The Insider's Checklist For Hardening an In-Memory Computing Cluster

Alexey Goncharuk

Jun 04 2019

Agenda

- Why bother?
- Check: what and how?
  - Resource planning
  - Topology planning
  - Test planning
  - Monitoring planning
- Summary

Why bother?

Why bother?

We are working with a distributed system

“A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable”

L. Lamport

Why bother?

Murphy's Law

"Whatever can go wrong, will go wrong"

Why bother?

Prepare as much as you can

- Be aware of known pitfalls
- Be prepared for unexpected events
- Be prepared for unexpected growth
- Be prepared before production!

Check: what and how?

Let’s dig in!

Capacity planning

Capacity planning: data set size

Problem definition

How much memory do I need to store an X GB file?

Capacity planning: data set size

Problem definition: wrong!

How much memory do I need to store an X GB file?

- Different file formats
- Different object model
- Indexes?

Capacity planning: data set size

Problem definition

- Given a model of N types
- Each type has a representative sample
- Each type has an estimated number of key-value pairs
- We know what indexes we will need
- How much memory (RAM, disk) needed to store it?

Capacity planning: data set size

Can it be calculated?

- Yes, but need to know the internals
- Hard to work with variable-length fields

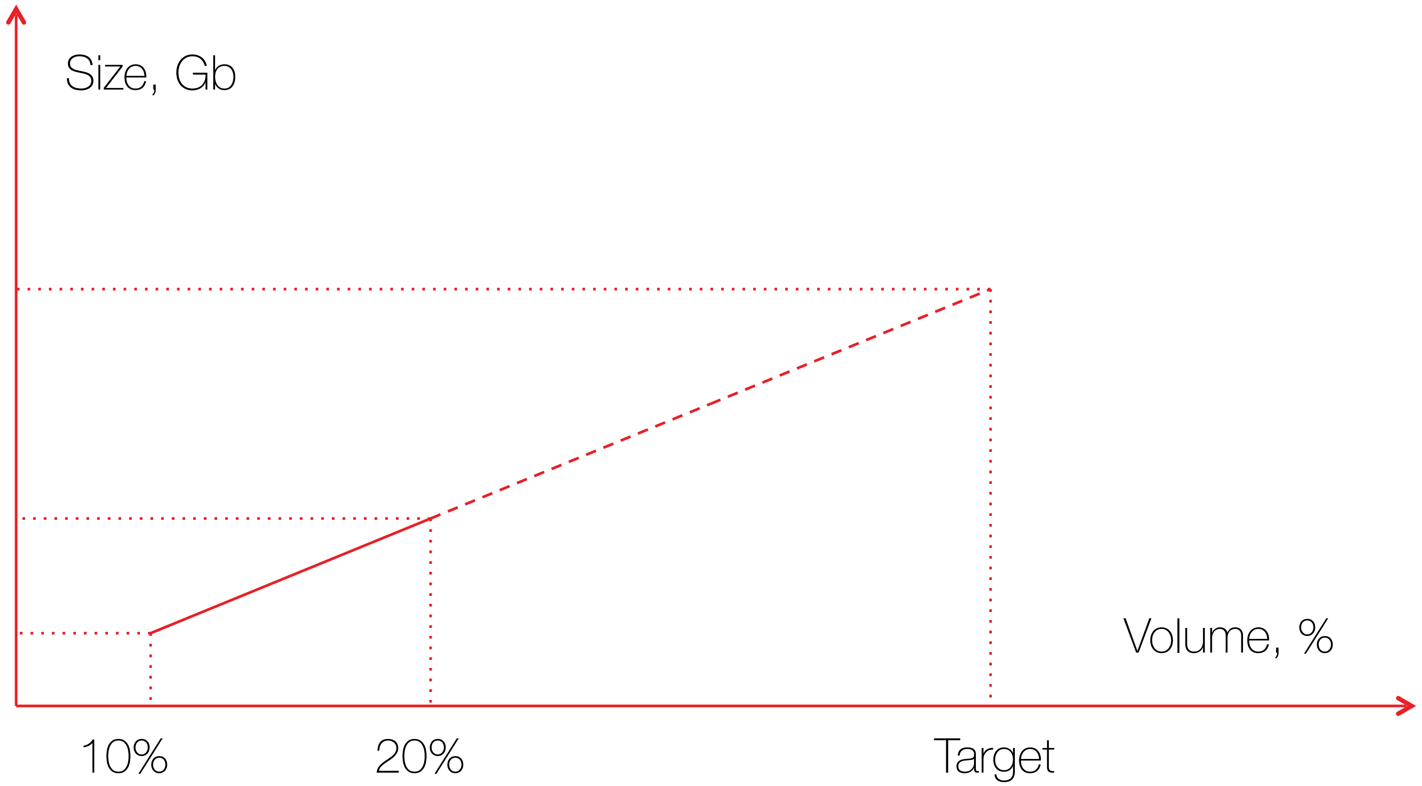
Capacity planning: data set size

Can be measured

- Can be done locally
- Easy to change model

Capacity planning: data set size

Can be measured



Capacity planning: data set size

Can be measured

- Can be done locally
- Easy to change model
- Pitfalls:
  - Make sure to have a representative sample
  - Make sure to have a large enough sample
  - Double-check random data generators for unexpected correlations

Capacity planning: data set size

Check for correlations

```
BusinessObject {
    String field1;
    String field2;
}

BusinessObject {“TestObject0001”, “TestObject0002”},
BusinessObject {“TestObject0003”, “TestObject0004”},
BusinessObject {“TestObject0005”, “TestObject0006”},
…
```

Capacity planning: RAM / Disk ratio

RAM is great, but…

Capacity planning: RAM / Disk ratio

You may want to offload to disk

What happens when data does not fit RAM?

- RAM miss leads to a disk read
- Disk reads number is limited (IOPs)
- Need to throw away a portion of cached data (replacement strategy)

Capacity planning: RAM / Disk ratio

You may want to offload to disk

How to minimize page replacement effects?

- Keep hot and cold data separately
  - For Ignite – use different DataRegions
