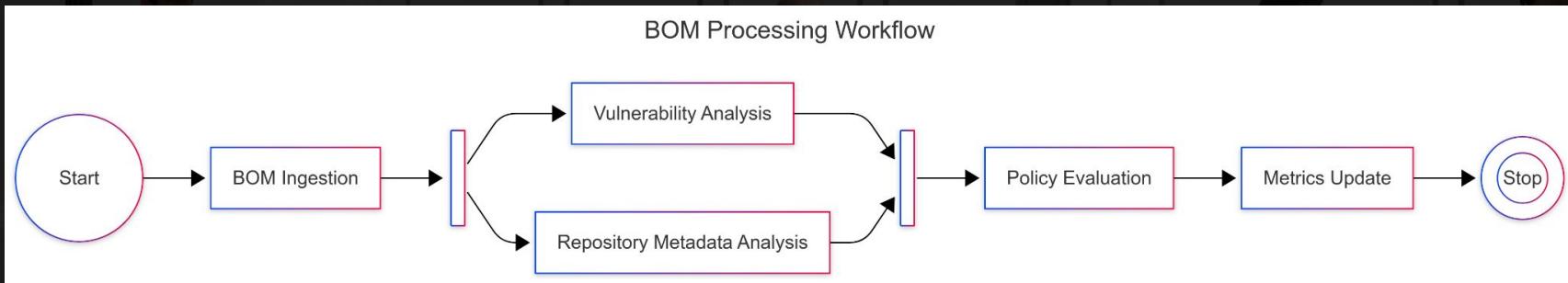
Kafka: Original Intent

- “Event-Driven”
- Reduce load on API server
- Offload async processing to other services
- Leverage stream processing techniques to reduce DB load
- Encourage loose coupling
- Enable horizontal scaling of services
- Ordering guarantees for lock-free concurrency control
- Tap into large, well-supported ecosystem
 - (Hosting Kafka has become a lot more approachable)

Kafka: What Went Well

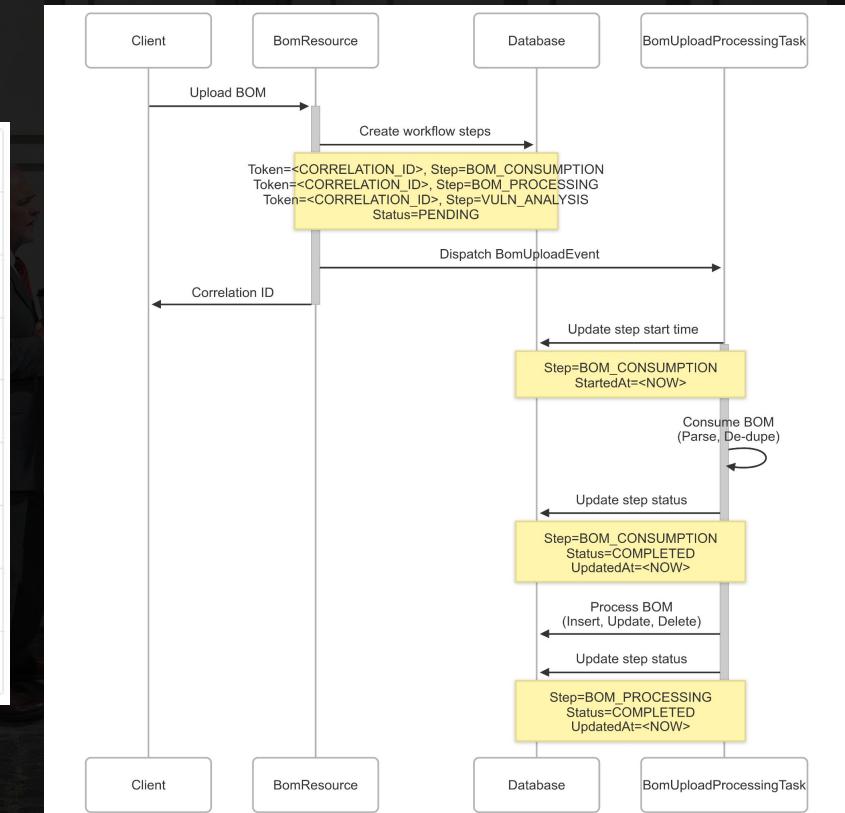
- Loose coupling is convenient for deployments
- Capable client libraries that are easy to integrate
- Great throughput, in particular when coupled with batch consumption
- Ordering guarantees are fantastic for caching
- Good medium to durably publish notifications on

