FACT SHEET

croit FlashStore OSD: a Ceph OSD rebuilt for NVMe flash

Preview release. Open source, wire compatible with Ceph, included in the croit frontier release at no extra cost.
croit FlashStore OSD is available as a preview for testing and evaluation. It is not yet recommended for production data.

What it is

croit FlashStore OSD is a Ceph object storage daemon, written from scratch in Rust for NVMe flash, without reusing Ceph's C++ code. It replaces BlueStore on a drive and speaks the same protocol as every other Ceph component, so it runs in an existing cluster next to classic OSDs.

What is different inside

- One pinned thread per CPU core. Every placement group belongs to one core, so a PG never waits for a lock.
- Its own per-core storage engine on the raw NVMe device, with no RocksDB and no BlueFS by default.
- Short I/O path: io_uring to the NVMe drive with direct I/O, no page cache.
- Durable before acknowledged: a write is confirmed only once it is on media on every replica.

CLASSIC CEPH OSD (BLUESTORE)	CROIT FLASHSTORE OSD
Messenger msgr2 over kernel TCP	**Messenger, same protocol** msgr2 over kernel TCP via io_uring
Shared op queues and thread pools locks and wake-ups between threads	**One pinned thread per core** every PG belongs to one core, lock-free rings
BlueStore on RocksDB and BlueFS metadata in an LSM tree, background compaction	**Per-core storage engine** on the raw NVMe device, no RocksDB by default
Kernel block layer and page cache RocksDB reads go through the page cache	**io_uring passthrough** direct I/O, 2 MiB hugepages
NVMe drive	**NVMe drive** one queue pair per core

Same cluster, same protocol
Monitors, managers, clients and classic OSDs talk to FlashStore OSDs without changes: msgr2, cephx, OSDMap and CRUSH. Classic and FlashStore OSDs can serve the same pool; the on-disk format differs, so a drive is redeployed, not converted.

+50 %
Throughput, replicated
up to +123 % (4K seq read)
Average over 10 rep3 cells (64K, 4K, mixed)

+54 %
Throughput, EC 2+2
up to +187 % (4K rand write)
Average over 10 EC 2+2 cells

-52 %
p99 latency, replicated
up to -94 % (4K rand read)
Average over 10 rep3 cells

-36 %
OSD CPU per operation
less CPU in all 20 cells
Average over 10 rep3 cells; EC 2+2: -37 %

2-5 GiB
Memory per OSD
median per OSD at cell end
BlueStore: up to 26 GiB kernel cache per OSD (EC 4K cells)

Measured against Ceph 21.2.0 BlueStore on 30 NVMe OSDs (5 hosts x 6 NVMe OSDs, AMD EPYC 9455P, 6 cores per OSD), 2 x 200 Gbit/s, kernel TCP, no RDMA, fio rados engine with stock librados, 48 clients x QD16. Each cell runs once per engine. Averages are geometric means of the per-cell ratio. Full method and every cell: croit FlashStore OSD vs BlueStore, benchmark report (version 09 Nov 2026).

Compatibility, requirements and availability

Compatibility

- Same messenger protocol (msgr2), cephx, OSDMap and CRUSH as classic Ceph
- Monitors, managers, clients and classic OSDs work with it unchanged
- Classic and FlashStore OSDs can serve the same pool
- Replicated and erasure-coded pools; RBD and CephFS run on top; the object classes RGW needs are already implemented
- Ceph versions: Ceph v21.2.0 (Umbrella), v20.2.5 and v20.2.4 (Tentacle); benchmarked on 21.2.0

Requirements

- NVMe drives with power-loss protection
- 4 KiB sector format (512 B namespaces must be reformatted)
- Linux kernel 5.14 or newer (RHEL / Alma 9) with io_uring enabled

Where the gain is

- The gain is in the OSD data path. RGW and the CephFS MDS are unchanged.
- NVMe only. There is no HDD path and none is planned.
- Drives are redeployed as FlashStore OSDs to benefit; clusters can mix both OSD types.
- Details per cell in the benchmark report and the press Q&A.

