Datera® Data Services Platform (DSP™) Architecture

Abstract

Legacy applications, combined with the proliferation of cloud native applications, big data and real time analytics workloads, modern microservices and container deployments are driving the need for a much more agile data infrastructure.

A wide spectrum of price/performance requirements and the limited effectiveness of point storage solutions across today's diverse applications, data use cases and delivery environments are presenting a unique challenge and an opportunity to deliver a highly agile services-based solution – or more succinctly, a data services platform.

The Datera Data Services Platform is a software-defined data infrastructure for virtualized environments, databases, cloud stacks, DevOps and microservices deployments that provides "ZeroOps" delivery and orchestration of data at scale for any application within a traditional datacenter, private cloud or hybrid cloud setting.

Datera combines the industry's only service level objective-based data services architecture, future-proof extensibility and game-changing price/performance to deliver operations-free agility, enterprise-class performance and latency, with better-than-public-cloud economics.

Audience

This white paper is intended for use by IT professionals responsible for the architecture, design, selection, implementation or operation of enterprise storage infrastructure, private clouds, hybrid clouds, virtualized and/or containerized infrastructure, or orchestration platforms who want to learn about the features, benefits and design of the Datera Data Services Platform.

This paper may also provide useful information for DevOps teams, application developers or others who have an interest in understanding infrastructure-as-code provisioning and as-a-Service consumption of storage.

Version 3.x

Datera Technical White Paper

Contents

Datera Technical White Paper

Introduction

A decade ago, Amazon launched AWS, delivering a self-service portal that enabled users to launch virtual servers at the click of a button, fundamentally changing the face of IT. Since then, only a few leading-edge cloud service providers and large-scale enterprises have been successful in replicating this programmable, elastic infrastructure.

Having mastered this new form of IT delivery, those companies have been rewarded with rapid product development cycles and dramatically lower infrastructure costs. For others, however, the transition to cloud-based and cloud-native infrastructure models has challenged the fundamental assumptions underlying their traditional datacenters.

We now live in the era of data, where exponential growth makes it practically impossible to migrate data from old to new, or even successfully “forklift” traditional storage infrastructures. In addition to the problem of managing the sheer scale of data, low asset utilization due to the creation of per-project or per-application silos has been a significant barrier to driving better returns on infrastructure investments.

The resulting fragmented datacenters, with their many implementations and processes, have led to fragile, handcrafted point-in-time deployments for specific applications, making it even more daunting for IT organizations to achieve the agility to operate at the speed of modern cloud-enabled businesses.

With the ever-increasing velocity and variety of data, the need for a common repository for structured, semi-structured and unstructured data is becoming essential, while having an elastic price band to fit the diverse economic value of this data is of paramount importance.

Meanwhile, traditional storage solutions that are tightly coupled to proprietary hardware are inhibiting the ability to quickly take advantage of new technology innovations. Fundamentally, this has led to a painful state of vendor lock-in, loss of flexibility and reduced negotiating leverage. As a result, customers are now demanding the agility and automation to easily scale their infrastructure, along with a more attractive consumption-based economic model.

Enlightened developers, cloud architects and their infrastructure counterparts view this inflection point as a clean-slate opportunity to deliver leap-frog advances to their organizations. They have witnessed web-scale giants like Amazon prove the very real benefits of a programmable, open, distributed, scale-out architecture. But, they also face some very real challenges in the transition from old infrastructure-centric deployment models to the modern world of application-driven infrastructure.

Challenges for Architects

The benefits of a programmable scale-out infrastructure are clear and much of the datacenter has evolved to support dynamic, application-specific provisioning for compute and network resources. However, the most critical infrastructure resource – storage – has not kept pace.

Every application has different storage requirements, yet today's storage solutions offer limited ability to optimize the infrastructure for the specific characteristics of each application. Optimize for performance and the infrastructure may be too expensive; optimize for cost and you may deliver unacceptable application performance.

What is a Data Services Platform?

A data services platform is a software-defined infrastructure for the storage and retrieval of data via a utility-like consumption model, offering different tiers and types of service that are dynamically tailored to meet the needs of each application or user.

Unlike traditional storage systems and all-flash arrays that are storage-centric, requiring an explicit understanding of their physical resources to use them, a data services platform is application-centric and fully abstracted from its physical resources, enabling the use of its services without any explicit knowledge about where its infrastructure is located or how it is configured.

This is analogous to plugging a household appliance into an outlet – the electricity flows automatically from the electrical grid, the appliance consumes only as much energy as it needs, and the cost of the electricity consumed varies with the day, time and amount of consumption. To use the electrical grid's resources, neither the appliance designer nor user must understand where or how its electricity is generated or transmitted to the outlet – the grid is virtually invisible.

Why is a Data Services Platform Needed?

Legacy applications, combined with the proliferation of cloud native applications, big data and real time analytics workloads, modern microservices and container deployments are driving the need for a much more agile data infrastructure.

A wide spectrum of price/performance requirements and the limited effectiveness of point storage solutions across today's diverse applications, data use cases and delivery environments are presenting a unique challenge and an opportunity to deliver a highly agile services-based solution – or more succinctly, a data services platform.

