



Εθνικό Μετσόβιο Πολυτεχνείο  
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

## Εργαστήριο Λειτουργικών Συστημάτων

### 3<sup>η</sup> Εργαστηριακή Άσκηση

Εξερεύνηση Συστήματος Αρχείων σε Περιβάλλον Linux

ΟΜΑΔΑ: oslab10

*Ασπρογέρακας Ιωάννης (03118942)  
Κανατάς Άγγελος-Νικόλαος (03119169)*

## Η εικόνα fsdisk1.img

Προσαρτούμε την δωσμένη εικόνα δίσκου **fsdisk1.img** στην εικόνική μηχανή μας επεκτείνοντας το utopia script που μας έχει δωθεί από την προηγούμενη άσκηση ως εξής:

```
# In the last 3 lines we add the virtual disk images for ext2fs exercise
exec $QEMU -enable-kvm -M pc -m $UTOPIA_MEMORY_MB \
-smp 2 -drive file=$QCOW2_PRIVATE_FILE,if=virtio \
-net nic -net user,hostfwd=tcp:$UTOPIA_SSH_INTERFACE:$UTOPIA_SSH_PORT-:22 \
-vnc 127.0.0.1:0 \
-nographic -monitor /dev/null \
-chardev socket,id=sensors0,host=lunix.cslab.ece.ntua.gr,port=49153,ipv4=on \
-device isa-serial,chardev=sensors0,id=serial1,index=1 \
-fsdev local,id=fsdev0,path="$SHARED_FS_DIR",security_model=none \
-device virtio-9p-pci,fsdev=fsdev0,mount_tag=shared \
-drive file=./disk-images/fsdisk1-c8e29319b.img,format=raw,if=virtio \
-drive file=./disk-images/fsdisk2-4acfb06a7.img,format=raw,if=virtio \
-drive file=./disk-images/fsdisk3-06a5d95bc.img,format=raw,if=virtio \
$*
```

Κάνοτας **lsblk** βλέπουμε τα block devices που υπάρχουν στο VM μας:

```
root@utopia:/home/user/shared/fs# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
fd0      2:0    1   4K  0 disk
sr0     11:0    1 1024M  0 rom
vda    254:0    0   11G  0 disk
`-vda1 254:1    0   11G  0 part /
vdb    254:16   0   50M  0 disk
vdc    254:32   0   20M  0 disk
vdd    254:48   0   20M  0 disk
root@utopia:/home/user/shared/fs# blkid
/dev/vda1: UUID="ce8cc13a-d361-471d-adac-3061bf165cf4" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="46795e59-01"
/dev/vdb: LABEL="fsdisk1.img" UUID="226fd12b-1252-4058-9f5f-48e0eff741f0" BLOCK_SIZE="1024" TYPE="ext2"
/dev/vdc: LABEL="fsdisk2.img" UUID="333a01a6-6bb5-4d3b-9222-0c66cafd289b" BLOCK_SIZE="1024" TYPE="ext2"
/dev/vdd: LABEL="fsdisk3.img" UUID="82fb644f-d666-4868-9422-60e57cf39e56" BLOCK_SIZE="1024" TYPE="ext2"
```

Η συσκευή που αντιπροσώπευε τον εικονικό δίσκο στο σύστημα μας ονομάζεται **vdb**.

Την προσαρτούμε στο **/mnt** directory:

```
root@utopia:~# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
fd0      2:0    1   4K  0 disk
sr0     11:0    1 1024M  0 rom
vda    254:0    0   11G  0 disk
`-vda1 254:1    0   11G  0 part /
vdb    254:16   0   50M  0 disk
vdc    254:32   0   20M  0 disk
vdd    254:48   0   20M  0 disk
root@utopia:~# mount /dev/vdc /mnt
```

Επίσης από το παραπάνω βλέπουμε και το μέγεθος του δίσκου που είναι **50mB**.

Το σύστημα αρχείων που χρησιμοποιεί ο παραπάνω δίσκος είναι **Ext2** το οποίο φαίνεται τρέχοντας την εντολή **blkid** που μας δίνει κάποια βασίκα στοιχεία για τον εικονικό δίσκο.

Τρέχουμε την εντολή **dumpfs** η οποία μας επιστρέφει κάποια βασικά μεταδεδομένα για το σύστημα αρχείων μας:

```
Filesystem volume name: fsdisk1.img
Last mounted on: /mnt
Filesystem UUID: 226fd12b-1252-4058-9f5f-48e0eff741f0
Filesystem magic number: 0xEF53
Filesystem revision #: 1 (dynamic)
Filesystem features: (none)
Filesystem flags: signed_directory_hash
Default mount options: user_xattr acl
Filesystem state: not clean
Errors behavior: Continue
Filesystem OS type: Linux
Inode count: 12824
Block count: 51200
Reserved block count: 2560
Free blocks: 49552
Free inodes: 12810
First block: 1
Block size: 1024
Fragment size: 1024
Blocks per group: 8192
Fragments per group: 8192
Inodes per group: 1832
Inode blocks per group: 229
Filesystem created: Fri Dec 16 15:32:29 2022
Last mount time: Sun Jan 15 17:53:52 2023
Last write time: Sun Jan 15 17:53:52 2023
Mount count: 2
Maximum mount count: -1
Last checked: Fri Dec 16 15:32:29 2022
Check interval: 0 (<none>)
Lifetime writes: 17 kB
Reserved blocks uid: 0 (user root)
Reserved blocks gid: 0 (group root)
First inode: 11
Inode size: 128
Default directory hash: half_md4
Directory Hash Seed: ba15e3b8-3d92-4b5e-a629-8f9529d2280a
```

Βλέπουμε πως το σύστημα αρχείων δημιουργήθηκε στις 16 Δεκεμβρίου του 2022, προσαρτίθικε τελευταία φορά στις 15 Ιανουαρίου του 2023 στον κατάλογο **/mnt** και τροποποιήθηκε στις 15 Ιανουαρίου του 2023 οπώς φαίνονται από τα παραπάνω πεδία.

Τα παραπάνω μπορούν να φανούν και κάνοντας parse μέσω του εργαλείου **hexedit** στο superblock του **fdisk1** και με την βοήθεια του field description table του ext2 να βρούμε τα bytes που περιέχουν αυτά που μας ενδιαφέρουν.

### To field description table του Superblock στο ext2:

Starting Byte	Ending Byte	Size In Bytes	Field Description
0	3	4	Total number of inodes in file system
4	7	4	Total number of blocks in file system
8	11	4	Number of blocks reserved for superuser (see offset 80)
12	15	4	Total number of unallocated blocks
16	19	4	Total number of unallocated inodes
20	23	4	Block number of the block containing the superblock (also the starting block number, NOT always zero.)
24	27	4	$\log_2$ (block size) - 10. (In other words, the number to shift 1,024 to the left by to obtain the block size)
28	31	4	$\log_2$ (fragment size) - 10. (In other words, the number to shift 1,024 to the left by to obtain the fragment size)
32	35	4	Number of blocks in each block group
36	39	4	Number of fragments in each block group
40	43	4	Number of inodes in each block group
44	47	4	Last mount time (in POSIX time <a href="#">δες</a> )
48	51	4	Last written time (in POSIX time <a href="#">δες</a> )
52	53	2	Number of times the volume has been mounted since its last consistency check ( <a href="#">fsck <a href="#">δες</a></a> )
54	55	2	Number of mounts allowed before a consistency check ( <a href="#">fsck <a href="#">δες</a></a> ) must be done
56	57	2	Ext2 signature (0xef53), used to help confirm the presence of Ext2 on a volume
58	59	2	File system state ( <a href="#">see below</a> )
60	61	2	What to do when an error is detected ( <a href="#">see below</a> )
62	63	2	Minor portion of version (combine with Major portion below to construct full version field)
64	67	4	<a href="#">POSIX time <a href="#">δες</a></a> of last consistency check ( <a href="#">fsck <a href="#">δες</a></a> )
68	71	4	Interval (in <a href="#">POSIX time <a href="#">δες</a></a> ) between forced consistency checks ( <a href="#">fsck <a href="#">δες</a></a> )
72	75	4	Operating system ID from which the filesystem on this volume was created ( <a href="#">see below</a> )
76	79	4	Major portion of version (combine with Minor portion above to construct full version field)
80	81	2	User ID that can use reserved blocks
82	83	2	Group ID that can use reserved blocks

Γνωρίζουμε πως το πρώτο block του δίσκου είναι reserved και πως το blocksize του παραπάνω file system είναι 1024 bytes συνεπώς το offset του superblock είναι 1024 bytes και έχει μέγεθος 1024 bytes.

