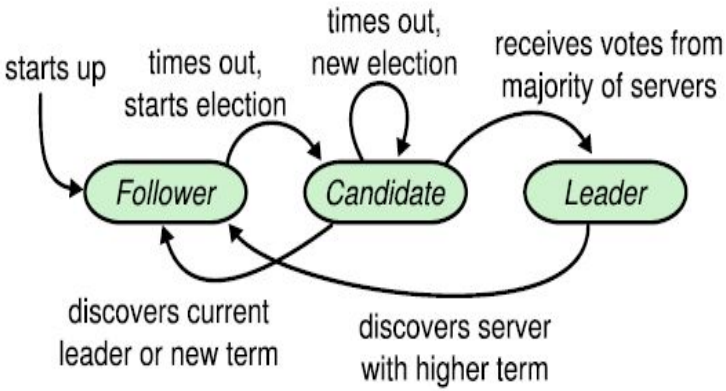
# Rancangan Fault Tolerance menggunakan RAFT Consensus Algorithm


| | |
|---|---|
| Faiq | 5115100007 |
| Naufal Pranasetyo | 5115100057 |
| Subhan Maulana | 5115100149 |

# Algoritma RAFT



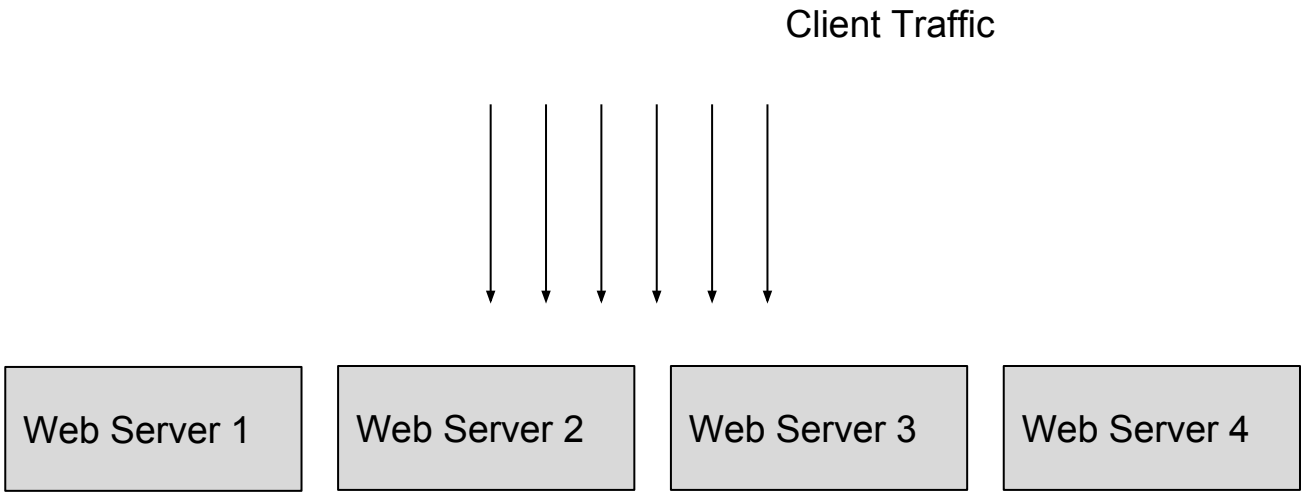

Raft termasuk ke dalam algoritma konsensus sistem terdistribusi seperti paxos.

Setiap cluster/term hanya memiliki satu leader node yang bertugas mengatur replikasi log.

Node dibagi menjadi 3 jenis yaitu follower, candidate, dan leader. Setiap follower akan menjadi candidate dan mengirimkan request election terhadap node lain.

