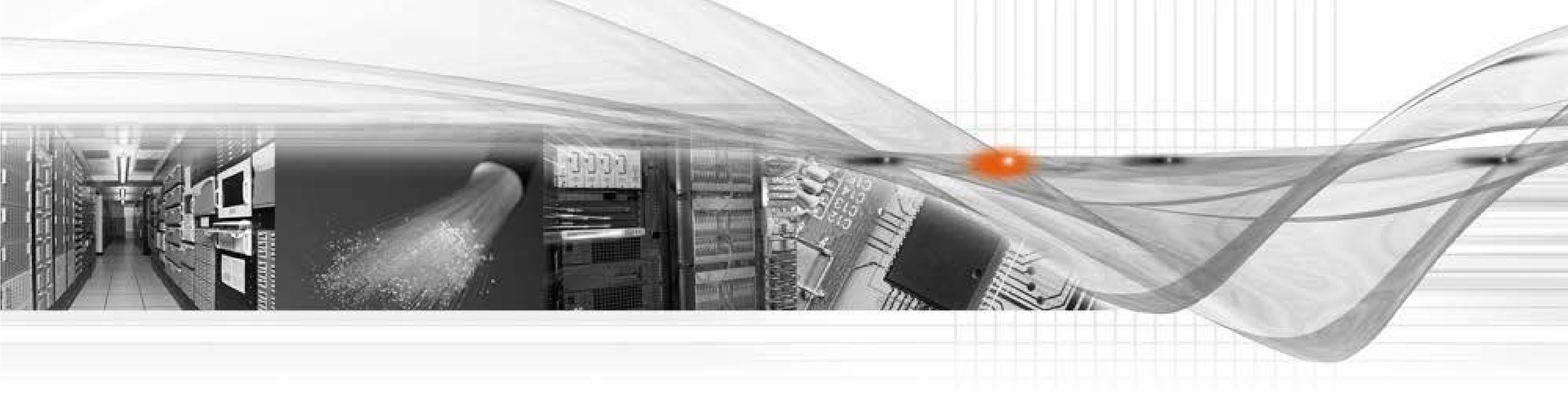


InfiniBand Scalable SA

An OFA Project

Susan Coulter, Sean Hefty, Ilya Nelkenbaum,
Hal Rosenstock, Ira Weiny, Eitan Zahavi

The Problem

n^2 SA load

Where's Carol?

I need to talk to Mike

Tell everyone I'm leaving

Has anyone seen Peter?

I need to talk to Mike, too

Greg didn't answer. Is this number correct?

I'm back, what was Alice's address again?

Where's Marcia?

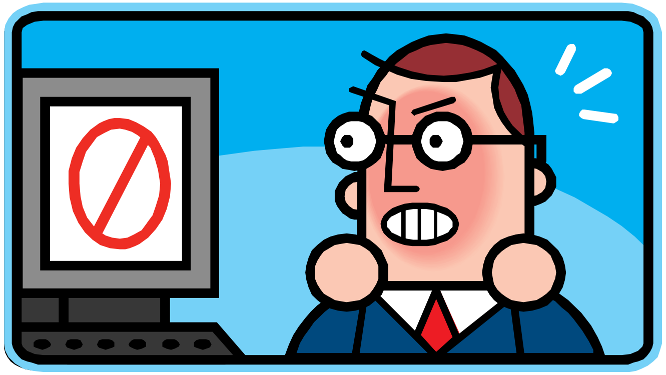
Cindy needs to talk to Bobby

I have a message for Jan

I need to see Marcia

Marcia, Marcia, Marcia

@#%&$!!!



Revisiting the Problem

- SA queried for every connection
- Communication between all nodes creates an n^2 load on the SA
- Other n^2 scalability issues
  - Name to address (DNS)
    - Mainly solved by a hosts file
  - IP address translation
    - Relies on ARPs

Doesn't IB ACM fix this?

