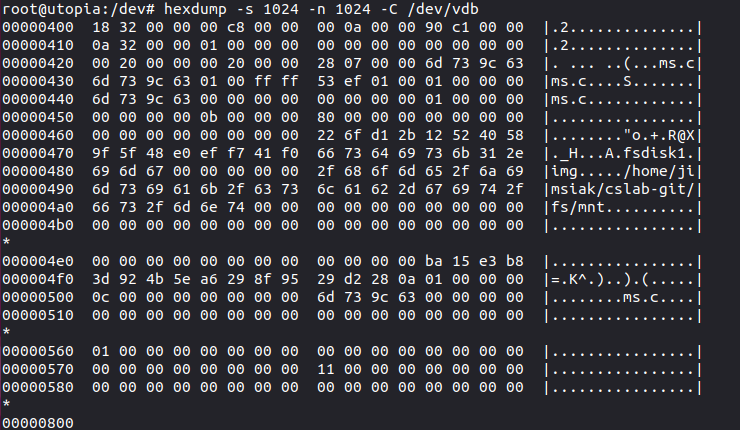
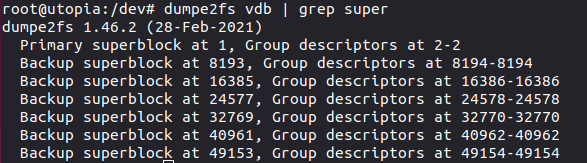
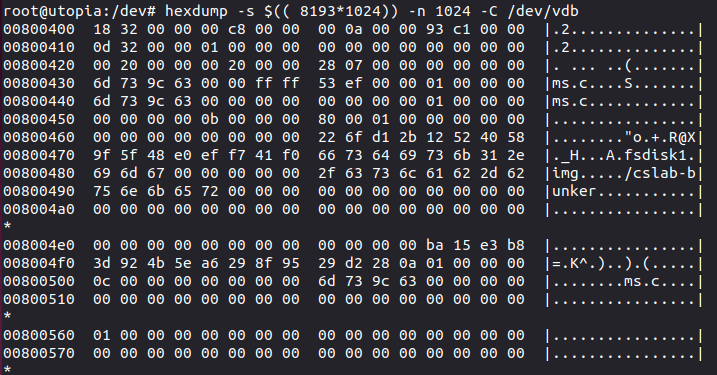
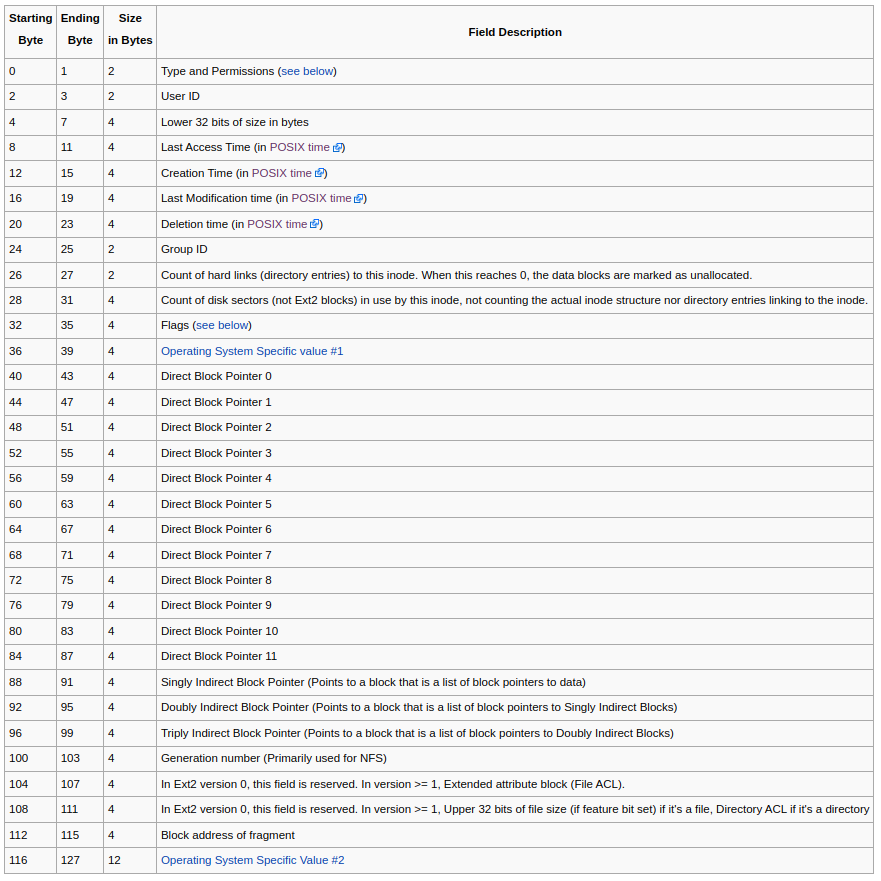
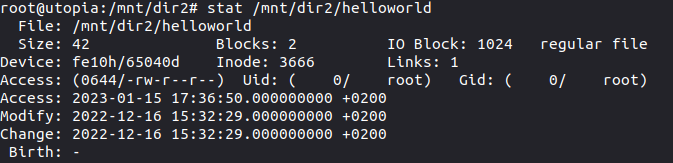
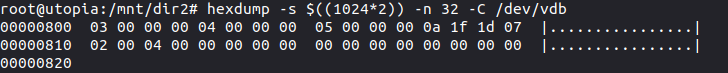
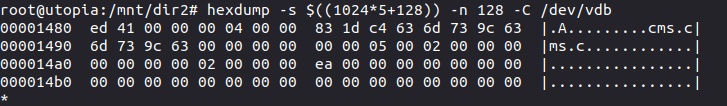
