Build Process Documentation

# Revision

Version 4

9/8/23 12:08 PM

# SME

Charles Wilson

# Abstract

This document describes the process to document the security aspects of the build process and ensure that justification for the choice of tools is well-reasoned.

# Group / Owner

DevOps / Information Systems Security Developer

# Motivation

# This document is motivated by the need to properly document the build process from a security perspective. This is necessary given the nature of safety-critical, cyber-physical systems, subject to certifications such as **ISO/SAE 21434** and **ISO 26262**.

# License

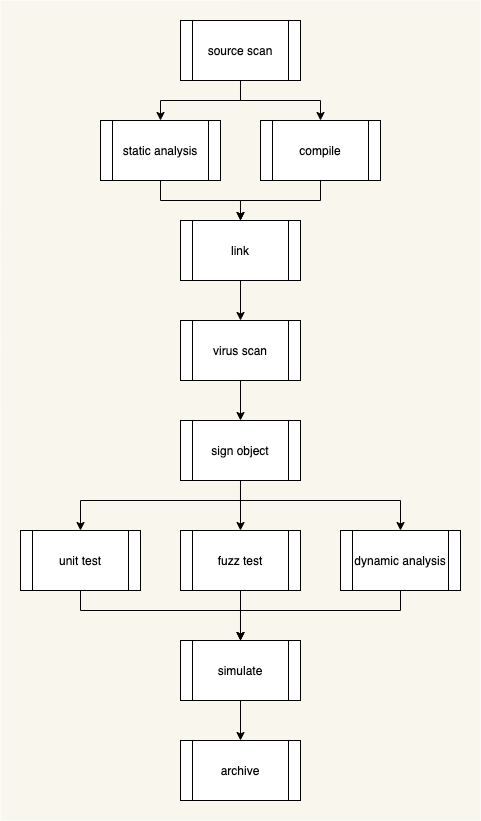
This work was created by **Motional** and is licensed under the **Creative Commons Attribution-Share Alike (CC4-SA)** License.

[**https://creativecommons.org/licenses/by/4.0/legalcode**](https://creativecommons.org/licenses/by/4.0/legalcode)

# Overview

Compilers and other development tools may provide security checking. Security-relevant tools must be enumerated, their security scope identified, and their use justified. It is important to realize that there is no single tool or process which when applied will detect every security issue. Because of this, multiple tools and processes may be employed to attain the level of coverage as deemed appropriate for the product and its application.

The following diagram illustrates a typical build workflow where the activities all have security relevancy:



# Documentation

This documentation is intended as a high-level summary showing how the various tools provide security coverage. It draws upon information curated in both during tool ingest and identification of available security-related options.

For each activity in the build process with security relevancy, the following must be detailed:

|  |  |
| --- | --- |
| **Information** | **Description** |
| **Activity** | Activity name |
| **Function** | Activity performed |
| **Tool** | Tool used to perform activity |
| **Input** | What the activity consumes |
| **Output** | What the activity produces |
| **Security relevancy** | Why the activity is security relevant |
| **Security coverage** | What security issues are detected |

For example:

|  |  |
| --- | --- |
|  |  |
| **Activity** | compile |
| **Function** | code compilation |
| **Tool** | clang |
| **Input** | source code |
| **Output** | object code, diagnostics |
| **Security relevancy** | function-scoped security checks |
| **Security coverage** | bad pointer behavior, uninitialized variable use |
|  |  |

**Note:** The **List of Approved Tools** AVCDL secondary document covers the ingest of tools used in product creation and their tracking.

**Note:** The **Secure Settings Document** AVCDL secondary document covers the specifics of individual tool security options and their justifications.

# References

1. **SAFECode Fundamental Practices for Secure Software Development**: *Use Current Compiler and Toolchain Versions and Secure Compiler Options*  
   <https://safecodedev.wpengine.com/wp-content/uploads/2018/03/SAFECode_Fundamental_Practices_for_Secure_Software_Development_March_2018.pdf>
2. **C-Based Toolchain Hardening**  
   <https://wiki.owasp.org/index.php/C-Based_Toolchain_Hardening>
3. **List of Tools Used** (AVCDL secondary document)
4. **Secure Settings Document** (AVCDL secondary document)