

safe_rm 用户手册

Product Name : safe_rm

Product Version : V1.2.1

Release Date : 2025.11.07

Contact : liyanqing1987@163.com

目录

- 一、简介..... 3
- 二、环境依赖 4
- 三、工具配置和引用 5
 - 3.1 工具配置 5
 - 3.1.1 保护路径配置..... 5
 - 3.1.2 蜜罐路径配置..... 7
 - 3.1.3 回收站功能配置..... 7
 - 3.1.4 日志功能配置..... 8
 - 3.1.5 报警功能配置..... 9
 - 3.1.6 其它配置..... 10
 - 3.2 工具引用 10
 - 3.2.1 单用户场景 10
 - 3.2.2 全服务器用户场景 10
 - 3.3 工具打包 11
- 四、工具使用示例 13
 - 4.1 单用户场景 13
 - 4.2 全服务器用户场景 15
- 五、高级功能 18
 - 5.1 日志检索 18
 - 5.2 DEBUG 功能 19
 - 5.3 登陆用户识别..... 20
- 六、行为测试 22
- 七、技术支持 26
- 附录..... 27
 - 附 1. 变更历史..... 27

一、简介

`safe_rm` 是一个安全的删除命令，用于取代 linux 系统上的 `rm` 命令，和 `rm` 命令相比，它主要增加了如下功能：

- **数据保护功能**

保护指定数据不被删除。

- **蜜罐检测功能**

监控蜜罐数据删除行为。

- **回收站功能**

回收指定数据。

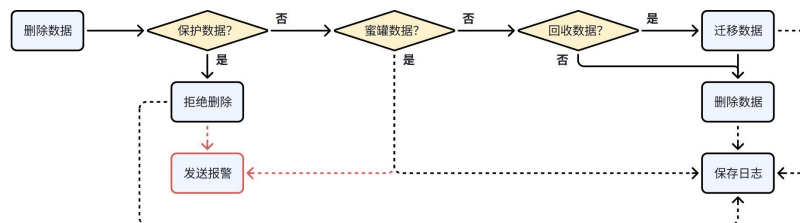
- **日志功能**

为所有的 `rm` 操作保留日志记录，可追溯。

- **报警功能**

试图删除保护数据或者蜜罐数据的行为，会直接触发安全报警。

其工作流如下。



`safe_rm` 底层默认仍然使用系统 `rm` 命令来完成删除操作，但是其数据保护功能和回收站功能可以最大限度地防止重要的目录和文件被误删，蜜罐检测功能可以及时感知危险的删除行为，并且日志和报警功能可以帮助 IT 管理员的事后追溯和安全管理，具有很高的安全价值。

请务必注意，`safe_rm` 的理念是“防止误删，但不阻止刻意地合理删除！”，因此被保护的路径无法被删除，**但是被保护路径的子路径是可以删除的。**

二、环境依赖

`safe_rm` 基于 `python3` 开发，需要调用系统的“`env`”、“`python3`”和“`rm`”等命令，理论上 `linux` 发行版均可满足要求。

