



链滴

5-PAM 认证机制

作者: [Carey](#)

原文链接: <https://ld246.com/article/1610715290706>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)



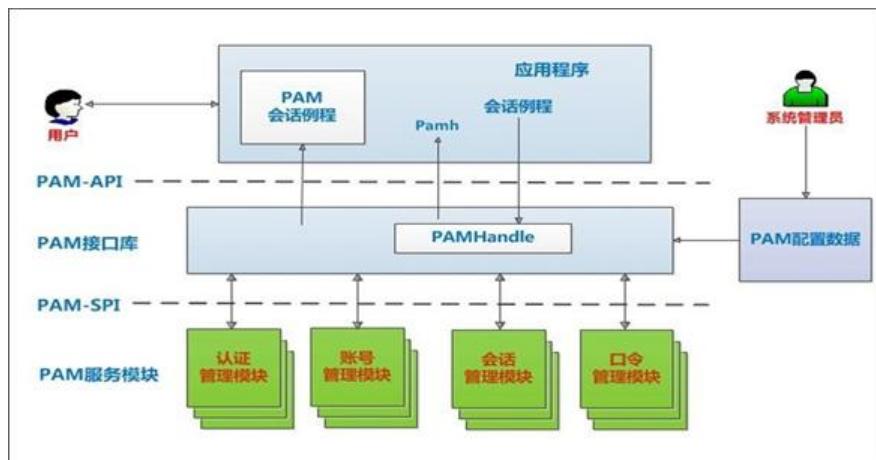
6 PAM认证机制

6.1 PAM介绍

认证库：文本文件，MySQL，NIS，LDAP等

PAM：Pluggable Authentication Modules，插件式的验证模块，Sun公司于1995年开发的一种与认证相关的通用框架机制。PAM只关注如何为服务验证用户的API，通过提供一些动态链接库和一套统一的API，将系统提供的服务和该服务的认证方式分开，使得系统管理员可以灵活地根据需要给不同服务配置不同的认证方式而无需更改服务程序一种认证框架，自身不做认证

6.2 PAM架构



PAM提供了对所有服务进行认证的中央机制，适用于本地登录，远程登录，如：telnet,rlogin,fs,ftp点对点协议PPP, su等应用程序中，系统管理员通过PAM配置文件来制定不同应用程序的不同认证策略；应用程序开发者通过在服务程序中使用PAM API(pam_xxxx())来实现对认证方法的调用；而PAM

务模块的开发者则利用PAM SPI来编写模块（主要调用函数pam_sm_xxxx()供PAM接口库调用，将同的认证机制加入到系统中；PAM接口库（libpam）则读取配置文件，将应用程序和相应的PAM服务模块联系起来

6.3 PAM相关文件

- 包名: pam
- 模块文件目录: /lib64/security/*.so
- 特定模块相关的设置文件: /etc/security/
- 应用程序调用PAM模块的配置文件
 - 主配置文件: /etc/pam.conf, 默认不存在，一般不使用主配置
 - 为每种应用模块提供一个专用的配置文件: /etc/pam.d/APP_NAME
 - 注意: 如/etc/pam.d存在, /etc/pam.conf将失效

6.3.1 范例：查看程序是否支持PAM

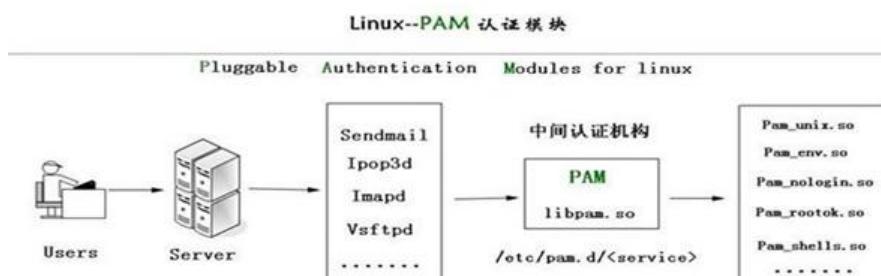
```
[14:32:28 root@centos8 ~]#ldd `which sshd` | grep libpam  
libpam.so.0 => /lib64/libpam.so.0 (0x00007f0e3e4d8000)  
[14:44:42 root@centos8 ~]#ldd `which passwd` | grep libpam  
libpam.so.0 => /lib64/libpam.so.0 (0x00007f5c1ffab000)  
libpam_misc.so.0 => /lib64/libpam_misc.so.0 (0x00007f5c1fda7000)
```

```
#不支持PAM  
[14:44:52 root@centos8 ~]#ldd /usr/sbin/httpd | grep libpam
```

