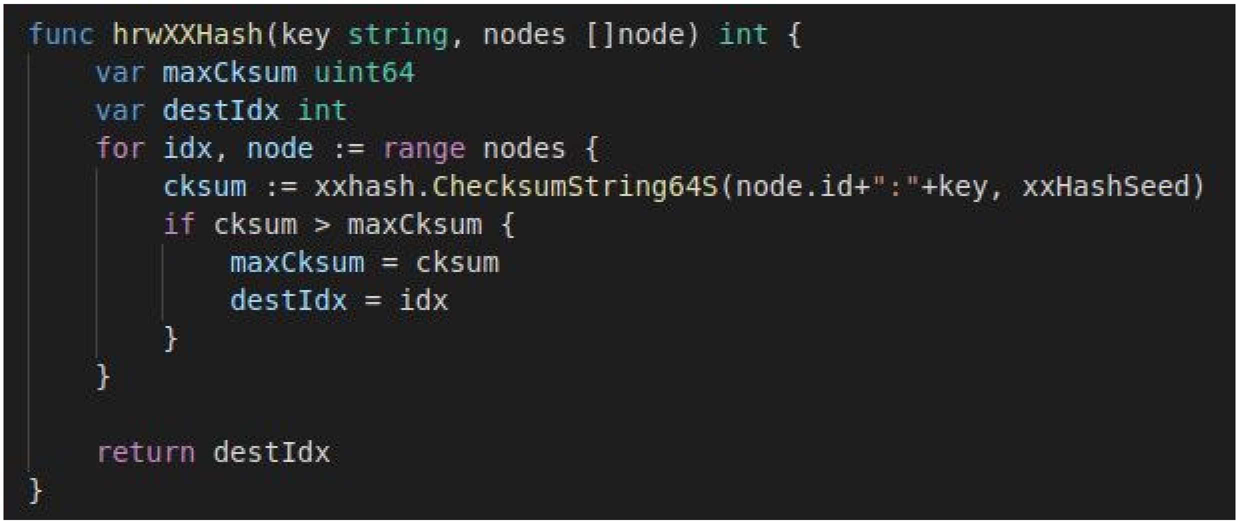
Present HRW algorithm with XXHash:

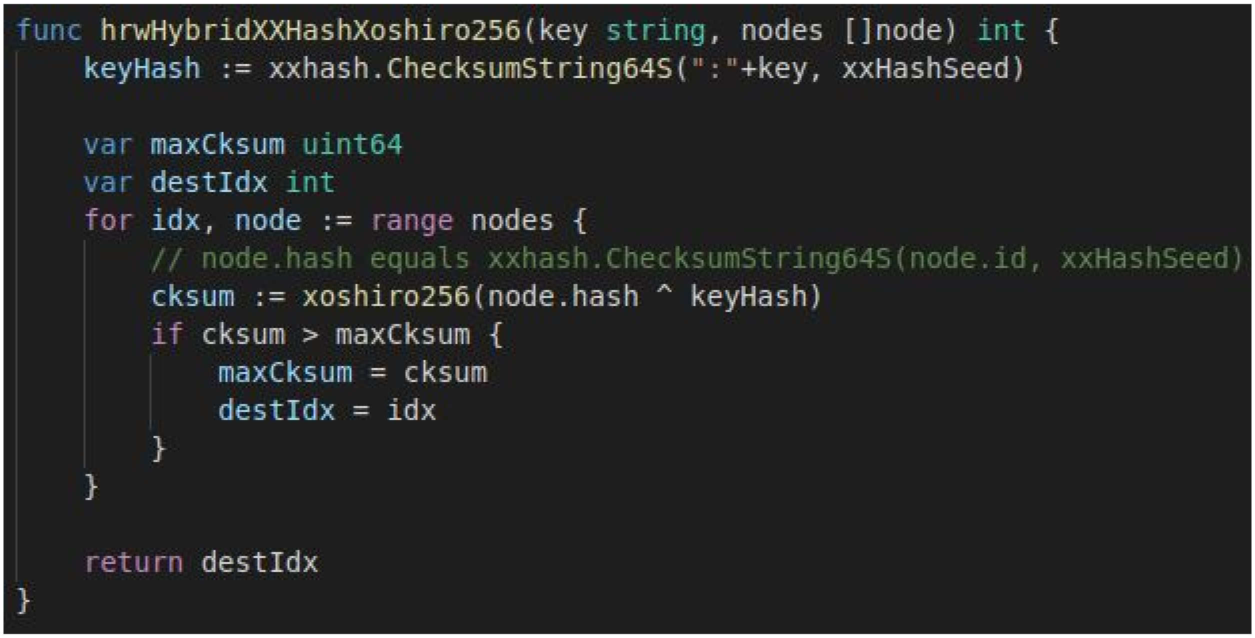


```
func hrwXXHash(key string, nodes []node) int {
    var maxCksum uint64
    var destIdx int
    for idx, node := range nodes {
        cksum := xxhash.ChecksumString64S(node.id+":"+key, xxHashSeed)
        if cksum > maxCksum {
            maxCksum = cksum
            destIdx = idx
        }
    }

    return destIdx
}
```

