ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Εργαστήριο Λειτουργικών Συστημάτων

Ακαδημαϊκό έτος 2023-2024

Ομάδα: oslab4

Ονοματεπώνυμα – Αριθμοί Μητρώου:

Κοκολάκης Γεώργιος-03114745

Πάντος Πάντος-03120408

Αναφορά 3ης Εργαστηριακής Άσκησης:

Συστήματα Αρχείων σε Περιβάλλον Linux

1ο Μέρος - Το σύστημα αρχείων ext2

2.3 Η εικόνα fsdisk1.img

1. Τροποποιήστε κατάλληλα το αρχείο utopia.sh ώστε να προσθέσετε στην εικονική μηχανή utopia έναν επιπλέον δίσκο για την εικόνα fsdisk1.img. Ποια είναι η προσθήκη που κάνατε; Ποια συσκευή στο utopia είναι αυτή που μόλις προσθέσατε;

drive format=raw,file=fsdisk1.img,if=virtio

lsblk -f

NAME FSTYPE FSVER LABEL UUID FSAVAIL FSUSE% MOUNTPOINT

fd0

loop0

ext2 1.0 fsdisk1.img

c63028e5-711b-410d-a263-e7ca2b15a8d3 45,9M 0% /mnt

2. Τι μέγεθος έχει ο δίσκος που προσθέσατε στο utopia;

Mount:

lsblk -f

NAME FSTYPE FSVER LABEL UUID FSAVAIL FSUSE% MOUNTPOINT

fd0

loop0

ext2 1.0 fsdisk1.img

c63028e5-711b-410d-a263-e7ca2b15a8d3 **45,9M** 0% /mnt

Hexedit:

Από το hexdump του superblock, παίρνουμε τον αριθμό των block & το blocksize, το γινόμενό τους μα δίνει το μέγεθος του δίσκου:

root@utopia:/home/user/shared# hexdump -s 1024 -n 1024 -C fsdisk1.img

00000400 18 32 00 00 **00 c8 00 00** 00 0a 00 00 90 c1 00 00 |.2..............| (αριθμός μπλοκ)

00000410 0a 32 00 00 01 00 00 00 **00 00 00 00** 00 00 00 00 |.2..............| (μέγεθος μπλοκ, αριθμός ολισθίσεων που πρέπει να γίνουν στο 1024)

1024* 51200(0xc800) = **52.428.800 bytes**

3. Τι σύστημα αρχείων περιέχει;

Mount:

root@utopia:/home/user/shared# file fsdisk1.img

fsdisk1.img: Linux rev 1.0 ext2 filesystem data (mounted or unclean), UUID=c63028e5-711b-410d-a263-e7ca2b15a8d3, volume name "fsdisk1.img"

Hexedit:

root@utopia:/home/user/shared# hexdump -s 1024 -n 1024 -C fsdisk1.img

00000430 1c 41 a1 65 06 00 ff ff **53 ef** 00 00 01 00 00 00

Bytes 56-57 του superblock, έχουν το ext2 signature (0xef53)

4. Πότε ακριβώς δημιουργήθηκε αυτό το σύστημα αρχείων; Δείξτε τη χρονοσφραγίδα [timestamp].

Mount:

root@utopia:/home/user/shared# dumpe2fs fsdisk1.img | grep "created:"

dumpe2fs 1.46.2 (28-Feb-2021)

Filesystem created: **Tue Dec 12 17:23:16 2023**

Hexedit:

Με hexdump του superblock (root@utopia:/home/user/shared# hexdump -s 1024 -n 1024 -v -C fsdisk1.img) ελέγχω τα bytes 265-268, τα οποία περιέχουν σε LE το timestamp e47a7865, που με μετατροπή σε BE και σε Human Read format, μας δίνει την ημερομηνία: Tue Dec 12 17:23:16 2023.
Τα bytes αυτά είναι αχρησιμοποίητα από το ext2 και κάποια προγράμματα τα χρησιμοποιούν για να αποθηκεύσουν πληροφορίες μεταγενέστερων συστημάτων αρχείων(πχ ext4).

(Γνωρίζοντας από τη dumpe2fs την ημερομηνία, μπορούμε να αναζητήσουμε με τον παρακάτω τρόπο:

root@utopia:/home/user/shared# hexdump -s 1024 -n 1024 -v -C fsdisk1.img | grep -b -o -E " e4 7a 78 65")

5. Πότε ακριβώς προσαρτήθηκε τελευταία φορά; Δείξτε τη χρονοσφραγίδα.

Mount:

root@utopia:/home/user/shared# stat /mnt

Access: 2024-01-05 20:31:06.000000000 +0200

root@utopia:/mnt# dumpe2fs -h $(df /mnt --output=source | tail -n 1)

Last mount time: **Fri Jan 5 20:30:58 2024**

Hexedit:

Στο byte 44 του superblock έχουμε το Last mount time

root@utopia:/home/user/shared# hexdump -s 1024 -n 1024 -v -C fsdisk1.img

00000420 00 20 00 00 00 20 00 00 28 07 00 00 **1c 41 a1 65**

Που μεταφράζεται σε **Fri Jan 12 15:39:40 2024**(Το hexdump έγινε άλλη μέρα από το προηγούμενο mount)

6. Σε ποιο μονοπάτι προσαρτήθηκε τελευταία φορά;

Mount:

root@utopia:/mnt# dumpe2fs -h $(df /mnt --output=source | tail -n 1)

dumpe2fs 1.46.2 (28-Feb-2021)

Filesystem volume name: fsdisk1.img

Last mounted on: **/ mnt**

root@utopia:/home/user/shared# df -h /mnt

Filesystem Size Used Avail Use% Mounted on

/dev/loop0 49M 16K 46M 1% /mnt

Hexedit:-

root@utopia:/home/user/shared# hexdump -s 1024 -n 1024 -v -C fsdisk1.img

00000480 69 6d 67 00 00 00 00 00 **2f 6d 6e 74** 00 00 00 00 |img...../mnt....|

7. Πότε ακριβώς τροποποιήθηκε τελευταία φορά; Δείξτε τη χρονοσφραγίδα.

Mount:

root@utopia:/dumpe2fs /dev/vdb

Last write time: **Fri Jan 12 15:39:40 2024**

Hexedit:

Στο byte 48 του superblock έχουμε το Last written time

root@utopia:/home/user/shared# hexdump -s 1024 -n 1024 -v -C fsdisk1.img

00000430 **1c 41 a1 65** 06 00 ff ff 53 ef 00 00 01 00 00 00
Που μεταφράζεται σε **Fri Jan 12 15:39:40 2024**

8. Τι είναι το μπλοκ σε ένα σύστημα αρχείων;

Σε ένα σύστημα αρχείων, το "μπλοκ" είναι η βασική μονάδα αποθήκευσης που χρησιμοποιείται για την οργάνωση των δεδομένων. Ένα αρχείο δεδομένων ή ένα σύστημα αρχείων χρησιμοποιεί τα μπλοκ για να αποθηκεύει πληροφορίες και να διαχειρίζεται τον χώρο στον αποθηκευτικό μέσο.

Κάθε μπλοκ έχει ένα σταθερό μέγεθος και περιέχει μια ποσότητα δεδομένων. Τα μπλοκ συνήθως έχουν μέγεθος σε δυνάμεις του 2 για ευκολία στη διαχείριση. Για παράδειγμα, συχνά βλέπουμε μπλοκ μεγέθους 4KB ή 8KB.

Όταν ένα αρχείο δημιουργείται ή τροποποιείται, τα δεδομένα αυτού του αρχείου αποθηκεύονται σε μπλοκ στον αποθηκευτικό χώρο. Το σύστημα αρχείων χρησιμοποιεί μια δομή δεδομένων που ονομάζεται "file allocation table" (FAT), "inode table", ή κάτι παρόμοιο για να παρακολουθεί ποια μπλοκ ανήκουν σε κάθε αρχείο.

Η χρήση μπλοκ επιτρέπει την αποτελεσματική διαχείριση του χώρου στον δίσκο, καθώς και τη γρήγορη ανάγνωση και εγγραφή δεδομένων.

