

VersaStack for IBM Cloud Private with Cisco UCS and IBM Storage

Design and Deployment Guide of VersaStack for IBM Cloud Private with Cisco UCS and IBM Storage

Last Updated: June 8, 2018



About the Cisco Validated Design Program

The Cisco Validated Design (CVD) program consists of systems and solutions designed, tested, and documented to facilitate faster, more reliable, and more predictable customer deployments. For more information, visit:

<http://www.cisco.com/go/designzone>.

ALL DESIGNS, SPECIFICATIONS, STATEMENTS, INFORMATION, AND RECOMMENDATIONS (COLLECTIVELY, "DESIGNS") IN THIS MANUAL ARE PRESENTED "AS IS," WITH ALL FAULTS. CISCO AND ITS SUPPLIERS DISCLAIM ALL WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE. IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THE DESIGNS, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

THE DESIGNS ARE SUBJECT TO CHANGE WITHOUT NOTICE. USERS ARE SOLELY RESPONSIBLE FOR THEIR APPLICATION OF THE DESIGNS. THE DESIGNS DO NOT CONSTITUTE THE TECHNICAL OR OTHER PROFESSIONAL ADVICE OF CISCO, ITS SUPPLIERS OR PARTNERS. USERS SHOULD CONSULT THEIR OWN TECHNICAL ADVISORS BEFORE IMPLEMENTING THE DESIGNS. RESULTS MAY VARY DEPENDING ON FACTORS NOT TESTED BY CISCO.

CCDE, CCENT, Cisco Eos, Cisco Lumin, Cisco Nexus, Cisco StadiumVision, Cisco TelePresence, Cisco WebEx, the Cisco logo, DCE, and Welcome to the Human Network are trademarks; Changing the Way We Work, Live, Play, and Learn and Cisco Store are service marks; and Access Registrar, Aironet, AsyncOS, Bringing the Meeting To You, Catalyst, CCDA, CCDP, CCIE, CCIP, CCNA, CCNP, CCSP, CCVP, Cisco, the Cisco Certified Internetwork Expert logo, Cisco IOS, Cisco Press, Cisco Systems, Cisco Systems Capital, the Cisco Systems logo, Cisco Unified Computing System (Cisco UCS), Cisco UCS B-Series Blade Servers, Cisco UCS C-Series Rack Servers, Cisco UCS S-Series Storage Servers, Cisco UCS Manager, Cisco UCS Management Software, Cisco Unified Fabric, Cisco Application Centric Infrastructure, Cisco Nexus 9000 Series, Cisco Nexus 7000 Series, Cisco Prime Data Center Network Manager, Cisco NX-OS Software, Cisco MDS Series, Cisco Unity, Collaboration Without Limitation, EtherFast, EtherSwitch, Event Center, Fast Step, Follow Me Browsing, FormShare, GigaDrive, HomeLink, Internet Quotient, IOS, iPhone, iQuick Study, LightStream, Linksys, MediaTone, MeetingPlace, MeetingPlace Chime Sound, MGX, Networkers, Networking Academy, Network Registrar, PCNow, PIX, PowerPanels, ProConnect, ScriptShare, SenderBase, SMARTnet, Spectrum Expert, StackWise, The Fastest Way to Increase Your Internet Quotient, TransPath, WebEx, and the WebEx logo are registered trademarks of Cisco Systems, Inc. and/or its affiliates in the United States and certain other countries.

All other trademarks mentioned in this document or website are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (o8ogR)

© 2018 Cisco Systems, Inc. All rights reserved.

Table of Contents

Executive Summary.....	6
Challenge.....	6
Solution.....	6
Solution Overview.....	7
Introduction.....	7
Audience.....	7
What's New?.....	7
Solution Summary.....	8
VersaStack Program Benefits.....	9
Solution Benefits.....	9
Technology Overview.....	10
VersaStack Converged Infrastructure.....	10
Option1: VersaStack with Cisco UCS M5 and IBM SVC (Validated).....	10
Option2: VersaStack with Cisco ACI and IBM SVC (Optional).....	11
Option3: VersaStack with Cisco UCS Mini and IBM Storwize V5000 iSCSI Storage (Optional).....	11
VersaStack Converged Infrastructure Components.....	11
Cisco Unified Computing System.....	12
Cisco Nexus 93180YC-EX.....	13
Cisco MDS 9396S Fabric Switch.....	14
IBM SAN Volume Controller.....	14
VMware vCenter Server.....	14
VersaStack for IBM Cloud Private Add-on Components.....	15
IBM Cloud Private 2.1.0.....	15
IBM Spectrum Connect (version 3.4.0).....	16
IBM Storage Enabler for Containers.....	16
Solution Design.....	18
VersaStack Architecture for ICP.....	19
VersaStack iSCSI Design.....	20
Cisco UCS LAN Connectivity.....	20
IBM Storage Systems.....	21
IBM SAN Volume Controller – I/O Groups.....	21
IBM SAN Volume Controller – iSCSI Connectivity.....	21
Cisco UCS Server Configuration for Vsphere.....	22
IBM Cloud Private Architecture.....	23
Boot Node.....	23
Master Node.....	23

Worker Node	23
Proxy Node.....	23
Management Node.....	24
IBM Cloud Private Configuration on VersaStack	25
Virtual Switching Architecture	26
iSCSI VLAN Configuration for ICP Worker Nodes	27
VersaStack for IBM Cloud Private Control and Data Paths	28
Deployment Hardware and Software.....	30
VersaStack Storage Configuration	30
Storage Pools Configuration	30
Create a New VMware Port Group for ICP Environment	31
IBM Spectrum Connect 3.4.0 Installation.....	33
Install Virtual Machine for IBM Spectrum Connect.....	33
IBM Spectrum Connect Configuration	37
Adding VersaStack Storage System to IBM Spectrum Connect 3.4.0	39
ICP Installation.....	46
IBM Cloud Private Enterprise 2.1.0 HA Environment Installation.....	46
Create Virtual Machines for ICP Nodes	47
Configure ICP Cluster Nodes	48
Access the IBM Cloud Private 2.1.0 Cluster Using Management Console.....	51
Install and Set Up kubectl 1.7.3	52
Configure kubectl.....	52
Configuring iSCSI Storage Access for IBM Cloud Private 2.1.0.1 Worker (Minion) Nodes	52
Installing IBM Storage Enabler for Containers	55
Create Storage Class Definitions	63
Add New ICP Worker Nodes.....	65
Remove ICP Worker Nodes	65
Applications Deployment on VersaStack with ICP	66
MongoDB	66
IBM DB2 Warehouse	72
Validated Hardware and Software	77
Validation.....	78
Test Plan.....	78
VersaStack Infrastructure Validation	78
IBM Cloud Private Environment Validation.....	78
Bill of Materials	78
Summary	79
References	80

Products and Solutions	80
Interoperability Matrixes.....	81
About the Authors.....	82
Acknowledgements	82

Executive Summary

Cisco Validated Designs (CVDs) deliver systems and solutions that are designed, tested, and documented to facilitate and improve customer deployments. These designs incorporate a wide range of technologies and products into a portfolio of solutions that have been developed to address the business needs of customers and to guide them from design to deployment.

Cisco and IBM have partnered to deliver a series of VersaStack solutions that enable strategic data center platforms with the above characteristics. VersaStack solutions deliver an integrated architecture that incorporates compute, storage and network design best practices thereby minimizing IT risks by validating the integrated architecture to ensure compatibility between various components. VersaStack solutions address common IT pain points by providing documented design and deployment guidance that can be referenced during various stages (planning, designing and implementation) of a deployment.

This “converged on-premises cloud” solution extends existing VersaStack capabilities that include both IBM and Cisco best-in-class hardware and software products. It adds easy-to-consume private cloud abilities to scalable and automate VersaStack infrastructure supporting virtualized and containerized environments.

Challenge

Microservices are an emerging architectural design pattern that brings agility and scalability to enterprise applications. Microservices need a responsive infrastructure (compute, network and storage) able to readily accommodate and intelligently connect the instantiation of dynamic, potentially short-lived business services in distinct containers. This means that having a cloud-service fabric is important to stand up microservices instances and have them connect and reliably offer the quality of service expected. These requirements represent a challenge to traditional enterprise data center infrastructures.

Solution

The VersaStack for IBM Cloud Private solution covered in this CVD is an end-to-end private cloud solution, including VersaStack with Cisco compute, Cisco network, and IBM storage. IBM Cloud Private technologies deliver the essential private cloud service fabric for building and managing on-premises, containerized applications, with guarantees to provide seamless access to persistent storage for stateful services such as database applications. This solution also comes with services for data, messaging, Java, blockchain, DevOps, analytics, and many others.

Solution Overview

Introduction

VersaStack solution is a pre-designed, integrated and validated architecture for data center that combines Cisco UCS servers, Cisco Nexus family of switches, Cisco MDS fabric switches and IBM Storwize and FlashSystem storage arrays into a single, flexible architecture. VersaStack is designed for high availability, with no single points of failure, while maintaining cost-effectiveness and flexibility in the design to support a wide variety of workloads.

As organizations are adopting both the private and provider clouds (public clouds), they expect continuous operation and easy access to resources. So IT cannot take a long time to add new servers and capacities or repurpose idle ones. VersaStack for IBM Cloud Private enables enterprises to move from technology silos to a cloud model that transforms enterprise data center infrastructure into pools of resources that can be easily allocated and repurposed. Applications can now run more efficiently within, between, and beyond data center boundaries and IT departments can evolve to platform-as-a-service (PaaS) to accelerate service delivery and increase revenue.

VersaStack for IBM Cloud Private abstracts the complexity of individual devices, hypervisors, and virtual machines (VMs) into a simplified model that makes them easy to manipulate and incorporate into automated processes. Users can request the resources they need through an easy-to-use, secure, self-service portal, and the system provisions the underlying infrastructure resources. The cloud infrastructure layers in organizations are synchronized and optimized, helping ensure that IT resources are available on demand. This intelligent system knows how objects fit together and can apply service profiles in a consistent manner. Thus, enterprises can have the confidence that the right equipment is quickly and easily provisioned for the workload that organizations need it to support.

The VersaStack for IBM Cloud Private design showcases:

Solution architecture, vital solution configuration, and the related solution configuration workflows with the following essential solution software and hardware modules:

- IBM Cloud Private 2.1.0 Enterprise Edition
- VersaStack Converged Infrastructure
- IBM Spectrum Connect
- IBM Storage Enabler for Containers
- Detailed technical configuration steps of building an end-to-end private cloud solution.

Audience

The intended audience of this document includes, but is not limited to, sales engineers, field consultants, professional services, IT managers, partner engineering, and customers who want to take advantage of an infrastructure built to deliver IT efficiency and enable IT innovation.

What's New?

The following design elements distinguish this version of VersaStack from previous models:

- Integration of IBM Cloud Private with VersaStack converged infrastructure
- Dynamic storage provisioning for application containers with IBM Spectrum connect

For more information about previous VersaStack models, please refer to the VersaStack documentation:

<http://www.cisco.com/c/en/us/solutions/enterprise/data-center-designs-cloud-computing/versastack-designs.html>

Solution Summary

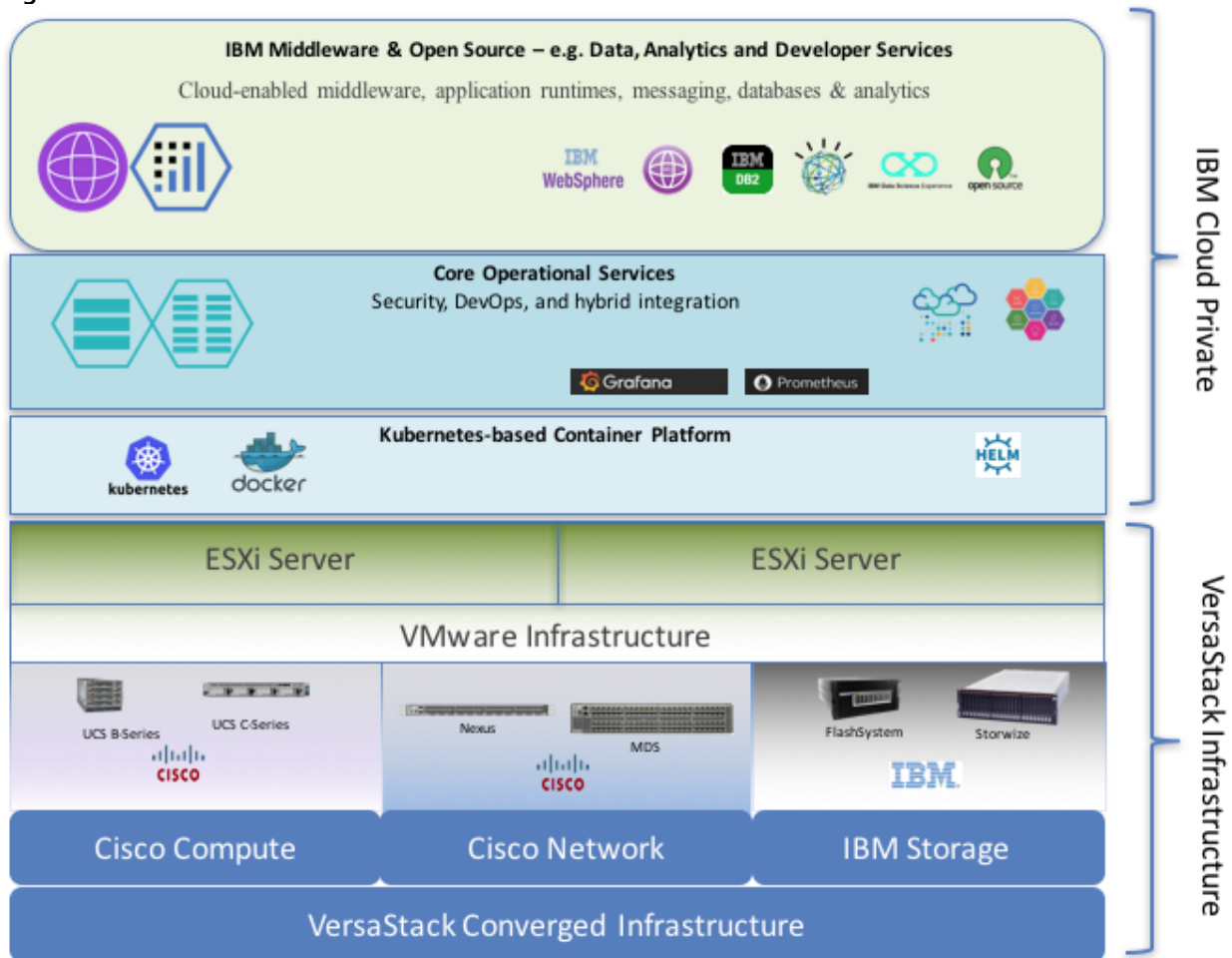
VersaStack for IBM Cloud Private provides a powerful solution offering everything your business needs to deploy a private cloud designed to deliver the economics and simplicity of the cloud, with the accessibility, virtualization, security and performance of an on-premises infrastructure.

This solution supports both traditional and emerging cloud native applications; it delivers extensive IT automation and cloud-like versatility for applications and data.

IBM Cloud Private Enterprise has at its core the infrastructure, analytics and services needed to make your private cloud foundation rock-solid. With Docker containers, Kubernetes, and Cloud Foundry at the base, enterprises can excel in their adoption of container technology as a strategic building block for increasing application developer productivity and decoupling teams to enable speed and agility.

Organizations can more easily and efficiently automate, deploy, scale and manage their containerized applications in their private data centers. With automated, built-in dashboards and analytics, organizations can manage applications and services across multiple clouds for flexibility and choice. Add in the services needed to do key functions like connecting APIs and data, while also monitoring, logging and managing events. A foundation built for rapidly producing quality code and accelerating business value.

Figure 1 VersaStack for IBM Cloud Private Solution Overview



VersaStack Program Benefits

Cisco and IBM have carefully validated and verified the VersaStack solution architecture and its many use cases while creating a portfolio of detailed documentation, information, and references to assist customers in transforming their data centers to this shared infrastructure model. This portfolio will include, but is not limited to the following items:

- Architectural design best practice
- Implementation and deployment instructions
- Technical specifications (rules for what is, and what is not, a VersaStack configuration)
- Cisco Validated Designs (CVDs) and IBM Redbooks focused on a variety of use cases

Cisco and IBM have also built a robust and experienced support team focused on VersaStack solutions. The team includes customer account and technical sales representatives as well as professional services and technical support engineers. The support alliance between IBM and Cisco provides customers and channel services partners with direct access to technical experts who are involved in cross-vendor collaboration and have access to shared lab resources to resolve potential multi-vendor issues.

Solution Benefits

IBM Cloud Private allows Cloud Operators to deploy Enterprise grade applications and middle-ware within minutes leveraging the VersaStack for IBM Cloud Private architecture. This platform was built on industry standard Open-source projects while also maintaining a consistent experience across Public and Private Clouds. Some of the key benefits for this solution stack are:

- Enterprise grade highly available and secure Infrastructure
- Persistent storage for application containers
- Multi-tenant containers and orchestration that is based on Kubernetes for creating microservices-based applications
- A common catalog of enterprise and open services to accelerate developer productivity
- A choice of compute models for rapid innovation in large enterprises, including infrastructure-as-code, Kubernetes, and Cloud Foundry
- Common base services to support the scalable management of microservices
- Automatic horizontal and non-disruptive vertical scaling of applications

Technology Overview

VersaStack for IBM Cloud Private, delivers unlimited capacities, cloud economics, ease of management, uncompromised versatility and agility ensuring enterprises stay ahead of the competitive curve. VersaStack for IBM Cloud Private consists of following components:

- VersaStack converged infrastructure (software and system hardware products from IBM and Cisco)
- IBM Cloud private (software)
- IBM Spectrum Connect (software)
- IBM Storage Enabler for Containers (software)

As part of the on-going collaboration spanning more than 15 years and tens of thousands of shared customers, IBM and Cisco offer VersaStack converged infrastructure solutions. VersaStack brings together Cisco UCS Integrated Infrastructure (including Cisco UCS servers, Cisco Nexus switches, and Cisco UCS Director management software) with market-leading IBM FlashSystem® and Storwize storage solutions. VersaStack supports a variety of Cisco and IBM component options that enable enterprises to easily build converged infrastructure solutions to address the full range of application workloads and business use cases.

The VersaStack components are connected and configured according to the best practices of both Cisco and IBM and provide an excellent platform for running a variety of enterprise workloads with confidence.

The technologies and solutions covered in each of these areas are outlined in the following sections.

VersaStack Converged Infrastructure

VersaStack for IBM Cloud Private includes VersaStack Converged Infrastructure and add-on IBM Cloud Private software components.

VersaStack for IBM Cloud Private requires VersaStack architecture with iSCSI storage. VersaStack converged infrastructure architecture in this solution is flexible and offers choice to customers, any supported VersaStack architecture with iSCSI storage can be used with in the solution as the infrastructure platform to support IBM Cloud Private. The compute, network and storage components are connected and configured according to the best practices of both Cisco and IBM; they provide an ideal platform for you to confidently run a variety of workloads.

The following are the validated and optional VersaStack architectures with iSCSI storage access, leveraging IBM SAN Volume Controller and IBM Storwize V5000 arrays.

Option1: VersaStack with Cisco UCS M5 and IBM SVC (Validated)

This particular VersaStack architecture has been leveraged as the private cloud infrastructure to support IBM Cloud Private for validating the solution.

VersaStack with Cisco UCS M5 servers and IBM SVC is comprised of the following infrastructure components for compute, network and storage.

- Cisco Unified Computing System (Cisco UCS)
- Cisco Nexus and Cisco MDS Switches
- IBM SAN Volume Controller, IBM FlashSystem and IBM Storwize family storage

The reference architecture is detailed in the following two Design and Deployment Guides:

Design Guide:

https://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/versastack_svc_ucsm5_designguide.html

Deployment Guide:

https://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/versastack_svc_ucsm5_depguide.html



While the current validated design guide utilizes VersaStack with Cisco UCS M5 and IBM SVC design referenced in the URLs above, customers can choose any supported VersaStack configuration as the converged infrastructure for private cloud including traditional and non-ACI VersaStack designs, some of the options are listed in the following section.

Option2: VersaStack with Cisco ACI and IBM SVC (Optional)

VersaStack with Cisco ACI and IBM SVC is comprised of the following infrastructure components for compute, network and storage:

- Cisco Unified Computing System (Cisco UCS)
- Cisco ACI, Cisco Nexus and Cisco MDS Switches
- IBM SAN Volume Controller, IBM FlashSystem and IBM Storwize family storage

The reference architecture is detailed in the following two Design and Deployment Guides:

Design Guide:

http://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/versastack_aci_svc_vmw6_design.html

Deployment Guide:

http://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/versastack_aci_svc_vmw6.html

Option3: VersaStack with Cisco UCS Mini and IBM Storwize V5000 iSCSI Storage (Optional)

VersaStack with Cisco UCS Mini and IBM Storwize V5000 is comprised of the following infrastructure components for compute, network and storage:

- Cisco Unified Computing System (Cisco UCS) Mini
- Cisco Nexus switches
- IBM Storwize V5000 family storage

The reference architecture is detailed in the following two Design and Deployment Guides:

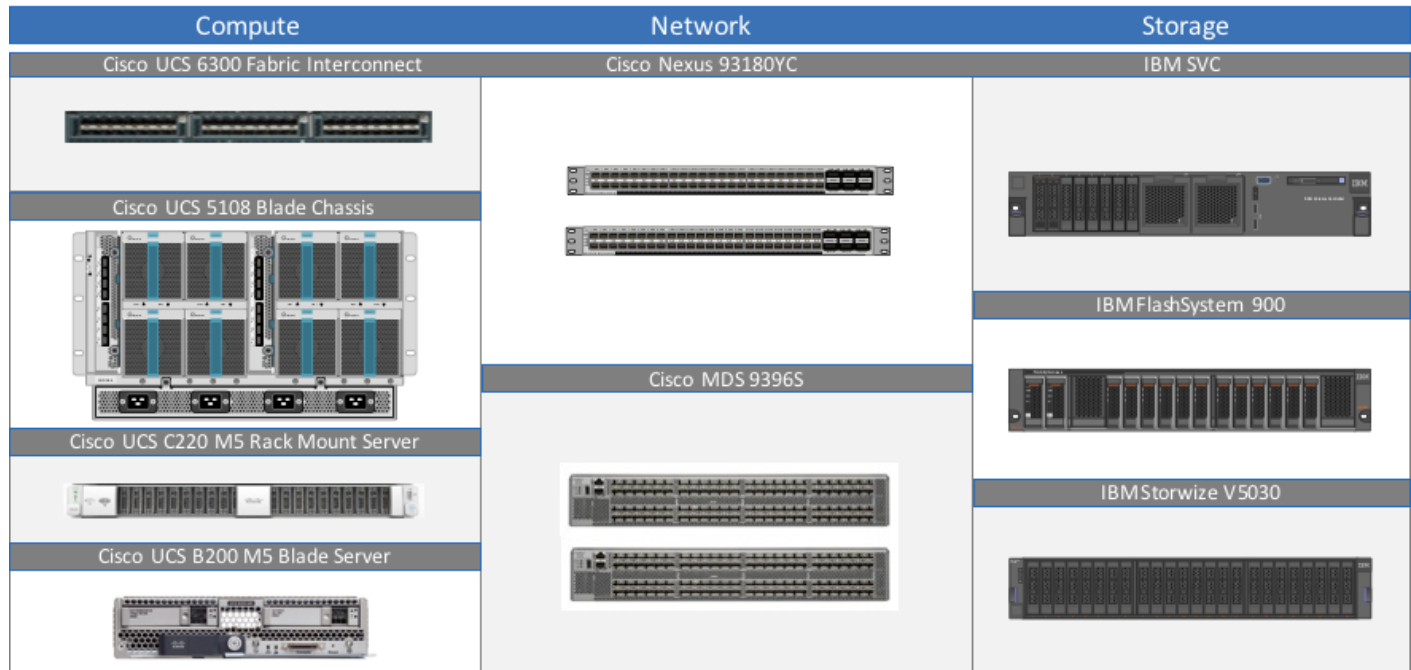
Design Guide:

https://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/versastack_n5k_2ndgen_mini_design.html

Deployment Guide: https://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/Versastack_n5k_mini.pdf

VersaStack Converged Infrastructure Components

The main components of the VersaStack data center are briefly introduced in this section and are not covered in detail in this document. For detailed information about these components and the design of the VersaStack Private Cloud, please refer to the Design and Deployment guides listed above.

Figure 2 VersaStack Converged Infrastructure – Components

One of the key benefits of VersaStack is the ability to maintain consistency in both scale up and scale down models. VersaStack can scale up for greater performance and capacity. In other words, you can add compute, network, or storage resources as needed; or it can also scale out where you need multiple consistent deployments like rolling out additional VersaStack modules. Each of the component families shown in Figure 2 (Cisco Unified Computing System, Cisco Switches, and IBM storage arrays) offer platform and resource options to scale the infrastructure up or down while supporting the same features and functionality.

Cisco Unified Computing System

The Cisco Unified Computing System (UCS) is a next-generation data center platform that integrates computing, networking, storage access, and virtualization resources into a cohesive system designed to reduce total cost of ownership (TCO) and to increase business agility. The system integrates a low-latency, lossless unified network fabric with enterprise-class, x86-architecture servers. The system is an integrated, scalable, multi-chassis platform where all resources are managed through a unified management domain.

