1. **Linux系统安全概述-sudo授权-pam认证机制-对称加密-非对称加密-md5-数字证书**

**本节所讲内容：**

**17.1 Linux系统安全概述-su-sudo授权**

**17.2 pam认证机制概述**

**17.3 对称加密-非对称加密方法**

**17.4 md5-sha1-哈希算法使用方法**

**17.5 CA证书的作用**

**本节所使用实验环境：RHEL 7 （如有使用RHEL 6，会特别注明）**

**17.1 Linux系统安全概述-sudo授权**

**17.1.1 Linux第三阶段资深知识整体概述：**

**/安全、稳定**

**/可用性－网络(协议)**

**/ \易用、易维护**

**/ \系统调优 （高性能、低成本）**

**应用/服务**

**\**

**\ /要想性能提升：硬件升级**

**\扩展性 －集群**

**\存储**

**\云/虚拟化**

**常见攻防手段：**

**1. 防防范手段整体分两部分**

**/技术**

**防范手段**

**\法规（规章制度）社会工程学**

**技术层面：**

**/OS漏洞**

**/1.系统安全－密码验证**

**｜ \权限**

**｜ /程序的配置文件（如ftp的黑白名单 samba的...）**

**安全意识< 2.服务安全－防火墙iptables**

**｜ \TCP wrapper**

**｜ \程序漏洞 例：nginx 0day 漏洞**

**｜**

**｜ /攻击、防范方法**

**\3.网络安全－用户验证**

**\伪装截获**

**1.系统安全**

**系统验证**

**/文本界面：login 比例：init 3**

**｜**

**1.本地登录**

**｜ /gdm桌面管理器**

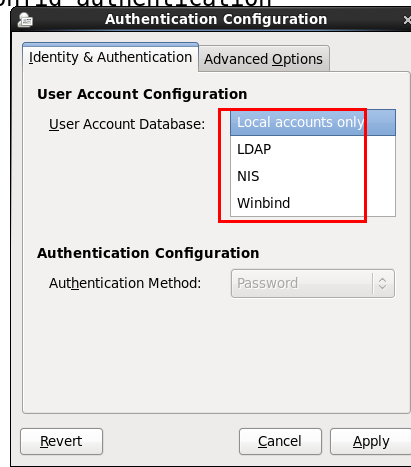
**\图形－kdm桌面管理器**

**\xdm桌面管理器**

**17.1.2 认证方法：**

**[root@xuegod63 Desktop]# system-config-authentication //如果没有此命令，需要安装**

**[root@xuegod63 ~]# yum -y install authconfig-gtk //安装**



**字符界面：[root@xuegod63 Desktop]# authconfig-tui**

**存放用户信息和密码信息：**

**[root@xuegod63 ~]# ls /etc/passwd\***

**/etc/passwd /etc/passwd-**

**[root@xuegod63 ~]# ls /etc/shadow\***

**/etc/shadow /etc/shadow-**

**密码：/etc/shadow- 是/etc/shadow 文件备份**

**查看是否一样：**

**[root@xuegod63 ~]# diff /etc/shadow /etc/shadow-**

**[root@xuegod63 ~]# diff /etc/passwd /etc/passwd-**

**或**

**[root@xuegod63 ~]# vimdiff /etc/passwd /etc/passwd-**

**密码加密技术：**

**RHEL5 使用 MD5 对/etc/shadow中密码加密**

**RHEL6 使用sha512 对/etc/shadow中密码加密**

**例：**

**[root@xuegod63 ~]# grep shadow /root/anaconda-ks.cfg**

**authconfig --enableshadow --passalgo=sha512**

**17.1.3 授权su/sudo权限**

**/su - root**

**su－须知道用户root的密码**

**\拥有root用户权限**

**/不需要知道root密码**

**sudo－仅拥有授权的用户权限**

**\配置简单**

**例：su 切换用户：**

**[root@xuegod63 ~]# su - root**

**17.1.4 通过sudo 给普通用户授权：**

**sudo 是在不切换用户的情况下，以其他用户的身份执行一个命令。它能够限制指定用户在指定的主机上运行某些指定的命令。**

**/etc/sudoers是sudo命令的配置文件**

**/etc/sudoers 文档格式：**

**[root@xuegod63 ~]# vim /etc/sudoers**

**命令格式**

**用户名或组名 ALL=(以谁的身份运行命令) Commd\_Alias运行的命令**

**注：**

**如果组：前面加% 。 如： %kill**

**Commd\_Alias运行的命令：注：1命令；2目录（必须以/结尾）最好使用绝对路径；**

**配置sudo方法：**

**方法1：visudo**

**visudo命令概述：**

**sudo都提供了一个编辑该文件/etc/sudoers的命令：visudo来对该文件/etc/sudoers进行修改。使用visudo命令修改 sudoers配置，因为它会帮你校验文件配置是否正确，如果不正确，在保存退出时就会提示你哪段配置出错的。**

