TECHNOLOGY BRIEF

ZEBWARE

ZebClient

ZEBWARE
Döbelnsgatan 21
111 40 Stockholm
SWEDEN
www.zebware.com

TABLE OF CONTENTS

ZEBCLIENT – LOCALIZE CLOUD DATA TO MEMORY

ZebClient optimizes the utilization of modern memory technology to provide sustainable, redundant hyper-speed access to your data. It is a data access point software that operates by localizing cloud data to memory and by disaggregating compute from storage. This results in a solution that supremely speeds up, simplifies, secures, and protects the information under its control.

Core to ZebClient is its innovative persistent memory cache and data placement mechanism. ZebClient optimally leverages the capacity of underlying hardware by using proprietary algorithms and striping techniques to boost performance and to ensure that available hardware is used to its fullest capacity. The performance boost is further supported by the disaggregation of compute and storage and by the novel tiering mechanism. In the process of adding data to the solution, ZebClient also adds data redundancy by applying the Zebware proprietary erasure code-based algorithm and by storing data in external storage.

ZebClient determines, based on recency of usage, what data that should reside inside ZebClient for fast access and what data that should exclusively be placed in or pulled from a connected external storage. The external storage is acting as a data lake where ZebClient can offload colder data and make it available for external processing. The memory / lake separation makes a distinction between hot, warm, and cold data, allowing cold data to be placed in a centrally managed storage and benefit from economies of scale. By selecting a cloud storage for external storage, durability is added to data.

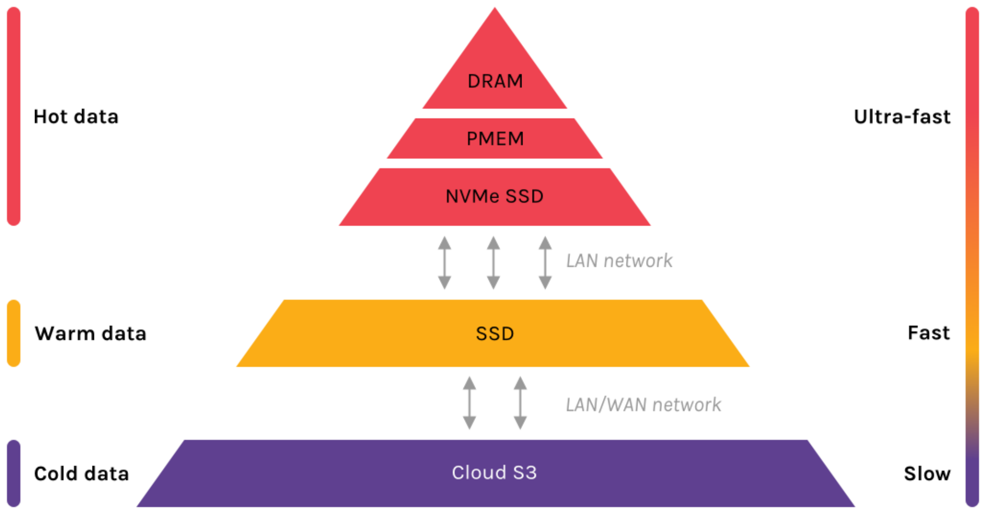
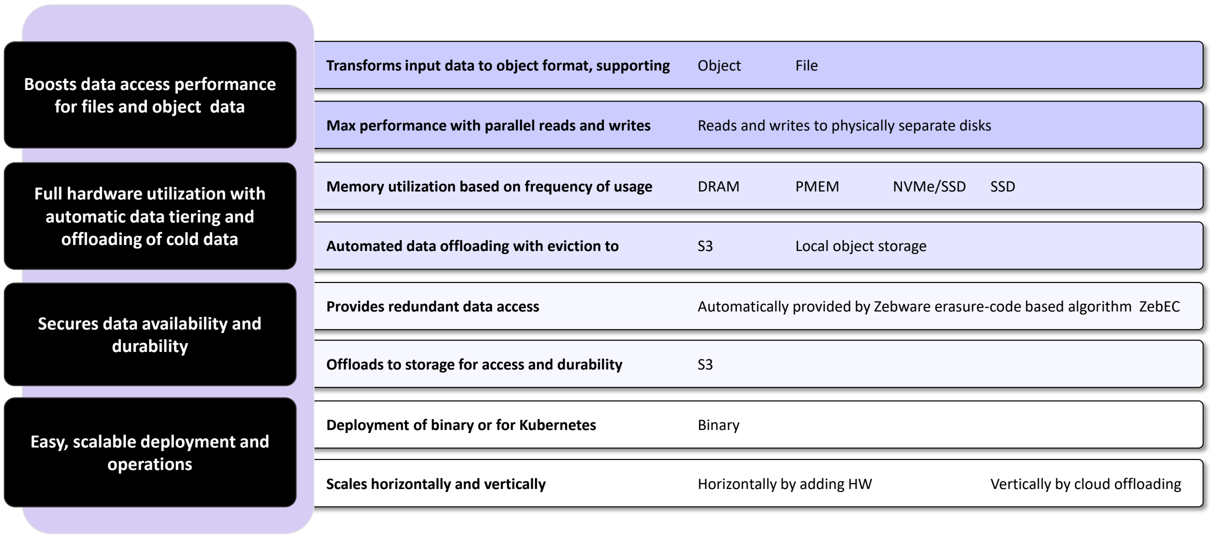