`safe_rm` 的开发和测试操作系统为 `centos` 和 `rocky`。

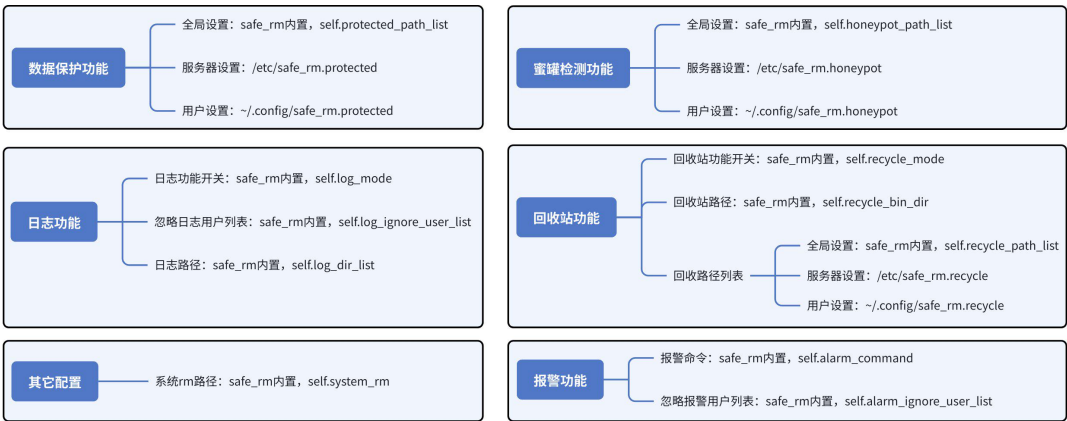
三、工具配置和引用

3.1 工具配置

safe_rm 是开箱即用型工具，**免安装，无需任何配置即可直接使用**。

如果想更好地使用路径保护功能/蜜罐检测功能/回收站功能/日志功能/报警功能，根据业务需求增加一些个性化的设置，则需要修改 safe_rm 的一些内置配置，或者增加一些配置文件。

下面用一张图来说明一下可配置项有哪些。



大多数配置项都在 safe_rm 脚本的 SafeRm 类的“__init__()”函数中配置，也有部分数据保护和回收站功能配置是在系统/用户下的配置文件中实现的。

3.1.1 保护路径配置

保护路径的配置分为 4 类。

- 全局设置：safe_rm 内置，self.protected_path_list，默认设置如下。

包含了用户 HOME 路径、系统根路径和系统文件路径。

```
# [Editable] self.protected_path_list : specify protected
paths (absolute path), globally effective.
self.protected_path_list = [
    '~',
    '/',
    '/bin',
    '/boot',
    '/dev',
```

```
    '/etc',  
    '/home',  
    '/initrd',  
    '/lib',  
    '/lib64',  
    '/mnt',  
    '/opt',  
    '/proc',  
    '/root',  
    '/run',  
    '/sbin',  
    '/srv',  
    '/sys',  
    '/usr',  
    '/var']
```

- **共享设置：**通过 `extend_list_with_config` 添加，一般为 `<共享目录>/safe_rm.protected`

依赖共享目录生效，效果同“全局设置”，但具备更高的可配置性，下面是一个例子。

```
/ic/data_trans
```

- **服务器设置：**通过 `extend_list_with_config` 添加，一般为 `/etc/safe_rm.protected`

建议指定服务器相关的受保护路径，下面是一个例子。（支持“*”符号，会做路径展开）

```
/ic/tech  
/ic/project/*
```

- **用户设置：**`extend_list_with_config`，一般为 `~/config/safe_rm.protected`

建议指定用户用户相关的受保护路径，下面是一个例子。

```
~/project_data  
~/project_data/syn_config
```

3.1.2 蜜罐路径配置

蜜罐路径的配置分为 3 部分。

- 全局设置：safe_rm 内置，self.honeypot_path_list，默认设置如下。

包含了“.honeypot”和“honeypot”两个默认的蜜罐文件名。

```
# [Editable] self.honeypot_path_list : specify honeypot
paths (file name or absolute path), globally effective.
self.honeypot_path_list = [
    '.honeypot',
    'honeypot']
```

共享设置/服务器设置/用户设置参照 #3.1.1。

3.1.3 回收站功能配置

safe_rm 中回收站功能用于保护次要数据，其相关的设置如下。

```
# [Editable] self.recycle_mode : enable recycle bin
mechnism, move the data to be deleted to the recycle bin.
# [Editable] self.recycle_dir : specify the recycle bin
path (will create self.recycle_dir/<USER> for current user).
# [Editable] self.recycle_path_list : specify recycle
paths (absolute path), the following files will be moved to the
recycle bin when they are deleted, globally effective.
self.recycle_mode = True
self.recycle_bin_dir = ''
self.recycle_path_list = [
    '~/.alias',
    '~/.bashrc',
    '~/.config',
    '~/.ssh']
```

