

KYLIN-5398 火焰图问题修复

一、节点所在机器 tmp 目录挂载的盘属性为 noexec ，火焰图功能无法使用

描述

若 kylin5 所在机器 tmp 目录挂载的盘属性为 noexec ，则火焰图功能无法使用

背景

在 Linux 系统中，至少有两个目录保存着系统的临时文件，一个就是 /tmp，另外一个为 /var/tmp。这两个目录有一个共同点就是所有用户在该目录下拥有可读写，可执行的权限。

问题：因为两个目录权限的问题，攻击者可以把病毒或者木马文件放到这些临时目录下，用于信息的收集或者伪装运行系统的程序而实际上运行自己的程序。

由此引发的问题：如果去修改临时目录的读写权限，则会影响系统上应用程序的正常运行。比如在挂载 /tmp 对应的盘时，增加了 noexec 属性，即便有 777 文件权限，应用也无法正常读取一些文件执行。

如何修改属性

以 /tmp 目录为例修改完成后不会再出现安全问题，但可能会导致应用在读取 /tmp 文件执行时失败

- 通过文件修改
- /etc/fstab 是用来存放文件系统的静态信息的文件。当系统启动的时候，系统会自动地从这个文件读取信息，并且会自动将此文件中指定的文件系统挂载到指定的目录。所以可以通过修改该文件进行更改，见下图，但只有系统重启才会显示，除非重新挂载：`mount /tmp -o remount`

```
#
# /etc/fstab
# Created by anaconda on Thu Sep 15 03:50:46 2022
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
/dev/mapper/centos-root / xfs defaults 0 0
UUID=b5d0376c-c7c0-47b3-8d80-54ab1de5f3c9 /boot xfs defaults 0 0
/dev/mapper/centos-home /home xfs defaults 0 0
/dev/mapper/centos-tmp /tmp xfs rw,nosuid,nodev,noexec 0 0
/dev/mapper/centos-swap swap swap defaults 0 0
```

- 直接通过命令更改，并重新挂载：`mount /tmp -o remount,noexec`

/etc/fstab 文件内容总共包含 6 列。

第一列：Device：磁盘设备文件或者该设备的 Label、UUID

第二列：Mount point：设备的挂载点，就是你要挂载到哪个目录下。

第三列：filesystem：磁盘文件系统的格式，包括 ext2、ext3、reiserfs、nfs、vfat 等。可以使用 `df -T` 查看

第四列：parameters：文件系统的参数

- async/sync 设置是否为同步方式运行，默认为 async（性能较佳）

- auto/noauto 当执行 `mount -a` 的命令时，此文件系统是否被主动挂载。默认为 auto

- rw/ro 是否以只读或者读写模式挂载

- exec/noexec 限制此文件系统内是否能够运行可执行文件。

- user/nouser 是否允许用户使用 `mount` 命令挂载

- suid/nosuid 是否允许 SUID 的存在

- usrquota 启动文件系统支持磁盘配额模式。涉及到磁盘配额的知识，有兴趣可以自行研究

- grpquota 启动文件系统对群组磁盘配额模式的支持

- defaults 同时具有 rw,suid,dev,exec,auto,nouser,async 等参数。基本上，默认使用 defaults 设置即可。

第五列：能否被 dump 备份命令作用：dump 是一个用来作为备份的命令。通常值为 0 或者 1。测试环境很少用。

第六列：是否检验扇区：开机的过程中，系统默认会以 fsck 检验我们系统是否为完整（clean）。一般来说，根目录设置为 1，其他的文件系统设置为 2。以前经常会在 IBM 的 AIX 系统遇到该问题。

如何确认

举例修改的是 noexec 属性，这里进行确认

- 先确认 /tmp 挂载盘：`df -h /tmp`

```
[root@localhost ~]# df -h /tmp/
Filesystem                Size      Used Avail Use% Mounted on
/dev/mapper/centos-tmp    25G       52M    25G   1% /tmp
```

- 直接查看目录属性或者挂载盘的属性：`mount -l | grep /tmp` 或 `mount -l | grep /dev/mapper/centos-tmp`

```
[root@localhost ~]# mount -l | grep /tmp
/dev/mapper/centos-tmp on /tmp type xfs (rw,nosuid,nodev,noexec,relatime,seclabel,attr2,inode64,noquota)
[root@localhost ~]# mount -l | grep /dev/mapper/centos-tmp
/dev/mapper/centos-tmp on /tmp type xfs (rw,nosuid,nodev,noexec,relatime,seclabel,attr2,inode64,noquota)
```

