HITSSSE
Höhere IT-Sicherheit durch
Sichere Software Entwicklung

IT-Sicherheit
IN DER WIRTSCHAFT

Secure Coding für Embedded Systems

Hilfestellung für die Nutzung der Standards MISRA und SEI CERT für C/C++

Gefördert durch:

Bundesministerium
für Wirtschaft
und Klimaschutz

aufgrund eines Beschlusses
des Deutschen Bundestages

Disclaimer

Dieses Dokument wurde im April 2022 und mit den folgenden Versionen der hauptsächlich fokussierten Ressourcen erstellt: MISRA C:2012 Third Edition - First Revision, MISRA C++:2008, SEI CERT C 2016 Edition, SEI CERT C++ 2016 Edition. Es können Kompatibilitätsprobleme mit neueren oder abweichenden Versionen entstehen. Jegliche Erklärungen, Einschätzungen und betrachtete Software/Standards/etc. ist unabhängig von Dritten erstellt worden und deshalb „anbieterneutral“. Jegliche hier dargestellten Informationen werden ohne Gewähr zur Verfügung gestellt. Sie dienen als Hilfestellung und sollten erst nach eigener Evaluation verwendet werden. Dieses Dokument richtet sich an Entwickler und Personen, die zur Entscheidung über den verwendeten Standard beitragen.

1 Einführung

Im Bereich der Embedded Systems Software werden robuste und sichere Applikationen immer wichtiger. Probleme und unabsichtliche Fehler in Implementierungen führen oft zu Sicherheitsproblemen wie Memory Leaks, Instabilitäten oder Pufferüberläufen, die die generelle Memory Safety beeinträchtigen. Memory Leaks können durch fehlerhafte Verwendung von Pointer Arithmetik entstehen, was ohne entsprechende Hilfestellungen wie Coding-Standards oft schwer zu überblicken ist. Software-Instabilitäten treten auf, wenn compilerabhängige Implementierungen gewählt werden, die ein unbekanntes Verhalten hervorrufen können. Hierbei können die Coding-Standards „MISRA“ und „SEI CERT“ Abhilfe schaffen, da diese die Sprachen C und C++ auf ein definiertes Subset einschränken und dadurch Probleme im Code verringern können.

Ziel des Dokuments ist, einen Überblick über bestehende Guidelines für Secure Coding im Bereich der Embedded Systems zu vermitteln. Die verschiedenen Guidelines werden kurz vorgestellt. Hierbei wird insbesondere auf die spezifischen Anwendungsbereich (wie z.B. Automotive) eingegangen und welche Vorteile und Nachteile diese jeweils haben. Da in Embedded Systems zum sehr großen Teil die Sprachen C/C++ eingesetzt werden, beschränken sich die aufgezählten Guidelines auf diese Sprachen.

Am Ende des Dokuments soll eine bessere Abschätzung getroffen werden können, welcher der beiden betrachteten Standards für welchen Zweck geeignet ist und welche Tools dafür eingesetzt werden können.

2 Wichtige Guidelines für C/C++

In dem nachfolgenden Abschnitt wird kurz auf den grundlegenden Aufbau der Standards eingegangen und wie die Regelsets definiert sind.

2.1 MISRA-Standard

Der MISRA-Standard (MISRA-C und MISRA-C++) ist ein Programmierstandard, der seinen Ursprung in der Automobilindustrie findet. Dieser wurde von der Motor Industry Software Reliability Association (MISRA) entwickelt und wird kontinuierlich erneuert. Der erste Standard MISRA-C wurde 1998 definiert und seitdem in unregelmäßigen Abständen erneuert. 2008 wurde der MISRA Standard um eine C++ Version erweitert, den MISRA-C++ Standard.

Der MISRA-Standard definiert ein Subset innerhalb C und C++, welches jeweils ein vorhersagbares Verhalten verspricht. Dadurch werden Fehlerquellen entweder komplett entfernt oder verringert und die Portabilität bzw. Compilerunabhängigkeit verbessert. In Tabelle 1 werden einige Vor- und Nachteile des MISRA-Standards gegenüber gestellt.

