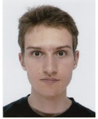
Alex Auvolat, Deuxfleurs Association

`https://garagehq.deuxfleurs.fr/`

Matrix channel: `#garage:deuxfleurs.fr`

Who I am



Alex Auvolat
PhD; co-founder of Deuxfleurs

Deuxfleurs
A non-profit self-hosting collective,
member of the CHATONS network

Our objective at Deuxfleurs

Promote self-hosting and small-scale hosting as an alternative to large cloud providers

Our objective at Deuxfleurs

Promote self-hosting and small-scale hosting as an alternative to large cloud providers

Why is it hard?

Our objective at Deuxfleurs

Promote self-hosting and small-scale hosting as an alternative to large cloud providers

Why is it hard?

Resilience

we want good uptime/availability with low supervision

Building a resilient system with cheap stuff

- Commodity hardware (e.g. old desktop PCs)

Building a resilient system with cheap stuff

Building a resilient system with cheap stuff

Building a resilient system with cheap stuff

- Commodity hardware (e.g. old desktop PCs)

  (can die at any time)

Building a resilient system with cheap stuff

- Commodity hardware (e.g. old desktop PCs)

  (can die at any time)

- Regular Internet (e.g. FTTB, FTTH) and power grid connections

Building a resilient system with cheap stuff

- Commodity hardware (e.g. old desktop PCs)

  (can die at any time)

- Regular Internet (e.g. FTTB, FTTH) and power grid connections

  (can be unavailable randomly)

Building a resilient system with cheap stuff

- Commodity hardware (e.g. old desktop PCs)

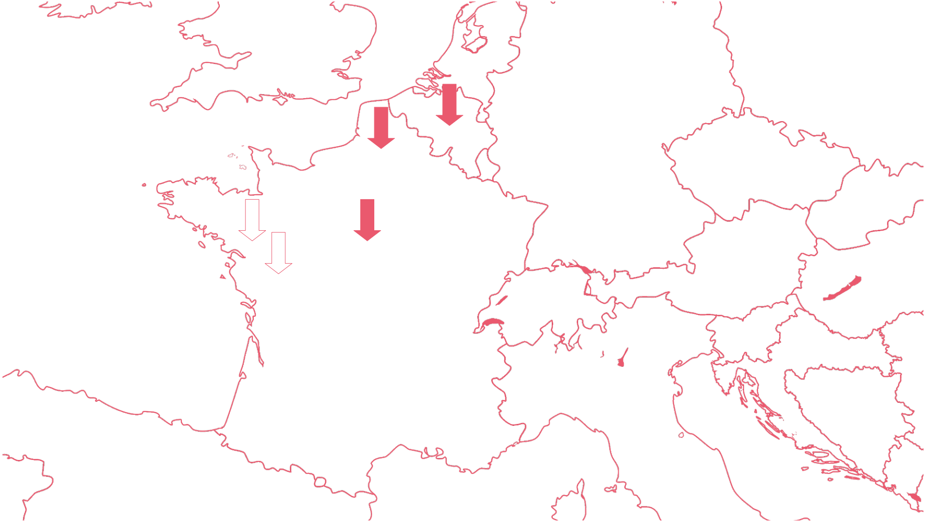
  (can die at any time)

- Regular Internet (e.g. FTTB, FTTH) and power grid connections

  (can be unavailable randomly)

- **Geographical redundancy** (multi-site replication)

Building a resilient system with cheap stuff



Object storage: a crucial component

Amazon S3

S3: a de-facto standard, many compatible applications

Object storage: a crucial component

S3: a de-facto standard, many compatible applications

MinIO is self-hostable but not suited for geo-distributed deployments

Object storage: a crucial component

S3: a de-facto standard, many compatible applications

MinIO is self-hostable but not suited for geo-distributed deployments

Garage is a self-hosted drop-in replacement for the Amazon S3 object store

CRDTs / weak consistency instead of consensus

Consensus can be implemented reasonably well in practice, so why avoid it?

CRDTs / weak consistency instead of consensus

Consensus can be implemented reasonably well in practice, so why avoid it?

- **Software complexity**

CRDTs / weak consistency instead of consensus

Consensus can be implemented reasonably well in practice, so why avoid it?

- **Software complexity**
- **Performance issues:**

CRDTs / weak consistency instead of consensus

Consensus can be implemented reasonably well in practice, so why avoid it?

- **Software complexity**
- **Performance issues:**
  - The leader is a **bottleneck** for all requests

CRDTs / weak consistency instead of consensus

Consensus can be implemented reasonably well in practice, so why avoid it?

- **Software complexity**
- **Performance issues:**
  - The leader is a **bottleneck** for all requests
  - **Sensitive to higher latency** between nodes

CRDTs / weak consistency instead of consensus

Consensus can be implemented reasonably well in practice, so why avoid it?

- **Software complexity**
- **Performance issues:**
  - The leader is a **bottleneck** for all requests
  - **Sensitive to higher latency** between nodes
  - **Takes time to reconverge** when disrupted (e.g. node going down)

CRDTs / weak consistency instead of consensus

Consensus can be implemented reasonably well in practice, so why avoid it?

- **Software complexity**
- **Performance issues:**
  - The leader is a **bottleneck** for all requests
  - **Sensitive to higher latency** between nodes
  - **Takes time to reconverge** when disrupted (e.g. node going down)

Internally, Garage uses only CRDTs (conflict-free replicated data types)

The data model of object storage

Object storage is basically a **key-value store**:

Key: file path + name	Value: file data + metadata

`index.html`	`Content-Type: text/html; charset=utf-8`
	`Content-Length: 24929`
	`<binary blob>`

`img/logo.svg`	`Content-Type: text/svg+xml`
	`Content-Length: 13429`
	`<binary blob>`

`download/index.html`	`Content-Type: text/html; charset=utf-8`
	`Content-Length: 26563`
	`<binary blob>`

The data model of object storage

Object storage is basically a **key-value store**:

Key: file path + name	Value: file data + metadata

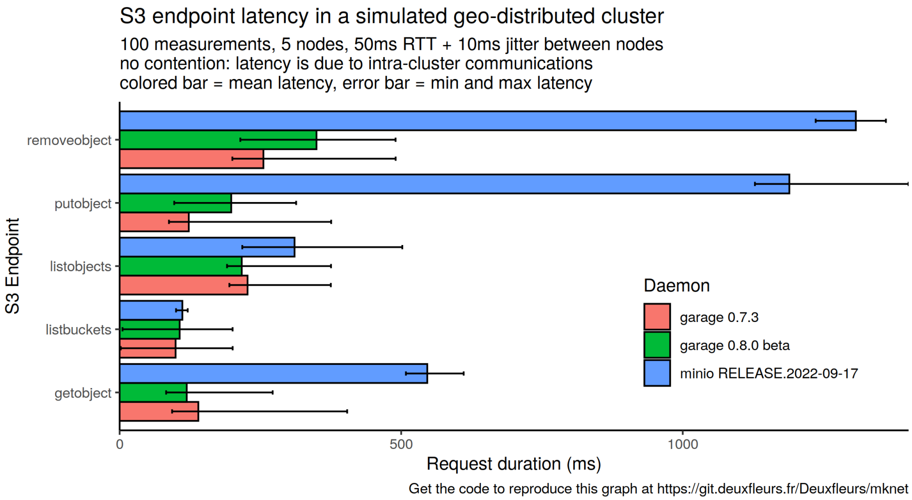
`index.html`	`Content-Type: text/html; charset=utf-8`
	`Content-Length: 24929`
	`<binary blob>`