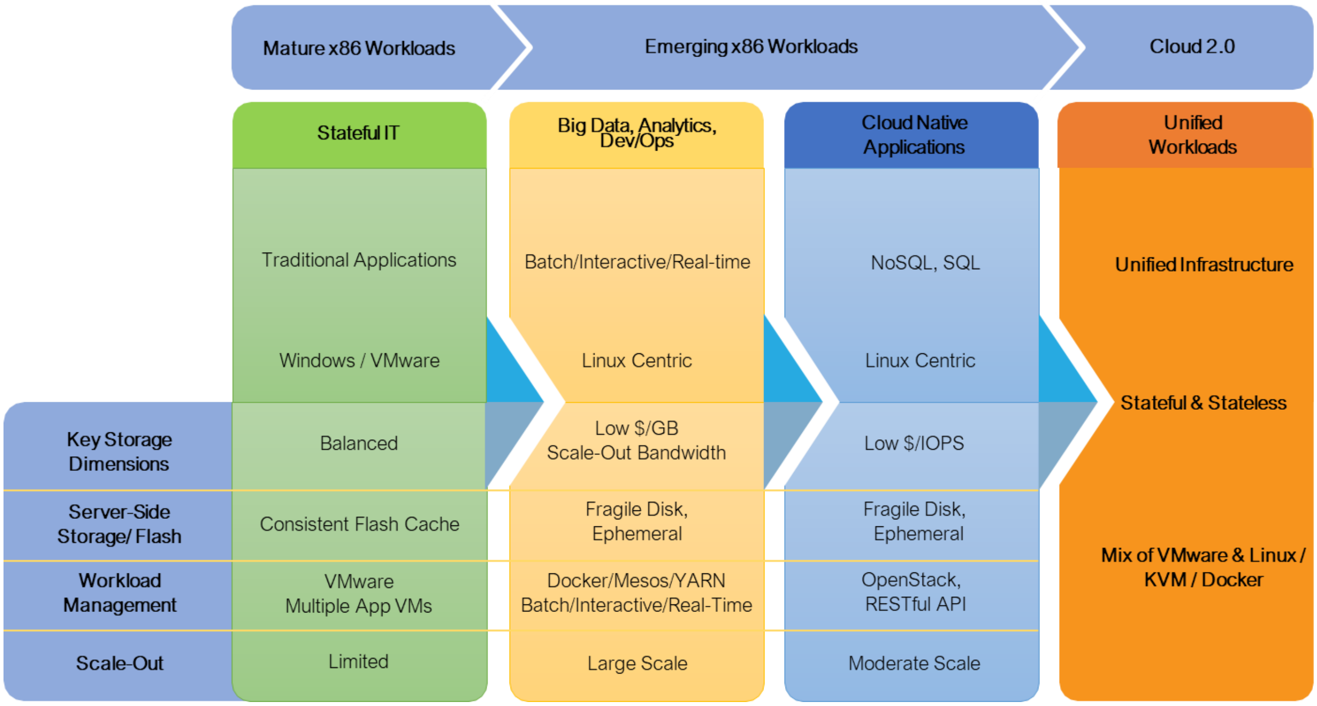

Figure 1: Infrastructure Unification

The raison d'être of a data services platform, or DSP, is to replace archaic, inelastic infrastructure-centric storage with modern, elastic application-centric data services that can support any application at scale within an enterprise IT or cloud environment, and deliver significantly better agility, simplicity and economics.

To deliver this functionality, a data services platform:

- Must be invisible: Applications and users must not need to be aware of the configuration or location of DSP resources to instantiate, provision or access its data services.

- Must be globally addressable: DSP data objects must be uniquely addressable within and across clouds, transparently supporting private, hybrid and/or public cloud deployments.

- Must be dynamically scalable: DSP resources must be scalable as needed without any service disruption, tooling, data migration or architectural changes.

- Must deliver consistent low latency: A DSP must deliver a consistent low-latency response for all workloads without the need for manual tuning.

- **Must be dynamically composable**: Application performance, data protection, security and/or other service level objectives (SLOs) must be able to be declared, with resources being dynamically composed or recomposed to meet them within the extent of hardware capability.

- **Must enable infrastructure-as-code**: A DSP must be completely programmable via RESTful APIs for provisioning, configuration and lifecycle operations management.

- **Must allow multi-tenancy and resource isolation**: A DSP must be able to manage multiple concurrent tenants and isolate its resources on a per-tenant, per-application and/or per-user basis, with the flexibility to deliver micro-segmentation within a tenancy.

- **Must be datacenter aware**: A DSP must understand datacenter layouts, enabling optimization for fault domains, performance and availability.

- **Must be SDN enabled:** A DSP must not be bound by physical switching or segmentation limitations and must automatically adapt to network changes without service disruption.

- **Must be software-defined**: DSP software must be fully abstracted from underlying hardware resources and provide data storage as a policy-driven set of services, sometimes called storage as-a-service.

At minimum, DSPs built upon these tenets should be able to:

- Deliver continuously available, ZeroOps data infrastructure at scale for any application
- Enable a high degree of agility and extensibility
- Automatically allocate and manage resources based on application-driven requirements
- Make applications agnostic to private, hybrid or public cloud deployment models
- Leverage industry-standard hardware and eliminate vendor lock-in
- Deliver better-than-public-cloud economics

The Datera Data Services Platform (DSP)

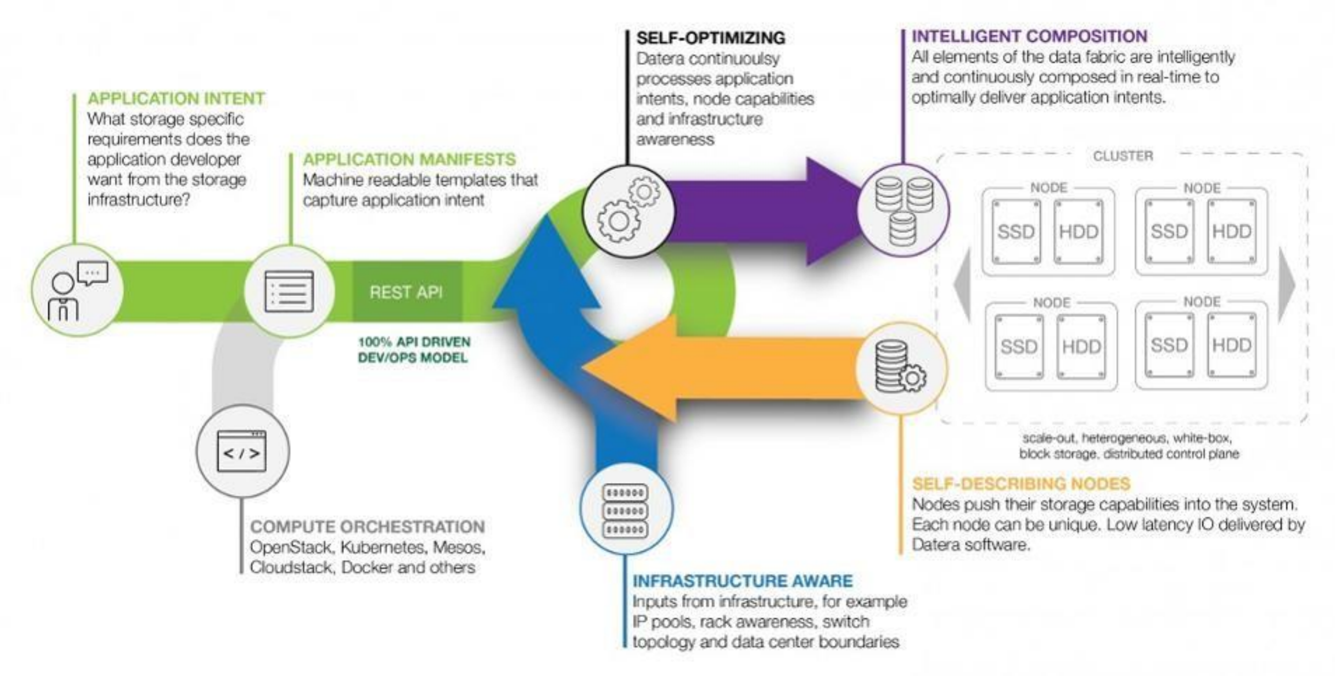

Figure 2: Self-Optimizing Data Fabric

The Datera Data Services Platform is a software-defined data infrastructure for virtualized environments, structured and non-structured databases, cloud stacks, DevOps and microservices deployments that require a set of critical data services, such as elastic block storage and object storage. Platform resources are accessed via infrastructure-as-code provisioning and consumed as-a-service.