But It Is Workflows, Actually



Tracking Workflow State

Name	Type	Nullable	Example
ID	SERIAL	✗	1
PARENT_STEP_ID	SERIAL FK	✓	0
TOKEN	VARCHAR(36)	✗	484d9eaa-7ea4-4476-97d6-f36327b5a626
STARTED_AT	TIMESTAMP	✓	1999-01-08 04:05:06
UPDATED_AT	TIMESTAMP	✓	1999-01-08 04:05:06
STEP	VARCHAR(64)	✗	METRICS_UPDATE
STATUS	VARCHAR(64)	✗	PENDING
FAILURE_REASON	TEXT	✓	Failed to acquire database connection

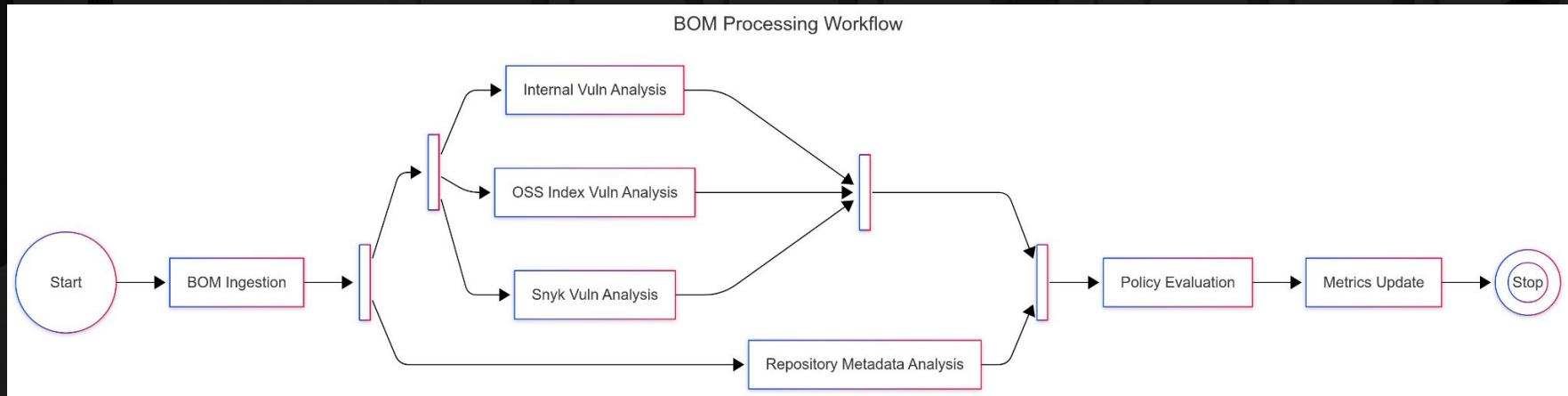


Tracking Workflow State

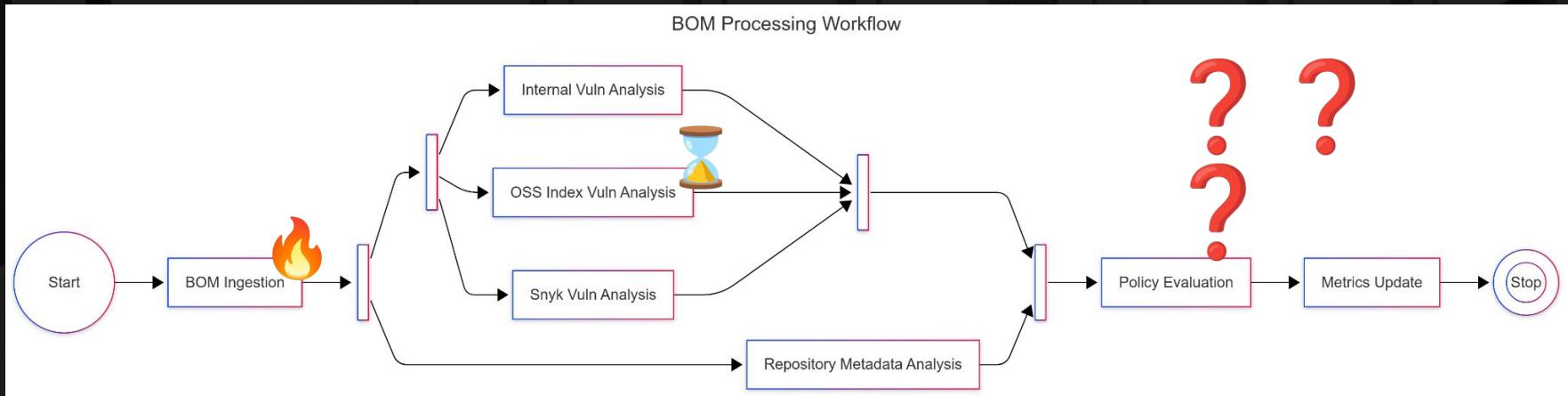
```
GET /api/v1/workflow/token/<CORRELATION_ID>/status
```

```
[  
