Carnegie Mellon University
Software Engineering Institute

Integration of Automated Static Analysis Alert Classification and Prioritization with Auditing Tools: Special Focus on SCALe

Lori Flynn
Ebonie McNeil
David Svoboda
Derek Leung
Zachary Kurtz
Jiyeon Lee

May 2019

TECHNICAL REPORT
CMU/SEI-2019-TR-007

CERT Division

http://www.sei.cmu.edu

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

This report was prepared for the SEI Administrative Agent AFLCMC/AZS 5 Eglin Street Hanscom AFB, MA 01731-2100

DM18-1040

Table of Contents

List of Figures

List of Tables

Acknowledgments

We thank collaborating and co-funding organizations (anonymous at their request) for providing feedback on our research ideas and for testing our prototypes.

Abstract

This report summarizes technical progress and plans as of late September 2018 for developing a system to perform automated classification and advanced prioritization of static analysis alerts. Many features and fields have been added to the Source Code Analysis Laboratory (SCALe) static analysis alert auditing tool to support this functionality. This report describes the new features and fields, and how to use them. It also describes the plan to connect this enhanced version of SCALe to an architecture that will provide classification and prioritization via API calls, and provides the API definition that has been developed. A prototype that instantiates the architecture is being developed; future work will complete the prototype and integrate the latest version of SCALe with it.

1 Introduction

This report summarizes technical progress and plans for developing a system to perform automated classification and advanced prioritization of static analysis alerts [Flynn 2018b].

Completed progress includes enhancements to an existing static analysis alert auditing tool, the Source Code Analysis Laboratory (SCALe). It also includes the development of an architecture that supports alert classification and prioritization that auditing tools can use with an API definition we developed. The team has also made significant, but incomplete, progress in developing a prototype instantiation of the architecture.

In future work, SCALe will interact with the other parts of the architecture and three other servers (described in Section 6 and Appendix A: Rapid Models API Definition) will provide external functionality. Currently, the enhancements to SCALe provide advanced prioritization and much of the classifier functionality that will be required in SCALe for a fully integrated system.

The planned system will provide an architecture with APIs and an open source prototype system that has the following benefits to users:

- They can quickly start to use automated classifiers for static analysis alerts. The system *will not* require
  - a labeled audit archive to be provided ahead of time since it uses test suites in a new way [Flynn 2018a]
  - a statistics expert
  - users to create their own frameworks for using classifiers
- They can quickly apply formulas that prioritize static analysis alerts by using factors they care about. These prioritization formulas can combine various fields, including classifier-derived confidence, with mathematical operators.
- They can employ the API definition to build upon the original prototype system, enabling the use of additional flaw-finding static analysis tools, code metrics tools [CCSM 2018, Yin 2018], adaptive heuristics, classification techniques, and so forth.

2 Terminology

Table 1 provides definitions for terminology used in this report.

Table 1: Definitions of Terms Used in this Report

Term	Definition
alert	A warning from a static analysis tool about a possible code defect
alert prioritization	Ordering of static analysis alerts in the auditor GUI
audit determination	Decision (also called a *verdict*) made by a human reviewer about the validity of an alert from a static analysis tool, such as True or False. This determination may be made with respect to either a message from the tool itself or a coding taxonomy to which the alert is mapped. The decision may be made specifically about a static analysis alert or it may be made specifically about a meta-alert. (The latter could potentially make a decision for multiple alerts.)
Checker	In this report, short for *checker ID*. (In other contexts, *checker* refers to the algorithm or code that performs inspection of code for a particular type of flaw by a particular static analysis tool.)
checker ID	An identifier for a particular type of alert from a particular type of tool. Flaw-finding static analysis tools often provide a short string ID for a checker (e.g., memset-ValueOutOfRange is one checker ID in the Cppcheck[1] tool [Marjamäki 2018]). We identify checkers from other tools with regular expressions.
coding taxonomy	A named set of coding rules, weaknesses, standards, or guidelines. Examples include MITRE's Common Weakness Enumeration (CWE) and the CERT Secure Coding Standards [MITRE 2018, SEI 2018b]. Each rule or weakness is considered to be a single condition.
condition	A constraint or property of validity with which code should comply. Flaw-finding static analysis tools try to detect whether code violates conditions. Each rule or weakness in a coding taxonomy is considered a single condition. For example, INT31-C is a condition from the CERT Secure Coding Standards [SEI 2018b].
CWE	MITRE's Common Weakness Enumeration (CWE) [MITRE 2018]
determination	Short for *audit determination*
diagnostic	Alert
fused alert	A meta-alert that has multiple alerts. The term also describes such a meta-alert's associated data and the GUI representations of such a meta-alert and its associated data.
GUI	Graphical user interface
meta-alert	A new data structure that has a unique entry for each unique tuple (line number, file path, condition) any alert is mapped to. If multiple alerts share the same tuple, then the meta-alert is mapped to multiple alerts. A meta-alert can be mapped to a single alert.
prioritization	Short for *alert prioritization*
taxonomy	Short for *coding taxonomy*
verdict	Static analysis alert determination, such as True or False

[1] http://cppcheck.sourceforge.net/

3 SCALe Development

The CERT SCALe tool provides a framework for auditing static analysis alerts [SEI 2018a]. It features a graphical user interface (GUI) front end for analysts and a back end that saves and exports auditing projects to data archives. The GUI enables analysts to see a list of alerts (warnings about potential code defects) from static analysis tools, examine source code associated with each alert, and mark determinations (e.g., True or False) for the alerts.

This report does not detail the full design of SCALe. Instead, this section focuses on enhancements we made to the previous version of SCALe to enable automated alert classification and advanced functionality for alert prioritization.

Note that table names in the SCALe database are given in **boldface** and field names are given in `fixed-width` typeface.

3.1 Previous Versions of SCALe

3.1.1 SCALe Version 1

SCALe version 1 (v1) is the initial version of SCALe. It only mapped alerts to CERT Secure Coding Standards and mapped an alert to, at most, one condition (a particular code flaw identified by a code flaw taxonomy, such as INT31-C[2]) [SEI 2018b]. The exported SCALe project database had a simple format. Each alert could have only one determination (a single verdict field in the **Diagnostics** table), for one verdict per alert.

Simple filters were provided in the GUI for auditors to limit the subset of alerts viewed (e.g., to a single CERT rule). A simple prioritization scheme allowed ordering based on single features of the alert (e.g., alert ID or `Remediation Cost`) and on one multiple-feature-derived value `Priority` (derived from `Severity`, `Remediation Cost`, and `Likelihood`) that is hard-coded for each CERT coding rule in its **Risk Assessment Table** (e.g., `Priority` is 6 for INT31-C).

[2] https://wiki.sei.cmu.edu/confluence/display/c/INT31-C.+Ensure+that+integer+conversions+do+not+result+in+lost+or+misinterpreted+data

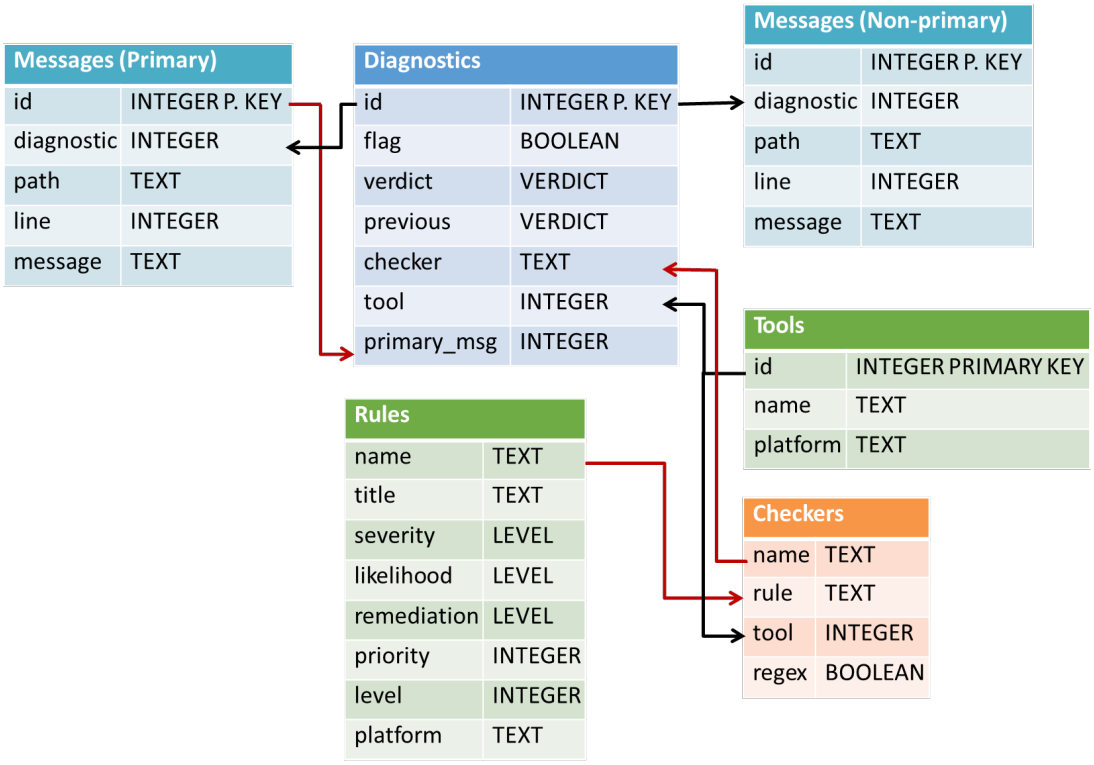


Figure 1: SCALe v1 Exported Database Format

3.1.2 SCALe Version 2

SCALe version 2 (v2) was released publicly on GitHub in August 2018 (the first-ever public release of SCALe). Prior to that, it was distributed only to research project collaborators. It contains a subset of the features of SCALe version 3 that are detailed below. Specifically, SCALe v2 supports the CWE taxonomy; alert fusion; and new and modified fields for basic determinations, supplemental determinations, and notes.

3.2 Database Enhancements

We made the following modifications to the SCALe databases to include new features important for prioritization and classification integration:

- *Inclusion of the CWE taxonomy* [MITRE 2018]. This enables the use of test suites based on Common Weakness Enumeration (CWE) to automatically generate labeled data for classifiers. CERT standards continue to be supported.
- *Alert fusion.* This reduces the effective number of (alert, condition) tuples to be audited. Tuples on any given line of code get fused into a single unit for auditing purposes. We use a new data structure called a meta-alert; at least one is associated with every alert, one for every condition that alert is mapped to. Alert fusion refers to multiple alerts being mapped to a single meta-alert (this happens when they share the same line number, file path, and mapped condition). A meta-alert can be mapped to a single alert (during the time it maps to only one alert, there is no alert fusion associated with it).

- *Support for additional software metrics used by classifiers*
- *New and modified fields* in accordance with our paper *Static Analysis Alert Audits: Lexicon & Rules* [Svoboda 2016]. They include improved auditor determinations, new supplemental determinations, and a new notes field. These standardized labels and fields work with the CERT auditing rules to enable precise and consistent audit determinations. Thus, they should improve the accuracy of classifiers based on the archive data resulting from the audit.
- *New tables for classification and prioritization.* These tables record schemes for various types of prioritization and classification, as well as scores for their resulting priority and confidence. The scores will be used in the GUI to order alerts for prioritized auditor work.
- *New fields for classification and prioritization.* Pre-existing **Project** tables have two new fields, `last_used_confidence_scheme` and `last_used_priority_scheme`. These fields enable the auditor to determine how the confidence and priority values were calculated, even if the SCALe project is exported and then re-imported into a different instance of SCALe. The **Project** tables also have a new field, `current_classifier_scheme`, that records the scheme for the most recent type of classifier that was created. (This is the scheme that was last used to create a classifier, although the classifier may not yet have been run on alerts that need to be labeled.)
- **Determination history**. This is a new table that stores the history of determinations for a meta-alert, including primary and supplemental verdicts, notes, flag, and timestamp. This data can be used to develop classifiers that use features such as determination changes.

The database design in Figure 2 shows the new format of an exported SCALe project (in sqlite3 database format).

The new database format has more fields and tables to incorporate new features. For example, we previously included the determinations in the **Diagnostics** table; they are now mapped to `MetaAlert` IDs. We made this change because an alert can now map to multiple conditions with multiple taxonomies; additionally, a new way to handle mappings now allows a checker ID to map to multiple conditions within a single taxonomy.

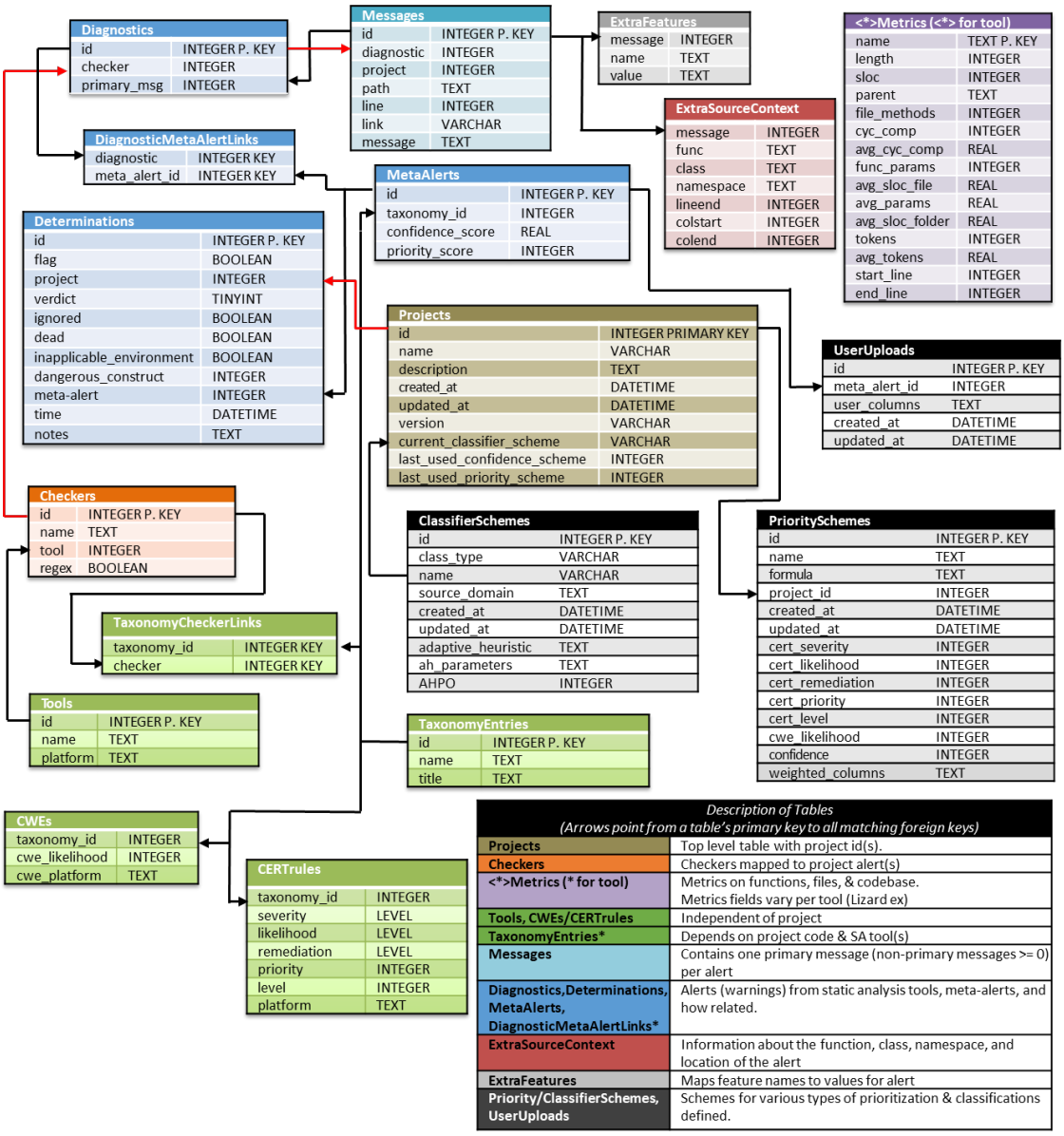


Figure 2: SCALe v 3.0.0.0 Exported Database Format

Figure 2 shows a diagram of the new database tables. Each table has a key that consists of one or more fields that are unique for each record in the table. When there is only one such field, it is marked `PRIMARY KEY` (or `P. KEY`). When there are two or more fields, they are marked `KEY`.

