REFERENCE ARCHITECTURE

DEPLOY ENTERPRISE DC/OS WITH SCALEIO AND REX-RAY

DELL EMC MESOSPHERE

EXECUTIVE SUMMARY

Innovators such as Twitter, Uber, and Netflix are at the forefront of building data-rich, modern applications; delivering personalized, real-time experiences for their customers. Now companies of all shapes and sizes in industries including telecom, financial services, healthcare, retail, and many more need to respond or face risk of disruption.

However, building these types of applications is a complex and time consuming process. It also requires adopting a host of new technologies — including Docker containers, microservices, big data, and machine learning — while also integrating processes like continuous integration and delivery (CI/CD), site reliability engineering (SRE), and Agile development methods. With such drastic change required, how can you ensure your success?

The Joint reference architecture presented from Dell EMC & Mesosphere enables enterprises to build a unified public cloud like platform for powering stateless docker containers and stateful applications and data services.

Mesosphere DC/OS, powered by Apache Mesos, provides an easy to use platform to underpin your data-rich, microservices based applications, and allows application developers and operators to easily build, deploy, and elastically scale Docker containers and data services. DC/OS provides a public cloud like experience on any infrastructure you choose; with developer-friendly container orchestration, single-click deploy of big data services, and operations tools that help ensure security, performance, and scale.

Dell EMC serves a key role in providing the essential infrastructure for organizations to build their digital future, transform IT and protect their most important asset, information. Dell EMC ScaleIO, ScaleIO Ready Node

and REX-Ray enables enterprise customers’ IT and digital business transformation through trusted hybrid cloud and big-data solutions, built upon a modern data center infrastructure incorporates industry-leading converged infrastructure, servers, storage, and cybersecurity technologies.

The joint solution enables enterprises to

- Accelerate time to market with platform services (like those on AWS) that run anywhere
- Operate with confidence with government-grade security and resilient and automated operations
- Save money with agile and efficient hybrid infrastructure

THE BUSINESS CASE FOR MODERN APPLICATIONS

About a decade ago, web companies like Google, Facebook and Netflix addressed the challenge of serving millions of users in real time and processing unprecedented volumes of data. This started a quiet revolution in the datacenter, and launched the start of the mobile-cloud era — dramatically raising user expectations, creating new businesses, and placing competitive pressures on industries like retail, manufacturing, healthcare, and financials, among others.

Today, mainstream enterprises are looking for ways to better engage customers, improve operational decision-making, and capture new value streams. Doing this requires the enterprise to develop three related capabilities.

- Developer agility — rolling out new services or product enhancements quickly, and reducing time to value of new services.
- Data agility — gaining real time actionable insights from the large volume of data enterprises are collecting.
- Operational agility -- operating and managing the application and the underlying infrastructure on any cloud

The idea that enterprises need to get faster and smarter is not controversial. The real battlefront, and the question business leaders need to answer, is how to find the right technologies, operating models and talent that can collectively enable developer, data and operational agility.

USE CASES

Developer Agility

Developer agility isn't about getting developers to code faster. It's about getting code to production faster. Rolling out new services quickly means enterprises need to change how they build and operate software. Rolling out software faster increases customer satisfaction and retention, improves employee morale, and allows the business to quickly respond to competitive threats.

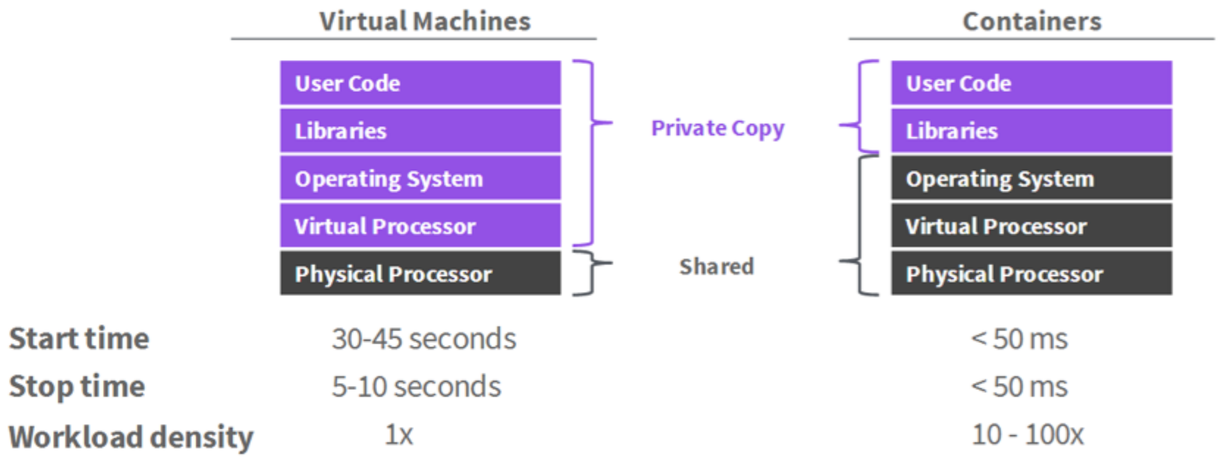
No longer can enterprises afford to build legacy-style, monolithic applications that run in virtual machines (VMs), Instead, the modern enterprise needs to adopt new ways of doing things (e.g., microservices in containers) and select a technology stack that eliminates the latency from concept to development to production, while managing risk.

In today's mobile-cloud era, software is evolving towards distributed systems and microservices (sometimes called "cloud native"). This approach provides service scalability and maintainability, and increases the speed that software can be deployed, especially when used in conjunction with continuous integration and continuous delivery models. Many think of cloud native apps as inherently stateless. However, modern enterprise apps cannot be entirely stateless since they need to rely on data — either to support a business function, or to engage customers in a meaningful way.

Docker containers

Containers are an emerging application development technology that provide standard way to package an application and all its dependencies in one file so that it can be moved between environments and run without changes. Containers provide efficiency, portability and reduce time to test application across different environments such as development, testing and production. The most popular open source container developer tool is Docker.