Έτσι ορίζουμε το offset, αριθμό bytes format και device που θέλουμε να διαβάσουμε:

```
root@utopia:~# hexdump -s 1024 -n 1024 -C /dev/vdb
00000400 18 32 00 00 00 c8 00 00 00 0a 00 00 90 c1 00 00 | .2.....
00000410 0a 32 00 00 01 00 00 00 00 00 00 00 00 00 00 00 | .2.....
00000420 00 20 00 00 00 20 00 00 28 07 00 00 90 21 c4 63 | .. ....(....!c
00000430 90 21 c4 63 02 00 ff ff 53 ef 00 00 01 00 00 00 |.!c....S.....
00000440 6d 73 9c 63 00 00 00 00 00 00 00 00 01 00 00 00 | ms.c.....
00000450 00 00 00 00 0b 00 00 00 80 00 00 00 00 00 00 00 | .....
00000460 00 00 00 00 00 00 00 00 22 6f d1 2b 12 52 40 58 | ....."o.+.R@X
00000470 9f 5f 48 e0 ef f7 41 f0 66 73 64 69 73 6b 31 2e | ._H...A.fsdisk1.
00000480 69 6d 67 00 00 00 00 00 2f 6d 6e 74 00 00 00 00 | img.... /mnt....
00000490 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | .....
*
000004e0 00 00 00 00 00 00 00 00 00 00 00 00 ba 15 e3 b8 | .....
000004f0 3d 92 4b 5e a6 29 8f 95 29 d2 28 0a 01 00 00 00 | =.K^.)...).(
00000500 0c 00 00 00 00 00 00 00 6d 73 9c 63 00 00 00 00 | .....ms.c.....
00000510 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | .....
*
00000560 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | .....
00000570 00 00 00 00 00 00 00 00 11 00 00 00 00 00 00 00 | .....
00000580 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | .....
*
00000800
```

To block size φαίνεται από τα bytes 24 – 27 που μας υποδεικνύουν πόσα shifts πρέπει να κάνουμε στο 1024 για να πάρουμε το block size στην περίπτωση μας  $\log_2(\text{block\_size}) - 10 = 0 \Rightarrow \text{block\_size} = 2^{10} = 1024$  bytes.

Για να βρούμε το συνολικό μέγεθος του fs αρκεί να διαβάσουμε το πόσα blocks έχει από τα bytes 4 – 7 και να το πολλαπλασιάσουμε με το block size έχουμε  $00\ 00\ c8\ 00 = 51200$  blocks άρα συνολικό μέγεθος  $51200 * 1024 = 52428800 = 50\text{mB}$ .

Αξίζει να σημειωθεί πως τα παραπάνω πεδιά αποθηκένονται σε **little endian** μορφή έτσι το πρώτο byte είναι το τελευταίο σε κάθε πεδίο.

Για να βρούμε τι σύστημα αρχείων περιέχει ο δίσκος κοιτάμε το πεδίο που έχει την υπογραφή του ext2 (0xef53) στα 56 – 57 byte του super block η οποία επιβεβαιώνει την ύπαρξη του ext2.

Τώρα θα βρούμε τα timestamps για το πότε δημιουργήθηκε, προσαρτίθηκε και τροποποιήθηκε τελευταία φορά μέσω ενός POSIX time converter:

Created time bytes 264-267 (0x639c8f8d) : December 16, 2022 15:32:29 PM

Last mount time bytes 44-47 (0x63c42109) : January 15, 2023 5:51:37 PM

Last mount time bytes 48-51 (0x63c42109) : January 15, 2023 5:51:37 PM

Και το μονοπάτι βρισκέται στα bytes 136-199  $0x2f6d6e74 = /mnt$ , ASCII 0 terminated string big endian

### Τι είναι ένα block σε ένα σύστημα αρχείων;

Το block σε ένα σύστημα αρχείων είναι ένας συνεχής χώρος διευθήσεων στον οποίο χωρίζουμε το δίσκο μας και δεν είναι κατά ανάγκη το ίδιο μέγεθος με ένα τομέα του δίσκου.

Όπως αναφέραμε και παραπάνω το block size του fsdisk1 είναι 1024 Bytes.

### Τι είναι το Inode σε ένα σύστημα αρχείων;

Το Inode είναι ένα struct που αναπαριστά ένα αρχείο ή κατάλογο ή συμβολικό σύνδεσμο. Δεν περιέχουν τα δεδομένα για το αρχείο αλλά έχουνε δείκτες για τα blocks που τα περιέχουν. Αυτό επιτρέπει στα Inodes να έχουνε καλά ορισμένο μέγεθος επιτρέποντας τους να μπούνε εύκολα σε ένα αριθμημένο πίνακα.

Σε αυτό το σύστημα αρχείων το Inode έχει μέγεθος 128 bytes το οποίο φαίνεται και από την tools μέθοδο στο dumpre2fs αλλά και διαβάζοντας το 88-89 bytes από την hexdump 0x0080=0d128.

Στο fsdisk1 έχουμε συνολικά 51200 blocks όπως φαίνεται από την dump2fs αλλά οπώς δειξάμε και παραπάνω για να βρούμε το συνολικό μεγέθος του δίσκου βρικάμε το ίδιο και από την hexdump διαβάζοντας τα 4-7 bytes του superblock.

#### Τι είναι το superblock στο σύστημα αρχείων ext2;

Το superblock είναι ένα ειδικό block σε κάθε block group το οποίο κρατάει σημαντικά μεταδεδομένα για όλο το σύστημα αρχείων όπως το layout ή ποιά χαρακτηριστικά χρησιμοποιήθηκαν για να δημιουργίθει το σύστημα αρχείων.

#### Πού βρίσκεται μέσα στον δίσκο σε ένα σύστημα αρχείων ext2;

Βρίσκεται στο πρώτο block κάθε block group για redundancy σε περιπτώση που δημιουργίθει κάποιο inconsistency τοπικά σε κάποιο block group έτσι ώστε να μπουρουμέ να επαναφέρουμε το δίσκο. Το πρώτο superblock βρίσκεται στο 1o block του δίσκου διότι το πρώτο είναι reserved.

Παρακάτω βλέπουμε την υπόλοιπη έξοδο του dump2fs:

```
Group 0: (Blocks 1-8192)
  Primary superblock at 1, Group descriptors at 2-2
  Block bitmap at 3 (+2)
  Inode bitmap at 4 (+3)
  Inode table at 5-233 (+4)
  7946 free blocks, 1821 free inodes, 2 directories
  Free blocks: 247-8192
  Free inodes: 12-1832

Group 1: (Blocks 8193-16384)
  Backup superblock at 8193, Group descriptors at 8194-8194
  Block bitmap at 8195 (+2)
  Inode bitmap at 8196 (+3)
  Inode table at 8197-8425 (+4)
  7959 free blocks, 1832 free inodes, 0 directories
  Free blocks: 8426-16384
  Free inodes: 1833-3664

Group 2: (Blocks 16385-24576)
  Backup superblock at 16385, Group descriptors at 16386-16386
  Block bitmap at 16387 (+2)
  Inode bitmap at 16388 (+3)
  Inode table at 16389-16617 (+4)
  7959 free blocks, 1830 free inodes, 1 directories
  Free blocks: 16618-24576
  Free inodes: 3667-5496

Group 3: (Blocks 24577-32768)
  Backup superblock at 24577, Group descriptors at 24578-24578
  Block bitmap at 24579 (+2)
  Inode bitmap at 24580 (+3)
  Inode table at 24581-24809 (+4)
  7959 free blocks, 1832 free inodes, 0 directories
  Free blocks: 24810-32768
  Free inodes: 5497-7328

Group 4: (Blocks 32769-40960)
  Backup superblock at 32769, Group descriptors at 32770-32770
  Block bitmap at 32771 (+2)
  Inode bitmap at 32772 (+3)
  Inode table at 32773-33001 (+4)
  7957 free blocks, 1832 free inodes, 0 directories
  Free blocks: 33003-33792, 33794-40960
  Free inodes: 7329-9160

Group 5: (Blocks 40961-49152)
  Backup superblock at 40961, Group descriptors at 40962-40962
  Block bitmap at 40963 (+2)
  Inode bitmap at 40964 (+3)
  Inode table at 40965-41193 (+4)
  7958 free blocks, 1831 free inodes, 1 directories
  Free blocks: 41195-49152
  Free inodes: 9162-10992

Group 6: (Blocks 49153-51199)
  Backup superblock at 49153, Group descriptors at 49154-49154
  Block bitmap at 49155 (+2)
  Inode bitmap at 49156 (+3)
```

Το superblock εμφανίζεται στην αρχή του κάθε block group αλλά αυτό σε μεταγενέστερες μορφές του ext μπορεί να μην υπάρχει σε όλα τα block group αλλά μόνο στα μονά ή στα ζυγά.