6.4 PAM工作原理

PAM认证一般遵循这样的顺序: Service(服务)→PAM(配置文件)→pam_*.so

PAM认证首先要确定那一项服务，然后加载相应的PAM的配置文件(位于/etc/pam.d下)，最后调用认证文件(位于/lib64/security下)进行安全认证



PAM认证过程示例：

1. 使用者执行/usr/bin/passwd 程序，并输入密码
2. passwd开始调用PAM模块，PAM模块会搜寻passwd程序的PAM相关设置文件，这个设置文件一是在/etc/pam.d/里边的与程序同名的文件，即PAM会搜寻/etc/pam.d/passwd此设置文件
3. 经由/etc/pam.d/passwd设定文件的数据，取用PAM所提供的相关模块来进行验证
4. 将验证结果回传给passwd这个程序，而passwd这个程序会根据PAM回传的结果决定下一个动作

重新输入密码或者通过验证)

6.5 PAM配置文件格式说明

通用配置文件/etc/pam.conf格式，此格式不使用

application type control module-path arguments

专用配置文件/etc/pam.d/格式

type control module-path arguments

参数说明：

application: 指服务名，如：telnet、login、ftp等，服务名字“OTHER”代表所有没有在该文件中确配置的其它服务

type: 指模块类型，即功能

control : PAM库该如何处理与该服务相关的PAM模块的成功或失败情况，一个关键词实现

module-path: 用来指明本模块对应的程序文件的路径名

Arguments: 用来传递给该模块的参数

模块类型(module-type)

- Auth 账号的认证和授权
- Account 帐户的有效性，与账号管理相关的非认证类的功能，如：用来限制/允许用户对某个服务访问时间，限制用户的位置(例如：root用户只能从控制台登录)
- Password 用户修改密码时密码复杂度检查机制等功能
- Session 用户会话期间的控制，如：最多打开的文件数，最多的进程数等
- -type 表示因为缺失而不能加载的模块将不记录到系统日志,对于那些不总是安装在系统上的模块有用

Control:

- required : 一票否决，表示本模块必须返回成功才能通过认证，但是如果该模块返回失败，失败结果不会立即通知用户，而是要等到同一type中的所有模块全部执行完毕，再将失败结果返回给应用程序，即为必要条件
- requisite : 一票否决，该模块必须返回成功才能通过认证，但是一旦该模块返回失败，将不再执行同一type内的任何模块，而是直接将控制权返回给应用程序。是一个必要条件
- sufficient : 一票通过，表明本模块返回成功则通过身份认证的要求，不必再执行同一type内的其他模块，但如果本模块返回失败可忽略，即为充分条件，优先于前面的required和requisite
- optional : 表明本模块是可选的，它的成功与否不会对身份认证起关键作用，其返回值一般被忽略
- include: 调用其他的配置文件中定义的配置信息

module-path:

- 模块文件所在绝对路径：
- 模块文件所在相对路径：/lib64/security目录下的模块可使用相对路径，如：pam_shells.so、pam_limits.so
- 有些模块有自己的专有配置文件，在/etc/security/*.conf目录下

Arguments

- debug：该模块应当用syslog()将调试信息写入到系统日志文件中no_warn：表明该模块不应把告信息发送给应用程序
- use_first_pass：该模块不能提示用户输入密码，只能从前一个模块得到输入密码
- try_first_pass：该模块首先用前一个模块从用户得到密码，如果该密码验证不通过，再提示用户输入新密码
- use_mapped_pass 该模块不能提示用户输入密码，而是使用映射过的密码
- expose_account 允许该模块显示用户的帐号名等信息，一般只能在安全的环境下使用，因为泄用户名会对安全造成一定程度的威胁

