

## Statische Codeanalyse (SAST) und Software Composition Analyse (SCA) unterstützen bei der Umsetzung der Norm ISO/SAE 21434 „Road Vehicles – Cybersecurity Engineering“

Mit zunehmender Vernetzung und Komplexität von Fahrzeugen steigt der Umfang der Software stetig. 100 Millionen Codezeilen sind für herkömmliche Automobile Standard. Für autonome Fahrzeuge werden bis zu 300 Millionen Codezeilen benötigt. Mit der Konnektivität und Komplexität wächst auch die Angriffsfläche für Cyberattacken. Das Ziel von ISO/SAE 21434 ist es, diese Sicherheitsbedrohung zu bekämpfen und die Cybersicherheitstechnik von Automobilsoftware zu verbessern.

Die 2021 herausgegebene Norm ist eine Ergänzung der ISO-Norm 26262 und soll Ziele, Anforderungen und Richtlinien für die Sicherung elektrischer und elektronischer Systeme in Kraftfahrzeugen definieren.

### Statische Codeanalysetools unterstützen bei der Umsetzung der Norm

SAST-Werkzeuge (Static Application Security Testing) sind nützlich, um bestehende Implementierungs- und Testverfahren zu ergänzen, und sind ein zusätzliches Mittel für das Aufdecken von Fehlern und Schwachstellen im Code.

Werkzeuge für den statischen Sicherheitstest von Applikation wie beispielsweise CodeSonar der Firma GrammaTech bieten vor allem folgende Vorteile:

- **Durchsetzung von Codierrichtlinien für Safety, Security und Style:** durch die Automatisierung der Codeanalyse während der Codeentwicklung wird die Qualität im Entwicklungsprozess täglich sichergestellt.
- **Verringerung des manuellen Aufwands für den Nachweis der Robustheit und des Verhaltens von Software:** SAST-Tools ergänzen die Softwaretests, indem sie die Softwarequalität besser gewährleisten.
- **Reduzierung der Anzahl von Fehlern während der Entwicklung:** Code, der auf Anhieb funktioniert, ist viel besser zu testen und zu integrieren als fehlerhafter Code. Die Beseitigung von Fehlern durch statische Analyse früh im Entwicklungsprozess reduziert Kosten und Risiken.
- **Auffinden schwerwiegender Mängel, die sonst nur schwer oder gar nicht aufzudecken sind:** Trotz umfangreicher Tests, die für Automobilsoftware erforderlich sind, finden SAST-Tools oft noch Fehler, die durch dynamische Tests nicht aufgedeckt wurden.
- **Beschleunigung des Zertifizierungsnachweises:** die Dokumentation der Ergebnisse der Cybersicherheitsvalidierung ist entscheidend für den Nachweis der Einhaltung von Zertifizierungsstandards. SAST-Tools verfügen über umfangreiche Berichtsfunktionen, welche die Zertifizierungsanforderungen unterstützen.

Darüber hinaus helfen Werkzeuge für statische Sicherheitstests bei der Erkennung von Schwachstellen und vermeiden so unangemessene Risiken im Produkt (siehe Kapitel 11 der Norm).

Die Tools bieten hierfür unter anderem folgende Möglichkeiten:

