Redbooks
ibm.com/redbooks

Practical Migration from x86 to IBM LinuxONE

Sandeep Batta
Youssef Largou
Pablo Paniagua
Cecilia Vales

IBM LinuxONE

IBM

Redbooks

IBM Redbooks

Practical Migration from x86 to IBM LinuxONE

September 2024

SG24-8377-02

Third Edition (September 2024)

This edition applies to IBM LinuxONE 4 LA1, IBM LinuxONE 4 LA2, IBM LinuxONE 4 AGL, and IBM LinuxONE Express.

Contents

Notices

Trademarks

Preface

IBM LinuxONE is a portfolio of hardware, software, and solutions for an enterprise-grade Linux environment. It is designed to run more transactions faster and with more security and reliability, specifically for the open-source community. It fully embraces open-source-based technology.

The following LinuxONE servers are now available: IBM LinuxONE 4 LA1, IBM LinuxONE 4 LA2, IBM LinuxONE 4 AGL, and IBM LinuxONE Express. We describe these servers in 1.2, “IBM LinuxONE servers” on page 5.

Aside from still running SUSE Linux Enterprise Server and Red Hat Enterprise Linux (RHEL) Servers, LinuxONE runs Ubuntu, which is popular on x86 hardware.

Ubuntu, which runs the cloud, smartphones, a computer that can remote control a planetary rover for NASA, many market-leading companies, and the Internet of Things, is now available on IBM LinuxONE servers. Together, these two technology communities deliver the perfect environment for cloud and DevOps. Ubuntu 16.04 on LinuxONE offers developers, enterprises, and Cloud Service Providers a scalable and secure platform for next-generation applications that include OpenStack, Kernel-based Virtual Machine (KVM), Docker, and JuJu.

The following reasons are why you would want to optimize your servers through virtualization by using LinuxONE:

- Too many distributed physical servers with low utilization
- A lengthy provisioning process that delays the implementation of new applications
- Limitations in data center power and floor space
- High total cost of ownership (TCO)
- Difficulty allocating processing power for a dynamic environment

This IBM Redbooks® publication provides a technical planning reference for IT organizations that are considering migrating from their x86 distributed servers to LinuxONE. This book walks you through some of the important considerations and planning issues that you might encounter during a migration project. Within the context of a pre-existing UNIX based or x86 environment, it presents an end-to-end view of the technical challenges and methods necessary to complete a successful migration to LinuxONE.

Authors

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Poughkeepsie Center.

Sandeep Batta started out as a systems programmer working on IBM GDPS® and other technologies. Since then, Sandeep has designed and managed technology refresh programs, cloud infrastructure deployments, and offerings for backup-as-a-service. He is now a Lead Solutions Architect for IBM® in the IBM Hyper Protect organization, where he works with clients in financial services, insurance, and digital assets sectors, where he addresses their data-protection requirements with solutions that bring the latest in cryptography and confidential computing to customer use cases.

Youssef Largou is the founding director of PowerM, a platinum IBM Business Partner in Morocco. He has 22 years of experience in systems, high-performance computing (HPC), middleware, and hybrid cloud, which include IBM Power, IBM Storage, IBM Spectrum®, IBM WebSphere®, IBM Db2®, IBM Cognos®, IBM WebSphere Portal, IBM MQ, ESB, IBM Cloud Pak®, SAP HANA, and Red Hat OpenShift. He has worked within numerous industries with many technologies. Youssef is an IBM Champion 2020, 2021, 2022, 2023, and 2024, and he is an IBM Redbooks Platinum Author that has designed many reference architectures. His company has been recognized as an IBM Beacon Award Finalist in Storage, software-defined storage, and LinuxONE five times. He is a regular speaker at IBM Think®, IBM TechXchange, and the Common Europe Congress. He holds an Engineering degree in Computer Science from the Ecole Nationale Supérieure des Mines de Rabat and an Executive MBA from EMLyon.

Pablo Paniagua is an IBM Z® Hybrid Cloud Client Technical Specialist in Spain. He has 5 years of experience with Red Hat OpenShift and LinuxONE. He holds a degree in Telecommunications Engineering from Universidad Pontificia de Comillas. Before IBM, Pablo worked as an IT consultant on distributed systems. During his time at IBM, he has helped many customers with their Linux on IBM Z environments and hybrid cloud strategies.

Cecilia Vales is an Expert Labs Delivery Specialist on IBM Z and LinuxONE in Spain. She has over 3 years of experience in the IBM Z and LinuxONE fields as a hardware technical specialist. She also works on project delivery for customers. She holds a degree in Industrial Electronics and Automation Engineering from Universidad Carlos III de Madrid. Her areas of expertise include security, cyber resiliency, and modernization in the IBM Z and LinuxONE fields. She has written extensively on hybrid cloud solutions for LinuxONE among other modernization topics for IBM Z and LinuxONE servers.

Thanks to the following people for their contributions to this project:

Lydia Parziale
IBM Redbooks, Poughkeepsie Center

Robert Haimowitz
IBM, Poughkeepsie Center

Stefan Schmitt and Ingo Adlung
IBM Germany

Filipe Miranda
IBM US

Rosana Rueda Elias and Beatriz Marciel Oliveira
Kyndryl, Brazil

Thanks to the authors of the previous editions of this book.

- Authors of the second edition, *Practical Migration from x86 to IBM LinuxONE*, January 2021, were:

  Michel M Beaulieu, Felipe Cardeneti Mendes, Guilherme Nogueira, and Lena Roesch

- Authors of the first edition, *Practical Migration from x86 to IBM LinuxONE*, published in September 2014, were:

  Lydia Parziale, Eduardo Simoes Franco, Craig Gardner, Berthold Gunreben, Tito Ogando, and Serkan Sahin

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience by using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an email to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

Part 1

Decision-making

This part describes the key benefits and reasons to migrate to IBM LinuxONE. It also describes some of the considerations when you decide to migrate to LinuxONE, and ends with some virtualization concepts to help you understand and compare x86 virtualization, and both Kernel-based Virtual Machine (KVM) and IBM z/VM® virtualization.

This part includes the following chapters:

- Chapter 1, “Benefits of migrating workloads to IBM LinuxONE” on page 3
- Chapter 2, “Analyzing and understanding” on page 25
- Chapter 3, “Virtualization concepts” on page 41

1

Benefits of migrating workloads to IBM LinuxONE

This chapter describes the benefits and reasons to migrate workloads to IBM LinuxONE. It also describes how IBM LinuxONE supports application modernization and the shift to a microservices architecture. It explains how any private, public, or hybrid cloud can benefit from LinuxONE with a cloud computing blueprint.

This chapter includes the following topics:

1.1 Benefits

IBM LinuxONE is an Enterprise Linux platform that is built for open innovation that combines the best of Linux and open technology with the best of enterprise computing in one system. The platform is designed with a focus on security, scalability, and performance to support customers who want an efficient and cost-effective solution to thrive in a data-centric economy.

The IBM LinuxONE hardened, Linux based software stack is built on the trusted s390x architecture and can run most open-source software packages, such as databases and data management, virtualization platforms, and containers; automation and orchestration software; and compute-intensive workloads, such as blockchain.

Linux is available on many computing platforms, from set-top boxes and hand-held devices to the largest servers. The flexibility of the operating system enables users to run applications without being tied to a particular hardware platform. You have control over the choice of the hardware platform that supports your application. Workloads running on IBM LinuxONE benefit from a hardware platform that includes specialized processors for artificial intelligence (AI), input/output (I/O), cryptographic cards, and a combination of hypervisors that grant unparalleled flexibility in Linux deployment. Successive generations of the IBM LinuxONE platform have delivered unparalleled cutting-edge capabilities. For more information, see IBM LinuxONE.

A significant benefit of Linux is that it is open source. The software is unencumbered by licensing fees, and its source code is freely available. Hundreds of Linux distributions are available for almost every computing platform. The following three enterprise Linux distributions[1] are supported on IBM LinuxONE:

- Red Hat: Red Hat Enterprise Linux (RHEL)
- SUSE: SUSE Linux Enterprise Server
- Canonical: Ubuntu Server

These Linux distributions offer various support options to customers who use Linux, which include 24x7 support with a 1-hour response time worldwide for customers running production systems. In addition to the Linux operating system, all major Linux distributions offer several other open-source products that they also support.

To simplify problem determination, IBM customers can contact IBM in the first instance, and if it is a new problem with Linux, IBM works with the distributors to resolve the problem.

The increased interest and usage of Linux resulted from its rich set of features, including virtualization, security, Microsoft Windows interoperability, development tools, a growing list of independent software vendor (ISV) applications, performance, and, most importantly, its multiplatform support.

This multiplatform support enables customers to run a common operating system across all computing platforms, which means lower support costs and, for Linux, no incremental license charges. It also enables customers to easily move applications to the most appropriate platform. For example, many IT organizations choose Linux because of its ability to scale databases across highly scalable hardware.

[1] A Linux distribution is a complete operating system and environment. It includes compilers, file systems, and applications such as Apache (web server), SAMBA (file and print), sendmail (mail server), Tomcat (Java application server), MySQL (database), and many others.

1.2 IBM LinuxONE servers

The first two LinuxONE products were named Emperor and Rockhopper. Emperor III and Rockhopper III, the third iteration of LinuxONE, were released in 2019 and early 2020. The latest iteration of the system is the IBM LinuxONE 4, which is available in the following models and features AI and quantum security technologies:

- IBM LinuxONE 4 LA1

  The newest member of the LinuxONE family, the IBM LinuxONE 4 was generally available in late 2022 and maintains the new form factor that was introduced in LinuxONE III, which features a 19-inch frame that flexibly scales 1 - 4 frames. It is designed around the new Telum processor with 8 cores per chip with Dual Chip Module packaging. It is configurable with up to 200 cores running at 5.2 GHz, up to 40 TB of RAM, and 10 TB of Redundant Array of Independent Memory (RAIM) per central processing drawer

- IBM LinuxONE 4 LA2

  Released in May 2023, the IBM LinuxONE 4 LT2 is the newest entry model into the IBM LinuxONE family of servers. It delivers a 19-inch single-frame (versus the option of up to four frames for the LT1) with an efficient design with a low entry cost that can easily coexist with other platforms in a cloud data center. This model is designed around the new Telum processor with 8 cores per chip with Dual Chip Module packaging. It is configurable with up to 68 cores running at 4.6 GHz, up to 16 TB of RAM, and 8 TB of RAIM per central processing drawer.

- IBM LinuxONE 4 AGL

  Released in May 2023, the IBM LinuxONE 4 AGL is a new entry model option that consists of a rack-mounted model. It delivers a rack-mount option 10U - 39U that can be collocated with other technologies with a client-supplied 19-inch rack. This model is designed around the new Telum processor with 8 cores per chip with Dual Chip Module packaging. It is configurable with up to 68 cores running at 4.6 GHz, up to 16 TB of RAM, and 8 TB of RAIM per central processing drawer.

- IBM LinuxONE Express

  IBM LinuxONE Express is a special offering that is based entirely on IBM LinuxONE 4 AGL hardware but preconfigured and bundled specifically as three sizes: small, medium, and large. The small size consists of 4 cores and 384 GB of RAM. The medium size consists of 6 cores and 512 GB of RAM. The large size consists of 12 cores and 736 GB of RAM.

IBM LinuxONE 4 is the industry's first quantum-safe enterprise Linux system, which integrates new hardware encryption capabilities that enable users to apply quantum-safe encryption to protect data workloads and infrastructure. Each core includes a dedicated coprocessor for cryptographic functions, which is known as the Central Processor Assist for Cryptographic Functions (CPACF). CPACF supports pervasive encryption and provides hardware acceleration for encryption operations.

The compression capabilities were improved and deliver greater compression throughput than previous-generation systems. This on-chip compression co-processor uses industry-standard compression algorithms and can reduce data storage requirements and costs. This compression can also increase data transfer rates to boost throughput above comparable x86 CPUs, all without adversely impacting response times.

The new Telum chip is the first server-class chip with a dedicated on-chip AI accelerator that provides IBM LinuxONE with the capacity to run real-time inferences at speed and scale by colocating data and AI.

For more information, see IBM LinuxONE.

1.2.1 IBM LinuxONE 4 LA1

The IBM LinuxONE 4 LA1 is built with a 19-inch, industry-standard frame that flexibly scales 1 - 4 frames, depending on the configuration that is required. This new form factor maintains approximately the same maximum floor space as older generations and enables most clients to reduce their floor space.

The doors are designed for acoustics and optimized for air flow. The IBM LinuxONE 4 LT1 offers air-cooled (internal radiator) systems or water-cooled systems (WCS).

At the heart of the LinuxONE 4 LA1 is the new Telum processor, which has 8 cores per processor chip and is included with Dual Chip Module packaging. It uses the density and efficiency of 7-nm silicon-on-insulator technology. It runs at 5.2 GHz, which delivers increased performance and capacity across a wide range of workloads.

Up to 200 client configurable cores are available, which are known as Integrated Facility for Linux (IFL) processors. IBM LinuxONE 4 LA1 includes processor capacity that is represented by feature codes.

Five processor capacity feature codes are available for the IBM LinuxONE 4 LA1: Max39, Max82, Max125, Max168, and Max200. The numbering signifies the number of IFL processors that can be configured, for example, a Max 39 can configure up to 39 IFLs (cores), Max82 for up to 82 IFLs, and so on.

The system offers 8 TB of RAIM per central processing complex (CPC) drawer and up to 40 TB per system. RAIM is intended to provide redundancy for primary memory, sockets, and memory channels for more reliability and availability.

IBM LinuxONE 4 LA1 data sheet

The data sheet for the LinuxONE 4 LA1 model is shown in Table 1-1. The complete data sheet can be found at IBM LinuxONE Emperor 4.

Table 1-1 LinuxONE 4 LA1 at a glance

IBM LinuxONE 4 LA1 features		
LinuxONE 4 models	**Cores**	**Memory: Min - Max**
LA1	Up to 200	512 GB - 40 TB
Cryptography		
Crypto-Express8S (2-port adapters)	Maximum: 30 adapters and 60 Hardware Security Modules (HSMs)	
Crypto-Express8S (1-port adapter)	Maximum: 16 adapters and 16 HSMs	
Connectivity		
IBM FICON® Express32S SX/LX	Maximum: 384 ports	
IBM Adapter for NVMe1.1	1 slot	
OSA Express 7S 1.2 (1000BT/1G)	Maximum: 96 ports	

IBM LinuxONE 4 LA1 features		
OSA Express 7S 1.2 (10G/25G)	Maximum: 48 ports	
Inter-logical partition (LPAR) connectivity		
HiperSockets	Up to 32 high-speed virtual local area networks (VLANs)	
Shared Memory Communications - Direct Memory Access (SMC-D)	Up to 32 Internal Shared Memory (ISM) virtual CHIPDs	
Supported distributors		
Linux	Canonical, Red Hat, and SUSE with their latest supported releases and versions. For the certified levels, see Linux on IBM Z/IBM LinuxONE tested platforms.	
Supported hypervisors		
IBM z/VM	IBM z/VM V7.2, IBM z/VM 7.3, and IBM z/VM 7.4	
Kernel-based Virtual Machine (KVM)	KVM hypervisor, which is offered with the Linux distribution	
IBM partitioning technology	Up to 85 LPARs for secure workload isolation	

1.2.2 IBM LinuxONE 4 LA2

The IBM LinuxONE 4 LA2 is an air-cooled, single-frame, and efficient design with a low entry cost. As with the other servers in the LinuxONE family, it is designed to help enable cloud-native development and deployment, achieve encryption everywhere, and provide cyber resiliency to help ensure scalable isolation of workloads to protect from threats while helping ensure continuous availability (CA) of services. The system offers up to 40 LPARs, enabling various workloads to run on a single server.

The IBM LinuxONE 4 LA2 is based on the same 8-core DCM processor as the LinuxONE 4 LA1, which leverages the density and efficiency of 7-nm silicon-on-insulator technology. This model LA2 is available with four feature-based sizing options: Max5, Max16, Max32, and Max68

The IBM LinuxONE 4 LA2 design incorporates two CPC drawers for the Max68. The numbering signifies the number of IFL processors that can be configured, for example, a Max16 can configure up to 16 IFLs (cores), Max32 for up to 32 IFLs, and so on. The cores run at 4.6 GHz.

The system offers 8 TB of RAIM per CPC drawer and up to 16 TB total per LinuxONE 4 LT2 system, depending on the configuration. RAIM is intended to provide redundancy for primary memory, sockets, and memory channels for added reliability and availability. IBM Virtual Flash Memory (VFM) is now in the RAIM and provides high levels of availability and performance.

LinuxONE 4 LA2 data sheet

The data sheet for the LinuxONE 4 LA1 model is shown in Table 1-2.

Table 1-2 LinuxONE 4 LA2 at a glance

IBM LinuxONE 4 LA2 features		
LinuxONE 4 models	**Cores**	**Memory: Min - Max**
LA2	Up to 68	64 GB - 16 TB
Cryptography		
Crypto-Express8S (2-port adapters)	Maximum: 20 adapters and 40 HSMs	
Crypto-Express8S (1-port adapter)	Maximum: 16 adapters and 16 HSMs	
Connectivity		
IBM FICON Express32S SX/LX	Maximum: 96 ports	
IBM Adapter for NVMe1.1	1 slot	
OSA Express 7S 1.2 (1000BT/1G)	Maximum: 96 ports	
OSA Express 7S 1.2 (10G/25G)	Maximum: 48 ports	
Inter-LPAR connectivity		
HiperSockets	Up to 32 high-speed VLANs	
Shared Memory Communications - Direct Memory Access (SMC-D)	Up to 32 ISM virtual CHIPDs	
Supported distributors		
Linux	Canonical, Red Hat, and SUSE with their latest supported releases and versions. For the certified levels, see Linux on IBM Z/IBM LinuxONE tested platforms.	
Supported hypervisors		
IBM z/VM	z/VM V7.2, z/VM 7.3, and z/VM 7.4	
KVM	KVM hypervisor, which is offered with the Linux distribution	
IBM partitioning technology	Up to 40 LPARs for secure workload isolation	

1.2.3 IBM LinuxONE 4 AGL

The IBM LinuxONE4 AGL is the latest addition to the LinuxONE family. It delivers a rack-mounted option 10U - 39U that can be collocated with other technologies with a client-supplied 19-inch rack.

IBM LinuxONE4 AGL is designed to help enable cloud-native development and deployment, achieve encryption everywhere, and provide cyber resiliency to help ensure scalable isolation of workloads to protect from threats while helping ensure CA of services. The system offers up to 40 LPARs, enabling various workloads to run on a single server.

The IBM LinuxONE 4 AGL is based on the same 8-core DCM processor as the IBM LinuxONE 4 LA1 and LA2, which leverages the density and efficiency of 7-nm silicon-on-insulator technology. Model AGL is available with three feature-based sizing options: Max5, Max32, and Max68.

The IBM LinuxONE 4 AGL design incorporates two CPC drawers for the Max68. The numbering signifies the number of IFL processors that can be configured, for example, a Max32 can configure up to 32 IFLs, and so on. The cores run at 4.6 GHz.

The system offers 8 TB of RAIM per CPC drawer and up to 16 TB total per LinuxONE 4 LT2 system, depending on the configuration. RAIM is intended to provide redundancy for primary memory, sockets, and memory channels for added reliability and availability. IBM VFM is now in the RAIM and provides high levels of availability and performance.

IBM LinuxONE 4 AGL data sheet

The data sheet for IBM LinuxONE 4 AGL model is shown in Table 1-3.

Table 1-3 LinuxONE 4 AGL at a glance

IBM LinuxONE 4 AGL features		
LinuxONE 4 models	**Cores**	**Memory: Min - Max**
AGL	Up to 68	64 GB - 16 TB
Cryptography		
Crypto-Express8S (2-port adapters)	Maximum: 20 adapters with 40 HSMs	
Crypto-Express8S (1-port adapter)	Maximum: 16 adapters with 16 HSMs	
Connectivity		
IBM FICON Express32S SX/LX	Maximum: 96 ports	
IBM Adapter for NVMe1.1	1 slot	
OSA Express 7S 1.2 (1000BT/1G)	Maximum: 96 ports	
OSA Express 7S 1.2 (10G/25G)	Maximum: 48 ports	
Inter-LPAR connectivity		
HiperSockets	Up to 32 high-speed VLANs	
Shared Memory Communications - Direct Memory Access (SMC-D)	Up to 32 ISM virtual CHIPDs	
Supported distributors		
Linux	Canonical, Red Hat, and SUSE with their latest supported releases and versions. For the certified levels, see Linux on IBM Z/IBM LinuxONE tested platforms.	
Supported hypervisors		
IBM z/VM	z/VM V7.2, z/VM 7.3, and z/VM 7.4	

IBM LinuxONE 4 AGL features	
KVM	KVM hypervisor, which is offered with the Linux distribution
IBM partitioning technology	Up to 40 LPARs for secure workload isolation

1.3 Reasons to choose IBM LinuxONE

IBM LinuxONE delivers the best of enterprise Linux on the industry's most reliable and highly scalable hardware. These systems are specialized scale-up enterprise servers that are designed exclusively to run Linux applications.

IBM LinuxONE provides the highest levels of availability (near 100% uptime with no single point of failure (SPOF)), performance, throughput, and security. End-to-end security is built in with isolation at each level in the stack, and provides the highest level of certified security in the industry.

IBM Secure Execution for Linux is a continuation and expansion of security features of IBM Z and LinuxONE. It supplements pervasive encryption, which protects data at-rest and data in-flight to also protect data in-use. IBM Secure Execution for Linux is a hardware-based security technology that provides a trusted execution environment (TEE) for "confidential computing". It provides scalable isolation for individual workloads to protect them from external attacks and insider threats. For more information, see *IBM Hyper Protect Platform: Applying Data Protection and Confidentiality in a Hybrid Cloud Environment*, SG24-8555.

The CryptoExpress (CEX) cards that are available on IBM LinuxONE 4 provide access to a Federal Information Processing Standards (FIPS) 140-2 Level 4 HSM that is used by Hyper Protect Services to provide encryption key management services and quantum-safe features for signing and encapsulation with Dilithium and Kyber.

Also, LinuxONE Systems facilitate transparent use of redundant processor execution steps and integrity checking, which is necessary in the financial services industries. LinuxONE servers typically enable hot-swapping of hardware, such as processors and memory. This swapping is typically transparent to the operating system, enabling routine repairs to be performed without shutting down the system.

IBM LinuxONE delivers on the promise of a flexible, secure, and smart IT architecture that can be managed seamlessly to meet the requirements of today's fast-changing business climate.

1.3.1 Best of Enterprise Linux and open source

LinuxONE provides the following benefits:

- A premium Linux experience with subsecond user response times and virtually unlimited scale.
- A broad portfolio of open-source and other vendor products and tools that are delivered on the platform.
- Choice of Linux (RHEL, SUSE, and Ubuntu) and tools that best fit your environment.
- Eliminates risks by running Linux on the industry's most secure and resilient hardware platform.

- Easy integration of data and applications with existing IBM Z based solutions.
- Overall increases the operational IT efficiency.

1.3.2 Hardware strengths

IBM LinuxONE provides these hardware strengths:

- Reliability:
  - Redundant processors, I/O, and memory.
  - Error correction and detection.
  - Remote Support Facility.
- Availability:
  - Fault tolerance.
  - Automated failure detection.
  - Non-disruptive hardware and software changes.
  - IBM LinuxONE machines have the industry's highest reliability levels for over a decade, with up to 99.99999% or greater availability.
- Virtualization:
  - High-performance logical partitioning by using IBM Processor Resource/Systems Manager (IBM PR/SM).[2]
  - Up to 85 (LinuxONE 4 LA1) or 40 (LinuxONE 4 LA2 or AGL) LPARs with independent virtual resources.
  - PR/SM is one of the most secure systems that are available, having achieved Common Criteria Evaluation Assurance Level 5+ (EAL5+) for partition isolation. This is one of the highest levels of certification that can be achieved by commercially available hardware.

    Note: For more information about Common Criteria, Evaluation Assurance Levels, Protection Profiles, and a list of certified products, see this website.

  - IBM Dynamic Partition Manager (DPM) provides facilities to define and run virtualized computing systems by using a firmware-managed environment that coordinates the physical system resources that are shared by the partitions. The partitions resources include processors, memory, network, storage, crypto, and accelerators.
  - Both the industry-leading virtualization hypervisor z/VM and the open-source hypervisor KVM are supported on all IBM LinuxONE models.
  - PR/SM, z/VM, and KVM employ hardware and firmware innovations that make virtualization part of the basic fabric of the IBM LinuxONE platform.
  - IBM HiperSockets allows up to 32 VLANs, thus allowing memory-to-memory TCP/IP communication between partitions.

[2] PR/SM is a standard component of all IBM LinuxONE models, which enables LPARs to share system resources. PR/SM divides physical system resources, both dedicated and shared, into isolated LPARs. Each partition is like an independent system running its own operating environment. It is possible to add and delete resources like processors, I/O, and memory across partitions while they are actively in use.

- Scalability:
  - IBM LinuxONE 4 Model LA1 scales to 200 physical processors and up to 40 TB of memory.
  - IBM LinuxONE III Models LA2 and AGL scale to 68 physical processors and up to 16 TB of memory.
  - LinuxONE 4 Model LA1 can process 1 trillion web transactions per day and support thousands of virtual servers or up to two million containers on a single system.
- Security:
  - The LinuxONE pervasive encryption capabilities enable you to encrypt massive amounts of data with little effect on your system performance. The LinuxONE hardware benefits from encryption logic and processing on each processor chip in the system.
  - The CPACF is well suited for encrypting large amounts of data in real time because of its proximity to the processor unit. CPACF supports the following protocols:
    - DES
    - TDES
    - AES-128
    - AES-256
    - SHA-1
    - SHA-2
    - SHA-3
    - SHAKE
    - DRNG
    - TRNG
    - PRNG

    With the LinuxONE 4, CPACF supports Elliptic Curve Cryptography clear key, which improves the performance of Elliptic Curve algorithms.

    The following algorithms are supported:
    - EdDSA (Ed448 and Ed25519)
    - ECDSA (P-256, P-384, and P-521)
    - ECDH (P-256, P-384, P521, X25519, and X448)

    Protected key signature creation is also supported.
  - Optional cryptography accelerators provide improved performance for specialized functions:
    - Can be configured as a secure key coprocessor or for Secure Sockets Layer (SSL) acceleration.
    - Certified at FIPS 140-3 and Common Criteria EAL 4+.
  - The IBM Hyper Protect Virtual Server offering is exclusive to IBM LinuxONE. It delivers more security capabilities to protect Linux workloads from internal and external threats throughout their lifecycle, that is, the build, management, and deployment phases. Some of the security benefits include the following ones:
    - Building images with integrity, which Secures continuous integration and delivery.
    - Managing infrastructure with least privileged access to applications and data.
    - Deploying images with trusted provenance.
  - The IBM LinuxONE 4 maintains Secure Execution for Linux. It is a hardware-based security technology that is designed to protect and isolate workloads on-premises or on IBM LinuxONE and IBM Z hybrid cloud environments. Users, and even system administrators, cannot access sensitive data in Linux-based virtual environments.

- Regulatory compliance

  Most security regulations feature include specific requirements regarding encryption of data and access to that data. LinuxONE addresses those two aspects at the core of its platform through capabilities, such as EAL 5+ isolation between LPARs, pervasive encryption, and protection against side-channel attacks and insider threats with tamper resistant encrypted keys.

- Just-in-time (JIT) deployment of resources:
  - On/Off Capacity on Demand provides temporary processing capacity to meet short-term requirements or test new applications.
  - Flex Capacity dynamically shifts production capacity between different LinuxONE servers at multiple sites.
  - Capacity BackUp (CBU) enables you to replace model capacity or specialty engines to a backup server in the event of an unforeseen loss of server capacity because of an emergency. CBU helps ensure that customers can access more capacity during a disaster recovery (DR) situation without having to purchase more capacity. Typically, this system allows customers to sign up for CBU on an IBM LinuxONE at another site and use this capacity for some contracted DPM tests or for a contracted time during a declared disaster at the customer site.
- Power and cooling savings

  With its low power and cooling requirements, IBM LinuxONE is an ideal platform for the consolidation of distributed servers.

1.4 Financial benefits of a migration

In today's rapidly evolving technological landscape, the importance of a secure and highly available infrastructure cannot be overstated. As organizations explore cost reduction initiatives; grapple with stringent privacy regulations; tackle data sovereignty challenges, Environmental, Social, and Governance (ESG) issues, especially environmental sustainability; embark on digital transformation journeys; and confront the complexities of AI and quantum computing, the need for a robust infrastructure is paramount.

One of the most pivotal advantages that is offered by IBM LinuxONE is its total cost of ownership (TCO) optimization, which extends to various areas such as software licenses, CO2 carbon footprint reduction, and the inclusion of enhanced reliability, availability, and security features.

Businesses leverage TCO to conduct a comprehensive cost comparison between IBM solutions and alternative options, whether from competitors or in-house solutions. When constructing a TCO for their next technology investment, it is imperative for them to consider several key components:

- Full environment coverage: Encompasses all environments, from testing and development to quality assurance, staging, production, and DPM.
- Comprehensive component inclusion: Involves accounting for various elements, such as hardware, software, cloud services, personnel, network infrastructure, security measures, storage solutions, and data center facilities.
- Business Value Assessment (BVA): Evaluating business value entails examining factors like time-to-market, customer retention rates, forecasting accuracy, scheduling efficiency, data privacy compliance, data sovereignty considerations, and potential service-level agreement (SLA) penalties.

- Quality of service metrics: This category focuses on assessing the availability, security robustness, reliability, performance levels, scalability potential, and portability of the solution.
- Time-related considerations: Involves factoring in aspects like upgrade cycles, refresh intervals, migration efforts, sunk costs, parallel costs during transition periods, and other time-sensitive factors.

IBM offers more support through its IT Economics team, which conducts BVAs. Unlike TCO studies, BVAs emphasize return on investment (ROI) analyses. For example, when evaluating IBM Cloud Paks on LinuxONE or Red Hat OpenShift deployments, the discussion might revolve around investments, particularly concerning transfers, upgrades, or license exchanges. These discussions transcend mere technological considerations; they delve into understanding how different organizations structure their licensing models and how such insights inform decision-making based on financial benefits.

Each of these features, which are described in 1.3, “Reasons to choose IBM LinuxONE” on page 10, contribute to reducing TCO in various ways:

- Zero trust security: By implementing a zero trust security model and incorporating features like Secure Execution for Linux and end-to-end encryption (including quantum-safe encryption), organizations can mitigate security risks and potential breaches. This in turn helps you to avoid costly security incidents, regulatory fines, and reputational damage that is associated with data breaches, which ultimately reduce overall TCO.
- Scalability with ease: The on-premises, cloud-like capacity model enables organizations to scale their infrastructure efficiently according to their changing needs. This scalability eliminates the need for over-provisioning resources, which can result in wasted investments.
- Performance improvement: Achieving an up to 10X performance improvement over x86 platforms leads to higher efficiency and productivity within the organization. With faster processing speeds and improved system responsiveness, organizations can accomplish tasks more quickly, which leads to increased operational efficiency and potentially lower labor costs, which helps reduce TCO.
- Low latency and colocation: Reduced latency and colocation capabilities result in faster data transfer and processing times, which is beneficial for latency-sensitive applications such as financial transactions or real-time analytics. By minimizing delays and optimizing data transfer, organizations can improve productivity and avoid potential revenue losses due to sluggish performance, which helps reduce TCO.
- High availability (HA): The promise of “seven 9's of availability” helps ensure minimal downtime and maximum uptime for critical business operations. This HA minimizes the risk of revenue loss that is associated with system outages and service disruptions. By maintaining continuous operation (CO), organizations can avoid costly downtime-related losses, which help enhance overall cost-effectiveness and reduce TCO.
- Software license savings and operational efficiency: By leveraging IBM LinuxONE features, organizations can optimize their software licensing costs through efficient resource utilization and workload consolidation. Also, the platform's energy-efficient design reduces power and cooling requirements, which lead to lower operational expenses. Furthermore, the reduced footprint saves on floor space, which can be a significant cost consideration in data center operations.

- Reduced energy consumption and operational costs: By leveraging a highly efficient system, organizations can decrease their energy consumption and operational expenses. Such systems are designed with energy-efficient components, optimized power management features, and advanced cooling mechanisms, which contribute to lower energy usage. As a result, organizations can save substantially on electricity bills and reduce their overall operational costs, which drive down TCO. Also, reduced energy consumption aligns with sustainability goals, which further enhances cost savings and corporate responsibility efforts.
- Reduction of floor space: IBM LinuxONE is designed to optimize space utilization in data centers. With its high consolidation ratio and support for virtualization technologies such as KVM and z/VM, LinuxONE enables organizations to run multiple workloads on a single physical server, which reduces the number of servers that are required and the floor space that is needed.
- Consistent transactional service levels: A massively scalable system helps ensure consistent transactional service levels, even during periods of high demand or peak loads. This reliability in performance helps organizations avoid disruptions, downtime, and service degradation, which can incur significant costs in terms of lost productivity, revenue, and customer satisfaction. By maintaining consistent service levels, organizations can minimize the need for costly emergency interventions, such as system upgrades or extra infrastructure investments, which lower overall TCO.

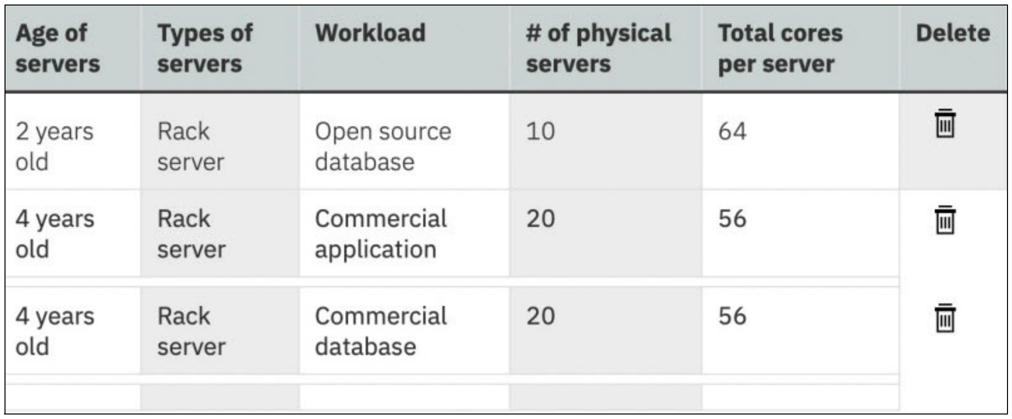
IBM LinuxONE or Linux on IBM Z TCO and CO2e Calculator

IBM LinuxONE or Linux on IBM Z TCO and CO2e Calculator is a tool that is designed to assess the environmental impact and TCO differences between the x86 architecture and IBM LinuxONE or Linux on IBM Z platforms when running similar workloads.

Here are some of the key features:

- CO2e Emission Comparison by evaluating the carbon dioxide equivalent (CO2e) emissions that are associated with running workloads on x86 versus IBM LinuxONE or Linux on IBM Z.
- License Consolidation for TCO Savings: The calculator demonstrates the potential TCO savings that are achieved by consolidating per-core licenses on IBM LinuxONE or Linux on IBM Z platforms.
- Floor space reduction: The TCO calculator evaluates the floor space gain when compared to x86 servers. By analyzing factors such as server density, power consumption, and cooling requirements, the calculator provides insights into the physical footprint reduction that is achievable with IBM LinuxONE.
- Cost reduction through enterprise software: Running enterprise software on IBM LinuxONE or Linux on IBM Z can lead to significant IT cost reductions.

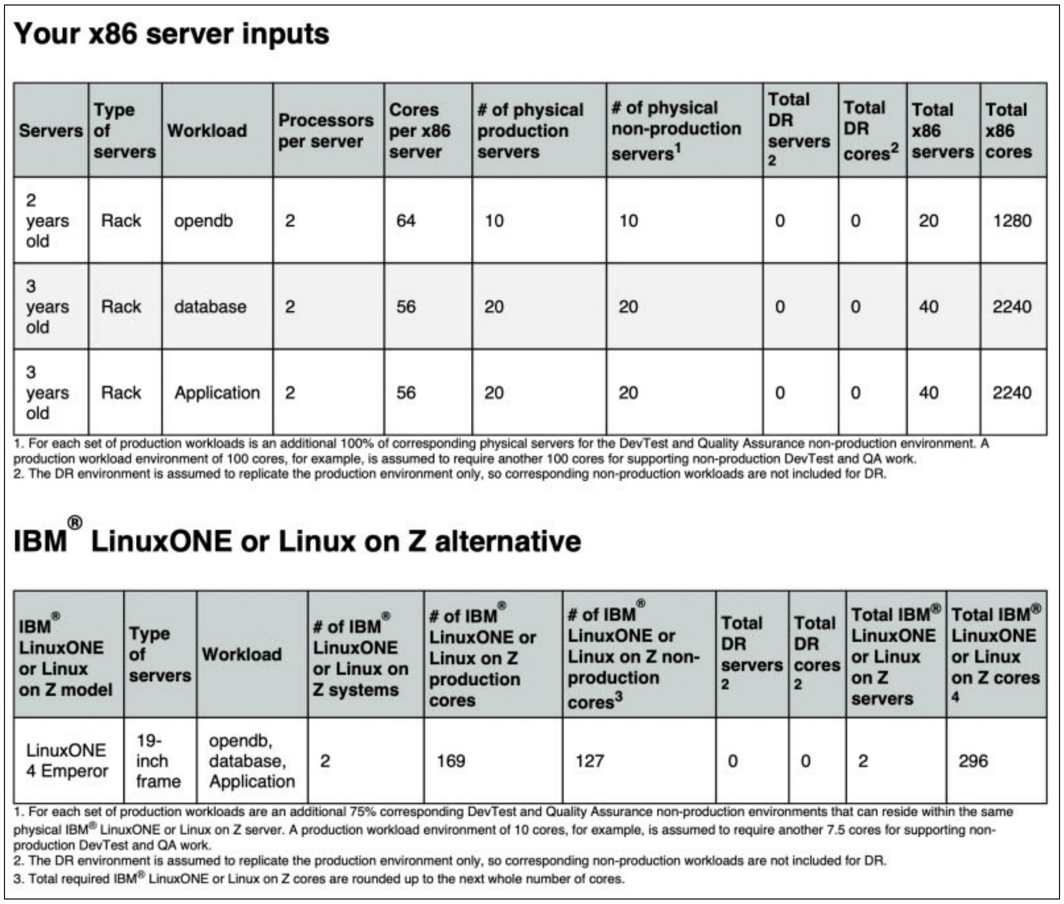
In Figure 1-1, we assess the TCO based on the current infrastructure setup of 10 rack servers with 64 cores per server running open-source databases, 20 rack servers with 56 cores per server running Commercial databases, and an extra 20 rack servers with 56 cores per server dedicated to running commercial applications.



Age of servers	Types of servers	Workload	# of physical servers	Total cores per server	Delete
2 years old	Rack server	Open source database	10	64	🗑
4 years old	Rack server	Commercial application	20	56	🗑
4 years old	Rack server	Commercial database	20	56	🗑

Figure 1-1 Existing x86 rack servers

On evaluation, the IBM LinuxONE or Linux on IBM Z TCO and CO2e calculator proposes consolidating all workloads onto two IBM LinuxONE 4 Emperor systems, with a total of 296 IFL processors, as shown in Figure 1-2, by showcasing the tangible benefits of migrating to the LinuxONE environment.



Your x86 server inputs

Servers	Type of servers	Workload	Processors per server	Cores per x86 server	# of physical production servers	# of physical non-production servers[1]	Total DR servers[2]	Total DR cores[2]	Total x86 servers	Total x86 cores
2 years old	Rack	opendb	2	64	10	10	0	0	20	1280
3 years old	Rack	database	2	56	20	20	0	0	40	2240
3 years old	Rack	Application	2	56	20	20	0	0	40	2240

1. For each set of production workloads is an additional 100% of corresponding physical servers for the DevTest and Quality Assurance non-production environment. A production workload environment of 100 cores, for example, is assumed to require another 100 cores for supporting non-production DevTest and QA work.
2. The DR environment is assumed to replicate the production environment only, so corresponding non-production workloads are not included for DR.

IBM® LinuxONE or Linux on Z alternative

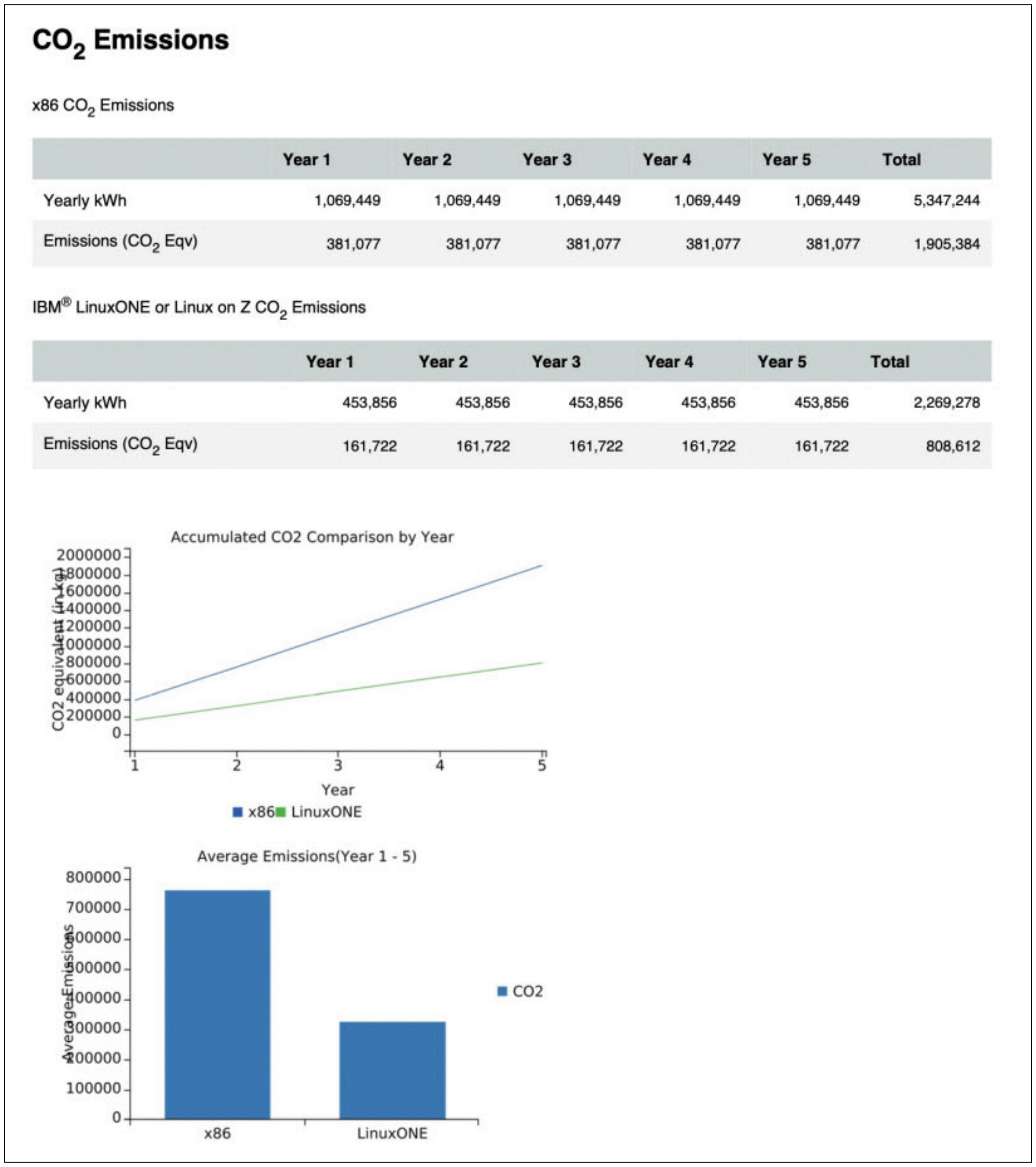
IBM® LinuxONE or Linux on Z model	Type of servers	Workload	# of IBM® LinuxONE or Linux on Z systems	# of IBM® LinuxONE or Linux on Z production cores	# of IBM® LinuxONE or Linux on Z non-production cores[3]	Total DR servers[2]	Total DR cores[2]	Total IBM® LinuxONE or Linux on Z servers	Total IBM® LinuxONE or Linux on Z cores[4]
LinuxONE 4 Emperor	19-inch frame	opendb, database, Application	2	169	127	0	0	2	296

1. For each set of production workloads are an additional 75% corresponding DevTest and Quality Assurance non-production environments that can reside within the same physical IBM® LinuxONE or Linux on Z server. A production workload environment of 10 cores, for example, is assumed to require another 7.5 cores for supporting non-production DevTest and QA work.
2. The DR environment is assumed to replicate the production environment only, so corresponding non-production workloads are not included for DR.
3. Total required IBM® LinuxONE or Linux on Z cores are rounded up to the next whole number of cores.

Figure 1-2 LinuxONE sizing based on x86 server inputs

This consolidation yields remarkable benefits:

- 58% lower energy consumption: By migrating to the LinuxONE infrastructure, energy consumption is reduced compared to the existing setup, which leads to substantial cost savings and environmental benefits, as shown in Figure 1-3.



CO_2 Emissions

x86 CO_2 Emissions

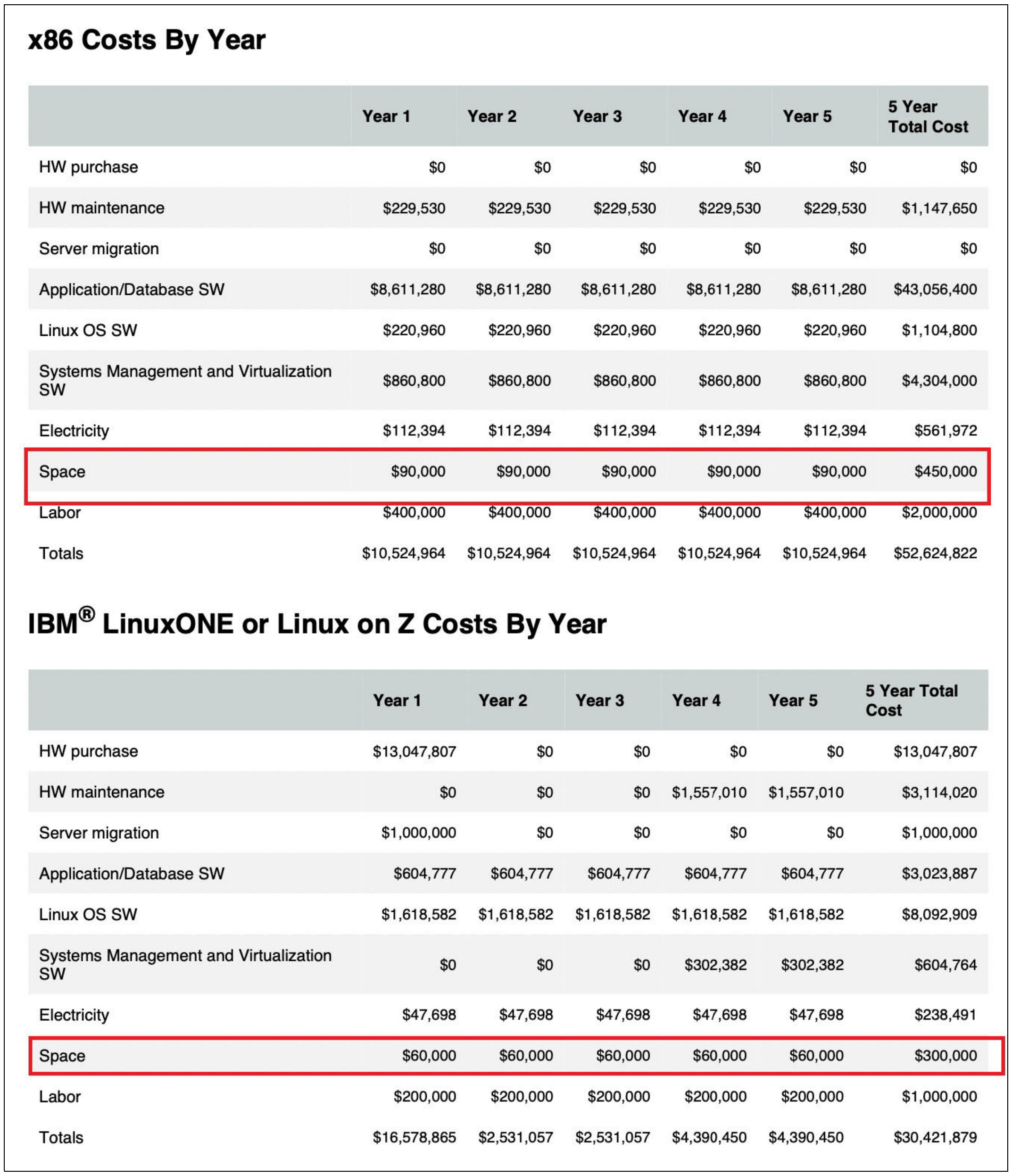
	Year 1	Year 2	Year 3	Year 4	Year 5	Total
Yearly kWh	1,069,449	1,069,449	1,069,449	1,069,449	1,069,449	5,347,244
Emissions (CO_2 Eqv)	381,077	381,077	381,077	381,077	381,077	1,905,384

IBM® LinuxONE or Linux on Z CO_2 Emissions

	Year 1	Year 2	Year 3	Year 4	Year 5	Total
Yearly kWh	453,856	453,856	453,856	453,856	453,856	2,269,278
Emissions (CO_2 Eqv)	161,722	161,722	161,722	161,722	161,722	808,612

Figure 1-3 CO2 emissions comparison

- 33% less floor space is used: The consolidation onto the LinuxONE systems leads to a more efficient usage of physical space, which requires less floor area for housing the infrastructure and optimizing data center space, as shown in Figure 1-4.[3]



x86 Costs By Year

	Year 1	Year 2	Year 3	Year 4	Year 5	5 Year Total Cost
HW purchase	$0	$0	$0	$0	$0	$0
HW maintenance	$229,530	$229,530	$229,530	$229,530	$229,530	$1,147,650
Server migration	$0	$0	$0	$0	$0	$0
Application/Database SW	$8,611,280	$8,611,280	$8,611,280	$8,611,280	$8,611,280	$43,056,400
Linux OS SW	$220,960	$220,960	$220,960	$220,960	$220,960	$1,104,800
Systems Management and Virtualization SW	$860,800	$860,800	$860,800	$860,800	$860,800	$4,304,000
Electricity	$112,394	$112,394	$112,394	$112,394	$112,394	$561,972
Space	$90,000	$90,000	$90,000	$90,000	$90,000	$450,000
Labor	$400,000	$400,000	$400,000	$400,000	$400,000	$2,000,000
Totals	$10,524,964	$10,524,964	$10,524,964	$10,524,964	$10,524,964	$52,624,822

IBM® LinuxONE or Linux on Z Costs By Year

	Year 1	Year 2	Year 3	Year 4	Year 5	5 Year Total Cost
HW purchase	$13,047,807	$0	$0	$0	$0	$13,047,807
HW maintenance	$0	$0	$0	$1,557,010	$1,557,010	$3,114,020
Server migration	$1,000,000	$0	$0	$0	$0	$1,000,000
Application/Database SW	$604,777	$604,777	$604,777	$604,777	$604,777	$3,023,887
Linux OS SW	$1,618,582	$1,618,582	$1,618,582	$1,618,582	$1,618,582	$8,092,909
Systems Management and Virtualization SW	$0	$0	$0	$302,382	$302,382	$604,764
Electricity	$47,698	$47,698	$47,698	$47,698	$47,698	$238,491
Space	$60,000	$60,000	$60,000	$60,000	$60,000	$300,000
Labor	$200,000	$200,000	$200,000	$200,000	$200,000	$1,000,000
Totals	$16,578,865	$2,531,057	$2,531,057	$4,390,450	$4,390,450	$30,421,879

Figure 1-4 Breakdown of x86 costs versus IBM LinuxONE costs by year

- 76% lower software costs: Through consolidation and the inherent efficiencies of LinuxONE, software costs are reduced, which contributes to savings over the long term (Figure 1-5 on page 19).

[3] The floor space cost is estimated to be $10,000 per rack per year.

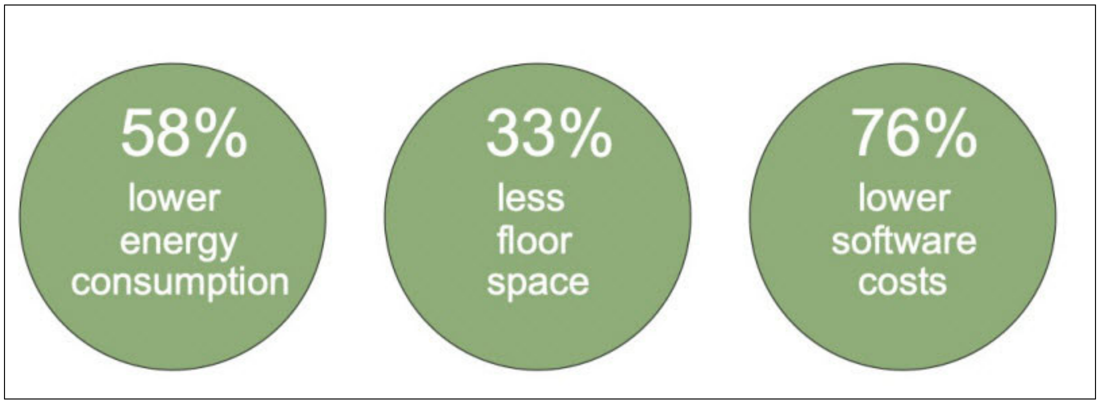


Figure 1-5 LinuxONE tangible benefits

Figure 1-6 provides a comprehensive overview of the 5-year category cost comparison highlights, which shows substantial reductions across various cost categories.

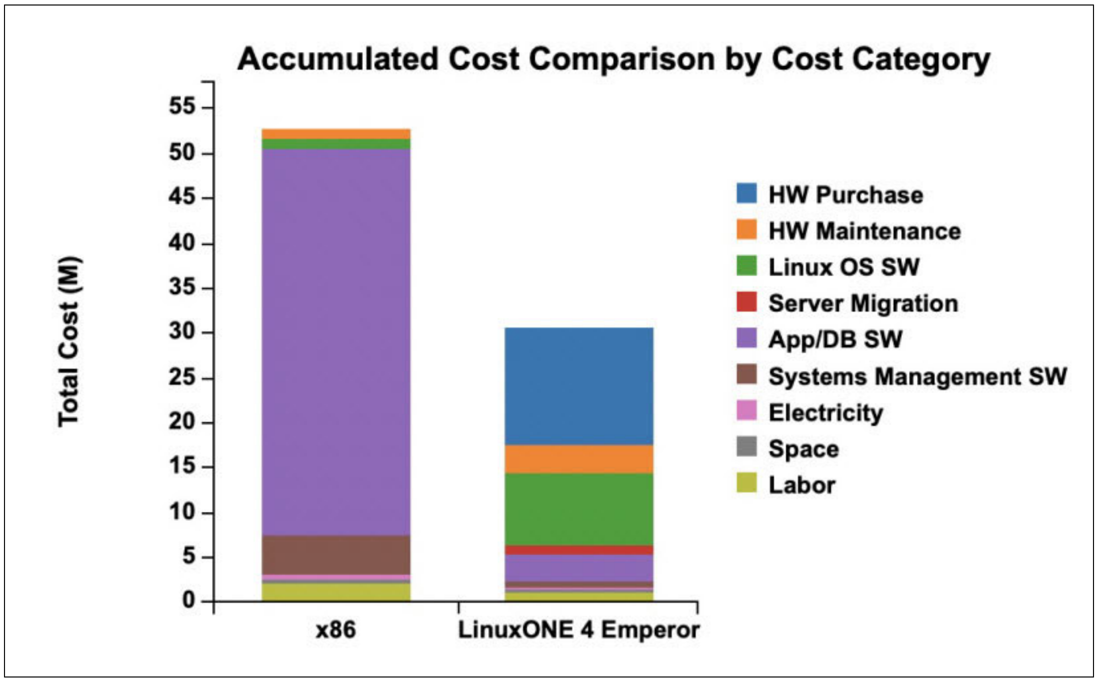


Figure 1-6 Accumulated Cost Comparison by Cost Category

1.5 Enabling a microservices architecture on IBM LinuxONE

Many organizations are trying to modernize their applications to use new approaches in the cloud. Microservices architecture is becoming the new standard for application development with more organizations moving away from traditional, monolithic applications. Breaking up large applications into a suite of smaller services allows businesses to deliver value faster, allowing their developers to work in parallel and independently.

Containers offer a convenient standard unit to encapsulate a small application component, which makes it a good infrastructure for building microservice applications. Running container-based applications on LinuxONE can support consolidation efforts because multiple containers can run in single virtual machines (VMs), which allows for fewer VMs to be created because containers provide application, memory, and data isolation.

Combining the applications and all associated dependencies allows applications to be developed and compiled to run on multiple architectures, which enable application portability and provides flexibility. By using containers, workloads can be deployed on any public, private, or hybrid cloud. Essential is interoperability to make it simpler to move workloads from one place to another.

IBM developed its Secure Service Container (SSC) technology, which is exclusive to IBM LinuxONE to provide a secure hosting appliance for container-based applications that run in hybrid cloud environments. SSC is a secure computing environment for microservices-based applications that can be deployed without any application code changes, which makes it an easily consumable solution for cloud-native development. It provides several unmatched security benefits, such as automatic pervasive encryption of data in-flight and at-rest, protection from privileged administrators, and tamper protection during installation and start time to protect against malware.

An example of how a traditional application architecture and a microservices architecture can run on the same LinuxONE platform is shown in Figure 1-7.

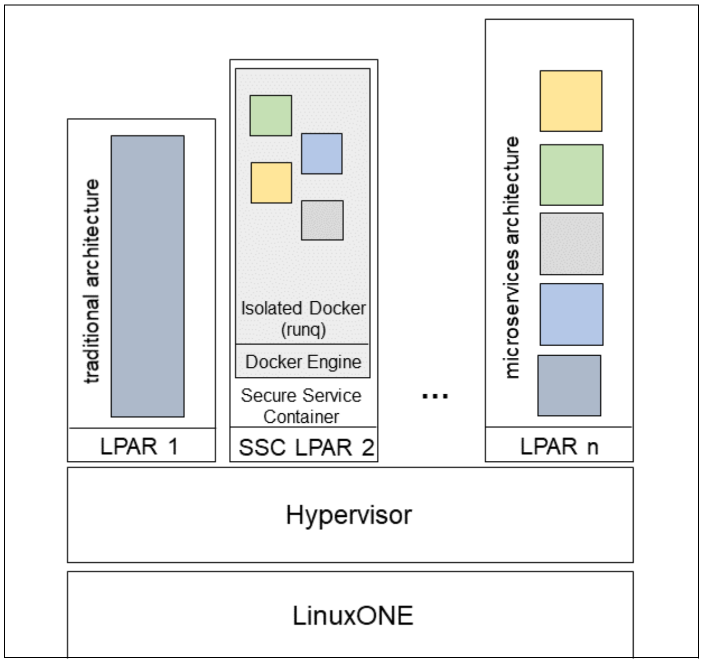


Figure 1-7 Traditional architecture, Secure Service Container, and microservices architecture

Although microservices bring many advantages, managing many services can be challenging. Orchestration tools, such as Kubernetes or Red Hat OpenShift Container Platform, can support overcoming this challenge by helping manage the deployment, placement, and lifecycle of containers.

Kubernetes is the most popular open-source container-orchestration tool, but businesses often require extra assurance that the open-source code is safe to deploy and has 24x7 support.

Red Hat OpenShift Container Platform is a container-based application platform that is built on open-source technologies and principles, such as Kubernetes, and enhances them and enables enterprises to also benefit from open source. Red Hat OpenShift Container Platform 4.2 was released for IBM LinuxONE servers in February 2020. Since then, versions have been released at the same time for x86, PPC64, and LinuxONE. With Red Hat OpenShift Container Platform, organizations can use enterprise server capabilities, such as security features, scalability, and reliability, to run cloud-native applications and accelerate their digital transformation. Application developers and operations teams can use Red Hat OpenShift Container Platform as a powerful tool to easily manage their container environment and see how they are organized through an intuitive web interface.

1.6 Cloud computing blueprint for IBM LinuxONE

IBM LinuxONE is designed to play a part in your private, public, or hybrid cloud infrastructure. A wide range of organizations, from small businesses to large institutions, are using the unique security capabilities of LinuxONE to power their private and hybrid cloud. Cloud service providers are using IBM LinuxONE technology to add next-level security and stability to its public cloud services. IBM LinuxONE also plays a critical role in the IBM public cloud offering because the Hyper Protect Service Portfolio is based on the technology.

Figure 1-8 shows the IBM LinuxONE platform for each layer of a cloud.

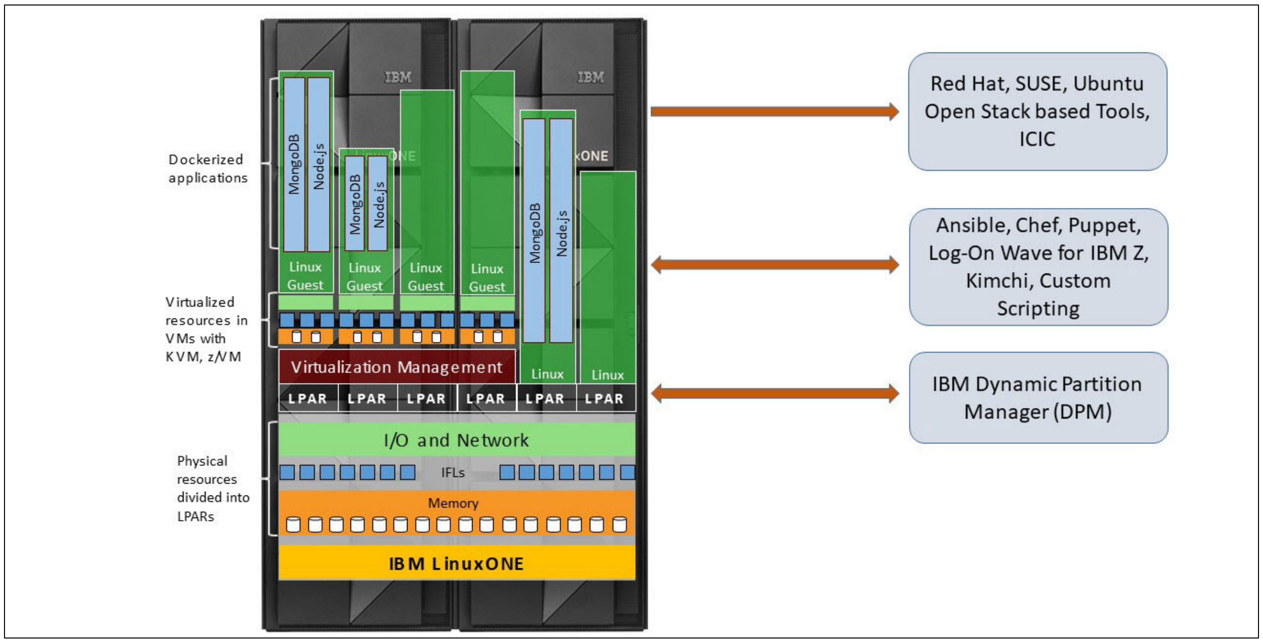

Dockerized applications
Virtualized resources in VMs with KVM, z/VM
Physical resources divided into LPARs
MongoDB
Node.js
Linux Guest
Linux
Virtualization Management
LPAR
I/O and Network
IFLs
Memory
IBM LinuxONE
Red Hat, SUSE, Ubuntu Open Stack based Tools, ICIC
Ansible, Chef, Puppet, Log-On Wave for IBM Z, Kimchi, Custom Scripting
IBM Dynamic Partition Manager (DPM)

Figure 1-8 IBM LinuxONE is "cloud ready"

1.6.1 Cloud solutions on IBM LinuxONE

Mission-critical applications are the backbone of an organization. Any downtime has a major impact to a business. These workloads require a cloud solution that has exceptional system uptime, excellent data security and privacy, and a powerful vertical scale architecture. For these workloads, IBM LinuxONE is the most securable, reliable, and scalable on-premises cloud solution.

IBM LinuxONE can provide the same agility and time to value as other cloud services, along with unparalleled enterprise qualities of service. IBM LinuxONE allows those delivering cloud services to rapidly deploy a trusted, scalable OpenStack-based Linux cloud environment that can start small and scale up massively to support thousands of virtual servers or up to two million containers on a single system.

Virtualization portfolio

Establishing cloud environments on IBM LinuxONE begins with virtualization technology. Customers have a choice of deploying z/VM, the world's first commercially available hypervisor to provide virtualization technology, or the newer industry-standard KVM. Both hypervisors enable you to bring new virtual servers online in a matter of minutes (or less) to accommodate growth in users, although each technology is designed with a different audience in mind.

The overall IBM virtualization portfolio includes the following applications for infrastructure and virtualization management:

- IBM LinuxONE hardware: IBM LinuxONE 4 LA1, IBM LinuxONE 4 LA2, and IBM LinuxONE 4 AGL:
  - Massively scalable.
  - Characterized by excellent economics and efficiencies.
  - Highly secure and available.
- z/VM:
  - Support more virtual servers than any other platform in a single footprint.
  - OpenStack support.
- KVM for IBM LinuxONE:
  - Provides a choice for clients who want open virtualization while taking advantage of the robustness, scalability, and security of the LinuxONE platform.
  - The standard interfaces that it provides allows for integration into an existing infrastructure.
- Linux for IBM LinuxONE

  Distributions are available from RHEL, SUSE Linux Enterprise Servers, and Ubuntu.
- IBM DPM

  A tool for LinuxONE configuration and setup to simplify and speed deployment of Linux servers by using only the Hardware Management Console (HMC).

Cloud solutions on IBM LinuxONE

To provide cloud management capability, z/VM and KVM are OpenStack-enabled, which is the industry standard for ubiquitous cloud computing platforms. Applications that use the OpenStack application programming interfaces (APIs) are supported on both hypervisors.

IBM Cloud Infrastructure Center (CIC) is an advanced infrastructure management offering, including on-premises cloud deployments of IBM z/VM-based and KVM-based Linux VMs on IBM LinuxONE. It is an industry-proven turn-key Infrastructure-as-a-Service (IaaS) solution that provides a consistent, industry-standard user experience to define, instantiate, and manage the lifecycle of virtual infrastructure, deployment of images (operating system and applications), and policies to maximize resource utilization.

For more information about how to install, configure, and use the IBM CIC, see the IBM CIC documentation, which is available at IBM Documentation.

IBM CIC delivers the following capabilities:

- Easy provisioning of VM instances into an on-premises cloud by way of a self-service portal.
- Infrastructure provisioning that can be confined by workflow-driven policies.
- Automated configuration of I/O and network resources.
- Image management that includes VM image capture, catalog, and deployment.
- Easy integration into higher-level cloud automation and orchestration tools.
- Federation of an on-premises cloud with OpenStack clouds by using OpenStack compatible APIs that establish a multi-region cloud.
- Manage existing VMs that are not created by compute nodes with the onboarding feature.

The IBM CIC architecture on z/VM is shown in Figure 1-9.

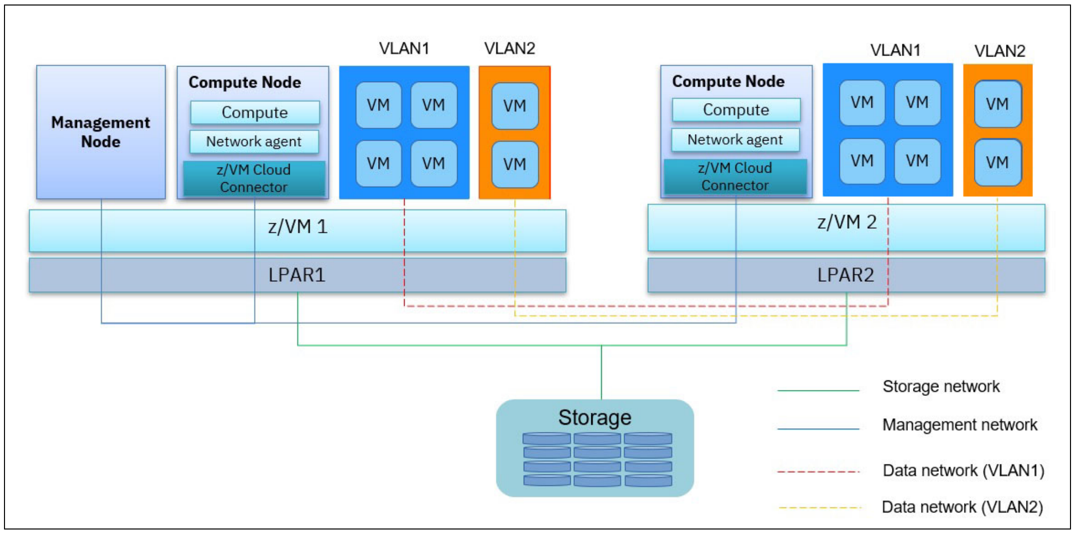


Figure 1-9 IBM Cloud Infrastructure Center Architecture on z/VM

Consider the following points regarding the IBM CIC Architecture on z/VM that is shown in Figure 1-10:

- Only one management node must be set up for managing the entire z/VM Cloud infrastructure.
- For each managed z/VM, one compute node is required.
- The management node can be in the same z/VM instance with one of the compute nodes, but they must be on different Linux VMs of that z/VM.

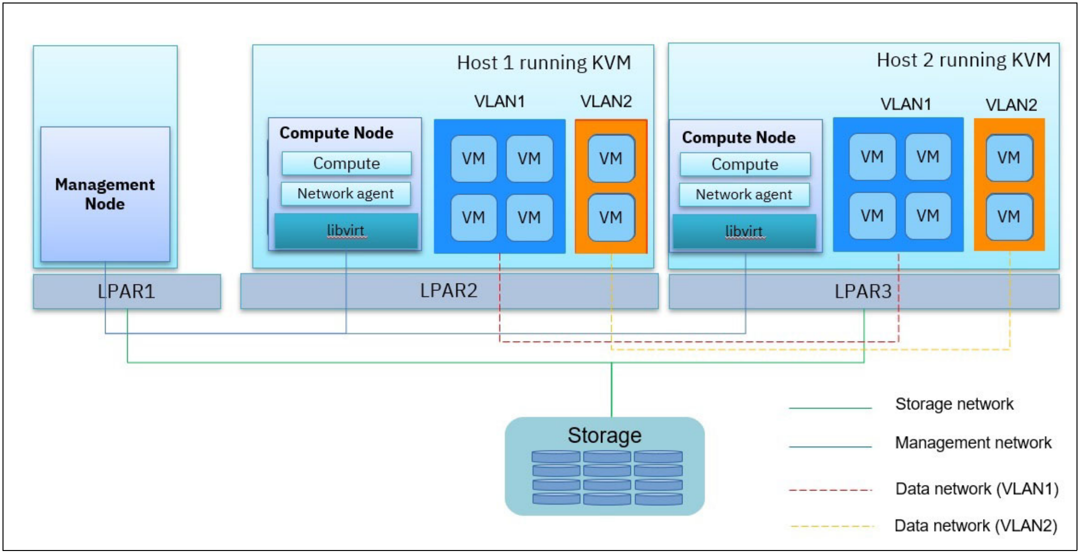


Figure 1-10 IBM Cloud Infrastructure Center Architecture on KVM

Consider the following points regarding the IBM CIC Architecture on KVM that is shown in Figure 1-10:

- Only one management node must be set up for managing the entire KVM Cloud infrastructure.
- For each managed KVM, one compute node is required.
- The management node can be on the same KVM instance with one of the compute nodes, but it is a best practice to separate the management compute nodes into different LPARs.

Other industry OpenStack-based cloud management solutions can also manage LinuxONE, including (but not limited to) the VMware vRealize Automation product.

Red Hat OpenShift Container Platform on IBM LinuxONE

As described in 1.4, “Financial benefits of a migration” on page 13, IBM announced the availability of Red Hat OpenShift Container Platform on IBM LinuxONE in 2020, making the integration of applications running on IBM LinuxONE with applications running elsewhere in the cloud simpler. Red Hat OpenShift Container Platform supports cloud-native applications that are built once and deployed anywhere. Application developers and operations teams use Red Hat OpenShift to easily and efficiently manage their container environment.

The minimum Red Hat OpenShift architecture consists of five Linux guests (bootstrap, three control nodes, and one worker node) that are deployed on top of IBM z/VM or KVM. Customers that use Red Hat OpenShift on IBM LinuxONE can use the IBM CIC to manage the underlying cluster infrastructure or to automatically deploy a cluster.

2 Analyzing and understanding

This chapter outlines some of the points that you must consider before migrating to Linux on IBM LinuxONE. It also provides a description of total cost of ownership (TCO) and helps analyze which workloads would make good candidates for migration. Also, it describes the financial benefits of migration.

This chapter includes the following topics:

2.1 Total cost of ownership analysis

The complexity of environments usually determines the magnitude of these costs.

Traditional TCO calculations often focus on readily quantifiable costs:

- Direct costs: Initial purchase price, hardware and middleware expenses, licensing fees, maintenance, and support contracts.
- Indirect costs: Training, downtime due to outages, productivity losses, potential opportunity costs, and sunk costs.

Failing to integrate security, privacy, and sustainability into your TCO analysis leads to an incomplete picture of potential risks and long-term costs:

- Security
  - Compromised systems, stolen data, and network infiltration carry massive consequences, which include direct remediation costs, legal fines, reputation damage, and lost business opportunities.
  - Proactive security measures like vulnerability assessments, penetration testing, employee training, and robust security systems reduce exposure but are often overlooked in basic TCO calculations.
  - Including security as a foundational pillar in TCO reveals its true value as an investment that protects against potentially crippling financial losses.
- Privacy
  - Stringent regulations like the GDPR mandate strict adherence to data protection principles. Failure to comply results in substantial fines and legal repercussions.
  - Consumers are increasingly privacy-conscious. Mishandling personal data can seriously damage an organization's reputation, leading to erosion of customer trust and market share.
  - Investing in privacy controls, data governance mechanisms, and secure data handling practices demonstrates good corporate citizenship and mitigates long-term risks.
- Sustainability
  - Technology has a significant environmental footprint. E-waste, energy consumption, and carbon emissions contribute to ecological damage and climate change.
  - Energy-efficient hardware, environmentally conscious disposal practices, and sustainable procurement strategies might incur upfront costs but reduce long-term operational expenses.
  - Consumer and stakeholder expectations are shifting toward ethical and sustainable business practices. Incorporating sustainability initiatives can enhance brand perception and align with corporate values.

A holistic TCO calculation that integrates security, privacy, and sustainability paints a far more accurate picture of an asset's true lifetime cost.

The business case to support a migration to IBM LinuxONE is focused on cost savings, security, and sustainability that are provided by server consolidation to the IBM LinuxONE platform and an overall simplification of the distributed environment. Here are a couple of examples:

- A bank in Vietnam decided to use IBM LinuxONE to power a bold new approach to banking to deliver rapid and reliable performance for new customer-centric services and scale seamlessly to meet fast-growing demands. The enterprise moved 21 production applications to IBM LinuxONE in 90 days. They realized a 40% TCO reduction with an architecture for large and variable workloads, which increased transactions 4x during peak season.
- A national weather service that provides weather forecasts for the public, for government, and for businesses in many sectors creates more than 3,000 tailored forecasts and briefings each day, and conducts weather- and climate-related research. This service uses post-processing systems to tailor its weather forecasts for specific clients' needs. The requirement to have the applications run continuously and deliver services every day is critical to its brand and its reputation. Running these systems on a distributed Linux infrastructure became complex and expensive, so this weather service decided to migrate suitable candidates from its distributed Linux landscape onto IBM LinuxONE.

2.2 Migration planning

The migration process requires several steps, and anticipating how much time is needed for each step is crucial.

Here are the phases of migration at the highest level:

- Planning and assessment: After you decide what will be migrated and how, the plan must specify the time, risks, and owner for each migration task and might include, but is not limited to, the following items:
  - Goals and scope definition.
  - Inventory and assessment.
  - Feasibility study.
  - Target environment design.
- Proof of concept (POC) and Test: Use a POC to check the compatibilities between the x86 and IBM LinuxONE environment, and give special focus to performance.
- Education: The technical staff needs the correct skills to work on an IBM LinuxONE migration and maintain the new environment.
- Build Environment: In this phase, the new infrastructure is readied for migration.
- Implementation: Communication between stakeholders is important during this process. All involved people must know of and approve the migration, and receive follow-up reporting on the progress. The migration execution might include the following items:
  - Pilot migration.
  - Phased migration.
  - Data migration.
  - Application deployment and configuration.

