1

IST 615 – CLOUD MANAGEMENT

CONTAINERS & KUBERNETES (1)

Carlos E. Caicedo, Ph.D.

Associate Professor
Director – Center for Emerging Network Technologies (CENT)
School of Information Studies
Syracuse University

1

Outline

2

- Announcements
  - Midterm
- Recap
- Containers, Kubernetes and OpenShift
- Lab #3 description
- *In-class lab: Containers*

2

Labs/Class activities

3

- Lab 3: Cloud service integration
  - Released today – AWS
  - Due October 8
- October 8: Midterm + regular class session (1/2)

3

Midterm

4

- Midterm exam will take place on October 8
  - We will use the first half of the class for the exam
    - ~~Via Blackboard / open-book exam / synchronous~~
    - Paper-based, closed-book, in-class
  - Arrive early
  - Exam duration = 60 to ***75*** minutes
  - Exam will evaluate up to today's session content
    - Sessions 1 through 6
  - Question types:
    - Multiple choice
    - Short answer (2 to 5 lines)
    - Essay (1 to 2 paragraphs)

4

5

Recap

5

TCO

6

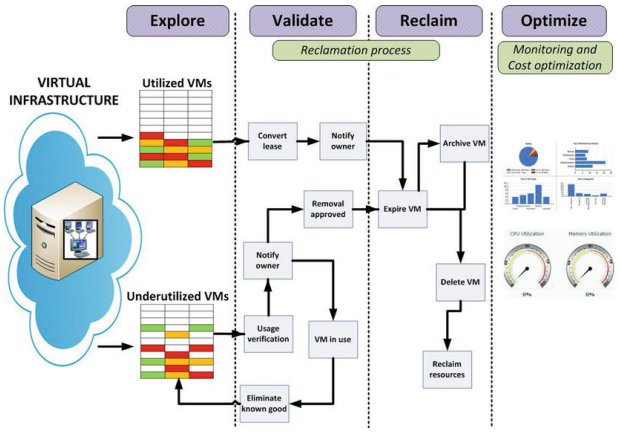
- Cloud's hidden costs
  - Service interruptions
  - Inappropriate service scaling
  - Denial of service attacks
  - Extra security
  - Disaster preparedness and recovery plan costs
  - Initial cost of cloud-readiness

6

Capacity management goals

7

- Efficient resource utilization
  - Manage workloads efficiently
- Reduce wasted resources
- Enable and support monitoring of service levels
- Controlling VM sprawl



Source: Cloud Capacity Management, P. Wali; N. Sabharwal, Apress, 2013

7

Capacity management goals (2)

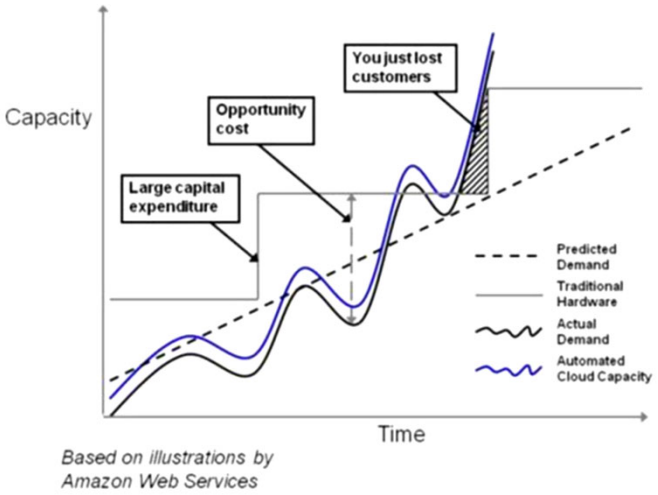
8

- Define rules for handling resources that fail
- Define auto-scaling mechanisms to handle unforeseen demand and/or seasonal variations in demand
- Ensure the operation of cloud/multi-cloud resources under appropriate economic constraints

8

Capacity utilization

9



Auto-scaling

9

Design for Capacity

10

- Cloud solution architecture, performance targets and costs have to be analyzed until a viable design is determined
- The elasticity of the cloud moves capacity planning/design form a worst-case (peak) scenario perspective to an automatic scaling (of capacity) perspective
  - Use of a few large units of capacity (large VMs) vs. many small units of capacity (small VMs)

10

Types of pricing in the Cloud (AWS)

11

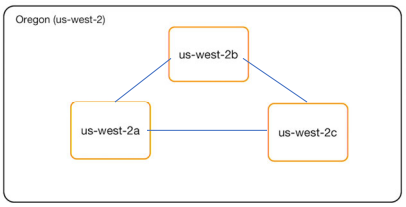
- On-Demand
- Reserved Instances
  - In AWS – Commitment levels: 1 year or 3 years
- Spot pricing
  - Bidding on spare capacity that the cloud provider has available
    - Can get up to 90% off the normal On-Demand price
  - Good for applications that have flexible start and end times
  - You don't pay more than your bid
    - You pay the market price until that price exceeds your bid
      - When the market price exceeds your bid, (in AWS), you have 2 minutes to shutdown your workload (or to migrate it).

11

AWS Regions

12

- https://aws.amazon.com/about-aws/global-infrastructure/regions_az/
- Each AWS Region consists of multiple, isolated, and physically separate Availability Zones within a geographic area.
  - Each AZ has independent power, cooling, and physical security and is connected via redundant, ultra-low-latency networks
  - Two to five AZs may be present in an AWS region
- Choosing between regions
  - Cost
  - Regulation (data sovereignty)
  - Latency

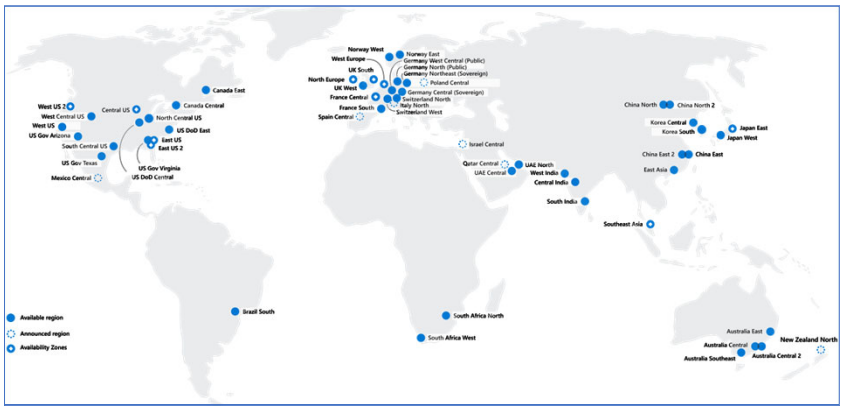


12

Azure -Regions

Azure offers more global regions than any other cloud provider with 60+ regions representing over 140 countries

- Regions are made up of one or more datacenters in close proximity.
- Provide flexibility and scale to reduce customer latency.
- Preserve data residency with a comprehensive compliance offering.



13

Understanding CPUs and vCPUs

14

- CPU cores vs threads
  - https://www.youtube.com/watch?v=hwTYDQ0zZOw
- The mapping/definition of a vCPU is dependent on the cloud provider and the virtual machine instance type
  - For EC2 general purpose instances (Spring/2021):
    - Each vCPU is a thread of either an Intel Xeon core or an AMD EPYC core, except for M6g instances, A1 instances, T2 instances, and m3.medium.
    - Each vCPU on T4g and M6g instances is a core of the AWS Graviton2 processor.
    - Each vCPU on A1 instances is a core of an AWS Graviton Processor.