self.recycle_mode: 指定是否打开回收站功能，默认打开，可以通过置为“False”关闭。打开后，safe_rm 并不会直接删除文件，而是选择将文件移入回收站。

self.recycle_bin_dir: 指定回收站路径，只有在 self.recycle_mode 为“True”的情况下生效，如未指定则默认为“/tmp/safe_rm/recycle_bin”。

self.recycle_path_list: 指定需要执行回收功能的目录，只有在 self.recycle_mode 为

“True”的情况下生效。如未指定则默认对所有路径生效。

下面是一个实例配置。(浅蓝色背景为修改部分)

```
self.recycle_mode = True
self.recycle_bin_dir =
'/ic/data/CAD/it/safe_rm/recycle_bin'
self.recycle_path_list = [
    '~/.alias',
    '~/.bashrc',
    '~/.config',
    '~/.ssh']
```

共享设置/服务器设置/用户设置参照 #3.1.1。

回收站路径列表中的路径可以被删除，但是 `self.recycle_path_list` 指定路径的数据会被自动移入回收站。

3.1.4 日志功能配置

`safe_rm` 中的日志功能用于记录所有的数据 `rm` 信息，其相关的设置如下。

```
# [Editable] self.log_mode : enable the behavior of saving
logs.
# [Editable] self.log_ignore_user_list : specify a user
list, will not save logs fro deletion operations of users in the
list.
# [Editable] self.log_dir_list : Specify log dir(s) to
save log file, make sure permission is "1777".
self.log_mode = True
self.log_ignore_user_list = []
self.log_dir_list = ['/tmp/safe_rm/log']
```

self.log_mode: 指定是否打开日志功能，默认打开，可以通过置为“False”关闭。

self.log_ignore_user_list: 指定忽略日志功能的用户，只有在 `self.log_mode` 为“True”的情况下生效。特殊系统用户可能有非常频繁的删除操作，记录日志的行为是没有意义的。

self.log_dir_list: 指定日志保存路径（可以配置多个），只有在 `self.log_mode` 为

“True”的情况下生效。如未指定则默认为“/tmp/safe_rm/log”。**log_dir 最好设置为 1777 权限，否则会出现用户执行 safe_rm 无法保存日志的问题。**

下面是一个实例配置。（浅蓝色背景为修改部分）

```
self.log_mode = True
self.log_ignore_user_list = []
self.log_dir_list = ['/tmp/safe_rm/log',
'/ic/data/CAD/it/safe_rm/log']
```

3.1.5 报警功能配置

safe_rm 中的报警功能用于为删除保护文件的行为发送报警，是一个安全功能（防止攻击行为），不配置则报警功能不生效。其相关的设置如下。

```
# [Editable] self.alarm_command : specify alarm command.
("<TITAL>", "<MESSAGE>" and "<USER>" are replaceable string.)
# [Editable] self.alarm_ignore_user_list : specify a user
list, no alarms will be sent fro deletion operations of users in
the list.
self.alarm_command = ''
self.alarm_ignore_user_list = []
```

self.alarm_command: 指定报警命令，如未指定，或者指定命令无法执行，则报警功能自动失效。self.alarm_command 中支持变量字符串“<MESSAGE>”，它会在执行报警命令时自动替换为对应的报警信息。

self.alarm_ignore_user_list: 指定忽略报警功能的用户，只有在 self.alarm_mode 为“True”的情况下生效。特殊系统用户可能有删除保护路径（包含系统和用户保护路径）的操作，这种行为有可能在特殊情况下是允许的。

下面是一个实例配置。（浅蓝色背景为修改部分）

```
self.alarm_command =
'/ic/software/cad_tools/bin/send_lark2 -T "Security Alarm: high
risk deletion" -c "<MESSAGE>" -r liyanqing.1987'
self.alarm_ignore_user_list = []
```

