一、Linux 安全基线排查

1. Linux账号管理与认证授权

1. 用户帐号口令设置

-
- awk -F: '($2==""){print$1}' /etc/shadow
  - 检查是否存在空口令账号
  - 有输出则存在，无输出则不存在

```
root@iZj6c4e8q41ch7wv5vx7z2Z:~# awk -F: '($2==""){print$1}' /etc/shadow
root@iZj6c4e8q41ch7wv5vx7z2Z:~#
```

  - 执行：vim /etc/login.defs
    - 检查以下参数
    - PASS_MAX_DAYS 密码最长过期天数 参考值90
    - PASS_MIN_DAYS 密码最小过期天数 参考值80
    - PASS_MIN_LEN 密码最小长度 参考值8
    - PASS_WARN_AGE密 码过期警告天数参考值 7

```
#
# Password aging controls:
#
#       PASS_MAX_DAYS   Maximum number of days a password may be used.
#       PASS_MIN_DAYS   Minimum number of days allowed between password changes.
#       PASS_WARN_AGE   Number of days warning given before a password expires.
#
PASS_MAX_DAYS   99999
PASS_MIN_DAYS   0
PASS_WARN_AGE   7
```

```
#NOLOGINS_FILE
#ISSUE_FILE
PASS_MIN_LEN = 8
#PASS_MAX_LEN
```

- 预期结果：密码设置符合安全策略，且不存在空口令账号

2. root 用户远程登录限制

-
- 设置SUDO
  - 新建用户
  - useradd user_01

- passwd user_01

```
root@iZj6c4e8q41ch7wv5vx7z2Z:~# useradd user_01
root@iZj6c4e8q41ch7wv5vx7z2Z:~# passwd user_01
New password:
Retype new password:
passwd: password updated successfully
```

这两个过程输入时不会有显示

- 设置用户sudo
- chmod u+w /etc/sudoers && vim /etc/sudoers
- “ 用户名 ALL=(ALL:ALL) ALL”

```
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL
user_01 ALL=(ALL:ALL) ALL
# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```

- chmod u-w /etc/sudoers
- 测试sudo用户是否可用

```
root@iZj6c4e8q41ch7wv5vx7z2Z:~# su user_01
$ id
uid=1002(user_01) gid=1002(user_01) groups=1002(user_01)
$ sudo id

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 user_01:
uid=0(root) gid=0(root) groups=0(root)
$
```

切换至 user_01 用户

执行 id 命令

使用 sudo 执行 id 命令

输入 user_01的用户密码（不会有输出）

- 关闭root用户远程登录
  - 修改sshd_config文件，

- vim /etc/ssh/sshd_config找到PermitRootLogin,删除前面的#号并且修改为no

```
# Authentication:

#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes
-- INSERT --
```

  - sshd,systemctl restart sshd 或 systemctl restart ssh
- 预期结果：root用户无法直接登陆ssh，以sudo新用户身份登录系统后，执行sudo命令可获取root操作权

1. 检查是否存在除 root 之外 UID 为 0 的用户

-
  - 执行命令 awk -F: '($3==0){print$1}' /etc/passwd

```
└─# awk -F: '($3==0){print$1}' /etc/passwd
root
```

- 预期结果：返回值应只有“root”条目，UID为0的任何用户都拥有系统的最高特权，保证只有root的UID为0

1. 启用密钥登陆，关闭密码登陆

-
  - 生成密钥： ssh-keygen

```
└─$ sudo ssh-keygen
[sudo] password for kali:
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa): /home/kali/.ssh/id_rsa
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/kali/.ssh/id_rsa
Your public key has been saved in /home/kali/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:5M0rDQT8I2a0+f1NHdDrs1Pli13RPG/2nxk5Q2UpsEA root@kali
The key's randomart image is:
+---[RSA 3072]----+
|     .. .E .  .  |
|      o.  . o. ..|
|     . +o  . ..o=|
|      *+oo    .*=|
|     o oSoo   .+*|
|        .o..  ooX|
|         . o. ooBB|
|          .  ...=B|
|               o+|
+----[SHA256]-----+
```

输入存放id_rsa的位置

- 配置id_rsa.pub
  - cat /home/kali/.ssh/id_rsa.pub >> /home/kali/.ssh/authorized_keys

```
└─# cat /home/kali/.ssh/id_rsa.pub >> /home/kali/.ssh/authorized_keys
```

- 取出 id_rsa
- 使用 id_rsa 登陆ssh

```
(base) ➜  ~ ssh kali@10.211.55.4 -i id_rsa
Linux kali 5.15.0-kali3-arm64 #1 SMP Debian 5.15.15-2kali1 (2022-01-31) aarch64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Sep 17 14:55:21 2023 from 10.211.55.2
┌──(kali㉿kali)-[~]
└─$
```

- 关闭密码登陆
  - vim /etc/ssh/sshd_config

```
# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication no
#PermitEmptyPasswords no
```

- ssh
  - systemctl restart sshd 或 systemctl restart ssh
- 预期效果：无法直接使用密码登陆，必须使用id_rsa

(1) Linux系统文件与访问控制

1. 远程连接的安全性配置

-
- 检查.netrc find / -name .netrc 2>/dev/null