- Post-migration: After implementation, create documentation that further references the project and documents all necessary maintenance and care procedures. Also, the project manager must have a signed acceptance agreement. Post-migration might include the following items:
  - Validation and testing.
  - Performance monitoring.
  - Optimization and fine-tuning.
  - Decommissioning of old systems.
- More considerations:
  - Develop a rollback plan in case of unforeseen issues during the migration process.
  - Help ensure data security throughout the migration process and implement the appropriate security measures in the new LinuxONE environment.
  - You can use support that is available from IBM and other vendors that are involved in the migration process.

Assume that the POC, associated testing, and final implementations are the most time-consuming parts of a migration.

2.3 Choosing workloads to migrate

When you decide to migrate and consolidate, the next step is to examine which workloads would be good candidates to be migrated.

Consider these variables, among others:

- Associated costs
- Application complexity
- Service-level agreements (SLAs)
- Skills and abilities of your support staff

Start with a fairly simple application that has a low SLA and a staff that has the associated skills.

For applications developed within the company, help ensure that you have the source code available. Regarding the operating system platform, even a workload from a different platform can be migrated, but start with servers running Linux. This process will substantially increase the success criteria of the migration effort. Applications that require close proximity to corporate data stored on IBM LinuxONE are also ideal candidates, as are applications that have high input/output (I/O) rates because I/O workloads are offloaded from general-purpose processors onto specialized I/O processors.

IBM LinuxONE Emperor 4 has a powerful processor with a clock speed of 5.2 GHz. The IBM LinuxONE Rockhopper 4 scales up to 68 x 4.6 GHz configurable Telum cores. Because IBM LinuxONE is designed to concurrently run disparate workloads, some workloads that require dedicated physical processors that are designed to run at high sustained CPU utilization rates might not be optimal candidates for migration to a virtualized Linux environment. This is because workloads that require dedicated processors do not take advantage of the virtualization and hardware sharing capabilities. An example of such an application might include video rendering, which requires specialized video hardware.

Chapter 5, “Migration analysis” on page 75 provides an in-depth analysis of the process of determining the most appropriate applications to migrate to an IBM LinuxONE environment.

2.4 Analysis of how to size workloads for migration

First, determine the expected consolidation ratio for a specific workload type to determine the theoretical maximum number of servers that can be consolidated.

Sizing the theoretical maximum number of servers that can be consolidated requires considering many factors, both technical and non-technical. Here are some key aspects to consider:

- Technical factors:
  - Server resources (see Table 2-1.):
    - CPU: The combined processing power of the target server (where workloads are consolidated) must be sufficient to handle the aggregate workload of the consolidated servers, which includes the processor speed (or speeds) of the servers and the average of CPU utilization of these servers.
    - Memory: The target server needs adequate memory capacity to accommodate the combined memory requirements of all consolidated workloads.
    - Storage: The storage capacity and performance of the target server should be sufficient to store and access the data from all consolidated servers efficiently.
    - Network bandwidth: The network infrastructure must have enough bandwidth to handle the increased traffic that is generated by consolidated workloads.

Table 2-1 Server resource sizing example

Server resources	
Description	**Example**
CPU	► Individual servers: – Server A: 4 cores at 2.5 GHz – Server B: 2 cores at 3 GHz – Server C: 2 cores at 3 GHz ► Target server: 16 cores at 5.0 GHz
Memory	► Individual servers: – Server A: 16 GB – Server B: 8 GB – Server C: 32 GB ► Target server: 64 GB
Storage	► Individual servers: – Server A: 2 TB SATA HDD – Server B: 1 TB NVMe SSD – Server C: 2 TB NVMe SSD ► Target Server: 10 TB NVMe SSD
Network bandwidth	► Individual servers: – Server A: 1 Gbps – Server B: 1 Gbps – Server C: 10 Gbps ► Required bandwidth: To be determined based on application traffic analysis

- Workload characteristics (see Table 2-2.):
  - Analyze the average and peak resource usage of individual servers to understand their processing, memory, and storage demands.
  - Help ensure that the target server supports the operating systems, applications, and middleware dependencies of the consolidated servers.
  - Different applications have varying performance needs. Analyze the required performance levels (for example, response times, bandwidth, and SLA) for each application to ensure that consolidation does not negatively impact the user experience.
  - If you use virtualization, consider the capabilities of the chosen virtualization platform and its resource impact when you calculate capacity.

Table 2-2 Workload characteristics

Workload characteristics	
Description	**Example**
Resource usage	► Server A: – CPU: 30% average with 60% peak – Memory: 80% average with 90% peak ► Server B: – CPU: 20% average with 40% peak – Memory: 50% average with 70% peak ► Server C: – CPU: 40% average with 50% peak – Memory: 50% average with 60% peak
Compatibility	► Server A: Linux Ubuntu 20.04, Application X. ► Server B: Linux Ubuntu 20.04, Application Y. ► Server C: Linux Red Hat Enterprise Linux (RHEL) 8, Application Z. ► Target Server: Needs to support Linux distributions, and Applications X, Y, and Z.
Performance requirements	► Application X: Requires subsecond response time for user interactions (less than 100 ms). ► Application Y: Can tolerate slightly higher response times (up to 500 ms). ► Application Z: Can tolerate slightly higher response times (up to 2 seconds). ► The target server must provide sufficient resources to meet these requirements for all applications.
Overhead: Consider the resource impact that is introduced by the chosen virtualization platform.	► Virtualization platform impact: 5% CPU and 10% memory. ► Adjust resource calculations for the target server to account for this impact.

- Non-technical factors:
  - Organizations must balance potential gains from consolidation with potential risks. Consolidating too many servers can increase the impact of a single point of failure (SPOF).
  - Increased server consolidation can make management and troubleshooting more complex. Assess the organization's IT staff capabilities and available tools to manage larger consolidated environments.

- Consider the organization's anticipated future growth when estimating the server capacity. You do not want to reach full capacity immediately after consolidation.
- Review software licensing terms to help ensure that they allow for consolidation and avoid potential licensing violations.

The theoretical maximum number of servers that can be consolidated does not necessarily translate to the optimal number. A thorough analysis of all these factors, along with real-world testing and performance monitoring, helps determine a practical and sustainable consolidation strategy.

Although this process might set limits on the upper boundaries for virtualization, the efficiency of the target platform and platform hypervisor might reduce consolidation ratios. In practice, service levels are often the determining factor.

IBM offers a tool to help Chief Information Officers (CIOs) in determining the IBM LinuxONE resources that are required to consolidate distributed workloads. It is a self-help web tool that is named *IBM Smarter Computing Workload Simulator* that gives a fast way to view areas of potential savings and efficiency through the lens of IBM Smarter Computing systems and technologies.

For more information, see The IBM Smarter Computer Workload Simulator.

Also, IBM Competitive Project Office can support you with a more detailed analysis of your expected consolidation ratio. Contact your IBM representative to start such a study.

Important: Other factors must be considered to get a complete TCO, including floor space, energy savings, scalability, security, and outages. For a more accurate sizing study, contact your IBM representative.

2.5 Project definition

Regardless of the size or scope of the migration, the stakeholders must start with a detailed migration plan. The success of a server consolidation project depends on several factors: Clear project definition, preparation for organizational change, sponsorship, and a good strategy. The plan gives stakeholders an understanding of the risks, savings, and deliverables, and provides an overview of the migration.

The stakeholders discuss the project plan and produce the principal goals of the migration plan. Documents must be created that represent the strategy that will be used to accomplish the migration goals. Of equal importance is defining the project timeline, considering potential delays, and identifying processes that can be parallelized in case any stop is found in one or more branches of the original migration plan. For a project like this, a project manager should be considered to coordinate this variation from the original plan and solve the potential bottlenecks in the process to make all the parts in the project aligned and coordinated.

When defining the project, consider the availability of all resources, especially human resources. For several processes of the migration, SMEs or experts might be needed. These technical specialists will not be necessary throughout the whole migration process but only for specific tasks that request deeper skills in a certain topic. Accounting for these roles is key to good project planning and following the expected timeline to avoid extra delays.

2.6 Planning checklists

Planning checklists are used to identify the hardware and software requirements, migration tasks, and project deliverables during a migration project. Although the approach and parameters for a migration planning checklist can vary somewhat from project to project or between organizations, the foundation of an effective planning checklist is similar to the generic checklists that are provided in this chapter.

The checklists that are shown in this chapter are created specifically for IBM LinuxONE. You can use this chapter along with *Consolidation Planning Workbook: Practical Migration from x86 to IBM LinuxONE*, REDP-5433, which provide you with more information about the planning worksheets, including blank copies to guide you through the process and a sample project plan.

2.6.1 Software products and tools worksheet

The software product and tools worksheet that is shown in Table 2-3 lists all the products and tools that are used in the source operating environment.

It provides a space where you can record whether the same or similar products and tools are available on the target IBM LinuxONE operating environment.

Table 2-3 Software products and tools worksheet

Application name: **Supporting business area:** **Technical application owner:**				
Name	**Version**	**Vendor/Source website**	**License type**	**IBM LinuxONE**

2.6.2 Application features worksheet

The application features worksheet dives one level deeper into the software product and tools worksheet, where each product or tool is drilled down to their features level. Any configurable requirements, such as specific accounts, groups, programs, or jobs that are required for an application to function should be included in this worksheet. Scenarios exist in which the same product does not offer the same features on all platforms. These details should be noted in this worksheet, as shown in Table 2-4.

Table 2-4 Application features worksheet

Application name: **Supporting business area:** **Technical application owner:**		
	Source (x86)	**Target (IBM LinuxONE)**
OS Name and Version		
Required Users		
Required Groups		
Required privileges (SUDO)		
Observations		
	Backup Solutions	
Operating System		
Database		
Hypervisor		
Container platform		
Other		
	Application-specific Dependencies	
Operating System Packages		
External Programs / Libraries		
Cron jobs		

Each product or tool that is listed in the product worksheet must be analyzed. All the parameters, dependencies, and optimization options must be considered in the source operating environment. The planning team must then assess whether the same kind of features or build options are available in the target operating environment.

If the same feature is not available with the same tools or product in the target environment, the team can assess other options:

- Obtain a similar feature by linking other product or tools in the target operating environment.
- Make note of the parameters available in the same tool in the target operating environment that can be combined to give the same characteristics as in the source environment.

- If the products or product options are fully incompatible or unavailable, replacing that part of the application stack would be a useful approach to minimize the effort involved in migration. But care must be taken to help ensure that all the features and parameters that are offered by the product in the source environment are also available in the assessing product for the target environment.
- Often, the optimization features or performance options for a product are only available for that specific platform. In such cases, the optimization features and performance options must be changed to offer the same characteristics to the target environment.

When completing the application features worksheet, verify whether changing parameters or options in the target operating environment has any side effects on the application or other tools that are used for application implementation.

If all the checklists are properly analyzed and applied, then the tools, products, and their implementation differences are accounted for in the actual migration. For example, if one or more tools must be updated, replaced, or otherwise reconfigured, this situation might impact the migration project plan, which might potentially add extra tasks, effort, time that is spent, costs, and extra technical specialists. Finding experts that are experienced in this same transition or migration project can be challenging. This identification and analysis process is critical, and proper execution reduces the risks so that the migration works smoothly.

2.6.3 Application flows worksheet

The source application to be migrated can be in the center of a complex process. The application can be interconnected with many other applications, inputs, outputs, and interfaces. For this reason, you must prepare a planning document that lists the resources that the source application must provide and all the services that it is providing. Table 2-5 lists examples of the resources that are required of some applications.

Make the descriptions as detailed as possible by providing the physical location, server hostname, IP address, network information, software product that is used, focal point, and any other information that you believe important to register about the services. The target environment must have the same infrastructure available to it as is available in the source environment.

Table 2-5 Application flows worksheet example

Source hostname	Source IP	Target hostname	Target IP	Protocol	Port
`lnxone-was1.pbm.ihost.com`	129.40.23.153	`lnxone-mq.pbm.ihost.com`	10.10.8.5	TCP	1414
`lnxone-was2.pbm.ihost.com`	129.40.23.154	`lnxone-mq.pbm.ihost.com`	129.40.15.15	TCP	1414
`lnxone-was1.pbm.ihost.com`	129.40.23.153	`lnxone-db2.pbm.ihost.com`	129.40.15.15	TCP	50000
`lnxone-was2.pbm.ihost.com`	129.40.23.154	`lnxone-db2.pbm.ihost.com`	129.40.15.15	TCP	50000
`lnxone-node.pbm.ihost.com`	10.10.8.6	`lnxone-mq.pbm.ihost.com`	10.10.8.5	TCP	9443

2.6.4 Training checklist

A critical element in achieving successful migrations is to help ensure that the migration team has skills in the new technology that will be migrated. Help ensure that a training checklist is in place during the planning process. Identify the people to train, the skills that need to be imparted, and a timetable of when the training must be completed to help ensure that staff are trained at the correct time.

Help ensure that training is a viable option or whether extra roles must be assigned to the team. If the process is performed by recruiting a specialist, the time that is needed to recruit them must also be accounted for.

A best practice is to find user groups and information about other customers that have gone through a similar process to identify possible difficulties, and gather as much information as possible to identify and re-mediate any skills challenges.

2.6.5 Hardware planning worksheet

The hardware planning worksheet lists the hardware resources that you must consider during a migration project. The source system resources are examined and mapped to a similar or more advanced technology that is on IBM LinuxONE. An example of how to complete this process is shown in Table 2-6.

Table 2-6 Example of a completed hardware planning worksheet

Hardware planning worksheet			
Server name:			
Resource	**Source**	**Destination**	**Observation**
Volume groups (VGs): VG OS: 20 GB VG DB: 150 GB VG DATA: 80 GB			
Number of CPUs	4	2	Real to Virtual
System memory (in GB)	8	8	
OS swap memory (in GB)	4	4x 512 M	Disk to VDisk
Network connection[a]			
Connection description	Gigabit Ethernet	Gigabit Ethernet	
Connection type	Gigabit Ethernet	vSwitch/GbE	
IP address/netmask	129.40.19.88/24	129.40.23.153/24	
Virtual local area network (VLAN) number: vSwitch	2	2: vSwitch1	
Disk resource[b]			
OS file system	/ : 30 : Ext3	/ : 10 :Ext4	Root
Mount point: Size (in GB): Type		/home : 3 :Ext4 VG OS	LV
Mount point: Size (in GB): Type		/opt : 5 :Ext4 VG OS	LV
Mount point: Size (in GB): Type		/tmp : 5 :Ext4 VG OS	LV

Hardware planning worksheet			
Server name:			
Mount point: size (in GB): Type		/var : 1 :Ext4 VG OS	LV
Data file system			
Mount point: Size (in GB): Type	/DB : 100 : Ext3	/DB:100:XFS VG DB	LV
Mount point: Size (in GB): Type	/WAS : 50 : Ext3	/WAS:50:XFS VG DATA	LV
Custom file system			
Mount point: Size (in GB): Type		/MGM:10:BTRFS VG DATA	LV

a. The following network connections are available for IBM LinuxONE: Ethernet/QETH, Open vSwitch (OVS), IBM HiperSockets, and a direct OSA-Express connection.
b. Logical Volume Manager (LVM) provides storage management flexibility and reduced downtime with online resizing of logical volumes (LVs).

2.6.6 Security and privacy worksheet

Whether you're managing personal data or protecting critical business assets, implementing robust security and privacy practices is essential. The comprehensive worksheet that is shown in Table 2-7 serves as a tool to assess your current approach and identify areas for improvement:

- Assess the effectiveness of your current security measures and identify any potential weaknesses.
- Pinpoint specific areas where your security and privacy practices might be strengthened.
- Formulate a roadmap for addressing identified shortcomings and bolstering your overall data protection strategy.

These details should be noted in this worksheet, which is shown in Table 2-7.

Table 2-7 Security and privacy worksheet

Security worksheet			
Control	**Description**	**Implemented (Yes/No)**	**Notes**
Physical security			
Secure physical access to servers and network equipment	For example, locked doors and security cameras		
Regular security checks of physical infrastructure			
Inventory of equipment and software			
Disposal of outdated or unused equipment in a secure manner			
Data security			
Strong password policies for user accounts	For example, minimum length and complexity requirements		

Security worksheet			
User access controls	For example, least privilege principle and role-based access		
Encryption of sensitive data at rest and in transit	For example, procedures for identifying, containing, and recovering from security incidents		
Vulnerability management program	For example, regular patching of software, and vulnerability scanning		
Incident response plan			
Regular backups of data with off-site storage			
Data loss prevention (DLP) controls	For example, to prevent unauthorized data transfer		
Network security			
Firewall to control network traffic			
Intrusion detection/prevention system (IDS/IPS)			
Secure configuration of network devices			
Monitoring of network activity for suspicious behavior			
Data collection and use			
Do you have a privacy policy that clearly explains what data you collect, how it is used, and with whom it is shared?			
Do you obtain user consent before collecting personal data?			
Do you provide users with the ability to access and update their personal data?			
Do you have procedures in place to securely dispose of personal data when it is no longer needed?			
Third-party data sharing			
Do you share user data with any third parties? If so, who are they and what data do you share?			
Do you have contracts in place with third parties that help ensure they protect user data in accordance with applicable privacy laws?			
Do you allow users to opt out of having their data shared with third parties?			

By using this worksheet, you can leverage security and privacy features that are provided by IBM LinuxONE such as pervasive encryption, technology-enforced (rather than administrator-enforced) isolation of workloads at massive scale, the usage of Trusted Platform Module, compliance with privacy regulations, and protecting sensitive data with quantum-safe cryptography.

2.6.7 Sustainability worksheet

Use the worksheet that is shown in Table 2-8 to assess your current server environment and plan for sustainable workload consolidation on IBM LinuxONE.

Table 2-8 Sustainability worksheet

Task	Completed (Yes/No)	Notes
Planning and assessment		
Analyzed current server resource utilization		
Assessed total server environment energy consumption		
Estimated future workload growth		
Implementation and optimization		
Selected energy-efficient LinuxONE models		
Implemented virtualization technologies		
Established plan for monitoring and optimizing resource usage		
Identified and partnered with e-waste disposal company		
Reporting and review		
Established method for tracking and reporting energy savings		
Scheduled regular review and update of sustainability plan		

Use this space to document any additional considerations or challenges that are related to sustainable workload consolidation.

Note: Consider seeking expert advice from IBM representatives or sustainability specialists for guidance throughout the planning and implementation process.

You can use the following tools to track the energy consumption for your existing x86 servers:

- Operating system-level tools:
  - Microsoft Windows:
    - Performance Monitor is a built-in tool that provides detailed power consumption data for various components.
    - Power Efficiency Diagnostics Report is a tool that analyzes system settings and provides recommendations for energy optimization.
  - Linux: Use a tool like `powertop` to monitor and analyze real-time power consumption, and get insights into resource usage and optimization opportunities.[1]
- Hardware-based tools:
  - Management Information Base (MIB): Embedded in some servers. It enables access to detailed power consumption data through Simple Network Management Protocol (SNMP) tools.
  - Intelligent Platform Management Interface (IPMI): A hardware interface that provides detailed power consumption data and remote management capabilities for some servers.
- Third-party software: Several vendor-specific or general-purpose IT monitoring solutions offer advanced power consumption tracking capabilities, which are often integrated with other system health and performance metrics.

You can use the following tools to track energy consumption for your IBM LinuxONE servers:

- Hardware Management Console (HMC): With IBM LinuxONE Rockhopper 4, a new Environmental Dashboard experience replaced the Energy Efficiency Statistics task on the HMC. This new experience has the following capabilities:
  - Monitoring the system and partition power consumption
  - Selecting time ranges and metrics to view historical data
  - The ability to chart and analyze the data
  - Ability to export the data for use elsewhere
- Optimizing for sustainability with IBM LinuxONE: Proving the energy efficiency of consolidation on IBM LinuxONE is important because it is core to the IBM sustainability software strategy, which is to advise, inform, and connect the boardroom to every layer of operations. By using IBM Instana™, Grafana, and IBM Envizi™, you can show the benefits of optimizing the power envelope of your workloads by migrating them from x86 to IBM LinuxONE.

  For more information about the IBM Envizi Environmental, Social, and Governance (ESG) Suite, see IBM Envizi ESG Suite.
- TCO and CO2e Calculator: A powerful online (public) calculator that estimates the TCO of a specific system, which includes its environmental (CO2e) aspects.
- Power Estimation Tool: A tool that estimates the power consumption, power cord phase currents, system weight, airflow, and exhaust air temperature for the specified configuration.

[1] https://access.RedHat.com/documentation/fr-fr/red_hat_enterprise_linux/8/html/monitoring_and_managing_system_status_and_performance/managing-power-consumption-with-powertop_monitoring-and-managing-system-status-and-performance

3

Virtualization concepts

Virtualization is a highly prized capability in the modern computing environment. Virtualization on LinuxONE offers industry-leading and large-scale cloud and IT optimization capabilities to drive down the costs of managing and maintaining the tremendous proliferation of servers in today's technology infrastructures.

This chapter provides helpful information about virtualization, particularly to compare and contrast the virtualization concepts of IBM LinuxONE with ones that are commonly used by x86 distributed systems. The two have many concepts in common, but other concepts are different. This brief comparison provides terminology, vocabulary, and diagrams that are helpful when migrating workloads to LinuxONE.

This chapter includes the following topics:

3.1 Demand for virtualization

As the computing environment grows in size and complexity, the sprawling infrastructure becomes more difficult to manage. As more physical servers are added to the environment, the resources, such as CPU, memory, and disk are too easily wasted and cannot be efficiently used.

Virtualization turns physical hardware into logical resources that can be shared, shifted, and reused. One of the most highly prized features of virtualization is dynamically sharing or dedicating more virtual resources, such as CPU, RAM, and disk, to a virtual guest while the virtual guest is running. This process greatly eases the system administration tasks of scaling the supply of services to meet demand.

Virtualization allows a single physical server to host numerous logical servers. The servers share the physical resources to allow all the guest servers to accomplish more than the single physical server can on its own, while maximizing the effective use of the physical resources. In such a virtual environment, the physical server is commonly called the “host” system and the logical servers are known as “guests.” Although software solutions in the industry use variations of these terms, this publication uses the terms “host” and “guest” as defined above.

Systems administrators rely on virtualization to ease and facilitate the complex work of managing increasingly complex environments. IT managers look to virtualization to address the ever-increasing demand for more computing power from customers while accommodating shrinking IT budgets.

The growing number of physical servers also increases the amount of electric power and air conditioning that is consumed in the data center. Virtualization helps to reduce the amount of electricity that is used, which reduces costs. The aspirations of a “green” data center can similarly be met in part by using virtualization.

Virtualization is widely adopted across various industries and organizations. The rise of cloud computing heavily relies on virtualization technologies to provide on-demand infrastructure and service, which further highlights its significance in the modern IT landscape.

Despite its more recent hype, virtualization has existed in advanced computing systems for some time. The conception of virtualization began in the late 1960s when IBM introduced the Control Program (CP)-67. This innovation grew to become a defining feature of IBM hardware, including all LinuxONE systems.

For more information about how virtualization works, see What is virtualization?

3.2 Typical x86 virtualization

Figure 3-1 shows a simple but typical way that systems administrators set up virtual services in a distributed x86 environment. A physical server employs virtualization software (such as Kernel-based Virtual Machine (KVM), Hyper-V, VMware, Virtual Box, or Xen) to install and run a Linux guest.

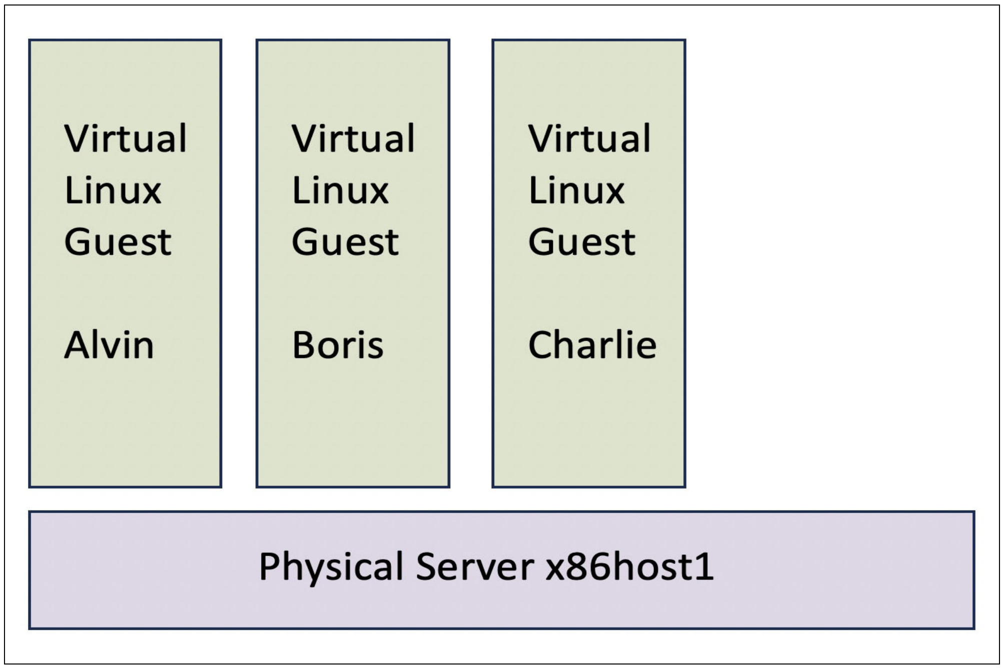

Virtual Linux Guest Alvin
Virtual Linux Guest Boris
Virtual Linux Guest Charlie
Physical Server x86host1

Figure 3-1 Typical x86 virtualization scheme

Figure 3-1 shows a physical server (hostname “x86host1”) with three separate virtual Linux guest operating systems that are contained on this physical host. The physical server has a fixed amount of CPU, RAM, and physical access to disk and network resources. The virtual guests are allocated CPU, RAM, and disk resources as a subset of what is available on the physical server, and the network resources are all equally shared by the guests and physical host.

In a typical x86 deployment of virtual services, the physical servers are deployed in pairs or trios, often called *clusters of servers*. The clusters provide for some standard level of high availability (HA) such that if one physical server fails, another takes over the running workload with negligible interruption.

Although x86 virtualization offers benefits for resource management and application isolation, it does come with some drawbacks:

- Virtualization inherently introduces overhead because the physical resources (CPU, memory, and storage) are shared between the host and guest machines, which can lead to performance degradation, especially when running resource-intensive applications.
- Coordinating timekeeping between the host and multiple virtual machines (VMs) can be challenging, which can lead to inaccuracies in applications that rely on precise timing, such as financial trading or scientific simulations.
- Security vulnerabilities in the virtualization software or its configuration can potentially expose the host system and guest machines to security breaches.

It is important to weigh these potential drawbacks against the benefits of x86 virtualization based on your specific needs and use case. If performance and security are critical, alternative solutions like IBM LinuxONE should be considered. However, for consolidating workloads, testing software, or creating isolated environments, x86 virtualization remains a widely adopted technology.

3.3 LinuxONE virtualization

Like virtualization systems that are deployed by using x86 clusters, LinuxONE accomplishes these many virtualization functions. Unlike x86, LinuxONE does so in a consolidated and comprehensive way, with multiple layers of virtualization that provide an extensive list of capabilities for the Linux guest operating system.

A LinuxONE system is divided into isolated logical partitions (LPARs), with each partition running as an independent host system with its own operating environment. This configuration means that it has its own CPUs, RAM, devices (such as disks and network connections), and other resources. LinuxONE 4 offers a wide range of configurations, starting with 16 engines and 1 TB of memory for the LinuxONE Express model, and scaling to 200 engines and a massive 48 TB of memory for the LinuxONE Multi Frame system, which enables exceptional density and hundreds of LPARs on a single server.

With LinuxONE partitions, it is routine to maintain all pre-release development of a system contained within one partition, such that anything that goes wrong in this test environment will not adversely affect anything that is running in the production environment. When development is completed, the workloads that have been developed in the test environment can be migrated to the production environment. One partition can be dedicated to development and testing activity, and a separate partition can be dedicated to the production environment.

Although LinuxONE can run dozens of partitions, and thus dozens of Linux instances, greater flexibility to virtualize further can be achieved by running an extra hypervisor within each partition. For more information about these technologies, see 3.3.3, “KVM hypervisor” on page 45, and 3.3.4, “z/VM hypervisor” on page 46.

3.3.1 Process Resource/System Manager hypervisor

The IBM Process Resource/Systems Manager (PR/SM) is a standard feature on all LinuxONE servers. It facilitates virtualization of all physical resources of a LinuxONE system into isolated partitions. PR/SM is responsible for the isolation of each partition from each other, and contains powerful tools for the finite management of each.

3.3.2 IBM Dynamic Partition Manager

The IBM Dynamic Partition Manager (DPM) provides a simple mode of operation for a LinuxONE system to easily manage and configure the system and its guests. DPM is not a new hypervisor, but rather a graphical interface that uses the existing PR/SM infrastructure.

A system administrator can easily create, modify, and manage the LinuxONE without needing to learn PR/SM and its associated components or commands. The DPM interface allows for dynamic reconfiguration of CPU, memory, and I/O resources. Wizards prompt for specific details about a system as it is being created, and automatically enter those and other values where they are required. Advanced menus offer greater control by experienced system administrators.

A system administrator can start the installation of a KVM hypervisor with DPM to simplify the deployment of Linux guests within a newly created partition.

DPM provides resource monitoring on a per-partition basis, which allowing for nearly real-time usage data and historical trends over time.

3.3.3 KVM hypervisor

The KVM hypervisor is a module that can be built into the Linux operating system. KVM enables the virtualization capabilities of a real processor to be used by a user application, which can then set up virtual resources to be used by a guest operating system. On LinuxONE, KVM enhances the capabilities of PR/SM while further protecting and isolating each VM that is established on the host.

Running Linux, extended with the KVM module, within an IBM LinuxONE partition allows multiple instances of the QEMU application, which provides emulation and virtualization to a guest operating system. Each QEMU instance runs as a separate process of the host Linux system, which separates guest instances and protects each set of virtual resources from each other and from the host system. QEMU communicates with the KVM interface to establish a guest Linux operating system as though it were running on its own private hardware.

For more information about KVM, see 3.6, “KVM hypervisor components” on page 51, and *Virtualization Cookbook for IBM Z Volume 5: KVM*, SG24-8463.

Using the earlier diagram of a typical x86 virtualization system as a model (see Figure 3-1 on page 43), a similar virtualization system as it relates to LinuxONE and KVM is shown in Figure 3-2.

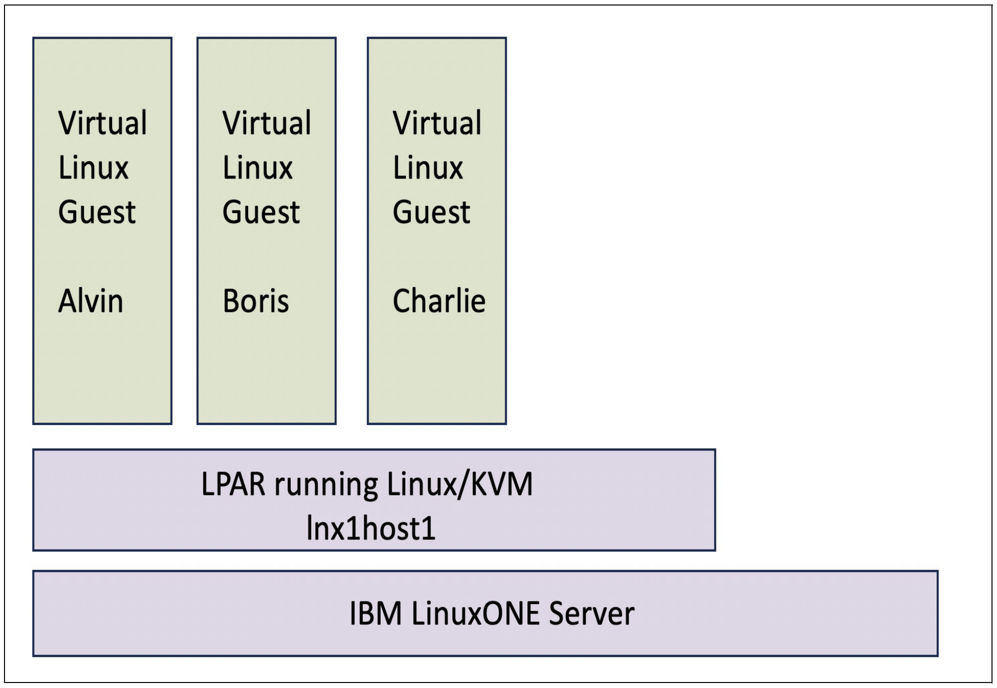

Virtual Linux Guest Alvin
Virtual Linux Guest Boris
Virtual Linux Guest Charlie
LPAR running Linux/KVM lnx1host1
IBM LinuxONE Server

Figure 3-2 Simple Linux/KVM setup with virtual Linux guests

3.3.4 z/VM hypervisor

z/VM is an operating system that facilitates and further enhances the PR/SM hypervisor. A systems administrator might likely know little about the details of PR/SM. z/VM exposes most of the features and interfaces of the PR/SM hypervisor while further protecting and isolating each VM from each other and from the physical resources.

The virtualization capabilities of z/VM provide added isolation, resource sharing, and resource management features that many systems administrators require.

For more information about z/VM, see the following resources:

- Section 3.7, “z/VM hypervisor components” on page 52
- *z/VM Version 7 Release 1 Getting Started with Linux on IBM Z*, SC24-6287
- *Introduction to the New Mainframe: z/VM Basics*, SG24-7316

Using the earlier diagram of a typical x86 virtualization system as a model (see Figure 3-1 on page 43), a similar virtualization system as it relates to LinuxONE and z/VM is shown in Figure 3-3.

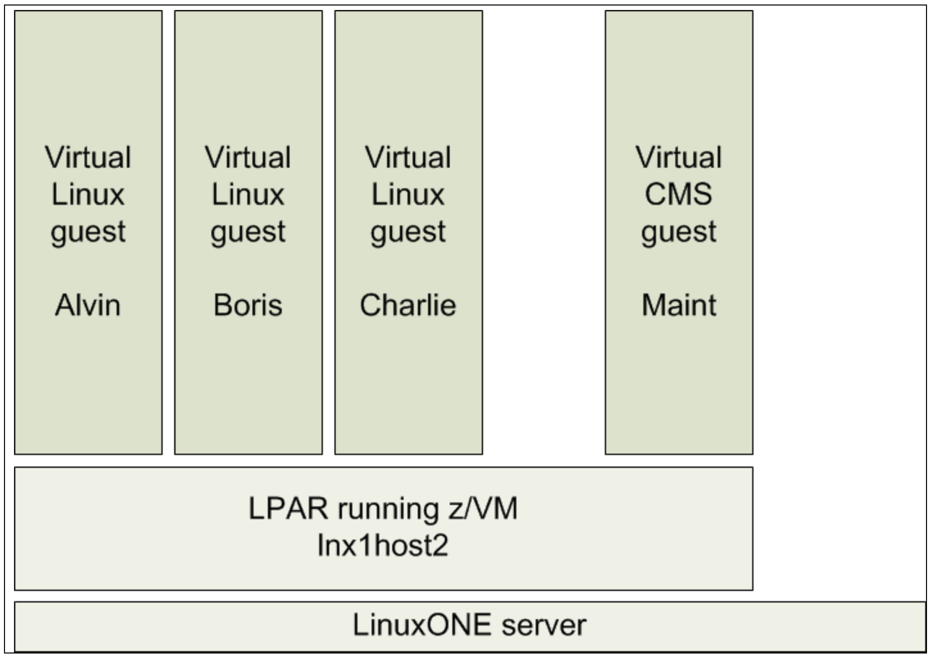

Virtual Linux guest Alvin
Virtual Linux guest Boris
Virtual Linux guest Charlie
Virtual CMS guest Maint
LPAR running z/VM lnx1host2
LinuxONE server

Figure 3-3 Simple z/VM setup with virtual Linux guests

3.4 Linux guest

Running Linux as a guest under Linux/KVM or z/VM is simple. Canonical, Red Hat, or SUSE provide Linux distributions that run on IBM LinuxONE hardware.

The work that IBM did in collaboration with these major Linux distributions provided code within the kernel and the core utilities that are tied to the kernel to facilitate the operation of the Linux kernel with LinuxONE hardware.

Figure 3-4 shows the work that IBM contributed to the Linux kernel and the Linux operating system to allow Linux to run on LinuxONE.

Note: All supported Linux distributions of Red Hat Enterprise Linux (RHEL), SUSE, and Ubuntu can run on LinuxONE. The Linux kernel alone does not make an operating system. For a Linux distribution to run on LinuxONE, the distribution must also include binutils, glibc, and other core components that are built for LinuxONE.

For this reason, the Linux distribution that is installed must be compiled specifically for the s390x architecture.

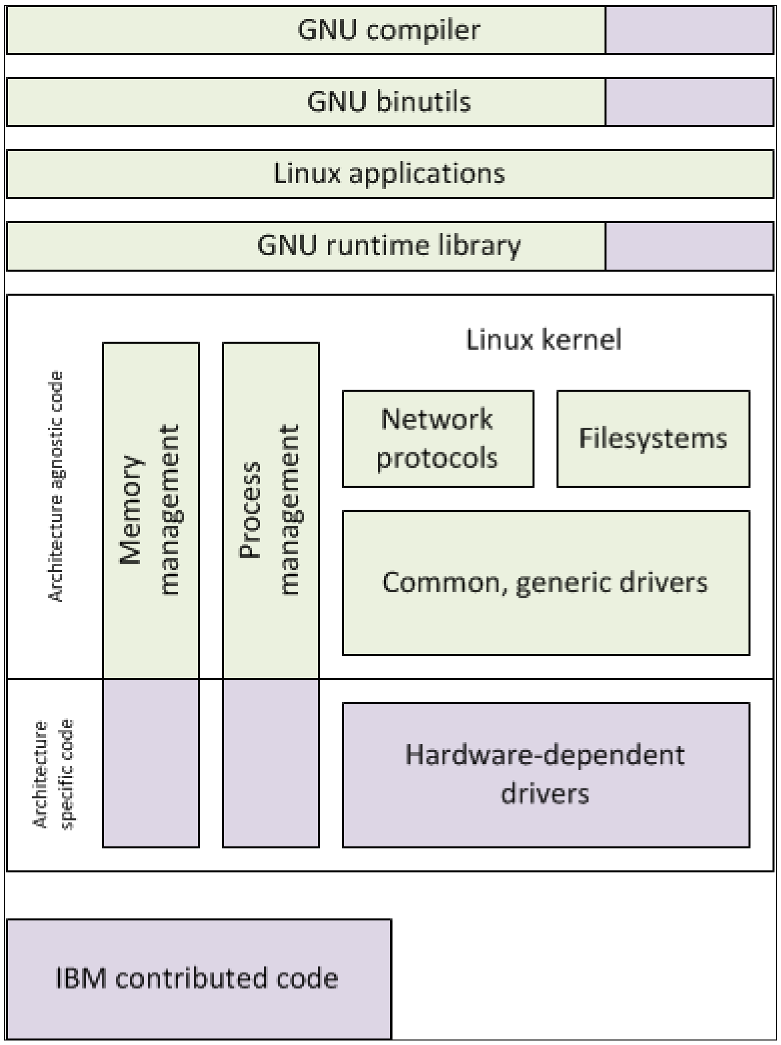


Figure 3-4 Linux kernel and core utilities characteristics on LinuxONE

Running a Linux guest on LinuxONE makes deployment of services faster. You are often able to spin up a running Linux server in a matter of minutes. Linux servers can be built, cloned, and deployed within the LinuxONE infrastructure without the pain of requisitioning, purchasing, mounting, and wiring a new physical server.

Development teams who need a new server for a proof of concept (POC) can set up and tear down a test environment repeatedly with no impact to running production systems. The LinuxONE infrastructure can use supported self-service virtualization providers to make these deployment tasks simpler.[1]

New projects that completed their development and are ready to be moved into a production environment can do so without the expense of moving or scaling physical resources. Production services can be effortlessly scaled to match the demand, and accommodate all manners of change management. LinuxONE can relocate guests, and resize, scale, and dynamically make changes. LinuxONE also has simpler management in terms of cabling, networking, or storage because they can be centralized usually in one LinuxONE box.

3.5 Guest mobility

A critical responsibility of systems administrators is to help ensure that systems are running under a vendor-supported level of the systems software, and that all maintenance and security fixes are applied. Protecting the systems from unexpected downtime and from security vulnerabilities ensures that applications are running at the latest patch release levels, and balancing loads across a diverse infrastructure are all tasks that concern systems administrators. This particularly demanding challenge is more acute in the data center, where downtime must be minimized and maintenance windows are scarce. Guest mobility helps to meet these challenges.

3.5.1 KVM guest migration

The KVM hypervisor allows a migration of guests, either while the guest is running or to pause the system to be restored at a later point. This migration is done by moving the guest state from one QEMU process to another. The guest itself is unaware of the underlying movement. Although it is possible to move a Linux guest to another LinuxONE partition, guest migration is more useful to move the guest to a wholly different LinuxONE system, perhaps because the hardware is being replaced. The migration can be performed between distinct LinuxONE systems, either in an ad hoc fashion or with a HA clustering solution that ties multiple systems together as one. In either case, this movement can be done live with guests servers active without any loss of service or connectivity, or independently if the movement is between LPARs or between different LinuxONE boxes.

The guest must have access to identical resources on both the source and destination systems to run satisfactorily. An attempt to perform a migration with a mismatch in resources fails because the guest might not behave correctly upon being started on the destination system. A migration test can be done before you attempt a live guest migration, which is a best practice because the test is a highly delicate process, especially with production guests and workloads.

In addition to moving between real servers, the KVM guest migration permits the state of a guest operating system to be saved to a file on the host. This process allows the guest operating system to be restarted later.

[1] IBM Cloud Infrastructure Center (CIC) is a private cloud Infrastructure-as-a-Service (IaaS) provider that enables Linux guests deployments and their lifecycle administration. For more information, see 3.7.4, “IBM Cloud Infrastructure Center” on page 54.

3.5.2 z/VM Single System Image and live guest relocation

z/VM Single System Image (SSI) is a clustering technology that provides multiple redundant host systems upon which virtualized guests run. Each member of the SSI cluster shares common disk pools, network devices, and user data.

Although running the SSI members on the same LinuxONE system is feasible, ideally the cluster members are contained on separate systems for optimum resiliency when an outage occurs. The members of the SSI cluster are managed together for HA reasons if an unplanned outage occurs. However, this management also helps planned guest mobility and centralized system administration. With SSI clustering, the members of the SSI cluster may be managed together.

Coupled with SSI, live guest relocation (LGR) facilitates the relocation of a Linux guest from one member of the SSI cluster to another. This relocation happens nearly instantaneously, without the Linux guest knowing about the relocation. Network processes and connections, disk operations, and user interactions on the Linux guest are unaware that the underlying infrastructure has moved to a different "physical" environment.

Figure 3-5 shows a simple representation of an SSI cluster that is composed of two members: lnx1host2 and lnx1host3. lnx1host2 hosts three Linux guests, and lnx1host3 hosts a single Linux guest.

Note: In this example, the SSI cluster has only two z/VM members, but there can be up to eight members on an SSI cluster in z/VM 7.3 and later releases, each of them with their guests.[a]

a. For more information, see https://www.vm.ibm.com/newfunction/index.html#ssi8.

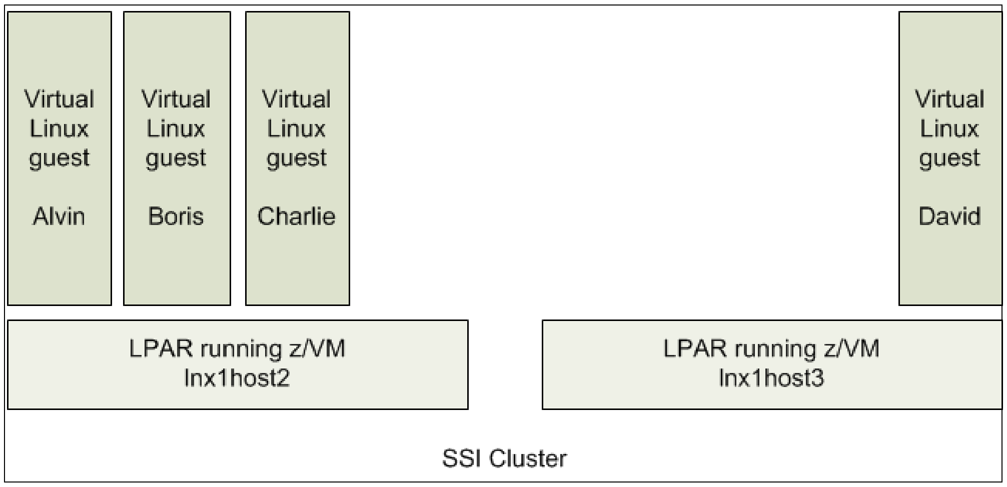

Virtual Linux guest Alvin
Virtual Linux guest Boris
Virtual Linux guest Charlie
Virtual Linux guest David
LPAR running z/VM lnx1host2
LPAR running z/VM lnx1host3
SSI Cluster

Figure 3-5 Simple representation of an SSI cluster before live guest relocation

The relocation of Linux guests from one SSI member to another makes it possible to perform maintenance on the individual SSI cluster members without disrupting the services running on the Linux guests. With all Linux guests relocated away from an SSI member, that SSI member can now be updated and reatarted with no impact to any running guests. When the maintenance on this SSI member is completed, Linux guests can be relocated back to their original host member. Perhaps all Linux guest systems can be relocated to this SSI member while similar maintenance is performed on other SSI members in the cluster.

An extra benefit of SSI and LGR is the ability to relocate workloads to accommodate a more balanced use of system resources. When an SSI cluster contains a configuration of multiple Linux guests overusing the network, a portion of the guests can be relocated to a different member of the SSI cluster where network utilization is lower.

Figure 3-6 shows that a Linux guest was relocated from lnx1host3 to lnx1host2 with no interruption in the services that are running from the Linux guest. Now that no guests are running on lnx1host3, the host can be restarted. After rebooting lnx1host3, Linux guests can be relocated back onto lnx1host3.

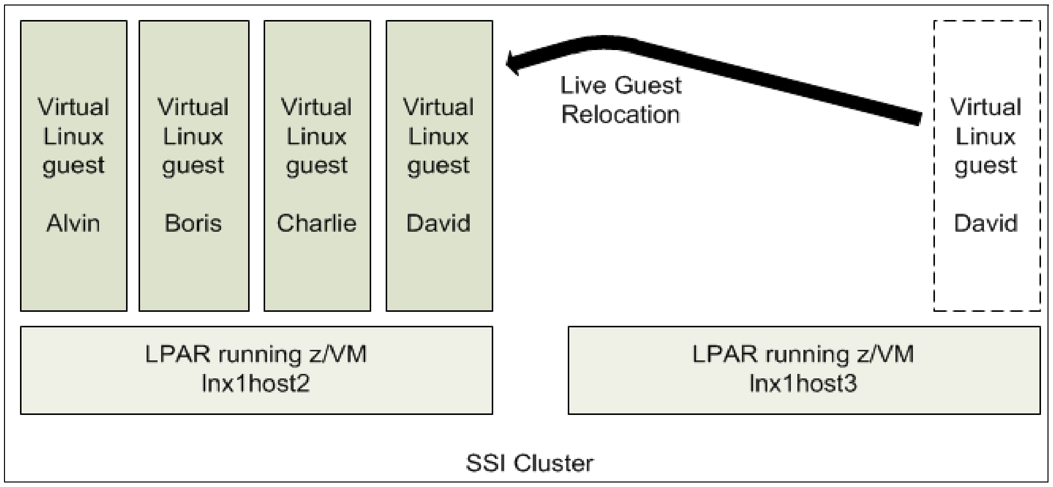

Virtual Linux guest Alvin
Virtual Linux guest Boris
Virtual Linux guest Charlie
Virtual Linux guest David
Live Guest Relocation
Virtual Linux guest David
LPAR running z/VM lnx1host2
LPAR running z/VM lnx1host3
SSI Cluster

Figure 3-6 A simple representation of live guest relocation of a Linux guest

This mechanism is a convenient way of relocating guests with LGR among the various SSI cluster members. Systems administrators need this flexibility to keep systems up to date while minimizing downtime. This mechanism also gives the administrators the ability to move workloads more freely within the infrastructure to make the best use of resources.

More to the point, knowing that z/VM, SSI, and LGR can be used in this way makes migrating workloads to LinuxONE all the more compelling.

For more information about SSI and LGR, see the following publications:

- *An Introduction to z/VM Single System Image (SSI) and Live Guest Relocation (LGR)*, SG24-8006
- *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, SG24-8147
- *Using z/VM v 6.2 Single System Image (SSI) and Live Guest Relocation (LGR)*, SG24-8039
- Part 5: "Single System Image Clusters Planning and Administration", in *z/VM Version 7 Release 3 CP Planning and Administration*, SC24-6271-73

3.6 KVM hypervisor components

KVM is one of many components that make up the entire virtualization solution. This section describes the major components that comprise the KVM hypervisor.

KVM relies on a basic infrastructure that is known as *virtio* to provide devices to a guest in a high-performance, low-processor requirement manner. These virtio interfaces connect different device types (disk, network, and so on) into a common platform used by both the KVM hypervisor and the guests that are connecting to it before being routed to the respective drivers that communicate with the actual device. For example, a disk is connected through virtio-blk by the guest before communicating with the Linux block driver on the host.

For more information about KVM, see *Virtualization Cookbook for IBM Z Volume 5: KVM*, SG24-8463.

3.6.1 Linux kernel and KVM module

The KVM module is a cross-platform virtualization technology that extends the Linux kernel into an enterprise-class hypervisor by using the virtualization capabilities of a real processor. In addition, several other modules and packages are included to provide the connectivity and resources necessary for a guest to be run under the host Linux system that runs KVM. This configuration includes, but is not limited to, the KVM and virtio modules that provide an interface to interact with hardware while separating access from individual VMs. Further, it provides all packages necessary to provide a useful environment for the system administrator, such as management tools.

The KVM module manages the sharing of all the virtual resources (processors, memory, and so on) onto the real resources, among the different VMs that are running in parallel.

3.6.2 QEMU

The QEMU application connects to the KVM interface of the host kernel and provides both emulation and virtualization of resources that are presented to a guest operating system that is running within the QEMU process.

The invocation of the QEMU process specifies all the resources that the guest has access to, such as processors, memory, disks, and network devices. An interface exists that allows certain resources to be added or removed from a running process; for example, to add a disk to that guest operating system.

3.6.3 The libvirt application programming interface

The libvirt application programming interface (API) is a collection of open source software that enables more human-friendly interfaces to the QEMU processes.

The API can be used to handle various tasks, such as starting or stopping a guest, adding or removing resources, or migrating a running guest to another system.

Libvirt includes a set of CLIs (the **`virsh`** command) for managing both the KVM itself and the VMs that run within the LinuxONE partition. Libvirt also provides the interface for resource management tools as described in 3.6.4, “Resource management” on page 52.

3.6.4 Resource management

Many tools can use the libvirt interfaces to manage the KVM and its VMs. This section describes a few common ones for LinuxONE, such as the IBM CIC and Cockpit, OpenStack, and the Virtual Machine Manager.

IBM CIC and Cockpit (with optional plug-in cockpit-machines) are tools for managing VMs by using a web browser. The IBM CIC is an OpenStack based tool and that can, from a single point, manage storage, network, and compute for VMs from several KVM hosts, even from different hypervisor hosts.[2]

OpenStack is a collection of services that manage storage, networking, and compute resources for cloud computing throughout the KVM lifecycle.

Virtual Machine Manager (**`virt-manager`**) is a graphical user interface for Linux that allows a user to manage multiple KVM hosts from a single location. It provides KVM lifecycle management, and some performance metrics of recent virtual resource usage of each VM.

3.7 z/VM hypervisor components

Two primary components of z/VM help PR/SM manage the virtualization environments:

- z/VM CP
- z/VM Conversational Monitor System (CMS)

These components are CLI operating environments that grant the system administrator control over the hypervisor. An extra component, the IBM CIC, enables a GUI that can be used for administration over the hypervisor and its guests. CIC can help minimize terminal and command interactions for daily guest management chores, but z/VM knowledge is still required from an effective infrastructure architecture management perspective, and for initial setup tasks.

This section describes and explains these components.

3.7.1 Control Program

The CP provides a guest (in this example, the Linux operating system, although z/VM virtualization supports other operating systems) with a complete VM environment with virtual resources that appear as real hardware resources.

Communication with the CP is through CP commands that are used by the z/VM administrator and Linux administrator to manage, query, and enable the definition of more resources.

When a Linux guest logs on to a z/VM session, it starts its own CP session. For production systems, this login is done automatically when the z/VM system is initially loaded or started. An entry exists in the z/VM directory for each VM that can be started.

[2] IBM CIC supports KVM, and z/VM virtualization. For more information, see 3.7.4, “IBM Cloud Infrastructure Center” on page 54.

Each entry contains detailed information about each user or guest and the virtual resources that are required by the guest operating system, and details about the relative priority of the VM. This information is used by CP to determine which VM is dispatched. Communication to the Linux system can be through the Linux VM console (which must be a 3270-type terminal emulator), or by using a Secure Shell (SSH) client terminal.

Note: If an administrator logs off the Linux VM console by using the conventional `LOGOFF CP` command, the guest machine ends the session at the z/VM host level, which causes the VM to power off and terminate all running work. The administrator must use the `DISCONNECT` command (not the `LOGOFF` command) to remove only the connection from the terminal to the console prompt and help ensure that the guest remains running in the background. This slight difference between `LOGOFF` and `DISCONNECT` can stop a running system.

3.7.2 Conversational Monitor System

The CMS is a single user operating system that runs only as a z/VM guest. CMS is used by the z/VM system administrator to manage the system components and to create and edit VM user profile entries in the z/VM environment. CMS provides an interactive environment for z/VM administration. Many instances of the CMS operating system support service machines, such as TCP/IP, print services, directory maintenance, accounting, and error recording.

For more information about z/VM, see *Introduction to the New Mainframe: z/VM Basics*, SG24-7316.

CP and CMS give the system administrator a more direct route to manipulate the available resources for the benefit of the Linux guest.

3.7.3 IBM Wave for IBM Z

Wave for IBM Z is an intuitive virtualization management software product that provides management, administration, provisioning, and enables automation of Linux virtual servers in a z/VM environment. Wave for IBM Z simplifies and accelerates the management and daily administration of highly virtualized z/VM and Linux server environments on IBM Z and IBM LinuxONE.

To reduce the complexity of z/VM management, Wave for IBM Z provides a solution to help system administrators in their daily tasks. Here is a list of features that can help with maintenance tasks:

- Display and manage virtual servers and resources, all from the convenience of a single graphical interface.
- Provision VMs, and install a guest operating system.
- Provision virtual resources, such as processors, memory, network, and storage.
- Capture and clone virtual servers across partitions.
- Create and configure virtual switches (VSWITCHes) and guest LANs.
- Relocate VMs to other partitions.
- Display, monitor, and manage z/VM hypervisor resources, such as paging.

3.7.4 IBM Cloud Infrastructure Center

IBM CIC is an advanced infrastructure management offering that provides on-premises cloud deployments of z/VM and Red Hat Linux KVM based Linux VMs (Red Hat, Ubuntu, or SUSE) on the LinuxONE platform and integration to higher-level cloud automation tools, such as IBM Cloud Automation Manager or VMware vRealize.

IBM CIC constitutes an alternative for former IBM Wave users for z/VM guest management that employed the tool to reduce the complexity of their daily system administration tasks. This tool covers those same needs, and provides more options.

IBM CIC includes the following features:

- Simplifies the VM lifecycle IaaS management tasks.
- Availability of a self-service portal for simple workload deployment.
- Provisions VMs, and installs guest operating systems.
- Enables a software-defined infrastructure.
- Integrates cloud management and Red Hat OpenShift by using OpenStack compatible APIs.
- Displays and manages virtual servers and resources from a single interface.
- Captures and clones virtual servers across partitions.
- Provisions VM virtual resources, network, storage, and compute, such as processors and memory.
- Creates and configures virtual switches (VSWITCHes) and guest LANs of various types.
- Relocates VMs to other partitions.
- Creates storage volumes, which include boot volumes, data volumes, and consistency groups backups.

For more information, see the following IBM CIC resources:

- IBM Cloud Infrastructure Center
- IBM Cloud Infrastructure Center documentation

3.8 Virtualized resources

A key feature of LinuxONE is how resource utilization is optimized and maximized.

In the current environment of distributed computing, the memory, CPU, or disk resources are underutilized most of the time the server is running. However, it is necessary to have the capacity available when the server reaches peak load. Consolidation of workloads enables resource sharing, which can be beneficial for non-constant workloads, and can be achieved with LinuxONE

With LinuxONE, a considerable amount of “overcommitment” is possible, such that memory, CPU, and I/O can adequately accommodate the workload when the workload needs it, and otherwise the resources can be used elsewhere without having to commit specific resources to any one workload.

Although resources can be rigidly committed to a specific workload, it is the flexibility of the virtual resources that is appealing. Overcommit is powerful for virtualized guests because not every guest needs all allocated resources concurrently. Having an idea of the consumption of resources of your partitions and guests is key to efficient virtualization and overcommitment. If the demand for resources is far over the physical limits, overcommitment is not efficient because resource management tasks become a large part of the total workload, and throughput is severely affected.

3.8.1 Virtualized CPU

LinuxONE is equipped with dozens of processor cores, which reflect directly on the performance of the Linux guests running in a partition. The number of virtual CPUs that are allocated to a single Linux guest should not exceed the number of logical CPUs that are available in a LinuxONE partition. For example, when the partition has four logical CPUs (two Integrated Facility for Linux (IFL) processors or two cores with SMT-2), do not allocate five virtual CPUs to a single Linux guest system. If a situation occurs where the Linux guest uses 100% of the CPUs, that usage might adversely affect the entire partition and all Linux guests that are running within it.

However, in a partition with four logical CPUs, you can assign three virtual CPUs to a LinuxA guest; that is, two virtual CPUs to a LinuxB guest and another two virtual CPUs to a LinuxC guest. All requests for CPU cycles are managed by the hypervisor. Generally, you can maintain a total of four or more active guests’ virtual CPUs to one logical CPU in a partition.

3.8.2 Virtualized memory

System memory is a resource that is shared across all LinuxONE guests. Each virtual guest is assigned a defined amount of virtual memory during logon.

The key to efficient memory management is to be aware of the total amount of virtual memory that is likely to be active at any time. Also, be aware of the amount of real memory that is allocated to the LinuxONE partition.

With both KVM and z/VM, you may overcommit memory, but keep the overcommitment ratio of the total amount of virtual memory likely to be active to the total amount of virtual memory to around 2:1. For test or development workloads, the ratio should be no more than 3:1.

A massive imbalance between total virtual memory and expected active virtual memory might lead to inefficient resource usage, which causes memory management workload and raises virtualization overhead.

The keys to determining the appropriate virtual memory size are to understand the working set for each VM, and to ensure that the Linux instances do not have any unneeded processes installed. Another recommendation is to use VDisks for swap, as described in “Swap device consideration” on page 58.

Memory management features

LinuxONE and its hypervisors include the following memory management features that you can use to reduce the amount of memory that is needed by virtual guests:

- Cooperative Memory Management (CMM)
- Cooperative Memory Management Assist (CMMA)
- Named Saved System (NSS)
- Discontiguous Saved Segment (DCSS)

As described next, these features are not available in a distributed x86 environment. Only LinuxONE can provide these versatile features, which dramatically reduce the amount of physical memory that is required to maintain a similar set of workloads.

CMM

CMM is used to reduce double paging that can happen between a Linux guest and z/VM. CMM requires the IBM Virtual Machine Resource Manager (VMRM) running on z/VM to collect performance data and notify the Linux guest about constraints when they occur. On Linux servers, the `cmm` kernel extension is required, and it is loaded with the **`modprobe`** command.

CMMA

CMMA enables a Linux guest to share the page status of all 4 KB pages of guest memory with the KVM or z/VM hypervisor. Linux does this sharing by marking the status of each page, which allows the hypervisor to preferentially steal unused and volatile pages and thus reduce paging.

NSS

NSS is a z/VM feature that allows virtual guests to share a read-only copy of a single operating system such as CMS or Linux. The benefit of this feature is that only one copy of the operating system is in storage accessible to all VMs. This feature decreases storage requirements and simplifies maintenance.

DCSS

DCSS is a z/VM feature that allows VMs to share reentrant code for applications, such as Oracle, which also reduces overall storage requirements. Figure 3-7 on page 57 shows how NSS and DCSS work. Linux guests use a single copy of the application in real memory. The NSS copy of Linux is also shared by all virtual guests.

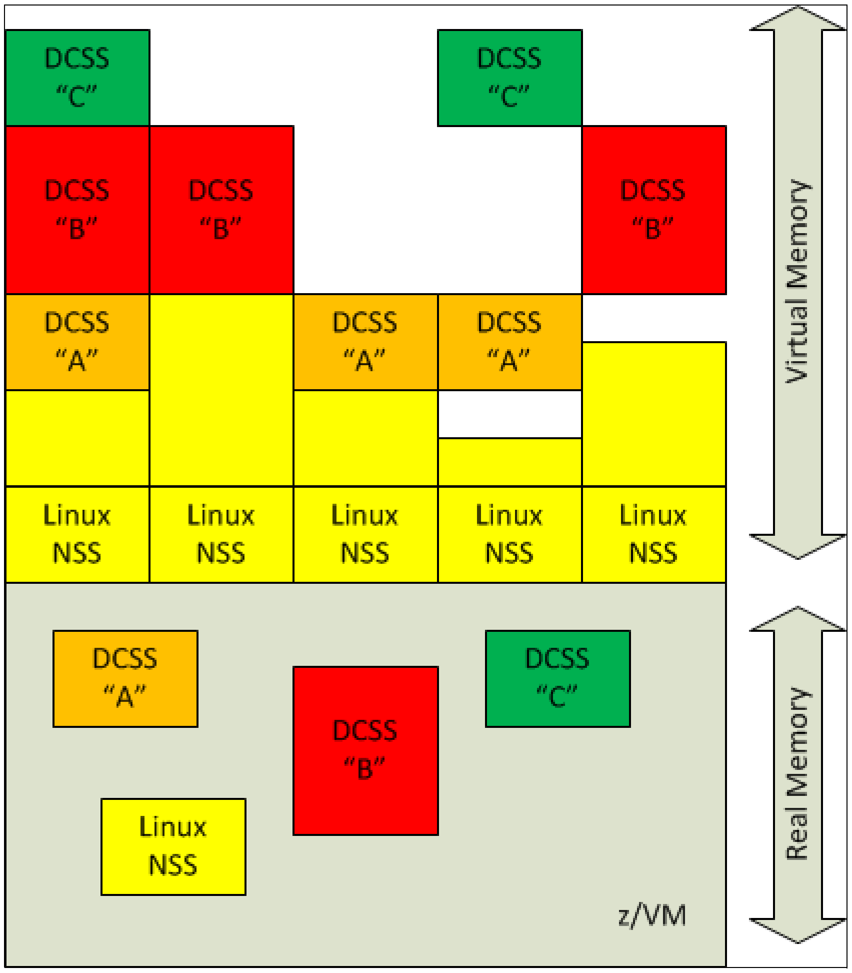


Figure 3-7 DCSS and NSS shared by multiple Linux guests on z/VM

For more information about setting up a DCSS and its use with the Execute-In-Place (XIP) file system, see *Using Discontiguous Shared Segments and XIP2 Filesystems With Oracle Database 10g on Linux for IBM System z*, SG24-7285.

Note: When defining memory requirements for virtual Linux guests, the Linux kernel uses all the extra available memory allocated to it as a file system cache. Although this feature is useful on a stand-alone system (where that memory would otherwise go unused), in a virtualized environment this causes the memory resource to be consumed in the partition. Therefore, it is important to assign only the memory needed for the running applications when they are at peak load.

Linux swap can be thought of as an overflow when an application cannot get enough memory resource. Thus, swapping activity indicates that the application must be analyzed to understand whether more memory is needed. Adding memory to a VM might not be the solution to the problem, but other memory swapping devices can be considered to manage guest memory usage more efficiently.

Swap device consideration

Understand that the concept of “swapping” is different today than when it was invented when large amounts of RAM were expensive. Modern operating system memory technology is more focused on paging than swapping.

Commit a specific amount of virtual memory to each Linux guest to accommodate no more than its intended workload, and fine-tune this amount of memory precisely so that Linux swapping does not normally occur. Generally, z/VM paging performs better than Linux swapping.

In the absence of the perfect memory configuration, and when workloads demand significant swapping, the ideal is to provide a VDisk device for this purpose. VDisks are virtual disks that are allocated in z/VM memory. They become a fast swap device for Linux. Swapping to a VDisk in memory is far more efficient than swapping to a regular disk, and it is less expensive, too, considering all factors. The Linux administrator must take care during the initial installation of the Linux guest to help ensure that the VDisk is formatted as a swap device. But more than that, the VDisk must also be formatted each time that the Linux guest starts.

For more information about optimizing memory on z/VM and Linux, see *Linux on IBM System z: Performance Measurement and Tuning*, SG24-6926.

3.8.3 Virtualized disk

Many x86 computing environments have disk storage that is maintained in storage area networks (SANs) and other similar, external storage arrays. LinuxONE is fully capable of using disk storage from a SAN or network-attached storage (NAS). Often, the system administrator chooses to maintain the data of a particular application on the storage array while migrating the application workload to LinuxONE. Whether maintaining the data on a SAN or migrating the data to the LinuxONE storage, the virtualized disk can be readily accessed by the workloads in the virtual environment. Often, leaving the data intact on the SAN eases and simplifies the migration effort.

With LinuxONE, SAN device support is expanded to include Small Computer System Interface (SCSI), connected to LinuxONE through Fibre Channel. In the x86 distributed world, the term Fibre Channel is often abbreviated as FC. To avoid confusion with FC in IBM terminology, referring to FICON channel devices, LinuxONE uses the phrase Fibre Channel Protocol (FCP) to refer to the connection to a SAN.

Typically, a SAN with Fibre Channel consists of independent and redundant fabrics, which provide connectivity between processor and peripheral devices. The Fibre Channel adapters each have their own unique worldwide name (WWN), which is put into zones within the fabric.

Modern Fibre Channel adapters can be virtualized by using N_Port ID Virtualization (NPIV). They provide several different virtual devices that all have their unique port name (WWPN) and can be put into separate zones, despite sharing physical resources on a server.

In theory, just one zone with all adapters and storage adapters would be sufficient. For actual production deployments, create a separate zone for each of the NPIV devices. The reason is that during logon and logoff of a single NPIV device, the whole zone is rediscovered. Although this process does not cause errors, it can cause short hangs, depending on the size of the zone. When a separate zone is created for each NPIV device, only the local zone is discovered, which has no effect on other zones.

Disk storage alone is not a virtual resource. The bits and stripes on the disk do not have the same characteristics for virtualization that memory does. Disk is a more permanent resource than memory. Nevertheless, allocating available disk space for a workload should be just as flexible as allocating virtual processing power or virtual memory. A competent hypervisor facilitates the management of disk storage.

KVM provides SCSI support by using the Linux `zfcp` and `scsi` drivers, and can pass a full SCSI Logical Unit Name (LUN) to a Linux guest through the virtio infrastructure. z/VM provides SCSI support by using its own drivers, and can pass either a full SCSI LUN, or a small piece of one as a minidisk, to a Linux guest. However, it is more common for z/VM to pass the FCP devices to a Linux guest, and allow the Linux to perform the SCSI configuration.

For a more detailed description of disk storage, see 5.2, “Storage analysis” on page 87.

3.8.4 Virtualized network

The physical network in LinuxONE consists of devices that are known as Open Systems Adapters (OSAs). An IBM LinuxONE 4 provides up to 48 OSA-Express7S adapters (up to 96 ports) for external network communications, handling up to 640 TCP/IP stacks simultaneously. The OSA supports copper and fiber Ethernet connections at speeds of up to 25 Gbps.

OSA devices can be virtualized through virtual switch devices to many Linux guests. It is available to KVM guests by using the Open vSwitch (OVS) interface, and to z/VM guests by using a VSWITCH controller. Each Linux guest connects by using a virtual device that is controlled by the qeth module to a virtual switch system in a LinuxONE partition. This network virtualization enables the creation of networks for guest groupings and services provisioning.

HiperSockets provide high-speed interconnectivity among guests that run on IBM LinuxONE, without any special physical device configuration or cabling. The guests communicate with one another internally by using the in-memory capabilities of the PR/SM hypervisor. However, HiperSockets are not intended to be used for sophisticated networking and should not be used for external traffic because their communication is internal only and does not reach the network layer.

HiperSockets are not based on any physical device like OSA Adapters. They use central processor computation capabilities for communication. This configuration might be important to consider if peak workloads are close to processor limits, since the use of HiperSockets may lead to a negative impact in performance.

OSA-Express and HiperSockets use the queued direct I/O (QDIO) mechanism to transfer data. This mechanism improves the response time by using system memory queues to manage the data queue and transfer between a hypervisor and the network device. Various examples are described in 5.1, “Network analysis” on page 76.

For more information about network in Linux, see *Advanced Networking Concepts Applied Using Linux on IBM System z*, SG24-7995.

Part 2

Migration

This part provides an overview of migrating from x86 to IBM LinuxONE, and helps you with migration planning.

The following chapters describe the key components of a migration analysis and provide an example of a hands-on migration. Planning checklists and worksheets are provided in this part to help you in your own planning and hands-on migration.

This part includes the following chapters:

4

Migration process

In the field of information technology, the term *migration* refers to the process of moving from one operating environment to another. Often, the move to a new platform involves various organizational and strategic changes.

This chapter provides you with information about the approaches that are involved in planning your migration and defines various types of stakeholders along with their roles and responsibilities. Not every organization uses the same titles for stakeholders, but the titles that you use should match the functions that are described in this book.

Also, this chapter describes the process for a migration project from identifying the stakeholders, assembling them, and identifying success criteria through to verifying both the migration itself and its success.

This chapter includes the following sections:

4.1 Stakeholder definitions

This section categorizes stakeholders as comparatively non-technical business stakeholders, or as more technically oriented information technology stakeholders. A stakeholder is anyone who is affected by the activities of the project. Conversely, it might also be stated that a stakeholder is anyone who affects the migration project. A stakeholder analysis is essential to facilitate the communication and cooperation between the project participants and to help ensure successful outcomes, whether the outcomes are individual milestones or the entire completed project. Help ensure that stakeholders are involved during the planning stages of the migration project, rather than just when they are needed to perform tasks for you in the execution stages of the migration project.

4.1.1 Business stakeholders

Business stakeholders are responsible for making the deciding about the business and provide direction for migration:

- Business owners or business managers

  These stakeholders lead business lines such as Chief Financial Officer (CFO), marketing, and sales. They are concerned with the business and financial resources that are used in the project. They often view information technology as a tool to accomplish business tasks efficiently and effectively. These stakeholders might have a staff member reporting on technical issues, including migration proposals that must be evaluated by the technology stakeholders. Conversely, proposals for migration might originate with the technology stakeholders, who must provide sufficient justification to the business owner. Migration justifications are discussed in Chapter 2, “Analyzing and understanding” on page 25.

  Large and complex consolidation projects require participation from several business owners and business lines. The business owners and IT management must be closely aligned and cooperate openly to achieve a successful migration.

- Business managers and supervisors

  These stakeholders are concerned with the workflow within their departments. They understand the importance of the application and how their employees use it. They select users who are the most qualified and motivated to participate in the migration project.

- Quality Auditors

  Large and complex consolidation projects require participation from quality auditors to create the Quality Indicators (QIs) and help ensure that the QI is achieved post-migration project.

- Users

  These stakeholders are the end customers. They use the application or consume the services that are provided by the application, and perform testing to help ensure that the application is working at least at the same level after the successful implementation of the migrated system. In a migration without enhancements, users should not see any changes. Availability and response times must meet the service level objectives agreed to by management and communicated to the users. Their perspective and input to the conversion project is valuable. Their satisfaction must be a criteria for the success of the migration project.

4.1.2 Operational stakeholders

Operational stakeholders are different from business stakeholders in that they are the people who are responsible for implementing the systems and changes:

- Chief Information Officer (CIO)

  The highest level of IT management is usually the CIO. In some companies, the highest level of IT management is a director or a manager. This stakeholder's role is to provide vision and leadership for information technology initiatives. The main concerns are to support business operations and services and to improve cost effectiveness, improve service quality, and develop new business process services. These stakeholders should clearly understand the benefits and risks of the migration project.

- Project manager

  This stakeholder is responsible for creating and managing the plans, interdependencies, schedule, budget, and required personnel for the migration effort.

  Other responsibilities include defining and obtaining agreement on the approach. The project manager tracks and reports to all key stakeholders on progress against plans, escalating any issues or risks where appropriate.

- IT managers and supervisors

  Some stakeholders are managers or supervisors of system administrators and other infrastructure personnel. Managers at this level have various types of influence on the migration project. Some projects are originated and championed by these stakeholders. They usually have a high level of technical competence and understanding of the technologies that are used in the migration project. These stakeholders should be intimately aware of the staffing and training considerations of the migration project. They should work closely with their staff to assess current skills and map out a training plan to acquire the required hardware and software-related skills.

- Infrastructure administrator, hardware administrator

  Infrastructure administrators are responsible for defining and configuring hardware and the virtualization layer in IBM LinuxONE. IBM LinuxONE offers several virtualization technologies: PR/SM, z/VM, and Kernel-based Virtual Machine (KVM).

  Unique to IBM LinuxONE, PR/SM is a Type-1 hypervisor that runs directly on bare metal, so that you can create multiple logical partitions (LPARs) on the same physical server. These stakeholders can easily define and configure hardware definitions by using the IBM Dynamic Partition Manager (DPM) tool to configure and define LPARs and their resources like processor, memory, storage, and network. In addition, the various cloud management capabilities provided by both z/VM and KVM are OpenStack-enabled, which makes the infrastructure administrators' duties more efficient.

- Linux, UNIX, and Windows administrators

  Linux administrators might help install Linux, or take over administration tasks after the Linux guest is installed. These stakeholders work closely with the system programmers when major configuration changes or additions are made (such as increasing the memory, disk space, or CPU). All other Linux administration duties are the same as on other platforms, such as Linux on x86.

  Various other Windows and UNIX administrators might be involved in the migration project. This involvement is partially dependent on where the source system is hosted (that is, the platform where the source application is). The administrator of the source system is heavily involved because that is the application that is being migrated.

Other services, such as DNS, mail servers, and security, run on UNIX or MS Windows servers. These and other services will usually be required by the application that is being migrated. The administrators of these services are required to make adjustments for the migrated application.

- Network engineers

  These stakeholders design, install, and maintain data communication equipment, such as routers, switches, local area networks (LANs), wide area networks (WANs), and other network appliances. They monitor the network for performance and errors. During migration, network engineers help to design the new network and deploy any changes to the existing network.

  For more information about IBM LinuxONE networking, see 5.1, “Network analysis” on page 76. The network concepts and tools outside of IBM LinuxONE is the same for these stakeholders.

- Database administrators (DBAs)

  The tasks that are performed by these stakeholders can be separated into two or more different but related job functions such as database analyst, DBA, and system administrator. The DBAs are responsible for installing and maintaining the database management system (DBMS) code base. They design and implement the corporate databases, help ensure the data integrity, and good database performance. They work closely with the application development group to help ensure that the application is running efficiently.

- Application architects and developers

  Applications that are developed in-house require porting and testing on the target Linux system. The effort that is involved can vary greatly, depending on what language the application is written in and how hardware-dependent the code is. Open source and commercial tools are available to help with tasks such as assessing the portability of your applications. IBM Consulting™, as part of its migration services offerings, uses tools developed in cooperation with IBM Research® to help with code assessment and conversion. The application architect and developers are the stakeholders who are responsible for this porting effort. See 5.3, “Application analysis” on page 97 for more information about the issues that need to be considered.

- Operators

  The operators monitor the application, the operating system, and the physical environment by checking the monitor consoles, logs, and alerts. They raise problem tickets, notify support teams, and escalate issues to management. New tools and procedures that result from the migration project are required for them.

- Service Desk staff

  These stakeholders are on the front line of support to the customer. They are usually the first ones to get a call when there is a real or perceived problem with the application. They need to be the first staff trained on the new environment, and should be heavily involved in the migration testing so they can provide meaningful support after the migration.