Some fields serve as foreign keys that link to a primary key in another table. In these cases, an arrow runs between tables from the primary key to the foreign key. The foreign key can also act as a primary key.

The **<*>Metrics** table schema varies for each metrics tool. Each metrics tool outputs values for a different set of fields, which vary greatly in number. For example, if the Lizard tool was used in the SCALe project, the **<*>Metrics** table from Figure 2 would be named **LizardMetrics** [Yin 2018]. SCALe version 2.0 and 3.0 will create metrics tables to store output from three tools: Lizard, CCSM, and a third proprietary tool we will call AnonymizedTool here. Accordingly, the **<*>Metrics** tables for these tools will be named **LizardMetrics**, **CCSMMetrics**, and **AnonymizedToolMetrics**.

3.3 New and Modified Tables

- **MetaAlerts** links to a condition (CERT rule or CWE). Each `MetaAlert.id` is unique to a path, line, and condition tuple. The same (path, line) tuple is specified in all of its associated messages. Verdict information for each `MetaAlert.id` is stored in the **Determinations** table.
- **TaxonomyEntryCheckerLinks** links checker IDs with condition IDs (which previously supported only CERT rule IDs). This manages the many-to-many relationships between checkers and conditions.
- **DiagnosticMetaAlertLinks** links diagnostic (unfused alert) IDs with meta-alert IDs. This manages the many-to-many relationship between diagnostics and meta-alerts. (A single alert may map to multiple conditions and therefore to multiple meta-alerts. A single meta-alert may map to multiple alerts, e.g., alerts from multiple tools for the same code flaw.)
- **TaxonomyEntries** contains two fields common to CWE and CERT rules. The `name` field contains the short name of the condition (e.g., INT31-C). The `title` field contains the condition's title. For a CWE, the title consists of everything after the short name and colon on the title of the CWE web page. For a CERT guideline, the title contains everything after the short name and period. The following are examples of titles:
  - For INT31-C, the title is "Ensure that integer conversions do not result in lost or misinterpreted data."
  - For CWE-190, the title is "Integer Overflow or Wraparound."
- **CWEs** integrates the CWE taxonomy.
- **CERTrules** implements the CERT rules that were previously included in SCALe but are now translated into our new format for extensible taxonomy additions.
- **PrioritySchemes** records schemes for various types of prioritization of alerts. It includes different weights for commonly considered factors, a field for weighting user-added factors, and a field for a custom formula used in prioritization. To support taxonomy extensibility during project creation, we add entries to this table for the weight of each taxonomy-specific field (e.g., `CERTrules.severity`, `CERTrules.likelihood`, `CWEs.cwe_likelihood`). Uploads of user-specific metrics can now be associated with a weight in the prioritization scheme. The classifier-generated confidence can also be used in the prioritization formula.
- **UserUploads** enables the user to upload custom field names to be used in the prioritization of alerts.
- **ClassifierSchemes** records classification schemes, including the type (if any) of automated hyper-parameter generation (e.g., caret) used prior to classifier development.

3.4 Example

To see the relationships between alerts, meta-alerts, and taxonomies in the new database, consider this example:

Suppose that two tools, Tool A and Tool B, both have checkers that detect integer overflow. This is addressed by two CERT rules (INT30-C and INT32-C) and two MITRE CWEs (125 and 190). The checker IDs are provided by the tool vendors, not by regular expressions.

The **Checkers** table contains the following entries. The column with bright blue font is not actually part of this table.

id	name	tool (here with name, not in actual Checkers table)	regex
1	integer__OVERFLOW	23 (Tool B)	False
2	int_overflow	24 (Tool A)	False
3	buffer_overflow	24 (Tool A)	False

The **TaxonomyCheckerLinks** table contains the following entries:

taxonomy_id	TaxonomyEntries.name	checker
2	INT30-C	1
4	INT32-C	1
1	CWE-125	1
3	CWE-190	1
2	INT30-C	2
4	INT32-C	2
1	CWE-125	2
3	CWE-190	2
5	CWE-121	3

The `taxonomy_id` field provides an index to this information in another table.

Suppose that line 5 of `file.c` triggers both checkers. The **Diagnostics** table (which no longer contains verdict information) is otherwise unchanged. (The columns with bright blue font are not actually part of the Diagnostics table. The `checker` and `primary_message` fields provide indexes to this information in other tables.)

id	checker	tool	Path	Line
1	integer__OVERFLOW	Tool B	file.c	5
2	int_overflow	Tool A	file.c	5
3	buffer_overflow	Tool A	file.c	99

When the database is created, the fusion process adds the following to the **MetaAlerts** table. (The columns with bright blue font are not actually part of the table. The fields `id` and `taxonomy_id` provide indexes to this information in other tables.) Empty fields in the example below would actually be filled with the known value (`taxonomy_id`) or initialization values.

id	flag	verdict	ignored	dead	inapplicable_environ-ment	dangerous_construct	note	previous	taxonomy_id	confidence_score	priority_score	Condition	Path	Line
1									2			INT30-C	file.c	5
2									4			INT32-C	file.c	5
3									1			CWE-125	file.c	5
4									3			CWE-190	file.c	5
5									5			CWE-121	File.c	99

This fusion means that an auditor can make up to four verdicts on source code line 5, which depends on the details of the four taxonomy items. The auditor can make another verdict on source code line 99.

Finally, the **DiagnosticMetaAlertLinks** table looks like this:

diagnostic	meta_alert_id
1	1
1	2
1	3
1	4
2	1
2	2
2	3
2	4
3	5

3.5 Fusion

The **MetaAlert** table contains entries for every alert and every condition to which that alert is mapped. If multiple alerts can be fused (i.e., they share the same line number, file path, and mapped condition), then the **MetaAlert** entry is mapped to multiple alerts. A **MetaAlert** entry can be mapped to a single alert. In that case, it serves as a placeholder for the potential future fusion of single alerts that currently have no other alerts to be fused with. Audit determinations are now made per meta-alert (per **MetaAlert** entry), not per alert (per **Diagnostics** entry).

The GUI has been changed to show meta-alerts, which saves the auditor time over auditing only unfused alerts. All alert messages are shown in an expanded view when the auditor selects the meta-alert in the GUI. This provides the auditor with information from the fused alerts in one view.

Counting meta-alerts provides a more accurate measure of the length of the alerting queue than counting unfused alerts. Alert fusion accounts for redundant alerts issued by the same tool or distinct tools. Our verdicts apply to meta-alerts; therefore, we now associate verdicts with meta-alerts rather than unfused alerts. Each alert inherits the verdict of its associated meta-alert; fused alerts cannot have different verdicts from their component alerts.

We changed the architecture to enable associating a single checker with multiple conditions (meaning to multiple `TaxonomyEntries.id`[s]). Previously, SCALe limited mappings to only one rule per checker. If two diagnostics match in path and line number, a meta-alert should be created for each checker's matched taxonomy items (e.g., each CERT rule or each CWE). If they share one of these taxonomy items, they will share a meta-alert.

- For a given `MetaAlerts.id`, there can be multiple **DiagnosticMetaAlertLinks** table entries (all with the same `DiagnosticMetaAlertLinks.meta_alert_id`, but with different `DiagnosticMetaAlertLinks.diagnostic`(s)).
- For every `MetaAlerts.id`, there is at least one `Diagnostics.id`. There may be more than one.
- For a given `Diagnostics.id`, there is at least one `MetaAlerts.id`. There may be more than one.
- A `Diagnostics.id` is unique to a given alert from a tool. It is uniquely an alert for a specific line, file, checker ID, *and* message text. (If a tool provides secondary messages, it may output multiple alerts with the same line, file, checker ID, and *primary* message but each with a different set of *secondary* messages. In that case, each set of secondary messages will be associated with a different `Diagnostics.id`.)
- Sometimes multiple tools warn about the same condition (e.g., CWE-190) on the same line of the same file. They have the same `MetaAlerts.id`, but different `Diagnostics.id`.
- Sometimes, an alert from one tool maps to multiple `TaxonomyEntries.id`(s). (For example, a single alert from one tool could map to both INT31-C and CWE-190.) In that case, one `Diagnostics.id` is associated with multiple `MetaAlerts.id`(s). (They are different integers but are still associated in the **DiagnosticMetaAlertLinks** table.)

If a second tool provides an alert about the same flaw (both alerts are for the same `TaxonomyEntries.id`) on the same line of the same file, then a new entry must be made in the **DiagnosticMetaAlertLinks** table. The existing `MetaAlerts.id` and the tool's alert `Diagnostics.id` are entered for `DiagnosticMetaAlertLinks.diagnostic`. The `Diagnostics.id` is different for different tools.

Therefore, two diagnostics from two different tools produce two entries in the **DiagnosticMetaAlertLinks** table:

- `DiagnosticMetaAlertLinks.diagnostic = Diagnostics.id` for tool #1, and `DiagnosticMetaAlertLinks.meta_alert_id = MetaAlerts.id` for the shared meta-alert
- `DiagnosticMetaAlertLinks.diagnostic = Diagnostics.id` for tool #2, and `DiagnosticMetaAlertLinks.meta_alert_id = MetaAlerts.id` for the shared meta-alert

3.6 Alert Viewer

Figure 3 shows a screenshot of the alert viewer.

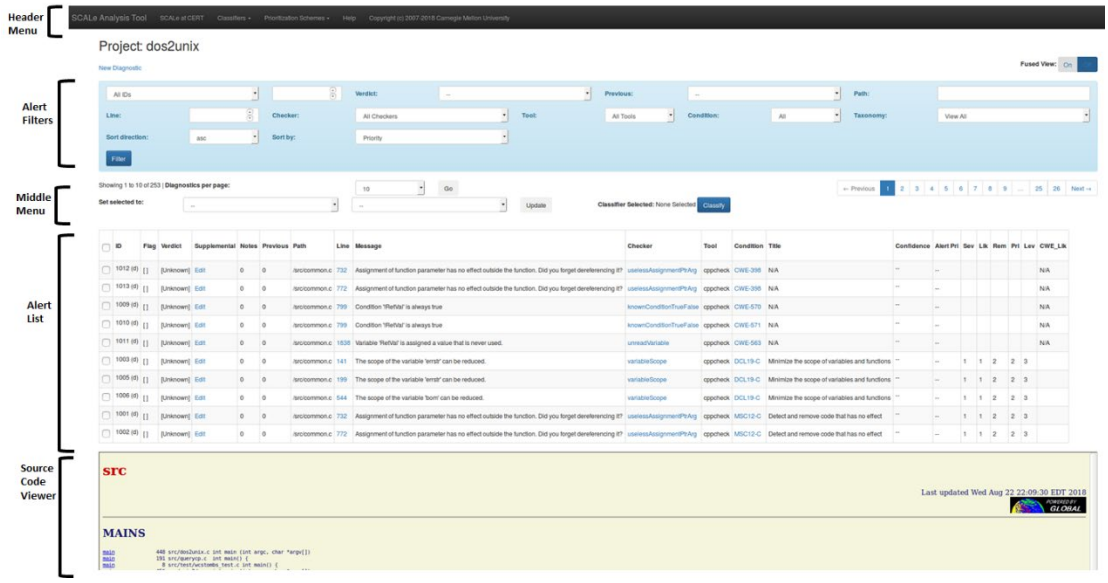


Figure 3: Alert Viewer

The following is a summary of new and modified Alert Viewer fields (listed in order from the top of Figure 3 to the bottom).

- The black **Header Menu** allows the user to
  - create, edit, and select classifier schemes
  - create, edit, and select prioritization schemes
  - upload new fields
- The light-blue **Alert Filters** section allows the user to filter the displayed alerts by various criteria. New filters now allow filtering by taxonomy or file path. They also allow prioritization using the `alert-priority`.
- The **Middle Menu** allows users to modify the alert list size, displays the `Classifier` scheme name if one has been selected, and provides a button for the user to manually start automated classification of alerts. (This functionality is not completely implemented, and selecting the button does not currently classify the alerts. It currently inserts example confidence values, not real ones.)
- The **Alert List** section displays meta-alerts in two ways:
  - single alerts that show an Alert ID on a white-shaded row
  - fused alerts that show a meta-alert ID on a yellow-shaded row. Audit determinations can be made on these alerts. Fused alerts can be expanded by selecting editable fields on the meta-alert row. The alerts that were fused are shown in separate rows (with blue-shading) below the fused alert.

  Alerts can be listed on multiple pages. To navigate these pages, the user can click the `Next/Previous` links near the top of the viewer or click on the desired page number.

3.7 Selecting a Prioritization Scheme

Prioritization scheme selection allows you to prioritize static analysis alerts using factors you care about. The formulas can combine classifier confidence and other values (e.g., risk, cost) used by the system, with math symbols including: `*`, `/`, `+`, `-`, `(`, and `)`.

To create a new prioritization scheme, select `Create New Scheme` from the header menu's `Prioritization Schemes` option.

Figure 4: Prioritization Selection

Next, select each taxonomy tab, select weights (see below), and enter a formula (or a single weighted field) to calculate the priority for meta-alerts for conditions in that taxonomy. Note that classifier-derived confidence and user-uploaded fields can be used in formulas for all taxonomies.

Weights (1 or higher) must be selected on the left for each field that will be used in the formula. Numbers and mathematical operators can also be entered using the keyboard.

Figure 5 shows a warning that appears if the button `Generate the Formula` is selected before a formula was entered for each taxonomy.

