

IAR C-STAT 静态代码分析使用指导

C-STAT 简要介绍和编码规则

C-STAT 是一种静态分析工具,它通过一次或多次规则执行检查来尝试查找与某些编码规则的偏差。在 C-STAT 的软件包中,检查规则选项分组排列。包括:

● STDCHECKS

包含来自 CWE 的检查规则,以及 C-STAT 的特殊检查规则。

● CERT

包含对 CERT 的检查规则。

• SECURITY

包含来自 SANS Top25 (违规最严重的 25 个检查规则), OWASP 和 CWE 的检查规则。

●MISRA C: 2004

包含对 MISRA C: 2004 标准选定的检查规则。该标准识别 C89 标准中的不安全代码结构。

●MISRA C++: 2008

包含对 MISRA C++: 2008 标准选定的检查规则。该标准识别 1998 C++标准中的不安全代码结构。

●MISRA C: 2012

包含对 MISRA C: 2012 标准选定的检查规则。该标准识别 C99 和 C89 标准中的不安全代码结构。

MISRA C 检查规则分为强制性的,必需的和劝告性的。 默认情况下,对强制性的规则和必需的规则进行检查, 而对劝告性的检查规则是关闭的。每个规则对应一个不安全的代码构造。

注意: C-STAT 可以对特定的文件指定分析的限制时间。达到时间限制后,将停止对其分析,继续进行下一个文 件的分析。

C-STAT 的各种使用方式

C-STAT 集成在 IAR Embedded Workbench 开发环境中,是其不可或缺的一部分:

- 可以在 Select C-STAT Checks 对话框中指定要执行检查条目。
- 通过从菜单 Project->C-STAT Static Analysis 中选择适当的命令,执行静态分析。
- 可以在 C-STAT 消息窗口中查看分析执行的结果。
- 通过从菜单 Project->C-STAT Static Analysis 中选择适当的命令,可以创建 HTML 格式的分析报告。

从命令行使用 C-STAT, 如果您使用 make 文件构建项目, 这非常有用:

- \bullet ichecks. exe 使用 ichecks 工具生成仅包含要执行检查规则的清单文件。
- ●icstat.exe 使用 icstat 工具对项目执行 C-STAT 静态分析,并将清单文件作为输入。
- ●ireport.exe 使用 ireport 工具生成以前执行的分析结果的 HTML 格式的报告。

最后,可以用命令行的 C-STAT 程序和 IAR 构建实用程序(iarbuild. exe)一起进行回归测试。

邮箱: info@bmrtech.com 网址: www.bmrtech.com



C-STAT 分析入门

1. 在执行静态分析之前,确保您的项目构建没有错误。

2. 选择Project->Options并选择Static Analysis类别。在C-STAT Static Analysis页面上,单击Select C-STAT Checks。

3. 在 Select C-STAT Checks 对话框中,选择检查要使用的规则包。例如: STDCHECKS。

			8		
Name	Severity	Used	Synopsis		
E STDCHECKS		152/213	C-STAT specific checks		
		28/43	Checks based on the CWE standard		
E CERT		All	Checks based on the CERT standard		
I MISRAC2004		116/135	Checks based on the MISRAC 2004 standard		
H MISRAC++2008		146/160	Checks based on the MISRAC++ 2008 standard		
MISRAC2012		175/226	Checks based on the MISRAC 2012 standard		

图 1. 选择规则包

4. 对于每个规则包,选择要检查的规则组或单独规则:

C-STAT checks Search:	The number of selected checks				
Name	Severity	Used	Synopsis	checks	
STDCHECKS		156/221	C-STAT specific checks		
ARR		5/6	Array bounds		
ARR-inv-index-pos	High		Array access may be out of bou		
ARR-inv-index-ptr	. Medi		A pointer to an array is potential		
	Select g	oups of	to an array is used out		
	. ugu	oupt of		ss is out of bounds.	
- ARR-neg-index	High		An array is accessed with a nega		
ARR-uninit-index	Medi	edi An array is indexed with an uni			
ATH		21/22	Arithmetic	errors	
🖶 🔽 CAST		None	Type casts		
Select or in dividual	deselect checks	F		OK Cancel	

图 2. 选择规则组

选择有关特定检查的信息,并按 F1 会打开相关的在线帮助系统。完成设置后,单击 OK,然后,再次单击 OK。

5. 要执行分析,确保项目处于活动状态并执行以下步骤之一:

● 要分析项目,请在 Workspace 窗口中选择项目,然后选择 Project->C-STAT Static Analysis ->Analyze Project。



● 要分析一个或多个单独的文件,请在 Workspace 窗口中选择文件,然后选择 Project->C-STAT Static Analysis->Analyze File(s)。

