User security

CIS 2235 Adv Linux System Administration

Overview

There are two main types of security issues:

- Attacks & impersonations

Attacks:

- Abuse valid services: ping, HTTP, deny-of-service

Impersonation:

- take over a valid account and have the entire OS to '*play*' in
- How?
  - Use vulnerability in the system.
  - Trick a user (social engineering) to give up their password
  - Steal/crack a user's password

User accounts

User account information is in `/etc/passwd`

One line per user account

Standard format – colon delimited

```
userid:x:UID:GID:user information(5):home-dir:login-shell
```

```
almardes:x:1002:1002:Almardef Saoud S.:/home/almardes:/bin/bash
badenc:x:1003:1002:Baden Christopher P.:/home/badenc:/bin/bash
belvalr:x:1004:1002:Belval, Randi S.:/home/belvalr:/bin/bash
bernardk:x:1005:1002:Bernard, Kenneth A.:/home/bernardk:/bin/bash
brownn:x:1006:1002:Brown, Nathan S.:/home/brownn:/bin/bash
dulalh:x:1007:1002:Dulal, Hem L.:/home/dulalh:/bin/bash
fortinh:x:1008:1002:Fortin, Harliss R.:/home/fortinh:/bin/bash
haynesp:x:1009:1002:Haynes, Patrick M.:/home/haynesp:/bin/bash
main:x:1010:1002:Mai, Nancy K.:/home/main:/bin/bash
pelchatt:x:1011:1002:Pelchat, Thomas J.:/home/pelchatt:/bin/bash
pokhreld:x:1012:1002:Pokhrel, Deo D.:/home/pokhreld:/bin/bash
weeningt:x:1013:1002:Weening, Thomas M.:/home/weeningt:/bin/bash
```

The 7 cols of /etc/passwd

```
kayb:x:1014:1001:Bob Kay,,,:/home/kayb:/bin/bash
keckj:x:1015:1001:Jane Keck,,,:/home/keckj:/bin/bash
```

- Username
  - <= 8 characters, usually
- Encrypted pw
  - 'x' means a shadow password file is being used
- UID
  - Unique number, which is really what the OS uses to differentiate users
  - Ubuntu starts at 1000
- GID
  - Primary group
  - Usually, the same name as user – /etc/groups
- User information
  - Five columns of information, which vary significantly between Unixes
  - "not important" – used for finger, etc. Called GECOS
- Home-dir
  - Usually /home/<userid>
- Login shell
  - Default = /bin/bash for Ubuntu

password encryption

- Once upon a time, passwords were encrypted using a DES-based hashing algorithm (crypt) and stored in a field in /etc/password.
- Now Linux commonly uses SHA-512-based hashes stored in /etc/shadow. Recent systems (Ubuntu) use yescrypt which resists GPU and ASIC attacks.
- User-supplied password is combined with a random salt and then hashed with SHA-512 (or yescrypt or something else) and stored in the /etc/shadow file
- Users must follow complexity rules

System	Default requirements	Where set
Red Hat CentOS	8+ characters, complexity enforced	/etc/login.defs /etc/security/pwquality.conf /etc/pam.d/system-auth
Debian Ubuntu	6+ characters, complexity enforced	/etc/login.defs /etc/pam.d/common-password
FreeBSD	No constraints	/etc/login.conf

/etc/shadow

/etc/passwd has to be world-readable to get username, uid, and gid information

- So everyone could read the hashed password.

It is now very easy to crack a weakly hashed password.

So, the shadow file is a *separate file* which stores the hashed password

`/etc/shadow` is <u>not</u> world-readable

```
ldamon@cis2230a:~$ ll -l /etc/passwd /etc/shadow
-rw-r--r-- 1 root root   3515 2011-11-02 08:31 /etc/passwd
-rw-r----- 1 root shadow 4388 2011-11-03 11:46 /etc/shadow
```