注意：修改PAM配置文件将马上生效

建议：编辑pam规则时，保持至少打开一个root会话，以防止root身份验证错误

6.6 PAM模块帮助

官方在线文档：<http://www.linux-pam.org/Linux-PAM-html/>

官方离线文档：<http://www.linux-pam.org/documentation/>

pam模块文档说明：/user/share/doc/pam-*

rpm -qd pam

man -k pam_

man 模块名 如：man 8 rootok

6.7 常用PAM模块

6.7.1 pam_shells模块

功能：检查有效shell

帮助：man pam_shells

案例：不允许使用/bin/csh的用户本地登录

```
[14:45:11 root@centos8 ~]#yum install csh -y
```

```
[14:56:14 root@centos8 ~]#vim /etc/pam.d/login #这个如果设置表示主机终端tty无法登录，  
他方式是可以登录的
```

```
[19:11:49 root@centos8 ~]#vim /etc/pam.d/sshd #设置这个表示使用ssh连接的无法登录  
#添加下面这行
```

```
auth required pam_shells.so  
#这个模块会检查/etc/shells文件中的shell是否合法，不存在或者注释表示不合法
```

```
[14:58:00 root@centos8 ~]#vim /etc/shells  
去掉/bin/csh
```

```
[14:58:14 root@centos8 ~]#useradd -s /usr/bin/csh testuser  
#testuser将不可登录  
[15:23:00 root@centos8 ~]#tail -f /var/log/secure #可以查看登录日志
```

6.7.2 pam_securetty模块

功能：只允许root用户在/etc/securetty列出的安全终端上登陆

案例：CentOS 7 允许root在telnet登陆

```
[15:26:38 root@centos8 ~]#vim /etc/pam.d/remote  
#将下面一行加上注释  
#auth required pam_securetty.so  
#或者/etc/securetty文件中加入  
pts/0,pts/1...pts/n  
  
#测试用root telnet登录
```

案例：在Centos8上实现pam_securetty.so模块禁止root远程登录telnet服务

```
#默认CentOS8 虽然有这模块,但没有使用此模块,所以允许root远程telnet登录  
[15:30:28 root@centos8 ~]#telnet 192.168.10.81  
Trying 192.168.10.81...  
Connected to 192.168.10.81.  
Escape character is '^]'.
```

```
Kernel 4.18.0-193.el8.x86_64 on an x86_64  
centos8 login: root  
Password:  
Last failed login: Thu Jan 14 15:34:00 CST 2021 from ::ffff:192.168.10.1 on pts/2  
There was 1 failed login attempt since the last successful login.  
Last login: Thu Jan 14 15:33:43 from ::ffff:192.168.10.1  
#修改配置不允许root远程telnet登录  
[15:35:45 root@centos8 ~]#vim /etc/pam.d/remote  
#添加此行  
auth required pam_securetty.so  
#创建空文件,CentOS8上默认不存在此文件  
[15:38:02 root@centos8 ~]#touch /etc/securetty  
  
#测试  
[15:38:06 root@centos8 ~]#telnet 192.168.10.81
```

6.7.3 pam_nologin.so模块

功能：如果/etc/nologin文件存在,将导致非root用户不能登陆,当该用户登陆时，会显示/etc/nologin文件内容，并拒绝登陆，要想再次登录删除/etc/nologin文件就可以了（适合服务器维护时使用）

```
[19:16:37 root@centos8 ~]#vim /etc/pam.d/sshd  
auth required pam_nologin.so #添加  
#生成nologin文件, 只要生成这个文件除了root用户其他用户都不可以登录  
[19:24:21 root@centos8 ~]#touch /etc/nologin  
#登录失败提醒设置, 登录失败后提醒  
[19:25:36 root@centos8 ~]#echo zhangzhuo > /etc/nologin
```

