KYLINSOFT
麒麟软件

银河麒麟高级服务器操作系统 V10 SP3 2403

安全指南

麒麟软件有限公司

2024 年 3 月

目录

1. 摘要 1

2. 安装麒麟安全增强组件 1

3. 开启麒麟安全 5

4. 关闭麒麟安全 10

5. 安装 PKS 组件 12

6. SELinux 15

6.1. SELinux 状态和模式的更改 16

6.2. 切换到 permissive 模式 16

6.3. 切换到 enforcing 模式 17

6.4. 在之前禁用的系统中启用 SELinux 安全机制 17

6.5. 禁用 SELinux 安全机制 19

6.6. 在引导时更改 SELinux 模式 22

6.7. 切换 ukmls/ukmcs/targeted 策略 22

6.8. 查看标记命令 24

6.9. 修改标记命令 24

6.10. 用户、角色和域的关系 25

7. KySEC 安全机制 26

7.1. KySEC 状态设置 27

7.1.1 查看 KySEC 安全状态 27

7.1.2 设置 KySEC 安全机制状态为宽松模式：.....27
7.1.3 设置 KySEC 安全机制状态为强制模式：.....27
7.1.4 通过命令临时设置 KySEC 安全机制状态：.....27
7.1.5 在开机启动时临时设置 KySEC 安全机制状态：.....28
7.2. 执行控制 30
7.2.1 开启执行控制安全状态 30
7.2.2 关闭执行控制安全状态 30
7.2.3 添加可执行程序的执行权限 30
7.2.4 取消可执行程序的执行权限 30
7.2.5 举例 30
7.3. 文件保护功能 34
7.3.1 开启文件保护 34
7.3.2 关闭文件保护 34
7.3.3 添加文件保护 35
7.3.4 取消文件保护 35
7.3.5 举例 35
7.4. 模块防卸载 36
7.4.1 开启模块防卸载 36
7.4.2 关闭模块防卸载 36
7.4.3 保护模块防卸载 36
7.4.4 取消保护模块防卸载 36

7.4.5 举例 36
7.5. 进程防杀功能 39
7.5.1 开启进程防杀 39
7.5.2 关闭进程防杀 40
7.5.3 保护进程防杀 40
7.5.4 取消保护进程防杀 40
7.5.5 举例 40
7.6. 联网管控功能 43
7.6.1 开启联网管控 43
7.6.2 关闭联网管控 43
7.6.3 禁止软件联网 43
7.6.4 允许软件联网 43
7.6.5 举例 44
8. 三权分立机制 46
8.1. 三权用户图形管理 46
8.2. 三权用户命令集 48
8.3. 启用三权分立 50
8.4. 关闭三权分立 51
9. 安全加固 53
10. 数据隔离保护机制 54
10.1. 开启文件保护箱机制 54

10.1.1 系统已安装安全组件下启用文件保护箱 54
10.1.2 仅安装保护箱组件的开启方式 55
10.1.3 在引导时开启文件保护箱 56
10.2. 关闭文件保护箱 57
10.2.1. 系统已安装安全组件下的关闭文件保护箱 57
10.2.2. 仅安装保护箱组件的关闭方式 57
10.2.3. 在引导时关闭文件保护箱 59
10.3. boxadm 的使用方法 60
11. USB 设备管控 60
11.1 用法 60
11.2 示例 61
12. 安全启动 65
13. 可信启动 70
13.1 设置开关状态 70
13.1.1 设置可信启动状态为 on 71
13.1.2 设置可信启动状态为 off 71
13.2 更新基准库 71
13.2.1 全量更新 71
13.2.2 更新指定文件 71
13.3 新增文件纳入可信启动基准库 71
13.4 移除已纳入可信启动基准库的文件 71

13.5 指定可信根设备 72

13.5.1 指定可信根设备（例可信根设备：/dev/tpm0） 72

13.6 查询可信根信息 72

13.7 查询可信报告 72

13.7.1 查询全部报告 72

13.7.2 查询 BIOS 阶段度量报告 72

13.7.3 查询 grub 阶段度量报告 73

13.8 安全保护设置 73

13.8.1 设置基准库的完整性保护口令 73

13.8.2 传入可信根的所有者保护口令（如 TPM 存储 hierarchy 设置了保护口令，开启可信启动功能时需传入） 74

13.9 PCR：8，9 扩展状态设置 74

13.9.1 设置 TPM 为可信根的 PCR 扩展 74

13.9.2 设置 PCR 扩展状态为 off 74

14. PKS 组件 75

14.1. 可信度量图形工具 75

14.1.1. 系统信任链 76

14.1.2. 系统启动度量 79

14.1.3. 系统启动度量配置 79

14.1.4. 查看系统启动度量详情 80

14.1.5. 重新采集基准值 80

14.2. 可信度量命令行工具 81
15. 动态度量 81
15.1 用法 82
15.1.1 检测环境 82
15.1.2 功能开关配置 84
15.1.3 查看动态度量策略 84
15.1.4 配置动态度量策略 87
15.2 示例 90
16. 麒麟安全管理工具-安全中心 92
17. 麒麟文件保护箱 93
18. 日志查看器 94
19. 安全功能配置 94
19.1 UID 唯一性增强 94
19.1.1 功能 94
19.1.2 用法 94
19.1.3 示例 95
19.2 PAM 配置国密 95
19.3 ssh 国密配置 95
19.3.1 功能 95
19.3.2 用法 96
19.3.3 示例 98

19.4 密码回文和相似性 99
19.4.1 功能 100
19.4.2 用法 100
19.4.3 示例 100
19.5 security.kysec 文件属性 100
19.5.1 功能 100
19.5.2 用法 100
19.5.3 示例 101

1. 摘要

本书帮助用户和管理员学习保护服务器免受本地和远程入侵、利用和恶意活动的流程和实践。

本文档侧重于银河麒麟高级服务器 V10 SP3 2403，介绍系统下如何配置 SELinux、KYSEC、三权分立、数据隔离保护等安全功能。

通过拥有正确的管理知识、对安全的重视及相关的工具，Linux 系统可以完全正常工作，并防止大多数安全入侵和攻击。

2. 安装麒麟安全增强组件

麒麟安全增强组件包括 KySEC，SELinux，BOX，AppArmor。其中 KySEC 具备执行管控，应用联网管控，进程防杀死，文件保护，模块防卸载，三权分立等安全功能子模块。

麒麟安全增强组件可以通过以下几种方式安装：系统安装时选择安装麒麟安全增强组件、系统安装后使用在线仓库源安装、系统安装后使用光盘源安装。安装方法如下：

系统安装时选择“麒麟安全增强工具”组件，如下图：

系统安装后使用源安装：

1. 执行 yum makecache，确认可以访问到源

```
[root@localhost ~]# yum makecache
Kylin Linux Advanced Server 10 - Os                  13 kB/s | 3.8 kB     00:00
Kylin Linux Advanced Server 10 - Updates             17 kB/s | 3.0 kB     00:00
Metadata cache created.
```

2. 执行 yum grouplist，确认组件“麒麟安全增强工具”可以安装

```
[root@localhost ~]# yum grouplist
Last metadata expiration check: 0:01:55 ago on 2024年03月14日 星期四 10时48分48秒.
Available Environment Groups:
   最小安装
   基础设施服务器
   文件及打印服务器
   基本网页服务器
   虚拟化主机
Installed Environment Groups:
   带 UKUI GUI 的服务器
Installed Groups:
   容器管理
   无图形终端系统管理工具
Available Groups:
   麒麟安全增强工具
   开发工具
   传统 UNIX 兼容性
   科学记数法支持
   安全性工具
   系统工具
   智能卡支持
   Man 手册
```

3. 执行 yun groupinstall，安装“麒麟安全增强工具”，按 y 确认安装

```
[root@localhost ~]# yum groupinstall 麒麟安全增强工具
Last metadata expiration check: 0:04:43 ago on 2024年03月14日 星期四 11时11分12秒.
Dependencies resolved.
================================================================================================
 Package                              Arch        Version                       Repository   Size
================================================================================================
Installing group/module packages:
 box-manager                          x86_64      3.1.3-1.se.09.ky10            Security    7.7 M
 box-utils                            x86_64      4.0.3-1.se.04.ky10            Security     28 k
 chpolicy                             x86_64      1.0.0-3.skl3.se.10.ky10       Security     11 k
 ksc-defender                         x86_64      1.0.1-5kord.se.24.ky10        Security     13 M
 kysec-daemon                         x86_64      1.1.1-01.se.18.ky10           Security    285 k
 kysec-sync-daemon                    x86_64      1.1.1-01.se.08.ky10           Security     55 k
 kysec-utils                          x86_64      3.4.1-01.se.24.ky10           Security     76 k
 kyseclog-daemon                      x86_64      1.1.1-01.se.07.ky10           Security     16 k
 kytrust-bima                         x86_64      2.0.14-07.ky10                Security     73 k
 libbox1                              x86_64      4.0.3-1.se.04.ky10            Security     37 k
 libkysec-extend                      x86_64      3.4.1-01.se.24.ky10           Security     80 k
 libkysecwhlist                       x86_64      3.4.1-01.se.24.ky10           Security     54 k
 libkytrusted-security                x86_64      1.0.1-08.ky10                 Security     65 k
 libpam-kysec                         x86_64      3.4.1-01.se.24.ky10           Security     34 k
 libsecurity-conf                     x86_64      1.2-1.se.03.ky10              Security     16 k
 libselinux-ruby                      x86_64      3.1-7.se.03.ky10              Security     47 k
 libtcmalg                            x86_64      1.0-se.04.ky10                Security     58 k
 python2-libselinux                   x86_64      3.1-7.se.03.ky10              Security    160 k
 python2-libsemanage                  x86_64      3.1-3.se.02.ky10              Security     70 k
 security-reinforce                   x86_64      1.1.0-10.ky10                 Security    1.0 M
 security-switch                      x86_64      1.3.1-1.se.04.ky10            Security     28 k
 selinux-policy-ukmcs                 noarch      3.14.2-81.se.29.06.ky10       Security    5.6 M
 setools                              x86_64      4.3.0-4.ky10                  Security    7.5 k
 setools-console                      x86_64      4.3.0-4.ky10                  Security     20 k
 trousers-tcm                         x86_64      0.3.15-05.ky10                Security    274 k
 trousers-tcm-tools                   x86_64      1.3.4-06.ky10                 Security    181 k
Installing dependencies:
 authselect-compat                    x86_64      1.2.1-2.p01.ky10              Security     32 k
 ecryptfs-utils                       x86_64      111-5.p01.ky10                Security    159 k
 ecryptfs-utils-python                x86_64      111-5.p01.ky10                Security     18 k
 glib                                 x86_64      1:1.2.10-57.ky10              Security    133 k
 grub2-efi-x64-modules                noarch      1:2.04-37.p08.se.06.ky10      Security    1.3 M
 grub2-efi-x64-trust-measure-modules  noarch      1:2.04-37.p08.se.06.ky10      Security     40 k
 gtk+                                 x86_64      1:1.2.10-90.ky10              Security    703 k
 json-c-devel                         x86_64      0.15-6.ky10                   Security     34 k
 libcurl-devel                        x86_64      7.71.1-32.ky10                Security    162 k
 libkysdk-module-authorize            x86_64      1.0.0-02.ky10                 Security     18 k
 libkysec-module-authorize            x86_64      1.0.0-1.se.04.ky10            Security     20 k
 openpgm                              x86_64      5.2.122-16.ky10               Security    169 k
 pkcs11-helper                        x86_64      1.25.1-1.ky10                 Security     59 k
 tpm2-tss-devel                       x86_64      3.0.3-2.p01.ky10              Security     57 k
 zeromq                               x86_64      4.1.7-2.ky10                  Security    190 k
Installing Groups:
 Kylin Security Enhance

Transaction Summary
================================================================================================
Install  41 Packages

Total size: 32 M
Installed size: 280 M
Is this ok [y/N]:
```

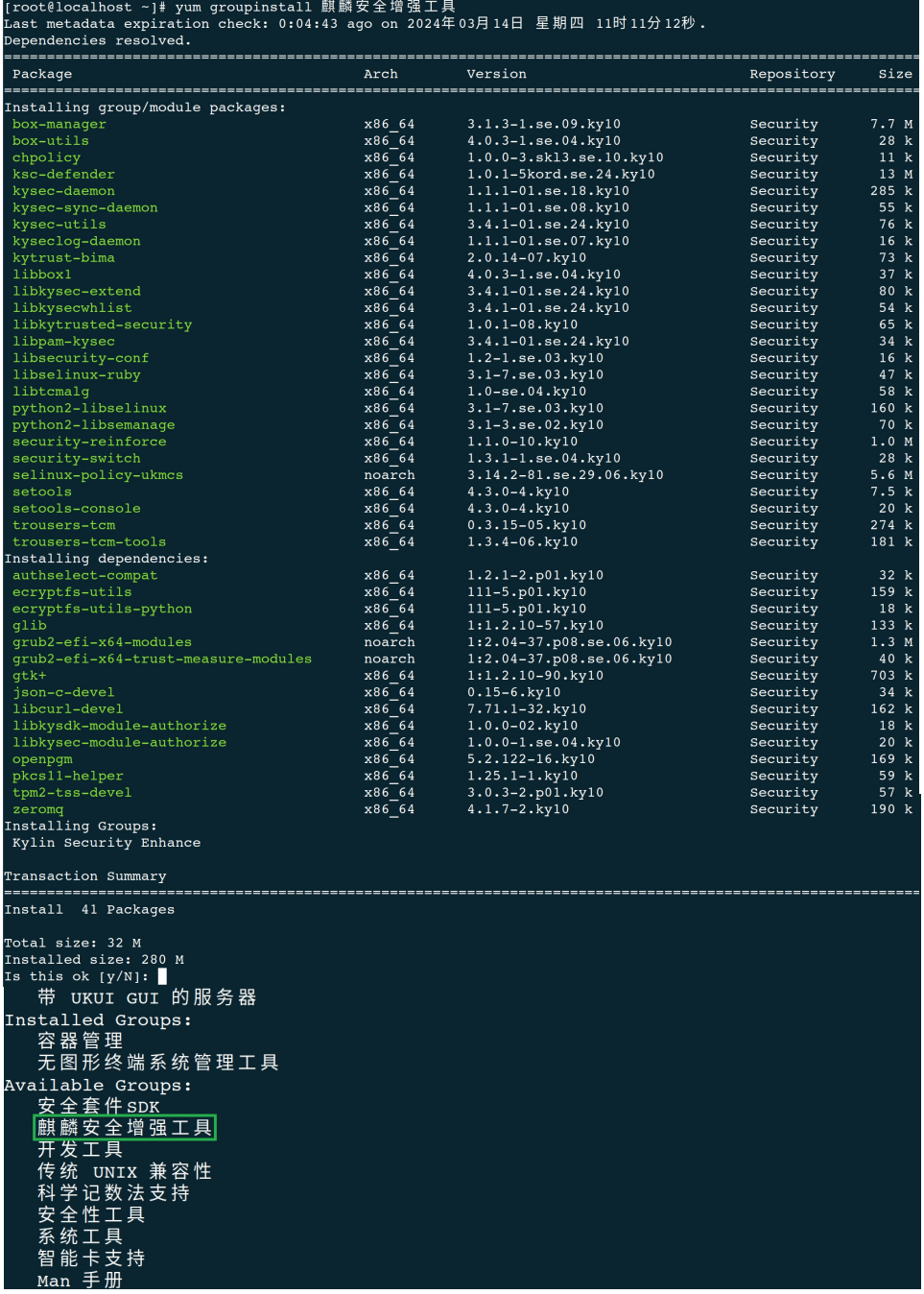
系统安装后使用光盘安装：

在终端中执行以下命令

1. 执行 lsblk，确认光盘挂载路径

```
[root@localhost ~]# lsblk
NAME           MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda              8:0    0   30G  0 disk
├─sda1           8:1    0    1G  0 part /boot
└─sda2           8:2    0   29G  0 part
  ├─klas-root  253:0    0   26G  0 lvm  /
  └─klas-swap  253:1    0    3G  0 lvm  [SWAP]
sr0             11:0    1  4.3G  0 rom  /run/media/root/Kylin-Server-10
```

2. 执行 mv /etc/yum.repos.d/kylin_x86_64.repo{,.bak},备份源文件（如果是其他架构，则按实际情况更改“kylin_x86_64.repo”，例如是



aarch64 架构的话，就改成“kylin_aarch64.repo”）

3. 执行 vi /etc/yum.repos.d/kylin_x86_64.repo，创建源文件，并写入以下内容：

```
[Security]
name = Security
baseurl = file:///run/media/root/Kylin-Server-10/
gpgcheck = 0
enabled = 1
```

4. 执行 yum makecache，确认可以访问光盘源

5. 执行 yum grouplist，确认组件“麒麟安全增强工具”可以安装

6. 执行 yum groupinstall，安装“麒麟安全增强工具”，按 y 确认

如果在安装时选择了“最小安装”，则使用以下步骤进行安装“麒麟安全增强组件”：

1. 安装安全组件：

```
yum install dracut bash python3 chpolicy kyseclog-daemon libkysec-adv
libkysec-core kysec-daemon kysec-sync-daemon kysec-utils libkysec
libkysec-extend libkysecwhlist libpam-kysec libsecurity1 libsecurity-conf
rpm security-switch libselinux selinux-policy selinux-policy-ukmcs
setools selinux-policy-devel security-reinforce
```

2. 执行 dracut 命令：dracut -force

3. 开启麒麟安全

麒麟安全功能的启用可通过命令行工具 security-switch 和 setstatus 启用。security-switch 是安全机制更换和管理工具，sestatus 是安全机制状态管理工具。其开启方法如下：

- 使用 security-switch 命令
  - security-switch --set default：开启 KySEC 安全机制（强制模式）

```
[root@localhost ~]# security-switch --set default
正在为 'Kysec' 检查内核配置... OK
正在为 'Kysec' 检查软件包安装状态... OK
正在为 'Kysec' 检查文件安装状态... OK
正在为开启 'Kysec' 检查系统配置状态... OK
正在为关闭 'SELinux' 检查系统配置状态... OK
正在为关闭 '三权分立' 检查系统配置状态... OK
正在启用 'Kysec' ... OK
正在禁用 'SELinux' ... OK
正在禁用 '三权分立' ... OK
正在更新系统启动配置... OK
正在写入系统安全配置... OK
系统安全级别切换成功，请立即重启系统生效!!!
```

  - security-switch --set strict：开启 KySEC 安全机制（强制模式）+

SELinux 安全机制 ukmcs 策略 + 三权分立机制

```
[root@localhost ~]# security-switch --set strict
正在为 'Kysec' 检查内核配置... OK
正在为 'Kysec' 检查软件包安装状态... OK
正在为 'Kysec' 检查文件安装状态... OK
正在为开启 'Kysec' 检查系统配置状态... OK
正在为 'SELinux' 检查内核配置... OK
正在为 'SELinux' 检查软件包安装状态... OK
正在为 'SELinux' 检查文件安装状态... OK
正在为开启 'SELinux' 检查系统配置状态... OK
正在为 '三权分立' 检查内核配置... OK
正在为 '三权分立' 检查软件包安装状态... OK
正在为 '三权分立' 检查文件安装状态... OK
正在为开启 '三权分立' 检查系统配置状态... OK
正在启用 'Kysec' ... OK
正在启用 'SELinux' ... OK
root 用户密码需要手动设置
请输入新密码：
请再次输入新密码进行确认：
secadm 用户密码需要手动设置
请输入新密码：
请再次输入新密码进行确认：
auditadm 用户密码需要手动设置
请输入新密码：
请再次输入新密码进行确认：
正在启用 '三权分立' ... OK
正在更新系统启动配置... OK
正在写入系统安全配置... OK
系统安全级别切换成功，请立即重启系统生效!!!
```

■ 使用 setstatus 命令

● 开启 KySEC 安全机制

1) 开启 KySEC 安全机制，并设置 KySEC 模式

◆ setstatus kysec –s enforcing：开启 KySEC 安全机制，并设置 KySEC 安全机制为强制模式

```
[root@localhost ~]# setstatus kysec -s enforcing
安全模块状态设置改变，需要重启系统才能生效！
```

◆ setstauts kysec –s permissive:开启 KySEC 安全机制，并设置 KySEC 安全机制为宽松模式

```
[root@localhost ~]# setstatus kysec -s permissive
安全模块状态设置改变，需要重启系统才能生效！
```

2) 设置 KySEC 安全机制子模块状态

- 设置执行管控子模块状态
  - 开启：setstatus kysec –f exectl –c on
  - 关闭：setstatus kysec –f exectl –c off
- 设置应用联网管控子模块状态
  - 开启：setstatus kysec –f netctl –c on
  - 关闭：setstatus kysec –f netctl –c off
- 设置进程防杀死子模块状态
  - 开启：setstatus kysec –f ppro –c on
  - 关闭：setstatus kysec –f ppro –c off
- 设置文件保护子模块状态
  - 开启：setstatus kysec –f fpro –c on
  - 关闭：setstatus kysec –f fpro –c off
- 设置模块防卸载子模块状态
  - 开启：setstatus kysec –f kmod –c on
  - 关闭：setstatus kysec –f kmod –c off

- 开启 SELinux 安全机制
  - setstatus selinux –s enforcing：开启 SELinux 安全机制，并设置 SELinux 安全机制为强制模式

```
[root@localhost ~]# setstatus selinux -s enforcing
安全模块状态设置改变，需要重启系统才能生效！
```

  - setstatus selinux –s permissive：开启 SELinux 安全机制，并设置 SELinux 安全机制为宽松模式

```
[root@localhost ~]# setstatus selinux -s permissive
安全模块状态设置改变，需要重启系统才能生效！
```

- 开启 BOX
  - setstatus box –s enable：开启 BOX 安全机制

```
[root@localhost ~]# setstatus box -s enable
安全模块状态设置改变，需要重启系统才能生效！
```

- 开启 AppArmor
  - setstatus apparmor –s enable：开启 AppArmor 安全机制

```
[root@localhost ~]# setstatus apparmor -s enable
安全模块状态设置改变，需要重启系统才能生效！
```

也可通过修改 grub 引导参数，在系统启用前配置好需要使用的安全机制及其状态，启用方式为：

■ 系统环境下修改：

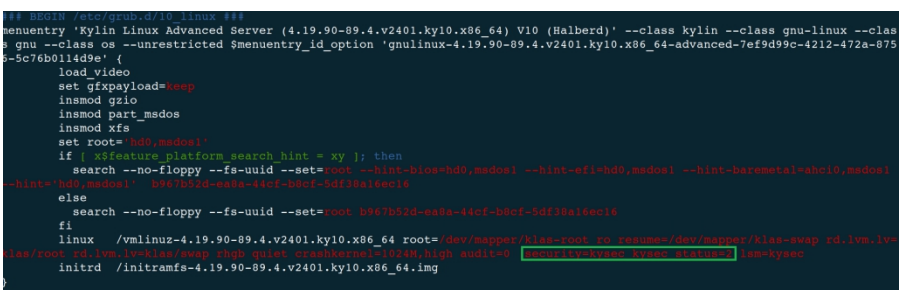
1. 找到 /boot 目录下系统启动使用的 grub.cfg 文件

```
[root@localhost ~]# cd /boot
[root@localhost boot]# find | grep grub.cfg
./grub2/grub.cfg
```

2. 修改 grub.cfg 文件

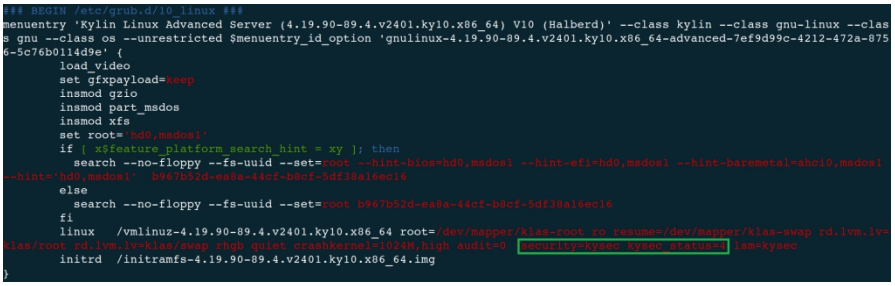
(1) 开启 KySEC 安全机制

- 开启 KySEC 安全机制，并设置为强制模式：在启动项内添加内核启动参数 security=kysec，kysec_status=2

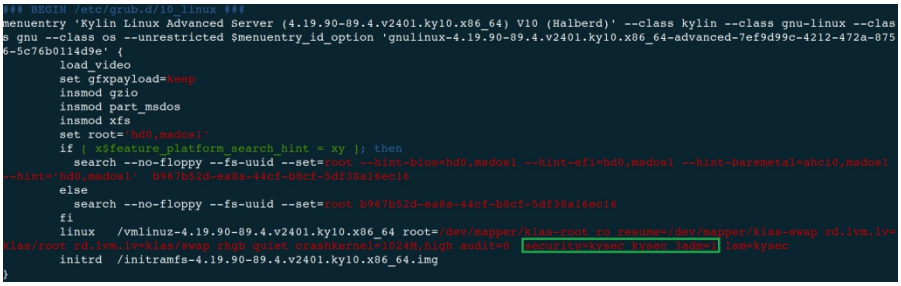


- 开启 KySEC 安全机制，并设置为宽松模式：在启动项内添加

内核启动参数 security=kysec，kysec_status=4

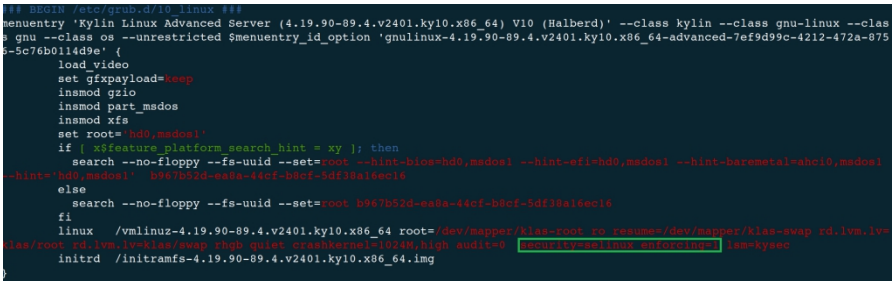


◆ 开启三权分立安全机制：在启动项内添加内核启动参数 security=kysec，kysec_3adm=1

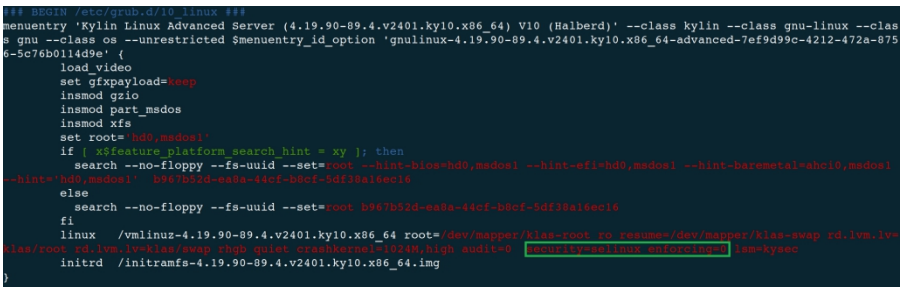


(2) 开启 SELinux 安全机制

◆ 开启 SELinux 安全机制，并设置为强制模式：在启动项内添加内核启动参数 security=selinux，enforcing=1



◆ 开启 SELinux 安全机制，并设置为宽松模式：在启动项内添加内核启动参数 security=selinux，enforcing=0



■ 系统引导时修改：

当出现 grub 菜单时，按“e”键，输入用户名和密码，进入 grub 启动项编辑界面，之后编辑内核启动参数，编辑内容同上。编辑完成后按 F10 或 Ctrl+x 进入系统引导。

