Fallgropar vid kvalificering av kompilatorbibliotek

EXPERTARTIKEL

Av Gerard Vink, Tasking

Gerard Vink har studerat maskinteknik och datavetenskap. Han har lång erfarenhet av att utveckla verktyg och processer för utveckling av mjukvara till inbyggda system.

Standarder för funktions- och cybersäkerhet betraktar kvalificering av verktyg och bibliotek som två oberoende uppgifter. Detta är oförenligt med perspektivet i standarden ISO C som ligger till grund för implementeringen av kompilatorn och dess bibliotek. Denna skillnad orsakar vissa utmaningar vid kvalificering av en kompilator med tillhörande kompileringsbibliotek.

Enligt standarderna ISO 26262 för funktionssäkerhet och ISO/SAE 21434 för cybersäkerhet måste programbibliotek, inklusive bibliotek som ingår i en kompilator, kvalificeras innan de integreras i fordonsmjukvaran. Båda standarderna betraktar kvalificering av verktyg och kvalificering av programbibliotek som två oberoende och orelaterade ämnen. Denna princip är giltig för de flesta typer av verktyg och bibliotek, men det är tveksamt om den också är giltig för en kompilator med tillhörande bibliotek.

Implementeringen av en C-kompilator med tillhörande bibliotek baseras på ISO/IEC 9899-standarden (ISO C), denna standard specificerar bland annat:

- egenskaper hos miljöer som översätter och exekverar C-program (Klausul 5), vilka ligger till grund för kraven på den så kallade startkoden
- språkets syntax, begränsningar och semantik (Klausul 6), som ligger till grund för implementeringen av kompilatorn (den körbara filen, verktyget) och tillhörande runtime-bibliotek
- och biblioteksfaciliteterna (Klausul 7), som beskriver innehållet i header-filerna och beteendet hos funktionerna i C-biblioteket

Varje paragraf i ISO C-standarden innehåller referenser till alla andra paragrafer, vilket indikerar beroenden mellan startkoden, kompilatorn, runtime-biblioteken, header-filerna och C-biblioteket. Det är därför inte klart i förväg om riktlinjerna för kvalificeringen av verktyg och eller bibliotek gäller för en specifik del av kompilatorns verktyg, och om ändringar i en del av verktygsuppsättningen ogiltigförklarar kvalificeringen av andra delar.

I den här artikeln beskrivs kortfattat kvalifikationskraven för verktyg och bibliotek i ISO 26262, hur en verktygsuppsättning för C-kompilatorer är uppbyggd, inklusive beroendena mellan komponenterna, och vilka kvalificeringskriterier som gäller för de olika komponenterna.

Att kvalificera en kompilator med avseende på funktionell säkerhet ses i allmänhet som ett problem som är löst, därför lägger denna artikel tyngdpunkten på kvalificering av programbibliotek. ISO FuSa-standarden tillhandahåller två metoder för kvalificering av programkomponenter som C-biblioteket. En metod är specifik för kommersiella standardkomponenter, den andra är för komponenter som utvecklats utanför sin kontext.

Kraven för att kvalificera kommersiella standardprogram (Commercial Off-the-Shelf) beskrivs i del 8, kapitel 12, i ISO FuSa-standarden. Två krav kan potentiellt orsaka problem vid kvalificering av ett C-bibliotek. För det första måste kraven för programkomponenten vara tillgängliga, inklusive krav på runtime-miljön och krav på den numeriska noggrannheten hos algoritmer som används i matematiska funktioner. För det andra är denna kvalificeringsmetod endast giltig för en icke modifierad implementation av programkomponenten.

Det är vanligtvis inte möjligt att hitta svaren för den här typen av detaljerade krav i användardokumentationen för en kompilator vilket gör det svårt för andra parter än utvecklaren av biblioteket att utföra en kvalificering. Dessutom kan problem som upptäcks under kvalificeringsprocessen inte åtgärdas eftersom detta skulle modifiera implementeringen av programkomponenten, man måste arbeta runt problemen.

