SCALe v2 and v3 New Features:
Detail and Demo

Lori Flynn
Senior Software Security Researcher

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

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.

DM18-1296

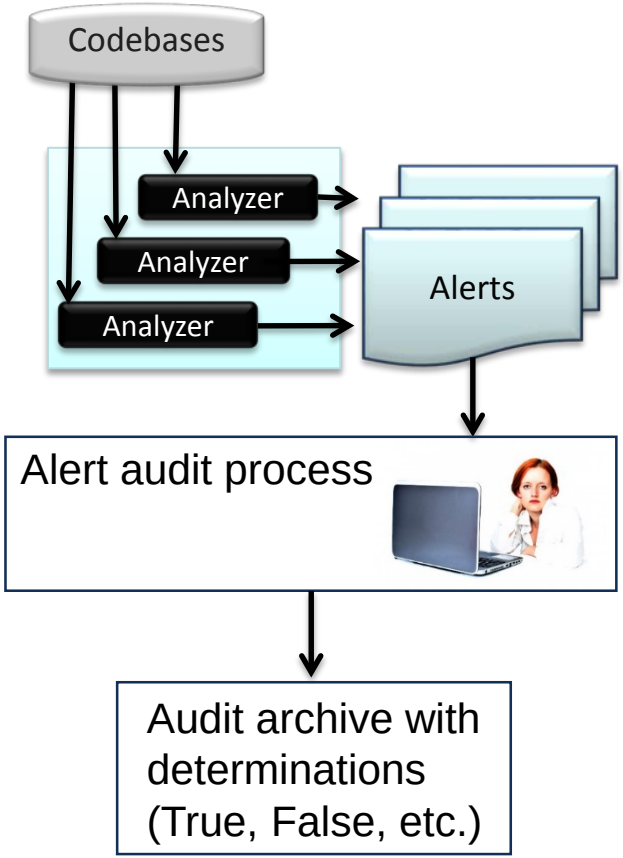
SCALe Static Analysis Alert Auditing Tool

Static analysis (SA) tools examine code without executing it

- Flaw-finding SA tools examine syntax, control flow, data flow, and/or type flow for indicators of particular flaws

SEI CERT's SCALe tool:

- Developed by CERT Secure Coding team since 2010
  - Add new features to enable research
  - Auditors (collaborators & CERT) test new features
- Imports source code plus raw output from SA tools
- Provides GUI to audit alerts and view related code
- Stores audit archive data to exportable database

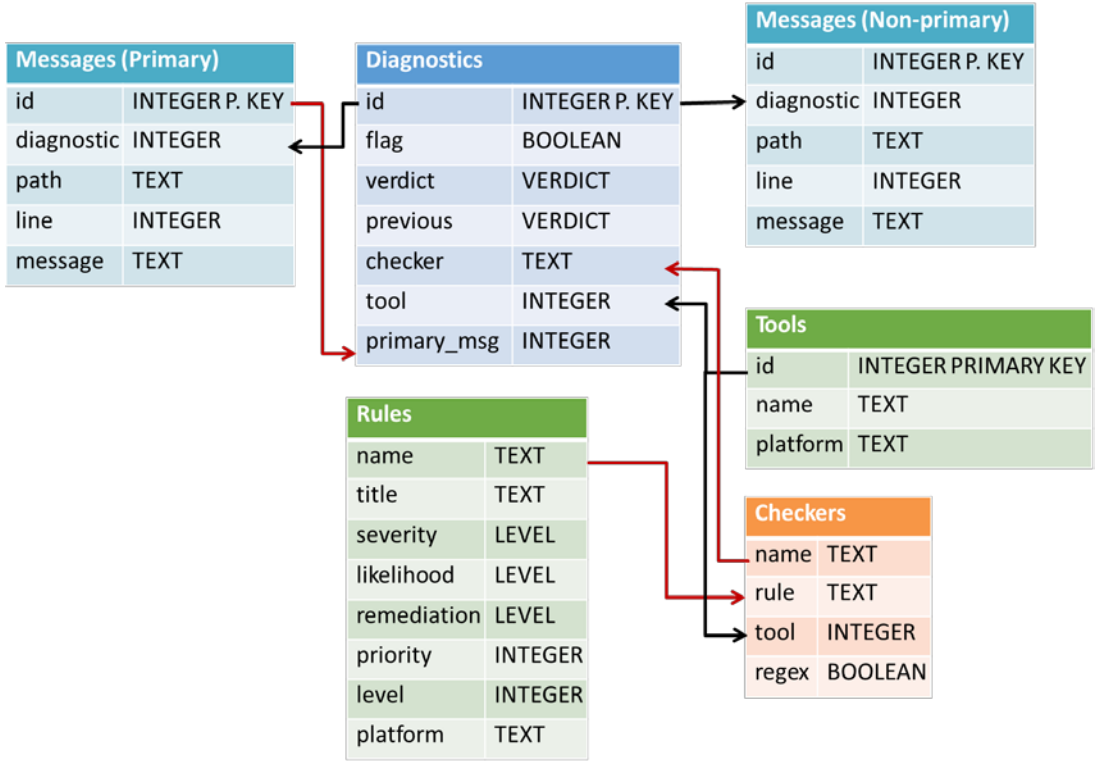


SCALe v1

Previously-released videos and technical reports only show SCALe v1

- First released outside SEI in 2015
- Enabled imports of 6 flaw-finding static analysis tool outputs
- Alert prioritization according to one metric (e.g., CERT rule ‘severity’ or ‘priority’)

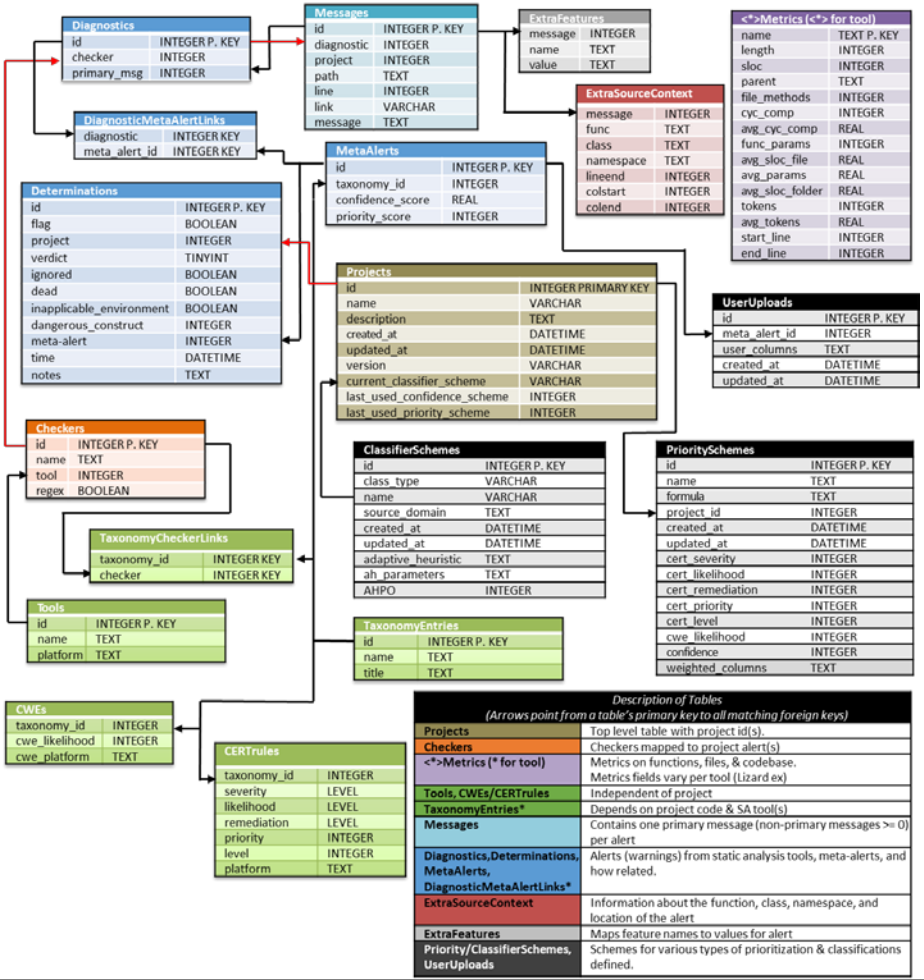
Exported Database Format



SCALe v3 Exported Database Format

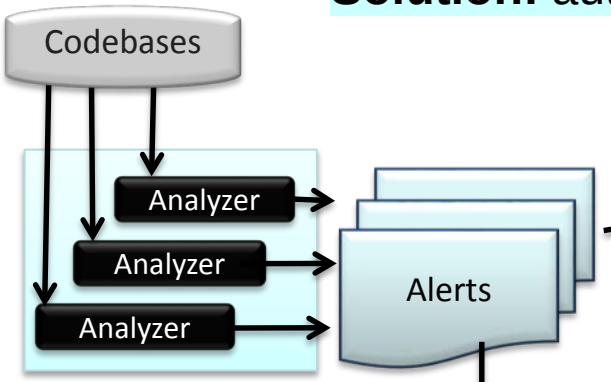
New data for:

