SCALe v2 and v3 New Features

Table of Contents

SCALe v2 and v3 New Features: Detail and Demo

**001 Presenter: And hello from the campus of Carnegie Mellon University in Pittsburgh, Pennsylvania. We welcome you to Virtual SEI. Our presentation today is "Improve Your Static Audits Using CERT SCALe's New Features". My name is Shane McGraw. I'll be your audience moderator for today's presentation, and I'd like to thank you for attending. We want to make today as interactive as possible, so we will address questions throughout today's presentation and again at the end of the talk. You can submit those questions at any time through the Chat or Q&A tabs on your page interface now. Also we ask that you will fill out a survey upon exiting today's event, as your feedback is always greatly appreciated, and a link to that survey will be added to the chat area soon.

And now I'd like to introduce our speaker for today. Dr. Lori Flynn is a senior software security researcher within the CERT division at SEI. Her research focuses on automated software security analysis, and in past work she co-invented a patented static analysis method that creates signatures for polymorphic viruses. Lori, welcome. All yours. Take it away.

Presenter: Thank you, Shane. So today I'm going to talk about the SCALe v2 and v3 new features. I'll talk about details of the new versions of the tool and then demonstrate them.

SCALe Static Analysis Alert Auditing Tool

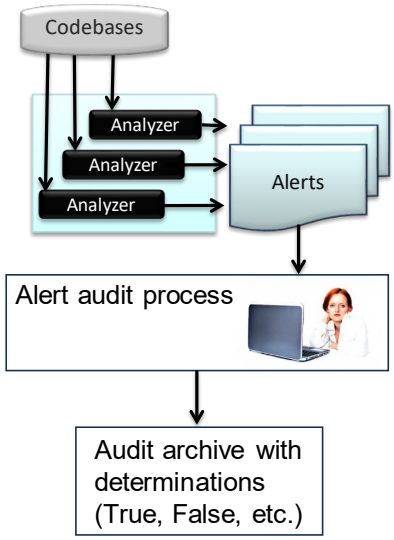
SCALe Static Analysis Alert Auditing Tool

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



**003 Static analysis tools examine code without executing the code. Flaw-finding static analysis tools examine syntax, control flow, data flow, and/or type flow for indicators

of particular flaws. SEI CERT's SCALe tool has been developed by the CERT Secure Coding Team since 2010. We add new features to enable research, and auditors, both collaborators and CERT, test the new features and provide data enabled by the new features.

SCALe imports source code plus raw output from static analysis tools. It provides a GUI to audit alerts and to view related code, and stores audit archive data to an exportable database. As you see on the right of the slide, the process is shown, starting from the top, where code bases are analyzed by one or more static analysis tools, each of which outputs a set of alerts.

Then there's an alert auditing process, shown here with the figure of the engineer, who is analyzing the alerts manually, and that results in an audit archive that includes the determinations and the alert information. So the determinations can be true, false, and some other things we'll talk about later in this presentation.

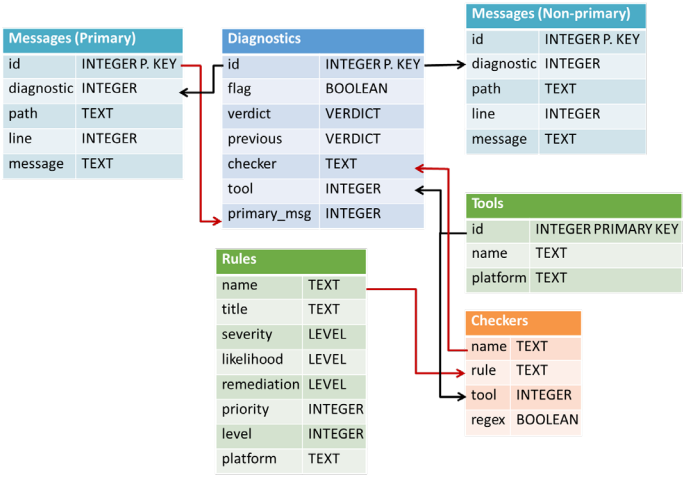
SCALe v1

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')

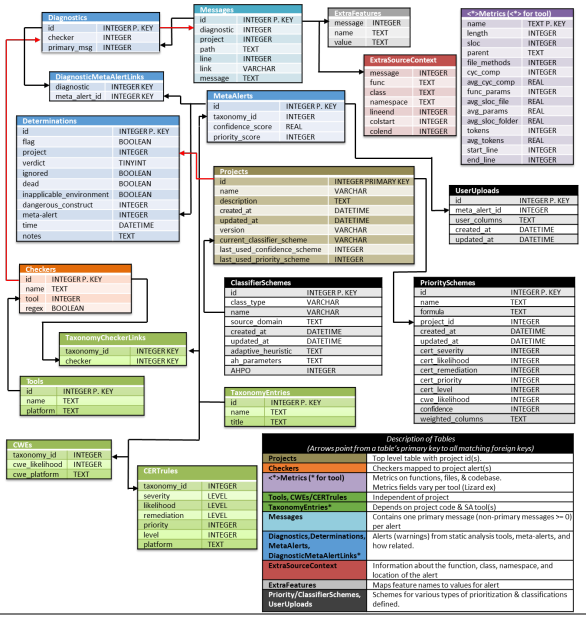
Exported Database Format



**004 Previously released videos and technical reports from the SEI only show SCALe v1. That was first released outside of the SEI in 2015, and it enabled imports of six flaw-finding static analysis tool outputs. Alert prioritization in v1 was done according to one metric, such as the per-CERT-rule severity or priority values, and the exported database format for v1 is shown to the right here. You can see that there are six tables-- it's a SQLite database-- and the tables contain information about the messages from the static analysis tools.

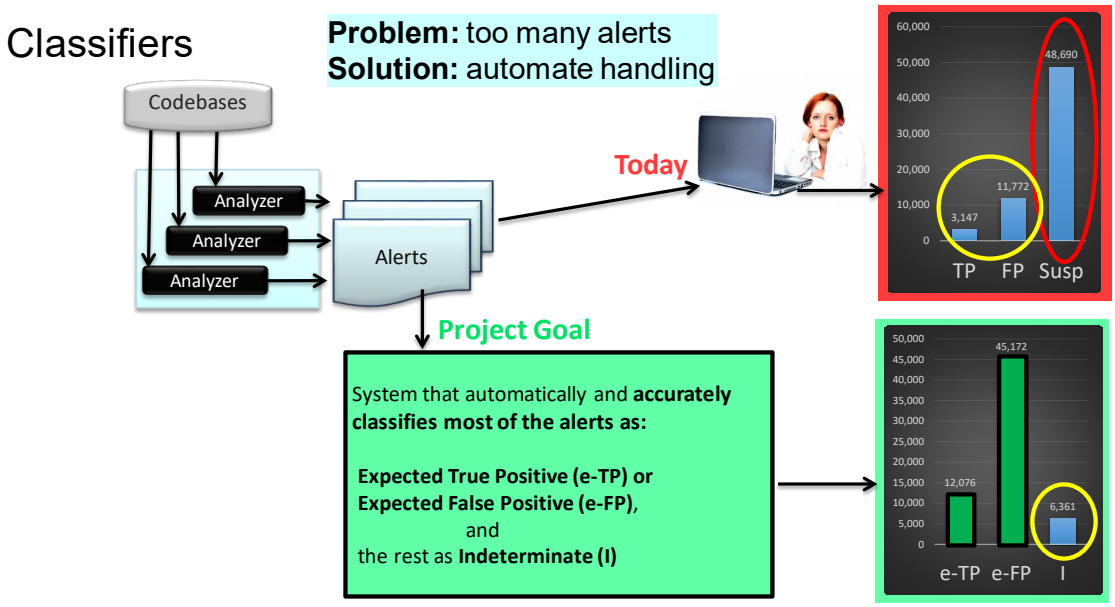
The checker ID from the static analysis tools, that's a condition that indicates a particular type of flaw that identifies what the tool looks for, and one of the tables has mappings between CERT coding rules and those per-tool checkers.

SCALe v3 Exported Database Format



**005 The SCALe v3 exported database format is shown on this slide. You can see we've added a lot, just looking at the number of tables and the number of fields. The new data is for machine learning classifiers, alert prioritization, and data quality.

Problem: too many alerts Solution: automate handling



**006 So this classifier research that we're doing that caused us to add that data to the exported database-- let's talk about that a little bit here. The problem that it addresses is that there are too many static analysis alerts for organizations to handle with manual auditing processes, and the solution this research works with and tries to improve is automating handling of alerts.

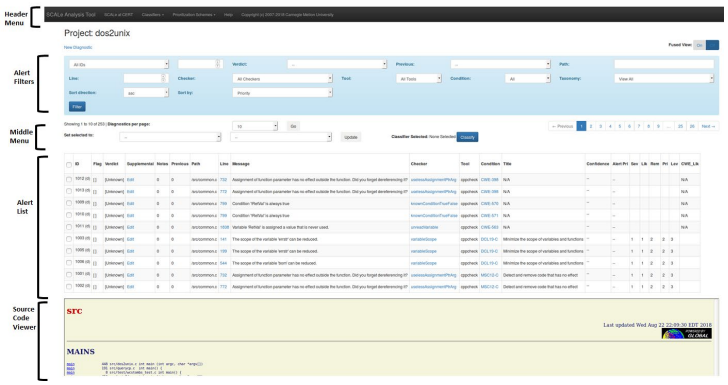
So again, you see the process starting from the left top this time, where code bases are run through one or more static analysis tools, outputting alerts. Today-- following the arrows on the top-- manual auditing is done in most cases, resulting in the chart shown at the top right, where some alerts have been manually determined to be true or false-positives, but many, many alerts are not able to be examined,

and so they're just suspicious, unknown. There might be severe security flaws which there were static analysis alerts for but they were never manually examined.