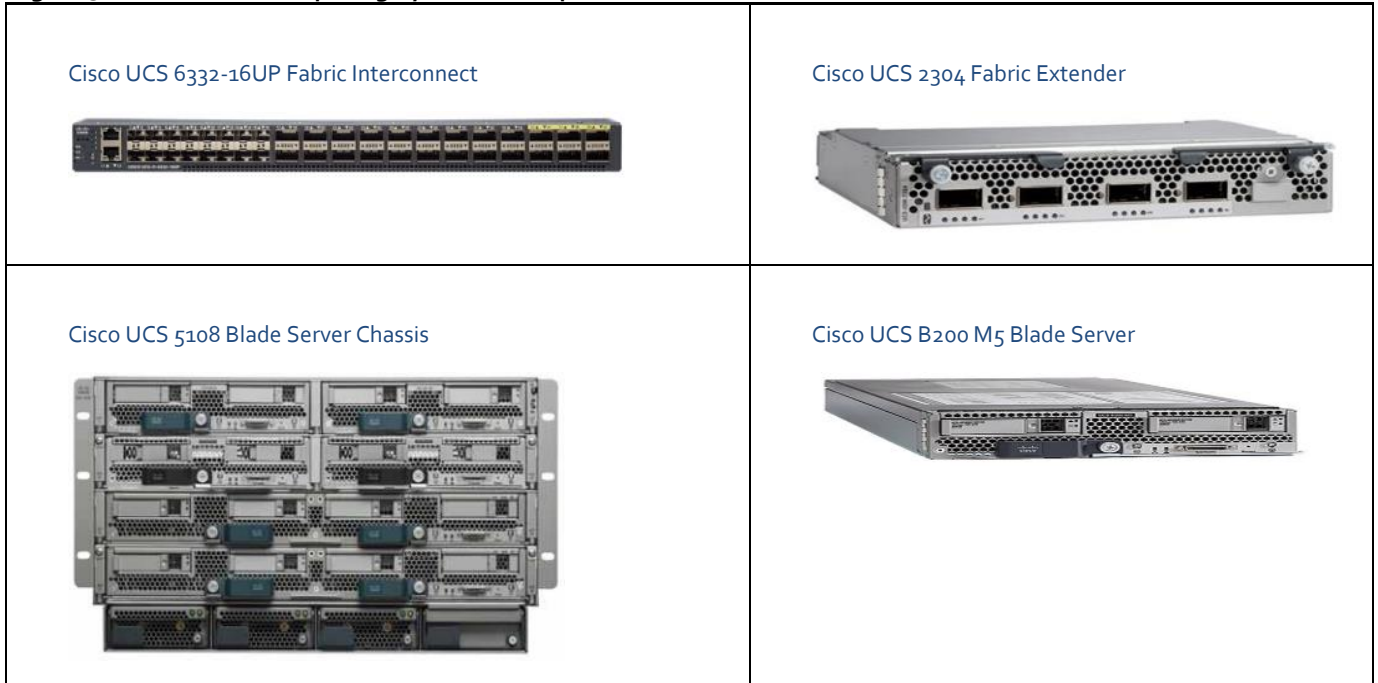
The Cisco Unified Computing System in the VersaStack architecture utilizes the following components:

- Cisco UCS Manager (UCSM)** provides unified management of all software and hardware components in the Cisco UCS to manage servers, networking, and storage configurations. The system uniquely integrates all the system components, enabling the entire solution to be managed as a single entity through Cisco UCSM software. Customers can interface with Cisco UCSM through an intuitive graphical user interface (GUI), a command-line interface (CLI), and a robust application-programming interface (API) to manage all system configuration and operations.
- Cisco UCS 6300 Series Fabric Interconnects** are a core part of Cisco UCS, providing both network connectivity and management capabilities for the system. The Cisco UCS 6300 Series offers line-rate, low-latency, lossless 10 and 40 Gigabit Ethernet, Fibre Channel over Ethernet (FCoE), and Fibre Channel functions. The unified fabric lowers costs by reducing the number of network adapters, switches, and cables, and by decreasing the power and cooling requirements.
- Cisco UCS 5108 Blade Server Chassis** supports up to eight blade servers and up to two fabric extenders in a six-rack unit (RU) enclosure.

- **Cisco UCS B-Series Blade Servers** provide performance, efficiency, versatility and productivity with the latest Intel based processors.
- **Cisco UCS C-Series Rack Mount Servers** deliver unified computing innovations and benefits to rack servers with performance and density to support a wide range of workloads.
- **Cisco UCS Network Adapters** provide wire-once architecture and offer a range of options to converge the fabric, optimize virtualization and simplify management.

For detailed information about the Cisco Unified Computing System product line, see:
<http://www.cisco.com/c/en/us/products/servers-unified-computing/index.html>

Figure 3 Cisco Unified Computing System – Components



Cisco Nexus 93180YC-EX

The Cisco Nexus 93180YC-EX Switch are 1RU switches that support 48 10/25-Gbps Small Form Pluggable Plus (SFP+) ports and 6 40/100-Gbps Quad SFP+ (QSFP+) uplink ports. All ports are line rate, delivering 3.6 Tbps of throughput. The switch supports Cisco Tetration Analytics Platform with built-in hardware sensors for rich traffic flow telemetry and line-rate data collection.

Figure 4 Cisco Nexus 93180YC-EX Switch



For detailed information about the Cisco Nexus 9000 product line, refer to:
<http://www.cisco.com/c/en/us/products/switches/nexus-9000-series-switches/models-listing.html>

Cisco MDS 9396S Fabric Switch

In the VersaStack design, Cisco MDS switches provide SAN connectivity between the IBM storage systems and Cisco UCS domain. The Cisco MDS 9396S 16G Multilayer Fabric Switch used in this design is the next generation of the highly reliable, flexible, and affordable Cisco MDS 9000 Family fabric switches. This powerful, compact, 2-rack-unit switch scales from 48 to 96 line-rate 16-Gbps Fibre Channel ports in 12 port increments. Cisco MDS 9396S is powered by Cisco NX-OS and delivers advanced storage networking features and functions with ease of management and compatibility with the entire Cisco MDS 9000 Family portfolio for reliable end-to-end connectivity. Cisco MDS 9396S provides up to 4095 buffer credits per group of 4 ports and supports some of the advanced functions such as Virtual SAN (VSAN), Inter-VSAN routing (IVR), port-channels and multipath load balancing and flow-based and zone-based QoS.



VersaStack for IBM Cloud Private solution leveraged iSCSI based storage only, however Cisco MDS switches were used to enable connectivity between the IBM SVC, IBM FS 900 and IBM V5030 storage arrays for cluster and node-to-node traffic.

Figure 5 Cisco MDS 9396S



For more information, refer to: <http://www.cisco.com/c/en/us/products/storage-networking/mds-9000-series-multilayer-switches/index.html>

IBM SAN Volume Controller

IBM SAN Volume Controller (SVC), is a combined hardware and software storage virtualization system with a single point of control for storage resources. SVC includes many functions traditionally deployed separately in disk systems and by including these in a virtualization system, SVC standardizes functions across virtualized storage for greater flexibility and potentially lower costs.

Built with IBM Spectrum Virtualize™ software—part of the IBM Spectrum Storage™ family—SVC helps organizations achieve better data economics by supporting the new workloads that are critical to business success. SVC systems can handle the massive volumes of data from mobile and social applications, enable rapid and flexible cloud services deployments, and deliver the performance and scalability needed to gain insights from the latest analytics technologies.

Additional SVC product information is available on the IBM SAN Volume Controller website: <http://www-03.ibm.com/systems/storage/software/virtualization/svc/>

VMware vCenter Server

VMware vCenter is the simplest and most efficient way to manage VMware vSphere hosts at scale. It provides unified management of all hosts and virtual machines from a single console and aggregates performance monitoring of clusters, hosts, and virtual machines. VMware vCenter Server gives administrators a deep insight into the status and configuration of compute clusters, hosts, virtual machines, storage, the guest OS, and other critical components of a virtual infrastructure. A single administrator can manage 100 or more virtualization environment workloads using VMware vCenter Server, more than doubling the typical productivity in managing physical infrastructure. VMware vCenter manages the rich set of features available in a VMware vSphere environment.

For more information, see: <http://www.vmware.com/products/vcenter-server/overview.html>

VersaStack for IBM Cloud Private Add-on Components

The following subsections describe the add-on solution components that are part of the solution along with the core VersaStack infrastructure.

IBM Cloud Private 2.1.0

IBM Cloud Private is an application platform for developing and managing on-premises, containerized applications. It is an integrated environment for managing containers that includes the container orchestrator Kubernetes, a private image repository, a management console, and monitoring frameworks.

IBM Cloud Private makes it easy to stand up an elastic runtime that is based on Kubernetes to address each of the following workload:

- Deliver packaged workloads for traditional middleware. IBM Cloud Private initially supports IBM Db2®, IBM MQ, Redis, and several other open source packages.
- Support API connectivity from 12-factor apps (software-as-a-service) to manage API endpoints within an on-premises datacenter.
- Support development and delivery of modern 12-factor apps with Microservice Builder.

Along with great capabilities to run enterprise workloads, IBM Cloud Private is delivers enhanced support to run processor-intensive capabilities, such as machine learning or data analytics quickly by taking advantage of GPU clusters.

The following are some of the features and use cases enabled by IBM Cloud Private:

Building and Deploying Cloud-Native Applications

Enterprises have a critical need to create cloud-native, 12 factor-based applications while adhering to the security and regulatory needs of their business. Cloud-native applications are built with a variety of runtimes, but application portability should be a key feature of any cloud platform, public or private. Here is where ICP enables users to build cloud-native applications with the tool chains with which they are familiar without compromising the enterprise's security and compliance requirements.

For more information about building and deploying cloud native microservices with IBM Cloud Private, please view this IBM Cloud Garage article: <https://www.ibm.com/cloud/garage/architectures/microservices/overview>

Application Modernization

Many enterprises have legacy applications that represent a great deal of investment. These applications are often monolithic and not easily extended to develop new applications. They can also be difficult to manage and require specialists with a great deal of specific expertise. For these reasons, enterprises want to modernize their legacy applications, making them cloud-enabled, componentized and consistently managed. One approach commonly considered for such application modernization is a microservices architecture. Essentially, the approach constructs systems out of a collection of small services, each with its own process that communicates via lightweight protocols. Refactoring legacy applications, or parts thereof, into microservices often makes the most sense to keep your existing systems running while you evolve to a more sustainable development model.

For a detailed discussion about refactoring legacy applications to microservices architecture, review this IBM Cloud Garage article: https://www.ibm.com/cloud/garage/content/code/microservices-practice/2_o

DevOps with Continuous Integration and Continuous Delivery

With legacy monolithic applications, there is a single build pipeline that all development work feeds into. All new features, enhancements and bug fixes must pass through this pipeline and be merged, built and tested before they can be released to the customer. As an application grows more complex, the release process also tends to become slower and more likely to break.

In contrast, there is no single release process for microservices. Each team building a micro-service can release an update at any time, without waiting for changes in other services to be merged, built and tested. A CI/CD process is critical to achieving the agility that microservices promise. A solid CI/CD process accommodates the multiple code bases and heterogeneous build environments that are common in micro-service architectures.

For CI/CD on ICP with Jenkins, please review this IBM Cloud Garage article:

<https://www.ibm.com/cloud/garage/content/course/cloud-private-jenkins-devops/o>

Metering

The metering service in IBM Cloud Private enables users to track products and provides insights into product usage.

The metering service helps track application instances and report usage metrics, while opening up possibilities to other hybrid cloud services. Using this service, organizations will have the opportunity to extend and achieve the benefits of cloud environments. For example, IT administrators will be able to register all of their products in a single dashboard, simplifying the management tasks. Capacity planners will be able to right-size their environments based on usage data.

The metering service runs within the IBM Cloud Private environment and receives information from the enabled software products. This information is then shown in the Metering section. The product and usage information about your enabled products is securely stored and viewed within the scope or context of that unique service.

For more information about the metering service, please visit the IBM Cloud Private InfoCenter:

https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0.2/manage_metrics/metering_service.html

Monitoring

Users can use the IBM Cloud Private cluster monitoring dashboard to monitor the status of their cluster and applications:

https://www.ibm.com/support/knowledgecenter/en/SSHLNR_8.1.4/com.ibm.pm.doc/install/integ_cloudprivate_monitorcluster.htm

RBAC

Role-based access control (RBAC) in ICP enables fine grained control over user permissions to perform certain tasks. Roles in ICP include super administrator, viewer, editor, operator and administrator.

For more information, please review:

https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0/user_management/assign_role.html

IBM Spectrum Connect (version 3.4.0)

IBM Spectrum Connect is a centralized server system that consolidates a range of IBM storage provisioning, automation, and monitoring solutions through a unified server platform. IBM Spectrum connect provides a single server backend location and enables centralized management of IBM storage resources for different virtualization and cloud platforms.

IBM Storage Enabler for Containers

Enables seamless access to IBM storage systems to the container orchestrator Kubernetes installed as part of IBM Cloud Private.

IBM Storage Enabler for Containers (open source project: Ubiquity) is made up of different loosely coupled components that are easy to extend. Refer to the following components:

- **IBM Storage Dynamic Provisioner for Kubernetes** – This container plug-in is responsible for creating or deleting a persistent volume. In case of a PersistentVolumeClaim (PVC) request, this container plug-in provisions the persistent volume if required based on storage class parameters.
- **IBM Storage Flex Volume for Kubernetes** – This container plug-in is responsible for attaching or detaching persistent volumes to a container. It has the ability to map and rescan, identify a multipath device, create a new file system, and mount volumes.
- **IBM Storage Enabler for Containers** - This container plug-in processes requests from flex and provisioner, and uses IBM Spectrum Connect for all storage actions. It also saves the state about the persistent volume name to a volume worldwide name (WWN) and the file system type (fstype).



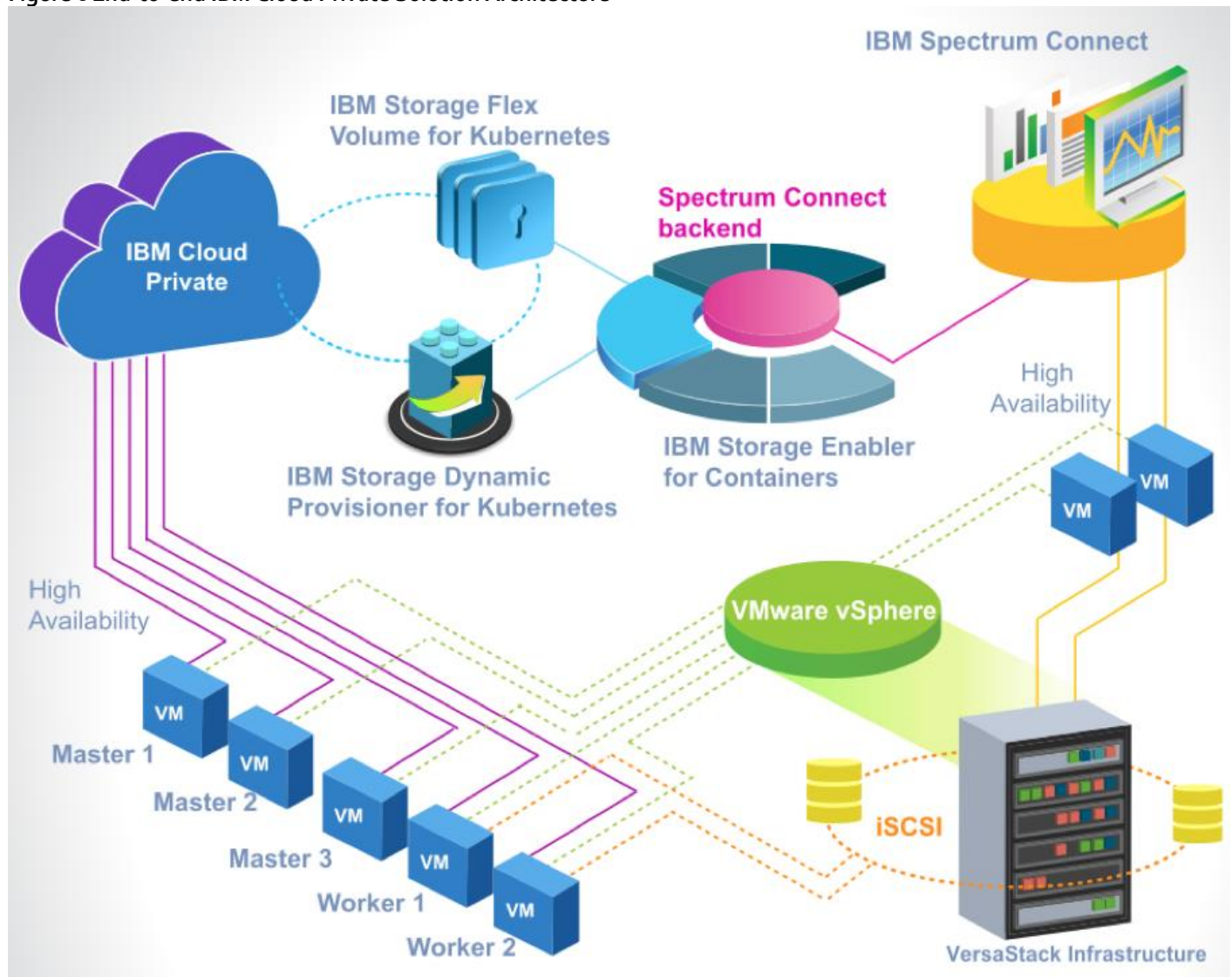
The IBM Ubiquity open source project enables persistent storage for the Kubernetes and Docker container frameworks. The IBM Storage Enabler for Containers is evolved from the IBM Ubiquity project.

Solution Design

Figure 6 illustrates the end-to-end private cloud solution architecture of VersaStack for IBM Cloud Private.

The Solution is highly scalable and consists of building blocks. These building blocks can be dynamically added in to the solution to add more capacity and performance to support the application needs. The IBM cloud private in this solution has been deployed on Ubuntu virtual machines running on VersaStack that consists of VMware vSphere 6.5 u1, installed on Cisco UCS M5 servers. The Cisco UCS M5 blade servers are connected over 40 Gbps Ethernet to Cisco UCS 6332 series fabric interconnects. The IBM Storwize IBM SAN Volume Controller (SVC) is connected to the Nexus based network fabric over 10Gbps Ethernet. The storage capacity from IBM FS900 and IBM Storwize V5030 has been provided to the hosts via IBM SVC storage system. The ESXi nodes supporting IBM cloud private environment and the ICP worker nodes access internet Small Computer System Interface (iSCSI) volumes for the boot, data stores and the application persistent volumes respectively.

Figure 6 End-to-end IBM Cloud Private Solution Architecture



VersaStack Architecture for ICP

The VersaStack for IBM Cloud Private solution is built on the VersaStack private data center architecture. The VersaStack private datacenter is the main building block with in this solution and is based on the VersaStack with Cisco UCS M5 and IBM SVC design.

The VersaStack with Cisco UCS M5 and IBM SVC architecture utilizes Cisco UCS platform with Cisco B200 M5 half-width blades connected and managed through Cisco UCS 6332-16UP Fabric Interconnects and the integrated Cisco UCS manager. These high performance servers are configured as stateless compute nodes where ESXi 6.5 U1 hypervisor is loaded using SAN (iSCSI and FC) boot. The boot disks to store ESXi hypervisor image and configuration along with the block and file based datastores to host application Virtual Machines (VMs) are provisioned on the IBM storage devices.

The link aggregation technologies play an important role in VersaStack solution providing improved aggregate bandwidth and link resiliency across the solution stack. Cisco UCS, and Cisco Nexus 9000 platforms support active port channeling using 802.3ad standard Link Aggregation Control Protocol (LACP). In addition, the Cisco Nexus 9000 series features virtual Port Channel (vPC) capability which allows links that are physically connected to two different Cisco Nexus devices to appear as a single "logical" port channel. Each Cisco UCS Fabric Interconnect (FI) is connected to both the Cisco Nexus 93180 switches using virtual port-channel (vPC) enabled 40GbE uplinks for a total aggregate system bandwidth of 80GBps. Additional ports can be easily added to the design for increased throughput. Each Cisco UCS 5108 chassis is connected to the UCS FIs using a pair of 40GbE ports from each IO Module for a combined 80GbE uplink. When Cisco UCS C-Series servers are used, each of the Cisco UCS C-Series servers connects directly into each of the FIs using a 10/40Gbps converged link for an aggregate bandwidth of 20/80Gbps per server.

The Design supports FC based storage access through the Cisco MDS 9396S switches and iSCSI based storage access through Cisco Nexus switches. Compute to storage system connectivity to support dynamic storage provisioning for IBM cloud private is iSCSI based through Cisco Nexus switches.



While storage access from the Cisco UCS compute nodes to IBM SVC storage nodes can be iSCSI or FC based, the IBM SVC nodes, FlashSystem 900 and Storwize V5030 communicate with each other on Cisco MDS 9396S based FC SAN fabric only.

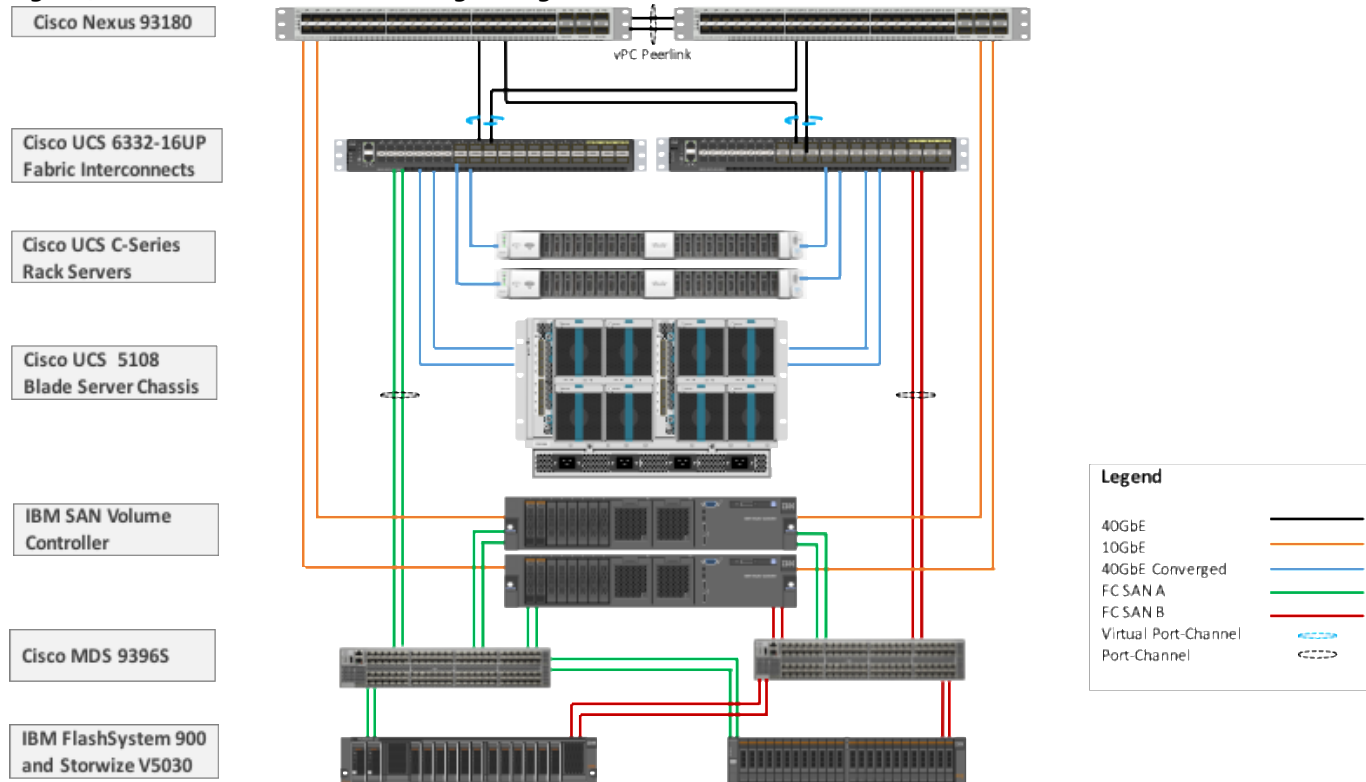
Figure 7 provides a high-level topology of the system connectivity. The VersaStack infrastructure satisfies the high-availability design requirements and is physically redundant across the network, compute and storage stacks. The integrated compute stack design presented in this document can withstand failure of one or more links as well as the failure of one or more devices.

IBM SVC nodes, IBM FlashSystem 900 and IBM Storwize V5030 are all connected using a Cisco MDS 9396S based redundant FC fabric. To provide FC based storage access to the compute nodes, Cisco UCS Fabric Interconnects are connected to the same Cisco MDS 9396S switches and zoned appropriately. To provide iSCSI based storage access, IBM SVC is connected directly to the Cisco Nexus 93180 switches. One 10GbE port from each IBM SVC node is connected to each of the two Cisco Nexus 93180 switches providing an aggregate bandwidth of 40Gbps.



VersaStack for IBM Cloud Private solution requires iSCSI storage access, but based on the customer requirements, the compute to storage connectivity in the SVC solution can be deployed as an iSCSI-only option or a combination of both. Figure 7 shows connectivity option to support both iSCSI and FC.

Figure 7 VersaStack iSCSI and FC Storage Design with IBM SVC



The VersaStack Private Cloud design is not discussed in detail but important elements to support VersaStack for IBM Cloud Private solution are covered. For detailed information on VersaStack architecture please refer to the following design and deployment guides:

https://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/versastack_svc_ucsm5_designguide.html

http://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/versastack_aci_svc_vmw6.html

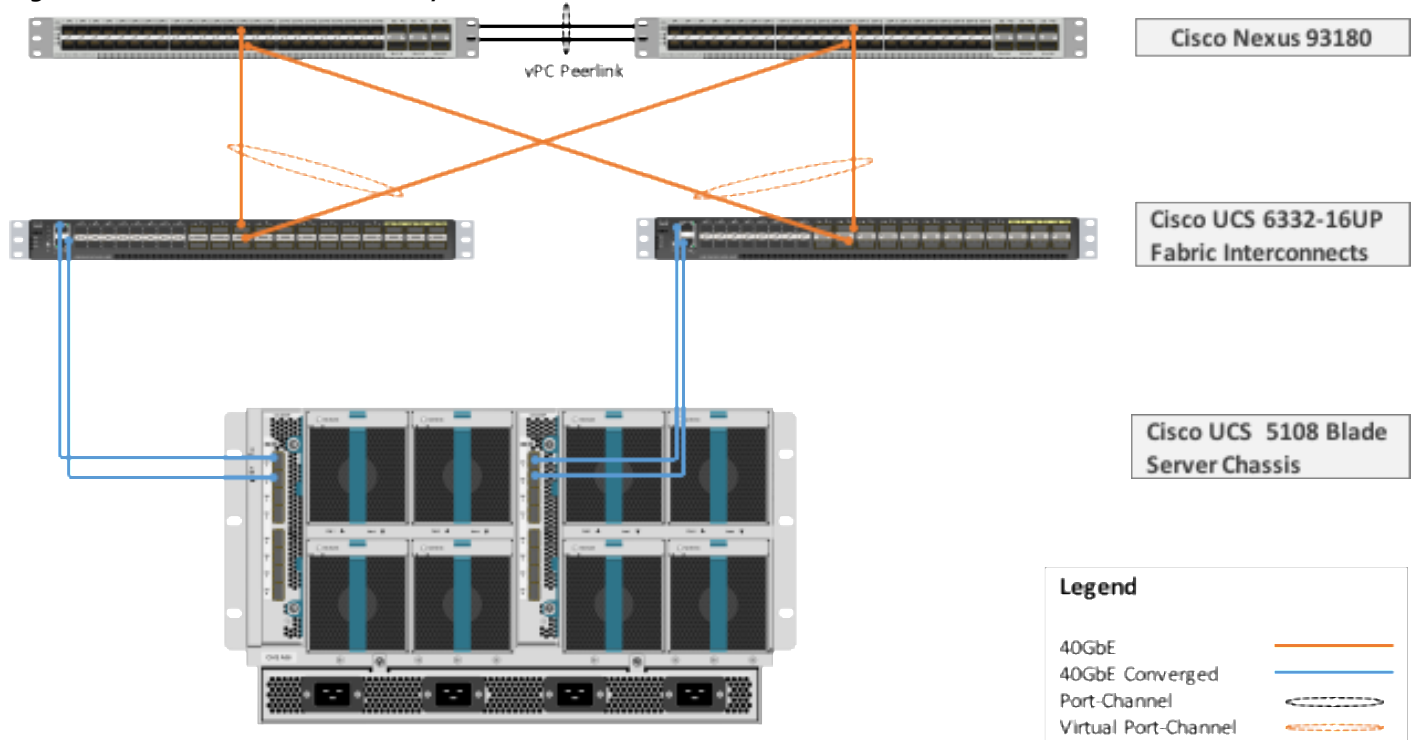
VersaStack iSCSI Design

IBM Cloud Private leverages internet small system interfaces (iSCSI) volumes for the container applications persistent storage. This section discusses the network connectivity and iSCSI storage design within the VersaStack architecture to enable dynamic storage provisioning for IBM Cloud Private.