Containers are different than virtual machines. With containers, multiple applications can be deployed on the same operating system, which allows greater workload density and utilization. Containers are also faster to spin up (and kill), resulting better ability to scale up and down to meet quick spikes of demands. Containers can also offer better portability across platforms using different hypervisor and network technology.

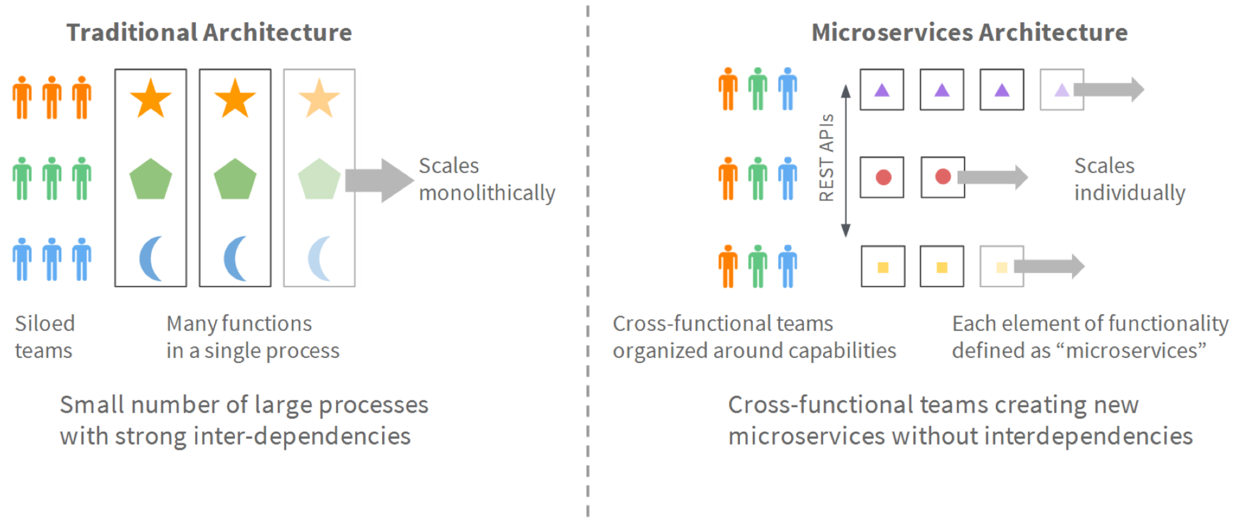


	Virtual Machines	Containers
Start time	30-45 seconds	< 50 ms
Stop time	5-10 seconds	< 50 ms
Workload density	1x	10 - 100x

Virtual Machines vs Containers

Microservices Architecture

The basic principle of microservices architecture is to break the application into a set of collaborating services, rolled out in the smallest deployable units. Each microservice implements a set of narrowly defined functions. Most commonly, these microservices interface with each other using standard protocols such as REST protocol. The benefits of microservices include the ability to have many teams working on different components in parallel, building and deploying independently, scaling only portions of an application that need the additional capacity, and being able to update/upgrade portions of the application without impacting users, as long as API contracts between microservices are maintained.


Traditional Architecture
Scales monolithically
Siloed teams
Many functions in a single process
Small number of large processes with strong inter-dependencies
Microservices Architecture
REST APIs
Scales individually
Cross-functional teams organized around capabilities
Each element of functionality defined as "microservices"
Cross-functional teams creating new microservices without interdependencies

Traditional vs Microservices Architectures

The real power of microservices architecture is in enabling small cross-functional teams to build and deploy functions independently using continuous integration and continuous deployment pipelines. These techniques enable enterprises to create and sustain autonomous and innovative teams that can build highly-scalable applications easily and new functionality quickly.

Continuous Integration/Continuous Delivery

Continuous Integration (CI) is a software development and DevOps practice where application is automatically built and tested after developers merge their individual code updates. Each merge/checkin is automatically tested. Continuous Integration allows application developers to quickly find and fix bugs, release software faster, and be more productive. Continuous integration is part of the modern software release process and involves technologies such as build servers (i.e Jenkins) and centralized code repositories (i.e git) and a software development process and culture.

Continuous delivery (CD) is another software development and DevOps practice that builds on Continuous integration and automatically deploys code changes to staging environment after the build and test stage. If the new application passes all the tests, application developers mark it ready for production, and either deploy it manually or have it automatically deployed to production through continuous deployment process.

Building and managing a CI/CD process is usually a large undertaking, especially in large organizations across multiple teams, yet a successful CI/CD process can dramatically accelerate the software development lifecycle and increase software quality.

DATA AGILITY

The way enterprises are using big data is shifting. The first wave of enterprise big data efforts essentially built large data warehouses (and tools) for use by analysts or data scientists. In what is now effectively the second wave, successful enterprises are building business applications powered by fast data and machine learning, capturing the value of actionable insights in real time.