或者,使用 Workspace 窗口中的菜单上的相应命令。

注意:下次执行分析时,如果上次分析后对源代码进行了更改,则应首先清理数据库,避免由于混合数据库中的旧数据和新数据而导致问题。选择 Project->C-STAT Static Analysis ->Clear Analysis Results进行清理。

6. 分析的执行结果将显示在 C-STAT 消息窗口中。



图 3: C-STAT 分析结果

选择有关特定检查的信息,并按F1打开相关的在线帮助系统。

注意:如果分析时出现任何问题, Build Log 窗口将显示详细信息。

7. 双击 C-STAT 消息可以在编辑器窗口中查看相应的源代码:



图 4: 跳转到对应的源码

使用鼠标指针指向消息,可以获取导致消息产生检查的有关工具提示信息。

8. 更正错误,并单击 C-STAT Messages 窗口中的下一条消息。继续,直到所有消息处理完成。

注意: C-STAT 有一个预定义的宏__CSTAT_,可以使用它来明确地包含或排除分析中源代码的特定部分,参见 EW_CSTATGuide. ENU. pdf 手册第 25 页的 "__CSTAT_"小节。还有一些特定的 C-STAT 编译指示指令可以抑制一个 或多个检查选定的源代码行,请参阅 EW_CSTATGuide. ENU. pdf 手册第 22 页中 Descriptions of compiler extensions for C-STAT 的说明。



生成分析报告

- 1. 执行分析。
- 2. 生成报告:

●在集成开发环境中,选择Project>C-STAT Static Analysis,然后,根据要生成的报告类型选择 Generate HTML Summary 或 Generate Full HTML Report。

这将是最后一次执行分析的报告。 如果在分析执行后修改过源代码文件,则需要在生成报告之前更新分析。

```
●在命令行上,指定 ireport 选项,例如:
```

ireport --db cstat.db --project project1 --output tutor_report.html

这将从数据库 cstat.db 生成名为 tutor_report.html 的摘要报告,其中 project1 作为项目的标识名称。您可以从 Web 浏览器或 IAR Embedded Workbench 集成开发环境中查看报告。

3. 这是报告摘要示例:

ile Edit View Project	Tools	Win	dow Help		
🗅 🗅 🔛 🕋 🔚 🕹 🕹	ĎÖ	115) C - < Q > \$ H	= < 📮 > 🖸) 🖸 📓 😫 🖷 🔷 💽 📮 👪 👪
orkspace	•	ф ×	Information Center main.c IAR C-STAT Report	Generator X	
ebug		•			
ïles	•	•			
UART - Debug CMS CMS CMS Driver UART.c Device	~	•	C-STAT	ana	alysis su
RTE_Device.h			Concernt information		
- MCLib		•	General Information	0.017	-2
— 🖹 Readme.txt			Project name	UART	
E Output	C-STAT version	1.4.5			
	Timestamp of analysis Tue Sep 5	14:53:23 2017			
	Course files analyzed	0			
			Source files analyzed	0	-
		-	Header files analyzed	15	-
			Functions analyzed	197	
			Total messages	20	
					-
		Top 10 messages by check		Top 10 files by messages	
			20		20
			15 -		15

执行回归测试

回归测试是一种在修改源代码之后测试整个或部分源代码的方法,以验证修改后未添加任何错误。



1. 使用 C-STAT 分析项目并可能更正了一些错误后,使用位于 common\bin 目录中的 IAR 命令行生成实用程序 (iarbuild. exe),执行回归测试会很有用。

要从旧错误中清除数据库,请使用如下命令行:

iarbuild.exe MyProject.ewp -cstat_clean

调试要分析项目中的所有文件,请使用如下命令行:

iarbuild.exe MyProject.ewp -cstat_analyze Debug

2.C-STAT 生成输出信息,例如:

Analyzing configuration: MyProject - Debug Updating build tree... Starting C-STAT analysis Analysis completed. 164 message(s)

3. 比较报告的消息数量与先前构建中生成的消息数量。如果数量增加,则由于早期开发而引入了新的错误。

4. 在集成开发环境中, 打开项目, 执行分析并找出新消息的原因。

或者,您可以从命令行创建 HTML 报告,例如:

ireport.exe --db cstat.db --project MyProject.ewp --full - output MyProject.html

这将创建 MyProject.html 报告。

5. 通常,您可能希望在夜间构建期间重复此过程,以持续控制现有代码不受新代码的影响。

有关 IAR 命令行工具的更多信息,参阅 IDE Project Management and Building Guide 手册。

用命令行完成分析

要从命令行使用 C-STAT 执行分析, 您需要:

● ichecks.exe - 使用 ichecks 工具生成仅包含要执行的检查规则的清单文件。

● icstat.exe - 使用 icstat 工具对项目执行 C-STAT 静态分析,并将清单文件作为输入。 icstat 的输入内容包括:

- 使用编译器命令行的应用程序源文件。
- 应用程序的链接器命令行。



- 列出将要执行的已启用检查规则的一个文件。您可以使用 ichecks 工具创建此文件。
- 与执行检查规则的偏差文件将存储在数据库中。

有关如何使用 C-STAT 执行静态分析的示例,请根据两个示例源代码文件 cstat1.c 和 ctat2.c 执行以下步骤。 您可以在 target\src 目录中找到这些文件。

使用 C-STAT 执行静态分析

1. 通过使用 ichecks 创建清单文件来选择要执行的检查规则,例如:

```
ichecks --default stdchecks --output checks.ch
```

checks. ch 文件列出了您选择的所有检查规则,在本例中,默认情况下为 stdchecks 包(--default) 启用的所 有检查规则。该文件将如下所示:

ARR-inv-index-pos ARR-inv-index-ptr-pos

要在检查规则层修改文件,您可以用手动添加或删除文件中的检查规则。

2. 确保项目构建没有错误。

3. 要分析应用程序,请指定 icstat 命令。例如这样:

icstat --db a.db --checks checks.ch analyze - iccxxxxx compiler_opts cstat1.c icstat --db a.db --checks checks.ch analyze - iccxxxxx compiler_opts cstat2.c icstat --db a.db --checks checks.ch link_analyze - ilinkxxxxx linker_opts cstat1.o cstat2.o

注意: iccxxxxx 是编译器的调用, ilinkxxxxx 是 ILINK Linker 的调用。xxxxx 应替换为 IAR Embedded Workbench 产品包独有的标识符。请参阅随产品提供的编译器文档,了解 xxxxx 的替换内容。

如果您的产品包附带 IAR XLINK 链接器而不是 IAR ILINK 链接器,则 ilinkxxxxx 应为 xlink,目标文件的文件 扩展名 o 应为 rxx,其中 xx 是标识产品包的数字部分。有关替换 xx 的内容,请参阅 IDE Project Management and Building Guide。

在这些示例命令行中,-db 指定存储结果数据库的文件,-check 指定 checks. ch 清单文件。命令将以串行方式执行。

或者,如果要想分析更多源文件,并希望加快分析速度,可以使用 commands 命令,这意味着您可以结合--parallel 收集特定文件中的所有命令。在这种情况下,icstat 将并行执行分析。命令行将如下所示:



icstat --db a.db --checks checks.ch commands commands.txt --parallel 4

commands.txt包含:

analyze -- iccxxxxx compiler_opts cstat1.c
analyze -- iccxxxxx compiler_opts cstat2.c
link_analyze -- ilinkxxxxx linker_opts cstat1.o cstat2.o

请参阅上面有关ilinkxxxx和文件扩展名的说明。

注意:下次执行分析时,应首先使用clear命令清理数据库,以避免由于混合数据库中的旧数据和新数据而导致问题。

4. 运行icstat对cstat1.c文件检查之后,控制台上列出了这些消息,并存储在数据库中(假设执行了所有默认检查):

"cstat1.c",15 Severity-High[PTR-null-fun-pos]: Function call `f1()' is immediately dereferenced, without checking for NULL. CERT-EXP34-C,CWE-476 15: ! - possible_null 15: > - Entering into f1 7: ! - Return NULL

"cstat1.c",18 Severity-Low[RED-unused-assign]: Value assigned to variable `ch' is never used. CERT-MSC13-C,CWE-563

注意: 第一条消息后跟的跟踪信息,该信息描述了触发规则偏离的所需执行路径,包括有关条件语句的假设信息。

5. 为cstat2.c文件列出了此消息:

"cstat2.c",16 Severity-High[ARR-inv-index]: Array `arr' 1st subscript 20 is out of bounds [0,9]. CERT-ARR33-C, CWE-119, CWE-120, CWE-121, CWE-124, CWE-126, CWE-127, CWE-129, MISRAC++2008-5-0-16, MISRAC2012-Rule-1 8.1

编辑源文件以删除问题并重复分析。

注意: C-STAT具有内置预处理器符号__CSTAT__,您可以使用它来明确地包含或排除分析中源代码的特定部分。 还有一些特定的C-STAT编译指示指令可以抑制对所选源代码行的一个或多个检查,请参阅EW_CSTATGuide. ENU. pdf 手册的第22页Descriptions of compiler extensions for C-STAT的说明。