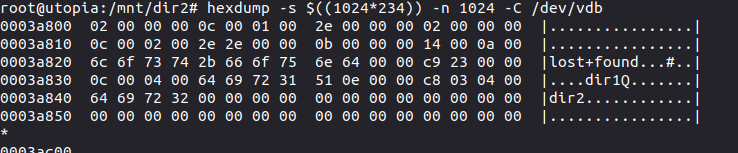
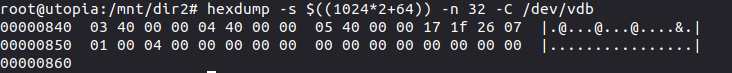
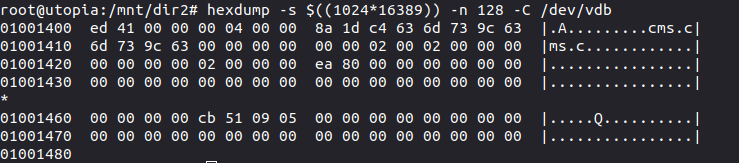
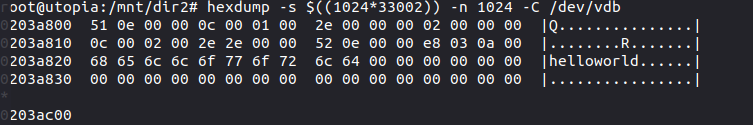
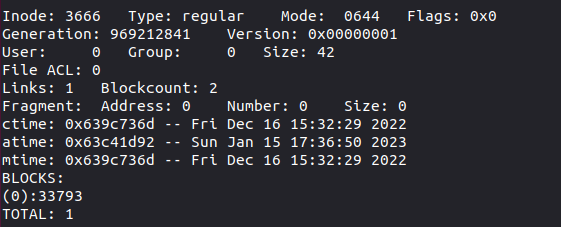
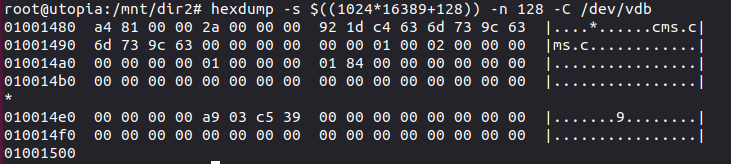
**Εργαστήριο Λειτουργικών Συστημάτων**