Migration

- Redeploy drives as FlashStore OSDs; Ceph backfills from the other copies
- One drive or one whole server at a time, depending on the cluster setup and its failure domains
- The pool stays available throughout; you can stop after any step
- Erasure-coded pools: same procedure

Enabling, licence and code

- croit settings (Settings, Add, value on): `croit enable frontier image` and `croit enable flashstore`
- Per drive: Disks, Set to OSD, Implementation: FlashStore
- Open source under LGPL-3.0. Code and deployment guide: github.com/croit/ceph-rust; benchmark scripts and raw data in the data kit (on request)
- Raw fio data of all measurements on request (redacted data kit)

Availability and price

- Available from 9 November 2026 in the croit frontier release
- Included in the croit platform at no extra cost
- Status: croit FlashStore OSD is available as a preview for testing and evaluation. It is not yet recommended for production data. General availability is planned for the first quarter of 2027, when our Ceph engineers are confident it is ready for production data
- croit customers get help from croit when they test it; hands-on testing is supervised: you tell our engineers what to test, and they run it live in front of you.

About croit. croit GmbH, founded in 2017 and based in Munich, builds software-defined storage based on Ceph and virtualization with Proxmox VE, delivered as software, 24/7 support, training and consulting for on-premises data centers. croit develops its Ceph platform in Europe, is a founding member of the Ceph Foundation and a Proxmox Gold Partner. Its US subsidiary, croit North America Inc., serves customers in the Americas.

Press contact
Deniz Savas
deniz.savas@croit.io

FACT SHEET

Rust client libraries: librustyrados and librustyrbd

Drop-in replacements for librados and librbd, written in Rust. Applications, QEMU/KVM and the Ceph tools use them without being rebuilt.

2.1x
Single RBD client, 4M writes
librustyrbd: 17.9 vs 8.5 GB/s
One fio job on one image, QD16, replicated size 3; cache acknowledgement on both sides, durable at flush

1.9x
Single RBD client, 4M reads
librustyrbd: 20.2 vs 10.4 GB/s
One fio job on one image, QD16, stock librbd as reference

1.7x
Single client, RADOS 4M reads
librustyrados: 18.3 vs 10.7 GB/s
One fio job, QD16, 256 objects of 4 MiB, stock librados as reference

1.6x
RADOS 4K reads of one object
librustyrados: 136k vs 87k IOPS
One fio job, QD32, all I/O to one object (one PG, one OSD session)

+56 %
RBD 4K random writes at QD1
librustyrbd: 85.1k vs 54.4k IOPS
Cache acknowledgement on both sides (default write-around cache), durable at flush

What they are

- librustyrados implements the librados C API (all 246 functions) plus the C++ interface that stock librbd, RGW and python-rados load against; librustyrbd implements the librbd C API
- Same library names (`librados.so.2`, `librbd.so.1`): existing binaries pick them up without a rebuild
- Installed as packages next to stock Ceph; the stock libraries stay installed and can be switched back at any time
- Works with Ceph v21.2.0 and v20.2.4 clusters, with classic OSDs and with FlashStore OSDs

KVM and RDMA

- QEMU/KVM virtual machines run their disks on RBD through librustyrbd, tested end to end in croit's lab
- Optional RDMA transport (RoCEv2) to FlashStore OSDs: in the lab 46 % more 4 MiB write throughput than kernel TCP at 44 % less client CPU
- KVM over hardware RDMA in the lab: 94 % more read bandwidth at 1 MiB, QD4, than over TCP
- Kernel TCP stays the default; RDMA is switched on per client

Status and results

- librustyrados and librustyrbd are available as a preview for testing and evaluation in the croit frontier release. They are not yet recommended for production data.
- Public preview since 9 November 2026, general availability planned for Q1 2027 (roadmap, page 5)
- Faster than or equal to stock librados/librbd in all 18 measured cells; every cell is on page 4. RADOS 4K random writes over 256 objects are at parity (1.005x at QD1, 1.004x at QD32): bound by the OSD cluster (both libraries)
- RBD write cells compare cache acknowledgement with cache acknowledgement: both libraries use the librbd default write-around cache, and data is durable at flush.
- Open source under MPL-2.0, in the same repository: github.com/croit/ceph-rust