Cisco UCS LAN Connectivity

Cisco UCS Fabric Interconnects are configured with two port-channels, one from each FI, to the Cisco Nexus 93180 switches. These port-channels carry all the data and IP-based storage traffic originated on the Cisco Unified Computing System. Virtual Port-Channels (vPC) are configured on the Cisco Nexus 93180 to provide device level redundancy. The validated design utilized two uplinks from each FI to the Cisco Nexus switches for an aggregate bandwidth of 160GbE (4 x 40GbE). The number of links can be increased based on customer data throughput requirements.

Figure 8 Cisco UCS - LAN Connectivity



IBM Storage Systems

IBM SAN Volume Controller, IBM FlashSystem 900 and IBM Storwize V5030, covered in this VersaStack design are deployed as high availability storage solutions. IBM storage systems support fully redundant connections for communication between control enclosures, external storage, and host systems.

Each storage system provides redundant controllers and redundant iSCSI to each controller to avoid failures at path as well as hardware level. For high availability, the storage systems are attached to two separate fabrics, A and B. If one fabric fault disrupts communication or I/O operations, the system recovers and retries the operation through the alternative communication path. Host (ESXi) systems are configured to use ALUA multi-pathing, and in case of network fault or node canister failure, the host seamlessly switches over to alternate I/O path.

IBM SAN Volume Controller – I/O Groups

The SAN Volume Controller design is highly scalable and can be expanded up to eight nodes in one clustered system. An I/O group is formed by combining a redundant pair of SAN Volume Controller nodes. In this design document, one pair of SVC IBM 2145-SV1 nodes (I/O Group 0) were deployed.

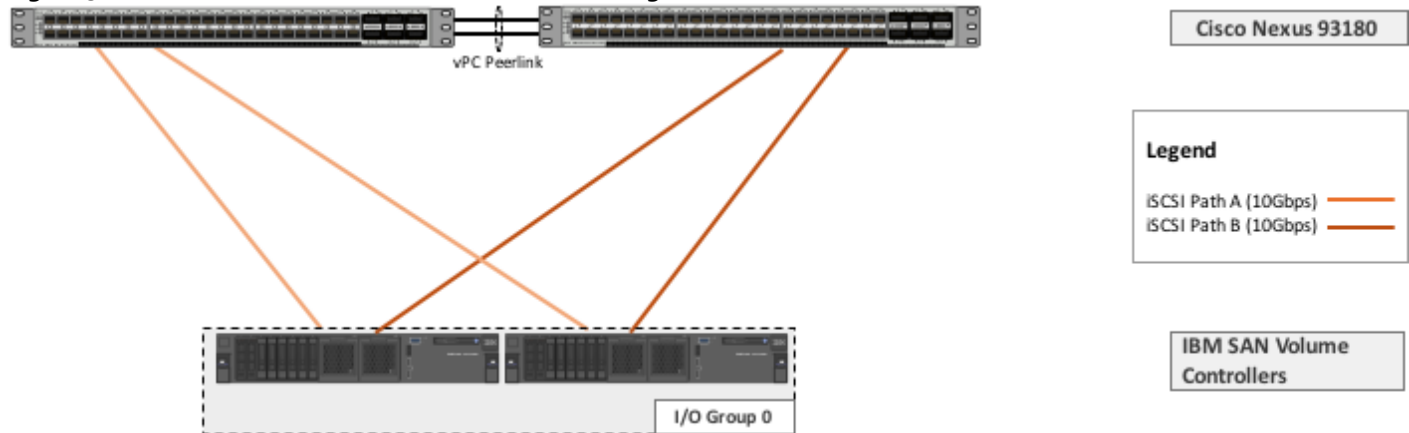


Based on the specific storage requirements, the number of I/O Groups in customer deployments will vary.

IBM SAN Volume Controller – iSCSI Connectivity

To support IP based iSCSI storage connectivity, each IBM SAN Volume Controller is connected to each of the Cisco Nexus 93180 switches. The physical connectivity is shown in Figure 9. Two 10GbE ports from each IBM SAN Volume Controller are connected to each of the two Cisco Nexus 93180 switches providing an aggregate bandwidth of 40Gbps. In this design, 10Gbps Ethernet ports between the SAN Volume Controllers I/O Group and the Nexus fabric utilize redundant iSCSI-A and iSCSI-B paths and can tolerate link or device failures. Additional ports can be easily added for additional bandwidth.

Figure 9 IBM SAN Volume Controller - iSCSI based Storage Access



Cisco UCS Server Configuration for Vsphere

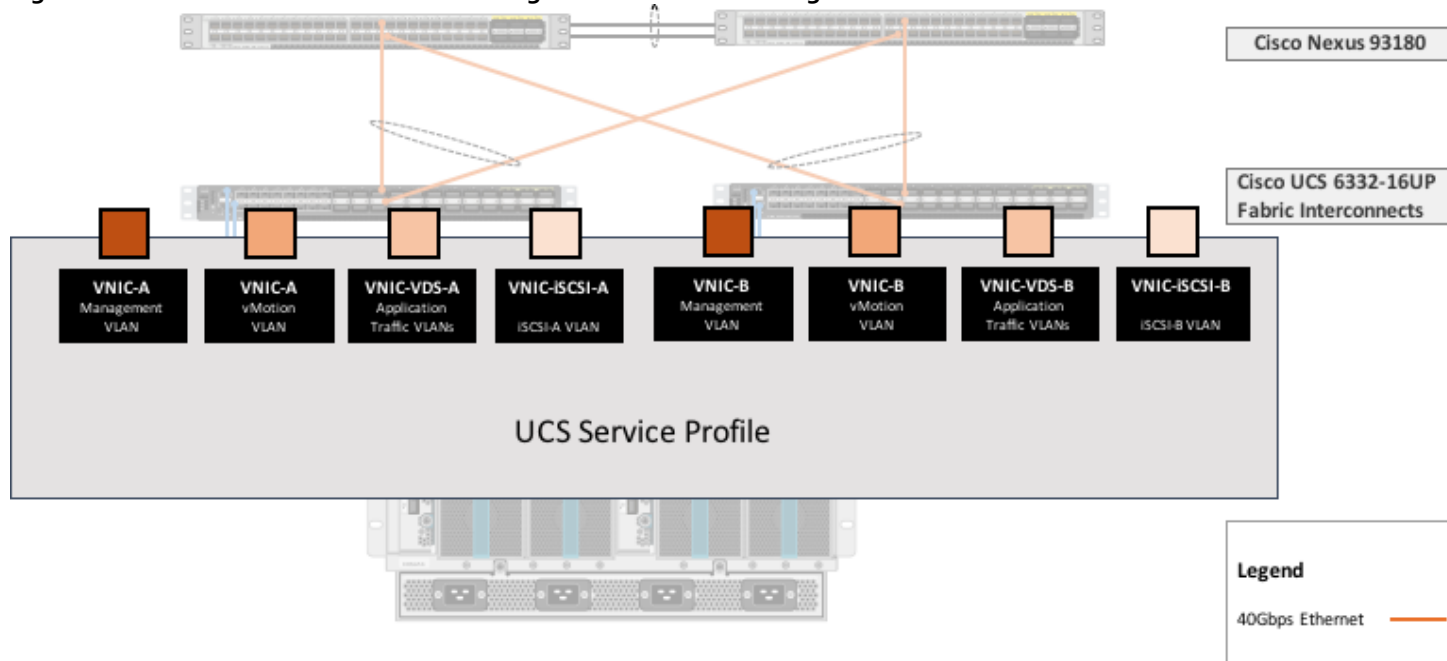
The ESXi nodes consist of Cisco UCS B200-M5 series blades with Cisco 1340 VIC. These nodes are allocated to a VMware High Availability (HA) cluster to support IBM cloud private infrastructure, services and applications. At the server level, the Cisco 1340 VIC presents multiple virtual PCIe devices to the ESXi node and the vSphere environment identifies these interfaces as VMNICs or VMhbas. The ESXi operating system is unaware of the fact that the NICs or HBAs are virtual adapters.

In the VersaStack design for iSCSI storage, eight vNICs are created and utilized as follows (Figure 10):

- One vNIC for iSCSI-A traffic
- One vNIC for iSCSI-B traffic
- Two vNICs for in-band management traffic
- Two vNICs for vMotion traffic
- Two vNICs for application related data including storage access if required. These vNICs are assigned to a distributed switch (vDS)

These vNICs are pinned to different Fabric Interconnect uplink interfaces and are assigned to separate vSwitches and virtual distributed switches based on type of traffic. The IBM cloud private virtual machines connectivity to virtual switches is covered later in the document.

Figure 10 Cisco UCS –Server Interface Design for iSCSI-based Storage



IBM Cloud Private Architecture

An IBM® Cloud Private cluster architecture consists of four main classes of nodes: boot, master, worker, and proxy.

You can optionally specify a management node in your cluster.

Boot Node

A boot or bootstrap node is used for running installation, configuration, node scaling, and cluster updates. Only one boot node is required for any cluster. You can use a single node for both master and boot.

Master Node

A master node provides management services and controls the worker nodes in a cluster. Master nodes host processes that are responsible for resource allocation, state maintenance, scheduling, and monitoring. Multiple master nodes are in a high availability (HA) environment to allow for failover if the leading master host fails. Hosts that can act as the master are called master candidates.

Worker Node

A worker node is a node that provides a containerized environment for running tasks. As demands increase, more worker nodes can easily be added to your cluster to improve performance and efficiency. A cluster can contain any number of worker nodes, but a minimum of one worker node is required.



The worker node is also referred to as minion, both terms have been used interchangeably in this document.

Proxy Node

A proxy node is a node that transmits external request to the services created inside your cluster. Multiple proxy nodes are deployed in a high availability (HA) environment to allow for failover if the leading proxy host fails. While you can use a

single node as both master and proxy, it is best to use dedicated proxy nodes to reduce the load on the master node. A cluster must contain at least one proxy node if load balancing is required inside the cluster.

Management Node

A management node is an optional node that only hosts management services like monitoring, metering, and logging. By configuring dedicated management nodes, you can prevent the master node from becoming overloaded.

Figure 11 IBM Cloud Private Architecture

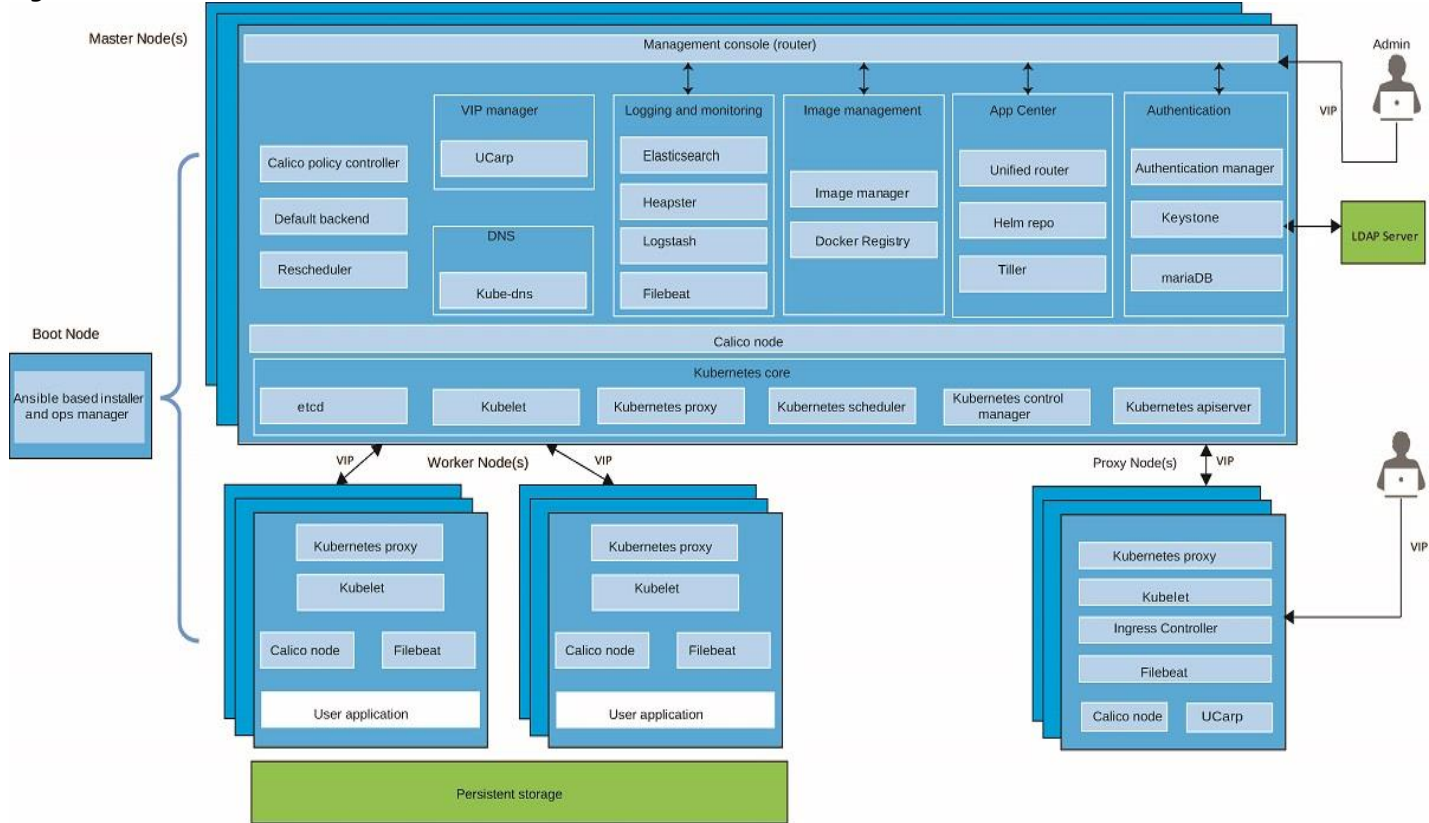


Table 1 below illustrates the IBM cloud private cluster nodes deployed in VersaStack for IBM Cloud Private solution validation. The master, proxy and management nodes are configured in the same server nodes.

For more details, refer to the Hardware requirements and recommendations for IBM Cloud Private 2.1.0 topic in IBM Knowledge Center at:

https://www.ibm.com/support/knowledgecenter/SSBS6K_2.1.0/supported_system_config/hardware_reqs.html

Table 1 ICP Cluster Nodes

Server Node	Function	Configuration
ICP Master1	Boot, Master, Proxy, Management	CPU : 8 vCPU Memory: 16 GB Storage: 200 GB

ICP Master2	Master, Proxy, Management	CPU : 8 vCPU Memory: 16 GB Storage: 200 GB
ICP Master3	Master, Proxy, Management	CPU : 8 vCPU Memory: 16 GB Storage: 200 GB
ICP Worker1	Worker	CPU : 8 vCPU Memory: 16 GB Storage: 200 GB
ICP Worker2	Worker	CPU : 8 vCPU Memory: 16 GB Storage: 200 GB

IBM Cloud Private Configuration on VersaStack

The ICP environment with in the design consists of a total of 7 virtual machines running on VMware ESXi 6.5 U1 servers.

Two virtual machines are created and configured with Red Hat Enterprise Linux 7.3 (64-bit) to install IBM Spectrum Connect version 3.4.0 with high availability configuration.

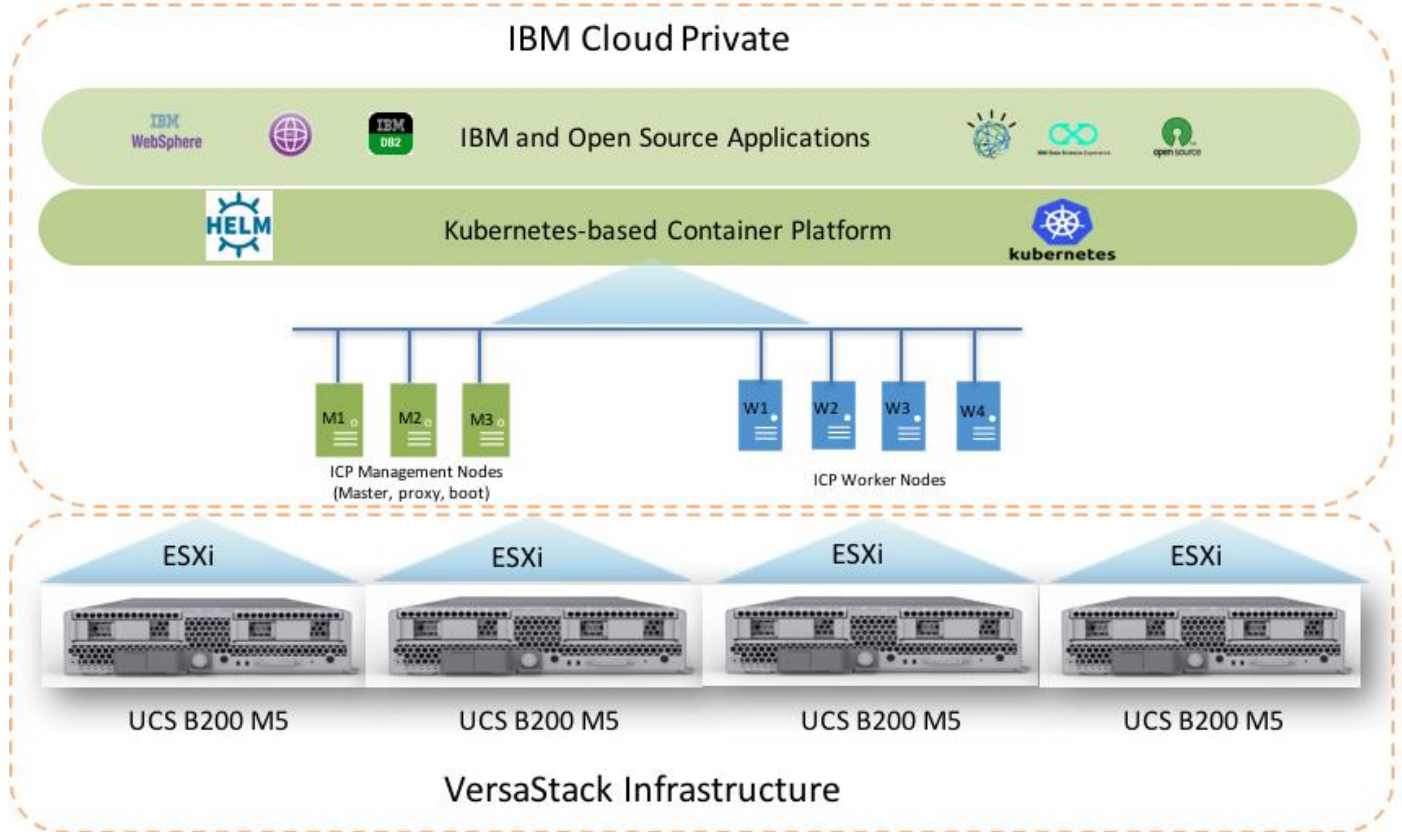
Five virtual machines are created and configured with Ubuntu Linux 16.04.3 (64-bit) to install IBM Cloud Private in three master, and two worker nodes configuration.

Table 2 Virtual Machines for Solution Validation

Solution module	Number of VMware vSphere VMs	Operating system
IBM Spectrum Connect	2	Red Hat Enterprise Linux 7.3 (64 bit)
IBM Cloud Private	5	Ubuntu 16.04.3 (64 bit)

Figure 12 illustrates VMware ESXi hypervisor running on Cisco UCS M5 servers that are part of VersaStack infrastructure. IBM Cloud Private components are deployed in virtual machines created on the VMware ESXi 6.5 U1 servers. For this solution validation, four Cisco UCS M5 servers were used to create a VMware high availability (HA) cluster.

Figure 12 VMware ESXi Running on Cisco UCS M5 Servers

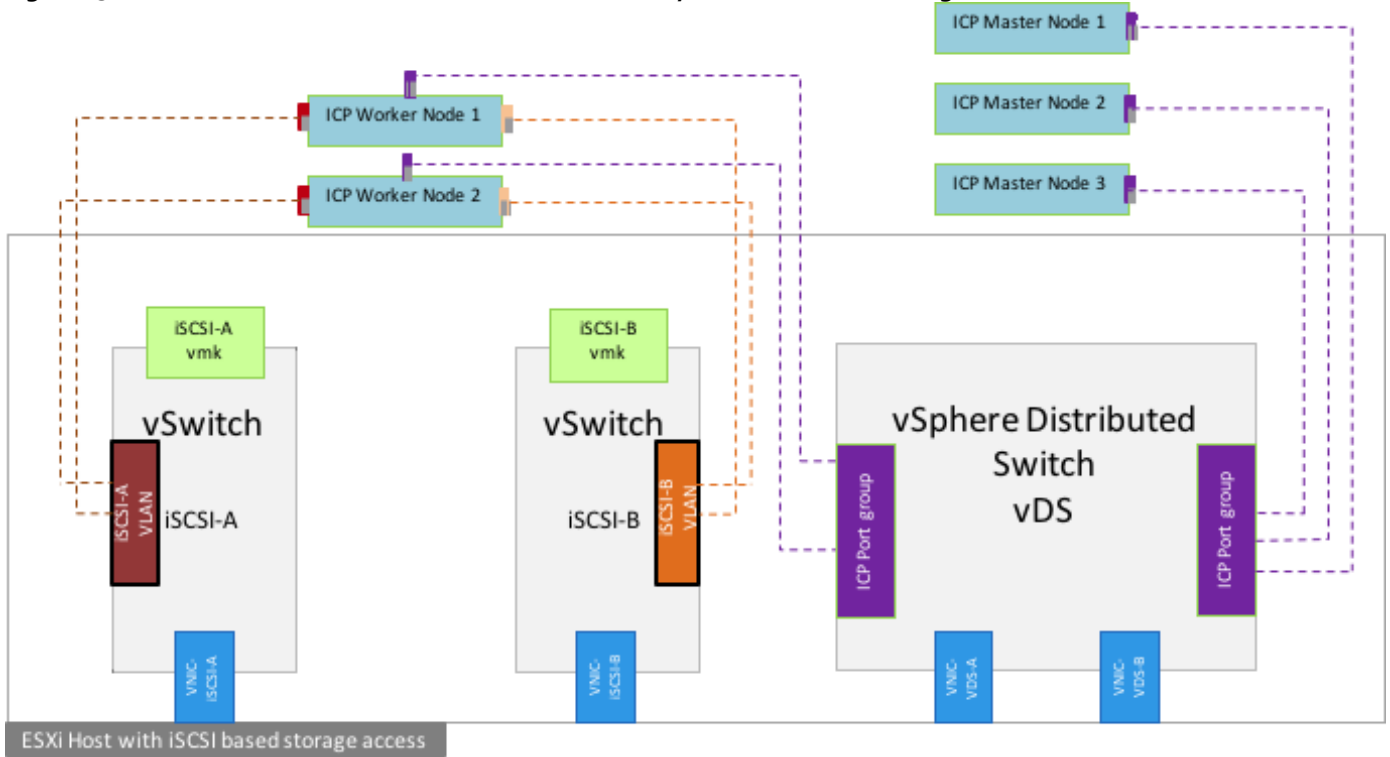


Virtual Switching Architecture

The VMware ESXi servers have, In-band management and vMotion traffic handled by infrastructure services vSwitches and iSCSI-A and iSCSI-B traffic is handled by two dedicated iSCSI vSwitches. The ESXi host configuration therefore has a combination of four vSwitches and a single distributed switch which handles application specific traffic.

IBM Cloud Private deployment utilizes one dedicated port group on the application VMware distributed switch (VDS) within the VMware ESXi servers on VersaStack. The Master nodes of ICP have one virtual NIC and has connectivity to this ICP port group on the vDS. However, the Worker nodes of ICP have three virtual NICs, one NIC connected to ICP port group for communication between the ICP nodes, the other two NICs are connected to the iSCSI virtual switches A and B to enable redundant storage access to the IBM SVC storage system.