### Τι είναι ένα block group στο σύστημα αρχείων ext2;

Ενά block group αποτελείται από data blocks και inodes τα οποία δημιουργούν στην ουσία μια συνεχή ομάδα από blocks. Κάθε block group αποτελείται από το superblock (στην περίπτωση μας), ένα block bitmap για τα unallocated blocks του group, ένα inode bitmap για τα unallocated inodes του group, έναν πίνακα που περιέχει όλα τα inodes structs που ανήκουν στο group και τέλος τα datablocks που αντιστοιχούν στο block group.

Στο fdisk1 έχουμε 7 block groups όπως φαίνεται και από την παραπάνω έξοδο, για να βρούμε αυτή την πληροφορία και από το hexdump αρκεί να διαβάσουμε πόσα inodes αντιστοίχουν σε κάθε block group και διαίρεσουμε με αυτό τον αριθμό τα συνολικά inodes. Διαβάζουμε τα  $40\text{-}43$  bytes  $\Rightarrow 0x728 = 1832$  inodes/group και τα  $0\text{-}3$  bytes για τα συνολικά inodes  $0x3218 = 12824$  inodes. Άρα groups = total inodes / inodes/group =  $12824/1832 = 7$  groups.

Τέλος εάν κανούμε hexdump με offset την κάθε αρχή του block group θα δούμε πως το superblock είναι το πρώτο block κάθε group block.

### Τι είναι ο block group descriptor στο σύστημα αρχείων ext2;

To block group descriptor περιέχει πληροφόριες για το που βρίσκονται σημαντικές δομές δεδομένων για το block group που αντιπροσωπεύει π.χ. που βρίσκεται το πρώτο inode στο block group.

Ο block group descriptor κάθε block group έχει μέγεθος 32 byte και εμφανίζεται σε ένα table μάζι με ολούς τους αλλούς BGD διατεταγμένοι σε σειρά για το κάθε BG. To table αυτό εμφανίζεται σε κάθε block group αμέσως μέτα από το redundant copy του superblock. Αυτό γίνεται και πάλι για redundancy σε περίπτωση βλάβης να μπορούμε να επαναφέρουμε τις σημαντικές πληροφορίες του BGD από κάποιο άλλο BG σε πιθανώς κάποιο άλλο sector που δεν έχει επιρεαστεί.

### Τι είναι το block bitmap και τι το inode bitmap; Πού βρίσκονται μέσα στον δίσκο;

Όπως λέει και το όνομα είναι bitmaps στα οποία το 1 αντιπροσωπεύει πως το block ή inode indexed στο bit που κοιτάμε (προσοχή στο little endian) είναι δεσμευμένα η αντίστοιχα εάν είναι 0 τότε είναι unallocated. Βρίσκονται και αυτά σε κάθε BG μετά το BGD table.

### Τι είναι τα inode tables; Πού βρίσκονται μέσα στον δίσκο;

To inode table είναι ένας λογικός συνεχής χώρος από blocks που περιέχει τα inode structs για το αντίστοιχο BG και βρίσκεται μετά από τα bitmaps σε κάθε BG.

### Τι πεδία περιέχει το κάθε inode; Πού αποθηκεύεται μέσα στον δίσκο;

Inode Data Structure			
Starting Byte	Ending Byte	Size	Field Description
0	1	2	Type and Permissions (see below)
2	3	2	User ID
4	7	4	Lower 32 bits of size in bytes
8	11	4	Last Access Time (in POSIX time <a href="#">#</a> )
12	15	4	Creation Time (in POSIX time <a href="#">#</a> )
16	19	4	Last Modification time (in POSIX time <a href="#">#</a> )
20	23	4	Deletion time (in POSIX time <a href="#">#</a> )
24	25	2	Group ID
26	27	2	Count of hard links (directory entries) to this inode. When this reaches 0, the data blocks are marked as unallocated.
28	31	4	Count of disk sectors (not Ext2 blocks) in use by this inode, not counting the actual inode structure nor directory entries linking to the inode.
32	35	4	Flags (see below)
36	39	4	Operating System Specific value #1
40	43	4	Direct Block Pointer 0
44	47	4	Direct Block Pointer 1
48	51	4	Direct Block Pointer 2
52	55	4	Direct Block Pointer 3
56	59	4	Direct Block Pointer 4
60	63	4	Direct Block Pointer 5
64	67	4	Direct Block Pointer 6
68	71	4	Direct Block Pointer 7
72	75	4	Direct Block Pointer 8
76	79	4	Direct Block Pointer 9
80	83	4	Direct Block Pointer 10
84	87	4	Direct Block Pointer 11
88	91	4	Singly Indirect Block Pointer (Points to a block that is a list of block pointers to data)
92	95	4	Doubly Indirect Block Pointer (Points to a block that is a list of block pointers to Singly Indirect Blocks)
96	99	4	Triply Indirect Block Pointer (Points to a block that is a list of block pointers to Doubly Indirect Blocks)
100	103	4	Generation number (Primarily used for NFS)
104	107	4	In Ext2 version 0, this field is reserved. In version >= 1, Extended attribute block (File ACL).
108	111	4	In Ext2 version 0, this field is reserved. In version >= 1, Upper 32 bits of file size (if feature bit set) if it's a file, Directory ACL if it's a directory
112	115	4	Block address of fragment
116	127	12	Operating System Specific Value #2

Το πρώτο inode του inode table του κάθε BG μπόρει να βρεθεί από το BGD.

**Πόσα μπλοκ και πόσα inodes περιέχει το κάθε block group σε αυτό το σύστημα αρχείων;**  
Τα inodes του κάθε BG όπως δείξαμε και παραπάνω είναι 1832 και τα blocks τα διάβαζουμε τα 32-35 bytes που είναι  $0x2000=8192$  blocks/group προσοχή το τελευταίο group έχει λιγότερα data blocks.

### helloworld file:

Αρχικά με την tools μέθοδο βρίσκουμε το directory που περιέχει το αρχείο και τρέχουμε stat

```
root@utopia:~# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
fd0      2:0    1   4K  0 disk 
sr0     11:0    1 1024M  0 rom 
vda    254:0    0  11G  0 disk 
`-vda1 254:1    0  11G  0 part /
vdb    254:16   0   50M  0 disk 
vdc    254:32   0   20M  0 disk 
vdd    254:48   0   20M  0 disk 
root@utopia:~# mount /dev/vdb /mnt
root@utopia:~# cd /mnt
root@utopia:/mnt# ls
dir1 dir2 lost+found
root@utopia:/mnt# cd dir2
root@utopia:/mnt/dir2# ls
helloworld
root@utopia:/mnt/dir2# cat helloworld
Welcome to the Mighty World of Filesystemsroot@utopia:/mnt/dir2# 
```

```
Inode: 3666  Type: regular  Mode: 0644  Flags: 0x0
Generation: 969212841  Version: 0x00000001
User:      0  Group:      0  Size: 42
File ACL: 0
Links: 1  Blockcount: 2
Fragment: Address: 0  Number: 0  Size: 0
ctime: 0x639c736d -- Fri Dec 16 15:32:29 2022
atime: 0x63c421a8 -- Sun Jan 15 17:54:16 2023
mtime: 0x639c736d -- Fri Dec 16 15:32:29 2022
BLOCKS:
(0):33793
TOTAL: 1
```

```
root@utopia:/mnt/dir2# dd if=/dev/vdb bs=1024 skip=33793 count=1
Welcome to the Mighty World of Filesystems1+0 records in
1+0 records out
1024 bytes (1.0 kB, 1.0 KiB) copied, 0.000208581 s, 4.9 MB/s
```

Όπως βλέπουμε το αρχείο έχει inode 3666. Εφόσον κάθε BG έχει 1832 inodes βρίσκεται στο 2ο block group το οποίο περιέχει inodes indexed από 3664-5496. Το block που περιέχει τα δεδομένα του αρχείου είναι το 33793 και το μέγεθος του αρχείου είναι 1024 bytes αφού έχει κάνει allocate ένα block. Το block που περιέχει το inode struct του συγκεκριμένου αρχείου έχει index  $8192*2 + 4 + \text{first\_block} = 16389$  αφού βρίσκεται στο 2ο block group +4 για το superblock, BGDT, Inode bitmap, block bitmap και φού είναι το 3ο κατα αντιστοιχεία inode δηλαδή  $3*128 < 1024$  αρά στο πρώτο block του inode table του 2ου BG.