- Machine learning classifiers
- Alert prioritization
- Data quality



Classifiers

Problem: too many alerts
Solution: automate handling

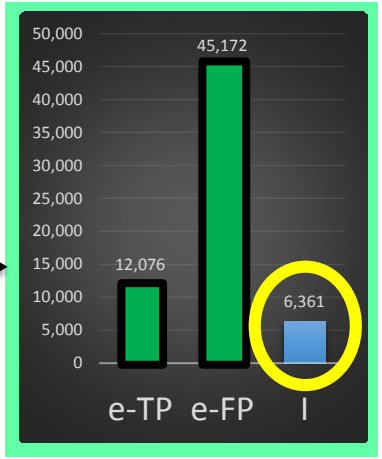


Today

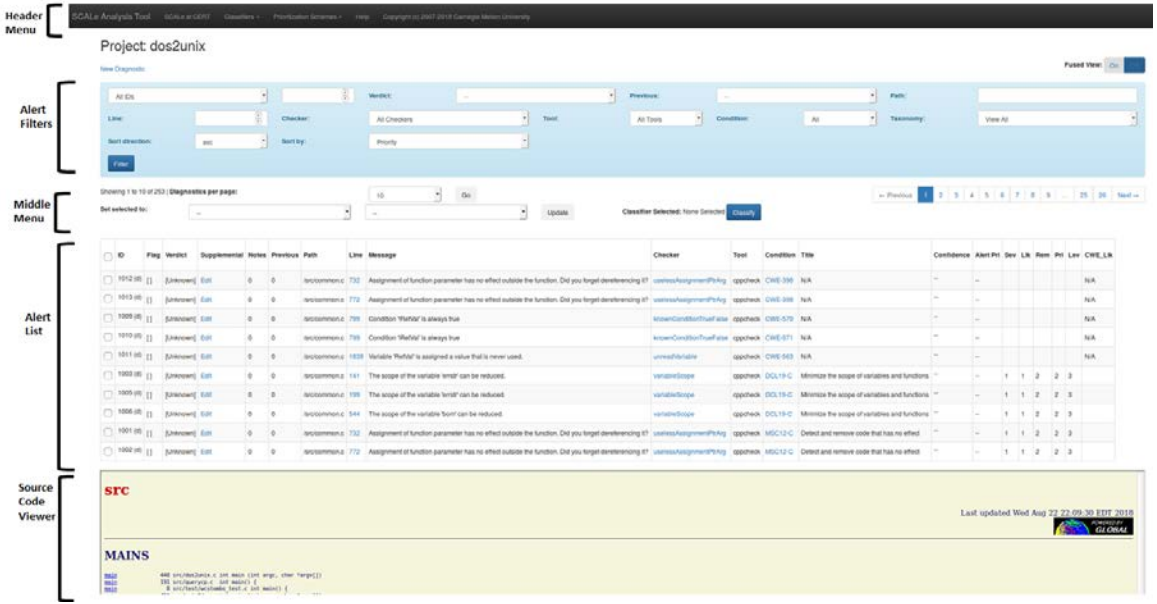
Project Goal

System that automatically and **accurately classifies most of the alerts as:**

Expected True Positive (e-TP) or Expected False Positive (e-FP),
and
the rest as **Indeterminate (I)**



SCALe Development



Used as a research platform

- Extend with new features
- Collaborators give us feedback
- Collaborators generate data required for our classifier research

Over last 3 years, new SCALe features are for classification and prioritization research.

- GitHub public release (SCALe v2), Aug. 2018
- SCALe v3 for research project collaborators

SCALe v2 and v3 Development

Since late 2015 to now, most SCALe development:

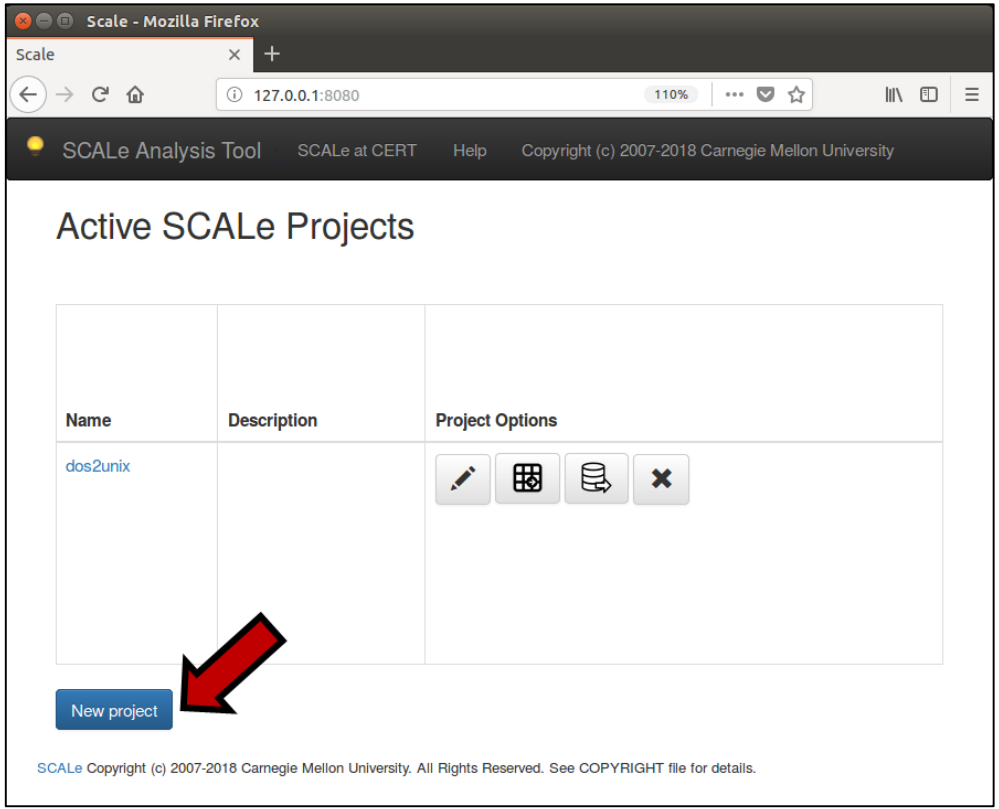
- Added features for classification and prioritization research
  - To provide new types of data for use by classifiers (e.g., as features)
  - To enhance quality of data used to develop classifiers
  - To enable outside organizations to share data with SEI
  - To enable selection of advanced prioritization and classifier schemes
- Done by developers on my research project teams. Including: Ebonie McNeil, David Svoboda, William Snavely, Derek Leung, Jiyeon Lee, Lucas Bengston, Jennifer Burns, Christine Baek, Baptiste Vauthy, Charisse Haruta, Shirley Zhou, Maria Rodriguez De La Cruz, and Elliot Toy.

