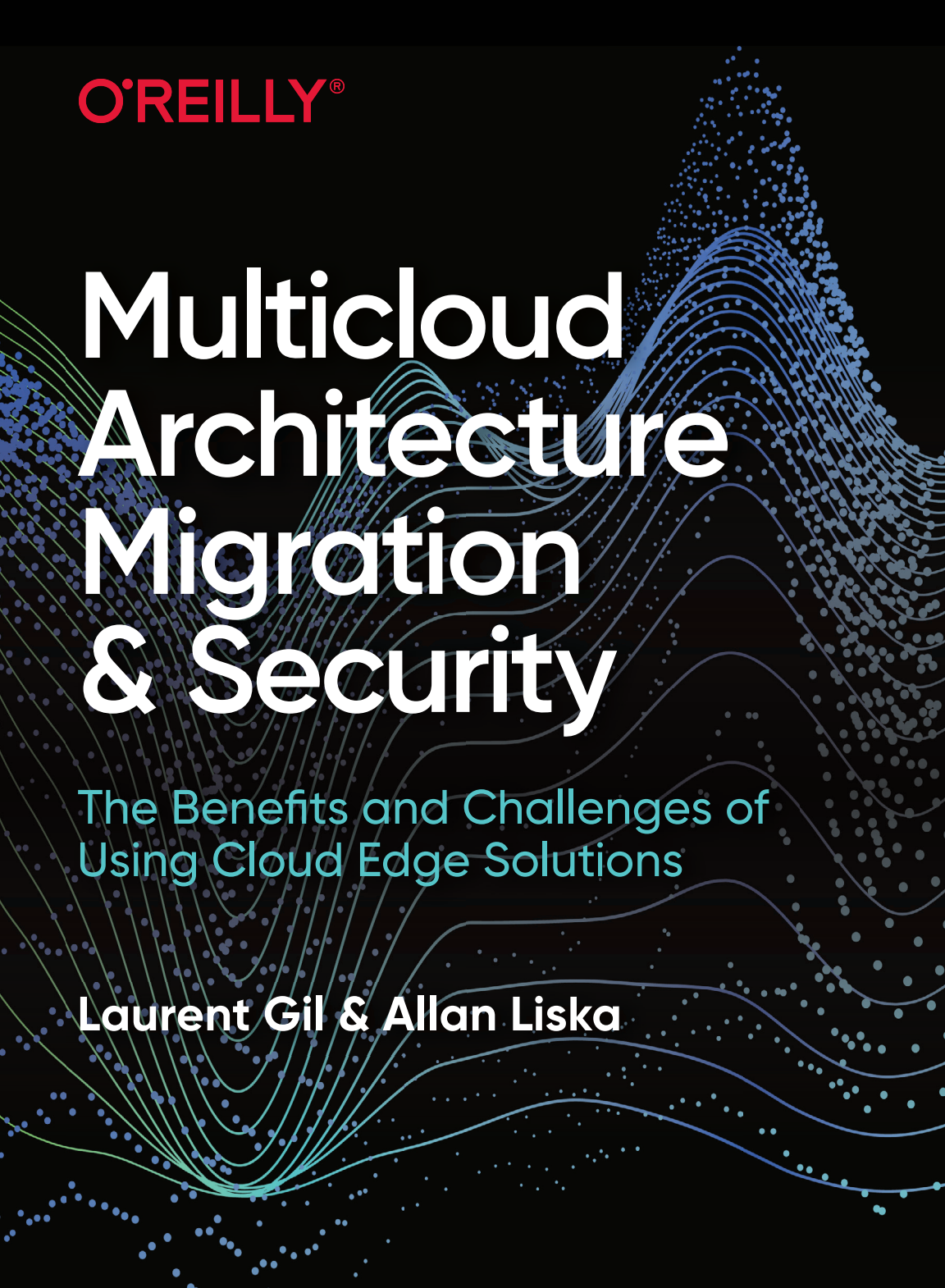
O'REILLY®
Multicloud Architecture Migration & Security
The Benefits and Challenges of Using Cloud Edge Solutions
Laurent Gil & Allan Liska
REPORT

Multicloud Architecture Migration and Security

The Benefits and Challenges of Using Cloud Edge Solutions

Laurent Gil and Allan Liska

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®

Multicloud Architecture Migration and Security
by Laurent Gil and Allan Liska

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*http://oreilly.com*). For more information, please contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Acquisitions Editor: Nikki McDonald
Development Editor: Virginia Wilson
Production Editor: Christopher Faucher
Copyeditor: Octal Publishing, LLC
Proofreader: Matthew Burgoyne
Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Rebecca Demarest

May 2019: First Edition

Revision History for the First Edition
2019-05-15: First Release
2019-11-05: Second Release

978-1-492-05039-1

[LSI]

Table of Contents

CHAPTER 1
Why Multicloud Architecture?

Organizations are increasingly relying on the cloud for the deployment and management of application stacks. In fact, most organizations support many cloud-based applications. According to a 2018 IDG study, 73% of organizations have at least one application hosted in the cloud. Services that used to be managed entirely on-premises, such as the sales process, timekeeping, and human-resources management, now reside entirely in the cloud. At the same time, organizations are moving their own homegrown applications to the cloud. Hosting in the cloud saves on initial infrastructure costs, planned and unexpected maintenance costs, and operational costs. But there are challenges and complexities associated with adopting a multicloud architecture that these organizations should consider.

Organizations need to weigh many factors when selecting a cloud service provider. One of the biggest challenges they face when migrating to the cloud is choosing the appropriate platform. Sometimes, the choice is made for you. For example, some outsourced services work only with certain cloud providers. In other cases, organizations have become locked into using a specific cloud vendor when applications were first moved to the cloud.

Another challenge associated with running a multicloud architecture centers on security. Organizations need to apply a consistent security policy across all cloud platforms that is consistent with the security policy of internal systems. In the end, it doesn't matter whether a system is onsite in the core of your network; sitting in an

Amazon, Microsoft, or Google datacenter; or residing at a specialized hosting provider. The security team is responsible for the system's integrity regardless of where it resides.

These challenges are certainly daunting, but they are not insurmountable. The goal of this book is to help managers and leaders understand the challenges of migrating to and securing a multicloud infrastructure. In this book, you learn about ways to manage and orchestrate multicloud environments, including some "edge" technologies that are designed to secure and protect your environment.

What Is a Multicloud Architecture?

Cloud architecture is generally broken down into two different types of deployments: multicloud and hybrid cloud. *Multicloud* architectures, in which multiple services are hosted by different cloud providers, are the most common deployment. In one example of a multicloud deployment, Domain Name System (DNS) services are hosted by one cloud provider, web applications are hosted by another, a helpdesk ticketing system is hosted by a third provider, and so on. This type of multicloud deployment involves multiple cloud providers, each with a standalone function operating relatively interdependently across multiple providers.