6.7.4 pam_limits.so模块（重点服务器优化必要）

功能：在用户级别实现对其可使用的资源的限制，例如：可打开的文件数量，可运行的进程数量，可内存空间

-H	设置硬资源限制，一旦设置不能增加。	ulimit - Hs 64；限制硬资源，线程栈大小为 64K。
-S	设置软资源限制，设置后可以增加，但是不能超过硬资源设置。	ulimit - Sn 32；限制软资源，32 个文件描述符。
-a	显示当前所有的 limit 信息。	ulimit - a；显示当前所有的 limit 信息。
-c	最大的 core 文件的大小，以 blocks 为单位。	ulimit - c unlimited；对生成的 core 文件的大小不进行限制。
-d	进程最大的数据段的大小，以 Kbytes 为单位。	ulimit - d unlimited；对进程的数据段大小不进行限制。
-f	进程可以创建文件的最大值，以 blocks 为单位。	ulimit - f 2048；限制进程可以创建的最大文件大小为 2048 blocks。
-l	最大可加锁内存大小，以 Kbytes 为单位。	ulimit - l 32；限制最大可加锁内存大小为 32 Kbytes。
-m	最大内存大小，以 Kbytes 为单位。	ulimit - m unlimited；对最大内存不进行限制。
-n	可以打开最大文件描述符的数量。	ulimit - n 128；限制最大可以使用 128 个文件描述符。
-p	管道缓冲区的大小，以 Kbytes 为单位。	ulimit - p 512；限制管道缓冲区的大小为 512 Kbytes。
-s	线程栈大小，以 Kbytes 为单位。	ulimit - s 512；限制线程栈的大小为 512 Kbytes。
-t	最大的 CPU 占用时间，以秒为单位。	ulimit - t unlimited；对最大的 CPU 占用时间不进行限制。
-u	用户最大可用的进程数。	ulimit - u 64；限制用户最多可以使用 64 个进程。
-v	进程最大可用的虚拟内存，以 Kbytes 为单位。	ulimit - v 200000；限制最大可用的虚拟内存为 200000 Kbytes。

修改限制的实现方式：

(1) ulimit命令

ulimit是linux shell的内置命令，它具有一套参数集，用于对shell进程及其子进程进行资源限制。ulimit的设定值是 per-process 的，也就是说，每个进程有自己的limits值。

使用ulimit进行修改，立即生效。

ulimit只影响shell进程及其子进程，用户登出后失效。

可以在profile中加入ulimit的设置，变相的做到永久生效。

-H 设置硬件资源限制。
-S 设置软件资源限制。
-a 显示当前所有的资源限制。
-c size:设置core文件的最大值.单位:blocks
-d size:设置数据段的最大值.单位:kbytes
-f size:设置创建文件的最大值.单位:blocks
-l size:设置在内存中锁定进程的最大值.单位:kbytes
-m size:设置可以使用的常驻内存的最大值.单位:kbytes
-n size:设置内核可以同时打开的文件描述符的最大值.单位:n
-p size:设置管道缓冲区的最大值.单位:kbytes
-s size:设置堆栈的最大值.单位:kbytes
-t size:设置CPU使用时间的最大上限.单位:seconds
-u size:最大用户进程数
-v size:设置虚拟内存的最大值.单位:kbytes unlimited 是一个特殊值，用于表示不限制

#说明

查询时，若不加H或S参数，默认显示的是软限制

修改时，若不加H或S参数，两个参数一起改变

(2) 配置文件：

pam_limits的设定值是基于 per-process 的

/etc/security/limits.conf

/etc/security/limits.d/*.conf

配置文件格式：

```
#每行一个定义  
<domain> <type> <item> <value>
```

格式说明：

应用与那些对象

Username 单个用户

@group 组内所有用户

* 所有用户

% 仅用于限制 maxlogins limit , 可以使用 %group 语法. 只用 % 相当于 * 对所有用户maxsyslogins limit限制. %group 表示限制此组中的所有用户总的最大登录数