Om ovanstående kan lösas, handlar kvalificeringsarbetet om att visa en kravtäckning i enlighet med ISO 26262 del 6, kapitel 9 Software Unit verification, vilket kräver tillgång till källkoden för C-biblioteket och runtime-biblioteken för kompilatorn, samt en testsvit för implementering av C-biblioteket för att uppfylla kravet på grentäckning för ASIL B och C plus kravet på MC/DC-täckning för ASIL D.

Kvalificering utanför kontexten

Kraven för kvalificering av ett säkerhetselement utanför kontext (SEooC) beskrivs i del 10, kapitel 12, i ISO FuSa-standarden.

Om C-biblioteket betraktas som ett SEooC omfattar kvalificeringen av biblioteket alla steg i säkerhetslivscykeln enligt ISO 26262 som även gäller för utveckling av programvara för fordon. Det tekniska säkerhetskonceptet måste utvecklas baserat på antaganden om användningen av programkomponenten. Därefter måste kraven på produktutveckling på programnivå uppfyllas, vilket inkluderar: tillämpning av en lämplig utvecklingsprocess, specifikation av säkerhetskrav, skapande av programarkitektur, design och implementering av programenheter, verifiering av programenheter och specificering av hur komponenterna ska integreras.

Förutom säkerhetslivscykeln för att säkerställa att komponenten inte introducerar ytterligare faror, måste man även ta hänsyn till den funktionella livscykeln för att säkerställa att komponenten fungerar som avsett.

Detta gör denna kvalificeringsmetod mycket mer komplicerad och dyr än den tidigare COTS-baserade metoden.

Fördelen för användaren av ett SEooC C-bibliotek är att biblioteket kan användas utan någon omkvalificering. Säkerhetsdokumentationen innehåller antaganden om säker användning och ger vägledning om hur man fastställer antagandenas giltighet under integrationen av biblioteket.

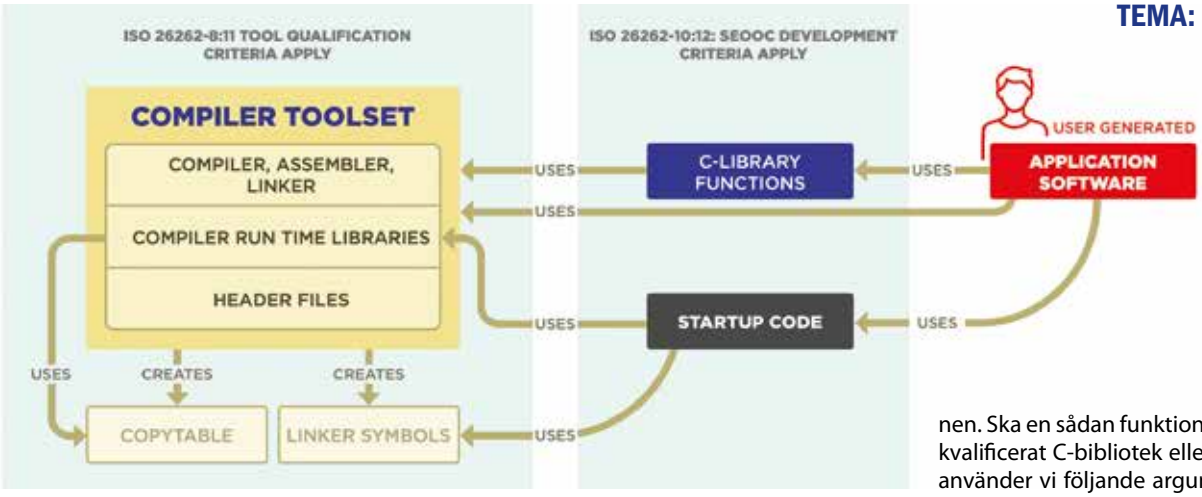
Struktur och kvalificering av verktygen

C-kompilatorer innehåller exekverbara filer som vanligtvis delas upp i en kompilator, en assembler och en linker. För kvalificering av dessa exekverbara filer gäller kvalificeringskriterierna för verktyg från ISO 26262 del 8, kapitel 11.