(Source: Wiki, ChatGPT)

9. Τι μέγεθος μπλοκ [block size] χρησιμοποιεί αυτό το σύστημα αρχείων;

Mount:

root@utopia:/mnt# stat -f -c %s /mnt

1024(bytes)

root@utopia:/mnt# dumpe2fs -h $(df /mnt --output=source | tail -n 1) |

grep "Block size"

dumpe2fs 1.46.2 (28-Feb-2021)

Block size: 1024

Hexedit:

Στο byte 24 του superblock, έχουμε τον αριθμό με τον οποίο πρέπει να ολισθήσουμε το 1024 για να πάρουμε το μέγεθος του block, ο οποίος αριθμός εδώ είναι 0, άρα το block size = 1024

root@utopia:/home/user/shared# hexdump -s 1024 -n 1024 -v -C fsdisk1.img

00000410 0a 32 00 00 01 00 00 00 00 00 00 00 00 00 00 00

10. Τι είναι το inode σε ένα σύστημα αρχείων;

Στα συστήματα αρχείων, το inode (Index Node) είναι ένα δομικό στοιχείο που περιέχει πληροφορίες για ένα αρχείο ή έναν κατάλογο. Κάθε αρχείο ή κατάλογος σε ένα σύστημα αρχείων UNIX ή UNIX-like συστήματος αρχείων αντιστοιχεί σε ένα μοναδικό inode. Τα περιεχόμενα του inode περιλαμβάνουν πληροφορίες όπως:

Αριθμός inode: Ένας μοναδικός αριθμός που αναγνωρίζει μοναδικά το συγκεκριμένο inode.
Τύπος αρχείου (file type): Αναφέρεται στο είδος του αρχείου, είτε πρόκειται για κανονικό αρχείο, κατάλογο, σύνδεσμο, κ.λπ.
Δικαιώματα πρόσβασης (permissions): Οι άδειες πρόσβασης που ισχύουν για το αρχείο.
Αριθμός συνδέσεων (link count): Ο αριθμός των συνδέσεων προς το inode. Για ένα κανονικό αρχείο, αυτό είναι συνήθως 1, ενώ για έναν κατάλογο αρχείων αυξάνεται κάθε φορά που προστίθεται ένα νέο υπο-αρχείο.
Ιδιοκτήτης (owner): Ο χρήστης που έχει δικαιώματα πρόσβασης στο αρχείο.
Ομάδα (group): Η ομάδα χρηστών που έχει δικαιώματα πρόσβασης στο αρχείο.
Μέγεθος αρχείου: Το μέγεθος του περιεχομένου του αρχείου σε bytes.
Χρόνοι δημιουργίας, τελευταίας πρόσβασης και τελευταίας τροποποίησης: Πληροφορίες σχετικά με τους χρόνους που σχετίζονται με το αρχείο.
Το inode παρέχει τη δομημένη απεικόνιση των μεταδεδομένων του αρχείου και επιτρέπει την αποτελεσματική διαχείριση των αρχείων στο σύστημα αρχείων.
(Source: Wiki, ChatGPT)

11. Τι μέγεθος έχει το inode σε αυτό το σύστημα αρχείων;

Mount:
root@utopia:/home/user/shared# dumpe2fs fsdisk1.img | grep "Inode size"

dumpe2fs 1.46.2 (28-Feb-2021)

Inode size: 128

Hexedit:

Η τιμή του μεγέθους των inode αποθηκεύεται στα byte 58-5B του superblock:

root@utopia:/home/user/shared# hexdump -s 1024 -n 1024 -v -C fsdisk1.img

00000450 00 00 00 00 0b 00 00 00 **80 00 00 00** 00 00 00 00

LE 80 00 00 00 => 128 decimal

12. Πόσα διαθέσιμα μπλοκ και πόσα διαθέσιμα inodes υπάρχουν σε αυτό το σύστημα αρχείων;

Mount:

root@utopia:/home/user/shared# dumpe2fs ./fsdisk1.img | grep 'Free'

Free blocks: 49552

Free inodes: 12810

Hexedit:

root@utopia:/home/user/shared# hexdump -s 1024 -n 1024 -C fsdisk1.img

00000400 18 32 00 00 00 c8 00 00 00 0a 00 00 **90 c1 00 00**
00000410 **0a 32 00 00** 01 00 00 00 00 00 00 00 00 00 00 00

Superblock bytes 12-15: Διαθέσιμα μπλοκ
Superblock bytes 16-19: Διαθέσιμα inodes

13. Τι είναι το superblock στο σύστημα αρχείων ext2;

Το superblock σε ένα σύστημα αρχείων ext2 είναι μια σημαντική δομή που περιέχει κρίσιμες πληροφορίες για το σύστημα αρχείων. Το superblock περιγράφει τα χαρακτηριστικά του συστήματος αρχείων, όπως το μέγεθος των μπλοκ, το μέγεθος των inodes, τον αριθμό των ελεύθερων μπλοκ και inodes, και άλλες σημαντικές παραμέτρους.
Το superblock βρίσκεται στην αρχή του συστήματος αρχείων και είναι αναγκαίο για τη σωστή λειτουργία του. Εάν το superblock καταστραφεί ή υποστεί κάποια βλάβη, το σύστημα αρχείων μπορεί να μην εκκινήσει ή να λειτουργήσει σωστά.
Συνήθως, χρησιμοποιούνται πολλαπλά αντίγραφα του superblock για ασφάλεια, και τα αντίγραφα αυτά αποθηκεύονται σε διάφορα μέρη του συστήματος αρχείων, ώστε να είναι δυνατή η ανάκτηση σε περίπτωση προβλημάτων.
(Source: Wiki, ChatGPT)

14. Πού βρίσκεται μέσα στον δίσκο σε ένα σύστημα αρχείων ext2;

Το superblock βρίσκεται πάντα στο byte 1024 από την αρχή του volume και έχει μήκος 1024 bytes, αλλά υπάρχουν και αντίγραφά του, αποθηκευμένα σε άλλα σημεία στο δίσκο.
(Source: Wiki, ChatGPT)

15. Για ποιο λόγο έχει νόημα να υπάρχουν εφεδρικά αντίγραφα του superblock στο σύστημα αρχείων ext2;

Τα εφεδρικά αντίγραφα του superblock στο σύστημα αρχείων ext2 είναι σημαντικά για διάφορους λόγους:

Ανάκτηση Δεδομένων: Σε περίπτωση που το κύριο superblock υποστεί ζημιά ή καταστραφεί (για παράδειγμα, λόγω απρόβλεπτης αποσύνδεσης του δίσκου), τα εφεδρικά αντίγραφα μπορούν να χρησιμοποιηθούν για την ανάκτηση των δεδομένων και την επαναφορά του συστήματος αρχείων.

Ευκολία Συντήρησης: Η ύπαρξη πολλαπλών αντιγράφων του superblock επιτρέπει τον εντοπισμό και την ανάκτηση του superblock ακόμη και αν ένα από τα αντίγραφα έχει υποστεί ζημιά. Αυτό καθιστά ευκολότερη τη συντήρηση του συστήματος αρχείων.

Αποκατάσταση μετά από Σοβαρά Προβλήματα: Σε περίπτωση που το κύριο superblock και όλα τα εφεδρικά αντίγραφα υποστούν ζημιά, μπορεί ακόμη να χρησιμοποιηθεί ένα ειδικό εφεδρικό superblock που βρίσκεται σε μια γνωστή τοποθεσία στον δίσκο.

Συνολικά, η πολυπληθής αποθήκευση αντιγράφων του superblock αποτελεί σημαντικό μέσο για την αποφυγή απώλειας δεδομένων και την εξασφάλιση της ακεραιότητας του συστήματος αρχείων.
(Source: Wiki, ChatGPT)

16. Σε ποια μπλοκ βρίσκονται αποθηκευμένα εφεδρικά αντίγραφα του superblock σε αυτό το σύστημα αρχείων;

Mount:
root@utopia:/home/user/shared# dumpe2fs /dev/vdb

Group 1: (Blocks 8193-16384)

Backup superblock at 8193

Group 2: (Blocks 16385-24576)

Backup superblock at 16385

Group 3: (Blocks 24577-32768)

Backup superblock at 24577

Group 4: (Blocks 32769-40960)

Backup superblock at 32769

Group 5: (Blocks 40961-49152)

Backup superblock at 40961

Group 6: (Blocks 49153-51199)

Backup superblock at 49153

Hexedit:
Από τα bytes 32-35 του superblock, αντλούμε το αριθμό των blocks ανά block group(8192),
root@utopia:/home/user/shared# hexdump -s 1024 -n 1024 -C fsdisk1.img
00000420 **00 20 00 00** 00 20 00 00 28 07 00 00 1c 41 a1 65
Μετά γνωρίζοντας το block size(1024) από προηγούμενο ερώτημα, έχουμε το block group size = 8192*1024=8.388.608, με την παρακάτω εντολή παίρνουμε ως αποτέλεσμα ένα αντίγραφο του superblock:
root@utopia:/home/user/shared# hexdump -s 8389632 -n 1024 -C fsdisk1.img
Μετά προχωρώντας ανά block group size bytes βρίσκουμε και τα υπόλοιπα αντίγραφα. Αυτή τη διαδικασία μπορούμε να την κάνουμε τόσες φορές όσα και τα block groups, των οποίων το πλήθος το βρίσκουμε διαιρώντας τον αριθμό των συνολικών block του συστήματος(superblock bytes 4-7) με το πλήθος των block ανά block group, 51200/8192= 6.25, συνεπώς το σύστημα θα έχει 7 block groups.