**Εργαστηριακή Άσκηση 3**

ΝΤΟΝΤΟΡΟΣ ΗΛΙΑΣ

el19206

**ΑΣΚΗΣΗ 1**

1. Η προσθήκη που έκανα ήταν η παρακάτω γραμμή στο utopia.sh και στο utopia εμφανίζεται ως η συσκευή vdb στον φάκελο /dev. 
2. Εκτελώντας την εντολή lsblk μπορούμε να δούμε ότι ο δίσκος vdb έχει μέγεθος 52428800 bytes, δηλαδή 50 megabytes. Επίσης με την εντολή hexdump -s 1024 -n 1024 -C /dev/vdb μπορούμε να δούμε τα περιεχόμενα του superblock όπου το block count είναι 51200 και το block size είναι 1024 άρα το συνολικό μέγεθος είναι 52428800 bytes. 
3. Εκτελώντας την εντολή lsblk -f μπορούμε να δούμε ότι περιέχει σύστημα αρχείων ext2.Eπίσης από το superblock μπορούμε να βρούμε στο byte 56-57 μπορούμε να δούμε ότι υπάρχει το id για το ext2: ef53
4. Με την εντολή dumpe2fs vdb μπορούμε να δούμε το Filesystem created  MISSING HEXDUMP
5. Με την παραπάνω εντολή φαίνεται το Last mount time. Επίσης στο superblock στα byte 44–47 φαίνεται η τιμή 6399c736d η οποία είναι σε posix time και αν την μετατρέψουμε σε κανονική είναι η ίδια με την παραπάνω.
6. Με την εντολή dumpe2fs vdb μπορούμε να δούμε το path που προσαρτήθηκε τελευταία φορά:  ενώ το ίδιο φαίνεται στο superblock στα byte 136-199.
7. Με την εντολή dumpe2fs vdb μπορούμε να δούμε το Last write time όπως φαίνεται και στην εικόνα της ερώτησης 4. Επίσης στο superblock στα byte 48-51 είναι φαίνεται η ημερομηνία που τροποποιήθηκε τελευταία φορά σε posix time.
8. Το ext2 χωρίζει τον χώρο του δίσκου σε λογικά κομμάτια συνεχόμενου χώρου σταθερού μήκους. Κάθε τέτοιο κομμάτι ονομάζεται Block.([πηγή](https://wiki.osdev.org/Ext2" \l "Creator_Operating_System_IDs))
9. Με την εντολή dumpe2fs vdb μπορούμε να δούμε το Block size : Επίσης στο superblock στα byte 24-27 βλέπουμε ότι η τιμή είναι 00000000 δηλαδή πρέπει να κάνουμε shift left 0 φορές το 1024 για να βρούμε το block size άρα τελικά το block size είναι 1024 bytes.
10. Το inode είναι ένα structure που αντιπροσωπεύει ένα αρχείο ή ένα φάκελο ή ένα symbolic link. Τα inodes δεν περιέχουν δεδομένα του στοιχείου που αντιπροσωπεύουν αλλά είναι σύνδεσμοι για το block που περιέχει το στοιχείο που αντιπροσωπεύουν και έτσι με αυτόν τον τρόπο μπορούν να έχουν σταθερό μέγεθος και για κάθε block group να είναι αποθηκευμένα σε ένα indexed array για γρήγορη πρόσβαση.([πηγή](https://wiki.osdev.org/Ext2" \l "Creator_Operating_System_IDs))
11. Με την εντολή dumpe2fs vdb εμφανίζεται το inode size: Επίσης, στο superblock στα byte 88-89 φαίνεται ότι το μεγέθος του inode είναι 0x80 δηλαδή 128 bytes.
12. Με την εντολή dumpe2fs vdb εμφανίζονται τα Free inodes και free blocks:  Επίσης στο superblock στα byte 12-15 εμφανίζεται ο αριθμός των ελεύθερων block (0xc190=49552) και στα byte 16-19 εμφανίζεται ο αριθμός των ελεύθερων inodes (0x320a=12810).
