TECHNOLOGY BRIEF

ZEBWARE

ZebClient

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

TABLE OF CONTENTS

ZEBCLIENT – BRING 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 cloud data memory bridge software that operates by bringing cloud data to memory and by disaggregating compute from storage. This results in a solution that supremely speeds up, simplifies, secures, and protects the data 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 storage from compute 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 cloud-based storage.

ZebClient determines, based on recency of usage, what data that should reside in hot tier 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. Furthermore, by selecting a cloud storage for external storage, durability is added to data.

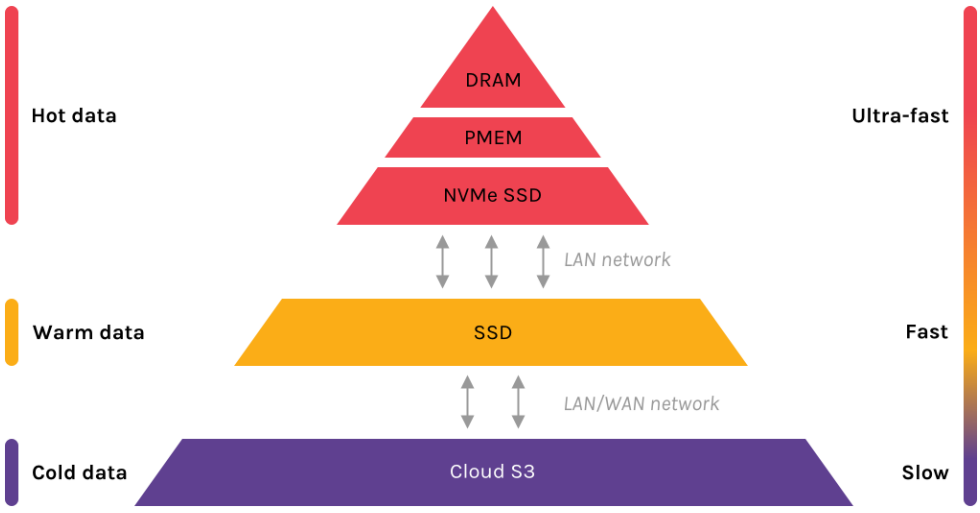

Figure 1: ZebClient memory tiering of hot, warm and cold data

The hot data resides close to application for fast access. Data is tiered between hot, warm and cold depending on access patterns. As a tier is filling up and approaches full utilization, data is automatically removed to make room for new data. Even when data is cold, it is still visible and available for the ZebClient and the connected application.

ZebClient operates 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 file 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	
Provides disaggregation of data storage from compute	**Flexibility**	Multicloud support	Compute nodes on-prem or in the cloud
	Vendor lock-in prevention	Select and change storage and compute node suppliers as you go along	
Secures data availability and protection	**Provides redundant data access**	Automatically provided by Zebware erasure-code based algorithm ZebEC	
	Offloads to storage for access and durability	Store sharded data in a selected S3 storage or across multiple S3 clouds	
Easy, scalable deployment and operations	**Scales horizontally and vertically**	Add application nodes for horizontal scaling	Add S3 storage for vertical scaling
	Automated with ZebClient Deployment Manager	In the cloud	On-prem

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 storage from compute. This brings significant value by letting ZebClient act as a localization engine for cloud data and by eliminating the need to manage storage locally. 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 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.

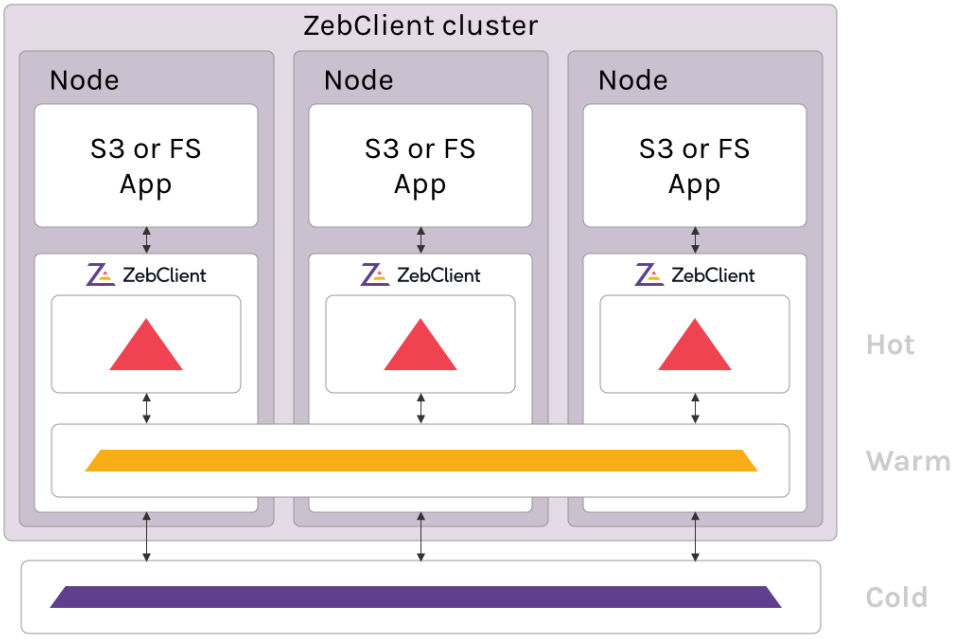

Figure 3: ZebClient in a cluster environment, simultaneously serving several applications with distributed data access and attached external storage

