

代码检查

常见问题

文档版本 01
发布日期 2026-01-08



版权所有 © 华为技术有限公司 2026。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

安全声明

漏洞处理流程

华为公司对产品漏洞管理的规定以“漏洞处理流程”为准，该流程的详细内容请参见如下网址：

<https://www.huawei.com/cn/psirt/vul-response-process>

如企业客户须获取漏洞信息，请参见如下网址：

<https://securitybulletin.huawei.com/enterprise/cn/security-advisory>

目录

1 技术类问题	1
1.1 执行代码检查任务时提示：任务正在执行，稍后重试	1
1.2 执行代码检查任务时提示：权限不足，请核对后再试	2
1.3 执行代码检查任务时提示：单个任务并发数超过套餐限制，任务无法下发	2
1.4 执行代码检查任务时提示：Cppcheck cannot tokenize the code correctly	2
1.5 执行代码检查任务时提示：no such file or directory	3
1.6 执行代码检查任务时提示：在 Maven 仓库中找不到依赖	3
1.7 执行代码检查任务时提示：调用 CodeArts Repo 失败，未授权	4
1.8 执行代码检查任务时提示：CC.00070400.500	4
1.9 单击任务名称时提示权限不足	5
1.10 C#前端使用 WPF 组件的项目检查失败	5
1.11 执行任务检查完成后没有代码问题	5
1.12 TypeScript 任务检查失败，日志显示 404	7
1.13 CC.00130044.400 Execute source plugin failed	8
1.14 执行代码检查任务时报错：CC.00050032.400 没有可检查文件	9
1.15 执行代码检查任务时报错：CC.00050050.400 源分支不存在	9
1.16 执行代码检查任务时报错：CC.00020413.553 调用 CodeArts Repo 失败	10
1.17 执行代码检查任务时报错：CC.10010226.403 操作权限不足	10
1.18 执行代码检查任务时报错：CC.10010263.500 自定义执行机不可用	11
1.19 执行代码检查任务时报错：CC.00030117.400 规则集无版本级规则，请检查规则集配置	12
1.20 执行代码检查任务时报错：CC.00030308.400	12
1.21 执行代码检查任务时报错：CC.00030009.400 规则集信息不存在	13
1.22 执行代码检查任务时报错：CC.00040010.400 代码检查失败，请到检查日志中查看具体错误信息	14
1.22.1 文件格式导致编译失败问题	14
1.22.2 unicode 解析问题	15
1.22.3 findbugs 规则在扫描 jsp 文件时报错	16
1.22.4 代码检查任务，sonarqube 引擎的 check 阶段失败	16
1.22.5 sonarqube 引擎的 check 阶段失败日志里 elasticSearch 启动报错	17
1.22.6 sonarqube 引擎 check 阶段失败日志里 elasticSearch 启动报错	18
1.22.7 检查任务日志显示 Java heap space [ErrorMessage] className is java.util.Arrays, methodName is copyOf, lineNumber is ***	19
1.22.8 Sonarqube 检查 css 文件日志显示报错无扩展名问题	19
1.22.9 执行启动脚本无权限问题	20

1.22.10 SCC 初始化锁文件失败.....	21
1.22.11 sonarqube 服务启动失败.....	22
1.22.12 sonarqube 服务的 ES 报错磁盘内存不足.....	22
1.22.13 编译脚本出错，导致检查失败.....	23
1.22.14 内存不够，导致检查失败.....	24
1.22.15 子模块代码下载失败.....	24
1.23 使用代码检查服务时报错：CC.10010295.403 用户已被禁用.....	25
2 API 类问题.....	27
2.1 使用公共 API 时提示没有权限.....	27
2.2 使用公用 API 时提示项目不存在.....	27

1 技术类问题

1.1 执行代码检查任务时提示：任务正在执行，稍后重试

问题现象

执行任务失败，提示异常信息：任务正在执行，稍后重试。

原因分析

该任务正在运行，造成流水线执行失败。

处理方法

- 步骤1** [访问流水线服务首页](#)。
- 步骤2** 在流水线任务列表页单击失败的流水线任务名称。
- 步骤3** 单击流水线任务任一执行历史。
- 步骤4** 单击代码检查插件。
- 步骤5** 单击“详情”，从流水线任务中访问当前使用的代码检查任务。