- Users

  Perhaps the most important stakeholders that are involved in a migration are those who will use the application every day. They need to be involved from the beginning because the success of the project depends in large measure on how simple the system is for them to use. Ideally, it should have the same interface to which they are accustomed. However, often a migration is often an opportunity for firms to improve the application, which often results in additional functions and procedures that they need to learn.

Note: Users are identified both as business stakeholders and as operational stakeholders.

- Vendors

  The third-party vendors have many resources that you can use, and they are often ready to help if you make your needs known. They can respond quickly and are often the most cost-effective source of information and solutions.

  For independent software vendor (ISV) applications that you are targeting for migration, you need to determine whether the vendors provide compatible versions that support the distribution of Linux that you plan to use. Many ISV applications have other third-party dependencies. Vendors should be able to help you map out all ISV dependencies, including middleware. Most leading middleware products are available on IBM LinuxONE, and there are often open source alternatives.

- Contractors

  Specialists can be called on to assist with transient needs. They can provide skills that your staff does not yet have, or skills that will not be needed after the migration project is completed. Contractors can be used to enhance the skills of your staff as they perform tasks on the migration project. Make sure that skills transfer takes place for persistent, recurring tasks.

4.1.3 Security stakeholders

The functional area of security has become more visible and critical as company assets become more exposed to the internet and available on mobile and wireless devices. The security stakeholders include *security administrators*.

The security administrators are the team responsible for data protection, including the authentication and authorization of users who access company applications. The target application must adhere to existent security policies or demonstrate heightened security methods and standards. For more details about IBM LinuxONE Security, see 5.6, “Security analysis” on page 116.

4.2 Identifying the stakeholders

The first phase of the migration involves identifying the stakeholders, as defined in 4.1, “Stakeholder definitions” on page 64. In turn, the stakeholders identify the business and operational requirements that affect the migration process. All stakeholders within the company must be consulted to help ensure that their requirements are factored into the migration planning.

Identify the following stakeholders, as defined in 4.1, “Stakeholder definitions” on page 64:

- Business stakeholders define the business and success criteria.
- Operational stakeholders provide information about the application requirements, database requirements, available network bandwidth, CPU load, and allowable downtime.
- Security and compliance teams define compliance requirements for the entire migration effort.

4.3 Assembling the stakeholders

Holding a meeting of stakeholders (or representatives of larger groups of stakeholders) is a useful way to set expectations and to address other planning considerations. Such a meeting helps to uncover whether extra administrator, manager, or user skill enhancements are needed. The participants are also the people to whom status and milestone results are reported. Some of these people might have never met, and a cohesive, efficient, and successful project requires personal relationships.

To make sure that all interests are considered, request a meeting of the key people who requested the migration and who are affected by it. Subsets of stakeholders with related tasks and responsibilities should also meet to enhance communications and encourage teamwork.

4.3.1 Communicating the change

Stakeholder meetings can be an efficient way to open communication channels. Effective communication plans help to “flatten out” the negative aspects of the acceptance curve.

A communications plan, which is coupled with proper training on the new system, should minimize the number of users who reject or oppose the project. It encourages users to start out with acceptance instead of dissatisfaction as the initial response, and lead to a quick transition into exploration and productive use.

These issues are even more important regarding the IT support team. A strategic decision to switch an operating system or platform can inadvertently create an impression of disapproval of the work that the team has done so far. This perception might cause staff to think that their current skills are being devalued.

Be able to articulate the objectives for your IBM LinuxONE migration and relate them to your key business drivers. Whether you are trying to gain efficiencies by reducing costs, increasing your flexibility, improving your ability to support and roll out new application workloads, or some other key business drivers, be sure to set up objectives that line up with these goals. Even the smallest of migrations should be able to do this process, and it will help guide your planning.

Defining metrics (increased performance, more uptime, open standards, enterprise qualities) early in the project helps the team stay focused and reduces opposition. Be sure that you have a means of tracking the metrics. Getting stakeholder agreement on your metrics early in the project helps ensure the support of everyone from executives to users.

Often, the migration to IBM LinuxONE is accompanied by other objectives. For example, some customers upgrade their database at the same time to get the latest features and performance enhancements, and to obtain support that works well with the latest distributions of Linux. As with any project, the scope must be defined to prevent project overrun. However, it is also important that you have a means to manage additions to the plan as business needs dictate.

Because cost is often a key motivator for migrating to IBM LinuxONE, give careful consideration to identifying where cost reduction is targeted. Identify metrics for defining return on investment (ROI) before beginning migration activities, and identify metrics for other success criteria.

4.4 Migration methodology

After the business value and need for migrating to IBM LinuxONE has been accepted by the stakeholders, it is time for the actual migration planning.

In a typical migration scenario, an entire environment must be identified, rationalized, and tested for compatibility with the new host operating environment. Figure 4-1 illustrates an approach to planning.

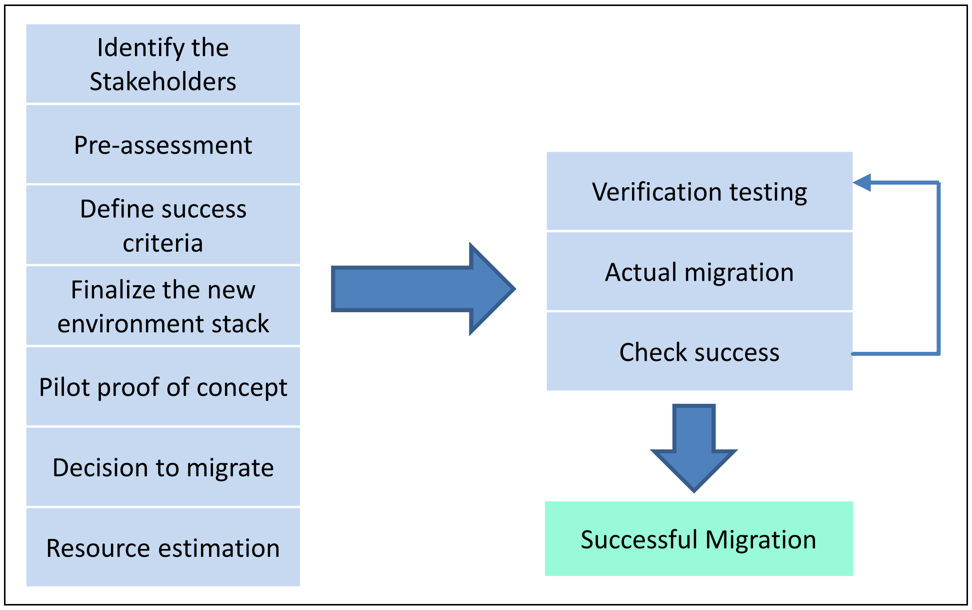


Figure 4-1 Typical migration approach

Identifying the stakeholders is described in 4.2, "Identifying the stakeholders" on page 67 and 4.3, "Assembling the stakeholders" on page 68. This section describes each of the remaining elements in this approach.

4.4.1 Pre-assessment

During the pre-assessment phase, a high-level analysis and initial feasibility study of the application architecture, source code dependencies, database compatibility, and build environment is performed. This task defines an overall scope for the migration to the target operating system. The applications that run on the current servers are assessed to determine whether they are available and certified to run on IBM LinuxONE. In addition, an evaluation of the risks that are related to migration is performed. This process helps to identify major risk areas at the earliest stage.

Also, perform a careful analysis of present and anticipated business needs and weigh the results against the pros and cons inherent in each option of migration. The outcome of this phase is a best practicer migration approach, and a high-level risk assessment and analysis report that identifies potential issues that can occur during the migration.

4.4.2 Defining the success criteria

In this phase, a consensus must be reached by all stakeholders regarding the “success criteria” of the porting project. Migration success might mean, for example, passing a percentage of system tests on the IBM LinuxONE platform or passing a level of performance criteria set out by the quality auditor in agreement with the other stakeholders.

Regardless of how the project success is defined, all stakeholders must understand and agree on the criteria before the porting effort starts. Any changes to the criteria during the porting cycle must be communicated to all stakeholders and approved before they replace the existing criteria.

4.4.3 Finalizing the new environment

Usually a migration involves moving custom-built or third-party applications to another operating environment. This task involves careful analysis of different tiers of the hierarchy based on a best fit between the database, the application requirements, and other environmental attributes.

Generally, perform a one-to-one mapping of the various middleware, compilers, third-party tools, and their respective build parameters. If any of the one-to-one mappings for any parameters are missing, you must list other parameters that are available in the tool that provide the same functions or features. For examples of forms that can be used to help document your software and hardware requirements, see 2.6, “Planning checklists” on page 32.

During this phase, most of the technical incompatibilities and differences in the environmental options are identified, and are fixed.

Custom-built applications

If custom-built applications are written in one or more programming languages, several tools might need to be validated on the target environment. These tools can include compilers, the source code management system, the build environment, and third-party add-on tools.

Also, an in-depth analysis should be carried out on the various build options that are specified to help ensure that the tools on the IBM LinuxONE platform provide the expected functions after the migration (for example, static linking, library compatibilities, and other techniques). The effort that is involved can vary greatly depending on how portable the application code is.

ISV applications

If you are running ISV applications on x86 that you are targeting for migration, you need to determine whether the vendor provides compatible versions that support the distribution and version of the target IBM LinuxONE. Many ISV applications have other third-party dependencies. Be sure to map out all ISV dependencies, including middleware. Most leading middleware and open source products are available on IBM LinuxONE.

Note: Many open source alternatives are available for many applications and services for IBM LinuxONE.

4.4.4 Pilot proof of concept

After you have a clear understanding of the target environment and the areas with possible issues and risks, you can proceed to a pilot proof of concept (POC). This phase is a subset of the actual migration, but with a reduced scope and duration. In this phase, you implement a small module or stand-alone code snippet from the application onto the target environment.

The POC phase should involve all the same tasks and activities of the full migration. The main objectives of the POC are to focus on the identified areas of risk, empirically test the best practices approaches, and prove that the full migration can be completed successfully.

In this way, the major potential migration risks that are identified during the pre-assessment can be addressed in a controlled environment, and the optimum solution can be selected. This service targets the areas of issue and risk, proves that the optimal resolution methods have been selected, and provides a minor scope of the whole migration.

Note: POC projects might require extra funding and can lengthen the project schedule, but will likely contribute to the project's success.

4.4.5 Decision to migrate

After the pilot is complete, you should have a complete analysis of the target operating system environment and a plan that details the resources, time, and costs that are required to migrate to IBM LinuxONE.

During this phase, analyze and discuss all key requirements with the stakeholders including timing, resource needs, and business commitments such as service-level agreements (SLAs). Also, discuss any related aspects of the migration, such as new workloads, infrastructure, and consolidation. The decision to implement the migration must be acceptable to all stakeholders involved in such activity, especially the business owner.

4.4.6 Resource estimation

Understanding the migration objectives and developing metrics with stakeholder involvement and agreement helps to provide a useful base from which to build a plan. Be sure to include all key requirements (such as resource needs) and business commitments (such as SLAs) for each stakeholder in the plan.

Migration activities rely heavily on having ready access to the personnel responsible for the development, deployment, and production support of the applications and infrastructure in question. Anticipating change and helping ensure the early involvement of affected teams are efficient ways to handle change issues. For example, support staff for hardware might be comfortable with x86 related hardware support and know where to go for help. However, practitioners who are expert in the previous environment might be less open to change if they feel threatened by new ways of doing things where they do not have expertise.

Consider the following areas when performing your resource estimation:

- Resources

  Determine what hardware and software are required. Identify the housing aspects required (for example, whether the electrical and cooling inputs are equal). Identify skill requirements. Decide what staff are needed to help with the crossover.

- Education

  Identify skills-related requirements and determine whether the staff has adequate Linux education. Decide whether there are special skills that are needed for the areas specific to hardware or Linux and hardware combination.

- SLAs

  While installing, configuring, and testing the change is occurring, determine what the support mechanisms are for both you and any vendors. Determine what your commitments are to current stakeholders while you are performing the migration.

- Related project aspects

  Determine what other projects are occurring in addition to the basic system changeover.

4.4.7 Actual migration

The scope of this phase is performing the actual migration of the applications and the infrastructure to the IBM LinuxONE environment, thus producing an environment that is ready for handover to the testing phase.

The team follows the planned approach and methodology during their migration activities. If needed, modifications are made to the application source code and build environment. The new application binary files are generated and checked for compliance with the target version of the operating system.

4.4.8 Verification testing

The purpose of performing a formal test is to provide objective evidence that the predefined set of test objectives is verified and the customer test requirements are validated on the target operational environment. This step is important before verification of a successful migration. The goal is to validate the post-migration environment and confirm that all expectations have been met before committing or moving to production.

Keep the following questions in mind for validation:

- Does it interoperate correctly?
- Can it handle the expected load?
- Does it have the expected performance?

If any performance issues are encountered during this stage, the target environment can be tuned for maximum performance.

4.4.9 Checking against the success criteria

After you successfully migrate the environment, reassess the original acceptance criteria with all stakeholders. If the criteria are achieved, move the environment to production and obtain a sign-off for the migration activities. Figure 4-2 shows three important criteria of success from a user perspective.

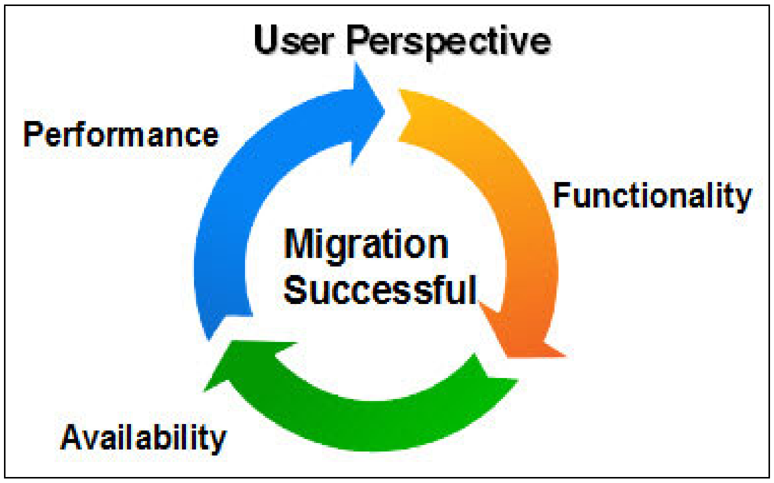


Figure 4-2 Criteria of success from the user perspective

If the success criteria are not achieved, the migration implementation must be reviewed. After the review is complete, the testing phase must be redone to help ensure that the application being migrated meets the acceptance criteria and is ready to go into production.

5

Migration analysis

This chapter helps you to understand new features that are found on Linux on IBM LinuxONE and provides a technical direction for your migration. Each section addresses a different part of your infrastructure, and uses scenarios to show how the migration affects the environment.

This chapter includes the following sections:

5.1 Network analysis

This section provides information about network migration configuration issues, explains how the virtual network can be configured, and the facilities that are available on LinuxONE and its hypervisors. The following terms and components are used throughout LinuxONE:

- Open Systems Adapter

  The Open Systems Adapter (OSA) serves the same function as an Ethernet card on x86, by acting as the hardware network controller and providing connectivity to clients on local area networks (LANs) or wide area networks (WANs). It can be directly attached on Linux, but typically is attached to virtual switches. For more technical information about OSA cards, see *IBM z16 Technical Introduction*, SG24-8950.

- OSA with Link Aggregation

  You can aggregate multiple physical OSA cards into a single logical link, which is called a link aggregation group (LAG). This configuration increases the bandwidth and provides a non-disruptive failover. For more information about how to configure it, see *Advanced Networking Concepts Applied Using Linux on IBM System z*, SG24-7995.

- HiperSockets

  HiperSockets is a LinuxONE microcode emulation of a Logical Link Control Layer of an OSA interface, and provides near zero latency at memory speed communications between servers running in different LPARs, but not external connections. If an external connection is required, a HiperSockets bridge must be implemented by using a virtual switch, or a Linux guest must be set up as a router.

  HiperSockets provide a fast connection between LPARs. This direct connection without involving real hardware is an important factor to simplify setups with many Linux systems. Some benefits are explained in *Set up Linux on IBM System z for Production*, SG24-8137.

- Virtual switch

  A virtual switch (VSWITCH) is a software program that enables one virtual host to communicate with another virtual host within a computer system. Virtual switches typically emulate the functions of a physical Ethernet switch. In LinuxONE, a VSWITCH provides direct attachment of Linux guests to the local physical network segment. The VSWITCH allows IP network architects and network administrators to treat Linux guests as a regular server in the network.

  The actual speed of a connection with a VSWITCH depends on several different variables. The type of traffic is as important as the real underlying hardware and the maximum transmission unit (MTU). MTU is the maximum size (in bytes) of one packet of data that can be transferred in a network. Common to all of those solutions is that the VSWITCH is faster than a real switch that is connected to the server would be.

  Implementing virtual local area networks (VLANs) also helps if different guests run in different security zones of a network. It is simple to configure network interfaces to Linux guests that provide only selected VLANs to the guest. These interfaces can be configured either as tagged VLANs or as single untagged VLAN on an interface.

5.1.1 Network facilities on LinuxONE and Kernel-based Virtual Machine

The switched network inside a Kernel-based Virtual Machine (KVM) hypervisor is commonly implemented with Open vSwitch (OVS) devices, which provide a robust, multilayer, open source virtual switch. OVS supports standard management interfaces and protocols, and can bond multiple OSA devices together for redundancy. OVS devices can communicate between virtual machines (VMs), or between a VM and an external network hosted by KVM.

Another networking device, MacVTap, virtualizes bridge networking and is supported on KVM. However, OVS bridging is preferred due to its richer features and a more granular control over devices.

VLAN and VLAN tagging are supported by both OVS and MacVTap devices.

5.1.2 Network facilities on LinuxONE and z/VM

The switched network inside a z/VM hypervisor is implemented in Control Program (CP) as a VSWITCH, managed by VSWITCH controller VMs. When running the VSWITCH as Layer 2, it behaves similar to a real switch just between VMs.

VSWITCHes do not need a connection to an OSA card to operate. They can also provide purely virtual networks. This feature also simplifies the setup of private interconnects between guest systems. When creating private interconnects in a Single System Image (SSI) with live guest relocation (LGR) enabled, use dedicated VLANs with external interfaces. This configuration is necessary to accomplish the private connection between guests that run on different nodes in the SSI.

The VSWITCH infrastructure provides two basic configuration options. One configures user-based access, and the other configures port-based access. From the possibilities, both are equivalent. Just the configurations differ.

You can read more about VSWITCH benefits on *Set up Linux on IBM System z for Production*, SG24-8137, and technical information about *Advanced Networking Concepts Applied Using Linux on IBM System z*, SG24-7995.

RoCE Express

The 25 GbE and 10 GbE RoCE Express2.1 features use Remote Direct Memory Access (RDMA) over Converged Ethernet (RoCE) to provide fast memory-to-memory communications between two LinuxONE servers.

These features are designed to help reduce consumption of CPU resources for applications that use the TCP/IP stack (such as IBM WebSphere that accesses an IBM Db2 database). They can also help reduce network latency with memory-to-memory transfers by using Shared Memory Communications over RDMA (SMC-R).

With SMC-R, you can transfer huge amounts of data quickly and at low latency. SMC-R is transparent to the application and requires no code changes, which enables rapid time to value.

Internal Shared Memory

Internal Shared Memory (ISM) is a virtual PCI network adapter that enables direct access to shared virtual memory, and provides a highly optimized network interconnect for LinuxONE platform intra-communications. Shared Memory Communications-Direct Memory Access (SMC-D) uses ISM. SMC-D optimizes operating systems communications in a way that is transparent to socket applications. It also reduces the CPU cost of TCP/IP processing in the data path, which enables highly efficient and application-transparent communications.

SMC-D requires no extra physical resources (such as RoCE Express features, PCIe bandwidth, ports, input/output (I/O) slots, network resources, or Ethernet switches). Instead, SMC-D uses System-to-System communication through HiperSockets or an OSA-Express feature for establishing the initial connection.

For more information about RoCE Express and ISM, see *IBM z16 Technical Introduction*, SG24-8950.

5.1.3 Network migration overview

Consider several different levels of network migration because LinuxONE allows for complete virtual network systems. Among other features, you can create multiple virtual switches in the same partition, and the created VLANs.

One VSWITCH instance operates at Layer 2 or Layer 3 of the OSI Reference Mode. It is virtually attached to the same network segment where the OSA card is physically connected.

This section covers some common scenarios and how they look on LinuxONE.

Single network scenario

One of the most common scenarios is the migration of several distributed machines from the same physical subnet to a single LinuxONE partition attached to the same network segment. Figure 5-1shows an example that involves a single distributed network.

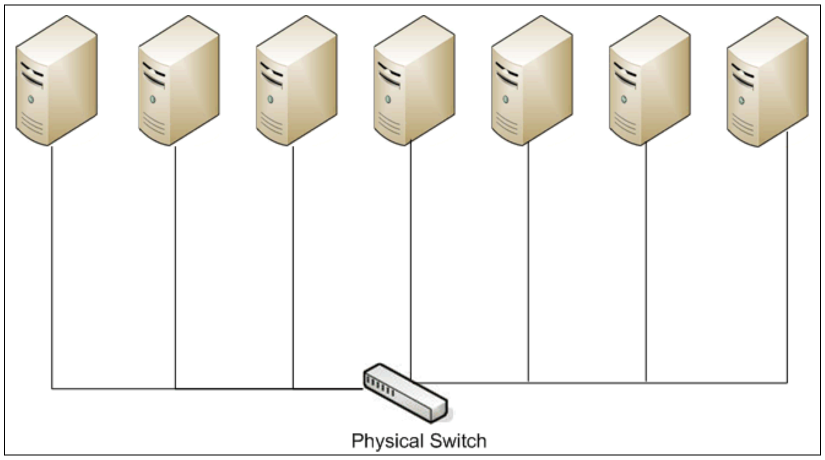


Figure 5-1 Single distributed network

Within this scenario, all physical machines can be migrated to a single LinuxONE machine running Linux and sharing a virtual switch that is attached to an OSA card. The OSA card is then connected to the physical network. Figure 5-2 shows this type of configuration.

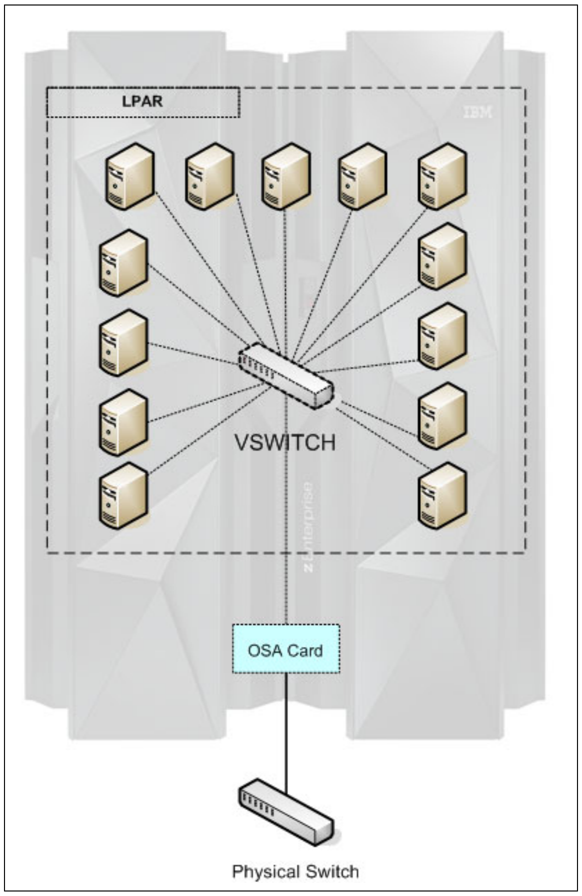


Figure 5-2 Single virtualized network

To increase the availability of each Linux guest, the preferred solution is to configure two or three OSA cards that are attached to different physical switches in the network. This configuration provides a network failover capability, as shown in Figure 5-3.

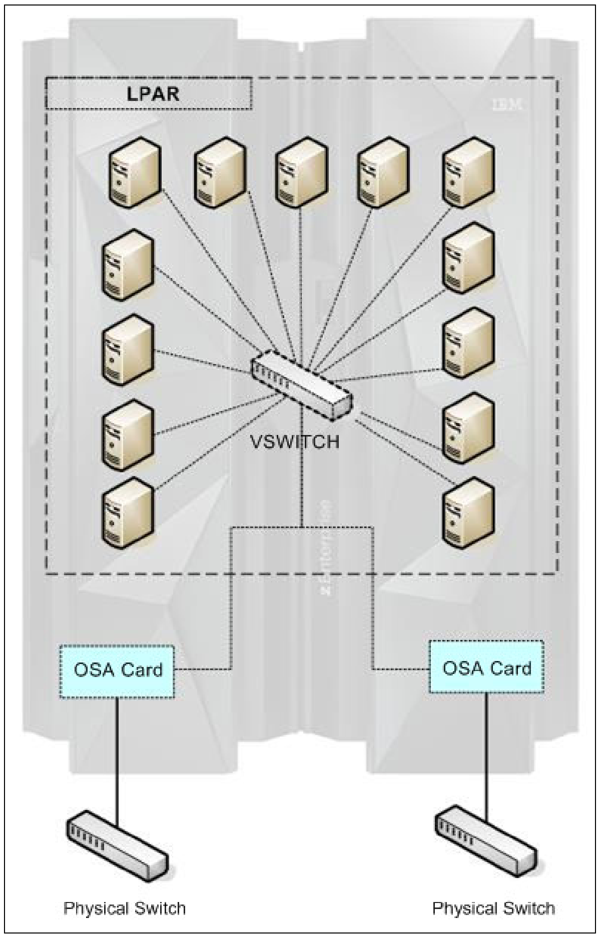


Figure 5-3 Single virtualized network with failover solution

In a Layer 2 VSWITCH configuration, all Linux guests have their own Media Access Control (MAC) address. In a Layer 3 VSWITCH configuration, the Linux guests respond with the OSA card's MAC address to requests from outside the LinuxONE LAN segment.

In a multiple partitions scenario where a single network segment is used, the preferred solution is to share the OSA card between partitions. Each partition's VSWITCH is connected to the OSA card, and the OSA card is directly connected to the physical network segment. This scenario is a common one where the development and production servers are in separate partitions. This configuration is shown in Figure 5-4.

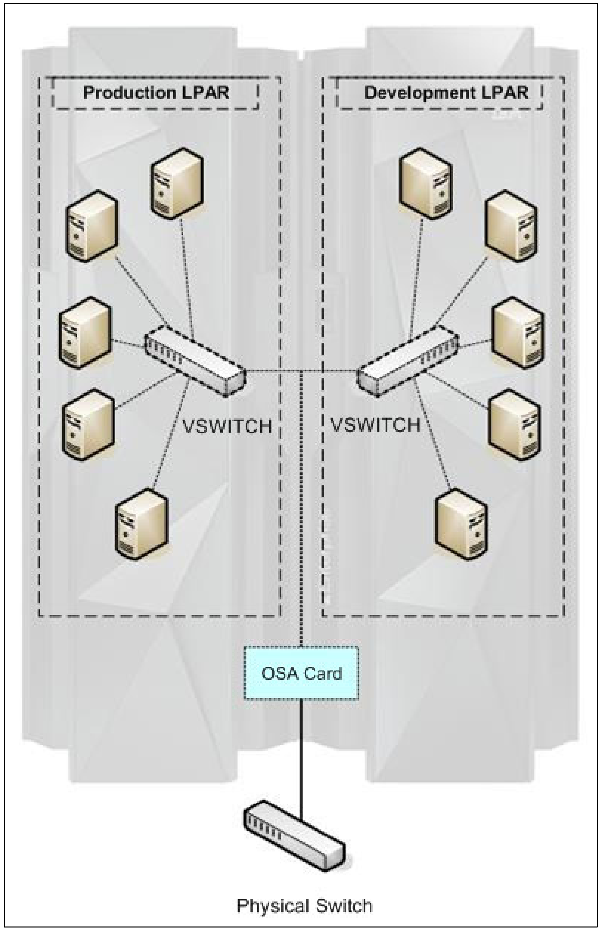


Figure 5-4 Single virtualized network with multiple logical partitions

Similarly, the failover solution that was described previously can also be applied in this case. Sharing the two OSA cards between partitions is shown in Figure 5-5.

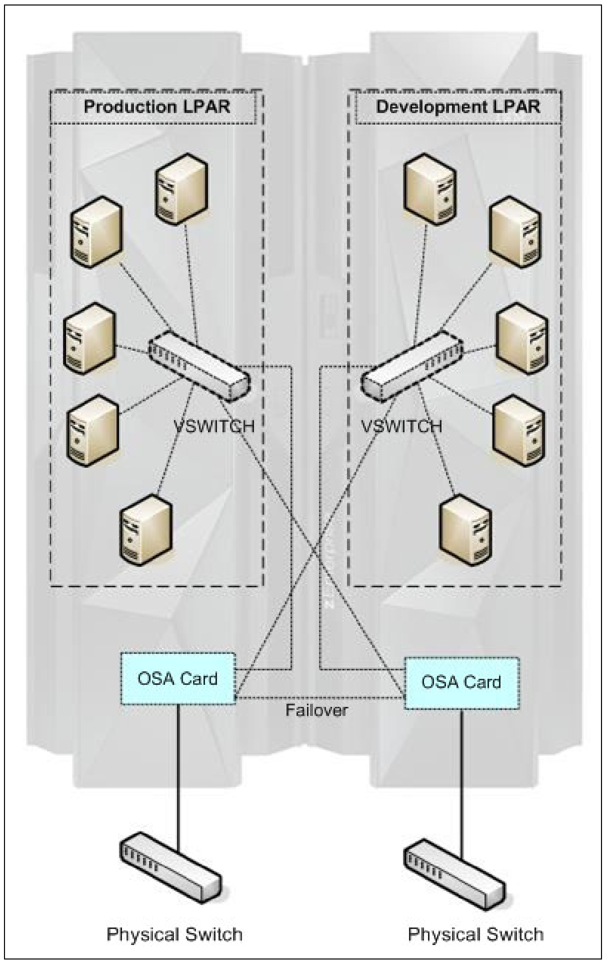


Figure 5-5 Single virtualized network with multiple logical partitions and failover

Multiple networks scenario

Several types of network solutions require multiple network segments. Some of these solutions demand package routing or the use of multiple logical partitions (LPARs). This section provides suggestions for each type of network design.

DMZ and secure network

In some scenarios, different network segments are migrated to LinuxONE and share a physical LinuxONE server. Analyze the DMZ and a secure network scenario. Figure 5-6 shows a DMZ network where the Web Application Server is placed, and a secure network where the database server is located.

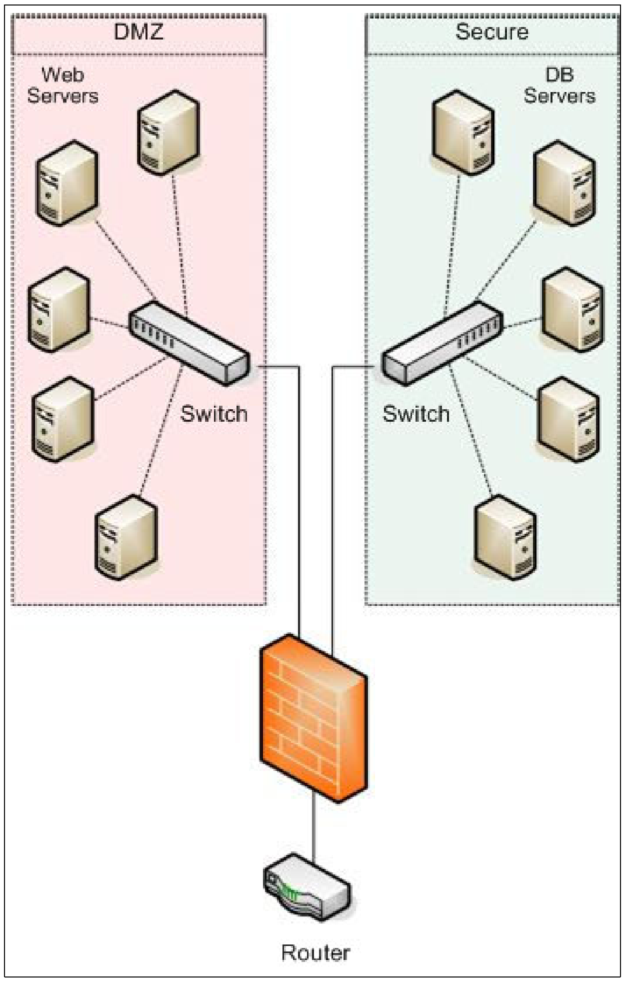


Figure 5-6 Multiple distributed network scenario: DMZ segmented network

You can set up the same scenario on LinuxONE. If you have in place a physical switch, a third-party firewall solution, and a router in your environment, you can reuse them as part of your network planning on LinuxONE. Otherwise, you can use some network facilities available on LinuxONE and its hypervisors.

The OSA card is connected to one physical switch (or two OSA cards, when the failover solution is configured). The physical firewall can be replaced by a Linux guest that can act as a router and firewall if you do not have an appliance firewall solution. All virtual Linux guests are connected to two VSWITCHs in two different network segments. Figure 5-7 shows a network that uses a Linux guest as a firewall.

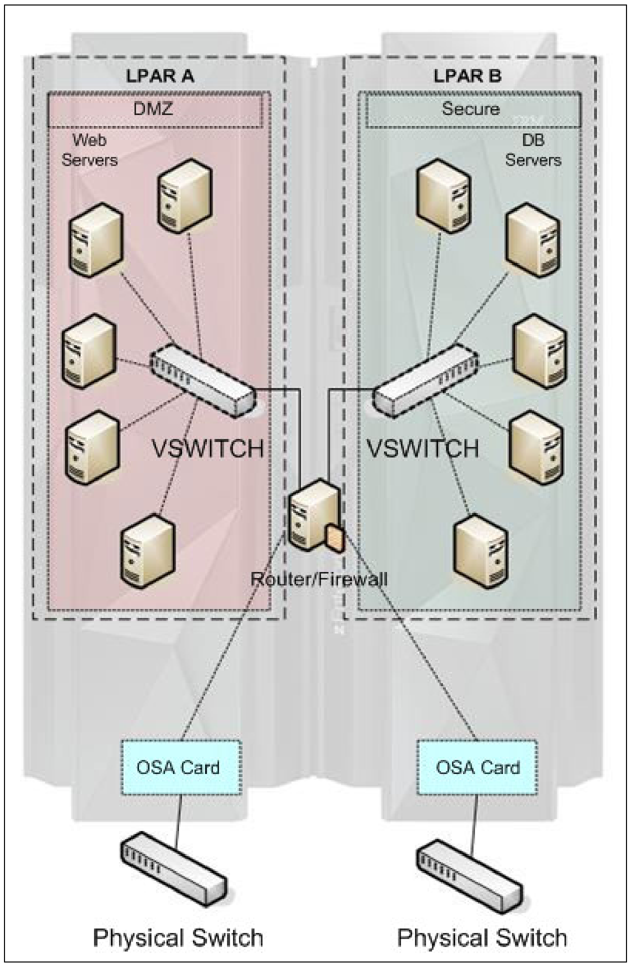


Figure 5-7 Multiple virtualized network scenario: DMZ and secure network

You might notice in Figure 5-7 on page 84 that the configuration does not share the OSA cards. It is possible to have the OSA card shared between multiple partitions on the same physical LinuxONE server. To create this solution, generally use an external firewall to manage the network filters. Figure 5-8 shows the solution that is described as a network-segmented partition.

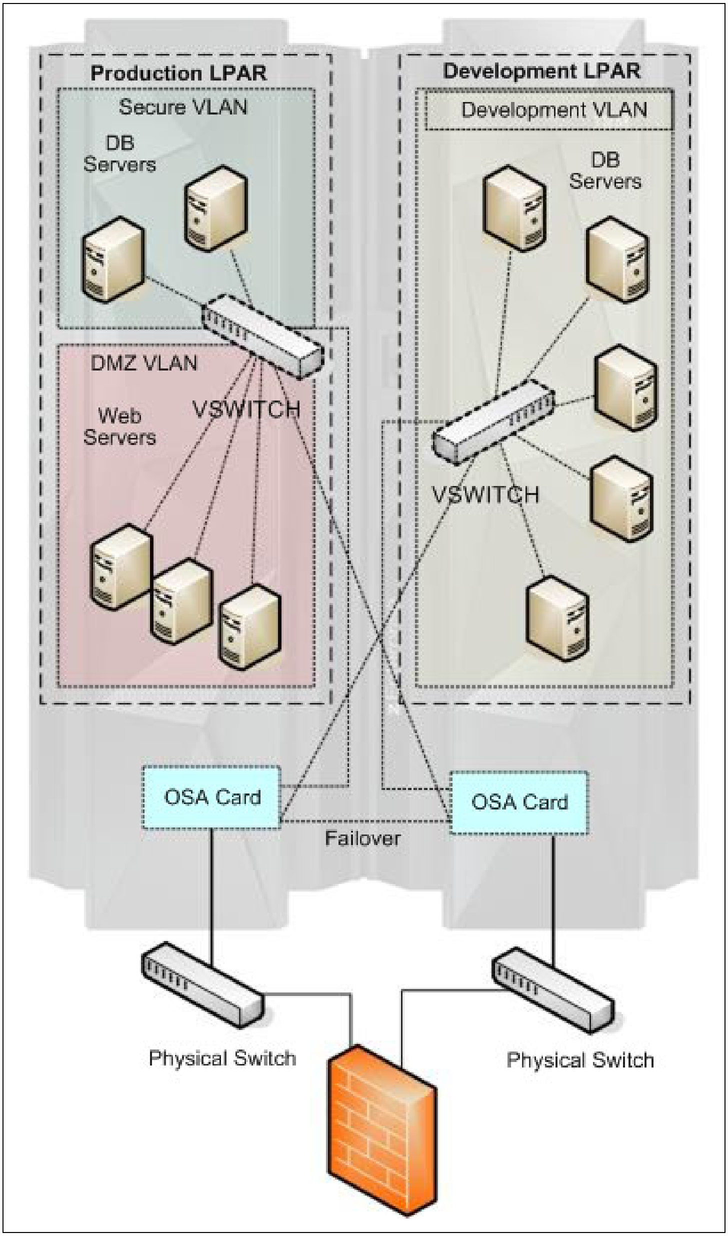


Figure 5-8 Multiple virtualized network scenario with failover: DMZ and secure network

You can isolate the entire secure network from the physical network segment by using multiple partitions. The communication between the partitions is managed by HiperSockets devices (see Figure 5-9).

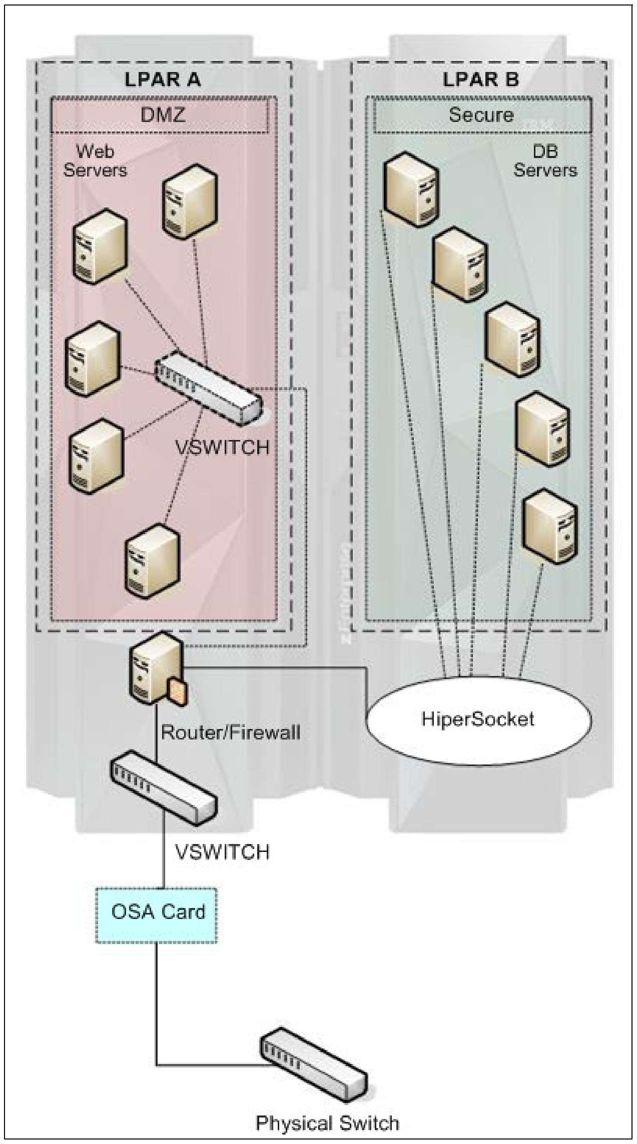


Figure 5-9 Multiple virtualized network scenario with multiple LPARs

Note: Although the use of HiperSockets for this scenario is possible, it might not be the best solution. If one of the LPARs is CPU-constrained, using HiperSockets might cause a delay of network traffic. For more information about HiperSockets, see *Set up Linux on IBM System z for Production*, SG24-8137.

VLAN segmented network

The use of the VLAN tag on virtual switches is supported. The VLAN configuration helps with the segmentation of network packages, bringing security and organization to the environment. It facilitates the administration of the network by grouping the guests with common requirements regardless of their physical location. The VSWITCH, like a physical switch, provides full authorization on a per port basis for membership in a VLAN segment.

For a high security network scenario, use the LPAR environment that is mixed with the multiple network segmented solution. As illustrated in Figure 5-8 on page 85, the entire LinuxONE environment is virtualized and all configurations are made per VM, which increases the security, reduces the complexity, and simplifies the environment.

5.1.4 Helpful steps for a network migration

The Linux administrators and network administrators should work together to engineer the best solution for your environment. Complete the following basic steps:

1. Determine the new IP addresses for the new servers. Each IP address should be on the same IP network to minimize the number of variables of the entire migration.
2. Determine the VLAN IDs of the Linux servers.
3. Configure the VSWITCH with the listed VLAN IDs.
4. Configure the Linux servers by using the designated IP addresses.

The target Linux server must be assigned a hostname that is different from the source server name:

1. Migrate the applications (for more information, see 5.3, "Application analysis" on page 97) and files from the source server to the target server.
2. Shut down the source server.
3. Change the Linux server's hostname.
4. Change the DNS registered name to the new Linux IP address.

If the application that is running is an IP-based application, you can change the IP address of the target Linux server to the source IP address.

5.2 Storage analysis

This section explains concepts and designs, such as online migration and offline migration, regarding the storage configuration possibilities for LinuxONE. Other storage migration issues are also covered.

5.2.1 Data migration

Two models of data migration are discussed in this section:

- *Online migration* refers to the case where the source server, target servers, and all services are up and running, and a system outage is not required.
- *Offline migration* requires a service outage to switch over from the source server to the target servers.

These migration models are covered in more detail in the following subsections.

In both types of data migration, some unexpected issues must be carefully considered. The result of not doing so might lead to an extended outage, unexpected downtime, data corruption, missing data, or data loss.

Online data migration

Some applications are eligible for online migration. To be eligible, an application must provide multi-operating system clustering support and be available on LinuxONE.

To perform an online migration, complete these steps:

1. Install and configure the target Linux guest. For more information, see 5.2.2, “LinuxONE: pre-installation considerations” on page 92.
2. Install the middleware application on the Linux guest.
3. Copy the application data to the target Linux guest.

   The software application selection depends on the type of data that needs to be copied. Solutions like the Linux **`scp`** program can be used in online data migrations where the application does not change or the changes are controlled.

   Otherwise, the Rsync software application can be used to synchronize the application data between the server in a small amount of time during the migration process.
4. Include the Linux guest in a cluster as a cluster node.
5. Monitor the Linux guest to verify that the application is responding to requests correctly.

   This step is not a test of the application on LinuxONE. The application must be tested on a development system to help ensure that the application is a LinuxONE compatible application (for more information, see 5.3, “Application analysis” on page 97).
6. Shut down the source servers.

Always consider the content of the data that is migrated before selecting online migrations as a solution.

To avoid such issues, online data migration must always be run during off-hours, and you should always take a data backup just before the actual data migration activity begins.

Offline data migration

Offline data migration can apply to all system migrations. This data migration can be accomplished by using several different approaches and functions:

- Using the network mount points NFS or Samba connections and either the DD or CP Linux command.
- Using an FTP server on the source or target server.
- Using an SCP/Secure Shell (SSH) server between the server and the target server.
- Using the Rsync synchronization application between the source or target server.
- Attaching the storage volume to a Fibre Channel device (Linux to Linux migration).

Using the Rsync application

For a better result when using the Rsync application, schedule service synchronization for an entire week before the outage by completing these steps:

1. On the first migration day, run the first synchronization.

   Run the first synchronization during a time when the use of the server is low. Rsync copies only files that are not locked, which avoids any issues with files in use. However, during this time server response might be slower than normal because of the extra read I/O activity.

2. During the migration week, you can run a daily synchronization on the server during off-peak hours.

   Only modified files are copied from the source to the target server.

3. The last synchronization day is the server outage day, when access to the source server is denied to users.

   Because there are no open files, the Rsync application is able to copy all files to the target servers.

4. Shut down the source servers and start all services on the target Linux servers.

Transferring files over the network

Database migrations are the most common example of the requirement for files to be transferred over the network because most database software needs an offline backup that includes a data export or data dump to a new file.

That exported or dumped file must be transferred across the network, and the database import procedure must be run at the target server. For more information, see 5.4, "Database analysis" on page 106.

Migrating storage volumes

When the source servers are Linux x86 and connected to an external storage device that uses Fibre Channel, and a zFCP device is part of the same storage area network (SAN), you can connect the source Linux volume to the target Linux guest on a LinuxONE server. However, both servers cannot share a volume at the same time.

Storage SAN Volume Controller

One option that is available to simplify storage and data migration for Fibre Channel disks that are involved in a migration to LinuxONE is to install the IBM SAN Volume Controller (SVC).

The SVC sits in the channel path and enables you to virtualize all Fibre Channel Protocol (FCP) storage from multiple vendors that sit behind it. Figure 5-10 shows where the SVC sits in the SAN. The SVC is visible to all supported storage on the SAN.

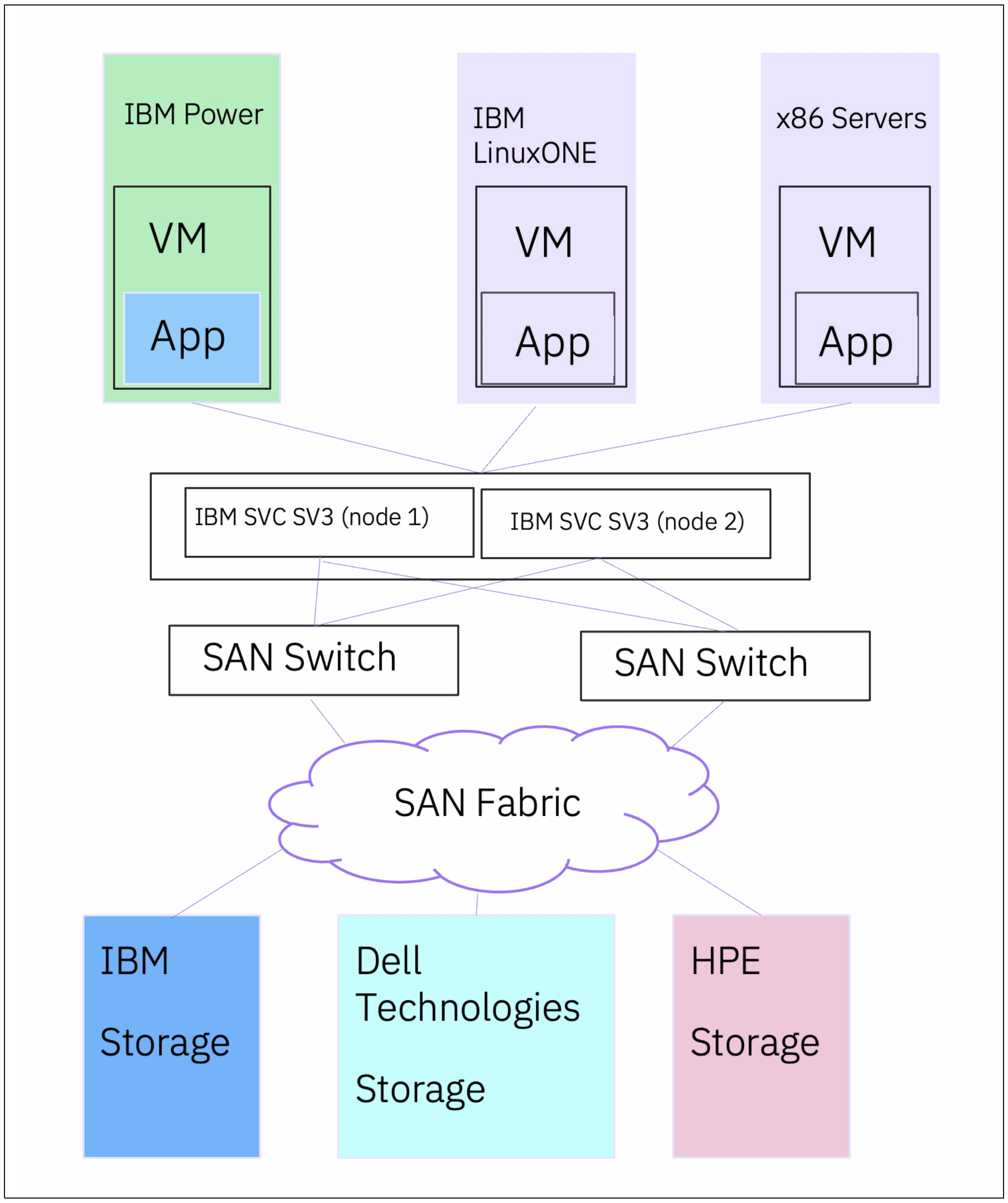


Figure 5-10 SAN Volume Controller

The following benefits are provided by the SVC:

- Single point of control for heterogeneous storage resources
- Dynamic data migration between heterogeneous storage devices on a SAN
- Ability to pool the storage capacity of multiple Storage Systems on a SAN
- Scalability to support up to 2048 host servers and up to 16,000 host mappable volumes (for SVC SV3)
- Instant copies of data across multiple Storage Systems with IBM FlashCopy®
- Copy data across metropolitan and global distances as needed to create HA storage solutions

- Up to 64 K snapshot generation can be created by IBM SVC (for SVC SV3)
- Up to 40 PiB of copies can be created, which enable more volumes to have more snapshots more often (for SVC SV3)
- A native crash-consistent snapshot scheduler that can create and manage both regular and Safeguarded copies
- Crash-consistent with Copy Data Management: Crash-consistent with a built-in scheduler and application-consistent with CDM

When migrating Linux systems from x86 to LinuxONE, the SVC can non-disruptively migrate data to LinuxONE. For more information about the IBM System Storage SVC, see IBM SAN Volume Controller.

For more information, see the following publications:

- *Implementation Guide for IBM Storage FlashSystem and IBM SAN Volume*, SG24-8542
- *Performance and Best Practices Guide for IBM Spectrum Virtualize 8.5*, SG24-8521
- *IBM SAN Volume Controller Model SV3 Product Guide (for IBM Storage Virtualize V8.6)*, REDP-5670
- *IBM SAN Volume Controller Best Practices and Performance Guidelines*, SG24-8502

Note: If you are using IBM Storage Virtualize with IBM FlashWatch, you can benefit from no-charge data migration for 90 days to seamlessly transfer your data from over 500 different storage controllers regardless of brand (IBM or other vendors).

Helpful steps for an auxiliary storage migration

The multiple possibilities provided by LinuxONE to store and access files lead to many types of solutions. The solution that you design for the target system dramatically affects the flexibility, efficiency, and performance of the migrated application.

For source applications that are on servers where storage is local or the external storage is not compatible with Fibre Channel data storage, all data must be copied by using the Network File System from the source server to the target server (LinuxONE):

1. Create a server file system with mount points for all data files.
2. Create a temporary file system to be used in the file transfer process on the target server.
3. Configure the target server as an NFS file server, a Samba file server, or an FTPS File Server to upload the files from the source server.

   Consider the following points:

   – If there is enough space at the source server to compact all data, consider using data compression features such as `zip`, or `tar` with `gzip` and `bzip` formats. Both of these formats are compatible with LinuxONE. The data can be transferred by using an FTP server that is configured on the target server.

   – If not enough space is available at the source server to compact the data, mount the NFS file system or map the Samba file system at the source machine, and copy the files across the network.
4. Verify the correct files permissions at the target directory. Adjust file permissions after the transfers for production work.

For file storage in an external storage system compatible with Fibre Channel, you can migrate to a LinuxONE server configured with zFCP adapters to connect directly to the volumes that should be migrated to LinuxONE servers.

5.2.2 LinuxONE: pre-installation considerations

The storage and file system design has a direct influence on system performance, system availability, and the capabilities for system expansion.

A best practice for LinuxONE is that only one version of a Linux OS distribution should be installed from scratch. Therefore, design the basic Linux file system to allow the highest possible model of servers and then clone all the other Linux guests in the environment from this source (known as the *golden image*). The file system that stores the application data is created after the cloning process, depending on the needs of the application that is on the server.

For more information about creating a golden image, see *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, SG24-8147.

Logical Volume Manager

All file systems, except the root (/) file system, should be created as Logical Volume Manager (LVM) devices. File systems that are created with an LVM make it possible to expand or reduce the file without a system outage.

The LVM is useful for Linux file systems because it you can dynamically manage file system size and has tools to help back up and restore failing partitions.

Basically, LVM volumes are composed of the following components:

- Physical volume

  A physical volume (PV) is a storage device, such as a Small Computer System Interface (SCSI) device connected over an FCP device.

- Logical volume

  A logical volume (LV) is the disk partition of the LVM system. An LV is the area that is formatted and is accessed by users and applications. The LV is exposed through a mount point.

- Volume group

  A volume group (VG) is the highest level of the LVM unit. A VG is created by one or more PVs and gathers the LVs.

Figure 5-11 on page 93 shows five minidisk devices that are used by a Linux guest to create a unique VG. It is then further organized or allocated into two LVs.

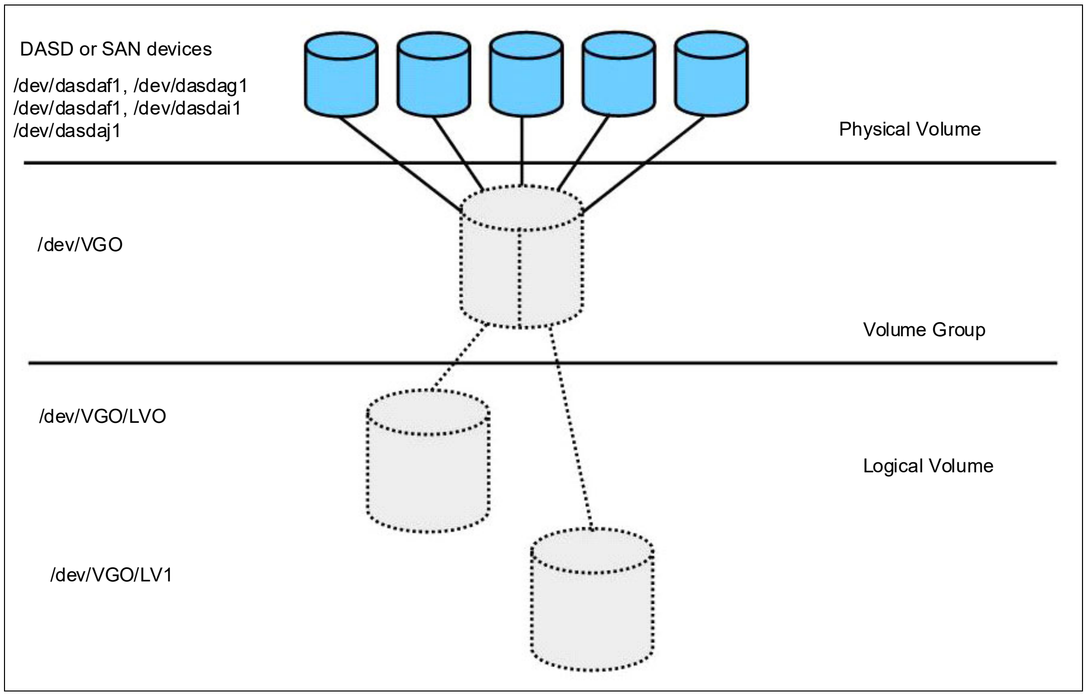


Figure 5-11 LVM example

However, a small performance price must be paid when using LVM. However, the flexibility of LVM often outweighs the cost of the performance loss.

For more information about the LVM setup during the installation, see *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, SG24-8147.

Linux file system

Design the basic Linux OS file system so that one single image (the golden image or prototype) can be cloned and used on as many Linux servers as possible.

It is a best practice to isolate each file system according to its purpose. This practice avoids space issues because each file system is designed and used differently. For example, the `/var` file system is critical for logging and creating temporary resources. If a user fills the `/home` file system, logging on `/var` is not affected.

Following this best practice, the golden image should include the following file systems:

- `root (/)`
- `/boot`
- `/var`
- `/tmp`
- `/opt`
- `/home`

Note: These best practices are better aligned to suit generic business needs. Plan your file system distribution according to your needs and consult your Linux distribution manual for further recommendations.

The following sections describe these file systems in more detail.

The root (/) file system

The `root` file system is the first file system to be created, and it is the base for all other file systems in the hierarchical structures of the Linux operating system. A size of 350 MB is enough for the root file system.

Important: The `root` (/) file system should not be placed on an LVM device because during an LVM failure, you can recover the system by using the single user mode.

The /boot file system

The `/boot` file system is often left as a subdirectory under root (/), but maintaining this directory structure as its own partition can be useful. The `/boot` file system contains the boot files, such as the kernel, the `parm` file, the initial ram disk, and the system map. In Linux, the `/boot` partition also contains the boot loader configurations, such as zIPL or GRUB. Because it holds the kernel files, it might be considered the most important partition of all. Keeping it as its own partition helps preserve its important status and maintain its integrity.

Important: Like `root (/)`, do not place the `/boot` file system on an LVM device. The preferred file system type for `/boot` is EXT3.

The /var file system

The `/var` file system is where all the variables files (such as spool files, cache files, and log files) are written. The `/var file system` has files that are constantly changing such as `/var/log/messages` and `/var/log/secure`.

The size of this file system depends on the number and type of applications that are running and how long the log files are kept on the server. Also, consider whether the application is designed to write files here, and their sizes and frequencies.

The services control files are also placed on the `/var` file system so it can never be scaled to be a shared file system and it must be always read/write.

Because it is a dynamic file system, place it on an LVM device to allow it to be extended or reduced as needed.

The /tmp file system

The `/tmp` file system was originally designed to store operating system and temporary application files. These files would be deleted every time that the system is restarted or deleted by the application immediately after the file is no longer in use. Some homemade applications use the `/tmp` file system as a dump area or an exchange file resource. In rare cases, the size of the `/tmp` needs to be increased.

Because it is a dynamic file system, place it on an LVM device to allow the capability to be extended or reduced as needed.

The /opt file system

Deploy all third-party applications in the `/opt` file system. As a preferred practice, further organize the `/opt` directory by the company or organization that developed the application or software. The next directory level then specifies the software package that is installed. For example, install a Db2 for Linux server at `/opt/ibm/db2`. Place a WebSphere Application Server in the `/opt/ibm/WebSphere` directory.

The file system size depends on the size of the software packages that are installed in it. It is simple to estimate the requirements for a single software package. However, upgrades, maintenance, and more software packages are not so simple to plan for. The `/opt` file system can also be a dynamic file system and should be configured on an LVM device.

The /home file system

The `/home` file system is designed to allocate user files. The size of the file system depends on the server function and the number of users who are defined on the server. For example, application production servers do not need a large `/home` file system because typically development staff store files on a production server. However, it *is* expected that applications will be developed on a development application server, so developers need sufficient file system space to create and transfer their files.

Depending on the situation, the `/home` file system can be a dynamic file system. If it is dynamic, configure it on an LVM device.

Other file systems

An example of extra file systems that might be created on a specific server during the migration process is the database server file system. Basically, you need to have at least one file system for data files and one for log files. Therefore, at a minimum two file systems must be created in addition to the file system where the application binary files are installed. For an IBM Db2 database server, the default location for the binary files is `/opt/ibm/DB2`.

Other database management systems (DBMSs) put their data files in other directories. For example, the MySQL database server's default location for data files is the `/var/lib/mysql` directory. If the server is a MySQL database server and you are using the Linux distribution from Red Hat Linux or SUSE Linux, consider including a new file system at the `/var/lib/mysql` mount point.

For each target database management server, make sure that you know where the binary files and the data files are located. Only with this information can you plan to create the devices and file systems for the target system.

There might be file location differences depending on the distribution of Linux that you install at your site. Make sure that you know these differences, if any, and plan for them.

More resources

For more recommendations, like VG and disk naming conventions, see *Set up Linux on IBM System z for Production*, SG24-8137.

Shared file system

The data storage in a LinuxONE environment can be shared physically by one or more Linux guests. However, because of limitations of the file system, it is not possible for two Linux guests to have read/write control to a device at the same time. However, this configuration might be possible at the hardware level.

In a shared disk environment, the file system changes that are performed by a guest machine that has the read/write control are available only to other guests that share the file system after unmount and mount of that system. As an example, think of the environment of a web cluster service where the application servers need only read access to the web pages and do not need to write to the same file system where the files are allocated.

In the example that is shown in Figure 5-12, only the special file system and mount points relevant to the solution are represented. The data file location is at the mount point `/srv/www/app`, which is the file system that is shared between the Linux guests. There is also the shared file system `/opt/ibm/IBMHTTP`, where the web server binary files are installed. For the IBMHTTP service, the log files are redirected to the local `/var/log/httpd` file system. All shared devices are the same device type, and are managed by the z/VM operating system.

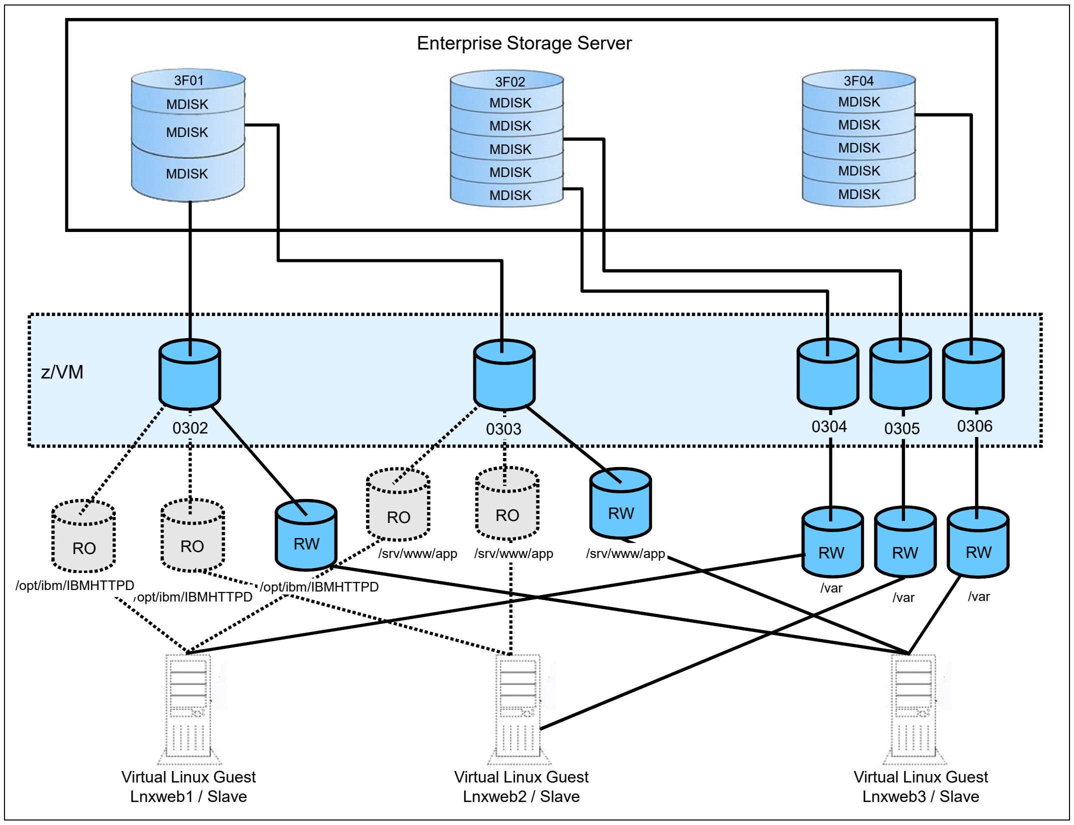


Figure 5-12 Shared devices example

The benefits of using a shared file system are based on economy of resource. You can reduce application binary space allocation and code updating efforts because you update only one master server and then remount it on the subordinate servers.

Note: System administrators must pay special attention to managing this environment because if the same file system is mounted as read/write in two different servers, all data might be lost.

Disk devices

A single Linux guest can talk to multiple disk devices of the same or different device type. This feature is helpful when using large file systems, such as in the case of database servers. If necessary, you can split up a single disk into partitions with the fdisk or fdasd Linux utilities, or into minidisks with z/VM.

A combination of both solutions can help you improve system performance and use storage resources efficiently. For more information, see *Linux on IBM System z: Performance Measurement and Tuning*, SG24-6926.

Software-defined storage

IBM LinuxONE supports IBM Spectrum Scale and other software-defined storage technologies. This newer virtualization abstracts the rich features that are found in a single enterprise storage system so that they become available across multiple storage facilities. This feature provides tremendous benefits in the clustering technologies that are used for high availability (HA) solutions and data replication or backup.

For more information about IBM Spectrum Scale, see this web page.

NVMe storage support

The LinuxONE platform supports an integrated storage option by featuring carrier cards into which NVMe SSDs can be plugged. It provides the low latency and high I/O throughput that can help with real-time analytics, memory-intensive and fast storage workloads, such as streaming, paging, and sorting, and traditional applications, such as relational databases.

For more information about Non-Volatile Memory express support on IBM LinuxONE, see *Maximizing security with LinuxONE*, REDP-5535.

5.3 Application analysis

This section describes the analysis that you must perform to identify applications that are good migration candidates to IBM LinuxONE.

5.3.1 Application architecture overview

A common way to ease the selection of migration candidates to LinuxONE typically involves a comprehensive architecture analysis of each application in your portfolio. It often consists of an architecture diagram that provides an abstraction of many aspects of an application.

An application’s infrastructure diagram helps you to understand the relationship among its interconnected components and assess the overall migration complexity. With a diagram available, it is possible to fully establish expectations, required efforts, and goals along with all involved stakeholders during the migration process, which typically speeds up the migration process.

Figure 5-13 shows an application's infrastructure architecture. Consider the following points:

- Five Linux servers are necessary.
- The WebSphere Application Servers and Db2 servers must be in the same network zone.
- The IBM MQ and Node.js systems must be in a restricted network zone.
- Only traffic from the WebSphere servers to the IBM MQ server must be permitted across the two network zones.

After the initial application's architecture assessment is complete and it is considered a candidate for migration, all stakeholders should have a deeper understanding about what the tasks that are necessary to achieve a successful migration.

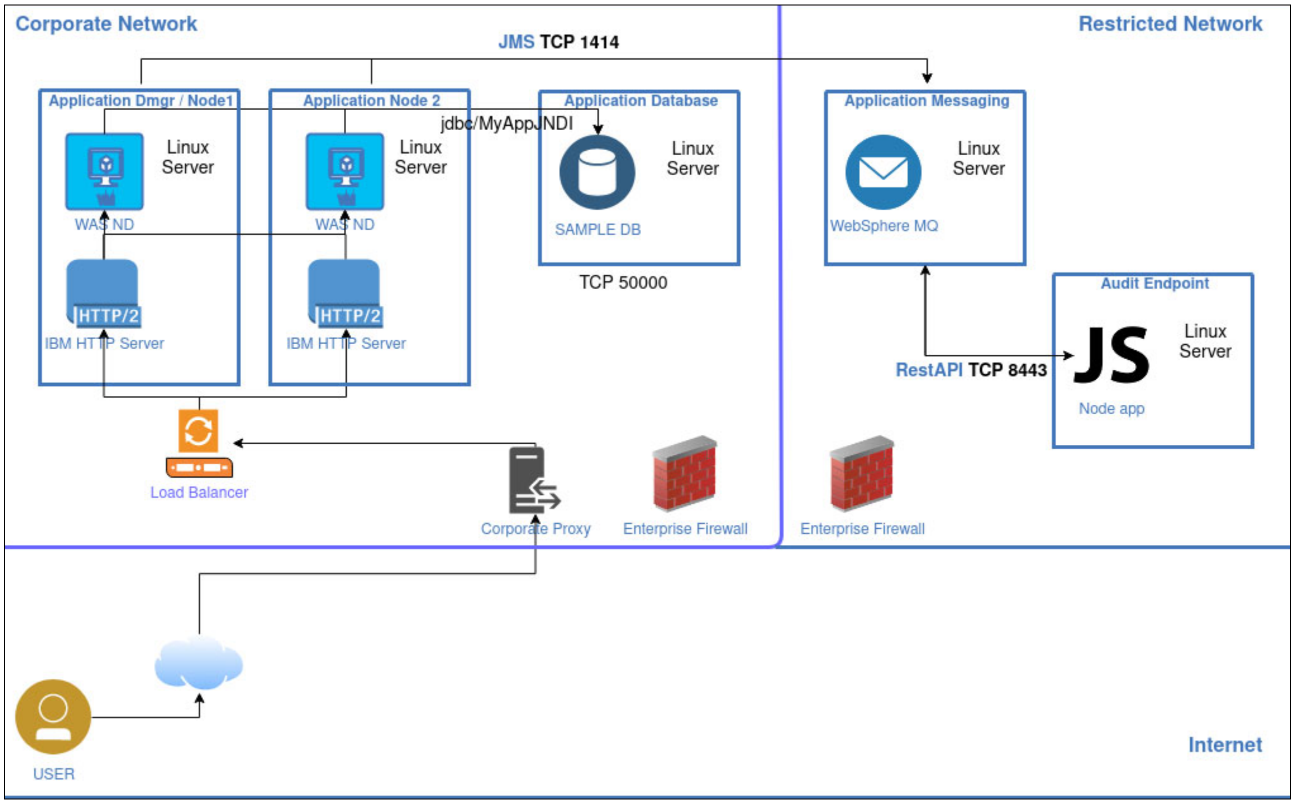


Figure 5-13 An application infrastructure diagram

5.3.2 Why migrate applications

Perform an application migration only after thorough planning. You also need a compelling reason to act, such as the following real-world situations:

- An application outgrew its original platform and is close to reaching the architectural limits of the platform.
- Software license costs are rapidly increasing as more servers are added to an application.
- Performance issues are arising between distributed application servers and centralized databases.
- Uncontrolled distributed server growth is leading to power and cooling issues in the data center.
- Complex distributed systems, which are costly to maintain, are suffering from increasing unreliability.

- New application development is required following a merger or acquisition.
- Regulatory requirements impose the need for a more secure environment.

Such situations present valid reasons for considering a migration to a more efficient platform like IBM LinuxONE. Usually, a migration to LinuxONE will help an organization realize significant cost savings over three to five years. The question is, which applications can you migrate and what risk factors are associated with the migration?

The output of this exercise is a list of an organization's applications ordered by complexity. The list is based on factors such as the number of servers or applications that make up the "IT systems", and can generally be grouped as large, medium, or small applications or number of servers.

5.3.3 Which applications can be migrated

Every computing platform offers specific areas of strength, and the aim of a migration should be to select applications that take advantage of the strengths of the target platform. The strengths of IBM LinuxONE include HA, high I/O bandwidth capabilities, the flexibility to run disparate workloads concurrently, and excellent disaster recovery (DR) capabilities.

Another key element in choosing the appropriate applications for migration is whether they are supported on LinuxONE. This consideration is normally not a problem with homegrown applications, depending on what language they were written in. Also, LinuxONE has a long list of supported open source applications available on the platform.

5.3.4 Selecting an application for migration to LinuxONE

This section lists and describes the basic rules for selecting an application to migrate to LinuxONE.

The following applications cannot or should not be migrated to LinuxONE:

- Applications that are available only on Intel platforms.

  Requesting independent software vendors (ISVs) to support their application on LinuxONE is a long process. However, they can join the IBM LinuxONE Partner Network, which is a program for ISVs to easily port, certify, and deploy applications on the IBM LinuxONE platform. ISVs also gain access to Red Hat OpenShift technologies for application development and a rich set of learning resources for skill development.
- Specialized workloads, such as graphics or sound processing.

  Graphics and sound processing typically requires specialized hardware and instructions that are not yet available under the platform. Although it is possible to run such type of workloads, the performance might not be beneficial for the user.

The following applications are suitable for migration:

- Applications or middleware (database, application servers, and others) that are supported by a software vendor on multiple platforms, including LinuxONE.

  There are no support issues and migration is simpler.
- Applications that need close proximity to data on IBM LinuxONE, or that are components of LinuxONE applications.

  You can boost the performance and speed of your Linux applications by putting them on the same physical server as their data source.

- Applications with high I/O or transactional I/O.

  Because of its design, LinuxONE excels at handling sustained high I/O rates.
- Applications with lower sustained CPU peaks and average memory needs.

  These workloads are ideal for LinuxONE. The platform is designed to run multiple workloads at a consistently high CPU and memory utilization.
- Application development environment for Linux on other platforms.

  The virtualized LinuxONE platform provides an ideal environment to test applications before their deployment to Linux on other platforms.

5.3.5 Best-suited applications for migration

The applications that are described in this section gain the most benefit from the LinuxONE platform strengths, including HA, high I/O bandwidth capabilities, the flexibility to run disparate workloads concurrently, and excellent DPM characteristics.

IBM Software

IBM has many of its software products available for LinuxONE. The benefit to customers is that a migration from one platform to another is often simpler because many of these products share their code base across multiple platforms.

Generally, migrating from IBM products on distributed servers to the same IBM products on LinuxONE is a relatively straightforward process. For more information and examples, see Chapter 6, “Hands-on migration” on page 147.

Db2

You can use Db2 for Linux, UNIX, and Windows products on LinuxONE. It works seamlessly in the virtualized environment without any extra configuration. In addition, autonomic features, such as self-tuning memory management and enhanced automatic storage, help the database administrator (DBA) to maintain and tune the Db2 server. For more information and a migration example from x86, see 6.2, “Migrating Db2 and its data” on page 154.

Oracle

Because the Oracle database is fully supported on LinuxONE and runs efficiently on this platform, it is a good candidate for migration to LinuxONE.

Oracle databases on LinuxONE also support Oracle Real Application Clusters (RACs), the Oracle HA clustering solution. The advantages for Oracle RAC on Linux are a high-availability cluster with low latency within the LinuxONE platform that is combined with HiperSockets for inter-LPAR communication.

Oracle WebLogic Server is also supported on LinuxONE. You can use to have a complete Oracle Java environment and high available Oracle database within the same LinuxONE machine.

Often, Oracle supports a mixed configuration mode where the database tier sits on Linux and applications for Oracle E-Business Suite, Oracle Siebel, and Oracle Business Intelligence run on distributed servers under Linux, Windows, or UNIX. For more information about which Oracle products are certified for LinuxONE, contact your Oracle representative or see this web page.

Big data Hadoop solutions on LinuxONE

Big data and analytics solutions that are built on the IBM LinuxONE platform harness the data explosion that is facing businesses. IBM took a leadership role in offering optimized solutions that are ready for immediate deployment. These solutions are built on software, such as IBM BigInsights® and IBM Streams.

IBM also took its leading role in the open source community seriously. IBM made important contributions to projects, such as Apache Hadoop, which enabled continuous development in the fields of analytics and high-performance computing. Clients and solution builders that want to innovate on top of a high-performance data analytics platform can take advantage of the flexibility, throughput, and resiliency of IBM LinuxONE Platform, and the immediate price-performance value that is provided by LinuxONE solutions.

MongoDB solutions on LinuxONE

MongoDB’s NoSQL technology eliminates the processor burden of object-relational mapping. It enables developers to build and deploy modern applications rapidly, without having to define a data schema in advance and contend with its restrictions. The main features of MongoDB include flexible data modeling, cloud and on-premises cluster management and automation, expressive query language, always-on global deployments, scalability, and high performance.