Figure 13 VMware ESXi Host and ICP Nodes Connectivity for iSCSI-based Storage Access

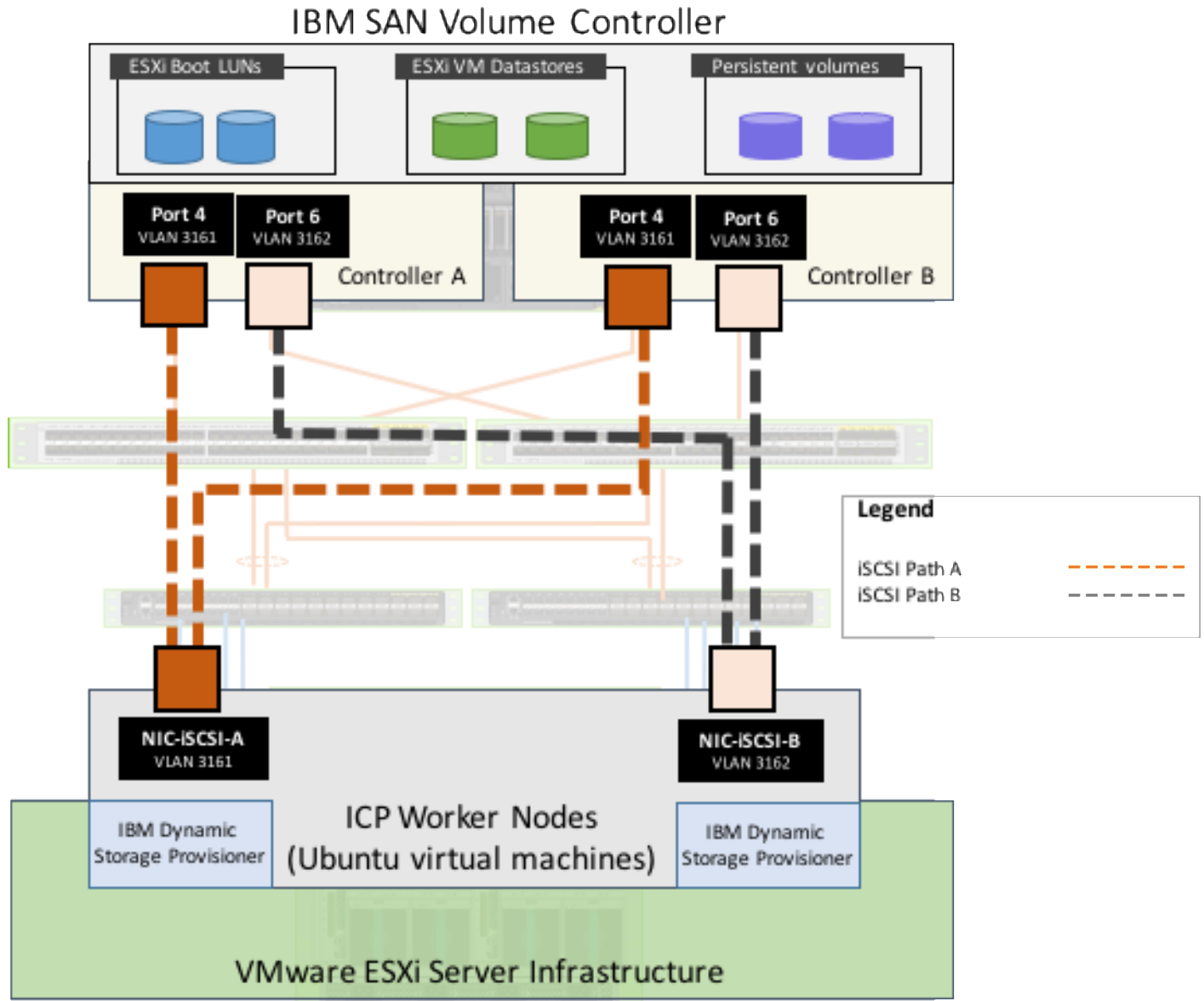


iSCSI VLAN Configuration for ICP Worker Nodes

To provide redundant iSCSI paths, two network interfaces are configured to use dedicated NICs for ICP worker node to storage connectivity. In this configuration, each virtual machine NIC port provided a different path that the iSCSI storage stack and its storage-aware multi-pathing plug-ins can use.

To setup iSCSI-A path between the worker nodes and the IBM SVC nodes, VLAN 3161 is configured on the Cisco UCS, Cisco Nexus and on the IBM SVC interfaces. To setup iSCSI-B path between the worker nodes and the IBM SVC, VLAN 3162 is configured on the Cisco UCS, Cisco Nexus and on the appropriate IBM SVC node interfaces. The dedicated iSCSI NICs on the ICP Worker nodes are allocated to the respective iSCSI port groups on the dedicated iSCSI vSwitches on VMware ESXi hosts.

Figure 14 Network Design – VLAN Mapping for iSCSI Storage Access



VersaStack for IBM Cloud Private Control and Data Paths

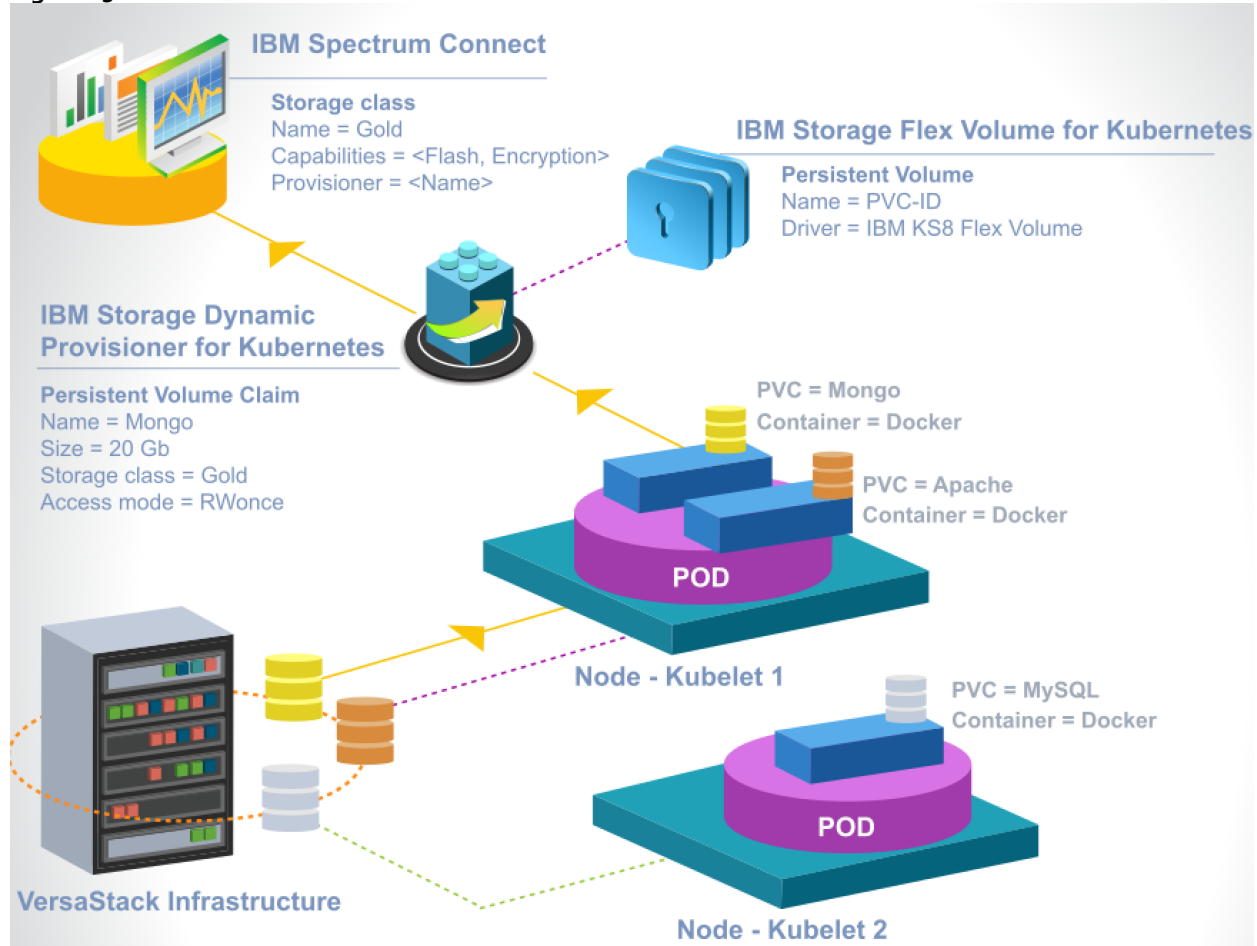
The Figure 15 illustrates the control and data paths of the VersaStack for IBM Cloud Private solution. In this diagram three storage classes such as gold, silver, and bronze types of volumes are created in the IBM VersaStack storage system. These storage volumes have established connection with IBM Cloud Private worker nodes using iSCSI configurations. IBM Storage Enabler for Containers dynamic provisioner (IBM Storage Kubernetes Dynamic Provisioner) creates a persistent volume based on a persistent volume claim, in this case, a MongoDB container as shown in Figure 15 with correct storage class definitions provided in IBM Spectrum Connect with the correct type of storage class volume. The IBM Storage Enabler for Containers flex volume driver then attaches or detaches the persistent volume with the MongoDB container.

IBM Storage Enabler for Containers enable a stateful container on IBM storage by provisioning Kubernetes flex volume driver and dynamic provisioner. IBM Storage Enabler for Containers integrates the following Kubernetes APIs.

- Kubernetes flex volume

- Kubernetes dynamic provisioner

Figure 15 Solution Architecture Control and Data Paths



Deployment Hardware and Software

The deployment of hardware and software for VersaStack for IBM Cloud Private covers the following:

- VersaStack Storage Configuration
- VMware virtual port-group configuration
- IBM Spectrum Connect installation and configuration
- IBM Cloud Private installation and configuration
- Sample application deployment

The existing deployment of the VersaStack architecture is assumed and the setup of these resources will have dependencies covered in the VersaStack with Cisco UCS M5 and IBM SAN Volume Controller Deployment Guide available here:

https://www.cisco.com/c/en/us/td/docs/unified_computing/ucs/UCS_CVDs/versastack_svc_ucsm5_depguide.html

VersaStack Storage Configuration

This section describes the configuration details of VersaStack IBM SVC storage system required to support ICP deployment. Detailed instructions to configure the storage system are not covered in this document. Please refer to the VersaStack with Cisco UCS M5 and IBM San Volume Controller design and deployment guides for more details.

The SAN Volume Controller makes it easy to configure multiple tiers of storage within the same SAN Volume Controller cluster. The system supports single-tiered storage pools, multi-tiered storage pools, or a combination of both. In a multi-tiered storage pool, MDisks with more than one type of disk tier attribute can be mixed, for example, a storage pool can contain a mix of generic_hdd and generic_ssd MDisks.

In the VersaStack with Cisco UCS M5 and IBM SVC design, both single-tiered and multi-tiered storage pools were utilized. These pools provide differentiated storage service to the IBM Cloud private application workloads as needed, based on characteristics such as capacity and performance.

Storage Pools Configuration

1. The following are the three storage pools defined to support dynamic storage provisioning for ICP, the storage capacity from these pools has been allocated for ICP solution storage configuration:
 - a. Gold Pool
 - b. Silver Pool
 - c. Bronze Pool

Name	State	Capacity
Bronze	✓ Online	135.00 GiB / 10.00 TiB (1%)
Gold	✓ Online	3.52 TiB / 10.00 TiB (35%)
Silver	✓ Online	2.02 TiB / 15.00 TiB (13%)

2. The following are the details of iSCSI configuration on IBM SVC system to support ICP solution on VersaStack:

a. IBM SVC nodes iSCSI interface details:

The screenshot shows the 'Ethernet Ports' configuration page in the VersaStack-SVC interface. The page title is 'Ethernet Ports' and it includes a sub-header: 'The Ethernet ports can be used for iSCSI connections, host attachment, and remote copy.' Below this is a table with columns: Name, Port, State, IP, Speed, Host Attach, Remote Copy, Storage Port IPv4, and Storage Port IPv6. The table lists configurations for two nodes (node1 and node2) across seven ports (1-7). Ports 4, 5, and 6 are configured with IP addresses and 10Gb/s speeds, while others are unconfigured or disabled.

Name	Port	State	IP	Speed	Host Attach	Remote Copy	Storage Port IPv4	Storage Port IPv6
io_grp0								
node1	1	Unconfigured		1Gb/s	No		Disabled	Disabled
node2	1	Unconfigured		1Gb/s	No		Disabled	Disabled
node1	2	Unconfigured			No		Disabled	Disabled
node2	2	Unconfigured			No		Disabled	Disabled
node1	3	Unconfigured			No		Disabled	Disabled
node2	3	Unconfigured			No		Disabled	Disabled
node2	4	Configured	10.29.161.250	10Gb/s	Yes		Enabled	Disabled
node1	4	Configured	10.29.161.249	10Gb/s	Yes		Enabled	Disabled
node2	5	Unconfigured			No		Disabled	Disabled
node1	5	Unconfigured			No		Disabled	Disabled
node2	6	Configured	10.29.162.250	10Gb/s	Yes		Enabled	Disabled
node1	6	Configured	10.29.162.249	10Gb/s	Yes		Enabled	Disabled
node2	7	Unconfigured			No		Disabled	Disabled
node1	7	Unconfigured			No		Disabled	Disabled

b. IBM SVC nodes iSCSI name (IQN) details:

The screenshot shows the 'iSCSI Configuration' page in the VersaStack-SVC interface. The page title is 'iSCSI Configuration' and it includes a sub-header: 'Configure system properties to connect to iSCSI-attached hosts.' The configuration includes a 'Name' section with a 'System Name' field containing 'VersaStack-SVC'. Below this is an 'iSCSI Aliases (optional)' section with a table for 'Node Name', 'iSCSI Alias', and 'iSCSI Name (IQN)'. The table lists configurations for node1 and node2 with their respective IQN values. There is also an 'iSNS (optional)' section with an 'iSNS Address' field and a 'Modify CHAP Configuration' link.

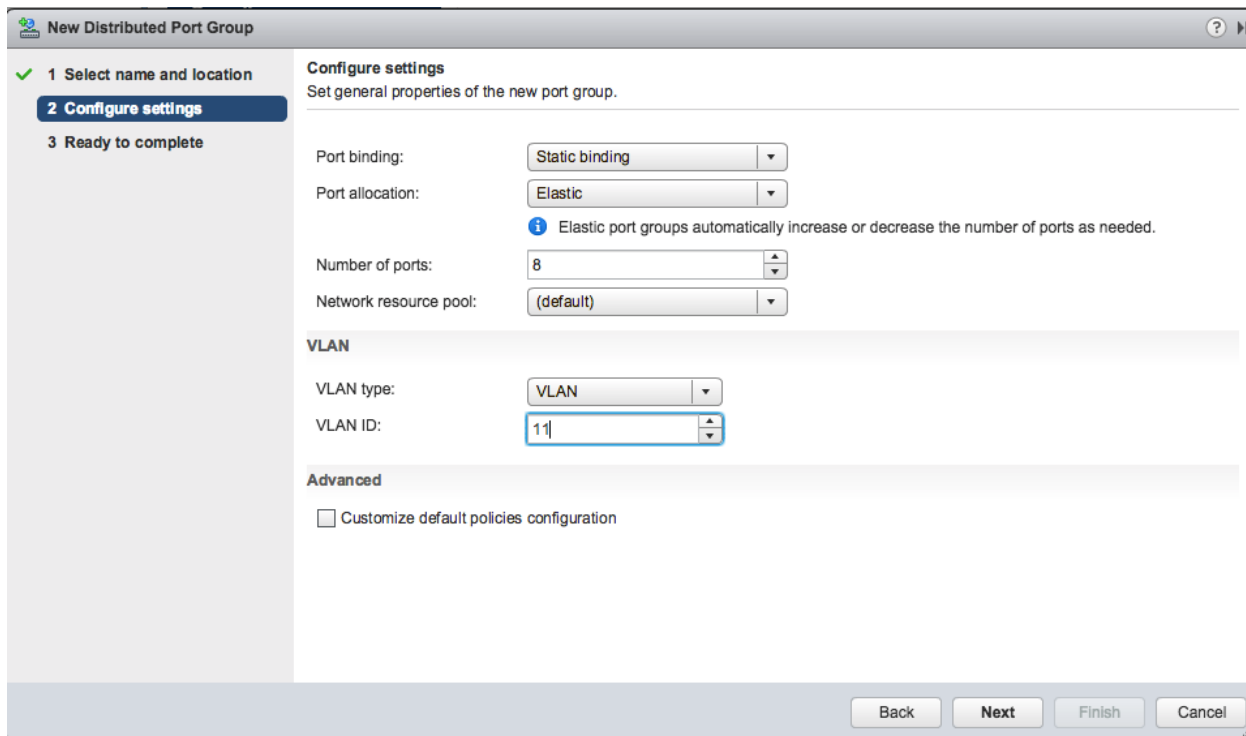
Node Name	iSCSI Alias	iSCSI Name (IQN)
node1		iqn.1986-03.com.ibm:2145.versastack-svc.node1
node2		iqn.1986-03.com.ibm:2145.versastack-svc.node2

Create a New VMware Port Group for ICP Environment

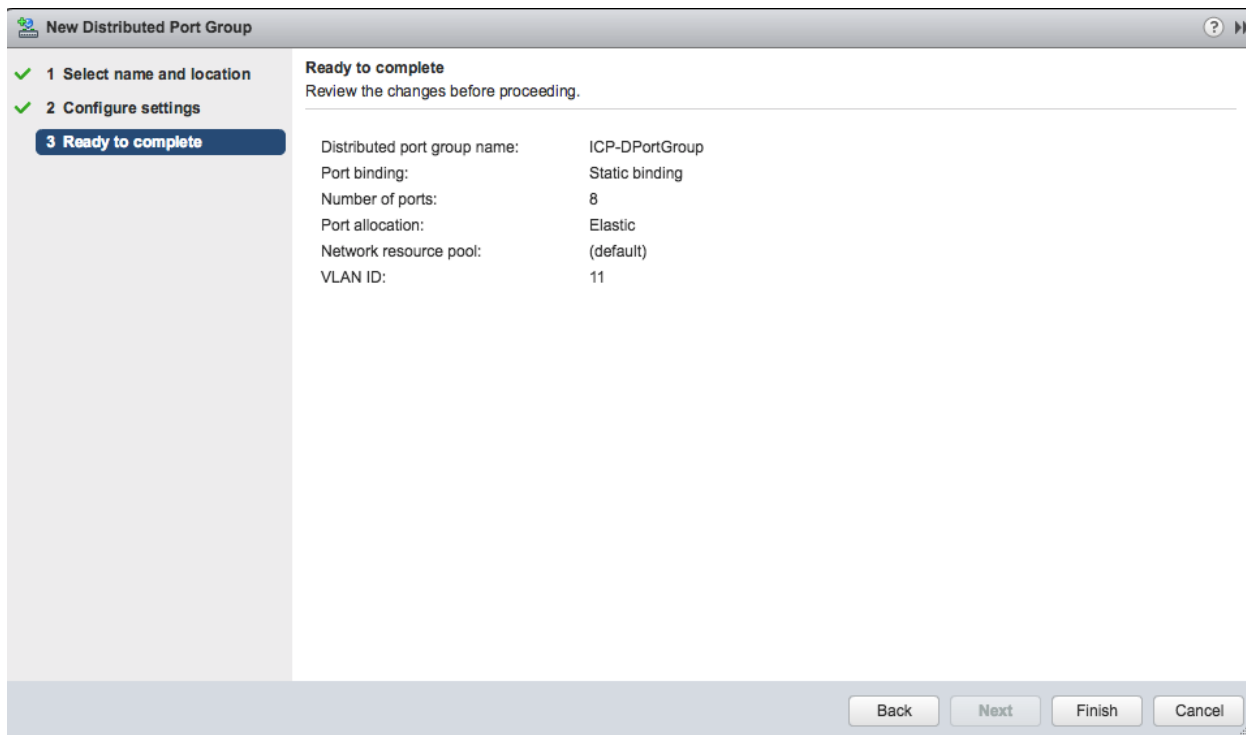
To create a new distributed port group on the VersaStack application (VDS) switch to support ICP traffic, complete the following steps:

1. Login to VMware vCenter web client and select the **Networking** inventory view.
2. Select Inventory > vSphere Distributed Switch > New Port Group.
3. Enter a Name (ICP-DportGroup) for the new distributed port group.

- 4. Select VLAN type and VLAN id for the ICP traffic.



- 5. Click **Next** and click **Finish**.



IBM Spectrum Connect 3.4.0 Installation

Table 3 lists the IBM spectrum connect environment and the virtual machine hardware specification on which IBM spectrum connect 3.4.0 is installed.

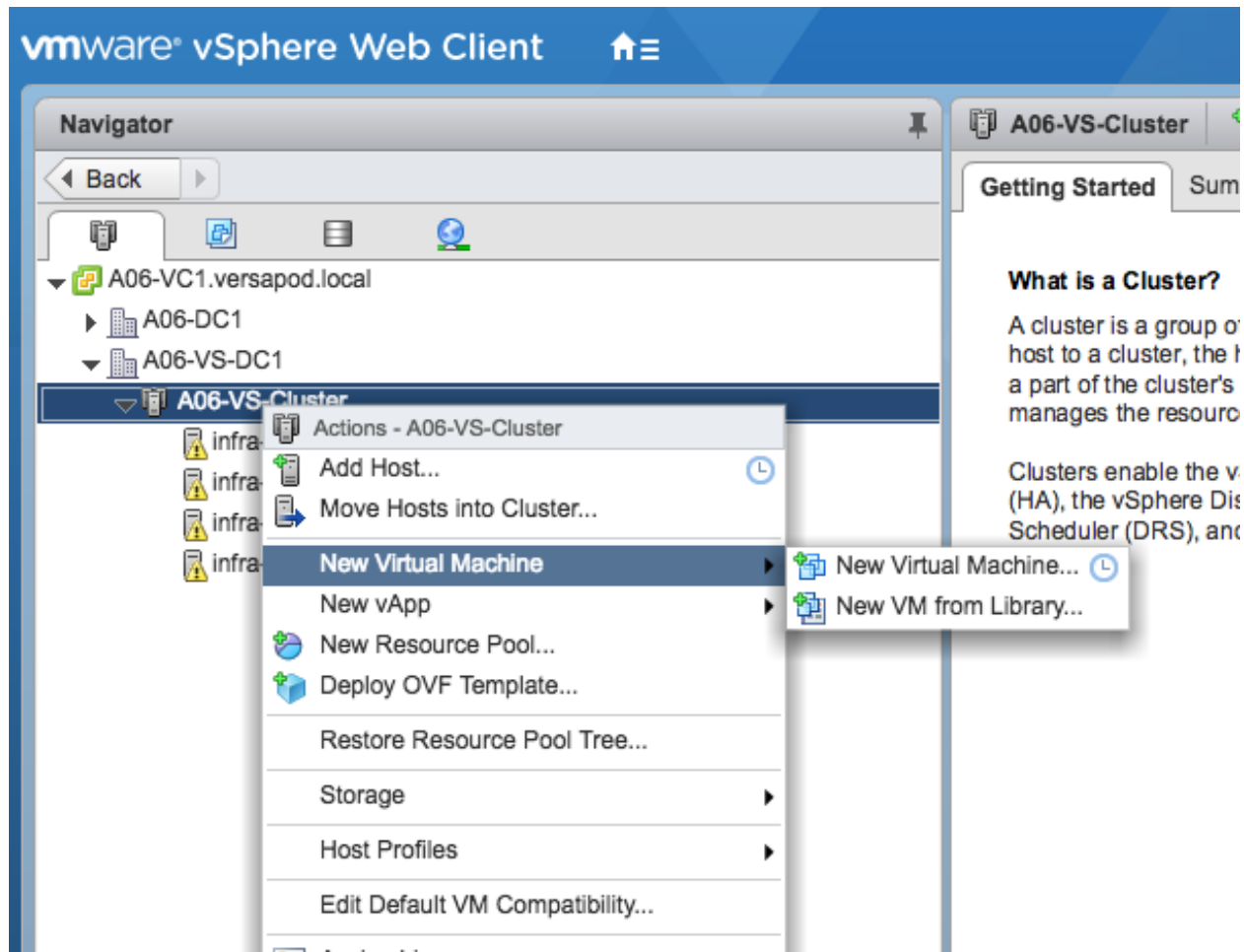
Table 3 IBM Spectrum Connect

Operating system	Processor	Memory and storage capacity
Red Hat Enterprise Linux 7.3 (64 bit)	8 vCPU (64-bit dual-core)	8 GB RAM and 200 GB free disk space

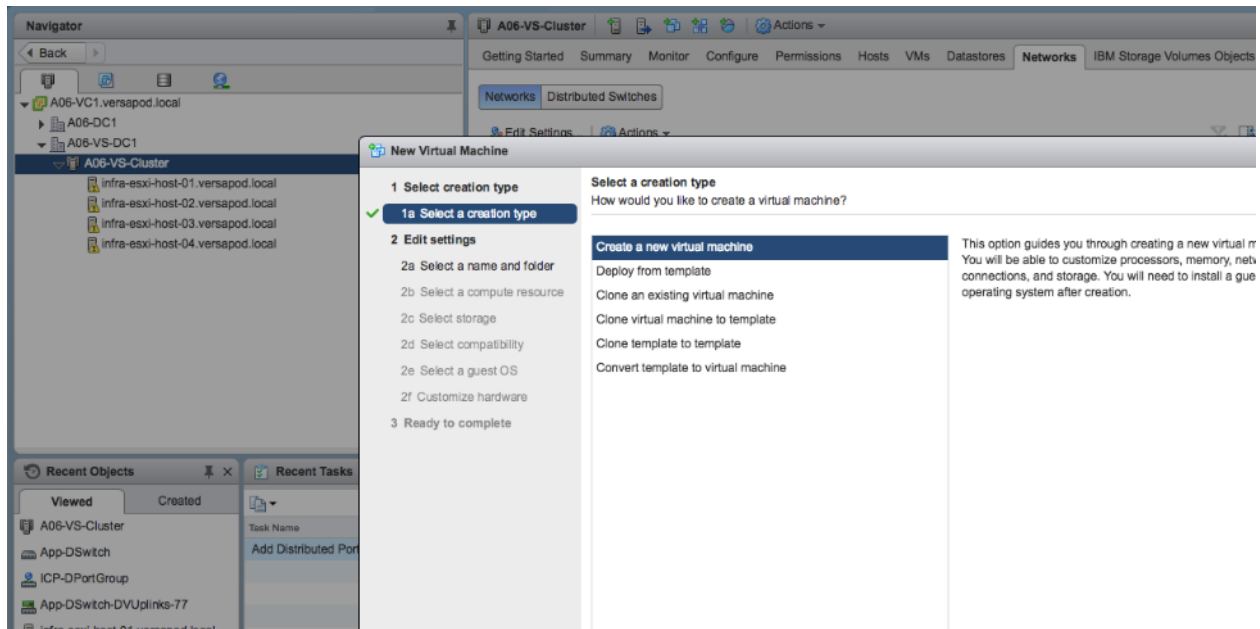
Install Virtual Machine for IBM Spectrum Connect

To create a new virtual machine for Spectrum connect installation, complete the following steps:

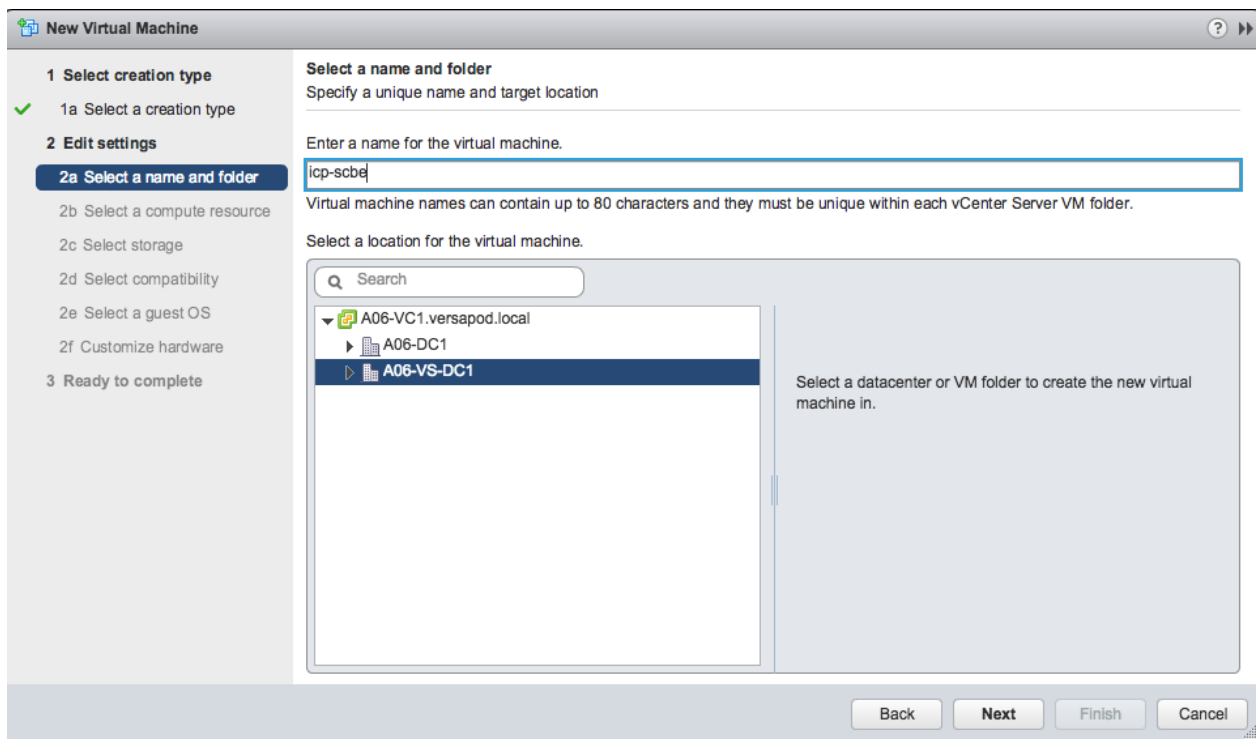
1. Login to VMware vCenter web client.
2. Right-click the <VersaStack> cluster and select New Virtual Machine.