Χρησιμοποιώντας hexdump:

Αρχικά για να βρούμε το συγκεκριμένο inode του αρχείου πρέπει να κάνουμε navigate στο inode του καταλόγου που το περιέχει εν προκειμένου στο dir2. Ξεκινάμε από το root inode που περιέχει το αριθμό του block με τα αντίστοιχα ονόματα και inodes του καταλόγου. Για να το βρούμε αυτό διαβάζουμε που ξεκινάει το inode table από το BGD του πρώτου block και πέρνουμε το δεύτερο inode που πάντα είναι το root. Επειτα για να βρούμε το datablock του root inode διαβάζουμε το DBP στα bytes 40-43 του inode.

### T<sub>0</sub> directory entries:

## Directory Entry

Starting Byte	Ending Byte	Size in Bytes	Field Description
0	3	4	Inode
4	5	2	Total size of this entry (Including all subfields)
6	6	1	Name Length least-significant 8 bits
7	7	1	Type indicator (only if the feature bit for "directory entries have file type byte" is set, else this is the most-significant 8 bits of the Name Length)
8	B+N-1	N	Name characters

Πλέον βρισκόμαστε στο root directory και γνωρίζουμε το inode του καταλόγου dir2 που είναι 3665 και βρίσκεται στο 2o BG στην 1η θέση, συνέπως θα διαβάσουμε το BGD του 2ou BG για να βρούμε την αρχή του inode table όπου και βρίσκεται το inode του dir2 directory.

```
root@utopia:~# hexdump -s$((GDT_OFFSET+2*GD_SIZE)) -n$((GD_SIZE)) -C /dev/vdb
00000840  03 40 00 00 04 40 00 00  05 40 00 00 17 1f 26 07  |. @... @... @... &.|  
00000850  01 00 04 00 00 00 00 00  00 00 00 00 00 00 00 00  |.....|  
00000860  
root@utopia:~# hexdump -s$((16389*BLOCK_SIZE)) -n$((INODE_SIZE)) -C /dev/vdb
01001400  ed 41 00 00 00 04 00 00  cc 21 c4 63 bd 21 c4 63  |. A.....!.c.!..c|  
01001410  bd 21 c4 63 00 00 00 00  00 00 02 00 02 00 00 00  |.!..c.....|  
01001420  00 00 00 00 06 00 00 00  ea 80 00 00 00 00 00 00 00  |.....|  
01001430  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00  |.....|  
*  
01001460  00 00 00 00 cb 51 09 05  00 00 00 00 00 00 00 00 00  |.....Q.|  
01001470  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00  |.....|  
01001480  
root@utopia:~# hexdump -s$((33002*BLOCK_SIZE)) -n$((128)) -C /dev/vdb
0203a800  51 0e 00 00 0c 00 01 00  2e 00 00 00 02 00 00 00  |Q.....|  
0203a810  0c 00 02 00 2e 2e 00 00  52 0e 00 00 e8 03 0a 00  |.....R.....|  
0203a820  68 65 6c 6c 6f 77 6f 72  6c 64 00 00 54 0e 00 00  |helloworld..T...|  
0203a830  d4 03 0f 00 2e 68 65 6c  6c 6f 77 6f 72 6c 64 2e  |.....helloworld.|  
0203a840  73 77 70 00 00 00 00 00  00 00 00 00 00 00 00 00 00  |swp.....|  
0203a850  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00  |.....|  
*  
0203a880
```

Όπως βλέπουμε επαναλάμβανουμε την παραπάνω διαδικασία για να φτάσουμε στα δεδομένα του dir2 directory αρχίκα το BGD του 2ου BG μας λέει πως το inode table του 2ου BG ξέκιναε στο block 0x4005 = 16389. Διαβάζουμε από αυτό το block το inode size και στο byte 40-43 βλέπουμε την διεύθηση του data block 0x80ea = 33002 στο οποίο βρίσκουμε το inode του helloworld αρχείου που είναι 0xe52 = 3666 δήλαδή το ακριβώς δίπλα inode στο BG 2. Αξίζει να σημειωθεί πως το δεύτερο helloworld δεν είναι σωστό directory entry διότι δεν έχει valid túpo.

```
root@utopia:~# hexdump -s$((16389*BLOCK_SIZE+INODE_SIZE)) -n$((INODE_SIZE)) -C /dev/vdb
01001480  a4 81 00 00 2a 00 00 00  a8 21 c4 63 6d 73 9c 63  |....*....!.cms.c|
01001490  6d 73 9c 63 00 00 00 00  00 00 01 00 02 00 00 00  |ms.c.....|
010014a0  00 00 00 00 01 00 00 00  01 84 00 00 00 00 00 00 00  |.....|
010014b0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00  |.....|
*
010014e0  00 00 00 00 a9 03 c5 39  00 00 00 00 00 00 00 00 00  |.....9.....|
010014f0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00  |.....|
01001500
root@utopia:~# hexdump -s$((33793*BLOCK_SIZE)) -n$((64)) -C /dev/vdb
02100400  57 65 6c 63 6f 6d 65 20  74 6f 20 74 68 65 20 4d  |Welcome to the M|
02100410  69 67 68 74 79 20 57 6f  72 6c 64 20 6f 66 20 46  |ighty World of F|
02100420  69 6c 65 73 79 73 74 65  6d 73 00 00 00 00 00 00 00  |ilesystems.....|
02100430  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00  |.....|
02100440
```

Τέλος πέρνουμε το πρώτο direct block pointer στα bytes 40-43 και βρίσκουμε τα δεδομένα του αρχείου (0x8401 = 0d33793).

## Η εικόνα *fdisk2.img*

Μας δίνεται μία νέα εικόνα δίσκου (virtual disk image) **fdisk2.img**, την οποία την συνδέουμε στην εικονική μηχανή μας, όπως και πριν επεκτείνοντας το script **utopia.sh**.

```
# In the last 3 lines we add the virtual disk images for ext2fs exercise
exec $QEMU -enable-kvm -M pc -m $UTOPIA_MEMORY_MB \
-smp 2 -drive file=$QCOW2_PRIVATE_FILE,if=virtio \
-net nic -net user,hostfwd=tcp:$UTOPIA_SSH_INTERFACE:$UTOPIA_SSH_PORT-:22 \
-vnc 127.0.0.1:0 \
-nographic -monitor /dev/null \
-chardev socket,id=sensors0,host=lunix.cslab.ece.ntua.gr,port=49153,ipv4=on \
-device isa-serial,chardev=sensors0,id=serial1,index=1 \
-fsdev local,id=fsdev0,path="$SHARED_FS_DIR",security_model=none \
-device virtio-9p-pci,fsdev=fsdev0,mount_tag=shared \
-drive file=./disk-images/fdisk1-c8e29319b.img,format=raw,if=virtio \
-drive file=./disk-images/fdisk2-4acfb06a7.img,format=raw,if=virtio \
-drive file=./disk-images/fdisk3-06a5d95bc.img,format=raw,if=virtio \
$*
```

Έπειτα, την προσαρτούμε στον κατάλογο **/mnt**:

```
root@utopia:~# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
fd0     2:0    1   4K  0 disk
sr0    11:0    1 1024M  0 rom
vda   254:0    0   11G  0 disk
`-vda1 254:1    0   11G  0 part /
vdb   254:16   0   50M  0 disk
vdc   254:32   0   20M  0 disk
vdd   254:48   0   20M  0 disk
root@utopia:~# mount /dev/vdc /mnt
```

Παραπάνω, βλέπουμε και το μέγεθος του δίσκου, μεταξύ άλλων, που είναι **20MB** (`/dev/vdc`).

Χρησιμοποιούμε την εντολή **touch** για να δημιουργήσουμε ένα κενό αρχείο **/file1** μέσα στο συγκεκριμένο ΣΑ:

```
root@utopia:/mnt# touch file1
touch: cannot touch 'file1': No space left on device
```

Παρατηρούμε, ότι η εντολή δεν πετυχαίνει με error message: **No space left on device**.