ZEBCLIENT PROVIDES ENDLESS SCALABILITY FOR YOUR DATASPHERE

Scaling is easily available both from a horizontal and vertical standpoint.
Horizontal scalability is achieved by adding application 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. ZebClient can never fill up as an external storage is connected. It will simply be dimensioned for holding the most relevant volume of data in the hot memory (on application nodes) or warm memory (in cluster nodes) while simultaneously keeping the data stored in external storage.

ZEBCLIENT ADDS OBJECT DATA PROPERTIES TO ALL INPUT DATA

The application interface to ZebClient supports both high-performance file and high-performance S3. Applications connected to ZebClient are allowed access to the same shared data, irrespectively of their data access protocol, ZebClient creates a unified platform that breaks down data silos. The data managed by ZebClient are stored in object format – allowing rich meta data to be used.

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, data will be stored in connected external storage
(2) Zebware's proprietary erasure-coding secures that data redundancy fragments are automatically stored in the ZebClient cluster nodes. Consequently, nodes can go offline but the information will still be accessible to the client.

The levels of redundancy that ZebClient provides, are verified to support different designs and use cases. The pre-packaged ZebClient setups of Start, Small, Medium and Large support a wide set of Application nodes configurations, performance and redundancy:

ZebClient Cluster	No of Application nodes	No of Cluster nodes	Performance
Start	1	1 (No redundancy)	1x
Small	1-5	2	2x
Medium	1-25	4	6x
Large	1-100	8	12x

ZebClient test results demonstrates a near linear performance boost as the number of applications nodes grows.

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, a distributed solution is used to support parallel data access to nodes in the cluster and to avoid unnecessary complexity.

THE ZEBCLIENT APPLICATION NODE - HOT DATA AT MEMORY SPEED

The ZebClient application node runs as a service and takes requests from applications using either file or object data. The memory storage of data is supported locally via the hot tier. A tier is a memory storage area that will hold the erasure-encoded data shards.

The hot tier is local to the application node and is generally based on high-performance class memory or disks like PMem or NVMe SSDs. Each application node is 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 rather than a smaller number of large units (e.g., 8x 256 GB modules are preferrable over 2x 1024 GB). Near-linear scaling of the read velocity is achieved by increasing the number of hardware units.

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

Scaling the number of application nodes is straightforward since these components only communicate with the cluster instances. This is particularly important for edge deployments.

THE ZEBCLIENT CLUSTER - WARM SHARED DATA

The ZebClient cluster assignment is to receive, temporarily store and share shards to ensure redundancy and global cluster access to data from any application node in the cluster. The process kicks off as soon as an application node has received an incoming write request from an application and is passing on the erasure-coded redundancy shards to the cluster nodes. Share is executed when an application node has received a read requests from an application and relevant data is not locally available to the application node. Communication between the Cluster nodes and the application nodes is via the gRPC protocol to comply with the distributed design of ZebClient and the requirements for scalability and low latency.

THE ZEBCLIENT COLD STORAGE TIER - COLD AND DURABLE DATA

ZebClient simultaneously writes data to hot and warm tiers as well as to cold data storage. Over time, as data in ZebClient moves from hot to warm to cold, it is eventually fully removed from the hot and warm data tiers but, via its placement in cold storage, it is still accessible for applications connected to ZebClient. Data then resides in cold data storage until explicitly removed. Furthermore, the cold data storage also supports ZebClient with an additional level of durability since data written to ZebClient is simultaneously written to the cold storage. In the event of a critical node failure, the data will still always be retrievable from cold storage.