Issues

Processing still centralized
SA must construct path record

Cached data must be kept current
- significant overhead
- static files limited to specific topologies and homogeneous clusters

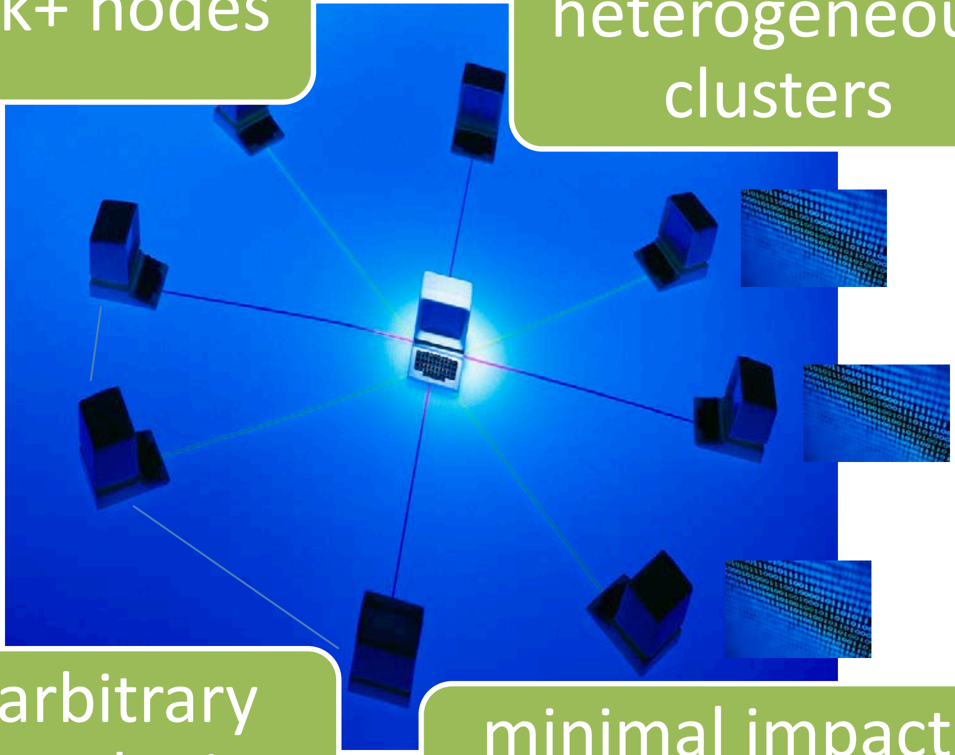
Heavy burden on single multicast group
Address resolution

A Truly Novel Solution…

Scalable SA (SSA)

- Extending the SA implementation
  - Improved opportunities for solutions
  - Open source
- Focused on scalability *and* reliability
  - Fault occurrence is likely
- Dependent on SM