Με χρήση της εντολής **strace**, βλέπουμε ποια κλήση συστήματος απέτυχε κατά την δημιουργία του νέου αρχείου και με ποιον κωδικό λάθους:

```
openat(AT_FDCWD, "file1", O_WRONLY|O_CREAT|O_NOCTTY|O_NONBLOCK, 0666) = -1 ENOSPC (No space left on device)
utimensat(AT_FDCWD, "file1", NULL, 0)    = -1 ENOENT (No such file or directory)
write(2, "touch: ", 7touch: )              = 7
write(2, "cannot touch 'file1'", 20cannot touch 'file1')    = 20
write(2, ": No space left on device", 25: No space left on device) = 25
write(2, "\n", 1                           = 1
)                                         = 1
close(1)                                = 0
close(2)                                = 0
exit_group(1)                            = ?
+++ exited with 1 +++
```

Βλέπουμε, ότι η κλήση συστήματος `openat()` απέτυχε με κωδικό λάθους **ENOSPC**.

Ανατρέχοντας στο man page της open() βλέπουμε:

**ENOSPC** *pathname* was to be created but the device containing *pathname* has no room for the new file.

Άρα, μας λέει ότι ο δίσκος δεν έχει αρκετό χώρο.

Ας διερευνήσουμε τον δίσκο μας, για να δούμε που ακριβώς βρίσκεται το πρόβλημα.

(Εμείς ξέρουμε ήδη το πρόβλημα: ~~No free inodes~~, αλλά ας προσποιηθούμε ότι δεν το έχουμε καταλάβει ακόμα).

Αρχικά, θα μετρήσουμε πόσα αρχεία και πόσους καταλόγους περιέχει το συγκεκριμένο σύστημα αρχείων.

Με την tools μέθοδο έχουμε:

```
root@utopia:~# find /mnt -type d | wc -l  
259  
root@utopia:~# find /mnt -type f | wc -l  
4868
```

Χρησιμοποιήσαμε την εντολή **find mnt -type <type>** που απαριθμεί με newlines τα αρχεία με συγκεκριμένο τύπο (**d** για directories και **f** για regular files), όπου και δίνουμε την έξοδο της στην εντολή **wc -l** μέσω pipe, η οποία μετράει τα newlines. Έτσι, έχουμε **259 directories** και **4868 regular files** (δεν έχουμε special files όπως FIFOs, symlinks, block/char devices κ.λπ.).

Μπορούμε να τα μετρήσουμε και με τον εξής τρόπο κάνοντας χρήση του **dump2fs**:

```
root@utopia:~# dump2fs /dev/vdc  
dump2fs 1.46.2 (28-Feb-2021)  
Filesystem volume name: fsdisk2.img  
Last mounted on: /home/jimsiak/cslab-git/fs/mnt  
Filesystem UUID: 333a01a6-6bb5-4d3b-9222-0c66cafd289b  
Filesystem magic number: 0xEF53  
Filesystem revision #: 1 (dynamic)  
Filesystem features: (none)  
Filesystem flags: signed_directory_hash  
Default mount options: user_xattr acl  
Filesystem state: clean  
Errors behavior: Continue  
Filesystem OS type: Linux  
Inode count: 5136  
Block count: 20480  
Reserved block count: 0  
Free blocks: 19555  
Free inodes: 0  
First block: 1  
Block size: 1024  
Fragment size: 1024  
Blocks per group: 8192  
Fragments per group: 8192  
Inodes per group: 1712  
Inode blocks per group: 214  
Filesystem created: Fri Dec 16 15:32:30 2022  
Last mount time: Mon Jan 16 15:44:36 2023  
Last write time: Mon Jan 16 15:46:04 2023  
Mount count: 5  
Maximum mount count: -1  
Last checked: Fri Dec 16 15:32:30 2022  
Check interval: 0 (<none>)  
Lifetime writes: 925 kB  
Reserved blocks uid: 0 (user root)  
Reserved blocks gid: 0 (group root)  
First inode: 11  
Inode size: 128  
Default directory hash: half_md4  
Directory Hash Seed: 0a40a844-6747-4994-81d1-a7f0fde543e4  
Group 0: (Blocks 1-8192)  
Primary superblock at 1, Group descriptors at 2-2  
Block bitmap at 3 (+2)  
Inode bitmap at 4 (+3)  
Inode table at 5-218 (+4)  
7787 free blocks, 0 free inodes, 84 directories  
Free blocks: 406-8192  
Free inodes:  
Group 1: (Blocks 8193-16384)  
Backup superblock at 8193, Group descriptors at 8194-8194  
Block bitmap at 8195 (+2)  
Inode bitmap at 8196 (+3)  
Inode table at 8197-8410 (+4)  
7891 free blocks, 0 free inodes, 83 directories  
Free blocks: 8494-16384  
Free inodes:  
Group 2: (Blocks 16385-20479)  
Backup superblock at 16385, Group descriptors at 16386-16386  
Block bitmap at 16387 (+2)  
Inode bitmap at 16388 (+3)  
Inode table at 16389-16602 (+4)  
3877 free blocks, 0 free inodes, 92 directories  
Free blocks: 16603-20479  
Free inodes:
```

Έτσι, βρίσκουμε ότι τα directories σε κάθε bg είναι 84, 83 και 92 αντίστοιχα και άρα συνολικά έχουμε **259 directories**. Για να βρούμε τον αριθμό των regular files βρίσκουμε τον αριθμό των κατειλημμένων inode σε κάθε bg και αφαιρούμε από αυτά τα directories στο bg. Ιδιαίτερη προσοχή χρειάζεται στο bg #0, όπου πρέπει να αφαιρέσουμε τα reserved inodes #1-#10. Ακριβέστερα, πρέπει να αφαιρέσουμε από τον υπολογισμό τα reserved inodes #1-#10 χωρίς το #2 που αντιστοιχεί στο root directory, αφού το έχουμε αφαιρέσει ήδη. Έτσι έχουμε: **#regular files=1712-9-84+1712-83+1712-92=4868** (υποθέταμε ότι δεν έχουμε special files).

(Αν η ερώτηση ζητάει τον συνολικό αριθμό των αρχείων (directories, regular files, special files) τότε είναι  $1712 * 3 = 5136$  files).

Με την **hexedit** μέθοδο, κάνουμε ακριβώς ότι κάναμε και παραπάνω, διαβάζοντας byte προς byte το περιεχόμενο του superblock και GDT (για τον υπολογισμό των directories ανά bg):

```
root@utopia:~# hexdump -s 1024 -n 1024 -C /dev/vdc #inodes / bg
000000400 10 14 00 00 00 50 00 00 00 00 00 00 63 4c 00 00 |.....P.....CL..|
000000410 00 00 00 00 01 00 00 00 00 00 00 00 00 00 00 00 |.....|.....|
000000420 00 20 00 00 00 20 00 00 b0 06 00 00 76 59 c5 63 |. .... .vY.c|
000000430 f8 59 c5 63 06 00 ff ff 53 ef 01 00 01 00 00 00 |.Y.c...S....|
000000440 6e 73 9c 63 00 00 00 00 00 00 00 01 00 00 00 00 |ns.c....|
000000450 00 00 00 00 0b 00 00 00 80 00 00 00 00 00 00 00 |.....|.....|
000000460 00 00 00 00 00 00 00 00 33 3a 01 a6 6b b5 4d 3b |.....3:..k.M;|
000000470 92 22 0c 66 ca fd 28 9b 66 73 64 69 73 6b 32 2e |.".f..(.fsdisk2.| 
000000480 69 6d 67 00 00 00 00 00 2f 68 6f 6d 65 2f 6a 69 |img....../home/ji|
000000490 6d 73 69 61 6b 2f 63 73 6c 61 62 2d 67 69 74 2f |msiak/cslab-git/|
0000004a0 66 73 2f 6d 6e 74 00 00 00 00 00 00 00 00 00 00 |fs/mnt....|
0000004b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|.....|
*
0000004e0 00 00 00 00 00 00 00 00 00 00 00 0a 40 a8 44 |.....@.D|
0000004f0 67 47 49 94 81 d1 a7 f0 fd e5 43 e4 01 00 00 00 |gGI.....c....|
000000500 0c 00 00 00 00 00 00 00 6e 73 9c 63 00 00 00 00 |.....ns.c....|
000000510 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|.....|
*
000000560 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|.....|
000000570 00 00 00 00 00 00 00 00 93 04 00 00 00 00 00 00 |.....|.....|
000000580 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|.....|
*
000000800
```

```
root@utopia:~# export BLOCK_SIZE=1024
root@utopia:~# export GDT_OFFSET=$((2*BLOCK_SIZE))
root@utopia:~# export GD_SIZE=32
root@utopia:~# export INODE_SIZE=128
root@utopia:~# hexdump -s$((GDT_OFFSET)) -n$((GD_SIZE)) -C /dev/vdc
000000800 03 00 00 00 04 00 00 00 05 00 00 00 6b 1e 00 00 |.....k...|
000000810 54 00 04 00 00 00 00 00 00 00 00 00 00 00 00 00 |T.....|
000000820 #directories / bg
root@utopia:~# hexdump -s$((GDT_OFFSET+GD_SIZE)) -n$((GD_SIZE)) -C /dev/vdc
000000820 03 20 00 00 04 20 00 00 05 20 00 00 d3 1e 00 00 |. ....|.....|
000000830 53 00 04 00 00 00 00 00 00 00 00 00 00 00 00 00 |S.....|
000000840
root@utopia:~# hexdump -s$((GDT_OFFSET+2*GD_SIZE)) -n$((GD_SIZE)) -C /dev/vdc
000000840 03 40 00 00 04 40 00 00 05 40 00 00 25 0f 00 00 |. @...@...@.%...|
000000850 5c 00 04 00 00 00 00 00 00 00 00 00 00 00 00 00 |\.....|.....|
000000860
```

Έτσι, με τους ίδιους ακριβώς υπολογισμούς έχουμε: **#regular files=4868, #directories=259**.

