**CA Filter**

**使用手册**

目录

[1. 背景 1](#_Toc427398172)

[2. 简介 1](#_Toc427398173)

[3. 使用方法 2](#_Toc427398174)

[3.1 根据标准CA告警文件过滤 2](#_Toc427398175)

[3.2 根据保存的结果过滤 3](#_Toc427398176)

[3.3 忽略特定文件的CA告警 4](#_Toc427398177)

[3.4 使用MCAnalyzer进行过滤 4](#_Toc427398178)

[4. 工作原理 5](#_Toc427398179)

# 背景

由于我们的SVTI项目原来的代码有大量的CA告警，对代码CA的时候我们自己的CA告警就淹没其中不太好找；另一方面，由于持续集成环境的需要每次对整个项目的代码进行CA，海量的CA告警会让上传者找不到自己产生的CA告警在哪里。于是写了这个小工具，如果有什么BUG及时反馈哦。

# 简介

CA\_Filter是一个CA内容过滤工具。CA\_Filter工作前要对一个标准的CA告警文件进行学习，然后对要过滤的CA告警文件进行过滤，如果某条CA告警是之前在学习阶段遇到的，就把这条CA告警过滤掉。

CA\_Filter可以嵌入到Source Insight中，CA\_Filter.exe的参数和CAnalyzer.bat完全一样，其他设置也一样。CA\_Filter可以保存和读入学习结果避免每次CA的时候都要进行重复的学习。CA\_Filter也可以进行MCA。

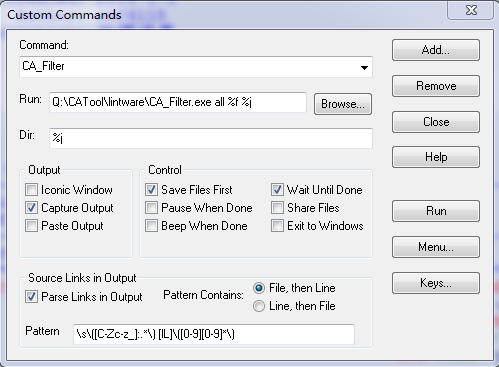


图 2.1 Source中嵌入CA Filter

# 使用方法

## 根据标准CA告警文件过滤

操作步骤：

（1）拷贝CA\_Filter.exe到CATool\lintware目录下

（2）在CATool\lintware目录下新建CA\_Filter\_Study.txt，用记事本打开，直接复制某次CA的输出，或者把希望被过滤的CA告警复制到其中。

（3）CA过滤时，在Source的“Option”---“Custom Commands”中参照图 2.1的设置，然后点“Run”即可。

在编辑CA\_Filter\_Study.txt时要注意以下几点：

1. 在编辑时最好使用Windows自带的记事本而不是Notepad++编辑。这是因为未过滤的原CA告警文件采用的是“windows的”ANSI编码。
2. 复制CA告警到这个文件时，确保一个字符都不能少或多（空行除外），这是因为CA\_Filter会计算告警信息的MD5，如果发生变化则认为是不同的。
3. 每个CA告警之间至少要有一个空行，每个CA告警和其他信息之间也最好用空行隔开。注意某些行看似空行但其实有一个下划线“\_”。
4. 第一行不要存放CA告警。建议第一行类似普通CA结果一样：

--- 10:55:47.01 ALL CAnalyzer X:\ XXX\XXXX\XXXXXX

完成过滤之后，如果希望查看原版的未经过滤的CA告警，打开CATool\lintware\CA\_Filter\_Result\_Origin.txt即可。

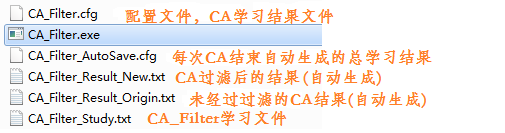


图 3.1 CA Filter目录结构

## 根据保存的结果过滤

CA\_Filter每次CA完都会将本次学习的结果保存到CA\_Filter\_AutoSave.cfg（包括读取的CA\_Filter.cfg）。但进行过滤工作时并不使用这个文件，而是CA\_Filter.cfg（该文件不存在则不读取）。**把CA\_Filter\_AutoSave.cfg重命名为CA\_Filter.cfg，这样CA过滤的时候就会根据保存的结果进行过滤**。