**方法2：vim直接编辑/etc/sudoers**

**[root@xuegod63 ~]# vim /etc/sudoers**

**实例：测试 sudo 命令**

**1、建立两个普通用户 liv ，  poppy 添加到组stu**

**[root@xuegod63 ~]# groupadd stu**

**[root@xuegod63 ~]# useradd -g stu liv**

**[root@xuegod63 ~]# useradd -g stu poppy**

**[root@xuegod63 ~]# echo redhat | passwd --stdin liv**

**[root@xuegod63 ~]# echo redhat | passwd --stdin poppy**

**切换到liv 测试fdisk -l**

**[root@xuegod63 ~]# su - liv**

**[liv@xuegod63 ~]$ fdisk -l /提示对以下设备权限不够，无法打开**

**fdisk: 打不开 /dev/sda: 权限不够**

**fdisk: 打不开 /dev/sr0: 权限不够**

**2 让liv具有root用户使用fdisk的权限。**

**[root@xuegod63 ~]# vim /etc/sudoers #在文件最后添加以下内容：**

**liv ALL=(root) /sbin/fdisk**

**（注：使用sudo命令时加上命令的全路径）**

**保存退出。不需要重启什么服务。**

**测试：**

**[liv@xuegod63 ~]$ sudo fdisk -l**

**We trust you have received the usual lecture from the local System**

**Administrator. It usually boils down to these three things:**

**#1) Respect the privacy of others.**

**#2) Think before you type.**

**#3) With great power comes great responsibility.**

**[sudo] password for liv: 123456**

**Sorry, try again.**

**[sudo] password for liv: redhat #需要输入liv的密码**

**Disk /dev/sda: 21.5 GB, 21474836480 bytes**

**255 heads, 63 sectors/track, 2610 cylinders**

**Units = cylinders of 16065 \* 512 = 8225280 bytes**

**Sector size (logical/physical): 512 bytes / 512 bytes**

**I/O size (minimum/optimal): 512 bytes / 512 bytes**

**Disk identifier: 0x000b8b35**

**Device Boot Start End Blocks Id System**

**/dev/sda1 \* 1 26 204800 83 Linux**

**Partition 1 does not end on cylinder boundary.**

**/dev/sda2 26 1301 10240000 83 Linux**

**/dev/sda3 1301 1428 1024000 82 Linux swap / Solaris**

**[liv@xuegod63 ~]$**

**例：再添加几个命令行 使liv能执行更多的命令**

**[root@xuegod63 ~]# visudo #添加以下内容**

**liv ALL=(root) /sbin/fdisk,NOPASSWD:/usr/bin/systemctl,/usr/sbin/lsof,/usr/sbin/service,/usr/sbin/ifconfig,/usr/bin/mount ## RHEL 7环境**

**liv ALL=(root) /sbin/fdisk,NOPASSWD:/usr/sbin/service,/usr/sbin/ifconfig,/usr/bin/mount ##RHEL 6 环境**

**注：添加上NOPASSWD: 后，NOPASSWD:之：后的命令都不需要输入密码。默认是PASSWD**

**测试：**

**[liv@xuegod63 ~]$ sudo fdisk -l #需要输入liv密码**

**RHEL 6 环境下：**

**[liv@xuegod63 ~]$ service httpd restart # 执行不了，没有权限。**

**rm: cannot remove `/var/run/httpd/httpd.pid': Permission deniedLED]**

**rm: cannot remove `/var/run/httpd/httpd.pid': Permission denied**

**Starting httpd: (13)Permission denied: make\_sock: could not bind to address [::]:80**

**(13)Permission denied: make\_sock: could not bind to address 0.0.0.0:80**

**no listening sockets available, shutting down**

**Unable to open logs**

**[FAILED]**

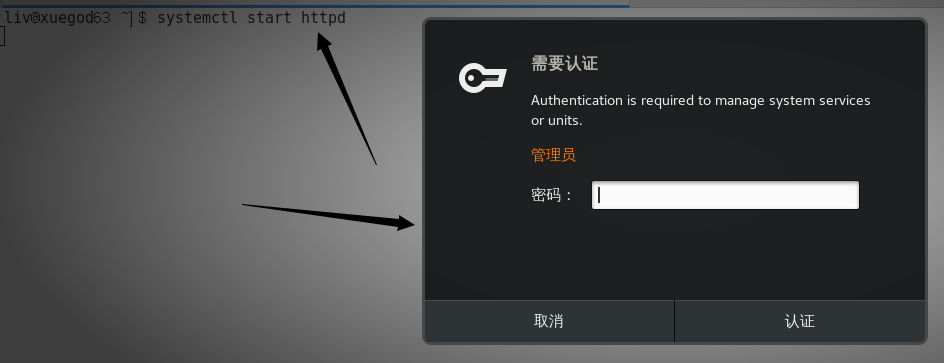
**[liv@xuegod63 ~]$ sudo service httpd restart #NOPASSWD:之后的命令都不需要输入密码**

