LPI 102.2- Maintain the integrity of filesystems

Curs 2021 - 2022

ASIX M01-ISO

maintain the integrity of filesystems

Description

Key concepts:

- ❏ Verify the integrity of filesystems.
- ❏ Monitor free space and inodes.
- ❏ Repair simple filesystem problems.

Commands anf files:

- ❏ du
- ❏ df
- ❏ fsck
- ❏ e2fsck
- ❏ mke2fs
- ❏ tune2fs
- ❏ xfs_repair
- ❏ xfs_fsr
- ❏ xfs_db

Maintaining integrity

Filesystems must be maintained in order to ensure that they continue to function correctly and that the system can continue to run.

Monitoring disk information

- df
- df -h
- df -hT
- df -t fstype
- df -i
- df --output=source,fstype,itotal,iused,ipcent

```
#1
# df -h
Filesystem               Size  Used Avail Use% Mounted on
devtmpfs                 3.7G     0  3.7G   0% /dev
tmpfs                    3.7G  107M  3.6G   3% /dev/shm
tmpfs                    3.7G  2.1M  3.7G   1% /run
/dev/mapper/fedora-root   49G   17G   31G  35% /
tmpfs                    3.7G  142M  3.6G   4% /tmp
```

```
/dev/nvme0n1p2           976M  231M  678M  26% /boot
/dev/mapper/fedora-home  177G   93G   75G  56% /home
/dev/nvme0n1p1           200M  8.6M  192M   5% /boot/efi
tmpfs                    747M  120K  747M   1% /run/user/1001
/dev/loop0               195M   12M  184M   6% /mnt

# df -hT
Filesystem              Type      Size  Used Avail Use% Mounted on
devtmpfs                devtmpfs  3.7G     0  3.7G   0% /dev
tmpfs                   tmpfs     3.7G  107M  3.6G   3% /dev/shm
tmpfs                   tmpfs     3.7G  2.1M  3.7G   1% /run
/dev/mapper/fedora-root ext4       49G   17G   31G  35% /
tmpfs                   tmpfs     3.7G  142M  3.6G   4% /tmp
/dev/nvme0n1p2          ext4      976M  231M  678M  26% /boot
/dev/mapper/fedora-home ext4      177G   93G   75G  56% /home
/dev/nvme0n1p1          vfat      200M  8.6M  192M   5% /boot/efi
tmpfs                   tmpfs     747M  120K  747M   1% /run/user/1001
/dev/loop0              xfs       195M   12M  184M   6% /mnt


# df -h -t ext4
Filesystem               Size  Used Avail Use% Mounted on
/dev/mapper/fedora-root   49G   17G   31G  35% /
/dev/nvme0n1p2           976M  231M  678M  26% /boot
/dev/mapper/fedora-home  177G   93G   75G  56% /home

# df -h -t tmpfs
Filesystem      Size  Used Avail Use% Mounted on
tmpfs           3.7G  103M  3.6G   3% /dev/shm
tmpfs           3.7G  2.1M  3.7G   1% /run
tmpfs           3.7G  142M  3.6G   4% /tmp
tmpfs           747M  120K  747M   1% /run/user/1001

# df -i
Filesystem               Inodes  IUsed    IFree IUse% Mounted on
devtmpfs                 951076    600   950476    1% /dev
tmpfs                    956016    104   955912    1% /dev/shm
tmpfs                    956016   1164   954852    1% /run
/dev/mapper/fedora-root 3276800 332025  2944775   11% /
tmpfs                    956016     74   955942    1% /tmp
/dev/nvme0n1p2            65536    103    65433    1% /boot
/dev/mapper/fedora-home 11788288  53184 11735104    1% /home
/dev/nvme0n1p1                0      0        0     - /boot/efi
tmpfs                    956016    120   955896    1% /run/user/1001
```

Monitor disk space

- du
- du -s
- du -sh
- du -d --max-depth
- du -c
- du -a
- du -S
- du --exclude

```
#2
# pwd
/home/images/VM

# du -sh
33G    .

# du -sh /boot/
237M   /boot/
```

```
# pwd
/home/images

# du -sh
59G     .
# pwd
/home/images

# du -d 1 -h
33G     ./VM
1.7M    ./OpenStack-Labs
13G     ./w10
59G     .

# du -d 2 -h -c
33G     ./VM
1.3M    ./OpenStack-Labs/labs-stable-rocky
1.7M    ./OpenStack-Labs
460K    ./w10/Logs
13G     ./w10
59G     .
59G     total

# du -d1 -h /home/
36K     /home/pere
59G     /home/images
36K     /home/anna
72M     /home/guest
16K     /home/lost+found
36K     /home/pau
35G     /home/ecanet
93G     /home/

# pwd
/home/images/VM

# du -sh --exclude=*.qcow2
15G     .
# du -sh
33G     .
```

Command: tune2fs

The tune2fs command can view and change some of the settings for ext2, ext3, and ext4 filesystems.