Although multicloud deployments might seem straightforward, keeping track of the different capabilities of each vendor and ensuring that updates and policies are correctly applied across all cloud partners can be very challenging. In a multicloud architecture, complexity stems from the process of managing the environment in the context of various shared responsibility models and geographic considerations. This type of architecture also means that each provider is a single point of failure for your organization.

A *hybrid cloud* is generally considered a combination of a public and private cloud, but this definition has been expanded to mean on-premises and cloud in some contexts. According to the National Institute of Standards and Technology (NIST), "the hybrid cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds)."

Hybrid cloud involves deploying a single system to multiple clouds. For example, rather than deploy a web application to a single cloud provider, an organization deploys it to multiple providers. An organization might decide to use Amazon Web Services (AWS) as the primary provider with Microsoft Azure as a backup provider. Alternatively, you might want both AWS and Azure to be live and alternate incoming requests between the two providers.

We can carry out a hybrid cloud deployment in multiple ways:

Public-private
: Part of the application runs on a public cloud provider, whereas backend systems run in a separate private cloud environment.

Active-passive
: A service is deployed primarily to a single cloud provider, but a backup of that service is deployed to a second provider. The second provider is activated only if the first provider experiences a failure of some sort.

Active-active
: The service or application is deployed to two or more cloud providers, all of which are live and serving content simultaneously based on geolocation information, load balancing algorithms, or a simple round-robin request.

Hybrid cloud solutions are much more complex to deploy and even more complex than multicloud solutions to manage on the backend. However, the uptime benefits and the lack of a single point of failure more than offset the costs incurred in terms of complexity. These hybrid cloud solutions also take more upfront planning prior to the initial deployment so that teams can understand how the data will be served from each provider and what the required investment will be to present a homogeneous experience to end users across all platforms.

Benefits of a Multicloud Architecture

There are a number of benefits to a multicloud architecture. One benefit is related to cost. Maintaining servers and infrastructure on-premises requires large and ongoing capital investment, whereas deploying in the cloud significantly reduces the costs. Beyond capital expenses, a multicloud strategy enables organizations to invest wisely, on a per-project basis, in the cloud provider best suited for a

particular service. AWS might make the most sense for BaaS, whereas Google Cloud Service might make the most sense for deploying a human-resources application. Maintaining multiple cloud providers means that you will be able to choose the best one for each purpose.

Beyond cost savings, there are other benefits to a multicloud platform, including the following:

- Freedom from cloud vendor lock-in
- Infrastructure security and resilience

At first glance, there appears to be a lot of similarities between cloud providers in terms of features and pricing, but a closer inspection can reveal many differences. If your organization is locked into a single provider, you won't get a chance to explore those differences. For example, two providers can offer a similar service, but one includes it as part of the base pricing; the other offers it as an add-on that can result in unpredictable monthly costs. Some providers offer capabilities in only certain regions or might not even offer a presence in a region that is important to your organization. Preventing provider lock-in means having the freedom to deploy applications where you need them, in the most optimal manner, whether that is a technical, performance, or pricing optimization.

A multicloud solution also provides better infrastructure resilience. Today's organizations demand high availability (HA) for the applications and services on which their customers and employees depend. The slightest interruption in availability can cause cascading disruptions and associated revenue losses. The idea of 99%—or even the much-talked-about five nines (99.999%) uptime—is an antiquated concept. Customers and employees need to know that they can reach your web applications 100% of the time, no matter what else is happening on the internet. Multicloud architectures can deliver the resilience and uptime that organizations require.

Case Study

In late February 2017, a series of cascading failures caused one of the largest cloud providers to be unavailable for four hours. Four hours might not seem like a very long time, but that was four hours that Netflix, Spotify, Pinterest, and hundreds of other sites were not

available to anyone. The outage did not just affect consumers. Many functions of Slack, a popular tool used by organizations for inter-office communication, weren't available during that time, as well, leaving many corporations unable to communicate internally.

The problem was restored as quickly as the cloud provider's engineers were able to fix it, but the affected companies fielded complaints from angry customers for days after the outage occurred. Outages like these can potentially cause millions of dollars in lost revenue.

This illustrates the inherent dangers in relying on a single cloud provider for hosting your web application. Even the most reliable providers will occasionally experience outages and their outages impact your customers. A diversified cloud presence allows an organization to better weather these types of outages and it means their customers will see little to no impact.

Modes of Distributing Workloads Across Multiple Cloud Providers

Each type of architecture deployment has its pros and cons. Often, the type of architecture used depends on the kind of application being deployed, but it can also depend on budgetary concerns and the maturity of the organization making the deployment.

None of the modes discussed in the subsections that follow are inherently better than the others. The best choice is largely dependent on an organization's specific business requirements, internal processes, and standards.

Active-Active Versus Active-Passive

Active-active and active-passive architectures are all about ensuring HA. They're designed to ensure that customers or employees in any part of the world can access web applications quickly, with very little latency. These architectures also ensure that if one cloud provider experiences a failure, end users will not notice, because the failover will be seamless.

Figure 1-1 provides a simplified example of what an active-active architecture looks like. A web application is deployed across multiple cloud providers, each with a load-balancing service enabled in

front of the application. The load balancers ensure that the service behind it is up and running and not overloaded with traffic. The load balancer then selects the best server and directs the incoming request to it. If one cloud provider becomes unavailable, the other provider is able to shoulder the traffic load.

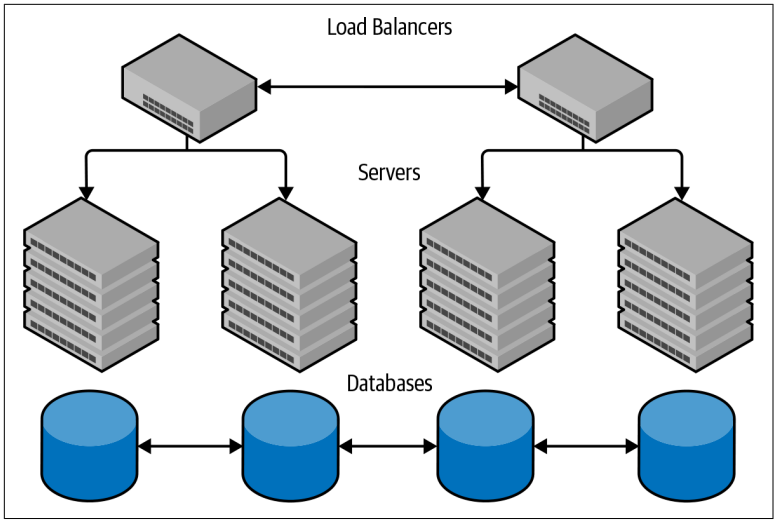

Figure 1-1. Active-active multicloud architecture

There are other ways to enable this kind of load balancing. Some organizations use advanced DNS services to replicate this kind of architecture.

Figure 1-2 shows a similar active-passive multicloud architecture. This solution has the same high-level architecture. The difference is that the systems in the second cloud provider's datacenter remain dormant unless there is a failure of some sort at the first cloud provider. That failure could be physical in nature, such as server crashes or interruptions in network services. But it could also be a threshold failure in which, for example, the site experiences a significant spike in traffic because of an advertising promotion or it begins to trend on social media.