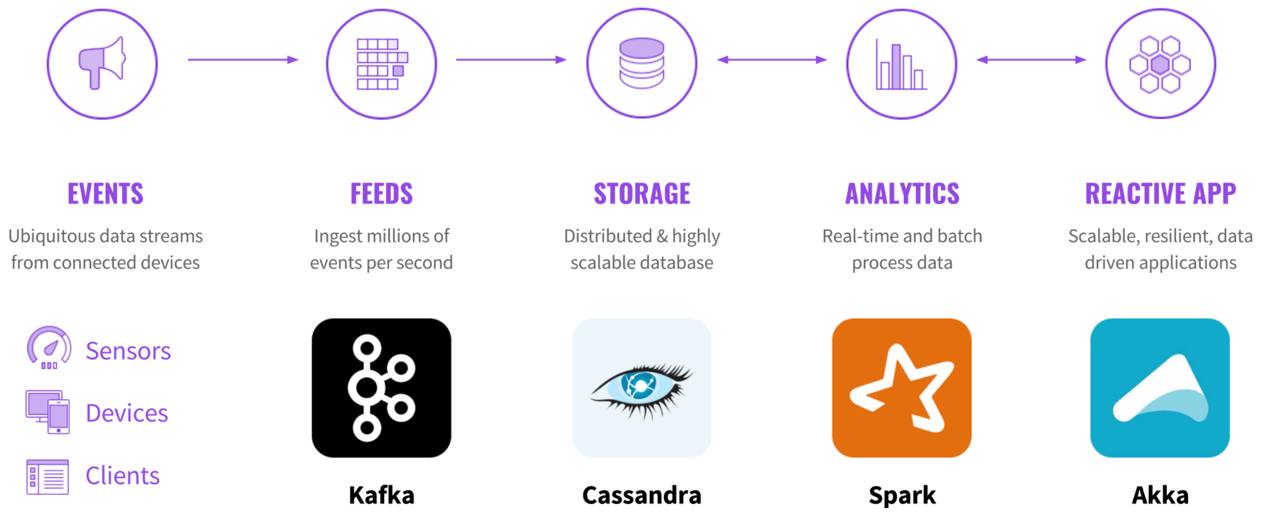
Successful enterprises leading the second wave of big data are shifting from building systems of record (e.g., “what were last quarter's sales of a product in a particular segment”) to systems of real-time insight and prediction (e.g., “what is an individual customer likely to buy, and what types of engagement can best influence their behavior?”). These businesses are deriving true value out of their big data efforts and for those still stuck in the first wave, competitive pressures are mounting.

Fast Data

Over the past two to three years, companies have started transitioning from *big* data, where analytics are processed after-the-fact in batch mode, to *fast* data, where data analysis is done in real-time to provide immediate insights. For example, in the past, retail stores such as Macy’s analyzed historical purchases by store to determine which products to add to stores in the next year. In comparison, Amazon drives personalized recommendations based on hundreds of individual characteristics about you, including what products you viewed in the last five minutes.

Another key shift in the architectural components of fast data systems is from the use of proprietary closed source systems to data pipelines stitched together from a variety of open source tools. In a recent survey, over 90% of respondents leveraged open source distributions for fast data applications, and almost 50% used open source exclusively. Open source tools are a force multiplier for developers getting started, and also can be used to avoid lock-in to proprietary solutions.

Today, most people think of Hadoop or NoSQL databases when they think of big data. Recently, several open source technologies have emerged to address the challenges of processing high-volume, real-time data, most prominently including Apache Kafka™ for data ingestion, Apache Spark™ for data analysis, Apache Cassandra™ for distributed storage, and Akka for building fast data applications.


EVENTS
Ubiquitous data streams from connected devices
FEEDS
Ingest millions of events per second
STORAGE
Distributed & highly scalable database
ANALYTICS
Real-time and batch process data
REACTIVE APP
Scalable, resilient, data driven applications
Sensors
Devices
Clients
Kafka
Cassandra
Spark
Akka

Fast Data Pipeline with Kafka, Cassandra, Spark, and Akka

Machine Learning

Machine learning is a specific type of Artificial intelligence that gives computers the ability to search through vast amount of structured and unstructured data, yet Instead of looking for known patterns, machine learning identifies new unknown patterns or predict future actions. Enterprises can then feed that into the appropriate application to act accordingly.

Some of the applications for machine learning include personalized advertising; autonomous vehicles; optimizing pricing, routing, and scheduling based on real-time data in travel and logistics; predicting personalized health outcomes; and optimizing merchandising strategy in retail.

VERTICAL	BIG DATA	FAST DATA
Automotive	Automakers analyze large sets of crash and car-based sensor data to improve safety features	Connected cars provide real-time traffic information and alerts for predictive maintenance.
Healthcare	Doctors provide care suggestions based on historical analysis of large datasets	Doctors provide insightful care recommendations based on predictive models and in-the-moment patient data.
Retail	Stores determine which products to stock based on analysis of previous quarter's purchase data.	Online retailers provide personalized recommendations based on hundreds of individual characteristics, including products you viewed in last five minutes.
Financial Services	Credit card companies create models for credit risk based on demographic data.	Credit card companies alert customers of potential fraud in real-time.
Manufacturing	Manufacturing plants improve efficiency based on throughput analysis.	Manufacturing plants detect product quality issues before they even occur.

Big Data Vs. Fast Data Examples

The truth is, Machine Learning landscape is fast moving, with new tools created or updated every day. Machine learning also requires vast amount of data processing for short burst of times, making it a great candidate for a hybrid infrastructure. Thus choosing the right platform not just increase the productivity of the hard-to-find talent such as data scientist and engineers, but also allows enterprises to quickly refine and update the models and continuously get the benefit from latest tools, technologies and best practices.

OPERATIONAL AGILITY

Infrastructure as a Service (IaaS) models essentially provide virtual machines on demand, with the virtual machine as unit of abstraction. While virtual machines have had tremendous impact in eliminating physical server management and improving traditional workload efficiency and manageability, the virtual machine-centric model is not suited for modern enterprise applications composed of distributed services.

Unfortunately, operational teams are stretched very thin trying to manage multiple silos of infrastructures, across on-premise and cloud infrastructure. Deploying, upgrading, running containers, microservices and data services is labor intensive and costly. More over, Large proportion of IT budget used to maintain expensive under-utilized legacy systems. and it is very arduous to introduce new technologies needed for innovation (e.g., Cassandra, Elastic Search).

While some organizations started to see public cloud as the solution, unfortunately moving to cloud has resulted in skyrocketing cost and eats the IT budget, In addition to having applications and operational tools being locked into a specific cloud provider.

One recent story highlights the risk of cloud lock-in, that of Snap (of Snapchat). In the S1 Registration Statement issued by Snap in February 2017, it came to light that Snap had handcuffed itself to Google Cloud. Of the annual loss of over $500 million, 80% was attributed to contractually obligated spend with Google. In the same filing, Snap states that they wrote their application to use some Google services “which do not have an alternative in the market.”