  {  
    "step": "BOM_CONSUMPTION",  
    "status": "COMPLETED",  
    "startedAt": "1999-01-08 04:05:06",  
    "updatedAt": "1999-01-08 04:05:06"  
  },  
  {  
    "step": "BOM_PROCESSING",  
    "status": "FAILED",  
    "startedAt": "1999-01-08 04:05:06",  
    "updatedAt": "1999-01-08 04:05:06",  
    "failureReason": "Failed to acquire database connection"  
  },  
  {  
    "step": "VULN_ANALYSIS",  
    "status": "CANCELLED"  
  }  
]
```

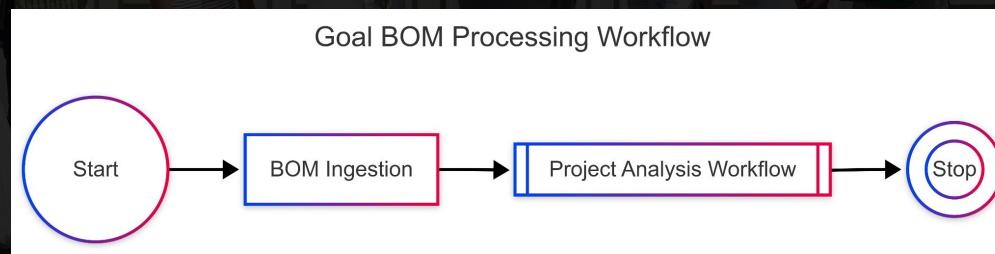
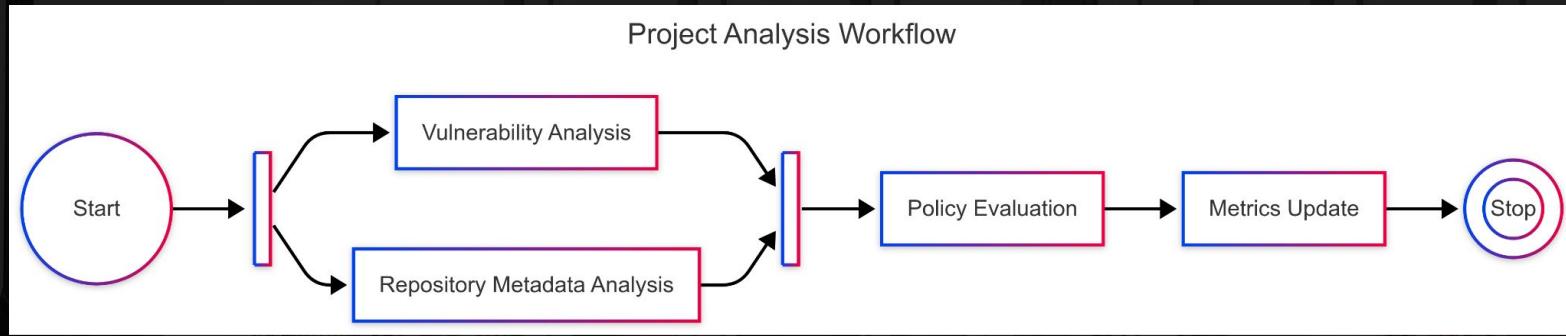
But It Is Complex Workflows, Actually



