**ΕΠΛ425 Simple Client – Server Program**

**Περιγραφή προβλήματος:**

Σε αυτή την άσκηση υλοποιήσαμε ένα απλό πρόγραμμα Client – Server στο οποίο το ο Client επικοινωνεί απομακρυσμένα με τον server και ανταλλάσσουν διάφορα μνήματα. Για τους σκοπούς της εργασίας δημιουργήσαμε ένα πρόγραμμα Client το οποίο παράγει ένα συγκεκριμένο αριθμό από Users, οι οποίοι, ο καθένας από αυτούς ανήκει σε ένα ξεχωριστό threat και στέλνει ένα αριθμό από αιτήματα στον Server. Δημιουργήσαμε επίσης ένα ακόμα πρόγραμμα Server το οποίο αναμένει αιτήματα για να τα απαντήσει πίσω στους Users. Προσομοιώσαμε την επικοινωνία αυτή μεταξύ Client και Server χρησιμοποιώντας το cloud system της amazon. Τρέξαμε το πρόγραμμα αρκετές φορές έτσι ώστε να μπορέσουμε να παράξουμε μετρήσεις και γραφικές.

**Περιγραφή Υλοποίησης:**

**Υλοποίηση Client:**

Το πρόγραμμα Client δέχεται παράμετρούς το IP του Server, το Port και τον αριθμό επαναλήψεων (repetitions) για κάθε χρήστη. Στο πρόγραμμα αυτό υπάρχει μία σταθερά η οποία αντιπροσωπεύει τον αριθμό των χρηστών που θέλουμε να έχει το πρόγραμμα. Έτσι ώστε να μπορούμε να αλλάζουμε τον αριθμό τον χρηστών σε κάθε εκτέλεση. Το πρόγραμμα χρησιμοποιώ αυτή την σταθερά για να δημιουργήσει ένα αριθμό threats όπου το κάθε threat αντιπροσωπεύει ένα ξεχωριστό πρόγραμμα κάποιου χρήστη που θέλει να επικοινωνήσει με τον Server. Δηλώσαμε μέσω κάποιας άλλης σταθεράς τον αριθμό τον requests που θέλουμε να στέλνει ο κάθε χρήστης - threat. Οπότε κάθε νήμα που δημιουργείται στέλνει συνεχώς request στον server. Κάθε threat μόλις τερματίσει εκτελείτε ξανά μέχρι να συμπληρώσει ένα συγκεκριμένο αριθμό επαναλήψεων, όπου ο αριθμός αυτός εξαρτάται από την τιμή που έδωσε ο χρήστης στην γραμμή εντολών. Τέλος, για τον υπολογισμό των μετρήσεων, το πρόγραμμα είναι ρυθμισμένο να δημιουργεί ένα αρχείο για κάθε repetition, έτσι ώστε να μπορούμε να πάρουμε τις μετρικές μετά το τέλος της εκτέλεσης και να εξάγουμε διάφορα συμπεράσματα. Το κάθε αρχείο περιέχει τα requests όλων των χρηστών για το συγκεκριμένο repetition και το latency time που αντιστοιχεί σε αυτό το request.

**Υλοποίηση Server:**

Το πρόγραμμα Server δέχεται παράμετρούς, τον αριθμό του Port στο οποίο δέχεται τα αιτήματα. Στο πρόγραμμα αυτό υπάρχει μία σταθερά η οποία αντιπροσωπεύει τα milliseconds. Αυτή η σταθερά χρησιμοποιείται για να καθορίσει κάθε πόση ώρα να μετρά το throughput. Το πρόγραμμα μόλις εκτελεστεί περιμένει μέχρι να δεχτεί κάποιο αίτημα, και όταν το αίτημα αποσταλεί, το πρόγραμμα δημιουργεί ένα threat το οποίο είναι υπεύθυνο να εξυπηρετήσει το αίτημα του χρήστη και να του στείλει πίσω μία απάντηση. Η κάθε απάντηση που αποστέλλει το πρόγραμμα Server πίσω στον Client, αποτελείται από μία κεφαλίδα και ένα σώμα μηνύματος το οποίο περιέχει ένα μήνυμα με κάποιο τυχαίο size payload, το οποίο κυμαίνεται από 300kb – 2000kb για να γίνει πιο ρεαλιστική η επικοινωνία μεταξύ Server - Client. Τέλος, για τον υπολογισμό των μετρήσεων, το πρόγραμμα είναι ρυθμισμένο να δημιουργεί ένα αρχείο για κάθε repetition, έτσι ώστε να μπορούμε να πάρουμε τις μετρικές μετά το τέλος της εκτέλεσης και να εξάγουμε διάφορα συμπεράσματα. Το κάθε αρχείο περιέχει όλα τα throughput που υπολογίστηκαν και αντίστοιχα το average CPU load και memory utilization για το αντίστοιχο throughput.

**Υπολογισμός μετρήσεων, μέθοδος:**

Για τον υπολογισμό των μετρήσεων χρησιμοποιήσαμε τις τιμές που πάραξε το πρόγραμμα Client και Server και χρησιμοποιώντας κάποια script μπορέσαμε να κάνουμε τις εξής μετρικές:

**Latency Time vs Number of Users:** Τρέξαμε το πρόγραμμα 6 φορές (για 10, 15, 20, 25, 30, 35 users). Σε κάθε εκτέλεση υπολογίζαμε μέσω script από όλα τα αρχεία το άθροισμα του latency time και βρίσκαμε τον μέσο όρο του.

**Throughput vs Number of Users:** Τρέξαμε το πρόγραμμα 6 φορές (για 10, 15, 20, 25, 30, 35 users). Σε κάθε εκτέλεση υπολογίζαμε από όλα τα αρχεία το άθροισμα του Throughput και έπειτα τον μέσο όρο του.

**Memory Utilization vs Throughput:** Πήραμε ένα τυχαίο δήγμα εκτέλεσης για 10 users όπου το πρόγραμμα μας είχε υπολογίσει σε κάθε Throughput το αντίστοιχο Memory Utilization για κάθε repetition.

**Average CPU load vs Throughput:** Πήραμε ένα τυχαίο δήγμα εκτέλεσης για 10 users όπου το πρόγραμμα μας είχε υπολογίσει σε κάθε Throughput το αντίστοιχο Average CPU load για κάθε repetition.

**Παράδειγμα εκτέλεσης Server:**

javac -cp libs/jackson-all-1.9.0.jar src/reverse\_server\_multi\_threaded/\*.java

java -cp .:libs/jackson-all-1.9.0.jar src.reverse\_server\_multi\_threaded.ReverseServer 9999

**Παράδειγμα εκτέλεσης Client:**

javac -cp libs/jackson-all-1.9.0.jar src/reverse\_server\_multi\_threaded/\*.java

java -cp .:libs/jackson-all-1.9.0.jar src.reverse\_server\_multi\_threaded.ReverseClient [IP] [PORT] [REPETITIONS]