Goals



40k+ nodes

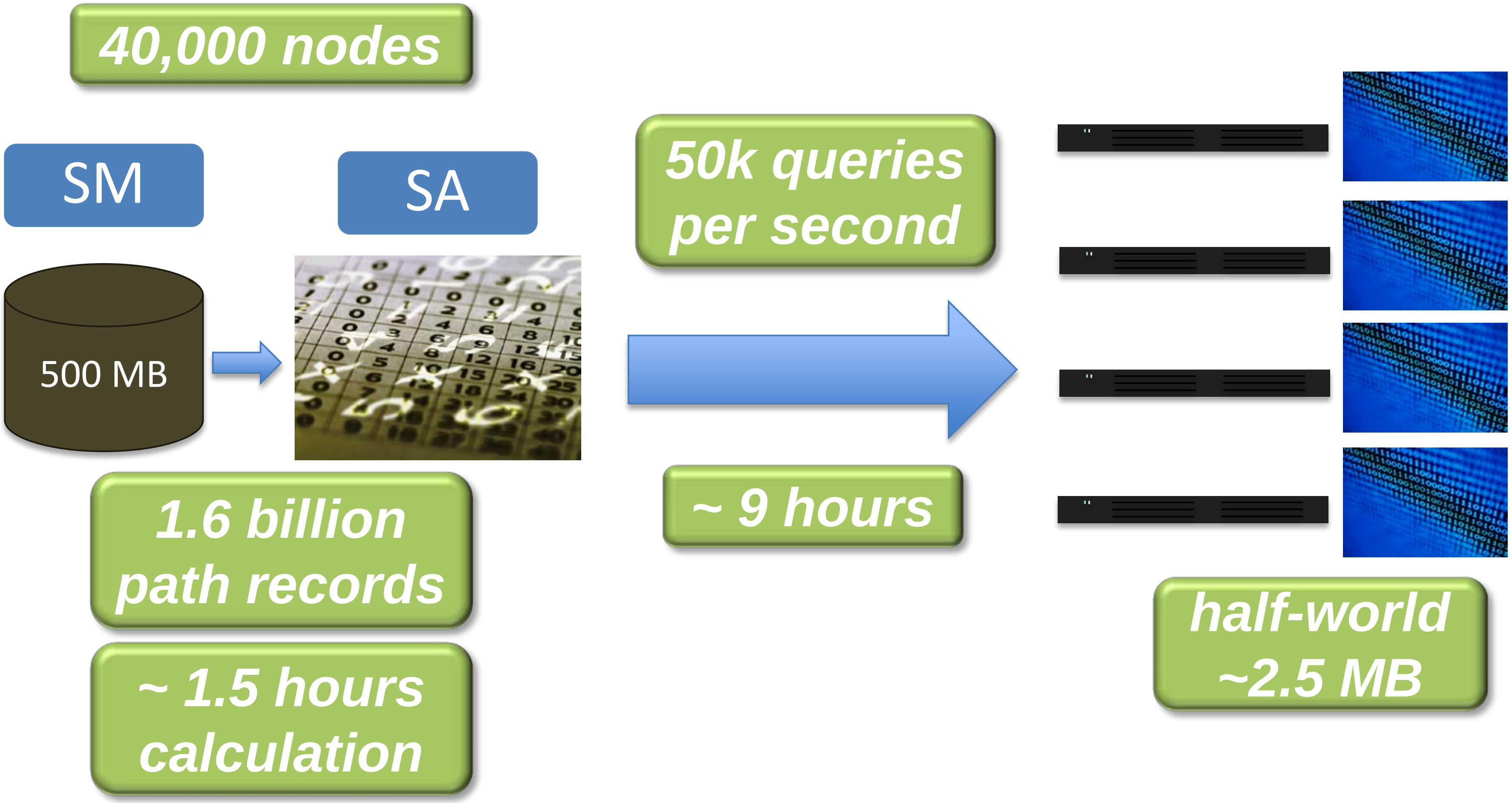
heterogeneous clusters

arbitrary topologies

minimal impact to compute nodes

- Support distributed processing
- Ensure consistency of cached data
- Avoid large multicast domains
  - Do not rely on IPoIB
- Work with existing RDMA CM apps