DSP provides continuously available, operations-free, SLO-based data delivery and orchestration at scale for any application including monolithic, cloud-enabled or cloud-native designs. DSP may be used as a data infrastructure for enterprise storage, on or off-premises private or hybrid storage clouds, or as service-provider infrastructure to enable delivery of AWS-like data services in the public cloud.

DSP is deployed as a scale-out configuration of industry-standard x86 servers containing a heterogeneous mix of commodity storage media types including persistent memory, NVMe flash, SATA flash and/or conventional HDDs.

As a cluster of heterogeneous nodes with different price/performance characteristics, DSP provides a highly-elastic price/performance band that can automatically stretch to support the changing requirements of applications throughout their lifecycle.

DSP is inherently multi-tenant, with role-based tenancies, micro-segmentation and resource isolation that eliminates "noisy neighbor" contention.

DSP is both intelligent and autonomic, maintaining constant awareness of itself, the datacenter and the network, enabling it to be dynamically self-adaptive and self-optimizing when encountering changes in infrastructure, scale or workloads.

Architectural Objectives

Datera DSP architecture is based on a comprehensive set of design objectives to ensure the delivery of best-in-class levels of agility, extensibility, performance and economics.

At a high level, the architectural design objectives include:

- Automated management of data services based on individual application SLOs
- Multi-tenant support with resource isolation and per-tenant QoS controls
- Infrastructure-as-code provisioning and as-a-service resource consumption models
- Cloud-based monitoring and predictive analytics
- Data security including user authentication and at-rest data encryption
- Delivery of sub-100 microsecond read/write latencies
- Elastic performance, latency, capacity and cost
- Extensibility to quickly adopt next-generation technologies
- Resilience to withstand media, server, network, datacenter and/or electrical grid failures
- Deployment on industry-standard x86 hardware with commodity storage media

Capabilities Overview

• Distributed Scale-Out Design

DSP is based on a distributed scale-out, shared-nothing architecture that is instantiated as a cluster of three or more heterogeneous nodes. The architecture is designed to scale to more than 100 nodes.

A DSP node consists of software that runs on top of a bare metal industry-standard x86 server.

The architecture is agnostic as to server brand, allowing the potential for any customer-preferred brand (or brands) to be deployed within a cluster provided it meets minimum configuration requirements.

• Continuous Operation

DSP is inherently capable of continuous operation, delivering both high resilience and high availability through a combination of properties, including its scale-out design, live node insertion, rolling software updates and future-proof extensibility.

In a scale-out system, the degree of resilience[1] is primarily a function of the number of nodes in a cluster. Simply stated, the more nodes there are the greater the opportunity to survive a failure without impact to applications or users.

As a scale-out system, DSP is highly redundant and becomes increasingly resilient as it scales, while also improving its capacity to absorb failures and rapidly execute any recovery actions.

DSP hardware platforms offered or supported by Datera are generally configured to provide node-level resilience using redundant components such as multiple power supplies, fans, network interfaces, storage interfaces and boot media. However, custom configurations with fewer or no redundant components are supportable, providing lower cost options for larger clusters or more cost-sensitive environments.

Legacy service methodologies such as hot standby, hot sparing or 4-hour on-site parts replacement are not necessary or warranted for DSP. So long as there is sufficient capacity available to absorb a component or node failure, there will be no impact to the cluster. This allows spare parts to be obtained and remedial maintenance to be performed when it is most convenient and/or cost effective for the user.

This is the opposite of a conventional scale-up storage array, which becomes less resilient as the number of components increase and can become severely degraded when dividing its more limited set of resources between ongoing I/O operations and recovery actions.

- ## Data Protection and Availability

DSP employs data replication as its primary method of protection against data loss or loss of availability. The system supports one to five "replicas" (i.e. instances) of a volume per cluster. Each replica is a primary instance. All replicas may service I/O requests and are synchronized automatically.

The number of replicas per volume is selectable upon provisioning, with each replica being distributed in chunks (or spans) across nodes and storage media to provide resilience against component or node failure. Therefore, the number of nodes in the cluster must be equal to or greater than the number of replicas specified.

In addition to replication, DSP supports zero-copy snapshots and clones.

A zero-copy snapshot is an instant, space-efficient method of making a point-in-time copy of a volume or a group of related volumes (also called a consistency group.) Snapshots may be scheduled to support any desired interval, e.g. hourly, daily, weekly, etc.

DSP supports up to 256 snapshots per volume and 16,384 snapshots per system.

A clone is a full read/write copy of a volume, volume snapshot or consistency group snapshot. It is useful for development and test, analytics, or for creating new read/write copies of golden application libraries, databases, container data, etc.

- Stretched Clustering

Stretched clustering is a deployment model in which the nodes of a distributed system are geographically separated, and its data synchronously replicated between them to provide resilience against localized failures such as network or power outages at the rack or datacenter site level, or against more widespread internet or electrical grid failures. DSP can be configured to support disaster recovery/business continuity scenarios via stretched clustering.

A DSP stretched cluster is created by distributing its nodes across two sites, configuring a network bridge between the sites, and connecting a shared “witness” server to each site to monitor them and provide arbitration in the event of a failure or loss of connection between the sites.

A site may be an individual rack or a datacenter, with one site being designated as the “preferred” or primary site and the other serving as the secondary site. The witness periodically checks for a heartbeat at each site. If the witness doesn’t receive a response from a site or loses network connectivity to it, it will take remedial action based on the failure mode to provide continued data availability and assure proper recovery when the non-responsive site comes back online.

A DSP stretched cluster is intended to be operated within metropolitan distance, where the expected roundtrip latency between sites is less than 20 milliseconds. However, roundtrip latencies of up to 1000 milliseconds are capable of being supported, allowing operation over extended distances depending on the specific application and network latencies that can be tolerated before invoking their error recovery processes.