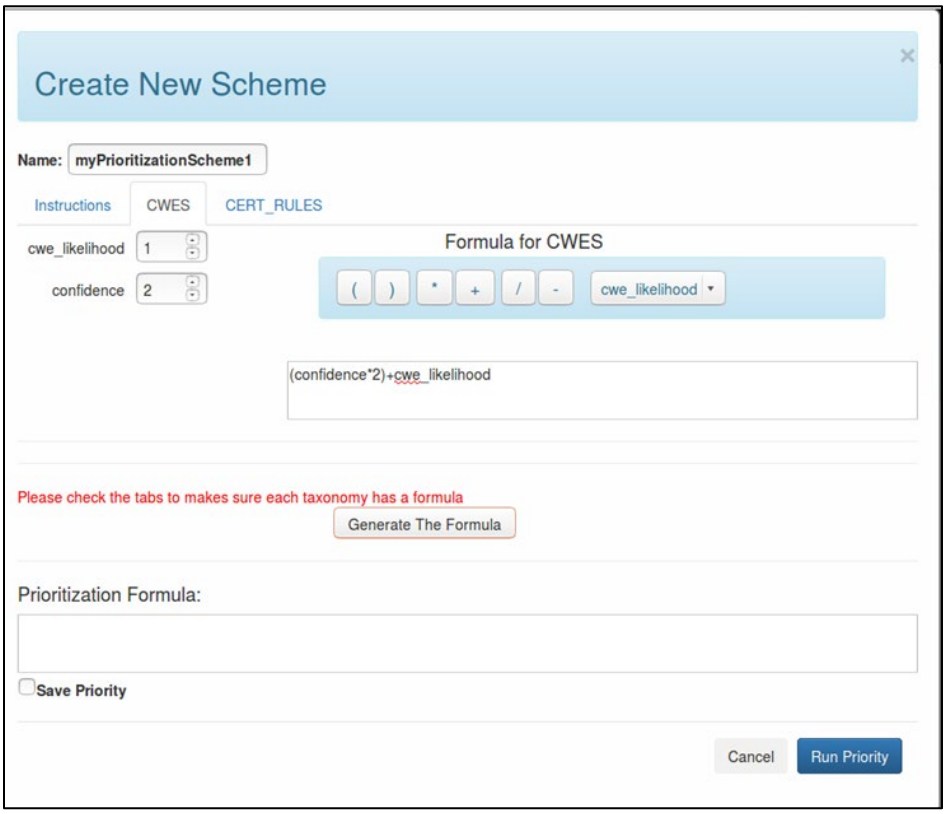


Figure 5: Create a New Prioritization Scheme

Next, enter the name of the prioritization scheme. For example, in Figure 5 and Figure 6, the name is `myPrioritizationScheme1`. To save the scheme, select `Save Priority`, as shown in Figure 6.

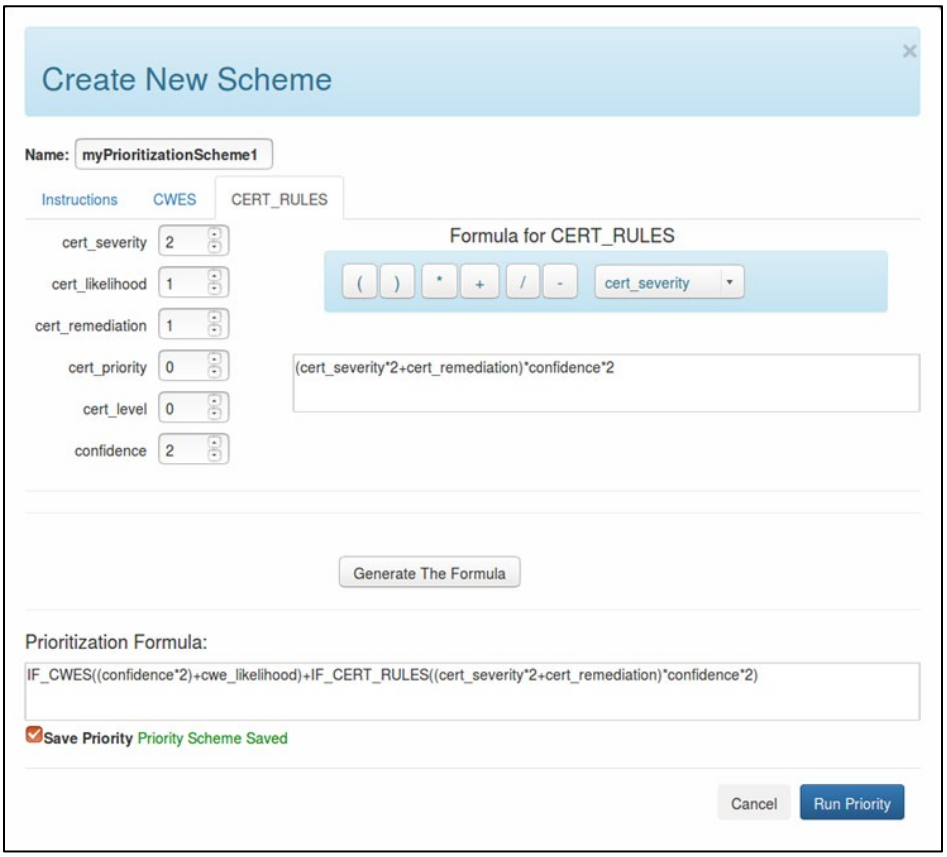


Figure 6: Save Prioritization Scheme

To run the prioritization, select the button `Run Priority` at the bottom-right of the pop-up window. This calculates values and puts them in the `Alert pri` (Alert Priority) field for the meta-alerts, as shown in Figure 7.

Confidence	Alert Pri	Sev	Lik	Rem
13.69	8	1	1	2
35.42	8	1	1	2
95.93	8	1	1	2
6.3	8	1	1	2
76.49	8	1	1	2
38.38	8	1	1	2
35.84	8	1	1	2
38.08	8	1	1	2
59.08	16	2	1	2
23.43	16	2	1	2

Figure 7: Alert Priority Field

To order meta-alerts by values from this priority scheme, follow these steps:

1. Under the filter section for `Sort by`, select `Alert Priority`.
2. For `Sort direction`, select `desc` (descending). The screenshot in Figure 8 shows these options selected.
3. Select the `Filter` button for the desired prioritization ordering.

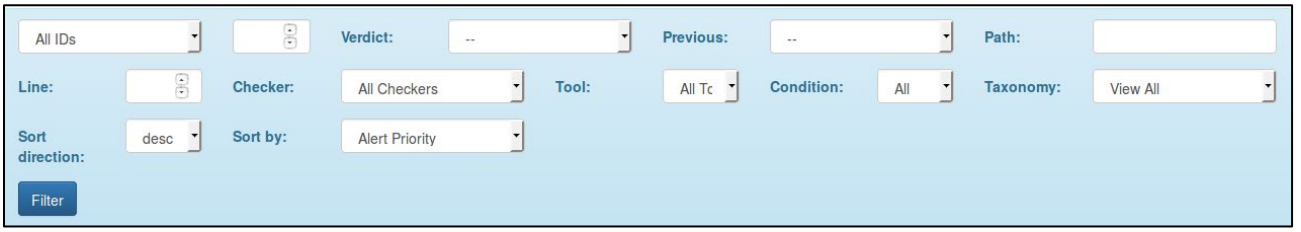


Figure 8: Alert Filters

After saving and running a new prioritization scheme, be aware that its full formula will not be displayed in the pop-up window for the previously saved scheme upon return to the `Prioritization` menu. This window does show the sub-formulas in each tab, however.

If the user selects the `Generate The Formula` button, the window displays the full formula.

3.8 Uploading Additional Fields

Users can upload additional fields by selecting the header menu item `Upload New Fields`. The popup window shown in Figure 9 appears.

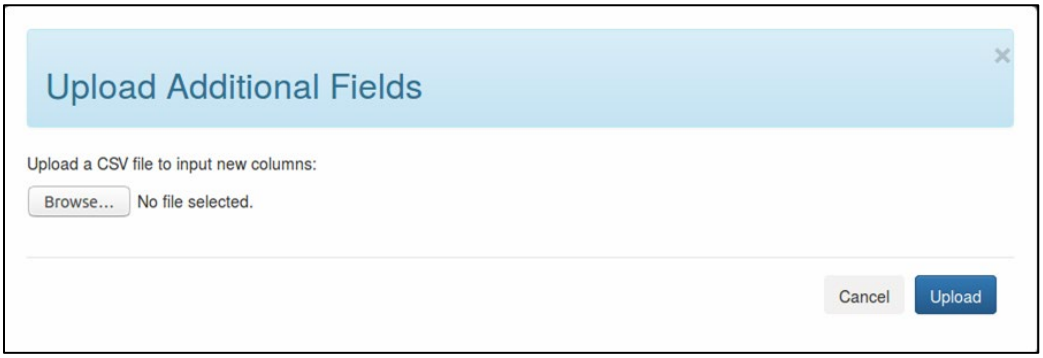


Figure 9: Upload Custom Fields

Currently, uploaded fields can be used in prioritization formulas but cannot be viewed in the GUI. Future versions of SCALe will include the fields in the GUI.

The upload-fields option is intended for advanced users that can work with SQL databases and that have or can generate values for new fields (e.g., based on advanced logic using the other alert fields or based on proprietary data). However, an extended and more user-friendly version of the concept could be added without technical difficulty beyond standard development. This approach would enable fewer technical users to generate values for new fields using mathematical formulas, advanced logic, and data beyond that provided in the initial SCALe database.

Uploaded files must be in the following comma-separated value (CSV) format:

- There is one line for every meta-alert ID that exists in the project.
- The left-most field contains the meta-alert ID.
- The top row contains labels for each field, with left-most top field being equivalent to `meta_alert_id`.
- The rest of the rows contain a value for each field under each meta-alert.

An example file (`user_upload_example.csv`) is located in the `demo` folder. It includes the following fields: `safeguard_countermeasure`, `vulnerability`, `residual_risk`, `impact`, `threat`, `risk`, `complexity`, `severity`, and `coupling`. The new user-uploaded fields capability allows you to upload those or any other new fields, along with values for those fields for each meta-alert.

3.9 Selecting a Classification Scheme

Classification is not available in the current version of SCALe. However, many features required for classification have already been integrated into it. Interfaces are provided for user testing and feedback.

To create a new classification scheme, select `Create New Classifier` from the `Classifiers` menu.

Next, select one of the options shown in Figure 10. It displays three options: `Xgboost`, `Random Forest`, and `Logistic Regression`.

Figure 10: Select a Classifier Scheme

After selecting a classifier type, the popup window in Figure 11 appears with the following options:

- `Projects Available`: This section lists available projects for labeled data (meta-alerts with determinations) that could be used to train the classifier. These project names are auto-populated from the current active set of SCALe projects. To use a project's labeled data to train the classifier, select the project name(s), then select the **Add** button.
- `Adaptive Heuristics:` This section provides options (including using none) to choose from various adaptive heuristics. An adaptive heuristic modifies classification values as new data relevant to the project comes in (e.g., as new audit determinations are made or as new static analysis tools are run on the code base and the tool output is loaded into the project). Many of the adaptive heuristics have user-editable parameters.
- `Automated Hyper-Parameter Optimization (AHPO):` This section provides AHPO options. You can select one or none. The options shown in the screenshot in Figure 11 are `caret` and `sei-ahpo`.

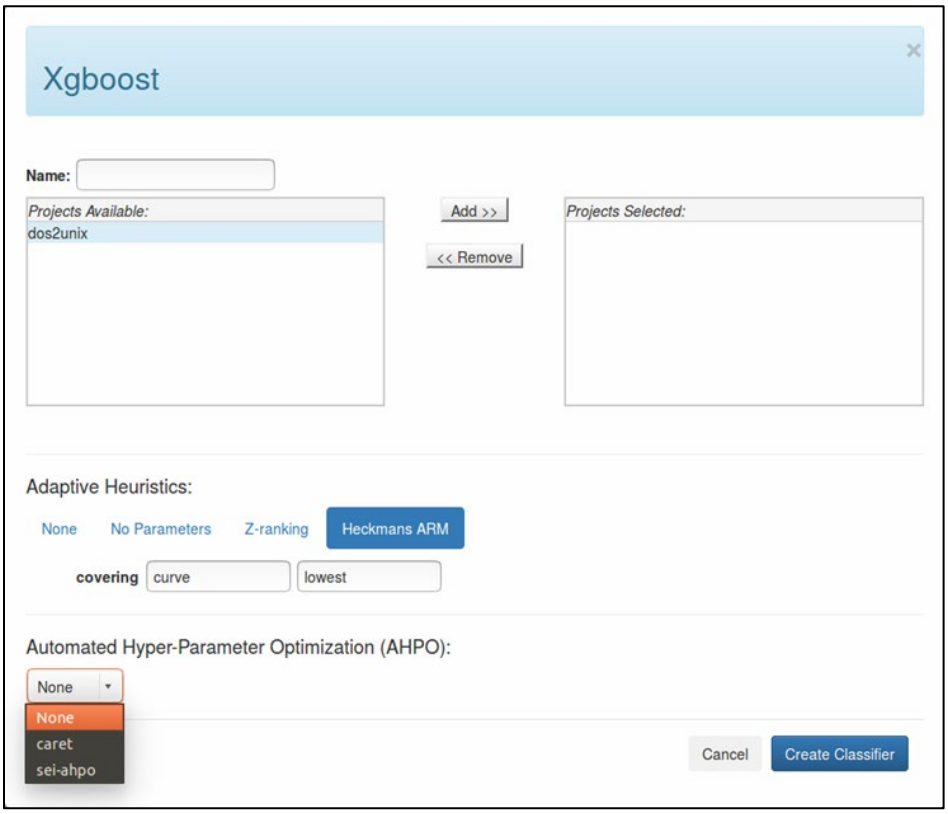


Figure 11: Edit a Classifier Scheme

Next, you can enter a name for the new classifier (e.g., `myNewClassifier`), then select the button `Create Classifier` at the bottom right.

Notice that the name of the selected classifier is now displayed next to the `Classify` button, as shown in the screenshot in Figure 12. Use this link to edit the values of the classifier as an alternative to using the **`Classifiers`** dropdown in the header menu.

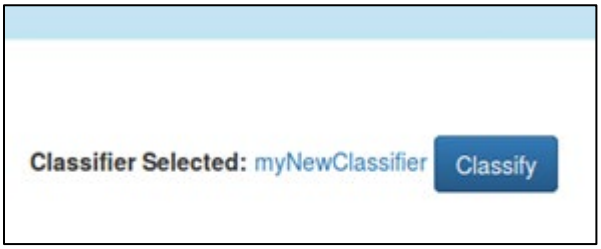


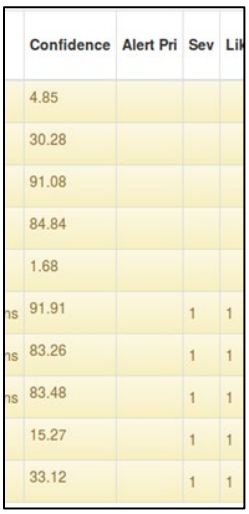
Figure 12: Selected Classifier Scheme

3.10 Running the Classifier

Classification is not available in this version of SCALe. However, many features required for classification have already been integrated into it. Interfaces are provided for user testing and feedback.