14

Auto Scaling

15

The autoscaling process can:

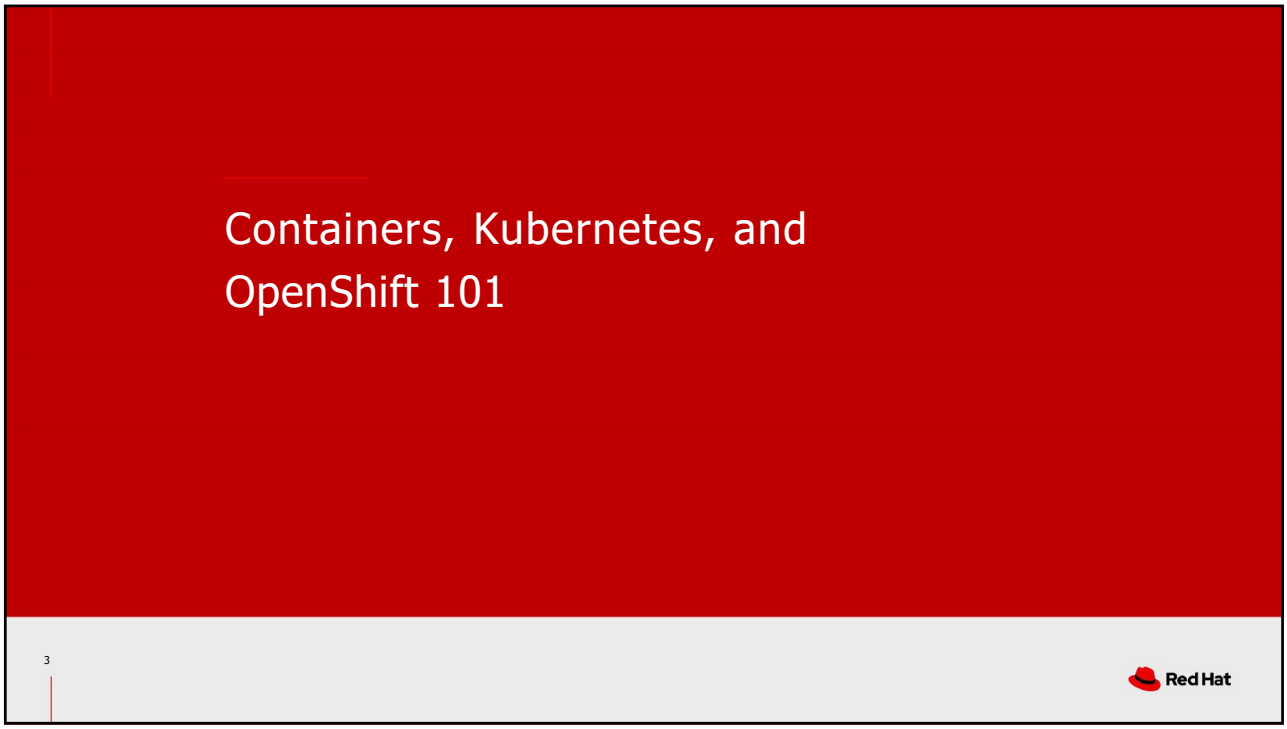
- Replace instances that have (performance) issues
  - Marked by the load balancer or hypervisor
  - Helps in building a self-healing infrastructure
- Maintain cluster size
- Grow cluster to meet increase in demand and scale down when needed

15

16

Containers, Kubernetes and OpenShift 101

16



17

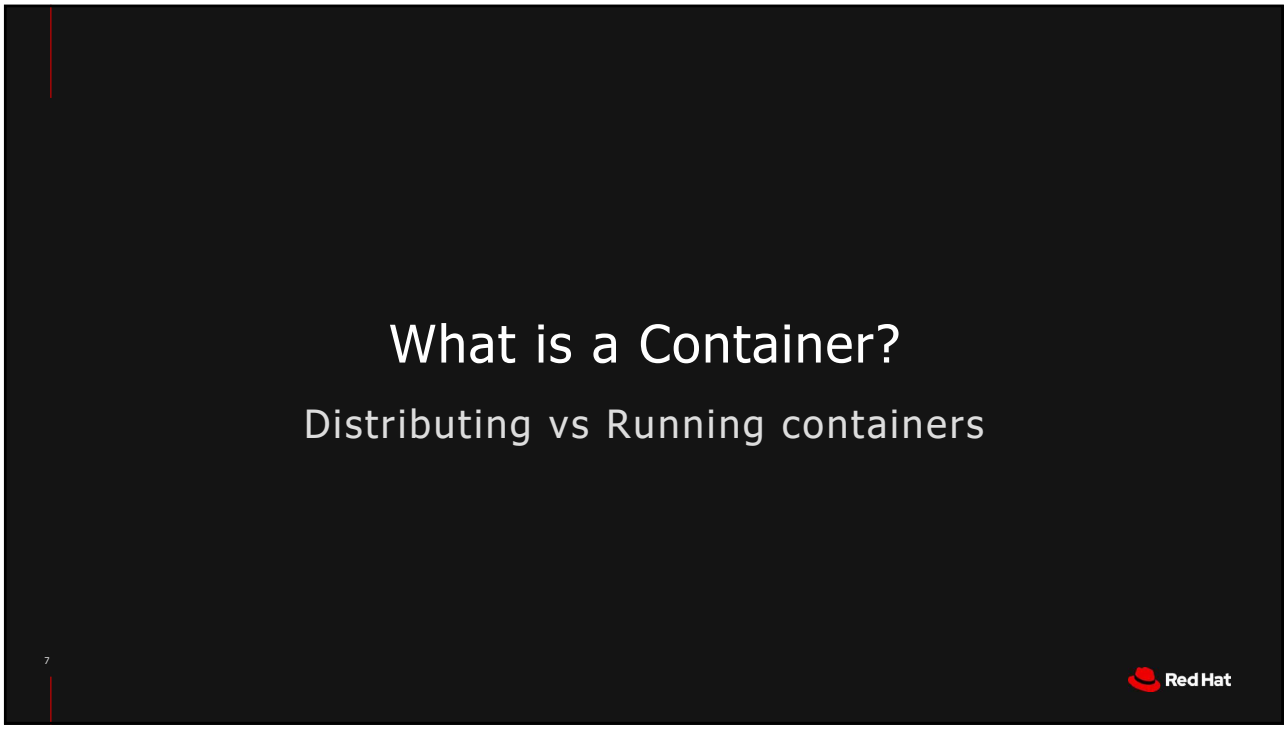
Trends on the IT field related to containers

- ✓ DevOps
- ✓ CI/CD
- ✓ Microservices
- ✓ AI/ML
- ✓ Cryptocurrency
- ✓ Cloud-native
- ✓ Multi-cloud
- ✓ Hybrid cloud
- ✓ 5G
- ✓ Edge computing