For example, a transaction processing application may generate one or more writes per transaction. The aggregate roundtrip latency of all writes, including any metadata such as record locks, may exceed the transaction timeout threshold of the application, resulting in application-initiated failover or error processing being invoked despite the DSP cluster remaining in a normal operating state.

Other applications that are less latency sensitive, such as data lakes, may tolerate greater latency and be suitable for use where the distance between sites is hundreds of miles or more.

• L2 and L3 Networking

DSP nodes may be interconnected via L2 network topology, or as a flat (CLOS) L3 network. The cluster interconnect protocol is independent of the physical connectivity (10GbE, 40GbE) and is designed to be either TCP/IP or RDMA based. This allows the cluster to support faster networks and multiple types of physical connectivity.

In an L2 network environment, separate networks are established for client access to data, internal control traffic between nodes, and management connectivity to a cloud-based analytics portal. Overlay networks including VLANs and VxLANs are supported.

In an L3 network environment, the access, internal and management networks are virtualized and share all available network connections, which are automatically managed via an internal software-based BGP router running on each node as a managed service, in a similar manner to Project Calico[2]. This provides flexibility in the configuration of the cluster's physical connectivity to optimize for any desired level of resilience, aggregation of bandwidth, port consumption or cost.

In L3 mode, the DSP nodes participate as discrete endpoints in the datacenter network. DSP supports iBGP or eBGP peering. In iBGP networks, DSP nodes participate as client peers and do not participate in mesh networking. DSP nodes use ECMP, while initiators may use ECMP, iSCSI multipath, bonding or no redundancy, based on the networking environment. Client multipathing is supported, with the caveat that ECMP may change the path distribution.

• IP Pools

IP pools enable fine-grained isolation of network address ranges on a per-volume or per-tenant basis. IP pools allow unique IP address ranges (L2 and L3 networks) and VLAN tags (L2 networks) to be used to segregate network traffic for access security or resource contention management purposes.

• Datacenter Awareness

In addition to resilience against failures within a cluster, DSP is architected to be datacenter-aware, inferring datacenter properties such as rack boundaries, availability zones and power domains and making it fully capable of supporting rack-scale design architectures.

• Elasticity[3]

DSP provides fine-grained, dynamic control of the performance, capacity and cost of data services based on individual tenant or application SLOs.

DSP architecture is designed to concurrently support different node types, such as hybrid-flash, all-flash or low-latency in mixed, heterogeneous fashion, unlike typical scale-out architectures that only support homogeneous nodes.

This enables the creation of an elastic price/performance band within a cluster, as well as the ability to manage the cost efficiency of data protection by using heterogeneous media types.

- Extensibility

DSP architecture is extensible by design, eliminating expensive and time-consuming tasks such as forklift upgrades and bulk data migrations.

As a software-defined architecture fully abstracted from hardware, additional DSP node types may be defined whenever needed or desired. This enables new technologies to be quickly integrated, unlike proprietary systems which are often slow to support new technologies.

Additionally, DSP architecture supports live insertion of nodes into operational clusters, allowing new technologies to be introduced at any time. When a node (of any type) is added, a live SLO-driven rebalancing of data and network resources is automatically performed. This rebalancing takes into consideration the properties of each node, including its server generation, performance and capacity.

Conversely, nodes may be decommissioned on demand, with automatic live migration (or draining) of data to other nodes, based on SLOs.

These extensibility features, in concert with DSP elasticity and resiliency, make it possible for a cluster to be expanded or contracted in an accordion-like fashion, providing both long-term future proofing and extended operating horizons for older servers and media.

- Native I/O Performance

DSP can deliver native I/O performance from commodity servers.

DSP code is directly executed on each node, eliminating the latency of virtualization layers and guest operating systems that can significantly reduce performance.

DSP further minimizes latency and maximizes throughput by separating data, control and management traffic into separate planes, resulting in more efficient use of underlying hardware.

The primary benefits of this are:

1. More elastic performance – greater software efficiency reduces hardware utilization and increases peak capacity, better absorbing spikes without visible impact to applications.

2. Better price/performance – greater software efficiency requires less hardware performance, allowing lower cost components to be used to satisfy performance SLOs.

Storage latency and throughput are also a significant component of application server efficiency. By delivering data much faster, DSP can improve application server utilization, potentially allowing greater virtual machine or container density, more peak capacity or the ability to support more concurrent users.

- 2D Auto-Tiering

As a heterogeneous scale-out architecture, the performance, capacity and cost of DSP nodes are independently configurable, enabling a tiered storage deployment within a cluster.

With DSP, any combination of hybrid flash, all-flash or low-latency node types may be mixed to create storage tiers with different characteristics. These tiers are self-managed within nodes (aka vertical tiering) and across nodes (aka horizontal tiering) to create a highly elastic price/performance band that dynamically adapts to changes in workloads or SLOs.

DSP exploits a thin layer of non-volatile memory as a cache in front of one or more types of storage media such as flash and/or disk, reducing latency and increasing throughput while optimizing the cost of capacity by storing data on less expensive media.

In horizontal tiering, data is intelligently distributed across nodes to deliver SLO-based performance at the best possible cost for each application.

To illustrate this concept, let's use an example of a three-node cluster consisting of two hybrid flash nodes and one all-flash node. An application requires a service level of 100k IOPS @ 500 µs average latency, which can be satisfied by any node type. The same application requires three copies (or replicas) of data for protection and availability. Depending on the application's media placement policies, DSP will automatically place one or more copies of the application's volumes across the flash media in each node, with the other copies being placed on flash media, if available, or on disk media. In this manner, the application performance SLO may be satisfied while optimizing the cost of capacity by selectively using lower cost media for replicas.

DSP also monitors and self-manages performance in real time. If an application performance SLO is changed or cannot be sustained, data will be live-migrated between nodes to ramp performance to the required level (to the extent possible based on cluster resources.)

Conversely, if an application's demand profile decreases over time, its data will automatically live-migrate to lower cost media to optimize price/performance (to the extent permitted by its SLOs.)

- Security

DSP supports both CHAP and Mutual CHAP protocols for authentication of iSCSI initiator access to iSCSI targets.

DSP provides selectable software-based data-at-rest encryption on a per-cluster basis. Data-at-rest encryption prevents unauthorized access to data in the event of the physical theft of storage media.

Data encryption may only be configured at initial cluster installation. Once selected, all user data and metadata are encrypted in perpetuity using the AES-XTS-256 standard. As an encrypted cluster scales, all nodes joining the cluster will also be encrypted.