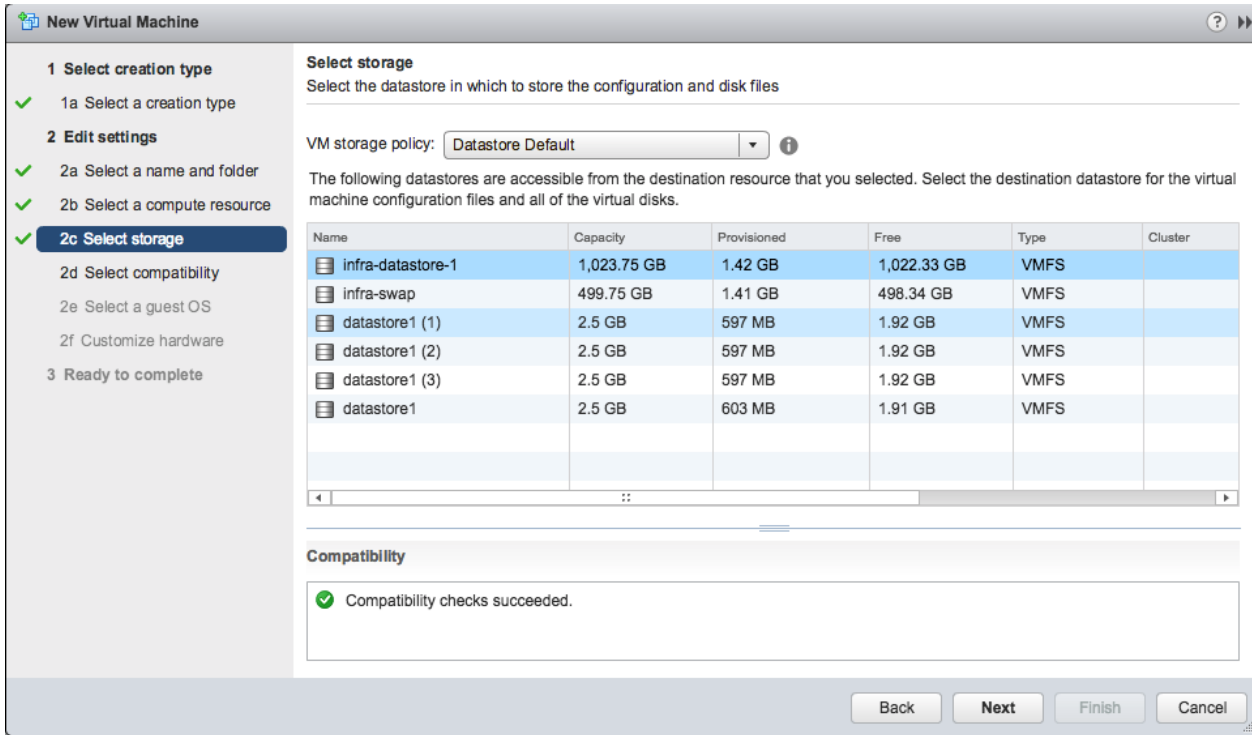
3. Select Create a new virtual machine and click **Next**.



4. Enter a name for the virtual machine, select the <VersaStack> datacenter for VM deployment.

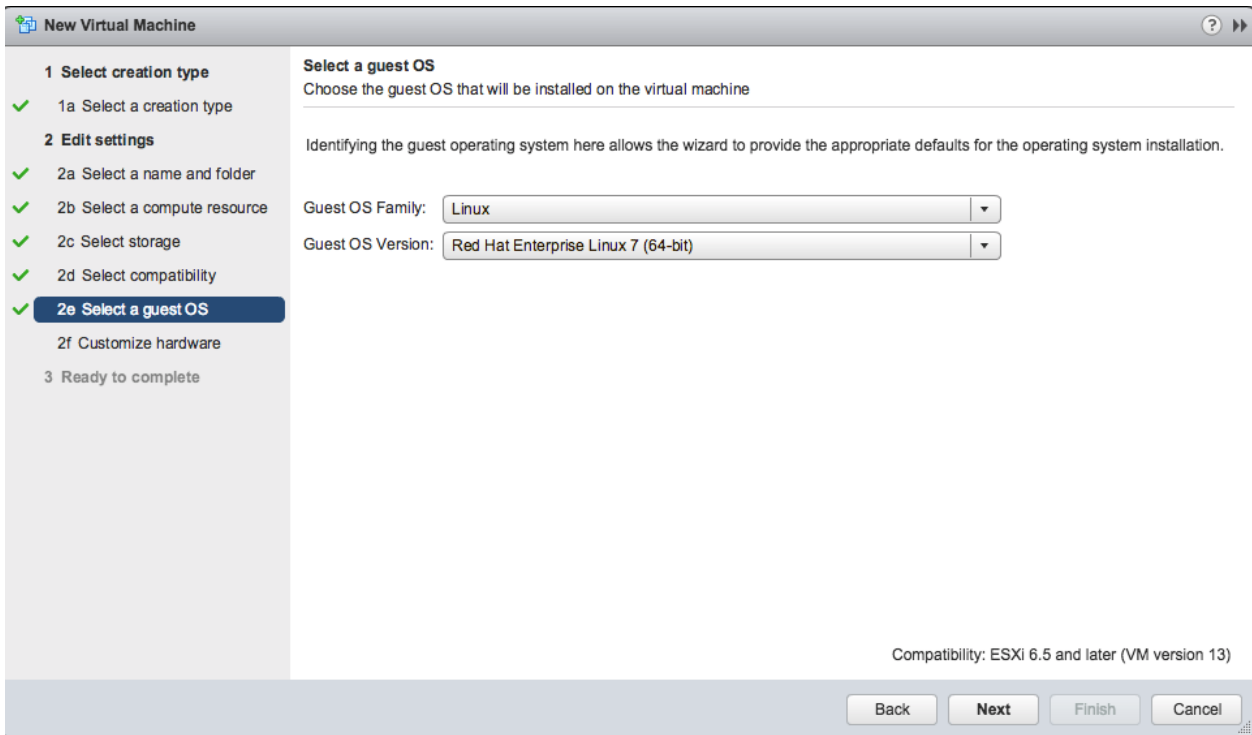


5. Click **Next**.
6. Select <VersaStack> cluster for the virtual machine, click **Next**.
7. Select datastore and click **Next**.



8. Select the compatibility from the drop-down list and click **Next**.

9. Select a guest operating system (RHEL 7) and click **Next**.



10. Customize the Virtual Machine Hardware according to the hardware recommendation for Spectrum connect virtual machine. Connect the operating system from the pre-loaded Content Library.



Optionally, the operating System disk can be mounted after the virtual machine is created for the OS installation.

New Virtual Machine

1 Select creation type
 1a Select a creation type
 2 Edit settings
 2a Select a name and folder
 2b Select a compute resource
 2c Select storage
 2d Select compatibility
 2e Select a guest OS
2f Customize hardware
 3 Ready to complete

Customize hardware
 Configure the virtual machine hardware

Virtual Hardware | VM Options | SDRS Rules

*CPU	2	
*Memory	4	GB
New Hard disk	20	GB
New SCSI controller	VMware Paravirtual	
*New Network	ICP-DPortGroup (App-DSwitch)	<input checked="" type="checkbox"/> Connect...
*New CD/DVD Drive	Content Library ISO File	<input checked="" type="checkbox"/> Connect... ✕
Video card	Specify custom settings	
VMCI device		
New SATA Controller		
Other Devices		

New device:

Compatibility: ESXI 6.5 and later (VM version 13)

11. Click **Finish** to complete the virtual machine creation.

New Virtual Machine

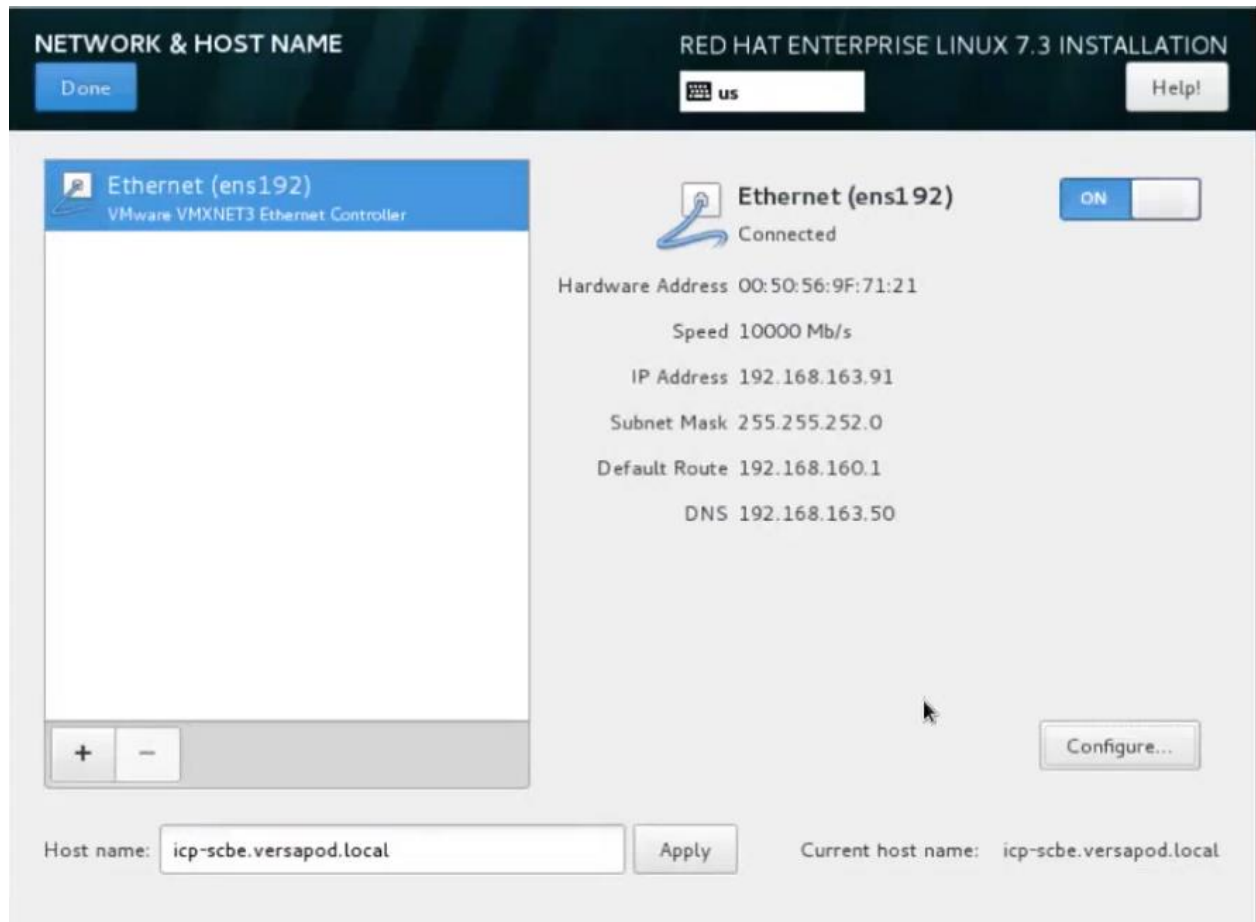
1 Select creation type
 1a Select a creation type
 2 Edit settings
 2a Select a name and folder
 2b Select a compute resource
 2c Select storage
 2d Select compatibility
 2e Select a guest OS
 2f Customize hardware
3 Ready to complete

Provisioning type:	Create a new virtual machine
Virtual machine name:	icp-scbe
Folder:	A06-VS-DC1
Cluster:	A06-VS-Cluster
Datastore:	infra-datastore-1
Guest OS name:	Red Hat Enterprise Linux 7 (64-bit)
CPUs:	2
Memory:	4 GB
NICs:	1
NIC 1 network:	ICP-DPortGroup (App-DSwitch)
NIC 1 type:	VMXNET 3
SCSI controller 1:	VMware Paravirtual
Create hard disk 1:	New virtual disk
Capacity:	20 GB
Datastore:	infra-datastore-1
Virtual device	SCSI(0:0)
node:	
Mode:	Dependent

Compatibility: ESXI 6.5 and later (VM version 13)

12. Boot the virtual machine and access the virtual machine console from VMware web client.

13. Continue with the operating system installation process.
14. Configure Network settings.



15. Complete the installation and power on the virtual machine.

IBM Spectrum Connect Configuration

1. Login to the spectrum connect virtual machine using SSH.
2. Check that the **zlib** library has been installed.

```
# rpm -qa | grep zlib
```

3. Check that the **bzip2** program has been installed.

```
# rpm -qa | grep bzip2
```

4. Verify that the PostgreSQL package is not installed on your host. The RHEL 7.x server may have a package of PostgreSQL version 8 installed, as part of operation system distribution. This might result in a version conflict of the package installed during the Spectrum Connect deployment.

```
# rpm -qa | grep postgres
```

- A new Linux user name **ibmsc** is created during installation to be used for the Spectrum Connect management operations. You can customize the user ID for **ibmsc** by adding a Linux user (using the `useradd` command in RHEL) before package installation. In this case, create the `/home/ibmsc` directory before starting the installation process.

- Open the 8440 TCP port.

```
# firewall-cmd --permanent --add-port=8440/tcp
# firewall-cmd -reload
```

- If you are using SELinux, allow Spectrum Connect to bind to the 8440 TCP port.

```
#setsebool -P nis_enabled 1
```

- Open the 5672 and 4369 TCP ports.

```
# firewall-cmd --permanent --zone=trusted --add-interface=lo
# firewall-cmd --permanent --zone=trusted --add-port=5672/tcp
# firewall-cmd --permanent --zone=trusted --add-port=4369/tcp
```

- Download the installation package and the `IBM_Spectrum_Connect_Signing_Key_Pub.key` file, used for the package validation. For more information, refer to:

https://www.ibm.com/support/knowledgecenter/SS6JWS_3.4.0/UG/sc_ug_sc_install.html

- Copy the installation package and the public key files to a local folder on the Linux host that will be used as Spectrum Connect server.

- Go to the local folder and then use the `gpg --import IBM_Spectrum_Connect_Signing_Key_Pub.key` file to import the IBM GNU Privacy Guard (GnuPG) public key to validate the installation files. This ensures that the files were received from IBM and were not manipulated in any way by a third party.



Downloading the installation package from a trusted, SSL-protected resource, such as FixCentral, ensures its authenticity and integrity. However, you can mark the key as trusted by entering `gpg --edit-key "IBM Spectrum Connect Signing Key"`, entering the trust command and selecting step 5.

- Extract the installation package file using the following command ('*' represents the build number):

```
# tar -xzf IBM_Spectrum_Connect-3.4.0-*x86_64.tar.gz
```

- Enter the following command to verify the digital signature of the installation files.

```
# gpg --verify ibm_spectrum_connect-3.4.0-xxxx-x86_64.bin.asc
ibm_spectrum_connect-3.4.0-xxxx-x86_64.bin
```

- Go to the extracted directory and then use the `rpm -iv *.rpm` command to run and install all the complementary RPM files. The IBM Storage Provider service starts automatically after the installation. For more information, refer to: https://www.ibm.com/support/knowledgecenter/STWMSg_3.2.0/UG/isis_ug_ch8_check_service.html

- Enter `chmod +x ibm_spectrum_connect-3.4.0-*.bin` to authorize the installation of the product BIN file.

- Enter `./ibm_spectrum_connect-3.4.0-*.bin` to start the installation.

- Review the license agreement, which is displayed after you run the installation file. Enter **1** to accept the license agreement and complete the installation.



To avoid unauthorized access to Spectrum Connect through the GUI, it is strongly recommended to change the default password for the admin user as soon as possible.

Refer to the IBM Spectrum Connect User Guide for more information at:

https://delivery04.dhe.ibm.com/sar/CMA/SDA/07h8a/4/IBM_Spectrum_Connect_3.4.0_UG.pdf

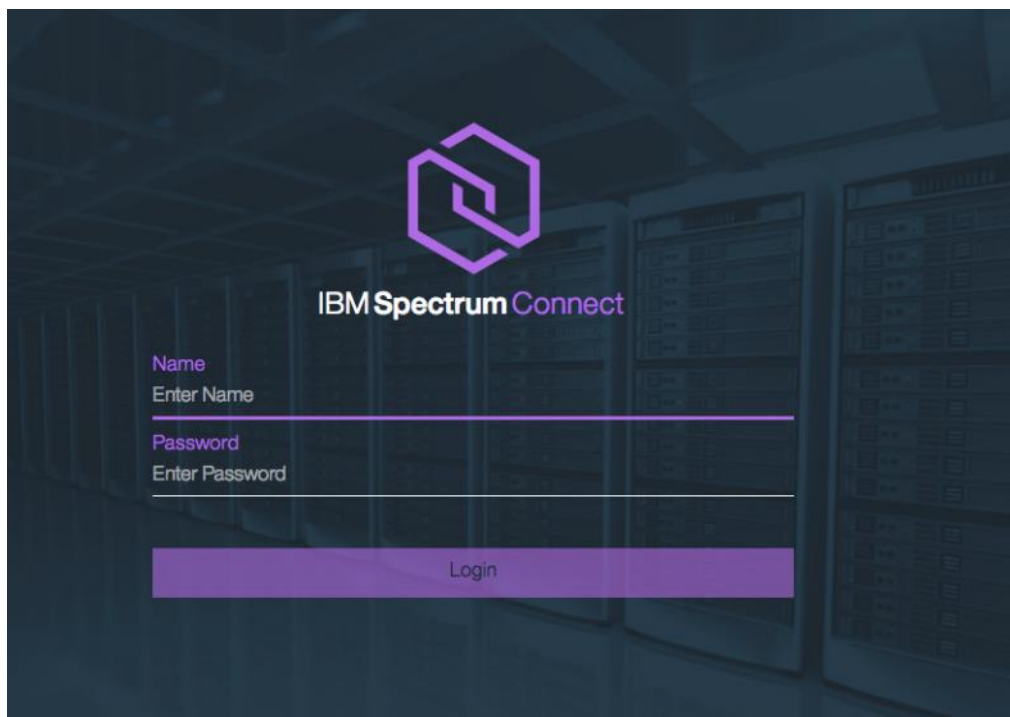
Adding VersaStack Storage System to IBM Spectrum Connect 3.4.0

To add the VersaStack Storage System to IBM Spectrum Connect 3.4.0, complete the following steps:

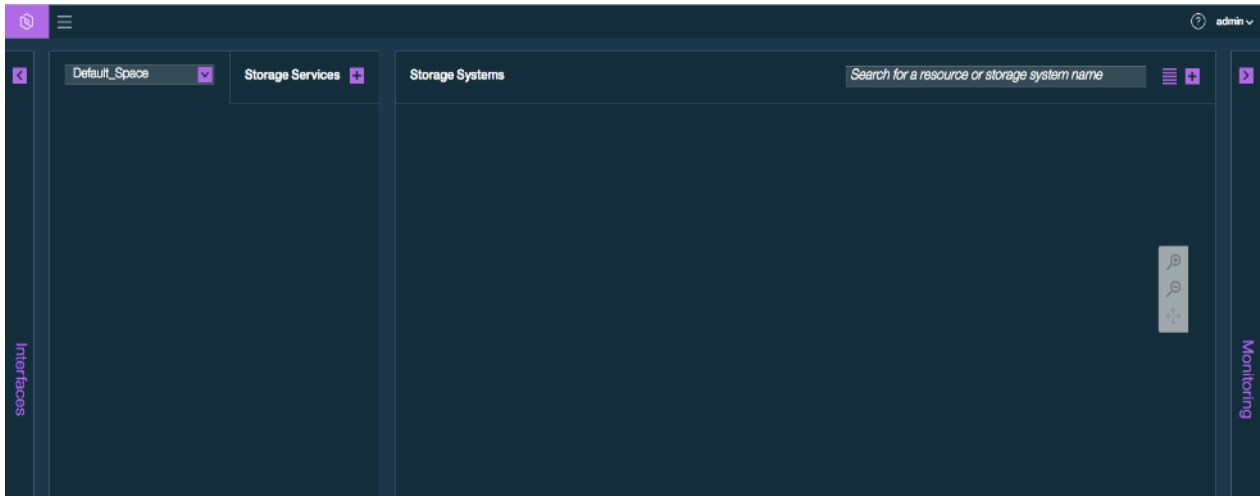
1. Open a browser and enter the web address (URL) of the Linux host on which Spectrum Connect is installed. Use the following format:

`https://[Spectrum Connect IP Address]:8440`

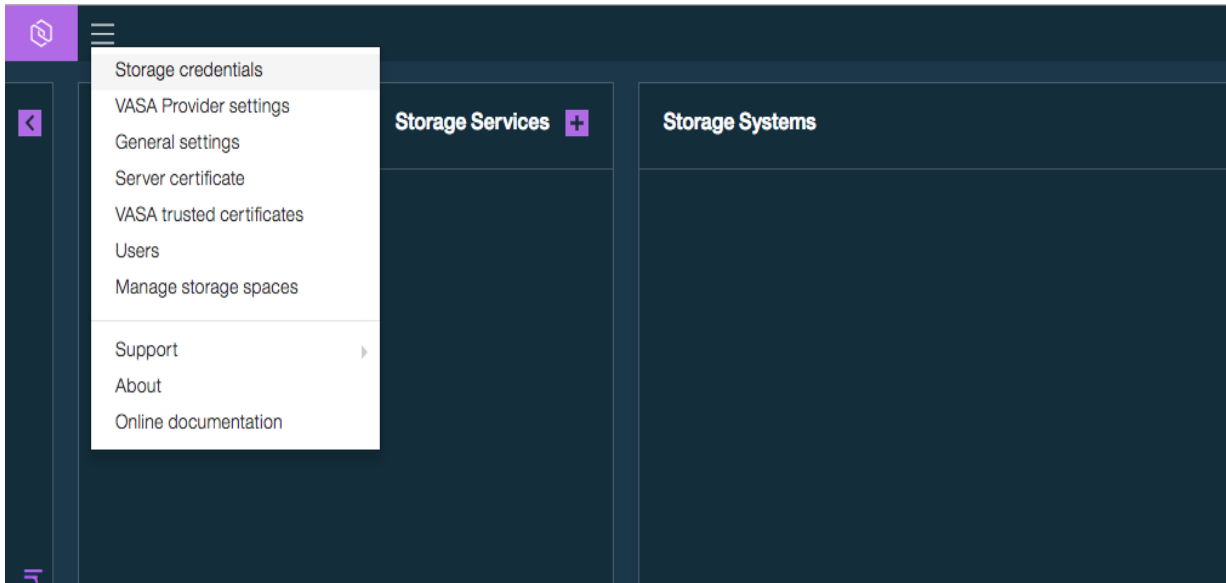
2. Login using the appropriate Spectrum Connect credentials.



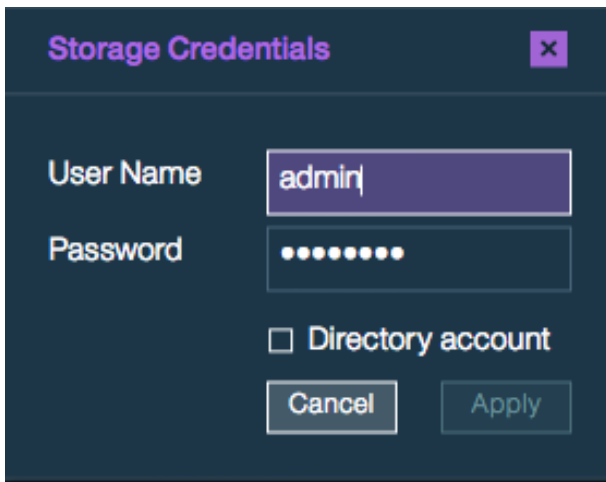
3. After successful login, the Storage Services and Storage Systems panes are displayed.