- **Linksverschiebung (Shift Left) bei der Erkennung und Verhinderung von Schwachstellen:** SAST-Tools wie GrammaTech CodeSonar sind in die Entwicklungsumgebung des Entwicklers und die Build-Systeme des Projekts integriert. Die frühzeitige Erkennung schlechter Sicherheitspraktiken und möglicher Schwachstellen erfolgt daher bereits, sobald der Code geschrieben wird. Probleme werden dadurch schon aufgedeckt, bevor sie in Code-Repositories oder Unit-Tests gelangen. Hierdurch werden nachgelagerte Ressourcen gespart.
- **Kontinuierliche Bewertung des Quellcodes:** häufig werden Tools für die statische Sicherheitsüberprüfung zunächst auf eine große Codebasis als Teil der Erstintegration angewandt. Die wahre Stärke des Static Application Security Testing liegt aber in der entwicklungsbegleitenden Nutzung. Jeder neu geschriebene Codeblock (Datei oder Funktion) kann von den SAST-Tools gescannt werden, und die Entwickler können die Fehler und Warnungen schnell und effizient bearbeiten. Dies geschieht bevor der Code in das Build-System eingecheckt wird.
- **Erkennung und Analyse verfälschter Daten (Tainted Data):** die Analyse der Datenflüsse von den Quellen (den Schnittstellen) zu den Senken (wo die Daten in einem Programm verwendet werden) ist entscheidend für die Erkennung potenzieller Schwachstellen durch verfälschte Daten. Jede Eingabe, sei es von einer Benutzerschnittstelle oder einer Netzwerkverbindung, stellt bei ungeprüfter Verwendung eine potenzielle Sicherheitslücke dar. Code-Injektion und Datenlecks sind mögliche Folgen dieser Angriffe, die schwerwiegende Folgen haben können.

## Software-Wiederverwendung und die Lieferkette

Die ISO/SAE 21434 schreibt eindeutig vor, dass das Cybersecurity-Risikomanagement auf die gesamte Lieferkette angewendet werden muss, um die Cybersicherheitstechnik zu unterstützen. Daher müssen Sicherheitspraktiken und deren Kontrolle auf Lieferanten und auf alle gelieferten und wiederverwendeten Komponenten, einschließlich Software, angewendet werden. In der Regel werden in der Lieferkette Vereinbarungen über Cybersecurity-Schnittstellen geschlossen, um bei verteilten Cybersecurity-Aktivitäten gemeinsame Richtlinien und Verfahren zwischen Lieferanten und OEMs zu haben.

Außerdem wird von den Lieferanten erwartet, dass sie die Norm einhalten. Die Fähigkeit eines Lieferantenkandidaten, Entwicklungs- und gegebenenfalls Nachentwicklungsaktivitäten gemäß der Norm durchzuführen, muss daher bewertet werden. Hierfür ist es hilfreich, wenn der Lieferant einen Nachweis über die „Cybersicherheitsfähigkeit“ vorlegt, der bewährte Praktiken aus den Bereichen Entwicklung, Nachentwicklung, Governance, Qualität und Informationssicherheit beinhaltet. Jede Transaktion in der Lieferkette muss der ISO/SAE 21434 entsprechen.

Obwohl Software-Stücklisten (SBOM) in der Norm nicht explizit erwähnt werden, sind sie ein wichtiger Bestandteil von Software-Transaktionen zwischen Lieferanten. Sie

spielen eine grundlegende Rolle bei der Offenlegung von Schwachstellen und bei der Gewährleistung der Sorgfaltspflicht in den Prozessen des Lieferanten.

## Nutzen von Software-Stücklisten für die Softwareentwicklung im Automobilbereich

In Anlehnung an die ISO/SAE 21434 trägt die Einführung eines Risikomanagements für die Software-Lieferkette und die Verwendung von Software-Stücklisten wesentlich zur Verbesserung der Softwaresicherheit bei.

Wie bei physischen Stücklisten, die zur Verwaltung der Lieferkette von Teilen verwendet werden, helfen Software-Stücklisten bei der Überwachung und Verwaltung von Softwarekomponenten im Hinblick auf Sicherheitslücken und Lizenzierungsprobleme.

Die Integration der Softwarekompositionsanalyse (SCA) und die Verwendung von Softwarestücklisten als kritisches Entwicklungsartefakt hat viele Vorteile, darunter:

- **Identifizieren** von Open-Source-Komponenten in Code von Drittanbietern und COTS-/Drittanbieter-Software sowie Erkennen bekannter (N-Day) und unbekannter (Zero-Day) Sicherheitslücken in diesen Komponenten.
- **Verwalten**: Treffen von fundierten Sicherheitsentscheidungen auf der Grundlage von Einblicken in Code bzw. Software. Einhaltung von Sicherheits-, Lizenzierungs- und Herstellerrisiko-Anforderungen.
- **Beheben** von Schwachstellen zur Minimierung des Softwarerisikos durch Cybersecurity-Bedrohungen.