Operational agility means that enterprises should be able to

- Pool data services and apps in a shared infrastructure for higher utilization and lower cost.

- Dynamically and automatically recovers workloads during hardware/software failures, reducing operational overhead and reallocate FTE spend to strategic initiatives.
- Build apps once and have them run anywhere with no modification and no lock-in.
- Enable hybrid cloud and use the best cloud for the job - buy the valleys and rent the peak and maintain cloud spend predictability and control.
- Maintain security, control and compliance for modern apps.

Pooling resources to reduce cloud spend

By pooling data services, containerized applications and traditional applications to all run on the same infrastructure, utilization will increase, resulting a reduced cloud footprint. By pooling resources, enterprises managed to approach an average utilization rates of over 90%, and reduced hardware and/or cloud costs by over 60%. Operators can easily bring new services such as Kafka online, or add more instances of the same service to a cluster with already available resources, increasing efficiency. In addition, app teams can use the specific version of software they prefer, without the need to create separate siloes.

To increase utilization further, multiple users and teams can share the same cluster. Capabilities such as fine grained access control lists, secrets management, and integration with directory services (LDAP) and single sign-on solutions (SAML, OpenID connect) allow companies to isolate access to specific services based on a user’s role, group membership, or responsibilities.

Hybrid Cloud

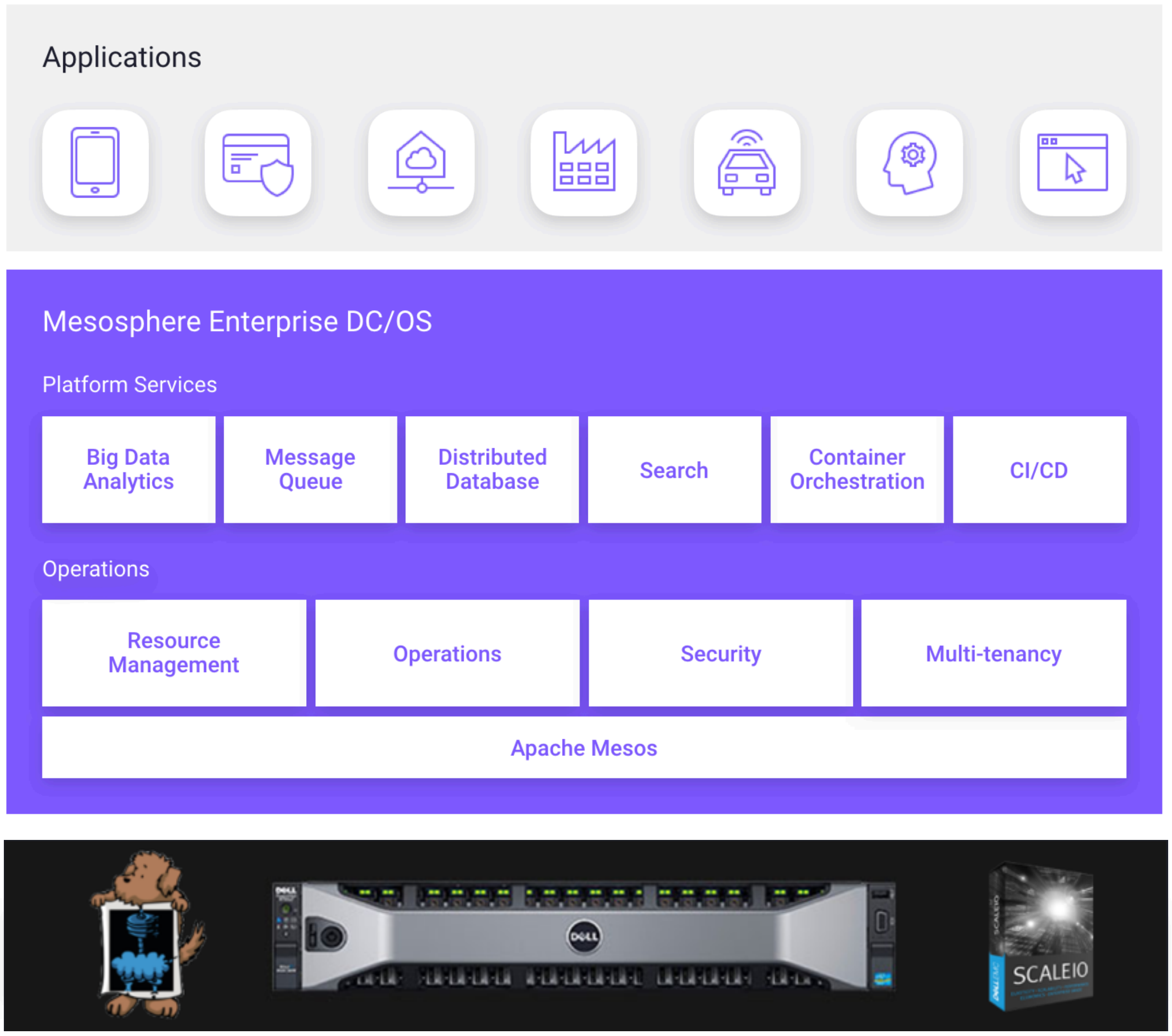
Many companies are shifting more workloads to the cloud or hybrid environments. The two key reasons for moving the cloud are:

1. To get immediate access to platform services such as analytics tools and databases. Cloud providers such as Amazon Web Services and Microsoft Azure package data services and make them easy to install and operate.
2. To enable scalability and elasticity, an imperative for bursting their workloads.

While public cloud provides clear advantages for many workloads, the two major downsides are rising costs and the risk of lock-in. Applications that are developed using public cloud platforms are tied to a specific cloud provider's APIs, and moving workloads after the fact is near impossible without rewriting them.