A LinuxONE server that is running Node.js and MongoDB can handle over 30 billion web events per day while maintaining 470 K read and writes per second. The MEAN stack runs up to 2x faster than on other platforms. IBM LinuxONE allows MongoDB to scale vertically with dynamically allocated resources instead of horizontally by sharding and replicating the database. LinuxONE and MongoDB provide strong consistency, which helps ensure that critical data remains consistent and minimizes sharding-related processor usage.

5.3.6 Other software

This section describes some non-IBM software that is good candidates for migrating to LinuxONE.

Infrastructure services

The following infrastructure services are good candidates for LinuxONE:

- Network infrastructure services, such as FTP, NFS, DNS, are well served on LinuxONE. These workloads are minimal, but are critical to the business. The main benefit of hosting these services on LinuxONE is the availability of the hardware’s DPM capabilities.
- Lightweight Directory Access Protocol (LDAP) security services fit well running on LinuxONE, including OpenLDAP products and commercial products, such as IBM Security® Directory Server, IBM Tivoli Directory Integrator, and IBM Tivoli Access Manager. By using LinuxONE, customers can build a robust identity services-oriented infrastructure.

Application development

LinuxONE provides the following benefits for application development:

- Whether for Java, C/C++, or most other programming languages, a virtualized Linux environment is an ideal platform for application development. Although developers usually develop on a stand-alone platform, testing and modifying are performed in a server environment. Developers can be given multiple virtual servers to perform interactive testing while troubleshooting or enhancing the application.
- Other major benefits include the ability to rapidly deploy virtual servers for user acceptance testing and integration testing and, when that process is finished, the virtual servers are shut down. If a developer inadvertently “damages” a virtual server, a new server can be cloned. You do not need to spend a great deal of time formatting disks and reinstalling the operating system and required applications.
- For new applications, virtual servers are deployed quickly and can be easily customized for a specific purpose. Many customers have standard server profiles that are pre-built; therefore to create another virtual server, only the suitable profile must be cloned, which can be done in minutes. When an application is discarded, the virtual servers also can be discarded.

5.3.7 Selecting an application for a proof of concept

When choosing an application for a proof of concept (POC), keep it as simple as possible. This simplicity is needed because a POC is performed to demonstrate that an application can also be successfully migrated to a LinuxONE environment. It also demonstrates that the application results are the same (or better) as the source’s production system.

Select an application that is reasonably self-contained and that does not rely too much on input from multiple sources and other applications. In addition, choose an application that does not require a major rewrite to run on LinuxONE.

The best candidates are Java-based applications because these applications are platform-independent. However, if you are moving to a different Java Platform, release or a different middleware product, some code changes might be necessary.

Applications that are written in C/C++ are also suitable if the source code is available. However, keep in mind that these applications must be recompiled for the IBM LinuxONE platform.

After you select an application to migrate, clearly define your goals and expectations. The POC’s results should achieve the same performance, usability, and functions as the source production environment.

5.3.8 Application interdependencies

Few applications are self-contained. Usually, an application obtains data from several other applications and its output is sent on to other applications. These applications can also be on different platforms and are often from entities outside your organization. A migration to LinuxONE provides an opportunity to simplify your infrastructure without affecting any interdependencies.

Many distributed applications grew in only a few years from a single server to tens or even hundreds of interconnected systems. These interconnected servers not only add network burden, but complexity and built-in fragility. If such an application is being considered for migration, make simplification part of the core of what needs to be done.

Because LinuxONE supports all modern communication methods, it is a straightforward process to receive data inputs and transmit data outputs in the same way as before the application was migrated. In this case, no changes to external applications are needed.

Note: The main thing to remember during migration planning is to completely map all application interdependencies. The aim is to identify any obsolete networking technologies and interfaces, which might in turn require another application to be migrated to a current network technology.

5.3.9 Successful application migration

This section outlines the considerations to keep in mind and the steps to follow to help you on a successful application migration for Java and C/C++ programs.

5.3.10 Special considerations for migrating a Java application

Migrating Java applications from one platform to another is simple compared to the migration effort that is required for C or C++ applications. Although Java applications are operating system-independent, the following implementation and distribution specifics must be considered:

- Most of the Java distributions have their own Java virtual machine (JVM) implementations. Differences exist in the JVM switches. These switches are used to make the JVM and the Java application run as optimally as possible on that platform. Each JVM switch that is used in the source Java environment must be verified for a similar switch in the target Java environment.
- Although Java SE Developer Kits (JDKs) are expected to conform to common Java specifications, each distribution features slight differences in the helper classes that provide functions to implement specific Java application programming interfaces (APIs). If the application is written to conform to a particular Java distribution, the helper classes that are referenced in the application must be changed to refer to the new Java distribution classes.
- Special procedures must be followed to obtain the best application migration. One critical point is to update the JVM to the current stable version. The compatibility with earlier versions is significant, and performance improvements benefit applications.
- The Java version lifecycle follows a predictable and defined cadence, and it provides a clear roadmap for developers and businesses that use the Java platform. New major releases occur every 6 months and are denoted by a three-digit version number, such as Java 18, 19, 21, and so on. These releases introduce new features, APIs, and improvements. The source (x86) and target (LinuxONE) should use the same and supported Java versions for seamless migration and opt for Long-Term Support (LTS) releases for extended stability and security.
- Enable the just-in-time (JIT) compiler.

- Generally, JVM aligns its page size with the native page size of the OS. On most modern systems, this page size is typically 4096 bytes. This harmony helps ensure efficient memory management and smooth virtual memory allocation by the JVM, which maximizes performance and minimizes overhead. Another advantage of LinuxONE over x86 regarding JVMs is the usage of large pages. On LinuxONE, you can use 1 MB pages, which reduce the number of pages and saves many CPU cycles for searching for this data in memory. To enable large pages, use the**-Xlp** flag along with your JVM, as shown in Example 5-1.

Example 5-1 Enabling large pages

```
$ java -Xlp -jar mylxapp.jar
```

- For JVMs with a high memory consumption footprint, set the minimal heap size (**-Xms**) equal to the maximal heap size (**-Xmx**). The size of the heap size should be always less than the total of memory that is configured for the server.
- LinuxONE runs Java code faster than x86 platforms because of its Pause-less Garbage Collection function, which enables applications to continuously run alongside the garbage collection (GC) process.

Note: Starting with IBM Java 8 SR5, enable the Pause-less Garbage Collection feature by using the **-Xgc:concurrentScavenge** argument to your JVM. For more information about how the Pause-less Garbage Collection feature works, see the IBM Java SDK documentation.

Understanding the “stop the world” phenomenon

Java built-in memory management can be a double-edged sword. Although it simplifies memory handling, “stop-the-world” GC pauses can impact application performance and scalability. These pauses halt the entire application, which leads to sluggish response times and hinders the application's ability to handle increased workloads.

LinuxONE offers industry-leading Java performance through several key features (Figure 5-14):

- Pause-Less Garbage Collection: This feature eliminates “stop-the-world” pauses during GC, helping ensure that your applications run faster compared to x86 alternatives.
- LinuxONE enhanced cryptographic capabilities further boost performance so that you can handle demanding tasks.
- Unlike x86 systems, LinuxONE offers superior vertical scalability, enabling you to handle heavy Java workloads effortlessly by adding more resources to a single server.

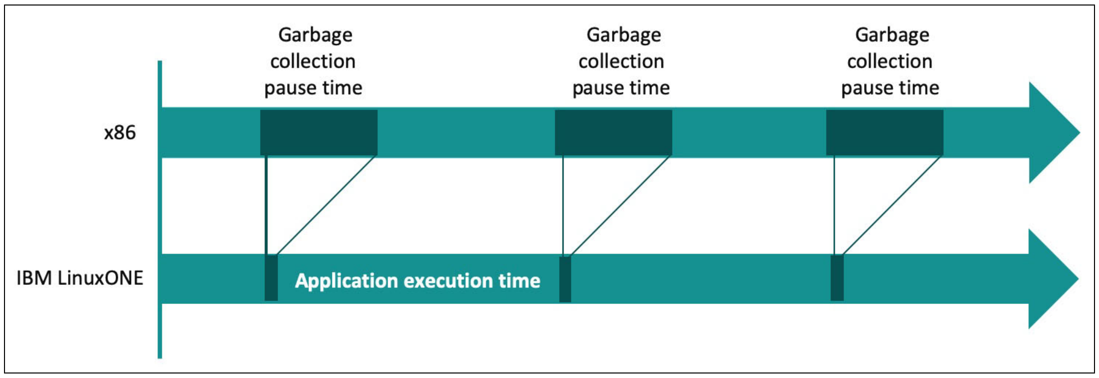

Garbage collection pause time
Garbage collection pause time
Garbage collection pause time
x86
IBM LinuxONE
Application execution time

Figure 5-14 GC on LinuxONE versus x86 servers

5.3.11 Special considerations for migrating C++ applications

When migrating C++ applications, you must be aware of a few special considerations, as explained in this section.

Architecture-dependent code

Programs that are in directories (on non IBM LinuxONE systems) with names, such as `/sysdeps` or `/arch` typically contain architecture-dependent code. Reimplement them for the hardware architecture to port any of these programs to LinuxONE.

Assembler code

Any assembler code must be rewritten. Opcodes must be changed to IBM S/390® opcodes or, if the code uses assembler header files, you need a suitable version of the header. Linux assembler code for Linux uses the S/390 opcodes, but follows the syntax conventions of GNU assembler. The GNU assembler manual is available at this web page.

The ptrace and return structure

Exercise caution when the ptrace and the return structure are used because they are architecture-dependent.

Little Endian to Big Endian

LinuxONE is a Big Endian system that stores multibyte numbers with the most significant byte at a lower address. Meanwhile, x86 servers are a Little Endian system, storing the most significant byte at a higher address. Any code that processes byte-oriented data that originated on a Little Endian system might need some byte-swapping. The data might have to be regenerated or, if that is not possible (for example, shared files), the application might need to be reworked to adjust for processing Little Endian data.

Changes to build scripts

Make suitable changes or updates to the `Configuration/build/Makefile` scripts or files, and a requirement to add support for the LinuxONE platform.

The /proc file system

The proc file system features the following differences:

- `/proc/cpuinfo` format is different
- `/proc/interrupts` is not implemented
- `/proc/stat` does not contain INTR information

Available languages and compilers

Many programming languages are available, such as Ruby, Perl, Go, and Python.

5.3.12 Middleware, libraries, and databases

Any middleware or libraries that are needed must be available for LinuxONE. Supported databases include examples of MariaDB, Postgres, Oracle, and Db2. As described in 5.3.5, “Best-suited applications for migration” on page 100, many middleware software is available for LinuxONE, such as Apache Tomcat, Red Hat JBoss, Oracle WebLogic, and more. For more information about how to install it on the IBM LinuxONE platform, see your product’s documentation.

5.3.13 Helpful steps for an application migration

A successful application migration depends on the combined efforts of the developer team, network team, systems administrators, and any other required technical stakeholders. Without the cooperation of all these groups, it is difficult to achieve a successful migration.

The following overall process might be helpful during your migration:

1. Perform source application mapping.

   Start by analyzing the source application, focusing on its suitability to migrate. Consider the following points:

   – Is the source code available to be compiled and deployed on the target server?

   – Is there a version of the middleware available for LinuxONE?

   – Are there performance reports of development tests to compare with after the migration?
2. Design the network solution for the application (see 5.1, “Network analysis” on page 76).
3. Design the file system for the application and middleware (see 5.2, “Storage analysis” on page 87).
4. Clone the Linux server (or servers) from the golden image.
5. Configure the network at the target server (or servers).
6. Create the custom file system at the target server (or servers).
7. Install and configure the middleware at the target server.
8. Copy the application code from the source to the target server.
9. Compile and deploy the application code to the target server.
10. Provide the first application test reports.
11. Start the performance test on the target server to understand the performance of the migrated application.
12. Size the CPU and memory to fit the migration expectations.
13. Run the application stress test.
14. Shut down the source server.
15. Change the IP address and hostname of the target server, or change the DNS configuration to the target application server.

5.4 Database analysis

This section provides information about the configurations of the database server on LinuxONE. Preferred practices for different DBMSs are also presented. Although this topic is presented by using offline migration methods, consult your DBMS vendor to review the various tools that are available to move data while online.

5.4.1 Before database migration

Database servers are well suited for migration to LinuxONE. However, a migration of a database server also demands detailed planning because technical configuration changes must be considered.

During the migration planning discussions, the workload of the instances and the databases that are running at the source environment must be considered, along with the number of concurrent users and the number of instances and databases that are running in a unique source server.

5.4.2 Migrating a single instance

For single instance servers, migration is fairly simple because the number of the variables that are involved from the source to the new destination environment is relatively small. Complete the following steps when migrating to the same database release:

1. Configure the LinuxONE network (follow steps 1 - 4 as described in 5.1.4, “Helpful steps for a network migration” on page 87).
2. Configure a temporary storage area at the source server and at the destination server.
3. Stop the database.
4. Export or dump the database data from the source server.
5. Transfer the exported data to the destination LinuxONE server.
6. Import the database data at the destination server.
7. Run database and applications tests.

5.4.3 Migrating multiple instances

For multiple instances that are running on a single server, or multiple instances on multiple servers, migration is more detailed and complicated. The benefits of migrating several instances to IBM LinuxONE include lower licensing costs, less data center space requirements, improved energy savings, and better performance.

Migrating multiple servers to LinuxONE

A significant factor to consider before the migration of multiple servers to LinuxONE is the peak load distribution during the service’s normal operations. Document and compare the system performance, including how the system resources, such as CPU and memory, are being used. Use Table 5-1 to map the workload consumption during the migration planning.

Table 5-1 Sample database server workload map

Server information			**Peak load measure**		**Peak load time**		
Name	**Total of CPU**	**Total of memory**	**%CPU used**	**% mem. used**	**Week day**	**Start time**	**Stop time**

A suitable capacity plan is the key to successfully migrate multiple databases to a single LinuxONE LPAR. An important point to consider when planning the capacity is the relationship between the source and target server resources. Typically, less resources are required in IBM LinuxONE servers when compared to their same workloads running under the x86 architecture. For more information about some of these aspects, see 5.4.4, “Technical considerations” on page 108.

Migrating a multiple instance server to LinuxONE

Consider the following questions when you migrate from a multiple instance server to multiple Linux virtual servers on LinuxONE:

- Is the source server running at maximum CPU capacity?
- Is the CPU usage balanced across all instances?
- Is there a unique instance that is using all of the CPU?
- What is the average CPU cycle that is used by each instance?
- During which period does the instance use more CPU cycles?
- Does the instance write or read more data onto the disk devices?
- How much memory does each instance have allocated?

Use Table 5-1 on page 107 to map the performance of your database instances. Change the Server name column to “Instance name” and document the appropriate information in its respective fields.

With this information, multiple servers can be migrated into a single LinuxONE LPAR while serving all user requests, with improved database response times and enhanced management. This process makes it simple to define the number of virtual CPUs that each server needs and avoid CPU constraints during peak usage hours.

Tip: If possible, gather data for an entire month instead of a single day. The more data that is available, the more accurate the performance analysis.

5.4.4 Technical considerations

Database management software requires careful analysis when planning for migration. Most database servers use shared memory segments and semaphores for communication.

Databases also use buffer pages to speed up table access. Database servers are memory and storage-bound and require proper capacity to operate efficiently and serve its users.

CPU

The number of virtual CPUs that are allocated to a database server is important. However, configuring the number of virtual CPUs to the same number of real CPUs does not guarantee better performance.

The number of processes in a processor queue is directly influenced by all other resources competing for CPU time. Typically, when many processes are competing against each other for CPU access, it is said that the system is “busy” and cannot serve all requests effectively. Memory or I/O constraints can also affect the processor queue directly.

Before deciding that the server does not have enough CPUs, analyze the CPU access times with Linux performance tools, such as `sar` and `top`.

Memory

Databases typically require a large memory area to achieve acceptable performance. Depending on the database and the amount of memory that is required, consider using large or huge pages. Memory access-intensive applications that use large amounts of virtual memory might obtain performance improvements by using large or huge pages.

IBM LinuxONE features impressive memory access speed times. During the migration of a database server, allocate less memory to the IBM LinuxONE machine. Then, increase or decrease it as necessary.

Swap memory

For most database servers, swapping should be avoided. Many database ISVs often recommend setting kernel parameters to completely avoid the system from swapping.

A LinuxONE preferred practice is to use z/VM VDISKs devices as swap devices. Because swap that is configured at VDisk devices provides desirable response times, the eventual memory paging (the process that moves memory blocks to and from real memory and to and from swap memory) is not considered a real problem. It is also not considered a problem if the server has no more than 50% of the swap memory allocated. However, this configuration involves variable paging and swapping allocation, which must be monitored to avoid database outages.

If the server often presents long periods of swapping, increase the guest's memory and continue monitoring the server to find its best memory size.

The Linux kernel provides a configurable kernel parameter that is called `vm.swappiness`, which determines whether more or fewer pages of memory are to be swapped in and out to disk. Consult your database product documentation and your DBA for more information about how to correctly tune this value to the wanted workload.

Table 5-1 on page 107 shows how to configure the **`vm.swappiness`** parameter of the `/etc/sysctl.conf` file.

Example 5-2 Configuring a swappiness value of 5 in the /etc/sysctl.conf file

```
# echo ‘vm.swappiness=5’ >> /etc/sysctl.conf
# sysctl -p
```

The correct swap size depends on your database requirements and how much memory it uses. The swap memory is used during high peaks only; therefore, set your swap size to a safe number to avoid an out of memory condition. As with memory sizing, frequently monitor the overall swap consumption and increase or decrease it as necessary.

Shared memory

Linux systems use the interprocess communication (IPC) facility for efficient communication of processes without the kernel intervention. The IPC uses three resources to allow communication among processes: Messages queues, semaphores, and shared memory.

Shared memory is a memory segment that is shared by more than one process. The size of the shared memory directly influences database performance because it can allocate more objects in real memory, which allows the system to perform less I/O.

Several Linux Kernel parameters can be tuned to improve the memory allocation to databases, depending on the workload requirements. Table 5-2 shows the best practice kernel parameters for a Db2 database.

Table 5-2 Guidelines for kernel parameters

Parameter	Description	Guideline
`kernel.shmmax`	Defines the maximum size of one shared memory segment in bytes.	A total of 90% of the total memory; however, if a large amount of storage is available, you can leave 512 MB - 1 GB for the operating system.
`kernel.shmall`	Defines the available memory for shared memory in 4 K pages.	Convert the shmmax value to 4 K (shmmax value x 1024 /4).

Parameter	Description	Guideline
`kernel.shmmni`	Defines the maximum number of shared memory segments.	A total of 4096. This amount enables large segments to be created, which avoids the need for thousands of small shared memory segments. This parameter varies depending on your application.
`kernel.sem`	Four values must be set in this parameter: ► The number of semaphores ► The maximum number of semaphores ► The maximum number of semaphores operations within one semop call ► The limit on the number of allocatable semaphores	250 256000 32 1024
`kernel.msgmni`	Maximum number of queues on the system.	1024
`kernel.msgmax`	Maximum size of a message in bytes.	65536
`kernel.msgmnb`	The default size of a queue in bytes.	65536

Note: Consult your database product documentation for its best practice kernel parameters and other tuning that might be necessary.

Storage

Data storage access on a database server is intensive and must be considered during server migration. To take advantage of the I/O processing capabilities of LinuxONE, the first consideration in design is to spread the I/O workload over as many paths as possible to the storage server.

For more information about how disk device accesses are made and how an external storage system provides its own disk page caching, see 3.8.3, “Virtualized disk” on page 58.

5.4.5 Migrating Db2 and Oracle from x86 to LinuxONE

The following sections provide an overview of the steps that are needed to migrate Db2 and Oracle from x86 to IBM LinuxONE. For more information about a full example of migrating your Db2 data, see Chapter 6, “Hands-on migration” on page 147.

Migrating Db2 databases across platforms

Although Db2 has many different ways of migrating the data from one operating environment to another, the simplest and most flexible way of migrating the data is by running the **`DB2MOVE`** command with the **`INSERT`** or **`LOAD`** parameter.

Four file formats are supported for import and export. The format that is chosen usually reflects the source that it comes from or the target tools to be used. Often, the files’ extension, such as `.ixf`, `.del`, or `.asc`, reveals the content format. For example, a file that is named `employee.ixf` contains uneditable Db2 interchange format data. Import can traverse the hierarchy of tables in `.ixf` format.

The following steps present a general overview of how to move an archived database between platforms:

1. Connect to the source Db2 database.
2. Use the export utility to export the database to any of the file formats that are supported by Db2.
3. Import the exported file to the target environment.

Migrating Oracle databases across platforms

Before Oracle 10g, one of the only supported ways to move an Oracle database across platforms was to export the data from the database and import it into a new database on the new server.

This method can require an excessive amount of downtime if your database is large. Oracle developed the following methods to migrate from one hardware platform to another one:

- Transportable table spaces: This technique was introduced in Oracle 8i to allow entire table spaces to be copied between databases in the time it takes to copy the data files.
- Data Pump export/import: These high-performance replacements are for the original Export and Import utilities.
- Backup and recover manager (RMAN): An Oracle Database client that performs backup and recovery tasks on your databases and automates administration of your backup strategies.
- Oracle GoldenGate: A comprehensive software package for real-time data integration and replication in heterogeneous IT environments.
- Oracle Streams is a technology within the Oracle Database suite that facilitates data sharing and replication between databases.
- IBM Data Replication is a software solution that is designed to synchronize and maintain consistent copies of your data across diverse data stores on different platforms and within your organization.
- Oracle Transportable Tablespace migration is a technique for efficiently moving large amounts of data between Oracle databases. It offers advantages over traditional import/export methods, especially for large data sets.
- Custom procedural approaches.

Note: The Transportable Tablespace migration process uses both Data Pump (for metadata export/import) and RMAN. Use RMAN Convert (convert the table spaces to the correct Endianness) to help ensure data compatibility before migrating table spaces.

5.4.6 Tips for a successful migration

Almost all databases use buffer pools in the shared memory area to manage the database memory context. Avoid the use of any automatic memory management systems to allocate shared memory. For example, if 6 GB of shared memory must be allocated to the database, force the database to allocate all memory at the system start.

If the database is not using all available memory, reduce the server memory until it starts paging. A system that is constantly swapping is the first indication of insufficient memory.

High load averages are not always an indication of CPU bottlenecks. Monitor the LPAR performance and determine whether any other process or server that is running in the same LPAR is competing for CPU time when the problem occurs.

Database data files and log files must be in different file systems and should be striped across the storage hardware. Have multiple paths to the data to help ensure availability.

The systems administrator and the DBA must work together during the sizing process. Database servers typically require adjustments at the Linux and database levels.

5.5 Backup analysis

This section provides a conceptual approach to migrating backed-up data from an operating environment to IBM LinuxONE.

5.5.1 Introduction to backup and archival concepts

This section provides a high-level introduction to the basic data and storage management paradigms that are used in the IT Industry. It covers data protection or backup, record retention or archiving, storage management, and security.

Backup concepts

The term *backup* refers to the creation of an extra copy of a data object to be used for operational recovery. The selection of data objects to be backed up must be done carefully to help ensure that, when restored, the data is still usable.

A data object can be a file, a part of a file, a directory, or a user-defined data object, such as a database table. Potentially, you can make several backup versions of the data, each version at a different point in time. These versions are closely tied together and related to the original object as a group of backups. The files are backed up by using backup operations, which typically occur daily, whenever the file is changed. The most recently backed-up file version is designated the “active” backup. All other versions are “inactive” backups.

If the original data object is corrupted or lost on the system, *restore* is the process of recovering the most current version of the backed-up data. The number and retention period of backup versions is controlled by backup policy definitions.

Old versions are automatically deleted as new versions are created under the following circumstances:

- The number of versions that are stored exceeds the defined limit
- After a defined period

On any system, several categories of data objects must be backed up, each with different retention needs. A database table can be backed up frequently at regular intervals, whereas an operating system configuration file is backed up only when it is changed.

Common backup types

The following most common types of backups are used:

- Normal

  This backup copies all selected files and marks each as having been backed up. With normal backups, you need only the most recent copy of the backup file to restore all files.

- Incremental

  This type of backup backs up only those files that were created or changed since the last normal or incremental backup. Performed periodically, this backup is faster than a normal backup and marks files as being backed up to assist with the next incremental backup. If you use a combination of normal and incremental backups, you need the last normal backup set and all the incremental backup sets to restore your data.

- Daily

  A *daily* backup copies all selected files that were modified on the day that the daily backup is performed. The backed-up files are not marked as being backed up.

Archiving concepts

Archiving refers to creating a copy of a file as a separate object in the storage repository to be retained for a specific time. Typically, you use this function to create an extra copy of data to be saved for historical purposes. For this reason, give special consideration to this task to help ensure that the data format is not dependent on anything. Vital records (data that must be kept because of government regulation, compliance, legal, or other business reasons) are likely candidates for the archive process.

The difference between a backup and an archive software is that the backup creates and controls multiple backup versions that are directly attached to the original client file, whereas the archive creates an extra stored object that is normally kept for a specific time, such as vital records.

5.5.2 KVM backup

Because the KVM hypervisor is a module that is loaded into the Linux kernel, the same tools that are described for a Linux guest in 5.5.4, “Linux backup” on page 114 apply to Linux guests that act as a KVM host. These tools can be used to back up the KVM hypervisor and its system configuration.

In addition, the KVM snapshot and managed save functions can be used to save the state of its managed Linux guests.

5.5.3 z/VM backup

The z/VM hypervisor can use utilities, such as IBM FlashCopy and DDR, to back up entire volumes. It is possible to perform a backup of entire Linux volumes when these methods are used. However, because Linux caches its writes in memory before writing changes to disk, it is important to help ensure that the guest is shut down or that the volumes to be backed up are mounted read-only at the target server before the backup takes place to ensure that no data is lost.

For more information about options for backing up data within Linux that offer better flexibility, see 5.5.4, “Linux backup” on page 114.

IBM Backup and Restore Manager for z/VM

IBM Backup and Restore Manager for z/VM is a complete solution to back up and restore data for Conversational Monitor System (CMS) or non-CMS systems (one file, a group of files, or an entire minidisk) in a VM environment. When integrated with the Tape Manager for z/VM, it can compress data during the backup, and support encryption exits.

For more information, see this web page.

5.5.4 Linux backup

Various methods can be used to perform backups with Linux. Whichever method is selected, the output must be a consistent data set that is usable when a recovery is necessary. These methods include CLI tools that are included with every Linux distribution, such as **`dd`**, **`dump`**, **`cpio`**, **`rsync`**, and **`tar`**. These tools are useful in the hands of a skilled administrator who has experience using them. The tools have withstood the test of time, but they do require considerable skill to wield effectively.

Other utilities are available that customize the use of the CLI tools. For example, Amanda adds an interface for the backup and restore procedures, which makes backup tasks simpler to manage. It includes a client and server component to facilitate a central backup solution for various remote clients, regardless of the platform. Amanda is typically included, or at least available, in most Linux distributions.

Another useful feature of Linux backups is evident in the capabilities of the file system. File systems, such as ZFS and BTRFS, can take snapshots. These mechanisms can aid the backup process by allowing the backup software to concern itself only with backing up the static snapshot while allowing new changes to the data to continue unimpeded. This process provides for greater efficiency of the backup process.

Several databases provide mechanisms to create backups, which ensure that memory buffers are flushed to disk and that a consistent data set is created. This feature can also be combined with storage facilities, such as FlashCopy, that perform instantaneous point-in-time copies.

Finally, commercial backup utilities, such as the IBM Spectrum Protect, are available for an enterprise environment. For more information about IBM Spectrum Protect, see this web page.

5.5.5 Migrating backed-up and archived data

When moving to a newer or modern environment, the archived data in the environment might no longer be supported, depending on the storage technology that is used. It becomes necessary to migrate archived data to a newer format. This process helps ensure compatibility with the production IT environment and maintains data availability.

Why migrate archived data

The following factors can force the migration of archived data, among others:

- Preserving data on the same medium would face two problems:
  - The lifetime of the medium.
  - The long-term availability of the technology for reading it.
- Eventually, technology changes and your solution becomes less competitive compared to emerging ones.

- Some older storage technologies directly affect the volume of data that can be stored and the space requirements because of the low MBytes/cm3 and Weight/MByte factors.
- End of support for your current solution.

5.5.6 General archival migration considerations

The following methods are available to migrate data from the existing operating environment to another operating environment:

- Change in the hardware environment
- Change in the hardware and software environment

Changes in the hardware environment

This scenario applies when the hardware (servers and storage devices) is replaced by newer and more efficient hardware environments.

Changes in the hardware environment at times can lead to a change of storage technology, which means reorganizing the media data content. Therefore, the data inventory structures might need to be reconverted to allow efficient data retrieval.

Because the operating system and the backup and archival management tools are going to be retained or upgraded, no incompatibility issues occur with the archived data. This fact also means that the migration is relatively straightforward because the storage backup and archival manager product can access the archived data.

Often, backup and archival managers include built-in migration tools that migrate the archived data from the source operating environment to the target environment. This period is a useful time at which to reorganize the archives and purge unwanted data so that you can efficiently reduce the storage needs of the archives.

Changes in the hardware and software environment

This scenario applies when the IT department decides to move to a new operating environment (hardware and software). In this case, the hardware and software technology must be replaced. The hardware can have a highly efficient virtualization server and the software can have new technologies that are proprietary or open source.

5.5.7 Migrating to new backup software

This section describes the migration approaches that you can use when the target environment's software stack is changed.

Software is incompatible with LinuxONE

In this approach, because your new guest is not compatible with the old software, all archived data must be restored to a staging server that is compatible with the old backup tool. Use the staging server to restore the archived data and share it with the new LinuxONE server that is connected to the new backup software.

The following example highlights this process:

1. From the archival software, restore the archived data to a staging server that is compatible with the old backup software.
2. Connect the new server running LinuxONE that is used for the current backups and archives to the staging server (for example, by using a shared file system) to access the restored data.
3. The new backup and archival software connects to LinuxONE, accesses the restored data, and rearchives it according to defined organizational attributes and backup policies.

Software is compatible with LinuxONE

In this approach, the archived data is restored from the old system to the new LinuxONE server. The exported archive data must be rearchived into the new archiving system. You can transfer all the data to the new backup software or transfer it on demand.

5.6 Security analysis

This section provides an overview of the security considerations that you must include in analyzing programs and functions that are going to be part of the migration. Available enterprise-wide authentication options and their possible role in migration are also described. Finally, because Secure Sockets Layer (SSL)/SSH likely is going to be used, the use of the cryptography hardware that is available is described. IBM LinuxONE also offers extensive capabilities for encrypting data with great performance.

A properly secured application, on a properly secured system, is a cornerstone of any computing environment that is used today. Thus, migrating an application from one server to another, regardless of the platforms that are involved, is a matter of validating that the same policies are available and in place in the new environment. This section covers the security options that can be implemented for an application that is migrated to LinuxONE, if none were already in place. It is entirely possible that these tasks are already implemented, and need validation as being intact after a migration.

This section discusses the following topics:

- Security migration overview
- Code and application analysis
- Availability and accountability
- Data integrity and confidentiality
- Security change management
- Enterprise authentication options
- Central Processor Assist for Cryptographic Functions

5.6.1 Security migration overview

Overall security is composed of three domains:

- Physical security
- System security
- Network security

In each domain, the concept of “principle of least privilege” is applied, which results in the security policy. That policy is where each individual is only granted the access that they need, no more. You need to establish individuals and their roles, and who is going to be allowed to do what. This process is vital for overall system security because if a compromise occurs, its exposure is to only the affected role.

Use mandatory access controls to not only help ensure that privileged access is given to only what is needed, but to also ensure that authorization is withdrawn when privileges are revoked.

A basic premise that underlies the concept of security is that you are only as strong as your weakest point. That is why security is time-consuming, and it is difficult to predict the amount of time that the analysis will take. If this is the first time that you are undertaking a security analysis, do not underestimate the time or scope involved in this task.

It is held that “security through obscurity” is not a valid method. Using open, well-established security methods implemented correctly provides the best defense. For example, instead of developing your own cryptographic libraries, instead use open, established ones that have been vetted for many years. Hiding information creates more system administration work, and any mistakes might cause a failure to protect against attacks.

System logs and application logs must be immutable. Logs must be kept in such a way that they cannot be altered by system users. If logs can be altered, overall system integrity comes into question if a hack is suspected. Therefore, it is important that all logs be kept in a way that makes them a permanent record of what occurred on the system.

Document the system security and all the assumptions made. Include all “what if” situations that can reasonably be expected to occur. Also, document security plans such as change control, audits, and procedures for break-ins in all domains.

Understanding the hypervisor foundation

The Linux VM is controlled by a hypervisor, perhaps KVM or z/VM. Thus, for a complete security survey to be done, you need both access and an understanding of its security.

The VM layer allows for many operating system images to run on the same hardware at the same time. The hypervisor allows for resources to be shared between each VM. It also allows for virtual devices to be created and consumed, like HiperSockets.

LinuxONE and existing security policies

Most organizations have an existing security policy dictating that certain hardware, especially those hosting virtual environments, must not be internet-facing. With the migration of a distributed environment to LinuxONE, this requirement often raises questions about the role of LinuxONE within the existing security policy. A useful approach regarding security policy is to conform with the existing policy as much as possible because it simplifies the migration process.

Processor Resource/System Manager (PR/SM) is certified through the Common Criteria at Evaluation Acceptance Level (EAL) 5+. More details about Common Criteria are covered in 1.3, “Reasons to choose IBM LinuxONE” on page 10.

To further help ensure the isolation of one partition from another, dedicate the OSAs used to connect to external networks by a hypervisor to the partition in question. These precautions help ensure that other guests or partitions cannot share an external-facing network. However, if the security policy states that nothing on a virtualized environment can be connected to the internet, you have the option of putting the web servers on x86 servers with a physical firewall between the web servers and the hypervisor.

Firewalls and existing security policies

Often, an organization's existing security policy identifies specific firewalls that have been approved for use on the corporate network. Most often these firewalls are hardware firewall appliances. Although the LinuxONE hypervisors can provide a virtual network between the virtual Linux servers, there is often a requirement to have a firewall between distributed servers, such as an application server talking to a database server. In a distributed environment, the firewall is in the communication path.

LinuxONE provides two options. The first is to implement a software firewall on a virtual server within the virtual Linux environment. This configuration has some challenges because the firewall software might not be used in the organization and as such must be certified, which might be a long and complicated process.

The second option is to continue to use the physical firewalls by having the inter-security level communication exit the virtual environment through an OSA, go through the physical firewall, and then return to the virtual environment through a different OSA. Figure 5-15 shows the use of an external firewall.

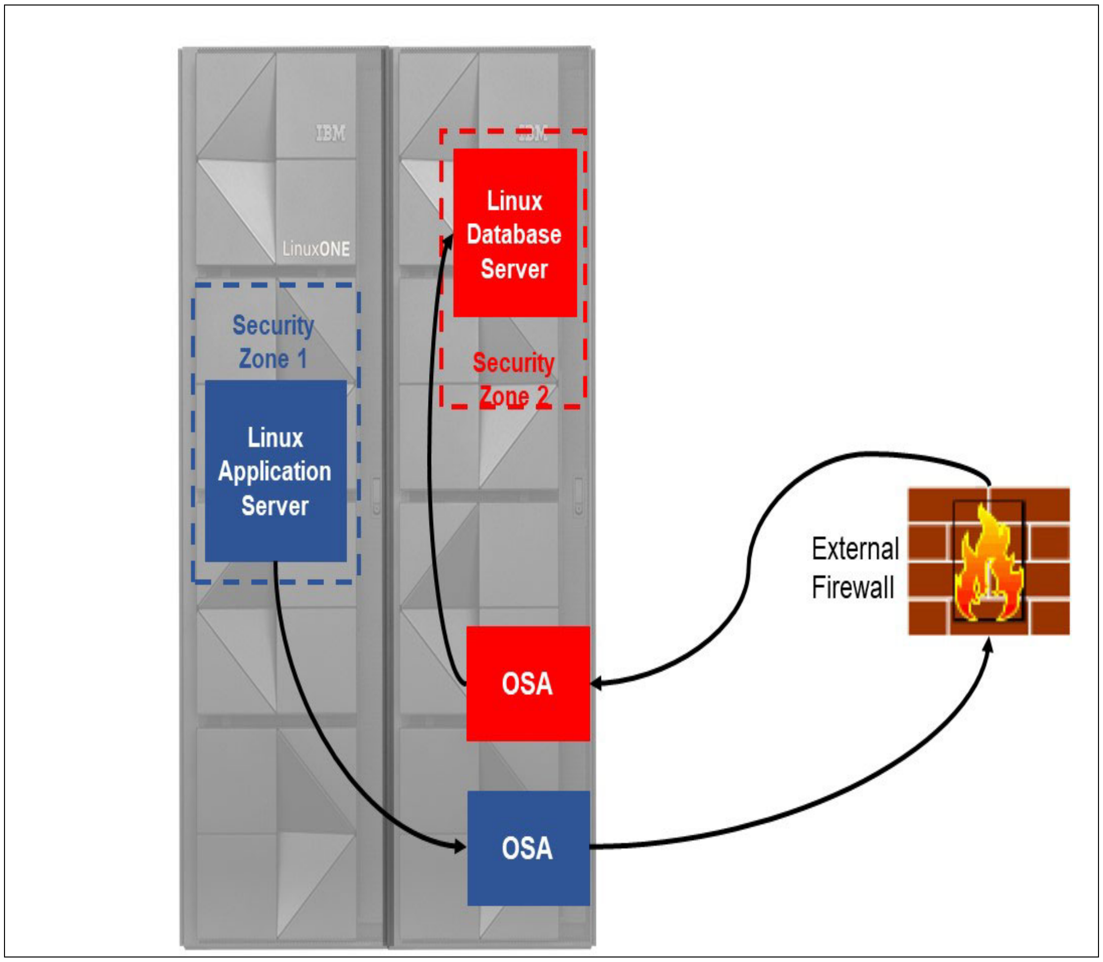


Figure 5-15 Using external firewalls between security zones

The different security zones that are shown can be in separate partitions or in the same partition. Customers reported that using external firewalls has minimal performance impact.

Conforming to the security policy can simplify a migration. However, the reality is that for applications within the LinuxONE footprint, there might be no requirement for firewalls if all incoming communications to LinuxONE are processed by external firewalls.

Control of the hypervisor

Who will own the hypervisor, and what is the protocol for requesting changes or actions? Regardless of whether the KVM or z/VM hypervisor is used, if you control the hypervisor, you need to fully understand it because it is the basis for all the VMs on that partition. It must be secure and its access should be highly controlled. Also, document a change request protocol and publish it to all stakeholders.

You also need to plan for hypervisor maintenance, which might require that some or all VMs be quiesced. Therefore, help ensure that a change window is set aside to allow for maintenance, and put a plan in place and set a schedule to allow for security and hypervisor updates and maintenance.

Security references

For more information about hosting Linux guests, and security and networks, see the following IBM publications:

- *Security on z/VM*, SG24-7471
- *Introduction to the New Mainframe: z/VM Basics*, SG24-7316
- *IBM Z Connectivity Handbook*, SG24-5444

Hardening the base Linux guest

The term *hardening* is commonly used in server security to mean the process of taking a generic operating system and changing it to provide only what is necessary for the production environment. This configuration provides a baseline for security for the operating system.

During migration, you might be given an already hardened Linux image. In that case, you know only what is allowed and not allowed with the image. However, if a hardened Linux image does not exist, create and maintain one.

Creating a hardened Linux guest

The basics of hardening a Linux on any platform consists of removing all unnecessary applications and services, and then securing the applications and services that remain. Explaining this process is beyond the scope of this book, but for more information, see *Security for Linux on System z*, SG24-7728.

Migrating to a hardened Linux guest

A hardened Linux should have most if not all applications and services removed or disabled. There might be more than one hardened Linux to choose from, so be sure to choose the version that provides the maximum number of applications and services that you need to perform the migration.

You need your migration analysis to determine what needs to be reenabled. If any applications are to be installed and services enabled, you need to provide credible business cases for each, individually or as a set. Completing the security analysis can provide such business cases. Make sure that the documentation includes all applications and services as a delta from the base hardened Linux image.

Important: Red Hat Enterprise Linux (RHEL) includes the SELinux security method, SUSE Linux Enterprise Server includes AppArmor for its enhanced security method, and Ubuntu uses AppArmor by default (although SELinux is available). Determine whether those environments are in use or required, and plan from there.

Those mechanisms are complex, so invest the time that you need to identify code and applications that have not been ported to work in these environments.

Maintaining a hardened Linux guest

It is necessary to maintain a base hardened Linux guest. Kernels change and security patches are issued, so you need to develop a plan for maintaining the base image and assigning the resources to accomplish it. Thus, successive migrations benefit from a properly maintained base hardened Linux.

5.6.2 Code and application analysis

When migrating an application from x86 to LinuxONE, the vendor might provide a supported package for LinuxONE. The application might be available only as source code (from a vendor, or something developed internally) that needs to be recompiled for the new LinuxONE platform. Either way, it is important to know what the current security methods are, and how they will be used in a VM on LinuxONE. Be sure to poll the stakeholders about the security requirements that must be met.

When moving an application to LinuxONE, consider isolating applications on different VMs. If possible, consider isolating application VMs from VMs that serve data. The more isolation, the more straightforward the security is.

If you know that there is a security issue with an application, do not use it. You need to address all security issues before the system is placed in production. If more secure ways to configure an application are available, invest the time to make those changes during migration. For example, you might place a database on a different VM than the application that uses it. The more separation, the more straightforward security is. Systems with security tend to be easier to defend and maintain.

Code dependencies

Almost all code uses APIs and other libraries to perform the tasks that it was designed for. Therefore, you need to review these dependencies *before* migrating. If you discover that a dependency exists on an item that has a known security issue, you must find and implement a suitable replacement.

Application dependencies

Generate and review a list of all application dependencies for known security issues. Only fixed or known secure versions should be used. You might be tempted to migrate the application over to the new Linux guest and test to prove that the migration is achievable. However, such testing is invalid if any application or its dependency is on code that has known security issues.

Checking user input

User input is the vector that is most commonly used to attack systems and programs, so all user interaction must be examined carefully. Check all input to make sure that it is within the range of the data that is needed to be processed. Never pass raw input to another application or system request.

Use exceptions to help ensure that input always conforms to the format that is expected, and, if the unexpected occurs, that it can be gracefully handled.

Planning for updates when migrating code

When code is migrated to an enterprise-class system, changes need to be addressed in a different manner. Unlike less critical code, changes must be allowed to be run while the application is still running. Thus, you must help ensure that a method is in place to signal that configuration and control files are updated and need to be reloaded.

A security issue might need to be addressed by configuration changes. In an enterprise environment, a program should not be stopped but only signaled to take on changes (for example, you might need to change the TCP port that an application uses). Help ensure that the code can handle such changes gracefully.

Carefully examine all configuration changes. Do not assume that the changes are valid, but instead verify that they are within the bounds of the setting. If they are not, handle the error gracefully.

Networking

If the code implements TCP sockets, make sure that its design and function are reviewed with the networking team that represents the firewall. That team probably needs to know the following information:

- What ports are used by the code, and for what purpose?
- What type of protocol is used: TCP, UDP, ICMP, and so on?
- Are special settings used on the port, as in TCP keepalive?
- How long can a connection tolerate a lack of response?
- How long is a connection allowed to idle?

Logging and recording events

All logs must be kept in a way so that they cannot be changed. They need to be a permanent record of what occurred on the system. Configure the Linux so that syslog (the Linux system log) not only keeps a local record, but also forwards it to a remote secure system. Also, make sure that all critical applications are properly configured to use syslog.

Implementing syslog logging when migrating code

The syslog-ng daemon runs on Linux. Take time to update the code as needed to send messages to this daemon. At the least, log all information that deals with security and critical state information. The benefit of implementing syslog functions is that log maintenance is performed by the system (log rotation and archiving).

Escalations of authority

Apply the “principle of least privilege”, which means that programs operate only with the authority that is needed to accomplish a goal. If the code accesses a database, it should access it only as a user with the access needed, and not as an administrator.

Migrating code

Analyze your code to determine any escalations of authority. Also, help ensure that it accounts for exceptions so that a de-escalation of authority exists. In other words, make sure that if the code is broken, it does not allow the user to operate at a different access level than is allowed.

Migrating applications

Programs that run as `root`, the super user, must be carefully examined and assurances given that they are operating as designed. Generally, do allow any code or program to run with such authority, if you can avoid it. Make sure that server applications are run at the suggested secure settings during all phases of the migration. You do not want to run applications as the administrator during development, only to discover during testing that certain functions do not work.

Security test plan and peer review

All code and applications that are to be migrated should be in their secure mode during development straight through to test and deployment. You need to validate the security assumptions made. This validation determines what you need to include in the security test plan.

Test everything that can be tested and document what was not tested and why. It is also worthwhile to test change control and verify the restore of backups. If an incident does occur, the only way to recover might be to patch the fault and restore data from the backups (assuming that they are not compromised).

5.6.3 Availability and accountability

Security involves more than who can access a system. It also involves keeping the system available to authorized users and not available to unauthorized users. Denial-of-service (DoS) attacks have become more frequent in recent years, and internet-facing systems must take the possibility of such threats into account.

Implementing executable system security requires an audit trail, without exceptions. All access to the system must be logged in a secure fashion to help ensure that if an authorized user commits an indiscretion, that it cannot be covered up.

Availability analysis

Sometimes attackers do not break into a system, but instead bring down a service by overwhelming it with requests. Thus, system or services availability needs to be understood and service-level agreements (SLAs) maintained.

Internet-facing Linux considerations

The internet is a public “space” where individuals are anonymous. Therefore, every effort must be made to mitigate malicious access if you have an internet-facing Linux system. Be able to identify individuals and their IP addresses so that, if necessary, you can work with the networking team to prevent malicious access while still allowing authorized users to have access.

Communicating availability

Establish a standard for communicating system availability that explains how to report issues and outages to help ensure that they are communicated to the appropriate staff. An unexpected interruption in availability can be the first sign that there is a security issue and that a potential security threat needs to be addressed.

Accountability analysis

All system logs and application logs must be immutable. If attackers gain access, they generally erase evidence of their presence to avoid detection. Also, if users attempt to perform unauthorized acts, they might try to cover their activities by erasing log files or incriminating evidence.

Making log files immutable

Configure syslog-ng to store logs on a separate secure server. Optimally, the logs should be stored in a Write Once Read Many (WORM) device. Do not delete logs, and keep a secure backup.

Another approach to securing system logs is to use a remote log server, as supported by syslog-ng. See an example of this approach in 8.6, “Deploying a central log server” on page 227. The logs on the remote log server are not necessarily immutable, but they are not directly writable from a system that is compromised.

Audit trails encompassing all security domains

Make sure that security audits can always be passed by verifying that you can trace an individual’s physical, network, and application access to systems across domains. Be able to show a system access audit trail from all domains, not just from system access.

Authentication

Help ensure that communication end-points are who they say they are. Attackers often “spoof” or pretend to be a system or user that they are not. To protect against such attacks, use “authentication” conversations:

- Users must be assured that they are connecting to the server that they think they are.
- Servers need to be assured that users are who they say they are.
- This authentication must be kept private so that eavesdropping cannot occur.

Disabling Telnet access and using Secure Shell (SSH) accomplishes this authentication. Using SSL with web servers also accomplishes this level of security, and is preferred over the default of no SSL.

5.6.4 Data integrity and confidentiality

A benefit of migrating to LinuxONE is that data can be stored on an enterprise-class system. However, you need to analyze the current state of the data and then determine how it fits in the new enterprise system.

Data integrity analysis

Data integrity refers to the assurance that data is unchanged from creation to reception. Data integrity also entails understanding the following items:

- Who can access what data and what is allowed
- Is there an audit trail in place to map who changed what and when
- Whether the data is corrupted in some way and how is it to be restored
- Whether a DPM plan is in place

Protecting data at rest from unauthorized access

Protecting access to a database is understood, but what about protecting raw data on the disk itself? Mobile computers with databases full of accounts or data are sometimes misplaced or stolen. Thus, you need to protect data “at rest” (meaning the files themselves) and help ensure that the data is kept in a secure way. Prevent offline copies of a database from being kept on portable devices or drives. Control of data is key. Be sure to communicate the data integrity policy to all individuals who have access, and monitor the audit trail to make sure that the policy is being enforced.

Data backups: Part of security

Part of your security plan needs to include backups and how they are stored. They need to be kept in a secure way. When backups are kept separate from the system for DPM purposes, use encryption to prevent unauthorized access. Understand the impact if the backups are stolen and mitigate the risk.

Confidentiality analysis

Confidentiality must first be communicated and then enforced. Thus, before users can access a system, they need to be told what the confidentiality of a system is and how any data or information is used or shared. In addition, a system needs to be in place to enforce the policy. This enforcement is normally done by auditing access logs. If a violation is detected, it needs to be communicated to the affected parties.

Understanding laws and regulations before an incident occurs

Before you can create a confidentiality policy, you need to understand what is legally expected:

- Are there national, regional, or state laws that need to be followed?
- Are there any industry compliance requirements (such as Payment Card Industry requirements) regarding the storage of credit card information?
- Is there a company policy? If so, it needs to be followed.
- Document all expectations regarding how long to keep the data (for example, “We expect or are required to keep the data for up to 5 years”).

Publishing your confidentiality policy

You need to communicate the confidentiality policy in such a way as to help ensure that all users of the system are aware of it and thus can be held accountable. When a user logs in to a system, use the Message of the Day (MOTD) found in `/etc/motd` as shown in Example 5-3 to communicate with your system users.

Example 5-3 Using /etc/motd to communicate system policy

```
****************************************************
*      .--.     Welcome to the Linux s/390x VM     *
*     |o_o |    SUSE Linux Enterprise Server 15 SP2*
*     |:_/ |    System Admin: John Doe             *
*    //   \ \                 jdoe@company.com     *
*   (|     | )  This system governed by corporate  *
*  /'\_   _/`\  Policy K49-r v21 please read       *
*  \___)=(___/  before accessing system            *
****************************************************
```

Tip: Use ANSI art or special characters to make the login window attractive. It is useful to display system information such as the Linux distribution with its version and release information, along with a greeting.

On web pages, create a link from the main page so that the system policy can be easily accessed. If you are allowing VNC login, display the policy by updating `/etc/gdm/custom.conf` as shown in Example 5-4 on page 125.

Example 5-4 Policy found in /etc/gdm/custom.conf

```
[greeter]
DefaultRemoteWelcome=false
RemoteWelcome=Connected to %n must read policy K49-R v21
```

Having a plan in place before an incident occurs

Have a plan in place in case confidentiality is violated. The plan should include these items:

- Who should be notified and what should be disclosed about the incident.
- If there is a requirement to notify the public, document how and what should be disclosed.

Communicate actions that will be taken to prevent future incidents.

5.6.5 Security change management

No system is perfect so there will be changes, however infrequent. Because security fixes are important to keep current, you need a plan to understand their impact on the system. If a Linux needs to be restarted, it must be done in an orderly and timely basis.

After the system is moved from test to production mode, it remains that way. Outages are expensive for companies, but failing to plan change windows and downtime also causes security problems. In the rare case that a VM needs to be restarted, you need the ability to allow for these types of changes.

Testing changes with a clone of the Linux guest

The advantage of migrating to LinuxONE is that you can clone a VM and test changes before you apply them to the production images. Run through the complete change from start to finish, rather than assuming that it works.

Record how long it takes to make changes and test worst-case scenarios. After testing the change on the clone is complete, report to production stakeholders how long the change will take and how long the worst case will take.

5.6.6 Enterprise authentication options

Migrating to an enterprise system means that user and identification management can be consolidated. This section describes enterprise authentication options and where to find the corresponding information about how to implement them.

A common centralized LDAP server

When migrating applications and code to LinuxONE, you can simplify user administration by storing user information in an LDAP server. Configuring the Linux guest to authenticate from a centralized LDAP server provides the following benefits:

- User management is simplified, and users can be managed across the enterprise.
- Changes made to a user are applied across all images.
- An offline VM might contain outdated user information. Using LDAP helps ensure that bringing an old image online does not compromise current security.

You can also configure Samba to use LDAP as its user repository. Thus, you can have one security domain across MS Windows, IBM AIX®, and Linux. For more information about this topic, see *Open Your Windows with Samba on Linux*, REDP-3780.

5.6.7 Central Processor Assist for Cryptographic Functions

When migrating to LinuxONE, the underlying hardware can accelerate cryptographic mathematics. The Central Processor Assist for Cryptographic Functions (CPACF) supports synchronous cryptographic functions. The work is processed by the crypto-assist processor that is integrated into every processor in IBM LinuxONE, or the Crypto-Express card, if it is installed.

The following APIs are supported:

- OpenCryptoki
- OpenSSL
- Global Security Kit

OpenCryptoki

An open source implementation of Public-Key Cryptography Standard #11 (PKCS#11), OpenCryptoki uses the libica shared library to access IBM cryptographic adapters through the z90crypt device driver.

OpenSSL

An open source implementation of SSL, OpenSSL can use the libica shared library for hardware encryption.

Global Security Kit

Provided as part of the IBM HTTP Server, Global Security Kit (GSKit) manages SSL certificates. It uses OpenCryptoki for hardware encryption.

Using this approach offloads the cycles and allows for more concurrent access to a web server that is using SSL or applications that use one of the supported APIs. To learn about how to configure your system so that your Linux guest takes advantage of the installed hardware, see *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, SG24-8147.

5.6.8 Data encryption

The IT landscape of any enterprise contains business assets that can cause great damage if lost or compromised. Regulations are in place for several industries, requiring data to be protected and guarded by the company. Changing laws in countries throughout the world are increasing the control necessary over data protection.

Data is at the core of every business decision, becoming increasingly complex with time. This value must be protected with a strong perimeter by using encryption. IBM LinuxONE is capable of strong encryption with efficiencies, enabling enterprises to benefit from IBM Pervasive Encryption for data at-rest, along with network and application cryptographic accelerations.

IBM Pervasive Encryption is a capability that offers extensive encryption for data that is in-flight and at-rest to substantially simplify encryption while reducing costs that are associated with protecting data and achieving compliance mandates.

The LinuxONE platform is designed to provide pervasive encryption capabilities to help protect data efficiently in the digital enterprise.

LinuxONE hardware capabilities

The IBM LinuxONE platform provides the hardware infrastructure in a balanced system design with the encryption capabilities that now make it possible to create a fortified perimeter around critical business data. The CPACF includes the cryptographic suite and performance characteristics that can enable bulk encryption of sensitive business data that makes it possible to fortify and intrinsically protect business data by using encryption technology.

Working with the new Crypto Express feature, the key materials that are used to create this fortified data perimeter are protected by using the unique protected key CPACF. The keys that are used in the encryption process are not visible to the applications and operating system in clear text form.

Why pervasive encryption

Organizations are recognizing that protecting only the data that is required to achieve compliance is the bare minimum. It is not uncommon to encounter regulations that were written some time ago that do not implement today's security best practices.

Organizations also realize that a move from selective encryption (protecting specific types of data only) to pervasive encryption (encrypting all data) is needed. Likewise, many barriers that are encountered today with current enterprise data protection policy and strategy can be removed with pervasive encryption, such as the following examples:

- Decoupling encryption from data classification

  This process allows organizations to implement their encryption strategy independent of any challenges they might face while identifying and classifying sensitive data. It also reduces the risk of unidentified or mis-classified data.

- Using encryption without interrupting business applications or affecting SLAs

  Changes to the application are not required if data is encrypted after it leaves the application and decrypted before it reaches the application.

- Reducing the high costs that are associated with processor overhead

  The cost of encryption is minimized by encrypting data in bulk and by using hardware encryption accelerators with high performance and low latency.

Pervasive encryption for LinuxONE is a combination of granularity versus complexity and overhead. The bigger the switch, the less freedom that is available to pick and choose different encryption algorithms, keys, and so on.

Similarly, the bigger the switch, the less overhead that is available for analyzing, classifying, encrypting, and maintaining the encryption environment. Ultimately, you gain a broader scope of encryption at the cost of granularity the further down the software or hardware stack you go.

LinuxONE cryptographic components

The following components for LinuxONE data at-rest encryption are in the Linux kernel:

- `dm-crypt`: A device mapper for a crypto-target (such as a storage device). It provides transparent encryption of volumes or block devices.
- `paes`: A module that performs protected key encryption-decryption and implements paes cipher. It also provides the cryptographic AES algorithm that is used by the infrastructure for protecting data at-rest (such as volume encryption).

- `pkey`: A module that is used for secure key and protected key management, including the following tasks:
  - Generates a secure key.
  - Transforms a secure key into a protected key.

  `pkey` supports a protected-key with AES as an encryption cipher for use with `dm-crypt`.

The following components for data at-rest encryption are in the Linux user space:

- `cryptsetup`: A utility that is used to create secure keys and to manage disk encryption. (It interfaces with `dm-crypt` and `zkey`.)

  By using `cryptsetup` commands, you can perform the following actions:
  - Open: Creates a mapping device.
  - Close: Removes the mapping device.
  - Status: Reports the mapping device status.
  - Resize: Resize an active mapping device.

  Format: Formats LUKS partitions.
- `zkey`: A utility that manages secure keys. You can create secure keys with a length of 128, 1926, or 256 bits.

How LinuxONE data at-rest encryption works

When the `paes` cipher is used with IBM LinuxONE data at-rest encryption, the following protected volume options are available:

- The LUKS2 format includes a header on the volume and a one-time formatting is required. The LUKS2 header is made up of multiple key slots. Each key slot contains key and cipher information.

  The volume's secure key is wrapped by a key-encrypting key (which is derived from a passphrase or a keyfile7) and stored in a keyslot. The user must supply the correct passphrase to unlock the keyslot. A keyfile allows for the automatic unlocking of the keyslot.
- The plain format does not include a header on the volume and no formatting of the volume is required. However, the key must be stored in a file in the file system. The key and cipher information must be supplied with every volume open.

For more information about setting up encryption for data at-rest by using IBM LinuxONE cryptographic capabilities, see *Getting Started with Linux on Z Encryption for Data At-Rest*, SG24-8436.

5.6.9 Secure boot

IBM LinuxONE platform hardware can boot in secure boot mode. Secure boot is a security feature many platforms requires to protect systems from the moment they are started. This protection is based on checking signatures for bootloader and kernel files that are chain-loaded during boot time.

The secure boot feature is part of the Unified Extensible Firmware Interface (UEFI), which is a central interface to the firmware, operating system, and individual components of the server. It protects from root-level attacks and malware that target boot-time vulnerabilities. The system checks images at boot time for a vendor-signed cryptographic key to verify that the image is from an official provider, and that the image was not tampered with or replaced by malicious third parties.

The system firmware first confirms that the system boot loader is signed with a verified cryptographic key. The system then confirms that the key was authorized by a database that is contained in the firmware and only recognized keys allow the system to boot.

For more information about Linux Secure Boot under LinuxONE, see *Maximizing security with LinuxONE*, REDP-5535.

5.7 Operational analysis

The source application comes with a complete support structure. Part of that support structure performs daily operational tasks. Depending upon the application, this support can be 24 hours a day, 7 days a week, and 365 days a year. The application relies on manual and automated intervention to start, stop, monitor, and maintain the services that are provided by the application.

This section describes some of the operational issues which, if present in the source application, must be addressed in the target application. A careful and detailed analysis about how the source application is supported by operations staff is required for a successful migration effort.

An analysis of the operational functions can highlight characteristics of the application that were not clear from the analysis of other application interfaces or from the code itself. The application code might be successfully ported, but it is as important that the application's operational support structures be migrated successfully as well.

5.7.1 Operational migration tasks

This section describes operational issues that might change when migrating the source application to the target application in a new environment:

- Starting and stopping the application

  These processes can be automated or manual. The source application probably had certain methods for starting and stopping its processes, but the target application will probably have different commands and methods for starting and stopping the application.

  If the target application is a manual process, the operators must be trained and the appropriate documentation must be written and published. If it is an automated process, the automation scripts need to be written, tested, documented, and explained to the operators.

- Notification of problems

  Sometimes automated messages or indicators can be sent that are unfamiliar, and the correct response is unknown. Operators and systems support staff need to know who to turn to for guidance when this type of problem arises, so the application owner needs to be clearly identified. If the application owner is not available or unresponsive, escalation procedures need to be in place. These details might change when the application is migrated to the target system.

- Normal intervention and monitoring

  Some applications need to be checked or modified during their lifecycle throughout the day. Often this process involves monitoring indicators or displays that show the health of the application. New procedures for the migrated target application must be communicated to the operators. Hands-on training sessions are optimal for operators so that they can learn by observation and perform required tasks.

- Hardware manipulation

  Some migrations include hardware consolidation or replacement. Operators need to be trained on how to operate and manipulate the new hardware. Even if the operators are not required to manipulate the hardware, it is still useful to tell them what is running on the new server and to have the appropriate documentation, labels, and signs available for reference.

- Hardware intervention and problem escalation

  There are fewer hardware interventions for operators to deal with on LinuxONE.

  For example, with the source application and server, an operator might be comfortable with an event that is required to restart a server by using the power switch. However, on LinuxONE, it is a serious error to use a power switch to react to a server or application problem.

  If a new hardware vendor is involved in the migration project, review the method that the operators must use to notify the vendor of an actionable message or event that needs to be communicated to the operators. Perform and then document a test of that procedure. Do not wait for a critical situation to occur before learning how to contact vendors or other support personnel. The contact information should include day shift, off hours, and weekend names and numbers. The requirements for the vendor contact needs to be clear. The vendor often requires precise, detailed information such as serial numbers, machine type, and location.

- Batch procedures and scheduling

  Most applications have batch processes that support the application. Automatic scheduling software is common at most installations to schedule and track those batch processes. Schedulers within the operations department need to be involved to create the necessary scheduling changes for the migrated application. The new schedules must then be communicated to the operators on each shift.

- Other considerations

  Not everything in your operating environment can be envisioned and described here. The intent of this chapter is to give you an idea of possible operational issues that are related to the migration project. Think of everything in your operating environment that might change or be affected by the migration of the source application to the target application. Then, create a plan to perform the required operational migration tasks. Finally, run your plan.

5.8 Disaster recovery and availability analysis

IT system outages can significantly impact businesses by rendering critical systems unavailable. The key to helping ensure that this problem does not occur is to analyze your systems and determine a hierarchy of availability needs. Keep in mind that not everything needs a remote hot site.

For better understanding, the following terms and definitions are used when discussing DPM, HA, and related concepts:

- Disaster recovery (DR)

  Planning for and using redundant hardware, software, networks, facilities, and so on, to recover the IT systems of a data center or the major components of an IT facility if they become unavailable for some reason.

- High availability (HA)

  Provide service during defined periods, at acceptable or agreed upon levels, and mask unplanned outages from users. HA employs fault tolerance, automated failure detection, recovery, bypass reconsideration, testing, problem, and change management.

- Continuous operation (CO)

  Continuously operate and mask planned outages from users. CO employs nondisruptive hardware and software changes, nondisruptive configuration changes, and software coexistence.

- Continuous availability (CA)

  Deliver nondisruptive service to users 7 days a week, 24 hours a day. With CA, there are no planned or unplanned outages.

The goal for mission-critical systems should be CA. Otherwise, the systems should not be defined as mission-critical.

5.8.1 Availability analysis

Migrating an application to a virtualized Linux environment on IBM LinuxONE offers an opportunity to implement an availability profile in line with the impact of the unavailability that the application has on the organization's overall business. However, this analysis is not always straightforward. For example, test and development workloads are not considered to be mission-critical. However, because they might be needed to correct an error in a production system, consider providing for some sort of test and development environment in your DR planning.

The challenge with DR is to achieve a balance between the impact of an unavailable system on the health of the business versus the cost of creating a resilient environment for the application. This planning should include the likely scenarios that might impact an application's availability, and unrelated events that might impact the ability of a business to function.

The usual IT issues such as server failure, network failure, power outage, disk failure, application failure, and operator error, can be planned for through duplication of resources and sites. Unrelated factors are rare and not directly related to IT, but they can have a huge impact on the ability of a business to function. These events include fire, natural disasters such as earthquake, severe weather, and flood, and civil disturbances. These events can have a major impact on the ability of people to work.

Although this chapter focuses on the IT-related issues, you also create a plan to deal with the other, non-IT related events.

5.8.2 Single points of failure

In determining the DR requirements of an application, you need to look at the probability of failure of a component and the cost to eliminate a single point of failure (SPOF).

Table 5-3 lists the components of an IBM LinuxONE virtualized environment running an application as a Linux guest and the relative costs of rectifying a SPOF.

Table 5-3 Potential single points of failure that can impact availability

SPOF	Probability of failure	Cost to rectify
LinuxONE hardware	Very low	High
LinuxONE LPAR	Very low	Low
LinuxONE hypervisor	Low	Low
Linux	Low	Very low
Disk system microcode	Low	Medium
Virtual network	Very low	Low
Physical network	Medium	Medium
Application	High	Very low

Apart from hardware and software failures, the following types of planned outages can impact an application's availability:

- Hardware upgrades that require a power-on reset
- Configuration changes that require a restart of the partition
- KVM or z/VM maintenance
- Linux kernel maintenance that requires a restart
- Application maintenance

5.8.3 LinuxONE features for high availability

IBM LinuxONE was designed around providing HA. Perhaps the most design effort has gone in to the transparent recovery of processor errors. During a hard processor error at an individual core level, the task is moved to a spare processor where processing continues transparently to the application. In IBM LinuxONE, several availability features were introduced to reduce the number of planned system outages. For example, the following actions are now fully concurrent and require no system outage:

- Adding LPARs
- Adding logical processors to a partition
- Adding logical channel subsystems (LCSSs) - I/O paths
- Adding subchannel sets
- Enabling dynamic I/O
- Adding a cryptographic processor to an LPAR

Also, many services enhancements were introduced to avoid planned outages:

- Concurrent firmware fixes
- Concurrent driver upgrades
- Concurrent parts replacement
- Concurrent hardware upgrades

The IBM LinuxONE offers several customer-initiated capacity on demand features. These billable features are designed to provide customers with additional capacity to handle the following events:

- A Customer Initiated Upgrade (CIU) is used for a permanent capacity upgrade.
- Capacity BackUp (CBU) is a predefined capacity for DR. A system at a DR site does not need to have the same capacity as the primary site. During a declared disaster, or for up to 5 DR tests, the customer can turn on the number of processors, including Integrated Facility for Linux (IFL) processors, required to handle the workload from the primary site.
- Capacity for Planned Event (CPE) is used to replace capacity that is lost within the enterprise due to a planned event such as a facility upgrade or system relocation.

On/Off Capacity on Demand provides extra capacity in 2-hour increments that is available to be turned on to satisfy peak demand in workloads.

5.8.4 Availability scenarios

The following scenarios present several different situations where a LinuxONE environment is set up with increasing degrees of availability and increasing levels of cost. The key to maximum availability is to eliminate single points of failure.

All scenarios assume that the IBM LinuxONE is configured with redundant LPARs, redundant paths to disk (FICON and FCP), redundant Open System Adapters connected to the organization’s network, redundant system consoles, and redundant HMCs. This setup is the normal setup for an IBM LinuxONE system.

The application design needs to include redundant software servers. The storage infrastructure should also include redundant Fibre Channel switches, mirrored disks, and data.

Design the communications network around redundancy with redundant network routers, switches, hubs, and wireless access points.

For mission-critical systems, provide an uninterrupted power supply and a second site far enough away from the primary site to avoid being affected by natural disasters.

Another important factor in the availability of applications is security and access controls. For more information, see 5.6, “Security analysis” on page 116.

Single LinuxONE LPAR: Clustered WebSphere Application Server

Figure 5-16 shows a LinuxONE partition that shares system resources to all Linux VMs in that partition. The WebSphere Application Servers are in a two-node cluster. In the unlikely event that a processor fails, IBM LinuxONE automatically switches the workloads to a spare or any unassigned processor without any disruption to the active task.

If a Linux VM that runs the WebSphere Application Server workload fails, the other node in the cluster takes over if you are running WebSphere Application Server Network Deployment. This failover is achieved because an application that is deployed to a cluster runs on all members concurrently. More availability is provided through the nondisruptive addition of new VMs to the cluster.

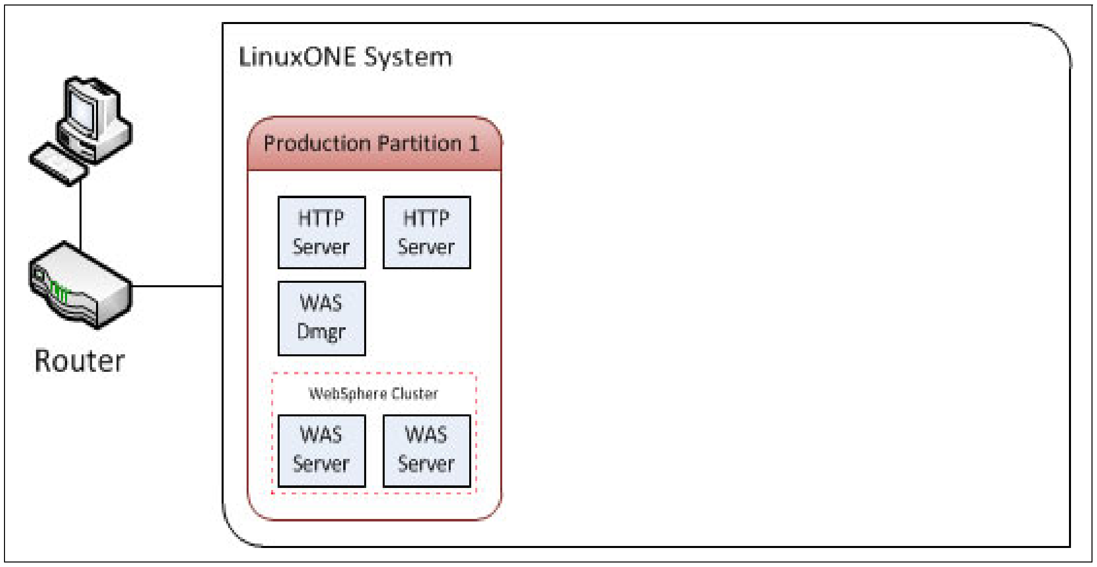


Figure 5-16 LinuxONE with a single partition running WebSphere Application Server cluster

This environment also provides extra availability through redundant HTTP servers.

Multiple LPARs: HA solution for Linux

Figure 5-17 shows a scenario with three partitions defined. Each partition can have one or more cores that are shared among partitions.

In this case, the production workload and WebSphere Application Server cluster are split across two LPARs, which give HA to WebSphere Application Server because a partition or hypervisor failure will not impact the availability of WebSphere Application Server.

Development and test workloads run in their own LPAR, so any errant servers have no impact on the production workloads. As in the first scenario, a failure of a LinuxONE processor is fixed automatically without any impact to the running application.

This configuration eliminates most failure points at a reasonable cost.

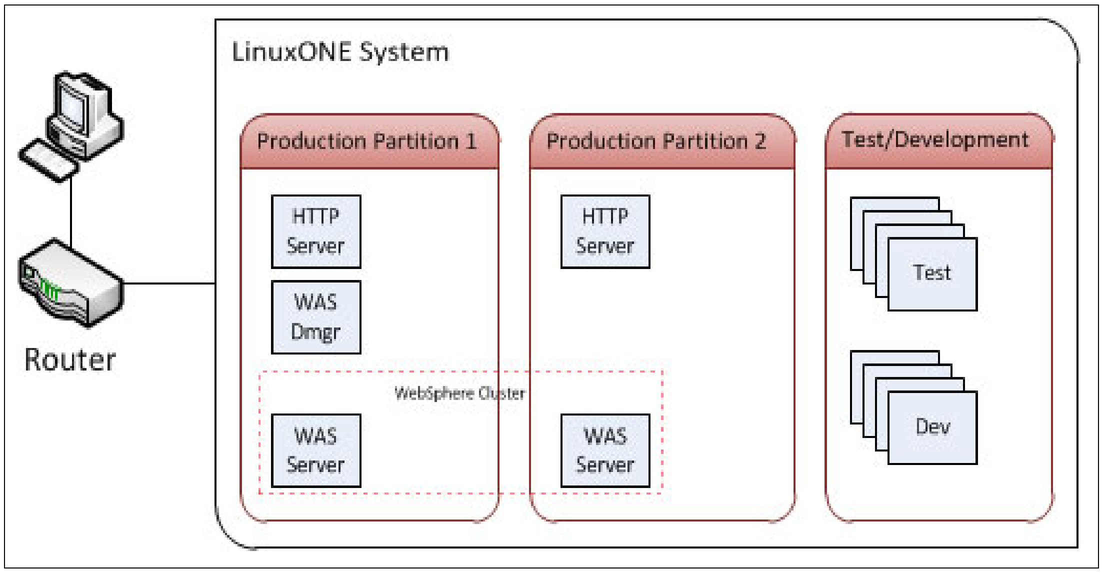


Figure 5-17 LinuxONE with multiple partitions running WebSphere Application Server cluster

Active/passive standby cluster with Pacemaker and Heartbeat

Beyond specialized facilities such as the WebSphere Application Server cluster, the Linux infrastructure has more generalized and robust clustering solutions available. A solution can be established with an open source product such as Pacemaker, or a commercial product such as Tivoli® System Automation for Multiplatforms (SA MP). In this scenario, the VMs are in an active/passive relationship. In this relationship, software runs on each server that enables monitoring and communication between each server. The software coordinates an automated failover of traffic to the standby server during a failure of the primary system.

Figure 5-18 shows this configuration, where a single IP address (the "service IP") acts as an alias to the network adapter of the primary partition.

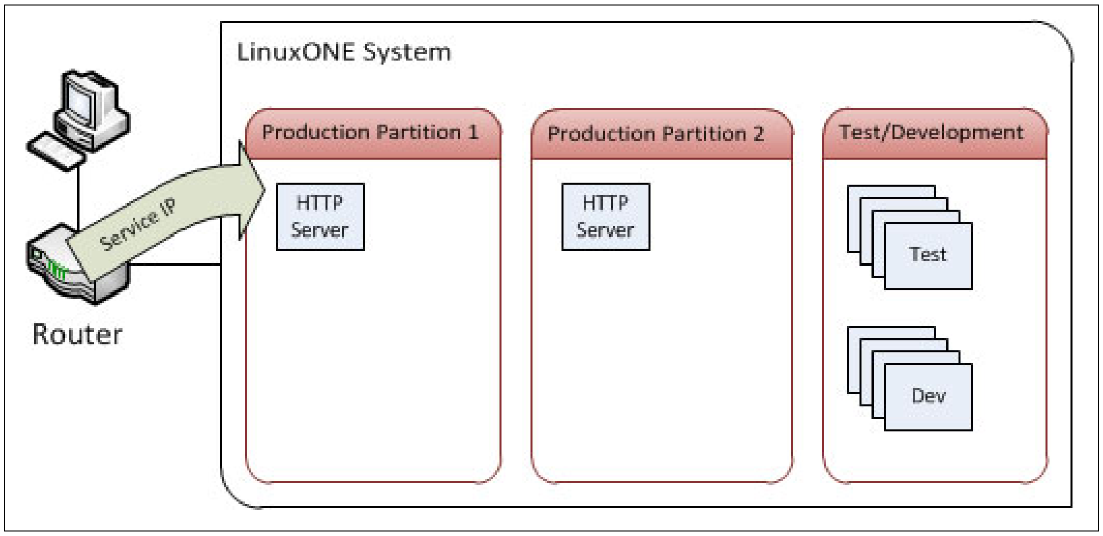


Figure 5-18 Active/passive cluster during normal operation

At regular intervals, the clustering software verifies that the physical partitions, the VMs, and the server applications (a web server in this example) are all responsive. If any component is not, then the service IP points to the network of the passive system, as shown in Figure 5-19. The cluster can be configured as to what action is performed, from notification to resource reboots of the failing system.