17. Τι είναι ένα block group στο σύστημα αρχείων ext2;

Στο σύστημα αρχείων ext2, ένα block group είναι ένα σύνολο συνεχόμενων μπλοκ που περιέχουν δεδομένα, inodes και άλλες δομές διαχείρισης. Το ext2 χωρίζει τον σκληρό δίσκο σε λογικές ομάδες μπλοκ (block groups), και κάθε ομάδα λειτουργεί σαν ανεξάρτητη μικρή δομή αρχείων στο εσωτερικό του συνολικού συστήματος αρχείων.

Κάθε block group περιέχει:

Superblock Backup: Κάθε block group έχει ένα αντίγραφο του κύριου superblock που χρησιμοποιείται για την ανάκτηση σε περίπτωση προβλημάτων με τον κύριο superblock.

Block Bitmap: Ένας χάρτης (bitmap) που δείχνει ποια μπλοκ είναι κατειλημμένα και ποια είναι ελεύθερα.

Inode Bitmap: Ένας χάρτης που δείχνει ποια inodes είναι κατειλημμένα και ποια είναι ελεύθερα.

Inode Table: Πίνακας που περιέχει τα inodes για τα αρχεία που ανήκουν στο συγκεκριμένο block group.

(Source: Wiki, ChatGPT)

18. Πόσα block groups έχει ένα σύστημα αρχείων ext2 και πώς κατανέμονται;

Ο αριθμός των block groups σε ένα σύστημα αρχείων ext2 υπολογίζεται με βάση τον αριθμό των συνολικών μπλοκ στο σύστημα αρχείων και τον αριθμό των μπλοκ ανά block group. Κάθε block group έχει τον δικό του superblock, block bitmap, inode bitmap, και inode table.

Η διαδικασία υπολογισμού του αριθμού των block groups είναι η εξής:

Υπολόγισε τον συνολικό αριθμό των μπλοκ στο σύστημα αρχείων (Block Count).

Καθορίσε τον αριθμό των μπλοκ ανά block group (Blocks per Group).

Χρησιμοποίησε τον τύπο: Block Groups = Block Count / Blocks per Group

Για παράδειγμα, αν έχουμε ένα σύστημα αρχείων με συνολικό αριθμό μπλοκ 51200 και 8192 μπλοκ ανά block group, τότε:

Block Groups = 51200 / 8192 = 6.25

Στην πραγματικότητα, το αποτέλεσμα θα πρέπει να είναι ένας ακέραιος αριθμός. Συνεπώς, θα έχουμε 7 block groups. Εάν υπάρχει υπόλοιπο, τότε το τελευταίο block group θα έχει λιγότερα μπλοκ από τα υπόλοιπα.

(Source: Wiki, ChatGPT)

19. Πόσα block groups περιέχει αυτό το σύστημα αρχείων;

Mount:

root@utopia:/home/user/shared# dumpe2fs /dev/vdb | grep 'Group '

dumpe2fs 1.46.2 (28-Feb-2021)

Group 0: (Blocks 1-8192)

  Primary superblock at 1, Group descriptors at 2-2

Group 1: (Blocks 8193-16384)

  Backup superblock at 8193, Group descriptors at 8194-8194

Group 2: (Blocks 16385-24576)

Backup superblock at 16385, Group descriptors at 16386-16386

Group 3: (Blocks 24577-32768)

Backup superblock at 24577, Group descriptors at 24578-24578

Group 4: (Blocks 32769-40960)

Backup superblock at 32769, Group descriptors at 32770-32770

Group 5: (Blocks 40961-49152)

Backup superblock at 40961, Group descriptors at 40962-40962

Group 6: (Blocks 49153-51199)

Backup superblock at 49153, Group descriptors at 49154-49154

Περιέχει 7 block groups, 0-6.

Hexedit: (Όπως απαντήθηκε για την ερώτηση 16 & 18)
Από τα bytes 32-35 του superblock, αντλούμε το αριθμό των blocks ανά block group(8192), έπειτα το πλήθος τους το βρίσκουμε διαιρώντας τον αριθμό των συνολικών block του συστήματος(superblock bytes 4-7) με το πλήθος των block ανά block group, 51200/8192= 6.25, συνεπώς το σύστημα θα έχει 7 block groups.

20. Τι είναι ο block group descriptor στο σύστημα αρχείων ext2;

Ο block group descriptor στο σύστημα αρχείων ext2 περιέχει πληροφορίες για κάθε block group στο σύστημα αρχείων. Περιλαμβάνει τα εξής στοιχεία για κάθε block group:

Αριθμός Block Bitmap: Διεύθυνση (block number) του bitmap που παρουσιάζει την κατάσταση των μπλοκ στο block group.

Αριθμός Inode Bitmap: Διεύθυνση του bitmap που παρουσιάζει την κατάσταση των inodes στο block group.

Αριθμός Inode Table: Διεύθυνση του πίνακα (table) που περιέχει τις πληροφορίες των inodes στο block group.

Αριθμός ελεύθερων μπλοκ: Το πλήθος των ελεύθερων μπλοκ στο block group.

Αριθμός ελεύθερων Inodes: Το πλήθος των ελεύθερων inodes στο block group.

Αριθμός φακέλων group: Το πλήθος των φακέλων μέσα στο group

Αριθμός ελεύθερων μπλοκ άνω του 32-bit word: Σε συστήματα αρχείων που χρησιμοποιούν 32-bit λέξεις, αυτός ο αριθμός παρέχει πρόσθετες πληροφορίες για τα ελεύθερα μπλοκ.

Οι πληροφορίες αυτές είναι σημαντικές για τη διαχείριση του χώρου στο σύστημα αρχείων και την εντοπισμό των αντίστοιχων δομών δεδομένων, όπως τα bitmaps και ο πίνακας των inodes.

Ο πίνακας των block group descriptor, αποθηκεύεται στο αμέσως επόμενο block από το superblock.
(Source: Wiki, ChatGPT)

21. Για ποιο λόγο έχει νόημα να υπάρχουν εφεδρικά αντίγραφα των block group descriptors στο σύστημα αρχείων ext2;

Σε αναλογία με τα superblocks, Τα εφεδρικά αντίγραφα των block group descriptors στο σύστημα αρχείων ext2 είναι σημαντικά για την ασφάλεια και την ανάκτηση δεδομένων σε περίπτωση προβλημάτων. Κάθε block group στο ext2 έχει το δικό του block group descriptor που περιλαμβάνει πληροφορίες σχετικά με την οργάνωση και την κατάσταση του εκάστοτε block group.

Οι λόγοι για τα εφεδρικά αντίγραφα είναι οι εξής:

Ανάκτηση Δεδομένων: Σε περίπτωση που ο ένας block group descriptor καταστραφεί λόγω σφάλματος στον δίσκο, τα εφεδρικά αντίγραφα επιτρέπουν την ανάκτηση πληροφοριών για τον συγκεκριμένο block group.

Ανοχή Σφαλμάτων: Η παρουσία εφεδρικών αντιγράφων επιτρέπει την ανοχή σε σφάλματα στον τομέα των block group descriptors, καθώς μπορεί να χρησιμοποιηθεί ένα εφεδρικό αντίγραφο για την αντικατάσταση του κατεστραμμένου.

Αυξημένη Αξιοπιστία: Η παρουσία εφεδρικών αντιγράφων συνεισφέρει στην αυξημένη αξιοπιστία του συστήματος αρχείων, καθώς δίνει τη δυνατότητα για αποκατάσταση και συντήρηση σε περίπτωση ζημιάς.

Συνολικά, τα εφεδρικά αντίγραφα ενισχύουν την ανθεκτικότητα του συστήματος αρχείων ext2 και βοηθούν στη διασφάλιση της ακεραιότητας των δεδομένων, παρέχοντας μηχανισμούς αντιμετώπισης προβλημάτων.
(Source: Wiki, ChatGPT)

22. Σε ποια μπλοκ βρίσκονται αποθηκευμένα εφεδρικά αντίγραφα των block group descriptors σε αυτό το σύστημα αρχείων;

Mount:

root@utopia:/home/user/shared# dumpe2fs /dev/vdb | grep "Group descriptors"

dumpe2fs 1.46.2 (28-Feb-2021)

Group descriptors at 8194-8194

Group descriptors at 16386-16386

Group descriptors at 24578-24578

Group descriptors at 32770-32770

Group descriptors at 40962-40962

Group descriptors at 49154-49154

Hexedit:

Αντλώντας το block group size και το πλήθος των group, όπως κάναμε νωρίτερα και ξεκινώντας από το block 2(αμέσως επόμενο από το superblock), όπου είναι αποθηκευμένος πρώτη φορά ο πίνακας των block group descriptor, μπορούμε να προχωρήσουμε ανά block group size bytes άλλες 6 φορές(όσα και τα υπόλοιπα group) για να οδηγηθούμε στα υπόλοιπα αντίγραφα των descriptor.