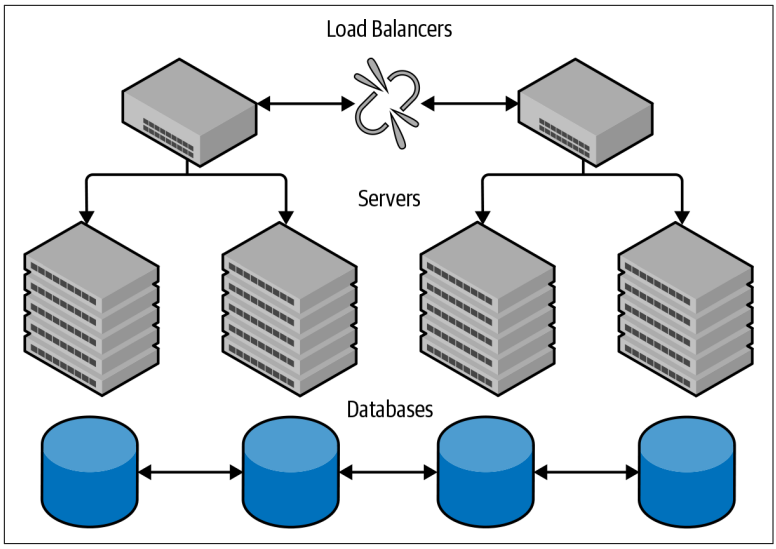

Figure 1-2. Active-passive multicloud architecture

In an active-passive architecture, the backup site is "hot," the infrastructure is online and listening, but no traffic is directed to that cloud provider unless the failure occurs. Active-passive architectures are more cost-effective than active-active multicloud environments, but there are a number of technical complexities that need to be taken into consideration. First and foremost, the infrastructure must be tested regularly to ensure seamless failover in the event of a failure at the primary site. The infrastructure of the passive site must be treated as if it were live. You can take advantage of the fact that you have a fully functional replication of your website or web application by testing new patches and configurations on the passive site before pushing them to the live site.

Public-Private Multicloud Architecture

Up to this point, we have been discussing public cloud infrastructure, but cloud infrastructure can also be private. This can be a non-public instance set up in a traditional cloud provider, such as Amazon or Google, or it can be a cloud instance set up in your own datacenter or colocation facility.

Private cloud infrastructure is usually designed to host highly-sensitive organizational information. A private cloud instance might hold customer data, employee information, or other sensitive

company documents. Access to this type of infrastructure is usually restricted and requires virtual private network (VPN) credentials or, at a minimum, firewall and proxy rules that prevent outsiders from accessing the data contained in the private cloud instance.

However, there is a third configuration option in the case of private cloud architectures. That is the hybrid public-private cloud. Figure 1-3 presents this design in which some information from the private cloud needs to be accessible to a larger audience or even publicly accessible. In these cases, the architecture is split between servers in a public cloud environment and servers in a private cloud environment with a firewall or other security device restricting access to only that information that needs to be shared with the public cloud space.

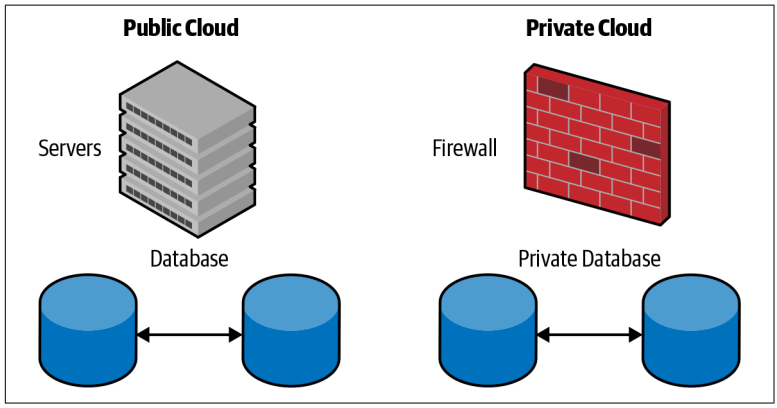

Figure 1-3. A public-private hybrid multicloud infrastructure

This can be a tricky configuration because there is the potential to expose sensitive data. For example, there have been multiple Amazon Simple Storage Service (Amazon S3) bucket data exposures within the AWS cloud. This doesn't mean that you shouldn't deploy hybrid architectures like this. In fact, they are becoming increasingly common as more data becomes available to the public. What it does mean is that security needs to be considered as part of these deployments from the start of the planning, something that we discuss further in Chapter 3.

Trade-Offs

Although the benefits of multicloud architecture are myriad and discussed in detail throughout this book, it is important to head into a multicloud strategy with eyes open. There are a number of trade-offs involved in migrating to this type of architecture, and some organizations might not be prepared for a multicloud world. There are three potential pitfalls that organizations need to be mindful of when considering a multicloud strategy:

- Increased networking complexity
- Staying abreast of changing public cloud product offerings
- Development agility

Let's explore these trade-offs in detail so that your team will be armed with the information it will need to make the best decision.

Increased Networking Complexity

Multicloud deployments, by their very nature, are complex. This complexity doesn't just apply to the networking infrastructure you employ or how you manage servers and databases. There are vagaries of each cloud provider (such as interfaces, controls, and limitations of the platform) that you need to consider as you plan your strategy.

But the primary difficulty is in the networking stack. Anyone who has worked as a network or security engineer for any length of time has had that "Oh, s**t" moment in which part of the network is accidentally knocked offline. Just imagine a mistake that takes down a worldwide infrastructure deployment across multiple datacenters. For example, in November 2018, a Border Gateway Protocol (BGP) misconfiguration at a Nigerian ISP resulted in many of Google's services being unavailable to much of the world for more than an hour.

Correctly designing the network infrastructure in a multicloud environment is critical. Proper multicloud design involves DNS management, load-balancer configuration, server redundancy, database replication, and firewall rules that allow this traffic to flow freely across multiple providers while not allowing unauthorized visitors access to sensitive data. Of course, this design is not static; it is a

living document that must be constantly updated and shared with all stakeholders, including your cloud providers, every time a change is made. If you do not keep the documentation current and regularly share it, one team could make a change that interferes with another team's change and crash the entire system.

Again, the benefit to doing this correctly is an unprecedented level of resiliency and uptime that would be almost impossible to replicate using your own datacenters. Even if you could, the cost of doing so would be daunting.

Staying Abreast of Changing Public Cloud Product Offerings

Many organizations are interested in deploying applications in a multicloud architecture because those organizations are maturing and eager to adopt newly available technologies. As organizations mature, they're able to take on more complex projects, and their systems become more intricate.

As your organization is maturing, so is your cloud provider. This means that there are often new services, pricing structures, and capabilities being offered by cloud providers and their competition. It is important to understand what new features are available, especially if these new features can reduce costs. After all, cost reduction is the primary driver for moving services and applications to the cloud.