- 步骤6** 确认当前任务是否正在运行，如果在执行请等待执行完成后，再开始运行流水线。
如果仍然未能解决，请[提交工单](#)联系技术支持工程师。

----结束

1.2 执行代码检查任务时提示：权限不足，请核对后再试

问题现象

执行任务失败，提示异常信息：权限不足，请核对后再试。

原因分析

当前用户操作权限不足，无法操作该任务，请用户根据权限矩阵，核对并联系项目管理员（项目创建者、项目经理）更改当前账号权限。

处理方法

步骤1 使用管理员账号访问CodeArts首页，选中要授权的项目（以Scrum项目为例）。

步骤2 进入项目“设置 > 权限管理”页面，查看自己的项目角色权限。

步骤3 根据权限矩阵，联系项目的管理员修改自己需要的“项目角色”。

----结束

1.3 执行代码检查任务时提示：单个任务并发数超过套餐限制，任务无法下发

问题现象

执行代码检查时提示：单个任务并发数超过套餐限制，任务无法下发！

原因分析

多条流水线同时执行同一个代码检查任务，超过了并发上限。

处理方法

建议同时执行的任务少一点。

1.4 执行代码检查任务时提示：Cppcheck cannot tokenize the code correctly

问题现象

代码检查提示“Cppcheck cannot tokenize the code correctly”。

原因分析

是Cppcheck的一条检查规则，在代码里面有语法错误，用Java语法写C代码造成的。

处理方法

要按照C语言的编码规范写代码，不能包含其他语言的编码规则。

1.5 执行代码检查任务时提示：no such file or directory

问题现象

代码检查任务失败，日志提示：no such file or directory。

原因分析

- 代码检查中用到自己封装的组件，未将组件上传到私有依赖库。
- 已经将组件上传到私有依赖库，但用户账号没有此私有依赖库的授权，导致从私有依赖库中下载组件失败。

处理方法

步骤1 将组件上传到私有依赖库。

步骤2 使用拥有私有依赖仓库管理员权限的账号进入私有依赖库的“独立用户权限”页面为用户账号分配此私有依赖库角色(开发者)来进行授权。

操作方法请参见：[管理用户权限](#)。

----结束

1.6 执行代码检查任务时提示：在 Maven 仓库中找不到依赖

问题现象

代码检查报错，错误日志中提示在Maven仓库中找不到依赖（该依赖为私有依赖）。

原因分析

项目中使用了私有依赖，但没有配置私有依赖扩展点。

处理方法 1

步骤1 在代码检查详情页，选择“设置 > 规则集 > 检查参数”。

步骤2 在编译命令中使用“-s settings.xml”指定Maven编译使用的settings。

----结束

处理方法 2

步骤1 在项目详情页，选择“设置 > 通用设置 > 服务扩展点管理”页面，增加“nexus repository”扩展点。

步骤2 在代码检查详情页，选择“设置 > 自定义环境 > 配置私有依赖仓扩展点”，选中新增的“nexus repository”扩展点。

----结束

1.7 执行代码检查任务时提示：调用 CodeArts Repo 失败，未授权

问题现象

执行任务失败，提示异常信息：调用CodeArts Repo失败：未授权。

原因分析

无Repo代码仓访问权限。

处理方法

参考[管理CodeArts权限](#)，为成员添加代码仓访问权限。

1.8 执行代码检查任务时提示：CC.00070400.500

问题现象

执行代码检查任务时提示，入库告警超30W限制。

可能原因

使用当前的规则集扫描，告警问题数量超过30W。

处理方法

检查结果中展现了TOP10问题规则名称及其数量。用户可根据任务的具体情况删除问题数较多的规则，直至总问题数降至30W以下后，再次进行代码检查。操作步骤如下：

步骤1 参考[查看代码问题](#)导出检查结果，并识别出问题数较多的规则。

步骤2 参考[创建自定义规则集](#)，在自定义的规则集中删除**步骤1**中识别的规则。

步骤3 参考[配置代码检查服务使用自定义规则集](#)，使用**步骤2**自定义的规则集重新执行代码检查任务。

步骤4 重复**步骤2~步骤3**，直至总问题数降至30W以下。

----结束

1.9 单击任务名称时提示权限不足

问题现象

在代码检查服务任务列表中，单击任务名称，提示权限不足。

原因分析

操作的用户项目权限不足，当账号的项目权限为“测试经理”、“测试人员（含跨租户）”、“浏览者（含跨租户）”时，没有访问查看代码检查任务的权限。

处理方法

联系项目管理员（项目创建者、项目经理）修改当前账号角色。

步骤1 管理员账号CodeArts首页，选中要授权的项目（以Scrum项目为例）。

步骤2 在项目内单击“设置 > 权限管理”。

步骤3 选择角色以及服务名称，勾选对应的权限。

----结束

1.10 C#前端使用 WPF 组件的项目检查失败

问题现象

C# WPF项目检查失败。

原因分析

Linux mono不支持Windows WPF（Windows Presentation Foundation）。

处理方法

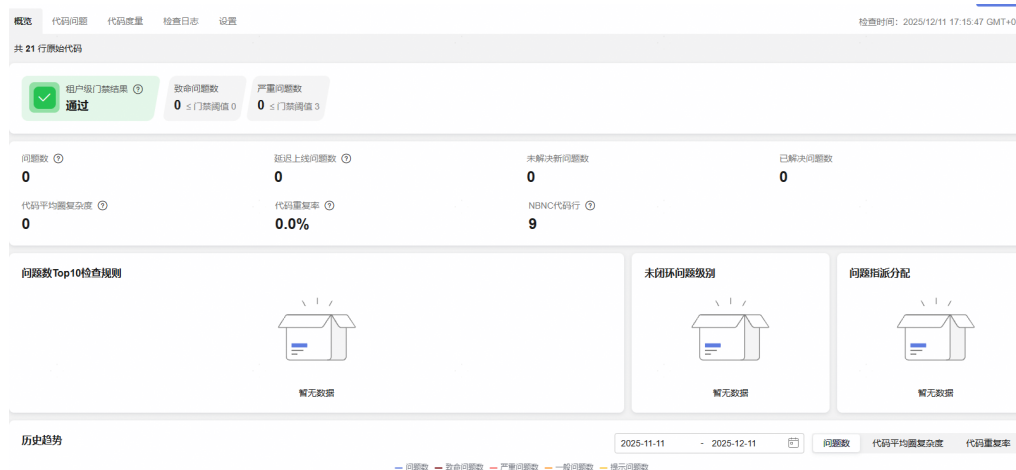
CodeArts中C#语言代码检查使用Linux mono方式时，C#前端WPF代码检查仅在Windows环境支持。

1.11 执行任务检查完成后没有代码问题

问题现象

在执行代码检查任务后没有代码问题，可能涉及以下三种原因：

图 1-1 执行任务检查完成后没有代码问题



- **可能原因一**：代码质量没有问题，无需处理。
- **可能原因二**：新建的代码任务语言规则集默认全部为打开状态，用户在已配置的代码仓库中新增了其他编程语言，但没有打开新增语言对应的规则集的开关，系统无法自动检测到这些新添加的语言，因此检查结果为空。
- **可能原因三**：代码仓库中存在多种编程语言，如果在任务设置中没有打开相应的语言开关，系统也不会对这些语言进行检查。因此，即使代码仓库中有相关文件，检查结果也会显示为空。

可能原因一的判断依据

步骤1 单击左侧导航栏“代码 > 代码检查”，进入“代码检查”页面，

步骤2 单击对应代码检查任务的名称，进入任务代码检查详情页面，单击“设置 > 规则集”。

如果用户期望扫描的语言和规则集选择正确且已打开，即代码质量没有问题，无需处理。


----结束


可能原因二的处理方法

步骤1 访问CodeArts Check服务首页。

步骤2 单击左侧导航栏“代码 > 代码检查”，进入“代码检查”页面，

步骤3 单击对应代码检查任务的名称，进入任务代码检查详情页面，单击“设置 > 规则集”。


步骤4 单击“已包含语言”所在行  按钮，重新获取代码仓语言。

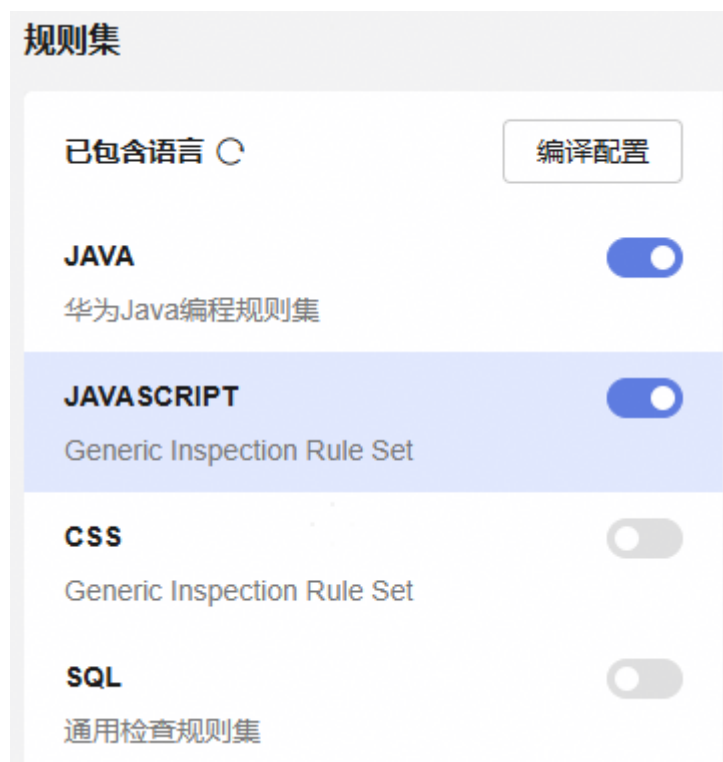
步骤5 单击  ，打开需要检查语言的开关。

步骤6 单击右上角的“执行检查”重新执行检查。

----结束

可能原因三的处理方法

- 步骤1** 访问[CodeArts Check服务首页](#)。
- 步骤2** 单击左侧导航栏“代码 > 代码检查”，进入“代码检查”页面，
- 步骤3** 单击对应代码检查任务的名称，进入任务代码检查详情页面，单击“设置 > 规则集”。
- 步骤4** 单击 ，打开需要检查语言的开关。