Τώρα, θα υπολογίσουμε πόσο χώρο καταλαμβάνουν τα δεδομένα και τα μετα-δεδομένα του ΣΑ. Τα μετα-δεδομένα στο ext2fs είναι: superblock, GDT, block/inode bit maps και inode table.

Επομένως, αφού έχουμε **3 block groups** (το βλέπουμε είτε με **dump2fs** είτε με inode count/inodes per bg), έχουμε: **metadata=3** blocks (superblocks – primary + backups) + 3 blocks (GDTs primary + backups – 1 block ανά GDT γιατί  $\text{ceil}(\#bg * 32 / 1024) = 1$ ) + 6 blocks (1 block / bit map) +  $3 * 214$  blocks (inode tables – 214 blocks ανά inode table γιατί inodes per  $bg * 128 / 1024 = 214$ ) = 654 metadata blocks =  $654 * 1024$  bytes = **654 KB**.

Για τα δεδομένα του ΣΑ έχουμε: **data=block count – free blocks – metadata blocks – boot block=20480 – 19555 – 654 – 1=270 data blocks=270KB**.

Τους παραπάνω υπολογισμούς τους κάνουμε με την **tools** μέθοδο, κάνοντας χρήση του **dump2fs**, ενώ με την **hexedit** μέθοδο βλέπουμε byte-bytes τα αντίστοιχα πεδία στο superblock.

Επαληθεύουμε τους υπολογισμούς μας με χρήση της εντολής **df -h**:

```
root@utopia:/mnt# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            482M    0  482M   0% /dev
tmpfs           98M  444K  98M   1% /run
/dev/vda1        11G  3.3G  6.9G  33% /
tmpfs           489M    0  489M   0% /dev/shm
tmpfs           5.0M    0  5.0M   0% /run/lock
shared          439G 132G 285G  32% /home/user/shared
tmpfs           98M    0  98M   0% /run/user/0
/dev/vdc         20M  270K  20M   2% /mnt
```

, όπου και έχουμε **270KB** για τα δεδομένα μας, όπως και υπολογίσαμε.

Το μέγεθος του συστήματος αρχείων φαίνεται παραπάνω από την εντολή **df -h**, καθώς και παραπάνω από την εντολή **lsblk**. Μπορούμε να το βρούμε επίσης και ως εξής:

```
angelos@desktop:~/utopia$ qemu-img info ./disk-images/fsdisk2*
image: ./disk-images/fsdisk2-4acfb06a7.img
file format: raw
virtual size: 20 MiB (20971520 bytes)
disk size: 20 MiB
```

Κάνοντας χρήση του **dump2fs** το βρίσκουμε ως εξής: **size=block count \* block size = 20MB**.

Τον ίδιο υπολογισμό κάνουμε και με την **hexedit** μέθοδο βλέποντας τα αντίστοιχα πεδία. Έτσι, έχουμε:

	<b>block count</b>	<b>log<sub>2</sub>(block size) -10</b>	<b>free blocks</b>								
root@utopia:~# hexdump -s 1024 -n 1024 -C /dev/vdc											
00000400 10 14 00 00 <u>00 50 00 00</u> 00 00 00 00 63 4c 00 00  .....P.....cL..	00000410 00 00 00 00 01 00 00 00 00 00 00 00 00 00 00 00  .....	00000420 00 20 00 00 00 20 00 00 b0 06 00 00 76 59 c5 63  . ....vY.c	00000430 f8 59 c5 63 06 00 ff ff 53 ef 01 00 01 00 00 00  .Y.c....S....	00000440 6e 73 9c 63 00 00 00 00 00 00 00 00 01 00 00 00  ns.c.....	00000450 00 00 00 00 0b 00 00 00 80 00 00 00 00 00 00 00  .....	00000460 00 00 00 00 00 00 00 00 33 3a 01 a6 6b b5 4d 3b  .....3...k.M;	00000470 92 22 0c 66 ca fd 28 9b 66 73 64 69 73 6b 32 2e  .".f..(.fsdisk2.	00000480 69 6d 67 00 00 00 00 00 2f 68 6f 6d 65 2f 6a 69  img...../home/ji	00000490 6d 73 69 61 6b 2f 63 73 6c 61 62 2d 67 69 74 2f  msiak/cslab-git/	000004a0 66 73 2f 6d 6e 74 00 00 00 00 00 00 00 00 00 00  fs/mnt.....	000004b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....
*											
000004e0 00 00 00 00 00 00 00 00 00 00 00 0a 40 a8 44  .....@.D	000004f0 67 47 49 94 81 d1 a7 f0 fd e5 43 e4 01 00 00 00  gGI.....C.....	00000500 0c 00 00 00 00 00 00 00 6e 73 9c 63 00 00 00 00  .....ns.c....	00000510 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....								
*											
00000560 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....	00000570 00 00 00 00 00 00 00 00 93 04 00 00 00 00 00 00  .....	00000580 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  .....									
*											
00000800											

Για να βρούμε πόσα block είναι διαθέσιμα/ελεύθερα στο ΣΑ με την μέθοδο **hexedit** βρίσκουμε το πεδίο **free blocks**, όπως φαίνεται παραπάνω. Με την **tools** μέθοδο, το βλέπουμε με χρήση του **dump2fs**. Έτσι, έχουμε: **#free blocks=0x4c63=19555 blocks**. Επομένως, το ΣΑ έχει αρκετό ελεύθερο χώρο, όπως φαίνεται και με την εντολή **df -h**.

Filesystem	Size	Used	Avail	Use%	Mounted on
udev	482M	0	482M	0%	/dev
tmpfs	98M	444K	98M	1%	/run
/dev/vda1	11G	3.3G	6.9G	33%	/
tmpfs	489M	0	489M	0%	/dev/shm
tmpfs	5.0M	0	5.0M	0%	/run/lock
shared	439G	132G	285G	32%	/home/user/shared
tmpfs	98M	0	98M	0%	/run/user/0
/dev/vdc	20M	270K	20M	2%	/mnt

Επομένως, αφού υπάρχουν διαθέσιμα block γιατί αποτυγχάνει η δημιουργία ενός νέου κενού αρχείου; Αυτό συμβαίνει, διότι όπως είδαμε παραπάνω με **dump2fs** και με **hexdump**, **δεν υπάρχουν διαθέσιμα inodes** – είναι όλα κατειλημένα. Έτσι, για να δημιουργήσουμε ένα νέο αρχείο πρέπει πρώτα να διαγράψουμε κάποιο άλλο.

## Η εικόνα **fdisk3.img**

Τέλος, έχουμε την εικόνα **fdisk3.img** (`/dev/vdd` όταν είναι συνδεδεμένη στην εικονική μηχανή – την συνδέουμε όπως κάναμε και για τις δύο παραπάνω εικόνες), η οποία περιέχει ένα σύστημα αρχείων ext2 με λάθη (είναι corrupted). Παρακάτω, θα εντοπίσουμε, θα αποδείξουμε και θα επιδιορθώσουμε αυτές τις αλλοιώσεις. Αφού επιβεβαιώσουμε ότι έχουμε την καθαρή εικόνα όπως μας δίνεται (με **sha256sum**) και ότι δεν έχουμε προσαρτήσει το ΣΑ (είναι επικίνδυνο να “τρέχουμε” fs tools όπως το **e2fsck** παρακάτω), ξεκινάμε.

```
angelos@desktop:~/utopia/disk-images$ sha256sum ./fdisk3*
06a5d95bc3787c680104291e7463254d4a5c1696e929d24deadec9cde1045bbf ./fdisk3-06a5d95bc.img
```

Για να εντοπίσουμε και να επιδιορθώσουμε τις αλλοιώσεις στο σύστημα αρχείων μας, χρησιμοποιούμε το εργαλείο **fsck**, το οποίο ελέγχει και επιδιορθώνει inconsistent συστήματα αρχείων. Στην πραγματικότητα, το **fsck** είναι ένα front-end για τα διάφορα εργαλεία επιδιόρθωσης κάθε τύπου ΣΑ (π.χ. **e2fsck** για ext2/ext3/ext4 fs).

