0. 引言
1. 需求
2. 问题
3. 解决方法
3.1 使用systemd控制用户资源
3.1.1 实例
3.1.2 控制参数
3.1.3 删除配置
3.1.4 方法的局限性
3.1.5 cgroup v2支持情况
3.2 直接安装rhel7的libcgroup包
3.2.1 降版本
3.2.2 方法的局限性
3.2.3 cgroup v2支持情况
3.3 使用libcgroup控制用户资源
3.3.1 libcgroup项目概况
3.3.2 编译libcgroup
3.3.3 制作rpm安装包
3.3.4 制作rpm过程种遇到的问题
3.3.5 方法的局限性
3.3.6 cgroup v2支持情况
4. 背景
4.1 libcgroup主要文件的说明
4.2 cgred的工作原理
4.3 海盒数据库SeaboxSQL的资源管理
5. 参考文献

0. 引言

在同一套物理主机环境中，如果部署多套数据库软件，常常面临资源隔离/限制需求，按照`Linux`用户来隔离资源是一种常见的解决方案。然而，我们发现网上搜索到的资料绝大部分过于陈旧，仅适用于`Redhat/CentOS Linux 7`等比较旧的操作系统，当我们在`Redhat/CentOS Linux 8`系统上部署时，方法根本不可用。本文试图探索其中的原因，给出多种解决方法。虽然本文以`SeaboxSQL`数据库为例探讨多套应用程序的资源隔离问题，但是，其方法是普适性的。方法在`Redhat 8.5`上验证通过，同样适用于`CentOS 8`。

1. 需求

在`Redhat Linux 8`系统上，使用`cgroup`限制指定用户的资源使用量。

例如，假设有这样一个需求，在一台物理服务器上，安装多套`SeaboxSQL`数据库，因为业务优先级不同，期望不同数据库允许使用的资源数量不同，譬如，高优先级的`SeaboxSQL`数据库允许使用更多的`CPU`资源，而低优先级的`SeaboxSQL`数据库仅允许使用很少量的`CPU`资源。

通常，我们可能会想到使用`Docker`或其它虚拟化技术进行资源隔离，但是虚拟化或多或少都有一些性能损失[1]。因此，更明智的方法是使用多个用户安装`SeaboxSQL`数据库，按用户来隔离资源。通过搜索，有很多关于使用`cgroup`限制`Linux`用户资源的文章，笔者为此整理成参考文献[2]。

本文假定您已经了解：

- `cgroup`基础

- `libcgroup`和`libcgroup-tools`工具使用方法
- `Redhat/CentOS Linux 7`使用`cgroup`限制用户资源的方法

本文所需的物料可从以下项目获取[3]：

https://github.com/fairyfar/cgroup-cn

2. 问题

不幸的是，当你在`Redhat Linux 8`或者其它更新的`Linux`发行版（例如统信`UOS V20`）上使用该方法时，却遇到了致命问题：关键服务`cgred`（对应`cgrulesengd`应用）已经被踢出`libcgroup-tools`包了。例如，在`Redhat Linux 8.5`上，即使已经安装了`libcgroup`和`libcgroup-tools`包的情况下，`cgred`服务仍然不存在：

```
[root@bogon ~]# yum list installed | grep libcgroup
libcgroup.x86_64                                0.41-19.el8
@base
libcgroup-tools.x86_64                          0.41-19.el8
@base

[root@bogon ~]# systemctl status cgred
Unit cgred.service could not be found.
```

查阅`libcgroup`的变更日志，有以下记录：

```
[root@bogon ~]# rpm -q --changelog libcgroup
* Tue Jan 14 2014 Peter Schiffer <pschiffe@redhat.com> 0.41-1
- resolves: #966008
  updated to 0.41
- removed deprecated cgred service
  please use Control Group Interface in Systemd instead
```