Figure 1: ZebClient memory tiering of hot, warm and cold data based on latest technology memory, traditional SSD/HDD technology and cloud storage.

The hot and warm data on the other hand resides inside ZebClient for fast access. The tiering mechanism allows ZebClient to operate with heterogeneous memory layers, consisting of persistent memory (PMEM) and NVMe SSDs. The act of accessing data will lead to warming whereby the relevant data is automatically moved to faster tiers, e.g. from NVMe SSDs to PMEM. The reverse process will occur if data is not accessed. As the cache is filling up and approaches full utilization, data is automatically removed to make room for new data if an external storage is present. Even when data is evicted to external storage, it is still visible and available for the ZebClient Agent and the connected application.

In addition to providing hyper-performant data access, ZebClient operates conveniently on several different data types. It bridges filesystem and object data and hence effectively breaks down data silos. This makes ZebClient an ideal solution for a wide variety of use cases.

Core features of ZebClient include:



Boosts data access performance for files and object data	**Transforms input data to object format, supporting**	Object	File		
	Max performance with parallel reads and writes	Reads and writes to physically separate disks			
Full hardware utilization with automatic data tiering and offloading of cold data	**Memory utilization based on frequency of usage**	DRAM	PMEM	NVMe/SSD	SSD
	Automated data offloading with eviction to	S3	Local object storage		
Secures data availability and durability	**Provides redundant data access**	Automatically provided by Zebware erasure-code based algorithm ZebEC			
	Offloads to storage for access and durability	S3			
Easy, scalable deployment and operations	**Deployment of binary or for Kubernetes**	Binary			
	Scales horizontally and vertically	Horizontally by adding HW		Vertically by cloud offloading	

Figure 2: ZebClient core features

This brief describes the basic technical architecture and functionality behind the ZebClient software.

SOLUTION OVERVIEW

ZebClient operates in cluster mode to serve demanding applications with a high performant and scalable data access solution (illustrated in Figure 3). As demand grows, new compute nodes can dynamically be added to the cluster. Any data placed in ZebClient can be read from any node, irrespectively of what node was used to write data to the cluster. An architectural cornerstone for ZebClient is the disaggregation of compute from storage. This brings significant value by letting ZebClient act as a localization engine for cloud data and by eliminating the need to locally manage storage. Benefits follow in terms of enhanced data access speed, simplified operation, and reduced cost.

To ensure support for a broad range of data intensive workloads, ZebClient allows external clients and applications to read and write data irrespectively if it is in file or object format. ZebClient is light in terms of CPU requirements and leverages Zebware's fast proprietary erasure coding. This simultaneously provides benefits in terms of performance, data protection via redundancy and security.