**Stopping httpd: [FAILED]**

**Starting httpd: [ OK ]**

**RHEL 7环境下：**

**[liv@xuegod63 ~]$ systemctl start httpd //需要认证，需要输入root密码才能够启动**



**[liv@xuegod63 ~]$ sudo systemctl start httpd /使用sudo是可以直接启动了，也不需要输入root用户密码**

**[liv@xuegod63 ~]$ sudo lsof -i :80**

**COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME**

**httpd 43356 root 4u IPv6 101979 0t0 TCP \*:http (LISTEN)**

**httpd 43357 apache 4u IPv6 101979 0t0 TCP \*:http (LISTEN)**

**httpd 43358 apache 4u IPv6 101979 0t0 TCP \*:http (LISTEN)**

**httpd 43359 apache 4u IPv6 101979 0t0 TCP \*:http (LISTEN)**

**httpd 43360 apache 4u IPv6 101979 0t0 TCP \*:http (LISTEN)**

**httpd 43361 apache 4u IPv6 101979 0t0 TCP \*:http (LISTEN)**

**[liv@xuegod63 ~]$mount /dev/cdrom /opt/ #执行不成功**

**[liv@xuegod63 ~]$sudo mount /dev/cdrom /opt/ #加sudo 就可以执行成功**

**[liv@xuegod63 ~]$ ls /opt/**

**[liv@xuegod63 ~]$sudo fdisk -l**

**总结：**

**/etc/sudoers中常用的写法如下：**

**例：一个用户可以执行另一个用户的某些命令**

**[root@xuegod63 ~]# vim /etc/sudoers**

**liv ALL=(poppy) /opt/poppy.sh**

**例2：让stu组中的所有用户，可以执行/sbin目录下的所有命令**

**%stu ALL=(root) /sbin/**

**例3： liv用户可以执行任何用户的任何命令。**

**[root@xuegod63 ~]# vim /etc/sudoers**

**liv ALL=(ALL) ALL**

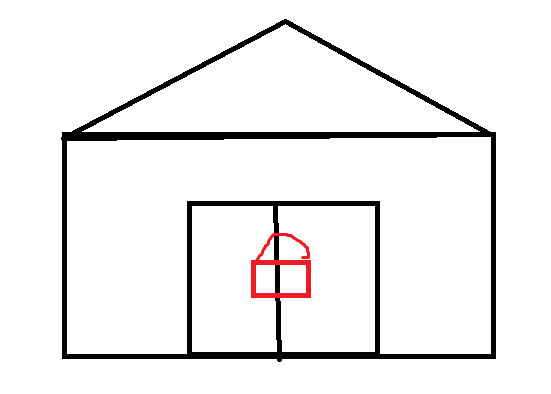
**17.2 PAM可插入式验证模块.**

**Pluggable（可插拔） Authentication（认证） Modules ）。sun公司提出出来的一种机制。它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的服务配置不同的认证方式而无需更改服务程序，同时也便于向系统中添加新的认证手段。**