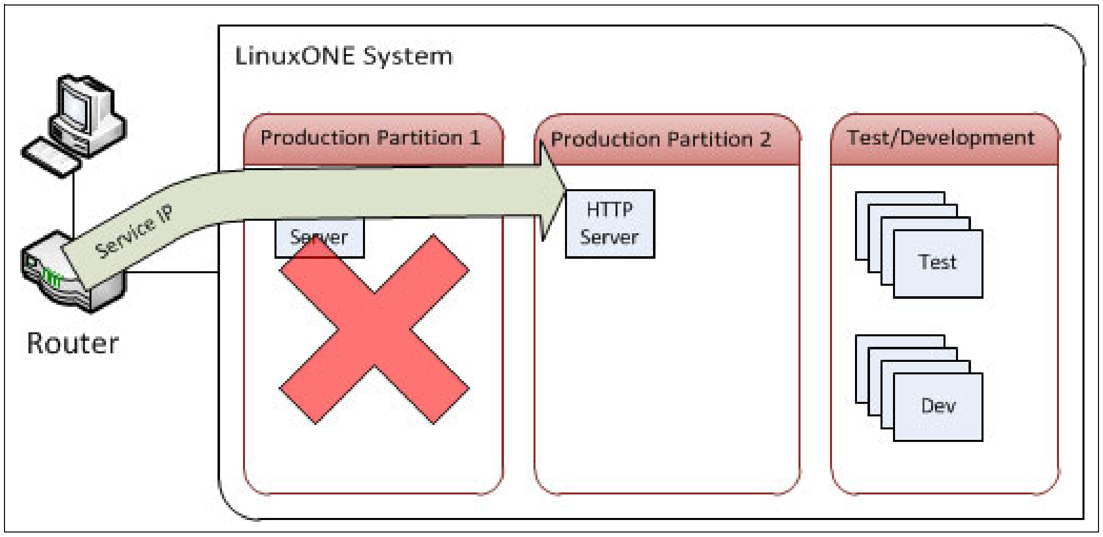


Figure 5-19 Active/passive cluster after an outage occurs

For more information about Tivoli System Automation for Multiplatforms, see this web page.

For more information about Pacemaker, see this website.

Active/active application server cluster

Figure 5-20 shows a WebSphere Application Server setup in an active/active configuration where the WebSphere Application Server Cluster spans two Linux VMs in two partitions. This setup handles the rare occurrence of the failure of a partition. More importantly, it also allows hypervisor maintenance to be performed without an outage to the WebSphere applications. This task is scheduled for a time when the processing load is light. Guest relocation or migration can also be used to avoid an outage because of hypervisor maintenance.

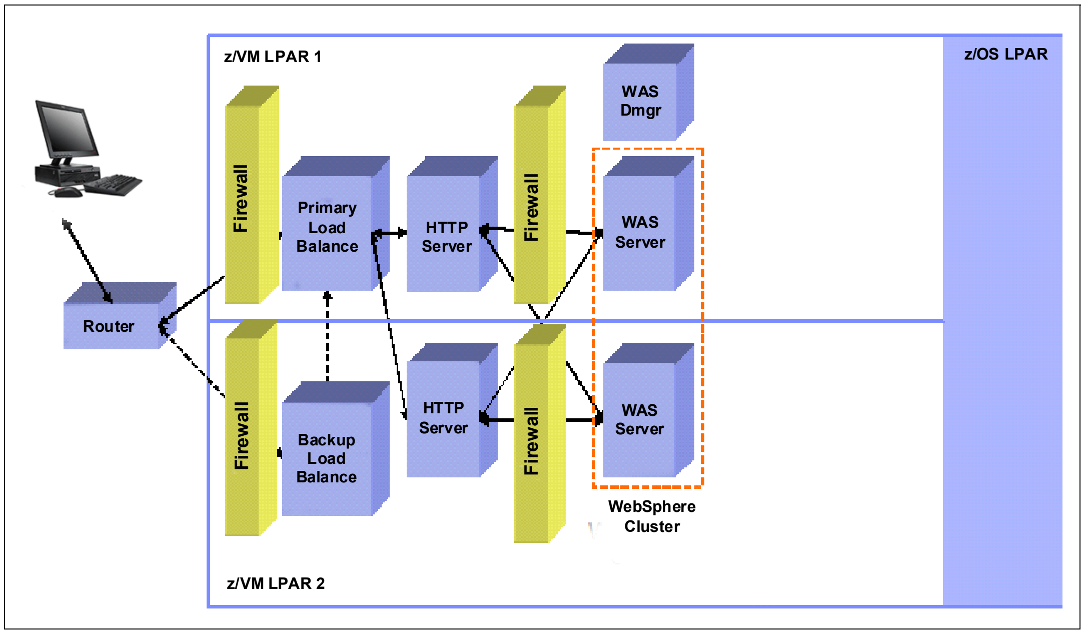


Figure 5-20 Active/active WebSphere Application Server cluster

Active/active application server cluster with database replication

Figure 5-21 shows a Db2 database that was added to an active/active WebSphere cluster. To provide HA for the Db2 database, the Db2 data replication feature, high availability and disaster recovery (HADR) is used. HADR protects against data failure by replication changes from the source (called *primary*) database to a target (called *standby*) database.

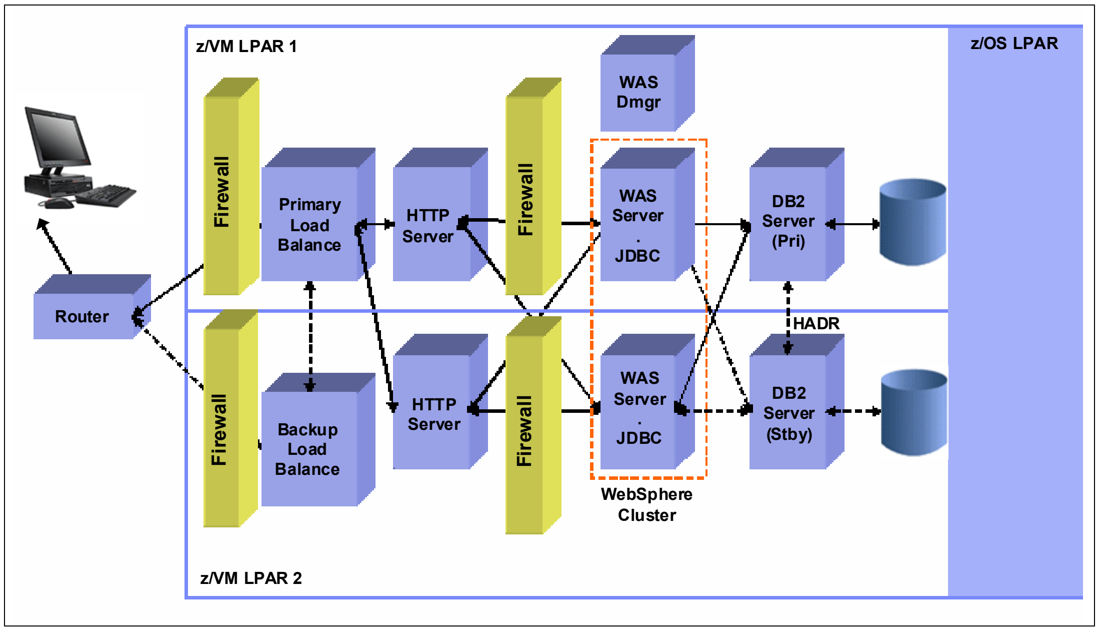


Figure 5-21 Active/active WebSphere Application Server cluster and Db2 HADR

During a partition-wide outage of the primary Db2 system, the standby Db2 system takes over in seconds, which provides HA. Communication between the primary and standby systems is through TCP/IP, which in this case is done by using the high-speed virtual network feature HiperSockets (available on LinuxONE).

The standby Db2 system can also be at a remote site to provide enhanced availability during a site failure.

IBM Tivoli SA MP running in both Db2 servers is designed to automatically detect a failure of the primary, and then issue commands on the standby for its Db2 to become the primary.

Other cluster management software can be used. However, SA MP and sample automation scripts are included with Db2 to manage the HA requirements of your Db2 database system.

Active/active application server cluster with database sharing

Figure 5-22 shows that database sharing was introduced by using Oracle RACs. Oracle RAC provides HA for applications by having multiple RAC nodes share a single copy of the data. If a cluster node fails, the in-flight transaction is lost but the other server in the RAC can receive all Java Database Connectivity (JDBC) requests.

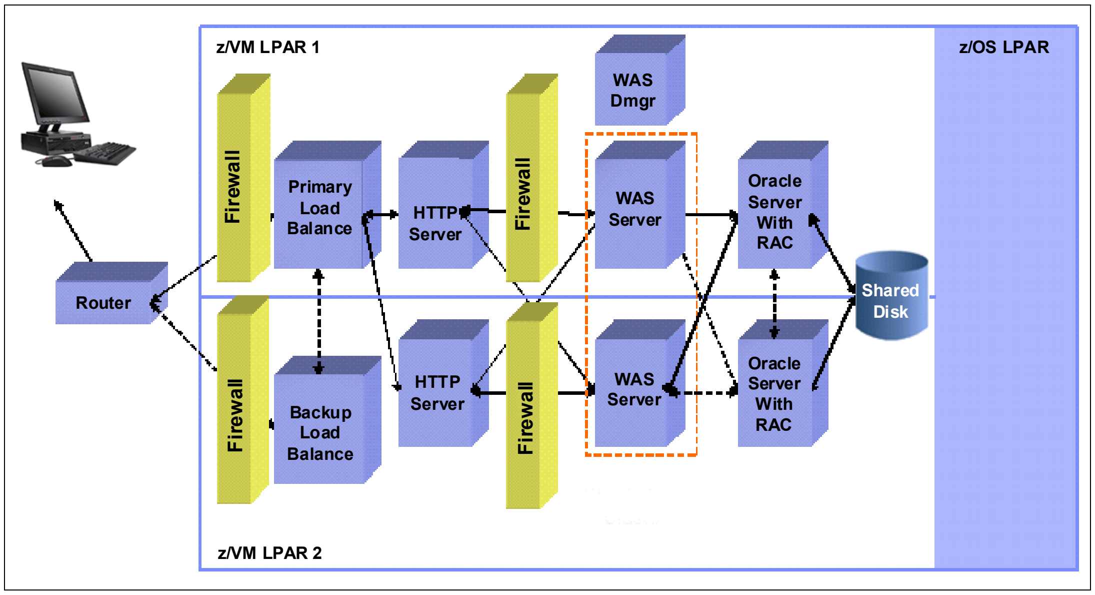


Figure 5-22 Database sharing that uses Oracle RAC

In a LinuxONE environment, communication between the database nodes uses a VLAN in the same LPAR or HiperSockets to other LPARs. Both methods are at memory-to-memory speeds with low latency.

For more information about Oracle RAC, see this website.

Active/active cluster with database sharing across cities

For the ultimate availability solution, You can have two sites a metropolitan distance apart, while providing data sharing between clusters at each site. This configuration is achieved with the IBM Globally Dispersed Parallel Sysplex® (IBM GDPS) Virtual Appliance. The GDPS Virtual Appliance supports both planned and unplanned outages, thus improving application availability and business continuity.

Distances greater than 100 km (62 miles) are also possible, although this configuration requires an asynchronous copy on the remote site, so it is not synchronized with the primary copy. For information about the GDPS Virtual Appliance, see *IBM GDPS: An Introduction to Concepts and Capabilities*, SG24-6374.

5.8.5 High availability capabilities by distribution

Several projects provide HA mechanisms for Linux on IBM LinuxONE. Consider your specific use case and necessities when you are looking for an HA third-party solution. Depending on the distribution that you chose for your environment, you might find add-ons that facilitate the HA implementation for you and help with installation, maintenance, and management. Highly flexible solutions allow for the creation of customized cluster environments with two or more nodes.

Generally available options

Projects that focus on HA are available for different layers of systems management, from managing network interfaces, application availability, and storage. A thorough analysis is required to determine what must be managed by an HA solution.

A combination of different solutions is required, each focusing on one or a few aspects of the application computing environment, as shown in the following examples:

- Storage: Distributed, highly available storage is the objective of a few projects:
  - GFS2: Created by Red Hat
  - OCFS2: Created by Oracle and named Oracle Cluster File System, is in its second version
  - GlusterFS: Now maintained by Red Hat
- Application availability:
  - Application service clustering is often managed by the application server, such as WebSphere (see “Active/active application server cluster” on page 137)
  - Pacemaker: Evolved from a Linux-HA project
  - Corosync: Group communication system, evolved from OpenAIS
  - Heartbeat: Simple clustering that is a main focus of the Linux-Ha project

Note: It is out of the scope of this book to discuss every available solution. For more information about implementation and options, see your distribution reference.

SUSE Linux Enterprise Server options

For more information about SUSE Linux Enterprise High Availability Extension, see this web page.

Red Hat Enterprise Linux options

For more information about Red Hat Enterprise Linux High Availability Add-On, see this web page.

Ubuntu options

For more information about Ubuntu HA, see this web page.

5.8.6 Understanding the availability requirements of your applications

This section describes how SLAs and the cost of providing availability can help you achieve a better understanding of the availability requirements of your applications.

Service-level agreements

To determine the availability requirements of applications that you want to migrate to LinuxONE, consider the needs of the business units that rely on these applications. Ideally, create SLAs that state requirements, such as availability needs, response time, maximum system utilization, and DR requirements. Use these SLAs as the basis for the design of the target system on Linux.

If SLAs do not exist, before designing a solution, discuss with the business units what levels of service you can offer and what level of investment that they are willing to make. The key to success for an SLA is that it is both achievable and measurable with defined penalties for failure to deliver. You also need to help ensure that SLAs are reviewed regularly because things will change.

According to IT Service Management principles, a SLA typically defines or covers the following topics:

- The services to be delivered
- Performance, tracking, and reporting mechanisms
- Problem and change management procedures
- Dispute resolution procedures
- The recipient's duties and responsibilities
- Security
- Legislative compliance
- Intellectual property and confidential information issues
- Agreement termination

Some of these components might not be relevant in an “in-house” SLA.

From an availability view point, an SLA for an “in-house” business application should focus on the first two items: What service is being delivered and how is it being measured:

- Application availability hours, for example:
  - 24 hours/day x 7 days a week.
  - 6:00 AM to 6:00 PM, weekdays.
  - 9:00 AM to 5:00 PM, weekdays.
  - Definition of how availability is measured and who will do the measurement. For example, system availability, application availability, database availability, and network availability.
- Minimum system response time

  Defined number and definition of where and how is it measured.

Cost of availability

As shown from the examples in this chapter, there is a great degree of difference in cost and complexity of the various availability options discussed. Providing CA and a DR plan is not an insignificant expense, but with the degree of reliance on IT systems by most businesses today, it is a cost that cannot be ignored.

If you have a web-facing revenue-generating application, you can calculate the cost of downtime by monitoring the average revenue that is generated in a specific amount of time. This amount provides an idea of the revenue that might be lost during an outage and how much you should spend to make the application more resilient. Other businesses have different ways of calculating the cost of downtime.

Keep in mind that for any HA configuration to be successful in a real DR situation, there needs to be a fully documented DR plan in place that is fully tested at least once every year.

5.9 Virtualized environment to LinuxONE cloud migration

Today's digital economy requires speed and agility to keep up with competitive demands for new applications and big data analytics, and maintain infrastructure. Many of these enterprises also operate large vendor-dependent virtualized infrastructures to support their mission-critical, scale-up applications and enterprise databases. They value various products for their compute, network, storage, and management technologies. But they are also encountering new use cases that demand the agility that is offered by cloud.

This section covers some of the technical aspects that needs to be considered before migrating a distributed virtualized environment onto IBM LinuxONE Cloud. This process is simple if the workloads targeted for LinuxONE cloud migration are virtualized. You cannot move the individual VMs directly on to a LinuxONE cloud, But you can create similar capabilities of the virtualized environment on the targeted LinuxONE cloud platform.

Before migration, create a chart of the following information for planning purposes:

- Performance and availability of a range of services and the linkages between them. Record performance metrics for each of the individual hypervisor.
- Application architectures and interdependencies.
- Network connectivity diagram for individual VMs and their applications.
- Security and isolation of networks among applications.
- Storage allocations, and their performance requirements.

5.9.1 Hypervisor considerations

Different hypervisors have different degrees of guest and function support. For example, in a OpenStack cloud platform, KVM and Xen Hypervisors are closely coupled with various network and storage support options. Here are the most important considerations:

- Standardize and harden VMs. Before the migration, plan and validate an approach about how the VMs are going to be patched, maintained, and monitored.
- Validate that the security, management, and monitoring tools can be extended or similar functions are available in the targeted LinuxONE Cloud platform.
- Validate the interoperability and isolation of applications' functional interfaces and cloud services interfaces.
- Use LinuxONE based Workload patterns for the deployment of standardized OS and middleware template for deployments.
- Analyze application portability on the targeted cloud platform and, if required, start application functional and performance testing activity on the cloud.

5.9.2 Network considerations

Even though cloud platforms mask underlying differences at the infrastructure layer to enable scale and dynamism, this homogeneity creates network bandwidth and provisioning challenges.

Bandwidth requirements

In a traditional virtualized environment, physical servers are connected by physical switches, which means that network administrators have total control of the network traffic and its security. When you move the workload onto a LinuxONE based cloud, monitor the current bandwidth utilization. In a cloud environment, the virtual switch has links from the physical switch through the physical NIC that attaches to VMs. Therefore, validate and identify whether LinuxONE infrastructure network cards and internal virtual switch connectivity are sufficient in terms of throughput and latency.

Virtual switch and VLAN tagging

In a LinuxONE Cloud Platform, virtualized networks also need a separation of VMs to help ensure data privacy of one tenant in the cloud from another client. Therefore, the networks need mechanisms to help ensure that they can share a physical network link without compromising or leaking information between networks. To enable access to a physical network, most cloud automation software (like OpenStack) uses the VLAN tagging model. This approach requires network administrators to create pools of VLAN IDs on a physical switch. When a new VM or virtual application is created from the cloud software, a cloud consumer uses these VLAN IDs without any delay.

5.9.3 Security considerations

Here are some security considerations for the LinuxONE cloud service:

- Authentication and authorization of users and administrators of the cloud service.
- Configuration and operation of encryption both for data that is stored within the cloud service and also for data that is transmitted to and from the cloud service.
- Firewalls and the configuration of other security capabilities.
- Virtual systems running on top of the hypervisor must have no means of network communication directly with the hypervisor.
- Hypervisor management traffic must use a separate physical network interface from the virtual guests.
- Create a separate security zone spanning LPARs and hypervisors, depending on the application and organizational requirements.
- For KVM-based hypervisors, extra technologies like SELinux can be enforced in addition to security or firewall zones.
- Standardized user interfaces, APIs, protocols, and data formats are defined for cloud services.
- Open technologies drive innovation and standardize open-source products for administration and business interfaces.
- Use existing or newer equivalent Access Management functions to authenticate and authorize access to VMs in cloud services.

5.9.4 LinuxONE cloud management

The following applications are available to help you with LinuxONE cloud management:

- OpenStack
- vRealize Automation for IBM LinuxONE
- IBM Cloud Infrastructure Center (CIC)

OpenStack

To provide cloud management capability, both z/VM and KVM are OpenStack-enabled, which is the industry standard for ubiquitous cloud computing platforms. Applications that use the OpenStack APIs are supported on both hypervisors. IBM CIC is an infrastructure as a service software product that helps establish a cloud infrastructure and simplifies management of a virtualized environment. It also offers more services such as identity management and orchestration that is accessed in the same programmatic manner through the API.

vRealize Automation for IBM LinuxONE

With VMware vRealize Automation (vRA), you can extend your existing vRealize Automation cloud to IBM LinuxONE. vRealize Automation interfaces with the OpenStack enabled cloud endpoints from IBM LinuxONE to provide cloud management services. This configuration validates the openness of IBM LinuxONE systems to allow clients to use the same cloud management that is used within their distributed cloud environment (Figure 5-23).

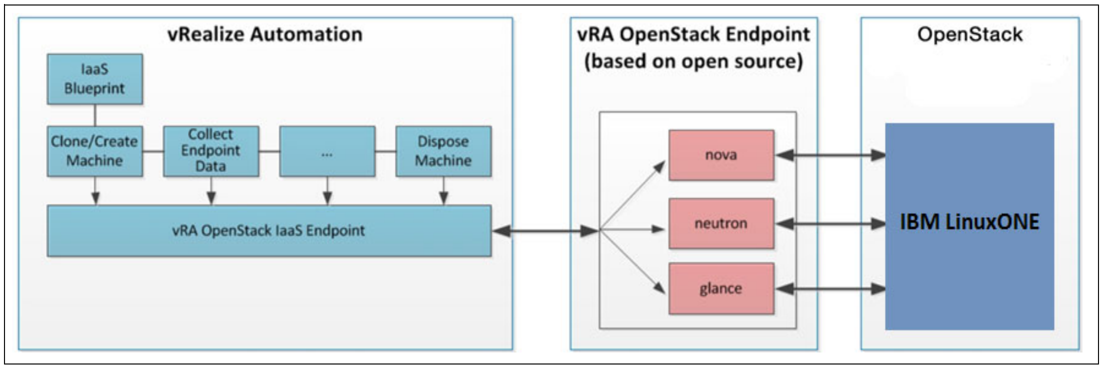


Figure 5-23 vRealize Automation with IBM LinuxONE framework

IBM Cloud Infrastructure Center

IBM CIC is an infrastructure management software that is designed to provide on-premises cloud deployments for enterprises. It is tailored for IBM Z and LinuxONE platforms, and offers many functions:

- Efficiently manage the lifecycle of VMs on IBM Z or IBM LinuxONE, inclusive of network and storage resources.
- Comprehensive capabilities encompass multi-tenancy, HA clustering, backup and restore functions, live migration, consistency grouping, snapshots, and other items.
- Leverage industry-standard OpenStack APIs for seamless and flexible automation and operations, which enable Infrastructure as Code (IaC) practices.
- Seamlessly integrate with various cloud platforms and tools, such as user service portals, IBM Instana, IBM Cloud Paks for Watson AIOps, Terraform, VMware vRealize, and others.
- Simplify the deployment and management of Red Hat OpenShift clusters, either as part of IBM Cloud Paks, or as stand-alone Red Hat OpenShift instances by using Ansible Playbooks.

IBM CIC provides the layer to manage the IBM Z/IBM LinuxONE based Infrastructure-as-a-Service (IaaS).

Figure 5-24 on page 145 shows the intersection between the infrastructure and the cloud platform, where IBM CIC is acting.

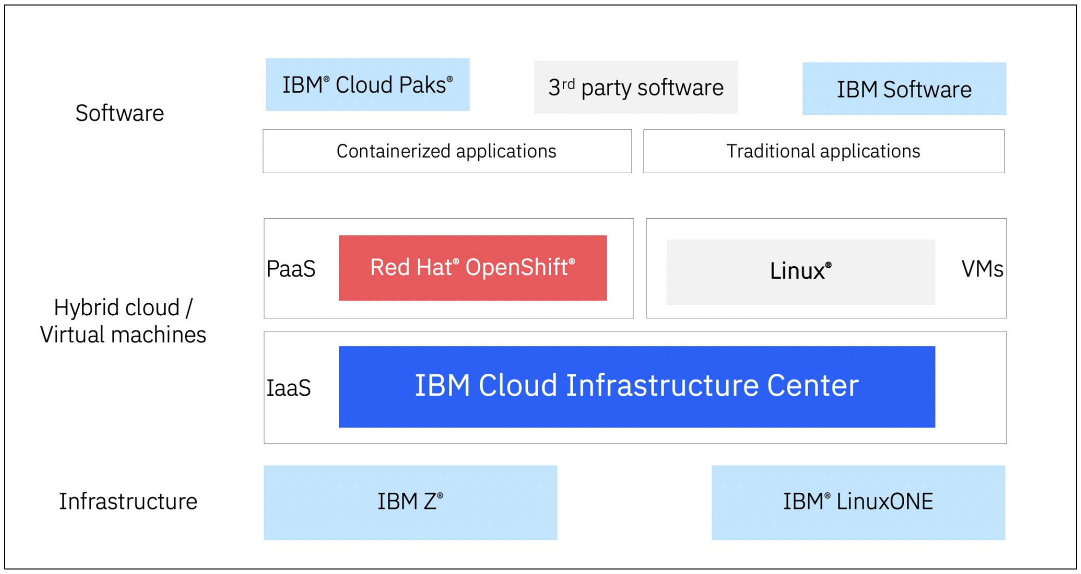


Figure 5-24 IBM Cloud Infrastructure Center

For more information about IBM CIC, see IBM Cloud Infrastructure Center 1.2.1.

For a sample configuration of a z/VM and KVM installation, see *Hybrid Cloud with On-premises Cloud on IBM Z or LinuxONE*, SG24-8530.

6

Hands-on migration

Now that the migration planning and analysis tasks are complete, test your migration plan by performing a hands-on migration in a test environment. During this process, load the test environment and help ensure that the performance meets business expectations and needs. After performing the test migration, plan for the migration of the production workloads.

Many methods of migrating your data from your source x86 servers to IBM LinuxONE are available, and many different ways of configuring your new IBM LinuxONE environment exist. A typical migration plan often involves a deep architectural understanding of the source application, as described in Chapter 5, “Migration analysis” on page 75.

After the main migration tasks complete, applications can be tuned to use many of the platform features, such as the Pause-less Garbage Collection for Java workloads, using the LinuxONE Guarded Storage Facility for improved throughput and response times, and Pervasive Encryption for encryption of data in-flight and at-rest.

This chapter describes a hands-on migration scenario that is performed in a lab in which we migrated a full working application, which was composed of a WebSphere Application Server Network Deployment cluster, Db2, IBM WebSphere MQ, and a simple Node.js application from several x86 servers to IBM LinuxONE. We also illustrate the migration process from an IBM Open Liberty® instance on x86 to IBM LinuxONE.

This chapter includes the following sections:

6.1 Environment setup

This section describes the required tasks to create virtual guests on IBM LinuxONE. First, analyze the reference application architecture to determine what resources are used in the x86 environment. During that process, complete the planning checklists regarding that architecture. Next, determine what to use in the IBM LinuxONE environment with performance in mind. Create the Linux guests in the logical partition (LPAR). Finally, perform the migration tasks.

6.1.1 Source application architecture

IBM LinuxONE supports various workloads and migration strategies, which meet most of the businesses needs. A deep understanding of an application architecture and how each of its components are interconnected are crucial for meeting a success criteria during your migration process.

Figure 6-1 shows our application architecture, which describes how each of the required components is connected to our application to function.

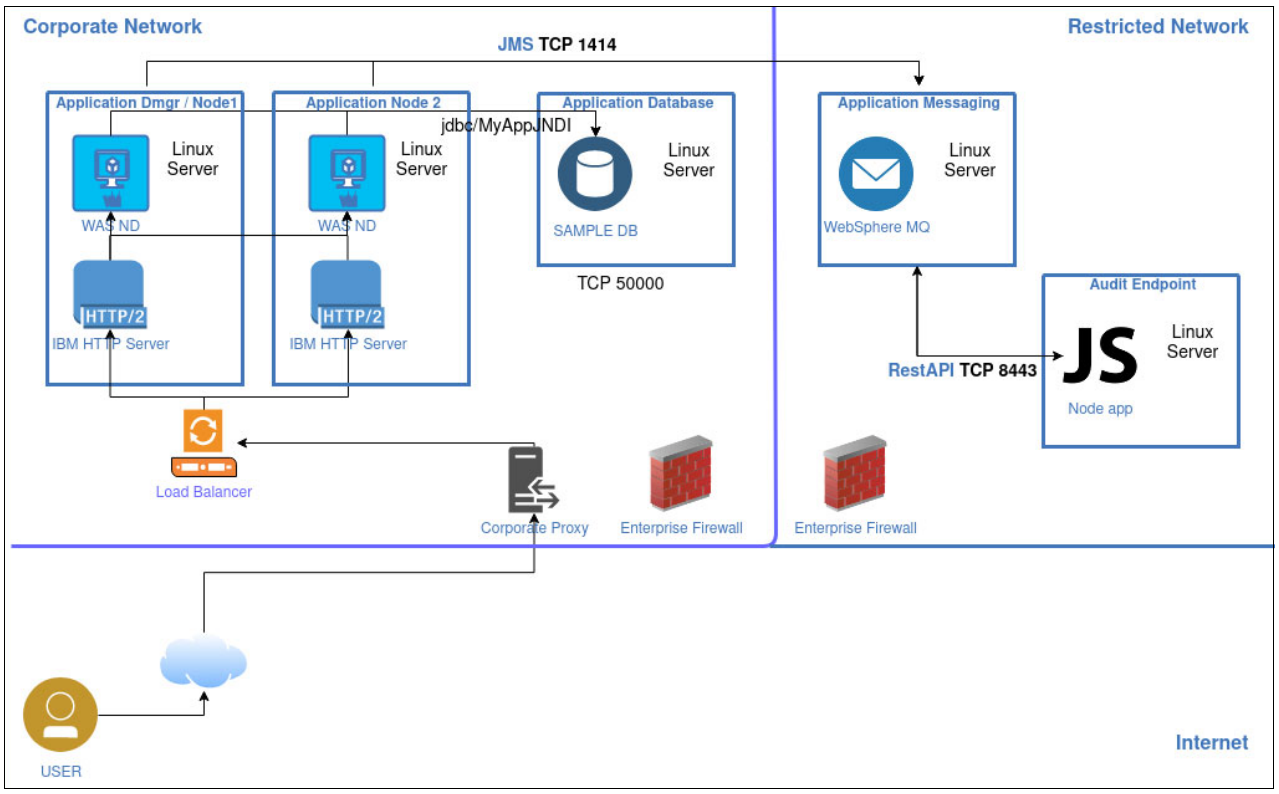


Figure 6-1 Application reference architecture

Our application represents most of the workloads that are used by the organizations. Simpler or more complex scenarios might exist, which can require different migration approaches in accordance to each application and businesses requirements.

6.1.2 Software products and tools checklist

During the meetings with our key stakeholders, it was decided that all components that are used by our reference application must be migrated to IBM LinuxONE. It was also decided that the same levels of each software technology that is used in the source x86 workloads must be in place under our target LinuxONE guests. This migration approach, on which the same levels of software are implemented in the target system while retaining the same application architecture, is often referred to "as-is".

Table 6-1 lists the software products that compose our source x86 application.

Table 6-1 Software products and tools checklist for the x86 environment

Software products and tools checklist for the x86 environment				
Name	**Version**	**Vendor**	**License type**	**LinuxONE**
Db2	11.5.4.0	IBM	Proprietary	Supported
WebSphere Application Server Network Deployment	9.0.5.5	IBM	Proprietary	Supported
WebSphere MQ	9.2.0.0	IBM	Proprietary	Supported
Node.js	12.16.1	Node.js	MIT and ASL 2.0 and ISC and BSD	Supported

6.1.3 Hardware checklist

This section lists the physical and virtual hardware resources that are needed. For the servers that are used in this project, the source environment's hardware resources were gathered and used as a reference before provisioning our IBM LinuxONE guests.

Table 6-2 shows one of the checklists that were used to map the source x86 workloads to our target IBM LinuxONE guests.

Table 6-2 Hardware checklist for the x86 environment

Hardware planning checklist			
Server name: lnx-x86-was			
Resource	**Source (x86)**	**Destination (LinuxONE)**	**Observation**
Number of CPU	4	2	Real to Virtual
System memory (in GB)	8	8	
OS swap Memory (in GB)	4	4	Disk to VDisk
Network connection[a]			
Connection Description	Gigabit Ethernet	Gigabit Ethernet	
Connection Type	Gigabit Ethernet	vSwitch/GbE	
Volume groups (VGs): vg_system VG: 40 GB vg_websphere VG: 40 GB vg_data VG: 10 GB			

Hardware planning checklist			
Server name: lnx-x86-was			
IP Address/Netmask	129.40.19.88/24	129.40.23.153/24	
Virtual local area network (VLAN) number: vSwitch	2	2: vSwitch1	
Disk resource[b]			
OS file system	/ : 80 : Ext3	/ : 40 :Ext4	ECKD DASD
Mount point: Size (in GB): Type		/opt : 3 :Ext4	LV
Mount point: Size (in GB): Type		/home: 10 :XFS	LV
Mount point: Size (in GB): Type		/var : 5 :Ext4	LV
Mount point: Size (in GB): Type		/tmp : 1 :Ext4	LV
Data file system			
Mount point: Size (in GB): Type	/opt/WebSphere : 10 : Ext3	/opt/WebSphere : 10 : XFS	LV (vg_websphere)
Mount point: Size (in GB): Type	/web : 5 : Ext3	/web : 5 : XFS	LV (vg_data)
Custom file system			
Mount point: Size (in GB): Type		/tempspace : 10 : XFS	LV (vg_websphere)
Volume groups (VGs): vg_system VG: 40 GB vg_websphere VG: 40 GB vg_data VG: 10 GB			

a. For IBM LinuxONE, the available network connections are QETH, HiperSockets, and Direct OSA-Express.
b. Logical Volume Manager (LVM) provides storage management flexibility and reduced downtime with online resizing of logical volumes (LVs)

6.1.4 Small Computer System Interface over Fibre Channel Protocol

The zfcp device driver is a Linux kernel module that acts as a host-bus adapter driver and supplements the Linux Small Computer System Interface (SCSI) stack, which supports many SCSI devices, various hardware adapters, and different topologies.

Figure 6-2 on page 151 shows how the device drivers work together to provide storage medium access through the Fibre Channel Protocol (FCP) under IBM LinuxONE. Do not confuse FCP devices with SCSI devices because although the former connects Linux to the Fibre Channel storage area network (SAN) and is managed through the zfcp driver, the latter represents a Logical Unit Name (LUN) and is controlled through the Linux SCSI subsystem.

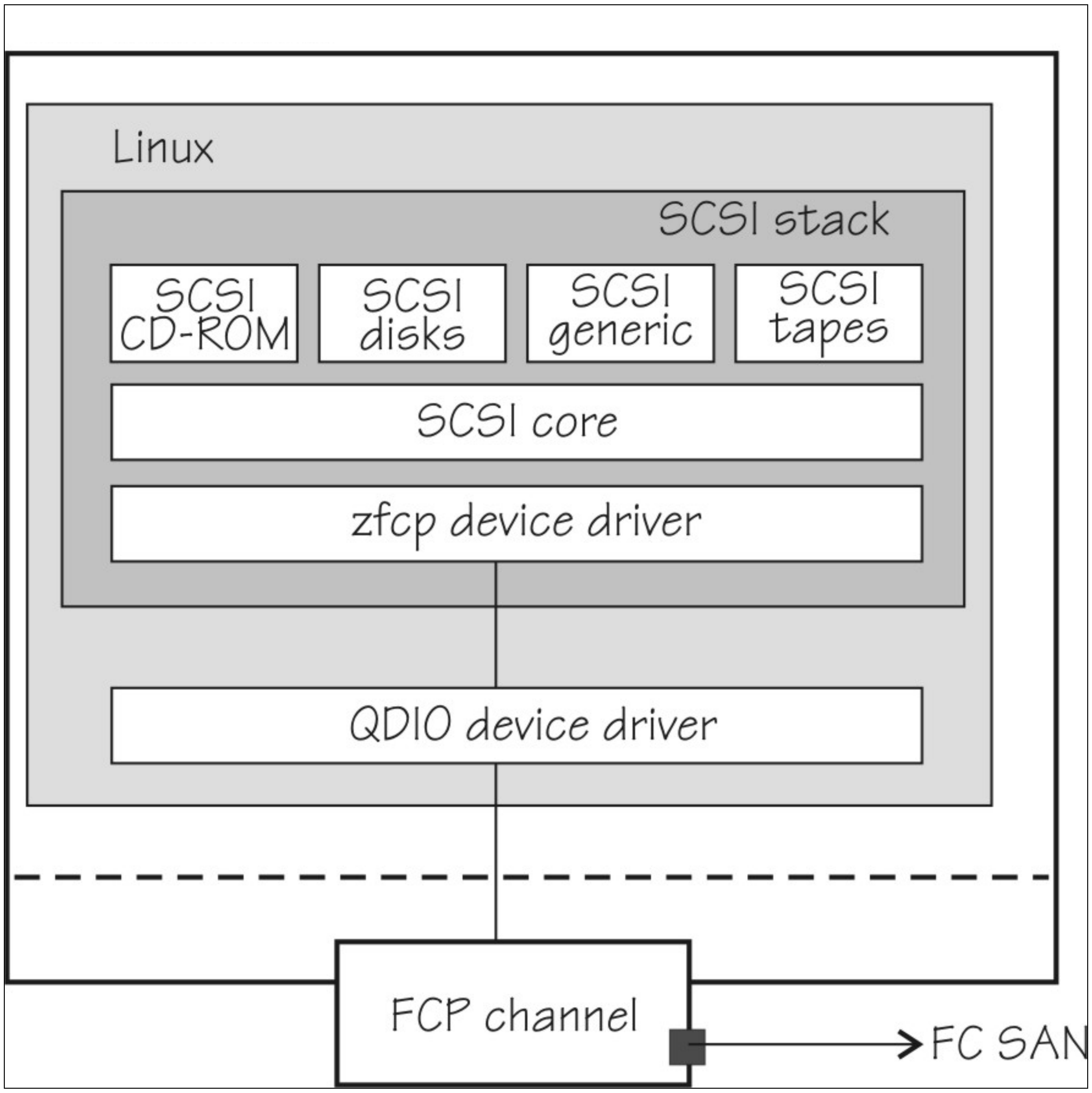


Figure 6-2 Device drivers that support the FCP environment

Because the zfcp module is handled by the Linux kernel, the failover configuration is not handled by PR/SM or z/VM, and must be done from within the Linux guest. Therefore, as described in 3.8.3, “Virtualized disk” on page 58, attach at least two N_Port ID Virtualization (NPIV) adapters to the guest system. Each adapter must be connected over different Fibre Channel fabrics for redundancy.

6.1.5 FCP migration setup tasks

The migration of Fibre Channel devices from distributed x86 systems to IBM LinuxONE with NPIV involves different tasks that often occur in the following order:

1. Configure an input/output definition file (IODF).
2. Create and define switch zoning.
3. Set up masking on a storage LUN.
4. Attach the FCP devices to Linux.
5. Configure zfcp and Linux multipathing.

Many of these activities require the presence of a multidisciplinary team and different approaches depending on the switch, storage, and virtualization technologies that are used in your organization. Because various configurations can exist, this chapter briefly covers the configuration of zfcp and Linux multipathing. It is assumed that the FCP devices are attached to your Linux system according to your organization's requirements.

For more information, see *How to use FC-attached SCSI devices with Linux on IBM Z Systems*, SC33-8413.

6.1.6 Configuring zfcp

By default, FCP devices are offline to prevent Linux from automatically scanning and performing input/output (I/O) against its targets. If you are migrating your data through FC-attached SCSI devices, help ensure that the source system suitably stopped all I/O activity on the affected LUNs and detached them before proceeding. Failure to do so can result in unpredictable data loss.

With the FCP devices attached to the Linux system, bring each one online by running the **chzdev** command along with its bus ID, as shown in Example 6-1.

Example 6-1 Bringing online an FCP device

```
chzdev -e zfcp-host 0.0.f100
```

Running this command automatically configures the FCP device for the active and persistent system configuration. To bring an FCP device for the active configuration only so that it does not persist across restarts, pass the **-a** parameter for the **chzdev** command, as shown in Example 6-2.

Example 6-2 Bringing online an FCP device only for the active system configuration

```
chzdev -e zfcp-host 0.0.f100 -a
```

Most modern Linux distributions automatically discover the available ports from their targets and perform auto LUN scanning from these ports. Ideally, putting the FCP device online should be the only activity that is required for the SCSI devices to be presented to the Linux guest. However, depending on your distribution or application requirements, you might want to override these settings.

Table 6-3 shows the zfcp module parameters that are responsible for handling these settings.

Table 6-3 Disabling or enabling auto port and LUN scanning

zfcp module parameter	Valid options	Description
`no_auto_port_rescan`	0 / 1	Enable/Disable auto port scanning.
`allow_lun_scan`	0 / 1	Disable/Enable auto LUN scanning.

The parameters for the zfcp kernel module are normally set under the Linux bootloader configuration to help ensure that they persist across system restarts. Example 6-3 shows how to disable automatic LUN scanning.

Example 6-3 Linux kernel bootloader entry

```
root=/dev/disk/by-path/ccw-0.0.0100 rd.dasd=0.0.0100 zfcp.allow_lun_scan=0
cio_ignore=all,!condev
```

Note: Many distributions implement their own tools for configuring and scanning through FCP. IBM also frequently releases documentation covering several administrative tasks that are applicable to the major LinuxONE supported distributions. For more information, see Introduction to Linux on IBM Systems.

6.1.7 Linux multipathing

As a best practice, all users of SCSI over Fibre Channel should have a resilient setup of redundant paths across their FC SAN. The device-mapper multipath is the Linux tool that is responsible for providing path redundancy when such a setup is present, which improves the LUN availability if a problem occurs within one of the SAN components.

Most Linux distributions include the multipath tool that is installed by default. Example 6-4 shows how to install and enable the multipathd daemon under a Red Hat Enterprise Linux (RHEL) 8.2 system.

Example 6-4 Installing and enabling the device-mapper multipath under an RHEL 8.2 system

```
dnf install device-mapper-multipath
mpathconf --enable
systemctl enable --now multipathd
```

The multipath daemon configuration file is stored at `/etc/multipath.conf`, and it is responsible for tuning several aspects concerning the way the SCSI devices should be handled by the system. The configuration of the multipath daemon depends on the storage device that is used. For IBM DS8000® Storage Systems, the configuration that is shown in Example 6-5 can be used.

Example 6-5 A multipath.conf sample configuration file

```
defaults {
  path_grouping_policy multibus
  failback 1
  rr_min_io 10
  path_checker tur
  checker_timeout 60
}

devices {
  device {
    vendor "IBM"
    product ""
    path_grouping_policy group_by_prio
    prio alua
  }

}
```

After the `multipathd` daemon is started, all available LUNs together with their paths can be checked by using the **`multipath -ll`** command.

6.2 Migrating Db2 and its data

The migration of Db2 and other database management systems (DBMSs) (as described in 4.1, "Stakeholder definitions" on page 64) involves input from the key business stakeholders, the database administrator (DBA), and the primary systems administrators from the source x86 and target IBM LinuxONE systems. The migration strategy that is used typically depends on several factors, such as the size of the databases; number of databases to be migrated; network speed, storage, and backup technologies that are used; and back-out plans.

Migrating databases typically involves downtime, especially when handling production data. Always help ensure that the source Db2 instances are stopped and that the applications that rely on them are routed to the new server after the migration completes. Failure to do so can result in the target system running out of sync with its source server in such a way that the migration steps must be repeated from scratch.

Note: Always perform a full database backup before the migration occurs. Taking this precaution prevents common human errors during the process and also allows for a point-in-time database restore (if necessary).

This section describes the migration of Db2 data from a source x86 system (`lnx-x86-db2`) to an IBM LinuxONE guest (`lnxone-db2`). We perform the migration of the same database by using two different methods: the `db2move/db2look` and the `LOAD FROM CURSOR` utilities.

6.2.1 Preliminary migration steps

Complete the following steps:

1. From within the source x86 system, switch to the Db2 instance ID (`db2inst1` in our environment) and perform a full database backup, as shown in Example 6-6. Notice how we help ensure that the database is not in use before proceeding with its backup.

Example 6-6 Performing a full database backup before any other activity occurs

```
lnx-x86-db2 $ db2 list applications
SQL1611W  No data was returned by Database System Monitor.
lnx-x86-db2 $ db2 deactivate db SAMPLE
DB20000I  The DEACTIVATE DATABASE command completed successfully.
lnx-x86-db2 $ db2 BACKUP DATABASE SAMPLE TO /db2_backup COMPRESS

Backup successful. The timestamp for this backup image is : 20201018112258

lnx-x86-db2 $ ls -ltr /db2_backup/
total 28768
-rw------- 1 db2inst1 db2admin 29458432 Oct 18 11:23
SAMPLE.0.db2inst1.DBPART000.20201018112258.001
```

2. Still under the source x86 systems, retrieve the list of authorization IDs that are required for your application to function, as shown in Example 6-7. In our application, we are required to have a group that is named `appdb`. The members of this group are granted privileges to our migrated databases. For more information about requirements, depending on your environment, see the Db2 product documentation and contact your stakeholders.

Example 6-7 Retrieving the list of grantees for our database

```
lnx-x86-db2 $ db2 connect to SAMPLE

   Database Connection Information

 Database server        = DB2/LINUXX8664 11.5.4.0
 SQL authorization ID   = DB2INST1
 Local database alias   = SAMPLE

lnx-x86-db2 $ db2 select '*' from SYSIBMADM.AUTHORIZATIONIDS

AUTHID AUTHIDTYPE
------ ----------
APPDB G
PUBLIC G
SYSDEBUG R
SYSDEBUGPRIVATE R
SYSTS_ADM R
SYSTS_MGR R
DB2INST1 U

  7 records selected.
```

3. From the target IBM LinuxONE system, create the Db2 instance ID that houses your databases. It is a best practice to also create a dedicated common group for all instances you are going to use. Also, create the authorization IDs and groups that are required for your application to function. Example 6-8 shows the creation of our required application Db2 IDs and groups.

Example 6-8 Required ID creation under lnxone-db2

```
lnxone-db2 # groupadd appdb
lnxone-db2 # useradd -g appdb -c 'User required by our application' -m appuser
lnxone-db2 # groupadd db2admin
lnxone-db2 # useradd -g db2admin -c 'Db2 instance ID' -m db2inst1
lnxone-db2 # useradd -g db2admin -c 'Db2 fenced ID' -m db2fenc1
```

4. After the Db2 product installation under the `lnxone-db2` server is done, create the application-required file systems and adjust the required kernel parameters. Then, create the Db2 instance. Example 6-9 shows the creation of the `db2inst1` Enterprise Server Edition (ese) instance that listens on TCP port 50000 and uses `db2fenc1` as its fenced ID.

Example 6-9 Db2 instance creation

```
lnxone-db2 /opt/ibm/db2/V11.5/instance # ./db2icrt -s ese -p 50000 -u db2fenc1
db2inst1
DBI1446I  The db2icrt command is running.
(... output suppressed ...)
The execution completed successfully.
```

```
For more information, see the Db2 installation log at "/tmp/db2icrt.log.3792".
DBI1070I  Program db2icrt completed successfully.
```

5. Start the previously created Db2 instance so that it can be used during the migration, as shown in Example 6-10.

Example 6-10 Starting the Db2 instance

```
lnxone-db2 # su - db2inst1
lnxone-db2 $ db2start
10/18/2020 11:33:08     0   0   SQL1063N  DB2START processing was successful.
SQL1063N  DB2START processing was successful.
```

6.2.2 Data migration by using db2move and db2look

Complete the following steps:

1. Use the **db2look** tool to extract the required Data Definition Language (DDL) statements to reproduce the database objects of one database into another database. The tool can also generate the required SQL statements to replicate the statistics from one database to the other, and the statements that are needed to replicate the database configuration, database manager configuration, and registry variables. The **db2look** command does not extract any table data. This task is described later in this section and uses the **db2move** tool.

   Example 6-11 shows how to use the **db2look** command to generate the required DDL statements for replicating our source x86 database objects to our target IBM LinuxONE system.

Example 6-11 Generating the DDL by using the db2look command

```
lnx-x86-db2 $ db2look -d SAMPLE -e -x -l -createdb -o /db2_temp/db2look/SAMPLE.sql
-- No user ID was specified, db2look tries to use Environment variable USER
-- USER is: DB2INST1
-- Creating DDL for tables
-- Output is sent to the file: /db2_temp/db2look/SAMPLE.sql
```

   The command includes the following parameters:

   – **-d**: Name of the database.

   – **-e**: Extracts the database objects.

   – **-x**: Generates authorization DDL statements, such as **GRANT** statements.

   – **-l**: Generates DDL statements for user-defined database objects.

   – **-createdb**: Generates the **CREATE DATABASE** command that is used to create the source database.

   – **-o**: The name of the output file.

   For more information about the **db2look** tool, see IBM Documentation.

   The resulting `Db2 Backup` file cannot be used to move data between x86 and IBM LinuxONE operating systems. Use the **db2move** utility to export the source x86 tables data, as shown in Example 6-12 on page 157. By default, the **db2move** utility generates its exported files in the current working directory. Help ensure that you switch to the wanted destination directory before running this command because several files are created for every table in the database.

Example 6-12 The db2move export utility

```
lnx-x86-db2 $ cd /db2_temp/db2move
lnx-x86-db2 $ db2move SAMPLE export
lnx-x86-db2 $ ls
db2move.lst     tab11.ixf  tab13.msg  tab16a.001.lob  tab17.msg  tab1.ixf
tab21.ixf  tab23.msg       tab25.msg  tab4a.001.xml  tab6.ixf  tab8.msg
EXPORT.out      tab11.msg  tab14.ixf  tab16.ixf       tab18.ixf  tab1.msg
tab21.msg  tab24.ixf       tab2.ixf   tab4.ixf       tab6.msg  tab9a.001.lob
tab10a.001.lob  tab12.ixf  tab14.msg  tab16.msg       tab18.msg  tab20.ixf
tab22.ixf  tab24.msg       tab2.msg   tab4.msg       tab7.ixf  tab9.ixf
tab10.ixf       tab12.msg  tab15.ixf  tab17a.001.xml  tab19.ixf  tab20.msg
tab22.msg  tab25a.001.xml  tab3.ixf   tab5.ixf       tab7.msg  tab9.msg
tab10.msg       tab13.ixf  tab15.msg  tab17.ixf       tab19.msg  tab21a.001.xml
tab23.ixf  tab25.ixf       tab3.msg   tab5.msg       tab8.ixf
```

2. After all the required Db2 data is exported, copy the results from the **`db2move`** and **`db2look`** utilities from the source x86 system to the target IBM LinuxONE server. We use the **`rsync`** tool to complete this task.
3. Connected to the target `lnxone-db2` system, create the database and its objects by using the DDL file that was generated by the **`db2look`** utility, as shown in Example 6-13.

Example 6-13 Creating Db2 database objects

```
lnxone-db2 $ db2 -tvf /db2_temp/SAMPLE.sql -z $HOME/SAMPLE_creation.log
```

Note: Review the output of the command that is shown in Example 6-13. Minor adjustments to the generated DDL file might be necessary, especially when migrating between different Db2 releases. If so, DROP the database and retry the operation after reviewing the SQL statements. For more information, see IBM Documentation.

4. After the Db2 objects migrate, load the data into the database by using the **`db2move`** utility. Finally, run **`SET INTEGRITY`** on any tables that might require it, as shown in Example 6-14.

Example 6-14 Data restore by using the db2move utility

```
lnxone-db2 $ db2move SAMPLE load
lnxone-db2 $ db2 connect to SAMPLE
lnxone-db2 $ db2 "SELECT tabname,status,const_checked FROM syscat.tables WHERE
status='C'"
TABNAME STATUS CONST_CHECKED
------- ------ -------------
EMPLOYEE C      YYYYNYYYYYYYYYYYYYYYYYYYYYYYYYYY
DEPARTMENT C YYYYNYYYYYYYYYYYYYYYYYYYYYYYYYYY
2 records selected.

lnxone-db2 $ db2 SET INTEGRITY FOR EMPLOYEE,DEPARTMENT IMMEDIATE CHECKED
DB20000I  The SQL command completed successfully.
```

You can now connect to the target database and query its data.

6.2.3 Migration by using the LOAD utility with the FROM CURSOR option

The **db2move** utility that is described in 6.2.2, "Data migration by using db2move and db2look" on page 156 is useful for migrating small and less complex databases. However, depending on your database requirements, complexity, and size, you might prefer to directly load its data to your target system without first exporting the source tables to a data exchange format and then transferring it over the network.

Complete the following steps:

1. From the `lnxone-db2` server, create and migrate the SAMPLE database objects by using the same file that was generated by using the **db2look** utility. Then, run the Db2 **CATALOG** command to define a connection to the source x86 database, as shown in Example 6-15.

Example 6-15 Database creation and source database catalog

```
lnxone-db2 $ db2 -tvf /db2_temp/SAMPLE.sql -z $HOME/SAMPLE_creation.log
lnxone-db2 $ db2 CATALOG TCPIP NODE X86DB2 REMOTE 129.40.19.90 SERVER 60000
DB20000I  The CATALOG TCPIP NODE command completed successfully.
DB21056W  Directory changes might not be effective until the directory cache is
refreshed.
lnxone-db2 $ db2 CATALOG DB SAMPLE AS X86SAMP AT NODE X86DB2 AUTHENTICATION SERVER
DB20000I  The CATALOG DATABASE command completed successfully.
DB21056W  Directory changes might not be effective until the directory cache is
refreshed.
```

2. Recycle the Db2 instance by using the **db2stop** and **db2start** commands. Example 6-16 shows the SQL statements that we created to load the required tables data from our source database to our target LinuxONE server.

Example 6-16 Loading from cursor SQL statements

```
DECLARE cur CURSOR DATABASE X86SAMP USER db2inst1 USING password FOR SELECT * FROM
DB2INST1.EMPLOYEE WITH UR;
LOAD FROM cur OF CURSOR REPLACE INTO DB2INST1.EMPLOYEE unrecoverable;

DECLARE cur CURSOR DATABASE X86SAMP USER db2inst1 USING password FOR SELECT * FROM
DB2INST1.DEPARTMENT WITH UR;
LOAD FROM cur OF CURSOR REPLACE INTO DB2INST1.DEPARTMENT NONRECOVERABLE;
```

3. Save the SQL statements to a file that is named `CURSOR.sql`. Then, issue a database **CONNECT** to the target database. Finally, run the `CURSOR.sql` statements, as shown in Example 6-17.

Example 6-17 Data loaded directly from source to destination

```
lnxone-db2 $ db2 CONNECT to SAMPLE

   Database Connection Information

 Database server        = DB2/LINUXZ64 11.5.4.0
 SQL authorization ID   = DB2INST1
 Local database alias   = SAMPLE

lnxone-db2 $ db2 -tvf CURSOR.SQL
DECLARE cur CURSOR DATABASE X86SAMP USER db2inst1 USING         FOR SELECT * FROM
DB2INST1.EMPLOYEE WITH UR
DB20000I  The SQL command completed successfully.
```

```
LOAD FROM cur OF CURSOR REPLACE INTO DB2INST1.EMPLOYEE NONRECOVERABLE
SQL3253N  The utility is beginning to load data from the SQL statement "
SELECT * FROM DB2INST1.EMPLOYEE WITH UR" in database "X86SAMP".
(...) output suppressed (...)
```

When the LOAD FROM CURSOR operation finishes, the database is ready for use.

6.3 Moving IBM MQ

The process of moving an IBM MQ queue manager to another system involves re-creating it on the target system. Our source system is named `lnx-x86-mq` and our target system is named `lnxone-mq`.

Complete the following steps:

1. In the source system, save the information of the queue manager that you want to move and its authorities, as shown in Example 6-18.

Example 6-18 Dumping the source queue manager configuration and saving its authorities

```
lnx-x86-mq $ dmpmqcfg -m QM1 -a > /tmp/QM1.mqsc
lnx-x86-mq $ amqoamd -m QM1 -s > /tmp/QM1AUT.mqsc
```

   The command includes the following parameters:

   – **`-m`**: Represents the Queue Manager from which we want to dump the configuration.
   – **`-a`**: Tells the `dmpmqcfg` program that we want to dump all attributes.
   – **`-s`**: Specifies that we want the output to be generated in the **`setmqaut`** command format.

2. After dumping its attributes, quiesce the source queue manager and stop it, as shown in Example 6-19. Do not proceed until the queue manager is fully down. You might need to manually stop applications that might be connected to the queue manager.

Example 6-19 Quiesced shutdown

```
lnx-x86-mq $ endmqm -c QM1
Quiesce request accepted. The queue manager stops when all outstanding work
is complete.
lnx-x86-mq $ dspmq
QMNAME(QM1)                                                STATUS(Ended normally)
```

3. Copy the generated files to the target LinuxONE system. Then, after completing the product installation and its preliminary setup for your environment, create and start the target queue manager, as shown in Example 6-20.

Example 6-20 Creating and starting the target queue manager under IBM LinuxONE

```
lnxone-mq $ crtmqm -q QM1
IBM MQ queue manager created.
Directory '/mnt/mqm/data/qmgrs/QM1' created.
The queue manager is associated with installation 'Installation1'.
Creating or replacing default objects for queue manager 'QM1'.
Default objects statistics : 83 created. 0 replaced. 0 failed.
Completing setup.
Setup completed.
lnxone-mq $ strmqm QM1
```

```
IBM MQ queue manager 'QM1' starting.
The queue manager is associated with installation 'Installation1'.
5 log records accessed on queue manager 'QM1' during the log replay phase.
Log replay for queue manager 'QM1' complete.
Transaction manager state recovered for queue manager 'QM1'.
IBM MQ queue manager 'QM1' started using V9.2.0.0.
```

4. Before restoring the queue manager object definitions, create the necessary user IDs and groups that are required for the application to function correctly. Review the `QM1AUT.mqsc` file and determine which privileges might be required for the application to function. When in doubt, consult with your application representatives.

   Example 6-21 shows how we retrieved the list of required IDs and groups from our `QM1AUT.mqsc` file. We proceeded with the creation of the required accounts.

Example 6-21 Retrieving and creating the necessary queue manager groups and IDs

```
lnxone-mq $ awk -F'-p' '{ print $2 }' QM1AUT.mqsc | awk '{ print "User: ",$1 }' |
sort -V | uniq
User:  mqm
User:  appuser
User:  wassrvr
lnxone-mq $ awk -F'-g' '{ print $2 }' QM1AUT.mqsc | awk '{ print "Group: ",$1 }' |
sort -V | uniq
Group:  appdb
Group:  mqm
lnxone-mq $ groupadd appdb
lnxone-mq $ useradd -c ‘WebSphere process ID’ -g appdb wassrvr
lnxone-mq $ useradd -c ‘Nodejs appuser’ -g appdb appuser
```

5. Import the source queue manager configuration and its authorities, as shown in Example 6-22.

Example 6-22 Restoring the queue manager objects and its authorities

```
lnxone-mq $ runmqsc QM1 < /mq_data/QM1.mqsc > /tmp/QM1.log
lnxone-mq $ $ tail -4 /tmp/QM1.log
       :
********************************************************************************
231 MQSC commands read.
No commands have a syntax error.
All valid MQSC commands were processed.

lnxone-mq $ chmod +x /mq_data/QM1AUT.mqsc
lnxone-mq $ ./mq_data/QM1AUT.mqsc
The setmqaut command completed successfully.
The setmqaut command completed successfully.
The setmqaut command completed successfully.
(...) output suppressed (...)
```

6. Review the output of all commands and help ensure that no errors were triggered. Finally, refresh the queue manager security, as shown in Example 6-23 on page 161. The queue manager is ready to be used.

Example 6-23 Refreshing security

```
lnxone-mq $ echo 'refresh security(*)' | runmqsc
5724-H72 (C) Copyright IBM Corp. 1994, 2020.
Starting MQSC for queue manager QM1.

1 : refresh security(*)
AMQ8560I: IBM MQ security cache refreshed.
One MQSC command read.
No commands have a syntax error.
All valid MQSC commands were processed.
```

6.3.1 Moving the IBM MQ Web Server and the RESTful API component

IBM MQ 9.0.4 introduced support for RESTful APIs. It enables users and applications to send **POST** and **DELETE HTTP** commands against specific URLs to put and receive data from IBM MQ queues. RESTful APIs enable faster integration of IBM MQ with other technologies, such as cloud and container-based solutions.

IBM improves and updates the RESTful API functions in IBM MQ. For more information about all the new features that were introduced since Version 9.0.4, see the IBM MQ product documentation.

Our application depends on the RESTful API functions that are present on IBM MQ to function correctly. To enable the required functions, help ensure that the MQSeriesWeb component is installed along with the IBM MQ product, as shown in Example 6-24.

Example 6-24 The MQSeriesWeb component is installed

```
lnxone-mq $ rpm -q MQSeriesWeb
MQSeriesWeb-9.2.0-0.s390x
```

Complete the following steps:

1. From the source x86 system, copy the `mqwebuser.xml` file to the destination LinuxONE server. This file is typically in the following folder:

   `<QM_PREFIX> /web/installations/installationName/servers/mqweb`

2. If your solution requires receiving RESTful API calls from systems other than localhost, enable remote connections to the mqweb server. Example 6-25 shows how to bind the mqweb server to all available network interfaces.

Example 6-25 Enabling the mqweb server on all available network interfaces

```
lnxone-mq $ setmqweb properties -k httpHost -v "*"
MQWB1100I: The 'setmqweb' command completed successfully.
```

   The IBM MQ web server typically requires a few adjustments to make it work. If any customizations were done on the source x86 system, you can display its modified properties by running the **dspmqweb** command, as shown in Example 6-26.

Example 6-26 Displaying the user modified values

```
lnx-x86-mq $ dspmqweb properties -u
name="httpHost" value="*"
name="managementMode" value="externallyprovisioned"
MQWB1100I: The 'dspmqweb' command completed successfully.
```

3. Migrate all modified settings to the IBM LinuxONE server, and start the mqweb server component, as shown in Example 6-27.

Example 6-27 Starting the IBM MQ web server

```
lnxone-mq $ strmqweb

Starting server mqweb.
Server mqweb started with process ID 5451.
```

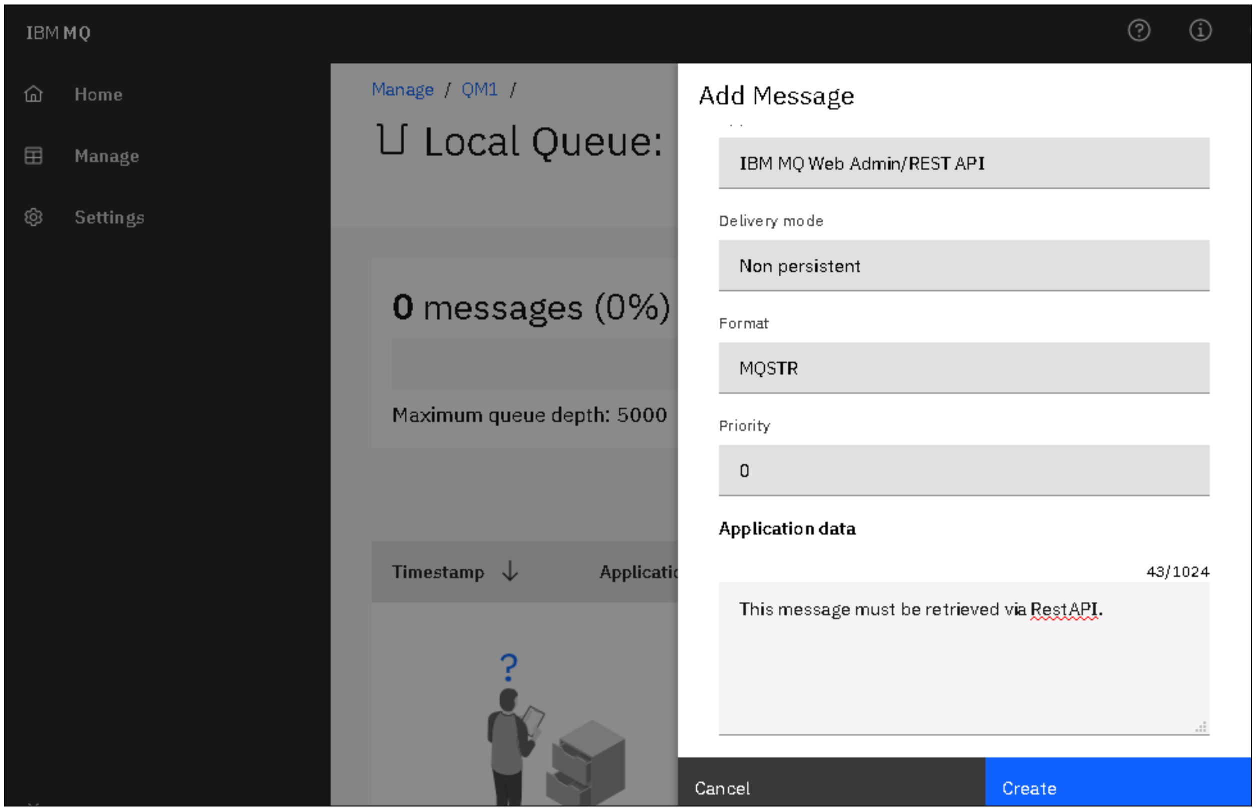
4. Analyze your web server `console.log` file for possible errors that might occur. Typical problems include missing keystores and truststores for Transport Layer Security (TLS) connectivity. Example 6-28 shows how to create a self-signed certificate by using the **`runmqckm`** tool. After these certificates are created, update the `mqwebuser.xml` file to point to the created keystores. For more information about how to set up TLS connectivity, such as the use of certificate authority signed certificates, see the IBM MQ product documentation.

Example 6-28 Creating a self-signed certificate by using the runmqckm tool

```
# Keystore
runmqckm -keydb -create -db user.p12 -type pkcs12 -pw password -stash
runmqckm -cert -create -db user.p12 -pw password -label default -dn
“CN=International Business Machines,O=IBM,C=US” -size 2048 -expire 365

# truststore
runmqckm -keydb -create -db trust.jks -pw password -type jks -stash
runmqckm -cert -import -db user.p12 -pw password -target trust.jks -target_pw
password -target_type jks
```

Figure 6-3 on page 163 shows our IBM MQ web server running after logging in with our administrative account. We selected **Manage** → **Queues** → **DEV.QUEUE.1** → **Create** to put a message on that queue.


IBM MQ

Home

Manage

Settings

Manage / QM1 /

Local Queue:

0 messages (0%)

Maximum queue depth: 5000

Timestamp Applicati

Add Message

IBM MQ Web Admin/REST API

Delivery mode

Non persistent

Format

MQSTR

Priority

0

Application data

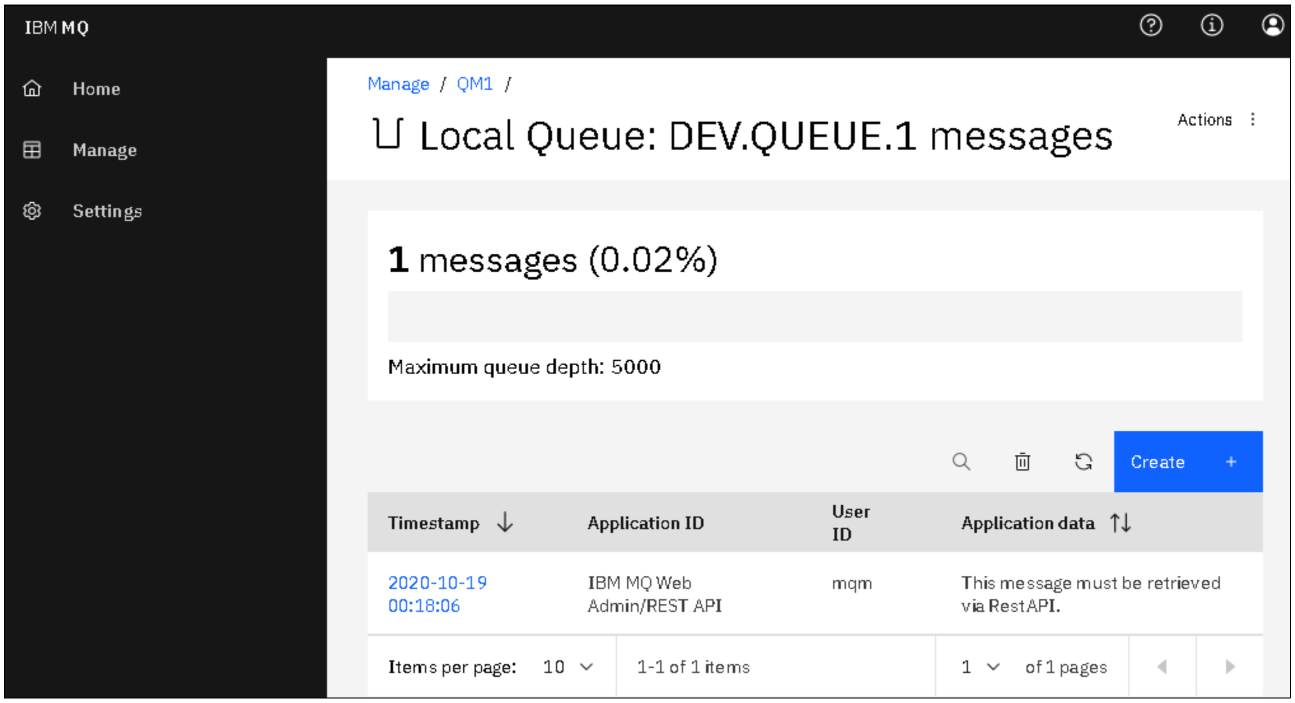
43/1024

This message must be retrieved via RestAPI.

Cancel Create

Figure 6-3 The IBM MQ web server manages and controls the queue manager

Figure 6-4 shows our queue current depth. Our message was successfully PUT as the mqm user ID. Do *not* delete that message now.


IBM MQ

Home

Manage

Settings

Manage / QM1 /

Local Queue: DEV.QUEUE.1 messages

Actions

1 messages (0.02%)

Maximum queue depth: 5000

Create +

Timestamp	Application ID	User ID	Application data
2020-10-19 00:18:06	IBM MQ Web Admin/REST API	mqm	This message must be retrieved via RestAPI.

Items per page: 10 1-1 of 1 items 1 of 1 pages

Figure 6-4 DEV.QUEUE.1 current depth through the IBM MQ web server console

6.4 Migrating a Node.js application

The process of migrating a Node.js program to IBM LinuxONE is straightforward. This component of our application consists of a logging service implementation that retrieves and logs all user access that is done to our main web application for audit purposes. It also performs RESTful API calls to our IBM MQ web server and logs the retrieved information in a hardened RHEL 8.2 server (lnxone-node) that is in a restricted network zone.

RHEL 8 introduced the concept of modules, which groups several packages by function and compatibility. The usage of modules is an elegant way to install different versions of software, depending on the user requirements. The release of Node.js that is enabled by default is Version 10.19. As described in 6.1.2, “Software products and tools checklist” on page 149, our application requires Node.js 12.16.1. Example 6-29 shows how to enable and install the Node.js module in the release that we require.

Example 6-29 Node.js and npm installation

```
lnxone-node # dnf module list nodejs
Updating Subscription Management repositories.
Last metadata expiration check: 0:12:26 ago on Mon 19 Oct 2020 10:39:07 PM EDT.
Red Hat Enterprise Linux 8 - AppStream
Name                Stream               Profiles
Summary
nodejs                          10 [d]                            common [d],
development, minimal, s2i                   JavaScript runtime
nodejs                          12                                common [d],
development, minimal, s2i                   JavaScript runtime

Extra Packages for Enterprise Linux Modular 8 - s390x
Name                Stream               Profiles
Summary
nodejs                          13                                default,
development, minimal                         JavaScript runtime

Hint: [d]efault, [e]nabled, [x]disabled, [i]nstalled
lnxone-node # dnf -y module enable nodejs:12
lnxone-node # dnf -y install nodejs npm
```

Complete the following steps:

1. Copy the necessary application files from the source x86 server to IBM LinuxONE. Next, create the user account that runs the program. For security reasons and unless strictly necessary, always avoid running processes as the root user. Finally, install the program dependencies and run it, as shown in Example 6-30.

Example 6-30 Node.js application execution

```
lnxone-node $ cd /app/app_audit
lnxone-node $ ls
package.json  package-lock.json  program.js

lnxone-node $ npm install
added 2 packages from 4 contributors and audited 2 packages in 5.175s
found 0 vulnerabilities

lnxone-node $ node -v
```

```
v12.16.1

lnxone-node $ node program.js
Retrieved 1 messages from IBM MQ
Output logged at /app/audit.log

lnxone-node $ cat /app/audit.log
This message must be retrieved through the RESTful API.
```

2. The application successfully used the message that we manually PUT, as described in 6.3.1, “Moving the IBM MQ Web Server and the RESTful API component” on page 161. Configure a cron job so that it can now run on its own, as shown in Example 6-31.

Example 6-31 Automated execution of a node.js program

```
lnxone-node $ echo '*/5 * * * * node /app/app_audit/program.js >>
/app/execution.log 2>&1' | crontab -
lnxone-node $ crontab -l
*/5 * * * * node /app/app_audit/program.js >> /app/execution.log 2>&1
```

6.5 Migrating a WebSphere Application Server Network Deployment cluster

IBM WebSphere Application Server Network Deployment is a full-featured middleware that supports various topologies and configurations and can integrate several aspects of your enterprise organization in accordance to your business needs.

Various methods are available to migrate a WebSphere Application Server Network Deployment cluster to IBM LinuxONE. As described in Chapter 4, “Migration process” on page 63, the migration strategy that is used often depends on the application’s architecture complexity and the preference of the key business stakeholders.

A common migration strategy is to install and configure manually (or with the help of automated wsadmin Jacl or Jython scripts) all aspects of a cluster topology. In this scenario, the WebSphere Application Server cluster is re-created from scratch and all required components, such as Java Database Connectivity (JDBC) data sources and JMS resources, are reconfigured. Finally, the application that uses these resources is redeployed in the new cluster and tested. This approach is typically done for simple applications that do not require much effort to be redeployed.

Another migration strategy involves the migration of the entire cell configuration directly from the source x86 servers to the new IBM LinuxONE systems. After the cell configuration is imported, the application is ready for use with minimal manual adjustments necessary. In this section, we migrate our WebSphere Application Server Network Deployment cell configuration by using this migration technique.

The source WebSphere Application Server cell is composed of two servers in a horizontal topology, as described in 6.1, “Environment setup” on page 148. The deployment manager node is in `xrhrbres1`, and this server also holds one node part of our application’s cluster. The second node member of our cluster is in `xrhrbres2`.

Complete the following steps:

1. Back up the current profiles configuration, including the deployment manager and application servers, as shown in Example 6-32.

Example 6-32 Backing up all WebSphere Application Server profiles before proceeding

```
xrhrbres1 $ <DMGR_ROOT>/bin/backupConfig.sh /tmp/DmgrBackupBefore.zip -nostop
ADMU0116I: Tool information is being logged in a file
           /opt/WebSphere/AppServer/profiles/Dmgr/logs/backupConfig.log
ADMU0128I: Starting tool with the Dmgr profile
ADMU5001I: Backing up config directory
           /opt/WebSphere/AppServer/profiles/Dmgr/config to file
           /tmp/DmgrBackupBefore.zip
(...) output suppressed (...)
ADMU5002I: 1,700 files successfully backed up

xrhrbres1 $ <APP_ROOT>/bin/backupConfig.sh /tmp/AppSrvrBackupBefore.zip -nostop
(...) output suppressed (...)

xrhrbres2 $ <APP_ROOT>/bin/backupConfig.sh /tmp/AppSrvrBackupBefore.zip -nostop
(...) output suppressed (...)
```

Because we are migrating our profiles to a different target architecture, we must help ensure that the WebSphere Application Server release is under the same level at the source and target systems. Otherwise, we must first install the same level that we are migrating to into the source x86 system. Because we are performing an "as is" migration, the WebSphere Application Server product is installed into our target LinuxONE servers under the same level as our source x86 systems.

Note: For more information about migrating between different WebSphere Application Server releases, see Migrating cells to new host machines using the command-line tool.

2. Transfer the profile's configuration from the source x86 systems to the target LinuxONE server. Because we are keeping the same topology, we transfer the deployment manager profile and the application server profile from `xrhrbres1` to `lnxone-was-1`. We also copy the application server profile from `xrhrbres2` to `lnxone-was-2`.
3. Create the profiles that compose the cluster topology into the LinuxONE servers. Keep the same node and cell names as we had on the source x86 systems. Example 6-33 shows the Deployment Manager profile creation at `lnxone-was-1`.

Example 6-33 Creating the target deployment manager profile by using the same cell and node names

```
lnxone-was-1 $ <WAS_ROOT>/bin/manageprofiles.sh -create -profileName Dmgr
-profilePath /opt/WebSphere/AppServer/profiles/Dmgr -templatePath
/opt/WebSphere/AppServer/profileTemplates/management -serverType
DEPLOYMENT_MANAGER -nodeName AppCellNode -cellName AppCell -hostName lnxone-was-1
-isDefault=false -enableAdminSecurity=false -disableWASDesktopIntegration

INSTCONFSUCCESS: Success: Profile Dmgr now exists. Consult
/opt/WebSphere/AppServer/profiles/Dmgr/logs/AboutThisProfile.txt for more
information about this profile.
```

4. After the deployment manager profile is created, restore its configuration, as shown in Example 6-34 on page 167.

Example 6-34 Deployment manager restore configuration

```
lnxone-was-1 $ <DMGR_PROFILE_ROOT>/bin/restoreConfig.sh /tmp/DmgrBackupBefore.zip
ADMU0116I: Tool information is being logged in a file
           /opt/WebSphere/AppServer/profiles/Dmgr/logs/restoreConfig.log
ADMU0128I: Starting tool with the Dmgr profile
ADMU0505I: Servers found in configuration:
ADMU0506I: Server name: dmgr
ADMU2010I: Stopping all server processes for node AppCellNode
ADMU0512I: Server dmgr cannot be reached. It appears to be stopped.
ADMU5502I: The directory /opt/WebSphere/AppServer/profiles/Dmgr/config already
           exists; renaming to
           /opt/WebSphere/AppServer/profiles/Dmgr/config.old
ADMU5504I: Restore location successfully renamed
ADMU5505I: Restoring file /tmp/DmgrBackupBefore.zip to location
           /opt/WebSphere/AppServer/profiles/Dmgr/config
ADMU5506I: 1,700 files successfully restored
ADMU6001I: Begin App Preparation -
ADMU6009I: Processing complete.
ADMU6002I: Begin Asset Preparation -
ADMU6009I: Processing complete.
```

5. The restored configuration still contains old references from the x86 servers. Update the internal deployment manager files to point to the new IBM LinuxONE servers. Example 6-35 shows how we update the `serverindex.xml` file to point to our IBM LinuxONE topology. Three files must be updated: two for `lnxone-was-1` and one for `lnxone-was-2`.