Ένα ΣΑ μπορεί να είναι corrupted για διάφορους λόγους, όπως:

- **Power failures**
- **Άστοχίες υλικού**
- **Bugs στον πυρήνα ή και σε user space software**
- **Bad disk controllers**
- **Κακόβουλα λογισμικά κ.α.**

Πιθανές αλλοιώσεις/inconsistencies που μπορεί να προκληθούν είναι:

1. Illegal file modes
2. Οι block pointers στα inodes να μην είναι valid
3. Τα directory entries να μην έχουν valid δομή

4. Τα directory entries να περιέχουν references σε inodes που δεν χρησιμοποιούνται
5. Τα directory entries να μην περιέχουν τα ./ και ../ entries
6. Το connectivity των directories να μην είναι valid
7. Λάθος στον αριθμό των hard links count σε ένα inode
8. Λάθη στα block και inode bit maps
9. Λάθη στο superblock (π.χ. free blocks count)
10. Λάθη στο GDT (π.χ. number of unallocated blocks in group) κ.α.

Τώρα, θα τρέξουμε το εργαλείο **e2fsck** για να επιδιορθώσουμε το ΣΑ μας:

```
root@utopia:~# e2fsck /dev/vdd                                     ERROR 1
e2fsck 1.46.2 (28-Feb-2021)
fsdisk3.img contains a file system with errors, check forced.
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
First entry 'BOO' (inode=1717) in directory inode 1717 (/dir-2) should be '.'
Fix<y>? 
```

Βλέπουμε, δηλαδή, ότι το directory με inode=1717 δεν έχει την σωστή δομή (δεν περιέχει το entry ./ αλλά στην θέση του έχει το BOO). Πατάμε **y** για να επιδιορθώσουμε το σφάλμα και προχωράμε.

```
Fix<y>? yes                                              ERROR 2
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Inode 3425 ref count is 1, should be 2. Fix<y>? 
```

Βλέπουμε ότι το reference count για το inode 3425 δεν είναι σωστό (πιθανώς αντιστοιχεί σε ένα directory και άρα θα έπρεπε να έχει 2+N ref count, όπου N τα “παιδιά” του). Πατάμε **y** για να επιδιορθωθεί το σφάλμα και προχωράμε.

```
Pass 5: Checking group summary information ERROR 3
Block bitmap differences: +34
Fix<y>? 
```

Τώρα, έχουμε σφάλμα στο block bit map (συγκεκριμένα, το block #34 είναι στην πραγματικότητα κατειλημμένο, ενώ το block bit map του bg #0 λέει ότι είναι free – πρόκειται για block που καταλαμβάνεται από το inode table του bg #0). Πατάμε, πάλι, **y** και προχωράμε.

```
Pass 5: Checking group summary information ERROR 4
Free blocks count wrong (925906233, counted=19800).
Fix<y>? 
```

Τώρα, έχουμε σφάλμα στο superblock και συγκεκριμένα στο free block count. Πατάμε πάλι **y**.

```
fsdisk3.img: ***** FILE SYSTEM WAS MODIFIED *****
fsdisk3.img: 23/5136 files (0.0% non-contiguous), 680/20480 blocks
```

**Έτσι, τελικά επιδιορθώσαμε το ΣΑ μας!**

Τώρα, αφού επαναφέρουμε τον δίσκο στην πρότερή του κατάσταση, από την αρχική του καθαρή εικόνα που μας δόθηκε, θα εντοπίσουμε τις αλλοιώσεις που επέδειξε το εργαλείο **fsck**, με χρήση της μεθόδου **hexedit** (**hexdump**).

Ταυτόχρονα, θα επιδιορθώσουμε κάθε αλλοίωση, ξεχωριστά, με χρήση του **hexedit**.

Ξεκινάμε, με το **ERROR 1**. (Έχουμε διαβάσει τα απαραίτητα πεδία του superblock για να κάνουμε τα παρακάτω).

```
root@utopia:~# export BLOCK_SIZE=1024
root@utopia:~# export GDT_OFFSET=$((2*BLOCK_SIZE))
root@utopia:~# export GD_SIZE=32
root@utopia:~# export INODE_SIZE=128
root@utopia:~# hexdump -s$((GDT_OFFSET+GD_SIZE)) -n$((GD_SIZE)) -C /dev/vdd
00000820  03 20 00 00 04 20 00 00  05 20 00 00 1e 1f a8 06  |. .... ....|
00000830  02 00 04 00 00 00 00 00  00 00 00 00 00 00 00 00  |....|
00000840
root@utopia:~# hexdump -s$((8197*BLOCK_SIZE+4*INODE_SIZE)) -n$((INODE_SIZE)) -C /dev/vdd
00801600  ed 41 00 00 00 04 00 00  71 73 9c 63 71 73 9c 63  |.A.....qs.cqs.c|
00801610  71 73 9c 63 00 00 00 00  00 00 02 00 02 00 00 00  |qs.c.....|
00801620  00 00 00 00 04 00 00 00  dc 20 00 00 00 00 00 00  |....|
00801630  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |....|
*
00801660  00 00 00 00 5f 05 b7 b4  00 00 00 00 00 00 00 00  |....|
00801670  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |....|
00801680          name length
root@utopia:~# hexdump -s$((8412*BLOCK_SIZE)) -n$((128)) -C /dev/vdd
00837000  b5 06 00 00 0c 00 03 00  42 4f 4f 00 02 00 00 00  |.....BOO....|
00837010  0c 00 02 00 2e 2e 00 00  b6 06 00 00 10 00 06 00  |....|
00837020  66 69 6c 65 2d 31 00 00  b7 06 00 00 10 00 06 00  |file-1.|
00837030  66 69 6c 65 2d 32 00 00  b8 06 00 00 c8 03 06 00  |file-2.|
00837040  66 69 6c 65 2d 33 00 00  00 00 00 00 00 00 00 00  |file-3.|
00837050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |....|
*
00837080
```

Παρατηρούμε, λοιπόν, ότι αντί για το entry ./ υπάρχει το BOO. Επιδιορθώνουμε το σφάλμα αλλάζοντας το BOO (=0x424f4f) σε ./ (=0x2e), καθώς και το name length από 0x03 σε 0x01 μέσω του **hexedit**:

```
root@utopia:~# hexedit /dev/vdd
```

Βάζουμε την διεύθυνση που βρίσκεται κοντά το σφάλμα:

```
00 00 00 00 00 00 00 00 00 00 00 00
                                         New position ? 0x837000
00 00 00 00 00 00 00 00 00 00 00 00
```

Και κάνουμε τις αλλαγές που θέλουμε:

00836FD0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	changes
00836FE4	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	
00836FF8	00 00 00 00 00 00 00 00 00 00 00 00 B5 06 00 00	0C 00 <u>01</u> 00 2E 00 00 00 B6 06 00 00 10 00 06 00	
0083700C	02 00 00 00 0C 00 02 00 2E 2E 00 00 B6 06 00 00	10 00 06 00	
-** vdd	--0x836EF4/0x14000000--41%-----		

Τρέχουμε το **e2fsck** σε “dry run” για να δούμε αν επιδιορθώθηκε το σφάλμα.

```
root@utopia:~# e2fsck -n /dev/vdd
e2fsck 1.46.2 (28-Feb-2021)
fsdisk3.img contains a file system with errors, check forced.
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Inode 3425 ref count is 1, should be 2. Fix? no
```