SOLUTION OVERVIEW

Dell EMC and Mesosphere have partnered to provide a complete solution for building and running modern data rich apps on your premise infrastructure. Dell EMC provides ScaleIO a software-defined storage and ScaleIO Ready Nodes for providing compute and storage infrastructure for your application. Mesosphere DC/OS runs on top of ScaleIO and ScaleIO Ready Nodes, provides the data center operating system that powers your modern data rich application



Mesosphere DC/OS: Cloud Native Platform Services on Any Infrastructure

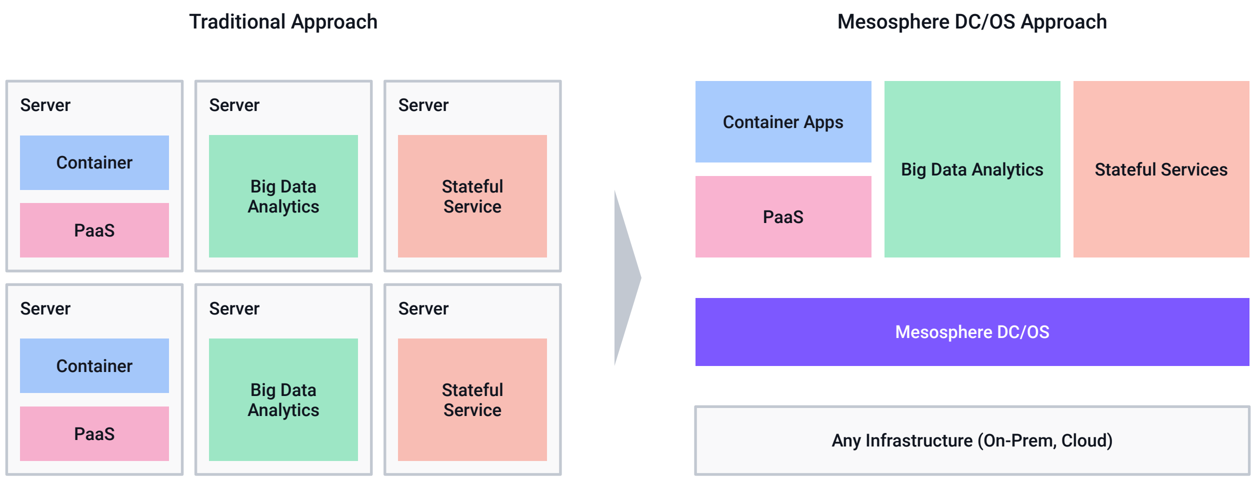
Introduction Mesosphere DC/OS

Faced with the challenge of serving billions of users in real time and rolling out new ideas quickly, Web scale companies (like Google, Twitter and Airbnb) realized that the traditional monolithic application stacks and the underlying virtual machine architecture could not meet the scalability and speed required to launch millions of work loads at a very short amount of time. They pioneered the use of modern applications, which leverages microservices architecture, containers and real time big data.

However, In order to get the benefit of modern applications, Web Scale companies needed to introduce a new paradigm of managing the underlying infrastructure, treating the the entire data-center as a single giant computer, and technologies such as Google Borg and Apache Mesos were the pillars of that paradigm.

THE DATACENTER OPERATING SYSTEM MODEL

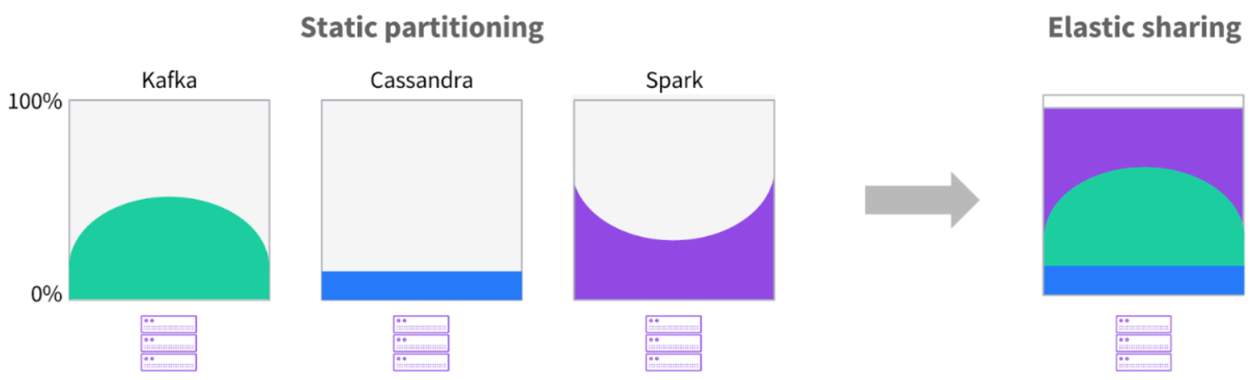
The Datacenter Operating System (DC/OS) model was conceived with the disruptive idea that running datacenter-scale services should not be more complex than using a single computer. DC/OS aggregates primitive infrastructure services so that both the developer and the operator work with a single form factor: the logical datacenter. In addition to providing elastic scalability for distributed systems, DC/OS ensures high availability and fault tolerance of services.



Elastic Data Infrastructure with Mesosphere DC/OS

The core of DC/OS is the Apache Mesos distributed systems kernel. Its power comes from the two-level scheduling that enables distributed systems to be pooled and share datacenter resources. Mesos provides the core primitives for distributed systems, such as resource allocation, isolation, and quota management.

Operators in a DC/OS model do not spend time managing individual machines (physical or virtual), dramatically reducing time and effort. A key benefit for operators is the ability to run at high levels of utilization, even as demand from multiple distributed services changes over time.



Elastic Resource Sharing Example

But the largest benefit of the DC/OS model is perhaps reducing the technical skills hurdle for running modern apps. With DC/OS, complex distributed systems like Spark, Kafka, Cassandra and many other services become dramatically easier to install and operate. Single commands in a DC/OS UI can launch datacenter-wide services, scale those services and maintain those services. DC/OS effectively applies prescriptive best practices in running these services, based on the production operational experience of others.