4. 关闭麒麟安全

■ 使用 security-switch 命令:

◆ security-switch --set none：关闭所有安全机制

```
[root@localhost ~]# security-switch --set none
正在为关闭 'Kysec' 检查系统配置状态... OK
正在为关闭 'SELinux' 检查系统配置状态... OK
正在为关闭 '三权分立' 检查系统配置状态... OK
正在禁用 'Kysec' ... OK
正在禁用 'SELinux' ... OK
正在禁用 '三权分立' ... OK
正在更新系统启动配置... OK
正在写入系统安全配置... OK
系统安全级别切换成功，请立即重启系统生效!!!
```

■ 使用 setstatus 命令:

◆ setstauts kysec -s disable:关闭 KySEC 安全机制

```
[root@localhost ~]# setstatus kysec -s permissive
安全模块状态设置改变，需要重启系统才能生效！
```

◆ setstauts selinux -s disable:关闭 SELinux 安全机制

```
[root@localhost ~]# setstatus selinux -s disable
安全模块状态设置改变，需要重启系统才能生效！
```

◆ setstauts box -s disable:关闭 BOX 安全机制

```
[root@localhost ~]# setstatus box -s disable
安全模块状态设置改变，需要重启系统才能生效！
```

◆ setstauts apparrmor -s disable:关闭 AppArmor 安全机制

```
[root@localhost ~]# setstatus apparmor -s disable
安全模块状态设置改变，需要重启系统才能生效！
```

通过 Grub 关闭

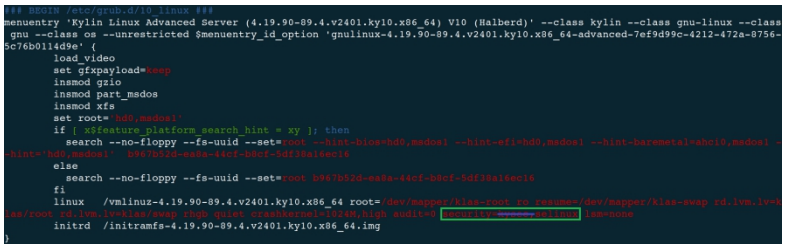
■ 系统环境下修改：

1. 找到 /boot 目录下系统启动使用的 grub.cfg 文件

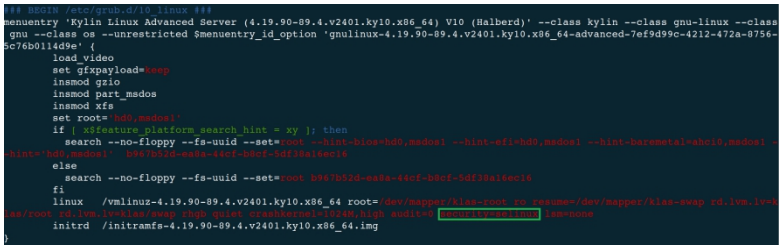
```
[root@localhost ~]# cd /boot
[root@localhost boot]# find | grep grub.cfg
./grub2/grub.cfg
```

2. 修改 grub.cfg 文件

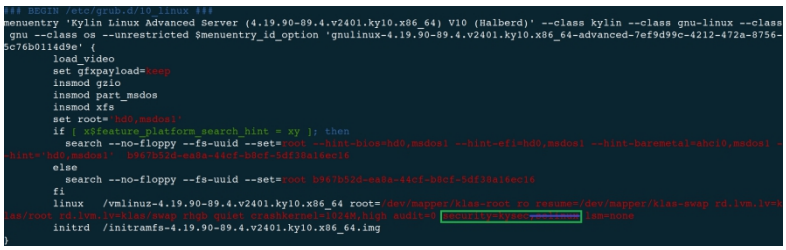
(1) 仅关闭 KySEC 安全机制，保留 SELinux 安全机制：在启动项内删除内核启动参数 security 等于号后面的 kysec，即将 security=kysec,selinux 改为 security=selinux



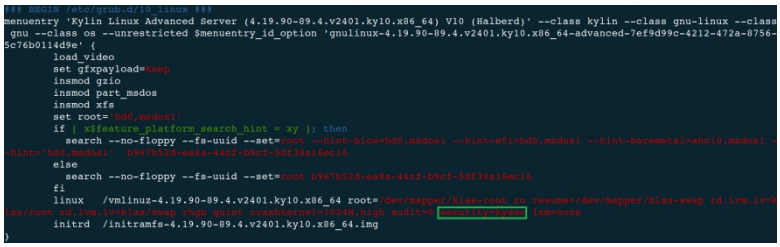
↓



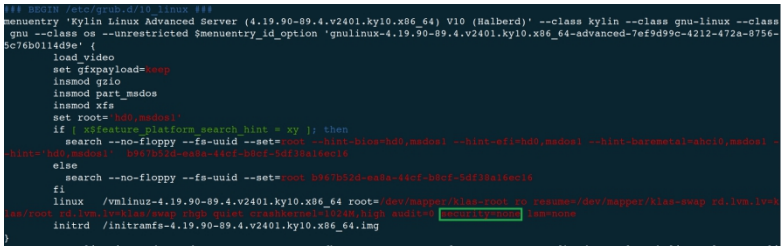
(2) 仅关闭 SELinux 安全机制，保留 KySEC 安全机制：在启动项内删除内核启动参数 security 等于号后面的 selinux，即将 security=kysec,selinux 改为 security=kysec



↓



(3) 关闭 KySEC 和 SELinux 安全机制：将启动项内的内核启动参数 security 等于号后面改为 none，即 security=none



■ 系统引导时修改：

当出现 grub 菜单时，按“e”键，输入用户名和密码，进入 grub 启动项编辑界面，之后编辑内核启动参数，编辑内容同上。编辑完成后按 F10 或 Ctrl+x 进入系统引导。

5. 安装 PKS 组件

PKS 组件可以通过以下几种方式安装：系统安装时选择安装 PKS 组件、系统安装后使用在线仓库源安装、系统安装后使用光盘源安装。安装方法如下：

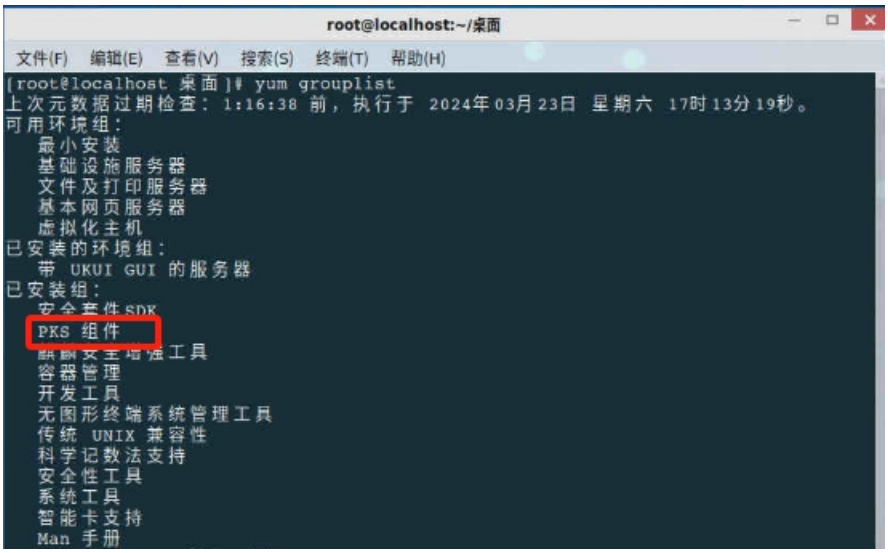
系统安装时选择“PKS 组件”，如下图

系统安装后使用源安装：

1. 执行 yum makecache，确认可以访问到源

```
[root@localhost ~]# yum makecache
Kylin Linux Advanced Server 10 - Os                  13 kB/s | 3.8 kB     00:00
Kylin Linux Advanced Server 10 - Updates             17 kB/s | 3.0 kB     00:00
Metadata cache created.
```

2. 执行 yum grouplist，确认组件“PKS 组件”可以安装



3. 执行 yun groupinstall，安装“PKS 组件”，按 y 确认安装

```
[root@localhost 桌面]# yum groupinstall "PKS 组件"
上次元数据过期检查：1:18:32 前，执行于 2024年03月23日 星期六 17时13分19秒。
依赖关系解决。
================================================================================
 Package            Architecture       Version                Repository      Size
================================================================================
Installing Groups:
 PKS Group

事务概要
================================================================================

确定吗？[y/N]：
```

系统安装后使用光盘安装：

在终端中执行以下命令

1. 执行 lsblk，确认光盘挂载路径

```
[root@localhost ~]# lsblk
NAME           MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda              8:0    0   30G  0 disk
├─sda1           8:1    0    1G  0 part /boot
└─sda2           8:2    0   29G  0 part
  ├─klas-root  253:0    0   26G  0 lvm  /
  └─klas-swap  253:1    0    3G  0 lvm  [SWAP]
sr0             11:0    1  4.3G  0 rom  /run/media/root/Kylin-Server-10
```

2. 执行 mv /etc/yum.repos.d/kylin_aarch.repo{,.bak},备份源文件（如　　果是其他架构，则按实际情况更改“kylin_aarch.repo”

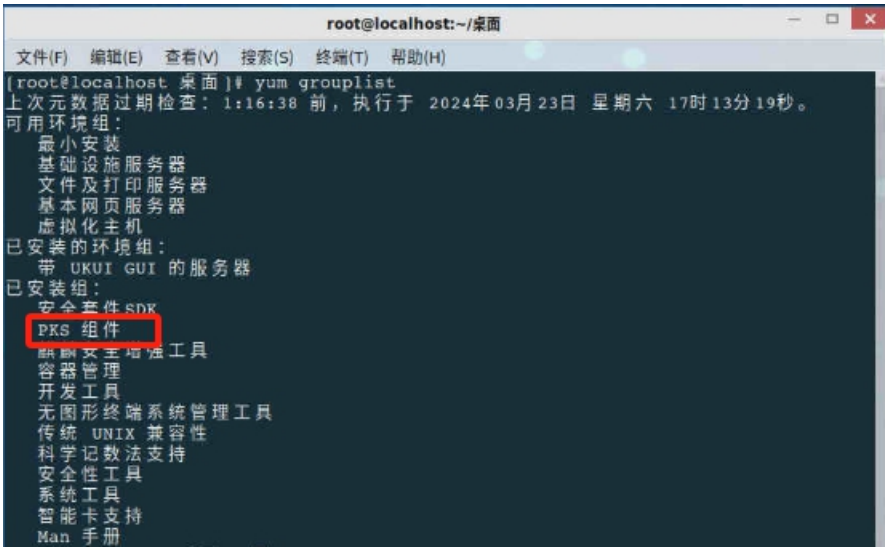
3. 执行 vi /etc/yum.repos.d/kylin_aarch.repo，创建源文件，并写入以下内容：

```
[Security]
name = Security
baseurl = file:///run/media/root/Kylin-Server-10/
gpgcheck = 0
enabled = 1
```

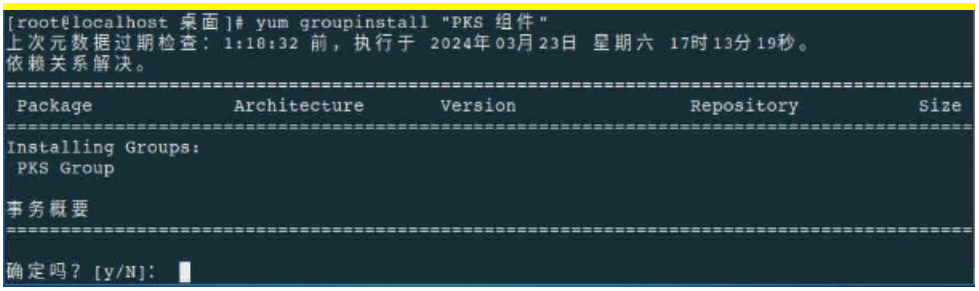
4. 执行 yum makecache，确认可以访问光盘源

```
[root@localhost ~]# yum makecache
Security                                        31 MB/s | 3.9 MB     00:00
Last metadata expiration check: 0:00:01 ago on 2024年03月14日 星期四 11时11分12秒.
Metadata cache created.
```

5. 执行 yum grouplist，确认组件“PKS 组件”可以安装



6. 执行 yum groupinstall，安装“PKS 组件”，按 y 确认



如果在安装时选择了“最小安装”，则按如下步骤安装“PKS 组件”：

1. 安装 PKS 组件：

```
yum install grub2-efi-aa64-modules kylin-log-viewer kyee-utils
libkyee-security libteec tee-supplicant hsdimm-lite ksc-defender
libkytrusted-security
```

2. 执行 grub 命令：grub2-install

6. SELinux

银河麒麟高级服务器操作系统 V10 SP3 2403 根据系统安全需求，定制了系统图形登录功能策略、三权分立功能策略、审计服务策略、执行控制功能策略、

白名单功能策略等系统功能策略。根据不同应用场景，提供 ukmls、ukmcs 和 target 三套 SE 基础策略。

6.1. SELinux 状态和模式的更改

（1）当前安全策略模式查询

我们提供了一些工具命令可以查询当前的安全策略模式，如 /usr/sbin/getenforce 和/usr/sbin/sestatus。所有用户都可以通过这些命令进行查询。

getenforce 命令输出若为 Permissive 则表示当前处于允许模式，若为 Enforcing 则表示当前处于强制模式。

sestatus 命令的输出可能为以下内容：

```
[root@localhost ~]#
[root@localhost ~]# sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             ukmcs
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      31
[root@localhost ~]#
```

它不仅反应了当前安全策略的模式，同时也可以得到其他安全策略的信息：安全策略开启状态、安全策略文件系统挂载位置、配置文件中的默认模式、安全策略版本和安全策略目录名称。

6.2. 切换到 permissive 模式

在系统出现故障时我们也可以先禁用我们的安全策略进入到维护模式进行

修复。进入方法如下：

a) 系统启动时进入 grub 菜单。

b) 输入 grub 密码修改内核启动参数，将参数 selinux=1 enforcing=1 修改为 selinux=1 enforcing=0 。

（2）启动系统。

这样系统就直接进入到维护模式，由 root 用户对系统进行修复。

6.3. 切换到 enforcing 模式

为了安全考虑所以系统开启安全后默认都会进入到强制模式，但是也提供了可以修改当前安全策略模式的命令。

setenforce 1 设置为强制模式；

setenforce 0 设置为允许模式。

三员模式（strict）下，系统三个管理员中只有安全管理员可以使用 setenforce 1 进入到强制模式，也只有安全管理员可以使用 setenforce 0 进入到允许模式。非三员模式下，root 用户可以使用 setenforce 修改 selinux 策略模式。

6.4. 在之前禁用的系统中启用 SELinux 安全机制

a）安装安全组件的状态下查看与开启

通过 security-switch --get、getstatus、getenforce 等命令查看当前安全级别、安全状态；

当安全级别为 none、default 时，通过 setstatus selinux -s enforcing 开启 SELinux 安全机制，如下图所示：

```
[root@localhost ~]#
[root@localhost ~]# getenforce
Disabled
[root@localhost ~]# getstatus
KySec status: disable
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]# security-switch  --get

当前安全级别 : 关闭(none)
-----------------------------------------------------
  安全模块      |     当前状态      |     默认状态
-----------------------------------------------------
[root@localhost ~]# setstatus  selinux -s enforcing
安全模块状态设置改变，需要重启系统才能生效！
[root@localhost ~]#
```

```
[root@localhost 桌面]# getenforce
Disabled
[root@localhost 桌面]# getstatus
KySec status: enforcing
        exectl  : on
        netctl  : on
        fpro    : on
        kmod    : on
        ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost 桌面]# security-switch --get

当前安全级别 : 推荐(default)
-----------------------------------------------------
  安全模块      |     当前状态      |     默认状态
-----------------------------------------------------
  Kysec          启用(Enforcing)    启用(Enforcing)
[root@localhost 桌面]# setstatus  selinux -s enforcing
安全模块状态设置改变，需要重启系统才能生效！
[root@localhost 桌面]#
```

b）未安装安全组件的状态下查看与开启

查看系统安全机制状态

```
[root@localhost 桌面]# getenforce
Disabled
[root@localhost 桌面]# sestatus
SELinux status:                 disabled
[root@localhost 桌面]#
```

开启系统 SELinux 安全机制

1） 使能安全机制，修改 boot 下的 grub.cfg 文件,在内核参数处增加

security=selinux　enforcing=1 参数；

```
### BEGIN /etc/grub.d/10_linux ###
menuentry 'Kylin Linux Advanced Server (4.19.90-89.4.v2401.ky10.x86_64) V10 (H
alberd)' --class kylin --class gnu-linux --class gnu --class os --unrestricted
 $menuentry_id_option 'gnulinux-4.19.90-89.4.v2401.ky10.x86_64-advanced-a55a64
10-a90d-4d22-a8ae-51da1bba9f65' {
        load_video
        set gfxpayload=keep
        insmod gzio
        insmod part_msdos
        insmod xfs
        set root='hd0,msdos1'
        if [ x$feature_platform_search_hint = xy ]; then
          search --no-floppy --fs-uuid --set=root --hint='hd0,msdos1'  73ebe1d
e-6e9f-4381-a646-c83677a1b8e4
        else
          search --no-floppy --fs-uuid --set=root 73ebe1de-6e9f-4381-a646-c836
77a1b8e4
        fi
        linux   /vmlinuz-4.19.90-89.4.v2401.ky10.x86_64 root=/dev/mapper/klas-
root ro resume=/dev/mapper/klas-swap rd.lvm.lv=klas/root rd.lvm.lv=klas/swap r
hgb quiet crashkernel=1024M,high audit=0 security=selinux enforcing=1
        initrd /initramfs-4.19.90-89.4.v2401.ky10.x86_64.img
}
```

2）修改/etc/selinux/config，调整 selinux 参数为 enforcing。

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
#SELINUX=disabled
SELINUX=enforcing
# SELINUXTYPE= can take one of these three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are p
rotected.
#     ukmls - Multi Level Security protection.
#     ukmcs -ukmcs variants of the SELinux policy.
#SELINUXTYPE=targeted
SELINUXTYPE=targeted

# SETLOCALDEFS= Check local definition changes
SETLOCALDEFS=0


~
~
"/etc/selinux/config" 20L, 704B                              20,0-1        全部
```

3）若是首次开启 SELinux 安全机制，则需要创建初始化文件。

```
[root@localhost ~]#
[root@localhost ~]# touch  /.autorelabel
[root@localhost ~]#
```

重启系统，重启过程中会进行打标签，耐心等候即可。

6.5. 禁用 SELinux 安全机制

① 在已安装安全组件的状态下查看与禁用

当安全级别为 strict 模式，需要通过 secadm 安全管理员用户来禁用 SELinux 机制。

查看系统安全机制状态

```
[secadm@localhost ~]$
[secadm@localhost ~]$
[secadm@localhost ~]$ getenforce
Enforcing
[secadm@localhost ~]$ getstatus
KySec status: enforcing
        exectl  : on
        netctl  : on
        fpro    : on
        kmod    : on
        ppro    : on
selinux status: enforcing
apparmor status: disable
box status: disable
[secadm@localhost ~]$ security-switch --get

当前安全级别 : 安全优先(strict)
-----------------------------------------------------
  安全模块      |     当前状态      |     默认状态
-----------------------------------------------------
  Kysec          启用(Enforcing)    启用(Enforcing)
  SELinux        启用(Enforcing)    启用(Enforcing)
  三权分立       启用               启用
[secadm@localhost ~]$
```

关闭 SELinux 安全机制

临时关闭：

```
[secadm@localhost ~]$ setenforce 0
[secadm@localhost ~]$ getenforce
Permissive
[secadm@localhost ~]$
```

永久关闭

```
[secadm@localhost ~]$ setstatus selinux -s  disable
安全模块状态设置改变，需要重启系统才能生效！
[secadm@localhost ~]$
```

当安全级别为 custom 时，通过 root 用户来关闭 SELinux 安全机制。

```
[root@localhost 桌面]# getenforce
Enforcing
[root@localhost 桌面]# getstatus
KySec status: disable
selinux status: enforcing
apparmor status: disable
box status: disable
[root@localhost 桌面]# security-switch  --get

当前安全级别 ：用户自定义(custom)
------------------------------------------------------
  安全模块      |     当前状态      |     默认状态
------------------------------------------------------
  SELinux        启用(Enforcing)    启用(Enforcing)
[root@localhost 桌面]#
```

临时关闭：

```
[root@localhost 桌面]#
[root@localhost 桌面]# setenforce 0
[root@localhost 桌面]# getenforce
Permissive
[root@localhost 桌面]#
```

永久关闭：

```
[root@localhost 桌面]#
[root@localhost 桌面]# setstatus selinux -s disable
安全模块状态设置改变，需要重启系统才能生效！
[root@localhost 桌面]#
```

② 在未安装安全组件的状态下查看与禁用

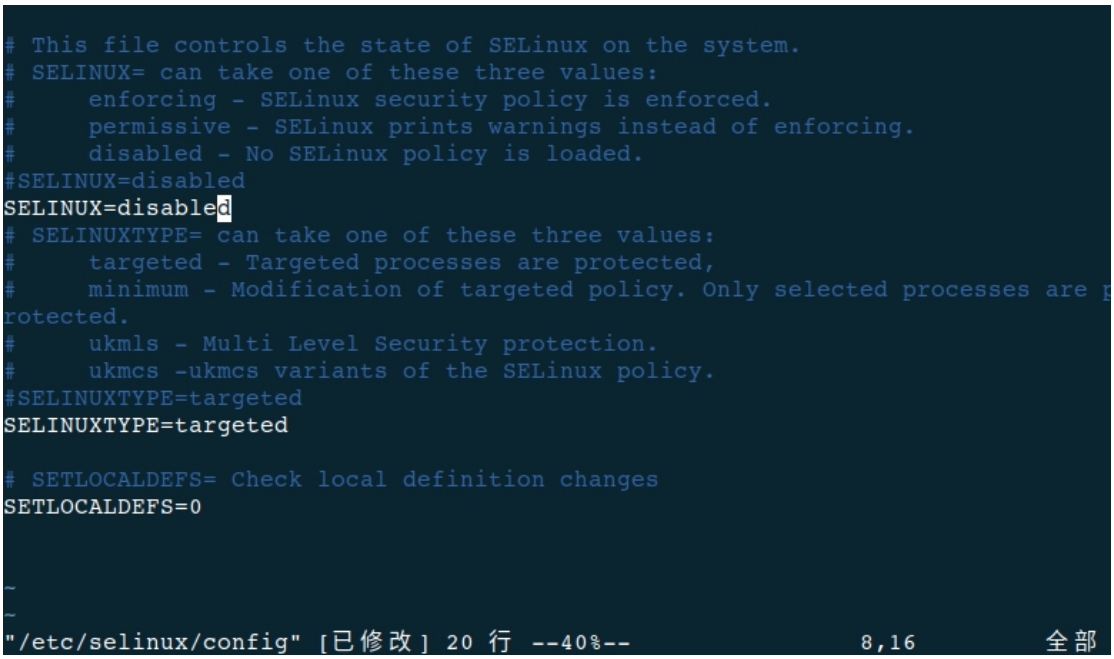
查看 SELinux 安全机制状态：

系统管理员通过 getenforce 命令来查看 SELinux 状态

```
[root@localhost ~]#
[root@localhost ~]# getenforce
Enforcing
[root@localhost ~]#
```

禁用 SELinux 安全机制：

系统管理通过修改/etc/selinux/config 配置文件，将文件中的 SELINUX 字段修改为 disabled，重启系统即可。

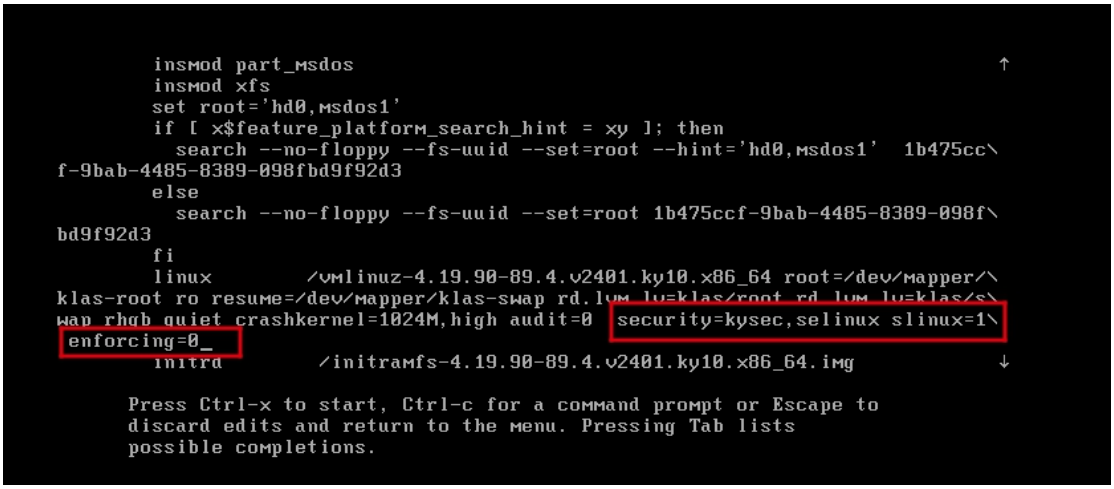


6.6. 在引导时更改 SELinux 模式

系统启动时进入 grub 菜单，输入 grub 密码修改内核启动参数：

selinux=1 时,表示开启 SELinux 安全机制，同时，enforcing=1，表示 SELinux 模式为强制状态，enforcing=0,表示 SELinux 模式为允许状态（不拦截，只记录日志）。

selinux=0 时，表示关闭 SELinux 安全机制。



6.7. 切换 ukmls/ukmcs/targeted 策略

方式一：

通过修改/etc/selinux/config 中的 SELINUXTYPE 字段的值来切换 ukmls、ukmcs、targeted 策略，重启后生效。

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
#SELINUX=disabled
SELINUX=enforcing
# SELINUXTYPE= can take one of these three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are p
rotected.
#     ukmls - Multi Level Security protection.
#     ukmcs -ukmcs variants of the SELinux policy.
#SELINUXTYPE=targeted
SELINUXTYPE=ukmcs

# SETLOCALDEFS= Check local definition changes
SETLOCALDEFS=0
```

