SPONSORED BY

GEEK GUIDE

An Architect's Guide: Linux in the Age of Containers

LINUX™ JOURNAL

Table of Contents

SOL LEDERMAN is a technical people-oriented professional with more than 30 years of broad experience in system administration, software design and development, technical support, training, documentation, troubleshooting and customer management. Sol currently divides his time providing IT, consulting and technical content marketing services. Learn more at http://SolLederman.com.

GEEK GUIDES:
Mission-critical information for the most technical people on the planet.

About the Sponsor

SUSE, a Micro Focus company, provides and supports enterprise-grade Linux and open-source solutions with exceptional service, value and flexibility. With partners and communities, we innovate, adapt and deliver secure Linux, cloud infrastructure and storage software to create solutions for mixed enterprise IT environments. We help customers harness the benefits and power of an open enterprise that can empower their possibilities.

An Architect's Guide: Linux in the Age of Containers

SOL LEDERMAN

Introduction

As the pace of technology accelerates and the demands on IT organizations rapidly increase, enterprises are faced with a tension between two opposing forces:

1. Maintaining a stable, high-performance, highly available, secure and scalable computing environment.

2. Becoming more agile to be able to deploy new, uncertain projects and technologies more quickly.

And, of course, enterprises can't abandon the critical components of the first demand to meet those of the second demand. In 2014, Gartner coined the term "Bimodal IT" to refer to a model for handling those two competing demands. Here's Gartner's IT Glossary (http://www.gartner.com/it-glossary/bimodal) definition:

> **Bimodal** is the practice of managing two separate but coherent styles of work: one focused on predictability; the other on exploration. Mode 1 is optimized for areas that are more predictable and well-understood. It focuses on exploiting what is known, while renovating the legacy environment into a state that is fit for a digital world. Mode 2 is exploratory, experimenting to solve new problems and optimized for areas of uncertainty.

My first ebook in this Geek Guide series, *An Architect's Guide: Linux for Enterprise IT* (https://geekguide.linuxjournal.com/content/architect's-guide-linux-enterprise-it), focused on Mode 1. I considered the major factors driving adoption of Linux in the traditional enterprise. I also looked at stability, performance, scalability, high availability, security, Windows integration, ease of deployment, management and upgrades, and support. And, I made the case that SUSE Linux Enterprise Server was a natural choice in high-performance computing environments for meeting those needs.

The Challenges of Mode 2 IT

This guide examines the needs of Mode 2 and focuses on

what it takes to bring new technologies into production quickly and in a massively scalable fashion if needed. In addition, I make the case that SUSE is well positioned to meet the needs of the Mode 2 IT organization with its enterprise-grade solutions designed for mixed IT environments and certified on all major hardware platforms.

Terms associated with Mode 1 are *monolithic*, *highly available*, *virtual host/guest* and *"always up" OS*. Mode 2 can be described with these terms: *containers*, *transactional updates*, *rapid cluster deployment and easy management*, and *"always up to date" OS*.

Before going into the various technologies that enable Mode 2 computing with ease, especially in mission-critical environments, let's consider the ten major challenges to realizing the benefits of Mode 2.

1. Public cloud support and, for some organizations, the need for an on-premises solution—that is, a private cloud is needed, along with the need to support hybrid clouds that combine public and private clouds.

2. Containers must be more "consumable", such that they reduce time to market for new products and services, host services with ease, and support applications across platforms through a modular architecture design and a common code base.

3. Reduce overhead, boost efficiency and realize the economic benefits of containers vs. hardware virtualization—these benefits can be achieved by

optimizing use of hardware through tight managing of resources and by utilizing only those resources needed.

4. Simplify container management by unifying tools that build, provision and manage container clusters. These unified tools should provide good defaults to get a clustered environment up and running quickly and also support pre-configuration of components for rapid deployment and redeployment. Upgrades should be easy to deploy as well.

5. Deploy containers on a lightweight and, ideally, container-centric operating system.

6. Clusters of containers need to scale to thousands of nodes easily and dynamically.

7. Keep the OS version current and always up to date with patches and automatically updated (if desired).

8. Make OS updates be atomic and have a mechanism for quickly and automatically reverting to a previous OS snapshot if an upgrade fails, with no downtime.