So the goal of our research on classifiers is to develop a system that automatically and accurately classifies most of the alerts as expected true-positive, expected false-positive, and the rest as indeterminate, where the number of indeterminate alerts is very small and there's enough organizational effort to be able to manually audit those.

SCALe Development

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

**007 So back to SCALe. SEI uses SCALe as a research platform where we extend it with new features, collaborators give us feedback, and they generate data required for our classifier research. Over the last

three years, new SCALe features have been added for classification and prioritization research, resulting in the SCALe v2 public release-- that's in GitHub-- in August this year- that was the first ever SCALe public release-- and SCALe v3, which we've released to research project collaborators in August and September. So a small screenshot of that is shown to the left here, but we'll show you a bigger screenshot and talk about it later in this presentation.

SCALe v2 and v3 Development

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.

**008 The new features that have been added for this classification and prioritization research provide new types of data for use by classifiers. They enhance the quality of data used to develop classifiers, with the intention to produce more accurate classifiers; they enable organizations

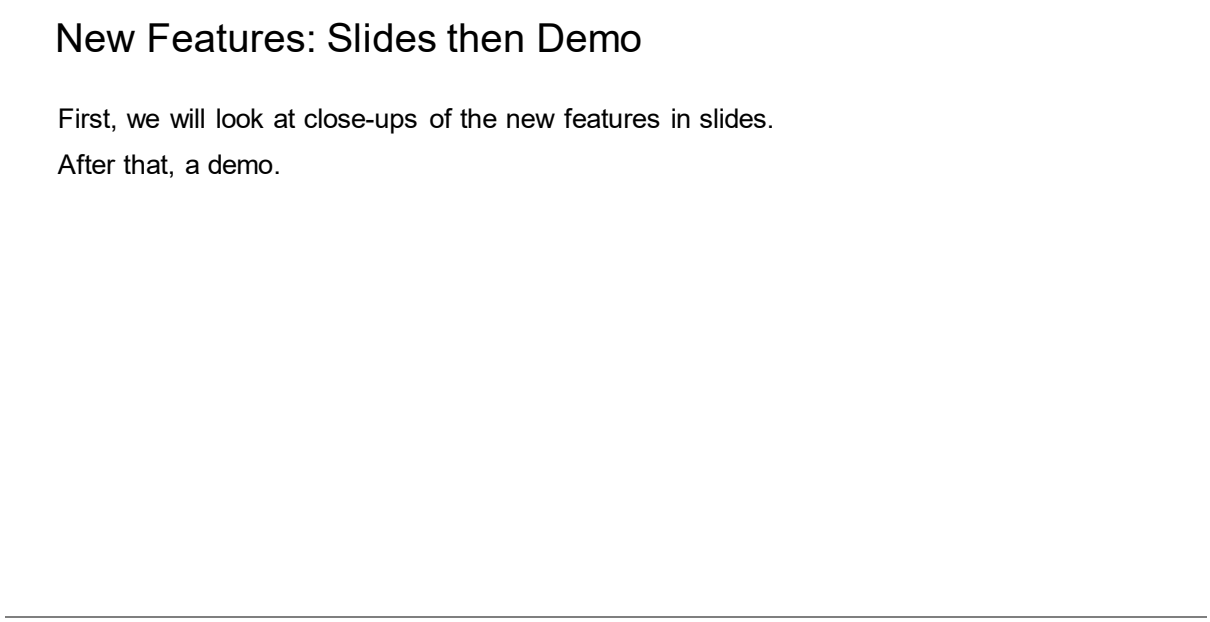
outside the SEI to share their data with SEI without sharing sensitive data; and they enable selection of prioritization and classification schemes.

So this development has been done by developers on my research project teams who are named here. There's a long list, and many thanks to them.

Presenter: So we had a quick question from Joseph asking, "I'm assuming this is going to be on GitHub as well, v3, and when would that be available?"

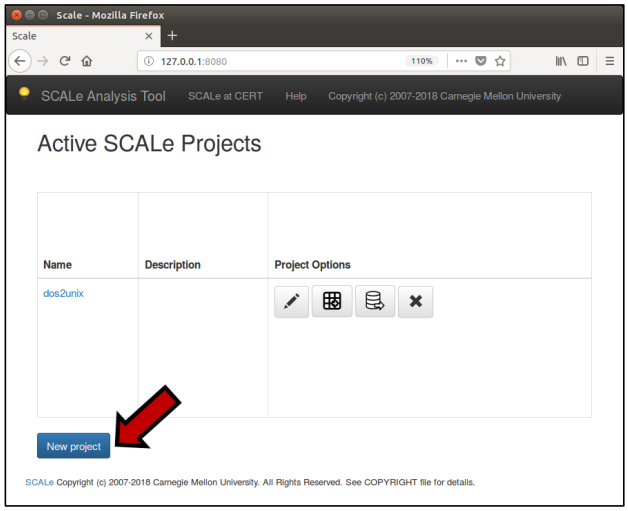
Presenter: Yes. So eventually the intention is to publish v3 on GitHub, and that will likely happen at the end of September, which this research project goes to the end of September, so then we'll publish that latest version after.

New Features: Slides then Demo



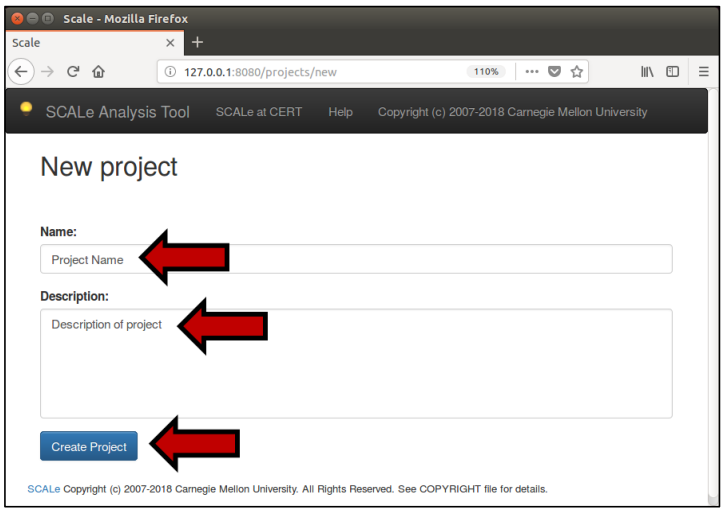
**009 For the rest of this presentation, I'll first of all show slides that detail close-ups of the new features, and after that, I'll demonstrate using the new features in SCALe.

Modified Project Creation



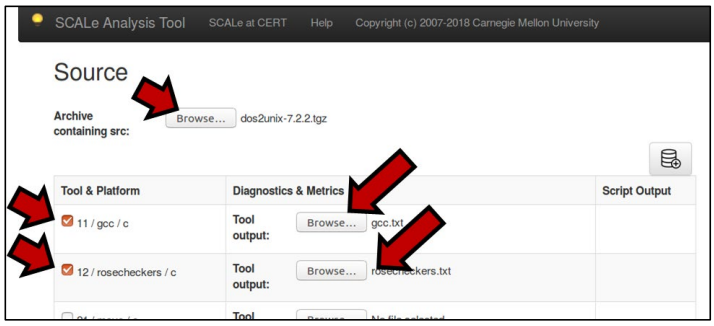
**010 So we modified project creation, the interface, using the Firefox web browser. It looks like the screenshot here. To create a new project you click on that left bottom button with the red arrow to it.

Modified Project Creation



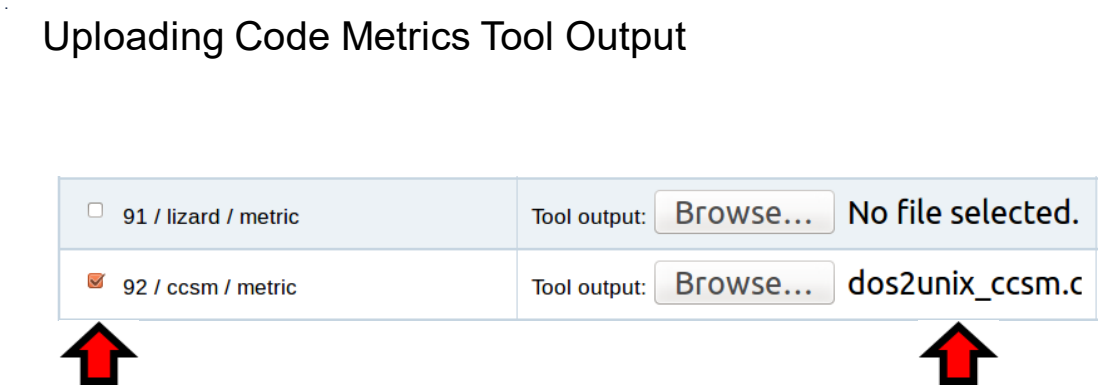
**011 Then you enter the project name and a description and click that Create Project button.

Uploading Source Code and Tool Output



**012 And next you upload source code and tool output. You upload source code as a tarball or as a zip file, with the arrow on the top, and then tool output shown in this example is GCC compiler output warnings, and the Rosecheckers static analysis tool, and you select the files with the raw output with this interface.