关于`cgred`被移除的讨论，详见参考文献[4]。

3. 解决方法

本文提供三种方法，但是每种方法都有其局限性，请权衡使用。

3.1 使用systemd控制用户资源

`Redhat Linux 8`从`libcgroup`包移除`cgred`服务，变更日志提到：

> please use Control Group Interface in Systemd instead

看来官方是希望用`systemd`来替代`cgred`，因此，首先想到使用`systemd`提供的接口来控制用户资源。以下通过一个具体例子来演示如何用`systemd`控制用户资源。

3.1.1 实例

假设，欲限制`Linux`用户`seabox`的CPU使用上限为`30%`（即，单个CPU核的`30%`）。步骤如下：

(1). 获取用户`UID`

```
[root@bogon ~]# id seabox
uid=1002(seabox) gid=1002(fairyfar) groups=1002(seabox)
```

(2). 设置配额

```
[root@bogon ~]# systemctl set-property user-1002.slice CPUQuota=30%
```

如果命令报以下错误：

```
[root@bogon ~]# systemctl set-property user-1002.slice CPUQuota=30%
Failed to set unit properties on user-1002.slice: Unit user-1000.slice is not
loaded.
```

则，需要在`set-property`之前先执行`start`命令：

```
[root@bogon ~]# systemctl start user-1002.slice
```

(3). 查看配置情况

```
[root@bogon ~]# systemctl cat user-1002.slice

# /etc/systemd/system/user-1002.slice.d/50-CPUQuota.conf
[Slice]
CPUQuota=30%
```

(4). 使配置立即生效

```
[root@bogon ~]# systemctl daemon-reload
```

3.1.2 控制参数

上述实例中，`systemctl set-property`命令中使用了参数`CPUQuota`，表示CPU使用率上限，`30%`表示单个CPU核的`30%`使用率。`systemctl`支持很多种参数，用于控制不同类型资源。详见参考文献[5]。

需要特别注意的是，`systemctl`支持的参数种类与`systemd`版本有关，查看`systemd`版本方法：

```
[root@bogon ~]# systemctl --version
systemd 239 (239-51.el8)
```

例如，`Redhat Linux 8.5`默认的`systemd`版本为`239`，从参考文献[5]可以查询到：

参数	作用	支持参数的systemd最低版本
CPUQuota	配置CPU使用率上限	213
AllowedCPUs	绑定CPU核（cpuset）	244

所以，很遗憾，`Redhat Linux 8.5`不支持通过`systemctl`绑定CPU核功能。

3.1.3 删除配置

在“实例”小节中，演示了配置资源限制，那么如何删除限制呢？有两种方法：

如果`systemd`版本大于等于229，则可以使用`systemctl revert`命令：

```
[root@bogon ~]# systemctl revert user-1002.slice
```

否则，低版本需要手动删除：

```
[root@bogon ~]# rm -rf /etc/systemd/system/user-1002.slice.d/
```

删除的路径在`systemctl cat`结果中可以找到，当然也可以有选择性地删除部分配置项。

3.1.4 方法的局限性

主要有两方面局限。

(1). 支持的参数类型受限于`systemd`版本。

`Redhat Linux 8`默认的`systemd`版本支持的参数类型很有限，一些常用的资源控制尚未支持。理所当然，笔者认为：

`Redhat Linux 8`从`libcgroup`中移除`cgred`，似乎操之过急。

(2). 用户登录方式影响`systemd`控制资源的有效性。

本方法可以控制用户通过`GDM`（`The GNOME Display Manager`）或`ssh`登录时的资源使用，但是不适用于`su`方式切换登录的用户。例如，以下方式登录的用户，不受上述`systemd`配置的资源限制。

```
[root@bogon ~]# whoami
root
[root@bogon ~]# su - seabox
# su方式登录的用户，资源不受限制。
[seabox@bogon ~]$ do someting
```

3.1.5 cgroup v2支持情况

本方法支持`cgroup v2`。

3.2 直接安装rhel7的libcgroup包

3.2.1 降版本

从`Redhat 7`获取`rpm`包，然后安装到`Redhat 8`。

(1). 从`Redhat 7`获取`rpm`包

在`Redhat 7`上，执行以下`yum`命令，只下载包但不安装：

```
[root@bogon ~]# yum install --downloadonly --downloaddir=/tmp/ libcgroup
[root@bogon ~]# yum install --downloadonly --downloaddir=/tmp/ libcgroup-tools
```

