NUTANIX PRESENTS

SILVERBACK EDITION

THE GORILLA GUIDE TO...®

Cloud Services and Beyond

Dan Sullivan, Ed Tittel, Joep Piscaer, Max Mortillaro

INSIDE THE GUIDE:

- The Benefits of Cloud Automation and Orchestration
- Staying Safe in the Cloud
- Controlling and Managing Cloud Costs

HELPING YOU NAVIGATE THE TECHNOLOGY JUNGLE!

ActualTech Media
www.actualtechmedia.com

In Partnership With

NUTANIX™

THE GORILLA GUIDE TO...

Cloud Services and Beyond

AUTHORS
Dan Sullivan, Ed Tittel, Joep Piscaer, Max Mortillaro

EDITOR
Keith Ward, ActualTech Media

PROJECT MANAGER
Wendy Hernandez, ActualTech Media

EXECUTIVE EDITOR
James Green, ActualTech Media

LAYOUT AND DESIGN
Olivia Thomson, ActualTech Media

ACTUALTECH MEDIA
6650 Rivers Ave Ste 105 #22489
North Charleston, SC 29406-4829
www.actualtechmedia.com

ENTERING THE JUNGLE

Introduction: Level Up Your Cloud Game 8

Chapter 1: Understanding Why APIs Are Critical to the Modern Enterprise 10

Digital Transformation: Altering the Business Landscape 11

Toward Infrastructure as Code 14

APIs Are Interfaces Between Microservices 15

Essential Automation 18

Chapter 2: Enterprise Cloud Automation and Orchestration 20

Managing Enterprise Cloud Automation and Orchestration 21

App Store for Enterprise IT 23

Any App, Any Cloud 30

App Store for Any Team 33

Get in the Driver's Seat 36

Chapter 3: Monitoring and Controlling Cloud Spend 38

Understanding Cloud Spend Trends 40

Where and How Organizations See and Use the Cloud 45

Business/Organization Segmentation 46

Optimizing Cloud Spend 50

Chapter 4: Automated Cloud Security Audits and Compliance Validation 55

Introduction to Multi-Cloud 56

Security Challenges 57

Risk Management 58

Complexity of IT and Multi-Cloud 58
Cloud-Native and Developer-First: Security at Risk 59
Global Visibility, Granular Enforcement 60
Start with Visibility 60
Monitor and Visualize the Network 61
Segment Risk 62
Policy-Based Security 63
Automatically Enforcing Security Policy at the Granular Level 64
Always-on Cloud Security Compliance Enforcement 65
Ensuring Compliance with Regulatory Policies 66
Policy-Based Management Is Key 67

Chapter 5: Architecting a Cloud Disaster Recovery Service 68
Business Considerations When Choosing a DR Solution 69
Traditional Pre-Cloud DR Solutions 71
First-Generation Cloud-Based DR Solutions 73
Integrated Cloud-Based DR Solutions 74
The Best of Both Models 75

Chapter 6: Managing Databases in the Cloud Era 77
Digital Transformation of Business Is Enabled by Agile Development 78
Database Provisioning Is a Bottleneck 79
Managing Databases across Platforms 80
Automation Policies 83
New Ways to Think About Databases 85

Chapter 7: How Containers Are Reshaping the Enterprise 87
Sketching the Cloud-Native Landscape 88
The Move to Containers 93

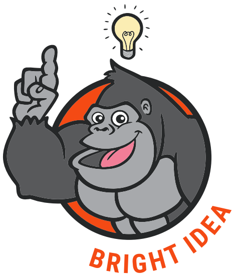
Kubernetes Concepts and Architecture....99
Kubernetes Solutions....105
Choosing the Right Solution....107

Chapter 8: Unleashing the Power of Edge/IoT....111
Defining Our Terms....111
The Impact of Edge/IoT....113
Use Cases for Edge/IoT....114
Challenges of Edge/IoT....121
4 Considerations for Designing Successful Edge Solutions....124
Managing IoT and Edge Computing with Nutanix Xi IoT....126
Getting Started with Xi IoT....128
The End of the Beginning....129

CALLOUTS USED IN THIS BOOK

The Gorilla is the professorial sort that enjoys helping people learn. In the School House callout, you'll gain insight into topics that may be outside the main subject but are still important.

This is a special place where you can learn a bit more about ancillary topics presented in the book.



When we have a great thought, we express them through a series of grunts in the Bright Idea section.

Takes you into the deep, dark depths of a particular topic.

Discusses items of strategic interest to business leaders.

ICONS USED IN THIS BOOK

DEFINITION

Defines a word, phrase, or concept.

KNOWLEDGE CHECK

Tests your knowledge of what you've read.

PAY ATTENTION

We want to make sure you see this!

GPS

We'll help you navigate your knowledge to the right place.

WATCH OUT!

Make sure you read this so you don't make a critical error!

INTRODUCTION

Level Up Your Cloud Game

Welcome to this Gorilla Guide To...® Cloud Services and Beyond! The purpose of this book is to show you how hyperconverged infrastructure, or HCI, can enable the next era of computing, which will have a strong focus on extending data centers into the cloud: whether that's strictly public cloud, strictly private cloud, or a hybrid version of the cloud that has both public and private components.

If you're in IT, you probably don't need to be told about the basic benefits cloud computing brings to an environment: they're touted everywhere, from a multitude of vendors. But how to get work done in this complicated new realm is much less well known. Not only are there the types of clouds mentioned earlier, but more and more, companies are also employing multiple public clouds, making things even more complicated than adding cloud to the mix already made them.

This book will provide essential insight into making proper use of various types of clouds, including multi-cloud scenarios. The information runs the gamut from APIs to the edge and Internet of Things (IoT), and covers traditional topics like cost control, database management, and disaster recovery from a cloud-based perspective.

When you're done with this Gorilla Guide, you'll have a solid understanding of the potential of the cloud, as well as the risks inherent in giving up some control of your data center operations (not to mention your data!) to others. It's a scary new world: one with great advantages, but also great potential to disrupt your infrastructure if you don’t do it right. Use this guide as a means to maximize the upside of cloud and minimize its downsides.

Ready to go? Then let’s get started. We’ll begin with one of the key factors to facilitating cloud usage: APIs.

CHAPTER 1

Understanding Why APIs Are Critical to the Modern Enterprise

Digital transformation is enabling businesses to create a range of new products and services at an accelerating pace. To capitalize on the opportunities presented by this shift, organizations need to be more flexible and adaptive than in the past. The term "Agile" is often used to describe this new dynamic mode. To be agile is to not only see openings in the market, but also able to design and execute plans and operations to benefit from them. This requires both organizational and technological agility. One element of agility is the ability to deploy compute and storage infrastructure as needed to support applications and services.

Many business systems in use today were designed as monolithic applications that combine data management, business logic, and user interface functionality into a single software application. This model works well when there are few distinct ways to use the software or any of its components, but that is often not the case.

Instead of isolating key functionality in a single application silo, developers are choosing to expose functions and services through application programming interfaces (APIs). This is enabling new ways to combine functional components to more rapidly build and deliver enterprise services.

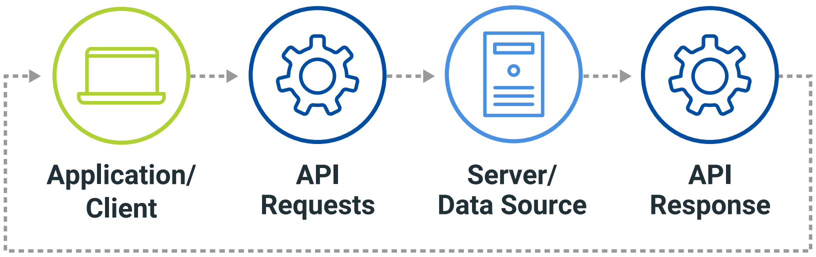


Figure 1: Simplified overview of how an API works.

APIs are changing the way organizations provision and manage their compute, storage, and network resources. **Figure 1** provides an overview.

Virtualization and cloud technologies enable more efficient and adaptive use of infrastructure, but they also present opportunities for inefficient allocation of resources. The challenge of controlling costs in public clouds is well known. Organizations need tools and practices for orchestrating the use of virtualized resources. Fortunately, APIs for creating, managing, and monitoring clouds, servers, containers, and other virtualized resources provide the foundation for those needed tools and practices.

To understand the need for infrastructure APIs, it helps to understand the business drivers and related technologies that are making them necessary.

Digital Transformation: Altering the Business Landscape

Enterprises have been developing and maintaining applications for decades. Many applications were designed around technologies equally as antiquated, including mainframe computers. They also designed applications as monolithic systems rather than ensembles of services interacting through APIs.

These applications include batch workloads, the workhorse of business computing, which process insurance claims, generate monthly account statements, and extract, transform, and load data to data warehouses. There are also more transaction-oriented applications, such as those used in support of back-office operations, including accounting, inventory, and order processing. These are still essential kinds of applications, but they are no longer the only type of application and not the kind that give a business competitive advantage.

Advances in hardware, software, and networking have enabled new ways of developing applications and delivering services. These new types of applications are creating an ecosystem outside the monolithic applications of the back office. Examples include web, mobile, IoT, and machine learning apps. Understanding the needs of these new kinds of applications will shed light on the need for more agile infrastructure management capabilities, particularly the need for APIs to enable diverse sets of services to function together.

To say that the web has transformed the way businesses reach customers is an understatement. The technology enables direct customer interaction. For example, retailers now sell directly to customers online with or without a physical store nearby. E-commerce applications also provide for a more customized experience for the consumer as well as access to large volumes of detailed data about customer interests. The data collected about user preferences and demographics are what enable these more personalized experiences.

These personalized services, however, require more functionality than found in a typical solution for tracking sales transaction. With APIs, those transaction processing systems can easily make use of recommendation software that may not have been anticipated when the original system was developed.

Mobile apps are also changing the way businesses interact with customers and provide even more data about them. For example, mobile apps can collect customer interaction details and geo-location data over extended periods of time, which provide further insights into customer interests.

They also enable a kind of always-on access to app services for customers, which in turn creates the need for always-on processing infrastructure to support those apps. Unlike with traditional business applications, predicting the level of use of mobile applications can be quite difficult.

There may be periods of high use, which would require additional infrastructure, as well as low utilization periods, which would require fewer infrastructure resources. Here again, APIs enable a new way to deliver services. Cloud providers allow customers to adjust their infrastructure through APIs. More resources are added when needed and released when workload drops programmatically through cloud APIs.

The Internet of Things (IoT) is another driver for more flexible and scalable infrastructure. IoT devices, such as sensors, monitor equipment and environmental conditions in ways that generate constant streams of time-series data.

This data needs to be ingested, stored, and analyzed at scales few organizations have had to address in the past. Time-series data is potentially much larger than other business data sets, and is typically queried and analyzed differently than other business data.

The benefit, of course, is that IoT data from equipment and machinery can be used to operate more efficiently and to predict failures and detect anomalies before they disrupt operations.

Analytics and machine learning are other emerging sets of technologies that use large volumes of detailed data. Advances in data science and machine learning are bringing solutions to new domains, such as machine vision and natural language processing, and enabling new kinds of products and services.

These techniques enable services such as recommendations for e-commerce customers, predictive analytics to help optimize just-in-time production and inventory management, and fraud detection in financial services operations. Machine learning is increasingly incorporated into workflows and applications like e-commerce sites and credit card transaction processing.

There are clear business opportunities and benefits to adopting web, mobile, IoT, and machine learning technologies, but they require a reliable and scalable infrastructure. Providing and managing that infrastructure efficiently has been one of the more difficult aspects of digital transformation. It does not have to be. Infrastructure orchestration based on APIs and automation is the key to having both an agile and cost-efficient infrastructure.

Toward Infrastructure as Code

Traditional business applications addressed well-understood use cases. Business and IT professionals could predict the workloads that would be run, and the volume of data involved when operating typical back-office applications.

Then, production environments were once fairly stable—but that's no longer the case. Modern applications exposed to customers and those that consume machine-generated data have more variable demands. A popular mobile app can generate spikes in demand for compute and storage services. An online flash sale at an e-commerce site can create short, high-volume bursts of processing load. Businesses operate in fast-moving environments, and their infrastructure has to keep up.

Software development has also changed. Instead of long development cycles driven by fixed waterfall design methodologies, modern software development employs agile practices, and they have implications for infrastructure management. Developers may need to start development and test environments at any time. As they roll out new features to production, an application’s need for compute and storage resources may change.

This kind of variation in workloads can lead to changes in the number and type of servers needed. Developers and system administrators could closely monitor workloads and add and remove servers as needed, but automated approaches are far more effective and reliable. These methods depend on infrastructure APIs.

Infrastructure as Code (IaC) is the managing and provisioning of infrastructure through code instead of using a manual process to configure devices or systems. In that sense, infrastructure is treated the same way as any other code.

APIs Are Interfaces Between Microservices

APIs are services that expose functionality for other applications. That functionality can be as simple as providing a list of servers in a cluster or as complex as enabling encryption on a storage device. APIs define how one service can interact with other services. It is essentially a software contract.

For example, APIs specify the particular data needed as input to a function as well as the structure of the function’s output. An API function for assigning a public IP address to a server might include the server identifier and the name of the network interface that should be assigned that IP address. The output would include a status code indicating whether the assignment was successful.

Note that there’s no need for users of the service to concern themselves with implementation details, such as how the API finds

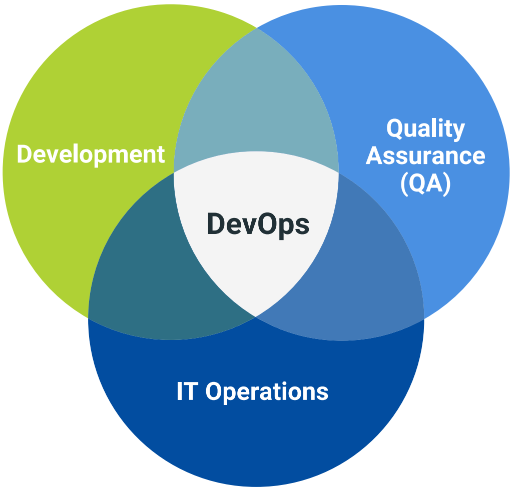


Figure 2: Use of APIs is essential to implement proper DevOps practices.

the right server or which configurations on the server need to be changed to assign the IP address. APIs hide implementation details, making it it easier for other applications to use them. Also, scripts and applications that use an infrastructure API can be written in any programming language, not necessarily the one used to implement the API.

DevOps, which is the combination of software development and systems operations, is a common way of managing code and infrastructure in agile environments (see **Figure 2**). One of the common practices in DevOps is to control how application code is developed, tested, and deployed. Developers use code repositories to manage application code. Just as application code is developed, tested, integrated and deployed, code that uses APIs to implement and modify infrastructure can also be controlled in the same way.

The operations part of DevOps includes tasks such as deploying a cluster of servers, modifying security configurations, and setting up performance monitoring alerts. All of these can be done using API such as the Nutanix API. Container management is a good example of what can be done with infrastructure as code.

A container is defined by a specification file that determines the base operating system and additional software installed and system configurations. DevOps engineers should be able to create, modify, and deploy containers programmatically, and the Nutanix platform APIs make this a simple operation. The infrastructure APIs allow for more than creating and monitoring containers and servers, though—they're also useful for automating storage management, including resources like storage pools, disks, and even encryption at rest.

Consider how this capability can be leveraged under dynamic workloads. For example, a typical flash sale on athletic clothing generates a 10-fold increase in traffic to the vendor's website. The vendor has a set date and time for the flash sale, and just prior to it, the DevOps staff executes a script to provision additional servers in the cluster running the e-commerce application. They may even deploy additional clusters, load balancers, and network-attached storage as a hot standby in case there is a problem with the primary cluster.

This kind of on-demand provisioning was not possible before virtualization. Without infrastructure APIs and orchestration tools, DevOps professionals would be left to manually deploy the needed resources, which is slower and more prone to error than the automated method enabled by APIs.

Infrastructure APIs are used for more than provisioning compute and storage resources. They're also used for ongoing maintenance and management operations. For example, APIs can be used to create monitoring alerts on servers and containers as they're deployed. These monitoring services can collect performance data, which is then streamed to a monitoring service that uses the data to generate dashboards that display performance data send notifications when intervention is required.

Let's assume servers are expected to run at about 65% CPU utilization. When the average CPU utilization of a cluster exceeds 80%, additional servers should be added to the cluster. Of course, a system administrator could manually add servers, but with the availability of infrastructure APIs, the problem of high CPU utilization can be remediated automatically. And similarly, when load drops and average CPU utilization falls below some threshold, APIs can be used again, this time to remove servers from the cluster.

Infrastructure APIs can also help improve security by consistently enforcing information security policies. For example, an organization may require that all data at rest be encrypted. APIs can be used to enable encryption at rest on storage devices when they are deployed as well as regularly check the status of encryption. They can also be used for related maintenance operations, such as verifying digital certificates are in place and not expired.

Infrastructure APIs allow developers and system administrators to create code that defines infrastructure. Just as with application code, infrastructure specifications can be tested, reviewed, and controlled in code repositories. This enables standardized management of infrastructure while also improving efficiency.

Essential Automation

The demands on IT infrastructure have changed dramatically from the days of back-office batch processing of well-understood workloads running on mainframe computers. Today, businesses are utilizing web, mobile, IoT, and machine learning technologies to deliver new kinds of products and services.

The infrastructure that supports these new workloads often takes advantage of virtualization and cloud technologies. This allows for more efficient use of resources because the infrastructure can be adjusted to meet the changing needs of workloads. Creating, modifying, and monitoring this kind of dynamic infrastructure requires automation. Infrastructure APIs are essential for that automation.

The Nutanix API is a comprehensive set of services that enable the kind of automated orchestration that agile businesses need to drive innovation and remain competitive while maintaining the most efficient operations possible.

Next Up

Speaking of automation, that's the focus of Chapter 2, which digs into that topic and its close relative, orchestration.

CHAPTER 2

Enterprise Cloud Automation and Orchestration

Automation is a key aspect of improving agility and productivity. IT teams are continuously looking for ways to automate and improve workflows to decrease the time spent on repetitive tasks, freeing up time for business-oriented innovation and projects.

One particularly hard area for automation is application development and delivery. The immense complexity and variation are a major challenge for IT teams, resulting in time wasted, re-work, and massive and complex scripts to work around limitations and issues.

ITOps and DevOps teams often work in silos, exacerbating these problems. Meetings, organizational overhead, and confusing handoffs between these teams greatly reduce the ability to deliver business value quickly. It's challenging for teams to keep up with business demands.

In frustration, teams often employ "shadow IT," including public cloud. This exposes the organization to risks of non-compliance, security breaches, high cost, and more.

In this chapter, we'll look at how enterprise cloud automation and orchestration help tackle these problems.

Managing Enterprise Cloud Automation and Orchestration

First, though, we need to define the term "enterprise cloud automation and orchestration." Let's split this up into its constituent parts:

Enterprise refers to features that large organizations need to manage their scale. This mostly comes down to features for compliance, identity management, and role-based access. It also includes support for multiple business units, as chances are the enterprise is big enough to have multiple development and operations teams, who may or may not be using multiple (public) clouds.

Automation and orchestration solve issues that are traditionally encountered when running IT at scale. If similar teams are doing similar work, there's opportunity to optimize a part of a process or workflow or automate repetitive work.

While there's a definitive reduction in time spent doing these tasks, another advantage is the reduction of human errors, resulting in unnecessary re-work. This directly translates into higher quality work, output, outcomes, and impact that these teams deliver.

With this basic understanding of *what* we're talking about, let's discuss *why* we need enterprise cloud automation and orchestration at all. What's the problem we're trying to solve?

Ease of Cloud

One of the factors contributing to the need for enterprise cloud automation and orchestration at all is that public cloud makes consumption of IT resources simple, especially in the following areas:

Cost and Billing. Cloud is famous for making it possible for anyone with a credit card to consume cloud services. The barrier to entry is as low as possible, and that's a deliberate choice for cloud providers.

Unfortunately, that easiest tier of cloud resources is also the most expensive.

Ease of Use. Cloud providers put significant effort into creating easy-to-use interfaces for their cloud. By creating great user experiences in both the graphical user interfaces as well as the programmatic interfaces (APIs, SDKs), users are drawn in—and locked in—to the cloud provider's services.

Service Portfolio. The breadth of services these providers offer is fantastic. Virtually every IT component is available as a cloud resource. That means that anything a system or software engineer needs is instantly available for consumption. Back in the day, data centers had to be built from scratch, which took a long time, was very expensive, and was even more error-prone.

These days, everything is available instantly and as a service—optimally configured, using best practices, and hardened for security and compliance. These services are turnkey. More importantly, they remove much of the toil of installing and configuring the underlying infrastructure, making the resource available instantly.