方式二：

当 SELinux 安全机制启用时，安全管理员通过执行命令 chpolicy 来切换 ukmls、ukmcs、targeted 三种不同的内置策略

```
[secadm@localhost ~]$ security-switch  --get

当前安全级别 ：安全优先(strict)
-----------------------------------------------------
  安全模块      |     当前状态      |     默认状态
-----------------------------------------------------
  Kysec          启用(Enforcing)    启用(Enforcing)
  SELinux        启用(Enforcing)    启用(Enforcing)
  三权分立       启用               启用
[secadm@localhost ~]$ sestatus
SELinux status:                 enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:             ukmcs
Current mode:                   enforcing
Mode from config file:          enforcing
Policy MLS status:              enabled
Policy deny_unknown status:     allowed
Memory protection checking:     actual (secure)
Max kernel policy version:      31
[secadm@localhost ~]$ chpolicy  targeted
策略配置完成，需要重启系统才能生效！
[secadm@localhost ~]$
```

6.8. 查看标记命令

系统中一些常用命令的-Z 选项都提供了查看文件或进程安全上下文的功能，如：ls -Z、id -Z 和 ps -Z，这些命令加了-Z 选项后就会显示该文件或进程的安全上下文，如 ls -Z 一个文件，输出以下内容：

```
[root@localhost ~]#
[root@localhost ~]# ls -Z  anaconda-ks.cfg
root:object_r:sysadm_home_t:s0 anaconda-ks.cfg
[root@localhost ~]#
```

其中 sysadm_u 就表示该文件属于 sysadm_u 安全策略用户，object_r 表示该文件属于客体类型的角色，admin_home_t 表示该文件的类型，我们的策略就是根据这些标记来制定规则进行访问控制的。

6.9. 修改标记命令

安全管理员可以通过一些命令对系统中所有文件的安全上下文进行操作，如：chcon、setfiles 和 restorecon 命令。

/bin/ls 命令为 bin_t 类型，安全管理员可以输入 chcon -t sbin_t /bin/ls 将 bin_t 修改为 sbin_t 类型。

chcon 命令的-u 参数可以修改该文件安全上下文中的安全策略用户，-r 参数可以修改它的角色，-l 参数可以修改它的安全级别，其他具体使用可以查看它的帮助手册。

setfiles 命令可以根据我们的安全策略的标记文件对系统中所有文件进行标记。可通过 sestatus 来查看当前策略类型，如：

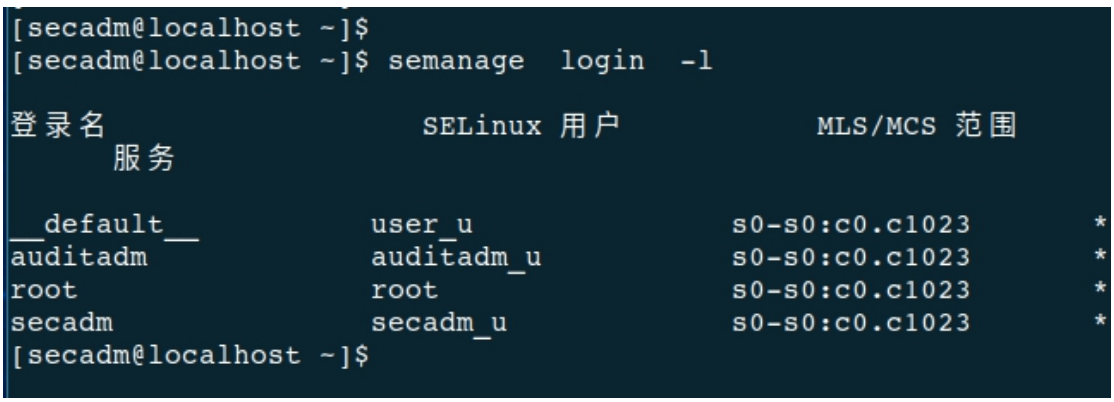
restorecon 命令可以恢复该文件的默认安全上下文，如：restorecon -r filename。

在对文件的安全上下文标记进行操作时首先都会检查这个标记的正确性，然后再去进行操作，具体使用可以查看用户手册。

6.10. 用户、角色和域的关系

登录程序如（login，sshd）负责映射 Linux 用户到安全策略用户，在登录时，如果安全策略用户标识符恰好和一个 Linux 用户标识符完全相同，匹配的安全策略用户标识符成为初始 shell 进程安全上下文的用户标识符。安全管理员也可使用 semanage 命令完成映射，如：semanage login -a -s charlie_u charlie，表示 Linux 用户 charlie 作为安全策略用户 charlie_u 登录系统。

使用 semanage login -l 命令可以查看，如下图：



一个用户可以对应多个角色，但在同一时刻只能作为其中一个角色，可以通过策略管理工具 semanage 输入 semanage user -a -R mgr_r -R charlie_r -P user charlie_u 来创建一个可以作为角色 mgr_r 和 charlie_r 的安全策略用户 charlie_u。

可通过 semanage user -l 进行查看，如下图：

```
[secadm@localhost ~]$ semanage  user -l

                标记中         MLS/       MLS/
SELinux 用户       前缀          MCS 级别     MCS 范围                        S
ELinux 角色

auditadm_u      audadm     s0         s0-s0:c0.c1023                  auditadm_
r
guest_u         user       s0         s0                              guest_r
root            sysadm     s0         s0-s0:c0.c1023                  sysadm_r
secadm_u        secadm     s0         s0-s0:c0.c1023                  secadm_r
staff_u         user       s0         s0-s0:c0.c1023                  staff_r s
ysadm_r system_r unconfined_r
sysadm_u        user       s0         s0-s0:c0.c1023                  sysadm_r
system_u        user       s0         s0-s0:c0.c1023                  system_r
unconfined_r
unconfined_u    user       s0         s0-s0:c0.c1023                  system_r
unconfined_r
user_u          user       s0         s0-s0:c0.c1023                  user_r
xguest_u        user       s0         s0                              xguest_r
[secadm@localhost ~]$
```

角色仅仅是一套域类型的集合，方便与用户建立联系，具体的访问控制都是通过进程的类型和客体的类型根据制定的规则来进行控制。

7. KySEC 安全机制

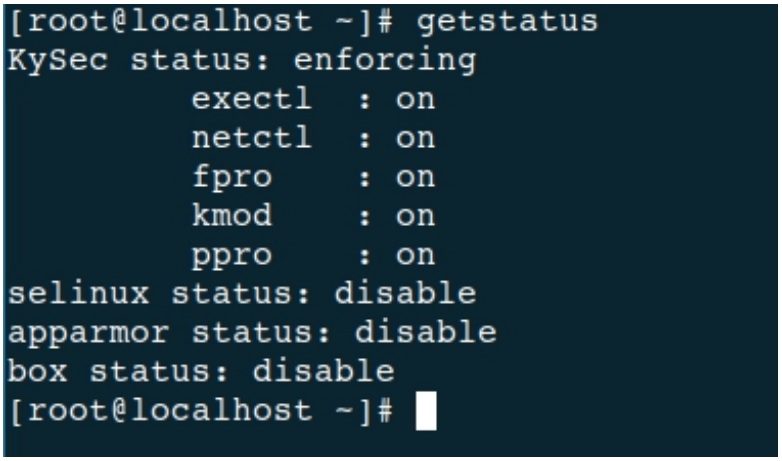
银河麒麟高级服务器操作系统 V10 SP3 2403 结合国产操作系统特点和现有国内外操作系统的安全机制，提出了基于标记的软件执行控制机制，实现对系统应用程序标记识别和执行约束，确保应用来源的可靠性和应用本身的完整性。执行控制机制控制文件执行、模块加载和共享库使用，分为系统文件、第三方应用程序，其中只允许具有合法标记的文件执行，任何网络下载、拷贝等外来软件均被禁止执行。

7.1. KySEC 状态设置

KySEC 共有两种状态；强制模式（enforcing）、软模式（permissive）。强制模式下不能执行非法应用程序，即禁止用户执行没有合法标记的应用程序，而软模式下，只是记录非法应用程序的执行行为，不禁止用户操作。系统默认 KySEC 安全状态为关闭状态，且执行控制、文件保护、内核保护、进程防杀、联网管控状态都处于关闭状态。

7.1.1 查看 KySEC 安全状态

执行 getstatus 可查看 KySEC 安全机制的状态



7.1.2 设置 KySEC 安全机制状态为宽松模式：

以 root 权限执行 setstatus kysec -s permissive

7.1.3 设置 KySEC 安全机制状态为强制模式：

以 root 权限执行 setstatus kysec -s enforcing

7.1.4 通过命令临时设置 KySEC 安全机制状态：

通过以下命令设置 kysec 状态，但仅影响本次开机，不影响重启后的状态。

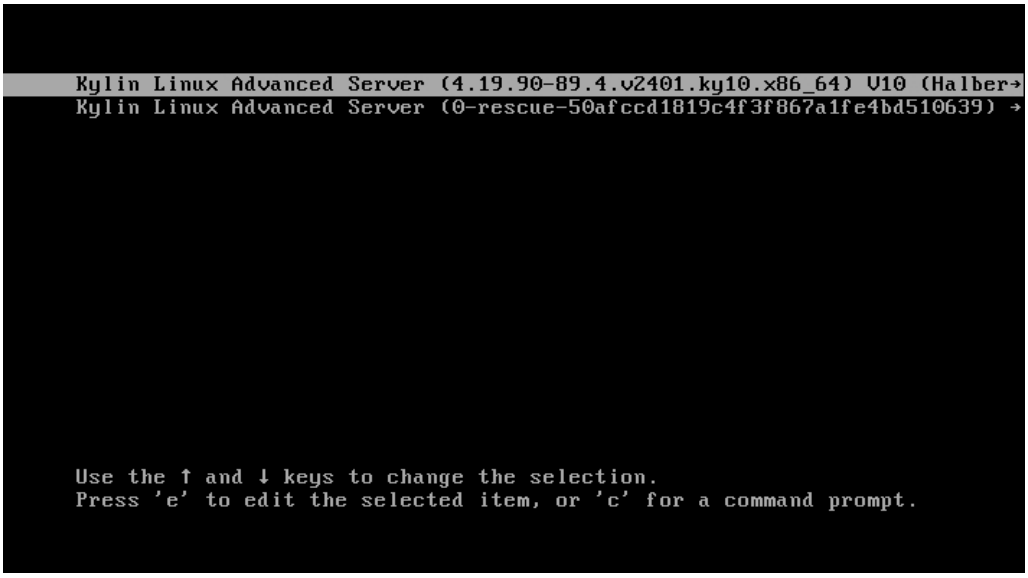
```
# 临时设置为软模式

setstatus kysec -s permissive -t
```

```
# 临时设置为强制模式

setstatus kysec -s enforcing -t
```

7.1.5 在开机启动时临时设置 KySEC 安全机制状态：

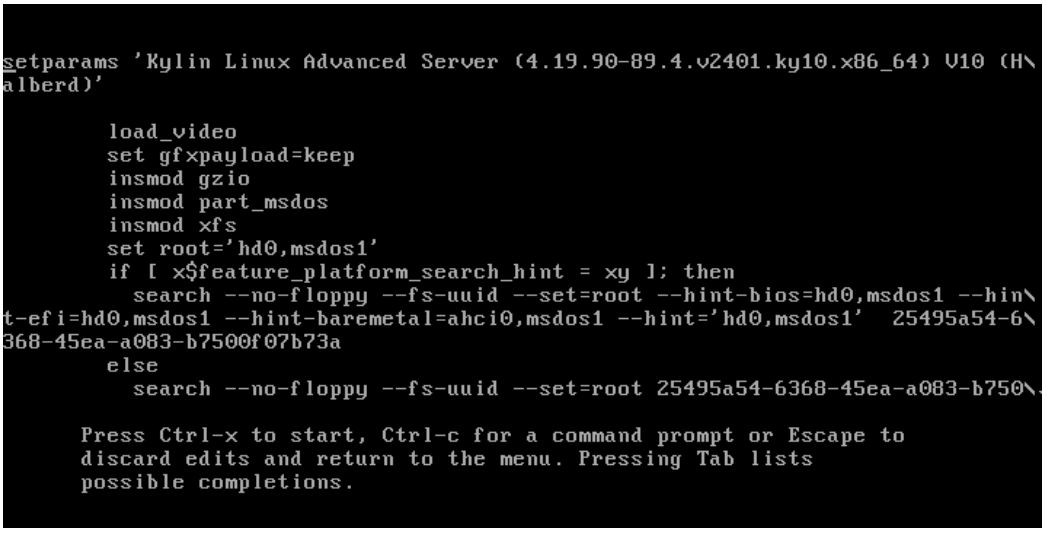
在开机过程中，可以在 grub 阶段设置本次开机时 kysec 的安全状态，同样仅影响本次开机。

开机过程中，到达以下界面时，按键盘的“e”键：



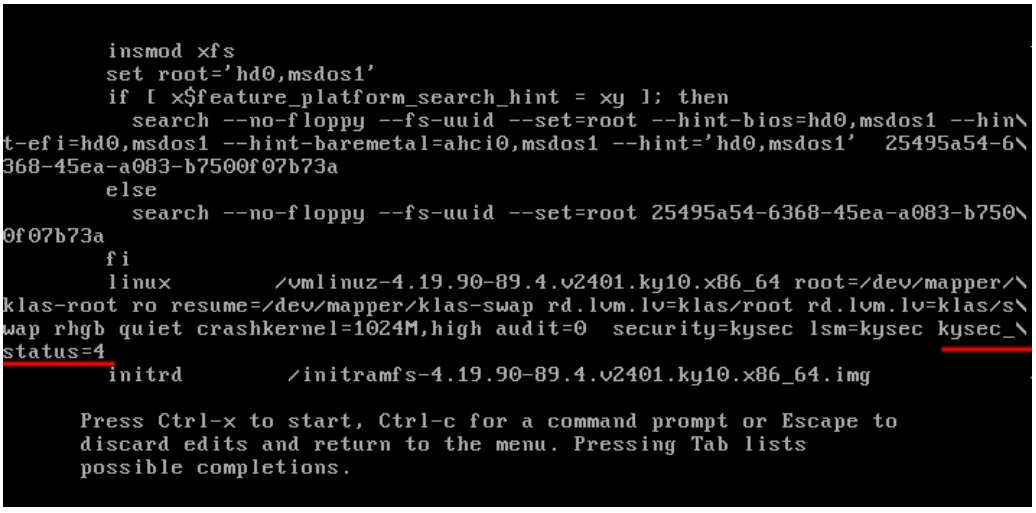
按下 e 键后，会进入以下界面，提示输入 grub 的用户和密码：

输入用户和密码后按回车键登录，会进入以下 grub 配置界面：



在此界面中按上下左右方向键移动光标，退格键删除文字。如果不小心改错，可以按 ESC 返回到进入 grub 编辑界面之前（按“e”键之前），想再次编辑可按“e”键然后输入用户名和密码。

在此界面配置中，在 linux 这一行的最后添加参数来改变 KySEC 安全机制的运行状态：



在划线处添加 kysec_status=4，本次开机 KySEC 安全机制状态会设置为软模式。

在划线处添加 kysec_status=2，本次开机 KySEC 安全机制状态会设置为强制

模式。

添加参数后，按 F10 或者“Ctrl + x”以当前 grub 参数继续开机。

7.2. 执行控制

银河麒麟高级服务器操作系统 V10 SP3 2403 添加了对文件执行、模块加载和共享库使用的控制，分为系统文件、第三方应用程序，其中只允许具有合法标记的文件执行，任何网络下载、拷贝等外来软件均被禁止执行。本来可运行的可执行文件在删除、修改、拷贝、移动、重命名等操作后，会被禁止执行。

7.2.1 开启执行控制安全状态

以 root 权限执行

```
setstatus kysec -f exectl -c on
```

7.2.2 关闭执行控制安全状态

以 root 权限执行

```
setstatus kysec -f exectl -c off
```

7.2.3 添加可执行程序的执行权限

以 root 权限执行

```
kysec_set exectl -v verified -f 文件路径
```

7.2.4 取消可执行程序的执行权限

以 root 权限执行

```
kysec_set exectl -c -f 文件路径
```

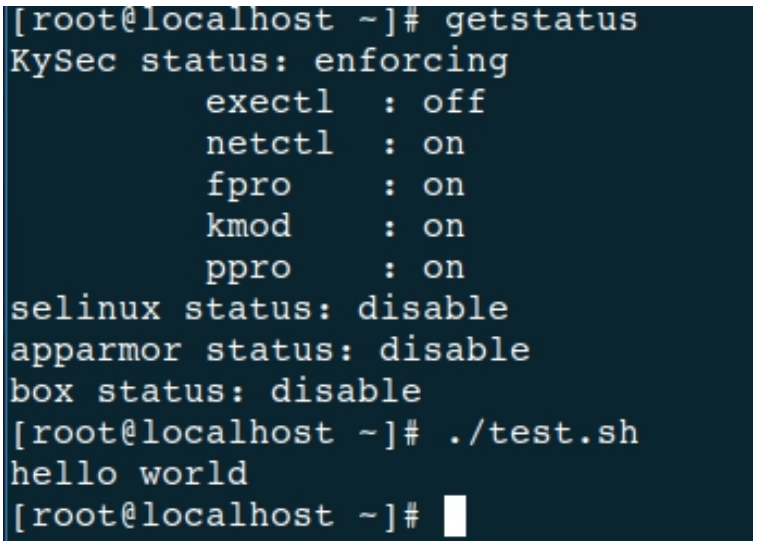
7.2.5 举例

若要验证执行控制的状态，可先创建 test.sh，填入以下内容：

```
#!/bin/bash
echo “hello world”
```

并以 root 权限执行 chmod +x test.sh。

开启前，getstatus 可看到 exectl 的状态为 off，且 test.sh 可以执行



使用 root 权限执行 setstatus kysec –f exectl –c on 开启执行控制，执行后 exectl 的状态为 on，test.sh 不可执行。

关闭前，getstatus 可看到 exectl 状态为 on，且 test.sh 不可执行。

```
[root@localhost ~]# getstatus
KySec status: enforcing
        exectl  : on
        netctl  : on
        fpro    : on
        kmod    : on
        ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]# ./test.sh
bash: ./test.sh: /bin/bash: 解释器错误：权限不够
[root@localhost ~]#
```

以 root 权限执行 `setstatus kysec -f execti -c off` 关闭执行控制，而后 execti 状态为 off，且 test.sh 可以执行。

```
[root@localhost ~]# setstatus kysec -f execti -c off
[root@localhost ~]# getstatus
KySec status: enforcing
        execti  : off
        netctl  : on
        fpro    : on
        kmod    : on
        ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]# ./test.sh
hello world
[root@localhost ~]#
```

在开启执行控制的情况下，getstatus 能看到 execti 状态为 on，执行 `kysec_get execti -f ./test.sh` 可看到 test.sh 的执行控制标记为 unknown，且 test.sh 不可执行。

```
[root@localhost ~]# getstatus
KySec status: enforcing
        exectl  : on
        netctl  : on
        fpro    : on
        kmod    : on
        ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]# kysec_get exectl -f ./test.sh
exectl: ./test.sh: unknown
[root@localhost ~]# ./test.sh
bash: ./test.sh: /bin/bash: 解释器错误：权限不够
[root@localhost ~]#
```

以 root 权限执行 kysec_set exectl −v verified −f ./test.sh 添加执行权限，而后通过 kysec_get exectl −f ./test.sh 可看到 test.sh 的执行控制标记为 verified，此时 test.sh 可以执行。

```
[root@localhost ~]# kysec_set exectl -v verified -f ./test.sh
[root@localhost ~]# kysec_get exectl -f ./test.sh
exectl: ./test.sh: verified
[root@localhost ~]# ./test.sh
hello world
[root@localhost ~]#
```

在开启执行控制的情况下，getstatus 能看到 exectl 状态为 on，执行 kysec_get exectl -f ./test.sh 可看到 test.sh 的执行控制标记为 verified，且 test.sh 可以执行。

```
[root@localhost ~]# getstatus
KySec status: enforcing
        execctl  : on
        netctl   : on
        fpro     : on
        kmod     : on
        ppro     : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]# kysec_get execctl -f ./test.sh
execctl: ./test.sh: verified
[root@localhost ~]# ./test.sh
hello world
[root@localhost ~]#
```

执行 kysec_set execctl −c −f ./test.s 命令可取消 test.sh 的执行权限，而后 kysec_get execctl −f ./test.sh 可看到 test.sh 的执行权限标记为 unknown，且不可以执行。

```
[root@localhost ~]# kysec_set execctl -c -f ./test.sh
[root@localhost ~]# kysec_get execctl -f ./test.sh
execctl: ./test.sh: unknown
[root@localhost ~]# ./test.sh
bash: ./test.sh: /bin/bash: 解释器错误：权限不够
[root@localhost ~]#
```

7.3. 文件保护功能

文件保护功能开启时，受保护的文件不能被修改、重命名、删除。这样能保护系统重要的配置文件不被误操作修改。

7.3.1 开启文件保护

以 root 权限执行

```
setstatus kysec −f fpro −c on
```

7.3.2 关闭文件保护

以 root 权限执行

```
setstatus kysec -f fpro -c off
```

7.3.3 添加文件保护

以 root 权限执行

```
kysec_set fpro -v readonly -f 文件路径
```

7.3.4 取消文件保护

以 root 权限执行

```
kysec_set fpro -c -f 文件路径
```

7.3.5 举例

给文件设置文件保护的标记：

```
#sudo kysec_set fpro -v readonly -f /tmp/testfile
```

给文件移除文件保护的标记：

```
#sudo kysec_set fpro -c -f /tmp/testfile
```

给文件/tmp/testfile 设置文件保护的标记后，则禁止修改,重命名,删除/tmp/testfile 文件。如下操作则会提示失败。

```
#echo “test1” >> /tmp/testfile
#mv /tmp/testfile /tmp/abc
#rm -f /tmp/testfile
```

```
[root@localhost ~]# touch /tmp/test.txt
[root@localhost ~]# kysec_set fpro -v readonly -f /tmp/test.txt
[root@localhost ~]# rm /tmp/test.txt
rm: 是否删除普通空文件 '/tmp/test.txt'? y
rm: 无法删除 '/tmp/test.txt': 权限不够
[root@localhost ~]# echo "123" > /tmp/test.txt
bash: /tmp/test.txt: 权限不够
[root@localhost ~]# kysec_set fpro -c -f /tmp/test.txt
[root@localhost ~]# echo "123" > /tmp/test.txt
[root@localhost ~]# cat /tmp/test.txt
123
[root@localhost ~]# rm /tmp/test.txt
rm: 是否删除普通文件 '/tmp/test.txt'? y
[root@localhost ~]# ls /tmp/test.txt
ls: 无法访问 '/tmp/test.txt': 没有那个文件或目录
[root@localhost ~]#
```

7.4. 模块防卸载

7.4.1 开启模块防卸载

以 root 权限执行

```
setstatus kysec –f kmod –c on
```

7.4.2 关闭模块防卸载

以 root 权限执行

```
setstatus kysec -f kmod -c off
```

7.4.3 保护模块防卸载

以 root 权限执行

```
kysec_set kmod –v deny –f 内核模块路径
```

7.4.4 取消保护模块防卸载

以 root 权限执行

```
kysec_set kmod –v allow –f 内核模块路径
```

7.4.5 举例

使用 getstatus 命令可用查看模块防卸载功能是否开启。

当 kmod 为 off 时，模块防卸载功能处于关闭状态：

```
[root@localhost ~]# getstatus
KySec status: enforcing
        execTl  : on
        netctl  : on
        fpro    : on
        kmod    : off
        ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]#
```

当 kmod 为 on 时，模块防卸载功能处于开启状态：

```
[root@localhost ~]# getstatus
KySec status: enforcing
        execTl  : on
        netctl  : on
        fpro    : on
        kmod    : on
        ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]#
```

使用 `setstatus kysec -f kmod -c on` 命令可开启模块防卸载功能：

```
[root@localhost ~]# getstatus
KySec status: enforcing
         exectl  : on
         netctl  : on
         fpro    : on
         kmod    : off
         ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]# setstatus kysec -f kmod -c on
[root@localhost ~]# getstatus
KySec status: enforcing
         exectl  : on
         netctl  : on
         fpro    : on
         kmod    : on
         ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]#
```

用 setstatus kysec –f kmod –c off 命令可关闭模块防卸载功能：

```
[root@localhost ~]# getstatus
KySec status: enforcing
         exectl  : on
         netctl  : on
         fpro    : on
         kmod    : on
         ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]# setstatus kysec -f kmod -c off
[root@localhost ~]# getstatus
KySec status: enforcing
         exectl  : on
         netctl  : on
         fpro    : on
         kmod    : off
         ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]#
```

以下以 adfs.ko 为例，介绍模块防卸载功能的表现，以及如何设置某个内核模块的是否允许防卸载。

首先保证模块防卸载功能处于开启状态（kmod 的值为 on）

加载 adfs 模块：以 root 权限执行以下命令：

```
[root@localhost ~]# insmod /usr/lib/modules/4.19.90-89.4.v2401.ky10.x86_64/ker
nel/fs/adfs/adfs.ko.xz
[root@localhost ~]# lsmod | grep adfs
adfs                   32768  0
[root@localhost ~]#
```

lsmod 命令能看到 adfs 模块名，证明 adfs 模块已加载。

通过 kysec_set 命令设置禁止卸载 adfs 模块后，rmmod 命令无法卸载 adfs 模块：

```
[root@localhost ~]# kysec_set kmod -v deny -f /usr/lib/modules/4.19.90-89.4.v2
401.ky10.x86_64/kernel/fs/adfs/adfs.ko.xz
[root@localhost ~]# rmmod adfs
rmmod: ERROR: could not remove 'adfs': Operation not permitted
rmmod: ERROR: could not remove module adfs: Operation not permitted
[root@localhost ~]#
```