INTRODUCTION DELL EMC SCALEIO

Dell EMC ScaleIO® is a data center grade software-defined storage that uses standard x86 hardware and Ethernet to deliver scale-out block storage, simplifying storage lifecycle management. ScaleIO abstracts, pools and automates block storage in servers, including high performance All-Flash or hybrid media, and provides shared storage with enterprise class reliability.

ScaleIO customers of any size are able to deploy storage in minutes using standard x86 and Ethernet. Adminsitrators can add, remove or refresh nodes when desired with automated, in-place, data balancing and redistribution and without impacting applications. ScaleIO enables enterprise to operate at web-scale efficiency regardless to their size by abstracting, pooling and automating resources previously locked within servers and deliver enterprise storage at data center scale.

INTRODUCTION DELL EMC SCALEIO READY NODE

The Dell EMC ScaleIO® Ready Node is the validated, pre-configured building block for a software-defined block storage system. It brings together next-generation Dell EMC PowerEdge servers with ScaleIO. Combining best-of-breed server hardware and software, ScaleIO Ready Node configurations are built to unleash the power of ScaleIO and provide a fully validated and supported solution. Reducing the time IT organizations spend planning and deploying a new architecture, it is the perfect building block to achieve unprecedented scale, elasticity and performance with unmatched flexibility to grow and adapt to ever-changing business needs. SclaeIO Ready Node offers you the ability to redesign your storage environment and deploying software-defined storage using a storage-only, hyper- converged (storage and application running on the same server) or mixed approach.

Key Capabilitiies:

- Offers all-flash configurations optimized for capacity or density
- Tuned, optimized, validated, and supported by Dell EMC
- Supports a range of hypervisors and operating systems
- Scales from 3 nodes to more than 1,000
- Provides performance that scales linearly: 10 million+ IOPS

DELL EMC SCALEIO AND REX-RAY WITH DC/OS

Making persistent data available to containers is different than virtual machines. Containers use the root file system of a container to implement a copy-on-write process that efficiently stores application dependencies. Changes here have the purpose of updating the application itself but not storing persistent data. Therefore, containers don't have persistent storage. You can create, modify or delete files just like any operating system, but once the container is deleted or migrated to a different host, data residing only in the container will be inaccessible.

Container storage by default has limitations likely to limit its usefulness in production:

- **Lack of external storage support** - By default, containers store all the volumes on the local storage of the node, which can become a capacity and performance bottleneck. It is also vulnerable to data loss if the node fails.
- **Data persistency** - Container data is not globally persistent. If a container is moved from one physical host to another or if the node running container fails, the container data is not persistent.
- **Performance and Capacity** - Performance and capacity available for container storage is limited to the node's available disks
- **Availability** - Uptime of data is limited to physical or logical disks

It is also important to consider that some modern persistent applications that operate in a scale-out fashion may include their own distributed persistence layer. In this case it may be prescribed by the application as optimal to consume local storage for the application instance. The downside to this is that an instance or node failure may cause data rebuild events which can hurt operational performance of the application. ScaleIO can be used in this case to ensure these application instances can run from any node while keeping access to their data avoiding costly rebuilds.

SOLUTION COMPONENTS

Hardware

ScaleIO Ready Node

ScaleIO Ready Nodes provide flexibility for customers who want to build their own software-based block storage infrastructure with their choice of components.

Note: ScaleIO version 2.0.0.2 is the minimum required version for ScaleIO Ready Node.

For detailed configuration of ScaleIO Ready Nodes , please consult this specification sheet: Hardware Specification Sheet.

Setup and Configuration

Operating System	CentOS
ScaleUI	Version 2.0.1
Docker	Version 1.12
Rex-Ray	Version 0.3.3

ScaleIO

The ScaleIO system consists of following software components:

Meta Data Manager - MDM

Configures and monitors the ScaleIO system. The MDM can be configured in redundant Cluster Mode, with three members on three servers.

ScaleIO Data Server - SDS

Manages the capacity of a single Ready Node server and acts as a back-end for data access. The SDS is installed on all servers contributing local disks to the ScaleIO distributed storage system. Combined these devices logically create storage pools and fault domains and are accessed through the SDS.

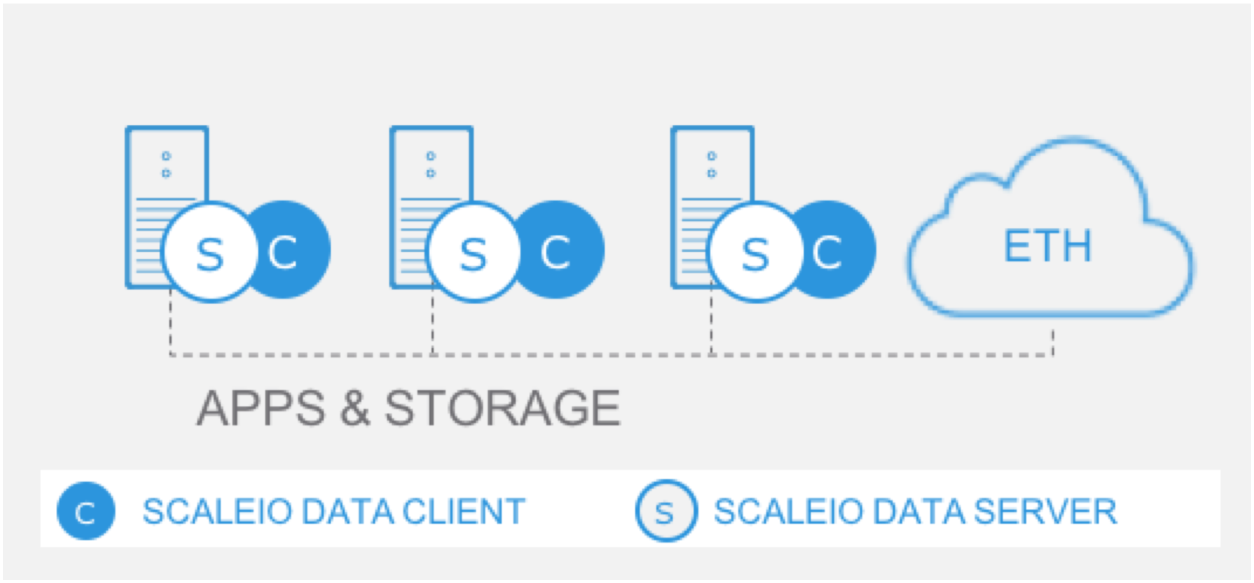
ScaleIO Data Client - SDC

A lightweight kernel device driver that exposes ScaleIO volumes as block devices to the application that resides on the same server on which the SDC is installed.