Proposed HRW algorithm with XXHash+Xoshiro256:**

- More details about Xoshiro256**: Click Here



```
func hrwHybridXXHashXoshiro256(key string, nodes []node) int {
    keyHash := xxhash.ChecksumString64S(":"+key, xxHashSeed)

    var maxCksum uint64
    var destIdx int
    for idx, node := range nodes {
        // node.hash equals xxhash.ChecksumString64S(node.id, xxHashSeed)
        cksum := xoshiro256(node.hash ^ keyHash)
        if cksum > maxCksum {
            maxCksum = cksum
            destIdx = idx
        }
    }

    return destIdx
}
```

Important factors which need to be considered when changing the underlying hashing technique:

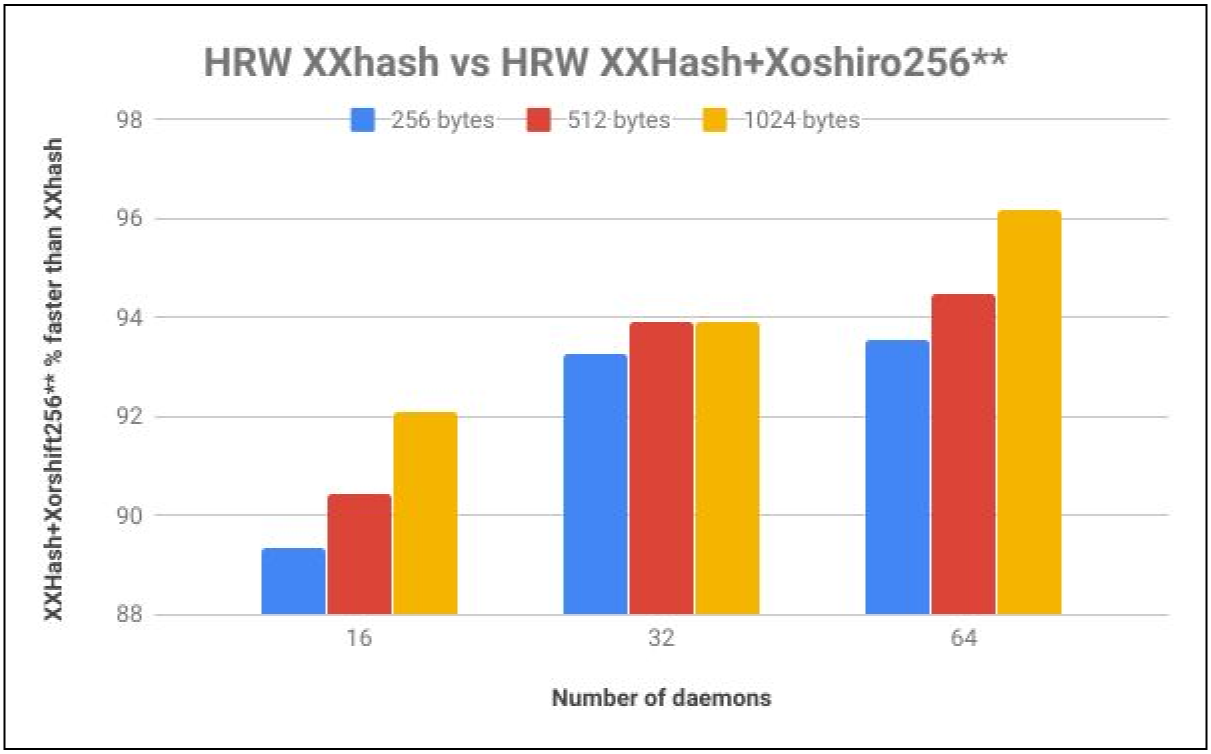
1. Performance
2. Distribution of objects across targets - if any approach allocates a large number of objects to one target then this could be a problem because now this target will exhaust its disk capacity faster than others causing LRU to be triggered. Also, this might cause a large amount of data to be rebalanced (if a large number of objects map to the newer target)