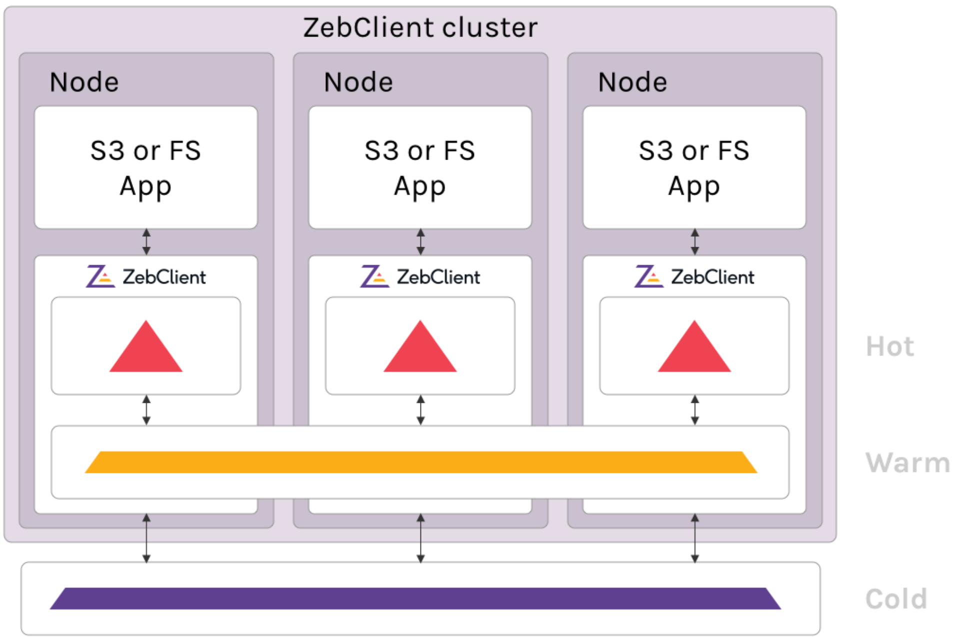


Figure 3: ZebClient in a cluster environment, simultaneously serving several applications with distributed data access and attached external storage for off-loading – e.g., massively scalable S3 object storage.

ZEBCLIENT PROVIDES ENDLESS SCALABILITY FOR YOUR DATASPHERE

Scaling is easily available both from a horizontal and vertical standpoint. Horizontal scalability is achieved by adding nodes to the cluster and vertical scalability from upgrading the nodes' compute, memory, and disk hardware. Utilizing several separate memory and disk areas for each node boosts performance by supporting enhanced striping during read and write. Note that ZebClient can never "fill up" if an external storage is connected. It will simply be dimensioned for holding the most relevant volume of data in the hot memory (on Agent nodes) or warm memory (on Server nodes) while simultaneously keeping a copy stored in external storage when this is available. In the presence of external storage, once the hot and warm memory are approaching its technical limit, the coldest data will be deleted to give room for new data. In absence of external storage, the hot and warm memory will be restricted from further writing until data is manually removed. However, it is important to note that only write operations will be restricted, ZebClient will always support read operations on existing data and hence favor business continuity.

ZEBCLIENT ADDS OBJECT DATA PROPERTIES TO ALL INPUT DATA

Applications communicate with ZebClient via file or S3 data access protocols. The support of these protocols enables an extensive set of traditional and modern applications to immediately draw advantage of the benefits ZebClient brings. Applications connected to ZebClient and other ZebClient cluster agents are both allowed access to the same data, irrespectively of their data access protocol, ZebClient creates a unified platform that breaks down data silos. The data managed by ZebClient will internally be stored in object format – allowing rich meta-data to be used. At rest, each object will be stored in erasure-coded fragments for enhanced read-performance, redundancy, and security. All data is hashed to guarantee integrity.