SCALEIO HYPER CONVERGED DEPLOYMENT

ScaleIO is software and can be deployed in any manner that best meets the workloads. A typical deployment is where you have dedicated SDS and MDM nodes that act solely as a storage platform. Alternatively in hyper-converged (co-resident) deployments the ScaleIO Data Clients (SDCs) and ScaleIO Data Servers (SDSs) run on the same node. This maximizes hardware utilization and reduces infrastructure requirements.

In hyper-converged environments where the storage servers and storage clients run on the same physical nodes, the resource utilization efficiency of the overall stack (storage, CPU and memory) is extremely high. This recommended approach works very well in environments where a single group manages both storage and compute.



Network Infrastructure for ScaleIO Nodes

While Kafka is the most popular message broker, other popular tools include Apache Flume™ and RabbitMQ. Apache Flume is a distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of data. RabbitMQ, backed by Pivotal, is a popular open source message broker that gives applications a common platform to send and receive messages. RabbitMQ is preferred for use-cases requiring support for Advanced Message Queuing Protocol (AMQP).

Network Infrastructure for ScaleIO Nodes

Leaf-Spine architecture (Recommended) :

- ScaleIO can scale out to 1024 nodes.
- It facilitates scale-out deployments.
- A leaf-spine topology allows the use of all network links concurrently.
- More predictable latency due to the elimination of uplink oversubscription.
- For networking consideration, please refer the ScaleIO networking best practices guide.

REX-Ray

REX-Ray is the leading container orchestration engine enabling persistence for cloud native workloads.

REX-Ray allows you to run any application in a container including databases, key-value stores, big data, real-time streaming, messaging, and any application storing data. REX-Ray is built on the libStorage library which ensures reusable and trusted interoperability that integrates cloud native and storage platforms.

REX-Ray uses a distributed client-server model. The client abstracts local host processes (requests for volume attachment, discovery, format, and mounting of devices) while the server provides the necessary abstraction of the control plane for multiple storage platforms. Irrespective of platform, REX-Ray provides the following common functionality

Multiple architectural choices allow flexibility for deployments. REX-Ray can be configured in a standalone or agent/controller model. The agent abstracts the host specific orchestration tasks (volume mount, discovery, format) from the controller specific lifecycle operations (create, remove).

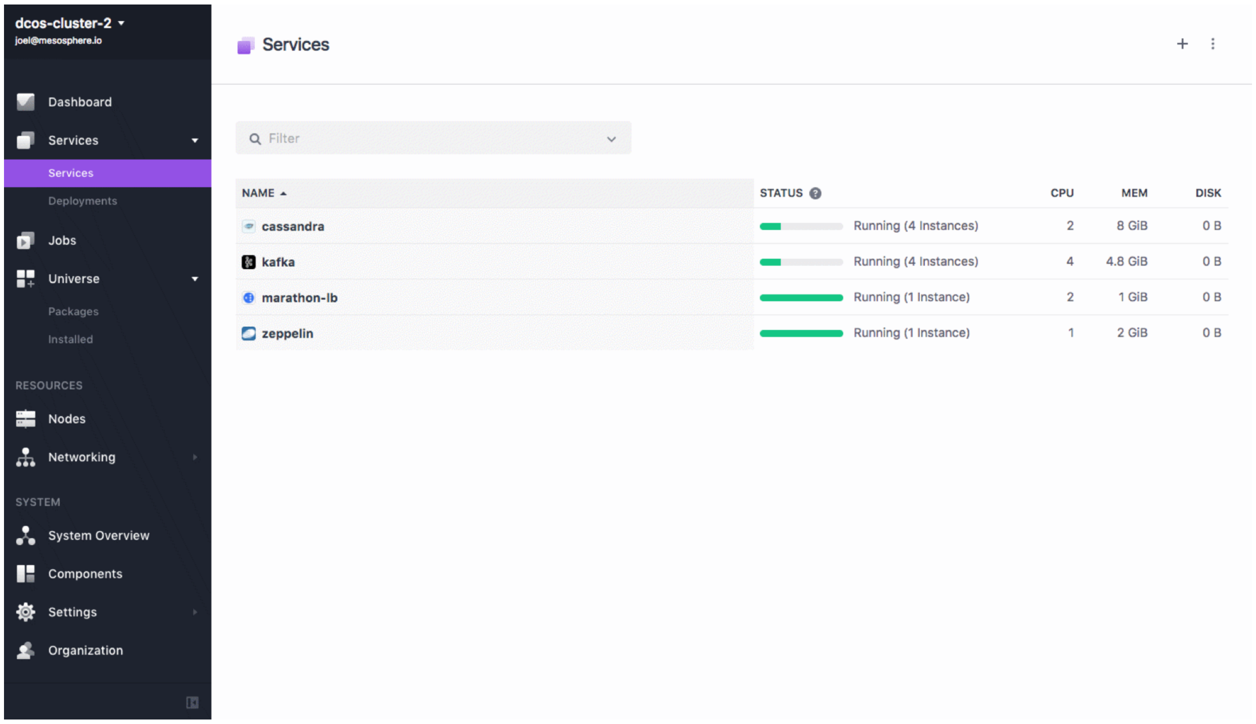
REX-Ray provides homogenous support across many storage platforms include Amazon AWS/EFS/EBS, Ceph RBD, Dell EMC ScaleIO/Isilon, Digital Ocean Block Storage, Fitted Cloud EBS Optimizer, GCE Persistent Disk, Azure Unmanaged Disk, and VirtualBox Virtual Media. In addition to running as a managed plugin in a container, it is tested with CentOS, CoreOS, RHEL, and Ubuntu. DC/OS is supported on CentOS, CoreOS and RHEL. For Mesos, it can provide persistent storage for any task.