9. Avoid costly security issues by obtaining signed container images from a secure and trusted public registry or from a private registry.

10. Integrate community-developed software without voiding the OS vendor's support agreement.

The Three Major Themes of Mode 2 IT

The ten challenges of Mode 2 IT have three major themes:

1. Virtualization beyond hypervisors, containers and clusters, with a focus on streamlining the processes of building, deploying and managing containers while keeping them secure.

2. Public, private and hybrid cloud support.

3. Agility (modules, packaging and cross-platform support via a common code base).

SUSE has developed a strategy and platform to address the ten challenges.

The CaaS Platform

SUSE defines CaaS Platform as "an application development and hosting platform for container-based applications and services that enables you to provision, manage and scale container-based applications and services" in its CaaS Platform introductory webinar (https://www.brighttalk.com/webcast/11477/242539). It defines the goal of CaaS Platform as being to "deliver a container infrastructure platform that automates the tedious management tasks allowing customers to focus on development of applications and meet their business goals faster."

SUSE CaaS Platform (http://suse.com/caas) unifies three technologies to create a platform that greatly facilitates

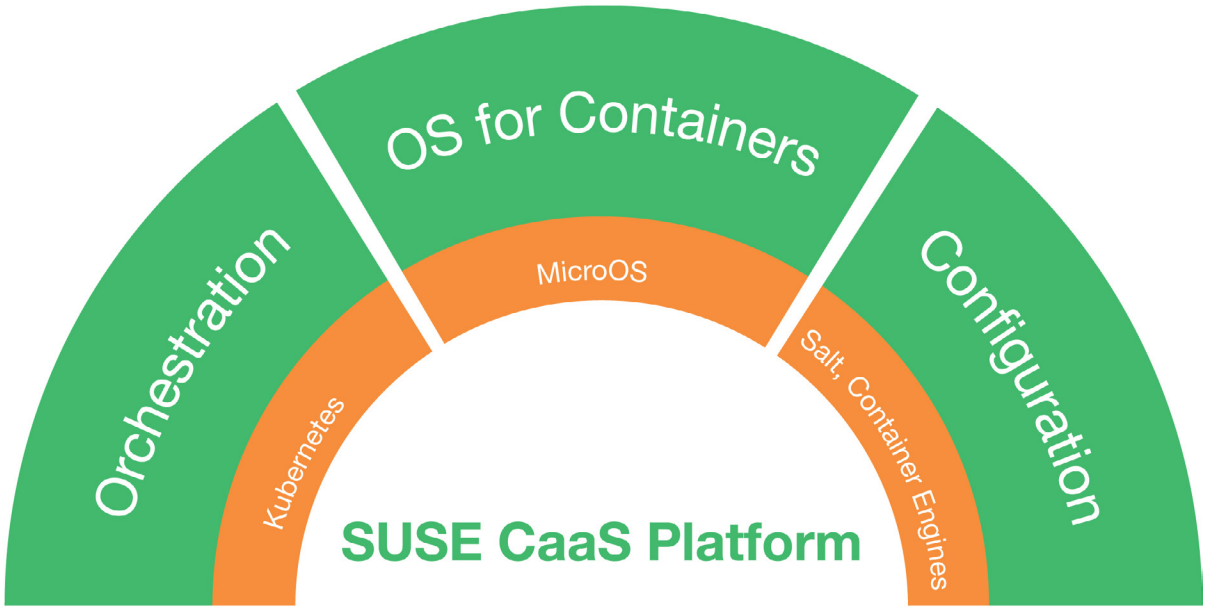

FIGURE 1. SUSE CaaS Platform (Image Courtesy of SUSE)

the process of working with containers. Figure 1 illustrates those unifying components.

Orchestration Clusters of containers in SUSE CaaS Platform are deployed automatically, scaled and managed with an orchestration solution based on Kubernetes.

OS for Containers SUSE has developed an operating system, MicroOS, designed for containers and microservices. The term "Micro" in Micro OS signifies microservices. MicroOS is optimized for large container deployments; it is small, efficient and can be distributed at a large scale reliably and securely. (I discuss MicroOS in more detail in the next section.)

