Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών

Λειτουργικά Συστήματα (ΗΥ321)

Ακαδημαϊκό Έτος 2017-2018

4η Εργαστηριακή Άσκηση

Τελευταία Ενημέρωση: 6 Μαΐου 2018

Περιεχόμενα

1 Εισαγωγικά

Ο δρομολογητής αιτήσεων I/O στο λειτουργικό σύστημα Linux είναι υπεύθυνος για τη δρομολόγηση των αιτήσεων των εφαρμογών για I/O προς τις αντίστοιχες συσκευές, με κατάλληλη σειρά. Ο δρομολογητής μπορεί να σχεδιαστεί ώστε να βελτιστοποιεί ένα ή περισσότερα από μια σειρά διαφορετικών και συχνά αντικρουόμενων κριτήριων, όπως ο ρυθμός εξυπηρέτησης αιτήσεων, η προβλεψιμότητα της αναμονής για την εξυπηρέτηση των αιτήσεων κλπ.

Στα πλαίσια αυτής της εργασίας καλείστε να υλοποιήσετε έναν δρομολογητή αιτήσεων I/O που εφαρμόζει τον αλγόριθμο *C-LOOK*. Όπως γνωρίζουμε από το μάθημα, ο δρομολογητής *C-LOOK* διατηρεί λίστα μη εξυπηρετημένων αιτήσεων I/O, ταξινομημένη με βάση τη θέση που πρέπει να προσπελαστεί στην επιφάνεια του δίσκου – ή για την ακρίβεια και ισοδύναμα με βάση τον αριθμό του λογικού block. Οι αιτήσεις εξυπηρετούνται καθώς η κεφαλή σαρώνει το δίσκο προς τη μία κατεύθυνση. Όταν εξυπηρετηθεί η διαθέσιμη αίτηση με τον μεγαλύτερο αριθμό λογικού block, η κεφαλή επιστρέφει για να ξεκινήσει και πάλι τη σάρωση, εξυπηρετώντας τη διαθέσιμη αίτηση με το μικρότερο αριθμό λογικού block.

Επίσης, θα χρειαστεί να υλοποιήσετε κατάλληλη εφαρμογή ώστε να αξιολογήσετε πειραματικά το δρομολογητή που υλοποιήσατε.

1.1 Προαπαιτούμενα

Πριν ξεκινήσετε την υλοποίηση θα πρέπει να έχετε διαβάσει τα κεφάλαια 13 και 14 από το βιβλίο ”Linux Kernel Development”. Επίσης, θα πρέπει να έχετε μελετήσει τον κώδικα του αρχείου *block/noop-iosched.c* του πηγαίου κώδικα του πυρήνα του Linux, το οποίο υλοποιεί έναν απλό δρομολογητή αιτήσεων με βάση τον αλγόριθμο *FCFS*. Ιδιαίτερα χρήσιμα είναι επίσης τα αρχεία *Documentation/block/biodoc.txt* (ιδιαίτερα το κεφάλαιο 4), *Documentation/block/request.txt* και *include/linux/blkdev.h*. Τέλος, άλλα σχετιζόμενα αρχεία είναι τα *block/elevator.c* και *include/linux/bio.h*.

Για αυτή την εργασία θα ξεκινήσετε από ένα ”καθαρό” δέντρο κώδικα του πυρήνα του Linux. Για να γίνει αυτό εκτελέστε τις ακόλουθες εντολές:

```
cd /usr/src
rm -rf linux-3.14.62-dev
cp -R linux-3.14.62-orig linux-3.14.62-dev
```

1.2 Προθεσμία και Τρόπος Παράδοσης

Η άσκηση θα πρέπει να παραδοθεί έως τα **μεσάνυχτα της Κυριακής 20/5/2018**. Η προθεσμία είναι τελική και δεν πρόκειται να δοθεί καμία παράταση – συνολικά ή ατομικά – για κανένα λόγο.