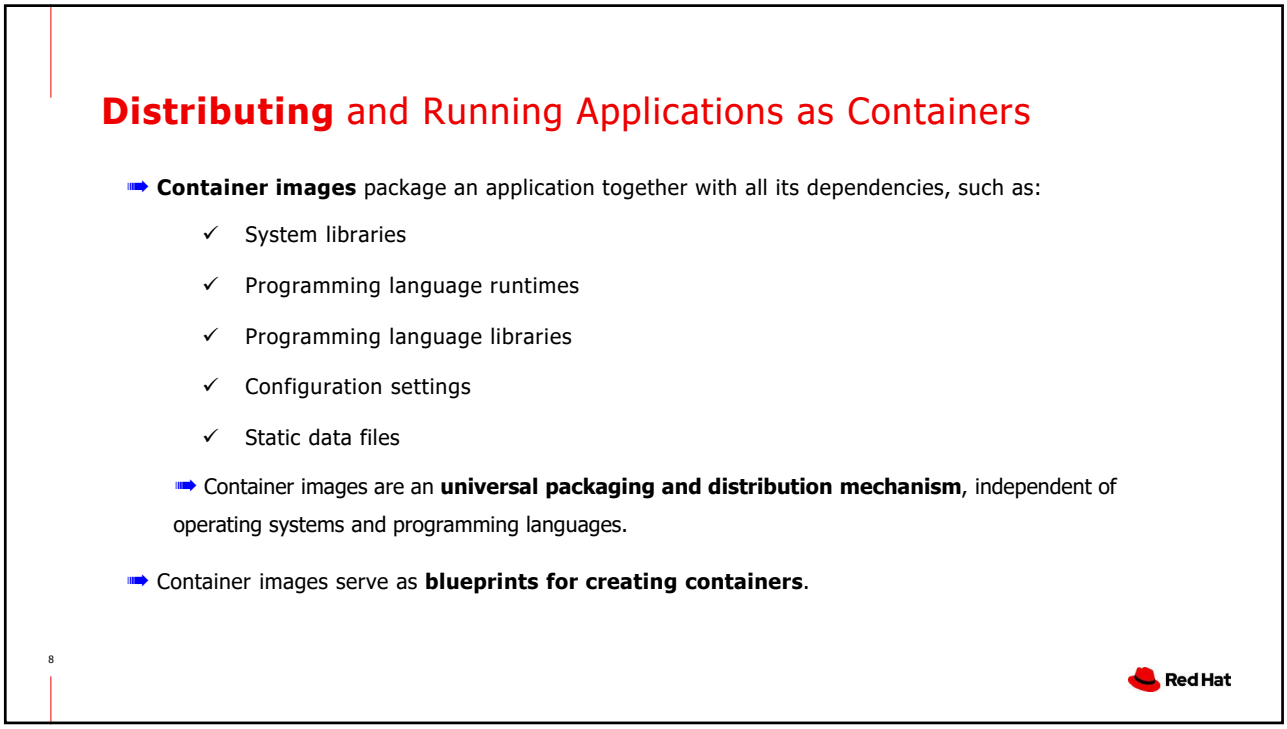
5

Red Hat

18


What is a Container?
Distributing vs Running containers
7
Red Hat

19


Distributing and Running Applications as Containers
Container images package an application together with all its dependencies, such as:
✓ System libraries
✓ Programming language runtimes
✓ Programming language libraries
✓ Configuration settings
✓ Static data files
Container images are an universal packaging and distribution mechanism, independent of operating systems and programming languages.
Container images serve as blueprints for creating containers.
8
Red Hat

20

Distributing and Running Applications as Containers

- Containers are operating system-level sandboxes that run applications isolated from each other and also from the core operating system services
- Containers create the illusion of an operating system dedicated to a single application and provide an exclusive set of operating system-managed resources, such as:
  - ✓ Processes
  - ✓ Filesystems (Files and folders)
  - ✓ Devices
  - ✓ Users and groups
  - ✓ Semaphores
  - ✓ Networks

9

Red Hat

21

Containers as a Packaging Mechanism

10

Red Hat

22

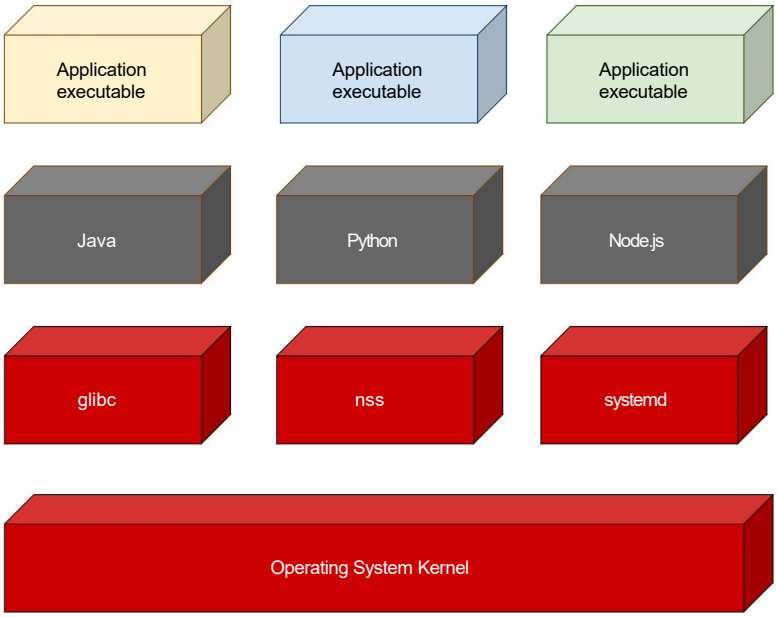
Containers are a standard way to **package** goods to ease **shipping** and delivery.

Software **containers** are a standard way to **package** applications to ease **deployment** and management.

11

Red Hat

23



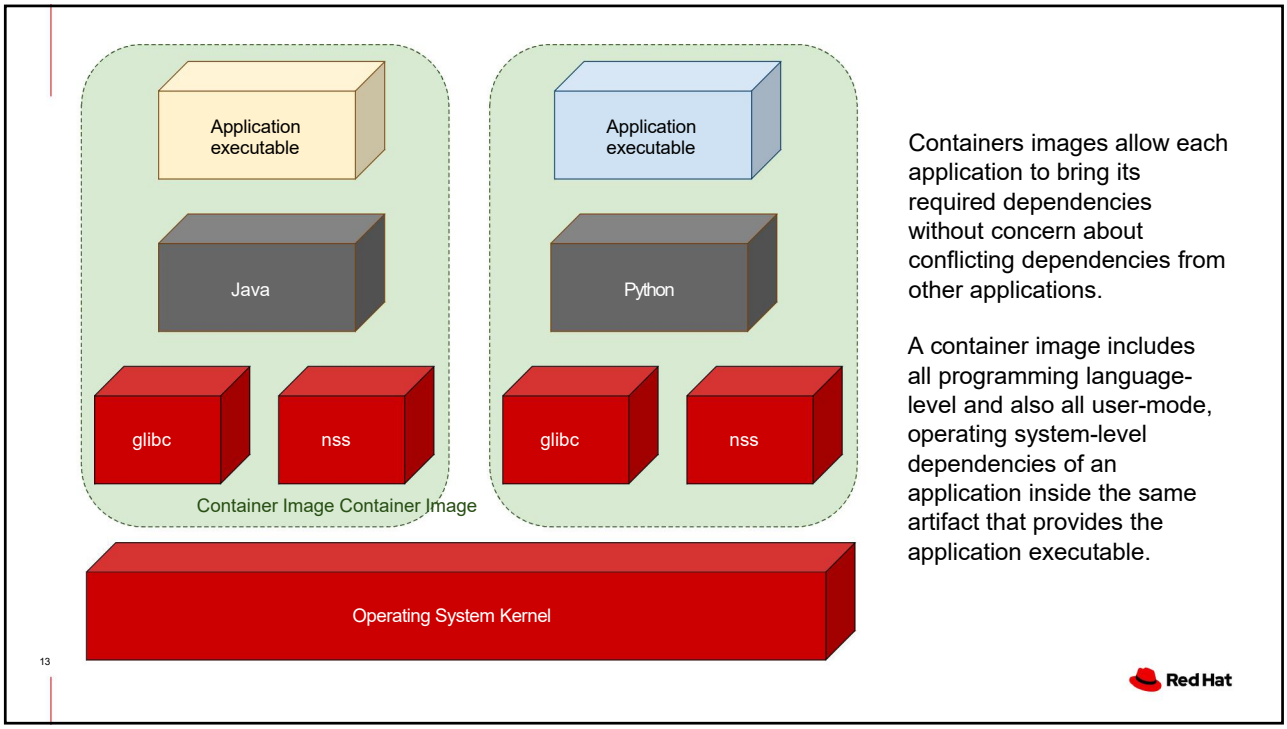
Without containers, applications share a number of operating system-level libraries and services, and also a number of programming language-level runtimes and libraries.

Making these shared libraries, services, and runtimes compatible with multiple applications on the same computer proved to be very hard.