Configuration Salt serves as the configuration engine to manage complete and automatic installation and configuration of components. A container engine from the

open-source Docker container engine project is also part of the configuration component.

Raj Meel introduces SUSE CaaS Platform in the SUSE Blog and makes the business case for using SUSE's platform and bypassing the cost and effort of "rolling your own" container infrastructure (https://www.suse.com/communities/blog/rise-caas-platform). Meel lists a number of factors that IT organizations would need to consider if architecting their own container environment from scratch that SUSE CaaS Platform handles in its unified solution:

> There are a lot of decisions to be made before you are ready for production/deployment of container apps. What orchestration tools to use, how to manage registry of images, can the registry images be trusted, how to securely collaborate in developing apps, how to surgically patch container images for vulnerabilities, how to scale, and so on.

Figure 2 shows the interaction of components of the CaaS Platform stack across different layers:

- The first (bottom) layer is the infrastructure, which can be physical or virtual. This infrastructure can live in any public, private or hybrid cloud environment.

- The next layer includes SUSE MicroOS, which is the container-aware operating system that manages the infrastructure and serves as the container host OS. MicroOS also includes container runtime and packaging code.

- The automation layer deals with unassisted configuration and management of the cluster nodes. Automation is critical, because SUSE CaaS Platform is designed to manage potentially thousands of nodes with multiple containers per node. Salt is the first of two components of the automation strategy. Salt is used to boot and configure the cluster nodes. The second automation component is cloud-init. Cloud-init is required in order to bootstrap the Salt dæmon with its dependencies. Cloud-init manages the SUSE MicroOS deployments.

- The next layer contains a number of services: basic

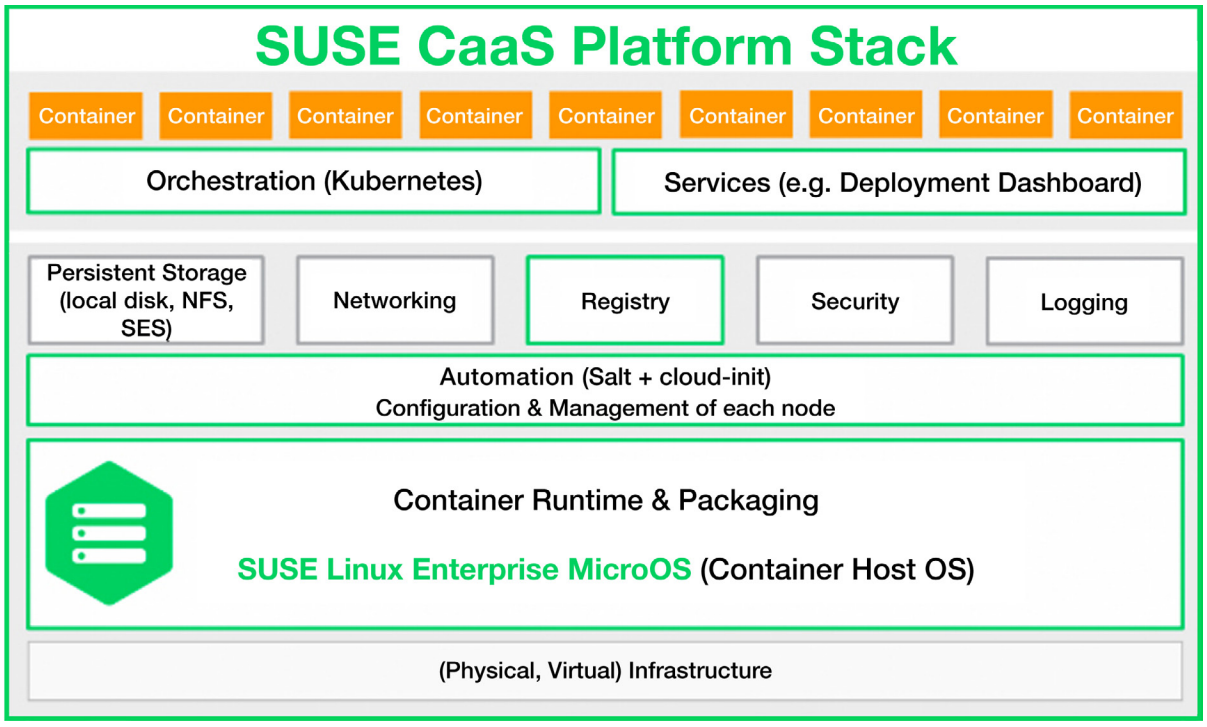

FIGURE 2. The CaaS Platform Stack (Image Courtesy of SUSE)