`img/logo.svg`	`Content-Type: text/svg+xml`
	`Content-Length: 13429`
	`<binary blob>`

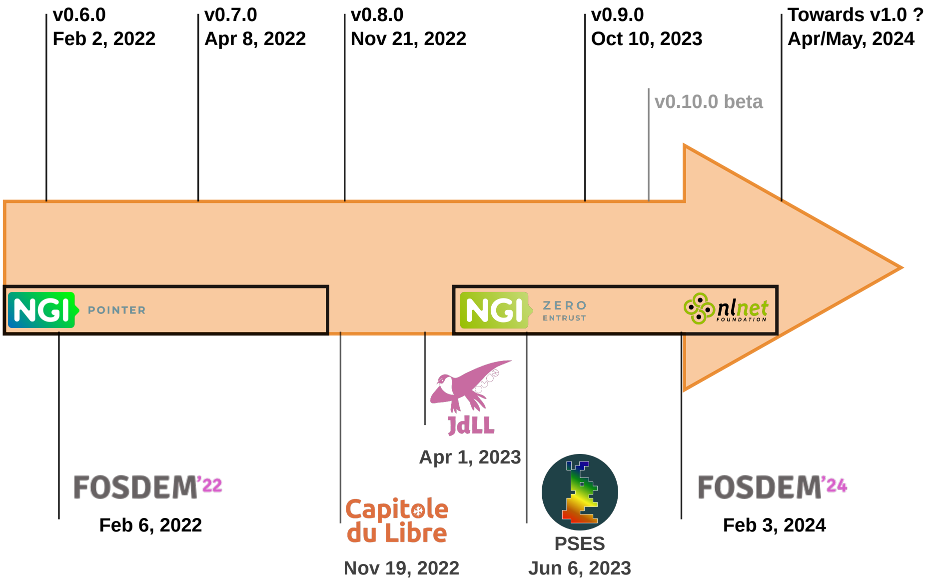
`download/index.html`	`Content-Type: text/html; charset=utf-8`
	`Content-Length: 26563`
	`<binary blob>`

- Maps well to CRDT data types

Performance gains in practice



Recent developments

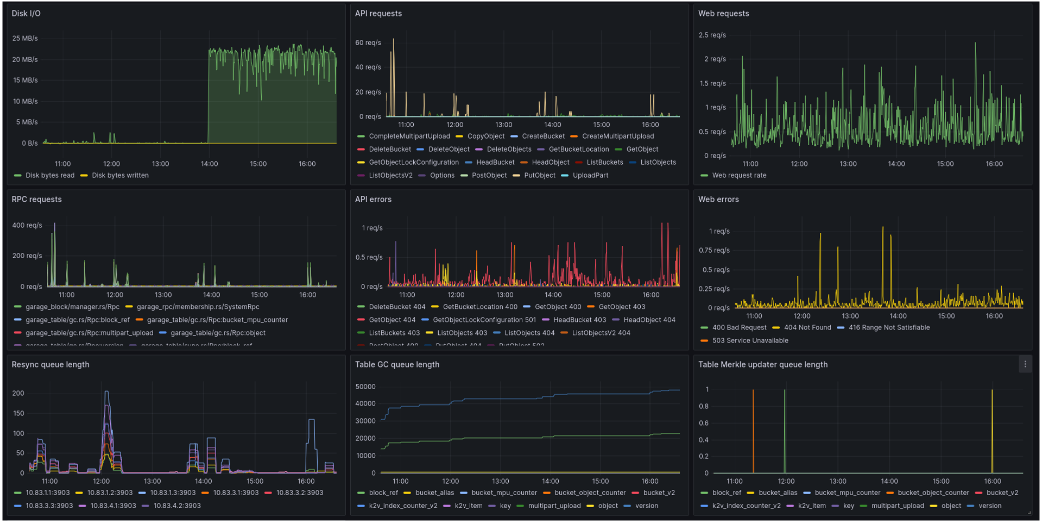

v0.6.0
Feb 2, 2022
v0.7.0
Apr 8, 2022
v0.8.0
Nov 21, 2022
v0.9.0
Oct 10, 2023
v0.10.0 beta
Towards v1.0 ?
Apr/May, 2024
NGI POINTER
NGI ZERO ENTRUST
nlnet FOUNDATION
FOSDEM'22
Feb 6, 2022
Capitole du Libre
Nov 19, 2022
JdLL
Apr 1, 2023
PSES
Jun 6, 2023
FOSDEM'24
Feb 3, 2024

April 2022 - Garage v0.7.0

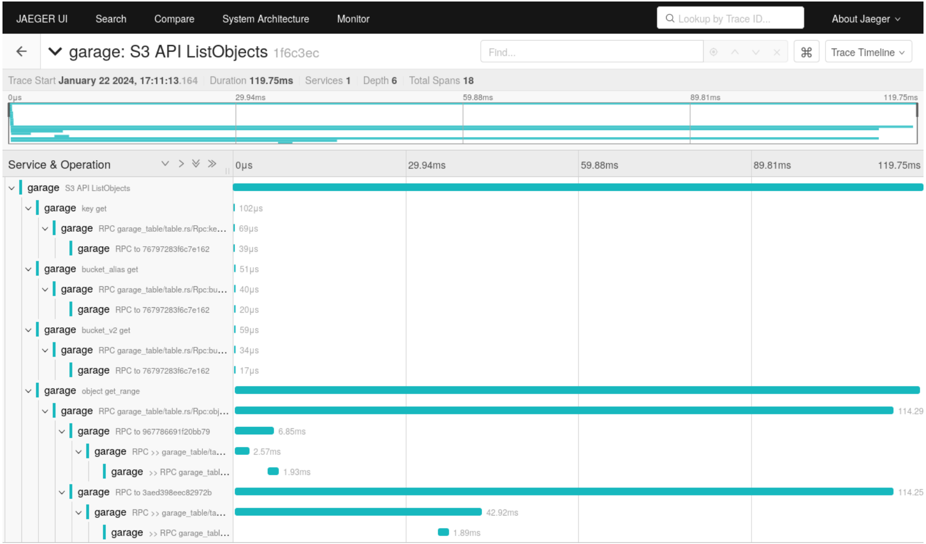
Focus on observability and ecosystem integration