It is a good idea to schedule quarterly meetings with your cloud provider, usually with a technical account manager (TAM). The TAM will review your account and go over usage with your team. TAMs also keep you abreast of any new services that might be relevant to the organization. You might find during these meetings that it makes sense to switch some services from one provider to another to save money. Many cloud providers offer their own cost analysis tools that you can use to look for cost savings or get a projection of costs before you deploy a new service.

How do you find out about new services being offered by cloud providers that are not part of your current deployment? Most cloud providers give you the option to review services on their website, and of course, a sales team is always happy to call on you. But those

options take time, and it's still possible to miss out on a new service that might be a good fit for your organization.

Another option is to rely on updates from your own team. Cloud providers often offer certification programs for your engineering team. This gives your staff added insight into the services available through the cloud provider and helps them stay abreast of the latest offerings.

Many organizations turn to Cloud Service Brokers (CSBs) such as Bluewolf, Jamcracker, or Cloud Foundry to help identify cloud providers that deliver the right services at the right price. CSBs have experience with a wide range of cloud providers and have worked with other customers, often with needs similar to yours. This extensive experience empowers them to help organizations find cloud services that you might not have previously considered. CSBs can make the process significantly less painful, and they can help organizations get up and running in a multicloud environment quickly.

Development Agility

The last trade-off in considering the move to a multicloud architecture is development agility. A multicloud architecture requires the ability to create code that can run on multiple cloud providers' platforms, ideally with no changes to the code from one provider to the next. Development standards within your organization help take care of this, so even if the code base is a mess, each provider can run that mess without different code bases.

Moving an application to a multicloud architecture might require your team to revisit application code. The code will need to be cleaned up and tested across your cloud providers to ensure that pushing out updates won't create problems that will make one of the cloud instances unavailable. But it is more than just editing your code; there are other tasks that you need to complete, such as making sure your provider has the libraries, language compilers/runtime, and server versions available in their repositories for whatever containers you are running. A good example of that is the Java Development Kit (JDK)—you might be at the point at which you can install only an open-JDK version. Another example is that you could have a version of Apache Tomcat that runs a significantly newer servlet specification.

Another problem could be that you are stuck with an old system that is part of your stack that requires something old that you cannot get from a cloud provider, like an older, unsupported relational database. This can cause expenses related to the time it takes developers to plan that upgrade and data migration, and unexpected license costs. None of this would be code cleanup, but it is necessary to factor in the time and cost when making the move.

Development agility is about more than just clean code that can run across multiple cloud environments. It also means adapting to the reality of more frequent updates to the code base. It means updating your code multiple times a month or week instead of a couple of times a year. The development team also needs to be able to rapidly write and test new security patches and deploy them quickly without taking the application offline. Truthfully, this is already a development reality, and being in the cloud to do it is not a requirement for this type of development process. Agile development is how a lot of businesses operate, irrespective of where the application resides.

Given that the Agile development process is the new norm, what changes for most organizations when migrating to a multicloud infrastructure is neither the code nor the development cycle. It is the *DevOps* process and the Continuous Integration (CI) tools that run DevOps. You now have to package, deploy, and run automated tests on multiple cloud providers. And maybe one app is on just one cloud provider, but another application that needs the redundancy is on multiple cloud providers. You are going to want your development cycle to automatically provision and deploy to the appropriate place. And your local development environment likely isn't a cloud provider (although it could be).

Evaluation

If you have read this far, you might be asking, how do I determine whether my project and organization are ready for multicloud? It is not hyperbole, or even controversial, at this point to state the future of most IT projects will be in the cloud. The benefits of cost savings, resiliency, and new feature enhancements that organizations realize by moving to the cloud are too great to ignore. Moving to the cloud allows an organization to focus on its core competencies instead of managing a datacenter.

But there is a difference between moving to the cloud and moving to a multicloud or hybrid cloud environment. Let's take a closer look at how to evaluate which strategy is right for your organization.

Weighing the Pros and Cons

As with any business decision, it is important to look at the big picture, ignore the hype, and weigh the pros and cons of moving to a multicloud environment.

You can begin by examining where the application sits now. Is it hosted in a datacenter managed by your organization, colocated, or is it already hosted in a cloud provider's datacenter? Making the jump from a locally hosted application to one that is hosted in multiple datacenters around the world might be too much of a jump (not always, but for many organizations it is). Most organizations start by moving their application to a single cloud provider first. This allows the organization to better understand the experience and the challenges associated with cloud migration. For example, the organization might find that its application code is not as portable as it once thought. After you have experience working in a cloud environment, the jump to a multicloud architecture is easier. (Note: We didn't say that it will be *easy*, just *easier*.)

It's also important to determine how much control your team is willing to surrender to the cloud providers. Cloud providers offer a range of services from the most basic, in which the cloud provider simply offers hardware and a range of IP addresses, all the way up to a fully managed service. The service you choose is also a budgetary matter. The more cloud provider services you take advantage of, the greater the costs.

And of course, all of this has to synchronize across multiple providers. For example, in a fully managed cloud environment, it's important to know how quickly your provider installs patches. You don't want to have half of your servers vulnerable to an exploit weeks after a patch is released, or have systems stop working because the code depends on a specific library that has been updated on one provider but not another.

It is important to fully document code and dependencies prior to moving to a multicloud environment—and you must demand the same from your cloud providers. You need to fully understand the services the organization is using, what the cloud providers'

patching cycles are, and how to get support if something goes wrong. You also need to share that information with your team so that everyone knows the limitations of each provider and is empowered to open support tickets.

Designing for a Multicloud Infrastructure

Moving to a multicloud architecture often means redesigning your application to accommodate a more complex networking and infrastructure environment. As with other documentation, this requires careful planning and assistance from the cloud providers themselves when possible. Many cloud providers have architects on staff who can help answer any questions from your team. They can also offer tips to make the migration process smoother. Remember that CSBs often have architects on staff who can advise organizations on different aspects of various cloud platforms.

Cloud architects are a valuable resource, and for many organizations, it pays to have one on staff. The architect could be an outside hire or a person who is already on staff who has been trained on the cloud platforms that the organizations intend to use. The advantage to having a cloud architect on staff is that you will have someone who knows and understands your application in a way that an outside consultant will not. On-staff cloud architects are empowered to give a more accurate assessment of what needs to be changed in order to maximize the success of the multicloud migration.

Conclusion

Migrating applications to a multicloud architecture will increase resiliency while saving your organization money in the long run. But multicloud migrations come with challenges and short-term costs. Before jumping into a multicloud infrastructure, it is important to take a step back and understand the maturity of your application and your team, and to make sure the organization is ready for the complexity of a multicloud deployment.

After completing an internal assessment, the next step is to understand which providers will meet your needs and what their offerings are. Then, it's time to design the architecture, working with a cloud architect whenever possible. This is also the time to document the new design for your application and make any changes needed to support the new architecture.