Self-Service. Hand in hand with the operational cost model, the self-service element lets anyone in the organization create and consume cloud resources. In the data center, IT admins had tight control over who could request, approve, create, and consume resources. In the cloud, that's no longer the case, increasing the likelihood of mistakes, and reducing the span of control of IT.

A Cloud for Each of Us

Those benefits have resulted in the massive shift of IT to the cloud, and often multiple clouds. This brings its own set of problems. For instance, each cloud vendor has their own strengths and weaknesses. Engineers have cloud vendor preferences. Purchasing departments have standing agreements with distributors that favor a cloud. In short, chances are that your organization uses more than one cloud vendor for its computing needs.

Chances are also high that IT doesn't have all (or even most) cloud subscriptions, accounts, and virtual private clouds (VPCs) in its sights and under control. This exposes the organization to risks including noncompliance, security breaches, high costs, and more.

With noncompliant cloud resources, mistakes can be made remarkably easy—exposing an object storage bucket with sensitive files to the Internet, for example, or failing to properly secure a database instance that leaves the entire database open.

And, not unimportantly, it fragments business processes across teams, departments, and business units, creating inefficiencies and waste in handovers, communication, team results and output, and more. Ownership is scattered, increasing the finger-pointing and decreasing release velocity. Expert-level knowledge is spread across the organization, making it hard to share lessons learned and leverage that expertise in other areas of the enterprise.

At the same time, engineers are pressured to develop and deliver applications more quickly. And with the applications within the enterprise growing in numbers and complexity, the multi-cloud management challenge can become a significant hurdle to the productivity gains promised by the cloud.

App Store for Enterprise IT

We now know that starting to consume cloud resources is simple—as easy as typing in a credit card number. But doing it in a way that's compliant and secure is far from simple.

And that's just for one cloud. Replicating those efforts for multiple clouds that may or may not be IT-managed ramps up the difficulties significantly.

The solution is to consider resources from across all private and public clouds as building blocks for enterprise IT consumption, then bringing them under management using an enterprise cloud automation and orchestration toolset.

Think of this as an Enterprise IT App Store, filled with components and services from the enterprise cloud, as well as public cloud vendors. This app store makes it easy to consume these services, while IT stays in control and compliant.

In a similar vein, the incredible breadth of services offered by public cloud providers removes almost all friction for software and system engineers to use IT resources. These can range from virtual machines (VMs) and containers to databases and services that run arbitrary code without caring about the underlying infrastructure.

Marketplaces offer a seamless and smooth self-service user experience for deploying new apps, regardless of whether these are building blocks and components that make up applications (like a web server or database), or the entire application itself (like an application consisting of a web server, database server and application server) for direct consumption. The marketplace presents the user with the right content based on their role.

Mobile App Stores

App stores made software distribution trusted, secure and took away manual work. For mobile phones, the Apple and Google app stores made downloading trusted apps a breeze. By tightly controlling distribution, both companies attached their names to third-party apps, adding trust and security by proxy.

Building Blocks

These building blocks represent huge business advantages, streamlining how IT is consumed and managed from a highly inert, high-friction, capital-intensive endeavor, to a pay-as-you-go, experiment-and-fail-quickly, low-risk environment.

This increase in agility lets teams move forward more quickly, and not be hindered by mistakes or the impact of changing direction for months and years. Instead, wrong decisions can be undone quickly and easily, and constant maneuvering in a better direction becomes the norm.

This reduces the pressure on system and software engineers, as they're not constantly burdened by technical debt and investments that are no longer fit for their purpose (but have to be used until the end of their economic lifespan).

Built by Pros

These compounding building blocks improve agility, as it's easier to swap out just a single component. This was much harder in the traditional data center. It reduces the reliance on those with expert-level knowledge, because consumption is standardized using repeatable blueprints.

The experts can focus on creating and maintaining the blueprints to be scalable, performant, secure, and compliant. The user is empowered to use them without human intervention, removing the factor that causes the most delay when requesting new resources. And, at the same time, consumption is governed by central IT compliance and security policies.

Building blocks can be sourced from any number of vendors, including the public clouds, specialized service providers for highly business-specific services, as well as your own enterprise cloud.

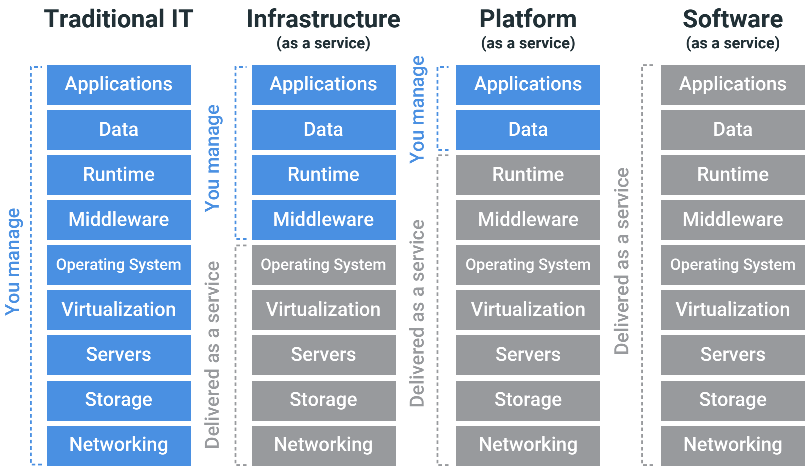


Figure 3: As applications move from the traditional IT model (left) to Software as a Service (right), the IT department or end user is responsible for managing less and less of the infrastructure.

This instantly leverages expertise from across the organization and beyond, and makes these high-quality services available for consumption.

These blocks include turnkey services for various audiences, including ITOps, developers, and business users.

Much like the layer cake of traditional IT, with Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS), the different audiences consume services on different layers in the cake, to offer the bedrock for higher-level services. See **Figure 3**.

For instance, ITOps might consume IaaS services to build and run commodity, off-the-shelf applications. Developers might use a PaaS (or these days, Kubernetes or Containers-as-a-Service) to build and run their in-house, custom-built applications. And business users are able to consume the services built by both IT and development as SaaS, with full self-service characteristics to, for instance, build new applications for new customers without human intervention from either ITOps or developers. The point is each higher-level construct is built on top of the lower layers.

Automate Composition

No matter who deploys what, they all need flexibility to pick and choose exactly those building blocks needed to accomplish their goals.

There's a tight balance to keep between a figurative big box of mixed Lego pieces from various sets, versus only the assembled and finished creations.

For optimal use, blueprints, the finished product, are composed in a specific way:

1. The blueprint itself adds immediate business value to its user
2. It consists of re-usable components
3. It consists of everything that makes up an app

The benefit of having these three characteristics is that the blueprint is complete enough to be used immediately without requiring extensive customizations, yet configurable enough to be fit for purpose if slight tweaks are needed. If they choose, users can look "inside" the blueprint, and re-use the components to create another blueprint.

This requires that components themselves are blueprints, each consisting of "lower level" components in the layer cake, until the lowest level of components is reached. This way, each layer of the technology stack is made up of re-usable components, such as VMs, containers, application binaries, and configuration files, as well as deployment automation and workflow operations that install and configure the application.

Individual component blueprints can be maintained within the organization, or come from a public marketplace, where blueprints are maintained by third parties. That way, admins within your organization don't need to create and maintain blueprints for third-party software, so that infrastructure teams can eliminate hours and days of toil, repetitive work devoid of real business value.

Also, blueprints span components from across the organization, capturing expertise from different IT disciplines. Components that provide a technical service, like a database or web server, can be maintained by the expert group while being used throughout the organization. This helps in capturing the latest organizational knowledge, codifying these learnings into blueprints.

With this layered-block approach, applications can be modeled, automated, and orchestrated to streamline complex provisioning and lifecycle tasks. The building blocks, or Legos, fit together to form a bigger whole, while the individual pieces can be re-used for different creations.

With the right visual modeling tools—like Nutanix Calm, for example—applications can be composed in a logical and visual way. The Nutanix Calm blueprints capture all components of an application, including infrastructure, artifacts (binaries and other files), task sequences, and configurations. See **Figure 4**.

These blueprints allow organizations to break free from cloud vendor lock-in, letting them simplify the setup and management of custom enterprise applications by deploying blueprints across public and enterprise clouds and hypervisor technologies. Policy-based deployment governance makes sure utilization and sizing are optimized.

Governance

In an enterprise, governance features are indispensable. While empowering different groups within the organization to provision and manage their own applications on public and enterprise cloud increases their self-service capabilities, speed of execution, and independence, it's also prone to loss of control, leading to unacceptable risks in cost, security posture, and unauthorized access.

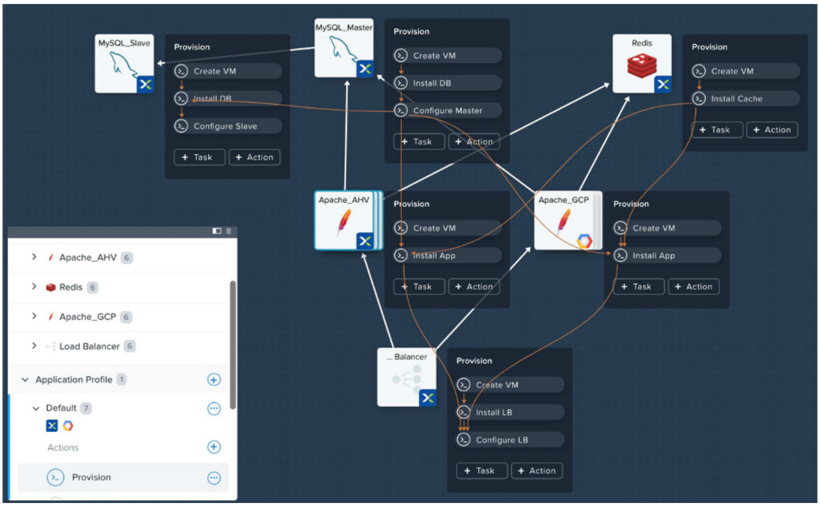


Figure 4: Nutanix Calm defines applications via simple blueprints that administrators can easily create and instantly deploy.

So, while providing access to resources and making the application lifecycle self-service are good goals to strive for, governance while doing it is crucial.

By logging all activity and changes that teams make when using the cloud automation and orchestration portal, security teams have end-to-end traceability, making the current and past state of any applications deployed through the portal verifiable.

Role-based access control (RBAC) makes sure only those that need it have access to manage the lifecycle of apps. This way, different groups and teams (such as developers and application owners across different lines of business) can deploy and access their applications, without being able to access and manage others.

All Part of the Pipeline

For some groups within the organization, à la carte consumption of applications via a GUI isn't the best way. These groups, often developers, need programmatic access to objects like blueprints or running applications. Instead of navigating the GUI, they call an API in a workflow to create, scale, or destroy applications from an automated pipeline.

Any App, Any Cloud

The reality of managing applications across their lifecycle is that ITOps is no longer in the driver's seat for most decisions. Application owners on the business side of the organization are becoming more IT-savvy, and decisions are increasingly business-driven.

That means that applications no longer exclusively run on infrastructure under the control of ITOps, but can run wherever business requirements, like time-to-value, cost, performance, and scalability are met best.

This brings IT teams to a choice: willfully ignore or fully embrace these changes. Embracing this new way of doing IT can be challenging for ITOps and infrastructure teams. Adopting cloud automation and orchestration can help to smooth out this transition.

It also lays the foundation for a proactive, conscious multi-cloud strategy. This distinguishes it from how many organizations end up with multi-cloud setups, in which it "just sort of happens." Multi-cloud often isn't a thought-out plan or part of any architecture, design, or budget discussion. It's often application owners and developers that need more than their own IT can provide and go shopping for themselves, resulting in many different cloud environments being used across the organization.

Cloud automation and orchestration portals are multi-cloud. Think of these portals as a cloud interface for multi-cloud, abstracting away the capabilities of individual public or private clouds into a single global enterprise cloud, servicing the needs of users across the enterprise's business units, as well as different roles like developers, application owners, and operations teams.

The single control plane allows blueprints to be deployed across different cloud flavors without change. The destination infrastructure is simply one of many variables in the blueprint.

Blueprints can also leverage multiple clouds concurrently to create a multi-cloud application architecture, taking advantage of each cloud's strong points or low cost. This removes dependence on technology, removes vendor lock-in, and truly makes applications portable.

Hybrid Cloud

While we're on the subject: Enterprise IT often isn't greenfield, nor are applications neatly isolated on one infrastructure. Application architectures often grow dynamically—and sometimes uncontrollably—over time.

The result is a hybrid architecture, which has become a reality for most applications in the enterprise. And over time, entropy ensures that virtually any combination of on-premises, service provider, SaaS, and public cloud will occur.

This reality means enterprise cloud automation and orchestration should be able to not just deal with hybrid applications, but actively embrace hybrid capabilities, as businesses seek to optimize application cost, performance, and resilience.

Flexibility in extending support for new deployment patterns and destinations in the future should not compromise the self-service capabilities of how teams deploy and manage the lifecycle of applications.

App-Centric Cloud

App management is broken up between different teams because of the knowledge split between teams. Each technology stack, be it a public cloud vendor or on-premises data center, requires a unique skillset, and teams are often split up into managing specific parts of the infrastructure, like the network, storage, backup, and software development. Application teams don't have all of these skills, leading to bottlenecks, dependencies, and teams not fully understanding their own application.

Harmonizing multi- and hybrid-cloud capabilities, while also automating workflows and hiding implementation details and other complexities, are undeniable considerations in the journey to an app-centric cloud approach, so that teams can start to understand their applications more completely.

Multi-Anything (or X as a Service)

Self-service should work across commodity, off-the-shelf applications and custom-built applications built by in-house developers equally. The complexity of the enterprise dictates the breadth of support needed to automate and orchestrate applications, not vice versa.

Only full support of all elements that make up an application, including the technical components like VMs, databases, existing services, and application configurations, will be meaningful to business users. Anything less will force them to jump through hoops instead of accomplishing the goal—self-service automation of their application lifecycle needs.

This includes the ability to cater to whatever users' requirements are. *X-as-a-Service* capabilities are important to automate and orchestrate those small, but often vital details in the application lifecycle that no one thought about up front, or are highly specific to a certain group, user, or application. Extensibility, to automate anything into a service, is an often-overlooked component of functionality.

Removing the operational complexity of multi-cloud, while still giving users real-time visibility of cloud costs and compliance states, standardizes and democratizes IT, so users can manage the lifecycle of their applications in a self-service manner.

App Store for Any Team

The consistent experience of providing lifecycle management opens up new use cases to users in the enterprise.

With orchestration tools, deployment and management of the applications' lifecycles is extended further into the enterprise; allowing, for instance, users at a satellite, remote, or branch office location to deploy and manage applications, or enabling edge and IoT use cases to become self-service and managed directly by the user.

Making Applications Self-Service

They key for making application lifecycle management self-service is the marketplace, where the self-service aspect and the orchestration engine come together. This is the interface between where experts publish their components or full blueprints and where users go to deploy and consume their applications.

These blueprints are tested, pre-integrated and validated before publication, making sure the user experience is truly "one-click" and as simple as possible, while still being compliant, secure, and resilient.

This approach hides most, if not all, of the operational complexity of deploying new applications, so that users are empowered to do this self-service, removing dependencies on other teams and bottlenecks like ticket requests.

Users will only see applications published to them based on their role and access rights, and can request these blueprints as needed. Optionally, approval workflows can be integrated, but for many requests, it's business as usual, giving teams the freedom to operate quickly and efficiently on their own.

Manage Applications Across Their Lifecycle

Managing lifecycles entails more than the request and deployment phases. For many application owners, day-to-day operations, scaling, and decommissioning are part of their responsibilities, too.

The orchestration engine is there to help application owners manage the entire lifecycle, from request to approval, provisioning, management, scaling, reclamation, and decommission.

But enterprise application landscapes are never greenfield—there's always a core part that's been there for ages, and it often gets overlooked. Bringing these existing resources and services into management is an important factor to consider when empowering users to do self-service management of the application lifecycle.

For ITOps

ITOps teams spend a fair share of their work on daily operations, many of which are recurring and reactive in nature, like responding to application-centric requests. An enterprise cloud automation and orchestration tool helps to empower self-service without losing visibility or control.

Integration with Nutanix Calm

Enterprise cloud automation and orchestration tools, like Nutanix Calm, offer the ability to integrate with existing resources, such as VMs, and bring them under management. These existing resources are called as a dependency or existing service to maintain the same level of automation.

By automating the work, as well as making it available directly for self-service consumption by the different user groups in the organization, IT teams are freed up from this repetitive daily grind.

ITOps teams have up-to-date information about ownership of each deployed application and real-time insights into the lifecycle status of each.

For Dev Teams

Development is characterized by a high degree of uncertainty. Developers build something new every day. This means that it's hard to predict what they need beforehand, and flexibility in infrastructure, middleware, and other IT resources helps them be productive and successful.

Traditionally, though, this unpredictability resulted in massive restrictions from ITOps. They could not keep up with the variety of requests, nor the volume. These developers are often blocked by these requests, and frustrated by ticketing systems or queues that elongate the blockage.

Self-service capabilities are obviously beneficial to developers—this is what drew them into using public cloud in the first place. Being able to quickly deploy a new database, container, or PaaS application to run their code, without human intervention, massively sped up their development velocity.

The downside was loss of control for IT. Spiraling costs, increased risk of data breaches, and potential compliance issues come with shadow IT, and prudent IT departments quickly stopped this uncontrolled cloud usage.

Cloud automation and orchestration tools help development teams create, scale, and destroy test/dev environments without creating tickets or waiting for the ITOps team. They're able to execute these tasks without human intervention. Meanwhile, ITOps is in full control over compliance, cost, and ownership of all resources deployed in the enterprise cloud, as well as public cloud instances.

For Business Users (Application Teams)

Similarly, business users are catered to via the marketplace. They're able to deploy and manage their own applications asynchronously—without any dependence on ITOps teams, ticketing systems, or other bottlenecks.

Get in the Driver's Seat

The cloud, with all its benefits, has had the unintended effect of making infrastructure resource consumption *too* easy, leading to shadow IT and issues with regulatory compliance and security.

However, by leveraging enterprise cloud automation and orchestration, IT admins can enable an enterprise cloud that spans public cloud, on-premises data centers, and SaaS so users can deploy and manage the application lifecycle in a self-service way.

The marketplace serves as single-pane-of-glass across all clouds, enabling hybrid and multi-cloud functionality and removing cloud vendor lock-in. It also increases application portability without losing control over compliance, cost, and other risks. This allows enterprises to work on a proactive, intentional multi-cloud strategy, harnessing the strengths of each cloud without the uncontrolled sprawl that leads to the "it just happened" multi-cloud.

By using a marketplace, ITOps and other subject-matter experts can create and publish resource, service, component, and entire application blueprints for self-service consumption by users, making resource and application delivery asynchronous and no longer dependent on ITOps. Blueprints are tested, secured, and verified to be truly turnkey for an optimal self-service and seamless user experience, so that developers, application owners, and business teams can efficiently consume IT resources.

With enterprise cloud automation and orchestration, ITOps puts the business in the driver's seat, and democratizes IT. It makes firing up any IT resources in the enterprise cloud as easy as clicking a button, while simultaneously remaining compliant and keeping the right people in control.

Next Up

All that automation, however, can have a drawback—if you're not *very* careful, you can end up spending far more than you intended on your cloud operations. We'll tackle the business of keeping costs under control next, and the importance of monitoring toward achieving that end.

CHAPTER 3

Monitoring and Controlling Cloud Spend

The world of IT has been completely reshaped by the introduction and widespread adoption of cloud computing technologies, both public and private. For the record, a public cloud is one to which a company or organization subscribes that is owned and operated by a third party. That means all public cloud customers share a common underlying infrastructure at the edge and throughout the public operator's network.

In contrast, a private cloud is one that the company or organization owns, controls, and operates itself. It controls its own edge and infrastructure elements completely. In both cases, however, this means access to a variety of highly virtualized services that can provide compute, storage, and networking capabilities.

An IT infrastructure that combines private and public cloud elements is called a hybrid cloud. In a hybrid cloud, organizations can exchange data and resources between private and public clouds.

An IT infrastructure that combines one or more each of private and public cloud elements is called a multi-cloud. That said, there is no guarantee that data or resource can or will be exchanged across two or more clouds, public or private. As you'll see, it's no exaggeration to say that when it comes to IT, "the cloud changes everything."