Example 6-35 Updating serverindex.xml to reflect the new system names

```
lnxone-was-1 $ find <DMGR_PROFILE_ROOT>/config/cells/ -name serverindex.xml
<DMGR_PROFILE_ROOT>/config/cells/AppCell/nodes/AppCellNode/serverindex.xml
<DMGR_PROFILE_ROOT>/config/cells/AppCell/nodes/AppNode/serverindex.xml
<DMGR_PROFILE_ROOT>/config/cells/AppCell/nodes/AppNode2/serverindex.xml
lnxone-was-1 $ sed -i ‘s/xrhrbres1/lnxone-was-1/g’
<DMGR_PROFILE_ROOT>/config/cells/AppCell/nodes/AppCellNode/serverindex.xml

lnxone-was-1 $ sed -i ‘s/xrhrbres1/lnxone-was-1/g’
<DMGR_PROFILE_ROOT>/config/cells/AppCell/nodes/AppNode/serverindex.xml

lnxone-was-2 $ sed -i ‘s/xrhrbres2/lnxone-was-2/g’
<DMGR_PROFILE_ROOT>/config/cells/AppCell/nodes/AppNode2/serverindex.xml
```

6. Update the x86 architecture references to the one that LinuxONE uses. Example 6-36 provides a one-liner that does the job for us, and checks the correct values to use.

Example 6-36 Updating the node-metadata.properties architecture

```
lnxone-was-1 $ find <DMGR_ROOT>/config/cells/ -name node-metadata.properties -exec
grep x86_64 {} \;
com.ibm.websphere.sdk.architecture.8.0_64=x86_64
com.ibm.websphere.sdk.nativeLibPath.8.0_64=${WAS_INSTALL_ROOT}/lib/native/linux/x8
6_64/
com.ibm.websphere.sdk.nativeLibPath.8.0_64=${WAS_INSTALL_ROOT}/lib/native/linux/x8
6_64/
com.ibm.websphere.sdk.architecture.8.0_64=x86_64
com.ibm.websphere.sdk.nativeLibPath.8.0_64=${WAS_INSTALL_ROOT}/lib/native/linux/x8
6_64/
com.ibm.websphere.sdk.architecture.8.0_64=x86_64

lnxone-was-1 $ find <DMGR_ROOT>/config/cells/ -name node-metadata.properties -exec
sed -i 's/x86_64/s390_64/g' {} \;
```

Note: Now, shut down your x86 cluster to avoid problems. Do *not* proceed if you failed to perform any of the previous steps.

7. Start the deployment manager for the cell, as shown in Example 6-37.

Example 6-37 Deployment Manager start

```
lnxone-was-1 $ <DMGR_ROOT>/bin/startManager.sh
ADMU0116I: Tool information is being logged in a file
           /opt/WebSphere/AppServer/profiles/Dmgr/logs/dmgr/startServer.log
ADMU0128I: Starting tool with the Dmgr profile
ADMU3100I: Reading configuration for server: dmgr
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server dmgr open for e-business; process ID is 272885
```

8. After the deployment manager process starts, create the application server profiles, restore their respective configurations, and synchronize them with the new deployment manager. Example 6-38 shows how this process was done under `lnxone-was-2`. Use the same cell and node names as the source x86 system for the profiles that you create.

Example 6-38 Application server configuration restore and synchronization with the new cell

```
lnxone-was-2 $ <WAS_INSTALL_ROOT>/bin/manageprofiles.sh -create -profileName
AppSrvr -profilePath /opt/WebSphere/AppServer/profiles/AppSrvr -templatePath
/opt/WebSphere/AppServer/profileTemplates/managed -nodeName AppNode2 -cellName
AppCell -hostName lnxone-was-2 -enableAdminSecurity=false -federateLater=true
-disableWASDesktopIntegration
INSTCONFSUCCESS: Success: Profile AppSrvr now exists. Consult
/opt/WebSphere/AppServer/profiles/AppSrvr/logs/AboutThisProfile.txt for more
information about this profile.

lnxone-was-2 $ <PROFILE_ROOT>/bin/restoreConfig.sh /tmp/AppSrvrBackupBefore.zip
ADMU0116I: Tool information is being logged in a file
           /opt/WebSphere/AppServer/profiles/AppSrvr/logs/restoreConfig.log
ADMU0128I: Starting the tool with the AppSrvr profile
ADMU0507I: No servers found in configuration under:

/opt/WebSphere/AppServer/profiles/AppSrvr/config/cells/AppCell/nodes/AppNode2/serv
ers
ADMU2010I: Stopping all server processes for node AppNode2
ADMU5502I: The directory /opt/WebSphere/AppServer/profiles/AppSrvr/config
           exists; renaming to
           /opt/WebSphere/AppServer/profiles/AppSrvr/config.old
ADMU5504I: Restore location successfully renamed
ADMU5505I: Restoring file /tmp/AppSrvrBackupBefore.zip to location
           /opt/WebSphere/AppServer/profiles/AppSrvr/config
ADMU5506I: 2,137 files successfully restored
ADMU6001I: Begin App Preparation -
ADMU6009I: Processing complete.
```

```
ADMU6002I: Begin Asset Preparation -
ADMU6009I: Processing complete.

lnxone-was-2 $ <PROFILE_ROOT>/bin/syncNode.sh lnxone-was-1 8879
ADMU0116I: Tool information is being logged in a file
           /opt/WebSphere/AppServer/profiles/AppSrvr/logs/syncNode.log
ADMU0128I: Starting the tool with the AppSrvr profile
(...) output suppressed (...)
ADMU0401I: Begin syncNode operation for node AppNode2 with Deployment Manager
           lnxone-was-1: 8879
ADMU0016I: Synchronizing configuration between node and cell.
ADMU0402I: The configuration for node AppNode2 has been synchronized with
           Deployment Manager lnxone-was-1: 8879
```

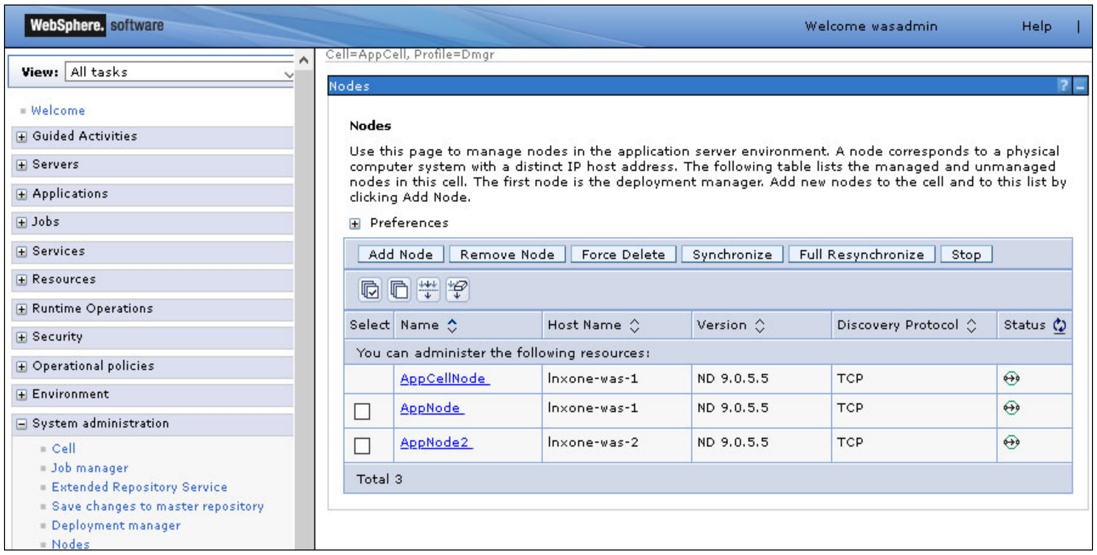
9. Start your application server nodes, as shown in Example 6-39.

Example 6-39 WebSphere Application Server node startup

```
lnxone-was-1 $ <APP_PROFILE_ROOT>/bin/startNode.sh
ADMU0116I: Tool information is being logged in a file
/opt/WebSphere/AppServer/profiles/AppSrvr/logs/nodeagent/startServer.log
ADMU0128I: Starting the tool with the AppSrvr profile
ADMU3100I: Reading configuration for server: nodeagent
ADMU3200I: Server launched. Waiting for initialization status.
ADMU3000I: Server nodeagent open for e-business; process ID is 274616

lnxone-was-2 $ <APP_PROFILE_ROOT>/bin/startNode.sh
(...) output suppressed (...)
```

The migration of WebSphere Application Server is now complete. You can now access the administrative console and check the status of your cluster. The username and password credentials to authenticate typically are the same as in the source x86 system. Figure 6-5 shows our WebSphere Application Server topology that is fully synchronized with our target IBM LinuxONE servers.



Nodes

Use this page to manage nodes in the application server environment. A node corresponds to a physical computer system with a distinct IP host address. The following table lists the managed and unmanaged nodes in this cell. The first node is the deployment manager. Add new nodes to the cell and to this list by clicking Add Node.

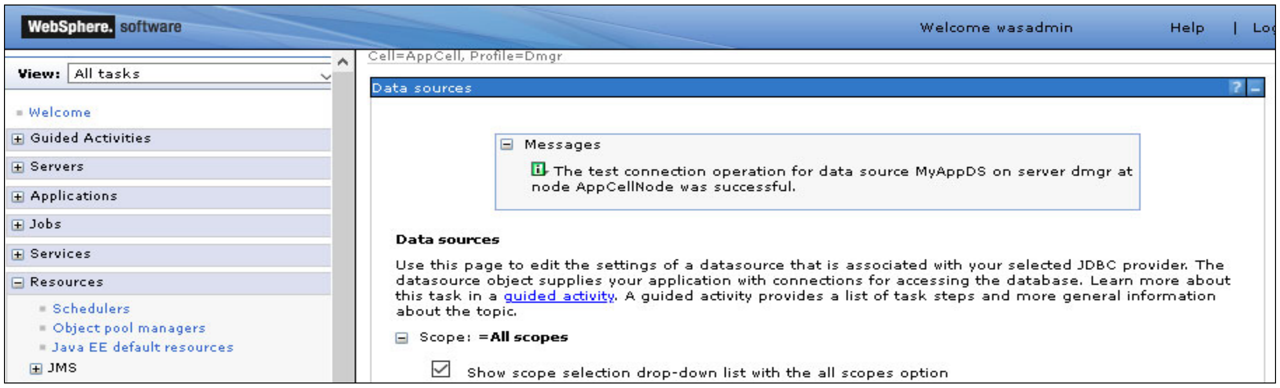
Select	Name	Host Name	Version	Discovery Protocol	Status
You can administer the following resources:					
	AppCellNode	lnxone-was-1	ND 9.0.5.5	TCP	⊕
☐	AppNode	lnxone-was-1	ND 9.0.5.5	TCP	⊕
☐	AppNode2	lnxone-was-2	ND 9.0.5.5	TCP	⊕
Total 3					

Figure 6-5 WebSphere cell migrated into IBM LinuxONE

Note: For enhanced security, modify the credentials to the administrative consoles and update the profile's property files, such as the `soap.client.props` and `ssl.client.props` files. For more information, see your product documentation.

Complete the following steps:

1. Before starting the application, update the data sources to reflect the new Db2 location. Then, test the connection to help ensure that everything was correctly set up. Figure 6-6 shows that the connectivity to the database running under IBM LinuxONE is successful.



Messages

The test connection operation for data source MyAppDS on server dmgr at node AppCellNode was successful.

Data sources

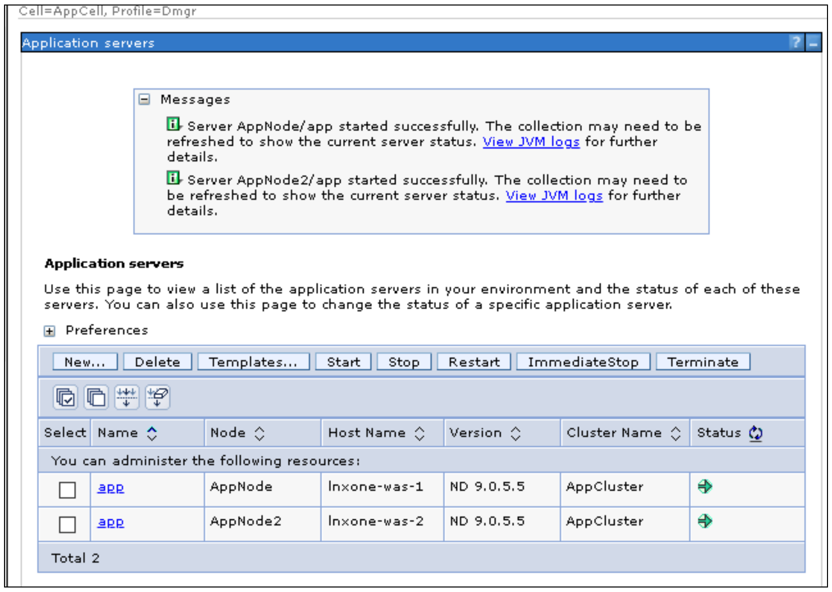
Use this page to edit the settings of a datasource that is associated with your selected JDBC provider. The datasource object supplies your application with connections for accessing the database. Learn more about this task in a guided activity. A guided activity provides a list of task steps and more general information about the topic.

Scope: =**All scopes**

Show scope selection drop-down list with the all scopes option

Figure 6-6 Data source connection test

2. After helping ensure that the application's required components are correctly set up for the newly migrated WebSphere cluster, start the application. Figure 6-7 shows the application servers running after the migration to IBM LinuxONE completed.



Messages

Server AppNode/app started successfully. The collection may need to be refreshed to show the current server status. View JVM logs for further details.

Server AppNode2/app started successfully. The collection may need to be refreshed to show the current server status. View JVM logs for further details.

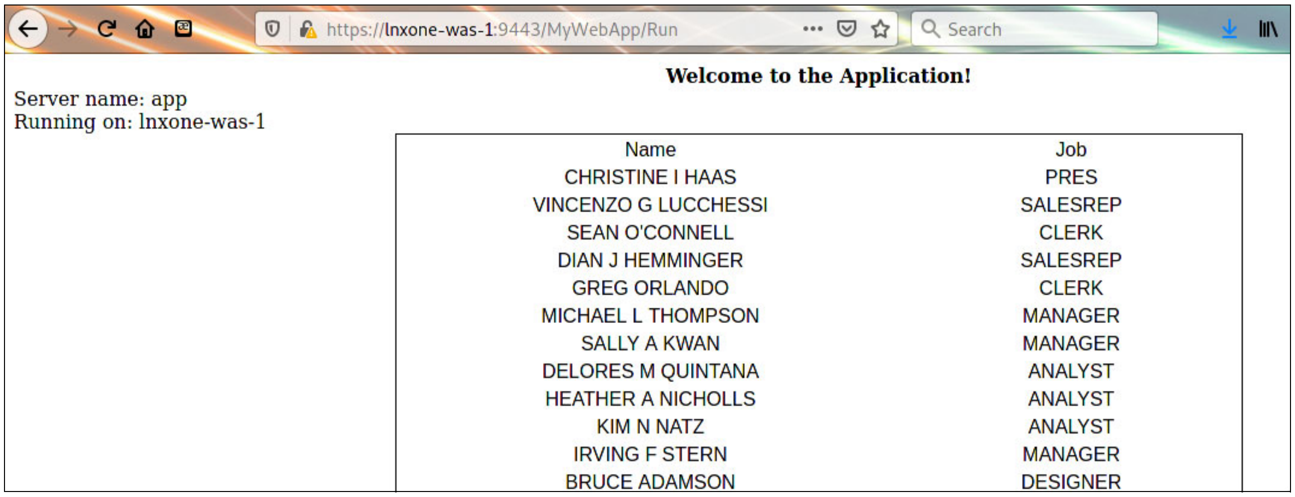
Application servers

Use this page to view a list of the application servers in your environment and the status of each of these servers. You can also use this page to change the status of a specific application server.

Select	Name	Node	Host Name	Version	Cluster Name	Status
You can administer the following resources:						
☐	app	AppNode	lnxone-was-1	ND 9.0.5.5	AppCluster	➔
☐	app	AppNode2	lnxone-was-2	ND 9.0.5.5	AppCluster	➔
Total 2						

Figure 6-7 Application server successfully started

3. Released the application for testing. Example 6-8 on page 155 shows the application running under IBM LinuxONE and pulling data from the Db2 database that we migrated in 6.2, "Migrating Db2 and its data" on page 154.



Welcome to the Application!

Server name: app
Running on: lnxone-was-1

Name	Job
CHRISTINE I HAAS	PRES
VINCENZO G LUCCHESSI	SALESREP
SEAN O'CONNELL	CLERK
DIAN J HEMMINGER	SALESREP
GREG ORLANDO	CLERK
MICHAEL L THOMPSON	MANAGER
SALLY A KWAN	MANAGER
DELORES M QUINTANA	ANALYST
HEATHER A NICHOLLS	ANALYST
KIM N NATZ	ANALYST
IRVING F STERN	MANAGER
BRUCE ADAMSON	DESIGNER

Figure 6-8 Fully migrated infrastructure to IBM LinuxONE

6.6 Application tests

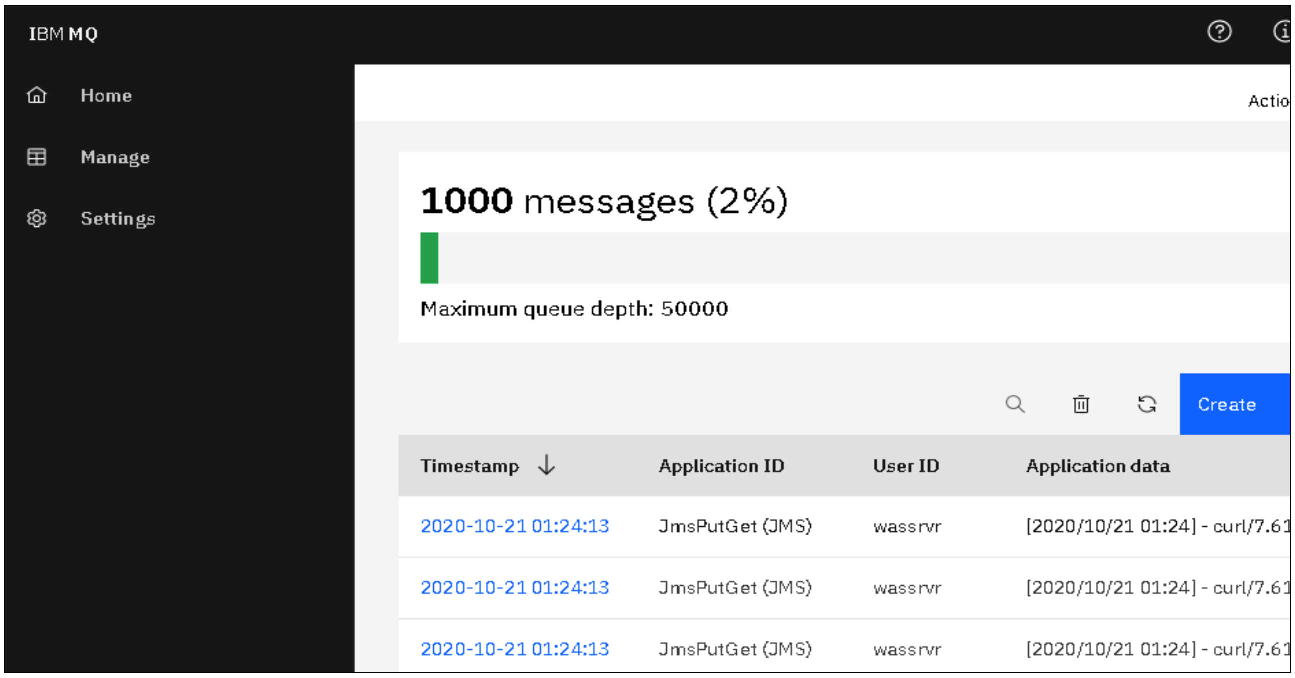
After placing the application behind a load balancer, we tested it. As required by the business, all hits to our web pages must be logged and sent out to our IBM MQ queues, which then are processed and stored by our Node.js application for future audits.

In our testing scenario, we used the **`curl`** tool to send one thousand simultaneous HTTP requests against our application, as shown in Example 6-40.

Example 6-40 One thousand simultaneous accesses to our WebSphere Application Server cluster

```
lnxone-test $ curl for i in {1..1000}; do curl https://was-lb-10/MyWebApp/Run
&>/dev/null & done
```

Then, we checked our IBM MQ queue depth by using its web server console, as shown in Figure 6-9.



1000 messages (2%)

Maximum queue depth: 50000

Timestamp	Application ID	User ID	Application data
2020-10-21 01:24:13	JmsPutGet (JMS)	wassrvr	[2020/10/21 01:24] - curl/7.61
2020-10-21 01:24:13	JmsPutGet (JMS)	wassrvr	[2020/10/21 01:24] - curl/7.61
2020-10-21 01:24:13	JmsPutGet (JMS)	wassrvr	[2020/10/21 01:24] - curl/7.61

Figure 6-9 IBM MQ queue depth shows that our cluster can hold over 1000 simultaneous requests

Finally, we ran our Node.js app and timed the amount of time that it took to process our messages, as shown in Example 6-41.

Example 6-41 Node.js processing time

```
lnxone-node $ time node program.js
Retrieved 1000 messages from IBM MQ
Output logged at /app/audit.log

real 0m25.216s
user 0m1.182s
sys 0m0.142s
```

Our entire workload successfully migrated and integrated to IBM LinuxONE. As is done for other platforms, infrastructure components can be easily moved to LinuxONE, with the benefit of better throughput, availability, security, and response times.

Note: For more information about how to scale up your infrastructure with IBM LinuxONE, see *Scale up for Linux on LinuxONE*, REDP-5540.

6.7 Migrating WebSphere Application Server Liberty and Open Liberty

WebSphere Liberty stands as a leading runtime solution that caters to the creation of cloud-native applications and the modernization of existing workloads. Its lightweight and efficient design makes it suitable for microservices, and its extensive application programming interface (API) support and adaptable architecture position it as an optimal choice for modernization efforts. WebSphere Liberty, including its open-source version Open Liberty®, caters to modern development and operational needs, and serves as an excellent option for updating applications from traditional servers like WebSphere Application Server.

Developers benefit from a streamlined experience that enhances productivity, which offers integrated development and testing capabilities both with and without containers. Its design aligns seamlessly with modern automation practices like continuous integration and continuous deployment (CI/CD).

Operations teams and DevOps practitioners can use the cost-effectiveness of deploying both microservices and monolithic applications with WebSphere Liberty. Its simplified deployment process requires no tuning, and it seamlessly integrates with Kubernetes for deployment and management.

IBM Open Liberty (an IBM Open Source Project) adheres to the Eclipse Public License (EPL) and supports Java Enterprise Edition (EE and Jakarta EE Full Platform 7 and 8, and MicroProfile 1 - 4.1.

6.7.1 Migrating WebSphere Application Server Liberty and Open Liberty

Although traditional migration approaches exist, embracing CI/CD patterns unlocks a more agile and efficient way to move your WebSphere Liberty applications from x86 to LinuxONE. This section explores two potential migration paths:

- WebSphere Application Server Liberty installation and pipeline deployment:
  - On LinuxONE, install the WebSphere Liberty run time.
  - Integrate your application and artifacts into your existing CI/CD pipeline. The pipeline handles the deployment process on the target LinuxONE environment, which streamlines automation and consistency.
- WebSphere Application Server Liberty installation and manual migration:
  - On LinuxONE, install the WebSphere Liberty run time.
  - From the source (x86) environment, copy the application directory and the `server.xml` configuration file to the LinuxONE server.
  - Modify the `server.xml` file to reflect the new LinuxONE server's hostname or IP address.

This section showcases the second alternative for illustrative purposes, and demonstrates a simplified migration approach that uses pre-configured components. However, carefully evaluate both options and choose the one that best aligns with your specific application needs and existing infrastructure.

Regardless of the chosen path, CI/CD integration plays a vital role in streamlining the migration process, helping ensure consistency, and minimizing manual intervention. This approach paves the way for faster deployments, reduced errors, and enhanced agility in your migration journey.

In this scenario, we use the IBM LinuxONE Community Cloud (only for illustration purposes), which offers a no-charge, 24x7, enterprise-grade, open access, and shared public cloud environment built on the IBM LinuxONE III platform. Using this platform, we undertake the migration of an Open Liberty server from an x86 server to LinuxONE by completing the following steps:

1. Sign in to the LinuxONE Community Cloud by using the following link:

   `https://linuxone.cloud.marist.edu/#/login`

   You gain access to the platform, as shown in Figure 6-10.

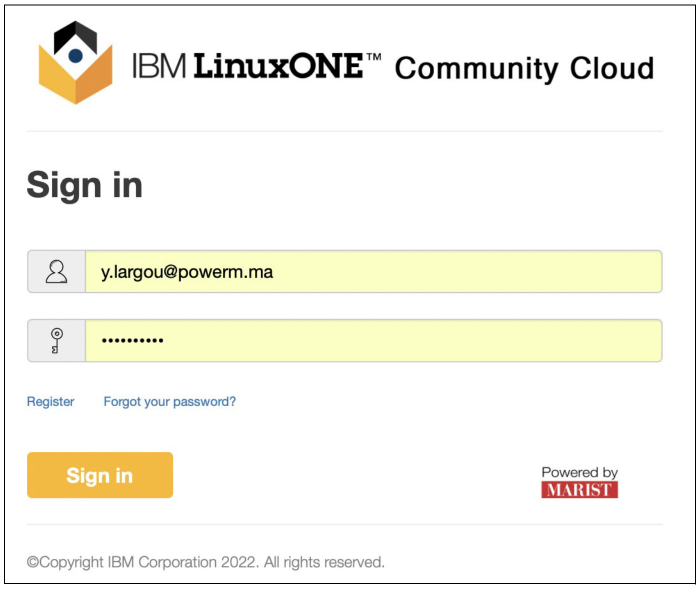


Figure 6-10 Logging in to IBM LinuxONE Community Cloud

2. Once you are logged in, go to the Service Catalog section and click **Manage instances**, where you initiate the process of creating an instance, as shown in Figure 6-11.

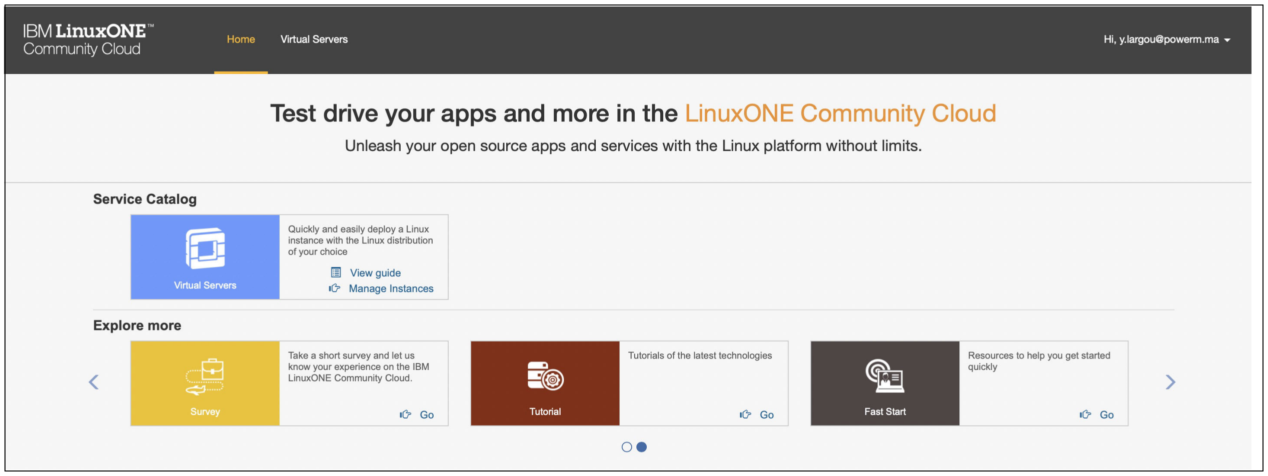
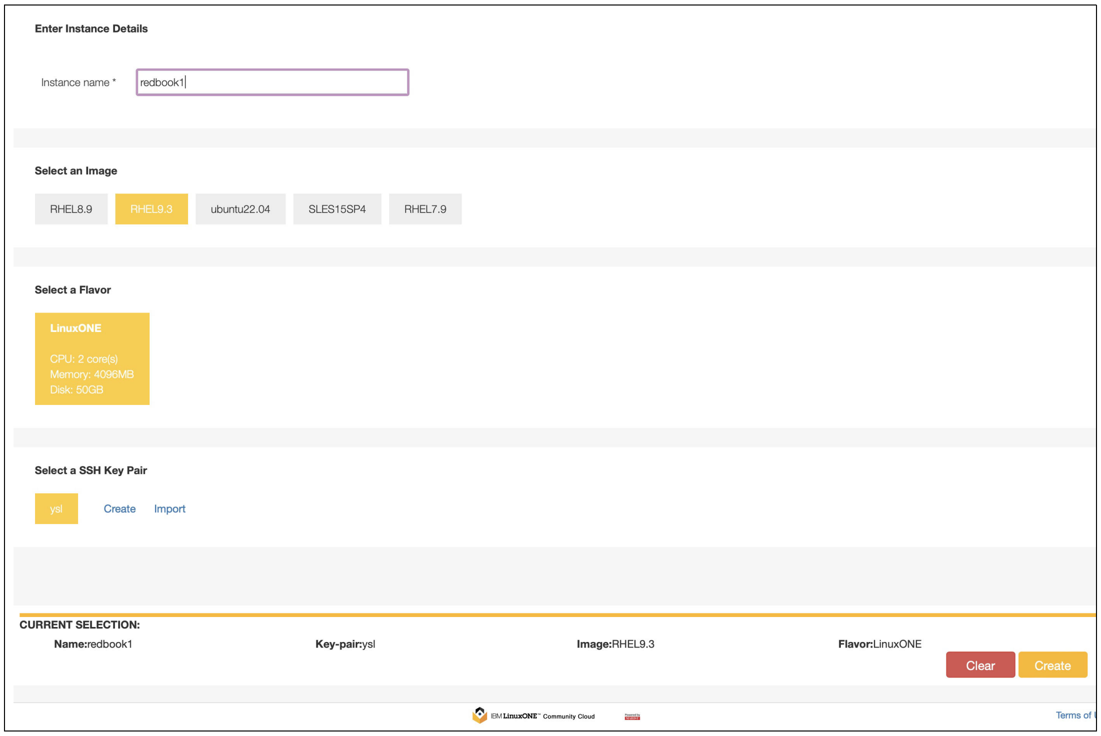

IBM LinuxONE™ Community Cloud
Home
Virtual Servers
Hi, y.largou@powerm.ma
Test drive your apps and more in the LinuxONE Community Cloud
Unleash your open source apps and services with the Linux platform without limits.
Service Catalog
Virtual Servers
Quickly and easily deploy a Linux instance with the Linux distribution of your choice
View guide
Manage Instances
Explore more
Survey
Take a short survey and let us know your experience on the IBM LinuxONE Community Cloud.
Go
Tutorial
Tutorials of the latest technologies
Go
Fast Start
Resources to help you get started quickly
Go

Figure 6-11 LinuxONE Community Cloud Service Catalog

3. Enter the necessary details for the new instance, including its name, preferred image, and type, and select a Secure Shell (SSH) key pair for secure access. After entering the information, click **Create** to proceed, as shown in Figure 6-12.



Enter Instance Details

Instance name * redbook1

Select an Image

Select a Flavor

LinuxONE
CPU: 2 core(s)
Memory: 4096MB
Disk: 50GB

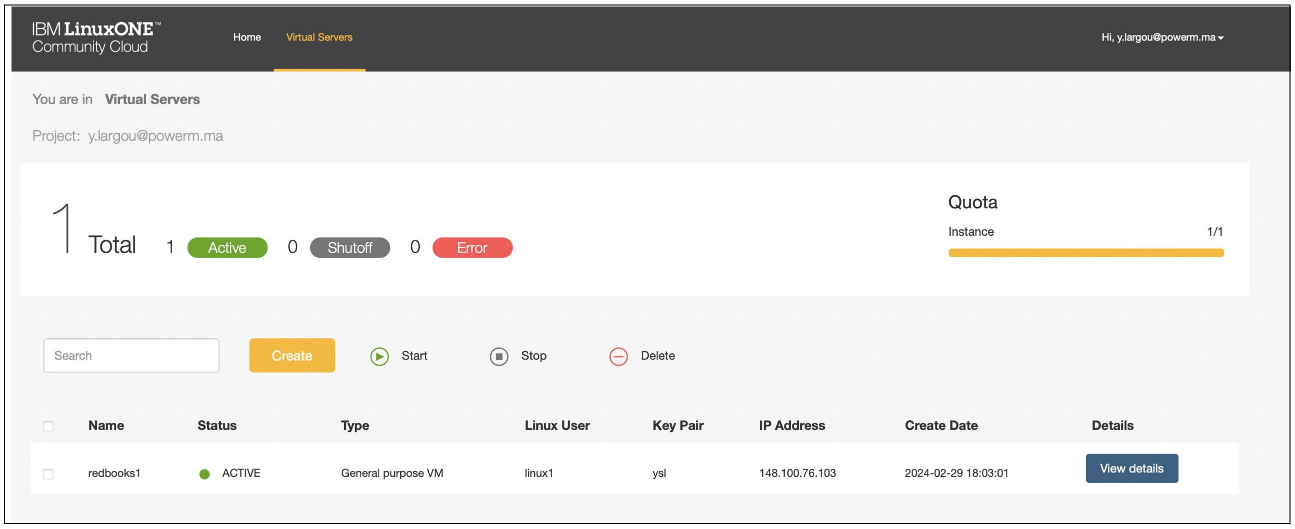
Select a SSH Key Pair

CURRENT SELECTION:

Name:redbook1 **Key-pair:**ysl **Image:**RHEL9.3 **Flavor:**LinuxONE

Figure 6-12 Creating a LinuxONE instance

4. After creation, verify that the instance is successfully provisioned and started. Check the status, which should indicate ACTIVE, which means that the instance is ready for use, as shown in Figure 6-13.



You are in Virtual Servers

Project: y.largou@powerm.ma

1 Total 1 Active 0 Shutoff 0 Error

Quota
Instance 1/1

Name	Status	Type	Linux User	Key Pair	IP Address	Create Date	Details
redbooks1	● ACTIVE	General purpose VM	linux1	ysl	148.100.76.103	2024-02-29 18:03:01	View details

Figure 6-13 Verifying a LinuxONE instance

5. Connect to the LinuxONE instance by using the SSH key, public IP address, and the designated user credentials. This step establishes a secure connection to the newly created instance, as shown in Example 6-42.

Example 6-42 Logging in to a LinuxONE instance

```
mac@Macs-MacBook-Pro-2 yslma % ssh -i ysl.pem linux1@148.100.76.103
 ___ ____  __  __      _     _                  ___  _   _ _____
|_ _| __ )|  \/  |    | |   (_)_ __  _   ___  __/ _ \| \ | | ____|
 | ||  _ \| |\/| |    | |   | | '_ \| | | \ \/ / | | |  \| |  _|
 | || |_) | |  | |    | |___| | | | | |_| |>  <| |_| | |\  | |___
|___|____/|_|  |_|    |_____|_|_| |_|\__,_/_/\_\\___/|_| \_|_____|

=================================================================================
Welcome to the IBM LinuxONE Community Cloud!

This server is for authorized users only. All activity is logged and monitored.
Individuals using this server must abide to the Terms and Conditions listed here:
https://www.ibm.com/community/z/ibm-linuxone-community-cloud-terms-and-conditions/
Your access will be revoked for any non-compliance.
==================================================================================
Last login: Sun Mar  3 11:35:46 2024 from 105.136.51.151
```

6. Once connected, verify the Integrated Facility for Linux (IFL) configuration of the instance by using the **lscpu** command, which provides insights into the CPU architecture and capabilities, as shown in Example 6-43.

Example 6-43 Verifying the LinuxONE CPU configuration

```
[linux1@redbooks1 ~]$ lscpu
Architecture:            s390x
  CPU op-mode(s):        32-bit, 64-bit
  Byte Order:            Big Endian
CPUs:                  2
  Online CPUs list:   0,1
Vendor ID:               IBM/S390
  Machine type:          8561
  Threads per core:    1
  Cores per socket:    1
  Sockets per book:    1
  Books per drawer:    1
  Drawers:             2
  CPU dynamic MHz:       5200
  CPU static MHz:        5200
  BogoMIPS:              3241.00
  Dispatching mode:      horizontal
  Flags:                esan3 zarch stfle msa ldisp eimm dfp edat etf3eh highgprs
te vx vxd vxe gs vxe2 vxp sort dflt sie
Virtualization features:
  Hypervisor:            z/VM 7.2.0
  Hypervisor vendor:     IBM
  Virtualization type:   full
Caches (sum of all):
  L1d:                   256 KiB (2 instances)
  L1i:                   256 KiB (2 instances)
  L2d:                   4 MiB (1 instance)
  L2i:                   8 MiB (2 instances)
  L3:                       256 MiB
  L4:                       960 MiB
NUMA:
  NUMA nodes:             1
  NUMA node0 CPUs:        0,1
Vulnerabilities:
  Gather data sampling:   Not affected
  Itlb multihit:          Not affected
  L1tf:                   Not affected
  Mds:                    Not affected
  Meltdown:               Not affected
  Mmio stale data:        Not affected
  Retbleed:               Not affected
  Spec store bypass:      Not affected
  Spectre v1:             Mitigation; __user pointer sanitization
  Spectre v2:             Mitigation; etokens
  Srbds:                  Not affected
  Tsx async abort:        Not affected
[linux1@redbooks1 ~]$
```

7. Download and install the official release of IBM Semeru Runtime Open Edition for Java 21 onto the LinuxONE instance, as shown in Example 6-44. Help ensure that you are using the same version as the source server (x86).

Example 6-44 Installing IBM Semeru Runtime Open Edition for Java

```
[root@redbooks1 ~]# rpm -ivh
https://github.com/ibmruntimes/semeru21-binaries/releases/download/jdk-21.0.2%2B13
_openj9-0.43.0/ibm-semeru-open-21-jdk-21.0.2.13_0.43.0-1.s390x.rpm
```

8. Set IBM Semeru Runtime as the default Java runtime environment on LinuxONE, as shown in Example 6-45.

Example 6-45 Setting the default Java run time

```
[root@redbooks1 ~]# update-alternatives --config java

There are 2 programs that provide 'java'.

  Selection    Command
-----------------------------------------------
*+ 1           java-11-openjdk.s390x
(/usr/lib/jvm/java-11-openjdk-11.0.22.0.7-2.el9.s390x/bin/java)
   2           /usr/lib/jvm/ibm-semeru-open-21-jdk/bin/java

Enter to keep the current selection[+], or type selection number: 2
```

9. Verify the Java version, as shown in Example 6-46.

Example 6-46 Verifying the Java version

```
[root@redbooks1 ~]# java -version
openjdk version "21.0.2" 2024-01-16 LTS
IBM Semeru Runtime Open Edition 21.0.2.0 (build 21.0.2+13-LTS)
Eclipse OpenJ9 VM 21.0.2.0 (build openj9-0.43.0, JRE 21 Linux s390x-64-Bit
Compressed References 20240116_94 (JIT enabled, AOT enabled)
OpenJ9   - 2c3d78b48
OMR      - ea8124dbc
JCL      - 78c4500a434 based on jdk-21.0.2+13)
[root@redbooks1 ~]#
```

10.Download Open Liberty 24.0.0.2, as shown in Example 6-47. Help ensure that it matches the version that is used on the source server.

Example 6-47 Downloading the Open Liberty packages

```
[root@redbooks1 ~]# wget
https://public.dhe.ibm.com/ibmdl/export/pub/software/openliberty/runtime/release/2
4.0.0.2/openliberty-jakartaee10-24.0.0.2.zip
--2024-03-03 14:12:11--
https://public.dhe.ibm.com/ibmdl/export/pub/software/openliberty/runtime/release/2
4.0.0.2/openliberty-jakartaee10-24.0.0.2.zip
Resolving public.dhe.ibm.com (public.dhe.ibm.com)... 170.225.126.18
Connecting to public.dhe.ibm.com (public.dhe.ibm.com)|170.225.126.18|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 133034245 (127M) [application/zip]
Saving to: ‘openliberty-jakartaee10-24.0.0.2.zip’

openliberty-jakartaee10-24.0.0.2.zip.2
100%[=============================================================================
=================================================================>] 126.87M
4.58MB/s    in 19s

2024-03-03 14:12:30 (6.83 MB/s) - ‘openliberty-jakartaee10-24.0.0.2.zip’ saved
[133034245/133034245]
```

11.Decompress the installation package archive, as shown in Example 6-48.

Example 6-48 Decompressing the installation archive

```
[root@redbooks1 a]# unzip openliberty-jakartaee10-24.0.0.2.zip
```

12.With the necessary components installed and configured, create and start a server Open Liberty instance on the LinuxONE environment, as shown in Example 6-49.

Example 6-49 Creating and starting an Open Liberty instance

```
[root@redbooks1 bin]# /root/wlp/bin/server create redbookserver1

Server redbookserver1 created.
[root@redbooks1 bin]# /root/wlp/bin/server start redbookserver1

Starting server redbookserver1.
```

13.To illustrate the migration process, we created a simple application that displays a message and shows the server CPU architecture, as shown in Example 6-50.

Example 6-50 Simple web application

```
@Path("properties")
public class HelloWorld {

    @GET
    @Produces(MediaType.APPLICATION_JSON)
    public String getProperties() {
        String message = "Hello IBM Residency Team - ZS-EP01 Practical migration 
from x86 to LinuxONE. The application is currently running from : "
                + System.getProperty("os.arch");
        return message;
    }
}
```

14.To test the application on the source serve (x86), use the **curl** command, as shown in Example 6-51.

Example 6-51 Testing the application from the source server

```
[root@yslserver bin]#curl http://yslserver:9080/LibertyProject/system/properties
Hello IBM Residency Team - ZS-EP01 Practical migration from x86 to LinuxONE. The 
application is currently running from : aarch64
```

15.Place the application in the `/apps` directory and modify the `server.xml` file to reflect the application's path and context, as shown in Example 6-52.

Example 6-52 The server.xml file on the source server (x86)

```
<?xml version="1.0" encoding="UTF-8"?>
<server description="new server">

    <!-- Enable features -->
    <featureManager>
        <feature>jakartaee-10.0</feature>
        <feature>microProfile-6.1</feature>
        <feature>restfulWS-3.1</feature>
        <feature>jsonb-3.0</feature>
    </featureManager>

    <!-- This template enables security. To get the full use of all the 
capabilities, a keystore and user registry are required. -->

    <!-- For the keystore, default keys are generated and stored in a keystore. To 
provide the keystore password, generate an
        encoded password by using bin/securityUtility encode and add it below in 
the password attribute of the keyStore element.
        Then uncomment the keyStore element. -->
    <!--
    <keyStore password=""/>
    -->

    <!--For a user registry configuration, configure your user registry. For 
example, configure a basic user registry by using the
        basicRegistry element. Specify your own username below in the name 
attribute of the user element. For the password,
        generate an encoded password by using bin/securityUtility encode and add 
it in the password attribute of the user element.
        Then uncomment the user element. -->
    <basicRegistry id="basic" realm="BasicRealm">
        <!--
        <username="yourUserName" password="" />
        -->
    </basicRegistry>
    <!-- To access this server from a remote client add a host attribute to the
following element, for example host="*" -->
    <httpEndpoint id="defaultHttpEndpoint"
                  host="yslserver"
                  httpPort="9080"
                  httpsPort="9443" />

    <!-- Automatically expand WAR files and EAR files -->
    <applicationManager autoExpand="true"/>

    <!-- Configures the application on a specified context root -->
    <webApplication contextRoot="${app.context.root}" location="demoapp.war" />

    <!-- Default SSL configuration enables trust for default certificates from the
Java runtime -->
    <ssl id="defaultSSLConfig" trustDefaultCerts="true" />

</server>
~
```

16.Copy the `server.xml` file and the `/apps` directory from the source server to LinuxONE. `yslserver` represents the Open Liberty server on the source (x86), and `redbookserver1` refers to the Open Liberty server on the target (LinuxONE), as shown in Example 6-53.

Example 6-53 Copying the server.xml file and the /apps directory from source to target

```
[root@yslserver bin]# scp -i ysl.pem /root/wlp/usr/servers/yslsource/server.xml
linux1@148.100.76.103:/root/wlp/usr/servers/redbookserver1

[root@yslserver bin]# scp -i ysl.pem -r /root/wlp/usr/servers/yslsource/apps/
linux1@148.100.76.103:/root/wlp/usr/servers/redbookserver1
```

17.Modify the `server.xml` file and update the hostname, as shown in Example 6-54.

Example 6-54 The server.xml file on the target server (LinuxONE)

```
[root@redbooks1 bin]# scp -i ysl.pem /root/wlp/usr/servers/yslsource/server.xml
<?xml version="1.0" encoding="UTF-8"?>
<server description="new server">

    <!-- Enable features -->
    <featureManager>
        <feature>jakartaee-10.0</feature>
        <feature>microProfile-6.1</feature>
        <feature>restfulWS-3.1</feature>
        <feature>jsonb-3.0</feature>
    </featureManager>

    <!-- This template enables security. To get the full use of all the
capabilities, a keystore and user registry are required. -->

    <!-- For the keystore, default keys are generated and stored in a keystore. To
provide the keystore password, generate an
        encoded password by using bin/securityUtility encode and add it below in
the password attribute of the keyStore element.
        Then uncomment the keyStore element. -->
    <!--
    <keyStore password=""/>
    -->

    <!--For a user registry configuration, configure your user registry. For
example, configure a basic user registry by using the
        basicRegistry element. Specify your own username below in the name
attribute of the user element. For the password,
        generate an encoded password by using bin/securityUtility encode and add
it in the password attribute of the user element.
        Then uncomment the user element. -->
    <basicRegistry id="basic" realm="BasicRealm">
        <!--
        <username="yourUserName" password="" />
        -->
    </basicRegistry>

    <!-- To access this server from a remote client add a host attribute to the
following element, for example host="*" -->
    <httpEndpoint id="defaultHttpEndpoint"
                  host="“ redbooks1"
                  httpPort="9080"
                  httpsPort="9443" />

    <!-- Automatically expand WAR files and EAR files -->
    <applicationManager autoExpand="true"/>

    <!-- Configures the application on a specified context root -->
    <webApplication contextRoot="${app.context.root}" location="demoapp.war" />

    <!-- Default SSL configuration enables trust for default certificates from the
Java runtime -->
    <ssl id="defaultSSLConfig" trustDefaultCerts="true" />

</server>
```

18. Verify that the application is correctly installed and functioning as expected on the LinuxONE platform, as shown in Example 6-55.

Example 6-55 Testing the application from the target server

```
[root@redbooks1 bin]# curl http://redbooks1:9080/LibertyProject/system/properties
Hello IBM Residency Team - ZS-EP01 Practical migration from x86 to LinuxONE. The
application is currently running from : s390x
```

Note: You can also the directory by dropping it from the source to the target. By default, the `dropins` directory is automatically monitored. If you drop an application into this directory, the application is automatically deployed on the server.

7 Post migration considerations

This chapter describes general post migration consideration concepts for getting acceptance, measuring performance, and tuning. Topics that are covered in this chapter include an acceptance list, performance measurement understanding, and key considerations for performance tuning.

Every migration poses a large challenge for IT organizations because each stakeholder has different expectations and requirements from the project. Most of the topics after migration center around *performance* and *functions*. IT organizations face the following difficult questions:

- What exactly has been done?
- Is there anything missing?
- Is everything working?
- Is the performance as expected?
- Is the process completed?
- Did we get approvals?

To answer these questions, you must take some important steps before and after the migration implementation phase.

This chapter includes the following sections:

- 7.1, “Gaining acceptance” on page 184
- 7.2, “Performance measurement” on page 184
- 7.3, “Performance tuning” on page 186

7.1 Gaining acceptance

Migration projects are recognized as major changes to the IT environment. Each change requires significant testing and acceptance by various stakeholders. These stakeholders must decide whether the migration was a success.

Acceptance requires an understanding of the full picture before and after migration:

- Before the implementation phase starts, complete these tasks:
  - Decide and document test scope.
  - Decide and document a test case (including test scenario).
  - Create post migration checklists for all components.
  - Collect performance data about the system.
  - Get acceptance from the stakeholders for testing.
- After the implementation is done, complete these tasks:
  - Use the post-migration checklists and check whether the implementation is complete.
  - Test the system by using documented test cases (complete and document all test scenarios).
  - Measure performance and compare it with the previous performance data.
  - If necessary, perform performance tuning.

Based on project scope and context, items that are used for acceptance testing can differ, but the following list is the most common acceptance tests that are performed before gaining stakeholder acceptance:

Application testing
Usability testing
Functional testing
Performance testing
Security testing
User acceptance testing

7.2 Performance measurement

This section describes performance measurement and its impact on the success of your migration. The most important point to consider is that you need to measure the performance of the application when it is running in production on the source environment. Compare that data with the performance of the application on the target environment.

This section also covers monitoring commands and tools that can assist you in identifying and resolving performance inhibitors.

7.2.1 What is performance

“Performance” in computer systems is a relative term. Usually computer performance is described in measurable terms, such as transactions per second, response time, or time to process a specific task. However, when a migration project is undertaken, it is important to understand the performance metrics that are used on the source environment so that you can understand the relative performance of the target system.

The initial performance of a new system is often not as expected, especially when changing hardware platforms. Therefore, tuning must be undertaken to improve the performance of the target system. Without having proper metrics, it is impossible to validate the performance of the new platform relative to the former platform. For this reason, the migration project team first needs to agree on what performance metrics from the source platform will be used in the migration project plan to measure the performance of the target platform.

7.2.2 Choosing what to measure

To determine the success of a migration, simply having the application on the target platform provide the same answers as the source platform does not prove success. The natural expectation of a migration onto IBM LinuxONE is that the application will be more resilient and available and provide equal or better performance than the source platform. To help ensure that the performance improvements are easy to show, it is important to choose the correct metrics. But what are these metrics, and how should they be measured?

Response time

Response time is a measure of the time that it takes for something to happen in a computer system. Generally, the response time of a unit of work that is called a *transaction* is measured. This transaction can entail something as simple as checking an account balance, or something as complex as the time taken to issue a new insurance policy or open a new bank account.

The point to remember with computer systems is that the response time of a single transaction is the sum of several response times. Figure 7-1 shows the various components that make up user response time.

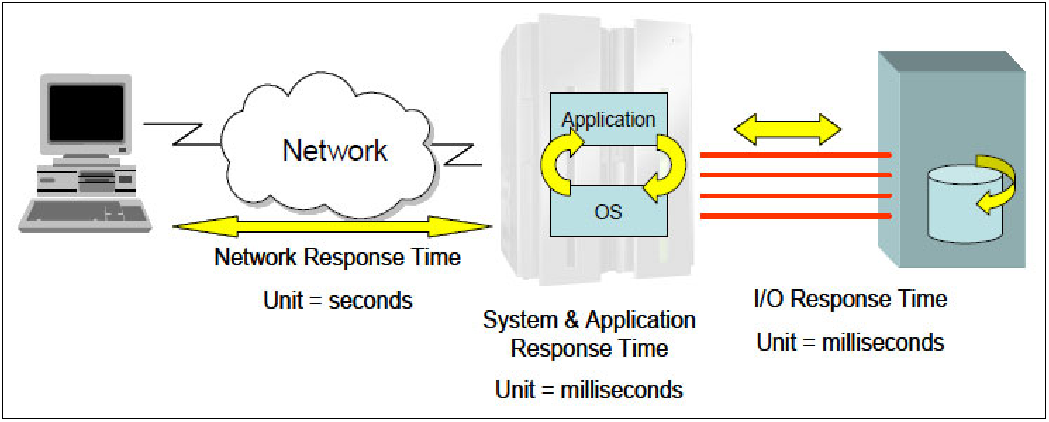


Figure 7-1 Components that make up the response time of transactions

Figure 7-1 on page 185 shows that there are two points where response time can be measured: System response time and user response time. When you are trying to understand the relative performance improvement from a new system, the only point to measure response time is from when a system receives the request and when it provides a response of some sort to the request.

In the case that is shown in Figure 7-1 on page 185, the system response time includes application time and the input/output (I/O) response time to access the data. If you choose to measure the response time of user experiences at their terminal or over the web, you add the network response time, which can vary greatly for the same transaction because it can be influenced by network load.

To compare the source and target systems directly, measure system response time on the source system, and assuming that the application has not changed greatly, measure the system response time on the target platform.

Transaction throughput

The transaction throughput performance metric might provide a more meaningful measure of system performance because it measures the number of transactions that are processed over a period of time. This period is typically one second, but can be any period that you prefer.

In both cases, you have baseline performance metrics for the source system to properly compare the old and new systems.

7.3 Performance tuning

Tuning any system follows some basic principles because every hardware and software platform has unique features and characteristics that must be considered when you tune your environment. The art of performance tuning a system requires a strict combination of performance analyses, multi-step tuning processes, and change management.

Regardless of which tools you choose, the best methodology for analyzing the performance of a system is to start from the outside and work down to the small tuning details in the system. Gather data about the overall health of the system hardware and processes. The following list is a sampling of the types of questions to answer about both your source and target systems:

- How busy is the processor during the peak periods of each day?
- What happens to I/O response times during those peaks?
- Do peaks remain fairly consistent, or do they elongate?
- Does the system get memory constrained every day, causing page waits?
- Can current system resources provide user response times that meet service-level agreements (SLAs)?

It is important to know what tuning tools are available and what type of information they provide. Equally important is knowing when to use those tools and what to look for. How can you know what is normal for your environment and what is problematic unless you check the system activity and resource utilization regularly? Conducting regular health checks on a system also provides utilization and performance information that you can use for capacity planning.

Tuning is not a one-size-fits-all approach. A system that is tuned for one type of workload often performs poorly with another type of workload. This consideration means that you must understand the workload that you want to run and be prepared to review your tuning efforts when the workload changes.

A simple workload is a server that shows one or more peaks during the day. A complicated workload is an application that is CPU-intensive during part of the day and I/O-intensive during another part. The most cost-efficient approach to running these workloads is to adjust the capacity of the server during the day. This is exactly what a hypervisor does. Portions of the virtual machine (VM) are brought in to run in main memory while inactive VMs are moved to paging to create space.

Multi-step tuning process requires the skills of a systems performance detective. A systems performance analyst identifies IT problems by using a detection process similar to that of solving a crime. In IT systems performance, the crime is a performance bottleneck or suddenly degrading response time. The performance analyst asks questions, searches for clues, researches sources and documents, formulates a hypothesis, tests that hypothesis by tuning or other means, and eventually solves the mystery. This process results in improved system performance. Bottleneck analysis and problem determination are facilitated by sophisticated tools such as IBM Tivoli OMEGAMON® XE on z/VM and Linux. These tools detect performance problems and alert a system administrator before degraded response time becomes evident.

Change management that is not strictly related to performance tuning is probably the single most important factor for successful performance tuning. The following considerations highlight this point:

- Implement a proper change management process before tuning any system.
- Never start tweaking settings on a production system.
- Never change more than one variable at a time during the tuning process.
- Retest parameters that supposedly improve performance. Sometimes statistics come into play.
- Document successful parameters and share them with the community no matter how trivial you think they are. System performance can benefit greatly from any results that are obtained in various production environments.

Part 3

Deployment

This part describes deploying workloads and various applications to help you during your deployment.

This part includes Chapter 8, “Deployment of workloads” on page 191.

8

Deployment of workloads

This chapter describes how to deploy workloads to LinuxONE. There are many things to analyze and consider leading up to the deployment of workloads to the platform. When the planning is completed, the migration should move smoothly.

As described in 5.3, “Application analysis” on page 97, many workloads are a “good fit” on LinuxONE. Not all can be demonstrated in this book. The migration of some practical applications, such as IBM Db2, is shown as a hands-on exercise in Chapter 6, “Hands-on migration” on page 147.

Mission-critical applications, ERP, CRM, business intelligence, and more, are good to run on LinuxONE, but only generic examples can be included in a guide such as this. Your specific migration does not necessarily distill into a demonstration. Following the guides, the checklists, and the information in this book, and by using this chapter of examples, will lead you to success.

Standard infrastructure applications are also well suited to the IBM LinuxONE, and they are as critical. In this chapter, the deployment of some standard services is demonstrated. Such an illustration of deploying standard services should likewise represent a pattern that can be followed.

This chapter provides examples of migrating workloads by using example applications, such as MongoDB, MediaWiki, Elasticsearch, Logstash, and Kibana (ELK), MariaDB, and Terraform. It provides an example of migrating OpenLDAP, a central log server, and a Samba file share service.

This chapter includes the following sections:

8.1 Deciding between containers and VMs

The LinuxONE Platform is equipped with some of the fastest general-purpose processors in the world, ideally suited for data processing throughput. The large number of cores available in LinuxONE systems and their high input/output (I/O) bandwidth mean that open source solutions can scale up and scale out.

Although the underlying hardware is ready for a highly scalable environment, advantages and disadvantages exist that are specific to having the solution on a container or virtual machine (VM). With containers, you can have many more applications in a single physical server than a VM can. However, a business might need application deployments that are based on VMs. All aspects of the enterprise application must be considered before deciding whether to run it under containers or in a single VM.

The following are the deciding factors for determining whether the solution should be on containers or VMs:

- Application packaging: If you want to run multiple copies of a single app, say MongoDB, use containers. However, if you want the flexibility of running multiple applications (MongoDB with a Java based Homegrown Application), use a VM.
- Dependencies: Usually, containers tend to lock in to a particular version of an operating system and its subsystems and libraries. This feature can be an advantage for an administrator because with containers you can create a portable, consistent operating environment including programs and libraries for development, testing, and deployment. From a VM perspective, no matter what hypervisor you use, you can deploy any operating environment. This feature is especially useful with in-house applications with specific dependencies.
- Resources: From a resource perspective, containers share an operating system, kernel instance, network connection, and base file system. Each instance of the application runs within a separate user space. This configuration significantly cuts back on the CPU usage that is associated with running multiple operating systems because a new kernel is not needed for each user session. This is one of the major reasons why containers are often used for running specific applications.
- Automation: Concerning speed to production, with the advent of the cloud and DevOps mode of application development, containers have an advantage because each container provides a microservice and can be part of a larger solution. This feature provides containers with the advantage of scale over the VM.
- Security: Without any alterations to the container, a VM is more secure than a container. VMs have the advantage of featuring hardware isolation, but containers share kernel resources and application libraries. This feature means that if a VM breaks down, it is less likely to affect other VMs in the same operating environment. For now, regular containers do not have hardware isolation. If your organization has high security requirements, stick with VMs.

Table 8-1 on page 193 shows some aspects to consider when deciding between containers and VMs.

Table 8-1 Containers versus VMs

Aspect	Containers	Virtual machines
Isolation	Lightweight, OS-level isolation.	Heavyweight, hardware-level isolation.
Resource usage	Shares the host OS kernel with less overhead.	Requires a separate OS with more resource overhead.
Performance	Faster startup and execution.	Slower startup and execution due to overhead.
Portability[a]	Easily portable between environments.	Less portable, and tied to a specific hardware or OS.
Flexibility	Can run multiple containers on a host.	Each VM runs a separate OS, and therefore is less flexible.
Resource sharing	Shares host resources efficiently.	Resources are allocated to each VM independently.
Security	Potential for container escape exploits.	Stronger isolation and less susceptible to attacks.
Scaling	Easier to scale horizontally.	Scalability that is limited by the VM infrastructure.

a. It is crucial to construct multi-architecture containers to help ensure portability across diverse computing environments (x64, x86, S/390, ppc64, arm64 ppc64le, and others).

These examples illustrate how containers and VMs cater to different types of workloads based on factors such as application architecture, resource requirements, security considerations, and compatibility needs:

- Containers:
  - Web servers: Containers are ideal for deploying web server applications like Apache HTTP Server or Nginx. Each container can encapsulate the web server along with its dependencies, making it simple to deploy and scale.
  - Microservices: Containerization is well suited for microservices architectures where applications are broken down into smaller, decoupled services. Each microservice can run in its own container, which enables simpler development, deployment, and scaling.
  - CI/CD pipelines: Continuous Integration and Continuous Deployment (CI/CD) pipelines often use containerized build environments. Containers provide consistency across different stages of the pipeline and enable developers to package their application along with its dependencies.
  - Dev and test environments: Containers are valuable for creating lightweight, isolated development and testing environments. Developers can quickly start containers with the necessary dependencies to test their code, which leads to faster development cycles.
  - Stateful applications: Although traditionally considered more challenging for containers, modern container orchestrators like Red Hat OpenShift support stateful workloads. Stateful applications like databases or key-value stores can be containerized for simpler management and scalability.

- VMs:
  - Legacy applications: VMs are suitable for running legacy applications that might have complex dependencies or compatibility requirements. VMs provide a more traditional environment similar to physical hardware, enabling legacy applications to run without modification.
  - Resource-intensive workloads: Workloads that require dedicated access to hardware resources, such as high-performance computing (HPC) applications, might benefit from running in VMs. VMs can be provisioned with specific CPU, memory, and storage resources to meet the workload's demands.
  - Security-sensitive workloads: VMs provide stronger isolation between workloads compared to containers. Security-sensitive applications or workloads that require strict isolation boundaries can be deployed in separate VMs to minimize the risk of security breaches or data leaks.
  - Workloads requiring different operating systems: VMs support running multiple operating systems concurrently on the same physical hardware. Workloads that require different operating systems or versions can be deployed in separate VMs, which provide flexibility in the choice of operating environment.

In real-world scenarios, it is common to have workloads where certain components are better suited to run in containers, and others are best deployed in VMs.

Consider a web application that consists of a front end (for example, React.js) that is built by using a modern microservices architecture (for example, Spring Boot) and a legacy back-end database (for example, an Oracle database). In this scenario, the front-end microservices can be containerized and deployed by using container orchestration platforms like Kubernetes. Containerization offers agility, scalability, and ease of deployment for the front-end components.

However, the legacy back-end database, which requires strict data consistency and stability, might be better suited to run in a VM. Hosting the database in a VM helps ensure strong isolation, dedicated access to resources, and compatibility with legacy software dependencies.

Most organizations run a mixture of both containers and VMs in their clouds and data centers. The economy of containers at scale makes too much financial sense for anyone to ignore. However, VMs still have their virtues and use cases. LinuxONE provides the best in class features for running containers and VMs.

Benefits of LinuxONE for mixed containers and VMs

IBM provides a secure platform for containers on IBM LinuxONE 4 hardware that is called IBM Secure Service Containers (SSCs). This technology enables mission-critical applications to be securely built, deployed, and managed in hybrid multicloud environments. For more information about IBM SSC, see IBM Secure Service Container for IBM Cloud Private 1.1.0.

LinuxONE security features, such as pervasive encryption and secure boot, provide robust protection for containerized workloads. Each container benefits from hardware-based encryption and isolation, safeguarding sensitive data and preventing unauthorized access.

LinuxONE advanced processor architecture and virtualization technology deliver exceptional performance for virtualized workloads. VMs running on LinuxONE benefit from high-speed I/O, low-latency access to shared resources, and hardware acceleration features, which help ensure optimal performance levels. LinuxONE also helps ensure that containerized workloads achieve excellent performance levels. The platform's scalability and advanced workload management features enable containers to start almost instantly and scale seamlessly to meet changing demands.

LinuxONE offers flexibility in deploying and managing VMs through a range of virtualization solutions, including IBM z/VM and Kernel-based Virtual Machine (KVM). It also supports a wide range of container orchestration tools, and management frameworks that offer flexibility in managing containerized environments. You can choose the tools that best fit your requirement, whether you use Kubernetes, Red Hat OpenShift, or other container platforms.

8.2 Setting up Docker

The Docker engine is available for installation on SUSE and Ubuntu distributions. As of Red Hat 8, support was substituted by the Red Hat OpenShift container management platform. In this section, we describe each distribution, while discussing the container landscape on Red Hat.

8.2.1 Containers on Red Hat

With the changes to how containerization is used in the enterprise computing setting, Red Hat dropped support for Docker-related tools and focused on its product Red Hat OpenShift. The Red Hat OpenShift platform provides Docker functions without exposing Docker tools directly.

Alongside the change, Red Hat developed tools, such as Podman, skopeo, and Buildah, which all help configuring and maintaining a container workflow with minimum overhead. These tools provide the following functions:

- `Podman`: Client tool to manage containers. Replaces most features that are provided by the **`docker`** command, which focuses on individual containers or images.
- `skopeo`: Tool to manage images by copying them to and from registries.
- `runc`: Runtime client for running and working with Open Container Initiative (OCI) format.
- `Buildah`: Tool to manage OCI-compliant images.

For more information about Red Hat OpenShift on the IBM LinuxONE platform, see *Red Hat OpenShift on IBM Z Installation Guide*, REDP-5605.

8.2.2 Installing and configuring Docker

The required packages for running Docker containers on Red Hat Linux, SUSE, and Ubuntu distributions are available by using different methods.

Installing the Docker service

In this section, we describe installing Docker on each distribution:

- Red Hat Enterprise Linux (RHEL)

  a. Before installing a new version, help ensure that any older versions are uninstalled, along with their associated dependencies. Also, uninstall Podman and its associated dependencies if they are installed, as shown in Example 8-1.

Example 8-1 Uninstalling older versions of Docker

```
[root@ylsprd Terraform]# sudo yum remove docker \
                  docker-client \
                  docker-client-latest \
                  docker-common \
                  docker-latest \
                  docker-latest-logrotate \
                  docker-logrotate \
                  docker-engine \
                  Podman \
                  runc
Updating Subscription Management repositories.
Unable to read consumer identity

This system is not registered with an entitlement server. You can use
subscription-manager to register.

No match for argument: docker-client
No match for argument: docker-client-latest
No match for argument: docker-common
No match for argument: docker-latest
No match for argument: docker-latest-logrotate
No match for argument: docker-logrotate
No match for argument: docker-engine
No match for argument: runc
Dependencies resolved.
==================================================================================
==================================================================================
=================================================================================
 Package                                   Architecture
Version                                 Repository
Size
==================================================================================
==================================================================================
=================================================================================
Removing:
 Podman                                     s390x
2:4.6.1-8.el9_3                         @rhel9-appstream
53 M
 podman-docker                              noarch
2:4.6.1-8.el9_3                         @rhel9-appstream
10 k

Transaction Summary
==================================================================================
==================================================================================
=================================================================================
Remove  2 Packages

Freed space: 53 M
Is this ok [y/N]: y
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
 Preparing   :
1/1
 Erasing  :podman-docker-2:4.6.1-8.el9_3.noarch
1/2
 Runningscriptlet:podman-2:4.6.1-8.el9_3.s390x
2/2
 Erasing   :podman-2:4.6.1-8.el9_3.s390x
2/2
 Runningscriptlet:podman-2:4.6.1-8.el9_3.s390x
2/2
 Verifying  :podman-2:4.6.1-8.el9_3.s390x
1/2
 Verifying  :podman-docker-2:4.6.1-8.el9_3.noarch
2/2
Installed products updated.

Removed:
  podman-2:4.6.1-8.el9_3.s390x
podman-docker-2:4.6.1-8.el9_3.noarch

Complete!

[root@ylsprd Terraform]#
```

b. Install the `yum-utils` package, which provides the **`yum-config-manager`** utility, and set up the repository, as shown in Example 8-2.

Example 8-2 Installing the yum-utils package and setting the repository

```
[root@ylsprd Terraform]# sudo yum install -y yum-utils
Updating Subscription Management repositories.
Unable to read consumer identity

This system is not registered with an entitlement server. You can use
subscription-manager to register.

Last metadata expiration check: 1:47:23 ago on Fri 08 Mar 2024 03:20:26 PM CST.
Dependencies resolved.
=================================================================================
=================================================================================
================================================================================
 Package                                  Architecture
Version                                    Repository
Size
=================================================================================
=================================================================================
================================================================================
Installing:
 yum-utils                                noarch
4.3.0-11.el9_3                             rhel9-base
45 k

Transaction Summary
==================================================================================
==================================================================================
=================================================================================
Install  1 Package

Total download size: 45 k
Installed size: 23 k
Downloading Packages:
yum-utils-4.3.0-11.el9_3.noarch.rpm
5.9 MB/s |  45 kB     00:00
----------------------------------------------------------------------------------
----------------------------------------------------------------------------------
---------------------------------------------------------------------------------
Total
4.3 MB/s |  45 kB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
 Preparing   :
1/1Installing :yum-utils-4.3.0-11.el9_3.noarch
1/1Runningscriptlet:yum-utils-4.3.0-11.el9_3.noarch
1/1Verifying  :yum-utils-4.3.0-11.el9_3.noarch
1/1 Installed products updated.

Installed:yum-utils-4.3.0-11.el9_3.noarch

Complete!
[root@ylsprd Terraform]#
[root@ylsprd Terraform]# sudo yum-config-manager --add-repo
https://download.docker.com/linux/rhel/docker-ce.repo
Updating Subscription Management repositories.
Unable to read consumer identity

This system is not registered with an entitlement server. You can use
subscription-manager to register.

Adding repo from: https://download.docker.com/linux/rhel/docker-ce.repo
```

c. Install Docker Engine, containerd, and Docker Compose, as shown in Example 8-3.

Example 8-3 Installing Docker Engine, containerd, and Docker Compose

```
[root@ylsprd Terraform]# sudo yum install docker-ce docker-ce-cli containerd.io
docker-buildx-plugin docker-compose-plugin
Updating Subscription Management repositories.
Unable to read consumer identity
```

```
This system is not registered with an entitlement server. You can use
subscription-manager to register.

Docker CE Stable - s390x
57 kB/s | 3.5 kB     00:00
Dependencies resolved.
==================================================================================
==================================================================================
=================================================================================
 Package                                       Architecture
Version                                 Repository
Size
==================================================================================
==================================================================================
=================================================================================
Installing:
 containerd.io                                   s390x
1.6.28-3.1.el9                          docker-ce-stable
28 M
 docker-buildx-plugin                            s390x
0.12.1-1.el9                            docker-ce-stable
12 M
 docker-ce                                      s390x
3:25.0.3-1.el9                          docker-ce-stable
17 M
 docker-ce-cli                                   s390x
1:25.0.3-1.el9                          docker-ce-stable
6.9 M
 docker-compose-plugin                           s390x
2.24.5-1.el9                            docker-ce-stable
12 M
Installing weak dependencies:
 docker-ce-rootless-extras                       s390x
25.0.3-1.el9                            docker-ce-stable
4.0 M

Transaction Summary
==================================================================================
==================================================================================
=================================================================================
Install  6 Packages

Total download size: 80 M
Installed size: 353 M
Is this ok [y/N]: y
Downloading Packages:
(1/6): docker-buildx-plugin-0.12.1-1.el9.s390x.rpm 27 MB/s |  12 MB     00:00
(2/6): docker-ce-cli-25.0.3-1.el9.s390x.rpm 19 MB/s | 6.9 MB     00:00
(3/6): containerd.io-1.6.28-3.1.el9.s390x.rpm 22 MB/s |  28 MB     00:01
(4/6): docker-ce-rootless-extras-25.0.3-1.el9.s390x.rpm 7.9 MB/s | 4.0 MB
00:00
(5/6): docker-ce-25.0.3-1.el9.s390x.rpm 11 MB/s |  17 MB     00:01
(6/6): docker-compose-plugin-2.24.5-1.el9.s390x.rpm 22 MB/s |  12 MB     00:00
```

```
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
-------------------------------------------------------------------------------
Total
44 MB/s |  80 MB     00:01
Docker CE Stable - s390x
23 kB/s | 1.6 kB     00:00
Importing GPG key 0x621E9F35:
 User ID     : "Docker Release (CE rpm) <docker@docker.com>"
 Fingerprint: 060A 61C5 1B55 8A7F 742B 77AA C52F EB6B 621E 9F35
 From       : https://download.docker.com/linux/rhel/gpg
Is this ok [y/N]: y
Key imported successfully
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
 Preparing   :
1/1
Installing :docker-compose-plugin-2.24.5-1.el9.s390x
1/6
 Runningscriptlet:docker-compose-plugin-2.24.5-1.el9.s390x
1/6
 Installing :docker-buildx-plugin-0.12.1-1.el9.s390x
2/6
 Runningscriptlet:docker-buildx-plugin-0.12.1-1.el9.s390x
2/6
 Installing :docker-ce-cli-1:25.0.3-1.el9.s390x
3/6
 Runningscriptlet:docker-ce-cli-1:25.0.3-1.el9.s390x
3/6
 Installing :containerd.io-1.6.28-3.1.el9.s390x
4/6
 Runningscriptlet:containerd.io-1.6.28-3.1.el9.s390x
4/6
 Installing :docker-ce-rootless-extras-25.0.3-1.el9.s390x
5/6
 Runningscriptlet:docker-ce-rootless-extras-25.0.3-1.el9.s390x
5/6
 Installing  :docker-ce-3:25.0.3-1.el9.s390x
6/6
 Runningscriptlet:docker-ce-3:25.0.3-1.el9.s390x
6/6
 Verifying  :containerd.io-1.6.28-3.1.el9.s390x
1/6
 Verifying  :docker-buildx-plugin-0.12.1-1.el9.s390x
2/6
 Verifying  :docker-ce-3:25.0.3-1.el9.s390x
3/6
 Verifying  :docker-ce-cli-1:25.0.3-1.el9.s390x
4/6
 Verifying :docker-ce-rootless-extras-25.0.3-1.el9.s390x
5/6
```

```
 Verifying :docker-compose-plugin-2.24.5-1.el9.s390x
6/6
Installed products updated.

Installed:
containerd.io-1.6.28-3.1.el9.s390x   docker-buildx-plugin-0.12.1-1.el9.s390x
docker-ce-3:25.0.3-1.el9.s390x   docker-ce-cli-1:25.0.3-1.el9.s390x
docker-ce-rootless-extras-25.0.3-1.el9.s390x
docker-compose-plugin-2.24.5-1.el9.s390x

Complete!
[root@ylsprd Terraform]#
```

   d. Once installed, start Docker, as shown in Example 8-4.

Example 8-4   Starting Docker

```
[root@ylsprd Terraform]# sudo systemctl start docker
```

- Ubuntu

  The installation on Ubuntu is straightforward. Because the packages are available on the main repository, they can be immediately installed, as shown in Example 8-5.

Example 8-5   Installing Docker onto Ubuntu

```
root@rdbk86ub:~# apt install docker
```

  By running that command, Docker and its related tools are installed.

- SUSE

  The Docker packages on SUSE are available by way of the Container module. For more information about how to set up that module, see your distribution manual. The installation step is also simple, as shown in Example 8-6.

Example 8-6   Installing Docker on SUSE

```
rdbk86sl:~ # zypper install docker
```

Configuring the Docker service

After the installation is complete, enable and start the Docker engine. The same command is applied to Ubuntu and SUSE, but we use SUSE in the next examples (see Example 8-7).

Example 8-7   Enabling and starting Docker service

```
rdbk86sl:~ # systemctl enable --now docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service ·
/usr/lib/systemd/system/docker.service.
rdbk86sl:~ #
```

Setting up user access to Docker

Because the Docker daemon binds to a UNIX socket instead of a TCP port, the Docker daemon always runs as a root user. To avoid having to use sudo when Docker commands are used, add the user to the Docker group, as shown in Example 8-8.

Example 8-8 Adding an unprivileged user to Docker group

```
rdbk86sl:~ # usermod -aG docker lnxadmin
rdbk86sl:~ #
```

Log off and log in again for the user privilege to take effect. You can then verify the Docker commands, as shown in Example 8-9.

Example 8-9 Verifying the Docker version

```
lnxadmin@rdbk86sl:~> docker --version
Docker version 19.03.11, build 42e35e61f352
lnxadmin@rdbk86sl:~>
```

8.2.3 Testing Docker

As part of the verification, access the pre-built Docker images that are part of the Docker Hub and run a test hello-world container, as shown in Example 8-10.