logging from the monitoring of containers and clusters (beyond other monitoring put in place by IT), basic security and user management, the SUSE private registry for reliable distribution of trusted container images, management of networking and management of persistent storage.

- The next layer consists of Kubernetes to orchestrate the container infrastructure plus a deployment dashboard and a set of services to facilitate working with containers.

- The final layer is the container platform layer that hosts applications.

About MicroOS

SUSE MicroOS is built on the foundation of the reliable SUSE Linux Enterprise Server, and it stands as one of the pillars of SUSE CaaS Platform. MicroOS is designed for containers and microservices, running only those microservices that containers require, and MicroOS is optimized for scalability across large deployments. MicroOS is designed to work with clouds and Kubernetes. And, MicroOS is designed to reduce end-user interaction through automation and pre-configuration of resources.

"SUSE Formalizes Container Strategy with a New Linux Distro, MicroOS" on the New Stack website introduces the SUSE CaaS Platform strategy and summarizes what goes into SUSE MicroOS (https://thenewstack.io/micro-os-suses-answer-container-os): "What goes into MicroOS is

comparable to a minimal system of SUSE Linux Enterprise Server. It contains everything to be able to boot on your machine, whether on bare metal or a virtual machine like KVM, Xen, VMware", explained Thorsten Kukuk, a SUSE distinguished engineer and senior architect. "It contains the kernel, glibc, systemd, dependencies and all of that we need to boot the machine and start container engine, Kubernetes, etc., whatever we need for it."

SUSE MicroOS uses core technologies from SUSE Linux Enterprise Server, such as Btrfs, with these characteristics:

- The base OS and snapshots are read-only, thus making them more secure.

- Updates, with the use of Btrfs, are transactional. Updates are automatic, but that mechanism can be disabled. Kukuk (quoted previously) summarizes the elements of transactional updates on a SUSE mailing list post (https://lists.opensuse.org/opensuse-factory/2017-01/msg00367.html): "[Updates are] atomic. The update does not influence your running system. [Updates are] either successfully applied or nothing is changed. So no broken RPMs are flying around in the system. [Updates] can be rolled back. If the upgrade fails or if the update is not compatible, you can quickly restore the situation as it was before the upgrade."

- Subvolumes for data storage are read-write.

- OverlayFS is used for /etc (for cloud-init and Salt).

The New Stack article mentioned above explains the motivation behind using Btrfs and snapshots and how MicroOS works with updates:

> But these systems also need to be fail-safe. If an update breaks something, it should not stop services. MicroOS is relying on the snapshot feature of the Btrfs filesystem, which differentiates it from CoreOS's Container Linux, Ubuntu Core and Project Atomic. Other systems have two partitions where one is updated while the other one runs and they switch after reboot. With Btrfs you have one partition but you can keep as many snapshots as you want and switch among them. All updates are created as a snapshot, and after the system runs the latest snapshot after reboot, if something fails, it automatically boots into the previously working snapshot. And since we are talking about clusters, there is no downtime.

Public, Private and Hybrid Cloud Computing

SUSE CaaS Platform was designed to run seamlessly on any cloud architecture. SUSE introduces its public, private and hybrid cloud offerings at its website (https://www.suse.com/solutions/cloud). Clouds are the foundation for scalable container computing.

Public clouds are the easiest for an IT organization to get started with. Operation and management of public clouds are handled by the cloud service provider in much the same way that an ISP manages internet service for its

customers. Amazon AWS, Microsoft Azure and Google Cloud Platform are perhaps the best known public clouds. SUSE has partnered with all of the major cloud providers, and with dozens of others, to ease transition to public clouds (https://www.suse.com/promo/cloud/public).