通过 kysec_set 命令设置允许卸载 adfs 模块后，rmmod 命令可以卸载 adfs 模块：

```
[root@localhost ~]# kysec_set kmod -v allow -f /usr/lib/modules/4.19.90-89.4.v
2401.ky10.x86_64/kernel/fs/adfs/adfs.ko.xz
[root@localhost ~]# rmmod adfs
[root@localhost ~]# lsmod | grep adfs
[root@localhost ~]#
```

卸载后 lsmod 命令看不到 adfs 模块名。

7.5. 进程防杀功能

进程防杀死功能会保护正常运行的进程，防止被通过 kill 命令等方式杀死。

进程防杀死功能除了单独开关以外，需要将想保护的进程添加到防杀列表中。

7.5.1 开启进程防杀

以 root 权限执行

```
setstatus kysec -f ppro -c on
```

7.5.2 关闭进程防杀

以 root 权限执行

```
setstatus kysec-f ppro -c off
```

7.5.3 保护进程防杀

以 root 权限执行

```
kysec_set fpro -v deny -f 文件路径
```

7.5.4 取消保护进程防杀

以 root 权限执

```
kysec_set fpro -v allow -f 文件路径
```

7.5.5 举例

使用 getstatus 命令可用查看模块防卸载功能是否开启。

当 ppro 状态为 off 时，进程防杀死功能处于关闭状态：

```
[root@localhost ~]# getstatus
KySec status: enforcing
        exectl  : on
        netctl  : on
        fpro    : on
        kmod    : on
        ppro    : off
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]#
```

当 ppro 状态为 on 时，进程防杀死功能处于开启状态：

```
[root@localhost ~]# getstatus
KySec status: enforcing
         exectl  : on
         netctl  : on
         fpro    : on
         kmod    : on
         ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]#
```

以 root 权限执行 `sestatust kysec –f ppro –c on` 命令可开启进程防杀死功能：

```
[root@localhost ~]# getstatus
KySec status: enforcing
         exectl  : on
         netctl  : on
         fpro    : on
         kmod    : on
         ppro    : off
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]# setstatus kysec -f ppro -c on
[root@localhost ~]# getstatus
KySec status: enforcing
         exectl  : on
         netctl  : on
         fpro    : on
         kmod    : on
         ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]#
```

以 root 权限执行 `setstatus kysec –f ppro –c off` 命令可关闭进程防杀死功能：

```
[root@localhost ~]# getstatus
KySec status: enforcing
        execctl  : on
        netctl   : on
        fpro     : on
        kmod     : on
        ppro     : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]# setstatus kysec -f ppro -c off
[root@localhost ~]# getstatus
KySec status: enforcing
        execctl  : on
        netctl   : on
        fpro     : on
        kmod     : on
        ppro     : off
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]#
```

当进程防杀死功能开启时，可以为可执行文件和脚本等添加进程防杀死，这里以/usr/bin/sleep 为例。

使用 kysec_set ppro –v deny –f 文件路径 命令可添加 sleep 命令进程防杀死：

```
[root@localhost ~]# kysec_set ppro -v deny -f /usr/bin/sleep
/usr/bin/sleep: 添加成功，该进程未启动
[root@localhost ~]#
```

新开一个终端，运行 sleep 10000，运行之后，尝试以 root 权限执行 kill 命令杀死运行起来的 sleep 进程，会无法杀死：

```
[root@localhost ~]# kysec_set ppro -v deny -f /usr/bin/sleep
/usr/bin/sleep: 添加成功，该进程未启动
[root@localhost ~]# ps -ef | grep sleep
root        2902    2783  0 13:41 pts/1    00:00:00 sleep 10000
root        2906    2696  0 13:41 pts/0    00:00:00 grep --color=auto sleep
[root@localhost ~]# sudo kill -9 2902
kill: 向 2902 发送信号失败：不允许的操作
[root@localhost ~]#
```

取消进程防杀死：使用 kysec_set ppro –v allow –f 文件路径 命令可取消进程防杀死，取消后 sleep 可以杀死：

```
[root@localhost ~]# kysec_set ppro -v allow -f /usr/bin/sleep
[root@localhost ~]# ps -ef | grep sleep
root        2902    2783  0 13:41 pts/1    00:00:00 sleep 10000
root        2929    2696  0 13:42 pts/0    00:00:00 grep --color=auto sleep
[root@localhost ~]# sudo kill -9 2902
[root@localhost ~]# ps -ef | grep sleep
root        2941    2696  0 13:43 pts/0    00:00:00 grep --color=auto sleep
[root@localhost ~]#
```

最后的 ps 命令看不到 sleep，说明 sleep 已被杀死。

使用 kysec_get ppro --list 查看当前已下发的所有进程防杀策略。

```
[root@localhost ~]# kysec_get ppro --list
pid:   833, 进程名称：kyseclogd, 进程路径：/usr/sbin/kyseclogd, 防杀死保护：
开
pid:  1569, 进程名称：kysec-sync-daemon, 进程路径：/usr/sbin/kysec-sync-daemo
n, 防杀死保护：开
pid:  2105, 进程名称：marco, 进程路径：/usr/bin/marco, 防杀死保护：开
pid:  2150, 进程名称：pulseaudio, 进程路径：/usr/bin/pulseaudio, 防杀死保护：
开
pid:  3107, 进程名称：sleep, 进程路径：/usr/bin/sleep, 防杀死保护：开
[root@localhost ~]#
```

7.6. 联网管控功能

联网管控提供了控制软件访问网络的能力。

7.6.1 开启联网管控

以 root 权限执行

```
setstatus kysec -f netctl -c on
```

7.6.2 关闭联网管控

以 root 权限执行

```
setstatus kysec -f netctl -c off
```

7.6.3 禁止软件联网

以 root 权限执行

```
kysec_set netctl -v deny -f  文件路径
```

7.6.4 允许软件联网

以 root 权限执行

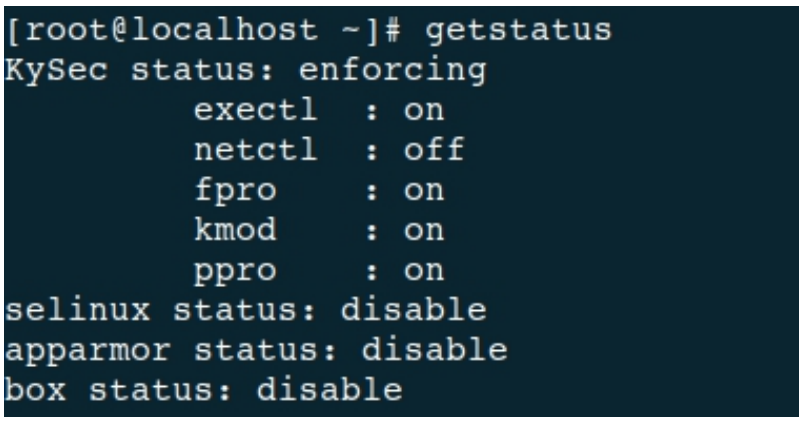
```
kysec_set netctl -v allow -f 文件路径
```

7.6.5 举例

通过 getstatus 查看联网管控功能是否开启。

当 netctl 为 on 时，联网管控开启：

当 netctl 为 off 时，联网管控关闭：



通过 `setstatus kysec -f netctl -c on` 开启联网管控：

```
[root@localhost ~]# getstatus
KySec status: enforcing
         exectl  : on
         netctl  : off
         fpro    : on
         kmod    : on
         ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]# setstatus kysec -f netctl -c on
[root@localhost ~]# getstatus
KySec status: enforcing
         exectl  : on
         netctl  : on
         fpro    : on
         kmod    : on
         ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]#
```

通过 setstatus kysec –f netctl –c off 关闭联网管控：

```
[root@localhost ~]# getstatus
KySec status: enforcing
         exectl  : on
         netctl  : on
         fpro    : on
         kmod    : on
         ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
[root@localhost ~]# setstatus kysec -f netctl -c off
[root@localhost ~]# getstatus
KySec status: enforcing
         exectl  : on
         netctl  : off
         fpro    : on
         kmod    : on
         ppro    : on
selinux status: disable
apparmor status: disable
box status: disable
```

通过 kysec_set netctl –v deny –f /usr/bin/curl 禁止 curl 联网：

```
[root@localhost ~]# kysec_set netctl -v deny -f /usr/bin/curl
[root@localhost ~]# curl kylinos.cn
curl: (6) Could not resolve host: kylinos.cn
```

通过 kysec_set netctl –v allow –f /usr/bin/curl 允许 curl 联网：

```
[root@localhost ~]# kysec_set netctl -v allow -f /usr/bin/curl
[root@localhost ~]# curl kylinos.cn
<html>
<head><title>301 Moved Permanently</title></head>
<body>
<center><h1>301 Moved Permanently</h1></center>
<hr><center>nginx</center>
</body>
</html>
```

通过 kysec_set netctl ––list 查看已下发的策略：

```
[root@localhost ~]# kysec_get netctl --list
/usr/bin/curl                    : 允许
[root@localhost ~]#
```

8. 三权分立机制

三权分立机制将传统意义上超级管理员 root 的权能进行划分，分别为 root(uid=0)、secadm(uid=600)和 auditadm(uid=700)分别作为超级用户的别名存在，它们分别对应 SELinux 三权分立角色中的一个角色，从而保持传统系统用户身份鉴别系统结构的同时，达到三权分立的目的。

8.1. 三权用户图形管理

当系统安全机制为三权分立机制,即安全状态为 strict 模式时。系统管理员、安全管理员、审计管理员在图形管理权限也不是不同的。如：

当系统安全机制为三权分立机制时，只有系统管理员具有以下应用使用权限：

- “分区管理功能”。该功能使系统中只有系统管理员可对系统磁盘进行分区管理。系统管理员可在图形界面下，点击开始菜单->所以程序->附

件->磁盘。系统管理员可以成功打开分区编辑器。

- “时间管理功能”。该功能使系统中只有系统管理员可对操作系统的时间进行设置，系统管理员可在图形界面下，点击开始菜单->控制面板->时间和日期。
- “系统重启和关机功能”。该功能使系统中只有系统管理员可对操作系统下发关机或重启指令，系统管理员可在图形界面下，点击开始菜单->关机。
- “用户管理功能”。该功能使系统中只有系统管理员可对用户进行管理，系统管理员可在图形界面下，点击开始菜单->控制面板->用户账户。
- “网络管理功能”。该功能使系统中只有系统管理员可对网络进行管理，当安全模式为 strict 时，点击开始菜单–>控制面板–>网络连接。
- “防火墙管理功能”。该功能使系统中只有系统管理员可对防火墙策略进行管理，系统管理员可在图形界面下，点击开始菜单–>系统工具–>防火墙。
- “系统监视器功能”。该功能使系统中只有系统管理员可对系统资源进行管理，系统管理员可在图形界面下，点击开始菜单–>系统工具–>系统监视器。
- “麒麟备份还原功能”。该功能使系统中只有系统管理员可对系统资源进行备份和还原，系统管理员可在图形界面下，点击开始菜单–>系统工具–>麒麟备份还原工具。
- “网络代理”。该功能使系统中只有系统管理员可对系统网络设置代理，系统管理员可在图形界面下，点击开始菜单–>控制面板–>网络代理。

当系统安全机制为三权分立机制时，只有审计管理员可查看日志信息，即审计管理员可在图形界面下，点击开始菜单–>系统工具–>日志查看器。打开“日志查看器”查看系统日志信息。

8.2. 三权用户命令集

表 8-2 三权用户命令集

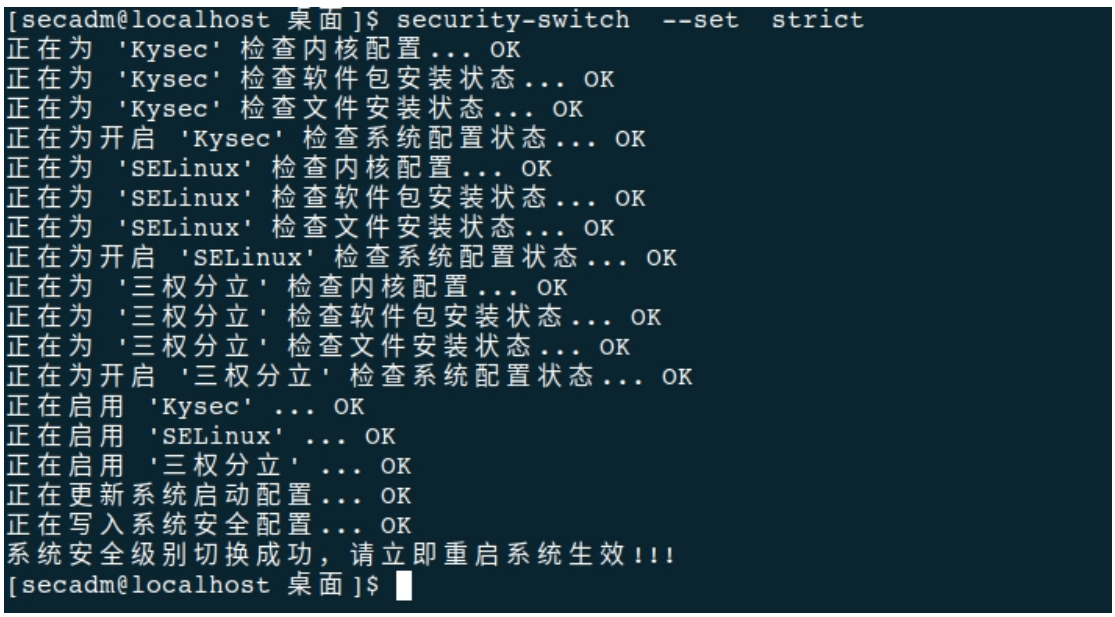
用户	命令	备注
系统管理员	reboot	系统重启命令
	init	管理系统运行模式
	ifconfig	网络接口管理（设置类的操作）
	rpm	软件包管理
	useradd	添加用户。如：useradd abc
	passwd	修改普通用户密码。如：passwd abc
	userdel	删除用户。如：userdel -r abc
	insmod	插入模块
	rmmod	删除模块
	iptables	防火墙相关
	system-config-printer	打印机配置
	mount	分区挂载（设置类的操作）
安全管理员	setenforce	开启和关闭 SELinux 的强制模式
	checkpolicy	将可读策略文件编译生成二进制策略文件，该文件和所在文件夹必须为 policy_src_t 类型。 如：checkpolicy -M policy.conf -o policy.24
	load_policy	如：load_policy -bq(服务器) 更换内核中

		的安全策略，保持使用当前的 Boolean 值
	chcon	修改文件和文件夹的安全上下文。如：chcon -t bin_t filename
	chcat	修改文件和文件夹的敏感级。如：chcat s0:c0.c255 filename
	fixfiles	检查修复文件安全上下文。如：fixfiles -F restore filename
	setfiles	初始化文件标记，检查标记正确性。如：setfiles /etc/selinux/targeted/contexts/files/file_conte xts /
	restorecon	恢复默认文件安全上下文 如：restorecon -r filename
	semanage	SELinux 管理工具命令 如： semanage user -l semanage login -l semanage user -m -R secadm_r secadm_u
	semodule	管理 SELinux 策略模块 如： semodule -l semodule -i modulename
	setsebool	修改 SELinux 中的 bool 值来修改策略 如：setsebool xdm_sysadm_login = 1
	audit2allow	显示访问被拒绝后应添加的允许规则 如：audit2allow -a audit2allow -i /tmp/dmesg（将 dmesg 输出到/tmp/dmesg 中）
	audit2why	根据访问被拒绝的审计信息显示被拒绝的原

		因。如：audit2why < /tmp/audit.log （将 /var/log/audit/audit.log 拷贝到/tmp 下）
审计管理员	aureport	根据审计信息文件统计登录情况，avc，pid，file，event，config 等众多信息。如： aureport -au aureport -a aureport -p
	ausearch	搜索审计文件中的信息。如： ausearch -gi 0
	auditd	启停审计服务。如： systemctl start/stop/restart auditd
	auditctl	显示服务状态，审计规则相关 如：auditctl -s

8.3. 启用三权分立

1 ）执行下述命令开启三权分立安全功能：




```
security-switch --set strict
```

2）重启系统，使得修改生效

3）通过 security-switch --get 命令查看当前修改后的增强安全状态中“三权分立”为启用，命令如下：

```
[root@localhost 桌面]#
[root@localhost 桌面]# security-switch  --get

当前安全级别 ：安全优先(strict)
-----------------------------------------------------
  安全模块      |     当前状态      |     默认状态
-----------------------------------------------------
  Kysec          启用(Enforcing)    启用(Enforcing)
  SELinux        启用(Enforcing)    启用(Enforcing)
  三权分立       启用               启用
[root@localhost 桌面]#
```

8.4. 关闭三权分立

采用切换其他安全级别可以关闭三权分立安全功能：使用 secadm 安全管理员进行切换模式操作。

切换为 default 级别：

default 级别，kysec 安全机制开启状态，selinux 安全和三权分立机制均为关闭状态。

```
[secadm@localhost 桌面]$ security-switch  --get

当前安全级别 ：安全优先(strict)
-----------------------------------------------------
  安全模块      |     当前状态      |     默认状态
-----------------------------------------------------
  Kysec          启用(Enforcing)    启用(Enforcing)
  SELinux        启用(Enforcing)    启用(Enforcing)
  三权分立       启用               启用
[secadm@localhost 桌面]$ security-switch  --set default
正在为 'Kysec' 检查内核配置... OK
正在为 'Kysec' 检查软件包安装状态... OK
正在为 'Kysec' 检查文件安装状态... OK
正在为开启 'Kysec' 检查系统配置状态... OK
正在为关闭 'SELinux' 检查系统配置状态... OK
正在为关闭 '三权分立' 检查系统配置状态... OK
正在启用 'Kysec' ... OK
正在禁用 'SELinux' ... OK
正在禁用 '三权分立' ... OK
正在更新系统启动配置... OK
正在写入系统安全配置... OK
系统安全级别切换成功，请立即重启系统生效!!!
[secadm@localhost 桌面]$
```

切换 none 模式

none 模式，kysec 安全机制、SELinux、三权分立均是关闭状态。

```
[secadm@localhost 桌面]$ security-switch  --get

当前安全级别 ：安全优先(strict)
-----------------------------------------------------
  安全模块      |      当前状态      |      默认状态
-----------------------------------------------------
  Kysec           启用(Enforcing)     启用(Enforcing)
  SELinux         启用(Enforcing)     启用(Enforcing)
  三权分立        启用                启用
[secadm@localhost 桌面]$ security-switch --set none
正在为关闭 'Kysec' 检查系统配置状态... OK
正在为关闭 'SELinux' 检查系统配置状态... OK
正在为关闭 '三权分立' 检查系统配置状态... OK
正在禁用 'Kysec' ... OK
正在禁用 'SELinux' ... OK
正在禁用 '三权分立' ... OK
正在更新系统启动配置... OK
正在写入系统安全配置... OK
系统安全级别切换成功，请立即重启系统生效!!!
[secadm@localhost 桌面]$
```

可通过开机时 grub 界面增加 kysec_3adm=0 参数关闭三权分立

```
        insmod xfs                                                      ↑
        set root='hd0,msdos1'
        if [ x$feature_platform_search_hint = xy ]; then
          search --no-floppy --fs-uuid --set=root --hint='hd0,msdos1'  1b475cc\
f-9bab-4485-8389-098fbd9f92d3
        else
          search --no-floppy --fs-uuid --set=root 1b475ccf-9bab-4485-8389-098f\
bd9f92d3
        fi
        linux        /vmlinuz-4.19.90-89.4.v2401.ky10.x86_64 root=/dev/mapper/\
klas-root ro resume=/dev/mapper/klas-swap rd.lvm.lv=klas/root rd.lvm.lv=klas/s\
wap rhgb quiet crashkernel=1024M,high audit=0  security=kysec,selinux lsm=kyse\
c,selinux kysec_3adm=0
        initrd        /initramfs-4.19.90-89.4.v2401.ky10.x86_64.img


      Press Ctrl-x to start, Ctrl-c for a command prompt or Escape to
      discard edits and return to the menu. Pressing Tab lists
      possible completions.
```

9. 安全加固

安全加固提供安全服务、内核参数、安全网络、系统命令、系统审计、系统设置、潜在危险、文件权限、风险账户、磁盘检查、密码强度、账户锁定、系统安全、系统维护、资源分配等多维度的扫描与一键加固，及时发现并处理系统安全隐患。

安全服务：确保所有安全相关的服务（如防火墙、入侵检测系统、防病毒软件等）都已安装、配置并正常运行。还有禁用不必要的服务、限制远程访问等。

内核参数：调整操作系统的内核参数，以增强系统的安全性。例如，禁用不必要的网络功能、限制内核模块加载等。

安全网络：实施网络隔离和访问控制策略，限制不必要的网络流量和访问。

系统命令：限制对系统命令的访问，确保只有授权用户才能执行关键命令。记录命令执行的日志，以便审计和追踪潜在的安全事件。

系统审计：启用系统审计功能，记录关键的系统事件和用户活动。

系统设置：优化系统设置，减少安全漏洞。

潜在危险：识别并消除系统中的潜在危险，如空链接文件，不安全组件，危险文件等。

文件权限：严格控制文件和目录的访问权限，确保只有授权用户才能访问敏

感数据。

风险账户：识别并管理风险账户，如弱密码账户、过期账户等。

磁盘检查：对磁盘进行健康检查和使用率进行检查，防止因磁盘故障或者磁盘满溢导致的数据丢失或损坏。

密码强度：强制实施强密码策略，要求用户设置复杂且不易猜测的密码。定期更换系统密码和关键账户的密码。

账户锁定：设置账户锁定策略，在多次错误登录后自动锁定账户。

系统安全：开启安全防护功能，如管理员分权，进程防杀，联网管控等功能。启用系统自带的安全功能和防护机制，如防火墙，selinux 等。

系统维护：仅允许系统管理员进入维护模式。

资源分配：合理分配系统资源，避免资源过度消耗或不足。

详细内容请参考《银河麒麟安全中心用户手册》。

10. 数据隔离保护机制

银河麒麟高级服务器操作系统 V10 SP3 2403 实现用户私有数据的机密性隔离存储；并基于该保护机制，研制了文件保护箱功能，确保每个用户的文件保护箱相互隔离，即使非法用户获得管理员权限也无法获取其他用户的私有数据。

10.1. 开启文件保护箱机制

10.1.1 系统已安装安全组件下启用文件保护箱

执行以下命令，然后重启操作系统

```
setstatus box -s enable
```

10.1.2 仅安装保护箱组件的开启方式

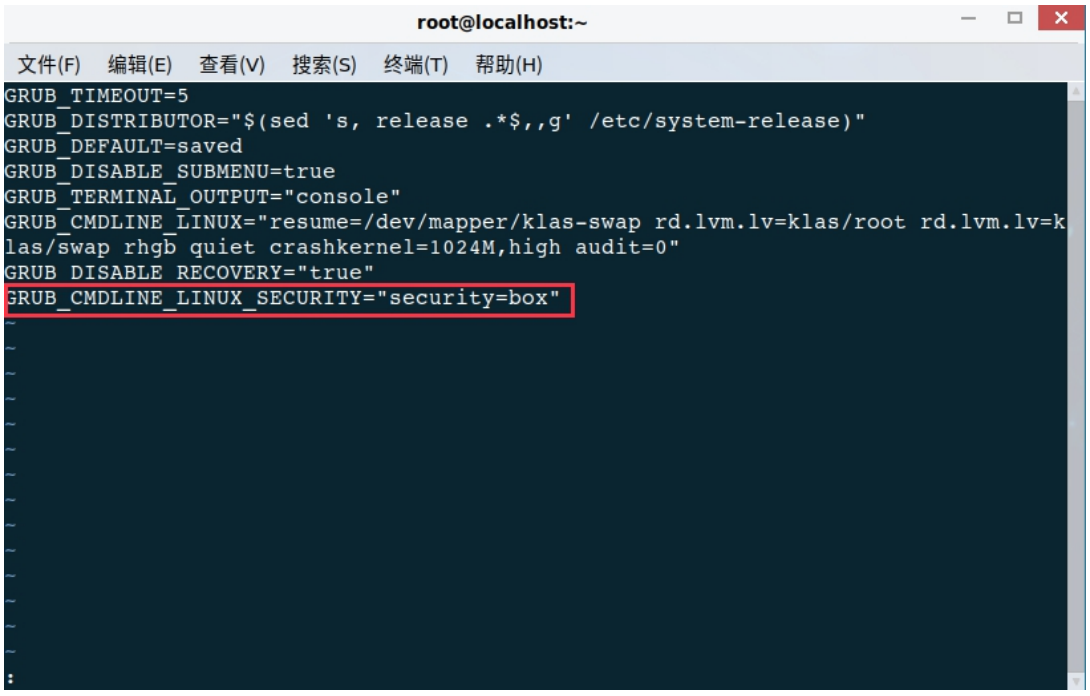
setstatus 命令不存在时，先查看保护箱组件安装状态，

rpm –qa|grep box

```
[root@localhost ~]# rpm -qa | grep box
libbox1-4.0.3-1.se.04.ky10.x86_64
box-utils-4.0.3-1.se.04.ky10.x86_64
box-manager-3.1.3-1.se.09.ky10.x86_64
```