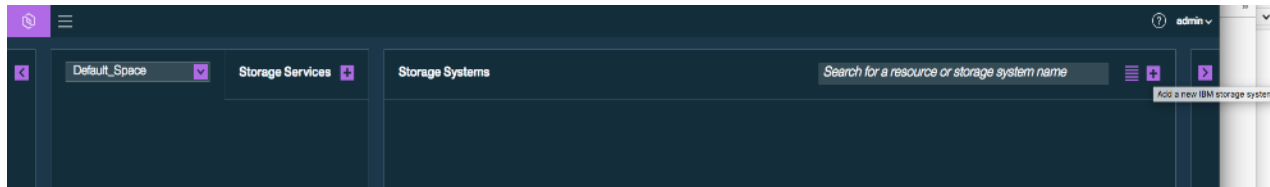
4. Click the **Settings** button and click **Storage credentials**.



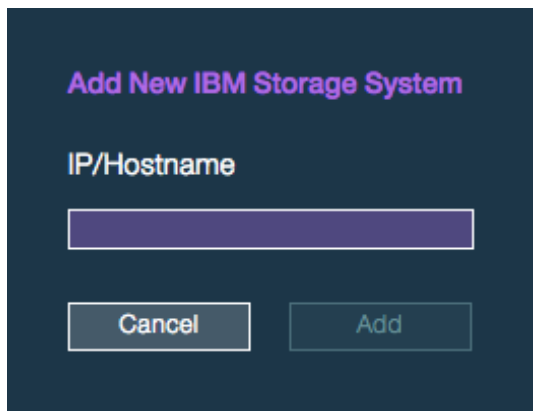
5. Enter the user name and password of the storage admin user who was defined on all your IBM storage systems.



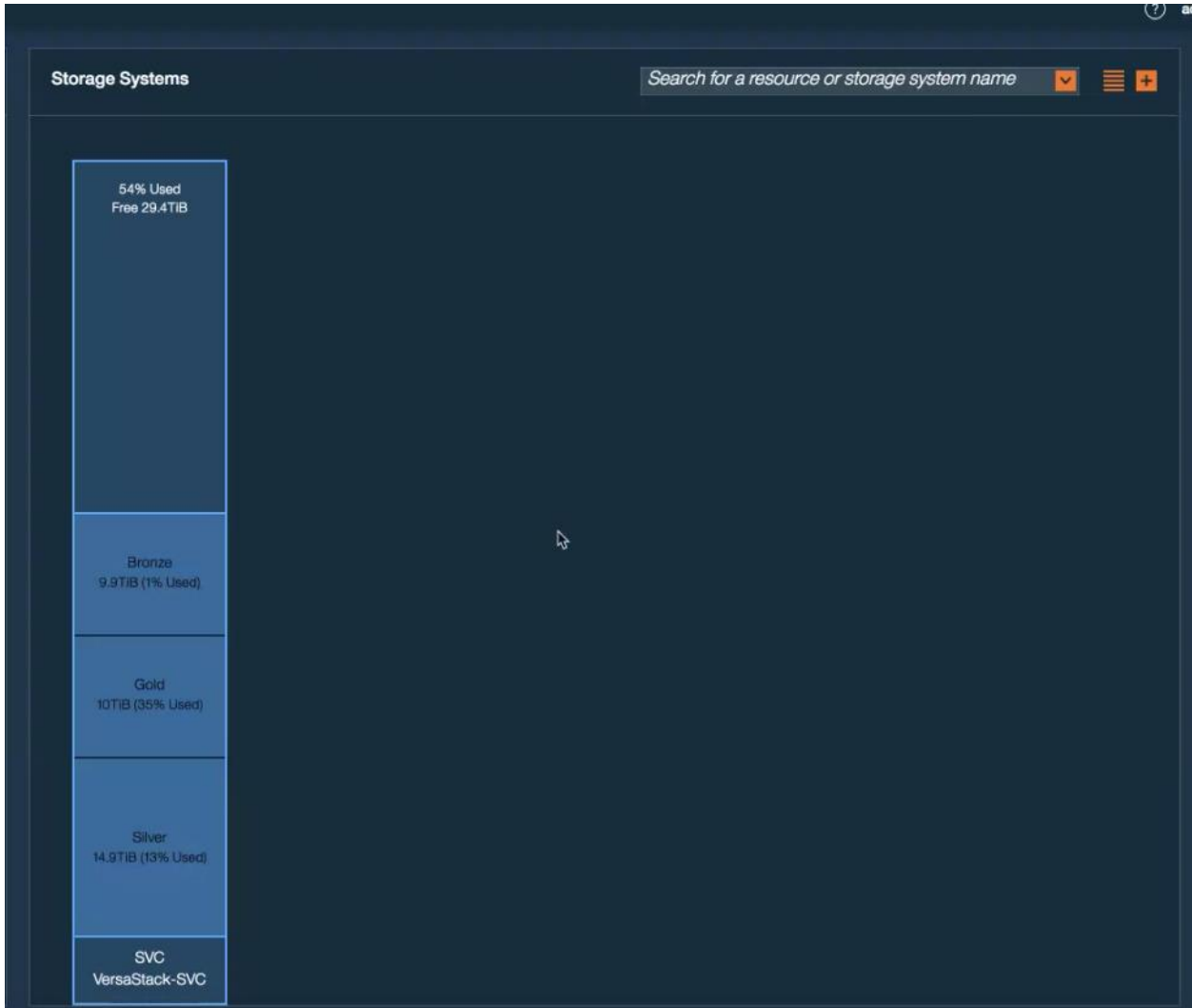
6. If the storage admin user account is defined on a directory server, select the **Directory account** check box. If the storage admin user account is locally-defined on the storage system, clear the check box.
7. Click **Apply**.
8. Click the **Add a new IBM storage system** button on the Storage Systems pane.



9. Enter the management IP address or host name of the array.



10. Click **Add**. If the credentials are correct and the IP connection is established, the storage system is added to the Storage Systems pane.



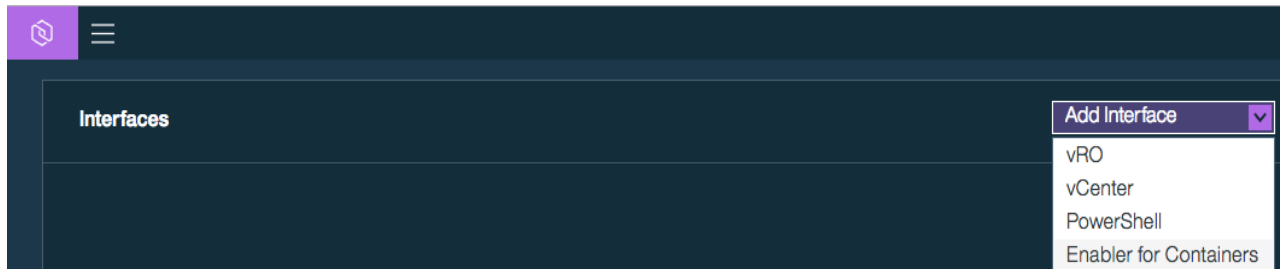
Spectrum Connect fetches information about storage resources on a system every ten minutes by default. You can refresh the storage resource information immediately by right-clicking a system that you want to refresh and then click **Refresh**.

11. Click the Table View button to display the existing storage systems in a table format.

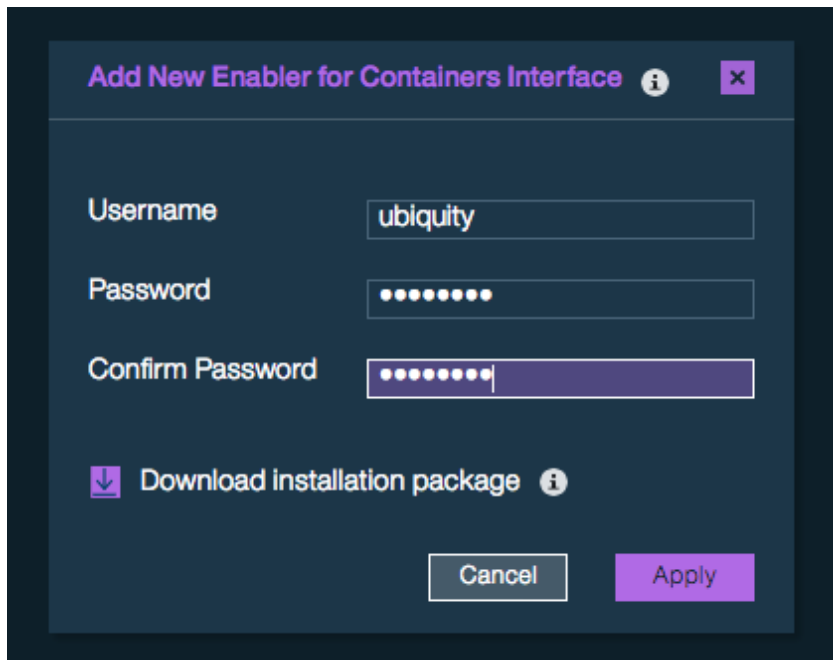
The screenshot shows the 'Storage Resources' table view. At the top, there is a search bar with the text 'Search for a resource or storage system name'. Below the search bar, a table displays the following data:

RESOURCE	STORAGE ARRAY	USAGE	SIZE(GIB)	SERVICE	TYPE
Bronze	VersaStack-SVC	1%	10239	SVC	SVC
Gold	VersaStack-SVC	32%	10240	SVC	SVC
Silver	VersaStack-SVC	13%	15358	SVC	SVC

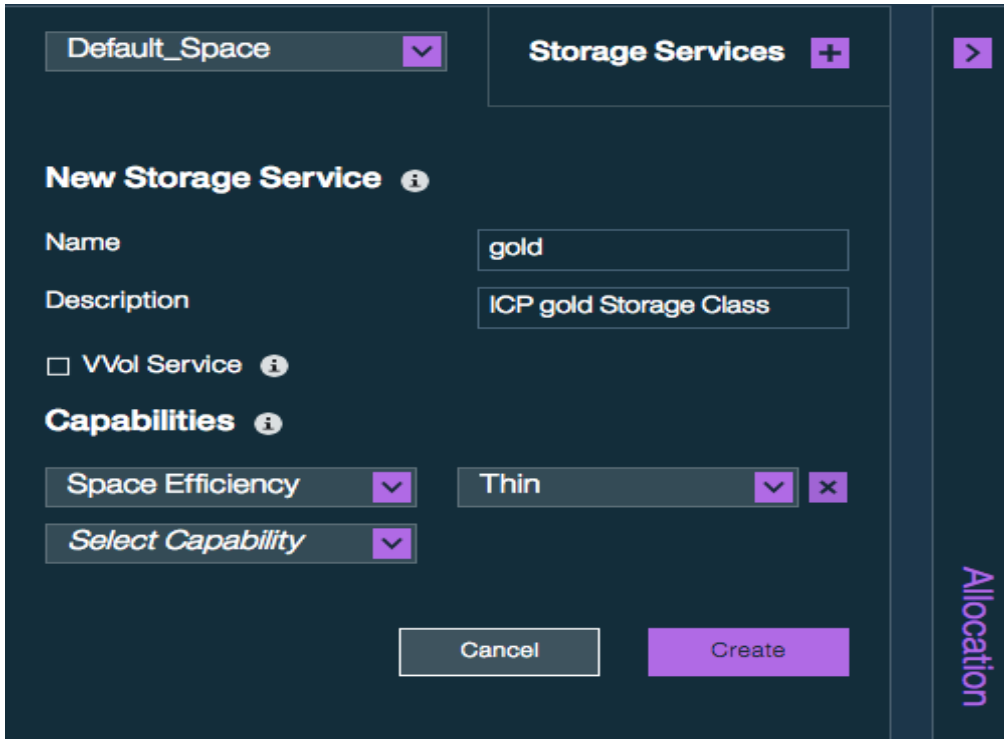
12. Add the Enabler for Containers interface.



13. Provide the user credentials and click **Apply**.



14. Add a new storage service and provide the essential parameters and click **Create**.



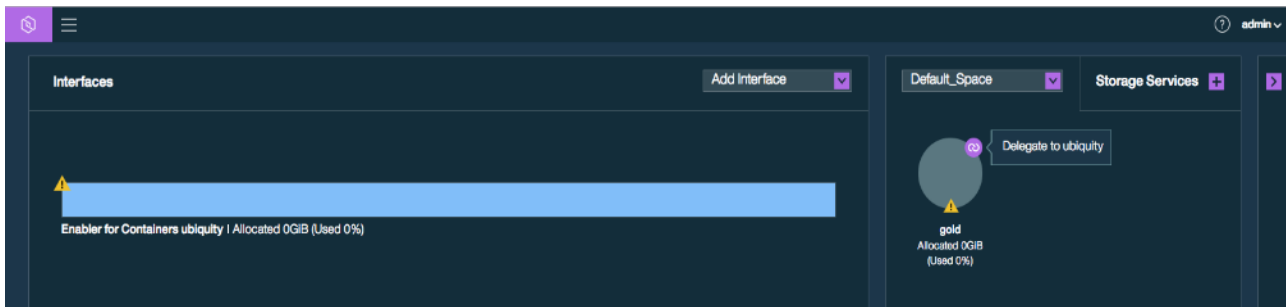
Storage services contain one or more physical storage pools. Ideally storage services can represent distinct types or classes of storage pools depending on an organization’s service level agreement (SLA). Example, gold, silver and bronze representing solid-state drive (SSD), hard disk drive (HDD), and nearline SAS-based storage pools. In addition to specific storage pools type and capacity, a storage service has a set of capabilities, defining the storage quality, such as thin/thick provisioning, compression, encryption, and so on.

15. **Optional:** Assign an appropriate storage resource with the required storage capacity from the storage pool and click **Add**.

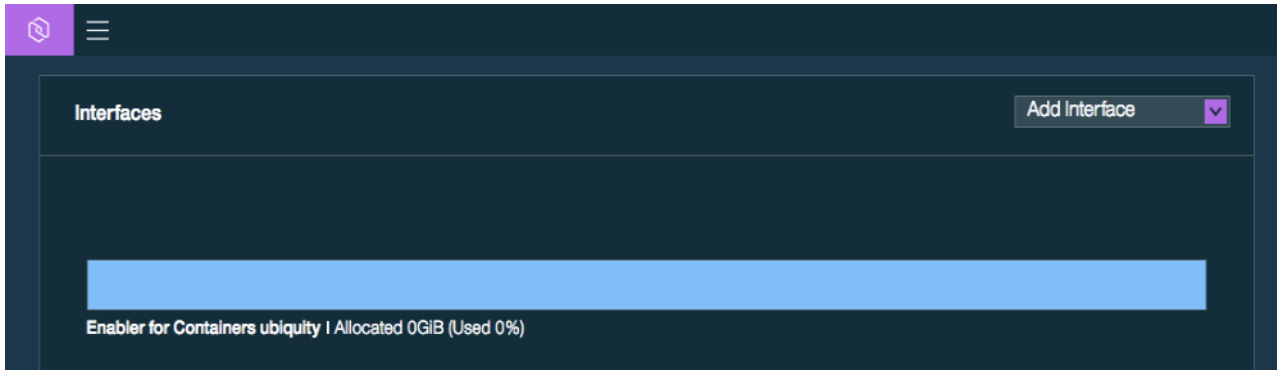


The storage pools with the matching storage capabilities as mentioned above can be added as resources to a storage service.

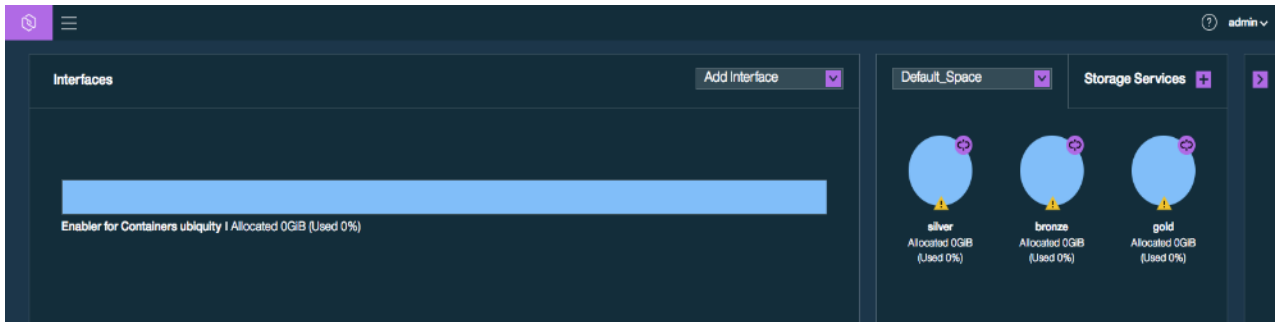
16. Click **Delegate to ubiquity** to delegate newly created storage service.



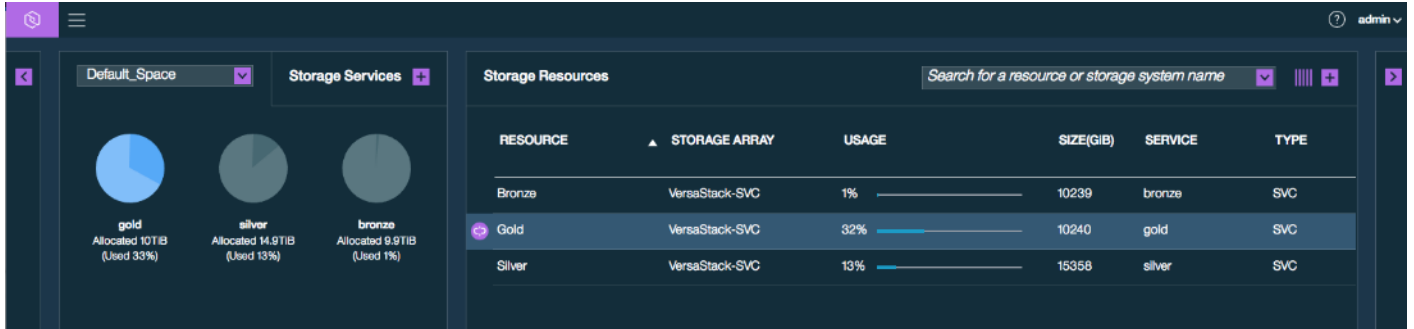
17. Notice that IBM Storage Enabler for Container is successfully getting configured with IBM Spectrum connect 3.4.0.



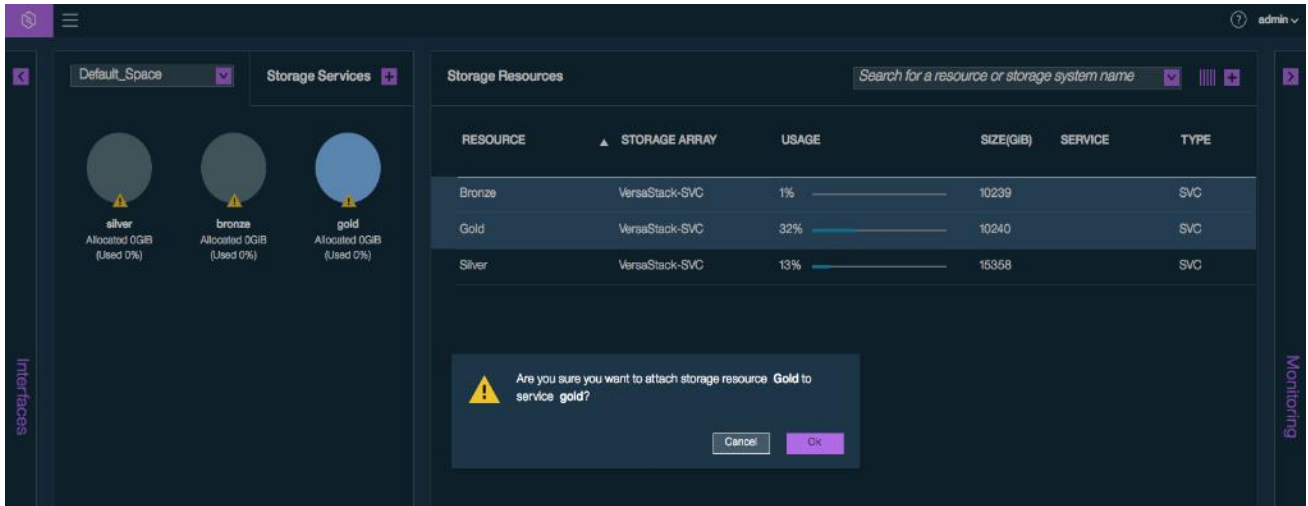
18. Create and designate the other storage services representing Bronze and Silver pools on the IBM SVC storage system.



19. Highlight the storage service created and click on the Storage Resource on the right hand panel under Storage Resources to attach the storage resources to storage service.



20. Attach storage resources to the storage services created from the storage resources pane. Click **Attach** to attach the storage resource.



21. Click **OK** to confirm, repeat the procedure for the other storage services.

ICP Installation

ICP environment is installed on the VMware virtual infrastructure running on VersaStack. Follow the below procedure to complete ICP installation.

IBM Cloud Private Enterprise 2.1.0 HA Environment Installation

ICP environment is installed on five Ubuntu 16.04.3 (64 bit) virtual machines with the hardware specification shown in Table 4 and Table 5 for master, proxy, worker, and management roles for HA nodes.

Table 4 IBM Cloud Private System Configuration

Operating system	Processor	Memory and storage capacity
Ubuntu16.04.3	64-bit dual-core	8 vCPU, 16 GB RAM and 200 GB free disk space

Table 5 IBM Cloud Private System Network Port Configuration

Node	Network Interface	Port Group on VMware Switch
ICP Master1	NIC1	ICP-DPortGroup
ICP Master2	NIC1	ICP-DPortGroup
ICP Master3	NIC1	ICP-DPortGroup
ICP Worker1	NIC1	ICP-DPortGroup

Node	Network Interface	Port Group on VMware Switch
	NIC2	iSCSIBootPG
	NIC3	Vmkernel-iSCSI-B
ICP Worker2	NIC1	ICP-DPortGroup
	NIC2	iSCSIBootPG
	NIC3	Vmkernel-iSCSI-B

For detailed hardware requirements and recommendations for IBM Cloud private 2.1.0 server nodes, refer to the IBM Knowledge Center:

https://www.ibm.com/support/knowledgecenter/SSBS6K_2.1.0/supported_system_config/hardware_reqs.html



For this solution validation, three nodes are configured for master, proxy, and management roles and two nodes are configured for worker nodes. The number of nodes can be increased or decreased based on the application requirements.

Create Virtual Machines for ICP Nodes

Create five Virtual machines with the hardware specifications shown in Table 4 and Table 5 .



Follow the procedure of creating virtual machines detailed in section [Install Virtual Machine for IBM Spectrum Connect](#).

To create virtual machines for ICP nodes, completes the following steps;

1. Power On the Virtual machines.



2. Continue with the installation.
3. When on the network screen window, enter the appropriate management network settings (IP address, hostname, gateway, name server) and complete the installation.
4. Allocate the iSCSI IP addresses on the iSCSI interfaces of worker nodes, use the IPs from iSCSI pool defined for VersaStack.



iSCSI network interfaces and IP addresses are only required on the ICP worker nodes.

Configure ICP Cluster Nodes

To configure ICP cluster nodes, complete the following steps:

1. Configure the /etc/hosts file on each node in your cluster:

```

192.168.163.85 icp-master-1.versapod.local icp-master-1
192.168.163.86 icp-master-2.versapod.local icp-master-2
192.168.163.87 icp-master-3.versapod.local icp-master-3
192.168.163.88 icp-minion-1.versapod.local icp-minion-1
192.168.163.89 icp-minion-2.versapod.local icp-minion-2

192.168.163.92 icp-master-vip.versapod.local icp-master-vip
192.168.163.93 icp-proxy-vip.versapod.local icp-proxy-vip
192.168.163.92 mycluster.icp
    
```

2. On the master nodes, ensure that the vm.max_map_count setting is at least 262144.

- a. Determine the value of the `vm.max_map_count` parameter using the following command:

```
sudo sysctl vm.max_map_count
```

- b. If the `vm.max_map_count` value is not at least 262144, run the following command:

```
sudo sysctl -w vm.max_map_count=262144
```

- c. To ensure that this value is maintained between session and system restarts, as a root user, run the following command:

```
echo "vm.max_map_count=262144" | tee -a /etc/sysctl.conf
```

3. Share the Secure Shell (SSH) keys among cluster nodes.

4. Generate SSH key using the following command

```
ssh-keygen -b 4096 -t rsa -f ~/.ssh/master.id_rsa -N ""
```



For detailed information, refer to:

https://www.ibm.com/support/knowledgecenter/SSBS6K_2.1.0/installing/ssh_keys.html

5. Synchronize the clocks in each node in the cluster. To synchronize your clocks, you can use network time protocol (NTP). For more information about setting up NTP, see the user documentation for your operating system.
6. Make sure that an SSH client is installed on each node.
7. On each node in your cluster, confirm that a supported version of Python is installed. Python versions 2.6 to 2.9.x are supported. Update the version if required using following commands.

```
apt-get install python_setuptools
```

```
apt-get install python-pip
```

8. Install Docker or configure your nodes for the automatic installation of Docker. IBM Cloud Private requires Docker.



For detailed information, refer to:

https://www.ibm.com/support/knowledgecenter/SSBS6K_2.1.0/installing/install_docker.html

```
sudo apt-get update
```

```
sudo apt-get install docker-ce=17.09.1~ce-0~ubuntu
```



Make sure docker is enabled to start on boot.

```
systemctl enable docker ; systemctl start docker ; systemctl status docker
```

9. Make sure the firewall is not enabled on the nodes.
10. Download the installation files for IBM Cloud Private. You must download the correct file or files for the type of nodes in your cluster. You can obtain these files from the IBM Passport Advantage® website at:

https://www-01.ibm.com/software/passportadvantage/pao_customer.html

11. Extract the images and load them into Docker using the following command. Extracting the images might take a few minutes.

```
tar xf ibm-cp-app-mod-x86_64-2.1.0.1.tar.gz -O | sudo docker load
```

12. Create an installation directory to store the IBM Cloud Private configuration files.

```
mkdir /opt/ibm-cp-app-mod-2.1.0.1
```

13. Change to your installation directory.

```
cd /opt/ibm-cp-app-mod-2.1.0.1
```

14. Extract the sample configuration file from the installer image.

```
docker run -v $(pwd):/data -e LICENSE=accept ibmcom/icp-inception:2.1.0.1-ee
```

```
cp -r cluster /data
```

15. A cluster directory is created inside your installation directory. For example, if your installation directory is `/opt/ibm-cp-app-mod-2.1.0.1`, the `/opt/ibm-cp-app-mod-2.1.0.1/cluster` folder is created. The cluster directory contains the following files:

- **config.yaml**: The configuration settings that are used to install IBM Cloud Private to your cluster.
- **hosts**: The definition of the nodes in your cluster.
- **misc/storage_class**: A folder that contains the dynamic storage class definitions for your cluster.
- **ssh_key**: A placeholder file for the SSH private key that is used to communicate with other nodes in the cluster.