En kompilator innehåller också så kallade runtime-bibliotek och ett C-bibliotek. Det finns en grundläggande skillnad mellan dessa bibliotek. Runtime-biblioteken betraktas som en integrerad del av de exekverbara filer som utgör kompilatorn och kvalificeras med hjälp av kraven för verktyg. C-biblioteket betraktas som en programkomponent som är integrerad i fordonsmjukvaran och därför måste kvalificeras på ett annat sätt.

Kompilatorns runtime-bibliotek

För att förstå varför runtime-bibliotek anses vara en integrerad del av kompilatorn är det nödvändigt att förstå vad ett ”compiler runtime-library” är och varför sådana bibliotek existerar. Anta att målhårdvaran stöder 32-bitars flyttalsoperationer, men inte 64-bitars. Nu måste utvecklarna av kompilatorn implementera 64-bitars flyttalsoperationer



med hjälp av andra instruktioner. De subrutiner som utvecklats för detta ändamål lagras i ett så kallat runtime-bibliotek för flyttal. Dessa subrutiner skapas av kompilatorutvecklaren, anropas endast från kompilatorgenererad kod, specificeras inte i användardokumentationen och anropas aldrig direkt från användarens applikationsprogram. Därför betraktas dessa subrutiner och det tillhörande biblioteket som en integrerad del av kompilatorn och kvalifikationskraven för verktyget gäller. Organ för bedömning av överensstämmelse tenderar att följa denna argumentation och anser att runtime-biblioteken också är kvalificerade i samband med en kompilatorkvalificering.

Header-filerna

Det kan diskuteras om header-filerna skall betraktas som en integrerad del av kompilatorn eller av C-biblioteket. Innehållet i header-filerna, även kallade standard include-filer, specificeras i ISO C-standarden i kapitlet Library. Huvudfilerna innehåller prototyper av biblioteksfunktioner, makron och typdefinitioner som används av biblioteksfunktioner. Så kvalifikationskriterier för programkomponenter bör gälla.

Det finns dock även beroenden mellan header-filerna och kompilatorimplementeringen, till exempel är värdena för makrona i header-filen ”limits.h” alla beroende av och måste överensstämma med kompilatorimplementeringen. Därför bör kvalificeringskriterierna för verktyg gälla.

Organ för bedömning av överensstämmelse brukar anse att om rubrikfilerna har utvecklats av samma ingenjörer som utvecklade kompilatorn, så anses rubrikfilerna vara en del av kompilatorn och kvalificeras i samband med en verktygskvalificering. Om en rubrikfil ändras i samband med en bibliotekskvalificering upphör verktygskvalificeringen att gälla och måste göras om.

C-bibliotekets funktioner

C-biblioteket innehåller funktioner som utför grundläggande operationer inklusive minnesoperationer och matematiska lågnivåfunktioner som kvadratrot, potensfunktioner och trigonometriska funktioner som ofta används i fordonsmjukvara. Funktionerna i C-biblioteket beskrivs i användardokumentationen, anropas från användarens applikationsprogram och integreras i fordonsmjukvaran på samma sätt som andra mjukvarukomponenter. Det är därför uppenbart att en metod för kvalificering av programvarukomponenter skall tillämpas vid kvalificering av C-biblioteket. Men när man dyker ner i detaljerna blir det rörigt, nedan följer några exempel.

Funktioner som implementerar matematiska operationer med flyttal är vanligtvis mycket optimerade och använder funktioner hämtade från ett runtime-bibliotek för flyttal. Detta kvalificerades i samband med en verktygskvalificering, men nu har den avsedda användningen ändrats och kvalificeringskriterierna för programkomponenter gäller för de runtime-funktioner som används av C-biblioteket.