After selecting a classification scheme, click the `Classify` button. When fully functional, this will cause meta-alerts to be classified (e.g., confidence True or False will be predicted). Currently,

when the button is selected, example metrics are loaded for the `Confidence` field of meta-alerts. However, these metrics are for demonstration and usability purposes only. These values do not currently come from a classifier.



	Confidence	Alert Pri	Sev	Lik
	4.85			
	30.28			
	91.08			
	84.84			
	1.68			
ns	91.91		1	1
ns	83.26		1	1
ns	83.48		1	1
	15.27		1	1
	33.12		1	1

Figure 13*: Confidence Values*

3.11 Alert Determination History

The `Previous` field in the GUI indicates how many previous determinations were made. Selecting the hyperlinked number brings up a comprehensive history of audit determinations for that particular meta-alert, including primary determinations, supplemental determinations, the `Note` field, and the `Flag` field. Auditors can use this view to browse the old determinations for each meta-alert. In the exported database (see Figure 2), the determinations history is kept in the **Determinations** table, and historical determinations are distinguished using the `time` field. In the future, we plan to test whether using determination history as a feature increases the accuracy of the classifier.

3.12 Cascade Determinations

This feature allows you to take determinations made from a previous SCALe audit and automatically apply them to alerts generated by a new SCALe audit. The section `Upload Determinations from Project` in the `Edit Project` page (shown in Figure 14) provides this functionality. It uses the UNIX `diff` utility to determine if a code line within a file from a previous version of a code base matches a code line within the current version of the code base. If the lines match (i.e., `diff` mapped them to each other and the content of the line is unchanged) and there was a previously made determination for the meta-alert, then it infers the same determination for the current meta-alert. A new `Note` with a timestamp is added that identifies the determination as being cascaded (see Figure 15).

Caution: Cascaded determinations are not as trustworthy as direct (un-cascaded) determinations. Data, control, and type flow changes in source code may cause the previously correct determination to change. For example, with different control flow, a previously correct True determination can become a newly correct False determination.

Future work will analyze the accuracy of classifiers created using cascaded determinations compared to classifiers using only regular determinations.

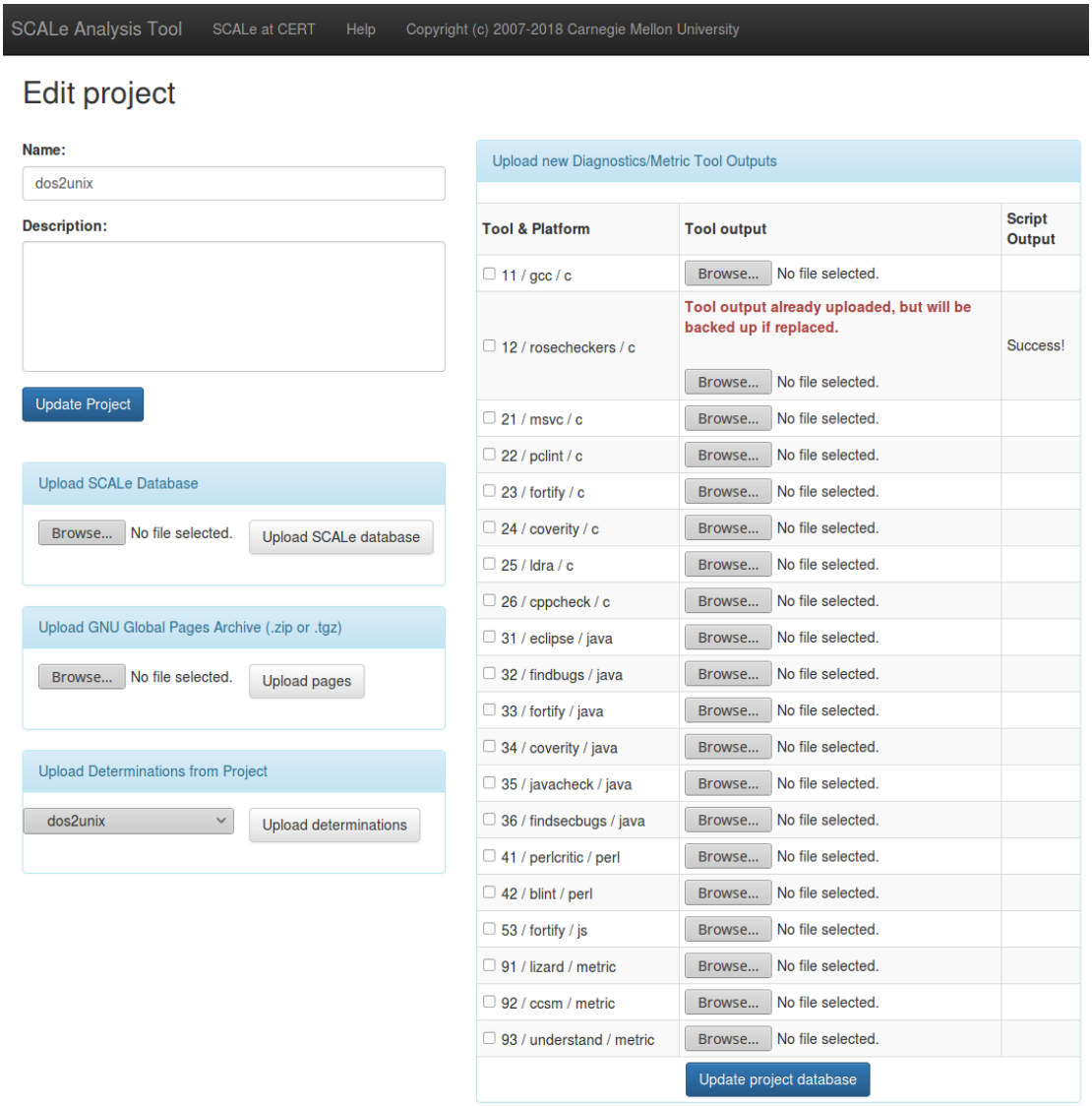


Figure 14: Upload Determinations

After importing cascaded determinations, the auditor view shows `Notes` entries for cascaded determinations in the partial screenshot in Figure 15.

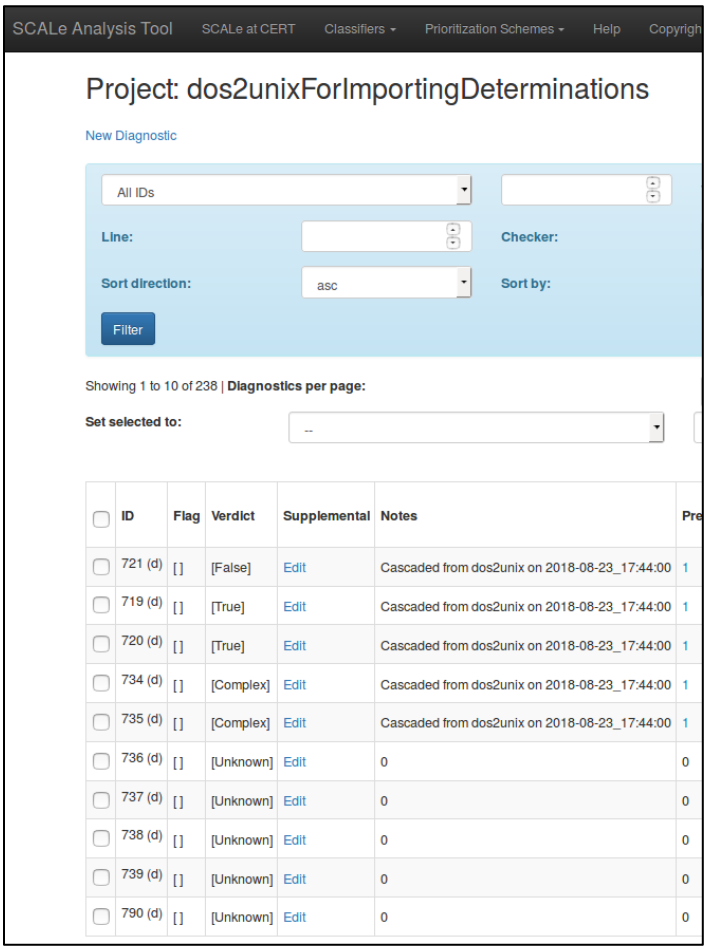


Figure 15: Cascaded Determinations

3.13 Docker

A Dockerfile has been added to the root directory of the SCALe project. With Docker installed, SCALe can be automatically installed and run in Docker containers [Docker 2018]. This functionality is currently being used in development and has not yet been tested in production. It may be used for continuous integration, continuous deployment, and automating the distribution of SCALe in the future.

4 Alert Prioritization and User-Uploaded Fields

As discussed in Section 3.8, users can now upload additional fields that can be used in prioritization schemes, specifying a field name and value entry for each meta-alert. The user-uploaded fields appear on the prioritization scheme interface as shown in Figure 16 in the `Additional Fields` section, and weights selected for these fields remain set for all taxonomies.

The following is an example of a prioritization formula that uses the user-uploaded field `safeguard_countermeasure` from the example file provided with SCALe (`scale.app/demo/user_upload_example.csv`):

```
IF_CWES(cwe_likelihood*safeguard_countermeasure) +

IF_CERT_RULES((cert_severity*2+cert_remediation)/3*safeguard_countermeasure)
```

Figure 16: User-Uploaded `Additional Fields` in Prioritization Scheme Interface

5 Testing by External Users

Development of SCALe has been, and continues to be, tested by external users during development on this project, resulting in bug report filings and some bug fixes.

- One of our collaborators (who has been using SCALe for 18 months) used SCALe version 2.1.3.0 to audit two contract projects. One project consisted of four Java (Android) code-bases, while the other consisted of two C++ codebases.
- Another organization contracted SEI services to audit code. This audit was performed using SCALe version 2.2.6.0.
- A version of SCALe almost wholly developed by this project starting in February 2018, with bug fixes and performance fixes, has been released to the public on GitHub [SCALe 2018a]. Potentially, this version (2.1.4.2 as of this writing) could be widely tested.

6 Architecture Development

This project developed an architecture to enable simplified use of automated static analysis alert classification and alert prioritization for various static analysis alert auditing frameworks. The architecture involves four servers with API calls for all communications among the servers. The four servers are

- **DataHub Module** stores information about codebases, test suite metadata, alerts, alert determinations, tools, and taxonomies. It stores and retrieves data for the User Interface (UI) Module. It also forwards project data to the Statistics Module when forwarding is required by an adaptive heuristic.
- **Statistics Module** handles classification, adaptive heuristics, reclassification, and automated hyper-parameter optimization (AHPO) prior to running classifiers. It stores classification schemes (including classifier algorithm, adaptive heuristic choice, and AHPO choice) and sends calculated data to the UI Module.
- **UI Module** supports static analysis alert auditing tools that have an auditing GUI, take static analysis tool output and codebases as input, and provide a data storage back end. Tools such as SCALe, CERDEC Software Assurance Tool (SwAT), and DHS SWAMP could function as a UI Module as long as they instantiate the API-required functions needed to interact with the rest of the architecture.
- **Prioritization Module** stores, retrieves, and modifies prioritization formulas, enabling sophisticated formulas that can combine classifier confidence and other values. It communicates only with UI Modules.

Figure 17 shows a high-level view of the architecture.

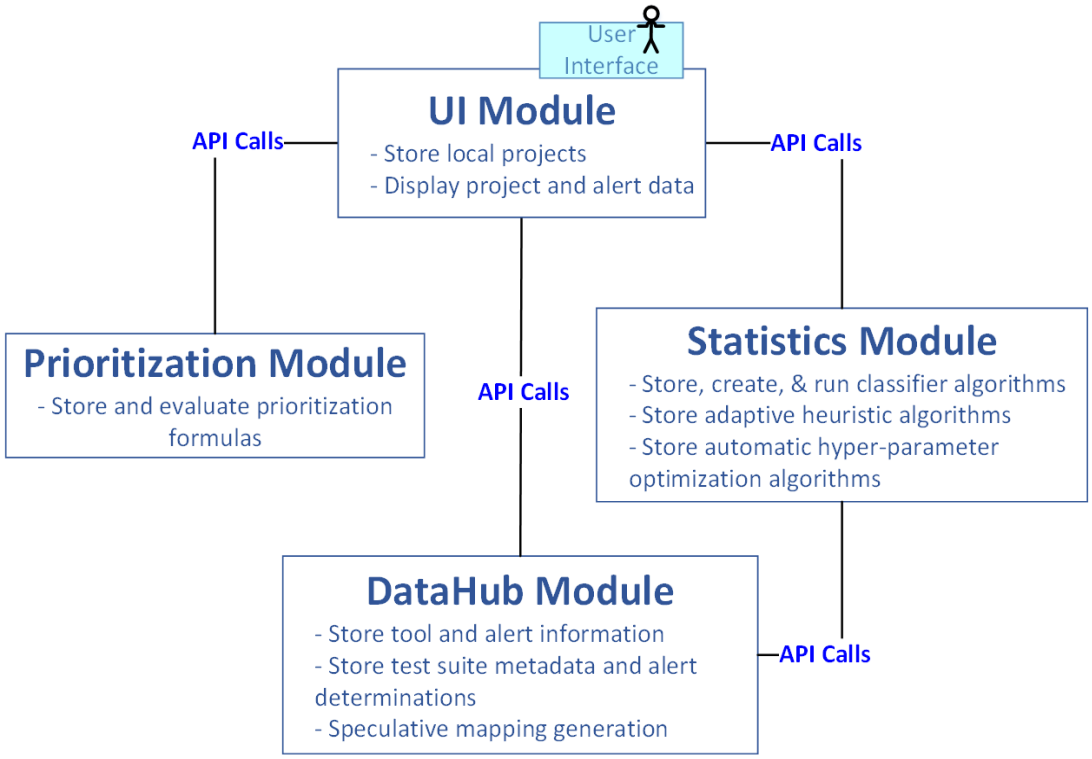


Figure 17: Architecture Overview

7 Application Programmer Interface (API) Development

This project developed an API for the four-server architecture. We developed a Representation State Transfer (RESTful) API[3] with the widely used swagger[4] tools [Swagger 2018]. (We used open-source `swagger-editor v2`, testing with `swagger-ui`, and automated code generation with `swagger-codegen`). The current API definition is provided in Appendix A: Rapid Models API Definition. Design considerations included a goal for the system to eventually be able to integrate static analysis alert auditing frameworks, including SCALe, DHS SWAMP, and Army CERDEC SwAT [SCALe 2018b, SWAMP 2018, CERDEC 2018]. Additional design considerations included goals of low-latency classification and prioritization, low network bandwidth, low memory use, and low disk space, along with security.

We used `swagger-codegen` on the API definition to automatically generate function stubs for the API functions. (Stubs have function names, parameters, and return values set but do not have internal function code.)