Analysis



40,000 nodes

SM

SA

500 MB

1.6 billion path records

~ 1.5 hours calculation

50k queries per second

~ 9 hours

half-world ~2.5 MB

SSA Model

Computation

- Distributed – select nodes
- Multithreaded - lockless

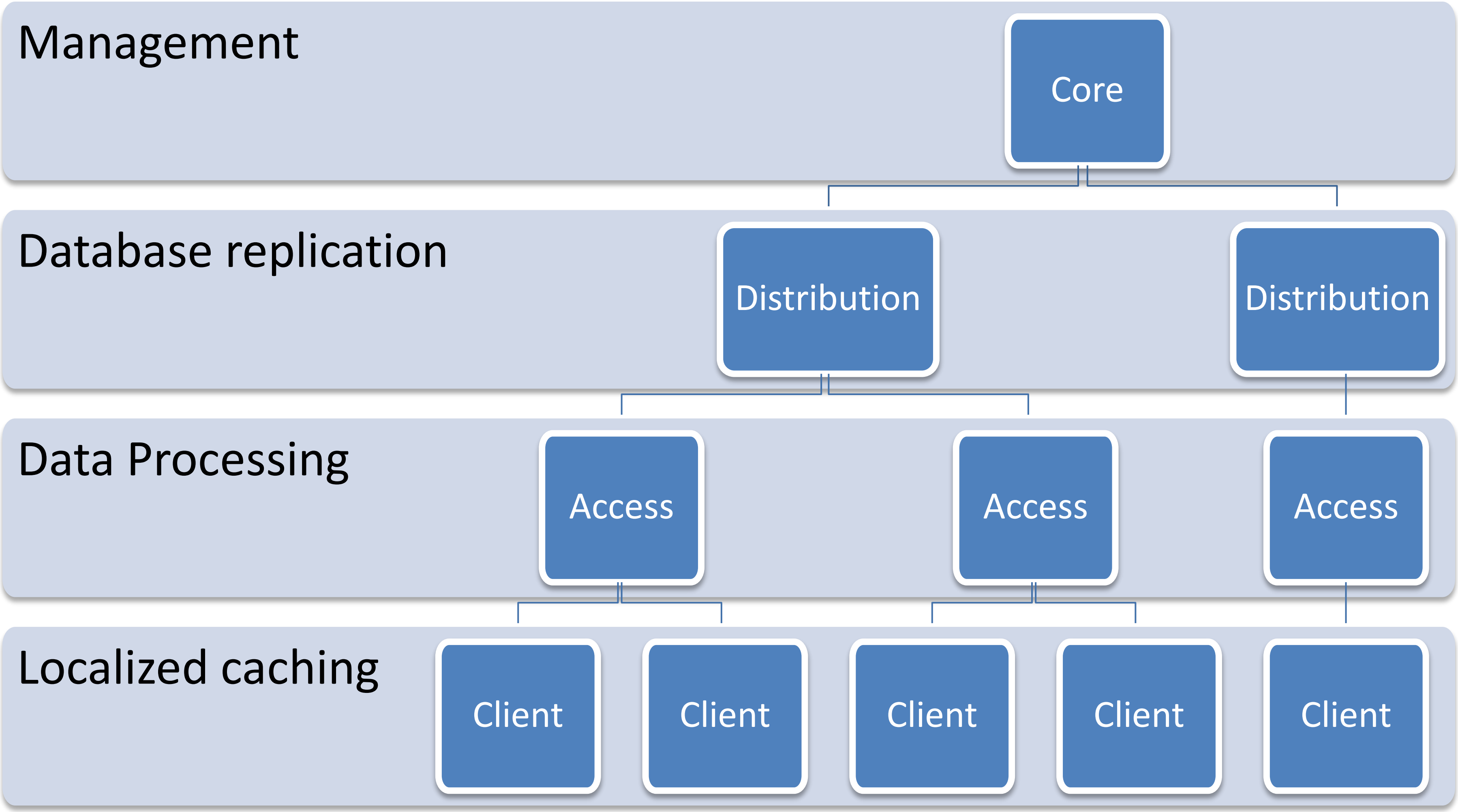
Data

- Weakly coherent
- Incremental updates

Communication

- Minimize subnet impact
- Detect and report errors
- Failover

Architecture



Management

Core

Database replication