Although there are benefits to outsourcing cloud management, there are two major sets of concerns: 1) security and privacy and 2) uptime and availability. CEPIS, the Council of European Professional Informatics Societies, articulates the first set of concerns (https://www.cepis.org/index.jsp?p=641&n=825&a=4758):

> There seems to be no area of ICT [Information and Communication Technology] that is not affected by Cloud Computing. Two main issues exist with security and privacy aspects of Cloud Computing:
>
> 1) Loss of control over data.
>
> 2) Dependence on the Cloud Computing provider.
>
> These two issues can lead to a number of legal and security concerns related to infrastructure, identity management, access control, risk management, regulatory and legislative compliance, auditing and logging, integrity control as well as Cloud Computing provider dependent risks.

The second set of concerns, uptime and availability, are closely related to the first set. Dependence on the cloud computing provider and loss of control over data

Private clouds don't remove the burden of managing security, privacy and accessibility; they put that responsibility and control into the hands of the enterprise, and that responsibility comes with a variety of pluses and minuses.

introduces risks. As an example, Wikipedia provides information on five major Amazon Web Service outages that occurred between October 2012 and February 2017 (https://en.wikipedia.org/wiki/Timeline_of_Amazon_Web_Services#Amazon_Web_Services_Outages).

Private clouds don't remove the burden of managing security, privacy and accessibility; they put that responsibility and control into the hands of the enterprise, and that responsibility comes with a variety of pluses and minuses. Beyond the concerns of the effort involved in managing a private cloud, there are compelling reasons for organizations to embrace private clouds. SUSE published a white paper detailing five major reasons to embrace private clouds (https://suse.lookbookhq.com/private-cloud-pain-free-experience/5-reasons), which are the following:

1. Cost savings.

2. Agile environment.

3. Deployment of production workloads.

4. Enhanced flexibility.

5. Security and compliance.

For some organizations, hybrid clouds, which combine public and private clouds, may be ideal. InformationWeek considers the pros and cons of the two architectures and makes the case for hybrids clouds in an article "Hybrid Cloud: 7 Ways It's The Best Of Both Worlds" (http://www.informationweek.com/cloud/infrastructure-as-a-service/hybrid-cloud-7-ways-its-the-best-of-both-worlds/d/d-id/1322559):

> The decision to use one cloud architecture over the other depends on many factors, including security, latency, redundancy, and required speed of implementation. The goal of a hybrid cloud is to make it so that end users have no idea if an application or resource is in a public or private cloud. To them, it's all the same. Each application and data resource is carefully reviewed and a decision is made about where the resource will operate best.

Some organizations have shied away from deploying private clouds due to the complexities of managing all of the components. SUSE CaaS Platform was designed to provide a platform for containers to enable easy deployment of containerized applications and it does so

in a number of ways:

1. Orchestration with Kubernetes provides a complete solution for deploying, managing and scaling container-based workloads. Kubernetes has self-healing mechanisms that allow users to define the desired application state in the face of shifting resource needs and resource failures. Kubernetes also provides dashboard and UI capabilities.

2. The MicroOS Admin node is installed in one step. From there, the Admin dashboard is easy to set up, and nodes are deployed automatically via an AutoYaST profile.

3. SUSE CaaS Platform supports any type of cloud. In particular, it supports the SUSE OpenStack Cloud, which is documented at the SUSE website (https://www.suse.com/products/suse-openstack-cloud). Two key features are worth noting:

   > [SUSE OpenStack Cloud] delivers enterprise-ready technology for building Infrastructure-as-a-Service (IaaS) private clouds, giving you access to automated pools of IT resources to efficiently develop and run applications and workloads in your data center. SUSE OpenStack Cloud closely integrates with SUSE Enterprise Storage, powered by Ceph, for highly scalable and resilient software-defined storage capabilities.

4. The SUSE CaaS Platform platform provides the "Run

Regional outages can cripple a business. Being able to move resources away from particular data centers manually or automatically is key for managing workloads that demand zero downtime.