- 步骤5** 单击右上角的“执行检查”重新执行检查。

----结束

通过以上操作可以确保代码仓库中的所有语言都被正确识别并进行检查，从而避免因语言未被识别或开关未打开而导致的检查结果为空的问题。如果还有其他疑问或需要进一步的帮助，请[提交工单](#)联系技术支持工程师。

1.12 TypeScript 任务检查失败，日志显示 404

问题现象

TypeScript语言的任务检查失败，日志显示404。

原因分析

项目文件中包含了“package-lock.json”文件。

在检查TypeScript语言的代码质量时，通常不应该包含“package-lock.json”文件，因为这个文件主要用于管理项目的依赖关系和版本锁定。在代码质量检查过程中，通常

会忽略这个文件，因为它是根据项目中的“package.json”文件自动生成的，并且包含了依赖项的具体版本信息，不应该被手动修改或包含在代码质量检查的范围内。

另外，“package-lock.json”文件通常会随着项目的依赖项安装或更新而自动生成或更新，因此将其包含在代码质量检查中可能会导致不必要的冲突或误报。

处理方法

请检查上传的项目文件中是否包含“package-lock.json”文件。

- 如果包含，则删除“package-lock.json”文件，重新提交代码进行检查。
- 如果不包含，请[提交工单](#)联系技术支持工程师。

1.13 CC.00130044.400 Execute source plugin failed.

问题现象

- 如果日志中包含“The agent has not been fully initialized yet”表示引擎任务执行失败。参考本页面指导进行处理。
- 如果日志中包含“ERROR during SonarScanner execution”，表示文件格式导致的编译失败。参考[文件格式导致编译失败问题](#)进行处理。

原因分析

- 扫描使用的规则集中规则过多。
- 扫描代码量过大，导致执行机内存溢出。

处理方法

减少单个代码检查任务的代码扫描量，可通过以下3种方式。

- 将报错的代码检查任务拆分多个任务，并在拆分的任务中分别设置不同语言的规则集。设置规则集可参考[设置规则集](#)。
例如：报错的代码检查任务A中有多种语言的代码，Java、C、C#，则可以将任务A拆分为任务A1、任务A2和任务A3，其中A1使用Java语言的规则集扫描，A2使用C语言的规则扫描，A3使用C#语言的规则集扫描。
- 将报错的代码检查任务拆分多个任务，并在拆分的任务中，分别排除部分文件后进行扫描。设置排除文件可参考[设置检查模式](#)。
例如：将报错的代码检查任务A拆分为任务A1、任务A2和任务A3，其中A1扫描第1个文件，排除第2个和第3个文件，A2扫描第2个文件，排除第1个和第3个文件，A3扫描第3个文件，排除第1个和第2个文件。
- 将报错的代码检查任务使用的代码仓拆分成多个子仓分别创建代码检查任务进行扫描。

1.14 执行代码检查任务时报错：CC.00050032.400 没有可检查文件

问题现象

执行代码检查任务时，错误信息提示CC.00050032.400，没有可检查文件。

原因分析

- 配置了错误的检查目录，导致可扫描的代码源文件为空。
- 设置了忽略文件后，未忽略的代码源文件为空。

处理方法

参考[设置代码检查任务检查模式](#)，填写正确的检查目录路径。

1.15 执行代码检查任务时报错：CC.00050050.400 源分支不存在

问题现象

执行代码检查任务时，错误信息提示CC.00050050.400，源分支不存在。

原因分析

代码托管中，对应代码仓的设置中开启了“新建合并请求，默认开启合并后删除源分支”开关，同时流水线配置“执行计划 > 事件触发 > 合并请求时触发”中，开启了“合并”选项，此时，源分支已删除，导致代码检查无法做差分扫描。

处理方法

步骤1 进入代码托管服务。

步骤2 在对应代码仓详情页面，选择“设置 > 策略设置 > 合并请求 > MR设置”，取消勾选“新建合并请求，默认开启合并后删除源分支”选项，如下图所示。



----结束

1.16 执行代码检查任务时报错：CC.00020413.553 调用 CodeArts Repo 失败

问题现象

执行代码检查任务时，错误信息提示CC.00020413.553，代码检查调用CodeArts Repo的接口失败。

原因分析

- 原因一：仓库不存在或仓库已被删除。
- 原因二：未加入代码仓或者没有代码仓的读权限。

处理方法

- 原因一：
 - a. 在代码检查任务列表页，导航栏选择“代码 > 代码托管”，查看代码检查任务使用的代码仓是否存在。
 - b. 若不存在，需修改代码检查任务中使用的代码仓。若存在，联系技术支持处理。
- 原因二：
 - a. 在代码检查任务列表页，单击导航栏“设置 > 成员管理”，查看当前用户在当前项目的角色。
 - b. 参考[代码托管角色权限](#)，检查当前用户的角色，是否有代码检查对应的仓库的读权限。

1.17 执行代码检查任务时报错：CC.10010226.403 操作权限不足

问题现象

执行代码检查任务时，错误信息提示：CC.10010226.403，操作权限不足，请在项目内的设置-成员管理确认成员角色。

原因分析

操作代码检查任务时，没有代码检查相关的权限。

处理方法

- 步骤1** 在代码检查任务列表页，单击导航栏“设置 > 成员管理”，查看当前用户在当前项目的角色。
 - 步骤2** 单击导航栏“设置 > 权限管理”，根据**步骤1**得到的角色，单击对应的角色，然后展开“代码检查”，查看该角色在当前项目的代码检查权限。
 - 步骤3** 根据执行代码检查任务的实际操作，查看当前角色是否有对应的代码检查权限。
例如，修改代码检查任务的名称，需要有代码检查的检查任务的编辑权限。
 - 步骤4** 如果没有对应的代码检查权限，需联系项目创建者，编辑该角色对应的代码检查权限。
- 结束

1.18 执行代码检查任务时报错：CC.10010263.500 自定义执行机不可用

问题现象

执行代码检查任务时，错误信息提示：CC.10010263.500，自定义执行机不可用。

原因分析

用户设置了自定义执行机，但是自定义执行机没有配置正确。

处理方法

- 步骤1** 在代码检查任务列表页，单击代码检查任务名称。
 - 步骤2** 依次单击“设置 > 自定义环境”，选择“自定义资源池”。
 - 步骤3** 在“自定义资源池”区域，下拉“代理资源池”，可查看到当前设置的资源池的名称。
 - 步骤4** 单击“新建资源”，跳转到资源池管理页面。
 - 步骤5** 根据**步骤3**得到的资源池名称，在资源池管理页面单击该资源池名称，然后单击“新建代理”。根据弹窗的页面操作指导，完成代理的新建。
 - 步骤6** 重新执行检查任务即可。
- 结束