Enterprises Lead the Way in Cloud Adoption and Use

According to the Nutanix Enterprise Cloud Index (2019 Edition),[1] 85% of enterprises continue to rank hybrid cloud as the "ideal" IT operating model, while 73% are migrating applications away from the cloud back to on-premises infrastructure. Overall, 60% reported that security is the biggest factor impacting future enterprise cloud strategies.

Enterprises plan to shift investment aggressively to hybrid cloud architectures, though short-term plans for cloud deployment slowed in 2019. Interestingly, almost three-quarters of 2019 enterprises surveyed reported moving some applications from the public cloud back on premises. Thus, use of traditional, non-cloud-enabled datacenters increased slightly, rather than dropping by 20% as forecast in 2018. Nevertheless, enterprises surveyed report steady, serious hybrid deployment plans through 2024. Persistent preference for hybrid cloud stems from a number of factors, including:

- **Security concerns:** Hybrid cloud is believed to be the most secure, even compared to private clouds and on-prem data centers
- **Flexibility:** Enterprises can choose an optimum IT infrastructure for individual business applications dynamically
- **Expanding cloud options:** IT departments no longer face a cloud/no-cloud choice; rather, they decide which cloud(s) to use on a per-application basis.

These trends and decisions make it clear: private and public clouds are here and are already ubiquitous. And while enterprise is in the lead in this technology area, SMB players don't trail too far behind, either.

[1] https://www.nutanix.com/enterprise-cloud-index

Dominant Cloud Providers

According to a November 2019 technavio blog post,[2] the leading public cloud providers are as follows:

- Amazon Web Services (AWS)
- Microsoft Azure
- Alibaba Ali Cloud
- Google Cloud Platform (GCP)
- IBM

A Typical Enterprise Scenario

Multi-cloud adoption is a common strategy that enterprises employ to avoid vendor-lock in. But hybrid clouds play an even more important role. The number of enterprises planning to adopt more than one public cloud increased just slightly in 2018 and 2019, up from 12% to 18% over that two-year period. During the same time period, 41% planned to adopt hybrid cloud architectures.

This strongly implies that some organizations will seek to avoid lock-in by making data and applications easily portable between some public cloud and their private cloud. This appears to be more attractive and secure than bringing multiple public clouds into their IT infrastructures. Undoubtedly, this is because private clouds provide more control and better opportunities for cost planning and governance (often automated) than do public clouds.

Understanding Cloud Spend Trends

In this chapter, we'll examine what's occurring in the cloud services marketplace, with an emphasis on what is generally called

[2] https://blog.technavio.com/blog/top-10-cloud-computing-service-providers-2017

"cloud spend." This phrase covers how much organizations spend on cloud-related services across the board, including subscription fees, usage fees, and anything else likely to show up on a billing statement from a cloud services provider.

Average Cloud Spend Elements

Cloud services break into multiple service categories, including the following:

- **IaaS** provides computer infrastructure to support IT operations, including storage, servers and network components. In the cloud, all such elements are virtualized, made available to customers by reservation or on-demand. Normally, additional services such as detailed billing, monitoring, logging, security, load balancing, and clustering options are included, with additional charges for storage resiliency elements, such as backup, replication, and recovery (some providers even go so far as to offer disaster recovery and business continuity options). Increasingly, policy-driven controls invoke automation and orchestration for infrastructure tasks and services.
- **PaaS** provides hardware and software tools, typically those used for software development, modeling, simulation, analytics, and so forth, via the Internet. The provider hosts the necessary hardware and software on its own infrastructure, then makes the whole shebang available on demand or by reservation to customers.
- **SaaS** provides access to hosted applications via the Internet. That is, the provider gives its customers Internet access to an application that the provider has created especially for SaaS distribution. Under a single subscription (usually per organization), multiple users share the same data and code base, which may be stored both locally and in the cloud. Customers may also integrate SaaS applications with other programs using APIs provided to support customized software tools that integrate SaaS application services and capabilities.

Of these service categories, IaaS services remain the dominant component: almost 70% of the total cloud spend across the board goes for compute, storage, and networking services that fall under the IaaS umbrella. Across all market segments, the majority of PaaS outlays go for databases and data analytics (big data). Of the various market segments, enterprises spend more on cloud-based networking, analytics and security services than do other segments (government, academia, SMBs, and so forth).

At the other end of the organizational-size spectrum, SMBs are spending significant portions of their total expenditures on emerging services, such as artificial intelligence (AI) and machine learning (ML), the Internet of Things (IoT), and blockchain technologies. They appear to be seeking competitive advantage or market differentiation through use of leading/bleeding edge technologies. Not coincidentally, Microsoft has made significant investments in all of these areas. In fact, Microsoft makes support for them available to customers in a variety of innovative production environments.

AWS Outlay Is About Double That for Azure

In looking at the overall patterns, Gartner estimates that the total spend at the end of 2018 for AWS was approximately double that for Azure. At the same time, it appears that average total cloud spend per organization at least doubled in 2018, which shows enormous growth in cloud adoption across all market segments.

Nevertheless, Azure spend is growing at least twice as fast as AWS spend across all segments, so it appears that these two market leaders are headed for parity sometime in the next year or two.

Other Typical Outlays Include Major, Minor, and Specialty Players

Other inhabitants of the various leading cloud provider rankings also garner a share of total cloud spend. As with Azure, AWS, and GCP, much of this outlay goes for similar IaaS offerings from other players. In addition to further outlays for PaaS offerings—primarily for analytics, IoT, AI/ML, and databases—the other chief target for cloud spend is for providers like SAP, whose SaaS offerings are widely used throughout business, government, and industry.

Cloud Consumption Patterns

The distribution of cloud spend by service category is worth contemplating in graphical form. It shows how spending breaks across categories such as compute, networking, storage, database, security, analytics, emerging services (AI/ML, IoT and blockchain), and various other categories, as shown in **Figure 5**.

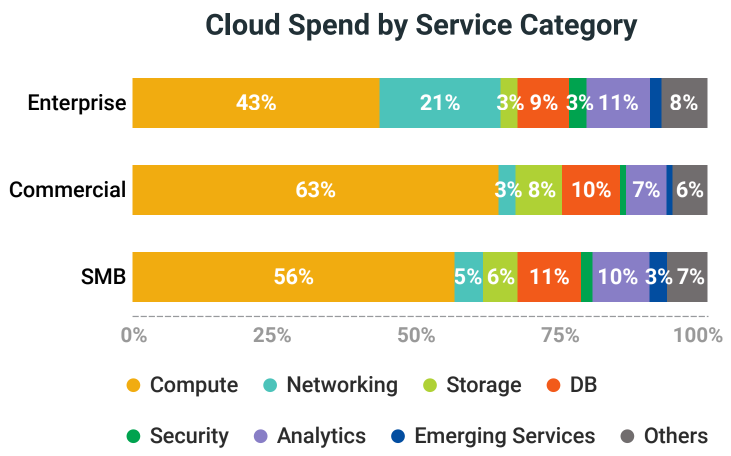


Figure 5: Compute is the dominant single item for spending across all market segments. Things get much more interesting moving to the other end of each graph. (Source: 2019 Cloud Usage Report)

Compute Services

Across the board, IaaS spending dominates total cloud outlays, with the compute service category topping the compute-networking-storage triad. As far as the next element of that triad goes, enterprises significantly outspend commercial and SMB market segments on networking (21% enterprise vs. 3% commercial vs. 5% SMB). This reflects relative differences in scale, scope, bandwidth consumption and capabilities of the virtual, cloud-based networks involved. Reflecting greater capital investment and higher use of private networks, enterprises spend least on storage (3% vs. 8% for commercial and 6% for SMB).

AWS S3 and Azure Storage are the most widely consumed storage services, though enterprises seem more likely to spend on cloud-based backup and recovery to protect their cloud than other segments (commercial and SMB).

Database Services

Databases seem almost equally important across the business spectrum, and almost equally used in all market segments, at 9% for enterprise, 10% for commercial, and 11% for SMB. Thus, it appears that databases are important for business in general, and they consume an appreciable part of IT and cloud budgets of all kinds.

In this mix, AWS RDS and Azure Data Factory lead the pack for cloud database services, but DynamoDB and Amazon Redshift have more presence in the commercial space than in other segments (enterprise and SMB). Azure Cosmos DB and Azure Redis Cache show significant uptake in enterprise, whereas PostgreSQL and SQL DB lead in the commercial and SMB segments, respectively.

Analytics and Emerging Services

With regard to analytics and emerging services, AI/ML and IoT are growing the fastest, in terms of cloud spend. SMBs show the greatest predilection for these kinds of services, often because they’re technology startups developing or investing heavily in these technologies. Blockchain services remain relatively new in the cloud (though Microsoft has some interesting offerings) and have yet to gain real traction in the form of significant cloud spend.

Where and How Organizations See and Use the Cloud

Certain enterprises—and other select organizations—are already considerably proficient in and expert users of cloud technologies. In particular, such enterprises employ a greater variety and number of cloud services than do other market segments. It all goes to fill their growing need for access to (and services from) the cloud. To better understand how organizations see and use the cloud, we can classify them according to a basic metric of cloud maturity, to position them against the total population and other companies and organizations in the marketplace.

Classifying Cloud Maturity Levels

The survey of Nutanix Xi Beam customers showed that, when it comes to cloud adoption and use, companies and organizations fall into three general categories:

- **Beginner**: Number of cloud services used is under 16, with primary focus on IaaS and some managed database services.
- **Proficient**: Number of cloud services used falls between 16 and 30, within which mix advanced cloud services are leveraged, with heavy use of IaaS, plus a mix of PaaS services (managed databases, some analytics, and more).
- **Expert**: Number of cloud services used exceeds 30; such organizations possess highly advanced skills so they can leverage a broad set of cloud-based services, such as serverless computing, real-time and batch analytics, IoT and AI/ML platforms and offerings.

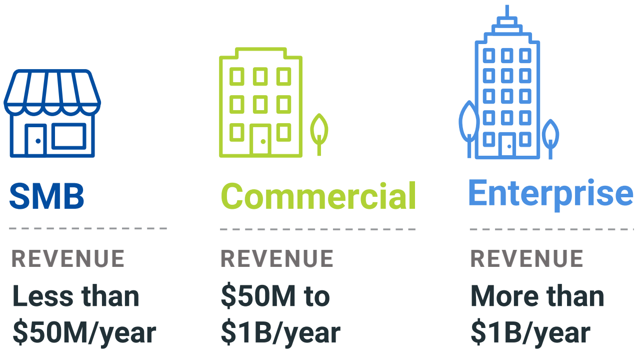


Figure 6: The segment breakdown is by annual revenue with SMB <$50M, Commercial >$50M but <$1B, and Enterprise $1B+.

Business/Organization Segmentation

With the survey of Nutanix Beam customers, the market segment definitions shown in **Figure 6** emerged.

Using this segmentation, and the preceding definitions for maturity levels, the distribution for each segment by maturity level looks something like what's shown in **Figure 7**.

This distribution shows that the enterprise segment, which has been investing and developing expertise in cloud technologies the longest, is skewed toward to the expert end of the "beginner—proficient—expert" classification scheme explained above.

Surprisingly, though, both down-market segments (commercial and SMB) are farther along than one might expect, and most heavily weighted in the Proficient area of the classification (half for commercial, and almost half for SMB). This completely validates the notion that businesses and organizations of all sizes and scales are already invested in the cloud, some quite heavily indeed.

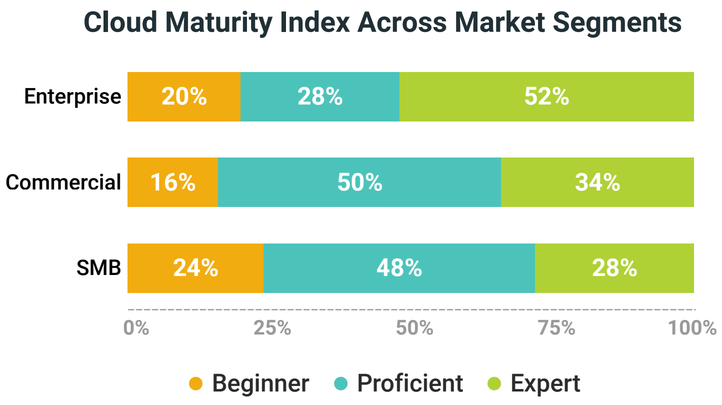


Figure 7: The distribution for each market segment by cloud maturity level.

Other interesting general observations about cloud maturity that appear in the BEAM report include:

- AWS users are more mature in their cloud usage (thanks in part to Amazon's broad and comprehensive collection of cloud services packages and offerings), though Azure is catching up quickly. (Microsoft is investing heavily in emerging technology areas such as blockchain, AI/ML, IoT, and big data/analytics, and customers are increasingly partaking of related Azure offerings.)

- SMBs find it easiest to invest in public cloud platforms and technologies (particularly because of the limited capital investments required, and the ability to finance cloud use from cash flow). That said, their limited staff and size makes it challenging to acquire and develop deep cloud expertise in-house. Thus, they often turn to vendor- or industry-focused consultants for help.

- The technology, media, and telecom (TMT) vertical has high cloud maturity, thanks to its involvement in and extensive use of cloud technologies. Healthcare and education have come more recently to the cloud. Thus, they're more heavily weighted in the Beginner and Proficient classifications.

Understanding Cloud Maturity Indices

SMB

SMB is most heavily weighted in the middle of the cloud maturity index classification scheme—that is, at the Proficient level. Surprisingly, the Expert level is 4% higher than the Beginner level (28% vs. 24%). This speaks to the profound attraction of the public cloud, even in organizations less likely or able to invest as heavily in private cloud infrastructures (owing to more limited budgets and capital investment power). This also explains their challenges in nurturing and developing deep cloud expertise in-house: Small or limited IT departments usually house Jacks (or Jills) of all IT trades, rather than accomplished but narrowly-focused experts or specialists.

Commercial

Like SMB, the commercial segment is also (and even more) heavily weighted in the middle Proficient level of maturity. But their Expert level is more than double their Beginner level (34% vs. 16%, or 2.125:1 Expert:Beginner). This speaks to bigger budgets with which to invest in cloud platforms and technologies, and more staff to learn and become expert in such things. And, of course, from bigger budgets also come bigger cloud spends as well.

Enterprise

When it comes to all things cloud, the enterprise segment still enjoys a commanding lead. It's the only segment for which the Expert level metric (52%) represents the majority. Thus, it also exceeds the combined levels for Beginner (20%) and Proficient (28%), as well.

With more time on the scene, bigger budgets to bring to bear, and more staff to dig into and really understand how to maximize return on their cloud investments, it's no surprise at all that enterprise companies and organizations continue to make the most and best use of cloud platforms, services, and technologies.

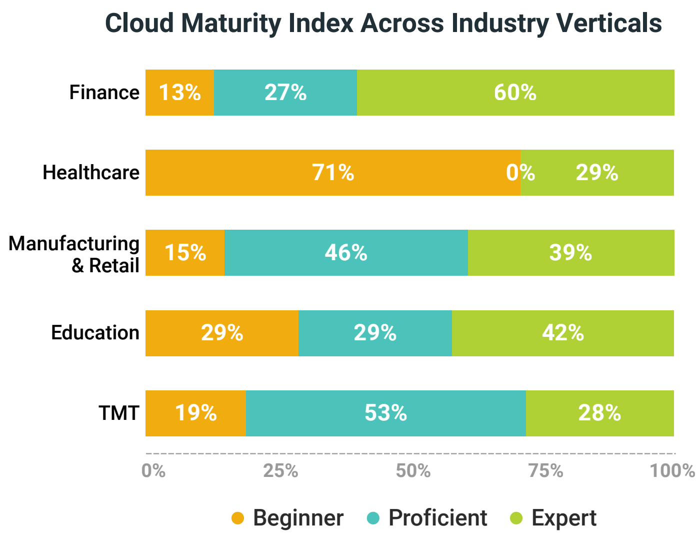


Figure 8: Right away, it's easy to see sometimes radically different maturity distributions by industry vertical (TMT vs. Healthcare, for example).

Verticals

The maturity distribution also falls out differently, depending on a company's or organization's industry vertical alignment. The graph in **Figure 8** contrasts the ratios of Beginner/Proficient/Expert for Finance, Healthcare, Manufacturing & Retail, Education, and TMT).

Finance clearly shows itself to have invested longest and most heavily in the cloud, as the only vertical where Expert exceeds the combined totals for Beginner and Proficient. But with ample money to spend, and the understanding that expertise is a considerable source of competitive advantage, this is exactly what one should expect to see.

Healthcare's late entry into cloud technologies (in the wake of HIPAA and EMR requirements) shows clearly in the absence of a "middle" (no Proficient) in its distribution, and in the preponderance of Beginner (71%) vs. Expert (29%).

Manufacturing, retail, and TMT all show their maturity with the kinds of distributions we've already seen for the SMB and commercial market sectors: weighted in the middle, with a heavier Expert level and a lighter Beginner level to match. Education's immaturity is evinced in matching Beginner and Proficient levels (both at 29%) that combine to edge out the Expert level (42% vs. a combined 58%).

CMI: Cloud Maturity Index

The Cloud Maturity Index that these various distributions represent shows not only where each one is in terms of its time involved with cloud technologies, but also its ability to muster the staff and funding necessary to achieve expertise. This makes it relatively easy to see where the biggest growth opportunities lie—namely in those segments or verticals where the Expert level shows room to grow, and especially (as with Healthcare) where the Beginner level shows room (in Healthcare's case, significant room) to shrink.

Optimizing Cloud Spend

Given current and projected levels of cloud activity, there's no denying that big money is already in play, with bigger and bigger outlays

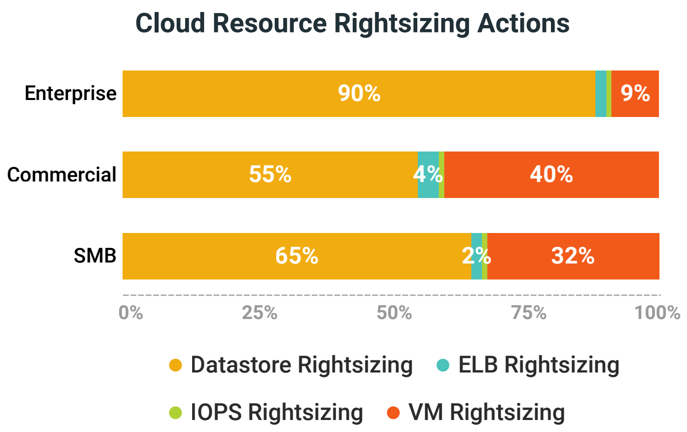


Figure 9: Rightsizing means comparing actual workloads to resource allocations for the VMs or infrastructures that run them, and sizing those allocations accordingly. Too much, and you're wasting money; not enough, and your users' productivity suffers.

still to follow. In a variation on the theme of "It's not what you've got, but how you use it," optimizing cloud spend is best understood as getting the most in value, return, and opportunity for dollars that go from companies' and organizations' bank accounts into the hands of cloud providers. To that end, here are some strategies that can help players of all sizes in the cloud game to optimize their cloud spend. Remember: every dollar saved on cloud spend is a dollar that can be spent (or paid out) to cover more innovation, more staff, stock buybacks, or dividends.

Figure 9 and **Figure 10** show snapshots from the 2019 Xi Beam report that document where various market segments found their savings from cloud cost optimization. They are real eye-openers: It's not only worth studying them in detail, it's also worth learning how to detect the various means to obtain your own cost savings in the cloud.

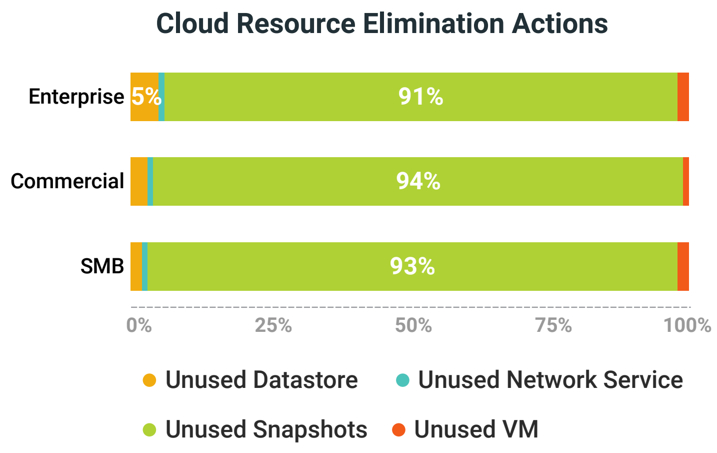


Figure 10: Eliminating unused resources stops unnecessary waste in its tracks. Why pay for cloud resources you're not using? Basic monitoring should provide this information quickly and easily.