限制类型

Soft 软限制,普通用户自己可以修改

Hard 硬限制,由root用户设定,且通过kernel强制生效

- 二者同时限定

限制的资源

nofile 所能够同时打开的最大文件数量,默认为1024

nproc 所能够同时运行的进程的最大数量,默认为1024

指定具体值

案例：系统的各种资源的默认值

```
[15:38:52 root@centos8 ~]#ulimit -a  
core file size      (blocks, -c) unlimited #unlimited表示不限制  
data seg size       (kbytes, -d) unlimited  
scheduling priority (-e) 0  
file size          (blocks, -f) unlimited  
pending signals     (-i) 3586  
max locked memory   (kbytes, -l) 64  
max memory size    (kbytes, -m) unlimited  
open files          (-n) 1024      #一个进程可以打开的最大文件数  
pipe size          (512 bytes, -p) 8  
POSIX message queues (bytes, -q) 819200  
real-time priority  (-r) 0  
stack size          (kbytes, -s) 8192  
cpu time            (seconds, -t) unlimited  
max user processes  (-u) 3586      #一个用户可以开启进程/线程的最大数目  
virtual memory      (kbytes, -v) unlimited  
file locks          (-x) unlimited
```

案例：ulimit命令修改用户打开的文件个数

```
[15:45:07 root@centos8 ~]#ulimit -n  
1024
```

```
[15:45:56 root@centos8 ~]#ulimit -n 1048577
-bash: ulimit: open files: cannot modify limit: Operation not permitted
[15:46:02 root@centos8 ~]#ulimit -n 1048576
[15:46:08 root@centos8 ~]#ulimit -a
core file size      (blocks, -c) unlimited
data seg size       (kbytes, -d) unlimited
scheduling priority (-e) 0
file size           (blocks, -f) unlimited
pending signals     (-i) 3586
max locked memory   (kbytes, -l) 64
max memory size     (kbytes, -m) unlimited
open files          (-n) 1048576
pipe size           (512 bytes, -p) 8
POSIX message queues (bytes, -q) 819200
real-time priority  (-r) 0
stack size          (kbytes, -s) 8192
cpu time            (seconds, -t) unlimited
max user processes   (-u) 3586
virtual memory       (kbytes, -v) unlimited
file locks           (-x) unlimited
[15:46:16 root@centos8 ~]#echo 2^20 | bc
1048576
```

案例：限制用户最多打开的文件数和运行进程数，并持久保存

```
[15:46:32 root@centos8 ~]#cat /etc/pam.d/system-auth
session    required    pam_limits.so
```

```
[15:47:28 root@centos8 ~]#vim /etc/security/limits.conf
#apache可打开10240个文件
apache - nproc 10240
#cy不能运行超过20个进程
cy hard nproc 10
#用cy登录多次运行bash，观察结果
```

```
[16:02:03 root@centos8 ~]#cat /etc/security/limits.conf
zhang - nproc 6666
zhang - nporc 5
cy - nproc 8888
[16:02:07 root@centos8 ~]#su - zhang
Last login: Thu Jan 14 14:31:59 CST 2021 on pts/1
[zhang@centos8 ~]$ ulimit -n
6666
```

案例：限制zhang用户最大的同时登录次数

```
[16:04:03 root@centos8 ~]#tail -n1 /etc/security/limits.conf
zhang - maxlogins 2
[16:04:33 root@centos8 ~]#who
root  pts/0    2021-01-14 15:58 (192.168.10.1)
root  pts/1    2021-01-14 15:58 (192.168.10.1)
zhang pts/2    2021-01-14 16:04 (192.168.10.1)
zhang pts/3    2021-01-14 16:04 (192.168.10.1)
```

生成案例：（重点）

```
[zhang@centos8 ~]$ tail -n5 /etc/security/limits.conf
```

```
* - core    unlimited  
* - nproc   1000000 #最大进程数  
* - nofile  1000000 #打开文件描述符最大数目  
* - memlock 32000  
* - msgqueue 8192000
```

6.7.5 pam_succeed_if模块

功能：根据参数中的所有条件都满足才返回成功

案例：ubuntu默认不允许root登录桌面图形

用root登录桌面失败，查看日志，可看到Pam原因

```
Vim /etc/pam.d/gdm-password #将下面行注释  
#auth required pam_succeed_if.so user !=root quiet_success
```