Η παράδοση θα γίνει στο e-class.

Στο συνοδευτικό κείμενο θα πρέπει να υπάρχουν τα ονοματεπώνυμα, τα Α.Ε.Μ. και τα e-mail των μελών της ομάδας. Επίσης μπορείτε να συμπεριλάβετε και οτιδήποτε άλλο μήνυμα θέλετε να μας μεταφέρετε σχετικά με την άσκησή σας.

Τέλος, η εργασία σας θα πρέπει να συμπεριληφθεί στο μήνυμα ως ένα συνημμένο αρχείο. Για τις λεπτομέρειες δημιουργίας του αρχείου δείτε την παράγραφο 4.

Το πολύ 24 ώρες μετά τη λήξη της προθεσμίας υποβολής θα ανακοινωθεί κατάλογος των ομάδων που έχουν υποβάλλει εργασία. Σε περίπτωση που έχετε υποβάλλει εργασία και δεν βρίσκεστε στον κατάλογο, επικοινωνήστε άμεσα με τον μεταπτυχιακό φοιτητή Β. Βασιλειάδη ή τον διδάσκοντα (Χ. Αντωνόπουλο).

1.3 Κώδικας Ηθικής

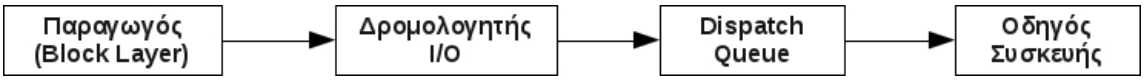
Κάθε ομάδα θα πρέπει να εργαστεί ανεξάρτητα. Είναι δεκτό και θεμιτό διαφορετικές ομάδες να ανταλλάξουν απόψεις σε επίπεδο γενικής ιδέας ή αλγόριθμου. Απαγορεύεται όμως κάθε συζήτηση / ανταλλαγή κώδικα ή ψευδοκώδικα. Σημειώστε ότι οι ασκήσεις θα ελεγχθούν τόσο από αυτόματο σύστημα όσο και από εμάς για ομοιότητες μεταξύ των ομάδων αλλά και για ομοιότητες με τυχόν λύσεις που μπορεί να βρεθούν στο διαδίκτυο ή λύσεις που έχουν δοθεί σε προηγούμενα έτη. Σε περίπτωση αντιγραφής οι ασκήσεις (όλων των φάσεων) όλων των εμπλεκόμενων φετινών ομάδων μηδενίζονται χωρίς καμία περαιτέρω συζήτηση.

Όλα τα μέλη στο εσωτερικό κάθε ομάδας θα πρέπει να έχουν ισότιμη συμμετοχή στην ανάπτυξη κάθε φάσης. Διατηρούμε το δικαίωμα να επιβεβαιώσουμε αν αυτό τηρείται με προσωπικές συνεντεύξεις / προφορική εξέταση.

2 Συνοπτική Περιγραφή Λειτουργικότητας Δρομολογητή I/O

Για να υλοποιήσετε το δρομολογητή I/O, θα πρέπει να έχετε όσο τη δυνατόν καλύτερη εικόνα της διαδρομής των αιτήσεων I/O στο Linux. Η διαδρομή αυτή συνοψίζεται στην εικόνα 1.

Η δική σας δουλειά είναι να γράψετε τον κώδικα για το 2ο κομμάτι της αλυσίδας. Αυτό θα γίνει υλοποιώντας αρκετές από τις συναρτήσεις που περιγράφονται στη δομή *struct elevator_type*,



Εικόνα 1: Η διαδρομή των αιτήσεων I/O στο Linux

η οποία ορίζεται στο αρχείο *include/linux/elevator.h*. Η συνάρτηση που χρησιμοποιείται από το Block Layer για να δώσει αιτήσεις I/O στο δρομολογητή είναι η *elevator_add_req_fn()* ενώ αυτή που χρησιμοποιείται από το δρομολογητή I/O για να στείλει αιτήσεις (με την επιλεγμένη σειρά) στη Dispatch Queue είναι η *elevator_dispatch_fn()*. Οι παραπάνω συναρτήσεις υλοποιούν ουσιαστικά το 1ο και το 2ο βέλος της εικόνας 1 αντίστοιχα.