1.19 执行代码检查任务时报错：CC.00030117.400 规则集无版本级规则，请检查规则集配置

问题现象

执行代码检查任务时，错误信息提示：CC.00030117.400，规则集无版本级规则，请检查规则集配置。

原因分析

用户配置的规则集没有版本级规则，但是执行了版本级扫描，需要重新配置规则集。

处理方法

- 步骤1** 在代码检查任务列表页，单击代码检查任务名称，进入代码检查任务详情页。
- 步骤2** 依次单击“设置 > 规则集”，在右侧界面中选中当前选择的规则集，单击“查看详情”。
- 步骤3** 跳转到规则集界面后，单击左侧的“使用状态 > 未启用”。
- 步骤4** 在右侧区域勾选任意一条“适用范围”有“版本级”的规则，并在右上角单击“保存”。
- 步骤5** 重新执行检查任务即可。

----结束

1.20 执行代码检查任务时报错：CC.00030308.400

问题现象

执行代码检查任务时，错误信息提示：CC.00030308.400。

原因分析

- 原因一：代码量过大，任务执行时间超出默认的超时时间。
- 原因二：执行机规格不满足当前业务需求。
- 原因三：代码检查任务的执行时间超过了配置的超时时间。

处理方法

原因一：

减少单个代码检查任务的代码扫描量，可通过以下3种方式。

- 将报错的代码检查任务拆分多个任务，并在拆分的任务中分别设置不同语言的规则集。设置规则集可参考[设置规则集](#)。
例如：报错的代码检查任务A中有多种语言的代码，Java、C、C#，则可以将任务A拆分为任务A1、任务A2和任务A3，其中A1使用Java语言的规则集扫描，A2使用C语言的规则扫描，A3使用C#语言的规则集扫描。

- 将报错的代码检查任务拆分多个任务，并在拆分的任务中，分别排除部分文件后进行扫描。设置排除文件可参考[设置检查模式](#)。
例如：将报错的代码检查任务A拆分为任务A1、任务A2和任务A3，其中A1扫描第1个文件，排除第2个和第3个文件，A2扫描第2个文件，排除第1个和第3个文件，A3扫描第3个文件，排除第1个和第2个文件。
- 将报错的代码检查任务使用的代码仓拆分成多个子仓分别创建代码检查任务进行扫描。

原因二：

请联系技术支持，使用更大规格的执行机。

原因三：

参考[配置超时时间](#)重新配置代码检查任务的超时时间。

1.21 执行代码检查任务时报错：CC.00030009.400 规则集信息不存在

问题现象

执行代码检查任务时，错误信息提示：CC.00030009.400，规则集信息不存在。

原因分析

没有配置扫描的规则集，比如代码仓为空或者缺少必要的代码文件，会导致无法选中规则集。

处理方法

步骤1 在代码检查任务列表页，单击代码检查任务名称，进入代码检查任务详情页。

步骤2 单击“设置 > 规则集”，单击右侧界面中的“已包含语言”右侧的刷新按钮。



步骤3 如果刷新后仍没有出现规则集，请检查扫描代码仓中是否包含支持扫描的代码文件，新增相应代码文件后再次单击刷新按钮。

步骤4 重新执行检查任务。
----结束

1.22 执行代码检查任务时报错：CC.00040010.400 代码检查失败，请到检查日志中查看具体错误信息

1.22.1 文件格式导致编译失败问题

问题现象

日志中有“Error during SonarScanner execution”，且后续存在类似这种形式的日志。

原因分析

代码文件不规范导致解析失败。

处理方法

- 解决方案一：修改代码。
 - 修改日志指出的文件的代码。通过对日志进行全文搜索“Caused by”，可看到具体代码问题。

```
-----  
[INFO] - ERROR: Error during SonarScanner execution  
[INFO] - java.lang.IllegalStateException: Can not execute Checkstyle  
[INFO] - at org.sonar.plugins.checkstyle.CheckstyleExecutor.executeWithClassLoader(CheckstyleExecutor.java:110)  
[INFO] - at org.sonar.plugins.checkstyle.CheckstyleExecutor.execute(CheckstyleExecutor.java:78)  
[INFO] - at org.sonar.plugins.checkstyle.CheckstyleSensor.execute(CheckstyleSensor.java:42)  
[INFO] - at org.sonar.scanner.sensor.AbstractSensorWrapper.analyse(AbstractSensorWrapper.java:48)  
[INFO] - at org.sonar.scanner.sensor.ModuleSensorsExecutor.execute(ModuleSensorsExecutor.java:85)  
[INFO] - at org.sonar.scanner.sensor.ModuleSensorsExecutor.lambda$execute$1(ModuleSensorsExecutor.java:59)  
[INFO] - at org.sonar.scanner.sensor.ModuleSensorsExecutor.withModuleStrategy(ModuleSensorsExecutor.java:77)  
[INFO] - at org.sonar.scanner.scan.ProjectScanContainer.scan(ProjectScanContainer.java:392)  
[INFO] - at org.sonar.scanner.scan.ProjectScanContainer.doAfterStart(ProjectScanContainer.java:82)  
[INFO] - at org.sonar.core.platform.ComponentContainer.startComponents(ComponentContainer.java:137)  
[INFO] - at org.sonar.core.platform.ComponentContainer.execute(ComponentContainer.java:123)  
[INFO] - at org.sonar.scanner.scan.ProjectScanContainer.scanRecursively(ProjectScanContainer.java:388)  
[INFO] - at org.sonar.scanner.scan.ProjectScanContainer.doAfterStart(ProjectScanContainer.java:357)  
[INFO] - at org.sonar.core.platform.ComponentContainer.startComponents(ComponentContainer.java:137)  
[INFO] - at org.sonar.core.platform.ComponentContainer.execute(ComponentContainer.java:123)  
[INFO] - at org.sonar.batch.bootstrap.Batch.doExecute(Batch.java:72)  
[INFO] - at org.sonar.batch.bootstrap.Batch.execute(Batch.java:66)  
[INFO] - at org.sonarsource.scanner.api.internal.batch.BatchIsolatedLauncher.execute(BatchIsolatedLauncher.java:46)  
[INFO] - at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)  
[INFO] - at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(Unknown Source)  
[INFO] - at java.base/java.lang.reflect.Method.invoke(Unknown Source)  
[INFO] - at org.sonarsource.scanner.api.internal.IsolatedLauncherProxy.invoke(IsolatedLauncherProxy.java:68)  
[INFO] - at com.sun.proxy.$Proxy0.execute(Unknown Source)  
[INFO] - at org.sonarsource.scanner.api.EmbeddedScanner.doExecute(EmbeddedScanner.java:189)  
[INFO] - at org.sonarsource.scanner.api.EmbeddedScanner.execute(EmbeddedScanner.java:138)  
[INFO] - at org.sonarsource.scanner.cli.Main.execute(Main.java:112)  
[INFO] - at org.sonarsource.scanner.cli.Main.execute(Main.java:75)  
[INFO] - at org.sonarsource.scanner.cli.Main.main(Main.java:61)  
[INFO] - Caused by: com.puppycrawl.tools.checkstyle.api.CheckstyleException: Exception was thrown while processing ****code/wlms-stock/wlms-stock-  
ack/business/demandunfreezing/service/impl/DemandUnfreezingBillServiceImpl.java  
[INFO] - at com.puppycrawl.tools.checkstyle.Checker.processFiles(Checker.java:305)  
[INFO] - at com.puppycrawl.tools.checkstyle.Checker.process(Checker.java:224)
```