**PAM模块和应用程序之间的关系：就你锁和门的关系**

**锁==PAM认证模块**

**门==应用程序**

****

**有验证的程序(如login、ssh、vsftpd、samba、apache)：**

**1）使用通用的pam验证模块.**

**2）程序自写验证功能. 例：apache的验证是自己开发的**

**如何知道一个程序是否支持pam认证**

**ldd命令 显示进程运行所依赖的动态连接库的详细信息。**

**例：**

**[root@xuegod63 Desktop]# ldd /bin/login #/bin/login 命令的相关共享库文件列表**

**linux-vdso.so.1 => (0x00007ffccfad0000)**

**libpam.so.0 => /lib64/libpam.so.0 (0x00007f1786dd9000)**

**libpam\_misc.so.0 => /lib64/libpam\_misc.so.0 (0x00007f1786bd5000)**

**libaudit.so.1 => /lib64/libaudit.so.1 (0x00007f17869ac000)**

**libselinux.so.1 => /lib64/libselinux.so.1 (0x00007f1786785000)**

**libc.so.6 => /lib64/libc.so.6 (0x00007f17863c2000)**

**libdl.so.2 => /lib64/libdl.so.2 (0x00007f17861bd000)**

**libcap-ng.so.0 => /lib64/libcap-ng.so.0 (0x00007f1785fb7000)**

**libpcre.so.1 => /lib64/libpcre.so.1 (0x00007f1785d55000)**

**/lib64/ld-linux-x86-64.so.2 (0x0000562d55f13000)**

**libpthread.so.0 => /lib64/libpthread.so.0 (0x00007f1785b38000)**

**例：查看login是否支持pam**

**[root@xuegod63 Desktop]# ldd /bin/login |grep pam**

**libpam.so.0 => /lib64/libpam.so.0 (0x00007fe1dc652000)**

**libpam\_misc.so.0 => /lib64/libpam\_misc.so.0 (0x00007fe1dc44e000)**

**可以查看到pam相关的库，说它支持pam认证。**

**pam模块：每个模块的功能的都是专用的，独特的。**

**17.2.1 pam相关配置文件：**

**1、pam模块位置：**

**64位 操作系统：[root@xuegod63 Desktop]# ls /lib64/security/**

**32位 操作系统：[root@xuegod63 Desktop]# ls /lib/security/**

**2、每个pam模块的配置文件：ls /etc/security/**

**例：查找模块pam\_access.so位置及对应的模块配置文件**

**[root@xuegod63 Desktop]# ls /lib64/security/pam\_access.so**

**/lib64/security/pam\_access.so**

**[root@xuegod63 ~]# ls /etc/security/access.conf**

**/etc/security/access.conf**

**3、每个服务加载的pam模块的配置文件位置**

**ls /etc/pam.d**

**例：[root@iZ25b6alxstZ ~]# vim /etc/pam.d/sshd #每个服务器的pam配置文件一般以服务名开头**

**[root@iZ25b6alxstZ ~]# vim /etc/pam.d/sshd**

**#%PAM-1.0**

**auth required pam\_sepermit.so**

**auth include password-auth**

**account required pam\_nologin.so**

**account include password-auth**

**注释：配置文件格式**

**验证类型 控制标识 模块**

**auth include包含 文件**

**account required需要 root 模块 [参数]**

**requisite（必要） 模块**

**optional可选 模块**

**sufficient（充足）充分 模块**

**验证类型：**

**auth 与用户名密码相关的 ，如： 用户密码是否正确 ，用户是否锁定，密码是否锁定。**

**account 与用户名密码无关的，如：对用户登录的时间做检查，是否过期。是否允许在ttyx上登录。**

**控制标识：**

**需要required：表示后面跟的模块的功能是必要条件，必须成功，最后才可能验证成功、**

**如果不成功，继续往后验证，最后才返回失败。**

**例：验证用户名和密码时，如果用户名出错，然后继续往后验证，最后才返回失败。 好处： 不让黑客知道自己是用户名错了还是密码。**

**必要requisite：表示后面跟的模块的功能是必要条件，必须成功，最后才可能验证成功、**

**如果不成功，马上返回失败，而不继续往后验证。**

**例：比如判断一个用户是否是root用户，如果不是，直接返回权限不够。**

**充分sufficient：充分条件，表示如果后面跟的模块的功能成功，则马上返回成功**

**如果不成功，忽略不计，继续往后验证。**

**例：权限验证身份是否是root， 如果是root用户，直接执行**

**可选optional：表示后面跟的模块的功能是可选条件，不影响最后验证结果**

**包含include：包含后面指定的文件中的相同验证类型的行。**

**例1：限制普通用户 Kill 不能在tty2上登录。 在字符login程序下运行。 gnome图面不生效。在gnome图形界面下，是另一套认证机制。**

**生成用户：**

**[root@xuegod63 ~]# useradd kill**

**[root@xuegod63 ~]# echo 123456 | passwd --stdin kill**

**Changing password for user kill.**

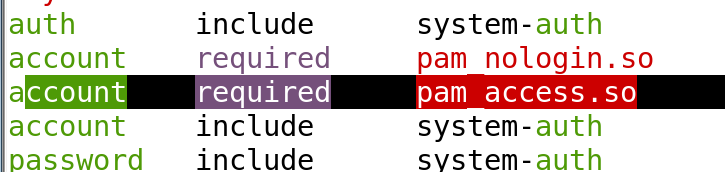
**passwd: all authentication tokens updated successfully.**

**第一步： 让字符界面登录认证程序login加访问控制功能"pam\_access模块”。**

**[root@xuegod63 Desktop]# vim /etc/pam.d/login #添加以下红色内容**

**account required pam\_access.so**

**如图：**



**第二步：修改pam\_access模块的配置文件/etc/security/access.conf：**

**[root@xuegod63 ~]# ls /lib64/security/pam\_access.so #pam\_access模块**

**/lib64/security/pam\_access.so**

**[root@xuegod63 ~]# ls /etc/security/access.conf # pam\_access模块配置文件**

**/etc/security/access.conf**

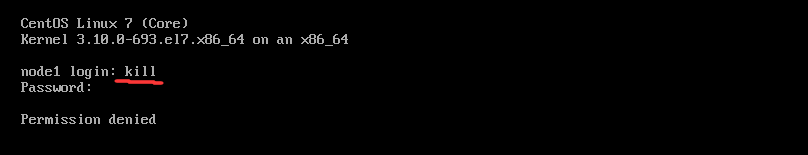
**[root@xuegod63 ~]# vim /etc/security/access.conf #一定要顶头写，不要有空格。在最后添加**