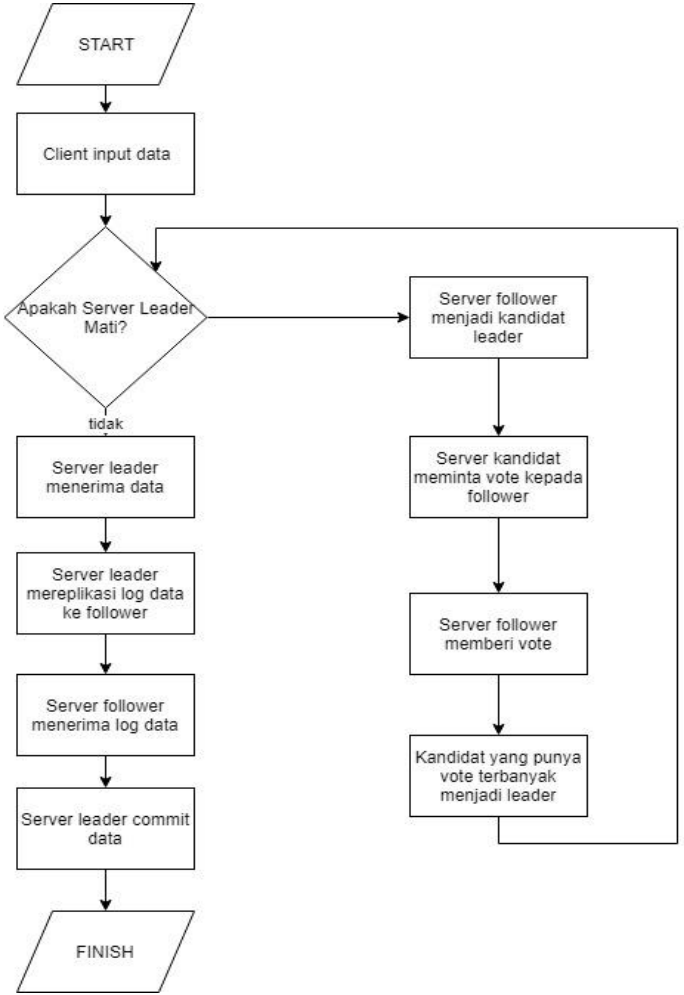
Melalui pemilihannya dari tiap node, akan terpilih leader node dari jumlah pilihan terbanyak

# Web Server



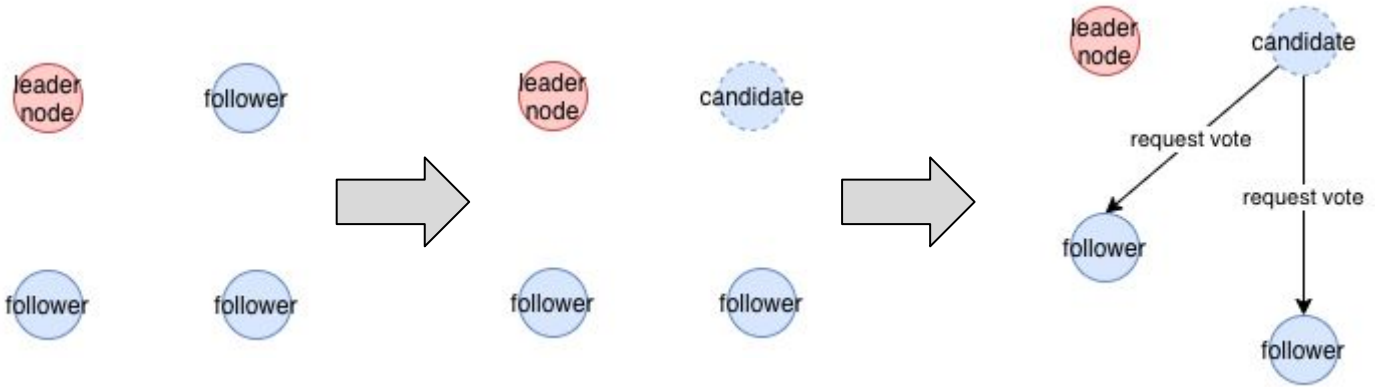

Dengan gambar arsitektur tersebut, buatlah sebuah rancangan pembagian beban dan fault tolerance protocol dengan menggunakan raft consensus protocol.