**Επιδιορθώθηκε!** Συνεχίζουμε με το επόμενο (ERROR 2):

```
root@utopia:~# hexdump -s$((GDT_OFFSET+2*GD_SIZE)) -n$((GD_SIZE)) -C /dev/vdd
00000840  03 40 00 00 04 40 00 00  05 40 00 00 25 0f ac 06  |..@...@...@..%...|
00000850  01 00 04 00 00 00 00 00  00 00 00 00 00 00 00 00  |.....|
00000860                           ref count
root@utopia:~# hexdump -s$((16389*BLOCK_SIZE)) -n$((INODE_SIZE)) -C /dev/vdd
01001400  ed 41 00 00 00 04 00 00  71 73 9c 63 71 73 9c 63  |.A.....qs.cqs.c|
01001410  71 73 9c 63 00 00 00 00  00 00 01 00 02 00 00 00  |qs.c.....|
01001420  00 00 00 00 04 00 00 00  e8 00 00 00 00 00 00 00 00  |.....|
01001430  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00  |.....|
*
01001460  00 00 00 00 d3 4c 70 ef  00 00 00 00 00 00 00 00 00  |.....Lp.....|
01001470  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00  |.....|
01001480
```

Αλλάζουμε, λοιπόν τα ref count από **0x01** σε **0x02**.

```
root@utopia:~# hexedit /dev/vdd
```

Βάζουμε την διεύθυνση που βρίσκεται κοντά το σφάλμα:

```
0 00 00 00 00 00 00 00 00 00 00 00 ...  
0 00 00 00 00 00 00 00 00 00 00 00 ...  
  
      New position ? 0x1001410  
  
0 00 00 00 00 00 00 00 00 00 00 00 ...  
0 00 00 00 00 00 00 00 00 00 00 00 ...  
0 00 00 00 00 00 00 00 00 00 00 00 ...
```

Και κάνουμε τις αλλαγές που θέλουμε: **changes**

```
010013DC  **** * * * * * * * * * * * *  
010013F0  FF FF FF FF FF FF FF FF F  
01001404  00 04 00 00 71 73 9C 63 71 7  
01001418  00 00 02 00 02 00 00 00 00 00  
-** vdd    --0x100137B/0x1400000--80%
```

Τρέχουμε το **e2fsck**, πάλι, σε “dry run” για να δούμε αν επιδιορθώθηκε το σφάλμα.

```
Pass 4: Checking reference counts
Pass 5: Checking group summary information
Block bitmap differences: +34
Fix? no
Free blocks count wrong for group #0 (7957, counted=7958).
Fix? no
Free blocks count wrong (925906233, counted=19801).
Fix? no
fsdisk3.img: ***** WARNING: Filesystem still has errors *****
fsdisk3.img: 23/5136 files (0.0% non-contiguous), 18446744072783665863/20480 blocks
```

Επιδιορθώθηκε και αυτό! Συνεχίζουμε με το επόμενο (ERROR 3):

```
root@utopia:~# hexdump -s$((GDT_OFFSET)) -n$((GD_SIZE)) -C /dev/vdd
00000800 03 00 00 00 04 00 00 00 05 00 00 00 15 1f a5 06 |.....
00000810 02 00 04 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....
00000820
root@utopia:~# hexdump -s$((3*BLOCK_SIZE)) -n 1024 -C /dev/vdd
00000c00 ff ff ff ff fd ff |.....
00000c10 ff |.....
00000c20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....
*
00000c40 07 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....
00000c50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....
*
00001000
root@utopia:~# xxd -s$((3*BLOCK_SIZE)) -l 1024 -b /dev/vdd
00000c00: 11111111 11111111 11111111 11111111 11111101 11111111 .....  
[The byte at address 0x00000c00 is circled in red]
00000c06: 11111111 11111111 11111111 11111111 11111111 11111111 .....  

00000c0c: 11111111 11111111 11111111 11111111 11111111 11111111 .....  

00000c12: 11111111 11111111 11111111 11111111 11111111 11111111 .....  

00000c18: 11111111 11111111 11111111 11111111 11111111 00000000 .....
```

Αλλάζουμε, λοιπόν, το υπογραμμισμένο bit από **0** σε **1** (ή ισοδύναμα το υπογραμμισμένο byte από **0xFD** σε **0xFF**):

```
root@utopia:~# hexedit /dev/vdd
```

Βάζουμε την διεύθυνση που βρίσκεται κοντά το σφάλμα:

```
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
New position ? 0xc00
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Και κάνουμε τις αλλαγές που θέλουμε:

	changes															
000000B90	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000BA4	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000BB8	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000BCC	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000BE0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000BF4	00	00	00	00	00	00	00	00	00	00	FF	FF	FF	FF	FF	FF
-** vdd	--	0xAED	/0x14000000	--	0%	--	--	--	--	--	--	--	--	--	--	--

Τρέχουμε το **e2fsck**, πάλι, σε “dry run” για να δούμε αν επιδιορθώθηκε το σφάλμα.

```
Pass 5: Checking group summary information
Free blocks count wrong (925906233, counted=19800).
Fix? no
fsdisk3.img: 23/5136 files (0.0% non-contiguous), 18446744072783665863/20480 blocks
```

**Επιδιορθώθηκε και αυτό!** Αξίζει να παρατηρήσουμε ότι προηγουμένως εμφανίστηκε ένα ακόμα σφάλμα, το οποίο δεν υπήρχε όταν πριν τρέξαμε το **e2fsck** για να επιδιορθώσει τα σφάλματα “μόνο του”: **Free blocks count wrong for group #0 (7957, counted=7958)**. Αυτό συμβαίνει, διότι η μη επιδιόρθωση του block bit map οδηγεί και σε λάθος στο free block count στο GD του bg #0 (το free block count έχει την σωστή τιμή, ενώ στο στάδιο αυτό το **fsck** βγάζει error καθώς το συγκρίνει με το block bit map που έχει σφάλμα).

Ας επιδιορθώσουμε και το τελευταίο σφάλμα (**ERROR 4**):

```
root@utopia:~# hexdump -s 1024 -n 1024 -C /dev/vdd
000000400 10 14 00 00 00 50 00 00 00 04 00 00 39 35 30 37 |.....P.....9507|
000000410 f9 13 00 00 01 00 00 00 00 00 00 00 00 00 00 00 |
000000420 00 20 00 00 00 20 00 00 b0 06 00 00 71 73 9c 63 |
000000430 e8 63 c6 63 00 00 ff ff 53 ef 01 00 01 00 00 00 |
000000440 e8 63 c6 63 00 00 00 00 00 00 00 00 01 00 00 00 |
000000450 00 00 00 00 0b 00 00 00 80 00 00 00 00 00 00 00 |
000000460 00 00 00 00 00 00 00 00 82 fb 64 4f d6 66 48 68 |
000000470 94 22 60 e5 7c f3 9e 56 66 73 64 69 73 6b 33 2e |
000000480 69 6d 67 00 00 00 00 00 2f 68 6f 6d 65 2f 6a 69 |
000000490 6d 73 69 61 6b 2f 63 73 6c 61 62 2d 67 69 74 2f |
0000004a0 66 73 2f 6d 6e 74 00 00 00 00 00 00 00 00 00 00 |
0000004b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |
*
0000004e0 00 00 00 00 00 00 00 00 00 00 00 23 e1 8f 01 |.....#...|
0000004f0 6d e1 44 ed 81 23 b1 26 6e 12 bd d7 01 00 00 00 |
000000500 0c 00 00 00 00 00 00 00 71 73 9c 63 00 00 00 00 |
000000510 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |
*
000000560 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |
000000570 00 00 00 00 00 00 00 00 2d 00 00 00 00 00 00 00 |
000000580 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |
*
000000800
```

Αλλάζουμε, λοιπόν, το free blocks count από 925906233 (=0x37303539) σε 19800 (=0x00004d58):

```
root@utopia:~# hexedit /dev/vdd
```

Βάζουμε την διεύθυνση που βρίσκεται κοντά το σφάλμα:

```
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....|
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....|
New position ? 0x400|
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....|
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

Και κάνουμε τις αλλαγές που θέλουμε:

<pre>0000003C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0000003D4 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0000003E8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0000003FC 00 00 00 00 10 14 00 00 00 50 00 00 00 04 00 00 000000410 F9 13 00 00 01 00 00 00 00 00 00 00 00 00 00 00</pre>	<span style="color: green;">changes</span>
<pre>-** vdd --0x334/0x1400000--0%</pre>	

Τρέχουμε το **e2fsck** για να δούμε αν επιδιορθώθηκε και το τελευταίο σφάλμα.

```
root@utopia:~# e2fsck -f -n /dev/vdd
e2fsck 1.46.2 (28-Feb-2021)
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
fsdisk3.img: 23/5136 files (0.0% non-contiguous), 680/20480 blocks
```

**Όλα τα σφάλματα του ΣΑ επιδιορθώθηκαν!**