New Features: Slides then Demo

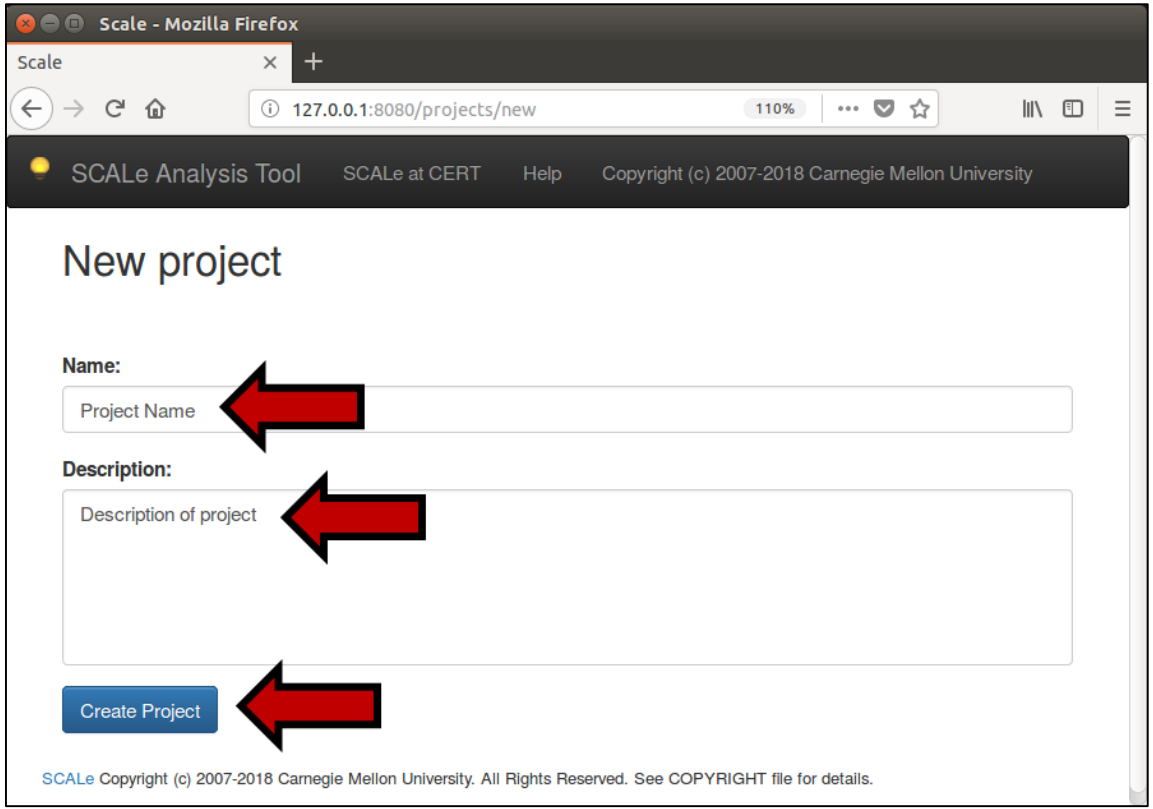
First, we will look at close-ups of the new features in slides.

After that, a demo.