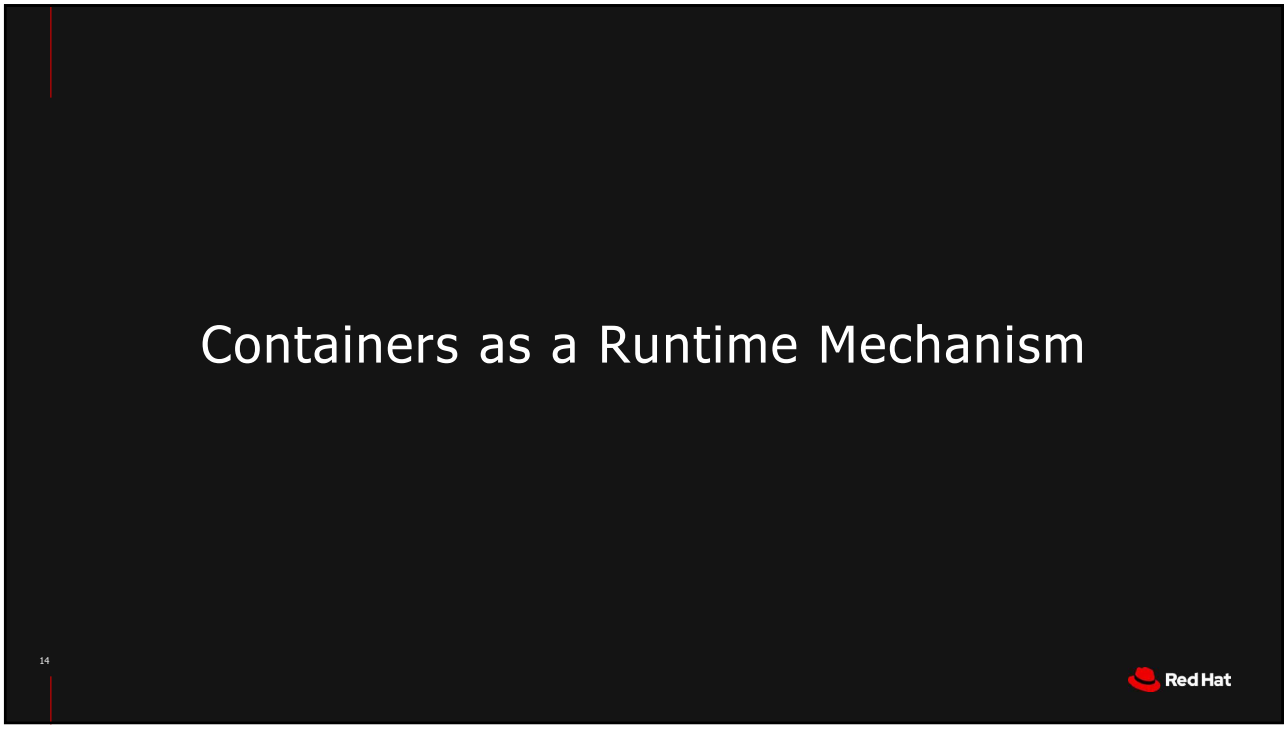
12

Red Hat

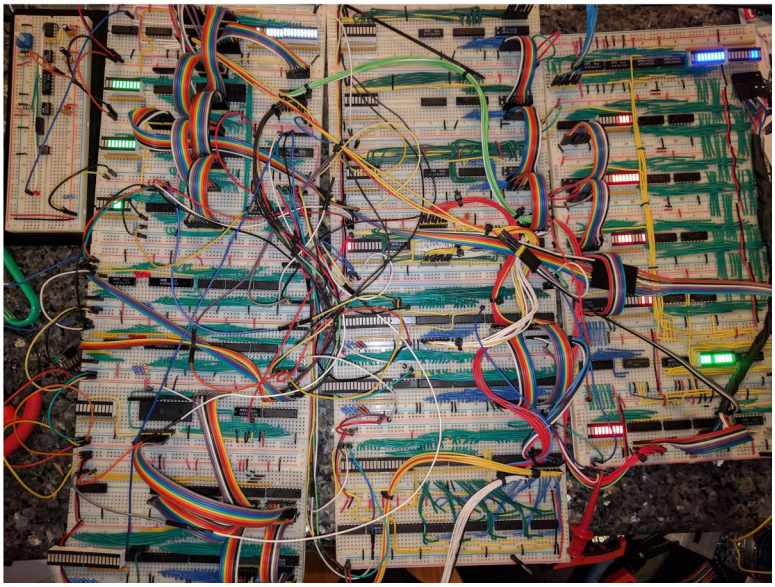
24


Application executable
Java
glibc
nss
Container Image
Application executable
Python
glibc
nss
Container Image
Operating System Kernel
13
Containers images allow each application to bring its required dependencies without concern about conflicting dependencies from other applications.
A container image includes all programming language-level and also all user-mode, operating system-level dependencies of an application inside the same artifact that provides the application executable.
Red Hat

25


Containers as a Runtime Mechanism
14
Red Hat

26

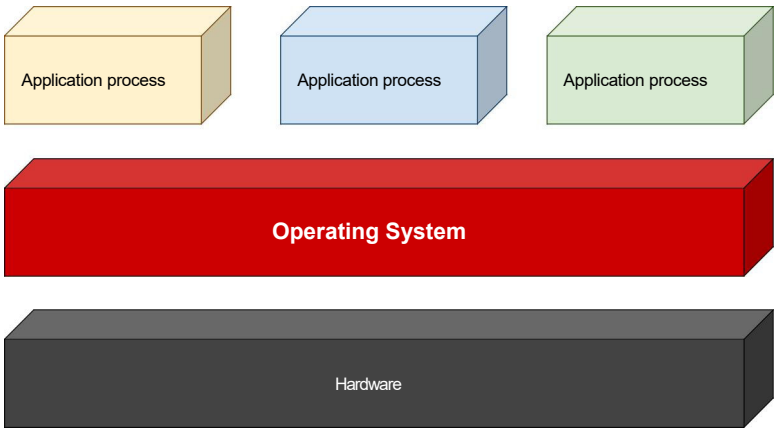


In the beginning, applications had complete control over the hardware. There was no sandbox protecting an application.

15

Red Hat

27



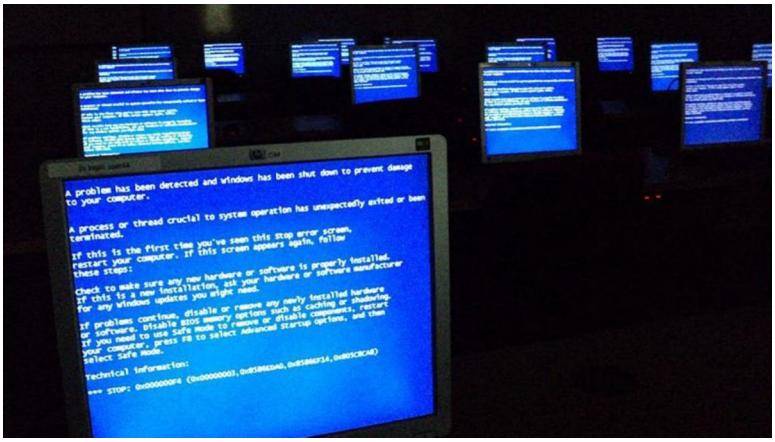
Then **operating systems** came and made applications run inside sandboxes called **processes** that prevented one application from accessing the memory of other applications.

Processes were also forced to release the CPU to other processes after a time slice.

16

Red Hat

28



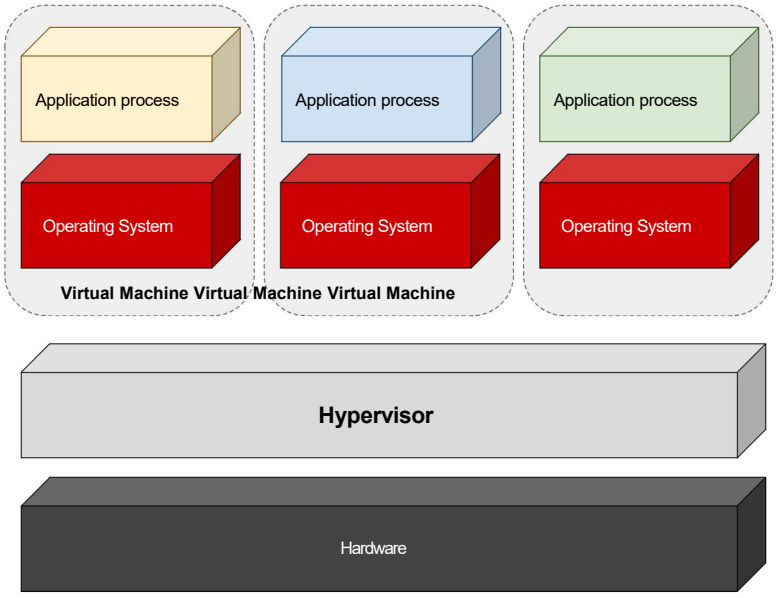
The process sandbox was not strong enough. Applications could still interfere with each other and the operating system by indirect means.

The contact surface between applications was still too large.