为便于下载，笔者已经将两个`rpm`包上传至`GitHub`，下载地址：

- libcgroup：https://github.com/fairyfar/cgroup-cn/raw/refs/heads/main/archive/libcgroup-0.41-21.el7.x86_64.rpm
- libcgroup-tools：https://github.com/fairyfar/cgroup-cn/raw/refs/heads/main/archive/libcgroup-tools-0.41-21.el7.x86_64.rpm

(2). 卸载 `Redhat 8` 默认的 `libcgroup`

```
[root@bogon ~]# yum remove libcgroup-tools
[root@bogon ~]# yum remove libcgroup
```

(3). 在 `Redhat 8` 上安装 `rpm` 包

```
[root@bogon ~]# rpm -i ./libcgroup-0.41-21.el7.x86_64.rpm
[root@bogon ~]# rpm -i ./libcgroup-tools-0.41-21.el7.x86_64.rpm
```

3.2.2 方法的局限性

本方法在下列系统上验证通过：

- `Redhat/CentOS Linux 8.5`
- `UnionTech OS Server release 20 (kongzi)`
- `openEuler release 20.03 (LTS-SP3)`

其它版本 `Linux` 是否可行，有待验证。

3.2.3 cgroup v2支持情况

本方法不支持 `cgroup v2`，因为 `Redhat Linux 7` 的 `libcgroup` 版本为 `v0.41`，尚未支持 `cgroup v2`。如果启用了 `cgroup v2`，则启动 `cgred` 服务报错：

```
[root@bogon ~]# systemctl start cgred
Job for cgred.service failed because the control process exited with error code.
See "systemctl status cgred.service" and "journalctl -xe" for details.

[root@bogon ~]# systemctl status cgred
● cgred.service - CGroups Rules Engine Daemon
   Loaded: loaded (/usr/lib/systemd/system/cgred.service; disabled; vendor preset: disabled)
   Active: failed (Result: exit-code) since Fri 2025-01-08 04:55:33 EST; 2min 52s ago
  Process: 5534 ExecStart=/usr/sbin/cgrulesengd $OPTIONS (code=exited, status=81)

Jan 8 04:55:33 bogon systemd[1]: Starting CGroups Rules Engine Daemon...
Jan 8 04:55:33 bogon cgrulesengd[5534]: Error: libcgroup initialization failed, Cgroup is not mounted
Jan 8 04:55:33 bogon systemd[1]: cgred.service: Control process exited, code=exited status=81
Jan 8 04:55:33 bogon systemd[1]: cgred.service: Failed with result 'exit-code'.
Jan 8 04:55:33 bogon systemd[1]: Failed to start CGroups Rules Engine Daemon.
```

3.3 使用libcgroup控制用户资源

3.3.1 libcgroup项目概况

`libcgroup`是一个开源项目，提供一系列`cgroup`应用工具、系统服务，以及开发库。该项目源码最早托管在`SourceForge`：

https://sourceforge.net/projects/libcg/

现在已迁移至`GitHub`：

https://github.com/libcgroup/libcgroup

从`libcgroup v2.0`版本开始，全面支持`cgroup v2`。

请注意：

- `libcgroup`项目一直在维护`cgrulesengd`（`cgred`服务），移除`cgred`是`Linux`发行版的分叉行为。
- 大部分`Linux`发行版提供`libcgroup`和`libcgroup-tools`两个安装包，但是，这两个包中的应用程序和库均源自`libcgroup`项目，只是按功能拆成了两个安装包（实际上还有个`libcgroup-pam`包），`libcgroup`安装包主要提供`.so`库文件，而`libcgroup-tools`安装包提供应用程序和系统服务。

3.3.2 编译libcgroup

从`libcgroup`项目下载源代码，自行编译`libcgroup`，以`Redhat Linux 8.5`环境为例。

(1). 下载源码，然后解压缩。

```
[root@bogon ~]# wget
https://github.com/libcgroup/libcgroup/releases/download/v0.41/libcgroup-
0.41.tar.bz2
```