**- : kill : tty2**

**#禁止kill从tty2上登录。**

**测试：**

**ctrl+alt+F2 切换到： tty2上测试，**



**查看日志：**

**[root@xuegod63 ~]#tail -f /var/log/secure #与验证有关的都是此日志里一定要学会看帮助和日志**

**May 7 20:14:11 localhost login: pam\_access(login:account): access denied for user `kaixin' from `tty2'**

**May 7 20:14:11 localhost login: Permission denied**

**总结使用pam模块流程**

**1、配置服务的pam配置文件来调用pam模块A （选一个锁）**

**2、配置pam模块A的配置文件，实现某个功能 （上锁）**

**3、测试**

**实例2： su 为什么root切换普通用户不用输入密码,普通用户切换root却要输入密码？？？**

**17.3 对称加密-非对称加密方法**

**17.3.1加密算法：**

**1）对称加密(一把密钥配) 容易泄露，效率高**

**2）非对称加密(一对密钥配(公钥用来加密，私钥用来解密)) 更安全，效率低**

**对称加密算法：AES 高级加密算法**

**DES 标准加密算法**

**3DES 三重数据加密算法（TDEA，Triple Data Encryption Algorithm）块密码的通称。它相当于是对每个数据块应用三次DES加密算法**

**非对称加密算法：DSA**

**RSA**

**例1： 演示对称加密：3des算法**

**[root@xuegod63 Desktop]# yum install openssl**

**openssl 是一套加密工具包。**

**[root@xuegod64 ~]# openssl --help #使用方法**

**[root@xuegod64 ~]#openssl enc --help #查看加密算法**

**例：加密**

**[root@xuegod63 Desktop]# openssl enc -e -des3 -a -in /etc/passwd -out /passwd.des3**

**enter des-ede3-cbc encryption password:123456**

**Verifying - enter des-ede3-cbc encryption password:123456**

**参数：**

**选项： options are**

**-in <file> input file**

**-out <file> output file**

**-e encrypt 加密 ； 后面可以加很多加密算法，如： -des3 加密码算法**

**-d decrypt 解码**

**-a/-base64 base64 encode/decode, depending on encryption flag。# base64编码/解码，这取决于加密标志。**

**注：Base64是网络上最常见的用于传输8Bit字节代码的编码方式之一。**

**查看加密效果。**

**[root@xuegod63 Desktop]# vim /passwd.des3**

**查看不使用-a base64编译的效果：**

**[root@xuegod63 ~]# openssl enc -e -des3 -in /etc/passwd -out /passwd.des3-v2**

**enter des-ede3-cbc encryption password:**

**Verifying - enter des-ede3-cbc encryption password:**

**[root@xuegod63 ~]# vim /passwd.des3-v2**

**这次对称加密的公钥是什么？**

**解密：**

**[root@xuegod63 Desktop]# openssl enc -d -des3 -a -in /passwd.des3 -out /root/passwd**

**-d decrypt 解码**

**查看解密完的文件和原文件是否一致**

**[root@xuegod63 ~]# diff /etc/passwd /root/passwd #没有任何输出说明，两个文件内容一样**

**实例2：使用RSA算法进行非对称加密演示过程**

**安装：加密工具**

**[root@xuegod63 ~]# rpm -qf `which gpg`**

**gnupg2-2.0.14-4.el6.x86\_64**

**xuegod63主机上**

**说明： gpg命令要在图形界面执行。因为执行过程中会弹出窗口。**

**[root@xuegod63 Desktop]# gpg --gen-key #生成密钥对**

**#产生一个新的密钥对 --gen-key generate（ˈjenəˌrāt产生） a new key pair**

**gpg (GnuPG) 2.0.14; Copyright (C) 2009 Free Software Foundation, Inc.**

**This is free software: you are free to change and redistribute it.**

**There is NO WARRANTY, to the extent permitted by law.**

**Please select what kind of key you want:**

**(1) RSA and RSA (default)**

**(2) DSA and Elgamal**

**(3) DSA (sign only)**

**(4) RSA (sign only)**

**Your selection? #直接回车，选择 RSA算法**

**RSA keys may be between 1024 and 4096 bits long.**

**What keysize do you want? (2048) #回车， 密钥长度： 2048**

**Requested keysize is 2048 bits**

**Please specify how long the key should be valid. #请指定密钥多长时间有效**

**0 = key does not expire #0表示，永不过期**

**<n> = key expires in n days**

**<n>w = key expires in n weeks**

**<n>m = key expires in n months**

**<n>y = key expires in n years**

**Key is valid for? (0) #回车**

**Key does not expire at all**

**Is this correct（正确）? (y/N) y**

**GnuPG needs to construct a user ID to identify your key.**

**Real name: xuegod #密钥的名字**

**Email address: 1@163.com**

**Comment: xuegodrsa**

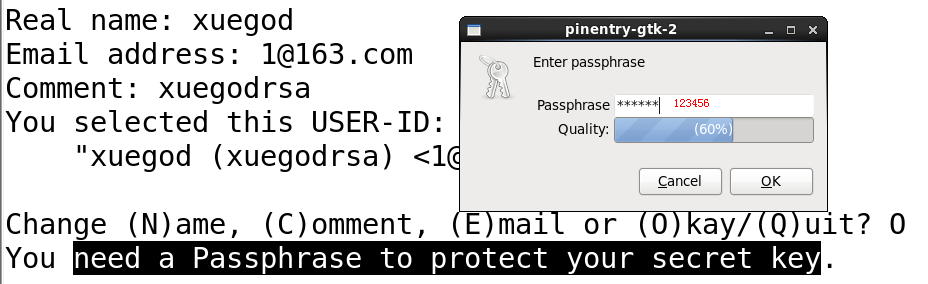
**You selected this USER-ID:**

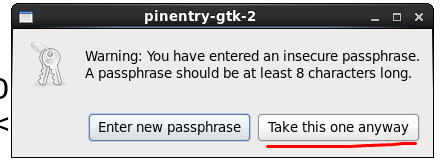
**"xuegod (xuegodrsa) <1@163.com>"**

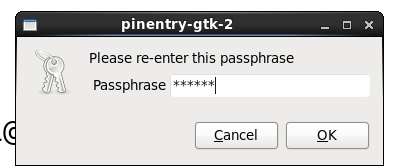
**Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O**