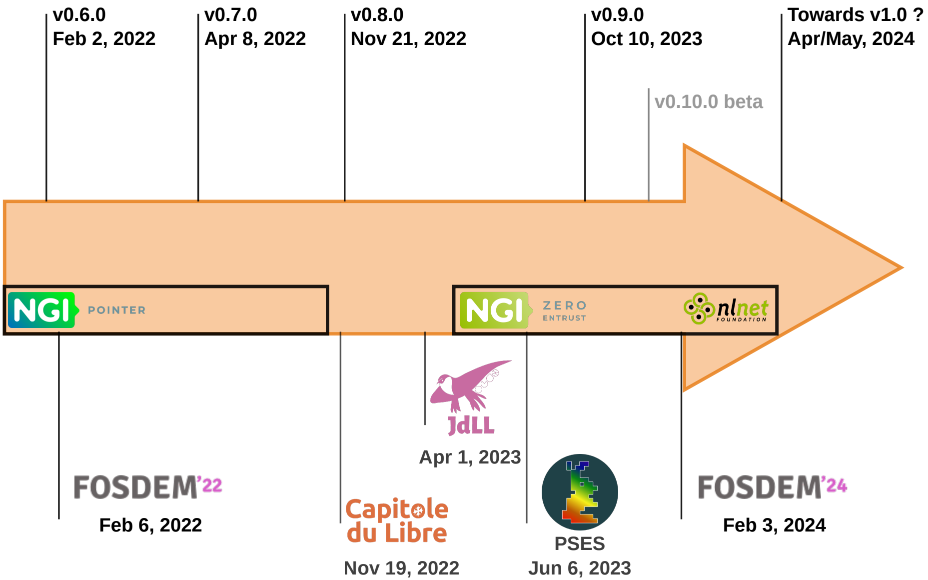
- **Monitoring:** metrics and traces, using OpenTelemetry
- Replication modes with 1 or 2 copies / weaker consistency
- Kubernetes integration
- Admin API (v0.7.2)
- Experimental K2V API (v0.7.2)

Metrics (Prometheus + Grafana)



Traces (Jaeger)




v0.6.0
Feb 2, 2022
v0.7.0
Apr 8, 2022
v0.8.0
Nov 21, 2022
v0.9.0
Oct 10, 2023
Towards v1.0 ?
Apr/May, 2024
v0.10.0 beta
NGI POINTER
NGI ZERO ENTRUST
nlnet FOUNDATION
FOSDEM'22
Feb 6, 2022
Capitole du Libre
Nov 19, 2022
JdLL
Apr 1, 2023
PSES
Jun 6, 2023
FOSDEM'24
Feb 3, 2024

November 2022 - Garage v0.8.0

Focus on performance

- **Alternative metadata DB engines** (LMDB, Sqlite)
- **Performance improvements:** block streaming, various optimizations...
- Bucket quotas (max size, max #objects)
- Quality of life improvements, observability, etc.

About metadata DB engines

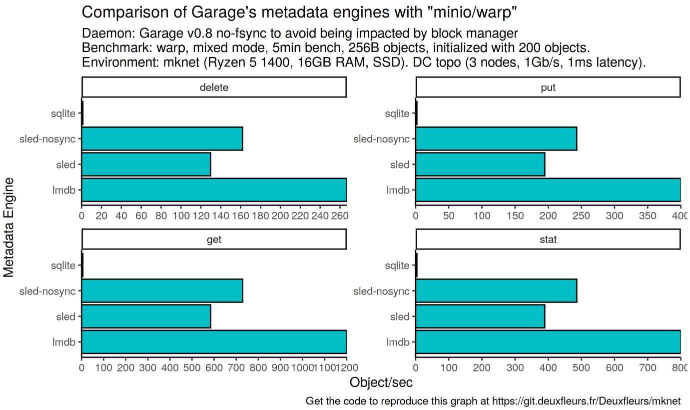
Issues with Sled:

- Huge files on disk
- Unpredictable performance, especially on HDD
- API limitations
- Not actively maintained