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

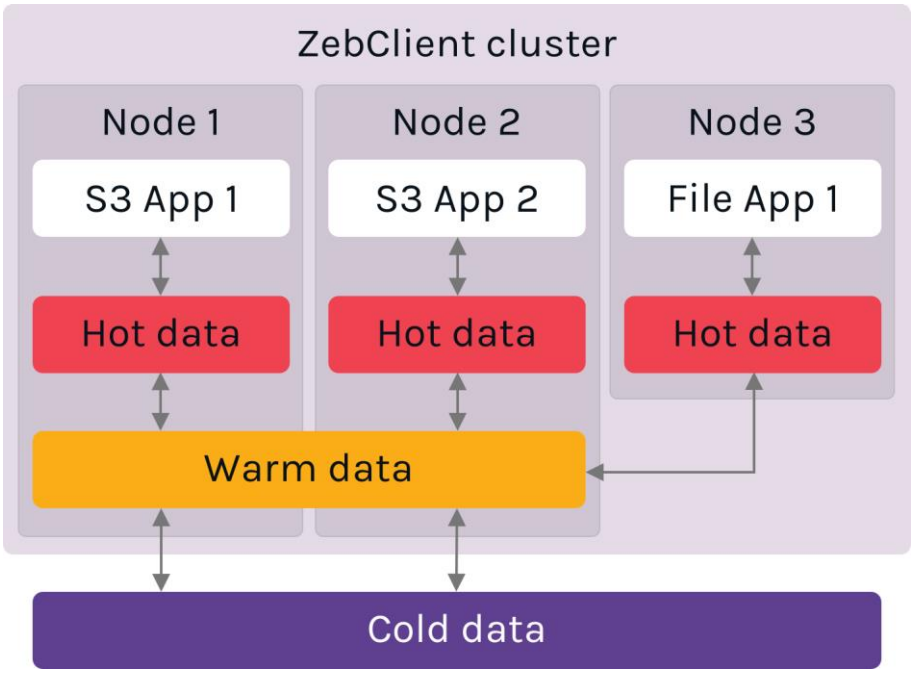

Figure 4: ZebClient in a three application and two cluster nodes setup.

KEY FUNCTIONS

DISAGGREGATED STORAGE

A key element of the ZebClient design is the disaggregation of storage from compute. On compute nodes, ZebClient is exclusively focused on providing data access. The data storage associated with the cluster is external and centralized - normally placed in the cloud. As the need for local storage is eliminated, operations of service is greatly simplified and data governance is enabled end-to-end. The result is an opportunity to achieve huge cost savings. Furthermore, this 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 cold data storage.

UNIFIED DATA

ZebClient supports both those applications working with file and those working with 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 the presence of rich meta-data

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. 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 utilize the underlying hardware to deliver an optimal performance based on the requirements of the applications. Each tier represents a homogenous memory class from a performance perspective. Typically, the hot data tier consists of PMEM or NVMe SSDs (fast) and the warm data tier is based on NVMe SSDs or SATA SSDs.

To provide the best operational simplicity and availability of service, ZebClient 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. This design eliminates the need to locally manage storage and brings operational simplicity and huge 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 is sent to external storage.

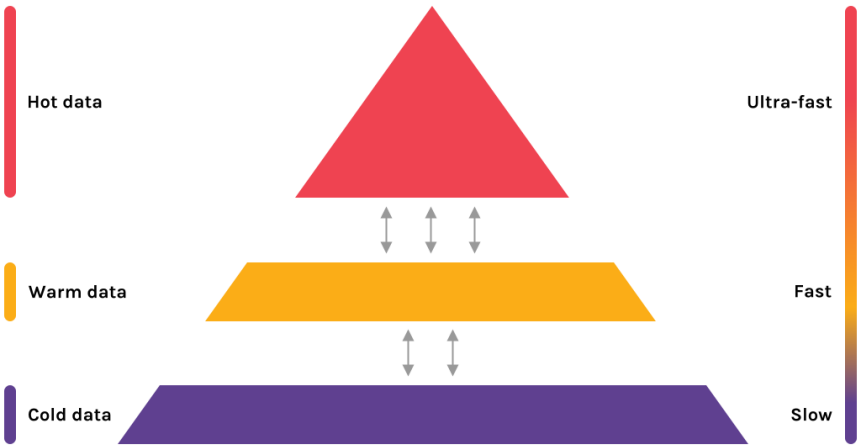

Figure 5: ZebClient shard tiering with hot data at Agent level, warm data at Cluster level and cold data storage.

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 cloud-based storage.

To achieve ultra-high 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 is 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 separate hardware units, allowing data to be read and written in parallel to the units (“striping”). This ensures that the underlying hardware is optimally leveraged to achieve a maximum data access performance.

REDUNDANCY

A key delivery of ZebClient is to secure that applications constantly 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 process secures that if anything happens to the data in the ZebClient application or cluster nodes, a shard is still always retrievable from another node or from the cloud.

(2) One vital element of ZebClient is the Zebware proprietary and hyper-efficient erasure-coding algorithm. The algorithm is rateless and can be configured for any redundancy level to match specific use cases. All incoming data is automatically split into shards and redundancy fragments and immediately stored in the ZebClient cluster nodes. Since all application nodes are connected to all cluster nodes, the application node that locally holds the data can go offline but the information is still available to all other application nodes. The desired redundancy level is delivered according to what ZebClient package is deployed: Start, Small, Medium or Large.

(3) Finally, ZebClient can be configured to disperse shards to specific memory hardware units or across several cold cloud-based storage. By equipping the application and cluster nodes with several memory units or by dispersing data shards over multiple clouds for cold storage, the required number of shards will still be available to reconstruct the original information even if a hardware unit or the connection to one cloud provider 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 application and cluster nodes - 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 is 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 provides an easy-to-understand and customizable view for insights gathering.