Measured on 8 November 2026 against stock Ceph 21.2.0 librados/librbd on the same cluster: the same 30-OSD croit FlashStore cluster as the OSD benchmark (5 hosts x 6 NVMe OSDs, AMD EPYC 9455P, 6 OSDs per host on hosts 1-5, kernel NVMe tier, build 422be2684899), replicated size 3 pool with 512 PGs, kernel TCP; librustyrados/librustyrbd build 8ace974c4e. One client host, one fio 3.43 job (rados and rbd engines, same fio binary for both; only the loaded library differs), 60 s per cell, each cell once per library; each library has its own pool; no OSD restart or cache drop before the cells. 18 cells per library, 36 in total, all complete without errors or timeouts. RDMA and KVM figures: croit lab, separate setups.

croit

Client libraries: every measured cell

Stock librados/librbd against librustyrados/librustyrbd, same cluster and fio binary. Ratio above 1.0 = Rust library faster; a ratio between 0.990x and 1.010x (three decimals) = parity. No cell is below stock.

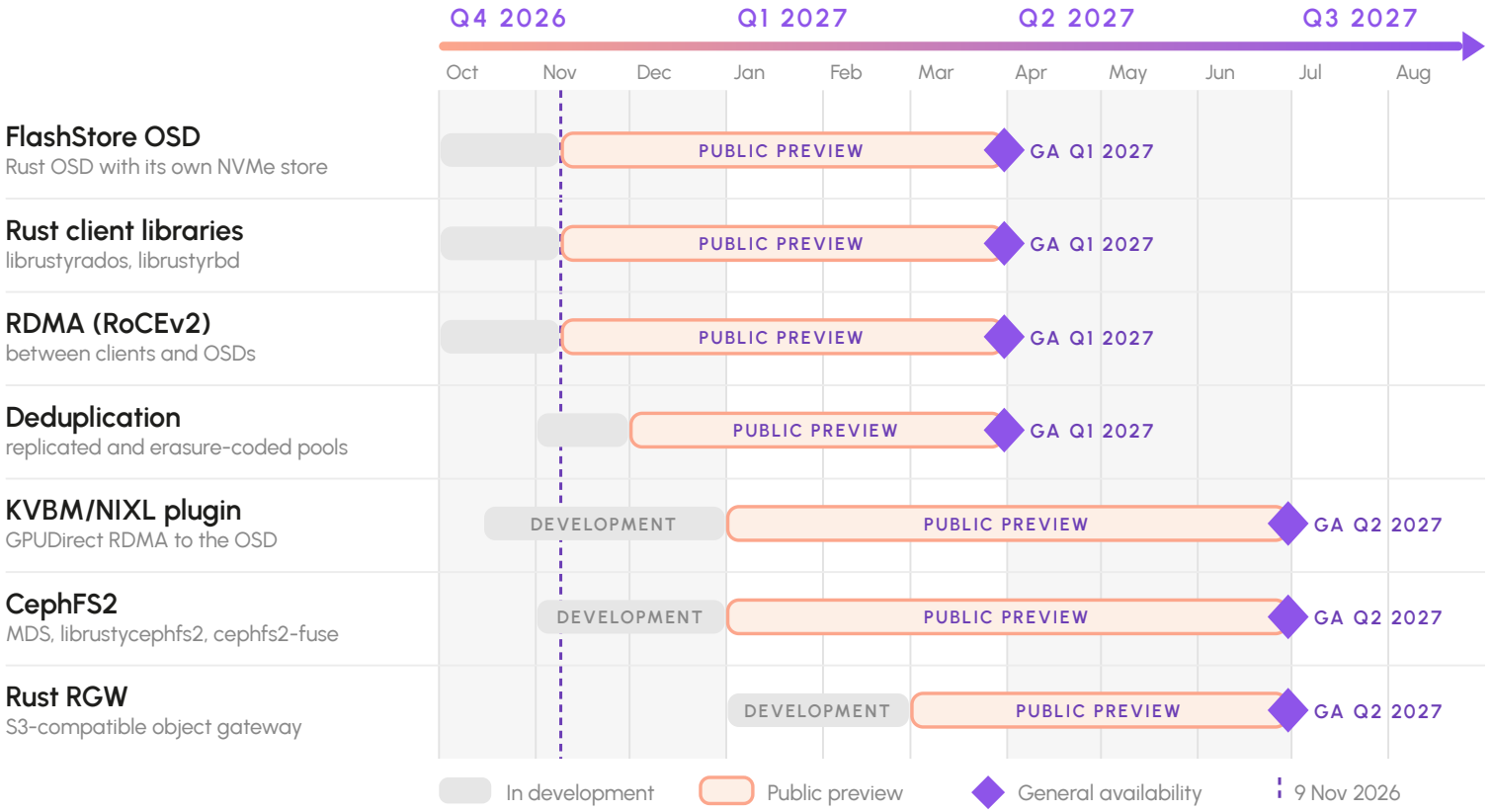
Cell	IOPS stock	IOPS Rust	Rust / stock	p99 stock µs	p99 Rust µs	MB/s stock	MB/s Rust
RADOS 1 obj 4K read QD1	22,518	25,326	**1.12x**	46	45	92	104
RADOS 1 obj 4K read QD32	86,989	135,882	**1.56x**	510	375	356	557
RADOS 1 obj 4K write QD1	7,240	7,414	**1.02x**	167	183	30	30
RADOS 1 obj 4K write QD32	26,396	34,229	**1.30x**	1,548	1,483	108	140
RADOS 1 obj fill 4 MiB	–	–	**1.18x**	5,276	4,489	–	–
RADOS 256 obj 4K read QD1	14,030	14,724	**1.05x**	140	138	57	60
RADOS 256 obj 4K read QD32	213,453	276,790	**1.30x**	338	403	874	1,134
RADOS 256 obj 4K write QD1	6,185	6,216	1.005x	241	245	25	25
RADOS 256 obj 4K write QD32	124,326	124,786	1.004x	537	569	509	511