- Keep an eye on disk saturation
- May want to use topology tricks

Capacity planning: Disk

I know disk size, what else?

- Disks have limited IOPs (both read and write)
- Write TPS is limited by IOPs
- Separate Journal, Checkpoint and Backup volumes

Capacity planning: Summary

Capacity checklist

- Estimate data set size
- Estimate RAM / Disk ratio
- Check disk characteristics

Topology planning

Topology planning: bring computation to data

Use-case: use compute capabilities

- Build results based on local data
- Send compute, not data

Topology planning: split into cells

Use-case: logically co-located data

- Multiple partitions per city
- Users usually interact within cities they live

Topology planning: split into cells

Regular partitioning

- Usually primary and backup nodes are selected evenly
- Goal is to minimize load during node failure

Topology planning: split into cells

Split large topology into sub-cells

- Split nodes into groups of N nodes in each group
- Assign partitions to groups using data locality where possible
  - Example: Same city means same group

Topology planning: separate functional groups

Use-case: clear functional groups

- Two kind of tasks
  - CPU-intensive compute
  - Disk-intensive writes
- Different resource requirements

Topology planning: separate functional groups

Make use of heterogeneous cluster

- Different node roles require different resources
- Different load patterns mean different resource utilization patterns
- Fewer cross-domain effects
- Ignite: NodeFilter and node attributes

Topology planning: Summary

Topology checklist

- Make use of functional data locality
- Make use of separate functional groups
- Check product-specific features

Test Planning

Test Planning

What additional check do I need?

- Check relevant load scenarios
  - Maximize utilization, but avoid 100%
- Check on target data set sizes
  - Verify rebalance speed (i.e. backup factor recovery time)
  - Verify performance
- Test before going to production

Monitoring Planning

Monitoring planning

Prepare instruments to resolve incidents

- Record critical metrics and events
- Always have GC logs enabled
  - Rule out latency spike causes
  - Rule out ‘response timed out’ causes
- Allow runtime logging changes

Summary

Let’s Sum Up

“A week of thinking saves four months of development”

Summary

Let’s Sum Up

- Resources
  - Estimate data set size
  - Estimate RAM / Disk ratio
  - Be aware of resource saturation
- Exploit topology benefits
- Test dist-sys specific scenarios
- Monitor your system

Q&A

Thank you for your attention!