Format of /etc/shadow

```
username:encoded_password:changed:minlife:maxlife:w
arn:inactive:expires:unused

 root:*:17737:0:99999:7:::
 daemon:*:17737:0:99999:7:::
 bin:*:17737:0:99999:7:::
 sys:*:17737:0:99999:7:::
 …
 sshd:*:17737:0:99999:7:::
 ldamon:$6$0zaGQLvLzstVENrk$wpEtnBQ1duy4q1GO6T5mftE8/
 PmVI9S2HfK84Py.BB7.3z4/
 QVjN0OfnrrHcfY.B3iBYlUz5oHSSrSwW9zWUv.:17914:0:99999:7:::
 ftp:*:17975:0:99999:7:::
 statd:*:17975:0:99999:7:::
```

Account password settings

View "status" (-S):

```
$ sudo passwd -S <username>
$ passwd -S ldamon
ldamon P 02/02/2021 0 99999 7 -1
```

Output:

name P,L,NP change-date min max warning inactivity

P means the account has valid password
NP means the account doesn't have valid password
L means locked

Account password settings

To lock: `$ sudo passwd -l <username>`

To unlock: `$ sudo passwd -u <username>`

Hint: view `/etc/shadow` after locking and unlocking

To <u>expire</u> an account and *force new pw*:

`$ sudo passwd -e <username>`

Administering passwords

Passwords should be *both*

1. Easy to remember
2. Hard to guess

Random passwords are great for 2), but not for 1).

If they are hard to remember, users write them down (next to the monitor), defeating the purpose.

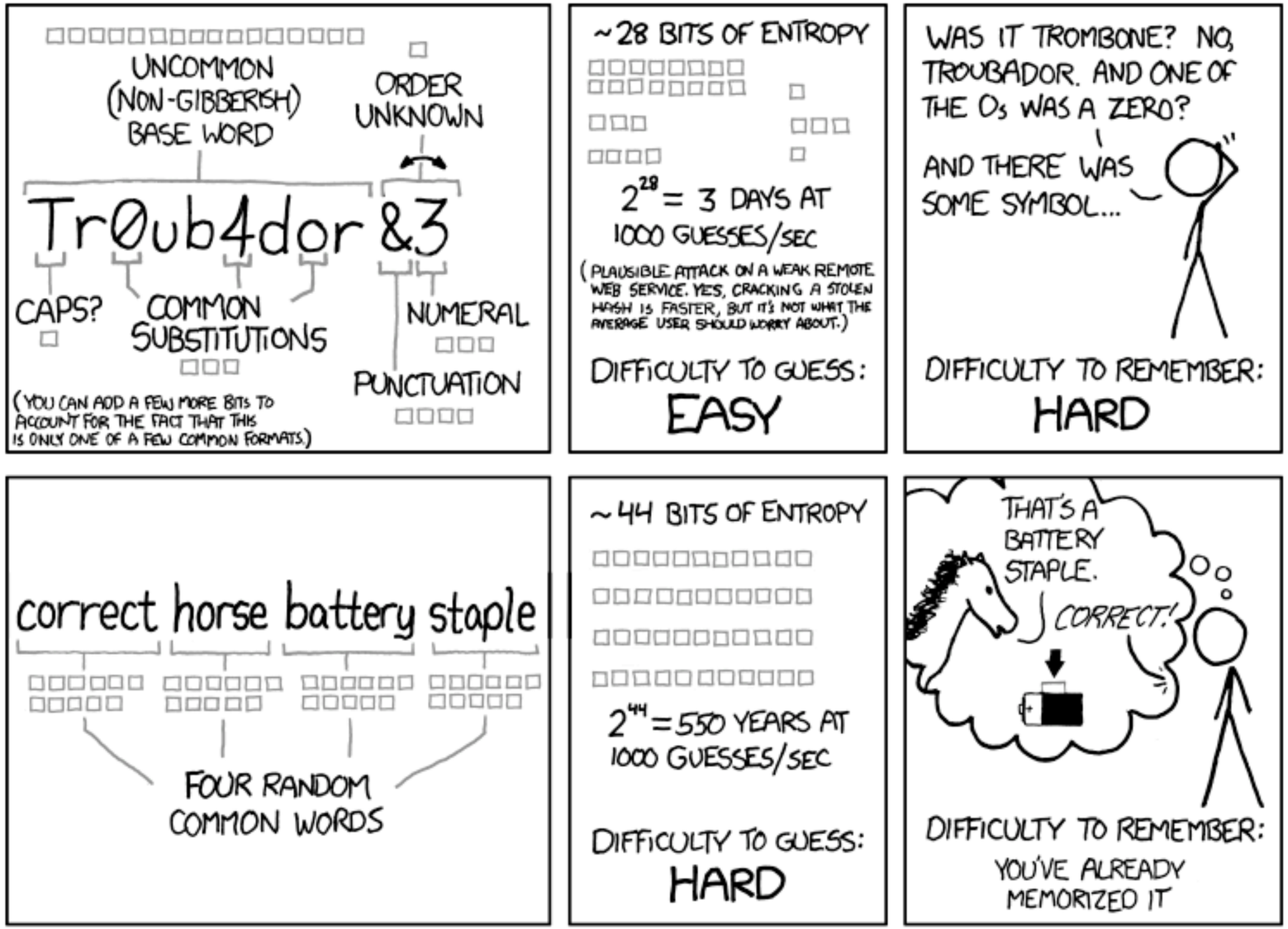
Almost any PC can now do "dictionary attacks."