Authentication passwords are used to generate encryption keys. All encryption keys are managed by DSP without user controls. DSP employs cryptographic erasure to prevent unauthorized access to data if a storage device is removed from a DSP node.

To assure complete flexibility of storage media selection, hardware-based device encryption is not supported.

Design Overview

The DSP architecture is based on a multi-planar design, consisting of a distributed control plane, high-performance data plane and an automated management plane.

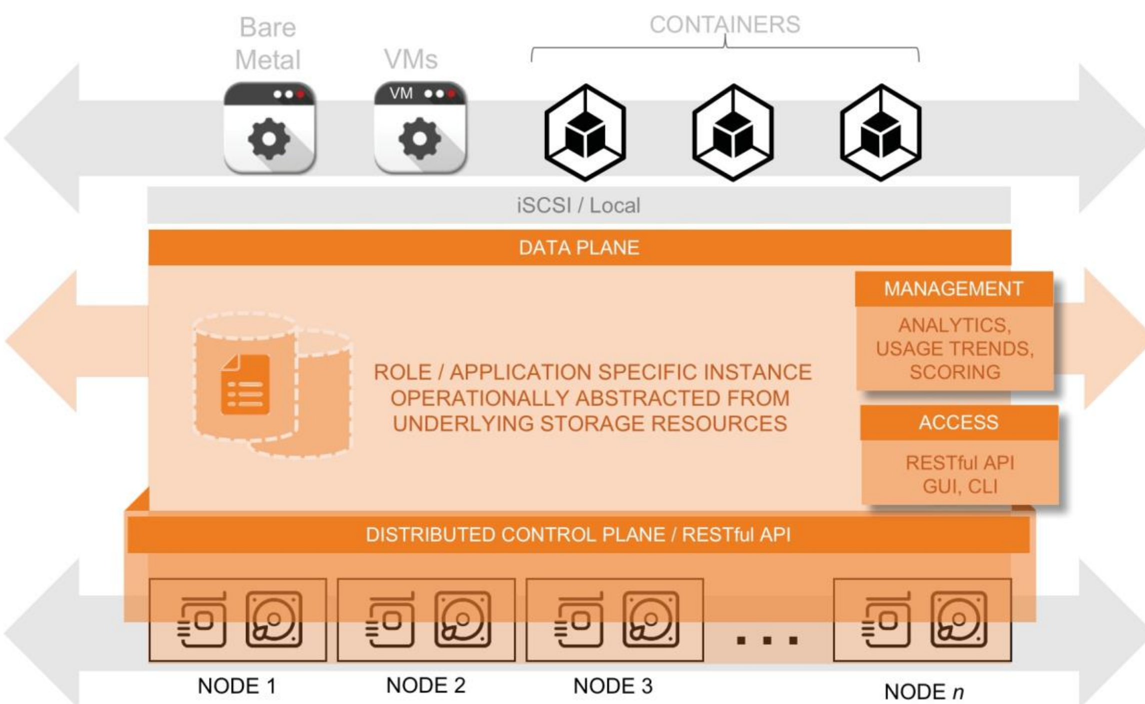

Figure 3: System Architecture

- ### Distributed Control Plane

DSP employs a symmetric scale-out control plane to build a coherent distributed system cluster from a collection of individual server nodes.

The control plane utilizes an internal network to communicate between nodes and eliminate any contention with client access and/or management traffic, which use separate physical or virtual networks depending on the networking environment.

The control plane discovers, configures, monitors and actively administers all nodes in the cluster and the components inside each node. Upon joining the cluster, each node self-describes its resources and capabilities. The control plane handles component or node failure and orchestrates recovery processes accordingly. This is central to the self-healing and self-optimizing capability of the DSP cluster.

The control plane manages several key aspects of the cluster:

Node Insertion/Deletion

The control plane enables the live insertion (i.e. non-disruptive addition) and deletion (i.e. non-disruptive removal) of nodes from an operating cluster. Nodes of any type may be inserted into or deleted from a cluster at any time, with rebalancing of data performed automatically within the bounds of application- or tenant-level SLOs.

This capability eliminates the need to plan or manually execute bulk data migrations and allows nodes featuring the latest technology to be incorporated as they become available, providing future-proof extensibility over an unlimited time horizon.

When inserting (or commissioning) a node, it will be automatically upgraded or downgraded to the same software version as the current nodes before coming online.

When deleting (or decommissioning) a node, the control plane will automatically replicate its data to other nodes to maintain compliance with all applicable policies. This makes real time decommissioning a simple process that can be performed without planning or disruption.

Policy Implementation

The control plane translates user-declared cluster-, tenant-, application- or volume-level SLOs into storage policies configured within the system that are enforced via corresponding micro services. Policies may be inherited or overridden, allowing volumes to have different policies within an application instance or tenancy.

For example, a SQL database application may consist of multiple volumes, with inherited policies governing a shared attribute such as a snapshot schedule, or override policies applied to create unique volume attributes such as performance or media placement. In this manner, entities such as log files, indexes, tables and temp space may share common attributes as well as possess unique attributes. Additionally, a template for the SQL database application may be created so that different instances of the database can be easily created and modified if needed only where they differ.

Configurable storage policies include:

Performance: Specifies the performance per volume, specified as IOPS and bandwidth.

If used, this determines the maximum performance of a volume. By default, there are no rate limitations on a volume. IOPS per volume may be set from 10 – 100,000. Bandwidth per volume may be set from 1000 KB/sec. – 2,400,000 KB/sec.

Media Placement: Specifies the type of media on which volumes and replicas are to be placed.

The options are:

- **Hybrid** – the volume and replicas may be placed on any node. If resources are available, the volume and at least one replica will be placed on an SSD, otherwise it will be placed on disk.
- **One Replica Flash** – the volume and replicas may be placed on any node. The volume and at least one replica will be placed on an SSD.
- **All Replica Flash** – the volume and replicas will be placed on any all-flash node. At least one all-flash node must be present to use this policy.

Replication: Specifies the number of replicas per volume (from 1 – 5.)

Three replicas are recommended as a minimum when using the Hybrid policy, assuring that multiple copies of data remain available if a replica becomes unavailable.

Two replicas are recommended as a minimum when using One Replica Flash or All Replica Flash policies.

A single replica may be used when data protection is provided by host or application-based replication or where data protection for the volume is not required.

Snapshot: If used, specifies the frequency and number of retained snapshots for a volume or consistency group. Note that snapshots may be taken on-demand independent of any policy settings.