RADOS 256 obj 4M read QD16	2,552	4,370	**1.71x**	29,229	17,170	10,705	18,328
RADOS 256 obj 4M write QD16	3,236	3,607	**1.11x**	7,504	8,225	13,574	15,130
RADOS 256 obj fill 1 GiB	2,560	3,325	**1.30x**	9,241	11,207	10,737	13,945
RBD 4K read QD1	12,129	14,392	**1.19x**	155	140	50	59
RBD 4K read QD32	187,599	219,134	**1.17x**	395	444	768	898
RBD 4K write QD1	54,374	85,076	**1.56x**	23	8	223	348
RBD 4K write QD32	111,157	131,021	**1.18x**	395	293	455	537
RBD 4M read QD16	2,476	4,826	**1.95x**	27,132	15,270	10,384	20,240
RBD 4M write QD16	2,025	4,278	**2.11x**	9,372	7,307	8,495	17,943

One client host, one fio job per cell, 60 s, each cell once per library, replicated size 3 pool with 512 PGs; client-library build 8ace974c4e. IOPS = operations per second at the stated block size (4M cells: 4 MiB operations), bandwidth in MB/s, p99 = completion latency. Ratio = Rust IOPS / stock IOPS. RADOS 1 obj fill = a single 4 MiB write, so it has no IOPS value: the ratio is the completion latency, 5.28 ms stock vs 4.49 ms Rust. RADOS 256 obj fill writes 1 GiB in 100 ms (stock) vs 77 ms (Rust). Cells marked 256 obj: 256 objects of 4 MiB over the pool; 1 obj: one object; RBD: one 1 GiB image. 256-object 4K random writes are at parity, bound by the OSD cluster (both libraries). RBD write cells compare cache acknowledgement with cache acknowledgement: both libraries use the librbd default write-around cache, and data is durable at flush.

croit

Roadmap

Public previews from November 2026, general availability in Q1 and Q2 2027. What croit plans next around FlashStore OSD: native Rust clients, RDMA and GPUDirect, a new file system, deduplication and a Rust RGW.



All dates are indicative and subject to change based on croit customer demand.