We compare three approaches - current algorithm, xorshift64* and xoshiro256**

Experiment 1: Compare the performance of the various approaches

- Vary the number of daemons - {16, 32, 64}
- Vary the length of the object name which is given as input to HRW: {256, 512, 1024} bytes

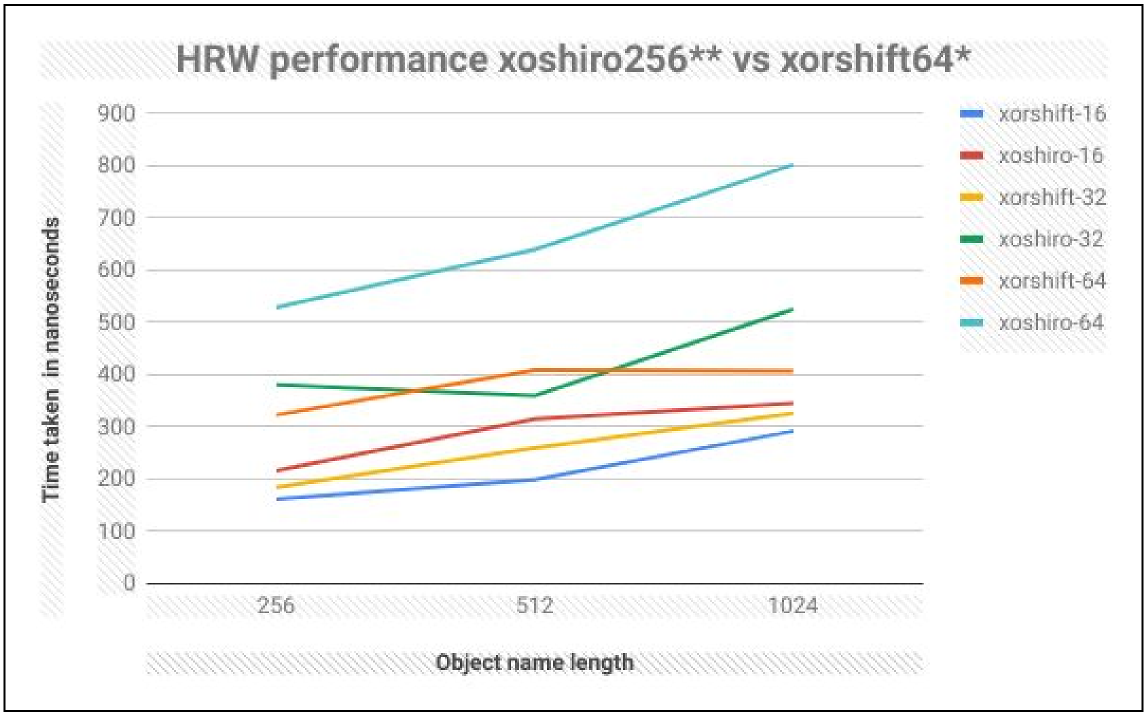
The below graph describes by how much percentage is HRW with XXhash + Xoshiro256 faster than HRW with XXHash.**



The below table contains the time taken (in nanoseconds) to find a daemon given the number of daemons and the length of the object name.

hash_algo-num_daemons	256 bytes	512 bytes	1024 bytes
xxhash-16	2030	3305	4361
Xoshiro256**-16	216	315	345
xxhash-32	5652	5920	8659
Xoshiro256**-32	381	360	525
xxhash-64	8233	11657	20945
Xoshiro256**-64	529	640	802

Conclusion: HRW XXHash + Xoshiro256** is roughly twice as faster than HRW XXhash.