Authentication: If used, specifies the protocol to authenticate initiator access to iSCSI targets. CHAP and Mutual CHAP are supported.

Data Compression: If used, specifies data compression on a per-cluster basis.

Encryption: If used, specifies at-rest encryption of storage media on a per-cluster basis.

Volume Provisioning

When a volume is provisioned, the control plane abstracts it into one or more spans – 16 gigabyte chunks of addressable data – which are subsequently placed across nodes based on storage policies. For each volume, the control plane builds a span map, determines span placement based on media placement policies, controls the extent of thin provisioning, and optimizes the access control placement.

This fine-grained resource handling inherently controls policy-defined QoS thresholds such as IOPS, bandwidth, media-type affinity and replication-driven data durability.

- ## High-Performance Data Plane

DSP employs a high-performance data plane to distribute I/O across nodes based on the SLOs and cluster-level policy codified by the control plane, further augmented by its self-awareness of individual node characteristics and media tiers.

The data plane encompasses the data export, data distribution, and data tiering layers.

Data Export Layer

The data export layer is based on the LinuxIO SCSI target (LIO™) – a widely used open source target-mode stack developed and maintained by Datera. This provides seamless interoperability with every block initiator.

DSP currently supports iSCSI and iSER as block interfaces, however, it is easily extensible to the wide spectrum of block protocols that LIO supports, including Fibre Channel, Fibre Channel over Ethernet, SRP, etc.

Data Distribution Layer

The data distribution layer consists of a software process called the DSS Proxy, which disburses data across nodes and heterogeneous storage media. The design incorporates a unique, lockless distributed-coherence protocol which enables enterprise-class performance with multiple access points to the storage.

The DSS Proxy continuously optimizes data placement based on changes in the underlying cluster, as well as any changes to the application environment.

Leveraging the inherent parallelism of the data distribution layer, data rebalancing and drive rebuilds can be performed in as little as 30 minutes, as compared to traditional arrays which may throttle rebalancing activity and can experience multi-day rebuild times for multi-terabyte drives.

Data Tiering Layer

The data tiering layer (DTL) employs a proprietary, low-latency log-structured data store to manage the physical I/O within each DSP node. The DTL has two primary functions:

1. Mapping of logical block addresses (used by volumes) to their corresponding physical block addresses on the storage media

2. Optimization of I/O via read and write caching, write coalescence and I/O serialization

The DTL physically consists of a non-volatile (NVDIMM) primary data cache backed by one or two layers of permanent storage media depending on the node type. The DTL is managed by a software process called the DSS Server.

When read requests are received from the data distribution layer, the DSS Server will first attempt to service the request from the primary data cache. In the event of a cache miss, the data is read from permanent storage media, with a copy placed in the primary cache and a prefetch algorithm applied to bring proximal data into cache. A least-recently used (LRU) algorithm is employed to keep "hot" data in the primary cache for low-latency service.

When write requests are received from the DSS Proxy, the DSS Server writes the data to the primary cache, providing memory-speed write latency. The data is then asynchronously de-staged to permanent storage media based on its media placement policy, with a copy remaining in primary cache subject to the LRU algorithm.

The DSS Server also optimizes writes to permanent media based on the media type. For SSDs, data is first coalesced (re-ordered into sequence to eliminate random writes) and then written in buckets sized to minimize read/erase/write cycles, optimizing the durability of the flash and reducing garbage collection. For HDDs, data is first coalesced and written in buckets sized to minimize head movement and maximize data transfer rate.

- ## Automated Management Plane

DSP employs an automated management plane to provide SLO-based data delivery that is accessed via infrastructure-as-code provisioning and consumed as-a-service.

This enables the replacement of archaic, inelastic, infrastructure-centric storage with modern, elastic application-centric data services that can support any application at scale within an enterprise IT or cloud environment, and deliver significantly better agility, simplicity and economics.

Application SLOs are codified as context-aware policies that dynamically drive the elasticity of the underlying data fabric (e.g. data layout, iSCSI/ iSER exports, etc.) based on tenancy, failure domains, and network topology.

The management plane delivers a self-optimizing consolidated infrastructure with robust workload isolation and cohesive QoS.

Application Provisioning

While the API allows for traditional storage provisioning, such as volumes and data services, it also enables a modern application-driven provisioning model. With application-driven provisioning, the workload or application view of storage and its associated services are modeled through an application template (AppTemplate.)

Some common applications, such as Cassandra, MySQL, Hadoop, VMware environments and test/development workloads, are pre-modeled through default templates. Users may also build custom templates from scratch or by cloning and modifying existing templates.

The AppTemplate is the interface used to express SLOs on a per-application basis, by configuring application-specific characteristics such as:

- Host access controls (initiators or hosts accessing the data)
- Host targets or exports
- Data volumes
- Performance
- Security
- Data durability requirements
- Data management services, such as snapshots, and their schedule

