ΠΑΝΑΓΙΩΤΗΣ ΚΟΝΤΟΕΙΔΗΣ

ΑΜ 1115201900266

ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ /ΠΕΡΙΟΔΟΣ 2021-2022

ΕΡΓΑΣΙΑ 2^{Η}

Αρχικά, για την υλοποίηση της Αντιγραφής κατά την Εγγραφή θα πρέπει να γίνουν κάποιες προσθήκες στις ήδη υπάρχουσες συναρτήσεις στο αρχείο kalloc.c. Συγκεκριμένα, προστέθηκε μια δομή ref_counter που περιέχει, μια δομή spinlock και έναν πίνακα με αριθμό θέσεων PHYSTOP/PGSIZE, όσο είναι και ο μέγιστος επιτρεπτός αριθμός των Physical Addresses(PA) που μπορούν να ανατεθούν. Κάθε θέση του πίνακα αυτού περιέχει τον αριθμό των αναφορών για κάθε PA.

Για την τροποποίηση του αριθμού των αναφορών(ref count) έχουν σχηματιστεί οι κατάλληλες συναρτήσεις init_ref_counter, increase_ref_counter, decrease_ref_counter και return_ref_counter. Μέσα σε κάθε συνάρτηση από αυτές γίνεται acquire και release του ref_counter.lock σε περίπτωση που έχουμε race condition.

Στην συνάρτηση freerange όπου και γίνεται η αρχική ανάθεση όλων των σελίδων στο σύστημα, όλες οι σελίδες με την κλήση της init_ref_counter αρχικοποιούνται με ref count = 1 . Στην συνέχεια, με την κλήση της kfree δεν αφαιρείται η σελίδα για την οποία κλήθηκε η συνάρτηση, για να την προσθέσει στην λίστα των διαθέσιμων σελιδών του συστήματος, αλλά ελέγχει πρώτα αν το ref count της σελίδας αυτής είναι >=2. Αν ισχύει αυτό τότε απλά μειώνει το ref count της σελίδας κατά ένα, αλλιώς κάνει αυτό που έκανε και πριν. Με αυτόν τον τρόπο εξασφαλίζεται ότι όλες οι αρχικοποιήσεις γίνονται σωστά και δεν θα υπάρχει κάποιο πρόβλημα με την αποδέσμευση. Στην συνάρτηση kalloc επίσης όλες οι σελίδες που ανατίθενται, αρχικοποιούνται με ref count =1.

Η λειτουργία της συνάρτησης uvmcopy είναι να αντιστοιχίζει τις φυσικές σελίδες του γονέα στο παιδί και όχι να αναθέτει καινούργιες. Για να γίνει αυτό πρέπει να αντιγραφούν το Page Table Entry (PTE) αλλά και το PA. Αφού πάρουμε το PTE του γονέα και ελέγξουμε ότι είναι Valid, κάνουμε reset τα flag bits PTE_W, τα οποία καθιστούν το αρχείο writable. Μετά κάνουμε map την παλιά σελίδα στην καινούργια και αντιγράφουμε τα flags από την παλιά στην καινούργια. Τέλος, κάνουμε reset τα flag bits PTE_W του γονέα και αυξάνουμε τον ref count του PA.

Όταν το σύστημα προσπαθεί να γράψει σε μία σελίδα που είναι COW επειδή η σελίδα αυτή είναι READ-ONLY θα προκαλέσει ένα Page Fault. Όταν προκληθεί αυτό, η συνάρτηση usertrap θα το ελέγξει το Page Fault και θα καλέσει την συνάρτηση cow_handler.