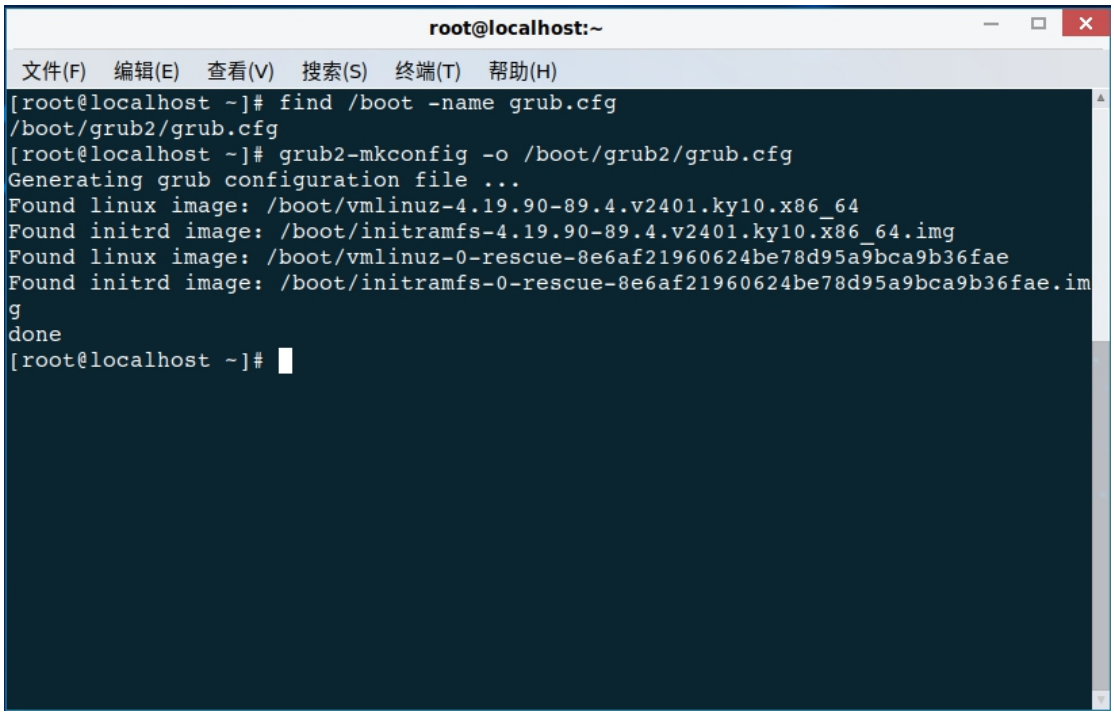
如果未安装保护箱组件，通过命令“yum install box-utils box-manager libbox1”进行安装。安装完成后，通过如下方式进行开启。

1) 编辑 /etc/default/grub 文件，在文件的最后增加一行 GRUB_CMDLINE_LINUX_SECURITY="security=box"，如下图所示，然后保存文件



2) 通过“find /boot –name grub.cfg”命令查找系统中 grub.cfg 文件路径，找到后，执行命令 grub2–mkconfig –o /boot/xxx/grub.cfg（找到的 grub.cfg 的所在路径，如果找到多个不同文件路径的 grub.cfg 文件，grub2–mkconfig 命令要执行多

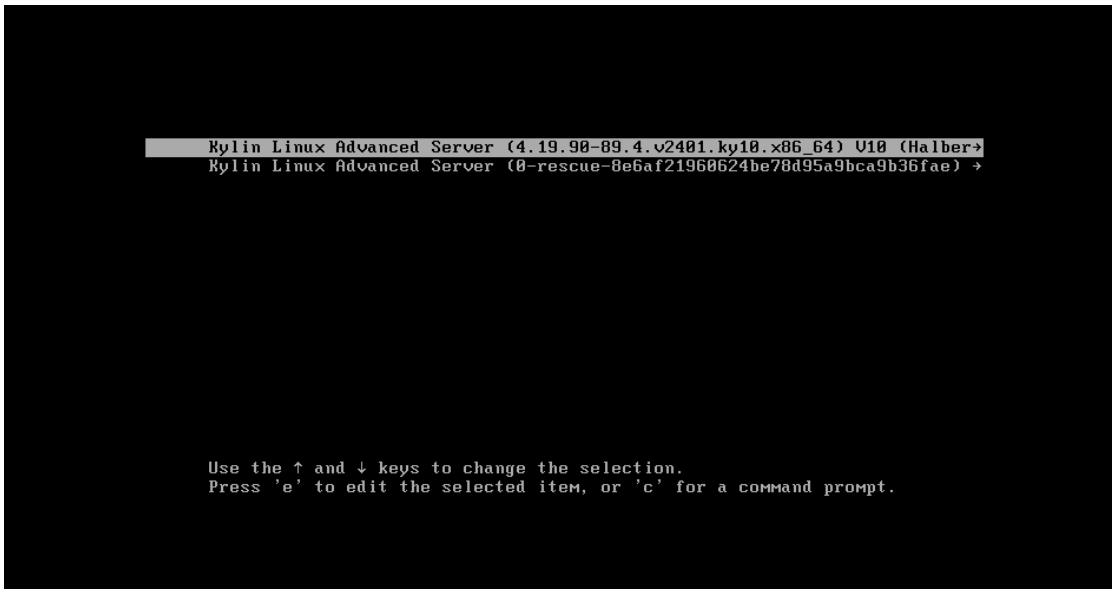
次，每次对应一个文件路径），如下图所示：



3) 重启操作系统

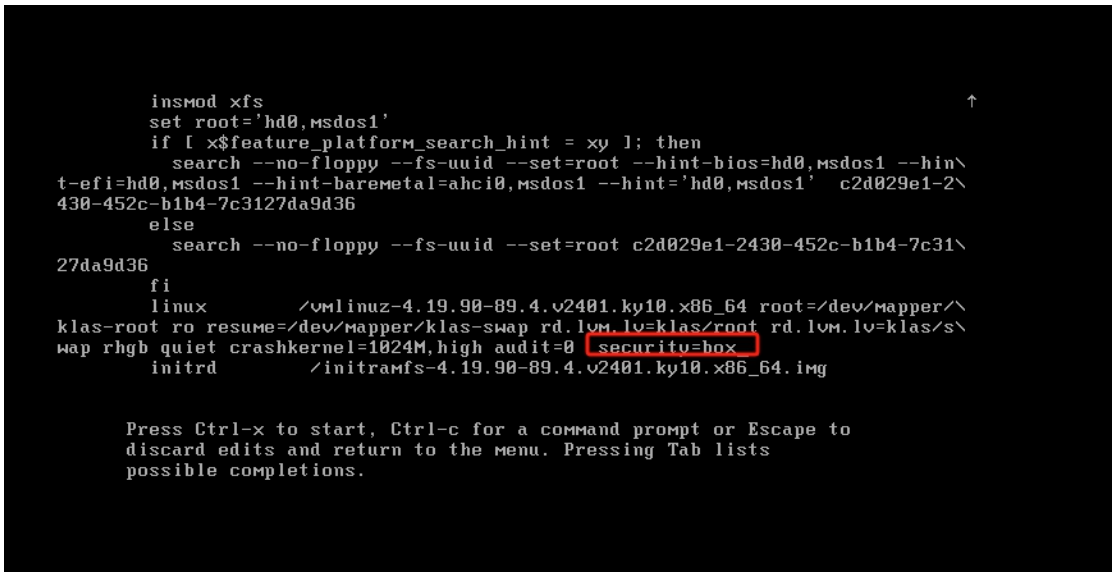
10.1.3 在引导时开启文件保护箱

1）进入如下引导界面的时候，按 e 进入编辑界面



2）在编辑界面，找到 linux 开头的以下行，在最后加上“ security=box”，

然后按 Ctrl+x 进入系统即可



10.2. 关闭文件保护箱

10.2.1.系统已安装安全组件下的关闭文件保护箱

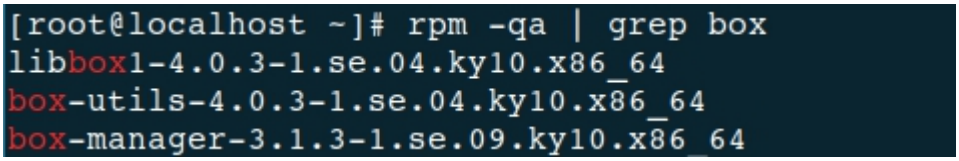
执行以下命令，然后重启操作系统

```
setstatus box -s disable
```

10.2.2.仅安装保护箱组件的关闭方式

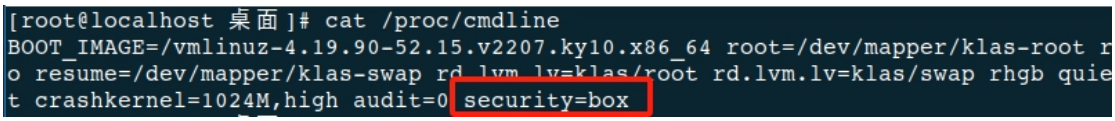
setstatus 命令不存在时，先查看保护箱组件安装状态，

rpm –qa|grep box



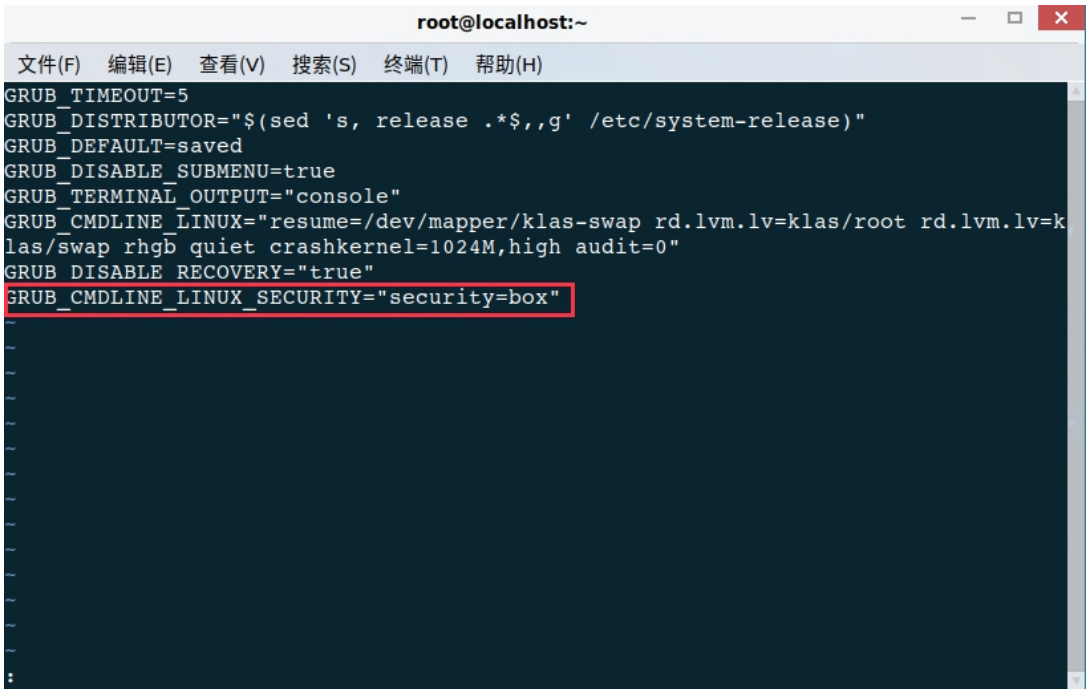
查看系统启动参数

cat /proc/cmdline

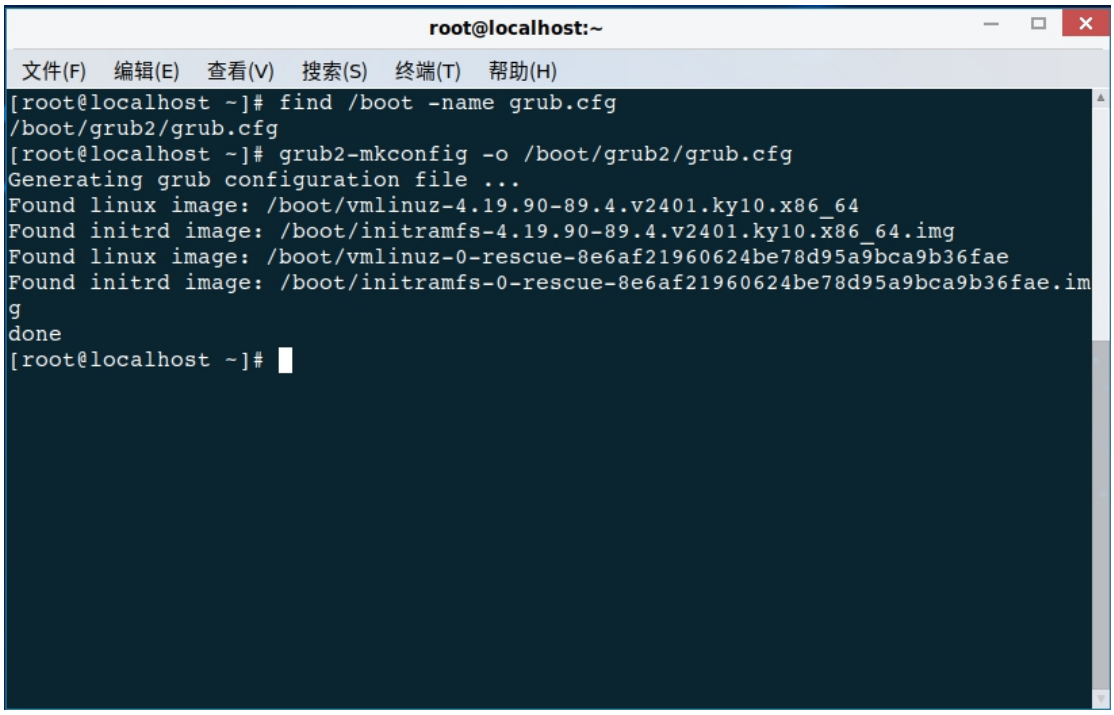


启动参数包含”security=box”，文件保护箱处于启用状态，通过下面操作说明可以进行关闭。

1) 编辑 /etc/default/grub 文件，删除文件中以“GRUB_CMDLINE_LINUX_SECURITY=”开头的行，如下图所示，然后保存文件



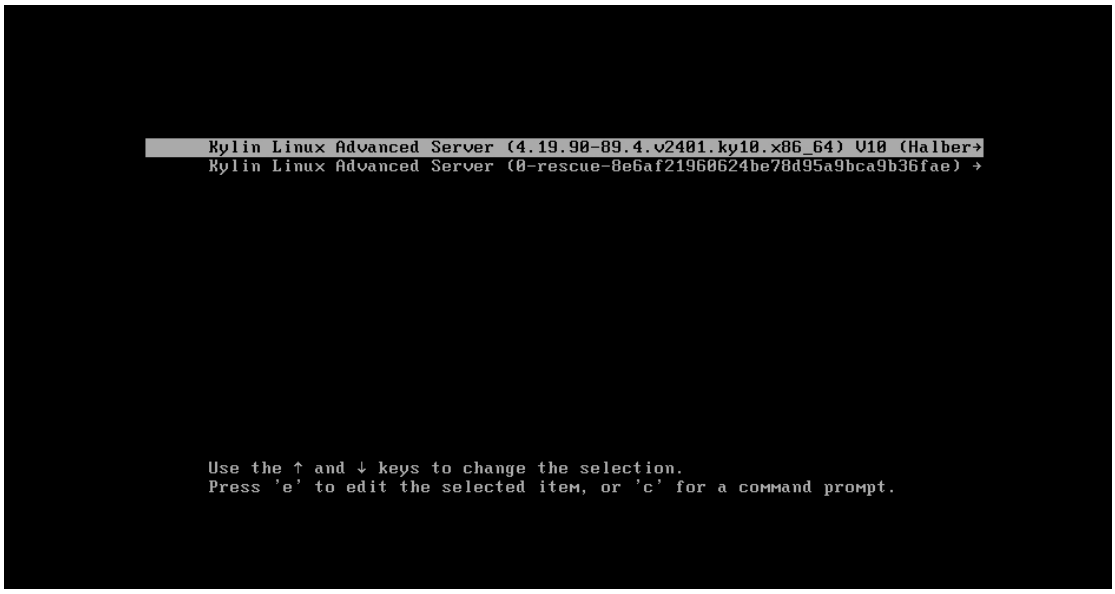
2) 通过“find /boot –name grub.cfg”命令查找系统中 grub.cfg 文件路径，找到后，执行命令 grub2–mkconfig –o /boot/xxx/grub.cfg（找到的 grub.cfg 的所在路径，如果找到多个不同文件路径的 grub.cfg 文件，grub2–mkconfig 命令要执行多次，每次对应一个文件路径），如下图所示：



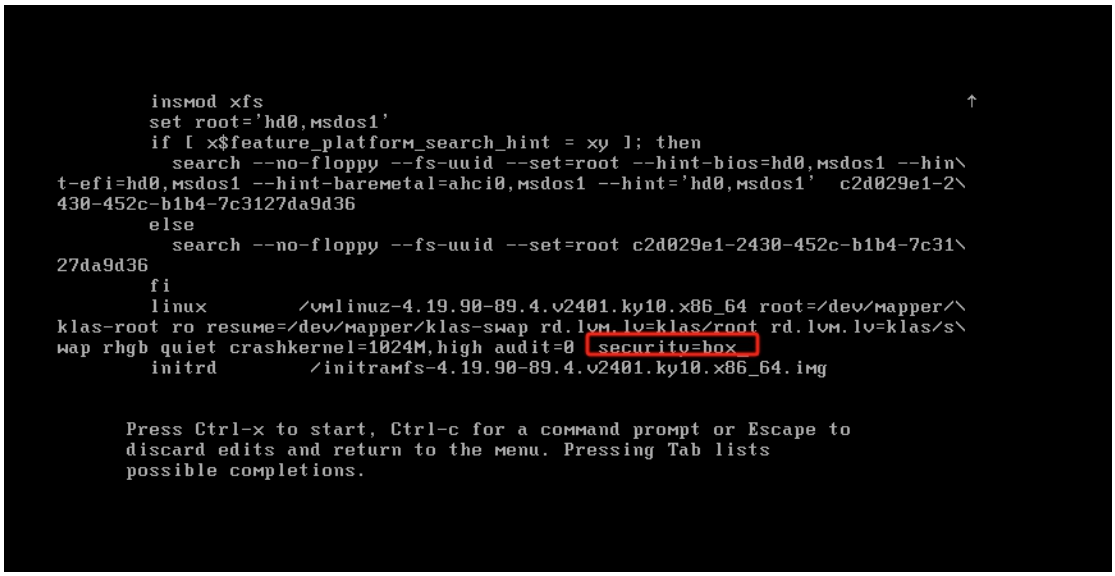
3）重启操作系统

10.2.3.在引导时关闭文件保护箱

1）进入如下引导界面的时候，按 e 进入编辑界面



2）在编辑界面，找到 linux 开头的以下行， 删除行尾的 “security=box”，然后按 Ctrl+x 进入系统即可



10.3. boxadm 的使用方法

详细内容请参考《银河麒麟文件保护箱用户手册》。

11. USB 设备管控

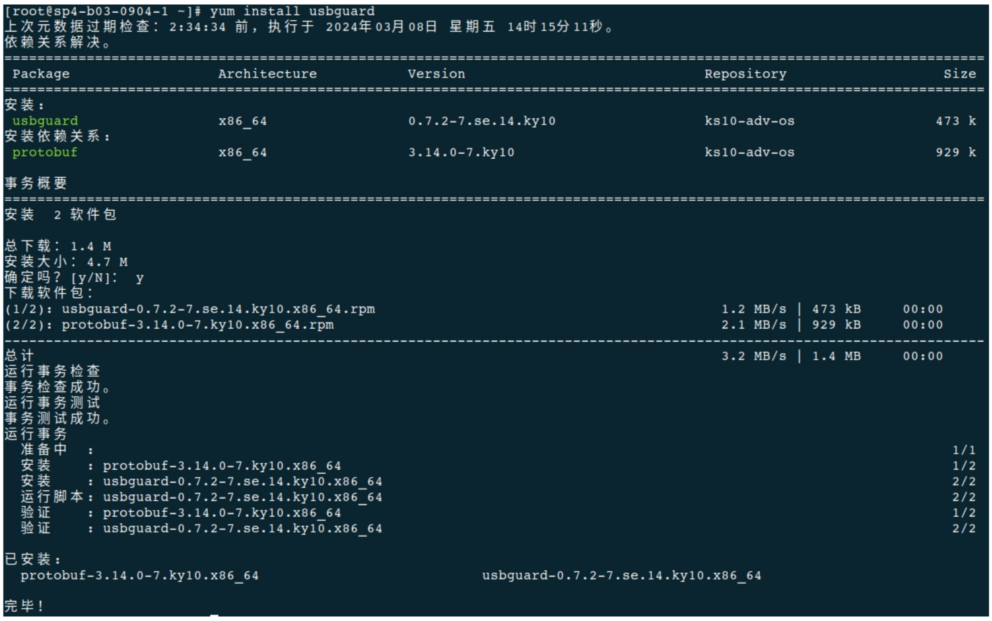
USB 设备可能会加载间谍软件、恶意软件或特洛伊木马，这可能会窃取您的数据或破坏您的系统。您可以使用麒麟 USB 设备管控工具来防止此类 USB 攻击。

11.1 用法

1）查看系统是否已安装麒麟 USB 设备管控软件包:

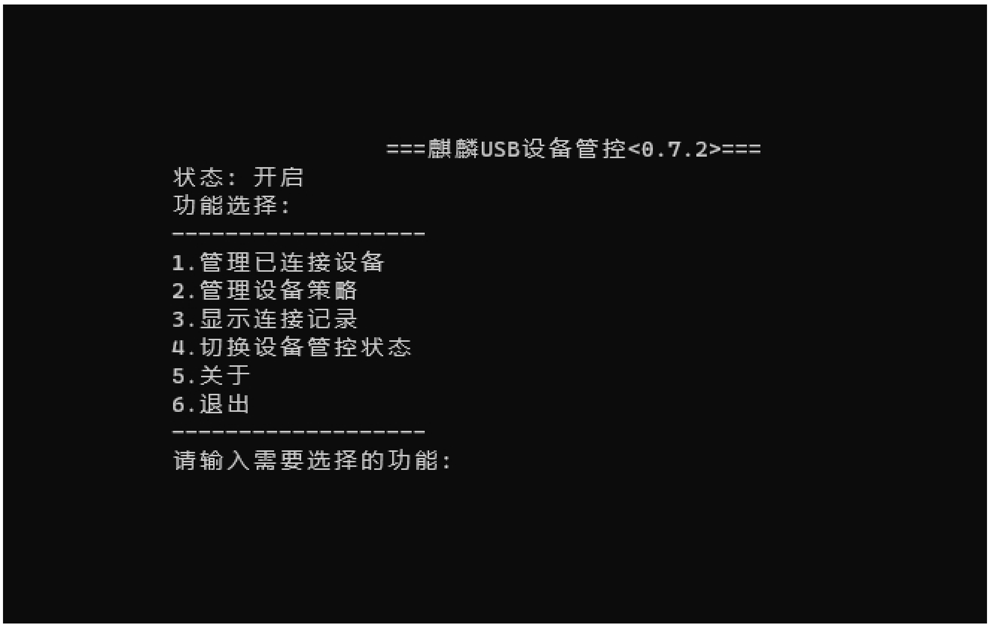
```
[root@sp4-b03-0904-1 ~]# rpm -qa usbguard
usbguard-0.7.2-7.se.14.ky10.x86_64
```

2）若未安装程序软件包，需先安装:



```
[root@sp4-b03-0904-1 ~]# yum install usbguard
上次元数据过期检查：2:34:34 前，执行于 2024年03月08日 星期五 14时15分11秒。
依赖关系解决。
==========================================================================================================
 Package                  Architecture          Version                              Repository       Size
==========================================================================================================
安装：
 usbguard                 x86_64                0.7.2-7.se.14.ky10                   ks10-adv-os     473 k
安装依赖关系：
 protobuf                 x86_64                3.14.0-7.ky10                        ks10-adv-os     929 k

事务概要
==========================================================================================================
安装  2 软件包

总下载：1.4 M
安装大小：4.7 M
确定吗？[y/N]： y
下载软件包：
(1/2): usbguard-0.7.2-7.se.14.ky10.x86_64.rpm                             1.2 MB/s | 473 kB     00:00
(2/2): protobuf-3.14.0-7.ky10.x86_64.rpm                                  2.1 MB/s | 929 kB     00:00
----------------------------------------------------------------------------------------------------------
总计                                                                      3.2 MB/s | 1.4 MB     00:00
运行事务检查
事务检查成功。
运行事务测试
事务测试成功。
运行事务
  准备中  :                                                                                          1/1
  安装    : protobuf-3.14.0-7.ky10.x86_64                                                            1/2
  安装    : usbguard-0.7.2-7.se.14.ky10.x86_64                                                       2/2
  运行脚本: usbguard-0.7.2-7.se.14.ky10.x86_64                                                       2/2
  验证    : protobuf-3.14.0-7.ky10.x86_64                                                            1/2
  验证    : usbguard-0.7.2-7.se.14.ky10.x86_64                                                       2/2

已安装：
  protobuf-3.14.0-7.ky10.x86_64                           usbguard-0.7.2-7.se.14.ky10.x86_64

完毕！
```

3）输入 kylin_usbguard 并回车，打开交互式工具



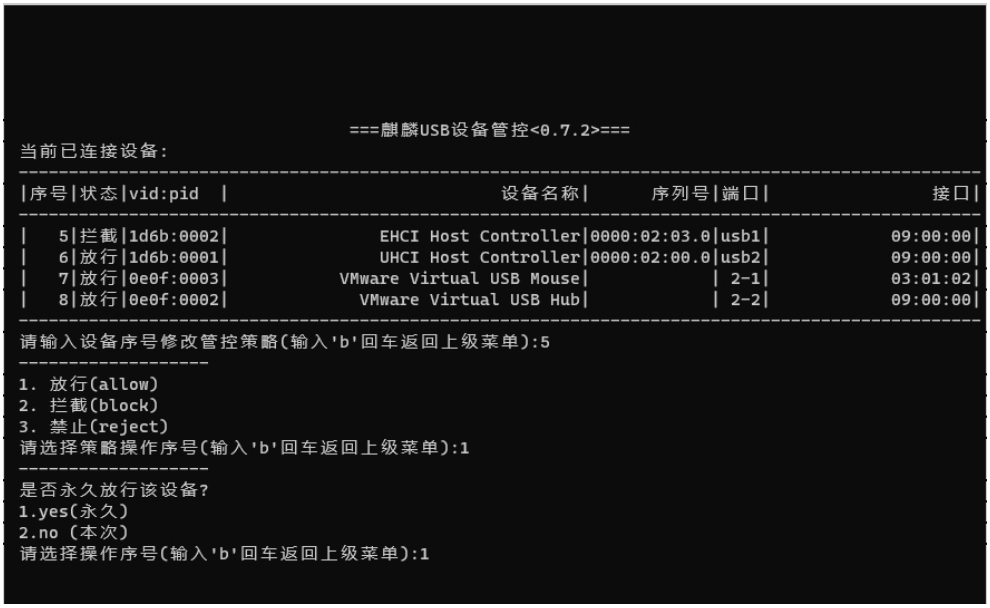
```
                ===麒麟USB设备管控<0.7.2>===
状态：开启
功能选择：
-------------------
1.管理已连接设备
2.管理设备策略
3.显示连接记录
4.切换设备管控状态
5.关于
6.退出
-------------------
请输入需要选择的功能：
```

