Rapid Construction of Accurate Automatic Alert Handling System

Presenters: Dr. Lori Flynn (PI) and Ebonie McNeil

Other Team Members: Matt Sisk, Derek Leung, and David Svoboda
Rapid Construction of Accurate Automatic Alert Handling System

Overview
Rapid Construction of Accurate Automatic Alert Handling System

Overview
FY18-19 Artifacts
Status of SCAIFE
Impacts Time Frame
Overview

Problem: too many alerts
Solution: automate handling

Static analysis (SA): analysis of code without executing it
- Automated SA is widely used.
- It is a normal part of testing by DoD and commercial organizations.

In this presentation, alert represents alert, meta-alert, or alertCondition as defined in our previous publications.
FY16-19 Static Analysis Alert Classification Research

Goal: Enable practical automated alert classifier use so all alerts can be addressed.

FY16
- Issue addressed: classifier accuracy
- Novel approach: multiple static analysis tools as features
- Result: increased accuracy

FY17
- Issue addressed: too little labeled data for accurate classifiers for some conditions (e.g., CWEs, coding rules)
- Novel approach: use test suites to automate the production of labeled (True/False) alert archives* for many conditions
- Result: high precision for more conditions

FY18-19
- Issue addressed: little use of automated alert classifier technology (requires $$, data, experts)
- Novel approach: develop extensible architecture with novel test-suite data method
- Result: enabled wider use of classifiers (less $$, data, experts) with extensible architecture, API, software to instantiate architecture, and adaptive heuristic research

* By the end of FY18, ~38K new labeled (T/F) alerts from eight SA tools on the Juliet test suite (vs. ~7K from CERT audit archives over 10 years)
Rapid Construction of Accurate Automatic Alert Handling System

FY18-19 Artifacts
SEI SCALe Framework: Background

Static Analysis Alert Auditing Framework
Developed by the SEI for ~10 years.
- GUI front end to examine alerts and associated code
- Alert adjudications (true, false) stored in database

Use for Research Projects
- Enhance with features for research.
- Collaborators use it on their codebases.
- Researchers analyze audit data.
SCAIFE Definitions

SCAIFE is a modular architecture that enables static analysis alert classification plus advanced prioritization.

• The **SCAIFE API** defines interfaces between the modular parts.
• **SCAIFE systems** are software systems that instantiate the API.
• Our SCAIFE system releases include a SCALe module plus much more.

SCAIFE = Source Code Analysis Integrated Framework Environment
FY18 Software Artifacts

Developed code to develop and test classifiers.
Code includes novel functionality:
- enables cross-taxonomy test suite classifiers (using precise mappings)
- enables “speculative mappings” for tools (e.g., GCC)

SCALe v2.1.3.0 released to collaborators
(December 2017-February 2018)
- New features for prioritization and classification
  - Fused alerts, CWEs, new determinations, etc. for collaborators to generate data

Started API definition (swagger) and code development

SCALe v2 GitHub release (August 2018)

SCALe v3.0.0.0 released to collaborators
- New features for advanced prioritization schemes, user-uploaded fields, adjudication history, and classifier selection

First Public SCALe Release (2.1.4)
# FY18: Non-Code Publications

<table>
<thead>
<tr>
<th>Publication Goal</th>
<th>Publications and Papers</th>
</tr>
</thead>
</table>
| Help developers and analysts provide feedback on our API and use new SCALe features. | • SEI special report: *Integration of Automated Static Analysis Alert Classification and Prioritization with Auditing Tools* (August 2018)  
• SEI blog post: *SCALe: A Tool for Managing Output from Static Code Analyzers* (September 2018) |
| Explain classifier development research methods and results. | • 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* (April 2018)  
• SEI podcast (video): *Static Analysis Alert Classification with Test Suites* (September 2018) |
| Enable developers and analysts to better understand tool coverage for code flaws using our inter-taxonomy precise mapping method. | • CERT manifest for Juliet (created to test CWEs) to test CERT rule coverage with tens of thousands of tests (previously under 100)  
• Per-rule precise CWE mapping in two new CERT C Standard sections [1] [2] |
Juliet Test Suite Classifiers: Initial Results (Hold-Out Data)

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Accuracy</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>rf</td>
<td>0.938</td>
<td>0.893</td>
<td>0.875</td>
</tr>
<tr>
<td>lightgbm</td>
<td>0.942</td>
<td>0.902</td>
<td>0.882</td>
</tr>
<tr>
<td>xgboost</td>
<td>0.932</td>
<td>0.941</td>
<td>0.798</td>
</tr>
<tr>
<td>lasso</td>
<td>0.925</td>
<td>0.886</td>
<td>0.831</td>
</tr>
</tbody>
</table>

**Accuracy**

\[
\text{Accuracy} = \frac{\Sigma \text{True positive} + \Sigma \text{True negative}}{\Sigma \text{Total population}}
\]

**Precision**

\[
\text{Precision} = \frac{\Sigma \text{True positive}}{\Sigma \text{Predicted condition true}}
\]

**Recall (sensitivity)**

\[
\text{Recall} = \frac{\Sigma \text{True positive}}{\Sigma \text{(Condition true)}}
\]

**False positive rate**

\[
\text{False positive rate} = \frac{\Sigma \text{False positive}}{\Sigma \text{(Condition false)}}
\]
FY19 Releases: Software and YAML API Definitions

SCAle v4 release:
- New features required for SCALIE integration, bugfixes, and automated tests (April 2019)

SCAle DevOps
improvements to improve transitionability of research (January–February 2019, mostly)