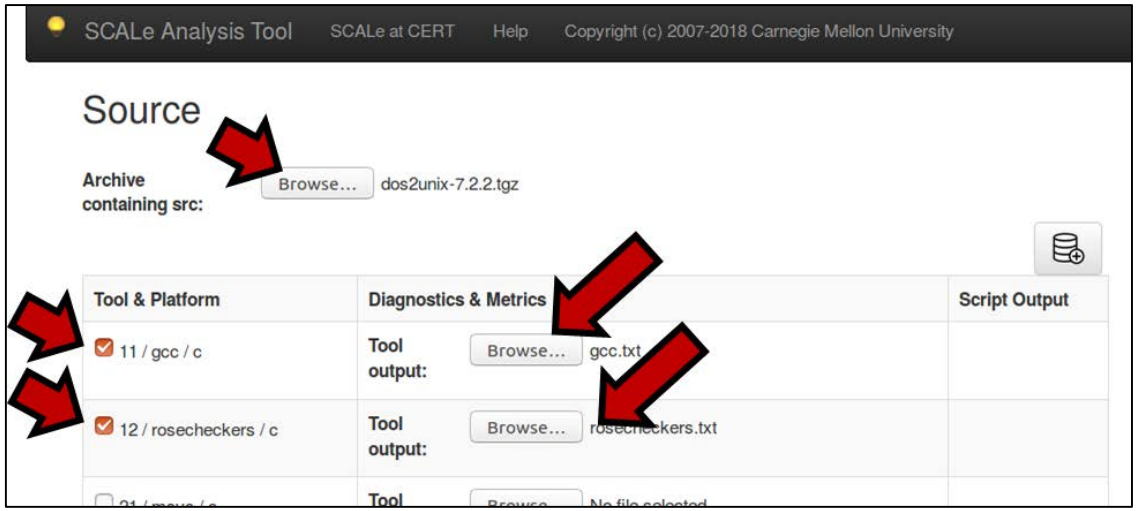
Modified Project Creation



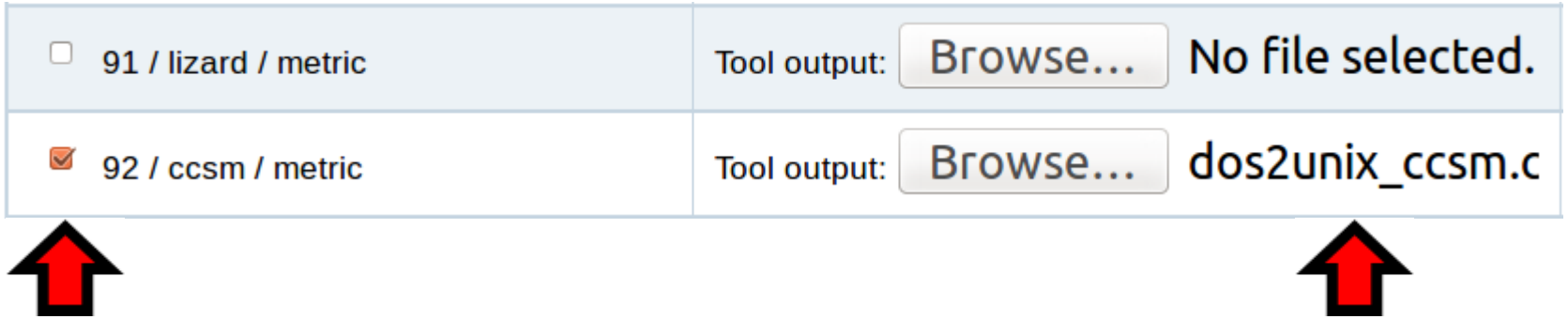
Modified Project Creation



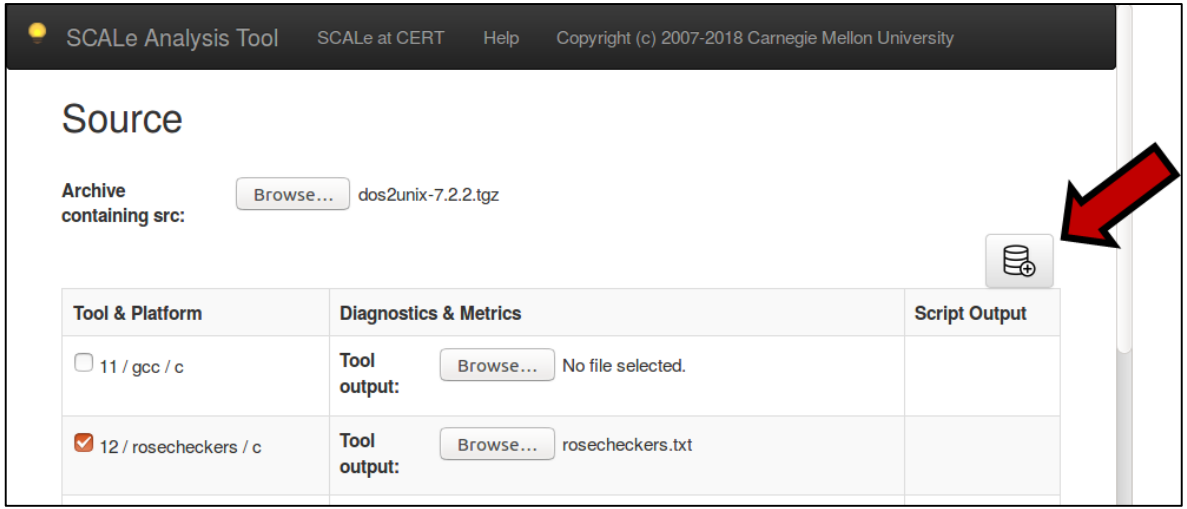
Uploading Source Code and Tool Output



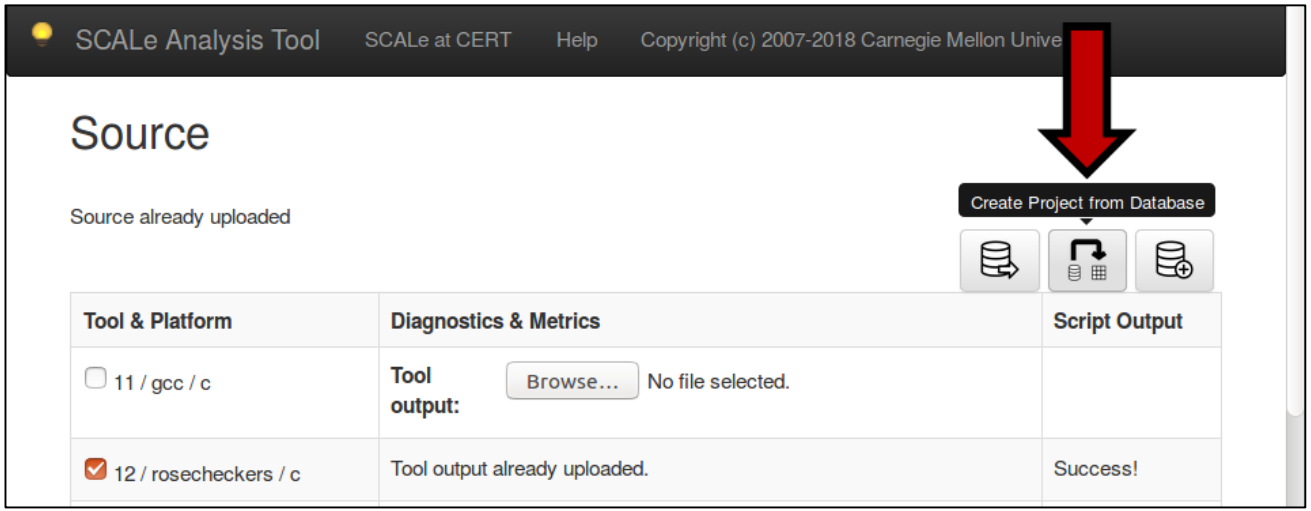
Uploading Code Metrics Tool Output



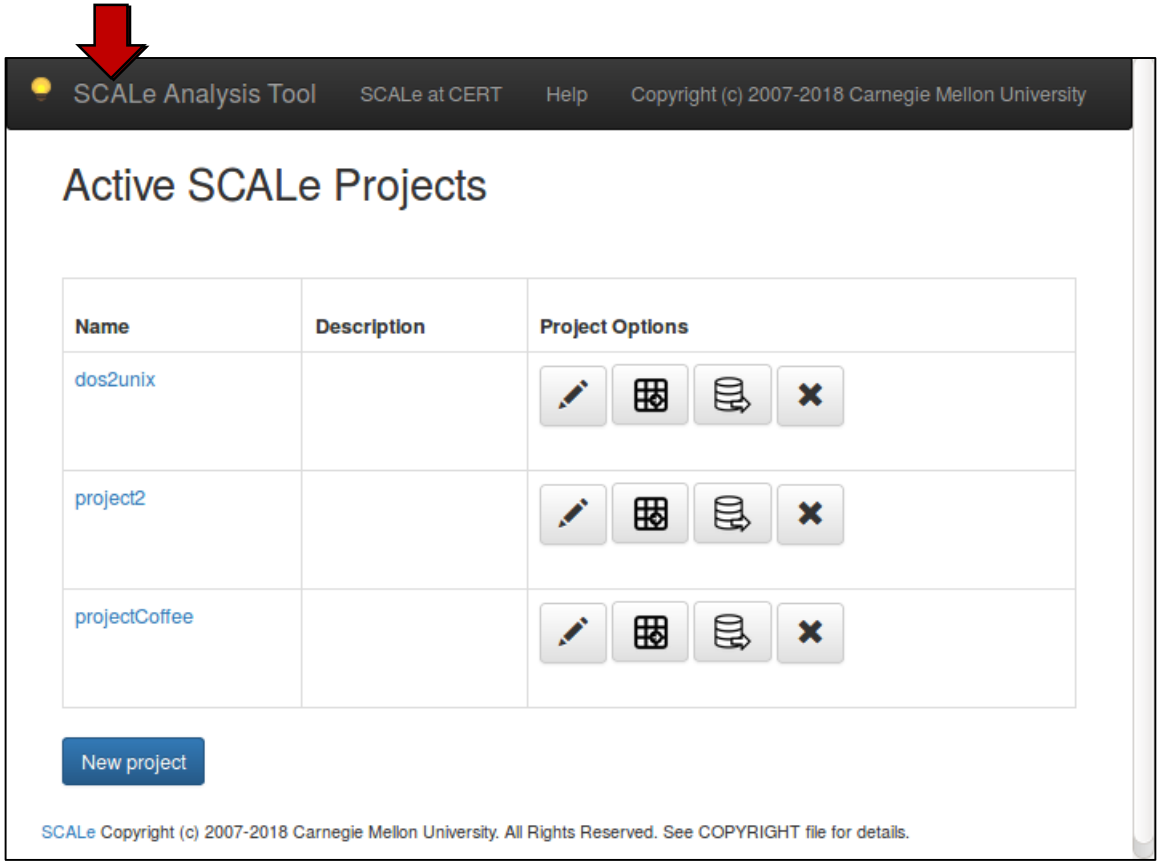
Next, Create Project with Two Icon Selections: Icon #1