ZEBCLIENT ADDS REDUNDANCY – ELIMINATING THE NEED FOR SEPARATE BACKUP

Another key benefit of ZebClient is its design for redundant operation. This is achieved on two levels: (1) When data is written to ZebClient, a copy will simultaneously be stored in external storage and (2) by leveraging Zebware's proprietary erasure-coding where data redundancy fragments automatically are stored in the warm memory. Consequently, nodes can go off-line but the information will still be accessible to the client. The desired redundancy level is configurable for the cluster.

SOFTWARE ARCHITECTURE – KEY COMPONENTS

From a software architectural standpoint, ZebClient is designed to simultaneously meet needs for high performance, redundancy, scalability, and operational simplicity. For this to be achievable it is essential to have a distributed solution, supporting data access to nodes in the cluster in parallel, and avoid unnecessary complexity.

THE ZEBCLIENT AGENT - HOT DATA AT MEMORY SPEED

The Agent runs as a service and has a single configurable frontend that takes requests from applications using either file or object data. The memory storage of data is supported locally via a configurable number of memory tiers. A tier is a memory storage area that will hold the erasure-encoded data shards.

Agent tiers will be local to the Agent and is generally based on PMEM and disks of a high-performance class (e.g. NVMe Optane-SSDs). Each Agent must be configured to hold a minimum of k shards locally (data shards). To reconstruct an object, a minimum of k shards is needed. By distributing the local shards over disparate hardware units, e.g. different disks, reading and writing can be done in parallel during the decoding process, resulting in enhanced performance. To achieve this, each tier should ideally be comprised of several smaller, disparate hardware units than a smaller number of large units (e.g., 8x 256 GB PMEM modules are preferable over 2x 1024 GB). Near-linear scaling of the read velocity can be achieved by increasing the number of hardware units.

The Agent also communicates with a configurable number of ZebClient Cluster instances. These instances will host redundancy shards which will be used to reconstruct the object and respond to requests from any Agent whereby global cluster access is achieved. The Agent communicates with the ZebClient Cluster instances via the gRPC protocol and supports transfer of data to cluster-external storage services.

Scaling the number of Agents is straightforward since these components only communicates with the Cluster instances and there is no need for Agent-to-Agent communication. This is particularly important for edge deployments.

THE ZEBCLIENT CLUSTER - WARM DATA

The ZebClient Cluster assignment is to receive, temporarily store and share shards to ensure redundancy and global cluster access to data from any Agent in the cluster. This is executed when Agents have received an incoming write request from applications and are passing on the erasure-coded redundancy shards to the server nodes.

Share is executed when Agents have received read requests from applications and relevant data is not locally available to the Agent. Communication between the cluster instances and Agents is via the gRPC protocol to align with ZebClient’s distributed solution and requirements for low latency and scalability.

THE ZEBCLIENT COLD STORAGE TIER - COLD AND DURABLE DATA

Data written to ZebClient is simultaneously written to the hot and warm memory as well as to the external storage for long-term persistence. Over time, as data in ZebClient moves from hot to warm to cold, it will eventually be fully removed from the hot and warm memory but, via its placement in external storage, still accessible for applications interacting with ZebClient. Data will reside in external storage until explicitly removed by an application interacting with ZebClient. Furthermore, the external storage also supports ZebClient with an additional level of durability since data written to ZebClient simultaneously is written to the external storage. Hence, in the event of a critical node failures, the data will still always be retrievable.

ZebClient delivers read / write memory performance with availability across the cluster