13. Το superblock είναι ένα block το οποίο περιέχει πληροφορίες για το σύστημα αρχείων όπως τα παραπάνω και διάφορες προαιρετικές λειτουργίες που υπάρχουν.
14. Το superblock βρίσκεται πάντα στο byte 1024 και έχει μέγεθος 1024 bytes.
15. Γιατί το superblock είναι πολύ σημαντικό και χωρίς αυτό δεν μπορεί να γίνει προσάρτηση του συστήματος αρχείων. Με τα εφεδρικά αντίγραφα μπορεί εύκολα να γίνει επαναφορά του χωρίς την απώλεια δεδομένων.
16. Αντίγραφα του superblock υπάρχουν στην αρχή κάθε block group όπως σε κάθε σύστημα αρχείων ext2 (revision 0). Μπορούμε να το επιβεβαιώσουμε με την εντολή dumpe2fs /dev/vdb | grep super : ή με την εντολή hexdump –s $((8193\*1024)) –n 1024 –C /dev/vdb οπου εμφανίζεται το ίδιο superblock στο block 8193:
17. Τα block groups είναι ομάδες συνεχόμενων blocks και inodes τα οποία ομαδοποιούνται για να ελαττωθεί το fragmentation και να υπάρχει πιο γρήγορη πρόσβαση σε πολλά συνεχόμενα δεδομένα.([πηγή](https://docs.kernel.org/filesystems/ext2.html))
18. Το μέγιστο μέγεθος κάθε block group είναι 8 φορές το μέγεθος του block καθώς κάθε block group έχει ένα block usage bitmap μεγέθους 1 block. Άρα ο αριθμός των block groups σε ένα σύστημα αρχείων είναι όσο το μέγεθος του συστήματος αρχείων σε blocks διά το μέγεθος του block group στρογγυλοποιημένο προς τα πάνω. ([πηγή](https://docs.kernel.org/filesystems/ext2.html))
19. Για να βρούμε τον αριθμό των block groups μπορούμε να πάρουμε το μέγεθος του συστήματος αρχείων και να το διαιρέσουμε με το block size και να βρούμε το block count (51200) (μπορούμε να το βρούμε και πιο γρήγορα με την εντολή dumpe2fs vdb ή διαβάζοντας το superblock). Έπειτα παίρνουμε το αποτέλεσμα και το πολλαπλασιάζουμε το 8 και έχουμε το block group size (8192). Τέλος διαιρούμε το block count με το block group size και το στρογγυλοποιούμε προς τα πάνω. Άρα στο συγκεκριμένο σύστημα αρχείων έχουμε 7 block groups.
20. Ο block group descriptor βρίσκεται μέσα στο block group descriptor table και περιέχει πληροφορίες για κάθε block group που αντιπροσωπεύει, όπως για παράδειγμα την διεύθυνση του block usage bitmap.([πηγή](https://wiki.osdev.org/Ext2" \l "Determining_the_Number_of_Block_Groups))
21. Όπως και το superblock κάθε block group descriptor είναι πολύ σημαντικός για το σύστημα αρχείων και διατηρούνται αντίγραφα σε κάθε block group (αμέσως μετά το αντίγραφο του superblock) σε περίπτωση που υπάρχει αλλοίωση του πρωτότυπου.([πηγή](https://www.science.unitn.it/~fiorella/guidelinux/tlk/node98.html))
22. Στο συγκεκριμένο σύστημα αρχείων υπάρχει αντίγραφο του block group descriptor table στην αρχή του κάθε block group αμέσως μετά το αντίγραφο του superblock. Αυτό μπορούμε να το δούμε και με την εντολή dump2efs vdb και αν εμφανίσουμε το δεύτερο block του κάθε block group με την εντολή hexdump.