# Flowchart Fault Tolerance Dengan RAFT Algorithm



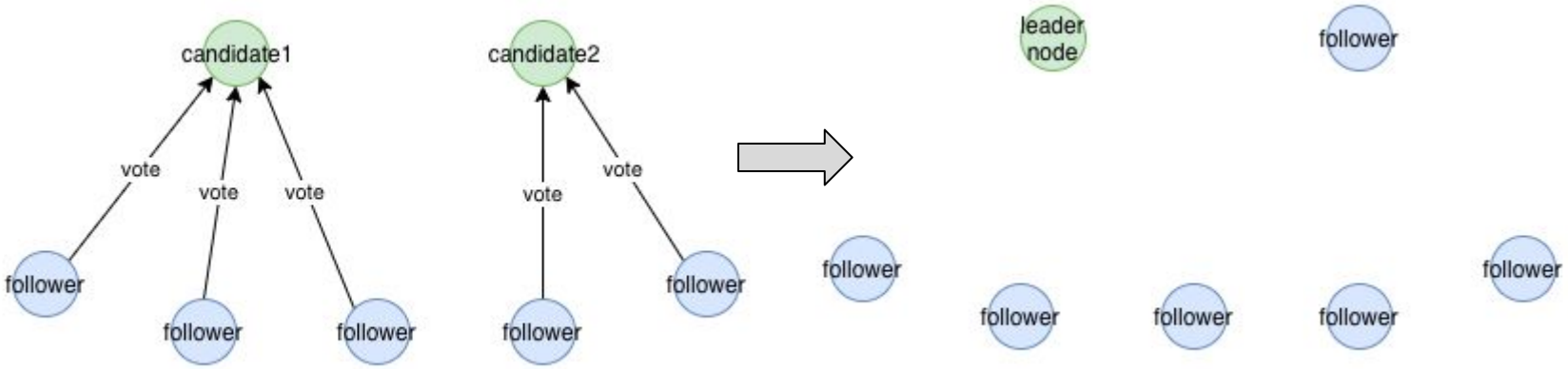
# Flowchart Rancangan Fault Tolerance

1. Ketika node server leader mati, maka node yang lain akan melakukan Consensus Protocol.
2. Setelah beberapa waktu (timeout), node pertama yang menyelesaikan timeout akan menjadi kandidat.
3. Kandidat akan request node yang lain untuk vote.



# Flowchart Rancangan Fault Tolerance (2)

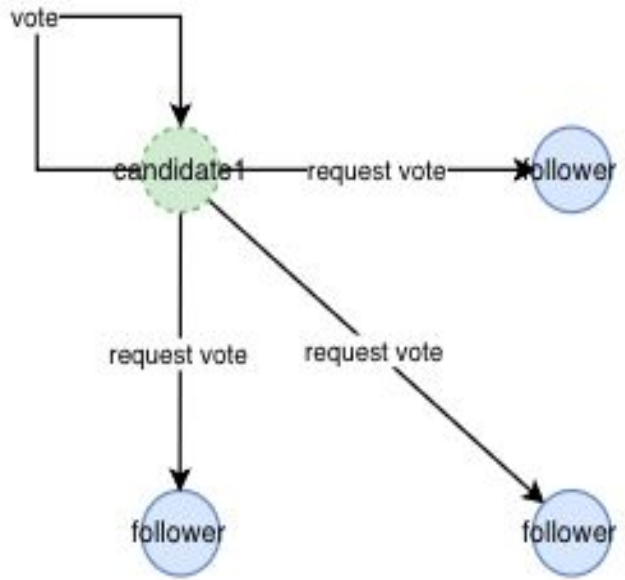
4. Kandidat dengan vote terbanyak akan menjadi node leader. Misalnya jika ada dua kandidat seperti ini:




candidate1 akan menjadi leader karena memiliki vote lebih banyak dari candidate2

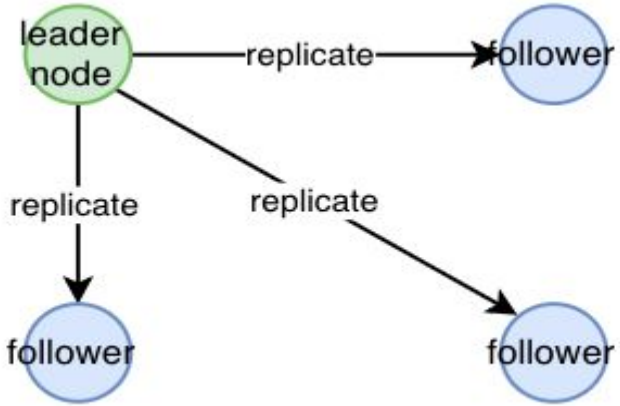
# Pembagian State

Di awal proses, setiap node atau web server merupakan **follower** state dan memiliki election timeout, tiap node atau follower akan melakukan pencarian apakah sudah ada **leader** atau belum. Jika time out habis atau belum ada leader, maka semua node atau web server menjadi **candidate** state, masing-masing candidate mengirimkan request ke web server lain dan menerima reply dari node lain. Node **candidate** yang memiliki vote terbanyak dari semua follower akan menjadi **leader**.



# Strategi Komunikasi Antar Web Server

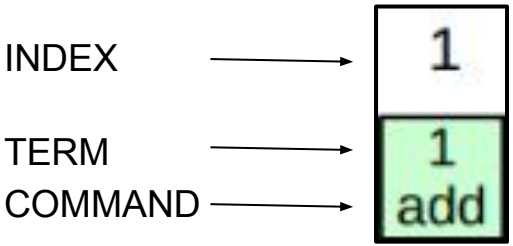
Setiap log data dari server leader diubah, maka server follower akan mereplikasi update log data agar tidak tertinggal dengan log data server leader dan semua server memiliki term yang sama.



# Log Structure

Struktur log data dalam skema Fault Tolerance adalah setiap log data harus mencantumkan index log yang merupakan angka untuk mengidentifikasi posisi log, command log merupakan pesan yang berisi data, dan log term sebagai parameter kondisi.

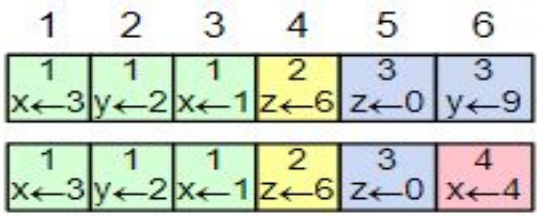
Log disimpan dalam storage supaya dapat bertahan saat terjadi kerusakan pada server



# Log Consistency

Dalam skema Fault Tolerance, konsistensi data dipastikan dengan mereplikasi log data dari server leader ke setiap follower, agar data tidak terubah dan server follower manapun bisa menjadi leader tanpa menghilangkan log data leader sebelumnya.

Entri Log akan menyimpan perintah yang sama dan log tersebut identik di semua entri sebelumnya.



# Evaluasi terhadap Konsistensi Data

Untuk memastikan bahwa tiap *Append Entries* sudah berisi indeks dan term, maka dilakukan commit pada entry tersebut.

Jika salah satu web server *leader* rusak, maka harus melakukan vote pemilihan leader baru. Dan *leader* baru akan *menghapus entri yang berlebih* dan *mengisi entri yang kosong* untuk membuat log konsisten

Jika *entry* tidak cocok, maka *follower* akan menolaknya. Setiap **append command** berisi indeks entri sebelumnya.