6.7.6 pam_google_authenticator模块

什么是 MFA？

Multi-Factor Authentication (MFA) 是一种简单有效的最佳安全实践方法，它能够在用户名和密码外再额外增加一层安全保护。

功能：实现SSH登录的两次身份验证，先验证APP的数字码，再验证root用户的密码，都通过才可以登录。

官方网站：<https://github.com/google/google-authenticator-android>

请在手机端或微信小程序下载并安装 Google Authenticator 应用



Android手机下载



iPhone手机下载

范例：

- 在手机应用市场搜索：身份验证器或authenticator，并安装APP或者也可以使用微信小程序MinaOP
- 安装google-authenticator，需要有epel源
- 执行google-authenticator
- 在/etc/pam.d/sshd中添加auth required pam_google_authenticator.so
- 修改sshd配置文件/etc/ssh/sshd_config中ChallengeResponseAuthentication yes
- 重启ssh服务

#查看

```
[19:36:30 root@centos8 ~]#yum info google-authenticator.x86_64
```

#安装，需先安装epel源

```
[19:50:30 root@centos8 ~]#yum install -y epel-release
```

```
[19:44:00 root@centos8 ~]#yum install -y google-authenticator.x86_64
```

#执行，所有选项都执行y

```
[19:51:36 root@centos8 ~]#google-authenticator
```

Do you want authentication tokens to be time-based (y/n) y

Warning: pasting the following URL into your browser exposes the OTP secret to Google:

#浏览器打开下面的网站用安装的app扫二维码添加主机，需要科学上网

```
https://www.google.com/chart?chs=200x200&chld=Mj0&cht=qr&chl=otpauth://totp/root  
centos8%3Fsecret%3D4DME7Q3DLMD75TQY722B2KB4ME%26issuer%3Dcentos8
```

Failed to use libqrencode to show QR code visually for scanning.

Consider typing the OTP secret into your app manually.

Your new secret key is: 4DME7Q3DLMD75TQY722B2KB4ME

Enter code from app (-1 to skip): 509005 #

Code confirmed

Your emergency scratch codes are:

#下面生成的数字需要保存，以防止手机丢失无法登录时使用，那个使用过后就消失了

需要添加的话可以在/root/.google_authenticator文件中添加

21350850

32992678

17672106

41326673

97208182

Do you want me to update your "/root/.google_authenticator" file? (y/n) y

Do you want to disallow multiple uses of the same authentication token? This restricts you to one login about every 30s, but it increases your chances to notice or even prevent man-in-the-middle attacks (y/n) y

By default, a new token is generated every 30 seconds by the mobile app.

In order to compensate for possible time-skew between the client and the server, we allow an extra token before and after the current time. This allows for a time skew of up to 30 seconds between authentication server and client. If you experience problems with poor time synchronization, you can increase the window from its default size of 3 permitted codes (one previous code, the current code, the next code) to 17 permitted codes (the 8 previous codes, the current code, and the 8 next codes). This will permit for a time skew of up to 4 minutes between client and server.

Do you want to do so? (y/n) y

If the computer that you are logging into isn't hardened against brute-force login attempts, you can enable rate-limiting for the authentication module.

By default, this limits attackers to no more than 3 login attempts every 30s.

Do you want to enable rate-limiting? (y/n) y

#在pam配置文件sshd文件中添加模块

```
[19:58:37 root@centos8 ~]#vim /etc/pam.d/sshd
```

```
auth    required    pam_google_authenticator.so
```

```
#修改ssh服务配置文件  
[20:01:45 root@centos8 ~]#vim /etc/ssh/sshd_config  
ChallengeResponseAuthentication yes
```

```
#重启ssh服务  
[20:02:44 root@centos8 ~]#systemctl restart sshd
```

使用ssh登录时需要在红色标记里面使用app动态密钥验证后才能使用密码登录

```
[20:04:28 root@centos7 ~]#ssh 192.168.10.81  
Verification code:  
Password:  
Activate the web console with: systemctl enable --now cockpit.socket  
  
Last login: Fri Jan 15 20:04:21 2021  
[20:05:07 root@centos8 ~]#
```