Reduce Overprovisioning (aka "Rightsizing") for Cloud Resources

When dealing with cloud spend, cost monitoring is only one side of the equation. Indeed, it's essential to monitor where the money is going. It's not only important for controlling resource usage, it's also an important clue about where to look for savings (the more you spend, the greater the opportunity for savings).

The other side of the equation depends on knowing the workloads your organization runs in the cloud, then comparing the resources that get allocated by the provider to run such workloads against resources that those workloads actually consume while running.

The most common resource rightsizing actions including rightsizing of datastores (to give workloads sufficient storage capacity without too much unused slack space); VMs (to make sure their compute, network, and storage allocations deliver reasonable performance without laying on to many CPUs, too much capacity, or bandwidth); elastic load balancing (ELB) capability, which means a reasonable range in which demand can shrink and grow without too much headroom; and sufficient but not excessive IOPS (input/output operations per second, a metric commonly used for database access and other high-volume data processing operations).

Get Rid of Underused and Unused Resources

As **Figure 10** shows, there are different kinds of unused (or severely underused) resources that can provide a potential source of savings on cloud spend. The trick is to find idle or unused instances, and then to get rid of them. Unused resources generally fall into one of these four categories (listed in order of occurrence, shown by a percentage range in parentheses following the category name:

- **Unused Snapshot** (91% to 94%): A snapshot is a point-in-time VM image that represents a checkpoint, a potential backup or rollback, and a means of VM start-up, repair, or restore operations. A great, great many go unused—as the graphic clearly shows—so this represents an easy and massive savings opportunity for organizations of all sizes.

- **Unused Datastore** (2% to 5%): A datastore is a virtual storage facility usually associated with a VM, cluster, application, or service in the cloud. It's where the data lives, but if there's no data there, there's no point in paying for unused storage. A relatively small, but still worthwhile, opportunity for savings presents from this category, too.
- **Unused VM** (<1% to 2%): A VM is an instance of a runtime environment for some application, service, or capability that runs in the cloud. You must pay for each one you use as long as it's running. If nobody is using it, why bother? Another small but still worthwhile opportunity for savings here.
- **Unused Network Service** (<1% to 2%): Network services provide the connections that let virtualized platforms, services, and applications interact with users (to obtain requests and provide corresponding responses). Capacity costs while it's active, so if nobody's using it, you might as well turn it off and pocket the savings. The smallest of the savings opportunities, optimizing network services nevertheless remains worthwhile.

Overall, the Beam report indicates that across all customers (all segments, all verticals), savings of 10% to 15% on cloud spend are possible by employing the two preceding strategies: rightsizing or eliminating unused cloud resources. In particular, eliminating unused resources can be attractive, because it easily and naturally combines monitoring (to detect when instances go idle) and automation (to shut those instances down—and cause associated charges to cease).

Another category not present in the report is also worth mentioning, and investigating for those who subscribe to and consume cloud services. This is “Reserved Instance Utilization,” which refers to VMs and environments set up in advance at lower costs to reflect their likely and constant use.

Though such reserved instances are cheaper than their on-demand counterparts, letting them sit idle still represents overspending on unused cloud resources. Organizations should review this category carefully, as pruning here can add as much as 20% more to those savings afforded by the other strategies already mentioned.

Next Up

Another big challenge for cloud environments is security, and its close cousin, compliance—after all, if you don’t have the first squared away, the second could be extremely uncomfortable when various auditors or regulators come a’-calling. That’s what Chapter 4 deals with.

CHAPTER 4

Automated Cloud Security Audits and Compliance Validation

Security is top of mind for many IT practitioners. No surprises there, with the increased complexity of using multiple clouds, mixing private and public cloud, using both virtual machines and containers, and making full use of what Infrastructure as a Service, Platform as a Service, and Software as a Service are offering you.

Enterprises are relying on technology more and more to provide value to their customers. And with every new system, application, or service, security risks increase. It only takes one security vulnerability in one system to compromise confidentiality or the integrity of data. It's often the more overlooked aspects of the data center, like non-production systems, that are the most vulnerable, providing hackers an entry point into the entire IT estate.

No wonder that the role of security in enterprises is changing rapidly. No longer is it just about the nitty-gritty details of access control lists, role-based access control (RBAC), and encryption. Nor is it solely about risk management and mitigation.

Instead, security has become a strategic part of doing business. Boardroom discussions around improving the security posture or, more precisely, around how to stay out of the news, are now a routine exercise for many CEOs and security professionals.

Introduction to Multi-Cloud

The cloud is an undeniable force in enterprise IT. The flexibility, ease-of-use, and low barriers to entry are some of the reasons cloud is so popular today.

Anyone with a credit card can start consuming public cloud resources. And with different cloud vendors having different strengths and weaknesses, it's no wonder that many organizations are using cloud services from more than one vendor.

But, unfortunately, doing cloud "*right*" is harder than it looks. Those low barriers to entry and cloud's ease of use often result in impromptu setups with insecure, non-compliant use of multiple cloud vendors. These multi-cloud environments sometimes aren't planned in advance: application owners and developers need more than their own IT department can provide and go shopping for services that suit them specifically, resulting in many different clouds being used across the organization.

This scattershot cloud growth is dangerous from a security perspective, as anyone can start using cloud services without talking to security and purchasing specialists. This approach not only risks being more expensive (due to the lack of cost control), it also risks a weakened security posture due to misconfiguration and cutting corners.

The combination of low barrier to entry coupled with the vast array of services, each with their own best practices, is a serious problem for an organization's security posture in the public cloud. There are too many knobs to control, and too many resources to configure in too many different ways across many different clouds.

This proliferation of options makes it impossible to ensure each and every resource is configured per security best practices. Even the knowledge required to be current with security best practices is different for different clouds. All of this leads to easily misconfigured cloud resources. This is a primary factor in security risks.

It's clear that this is *not* the way to do multi-cloud. A proactive multi-cloud strategy is needed, with security taking a lead role so that organizations stay in control and avoid security blind spots.

Security Challenges

Shared Security

A system is only as secure as its weakest link. Often, that weakest link is a person. And not surprisingly, it's human oversight and error that's often the cause of lapses in security. Deadlines and stress cause people to cut corners and not think of security until the deadline has passed and the project is running in production. Or not think of the security aspects at all, until something bad happens.

So even when security teams create and implement security policies, it only takes a single person to ignore, forget, or misunderstand the policies to create problems. This is often the case when software

Beware 'Shadow IT'

"Shadow IT" can be defined as information technology systems built and used within organizations without explicit organizational approval. These rogue systems are specified and deployed by departments or individuals other than the IT department, and operate outside of the normal oversight parameters.

Given how expensive downtime is in this era of always-on operations, Shadow IT can be a significant drain on a company's coffers.

or system engineers spin up a new cloud service to solve an acute problem, without thinking of the consequences in cost or risk.

There are simply far too many security settings for a human to control. It's not realistic to handle this chore manually; automated and programmatic testing and validation to enforce security control is required.

Risk Management

And this is where risk management comes in. In the era of fairly static on-premises data centers, security was baked in by highly skilled engineers. Nothing other than that single computing platform was used throughout the organization. The security team had a handle on the security perimeter and knew what to secure. New assets were brought in via a lengthy change process which included the right stakeholders from the get-go—including security teams.

In the new world of cloud, this completely changed. Anyone can start using a cloud service without involving the security team. Without security experts involved, services are easily misconfigured and best practices are not followed, especially in larger environments with hundreds or thousands of instances, objects and services.

So how do security teams stay in the lead? They must work with IT to offer cloud services to anyone in the organization needing them, not blocking progress or innovation, and not standing in the way of teams who have good reasons to consume public cloud resources.

Complexity of IT and Multi-Cloud

The freedom of choice in public cloud is incredible, but complexity is the enemy of security. With cloud making it so attractive to cherry-pick the best services from each vendor, the number of moving parts is exploding rapidly. And with every new cloud service in use, the complexity increases.

Each vendor has its own implementation of security and compliance policies that may or may not be compatible or interchangeable with other policies. Simply put: each new vendor adds another technology pillar for the security professional to master and secure. And each vendor has its own tools for securing its services, too.

For example, while the concept of securing a database is applicable to databases from any of the cloud vendors, the implementation details of securing an Aurora database running in Amazon Web Services (AWS) are significantly different than implementing security best practices on an Azure SQL instance.

Similarly, while the concepts of RBAC, least-privilege and just-in-time privileged access elevation are broadly applicable, the way to do this in Azure Active Directory is vastly different than in the Amazon Identity and Access Management (IAM) tool.

And this is in addition to the technology stacks that you run on-premises, which may be different than any stack inherited through mergers and acquisitions.

Cloud-Native and Developer-First: Security at Risk

With developers in the driver's seat more often, things aren't getting much better, security-wise.

Public cloud, containers, and Kubernetes are democratizing infrastructure components, making them directly available to developers.

Developers are growing accustomed to this level of access, expect all cloud services to be available immediately. The downside is that developers can struggle to keep up with cloud security best practices due to proliferation of different services in use, as well as the scale of the number of services used.

And the diversity of tools in their toolbox is growing, as well as their complexity. Public cloud vendors are creating attractive services aimed specifically at developers, ranging from databases and message queues to object storage, API gateways, DNS and service discovery, load balancers, and content delivery networks.

So how does the security team keep tabs on what services developers are using and know how to secure all of these services granularly?

Global Visibility, Granular Enforcement

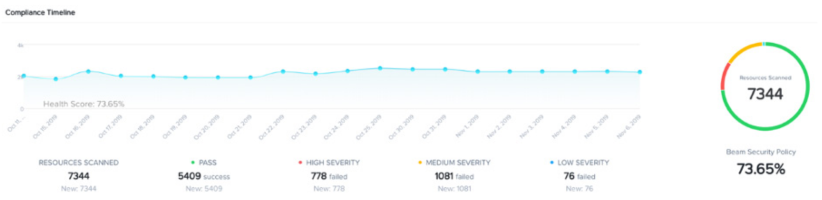
The key to solving security challenges in multi-cloud is twofold, summarized as "global visibility, granular enforcement."

Global visibility gives you that 10,000-foot overview of all secured assets, as well as a view of policies and the desired state of regulatory compliance. Knowing what assets comply with security and compliance policies is half the battle. That awareness changes the way security teams do business from reactively chasing incoming threats to knowing what threats to target. This helps security teams prioritize where to spend time helping other teams be secure and compliant.

Automatically enforcing these security policies at a granular level frees security teams from needing to know the nitty-gritty details or wasting time on *toil* and other repetitive tasks to implement the right security configuration and controls for each individual instance of different cloud services.

Start with Visibility

It all starts with making the current and desired state of security visible (**Figure 11**).



RESOURCES SCANNED	● PASS	● HIGH SEVERITY	● MEDIUM SEVERITY	● LOW SEVERITY
7344	5409 success	778 failed	1081 failed	76 failed
New: 7344	New: 5409	New: 778	New: 1081	New: 76

Figure 11: Xi Beam's compliance timeline helps track priorities.

Not knowing what you're aiming for in the longer term, what to do to get there, or what to do first forces you into a reactive state of mind and execution.

Instead, understanding the scope of the challenge using the cold, hard facts (like security issue severity) help you plan your road to success.

Monitor and Visualize the Network

From firewalls to VLANs, the network perimeter is where security was traditionally applied. The network was a good place to start in static data center environments, with an application landscape that rarely changed. If it did, it was a major overhaul involving the security team and other stakeholders. Applications tended to be commodity/off the shelf, arrived with proper documentation, and were well-understood by an army of system engineers and consultants.

Monitoring the network in this static world was relatively easy, even if some parts of the application ran as SaaS or in the public cloud.

In the cloud, things are different. Applications are created using standard cloud services (like databases, message queues, object storage, Java runtimes, and web servers), but the application itself is bespoke and unique to the organization. And sometimes the application spans across the private cloud, as well as multiple public clouds and SaaS.

The differentiator, though, is the speed at which the applications can change. In on-premises data centers built using legacy 3-tier architectures, changes took weeks to months. In public cloud, changes can be instantaneous.

Keeping track of these changes is no longer something humans should spend time on. Keeping track of the security posture, given all these changes, isn't either. You need automated tooling to create visibility into the networking between the enterprise and public clouds, no matter where (parts of) the application is running.

Segment Risk

And, unfortunately, the cloud isn't some magically secure place. Instead, the cloud is a giant toolbox, full of useful tools to create solutions for your (business) problems. But just as with any tool, you need to be able to use them.

And many of these tools aren't secure by default. Just as with any other piece of software, infrastructure, or tooling, you need to mold them into the security policies your organization defines. This doesn't mean the cloud is insecure, but the boundary of what they keep safe is more clearly defined. The service itself is secured by their brilliant security professionals; what you do with it and if you're using it securely is your responsibility.

Luckily, due to the way security *can* be implemented in cloud services, it's easier to actually take security further than the perimeter-based approach in legacy data centers.

Instead, a *zero-trust* model is the better approach, but needs more granular control than many legacy tools can provide. Examples of security approaches that do support this level of granularity are using just-in-time elevation for privileged access, microsegmentation for virtual machines and containers, and employing automated and

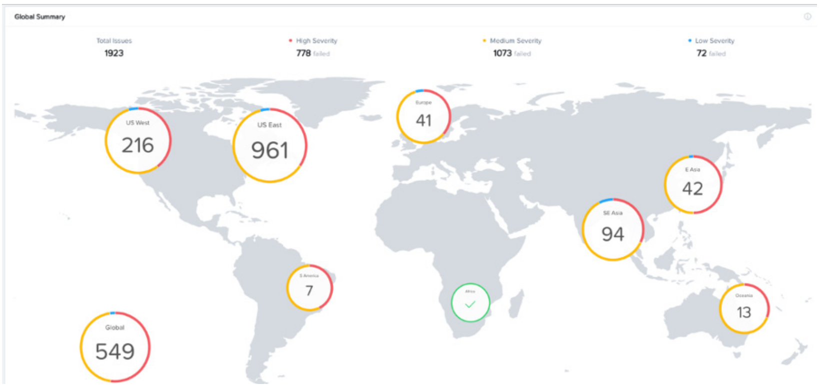


Figure 12: Multi-cloud security: buyer beware!

real-time checks against popular attacks (like SQL injection) for newly discovered cloud services.

This granular level of security prevents threats from propagating throughout the different public and enterprise clouds as the scope of the risk grows with cloud usage (**Figure 12**).

Policy-Based Security

But to have that level of granular security without being swamped with the implementation details for securing each individual service across Azure, AWS, and Google Cloud Platform (GCP), as well as the private cloud, requires an additional layer of abstraction.

This is provided by *policy-based* security, which is a rule-based, human-readable, business-oriented approach. In the policies, which are sets of rules, the desired security posture is described. These rules are high-level abstractions that humans can read and understand, declared in business-oriented language. For example: *All production storage assets must not allow global read/write access over public IP, and must always have encryption enabled on them.*

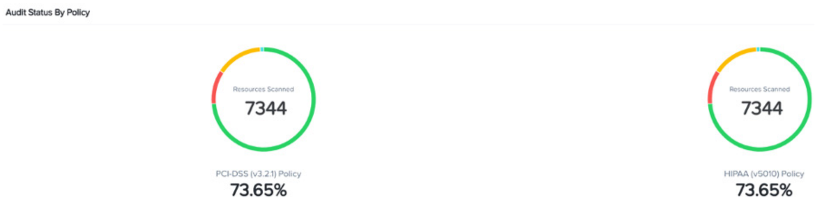


Figure 13: Xi Beam codifies regulatory compliance rules into actionable audit status checks.

Tools like Xi Beam codify these policy rule sets into actionable data, by scanning existing assets' compliance, as well as suggesting and automatically executing remediating steps (**Figure 13**).

Beam continuously scans the entire multi-cloud environment, spanning the enterprise cloud, public cloud vendors and SaaS providers to identify common security risks and vulnerabilities, as well as compare the current state of compliance with the desired state of compliance like PCI-DSS or HIPAA.

Automatically Enforcing Security Policy at the Granular Level

While real-time audit status detection is vital to knowing the current status of compliance, it's only half the story. The other, vital part of any out-of-compliance issue is remediation.

Normally, remediation makes the life of many a security professional very difficult. Because, while every cloud provider has a set of security best practices for their services, these guidelines are often not implemented correctly, or at all. This misconfiguration leads to security issues that are hard to pinpoint, let alone fix.

These issues require expert-level knowledge of not only the regulatory compliance details, but also the specific service's granular security best practices, which differ from cloud service to cloud service.

Always-on Cloud Security Compliance Enforcement

This highly involved and complex process of remediation simply doesn't resolve security issues and increase compliance coverage.

Real-Time Audits Drive Compliance

With the right tools, this manual process of remediation is automated and integrated with the real-time visibility to be able to act immediately on compliance issues as they occur.

For instance, take that compliance rule to encrypt all data storage for production environments. An automated check runs continuously, scanning all production storage assets. If a new unencrypted object storage bucket in the public cloud is spun up for production purposes, the check will flag this violation and enforce the encryption policy immediately.

Automated Remediation

This example highlights one of the aspects of public cloud that's normally an advantage, but works disadvantageously from a security perspective—how easy it is to spin up a new resource, in this case a storage bucket. Without real-time compliance-enforcement tools, this bucket would be left unencrypted, potentially exposing customer or financial data.

Similarly, the rate of change of cloud resources has made regular change management processes cumbersome and feel like a millstone rather than actually protecting the organization against issues. Many organizations are letting go of the traditional IT service management (ITSM) approach in favor of agile and DevOps.

What do all of these have to do with security audits and compliance validation? You need real-time insights, cloud security best practices and immediate remediation to stand a chance against hackers, common flaws, and pitfalls.

Ensuring Compliance with Regulatory Policies

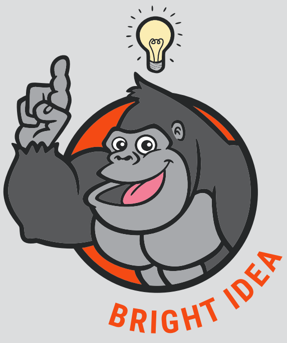
Regulatory compliance is increasing requirements around notification of the public after unintentional breaches. These requirements should make any company think twice about not having real-time violation detection and remediation in place.

Regulatory Compliance

Staying up-to-date with the latest regulatory and compliance requirements is hardly a fun part of any security professional's job. It's hard enough to understand just the four most common regulatory frameworks; GDPR, SOX, HIPPAA and PCI-DSS.

Beam stays up-to-date with these ever-changing regulatory rule sets, automatically codifying regulatory compliance frameworks into rules and checks, so security professionals don't have to.

Custom Audits



While industry regulatory compliance is a great start, many organizations want to take security audits to the next level.

By using the same real-time audit check engine for custom audits, any organization can create their own security controls, without the hassle of manual audits or remediations. Xi Beam supports this custom audit functionality with with Python scripts and a Query-based rule engine to automatically detect vulnerabilities, misconfigurations, or events captured in audit logs.

Policy-Based Management Is Key

This chapter married the concepts of real-time visibility across the multi-cloud environment with granular, automatic remediation and enforcement of compliance policies.

Fragmentation inhibits this visibility. A single-pane-of-glass perspective for all of the organization's cloud assets is critical. Real-time detection helps to actually stop the bad guys, instead of being warned after the fact when the hackers have made off with your valuable data.

As the complexity and number of cloud resources grows exponentially, enforcing and remediating security policies manually no longer scales. Humans simply can't keep up. Policy-based management is a key requirement for scaling up these efforts, putting the humans in the driver's seat, but automating the enforcement of policies and remediation of any violations.

Next Up

But what happens when there's a failure in the system that causes data loss, or a hurricane that wipes out a data center? That's when you need to rely on disaster recovery, and it's what we turn to next.

CHAPTER 5

Architecting a Cloud Disaster Recovery Service

Disaster recovery (DR) is an integral part of business operation continuity. A severe disruption of normal operating facilities, including hurricanes, floods, earthquakes, and human-made disasters, rarely stops customers, clients, and other business dependents from expecting service. Businesses are often expected to be available 24x7, and that means they need to plan for DR. Typically, in the event of a disaster, DR involves the restoration of business services by redeploying applications and infrastructure in new datacenters or locations.

Prior to receiving and providing services, customers and internal business units often negotiate service-level agreements (SLAs) that must be met. One of the most important steps in DR planning is working out agreements with stakeholders. The agreements must meet the demands of the average customer or client yet still be feasible to carry out while meeting the SLA time requirements. Specifically, the SLAs will include a set of service-level objectives, such as time to recovery and point of recovery.

Without a functional DR plan, a business is at risk. The cost of services being unavailable includes lost revenue during the outage, as well as long-term impact on customer confidence in your business's ability to deliver services.