说明：示例中的 `send_lark2` 是内部自研工具，用于在研发环境中向指定用户的飞书发送消息。

3.1.6 其它配置

还可以通过如下命令配置系统“rm”命令的路径，也可以不配置，不配置的情况下，`safe_rm` 会默认从 `/usr/bin/system_rm`、`/usr/bin/rm`、`/bin/system_rm`、`/bin/rm` 中选择一个二进制文件来当做系统“rm”命令。

```
# [Editable] self.system_rm : specify system rm.
self.system_rm = ''
```

3.2 工具引用

针对使用场景的不同，`safe_rm` 推荐的引用方式有如下两种。

3.2.1 单用户场景

典型的应用场景是 root 用户使用，防止权限过大误删重要数据。

这种情况下，推荐配置用户 `alias` 将“rm”命令指向 `safe_rm`，这样可以实现轻量化的命令替代。为了使这个 `alias` 持久生效，可以配置在用户的“`~/.alias`”配置文件中。

```
[root@ic-monitor01 ~]# cat ~/.alias
alias rm='/ic/software/cad_tools/it/tools/safe_rm'
[root@ic-monitor01 ~]#
[root@ic-monitor01 ~]# source ~/.alias
[root@ic-monitor01 ~]# which rm
alias rm='/ic/software/cad_tools/it/tools/safe_rm'
      /ic/software/cad_tools/it/tools/safe_rm
```

3.2.2 全服务器用户场景

为了使 `safe_rm` 对当前服务器上的全体用户生效，需要用 `safe_rm` 取代系统默认的“rm”命令，同时确保它保持 755 的权限。

注意，这个操作只有通过 root 账号来执行。

```
[root@ic-monitor01 ~]# which rm
/usr/bin/rm
[root@ic-monitor01 ~]# mv /usr/bin/rm /usr/bin/system_rm
[root@ic-monitor01 ~]# cp -f
/ic/software/cad_tools/it/tools/safe_rm /usr/bin/rm
[root@ic-monitor01 ~]# chmod 755 /usr/bin/rm
```

同时将 `safe_rm` 中的 `self.system_rm` 指向真正的系统“rm”。

```
self.system_rm = '/usr/bin/system_rm'
```

当然，如果真正的 `rm` 命令“`/usr/bin/rm`”被迁移到“`/usr/bin/sytem_rm`”，那么此处无需指定它也可以自己找到的。

3.3 工具打包

`safe_rm` 本身是一个 `python` 脚本，明文存放，配置可读可篡改，在安全场景下使用存在一定的安全漏洞，因此建议在实际使用的时候，将 `safe_rm` 使用 `pyinstaller` 打包为不可读的独立可执行文件使用。

```
[root@ic-monitor01 install]# pyinstaller --onefile safe_rm
5 DEPRECATION: Running PyInstaller as root is not necessary nor
sensible. Do not use PyInstaller with sudo. PyInstaller 7.0 will
block this.
871 INFO: PyInstaller: 6.11.1, contrib hooks: 2025.1
...
11708 INFO: Copying bootloader EXE to /root/install/dist/safe_rm
11710 INFO: Appending PKG archive to custom ELF section in EXE
11752 INFO: Building EXE from EXE-00.toc completed successfully.
```

生成的目录结构如下。

```
[root@ic-monitor01 install]# ls
build  dist  safe_rm  safe_rm.spec
```

其中关键打包文件位于 `dist` 目录下，尺寸约为 17M，使用此文件更安全。

```
[root@ic-monitor01 install]# du -hs dist/safe_rm  
17M      dist/safe_rm
```

四、工具使用示例

下面演示两种主要的使用实例，实操过程中可以参照实施。

4.1 单用户场景

场景：重要的管理节点，root 为主要用户，需要防止 root 误操作删除重要数据。

配置：

- safe_rm 相关的配置，同#3.1。
- 引用设置同#3.2.1：

用户 rm 确认：

实际使用的 rm 是个 alias，指向“/ic/software/cad_tools/it/tools/safe_rm”。

```
[root@ic-monitor01 ~]# which rm
alias rm='/ic/software/cad_tools/it/tools/safe_rm'
      /ic/software/cad_tools/it/tools/safe_rm
```

