What have we looked at so far?
I DON'T ALWAYS BUILD SYSTEMS

BUT WHEN I DO I BUILD APPROPRIATE ONES
Assumptions

● CAP Theorem

● SQL and NoSQL

● Hashing
Origin’s of Dynamo
This is year 2004

One Amazon was growing and other shrinking
What led to Dynamo?

Servers go down

And everyone loses their minds
What led to Dynamo?

- Amazon was using Oracle enterprise edition
- Despite access to experts at Oracle, the DB just couldn’t handle the load.
What did folks at Amazon Do?

BRACE YOURSELF

DATA ANALYSIS IS COMING
90% of operations weren't using the JOIN functionality that is core to a relational database.
Goals which Dynamo wanted to achieve

- Highly Always available
- Consistent performance
- Horizontal Scaling
- Decentralized
Goals which Dynamo wanted to achieve

- Highly Always available
- Consistent performance
- Horizontal Scaling
- Decentralized
Major aspects of Dynamo design

- Interface
- Data Partitioning
- Data Replication
- Load Balancing
- Eventual Consistency
- And a lot of other this and that, hopefully we will cover all of it.
Consistency Model

THE ONLY RULE OF CAP THEOREM

YOU CAN ONLY PICK 2 OUT OF 3
Eventually Consistent

- The reads can contain stale data for some bounded time.
Amazon chose the Eventual Consistency Model

- Application will work just fine with eventual consistency

- They needed a scalable DB
Let’s Finally get to Dynamo !!
This is Dynamo!!
Origin of this ring?

- **Consistent Hashing?**
- How can we increase or decrease number of nodes in distributed cache without re-calculating the full distribution of hash table?
→ 128 bit hash space
→ First one overlaps last one (circle)
- Each node is assigned a spot in the ring.
- A data point is the responsibility of the first node in the clockwise direction (coordinator node).
Some issues with Consistent Hashing

- Random Assignment
- Heterogeneous Performance of Node
How replication work?

- The coordinator node replicates to next N-1 nodes.
- N is the replication factor.
Data Versioning

- Eventual Consistency

- Multiple Versions of same data might exist in systems

- Come Vector Clocks
Vector Clocks

D1 ([Sx,1])
- write
- handled by Sx

D2 ([Sx,2])
- write
- handled by Sx

D3 ([Sx,2],[Sy,1])
- write
- handled by Sy

D4 ([Sx,2],[Sz,1])
- write
- handled by Sz

D5 ([Sx,3],[Sy,1],[Sz,1])
- reconciled
- and written by Sx
Dynamo DB deployment

- Loadbalancer
- Client Aware library
Dynamo DB query interface

- get() and put() operations

- Configurable R and W.

- $R =$ Min Number of Nodes to read from before returning
- $W =$ Min number of Nodes on which data should be written before returning
Making Dynamo Consistent

- If \( R+W > N \)
  - Dynamo becomes consistent

- Availability and Performance takes a hit.
Handling Failures

- Hinted Handoff
- Replica Synchronization
Hinted Handoff
Replica Synchronization

- Each node maintains separate Merkle Tree of the key ranges it's handling.

- A background job runs trying to do a quick match and find which set of replicas need to be merged.
Failure Detection

● If a node is not reachable the request is routed to the next node,

● No need to explicitly detect failure. As node removal is explicit operation.
Differences between GFS/BigTable and Dynamo

- No centralized control
- No locks on data.
Optimizations done later

- Instead of write to disk, write to buffer
- Separate writer, write to disk
- Faster write performance
Change in key partition strategy

- The one described -
  - Random
  - Hash space not uniform

- Problems-
  - Data copy difficult
  - Merkle Tree reconstructed
New Partition Strategy

- Divide hash space equally in Q portions
- Each node S is given Q/S tokens
- A new node randomly picks its Q/S+1 tokens
- A removal of node randomly distributes Q/S tokens
Impact

- A lasting impact on industry, forced SQL advocated to build distributed SQL DB’s
- Cassandra, Couchbase
- Established scalability of NoSQL databases.
Questions
Adding a node to the ring

- The administrator issues a request to one of the nodes in the ring.

- The serving request node makes a persistent copy of the membership change and propagates via gossip protocol.
Node on startup