LMDB: very stable, good performance, reasonably small files on disk

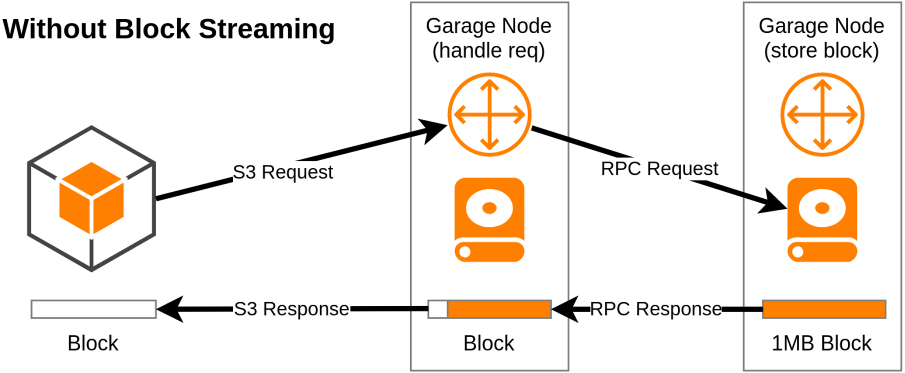
Sled will be removed in Garage v1.0

DB engine performance comparison

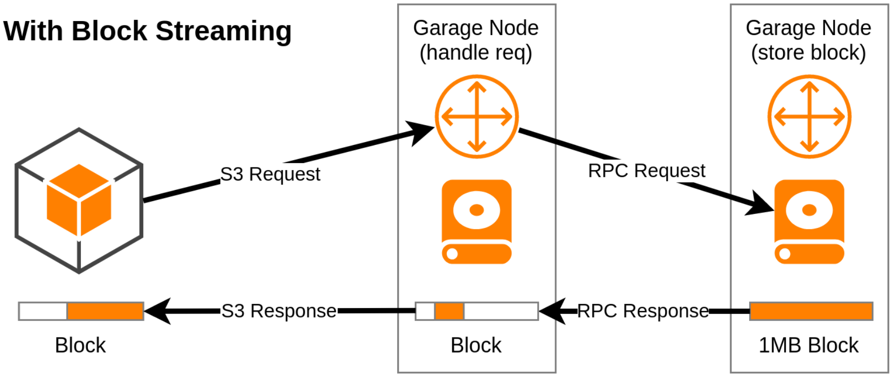


NB: Sqlite was slow due to synchronous mode, now configurable

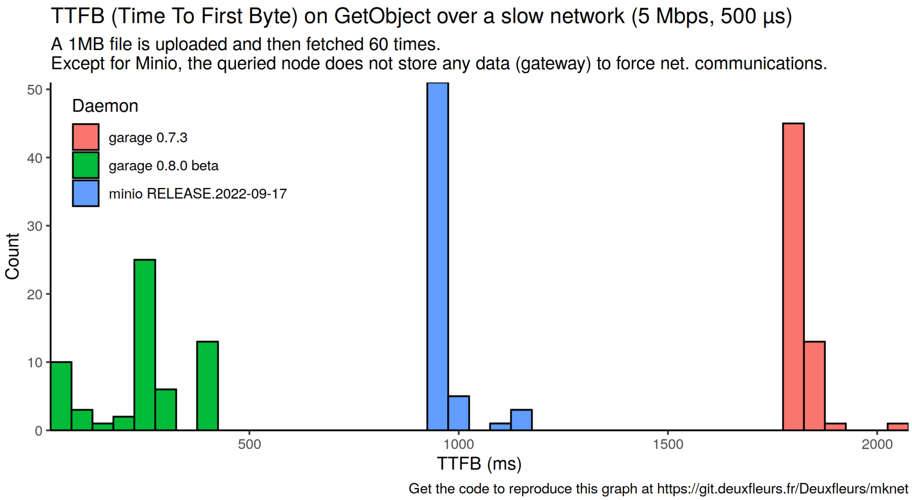
Block streaming



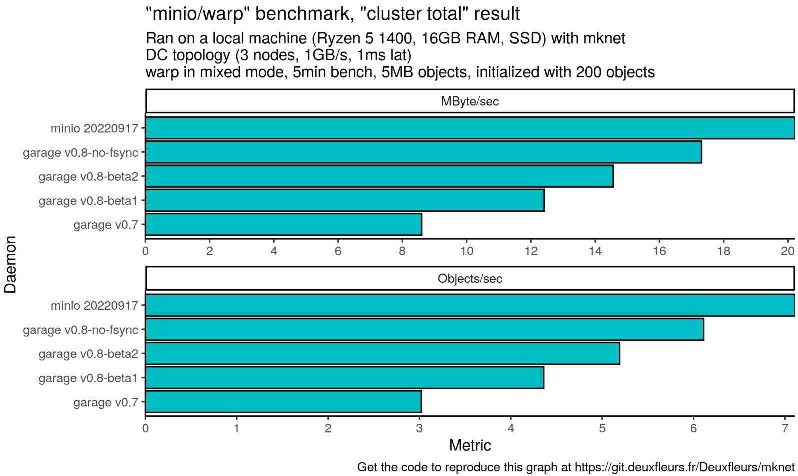
Block streaming

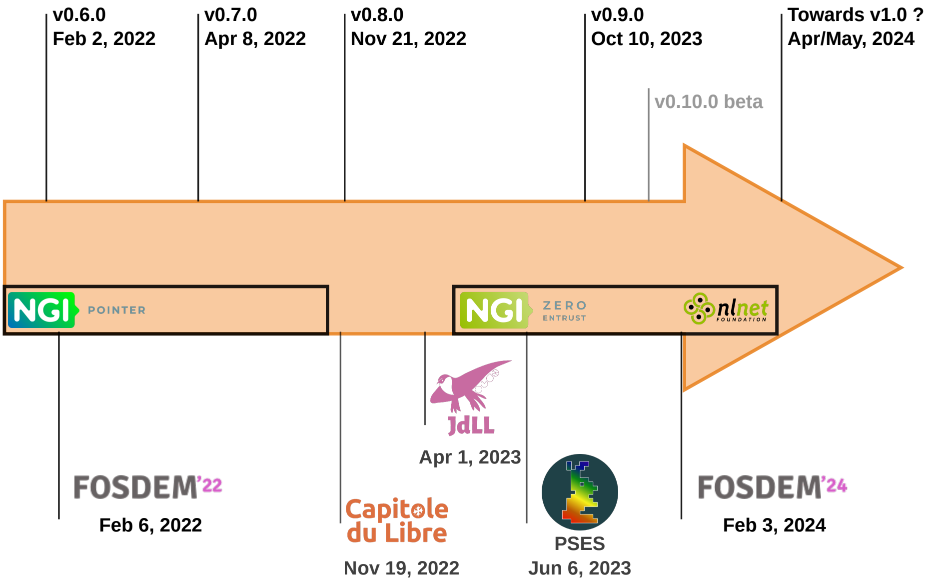


TTFB benchmark



Throughput benchmark




v0.6.0
Feb 2, 2022
v0.7.0
Apr 8, 2022
v0.8.0
Nov 21, 2022
v0.9.0
Oct 10, 2023
Towards v1.0 ?
Apr/May, 2024
v0.10.0 beta
NGI POINTER
NGI ZERO ENTRUST
nlnet FOUNDATION
JdLL
Apr 1, 2023
FOSDEM'22
Feb 6, 2022
Capitole du Libre
Nov 19, 2022
PSES
Jun 6, 2023
FOSDEM'24
Feb 3, 2024

October 2023 - Garage v0.9.0

Focus on streamlining & usability

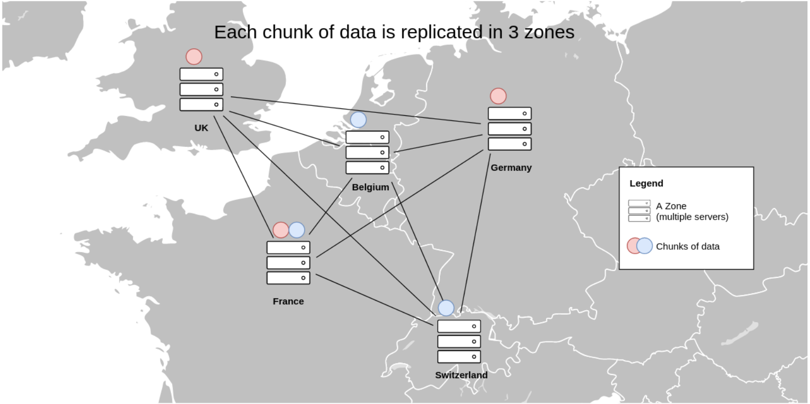
- Support multiple HDDs per node
- S3 compatibility:
  - support basic lifecycle configurations
  - allow for multipart upload part retries
- LMDB by default, deprecation of Sled
- New layout computation algorithm

Layout computation

```
[root@celeri:/home/lx]# docker exec -ti e338 /garage status
==== HEALTHY NODES ====
ID                Hostname   Address                                   Tags                               Zone     Capacity
5fcb3b6e39db3dcb  concombre  [2001:470:ca43::31]:3901                  [concombre,neptune,france,alex]    neptune  500.0 GB
942dd71ea95f4904  df-ymf     [2a02:a03f:6510:5102:6e4b:90ff:fe3a:6174]:3901  [df-ymf,bespin,belgium,max]  bespin   500.0 GB
fdfaf7832d8359e0  df-ymk     [2a02:a03f:6510:5102:6e4b:90ff:fe3b:e939]:3901  [df-ymk,bespin,belgium,max]  bespin   500.0 GB
0a03ab7c082ad929  ananas     [2a01:e0a:e4:2dd0::42]:3901               [ananas,scorpio,france,adrien]     scorpio  2.0 TB
a717e5b618267806  courgette  [2001:470:ca43::32]:3901                  [courgette,neptune,france,alex]    neptune  500.0 GB
2032d0a37f249c4a  abricot    [2a01:e0a:e4:2dd0::41]:3901               [abricot,scopio,france,adrien]     scorpio  2.0 TB
8cf284e7df17d0fd  celeri     [2001:470:ca43::33]:3901                  [celeri,neptune,france,alex]       neptune  2.0 TB
17ee03c6b81d9235  df-ykl     [2a02:a03f:6510:5102:6e4b:90ff:fe3b:e86c]:3901  [df-ykl,bespin,belgium,max]  bespin   500.0 GB
```

Garage stores replicas on different zones when possible

Layout computation



Garage stores replicas on different zones when possible

What a "layout" is

A layout is a precomputed index table:

Partition	**Node 1**	**Node 2**	**Node 3**
Partition 0	Io (jupiter)	Drosera (atuin)	Courgette (neptune)
Partition 1	Datura (atuin)	Courgette (neptune)	Io (jupiter)
Partition 2	Io(jupiter)	Celeri (neptune)	Drosera (atuin)
⋮	⋮	⋮	⋮
Partition 255	Concombre (neptune)	Io (jupiter)	Drosera (atuin)

What a "layout" is

A layout is a precomputed index table:

Partition	Node 1	Node 2	Node 3
Partition 0	Io (jupiter)	Drosera (atuin)	Courgette (neptune)
Partition 1	Datura (atuin)	Courgette (neptune)	Io (jupiter)
Partition 2	Io(jupiter)	Celeri (neptune)	Drosera (atuin)
⋮	⋮	⋮	⋮
Partition 255	Concombre (neptune)	Io (jupiter)	Drosera (atuin)

The index table is built centrally using an optimal algorithm,
then propagated to all nodes

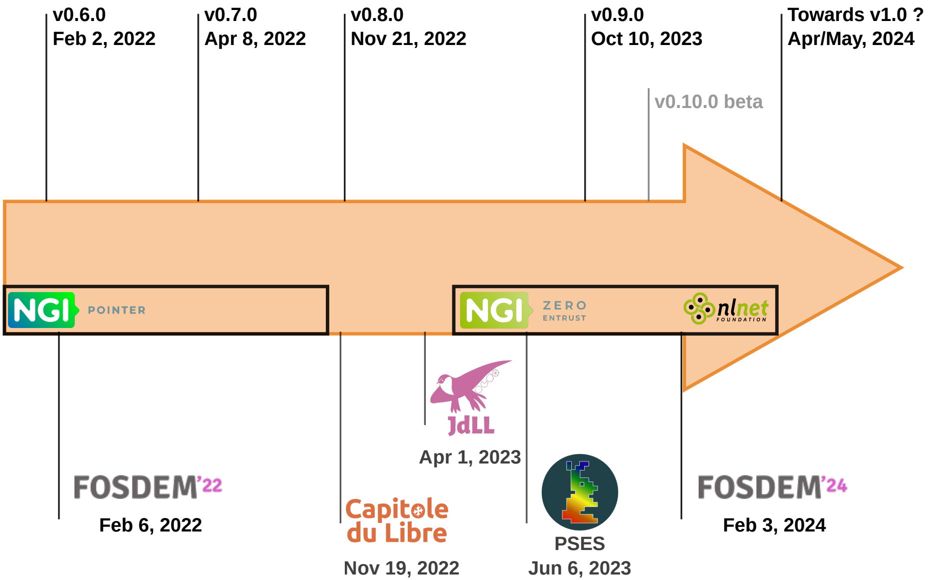
What a "layout" is

A layout is a precomputed index table:

Partition	Node 1	Node 2	Node 3
Partition 0	Io (jupiter)	Drosera (atuin)	Courgette (neptune)
Partition 1	Datura (atuin)	Courgette (neptune)	Io (jupiter)
Partition 2	Io(jupiter)	Celeri (neptune)	Drosera (atuin)
⋮	⋮	⋮	⋮
Partition 255	Concombre (neptune)	Io (jupiter)	Drosera (atuin)

The index table is built centrally using an optimal algorithm,
then propagated to all nodes

Oulamara, M., & Auvolat, A. (2023). *An algorithm for geo-distributed and redundant storage in Garage*. arXiv preprint arXiv:2302.13798.


v0.6.0
Feb 2, 2022
v0.7.0
Apr 8, 2022
v0.8.0
Nov 21, 2022
v0.9.0
Oct 10, 2023
Towards v1.0 ?
Apr/May, 2024
v0.10.0 beta
NGI POINTER
NGI ZERO ENTRUST
nlnet FOUNDATION
JdLL
Apr 1, 2023
FOSDEM'22
Feb 6, 2022
Capitole du Libre
Nov 19, 2022
PSES
Jun 6, 2023
FOSDEM'24
Feb 3, 2024

October 2023 - Garage v0.10.0 beta

Focus on consistency

- Fix consistency issues when reshuffling data

Working with weak consistency

Not using consensus limits us to the following:

Working with weak consistency

Not using consensus limits us to the following:

- **Conflict-free replicated data types** (CRDT)

  Non-transactional key-value stores such as S3 are equivalent to a simple CRDT:
  a map of **last-writer-wins registers** (each key is its own CRDT)

Working with weak consistency

Not using consensus limits us to the following:

- **Conflict-free replicated data types** (CRDT)

  Non-transactional key-value stores such as S3 are equivalent to a simple CRDT: a map of **last-writer-wins registers** (each key is its own CRDT)

- **Read-after-write consistency**

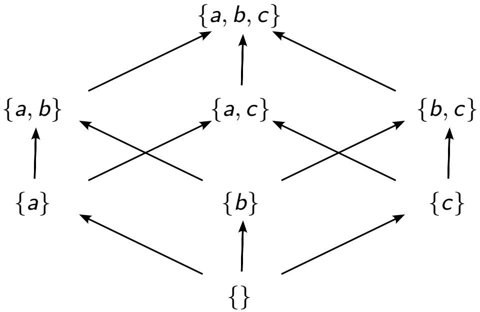
  Can be implemented using quorums on read and write operations

CRDT read-after-write consistency using quorums

Property: If node A did an operation *write*(x) and received an OK response, and node B starts an operation *read*$()$ after A received OK, then B will read a value $x' \sqsupseteq x$.

CRDT read-after-write consistency using quorums

Property: If node A did an operation $write(x)$ and received an OK response,
and node B starts an operation $read()$ after A received OK,
then B will read a value $x' \sqsupseteq x$.

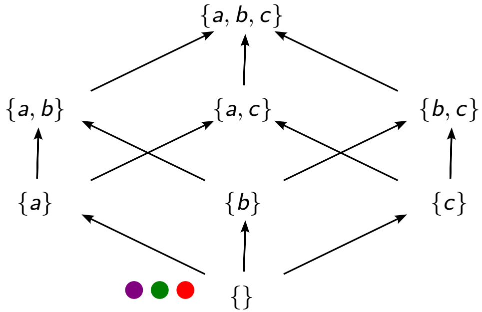


CRDT read-after-write consistency using quorums

Property: If node A did an operation $write(x)$ and received an OK response,
and node B starts an operation $read()$ after A received OK,
then B will read a value $x' \sqsupseteq x$.

$write(\{a\})$:

● $\not\sqsupseteq \{a\}$
● $\not\sqsupseteq \{a\}$
● $\not\sqsupseteq \{a\}$

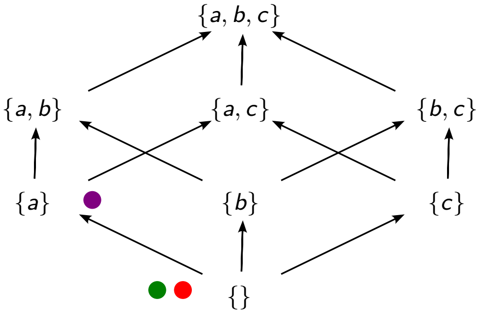


CRDT read-after-write consistency using quorums

Property: If node A did an operation $write(x)$ and received an OK response,
and node B starts an operation $read()$ after A received OK,
then B will read a value $x' \sqsupseteq x$.