操作：

- 删除普通数据（行为如常）

```
[root@ic-monitor01 ~]# cd
[root@ic-monitor01 ~]# ls
test_1.txt  test_2
[root@ic-monitor01 ~]# rm -rf test_1.txt test_2/
[root@ic-monitor01 ~]# ls
[root@ic-monitor01 ~]#
```

- 删除保护数据

```
[root@ic-monitor01 ~]# cd
[root@ic-monitor01 ~]# ls
project_data
[root@ic-monitor01 ~]# rm -rf /
*Warning*: Cannot remove protected path "/", skip!
```

我们看到删除保护路径“/”的行为则被直接拒绝，同时，由于设置了报警功能，还

会收到如下飞书报警信息。

Security Alarm: high risk deletion

Time: 2025-03-03 14:37:31
User: root
Host: ic-monitor01
Cwd: /root
Command: rm -rf / (skip)

- 删除蜜罐数据

```
[root@ic-monitor01 ~]# mkdir test
[root@ic-monitor01 ~]# touch test/.honeypot
[root@ic-monitor01 ~]# rm -rf test
```

我们看到 **test** 目录被正常删除，但是由于其下一层目录包含了蜜罐文件，因此会收到飞书报警。

Security Alarm: high risk deletion

Time: 2025-03-03 14:40:28
User: root
Host: ic-monitor01
Cwd: /root
Command: rm -rf test (honeypot)

- 删除可回收数据

```
[root@ic-monitor01 ~]# ls ~/.config
abrt gconf JetBrains safe_rm.protected safe_rm.recycle
[root@ic-monitor01 ~]# rm -rf ~/.config/abrt
[root@ic-monitor01 ~]# ls ~/.config
gconf JetBrains safe_rm.protected safe_rm.recycle
[root@ic-monitor01 ~]# ls
/ic/software/cad_data/it/safe_rm/recycle_bin/root/
abrt
```

我们可以看到，“**~/.config/abrt**”确实被删除了，并被转移到回收站“**/ic/software/cad_data/it/safe_rm/recycle_bin/root**”下面。

如果同名数据被反复删除，回收站下面则会为后保存的数据打上时间戳。

```
[root@ic-monitor01 ~]# mkdir ~/.config/abrt
[root@ic-monitor01 ~]# rm -rf ~/.config/abrt
[root@ic-monitor01 ~]# ls
/ic/software/cad_data/it/safe_rm/recycle_bin/root
abrt  abrt.20240426172834
```

4.2 全服务器用户场景

场景：普通节点，普通用户是主要用户，希望能够为用户数据提供一些防误删保障。

配置：

- `safe_rm` 相关配置，同#3.1。
- 引用设置同#3.2.2。

用户 `rm` 确认：

实际使用的确实是 `/usr/bin/rm`，也就是 `safe_rm`。

```
[liyanqing.1987@ic-monitor01 ~]$ which rm
/bin/rm
[liyanqing.1987@ic-monitor01 ~]$ ls -al /bin
lrwxrwxrwx. 1 root root 7 Dec 27 2020 /bin -> usr/bin
```

操作：

- 删除普通数据（行为如常）

```
[liyanqing.1987@ic-monitor01 ~]$ cd
[liyanqing.1987@ic-monitor01 ~]$ ls
bin Desktop important_backup project_data scripts_cad_tools_bk
test_1.txt test_2.log test_3 tools
[liyanqing.1987@ic-monitor01 ~]$ rm -rf test_1.txt test_2.log
test_3
[liyanqing.1987@ic-monitor01 ~]$ ls
bin Desktop important_backup project_data scripts_cad_tools_bk
tools
```

- 删除保护数据

```
[liyanqing.1987@ic-monitor01 ~]$ rm -rf project_data
*Warning*: Cannot remove protected path "project_data", skip!
```

我们看到删除保护路径“project_data”的行为则被直接拒绝，同时由于设置了报警功能，还会收到如下飞书报警信息。

Security Alarm: high risk deletion

Time: 2025-03-03 14:41:30
User: liyanqing.1987(root)
Host: ic-monitor01
Cwd: /home/liyanqing.1987
Command: rm -rf project_data (skip)