CA\_Filter.cfg的内容可以直接编辑，由两部分组成，第一部分是要忽略CA的项目文件（见3.3节），第二部分是每条CA告警的MD5。

CA\_Filter工作流程如图3.2所示，CA\_Filter在每次过滤前，会先读取CA\_Filter.cfg（如果不存在则跳过步骤①），然后对CA\_Filter\_Study.txt进行学习（如果不存在则跳过步骤②）。CA\_Filter\_Study.txt是标准的CA告警文件，用户可以往里面添加要过滤掉的CA告警内容。

图 3.2 CA Filter的工作流程

①读取CA\_Filter.cfg中保存的学习结果

②读取CA\_Filter\_Study.txt进行学习

③调用CA脚本然后过滤

举个CA\_Filter\_Study.txt + CA\_Filter.cfg 结合使用的例子。其中CA\_Filter.cfg存放项目原来存在的大量CA告警的学习结果，CA\_Filter\_Study用来存放我们自己产生的又要忽略的CA告警（方便经常性地修改）：

1. 新建CA\_Filter\_Study.txt，将项目原来存在的大量CA告警拷贝到其中。删除CA\_Filter.cfg，运行CA\_Filter.exe，得到CA\_Filter\_AutoSave.cfg
2. 将CA\_Filter\_AutoSave.cfg重命名为CA\_Filter.cfg
3. 删除然后新建CA\_Filter\_Study.txt，添加我们自己产生的CA告警到里面

## 忽略特定文件的CA告警

有时候我们想过滤某个文件的所有CA告警，则可以在CA\_Filter.cfg中添加希望要忽略CA告警的文件名（不带路径名），每个文件名写一行，且以“--”开头，如图 3.3所示：

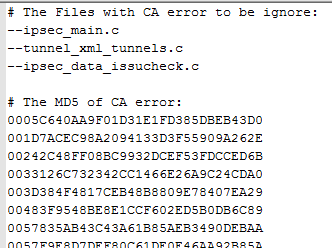


图 3.3 CA\_Filter.cfg内容

## 使用MCAnalyzer进行过滤

将CA\_Filter.exe复制一份重命名为MCA\_Filter.exe即可用来进行MCA，Source Insight中的设置与MCAnalyzer一致，如图 3.4所示。

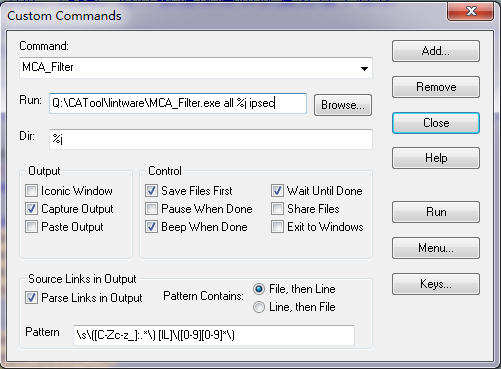


图 3.4 MCA在Source Insight中的配置

# 工作原理

CA\_Filter识别某条CA告警是通过对CA告警的特征进行MD5运算来实现的，**每条CA告警用一个MD5字符串来唯一标识**。若两个CA告警的特征相同，则他们的MD5也相同，反之亦然。

而一个CA告警的特征由以下四部分组成：

1. 引起CA告警的那一行代码
2. 引起CA告警的文件名
3. CA告警类型和告警ID
4. CA告警错误消息

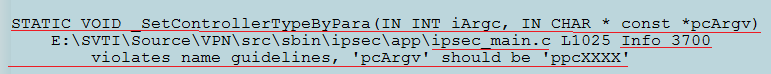


图 4.1 CA告警的特征提取

如果两个CA告警的以上四点完全一致（每个字符都相同），那么CA\_Filter则认为他们是两个相同的CA告警。

CA\_Filter简要工作流程如下：

1. 尝试读取配置文件（里面含有学习后的MD5列表）
2. 尝试读取一个标准的CA告警文件学习，里面存放希望被过滤的CA告警：
3. 把每条告警按段分隔开来
4. 计算每条CA告警的特征的MD5（若告警本身含有MD5则用本身的）
5. 将所有的MD5保存成”MD5列表”到内存中
6. 调用CAnalyzer.bat脚本进行CA，得到原CA告警信息，并保存成文件。
7. 读取原CA告警信息的文件，查询保存的MD5列表进行过滤：
8. 把每条告警按段分隔开来
9. 读取一段CA告警，计算其特征的MD5（若告警本身含有MD5则用本身的）
10. 如果这段CA告警的MD5在”MD5列表”中找不到，则说明这条告警是新产生的，则把这段告警写到新的文件中。如果找到了则不写入。
11. 不断地重复步骤b和c知道原CA告警文件读取结束