23. Το block bitmap και το inode bitmap αντίστοιχα είναι blocks που βρίσκονται στην αρχή κάθε block group μετά το block group descriptor table τα οποία αποτελούνται από μία αλληλουχία από bits και κάθε ένα bit αντιπροσωπεύει ένα block (inode) και η τιμή του bit δηλώνει ότι το block (inode) είναι ελεύθερο (για τιμή 0) ή δεσμευμένο (για τιμή 1).([πηγή](http://www.science.smith.edu/~nhowe/262/oldlabs/ext2.html))
24. Τα inode tables είναι μια σειρά από blocks που βρίσκονται σε κάθε block group μετά από το inode bitmap και περιέχουν inodes για τα blocks που υπάρχουν μέσα στο block group.([πηγή](http://www.science.smith.edu/~nhowe/262/oldlabs/ext2.html))
25. Τα πεδία του inode φαίνονται στον παρακάτω πίνακα και κάθε inode βρίσκεται στο inode table του block group στο οποίο βρίσκεται το block που αντιπροσωπεύει.([πηγή](https://wiki.osdev.org/Ext2" \l "Determining_the_Number_of_Block_Groups)) 
26. Με την εντολή dumpe2fs vdb μπορούμε να δούμε το blocks per group και το inodes per group:  Επίσης η ίδια πληροφορία βρίσκεται και στο superblock στα byte 32-32 (blocks) και 40-43 (inodes).
27. Για να βρούμε το inode του συγκεκριμένου αρχείου μπορούμε να προσαρτήσουμε το σύστημα αρχείων με την εντολή mount /dev/vdb /mnt και μετά να εκτελέσουμε την εντολή stat /mnt/dir2/helloworld όπου θα εμφανίσει το inode του συγκεκριμένου αρχείου: Επίσης μπορούμε να μπορούμε να βρούμε το inode ακολουθώντας τα εξής βήματα: Βρίσκουμε το inode table του block group 0 και στα byte 8-11 βλέπουμε την τιμή 0x00000005 που αυτή είναι ο αριθμός του πρώτου block του inode table του block group 0. Το inode του root directory βρίσκεται πάντα στο inode #2 άρα στα byte 128-256 του block #5 όπου βρίσκουμε το directory entry block του root directory (”/”) είναι το block #234. Διαβάζοντας τα περιεχόμενα του block #234 βλέπουμε τα περιεχόμενα του root directory και βρίσκουμε ότι το inode του directory dir2 είναι το inode #3665 το οποίο είναι το 1o inode στο block group #2  Από το block group descriptor βρίσκουμε το πρώτο block του inode table του block group #2 το οποίο είναι το block #16389 Διαβάζουμε το πρώτο inode του block group #2 στο block #16389 και βρίσκουμε το directory entry block του directory /dir2 το οποίο είναι το block #33002 Διαβάζοντας το block #33002 βρίσκουμε ότι το αρχείο helloworld έχει inode το inode #3666.
28. Κάθε block group έχει 1832 inodes άρα το inode #3666 είναι το 2o inode του block group #2
29. To inode table του block group #2 μπορούμε να το δούμε με την εντολή dumpe2fs /dev/vdb ή όπως το βρήκαμε παραπάνω στο block #16389.
30. Μπορούμε να δούμε τα πεδία του inode #3666 με την εντολή debugfs –R ‘stat <3666>’ /dev/vdb  Ή με την εντολή hexdump –s $((1024\*16389)) -n 128 /dev/vdb 