- 删除蜜罐数据

```
[liyanqing.1987@ic-monitor01 ~]$ mkdir test
[liyanqing.1987@ic-monitor01 ~]$ touch test/.honeypot
[liyanqing.1987@ic-monitor01 ~]$ rm -rf test
```

我们看到 test 目录被正常删除，但是由于其下一层目录包含了蜜罐文件，因此会收到飞书报警。

Security Alarm: high risk deletion

Time: 2025-03-03 14:44:17
User: liyanqing.1987(root)
Host: ic-monitor01
Cwd: /home/liyanqing.1987
Command: rm -rf test (honeypot)

- 删除可回收文件

```
[liyanqing.1987@ic-monitor01 ~]$ ls ~/.config | grep Code
Code
[liyanqing.1987@ic-monitor01 ~]$ rm -rf ~/.config/Code
[liyanqing.1987@ic-monitor01 ~]$ ls ~/.config | grep Code
[liyanqing.1987@ic-monitor01 ~]$ ls
/ic/software/cad_data/it/safe_rm/recycle_bin/liyanqing.1987/
```


Code

我们可以看到，“~/config/Code”确实被删除了，并被转移到回收站“/ic/software/cad_data/it/safe_rm/recycle_bin/liyanqing.1987”下面。

五、高级功能

5.1 日志检索

比如执行如下 `rm` 操作，有正常删除，也有保护数据的违规删除，还有回收机制数据的删除。

```
[liyanqing.1987@ic-monitor01 ~]$ rm -rf test.txt test.log /
~ .config/abrt
*Warning*: Cannot remove protected path "/", skip!
*Warning*: Cannot remove protected path "/home/liyanqing.1987",
skip!
```

那么在 `safe_rm` 的用户日志中，关键词“(skip)”和“(recycle_command)”是需要特殊关注的，前者代表了尝试删除保护文件，有一定的安全风险，后者代表了删除需回收数据，可以确认下是否需要找回。

```
{"time": "2025-03-03 14:37:31", "message_level": "Info", "user":
"root", "login_user": "root", "host": "ic-monitor01", "cwd":
"/root", "message": "[command=rm -rf /]"}
{"time": "2025-03-03 14:37:31", "message_level": "Warning",
"user": "root", "login_user": "root", "host": "ic-monitor01",
"cwd": "/root", "message": "*Warning*: Cannot remove protected
path \"/\", skip!"}
{"time": "2025-03-03 14:37:31", "message_level": "Info", "user":
"root", "login_user": "root", "host": "ic-monitor01", "cwd":
"/root", "message":
"[alarm_command=/ic/software/cad_tools/bin/send_lark -T \"Security
Alarm: high risk deletion\" -c \"Time: 2025-03-03 14:37:31\n User:
root\n Host: ic-monitor01\n Cwd: /root\n Command: rm -rf /
(skip)\n -r liyanqing.1987]"}
...
{"time": "2025-03-03 14:49:24", "message_level": "Info", "user":
"root", "login_user": "root", "host": "ic-monitor01", "cwd":
"/root", "message": "[command=rm .alias]"}
{"time": "2025-03-03 14:49:24", "message_level": "Info", "user":
"root", "login_user": "root", "host": "ic-monitor01", "cwd":
"/root", "message": "[recycle_command=/bin/cp -f .alias
/ic/data/CAD/it/safe_rm/recycle_bin/root/.alias]"}

```

5.2 debug 功能

设置环境变量“**SAFE_RM_DEBUG**”可以开启 `safe_rm` 的 debug 模式，主要是打印更多的信息，或者开启虚拟删除操作。

值	行为
1	打印 <code>safe_rm</code> 本身的 warning，主要是一些行为故障，比如无法保存 log 等。
2	输出内部执行的实际命令。
3	打印出各种配置信息。
4	虚拟执行 <code>rm</code> 和 <code>mv</code> ，但是并不会真正执行。