7.1 Current Status and Next Steps

Project developers are currently working to incorporate techniques used for adaptive (e.g., real-time) re-calculation of classifiers and alert re-ordering as alert determinations are made (e.g., marking an alert True or False), which include weighting more recent data (audit determinations) more heavily [Ruthruff 2008, Heckman 2011]. Developers are also in the process of implementing and testing internal API function code for a prototype system that instantiates the architecture. When that code is complete, we will integrate SCALe with the architecture system for classification and prioritization.

[3] https://searchmicroservices.techtarget.com/definition/RESTful-API

[4] https://swagger.io/tools/

Appendix A: Rapid Models API Definition

API to facilitate auditing static analysis alerts using classifiers, optional adaptive heuristics, and alert prioritization. The API enables jump-starting labeled datasets using test suites. It is intended to enable a wide range of users (with widely varying datasets, static analysis tools, ranging from having no labeled data to having a lot of labeled data, and regardless of if they have their own statistical experts) to benefit from using classifiers and sophisticated prioritization to automatically triage handling their static analysis alerts.
More information: https://www.sei.cmu.edu/research-capabilities/all-work/display.cfm?customel_datapageid_4050=6453
Contact Info: lflynn@cert.org
Version: 0.0.1

http://apache.org/licenses/LICENSE-2.0.html

Methods

[Jump to Models]

Table of Contents

- DELETE /projects/{project_id}
- POST /projects/multiple/{num_of_projects}
- GET /projects/{project_id}
- GET /taxonomies
- GET /taxonomies/{taxonomy_id}/{version}
- GET /test_suites
- GET /tools/{tool_id}/{version}
- GET /tools
- GET /projects
- POST /alerts/{project_id}/
- GET /alerts/{project_id}/
- PUT /projects/multiple/{num_of_projects}
- POST /projects/{project_id}
- POST /test_suites
- POST /tools

UIToPrioritization

- POST /priorities
- DELETE /priorities/{priority_id}
- GET /priorities
- GET /priorities/{priority_id}
- PUT /priorities/{priority_id}

UIToStats

- POST /classifiers/adaptive_heuristics/close
- POST /classifiers
- GET /classifiers/{classifier_instance_id}/analysis
- GET /classifiers
- POST /classifiers/{project_id}/{classifier_instance_id}

DataHubToStats

Up

POST /classifiers/{project_id}

Forward new Alerts that have been uploaded to the DataHub and have a current open AH request for its respective project. Returns status message for the DataHub to track if the request was completed. (**sendAlertUpdatesForClassifier**)

Path parameters

project_id (required)

Path Parameter — The id of the project associated with these alerts

Consumes

This API call consumes the following media types via the Content-Type request header:

- application/json

Request body
alert_data alert_data **(required)**
Body Parameter — Updated alert data and test_suite_id, if applicable
Return type
inline_response_200_1
Example data
Content-Type: application/json

```
{
  "project_id" : "project_id",
  "message" : "message"
}
```

Responses
200
OK inline_response_200_1
400
Unable to Upload Alerts
404
Invalid Project
default
Unexpected Error error

Up
PUT /classifiers
Send FFSA or code metrics tool info to the Stats Module. When a new tool is uploaded, the DataHub can send new tool info for projects with open AH requests automatically to keep the Stats Module in sync. (**sendNewToolInfo**)
Consumes
This API call consumes the following media types via the Content-Type request header:

- application/json

Request body
tool_info_w_projects tool_info_w_projects **(required)**
Body Parameter — Tool info, including name, version, plus FFSA checker info OR code metrics field info and project ids associated with this new tool.
Request headers
Return type
inline_response_200
Example data
Content-Type: application/json

```
{
  "message_id" : 0
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses
200
OK inline_response_200
400
Unable to Load Tool Information
default
Unexpected Error error

StatsToDataHub

Up
`POST /projects/adaptive_heuristics/close`
Send a list of project_ids to implement the adaptive heuristic alert forwarding close request on the DataHub. The request stops the forwarding of project alerts to the Stats Module. (**closeHeuristicProjects**)
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
project_ids string (required)
Body Parameter — project_ids to close adaptive heuristic alert forwarding request

Return type
stop_data_forwarding_response
Example data
Content-Type: application/json

```
""
```

Responses
200
OK stop_data_forwarding_response
400
Unable to Complete Request
default
Unexpected Error error

Up
`POST /projects/multiple`
Get multiple projects at once. Send tool and taxonomy information the stats module already has, along with projects it is requesting. This way, a response from the DataHub can efficiently leave out taxonomies and tools that the Stats Module already has info for. (See DataHub_to_Stats. If DataHub gets new alerts or code metrics for the project_id that the Stats Module didn't mention in this request, the DataHub will send that tool info to the Stats Module.) (**getMultipleProjects**)
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
projects_requested projects_requested **(required)**
Body Parameter — Tool and taxonomy information the stats module already has, along with projects it is requesting.
Return type
multiple_projects
Example data
Content-Type: application/json

```
""
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK multiple_projects
206
Some Errors Occurred multiple_projects
404
Invalid Request
default
Unexpected Error error

Up

`GET /projects/{project_id}`

Get an existing project from the DataHub Module. Send the project_id to the DataHub to retrieve the project information stored. (**getProjectById**)
Path parameters
project_id (required)
Path Parameter — The id of the project to retrieve
Return type
project
Example data
Content-Type: application/json

```
{
  "alerts" : [ {
    "alert_id" : "alert_id",
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
```

```
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "checker_id" : "checker_id",
    "more_messages" : [ "", "" ]
  }, {
    "alert_id" : "alert_id",
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
```

```
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "checker_id" : "checker_id",
    "more_messages" : [ "", "" ]
  } ],
  "test_suite_id" : "test_suite_id",
  "meta_alerts" : [ {
    "verdict" : {
      "key" : [ "verdict", "verdict" ]
    },
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
```

```
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "meta_alert_id" : "meta_alert_id",
    "condition_id" : "condition_id"
  }, {
    "verdict" : {
      "key" : [ "verdict", "verdict" ]
    },
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
```

```
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "meta_alert_id" : "meta_alert_id",
    "condition_id" : "condition_id"
  } ],
  "updated_at" : "2000-01-23T04:56:07.000+00:00",
  "project_id" : "project_id",
  "taxonomies" : [ {
    "taxonomy_id" : "taxonomy_id",
    "conditions" : [ {
      "cwe" : {
```

```
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    }, {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    } ],
    "taxonomy_name" : "taxonomy_name",
    "version" : "version"
  }, {
    "taxonomy_id" : "taxonomy_id",
    "conditions" : [ {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    }, {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
```

```
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    } ],
    "taxonomy_name" : "taxonomy_name",
    "version" : "version"
  } ],
  "test_suite_or_not" : true,
  "organization_id" : "organization_id",
  "description" : "description",
  "created_at" : "2000-01-23T04:56:07.000+00:00",
  "project_name" : "project_name",
  "tools" : [ {
    "tool_name" : "tool_name",
    "checker_data" : [ {
      "checker_name" : "checker_name",
      "condition_ids" : [ "condition_ids", "condition_ids" ],
      "checker_id" : "checker_id"
    }, {
      "checker_name" : "checker_name",
      "condition_ids" : [ "condition_ids", "condition_ids" ],
      "checker_id" : "checker_id"
    } ],
    "code_metrics_data" : "{}",
    "tool_id" : "tool_id",
    "version" : "version"
  }, {
    "tool_name" : "tool_name",
    "checker_data" : [ {
      "checker_name" : "checker_name",
      "condition_ids" : [ "condition_ids", "condition_ids" ],
      "checker_id" : "checker_id"
    }, {
      "checker_name" : "checker_name",
      "condition_ids" : [ "condition_ids", "condition_ids" ],
      "checker_id" : "checker_id"
    } ],
    "code_metrics_data" : "{}",
    "tool_id" : "tool_id",
    "version" : "version"
  } ]
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK project
404
Project Unavailable
default
Unexpected Error error

Up

`GET /taxonomies`

Get a list of Taxonomies available in the Modules. (**getTaxonomies**)

Return type

taxonomy_response

Example data

Content-Type: application/json

```
""
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

Returns a list of taxonomy ids, names and versions taxonomy_response

default

Unexpected Error error

Up

`GET /taxonomies/{taxonomy_id}/{version}`

Get data of a specific taxonomy based on the id and version. (**getTaxonomy**)

Path parameters

taxonomy_id (required)

Path Parameter — Taxonomy ID to retrieve data

version (required)

Path Parameter — Taxonomy version to retrieve data

Return type

taxonomy

Example data

Content-Type: application/json

```
{
  "taxonomy_id" : "taxonomy_id",
  "conditions" : [ {
    "cwe" : {
      "cwe_likelihood" : 9
    },
    "cert" : {
      "severity" : 5,
      "remediation" : 1,
      "likelihood" : 5,
      "level" : 7,
      "priority" : 2
    },
    "condition_name" : "condition_name",
    "title" : "title",
    "condition_id" : "condition_id",
    "platform" : "platform"
  }, {
    "cwe" : {
      "cwe_likelihood" : 9
```

```
    },
    "cert" : {
      "severity" : 5,
      "remediation" : 1,
      "likelihood" : 5,
      "level" : 7,
      "priority" : 2
    },
    "condition_name" : "condition_name",
    "title" : "title",
    "condition_id" : "condition_id",
    "platform" : "platform"
  } ],
  "taxonomy_name" : "taxonomy_name",
  "version" : "version"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Returns data for a specific taxonomy taxonomy
404
Taxonomy Unavailable
default
Unexpected Error error

Up
`GET /tools/{tool id}/{version}`

Get specific tool data based on id and version information. (**getTooldata**)
Path parameters
tool_id (required)
Path Parameter — The id of the tool to retrieve data about the tool
version (required)
Path Parameter — Tool version
Return type
tool_info
Example data
Content-Type: application/json

```
{
  "tool_name" : "tool_name",
  "checker_data" : [ {
    "checker_name" : "checker_name",
    "condition_ids" : [ "condition_ids", "condition_ids" ],
    "checker_id" : "checker_id"
  }, {
    "checker_name" : "checker_name",
    "condition_ids" : [ "condition_ids", "condition_ids" ],
    "checker_id" : "checker_id"
```

```
  } ],
  "code_metrics_data" : "{}",
  "tool_id" : "tool_id",
  "version" : "version"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Returns data for a particular tool tool_info
404
Tool Information Unavailable
default
Unexpected Error error

Up
`GET /tools`

Get a list of tool ids, versions and names available in the Module. (**getTools**)
Return type
array[tool_response]
Example data
Content-Type: application/json

```
[ {
  "tool_name" : "tool_name",
  "tool_id" : "tool_id",
  "version" : "version"
}, {
  "tool_name" : "tool_name",
  "tool_id" : "tool_id",
  "version" : "version"
} ]
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Returns a list of tool id, version and name
default
Unexpected Error error

Up
`POST /projects/adaptive_heuristics/multiple`

Request multiple projects from the DataHub and send an open request for AH updates using the adapt_status parameter. The adapt_status will determine the AH updates field each

project. A list of tools and taxonomies present in the Stats Module will be sent to eliminate the DataHub sending duplicate information. (**openAHForMultipleProjects**)

Request body

projects_requested_w_ah projects_requested_w_ah **(required)**

Body Parameter — Multiple project_ids with adaptive heuristic request

Return type

array[project_alert_forwarding_response]

Example data

Content-Type: application/json


```
[ {
  "ah_forwarding" : true,
  "project_data" : {
    "alerts" : [ {
      "alert_id" : "alert_id",
      "primary_message" : {
        "file_path" : "file_path",
        "line_start" : 0,
        "line_end" : 6
      },
      "determinations" : {
        "flag_list" : [ {
          "flag" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "flag" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "inapplicable_environment_list" : [ {
          "inapplicable_environment" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "inapplicable_environment" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "ignored_list" : [ {
          "ignored" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "ignored" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "verdict_list" : [ {
          "verdict" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "verdict" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "determination_id" : "determination_id",
        "dead_list" : [ {
          "dead" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "dead" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "dangerous_construct_list" : [ {
          "dangerous_construct" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
```

```
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "checker_id" : "checker_id",
    "more_messages" : [ "", "" ]
  }, {
    "alert_id" : "alert_id",
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
```

```
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "checker_id" : "checker_id",
    "more_messages" : [ "", "" ]
  } ],
  "test_suite_id" : "test_suite_id",
  "meta_alerts" : [ {
    "verdict" : {
      "key" : [ "verdict", "verdict" ]
    },
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
```

```
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "meta_alert_id" : "meta_alert_id",
    "condition_id" : "condition_id"
  }, {
    "verdict" : {
      "key" : [ "verdict", "verdict" ]
    },
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
```

```
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "meta_alert_id" : "meta_alert_id",
    "condition_id" : "condition_id"
  } ],
  "updated_at" : "2000-01-23T04:56:07.000+00:00",
  "project_id" : "project_id",
  "taxonomies" : [ {
    "taxonomy_id" : "taxonomy_id",
    "conditions" : [ {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    }, {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    } ],
    "taxonomy_name" : "taxonomy_name",
```

```
    "version" : "version"
  }, {
    "taxonomy_id" : "taxonomy_id",
    "conditions" : [ {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    }, {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    } ],
    "taxonomy_name" : "taxonomy_name",
    "version" : "version"
  } ],
  "test_suite_or_not" : true,
  "organization_id" : "organization_id",
  "description" : "description",
  "created_at" : "2000-01-23T04:56:07.000+00:00",
  "project_name" : "project_name",
  "tools" : [ {
    "tool_name" : "tool_name",
    "checker_data" : [ {
      "checker_name" : "checker_name",
      "condition_ids" : [ "condition_ids", "condition_ids" ],
      "checker_id" : "checker_id"
    }, {
      "checker_name" : "checker_name",
      "condition_ids" : [ "condition_ids", "condition_ids" ],
      "checker_id" : "checker_id"
    } ],
    "code_metrics_data" : "{}",
    "tool_id" : "tool_id",
    "version" : "version"
  }, {
    "tool_name" : "tool_name",
    "checker_data" : [ {
      "checker_name" : "checker_name",
```

```
        "condition_ids" : [ "condition_ids", "condition_ids" ],
        "checker_id" : "checker_id"
      }, {
        "checker_name" : "checker_name",
        "condition_ids" : [ "condition_ids", "condition_ids" ],
        "checker_id" : "checker_id"
      } ],
      "code_metrics_data" : "{}",
      "tool_id" : "tool_id",
      "version" : "version"
    } ]
  },
  "message" : "message"
}, {
  "ah_forwarding" : true,
  "project_data" : {
    "alerts" : [ {
      "alert_id" : "alert_id",
      "primary_message" : {
        "file_path" : "file_path",
        "line_start" : 0,
        "line_end" : 6
      },
      "determinations" : {
        "flag_list" : [ {
          "flag" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "flag" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "inapplicable_environment_list" : [ {
          "inapplicable_environment" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "inapplicable_environment" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "ignored_list" : [ {
          "ignored" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "ignored" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "verdict_list" : [ {
          "verdict" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "verdict" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "determination_id" : "determination_id",
        "dead_list" : [ {
          "dead" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "dead" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
```