When determining SLAs, it's important to keep in mind the opportunity cost of having your business systems down. Depending on your type of operation, you may incur lost revenue due to your sales system being down. Sales won't be made, and cash flow will halt. If you provide a long-term service or business solution, customer frustration will increase. This is especially the case if you provide support services or self-serve services customers depend on for their needs, such as checking the status of or reviewing personal accounts, that go down in a disaster.

Ideally, DR plans should minimize the time to recovery but also keep the cost of the DR operation to a minimum. However, these objectives can be in conflict when using traditional, dedicated backup data centers. The closer the configuration of the backup data center is to the production data center, the faster you'll likely restore services and avoid incidents while operating from a DR center.

Business Considerations When Choosing a DR Solution

While building a DR plan, businesses need to prioritize applications, workloads, and data. There must also be a set order in which applications and data will be restored to avoid further conflicts and interruptions.

First, a DR plan should include a complete inventory of applications and the data used by them. Customer-facing applications and services should be at the top of the priorities list, along with key operational systems that must work in order to keep other applications running.

Some back-end services that aren't customer-facing can be placed lower on the list of priorities. For example, it may be more important to get your order processing system up and running first and then tend to the inventory tracking system later. In this instance, sales will continue, and customers can continue to do business. Inventory tracking, while important, can be caught up on later after a more complete recovery has been made.

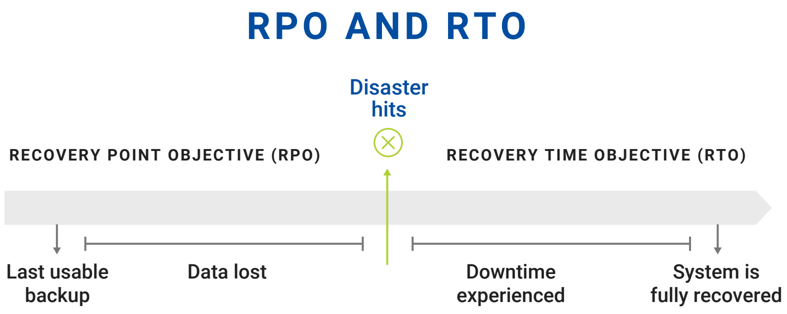

RPO AND RTO
Disaster hits
RECOVERY POINT OBJECTIVE (RPO)
RECOVERY TIME OBJECTIVE (RTO)
Last usable backup
Data lost
Downtime experienced
System is fully recovered

Figure 14: The differences between RTO and RPO

Additionally, business intelligence and analytics systems, such as machine learning workloads, depend on data from other systems. These workloads would likely have a lower priority than most online transaction processing systems.

When creating SLAs, businesses must define two objectives: recovery point objectives (RPOs) and recovery time objectives (RTOs). Your chosen recovery point will depend on your tolerance for data loss. See **Figure 14**.

Different use cases will have different tolerances. For instance, accounting systems can't tolerate any data loss that will leave them in an inconsistent or incorrect state. On the other hand, an inventory system has some room for error. In a system that tracks the SKU of each item in a warehouse, some data loss may cause an item to be overcounted or undercounted by a small amount. This problem can be remedied later and isn't usually disastrous to business operations.

In an even more extreme case, if data from a data mart were lost completely, administrators could rebuild the contents of the repository by re-running extraction, transformation, and load operations.

RTOs depend on a business's tolerance for downtime. Your business's services and customers' expectations will determine how much downtime is tolerable.

The more stringent the RTOs and RPOs, the greater the cost to a business. Cost-benefit tradeoffs usually come down to how similar a business needs a DR environment to be to their normal operation environment. A DR environment with small capacity and resource availability will have a low cost, but may lead to decreased performance or the elimination of low-priority systems and workloads.

On the other hand, a DR environment identical to the normal production environment will be able to carry out all normal business needs, but will have a higher cost.

Traditional Pre-Cloud DR Solutions

Traditional DR solutions tended to use either a cold site or active site recovery model. They each have benefits and drawbacks.

When using a cold site recovery model, a business contracts with a third party to provide needed hardware and network infrastructure during a disaster, which can lead to longer recovery times because teams will need to deploy a functional production environment on the infrastructure.

With cold site recovery, you have access to infrastructure, but your applications aren't deployed and data isn't up-to-date, and it will take time to deploy both. Cold site recovery is best for use cases with long recovery time objectives. Businesses that use this model should also have the personnel available to spend the time needed to install and configure all necessary systems and data.

The active site recovery model avoids some of the biggest drawbacks of the cold site recovery model, especially the longer time to recovery. This approach requires provisioning and maintaining dedicated hardware in a second facility. Applications are available and maintained in the active site even when the site isn't used for DR purposes.

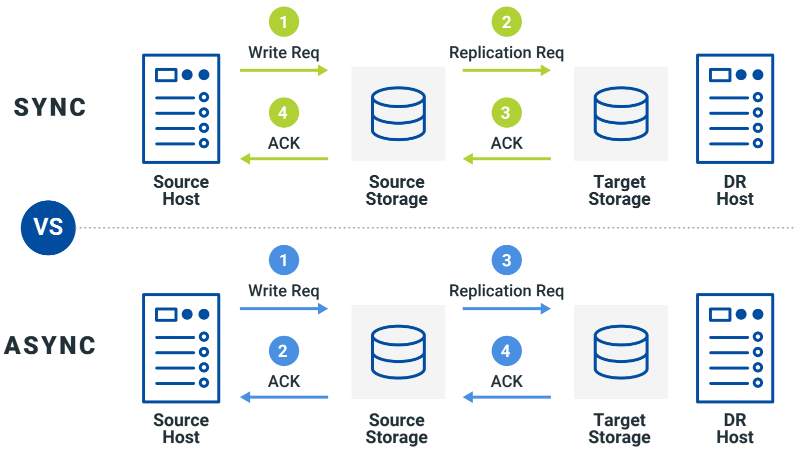


Figure 15: Synchronous vs. asynchronous replication.

Avoiding long recovery time comes with an ongoing cost of its own: With active recovery, you must keep replicating data because data is constantly changing in a production environment. If the DR site is to be ready at any time to take over production workloads, the data in the two sites must be kept up to date. This data replication may be done synchronously or asynchronously (see **Figure 15**).

Synchronous replication minimizes RTOs because each time a change is made in a production database, it's made in the corresponding database in the DR site. When a transaction is performed on the production database, the transaction isn't considered complete until both databases have been updated.

For example, if an inventory record is updated in the production database, the database management system would then initiate another update operation on the DR database. When the update is completed on the DR database, that database signals the production database that the data has been successfully updated and the transaction on the production system can be completed.

One of the considerations when choosing between synchronous and asynchronous is tolerance for increased latency. Synchronous replication can entail longer transaction times because data has to be transmitted between data centers during the transaction (Figure 15).

With asynchronous replication, data is updated in the DR database, but the transaction on the production system doesn't need to wait for the DR database transaction to be completed first. This enables longer distances between sites because transactions in the production system will be completed once data is sent to the backup site, without having to wait for the backup database to write and confirm the transaction.

This approach allows for longer latency when writing to the DR database without slowing the production system. In cases where the DR database can be out of synchronization for longer periods of time, a business may even choose to use batch processes to update the disaster recovery database.

Whichever form of DR you choose, you will need to maintain and test the DR infrastructure and systems. In addition to the maintenance costs, a DR environment is essentially unused capacity most of the time. Fortunately, virtualization and cloud technologies offer a better solution.

First-Generation Cloud-Based DR Solutions

Virtualization is redefining DR options. With virtualized infrastructure, there's no need to have dedicated hardware already provisioned in an active site or to wait for a team of system administrations and software engineers to deploy the latest versions of applications and data to a cold site.

Instead, virtual machines (VMs) running on-premises can be replicated in the cloud, essentially creating an active recovery site on demand. The key advantage to this approach is that the business doesn't have to pay for unused capacity; it only pays for the recovery site infrastructure while it's in use.

Many tasks remain the same as pre-cloud solutions. Network engineers, for example, would have to create virtual private networks and implement firewall rules in the recovery site. Authentication and authorization controls would also have to be established in the DR environment, and are especially important to keep up to date. Someone who had a privilege revoked a week ago shouldn't have that privilege available in a recovery site because authorization controls weren't kept in sync.

Maintaining up-to-date data in the recovery site, however, is still a challenge. One option is to maintain a hot standby in the cloud, much like a synchronously updated recovery database in the active site model.

Alternatively, a business may be able to take advantage of snapshotting disk images and storing them in lower-cost object storage. From there, they can be used to initialize disks on VMs that are started only in response to a disaster at the production site.

Integrated Cloud-Based DR Solutions

DR doesn't have to be a time-consuming, manual, and sometimes error-prone activity. Automation can help. IT engineers can automate many of the tasks of defining and creating a DR environment.

For example, system administrators can define VM templates or golden images so that they can be started rapidly. Database administrators can replicate data continuously so that it's up to date and available in the cloud. Network configurations can be scripted. The added advantage is that configuration code can be treated like any other application code and kept under version control.

Adopt the practice of using policy-driven operations and have well-defined policies in place so that engineers don't have to make guesses on the fly. For example, there should be no question about what AWS region should be used for a disaster, or the order in which applications are brought online. Policies should clearly specify where to establish DR resources. Replication policies should also be in place.

Keep in mind that different systems may have different replication policies. High-priority systems may require synchronous replication while lower-priority applications can be maintained using batch updates. Policies and plans should be tested regularly to verify that images and scripts function as expected in the DR environment.

Ideally, system administrators and application owners should be able to manage recovery operations from a single pane of glass. This kind of platform should give administrators everything they need in one place, so they don't have to jump from one dashboard to the next trying to piece together a comprehensive picture of the state of a DR operation.

The Best of Both Models

DR has traditionally been costly and difficult to implement. Businesses often had to balance the need for rapid recovery with the costs of maintaining unused infrastructure in an active recovery site.

Virtualization and cloud technologies now enable businesses to have the best of both models. VMs and applications can be started in the cloud when they're needed. Defining infrastructure and network configuration as code enables more automation, which reduces the time to deploy, as well as the risk of error.

Combine that with proper administration tools, such as a single pane of glass for managing DR operations, and DR is no longer the costly and difficult process it has been.

Next Up

That's also true of the lifeblood of so many businesses—their databases. The cloud can add efficiencies and increase productivity, which is why it's the focus of the next chapter.

CHAPTER 6

Managing Databases in the Cloud Era

Databases are ubiquitous in modern applications. Web applications use databases for user configuration data. Back office systems depend on online transaction processing databases for core business operations such as finance and inventory. Business analysts use data warehouses and specialized data marts to extract insights from increasingly large volumes of data generated and collected by other applications.

As crucial as databases are, they've traditionally been something of a bottleneck for agile software developers and require database administrators (DBAs) with specialized knowledge to support. While a host of software engineering tools have evolved to support rapid development and deployment of new application features, until recently, database deployments remained largely unchanged.

Today, virtualization and cloud technologies enable new ways to create, deploy, and manage databases. With the right practices, such as the use of standardized images and automation policies, DBAs and IT administrators can now realize the same agility with databases that they've experienced with virtual machines.

The combination of four core services can significantly reduce DBA workloads. They are:

- Support for database and server provisioning
- Copy data management

- Data protection
- Database patch management

To see why this is the case, consider the role of databases in today's enterprises.

Digital Transformation of Business Is Enabled by Agile Development

Digital transformation has allowed businesses to streamline processes, create new products and services, and drive other innovations. However, this evolution of the business environment wouldn't be possible without agile development and deployment techniques practiced by modern software engineering teams.

Developing an application, testing it, and then pushing an update to production is easier than ever because of agile tools.

Rapid development and deployment to production environments does require a specific kind of software development environment. First, integrated development environments should be flexible, with strong support for coding.

Code repositories are another crucial element of agile development. Repositories, such as GitHub and BitBucket, are needed to support collaboration so that developers can track their code and integrate it in controlled ways with other developers.

Finally, testing and deployment tools that automate unit testing and integration testing allow for fast, streamlined deployment of code to production environments. Together, these tools make for a fast and efficient development process. See **Figure 16**.

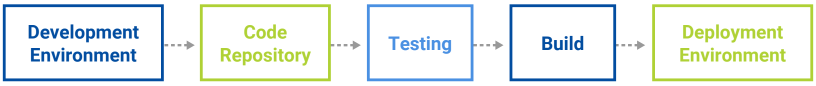


Figure 16: Agile software engineer practices enable a streamlined production pipeline of software development.

Database Provisioning Is a Bottleneck

Cloud computing and virtualization on-premises make it relatively easy for developers to create new virtual machines (VMs)—even entire clusters of VMs. With this ability, developers can quickly create development and test environments, as well as tear down these environments to make room for new applications and testing spaces. This is a great boon to any business operation, but it does come with some challenges, especially regarding deploying databases.

Many virtualized business applications will require the use of one or more databases. When using only on-premises systems for development and testing, database provisioning can become the cause of a serious labor bottleneck for developers, database administrators, network engineers, and even those working on infrastructure.

To begin, developers will usually create a ticket for a database administrator (DBA) requesting a new database. The DBA then sends a request to the infrastructure team to provision the hardware for this database. If the DBA is fortunate, hardware is already racked and available. Otherwise, there's a delay while equipment is procured and installed.

After the infrastructure team makes the hardware available, which may take time if the hardware is already in use for another job, the DBA begins installing the database software and any appropriate patches. Finally, the DBA and network engineers will work together to configure security controls, such as firewalls and access controls, to ensure the proper and safe use of the new database. There may be additional security checks, such as penetration testing, before developers are allowed to access the newly minted database.

As you can tell, this turns out to be a highly manual, time-intensive process. Overall, the manual intervention required to properly provision on-premises databases slows agile development. Developers can write code, integrate code from their peers, ready new features for release, and spin up servers for testing at a rapid pace. But this process can be suddenly blocked by the provisioning process, where databases will take longer to set up than the servers that would take advantage of them in testing and deployment.

Furthermore, even once a database has been provisioned, it may require configuration changes in response to changes in workloads, access permission changes, or new feature releases. A workload change that requires changes to the database itself, such as needing more persistent storage space, will increase the manual steps and downtime even more.

This is a hindrance for not only developers but also DBAs. The time and effort required to perform these manual processes present a significant opportunity cost for DBAs who could be working with developers to tune queries, optimize logical models, and make other performance improvements.

Although these cumbersome processes were necessary before, enterprises now have the opportunity to leverage virtualization technologies for their databases—particularly through support for automated database and database server provisioning as well as copy data management.

Managing Databases across Platforms

Cloud technologies enable a new way to manage databases. These new technologies mean that some time-consuming steps, such as provisioning hardware, can now be avoided. Instead of provisioning hardware for VMs manually, DBAs can now create VMs on cloud infrastructure, and with cloud resources they can apply existing policies to define network and security configurations to those VMs without the help of a network engineer. DBAs can now create managed images and snapshots that can be used to create new or cloned databases, which reduces the number of times DBAs must apply patches and modify configurations.

As you can see, virtualization and cloud technologies are streamlining the deployment of databases. For example, a DBA can create a VM from the command line at their desk, without the need for help from the infrastructure team.

Additionally, because cloud database services offer virtualized storage, the DBA can then quickly provision and configure block storage on a network attached storage system, and, in turn, attach it to the VMs they just created. Giving DBAs that ability to access cloud infrastructure eliminates the hardware provisioning time delay and actively promotes agile development.

Once databases are provisioned, the responsibility of DBAs then shifts to ensuring availability of the database with backup and recovery processes, as well as patching the data platforms.

DBAs are increasingly responsible for databases deployed on-premises, in hybrid clouds, and on public cloud platforms. While this can benefit developers and application owners who can choose the optimal platform for their requirements, it can make managing those databases more difficult. One way to compensate for this added complexity is to use database management tools that provide a single control plane for databases across platforms.

It's also true that new technology presents new challenges, and cloud databases are no different. With the ability to rapidly provision servers and storage comes the ability to rapidly deploy misconfigured resources. In particular, overprovisioning of storage and incorrect use of expensive storage and server types can lead to high costs with little reward.

As such, it's important to standardize deployment practices with policies and golden images. Each step that requires human intervention is an opportunity to make a mistake. The configuration of the databases needed should be used to define standard images. These images can then be used to deploy properly configured databases in an automated way. See **Figure 17**.

In the end, this standardization reduces the need for DBAs to learn best practices for multiple databases and instead promotes a structured approach to utilizing any database system.

For instance, security configurations can be database- and operating system-specific. Different databases may have different role-based access control and identity management commands. High-availability databases require the replication of data and running of multiple servers. Understanding how to specify clusters like Oracle

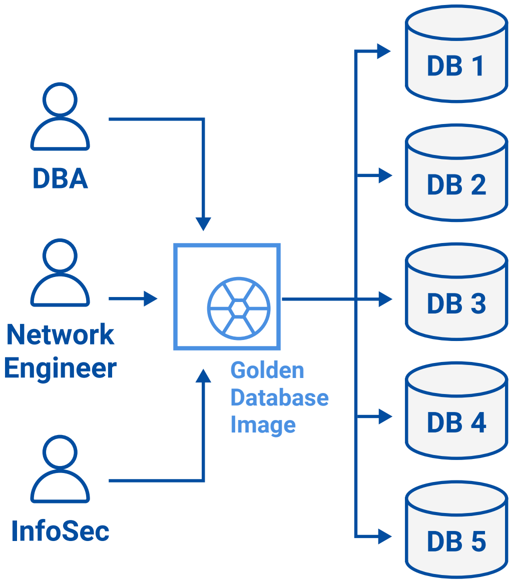


Figure 17: Standardized images improve consistency and time to deployment.

RAC or high-availability PostgreSQL is complicated. Learning how to optimally configure on these platforms can take months of the experience.

Additionally, operating system tuning for the database is idiosyncratic, even when running databases on a single type of database, such as Linux or Windows Server. Differences in kernels and libraries included in different Linux distributions can introduce subtle but impactful effects on performance.

It's important to first find the most cost-effective way of provisioning databases that suit your particular business needs, and then use those practices as a foundation for future deployments and provisioning.

Another advantage of using standard images is that security and regulation compliance become easier to enforce and document. For example, security engineers can help DBAs harden an operating system and lock down the database management system. The process can be time-consuming when done manually and repeatedly for each database instance, but by using standard images, it can be done once and then replicated as many times as needed.

Automation Policies

In order to streamline management of databases, it's important to implement a centralized management system from a single pane of glass and to define automation policies for the most common administration tasks. With a centralized management system, you should be able to patch databases quickly. This is most important for security patches, as their deployment can have far-reaching consequences, but can also apply to performance-enhancing fixes.

A centralized, single-pane-of-glass system will also allow for easier, more efficient monitoring of the databases in use. This includes infrastructure monitoring, such as CPU or memory utilization, as well as database-specific monitoring, such as identifying the longest-running queries. Identifying these top queries will help admins understand cache hit rates so that future queries and configuration can be more effectively tuned.

A strong centralized management system will also provide an alerting system. Alerting is needed to notify DBAs and developers when something about the database system needs attention. Proper alerts can help prevent running out of persistent storage by notifying someone when a disk reaches a particular threshold, such as when 80% of allocated storage is in use. Not only will these alerts increase performance, but they'll also help keep costs down.

Automating common administrative tasks is also an important part of streamlining the use of cloud technologies. For instance, define policies that purge old, unneeded data from databases to free up storage space for data in use. Data retention policies like this will help DBAs avoid storage bottlenecks and overprovisioning.

Policies dictating access controls and role assignment are also possible. For instance, a developer may need access to a secure database for a short-term project. They can be assigned a role that allows them access for only the allotted project time. When the deadline is met, access permissions will be returned to their previous state. Again, this type of automation frees up DBAs to work on other tasks.

There are many benefits to automating administrative tasks. It promotes agile development, increases the number of databases that a single DBA can support, and improves the monitoring of cloud services. Excellent monitoring is important in all industries, but it's most needed when handling medical or financial information. Monitoring will ensure industry regulation compliance and customer safety. Finally, not only will DBAs be able to support more databases, you'll be able to leverage their expertise for higher-level tasks, such as data architecture and database application optimization, adding more value to your operation.

New Ways to Think About Databases

Developers need DBAs to provision databases for their workloads. Manual provisioning can often create bottlenecks in a development pipeline, slowing development and eating up resources. Now, with cloud technologies, many parts of this process can be automated and accelerated.

Virtualization is changing how organizations think about database provisioning, although it does present new challenges. With such ease and speed of use, virtualized databases can be improperly provisioned, causing problems for both developers and DBAs.

A way to combat improper provisioning is to use standardized images, replicating databases, and VMs so that future deployments will follow a tested, properly provisioned structure. This kind of automation does require new supporting tools and practices.