- -l List the superblock information for a filesystem.
- -c Change the maximum number of times that a filesystem may be mounted before it is required to have a full filesystem check. The default value is normally 30 times. This can be disabled by setting the value to 0.
- -i Change the maximum time interval between when a filesystem is forced to have a full filesystem check. The default value is 180, meaning 180 days. This can be disabled by setting the interval to 0.
- -j Create a journal file for an ext2 filesystem, allowing it to be mounted as an ext3 or ext2 filesystem.
- -o Specify default mount options. By default, the RedHat derived distributions specify that acl and user_xattroptions are added to filesystems created during installation. When applying multiple options, they need to be comma separated.

By default, each filesystem will have a full system check during the boot process either every 180 days or after 30 mounts, whichever comes first.

```
#3
# mount linux.img /mnt/

# losetup
NAME       SIZELIMIT OFFSET AUTOCLEAR RO BACK-FILE                 DIO LOG-SEC
/dev/loop0         0      0         1  0 /home/images/VM/linux.img   0     512

# tune2fs -l /dev/loop0
tune2fs 1.45.5 (07-Jan-2020)
Filesystem volume name:   Linux
Last mounted on:          /mnt
Filesystem UUID:          c2e64874-2258-492d-8954-0fa7643b8c10
Filesystem magic number:  0xEF53
Filesystem revision #:    1 (dynamic)
Filesystem features:      has_journal ext_attr resize_inode dir_index filetype
needs_recovery extent 64bit flex_bg sparse_super large_file huge_file dir_nlink
extra_isize metadata_csum
Filesystem flags:         signed_directory_hash
Default mount options:    user_xattr acl
Filesystem state:         clean
Errors behavior:          Continue
Filesystem OS type:       Linux
Inode count:              51200
Block count:              204800
Reserved block count:     10240
Free blocks:              192683
Free inodes:              51188
First block:              1
Block size:               1024
Fragment size:            1024
Group descriptor size:    64
Reserved GDT blocks:      256
Blocks per group:         8192
Fragments per group:      8192
Inodes per group:         2048
Inode blocks per group:   256
Flex block group size:    16
Filesystem created:       Tue Oct 26 17:13:43 2021
Last mount time:          Tue Oct 26 18:53:53 2021
Last write time:          Tue Oct 26 18:53:53 2021
Mount count:              6
Maximum mount count:      -1
Last checked:             Tue Oct 26 17:13:43 2021
Check interval:           0 (<none>)
Lifetime writes:          333 kB
Reserved blocks uid:      0 (user root)
Reserved blocks gid:      0 (group root)
First inode:              11
Inode size:              128
Journal inode:            8
Default directory hash:   half_md4
Directory Hash Seed:      b66bf3ca-4719-497a-a680-e24449437953
Journal backup:           inode blocks
Checksum type:            crc32c
Checksum:                 0x9c6a0341
```

```
#4
# tune2fs -c0 -i0 /dev/sdb1

# tune2fs -o acl,user_xattr /dev/sdb1

# tune2fs -l /dev/sdb1
```

Fixing filesystems

Filesystems may require repair for a number of reasons:

- Sudden power outage resulting in unclean dismount of partitions. An uninterruptible power supply (UPS) is typically used to prevent this.
- Ejecting removable read/write media, such as a USB key or flash drive, before a clean dismount has occurred (i.e., removing the media before issuing the umount command).
- Loss of network connectivity between a host and a partition that is mounted as a network file system (NFS).
- Normal wear leading to corrupted disk sectors and gradual disk failure.
- Electrical surges or exposure to magnetic fields.
- Malware.

The ***fsck*** command is used to check filesystems for consistency issues and to repair those issues when found. fsck is a front-end command.

Another non-interactive approach is to force a filesystem check to occur, by executing the `touch /forcefsck` command as the root user. If this file exists at boot time, then the fsck command is executed on all filesystems in the /etc/fstab file that have a non-zero value for the FSCK column.