- 如果通过“Caused by”搜索不到错误文件，可搜索“sonarqube error logs begin”进行查找。

```
1676 [K:secolar] [INFO] [ProcessingListener:140] ... 38 more  
1677 [K:secolar] [INFO] [ProcessingListener:140] - Caused by: org.antir.vd.runtime.InputMismatchException  
1680 [K:secolar] [INFO] [ProcessingListener:140] ... 92 more  
1681 [K:secolar] [INFO] [ProcessingListener:140] - ERROR: No run SonarScanner using the -x switch to enable full debug logging.  
1682 [K:secolar] [INFO] [ProcessingListener:140] - ERROR: result code: 1  
1683 [K:secolar] [ERROR] [SonarQubeScanner:275]   
1684 [K:secolar] [ERROR] [SonarQubeScanner:275] Picked up JAVA_TOOL_OPTIONS: -Dfile.encoding=UTF-8  
1685 [K:secolar] [ERROR] [SonarQubeScanner:275] ERROR: Unable to parse source file: /secretdata/testcases/06111_HTTP_Response_Splitting/06111_HTTP_Response_Splitting_environment_additionalServlet_01.java [H]  
1686 [K:secolar] [ERROR] [SonarQubeScanner:275] ERROR: Parse error at line 41 column 8: Syntax error, insert "VariableDeclarator" to complete localVariableDeclaration() [H]  
1687 [K:secolar] [ERROR] [SonarQubeScanner:275] Jun 29, 2025 2:44:32 AM net.sourceforge.pmd.RuleSetFactory.parseRuleSetFromCode [H]  
1688 [K:secolar] [ERROR] [SonarQubeScanner:275] WARNING: Discontinue using http://www.cerify.com/category/java/robotframework/robotframework-11 is scheduled for removal from PMD. PMD 7.0.0 will remove support for this Rule. [H]  
1689 [K:secolar] [ERROR] [SonarQubeScanner:275] Jun 30, 2025 2:44:32 AM net.sourceforge.pmd.RuleSetFactory.parseRuleSetFromCode [H]
```

- 解决方案二：忽略问题文件。

- 方式一：代码根目录下新建“sonar-project.properties”文件，文件中配置“sonar.exclusions”属性忽略该检查文件，如下图所示。

```
sonar.exclusions=**/src/main/java/com/boco/scms/stock/business/demandunfreezing/service/impl/DemandUnfreezingBillServiceImpl.java,
```

多个文件可用逗号隔开。

- 方式二：2023.1130之后的版本可通过在[设置代码检查任务检查模式](#)中手动勾选忽略的文件集合，如下图所示。



1.22.2 unicode 解析问题

问题现象

sonarqube引擎的check阶段显示失败，并且日志里提示存在无法解析的字符。

原因分析

Java编译器不仅会编译代码，同时也会解析unicode字符，并且unicode字符的优先级最高。“\u000d”是换行符，在编译器中，会识别后续代码为下一行，所以不会显示格式问题。

而在实际sonarqube分析中，“\u000d”所在行都已被注释，因此会多一个右括号，在使用涉及编译的规则时导致编译出错。

```
137         return count;
138     } finally {
139         //\u000d if (rs != null) {
140             rs.close();
141         }
142         //\u000d if (ps != null) {
143             ps.close();
144         }
145         if (conn != null) {
146             conn.close();
147         }
148     }
```

处理方法

删除“//”，或者将左括号手动换行。

1.22.3 findbugs 规则在扫描 jsp 文件时报错

问题现象

```
[2023/12/19 16:15:44.888] [INFO] [CHECK:secular] [INFO] - INFO: Sensor Findbugs Sensor [findbugs]
[2023/12/19 16:15:44.889] [INFO] [CHECK:secular] [INFO] - INFO: Findbugs plugin version: 4.2.3
[2023/12/19 16:15:45.332] [INFO] [CHECK:secular] [INFO] - INFO: javaNucleusLocator.binaryDir[] not available before SonarQube 9.8
[2023/12/19 16:15:45.337] [INFO] [CHECK:secular] [INFO] - WARN: JSP files were found in the context (subject: ****/code) but Findbugs requires their precompiled form, for more information on how to configure JSP precompilation: https://github.com/find-sec-bugs/find-sec-bugs/wiki/JSP-precompilation
[2023/12/19 16:15:45.337] [INFO] [CHECK:secular] [INFO] - INFO: javaNucleusLocator.testClasspath[] not available before SonarQube 9.8
[2023/12/19 16:15:45.337] [INFO] [CHECK:secular] [INFO] - WARN: Findbugs needs sources to be compiled. Please build project before executing sonar or check the location of compiled classes to make it possible for Findbugs to analyze your (subject: ****/code).
[2023/12/19 16:15:45.338] [INFO] [CHECK:secular] [INFO] - WARN: Property 'sonar.java.compiler' is not declared as multi-values-property set but was read using 'getStringProperty' method. The SonarQube plugin declaring this property should be updated.
```

原因分析

Findbugs分析的不是java源代码，而是编译后的class文件。通过下图进行配置忽略未编译的jsp文件。

日志里有这种涉及jsp文件的警告时，需要在代码仓里排除.jsp文件。

处理方法

- secsolar1130之后的版本可参考[设置代码检查任务检查模式](#)设置忽略jsp文件。
- secsolar1130之前的版本需要添加properties文件排除。
 - a. 在代码仓根目录下新建名为“sonar-project.properties”的文件。
 - b. 将以下代码输入至“sonar-project.properties”文件中。

```
sonar.findbugs.allowuncompiledcode=true
sonar.exclusions=**/*.jsp
```

1.22.4 代码检查任务，sonarqube 引擎的 check 阶段失败

问题现象

代码检查任务，在sonarqube检查的check阶段失败，日志里存在elasticSearch启动报错“max virtual memory areas vm.max_map_count [65530] is too low, increase to at least [262144]”。

原因分析

系统默认环境配置的vm.max_map_count过低。

处理方法

方法一：

1. 以root用户登录执行代码检查任务的执行机。
2. 执行**sudo vi /etc/sysctl.conf**命令修改“sysctl.conf”文件。
3. 在文件中增加以下代码。

```
vm.max_map_count=655360
```
4. 按“Esc”键返回普通模式。
5. 输入“:wq”保存并退出。退出文件后，执行**sysctl -p**命令。
6. 重新执行代码检查任务。

方法二：

登录执行机直接运行命令**echo "vm.max_map_count=655360" >>/etc/sysctl.conf&&sysctl -p**。

1.22.5 sonarqube 引擎的 check 阶段失败日志里 elasticSearch 启动报错

问题现象

elasticSearch启动报错“max file descriptors [4096] for elasticsearch process is too low, increase to at least [65535]”。