In particular, DBAs benefit from support for:

- New database and database server provisioning
- Copy data management services to reduce storage
- Data protection through snapshotting
- Database patch management

Centralized management systems allow automated patching and alerting, as well as database monitoring from a single pane of glass. Overall, proper automation and provisioning practices will greatly increase the effectiveness of DBAs and developers, allowing them to take on more complex, forward-thinking tasks.

Up Next

Now we turn from the mature, well-understood technology of databases to the shiny and new stuff. That means containers—and using containers increasingly means using Kubernetes. It's what you'll learn about next.

CHAPTER 7

How Containers Are Reshaping the Enterprise

Kubernetes has taken the world by storm. There's hardly anybody in IT infrastructure not talking or learning about Kubernetes, and this will not change in the coming years.

This is mostly due to two changes in the IT landscape:

1. Self-service cloud and infrastructure as code let developers safely provision IT infrastructure independently from IT operations teams

2. The application development landscape is shifting to cloud-native architectures

In this chapter, we'll talk about how both of these shifts in IT are affecting IT infrastructure, shine a light on the immense popularity of Kubernetes for developers, and dive into the advantages during development work.

We'll also look under the technology hood to learn about Kubernetes concepts and architecture. We'll give special attention to the operational part of running Kubernetes in production, an aspect that has incorrectly taken the back seat in many organizations due to its complexity and the fact that the majority of enterprise organizations lack the experience and internal IT resources to manage Kubernetes and serve the needs of developers in a productive manner.

So, if you're ready to find out how to not only use Kubernetes, but get more from it, read on.

Sketching the Cloud-Native Landscape

The cloud has changed many aspects of how we run IT. Applications are easier to use as a cloud service (Software as a Service), infrastructure is no longer a capital-intensive and high-friction endeavor, and deploying new resources is as easy as clicking a single button or running a single script.

The National Institute of Standards and Technology (NIST)[3] defined cloud along three axes to differentiate it from the data centers of yesteryear. Let's recap these three axes to fully understand the differences between cloud and our own data centers:

1. Cloud Characteristics
2. Cloud Deployment Models
3. Cloud Service Models

5 Key Cloud Characteristics

Cloud computing has a set of five unique characteristics that make it fundamentally different from on-premises data centers, colocation, or hosting:

- **On-demand self-service:** A customer can provision computing capabilities as needed, without requiring human interaction.
- **Broad network access:** Capabilities are available over the Internet without special networking requirements.
- **Resource pooling:** The provider's resources are pooled to serve multiple customers using a multi-tenant model.
- **Rapid elasticity:** Capabilities can be provisioned and released to scale rapidly outward and inward commensurate with demand; resources often appear to be unlimited in capacity.

[3] https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

- **Measured service:** Cloud systems leverage a pay-per-use billing mechanism, shifting the cost for infrastructure from capitally intensive to operational cost; resource usage can be monitored, controlled, and reported, providing transparency for both the provider and customer of the utilized service.

3 Cloud Deployment Models

Note that these five characteristics don't mention ownership or who manages the infrastructure. Often, cloud is synonymous with public cloud, but there are a few more flavors:

- **Private cloud:** A single-tenant cloud infrastructure, provisioned for exclusive use by a single organization comprising multiple customers (like brands, business units, or subsidiaries). It may be owned, managed, and operated by the organization, a third party, or some combination of them, and it may exist on-premises or off. The key is that even this single-tenant cloud infrastructure adheres to the five key characteristics.
- **Public cloud:** A multi-tenant cloud infrastructure, open for use by the general public. It may be owned, managed, and operated by a commercial entity.
- **Hybrid cloud:** A composition of two or more distinct private and/or public cloud vendors with data and application portability across clouds.

3 Cloud Service Models

Different vendors offer different cloud services. The most common services can be attributed to one of these three service pillars.

- **Software as a Service (SaaS):** Offering as a cloud service a specific application, which is often end-user facing. The consumer doesn't manage or control the underlying infrastructure or even individual application capabilities.

- **Platform as a Service (PaaS):** A service to deploy and run a custom application, often built in-house by the customer. The deployment and runtime platform have support for certain programming languages, libraries, services, and tools only. Again, the consumer doesn't manage or control the underlying infrastructure, but has control over the deployed applications.
- **Infrastructure as a Service (IaaS):** The offering of infrastructural building blocks of compute, storage, and networking capabilities. Consumers can provision these at will to deploy and run arbitrary software, which can include operating systems and applications.

The Definition of Cloud Native

So, we know that clouds are characterized by five traits, can be deployed in different ways, and offer services up and down the technology stack, from infrastructure to application level.

But what does that have to do with the cloud-native landscape? Well, it's important to understand the broader context of cloud computing across all these options to appreciate what cloud native is.

Cloud native is **an approach to application development that uses discrete and reusable cloud services as building blocks for building and running software.** Cloud native is about *how* applications are created and deployed, not where, and takes full advantage of cloud characteristics of elasticity. A cloud-native application is specifically designed to run in a distributed architecture and in a loosely-coupled manner—this helps prevent lock-in to underlying infrastructure components.

There are a number of key aspects of cloud-native applications that are widely adopted:

- Cloud-native applications are packaged as a collection of single-purpose microservices

- Services are lightweight workloads, running in containers, isolated from server and operating system dependencies
- Cloud-native applications are not bound to the underlying infrastructure. Instead, they use Infrastructure-as-Code to define the requirements of the infrastructure underneath.
- Infrastructure is no longer a generic, pre-existing platform; underlying infrastructure resources are created on-demand for a specific application, and can be modified, scaled and deleted based on the application load
- Infrastructure resources are consumed on-demand and in a self-service manner by the application developers and DevOps teams, no longer requiring a separate Infrastructure team to provision resources and manage lifecycles

The Cloud Native Computing Foundation (CNCF), launched by the Linux Foundation in 2015, defines cloud native this way:

> *[Cloud-native] technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.*
>
> *These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.*

You could argue that cloud native is a broad set of architectural patterns for scalability, dynamic behavior, resiliency, immutability and high-change environments, with a focus on operational excellence with abilities like decoupling, observability, and automation.

Note that it's not specifically a single cloud vendor or technology, but rather all clouds and technologies that enable software and system engineers alike to build and run applications. This comprehensive approach makes cloud native appealing. It's not just about technology, but rather about how technology is used within organizations, and what outcomes they achieve.

Flow and Decomposition

Implied in this definition are the use of *atomic* cloud services that serve just a single purpose as a service. Many of the moving parts of an application stack are standard and common technology components, not specific to the organization. Instead of re-inventing the wheel, using open source software for those components makes sense. Other than a few niche and extreme use cases, why build your own database engine, caching layer, or web server?

That's why many public cloud providers offer those components and middleware as a service. The goal is to make consumption as frictionless as possible. Developers can simply configure their desired software stack with a few clicks, using databases, message queues, object storage, API gateways, DNS and service discovery, load balancers, and content delivery networks.

Those familiar with lean, agile, and DevOps might recognize aspects of those methodologies in cloud native. Cloud native is very similar, in regard to reducing complexity, reducing size of components (or work), and creating (one-piece) flow. Where these concepts in lean, agile, and DevOps apply to how people work, in cloud native they're applied to the architecture of the application.

In the new cloud-native paradigm, developers actively break down their work into smaller chunks and create (single-piece) flow of changes to production. This takes control and ownership over the pipeline that brings code from local testing all the way to production. Containers, microservices, and cloud-native services are facilitating this.

The Move to Containers

One of the most prevalent deconstructions in cloud native is the container. Containers have been a cornerstone for the cloud-native movement, and many of the advantages of cloud native come from using containers.

What Is a Container, Anyway?

In many ways, a container is similar to a virtual machine. They're both abstraction layers that isolate different layers in the infrastructure. VMs isolate the physical hardware and everything from the operating system and up. Containers isolate application binaries, configuration, and data from the underlying operating system. Both serve to simplify the deployment and operations of applications in different ways. See **Figure 18**.

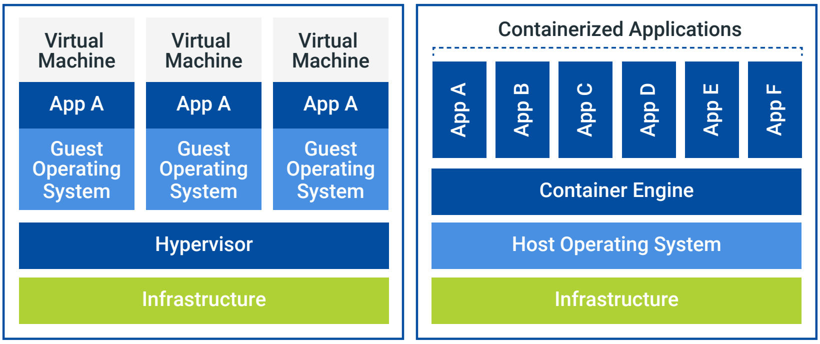


Figure 18: Virtual machines versus containers.

Unlike VMs, containers are lightweight and easy to move between infrastructure platforms. Running containers share not only the underlying hardware, but also the underlying operating system. Where a VM does include an operating system per application, a container does not. This completely decouples an application from the underlying commodity infrastructure, including the operating system.

Harder, Better, Faster, Stronger

Their small size makes it easier to tear down a container and replace it with a newer version. Combined with the decoupling of a container's state (like application data) from the running code, updating applications or a dependency is a much simpler operation when compared to VMs. Because the layers are decoupled, containers are rarely patched or updated. Instead, a rolling upgrade mechanism deploys containers with the new version and diverts traffic to the new version; the old version is killed soon after.

This is especially valuable in developers' workflow, which might require multiple or even dozens of build-run-teardown cycles per day. Decreasing the time it takes each cycle makes developers more productive and allows them to deliver better code more quickly. This reduces deployment time, encouraging continuous testing, delivery, and deployment while improving the quality of code that gets pushed to production.

Containers are also easy to spin up on different hosts, clusters, or clouds without changing the container image or its configuration. The operating system becomes a commodity to developers that they no longer care about and "just works."

The Cloud Fallacy, or How 'Cloud' Does Not Equal 'Cloud Native'

Often, the benefits of containers and cloud native are discussed as if they're interchangeable. And while containers fit into the cloud-native narrative perfectly, cloud native encompasses more than just containers.

One of the key traps many organizations fall into is the *lift and shift* fallacy, where they try to move existing, often monolithic, legacy applications to the cloud, and discover they're not taking advantage of the essential characteristics of cloud. In other words: while they **are** running in the cloud, they **are not** cloud native.

In order for applications to take full advantage of cloud native, they need to be re-written to handle the ephemeral, scalable nature of cloud services. In practice, much of this re-writing comes down to containerization and decomposition.

Removing hardwired dependencies on underlying infrastructure is critical for successful container deployments. This makes sure that applications can run on ever-changing underlying hardware, as well as scale beyond the one-to-one mappings that often exist in monoliths. The intent is to prevent issues with scaling beyond, for instance, a hardcoded single database server.

Deconstructing the application into its atomic parts, or microservices, makes sure each container can independently scale or be updated by a development team without a lot of toil, rework, or dependencies. Each of these services is responsible for a self-contained business function.

The Operational Nightmare of Managing Individual Containers

Moving applications to containers, and deconstructing them into microservices often creates container sprawl. Managing this explosion of microservices turns out to be an operational nightmare. It isn't *just* the sheer number of moving parts, it's also the immense churn of containers due to their ephemeral nature.

Manual control and static configuration for containers at this scale just isn't feasible. The deployment and operations of containers in production becomes a nightmare, and a better solution to orchestrate the deployment and operations of containers is required to handle the scale and complexity automatically.

A Need for a Container Orchestrator

And this is where Kubernetes comes in. Kubernetes has emerged as the de facto platform for running and managing containers at scale.

Kubernetes is a container orchestration platform that helps users build, scale and manage modern applications and their dynamic lifecycles. The cluster scheduler capability lets developers focus on code rather than operational work. Kubernetes future-proofs infrastructure management on-premises or in the cloud, without vendor or cloud provider lock-in. Kubernetes is specifically designed to manage the ephemeral nature of thousands of containers spinning up, scaling up, winding down, and upgrading.

The platform brings together infrastructure operations and software development by design. It uses declarative, infrastructure-agnostic constructs to describe applications and how they interact, without the traditional close ties into the underlying infrastructure.

Kubernetes runs just as well on traditional on-premises infrastructure stacks as it does for third-party service providers and public cloud environments.

Kubernetes, at first glance, seems to do a simple task. Start, keep track of, and shut down containers. But with highly ephemeral containers, no-downtime rolling upgrades, and self-healing capabilities, Kubernetes becomes complex, quickly.

Beneath the covers, Kubernetes manages the horizontal scaling of containers to optimize application performance, handles versioning of containers, figures out how containers should talk to each other

Just Like vSphere

For those of you that have experience with virtualization platforms like VMware vSphere or Nutanix Prism, this sounds familiar. Just like those virtualization platforms, Kubernetes is the platform for managing workloads, where everything in the infrastructure comes together and plugs into a central platform.

Just like how storage, networking and security, backup and disaster recovery, monitoring and logging, databases, orchestration and automation, and more all plug in to vSphere or Nutanix, a new ecosystem of similar solutions is forming around Kubernetes.

Kubernetes can be thought of as a container-centric computing platform. It has much of the flexibility of Infrastructure as a Service in terms of managing compute, storage, and networking resources. On top of that it brings the developer-friendly workflows and constructs found in Platform as a Service. These include deployment, scaling, load balancing, logging, monitoring, and composition of application containers across clusters of container hosts.

over the network, exposes services running inside containers to the network, and handles storage claims and assignments across the container lifecycle. It also deals with failed hardware, maintaining container availability, and self-healing containers without downtime.

Even with this level of complexity, Kubernetes is great for deployment operations. No matter how complex an operation is, Kubernetes executes reliably with just a few simple commands or changes to container configuration files. Using declarative language, Kubernetes continuously drives the current state toward the provided desired state, without any effort or babysitting on the engineer’s part.

Why Kubernetes: The Platform of Platforms

We begin to see that Kubernetes is more than “just” a container orchestrator or resource scheduler. On the infrastructure side, it aims to remove the toil of orchestrating compute, network, and storage resources, as well as abstracting those constructs so application developers and operators can focus entirely on container-centric workflows and self-service operation, completely decoupling containers from the underlying infrastructure.

Kubernetes is commonly used for running microservices in containers. In its early years, it mostly ran stateless apps, but as the platform matured, more and more storage integrations natively support stateful applications by storing the state on an external data store, like a database, object store, or storage array. Many organizations deploying Kubernetes actually use existing storage assets for stateful container storage.

On the container side, Kubernetes provides a platform for building customized workflows and higher-level automation. It integrates into the continuous integration/continuous delivery (CI/CD) pipelines developers use to bring code into production in a controlled, tested and automated fashion.

Kubernetes manages some of the complexity of running very dynamic systems at scale, with hundreds, thousands, or more containers coming online and being destroyed again daily. Kubernetes keeps track of which services run in the containers, exposing them via DNS for service discovery and load balancing network traffic across all containers for that service.

Invariably, when you deal with infrastructure and applications, you’re dealing with secrets for deployment and configuration management purposes. These secrets include passwords, tokens, keys, and certificates, and they cannot be included in the actual container images for security reasons. Instead, Kubernetes lets you store and manage these secrets for use during deployment or operations, without exposing secrets in the container configuration or image.

Kubernetes Concepts and Architecture

Kubernetes is a very flexible and extensible platform. It allows you to consume its functionality à la carte, or plug in your own. To get a better understanding of how Kubernetes works, and how it can be extended, let's look at its core constructs and concepts. See **Figure 19**.

1. Control Plane: cluster and nodes
2. Workloads: pods and services
3. Networking
4. Storage

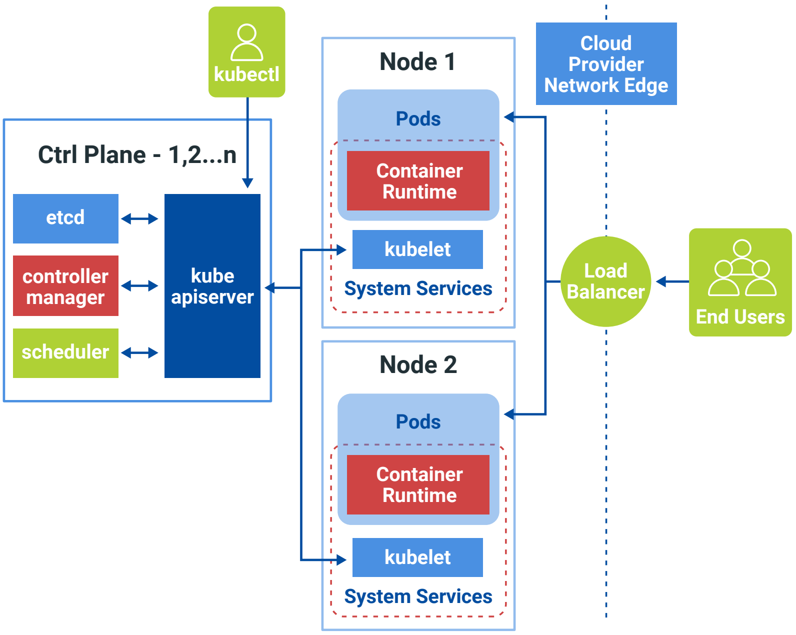


Figure 19: An architectural overview of Kubernetes.

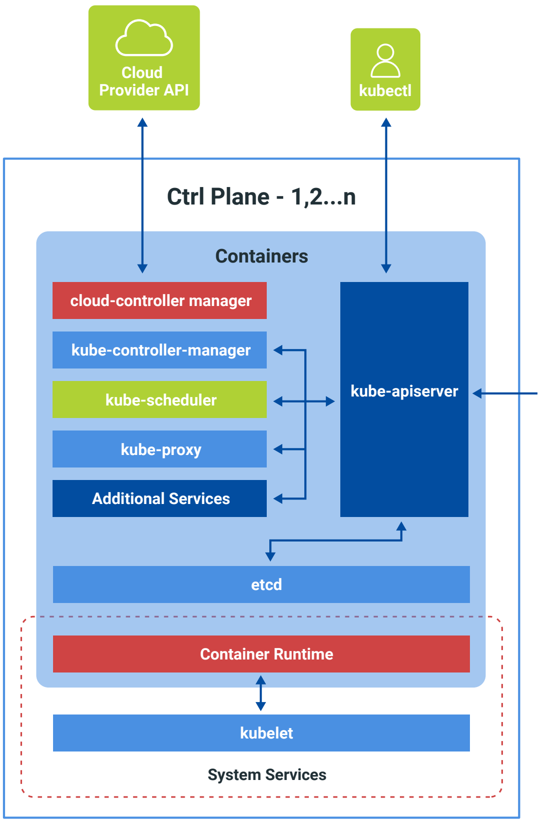


Figure 20: The Kubernetes control plane.

Control Plane

The Kubernetes control plane, as shown in **Figure 20**, is the set of self-driving processes that maintains a record of all Kubernetes objects. These processes continuously respond to changes in the cluster to drive the state of these objects to the desired state.

The control plane advertises containers that are ready to serve traffic, restarts containers that fail, replaces containers with new versions using rolling upgrade mechanisms, and kills containers that don't respond to user-defined health checks.

Let's zoom in on the specifics.

The control plane is made up of four major components:

1. **Kube-apiserver**, which provides APIs that act as the gateway to the cluster. It has a backing datastore to store the state of the objects configured in the system.
2. **Kube-controller-manager**, which manages the various controllers and makes changes to drive the cluster toward the desired state.
3. **Controllers** work to drive the actual (observed) state toward the desired state (specification), using labels and annotations. There are various controllers to drive state, including for nodes, replication (autoscaling), endpoints (services and pods), and namespaces. This includes cloud controllers, which are specific to each public cloud to optimize support for availability zones, VM instance types, storage services, and networking services for DNS, routing, and load balancing.
4. **Kube-scheduler** is responsible for the actual scheduling of containers across the nodes in a cluster, taking various constraints like resource limitations, guarantees, and affinity into account.

These can all run on a single master node or be replicated across multiple master nodes in the cluster for high availability.

Cluster Nodes

Cluster nodes, also called worker nodes, are the virtual or physical machines with a container runtime engine, managed by the master modes. The **Kubelet**, the cluster node manager, runs on each node and is responsible for driving the container runtime engine, which is often the Docker Engine.

Pods, Services, and Namespaces

Pods are arguably the most important concept in Kubernetes, and the main construct developers interact with. A pod packages up a single service or application, and can consist of multiple containers, storage volumes, and secrets. See **Figure 21**.