Finally, understand that this is not a "set it and forget it" process. The cloud provider is constantly changing, and it is important to stay abreast of the latest cloud provider offerings. If necessary, you can use a CSB to help you find the best match for the cloud services you need.

Multicloud is the future for most organizations—and the benefits outweigh any initial pain associated with the migration. But as with any business decision, it's important to properly plan the migration and to use a realistic timeline.

Chapter 2 discusses orchestration management in a multicloud architecture. We revisit some of the challenges discussed in this chapter and review ways to effectively manage the migration process and the ongoing maintenance of a multicloud architecture.

CHAPTER 2

Multicloud Infrastructure Orchestration and Management

In an ideal world, IT teams would have a completely heterogeneous environment in which to deploy their entire infrastructure. Reality can be quite different. An organization could have a legacy system that is difficult to maintain or could be out of compliance with current standards.

Deploying new applications in a multicloud environment can create infrastructure management problems because each cloud provider has a different management interface that requires different mechanisms for access. One way to ease these challenges is to use containers across all cloud providers. Those of you with a security background will read that and think, "Wait, containers are just a fancy way of saying virtual machine (VM), right?"

Not exactly. Although VMs and containers share many similarities, there are some key differences. Containers are isolated systems designed to be platform agnostic. But unlike VMs, a container image does not have a full operating system (OS). Instead, it uses extensions from the kernel of the host OS in order to run a specific application or function. Hence, a container mediates what an application does with an OS.

This means that container images are both smaller and more purpose driven than VMs. A container image might run MySQL or Apache, and multiple container images can live on a host OS. This allows for isolation of one application from another and it enables

organizations to easily redeploy an application from one cloud provider to another.

Containers have additional security benefits, as well. By stripping out unnecessary tools from the container and leaving just the application and its dependencies, you create a smaller footprint for bad actors to attack. Even if an attacker does manage to exploit an unpatched vulnerability on a container, they will have a difficult time. That's because the native tools an attacker normally uses to move around the system will not be there. After the attacker is detected, it is simply a matter of destroying the container and replacing it with a newly patched version.

Even though containers bring added benefits, they also come wit challenges, including management, security, and complexity concerns, which we discuss in this chapter.

Powerful Orchestration Interfaces Are Crucial

The cloud marketplace is constantly changing, with providers adding new features and enhancing their services regularly. The market is dynamic, and this means that your architecture must be able to quickly adapt to new capabilities. The web applications your organization deployed five years ago look nothing like the web applications you deploy today, and two years from now they will be completely remade again.

Infrastructure migration has always been painful, as evidenced by the large number of legacy systems that still exist today. But migration is the new norm. Containers help organizations adapt to rapidly changing infrastructure requirements, but they are only the first step. In addition to using containers, organizations need to use management and orchestration tools to effectively manage containers across all cloud providers. Using these tools, your organization will be able to quickly take advantage of new features and services.

One of the benefits of container deployment, discussed in the previous chapter, is that it is not dependent on the underlying hardware. Because the container has the application and all required libraries built into the platform, you can deploy it in almost any environment and function in the same way across all deployed platforms. However, in a multicloud environment, there will be dependencies between containers. Therefore, your management system needs to

be able to manage not just the container itself, but all of the dependent or clustered containers associated with it. It needs to be aware of the relationships between containers and be able to deploy those containers with their dependent containers across the multicloud architecture.

In general, a container management platform should do the following:

- Be cloud-provider agnostic whenever possible.
- Automate the deployment and decommissioning process.
- Manage the operating systems used by the containers.
- Monitor the health of hardware and applications running within the container.
- Scale with application usage.
- Allow secure connections for all communication and management.

There can be other requirements specific to your organization. As with any other technology selection, it is best to make a list of requirements before you start the test process. That will give you a formalized checklist to drive and validate testing of container management platforms.

Simple management platforms are great for small multicloud installations. Five applications running in 20 containers are easy to manage manually. But as your multicloud architecture grows, manual management will no longer be sufficient. This is where container orchestration comes into play.

Orchestration enables organizations to manage dozens of applications across thousands of containers using scripts to automate many processes, like the process of provisioning new containers, monitoring services on those containers, or decommissioning retired containers.

Container orchestration tools use JavaScript Object Notation (JSON) or YAML (short for YAML Ain't Markup Language) files to define a particular container. The file contains the information the orchestration tools need to deploy the container, such as hostname, IP address, open ports, installed applications, logging facility, and

more. Following is a partial example of a JSON container file taken from the Docker documentation:

```
{
    "Id": "8dfafdbc3a40",
    "Names":["/boring_feynman"],
    "Image": "ubuntu:latest",
    "ImageID": "xxxxxx",
    "Command": "echo 1",
    "Created": 1367854155,
    "State": "exited",
    "Status": "Exit 0",
    "Ports": [{"PrivatePort": 2222, "PublicPort": 3333,
      "Type": "tcp"}],
    "Labels": {
            "com.example.vendor": "Acme",
            "com.example.license": "GPL",
            "com.example.version": "1.0"
    },
    "SizeRw": 12288,
    "SizeRootFs": 0,
    "HostConfig": {
            "NetworkMode": "default"
    },
    "NetworkSettings": {
            "Networks": {
                    "bridge": {
                            "IPAMConfig": null,
                            "Links": null,
                            "Aliases": null,
                            "NetworkID": "xxxxxx",
                            "EndpointID": "xxxxxx",
                            "Gateway": "172.17.0.1",
                            "IPAddress": "172.17.0.2",
                            "IPPrefixLen": 16,
                            "IPv6Gateway": "",
                            "GlobalIPv6Address": "",
                            "GlobalIPv6PrefixLen": 0,
                            "MacAddress": "02:42:ac:11:00:02"
                    }
```

Fully defining containers this way makes it easier to deploy and redeploy them across cloud providers. It is also easier to make changes to specific containers without having to log directly into the container. For example, if you need to deploy more containers to a cloud provider that is seeing a large increase in traffic, you can do so quickly. You could also program your orchestration to automatically scale when traffic or central processing unit (CPU) usage hits a certain threshold.

Orchestration tools are powerful and they give organizations lots of flexibility in how they deploy multicloud architectures. But you need to know how to correctly take advantage of them. One challenge that organizations face (especially organizations new to multicloud architecture) is learning the full capabilities of the orchestration tools. A second is ceding too much control to automation. For these reasons, it often takes time to fully understand the capabilities and make the most of features available in orchestration tools.

Building Container Infrastructure in Multiple Cloud Environments

Even though containers are highly portable, making them perfect for multicloud deployments, there are limitations to their portability. It's important to keep this fact in mind when planning a multicloud architecture.

Given the complex nature of multicloud deployments, there are several steps that need to go into the planning process. The most important thing to do is to document everything. That documentation is going to be critical for others in the organization to reference. It will also come in handy three years from now when you go back and try to remember why certain decisions were made. Good documentation makes it easier for developers, IT staff, security teams, and management to work together from a common set of criteria. It also ensures that everyone understands the capabilities and objectives of the multicloud deployment project.