示例，`SAFE_RM_DEBUG=1`

```
[liyanqing.1987@ic-monitor01 ~]$ export SAFE_RM_DEBUG=1
[liyanqing.1987@ic-monitor01 ~]$ rm a
[WARNING] Specified system rm "/usr/bin/system_rm" is missing.
```

示例，`SAFE_RM_DEBUG=2`

```
[liyanqing.1987@ic-monitor01 ~]$ export SAFE_RM_DEBUG=2
[liyanqing.1987@ic-monitor01 ~]$ rm b
[COMMAND] /usr/bin/system_rm b
```

示例，`SAFE_RM_DEBUG=3`

```
[liyanqing.1987@ic-monitor01 ~]$ export SAFE_RM_DEBUG=3
[liyanqing.1987@ic-monitor01 ~]$ rm c
[CONFIG] protected_path_list : ['/', '/bin', '/boot', '/dev', ...]
[CONFIG] honeypot_path_list : ['.honeypot', 'honeypot']
[CONFIG] recycle_mode : True
[CONFIG] recycle_dir : /ic/data/CAD/it/safe_rm/recycle_bin
[CONFIG] recycle_path_list : ['/home/liyanqing.1987/.alias',
'/home/liyanqing.1987/.bashrc', '/home/liyanqing.1987/.config',
```

```
'/home/liyanqing.1987/.ssh']  
[CONFIG] log_dir(s) : ['/ic/data/CAD/it/safe_rm/log']  
[CONFIG] alarm_command : /ic/software/cad_tools/bin/send_lark -T  
"Security Alarm: high risk deletion" -c "<MESSAGE>" -r  
liyanqing.1987  
[CONFIG] system_rm : /usr/bin/system_rm  
[COMMAND] /usr/bin/system_rm c
```

示例, SAFE_RM_DEBUG=4

```
[liyanqing.1987@ic-monitor01 ~]$ export SAFE_RM_DEBUG=4  
[liyanqing.1987@ic-monitor01 ~]$ rm d  
...  
[CONFIG] system_rm : /usr/bin/systemd-rm  
[COMMAND] /usr/bin/systemd-rm d  
[liyanqing.1987@ic-monitor01 ~]$  
[liyanqing.1987@ic-monitor01 ~]$ ls d  
d
```

5.3 登陆用户识别

如果 root 用户切换为其它用户, 并尝试危险删除动作, 这个时候可以识别出 login user 和 current user, 并加以标注。

下面的实例中, root 用户切换成 liyanqing.1987, 并尝试删除受保护数据。

```
[root@ic-monitor01 ~]# su - liyanqing.1987  
Last login: Wed Jun 19 12:55:41 CST 2024 on pts/0  
Welcome liyanqing.1987 ~  
[liyanqing.1987@ic-monitor01 ~]$ rm /  
*Warning*: Cannot remove protected path "/", skip!
```

收到的报警信息如下, User 为“liyanqing.1987(root)”, 即 liyanqing.1987 是由 root 用户切换而来。

Security Alarm: high risk deletion

Time: 2025-03-03 16:19:49

User: liyanqing.1987(root)

Host: ic-monitor01

Cwd: /home/liyanqing.1987

Command: rm / (skip)

同时在 log 中也会明确标识出 user 和 login_user。

```
{"time": "2025-03-03 16:21:11", "message_level": "Info", "user":  
"liyanqing.1987", "login_user": "root", "host": "ic-monitor01",  
"cwd": "/home/liyanqing.1987", "message": "[command=rm /]"}
```