Compare words in the dictionary to the encrypted password on the system

"jtr" & "crack" are password-cracking programs that are easy to set up.

Password Strength

I< < Prev Random Next > >I



THROUGH 20 YEARS OF EFFORT, WE'VE SUCCESSFULLY TRAINED EVERYONE TO USE PASSWORDS THAT ARE HARD FOR HUMANS TO REMEMBER, BUT EASY FOR COMPUTERS TO GUESS.

Breaking a password

The algorithm used to encode the password field is a one-way hash function (such as SHA-512)

A password is concatenated with a randomly generated value called the salt.

The salt is stored in the shadow file as part of the password string:

```
$6$GmWA/qEI$laTZxcKrAgBCJX2…
```

6 indicates the hash that is used (SHA-512)

GMWA/qEI is the salt, and the remainder is the hashed password

During login:

The salt is retrieved from the stored password.

The supplied password is hashed with the salt value and

The resultant value is compared with the stored hash.

It is very computationally difficult to recover the original password.

salt

More details on /etc/shadow enc_passwd columns

```
user:$type$salt$hash:options
```

Note: '$' sub-cols

```
root@ubuntu: ~
steve@ubuntu:~$ sudo su -
[sudo] password for steve:
root@ubuntu:~# passwd user06
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@ubuntu:~# passwd user07
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
root@ubuntu:~# cat /etc/shadow | grep user0[67]
user06:$6$JRtNvp1y$slTs.T4xRW0vntYfK8QDjzdW7fJpsvKjhoC1DFYhdLb8N1YphhYHtYmI5pyY7
wvWyBwLUcVxfHnIoyDed/xuT.:16925:0:99999:7:::
user07:$6$5tv2pvDH$5tNgL.3ORdxp4hYh.VL0UC8opZncYWj9Dxeh9BsTO2769Dgh0TVsiazUJg2no
0YxwaMjMABpK4P60DwVqd7cp.:16925:0:99999:7:::
root@ubuntu:~#
```

2 users with exact same pw

But different salts and therefore different hash

Group definitions

Why groups?

List groups with `$ groups` command

A group is defined in one of 2 ways:

1. A new number in the 4th col of `/etc/passwd`
2. Or an entry in `/etc/group`

```
name:x:GID:additional_users
```

The 2nd col used to be encrypted group password. There is also an `/etc/gshadow` password file.

The 4th col is a comma-separated list of users in addition to what's defined in `/etc/passwd`

Group definitions

```
ldamon@cis2230a:~$ cat /etc/group
libvirtd:x:125:ldamon,susie
kvm:x:126:ldamon,susie
```

Group Passwords

Linux only

Group password:

If there is a valid group password, then other users (non-group members) can access the group with that group password.

If the password is "*" or "!", then the group is locked, and only members have access to the group

Group shadow file `/etc/gshadow` format:

```
group_name:encoded_passwd:group_admins:additional_users
```

```
$ ls -l /etc/group /etc/gshadow
-rw-r--r-- 1 root root   1056 Nov 23 21:58 /etc/group
-rw-r----- 1 root shadow  870 Nov 23 21:58 /etc/gshadow
```

Breaking a password

However, it is still *relatively simple* nowadays to encrypt a dictionary of words using all possible 4096 salt values.