Next, you need to decide on a container engine. Most developers use Linux as the engine on which to run their containers. But there are some container services that run on Windows services (most notably, Microsoft's Azure). Whichever platform you decide to run on, it's important to make sure that all cloud providers you plan on using support it.

You will also need to choose a management or orchestration tool. Make sure tool works with the cloud providers you're using, and that it is flexible enough to support expected growth. Make sure it has support for HA and redundancy because you don't want your orchestration tool to unexpectedly crash.

Storage is an important part of the planning process, as well. Remember, these container systems are designed to be ephemeral and turned up and down as needed. Storage needs to be permanent with full redundancy and backups. Plan in advance where data will be stored and understand how the containers in each cloud provider will access that data.

Finally, understand how networking will work within and between the cloud providers being used. Will the containers connect directly to the host system, or will the containers run on an internal network segment? How will the different cloud providers connect to or peer with one another? Can you create VPN access between different cloud providers on management virtual local area networks, or do you have to find another way to connect different installations?

Kubernetes on Multicloud

Kubernetes is the most widely adopted container-management and orchestration tool, especially when it comes to multicloud environments. Originally developed by Google, Kubernetes is currently supported by Google Cloud Engine, AWS, Microsoft Azure, Oracle Cloud Infrastructure, OpenStack, and a host of other cloud providers.

The key to Kubernetes is its flexibility. It allows you to deploy fully configured systems across all cloud providers in your multicloud architecture. You begin by building the various components that are necessary to run the web application. You can then cluster those components together. You then can reuse these clustered components across different workloads and deploy them as needed.

Using the Kubernetes application programming interface (API), it is easy to quickly deploy new systems and clusters of systems to different cloud providers. It's also easier to deal with security patches. When a vulnerability is announced and patched, you can build and automatically deploy a new container across your entire architecture. This makes it less likely that a vulnerable system will remain exposed on the internet for long periods of time.

Kubernetes is very intelligent in its management of deployed containers. It maintains awareness of the state of each container and monitors system resources being used on each container. As previously mentioned, you can configure Kubernetes to deploy new containers automatically when CPU resources reach a certain threshold.

Kubernetes is also extensible with a number of projects such as Helm, a package manager, and Kubespray, for building clusters across multiple environments.

Finally, Kubernetes supports centralized logging for better security awareness. Collecting logs from deployed containers enables teams to analyze them for potential security threats and opportunities to improve performance. Organizations can tie Kubernetes logs directly into a security information and event management (SIEM) system such as Splunk to automate the correlation and analysis process.

DevOps and Multicloud

DevOps lends itself nicely to building out multicloud environments. With its emphasis on shortening the development life cycle by combining development and operations into a single function that manages both, DevOps perfectly mirrors the fast-paced nature of multicloud architecture.

As with multicloud deployments, moving to a DevOps development process requires careful planning and possibly a redesign of the underlying systems. But the benefits of the DevOps development cycle make the migration worth it.

Specifically, the use of containers enables users to quickly spin up testing environments for smaller project-based work. As the short-term work is completed, those containers can be destroyed, and the services can be reprovisioned for other uses. Multicloud environments allow DevOps teams to quickly test out new features offered by cloud providers. They also allow developers to quickly build new infrastructure every time they test code. Rather than testing and retesting on the same VM, developers can run the test, destroy the container, make changes to code, quickly spin up a new identical container, and rerun the tests. This ensures a fresh test environment each time. Although this type of rapid testing can be done using VMs, containers make the process easier.

Provisioning

One of the goals of DevOps is to quickly incorporate new requirements into the application development process. This makes the methodology a perfect fit for building container-based multicloud

architectures. Building and deploying containers to accommodate new features and deploying those containers across multiple cloud providers simultaneously is just a matter of editing a few files and making a few API calls. The same thing can be accomplished with VMs using tools like Puppet, but, here again, containers simplify the process. This kind of provisioning flexibility plays off the strengths of the DevOps process.

The use of containers is a perfect match, because it alleviates many of the platform-specific concerns that traditionally have accompanied DevOps cloud provisioning. By standardizing on a container-based platform, application performance is expected to be consistent across cloud providers. Although, that is not always the case. Containers do make it easier for deploying on different cloud providers, but there are lots of variables that are used to measure performance and one cloud provider can be noticeably different than another.

All of this assumes that the underlying security and management processes are in place and being followed. As long as security is built in to DevOps and container processes, you can rapidly and securely carry out provisioning.

Management

Proper management of resources is critical for successful DevOps and multicloud deployments. Multicloud deployments are inherently complex with many moving components, which is why management tools like Kubernetes are so important. But a tool is only as good as the management policies behind it.

Proper multicloud management enhances the DevOps process by providing a standardized view into the deployed systems and universal logging of the deployed containers. Having access to logs for troubleshooting and to better understand the resources being used can be invaluable when it's time to improve code or create new code.

Good management also empowers organizations to provide customized information technology services to DevOps teams. Having multiple cloud providers means that DevOps teams can access a menu of services available to them. Using these services, organizations can essentially offer Infrastructure as a Service (IaaS) to DevOps teams with proper management.

Orchestration Using Terraform

Terraform is another widely used orchestration tool for container management. Terraform treats infrastructure as code and maintains version-controlled container repositories. This makes it easy to track changes and revisit older containers when needed.

It also means that changes made on the fly can quickly be saved as either an updated repository or a fork of an existing container. You can store special configuration requirements or changes in case they are needed later. Much like Kubernetes, Terraform allows you to deploy clustered systems together. If an organization needs to quickly redeploy an application or add a new cloud provider, it is a simple API call to replicate the existing infrastructure.

Terraform also helps build execution plans. These plans help you walk through what will happen when you make the API call. As a result, organizations can ensure that everything will publish and connect as expected before actually deploying.

Terraform also allows organizations to work with a DNS and canonical name (CNAME) entries. Rather than rely on the rather obscure hostnames provided by many cloud providers, users can provision new containers with a DNS name already populated and get everything up and running that much faster.

Conclusion

Truly adaptive multicloud architecture makes use of containers for rapid deployment across different cloud providers. Although the use of containers greatly improves the agility of your organization's deployment, it also means greater complexity.

The best way to deal with that complexity is to use a container orchestration tool. The orchestration tool enables users to centralize and automate the deployment, management, and monitoring of all containers across all cloud providers.

This type of deployment requires well-documented underlying processes and policies, including security policies, to ensure that your organization consistently and securely deploys that infrastructure across all cloud providers.

CHAPTER 3
Security in Multicloud Environments

Security should not be an afterthought or an add-on to building a multicloud environment. Instead, security should be baked in to the development and deployment workflow. Rather than thinking of security as something that is handled by the security team, security should be at the forefront of everyone's thought process.

In Chapter 2, we talked about the importance of including security as part of the development cycle and the advantages of building secure, containerized systems. In this chapter, we take a closer look at securing your entire multicloud infrastructure. It is not enough to have a secure development and deployment process. It's also necessary to continuously monitor multicloud deployments and fully understand the threats your organization faces.