Για περισσότερες πληροφορίες σε ότι αφορά τη δομή και τις συναρτήσεις, δείτε την παράγραφο 4.1 του αρχείου *Documentation/block/biodoc.txt*.

3 Ζητούμενα

3.1 Υλοποίηση Δρομολογητή Αιτήσεων I/O C-LOOK

Μπορείτε να ενεργοποιήσετε το δρομολογητή C-LOOK βασισμένοι στον noop scheduler που χρησιμοποιήσαμε στην 1η εργασία.

Ακολουθήστε τα παρακάτω βήματα:

1. Δημιουργήστε ένα αντίγραφο του αρχείου *block/noop-iosched.c*, που υλοποιεί τον δρομολογητή I/O noop. Ονομάστε το αντίγραφο *block/clook-iosched.c*.

2. Κάντε τις απαραίτητες ενέργειες ώστε ο νέος δρομολογητής να αναγνωρίζεται και να συμπεριλαμβάνετε κατά τη μεταγλώττιση του λειτουργικού:

   - Προσθέστε την ακόλουθη γραμμή στο *block/Makefile*, ώστε ο δρομολογητής να συμπεριλαμβάνεται κατά τη μεταγλώττιση, εφόσον έχετε επιλέξει κάτι τέτοιο κατά τη φάση του configuration (όπως περιγράφεται στην παράγραφο 3.2:
     obj-$(CONFIG_IOSCHED_CLOOK) += clook-iosched.o
   - Ανοίξετε το *block/Kconfig.iosched*. Πρσθέσετε τις απαραίτητες γραμμές ώστε ο δρομολογητής σας να εμφανίζεται κατά τη διαδικασία του configuration (make config ή menuconfig ή xconfig):

– Αντιγράψτε ένα από τα υπάρχοντα *config IO_SCHED_...* blocks και επικολλήστε το ως *config IOSCHED_CLOOK* block, πριν το ”*choice*” block.
– Αλλάξτε το όρισμα της παραμέτρου tristate σε ”*CLOOK I/O scheduler*”. Αν θέλετε, καλό θα ήταν να αλλάξετε κατάλληλα και το βοηθητικό κείμενο. Επίσης, προσθέστε τα παρακάτω στο ”*choice*” block:
config DEFAULT_CLOOK
bool ”CLOOK” if IOSCHED_CLOOK=y

3. Αντικαταστήστε το περιεχόμενο του πεδίου *elevator-name* της δομής με το όνομα του νέου δρομολογητή (*clook*) και τροποποιήστε κατάλληλα τα ορίσματα στα macros:
MODULE_AUTHOR και *MODULE_DESCRIPTION*.

4. Προχωρήστε στο βασικό μέρος της εργασίας: Αλλάξτε το αρχείο *clook-iosched.c* έτσι ώστε οι αιτήσεις I/O να εξυπηρετούνται με τον τρόπο που περιγράφηκε στην παράγραφο 1. Πέραν των υπολοίπων, αλλάξτε και τα ονόματα των συναρτήσεων ώστε να ξεκινάνε με *clook* (και όχι με *noop*). Συνίσταται να χρησιμοποιήσετε την υλοποίηση διασυνδεμένων λιστών που παρέχεται στον kernel για την ουρά του δρομολογητή. Μελετήστε τη δομή *elevator_ops* στο αρχείο *include/linux/elevator.h* για να δείτε ποιες συναρτήσεις θα χρειαστεί να υλοποιήσετε. Θα χρειαστεί, κατ’ ελάχιστον, να υλοποιήσετε τις:

   • *elevator_init_fn()*
   • *elevator_add_req_fn()*
   • *elevator_dispatch_fn()*
   • *elevator_exit_fn()*
   • *elevator_queue_empty_fn()*
   • Πιθανότατα και άλλες, όπως την *elevator_former_req_fn()* και *elevator_latter_req_fn()*

   Υπενθυμίζεται ότι για περισσότερες πληροφορίες σε ότι αφορά τη δομή και τις συναρτήσεις, μπορείτε να δείτε την παράγραφο 4.1 του αρχείου *Documentation/block/biodoc.txt*.

5. Εκτυπώστε μηνύματα με χρήση της *printk()*, ώστε να είστε σε θέση να επιβεβαιώσετε ότι ο αλγόριθμός σας δουλεύει, και μάλιστα σωστά. Πιο συγκεκριμένα:

   • Κάθε φορά που μία αίτηση I/O εισάγεται στην ουρά του δρομολογητή, τυπώστε ένα μήνυμα με την παρακάτω μορφή:

[CLOOK] add <action> <sector>

όπου <*action*> είναι είτε *R* (για τις αναγνώσεις), είτε *W* (για τις εγγραφές) και <*sector*> είναι το 1ο sector που θα χρειαστεί να προσπελαστεί για την ενέργεια.

- Κάθε φορά που μια αίτηση I/O προωθείται προς τη Dispatch Queue, τυπώστε ένα μήνυμα με την παρακάτω μορφή:

[CLOOK] dsp <action> <sector>

Ως συνήθως, τα μηνύματα, όταν ενεργοποιηθεί ο δομολογητής C-LOOK, θα εμφανίζονται στο αρχείο */var/log/messages*, ενώ θα μπορείτε να τα δείτε και με την εντολή *dmesg*.

3.2 Μεταγλώττιση Δρομολογητή Αιτήσεων I/O C-LOOK

Για να μεταγλωττίσετε τον δρομολογητή αιτήσεων με τη μορφή ενός module και να μπορεί να χρησιμοποιηθεί από το νέο πυρήνα, κάντε τα ακόλουθα βήματα:

- Στον κατάλογο -dev γράψτε:
  sudo make xconfig (ή *sudo make menuconfig* αν δε χρησιμοποιείτε γραφικό περιβάλλον).
- Φορτώστε το configuration που σας δώσαμε στις ασκήσεις 1 και 3.
- Πηγαίνετε στο *Enable the block layer –> IO Schedulers* και επιλέξτε το νέο δρομολογητή, ορίζοντας ταυτόχρονα ότι πρέπει να μεταγλωττιστεί ως module. **ΠΡΟΣΟΧΗ:** Μην πειράξετε την προκαθορισμένη επιλογή δρομολογητή I/O.
- Σώστε τις επιλογές σας.
- Μεταγλωττίστε και εγκαταστείστε τον πυρήνα κατά τα γνωστά. Εφόσον οι αλλαγές σας περιορίζονται στο module, μετά την πρώτη μεταγλώττιση θα χρειάζεται ελάχιστος χρόνος για να μεταγλωττίσετε τυχόν αλλαγές. Επίσης, μετά την 1η φορά τυπικά δε θα χρειάζονται τα βήματα μετά το *sudo make modules_install*, αν και είναι ασφαλές να τα εκτελείτε.
- Επανεκκινήστε το σύστημα διαλέγοντας τον πυρήνα -dev.