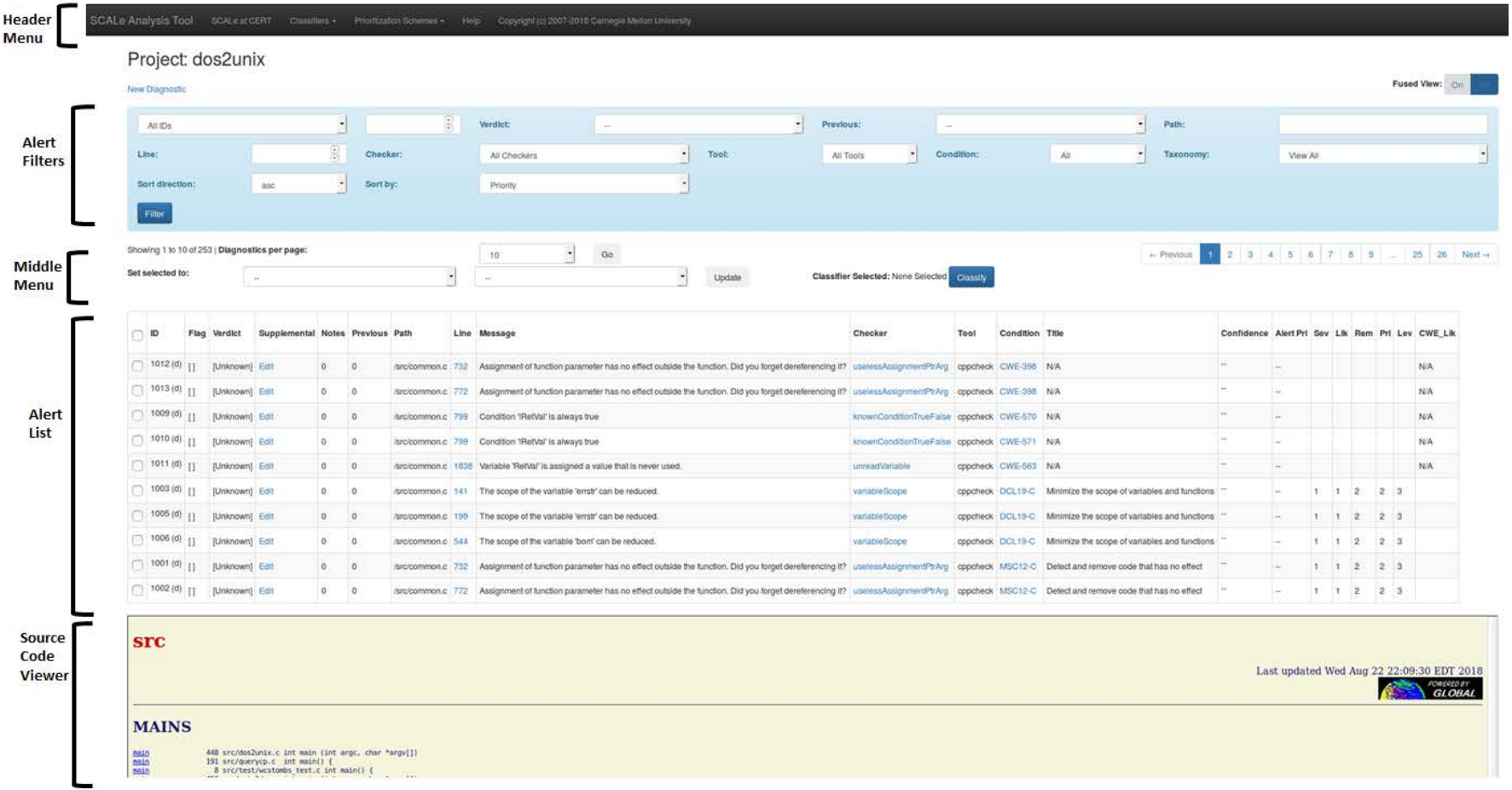


Next, Create Project with Two Icon Selections: Icon #2

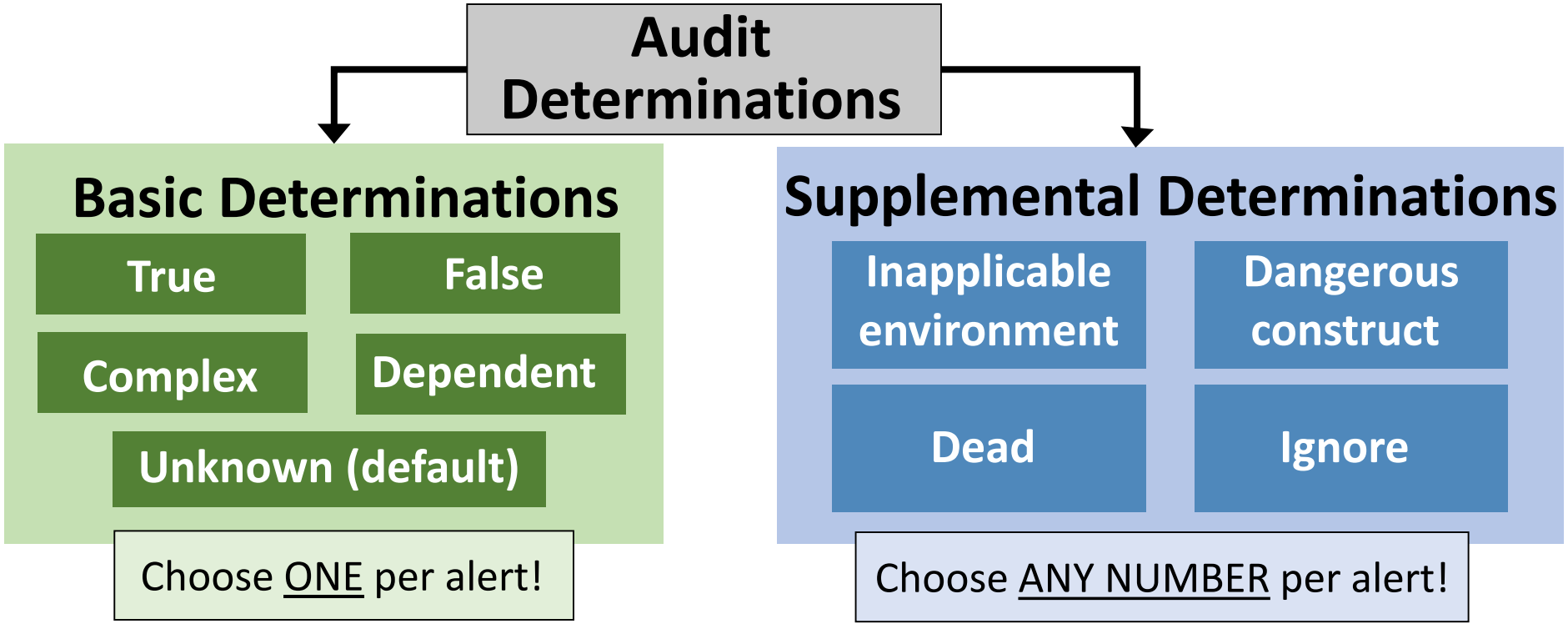


SCALe Homepage



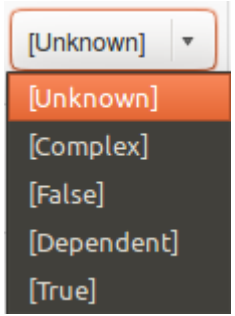


New Features: Audit Determinations



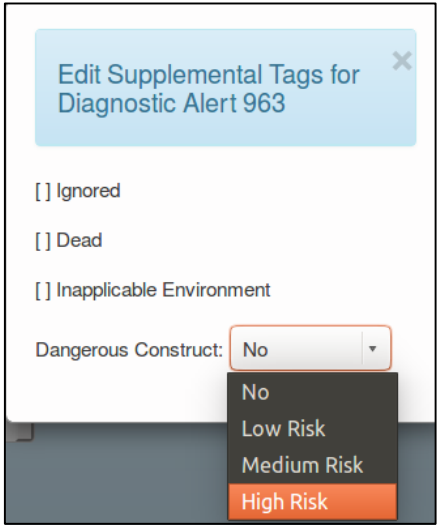
Determinations in GUI

Drop-down for primary verdict



Supplemental determination popup:

- select any number



Flag field can have org-defined meaning

	ID	Flag	Verdict	Supplemental	Notes
☐	963 (d)	[x]	[True]	Ignored Dangerous Construct - Med Edit	0
☐	964 (d)	[ ]	[False]	Ignored Edit	var Y possible integer overflow
☐	953 (d)	[ ]	[Unknown]	Edit	0

New Features: CWE Taxonomy Added

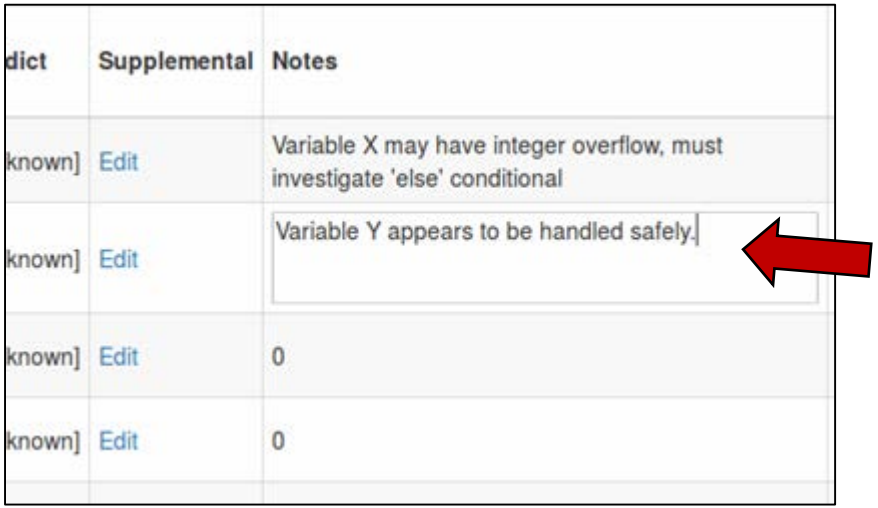
Tool checkers mapped to CWEs and CERT rules.

Checker	Tool	Condition	Title	Confidence	Alert Pri	Sev	Lik	Rem	Pri	Lev	CWE_Lik
uselessAssignmentPtrArg	cppcheck	CWE-398	N/A								N/A
uselessAssignmentPtrArg	cppcheck	CWE-398	N/A								N/A
knownConditionTrueFalse	cppcheck	CWE-570	N/A								N/A
knownConditionTrueFalse	cppcheck	CWE-571	N/A								N/A
unreadVariable	cppcheck	CWE-563	N/A								N/A
variableScope	cppcheck	DCL19-C	Minimize the scope of variables and functions			1	1	2	2	3	
variableScope	cppcheck	DCL19-C	Minimize the scope of variables and functions			1	1	2	2	3	
variableScope	cppcheck	DCL19-C	Minimize the scope of variables and functions			1	1	2	2	3	
uselessAssignmentPtrArg	cppcheck	MSC12-C	Detect and remove code that has no effect			1	1	2	2	3	
uselessAssignmentPtrArg	cppcheck	MSC12-C	Detect and remove code that has no effect			1	1	2	2	3	

- Some CWEs have CWE Likelihood.
- Can filter by CWE or CERT Rules taxonomy
- Can filter for single rule/CWE