A pod can be horizontally integrated for horizontal scaling, meaning it's an independent unit of scale, containing a single service or application role. Or a pod can be vertically integrated for multi-tenancy, containing a complete application (like a WordPress stack with a web server and database in the same pod).

Pods are, like containers, volatile. They have a limited time span. When scaling down, or upgrading to a new version, pods eventually die. The transient nature of pods creates the problem of keeping track of which pods do what, and what their state is. This is where **services** come in. Services are the long-living access points into the ever-changing collection of pods that make up an application. Services are somewhat comparable to a Virtual IP (VIP) on a load balancer. The VIP is a way to address the underlying servers as a

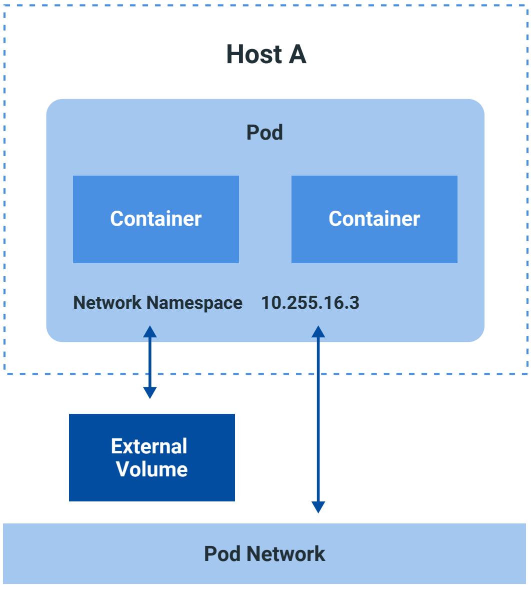


Figure 21: A typical pod architecture.

whole. A Kubernetes services is a way to address all pods that are labeled with that service.

Kubernetes uses dynamic labels to assign pods to services, which makes upgrading versions or scaling a service easy. Anytime a pod with the correct labels spins up, it's assigned to the service.

Labels are a fundamental part of how Kubernetes works operationally. They can be used for assigning pods to services or annotating release state (development, testing, production), app tier (front-end, back-end), or affinity (customer, brand). See **Figure 22**.

Namespaces are virtual clusters within a Kubernetes cluster. They're useful for carving a cluster into logical resources, so different teams can use Kubernetes without getting in each other's way. Namespaces are a way to divide cluster resources between multiple

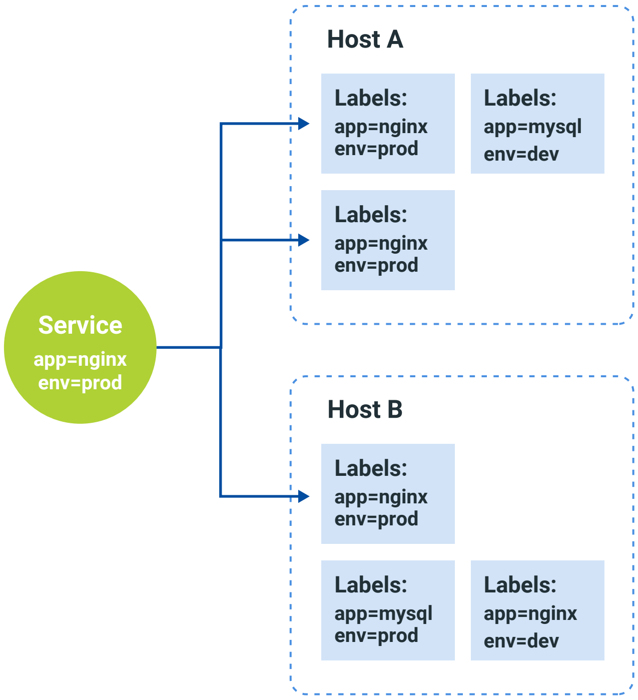

Host A
Labels: app=nginx env=prod
Labels: app=mysql env=dev
Labels: app=nginx env=prod
Service app=nginx env=prod
Host B
Labels: app=nginx env=prod
Labels: app=mysql env=prod
Labels: app=nginx env=dev

Figure 22: Labels are a fundamental part of how Kubernetes works.

users. Namespaces have their own services, labels, and pods: Names of resources need to be unique within a namespace, but not across namespaces.

To tie this all together, imagine a WordPress application. You run a single pod with the web server (and WordPress application) in one container, and the database in another, and label it with the the "app=wordpress" tag. You create a service that exposes port 443/ HTTPS publicly, and attach it to the label. Using this labeling system, you're now exposing the pod to the Internet. If the pod is replaced by a new version or substituted after a host failure, a new pod takes its place. Because of the label, the pod is automatically published via the service. The database doesn't need a service, because it's directly reachable inside of the pod by the web container.

Networking

Kubernetes has a distinctive networking model for cluster-wide, pod-to-pod networking.

Kubernetes uses a simple overlay network (like Flannel) to obscure the underlying network from the pod by using traffic encapsulation (like VXLAN). It can also use a fully routed solution like Calico. In both cases, pods communicate over a cluster-wide pod network, managed by a Container Network Interface (CNI) provider like Flannel or Calico.

Within a pod, containers can communicate without any restrictions. Containers within a pod exist within the same network namespace and share an IP. This means containers can communicate over localhost. Pods can communicate with each other using the pod IP address, which is reachable across the cluster.

By default, services are only reachable inside the cluster, but Kubernetes can automate the cloud provider's load balancer configuration to make them available publicly. Recently, Kubernetes introduced support for Ingress,[4] the new way to access pods from outside the cluster, which is becoming the predominant approach.

Given the ephemeral nature of pretty much anything that goes on in Kubernetes, discovering and publishing services is a crucial part of a healthy cluster. Kubernetes relies heavily on its integrated CoreDNS service to create, update, and delete DNS records for services and Pods in the cluster.

Persistent Storage

Kubernetes has support for persistent storage. A *persistent volume* is any storage object that has been provisioned by an admin to be used by the cluster. A *claim* is a request by a pod to consume the storage resources configured in the persistent volume.

There are many options for mounting both file and block storage to a pod. The most common ones are public cloud object and block storage services, or types that hook into an existing physical storage infrastructure.

Additionally, StorageClasses act as a way to differentiate the quality of the underlying storage. They can be used to label different characteristics, such as performance or resilience.

Kubernetes Solutions

You've seen that Kubernetes solves the problem of managing containers in production.

While Kubernetes is laser-focused on deployment and management of microservices and cloud-native applications, it really supports deployment of any type of workload, including batch jobs, stateful services, and legacy monolithic applications.

[4] https://kubernetes.io/docs/concepts/services-networking/ingress/

Kubernetes is built as a collection of pluggable components and layers. This architecture allows for many internal components to be replaced or extended by third-party solutions and add-ons. And the architecture clearly prioritizes high flexibility and customizability over ease of use.

Do-It-Yourself

All this flexibility makes installing Kubernetes non-trivial. Correctly configuring and maintaining a large production system is complex and requires expert-level knowledge, and the learning curve to become an expert is extraordinarily steep. Building your own Kubernetes deployment is daunting, even with tools like *kubeadm*, which bootstraps best-practice minimum viable clusters.

Packaged Distribution

Many are choosing to not "do Kubernetes the hard way," but instead rely on packaged solutions. Popular distributions that make the deployment of Kubernetes simple are K3S, Minikube, and MicroK8s. While these make installing and configuring easier, they don't take away all the complexity.

Public Cloud Service

A more popular way of using Kubernetes is by leveraging a turnkey cloud solution. Many of the public cloud providers offer Kubernetes as a Service, taking care of the control plane of the cluster. Examples include Amazon Elastic Kubernetes Service (EKS), Google Kubernetes Engine (GKE), and Azure Kubernetes Service (AKS). Unfortunately, these services tend to lock you into exclusively using cloud-based worker nodes, which isn't always an option.

Managed Turnkey Service

For enterprises, various cloud-like managed services exist and remove much of the friction of designing and implementing a Kubernetes environment yourself. These services make best-practice decisions around multi-master configuration, data store colocation and resiliency based on your sizing inputs. These kinds of solutions also integrate into existing enterprise compute, storage, and networking infrastructure more easily.

Choosing the Right Solution

Every organization has to evaluate cost, risk, and current investments to ultimately make the right choice for running Kubernetes.

Building your own gives you full flexibility, but is very hard, and very time-consuming, to do. The consequences of making the wrong decisions can haunt you for a long time after the platform goes to production and can impact application availability and performance.

Putting your eggs all in one cloud vendor's basket will reduce your freedom of choice in more than just the Kubernetes service. These cloud vendors are more than happy to lock you into their ecosystem.

While Kubernetes itself doesn't move the business forward, quickly deploying new applications and versions to customers does. For developers—Kubernetes end users—Kubernetes is the means to an end, not the goal itself. Developers don't care who builds it or how it gets there. They just want to get their hands on it and for it to work well.

A managed service takes care of that problem by enabling frictionless and quick deployment of Kubernetes inside the enterprise data center, automating Kubernetes cluster lifecycle management and ongoing operations.

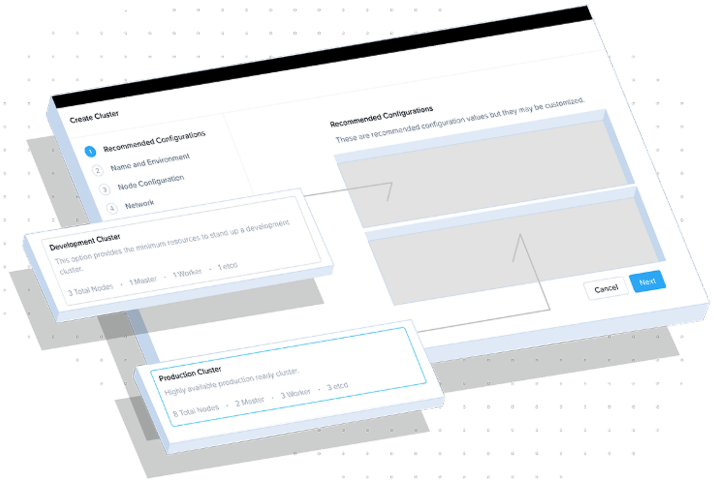

Create Cluster
1 Recommended Configurations
2 Name and Environment
3 Node Configuration
4 Network
Recommended Configurations
These are recommended configuration values but they may be customized.
Development Cluster
This option provides the minimum resources to stand up a development cluster.
3 Total Nodes · 1 Master · 1 Worker · 1 etcd
Production Cluster
Highly available production ready cluster.
8 Total Nodes · 2 Master · 3 Worker · 3 etcd
Cancel
Next

Figure 23: Nutanix Karbon is integrated into the Nutanix platform for turn-key Kubernetes support.

These reasons make it clear that for many organizations, a managed Kubernetes solution that seamlessly integrates with their existing datacenter and infrastructure investments is the best option.

A good example of a managed enterprise Kubernetes solution is Nutanix Karbon. Karbon runs as a completely integrated service on top of Nutanix HCI platform. Karbon's enterprise features help IT administrators make sure that their Kubernetes efforts are secure and compliant without the manual effort of installation and ongoing operations.

Karbon is highly integrated into the hyperconverged stack, leveraging the underlying storage, compute, and networking services automatically. See **Figure 23**. Karbon includes an integrated container storage interface that delivers persistent block, file, and object storage.

Deployment is quick and simple using a UI-driven deployment process to instantiate clusters in a recommended configuration. In line with the one-click Nutanix philosophy, Kubernetes control plane and cluster upgrades happen automatically and with zero downtime.

Karbon Is the Central Platform to Plug Into

Nutanix deliberately chose an upstream, vanilla version of Kubernetes for Karbon. Being CNCF certified also means that applications run on any other open source-certified Kubernetes, promoting portability and avoiding lock-in to proprietary distributions.

Because for many organizations, it's not just Kubernetes. Getting started with Kubernetes often is the start of a journey that involves new tooling for logging, monitoring, distributed tracing, continuous integration, testing, deployment and delivery, various databases and key/value stores, pub/sub message queueing systems, and more.

Setting all these up before a developer can start using the ecosystem simply takes too much time and can overwhelm an IT department. That's why Karbon is part of a complete Cloud Native solution from Nutanix including storage (Volumes, Buckets and Files), database automation (Era), infrastructure management (Prism Pro), and enhanced automation (Calm) as well as networking solutions from Flannel and cluster-level logging using ElasticSearch, Fluentd, and Kibana.

Similarly, Karbon supports Helm. Helm is a package manager for Kubernetes that allows easy deployment for common open source and commercial software as a set of Kubernetes pods. Helm is considered a vital part of a well-functioning Kubernetes cluster.

Helm uses public and private repositories for managing the application templates, called Charts. These application packages (Charts) package up the configuration of containers, pods, and anything else for easy deployment on Kubernetes.

Helm is used to find and install popular software packages and manage the lifecycle of those applications. It's somewhat comparable to a Docker Registry, only Helm charts might contain multiple containers and Kubernetes-specific information, like pod specifications.

Helm includes a default repository for many software packages that developers use as part of their development toolkit, as well as for the applications they're building, making the life of application developers easier by eliminating the toil of finding, installing, and managing the lifecycle of these components.

Kubernetes FTW

In this chapter, we've taken a deeper look at the cloud-native journey. We've learned that ultimately, cloud native is deconstructing applications into atomically small components that are easy to develop, build, run, and maintain. Containers play a vital role in this transition, and Kubernetes is the platform to orchestrate and manage containers, no matter where they run.

But Kubernetes is notoriously difficult to install and maintain. Many different solutions exist to run and use Kubernetes that don't require a high level of expertise. Instead, these managed solutions turn Kubernetes into a commodity that's easy to consume without compromising any of the flexibility and extensibility.

Next Up

Containers and Kubernetes have led to a revolution in cloud computing. But where is that revolution taking us? In the next chapter, we'll discuss two of the most important directions: edge computing and the Internet of Things (IoT).

CHAPTER 8

Unleashing the Power of Edge/IoT

What is edge computing and IoT? The terms are everywhere, especially IoT. These technologies are omnipresent. They impact not only individuals using consumer-grade IoT devices, but also organizations. Unwittingly, citizens are also starting to be affected in their daily lives as more and more infrastructure projects and "smart cities" initiatives are powered by IoT and edge computing.

Defining Our Terms

IoT is an acronym for the *Internet of Things*, a way to designate any *thing* or device that would connect to the Internet, but isn't a computer or server. IoT devices have various form factors: They can be single-function devices, such as sensors, cameras, control systems, lighting systems, and so on, or larger devices in manufacturing plants, such as smart tooling and industrial robots.

Analyst firm Gartner predicted that up to 14.2 billion connected "things" (IoT devices) would be in use in 2019, growing to 25 billion in 2021.[5] The Gartner estimates are inclusive of consumer-grade IoT devices. When excluding PCs, smart TVs, and game consoles, analyst firm 451 Research predicted 7.9 billion devices in 2019, with a CAGR

[5] NetworkWorld - Gartner's top 10 IoT trends for 2019 and beyond - https://www.networkworld.com/article/3322517/a-critical-look-at-gartners-top-10-iot-trends.html

of 12%, leading to up to 14 billion in 2024[6]—a more conservative view, but closer to the reality of enterprise IoT.

IoT devices gather a lot of data in real time; this data, which is gathered for a variety of reasons, requires processing to make sense of it. Many IoT solutions send data primarily to the cloud for processing, but there's an increasing amount of cases where raw data must be processed locally. For example, in manufacturing facilities, IoT devices can be used for real-time quality checks, hence the importance of locally processing data.

This local processing of data, close to the source devices where data has been created, is called *edge computing*. Edge computing is necessary when latency becomes critical to data processing.

For example, intelligent, semi-autonomous systems such as smart cars are heavily dependent on uninterrupted streams of telemetry data coming from dozens or hundreds of sensors, all capturing data in real time. Data must be analyzed as fast as possible—any unnecessary latency must be eliminated to reduce the risk of a car accident and potential injuries or worse.

Once this data has been processed locally, the results of the processed data can be then shared with central systems on the cloud. This data can consist of insights from locally processed data, or a set of anomalies that need to be analyzed centrally. Machine learning can then be applied to those anomalies to enrich local processing capabilities.

5G and the promise of faster wireless networks may curb the issue of latency or bandwidth, but that doesn't mean edge computing will disappear. Quite the opposite—it can be expected that with the uninterrupted growth of devices, use cases, and datasets, 5G will

[6] 451 Research - 451 Research's Analysis of the Internet of Things Market Indicates that Total Connected Devices Will Reach 13.8 Billion by 2024 - https://451research.com/451-research-analysis-of-iot-market-indicates-total-connected-devices-will-reach-13-billion-by-2024

become a necessary speed/bandwidth upgrade. Indeed, 5G may cause edge sites to become data gravity centers. Applications will need to run at the edge (or become more distributed). Data will be primarily processed at the edge, and served to 5G-enabled devices also at the edge.

The Impact of Edge/IoT

Edge computing and IoT have already had a transformational impact on organizations. In fact, many have started calling this "the Fourth Industrial Revolution."[7] This explains why the term "Industry 4.0" is gaining traction in the world of Industrial IoT. But are IoT and edge computing the initiators of this transformation, or just enablers?

The core tenet of this transformation lies with today's data processing. This is not a new concept: For as long as computers have existed (and even before), data has been collected, processed, and stored.

Current technological capabilities have changed the game, artificial intelligence (AI) and machine learning (ML) enable organizations to interpret data in ways that were not possible before. Furthermore, today's computational power allows the use of certain technologies (such as image recognition) in real-time, and to derive actions from the data captured.

This confirms that edge computing and IoT are enablers: They allow massively distributed sensors to gather incredible amounts of data and process it at the edge (when real-time processing is needed) or in the cloud when processing isn't time-critical.

[7] https://www.forbes.com/sites/bernardmarr/2018/08/13/the-4th-industrial-revolution-is-here-are-you-ready/#60dc816c628b

Use Cases for Edge/IoT

The scope of applicability of edge computing and IoT is almost endless, and their use is increasing on a daily basis.

Some of the most popular use cases for IoT and edge computing include:

- **Manufacturing:** Condition monitoring; supply chain optimizations; predictive maintenance
- **Retail:** Cashier-less checkout systems; loss prevention; inventory management
- **Smart cities:** Adaptive traffic management; smart surveillance; smart metering/grids
- **Oil and gas:** Upstream—reduce drilling issues; midstream—quickly find pipeline issues; downstream—improve customer interactions
- **Health care:** Track patients, staff, and physical assets; non-intrusive equipment monitoring; reduce unplanned downtime of crucial equipment

Let's take a look at these in more detail.

Manufacturing and Supply Chain Operations

In the manufacturing and supply chain operations industry, IoT delivers the best bang for the buck when it comes to improving operational efficiency.

Optimization of supply chain processes is critical, because anything that means halting operations is dangerous to a company. The improvement of manufacturing and logistical processes requires the implementation of non-disruptive solutions that can coexist with existing systems.

IoT can deliver value by ensuring products are manufactured seamlessly, with the expected quality and within the expected time frame. IoT sensors can be used for quality checks, leading to the disposal of imperfect products before they're packaged and shipped. This not only avoids extra costs for returns and replacements, but also improves customer experience.

QA, of course, isn't the only area where IoT can help in the manufacturing and supply chain industries. Environmental monitoring is also a critical aspect of IoT. Sensors can be used to assess fluctuations in various conditions such as temperature and humidity, since those can impact the quality of production batches.

Sensors can also be embedded in critical production equipment to assess the wear and tear on production components, predict failures, and ensure proactive replacement of worn-out parts. The financial gains are massive—unplanned, costly production shutdowns can be avoided, which is a key advantage for lean manufacturing systems. Improved reliability increases production and efficiency, and allows manufacturing sites to move toward always-on production shifts.

Finally, IoT sensors can also be used to improve security and avoid issues by monitoring within facilities. Image recognition can be used to detect whether or not personnel are wearing mandatory safety equipment. Facial recognition can be implemented to detect unauthorized personnel accessing restricted areas of a production site. Improper placement of crates, pallets, or other devices can also be detected.

A great example of what can achieved in this vertical is showcased on Nutanix's IoT website.[8]

[8] https://www.nutanix.com/blog/one-click-iot-and-ai-based-smart-supply-chain-solutions

Retail

Retail is one of the industry verticals where consumers are the most likely to interact with—and benefit from—IoT technologies.