Tabelle 1: Vor- und Nachteile des MISRA-Standards

Vorteile	Nachteile
• Vorhersagbares Verhalten (Reduzierung der Sprache auf das definierte Subset) • Verringerung der Compilerabhängigkeit • Detailreiches Regelset um Safety & Security sicherzustellen • Lizenzkosten des Regeltext: ca. 20 €	• Lizenzkosten für Linter/statische Analysetools • Komplexer Regelsatz, der fundiertes Sprachwissen voraussetzt

Der Regeltext selbst beläuft sich auf ca. 20 €. Theoretisch sind damit alle Kosten gedeckt, die zum Einsatz von MISRA nötig sind. Jedoch werden zum automatisierten Testen entsprechende statische Analysetools oder Linter benötigt. Diese sind meistens mit kommerziellen Lizenzen versehen und bringen entsprechende Lizenzkosten mit sich.

Der MISRA-C Standard arbeitet mit Regeln, die in 3 Kategorien unterteilt sind:

Mandatory: Regeln müssen eingehalten werden und es darf nicht abgewichen werden.

Required: Regeln sollen eingehalten werden. Abweichungen müssen aber über einen formalen Prozess beantragt, begründet, geprüft und dokumentiert werden.

Advisory: Regeln sollen eingehalten werden. Abweichungen dürfen aber mit Dokumentation ohne formalen Prozess umgesetzt werden.

Konformität zu diesen Regeln garantiert die *MISRA-Compliance*. Jedoch müssen dazu die Regeln nicht blind eingehalten werden. Abweichungen sind gestattet, wenn sie wie in den Regelkategorien erlaubt und dokumentiert sind.

MISRA-C++ hat zwei kleine Unterschiede zu MISRA-C. Der erste Unterschied ist, dass es keine „Mandatory“ Kategorie gibt, sondern die meisten Regeln unter „Required“ gelistet sind. Dadurch sind flexiblere Regelauslegungen möglich. Der zweite Unterschied ist eine zusätzliche, allgemeinere Kategorie „Document“. Dies sind auch vorgeschriebene Regeln, wie die „Mandatory“ Kategorie, beziehen sich jedoch auf allgemeinere Methoden wie z.B. jede Verwendung

von Inline-Assembler muss dokumentiert werden.

2.2 CERT C/C++

Der CERT Standard für sichere Programmierung wurde für verschiedene Sprachen (C, C++ und Java) vom Software Engineering Institute (SEI) der Carnegie Mellon University in Pittsburgh, Pennsylvania entwickelt. Der CERT C/C++ Standard beschreibt hierbei die C bzw. C++ Versionen des gesamten CERT Standards, auf welche hier der Fokus gelegt wird.

Der CERT Standard nutzt eine priorisierungsbasierte Bewertung von Sicherheitsproblemen und Coding Regeln. Diese Priorisierung ist in drei Level (L1, L2 und L3) unterteilt und berechnet sich aus mehreren Faktoren (Schweregrad, Wahrscheinlichkeit und Behebungskosten). Hierbei stellt L1 Probleme und Verstöße dar, deren Behebung hohe Priorität hat, da sie auf schwerwiegende Probleme hinweisen, die einfach zu beheben sind.

In Tabellen 2 bis 4 werden die einzelnen Faktoren und ihre Bedeutung aufgelistet.

Tabelle 2: **Schweregrad** – Wie schwer können Konsequenzen bei Missachtung der Regel sein?

Wert	Bedeutung	Beispiel einer Schwachstelle
1	Niedrig	DoS Attacke
2	Mittel	Verletzung der Daten Integrität (Data Integrity Violation)
3	Hoch	Ausführung von beliebigem Code (Arbitrary Code Execution)

Tabelle 3: **Wahrscheinlichkeit** – Wie wahrscheinlich kann ein Fehler durch Missachtung der Regel zu einer Schwachstelle führen?

Wert	Bedeutung
1	Unwahrscheinlich
2	Wahrscheinlich
3	Voraussichtlich

Tabelle 4: **Behebungskosten** – Wie teuer ist die Einhaltung der Regel?

Wert	Bedeutung	Erkennung	Behebung
1	Hoch	Manuell	Manuell
2	Mittel	Automatisch	Manuell
3	Niedrig	Automatisch	Automatisch

Jeder dieser drei Werte wird miteinander multipliziert um die Priorität zu berechnen. Anhand derer wird das Level der Schwachstelle ermittelt, was dabei helfen kann, das Beheben der Verstöße richtig zu priorisieren. In Tabelle 5 sind die Prioritäten zu den jeweiligen Level dargestellt.