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

Το "block bitmap" και το "inode bitmap" είναι δομές που χρησιμοποιούνται σε συστήματα αρχείων, όπως το ext2, για τη διαχείριση των blocks και των inodes αντίστοιχα. Ας εξηγήσουμε κάθε ένα ξεχωριστά:

Block Bitmap:

Το "block bitmap" καταγράφει ποια από τα blocks του δίσκου χρησιμοποιούνται και ποια είναι διαθέσιμα. Κάθε bit του "block bitmap" αντιστοιχεί σε ένα block, και ένα bit 0 σημαίνει ότι το αντίστοιχο block είναι διαθέσιμο, ενώ ένα bit 1 σημαίνει ότι το block είναι σε χρήση. Βρίσκεται σε συγκεκριμένο block στον δίσκο.

Inode Bitmap:

Το "inode bitmap" καταγράφει ποια από τα inodes του συστήματος αρχείων είναι σε χρήση και ποια είναι διαθέσιμα. Κάθε bit του "inode bitmap" αντιστοιχεί σε ένα inode, και ένα bit 0 σημαίνει ότι το αντίστοιχο inode είναι διαθέσιμο, ενώ ένα bit 1 σημαίνει ότι το inode είναι σε χρήση. Βρίσκεται σε συγκεκριμένο block στον δίσκο.

Και τα δύο (block bitmap και inode bitmap) αποθηκεύονται σε συγκεκριμένα blocks, και η θέση τους καθορίζεται από το "block group descriptor" για το αντίστοιχο block group.

(Source: Wiki, ChatGPT)

24. Τι είναι τα inode tables; Πού βρίσκονται μέσα στον δίσκο;

Το "inode table" είναι μια δομή που περιέχει όλα τα inodes ενός συστήματος αρχείων. Κάθε αρχείο στο σύστημα αρχείων, καθώς και κάθε κατάλογος (directory), αντιστοιχεί σε ένα inode. Τα inodes περιέχουν πληροφορίες για τα αρχεία, όπως τους τύπους των αρχείων, τους ιδιοκτήτες, τις άδειες πρόσβασης, τις χρονοσφραγίδες (timestamps) και άλλες σχετικές πληροφορίες.

Το "inode table" είναι απλά μια σειρά από inodes, και ανάλογα με το σύστημα αρχείων, μπορεί να βρίσκεται σε συγκεκριμένα μπλοκ στον δίσκο.

Στο σύστημα αρχείων ext2 που εξετάζουμε, τα "inode tables" βρίσκονται στα block groups. Συγκεκριμένα, στην έξοδο της εντολής `dumpe2fs`, μπορούμε να δούμε πού βρίσκεται το "inode table" για κάθε block group ή εναλλατικά αναζητώντας απευθείας τα bytes του block group descriptor.

(Source: Wiki, ChatGPT)

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

Κάθε inode περιέχει πληροφορίες για ένα αρχείο ή έναν κατάλογο στο σύστημα αρχείων. Τα πεδία ενός inode περιλαμβάνουν συνήθως τα εξής:

1. Τύπος Αρχείου (File Type): Δηλώνει εάν το inode αντιστοιχεί σε αρχείο, κατάλογο ή άλλο τύπο αρχείου.

2. Άδειες Πρόσβασης (Permissions): Καθορίζουν τα δικαιώματα πρόσβασης των χρηστών στο αρχείο.

3. Ιδιοκτήτης (Owner): Ο αριθμός του χρήστη που είναι ο ιδιοκτήτης του αρχείου.

4. Ομάδα (Group): Ο αριθμός της ομάδας που έχει πρόσβαση στο αρχείο.

5. Μέγεθος (Size): Το μέγεθος του αρχείου σε bytes.

6. Χρονοσφραγίδες (Timestamps): Συνήθως περιλαμβάνουν τρεις χρονοσφραγίδες που αντιστοιχούν στη δημιουργία (creation), τελευταία τροποποίηση (modification), και τελευταία πρόσβαση (access) του αρχείου.

7. Δείκτες (Pointers): Οι δείκτες προς τα μπλοκ που περιέχουν τα δεδομένα του αρχείου. Στα μεγάλα αρχεία, χρησιμοποιούνται περισσότεροι από έναν δείκτες για την αποθήκευση των δεδομένων.

Το inode ίδιο αποθηκεύεται σε ένα προκαθορισμένο μέρος του δίσκου. Η τοποθεσία αυτή εξαρτάται από το συγκεκριμένο σύστημα αρχείων. Στο σύστημα αρχείων ext2, τα inodes αποθηκεύονται σε ειδικούς πίνακες που βρίσκονται στα block groups, στον ίδιο φάκελο όπου βρίσκονται και τα "inode tables".

(Source: Wiki, ChatGPT)

26. Πόσα μπλοκ και πόσα inodes περιέχει το κάθε block group σε αυτό το σύστημα αρχείων;

Mount:

root@utopia:/home/user/shared# dumpe2fs /dev/vdb

Blocks per group: 8192

Inodes per group: 1832

Hexedit:

Superblock bytes: 32-25 Αριθμός blocks σε κάθε block group
Superblock bytes: 40-43 Αριθμός inode σε κάθε block group

root@utopia:/home/user/shared# hexdump -s 1024 -n 1024 -C fsdisk1.img

00000420 **00 20 00 00** 00 20 00 00 **28 07 00 00** 1c 41 a1 65

27. Σε ποιο inode αντιστοιχεί το αρχείο /dir2/helloworld σε αυτό το σύστημα αρχείων;

Mount:

root@utopia:/home/user/shared# stat /mnt/dir2/helloworld | grep "Inode"

Device: 700h/1792d Inode: 9162 Links: 1

Hexedit:

Ξεκινάμε από το inode του root directory, το οποίο είναι πάντα το inode#2 και βρίσκεται το Block Group #0. Βρίσκουμε σε ποιο block είναι το inode table του Block Group #0, bytes 8-11

root@utopia:/home/user/shared# hexdump -s 2048 -n 32 -C /dev/vdb

00000800 03 00 00 00 04 00 00 00 **05 00 00 00** 08 1f 1d 07 |................|

00000810 02 00 04 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

root@utopia:/home/user/shared# export bsz=1024

root@utopia:/home/user/shared# hexdump -s$((5*bsz+128)) -n 128 -C /dev/vdb

00001480 ed 41 00 00 00 04 00 00 b5 6b a2 65 e4 7a 78 65 |.A.....k.e.zxe|

00001490 e4 7a 78 65 00 00 00 00 00 00 05 00 02 00 00 00 |.zxe............|

000014a0 00 00 00 00 02 00 00 00 **ea 00 00 00** 00 00 00 00 |................|

000014b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

Στο byte #40 του inode βρίσκεται το Direct Block Pointer 0

Βρήκαμε το directory entry block του “/” block #234 (0x00EA)

root@utopia:/home/user/shared# hexdump -s$((234*bsz)) -n 128 -C fsdisk1.img

0003a800 02 00 00 00 0c 00 01 00 2e 00 00 00 02 00 00 00 |................|

0003a810 0c 00 02 00 2e 2e 00 00 0b 00 00 00 14 00 0a 00 |................|

0003a820 6c 6f 73 74 2b 66 6f 75 6e 64 00 00 29 07 00 00 |lost+found..)...|

0003a830 0c 00 04 00 64 69 72 31 c9 23 00 00 **c8 03 04 00** |....dir1.#......|

0003a840 **64 69 72 32** 00 00 00 00 00 00 00 00 00 00 00 00 |dir2............|

0003a850 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

Διαβάζουμε το directory entry block #234 Ψάχνουμε μέσα σε αυτό το όνομα dir2

Inode#: 9161
Size: 968
len: 4

type: 0 (unknown)
name: dir2

Inodes per block group:1832, 9161/1832 = 5, >5 => ανήκει στο BG#5 (δηλαδή το 6ο)

Είναι το 1ο inode του BG#5

Διαβάζουμε από το BGD του BG #5 το block του inode table

root@utopia:/home/user/shared# hexdump -s$((gdt_ofs+5*gd_sz)) -n $((gd_sz)) -C fsdisk1.img

000008a0 03 a0 00 00 04 a0 00 00 **05 a0 00 00** 17 1f 26 07 |..............&.|

000008b0 01 00 04 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

Inode table του BG #5 στο block #40965 (0xa005)