Type and permissions : 0x81a4

User ID : 0x0000

Lower 32 bits of size in bytes :0x0000002a

Last access time : 0x63c41d92

Creation time : 0x639c736d

Last modification time : 0x639c736d

Deletion time : 0x00000000

Group id : 0x0000

Count of hard links : 0x0001

Count of diks sectors : 0x00000002

Flags : 0x00000000

Operation system specific value #1 : 0x00000001

Direct block pointer 0 : 0x00008401

Direct block pointer 1-11: 0x00000000

Singly indirect block pointer : 0x00000000

Doubly indirect block pointer : 0x00000000

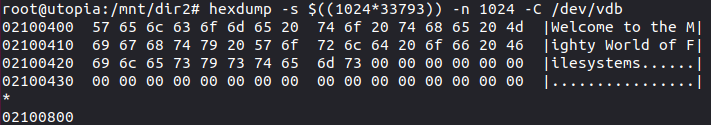
Triply indirect block pointer : 0x00000000

Generation number : 0x39c503a9

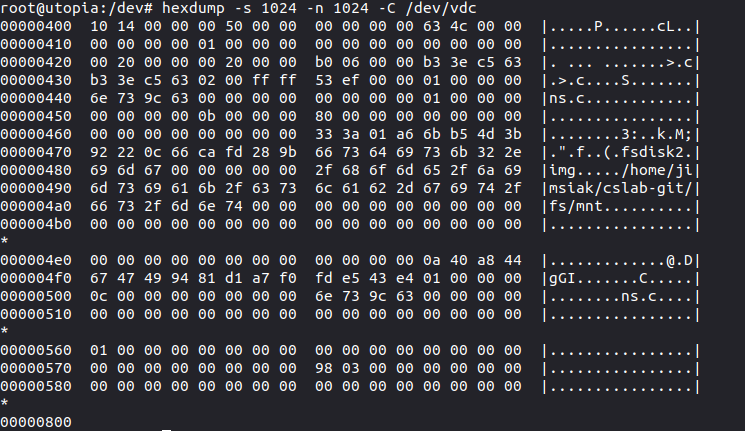
Τα byte 104-111 είναι reserved στο ext2 revision 0 και είναι όλα 0

Block address of fragment : 0x000000

Operating system specific value #2 : 0

1. Είναι αποθηκευμένα στο block που δείχνει ο direct block pointer 0 δηλαδή στο block #33793 όπως φαίνεται και παραπάνω στα δεδομένα του inode #3666
2. Όπως φαίνεται και παραπάνω με τις εντολές debugfs –R ‘stat <3666>’ /dev/vdb και hexdump –s $((1024\*16389)) -n 128 /dev/vdb το μέγεθος του αρχείου είναι 42 bytes.
3. Με την εντολη cat /mnt/dir2/helloworld μπορούμε να δούμε τα περιεχόμενα του αρχείου Επίσης μπορούμε να δούμε τα περιεχόμενα του block #33793 τα οποίο είναι το block που περιέχει τα δεδομένα του αρχείου helloworld. 

**ΑΣΚΗΣΗ 2**

1. Για να συνδέσουμε την εικόνα του δίσκου fdisk2.img στο κατάλογο /mnt εκτελούμε την εντολή mount /dev/vdc /mnt.
2. Για να δημιουργήσουμε ένα κενό αρχείο με την εντολή touch εκτελούμε την εντολή touch /mnt/file1.
3. Η εντολή δεν εκτελέστηκε και βγάζει μήνυμα ότι δεν υπάρχει ελεύθερος χώρος στην συσκευή.
4. Η εντολή touch προσπάθησε να καλέσει την openat αλλά αυτή επέστρεψε -1 (ENOSPC)
5. Με την εντολή find . -type f | wc –l μπορούμε να δούμε τον αριθμό των αρχείων μέσα στον φάκελο mnt  Επίσης μπορούμε να δούμε από το superblock τα διαθέσιμα inodes τα οποία είναι 0 και το συνολικό αριθμό των inodes στο σύστημα αρχείων που είναι 5136 αν από αυτά αφαιρέσουμε τα πρώτα 11 που είναι δεσμευμένα και προσθέσουμε το inode για το root directory και για το lost+found τότε τα συνολικά inodes είναι 5127. Αν τώρα διαβάσουμε τους block group descriptors μπορούμε να βρούμε το συνολικό αριθμό των directories στα block groups τα οποία είναι 259. Αφαιρούμε λοιπόν τα inodes που είναι δεσμευμένα από τα directories και βρίσκουμε ότι το συνολικός αριθμός των inodes που είναι δεσμευμένα από αρχεία είναι 4868 άρα ο συνολικός αριθμός αρχείων είναι 4868.
6. Τα συνολικά blocks του συστήματος είναι 20480 από τα οποία χρησιμοποιούνται τα 19555 άρα τα δεδομένα καταλαμβάνουν 925 blocks δηλαδή 947200 bytes. Την πληροφορία αυτή μπορούμε να την βρούμε και από την εντολή dumpe2fs /dev/vdc και κοιτώντας τις πληροφορίες μέσα στο superblock. Τα μεταδεδομένα είναι ο όγκος που καταλαμβάνουν τα inode στο σύστημα αρχείων δηλαδή 32\*4868 δηλαδή 155776 bytes.
7. ΤΟ συνολικό μέγεθος του συστήματος αρχείων είναι όσο τα blocks επί το block size δηλαδή 2097152 bytes.
8. Το σύστημα αρχείων έχει 19555 blocks ακόμα διαθέσιμα κάτι το οποίο μπορούμε να δούμε με την εντολή dumpe2fs /dev/vdc και από το superblock.
9. Στο συγκεκριμένο σύστημα αρχείων υπάρχουν αρκετά διαθέσιμα blocks και θα μπορούσαμε θεωρητικά να δημιουργήσουμε πολλά αρχεία αλλά όπως φαίνεται και παραπάνω δεν υπάρχουν διαθέσιμα inodes ώστε να μπορέσουν να αποθηκευτούν οι πληροφορίες για το νέο αρχείο.