- Then, compare the encoded passwords in your /etc/shadow file with their database.
- This is a dictionary attack, one of the most common methods for gaining or expanding unauthorized access to a system.

A dictionary of 400,000 common words, names, passwords, and simple variations, with hash values using all possible salts, would be a small database by today's standards. *These can be checked in hours, not days.*

How to protect your system

- Don't allow too many tries
- Don't allow easily guessed passwords
  - Too short
  - Dictionary word
  - A word similar to the user's information
  - A proper noun that is well-known ("Vader").
- You, as the system admin, might try to crack your users' passwords using white-hat techniques, but only with permission!
- What if it is guessed? (maybe from some other account)
  - Force users to change passwords (password expiration??)
  - Don't allow past passwords

Password Aging

Per account aging values stored in /etc/shadow
Typical format:

```
username:enc_pw:changed:minlife:maxlife:warn:inactive:expires:unused
```

Password aging settings

changed – date of last change (# days since 1/1/70)

minlife – # days the user has to keep a password; 0 for changing it whenever

maxlife – # days until the user has to change the password

warn – give warning within this many days of maxlife.

inactive – number of days after password expiration when the account will automatically be disabled

expires – the account automatically expires on this date

Password aging in Linux

Command to change these options in Linux: `chage`

e.g. set the max days before user HAS to define a new password (-M):

```
ldamon@cis2230a:/etc$ sudo grep susie shadow
susie:$6$mPJY...:15264:0:99999:7:::

ldamon@cis2230a:/etc$ sudo chage -M 90 susie

ldamon@cis2230a:/etc$ sudo grep susie shadow
susie:$6$mPJY...:15264:0:90:7:::
```

List password age settings

```
root@ubuntu:~# chage -l user06
Last password change                                    : May 04, 2016
Password expires                                        : never
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 0
Maximum number of days between password change          : 99999
Number of days of warning before password expires       : 7
root@ubuntu:~#
```

```
root@ubuntu:~# chage -M 30 user06
root@ubuntu:~# chage -l user06
Last password change                                    : May 04, 2016
Password expires                                        : Jun 03, 2016
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 0
Maximum number of days between password change          : 30
Number of days of warning before password expires       : 7
root@ubuntu:~#
```

Replace `passwd` command

You can replace default `passwd` with freely available '`npasswd`'
`npasswd` checks for password triviality before changing it in `/etc/shadow`.

For Linux, this is rarely done because of the PAM system.

password complexity

- We previously mentioned complexity rules per system

System	Default requirements	Where set
Red Hat CentOS	8+ characters, complexity enforced	/etc/login.defs /etc/security/pwquality.conf /etc/pam.d/system-auth
Debian Ubuntu	6+ characters, complexity enforced	/etc/login.defs /etc/pam.d/common-password
FreeBSD	No constraints	/etc/login.conf

/etc/login.defs

Some options in `/etc/login.defs` for protection:

- PASS_MIN_DAYS
- PASS_MAX_DAYS
- OBSCURE_CHECKS_ENABLE *(see next page)*
- PASS_CHANGE_TRIES
- PASS_MIN_LEN

Default OBSCURE check

Enable some "very minimal" extra checks on password strength.

- Palindrome
  - Verifies that the new password is not a palindrome of (i.e., the reverse of) the previous one.
  - Hithere → erehtiH
- Case Change Only
  - Verifies that the new password isn't the same as the old one with a change of case.
  - Waycool → WAYCOOL
- Similar
  - Verifies that the new password isn't too much like the previous one.
  - blahblah → blahblah9
- Simple
  - Is the new password too simple? This is based on the length of the password and the number of different types of characters (alpha, numeric, etc.) used.
  - abc123
- Rotated
  - Is the new password a rotated version of the old password?
  - billy → illyb

Additional PAM modules for Linux

- There are lots of available PAM mods
- Each one can be added to the 'stack'
- The list is pretty dynamic. Even this list is out-of-date.