3.3 Ενεργοποίηση και Χρήση Δρομολογητή Αιτήσεων I/O C-LOOK

3.3.1 Χρήση Ξεχωριστού Ιδεατού Δίσκου για τους Σκοπούς της Πειραματικής Αξιολόγησης

Ο πειραματισμός με τα τμήματα του Linux που αφορούν το σύστημα αρχείων είναι – όπως είναι αυτονόητο – επικίνδυνος, καθώς οποιοδήποτε λάθος ενδεχομένως να οδηγήσει σε μη αναστρέψιμη κατάρρευση του συστήματος αρχείων. Για το λόγο αυτό, θα προσθέσουμε έναν παραπάνω εικονικό δίσκο στο σύστημά μας, ειδικά για τον πειραματισμό με το δρομολογητή αιτήσεων I/O.

Κατεβάστε το αρχείο που σας δίνεται στο e-class (Έγγραφα, Εργασίες), τοποθετήστε το στον κατάλογο όπου υπάρχουν και τα υπόλοιπα αρχεία της εικονικής μηχανής σας, και αποσυμπιέστε το (*tar -xzf testdisk.tar.gz*). Ο εικονικός δίσκος έχει μέγεθος 1 GB και είναι αρχικά άδειος. Μπορείτε να τον γεμίσετε με ό,τι νομίζετε για την πειραματική σας αξιολόγηση.

Ακολούθως, θα προσαρτήσουμε το νέο εικονικό δίσκο στην εικονική μηχανή.

- Επιλέξτε την εικονική μηχανή και πατήστε *Edit virtual machine settings*.
- Στην καρτέλα *Hardware* και κάτω από τα *Devices* πατήστε *Add*.
- Επιλέξτε *Hard Disk* και κατόπιν *Use an existing virtual disk*.
- Επιλέξτε το όνομα του αρχείου που περιέχει τον εικονικό δίσκο.
- Πλέον στην καρτέλα του *Hardware* θα πρέπει να έχει εμφανιστεί ο νέος εικονικός δίσκος. Επιλέξτε τον και πατήστε στο κουμπί *Advanced*.
- Επιλέξτε ο νέος δίσκος να φαίνεται ως SCSI 0:1. Αυτό σημαίνει ότι στο εσωτερικό του Linux ο δίσκος αυτός θα είναι ορατός μέσω του ψευδοαρχείου συσκευής */dev/sdb1*.

Ο νέος δίσκος θα πρέπει πλέον να προσαρτηθεί κάτω από κάποιο σημείο του δέντρου αρχείων, έτσι ώστε να είναι διαθέσιμος στις εφαρμογές:

- Πηγαίνετε κάτω από τον κατάλογο */media* και δημιουργήστε έναν κατάλογο με όνομα *test_disk*.

```
cd /media
sudo mkdir test_disk
```

- Κάθε φορά που επανεκκινείτε το σύστημα, θα πρέπει να προσαρτάτε τον εικονικό δίσκο κάτω από τον κατάλογο */media/test_disk*.

sudo mount /dev/sdb1 /media/test_disk

Από αυτό το σημείο και πέρα, είναι δυνατόν να γράψετε στον ή να διαβάσετε από τον εικονικό δίσκο γράφοντας ή διαβάζοντας κάτω από τον κατάλογο */media/test_disk*.

3.3.2 Φόρτωση του Module

Μετά από κάθε επανεκκίνηση και προκειμένου το module σας να είναι έτοιμο προς χρήση, θα πρέπει να το φορτώσετε. Αυτό γίνεται με την εντολή:

sudo modprobe clook-iosched

Όταν θελήσετε να απενεργοποιήσετε το module και να το απομακρύνετε από τη μνήμη, εκτελέστε την εντολή:

sudo modprobe -r clook-iosched

Η διαδικασία είναι πρακτικά ισοδύναμη με τη χρήση *insmod* και *rmmod* που είχαμε δει στην 1η εργασία. Η διαφορά είναι ότι τώρα η μεταγλώττιση του module έχει γίνει στα πλαίσια της μεταγλώττισης του πυρήνα. Το module είναι, για το λόγο αυτό, γνωστό στο σύστημα με το όνομά του και δε χρειάζεται να προσδιορίσουμε το όνομα αρχείου (όπως γίνεται με την *insmod*).