**ΑΣΚΗΣΗ 3**

1. Στα linux μπορούν να χρησιμοποιηθούν και το fsck και το e2fsck (το οποίο είναι αποκλειστικό για τα ext2, ext3 και ext4) για να ελέγξουν ένα σύστημα αρχείων ext2 για αλλοιώσεις.
2. 10 λόγοι που προκαλούν αλλοιώσεις στο σύστημα αρχείων είναι :

1) Λάθος λογισμικού δηλαδή κάποιο λογισμικό προσπάθησε να γράψει κάτι σε εάν σύστημα αρχείων άλλαξε κάποια άλλη σημαντική οντότητα στο δίσκο (π.χ. το superblock).

2) Αστοχία hardware , για παράδειγμα ένα δίσκος μπορεί να μην λειτουργεί σωστά με αποτέλεσμα να γραφεί δεδομένα σε λάθος block.

3) Λάθος από την μεριά του χρήστη, δηλαδή μπορεί ο χρήστης να αλλάξει κάτι στο δίσκο που να μην έπρεπε να αλλαχτεί (π.χ. το group descriptor table).

4) Ιοί οι οποίοι καταστρέφουν τις απαραίτητες δομές για την λειτουργία του συστήματος αρχείων.

5) Απότομη παύση του συστήματος την ώρα που γίνονται διεργασίας πάνω στο σύστημα αρχείων.

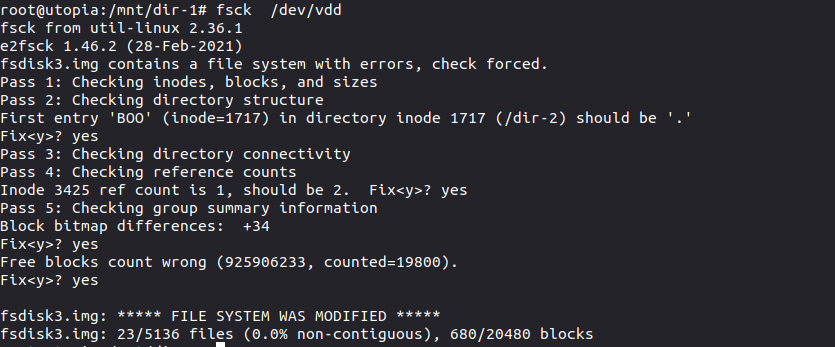
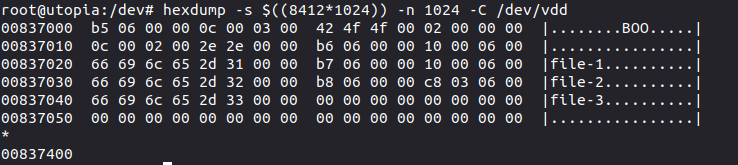
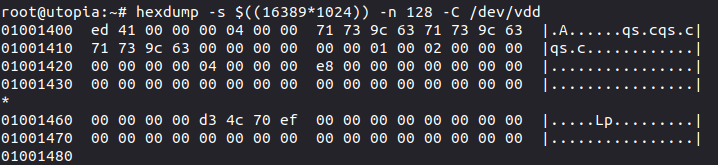
6) Χρήση μη συμβατού driver ή driver με λάθη στον κώδικά του.

7) Χρήση λειτουργικού συστήματος που δεν υποστηρίζει το σύστημα αρχείων.