- fsck /dev/sda1
- fsck -f /dev/sda1
- fsck -y /dev/sda1

- -A This will check all filesystems listed in /etc/fstab.
- -C Displays a progress bar when checking a filesystem. Currently only works on ext2/3/4 filesystems.
- -N This will print what would be done and exit, without actually checking the filesystem.
- -R When used in conjunction with -A, this will skip checking the root filesystem.
- -V Verbose mode, prints more information than usual during operation. This is useful for debugging.
- -p This will attempt to automatically fix any errors found. If an error that requires intervention from the system administrator is found, e2fsck will provide a description of the problem and exit.
- -y This will answer y (yes) to all questions.
- -n The opposite of -y. Besides answering n (no) to all questions, this will cause the filesystem to be mounted read-only, so it cannot be modified.
- -f Forces e2fsck to check a filesystem even if is marked as “clean”, i.e. has been correctly unmounted.

```
#5
# fsck /dev/loop0
fsck from util-linux 2.35.2
```

```
e2fsck 1.45.5 (07-Jan-2020)
/dev/loop0 is mounted.
e2fsck: Cannot continue, aborting.

# fsck /dev/nvme0n1p1
fsck from util-linux 2.35.2
fsck.fat 4.1 (2017-01-24)
0x25: Dirty bit is set. Fs was not properly unmounted and some data may be corrupt.
1) Remove dirty bit
2) No action

# losetup /dev/loop0 linux.img

# fsck /dev/loop0
fsck from util-linux 2.35.2
e2fsck 1.45.5 (07-Jan-2020)
Linux: clean, 12/51200 files, 12117/204800 blocks

# fsck -f  /dev/loop0
fsck from util-linux 2.35.2
e2fsck 1.45.5 (07-Jan-2020)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
Linux: 12/51200 files (0.0% non-contiguous), 12117/204800 blocks
```

The ***e2fsck*** command is the filesystem checker called by fsck for ext2, ext3, and ext4 filesystems.

```
#6
# e2fsck -f  /dev/loop0
e2fsck 1.45.5 (07-Jan-2020)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
Linux: 12/51200 files (0.0% non-contiguous), 12117/204800 blocks
```

The ***superblock*** is a critical component to the filesystem, and when corrupted, it is difficult to recover. The backup superblocks are used by the fsck command to fix a corrupted superblock. Use fsck -b backup-superbloc-number when the primary superblock has been corrupted.

```
#7
# dumpe2fs 1.45.5 (07-Jan-2020)
  Primary superblock at 1, Group descriptors at 2-3
  Backup superblock at 8193, Group descriptors at 8194-8195
  Backup superblock at 24577, Group descriptors at 24578-24579
  Backup superblock at 40961, Group descriptors at 40962-40963
  Backup superblock at 57345, Group descriptors at 57346-57347
  Backup superblock at 73729, Group descriptors at 73730-73731

# e2fsck -b 8193 /dev/sdb1
```

In most cases, after running the fsck command, the fixed filesystem is mounted and no further action is required. However, if the fsck command produces any unreferenced file errors, it may be prudent to look inside the *lost+found directory* that is located in the mount point directory of the filesystem.

```
#8
```

```
# mount linux.img /mnt/

# ls /mnt/
lost+found  message.txt
```

Repairing xfs filesystems

XFS filesystems are repaired differently than other file systems. As XFS is capable of containing immensely large numbers of files (inodes), this would make boot-time checking impractical, even using a file system journal, which XFS does.

XFS relies on the journal for most error correction, or when dealing with the filesystem being marked as not having been properly unmounted. If a filesystem is not cleanly unmounted, it is marked as needing to be checked, which XFS does by replaying the journal log—usually this is enough to ensure that the data is written correctly.

In the eventuality that you must attempt to repair an XFS filesystem using ***xfs_repair***, it's fairly straightforward to use. The xfs_repair command can only be run on an unmounted filesystem.

```
#9
# losetup /dev/loop0 xfs.img

# mount /dev/loop0 /mnt/
# umount /mnt

# xfs_repair /dev/loop0
Phase 1 - find and verify superblock...
Phase 2 - using internal log
        - zero log...
        - scan filesystem freespace and inode maps...
        - found root inode chunk
Phase 3 - for each AG...
        - scan and clear agi unlinked lists...
        - process known inodes and perform inode discovery...
        - agno = 0
        - agno = 1
        - agno = 2
        - agno = 3
        - process newly discovered inodes...
Phase 4 - check for duplicate blocks...
        - setting up duplicate extent list...
        - check for inodes claiming duplicate blocks...
        - agno = 1
        - agno = 3
        - agno = 2
        - agno = 0
Phase 5 - rebuild AG headers and trees...
        - reset superblock...
Phase 6 - check inode connectivity...
        - resetting contents of realtime bitmap and summary inodes
        - traversing filesystem ...
        - traversal finished ...
        - moving disconnected inodes to lost+found ...
Phase 7 - verify and correct link counts...
done
```

```
# 10
# xfs_repair -L /dev/loop0
Phase 1 - find and verify superblock...
Phase 2 - using internal log
        - zero log...
        - scan filesystem freespace and inode maps...
        - found root inode chunk
Phase 3 - for each AG...
        - scan and clear agi unlinked lists...
        - process known inodes and perform inode discovery...
        - agno = 0
        - agno = 1
        - agno = 2
        - agno = 3
        - process newly discovered inodes...
Phase 4 - check for duplicate blocks...
        - setting up duplicate extent list...
        - check for inodes claiming duplicate blocks...
        - agno = 0
        - agno = 1
        - agno = 2
        - agno = 3
Phase 5 - rebuild AG headers and trees...
        - reset superblock...
Phase 6 - check inode connectivity...
        - resetting contents of realtime bitmap and summary inodes
        - traversing filesystem ...
        - traversal finished ...
        - moving disconnected inodes to lost+found ...
Phase 7 - verify and correct link counts...
Maximum metadata LSN (1:20) is ahead of log (1:2).
Format log to cycle 4.
done
```