Der CERT Standard hilft dabei, Sicherheitsprobleme und Coding Regeln zu kategorisieren und die Behebung dieser einfacher zu gestalten. Dabei verweist der Standard an verschiedenen Stellen auf andere Standards wie MISRA und „Common Weakness Enumeration (CWE)“. CERT hat ähnlich zu MISRA verpflichtende und empfohlene Regeln, welche alle einen Wert

Tabelle 5: **Priorität und Level**

Level	Priorität	Mögliche Interpretation
L1	12, 18, 27	**Schweregrad:** Hoch **Wahrscheinlichkeit:** Voraussichtlich **Behebungskosten:** Kostengünstig
L2	6, 8, 9	**Schweregrad:** Mittel **Wahrscheinlichkeit:** Wahrscheinlich **Behebungskosten:** Mittlere Behebungskosten
L3	1, 2, 3, 4	**Schweregrad:** Niedrig **Wahrscheinlichkeit:** Unwahrscheinlich **Behebungskosten:** Kostspielig

für Schweregrad, Wahrscheinlichkeit und Behebungskosten haben. Daraus errechnet sich für jede dieser Regeln die jeweilige Priorität und daraus das Level anhand der vorigen Tabellen.

Die Regeln sind aufgeteilt in einen Titel, eine Beschreibung und jeweils ein nicht-konformes und ein konformes Codebeispiel. Ein Beispiel für eine der Regeln ist hier zu finden.

3 Tools und Auswahlkriterien für Secure Coding Standards

Nachfolgend wird in Tabelle 6 verschiedene Tools aufgelistet, welche den MISRA oder CERT Standard unterstützen.

Tabelle 6: **Tools**

	MISRA C/C++	**SEI CERT C/C++**
Cppcheck (Open Source)	✓	✓
SonarQube (Open Source)	✓	✓
PC-Lint	✓	✓
Parasoft C/C++test	✓	✓
PVS-Studio	✓	✓
Polyspace	✓	✓
Klocwork		✓
IAR Systems Compiler	✓	✓
TASKING Compiler	✓	✓

Man kann erkennen, dass die meisten Tools beide Standards schon unterstützen. Daher kann der Einsatz beider Standards einfach erprobt werden. Lizenzkosten variieren hierbei stark und sind sehr individuell, weshalb für das jeweilige Tool und die Größe des Teams/des Unternehmens individuell betrachtet werden muss.

Beide vorgestellten Standards zielen auf das gleiche Ergebnis ab: Sichere und unabhängige Software im Embedded Bereich zu entwickeln. Zur Entscheidung, welcher der Standards als Grundlage für die Entwicklung sicherer Software verwendet werden soll, hilft meist eine Anforderungsanalyse, denn oftmals kommen die Vorgaben von Kunden oder Projektrichtlinien. Wenn die Software bzw. das Projekt im Sicherheitsbereich wie z.B. Automotive verortet ist, dann sind bestimmte Standards meist vorgegeben (im Fall von Automotive ist meist MISRA gefordert). Die Weiterentwicklung von SEI CERT ist im Vergleich zu MISRA schneller, was Vorteile und Nachteile bringen kann. Durch die hohe Verfügbarkeit von Tools und Compiler mit bereits integrierten Standards liegt es auch im Bereich des möglichen, beide Standards gleichzeitig abzudecken.

MISRA zielt durch den Ursprung des Standards mehr auf Safety-kritische Embedded Applikationen ab, während SEI CERT einen ursprünglich generischeren Ansatz verfolgt. MISRA ist restriktiver als SEI CERT, was unter Umständen nicht erwünscht sein kann.

Um die Entscheidung für den ersten verwendeten Standard zu vereinfachen, können drei Punkte betrachtet werden.

1. Wenn die Anforderungen einen Coding-Standard vorschreiben bzw. die Compliance zu einem Standards, dann ist es üblich MISRA dafür heran zu ziehen.
2. Sofern es keine Compliance Vorgaben oder Anforderungen gibt, lohnt es sich weiterhin, MISRA einzusetzen, da hierbei die höchste Codeintegrität gewährleistet wird.
3. Wenn hohe Zuverlässigkeit des Codes gewünscht,jedoch aber MISRA zu restriktiv erscheint, empfiehlt es sich, nur SEI CERT einzusetzen.

Abschließend kann gesagt werden, dass die Entscheidung für den ein oder anderen Secure Coding Standard immer individuell gefällt werden muss. Sicher ist jedoch, dass der Einsatz eines solchen Regelwerks die Qualität der entwickelten Software steigern wird, denn die Reduzierung der Programmiersprache auf ein definiertes Subset die Sicherheit gibt, dass jeder Codepfad ein vordefiniertes Verhalten und vorhersagbares Verhalten hat. Dies trägt maßgeblich zur Softwaresicherheit bei.