**You need a Passphrase to protect your secret key.**

**#输入密码保护密钥**







**can't connect to `/root/.gnupg/S.gpg-agent': No such file or directory**

**gpg-agent[6161]: directory `/root/.gnupg/private-keys-v1.d' created**

**We need to generate a lot of random bytes. (我们需要生成大量的随机字节。可以再打开一个终端执行： find / 命令产生所机值). It is a good idea to perform**

**some other action (type on the keyboard, move the mouse, utilize the**

**disks) during the prime generation; this gives the random number**

**generator a better chance to gain enough entropy.**

**We need to generate a lot of random bytes. It is a good idea to perform**

**some other action (type on the keyboard, move the mouse, utilize the**

**disks) during the prime generation; this gives the random number**

**generator a better chance to gain enough entropy.**

**gpg: key 01B94CBE marked as ultimately trusted**

**public and secret key created and signed.**

**gpg: checking the trustdb**

**gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model**

**gpg: depth: 0 valid: 1 signed: 0 trust: 0-, 0q, 0n, 0m, 0f, 1u**

**pub 2048R/01B94CBE 2014-12-20**

**Key fingerprint = 91DD 87D1 DF7D F82E F6EC 9165 E961 6018 01B9 4CBE**

**uid xuegod (xuegodrsa) <1@163.com>**

**sub 2048R/5B3C7567 2014-12-20**

**产生成随机事件：**

**[root@xuegod64 ~]# find /**

**17.3.2 查看密钥对**

**[root@xuegod63 Desktop]# file /root/.gnupg/pubring.gpg**

**/root/.gnupg/pubring.gpg: GPG key public ring**

**[root@xuegod63 Desktop]# file /root/.gnupg/secring.gpg**

**实战：实现xuegod63和xuegod64 之间使用非对称加密方法传输数据**

**#复制公钥到另一台机器xuegod64上**

**[root@xuegod63 ~]# scp /root/.gnupg/pubring.gpg 192.168.1.64:/root/**

**root@192.168.1.64's password:**

**pubring.gpg 100% 1189 1.2KB/s 00:00**

**测试加密解密：**

**客户端： xuegod64：**

**[root@xuegod64 ~]# which gpg**

**/usr/bin/gpg**

**[root@xuegod64 ~]# rpm -qf /usr/bin/gpg**

**gnupg2-2.0.14-4.el6.x86\_64**

**查看公钥：**

**[root@xuegod64 ~]# gpg --list-keys**

**17.3.3 导入公钥：**

**[root@xuegod64 ~]# gpg --import /root/pubring.gpg**

**gpg: key 01B94CBE: public key "xuegod (xuegodrsa) <1@163.com>" imported**

**gpg: Total number processed: 1**

**gpg: imported: 1 (RSA: 1)**

**[root@xuegod64 ~]# gpg --list-keys**

**/root/.gnupg/pubring.gpg**

**------------------------**

**pub 2048R/01B94CBE 2014-12-20**

**uid xuegod (xuegodrsa) <1@163.com>**

**sub 2048R/5B3C7567 2014-12-20**

**17.3.4 使用公钥加密：**

**[root@xuegod64 ~]# cp /etc/passwd /root**

**[root@xuegod64 ~]# gpg -ear xuegod /root/passwd**

**参数：**

**-a, --armor 装甲 ['ɑ:mə] create ascii armored output 以ASCII的方式输出**

**-r, --recipient接受者[rɪˈsɪpiənt] USER-ID encrypt for USER-ID指定加密使用的公钥的名字。