原因分析

单个进程可以同时打开的文件数太少。

处理方法

步骤1 以root用户登录执行代码检查任务的执行机。

步骤2 执行以下命令，查看当前可打开的文件数。

```
ulimit -Hn  
ulimit -Sn
```

步骤3 执行“vim /etc/security/limits.conf”命令，打开文件。

步骤4 增加以下配置命令，用户退出后重新登录生效。

```
* soft nofile 65536  
* hard nofile 65536
```

命令前需要注意带“*”号。

步骤5 按“Esc”键返回普通模式。

步骤6 输入“:wq”保存并退出。

步骤7 执行以下命令，查看当前可打开的文件数是否已修改为65536。

```
ulimit -Hn
ulimit -Sn
```

----结束

1.22.6 sonarqube 引擎 check 阶段失败日志里 elasticSearch 启动报错

问题现象

2、max number of threads [3818] for user [es] is too low, increase to at least [4096]

问题同上，最大线程个数太低。修改配置文件/etc/security/limits.conf（和问题1是一个文件），增加配置

```
1 | *          soft  nproc    4096
2 | *          hard  nproc    4096
```

可通过命令查看

```
1 | ulimit -Hu
2 | ulimit -Su
```

```
[root@localhost ~]# ulimit -Hu
3818
[root@localhost ~]# ulimit -Su
3818
[root@localhost ~]#
```

修改后的文件:

```

#*          soft  core      0
#*          hard  rss       10000
#@student  hard  nproc     20
#@faculty  soft  nproc     20
#@faculty  hard  nproc     50
#ftp       hard  nproc     0
#@student  -     maxlogins 4
*          soft  nofile    65536
*          hard  nofile    65536
*          soft  nproc     4096
*          hard  nproc     4096
End of file

```

原因分析

系统最大线程个数太小。

处理方法

步骤1 以root用户登录执行代码检查任务的执行机。

步骤2 执行以下命令，查看当前系统最大线程数。

```
ulimit -Hu
ulimit -Su
```

步骤3 执行“vim /etc/security/limits.conf”命令，打开文件。

步骤4 增加以下配置命令，用户退出后重新登录生效。

```
* soft nproc 4096
* hard nproc 4096
```

命令前需要注意带“*”号。

步骤5 按“Esc”键返回普通模式。

步骤6 输入“:wq”保存并退出。

步骤7 执行以下命令，查看当前可打开的文件数是否已修改为4096。

```
ulimit -Hn
ulimit -Sn
```

----结束

Exception in thread "main" java.nio.file.AccessDeniedException: /usr/local/elasticsearch/elasticsearch-6.2.2-1/config/jvm.options
elasticsearch用户没有该文件夹的权限，执行命令

```
chown -R es:es /usr/local/elasticsearch/
```

1.22.7 检查任务日志显示 Java heap space [ErrorMessage] className is java.util.Arrays, methodName is copyOf, lineNumber is ***

问题现象

Sonarqube引擎的check阶段失败，日志显示Java堆内存失败“Java heap space [ErrorMessage] className is java.util.Arrays, methodName is copyOf, lineNumber is ***”

```
2024/07/10 15:19:47.416 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:47.399][INFO] [main][com.huawei.seccolar.plugin.report.util.ScannerReportUtil:215] - extract defects finished
2024/07/10 15:19:47.648 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:47.644][INFO] [main][com.huawei.seccolar.plugin.business.SonarQubeExecutor:167] - generate defect num: 128263
2024/07/10 15:19:47.649 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:47.645][INFO] [main][com.huawei.seccolar.plugin.core.util.GenerateReportUtil:40] - generate defect report and metric report
2024/07/10 15:19:47.649 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:47.645][INFO] [main][com.huawei.seccolar.plugin.core.util.GenerateReportUtil:50] - report path: ****/upload/sonarqube/seccolar_report.json
2024/07/10 15:19:49.871 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:49.870][ERROR][main][com.huawei.seccolar.plugin.business.SonarQubeExecutor:177] - write report file failed: java heap space [ErrorMessage] className is java.util.Arrays, methodName is copyOf, lineNumber is 3745
2024/07/10 15:19:49.871 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:49.871][ERROR][main][com.huawei.seccolar.plugin.business.SonarQubeExecutor:177] - write report file failed: java heap space
2024/07/10 15:19:49.871 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:49.871][INFO] [main][com.huawei.seccolar.plugin.core.util.GenerateReportUtil:40] - generate defect report and metric report
2024/07/10 15:19:49.871 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:49.871][INFO] [main][com.huawei.seccolar.plugin.core.util.GenerateReportUtil:50] - report path: ****/upload/sonarqube/seccolar_report.json
2024/07/10 15:19:49.872 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:49.871][ERROR][main][com.huawei.seccolar.plugin.core.util.GenerateReportUtil:91] - write defect report defect result is null
2024/07/10 15:19:49.872 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:49.871][ERROR][main][com.huawei.seccolar.plugin.core.util.GenerateReportUtil:83] - write report file failed!
2024/07/10 15:19:49.872 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:49.871][INFO] [main][com.huawei.seccolar.plugin.process.AnalyzeToolProcessor:123] - analyze result: FAILURE
2024/07/10 15:19:49.872 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:49.872][INFO] [main][com.huawei.seccolar.plugin.process.AnalyzeToolProcessor:139] - ===== finish all analysis execute, cost time is 118
2024/07/10 15:19:49.872 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:49.872][INFO] [main][com.huawei.seccolar.plugin.Boot:100] - ===== Total time: 118 seconds =====
2024/07/10 15:19:49.878 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:49.878][ERROR][main][com.huawei.seccolar.plugin.Boot:107] - Base info is: BaseInfo{Param{queryParams={4452484809c24a339a518003366b6f62, requestId=041ab2275b64beeb1ac6d88bc9f5f6}, projectId=c32454594d944436801697d431bcb, projectName=phoenix-codecheck-voip-ab93a7, region=green, environment=cn-north-5, type=version, engineName=sonarqube, engineVersion=0, projectBoot=****code, toolParameterMap={fooParameterMap{seccolar=null}}, includePaths= MavenOptions{null, udepart=null}}
2024/07/10 15:19:49.878 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:49.878][INFO] [main][com.huawei.seccolar.plugin.Boot:108] - SecSolar Plugin Failed.
2024/07/10 15:19:49.880 GMT+08:00 [INFO] [CHECKSseccolar] : [2024-07-10 07:19:49.879][INFO] [Thread-5][com.huawei.seccolar.plugin.util.HttpClientUtil:70] - begin close HttpClient
2024/07/10 15:19:49.955 GMT+08:00 [ERROR] [CHECKSseccolar] : Some unexpected problems occurred while executing the SecolarPlugin.
2024/07/10 15:19:49.962 GMT+08:00 [ERROR] [CHECKSseccolar] : source execute jar failed!
2024/07/10 15:19:49.962 GMT+08:00 [ERROR] [CHECKSseccolar] :
2024/07/10 15:19:49.962 GMT+08:00 [ERROR] [CHECKSseccolar] : Some unexpected problems may occur while the SecolarPluginOtcSource is running.
```