Impressum und Kontakt

Projekt HITSSSE – Höhere IT-Sicherheit durch Sichere Software Entwicklung

Immer mehr kleine und mittlere Unternehmen (KMUs) entwickeln Software für eigene Infrastrukturen oder Produkte. Hierbei herrscht meist ein hoher Zeitdruck und es stehen oft nur beschränkt personelle Ressourcen zur Verfügung. Oft werden inzwischen auch agile Softwareentwicklungsmethoden eingesetzt, die schnell einsetzbare Lösungen liefern sollen. Dadurch spielt die Sicherheit dieser Software oft eine untergeordnete Rolle, was sich letztendlich auch auf die IT-Sicherheit dieser Unternehmen und ihrer Kunden auswirkt. Im Fördervorhaben HITSSSE soll die IT-Sicherheit durch sichere Software Entwicklung für KMUs verbessert werden. Hierfür werden im Forschungsprojekt Handlungsempfehlungen sowie technische Hilfsmittel erstellt, die zuerst bei den assoziierten Partnern des Projekts konkret erprobt werden, um daraufhin generische Lösungsansätze für kleine und mittlere Unternehmen in Deutschland zu schaffen. Durch die Zusammenarbeit mit der Transferstelle „IT-Sicherheit in der Wirtschaft" soll die Breitenwirkung der entwickelten Angebote verstärkt werden. Gefördert wird das Projekt HITSSSE durch das Bundesministerium für Wirtschaft und Klimaschutz im Förderschwerpunkt Mittelstand-Digital. www.hitssse.de

Projektleitung

Prof. Dr.-Ing. Dominik Merli
Leiter HSA_innos
dominik.merli@hs-augsburg.de

Prof. Dr.-Ing. Alexandra Teynor
Leiterin HSA_ias
alexandra.teynor@hs-augsburg.de

Prof. Dr. Phillip Heidegger
HSA_ias
phillip.heidegger@hs-augsburg.de

Autoren

Philipp Schloyer, M.Sc.
philipp.schloyer@hs-augsburg.de

HSA_innos – Institut für innovative Sicherheit

HSA_innos hilft Unternehmen dabei, sich individuell zu schützen. Neben der Aus- und Weiterbildung von Sicherheitsexperten liegt der Schwerpunkt des Instituts auf der Entwicklung von Technologien und Prozessen für die IT-Sicherheit zur Anwendung in der Praxis. Zusammen mit HSA_innos schützen Unternehmen und andere Organisationen ihre Investitionen und Kunden vor digitalen Bedrohungen. Mehr Informationen zu HSA_innos finden Sie unter www.hsainnos.de.

HSA_innos
Institut für innovative Sicherheit

Hochschule
Augsburg University of
Applied Sciences

HSA_ias – Institut für agile Softwareentwicklung

Das Institut für agile Softwareentwicklung (HSA_ias) forscht in enger Zusammenarbeit mit Partnern aus Industrie und Wissenschaft zu den Schwerpunkten agile Softwareentwicklung, Programmiersprachen & Sicherheit, Prozessdigitalisierung sowie Anwendungen der KI. Die hierbei entstehenden Projekte decken eine breites Feld an Anwendungen ab, wie z.B. digitale Gesundheit, Produktionstechnik oder Digitalisierung der öffentlichen Verwaltung. Die Aus- und Weiterbildung von Software-IngenieurInnen für die Herausforderungen der Zukunft ist dabei ein zentrales Anliegen des Instituts.

Was ist Mittelstand Digital?

Das Mittelstand-Digital Netzwerk bietet mit den *Mittelstand-Digital Zentren*, der *Initiative IT-Sicherheit in der Wirtschaft* und *Digital Jetzt* umfassende Unterstützung bei der Digitalisierung. Kleine und mittlere Unternehmen profitieren von konkreten Praxisbeispielen und passgenauen, anbieterneutralen Angeboten zur Qualifikation und IT-Sicherheit. Das Bundesministerium für Wirtschaft und Energie ermöglicht die kostenfreie Nutzung und stellt finanzielle Zuschüsse bereit. Weitere Informationen finden Sie unter www.it-sicherheit-in-der-wirtschaft.de.

Mittelstand-
Digital