Workaround

- 通过设置额外的 tmp 目录方式来绕过此问题

```
1 //java -Djava.io.tmpdir=/path/to/tmpdir
2 -Djava.io.tmpdir=/home/guoliang/tsTmp
```

- 详见
 - 查询可修改 `conf/setenv.sh`，在 `KYLIN_JVM_SETTINGS` 后追加 `-Djava.io.tmpdir=/path/to/tmpdir`，因为 kylin5 和 Sparder Driver 在一个进程中，所以需要重启 kylin5
 - 构建则是在项目级参数中增加配置项：`kylin.engine.spark-conf.spark.driver.extraJavaOptions=-Djava.io.tmpdir=/home/guoliang/tsTmp`

Fix Design

变更 native 库文件加载路径，由原先的 /tmp 目录改为自适应，避免 noexec 属性导致包加载失败。

调整 native 包的名字，并更新原先的旧版本，包的后缀代表了不同机器所需加载的库，分别为：

- libasyncProfiler-mac.so
- libasyncProfiler-linux-x64.so
- libasyncProfiler-linux-arm64.so

出包时，所需的 native 包会置于 `{KYLIN_HOME}/lib` 目录下，这里 Sparder（查询引擎）和构建引擎会从不同的目录加载，具体情况取决于是否为 Spark Driver 节点以及部署模式等，即：

- Sparder 和 kylin5 在一个 JVM 中，所以 Driver 端会从 `{KYLIN_HOME}/lib` 加载
- 构建引擎的 Driver 取决于其部署模式，即 `spark.submit.deployMode` 参数值
 - client 模式时从 `{KYLIN_HOME}/lib` 加载
 - cluster 模式时从 `Container` 加载
- Executor 始终从 `Container` 加载

Test

测试考虑覆盖的点：

- 准备一个 /tmp 目录单独挂载盘的 Linux，将其属性调整为 noexec，验证查询火焰图
 - 构建引擎配置为 YARN-client 模式，验证构建火焰图
 - 构建引擎配置为 YARN-cluster 模式，验证构建火焰图
- 在 /tmp 目录未单独挂载的情况下，验证查询火焰图
 - 构建引擎配置为 YARN-client 模式，验证构建火焰图
 - 构建引擎配置为 YARN-cluster 模式，验证构建火焰图