$write(\{a\})$:

- ● $\sqsupseteq \{a\} \to$ OK
- ● $\not\sqsupseteq \{a\}$
- ● $\not\sqsupseteq \{a\}$

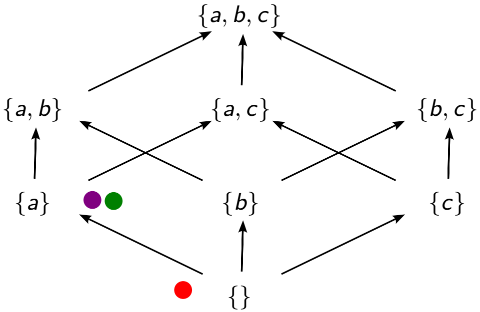


CRDT read-after-write consistency using quorums

Property: If node A did an operation $write(x)$ and received an OK response,
and node B starts an operation $read()$ after A received OK,
then B will read a value $x' \sqsupseteq x$.

$write(\{a\})$:

● $\sqsupseteq \{a\} \rightarrow$ OK
● $\sqsupseteq \{a\} \rightarrow$ OK
● $\not\sqsupseteq \{a\}$
return OK



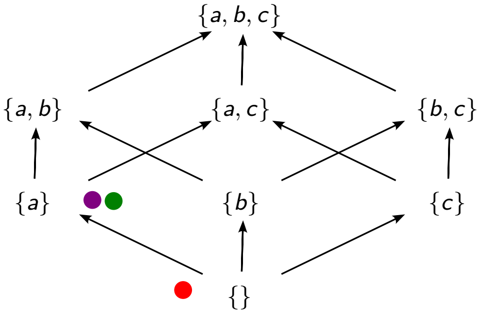
CRDT read-after-write consistency using quorums

Property: If node A did an operation $write(x)$ and received an OK response,
and node B starts an operation $read()$ after A received OK,
then B will read a value $x' \sqsupseteq x$.

$write(\{a\})$:

● $\sqsupseteq \{a\} \rightarrow$ OK
● $\sqsupseteq \{a\} \rightarrow$ OK
● $\not\sqsupseteq \{a\}$
return OK

$read()$:



CRDT read-after-write consistency using quorums

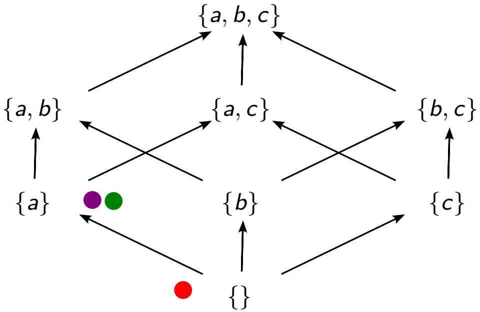
Property: If node A did an operation *write*(x) and received an OK response,
and node B starts an operation *read*$()$ after A received OK,
then B will read a value $x' \sqsupseteq x$.

write$(\{a\})$:

$\sqsupseteq \{a\} \rightarrow$ OK
$\sqsupseteq \{a\} \rightarrow$ OK
$\not\sqsupseteq \{a\}$
return OK

read$()$:

$\rightarrow \{\}$



CRDT read-after-write consistency using quorums

Property: If node A did an operation $write(x)$ and received an OK response,
and node B starts an operation $read()$ after A received OK,
then B will read a value $x' \sqsupseteq x$.

$write(\{a\})$:

● $\sqsupseteq \{a\} \rightarrow$ OK

● $\sqsupseteq \{a\} \rightarrow$ OK

● $\not\sqsupseteq \{a\}$

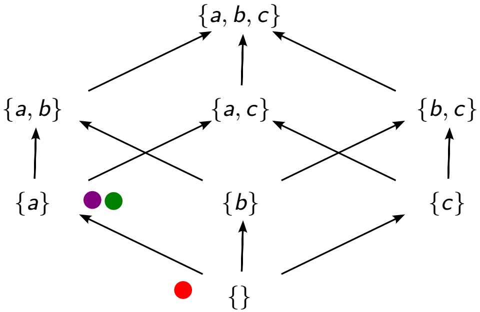
return OK

$read()$:

● $\rightarrow \{\}$

● $\rightarrow \{a\}$

return $\{\} \sqcup \{a\} = \{a\}$



CRDT read-after-write consistency using quorums

Property: If node A did an operation $write(x)$ and received an OK response,
and node B starts an operation $read()$ after A received OK,
then B will read a value $x' \sqsupseteq x$.

$write(\{a\})$:

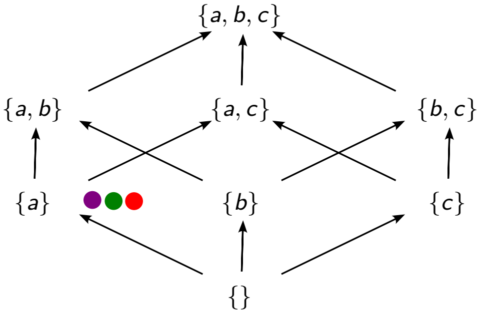
- $\sqsupseteq \{a\} \rightarrow$ OK
- $\sqsupseteq \{a\} \rightarrow$ OK
- $\sqsupseteq \{a\}$

return OK

$read()$:

- $\rightarrow \{\}$
- $\rightarrow \{a\}$

return $\{\} \sqcup \{a\} = \{a\}$



CRDT read-after-write consistency using quorums

Property: If node A did an operation $write(x)$ and received an OK response,
and node B starts an operation $read()$ after A received OK,
then B will read a value $x' \sqsupseteq x$.

Algorithm $write(x)$**:**

1. Broadcast $write(x)$ to all nodes
2. Wait for $k > n/2$ nodes to reply OK
3. Return OK

Algorithm $read()$**:**

1. Broadcast $read()$ to all nodes
2. Wait for $k > n/2$ nodes to reply with values $x_1, \ldots, x_k$
3. Return $x_1 \sqcup \ldots \sqcup x_k$

A hard problem: layout changes

- We rely on quorums $k > n/2$ within each partition:

$$n = 3, \qquad k \geq 2$$

A hard problem: layout changes

- We rely on quorums $k > n/2$ within each partition:

$$n = 3, \qquad k \geq 2$$

- When rebalancing, the set of nodes responsible for a partition can change:

$$\{A, B, C\} \rightarrow \{A, D, E\}$$

A hard problem: layout changes

- We rely on quorums $k > n/2$ within each partition:

$$n = 3, \qquad k \geq 2$$

- When rebalancing, the set of nodes responsible for a partition can change:

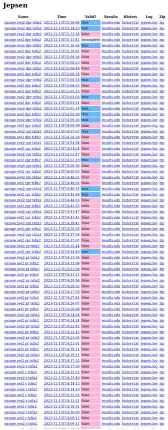
$$\{A, B, C\} \rightarrow \{A, D, E\}$$

- During the rebalancing, D and E don't yet have the data,
  and B and C want to get rid of the data to free up space

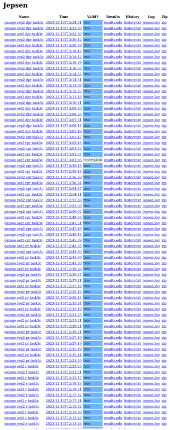
$\rightarrow$ risk of inconsistency, **how to coordinate?**