Even the most secure web applications are potentially open to security breaches either from previously unknown threats or vulnerabilities introduced in the deployment process. That's why it's important to create a security strategy that keeps everyone updated on the ever-changing threat landscape. You also need to continuously monitor for malicious activity.

This can require organizations to take a fresh look at how they view security. It can also require them to deploy new technologies that address the realities and dangers of deploying applications in a multicloud environment.

Edge Management Principles

IT security has never been easy. But one thing that used to be easier was the accepted definition of the network "edge." The phrase *edge of the network* used to refer to the firewall, or possibly the gateway routers. Everything behind that easily defined demarcation was the responsibility of the security team. Everything else was not. This resulted in "castle and moat" analogies that persist today about organizational boundaries. But those boundaries have eroded in recent years.

As more operations have moved to the cloud, the definition of the edge has changed. Regardless of your organization's definition of the edge, however, the principles of edge management have not changed.

What Is the "Edge"?

Before we continue let's define what we mean by the term "the edge," given that this will inform the rest of our security discussion. Most definitions of the edge conflate it with the Internet of Things (IoT), comprised of networked and interconnected devices. Such 'things' are certainly a big part of the edge, as these devices generally serve as gateways to and from networks.

But the edge is more than that, especially in a multicloud environment. A multicloud strategy means that every cloud provider your organization uses effectively becomes part of your organization's edge.

By viewing the multicloud environment as part of the edge, your organization can use available cloud providers' tools to restrict and redirect traffic, minimizing the attack surface of your cloud infrastructure, while making application services as widely available as possible.

Therefore, our new definition of the edge includes not just traditional IoT devices, but also the infrastructure deployed across the multicloud environment that is used to filter and redirect access to multicloud applications, such as DNS or Web Application Firewalls (WAFs).

Asset Management at the Edge

Before security teams can properly manage and protect the edge of the organization, they need to know what systems are out there. Good asset management is fundamental to good edge security. Security teams need to know what is deployed in each instance of a multicloud environment and on what hardware each of those systems is running. Asset management should not be a static process. Just as the development cycle is no longer a static cycle, measured in six-month or year-long intervals, asset management should be dynamic with continuous monitoring of assets deployed to the cloud.

One thing that all systems deployed to the edge have in common is that they are publicly accessible. This requires a different type of security strategy, compared to systems that are not deployed to the edge. Managing and securing edge devices requires security teams to distinguish between legitimate traffic and potentially malicious traffic. It also requires teams to block malicious traffic in real time. Isolation is far less feasible. The edge infrastructure is publicly accessible, which makes it a target. It might be a target of convenience, or it might be a malicious actor looking specifically to expose your organization's data. Either way, you need to closely monitor traffic hitting edge systems for potential attacks. Looking for known exploits alone is not enough. It's also important to look for malicious traffic that might not contain indicators of a known attack.

Finally, your security team needs to be able to correlate malicious activity across all edge devices and internal systems. Attacks, especially targeted attacks, no longer occur at a single point. An adversary will probe as many attack vectors as they can find. If the "security" for your edge devices sits in a console window separate from the rest of your security infrastructure, it can be difficult to make that correlation. The security team should be able to pull logs and traffic information from edge devices and quickly correlate that with other security incidents happening on your network. This ideally should happen within the security tools your organization primarily uses, whether that is a SIEM system, a managed security service provider (MSSP), or some other security tool.

Building on the secure development principles discussed in Chapter 2, the first steps to secure the edge of your network involve good asset management of cloud infrastructure, effective monitoring of

traffic to edge devices, and correlation of incidents affecting edge devices with other security activity happening in your network.

Why Protecting Assets on the Edge Is Critical

As discussed in Chapters 1 and 2, multicloud environments are complex and require careful planning, and this planning extends to implementing proper security controls. A misconfigured firewall rule, poorly deployed VPN, or lax API access control can result in the exposure of thousands or millions of customers' data. These vulnerabilities can also give attackers more ways to access your network.

Strong edge security can also help protect organizations against potential development security flaws. For example, protecting against cross-site scripting (XSS) attacks at the edge means sensitive customer data is less likely to be exposed.

Protecting edge assets is not just about ensuring that the rest of the cloud infrastructure is secure. It also means securing the edge devices themselves. Securing the edge in the cloud begins with securing the edge devices that protect that cloud infrastructure. You need to closely monitor and quickly patch systems deployed at the edge of cloud environments. An attacker who manages to exploit and gain access to one of these systems will potentially have access to not just the multicloud infrastructure, but possibly the entire network. Organizations need to treat edge devices in the cloud as critical assets, just as they would edge devices in the network.

Taking the necessary steps to secure and monitor the newly defined edge goes a long way toward building on the work that the development team did when building security into their process. This creates a multilayered security strategy and means that one mistake won't necessarily result in a breach, because other security measures will kick in and compensate for those failures.

API Gateways as a Mechanism for Centralizing Security Policies

One way to better manage security at the edge is through the use of API gateways. An API gateway is a system that sits in front of services deployed in a multicloud environment and acts as an aggregator. Rather than have each service talk directly to each other, they

communicate through the API gateway, which handles authentication and access policies.

At first glance, API gateways seem like they add complexity to an already complex multicloud deployment, but when implemented correctly, API gateways can reduce complexity. Although it certainly adds another layer to the environment, an API gateway centralizes security policies across different services and across multiple cloud environments. This helps ensure that access controls are universally deployed and enforced. An API gateway simplifies security and improves the security of API deployments.

In its most basic deployment, an API gateway manages authentication for APIs across multiple services, as shown in Figure 3-1, ensuring that only authorized users or systems have access to restricted API data and limiting the number of API requests that can be made. This also allows the API gateway to manage API key authentication.

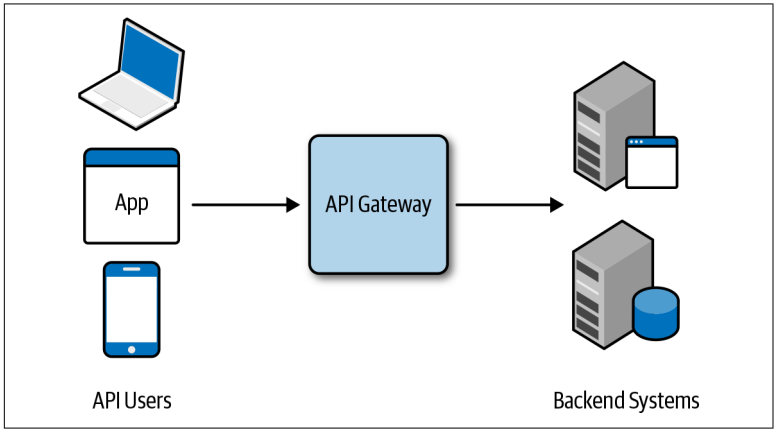

Figure 3-1. An API gateway

But API gateways can do so much more to enhance API security. API gateways can also act as a log aggregate across all API infrastructure, presenting API log messages in a single dashboard that can be imported into a SIEM system or sent to an MSSP. Of course, the effectiveness of the API gateway as a log aggregate is dependent on the quality of error message reported by the API itself. If the API reports only simple 200 "OK," 404 "Resource Not Found," or 500 "Internal Error" messages, the gateway itself can be less effective.