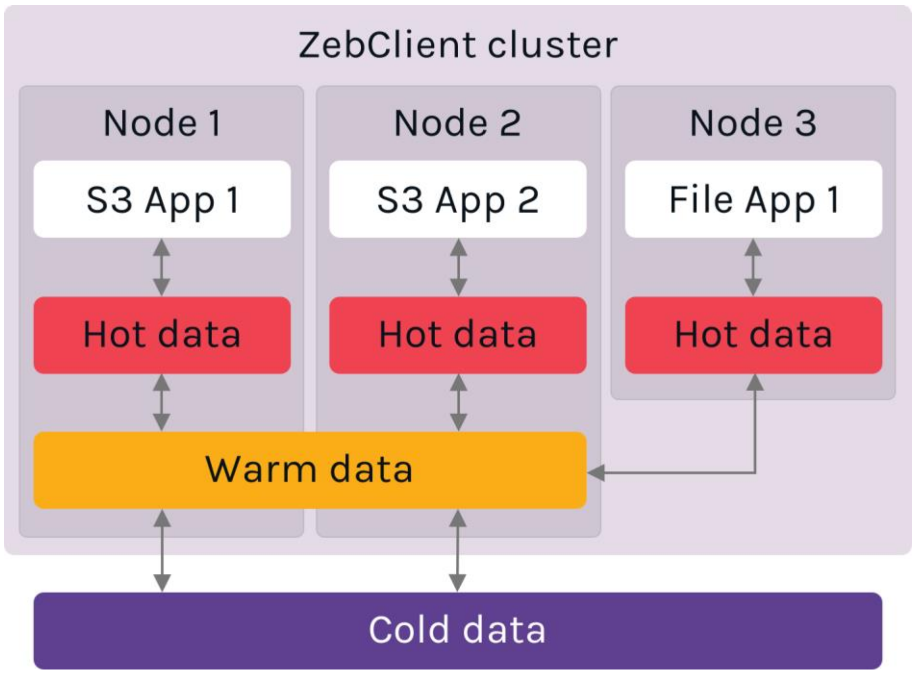


Figure 4: ZebClient in a three Agent and two Cluster instances.

KEY FUNCTIONS

DISAGGREGATED STORAGE

A key element of the ZebClient design is the disaggregation of compute and storage. On the compute nodes, ZebClient is exclusively focused on providing data access whereas storage for the cluster is external and centralized - normally placed in the cloud. The avoidance of local storage greatly simplifies operations, data governance and reduces cost. Furthermore, the design secures data durability as the disaggregated compute nodes act like stateless applications and are easy to recover or restore as all data are available in either the ZebClient cluster or the external storage.

UNIFIED DATA

ZebClient supports applications working with file and object data. Irrespectively of the inbound data format, ZebClient will internally manage the information as objects. This provides several benefits:

(1) An opportunity to augment the data with rich meta-data (not possible using file systems),

(2) Improved data analytics thanks to presence of rich meta-data

For example, by leveraging rich meta-data capabilities, search & discovery is facilitated, and rules can be set to enforce data life-cycle management and security policies.

MEMORY TIERING AND OFFLOADING TO EXTERNAL STORAGE

In addition to its distributed capabilities, ZebClient supports usage of several memory tiers to maximize the performance/cost ratio (see Figure 5). Data is continuously and automatically moved between tiers based on recency of usage by the applications connected to the ZebClient cluster. Tiering of memory is a logical construct where the number and the size of the tiers can be configured to optimally reflect the underlying hardware for maximum performance. Each tier represents a homogenous memory class from a performance perspective – e.g., PMEM, NVMe Optane SSDs (fast), NVMs SSDs (medium fast) and SATA SSDs.

Usually, the hot data tier will consist of PMEM and NVMe SSDs (fast) and the warm data tier of NVMe SSDs and / or SATA SSDs.

Finally, to provide utmost operational simplicity ZebClient also features a connection to an external storage in the form of an object storage. This gives access to low-cost, massively scalable, and durable storage for off-loading and archiving purposes. The design also eliminates the need to locally manage storage and brings operational simplicity and cost saving benefits. Data is placed in external storage upon write requests from applications.

Simultaneously as shards are written to the hot and warm memory, a copy of the full object will be sent to external storage.