Example 8-10 Running the hello-world container

```
[root@ylsprd Terraform]# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
f19d1e240d64: Pull complete
Digest: sha256:d000bc569937abbe195e20322a0bde6b2922d805332fd6d8a68b19f524b7d21d
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (s390x)
 3. The Docker daemon created a new container from that image, which runs the
    executable that produces the output you are reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

[root@ylsprd Terraform]# docker image list
REPOSITORY     TAG       IMAGE ID       CREATED         SIZE
hello-world    latest    37fc523148ea   10 months ago   9.06 kB
```

You successfully deployed Docker.

8.3 Deploying MongoDB on IBM LinuxONE

MongoDB is an open-source database that is a well known and growing NoSQL database. It works in areas where traditional SQL databases have trouble. It can work with large sets of unstructured data, and has good read times on the data that is stored. Although it is not a replacement for all SQL applications that store structured data, it provides a modern solution for massive amounts of unstructured data and mobile traffic.

With the performance and virtualization capabilities of LinuxONE, it makes an ideal platform for scaling out and scaling up MongoDB based NoSQL workloads.

To install MongoDB on LinuxONE, there are several options:

- Many Linux distributions provide MongoDB packages in their official repositories. You can use your distribution's package manager (such as `apt` for Ubuntu-based systems or `yum` for Red Hat pr CentOS-based systems) to install MongoDB.
- MongoDB provides official repositories for various Linux distributions. You can add the MongoDB repository to your system and then install MongoDB by using your package manager. With this method, you can install the latest stable version of MongoDB.
- Alternatively, you can download the MongoDB binary files from the MongoDB website and manually install them on your system. With this method, you have more control over the installation process, but you must handle the dependencies and configuration manually.
- Another option is to use containerization tools like Docker or Podman to run MongoDB in a containerized environment. This approach isolates MongoDB and its dependencies from the host system and simplifies deployment and management.

This section covers installing MongoDB by using Docker and Package Manager.

8.3.1 Deploying MongoDB as a Docker container

This section outlines and describes the deployment of MongoDB as a Docker container.

8.3.2 Work environment

This example uses Ubuntu 20.04 LTS as the host operating system for the MongoDB deployment. Because we decided to install MongoDB as a Docker container, the first step is to set up Docker on the host systems. For more information about deploying Docker on Ubuntu, see *The Virtualization Cookbook for IBM z Systems Volume 4: Ubuntu Server 16.04*, SG24-8354.

Important: The Docker installation package available in the official Ubuntu 20.04 repository might not be the latest version. To get the latest version, install Docker from the official Docker repository.

After Docker is configured, enable its service and run it on the host operating system. Example 8-11 shows verifying the Docker configuration.

Example 8-11 Verifying Docker

```
lnxadmin@rdbk86ub:~$ docker version
Client:
 Version:           19.03.8
 API version:       1.40
 Go version:        go1.13.8
 Git commit:        afacb8b7f0
 Built:             Tue Jun 23 22:26:11 2020
 OS/Arch:           linux/s390x
 Experimental:      false

Server:
 Engine:
  Version:          19.03.8
  API version:      1.40 (minimum version 1.12)
  Go version:       go1.13.8
  Git commit:       afacb8b7f0
  Built:            Thu Jun 18 08:26:54 2020
  OS/Arch:          linux/s390x
  Experimental:     false
 containerd:
  Version:          1.3.3-0ubuntu2
  GitCommit:
 runc:
  Version:          spec: 1.0.1-dev
  GitCommit:
 docker-init:
  Version:          0.18.0
  GitCommit:
lnxadmin@rdbk86ub:~$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
lnxadmin@rdbk86ub:~$
```

IBM has been working on containerizing important open source products and tools for its various platforms and also making them available on the Docker Hub public registry for download. Docker Hub is a cloud-based registry service that you can use to link to code repositories, build your images and test them, and store manually pushed images and links to Docker Cloud so you can deploy images to your hosts. It provides a centralized resource for container image discovery, distribution, and change management.

Run the **`docker search`** command to search for repositories specific to a platform in Docker Hub, as shown in Example 8-12. The command returns the pre-built Docker images for LinuxONE from the public registry.

Example 8-12 Pre-built Docker images for LinuxONE

```
lnxadmin@rdbk86ub:~$ docker search --filter=is-official=true mongodb
NAME               DESCRIPTION                                STARS          OFFICIAL
AUTOMATED
mongo               MongoDB document databases provide high avai…   7227             [OK]
mongo-express       Web-based MongoDB admin interface, written w…   788              [OK]
lnxadmin@rdbk86ub:~$
```

8.3.3 MongoDB container deployment

Now that you have a pre-built image for MongoDB from the Docker Hub, run commands to Docker to download and register the image to the local host system, as shown in Example 8-13. These images are read-only snapshots of defined layers and commands.

Example 8-13 Downloading LinuxONE MongoDB Image from Docker Hub

```
lnxadmin@rdbk86ub:~$ docker pull mongo
Using the default tag: latest
latest: Pulling from library/mongo
dd2de95b9a1c: Pull complete
c38a48ef4dfa: Pull complete
eced51184728: Pull complete
a9288641caad: Pull complete
bf72d8578ae0: Pull complete
891f80311986: Pull complete
b862844c0ef4: Pull complete
185ef58863de: Pull complete
36be598b0fb9: Pull complete
afd583fd9ef0: Pull complete
Digest: sha256:a4448eb5f6e6097353d0ab97eb50aeb0238bb4e60c37e401920d3c2c4fc73eb9
Status: Downloaded newer image for mongo:latest
docker.io/library/mongo:latest
lnxadmin@rdbk86ub:~$
```

Verify that the image was correctly registered with the local Docker registry and allocated a local image ID, as shown in Example 8-14.

Example 8-14 Verifying a MongoDB Docker image pull

```
lnxadmin@rdbk86ub:~$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
mongo               latest              abc0c0551238        2 weeks ago         491 MB
lnxadmin@rdbk86ub:~$
```

When the MongoDB container was built, the directories `/data/configdb` and `/data/db` were used as mount points for external storage, and it exposes ports 27017 and 28017. This technique allows connections from outside the container to access the MongoDB container. Example 8-15 shows the configuration.

Example 8-15 Docker Inspect

```
lnxadmin@rdbk86ub:~$ docker inspect mongo
[
    {
     "Id": "sha256:abc0c05512382652300ed10d809912071b8ee0ca14c10891cabccd8228c6dc94",
     "Created": "2020-09-25T23:30:04.442106691Z",
     ...
     "Image": "mongo/mongo",
     "Volumes": {
        "/data/configdb": {},
        "/data/db": {}
        },
    },
```

As in Example 8-15 on page 205, the pre-built MongoDB container stores the data on the /data/ folder on the host system. The idea is to create a data directory on the host system (outside the container) and mount it to a directory visible from inside the container. This configuration places the database files in a known location on the host system, and makes it easy for tools and applications on the host system to access the files. Create a folder on the host system as shown in Example 8-16.

Example 8-16 Creating a data directory

```
lnxadmin@rdbk86ub:~$ sudo mkdir -p /data/db
lnxadmin@rdbk86ub:~$ sudo mkdir -p /data/configdb
```

8.3.4 Running a MongoDB container

Starting the MongoDB container by running the **docker run** command. Example 8-17 shows that the Docker should instantiate the image that is named mongo/mongo and assign the newly instantiated container with the name itsomongo, while mounting local volumes to serve as persistent storage. Using this technique allows you to refer to the container by name rather than having to use the ID hash. If you do not provide a name, Docker assigns one from some randomly selected keywords. Also, specify the ports so that it maps the default MongoDB port 27017 to an external port.

The response to the successful instantiation is a return of a hash that is the full ID of the new running container.

Example 8-17 Starting a MongoDB container

```
lnxadmin@rdbk86ub:~$ docker run -p 27017:28017 --name itsomongo --mount
type=bind,source=/data/configdb/,target=/data/configdb --mount
type=bind,source=/data/db/,target=/data/db -d mongo
c440495d1e03ba855561b03e6114ee6dc2efe68d412b4ac9bc181a74ce29e75d
lnxadmin@rdbk86ub:~$
```

8.3.5 Verifying and accessing a MongoDB container

Check whether the container named itsomongo has started by running the **docker ps** command, as shown in Example 8-18. The status column of the command output shows that the MongoDB container is up and already listening. In addition, the output provides the mapping of 27017 and 28017 as the container's local ports.

Example 8-18 Container startup verification

```
lnxadmin@rdbk86ub:~$ docker ps
CONTAINER ID       IMAGE             COMMAND               CREATED           STATUS
PORTS                               NAMES
c440495d1e03        mongo              "docker-entrypoint.s…"   8 seconds ago       Up 7
seconds        27017/tcp, 0.0.0.0:27017->28017/tcp   itsomongo
lnxadmin@rdbk86ub:~$
```

For more information about the container, inspect the container by running the **docker inspect <container id>** command.

You can use the following methods to access MongoDB from the host:

- Use MongoDB client tools.
- Use Docker to connect to the MongoDB container shell and verify the database.

This example uses the latter option. Therefore, start a Docker interactive shell into the Mongo container and start a Mongo shell for creating a sample database (see Example 8-19).

Example 8-19 Accessing a MongoDB container by using Docker

```
lnxadmin@rdbk86ub:~$ docker exec -it itsomongo /bin/bash
root@c440495d1e03:/# mongo
MongoDB shell version v4.4.1
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("3dd136fc-e749-469a-a6b4-61c178184e9d") }
MongoDB server version: 4.4.1
Welcome to the MongoDB shell.
For more comprehensive documentation, see
        https://docs.mongodb.com/
Questions? Try the MongoDB Developer Community Forums
        https://community.mongodb.com
> use mongodb
switched to db mongodb
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
> db.itso.insert({"Redbooks":"LinuxONE"})
WriteResult({ "nInserted" : 1 })
> show dbs
admin    0.000GB
config   0.000GB
local    0.000GB
mongodb  0.000GB
>
```

This method provides a quick way to have a highly scalable environment to work on for any solution that involves MongoDB containers that are deployed on LinuxONE.

8.3.6 Migrating MongoDB data

As with any other database, MongoDB offers different options to migrate data. The most straightforward way of moving data between installations is to dump and import. This method means generating a dump on the source system, transferring it to the target system, then importing it from scratch.

Attention: This method might not be suited for all installations and environments, depending on database size or other constraints. Consult the MongoDB manuals for more options about migrating data.

Generating a dump from the source system

To generate a dump with the source system's data, you can run the **mongodump** command. In our example, we connect to the source system xrhrbres2 and run the command to generate the dump file, as shown in Example 8-20.

Example 8-20 Dumping data from the source system

```
root@xrhrbres2:/data# mongodump --out /data/db/backup/`date +"%y-%m-%d"`
2020-10-23T15:06:44.077+0000    writing mongodb.itso to
/data/db/backup/20-10-23/mongodb/itso.bson
2020-10-23T15:06:44.078+0000    done dumping mongodb.itso (1 document)
root@xrhrbres2:/data#
```

As shown in Example 8-20, the files were exported to path /data/db/backup/20-10-23/. We can now tar and compress the file, moving it to our target server rdbk86ub after using rsync. (see Example 8-21).

Example 8-21 Compressing and transferring the dump to the target system

```
root@xrhrbres2:/data/db/backup# tar -jcf 20-10-23.tar.bz2 20-10-23
root@xrhrbres2:/data/db/backup# rsync -v 20-10-23.tar.bz2
lnxadmin@rdbk86ub.pbm.ihost.com:/data/
root@xrhrbres2:/data/db/backup#
```

Importing data into the target system

On the target system, because we use Docker to run MongoDB, we must place the file on a volume that is exported to the Docker container. In our case, we are also using /data/db/backup on the target system. After the file is transferred, we extract the data to prepare for importing it onto MongoDB, as shown in Example 8-22.

Example 8-22 Extracting data to be imported

```
lnxadmin@rdbk86ub:/data/db/backup$ tar -xf 20-10-23.tar.bz2
lnxadmin@rdbk86ub:/data/db/backup$ ls -l
total 8
drwxr-xr-x 3 lnxadmin lnxadmin 4096 Oct 23 11:06 20-10-23
-rw-r--r-- 1 lnxadmin lnxadmin  385 Oct 23 11:11 20-10-23.tar.bz2
lnxadmin@rdbk86ub:/data/db/backup$
```

We can now import data onto MongoDB. In this case, we must open a shell to the active container and then run the mongorestore command to import the data, as in Example 8-23.

Example 8-23 Importing data into the deployed container

```
lnxadmin@rdbk86ub:/data/db/backup$ docker exec -it itsomongo /bin/bash
root@c440495d1e03:/data/db/backup# mongorestore --db mongodb --drop
/data/db/backup/20-10-23/mongodb/
2020-10-23T15:25:59.978+0000    building a list of collections to restore from
/data/db/backup/20-10-23/mongodb dir
2020-10-23T15:25:59.980+0000    reading metadata for mongodb.itso from
/data/db/backup/20-10-23/mongodb/itso.metadata.json
2020-10-23T15:25:59.986+0000    restoring mongodb.itso from
/data/db/backup/20-10-23/mongodb/itso.bson
2020-10-23T15:25:59.987+0000    no indexes to restore
2020-10-23T15:25:59.987+0000    finished restoring mongodb.itso (1 document, 0
failures)
```

```
2020-10-23T15:25:59.987+0000    1 documents restored successfully. 0 documents
failed to restore.
root@c440495d1e03:/data/db/backup#
```

After the import is done, we can select the data from the imported dump file, as shown in Example 8-24.

Example 8-24 Selecting data after import

```
lnxadmin@rdbk86ub:~$ docker exec -it itsomongo /bin/bash
root@c440495d1e03:/# mongo
> use mongodb
switched to db mongodb
> db.itso.find()
{ "_id" : ObjectId("5f92f5383d2effdcc886b655"), "Redbooks" : "LinuxONE" }
>
```

In this example, the database that was created on a x86 server installation was moved to a LinuxONE server that is running MongoDB as a container without any issues or special processes.

8.3.7 Deploying MongoDB by using a package manager

This section describes the deployment of MongoDB by using a package manager.

Work environment

This example uses RHEL 9.3 as the host operating system for the MongoDB deployment. Use the package manager to install MongoDB on the host system.

MongoDB installation and configuration

Complete the following steps:

1. Install essential development tools and libraries by using **yum**. Install the following packages: `gcc`, `cpp`, `make`, `autoconf`, `automake`, `libtool`, `openssl`, `openssl-devel`, `bzip2-devel`, `libcurl-devel`, `libstdc++-devel`, `systemd`, and `systemd-libs`, as shown in Example 8-25.

Example 8-25 Installing the dependencies

```
[root@redbooks1 linux1]# yum install cyrus-sasl cyrus-sasl-plain cyrus-sasl-gssapi krb5-libs
lm_sensors-libs net-snmp-agent-libs net-snmp openssl rpm-libs
```

2. Install the `compat-openssl11` package, as shown in Example 8-26.

Example 8-26 Installing the compat-openssl11 package

```
[root@redbooks1 linux1]# rpm -ivh compat-openssl11-1.1.1k-4.el9.s390x.rpm
warning: compat-openssl11-1.1.1k-4.el9.s390x.rpm: Header V3 RSA/SHA256 Signature, key ID
8483c65d: NOKEY
Verifying...                          ################################# [100%]
Preparing...                          ################################# [100%]
Updating / installing...
   1:compat-openssl11-1:1.1.1k-4.el9  ################################# [100%]
[root@redbooks1 linux1]#
```

3. Go to the MongoDB download page and select the appropriate MongoDB Enterprise package for your Red Hat or CentOS version (7.0.6) and architecture (S/390).
4. Once the package is downloaded, use the **rpm** package manager to install the MongoDB Enterprise RPM package, as shown in Example 8-27.

Example 8-27 Installing the MongoDB RPM

```
[root@redbooks1 linux1]# rpm -ivh mongodb-enterprise-server-7.0.6-1.el8.s390x.rpm
warning: mongodb-enterprise-server-7.0.6-1.el8.s390x.rpm: Header V4 RSA/SHA256 Signature, key ID
1785ba38: NOKEY
Verifying...                          ################################# [100%]
Preparing...                          ################################# [100%]
Updating / installing...
   1:mongodb-enterprise-server-7.0.6-1################################# [100%]
Created symlink /etc/systemd/system/multi-user.target.wants/mongod.service ·
/usr/lib/systemd/system/mongod.service.
```

5. Verify the MongoDB version, as shown in Example 8-28.

Example 8-28 Verifying the MongoDB version

```
[root@redbooks1 lib]# mongod --version
db version v7.0.6
Build Info: {
    "version": "7.0.6",
    "gitVersion": "66cdc1f28172cb33ff68263050d73d4ade73b9a4",
    "openSSLVersion": "OpenSSL 1.1.1k  25 Mar 2021",
    "modules": [
        "enterprise"
    ],
    "allocator": "tcmalloc",
    "environment": {
        "distmod": "rhel83",
        "distarch": "s390x",
        "target_arch": "s390x"
    }
}
[root@redbooks1 lib]#
```

6. Enable access to TCP port 27017 s for SELinux if you use enforcing mode, as shown in Example 8-29.

Example 8-29 Enabling access to a MongoDB port

```
[root@redbooks1 linux1]# semanage port -a -t mongod_port_t -p tcp 27017
```

7. Start the `mongod` process as shown in Example 8-30.

Example 8-30 Starting MongoDB

```
[root@redbooks1 linux1]# sudo service mongod start
```

8. The MongoDB Shell (**mongosh**) is a handy tool for interacting with your MongoDB database. Download the appropriate MongoDB Shell RPM package from the MongoDB website and install it by using RPM, as shown in Example 8-31 on page 211.

Example 8-31 Downloading and installing MongoDB Shell

```
[root@redbooks1 mongosh]# wget
https://downloads.mongodb.com/compass/mongosh-2.1.5-linux-s390x.tgz
--2024-03-04 14:15:26--  https://downloads.mongodb.com/compass/mongosh-2.1.5-linux-s390x.tgz
Resolving downloads.mongodb.com (downloads.mongodb.com)... 18.164.96.33, 18.164.96.65,
18.164.96.74, ...
Connecting to downloads.mongodb.com (downloads.mongodb.com)|18.164.96.33|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 60269129 (57M) [application/octet-stream]
Saving to: ‘mongosh-2.1.5-linux-s390x.tgz’

mongosh-2.1.5-linux-s390x.tgz
100%[==========================================================================================
===================================================>]  57.48M  17.9MB/s    in 3.2s

2024-03-04 14:15:29 (17.9 MB/s) - ‘mongosh-2.1.5-linux-s390x.tgz’ saved [60269129/60269129]

[root@redbooks1 mongosh]# ls
mongosh-2.1.5-linux-s390x.tgz
[root@redbooks1 mongosh]# gunzip mongosh-2.1.5-linux-s390x.tgz
[root@redbooks1 mongosh]# tar xvf mongosh-2.1.5-linux-s390x.tar
```

9. Connect to the MongoDB Shell (**mongosh**) and create a database, as shown in Example 8-32.

Example 8-32 Creating a database

```
./mongosh
Current Mongosh Log ID: 65e62c2f8f4d71346332d86c
Connecting to:
mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2
.1.5
Using MongoDB:      7.0.6
Using Mongosh:      2.1.5

For mongosh info see: https://docs.mongodb.com/mongodb-shell/


To help improve our products, anonymous usage data is collected and sent to MongoDB periodically
(https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

------
   The server generated these startup warnings when booting
   2024-03-04T11:43:50.655-06:00: Using the XFS file system is recommended with the WiredTiger
storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
   2024-03-04T11:43:50.983-06:00: Access control is not enabled for the database. Read and write
access to data and configuration is unrestricted
   2024-03-04T11:43:50.983-06:00: /sys/kernel/mm/transparent_hugepage/enabled is 'always'. We
suggest setting it to 'never'
   2024-03-04T11:43:50.983-06:00: vm.max_map_count is too low
------

Enterprise test>
Enterprise test> use ysldb
switched to db ysldb
```

```
Enterprise ysldb> show dbs
admin   40.00 KiB
config  60.00 KiB
local   72.00 KiB
ysldb1  40.00 KiB
```

Migrating MongoDB data

Use the procedure that is described in 8.3.6, “Migrating MongoDB data” on page 207.

8.4 Deploying MediaWiki and MariaDB

An application for Linux is MediaWiki, the general-purpose wiki that originated with Wikipedia. It is written in PHP and uses MariaDB as its backend database. This configuration is commonly known as a LAMP server, meaning that the application employs Linux, Apache, MariaDB, and PHP. This PHP application that has wide use and is an ideal example of a simple application migration for LinuxONE.

The Linux environment on x86 is largely the same as it is on LinuxONE, with a few notable exceptions. Configuration files on the x86 are in the same place on your Linux images on LinuxONE, unless you deliberately choose to keep them in a different place. Hence, the MariaDB configuration files, for example, typically need to be copied only from the x86 server to the LinuxONE server and placed in the same location in the file system (`/etc/my.cnf`).

Migrating to LinuxONE should be tested first in a test environment before performing the migration to the production environment.

8.4.1 Analysis and planning

Following the guidelines and recommendations that are outlined in Chapter 4, “Migration process” on page 63, and Chapter 5, “Migration analysis” on page 75, appropriate planning and analysis should be performed before these migration activities. The checklists are helpful in identifying how virtual resources should be dedicated and organized.

For this example scenario, the Linux image is set up and a minimal Linux operating system installed. The Linux guest is called `rdbk86sl` and is running SUSE Linux Enterprise Server 15 SP2, with one virtual CPU and 1 GB of virtual memory. It is assumed that an adequate package management (RPM) repository for installation source is set up and available for the installation of the application software.

8.4.2 Installing the LAMP stack

The LAMP stack is available on all distributions; however, not every distribution packs them together in a single bundle.

The installation of the application software can be done by using YaST for SUSE Linux Enterprise Server 15.

Installing LAMP on SUSE Linux Enterprise Server

First, help ensure that SUSE Linux Enterprise Server has a pattern (a collective group of related packages) for a LAMP server. Issue **`zypper info -t pattern lamp_server`** to see the packages that are associated with a LAMP server.

Example 8-33 shows the helpful information that is displayed about LAMP by running the following command:

```
zypper info -t pattern lamp_server
```

Example 8-33 LAMP pattern output from zypper

```
rdbk86sl:~ # zypper info -t pattern lamp_server
Information for pattern lamp_server:
------------------------------------
Repository      : Server-Applications-Module 15.2-0
Name            : lamp_server
Version         : 20180302-11.1
Arch            : s390x
Vendor          : SUSE LLC <https://www.suse.com/>
Installed       : No
Visible to User : Yes
Summary         : Web and LAMP Server
Description     :
    Software to set up a Web server that is able to serve static, dynamic, and
interactive content (like a Web shop). This includes Apache HTTP Server, the
    database management system MySQL, and scripting languages such as PHP, Python,
Ruby on Rails, or Perl.
Contents        :
    S | Name                        | Type    | Dependency
    --+-----------------------------+---------+------------
      | apache2                     | package | Required
    i | patterns-base-basesystem    | package | Required
      | patterns-server-lamp_server | package | Required
      | apache2-doc                 | package | Recommended
      | apache2-prefork             | package | Recommended
      | libapr-util1                | package | Recommended
      | libapr1                     | package | Recommended
      | mariadb                     | package | Recommended
    i | perl                        | package | Recommended
rdbk86sl:~ #
```

Note: The `lamp_server` pattern includes the Apache and MariaDB components, but is missing the PHP component because the "P" in "LAMP", stands for "Perl," which is often used as the server-side dynamic web page engine.

Install the packages under that pattern by running the following command:

```
zypper install -t pattern lamp_server
```

The **`zypper`** command reports which packages are expected to be installed, then prompts for confirmation to continue. Press `y` and `Enter` to install the packages.

Install the remaining PHP packages by running the following command:

```
zypper install apache2-mod_php7 php7-mysql
```

Attention: The PHP packages are under SUSE Linux Enterprise Server Web and Scripting Module. For more information about activating this support module, see the distribution's documentation.

8.4.3 Starting and testing LAMP components

Before migrating the MediaWiki software, configure and test Apache, PHP, and MariaDB on the target system to help ensure that they are working. This process reduces the number of variables to debug if something goes wrong.

The Apache and MariaDB configurations in this example scenario are simple, whereas your configuration might be more complex. Migrating the Apache and MariaDB configurations can be a more complex process. This example presumes that MediaWiki is the only application that is configured for Apache and that no other data exists in the MariaDB database than what is used by MediaWiki.

Confirm that the version of Apache is what is expected. A common method of displaying the version is by running the **`apachectl -v`** command, which is available by default on the distribution.

Example 8-34 shows the version of `apache2` as displayed by running the **`apachectl -v`** command in SUSE Linux Enterprise Server.

Example 8-34 Output of the apachectl -v command

```
rdbk86sl:~ # apachectl -v
Server version: Apache/2.4.43 (Linux/SUSE)
Server built:   2020-04-03 13:57:53.000000000 +0000
rdbk86sl:~ #
```

Historically, it was common to have the installed services started automatically when the package was installed. Today, it is more common that the installer help ensure that other software is not started automatically. Therefore, it is necessary to start Apache manually, and to set it to start automatically each time the system is started.

Apache services

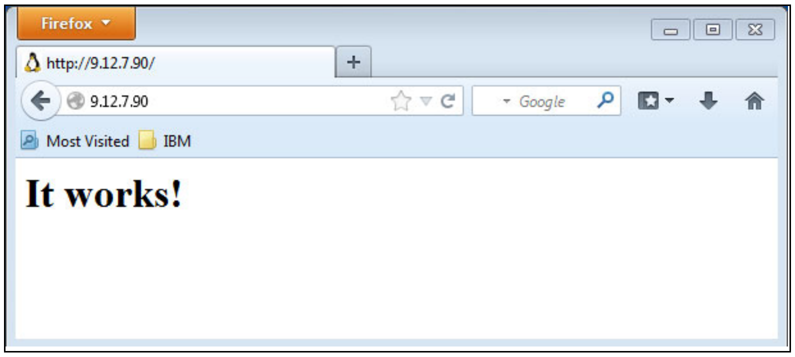
Set the `apache2` service to automatically start each time that the server is started and start the service by running the commands that are shown in Example 8-35.

Example 8-35 Starting the apache2 web service

```
rdbk86sl:~ # systemctl enable --now apache2
Created symlink /etc/systemd/system/httpd.service ·
/usr/lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/apache.service ·
/usr/lib/systemd/system/apache2.service.
Created symlink /etc/systemd/system/multi-user.target.wants/apache2.service ·
/usr/lib/systemd/system/apache2.service.
rdbk86sl:~ #
```

Verifying that the web server is running

With the web service started, use a web browser to verify that the web server is working as expected, as shown in Figure 8-1 on page 215. Start a web browser and point it to the IP address of the Linux server. In this example, the URL is `http://9.12.7.90`.



It works!

Figure 8-1   Successful test of the Apache installation

Verifying that PHP is working

Before a test can be conceived and run for PHP, the location of the DocumentRoot directory of the Apache server must be determined.

In SUSE Linux Enterprise Server, the default location is `/srv/www/htdocs`. However, a non-default location might be configured. The document root directory can be determined by running the commands that are shown in Example 8-36.

Example 8-36   Finding the DocumentRoot on SUSE Linux Enterprise Server

```
rdbk86sl:~ # grep 'DocumentRoot "' /etc/apache2/default-server.conf
DocumentRoot "/srv/www/htdocs"
rdbk86sl:~ #
```

After confirming the Document Root of the Apache server, a one-line PHP script is created that prints the standard PHP installation information. Using vi or some other suitable text editor, create a script file that is called `phpinfo.php`, as shown in Example 8-37, and place the script file in the appropriate DocumentRoot directory.

Example 8-37   Simple PHP script that displays functional characteristics

```
<?php phpinfo(); ?>
```

The necessary modules to run the PHP script were installed as described in “Installing LAMP on SUSE Linux Enterprise Server” on page 212 but are not enabled by default. To enable the PHP 7 module under Apache 2, run the commands as shown in Example 8-38.

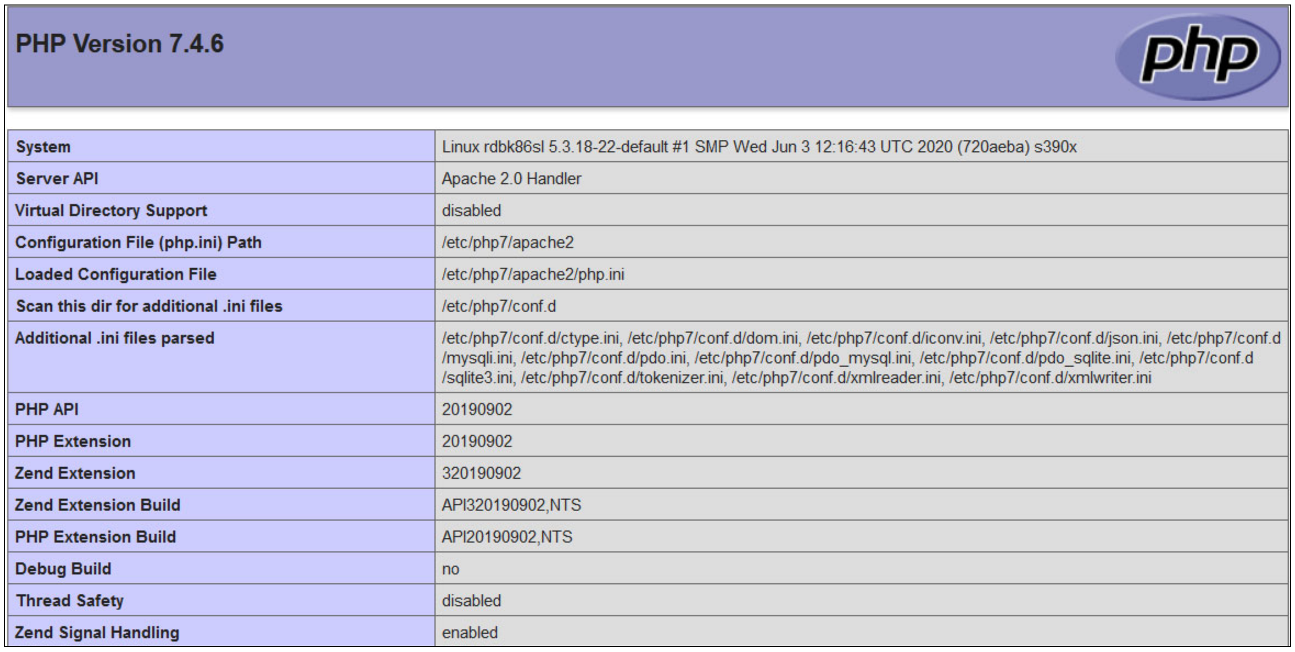
Example 8-38   Activating the PHP7 module for Apache

```
rdbk86sl:~ # a2enmod php7
rdbk86sl:~ #
```

With the PHP script file in the DocumentRoot directory and the module enabled, the PHP script can be run by using a web browser. Connect to your web serve by using the following URL as an example:

```
http://9.12.7.90/phpinfo.php
```

Figure 8-2 shows the expected PHP information that is generated in the browser by the PHP script running on SUSE Linux Enterprise Server 15 SP2.


PHP Version 7.4.6

System	Linux rdbk86sl 5.3.18-22-default #1 SMP Wed Jun 3 12:16:43 UTC 2020 (720aeba) s390x
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php7/apache2
Loaded Configuration File	/etc/php7/apache2/php.ini
Scan this dir for additional .ini files	/etc/php7/conf.d
Additional .ini files parsed	/etc/php7/conf.d/ctype.ini, /etc/php7/conf.d/dom.ini, /etc/php7/conf.d/iconv.ini, /etc/php7/conf.d/json.ini, /etc/php7/conf.d/mysqli.ini, /etc/php7/conf.d/pdo.ini, /etc/php7/conf.d/pdo_mysql.ini, /etc/php7/conf.d/pdo_sqlite.ini, /etc/php7/conf.d/sqlite3.ini, /etc/php7/conf.d/tokenizer.ini, /etc/php7/conf.d/xmlreader.ini, /etc/php7/conf.d/xmlwriter.ini
PHP API	20190902
PHP Extension	20190902
Zend Extension	320190902
Zend Extension Build	API320190902,NTS
PHP Extension Build	API20190902,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled

Figure 8-2 PHP configuration information that is generated by phpinfo.php

Starting the MariaDB service

Set the MariaDB database service to automatically start each time that the server is started, then manually start the service by using the commands as shown in Example 8-39.

Example 8-39 Enabling and starting the MariaDB service

```
rdbk86sl:~ # systemctl enable --now mariadb
Created symlink /etc/systemd/system/mysql.service ·
/usr/lib/systemd/system/mariadb.service.
Created symlink /etc/systemd/system/multi-user.target.wants/mariadb.service ·
/usr/lib/systemd/system/mariadb.service.
rdbk86sl:~ #
```

Verifying that MariaDB is working

MariaDB must be configured and working properly before MediaWiki can even be installed. Complete the following steps to verify that MariaDB is working:

1. Set a temporary password for the database administrative user. Remember this password because it is required during a few more steps of the process before migrating the MediaWiki application from the x86 server. This password can be the same password as the MariaDB database that was migrated, although it is not necessary. Run the command that is shown in Example 8-40 to set the password.

   Example 8-40 Setting an administrative password for the MariaDB service

   ```
   mysqladmin -u root password ‘agoodpassword’
   ```

   With the admin password set for the root user, all future interactions with the MariaDB database require providing a password. General administrative functions require the root password, whereas commands that involve MediaWiki use a different password.

Note: Quotation marks in Linux can be a bit tricky. When setting the root password, keep in mind that the quotation marks are not strictly necessary. If the password contains special characters (such as a space), the quotation marks *are* necessary.

Do not use quotations marks unless you are certain that they are necessary. Copying a string from somewhere and pasting the string as the password can give unexpected results, and might make reproducing the password later an inconvenient mystery.

2. Test the MariaDB capabilities by running the following sample command:

   ```
   mysql -u root -p -e "show tables" mysql
   ```

   The preceding command prompts you for the root password that you set in the previous steps with the **`mysqladmin`** command. The sample output that is shown in Example 8-41 shows the list of tables that are contained in the `mysql` database, proving that you have properly set the password.

Example 8-41 Output from the "show tables" mysql command after providing a password

```
rdbk86sl:~ # mysql -u root -p -e "show tables" mysql
Enter the password:
+---------------------------+
| Tables_in_mysql           |
+---------------------------+
| column_stats              |
| columns_priv              |
| db                        |
| event                     |
| func                      |
| general_log               |
| global_priv               |
| gtid_slave_pos            |
| help_category             |
| help_keyword              |
| help_relation             |
| help_topic                |
| index_stats               |
| innodb_index_stats        |
| innodb_table_stats        |
| plugin                    |
| proc                      |
| procs_priv                |
| proxies_priv              |
| roles_mapping             |
| servers                   |
| slow_log                  |
| table_stats               |
| tables_priv               |
| time_zone                 |
| time_zone_leap_second     |
| time_zone_name            |
| time_zone_transition      |
| time_zone_transition_type |
| transaction_registry      |
| user                      |
+---------------------------+
rdbk86sl:~ #
```

With the MariaDB administrative password properly set, you can install the MediaWiki software. If the MariaDB administrative password was set up incorrectly, an error message that is similar to the message that is shown in Example 8-42 is displayed.

Example 8-42 A bad password that is supplied to MariaDB

```
rdbk86sl:~ # mysql -u root -p -e "show tables" mysql
Enter the password:
ERROR 1698 (28000): Access denied for user 'root'@'localhost'
rdbk86sl:~ #
```

To correct this problem, run the **mysqladmin** command again as shown in Example 8-40 on page 216, taking extra care to set the password to a value that you remember. If the original password cannot be remembered or is otherwise lost, you must reinstall MariaDB.

With the preliminary Apache, MariaDB, and PHP configurations functioning properly on the new LinuxONE server, the application can now be migrated from the x86 server.

8.4.4 Migrating to LinuxONE

The MediaWiki application is installed on an x86 server that is called `xrhrbres1`, running on RHEL 8. It uses MariaDB and Apache 2 to run the application. It is installed on path `/var/www/html/mw/`.

The target server `rdbk86sl` is running SUSE Linux Enterprise Server 15 SP 2.

To migrate the application, we complete the following steps:

1. Dump the database contents on the source server.
2. Copy the database dump and application files to the target server.
3. Import the database dump into the target server.
4. Test the application.

Dumping the database on the source server

The first step is to dump the database contents from the source server installation. To achieve that, we used the **mysqldump** command, as shown in Example 8-43.

Example 8-43 Dumping the database by using the mysqldump command

```
[root@xrhrbres1 ~]# mysqldump --databases my_wiki -p -r ~/sqldump.sql
Enter the password:
[root@xrhrbres1 ~]#
```

A `.sql` file is created that includes the databases contents. This file must now be transferred to the target system by using `rsync`.

Copying data to the target server

The required data must now be sent to the target server. We use `rsync` to transfer the files:

- Application data: From source `/var/www/html/mw/` to target `/srv/www/htdocs/mw/`.
- Database dump: From source `~/sqldump.sql` to target `/root/`.
- Services configuration: Files for `apache` and `mariadb` must be manually analyzed and copied over. This process is out of the scope of this example.

This **rsync** command copies the required data to move the application, as shown in Example 8-44.

Example 8-44 Copying data to the target server

```
[root@xrhrbres1 ~]# rsync -a /var/www/html/mw/
rdbk86sl.pbm.ihost.com:/srv/www/htdocs/mw/
Password:
[root@xrhrbres1 ~]# rsync -a ~/sqldump.sql rdbk86sl.pbm.ihost.com:/root/
Password:
[root@xrhrbres1 ~]#
```

After the copy, verify the configuration files for `apache` and `mariadb` as needed.

Importing the database data to the target server

The database dump can be imported by using the **mysql** command. To start the import process, use the database name `my_wiki` and direct the dump file as standard in for the command with the < character, as shown in Example 8-45.

Example 8-45 Importing SQL data

```
rdbk86sl:/ # mysql my_wiki <~/sqldump.sql
rdbk86sl:/ #
```

No output means that the command was completed successfully. Now, we can run some SQL commands to verify that the data was inserted into the database (see Example 8-46).

Example 8-46 Querying data on the target server after importing

```
rdbk86sl:~ # mysql my_wiki
MariaDB [my_wiki]> show tables;
+-----------------------+
| Tables_in_my_wiki     |
+-----------------------+
| actor                 |
| archive               |
| bot_passwords         |
| category              |
| categorylinks         |
| change_tag            |
| comment               |
| content               |
| content_models        |
| externallinks         |
| filearchive           |
| image                 |
| image_comment_temp    |
| imagelinks            |
| interwiki             |
| ip_changes            |
| ipblocks              |
| iwlinks               |
| job                   |
| l10n_cache            |
| langlinks             |
| log_search            |
| logging                |
| module_deps            |
| objectcache            |
| oldimage               |
| page                   |
| page_props             |
| page_restrictions      |
| pagelinks              |
| protected_titles       |
| querycache             |
| querycache_info        |
| querycachetwo          |
| recentchanges          |
| redirect               |
| revision               |
| revision_actor_temp    |
| revision_comment_temp  |
| searchindex            |
| site_identifiers       |
| site_stats             |
| sites                  |
| slot_roles             |
| slots                  |
| tag_summary            |
| templatelinks          |
| text                   |
| transcache             |
| updatelog              |
| uploadstash            |
| user                   |
| user_former_groups     |
| user_groups            |
| user_newtalk           |
| user_properties        |
| valid_tag              |
| watchlist              |
+------------------------+
58 rows in set (0.000 sec)
```

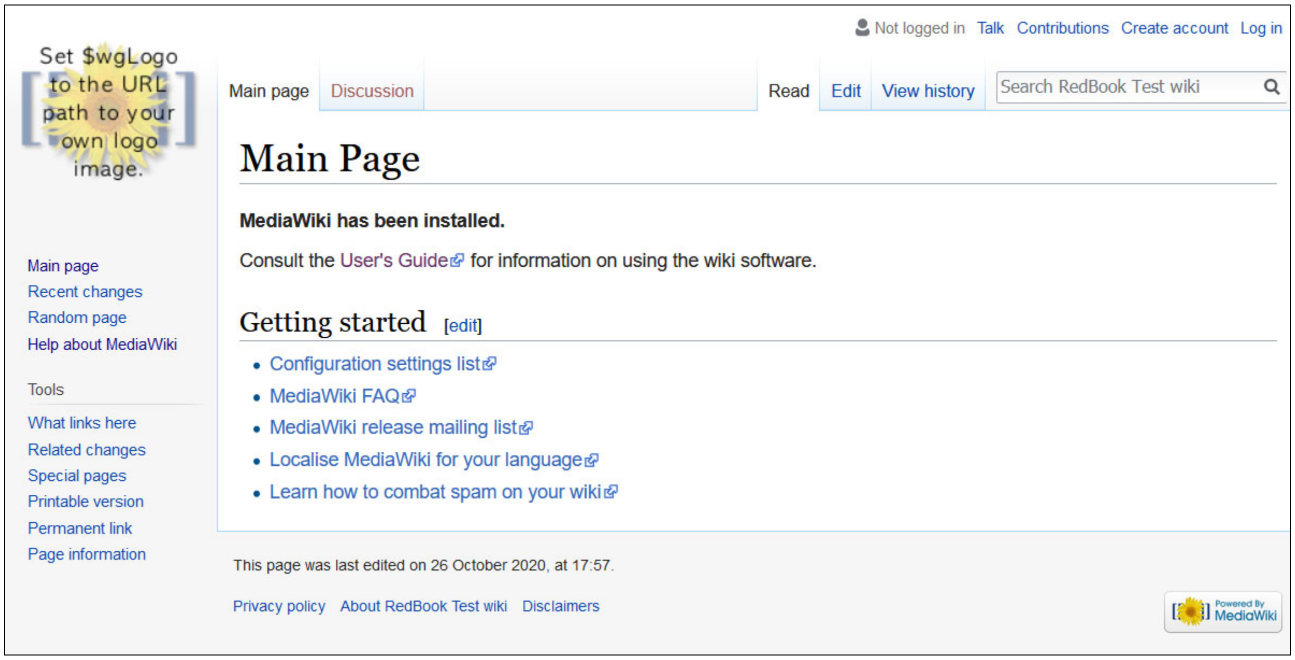
With the database imported, we can move on to validate the installation.

Completing the migration of services

The remaining tasks are critical and the most specialized. Copy the Apache and MariaDB configuration files from the x86 host to the LinuxONE guest, then compare and adapt them as necessary. Notice that the configuration of MediaWiki requires no special steps because it was all moved when the application files were copied. Sometimes, the configurations are simple enough to allow a basic copy of the files. Other circumstances can be more complex, which require rewriting the configuration files.

Note: Make sure that the application includes the correct password that is configured on the source and target system (in case they differ).

Having successfully migrated Apache, MariaDB, and their respective data (including the MediaWiki data), the MediaWiki application should now be functional. Open the MediaWiki URL by using a browser. The web page that is shown in Figure 8-3 represents a successful installation of MediaWiki.



Main Page

MediaWiki has been installed.

Consult the User's Guide for information on using the wiki software.

Getting started [edit]

- Configuration settings list
- MediaWiki FAQ
- MediaWiki release mailing list
- Localise MediaWiki for your language
- Learn how to combat spam on your wiki

This page was last edited on 26 October 2020, at 17:57.

Privacy policy About RedBook Test wiki Disclaimers

Figure 8-3 MediaWiki running on the target LinuxONE server with migrated data

8.5 Deploying OpenLDAP

Enterprises of all sizes need to manage the users of their computing resources. And with the user management comes the various characteristics of the user, such as user ID, authentication, file system rights, printer rights, and more, all needing to be managed. One of the most common products that are used for managing this data is the Lightweight Directory Access Protocol (LDAP).

LDAP is widely used throughout the industry for directory services as an open standard running over an IP network. Although several commercial LDAP products are available, OpenLDAP is the implementation that is most commonly used in Linux. OpenLDAP is a fully featured suite of tools and applications. It is readily available as a workload on LinuxONE from all available distributions. LDAP is a perfect workload for LinuxONE, due to the centrality of LinuxONE among many other systems and services, its fast I/O, and its low CPU and memory usage. And OpenLDAP is open source. Migrating OpenLDAP to LinuxONE is straightforward.

We installed a LAMP server with MediaWiki, and iSCSI external storage was used to facilitate the migration in 8.4, “Deploying MediaWiki and MariaDB” on page 212. In this example, the LDAP database on an x86 server is exported; the database is transferred to a Linux guest running on LinuxONE; and the data is imported into the LDAP service.

8.5.1 Analysis and planning

As with the examples that are described in 8.4, “Deploying MediaWiki and MariaDB” on page 212, it is important that you follow Chapter 4, “Migration process” on page 63, and Chapter 5, “Migration analysis” on page 75. Perform this planning and analysis before any migration activity. The checklists help identify the many considerations that should be considered to help prevent problems during migration.

This example assumes that the Linux guest is set up and a minimal Linux operating system is installed. The Linux guest is called `rdbk86sl` and is running SUSE Linux Enterprise Server 15 SP2, with four virtual CPUs and 8 GB of virtual memory. An OpenLDAP server typically does not require a large amount of CPU or RAM running on LinuxONE. It is presumed that an adequate RPM repository installation source is already set up and available for the installation of the application software.

The x86 server is called `zs4p01-r1` and is running RHEL 7. For this example, this is the current OpenLDAP server that provides directory services for the hypothetical organization. This server has a rudimentary (small) LDAP directory already configured.

Although there is much to consider when setting up an enterprise directory service, a simple OpenLDAP scenario is covered here. For more information, see OpenLDAP.

This example is a stand-alone server with a local, non-replicated directory service. Nevertheless, migrating an existing OpenLDAP installation on x86 to LinuxONE should be straightforward.

8.5.2 Installing LDAP software

The OpenLDAP server is a simple application, consisting of a single package. Therefore, installing the software is relatively simple. The software must first be installed on the LinuxONE guest before the other migration steps.

If you are going to install OpenLDAP on SUSE Linux Enterprise Server, run the following command to install the package:

```
zypper install openldap2
```

To install OpenLDAP on RHEL 7, run the following command:

```
yum install openldap-servers
```

On Ubuntu, OpenLDAP can be installed with the following command:

```
apt install slapd ldap-utils
```

Attention: On RHEL 8, Red Hat no longer supports openldap servers; instead, a subscription-based service is offered that is called Red Hat Directory Server (RHDS). The `openldap-clients` package is still offered. For more information, see RHEL 8 documentation.

8.5.3 Configuring the OpenLDAP service

The principal player in the OpenLDAP server suite of applications is the Standalone LDAP Daemon, which is known as `slapd`. This example configures the `slapd` service to operate as a stand-alone, local, non-replicated directory. The package, in RPM format for SUSE, contains functional sample configuration files, which serve as the basis of the example service that is configured here.

The initial configuration of OpenLDAP on SUSE Linux Enterprise Server running on LinuxONE is accomplished by using YaST, whereas configuration on Red Hat or Ubuntu is done by manually modifying configuration files and running commands.

Before migrating the LDAP database to LinuxONE, establish a basic configuration of OpenLDAP. Using different terminology, the OpenLDAP configuration must be started, also known as *bootstrapped*.

Note: OpenLDAP maintains its configuration by using one of two different configuration methods. The "old" method involves maintaining the primary configuration in `/etc/openldap/slapd.conf`. This method is simple, but does not have as many features. The "new" way (called the cn=config format) uses several configuration files under `/etc/openldap/slapd.d/`. The default behavior with OpenLDAP 2.4 is to use the cn=config method.

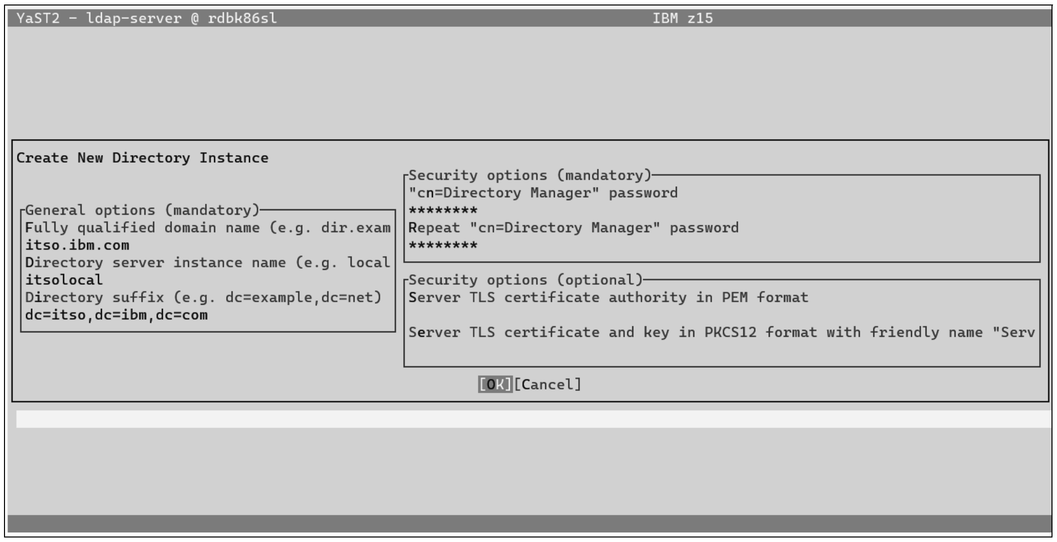
Configuring OpenLDAP on SUSE Linux Enterprise Server by using YaST

All activities to create a basic configuration of OpenLDAP are facilitated by the LDAP server YaST module. By following a few simple screens in YaST, the LDAP services can be configured and running in short order.

From a command prompt, start YaST, calling specifically the `ldap-server` module:

```
yast2 ldap-server
```

Figure 8-4 shows the configuration window.



```
YaST2 - ldap-server @ rdbk86sl                                  IBM z15

Create New Directory Instance
                                         ┌Security options (mandatory)
                                         "cn=Directory Manager" password
┌General options (mandatory)             ********
Fully qualified domain name (e.g. dir.exam   Repeat "cn=Directory Manager" password
itso.ibm.com                             ********
Directory server instance name (e.g. local
itsolocal                                ┌Security options (optional)
Directory suffix (e.g. dc=example,dc=net)   Server TLS certificate authority in PEM format
dc=itso,dc=ibm,dc=com
                                         Server TLS certificate and key in PKCS12 format with friendly name "Serv

                                    [OK][Cancel]
```

Figure 8-4 OpenLDAP configuration on YaST2

This window includes the following fields that must be completed to proceed:

- Fully qualified domain name
- Directory server instance name
- Directory suffix
- Directory manager password
- Server Transport Layer Security (TLS) certificate authority in PEM format (not used in this example)
- Server TLS certificate and key in PKCS12 format (not used in this example)

When all fields are completed, we can proceed and YaST will install and activate the service. The confirmation message is displayed at the end. The use of a suitable Secure Sockets Layer (SSL) certificate is a best practice, but not necessary for this demonstration.

Note: Using SSL for LDAP (also known as Lightweight Directory Access Protocol Over Secure Socket Links [LDAPS]) is essential. Without LDAPS, passwords and other sensitive data are exchanged with the LDAP server in plain text. This method makes the system vulnerable and is not recommended for production.

In a production environment, the correct distinguished name (DN) data must be entered, but it is adequate for this demonstration to use the sample values that are supplied by YaST. What is most important here is providing an administrator password. This password should not be the same as the system's root password. All other preferred practices for creating an administrative password should likewise be used.

With all the configuration information sufficiently gathered, the YaST configuration steps can be completed. The configuration files are written, and the `slapd` daemon is started. The running daemon process is shown in Example 8-47.

Example 8-47 The slapd daemon is shown running by using the cn=config method

```
rdbk86sl:~ # ps -ef | grep slapd
dirsrv   41259     1  2 13:36 ?        00:00:01 /usr/sbin/ns-slapd -D
/etc/dirsrv/slapd-itsorh -i /run/dirsrv/slapd-itsorh.pid
root     41871 36821  0 13:36 pts/0    00:00:00 grep --color=auto slapd
rdbk86sl:~ #
```

Configuring OpenLDAP manually on Ubuntu

The configuration on the Ubuntu server is also a relatively simple task because all that is needed is a basic, bootstrappable configuration. This basic configuration alone is not useful for running a suitable directory, but it allows the migration of the openLDAP directory from another server. OpenLDAP 2.4 and on uses the `cn=config` feature configuration format by default:

1. Help ensure that the `slapd` daemon is running:

   ```
   service slapd start
   ```

2. From a command prompt on the server, edit a basic OpenLDAP configuration file, such as using `vi`, as shown in the following example:

   ```
   vi /tmp/config.itso.ibm.com.ldif
   ```

   Add the content to the file as shown in Example 8-48 on page 225.

Example 8-48 /tmp/config.itso.ibm.com.ldif file to bootstrap the OpenLDAP database

```
dn: olcDatabase=hdb,cn=config
objectClass: olcDatabaseConfig
objectClass: olcHdbConfig
olcDatabase: {1}hdb
olcSuffix: dc=itso,dc=ibm,dc=com
olcDbDirectory: /var/lib/ldap
olcRootDN: cn=Administrator,dc=itso,dc=ibm,dc=com
olcRootPW: ldapadmin
olcAccess: to attrs=userPassword by dn="cn=Administrator,dc=itso,dc=ibm,dc=com"
write by anonymous auth by self-write by * none
olcAccess: to attrs=shadowLastChange by self-write by * read
olcAccess: to dn.base="" by * read
olcAccess: to * by dn="cn=Administrator,dc=itso,dc=ibm,dc=com" write by * read
```

3. Save the file and exit the editor.
4. Bootstrap the database and import the configuration from the file that was created as shown in Example 8-48 by using the following command:

   ```
   ldapadd -Y EXTERNAL -H ldapi:/// -f /tmp/config.itso.ibm.com.ldif
   ```

Now, the basic configuration of OpenLDAP allows a migration of the database.

8.5.4 Exporting OpenLDAP data from x86 server

The LDAP directory tree running on the x86 server now needs to be exported so that the data can be transferred to the Linux guest on LinuxONE. To do this task, complete the following steps:

1. Connect to the x86 host `zs4p01-r1` by using Secure Shell (SSH). This example is on an RHEL server.
2. Stop the slapd daemon so that the data can be exported from OpenLDAP:

   ```
   service slapd stop
   ```

3. Export the data from the OpenLDAP database. The tool that is used to accomplish this task is called *slapcat*, which is a common method of extracting whole data sets from OpenLDAP. The output is written in LDAP Data Interchange Format (LDIF), which is a standard plain text data interchange format for representing LDAP:

   ```
   slapcat -b 'dc=itso,dc=ibm,dc=com' -l /tmp/migrate.ldif
   ```

   The `-l` argument tells slapcat to export the database (in the LDIF format) to the file `/tmp/migrate.ldif`. The `-b` argument identifies the specific domain of data to export (known as the suffix in the OpenLDAP vernacular).
4. (Optional) Restart the `slapd` daemon on `zs4p01-r1`. Because the daemon is being migrated to another server, it might not be necessary to restart it.

   ```
   service slapd start
   ```

5. Transfer the database file to the Linux guest running on LinuxONE. Use the transfer mechanism that is most suitable. This example uses a utility software and network protocol called `rsync`:

   ```
   rsync /tmp/migrate.ldif 9.12.7.90:/tmp/
   ```

The server with the IP address `9.12.7.90` is rdbk86sl and is the Linux guest on LinuxONE. Provide the correct credentials when prompted. When the transfer is complete, the process of exporting the data from this x86 server to the Linux guest running on LinuxONE is completed.

8.5.5 Importing OpenLDAP data to LinuxONE

In the previous section, the OpenLDAP database export file was transferred to `rdbk86sl`, which is the Linux guest running on LinuxONE. All that is required now is to import the data and start the OpenLDAP daemon:

1. Reconnect to the Linux guest `rdbk86sl` by using SSH.
2. Help ensure that `slapd` is not running. Importing data for migration requires that the service is not running:

   ```
   service slapd stop
   ```

3. Import the data that was copied. This process employs a tool that is called *slapadd*. This is a common method of importing whole data sets into OpenLDAP:

   ```
   slapadd -F /etc/openldap/slapd.d \
   -b 'dc=itso,dc=ibm,dc=com' -l  /tmp/migrate.ldif
   ```

   Because the basic configuration was established as described in 8.5.3, “Configuring the OpenLDAP service” on page 223, the `itso.ibm.com` domain exists in the new OpenLDAP database, which makes it simple to import the data. The `-b` argument identifies the domain, and the `-l` argument indicates the LDIF file from which the database information will be imported.

   A successful import shows 100% success, as shown in Example 8-49. Any value other than 100% means that something went wrong and the import of the data was not successful.

   Example 8-49 Import of OpenLDAP data is 100% successful

   ```
   rdbk86sl:- # slapdd -F /etc/openldap/slapd.d -b ‘dc=itso,dc=ibm,dc=c
   om’ -l /tmp/migrate.ldif
   hdb_monitor_db_open: monitoring disabled; configure monitor database to enable
   _################## 100.00% eta none elapsed none fast!
   Closing DB...
   ```

4. After the database is successfully imported, OpenLDAP can be started again and is ready to receive queries:

   ```
   service slapd start
   ```

8.5.6 Verifying that OpenLDAP is working

The `slapd` process is running, and sample data is presumed to exist in the directory, but that does not necessarily mean that OpenLDAP is usable by any clients. Test that the LDAP server responds to client requests. In this example, the user `fred` is queried:

```
ldapsearch -xLLL -H ldapi:/// -b "dc=itso,dc=ibm,dc=com" uid=fred sn givenName cn
```

Example 8-50 on page 227 shows the results of the `ldapsearch` query.

Example 8-50 Output from ldapsearch that shows that user fred exists in the directory

```
rdbk86sl:- # ldapsearch -xLLL -H ldapi:/// -b "dc=itso,dc=ibm,dc=com" uid=fred sn
 givenName cn
dn: uid=fred,ou=employees,dc=itso,dc=ibm,dc=com
sn: frandsen
cn: fred
```

As shown in Example 8-50, the OpenLDAP client and the server are both running on the same system, which is not necessarily a convincing demonstration. A better verification is whether an external client can query the OpenLDAP server over the network. Example 8-51 shows that a different client, `zs4p01-s1`, queries the LDAP directory that is running on `rdbk86sl` (`9.12.7.90`).

Example 8-51 Output from ldapsearch querying the LDAP directory over the network

```
zs4p01-s1:~ # ldapsearch -xLLL -H ldap://9.12.7.90 \
>                         -b "dc=itso,dc=ibm,dc=com" \
>                         uid=fred sn givenName cn
dn: uid=fred,ou=employees,dc=itso,dc=ibm,dc=com
sn: frandsen
cn: fred
```

This second verification in Example 8-51 indicates a successful migration of an OpenLDAP service from Linux on x86 to LinuxONE. Not only that, but the service was migrated from a system running RHEL to one running SUSE Linux Enterprise Server. OpenLDAP, Linux, and LinuxONE are all happy regardless of the distribution, and the migration of OpenLDAP is unhampered regardless of the distribution.

8.6 Deploying a central log server

As you saw in "Logging and recording events" on page 121, forwarding local log records to a remote secure system is a good practice to keep your log records safe. When someone does attempt to attack one of the servers, they probably try to clean up their tracks. By using remote centralized log servers, you can keep a safe copy even if they remove the local copies or stop the service. Also, you can centralize all logs from your environment and use a real-time search and analytics tool to create business insights or a monitoring tool.

To create a centralized application log server, use the default log daemon from SUSE, `rsyslog` (version 8.39). It is also the default logging server on RHEL 8 and Ubuntu 20.04.

Note: After `systemd` was introduced, system and service messages are logged by `systemd-journald` and stored in its own format. To forward system-specific logs, see your distribution manual for more information about configuring `journald`.

8.6.1 Analysis and planning

Use the Logical Volume Manager (LVM) to create a logical volume (LV) for log files because log files tend to grow fast. With different hosts writing logs to a centralized log server at the same time, log files can fill your disk even faster. Therefore, leave some space available in a volume group (VG) that can be used during an emergency.

8.6.2 Initial configuration

The default path for the `rsyslog` configuration file is `/etc/rsyslog.conf`. It is composed of key words that define the message route and global options. You can see all the available global options by running the **`man rsyslog.conf`** command.

Modular nature

The `rsyslog` server is based on several modules that can be enabled to extend the regular functions of the service. Every module can provide more options for how the log sources are managed and how the output is handled. For more information about each module, see the `rsyslog.conf` manual.

Global directives

You can specify several global options (called *directives*) in the statements of your `rsyslog` configuration file. You can define how the hostname of the client appears in the log files, enable or disable the DNS cache, use the ownership of the files, enable and configure modules, and some other features that you use depending on the size or specific requirements of your environment.

Selectors

A selector is a combination of a facility and a priority, which are separated by a period **"."**. They can be found on the man page of `syslog(3)`. They can be represented by an asterisk **"*"**, meaning all items under that category. Example 8-52 shows some of the default facilities and priorities.

Example 8-52 Sample facilities and priorities as seen on rsyslog.conf under SUSE 15

```
*.info;mail.none;authpriv.none;cron.none
authpriv.*
mail.*
```

Configuration modularity

The configuration model also supports modular configuration files. A statement in the main `rsyslog.conf` includes all files under `/etc/rsyslog.d/*.conf` and reads them at service startup. This eases the maintenance of custom configuration files throughout the infrastructure.

For example, a configuration file can be created to forward all the server's logs to a central logging server and then deploy that configuration file on all systems. If an update is needed, only that file must be updated without touching the main configuration file.

Actions

A rule's action field defines what is the destination of the messages. This destination is usually a log file, but it can also be used to forward the message to a logging server by way of the network.

Rules

A rule integrates a Selector and an Action, which defines how each facility is treated. How a default configuration handles `cron` related messages is shown in the following example:

```
cron.*          /var/log/cron
```

In this example, all `cron` message levels are forwarded to the `/var/log/cron` log file.

8.6.3 Server configuration

A complex server configuration is out of the scope of this book. The default configuration file of the service comes with definitions that split your files depending on the facility that is logged. Figure 8-5 shows a centralized `rsyslog` server receiving log copies from `rsyslog` clients.

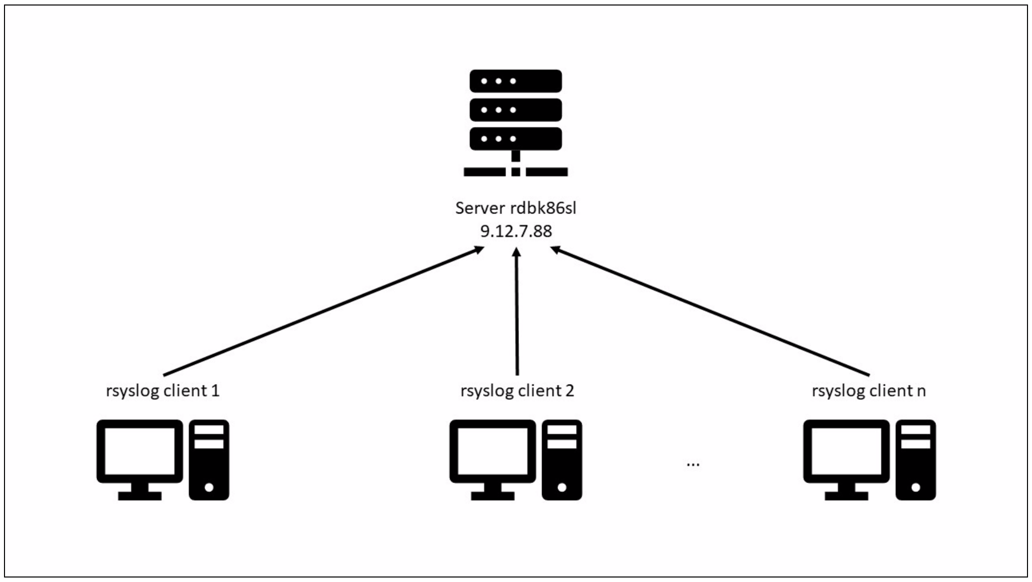


Figure 8-5 A centralized log server and some log clients

To enable the listener, define remote listen statements, as shown in Example 8-53.

Example 8-53 Source example for rsyslog server

```
# Load imudp module to receive syslog messages through UDP
$ModLoad imudp.so
# Start a UDP listener server on port 514
$UDPServerRun 514
```

Restart the syslog service to load the `rsyslog` configuration file, as shown in Example 8-54.

Example 8-54 Restarting the service to update the new configuration

```
rdbk86sl:~ # systemctl restart rsyslog
rdbk86sl:~ # systemctl status rsyslog
? rsyslog.service - System Logging Service
   Loaded: loaded (/usr/lib/systemd/system/rsyslog.service; enabled; vendor
preset: enabled)
   Active: active (running) since Tue 2020-10-20 12:53:35 EDT; 8s ago
     Docs: man:rsyslogd(8)
           http://www.rsyslog.com/doc/
  Process: 2455 ExecStartPre=/usr/sbin/rsyslog-service-prepare (code=exited,
status=0/SUCCESS)
 Main PID: 2457 (rsyslogd)
    Tasks: 6
   CGroup: /system.slice/rsyslog.service
           ··2457 /usr/sbin/rsyslogd -n -iNONE

Oct 20 12:53:35 rdbk86sl systemd[1]: Starting System Logging Service...
Oct 20 12:53:35 rdbk86sl rsyslogd[2457]: imuxsock: Acquired UNIX socket
'/run/systemd/journal/syslog' (fd 3) from systemd.  [v8.39.0]
Oct 20 12:53:35 rdbk86sl rsyslogd[2457]:  [origin software="rsyslogd"
swVersion="8.39.0" x-pid="2457" x-info="http://www.rsyslog.com"] start
Oct 20 12:53:35 rdbk86sl systemd[1]: Started System Logging Service.
rdbk86sl:~ #
```

To test the listener, run the **lsof** command, as shown in Example 8-55.

Example 8-55 Testing the listener with the lsof command

```
rdbk86sl:~ # lsof -i UDP:514
COMMAND   PID USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
rsyslogd 2457 root    6u  IPv4  32571      0t0  UDP *:syslog
rsyslogd 2457 root    7u  IPv6  32572      0t0  UDP *:syslog
```

You can also use TCP, but only the `udp` statement is needed to start a simple server, as shown in Example 8-53 on page 229.

8.6.4 Client configuration

To configure the rsyslog client, a destination must be created to point to the log server. In addition, a new log statement must be added to point the client source to the new destination. Create a `.conf` file under `/etc/rsyslog.d/` with the lines that are shown in Example 8-56 to configure the client by using your server's IP address.

Example 8-56 Client configuration to create a destination to the log server

```
root@rdbk86ub:~# cat /etc/rsyslog.d/99-send_remote.conf
*. *  @9.12.7.88:514
root@rdbk86ub:~#
```

In this example, all logs from the client are also sent to `9.12.7.88` (`rdbk86sl`) through UDP port `514`. This configuration is simple, but you can set up filters, new destinations, and sources, depending on the requirement of your environment.

Restart your `rsyslog` client as shown in Example 8-57.

Example 8-57 Restarting the rsyslog client to update the configuration

```
root@rdbk86ub:~# systemctl restart rsyslog
root@rdbk86ub:~#
```

8.6.5 Testing rsyslog

From the log server (`rdbk86ub`), you can run the **tail -f** command to monitor the messages that are written to the `/var/log/message` file (see Example 8-58).

Example 8-58 Monitoring the rsyslog server

```
rdbk86sl:~ # tail -f /var/log/messages
2020-10-20T13:07:00-04:00 rdbk86ub systemd[1]: rsyslog.service: Succeeded.
2020-10-20T13:07:00-04:00 rdbk86ub systemd[1]: Stopped System Logging Service.
2020-10-20T13:07:00-04:00 rdbk86ub systemd[1]: Starting System Logging Service...
```

To test the client, run the **logger** command:

```
logger "Testing syslog"
```

Example 8-59 shows the results from the log server (rdbk86sl).

Example 8-59 Getting logs from the clients

```
rdbk86sl:~ # tail -1 /var/log/messages
2020-10-20T13:08:28-04:00 rdbk86ub lnxadmin[200951]: Testing syslog
rdbk86sl:~ #
```

For alternative setups, see the man pages for the product.

8.6.6 Migrating by using rsyslog

You can use `rsyslog` as a tool for your migration. You can set up a centralized log server to keep a copy of the log files for all the servers that you are migrating. With this configuration, if a problem happens on the server and you lose access, you can easily fetch information or error messages.

If you are migrating a `rsyslog` client/server, check whether `rsyslog` is installed on the target server. Also, help ensure that the former configuration file is compatible with the version available on the target distribution. To migrate the old data, you can use an LVM snapshot to transfer the LV to the new server. Other commands, such as **tar** and **rsync**, can be used to transfer the old log files. For more information about a practical example of LVM snapshot, **tar**, and **rsync**, see *Set up Linux on IBM System z for Production*, SG24-8137.

8.7 Deploying Samba

Samba is an open software suite that runs the Server Message Block (SMB) protocol over the Internet Protocol network and provides seamless file and print services to users. Although several similar commercial products are available, Samba is the implementation that is most commonly used in Linux environments to share files and printers. It is available for LinuxONE from all distributions and allows interoperability between UNIX/Linux servers and Windows/Linux based clients. Samba runs easily on LinuxONE because its hardware has fast I/O that provides high-performance access to applications and files.

Before you deploy Samba, help ensure that appropriate analysis and planning was performed before any migration activity. The checklists that are provided in this book help identify the many areas to take into consideration to help prevent problems during migration.

This example assumes that the z/VM guest is set up and a minimal Linux operating system is installed. The Linux guest is named `rdbk86sl`, and includes SUSE Linux Enterprise Server 15 SP2 installed with one virtual CPU and 1 GB of virtual memory.

Like LDAP, a Samba server typically does not require a large amount of CPU or RAM to run on LinuxONE. It is presumed that an adequate RPM repository installation source is set up and available for the installation of the application software.

More extensive documentation about Samba is available at this website.

This example is a stand-alone server with a local, non-replicated directory service. Migrating a Samba installation on x86 to LinuxONE should be straightforward.

8.7.1 Installing the Samba software

Installing the software is relatively simple:

- To install Samba and its dependencies packages on SUSE Linux Enterprise Server, run the following command:

  `zypper install samba`

- To install Samba and its dependencies packages on RHEL, run the following command:

  `yum install samba`

- To install Samba and its dependencies packages on Ubuntu, run the following command:

  `apt install samba`

After the installation is complete, see your distribution's manual to create any firewall rules necessary to serve a Samba instance.

8.7.2 Configuring Samba

This section describes how to configure the Samba server.

Configuring a file server on SAMBA

The Samba server is straightforward in its most basic configuration. We start by adding a section to the server configuration file `/etc/samba/smb.conf`, as shown in Example 8-60.

Example 8-60 Initial configuration of Samba

```
[migration]
        comment = This is a sample share that is used for migration.
        path = /migrationdata
        read only = yes
```

After the initial installation step is completed, restart and enable the service, as shown in Example 8-61.

Example 8-61 Start and enable the samba service

```
rdbk86sl:~ # systemctl enable --now smb
Created symlink /etc/systemd/system/multi-user.target.wants/smb.service ·
/usr/lib/systemd/system/smb.service.
rdbk86sl:~ #
```

8.7.3 Mounting Samba shares

After completing the previous task, you should be able to map to the server from your client machine. As seen in Example 8-62, we mount the share that is named `migration` from server `rdbk86sl` as a local mount point.

Example 8-62 Mounting the Samba share from other server

```
root@rdbk86ub:~# mount -t cifs //rdbk86sl/migration /localmigration
Password for root@//rdbk86sl/migration:
root@rdbk86ub:~#
```

The client now can read data on the share, as shown in Example 8-63.

Example 8-63 Reading the data on the share

```
root@rdbk86ub:/localmigration# ls -l
total 102400
-rwxr-xr-x 1 root root 104857600 Oct 20 13:48 database_export.tar.gz
root@rdbk86ub:/localmigration#
```

Note: If you intend to use basic Linux authentication, that is, by using the `passwd` file, you must change the Samba user password by running the **`smbpasswd -a <userid>`** command.

LDAP settings for Samba

Many companies use LDAP to provide a single signon, where a password for a user is shared between services. The activities to create a working configuration are facilitated by the OpenLDAP server YaST module. Although the LDAP configuration on SAMBA is an important feature, the configuration is out of the scope of this book.

For more information about how to set up LDAP on Samba, see this website.

Configuration files

You can manually set up configuration files for Samba. The main configuration file on SUSE Linux Enterprise Server is stored in `/etc/samba/smb.conf`, and has two sections:

- `[ global ]` for general settings
- `[ share ]` to specify specific settings about sharing files and printers

For more information about Samba configuration, see your distribution's documentation.

Note: RHEL and Ubuntu use the similar structures as SUSE Linux Enterprise Server 15 for the Samba main configuration files.

8.8 Deploying ELK

Elasticsearch, Logstash, and Kibana (ELK) are a trio of open-source tools that work together to collect, process, and analyze large amounts of data. Each component plays a specific role:

- Elasticsearch: A powerful search engine that is built for speed and scalability, it stores and indexes data in a way that allows for fast and efficient retrieval.
- Logstash: A data pipeline that collects data from various sources, parses it into a structured format, and enriches it with more information before sending it to Elasticsearch.
- Kibana: A data visualization platform that you can use to visualize and explore the data that is stored in Elasticsearch, which offers insights and patterns through interactive dashboards and charts.

Figure 8-6 shows the classic ELK stack architecture.

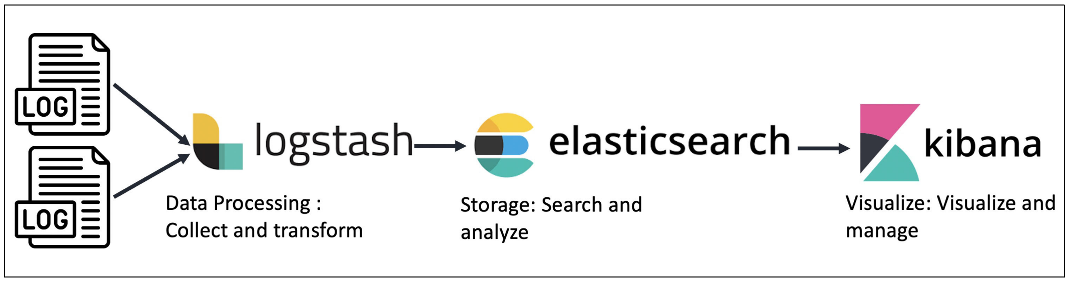


Figure 8-6 Classic ELK stack architecture

The ELK stack can be installed through various methods:

- Docker: Fast and scalable, but it requires some Docker knowledge.
- Source installation: Offers full control and potentially lower resource usage, but is more complex to set up.
- Package managers: Simple installation and updates, but has limited customization.

Choose the method that best suits your needs for ease of use, control, and resource efficiency.

8.8.1 Working environment

This section describes the process of building and installing the ELK stack on RHEL 9.3 (S/390). We also highlight ELK deployment by using Docker files.

Before proceeding, help ensure that the Docker packages are installed on your system, as described in 8.2, “Setting up Docker” on page 195, and that the current user belongs to the “docker” group.

Complete the following steps:

1. Verify the RHEL version, as shown in Example 8-64.

Example 8-64 Verifying the Red Hat Enterprise Linux version

```
[root@ysl-lx1 etc]# hostnamectl
 Static hostname: ysl-lx1
       Icon name: computer-vm
         Chassis: vm ?
      Machine ID: 7f70fd2c21b64de6b13cfec975d657da
         Boot ID: cbf3796461de470e9e007c2421a3b94c
  Virtualization: zvm
Operating System: Red Hat Enterprise Linux 9.3 (Plow)
     CPE OS Name: cpe:/o:Red Hat:enterprise_linux:9::baseos
          Kernel: Linux 5.14.0-362.8.1.el9_3.s390x
    Architecture: s390x
[root@ysl-lx1 etc]#
```

2. Verify the Docker version, as shown in Example 8-65.

Example 8-65 Vefrifying the Docker version

```
[root@ysl-lx1 etc]# docker version
Client: Docker Engine - Community
```

```
 Version:          25.0.3
 API version:      1.44
 Go version:       go1.21.6
 Git commit:       4debf41
 Built:            Fri Feb  9 07:00:46 2024
 OS/Arch:          linux/s390x
 Context:          default

Server: Docker Engine - Community
 Engine:
  Version:         25.0.3
  API version:     1.44 (minimum version 1.24)
  Go version:      go1.21.6
  Git commit:      f417435
  Built:           Fri Feb  9 06:57:18 2024
  OS/Arch:         linux/s390x
  Experimental:    false
 containerd:
  Version:         1.6.28
  GitCommit:       ae07eda36dd25f8a1b98dfbf587313b99c0190bb
 runc:
  Version:         1.1.12
  GitCommit:       v1.1.12-0-g51d5e94
 docker-init:
  Version:         0.19.0
  GitCommit:       de40ad0
[root@ysl-lx1 etc]#
```

Note: The installation and configuration that are outlined in this section were performed in our working environment. For more information, see Elastic Documentation.