**

**-e, --encrypt [ɪnˈkrɪpt] encrypt data**

**-d, --decrypt [di:ˈkrɪpt] decrypt data (default)**

**gpg: 5B3C7567: There is no assurance this key belongs to the named user**

**pub 2048R/5B3C7567 2014-12-20 xuegod (xuegodrsa) <1@163.com>**

**Primary key fingerprint: 91DD 87D1 DF7D F82E F6EC 9165 E961 6018 01B9 4CBE**

**Subkey fingerprint: 5122 62B6 72A9 F771 F33A 1834 6406 913C 5B3C 7567**

**It is NOT certain that the key belongs to the person named**

**in the user ID. If you \*really\* know what you are doing,**

**you may answer the next question with yes.**

**Use this key anyway? (y/N) y**

**参数：**

**-ear ：**

**-e, --encrypt加密 encrypt data**

**-a, --armor （盔甲） create ascii armored output #以ascII编码输出**

**-r, --recipient接收USER-ID encrypt for USER-ID 指定加密使用的公钥的名字。**

**-e, --encrypt（**[ɪnˈkrɪpt]**加密） encrypt data**

**-a, --armor （**['ɑ:mə] **盔甲） create ascii armored output #以ascII编码输出**

**-r, --recipient（**[rɪˈsɪpiənt] **接收） USER-ID encrypt for USER-ID 指定加密使用的公司UID**

**[root@xuegod64 ~]# ls passwd\***

**passwd passwd.asc （加密后生成的文件）**

**[root@xuegod64 ~]# vim passwd**

**[root@xuegod64 ~]# vim passwd.asc #查看加密后的数据**

**解密：xuegod63**

**[root@xuegod64 ~]# scp passwd.asc 192.168.1.63:/root/**

**在xuegod63解密码：**

**[root@xuegod63 ~]# gpg --list-key**

**/root/.gnupg/pubring.gpg**

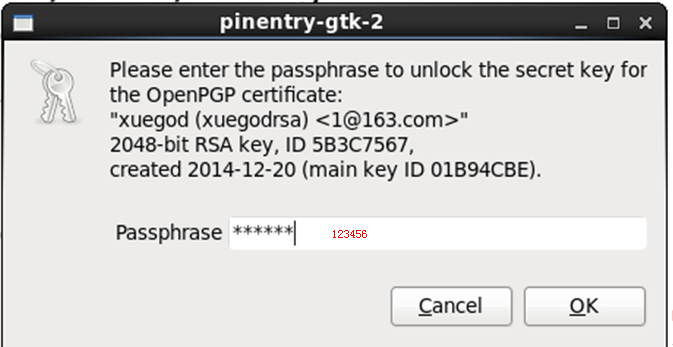
**------------------------**

**pub 2048R/EF0DC402 2016-03-18**

**uid xuegod (xuegodrsa) <1@163.com>**

**sub 2048R/63E9B374 2016-03-18**

**[root@xuegod63 ~]# gpg -dar xuegod -o passwd passwd.asc**

**密码为：私钥的密码**

**参数：**

**-d 解密**

**-a, --armor （盔甲） create ascii armored output #以ascII编码输出**

**-r, --recipient USER-ID encrypt for USER-ID 指定加密使用的公司UID**

**-o 指定解密后的内容输出到文件中。**

**例：公钥私钥实例，实现xuegod63登录xuegod64时，不输入密码**

**[root@ xuegod63~]# ssh-keygen**

**Generating public/private rsa key pair.**

**Enter file in which to save the key (/root/.ssh/id\_rsa):**

**#提示输入密匙文件的保存路径，选择默认继续**

**Enter passphrase (empty for no passphrase):**

**#提示输入密匙文件的保存路径，选择默认继续**

**下面要求输入密码，这里的passphrase 密码是对生成的私匙文件(/root/.ssh/id\_dsa)**

**的保护口令，如果不设置可以回车。**

**Enter same passphrase again:**

**Your identification has been saved in /root/.ssh/id\_rsa.**

**Your public key has been saved in /root/.ssh/id\_rsa.pub.**

**The key fingerprint is:**

**f5:e1:15:68:8e:52:38:64:6e:56:08:e3:0e:c3:09:83 root@localhost**

**The key's randomart image is:**

**+--[ RSA 2048]----+**

**| .o oooo. .. |**

**|E + o ++.. o . |**

**| = . +o.