二、查询火焰图使用的临时目录被误删后，无法打火焰图

描述

查询火焰图使用的临时目录被误删(人为、系统)后，无法打火焰图。

```
2022-07-27T17:21:53,469 DEBUG [http-nio-7070-exec-408] interceptor.ProjectInfoParser : Parsed project_global from request /kylin/api/query/profile/start
2022-07-27T17:21:53,471 ERROR [http-nio-7070-exec-408] asyncprofiler.AsyncProfiling : error clean cache directory
java.lang.IllegalArgumentException: /tmp/ke-async-profiler-result-4554322392257042165 does not exist
    at org.apache.commons.io.FileUtils.cleanDirectory(FileUtils.java:1637) ~[commons-io-2.4.jar!/:2.4]
    at io.kyligence.kap.query.asyncprofiler.AsyncProfiling$.cleanLocalCache(AsyncProfiling.scala:165) ~[kap-sparder-4.0.0-SNAPSHOT.jar!/:?]
    at io.kyligence.kap.query.asyncprofiler.AsyncProfiling$.start(AsyncProfiling.scala:67) ~[kap-sparder-4.0.0-SNAPSHOT.jar!/:?]
    at io.kyligence.kap.query.asyncprofiler.AsyncProfiling.start(AsyncProfiling.scala) ~[kap-sparder-4.0.0-SNAPSHOT.jar!/:?]
    at io.kyligence.kap.rest.controller.NQueryController.profile(NQueryController.java:156) ~[kap-server-base-4.0.0-SNAPSHOT.jar!/:?]
    at io.kyligence.kap.rest.controller.NQueryController$$FastClassBySpringCGLIB$$ee1baf11.invoke(<generated>) ~[kap-server-base-4.0.0-SNAPSHOT.jar!/:?]
    at org.springframework.cglib.proxy.MethodProxy.invoke(MethodProxy.java:204) ~[spring-core-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.aop.framework.CglibAopProxy$CglibMethodInvocation.invokeJoinpoint(CglibAopProxy.java:737) ~[spring-aop-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:157) ~[spring-aop-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.aop.aspectj.MethodInvocationProceedingJoinPoint.proceed(MethodInvocationProceedingJoinPoint.java:96) ~[spring-aop-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at io.kyligence.kap.rest.aspect.InsensitiveNameAspect.around(InsensitiveNameAspect.java:71) ~[kap-server-base-4.0.0-SNAPSHOT.jar!/:?]
    at sun.reflect.GeneratedMethodAccessor337.invoke(Unknown Source) ~[?:?]
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43) ~[?:1.8.0_201]
    at java.lang.reflect.Method.invoke(Method.java:498) ~[?:1.8.0_201]
    at org.springframework.aop.aspectj.AbstractAspectJAdvice.invokeAdviceMethodWithGivenArgs(AbstractAspectJAdvice.java:627) ~[spring-aop-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.aop.aspectj.AbstractAspectJAdvice.invokeAdviceMethod(AbstractAspectJAdvice.java:616) ~[spring-aop-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.aop.aspectj.AspectJAroundAdvice.invoke(AspectJAroundAdvice.java:70) ~[spring-aop-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:168) ~[spring-aop-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.aop.interceptor.ExposeInvocationInterceptor.invoke(ExposeInvocationInterceptor.java:92) ~[spring-aop-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:179) ~[spring-aop-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.aop.framework.CglibAopProxy$DynamicAdvisedInterceptor.intercept(CglibAopProxy.java:672) ~[spring-aop-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at io.kyligence.kap.rest.controller.NQueryController$$EnhancerBySpringCGLIB$$384cia20.profile(<generated>) ~[kap-server-base-4.0.0-SNAPSHOT.jar!/:?]
    at sun.reflect.GeneratedMethodAccessor3806.invoke(Unknown Source) ~[?:?]
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43) ~[?:1.8.0_201]
    at java.lang.reflect.Method.invoke(Method.java:498) ~[?:1.8.0_201]
    at org.springframework.web.method.support.InvocableHandlerMethod.doInvoke(InvocableHandlerMethod.java:204) ~[spring-web-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.web.method.support.InvocableHandlerMethod.invokeForRequest(InvocableHandlerMethod.java:132) ~[spring-web-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.web.servlet.mvc.method.annotation.ServletInvocableHandlerMethod.invokeAndHandle(ServletInvocableHandlerMethod.java:97) ~[spring-webmvc-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.invokeHandlerMethod(RequestMappingHandlerAdapter.java:854) ~[spring-webmvc-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.handleInternal(RequestMappingHandlerAdapter.java:765) ~[spring-webmvc-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.web.servlet.mvc.method.AbstractHandlerMethodAdapter.handle(AbstractHandlerMethodAdapter.java:85) ~[spring-webmvc-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.web.servlet.DispatcherServlet.doDispatch(DispatcherServlet.java:967) ~[spring-webmvc-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.web.servlet.DispatcherServlet.doService(DispatcherServlet.java:901) ~[spring-webmvc-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
    at org.springframework.web.servlet.FrameworkServlet.processRequest(FrameworkServlet.java:970) ~[spring-webmvc-4.3.30.RELEASE.jar!/:4.3.30.RELEASE]
```

调用 **start接口**，会报上图的异常。

调用 **dump接口**，下载下来的压缩包是空的。

手动创建临时目录即可恢复正常查询火焰图使用。

Root Cause

```
class BuildAsyncProfilerDriverPlugin extends DriverPlugin with Logging {  
  private val checkingInterval: Long = 1000  
  private val localCacheDir = Files.createTempDirectory( prefix = "ke-build-async-profiler-result-").toFile  
  localCacheDir.deleteOnExit()  
  private val resultCollectionTimeout = sys.props.get("spark.profiler.collection.timeout").getOrElse("60000").toLong  
  private val profilingTimeout = sys.props.get("spark.profiler.profilng.timeout").getOrElse("300000").toLong  
  private var timeoutExecutionThread: Thread = _  
}
```

一个kylin5实例只会使用一个固定的临时目录

临时目录被删后，后续所有查询火焰图都不能打了。

Fix Design

当开始执行收集火焰图时，判断临时目录是否存在，不存在的话会重新创建一个临时目录，并将该目录作为火焰图收集生成结果的目录，其他行为保持不变。