Everywhere" capability. SUSE defines this in its SUSE CaaS Platform 1.0 release notes (https://www.suse.com/releasenotes/x86_64/SUSE-CAASP/1.0):

> You have the ability to quickly and intelligently respond to demand across private and public clouds. SUSE Container as a Service Platform helps you manage peak demand without downtime or manual intervention.

Kubernetes supports the "Run Everywhere" model for containers.

One final consideration for those considering deployment of any cloud architecture is geographic clustering. Regional outages can cripple a business. Being able to move resources away from particular data centers manually or automatically is key for managing workloads that demand zero downtime. *Geo Clustering for SUSE Linux Enterprise High Availability Extension* allows

enterprises to maintain geographically dispersed clouds and failover clusters as needed (https://www.suse.com/products/highavailability/geo-clustering).

Agility

Mode 2 IT is synonymous with being agile. The ten major drivers and three major themes I discussed earlier were agility-focused. Cloud computing in its various flavors combined with the SUSE CaaS Platform provides a powerful framework for rapid development of clustered container applications accelerating time to market and increasing business agility. In this section, I look at SUSE tools that promote agile development and deployment.

SUSE CTO Dr Thomas Di Giacomo clearly states the importance and the urgency of being agile in his article "How DevOps Can Support Business Agility for All Companies to Stay Business-Relevant" (https://www.suse.com/docrep/documents/id6s306m58/stay_business_relevant_and_achieve_business_agility_with_devops.pdf). He leads with the reminder that only the fittest survive:

> In our modern, fast-paced, digital-first world, responding quickly to internal and external changes without losing vision is absolutely key for all companies who want to survive, thrive and ultimately surpass the competition.

A modular operating system design serves as the foundation for agile package updates. SUSE Linux Enterprise Server 12, the OS that serves as the basis

for MicroOS, introduced modular packaging (https://www.suse.com/docrep/documents/huz0a6bf9a/suse_linux_enterprise_server_12_modules_white_paper.pdf):

> SUSE Linux Enterprise Server Modules are sets of packages grouped into their own repository and updated independently of service pack lifecycles. The lifecycle for each module is different. The packages in each module have a common use case and a common support status. SUSE fully maintains and supports the modules.

Modular packaging allows users to deploy new versions of supported packages without needing to wait for the next service pack or a new OS version release. Figure 3 lists the seven available modules (as of July 2017).

The SUSE Package Hub provides another mechanism for quickly installing new packages on SUSE Linux Enterprise Server 12 (https://packagehub.suse.com). SUSE Package Hub serves as a clearinghouse for more than 800 popular

The Modules

There are seven modules available from SUSE

Module Name	Content	Lifecycle
Web and Scripting Module	PHP, Python, Future Ruby on Rails, Node.js	3 years with 18-month overlap
Legacy Module	Sendmail, old IMAP stack, old Java, etc.	September 2017
Public Cloud Module	Public cloud initialization codes and tools	Frequent releases
Toolchain Module	GNU Compiler Collection (GCC)	Yearly delivery
Certifications Module	FIP140-2 certification-specific packages	Certification-dependent
Containers Module	Container engines and SUSE prepackaged images	Frequent releases
Advanced Systems Management Module	CFEngine, Puppet and the new Machinery tool	Frequent releases

FIGURE 3. SUSE Linux Enterprise Server Modules (Image Courtesy of SUSE)

open-source packages (as of July 2017), which are pre-built for quick installation. Packages are built and maintained by the community, and although they are not officially supported by SUSE, their installation and use does not void support for the server.

At a more fundamental level than packaging, SUSE supports agility through a common source code base that ensures a more stable operating system with less code to maintain. The SUSE blog article "Introducing SUSE Linux Enterprise Server for ARM" explains what common code base is and what it means (https://www.suse.com/communities/blog/introducing-suse-linux-enterprise-server-arm):

> [Common code base] means that the versions, and thereby the source code, of all core packages of the SUSE Linux Enterprise product family are the same—from the desktop to the mainframe. The tool chain, like compilers and libraries, is the same across the supported hardware architectures. The common code base guarantees product consistency and a persistent look and feel, which lets you leverage the skills of your IT staff. It also provides for the highest code quality, better supportability and preemptive code maintenance.