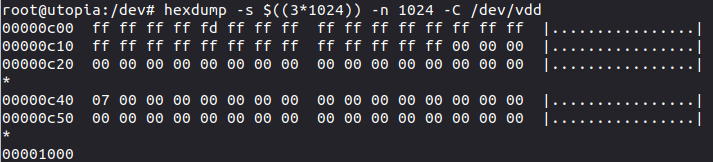
8) Αλλαγή στις σημαντικές δομές του δίσκου από εξωτερικούς παράγοντες όπως για παράδειγμα από υπέρταση ή έκθεση σε έντονα ηλεκτρομαγνητικά πεδία.

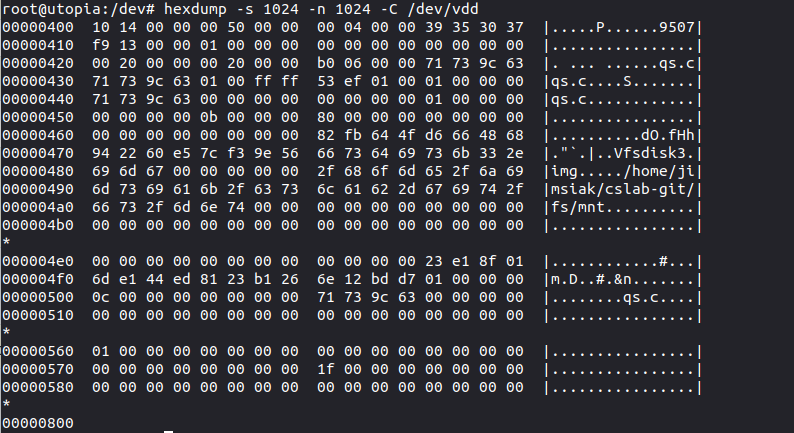
9) Σφάλμα κατά την δημιουργία του συστήματος αρχείων το οποίο να μην εμφανίσει αμέσως επιπτώσεις.

10) Απότομη παύση της διαδικασίας μεταφοράς αρχείων στο σύστημα αρχείων.

1. Η πρώτη αστοχία που εμφανίζεται είναι στο directory entry του καταλόγου dir-2 και το πρώτο entry είναι ‘BOO’ ενώ θα έπρεπε να είναι ‘.’. Η δεύτερη αστοχία είναι στο reference count του inode #3425 και είναι 1 ενώ θα έπρεπε να είναι 2. Η τρίτη αστοχία είναι στο block bitmap όπου το block #34 εμφανίζεται ως διαθέσιμο ενώ στην πραγματικότητα δεν είναι. Η τέταρτη αστοχία ήταν στο free block count που ήταν 925906233 ενώ θα έπρεπε να είναι 19801.
2. Την πρώτη αστοχία μπορούμε να την δούμε στο block που δείχνει το inode #1717 δηλαδή στο block #8412. Την δεύτερη αστοχία μπορούμε να την δούμε στο inode #3435 το οποίο είναι το πρώτο inode του block group #2 και βρίσκεται στο block #16389. 

Την Τρίτη αστοχία μπορούμε να την δούμε κοιτώντας στο block bitmap όπου το bit #34 έχει τιμή 0 ενώ θα έπρεπε να είναι 1.



Την τέταρτη αστοχία μπορούμε να την δούμε στο superblock που τα bytes 11-15 είναι 0x37303539 δηλαδή 925906233.

1. Η πρώτη επιδιόρθωση που κάνουμε είναι τα byte 0x837006-0x83700A από 03 00 42 4F 4F τα κάνουμε 01 00 2E 00 00. Η δεύτερη επιδιόρθωση που κάνουμε είναι το byte 0x100141A από 01 σε 02. Η τρίτη επιδιόρθωση που κάνουμε είναι να αλλάξουμε την τιμή του bit που αναπαριστά την διαθεσιμότητα του block #34 στο block bitmap από 0 σε 1, δηλαδή αλλάζουμε την τιμή του byte 0xC04 από FD σε FF. Την τέταρτη επιδιόρθωση που κάνουμε είναι να αλλάξουμε τα byte 0x40C –0x40F από 39 35 30 37 σε 58 4D 00 00. Με αυτές τις αλλαγές όταν εκτελούμε την εντολή fsck –n /dev/vdd φαίνεται πως δεν υπάρχει κάποιο λάθος στο σύστημα αρχείων.