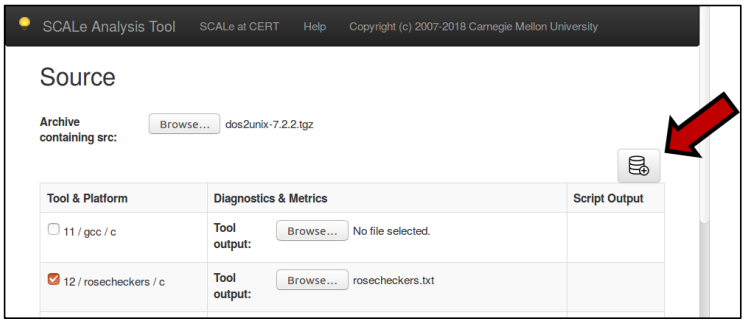
Uploading Source Code and Tool Output



**013 And next, you can also upload code metrics tool output from this interface. In this example, the CCSM Clang plugin output is being uploaded.

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

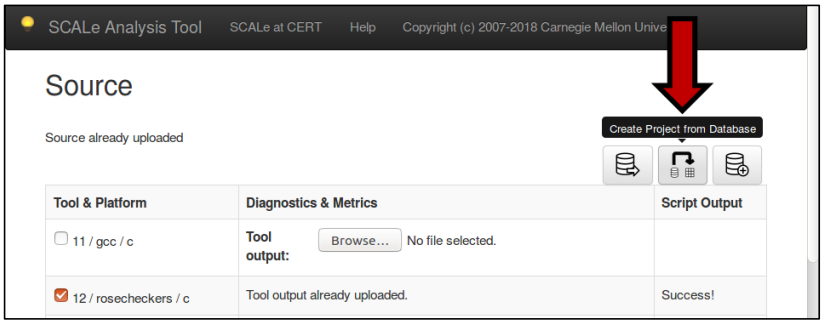
Next, Create Project with Two Icon Selections: Icon #1



**014 Next, you create a project using two icon selections. Icon number one is shown here at the top right.

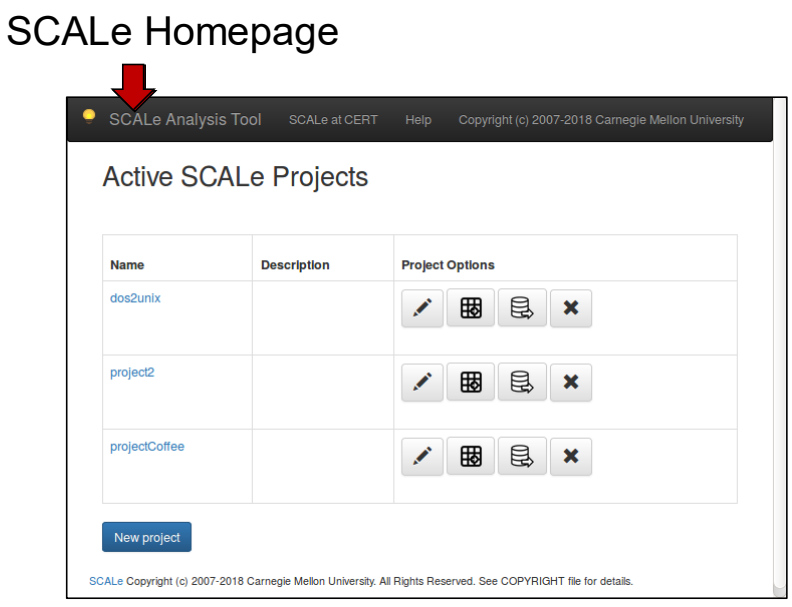
Next, Create Project with Two Icon Selections: Icon #2

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



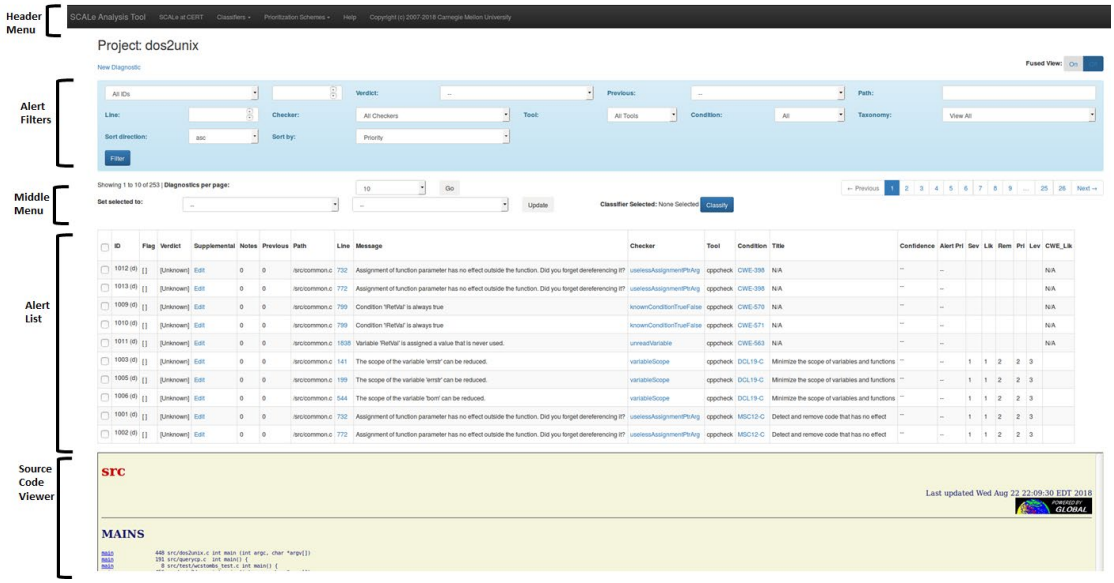
**015 You select that to create a database, and then you get the option to do two things-- to export that database, or to create the project. If you hover over that Create Project icon, you can see the text that says that's the one you want to choose. So you select that.

SCALe Homepage



**016 And then the project is created. Now, to get to the SCALe homepage that lists all active SCALe projects, you select the hyperlinked text in that top black menu bar that has the red arrow pointing to it right now, and then you see the website that's shown here that lists your active SCALe projects. Three are shown here.

Auditing Interface



**017 Now let's look at the SCALe project auditing interface. There are five sections of it. There's the top black header menu toolbar that allows you to edit and export projects; to create, edit and change classifier schemes and select them; and to create, edit and select prioritization schemes; and it provides information about SCALe.

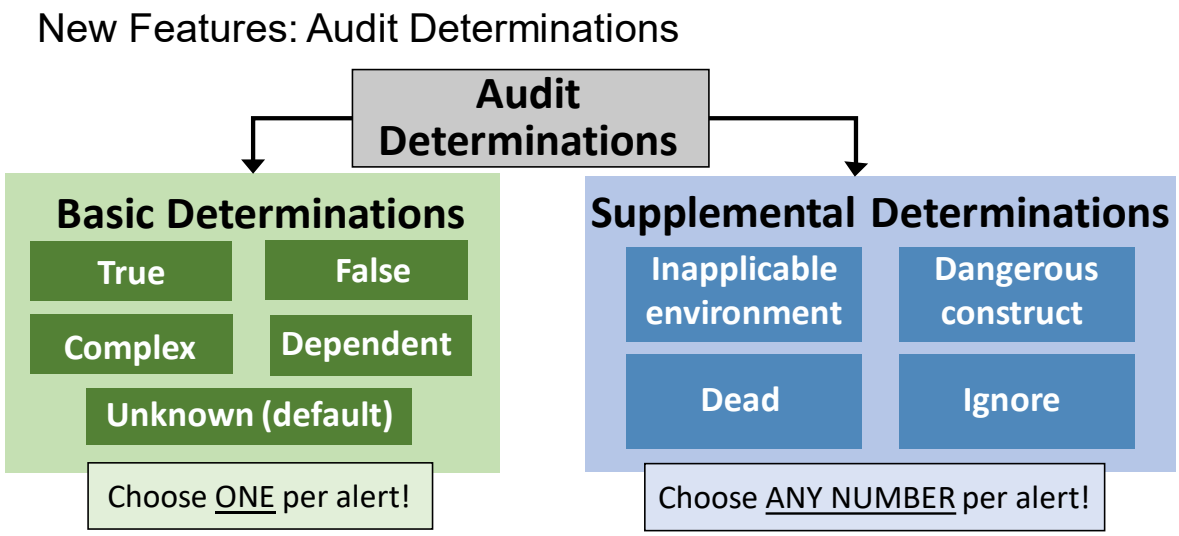
The light blue Alert Filters section allows you to select and filter information about different types of alerts-- for instance, to only look at CWEs or only look at CERT rules, or to look at just particular CWEs or CERT rules, as opposed to the default view, which shows all of them.

The middle menu allows you to determine how many alerts you look at per page, and to select which page of alerts you look at. And then that's

followed by the Alert List section, which has the tool warnings and associated information, one per line, except for some fused alerts, which are also shown one per line, and I'll talk about that later.

And then the bottom yellow section is the Source Code Viewer section. When you click on the hyperlinked line number from the Alert List, it will take you directly to the section of code, to that file and to that line number, and it'll highlight it in yellow in the Source Code Viewer.