3.3.3 Ενεργοποίηση του Δρομολογητή I/O C-LOOK για τον Πειραματικό Δίσκο

Όταν θέλετε να χρησιμοποιήσετε τον δρομολογητή C-LOOK για τη δρομολόγηση των αιτήσεων I/O προς τον πειραματικό εικονικό δίσκο, θα πρέπει να κάνετε αντίστοιχα βήματα με αυτά που είχαμε περιγράψει στην 1η εργασία για τον δρομολογητή *noop*.

Ελέγξετε τα I/O scheduling modules που είναι διαθέσιμα για τη συσκευή /dev/sdb. Αυτό μπορεί να πραγματοποιηθεί με την εντολή *cat /sys/block/sdb/queue/scheduler*. Μία ενδεικτική έξοδος από την εκτέλεση της εντολής είναι η παρακάτω:

```
noop deadline [cfq] clook
```

Όπως μπορούμε να παρατηρήσουμε, υπάρχουν τέσσερις διαθέσιμοι schedulers. Το όνομα του scheduler που χρησιμοποιείται είναι αυτό που περικλείεται από τις αγκύλες. Για να ενεργοποιήσετε το δικό σας scheduler θα πρέπει να εκτελέσετε την εντολή

sudo bash -c ’echo clook > /sys/block/sdb/queue/scheduler’

Μετά από αυτή την εντολή, η έξοδος από την εκτέλεση της εντολής

cat /sys/block/sdb/queue/scheduler

θα πρέπει να είναι η ακόλουθη:

```
noop deadline cfq [clook]
```

Όπως παρατηρείτε, η διαχείριση του I/O scheduling θα πραγματοποιείται πλέον από τον δρομολογητή *C-LOOK*.

Κάθε φορά που τελειώνετε την πειραματική αξιολόγηση, θα πρέπει *οπωσδήποτε* να επαναφέρετε τον αρχικό scheduler για τη συσκευή /dev/sdb. Γι' αυτό το σκοπό, εκτελέστε την εντολή

sudo bash -c 'echo cfq > /sys/block/sdb/queue/scheduler'

3.4 Δεύτερο Μέρος: Πειραματική Αξιολόγηση

Για να επιβεβαιώσετε τη σωστή λειτουργία του νέου δρομολογητή αιτήσεων I/O σχεδιάστε και εκτελέστε τα κατάλληλα πειράματα, τα οποία θα δείξουν ότι ο δρομολογητής σας δουλεύει με τον αναμενόμενο τρόπο. Μη βασιστείτε απλά σε προγράμματα που εκτελούν I/O με τυχαίο τρόπο.

Θυμηθείτε ότι το λειτουργικό σύστημα υλοποιεί και χρησιμοποιεί cache για το σύστημα αρχείων, η οποία μάλιστα βρίσκεται σε παραπάνω επίπεδο από τον δρομολογητή αιτήσεων I/O. Με άλλα λόγια, ο δρομολογητής σας "βλέπει" μόνο τις αιτήσεις που "περνάνε" (δεν μπορούν να εξυπηρετηθούν) από την cache. Για παράδειγμα, η 2η ανάγνωση από ένα sector θα εξυπηρετηθεί σχεδόν πάντα από την cache. Οι εγγραφές δεν είναι τόσο προβληματικές, καθώς τελικά θα χρειαστεί να κατευθυνθούν στο δίσκο – έστω και καθυστερημένα, έστω και συνδυασμένες με άλλες εγγραφές στο ίδιο sector.

Ο δρομολογητής σας θα πρέπει φυσικά να δουλεύει σωστά τόσο για αναγνώσεις όσο και για εγγραφές. Ενδεχομένως να είναι καλή ιδέα να δοκιμάσετε τις αναγνώσεις σε ένα αρκετά μεγάλο αρχείο.

