Experiment Setup:

- KVM hypervisor, virt-manager, Ubuntu 16.04 VM, 2GB RAM, 2 CPUs
- Avg workload:
    - CPU1: ~15%
    - CPU2: ~15%
    - MEM: ~50%
- We have used following set of parameters for comparison
    - % read=100, threads=8, ops/thread=50000
    - % read=99, threads=8, ops/thread=50000
    - % read=95, threads=8, ops/thread=50000

Result:

- Average time:
    - Barring Spinlock which does not differentiate between writers and readers, avg time taken by all other locks is increased
    - For no/few writers, RCU is most efficient as it does not have any overhead of changing a variable indicative of a lock
    - As % writers increases, Seqlock tends to become most efficient. This is because seqlock is very favourable to writers
- Average Read time:
    - Spinlock read time is invariant to the % writers because of previously mentioned reason
    - For each case, RCU takes least time followed by seqlock, RWlock and custom RW
    - As % writers increases, RCU read time remains same, RWlock read time increases (because of spin lock waiting and writer starvation), Seqlock read time increases (because of reread by readers as writer enters the CS whenever they want)
- Average Write Time:
    - RCU is very very bad for writers, its value increases with positive second derivative (Note: RCU bars are only indicative, they does not represents the actual time taken)
    - Spinlock write time is same irrespective of % writers
    - As % writers increases, RWlock write time decreases. This is because of lesser possibility of writers starvation per writer as % readers decreases
    - With the increase of % writers, Seqlock write time almost remains same as writers can enter CS whenever they want

Insights:

- RCU lock becomes exponentially inefficient as the no. of % writes increases. This is because of the fact that, in write call, RCU copies the shared data to a new location with large overhead subjected to the size of shared data.
- Spinlock is most fair lock as it does not differentiate between writers and readers
- Seqlock is the most unfair lock as it allows every writer to write whenever they want causing the active readers to reread the data again
- Custom RW lock is very similar to RWlock. Hence, they maintain the same kind of behaviour throughout.
- Average deviation is highest for the RCU because of indeterministic memory congestion while allocating, freeing and copying shared data