Κωνσταντίνος Κανελλάκης
Αριθμός Μητρώου: 1115202000064
Εργασία 2-Λειτουργικά Συστήματα

Για την ανάλυση της λειτουργικότητας του κώδικα της δεύτερης εργασίας του μαθήματος θα γίνει επεξήγηση ανά βήμα (1,2,3) όπως και χωρίζεται στην εκφώνηση.

Βήμα 1 – Εκτυπώστε τον πίνακα σελίδων

Η vmprint() ορίστηκε στα κατάλληλα αρχεία και με τα κατάλληλα ορίσματα όπως ορίζει η εκφώνηση ενώ κλήθηκε επίσης στο τέλος της exec(). Χρησιμοποιήθηκε και μια βοηθητική συνάρτηση print() η οποία θα καλείται αναδρομικά για την εμφάνιση όλων των στοιχείων του pagetable που θέλουμε. Για την ανάπτυξη του κώδικα χρησιμοποιήθηκε ως πρότυπο η freewalk() η οποία προσπελαύνει όλα τα στοιχεία του pagetable που δίνουμε ως όρισμα και μηδενίζει όλα τα στοιχεία του page table που είναι παρών και χρησιμοποιούνται για διάβασμα/εγγραφή/εκτέλεση. Αν πέσει σε κάποιο στοιχείο-φύλλο βγάζει ερρορ καθώς αυτό θα έπρεπε να είχε διαγραφεί πιο πριν. Αντίστοιχα η vmprint() διασχίζει τον pagetable που της δίνουμε σαν όρισμα και εμφανίζει όλα τα στοιχεία που είναι παρών εμφανίζοντας τον κατάλληλο αριθμό κενών και τελείων ανάλογα με το επίπεδο που βρίσκεται. Αν επίσης αναφέρεται και σε κάποιο pagetable χαμηλότερου επιπέδου καλεί αναδρομικά τη συνάρτηση print() έτσι ώστε να εμφανίσει τα στοιχεία και αυτού καθώς είναι μέρος του αρχικού pagetable. Για να βρούμε την διεύθυνση της φυσικής μνήμης χρησιμοποιούμε την ορισμένη συνάρτηση PTE2PA() βάζοντας μάσκα στην εικονική μνήμη pte. Το φορ εκτελείται για κάθε κλήση pagetable 512 φορές, όσα και τα στοιχεία ενός pagetable.

Βήμα 2 – Απομακρύνετε την εκχώρηση μνήμης από την sbrk()

Για την ολοκκλήρωση αυτού του βήματος ακολούθησα κατά γράμμα τις οδηγίες της εκφώνησης ενώ επίσης όρισα έναν νέο pointer τον p έτσι ώστε να είναι πιο ευδιάκριτος ο κώδικας. Αφού έκανα σχόλιο την κλήση της growproc() και αύξησα το μέγεθος του myproc() κατά n, είδα το αντίστοιχο έρρορ που αναφέρεται στην εκφώνηση να εμφανίζεται. Ωστόσο αργότερα για το βήμα 3 άλλαξα και πάλι την sbrk().

Κωνσταντίνος Κανελλάκης
Αριθμός Μητρώου: 1115202000064
Εργασία 2-Λειτουργικά Συστήματα

Βήμα 3 – Νωχελική εκχώρηση (lazy allocation)

Για να διαχειριστώ το πρόβλημα που παρουσιαζόταν όταν πήγαινα να εκτελέσω την εντολή echo hi τροποποίησα την συνάρτηση usertrap() έτσι ώστε να διαχερίζεται τα σφάλματα σελίδας όταν τα εντοπίζει. Όταν δηλαδή η r_scause() έχει την τιμή 13 ή 15. Επίσης στη συνάρηση αυτή χρησιμοποίησα και την μεταβλητή cause η οποία περιέχει το αποτέλεσμα της r_scause() ώστε ο κώδικας να είναι πιο ευδιάκριτος. Έτσι δημιούργησα ένα else if για τις περιπτώσεις όπου το λειτουργικό καταλήγει σε σφάλμα σελίδας, έτσι ώστε να το διαχειριστούμε κατάλληλα αφού δημιουγείται σκόπιμα από εμάς έτσι ώστε να δεσμεύσουμε τον χώρο που το πρόγραμμα χρειάζεται, όταν πραγματικά τον χρειαστεί και για πράγματα που πραγματικά θα χρησιμοποιήσεικ. Έτσι, όταν παρουσιάζεται σφάλμα σελίδας καλούμε την συνάρτηση lazy() η οποία είναι υπεύθυνη για την δέσμευση του απαιτούμενου χώρου και για την αντιστοίχση αυτού με τις εικονικές διευθύνσεις. Σε περίπτωση λάθους κατά την κλήση αυτής της συνάρτησης η εργασία p τερματίζεται.