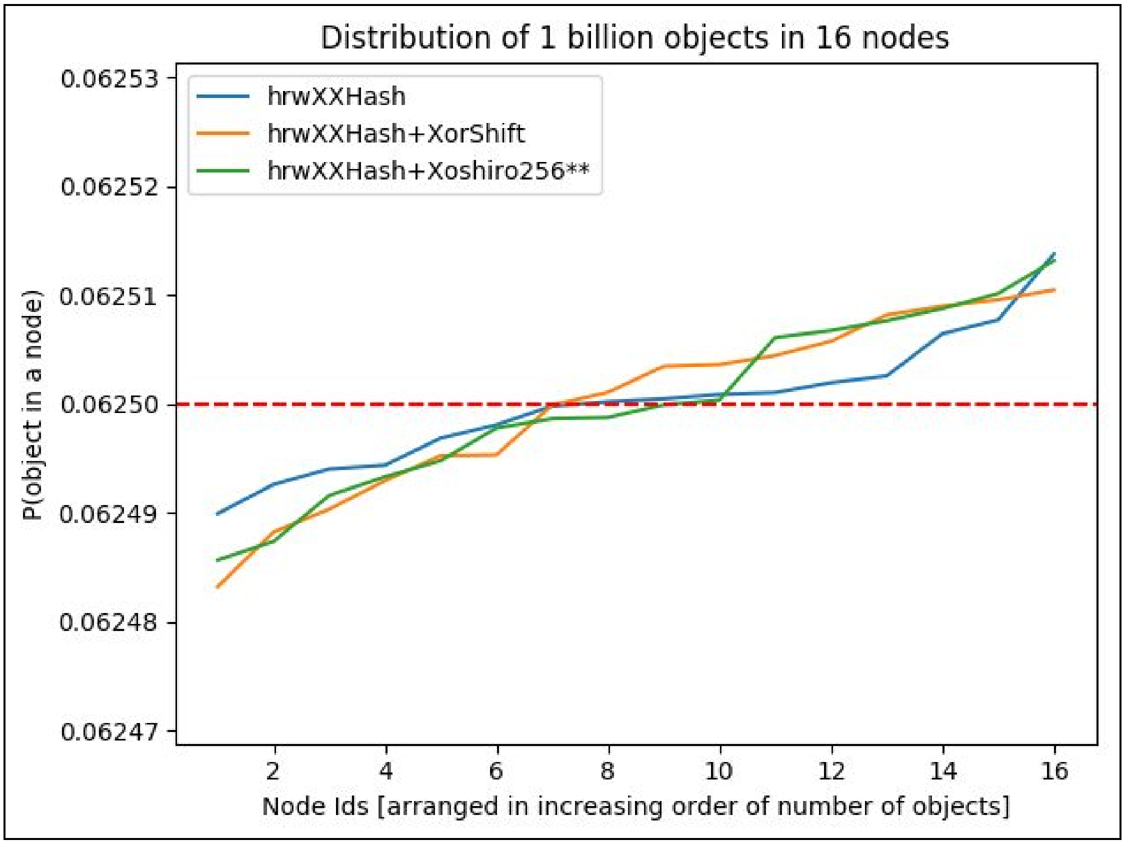


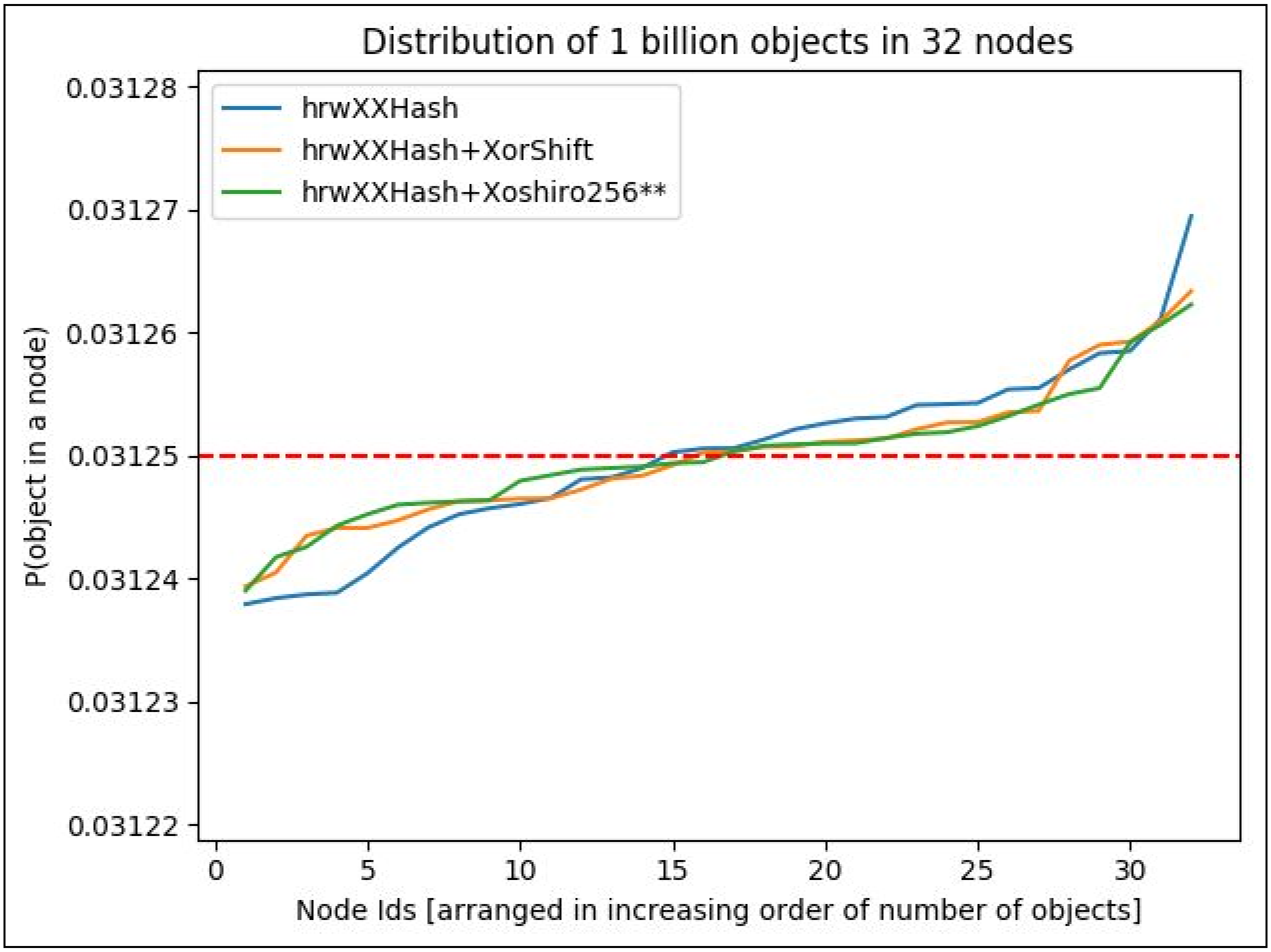
Experiment 2: Measure the distribution of objects