Handling layout changes without losing consistency

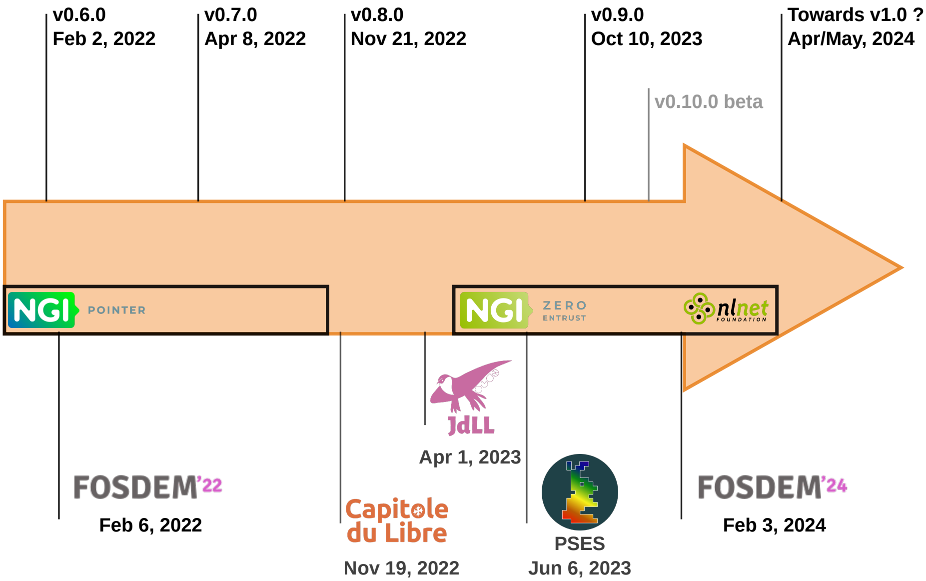
- **Solution:**
  - keep track of data transfer to new nodes
  - use multiple write quorums
    (new nodes + old nodes
    while data transfer is in progress)
  - switching reads to new nodes
    only once copy is finished
- **Implemented** in v0.10
- **Validated** with Jepsen testing



Garage v0.9.0



Garage v0.10 beta


v0.6.0
Feb 2, 2022
v0.7.0
Apr 8, 2022
v0.8.0
Nov 21, 2022
v0.9.0
Oct 10, 2023
v0.10.0 beta
Towards v1.0 ?
Apr/May, 2024
NGI POINTER
NGI ZERO ENTRUST
nlnet FOUNDATION
FOSDEM'22
Feb 6, 2022
Capitole du Libre
Nov 19, 2022
JdLL
Apr 1, 2023
PSES
Jun 6, 2023
FOSDEM'24
Feb 3, 2024

Towards v1.0

Focus on security & stability

- **Security audit** in progress by Radically Open Security
- Misc. S3 features (SSE-C, ...) and compatibility fixes
- Improve UX
- Fix bugs

Operating big Garage clusters

Operating Garage

```
$ garage status
==== HEALTHY NODES ====
ID                Hostname   Address                                    Tags         Zone     Capacity  DataAvail
ec5753c546756825  df-pw5     [2a02:a03f:6510:5102:223:24ff:feb0:e8a7]:3991  [df-pw5]     bespin   500.0 GB  429.1 GB (89.0%)
76797283f6c7e162  carcajou   [2001:470:ca43::22]:3991                   [carcajou]   neptune  200.0 GB  166.3 GB (73.5%)
8073f25ffb7d6944  piranha    [2a01:cb05:911e:ec00:223:24ff:feb0:ea82]:3991  [piranha]    corrin   500.0 GB  457.3 GB (94.0%)
3aed398eec82972b  origan     [2a01:e0a:5e4:1d0:223:24ff:feaf:fdec]:3991    [origan]     jupiter  500.0 GB  457.1 GB (93.1%)
967786691f20bb79  caribou    [2001:470:ca43::23]:3991                   [caribou]    neptune  500.0 GB  453.1 GB (92.3%)
```

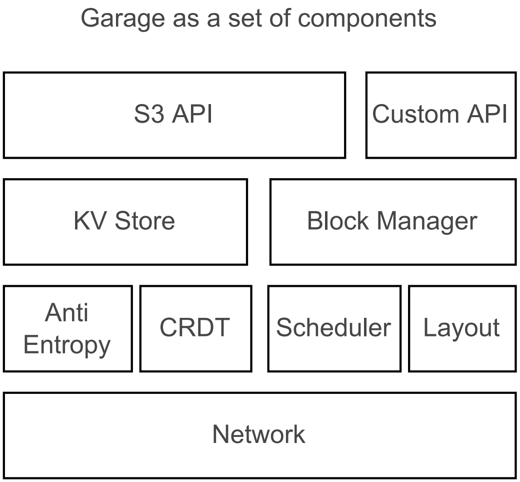
Operating Garage

```
$ garage status
==== HEALTHY NODES ====
ID                Hostname  Address                                     Tags        Zone     Capacity  DataAvail
ec5753c546756825  df-pw5    [2a02:a03f:6510:5102:223:24ff:feb0:e8a7]:3991  [df-pw5]    bespin   500.0 GB  429.1 GB (89.0%)
76797283f6c7e162  carcajou  [2001:470:ca43::22]:3991                    [carcajou]  neptune  200.0 GB  166.3 GB (73.5%)
8073f25ffb7d6944  piranha   [2a01:cb05:911e:ec00:223:24ff:feb0:ea82]:3991  [piranha]   corrin   500.0 GB  457.3 GB (94.0%)
3aed398eec82972b  origan    [2a01:e0a:5e4:1d0:223:24ff:feaf:fdec]:3991  [origan]    jupiter  500.0 GB  457.1 GB (93.1%)
967786691f20bb79  caribou   [2001:470:ca43::23]:3991                    [caribou]   neptune  500.0 GB  453.1 GB (92.3%)
```

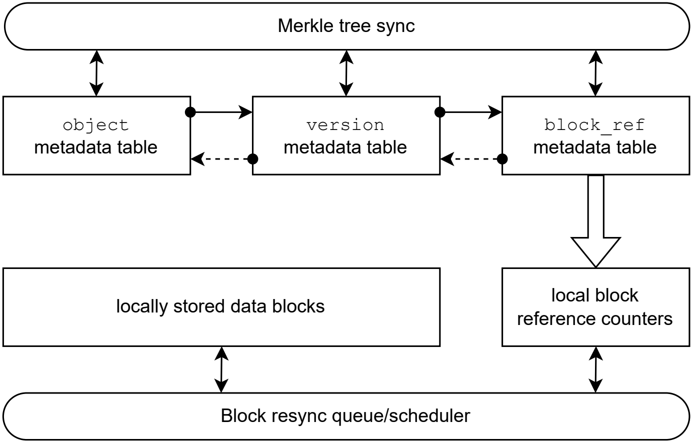
```
$ garage status
==== HEALTHY NODES ====
ID                Hostname  Address                                     Tags        Zone     Capacity  DataAvail
76797283f6c7e162  carcajou  [2001:470:ca43::22]:3991                    [carcajou]  neptune  200.0 GB  166.3 GB (73.5%)
8073f25ffb7d6944  piranha   [2a01:cb05:911e:ec00:223:24ff:feb0:ea82]:3991  [piranha]   corrin   500.0 GB  457.3 GB (94.0%)
3aed398eec82972b  origan    [2a01:e0a:5e4:1d0:223:24ff:feaf:fdec]:3991  [origan]    jupiter  500.0 GB  457.1 GB (93.1%)
967786691f20bb79  caribou   [2001:470:ca43::23]:3991                    [caribou]   neptune  500.0 GB  453.1 GB (92.3%)

==== FAILED NODES ====
ID                Hostname  Address                                     Tags      Zone    Capacity  Last seen
ec5753c546756825  df-pw5    [2a02:a03f:6510:5102:223:24ff:feb0:e8a7]:3991  [df-pw5]  bespin  500.0 GB  5 minutes ago
```