New Features: Audit Determinations



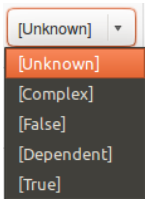
**018 So new features that we've added this year. The first new feature I'll talk about is audit determinations. We have changed the set of audit determination options to be a set of basic determinations, shown here-- true, false, complex,

dependent, and unknown-- one can be chosen per alert-- and a set of supplemental determinations where any number of those can be chosen per alert. Our explanation for why we have changed to that set of determinations is in a paper we published that-- a link to that paper is provided in the final slide of this slide set. It's from IEEE SecDev in 2016.

Determinations in GUI

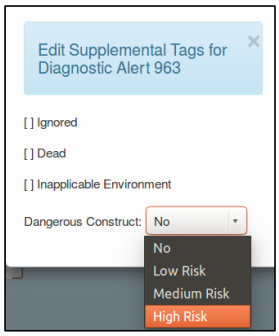
Determinations in GUI

Drop-down for primary verdict

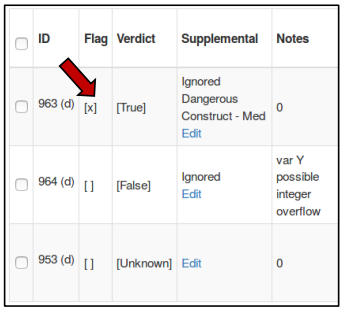


Supplemental determination popup:

- 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

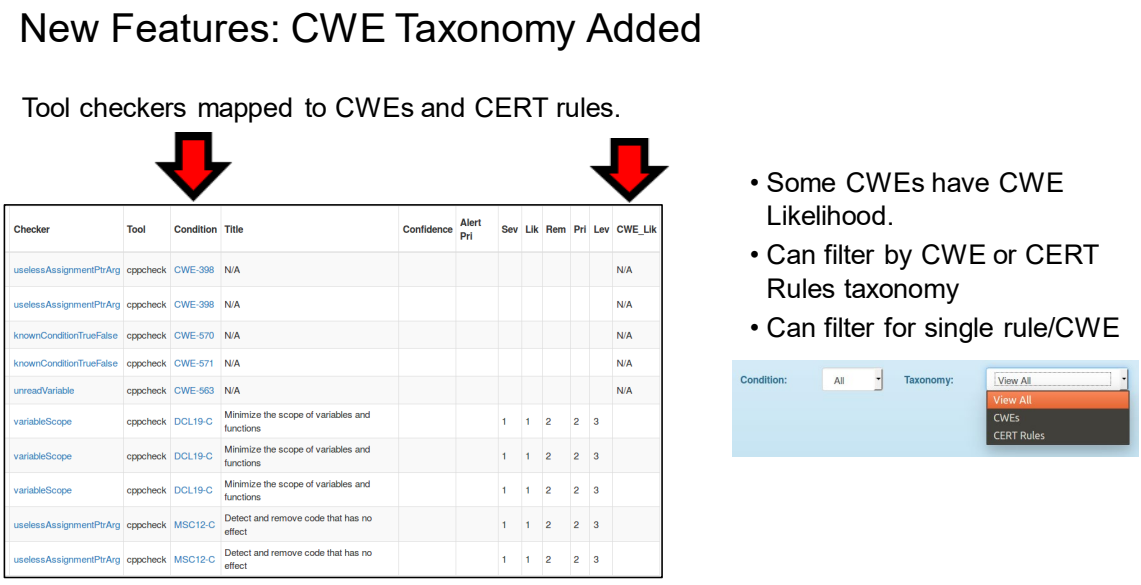
**019 In the GUI, the way you select these determinations is with a drop-down for the primary verdict, and the supplemental determination is selected with a pop-up. You select any number of those supplemental determinations, including-- for the dangerous construct, there is a drop-down, and you can select high risk, medium risk, or low risk.

Also a flag field can have an

organization-defined meaning. So it is a field that we have entered. If your organization wants to enter determinations that aren't supplied by the list I've talked about before, you can use the flag field, including in combination with any of the other determinations, and then define that combination of flag and any other determination to mean what you want-- any additional determination you want your organization to use.

And by the way, if you do that, please let us know, because our set of determinations is a work in progress and if more are needed, we want to add determinations as needed.

New Features: CWE Taxonomy Added



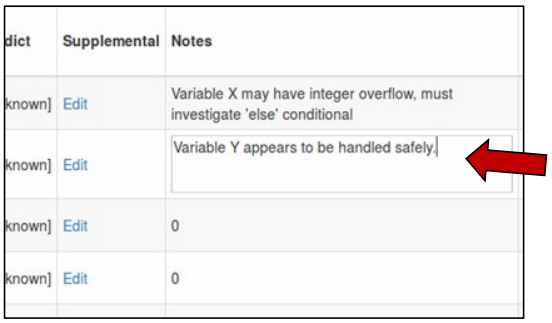
**020 The CWE taxonomy addition is a new feature to SCALe v2 and v3. We have mapped those tool checker

IDs I talked about before-- we've mapped them to both CWEs and CERT rules. So in this screenshot shown on the left, you can see that the alert list in SCALe has a newly labeled field called Condition. It used to be called Rule because we only had CERT rules. Now Condition can work for CWEs as well as CERT rules. And on the far right, there's a field CWE Likelihood. This is a metric which was scraped from the MITRE CWE, or Common Weakness Enumeration pages-- which exists-- it's a metric which exists and is filled in for some CWEs, and does not exist in other CWEs. You can see on the screenshot we've shown those particular CWEs didn't have that metric, so we just have N/A there, but for other CWEs, we have the metric that was in the page.

New Feature: Notes

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.



**021 Another new feature that we've added is the Notes field. The auditor can add notes as they audit determinations or after they've audited determinations, and the idea is for the text to help later auditors reviewing the same or similar conditions, and that later auditor might be the same auditor who comes back to the work later. So on the right you see a screenshot of the Notes field being filled in right now.

So I'm going to talk about a lot more features, but now's a time I can break to ask if anyone has questions so far.

Presenter: We had a general question come in from Kim to say: What sets us apart from-- so industry has a bunch of static analysis, but what sets SCALe apart from tools some of these vendors are providing as well?

Presenter: Great question. So SCALe itself doesn't do the static analysis of the code base. So there are many static analysis tools and SCALe imports the output of static analysis tools-- of multiple static analysis tools. It is a static analysis tool output aggregator, and there are similar aggregator tools like DHS SWAMP's publicly available tool, and there are proprietary aggregator tools as well. Now, SCALe is developed as this research prototype where we add the research features with the purpose being to enable gathering data that we can use for research, and if we're successful, if it turns out

that the features that we've added produce data that's helpful for automating classifier handling or making it more efficient in other ways, then a measure of our effectiveness is if other tool developers actually add the feature, if it's useful. Whereas commercial tools, they're different in that their intention is to sell.

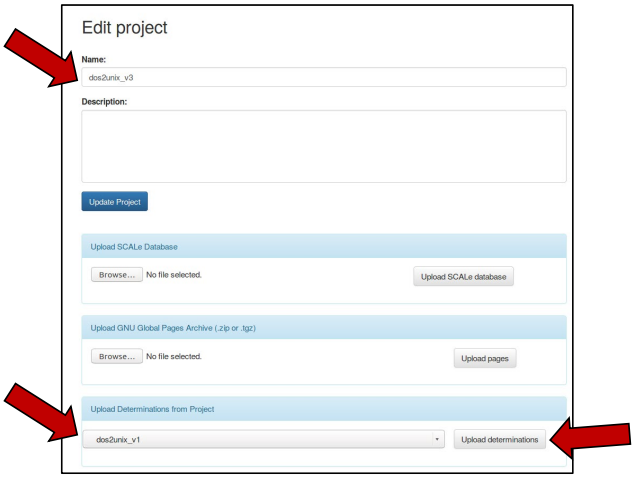
Presenter: Very good. Okay.

New Features: Cascade Determinations

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.



**022 Presenter: Let's talk about more new features. We've added cascade determinations. That uses the upload of determinations from the same static analysis tool on previous versions of the code. It uses diff for line matches, and when an alert and a line are matched, then it auto-cascades the determination--meaning it marks the same

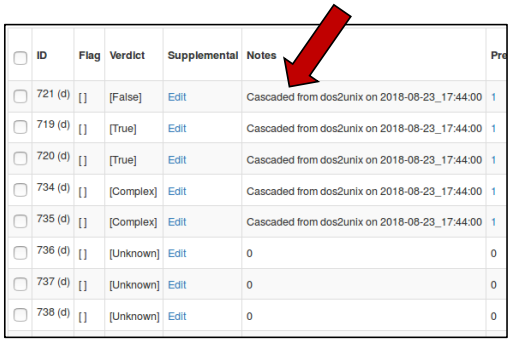
determination that was previously developed for the alert on the new version of the code. Now this feature should be used with caution because data flow, control flow, and type flow changes may cause a previously correct determination to change in the new version of the code.