Η cow_handler χρησιμοποιείται σε δύο περιπτώσεις:

Η πρώτη περίπτωση είναι όταν καλείται από την usertrap όπου και δέχεται σαν ορίσματα το VA και το Page table από τα οποία προκλήθηκε το Page Fault. Εκεί ελέγχεται αρχικά, αν το VA είναι μεγαλύτερο της μέγιστης τιμής που μπορεί, αν το PTE που επιστρέφει η walk είναι μηδέν, αν το PA που προκύπτει από το PTE είναι μηδέν και αν το PA έχει το σωστό μέγεθος και δεν ξεπερνά τα όρια της μνήμης που έχουν οριστεί για τα PA. Ύστερα, ελέγχεται αν το PTE είναι COW PTE, δηλαδή θα ελέγχονται τα flag bits που δηλώνουν αν το PTE είναι Valid, User και Read-only. Στην συνέχεια, καλείται η kalloc που επιστρέφει έναν δείκτη στην καινούργια και κάνουμε reset τα flag bits της καινούργιας σελίδας, έτσι ώστε να μπορεί να είναι writable. Αντιγράφουμε τις πληροφορίες του παλιού PA στο καινούργιο και κάνουμε reset τα flag bits και του παλιού. Τέλος μειώνουμε τον ref_counter της παλιάς σελίδας εφόσον αναφέρεται σε αυτή ένας λιγότερος πίνακας σελίδων. Σε οποιαδήποτε περίπτωση λάθους η cow_handler επιστρέφει -1 και τερματίζεται η διεργασία.

Η δεύτερη περίπτωση είναι όταν καλείται από την copyout. Η copyout καλεί την walkaddr που επιστρέφει μέσω software ένα PA, κατευθείαν από ένα VA χωρίς να περάσει από τον MMU του hardware του RISCV, πράγμα οποίο δεν μπορεί να δημιουργήσει Page Fault σε περίπτωση που το PA είναι Read-Only. Επομένως, αντί να καλείται απευθείας η walkaddr καλείται η cow_handler. Αν η cow_handler επιστρέψει -1 τερματίζεται η διεργασία, αλλιώς αν μετά από τους ελέγχους που κάνει το ref_count του PA είναι ίσο με 1 το μόνο που χρειάζεται είναι να γίνει reset το flag bit PTE_W του PA για να γίνει writable. Οπότε η cow_handler επιστρέφει 0 και η copyout μπορεί να χρησιμοποιήσει την walkaddr κανονικά. Σε περίπτωση που ο ref_count του PA είναι μεγαλύτερος του 1 πραγματοποιείται η διαδικασία που θα γινόταν σε περίπτωση Page Fault και επιστρέφεται το παλιό PA.

Συμπεράσματα:

Όπως θα δείτε και εσείς τα cowtest και τα usertests εκτελούνται με επιτυχία. Όταν εκτελώ τα usertests με *make qemu* ενώ κάτω από όλα τα usertest τυπώνεται από δίπλα OK, στο τέλος τυπώνεται αυτό το μήνυμα:

FAILED -- lost some free pages 26218 (out of 26222)

Ακολουθώντας τις οδηγίες του κ. Πασκαλή, κατέληξα ότι το test, από τα usertets, στο οποίο δεν γίνεται σωστά η αποδέσμευση κάποιων σελιδών είναι στο sbrkfail. Το sbrkfail φαίνεται να δεσμεύει όλη την μνήμη και να ελέγχει αν στο τέλος έγινε clean up του last failed allocation.

Η αιτία για αυτό bug θεωρώ ότι είναι εσφαλμένη εφαρμογή του spinlock, που έχω ορίσει στο struct ref_count στο αρχείο kalloc.c , την οποία δοκίμασα με πολλούς και διαφορετικούς τρόπους.

Διαβάζοντας με το manual του riscv παρατήρησα τα παρακάτω:

«A race condition is a situation in which a memory location is accessed concurrently, and at least one access is a write»

«The outcome of a race depends on the exact timing of the two CPUs involved and how their memory operations are ordered by the memory system, which can make race-induced errors difficult to reproduce and debug»

Σύμφωνα με τα παραπάνω δοκίμασα να αλλάξω την γραμμή 285 του Makefile και από CPUS:=3 να το κάνω σε τρέχει μόνο σε έναν CPU, CPUS:=1 και με αυτήν την αλλαγή το bug αυτό δεν υπάρχει.

Συνοψίζοντας, θεωρώ πως η δικιά μου υλοποίηση της Αντιγραφής κατά την εγγραφή, είναι πλήρης και βελτιστοποιεί την δημιουργία των διεργασιών, ελαχιστοποιώντας τον αριθμό των νέων σελίδων που πρέπει να εκχωρηθούν για κάθε νεοδημιουργημένη διεργασία.