(2). 准备编译环境

安装依赖库，并创建安装路径：

```
[root@bogon ~]# yum install pam-devel
[root@bogon ~]# mkdir /root/libcgroup
```

(3). 编译

```
[root@bogon ~]# cd libcgroup-0.41
[root@bogon libcgroup-0.41]# ./configure --prefix=/root/libcgroup
[root@bogon libcgroup-0.41]# make
[root@bogon libcgroup-0.41]# make install
```

编译和安装成功后，可以在`/root/libcgroup`目录下查看编译结果。

3.3.3 制作rpm安装包

上一小节，我们自行编译，得到了需要的`cgrulesengd`应用程序，其实现在我们可以仿照`Redhat 7`中的`libcgroup-tools`手动创建`cgred`服务。但是，这样不便于维护。如果把编译好的文件制作成rpm安装包，那么就可以一键安装好全部工具和服务。

下面以`libcgroup-tools`为例演示`rpm`制作过程。制作`libcgroup`的`rpm`方法与此类似，当然，如果你乐意，也可以将两个`rpm`二合一。

(1). 准备环境

安装`rpm-build`（有的发行版`Linux`中名称为`rpmbuild`）和`rpmdevtools`：

```
[root@bogon ~]# yum install rpm-build
[root@bogon ~]# yum install rpmdevtools
```

(2). 创建`rpmbuild`目录结构

使用`rpmdev-setuptree`工具创建目录。当然，`rpmbuild`目录及其子目录也可以手动`mkdir`创建。默认情况下，目录位置和名称都是确定的，没有必要刻意去修改。

```
[root@bogon ~]# rpmdev-setuptree
[root@bogon ~]# ll rpmbuild/
total 0
drwxr-xr-x 2 root root 6 Jan  8 04:21 BUILD
drwxr-xr-x 2 root root 6 Jan  8 04:21 RPMS
drwxr-xr-x 2 root root 6 Jan  8 04:21 SOURCES
drwxr-xr-x 2 root root 6 Jan  8 04:21 SPECS
drwxr-xr-x 2 root root 6 Jan  8 04:21 SRPMS
```

`rpmbuild/SPECS`目录中创建并编写一个`libcgroup-tools-0.41-19.el8.x86_64.spec`文件（名字可自选），`.spec`是打包控制文件，内容如下：

```
Name:           libcgroup-tools
Version:        0.41
Release:        19.el8
Summary:        libcgroup tools package.

Group:          Applications/System
License:        GPL
URL:            www.seaboxdata.com

%description
cgroup tools rpm package.

%prep
%build
%install
mkdir -p $RPM_BUILD_ROOT/usr/
mkdir -p $RPM_BUILD_ROOT/usr/bin/
mkdir -p $RPM_BUILD_ROOT/usr/lib/systemd/system/
mkdir -p $RPM_BUILD_ROOT/usr/sbin

cp -r ../BUILD/etc/ $RPM_BUILD_ROOT/etc/
mkdir -p $RPM_BUILD_ROOT/etc/cgrules.d
cp -r ../BUILD/usr/bin/cgclassify $RPM_BUILD_ROOT/usr/bin/
cp -r ../BUILD/usr/bin/cgcreate $RPM_BUILD_ROOT/usr/bin/
cp -r ../BUILD/usr/bin/cgdelete $RPM_BUILD_ROOT/usr/bin/
cp -r ../BUILD/usr/bin/cgexec $RPM_BUILD_ROOT/usr/bin/
cp -r ../BUILD/usr/bin/cgget $RPM_BUILD_ROOT/usr/bin/
cp -r ../BUILD/usr/bin/cgset $RPM_BUILD_ROOT/usr/bin/
cp -r ../BUILD/usr/bin/cgsnapshot $RPM_BUILD_ROOT/usr/bin/
cp -r ../BUILD/usr/bin/lscgroup $RPM_BUILD_ROOT/usr/bin/
cp -r ../BUILD/usr/bin/lssubsys $RPM_BUILD_ROOT/usr/bin/
```

```
cp -r ../BUILD/usr/lib/systemd/system/cgconfig.service $RPM_BUILD_ROOT/usr/lib/systemd/system/
cp -r ../BUILD/usr/lib/systemd/system/cgred.service $RPM_BUILD_ROOT/usr/lib/systemd/system/
cp -r ../BUILD/usr/sbin/cgclear $RPM_BUILD_ROOT/usr/sbin/
cp -r ../BUILD/usr/sbin/cgconfigparser $RPM_BUILD_ROOT/usr/sbin/
cp -r ../BUILD/usr/sbin/cgrulesengd $RPM_BUILD_ROOT/usr/sbin/
cp -r ../BUILD/usr/share $RPM_BUILD_ROOT/usr/share/

%files
/etc/cgconfig.conf
/etc/cgconfig.d/
/etc/cgrules.d/
/etc/cgrules.conf
/etc/cgsnapshot_blacklist.conf
/etc/sysconfig/cgred
/usr/bin/cgclassify
/usr/bin/cgcreate
/usr/bin/cgdelete
/usr/bin/cgexec
/usr/bin/cgget
/usr/bin/cgset
/usr/bin/cgsnapshot
/usr/bin/lscgroup
/usr/bin/lssubsys
/usr/lib/systemd/system/cgconfig.service
/usr/lib/systemd/system/cgred.service
/usr/sbin/cgclear
/usr/sbin/cgconfigparser
/usr/sbin/cgrulesengd
/usr/share/doc/libcgroup-tools-0.41
/usr/share/man/*
```