After Cascaded Import

After Cascaded Import

After cascaded import

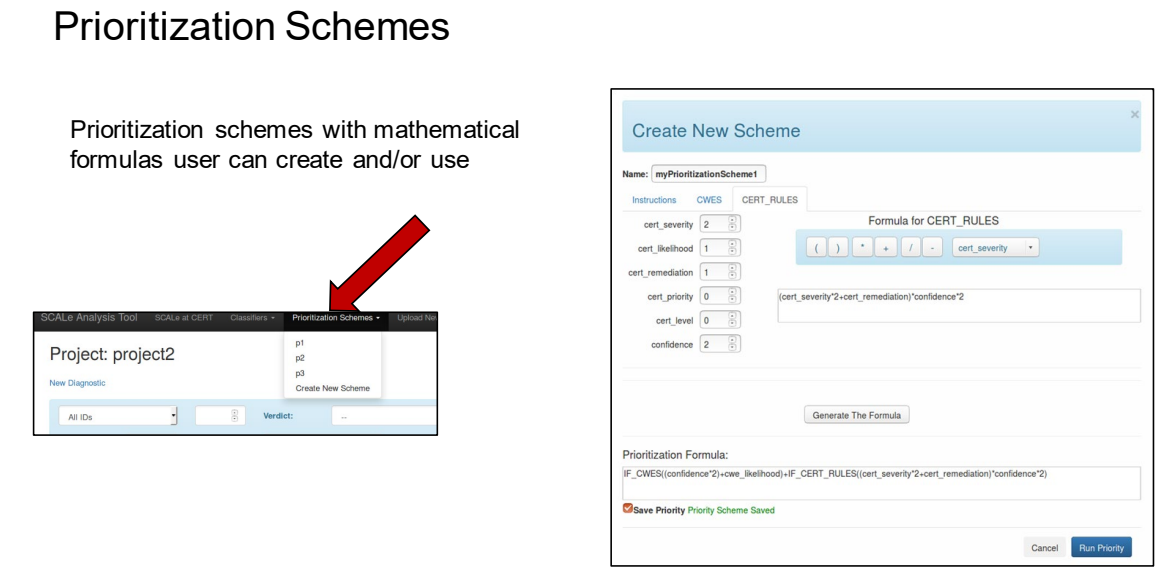
- Notes field show determination was cascaded
- Database records note about cascaded determination



	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

**023 After the cascades import, the Notes field shows that the determination was cascaded. The database records note includes that cascaded determination note, as you can see in the screenshot on the right.

Prioritization Schemes



**024 We've added prioritization schemes to SCALe v2 and v3, where the prioritization schemes have mathematical formulas that the user can create and/or use. The user selects the Prioritization Schemes option from the header, as you can see with the arrow pointing to it, and if they select Create New Scheme, then the pop-up, seen in the screenshot on the right-- the pop-up appears. Working from the top of that screenshot, you can see that you enter the prioritization scheme name, and then there's a row with tabs, where the leftmost tab shows the instructions, and then there's a CWE and a CERT Rules tab, where the user selects weights on the left-- weights for a number of fields that are prefilled for CERT rules or CWEs. And then towards the middle, the user creates a formula for either CERT rules or CWEs using the

mathematical characters shown-- the times, divide, add and subtract and parentheses-- and using the drop-down to get to those particular fields that can be used.

After the CWE and CERT Rules-- both have a formula-- then the button towards the button-- the button labeled Generate the Formula-- can be selected, and that creates the prioritization formula shown at the bottom of this slide. If you select this Save Priority checkbox, then it saves the prioritization scheme for use in future projects, and then you select the Run Priority button on the bottom right to create prioritization values for your set of alerts.

Presenter: So a quick question from Brad asking: Did you say which tools you have map checkers to CWEs for?

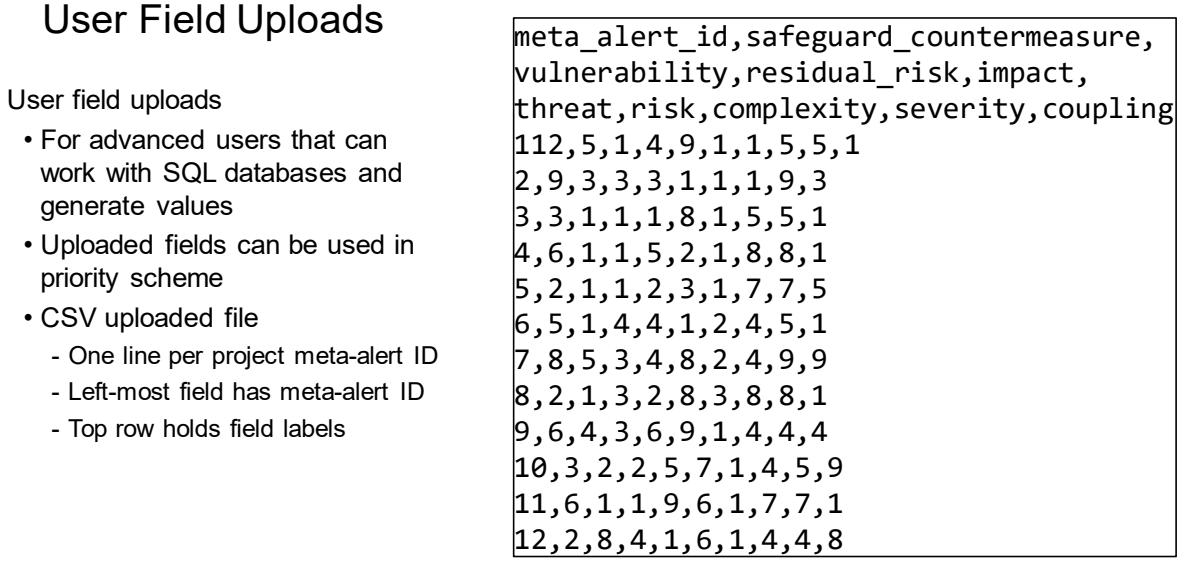
Presenter: I didn't. Let's see. So we are using mappings by tool vendors, actually. So for CERT rules, we're using combination of mappings to checkers that CERT engineers have made, and mappings to checkers that tool vendors have made. Many of them are available on the CERT Secure Coding standards pages.

Presenter: The wiki page? Okay.

Presenter: Yes. Yeah. For the CWEs, we went to tool websites for most cases, and downloaded their mappings and parsed them into a format that we can use in our tool,

and sometimes we work with vendors to get non-public mappings as well.

User Field Uploads

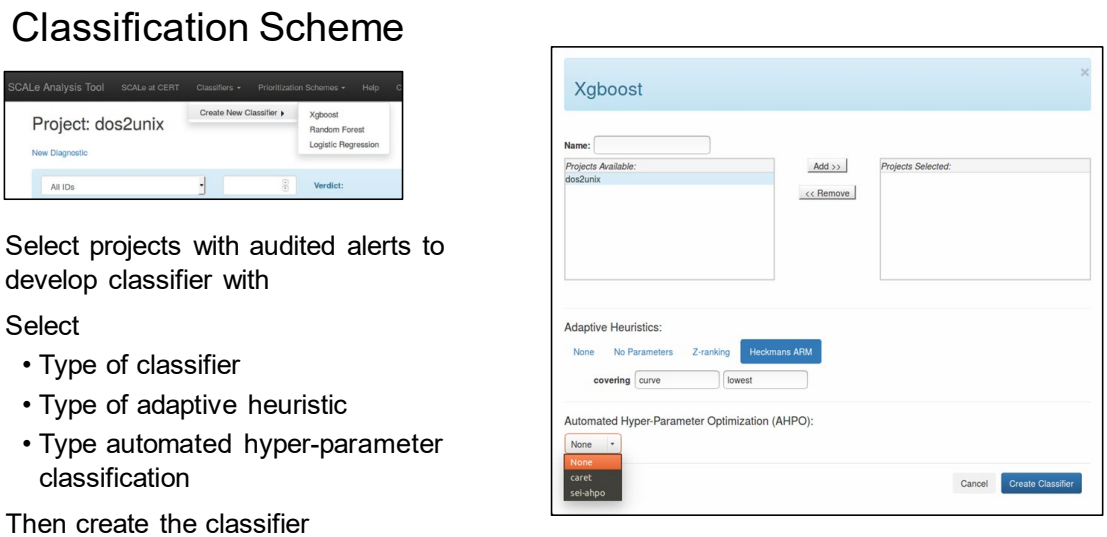


**025 Another new feature is user field uploads, and this feature is currently for advanced users that can work with SQLite databases and generate values. The idea is that these uploaded fields can be used in the priority scheme that I showed a couple slides ago. The user uploads a common-separated value file, or CSV formatted file, with one line per project-- actually, on the right you see an example of such a file, where the requirements include there's one line per project, meta-alert ID. Meta-alerts have a more complicated definition, but right now I'll just say that all alerts that share a file path, line number, and condition-- like a CWE or a CERT rule-- those are part of a meta-alert-- they share a meta-

alert ID. So one line per project, meta-alert ID. The leftmost field has the meta-alert ID, and the top row holds the field labels.

So looking at the example on the right, the top row, the meta-alert ID is the first label. The second label is Safeguard Countermeasure. So after this file has been uploaded, after these user fields have been uploaded with this file, when the user creates a new priority scheme, they will automatically see as options for new fields that can be in their formula. They'll see all of these new fields.

