

The diagram illustrates the relationship between three entities: Component, Vulnerability, and VulnerableSoftware. Each entity is represented by a 10x10 grid of squares. The 'Component' grid has 10 components, with 10 vulnerabilities (yellow squares) distributed across them. The 'Vulnerability' grid has 10 vulnerabilities, with 10 components (yellow squares) distributed across them. The 'VulnerableSoftware' grid has 10 vulnerable software instances, with 10 vulnerabilities (yellow squares) distributed across them.

The figure displays four 8x8 grids, each representing a different entity type: Project, ServiceComponent, Cpe, and ICpe. Each grid contains 64 cells. The 'Project' grid has 1 yellow cell at (1,1). The 'ServiceComponent' grid has 1 yellow cell at (1,1). The 'Cpe' grid has 2 yellow cells at (1,1) and (1,2). The 'ICpe' grid has 0 yellow cells.

NotificationRule

ProjectProperty

AnalysisComment

Finding

PolicyTest

PolicyConfirmationTest

LicenseGroupTest

CweTest

TagTest

UniversityMainClass

The figure displays a 4x4 grid of resource tests. Each test is represented by a 4x4 grid of colored squares (green and yellow) indicating different states or metrics.

Test Name	Row 1	Row 2	Row 3	Row 4
UserResourceAuthenticatedTest	Green, Yellow, Yellow, Yellow	Yellow, Yellow, Yellow, Yellow	Yellow, Yellow, Yellow, Yellow	Yellow, Yellow, Yellow, Yellow
VulnerabilityResourceTest	Green, Green, Green, Green	Green, Green, Green, Green	Green, Green, Green, Green	Green, Green, Green, Green
OidcResourceAuthenticatedTest	Green, Yellow, Yellow, Yellow	Green, Yellow, Yellow, Yellow	Green, Yellow, Yellow, Yellow	Green, Yellow, Yellow, Yellow
PermissionResourceTest	Green, Green, Yellow, Green	Green, Green, Yellow, Green	Green, Green, Yellow, Green	Green, Green, Yellow, Green
ProjectResourceTest	Green, Green, Green, Green	Green, Green, Green, Green	Green, Green, Green, Green	Green, Green, Green, Green
NotificationRuleResourceTest	Green, Green, Yellow, Yellow	Green, Green, Yellow, Yellow	Green, Green, Yellow, Yellow	Green, Green, Yellow, Yellow
UserResource	Yellow, Yellow, Yellow, Yellow	Yellow, Yellow, Yellow, Yellow	Yellow, Yellow, Yellow, Yellow	Yellow, Yellow, Yellow, Yellow

org.dependencytrack.persistence

- QueryManager
  - PolicyQueryManager
    - ProjectQueryManager
  - VulnerabilityQueryManager
    - ComponentQueryManager
    - MetricsQueryManager
  - DefaultObjectGenerator
    - ServiceComponentQueryManager
    - FindingsQueryManager
  - VulnerableSoftwareQueryManager
    - RepositoryQueryManager
    - LicenseQueryManager
    - RepositoryQueryManager
    - CacheQueryManager
    - CweImporter
  - NotificationQueryManager
    - DefaultObjectGeneratorTest
    - TestManager

The diagram illustrates the layout of the Java IDE's main window, which is divided into several panes. The panes are arranged in a grid-like structure, with some panes containing sub-panes. The panes are labeled as follows:

- IndexManager**: Located at the top left, it contains a large yellow rectangle and a smaller yellow rectangle.
- ComponentIndexer**: Located at the top center, it contains a grid of yellow and green rectangles.
- ProjectIndexer**: Located at the top right, it contains a grid of yellow and green rectangles.
- SearchComponentIndexer**: Located in the middle left, it contains a grid of yellow and green rectangles.
- VulnerabilityIndexer**: Located in the middle right, it contains a grid of yellow and green rectangles.
- CpelIndexer**: Located in the bottom left, it contains a grid of yellow and green rectangles.
- VulnerabilityIndexer (repeated)**: Located in the bottom center, it contains a grid of yellow and green rectangles.
- VulnerabilityIndexer (repeated)**: Located in the bottom right, it contains a grid of yellow and green rectangles.
- SearchManager**: Located at the bottom left, it contains a grid of yellow and green rectangles.
- LicenseIndexer**: Located in the bottom center, it contains a grid of yellow and green rectangles.
- VulnerabilityIndexer (repeated)**: Located in the bottom right, it contains a grid of yellow and green rectangles.
- VulnerabilityIndexer (repeated)**: Located in the bottom right, it contains a grid of yellow and green rectangles.
- VulnerabilityIndexer (repeated)**: Located in the bottom right, it contains a grid of yellow and green rectangles.

The dashboard displays the following sections and their visual representations:

- NugetMetaAnalyzer**: A bar chart with two yellow bars and two green bars.
- GoModMetaAnalyzer**: A bar chart with two yellow bars and two green bars.
- CargoMetaAnalyzer**: A bar chart with two yellow bars and two green bars.
- GemMetaAnalyzer**: A bar chart with two yellow bars and two green bars.
- HexMetaAnalyzer**: A bar chart with two yellow bars and two green bars.
- PyPiMetaAnalyzer**: A bar chart with two yellow bars and two green bars.
- ComposerMetaAnalyzer**: A bar chart with two yellow bars and two green bars.
- ComposerMetaAnalyzer/Fast**: A bar chart with two yellow bars and two green bars.
- NpmMetaAnalyzer**: A bar chart with two yellow bars and two green bars.
- MavenMetaAnalyzer**: A bar chart with two yellow bars and two green bars.
- MetaModel**: A central green box with the text "MetaModel" and four small white squares below it.

The diagram illustrates a CI/CD pipeline with the following tasks and their dependencies:

- OssIndexAnalysisTask** (yellow) depends on **BaseComponentsAnalyzerTask** (yellow).
- NpmAuditAnalysisTask** (yellow) depends on **BaseComponentsAnalyzerTask** (yellow).
- BaseComponentsAnalyzerTask** (yellow) depends on **InternalAnalysisTask** (yellow).
- AuditWebAppTask** (yellow) depends on **InternalAnalysisTask** (yellow).
- InternalAnalysisTask** (yellow) depends on **CodeQualityTask** (yellow).
- CodeQualityTask** (yellow) depends on **ScanTask** (green).
- ScanTask** (green) is the final task in the pipeline.

ComponentVersion	DateUtil	HashUtil
DateUtilTest	PurUtil	CompUtil
HashUtilTest	HttpUtil	XmlUtil
		JsonUtil

GitHubSecurityAdvisory

Advisory Results

This bar chart displays 40 individual advisory results. The bars are arranged in a grid-like fashion, with heights varying significantly, representing a distribution of values. The chart is titled 'Advisory' and 'AdvisoryResults'.

The diagram illustrates a 3D volume represented by a 3x3 grid of cells. The cells are colored orange or green, and each contains a title and a set of colored squares (green, yellow, orange, red) representing different data series.

Cell Color	Cell Title	Cell Content (Squares)
Orange	AnalysisRequest	5 green squares
Orange	ValidationRequest	5 green squares
Green	TransformationRequest	5 green squares
Orange	AnnotationRequest	1 yellow square, 1 orange square, 1 red square
Green	SegmentationRequest	2 green squares, 1 orange square, 1 red square
Green	ClassificationRequest	5 green squares
Green	ClassificationRequest	5 green squares
Orange	ClassificationRequest	5 green squares
Green	ClassificationRequest	5 green squares

The diagram illustrates the execution of tasks in a thread pool. At the bottom, a yellow box labeled 'TaskScheduler' is shown. Above it, a grid of colored rectangles represents tasks. The tasks are organized into three main sections: 'VulnDbSyncTask' (blue), 'NistMirrorTask' (orange), and 'DatabaseSyncTask' (green). The tasks are arranged in a way that shows the flow of execution from the scheduler to the worker threads. The tasks are represented by colored rectangles: blue for 'VulnDbSyncTask', orange for 'NistMirrorTask', and green for 'DatabaseSyncTask'. The tasks are organized into a grid, with the 'TaskScheduler' at the bottom and the tasks above it. The tasks are arranged in a way that shows the flow of execution from the scheduler to the worker threads.