17

Red Hat

29

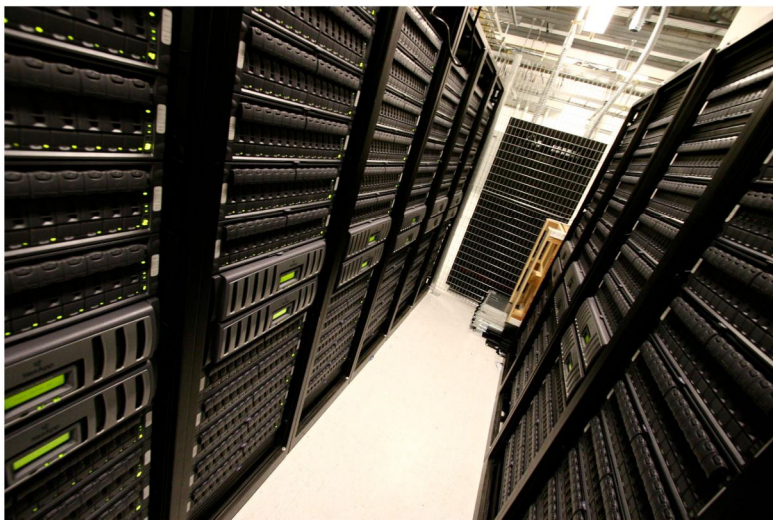


Then came the **hypervisor** and put both applications and their required operating systems inside sandboxes named **virtual machines**.

18

Red Hat

30



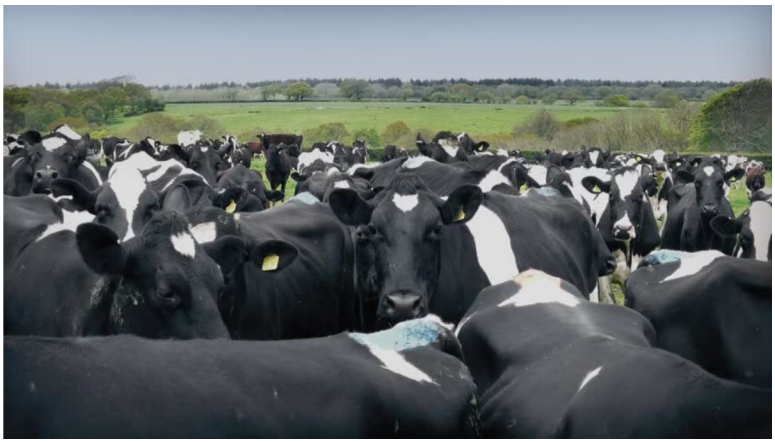
With more computers and more virtual machines, the larger cost factor changed from **computers** to **people**.

Also, the reliability of **people** instead of **software and hardware** become the largest risk factor.

19

Red Hat

31



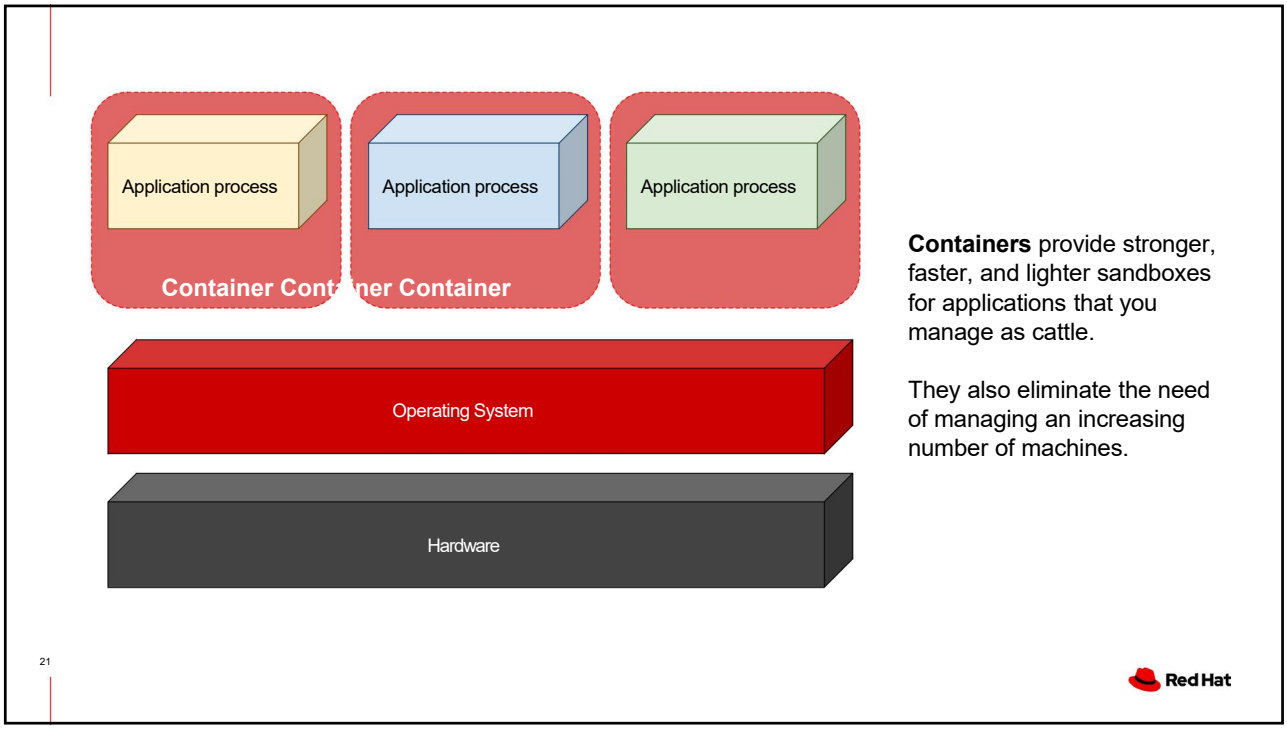
The solution proposed by the cloud and agile practitioners is managing applications as **cattle** instead of not as pets.

Seeing all servers as indistinct from each other leads to easier automation and thus making people more scalable and reliable.

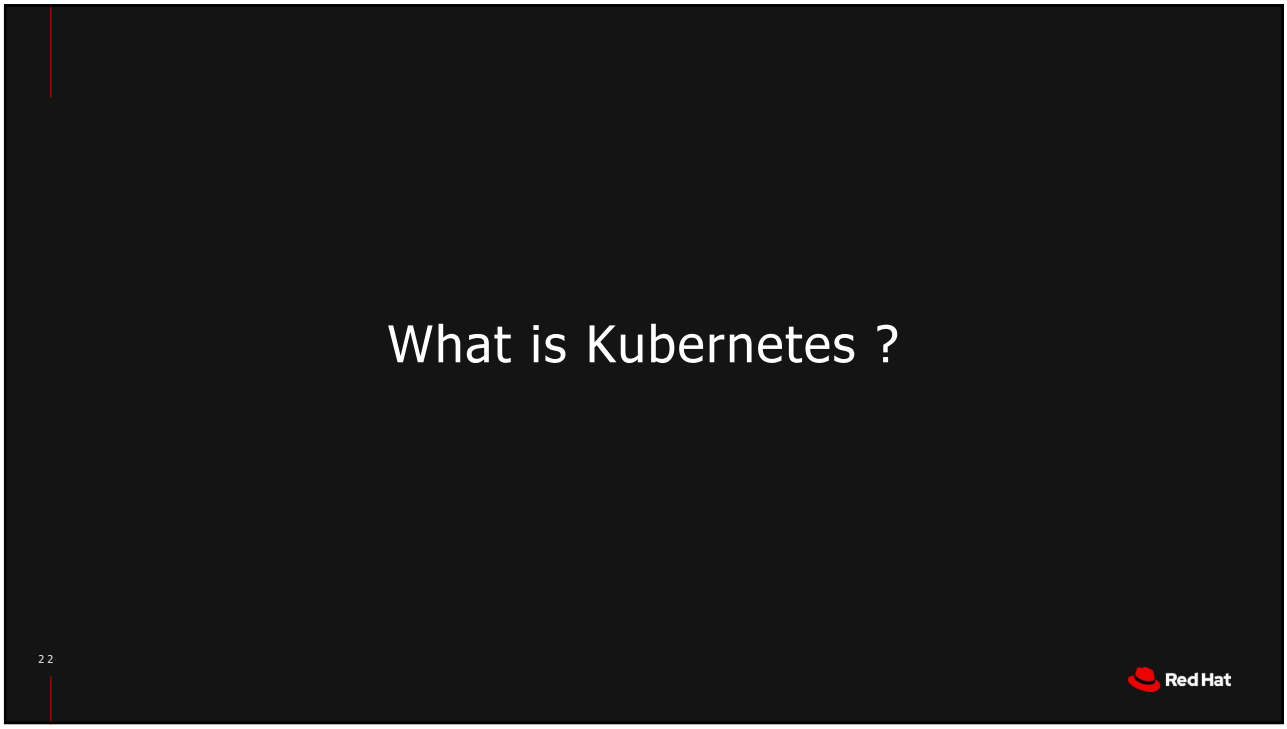
20

Red Hat

32


Application process
Application process
Application process
Container Container
Operating System
Hardware
Containers provide stronger, faster, and lighter sandboxes for applications that you manage as cattle.
They also eliminate the need of managing an increasing number of machines.
21
Red Hat