4 Πακετάρισμα των Αλλαγών – Δημιουργία του προς Υποβολή Συνημμένου Αρχείου

4.1 Απομόνωση των Αλλαγών στον Κώδικα του Λειτουργικού

Προφανώς δεν είναι ιδιαίτερα πρακτικό να παραδώσετε τις αλλαγές στέλνοντας όλο το νέο κώδικα του λειτουργικού, δεδομένου και ότι τα αρχεία τα οποία στην πραγματικότητα θα έχετε πειράξει είναι ελάχιστα. Η εφαρμογή *diff* αναλαμβάνει να αναγνωρίσει τις αλλαγές που κάνατε στον πηγαίο κώδικα του λειτουργικού, συγκρίνοντάς τον με το αντίγραφο του αρχικού κώδικα που έχουμε κρατήσει στον κατάλογο /usr/src/linux-3.14.62-orig.

Κατ' αρχήν πηγαίνετε στον κατάλογο /usr/src/linux-3.14.62-dev (cd /usr/src/linux-3.14.62-dev) και δώστε την εντολή *make distclean*. Με τον τρόπο αυτό σβήνονται όλα τα αρχεία (.o, εκτελέσιμα κλπ) που δημιουργήθηκαν κατά τη μεταγλώττιση. Κάνετε το ίδιο – για καλό και για κακό – και στον κατάλογο /usr/src/linux-3.14.62-orig. **ΜΗΝ ΞΕΧΑΣΕΤΕ ΑΥΤΑ ΤΑ 2 ΒΗΜΑΤΑ!!!**

Πλέον είστε έτοιμοι να "συγκρίνετε" τους 2 καταλόγους. Πηγαίνετε στο /usr/src και από εκεί δώστε την εντολή

sudo bash -c 'diff -ruN linux-3.14.62-orig linux-3.14.62-dev > patch_4'

Το πρόγραμμα diff εντοπίζει αναδρομικά τις αλλαγές μεταξύ των καταλόγων linux-3.14.62-orig και linux-3.14.62-dev. Το τελικό αποτέλεσμα αποθηκεύεται στο αρχείο patch_4 το οποίο δημιουργείται μέσα στον κατάλογο από τον οποίο καλέσατε την *diff* (δηλαδή τον κατάλογο /usr/src). Αν θέλετε, διαβάστε το αρχείο patch_4 (είναι ένα αρχείο κειμένου) και προσπαθήστε να καταλάβετε τη δομή του.

Αν θέλετε να επιβεβαιώσετε ότι το patch φτιάχτηκε σωστά, μπορείτε να κάνετε τα ακόλουθα: Η εφαρμογή *patch* μπορεί να πάρει ένα patch και να εφαρμόσει τις αλλαγές που περιγράφει το patch σε έναν κατάλογο πηγαίου κώδικα. Κατόπιν, μεταβείτε στον κατάλογο /usr/src/linux-3.14.62-orig και δώστε την εντολή

patch -p1 –dry-run < ../patch_4

Με την εντολή αυτή θα γίνει προσομοίωση εφαρμογής στον κατάλογο που βρισκόμαστε του patch που περιέχεται στο αρχείο patch_4 (το οποίο patch_4 είναι τοποθετημένο στον στον αμέσως προηγούμενο κατάλογο, γι' αυτό και το ../). Η παράμετρος *–dry-run* λέει στην εφαρμογή patch να προσποιηθεί ότι εφαρμόζει το patch χωρίς να κάνει στην πραγματικότητα οποιεσδήποτε αλλαγές στα

αρχεία. Το patch θα πρέπει να εφαρμοστεί “καθαρά”, δε θα πρέπει δηλαδή να σας εμφανιστούν μηνύματα λάθους. Δώστε προσοχή σε αυτό το βήμα, γιατί προϋπόθεση για τη βαθμολόγηση κάθε άσκησης είναι το patch που θα μας στείλετε να μπορεί να εφαρμοστεί “καθαρά”.