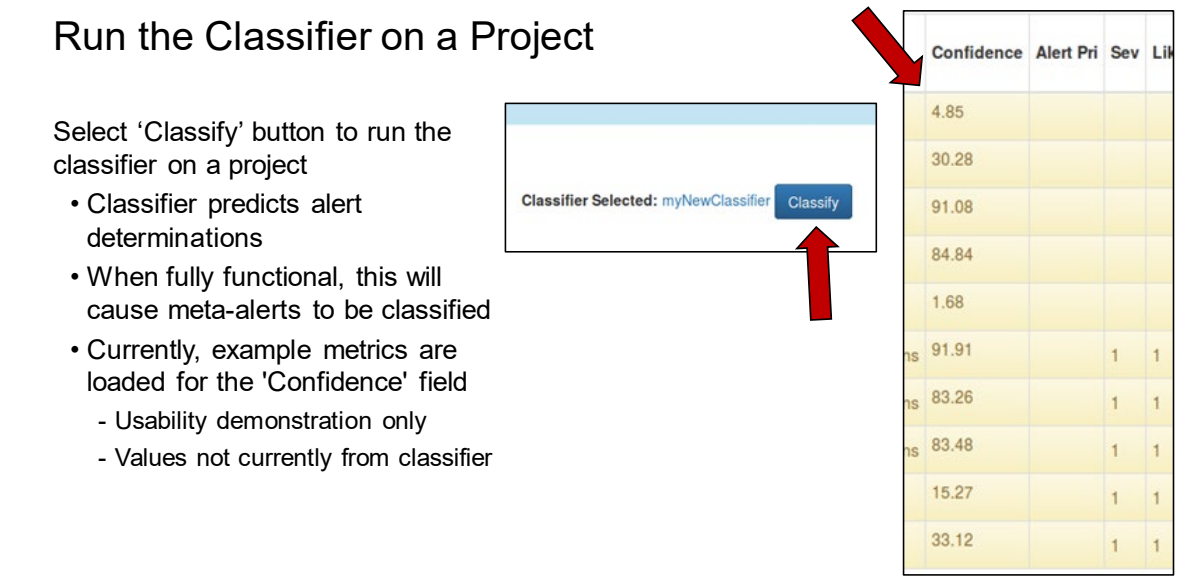
Classification Scheme



**026 We've also added a classification scheme option, where the user can select projects with audited alerts to develop the classifier with. Classifiers are developed using existing audit

archives that have labeled data that's, say, true or false for each alert. The user selects creation of a new classifier with a drop-down menu shown on the top left, and then with a pop-up-- an example is shown on the right for Xgboost-- they select the type of classifier, the type of adaptive heuristic, and the type of automated hyper-parameter classification, and then create the classifier with that button on the bottom right.

Run the Classifier on a Project



**027 To run the classifier on the project, the user should select the Classify button, as shown in the figure from the screenshot in the middle. The classifier predicts alert determinations, and when it's fully functional, this will cause meta-alerts to be classified. However, currently example metrics are loaded for the

Confidence field. So currently this is for usability demonstration only, and the values are not currently from a classifier. So we would welcome feedback on the usability from people that test this, and later in this presentation I'll talk about our plans for integrating classification.

This results in the figure shown on the right. Running the classifier results in the figure shown on the right, where the Confidence column from that alert list-- the Confidence column gets filled in, including with the example metrics.

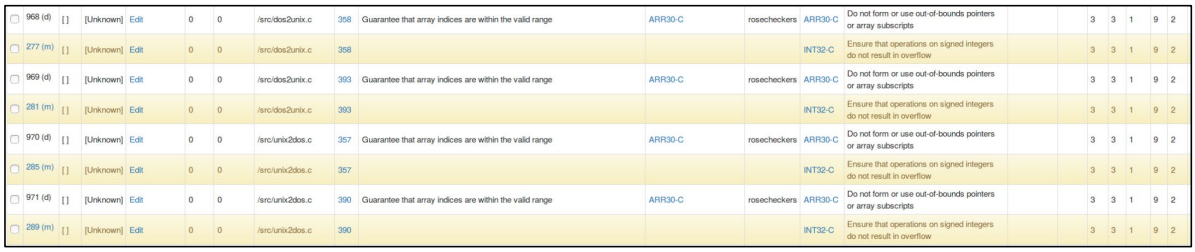
Alert Fusion

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

**028 Now let's talk more about alert fusion, which I alluded to before with the meta-alerts. So alert fusion is done by SCALe versions 2 and 3 for file path line and condition for alerts that are the same. The

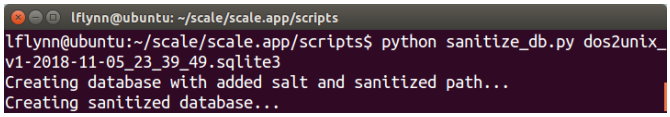
purpose is to reduce auditor effort. Multiple tools may indicate the same flaw. Auditors need to make the determination only one time. Even if Tool X, Y and Z all indicated that on Line 99 there was an INT31-C flaw, the auditor can one time make a determination that that was true. The auditor sees the messages and gains insight about the flaw from all the tools at once. So that also helps them to be efficient when they're auditing the alert.

The screenshot here shows fused alerts that are yellow, and unfused alerts that are white. The fused alerts are not expanded here because in this example they are proprietary tools, and due to terms of use agreements we're not allowed to show the checker IDs, which would show. But to see all the warnings and when making the determination, the auditor simply clicks on the yellow row and then they see the warnings from all of the tools all together.

New Feature: Archive Sanitizer

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.

**029 Another new feature is our Archive Sanitizer. We've added a data sanitizer Python script, which anonymizes sensitive fields and possibly sensitive fields using a SHA-256 hash with a salt. This enables analysis of features that are correlated to create an alert confidence.

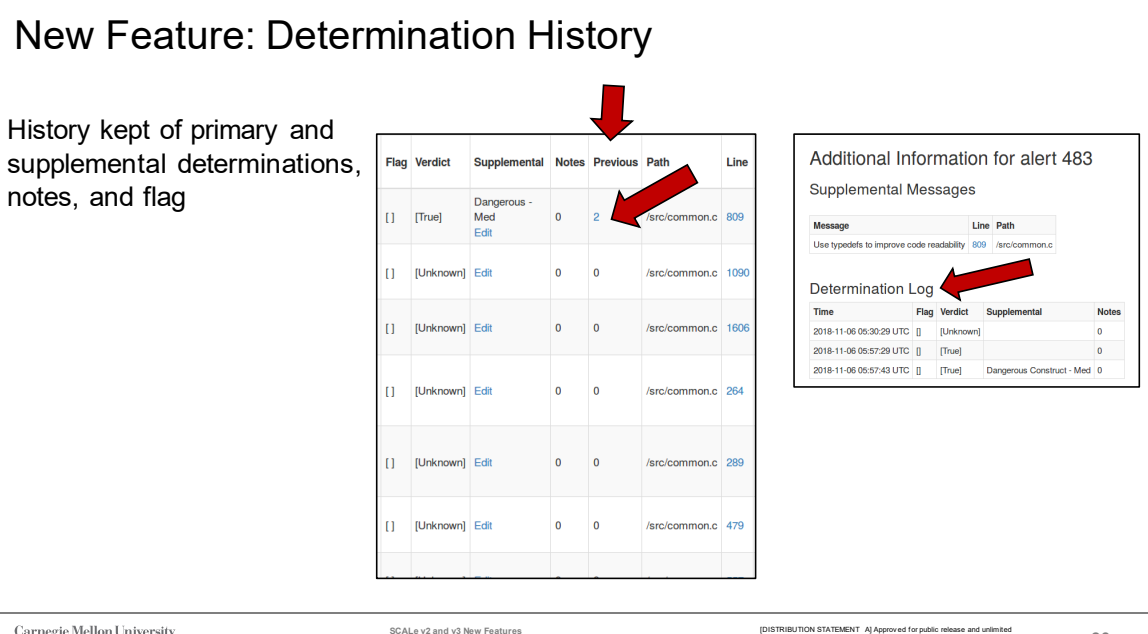
So the users run this Python script on the exported project database, as is shown in the screenshot on the top right. So in this bash shell terminal, you can see that from the top directory where SCALe is kept, the Python script is run and it creates a new database that stores the salt and a new database that doesn't have the salt, that just has the sanitized data. The fully sanitized database is the one that one would be shared with other organizations. An organization

should keep the salt private for increased security of their sanitized data.

The sanitizing script anonymizes or discards fields, including the diagnostic message, which it discards. The rest of the fields listed here it sanitizes by concatenating the salt and then running the SHA-256 hash on file paths, function names, class names, namespace or package, and project file names.

Users should be cautious. The GitHub sanitizer script is not fully updated for the SCALe v2 database, so don't count on it to sanitize your database. That needs to be updated.

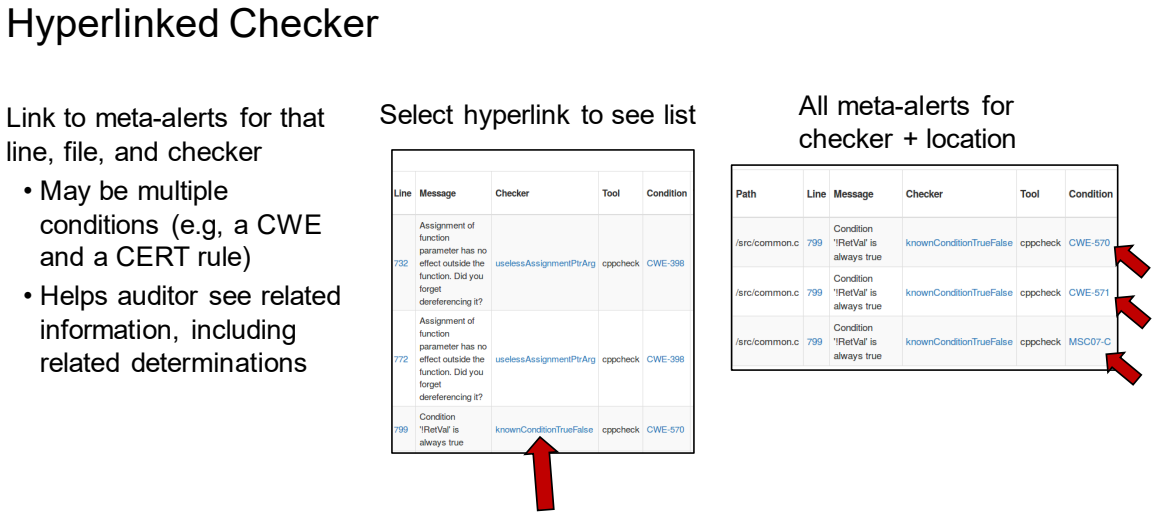
New Feature: Determination History



**030 Another new feature includes determination history. History is kept of primary and supplemental determinations now, of notes, and of

the flag value. As shown in the middle screenshot, from the list of alerts, there's a new column with the label Previous. That counts the number of previous audit determinations that have been made for that alert or meta-alert. In this case, what's pointed to is 2. If that number is anything other than zero, it's hyperlinked, and if the user selects that hyperlink, they can see the entire determination history. So we intend to use this history for increasing the accuracy of our classifiers.