```
└─# find / -name .netrc 2>/dev/null
```

> .netrc 文件是一种用于存储网络认证信息的文件，通常用于命令行工具（例如FTP、Git、Curl等）在与远程服务器进行身份验证时使用。这个文件包含了一系列匹配规则和对应的用户名密码信息，当需要连接到远程服务器时，相应的工具会检查该文件中是否有匹配的规则，并使用其对应的用户名和密码进行认证。

- 检查.rhosts find / -name .rhosts 2>/dev/null
  -

```
└─# find / -name .rhosts 2>/dev/null
```

> .rhosts 文件是Unix和类Unix系统中用于远程主机认证的文件。通常，它包含了一系列允许访问本地服务器上的文件或服务的主机名和用户列表。当远程主机尝试访问本地主机时，本地主机会检查远程主机是否在.rhosts 文件中，如果是，则允许它通过身份验证并访问相应的文件或服务。

- 预期结果：系统中无.netrc 与 .rhosts

1. umask 安全配置
   -
     - cat /etc/profile /etc/csh.login /etc/csh.cshrc /etc/bashrc 2>/dev/null |grep umask

```
└─# cat /etc/profile /etc/csh.login /etc/csh.cshrc /etc/bashrc 2>/dev/null |grep unmask
```

- 检查是否包含umask值

> umask 是一个Unix/Linux系统中用于设置文件和目录的默认访问权限的命令或参数。它代表用户掩码（user mask）。
> 在Unix/Linux系统中，每个文件和目录都分配了一组访问权限，包括读取（r）、写入（w）和执行（x）权限。这些权限决定了谁可以对文件进行何种操作。当创建新文件或目录时，默认的权限由 umask 值来确定。
> 例如，如果 umask 值设置为 022，则新创建的文件将被设置为权限 644（即用户可读写，其他人只能读取），新创建的目录将被设置为权限 755（即用户具有完全权限，其他人只能访问）。

- 预期结果：无umask值

1. 重要目录和文件的权限设置
   - 操作步骤：
   - 查找未授权的 SUID/SGID
     - find / -perm -04000 2>/dev/null
     - find / -perm -02000 2>/dev/null

```
└─#       find / -perm -04000 2>/dev/null
/var/lib/docker/overlay2/138d9c1ed1bf3ab565291ede97f5c583037003e9e8510d3cf25cf
/var/lib/docker/overlay2/138d9c1ed1bf3ab565291ede97f5c583037003e9e8510d3cf25cf
/var/lib/docker/overlay2/138d9c1ed1bf3ab565291ede97f5c583037003e9e8510d3cf25cf
/var/lib/docker/overlay2/138d9c1ed1bf3ab565291ede97f5c583037003e9e8510d3cf25cf
/var/lib/docker/overlay2/0a8369d606b26bc7ed605cd42f3069c2c8eb160edce10095b0f8a
/var/lib/docker/overlay2/0a8369d606b26bc7ed605cd42f3069c2c8eb160edce10095b0f8a
/var/lib/docker/overlay2/0a8369d606b26bc7ed605cd42f3069c2c8eb160edce10095b0f8a
/var/lib/docker/overlay2/0a8369d606b26bc7ed605cd42f3069c2c8eb160edce10095b0f8a
/var/lib/docker/overlay2/0a8369d606b26bc7ed605cd42f3069c2c8eb160edce10095b0f8a
/var/lib/docker/overlay2/0a8369d606b26bc7ed605cd42f3069c2c8eb160edce10095b0f8a
/var/lib/docker/overlay2/0a8369d606b26bc7ed605cd42f3069c2c8eb160edce10095b0f8a
/var/lib/docker/overlay2/0a8369d606b26bc7ed605cd42f3069c2c8eb160edce10095b0f8a
/usr/sbin/mount.cifs
/usr/sbin/pppd
/usr/sbin/mount.nfs
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/xorg/Xorg.wrap
```

- 检查任何人都有写权限的目录： find / -type d -perm -o+w 2>/dev/null
- 检查隐藏文件和目录
  - find / -name ".*" -print -xdev
  - find / -name ".. *" -print -xdev
- 预期结果：无未授权的suid/sgid, 无未授权的可写权限目录，无未授权的隐藏文件或目录

1. 系统 core dump 状态

> 核心转储（core dump）是指在程序崩溃或异常结束时，将程序的内存内容保存到磁盘上的一种机制。虽然核心转储对于调试应用程序非常有用，但也可能带来安全风险。

-
  - vim /etc/security/limits.conf
  -
    - * soft core 0
    - * hard core 0

```
*               soft    core            0
*               hard    core            0
#*              hard    rss             10000
#@student       hard    nproc           20
#@faculty       soft    nproc           20
#@faculty       hard    nproc           50
#ftp            hard    nproc           0
#ftp            -       chroot          /ftp
#@student       -       maxlogins       4
```

  - 预期结果：关闭core dump

(2) Linux日志审计

1. syslog 登录事件记录

-
- more /etc/rsyslog.conf
- 查看参数authpriv值

```
#
# First some standard log files.  Log by facility.
#
auth,authpriv.*                 /var/log/auth.log
*.*;auth,authpriv.none          -/var/log/syslog
#cron.*                         /var/log/cron.log
daemon.*                        -/var/log/daemon.log
kern.*                          -/var/log/kern.log
lpr.*                           -/var/log/lpr.log
mail.*                          -/var/log/mail.log
user.*                          -/var/log/user.log
```

- 预期结果：对所有登录事件都记录