4.2 Πακετάρισμα των Προς Υποβολή Στοιχείων σε Αρχείο

Το τελευταίο βήμα είναι να πακετάρετε όλα τα αρχεία που πρέπει να μας στείλετε σε ένα αρχείο. Πηγαίνετε στον home directory του λογαριασμού σας. Φτιάξτε εκεί με την εντολή *mkdir* έναν κατάλογο με όνομα project_4_AEM_Melwn_omadas>. Για παράδειγμα, η ομάδα με μέλη με ΑΕΜ 123 234 345 θα πρέπει να δώσει την εντολή *mkdir project_4_123_234_345*. Αντιγράψτε (με την εντολή *cp*) το patch που φτιάξατε σε αυτό τον κατάλογο με την εντολή. Π.χ. η ομάδα με μέλη με ΑΕΜ 123 234 345 θα δώσει την εντολή: *cp /usr/src/patch_4 /project_4_123_234_345*.

Κάντε το ίδιο και με τα αρχεία που απαρτίζουν τον κώδικα της εφαρμογής ή των εφαρμογών που αναπτύξατε για την πειραματική αξιολόγηση. Δώστε ένα παράδειγμα εξόδου του δρομολογητή που αποδεικνύει ότι ο δρομολογητής δουλεύει σωστά. Τέλος, συμπεριλάβετε το αρχείο κειμένου στο οποίο θα περιγράφετε τα δεδομένα που βάλατε στον εικονικό δίσκο, τα σενάρια που εκτελέσατε κατά την πειραματική αξιολόγηση, θα περιλαμβάνετε την ανάλυση της εξόδου του δρομολογητή και τις παρατηρήσεις σας.

Επίσης, δημιουργήστε μέσα στον κατάλογο ένα αρχείο με όνομα README.txt στον οποίο θα γράψετε τα ονόματα, ΑΕΜ και e-mail των μελών της ομάδας, καθώς και αναλυτικές οδηγίες για τη μεταγλώττιση της πειραματικής εφαρμογής. Μπορείτε να συμπεριλάβετε στο αρχείο και οτιδήποτε άλλο θέλετε να έχουμε υπόψη μας κατά τη διόρθωση.

Ακολούθως, πηγαίνετε στον πηγαίο κατάλογο του λογαριασμού σας και από εκεί δώστε την εντολή

tar -cvf project_4_AEM_Melwn_omadas>.tar project_4_AEM_Melwn_omadas>

Για παράδειγμα, η ομάδα με μέλη με ΑΕΜ 123 234 345 θα δώσει την εντολή

tar -cvf project_4_123_234_345.tar project_4_123_234_345

Με την εντολή tar θα πακεταριστούν τα περιεχόμενα του καταλόγου σε ένα αρχείο με κατάληξη .tar.

Αφού φτιαχτεί το αρχείο, ελέγξτε το μέγεθός του. Αν το μέγεθος είναι αδικαιολόγητα μεγάλο (θυμηθείτε ότι το αρχείο στην ουσία περιέχει μερικά πολύ μικρά αρχεία κειμένου) έχετε κάνει κάτι λάθος (το πιθανότερο είναι ότι έχετε ξεχάσει το βήμα make distclean ή έχετε συμπεριλάβει και εκτελέσιμα αρχεία).

Τέλος, δώστε την εντολή:

bzip2 project_4_AEM_Melwn_omadas>.tar

Θα δημιουργηθεί ένα αρχείο με κατάληξη .bz2, το οποίο περιέχει συμπιεσμένο το αρχείο με κατάληξη .tar. Για παράδειγμα, η ομάδα 124 234 345 θα δώσει την εντολή:

bzip2 project_4_123_234_345.tar

Το αρχείο με κατάληξη .bz2 είναι αυτό που θα πρέπει να επισυνάψετε κατά την παράδοση της άσκησης στο e-class