Garage's architecture



Garage's architecture



Digging deeper

```
$ garage stats

Garage version: 20240116133343 [features: k2v, sled, lmdb, sqlite, consul-discovery, kubernetes-discovery, metrics, telemetry-otlp, bundled-libs]
Rust compiler version: 1.68.0

Database engine: LMDB (using Heed crate)

Table stats:
  Table      Items   MklItems  MklTodo  GcTodo
  bucket_v2  19      20        0        0
  key        12      14        0        0
  object     67391   80964     0        0
  version    33909   42045     0        0
  block_ref  334735  370927    0        0

Block manager stats:
  number of RC entries (~= number of blocks): 42376
  resync queue length: 0
  blocks with resync errors: 0

If values are missing above (marked as NC), consider adding the --detailed flag (this will be slow).

Storage nodes:
  ID                Hostname  Zone     Capacity  Part.  DataAvail                   MetaAvail
  ec5753c546756825  df-pw5    bespin   500.0 GB  175    429.1 GB/482.1 GB (89.0%)  429.1 GB/482.1 GB (89.0%)
  76797283f6c7e162  carcajou  neptune  200.0 GB  70     166.3 GB/226.2 GB (73.5%)  166.3 GB/226.2 GB (73.5%)
  8073f25ffb7d6944  piranha   corrin   500.0 GB  173    457.3 GB/486.4 GB (94.0%)  457.3 GB/486.4 GB (94.0%)
  3aed398eec82972b  origan    jupiter  500.0 GB  175    457.1 GB/490.7 GB (93.1%)  457.1 GB/490.7 GB (93.1%)
  967786691f20bb79  caribou   neptune  500.0 GB  175    453.1 GB/490.8 GB (92.3%)  453.1 GB/490.8 GB (92.3%)

Estimated available storage space cluster-wide (might be lower in practice):
  data: 608.3 GB
  metadata: 608.3 GB
```

Digging deeper

```
$ garage worker list
TID  State  Name                          Tranq  Done  Queue  Errors  Consec  Last
1    Idle   Block resync worker #1        0      -     0      -       -
2    Idle   Block resync worker #2        0      -     0      -       -
3    Idle   Block resync worker #3        0      -     0      -       -
4    Idle   Block resync worker #4        0      -     0      -       -
5    Idle   Block resync worker #5        -      -     -      -       -
6    Idle   Block resync worker #6        -      -     -      -       -
7    Idle   Block resync worker #7        -      -     -      -       -
8    Idle   Block resync worker #8        -      -     -      -       -
9    Idle   Block scrub worker            4      -     -      -       -
10   Idle   bucket_v2 Merkle              -      -     0      -       -
11   Idle   bucket_v2 sync                -      -     0      1       0       17 hours ago
12   Idle   bucket_v2 GC                  -      -     0      -       -
13   Idle   bucket_v2 queue               -      -     0      -       -
14   Idle   bucket_alias Merkle           -      -     0      -       -
15   Idle   bucket_alias sync             -      -     0      1       0       17 hours ago
16   Idle   bucket_alias GC               -      -     0      -       -
17   Idle   bucket_alias queue            -      -     0      -       -
18   Idle   key Merkle                    -      -     0      -       -
19   Idle   key sync                      -      -     0      1       0       17 hours ago
20   Idle   key GC                        -      -     0      -       -
21   Idle   key queue                     -      -     0      -       -
22   Idle   object Merkle                 -      -     0      -       -
23   Idle   object sync                   -      -     0      4       0       17 hours ago
24   Idle   object GC                     -      -     0      -       -
25   Idle   object queue                  -      -     0      -       -
26   Idle   bucket_object_counter Merkle  -      -     0      -       -
27   Idle   bucket_object_counter sync    -      -     0      4       0       17 hours ago
28   Idle   bucket_object_counter GC      -      -     0      -       -
29   Idle   bucket_object_counter queue   -      -     0      -       -
30   Idle   multipart_upload Merkle       -      -     0      -       -
31   Idle   multipart_upload sync         -      -     0      5       0       17 hours ago
32   Idle   multipart_upload GC           -      -     0      -       -
33   Idle   multipart_upload queue        -      -     0      -       -
34   Idle   bucket_mpu_counter Merkle     -      -     0      -       -
35   Idle   bucket_mpu_counter sync       -      -     0      -       -
36   Idle   bucket_mpu_counter GC         -      -     0      -       -
37   Idle   bucket_mpu_counter queue      -      -     0      -       -
38   Idle   version Merkle                -      -     0      -       -
39   Idle   version sync                  -      -     0      50      0       17 hours ago
40   Idle   version GC                    -      -     0      -       -
41   Idle   version queue                 -      -     0      -       -
42   Idle   block_ref Merkle              -      -     0      -       -
43   Idle   block_ref sync                -      -     0      45      0       17 hours ago
44   Idle   block_ref GC                  -      -     0      -       -
45   Idle   block_ref queue               -      -     0      -       -
46   Idle   object lifecycle worker       -      -     -      -       -
```

Digging deeper

```
$ garage worker get
8073f25ffb7d6944  lifecycle-last-completed    2024-01-23
8073f25ffb7d6944  resync-tranquility          1
8073f25ffb7d6944  resync-worker-count         4
8073f25ffb7d6944  scrub-corruptions_detected  0
8073f25ffb7d6944  scrub-last-completed        2023-12-27T13:49:33.234Z
8073f25ffb7d6944  scrub-next-run              2024-01-31T03:23:02.234Z
8073f25ffb7d6944  scrub-tranquility           4

$ garage worker get -a resync-tranquility
3aed398eec82972b  resync-tranquility  1
76797283f6c7e162  resync-tranquility  1
8073f25ffb7d6944  resync-tranquility  1
967786691f20bb79  resync-tranquility  1
ec5753c546756825  resync-tranquility  1
```

Potential limitations and bottlenecks

- Global:
  - Max. ∼100 nodes per cluster (excluding gateways)
- Metadata:
  - One big bucket = bottleneck, object list on 3 nodes only
- Block manager:
  - Lots of small files on disk
  - Processing the resync queue can be slow

Deployment advice for very large clusters

- Metadata storage:
  - ZFS mirror (x2) on fast NVMe
  - Use LMDB storage engine
- Data block storage:
  - Use Garage's native multi-HDD support
  - XFS on individual drives
  - Increase block size (1MB $\rightarrow$ 10MB, requires more RAM and good networking)
  - Tune `resync-tranquility` and `resync-worker-count` dynamically
- Other :
  - Split data over several buckets
  - Use less than 100 storage nodes
  - Use gateway nodes

Current deployments: < 10 TB, we don't have much experience with more

Where to find us

Garage

`https://garagehq.deuxfleurs.fr/`

`mailto:garagehq@deuxfleurs.fr`

`#garage:deuxfleurs.fr` on Matrix

AGPL v3 Free Software

Free as in Freedom