33


What is Kubernetes ?
22
Red Hat

34

Cloud and mobile applications are usually composed of multiple **microservices** that collaborate with each other.

23

Red Hat

35

The more pieces you have, the more things could go wrong...

24

Red Hat

36

You need a good **helmsman** to take care of all containers on your ship.

25

Red Hat

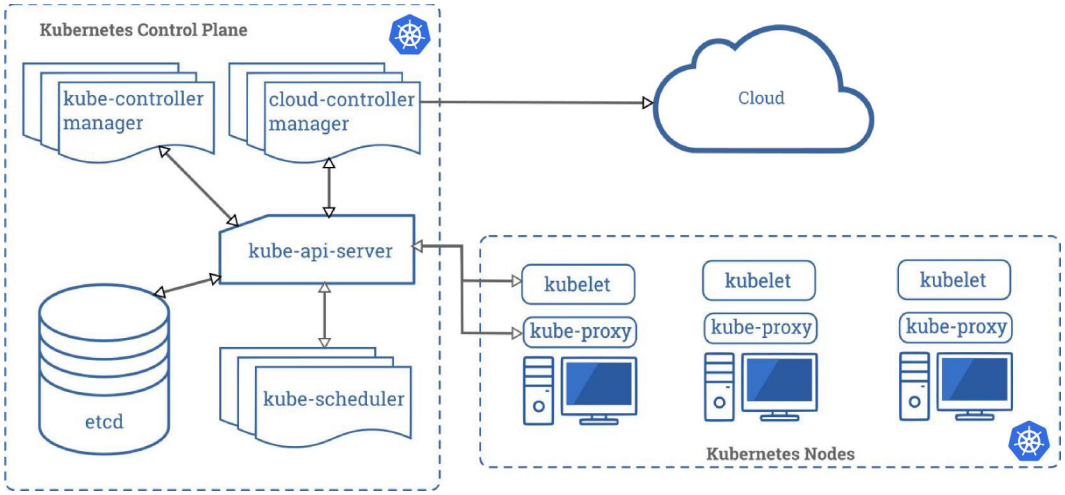
37

You need a *good* **Kubernetes** to take care of all containers on your cluster of computers.

26

Red Hat

38



It looks harder than it actually is.
It is about *essential* complexity instead of *accidental* complexity

27

Red Hat

39

What Does Kubernetes Do for You?

- ✓ Find a computer in a cluster with sufficient available capacity to run a container for an application.
- ✓ Route network connections between applications to the right container no matter in which computer it is running.
- ✓ Run more or less containers of the same application according to demand (scaling).
- ✓ Load balance network connections among multiple containers of the same application.
- ✓ Start a new container to replace a failed container (or all containers from a failed computer).
- ✓ Re-attach applications to its data when their containers restart in different computers.

https://www.youtube.com/watch?v=IMOZCDhH7do

28

Red Hat

40

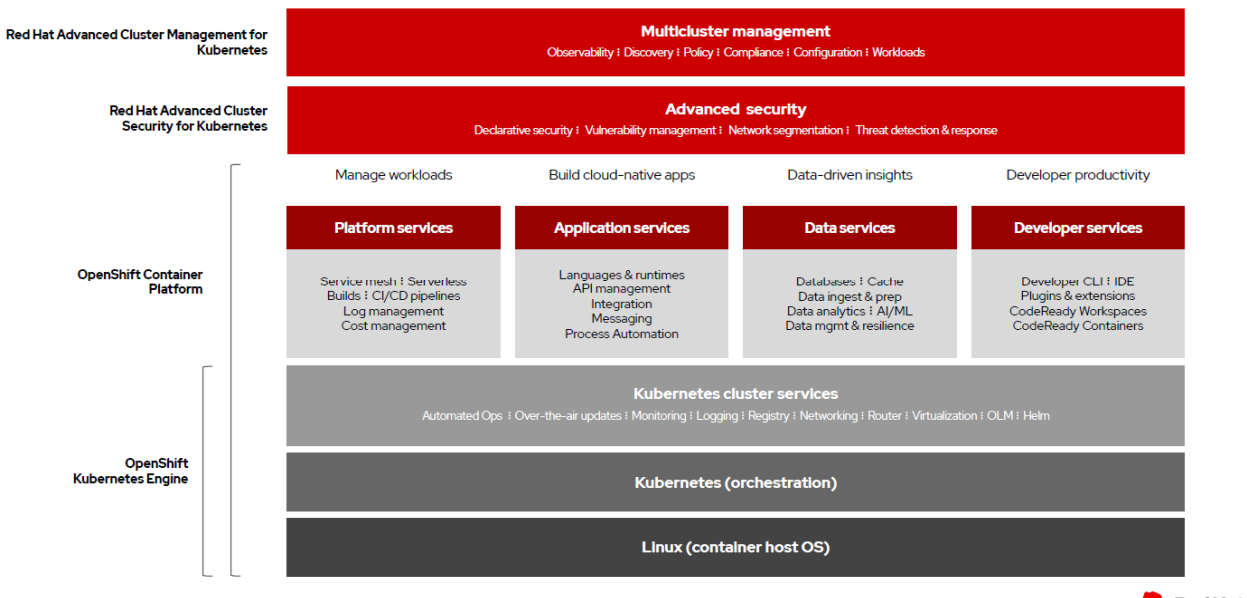
What is OpenShift ?