```
        "dangerous_construct_list" : [ {
          "dangerous_construct" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "dangerous_construct" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "notes_list" : [ {
          "notes" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "notes" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ]
      },
      "checker_id" : "checker_id",
      "more_messages" : [ "", "" ]
    }, {
      "alert_id" : "alert_id",
      "primary_message" : {
        "file_path" : "file_path",
        "line_start" : 0,
        "line_end" : 6
      },
      "determinations" : {
        "flag_list" : [ {
          "flag" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "flag" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "inapplicable_environment_list" : [ {
          "inapplicable_environment" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "inapplicable_environment" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "ignored_list" : [ {
          "ignored" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "ignored" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "verdict_list" : [ {
          "verdict" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "verdict" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "determination_id" : "determination_id",
        "dead_list" : [ {
          "dead" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "dead" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
```

```
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "checker_id" : "checker_id",
    "more_messages" : [ "", "" ]
  } ],
  "test_suite_id" : "test_suite_id",
  "meta_alerts" : [ {
    "verdict" : {
      "key" : [ "verdict", "verdict" ]
    },
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
```

```
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "meta_alert_id" : "meta_alert_id",
    "condition_id" : "condition_id"
  }, {
    "verdict" : {
      "key" : [ "verdict", "verdict" ]
    },
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
```

```
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "meta_alert_id" : "meta_alert_id",
    "condition_id" : "condition_id"
  } ],
  "updated_at" : "2000-01-23T04:56:07.000+00:00",
  "project_id" : "project_id",
  "taxonomies" : [ {
    "taxonomy_id" : "taxonomy_id",
    "conditions" : [ {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    }, {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
```

```
      "platform" : "platform"
    } ],
    "taxonomy_name" : "taxonomy_name",
    "version" : "version"
  }, {
    "taxonomy_id" : "taxonomy_id",
    "conditions" : [ {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    }, {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    } ],
    "taxonomy_name" : "taxonomy_name",
    "version" : "version"
  } ],
  "test_suite_or_not" : true,
  "organization_id" : "organization_id",
  "description" : "description",
  "created_at" : "2000-01-23T04:56:07.000+00:00",
  "project_name" : "project_name",
  "tools" : [ {
    "tool_name" : "tool_name",
    "checker_data" : [ {
      "checker_name" : "checker_name",
      "condition_ids" : [ "condition_ids", "condition_ids" ],
      "checker_id" : "checker_id"
    }, {
      "checker_name" : "checker_name",
      "condition_ids" : [ "condition_ids", "condition_ids" ],
      "checker_id" : "checker_id"
    } ],
    "code_metrics_data" : "{}",
    "tool_id" : "tool_id",
    "version" : "version"
  }, {
```

```
      "tool_name" : "tool_name",
      "checker_data" : [ {
        "checker_name" : "checker_name",
        "condition_ids" : [ "condition_ids", "condition_ids" ],
        "checker_id" : "checker_id"
      }, {
        "checker_name" : "checker_name",
        "condition_ids" : [ "condition_ids", "condition_ids" ],
        "checker_id" : "checker_id"
      } ],
      "code_metrics_data" : "{}",
      "tool_id" : "tool_id",
      "version" : "version"
    } ]
  },
  "message" : "message"
} ]
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK
default
Unexpected Error [error](#)

Up

`POST /projects/adaptive_heuristics/{project_id}`

Request project data from the DataHub and send an open request for AH updates. Set the AH updates field to true for this project. A list of tools and taxonomies present in the Stats Module will be sent to eliminate the DataHub sending duplicate information. (**openAH-ForProject**)
Path parameters
project_id (required)
Path Parameter — The id of the project to return
Request body
tools_taxonomies_present [tools_taxonomies_present](#) **(optional)**
Body Parameter — Tools and Taxonomy Data present in the Stats Module
Return type
[project_alert_forwarding_response](#)
Example data
Content-Type: application/json

```
{
  "ah_forwarding" : true,
  "project_data" : {
    "alerts" : [ {
      "alert_id" : "alert_id",
      "primary_message" : {
```

```
        "file_path" : "file_path",
        "line_start" : 0,
        "line_end" : 6
      },
      "determinations" : {
        "flag_list" : [ {
          "flag" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "flag" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "inapplicable_environment_list" : [ {
          "inapplicable_environment" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "inapplicable_environment" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "ignored_list" : [ {
          "ignored" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "ignored" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "verdict_list" : [ {
          "verdict" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "verdict" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "determination_id" : "determination_id",
        "dead_list" : [ {
          "dead" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "dead" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "dangerous_construct_list" : [ {
          "dangerous_construct" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "dangerous_construct" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "notes_list" : [ {
          "notes" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "notes" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ]
      },
      "checker_id" : "checker_id",
      "more_messages" : [ "", "" ]
    }, {
      "alert_id" : "alert_id",
```

```
      "primary_message" : {
        "file_path" : "file_path",
        "line_start" : 0,
        "line_end" : 6
      },
      "determinations" : {
        "flag_list" : [ {
          "flag" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "flag" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "inapplicable_environment_list" : [ {
          "inapplicable_environment" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "inapplicable_environment" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "ignored_list" : [ {
          "ignored" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "ignored" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "verdict_list" : [ {
          "verdict" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "verdict" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "determination_id" : "determination_id",
        "dead_list" : [ {
          "dead" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "dead" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "dangerous_construct_list" : [ {
          "dangerous_construct" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "dangerous_construct" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "notes_list" : [ {
          "notes" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "notes" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ]
      },
      "checker_id" : "checker_id",
      "more_messages" : [ "", "" ]
    } ],
```

```
  "test_suite_id" : "test_suite_id",
  "meta_alerts" : [ {
    "verdict" : {
      "key" : [ "verdict", "verdict" ]
    },
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
```

```
      } ]
    },
    "meta_alert_id" : "meta_alert_id",
    "condition_id" : "condition_id"
  }, {
    "verdict" : {
      "key" : [ "verdict", "verdict" ]
    },
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
```

```
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "meta_alert_id" : "meta_alert_id",
    "condition_id" : "condition_id"
  } ],
  "updated_at" : "2000-01-23T04:56:07.000+00:00",
  "project_id" : "project_id",
  "taxonomies" : [ {
    "taxonomy_id" : "taxonomy_id",
    "conditions" : [ {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    }, {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    } ],
    "taxonomy_name" : "taxonomy_name",
    "version" : "version"
  }, {
    "taxonomy_id" : "taxonomy_id",
    "conditions" : [ {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
```

```
        "condition_id" : "condition_id",
        "platform" : "platform"
      }, {
        "cwe" : {
          "cwe_likelihood" : 9
        },
        "cert" : {
          "severity" : 5,
          "remediation" : 1,
          "likelihood" : 5,
          "level" : 7,
          "priority" : 2
        },
        "condition_name" : "condition_name",
        "title" : "title",
        "condition_id" : "condition_id",
        "platform" : "platform"
      } ],
      "taxonomy_name" : "taxonomy_name",
      "version" : "version"
    } ],
    "test_suite_or_not" : true,
    "organization_id" : "organization_id",
    "description" : "description",
    "created_at" : "2000-01-23T04:56:07.000+00:00",
    "project_name" : "project_name",
    "tools" : [ {
      "tool_name" : "tool_name",
      "checker_data" : [ {
        "checker_name" : "checker_name",
        "condition_ids" : [ "condition_ids", "condition_ids" ],
        "checker_id" : "checker_id"
      }, {
        "checker_name" : "checker_name",
        "condition_ids" : [ "condition_ids", "condition_ids" ],
        "checker_id" : "checker_id"
      } ],
      "code_metrics_data" : "{}",
      "tool_id" : "tool_id",
      "version" : "version"
    }, {
      "tool_name" : "tool_name",
      "checker_data" : [ {
        "checker_name" : "checker_name",
        "condition_ids" : [ "condition_ids", "condition_ids" ],
        "checker_id" : "checker_id"
      }, {
        "checker_name" : "checker_name",
        "condition_ids" : [ "condition_ids", "condition_ids" ],
        "checker_id" : "checker_id"
      } ],
      "code_metrics_data" : "{}",
      "tool_id" : "tool_id",
      "version" : "version"
    } ]
  },
  "message" : "message"
}
```


Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK project_alert_forwarding_response
default
Unexpected Error error

StatsToUI

Up
`POST /displays/{project_id}/{classifier_instance_id}`
Send confidence data for a specific project. When there is an open Adaptive Heuristic for projects associated with the UI Module, the Stats Module will automatically re-send the confidence data for the updated classifier based on some threshold set by the Adaptive Heuristic. (**sendUpdatedClassifierConfidence**)
Path parameters
project_id (required)
Path Parameter — project id of the project used as the target domain
classifier_instance_id (required)
Path Parameter — The id of the classifier instance to run on the target domain
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
classifier_results classifier_results **(required)**
Body Parameter — Updated confidence information
Responses
200
OK
404
Invalid Project
default
Unexpected Error error

UIToDataHub

Up
`POST /projects`

Create a new instance of a project. This request will return the project_id and the organization_id that should be used by the UI Module for referencing this project within the DataHub. (**createProject**)

Consumes

This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body

project_metadata project_metadata (required)

Body Parameter — Name, test_suite (_id and _or_not parameters), organization name, description, and source file for the project to create

Return type

project_metadata_response

Example data

Content-Type: application/json

```
{
  "test_suite_id" : "test_suite_id",
  "project_id" : "project_id",
  "organization_id" : "organization_id",
  "description" : "description",
  "project_name" : "project_name"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

OK project_metadata_response

400

Unable to Create Project

404

Invalid Organization

default

Unexpected Error error

Up

`DELETE /projects/{project_id}`

Delete a specific project from the DataHub by supplying the project_id for the project to delete. (**deleteProjectById**)

Path parameters

project_id (required)

Path Parameter — The id of the project to delete

Request headers

Responses

200

OK

default
Unexpected Error error

Up
`POST /projects/multiple/{num_of_projects}`
Create new instances of multiple projects. This request will return the project_ids and the organization_id that should be used by the UI Module for referencing these projects within the DataHub. (**getMultipleProjectIds**)
Path parameters
num_of_projects (required)
Path Parameter — Number of projects to create
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
project_metadata_list project_metadata (required)
Body Parameter — List of name, description, organization name and source file for projects
Return type
array[project_metadata_response]
Example data
Content-Type: application/json

```
[ {
  "test_suite_id" : "test_suite_id",
  "project_id" : "project_id",
  "organization_id" : "organization_id",
  "description" : "description",
  "project_name" : "project_name"
}, {
  "test_suite_id" : "test_suite_id",
  "project_id" : "project_id",
  "organization_id" : "organization_id",
  "description" : "description",
  "project_name" : "project_name"
} ]
```

Responses
200
OK
400
Unable to Create Projects
default
Unexpected Error error

Up
`GET /projects/{project_id}`
Get an existing project from the DataHub Module. Send the project_id to the DataHub to retrieve the project information stored. (**getProjectById**)
Path parameters
project_id (required)
Path Parameter — The id of the project to retrieve
Return type
project
Example data

Content-Type: application/json


```
{
  "alerts" : [ {
    "alert_id" : "alert_id",
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
```

```
    },
    "checker_id" : "checker_id",
    "more_messages" : [ "", "" ]
  }, {
    "alert_id" : "alert_id",
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
```

```
      } ]
    },
    "checker_id" : "checker_id",
    "more_messages" : [ "", "" ]
  } ],
  "test_suite_id" : "test_suite_id",
  "meta_alerts" : [ {
    "verdict" : {
      "key" : [ "verdict", "verdict" ]
    },
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
```

```
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "meta_alert_id" : "meta_alert_id",
    "condition_id" : "condition_id"
  }, {
    "verdict" : {
      "key" : [ "verdict", "verdict" ]
    },
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
```

```
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
    },
    "meta_alert_id" : "meta_alert_id",
    "condition_id" : "condition_id"
  } ],
  "updated_at" : "2000-01-23T04:56:07.000+00:00",
  "project_id" : "project_id",
  "taxonomies" : [ {
    "taxonomy_id" : "taxonomy_id",
    "conditions" : [ {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    }, {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    } ],
    "taxonomy_name" : "taxonomy_name",
    "version" : "version"
  }, {
    "taxonomy_id" : "taxonomy_id",
    "conditions" : [ {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
```

```
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    }, {
      "cwe" : {
        "cwe_likelihood" : 9
      },
      "cert" : {
        "severity" : 5,
        "remediation" : 1,
        "likelihood" : 5,
        "level" : 7,
        "priority" : 2
      },
      "condition_name" : "condition_name",
      "title" : "title",
      "condition_id" : "condition_id",
      "platform" : "platform"
    } ],
    "taxonomy_name" : "taxonomy_name",
    "version" : "version"
  } ],
  "test_suite_or_not" : true,
  "organization_id" : "organization_id",
  "description" : "description",
  "created_at" : "2000-01-23T04:56:07.000+00:00",
  "project_name" : "project_name",
  "tools" : [ {
    "tool_name" : "tool_name",
    "checker_data" : [ {
      "checker_name" : "checker_name",
      "condition_ids" : [ "condition_ids", "condition_ids" ],
      "checker_id" : "checker_id"
    }, {
      "checker_name" : "checker_name",
      "condition_ids" : [ "condition_ids", "condition_ids" ],
      "checker_id" : "checker_id"
    } ],
    "code_metrics_data" : "{}",
    "tool_id" : "tool_id",
    "version" : "version"
  }, {
    "tool_name" : "tool_name",
    "checker_data" : [ {
      "checker_name" : "checker_name",
      "condition_ids" : [ "condition_ids", "condition_ids" ],
      "checker_id" : "checker_id"
    }, {
      "checker_name" : "checker_name",
      "condition_ids" : [ "condition_ids", "condition_ids" ],
      "checker_id" : "checker_id"
    } ],
    "code_metrics_data" : "{}",
    "tool_id" : "tool_id",
    "version" : "version"
  } ]
```

```
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK project
404
Project Unavailable
default
Unexpected Error error

Up
`GET /taxonomies`

Get a list of Taxonomies available in the Modules. (**getTaxonomies**)
Return type
taxonomy_response
Example data
Content-Type: application/json

```
""
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Returns a list of taxonomy ids, names and versions taxonomy_response
default
Unexpected Error error

Up
`GET /taxonomies/{taxonomy_id}/{version}`

Get data of a specific taxonomy based on the id and version. (**getTaxonomy**)
Path parameters
taxonomy_id (required)
Path Parameter — Taxonomy ID to retrieve data
version (required)
Path Parameter — Taxonomy version to retrieve data
Return type
taxonomy
Example data
Content-Type: application/json