11.2 示例

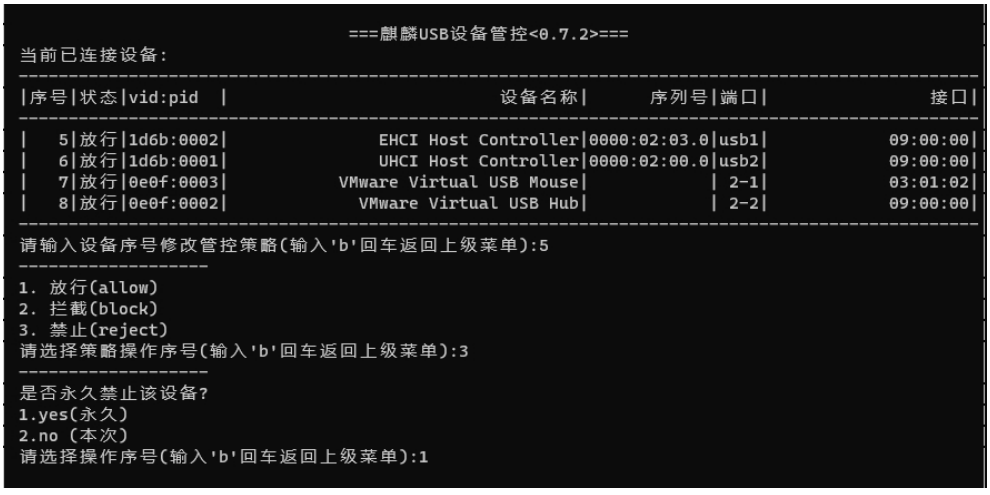
1）管理已连接设备

设置 USB 设备拦截

设置 USB 设备放行

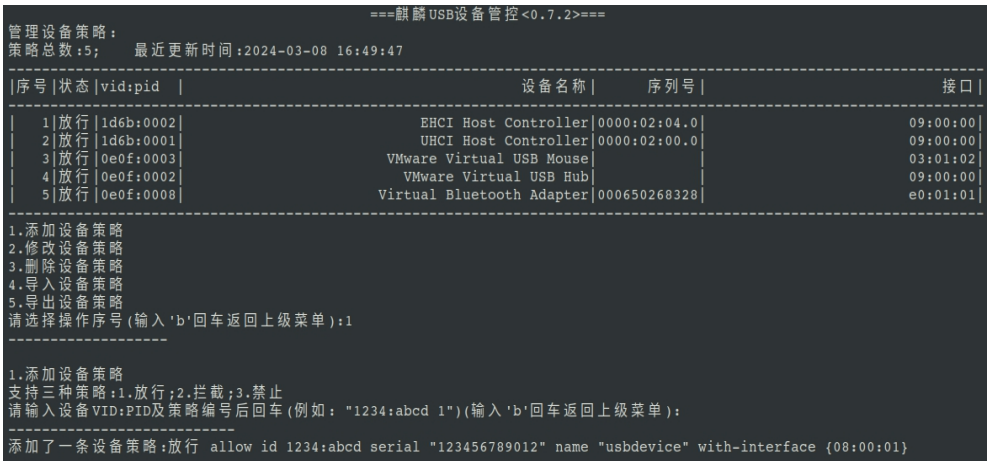


设置 USB 设备禁止

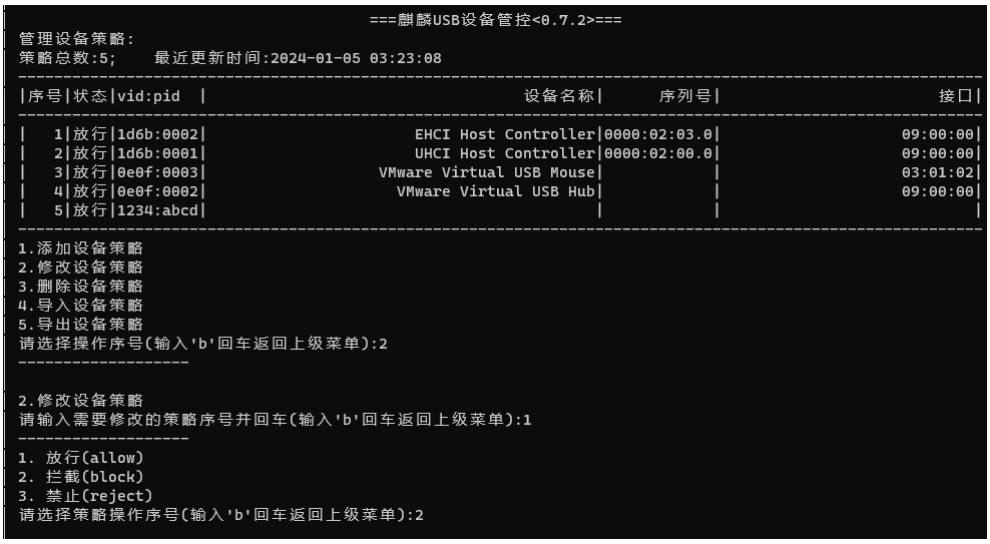


2）管理设备策略

添加策略

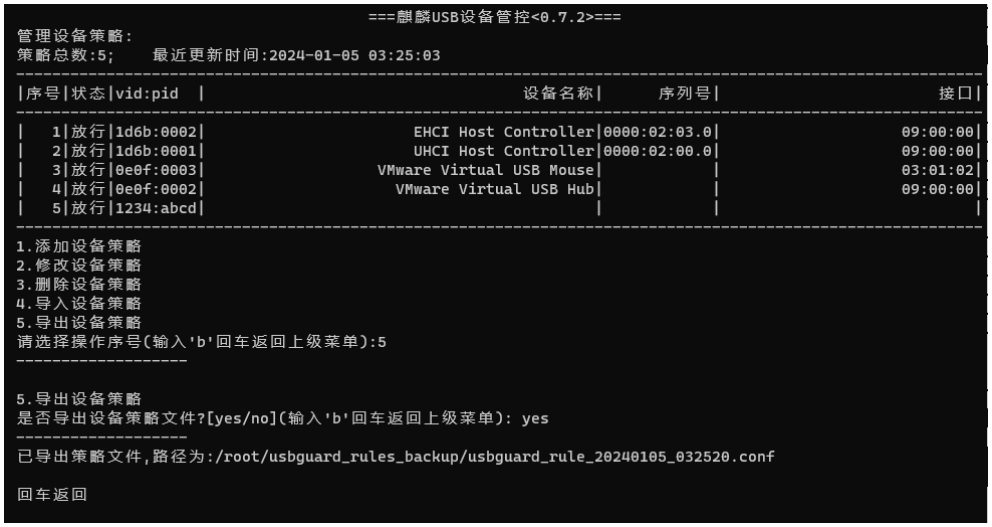


修改策略



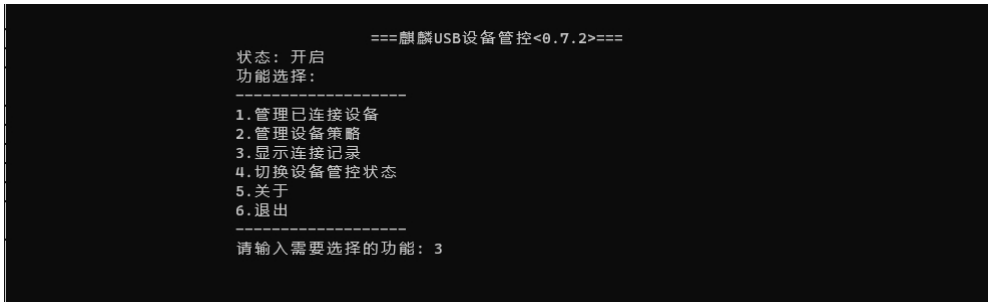
删除策略

导出策略



导入策略

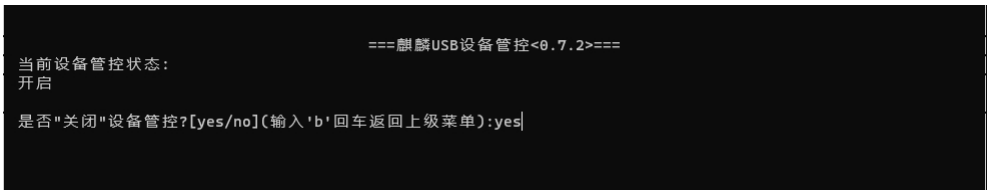
3）显示连接记录



显示连接记录

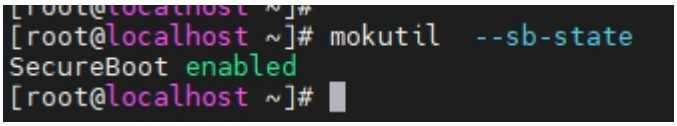
4）切换设备管控状态

关闭设备管控



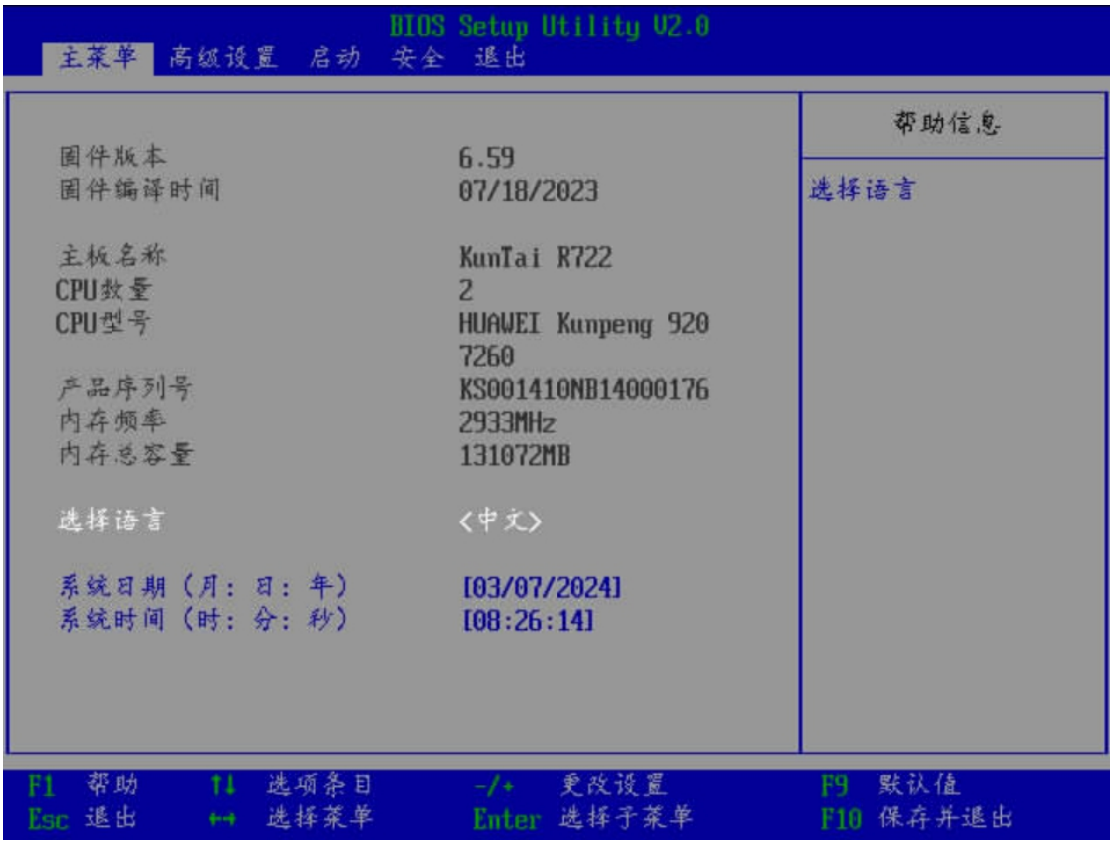
12. 安全启动

1. 安全启动在系统侧查看当前状态

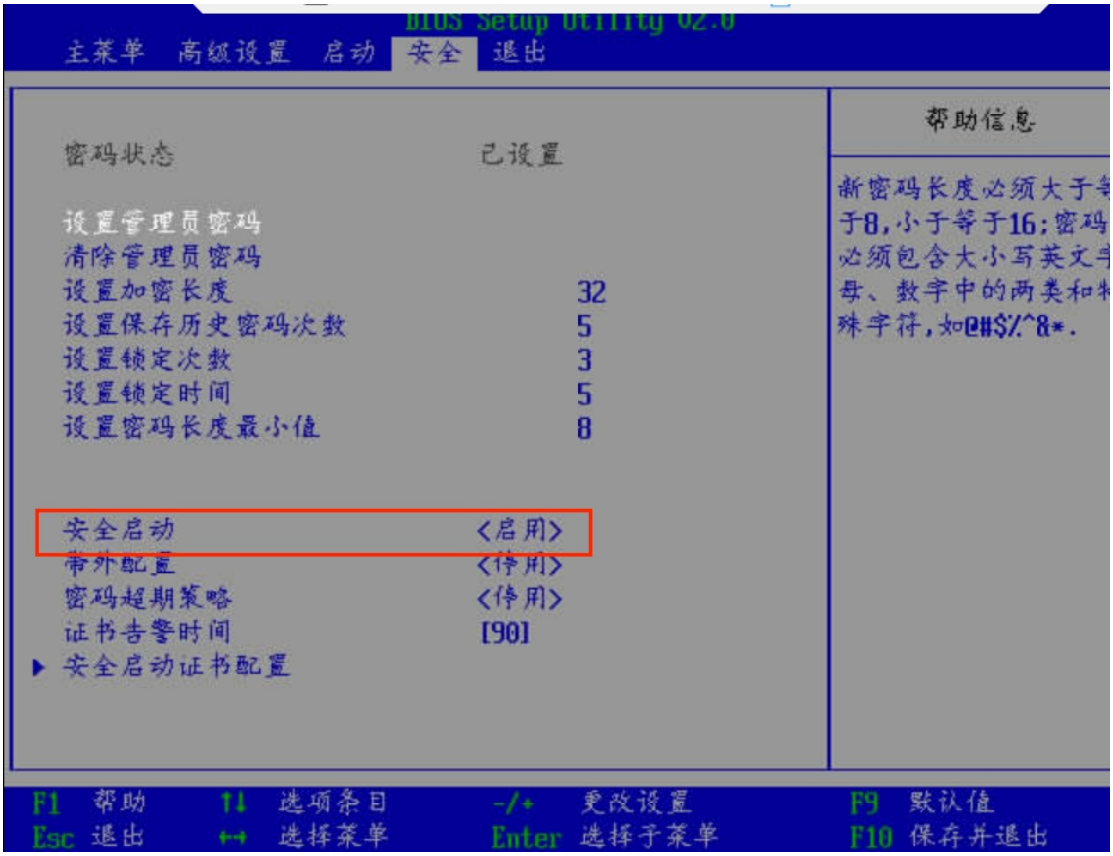


2. 安全启动在固件侧查看当前状态

目前以如下的机器固件版本为例，进行说明。

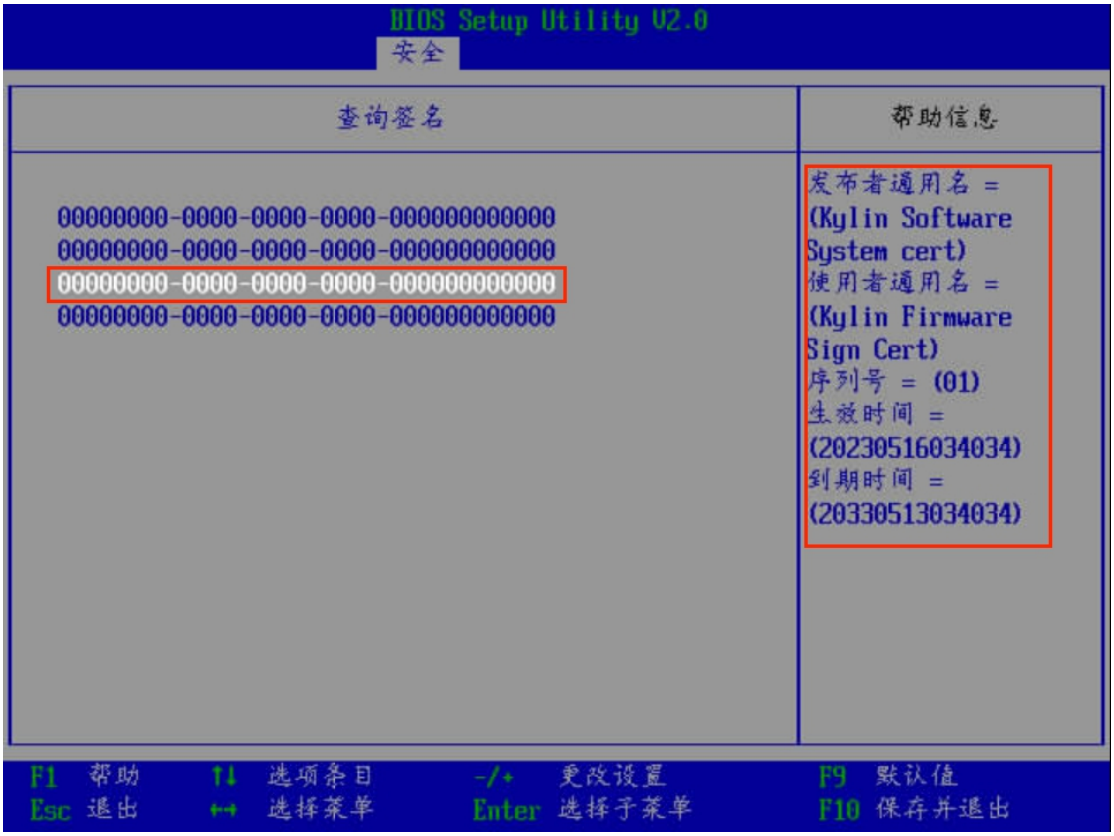


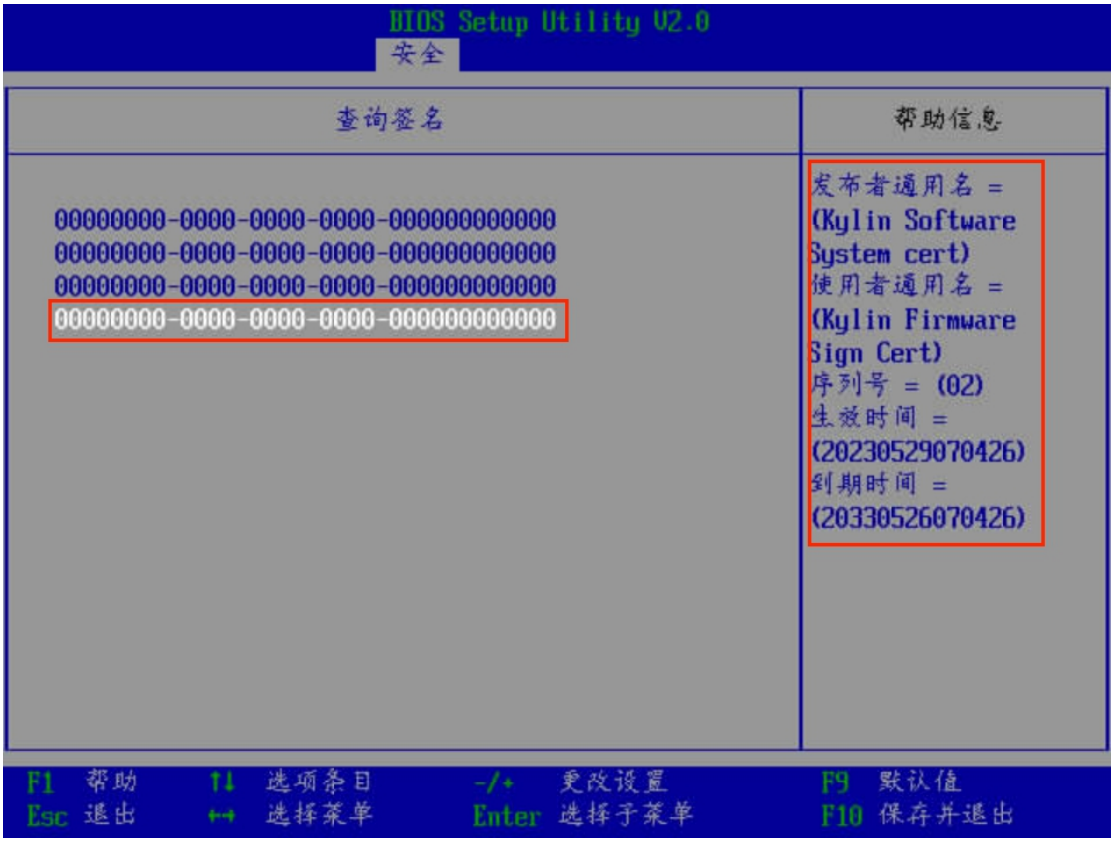
查看安全启动的状态，进入安全选项界面，如下查看安全启动状态。



3. 查看麒麟安全启动证书是否已经导入固件中

在安全启动证书配置选项进入 -> 安全启动模式 -> 选择自定义模式 -> 安全启动自定义模式相关选项 -> DB 相关选项 -> 查询签名





4. 安全启动开启方式

选择安全启动选项，按 enter 键会弹出如下界面，选择启用，进行打开安全启动，然后进行按 F10 键进行保存配置并重启。

重启后进入系统，通过系统侧查看安全启动的状态，如下图：

```
[root@localhost ~]# mokutil  --sb-state
SecureBoot enabled
[root@localhost ~]#
```

5. 安全启动关闭方式

同样采用如上步骤选择停用选项，进行关闭安全启动，然后进行按 F10 键进行保存配置并重启。

重启后进入系统，通过系统侧查看安全启动的状态，如下图：

```
[root@localhost ~]#
[root@localhost ~]# mokutil  --sb-state
SecureBoot disabled
[root@localhost ~]#
```

13. 可信启动

银河麒麟高级服务器操作系统 V10 SP3 2403 实现了基于 TPM、KYEE 两类物理可信根的可信启动功能，对操作系统的完整性保护提供了完善保障。

可信启动交互操作命令如下：

```
[root@localhost ~]# kytrust_config -h

Usage: kytrust_config [option]... [parameter]...
-s  [on/off]                    配置可信启动的启停状态（on：开启，off：关闭）
-u  [all/filepath/off]          更新基准库（可信根为tpm或soft时，all：全量更新，filepath：更新指定文件基准值，立即生效）
                                （可信根为kyee时，all：全量更新，off:关闭更新，重启生效）
-d  [devicepath]                配置可信根（不可单独使用，需配合-s参数）
-v                              查询可信根信息
-r  [all/bios/grub/tpcm/uefi]   查询可信报告（可信根为tpm或soft时，all：全部报告，bios：bios阶段报告，grub：grub阶段报告）
                                （可信根为kyee时，all：全部报告，grub：grub阶段报告，tpcm：tpcm阶段报告，uefi：uefi阶段报告）
-a  [filepath]                  新增文件纳入可信启动基准库（只支持可信根为tpm或soft）
-o  [filepath]                  移除已纳入可信启动基准库的文件（只支持可信根为tpm或soft）
-p  [nv_password]               传入基准库的完整性保护口令
-P  [hierarchy_password]        传入可信根的所有者(Owner)授权数据
-E  [pcr_extend]                pcr扩展(tpm/2/off)
-h                              帮助手册
```

13.1 设置开关状态

可信启动共有两种状态；开启(on)、关闭(off)。

13.1.1 设置可信启动状态为 on

```
#kytrust_config -s on
```

```
[root@localhost ~]# kytrust_config -s on
measure /boot/vmlinuz-0-rescue-40884170c7ef4ff38cc1a3b2179670a6
measure /boot/vmlinuz-4.19.90-89.2.v2401.ky10.x86_64
measure /boot/vmlinuz-4.19.90-89.2.v2401.ky10.x86_64-back
```

13.1.2 设置可信启动状态为 off

```
#kytrust_config -s off
```

```
[root@localhost ~]# kytrust_config -s off
```

13.2 更新基准库

13.2.1 全量更新

```
#kytrust_config -u all
```

```
[root@localhost ~]# kytrust_config -u all
update /boot/vmlinuz-0-rescue-40884170c7ef4ff38cc1a3b2179670a6
measure_base_value: 302cd66db2d230a6ec21255d3cfd9adafd82148a6762c1ec40927bd8615bb857
update time 2024-03-11 16:43:38
```

13.2.2 更新指定文件

```
# kytrust_config -u filepath
```

```
[root@localhost ~]# kytrust_config -u /boot/efi/EFI/kylin/grub.cfg
update /boot/efi/EFI/kylin/grub.cfg
measure_base_value: 0b613e293e9a548105fecc27f4333975d952f14d82c58fd4f4b84eded508cb1f
update time 2024-03-11 17:30:56
```

13.3 新增文件纳入可信启动基准库

```
# kytrust_config -a filepath
```

```
[root@localhost ~]# kytrust_config -a /boot/test1
File descriptor 3 (/dev/tpmrm0) leaked on vgs invocation. Parent PID 5700: grub2-probe
File descriptor 4 (pipe:[174538]) leaked on vgs invocation. Parent PID 5700: grub2-probe
File descriptor 5 (pipe:[174560]) leaked on vgs invocation. Parent PID 5700: grub2-probe
File descriptor 3 (/dev/tpmrm0) leaked on vgs invocation. Parent PID 5700: grub2-probe
File descriptor 4 (pipe:[174538]) leaked on vgs invocation. Parent PID 5700: grub2-probe
File descriptor 5 (pipe:[174560]) leaked on vgs invocation. Parent PID 5700: grub2-probe
mount: /backup: /dev/mapper/klas-backup 已挂载于 /backup.
measure /boot/test1
```

13.4 移除已纳入可信启动基准库的文件

```
# kytrust_config -o filepath
```

```
[root@localhost ~]# kytrust_config -o /boot/test1
delete /boot/test1
```