## Unterstützung durch Tools für die Binärcode-Analyse

Tools für die Binärcode-Analyse wie GrammaTech CodeSentry analysieren Open-Source-Software, Software von Drittanbietern und kommerzielle Standardsoftware (COTS) und zeigen die hierin enthaltenen Komponenten selbst dann auf, wenn nur Binärdateien verfügbar sind. Dabei wird eine Software-Stückliste und ein Schwachstellenbericht erstellt, der das Risiko der SOUP-Komponente bestimmt.

Des Weiteren bieten Software-Stücklisten folgende Vorteile:

- **Identifizierung und Vermeidung von Schwachstellen** in wiederverwendeten Komponenten selbst entwickelter bzw. im Unternehmen erworbenen Software.
- **Management von Risiken in der Software-Lieferkette** durch Bereitstellen von Daten für Entscheidungen über Softwarekäufe und die Wiederverwendung von Open Source Software.
- **Qualifizierung der Lieferkette** zur Gewährleistung von Konsistenz und Verantwortlichkeit der Lieferanten. Hierbei werden Lieferanten, die die Anforderungen erfüllen, bevorzugt behandelt.

- **Verbesserte Sicherheit und nachgelagerte Vorteile**, die sich aus dem Risikomanagement und der Risikominderung ergeben. Sicherheitsrisiken werden erkannt und vermieden, bevor sie ins Produkt gelangen.
- **Gemeinsames Verständnis von Software-Assets** zwischen Software-Entwicklern, Lieferanten und Open-Source-Projekten über standardisierte Software-Stücklisten. Hiermit werden Bestandteile der Software und deren Abhängigkeiten innerhalb und außerhalb der einer Organisation kommuniziert.

Software-Stücklisten werden in Zukunft als wichtiges Artefakt in der Software-Lieferkette das Mittel sein, um die Herkunft der in der Automobilbranche erworbenen Software nachzuweisen.

## Sichere Entwicklung und Codeprüfung

Die ISO/SAE 21434 verdeutlicht die Notwendigkeit eines organisatorischen Top-Down-Ansatzes für die Sicherheit. Diese muss als Teil der Unternehmenskultur bei jedem Entscheidungsschritt von Geschäftsleitung, Projektmanagement und der gesamten Lieferkette berücksichtigt werden. Als kontinuierlicher Prozess muss die Sicherheit die Grundlage der Softwareerstellung sein und darf nicht erst später eingebaut oder "hineingetestet" werden.

Spezifische Praktiken bei der Softwareentwicklung und dem Einsatz von Sicherheitstools beschreibt die Norm in den Kapiteln 10 (Produktentwicklung) und 11 (Validierung der Cybersicherheit). Das Ziel bei der Produktentwicklung ist es, die Definition, den Entwurf und die Validierung von Sicherheitsspezifikationen zu gewährleisten, gleichzeitig Schwachstellen zu identifizieren und einen Prüfpfad von der Spezifikation bis zur Verifizierung und Validierung aufzubauen. Bei der Cybersicherheitsvalidierung müssen Sicherheitsansprüche an das Produkt nachgewiesen werden. Es ist sicherzustellen, dass keine unangemessenen Sicherheitsrisiken verbleiben.

## Integration und Verifikation

Im Abschnitt 10.4.2 der ISO/SAE 21434 wird die statische Analyse ausdrücklich als empfohlene Methode zur Überprüfung von Sicherheitsspezifikationen erwähnt. Darin wird auch die Übereinstimmung mit den Modellierungs-, Design- und Kodierungsrichtlinien gefordert. Des Weiteren sind Programmierrichtlinien wie MISRA C oder CERT C und andere Methoden zur Durchsetzung guter Programmierpraktiken anzuwenden.