Distribution

Distribution

Data Processing

Access

Access

Access

Localized caching

Client

Client

Client

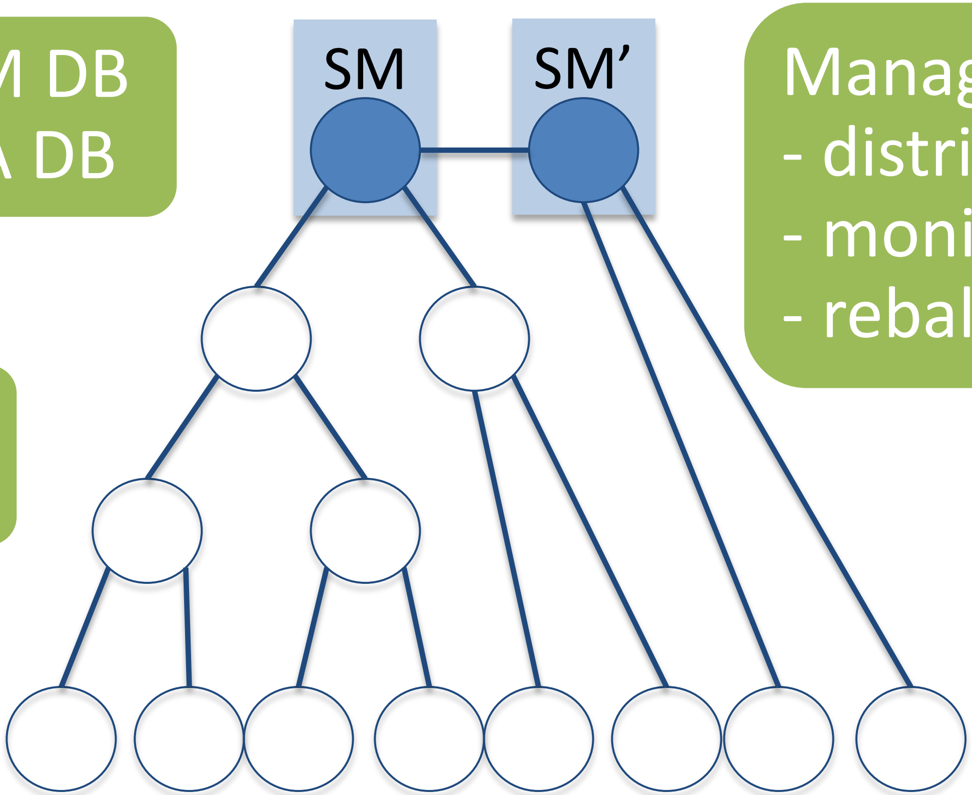
Client

Client

Core Layer

Core found
at SM LID

raw SM DB
→ SSA DB



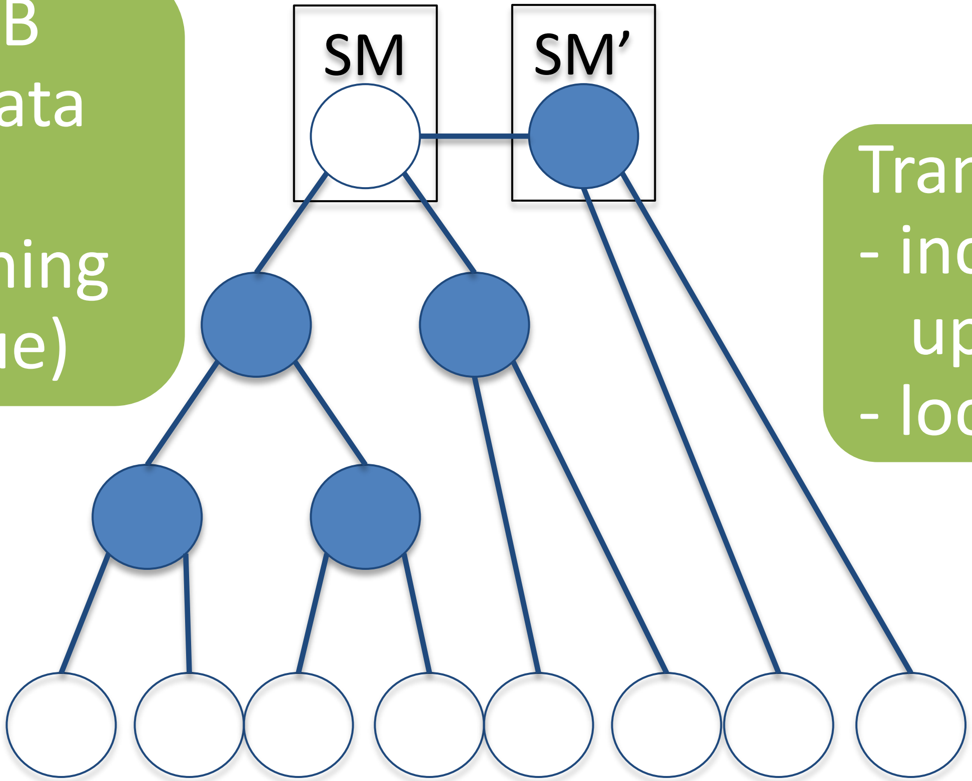
Manage SSA group
- distribution control
- monitoring
- rebalancing

Nodes join
SSA tree

Distribution Layer

Data agnostic

Distributes DB
- relational data model
- data versioning (epoch value)

