

Verifysoft
TECHNOLOGY

Willkommen zu den
8. Static Analysis Days

Mi 5. und Do 6. Mai 2021

SAD 2021
Static Analysis Days
Online 5.-6. Mai 2021

www.verifysoft.com

1

✓ Static Analysis Days

Verifysoft
TECHNOLOGY



Seit 2014 in Offenburg
2021 erstmals digital
2022 hoffentlich wieder live + ...







8. STATIC ANALYSIS DAYS Verifysoft
Code-Analyse für Safety und Security
5. und 6. Mai / online
SAD 2021
Static Analysis Day
JETZT ANMELDEN
www.verifysoft.com

www.verifysoft.com

2

2


Static Analysis Days


| | | |
|--|--|--|
|  <p>Paul Anderson VP of Engineering GrammaTech Inc., USA</p> |  <p>Jan-David Baltzer Support Engineer Verifysoft Technology GmbH</p> |  <p>John Blattner CEO and Founder Imagix Inc., USA</p> |
|  <p>Walter Capitani Director Technical Product GrammaTech Inc., USA</p> |  <p>Royd Lüttke Direktor Statische Analyse Verifysoft Technology GmbH</p> |  <p>Klaus Lambertz Geschäftsführer Verifysoft Techn. GmbH</p> |

www.verifysoft.com
3

3


Verifysoft Technology GmbH







Gründung 2003
Offenburg, Deutschland
Reseller und Support
 für Code Coverage, Statische Analysetools
2013 Übernahme der Testwell Tools *Testwell*
Distribution und Support komplementärer Tools




Seminare





www.verifysoft.com
4

4

Testwell CTC++

Code Coverage Analyser

Analysiert für Code Coverage Stufen (bis MC/DC und MCC)

Unterstützt alle Compiler und alle embedded Targets

Qualification-Kit verfügbar (DO-178C, ISO 26262, EN 61508, IEC 61508, ...)

Source file: C:\Projects\hcontrol\regulators.c
Instrumentation mode: multicondition
TER: 71 % (20/28) structural, 71 % (17/24) statement
To files: Previous | Next

| TER % - multicondition | TER % - statement | Calls | Line |
|------------------------|-----------------------|-------|------|
| 75 % - (6/8) | 83 % - (5/6) | 14 | 4 |
| 100 % (2/2) | 100 % (1/1) | 4 | 20 |
| 100 % (2/2) | 100 % (1/1) | 8 | 25 |
| 100 % (2/2) | 100 % (3/3) | 4 | 30 |
| 100 % (2/2) | 100 % (1/1) | 4 | 37 |
| 0 % - (0/2) | 0 % - (0/1) | 0 | 42 |
| 60 % - (6/10) | 55 % - (6/11) | 8 | 47 |
| 71 % - (20/28) | 71 % - (17/24) | | |

```

14      4 void lights(enum light_status goal)
15      5 {
16      6 if (goal == off)
17      7 {
18      8     printf("Light is switched off.\n");
19      9 }
20      6 else if (goal == on)
21      7 {
22      8     printf("Light is switched on.\n");
23      9 }
24      6 else if (goal == dimmed)
25      7 {
26      8     printf("Lights are dimmed.\n");
27      9 }
28      4 }
    
```

5

CODESonar®

Advanced Static Analysis: Quellcode und Binärcode

Aufdecken von Bugs

Überprüfung von Coding Standards

Softwaremetriken

Architekturvisualisierung

Taint Data Tracking

CODESONAR Search code in this analysis for

Home » perf_ovl » perf_ovl_analysis_3 » Warning 49.4074

< Prev (Warning 1 of 582) Next >

Amendable alignment at 2:build.c:217 No properties have been set | edit properties
Jump to warning location

Options

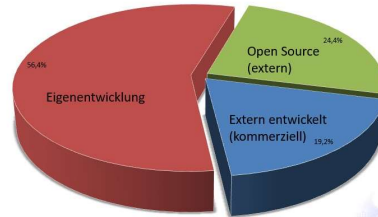
```

@Users\Luedke\Documents\perf_ovl\master\perf2-build.c
207 // previous one.
208 precedence--;
209 body.operators = owl_next(body.operators);
210 }
211 }
212
213 // Row fill in the automata according to the contents of each parsed rule.
214 rule_index = 0;
215 for (i = 0; rule_i != empty; i = owl_next(i), rule_index++) {
216     struct parsed_rule parsed_rule = parsed_rule_get(i);
217     struct rule_rule = grammar_rule(parsed_rule);
218
219     Amendable alignment
220     Pointed structure rule (about rule) is not well aligned (aligned at 80) 173 bytes out of 184 required bytes
221     automaton(40B), token_index(4B), token_examples_allocated_byte(4B), number_of_token_examples(
222     context_allocated_offset(8B), number_of_brackets(4B), beyond_tokens_allocated_offset(4B), number
223
224     // Store the rule index in our context object so we don't have to pass
225     // it around everywhere while we're building the rule's automata.
226     context.rule_index = rule_index;
227     context.next_rule_index = i;
228
229     struct parsed_body body = parsed_body_get(parsed_rule.body);
230     if (body.identifier.empty) {
231         // this is a simple rule with no choices. Create the automaton
232         // directly.
    
```