In situations where manual repair of the XFS filesystem is desired, the xfs_db command can be used to initiate such repair on an XFS filesystem. The ***xfs_db*** command is used to perform debugging options and possible repairs to an XFS filesystem

```
#11
# xfs_db -x /dev/loop0
xfs_db>
ablock filoff -- set address to file offset (attr fork)
addr [field-expression] -- set current address
agf [agno] -- set address to agf header
agfl [agno] -- set address to agfl block
agi [agno] -- set address to agi header
back -- move to the previous location in the position ring
blockfree -- free block usage information
blockget [-s|-v] [-n] [-t] [-b bno]... [-i ino] ... -- get block usage and check consistency
blockuse [-n] [-c blockcount] -- print usage for current block(s)
bmap [-ad] [block [len]] -- show block map for current file
btdump [-a] [-i] -- dump btree
btheight [-b blksz] [-n recs] [-w max|-w min] btree types... -- compute btree heights
convert type num [type num]... type -- convert from one address form to another
crc [-i|-r|-v] -- manipulate crc values for V5 filesystem structures
daddr [d] -- set address to daddr value
dblock filoff -- set address to file offset (data fork)
debug [flagbits] -- set debug option bits
dquot [-g|-p|-u] id -- set current address to a group, project or user quota block for given ID
echo [args]... -- echo arguments
forward -- move forward to next entry in the position ring
frag [-a] [-d] [-f] [-l] [-q] [-R] [-r] [-v] -- get file fragmentation data
freesp [-bcdfs] [-A alignment] [-a agno]... [-e binsize] [-h h1]... [-m binmult] -- summarize free space for filesystem
fsblock [fsb] -- set address to fsblock value
fsmap [start_fsb] [end_fsb] -- display reverse mapping(s)
hash string -- calculate hash value
help [command] -- help for one or all commands
info -- pretty-print superblock info
inode [inode#] -- set current inode
label [label] -- write/print FS label
log [stop|start <filename>] -- start or stop logging to a file
logres -- dump log reservations
metadump [-a] [-e] [-g] [-m max_extent] [-w] [-o] filename -- dump metadata to a file
ncheck [-s] [-i ino] ... -- print inode-name pairs
pop -- pop location from the stack
print [value]... -- print field values
push [command] -- push location to the stack
quit -- exit xfs_db
ring -- show position ring or move to a specific entry
sb [agno] -- set current address to sb header
source source-file -- get commands from source-file
```

```
stack -- view the location stack
type [newtype] -- set/show current data type
uuid [uuid] -- write/print FS uuid
version [feature | [vnum fnum]] -- set feature bit(s) in the sb version field

# xfs_db -r /dev/sda3
```

The ***xfs_fsr*** command works on one file at a time, noting where it's data is, compacting and reorganizing files to be as streamlined as possible in order to improve read and write times, thus affecting system performance.

```
# xfs_fsr -v /dev/sda3

# xfs_fsr -t 3600
```

Example exercices

1. Show the disk partitions usage.
2. Show the disk usage of ext4 devices.
3. Show the disk usage of tmpfs devices.
4. Calculate the disk usage of /home.
5. Calculate the disk usage of /home grouping by user.

6. Change the number of mounts needed to check the linux.img filesystem.
7. Change the interval of time necessary for check the linux.img filesystem.
8. Show the linux.img disk information using tune2fs.
9. Show the superblocks of linux.img

10. Check the root filesystem.
11. Check the linux.img filesystem.
12. Check the windows.img filesystem.
13. Check the xfs.img filesystem

14. LPI Exercices 104.2 Maintain the integrity of filesystems