But It Is Complex Resilient Workflows, Actually



But It Is Complex Resilient Reusable Workflows, Actually



Kafka: Why It Wasn't The Right Choice

- Dual writes threaten reliability of notifications
- Non-blocking retries are hard to achieve
- No support for prioritization
- Message size limitations
- Forces small messages that are processed quickly
- Observability requires special tooling
- End-to-end processes hard to reason about
- Constraints of user environments
- Spotty support for advanced Kafka features like transactions
- Topic management is kind of a mess
- Community support will be hard

Kafka: Letting Go

ADR-001
Drop Kafka Dependency



<https://github.com/DependencyTrack/hyades/blob/bc5ac65943f63a6af84eb35d92fc13a62c10bc27/docs/architecture/decisions/001-drop-kafka-dependency.md>

What We're Doing Instead

- Choreography -> Orchestration
- Leverage "Durable Execution" model
 - Orchestrations are normal application code
- Heavily inspired by Microsoft's DurableTask Framework
 - Which inspired Cadence, which inspired Temporal
- Embedded engine, no additional services to deploy
- Tasks are executed in lightweight virtual threads
- Message passing via Protobuf
- Optimized for Postgres, optimized for throughput
 - Almost all database interactions use batching
 - Buffering of persistence operations
- Can be configured to use separate database

