RFC

השרת והלקוח מתכתבים באמצעות שימוש ב - dictionary וב - jason, עד שהלקוח מגיע למחרוזת הגבוב (hash) הנכונה או שמקבל את ההודעה "success" מהשרת.
כך תיראה ההתכתבות ביניהם:

server - dictionary("type" :"new hash", "hash": hash_variable)
אחרי התחברות הלקוח לשרת, השרת ישלח את מחרוזת הגבוב שאליה הלקוח צריך למצוא התאמה.

client - dictionary("type": "cores", "cores_number" : core_num)
הלקוח בתגובה ישלח את מספר הליבות שלו אל השרת.

server - dictionary("type" : "range", "sequence_number" :sequence_num)
השרת ישלח אל הלקוח את המספר ממנו יצטרך להתחיל לחשב את מחרוזות הגבוב (יש הערה למטה).

במידה והלקוח מצא את המחרוזת הגבוב הנכונה:
client - dictionary("type": "operation", "status": "succeed", "hash_reverse": number_that_was_found)
בנוסף לכך ישלח הלקוח גם את אותו המספר שבעזרתו הגיע לפלט הגבוב הנכון.

השרת ישלח הודעה לכל הלקוחות להפסיק את עבודתם כי המספר נמצא:
server - dictionary("type": "operation", "status": "success")

במידה והלקוח לא מצא את המספר הוא ישלח:
client - dictionary("type": "operation", "status": "fail")

והשרת יחזיר לו טווח מספרים חדש:
server - dictionary("type" : "range", "sequence_number" :sequence_num)

כל לקוח יקבל טווח מספרים בהתאם למסבר הליבות שלו,

לדוג' לקוח בעל ליבה אחת אשר ה - sequence_num שהוא יקבל יהיה 0 יצטרך לחשוב את כל ה hash למחרוזות של המספרים מ - 0 עד 2.5 מיליון, לקוח בעל שני ליבות יצטרך לחשב בטווח של מהמספר שהוא קיבל עד עוד 5 מילון וכך הלאה.
הנוסחה לטווח: 2.5 * core_num
הערה: הלקוח צריך לחשב לבד את הטווח מה - suq number.