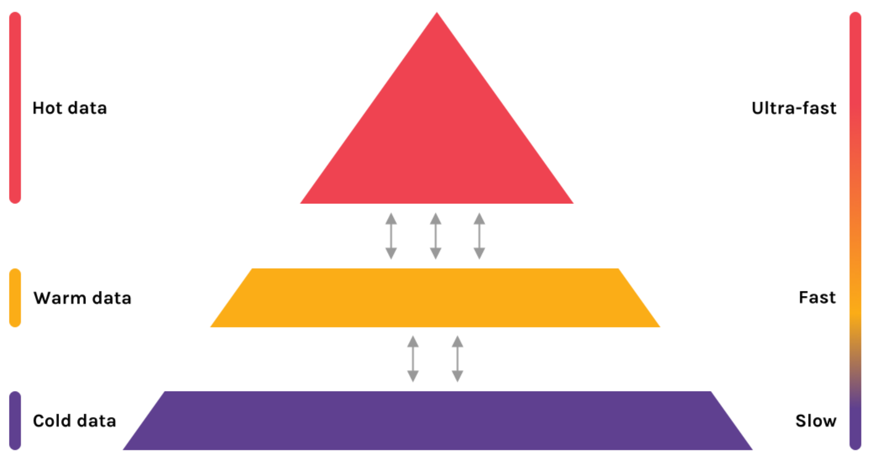


Figure 5: ZebClient shard tiering with hot data at Agent level, warm data at Cluster level and cold data storage.

PERFORMANCE

ZebClient is specifically designed to provide memory-speed access to your data. Key architectural decisions are aimed at performance efficiency, holding hot and warm data local to the applications for hyper-fast access and colder data in a scalable, durable, and cost-efficient external storage.

To achieve unparalleled performance, ZebClient performs smart memory tiering, constantly moving data to faster tiers as they are requested by the applications. Furthermore, driven by customer needs and innovation, Zebware has developed a proprietary erasure-code (ZebEC), designed for exceptional performance and low CPU usage. ZebEC has been proven to significantly outperform standard erasure-coding libraries in bench tests and is designed to be highly performant across CPUs across all architectures. In presence of large multi-core CPUs or GPU processors, the algorithm scales near-linearly and is fully optimized for lowest possible number of CPU cycles per operation. Data shards from the erasure-coding can be dispersed over distinct and separate hardware units (PMEMs, several NVME SSDs etc.), allowing data to be read and written in parallel to the units (a.k.a. “striping”). This ensures that the underlying hardware is optimally leveraged to achieve maximum data access performance.

REDUNDANCY

It is imperative that the applications using ZebClient always will have access to the data. This is achieved through a multidimensional approach.

(1) Data written to ZebClient is simultaneously written to the hot and warm memory areas and to the durable external storage. This ensures that if anything were to happen with the data in the ZebClient Agent and Server nodes, a copy will still always be retrievable.

(2) Integral to ZebClient is Zebware's proprietary and hyper-efficient erasure-coding algorithm. The algorithm is rateless and can be configured for any redundancy level for a specific use case without noticeable performance impact. All incoming data is split into shards and redundancy fragments automatically and immediately stored in the ZebClient Servers. Since all Agents are connected to all Servers, the Agent that locally holds the data can go off-line but the information will still be available to all other Agents. The desired redundancy level is configurable for the cluster.

(3) Finally, ZebClient can be configured to disperse shards to specific memory hardware units. E.g., by equipping the Agent and Servers with several PMEMs, several separate NVMe SSDs etc., the required number of shards will still be available to reconstruct the original information even if a hardware unit were to fail.

SECURITY

TLS is used to secure traffic coming to and leaving ZebClient. That is, between applications using ZebClient frontends (northbound in Figure 2) and between ZebClient and external object storage services (southbound in Figure 2). Furthermore, all traffic internal to ZebClient - between Agent and Servers - is also protected with TLS.

In addition to securing the traffic, ZebClient also stores the data at rest in shards. This is fragmented information and not immediately consumable as files and objects. To ensure data integrity, all data is hashed and hence malicious tampering, bit rot etc. will be automatically and immediately detected.

METRICS & TRACING

To monitor key metrics and be able to "deep dive" into performance aspects of the cluster, support to add a Prometheus' time-series metrics database has been embedded in the ZebClient solution. Prometheus is used to collect general metrics from ZebClient and can be connected to a graphing solution of choice, e.g, Grafana. This will provide users with a palatable and customizable view for insights gathering.