## Zusammenfassung

Tools zur Erstellung von Software-Stücklisten (SBOMs), zur Analyse der Softwarezusammensetzung (SCA) und für die statische Sicherheitsüberprüfung von Applikationen (SAST) spielen eine bedeutende Rolle bei der Entwicklung von

sicherer Software für Kraftfahrzeuge. Um die Sicherheit und Integrität der Software-Lieferkette zu gewährleisten, werden SCA-Tools bei der Generierung und Verifizierung von Software-Stücklisten für Open-Source-Software und Software von Drittanbietern genutzt.

Darüber hinaus helfen SAST-Tools dem Softwareentwicklungsteam bei der Einhaltung der Richtlinien und Standards bezüglich Softwarequalität, -sicherheit und -schutz. In Verbindung mit kontinuierlichen Integrations- und Auslieferungspipelines können automatisierte SAST-Tools Schwachstellen erkennen, bevor diese in das Code-Repository gelangen.

SCA- und SAST-Tools spielen eine immer wichtigere Rolle beim Nachweis der Sorgfaltspflicht der Hersteller, die ein wichtiger Bestandteil der Einhaltung von Normen wie der ISO/IEC 21434 ist.

## Weitere Informationen

GammaTech CodeSonar:

[https://www.verifysoft.com/de\\_grammatech\\_codesonar.html](https://www.verifysoft.com/de_grammatech_codesonar.html)

GammaTech CodeSentry

[https://www.verifysoft.com/de\\_grammatech\\_codesentry.html](https://www.verifysoft.com/de_grammatech_codesentry.html)

## Autor

Christian Simko, GammaTech (Ithaca/New York)



Deutsche Übersetzung: Verifysoft Technology GmbH (Offenburg) [www.verifysoft.com](http://www.verifysoft.com)

## Glossar

### ISO

Die International Organization for Standardization (ISO), auf Deutsch Internationale Organisation für Normung, erarbeitet internationale Normen in allen Bereichen mit Ausnahme der Elektrik und der Elektronik, für die die Internationale elektronische Kommission (IEC) zuständig ist, und mit Ausnahme der Telekommunikation, für die die Internationale Fernmeldeunion (ITU) zuständig ist.

### SAST

Static Application Security Testing (SAST) ist ein besonderes Sicherheitskonzept in der Softwareentwicklung, bei dem Überprüfungen bereits früh in der Entwicklungsphase stattfinden.

### SAE

Die SAE International, vormals Society of Automotive Engineers (SAE bzw. auf Deutsch „Verband der Automobilingenieure“) ist eine gemeinnützige Organisation für Technik und Wissenschaft, zur Förderung des Fortschritts in der Mobilitätstechnologie.