29

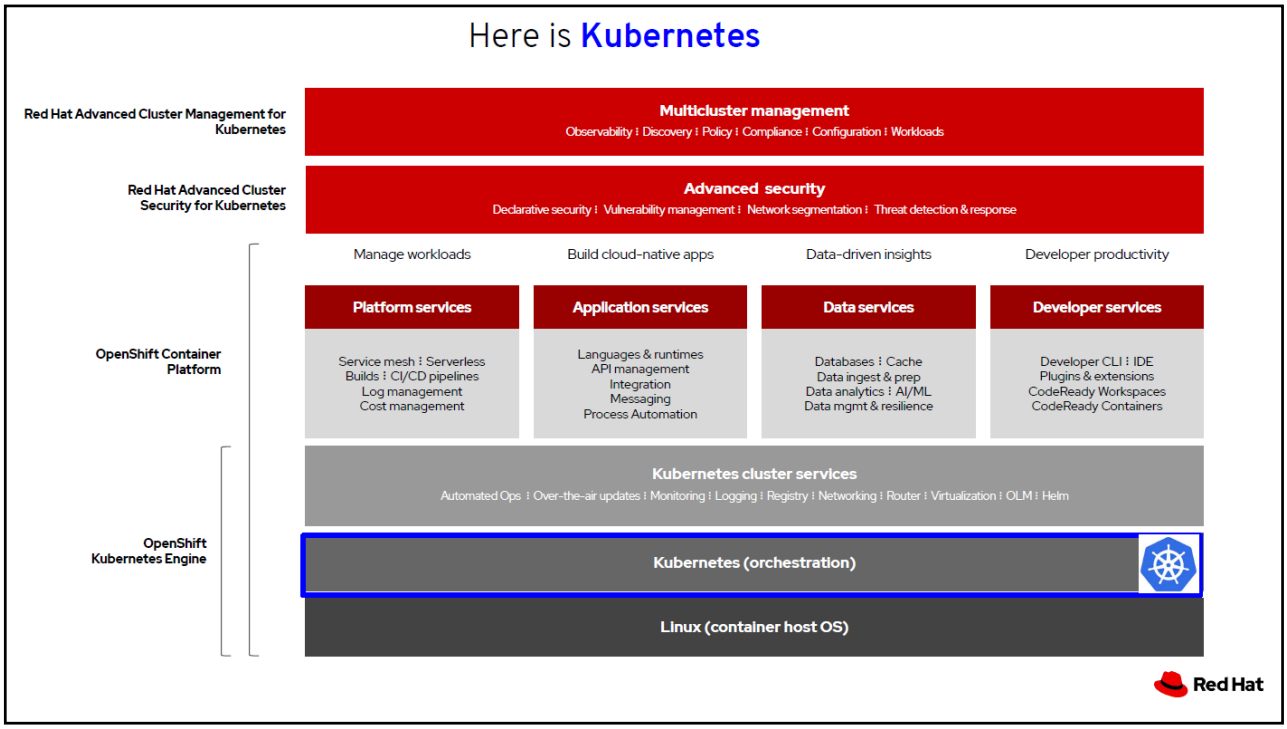
Red Hat

41

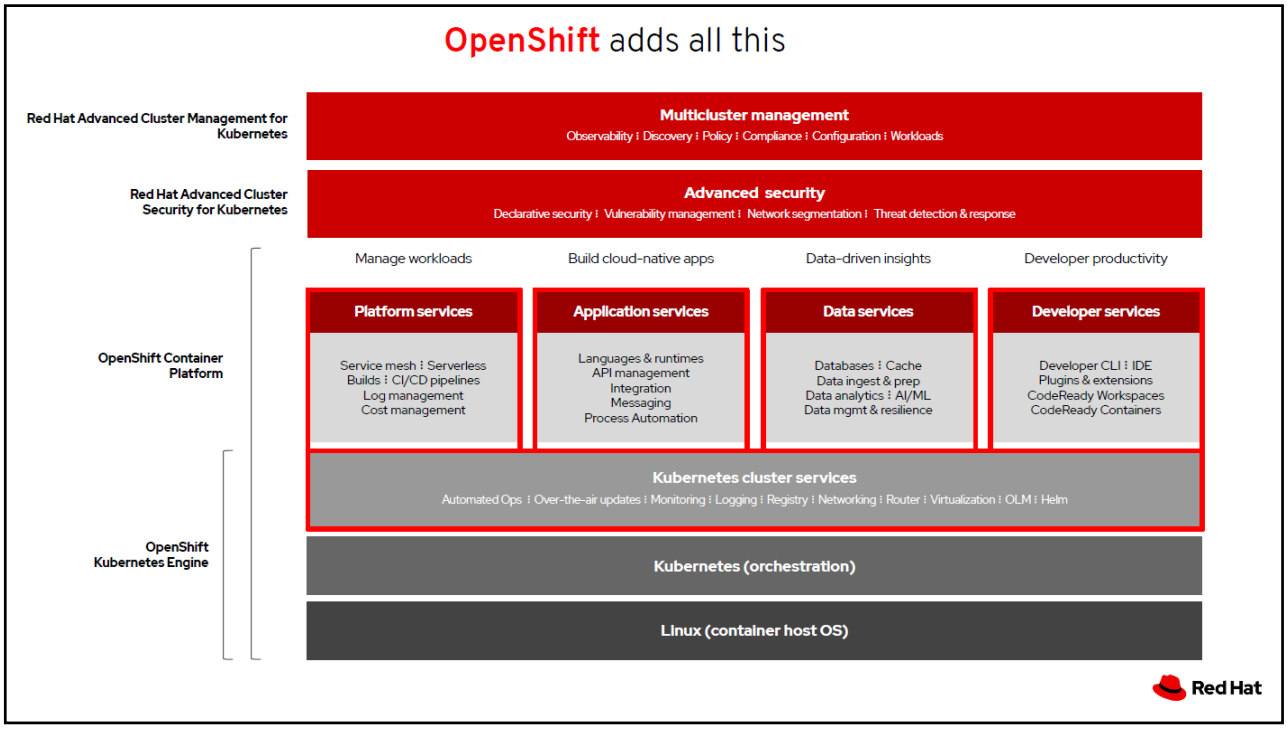
OpenShift is Red Hat's enterprise distribution of Kubernetes



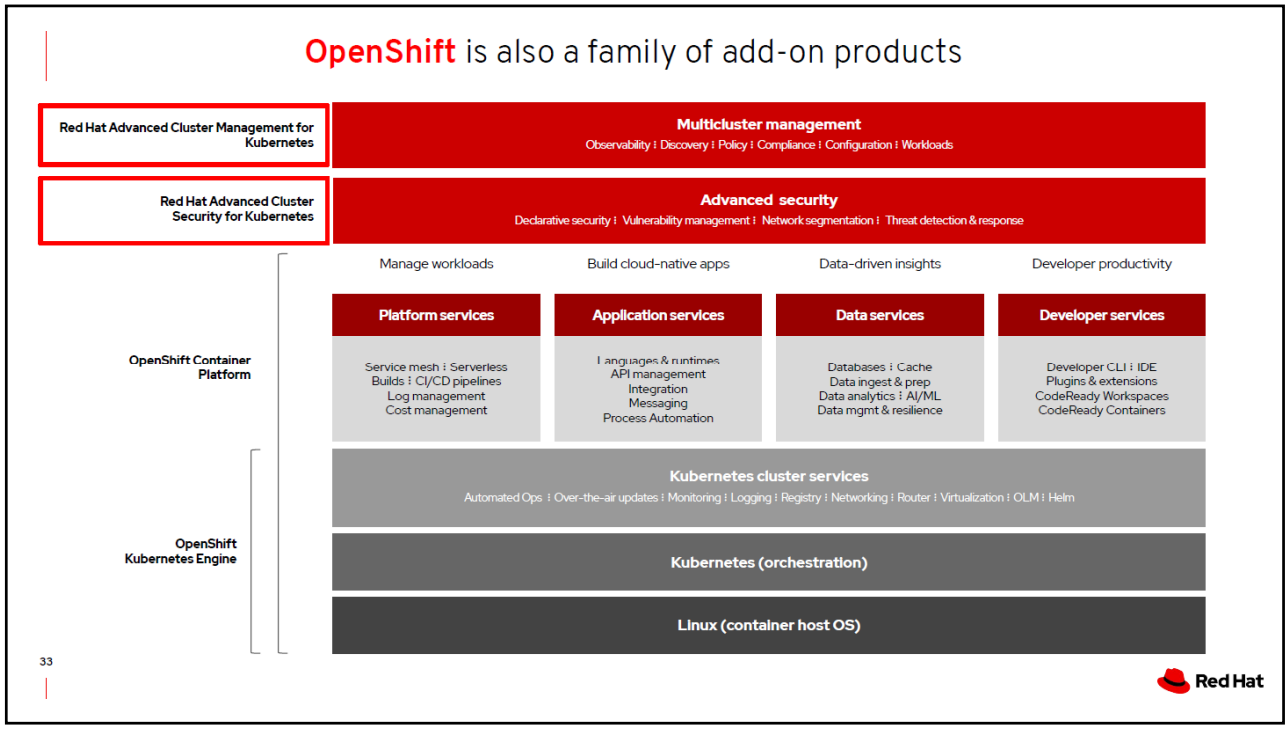
42


Here is Kubernetes
Red Hat Advanced Cluster Management for Kubernetes
Multicluster management
Observability ⁞ Discovery ⁞ Policy ⁞ Compliance ⁞ Configuration ⁞ Workloads
Red Hat Advanced Cluster Security for Kubernetes
Advanced security
Declarative security ⁞ Vulnerability management ⁞ Network segmentation ⁞ Threat detection & response
OpenShift Container Platform
Manage workloads
Build cloud-native apps
Data-driven insights
Developer productivity
Platform services
Application services
Data services
Developer services
Service mesh ⁞ Serverless
Builds ⁞ CI/CD pipelines
Log management
Cost management
Languages & runtimes
API management
Integration
Messaging
Process Automation
Databases ⁞ Cache
Data ingest & prep
Data analytics ⁞ AI/ML
Data mgmt & resilience
Developer CLI ⁞ IDE
Plugins & extensions
CodeReady Workspaces
CodeReady Containers
OpenShift Kubernetes Engine
Kubernetes cluster services
Automated Ops ⁞ Over-the-air updates ⁞ Monitoring ⁞ Logging ⁞ Registry ⁞ Networking ⁞ Router ⁞ Virtualization ⁞ OLM ⁞ Helm
Kubernetes (orchestration)
Linux (container host OS)
Red Hat