+. . |**

**| + o...o.o |**

**| . S. o |**

**| |**

**| |**

**| |**

**| |**

**+-----------------+**

**复制公钥到xuegod64上：**

**[root@xuegod63 ~]# ssh-copy-id -i /root/.ssh/id\_rsa.pub 192.168.1.64**

**或：**

**[root@xuegod63 ~]# ssh-copy-id root@192.168.1.64**

**root@192.168.1.64's password:**

**Now try logging into the machine, with "ssh '192.168.1.63'", and check in:**

**.ssh/authorized\_keys**

**测试：**

**[root@xuegod64 ~]# ssh 192.168.1.64**

**Last login: Tue Dec 18 22:22:39 2012**

**[root@xuegod63 ~]#**

**17.4 md5-sha1-哈希算法使用方法**

**17.4.1 使用哈希算法保证数据完整性。**

**常见的哈希算法有：MD5 和 SHA1**

**MD5 ，SHA1 的区别：**

**MD5的全称是Message-Digest Algorithm 5（信息-摘要算法）。MD5 算法的哈希值大小为 128 位。是一种不可逆的算法。**

**SHA1的全称是Secure Hash Algorithm(安全哈希算法)。SHA1 算法的哈希值大小为 160 位。是一种不可逆的算法。**

**SHA1抗暴力破解能力比MD5强。**

**缺点：产生SHA-1 报文摘要的运行速度比MD5 慢。**

**哈希算法具有以下特点：**

**1、压缩性：任意长度的数据，算出的哈希值长度都是固定的。**

**2、容易计算：从原数据计算出哈希值很容易。**

**3、抗修改性：对原数据进行任何改动，哪怕只修改1个字节，所得到的哈希值都有很大区别。**

**4、弱抗碰撞：已知原数据和其哈希值，想找到一个具有相同哈希值的数据（即伪造数据）是非常困难的。**

**实例1：产成固定长度MD5消息摘要**

**[root@xuegod63 ~]# md5sum /etc/passwd**

**d5712ff41f92511823453e10e433b948 /etc/passwd**

**[root@xuegod63 ~]# sha1sum /etc/passwd**

**e3cd228d75a0cee98c82848d4f1d82fa4b4b7710 /etc/passwd**

**实例2：使用MD5加密密码：**

**[root@xuegod63 ~]# echo 123 | md5sum**

**ba1f2511fc30423bdbb183fe33f3dd0f -**

**[root@xuegod63 ~]# echo 1234 | md5sum**

**e7df7cd2ca07f4f1ab415d457a6e1c13 -**

**实例3：演示MD5保障数据的完整性：**

**使用MD5检查先前使用rsa私钥解密后的文件和原文件是否一致**

**[root@xuegod63 ~]# scp /root/passwd 192.168.1.64:/opt/ #把解密的文件复制到xuegod64**

**[root@xuegod64 ~]# md5sum /etc/passwd /opt/passwd #在xuegod64上进行对比，如果哈希值一样，说明正确。**

**d5712ff41f92511823453e10e433b948 /etc/passwd**

**d5712ff41f92511823453e10e433b948 /opt/passwd**

**[root@xuegod63 ~]# md5sum /dev/sda1 #可以分区取一个哈希值。**

**扩展：**

**弱抗碰撞：已知原数据和其MD5值，想找到一个具有相同MD5值的数据（即伪造数据）是非常困难的。**

**百度网盘中，如何实现秒传？比如rhel7.2.ios 3G的文件1秒就可以上传成功。百度网盘中存一个MD5值表， 判断两个文件的MD5是不是一样，如果一样，直接创建软链接。**

**百度网盘上是不允许上传cang.avi的。**

**mv cang.avi cjk.avi ? 改一下名字，就可以上传了吗？ 答案： 一样不行的**

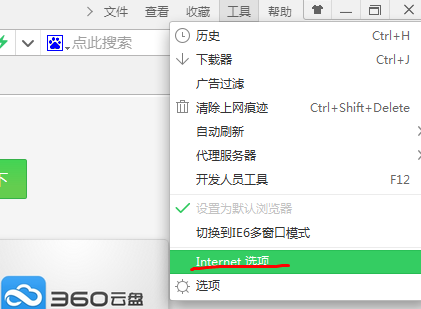
**身份认证---》 数据的不可否认性**

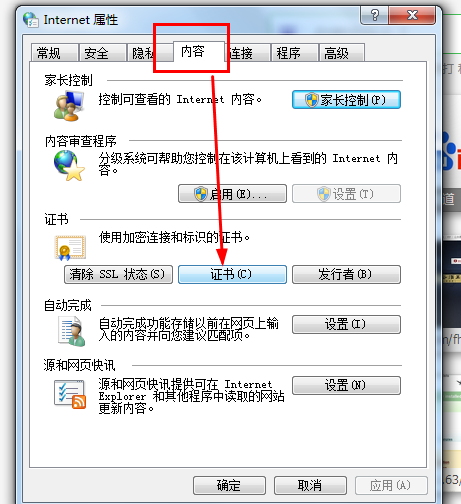
**例：**

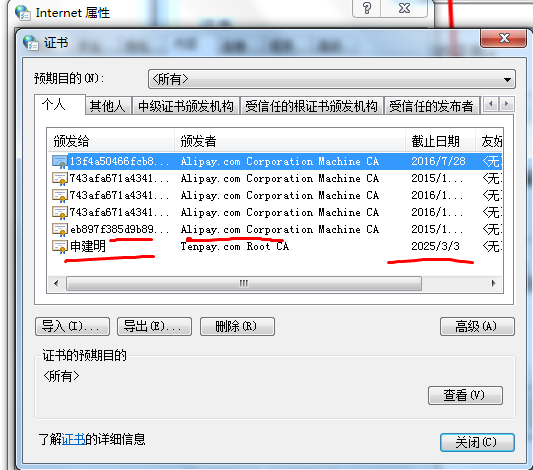
**http://www.12306.cn/mormhweb/**

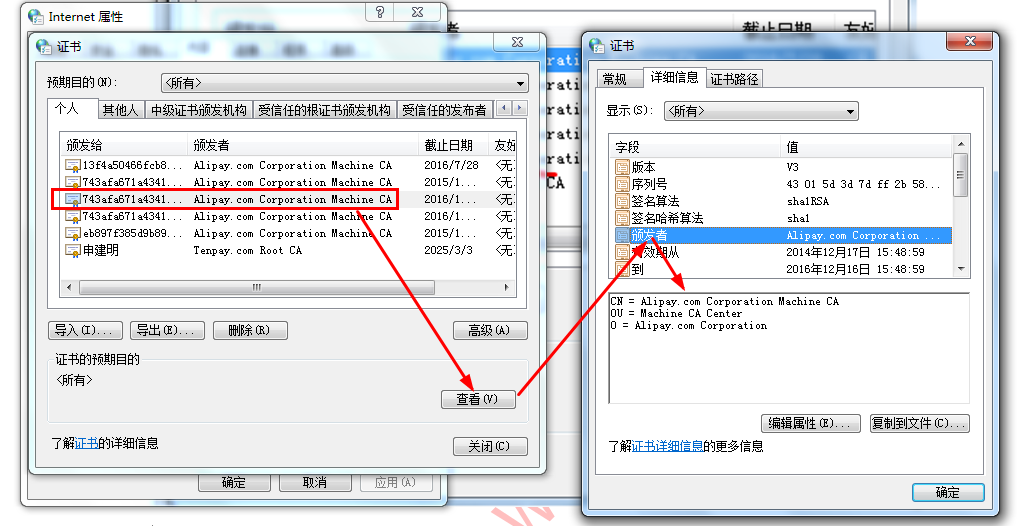


**查看浏览器中的证书：**









**CN = Alipay.com Corporation Machine CA**

**OU = Machine CA Center**

**O = Alipay.com Corporation**

**总结：**

**17.1 Linux系统安全概述-su-sudo授权**

**17.2 pam认证机制概述**

**17.3 对称加密-非对称加密方法**

**17.4 md5-sha1-哈希算法使用方法**

**17.5 CA证书的作用**