(3). 将需要打包的文件拷贝至`rpmbuild/BUILD`目录下对应子目录

按照`.spec`文件中的`%files`的文件列表，将需要打包的文件手动拷贝到`rpmbuild/BUILD`目录下的相应子目录。文件来源有两部分：

一部分是来自上一小节编译好的文件。例如，列表项的`/usr/bin/cgclassify`文件，需要手动将编译好的`cgclassify`文件拷贝到`rpmbuild/BUILD/usr/bin/`目录：

```
[root@bogon rpmbuild]# pwd
/root/rpmbuild
[root@bogon rpmbuild]# mkdir -p BUILD/usr/bin/
[root@bogon rpmbuild]# cp /root/libcgroup/bin/cgclassify BUILD/usr/bin/
```

另一部分文件需要手动编写（当然，可以从发行版`Linux`现成安装包里提取），例如，列表项中的`/usr/lib/systemd/system/cgconfig.service`可以从`Redhat 7`的`libcgroup-tools-0.41-21.el7.x86_64.rpm`中提取，然后拷贝到`rpmbuild/BUILD/usr/lib/systemd/system/cgconfig.service`。为了简化这一步操作，笔者已经将这部分文件上传至`GitHub`：

- libcgroup：https://github.com/fairyfar/cgroup-cn/raw/refs/heads/main/archive/rpmbuild-libcgroup.tar
- libcgroup-tools：https://github.com/fairyfar/cgroup-cn/raw/refs/heads/main/archive/rpmbuild-libcgroup-tools.tar

(4). 执行打包命令

```
[root@bogon rpmbuild]# rpmbuild -bb ./SPECS/libcgroup-tools-0.41-19.el8.x86_64.spec
```

打包成功后，`rpm`包将生成在`rpmbuild/RPMS`目录下。

```
[root@bogon rpmbuild]# ll RPMS/x86_64/
total 92
-rw-r--r-- 1 root root 91168 Jan  9 07:20 libcgroup-tools-0.41-19.el8.x86_64.rpm
```

按照上述步骤继续制作`libcgroup rpm`安装包。

3.3.4 制作rpm过程种遇到的问题

(1). `contains an invalid rpath`

错误内容如下：

```
ERROR 0002: file 'xxx' contains an invalid rpath 'yyy' in [yyy]
```

解决方法：修改`~/.rpmmacros`文件，注释掉以下行：

```
%__arch_install_post \
    ... /usr/lib/rpm/check-rpaths /usr/lib/rpm/check-buildroot
```