8.8.2 Building and installing ELK by using building scripts

In this section, we build and install the ELK stack by using ELK building scripts. The first thing that you do is install the wget tool for downloading files, as shown in Example 8-66.

Example 8-66 Installing the wget tool

```
[root@ysl-lx1 ~]# sudo yum install wget
```

Building and installing Elasticsearch

To build and install Elasticsearch, complete the following steps:

1. Use a script to compile the source code of Elasticsearch, as shown in Example 8-67.

Example 8-67 Compiling the source code of Elasticsearch

```
[root@ysl-lx1 ~]# wget -q
https://raw.githubusercontent.com/linux-on-ibm-z/scripts/master/Elasticsearch/8.11
.1/build_elasticsearch.sh

[root@ysl-lx1 ~] # bash build_elasticsearch.sh
```

```
Detected Red Hat Enterprise Linux 9.3 (Plow)
Request details : PACKAGE NAME= elasticsearch , VERSION= 8.11.1
Docker : Yes
Docker Compose : Yes
User root belongs to the group docker
As part of the installation, dependencies would be installed/upgraded.
Do you want to continue (y/n) ? :  y
Installing elasticsearch 8.11.1 for rhel-9.3 and Temurin17
Installing dependencies... it may take some time.
Updating Subscription Management repositories.
Unable to read consumer identity

This system is not registered with an entitlement server. You can use
subscription-manager to register.

Last metadata expiration check: 0:07:38 ago on Sat 09 Mar 2024 05:04:30 AM CST.
Package curl-7.76.1-26.el9_3.3.s390x is already installed.
Package git-2.39.3-1.el9_2.s390x is already installed.
Package gzip-1.12-1.el9.s390x is already installed.
Package tar-2:1.34-6.el9_1.s390x is already installed.
Package wget-1.21.1-7.el9.s390x is already installed.
Package patch-2.7.6-16.el9.s390x is already installed.
Package make-1:4.3-7.el9.s390x is already installed.
Package gcc-11.4.1-2.1.el9.s390x is already installed.
Package gcc-c++-11.4.1-2.1.el9.s390x is already installed.
Dependencies resolved.
Nothing to do.
Complete!

Configuration and Installation started
Download and install Java
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100  171M  100  171M    0     0  85.1M      0  0:00:02  0:00:02 --:--:–  101M

(…)

Building Elasticsearch
Build might take some time. Sit back and relax
NOTE: Picked up JDK_JAVA_OPTIONS: --illegal-access=permit
Picked up JAVA_TOOL_OPTIONS: -Dfile.encoding=UTF8


> Task :distribution:docker:builds390xDockerImage

> Task :distribution:docker:docker-s390x-export:exports390xDockerImage
> Task :distribution:docker:docker-s390x-export:assemble

BUILD SUCCESSFUL in 2m 18s
536 actionable tasks: 8 executed, 528 up-to-date
Created docker distribution.
```

```
Installing Elasticsearch

Creating group elastic.
elasticsearch installation completed.

*********************************************************************************
******************

* Getting Started *
Note: Environment Variables that are needed were added to /root/setenv.sh
      To set the Environment Variables needed for Elasticsearch, run: source
/root/setenv.sh


Start Elasticsearch by using the following command: elasticsearch
*********************************************************************************
******************
[root@ysl-lx1 ~]#
```

2. Install Elasticsearch, as shown in Example 8-68.

Example 8-68 Installing Elasticsearch

```
[root@ysl-lx1 ~]# sudo mkdir /usr/share/elasticsearch
[root@ysl-lx1 ~]# sudo tar -xzf
distribution/archives/linux-s390x-tar/build/distributions/elasticsearch-8.11.1-SNA
PSHOT-linux-s390x.tar.gz -C /usr/share/elasticsearch --strip-components 1
[root@ysl-lx1 ~]# sudo ln -sf /usr/share/elasticsearch/bin/* /usr/bin/
[root@ysl-lx1 ~]# sudo /usr/sbin/groupadd elastic
[root@ysl-lx1 ~]# sudo chown "$(whoami)":elastic -R /usr/share/elasticsearch/
```

3. Modify the configuration within the container (potentially through configuration files that are mounted into the container) to turn off the **`xpack.ml`** feature, as shown in Example 8-69.

Example 8-69 Turning off xpack.ml

```
[root@ysl-lx1 ~]# sudo echo 'xpack.ml.enabled: false' >>
/usr/share/elasticsearch/config/elasticsearch.yml
```

4. Check the Elasticsearch version, as shown in Example 8-70.

Example 8-70 Checking the Elasticsearch version

```
[root@ysl-lx1 ~]# elasticsearch --version
Version: 8.11.1-SNAPSHOT, Build:
tar/6f9ff581fbcde658e6f69d6ce03050f060d1fd0c/2024-03-05T23:11:36.762543278Z, JVM:
17.0.9
```

5. Start the Elasticsearch server process, as shown in Example 8-71.

Example 8-71 Checking the Elasticsearch version

```
[linux1@ysl-lx1 ~]# elasticsearch &
```

Note: You cannot run **`elasticsearch`** as root.

6. To help ensure correct Elasticsearch startup, check the log file at `/usr/share/elasticsearch/logs/elasticsearch.log` for any errors after starting the service.
7. Access the Elasticsearch web interface by using the provided URL https://ysl-lx1:9200 in a web browser. With this interface, you can interact with Elasticsearch and manage your data.

Note: The typical installation of Elasticsearch might take several minutes to finalize. As a best practice, check the logs for any errors or more information during the process.

Building and installing Logstash

To build and install Logstash, complete the following steps:

1. Install OpenJDK 17 and help ensure that it is correctly installed, as shown in Example 8-72.

Example 8-72 Installing OpenJDK 17

```
[root@ysl-lx1 ~]# sudo yum install -y java-17-openjdk-devel
```

2. Verify the Java version, as shown in Example 8-73.

Example 8-73 Verifying the Java version

```
[root@ysl-lx1 ~]# java -version
openjdk version "17.0.9" 2023-10-17
OpenJDK Runtime Environment Temurin-17.0.9+9 (build 17.0.9+9)
OpenJDK 64-Bit Server VM Temurin-17.0.9+9 (build 17.0.9+9, mixed mode, sharing)
[root@ysl-lx1 ~]#
```

3. Set `LS_JAVA_HOME` to OpenJDK 17, as shown in Example 8-74.

Example 8-74 Setting LS_JAVA_HOME

```
[root@ysl-lx1 ~]# rpm -qi -l java-17-openjdk-17.0.10.0.7-2.el9.s390x
Name        : java-17-openjdk
Epoch       : 1
Version     : 17.0.10.0.7
Release     : 2.el9
Architecture: s390x
Install Date: Thu 07 Mar 2024 04:10:46 AM CST
Group       : Development/Languages
Size        : 1180099
License     : ASL 1.1 and ASL 2.0 and BSD and BSD with advertising and GPL+ and
GPLv2 and GPLv2 with exceptions and IJG and LGPLv2+ and MIT and MPLv2.0 and Public
Domain and W3C and zlib and ISC and FTL and RSA
Signature   : RSA/SHA256, Fri 12 Jan 2024 10:25:04 PM CST, Key ID 199e2f91fd431d51
Source RPM  : java-17-openjdk-17.0.10.0.7-2.el9.src.rpm
Build Date  : Fri 12 Jan 2024 03:01:25 PM CST
Build Host  : s390x-005.build.eng.rdu2.Red Hat.com
Packager    : Red Hat, Inc. <http://bugzilla.Red Hat.com/bugzilla>
Vendor      : Red Hat, Inc.
URL         : http://openjdk.java.net/
Summary     : OpenJDK 17 Runtime Environment
Description :
```

```
The OpenJDK 17 runtime environment.
/usr/lib/.build-id
/usr/lib/.build-id/1f
/usr/lib/.build-id/1f/6d19a67fd41e996236c7111a0ac93e0f5c3a87
/usr/lib/.build-id/73
/usr/lib/.build-id/73/13f35eaa85e7879b24349b3ba932f470aa1c1c
/usr/lib/.build-id/86
/usr/lib/.build-id/86/e1dfa07724e44876c0ef07048155afc1cbce24
/usr/lib/jvm/java-17-openjdk-17.0.10.0.7-2.el9.s390x/lib/libawt_xawt.so
/usr/lib/jvm/java-17-openjdk-17.0.10.0.7-2.el9.s390x/lib/libjawt.so
/usr/lib/jvm/java-17-openjdk-17.0.10.0.7-2.el9.s390x/lib/libsplashscreen.so
/usr/share/icons/hicolor/16x16/apps/java-17-openjdk.png
/usr/share/icons/hicolor/24x24/apps/java-17-openjdk.png
/usr/share/icons/hicolor/32x32/apps/java-17-openjdk.png
/usr/share/icons/hicolor/48x48/apps/java-17-openjdk.png
[root@ysl-lx1 bin]# export
LS_JAVA_HOME=/usr/lib/jvm/jre-17-openjdk-17.0.10.0.7-2.el9.s390x/
[root@ysl-lx1 bin]# export PATH=$LS_JAVA_HOME/bin:$PATH
```

4. Use a script to compile the source code of Logstash, as shown in Example 8-75.

Example 8-75 Compiling the source code of Logstash

```
[root@ysl-lx1 bin]# wget -q
https://raw.githubusercontent.com/linux-on-ibm-z/scripts/master/Logstash/8.12.2/bu
ild_logstash.sh
[root@ysl-lx1 ~]# bash build_logstash.sh
Sudo : Yes

As part of the installation, dependencies would be installed/upgraded.
Do you want to continue (y/n) ? :  y
The user responded with Yes.
Detected Red Hat Enterprise Linux 9.3 (Plow)
Request details : PACKAGE NAME= logstash , VERSION= 8.12.2
Installing logstash 8.12.2 for rhel-9.3
Updating Subscription Management repositories.
Unable to read consumer identity

This system is not registered with an entitlement server. You can use
subscription-manager to register.

Last metadata expiration check: 2:05:21 ago on Wed 06 Mar 2024 12:29:06 AM CST.
Package gcc-11.4.1-2.1.el9.s390x is already installed.
Package make-1:4.3-7.el9.s390x is already installed.
Package tar-2:1.34-6.el9_1.s390x is already installed.
Package wget-1.21.1-7.el9.s390x is already installed.
Dependencies resolved.
Nothing to do.
Complete!
Configuration and Installation started

Installing OpenJDK 11 . . .
Updating Subscription Management repositories.
Unable to read consumer identity
```

```
This system is not registered with an entitlement server. You can use
subscription-manager to register.

Last metadata expiration check: 2:05:22 ago on Wed 06 Mar 2024 12:29:06 AM CST.
Dependencies resolved.
==================================================================================
===================
 Package                              Arch    Version
Repository       Size
==================================================================================
===================
Installing:
 java-11-openjdk-devel                s390x   1:11.0.22.0.7-2.el9
rhel9-appstream  3.3 M
Installing dependencies:
 ModemManager-glib                    s390x   1.20.2-1.el9
rhel9-base       330 k
 adobe-source-code-pro-fonts          noarch  2.030.1.050-12.el9.1
rhel9-base       836 k



 2024-03-06 02:35:10 (68.1 MB/s) - 'logstash-oss-8.12.2-linux-aarch64.tar.gz'
saved [345778933/345778933]

logstash installation completed. Check the Usage to start the service.

logstash -V
Using LS_JAVA_HOME defined java: /usr/lib/jvm/java-11-openjdk.
WARNING: Logstash comes bundled with the recommended JDK(Temurin-17.0.10+7), but
is overridden by the version that is defined in LS_JAVA_HOME. Consider clearing
LS_JAVA_HOME to use the bundled JDK.
logstash 8.12.2

**********************************************************************************
**********************

* Getting Started *
Note: Environmental Variables that are needed were added to /root/setenv.sh
Note: To set the Environmental Variables needed for Logstash, run: source
/root/setenv.sh
Run Logstash:
    logstash -V

For more information, see https://www.elastic.co/support/matrix#matrix_jvm.

**********************************************************************************
**********************
```

5. Install Logstash, as shown in Example 8-76.

Example 8-76   Installing Logstash

```
[root@ysl-lx1 ~]# wget
https://artifacts.elastic.co/downloads/logstash/logstash-oss-8.12.2-linux-aarch64.
tar.gz
```

```
[root@ysl-lx1 ~] sudo mkdir /usr/share/logstash
[root@ysl-lx1 ~] sudo tar -xzf logstash-oss-8.12.2-linux-aarch64.tar.gz -C
/usr/share/logstash --strip-components 1
[root@ysl-lx1 ~] sudo ln -sf /usr/share/logstash/bin/* /usr/bin
[root@ysl-lx1 ~] sudo groupadd elastic
[root@ysl-lx1 ~] sudo chown "$(whoami)":elastic -R /usr/share/logstash/
```

6. Check the Logstash version, as shown in Example 8-77.

Example 8-77 Checking the Logstash version

```
[root@ysl-lx1 bin]# logstash -V
Using LS_JAVA_HOME defined java:
/usr/lib/jvm/jre-17-openjdk-17.0.10.0.7-2.el9.s390x/.
WARNING: Logstash comes bundled with the recommended JDK(Temurin-17.0.10+7), but
is overridden by the version that is defined in LS_JAVA_HOME. Consider clearing
LS_JAVA_HOME to use the bundled JDK.
```

Building and installing Kibana

To build and install Kibana, complete the following steps:

1. Install `lcov`, as shown in Example 8-78.

Example 8-78 Installing lcov

```
[root@ysl-lx1 ~]# git clone https://github.com/linux-test-project/lcov.git
Cloning into 'lcov'...
remote: Enumerating objects: 4711, done.
remote: Counting objects: 100% (1040/1040), done.
remote: Compressing objects: 100% (144/144), done.
remote: Total 4711 (delta 917), reused 899 (delta 895), pack-reused 3671
Receiving objects: 100% (4711/4711), 2.01 MiB | 10.96 MiB/s, done.
Resolving deltas: 100% (3436/3436), done.
[root@ysl-lx1 ~]# cd lcov
[root@ysl-lx1 lcov]# make install
```

2. Verify that the correct `ncat` version is installed, as shown in Example 8-79.

Example 8-79 Checking the ncat version

```
[root@ysl-lx1 ~]# rpm -qa | grep -i nmap-ncat
nmap-ncat-7.92-1.el9.s390x
[root@ysl-lx1 ~]# ncat --version
Ncat: Version 7.92 ( https://nmap.org/ncat )
```

3. Use a script to compile the source code of Kibana, as shown in Example 8-80.

Example 8-80 Compiling the source code of Kibana

```
[root@ysl-lx1 ~]# wget -q
https://raw.githubusercontent.com/linux-on-ibm-z/scripts/master/Kibana/8.11.0/buil
d_kibana.sh
# Build Kibana
[root@ysl-lx1 ~]# bash build_kibana.sh
```

Note: The `ncat` networking utility replaces `netcat` in RHEL 7. Help ensure that you replace `netcat` by using `nmap-ncat` in the **`build_kibana.sh`** script.

4. Install Kibana, as shown in Example 8-81.

Example 8-81 Installing Kibana

```
[root@ysl-lx1 ~]# sudo mkdir /usr/share/kibana/
[root@ysl-lx1 ~]# sudo tar -xzf target/kibana-8.11.0-SNAPSHOT-linux-s390x.tar.gz
-C /usr/share/kibana --strip-components 1
[root@ysl-lx1 ~]# sudo ln -sf /usr/share/kibana/bin/* /usr/bin/
[root@ysl-lx1 ~]# sudo chown "$(whoami)":elastic -R /usr/share/kibana/
```

5. Access the Kibana web interface by using the URL https://ysl-lx1:5601 in a web browser.

8.8.3 Building and installing the ELK stack by using a Dockerfile

In this section, you build and install the ELK stack by using an ELK Dockerfile.

Building a Logstash by using a Dockerfile

To build a Logstash container by using a Dockerfile, complete the following steps:

1. Download the Logstash Dockerfile from GitHub.
2. Browse to the directory that contains the Dockerfile, as shown in Example 8-82.

Example 8-82 Logstash directory structure

```
[root@ylsprd logstash]# ls -alrt
total 40
dr-xr-x---. 14 root root 4096 Mar  9 06:30 ..
-rw-r--r--.  1 root root 3733 Mar  9 06:32 Dockerfile
drwxr-xr-x.  2 root root 4096 Mar  9 06:33 bin
drwxr-xr-x.  2 root root 4096 Mar  9 06:36 config
drwxr-xr-x.  2 root root 4096 Mar  9 06:37 env2yaml
drwxr-xr-x.  2 root root 4096 Mar  9 06:38 pipeline
-rw-r--r--.  1 root root 4728 Mar  9 06:38 dockerfile_netty_tcnative
-rw-r--r--.  1 root root 3818 Mar  9 06:38 dockerfile_openssl_dynamic
drwxr-xr-x.  6 root root 4096 Mar  9 06:38 .
[root@ylsprd logstash]#
```

3. Build the Logstash Docker image, as shown in Example 8-83.

Example 8-83 Building the Logstash Docker image

```
[root@ylsprd logstash]# docker build -t ysllogstash .
[+] Building 43.5s (21/21) FINISHED
docker:default
 => [internal] load build definition from Dockerfile 0.0s
 => => transferring dockerfile: 3.83kB 0.0s
 => [internal] load metadata for docker.io/library/golang:1.15 0.6s
 => [internal] load metadata for docker.io/s390x/ubuntu:20.04 0.7s
 => [internal] load .dockerignore 0.0s
 => => transferring context: 2B 0.0s
 => CACHED [stage-1  1/12] FROM
docker.io/s390x/ubuntu:20.04@sha256:ec4322e32bcfa82398237d2c6c302aff8888667d80be51
0.0s
 => => resolve
docker.io/s390x/ubuntu:20.04@sha256:ec4322e32bcfa82398237d2c6c302aff8888667d80be51
ee3c91499e868912f  0.0s
 => [internal] load build context 0.1s
 => => transferring context: 7.18kB 0.0s
 => [builder 1/3] FROM
docker.io/library/golang:1.15@sha256:ea080cc817b02a946461d42c02891bf750e3916c52f7e
a8187bcc  15.4s
 => => resolve
docker.io/library/golang:1.15@sha256:ea080cc817b02a946461d42c02891bf750e3916c52f7e
a8187bccde8f312b5  0.1s
 => => sha256:ea080cc817b02a946461d42c02891bf750e3916c52f7ea8187bccde8f312b59f
2.36kB / 2.36kB                    0.0s
 => => sha256:9340537ea284f199de609d1636190c5d0649b97aa16991a92c00b431a4c51875
7.00kB / 7.00kB                    0.0s
 => => sha256:a6c0758de0b056dd898de174a841d1ee246d0c7a0d62744c7b32e4ac60dde133
1.79kB / 1.79kB                    0.0s
 => => sha256:8776c37054ddd28ba8fe526b8dbc08bf7b521ae9ddb8eacd008d05185571cde0
49.00MB / 49.00MB                  1.7s
 => => sha256:3c06f635d71fe906f56e25278d5659aa940f095c1ecf199e41436e197534383a
7.40MB / 7.40MB                    1.9s
 => => sha256:1f57129f34059279bed7373d2b35f16c94df2418b85c5ae20dc715f2e1999498
9.88MB / 9.88MB                    0.9s
 => => sha256:e2f6521600d1dbe8e9e84511427b15a7cb2dd42039504bb9679b6ed7e7ee0dca
51.38MB / 51.38MB                  2.8s
 => => extracting
sha256:8776c37054ddd28ba8fe526b8dbc08bf7b521ae9ddb8eacd008d05185571cde0 1.7s
 => => sha256:e6a3d49530f5e2c2c0d44738819635926fd50686b934183262b7f33d136e4f76
101.20MB / 101.20MB                8.7s
 => => sha256:1228c41a068ba9c3aca67ff810dadc0c8a62694a090ea82e610e8728cfbbb031
56.82MB / 56.82MB                  6.2s
 => => sha256:18b69ebcfa1034a0b57807a50f21af2e6d1a87c777459d3184a90ff7fcdf52ee
155B / 155B                        2.9s
 => => extracting
sha256:3c06f635d71fe906f56e25278d5659aa940f095c1ecf199e41436e197534383a 0.3s
 => => extracting
sha256:1f57129f34059279bed7373d2b35f16c94df2418b85c5ae20dc715f2e1999498 0.3s
 => => extracting
sha256:e2f6521600d1dbe8e9e84511427b15a7cb2dd42039504bb9679b6ed7e7ee0dca 1.8s
 => => extracting
sha256:1228c41a068ba9c3aca67ff810dadc0c8a62694a090ea82e610e8728cfbbb031 1.6s
 => => extracting
sha256:e6a3d49530f5e2c2c0d44738819635926fd50686b934183262b7f33d136e4f76 4.6s
 => => extracting
sha256:18b69ebcfa1034a0b57807a50f21af2e6d1a87c777459d3184a90ff7fcdf52ee 0.0s
 => [stage-1  2/12] RUN export DEBIAN_FRONTEND=noninteractive &&     apt-get
update &&     apt-get install -y pro  38.2s
 => [builder 2/3] ADD env2yaml/env2yaml.go /go/ 4.5s
 => [builder 3/3] RUN go get gopkg.in/yaml.v2 && go build /go/env2yaml.go 3.2s
 => [stage-1  3/12] WORKDIR /usr/share/logstash 0.0s
 => [stage-1  4/12] ADD config/pipelines.yml config/pipelines.yml 0.1s
 => [stage-1  5/12] ADD config/logstash-oss.yml config/logstash.yml 0.0s
 => [stage-1  6/12] ADD config/log4j2.properties config/ 0.1s
 => [stage-1  7/12] ADD pipeline/default.conf pipeline/logstash.conf 0.0s
 => [stage-1  8/12] RUN chown --recursive logstash:root config/ pipeline/ 0.3s
 => [stage-1  9/12] RUN locale-gen en_US.UTF-8 1.3s
 => [stage-1 10/12] ADD bin/docker-entrypoint /usr/local/bin/ 0.1s
 => [stage-1 11/12] RUN chmod 0755 /usr/local/bin/docker-entrypoint 0.4s
 => [stage-1 12/12] COPY --from=builder /go/env2yaml /usr/local/bin/ 0.1s
 => exporting to image 2.0s
 => => exporting layers 2.0s
 => => writing image
sha256:c543c2f45ba49969607832451c3babd0d8efef514338302c9477142a87010d54 0.0s
 => => naming to docker.io/library/ysllogstash 0.0s
[root@ylsprd logstash]#
```

4. Verify the newly created Docker image, as shown in Example 8-84.

Example 8-84 Verifying the Docker images

```
[root@ylsprd logstash]# docker image list
REPOSITORY                                      TAG               IMAGE ID
CREATED            SIZE
ysllogstash latest c543c2f45ba4   41 seconds ago 684MB
docker.elastic.co/elasticsearch/elasticsearch 8.11.1-SNAPSHOT 87f974c1cf2e 48
minutes ago 591MB
elasticsearch 8.11.1-SNAPSHOT   87f974c1cf2e   48 minutes ago 591MB
elasticsearch s390x 87f974c1cf2e   48 minutes ago 591MB
elasticsearch test 87f974c1cf2e   48 minutes ago 591MB
osixia/openldap 1.4.0 b77e8b39f29a   About an hour ago 265MB
osixia/light-baseimage 1.2.0 346bc72577ad   About an hour ago 141MB
yslterraform1 latest 258507e39601   3 hours ago 104MB
yslterraform latest 54d5f158756c   13 hours ago 104MB
ubuntu 20.04 1b2a45ccd935   3 weeks ago 69.2MB
hello-world latest 37fc523148ea   10 months ago 9.06kB
```

5. Run the Logstash Docker container, as shown in Example 8-85.

Example 8-85 Running the Logstash Docker container

```
[root@ylsprd logstash]# docker run --name ysllogstashcontainer  -d ysllogstash
a132b7b8df99d413f674cbd864af33507faba53f9910786fe001471d22acb18b
```

6. Verify that the Logstash container is running, as shown in Example 8-86.

Example 8-86 Verifying that the Logstash Docker container is running

```
[root@ylsprd logstash]# docker ps
CONTAINER ID   IMAGE COMMAND CREATED STATUS PORTS NAMES
a132b7b8df99   ysllogstash   "/usr/local/bin/dock…"   11 seconds ago   Up 9
seconds   5044/tcp, 9600/tcp   ysllogstashcontainer
[root@ylsprd logstash]#
```

Building an Elasticsearch container by using a Dockerfile

To build an Elasticsearch container by using a Dockerfile, complete the following steps:

1. To create an Elasticsearch Dockerfile, see Building Elasticsearch. You can incorporate the manual setup steps into a Dockerfile similar to the example that is shown in Example 8-87.

Example 8-87 Elasticsearch Dockerfile example

```
[root@yslprd elk]# vi Dockerfile
FROM openjdk:9-jdk
ENV SOURCE_ROOT /<source_root>/
ENV GRADLE_USER_HOME $SOURCE_ROOT/.gradle

RUN apt-get update && apt-get install -y \
    curl \
    git \
    gzip \
    tar \
    wget \
    patch \
    locales \
    make \
    gcc \
    g++ \
 && locale-gen en_US.UTF-8 \
 && apt-get install -y openjdk-17-jdk \
 && export LANG="en_US.UTF-8" \
 && export JAVA_HOME=/<Path to JDK> \
 && export ES_JAVA_HOME=/<Path to JDK>/ \
 && export JAVA17_HOME=/<Path to JDK>/ \
 && export PATH=$ES_JAVA_HOME/bin:$PATH \
 && cd $SOURCE_ROOT \
 && git clone https://github.com/fusesource/jansi.git \
 && cd jansi \
 && git checkout jansi-2.4.0 \
 && make clean-native native OS_NAME=Linux OS_ARCH=s390x \
 && mkdir -p $SOURCE_ROOT/jansi-jar \
 && cd $SOURCE_ROOT/jansi-jar \
 && wget
https://repo1.maven.org/maven2/org/fusesource/jansi/jansi/2.4.0/jansi-2.4.0.jar \
 && jar xvf jansi-2.4.0.jar \
 && cd org/fusesource/jansi/internal/native/Linux \
 && mkdir s390x \
 && cp $SOURCE_ROOT/jansi/target/native-Linux-s390x/libjansi.so s390x/ \
 && cd $SOURCE_ROOT/jansi-jar \
 && jar cvf jansi-2.4.0.jar . \
 && mkdir -p
$SOURCE_ROOT/.gradle/caches/modules-2/files-2.1/org.fusesource.jansi/jansi/2.4.0/3
21c614f85f1dea6bb08c1817c60d53b7f3552fd/ \
 && cp jansi-2.4.0.jar
$SOURCE_ROOT/.gradle/caches/modules-2/files-2.1/org.fusesource.jansi/jansi/2.4.0/3
21c614f85f1dea6bb08c1817c60d53b7f3552fd/ \
 && export sha256=$(sha256sum jansi-2.4.0.jar | awk '{print $1}') \
 && cd $SOURCE_ROOT \
 && export
PATCH_URL="https://raw.githubusercontent.com/linux-on-ibm-z/scripts/master/Elastic
search/8.11.1/patch" \
 && git clone https://github.com/osixia/docker-light-baseimage.git \
 && cd docker-light-baseimage/ \
 && git checkout v1.2.0 \
 && curl -sSL "${PATCH_URL}/docker-light-baseimage.patch" | git apply - \
 && make build \
 && cd $SOURCE_ROOT \
 && git clone https://github.com/osixia/docker-openldap.git \
 && cd docker-openldap/ \
 && git checkout v1.4.0 \
 && curl -sSL "${PATCH_URL}/docker-openldap.patch" | git apply - \
 && make build \
 && cd $SOURCE_ROOT \
 && git clone https://github.com/elastic/elasticsearch \
 && cd elasticsearch \
 && git checkout v8.11.1 \
 && export
PATCH_URL="https://raw.githubusercontent.com/linux-on-ibm-z/scripts/master/Elastic
search/8.11.1/patch/elasticsearch.patch" \
 && curl -o elasticsearch.patch $PATCH_URL \
 && git apply elasticsearch.patch \
 && sed -i
's|6cd91991323dd7b2fb28ca93d7ac12af5a86a2f53279e2b35827b30313fd0b9f|'"${sha256}"'|
g' $SOURCE_ROOT/elasticsearch/gradle/verification-metadata.xml \
 && mkdir -p
$SOURCE_ROOT/elasticsearch/distribution/docker/ubi-docker-s390x-export/ \
 && mkdir -p
$SOURCE_ROOT/elasticsearch/distribution/docker/cloud-docker-s390x-export/ \
 && mkdir -p
$SOURCE_ROOT/elasticsearch/distribution/docker/cloud-ess-docker-s390x-export/ \
 && mkdir -p $SOURCE_ROOT/elasticsearch/distribution/docker/docker-s390x-export/ \
 && mkdir -p
$SOURCE_ROOT/elasticsearch/distribution/docker/ironbank-docker-s390x-export/ \
 && cd $SOURCE_ROOT/elasticsearch \
 && CPU_NUM="$(grep -c ^processor /proc/cpuinfo)" \
 && export GRADLE_USER_HOME=$SOURCE_ROOT/.gradle \
 && ./gradlew :distribution:archives:linux-s390x-tar:assemble
--max-workers="$CPU_NUM" --parallel \
 && cd $SOURCE_ROOT/elasticsearch \
 && sudo mkdir /usr/share/elasticsearch \
 && sudo tar -xzf
distribution/archives/linux-s390x-tar/build/distributions/elasticsearch-8.
….
```

2. Help ensure that within the working directory that you create the `build.gradle` file for the build process.
3. Build the Elasticsearch Docker image, as shown in Example 8-88.

Example 8-88 Building the Elasticsearch Docker image

```
[root@ylsprd elk]# docker build -t yslelasticsearch .
```

4. Verify the newly created Docker image, as shown in Example 8-89 on page 247.

Example 8-89 Verifying the Docker image

```
[root@yslprd elk]# docker image list
REPOSITORY TAG              IMAGE ID       CREATED SIZE
yslelasticsearch s390x            3873dc10d865   47 minutes ago      591MB
elasticsearch test            3873dc10d865   47 minutes ago      591MB
osixia/openldap 1.4.0           5180f177b0ae   About an hour ago   265MB
osixia/light-baseimage 1.2.0 0fde6613d785   About an hour ago   141MB
ubuntu 20.04            1b2a45ccd935   3 weeks ago         69.2MB
```

5. Run the Elasticsearch Docker container, as shown in Example 8-90.

Example 8-90 Running the Elasticsearch Docker container

```
[root@yslprd elk]# docker run -d --name ysl-elasticsearch-container -p 9200:9200
-p 9300:9300 yslelasticsearch:s390x
4a899ae061c767288a909fb5465c446d9b799fc12744b88835a42aea8a3c2344
```

6. Verify that the Elasticsearch container is running, as shown in Example 8-91.

Example 8-91 Verifying that the Elasticsearch Docker container is running

```
[root@yslprd e]# docker ps
CONTAINER ID   IMAGE COMMAND                 CREATED        STATUS        PORTS
NAMES
4a899ae061c7   yslelasticsearch:s390x   "/bin/tini -- /usr/l…"   5 seconds ago
Up 4 seconds   0.0.0.0:9200->9200/tcp, :::9200->9200/tcp, 0.0.0.0:9300->9300/tcp,
:::9300->9300/tcp   ysl-elasticsearch-container
[root@yslprd elasticsearch]#
```

7. Retrieve the logs for the Elasticsearch container, as shown in Example 8-92.

Example 8-92 Retrieving the Elasticsearch logs

```
[root@yslprd elk]# docker logs 4a899ae061c7
```

Building a Kibana container by using a Dockerfile

To build a Kibana container by using a Dockerfile, complete the following steps:

1. Create a Kibana Dockerfile, as shown in Example 8-93.

Example 8-93 Creating a Kibana Dockerfile

```
[root@yslprd kibana]# vi Dockerfile

FROM ibmjava:8-sdk
WORKDIR "/root"

ENV PATH=/usr/share/node-v12.22.12/bin:/usr/share/kibana/bin:$PATH

RUN  apt-get update && apt-get install -y apache2 g++ gcc git make nodejs python3
unzip wget tar \
 && wget https://nodejs.org/dist/v12.22.12/node-v12.22.12-linux-s390x.tar.gz \
&& tar xvzf node-v12.22.12-linux-s390x.tar.gz \
 && mv /root/node-v12.22.12-linux-s390x/ /usr/share/node-v12.22.12 \
&& cd /root/ \
&& wget
https://artifacts.elastic.co/downloads/kibana/kibana-6.3.1-linux-x86_64.tar.gz \
&& tar xvf kibana-6.3.1-linux-x86_64.tar.gz \
&& mv /root/kibana-6.3.1-linux-x86_64 kibana-6.3.1 \
&& cd /root/kibana-6.3.1 \
&& mv node node_old \
&& ln -s /usr/share/node-v12.22.12/bin/node node \
&& mkdir /etc/kibana \
&& cp config/kibana.yml /etc/kibana \
&& mv /root/kibana-6.3.1/ /usr/share/kibana \
&& apt-get remove -y git make unzip  wget \
&& apt-get autoremove -y && apt-get clean \
&& rm -rf /root/kibana-6.3.1-linux-x86_64.tar.gz
/root/node-v12.22.12-linux-s390x.tar.gz \
&& rm -rf /var/lib/apt/lists/*

EXPOSE 5601 80

CMD ["kibana","-H","0.0.0.0"]
```

Note: For illustrative purposes, in the provided Dockerfile, we installed Kibana 6.3 along with Node.js 12. However, the compatibility between Kibana, Node.js, and other components can vary depending on the versions that are used. As a best practice, consult the official documentation of Kibana and Node.js to help ensure compatibility.

2. Build the Kibana Docker image, as shown in Example 8-94.

Example 8-94 Building the Kibana Docker image

```
[root@ylsprd kibana]# docker build -t yslkibana .
[+] Building 63.1s (7/7) FINISHED
docker:default
 => [internal] load build definition from Dockerfile 0.0s
 => => transferring dockerfile: 1.40kB 0.0s
 => [internal] load metadata for docker.io/library/ibmjava:8-sdk 0.1s
 => [internal] load .dockerignore 0.0s
 => => transferring context: 2B 0.0s
 => [1/3] FROM
docker.io/library/ibmjava:8-sdk@sha256:5f8532deba4d479734e71d9deeaff9c6e162ad0d995
8e68c142ec7e68764  0.0s
 => CACHED [2/3] WORKDIR /root 0.0s
 => [3/3] RUN  apt-get update && apt-get install -y apache2 g++ gcc git make
nodejs python3 unzip wget tar  && wg  43.2s
 => exporting to image 9.5s
 => => exporting layers 19.4s
 => => writing image
sha256:14bc1931b6b053f9e9a38360c809dbb0563d8116390802644f5c2b5fa732b200 0.0s
 => => naming to docker.io/library/yslkibana 0.0s
```

3. Verifying the newly created Docker image, as shown in Example 8-95.

Example 8-95 Verifying the Docker image

```
[root@yslprd kibana]# docker image list
REPOSITORY        TAG       IMAGE ID       CREATED          SIZE
yslkibana         latest    74a2e2c895c3   19 minutes ago   1.25GB
ysllogstash       latest    acd279e04202   7 hours ago      684MB
```

4. Run the Kibana Docker container, as shown in Example 8-96.

Example 8-96 Running the Kibana Docker container

```
[root@yslprd kibana]# docker run --name ysl-kibana-container -p 5601:5601 -d
yslkibana
```

5. Verify that the Kibana container is running, as shown in Example 8-97.

Example 8-97 Verifying that the Kibana Docker container is running

```
[root@yslprd kibana]# docker ps
CONTAINER ID   IMAGE        COMMAND                CREATED         STATUS
PORTS                                                NAMES
8e1c09f2e7f7   yslkibana     "kibana -H 0.0.0.0"      4 seconds ago    Up 3
seconds    80/tcp, 0.0.0.0:5601->5601/tcp, :::5601->5601/tcp
ysl-kibana-container
4ef6f600e729   ysllogstash   "/usr/local/bin/dock…"   18 minutes ago   Up 18
minutes   5044/tcp, 9600/tcp                                 ysllogstashcontainer
[root@yslprd kibana]#
```

6. Retrieve the logs for the Kibana container, as shown in Example 8-98.

Example 8-98 Retrieving the Kibana logs

```
[root@yslprd kibana]# docker logs 8e1c09f2e7f7
```

8.9 Deploying Terraform

HashiCorp Terraform is an open-source Infrastructure as Code (IaC) tool. By using it, users can define and provision infrastructure resources by using a declarative configuration language that Terraform converts into an execution plan, which outlines the actions it takes to achieve its goal. This plan can be reviewed before applying changes to the infrastructure.Once approved, Terraform applies the plan and provisions or updates the infrastructure. It manages the entire lifecycle of infrastructure resources from creation to modification to deletion.

To deploy Terraform by using a Dockerfile, complete the following steps:

1. Help ensure that Docker is running and ready to be used.
2. Download the Terraform Dockerfile, as shown in Example 8-99.

Example 8-99 Downloading the Terraform Dockerfile

```
[root@ylsprd prd]# sudo mkdir /root/Terraform
[root@ylsprd prd]# wget
https://raw.githubusercontent.com/linux-on-ibm-z/dockerfile-examples/master/Terraf
orm/Dockerfile
```

3. Update the Terraform Dockerfile to deploy the latest version, as shown in Example 8-100.

Example 8-100 Updating the Terraform Dockerfile

```
ARG TERRAFORM_VERSION=v1.7.4
```

4. Install Terraform by using the Dockerfile, as shown in Example 8-101.

Example 8-101 Installing Terraform

```
[root@ylsprd prd]# cd /root/Terraform
[root@ylsprd prd]# docker build -t ylsterraform .
[+] Building 300.5s (11/11) FINISHED
docker:default
 => [internal] load build definition from Dockerfile 0.0s
 => => transferring dockerfile: 1.94kB 0.0s
 => [internal] load metadata for docker.mirror.hashicorp.services/alpine:latest
2.2s
 => [internal] load metadata for docker.mirror.hashicorp.services/golang:alpine
2.5s
 => [internal] load .dockerignore 0.0s
 => => transferring context: 2B 0.0s
 => [builder 1/4] FROM
docker.mirror.hashicorp.services/golang:alpine@sha256:fc5e5848529786cf1136563452b3
3d713d5c60b2c787f6b2a077fa6eeefd9114 8.3s
 => => resolve
docker.mirror.hashicorp.services/golang:alpine@sha256:fc5e5848529786cf1136563452b3
3d713d5c60b2c787f6b2a077fa6eeefd9114 0.0s
 => => sha256:eb8fba61d86413beda3240c40c599041e040e658cd8314e38ee15e67ea57d349
3.24MB / 3.24MB 1.5s
 => => sha256:61981fca6cbfb460db1b5bca98f87e0d4bf4677082bf9a97e229d13fe4656d66
285.19kB / 285.19kB 0.8s
 => => sha256:3b54f06104ceea5cb6d57308b8530d1ad46f64fa50cbed93e75e79c2dfa17375
68.39MB / 68.39MB 4.3s
 => => sha256:fc5e5848529786cf1136563452b33d713d5c60b2c787f6b2a077fa6eeefd9114
1.65kB / 1.65kB 0.0s
 => =>
sha256:20d1b4c57bf0e1ca0f6f9e7b2b6c6376b55c281730c63d9116162059b04a98101.36kB
/1.36kB 0.0s
 => => sha256:f179f0fe0c7170534c5b43793d57a0f5464042b485e267b247d47cd8dc77133e
2.13kB / 2.13kB 0.0s
 => => sha256:87e02b54b6f2f5aa57c73eef9a1f62166c8eba59be5f0bb100c02d7254dc09e1
174B / 174B 1.1s
 => => sha256:4f4fb700ef54461cfa02571ae0db9a0dc1e0cdb5577484a6d75e68dc38e8acc1 32B
/ 32B 1.4s
```

```
 => => extracting
sha256:61981fca6cbfb460db1b5bca98f87e0d4bf4677082bf9a97e229d13fe4656d66 0.0s
 => => extracting
sha256:3b54f06104ceea5cb6d57308b8530d1ad46f64fa50cbed93e75e79c2dfa17375 2.9s
 => => extracting
sha256:87e02b54b6f2f5aa57c73eef9a1f62166c8eba59be5f0bb100c02d7254dc09e1 0.0s
 => => extracting
sha256:4f4fb700ef54461cfa02571ae0db9a0dc1e0cdb5577484a6d75e68dc38e8acc1 0.0s
 => [stage-1 1/3] FROM
docker.mirror.hashicorp.services/alpine:latest@sha256:c5b1261d6d3e43071626931fc004
f70149baeba2c8ec672bd4f27761f8e1ad6b 1.6s
 => => resolve
docker.mirror.hashicorp.services/alpine:latest@sha256:c5b1261d6d3e43071626931fc004
f70149baeba2c8ec672bd4f27761f8e1ad6b 0.0s
 => => sha256:eb8fba61d86413beda3240c40c599041e040e658cd8314e38ee15e67ea57d349
3.24MB / 3.24MB 1.5s
 => => sha256:c5b1261d6d3e43071626931fc004f70149baeba2c8ec672bd4f27761f8e1ad6b
1.64kB / 1.64kB 0.0s
 => => sha256:5d0da60400afb021f2d8dbfec8b7d26457e77eb8825cba90eba84319133f0efe
528B / 528B 0.0s
 => => sha256:8fc740d8c40e45ea330a3f324fe009148dfc1f771bc90254eaf8ff8bbcecfe02
1.47kB / 1.47kB 0.0s
 => => extracting
sha256:eb8fba61d86413beda3240c40c599041e040e658cd8314e38ee15e67ea57d349 0.1s
 => [builder 2/4] RUN apk add --no-cache git bash openssh     && git clone -b
v1.7.2 https://github.com/hashicorp/terraform.git
/go/src/github.com/hashicorp/terraform 23.0s
 => [builder 3/4] WORKDIR /go/src/github.com/hashicorp/terraform 0.0s
 => [builder 4/4] RUN /bin/bash ./scripts/build.sh 258.3s
 => [stage-1 2/3] COPY --from=builder go/bin /bin 1.3s
 => exporting to image 0.2s
 => => exporting layers 0.2s
 => => writing image
sha256:54d5f158756c1ef22abdda2b053a92f21e279d4f5f40a8361e1d86d9cc0e471c 0.0s
 => => naming to docker.io/library/yslterraform 0.0s
[root@ylsprd prd]#
[root@ylsprd prd]# docker image list
REPOSITORY     TAG       IMAGE ID       CREATED         SIZE
yslterraform   latest    54d5f158756c   7 minutes ago   104MB
```

5. Verify the Terraform version, as shown in Example 8-102.

Example 8-102 Verifying the Terraform version

```
[root@ylsprd ~]# docker run yslterraform version
Terraform v1.7.4
on linux_s390x
```

6. Start the Terraform console, as shown in Example 8-103.

Example 8-103 Starting the Terraform console

```
[root@ylsprd prd]# docker run -it --name yslcontainer yslterraform console
```

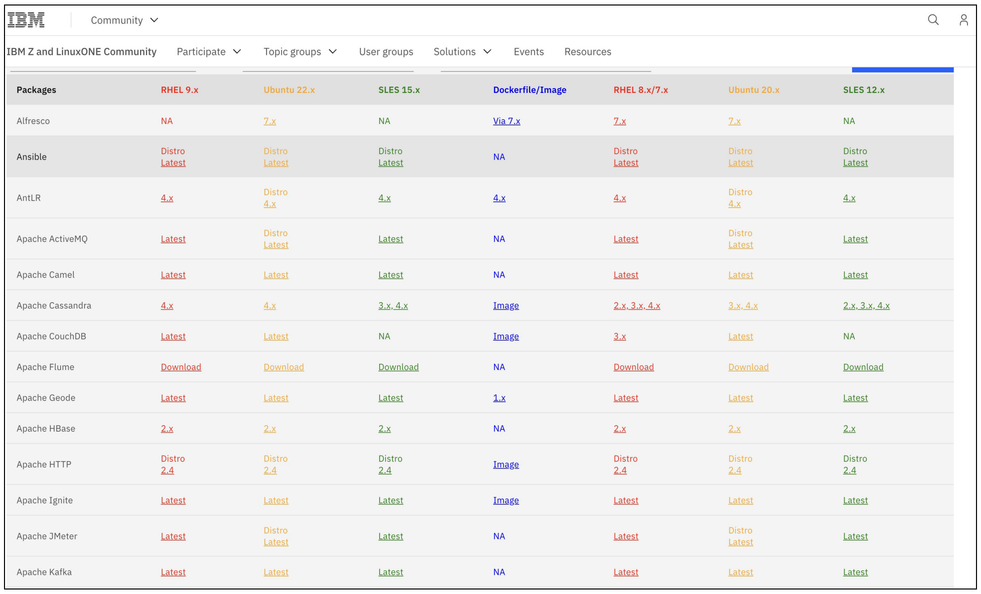
For more information about how to use Terraform, see Terraform CLI Documentation.

8.10 Deploying validated open-source software

Many open-source packages have been successfully ported or validated on the corresponding IBM distributions. For more information, see Validated Open Source Software.

Figure 8-7 shows the latest data on open-source packages that were adapted or confirmed for compatibility with the corresponding IBM distribution versions.

This information includes links to packaged binary files or documentation for building them on Linux on IBM Z. Also, the Dockerfile/Image column provides links to Dockerfile or docker images for certain packages.



Packages	RHEL 9.x	Ubuntu 22.x	SLES 15.x	Dockerfile/Image	RHEL 8.x/7.x	Ubuntu 20.x	SLES 12.x
Alfresco	NA	7.x	NA	Via 7.x	7.x	7.x	NA
Ansible	Distro Latest	Distro Latest	Distro Latest	NA	Distro Latest	Distro Latest	Distro Latest
AntLR	4.x	Distro 4.x	4.x	4.x	4.x	Distro 4.x	4.x
Apache ActiveMQ	Latest	Distro Latest	Latest	NA	Latest	Distro Latest	Latest
Apache Camel	Latest	Latest	Latest	NA	Latest	Latest	Latest
Apache Cassandra	4.x	4.x	3.x, 4.x	Image	2.x, 3.x, 4.x	3.x, 4.x	2.x, 3.x, 4.x
Apache CouchDB	Latest	Latest	NA	Image	3.x	Latest	NA
Apache Flume	Download	Download	Download	NA	Download	Download	Download
Apache Geode	Latest	Latest	Latest	1.x	Latest	Latest	Latest
Apache HBase	2.x	2.x	2.x	NA	2.x	2.x	2.x
Apache HTTP	Distro 2.4	Distro 2.4	Distro 2.4	Image	Distro 2.4	Distro 2.4	Distro 2.4
Apache Ignite	Latest	Latest	Latest	Image	Latest	Latest	Latest
Apache JMeter	Latest	Distro Latest	Latest	NA	Latest	Distro Latest	Latest
Apache Kafka	Latest	Latest	Latest	NA	Latest	Latest	Latest

Figure 8-7 Validated open source software on IBM LinuxONE

Note: Binary files or Docker images that are provided by the community might not be regularly validated. For more information, consult the official documentation.

Part 4

Appendix

This part contains Appendix A, “Additional use case scenarios” on page 255, and provides more use cases.

A

Additional use case scenarios

The complexity of a migration from Linux on the x86 depends on the platform architecture and context of the migration. The Linux operating system is more straightforward and well-known, and makes migration more efficient. However, when you consider an application, database management system (DBMS), or middleware migration, you also must consider degrees of complexity, cost, and risk.

This appendix provides more use case scenarios in which a telecommunications company, a healthcare company, and an energy and utilities company all want to migrate from x86 to IBM LinuxONE. It describes the challenges that are inherent to each industry and their respective migration scenarios.

This appendix includes the following sections:

- “Telecom industry consolidation on the cloud” on page 256
- “Healthcare industry: Mobile and internet solution” on page 257
- “Additional use cases” on page 263

Telecom industry consolidation on the cloud

In this scenario, Fictional Telco Company T1 selects the IBM LinuxONE platform for their Linux operating system consolidation and virtualization. Telco Company T1 wants to build a cloud platform, but they also want to reduce their cost of operation and overall data center footprint. The company's strategy is to improve provisioning time for its business support system (BSS) and operational support system (OSS) to satisfy server requests of its users. In this example, the following technology can be employed:

- Telco Company T1 uses the following consolidated hardware infrastructure:
  - IBM LinuxONE III T2 model
  - IBM z/VM 7.2
  - Red Hat Enterprise Linux (RHEL) or SUSE Linux Enterprise Servers on the LinuxONE platform
  - IBM Virtual Tape Library
- Telco Company T1 uses the following cloud automation products:
  - Automation with cloud: IBM Tivoli System Automation
  - Automated provisioning: Tivoli Provisioning Manager

Telco Company T1 uses the following build monitoring and system management tools:

- IBM Tivoli OMEGAMON on z/VM and Linux provides information about your Linux instances that are running as z/VM guests and their Linux workloads. IBM Tivoli OMEGAMON on z/VM and Linux reveals how the instances are performing and affecting z/VM and each other. Also, IBM Tivoli OMEGAMON on z/VM and Linux helps with the following tasks:
  - Compare Linux operations side by side with detailed performance metrics.
  - Data collection from the Performance Toolkit for VM (the Performance Toolkit is a prerequisite) complements data collection by the IBM Tivoli Monitoring for Linux for IBM LinuxONE agent.
  - With new Dynamic Workspace Linking, you can easily browse among Tivoli Enterprise Portal workspaces.
  - View and monitor workloads for virtual machines (VMs), groups, response times, and logical partition (LPAR) reporting.
  - View reports about z/VM and Linux usage of resources, such as CPU utilization, storage, mini-disks, and TCP/IP.
  - High-level views help executives understand how systems performance influences business and the bottom line.
  - With granular views, IT staffs can more easily track complex problems that span multiple systems and platforms, and share related information.
- Log-On Wave for IBM Z: Wave for IBM Z is a virtualization management product for z/VM and Linux virtual servers that simplifies and accelerates the management and daily administration of highly virtualized z/VM and Linux server environments on IBM Z and IBM LinuxONE platforms. It includes the following features:
  - Automate and simplify management, and monitor virtual servers and resources from a single dashboard.
  - Perform complex virtualization tasks in a fraction of the time compared to manual execution.

– Provision virtual resources (servers, networks, or storage) to accelerate the transformation to cloud infrastructure.
– Use advanced z/VM management capabilities, such as live guest relocation (LGR) with a few clicks.
– Delegate responsibility and provide more self-service capabilities to the appropriate teams.

Figure A-1 shows the solution architecture overview for a cloud solution that uses IBM LinuxONE.

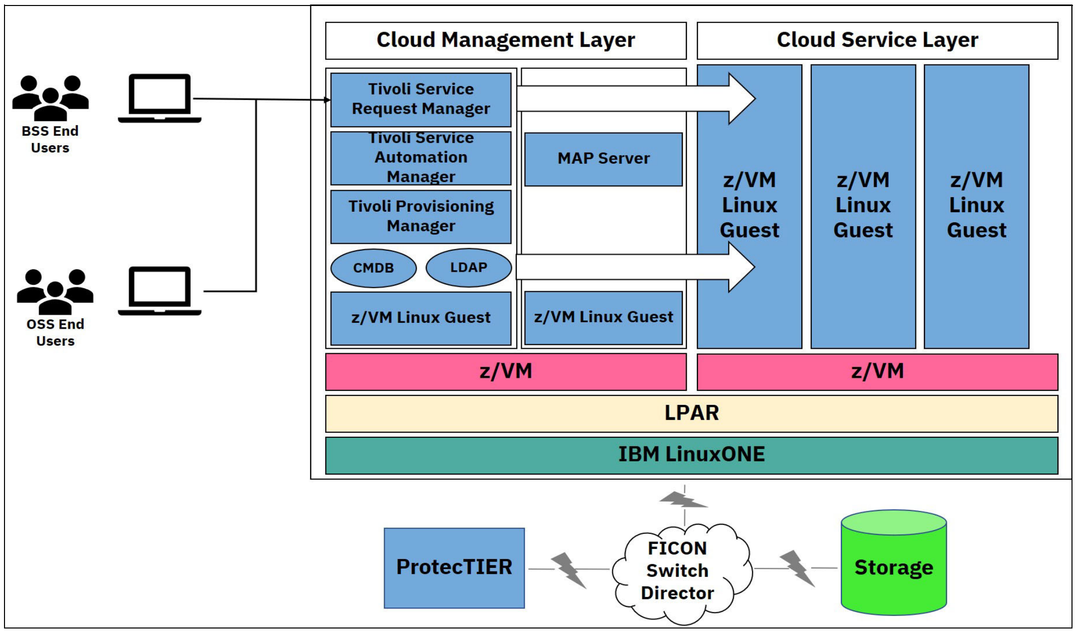


Figure A-1 Cloud solution that uses IBM LinuxONE

Healthcare industry: Mobile and internet solution

In this scenario, Fictional Hospital H1 also chooses IBM LinuxONE as its mobile application platform. Hospital H1 wants to build a secure platform, increase responsiveness and value perception, and reduce multi-platform development costs.

IBM Cloud® provides an extensible authentication model as part of its function. To comply with the Federal Information Processing Standards (FIPS), Hospital H1 uses WebSphere Application Server for added protection. The hospital configures WebSphere Application Server to protect the application and adapters for the back-end servers and data.

Using advanced granular controls, Hospital H1 can grant access to data on a role, time, and location basis. Doctors can access patient records on mobile devices. However, it requires extra authentication approval if they are at home or on call to review the latest observations of patients. In addition, although doctors have access to the information of their patients, medical suppliers have access to check inventory and update stock.

The solution is designed to increase responsiveness and perceived value perception:

- Hospital H1 is looking for a communication solution to find employees anywhere in the hospital. Using the LinuxONE high availability (HA) and advanced cryptographic capabilities, the hospital can build an application that helps enable instant and secure communication. Doctors and nurses can quickly find colleagues without stopping what they are doing.
- Doctors at Hospital H1 must enter prescriptions when their mobile devices are not connected to the network. The document-oriented storage system that is hosted on LinuxONE uses an encrypted container and helps ensure that the documents in the application are always available to doctors, even when the devices that are running the application are offline.
- With the application, patients can pre-register for appointments and enter their allergy information and health history by using mobile devices. IBM LinuxONE provides fast encryption for Transport Layer Security (TLS) with server identity verification and enables communication over HTTPS to protect the information.

The solution also reduces multi-platform development costs. Linux distributions on IBM LinuxONE provide a standards-based platform and enable Hospital H1 to use third-party libraries and frameworks as they did on x86.

Figure A-2 shows the secured access from a mobile device to a back-end transactional core system on the LinuxONE platform by using the global security policies and end-to-end secure transactions.

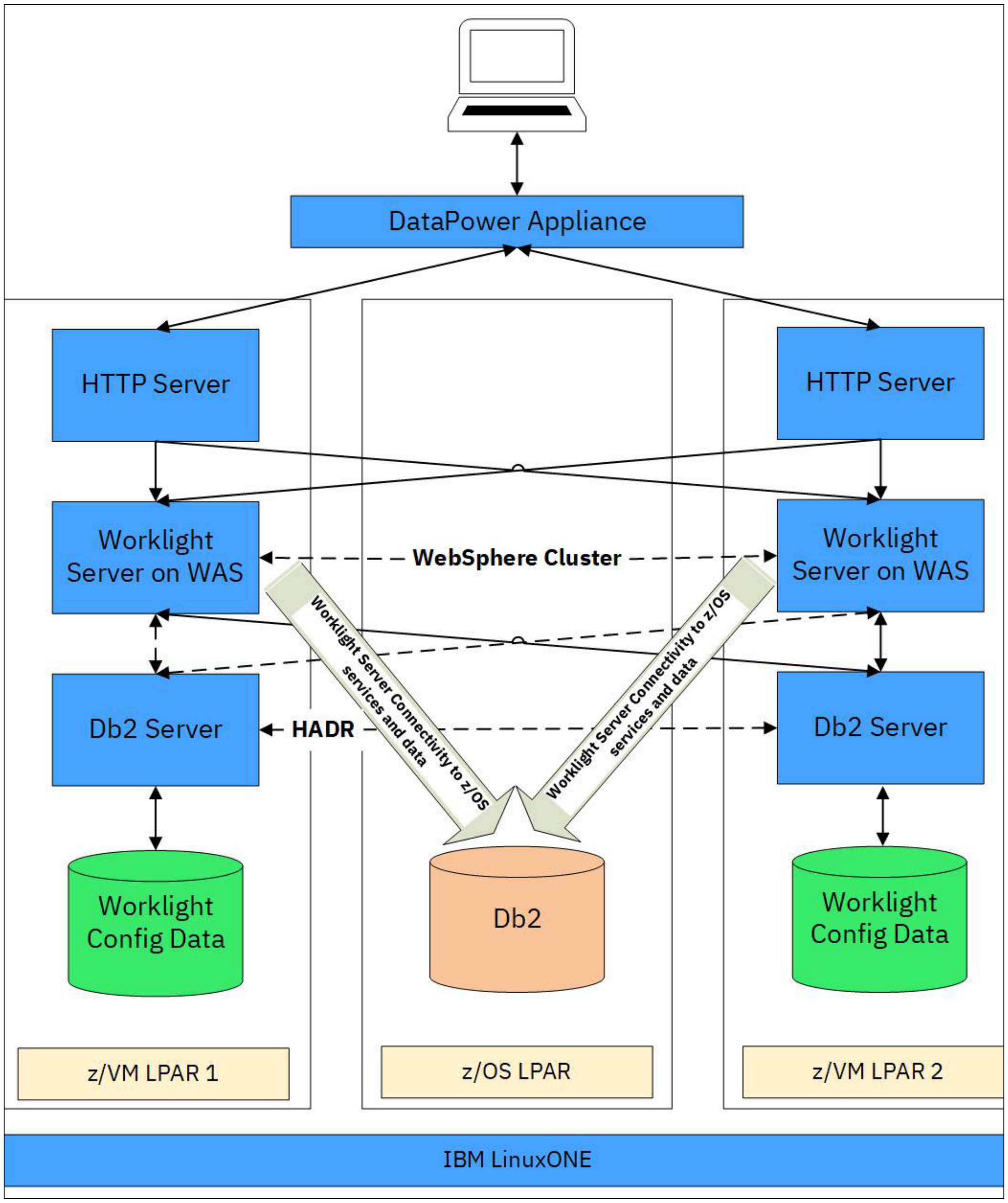


Figure A-2 Access from a mobile device to a back-end transactional core system

Transforming government services in Morocco: a case for efficiency and innovation

A government agency in Morocco is revolutionizing citizen services through a comprehensive e-government initiative. Aiming to deliver greater value and efficiency, the agency collaborated with IBM Business Partner PowerM and IBM to adopt a robust technology stack:

- IBM WebSphere Liberty: This lightweight application server provides a flexible and scalable foundation for applications.
- IBM Semeru Runtimes: Leveraging Java run times that are optimized for IBM LinuxONE, the agency helps ensure optimal performance and compatibility for their Java applications.
- IBM LinuxONE: as the underlying platform, LinuxONE offers unparalleled reliability with 100% availability for mission-critical applications, which helps ensure that citizens can access services uninterrupted.

Drivers and challenges

The agency’s main drivers include cost reduction while maintaining state-of-the-art information technology systems to support tax application processing and analytics:

- A powerful, integrated, open, and secure information system for data processing and analytics.
- IT resources that match the agency’s ambitions.
- Reduce costs.
- Organization and governance that promote efficiency and administration openness.

The agency's primary challenges revolve around performance optimization, security enhancements, and software compliance. They are actively seeking a comprehensive and global solution to address these critical challenges:

- Performance:
  - User experience
  - Page load times
  - Latency
  - Resource utilization
  - Throughput
  - Request processing time
  - Batch window
  - Backup window
- Software compliance:
  - Licensing complexity
  - License metric changes
  - Risk of vendor audit
  - Virtualization and cloud challenges
- Cybersecurity:
  - Data breaches
  - Privacy and confidentiality
  - Insider threats
  - Compliance and regulatory concerns
  - Requests and data encryption

Challenges

Seeking greater operational efficiency and faster public services, the agency embarked on a path of digital transformation. A key component of this journey involved harnessing application programming interfaces (APIs) to achieve the following goals:

- Boost data sharing by streaming communication and data exchange between different government entities.
- Automate processes by reducing manual tasks and improving service delivery speed and accuracy.

However, the agency faced a critical challenge: their IBM WebSphere Application Server and portals, which support numerous mission-critical applications, had not been updated in years. This "technical debt" created concerns about the following items:

- Application performance and reliability: Outdated software might potentially compromise the stability and functions of crucial services.
- Cybersecurity vulnerabilities: Unpatched software might leave the system exposed to evolving cyberthreats.

Solution

Recognizing the need for an upgrade to support their broader e-government vision, the agency decided to modernize their IBM WebSphere environment. This modernization helps ensure the following outcomes:

- A smooth operation that minimizes the risk of service disruptions.
- The latest security fixes and features safeguard the environment against current threats.

The agency implemented a cost-effective modernization strategy by leveraging the "zero-migration" feature of IBM WebSphere Liberty. This feature eliminates the need for time-consuming and expensive code changes during upgrades, and helps achieve the following goals:

- Seamlessly update to the latest Liberty version without incurring significant planning and risk mitigation expenses that are associated with traditional migrations.
- Maintaining compatibility with existing code while introducing new features. The agency can update their Liberty environment and access advanced runtime features at their own pace.
- Updating the agency server configurations and applications separately so that the agency can benefit from the latest features when they want.

This agency's experience further emphasizes the efficiency benefits of IBM LinuxONE:

- Several x86 servers were consolidated onto a single IBM LinuxONE server with Integrated Facility for Linux (IFL) processors, as shown in Figure A-3.

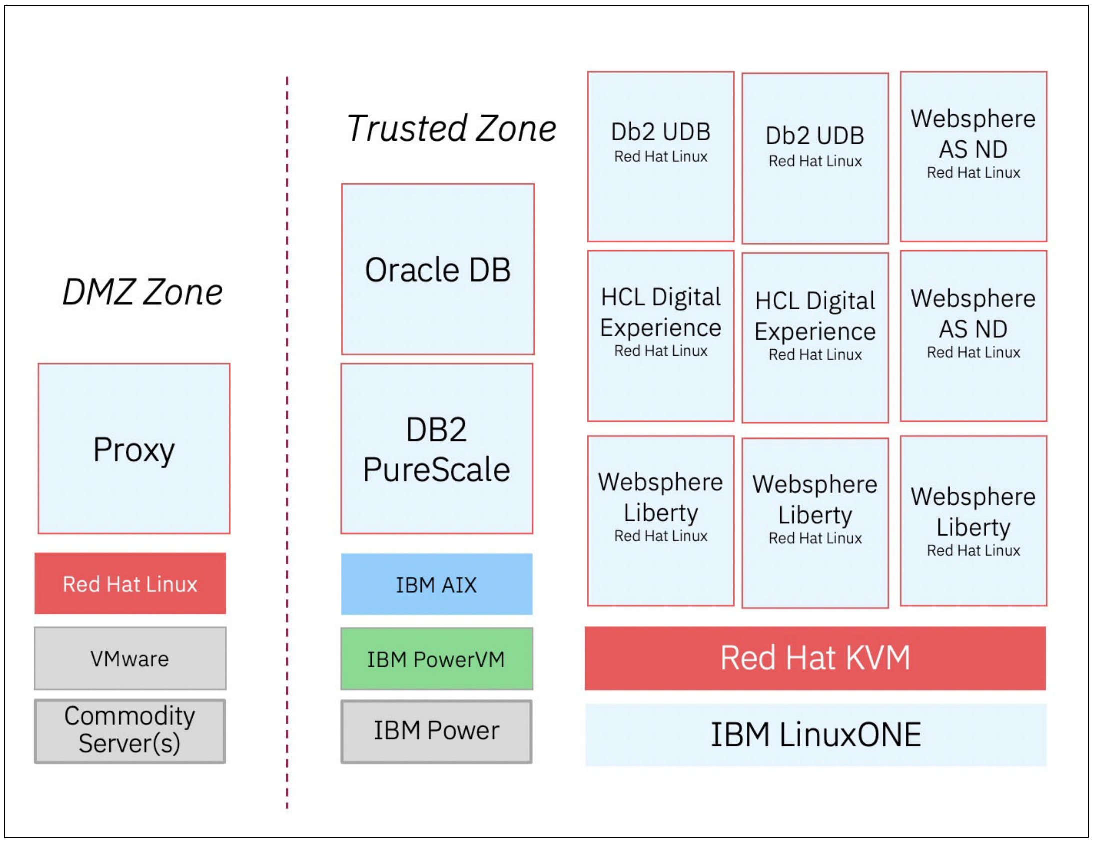
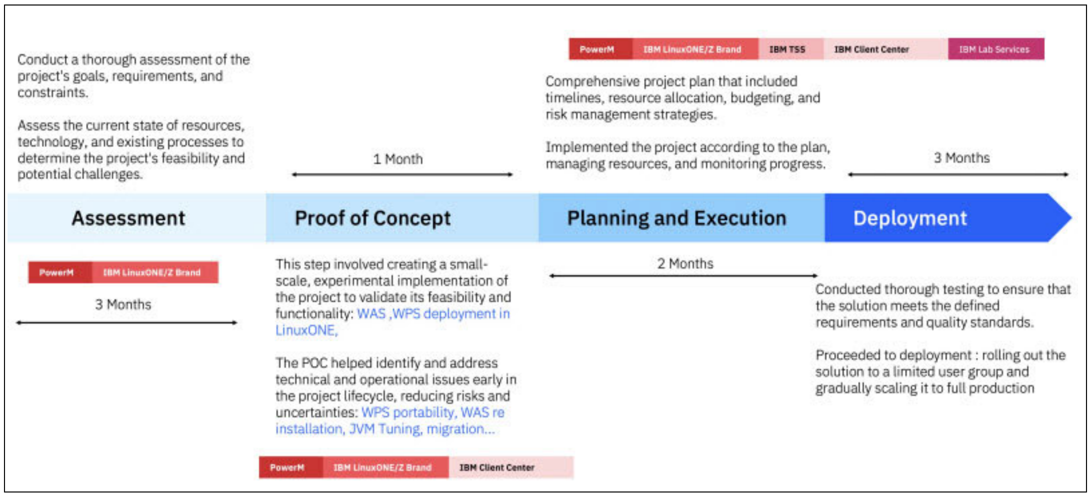

DMZ Zone
Proxy
Red Hat Linux
VMware
Commodity Server(s)
Trusted Zone
Oracle DB
DB2 PureScale
IBM AIX
IBM PowerVM
IBM Power
Db2 UDB Red Hat Linux
Db2 UDB Red Hat Linux
Websphere AS ND Red Hat Linux
HCL Digital Experience Red Hat Linux
HCL Digital Experience Red Hat Linux
Websphere AS ND Red Hat Linux
Websphere Liberty Red Hat Linux
Websphere Liberty Red Hat Linux
Websphere Liberty Red Hat Linux
Red Hat KVM
IBM LinuxONE

Figure A-3 Consolidating Java workloads on IBM LinuxONE

- Kernel-based Virtual Machine (KVM) virtualization technology further optimizes resource utilization and server management.
- Approximately 95% of Java workloads, including WebSphere Application Server and HCL Digital Experience (formerly known as WebSphere Portal), were hosted on the x86 architecture. By harnessing the combined power of the IBM LinuxONE exceptional computing capabilities and IBM FlashSystem® NVMe storage performance, and implementing Java Pauseless Garbage Collection, the agency achieved remarkable performance gains and their improved license consolidation ratio.
- The agency achieved a licensing cost reduction ratio of almost 12:1 by consolidating four x86 servers into a single IBM LinuxONE solution.
- Energy consumption: The cost of 1 kWh of electricity in Morocco for Commercial and Industrial usage is relatively high. LinuxONE offers a reduction in electricity expenses for the agency when compared to existing x86 servers while producing less CO_2.
- Security and compliance: IBM LinuxONE empowers the agency to implement pervasive encryption across the entire system at rest and in flight without compromising response times, which helps protect 100% of their data, 100% of the time, and also helping ensure that “what happens in a VM stays in a VM”.

IBM and PowerM delivered a successful project that showcased the power of collaboration. By leveraging their combined expertise and resources, they guided the agency through every step of the journey, as shown in Figure A-4:

- Assessment: IBM and PowerM worked together to thoroughly understand the agency's needs and challenges.
- Proof of concept (POC): A collaborative effort developed and tested a solution, and demonstrated its viability and potential benefits.
- Planning and execution: Combining strategic planning with flawless execution helped ensure a smooth and efficient project trajectory.
- Deployment: The meticulously planned solution was implemented with precision, which helped ensure a successful rollout and operational readiness.



Conduct a thorough assessment of the project's goals, requirements, and constraints.

Assess the current state of resources, technology, and existing processes to determine the project's feasibility and potential challenges.

This step involved creating a small-scale, experimental implementation of the project to validate its feasibility and functionality: WAS ,WPS deployment in LinuxONE,

The POC helped identify and address technical and operational issues early in the project lifecycle, reducing risks and uncertainties: WPS portability, WAS re installation, JVM Tuning, migration...

Comprehensive project plan that included timelines, resource allocation, budgeting, and risk management strategies.

Implemented the project according to the plan, managing resources, and monitoring progress.

Conducted thorough testing to ensure that the solution meets the defined requirements and quality standards.

Proceeded to deployment : rolling out the solution to a limited user group and gradually scaling it to full production

Figure A-4 Steps to migrate to IBM LinuxONE

For more information about this case study and how it demonstrates how combining zero-migration technology with a powerful platform like IBM LinuxONE empowers organizations to achieve cost-effective modernization and optimize resource utilization, see Enhancing mission-critical government services.

Additional use cases

For more information about other use cases, see the IBM LinuxONE client web page.

Related publications

The publications that are listed in this section are considered suitable for a more detailed description of the topics that are covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide more information about the topics in this document. Some publications that are referenced in this list might be available in softcopy only.

- *Advanced Networking Concepts Applied Using Linux on IBM System z*, SG24-7995
- *Consolidation Planning Workbook: Practical Migration from x86 to IBM LinuxONE*, REDP-5433
- *DB2 10 for Linux on System z Using z/VM v6.2, Single System Image Clusters and Live Guest Relocation*, SG24-8036
- *Experiences with Oracle 11gR2 on Linux on System z*, SG24-8104
- *Experiences with Oracle Solutions on Linux for IBM System z*, SG24-7634
- *IBM System Storage SAN Volume Controller, IBM Storwize V7000, and IBM FlashSystem 7200 Best Practices and Performance Guidelines*, SG24-7521
- *IBM Z Connectivity Handbook*, SG24-5444
- *IBM Wave for z/VM Installation, Implementation, and Exploitation*, SG24-8192
- *IBM zEnterprise EC12 Technical Guide*, SG24-8049
- *Implementing FlashSystem 840 with SAN Volume Controller*, TIPS1137
- *Implementing the IBM System Storage SAN Volume Controller with IBM Spectrum Virtualize V8.2.1*, SG24-7933
- *Introduction to Storage Area Networks*, SG24-5470
- *Introduction to the New Mainframe: z/VM Basics*, SG24-7316
- *An Introduction to z/VM Single System Image (SSI) and Live Guest Relocation (LGR)*, SG24-8006
- *Linux on IBM eServer zSeries and S/390: Application Development*, SG24-6807
- *Linux on IBM System z: Performance Measurement and Tuning*, SG24-6926
- *Security for Linux on System z*, SG24-7728
- *Security on z/VM*, SG24-7471
- *Set up Linux on IBM System z for Production*, SG24-8137
- *Using z/VM v 6.2 Single System Image (SSI) and Live Guest Relocation (LGR)*, SG24-8039
- *The Virtualization Cookbook for IBM Z Volume 1: IBM z/VM 7.2*, SG24-8147

You can search for, view, download, or order these documents and other Redbooks, Redpapers, web docs, drafts, and additional materials at the following website:

ibm.com/redbooks

Online resources

The following websites are also relevant as further information sources:

- LinuxONE solutions:

  `http://www.ibm.com/systems/linuxone/solutions`

- GNU assembler manual:

  `http://www.gnu.org/software/binutils`

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Abbreviations and acronyms

AI	artificial intelligence
API	application programming interface
BSS	business support system
BVA	Business Value Assessment
CA	continuous availability
CBU	Capacity BackUp
CEX	CryptoExpress
CFO	Chief Financial Officer
CIC	Cloud Infrastructure Center
CIO	Chief Information Officer
CIU	Customer Initiated Upgrade
CMM	Cooperative Memory Management
CMMA	Cooperative Memory Management Assist
CMS	Conversational Monitor System
CO	continuous operation
CO2e	carbon dioxide equivalent
CP	Control Program
CPACF	Central Processor Assist for Cryptographic Functions
CPC	central processing complex
CPE	Capacity for Planned Event
DBA	database administrator
DBMS	database management system
DCSS	Discontiguous Saved Segment
DDL	Data Definition Language
DLP	data loss prevention
DN	distinguished name
DoS	denial-of-service
DPM	Dynamic Partition Manager
DR	disaster recovery
EAL	Evaluation Acceptance Level
EE	Enterprise Edition
ELK	Elasticsearch, Logstash, and Kibana
EPL	Eclipse Public License
ESG	Environmental, Social, and Governance
FCP	Fibre Channel Protocol
FIPS	Federal Information Processing Standards
GC	garbage collection
GSKit	Global Security Kit
HA	high availability
HADR	high availability and disaster recovery
HMC	Hardware Management Console
HPC	high-performance computing
HSM	Hardware Security Module
IaaS	Infrastructure-as-a-Service
IaC	Infrastructure as Code
IBM	International Business Machines Corporation
IFL	Integrated Facility for Linux
I/O	Input-Output
IODF	input/output definition file
IPC	interprocess communication
IPMI	Intelligent Platform Management Interface
ISM	Internal Shared Memory
ISV	independent software vendor
JDBC	Java Database Connectivity
JDK	Java SE Developer Kit
JIT	just-in-time
JVM	Java virtual machine
KVM	Kernel-based Virtual Machine
LAG	link aggregation group
LAN	local area network
LCSS	logical channel subsystem
LDAP	Lightweight Directory Access Protocol
LDIF	LDAP Data Interchange Format
LGR	live guest relocation
LPAR	logical partition
LUN	Logical Unit Name
LV	logical volume
LVM	Logical Volume Manager
MAC	Media Access Control
MIB	Management Information Base
MOTD	Message of the Day
MTU	maximum transmission unit
NAS	network-attached storage
NPIV	N_Port ID Virtualization

NSS	Named Saved System
OCI	Open Container Initiative
OSA	Open Systems Adapter
OSS	operational support system
OVS	Open vSwitch
POC	proof of concept
PV	physical volume
QDIO	queued direct I/O
QI	Quality Indicator
RAC	Real Application Cluster
RAIM	Redundant Array of Independent Memory
RDMA	Remote Direct Memory Access
RHDS	Red Hat Directory Server
RHEL	Red Hat Enterprise Linux
RHOCP	Red Hat OpenShift Container Platform
RMAN	recover manager
ROI	return on investment
SAN	storage area network
SCSI	Small Computer System Interface
SLA	service-level agreement
SMB	Server Message Block
SPOF	single point of failure
SSC	Secure Service Container
SSH	Secure Shell
SSI	Single System Image
SSL	Secure Sockets Layer
SVC	SAN Volume Controller
TCO	total cost of ownership
TEE	trusted execution environment
TLS	Transport Layer Security
UEFI	Unified Extensible Firmware Interface
VFM	Virtual Flash Memory
VG	volume group
VLAN	virtual local area network
VM	virtual machine
VMRM	Virtual Machine Resource Manager
vRA	VMware vRealize Automation
WAN	wide area network
WCS	water-cooled system
WORM	Write Once Read Many
WWN	worldwide name

Redbooks

Practical Migration from x86 to IBM LinuxONE

SG24-8377-02

ISBN 0738461725

(0.5” spine)
0.475”<->0.873”
250 <-> 459 pages

SG24-8377-02

ISBN 0738461725

Printed in U.S.A.