```
{
  "taxonomy_id" : "taxonomy_id",
```

```
  "conditions" : [ {
    "cwe" : {
      "cwe_likelihood" : 9
    },
    "cert" : {
      "severity" : 5,
      "remediation" : 1,
      "likelihood" : 5,
      "level" : 7,
      "priority" : 2
    },
    "condition_name" : "condition_name",
    "title" : "title",
    "condition_id" : "condition_id",
    "platform" : "platform"
  }, {
    "cwe" : {
      "cwe_likelihood" : 9
    },
    "cert" : {
      "severity" : 5,
      "remediation" : 1,
      "likelihood" : 5,
      "level" : 7,
      "priority" : 2
    },
    "condition_name" : "condition_name",
    "title" : "title",
    "condition_id" : "condition_id",
    "platform" : "platform"
  } ],
  "taxonomy_name" : "taxonomy_name",
  "version" : "version"
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Returns data for a specific taxonomy taxonomy
404
Taxonomy Unavailable
default
Unexpected Error error

Up
`GET /test_suites`

Get a list of Test Suite ids, names and versions that are available (**getTestSuites**)
Return type
array[test_suite_response]
Example data
Content-Type: application/json


```
[ {
```

```
  "test_suite_id" : "test_suite_id",
  "version" : "version",
  "test_suite_name" : "test_suite_name"
}, {
  "test_suite_id" : "test_suite_id",
  "version" : "version",
  "test_suite_name" : "test_suite_name"
} ]
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Returns a list of test suites

400
Test Suites Unavailable
default
Unexpected Error error

Up
`GET /tools/{tool_id}/{version}`
Get specific tool data based on id and version information. (**getTooldata**)
Path parameters
tool_id (required)
Path Parameter — The id of the tool to retrieve data about the tool
version (required)
Path Parameter — Tool version
Return type
tool_info
Example data
Content-Type: application/json

```
{
  "tool_name" : "tool_name",
  "checker_data" : [ {
    "checker_name" : "checker_name",
    "condition_ids" : [ "condition_ids", "condition_ids" ],
    "checker_id" : "checker_id"
  }, {
    "checker_name" : "checker_name",
    "condition_ids" : [ "condition_ids", "condition_ids" ],
    "checker_id" : "checker_id"
  } ],
  "code_metrics_data" : "{}",
  "tool_id" : "tool_id",
  "version" : "version"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Returns data for a particular tool tool_info
404
Tool Information Unavailable
default
Unexpected Error error

Up
`GET /tools`

Get a list of tool ids, versions and names available in the Module. (**getTools**)
Return type
array[tool_response]
Example data
Content-Type: application/json

```
[ {
  "tool_name" : "tool_name",
  "tool_id" : "tool_id",
  "version" : "version"
}, {
  "tool_name" : "tool_name",
  "tool_id" : "tool_id",
  "version" : "version"
} ]
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
Returns a list of tool id, version and name
default
Unexpected Error error

Up
`GET /projects`

Retrieve a list of all projects available (**listProjects**)
Return type
array[project_metadata_response]
Example data
Content-Type: application/json

```
[ {
  "test suite id" : "test suite id",
  "project_id" : "project_id",
  "organization id" : "organization id",
  "description" : "description",
  "project_name" : "project_name"
}, {
  "test suite id" : "test suite id",
  "project_id" : "project_id",
```

```
  "organization_id" : "organization_id",
  "description" : "description",
  "project_name" : "project_name"
} ]
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK
404
No Projects Available
default
Unexpected Error error

Up
`POST /alerts/{project_id}/`
Send alerts for a specific project and upload the alerts to the DataHub Module. (**sendAlerts**)
Path parameters
project_id (required)
Path Parameter — The id of the project to update with the new alerts
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
multiple_alerts multiple_alerts **(required)**
Body Parameter — Alert data to send and upload
Responses
200
OK
404
Invalid Project
default
Unexpected Error error

Up
`GET /alerts/{project_id}/`
Get alerts for a specific project. Request only the alerts for a project in the DataHub. (**showAlertsById**)
Path parameters
project_id (required)
Path Parameter — The id of the project to retrieve alerts from
Return type
get_alerts_response
Example data
Content-Type: application/json

```
{
```

```
"project_id" : "project_id",
"multiple_alerts" : {
  "alerts" : [ {
    "alert_id" : "alert_id",
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ]
```

```
    },
    "checker_id" : "checker_id",
    "more_messages" : [ "", "" ]
  }, {
    "alert_id" : "alert_id",
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "notes" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
```

```
      } ]
    },
    "checker_id" : "checker_id",
    "more_messages" : [ "", "" ]
  } ],
  "meta_alerts" : [ {
    "verdict" : {
      "key" : [ "verdict", "verdict" ]
    },
    "primary_message" : {
      "file_path" : "file_path",
      "line_start" : 0,
      "line_end" : 6
    },
    "determinations" : {
      "flag_list" : [ {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "flag" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "inapplicable_environment_list" : [ {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "inapplicable_environment" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "ignored_list" : [ {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "ignored" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "verdict_list" : [ {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "verdict" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "determination_id" : "determination_id",
      "dead_list" : [ {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dead" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "dangerous_construct_list" : [ {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      }, {
        "dangerous_construct" : true,
        "timestamp" : "2000-01-23T04:56:07.000+00:00"
      } ],
      "notes_list" : [ {
        "notes" : true,
```

```
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "notes" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ]
      },
      "meta_alert_id" : "meta_alert_id",
      "condition_id" : "condition_id"
    }, {
      "verdict" : {
        "key" : [ "verdict", "verdict" ]
      },
      "primary_message" : {
        "file_path" : "file_path",
        "line_start" : 0,
        "line_end" : 6
      },
      "determinations" : {
        "flag_list" : [ {
          "flag" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "flag" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "inapplicable_environment_list" : [ {
          "inapplicable_environment" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "inapplicable_environment" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "ignored_list" : [ {
          "ignored" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "ignored" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "verdict_list" : [ {
          "verdict" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "verdict" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "determination_id" : "determination_id",
        "dead_list" : [ {
          "dead" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "dead" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ],
        "dangerous_construct_list" : [ {
          "dangerous_construct" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "dangerous_construct" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
```

```
        } ],
        "notes_list" : [ {
          "notes" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        }, {
          "notes" : true,
          "timestamp" : "2000-01-23T04:56:07.000+00:00"
        } ]
      },
      "meta_alert_id" : "meta_alert_id",
      "condition_id" : "condition_id"
    } ]
  }
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK get_alerts_response
default
Unexpected Error error

Up
`PUT /projects/multiple/{num of projects}`
Upload multiple projects. After creating mutiple project instances, use the project_ids and organization_id from the DataHub to upload the project information. (**uploadMultipleProjects**)
Path parameters
num_of_projects (required)
Path Parameter — Number of projects to upload
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
multiple_projects multiple_projects (required)
Body Parameter — Project data to upload
Responses
200
OK
206
Partial Upload Completed partial_project_upload
400
Unable to Upload Projects
default
Unexpected Error error

Up
`POST /projects/{project_id}`

Upload a project. After creating a project instance, use the project_id and organization_id from the DataHub to upload the project information. (**uploadProject**)
Path parameters
project_id (required)
Path Parameter — The id of the project to upload
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
project_data project (required)
Body Parameter — Project data to upload
Responses
200
OK
400
Unable to Upload Project
default
Unexpected Error error

Up
`POST /test_suites`
Upload a Test Suite to the DataHub Module. (**uploadTestSuite**)
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
test_suite_data test_suite (required)
Body Parameter — Test Suite information to upload
Return type
test_suite_response
Example data
Content-Type: application/json

```
{
  "test_suite_id" : "test_suite_id",
  "version" : "version",
  "test_suite_name" : "test_suite_name"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK test_suite_response
400
Unable to Upload Test Suite
default

Unexpected Error error

Up

POST /tools

Upload new tool data to the Module. Returns a tool_id for future referencing the tool in the DataHub Module. (**uploadTool**)

Consumes

This API call consumes the following media types via the Content-Type request header:

- application/json

Request body

tool_data tool_data **(required)**

Body Parameter — Tool information to upload

Return type

tool_response

Example data

Content-Type: application/json

```
{
  "tool_name" : "tool_name",
  "tool_id" : "tool_id",
  "version" : "version"
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- application/json

Responses

200

OK tool_response

400

Unable to Upload Tool Information

default

Unexpected Error error

UIToPrioritization

Up

POST /priorities

Create a new prioritization scheme to upload to the Prioritization Module. (**createPrioritization**)

Consumes

This API call consumes the following media types via the Content-Type request header:

- application/json

Request body

priority_scheme_data priority_scheme_data **(required)**

Body Parameter — Prioritization scheme to create
Return type
priority_response
Example data
Content-Type: application/json

```
{
  "priority_id" : "priority_id",
  "priority_name" : "priority_name"
}
```

Responses
200
OK priority_response
400
Unable to Create Prioritization
default
Unexpected Error error

Up
`DELETE /priorities/{priority_id}`
Delete a specific prioritization scheme (**deletePrioritizationById**)
Path parameters
priority_id (required)
Path Parameter — The id of the prioritization scheme
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
project_info project_info (required)
Body Parameter — Project ID and Organization ID associated with this prioritization scheme
Responses
200
OK
default
Unexpected Error error

Up
`GET /priorities`
List all prioritization schemes available in the Prioritization Module. (**listPrioritizations**)
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
project_info project_info (optional)
Body Parameter — Tailor the list to only prioritization schemes associated with a project ID and/or organization ID
Return type
prioritization_list
Example data
Content-Type: application/json

```
""
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK prioritization_list
404
Prioritization Not Available
default
Unexpected Error error

Up
`GET /priorities/{priority_id}`

Get prioritization scheme for a specific project (**showPrioritizationById**)
Path parameters
priority_id (required)
Path Parameter — The id of the prioritization scheme
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
project_info project_info (optional)
Body Parameter — Project ID and Organization ID associated with this prioritization scheme
Return type
priority_scheme_data
Example data
Content-Type: application/json

```
{
  "priority_name" : "priority_name",
  "project_id" : "project_id",
  "organization_id" : "organization_id",
  "formula" : "formula",
  "weighted_columns" : "{}"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK priority_scheme_data
400
Invalid Request

404
Prioritization Scheme Unavailable
default
Unexpected Error error

Up
`PUT /priorities/{priority_id}`
Update an existing prioritization scheme in the Prioritization Module (**updatePrioritizationById**)
Path parameters
priority_id (required)
Path Parameter — The id of the prioritization scheme
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
update_priority_data update_priority_data **(required)**
Body Parameter — Prioritization Scheme to update
Responses
200
OK
400
Unable to Update Prioritization Scheme
default
Unexpected Error error

UIToStats

Up
`POST /classifiers/adaptive_heuristics/close`
Stop adaptive heuristic forward request. Send a request to close (set to false) the adaptive heuristic for the projects listed in the classifier instance. (**closeAdaptiveHeuristicDataForwarding**)
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
close_data_forwarding close_data_forwarding **(required)**
Body Parameter — Information to send close adaptive heuristic request
Return type
inline_response_200_2
Example data
Content-Type: application/json

```
{
  "message" : "message",
  "classifier_instance_id" : "classifier_instance_id"
}
```

Responses
200
OK inline_response_200_2
400
Invalid Close Request
404
Classifier Instance Unavailable
default
Unexpected Error error

Up
`POST /classifiers`

Create a new classifier. Send Classifier information including Automated Hyper-Parameter Optimization (AHPO) and Adaptive Heuristics (AH) to the Stats Module along with project_ids for projects to use in creating/training a classifier. Returns an id that is used to then run the classifier and any additional information for the classifier. (**createClassifier**)
Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body
classifier_instance classifier_instance (required)
Body Parameter — Classifier information to create
Return type
create_classifier_response
Example data
Content-Type: application/json

```
{
  "analysis_messages" : "{}",
  "classifier_instance_id" : "classifier_instance_id"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK create_classifier_response
400
Unable to Create Classifier
default
Unexpected Error error

Up
`GET /classifiers/{classifier_instance_id}/analysis`

Get analysis for a specific Classifier including performance metrics. (**getClassifierAnalysis**)
Path parameters
classifier_instance_id (required)
Path Parameter — The id of the classifier to get analysis info
Consumes

This API call consumes the following media types via the Content-Type request header:

- `application/json`

Return type
analysis_results
Example data
Content-Type: application/json

```
{
  "project_id" : "project_id",
  "classifier_analysis" : "{}"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`
- `text/html`

Responses
200
OK analysis_results
404
Classifier Information Unavailable
default
Unexpected Error error

Up
`GET /classifiers`
List all classifiers and their associated data. Use the ids returned from this request to work with classifiers. (**listClassifiers**)
Return type
classifier_list
Example data
Content-Type: application/json

```
""
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK classifier_list
404
Classifiers Unavailable
default
Unexpected Error error

Up

`POST /classifiers/{project_id}/{classifier_instance_id}`

Run a specific Classifier. Run the classifier on the project identified by id in the request URL. The response contains confidence data and analysis of classifier performance. (**runClassifierById**)

Path parameters

project_id (required)

Path Parameter — The id of the target project to run the classifier on

classifier_instance_id (required)

Path Parameter — The id of the classifier instance to run on the target domain

Consumes

This API call consumes the following media types via the Content-Type request header:

- `application/json`

Return type

classifier_results

Example data

Content-Type: application/json

```
{
  "project_id" : "project_id",
  "classifier_analysis" : "{}",
  "confidence_data" : [ {
    "confidence" : 0.8008281904610115,
    "meta_alert_id" : "meta_alert_id"
  }, {
    "confidence" : 0.8008281904610115,
    "meta_alert_id" : "meta_alert_id"
  } ]
}
```

Produces

This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses

200

OK classifier_results

400

Unable to Run Classifier

404

Invalid Request

default

Unexpected Error error

Models

[Jump to Methods]

Table of Contents

2. alert_data -
3. analysis_results -
4. cert -
5. checker -
6. classifier_instance -
7. classifier_list -
8. classifier_list_inner -
9. classifier_list_inner_adaptive_heuristics -
10. classifier_list_inner_ahpos -
11. classifier_results -
12. classifier_results_confidence_data -
13. close_data_forwarding -
14. condition -
15. create_classifier_response -
16. cwe -
17. determination -
18. determination_dangerous_construct_list -
19. determination_dead_list -
20. determination_flag_list -
21. determination_ignored_list -
22. determination_inapplicable_environment_list -
23. determination_notes_list -
24. determination_verdict_list -
25. error -
26. get_alerts_response -
27. inline_response_200 -
28. inline_response_200_1 -
29. inline_response_200_2 -
30. message -
31. meta_alert -
32. multiple_alerts -
33. multiple_projects -
34. partial_project_upload -
35. partial_project_upload_inner -
36. prioritization_list -
37. prioritization_list_inner -
38. priority_response -
39. priority_scheme_data -
40. project -
41. project_alert_forwarding_response -
42. project_info -
43. project_metadata -
44. project_metadata_response -
45. projects_requested -
46. projects_requested_w_ah -
47. projects_requested_w_ah_project_status -
48. stop_data_forwarding_response -
49. stop_data_forwarding_response_inner -
50. taxonomy -
51. taxonomy_response -
52. taxonomy_response_inner -
53. test_suite -
54. test_suite_response -
55. tool_data -
56. tool_info -

57. tool_info_w_projects -
58. tool_response -
59. tools_taxonomies_present -
60. tools_taxonomies_present_taxonomies_present -
61. tools_taxonomies_present_tools_present -
62. update_priority_data -
63. secondary_message -