New Feature: Notes

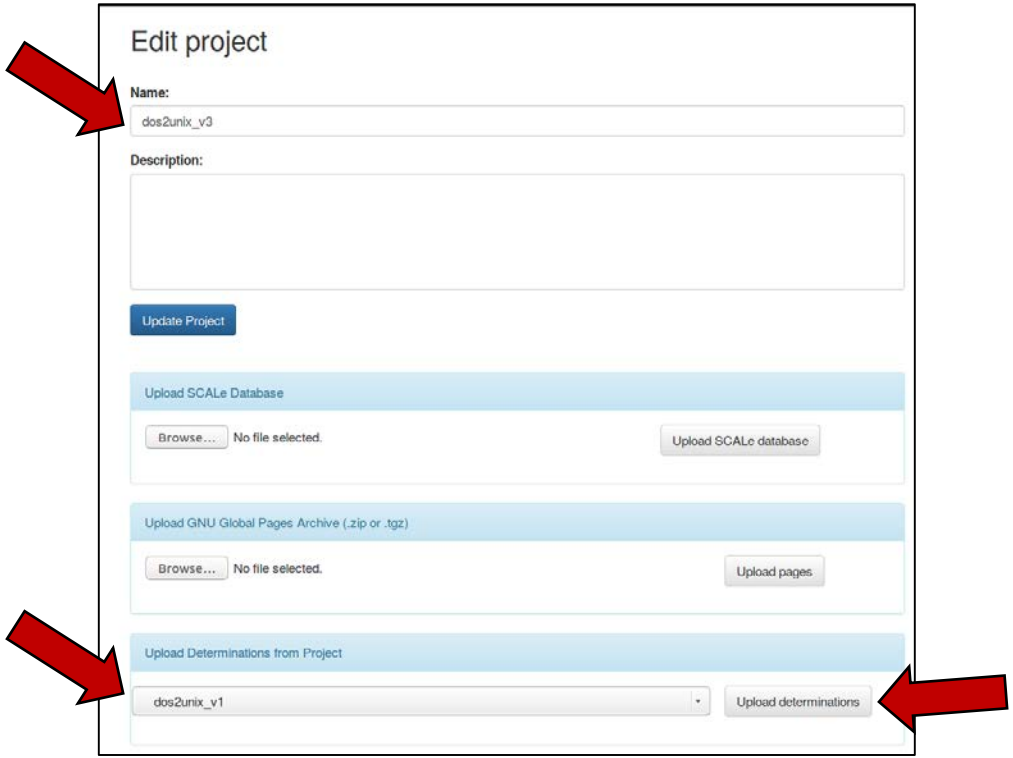
- Notes by auditor about determinations, alert, meta-alert, checker, condition, or language.
- The text can help later auditors reviewing same or similar issues.



New Features: Cascade Determinations

Edit project

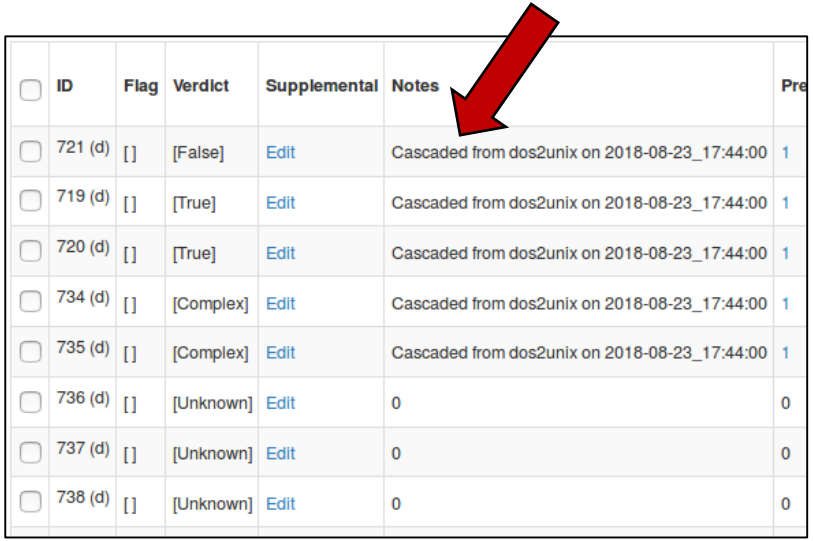
- Upload determinations from same tool on previous version of code
- Uses `diff` for line matches
- Match alert and line, then auto-cascade determination
- Caution: Data, control, and type flow changes may cause a previously-correct determination to change.



After Cascaded Import

After cascaded import

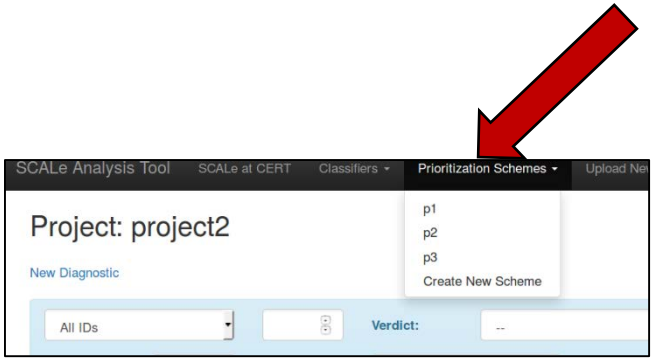
- Notes field show determination was cascaded
- Database records note about cascaded determination

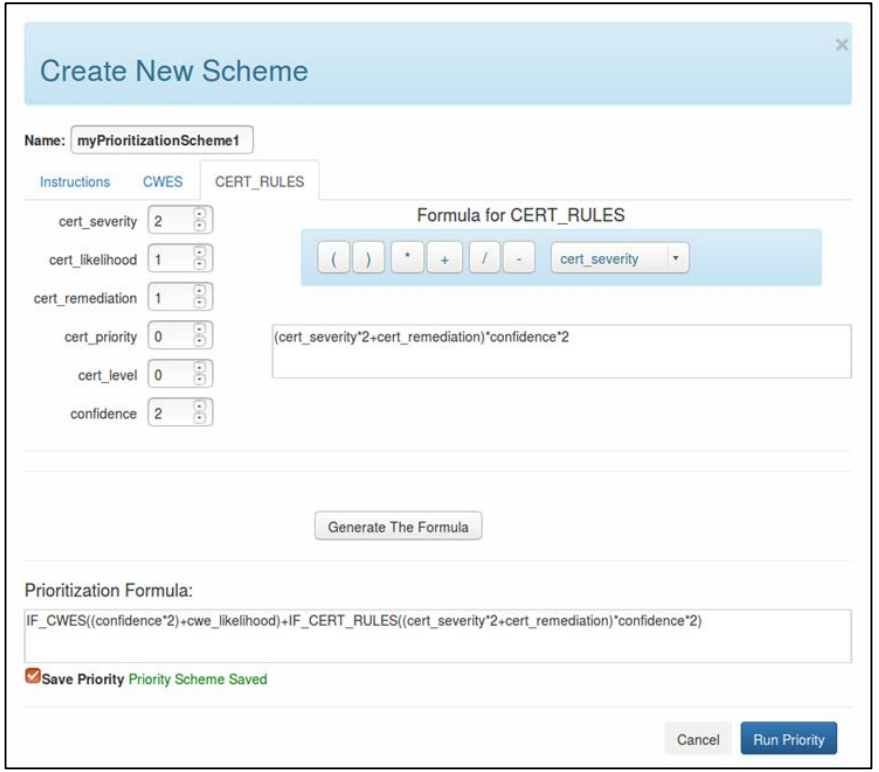


	ID	Flag	Verdict	Supplemental	Notes	Pre
☐	721 (d)	[ ]	[False]	Edit	Cascaded from dos2unix on 2018-08-23_17:44:00	1
☐	719 (d)	[ ]	[True]	Edit	Cascaded from dos2unix on 2018-08-23_17:44:00	1
☐	720 (d)	[ ]	[True]	Edit	Cascaded from dos2unix on 2018-08-23_17:44:00	1
☐	734 (d)	[ ]	[Complex]	Edit	Cascaded from dos2unix on 2018-08-23_17:44:00	1
☐	735 (d)	[ ]	[Complex]	Edit	Cascaded from dos2unix on 2018-08-23_17:44:00	1
☐	736 (d)	[ ]	[Unknown]	Edit	0	0
☐	737 (d)	[ ]	[Unknown]	Edit	0	0
☐	738 (d)	[ ]	[Unknown]	Edit	0	0

Prioritization Schemes

Prioritization schemes with mathematical formulas user can create and/or use





User Field Uploads

User field uploads

- For advanced users that can work with SQL databases and generate values
- Uploaded fields can be used in priority scheme
- CSV uploaded file
  - One line per project meta-alert ID
  - Left-most field has meta-alert ID
  - Top row holds field labels

```
meta_alert_id,safeguard_countermeasure,
vulnerability,residual_risk,impact,
threat,risk,complexity,severity,coupling
112,5,1,4,9,1,1,5,5,1
2,9,3,3,3,1,1,1,9,3
3,3,1,1,1,8,1,5,5,1
4,6,1,1,5,2,1,8,8,1
5,2,1,1,2,3,1,7,7,5
6,5,1,4,4,1,2,4,5,1
7,8,5,3,4,8,2,4,9,9
8,2,1,3,2,8,3,8,8,1
9,6,4,3,6,9,1,4,4,4
10,3,2,2,5,7,1,4,5,9
11,6,1,1,9,6,1,7,7,1
12,2,8,4,1,6,1,4,4,8
```