Διαβάζουμε το 1ο inode του inode table και βρίσκουμε το block του directory entry για το /dir2

root@utopia:/home/user/shared# export inode_sz=128

root@utopia:/home/user/shared# hexdump -s$((40965*bsz)) -n $((inode_sz)) -C fsdisk1.img

02801400 ed 41 00 00 00 04 00 00 b8 6b a2 65 e4 7a 78 65 |.A.......k.e.zxe|

02801410 e4 7a 78 65 00 00 00 00 00 00 02 00 02 00 00 00 |.zxe............|

02801420 00 00 00 00 02 00 00 00 **f7 00 00 00** 00 00 00 00 |................|

02801430 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

*

02801460 00 00 00 00 c5 b2 72 92 00 00 00 00 00 00 00 00 |......r.........|

02801470 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

Είναι το block #247 (0x00f7). Διαβάζουμε το directory entry block #247:

root@utopia:/home/user/shared# hexdump -s$((247*bsz)) -n $((inode_sz)) -C fsdisk1.img

0003dc00 c9 23 00 00 0c 00 01 00 2e 00 00 00 02 00 00 00 |.#..............|

0003dc10 0c 00 02 00 2e 2e 00 00 **ca 23 00 00 e8 03 0a 00** |.........#......|

0003dc20 **68 65 6c 6c 6f 77 6f 72 6c 64** 00 00 00 00 00 00 |helloworld......|

0003dc30 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

Ψάχνουμε μέσα σε αυτό το όνομα helloworld

Inode#: **9162** (0x23ca)
size: 1000 (0x03e8)
len: 10

type: 0 (unknown)
name: helloworld (68 65 6c 6c 6f 77 6f 72 6c 64)

Έτσι βρήκαμε ποιο inode αντιστοιχεί στο /dir2/helloword

28. Σε ποιο block group αντιστοιχεί αυτό το inode;

Mount:

Μπορούμε να υπολογίσουμε σε ποιο block group αντιστοιχεί το συγκεκριμένο inode χρησιμοποιώντας την πληροφορία που παρέχεται από την εντολή `dumpe2fs`. Συγκεκριμένα, το inode 9162 θα ανήκει στο Group 5, καθώς οι αριθμοί inode που αντιστοιχούν στο Group 5 κυμαίνονται από 9163 έως 10992.

Στο αποτέλεσμα της εντολής `dumpe2fs`, οι πληροφορίες για το Group 5 είναι οι εξής:

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)

7959 free blocks, 1830 free inodes, 1 directories

Free blocks: 41194-49152

Free inodes: 9163-10992

Επομένως, το inode 9162 ανήκει στο Group 5.

Hexedit:

Ακολουθώντας τη διαδικασία του προηγούμενου ερωτήματος έχουμε ήδη βρει πως ανήκει στο Block Group#5

29. Σε ποιο μπλοκ του δίσκου υπάρχει το inode table που περιέχει το παραπάνω inode;

Mount:

Το inode 9162 ανήκει στο Group 5, και σύμφωνα με τις πληροφορίες από την εντολή `dumpe2fs`, το inode table για το Group 5 βρίσκεται στα μπλοκ 40965-41193.

Άρα, το inode table που περιλαμβάνει το inode 9162 αρχίζει από το μπλοκ 40965.

Hexedit:

Ομοίως με την προηγούμενη ερώτηση, η διαδικασία για την εύρεση του inode table block, περιγράφεται αναλυτικά στη Hexedit μέθοδο του ερωτήματος 28

30. Δείξτε όλα τα πεδία αυτού του inode [128 bytes]

Mount:

debugfs -R "stat <9162>" /dev/vdb

Inode: 9162   Type: regular   Mode: 0644   Flags: 0x0

Generation: 2739270588   Version: 0x00000001

User: 0   Group: 0   Size: 42

File ACL: 0

Links: 1   Blockcount: 2

Fragment: Address: 0   Number: 0   Size: 0

ctime: 0x65787ae4 -- Tue Dec 12 17:23:16 2023

atime: 0x659956ee -- Sat Jan 6 15:34:38 2024

mtime: 0x65787ae4 -- Tue Dec 12 17:23:16 2023

BLOCKS:

(0):1025

TOTAL: 1

Hexedit:

Το inode #9162   είναι το 2ο inode του BG #5, του οποίου το inode table ξεκινά στο μπλοκ #40965. Οπότε διαβάζουμε το 2^{ο} inode του πίνακα

root@utopia:/home/user/shared# hexdump -s$((40965*bsz+inode_sz)) -n $((inode_sz))

 -C fsdisk1.img

02801480 a4 81 00 00 2a 00 00 00 ba 6b a2 65 e4 7a 78 65 |....*....k.e.zxe|

02801490 e4 7a 78 65 00 00 00 00 00 00 01 00 02 00 00 00 |.zxe............|

028014a0 00 00 00 00 01 00 00 00 01 04 00 00 00 00 00 00 |................|

028014b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

*

028014e0 00 00 00 00 bc f3 45 a3 00 00 00 00 00 00 00 00 |......E.........|

028014f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

Starting Byte	Ending Byte	Size in Bytes	Field Description
0	1	2	Type and Permissions 0x81a4
2	3	2	User ID 0x0
4	7	4	Lower 32 bits of size in bytes **0x2a**
8	11	4	Last Access Time (in POSIX time) **0x65a26bba**
12	15	4	Creation Time (in POSIX time) **0x65787ae4**
16	19	4	Last Modification time (in POSIX time) **0x65787ae4**
20	23	4	Deletion time (in POSIX time) **0x0**
24	25	2	Group ID **0x0**
26	27	2	Count of hard links (directory entries) to this inode. When this reaches 0, the data blocks are marked as unallocated. **0x1**
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. **0x2**
32	35	4	Flags **0x0**
36	39	4	Operating System Specific value #1 **0x1**
40	43	4	Direct Block Pointer 0 **0x0401**
100	103	4	Generation number (Primarily used for NFS) **0xa345f3bc**

31. Σε ποιο μπλοκ είναι αποθηκευμένα τα δεδομένα αυτού του αρχείου;

Mount:

Από τις πληροφορίες της προηγούμενης εντολής (debugfs -R "stat <9162>" /dev/vdb

), τα δεδομένα του αρχείου βρίσκονται στο μπλοκ 1025

Hexedit:

Το inode #9162 είναι το 2ο inode του BG #5, του οποίου το inode table ξεκινά στο μπλοκ #40965. Οπότε διαβάζουμε το 2ο inode του πίνακα και βρίσκουμε το 1ο μπλοκ του αρχείου.

root@utopia:/home/user/shared# hexdump -s$((40965*bsz+inode_sz)) -n $((inode_sz))

-C fsdisk1.img

02801480 a4 81 00 00 2a 00 00 00 ba 6b a2 65 e4 7a 78 65 |....*....k.e.zxe|

02801490 e4 7a 78 65 00 00 00 00 00 00 01 00 02 00 00 00 |.zxe............|

028014a0 00 00 00 00 01 00 00 00 **01 04 00 00** 00 00 00 00 |................|

028014b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

*

028014e0 00 00 00 00 bc f3 45 a3 00 00 00 00 00 00 00 00 |......E.........|

028014f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

Είναι το **block #1025 (0x0401)**

32. Τι μέγεθος έχει αυτό το αρχείο;

Mount:

Το αρχείο που αντιστοιχεί στο inode 9162 έχει μέγεθος 42 bytes, όπως αναφέρεται στις πληροφορίες του inode από την εντολή: debugfs -R "stat <9162>" /dev/vdb.

Hexedit:

Από το hexdump της ερώτησης 30, Lower 32 bits of size in bytes 0x2a = 42

33. Δείξτε τα περιεχόμενα αυτού του αρχείου.