16. Modify the `<install_directory>/cluster/hosts` file as shown below.

```
[root@icp-master-1:/opt/ibm-cloud-private-2.1.0.1/cluster# cat hosts
[master]
192.168.163.85
192.168.163.86
192.168.163.87

[worker]
192.168.163.88
192.168.163.89

[proxy]
192.168.163.85
192.168.163.86
192.168.163.87
```



In the solution validation environment master, proxy, and management nodes are configured in the same server nodes.

17. If you use SSH keys to secure your cluster, in the `<installation_directory>/cluster` folder, replace the **ssh_key** file with the private key file that is used to communicate with the other cluster nodes.

```
cp ~/.ssh/master.id_rsa <installation_directory>/cluster/ssh_key
```

18. Move the image files for your cluster to the `/<installation_directory>/cluster/images` folder.

```
mkdir -p cluster/images; \
mv /<path_to_installation_file>/ibm-cloud-private-x86_64- 2.1.0.tar.gz
cluster/images/
```

19. Customize the `cluster.yaml` file.



For more information, refer to:

https://www.ibm.com/support/knowledgecenter/SSBS6K_2.1.0.1/installing/config_yaml.html

20. Refer to the important network configuration settings in the `config.yaml` file as shown below.

```
## Network Settings
network_type: calico
## Network in IPv4 CIDR format
network_cidr: 10.1.0.0/16

## Kubernetes Settings
service_cluster_ip_range: 10.0.0.1/24

calico_ip_autodetection_method: interface=ens160

## High Availability Settings for master nodes
vip_iface: ens160
cluster_vip: 192.168.163.92

## High Availability Settings for Proxy nodes
proxy_vip_iface: ens160
proxy_vip: 192.168.163.93

kubelet_nodename: hostname
```



For more information, refer to:

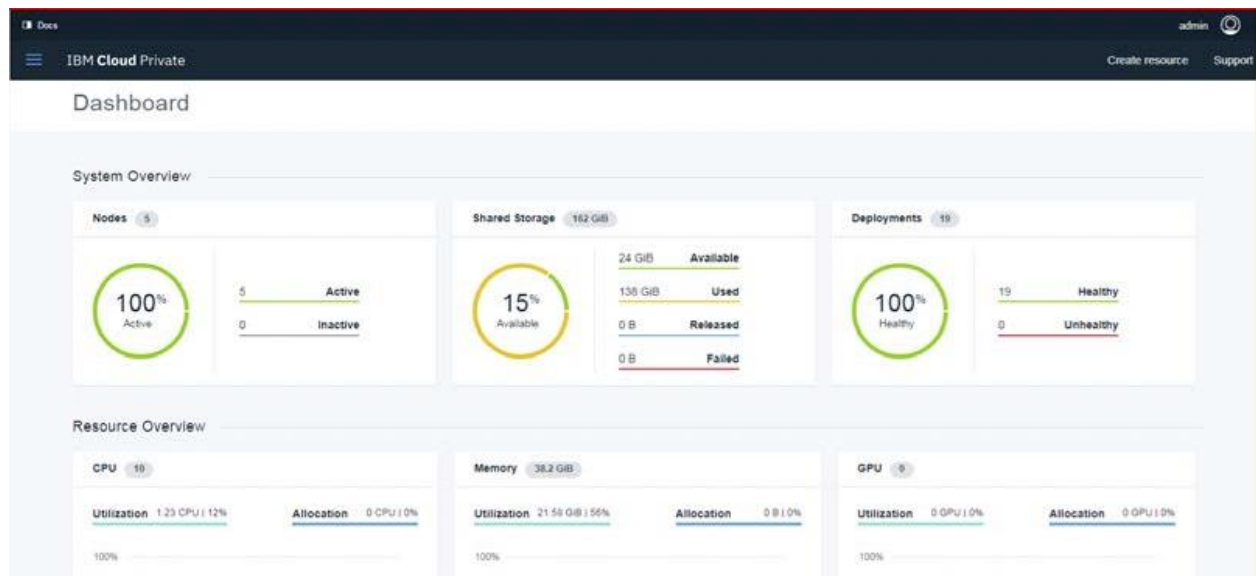
https://www.ibm.com/support/knowledgecenter/SSBS6K_2.1.0.1/installing/config_yaml.html#network_setting

21. Deploy IBM Cloud Private 2.1.0 from the cluster folder installation directory.

```
docker run --net=host -t -e LICENSE=accept -v $(pwd):/installer/cluster \
ibmcom/icp-inception:2.1.0-ee install | tee
```

Access the IBM Cloud Private 2.1.0 Cluster Using Management Console

Obtain the cluster management console URL and default credentials. The URL is `https://master_ip:8443`, where `master_ip` is the IP address of the master node of the IBM Cloud Private cluster.



Install and Set Up kubectl 1.7.3

Kubernetes command-line tool, *kubectl* is essential to deploy and manage applications on Kubernetes. Using *kubectl*, users can inspect Kubernetes cluster resources; create, delete, and update components; and verify new cluster and bring up example apps.

The solution environment *kubectl* is installed on the master node using following command:

```
docker run -e LICENSE=accept --net=host -v /usr/local/bin:/data
ibmcom/kubernetes:v1.7.3-ee cp /kubectl /data
```

Configure kubectl

Before you run commands in the kubectl command-line interface for this cluster, you must configure the client. Access the cluster management console and in the dashboard, click the user (example default user: admin) and follow the instructions to configure client selection.

The IBM Cloud Private 2.1.0.1 account token expires after 12 hrs. In order to overcome this limitation, extract the *service-account-token* and include it in the config file using the following commands.

```
kubectl config set-credentials {IBM Cloud Private Cluster-user} --
token=$(kubectl get secret $(kubectl get secret | grep service-account | awk
'{print $1}') -o yaml | grep token: | awk '{print $2}' | base64 -d)
```



Replace {IBM Cloud Private Cluster-user} with the appropriate user information provided in the configure client selection.

Configuring iSCSI Storage Access for IBM Cloud Private 2.1.0.1 Worker (Minion) Nodes

IBM Cloud Private 2.1.0 worker (minion) nodes are required to be configured to access IBM VersaStack storage system via iSCSI.

This section describes how to configure the IBM Storwize storage system configured with IBM VersaStack and Ubuntu-based IBM Cloud Private 2.1.0 worker (minion) nodes.

To install and configure iSCSI on the worker nodes, complete the following steps:

1. Install iSCSI tools on ICP worker nodes.

```
apt-get install sg3-utils multipath-tools
```

2. Create and edit udev rules file with the following command

```
vi /etc/udev/rules.d/99-ibm-2145.rules
```

3. Add the following line to set SCSI command timeout to 120s (default == 30 or 60) for IBM 2145 devices

```
SUBSYSTEM=="block", ACTION=="add", ENV{ID_VENDOR}=="IBM", ENV{ID_MODEL}=="2145",  
RUN+="/bin/sh -c 'echo 120 >/sys/block/%k/device/timeout'"
```

4. Multipath settings: The following settings as shown are the preferred multipath settings for Ubuntu operating system. Update the `/etc/multipath.conf` file with the below settings. Create the `/etc/multipath.conf` file, if it doesn't exist.

```
devices {  
    device {  
    }  
    vendor "IBM"  
    product "2145"  
    path_grouping_policy "group_by_prio"  
    path_selector "round-robin 0"  
    prio "alua"  
    path_checker "tur"  
    failback "immediate"  
    no_path_retry 5  
    rr_weight uniform  
    rr_min_io_rq "1"  
    dev_loss_tmo 120  
    }  
}
```

5. Update the iSCSI initiator name in the file `/etc/iscsi/initiatorname.iscsi` with worker node `<hostname>` inserted.

For example:

```
InitiatorName=iqn.1993-08.org.debian:01:icp-minion-1-d8a8355be169
```

6. Add host definition on the IBM Storwize storage array by selecting hosts from the GUI console. The host name should exactly match the host name of your Kubernetes worker node name.



In the validation environment, the host name is same as worker node name. Also, provide the iSCSI initiator name shown in the previous step in the file.

Required Fields

Name:

Host connections: Fibre Channel ISCSI

ISCSI port: ⊕ ⊖

Optional Fields

CHAP authentication:

CHAP secret:

Host type:

I/O groups:

Host cluster:

7. For the iSCSI initd script startup, set a session to `automatic` in `/etc/iscsi/iscsid.conf`

```
node.startup = automatic
```

8. Discover the iSCSI target by using the `iscsiadm` CLI.

```
iscsiadm -m discoverydb -t st -p <IP Address configured for iSCSI @ Storwize Storage Array >:3260 --discover
```

For Example:

```
iscsiadm -m discoverydb -t st -p 10.29.161.249:3260 --discover
```

```
iscsiadm -m discoverydb -t st -p 10.29.162.249:3260 --discover
```

```
iscsiadm -m discoverydb -t st -p 10.29.161.250:3260 --discover
```

```
iscsiadm -m discoverydb -t st -p 10.29.162.250:3260 --discover
```

9. Log in to iSCSI target by using iscsiadm CLI.

```
iscsiadm -m node -p <IP Address configured for iSCSI @ Storwize Storage Array>:3260 -login
```

For Example:

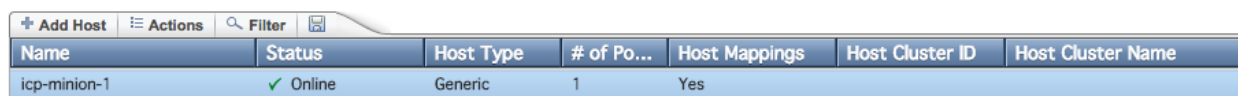
```
iscsiadm -m node -p 10.29.161.249:3260 --login
```

```
iscsiadm -m node -p 10.29.162.249:3260 --login
```

```
iscsiadm -m node -p 10.29.161.250:3260 --login
```

```
iscsiadm -m node -p 10.29.162.250:3260 --login
```

10. Verify the host login status using the IBM Storwize GUI console.



Name	Status	Host Type	# of Po...	Host Mappings	Host Cluster ID	Host Cluster Name
icp-minion-1	✓ Online	Generic	1	Yes		

Installing IBM Storage Enabler for Containers

This section describes the essential details of installing IBM Storage Enabler for Containers using installation scripts. Perform the following steps to install IBM Storage Enabler for Containers:

Download the installer for IBM Storage Enabler for Containers scripts from Fix Central at: <https://www-945.ibm.com/support/fixcentral/>

1. Extract the IBM Storage Enabler for Containers installation script using the **tar** command in the IBM Cloud Private 2.1.0 management node. In the solution validation setup, the management node is **icp-master-1**.
2. The IBM Storage Enabler for Containers script provides the Docker hub automatic installation configuration links of the IBM Storage Enabler for Containers images.
3. Pull the IBM Storage Enabler for Containers from the Docker hub using the following steps:

- a. Log in to the Docker hub with appropriate login credentials.

```
docker login --username=yourhubusername --email=youremail@company.com
```

- b. Pull images from the Docker hub using the following commands:

```
docker pull ibmcom/ibm-storage-enabler-for-containers:1.0.0
docker pull ibmcom/ibm-storage-enabler-for-containers-db:1.0.0
docker pull ibmcom/ibm-storage-dynamic-provisioner-for-kubernetes:1.0.0
docker pull ibmcom//ibm-storage-flex-volume-for-kubernetes:1.0.0
```

- c. Log in to the IBM Cloud Private 2.1.0 registry.

```
docker login mycluster.icp:8500
```



By default, IBM Cloud Private 2.1.0 configures the cluster name as **mycluster.icp**. In case you have configured the custom cluster name, validate `/etc/hosts` for the correct IBM Cloud Private 2.1.0 cluster name.

- d. Tag IBM Storage Enabler for Containers images to the IBM Cloud Private 2.1.0 registry.

```
docker tag ibmcom/ibm-storage-enabler-for-containers:1.0.0 mycluster.icp:8500/
ibm-storage-enabler-for-containers /ibm-storage-enabler-for-containers:1.0.0
```

```
docker tag ibmcom/ibm-storage-enabler-for-containers-db:1.0.0
mycluster:8500/ibm-storage-enabler-for-containers /ibm-storage-enabler-for-
containers-db:1.0.0
```

```
docker tag ibmcom/ibm-storage-dynamic-provisioner-for-kubernetes:1.0.0
mycluster.icp:8500/ibm-storage-enabler-for-containers /ibm-storage-dynamic-
provisioner-for-kubernetes:1.0.0
```

```
docker tag ibmcom/ibm-storage-flex-volume-for-kubernetes:1.0.0
mycluster.icp:8500/ ibm-storage-enabler-for-containers /ibm-storage-flex-volume-
for-kubernetes:1.0.0
```

- e. Push the IBM Storage Enabler for Containers images to the IBM Cloud Private 2.1.0 registry.

```
docker push mycluster.icp:8500/ ibm-storage-enabler-for-containers /ibm-storage-
enabler-for-containers:1.0.0
```

```
docker push mycluster:8500/ ibm-storage-enabler-for-containers/ibm-storage-
enabler-for-containers-db:1.0.0
```

```
docker push mycluster.icp:8500/ ibm-storage-enabler-for-containers/ ibm-storage-
dynamic-provisioner-for-kubernetes:1.0.0
```

```
docker push mycluster.icp:8500/ ibm-storage-enabler-for-containers/ ibm-storage-
flex-volume-for-kubernetes:1.0.0
```

4. Update the **ubiquity_installer.conf** file extracted from step 2 with your environment setup values as shown below

```
root@icp-master-1:~# cat /root/ubiquity_installer.conf
# -----
# Description:
# This is a configuration file for the ubiquity_installer.sh script.
# When run (# ubiquity_installer.sh -s update-ymls -c <conf file>),
# this script replaces the relevant values in all the yaml files of the
installer.
#
# Attention:
# 1. Replace the "VALUE"s placeholder below with the relevant value for your
environment.
# 2. Any change required after running this installer script must be performed
manually in the corresponding *.yaml file itself.
# -----
```



```
# The Docker images of IBM Storage Enabler for Containers.
UBIQUITY_IMAGE=ibmcom/ibm-storage-enabler-for-containers:1.0.0
UBIQUITY_DB_IMAGE=ibmcom/ibm-storage-enabler-for-containers-db:1.0.0
UBIQUITY_K8S_PROVISIONER_IMAGE=ibmcom/ibm-storage-dynamic-provisioner-for-
kubernetes:1.0.0
UBIQUITY_K8S_FLEX_IMAGE=ibmcom/ibm-storage-flex-volume-for-kubernetes:1.0.0

# Parameters in ubiquity-configmap.yml that impact on ubiquity deployment
#-----
# IP or FQDN of SCBE server.
SCBE_MANAGEMENT_IP_VALUE=192.168.163.91

# Communication port of SCBE server.
SCBE_MANAGEMENT_PORT_VALUE=8440

# Default SCBE storage service to be used, if not specified by the storage
class.
SCBE_DEFAULT_SERVICE_VALUE=gold

# A prefix for any new volume created on the storage system.
UBIQUITY_INSTANCE_NAME_VALUE=icp-versapod1

# The fstype of a new volume if not specified by the user in the storage class.
# File system type. Allowed values: ext4 or xfs.
DEFAULT_FSTYPE_VALUE=ext4

# The default volume size (in GB) if not specified by the user when creating a
new volume.
DEFAULT_VOLUME_SIZE_VALUE=1

# Parameter in ubiquity-configmap.yml that impact on "ubiquity-k8s-flex"
daemonset
```

```

#-----
# Allowed values: true or false. Set to true if the nodes have FC connectivity.
SKIP_RESCAN_ISCSI_VALUE=false

# Parameters in ubiquity-configmap.yml that impact on "ubiquity" and "ubiquity-
k8s-provisioner" deployments, And "ubiquity-k8s-flex" daemonset
#-----
# Log level. Allowed values: debug, info, error.
LOG_LEVEL_VALUE=info

# SSL verification mode. Allowed values: require (no validation is required) and
verify-full (user-provided certificates).
SSL_MODE_VALUE=require

# Parameters in scbe-credentials-secret.yml that impact ubiquity and ubiquity-
k8s-provisioner deployments, And ubiquity-k8s-flex daemonset
#-----
# Username and password defined for IBM Storage Enabler for Containers interface
in SCBE.
SCBE_USERNAME_VALUE=ubiquity
SCBE_PASSWORD_VALUE=<password>

# Parameters in ubiquity-db-credentials-secret.yml that impact ubiquity and
ubiquity-db deployments
#-----
# Username and password for the deployment of ubiquity-db database. Note : Do
not use the "postgres" username, because it already exists.
UBIQUITY_DB_USERNAME_VALUE=ubiquity
UBIQUITY_DB_PASSWORD_VALUE=ubiquity

```

```

# Parameters to create the first Storage Class that also be used by ubiquity for
ibm-ubiquity-db PVC.

# The parameters in the following files: yamls/storage-class.yml, ubiquityt-db-
pvc.yml, sanity_yamls/sanity-pvc.yml

#-----
# Storage Class name

STORAGE_CLASS_NAME_VALUE=gold

# Storage Class profile parameter should point to the SCBE storage service name

STORAGE_CLASS_PROFILE_VALUE=gold

# Storage Class file-system type, Allowed values: ext4 or xfs.

STORAGE_CLASS_FSTYPE_VALUE=ext4

```

5. Refer to the detailed IBM Storage Enabler for Containers or IBM Ubiquity documentation in case you need to configure the `SSL_MODE=verify-full` option.
6. Run the IBM Storage Enabler for Containers installation script (`ubiquity_installer.sh`) to update the (`.yml`) configuration files from the `ubiquity.config` setting.

```
./ubiquity_installer.sh -s update-ymls -c ubiquity_installer.conf
```

7. Run the first phase of the IBM Storage Enabler for Containers installation script (`ubiquity_installer.sh`).

```
./ubiquity_installer.sh -s install -k <k8s-config>
```



The `-k <k8s-config>` file is very important. It is the Kubernetes API server configuration file that allow the provisioner to communicate with Kubernetes and listen to persistent volume claims.

8. Usually this file is located in `~/.kube/config`. Validate that the token inside this file is not expired. If already expired, the IBM Storage Enabler for Containers provisioner will not work as expected,
9. During the solution validation, the installation script was run as follows:

```
./ubiquity_installer.sh -s install -k ~/.kube/config
```

For a sample of a successful run of the first phase of the IBM Storage Enabler for Containers installation status, see below:

```

....
configmap "ubiquity-k8s-flex.conf" created
daemonset "ubiquity-k8s-flex" created

"IBM Storage Enabler for Containers" installation finished,
but its NOT ready yet.

  You must do : (1) Manually restart kubelet service on all
minions to reload the new flex driver
                (2) Deploy ubiquity-db by      $>
./ubiquity_installer.sh -s create-ubiquity-db -n ubiquity
                Note : View ubiquity status by $>
./ubiquity_cli.sh -a status -n ubiquity

```

10. On all IBM Cloud Private 2.1.0 master nodes, update the controller manager static POD as shown in the following steps:

a. Copy `/etc/cfc/pods/master.json` to a temp directory.

```
cp /etc/cfc/pods/master.json /var/tmp/master.json
```

11. Create a backup copy of the `master.json`.

```
cp /var/tmp/master.json /var/tmp/master.json.bkup
```

12. Using an editor (`vi` or `nano`), modify the spec `{"name": "controller-manager", "volumeMounts": [...] }` section in `master.json` as shown below.

```
"spec":{
  "hostNetwork": true,
  "containers":[
    {
      "name": "controller-manager",
      ...
      "volumeMounts": [
        {
          "name": "data",
          "mountPath": "/etc/cfc/conf"
        },
        {
          "name": "audit",
          "mountPath": "/var/lib/icp/audit"
        },
        {
          "name": "flexvolume-dir",
          "mountPath": "/usr/libexec/kubernetes/kubelet-plugins/volume/exec"
        }
      ]
    },
  ],
}
```

13. Update "spec": {"volumes": [...] }, as shown in the above sample file, using an editor of your choice.

```

"spec":{
  ...
  "containers":[
    {
      "name": "controller-manager",
      ...
      "volumeMounts": [
        ...
        {
          "name": "flexvolume-dir",
          "mountPath": "/usr/libexec/kubernetes/kubelet-plugins/volume/exec"
        }
      ]
    }
    ...
  "volumes": [
    {
      "name": "data",
      "hostPath": {
        "path": "/etc/cfc/conf"
      }
    },
    {
      "name": "audit",
      "hostPath": {
        "path": "/var/lib/icp/audit"
      }
    },
    {
      "name": "flexvolume-dir",
      "hostPath": {
        "path": "/usr/libexec/kubernetes/kubelet-plugins/volume/exec"
      }
    }
  ]
},

```

14. Copy `/var/tmp/master.json` to `/etc/cfc/pods/master.json`.

```
cp /var/tmp/master.json /etc/cfc/pods/master.json
```



Kubelet should automatically re-create the `k8s-master-<IP Address of master>` with the new parameters, but if it does not, then a `systemctl restart kubelet` should. In a HA IBM Cloud Private environment, wait for the system to settle and then repeat these steps on rest of the other master nodes.

15. Next, restart the kubelet service on all IBM Cloud Private worker and master nodes using the `systemctl` command.

```
systemctl restart kubelet
```

16. Run the second phase of the installation script that will create the ubiquity DB deployment.

```
./ubiquity_installer.sh -s create-ubiquity-db -n ubiquity
```

17. Verify the IBM Storage Enabler for Containers installation status.

```
./ubiquity_cli.sh -a status
```

```

NAME          DATA    AGE
cm/k8s-config      1        7h
cm/ubiquity-configmap 10        7h

NAME          CAPACITY  ACCESSMODES  RECLAIMPOLICY  STATUS  CLAIM                STORAGECLASS  REASON  AGE
pv/ibm-ubiquity-db 20Gi      RWO          Delete         Bound   ubiquity/ibm-ubiquity-db  ibmc-block-gold  7h

NAME          STATUS    VOLUME          CAPACITY  ACCESSMODES  STORAGECLASS  AGE
pvc/ibm-ubiquity-db Bound     ibm-ubiquity-db 20Gi      RWO          ibmc-block-gold 7h

NAME          CLUSTER-IP  EXTERNAL-IP  PORT(S)  AGE
svc/ubiquity  10.0.0.95   <none>       9999/TCP 7h
svc/ubiquity-db 10.0.0.93   <none>       5432/TCP 7h

NAME          DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE-SELECTOR  AGE
ds/ubiquity-k8s-flex 5         5        5      5            5          <none>         7h

NAME          DESIRED  CURRENT  UP-TO-DATE  AVAILABLE  AGE
deploy/ubiquity 1         1        1            1          7h
deploy/ubiquity-db 1         1        1            1          7h
deploy/ubiquity-k8s-provisioner 1         1        1            1          7h

kubectrl get --namespace ubiquity pod | egrep "^ubiquity|^NAME"
-----
NAME          READY  STATUS  RESTARTS  AGE
ubiquity-4072355995-r6hg4 1/1    Running  0          7h
ubiquity-db-247393488-hbw10 1/1    Running  0          7h
ubiquity-k8s-flex-2mm37 1/1    Running  0          7h
ubiquity-k8s-flex-3pq87 1/1    Running  0          7h
ubiquity-k8s-flex-dznt2 1/1    Running  0          7h
ubiquity-k8s-flex-wmb4v 1/1    Running  0          7h
ubiquity-k8s-flex-xv8jq 1/1    Running  0          7h
ubiquity-k8s-provisioner-1266080725-jkv2s 1/1    Running  0          7h

```

Create Storage Class Definitions

You can define additional storage classes using the (YAML) configuration files if needed. These storage classes can be used for persistent volume creation during the application containers deployment. Refer to the samples from the gold.yaml, silver.yaml, and bronze.yaml configuration files for storage class definitions provided below:

- a. gold.yaml file

```

kind: StorageClass
apiVersion: storage.k8s.io/v1beta1

metadata:
  name: "gold"
  labels:
    product: ibm-storage-enabler-for-containers
  annotations:
    storageclass.beta.kubernetes.io/is-default-class: "true"
provisioner: "ubiquity/flex"
parameters:
  profile: "gold"
  fstype: "ext4"

```

```
backend: "scbe"
```



Use the configuration setting as in the following annotation for the default storage class:

```
storageclass.beta.kubernetes.io/is-default-class: "true"
```



Also, `ubiquity_install.conf` uses `STORAGE_CLASS_NAME_VALUE`, and `STORAGE_CLASS_PROFILE_VALUE` parameters to configure the default storage class to use. Refer to the sample file above for more information.

b. silver.yaml file

```
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: "silver"
  labels:
    product: ibm-storage-enabler-for-containers
provisioner: "ubiquity/flex"
parameters:
  profile: "ibmc-silver"
  fstype: "ext4"
  backend: "scbe"
```

c. bronze.yaml file

```
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: "bronze"
  labels:
    product: ibm-storage-enabler-for-containers
provisioner: "ubiquity/flex"
parameters:
  profile: "ibmc-block-bronze"
  fstype: "ext4"
```



```
backend: "scbe"
```

18. Use `kubectl` to create storage classes (as follows) from the IBM Cloud Private management node.

```
kubectl create -f silver.yaml
```

```
kubectl create -f bronze.yaml
```

Add New ICP Worker Nodes

A cluster can contain any number of worker nodes, but a minimum of one worker node is required. Worker nodes provide a containerized environment for running tasks. As demands increase, more worker nodes can easily be added to your ICP cluster to improve performance and efficiency.



A bare metal Cisco UCS server can be added as a new worker node, if adding a bare metal server. Create a new UCS service profile and install the Ubuntu operating system.

1. Install a new virtual machine based on the hardware recommendation for worker node and also considering the resources required to support your application needs.
2. Complete the iSCSI configuration on the new worker node as detailed in the above section "Configuring iSCSI for IBM Cloud Private 2.1.0.1 worker (minion) nodes".
3. Add host definition for the new worker node on the IBM SVC system from the GUI console. Make sure that the new worker node virtual machine is logging in to the storage system.
4. When the virtual machine is configured and prepared to be added to the cluster, run the following command to add the new worker node in to a specific cluster.

```
docker run -e LICENSE=accept --net=host \
-v "$(pwd)":/installer/cluster \
ibmcom/icp-inception:2.1.0.1-ee worker -l \
ip_address_workernode1,ip_address_workernode2
```



The kubernetes flexvolume driver automatically gets installed on the new worker node.