For further details, please follow the {code} by Dell EMC REX-Ray project.

Once you have installed REX-Ray, it requires a configuration file for storing details used to communicate with storage providers. This can include authentication credentials and driver-specific configuration options. Refer to the libStorage Storage Providers documentation for sample configurations of all supported storage platforms. We will focus on the configuration file for ScaleIO.

DEMONSTRATION

DC/OS is a platform for running distributed containerized software, like apps, jobs, and services. As a platform, DC/OS is distinct from and agnostic to the infrastructure layer. This means that the infrastructure may consist of virtual or physical hardware as long as it provides compute, storage, and networking.



NAME	STATUS	CPU	MEM	DISK
cassandra	Running (4 Instances)	2	8 GiB	0 B
kafka	Running (4 Instances)	4	4.8 GiB	0 B
marathon-lb	Running (1 Instance)	2	1 GiB	0 B
zeppelin	Running (1 Instance)	1	2 GiB	0 B

Create an Application with External Volumes

Create a Portable Persistent Application with a Marathon App Definition

When launching applications with ScaleIO external volumes it ensures the data for the application is stored separately from the application's container. This enables any node under DC/OS to be able to run the persistent application with its data.

You can specify an external volume in your Marathon app definition. Learn more about Marathon application definitions.

```
{
  "id": "hello",
  "instances": 1,
  "cpus": 0.1,
  "mem": 32,
  "cmd": "/usr/bin/tail -f /dev/null",
  "container": {
    "type": "MESOS",
    "volumes": [
      {
        "containerPath": "test-rexray-volume",
        "external": {
          "size": 100,
          "name": "my-test-vol",
          "provider": "dvdi",
          "options": { "dvdi/driver": "rexray" }
          },
        "mode": "RW"
      }
    ]
  },
  "upgradeStrategy": {
    "minimumHealthCapacity": 0,
    "maximumOverCapacity": 0
  }
}
```

Using a Mesos Container

In the app definition above:

- `containerPath` specifies where the volume is mounted inside the container. For Mesos external volumes, this must be a single-level path relative to the container; it cannot contain a forward slash (`/`). For more information, see the REX-Ray documentation on data directories.
- `name` is the name that your volume driver uses to look up your volume. When your task is staged on an agent, the volume driver queries the storage service for a volume with this name. If one does not exist, it is created implicitly. Otherwise, the existing volume is reused.
- The `external.options["dvdi/driver"]` option specifies which Docker volume driver to use for storage. The only Docker volume driver provided with DC/OS is `rexray`. Learn more about REX-Ray.
- You can specify additional options with `container.volumes[x].external.options[optionName].` The dvdi provider for Mesos containers uses `dvdcli`, which offers the options documented here. The availability of any option depends on your volume driver.
- Create multiple volumes by adding additional items in the `container.volumes` array.
- Volume parameters cannot be changed after you create the application.

Important: Marathon will not launch apps with external volumes if `upgradeStrategy.minimumHealthCapacity` is greater than 0.5, or if `upgradeStrategy.maximumOverCapacity` does not equal 0.

Using a Docker Container

```
{
  "id": "/test-docker",
  "instances": 1,
  "cpus": 0.1,
  "mem": 32,
  "cmd": "/usr/bin/tail -f /dev/null",
  "container": {
    "type": "DOCKER",
    "docker": {
      "image": "alpine:3.1",
      "network": "HOST",
      "forcePullImage": true
    },
    "volumes": [
      {
        "containerPath": "/data/test-rexray-volume",
        "external": {
          "name": "my-test-vol",
          "provider": "dvdi",
          "options": { "dvdi/driver": "rexray" }
        },
        "mode": "RW"
      }
    ]
  },
  "upgradeStrategy": {
    "minimumHealthCapacity": 0,
    "maximumOverCapacity": 0
  }
}
```

The `containerPath` must be absolute for Docker containers.

Important: Refer to the REX-Ray documentation to learn which versions of Docker are compatible with the REX-Ray volume driver.

SUMMARY

This combined reference architecture describes a solution for building and running modern applications that use containers and big data services within your on-premise infrastructure. In this solution, Dell EMC provides ScaleIO, ScaleIO Ready Nodes and Rex-Ray, a complete scale-out solution for providing compute and storage infrastructure for your application and supporting technology. Mesosphere's "Datacenter Operating System" (DC/OS) with ScaleIO and ScaleIO Ready Node to provide a platform that aggregates both compute and storage (hyper converged) and provides a simple UI or CLI for application and container management and orchestration, single-click deployment of 100's of data and devops services, and elastic scaling. Customers who follow the reference architecture will be using a proven solution that minimizes risk and simplifies deployment, and are supported by both Mesosphere and Dell EMC.

RESOURCES AND ADDITIONAL LINKS

ScaleIO Download Page: http://www.emc.com/products-solutions/trial-software-download/scaleio.htm ScaleIO User

ScaleIO Installation Guide:

https://www.emc.com/collateral/technicaldocument/emc-scaleio-installation-guide.pdf

ScaleIO Design Considerations and Best Practices: https://www.emc.com/collateral/white-papers/h15148-emc-scaleio-deployment-guide.pdf

ScaleIO Networking Best Practices: https://www.emc.com/collateral/white-papers/h14708-scaleio-networking-best-practices.pdf

Mesosphere Enterprise DC/OS

https://mesosphere.com/product/