Η συνάρτηση lazy όπως είναι φανερό αποτελεί την βάση της νωχελικής εκχώρησης. Αρχικά γίνεται έλεγχος για να διαπιστωθεί αν η εικονική διεύθυνση στην οποία προκλήθηκε σφάλμα σελίδας βρίσκεται υψηλότερα από οποιαδήποτε άλλη ανατεθείσα εικονική διεύθυνση ή αν βρίσκεται κάτω από την στοίβα της διεργασίας. Σε αυτές τις περιπτώσεις επιστρέφουμε 0 καθώς δεν επιτρέπεται να γράφουμε πέρα από τα όρια της στοίβας της διεργασίας($2^η$ συνθήκη) και να αφήνουμε κενά μη ανατεθείσων εικονικών μνημών($1^η$ συνθήκη).

Σε διαφορετική περίπτωση συνεχίζεται η δέσμευση μνήμης μέσω της kalloc() η οποία αν επιστρέψει 0 σημαίνει ότι δεν κατάφερε να δεσμεύσεις χώρο μνήμης και άρα τερματίζουμε τη συνάρτηση επιστρέφοντας 0. Σε αντίθετη περίπτωη η kalloc() δεσμεύει 1 σελίδα 4KB φυσικής μνήμης και επιστρέφει την διέυθυνση της έτσι ώστε εμείς να την αρχικοποιήσουμε με την βοήθεια της memset() με 0. Ύστερα χρησιμοποιούμε την mappages έτσι ώστε να αντιστοιχήσουμε τις εικονικές θέσεις μνήμης με τις φυσικές. Σε περίπτωση που η αντιστοίχιση αποτύχει τότε ελευθερώνουμε την φυσική μνήμη που δεσεμεύσαμε νωρίτερα και επιστρέφουμε 0.

Κωνσταντίνος Κανελλάκης
Αριθμός Μητρώου: 1115202000064
Εργασία 2-Λειτουργικά Συστήματα

Η ανάπτυξη του κώδικα για την συνάρτηση lazy() έγινε με την βοήθεια του κώδικα της uvmalloc().

Επίσης προσαρμόστηκε η uvmunmap() έτσι ώστε να μην προκαλέσει panic σε περίπτωση που κάποιες εικονικές διευθύνσεις δεν έχουν αντιστοιχηθεί ακόμα(θα αντιστοιχηθούν αργότερα από την lazy()). Αντίστοιχα και για τους ίδιους λόγους εργαζόμαστε και στην uvmcopy().

Μία ακόμη σημαντική αλλαγή είναι αυτή στην walkaddr(). Σε περίπτωση που η walk() δεν βρει κάποια αντιστοιχία σε φυσική μνήμη για την εικονική διεύθυνση va είτε δεν είναι έγκυρη(παρών) είτε δεν έχει αντιστοιχηθεί σε χρήση user(ενώ η συνάρτηση αυτή αφορά μόνο σελίδες χρήστη και άρα αν δεν έχει αντιστοιχηθεί με σελίδα χρήστη πρέπει να την αντιστοιχήσουμε εμείς) τότε καλούμε την συνάρτηση lazy() για να δεσμεύσουμε τον απαραίτητο χώρο και να κάνουμε όποια άλλη ενέργεια χρειάζεται(αναφέρονται στην περιγραφή της lazy()). Σε περίπτωση που η lazy() αποτύχει για κάποιον λόγο επιστρέφεται 0.

Τέλος, πρέπει να αναφερθεί ότι η sbrk() τελικά τροποποιήθηκε και πάλι σε σχέση με το βήμα 2 για να αντιμετωπίζει και το αρνητικό όρισμα n. Όταν το όρισμα n είναι αρνητικό σημαίνει ότι πρέπει να ελευθερώσουμε χώρο. Έτσι, βάσει και της growproc() όταν το όρισμα n είναι αρνητικό καλούμε την uvmdealloc με τα κατάλληλα ορίσματα και το αποτέλεσμα της αποθηκεύεται ως το νέο μέγεθος της διεργασίας. Σε αντίθετη περίπτωση απλά αυξάνουμε το μέγεθος της διεργασίας «εικονικά», χωρίς δηλαδή να δεσεμεύσουμε χώρο καθώς αυτό θα γίνει αργότερα όπως περιγράφηκε πιο πάνω.

Όλα τα τεστ παιρνούνται επιτυχώς τρέχοντας είτε το make grade είτε εκτελώντας τις εντολές lazytests,usertests εντός του terminal του xv6.