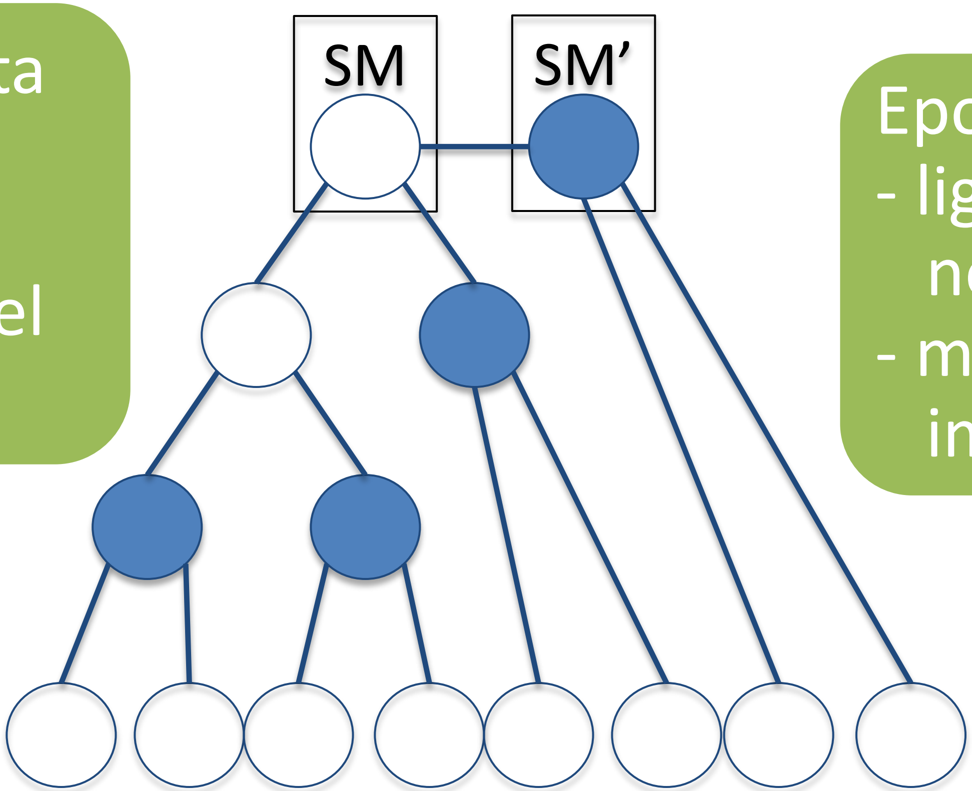


Transaction log
- incremental updates
- lockless

Access Layer

Data aware

Formats data
- select SA queries
- higher-level queries

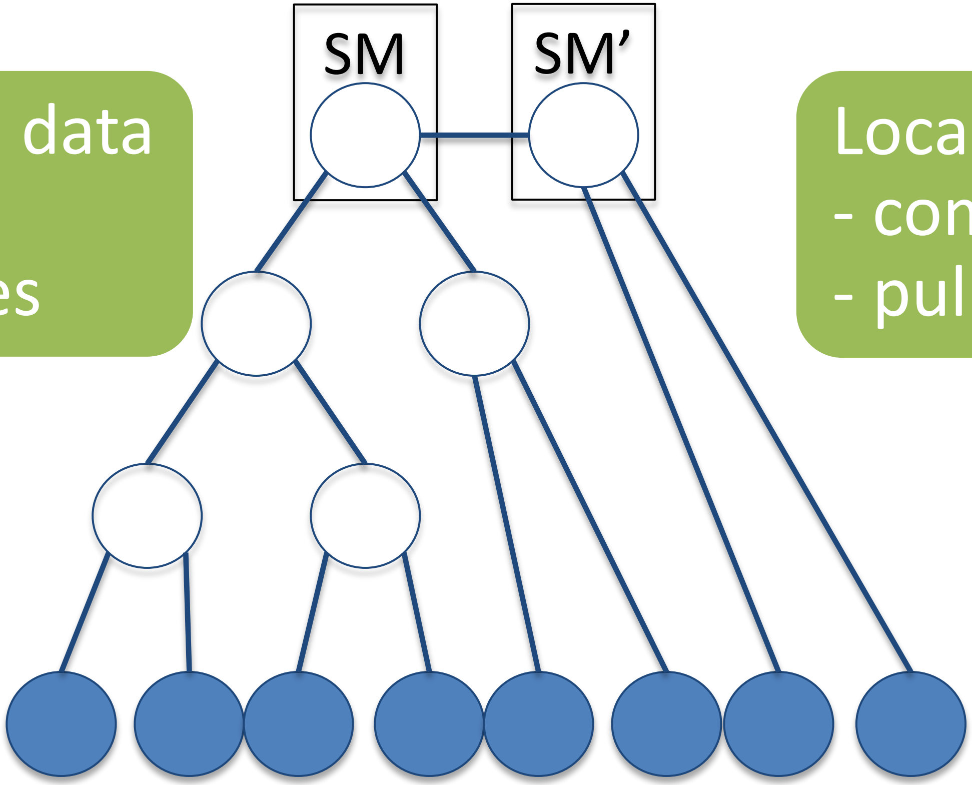


Epoch value
- lightweight notification
- minimal job impact

Client

Integrated with IB ACM
- via librdmacm

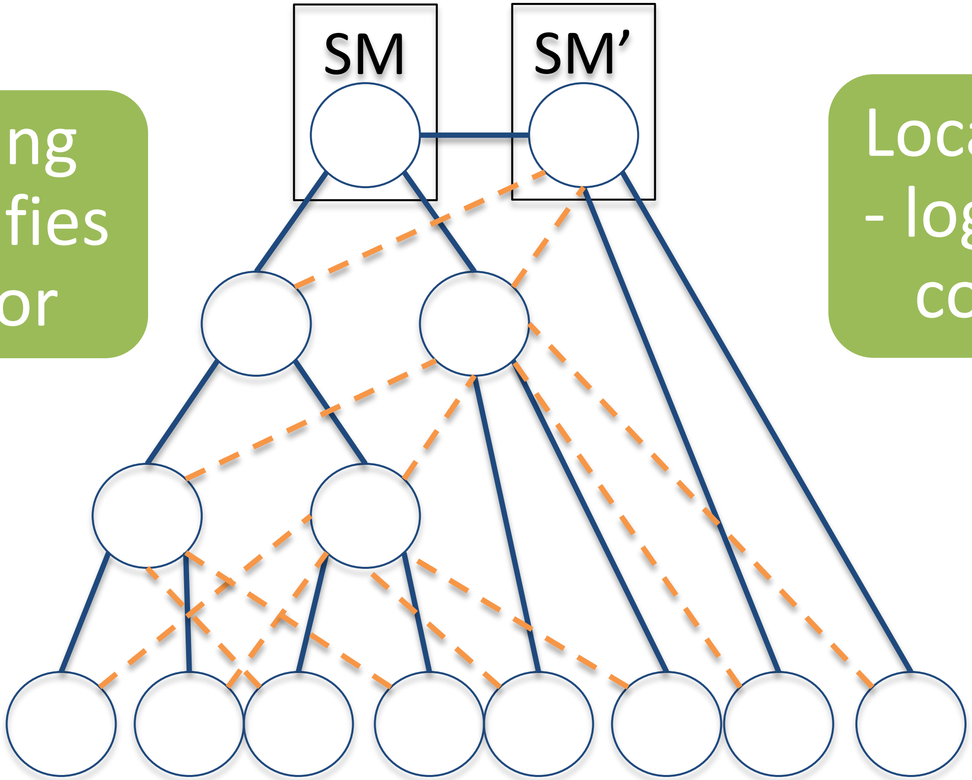
Publish local data
- hostname
- IP addresses



Localized cache
- compares epoch
- pull updates

Reliability

Primary and backup parents



Error reporting - parent notifies core of error

Local databases - log files for consistency

Summary

- A scalable, distributed SA
- Works with existing apps
- Fault tolerant

What's the catch?