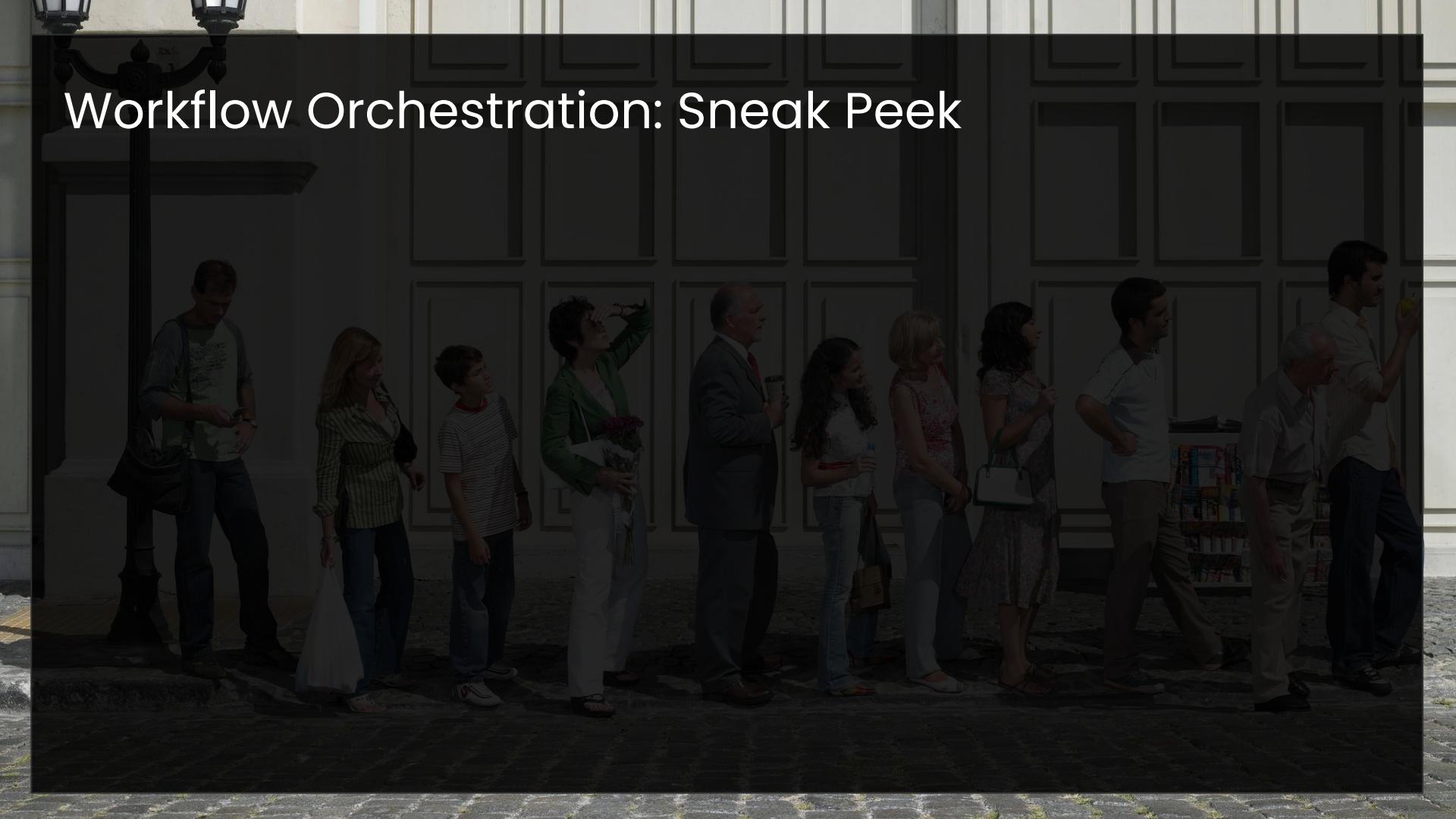
What We're Doing Instead

ADR-002
Workflow Orchestration



<https://github.com/DependencyTrack/hyades/blob/bc5ac65943f63a6af84eb35d92fc13a62c10bc27/docs/architecture/decisions/002-workflow-orchestration.md>

Workflow Orchestration: Sneak Peek



Related: Notification Publishing

ADR-003
Notification Publishing



<https://github.com/DependencyTrack/hyades/blob/bc5ac65943f63a6af84eb35d92fc13a62c10bc27/docs/architecture/decisions/003-notification-publishing.md>

Summary

- Dependency-Track v5 will not depend on Kafka
- Async processes will be more resilient to failure
- (Most) async processes will be observable from the UI

A photograph of a massive, sprawling pile of trash under a vast, cloudy sky. The trash consists of various discarded items, including plastic bags, cardboard boxes, and other unidentifiable debris, stretching across the frame. The sky above is a mix of bright white clouds and deep blue space.

DATA RETENTION

Data Retention

- Metrics account for the most amount of data over time
- Need a mechanism to prevent it from growing indefinitely
- Deleting individual records is problematic for Postgres
 - Table bloat, I/O spikes, Replication overhead
- Solution: table partitioning
 - One partition per day, e.g. PROJECTMETRICS_20250507
 - To enforce retention, simply drop entire partition
 - No table bloat, no I/O spikes, cheap replication
- Retention span is configurable, defaults to 90 days
- Partitions managed by the application
- Will follow same approach for historic workflows

Data Retention

hyades/#1744

Metrics should use partitioned tables



<https://github.com/DependencyTrack/hyades/issues/1744>



CONTRIBUTORS

For Contributors

- Larger changes require an Architecture Decision Record (ADR)
- Docker / Podman required for tests
 - We're making heavy use of testcontainers
- The code base still evolves rapidly, prepare for frequent rebasing



WHAT'S NEXT

What's next

- Complete transition away from Kafka ASAP
 - Most other changes can be done after v5 release
 - The sooner we can stop supporting two release trains the better
- Flesh out access control improvements
 - Might cause breaking changes
 - Migration paths for v4 → v5