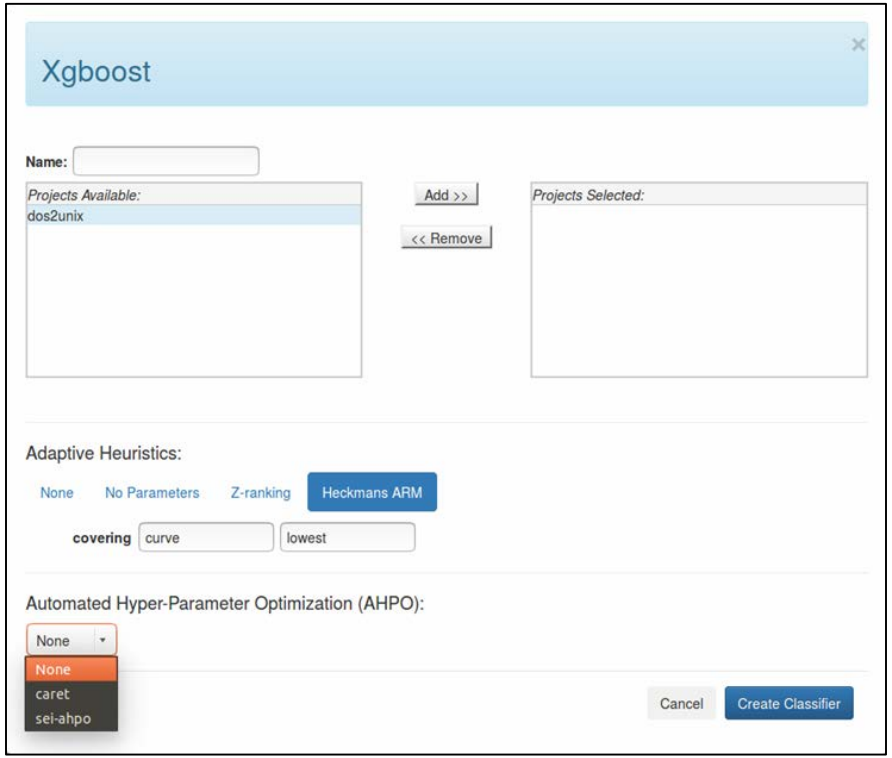
Classification Scheme

Select projects with audited alerts to develop classifier with

Select

- Type of classifier
- Type of adaptive heuristic
- Type automated hyper-parameter classification

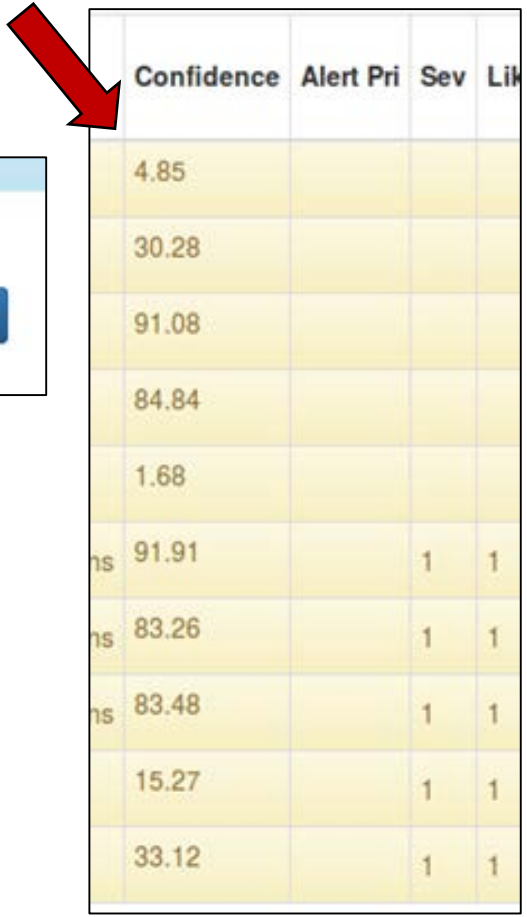
Then create the classifier



Run the Classifier on a Project

Select ‘Classify’ button to run the classifier on a project

- Classifier predicts alert determinations
- When fully functional, this will cause meta-alerts to be classified
- Currently, example metrics are loaded for the 'Confidence' field
  - Usability demonstration only
  - Values not currently from classifier



Alert Fusion

- Alert fusion for {filepath, line, condition} reduces auditor effort
  - Multiple tools may indicate the same flaw
  - Make determination one time
  - See messages and insight about the flaw from all the tools at once

Screenshot shows fused (yellow) and unfused alerts.

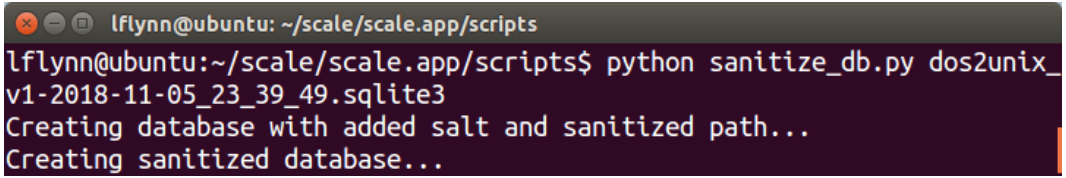
- Fused alerts not expanded here (proprietary tools).

☐	968 (d)	[ ]	[Unknown]	Edit	0	0	/src/dos2unix.c	358	Guarantee that array indices are within the valid range	ARR30-C	rosecheckers	ARR30-C	Do not form or use out-of-bounds pointers or array subscripts			3	3	1	9	2
☐	277 (m)	[ ]	[Unknown]	Edit	0	0	/src/dos2unix.c	358				INT32-C	Ensure that operations on signed integers do not result in overflow			3	3	1	9	2
☐	969 (d)	[ ]	[Unknown]	Edit	0	0	/src/dos2unix.c	393	Guarantee that array indices are within the valid range	ARR30-C	rosecheckers	ARR30-C	Do not form or use out-of-bounds pointers or array subscripts			3	3	1	9	2
☐	281 (m)	[ ]	[Unknown]	Edit	0	0	/src/dos2unix.c	393				INT32-C	Ensure that operations on signed integers do not result in overflow			3	3	1	9	2
☐	970 (d)	[ ]	[Unknown]	Edit	0	0	/src/unix2dos.c	357	Guarantee that array indices are within the valid range	ARR30-C	rosecheckers	ARR30-C	Do not form or use out-of-bounds pointers or array subscripts			3	3	1	9	2
☐	285 (m)	[ ]	[Unknown]	Edit	0	0	/src/unix2dos.c	357				INT32-C	Ensure that operations on signed integers do not result in overflow			3	3	1	9	2
☐	971 (d)	[ ]	[Unknown]	Edit	0	0	/src/unix2dos.c	390	Guarantee that array indices are within the valid range	ARR30-C	rosecheckers	ARR30-C	Do not form or use out-of-bounds pointers or array subscripts			3	3	1	9	2
☐	289 (m)	[ ]	[Unknown]	Edit	0	0	/src/unix2dos.c	390				INT32-C	Ensure that operations on signed integers do not result in overflow			3	3	1	9	2

New Feature: Archive Sanitizer

Added data sanitizer script

- Anonymizes sensitive fields
- SHA-256 hash with salt
- Enables analysis of features correlated with alert confidence



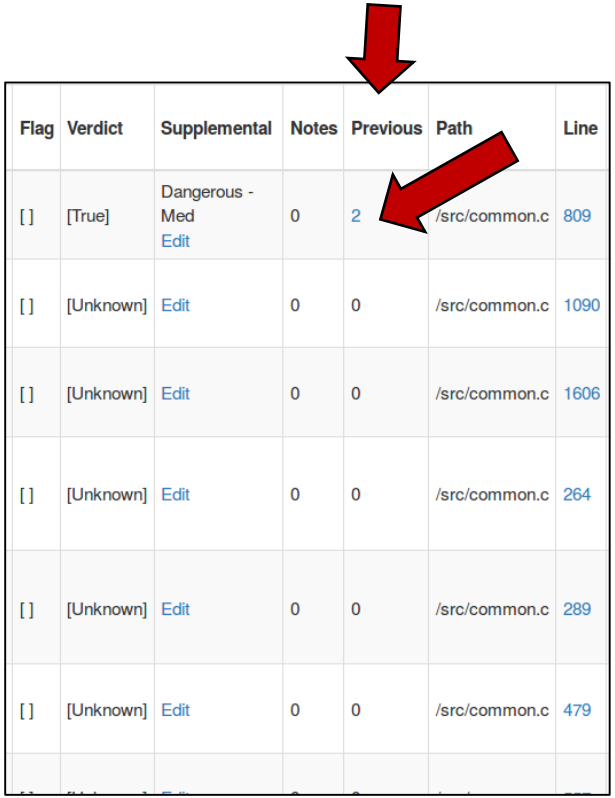
Audit archive for project is in a database

- DB fields may contain sensitive information
- Sanitizing script anonymizes or discards fields
  - Diagnostic message
  - Path, including directories and filename
  - Function name
  - Class name
  - Namespace/package
  - Project filename

Caution: GitHub sanitizer not fully updated for SCALe v2 database – don't count on it.