SEI GitHub release: SCALIE beta version (0.0.2, 0.0.4) API in a YAML specification (June 2019, Aug 2019)

Releases to four DoD collaborators (August, September 2019)
- SCALIE System VM Beta v1 and v2
- SCALIE versions r.4.2.0.0.A and r.5.0.0.0.A

Coming Soon
SCALIE System VM v1 release
FY19: Select Non-Code Publications –1

Publications to Explain Research and Development Methods and Results

- SEI blog post: An Application Programming Interface for Classifying and Prioritizing Static Analysis Alerts by Lori Flynn, and Ebonie McNeil (July 2019)

- SEI whitepaper: SCAIFE API Definition Beta Version 0.0.2 for Developers by Lori Flynn and Ebonie McNeil (June 2019)

- SEI technical report: Integration of Automated Static Analysis Alert Classification and Prioritization with Auditing Tools: Special Focus on SCALe by Lori Flynn, Ebonie McNeil, David Svoboda, Derek Leung, Zach Kurtz, and Jiyeon Lee (May 2019)

- SEI blog post: SCALe v3: Automated Classification and Advanced Prioritization of Static Analysis Alerts by Lori Flynn and Ebonie McNeil (December 2018)

FY19: Select Non-Code Publications –2

Publications to Demonstrate New Features of SCALe and SCAIFE


• [SEI Cyber Minute](#) by Ebonie McNeil (August 2019)

• SEI webinar: *How can I use new features in CERT’s SCALe tool to improve how my team audits static analysis alerts?* ([video](#) and [slides](#)) by Lori Flynn (November 2018)

• SwACon paper: *Introduction to Source Code Analysis Laboratory (SCALe)* (one-hour presentation, including demo at Software Assurance Conference [SwACon]) by Lori Flynn (November 2018)

Coming Soon

• Paper submissions to conferences (e.g., ICSE 2020) on classifier results and architecture model development
Source Code Analysis Integrated Framework Environment (SCAIFE)

Rapid Construction of Accurate Automatic Alert Handling System

Status of SCAIFE
SCAIFE Architecture Approach

For efficient development of a robust API to enable widespread classifier use, we need a system architecture that

- Integrates with existing static analysis tools and aggregators (including SCALe)
- Supports classification and adaptive heuristic functionality
- Demonstrates fast response time for average and worst-case scenarios
- Provides extensibility for future research in static analysis, classification, architecture, and SecDevOps

Swagger/OpenAPI Open-Source Development Toolset

- Quickly develops APIs following the OpenAPI standard
- Auto-generates code for servers and clients in many languages
- Test server and client controllers with Swagger UI
- Widely used (10,000 downloads/day)

- Big O analysis was useful.
- Design decisions required balancing goals and analyzing tradeoffs.
SCAIFE Architecture

Source Code Analysis Integrated Framework Environment (SCAIFE)

- Registration Module
  - Generates registration tokens
  - Provides authentication and basic authorization for other servers

- DataHub Module
  - Stores tool and alert information
  - Stores test suite meta-data and alert determinations
  - Generates speculative mappings

- Statistics Module
  - Creates, runs, and stores classifiers
  - Stores adaptive heuristic algorithms
  - Stores automated hyperparameter optimization algorithms

- Prioritization Module
  - Stores prioritization formulas and user-uploaded prioritization fields

- UI Module
  - Stores local projects
  - Displays project and alert data

Any static analysis tool can instantiate APIs to become a UI Module. For example:
- SEI SCaLe
- DHS SWAMP
- CCDC C5ISR SwAT
- Other aggregator tools
- Single static analysis tools
SCAIFE Alert Dataflow with SCALe Module

1. Create classifier
2. Run classifier on a project
3. Get back classifier results
Status of SCAIFE v1

• 87% of the Statistics module functionality is complete.
  - The Statistics module is coded with three classification types and three adaptive heuristic (AH) types.

• 91% of the Registration, DataHub, and Prioritization modules’ functionality is complete.
  - DataHub auto-adjudicates test suite data using SARD-style manifests.
  - Prioritization server stores prioritization schemes and restricts availability based on user organization ID, project ID, and scheme sharing type.
  - From UI (SCALe), users can register on SCAIFE, upload data to the SCAIFE DataHub, select a classifier and AH, and run classifier.

• SCAIFE passes automated integration tests, showing correct multi-server functionality.

• SCAIFE fields were added(modified to improve future integration as a result of reviewing multiple static analysis tool APIs.

• AHs require updates (e.g., new manual adjudications), resulting in new confidence values.
  - Various system dataflows are being considered to enable future AH use.
Rapid Construction of Accurate Automatic Alert Handling System

Impacts Time Frame

AI Engineering-Related Topics
• Robust Systems: V&V, Tools & Process, Secure Coding
• Data, Devices, and Computing: Scalability, Performance and Evaluation
## Project Impacts Time Frame

<table>
<thead>
<tr>
<th>NEAR</th>
<th>MID</th>
<th>FAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>The public can review/use the API.</td>
<td>More collaborators (DoD and non-DoD) to test SCAIFE with CI.</td>
<td>A wide variety of systems will do automated alert classification, using</td>
</tr>
</tbody>
</table>
| DoD collaborators can further test SCAIFE to  
  - provide data and feedback  
  - integrate their tools using the API | Design improvements for transition include  
  - classification precision  
  - latencies  
  - bandwidth/disk/memory use  
  - business continuity  
  - scalability | • SCAIFE System  
  • SCAIFE API |
| The FY20-21 research project incorporates continuous integration (CI) into architecture design. | Goal: Provide better software security, or less time and cost for the same security (DoD and non-DoD). | |