6

CODESentry™

Binary Software Composition Analysis



Quelle: VDC Research
2016

CODESENTRY customer.discover.grammatech.com

Dashboard: 5 Components, 1 Scan

090920P > 090920A > libcurl.so Scan Complete, Scan Depth: Average, File size: 120,27 MB

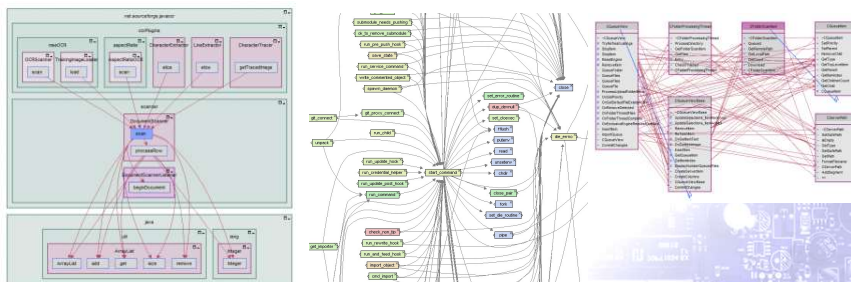
Bill of Materials Vulnerabilities Scan Tree

| File/Fail | Component | Version | Match | CVSS Distribution | Artifact |
|-----------|-----------|---------|--------|-------------------|------------|
| High | libjpeg | 1.4.2 | High | High, Medium, Low | libcurl.so |
| Medium | libzlib | 2.8.1 | Medium | High, Medium, Low | libcurl.so |
| Medium | libpng | 1.7.0 | Medium | High, Medium, Low | libcurl.so |
| Low | libssl | 1.1.3 | Low | High, Medium, Low | libcurl.so |
| Low | libnss | 1.7.2 | Low | High, Medium, Low | libcurl.so |

7

Imagix

Analyse von Kontrollfluss und Abhängigkeiten
Besseres Verständnis von Legacy- und Third-Party-Code
Aufdecken von Problemen in der Datennutzung und bei Nebenläufigkeiten
Reverse Engineering



8

✓ Seminare / Recorded Webinars / Videos



Testen von Embedded Software - Online
Mehrtägiges Seminar mit Dipl.-Ing. M. Heininger
11./12./16./19. Mai 2021 (deutsch)
Online
Hier Video ansehen



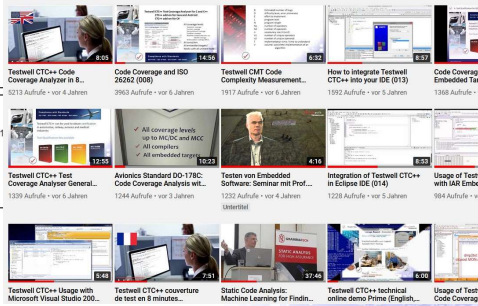
Kostenfreies Webinar: Imagix 4D and Refactoring is just a Bowl of Cherries
Dienstag, 8. Juni 2021, 14.00 Uhr MEZ (Berlin, Zürich, Wien) (englisch)



Testen von Embedded Software
2-tägiges Seminar mit Dipl.-Ing. M. Heininger
Mittwoch, 20. Oktober - Donnerstag, 21. Oktober 2021
Technologiepark, Offenburg
Hier Video ansehen

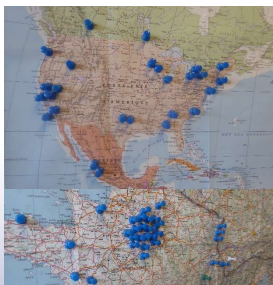
Inhouse-Schulungen
Verifysoft bietet folgende Inhouse-Schulungen an

Aktuelle Seminare:
www.verifysoft.com



www.youtube.com/c/Verifysoft

✓ Kunden



über 675
zufriedene Kunden
in 39 Ländern
auf allen Kontinenten
(Stand April 2021)





Agenda Tag 1



Mittwoch, 5. Mai

09:45 Begrüßung und Einführung

10:00 Was ist eigentlich „Statische Codeanalyse“?

11:00 Refactoring: Chancen und Vorgehensweisen

15:00 DevSecOps – Detecting 0-day and N-day vulnerabilities, everyday.

16:00 Finding the Serious Bugs that Matter with Advanced Static Analysis



Agenda Tag 2



Donnerstag, 6. Mai

09:45 Begrüßung und Einführung in den zweiten Tag

10:00 Softwaremetriken:

Ein bewährtes Entwicklungswerkzeug oder nur ungeliebte Magic Numbers

11:00 Testen zur Laufzeit und statische Codeanalyse – zwei komplementäre Verfahren

15:00 Finding N-day Security Vulnerabilities in Third-party Software

16:00 Tools to Perform a Security Review on Unknown Code



Vielen Dank für Ihre Teilnahme ☺

Die Referenten und Ihr Team von Verifysoft Technology