Figure 4 illustrates the common source code that serves as the foundation for the SUSE Linux Enterprise code stack. New architecture support is added to the gray section of the stack diagram.

64-bit ARM architecture support is a good example of needing to extend only the binary code base to include

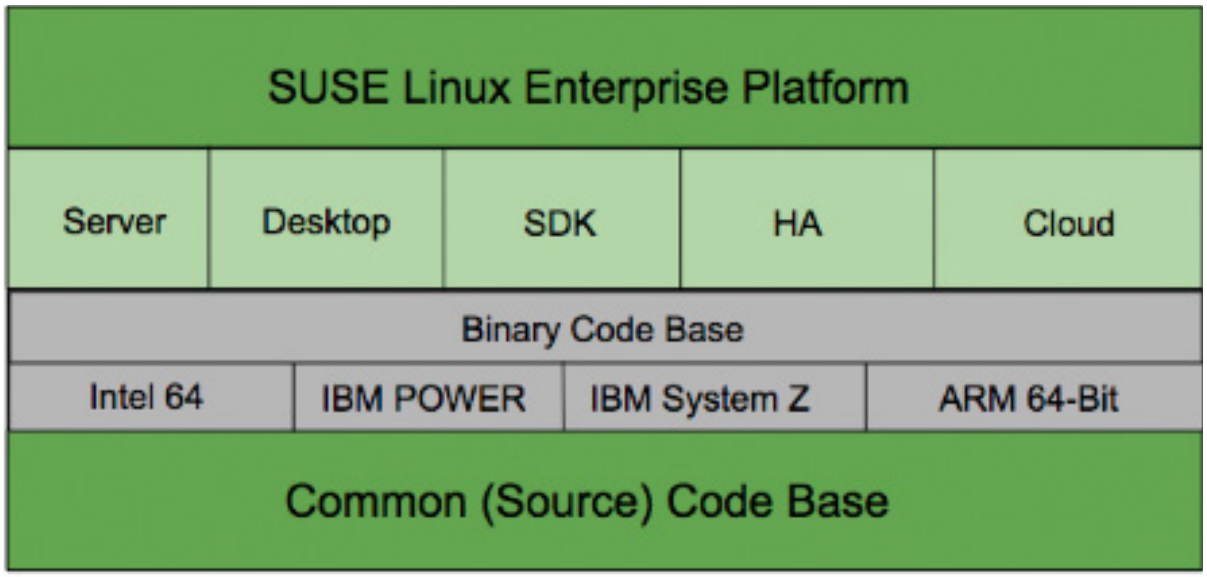

FIGURE 4. SUSE Linux Enterprise 12 Common Code Base and Architectures (Image Based on a SUSE Figure)

support for the new platform, and everything else in the code stack just works. Shorter lead time for new platform support translates to greater agility, especially in cross-platform environments. And, as noted in its ARM introduction, SUSE is first to market with 64-bit ARM support (https://www.suse.com/products/arm):

> SUSE Linux Enterprise Server for ARM is the first generally available commercial enterprise-grade Linux distribution that is optimized for servers based on the 64-bit ARM v8-A architecture, enabling hardware and software solution providers to exploit ARM-based capabilities on a well-established Linux distribution with enterprise-class security and superior support.

A prime example of the value of 64-bit ARM support is that SUSE was able to provide the first enterprise-grade distribution for 64-bit Raspberry Pi and 64-bit ARM (https://www.suse.com/products/arm/raspberry-pi). The stable common code base in SUSE Linux Enterprise Server 12 SP2 that supports other 64-bit processor architectures was used to add 64-bit ARM support.

Closing Thoughts

Hopefully I've made a compelling case that the shifting demands of IT require a shift in the types of tools you need to meet those changing demands. The need for agility drives enterprises to cut development and deployment times but ideally without cutting corners. Virtualization is here to stay. Containers are the leading edge. Leading-edge tools to manage their complexity contribute directly to agility. Tools to manage containers efficiently and seamlessly in mixed cloud environments are also critical to agility, as no two IT environments are the same. I hope this guide has given you some insights on how SUSE has embraced the challenge of lessening the growing pains of leading-edge IT.■