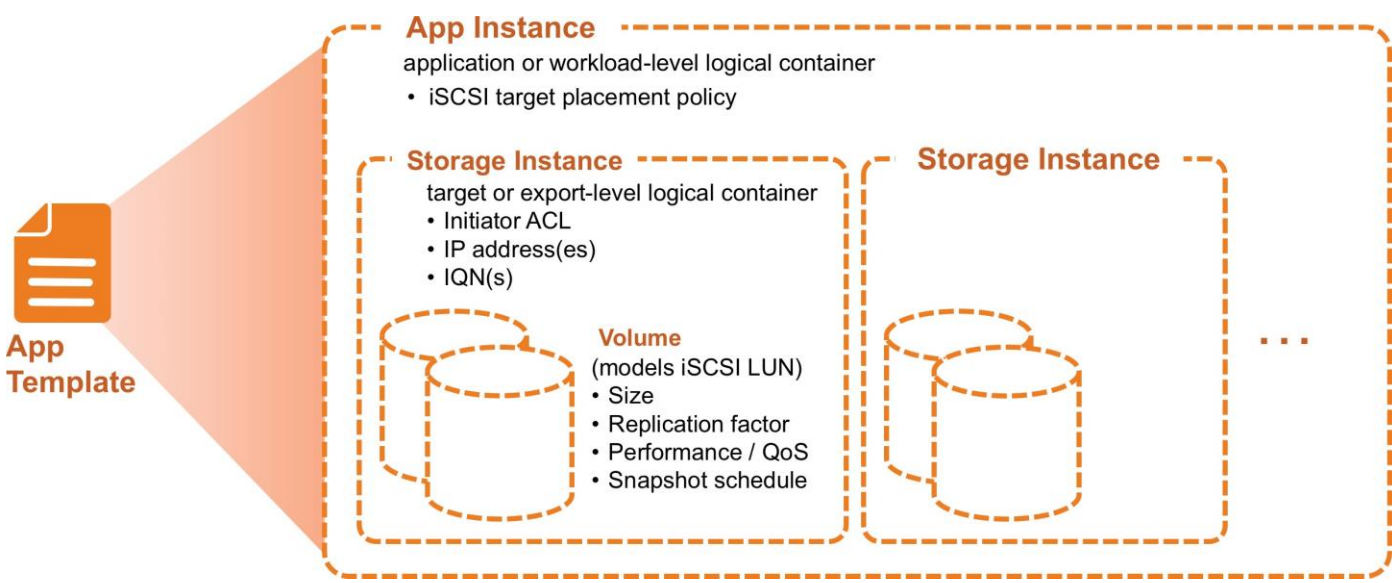

Figure 4: AppTemplate/AppInstance Layout

AppTemplates may be used to instantiate and deploy one or more application instances (AppInstance.) An AppInstance is the provisioned entity within the system.

An AppInstance may be associated with a parent AppTemplate, thereby inheriting any configuration/policy changes made to the AppTemplate, or it may be disassociated from the parent AppTemplate.

The application-specific characteristics referenced in the AppTemplate are known as policies. These policies are associated with the AppInstance at the time of instantiation, based on the location of the tenant or user issuing the deployment.

Policies are designed with the flexibility to be inherited or overridden through the template hierarchy based on the requirements. This provides the system with inherent multi-tenancy, policy association and configurability.

Policy Implementation and Multi-tenancy

Let's take an example using Hadoop, MySQL and MongoDB. Each of these applications have different SLOs and requirements. Their SLOs may be expressed in terms of performance, consistency model, replication expectations, data protection policy, and so on. An AppTemplate is used to describe the SLOs for each application.

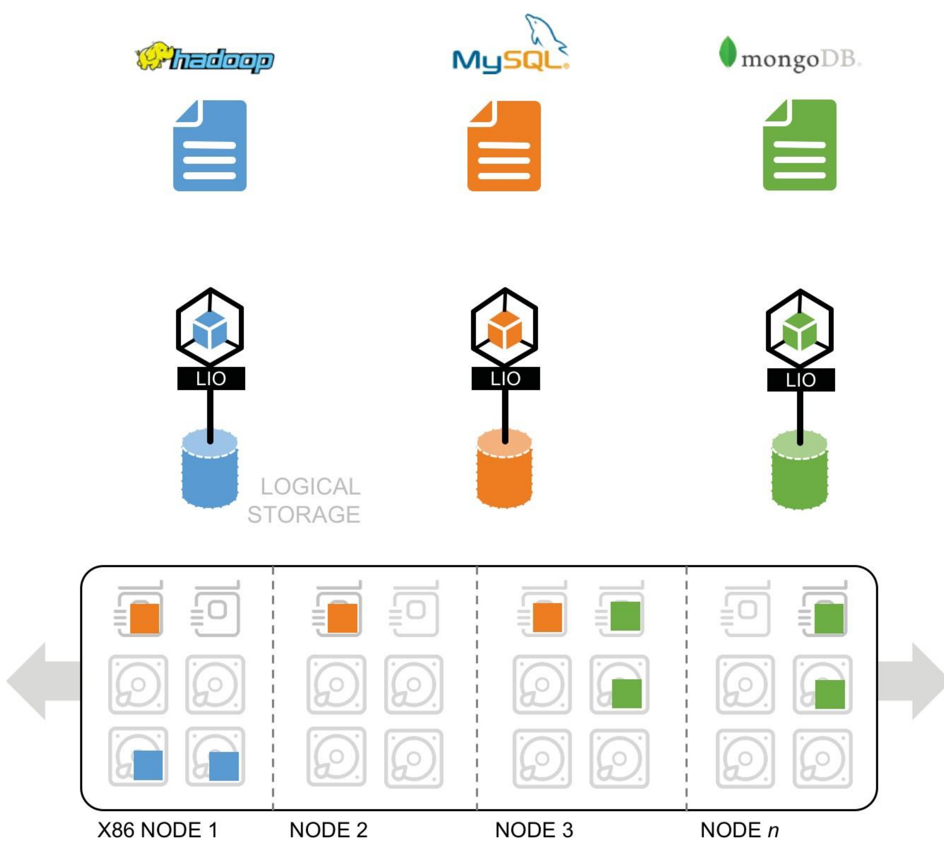

Figure 5: Deploying Multiple Applications based on Policies

Each of the polices expressed within an AppTemplate may be inherited or modified based on the tenant instantiating the AppTemplate. The tenant may be a user, an organization or a sub-organization.

For example, Development, Test and Production tenants might require different policies for the same applications. Therefore, based on the tenant instantiating the AppTemplate, different SLOs can be overlaid onto an application.