原因分析

检测到的缺陷数过多，执行机内存有限，在转换缺陷报告过程中堆内存溢出导致失败。

处理方法

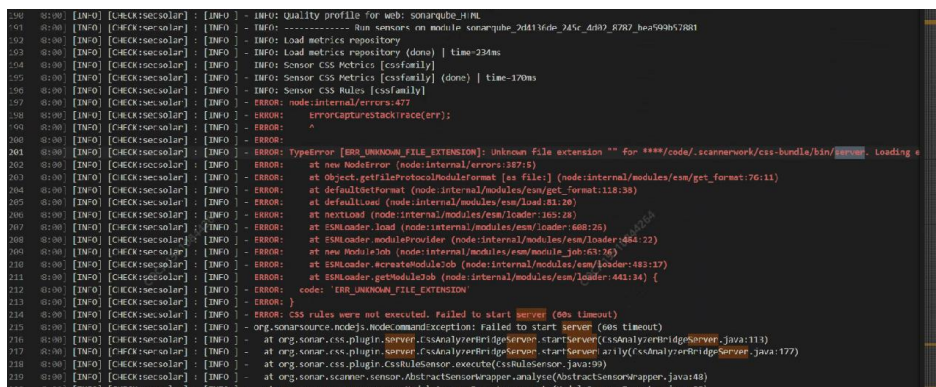
- 参考配置自定义规则集，减少规则集中使用的规则，直至该报错不再出现。
- 使用更大规格执行机重新扫描。

1.22.8 Sonarqube 检查 css 文件日志显示报错无扩展名问题

问题现象

sonarqube引擎的check阶段失败，日志显示如图1-2。

图 1-2 检查日志

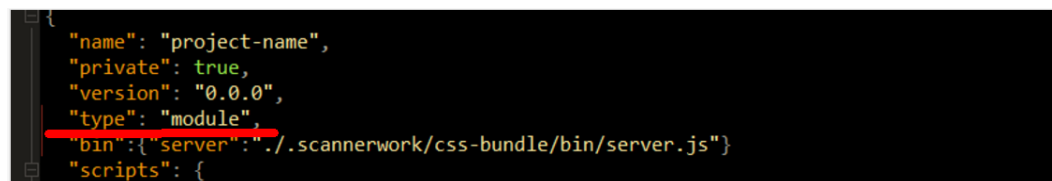


原因分析

当前检查引擎sonar版本功能异常，在package.json文件中设置了“type": "module"”，导致CSS扫描程序在导入时缺少文件扩展名。未执行CSS规则，无法启动服务器。

处理方法

- 步骤1 访问实际使用的代码仓。
- 步骤2 在“package.json”文件中删除“type": "module"”。



- 步骤3 在代码仓根目录下新建名为“sonar-project.properties”的文件。
- 步骤4 将以下代码输入至“sonar-project.properties”文件中。

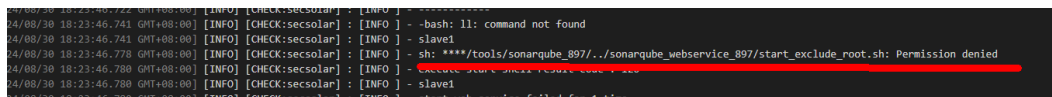
```
sonar.working.directory=projectdir
#projectdir为项目代码的相对目录，同时需要确保package.json不在projectdir目录下。
```

----结束

1.22.9 执行启动脚本无权限问题

问题现象

sonarqube引擎的check阶段失败，日志显示“permission denied”。



原因分析

执行机上，“/devcloud”目录没有“slave1”权限。

处理方法

步骤1 以root用户登录执行代码检查任务的执行机。

步骤2 在执行机上执行如下命令，增加“slave1”权限。

```
chown -R 20001:20001 /devcloud
```

----结束

1.22.10 SCC 初始化锁文件失败

问题现象

sonarqube-check失败，日志显示“scc initialize locking file failed”。

```
[2024/12/10 16:52:24.020 GMT+08:00] [INFO] [CHECK:secsolar] : [INFO] - current run analyzer: sonarqube
[2024/12/10 16:52:24.021 GMT+08:00] [INFO] [CHECK:secsolar] : [INFO] - -----> start secsolar source plugin analyze.
[2024/12/10 16:52:24.022 GMT+08:00] [INFO] [CHECK:secsolar] : [INFO] - -----> 2. start analyze.
[2024/12/10 16:52:24.025 GMT+08:00] [INFO] [CHECK:secsolar] : [INFO] - -----> 2.1 start sonarqube analyze.
[2024/12/10 16:52:24.033 GMT+08:00] [INFO] [CHECK:secsolar] : [INFO] - there are 0 jar file in dir rules
[2024/12/10 16:52:24.087 GMT+08:00] [INFO] [CHECK:secsolar] : [ERROR] - SCC initialize locking file failed!
[2024/12/10 16:52:24.109 GMT+08:00] [INFO] [CHECK:secsolar] : [ERROR] - scc component decrypt failed!
[2024/12/10 16:52:24.109 GMT+08:00] [INFO] [CHECK:secsolar] : [ERROR] - decrypt message for sql failed
[2024/12/10 16:52:24.130 GMT+08:00] [INFO] [CHECK:secsolar] : [ERROR] -
[2024/12/10 16:52:24.130 GMT+08:00] [INFO] [CHECK:secsolar] : com.huawei.secsolar.plugin.exception.AnalyzeException: CC.00130022.400 Decrypt failed.
[2024/12/10 16:52:24.130 GMT+08:00] [INFO] [CHECK:secsolar] : at com.huawei.secsolar.plugin.business.SonarQubeExecutor.startWebService(SonarQubeExecutor.java:170) ~[?:?]
[2024/12/10 16:52:24.130 GMT+08:00] [INFO] [CHECK:secsolar] : at com.huawei.secsolar.plugin.business.SonarQubeExecutor.executeAnalyze(SonarQubeExecutor.java:117) ~[?:?]
[2024/12/10 16:52:24.130 GMT+08:00] [INFO] [CHECK:secsolar] : at com.huawei.secsolar.plugin.plugin.SonarQubePlugin$SonarQubeAnalyze.executeAnalyze(SonarQubePlugin.java:67) ~[?:?]
[2024/12/10 16:52:24.131 GMT+08:00] [INFO] [CHECK:secsolar] : at com.huawei.secsolar.plugin.process.AnalyzeProcessor.analyze(AnalyzeProcessor.java:45) ~[secsolar-plugin-launcher-
[2024/12/10 16:52:24.131 GMT+08:00] [INFO] [CHECK:secsolar] : at com.huawei.secsolar.plugin.process.AnalyzeToolProcessor.startCheck(AnalyzeToolProcessor.java:95) ~[secsolar-plugi
[2024/12/10 16:52:24.131 GMT+08:00] [INFO] [CHECK:secsolar] : at com.huawei.secsolar.plugin.Boot.main(Boot.java:89) ~[secsolar-plugin-launcher-1.0.20-jar:?]
[2024/12/10 16:52:24.131 GMT+08:00] [INFO] [CHECK:secsolar] : [ERROR] - Some unexpected exception occurred!
```

原因分析

没有可用的锁。磁盘不支持lock锁，或者磁盘nfslock服务没有开启。

处理方法

步骤1 以root用户登录执行代码检查任务的执行机。

步骤2 执行以下命令，确保rpcbind服务正在运行。

```
bash
systemctl start rpcbind
systemctl enable rpcbind
```