### SBOM

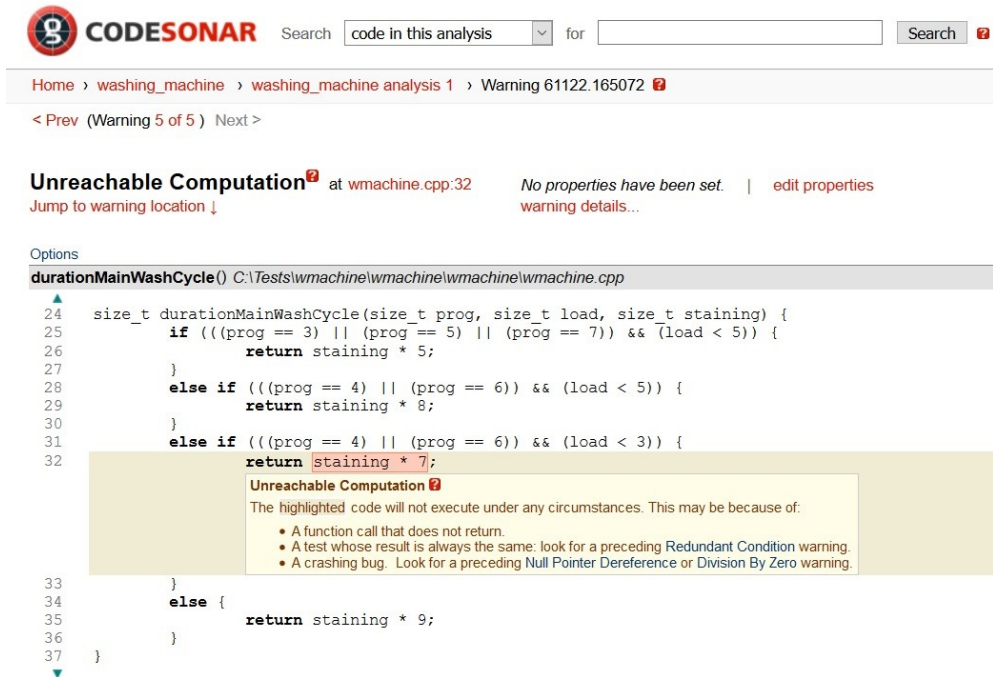
Software Bill of Materials (Software-Stücklisten) sind formale, maschinenlesbare Inventare von Softwarekomponenten und Abhängigkeiten. Sie wurden entwickelt, um die Details und Lieferkettenbeziehungen von Softwarekomponenten, ihre Abhängigkeiten und ihre hierarchischen Beziehungen zu erfassen.

### SCA

Software Composition Analysis (SCA) ist ein automatisierter Prozess, der die Nutzung von Open-Source-Software (OSS) zum Zwecke des Risikomanagements sowie der Sicherheit und der Einhaltung von Lizenzbestimmungen transparent macht.

### SOUP

Software of unknown (or uncertain) pedigree (or provenance). Zu Deutsch: Software unbekannter oder unsicherer Herkunft.



**CODESONAR** Search  for  Search

Home > washing\_machine > washing\_machine analysis 1 > Warning 61122.165072

< Prev (Warning 5 of 5) Next >

**Unreachable Computation** at `wmachine.cpp:32` *No properties have been set.* | [edit properties](#)  
[Jump to warning location](#) | [warning details...](#)

Options

`durationMainWashCycle()` C:\Tests\wmachine\wmachine\wmachine\wmachine.cpp

```

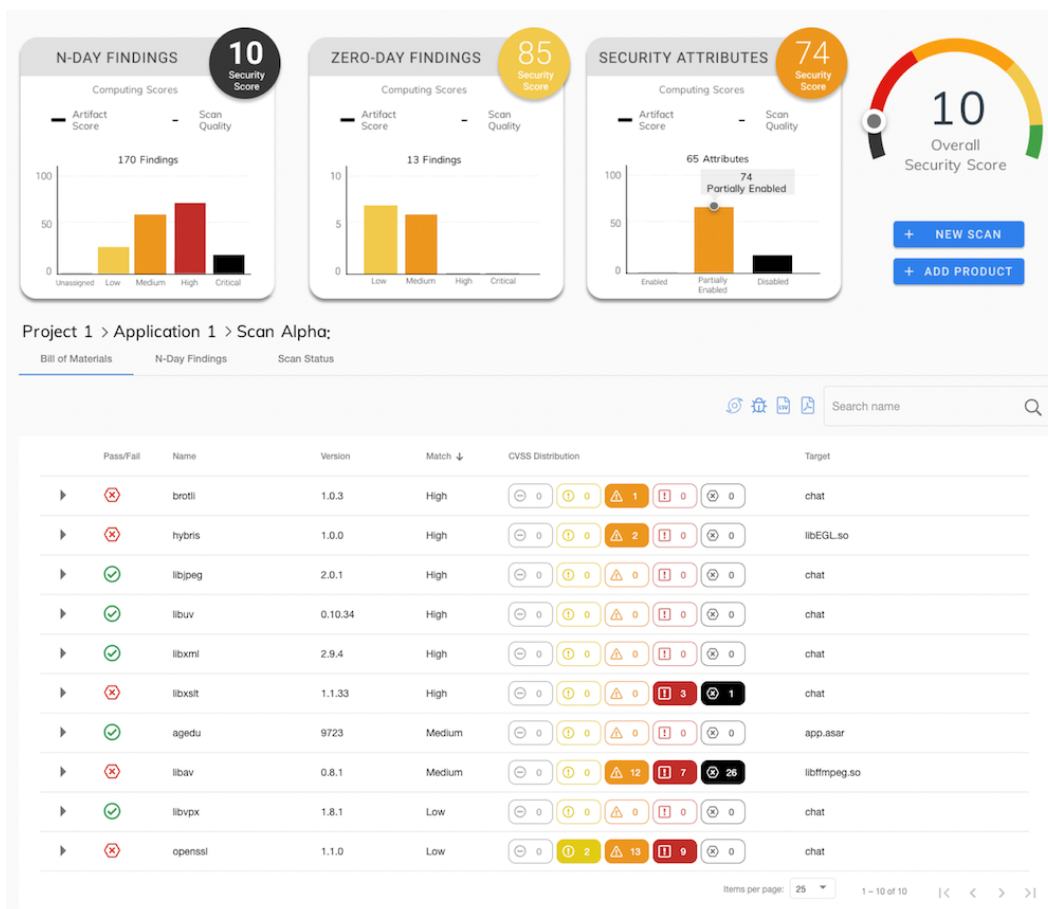
24 size_t durationMainWashCycle(size_t prog, size_t load, size_t staining) {
25     if ((prog == 3) || (prog == 5) || (prog == 7) && (load < 5)) {
26         return staining * 5;
27     }
28     else if ((prog == 4) || (prog == 6) && (load < 5)) {
29         return staining * 8;
30     }
31     else if ((prog == 4) || (prog == 6) && (load < 3)) {
32         return staining * 7;
33     }
34     else {
35         return staining * 9;
36     }
37 }

