استفاده از screen در کلاس درس لینوکس

من معمولا ضمن درس دادن از ویدیو پروژکتور استفاده میکنم. چند وقت پیش فکر میکردم اگه بشه از screen یه طوری استفاده کرد که دانشجو ها بتونن با ssh به سیستم من connect بشن بعد به شکل read-only وارد screen session من بشن، میتونن رو مانیتور خودشون ببینن که من چی تایپ میکنم. هم دیگه نیازی به ویدیو پروژکتور نیست هم شاید بشه با این روش یه آموزشگاه لینوکس به شکل remote راه انداخت.
برای اینکه screen بتونه session هاش رو share کنه باید فلگ setuid در permission های فایل باینری screen علامت خورده باشه، یعنی اینکه screen همیشه باید با permission های کاربر root اجرا بشه، حتی اگه کاربری غیر از root اون رو اجرا کرده باشه.
ست کردن فلگ setuid یا همون SUID ، یکی از مسائل حساس امنیتی لینوکس محسوب میشه. این روزها اکثر برنامه ها ازش استفاده نمی کنن، و اکثر admin ها ترجیح میدن نرم افزار هایی رو که اجرا شدنه شون وابسته به SUID باشه با نرم افزاهای دیگه جایگزین کنن. به عنوان مثال فرض کنید در همین فایل screen که تصمیم داریم SUID روش ست کنیم یک bug امنیتی وجود داشته باشه. هر کاربری که اون رو اجرا کنه، انگار که root اجراش کرده و اگه بدشانس باشیم می تونه هر جور آسیبی به سیستم بزنه چون به همه چیز دسترسی داره. با در نظر داشتن این نکته پیشنهاد میکنم هر چقدر که مقدوره از SUID رو سرور های production کمتر استفاده کنید.

برای ست کردن فلگ SUID روی فایل باینری screen این مراحل رو انجام میدم :

آدرس فایل باینری screen رو پیدا میکنم :

```
# which screen
/usr/bin/screen
```

چک میکنم آدرسی که پیدا کردم حتما خوده فایل باینری باشه و نکنه یه وقت symbolic link باشه. این کار رو هم میشه با ls -lh انجام داد هم با readlink -f به این شکل:

```
# ls -lh /usr/bin/screen
lrwxrwxrwx 1 root root 12 2011-01-25 20:10 /usr/bin/screen -> screen-4.0.3*
```

یا ساده تر:

```
# readlink -f /usr/bin/screen-4.0.3
/usr/bin/screen-4.0.3
```

حالا با کمک یکی از روشهای زیر SUID رو ست میکنم:

```
# chmod  u+s /usr/bin/screen-4.0.3
```

یا

```
# chmod  4755 /usr/bin/screen-4.0.3
```

چک میکنم که SUID ست درست ست شده باشه:

```
# ls -lh /usr/bin/screen-4.0.3
-rwsr-xr-x 1 root root 312K 2007-04-11 01:26 /usr/bin/screen-4.0.3*
```

راستی یه نکته داخل پرانتز، اگه خواستین همه فایل هایی رو که رو سیستم تون SUID یا SGID شون علامت خورده پیدا کنید از این دستور استفاده کنید:

```
# find / -type f \( -perm -u+s -o -perm -g+s \) -exec ls -lh '{}' \; 2> /dev/null
```

مطمئن باشید که از زیاد بودنشون وحشت خواهید کرد. پرانتز بسته.

در مرحله بعد باید یک user محدود بسازم که دانشجوها بتونن با این user به سیستم من ssh کنن. یکی از امکانات جالبه bash اینه که میشه اون رو به شکل محدودشده اجرا کرد. کافیه به اسم rbash اجرا بشه. چک میکنم اگه rbash وجود نداره می سازمش:

```
# [ ! -e /bin/rbash ] && ln -sf bash /bin/rbash
```

به شکله استفاده کردن از شرط، بدون if ، تو دستور قبل توجه کنید. بخش اول دستور، یعنی قبل از && ، داره شرط رو چک میکنه، اگه فایل rbash وجود نداشت true بر میگرد .تنها در صورتی که بخش اول true برگردونه بخش دوم اجرا میشه و یه لینک میسازه به نام rbash .

حالا میتونم یه user بسازیم که از rbash به عنوان shell استفاده کنه:

```
# useradd -m -s /bin/rbash students
```

سویچ m- برای اینه که home directory هم ساخته بشه. سویچ s- شل رو مشخص میکنه. Students هم نام کاربریه که باید ساخته بشه.

به این user یک password میدیم که بتونه login کنه:

```
# passwd students
```

تو home directory کاربر جدید یه فایل میسازم. که قراره بلا فاصله بعد از login کردن user اجرا بشه. در ادرسه:

```
/home/students/.profile
```

با این محتوا:

```
trap "" 2 3 20
clear
echo "Welcome to the Linux Class"
echo "You can enter <Ctrl-a d> to detach at any time or just close your terminal"
echo -n "Press Enter to continue..." && read
screen -x root/linux-class
exit
```

دستور trap در خط اول، با از کار انداختن بعضی از سیگنال ها، جلوی خروج user از این script و دسترسی پیدا کردن به shell رو میگیره. اولین پارامتر دستوریه که قراره بعد از رسیدن سیگنال های خاصی اجرا بشه، "" یعنی هیچ چیز اجرا نشه، پارامتر های بعدی سیگنال ها رو مشخص می کنن. پس دستور trap باعث میشه بعد از رسیدن سیگنال های 2 و 3 و 20 هیچ دستوری اجرا نشه. لیست همه سگنال های موجود و نام اونها رو می تونید با دستور kill -l یا trap -l ببنید.