Mount:

root@utopia:/home/user/shared# cat /mnt/dir2/helloworld

Welcome to the Mighty World of Filesystems

Hexedit:

Διαβάζουμε το block #1025

root@utopia:/home/user/shared# hexdump -s$((1025*bsz)) -n $((inode_sz)) -C fsdisk

1.img

00100400 57 65 6c 63 6f 6d 65 20 74 6f 20 74 68 65 20 4d |Welcome to the M|

00100410 69 67 68 74 79 20 57 6f 72 6c 64 20 6f 66 20 46 |ighty World of F|

00100420 69 6c 65 73 79 73 74 65 6d 73 00 00 00 00 00 00 |ilesystems......|

00100430 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

*

2.4 Η εικόνα fsdisk2.img

1. Συνδέστε την εικόνα του δίσκου στην εικονική μηχανή σας, όπως κάνατε και για την εικόνα fsdisk1.img και προσαρτήστε τη στον κατάλογο /mnt.

root@utopia:/home/user/shared# mount -o loop fsdisk2.img /mnt

2. Χρησιμοποιήστε την εντολή touch για να δημιουργήσετε ένα νέο κενό αρχείο /file1 μέσα στο συγκεκριμένο σύστημα αρχείων. Βεβαιωθείτε ότι η εντολή σας αναφέρεται πράγματι στο συγκεκριμένο σύστημα αρχείων [σε ποιον κατάλογο το έχετε προσαρτήσει;], κι όχι στον ριζικό κατάλογο του συστήματος.

root@utopia:/home/user/shared# touch /mnt/file1

3. Πέτυχε η εντολή; Αν όχι, τι πρόβλημα υπήρξε;

touch: cannot touch '/mnt/file1': No space left on device

4. Ποια κλήση συστήματος προσπάθησε να τρέξει η touch, και με ποιον κωδικό λάθους απέτυχε; Αποδείξτε την απάντησή σας με χρήση της εντολής strace.

openat(AT_FDCWD, "/mnt/file1", O_WRONLY|O_CREAT|O_NOCTTY|O_NONBLOCK, 0666) = -1 ENOSPC (No space left on device)

5. Πόσα αρχεία και πόσους καταλόγους περιέχει το συγκεκριμένο σύστημα αρχείων;

Mount:

root@utopia:/home/user/shared# find /mnt -type f | wc -l

4868

root@utopia:/home/user/shared# find /mnt -type d | wc -l

259

Hexedit:

root@utopia:/home/user/shared# hexdump -s 1024 -n 1024 -C fsdisk2.img

00000400 10 14 00 00 00 50 00 00 00 00 00 00 63 4c 00 00

Τα πρώτα 4 bytes μας δίνουν τον αριθμό των inodes στο σύστημα, 5136(0x1410)

Στο επόμενο block από το superblock έχουμε τους Block Group Descriptor, οι οποίοι μας δίνουν τον αριθμό των inodes και των αριθμό των φακέλων, οπότε αθροίζοντας τον αριθμό των φακέλων όλων των group και μετά αφαιρώντας τα από τον αριθμό των inodes μαζί με τον αριθμό 9(για τα system reserved & unused inodes) παίρνουμε και τον αριθμό των αρχείων 4868

6. Πόσο χώρο καταλαμβάνουν τα δεδομένα και τα μεταδεδομένα του συγκεκριμένου συστήματος αρχείων;

Mount:

du -h --max-depth=0 /mnt

270K /mnt

Hexedit:

Αντλώντας από το hexdump του superblock το blocksize, total unallocated blocks & το total block count, έχουμε 1024(blocksize) * [20438- 19555 (block count - unallocated)] = 904.192 bytes

***

7. Πόσο είναι το μέγεθος του συγκεκριμένου συστήματος αρχείων;

Mount:

root@utopia:/home/user/shared# df -h /mnt

Filesystem Size Used Avail Use% Mounted on

/dev/loop0 20M 270K **20M** 2% /mnt

Hexedit:

Αντλώντας από το hexdump του superblock το blocksize & το total block count, έχουμε 1024(blocksize) * 20438(block count) = 20.928.512 bytes = 20mb

8. Πόσα μπλοκ είναι διαθέσιμα/ελεύθερα στο συγκεκριμένο σύστημα αρχείων; Ισοδύναμα, έχει ελεύθερο χώρο το συγκεκριμένο σύστημα αρχείων;

Mount:

root@utopia:/home/user/shared# dumpe2fs ./fsdisk2.img

Free blocks: 19555

Free inodes: 0

root@utopia:/home/user/shared# df -h /mnt

Filesystem Size Used Avail Use% Mounted on

/dev/loop0 20M 270K 20M 2% /mnt

Ενώ φαίνεται να υπάρχει διαθέσιμος χώρος στο σύστημα αρχείων, δεν υπάρχουν ελεύθερα inodes

Hexedit:

Με hexdump του superblock, βλέπουμε στα bytes 12-15 τον αριθμό των ελεύθερων μπλοκ(19555) και στα bytes 16-19 τον αριθμό των ελεύθερων inodes(0), που σημαίνει ότι ενώ υπάρχει χώρος στο σύστημα, είναι κατακερματισμένος και δε μπορεί να προστεθεί άλλο αρχείο.

9. Αφού υπάρχουν διαθέσιμα μπλοκ, τι σας αποτρέπει από το να δημιουργήσετε νέο αρχείο;

Φαίνεται πως έχει εξαντληθεί ο αριθμός των inode που αντιστοιχούν στο σύστημα αρχείων.

2.5 Η εικόνα fsdisk3.img

1. Ποιο εργαλείο στο Linux αναλαμβάνει τον έλεγχο ενός συστήματος αρχείων ext2 για αλλοιώσεις;

Για τον έλεγχο ενός συστήματος αρχείων ext2 για αλλοιώσεις, μπορούμε να χρησιμοποιήσουμε το εργαλείο `e2fsck` ή το `fsck`. Το `e2fsck` και το `fsck` ελέγχουν τη συνέπεια του συστήματος αρχείων και διορθώνουν ενδεχόμενα προβλήματα που μπορεί να εντοπίσουν.
(Source: Wiki, ChatGPT)

2. Ποιοι παράγοντες θα μπορούσαν δυνητικά να οδηγήσουν σε αλλοιώσεις στο σύστημα αρχείων; Αναφέρετε ενδεικτικά δέκα πιθανές αλλοιώσεις.

Υπάρχουν πολλοί παράγοντες που μπορεί να οδηγήσουν σε αλλοιώσεις σε ένα σύστημα αρχείων. Ορισμένοι ενδεικτικοί παράγοντες περιλαμβάνουν:

1. Διακοπή ρεύματος: Η απότομη διακοπή της τροφοδοσίας ενέργειας μπορεί να προκαλέσει αλλοιώσεις κατά τη διάρκεια της εγγραφής στο σύστημα αρχείων.

2. Κακή αποθήκευση υλικού: Προβλήματα στην αποθήκευση υλικού όπως σκληρού δίσκου ή μνήμης μπορούν να προκαλέσουν αλλοιώσεις.

3. Επίθεση malware: Επιθέσεις malware μπορούν να προκαλέσουν αλλοιώσεις στα δεδομένα του συστήματος αρχείων.

4. Σφάλματα λογισμικού: Σφάλματα στο λογισμικό του συστήματος αρχείων μπορούν να οδηγήσουν σε προβλήματα.

5. Μη σωστές διαδικασίες απενεργοποίησης: Η απενεργοποίηση του συστήματος χωρίς ασφαλή αποσύνδεση μπορεί να προκαλέσει αλλοιώσεις.

6. Σφάλματα στον πυρήνα του λειτουργικού συστήματος: Σφάλματα στον πυρήνα μπορεί να επηρεάσουν το σύστημα αρχείων.

7. Απρόσμενη αποσύνδεση εξωτερικών συσκευών: Η άσχημη αποσύνδεση εξωτερικών συσκευών από το σύστημα μπορεί να προκαλέσει προβλήματα.

8. Σφάλματα εφαρμογών: Σφάλματα στις εφαρμογές που χρησιμοποιούνται μπορούν να επηρεάσουν τα αρχεία.

9. Μετατροπή αρχείων κατά τη διάρκεια εγγραφής: Μετατροπή τύπων αρχείων κατά τη διάρκεια εγγραφής μπορεί να οδηγήσει σε προβλήματα.

10. Μη ασφαλείς εντολές χρήστη: Η χρήση μη ασφαλών εντολών από τους χρήστες μπορεί να προκαλέσει αλλοιώσεις στα δεδομένα.

(Source: Wiki, ChatGPT)

3. Τρέξτε το εργαλείο αυτό και επιδιορθώστε το σύστημα αρχείων. Αναφέρετε όλες τις αλλοιώσεις που εντόπισε, εξαντλητικά.

root@utopia:/home/user/shared# e2fsck fsdisk3copy.img

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>? yes

Pass 3: Checking directory connectivity

Pass 4: Checking reference counts

Inode 3425 ref count is 1, should be 2. Fix<y>? yes

Pass 5: Checking group summary information

Block bitmap differences: +34

Fix<y>? yes

Free blocks count wrong for group #0 (7960, counted=7961).

Fix<y>? yes

Free blocks count wrong (926431538, counted=19800).

Fix<y>? yes

fsdisk3.img: ***** FILE SYSTEM WAS MODIFIED *****

fsdisk3.img: 23/5136 files (0.0% non-contiguous), 680/20480 blocks

Βάσει των αποτελεσμάτων, εντοπίζονται τα εξής προβλήματα:

1. Πρόβλημα στη δομή του φακέλου (/dir-2): Η πρώτη καταχώρηση στον φάκελο inode 1717 είναι "BOO" αντί για το αναμενόμενο ".". Το εργαλείο ζητάει επιβεβαίωση για να διορθώσει το πρόβλημα.