步骤3 执行以下命令，启动nfslock服务。

```
bash
systemctl start nfs-server
systemctl enable nfs-server
```

如果系统提示nfslock服务不存在，可以尝试使用nfs-server或者nfs-kernel-server命令启动NFS服务。

步骤4 执行以下命令，检查nfslock服务是否成功启动，若未成功，需执行**步骤3**重新启动。

```
bash
systemctl status nfs-server
```

步骤5 确保“/etc/exports”文件中配置了需要共享的目录，并且nfslock服务的配置文件“/etc/sysconfig/nfs”已经正确设置。

步骤6 如果修改了NFS的配置文件，需执行以下命令，重启NFS服务，使修改生效。

```
bash
systemctl restart nfs-server
```

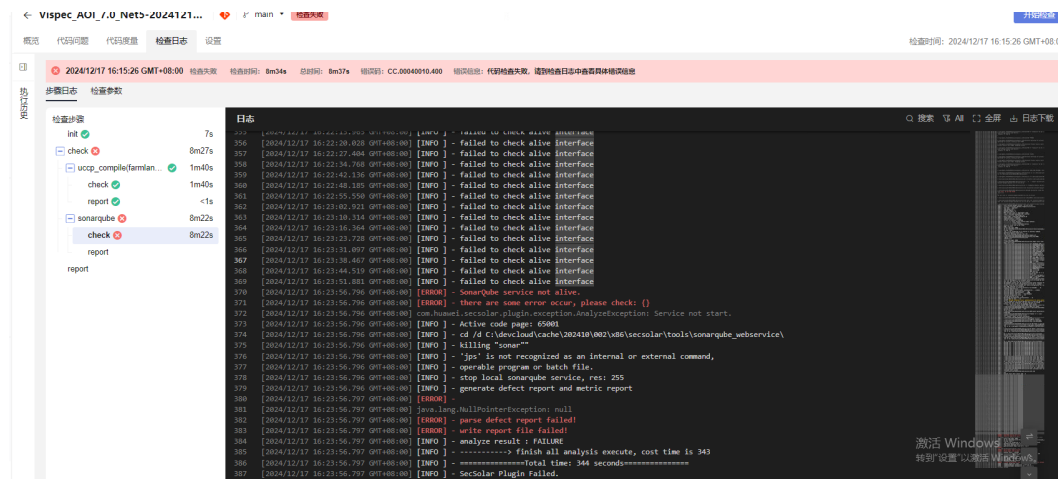
步骤7 检查防火墙设置，确保防火墙没有阻止NFS服务所需的端口。如果被阻止，需要添加相应的规则来允许NFS流量通过。

----结束

1.22.11 sonarqube 服务启动失败

问题现象

sonarqube引擎的check阶段失败，日志显示“failed to check alive interface , SonarQube service not alive”。



原因分析

可能elasticSearch初始化过程中恢复集群节点中止。

处理方法

步骤1 以root用户登录执行代码检查任务的执行机。

步骤2 通过cd命令访问实际的工具目录。

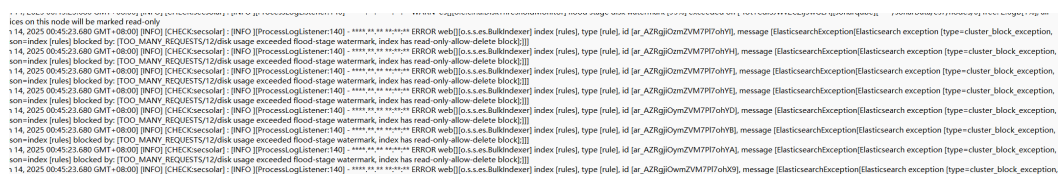
步骤3 在工具目录下删除“secsolar/tools/sonarqube_webservice97/”目录下“sonarData”，“sonarLogs”，“sonarTemp”三个文件夹。

----结束

1.22.12 sonarqube 服务的 ES 报错磁盘内存不足

问题现象

sonarqube引擎的check阶段失败，日志显示如下：



原因分析

执行机磁盘空间不足，elasticSearch无法写入临时数据。

处理方法

联系技术支持人员清理磁盘。

1.22.13 编译脚本出错，导致检查失败

问题现象

代码检查失败，日志显示：

[ERROR] : script returned exit code 127, exitMessage is:command run failed。

或 [ERROR] : script returned exit code 1, exitMessage is:command run failed。

或 [ERROR] : script returned exit code 2, exitMessage is:command run failed

原因分析

用户参数里配置的编译脚本执行失败。

在linux环境中，退出码0通常表示成功，非0表示失败。但具体到每个数字有不同含义：

退出码为1：通用错误，需要检查日志或者逻辑。

退出码为2：语法或者参数错误，需要验证脚本的输入和语法。

退出码为127：命令没找到，需要检查机器系统环境变量是否含有此命令。

处理方法

处理方法一：修改编译脚本命令。

步骤1 访问[CodeArts Check服务首页](#)。

步骤2 单击左侧导航栏“代码 > 代码检查”，进入“代码检查”页面，

步骤3 单击对应代码检查任务的名称，进入任务代码检查详情页面，单击“设置 > 规则集”。

步骤4 单击“编译配置”在弹框内核对“编译命令”是否正确。

注意：此处请填写本地已编译成功的对应编译命令。

步骤5 单击“确定”。

步骤6 单击右上角的“执行检查”重新执行检查。

----结束

处理方法二：如果确定脚本无误，请[提交工单](#)联系技术支持工程师，检查执行机上的构建环境。

原因分析

- 原因一：执行代码检查任务的子模块代码仓无权限。
- 原因二：执行代码检查任务的子模块代码仓不存在。

处理方法

原因一处理方法：

参考[配置代码仓库级的权限](#)，配置代码仓权限。

原因二处理方法：

- 步骤1 在页面导航栏中选择“代码 > 代码托管”。
 - 步骤2 单击检查的代码仓名称，进入代码仓详情页。
 - 步骤3 删除不存在子模块的文件。
- 结束

1.23 使用代码检查服务时报错：CC.10010295.403 用户已被禁用

问题现象

使用代码检查服务时，提示：用户已被禁用，请联系租户管理员确认用户状态。

原因分析

用户已被禁用。

处理方法


- 步骤1 租户管理员在导航栏中单击用户名，选择“租户设置”。
- 步骤2 在导航中依次选择“通用设置 > 成员管理”，进入成员管理页面。
- 步骤3 选择没权限的用户，根据实际需要启用该用户。

图 1-3 管理成员



步骤4 如果仍然未能解决，请[提交工单](#)联系技术支持工程师。

----结束

2 API 类问题

2.1 使用公共 API 时提示没有权限

问题现象

使用公共API报没有权限。

原因分析

- 登录的用户没有权限。
- Region信息不对。

处理方法

步骤1 确认登录的用户是否有权限，详情见[授权成员使用代码检查服务](#)。

步骤2 确认Region信息是否正确。

如果仍然未能解决，请[提交工单](#)联系技术支持工程师。

----结束

2.2 使用公用 API 时提示项目不存在

问题现象

使用公用API提示项目不存在。

原因分析

taskId错误导致。

处理方法

使用正确的taskId。如果仍然未能解决，请[提交工单](#)联系技术支持工程师。