- سیگنال شماره 2 یا SIGINT که بهش interrupt هم گفته میشه وقتی اتفاق میوفته که کاربر ctrl-c بزنه.
- سیگنال شماره 3 یا SIGQUIT که بهش quit هم گفته میشه وقتی اتفاق میوفته که کاربر ctrl-\ بزنه.
- سیگنال شماره 20 یا SIGTSTP که بهش temporary stop هم گفته میشه وقتی اتفاق میوفته که کاربر ctrl-z بزنه.

برای اطلاعات بیشتر در مورد سایر سیگنا ها میتونید به آدرس زیر مراجعه کنید:

http://en.wikipedia.org/wiki/Signal_%28computing%29

دستور clear صفحه رو پاک میکنه. دستور echo به user خوشامد میگه. دستور echo ی بعدی به user اطلاع میده که برای خروج از screen میتونه از crtl-a و سپس d استفاده کنه و اخرین دستور echo که با read به شکل AND اجرا میشه، منتظر زدن کلید enter میمونه و بعد از زدن enter توسط user دستور screen یک session از کاربر root به نام linux-class رو باز میکنه. و در نهایت با اجرای exit این script به پایان میرسه.

باید فایل جدیدی رو که ساختم تو فایل bash_profile. هم صدا بزنمش. تو این آدرس یه فایل جدید میسازم:

/home/students/.bash_profile

با این محتوا:

```
source  ~/.profile
```

در نهایت تنها کاری که مونده ساختن یه screen session که دانشجو ها به طور read-only واردش بشن. این کار رو توسط یه script انجام میدم. یه فایل میسازم در آدرس:

/root/linux-class.sh

با این محتوا:

```
source /etc/profile
screen -wipe
if ! ls /root/.screen/*linux-class* &> /dev/null; then
  screen -dmS linux-class
fi
TMP=$(mktemp)
cat > ${TMP} << EOF
multiuser on
acladd students
```

```
aclchg students -wx "#,?"
aclchg students +x "detach"
EOF
screen -r linux-class -X source ${TMP}
rm ${TMP}
```

دستور اول محتویات فایل profile رو میخونه که هر چی لازمه load بشه داخله screen . مثلا ، promopt ، PATH ، HOSTNAME .
دستور screen -wipe چک میکنه اگر session های dead وجود داشته باشن پاکشون میکنه. مثلا وقتی شما داخل یک session هستین و برق سیستم قطع میشه، یه dead session رو سیستم جا میمونه که بهتره پاک بشه.
دستور if چک میکنه اگه قبلا session ی به اسم linux-class وجود نداشته true برمیگردونه. نمی خوام دو تا session هم نام داشته باشم، اون وقت وارد شدن بهشون سخت میشه.
دستور screen بعدی یه session جدید میسازه به نام linux-session (با S-) و اون رو detach میکنه (با dm-)
دستور mktemp یه فایل temp میسازه و ادرسش رو در متغییر TMP قرار میده.
دستور cat چند خط بعدی رو می نویسه تو فایل TMP تا به خط EOF برسه.
دستور multiuser on به screen خواهد گفت که به user های دیگه هم اجازه بده به این session وصل بشن.
دستور acladd students به screen خواهد گفت که user ی به اسم students اجازه داره با این session وصل بشه.
اولین دستور aclchg میگه students اجازه نوشتن نداره(w-) در تمام پنجره ها (#) و اجازه اجرا کردن همه command ها (؟) رو، نداره (x-).
دومین دستور aclchg میگه students اجازه داره اجرا کنه(x+) دستور detach رو. یعنی میتونه با زدن ctrl-a و d از session خارج بشه.
EOF به دستور cat میگه که محتویات فایل TMP تا همینجا بود.
دستور screen بعدی فایل TMP رو توی session ی به نام linux-class اجرا میکنه.
و آخرین دستور فایل TMP رو پاک میکنه.

حالا باید به این فایل permission اجرایی بدم :

```
# chmod +x /root/linux-class.sh
```

واجرا کنمش:

```
# /root/linux-class.sh
```

چک میکنم که session جدید ساخته شده باشه :

```
# screen -ls
There is a screen on:
        25619.linux-class       (Multi, detached)
1 Socket in /root/.screen.
```

و حالا می تونم واردش بشم :

```
# screen -x linux-class
```

فرض کنیم IP سیستم من باشه 192.168.1.1 ، دانشجو ها میتونن با این دستور وارد session بشن:

ssh students@192.168.1.1

پاورقی: تو مستندات screen نوشته شده که میشه aclchg رو تو فایل screenrc. نوشت. نوشتم ولی کار نکرد. mailing list شون رو که گشتم دیدم سال 2008 و 2010 سه نفر همین سوال رو پرسیدن که چرا aclchg تو فایل screenrc. کار نمیکنه و کسی هم بهشون جواب نداده. احتمالا bug داره. در نهایت با استفاده از فایل TMP مشکل رو حل کردم.

آدرس این نوشته در فرمت های دیگر:
odt: http://pmoghadam.com/homepage/Pages/Drafts/gnu-screen-in-linux-classroom.odt
pdf: http://pmoghadam.com/homepage/Pages/Drafts/gnu-screen-in-linux-classroom.pdf

پژمان مقدم
زنجان - 89/09/19