The diagram shows a 3x3 grid of colored squares representing a 3D volume. The top row has four yellow squares, the middle row has three green squares, and the bottom row has two orange squares. Each square contains a small icon representing a different object or state.

The diagram illustrates a grid layout for a website or application. It consists of 12 rectangular blocks arranged in a 3x4 grid. The blocks are color-coded: orange, yellow, and green. Each block contains a small icon representing a different type of media or content. The icons include a document, a video camera, a film strip, a speech bubble, a magnifying glass, a play button, a book, a gear, a person, a star, a heart, and a share icon. The layout is designed to be flexible and adaptable to different content types and sizes.

org.dependencytrack.integrations.kenna

org.dependencytrack.parser.nvd

Component Report Sustainability

Component Report Sustainability

org.dependencytrack.notification

NotificationPublisherTest	NotificationPublisherTest
---------------------------	---------------------------

```

graph TD
    A[org.springframework.boot.autoconfigure.condition.Condition...] --> C[CweResolverTest]
    B[org.springframework...] --> D[...]
    C --> E[org.springframework.boot.autoconfigure.condition...]
    D --> F[org.springframework...]
  
```

The image shows a collection of colorful sticky notes (orange, yellow, green, and blue) with various text and diagrams. The notes are arranged in a grid-like fashion, with some overlapping. The text on the notes includes:

- Orange notes:**
  - Top left: "1998-2001" (partially visible).
  - Top right: "1998-2001" (partially visible).
  - Middle left: "logback integration".
  - Bottom left: "logback integration upgrade".
- Yellow notes:**
  - Middle right: "logback integration upgrade".
  - Bottom right: "logback integration upgrade".
- Green note:**
  - Bottom left: "logback integration upgrade".
- Blue note:**
  - Bottom right: "logback integration upgrade".

The diagrams on the notes include:

- A diagram showing a box labeled "logback" with a smaller box labeled "logback-core" inside it.
- A diagram showing a box labeled "logback" with a smaller box labeled "logback-core" inside it, and a box labeled "logback-classic" below it.
- A diagram showing a box labeled "logback" with a smaller box labeled "logback-core" inside it, and a box labeled "logback-classic" below it.

The diagram illustrates the lifecycle of a `ManagedHttpConnectionFactory` and its associated components. It is divided into two main sections: a left section for the factory and a right section for the client pool.

- Left Section (Factory Lifecycle):**
  - ManagedHttpConnectionFactory:** The top box, representing the factory object.
  - Factory State:** Below the factory box, a set of colored squares (yellow and green) represents the state of the factory's managed connections. A small green bar labeled "idle" indicates the state of the idle connections.
  - ManagedHttpConnectionFactoryPool:** The bottom box, representing the pool of factory objects.
- Right Section (Client Pool Lifecycle):**
  - HttpClientPool:** The top box, representing the pool of HTTP clients.
  - ManagedHttpClient:** A green box representing a single managed HTTP client.
  - Client State:** Below the client pool box, a set of colored squares (yellow and green) represents the state of the clients. A small green bar labeled "idle" indicates the state of the idle clients.

Arrows indicate the flow of objects and state changes between these components, showing how the factory manages the lifecycle of the clients and how the pool manages the lifecycle of the factories.

org.dependencytrack

ResourceTest

Progress bar with 10 segments, 8 of which are filled.

[illegible]

top-left	top-right	top-center
middle-left	middle-center	middle-right
bottom-left	bottom-center	bottom-right