`alert` - Up

alert_id
String
checker_id
String
tool_name
String
primary_message
message
more_messages (optional)
array[secondary_message]
determinations (optional)
determination

`alert_data` - Up

test_suite_id (optional)
String test_suite_id associated with this project
multiple_alerts
multiple_alerts

`analysis_results` - Up

project_id (optional)
String
classifier_analysis (optional)
Object

`cert` - Up

remediation (optional)
Integer
likelihood (optional)
Integer
severity (optional)
Integer
priority (optional)
Integer
level (optional)
Integer

`checker` - Up

checker_id (optional)

String
checker_name (optional)
String
condition_ids (optional)
array[String]

`classifier_instance` - Up

classifier_id
String
classifier_name (optional)
String
organization_id
String
project_ids
array[String]
ahpo_id (optional)
String
ahpo_data (optional)
map[String, Object]
adaptive_heuristic_id (optional)
String
ah_parameters (optional)
map[String, Object]

`classifier_list` - Up

`classifier_list_inner` - Up

classifier_id (optional)
String
classifier_name (optional)
String
ahpos (optional)
array[classifier_list_inner_ahpos]
adaptive_heuristics (optional)
array[classifier_list_inner_adaptive_heuristics]

`classifier_list_inner_adaptive_heuristics` - Up

adaptive_heuristic_id (optional)
String
adaptive_heuristic_name (optional)
String
ah_parameters (optional)
Object

`classifier_list_inner_ahpos` - Up

ahpo_id (optional)
String
ahpo_name (optional)
String

`classifier_results` - Up

project_id
String ID of project in the target domain
confidence_data
array[classifier_results_confidence_data]
classifier_analysis (optional)
Object

`classifier_results_confidence_data` - Up

meta_alert_id (optional)
String
confidence (optional)
Double format: double

`close_data_forwarding` - Up

classifier_instance_id (optional)
String classifier instance id that points to the projects that no longer need data forwarding.
organization_id (optional)
String

`condition` - Up

condition_id
String
condition_name (optional)
String
title
String
platform (optional)
String
cert (optional)
cert
cwe (optional)
cwe

`create_classifier_response` - Up

classifier_instance_id (optional)
String
analysis_messages (optional)
Object Additional information that will help to understand this classifier instance's performance

`cwe` - Up

cwe_likelihood (optional)
Integer

`determination` - Up

determination_id
String
flag_list (optional)
array[determination_flag_list]
verdict_list (optional)
array[determination_verdict_list]
ignored_list (optional)
array[determination_ignored_list]
dead_list (optional)
array[determination_dead_list]
inapplicable_environment_list (optional)
array[determination_inapplicable_environment_list]
dangerous_construct_list (optional)
array[determination_dangerous_construct_list]
notes_list (optional)
array[determination_notes_list]

`determination_dangerous_construct_list` - Up

dangerous_construct (optional)
Boolean
timestamp (optional)
Date format: date-time

`determination_dead_list` - Up

dead (optional)
Boolean
timestamp (optional)
Date format: date-time

`determination_flag_list` - Up

flag (optional)
Boolean
timestamp (optional)
Date format: date-time

`determination_ignored_list` - Up

ignored (optional)
Boolean
timestamp (optional)
Date format: date-time

`determination_inapplicable_environment_list` - Up

inapplicable_environment (optional)
Boolean
timestamp (optional)
Date format: date-time

`determination_notes_list` - Up

notes (optional)
Boolean
timestamp (optional)
Date format: date-time

`determination_verdict_list` - Up

verdict (optional)
Boolean
timestamp (optional)
Date format: date-time

`error` - Up

code
Integer
message
String

`get_alerts_response` - Up

project_id (optional)
String
multiple_alerts (optional)
multiple_alerts

`inline_response_200` - Up

message_id (optional)
Integer

`inline_response_200_1` - Up

project_id (optional)
String
message (optional)
String

`inline_response_200_2` - Up

classifier_instance_id (optional)
String
message (optional)
String

`message` - Up

line_start
Integer
line_end (optional)
Integer
file_path
String

`meta_alert` - Up

meta_alert_id
String
primary_message
message
tool_name
String
condition_id
String
determinations (optional)
determination
verdict (optional)
map[String, array[String]]

`multiple_alerts` - Up

meta_alerts (optional)
array[meta_alert]
alerts (optional)
array[alert]

`multiple_projects` - Up

`partial_project_upload` - Up

`partial_project_upload_inner` - Up

project_id
String
upload_status
Boolean Boolean to determine if project was successfully uploaded
message (optional)
String

`prioritization_list` - Up

`prioritization_list_inner` - Up

priority_id (optional)
String
priority_name (optional)
String
organization_id (optional)
String

`priority_response` - Up

priority_id (optional)
String
priority_name (optional)
String

`priority_scheme_data` - Up

project_id
String
priority_name
String
organization_id
String
formula
String
weighted_columns (optional)
Object

`project` - Up

project_id
String
project_name
String
description (optional)
String
organization_id
String
test_suite_or_not
Boolean
test_suite_id (optional)
String
meta_alerts (optional)
array[meta_alert]
alerts (optional)
array[alert]
tools (optional)
array[tool_info]
taxonomies (optional)
array[taxonomy]
created_at (optional)
Date format: date-time
updated_at (optional)
Date format: date-time

`project_alert_forwarding_response` - Up

project_data (optional)
project
ah_forwarding (optional)
Boolean Identifies if the adaptive heuristic flag has been set at the DataHub for data forwarding.
message (optional)
String

`project_info` - Up

project_id (optional)
String

organization_id
String

`project_metadata` - **Up**

project_name
String
description (optional)
String Description of the project
organization_name
String Name of the organization
source_file (optional)
byte[] source file to upload format: binary
test_suite_or_not
Boolean Boolean to determine if the project is associated with a test suite
test_suite_id (optional)
String ID of the associated test suite

`project_metadata_response` - **Up**

project_id
String
project_name
String
description (optional)
String
organization_id
String
test_suite_id (optional)
String

`projects_requested` - **Up**

List of projects requested from the datahub as well as a list of tools and taxonomies to avoid sending redundant information inside of each project structure. If the stats module already has information about a particular tool or taxonomy the datahub does not need to sent that information in the project structure. Instead, all other associated projects data should be sent in the response.
project_ids
array[String]
tools_taxonomies_present (optional)
tools_taxonomies_present

`projects_requested_w_ah` - **Up**

List of projects requested from the datahub (along with whether it includes an adaptive heuristic, e.g. adapt_status) and lists of tools and taxonomies to avoid sending redundant information inside of each project structure. If the stats module already has information about a particular tool or taxonomu the datahub does not need to sent that information in the project structure. Instead, all other associated projects data should be sent in the response. Each project with an adapt_status set to true denotes a project requesting alert forwarding based on an adaptive heuristic.
project_status
array[projects_requested_w_ah_project_status]

tools_taxonomies_present (optional)
tools_taxonomies_present

`projects_requested_w_ah_project_status` - Up

project_id (optional)
String
adapt_status (optional)
Boolean

`stop_data_forwarding_response` - Up

`stop_data_forwarding_response_inner` - Up

project_id (optional)
String
ah_forwarding_stopped (optional)
Boolean If true the data forwarding CLOSE request for this project has succeeded, else an error occured
message (optional)
String

`taxonomy` - Up

taxonomy_id
String
taxonomy_name (optional)
String
version
String
conditions (optional)
array[condition]

`taxonomy_response` - Up

`taxonomy_response_inner` - Up

taxonomy_id (optional)
String
taxonomy_name (optional)
String
version (optional)
String

`test_suite` - Up

test_suite_name (optional)
String
version (optional)
String
metadata_file (optional)
byte[] format: binary
source_file (optional)

byte[] format: binary
source_url (optional)
String format: uri
use_license_file (optional)
byte[] format: binary
author (optional)
String

`test_suite_response` - **Up**

test_suite_id (optional)
String
test_suite_name (optional)
String
version (optional)
String

`tool_data` - **Up**

tool_name
String
version
String
checker_data (optional)
array[checker]
code_metrics_data (optional)
Object

`tool_info` - **Up**

tool_id
String
tool_name
String
version
String
checker_data (optional)
array[checker]
code_metrics_data (optional)
Object

`tool_info_w_projects` - **Up**

project_ids
array[String]
tool_info
tool_info

`tool_response` - **Up**

tool_id
String
version

String
tool_name
String

`tools_taxonomies_present` - Up

tools_present
array[tools_taxonomies_present_tools_present] List of tools already present at the source module (stats). The destination module (datahub) will use this list to avoid sending duplicate tool information.
taxonomies_present
array[tools_taxonomies_present_taxonomies_present] List of taxonomies already present at the source module (stats). The destination module (datahub) will use this list to avoid sending duplicate taxonomy information.

`tools_taxonomies_present_taxonomies_present` - Up

taxonomy_id (optional)
String
version (optional)
String

`tools_taxonomies_present_tools_present` - Up

tool_id (optional)
String
version (optional)
String

`update_priority_data` - Up

project_id (optional)
String
priority_name
String
organization_id
String
formula (optional)
String
weighted_columns (optional)
Object

`secondary_message` - Up

line_start
Integer
line_end (optional)
Integer
file_path
String
message_text
String

References/Bibliography

URLs are valid as of the publication date of this document.

[CCSM 2018]
Bright Silence. [n. d.]. *GitHub Website, ccsm Repository.* [Accessed May 2, 2019] https://github.com/bright-tools/ccsm

[CERDEC 2018]
United States Army Communications-Electronics Research, Development and Engineering Center. *CERDEC Website.* [Accessed May 2, 2019] https://www.cerdec.army.mil/

[Docker 2018]
Docker Website. [Accessed May 2, 2019] https://www.docker.com/

[Flynn 2018a]
Flynn, Lori; Kurtz, Zachary; & Snavely, William. Static Analysis Alert Test Suites as a Source of Training Data for Alert Classifiers [blog post]. *SEI Blog*. April 2018. [Accessed May 2, 2019] https://insights.sei.cmu.edu/sei_blog/2018/04/static-analysis-alert-test-suites-as-a-source-of-training-data-for-alert-classifiers.html

[Flynn 2018b]
Flynn, Lori, et al. Prioritizing alerts from multiple static analysis tools, using classification models. *Proceedings of the 1st International Workshop on Software Qualities and Their Dependencies*. 2018. [Accessed May 2, 2019] https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=524678

[Heckman 2011]
Heckman, Sarah & Williams, Laurie. A systematic literature review of actionable alert identification techniques for automated static code analysis. *Information and Software Technology*. Volume 53. Issue 4. Pages 363-387. April 2011. [Accessed May 2, 2019] https://www.sciencedirect.com/science/article/pii/S0950584910002235?via%3Dihub

[Kong 2007]
Kong, Deguang, et al. ISA: a source code static vulnerability detection system based on data fusion. *Proceedings of the 2nd international conference on Scalable information systems*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering). 2007. [Accessed May 2, 2019] http://staff.ustc.edu.cn/~qzheng/45.pdf

[Marjamäki 2018]
Marjamäki, Daniel. Cppcheck: A tool for static C/C++ code analysis. *SourceForge Website.* [Accessed May 2, 2019] http://cppcheck.sourceforge.net/

[MITRE 2018]
MITRE Corporation. Common Weakness Enumeration (CWE). *MITRE Website.* [Accessed May 2, 2019] https://cwe.mitre.org/

[Ruthruff 2008]
Ruthruff, Joseph R., et al. Predicting accurate and actionable static analysis warnings: an experimental approach. *Proceedings of the 30th ACM/IEEE International Conference on Software Engineering*. May 10-18, 2008. [Accessed May 2, 2019] https://ieeexplore.ieee.org/document/4814145/

[SCALe 2018a]
SCALe. GitHub Website, cmu-sei Repository. [Accessed May 2, 2019] https://github.com/cmu-sei/SCALe

[SCALe 2018b]
Software Engineering Institute. SCALe Collection. SEI Digital Library. [Accessed May 2, 2019] https://resources.sei.cmu.edu/library/asset-view.cfm?assetID=473847

[SEI 2018a]
Software Engineering Institute. SEI CERT Division Releases Downloadable Source Code Analysis Tool. *Press Release.* [Accessed May 2, 2019] https://www.sei.cmu.edu/news-events/news/article.cfm?assetId=524804

[SEI 2018b]
Software Engineering Institute. SEI CERT Coding Standards. *Secure Coding Standards Wiki.* [Accessed May 2, 2019] https://wiki.sei.cmu.edu/confluence/display/seccode/SEI+CERT+Coding+Standards

[Svoboda 2016]
Svoboda, David; Flynn, Lori; & Snavely, William. Static Analysis Alert Audits: Lexicon & Rules. In *Proceedings of the IEEE Cybersecurity Development Conference (IEEE SecDev).* November 3-4, 2016. [Accessed May 2, 2019] https://resources.sei.cmu.edu/library/asset-view.cfm?assetID=484185

[Swagger 2018]
API Tools for Individuals, Teams, and Enterprises. *Swagger Website.* [Accessed May 2, 2019] https://swagger.io/tools/

[SWAMP 2018]
Software Assurance Marketplace (SWAMP). *Continuous Assurance Website.* [Accessed May 2, 2019] https://continuousassurance.org/about-us/faqs/

[Yin 2018]
Yin, Terry. [n. d.]. *GitHub Website, lizard Repository.* [Accessed May 2, 2019] https://github.com/terryyin/lizard

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
(Leave Blank)	May 2019	Final

4. TITLE AND SUBTITLE	5. FUNDING NUMBERS
Integration of Automated Static Analysis Alert Classification and Prioritization with Auditing Tools: Special Focus on SCALe	FA8702-15-D-0002

6. AUTHOR(S)

Lori Flynn, Ebonie McNeil, David Svoboda, Derek Leung, Zachary Kurtz, & Jiyeon Lee

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213	CMU/SEI-2019-TR-007

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFLCMC/PZE/Hanscom Enterprise Acquisition Division 20 Schilling Circle Building 1305 Hanscom AFB, MA 01731-2116	n/a

11. SUPPLEMENTARY NOTES

12A DISTRIBUTION/AVAILABILITY STATEMENT	12B DISTRIBUTION CODE
Unclassified/Unlimited, DTIC, NTIS	

13. ABSTRACT (MAXIMUM 200 WORDS)

This report summarizes technical progress and plans as of late September 2018 for developing a system to perform automated classification and advanced prioritization of static analysis alerts. Many features and fields have been added to the Source Code Analysis Laboratory (SCALe) static analysis alert auditing tool to support this functionality. This report describes the new features and fields, and how to use them. It also describes the plan to connect this enhanced version of SCALe to an architecture that will provide classification and prioritization via API calls, and provides the API definition that has been developed. A prototype that instantiates the architecture is being developed; future work will complete the prototype and integrate the latest version of SCALe with it.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Static analysis, auditing, static analysis alerts, API, architecture, classification, prioritization	108

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89) Prescribed by ANSI Std. Z39-18
298-102