Hyperlinked Checker



**031 We've also added a hyperlinked checker, which links to meta-alerts for that line, file and checker. So this is kind of similar to what we talked about before, but in this case, rather than showing all checker IDs for a particular condition

on a line and file path, instead this is for a checker ID. So there may be multiple conditions that are mapped to that checker ID-- for instance, one or more CWEs and one or more CERT rules. When the auditor selects this hyperlinked value, they can see a list of all meta-alerts for the checker and location, which helps the auditor to see related information, including any previously made determinations and notes for that location.

Demo

**033 Now I'll switch to a demonstration.

**DEMOIN

So here you can see the SCALe web interface using a Firefox browser. We're at the SCALe homepage, and you can see that there are two existing active SCALe projects. Let's

create a new project by selecting this button. We'll call this project DOS2UNIX. You can see I've practiced this a number of times. And then we'll create the project.

We need to upload source code. We'll upload a tarball for the source code, and let's enter Rosecheckers' static analysis tool output for it, and let's enter cppcheck static analysis output. There we go. Now we need to select the icon to create the database, and now we've got the option to create the project from the database, and let's select that.

So here is our auditing interface for the project that we just created. I'm going to scroll down a little bit so you can see there's a list of alerts here. Right now we've set the default diagnostics per page as 10. Let's go to the line number indicated here, line number and file path in the source code area. My laptop is a little small, so we'll scroll down. There we go. You can see it's highlighted and gone to that map file, that line number.

Now let us do some things like create a classifier. So let's create a new classifier-- you can see that there are already a couple of classifiers. Let's create an Xgboost one. We'll call it C3, and we'll use some labeled data from two of the existing projects, where auditors have already made determinations about true or false, and for adaptive heuristics we'll choose one of the options, and we'll choose to use our studio's caret for

automated hyper-parameter optimization, and now we'll create the classifier. So again, remember this right now uses example data. This isn't connected to a classifier in the back end. It will be in a future version of SCALe.

So now I want to point out that the classifier selected now shows this C3, but the Confidence field, which the classifier will fill, that isn't yet filled in, and that's because we haven't yet classified. Let's select the Classify button and run the classifier. Now example confidence fields are filled in.

Next we're going to try to fill in this Alert Priority field. First we have to select a prioritization scheme. Let's create a new scheme. You can see that there are a few that have already been created. We'll call it P4. You can see that the default is on the Instructions tab, and then there's a CWE tab and a CERT Rule tab. I previously have uploaded that file of new fields that I showed you on a previous slide, so that is why you see the additional fields down on the bottom of both the CWE and CERT Rule tabs, with the top one being Safety Countermeasure. Normally those wouldn't show, but since I uploaded the user new fields file, those show.

So let's select a formula. So for CERT Rules, let's say Severity 1 as a weight. Remediation, we'll set that to 1. And let's use Confidence from our classifier-- let's weight that as 2--

and let's wait the Safety Countermeasure as 1. Now let's create our formula. Here we will-- let's do a simple formula where we'll just multiply everything by each other. So CERT severity-- you see we have the drop-down that shows all the fields that we can use-- CERT Remediation-- and again, we'll just use simple times, and you can see that the weights were entered in. And Confidence here, the weight that we had entered was 2, and then lastly, let's multiply it by Safety Countermeasure, which automatically became part of the drop-down here.

So we now have a formula for CERT rules. Let's enter a formula for CWEs. You can see that the Safety Countermeasure value remains the same, and let's enter-- let's just use Confidence times two, and again we'll do kind of a simple formula, Confidence times the Safety Countermeasure.

Now let's scroll down and let's generate the combined formula for the prioritization heuristic. You can see that it just combines the CWE formula with the if-CWEs and the CERT rule formula part here.

And let's save the priority-- have we named it? Let me make sure we did. Yes. Let's save the priority and let's run it by selecting that button. And here-- oh, a live demo. What happened with running the priority? I'm not sure what happened, but you know what I'm going to do? I'm going to run one of the other

schemes that I ran before. I'm not going to delete it. Generate the formula, and run it. There we go.

So this is what it should look like. And now let's look at-- note, the alert priorities are filled in. I'm going to scroll down just that way you can see more values filled in. But currently the sorting is being done by the default, which was CERT Priority. Let's change that prioritization to use Alert Priority-- that's this new value using the scheme we just created-- and let's just change the sort direction to be descending, and we'll click this button, and now you can see that-- let me scroll down so you can see more of the alerts. You can see that the alert priorities that are highest are at the top.

And I can do one other thing. We've added sorting by confidence as well, so let's just sort by that without using the alert priority, and now the sorting is being done by confidence.

So now let's switch back to the presentation.

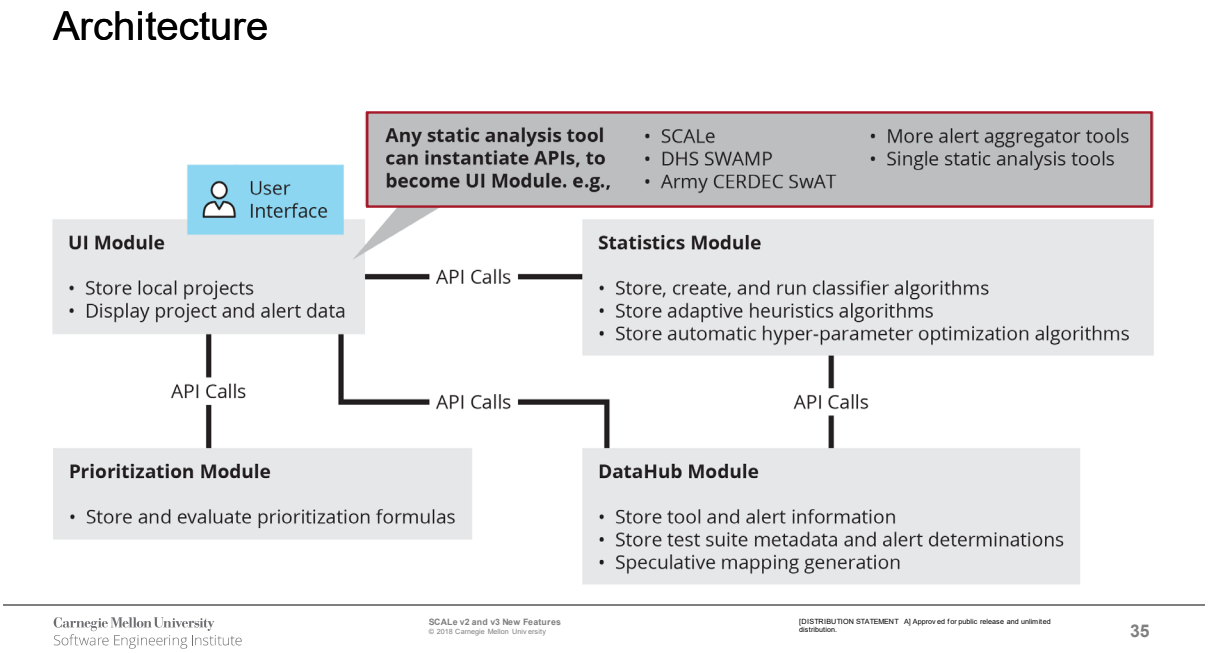
Presenter: Can we handle a question while you get that set up?

Presenter: Absolutely.

**DEMOOUT

Presenter: So a question asking: What are the different programming languages that are supported?

Architecture



**035 Presenter: So C, C++, Java and Perl are the four programming languages that currently we support. In the near future-- meaning within the next calendar year-- we expect to add Ruby, JavaScript, and Python.

Presenter: Then Greg asks: How large of a project have you been able to use SCALe with? For example, 20 million lines of code? A few hundred K?

Presenter: Yeah. I am not totally sure. I know that we've used it with projects that are at least a million lines of code. I don't know beyond that though.

Presenter: That's something maybe we can follow up with Greg about after the fact, get an answer.

Presenter: Most recently I can say

that we've started to work with output from running tools on the Juliet test suite within SCALe, and the many lines of warnings from the tools for this very big test suite-- we've been able to handle that. But it does take-- right now it takes about a half an hour to upload total output for that. So that is the biggest project that I personally have created projects with.

Presenter: And one more quick one, and maybe you're getting to this before we end, I'm sure, but how can a developer or an organization become a collaborator? How can they get involved with v3, and how do they get their hands on it? I know you said later in September when it's on GitHub, but how does one get involved?

Presenter: Oh, I would love it if-- anybody interested in collaborating should just contact me, or I think you'll provide an email address for--

Presenter: Yeah, we send a follow-up email tomorrow with an archive of the event and maybe how to get in touch with you.