43


OpenShift adds all this
Red Hat Advanced Cluster Management for Kubernetes
Multicluster management
Observability ⁞ Discovery ⁞ Policy ⁞ Compliance ⁞ Configuration ⁞ Workloads
Red Hat Advanced Cluster Security for Kubernetes
Advanced security
Declarative security ⁞ Vulnerability management ⁞ Network segmentation ⁞ Threat detection & response
OpenShift Container Platform
Manage workloads
Build cloud-native apps
Data-driven insights
Developer productivity
Platform services
Application services
Data services
Developer services
Service mesh ⁞ Serverless
Builds ⁞ CI/CD pipelines
Log management
Cost management
Languages & runtimes
API management
Integration
Messaging
Process Automation
Databases ⁞ Cache
Data ingest & prep
Data analytics ⁞ AI/ML
Data mgmt & resilience
Developer CLI ⁞ IDE
Plugins & extensions
CodeReady Workspaces
CodeReady Containers
OpenShift Kubernetes Engine
Kubernetes cluster services
Automated Ops ⁞ Over-the-air updates ⁞ Monitoring ⁞ Logging ⁞ Registry ⁞ Networking ⁞ Router ⁞ Virtualization ⁞ OLM ⁞ Helm
Kubernetes (orchestration)
Linux (container host OS)
Red Hat

44



45

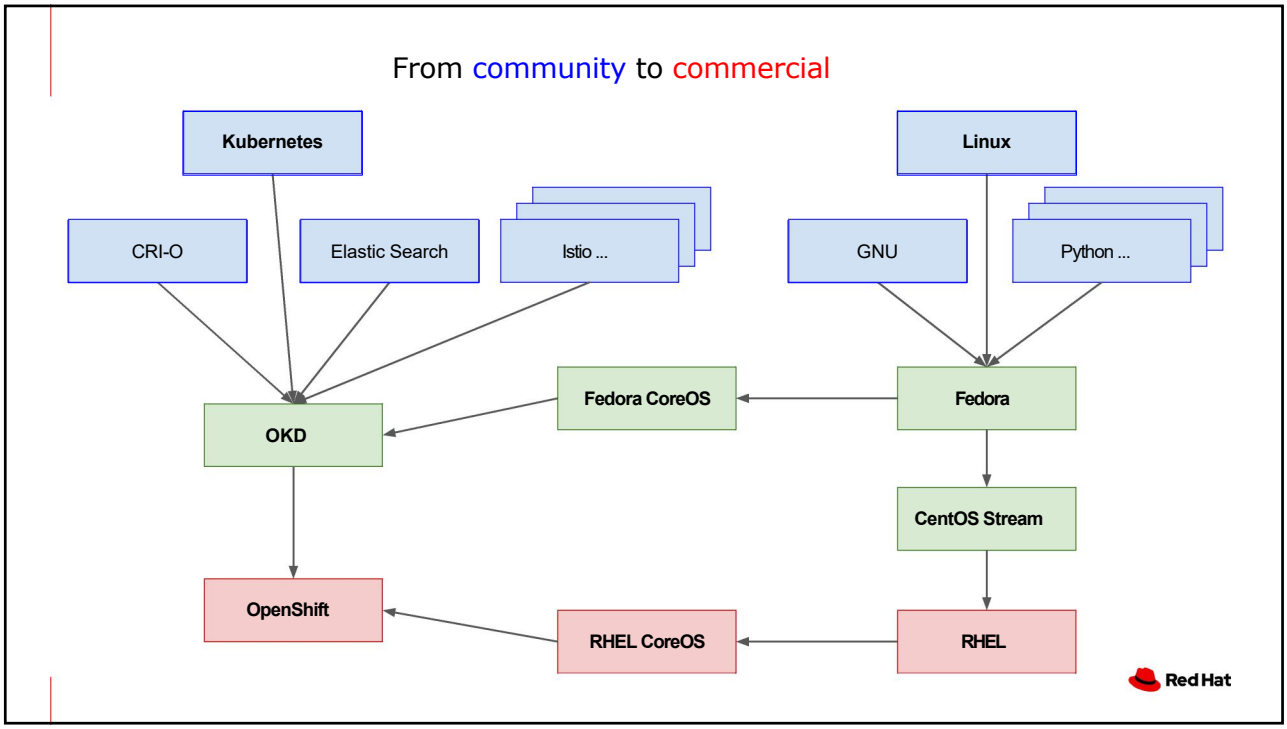
What OpenShift Does for You Beyond Pure Kubernetes

- ✓ Provide a registry server to store and distribute container images.
- ✓ Build container images from source code and CI/CD pipelines.
- ✓ Add strong security controls based on roles and authentication providers.
- ✓ Add centralized logging, monitoring, alerts, and metrics.
- ✓ Manage installation and patching of the operating system on computers in the cluster.
- ✓ Route and load balance network connections from outside the cluster to the correct application container inside the cluster.
- ✓ Integrate with network and storage providers for popular virtualization, cloud, and hardware vendors.

34

Red Hat

46


From community to commercial
Kubernetes
CRI-O
Elastic Search
Istio ...
Linux
GNU
Python ...
Fedora CoreOS
Fedora
OKD
CentOS Stream
OpenShift
RHEL CoreOS
RHEL
Red Hat

47


Containers Are Linux
37
Red Hat

48

Running Containers on My Computer

- **Container runtimes** and **Container Engines** are just tools that ask the Linux kernel to start an application inside a container sandbox. There are many to choose from.
- Standards from the **Open Container Initiative** (https://opencontainers.org) allow interoperability between different tools that produce and consume the same containers and container images.
  - The Red Hat ecosystem of operating systems (Fedora, CentOS, and Red Hat Enterprise Linux) come with the *Linux Container Tools* app stream: **podman**, **skopeo**, and **buildah**.
- To use, on your non-Linux work or personal machine, the same tools you would use in a production RHEL system, either configure your own virtual machine or install **Code Ready Containers** (CRC) that does that for you.

39

Red Hat

49

Linux Container Tools vs Docker

Linux Container Tools	Docker
✗ Little known outside of the RHEL, CentOS, and Fedora ecosystems	✓ Well-known among developers of all major platforms
✗ Still immature clients for Windows and Mac	✓ Good clients for Mac and Windows (daemon runs inside a Linux VM)
✓ Set of single-purpose, standalone tools	✗ Monolithic daemon that performs all tasks
✓ Easier to bug fix and secure, runs rootless	✗ "Big ball of mud" issues to fix and secure, daemon has to run as root
✓ Strong backwards-compatibility record and guarantees	✗ Incompatible changes pop-up by surprise (ask the Kubernetes developers)
✓ Strong community and sponsors	✗ Sponsor is financially challenged and not very engaged with the community

Red Hat

50

Lab #3 Description

51

- AWS S3 (Simple Storage Service)
  - https://www.youtube.com/watch?v=_I14_sXHO8U
- AWS IAM (Identity and Access Management)
  - https://www.youtube.com/watch?v=UI6FW4UANGc
- Lab 3 posted in Blackboard
  - Due October 8
  - Use of AWS S3 storage service
  - Integrating two AWS services to work together (EC2 & S3) via IAM
  - Additional details
    - Follow the instructor's description in class

51

In-class lab

52

- Hands-on activity
  - In-class lab on containers
    - Follow lab guide posted in Blackboard

52