(2). `/etc/ld.so.conf: No such file or directory`

警告内容如下：

```
/sbin/ldconfig: Warning: ignoring configuration file that cannot be opened: /etc/ld.so.conf: No such file or directory
```

警告可以忽略。

3.3.5 方法的局限性

过程相对繁琐，容易出错。但是，可以在任意发行版`Linux`上编译，与系统契合度更高，可按需自由定制。

3.3.6 cgroup v2支持情况

本方法是否支持`cgroup v2`取决于我们编译的`libcgroup`版本，以上示例我们是从`libcgroup v0.41`版本源码编译的，所以不支持`cgroup v2`，如果想支持`cgroup v2`，需要编译`libcgroup v2.0`以上版本。

经笔者验证，在`Redhat Linux 8.5`系统上，`libcgroup v2.0`可用。

4. 背景

本章节罗列一些背景资料，可选择性阅读。

4.1 libcgroup主要文件的说明

摘自参考文献[6]。

- `cgclassify`：指令用于将正在运行的任务移至一个或多个`cgroup`中。
- `cgclear`：指令用于删除层级中的全部`cgroup`。
- `cgconfigparser`：指令用于解析`cgconfig.conf`文件并且挂载层级。
- `cgcreate`：指令用于在层级中创建新的`cgroup`。
- `cgdelete`：指令用于移除指定的`cgroup`。
- `cgexec`：指令用于在指定的`cgroup`中运行任务。
- `cgget`：指令用于显示`cgroup`参数。
- `cgsnapshot`：指令用于从现存的子系统中生成配置文件。
- `cgrulesengd`：服务用于将任务分配到`cgroup`。
- `cgset`：指令用于为`cgroup`设定参数。
- `lscgroup`：指令用于将层级中的`cgroups`列表。
- `lssubsys`：指令将包含特定子系统的层级列表。
- `cgconfig.conf`：`cgroup`在`cgconfig.conf`文件中被定义。
- `cgred.conf`：是`cgred`服务的配置文件。
- `cgrules.conf`：包含可以确定任务何时归属于某一`cgroup`的规则。

4.2 cgred的工作原理

`libcgroup`通过注册`cgred`系统服务实现`Linux`用户资源管理，`cgred`服务实际上由应用程序`cgrulesengd`实现用户进程关联`cgroup`资源组功能。`cgrulesengd`应用程序的主要源代码对应`libcgroup`的`cgrulesengd.c`文件。

`cgrulesengd`通过`Netlink`机制获得应用程序创建与退出事件，然后根据应用程序所述用户将进程PID写入对应`cgroup`组的`cgroup.procs`或`tasks`接口文件。

`Netlink`机制作为一种内核与用户空间通信的机制，是一种特殊的`socket`通信方式，对于`Linux`内核与用户空间进行双向数据传输是非常好的方式，详见参考文献[7]。

4.3 海盒数据库SeaboxSQL的资源管理

开篇讨论的海盒`SeaboxSQL`数据库[8]，是北京东方金信科技股份有限公司积累多年数据库开发经验，打造的一款拥有完全自主知识产权、面向事务型业务处理的企业级关系型数据库。

本文讨论的利用`cgroup`控制多套数据库资源使用的解决方法属于普适方案，实际上`Seabox`数据库（`SeaboxSQL`和`SeaboxMPP`）支持资源组管理功能，可以按照数据库用户组限制多种资源配额，可以进行更细致的资源管理。

5. 参考文献

1. An updated performance comparison of virtual machines and Linux containers. IBM Research. DOI:10.1109/ISPASS.2015.7095802
2. Redhat Linux 7使用cgroup限制用户资源
3. cgroup-cn
4. Cgrulesengd is cut in rhel8-based systems
5. systemd.resource-control — Resource control unit settings
6. 使用libcgroup-附加资源
7. Introduction to Netlink
8. https://www.seaboxdata.com