De C-biblioteksfunktioner som deklareras i headerfilen ”fenv.h” har åtkomst till flyttalsmiljön som är en del av kompilatorns runtime-miljö. Dess konfiguration påverkar beteendet hos kompilatorgenererad kod och beteendet hos flyttalens runtime-funktioner. För dessa funktioner går det att diskutera om kvalificeringskriterier för verktyg, för programkomponenter eller båda skall tillämpas.

ASIL-nivån för en specifik C-biblioteksfunktion är också diskutabel. Vissa funktioner i C-biblioteket är osäkra på grund av svagheter i de tidigare versionerna av ISO C-standarden och önskan att hålla standarden bakåtkompatibel. För att rätta till tidigare misstag introducerade kommitten för ISO C vad som kallas Annex K, som innehåller säkra(re) varianter av de osäkra funktionerna. På Tasking använder vi följande argumentation när vi kvalificerar de osäkra funktionerna: om en funktion har utvecklats i enlighet med kraven i ASIL x är funktionen kvalificerad för användning vid ASIL x under förutsättning att användaren tillämpar de riktlinjer som ges i säkerhetshandboken. Dessutom lägger Tasking till den säkrare varianten i C-biblioteket och rekommenderar att den funktionen används.

C-biblioteket innehåller också funktioner som vanligtvis inte används i färdiga produkter men som ofta används under utvecklingsarbetet för loggning och felsökning. Det mest typiska exemplet är printf-funktionen. Ska en sådan funktion vara en del av ett kvalificerat C-bibliotek eller inte? På Tasking använder vi följande argumentation: för att inte förstöra ett befintligt program när man byter från ett icke-kvalificerat bibliotek till ett kvalificerat bibliotek bör det senare tillhandahålla samma uppsättning funktioner, där varje funktion är kvalificerad som lämplig för antingen QM, ASIL A, B, C eller D-användning.

Startkoden

Startkoden för kompileringsverktyget kan levereras på flera sätt, som en del av C-biblioteket, som ett separat bibliotek eller som en uppsättning källfiler som måste inkluderas i användarens projekt. I samtliga fall är det en programkomponent som är integrerad i fordonsmjukvaran och därför bör reglerna för kvalificering av programkomponenter tillämpas.

Syftet med startkoden är att initialisera C-körtidsmiljön. Startkoden initialiserar olika register i processorn med hjälp av länksymboler som skapas av kompilatorn/linkern och vars existens och värden beror på användarens källkod. Dessutom initierar startkoden RAM-minnet med hjälp av en datastruktur som kallas ”the copytable”, som skapas av kompilatorn/linkern och vars innehåll beror på användarens källkod.

Gränsen mellan vad som är en del av verktygskvalificeringen och vad som är en del av programkomponentkvalificeringen är suddig. Vanligtvis anses konstruktionen av länksymbolerna och datastrukturerna vara en del av kompilatorkvalificeringen och användningen av symbolerna och data som en del av programkomponentkvalificeringen.

Slutsats

Att kvalificera ett C-bibliotek i enlighet med ISO 26262 är en komplex och utmanande process, eftersom det kräver djupgående kunskaper om FuSa-standarden, specifikationen för ISO C-biblioteket och den övergripande utformningen av kompileringsverktyget. Det krävs stor uppmärksamhet på detaljerna för att övervaka beroendena mellan alla komponenter i kompilatorn. Den detaljnivå som krävs för säkerhets- och funktionskrav, konstruktion och gränssnitt mellan komponenter är endast känd av verktygstillverkaren. De strängare FuSa-standarderna från flygindustrin kräver därför att de diskuterade kvalificeringsaktiviteterna alltid måste utföras av verktygstillverkaren. ■