13.5 指定可信根设备

13.5.1 指定可信根设备（例可信根设备：/dev/tpm0）

```
# kytrust_config -s on -d /dev/tpm0
```

```
[root@localhost ~]# kytrust_config -s on -d /dev/tpm0
measure /boot/vmlinuz-0-rescue-40884170c7ef4ff38cc1a3b2179670a6
measure /boot/vmlinuz-4.19.90-89.2.v2401.ky10.x86_64
```

13.6 查询可信根信息

```
# kytrust_config -v
```

```
[root@localhost ~]# kytrust_config -v
chipType:     TPM
chipVersion:  2.0
specVersion:  1.38
manufacturer: HYG
chipname:
supportAlg:   rsa sha1 hmac aes mgf1 keyedhash xor sha256 sm3_256 s
m4 rsassa rsaes rsapss oaep ecdsa ecdh ecdaa sm2 ecschnorr kdf1_sp8
00_56a kdf1_sp800_108 ecc symcipher ctr ofb cbc cfb ecb
nvDefined:    4
```

13.7 查询可信报告

13.7.1 查询全部报告

```
# kytrust_config -r all
```

```
[root@localhost ~]# kytrust_config -r all
Bios     true
Grub     true
Kernel   true
Mersure report time           2024-03-11 17:02:13
Total number of measurements  23
Number of failures            0
```

13.7.2 查询 BIOS 阶段度量报告

```
# kytrust_config -r bios
```

```
[root@localhost ~]# kytrust_config -r bios
Measurement report for bios:
filename: bios_pcr0  time: 2024-03-11 17:02:12  result: true
filename: bios_pcr1  time: 2024-03-11 17:02:12  result: true
filename: bios_pcr2  time: 2024-03-11 17:02:12  result: true
filename: bios_pcr3  time: 2024-03-11 17:02:12  result: true
filename: bios_pcr4  time: 2024-03-11 17:02:12  result: true
filename: bios_pcr5  time: 2024-03-11 17:02:12  result: true
filename: bios_pcr6  time: 2024-03-11 17:02:12  result: true
filename: bios_pcr7  time: 2024-03-11 17:02:13  result: true
```

13.7.3 查询 grub 阶段度量报告

```
# kytrust_config -r grub
```

```
[root@localhost ~]# kytrust_config -r grub
Measurement report for grub:
filename: /boot/vmlinuz-0-rescue-40884170c7ef4ff38cc1a3b2179670a6  time: 2024-03-11 17:02:05  result: true
filename: /boot/vmlinuz-4.19.90-89.2.v2401.ky10.x86_64  time: 2024-03-11 17:02:05  result: true
filename: /boot/vmlinuz-4.19.90-89.2.v2401.ky10.x86_64-back  time: 2024-03-11 17:02:06  result: true
filename: /boot/initramfs-0-rescue-40884170c7ef4ff38cc1a3b2179670a6.img  time: 2024-03-11 17:02:09  result: true
filename: /boot/initramfs-4.19.90-89.2.v2401.ky10.x86_64.img  time: 2024-03-11 17:02:10  result: true
filename: /boot/initramfs-4.19.90-89.2.v2401.ky10.x86_64.img-back  time: 2024-03-11 17:02:12  result: true
filename: /boot/efi/EFI/kylin/grub.cfg  time: 2024-03-11 17:02:12  result: true
filename: /etc/java/java-11-openjdk/java-11-openjdk-11.0.21.9-1.ky10.x86_64/conf/security/nss.cfg  time: 2024-03-11 17:02:12  result: true
filename: /etc/java/java-1.8.0-openjdk/java-1.8.0-openjdk-1.8.0.392.b08-2.p02.ky10.x86_64/lib/security/nss.cfg  time: 2024-03-11 17:02:12  result: true
filename: /etc/texlive/web2c/updmap.cfg  time: 2024-03-11 17:02:12  result: true
filename: /etc/lynx-site.cfg  time: 2024-03-11 17:02:12  result: true
filename: /etc/lynx.cfg  time: 2024-03-11 17:02:12  result: true
filename: /etc/libblockdev/conf.d/00-default.cfg  time: 2024-03-11 17:02:12  result: true
filename: /etc/libblockdev/conf.d/10-lvm-dbus.cfg  time: 2024-03-11 17:02:12  result: true
filename: /etc/rdma/ibacm_opts.cfg  time: 2024-03-11 17:02:12  result: true
```

13.8 安全保护设置

13.8.1 设置基准库的完整性保护口令

13.8.1.1 开启可信启动功能时设置完整性保护口令（例口令：nvpassword）

```
# kytrust_config -s on -p nvpassword
```

```
[root@localhost ~]# kytrust_config -s on -p nvpassword
measure /boot/vmlinuz-0-rescue-40884170c7ef4ff38cc1a3b2179670a6
measure /boot/vmlinuz-4.19.90-89.2.v2401.ky10.x86_64
```

13.8.1.2 可信启动执行全量更新附加完整性保护口令（例口令：nvpassword）

```
# kytrust_config -u all -p nvpassword
```

```
[root@localhost ~]# kytrust_config -u all -p nvpassword
update /boot/vmlinuz-0-rescue-40884170c7ef4ff38cc1a3b2179670a6
measure_base_value: 302cd66db2d230a6ec21255d3cfd9adafd82148a6762c1ec40927bd8615bb857
update time 2024-03-11 17:46:12
```

13.8.1.3 可信启动执行更新指定文件操作附加完整性保护口令

（例口令：nvpassword）

```
# kytrust_config -u filepath -p nvpassword
```

```
[root@localhost ~]# kytrust_config -u /boot/efi/EFI/kylin/grub.cfg -p nvpassword
update /boot/efi/EFI/kylin/grub.cfg
measure_base_value: 0b613e293e9a548105fecc27f4333975d952f14d82c58fd4f4b84eded508cb1f
update time 2024-03-11 17:47:10
```

13.8.1.4 可信启动新增文件纳入可信启动基准库附加完整性保护口令

(例口令：nvpassword)

```
# kytrust_config -a filepath -p nvpassword
```

```
[root@localhost test]# kytrust_config -a /boot/test1 -p nvpassword
File descriptor 3 (/dev/tpmrm0) leaked on vgs invocation. Parent PID 346344: grub2-probe
File descriptor 4 (pipe:[612732]) leaked on vgs invocation. Parent PID 346344: grub2-probe
File descriptor 5 (pipe:[612754]) leaked on vgs invocation. Parent PID 346344: grub2-probe
File descriptor 3 (/dev/tpmrm0) leaked on vgs invocation. Parent PID 346344: grub2-probe
File descriptor 4 (pipe:[612732]) leaked on vgs invocation. Parent PID 346344: grub2-probe
File descriptor 5 (pipe:[612754]) leaked on vgs invocation. Parent PID 346344: grub2-probe
mount: /backup: /dev/mapper/klas-backup 已挂载于 /backup.
measure /boot/test1
```

13.8.1.5 移除已纳入可信启动基准库的文件(例口令：nvpassword)

```
# kytrust_config -o filepath -p nvpassword
```

```
[root@localhost test]# kytrust_config -o /boot/test1 -p nvpassword
delete /boot/test1
```

13.8.2 传入可信根的所有者保护口令

(如 TPM 存储 hierarchy 设置了保护口令，开启可信启动功能时需传入)

```
# kytrust_config -s on -P owner
```

```
[root@localhost ~]# tpm2 changeauth  -c  o "owner" -T device:/dev/tpm0
[root@localhost ~]# kytrust_config -s on -P owner
measure /boot/vmlinuz-0-rescue-40884170c7ef4ff38cc1a3b2179670a6
measure /boot/vmlinuz-4.19.90-89.2.v2401.ky10.x86_64
```

13.9 PCR：8，9 扩展状态设置

13.9.1 设置 TPM 为可信根的 PCR 扩展

```
# kytrust_config -E tpm
```

```
[root@localhost ~]# kytrust_config -E tpm
```

13.9.2 设置 PCR 扩展状态为 off

```
# kytrust_config -E off
```

```
[root@localhost ~]# kytrust_config -E off
```

14. PKS 组件

银河麒麟服务器操作系统将系统结构划分为两个不同的计算域，对应为双体系架构麒麟可信执行环境(KYLIN TEE)与麒麟通用计算环境（KYLIN REE）。

在麒麟可信执行环境和通用计算环境两个域内又进一步区分了内核态和用户态，分别对应内核模块和应用服务。从整体来看，可信固件在 CPU 最高特权级实现软硬结合的可信核与可信计算体系， 通用计算环境是对现有操作系统的继承并增加了可信防护设计，应用层的用户态则和现有用户态一致。系统硬件资源被划分为安全硬件资源和普通硬件资源两类， 并向通用计算环境开放普通硬件资源的访问权限。 用户态进程或者通用计算环境代码需要访问安全资源时，需要通用计算核态的安全代理通过安全内核服务接口请求麒麟可信执行环境的安全服务提供代理访问和计算操作。

整个系统实现内生可信计算、密码服务、系统安全框架、可信执行控制等安全可信功能的基本环境，实现了可信根、可信启动度量功能。 启动固件对麒麟可信执行环境加载度量， 麒麟可信执行环境对完整的 BIOS 固件（ UEFI/UBoot） 度量，BIOS 固件通过麒麟可信执行环境实现对 Bootloader（ Grub） 度量，Bootloader 通过调用麒麟可信执行环境实现对麒麟可信操作系统的内核加载程序、 内核等关键文件的加载和度量； 从而完成加电到进入操作系统的逐级度量， 实现启动阶段完整的信任链传递， 保证系统启动加载的安全。

14.1. 可信度量图形工具

从开始菜单点击“所有程序”，然后在系统工具子菜单中点击“安全中心”，可以打开安全中心工具，如下图所示：

安全中心的可信度量功能界面如下图：

14.1.1.系统信任链

基于 CPU-IP 核（即 TCM/TPCM）硬件资源构建可信根，提供可信基环境。

在可信度量界面可以对可信启动过程中所有环节的度量结果进行查看。系统启动过程中如果有度量值改变的文件，系统会被标记为不可信，并且用户可进行查看和信任操作。如图所示。

(1) 包括可信芯片、BIOS、引导程序、操作系统、应用程序，每次系统启动后更新信任链状态，用户进入可信度量后，动态展示各个环节的度量结果。

(2) 如果某个节点存在不可信的文件，则该节点标红；节点正常，则图标为蓝色；当可信芯片无法识别时，可信芯片与 BIOS 按钮置灰，无法点击。

(3) 点击“可信根”可查看详情。

查看详情：包括可信根版本、可信计算 CPU 核数量、总容量核可用容量。不同模式的可信根信息不一样，kyee 可信根详情如下图所示。

(4) 点击 TPCM、UEFI、GRUB 图标均可进入系统启动度量的详情页，如下图所示。

下图为度量详情-GRUB 阶段

14.1.2.系统启动度量

可信启动依据信任传递的思想，实现对可信根、TPCM、可信固件 UEFI 、GRUB 引导程序、系统内核、系统关键配置文件进行逐级度量，构建完整可信信任链，为设备启动建立初态的可信执行环境。

可信启动主要展示启动度量基准库条数、本次度量失败条数、本次启动可信状态、本次度量报告时间四项信息。如下图所示。

（1）启动度量基准库条数：显示为当前初始化后的基准库总条数；

（2）本次度量失败条数：显示为度量值与基准库不一致的度量失败条数 ；

（3）本次启动可信状态：显示设备本次启动度量的可信或不可信状态；

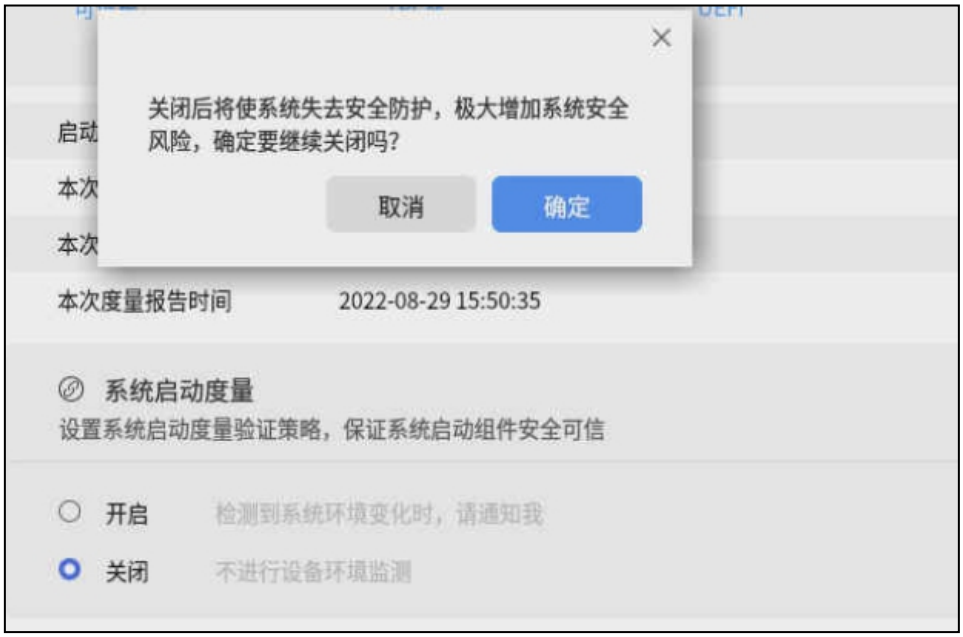
（4）本次度量报告时间：显示本次的可信度量报告产生时间；

14.1.3.系统启动度量配置

系统可信启动功能提供开启和关闭两种配置模式。如下图所示。

（1）开启：监测到系统启动环境变化时，记录日志并通知；

（2）关闭：不进行设备启动环境监测。



在弹出确认界面选择确定可改变配置，点击确定后需要重启系统后生效。

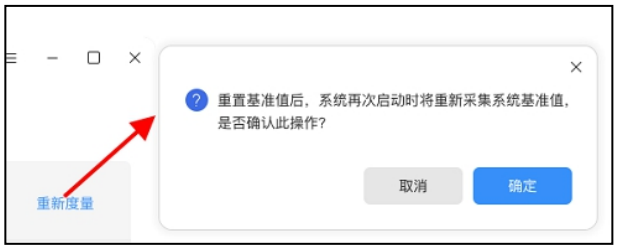
14.1.4.查看系统启动度量详情

点击“查看详情”，可查看本次可信启动的度量报告日志。如下图所示。在系统启动度量详情界面，可以选择“度量阶段”查看各阶段对应的度量详情。

14.1.5.重新采集基准值

当启动度量链上的对象改变，会出现度量失败信息，此时可信度量界面出现“重新度量”选项。当确认改变的对象是可信时，可以通过重新采集基准值，将失败的度量对象加入到信任链。

通过点击“重新采集基准值”按钮并“确定”后，将会进行基准库的重新初始化，恢复对当前度量失败项的可信状态。重启后生效。如下图所示。



14.2. 可信度量命令行工具

使用可信度量命令行工具可以对可信度量进行配置，该功能由 kytrust_config 工具提供，具体命令如下：

1. 开启可信度量

kytrust_config -s on

2. 关闭可信度量

kytrust_config -s off

3. 更新全部基准值

kytrust_config -u all

4. 读取所有度量日志：

kytrust_config -r all

```
[root@localhost 桌面]# kytrust_config -r all
Tpcm      1
Uefi      1
Grub      1
Mersure report time             2024-03-06 17:27:13
Total number of measurements    7
Number of failures              0
```

5. 读取各阶段度量日志：

kytrust_config -r [bios/grub/tpcm/uefi]

15. 动态度量

动态度量组件是一个用于监控软件及系统内核运行状态下是否有被篡改的

系统安全插件。该组件实现了针对软件及内核的关键段的周期性度量公共能，当被监控对象关键段(如：只读数据段、代码段)内容发生篡改后，动态度量组件会通过日志系统记录篡改事件，并可以根据策略配置决策是否终止发生篡改的程序。

15.1 用法

15.1.1 检测环境

检测软件包 lkrg-kyexten、libkydima 是否已安装：

rpm -qa | grep -e kydima -e lkrg

```
[root@localhost 4.19.90-89.4.v2401.ky10.x86_64]# rpm -qa | grep -e kydima -e lkrg
lkrg-kyextend-0.9.6-01.se.13.ky10.x86_64
libkydima-1.1.0-07.ky10.x86_64
```

如果未安装则安装软件包：

yum install lkrg–kyextend

```
[root@localhost 桌面]# yum install lkrg-kyextend
Last metadata expiration check: 0:01:09 ago on 2024年03月12日 星期二 13时49分54秒.
Dependencies resolved.
==============================================================================================
 Package                 Architecture     Version                      Repository          Size
==============================================================================================
Installing:
 lkrg-kyextend           x86_64           0.9.6-01.se.13.ky10          ks10-adv-os          5.5 M
Installing dependencies:
 dkms                    noarch           2.6.1-6.ky10                 ks10-adv-os           78 k

Transaction Summary
==============================================================================================
Install  2 Packages

Total download size: 5.6 M
Installed size: 13 M
Is this ok [y/N]: y
Downloading Packages:
(1/2): dkms-2.6.1-6.ky10.noarch.rpm                              321 kB/s |  78 kB     00:00
(2/2): lkrg-kyextend-0.9.6-01.se.13.ky10.x86_64.rpm                6.5 MB/s | 5.5 MB     00:00
==============================================================================================
Total                                                              6.6 MB/s | 5.6 MB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                    1/1
  Installing       : dkms-2.6.1-6.ky10.noarch                                           1/2
  Running scriptlet: dkms-2.6.1-6.ky10.noarch                                           1/2
  Installing       : lkrg-kyextend-0.9.6-01.se.13.ky10.x86_64                           2/2
  Running scriptlet: lkrg-kyextend-0.9.6-01.se.13.ky10.x86_64                           2/2

Creating symlink /var/lib/dkms/lkrg-kyextend/0.9.6/source ->
                 /usr/src/lkrg-kyextend-0.9.6

DKMS: add completed.

Kernel preparation unnecessary for this kernel.  Skipping...

Building module:
cleaning build area...
KDIR=/lib/modules/4.19.90-89.1.v2401.ky10.x86_64/build make...
cleaning build area...

DKMS: build completed.

kyrg.ko.xz:
Running module version sanity check.
 - Original module
   - No original module exists within this kernel
 - Installation
   - Installing to /lib/modules/4.19.90-89.1.v2401.ky10.x86_64/updates/
Adding any weak-modules

depmod....

DKMS: install completed.

  Running scriptlet: lkrg-kyextend-0.9.6-01.se.13.ky10.x86_64
  Verifying        : dkms-2.6.1-6.ky10.noarch
  Verifying        : lkrg-kyextend-0.9.6-01.se.13.ky10.x86_64

Installed:
  dkms-2.6.1-6.ky10.noarch                                                  lkrg-kyextend-

Complete!
[root@localhost 桌面]#
```

yum install libkydima

```
[root@localhost 桌面]# yum install libkydima
Last metadata expiration check: 0:06:43 ago on 2024年03月12日 星期二 13时49分54秒.
Dependencies resolved.
================================================================================
 Package                 Architecture          Version              Repository          Size
================================================================================
Installing:
 libkydima               x86_64                1.1.0-07.ky10        ks10-adv-os         322 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 322 k
Installed size: 2.0 M
Is this ok [y/N]: y
Downloading Packages:
libkydima-1.1.0-07.ky10.x86_64.rpm                  934 kB/s | 322 kB     00:00
--------------------------------------------------------------------------------
Total                                               931 kB/s | 322 kB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : libkydima-1.1.0-07.ky10.x86_64                         1/1
  Running scriptlet: libkydima-1.1.0-07.ky10.x86_64                         1/1
Created symlink /etc/systemd/system/multi-user.target.wants/kydima-init-policy.service → /usr/lib/systemd/system/kydima-init-policy.service.
Created symlink /etc/systemd/system/multi-user.target.wants/kydima-daemon.service → /usr/lib/systemd/system/kydima-daemon.service.

  Verifying        : libkydima-1.1.0-07.ky10.x86_64                         1/1

Installed:
  libkydima-1.1.0-07.ky10.x86_64

Complete!
[root@localhost 桌面]#
```

检测 auditd 服务是否正常启动：

systemctl status auditd

```
内核审计功能未开启!!!
[root@localhost 桌面]# systemctl status auditd
● auditd.service - Security Auditing Service
   Loaded: loaded (/usr/lib/systemd/system/auditd.service; enabled; vendor preset: enabled)
   Active: inactive (dead)
Condition: start condition failed at Tue 2024-03-12 14:42:33 CST; 56min ago
           └─ConditionKernelCommandLine=!audit=0 was not met
     Docs: man:auditd(8)
           https://github.com/linux-audit/audit-documentation

3月 12 14:42:33 localhost.localdomain systemd[1]: Condition check resulted in Security Auditing Service being skipped.
```

auditd 服务已启动，需要进一步检测内核 audit 审计是否开启，查看 grub 引导参数，是否配置启动了 audit 组件。

#cat /proc/cmdline

```
[root@localhost 桌面]# cat /proc/cmdline
BOOT_IMAGE=/vmlinuz-4.19.90-89.4.v2401.ky10.x86_64 root=/dev/mapper/klas-root ro
 resume=/dev/mapper/klas-swap rd.lvm.lv=klas/root rd.lvm.lv=klas/swap rhgb quiet
 crashkernel=1024M,high audit=0 console=ttyS0,115200 console=tty0 loglevel=8
[root@localhost 桌面]#
```

编辑 grub.cfg，将 audit 配置改成 audit=1，并重启系统。

```
### BEGIN /etc/grub.d/10_linux ###
menuentry 'Kylin Linux Advanced Server (4.19.90-89.4.v2401.ky10.x86_64) V10 (Halberd)' --c
lass kylin --class gnu-linux --class gnu --class os --unrestricted $menuentry_id_option 'g
nulinux-4.19.90-89.1.v2401.ky10.x86_64-advanced-774bbc96-910e-4617-acd7-53c3252f12e5' {
        load_video
        set gfxpayload=keep
        insmod gzio
        insmod part_msdos
        insmod xfs
        set root='hd0,msdos1'
        if [ x$feature_platform_search_hint = xy ]; then
          search --no-floppy --fs-uuid --set=root --hint-bios=hd0,msdos1 --hint-efi=hd0,ms
dos1 --hint-baremetal=ahci0,msdos1 --hint='hd0,msdos1'  17c109b0-ed07-4252-a3cc-dca56e5c81
79
        else
          search --no-floppy --fs-uuid --set=root 17c109b0-ed07-4252-a3cc-dca56e5c8179
        fi
        linux   /vmlinuz-4.19.90-89.4.v2401.ky10.x86_64 root=/dev/mapper/klas-root ro resu
me=/dev/mapper/klas-swap rd.lvm.lv=klas/root rd.lvm.lv=klas/swap rhgb quiet crashkernel=10
24M,high audit=1 console=ttyS0,115200 console=tty0 loglevel=8
        initrd /initramfs-4.19.90-89.4.v2401.ky10.x86_64.img
}
```

也可以修改 /etc/default/grub 文件，启用内核 audit 审计。

```
[root@localhost etc]# cat default/grub
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="resume=/dev/mapper/klas-swap rd.lvm.lv=klas/root rd.lvm.lv=k
las/swap rhgb quiet crashkernel=1024M,high audit=0"
GRUB_DISABLE_RECOVERY="true"
[root@localhost etc]#
```

15.1.2 功能开关配置

关闭：

kydima_set switch –s off

开启：

kydima_set switch –s on –c 5

15.1.3 查看动态度量策略

```
[root@localhost ~]# kydima_get
用法：kydima_get <switch|process|module|kernel|syscalls|idt|cycle|event> [-f] [-p] [-l]
功能：进程度量与内核防护策略查询
参数:
```

-h, --help 显示帮助信息

switch: 开关策略

cycle: 周期设置

process: 进程度量

-l, --list 查看列表

-f, --file 文件路径

-p, --pid 进程号

module: 内核模块度量

kernel: 内核度量

syscalls: 系统调用表度量

idt: 中断描述符表度量

event: 事件触发开关

pcr: PCR 扩展策略

举例:

kydima_get process -f /usr/lib/systemd/systemd

kydima_get process -p 1

kydima_get module

kydima_get kernel

kydima_get syscalls

kydima_get idt

kydima_get switch

kydima_get cycle

kydima_get event

kydima_get pcr

- 查看状态：

[root@localhost ~]# kydima_get switch

switch：on

- 查看周期度量时间：

[root@localhost ~]# kydima_get cycle

cycle：15 分钟

- 查看模块度量状态：

[root@localhost ~]# kydima_get module

module: on

- 查看进程度量信息：

[root@localhost ~]# kydima_get process /usr/lib/system/system

process: 进程路径：/usr/lib/systemd/systemd，度量策略：仅审计，度量结果：未篡改/已篡改

- 查看内核度量状态：

[root@localhost ~]# kydima_get kernel

kernel: on

- 查看系统调用表度量状态：

[root@localhost ~]# kydima_get syscalls

syscalls: on

- 查看中断描述符表度量状态：

[root@localhost ~]# kydima_get idt

idt: on

- 查看事件触发开关状态：

[root@localhost ~]# kydima_get event

event：off

- 查看 PCR 扩展策略：

[root@localhost ~]# kydima_get pcr

pcr：off

pcr：register: 13

pcr：tpm :tpm2

- kydima_get 获取所有的进程策略被篡改的日志：

[root@localhost ~]# kydima_get process - list

15.1.4 关于 pcr 配置注意事项

a) 检测 tpm/tcm 设备节点是否存在

#ls /dev/t[pc]m*

tcm0 对应设备类型 tcm1

tcm2 对应设备类型 tcm2

tpm* 对应设备类型 tpm2

b) 核实 tpm 是否为 pcie 外插 tpm 卡,该信息需要核实具体硬件无法冲软件层面查询。（动态度量针对 pcie 卡 tpm 后续支持）

15.1.5 配置动态度量策略

[root@localhost ~]# kydima_set

用 法 : kydima_set <switch|process|module|kernel|syscalls|idt|event> [-a] [-d] [-f] [-p] [-s] [-c] <value>

功能: 进程度量与内核防护策略设置

参数:

-h, --help 显示帮助信息

switch: 开关策略

-s, --status 状态 on/off。其中，on 表示开启，off 表示关闭

当功能关闭时,无法配置其他参数

-c, --cycle 周期设置(分钟)

process: 进程度量

-a, --add 添加进程

-d, --del 删除进程

-v, --value 度量策略

kill: 发现被篡改后杀死该进程

audit: 发现被篡改后仅记录审计日志

-f, --file 文件路径

-p, --pid 进程号

module: 内核模块度量

-s, --status 状态 on/off。其中，on 表示开启，off 表示关闭

kernel: 内核度量

-s, --status 状态 on/off。其中，on 表示开启，off 表示关闭

syscalls: 系统调用表度量

-s, --status 状态 on/off。其中，on 表示开启，off 表示关闭

idt: 中断描述符表度量

-s, --status 状态 on/off。其中，on 表示开启，off 表示关闭

event: 事件触发开关

-s, --status 状态 on/off。其中，on 表示开启，off 表示关闭

pcr: PCR 扩展策略

-s, --status 状态 on/off。其中，on 表示开启，off 表示关闭

-r, --register 度量值扩展位置，默认使用 13，可取值 13~23

-t, --tpm 指定硬件可信设备名称，如 tcm1、tcm2、tpm1、tpm2

举例:

```
kydima_set process -a -f /usr/lib/systemd/systemd -v audit
kydima_set process -d -f /usr/lib/systemd/systemd
kydima_set process -a -p 1 -v kill
kydima_set process -d -p 1
kydima_set module -s on
kydima_set kernel -s on
kydima_set syscalls -s on
kydima_set idt -s on
kydima_set switch -s on -c 5
kydima_set event -s on
kydima_set pcr -s on -r 13 -t tcm1
kydima_set pcr -s off
```

- 打开动态度量功能，并设置度量周期为 5 分钟

  [root@localhost ~]# kydima_set switch -s on -c 5

- 添加一条策略，将 systemd 进程设置为发现被篡改后仅记录审计日志：

  [root@localhost ~]# kydima_set process -a -f /usr/lib/systemd/systemd -v audit

- 添加一条策略，将进程号为 1 的程序设置为发现被篡改后仅记录审计日志：

  [root@localhost ~]# kydima_set process -a -p 1 -v audit

- 打开对内核模块的度量

  [root@localhost ~]# kydima_set module -s on

- 打开对内核的度量

  [root@localhost ~]# kydima_set kernel -s on

- 打开对系统调用表的度量

  [root@localhost ~]# kydima_set syscalls -s on

- 打开对中断描述符表的度量

  [root@localhost ~]# kydima_set idt -s on

- 设置 PCR 扩展策略

  [root@localhost ~]# kydima_set pcr -s on # pcr 扩展功能开启

  [root@localhost ~]# kydima_set pcr -s on -r 14 -t tcm1 # pcr 扩展开启，设置扩展到 14，采用 tcm1 可信设备

  [root@localhost ~]# kydima_set pcr -s off # pcr 扩展功能关闭

15.2 示例

kydima_get --help

```
[root@localhost 桌面]# kydima_get --help
用法：kydima_get <switch|process|module|kernel|syscalls|idt|cycle|event> [-f] [-p] [-l]
功能：  进程度量与内核防护策略查询
参数：
        -h, --help 显示帮助信息
        switch:   开关策略
        cycle:    周期设置
        process:  进程度量
                -l, --list    查看列表
                -f, --file    文件路径
                -p, --pid     进程号
        module:   内核模块度量
        kernel:   内核度量
        syscalls: 系统调用表度量
        idt:      中断描述符表度量
        event:    事件触发开关
        pcr:      扩展pcr策略
举例：
        kydima_get process -f /usr/lib/systemd/systemd
        kydima_get process -p 1
        kydima_get module
        kydima_get kernel
        kydima_get syscalls
        kydima_get idt
        kydima_get switch
        kydima_get cycle
        kydima_get event
        kydima_get pcr
```

kydima_set --help

```
[root@localhost 桌面]# kydima_set --help
用法：kydima_set <switch|process|module|kernel|syscalls|idt|event> [-a] [-d] [-f] [-p] [-s] [-c] <value>
功能：  进程度量与内核防护策略设置
参数：
       -h, --help 显示帮助信息
       switch:    开关策略
              -s, --status   状态 on/off。其中，on表示开启，off表示关闭
                             当功能关闭时,无法配置其他参数
              -c, --cycle    周期设置(分钟)
       process:    进程度量
              -a, --add   添加进程
              -d, --del   删除进程
              -v, --value   度量策略
                     kill:    发现被篡改后杀死该进程
                     audit:   发现被篡改后仅记录审计日志
              -f, --file         文件路径
              -p, --pid          进程号
       module:    内核模块度量
              -s, --status   状态 on/off。其中，on表示开启，off表示关闭
       kernel:    内核度量
              -s, --status   状态 on/off。其中，on表示开启，off表示关闭
       syscalls: 系统调用表度量
              -s, --status   状态 on/off。其中，on表示开启，off表示关闭
       idt:      中断描述符表度量
              -s, --status   状态 on/off。其中，on表示开启，off表示关闭
       event:    事件触发开关
              -s, --status   状态 on/off。其中，on表示开启，off表示关闭
       pcr:    扩展pcr策略
              -s, --status   状态 on/off。其中，on表示开启，off表示关闭
              -r, --register   度量值扩展位置。默认使用13，可取值13~23
              -t, --tpm      指定硬件可信设备名称。如tcm1、tcm2、tpm2
举例：
       kydima_set process -a -f /usr/lib/systemd/systemd -v audit
       kydima_set process -d -f /usr/lib/systemd/systemd
       kydima_set process -a -p 1 -v kill
       kydima_set process -d -p 1
       kydima_set module -s on
       kydima_set kernel -s on
       kydima_set syscalls -s on
       kydima_set idt -s on
       kydima_set switch -s on -c 5
       kydima_set event -s on
       kydima_set pcr -s on
       kydima_set pcr -s on -r 14 -t tpm2
       kydima_set pcr -s off
```

kydima 基本操作：

```
[root@localhost 桌面]# kydima_set switch -s off
[root@localhost 桌面]# kydima_get switch
kyrg模块未加载！！！
[root@localhost 桌面]# kydima_set switch -s on -c 5
[root@localhost 桌面]# kydima_get switch
switch:  on
[root@localhost 桌面]# kydima_set module -s on
[root@localhost 桌面]# kydima_get switch
switch:  on
[root@localhost 桌面]# kydima_get module
module:  on
[root@localhost 桌面]# kydima_set kernel -s on
[root@localhost 桌面]# kydima_get kernel
kernel:  on
[root@localhost 桌面]# kydima_set syscalls -s on
[root@localhost 桌面]# kydima_get syscalls
syscalls:  on
[root@localhost 桌面]# kydima_set idt  -s on
[root@localhost 桌面]# kydima_get idt
idt:  on
[root@localhost 桌面]#
```

```
[root@localhost 桌面]# kydima_set kernel -s off
[root@localhost 桌面]# kydima_get kernel
kernel:  off
[root@localhost 桌面]# kydima_set module -s off
[root@localhost 桌面]# kydima_get module
module:  off
[root@localhost 桌面]# kydima_set syscalls -s off
[root@localhost 桌面]# kydima_get syscalls
syscalls:  off
[root@localhost 桌面]# kydima_set idt -s off
[root@localhost 桌面]# kydima_get idt
idt:  off
```

配置进程监控策略配置：

kydima_set process -a -f /usr/lib/systemd/systemd -v audit

kydima_set process -a -p 1 -v audit

```
[root@localhost ~]#  kydima_set process -a -f /usr/lib/systemd/systemd -v audit
[root@localhost ~]#  kydima_set process -a  -p  1 -v audit
[root@localhost ~]# kydima_get process  -l
       id |进程路径                                                pid     度量结果        度量策略

        1 |/usr/lib/systemd/systemd                                                      仅审计(audit)
          /usr/lib/systemd/systemd                                 2348    未篡改          仅审计(audit)
          /usr/lib/systemd/systemd                                 2346    未篡改          仅审计(audit)
          /usr/lib/systemd/systemd                                 1       未篡改          仅审计(audit)
```

16. 麒麟安全管理工具-安全中心

安全中心是一款基于麒麟安全框架 KYSEC 的管理工具,提供系统安全加固、账户保护配置、联网控制、应用执行控制和应用防护等功能，保障系统运行环境的安全和稳定。

安全中心集安全加固、账户保护、网络保护和应用保护等功能于一体，全面保障系统运行环境的安全。

安全加固：提供安全服务、内核参数、安全网络、系统命令、系统审计、系统设置、潜在危险、文件权限、风险账户、磁盘检查、密码强度、账户锁定、系统安全、系统维护、资源分配等多维度的扫描与一键加固，及时发现并处理系统安全隐患。

账户保护：提供系统账户密码强度检查和账户锁定机制，实现对系统账户的统一管控，提升系统账户安全防御能力，有效防止密码被暴力破解。

网络保护：提供应用联网控制功能，实时防护未知应用网络行为，阻断主动外联及其它异常网络活动，提高网络访问安全性。

应用控制：提供应用程序执行控制功能，阻止未知软件、应用程序的恶意执行，避免木马病毒攻击，保障系统运行环境的安全性、可靠性。

应用防护：提供进程防杀死、内核模块防卸载和文件防篡改功能，保护系统关键文件完整性，阻止系统关键应用服务异常中断。

可信度量：提供对计算子系统的主动度量（可信启动、静态度量、动态度量）和主动控制（可信策略）机制，保障系统执行环境的安全性、可靠性。

安全内存：提供用户可配置的内存数据安全机制，保护系统内存数据安全。

详细内容请参考《银河麒麟安全中心用户手册》。

17. 麒麟文件保护箱

麒麟文件保护箱是基于内核级数据隔离机制的保护工具，提供用户间数据隔离和加密保护功能，支持国密算法，实现一箱一密、一文一密的细粒度控制，保障用户数据安全。

具有如下特点：

多重防护：支持用户间数据隔离以及细粒度的权限控制，保障数据安全。

安全加密：支持一箱一密、一文一密的透明加密机制，且对密钥进行安全管理，能够满足政企和金融级客户的核心安全诉求。

丰富算法：支持标准国际算法、国密算法和硬件级加密算法，能够满足不同安全等级的加密应用场景。

高兼容性：支持保护箱版本兼容机制，用户升级适配无感知，保证用户数据安全存储、永不丢失。

简单易用：支持内置文件管理器，实现统一管理，操作简单、易于上手。

详细内容请参考《银河麒麟文件保护箱用户手册》。

18. 日志查看器

日志查看器是一款系统日志集中展示工具，提供日志解析和分类显示功能。

详细内容请参考《银河麒麟高级服务器操作系统–日志查看器用户手册》。

19. 安全功能配置

19.1 UID 唯一性增强

19.1.1 功能

UID 唯一性开关配置文件。银河麒麟高级服务器操作系统 V10 SP3 2403，默认未开启 UID 唯一性检查。开启状态下，通过 uid 检查库，对用户 uid 检查验证，确保用户 uid 唯一；当用户被删除后，它使用过的 UID 不能再被其他用户使用；passwd 文件中存在相同 uid 时，锁定可疑账号。

19.1.2 用法

UID 唯一性开关配置文件：/etc/chkuid_state

state=off　　关闭 UID 唯一性功能

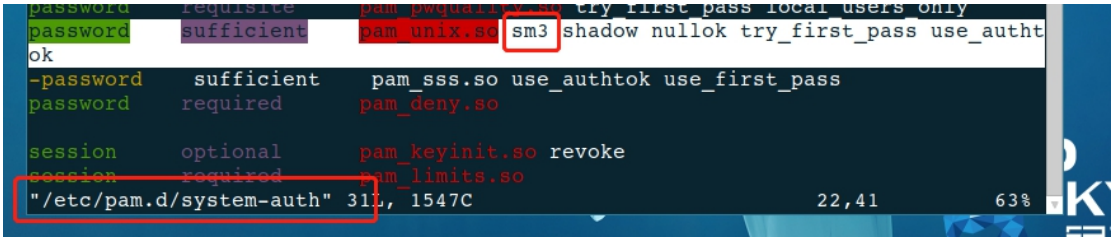
State=on　　开启 UID 唯一性功能

19.1.3 示例

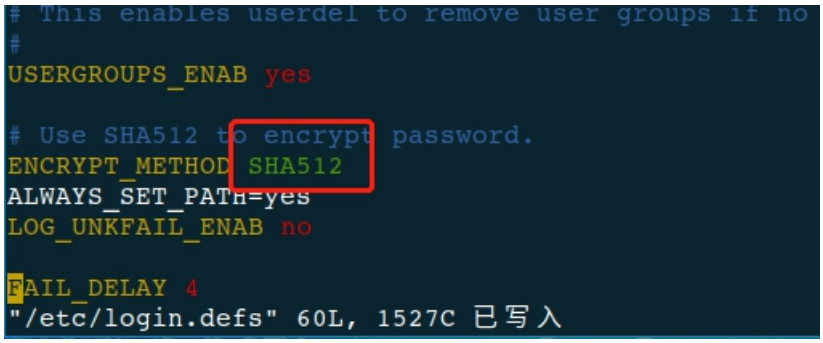
图 91 关闭 UID 唯一性功能

19.2 PAM 配置国密

首先查看/etc/pam.d/system-auth 文件中是否有对应的密码算法设置，默认为 SHA512，如下图改为 sm3，之后新建的用户使用的密码为国密 sm3。



如果/etc/pam.d/system-auth 文件中未设置任何密码算法，也可以修改下面的配置文件，将算法设置为 SM3。以/etc/pam.d/system-auth 中的设置优先。



19.3 ssh 国密配置

19.3.1 功能

Openssh 在远程连接和通信过程增加了国密支持，在麒麟系统中分为国密模式和默认模式，默认模式就是非国密的算法，国密模式为国密支持的算法。银河麒麟高级服务器操作系统 V10 SP3 2403 默认状态为非国密算法。

19.3.2 用法

openssh 国密支持配置比较方便，提供了 ssh-change-mode 脚本工具进行切换。

12.3.2.1. 切换为国密

使用如下命令进行切换：

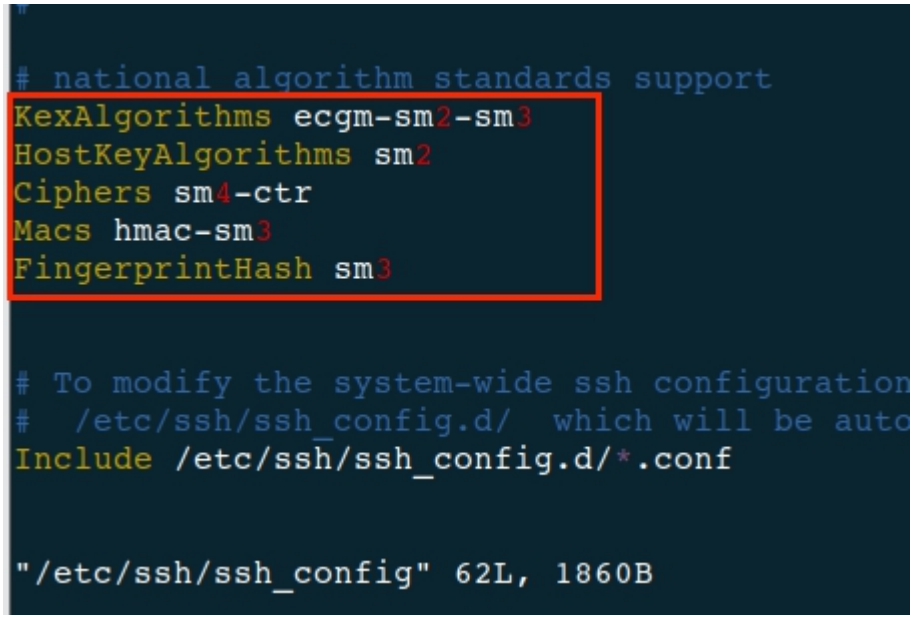
$ ssh-change-mode gm

$ systemctl daemon-reload

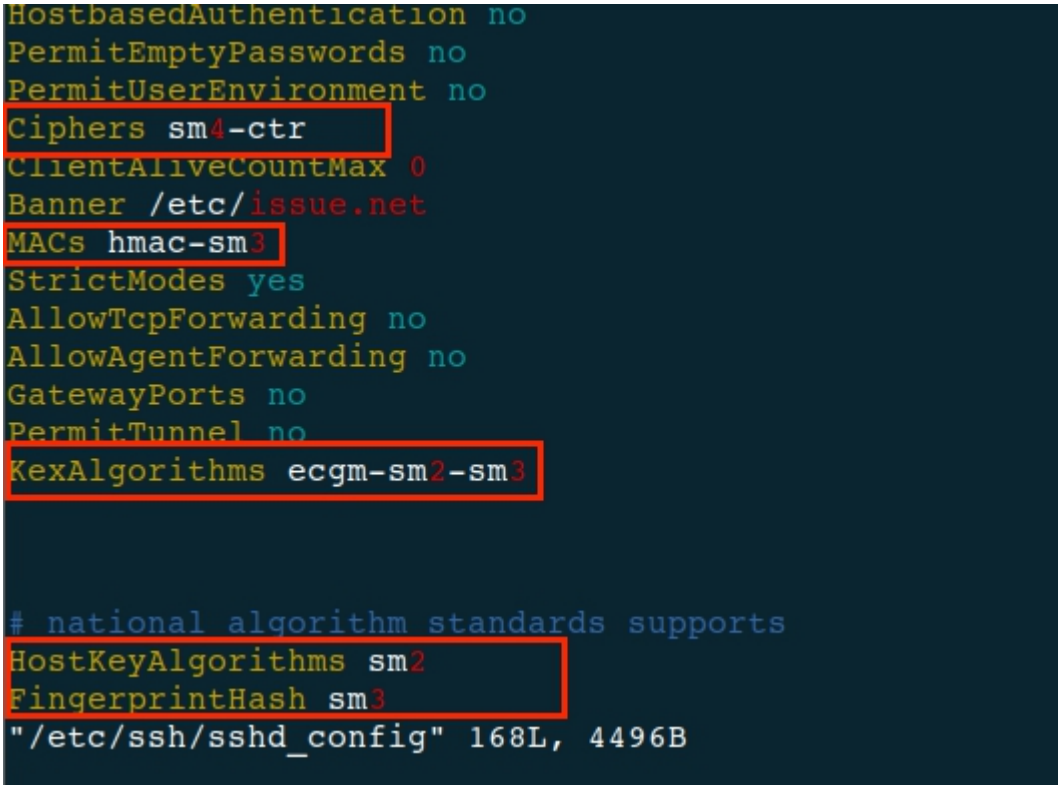
$ systemctl stop sshd

$ systemctl start sshd

切换国密后，配置文件/etc/ssh/ssh_config 的配置更新参数为如下图：



配置文件/etc/ssh/sshd_config 配置更新参数如下图：



12.3.2.2. 由国密切换为非国密

使用如下命令进行切换：

$ ssh-change-mode default

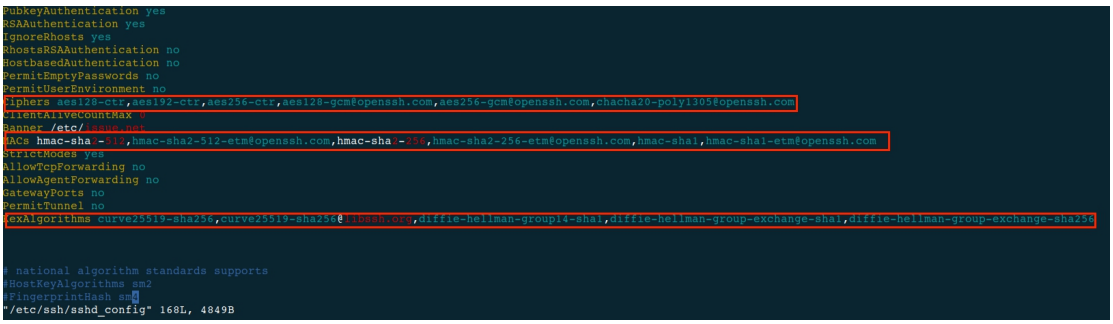
$ systemctl daemon-reload

$ systemctl stop sshd

$ systemctl start sshd

切换会默认模式后，查看当前 ssh 国密配置是否为非国密算法参数。

查看配置文件/etc/ssh/sshd_config 配置参数应为如下图：



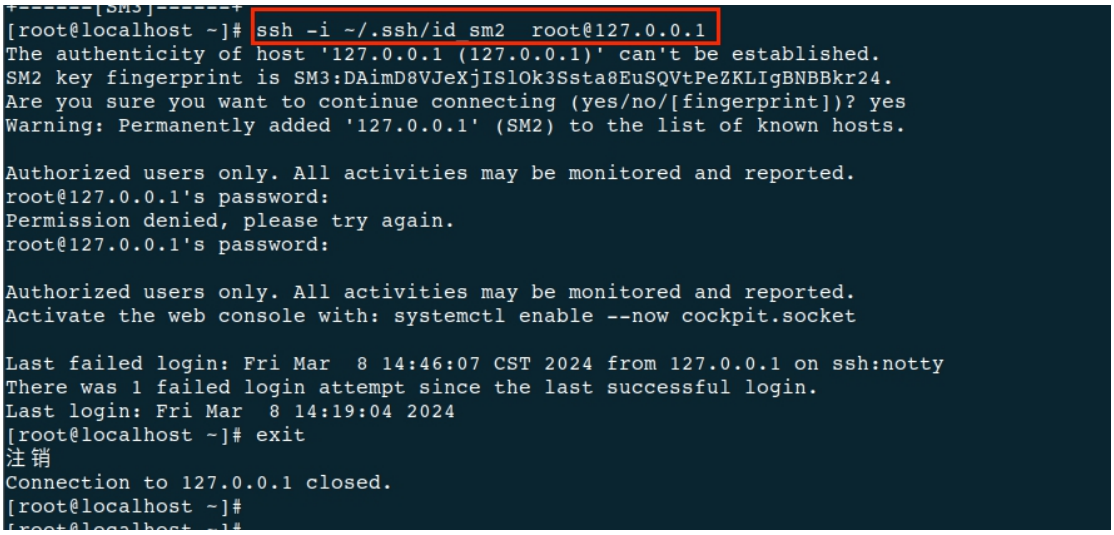
查看配置文件/etc/ssh/ssh_config 配置参数应为如下图：

19.3.3 示例

国密模式下，如执行远程连接服务器主机：

```
$ ssh -i ~/.ssh/id_sm2 username@ip
```

示例如下图：



注意，若为非跳转往外连接则不需要添加–i 的参数（即 A 直接连 B 不需要–i 参数，A 连接 C 再连接 B 则需要–i 参数）。如果彻底不需要–i 参数，可以通过配置文件/etc/ssh/ssh_config，在原有基础上，添加 IdentityFile ~/.ssh/id_sm2 如下：

```
# national algorithm standards support
KexAlgorithms ecgm-sm2-sm3
HostKeyAlgorithms sm2
Ciphers sm4-ctr
Macs hmac-sm3
FingerprintHash sm3

IdentityFile ~/.ssh/id_sm2

# To modify the system-wide ssh configuration, create a  *.conf  file under
#  /etc/ssh/ssh_config.d/  which will be automatically included below
Include /etc/ssh/ssh_config.d/*.conf

"/etc/ssh/ssh_config" 63L, 1887B
```

这样就不用加-i 了，如下所示：

```
[root@localhost ~]# ssh root@127.0.0.1

Authorized users only. All activities may be monitored and reported.
root@127.0.0.1's password:

Authorized users only. All activities may be monitored and reported.
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Fri Mar  8 16:50:42 2024 from 127.0.0.1
[root@localhost ~]# exit
注销
Connection to 127.0.0.1 closed.
[root@localhost ~]#
```

同时注意，切换回非国密时，需要去掉手动添加的 IdentityFile 配置参数。

非国密模式下，如下执行远程连接服务器主机：

ssh username@ip

示例如下图：

```
[root@localhost ~]# ssh root@127.0.0.1
The authenticity of host '127.0.0.1 (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:6XDJa4jxWNAf10ohrABaeFgzm2ztQEFtVljBuU6zgL0.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '127.0.0.1' (ECDSA) to the list of known hosts.

Authorized users only. All activities may be monitored and reported.
root@127.0.0.1's password:

Authorized users only. All activities may be monitored and reported.
Activate the web console with: systemctl enable --now cockpit.socket

Last login: Fri Mar  8 14:46:11 2024 from 127.0.0.1
[root@localhost ~]# exit
注销
Connection to 127.0.0.1 closed.
[root@localhost ~]#
```

19.4 密码回文和相似性

19.4.1 功能

密码回文检查和相似性配置。

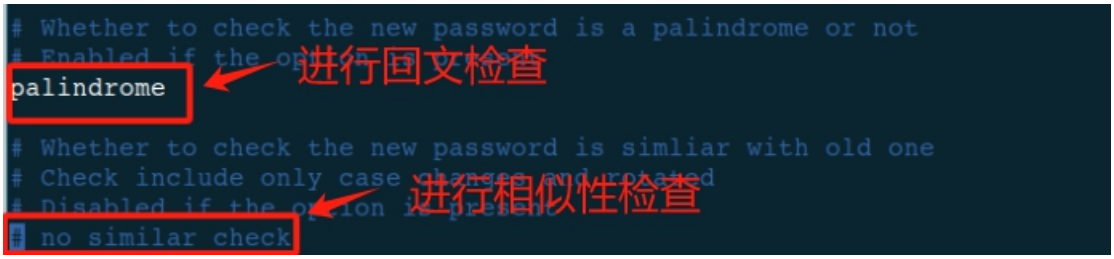
19.4.2 用法

密码回文检查和相似性配置文件：/etc/security/pwquality.conf

palindrome　　//回文检查，打开则检查

no_similar_check　　//相似性检查，打开则不检查

19.4.3 示例



19.5 security.kysec 文件属性

19.5.1 功能

文件属性管理。

19.5.2 用法

文件属性管理配置文件：/etc/xattr.conf

permissions　//许可

skip　　//略过

```
system.nfs4_acl                 permissions
system.nfs4acl                  permissions
system.posix_acl_access         permissions
system.posix_acl_default        permissions
trusted.SGI_ACL_DEFAULT         skip            # xfs specific
trusted.SGI_ACL_FILE            skip            # xfs specific
trusted.SGI_CAP_FILE            skip            # xfs specific
trusted.SGI_DMI_*               skip            # xfs specific
trusted.SGI_MAC_FILE            skip            # xfs specific
xfsroot.*                       skip            # xfs specific; obsolete
user.Beagle.*                   skip            # ignore Beagle index data
#security.evm                   skip            # may only be written by kernel
```

19.5.3 示例

通过在配置文件/etc/xattr.conf 中添加类似如下行内容，可以避开跨分区时 mv 文件的警告信息（setting attribute ‘security.kysec’ for ‘security.kysec’）：

```
security.kysec   permissions
```