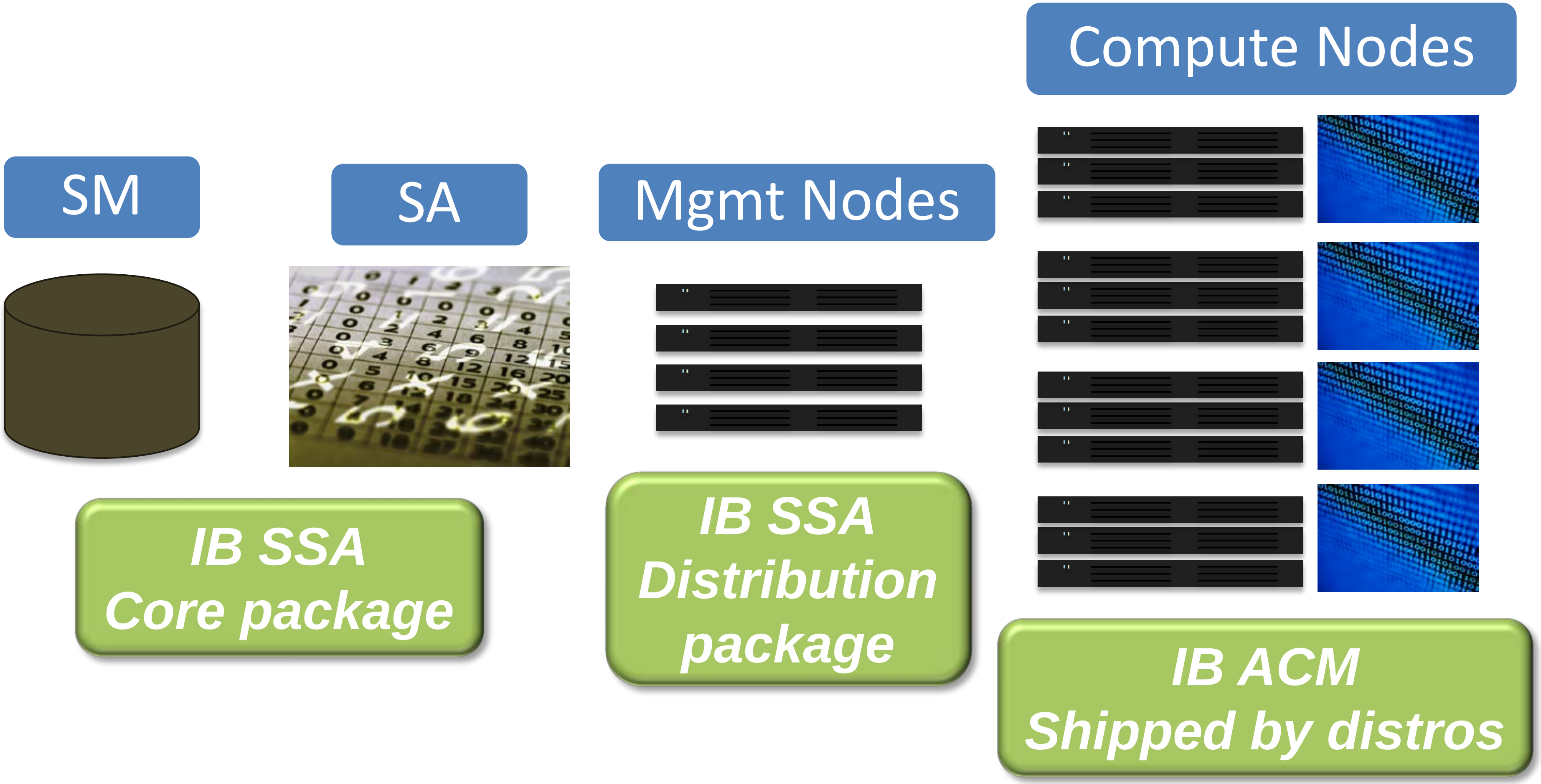
Development Phases

1. Path record distribution
    1. ACM to SSA core
    2. Add distribution nodes
2. Address resolution
    1. Collect <address/name, port> up SSA tree
    2. Redistribute mappings
    3. Resolve path records directly from address/names
3. Event collection and reporting
    1. Performance monitoring

we are here

Deployment

Target 2013 Preview Release



SM

SA

Mgmt Nodes

Compute Nodes

IB SSA Core package

IB SSA Distribution package

IB ACM Shipped by distros

Thank you

Please come again

Okay now

Buy .. Buh-buy .. Buy