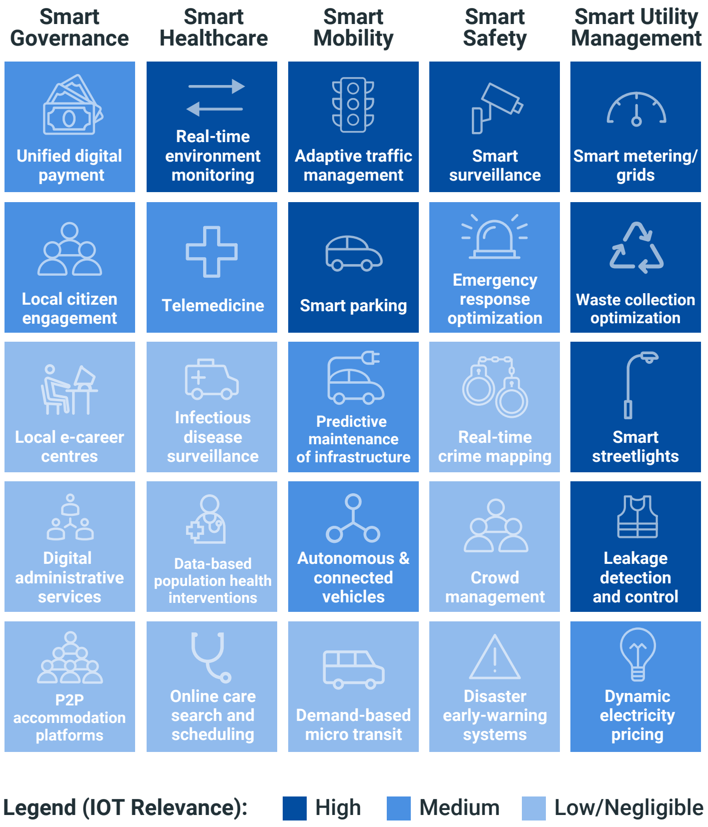
IoT technologies can be used in retail to analyze stock and consumption in real time. This can be achieved by combining smart shelves and RFID tag scanning. Both allow retailers to understand which products are being sold faster than others, to restock in a timely fashion, and to inform customers of prices without having to replace price labels. For organizations that combine digital sales with brick and mortar stores, this ensures further alignment on pricing across sales channels.

The consumer experience can also be greatly improved by the use of customized signage, augmented experience (via smart mirrors, for example—mirrors that double-up as displays and provide additional information on the product being tried), and self-checkout or automated-checkout capabilities. Dynamic customer interactions (signage reacting to the customer's facial expressions, for example) can further complement their experience.

Another aspect of IoT-enabled retail is the ability for those organizations to better understand customer behavior. This ranges from understanding how customers move around facilities, where they are more likely to stop, which areas are crowded, and which are empty. This opens a treasure trove of insights that can be used to improve traffic, eliminate bottlenecks, allow better placement of stores, and so on.

Smart Cities

Perhaps no other aspect of human lives may be touched by IoT as much as smart cities infrastructure projects. "Smart cities" is a term employed to identify cities that are implementing IoT projects, which can include quite a broad spectrum of possibilities.



Smart Governance	Smart Healthcare	Smart Mobility	Smart Safety	Smart Utility Management
▣ Unified digital payment	■ Real-time environment monitoring	■ Adaptive traffic management	■ Smart surveillance	■ Smart metering/ grids
▣ Local citizen engagement	▣ Telemedicine	■ Smart parking	▣ Emergency response optimization	■ Waste collection optimization
□ Local e-career centres	□ Infectious disease surveillance	▣ Predictive maintenance of infrastructure	□ Real-time crime mapping	■ Smart streetlights
□ Digital administrative services	□ Data-based population health interventions	▣ Autonomous & connected vehicles	□ Crowd management	■ Leakage detection and control
□ P2P accommodation platforms	□ Online care search and scheduling	□ Demand-based micro transit	□ Disaster early-warning systems	▣ Dynamic electricity pricing

Legend (IOT Relevance): ■ High ▣ Medium □ Low/Negligible

Figure 24: IoT mapping of use cases across smart cities.

Smart cities take a different approach to infrastructure management (see **Figure 24**). Where regular cities work in silos with loosely related initiatives, objectives and separate budgets, smart cities break down the silos. Smart cities address each of those challenges as an ecosystem where each area of focus has an impact on other areas.

This delivers better outcomes for citizens, improves quality of live, and fuels investments and further innovation.

Smart city project managers may decide to implement IoT to address one or more of the following areas of interest:

- **Unified digital payments.** Implement smart payment mechanisms into IoT systems that allow for seamless identification and payment
- **Real-time environment monitoring.** IoT sensors can monitor air and water quality in real time and provide recommendations to authorities and citizens. Air quality data can be further used by authorities, for example, to implement decisions about speed limits, or to restrict vehicle access to certain city areas
- **Adaptative traffic management.** Smart cities can use video analytics and traffic signal priority (TSP) tools to monitor traffic, improve incident reporting, review incident footage and gather valuable data to further improve traffic
- **Smart surveillance.** IoT devices can be used to monitor crowds and inform authorities in case of threshold violations, while ensuring access to security systems is restricted and security hardened
- **Smart metering/grids.** Energy consumption can be monitored through IoT-enabled electric meters and power grids

Beyond smart cities, IoT can be used to monitor structural elements of major infrastructure components in real time, such as bridges, tunnels, roads, and railroad tracks.

Oil and Gas

The oil and gas industry can benefit from IoT and edge computing at different locations, as well as different stages of production and delivery. They include:

Upstream. Edge computing allows onsite analysis of massive amounts of real-time data coming from IoT sensors. This data can be used to increase productivity and efficiency, while also reducing drilling issues. IoT sensors can be used to anticipate wear and failures in key drilling equipment, which is particularly vital at offshore sites where the logistics of replacements can be complicated by distance and weather conditions. Also, production levels can be adapted based on price fluctuations, and extraction can be optimized while reducing overall costs.

Midstream. The use of IoT and edge computing allows organizations to monitor their production and distribution networks in real time. Weak spots in the distribution infrastructure can be detected better and faster, helping with proactive replacement of weakened or damaged portions of the distribution network before leaks happen. Existing data from SCADA systems can be collected and aggregated to provide insights.

Downstream. Preference and purchasing data can be collected across gas station to improve customer interactions and accelerate commercial initiatives.

Healthcare

There are multiple use cases for IoT devices in healthcare. Areas such as tracking and managing medicine supplies and assets; monitoring critical medical equipment; monitoring patient room equipment; and analyzing ambient video streams are among the areas delivering the most value.

Tracking patients, staff, and medical assets allows personnel to spend more time caring for patients instead of looking for medicines and supplies.

Real-time predictive maintenance of crucial medical equipment ensures that the most precious assets are always available, especially during critical moments when patients need them the most. This allows hospitals to keep those crucial assets always online and operational.

What About Smart Buildings?

While not exactly an industry vertical, it's worth mentioning smart buildings, because the technologies implemented in smart buildings leverage IoT and can be used across many verticals and locations.

In this context, IoT devices are used to understand and optimize environmental factors such as heat regulation and lighting, which ultimately affects power consumption.

This can be a combination of electric curtains, air conditioning, and smart lights, where each of the elements can be tuned to achieve the desired results at different times of the day.

A combination of sensors and cameras can determine if one or more work areas are occupied by people or not, and intelligently throttle air-conditioning and lighting in unused areas.

While the gains may seem minimal on the scale of a single home, IoT-enabled smart buildings can lead to comprehensive savings when implemented in commercial and office spaces.

Patients' health and sometimes lives depend on the continuous operation of an array of equipment deployed in their rooms, as well as environmental criteria, such as room temperature. IoT allows for non-intrusive monitoring of the equipment, as well as detection of issues and movements through bed sensors, and so on.

Challenges of Edge/IoT

Although IoT and edge computing are the fuel for Industry 4.0, there are challenges that may hamper adoption, or slow it down:

- Bandwidth constraints between edge and cloud
- Lack of scalability
- Disconnected operations
- Compliance and privacy issues

Bandwidth and Latency Constraints

IoT devices can generate overwhelming amounts of data. Not all the data can be sent to cloud systems for processing, and not all of it is meant to be sent to the cloud. Certain types of IoT devices feed decision systems in real time, requiring local processing at the edge for latency reasons.

Latency isn't the only culprit: bandwidth can also be problematic. Organizations operating with many remote sites do not always have the luxury of unlimited bandwidth, especially in rural areas. Limited bandwidth isn't available for edge computing traffic alone—it needs to be shared with other applications and processes which are critical for business operations. If those existing apps stop working, your business might, too. This is unacceptable in a production environment, making it essential that IoT is introduced in the least intrusive way possible.

Some think that 5G is the answer. But technical constraints, such as the availability of 5G frequencies, the ability to deploy 5G in time, and the exponential growth of IoT devices, means that 5G networks may not solve throughput problems.

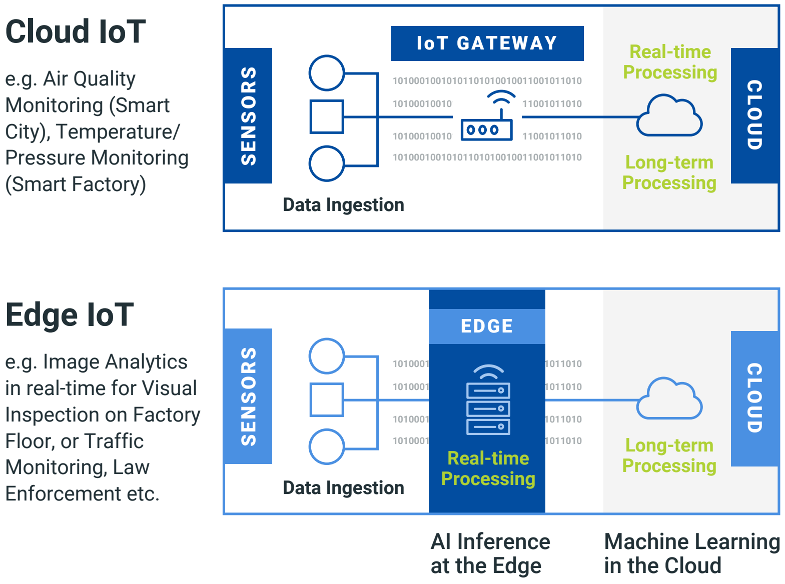

Cloud IoT
e.g. Air Quality Monitoring (Smart City), Temperature/ Pressure Monitoring (Smart Factory)
SENSORS
IoT GATEWAY
Data Ingestion
Real-time Processing
Long-term Processing
CLOUD
Edge IoT
e.g. Image Analytics in real-time for Visual Inspection on Factory Floor, or Traffic Monitoring, Law Enforcement etc.
SENSORS
EDGE
Data Ingestion
Real-time Processing
Long-term Processing
CLOUD
AI Inference at the Edge
Machine Learning in the Cloud

Figure 25: Architectural differences between cloud IoT and edge IoT.

Lack of Scalability

Managing a few IoT devices at a single location isn't a technical hurdle—but as the count of locations and sensors increases, using a custom-built IoT management solution puts a strain on scalability and resources diverted to develop and maintain the said management platform.

The challenge of scalability has multiple facets. First, IoT will permeate through many verticals and use cases. Organizations will sooner or later be faced with the challenge of managing a multitude of IoT devices of all sorts, at multiple locations, for different use cases, and for different target applications.

Second, bandwidth isn't an infinite resource. In cloud-centric IoT models, raw data from the IoT sensors is directly sent to the cloud through an IoT gateway and processed there (see **Figure 25**). Systems that leverage this data are then provided the necessary insights.

While cloud providers usually don't charge for ingress traffic, there is a price to pay for egress traffic. At scale, this has the potential to put a heavy toll on cloud budgets, unless dedicated lines are used.

Obviously, cloud-based processing instances need to run 24x7. The cost factor of running always-on cloud-based workloads is usually two to three times more expensive than running a dedicated solution. While cloud allows for scaling workloads up and down, this type of scalability—where systems need to run non-stop—comes at a price.

Third, manual configuration and management of connections between source systems (IoT devices) and target systems (applications) cannot scale at all. Too much time is wasted identifying protocols, building data transformation libraries, and configuring connections manually. The same goes for deployment of common applications.

Finally, edge computing environments must also be managed. Hardware must be procured, configured, supported and regularly upgraded. The lack of skilled support personnel on remote locations (especially when operating at scale) is another constraint.

Disconnected Operations

When operating at the Edge, connectivity between edge and cloud may not always be stable. This is particularly true for operations taking place in remote locations, where no land lines exist and operations have to be carried over cellular, radio, or satellite links. These connection types are unstable by nature, and loss of connectivity needs to be factored into any data communication design.

Take remote equipment monitoring, for instance. In these environments, edge computing is preferred over cloud-based operations, since real-time analysis of how equipment like oil rig drills is operating is absolutely essential. These drills can cost in the hundreds of millions of dollars, and waiting for data to be shuttled back and forth between the cloud is a non-starter.

When data is collected and analyzed locally, as in edge scenarios, personnel is informed in real time and empowered to take appropriate actions in a timely fashion. Over an unstable link to the cloud, the inability to send vital data at the right time could cause alerts to be missed and lead to equipment failures or shutdowns, which can cause widespread disruption and massive financial impact.

Losing Control of Your Data

In typical cloud IoT models (those leveraging an IoT gateway), sensors send their data to the IoT gateway, which then sends the data to a public cloud provider. These models can impose a target public cloud where the data must be processed, without giving organizations the choice of where the data should be sent.

This raises a variety of concerns. Organizations must abide by country-specific laws, and these may impose stringent data sovereignty requirements, forcing data to reside in specific geographical boundaries. These requirements may cause organizations to look at alternatives to cloud IoT models—specifically at solutions that can, for instance, send data to multiple public clouds, kept locally, or directed to a private cloud. Locking companies into a single public cloud can impair or destroy the ability to do edge computing.

Still other businesses may have internal requirements about cloud/data mobility, where the organizations want to have a final say about where their data lives. In other words, control should be kept in *your* hands.

4 Considerations for Designing Successful Edge Solutions

With use cases and challenges in mind, how can organizations prepare themselves for a successful design and implementation of IoT and edge computing? Here are four crucial aspects to consider.

1. **Do more with IoT data.** Start with making sense of massive amounts of data. IoT devices are a treasure trove of data, waiting to be mined, analyzed and turned into useful business intelligence. Looking at how connected devices work and learning from these is a great way to get started.

2. **Choose a flexible platform.** Selecting an agnostic platform that allows workloads to run on any cloud is recommended. The platform should support end-to-end security, and should also easily integrate into existing environments.

3. **Focus on business logic.** Less time should be spent on coding (in this instance, a low-code platform can help). Existing continuous integration/continuous delivery (CI/CD) pipelines should be leveraged whenever possible so that the focus on business logic isn't lost. The use of a pluggable architecture reduces complexities and allows for seamless integration between components.

4. **Design AI-based applications.** Analyzing downtime and failure patterns is a good way to start building models to predict downtime before it happens. Get smarter by extending AI analysis to other fields or sensor data. Learn across applications, to identify techniques or metrics that could be reused in other systems or applications.

Doing all this requires proper management, which is really the key. Management capabilities should provide sufficient abstraction, perform at scale, and allow organizations to connect data sources (IoT devices) to data targets (applications) without having to build complex transformation functions to convert raw IoT data into data that applications can process.

Such a management platform should allow applications to be deployed easily in multiple locations, without having to go through the hassle of manual deployments. Most importantly, it should allow developers to focus on the development of IoT data processing and interpretation of code, rather than spending precious time manually configuring every single step to move data around.

Managing IoT and Edge Computing with Nutanix Xi IoT

Nutanix has such a platform, called “Xi IoT”. Xi IoT is the Nutanix approach and vision for deploying and managing edge/IoT systems. Xi IoT bridges the gap between multiple clouds and the edge into a single data processing fabric, enabling data to be processed where it makes the most sense.

Xi IoT is an AI-driven, edge-based real-time data processing platform that allows organizations to more fully utilize the power of IoT and edge computing. It consists of the following components:

- An edge PaaS platform that runs as a VM or bare metal
- Cloud-based SaaS lifecycle management
- Connectors for public and private clouds

The edge PaaS is where IoT sensor data is gathered and processed. This platform can be delivered in multiple form factors, and is particularly well adapted to run on HCI environments as a virtual machine (see the callout “Hyperconverged Infrastructure Integration” later in this chapter). This emphasizes the increasingly central role of HCI architectures in enterprises.

Faithful to the Nutanix principle of platform agnosticism, Xi IoT can also be deployed as a bare-metal solution. This is particularly useful for use cases requiring ruggedized implementations, or for locations where the footprint is too small to leverage an existing HCI solution.

The cloud-based lifecycle management allows operators and developers to push their applications to thousands of edge sites via a single dashboard. With a simple and easy-to-use interface, operators and developers can perform their activities without the hurdles of manual work.

Xi IoT includes a Kubernetes-orchestrated environment that allows containers to be deployed and executed at the edge. One of the key components of Xi IoT is the ability to simply configure data pipelines through the dashboard. Data sources can be selected and connected to data targets (applications or functions), and users have the ability to specify if the source data format needs to be transformed before it's provided to the target. Xi IoT supports multiple transformation libraries, and allows developers to write their own transformation code if necessary.

Hyperconverged Infrastructure Integration

Nutanix is best known for its hyperconverged infrastructure (HCI) platform. Those using it will be happy to know that Xi IoT integrates superbly with Nutanix HCI. Organizations that already run HCI clusters at edge locations can simply enable Xi IoT and no longer need to worry about building a cloud-based IoT processing platform from the ground up.

Enterprise-grade, scalable and extensible HCI architectures such as Nutanix Enterprise Cloud allow organizations to look beyond traditional virtualization workloads by delivering bleeding-edge capabilities such as edge solutions. This allows organizations to keep their TCO under control while increasing their ROI.

(It's important to note as well that Xi IoT is platform agnostic. It will run on virtualization platforms [HCI or non-HCI, regardless of the vendor], via the Xi Edge VM, or directly on bare metal.)

Xi IoT support multiple clouds, including private clouds, and establishes secure connections between sensors and its own infrastructure components, whether at the edge or on the public clouds such as Amazon Web Services, Microsoft Azure, and Google Cloud Platform. In other words, it works at planet scale.

IoT promises to deliver endless possibilities and can be a tremendous factor for growth. But challenges need to be taken into consideration when designing a solution to avoid common pitfalls. The approach taken by Nutanix—analyzing data at the edge, machine learning in the cloud—offers the best of both worlds.

Xi IoT delivers a solution that works out of the box, is simple to manage, and scales across thousands of sites. With Xi IoT, developers no longer need to write their own custom code and solution to support IoT projects. It just takes developers a few clicks to deploy AI and IoT applications to the edge via the SaaS lifecycle management.

Getting Started with Xi IoT

As you can see, IoT is complex and can seem intimidating to implement and manage—fortunately, help is available, in the form of Xi IoT.

Getting started with Xi IoT is easier than it seems. Nutanix's website at nutanix.com offers a lot of valuable content to get started with Xi IoT. Notably, Nutanix offers a trial of Xi IoT at nutanix.com/iot-trial.

To help with the trial and kickstart IoT initiatives, Nutanix has created several Xi IoT-related resources. These quick start guides, getting started pages, and application deployment guides are available at nutanix.handsonworkshops.com/workshops.

The End of the Beginning

If you’ve made it this far, congratulations! You’ve reached the end of this Gorilla Guide. Through its pages, you’ve learned a lot about cloud computing: how we got here, the current state of the cloud, and where it’s headed in the future.

You’ve also gained lots of knowledge about practical aspects of using the cloud to make your operations more efficient by eliminating silos; how to stay secure in the cloud; using containers in a cloud-native environment; and much more. This has been a long journey, but hopefully you’ve found it worth your time.

The cloud has a lot to offer, but also many potential pitfalls you need to be aware of (jungles are like that). We wish you luck on your continued efforts in the cloud. It’s truly the dawn of a new era, and is an exciting time to be in IT.

ABOUT NUTANIX

NUTANIX™

Nutanix makes infrastructure invisible, elevating IT to focus on the applications and services that power their business. The Nutanix Enterprise Cloud OS delivers the agility, pay-as-you-grow economics and operational simplicity of the public cloud, without sacrificing the predictability, security and control of on-premises infrastructure. Nutanix solutions leverage web-scale engineering and consumer-grade design to natively converge compute, virtualization and storage into a resilient, software-defined solution that delivers any application at any scale.

For more information, visit www.nutanix.com

ABOUT ACTUALTECH MEDIA

ActualTech Media

ActualTech Media is a B2B tech marketing company that connects enterprise IT vendors with IT buyers through innovative lead generation programs and compelling custom content services.

ActualTech Media's team speaks to the enterprise IT audience because we've been the enterprise IT audience.

Our leadership team is stacked with former CIOs, IT managers, architects, subject natter experts and marketing professionals that help our clients spend less time explaining what their technology does and more time creating strategies that drive results.

For more information, visit www.actualtechmedia.com