For more information about adding worker and other nodes to clusters, please visit the ICP InfoCenter:

https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0.1/installing/add_node.html

Remove ICP Worker Nodes

As demand decreases, worker nodes that are no longer needed can be removed from your cluster with the following command:

```
docker run -e LICENSE=accept --net=host \
-v "$(pwd)":/installer/cluster \
ibmcom/icp-inception:2.1.0.1-ee uninstall -l \
ip_address_clusternode1,ip_address_clusternode2
```

For additional information about removing nodes from clusters, please visit the ICP InfoCenter:
https://www.ibm.com/support/knowledgecenter/en/SSBS6K_2.1.0.1/installing/remove_node.html

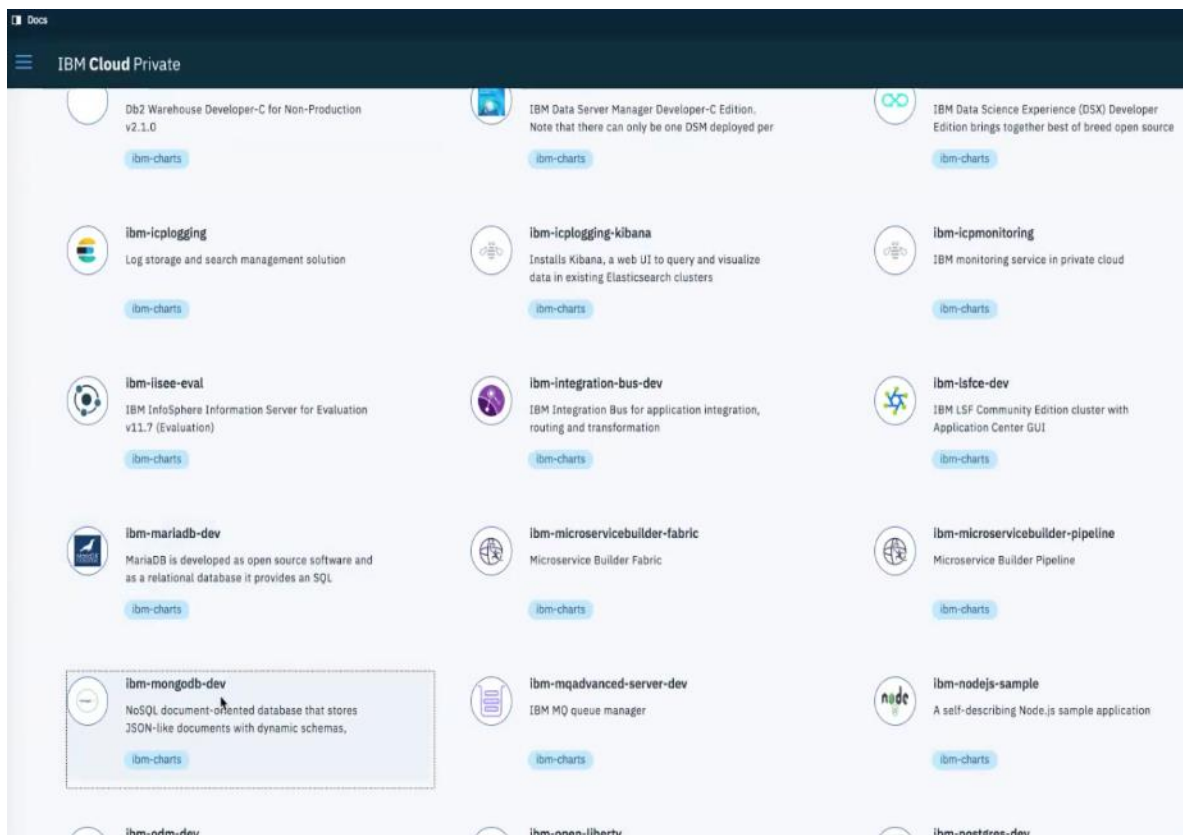
Applications Deployment on VersaStack with ICP

A number of containerized applications can be deployed that are available in the ICP catalog. The following procedure provides a few sample applications (MongoDB and IBM DB2 Warehouse) deployment.

MongoDB

To deploy a MongoDB instance using IBM Cloud Private catalog (Helm charts) and properly provisioning persistent storage to the MongoDB instance, complete the following steps:

1. In the IBM Cloud Private 2.1.0 management web UI, select **catalog** and from the Helm charts and click `ibm-mongodb-dev`.



2. Click **Configure**.

resources.requests.cpu Container CPU requested 100m

resources.requests.memory Container Memory requested 258Mi

dataVolume.name Name of the PVC to be created datavolume

dataVolume.storageClassName Storage class of backing PVC si1 (uses alpha storage class annotation)

dataVolume.existingClaimName Name of the Existing Claim to be used si1

dataVolume.size Size of data volume 1Gi

The above parameters map to the env variables defined in values.yaml.

Specify each parameter using the `--set key=value[,key=value]` argument to `helm install`. For example,

```
$ helm install --name my-release \
--set database.password=mypassword,database.user=myuser \
stable/ibm-mongodb-dev
```

The above command sets the MongoDB `myuser` account password to `mypassword`.

Alternatively, a YAML file that specifies the values for the parameters can be provided while installing the chart. For example,

```
$ helm install --name my-release -f values.yaml stable/ibm-mongodb-dev
```

Persistence

The MongoDB image stores the MongoDB data at the `/data/db` path of the container.

The chart mounts a `Persistent Volume` volume at this location. The volume is created using dynamic volume provisioning.

3. Provide a proper release name in the Release name field, select the I have read and agreed to the license agreements check box.

Configuration

NoSQL document-oriented database that stores JSON-like documents with dynamic schemas, simplifying the integration of data in content-driven applications. Edit these parameters for configuration

Release name

Target namespace

I have read and agreed to the [license agreements](#)

image

repository

tag

imagePullPolicy

4. Specify `useDynamicProvisioning` as **true**.
5. Continue in the same configuration web UI and in the **dataVolume** section provide the storage class name as exactly defined in one of the appropriate `storageClass` YAML files. For more information, refer to the **"Storage Class Definitions"** section.

persistence

enabled	<input type="text" value="true"/>	useDynamicProvisioning	<input type="text" value="true"/>
----------------	-----------------------------------	-------------------------------	-----------------------------------

dataVolume

name	<input type="text" value="datavolume"/>	existingClaimName	<input type="text" value="Enter value"/>
storageClassName	<input type="text" value="gold"/>	size	<input type="text" value="20Gi"/>



Provide the appropriate storage class based on the application needs.

- In the **SERVICE** section of the same web UI, provide any additional MongoDB configuration parameters.

resources

requests.memory	<input type="text" value="256Mi"/>	requests.cpu	<input type="text" value="100m"/>
limits.cpu	<input type="text" value="2"/>	limits.memory	<input type="text" value="4Gi"/>

service

name	<input type="text" value="ibm-mongodb-dev"/>	type	<input type="text" value="NodePort"/>
port	<input type="text" value="27017"/>		

- Click **Install**.

Configure ibm-mongodb-dev V 1.0.2

Configuration

NoSQL document-oriented database that stores JSON-like documents with dynamic schemas, simplifying the integration of data in content-driven applications. Edit these parameters for configuration

Release name ? **Target namespace** ?

I have read and agreed to the [license agreements](#)

image

repository **tag**

imagePullPolicy

- Validate the deployment by clicking Workloads >Helm releases in IBM Cloud Private.

Helm releases

Search items

20 Items per page | 1-1 of 1 items 1 of 1 pages < 1 >

NAME-	NAMESPACE-	STATUS-	UPDATED-	CHART NAME-	CHART VERSION-	ACTION
my-first-mongodb	default	DEPLOYED	Feb 22nd 2018	ibm-mongodb-dev	1.0.2	⋮

- Verify **PERSISTENTVOLUMECLAIM** by clicking the name of the MongoDB instance deployed and review the notes on how to access the MongoDB instance.

PersistentVolumeClaim / my-first-mongodb-ibm-mongodb-dev-datavolume /
my-first-mongodb-ibm-mongodb-dev-datavolume

[Overview](#) [Events](#)

PersistentVolumeClaim details

Type	Detail
Name	my-first-mongodb-ibm-mongodb-dev-datavolume
Namespace	default
Labels	app=my-first-mongodb-ibm-mongodb-dev,chart=ibm-mongodb-dev-1.0.2,heritage=Tiller,release=my-first-mongodb
Resource requests	20Gi
Access modes	RWO
Status	Bound
PersistentVolume	pvc-e0e93754-1805-11e8-84a3-0050569f2a69
Creation time	Feb 22nd 2018 at 2:23 PM

10. Verify the IBM SVC management web UI to validate the newly created persistent volume.
11. Click the provided URL for MongoDB service.

Services / my-first-mongodb-ibm-mongodb-dev /

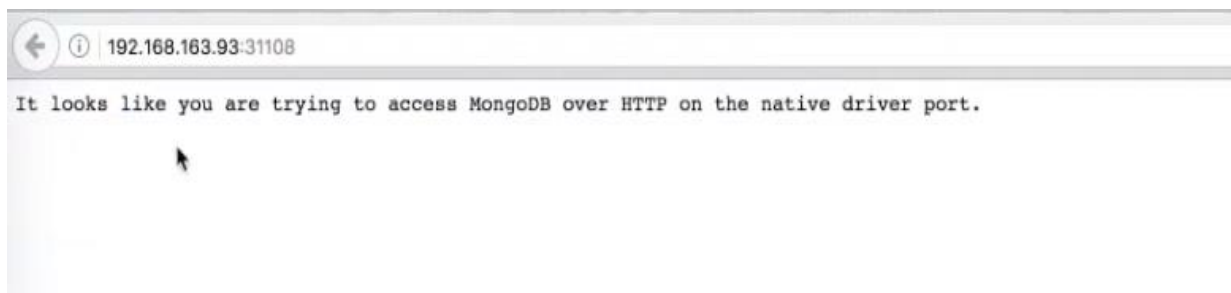
my-first-mongodb-ibm-mongodb-dev

Overview

Service details

Type	Detail
Name	my-first-mongodb-ibm-mongodb-dev
Namespace	default
Creation time	Feb 22nd 2018 at 2:23 PM
Type	NodePort
Labels	app=my-first-mongodb-ibm-mongodb-dev,chart=ibm-mongodb-dev-1.0.2,he
Selector	app=my-first-mongodb-ibm-mongodb-dev
IP	10.0.0.198
Port	ibm-mongodb-dev 27017/TCP
Node port	ibm-mongodb-dev 31108/TCP
Session affinity	None

12. Make sure the service installed successfully.

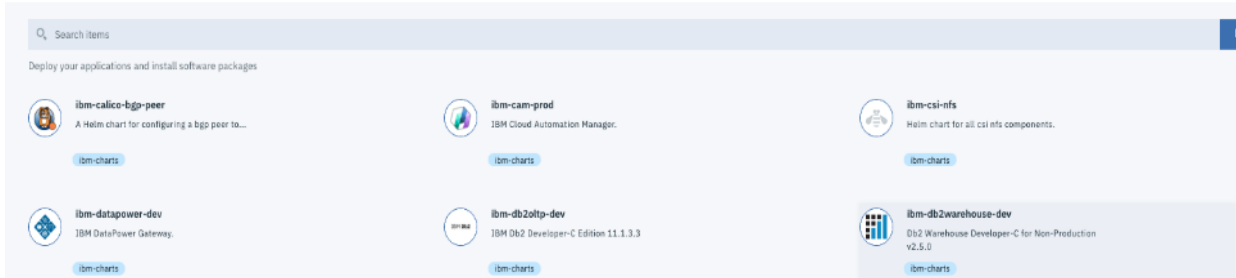


IBM DB2 Warehouse

This section provides detail steps for deploying IBM DB2 Warehouse developer instance using IBM Cloud Private catalog (Helm charts) and properly provisioning persistent storage to the DB2 Warehouse instance.

1. In the IBM Cloud Private 2.1.0 management web UI, select catalog and from the Helm charts and click `ibm-db2warehouse-dev`.

Catalog



2. Click **Configure**.

ibm-db2warehouse-dev V 2.0.0

The screenshot shows the configuration page for the "ibm-db2warehouse-dev" Helm chart. On the left sidebar, there is a "View Licenses" link, the version "2.0.0", the publication date "8th Mar 2018", and the type "Helm Chart". The main content area has a title "IBM Db2 Warehouse" and an introduction. A "Prerequisites" section contains a code block for a PersistentVolume configuration:

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv0001
  labels:
    *entireLabel: labelvalue
spec:
  accessModes:
    - ReadWriteOnce
  capacity:
    storage: 25Gi
  storageClassName:
  volumeMode: Filesystem
  path: /mnt/clusterfs/
```

3. Provide a proper release name in the Release name field, select the I have read and agreed to the license agreements check box.

Configure ibm-db2warehouse-dev V 2.0.0

4. Enter the password for the default admin user in the password field.

5. Specify useDynamicProvisioning as **true**.
6. Continue in the same configuration web UI and in the "Storage Class Name" section provide the storage class name as exactly defined in one of the appropriate storageClass YAML files. For more information, refer to the "**Storage Class Definitions**" section.



Provide the appropriate storage class based on the application needs.

7. In the "Resource Configuration" section of the same web UI, change Memory and CPU if required optionally and click **Install**.

8. Validate the deployment by clicking Workloads >Helm releases in IBM Cloud Private.

Helm Releases

NAME	NAMESPACE	STATUS	CHART NAME	CURRENT VERSION	AVAILABLE VERSION	UPDATED	ACTION
warehouse1	default	Deployed	ibm-db2warehouse-dev	2.0.0	Up To Date	Jun 1st 2018 11:44am	Launch

- Verify PERSISTENTVOLUMECLAIM by clicking the name of the DB2 Warehouse instance deployed and review the notes on how to access the DB2 Warehouse instance.

PersistentVolumeClaim / warehouse1-ibm-db2warehouse-dev-pvc1 /
 warehouse1-ibm-db2warehouse-dev-pvc1
 Overview Events

PersistentVolumeClaim details

Type	Detail
Name	warehouse1-ibm-db2warehouse-dev-pvc1
Namespace	default
Labels	app=warehouse1-ibm-db2warehouse-dev,chart=ibm-db2warehouse-dev-2.0.0,heritage=Tiller,release=warehouse1
Resource requests	25Gi
Access modes	RWO
Status	Bound
PersistentVolume	pvc-a9ba9fa1-65b2-11e8-8cb4-0050569f02cc
Created	1 hour ago

- Verify the IBM SVC management web UI to validate the newly created persistent volume.

Name	State	Synchronized	Volume Group	UID	Host Mappings	C
infra_datastore_1	Online			600507680C8281138800000000...	Yes	
infra_datastore_2	Online			600507680C8281138800000000...	Yes	
infra_ISCSI_datastore_1	Online			600507680C8281138800000000...	Yes	
infra_swap	Online			600507680C8281138800000000...	Yes	
test1	Online			600507680C8281138800000000...	No	
u_icp_ibm-ubiquity-db	Online (formatting)			600507680C8281138800000000...	Yes	
u_icp_pvc-00f2fdb4-65b1-11e8-8cb4-0050569f02cc	Online (formatting)			600507680C8281138800000000...	Yes	
u_icp_pvc-a9ba9fa1-65b2-11e8-8cb4-0050569f02cc	Online (formatting)			600507680C8281138800000000...	Yes	

- Click the provided URL for Db2 Warehouse service.

Services / warehouse1-ibm-db2warehouse-dev /

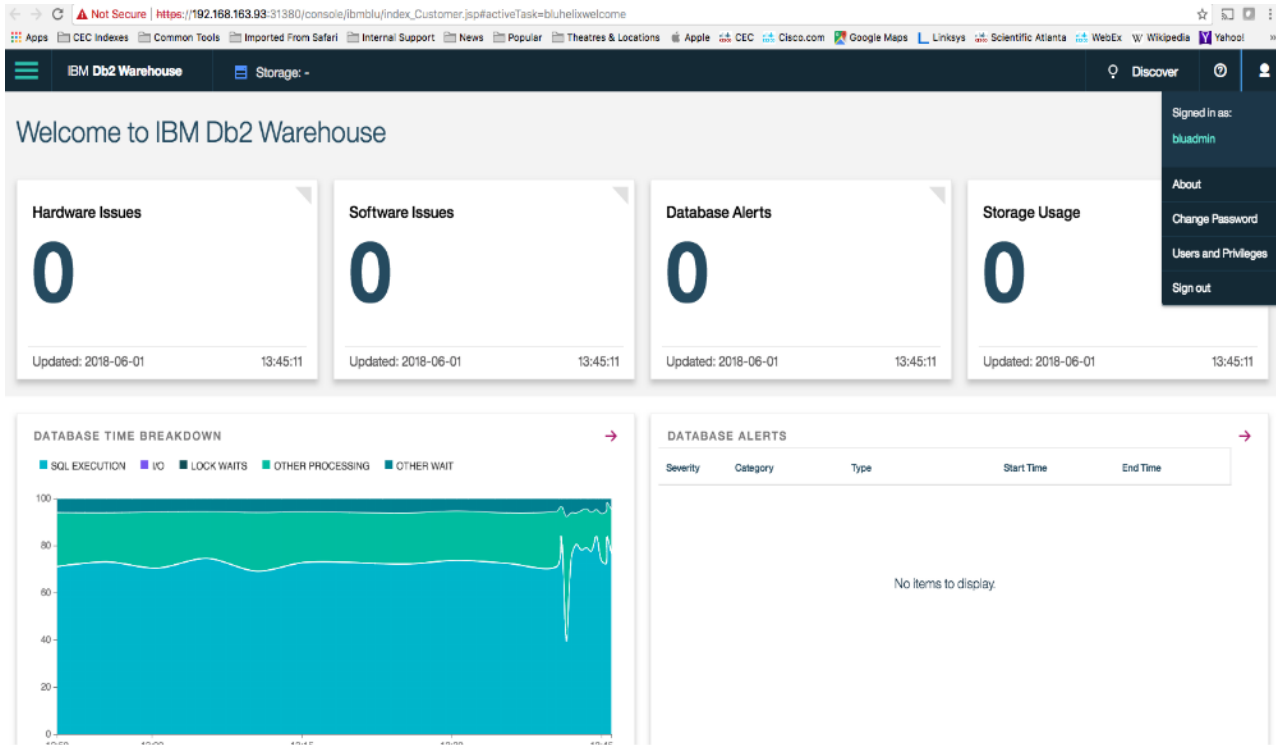
warehouse1-ibm-db2warehouse-dev

[Overview](#)

Service details

Type	Detail
Name	warehouse1-ibm-db2warehouse-dev
Namespace	default
Created	1 hour ago
Type	NodePort
Labels	app=warehouse1-ibm-db2warehouse-dev,chart=ibm-db2warehouse-dev-2.0.0,heritage=Tiller,release=warehouse1
Selector	app=warehouse1-ibm-db2warehouse-dev
Cluster IP	10.0.0.29
External IP	-
Port	port50000 50000/TCP; port50001 50001/TCP; port8443-https 8443/TCP
Node port	port50000 32361/TCP port50001 32254/TCP port8443-https 31380/TCP
Session affinity	None

12. Make sure the service installed successfully by logging in with the default user (Blueadmin) and password provided during installation.



For more information about deploying applications on IBM cloud private, please refer to:
<https://www.ibm.com/cloud/garage/architectures/private-cloud>.

Validated Hardware and Software

Table 6 lists the hardware and software versions used for the solution validation. It is important to note that Cisco, IBM, and VMware have interoperability matrices that should be referenced to determine support for any specific implementation of VersaStack. See the following links for more information:

- [IBM System Storage Interoperation Center](#)
- [Cisco UCS Hardware and Software Interoperability Tool](#)
- [VMware Compatibility Guide](#)

Table 6 Hardware and Software Revisions Validated

Layer	Device	Image	Comments
Compute	Cisco UCS Fabric Interconnects 6300 Series, Cisco UCS B-200 M5, Cisco UCS C-220 M5	3.2(3d)	Includes the Cisco UCS-IOM 2304, Cisco UCS Manager, and Cisco UCS VIC 1340
	Cisco nenic Driver	1.0.16.0	Ethernet driver for Cisco VIC
	Cisco fnic Driver	1.6.0.37	FCoE driver for Cisco VIC
Network	Cisco Nexus Switches	7.0(3)I4(7)	NXOS
	Cisco MDS 9396S	7.3(0)DY(1)	FC switch firmware version
Storage	IBM SVC	7.8.1.4	Software version
	IBM FlashSystem 900	1.4.5.0	Software version
	IBM Storwize V5030	7.8.1.4	Software version
Software	VMware vSphere ESXi	6.5 update 1	Software version
	VMware vCenter	6.5 update 1	Software version
	IBM Cloud Private	1.2.0.1	Software version
	Spectrum Connect	3.4.0	Software version
	IBM storage enabler for containers	1.1.0	Software version

Validation

Test Plan

The VersaStack for IBM Cloud Private solution was validated by deploying multiple containerized applications on IBM Cloud Private running on VersaStack infrastructure. The system was validated for resiliency by failing various aspects of the system under the load. The types of tests executed on the system are as follows:

VersaStack Infrastructure Validation

- Failure and recovery of links from Cisco UCS Chassis (IOM) to FI-A and FI-B, one at a time
- Rebooting Cisco UCS FI, one at a time
- Removing the physical cables between FI and Cisco Nexus 93180 switches to simulate path failure
- Fail/power off both Cisco 93180 switches, one after other
- Failure and recovery of links with in vPC from UCS FI and Cisco Nexus 93180 switches
- Failure and recovery of the links between Cisco Nexus 93180 switches and IBM SVC Nodes

IBM Cloud Private Environment Validation

- Failure and recovery of ESXi hosts in a cluster (rebooting of hosts, shutting down of hosts, etc.)
- In case of a host failure, verify ICP nodes auto restart within the HA cluster
- ICP node vMotion across ESXi servers
- Failure and recovery of ICP master and worker nodes in the ICP cluster
- Installation of multiple containerized applications such as MongoDB, IBM DB2 developer edition, IBM WebSphere Liberty and some other multi-tiered micro-services applications such as IBM bluecompute.

Bill of Materials

To find various components of VersaStack system, complete the following steps:

1. Go to the Main CCW page: <https://apps.cisco.com/Commerce/home>.
2. Under Find Products and Solutions, Click on the Search for solutions.
3. Type **VersaStack**. System will pull all the VersaStack variations.
4. Select one of the solutions and click **View Components**.

Summary

Private Cloud and Converged Infrastructure are currently two of the most effective and active IT solution domains in the marketplace. Global IT industry leaders, IBM and Cisco, have responded to this skyrocketing customer demand by offering VersaStack for IBM Cloud Private. The solution enables enterprises to exploit and manage the dynamics of cloud while still addressing compliance and regulatory requirements.

With VersaStack for IBM Cloud Private, development and administrative teams share a flexible cloud environment behind their firewalls to create new microservices-based applications, modernize existing apps using cloud-enabled middleware and securely integrate between the two. With a consistent underlying Kubernetes-based platform, IBM has enabled key elements of its existing and new middleware, data and analytics portfolio to take advantage of the platform capabilities, including rapid provisioning and de-provisioning of applications, improved resource usage and simplified management.

References

Products and Solutions

IBM Cloud Private:

<https://www.ibm.com/cloud/private>

IBM Spectrum Connect:

<https://www.ibm.com/us-en/marketplace/spectrum-connect>

IBM Storage Enabler for Containers:

<https://www.ibm.com/blogs/systems/tag/ibm-storage-enabler-for-containers>

IBM Garage:

<https://www.ibm.com/cloud/garage/>

Cisco Unified Computing System:

<http://www.cisco.com/en/US/products/ps10265/index.html>

Cisco UCS 6200 Series Fabric Interconnects:

<http://www.cisco.com/en/US/products/ps11544/index.html>

Cisco UCS 5100 Series Blade Server Chassis:

<http://www.cisco.com/en/US/products/ps10279/index.html>

Cisco UCS B-Series Blade Servers:

<http://www.cisco.com/c/en/us/products/servers-unified-computing/ucs-b-series-blade-servers/index.html>

Cisco UCS C-Series Rack Servers:

<http://www.cisco.com/c/en/us/products/servers-unified-computing/ucs-c-series-rack-servers/index.html>

Cisco UCS Adapters:

http://www.cisco.com/en/US/products/ps10277/prod_module_series_home.html

Cisco UCS Manager:

<http://www.cisco.com/en/US/products/ps10281/index.html>

Cisco Nexus 9000 Series Switches:

<http://www.cisco.com/c/en/us/support/switches/nexus-9000-series-switches/tsd-products-support-series-home.html>

Cisco Application Centric Infrastructure:

<http://www.cisco.com/c/en/us/solutions/data-center-virtualization/application-centric-infrastructure/index.html>

VMware vCenter Server:

<http://www.vmware.com/products/vcenter-server/overview.html>

VMware vSphere:

https://www.vmware.com/tryvmware_tpl/vsphere-55_evalcenter.html

IBM SAN Volume Controller

<http://www-03.ibm.com/systems/storage/software/virtualization/svc/>

IBM FlashSystem 900:

<http://www-03.ibm.com/systems/storage/flash/900/>

IBM Storwize V5000:

http://www-03.ibm.com/systems/storage/disk/storwize_v5000/overview.html

Cisco CloudCenter

<http://www.cisco.com/c/en/us/products/cloud-systems-management/cloudcenter/index.html>

IBM Spectrum Copy Data Management

<https://www.ibm.com/us-en/marketplace/spectrum-copy-data-management>

IBM Cloud

<https://www.ibm.com/cloud-computing/bluemix/>

Interoperability Matrixes

Cisco UCS Hardware Compatibility Matrix:

<http://www.cisco.com/c/en/us/support/servers-unified-computing/unified-computing-system/products-technical-reference-list.html>

VMware and Cisco Unified Computing System:

<http://www.vmware.com/resources/compatibility>

IBM System Storage Interoperation Center:

<http://www-03.ibm.com/systems/support/storage/ssic/interoperability.wss>

IBM System Storage Interoperation Center:

<http://www-03.ibm.com/systems/support/storage/ssic/interoperability.wss>

About the Authors

Sreenivasa Edula, Technical Marketing Engineer, UCS Data Center Solutions Engineering, Cisco Systems, Inc.

Sreeni has over 18 years of experience in Information Systems with expertise across Cisco Data Center technology portfolio, including DC architecture design, virtualization, compute, network, storage and cloud computing.

Acknowledgements

The author would like to acknowledge the following for their support and contribution to the design, validation and creation of this Cisco Validated Design:

- Matt Levan, Solutions Engineering Architect, IBM
- Cedric Cook, Principal Solutions Offering Manager, IBM
- Nita Maheswaren, Principal Offering Manager, IBM