```

**Unreachable Computation**  
The highlighted code will not execute under any circumstances. This may be because of:

- A function call that does not return.
- A test whose result is always the same: look for a preceding Redundant Condition warning.
- A crashing bug. Look for a preceding Null Pointer Dereference or Division By Zero warning.

Bild 1: Statische Codeanalysetools wie GrammaTech CodeSonar decken Fehler und Schwachstellen im Code auf



**N-DAY FINDINGS** 10 Security Score  
Computing Scores: Artifact Score, Scan Quality  
170 Findings

**ZERO-DAY FINDINGS** 85 Security Score  
Computing Scores: Artifact Score, Scan Quality  
13 Findings

**SECURITY ATTRIBUTES** 74 Security Score  
Computing Scores: Artifact Score, Scan Quality  
65 Attributes (74 Partially Enabled)

**Overall Security Score: 10**  
+ NEW SCAN  
+ ADD PRODUCT

Project 1 > Application 1 > Scan Alpha:  
Bill of Materials | N-Day Findings | Scan Status

Pass/Fail	Name	Version	Match ↓	CVSS Distribution	Target
✗	brotli	1.0.3	High	0 0 1 0 0	chat
✗	hybris	1.0.0	High	0 0 2 0 0	libEGL.so
✓	libjpeg	2.0.1	High	0 0 0 0 0	chat
✓	libuv	0.10.34	High	0 0 0 0 0	chat
✓	libxml	2.9.4	High	0 0 0 0 0	chat
✗	libxslt	1.1.33	High	0 0 0 3 1	chat
✓	agedu	9723	Medium	0 0 0 0 0	app.asar
✗	libav	0.8.1	Medium	0 0 12 7 26	libffmpeg.so
✓	libvpx	1.8.1	Low	0 0 0 0 0	chat
✗	openssl	1.1.0	Low	0 2 19 9 0	chat

Items per page: 25 | 1 - 10 of 10

Bild 2: GrammaTech CodeSentry erstellt automatisch Software-Stücklisten (SBOM) für Open-Source- und COTS-Software sowie Binärcode aus Auftragsentwicklung