New Feature: Determination History

History kept of primary and supplemental determinations, notes, and flag



Flag	Verdict	Supplemental	Notes	Previous	Path	Line
[ ]	[True]	Dangerous - Med Edit	0	2	/src/common.c	809
[ ]	[Unknown]	Edit	0	0	/src/common.c	1090
[ ]	[Unknown]	Edit	0	0	/src/common.c	1606
[ ]	[Unknown]	Edit	0	0	/src/common.c	264
[ ]	[Unknown]	Edit	0	0	/src/common.c	289
[ ]	[Unknown]	Edit	0	0	/src/common.c	479

Additional Information for alert 483

Supplemental Messages

Message	Line	Path
Use typedefs to improve code readability	809	/src/common.c

Determination Log

Time	Flag	Verdict	Supplemental	Notes
2018-11-06 05:30:29 UTC	[]	[Unknown]		0
2018-11-06 05:57:29 UTC	[]	[True]		0
2018-11-06 05:57:43 UTC	[]	[True]	Dangerous Construct - Med	0

Hyperlinked Checker

Link to meta-alerts for that line, file, and checker

- May be multiple conditions (e.g, a CWE and a CERT rule)
- Helps auditor see related information, including related determinations

Select hyperlink to see list

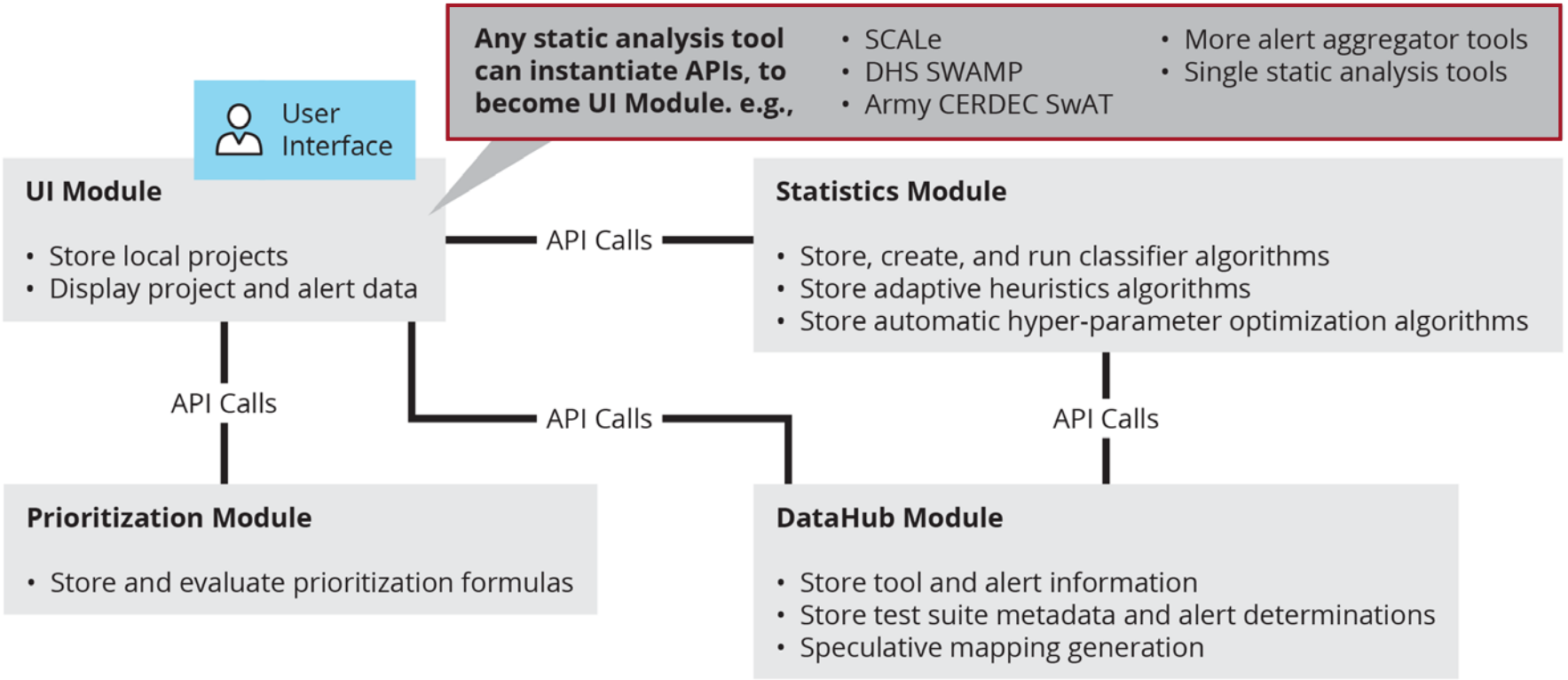
Line	Message	Checker	Tool	Condition
732	Assignment of function parameter has no effect outside the function. Did you forget dereferencing it?	uselessAssignmentPtrArg	cppcheck	CWE-398
772	Assignment of function parameter has no effect outside the function. Did you forget dereferencing it?	uselessAssignmentPtrArg	cppcheck	CWE-398
799	Condition '!RetVal' is always true	knownConditionTrueFalse	cppcheck	CWE-570

All meta-alerts for checker + location

Path	Line	Message	Checker	Tool	Condition
/src/common.c	799	Condition '!RetVal' is always true	knownConditionTrueFalse	cppcheck	CWE-570
/src/common.c	799	Condition '!RetVal' is always true	knownConditionTrueFalse	cppcheck	CWE-571
/src/common.c	799	Condition '!RetVal' is always true	knownConditionTrueFalse	cppcheck	MSC07-C

Demo

Architecture



Architecture Development

Representational State Transfer (REST)

- Architectural style that defines a set of constraints and properties based on HTTP
- RESTful web services provide interoperability between systems
- Client-server

We chose to develop a RESTful API

- Swagger/OpenAPI open-source development toolset
  - Develop APIs
  - Auto-generate code for server stubs and clients
  - Test server controllers with GUI
  - Wide use (10,000 downloads/day)

SCALe Development for Architecture Integration

SCALe will make UI Module API calls in prototype system.

- Other alert auditing tools (e.g., DHS SWAMP) also can instantiate UI Module API.

Next Steps and Collaboration Opportunities

Code development to complete 4-server system instantiation with SCALe as UI Module

- Collaboration opportunities:
  - Implementation of API by collaborators to extend their own alert auditing tools
    - Feedback on API, code system, and adaptive heuristics
  - Alert audit data needed (sanitized fine)
  - **Additional ideas welcome!**

References

- Paper “Static Analysis Alert Audits: Lexicon & Rules”, IEEE Cybersecurity Development Conference, Nov 2016.
- GitHub SCALe v2 publication Aug. 2018
- Paper “Prioritizing Alerts from Multiple Static Analysis Tools, using Classification Models,” SQUADE (ICSE workshop)
- SEI blog post: “Test Suites as a Source of Training Data for Static Analysis Alert Classifiers” (Apr. 2018)
- SEI Podcast (video): “Static Analysis Alert Classification with Test Suites” (Sep. 2018)
- SEI blog post: “SCALe: A Tool for Managing Output from Static Code Analyzers” (Sep. 2018)
- SEI Technical Report “Integration of Automated Static Analysis Alert Classification and Prioritization with Auditing Tools” (Publication expected November 2018)
- Presentation Automating Static Analysis Alert Handling with Machine Learning: 2016-2018 (Oct. 2018)

Contact Information

Lori Flynn
Senior Software Security Researcher
Telephone: +1 412.268.7886
Email: lflynn@sei.cmu.edu