Presenter: Okay, wonderful. Yes, I'd love to have more folks test using SCALe v3 to give us data, sanitized data output, from their use of SCALe, or other static analysis tools. We really need labeled data, meaning data with alert information and the true-false or whatever else

determination made manually by auditors, and sanitized data is just great.

Also, as I've shown on this slide, we're developing an architecture in this year's and next year's research that includes static analysis tools as one server, the UI module on the left, and the idea of this architecture is to enable static analysis tools such as SCALe or DHS SWAMP or other aggregators or other single static analysis tools-- enable those static analysis tools to easily start to use classifiers and advanced prioritization with low cost, because they'd be able to use capabilities that we've coded in these other servers they can use with API calls. So if possible collaborators would be interested in implementing API calls or helping us to improve our initial API definition, that would be very welcome as well.

Presenter: Great.

Presenter: So I want to talk just a little bit about this architecture, a little bit more about it. You can see that there are four servers in this architecture we've developed. The Statistics module on the top right stores, creates, and runs the classifier algorithms. The Data Hub module stores data. The Prioritization module stores and evaluates prioritization formulas, and the top left UI module is where the static analysis tools such as SCALe, DHS SWAMP, Army CERDEC SwAT, and other aggregator and single static analysis tools would fit, and the idea is that they just need to

instantiate the API calls and respond to the defined API calls in the architecture as the UI module to be able to benefit from those things.

Architecture Development

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)

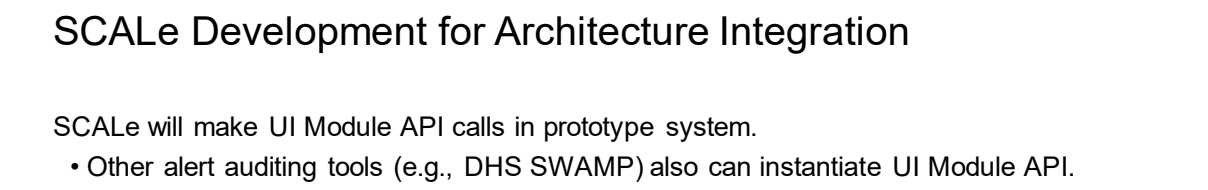
**036 How much time do I have?

Presenter: We got about nine minutes left.

Presenter: Okay, great. So our architecture development has used the representational state transfer, or REST, this architectural style, that defines a set of constraints and properties based on HTTP with web services that provide interoperability between systems that are client-server. So we're developing, or we have developed, this RESTful API and we're continuing to develop and improve that.

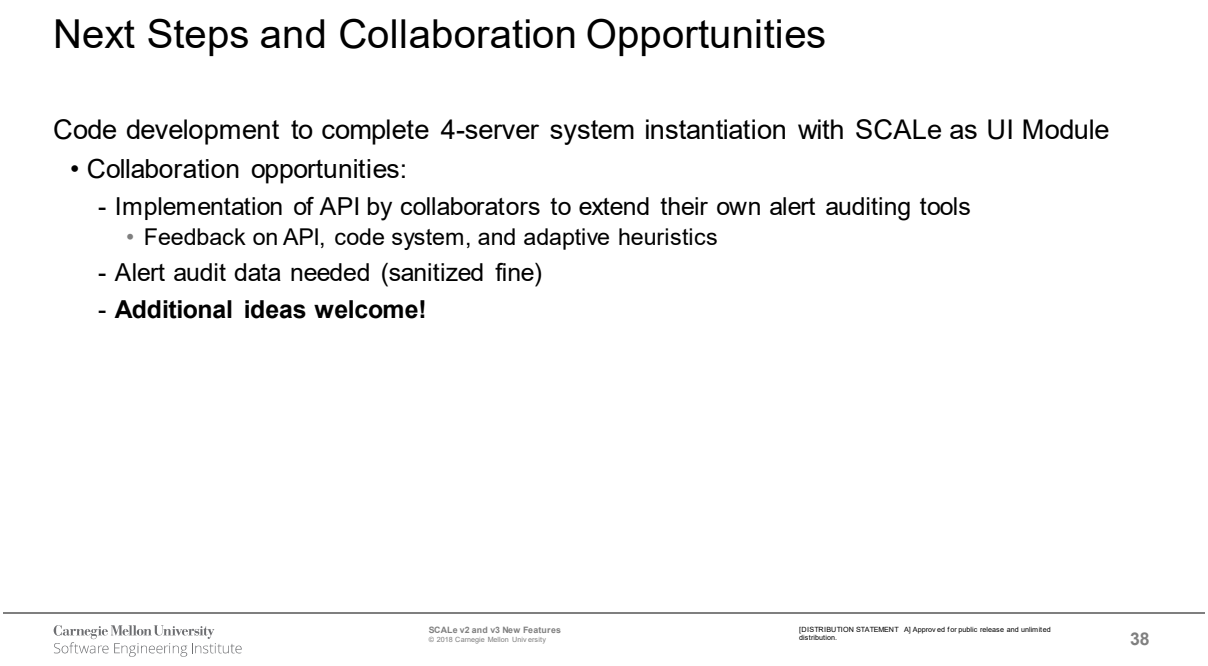
Using this Swagger/OpenAPI open-source development toolset, the main reason we chose to use that is because it is already so widely used. There are about 10 thousand downloads a day, and so we use it to develop the APIs and to auto-generate code for servers, stubs, and clients, and then our developers are filling in the functional code for those three servers. We're filling in the code in those stub functions, and the goal is to make all the code open source, make the code for those three servers open source, so organizations or single developers can simply locally, on their own network, host those servers and then just connect their tool with it, with those servers, using the APIs.

SCALe Development for Architecture Integration



**037 I just talked about what's on this slide.

Next Steps and Collaboration Opportunities



**038 So this slide talks about some of the specific collaboration opportunities, and I just want to make sure that when I answered before that I hit each of these points. So let me see. So we're currently completing that fourth server system instantiation with SCALe as the UI module in our example code, which we hope to publicly release that version, which will be SCALe v4 or beyond, and collaboration opportunities include implementation of that API, feedback on the API, alert audit data, and additional ideas are welcome.

References

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)

**039 This slide simply lists references to our previous publications, including the second one, which is a link to the GitHub SCALe v2 publication. So the slides will be available from the SEI site about this webcast.

Presenter: Yes, and that will be included in the email that goes out tomorrow with the archive location, where to get the slides, and how to get in touch with you if they're interested in collaborating. So we can do that. So a couple questions from Brian. We got about five minutes left. One general question, then a couple specific ones, but I'm interested in you as a researcher to answer the general one. It just asks: In which ways does the SEI aid and abet national security priorities with CERT and government working together for good?

Presenter: Let me start from the end of that. So working together for good-- I think securing code is a good thing. So trying to more efficiently find true flaws helps us to secure code, and in fact to secure code at lower cost. So that's an example of good which I guess can apply to both government and non-government use. We have DoD collaborators on this project, so these organizations-- which I'm not allowed to name-- but they have very, very generously provided developer effort and manager effort for their organizations to use various versions of SCALe, including 2 and 3, and some versions between those. They've tested using those SCALe versions, provided us feedback, and, very importantly, they've provided us with sanitized data that we've used to test our classification heuristics and improve them, and they're continuing to help us with that.

Related to that architecture, there are DoD organizations that have committed to implementing APIs with their aggregator tools, and there are additional DoD organizations that we're hoping will be able to do the same. Specifically DHS SWAMP-- we're talking with them, and we might even bring in some commercial organizations which are using DHS SWAMP and they might help us with developer effort to implement the APIs. We are really hopeful that we'll be able to get it tested, the APIs, including DHS SWAMP.

Presenter: I'll add in, Brian, that

the SEI is a federally funded research and development center, so one area of work was secure coding, but CERT has expertise in resilient management, digital forensics, software automation-- so a number of areas that we have expertise in that we can loan out to or work with the government "for the good", as you put it. Next question from Brian was: Does Data Hub API support display project? Does that make sense? Does Data Hub API support display project?

Presenter: Let's see. No, because right now the user interface module is envisioned as that is where the display project-- displaying the project would happen. So directly at the Data Hub, we don't have API calls that display the project, but we do have API calls from the UI module that get the data that would be displayed there.

Presenter: Okay, and we got about a minute left, so we'll wrap up with one more, one last one: Why should a client with server in WLAN expect client loyalty from architecture development? And I can repeat that: Why should a client with server in WLAN expect client loyalty from architecture development?

Presenter: Let's see. Oh, so if I understand the question correctly-- I think the questioner is asking, for a proprietary tool, vendor that has clients, what would their motivation be for interacting with the API, and the way I think of it is that hopefully

they get work from us for free-- our code and algorithms that would enable them to start using classifiers, which a lot of research shows that classifiers work pretty well often. But many tools and many organizations, especially DoD organizations, don't yet take advantage of using classifiers because they don't have enough developers, they don't have enough labeled, audited data that's high quality to be able to create accurate classifiers. So the idea would be that the vendors would be able to-- with hopefully little cost from them, they'd be able to use what we're giving away for free.

Presenter: Very good. We got two o'clock here on the East Coast. Lori, great presentation. Thank you very much for spending the hour with us and talking about SCALe. Like I mentioned, we will send out a follow-up email tomorrow with an archive of the video as well as where to get the slides. Again, if you don't mind, please fill out the survey upon exiting today's event. That link for the survey will also be in the email, so we'd love to get your feedback on today's event. Thanks everyone for your time today and have a great day. Thank you.

Notices

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