2. Πρόβλημα με τα reference counts των inodes: Εντοπίζει πρόβλημα με το reference count του inode 3425, το οποίο είναι 1 αντί για το αναμενόμενο 2. Διορθώνει το πρόβλημα.

3. Διαφορές στα bitmaps των μπλοκ: Εντοπίζει διαφορές στα bitmaps των μπλοκ και τις διορθώνει.

4. Λάθος με τον υπολειπόμενο ελεύθερο χώρο: Το πλήθος των υπολειπόμενων ελεύθερων μπλοκ δεν συμφωνεί με τον αναμενόμενο αριθμό. Το εργαλείο επιδιορθώνει αυτό το πρόβλημα. Αυτό το σφάλμα εμφανίζεται 2 φορές, μια για το group#0 και μια για το συνολικό αριθμό μπλοκς.

Η τελική αναφορά αναφέρει πόσα αρχεία και μπλοκ έχουν διορθωθεί. Το σύστημα αρχείων τροποποιήθηκε, και η νέα κατάσταση είναι 23/5136 αρχεία και 680/20480 μπλοκ.

4. Επαναφέρετε τον δίσκο στην πρότερή του κατάσταση, από την αρχική εικόνα. Εντοπίστε τις αλλοιώσεις με χρήση της μεθόδου hexedit.

5. Επιδιορθώστε κάθε αλλοίωση, ξεχωριστά, με χρήση της μεθόδου hexedit. Για κάθε μία από τις αλλοιώσεις που επιδιορθώνετε, τρέξτε το εργαλείο fsck με τρόπο που δεν προκαλεί καμία αλλαγή [“dry run”], και επιβεβαιώστε ότι πλέον δεν την εντοπίζει.

Για το 1^{ο} σφάλμα κάνοντας αναζήτηση του ASCII String «BOO» με το hexedit, το βρίσκω και το διορθώνω

Πριν: 00837000 B5 06 00 00 0C 00 03 00 42 4F 4F 00 02 00 00 00 ....BOO

Μετά: 00837000 B5 06 00 00 0C 00 01 00 2E 00 4F 00 02 00 00 00 ....O.....

Αλλάζω το μήκος του ονόματος σε 1 byte και μετά αλλάζω το αντίστοιχο byte σε HEX ‘.’ και το επόμενο σε 00.

ΕΞΟΔΟΣ E2FSCK

root@utopia:/home/user/shared# fsck.ext2 -n fsdisk3copy.img

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

Pass 5: Checking group summary information

Block bitmap differences: +34

Fix? no

Free blocks count wrong for group #0 (7960, counted=7961).

Fix? no

Free blocks count wrong (926431538, counted=19801).

Fix? no

fsdisk3.img: ********** WARNING: Filesystem still has errors **********

fsdisk3.img: 23/5136 files (0.0% non-contiguous), 18446744072783140558/20480 blocks

Βλέπω πως δεν έχω πλέον μήνυμα για το σφάλμα που διόρθωσα.

Για το 2ο σφάλμα, που βρίσκεται στο inode 3425, το οποίο είναι το 1ο inode του BG#2, καθώς έχουν 1712 inode/bg

Βρίσκω το μπλοκ του inode table του BG#2 και διαβάζοντας τα δεδομένα του 1ου inode, βρίσκω το λάθος και το διορθώνω

root@utopia:/home/user/shared# hexdump -s$((gdtof+2*gdsz)) -n$((gdsz)) -C fsdisk3.img

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 |................|

Διαβάζουμε από το BGD του BG #2 το block του inode table

Inode table του BG #2 στο block #16389 (0x4005)

Διαβάζουμε το 1ο inode του inode table και βρίσκουμε το block του inode#3425

root@utopia:/home/user/shared# hexdump -s$((16389*bsz)) -n$((inodesz)) -C fsdisk3.img

01001400 ed 41 00 00 00 04 00 00 e9 7a 78 65 e9 7a 78 65 |.A.....zxe.zxe|

01001410 e9 7a 78 65 00 00 00 00 00 00 01 00 02 00 00 00 |.zxe............|

01001420 00 00 00 00 04 00 00 00 e8 00 00 00 00 00 00 00 |................|

01001430 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

01001460 00 00 00 00 e1 ce a8 18 00 00 00 00 00 00 00 00 |................|

01001470 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|

Πριν: 01001410   E9 7A 78 65 00 00 00 00 00 00 01 00 02 00 00 00 .zxe............

Μετά: 01001410 E9 7A 78 65 00 00 00 00 00 00 02 00 02 00 00 00 .zxe.....

ΕΞΟΔΟΣ E2FSCK

root@utopia:/home/user/shared# fsck.ext2 -n fsdisk3copy.img

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

Pass 5: Checking group summary information

Block bitmap differences: +34

Fix? no

Free blocks count wrong for group #0 (7960, counted=7961).

Fix? no

Free blocks count wrong (926431538, counted=19801).

Fix? no

fsdisk3.img: ********** WARNING: Filesystem still has errors **********

fsdisk3.img: 23/5136 files (0.0% non-contiguous), 18446744072783140558/20480 blocks

Block bitmap differences: +34

Ομοίως με πριν, δεν παρατηρώ πλέον μήνυμα για το 2ο σφάλμα

Για το 3ο σφάλμα, βρίσκω το blockbitmap, από το BGD του BG#0, και διορθώνω

Πριν: 00000C00 FF FF FF FF FD FF FF FF FF FF FF FF FF FF FF FF

Μετά: 00000C00 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

ΕΞΟΔΟΣ E2FSCK

root@utopia:/home/user/shared# e2fsck -n fsdisk3copy.img

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

Pass 5: Checking group summary information

Free blocks count wrong (926431538, counted=19800).

Fix? no

fsdisk3.img: 23/5136 files (0.0% non-contiguous), 18446744072783140558/20480 blocks

Free blocks count wrong for group #0 (7960, counted=7961).

Επιβεβαιώνω από την έξοδο πως δε βλέπω μήνυμα σχετικό με το 3ο σφάλμα **αλλά και το 4ο σφάλμα**

Για το 5ο και τελευταίο σφάλμα

Free blocks count wrong (926431538, counted=19801)

Ελέγχω τα bytes 12-15 του superblock
root@utopia:/home/user/shared# hexdump -s$((bsz)) -n$((1024)) -C fsdisk3.img

00000400 10 14 00 00 00 50 00 00 00 04 00 00 **32 39 38 37**

Τα διορθώνω
00000400 10 14 00 00 00 50 00 00 00 04 00 00 **4D 58 00 00**

ΕΞΟΔΟΣ E2FSCK

root@utopia:/home/user/shared# e2fsck -n fsdisk3copy.img

e2fsck 1.46.2 (28-Feb-2021)

fsdisk3.img was not cleanly unmounted, check forced.

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

Βλέπω πως δεν εντοπίζεται πλέον σφάλμα

2ο Μέρος - Το σύστημα αρχείων ext2-lite

Σε αυτό το μέρος μας ζητείται, να συμπληρώσουμε κομμάτια κώδικα σε C, για την ολοκλήρωση του συστήματος αρχείων ext2-lite.

Τα κομμάτια που συμπληρώθηκαν φαίνονται παρακάτω.

Κώδικας που συμπληρώθηκε στο super.c

```c
static int __init init_ext2_fs(void)
{
    int err = init_inodecache();
    if (err)
        return err;

    /* Register ext2-lite filesystem in the kernel */
    /* If an error occurs remember to call destroy_inodecache() */
    /* ? */
    err = register_filesystem(&ext2_fs_type);
        if (err) {
            pr_err("Failed to register ext2-lite filesystem: %d\n", err);
         destroy_inodecache();
        }
    return err;
}

static void __exit exit_ext2_fs(void)
{
    /* Unregister ext2-lite filesystem from the kernel */
    /* ? */
     unregister_filesystem(&ext2_fs_type);
    destroy_inodecache();
}
```