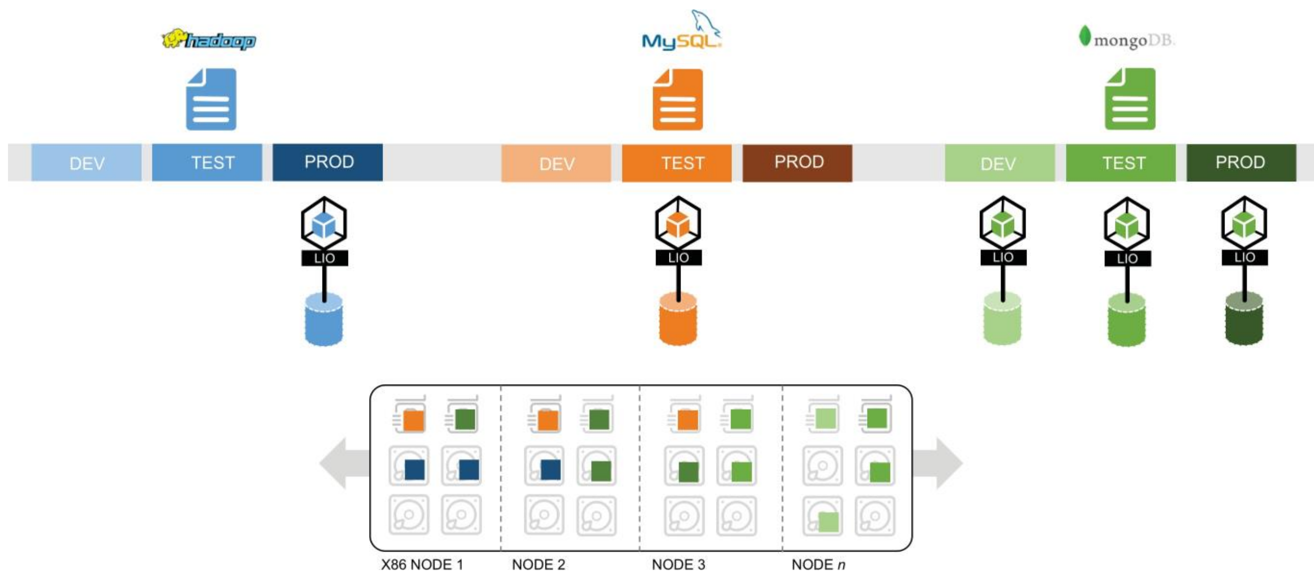

Figure 6: Overlaying Tenant Policies onto Applications

DSP also provides resource isolation and mapping of cluster resources to the appropriate AppInstance based on its tenancy, policy and associated AppTemplate. This allows for deep multi-tenancy controls across the cluster, beyond the visualization layer and access control lists for the data volumes.

Self-Service Provisioning

DSP replaces traditional, labor-intensive storage management and fragile operations automation with interactive self-service provisioning and operations-free agility.

DSP provides a policy-based REST API that makes the infrastructure fully programmable and dynamically composable, allowing instantiation of complete storage clouds with a single API call (infrastructure-as-code.)

The complete DSP scheme is query-able and binding interfaces auto-generated, making the platform consumable without any deep knowledge of the cluster layout.

The REST API can perform all resource provisioning, cluster configuration and management without any additional tools.

In addition, the management plane provides an API browser to provide an interactive experience.

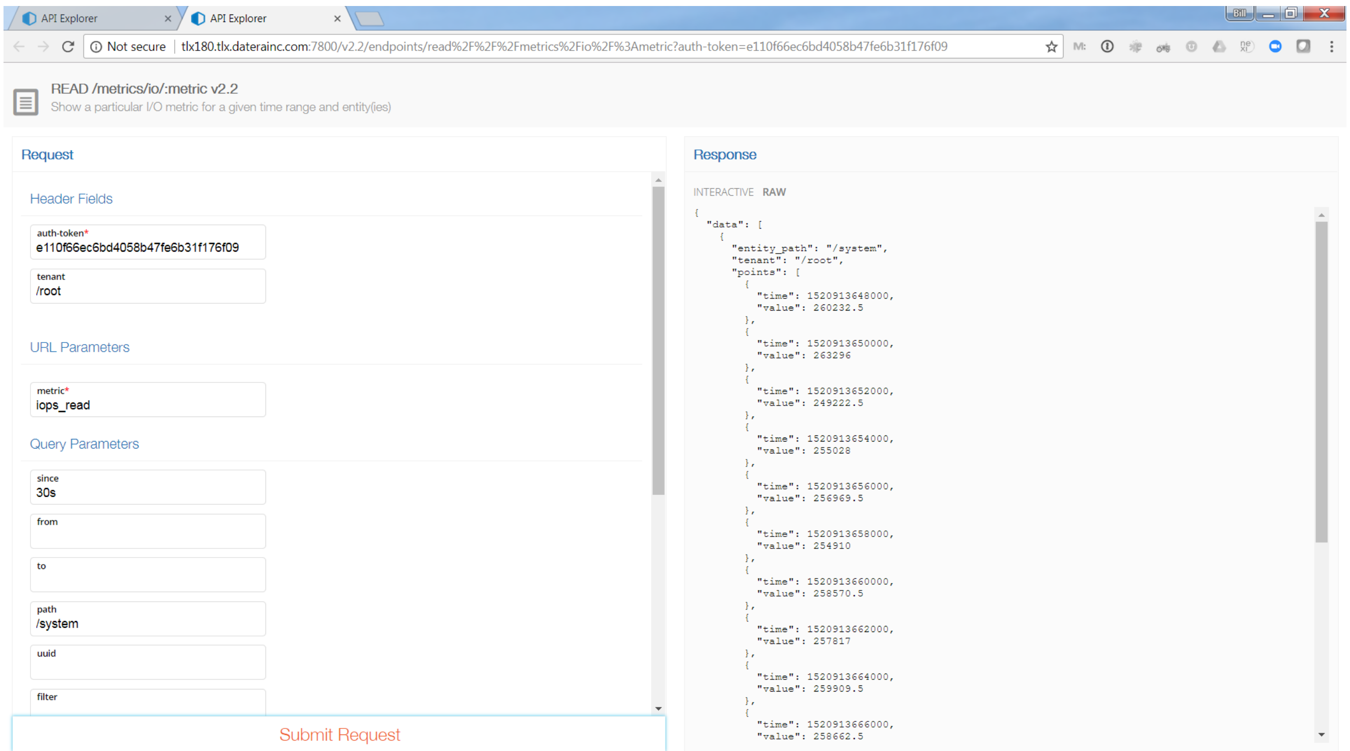

Figure 7: Datera API Browser

Conclusion

A wide spectrum of price/performance requirements and the limited effectiveness of point storage solutions across today's diverse applications and delivery environments are driving the need for a highly agile services-based solution – or more succinctly, a data services platform.

The Datera Data Services Platform addresses this need, delivering an architecture and solution specifically designed for the challenges of supporting traditional IT, hybrid IT and modern application development and delivery environments.

The Datera Data Services Platform is a software-defined data infrastructure for virtualized environments, databases, cloud stacks, DevOps, microservices and container deployments. It provides operations-free delivery and orchestration of data at scale for any application within a traditional datacenter, private cloud or hybrid cloud setting.

Datera combines the industry's only service level objective-based data services architecture, future-proof extensibility and game-changing price/performance to deliver operations-free agility, enterprise-class performance and latency, with better-than-public-cloud economics.

---

[1] Resilience is the ability of a system to avoid failure, while availability is the expression of system uptime as a percentage of total time.

[2] Project Calico is a new approach to virtual networking and network security for containers, VMs, and bare metal services, that provides a rich set of security enforcement capabilities running on top of a highly scalable and efficient virtual network fabric. Visit https://www.projectcalico.org/ for more information.

[3] Elasticity is the ability to expand or contract the services or capacity an IT infrastructure without detrimental effect to its reliability, resilience, availability, performance or other core attributes.