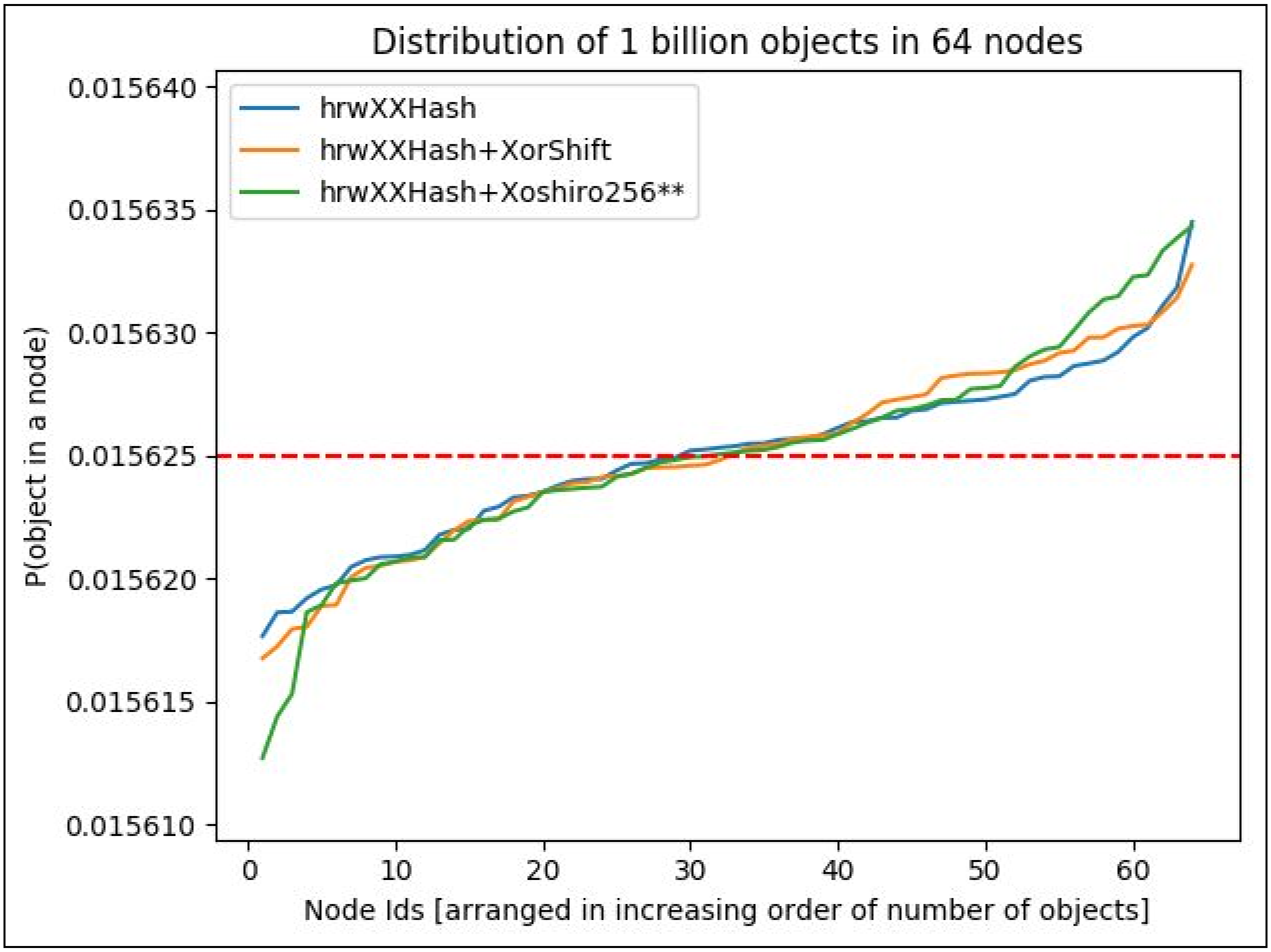
- Vary the number of daemons: {16, 32, 64}
- Total number of objects: 1 billion

How to interpret the below plots?

The closer the plot is to the red line the more equal is the distribution.




Distribution of 1 billion objects in 32 nodes
hrwXXHash
hrwXXHash+XorShift
hrwXXHash+Xoshiro256**
P(object in a node)
0.03128
0.03127
0.03126
0.03125
0.03124
0.03123
0.03122
0
5
10
15
20
25
30
Node Ids [arranged in increasing order of number of objects]


Distribution of 1 billion objects in 64 nodes
hrwXXHash
hrwXXHash+XorShift
hrwXXHash+Xoshiro256**
P(object in a node)
0.015640
0.015635
0.015630
0.015625
0.015620
0.015615
0.015610
0
10
20
30
40
50
60
Node Ids [arranged in increasing order of number of objects]