六、行为测试

safe_rm 在 Linux 系统中正式部署前，需要测试如下设置是否符合预期。

测试项	设置	期望结果
debug 设置 SAFE_RM_DEBUG	export SAFE_RM_DEBUG=1	输出如下[WARNING]信息： [WARNING] Specified system rm "xxx" is missing.
	export SAFE_RM_DEBUG=2	输出[WARNING]和[COMMAND]信息， [COMMAND]信息如下： [COMMAND] /usr/bin/system_rm a
	export SAFE_RM_DEBUG=3	输出[WARNING]、[COMMAND]和 [CONFIG]信息，[CONFIG]信息如下： [CONFIG] recycle_mode : True
	export SAFE_RM_DEBUG=4	打印所有如上信息，但是数据并不被真实删除（仅虚拟行为）
参数测试	执行 rm	输出如下信息： rm: missing operand Try 'rm --help' for more information.
	执行 rm --version	输出为空如下信息： rm (GNU coreutils) xxx Copyright (C) 2013 Free Software Foundation, Inc. ...
参数测试 (系统 rm 丢失)	执行 rm	输出如下信息： rm: missing operand Try 'rm --help' for more information.
	执行 rm --version	无任何信息输出

删除测试	rm xxx 删除普通文件	文件被删除
	rm -r xxx 删除普通目录	目录被删除
	rm "f f" 删除带空格的文件	文件被删除
	rm 'f'f' 删除带引号的文件	文件被删除
	rm xxx 删除不存在的文件	输出如下信息： rm: cannot remove 'xxx': No such file or directory
	rm xxx 删除无权限的文件 (文件上层目录没有写权限)	文件未被删除，输出如下信息： rm: cannot remove 'xxx': Permission denied
	删除保护数据	数据未被删除，输出如下信息： <i>Warning: Cannot remove protected path "xxx", skip!</i> 同时会收到报警（如配置）
	删除保护数据的上层路径	数据未被删除，输出如下信息： <i>Warning: Cannot remove path "xxx", there are protected data under it, skip!</i> 同时会收到报警（如配置）
	删除保护数据的下层路径	文件被删除
	删除蜜罐文件（或其上层路径）	文件被删除，无任何信息输出，会收到报警（如配置）
	删除回收站保护数据	文件被删除，无任何信息输出，在回收站中

		可以找到被删除的文件。
删除测试 (系统 rm 丢失)	rm xxx 删除普通文件	文件被删除
	rm -r xxx 删除普通目录	目录被删除
	rm "f f" 删除带空格的文件	文件被删除
	rm 'f'f' 删除带引号的文件	文件被删除
	rm xxx 删除不存在的文件	输出如下信息： rm: cannot remove 'xxx': No such file or directory
	rm xxx 删除无权限的文件 (文件上层目录没有写权限)	文件未被删除，输出如下信息： rm: cannot remove 'xxx': Permission denied
	删除保护数据	数据未被删除，输出如下信息： <i>Warning: Cannot remove protected path "xxx", skip!</i> 同时会收到报警（如配置）
	删除保护数据的上层路径	数据未被删除，输出如下信息： <i>Warning: Cannot remove path "xxx", there is protected data under it, skip!</i> 同时会收到报警（如配置）
	删除保护数据的下层路径	文件被删除
	删除蜜罐文件（或其上层路径）	文件被删除，无任何信息输出，会收到报警（如配置）

	删除回收站保护数据	文件被删除，无任何信息输出，在回收站中可以找到被删除的文件。
--	-----------	--------------------------------

七、技术支持

本工具为开源工具，由开源社区维护，可以提供如下类型的技术支持：

- 部署和使用技术指导。
- 接收 bug 反馈并修复。
- 接收功能修改建议。（需审核和排期）

获取技术支持的方式包括：

- 通过 Contact 邮箱联系开发者。
- 添加作者微信“liyanqing_1987”，注明“真实姓名/公司/safe_rm”，由作者拉入技术支持群。



附录

附 1. 变更历史

日期	版本	变更描述	备注
2024.07	1.0	发布第一个版本 <code>safe_rm</code> 。	
2025.02	1.1	增加蜜罐文件检测和报警功能。 支持多日志路径。	
2025.03	1.2	内置增加数据删除功能，可不依赖于系统 <code>rm</code> 。	
2025.11	1.2.1	修复了系统 <code>rm</code> 缺失情况下，无法删除带空格、引号等特殊字符文件的问题。	