Κώδικας που συμπληρώθηκε στο dir.c

```
ext2_dirent *ext2_find_entry(struct inode *dir, const struct qstr *child,
                             struct page **res_page)
{
    const char *name = child->name;
    int namelen = child->len;
    unsigned reclen = EXT2_DIR_REC_LEN(namelen);
    unsigned long npages = dir_pages(dir);
    unsigned long i;
    struct page *page = NULL;
    ext2_dirent *de;
    char *kaddr;

    if (npages == 0)
        return ERR_PTR(-ENOENT);

    *res_page = NULL;

    /* Scan all the pages of the directory to find the requested name. */
    for (i=0; i < npages; i++) {
        /* ? */
        page = ext2_get_page(dir, i, 0);
        if (IS_ERR(page)) {
            ext2_error(dir->i_sb, __func__, "bad page in #%lu", dir->i_ino);
            return ERR_CAST(page);
    }
    kaddr = page_address(page); //maybe +ext2_last_byte(dir,i) is needed
    de = (ext2_dirent *)kaddr;
    kaddr += ext2_last_byte(dir, i) - reclen;
        /* Iterate through the directory entries in the current page. */
        for ( (char *)de < kaddr ; de = ext2_next_entry(de);) {
            if (de->rec_len == 0)
                break;  // End of entries in this page.

            if (ext2_match(namelen, name, de)) {
                *res_page = page;
                return de;  // Entry found.
            }
        }

        ext2_put_page(page);
    }
    return ERR_PTR(-ENOENT);
}
```

Κώδικας που συμπληρώθηκε στο inode.c

```c
static struct ext2_inode *ext2_get_inode(struct super_block *sb, ino_t ino,
                                         struct buffer_head **p)
{
    struct buffer_head *bh;
    unsigned long block_group;
    unsigned long block;
    unsigned long offset;
    struct ext2_group_desc *gdp;
    unsigned long inodes_pg = EXT2_INODES_PER_GROUP(sb);
    int inode_sz = EXT2_INODE_SIZE(sb);
    unsigned long blocksize = sb->s_blocksize;

    *p = NULL;
    /* Check the validity of the given inode number. */
    if ((ino != EXT2_ROOT_INO && ino < EXT2_FIRST_INO(sb)) ||
        ino > le32_to_cpu(EXT2_SB(sb)->s_es->s_inodes_count))
        goto einval;

    /* Figure out in which block is the inode we are looking for and get
     * its group block descriptor. */
    /* ? */
    block_group = (ino - 1) / inodes_pg;
    gdp = ext2_get_group_desc(sb, block_group, NULL);
    if (!gdp)
        goto eio;
    /* Figure out the offset within the block group inode table */
    /* ? */
    offset = ((ino - 1) % inodes_pg) * inode_sz;
    /* Calculate the block number containing the inode */
    block = le32_to_cpu(gdp->bg_inode_table) + (offset >> EXT2_BLOCK_SIZE_BITS(sb));

    /* Read the block containing the inode table */
    *p = sb_bread(sb, block);
    if (!*p)
        goto eio;
    /* Return the pointer to the appropriate ext2_inode */
    /* ? */
    offset &= (blocksize - 1);
    return (struct ext2_inode *)((*p)->b_data + offset);//
einval:
    ext2_error(sb, __func__, "bad inode number: %lu", (unsigned long)ino);
    return ERR_PTR(-EINVAL);
eio:
    ext2_error(sb, __func__, "unable to read inode block - inode=%lu, block=%lu",
               (unsigned long)ino, block);
    return ERR_PTR(-EIO);
}
```

```
//> Setup the {inode,file}_operations structures depending on the type.
if (S_ISREG(inode->i_mode)) {
    /* ? */
    // Regular file
    inode->i_op = &ext2_file_inode_operations;
    inode->i_fop = &ext2_file_operations;
    inode->i_mapping->a_ops = &ext2_aops;

} else if (S_ISDIR(inode->i_mode)) {
    /* ? */
    // Directory
    inode->i_op = &ext2_dir_inode_operations;
    inode->i_fop = &ext2_dir_operations;
    inode->i_mapping->a_ops = &ext2_aops;
```

Κώδικας που συμπληρώθηκε στο balloc.c

```
static int ext2_allocate_in_bg(struct super_block *sb, int group,
                               struct buffer_head *bitmap_bh, unsigned long *count)
{
    ext2_fsblk_t group_first_block = ext2_group_first_block_no(sb, group);
    ext2_fsblk_t group_last_block = ext2_group_last_block_no(sb, group);
    ext2_grpblk_t nblocks = group_last_block - group_first_block + 1;
    ext2_grpblk_t first_free_bit;
    unsigned long num;

     first_free_bit = find_next_zero_bit_le(bitmap_bh->b_data, EXT2_BLOCKS_PER_GROUP
     (sb), 0);

    if (first_free_bit >= EXT2_BLOCKS_PER_GROUP(sb))
        return -1; //-ENOSPC;  // No free blocks in the group

    num = min(*count, EXT2_BLOCKS_PER_GROUP(sb) - first_free_bit);
    if (num == 0)
        return -1 ;//-ENOSPC;  // No free blocks in the group

    first_free_bit += group_first_block;  // Convert to filesystem-wide block number
    ext2_set_bit_atomic(sb_bgl_lock(EXT2_SB(sb), group), first_free_bit,
    bitmap_bh->b_data);

    *count = num;
    return first_free_bit - group_first_block;  // Return the group-relative
    allocated block
}
```

Ερώτημα σχετικά με τη mkdir:

.mkdir = ext2_mkdir

Δημιουργεί μία νέα σύνδεση ονόματος η οποία αφορά έναν κατάλογο. Αρχικά **αυξάνει τον αριθμό των links** του καταλόγου στον οποίο θα δημιουργηθεί η νέα σύνδεση (**αφήνουμε σαν ερώτημα το γιατί πρέπει να γίνει αυτό εδώ)** και εντοπίζει και δεσμεύει ένα ελεύθερο ext2 inode στο δίσκο καλώντας την ext2_new_inode. Αφού αρχικοποιήσει κατάλληλα το νέο inode, αρχικοποιεί και τα αντίστοιχα data blocks του inode καλώντας την ext2_make_empty. Τέλος, προσθέτει τη νέα σύνδεση ονόματος στον γονικό κατάλογοκαλώντας την ext2_add_link και αρχικοποιεί το VFS dentry καλώντας τηνd_instantiate_new.

Η "Αύξηση του Αριθμού των Συνδέσμων" στην αρχή της διαδικασίας είναι ένα κρίσιμο βήμα για τη διατήρηση της συνέπειας και το σωστό χειρισμό της δομής του συστήματος αρχείων. Οι λόγοι για τους οποίους γίνεται αυτή η ενέργεια νωρίς:

1. Αποφυγή Συνθηκών Ανταγωνισμού (Race Conditions):
   - Η αύξηση του αριθμού των συνδέσμων του γονικού καταλόγου νωρίς βοηθά στον αποκλεισμό συνθηκών ανταγωνισμού. Αν η αύξηση του αριθμού γίνει μετά από άλλες λειτουργίες (π.χ. δέσμευση inode, αρχικοποίηση μπλοκ δεδομένων), μπορεί να εισαχθεί ένα παράθυρο όπου πολλές διεργασίες ή νήματα που προσπαθούν ταυτόχρονα να δημιουργήσουν καταλόγους ενδέχεται να επηρεάσουν η μία την άλλη. Η αύξηση του αριθμού των συνδέσμων νωρίς βοηθά στην καθιέρωση μιας σαφούς σειράς των λειτουργιών.
2. Ακεραιότητα του Καταλόγου:
   - Ένας κατάλογος σε ένα σύστημα αρχείων είναι ουσιαστικά ένα ειδικό είδος αρχείου που περιέχει εγγραφές που δείχνουν σε άλλα αρχεία ή καταλόγους. Όταν δημιουργείται ένας νέος κατάλογος, πρέπει να συνδεθεί με τον γονικό του κατάλογο. Η αύξηση του αριθμού των συνδέσμων νωρίς εξασφαλίζει ότι ο γονικός κατάλογος είναι άμεσα ενήμερος για την ύπαρξη του νέου υποκαταλόγου.
3. Συνέπεια στη Λειτουργία:
   - Ο αριθμός των συνδέσμων αντιπροσωπεύει τον αριθμό των σκληρών συνδέσμων(hardlinks) προς έναν κατάλογο. Με την αύξησή του νωρίς, το σύστημα εξασφαλίζει ότι ο γονικός κατάλογος ενημερώνεται πριν συνεχιστούν άλλες λειτουργίες που σχετίζονται με τον νέο κατάλογο. Αυτό βοηθά στη διατήρηση μιας συνεπούς κατάστασης του συστήματος αρχείων.
4. Χειρισμός Σφαλμάτων και Αναστροφή:
   - Αν προκύψει σφάλμα κατά τη διάρκεια των επόμενων βημάτων (π.χ. δέσμευση inode), ο αριθμός των συνδέσμων μπορεί να αναστραφεί, αποφεύγοντας άστοχες αλλαγές στο σύστημα αρχείων.

Συνολικά, η "Αύξηση του Αριθμού των Συνδέσμων" στην αρχή εξασφαλίζει μια ορθή, ασφαλή και συνεπή εισαγωγή του νέου καταλόγου στο σύστημα αρχείων.