Required hardened passwords

PAM mods can be added to increase the default password strength

- Red Hat and Fedora systems include the `pam_cracklib` password complexity check in their <u>default</u> configuration.
- For Debian and Ubuntu systems, install either pam_cracklib (old) or pam_passwdqc.
  - To ensure strong passwords, install pam_passwdqc.
    - "password-quality-control"

e.g. `/etc/pam.d/common-password`:

```
# here are the per-package modules (the "Primary" block)
password    requisite          pam_passwdqc.so
password    [success=1 default=ignore] pam_unix.so obscure sha512
```

passwdqc

Install "`libpam-passwdqc`" from apt-get

It requires the user to use three different 'classes' of characters to ensure a dictionary word is not used

- Upper case
- Lower case
- Numbers
- Special characters

It also allows for a "pass-phrase."

- > three words
- 11 to 40 characters – a variety of characters
- Use sentence/verse/poem
  - 1) easy to remember, 2) hard to guess which one

Using libpam-passwdqc in Ubuntu

```
ldamon@vtc_cis2230:~$ sudo passwd fred

You can now choose the new password or passphrase.

A valid password should be a mix of upper and lower case letters,
digits, and other characters.  You can use an 8 character long
password with characters from at least 3 of these 4 classes, or
a 7 character long password containing characters from all the
classes.  An upper case letter that begins the password and a
digit that ends it do not count towards the number of character
classes used.

A passphrase should be of at least 3 words, 11 to 40 characters
long, and contain enough different characters.

Alternatively, if no one else can see your terminal now, you can
pick this as your password: "story$all_revolt".

Enter new password: 12345678
Weak password: not enough different characters or classes for this
length.
Try again.
```

Password history (remember=)

You can *require* a user to not-reuse a previous password

Some debate on if this is a good idea or not

Users most-always hate it

Setup in `/etc/pam.d/common-password`:

- Add "`remember=n`" to pam_unix to store pw
- Add pam_cracklib or pam_passwdqc to check past pws

```
# here are the per-package modules (the "Primary" block)
password   requisite          pam_passwdqc.so
password   [success=1 default=ignore]   pam_unix.so obscure use_authtok
                                  try_first_pass sha512 remember=5
```

Password history

When user changes their password, the old one is stored in /etc/security/opasswd

```
steve@vtc_cis2230:~$ sudo cat  /etc/security/opasswd
fred:1002:2:$1$Vd6RCQ74$H5rpby0qKkMnVeQ533dxM.,
            $1$vSYRGbq1$0VnVRhAxpfaYrds7FuWn01
```

user uid # comma-sep-enc-passwd

Then, it cannot be used for n times

```
fred@vtc_cis2230:~$ passwd
Enter new password:
Re-type new password:
Password has been already used. Choose another.
passwd: Authentication token manipulation error
passwd: password unchanged
```

Lock account after too many tries (pam_tally)

By default, a user can try to login as often they want

However, this *could be* an 'attack'

Add pam_tally.so to /etc/pam.d/common-password

```
auth required pam_tally.so onerr=fail deny=5 unlock_time=21600
```

Where:

deny=5 - Deny access if tally for this user exceeds 5 times.

unlock_time=21600 – Account reset/unlocked time. 21600 seconds = 6 hours. Default is the account is locked until the lock is removed by a manual intervention of the system administrator.

onerr=fail - If something weird happens (like unable to open the file), return with PAM_SUCESS if onerr=succeed is given, else with the corresponding PAM error code.

Lock account after too many tries (pam_tally)

Default file `/var/log/faillog` keeps login counts.

Also, you can see how many failures your user has by using

```
pam_tally --user root
```

And you can reset the failures to 0 with

```
pam_tally --user root --reset
```

Example

```
steve@vtc_cis2230:~$ su - fred
Account locked due to 7 failed logins
su: Authentication failure

steve@vtc_cis2230:~$ pam_tally
User steve (1000) has 1
User fred  (1002) has 7

steve@vtc_cis2230:~$ sudo pam_tally --user=fred --reset=0
User fred(1002) had 7

steve@vtc_cis2230:~$ su - fred
Password:
fred@vtc_cis2230:~$
```