API gateways can also be used to implement whitelists and blacklists across the API infrastructure. This allows security teams to be even more precise in their secure API implementation. For example, if certain APIs should be accessed only from internal networks, the security team can use the API gateway to block access to all external networks. Then, a whitelist can enable access to internal networks. Whitelisting combined with authentication security controls allows for more robust API security.

Conversely, if there are concerns about attacks originating from specific networks against an open API, the security team can block those networks while still allowing most of the internet uninterrupted access. You should use blacklists like this sparingly because IP infrastructure is ephemeral and a network being used for malicious purposes today could be used by legitimate traffic tomorrow—or even an hour from now.

You can also use API gateways to monitor for traffic indicative of an attack on your API infrastructure, including SQL injection attacks, oversized JavaScript Object Notation (JSON), or XML requests designed to crash the target system or act as a denial of service (DoS) attack. Pulling log data across all API infrastructure allows your security team to monitor for coordinated API attacks or focused API attacks designed to disrupt a single resource.

It is important to note that securing the API gateway itself is just as important as securing the API infrastructure. Many organizations opt to outsource their API gateway services to a third party, which is perfectly reasonable as long as the organization has the ability to query the API gateway itself for potential security events. Before choosing an outsourced API gateway provider, understand how much access your team has to log in and monitor security events, and how you can retrieve that data.

If you decide to deploy your own API gateway, ensure that you are putting the proper security controls in place, including the following:

- Limiting who has access to the gateway
- Enabling logging for the gateway itself
- Applying patches regularly

Just as compromising other aspects of the edge infrastructure can give an attacker access to sensitive information, accessing the API gateway can give an attacker the ability to pull sensitive information directly through your APIs.

Web Application Firewalls

Another way to secure multicloud environments at the edge is through the use of Web Application Firewalls (WAFs). WAFs have been around in one form or another since the late 1990s but didn't see widespread adoption until the early 2000s.

A WAF is a detective and preventive security control that sits between the edge of the network and a web application, protecting the app from malicious attacks. Although most security devices are focused on protecting the people behind them, WAFs are specifically designed to protect software. Most WAFs use a combination of signature-based detection and heuristics to monitor for malicious traffic trying to reach one or more web applications. The WAF stops the malicious traffic, preventing the attacker from gaining access to sensitive information or to the target system itself. The WAF is another tool in a multilayered security strategy, sometimes referred to as *defense in depth*.

Traditional WAFs are physical devices that you need to deploy to each cloud provider in a multicloud environment. However, there are also cloud-based WAF providers that offer the same level of protection without the physical, on-premises deployment. These cloud-based WAFs give your organization more flexibility and allow for faster implementation.

WAFs have risen in prominence as a security tool over the past few years because attacks against web applications continue to rise. Web application vulnerabilities increased by 23% in 2018 over 2017, according to a report from Imperva. More than half of web application vulnerabilities reported in 2018 have a publicly available exploit, and 38% of reported web application vulnerabilities do not have a solution, such as a vendor-issued patch, to prevent exploitation.

Those numbers are scary and demonstrate the importance of a multilayered security strategy. A WAF can help organizations compensate for vulnerable web applications by intercepting the most

common types of attacks. A WAF can protect against XSS attacks, SQL injection, and attacks designed to expose potentially sensitive data. WAFs can either block attacks completely or alert administrators about possible malicious activity.

WAFs are highly customizable, making them ideal for vulnerabilities that have no known defenses. For example, if a vulnerability is discovered for a web application that would allow an attacker to access a direct object in an insecure manner, the security team can create a signature that blocks access to that direct object and quickly deploy it across all cloud providers. The WAF protects the web application while allowing legitimate traffic to continue flowing freely until you can deploy a better solution.

Keeping up with the latest threats against web applications can be a challenge for even the most advanced organization. That is why most WAFs have a threat intelligence backend that automatically updates as new threats are discovered and ensures that protections are automatically deployed. Organizations get the best of both worlds: automatically deployed protections against the latest threats, and customized solutions specifically designed for their environments.

Even though WAFs can improve the security of a multicloud architecture, they also add yet another layer of complexity. Too often, organizations deploy WAFs in a manner that protects the wrong applications, or winds up offering no additional protection at all. You need to carefully plan and consider WAF deployments early in the architecture design, not as an afterthought. The late addition of a WAF and poor WAF management can result in spending a lot of money for a solution that essentially does nothing.

Network Monitoring

To this point we have discussed building complex multicloud environments with multilayered security protections in place. But we also need to ensure that everything stays up and running, even though the complexity of multicloud deployments makes monitoring infrastructure difficult. In this section, we discuss network monitoring; the next section discusses security monitoring.

The term "network monitoring" is actually a bit of a misnomer. Certainly, monitoring traffic flows on the network is important, and a

key part of network monitoring. But real network monitoring also includes monitoring the health of applications and the hardware on which they run.

Traditional network monitoring relies on tools like Netflow, Syslog collection, and the Simple Network Management Protocol (SNMP) to gauge the health of monitored systems. Multicloud solutions use many of these same tools. Monitoring Netflow helps to ensure that traffic is properly flowing between systems, whereas Syslog helps to measure the health of the applications themselves. You use SNMP to check on the health of the underlying hardware. With these three components, you can effectively measure the health of the systems and ensure that your infrastructure is operating efficiently.

There is one more dimension of network monitoring that is required for multicloud architecture: performance monitoring. To gain a full picture of the health of your web application, you need to know how it is responding to real queries from all over the world and across cloud providers. Performance monitoring allows organizations to test web applications and measure performance from specific locations. This type of monitoring can also alert you to network activity that other types of network monitoring cannot, such as potential distributed denial of service (DDoS) attacks or possible BGP hijacking incidents.

Combining performance monitoring with Netflow, Syslog, and SNMP monitoring gives teams a complete picture of the performance of web applications and enables organizations to be more confident about web application functionality.

Network monitoring is only part of the monitoring equation, though. Security monitoring is also necessary to ensure the health of multicloud environments.

Security Monitoring, Logging, and Notification

Having all the layers of security in place only improves the security of your multicloud infrastructure if alerts are being addressed properly and in a timely manner. Target famously ignored early alerts of its much-reported 2013 data breach, which wound up costing the company hundreds of millions of dollars. Target is not alone.

Companies routinely miss alerts or don't respond to alerts in a timely fashion, leading to costly data loss or exposure.

Security monitoring adds yet another layer of complexity. But the good news is that security monitoring is very manageable. Although it's not easy to implement, it can be done. And many organizations today are successfully monitoring their applications for attacks and other security incidents.

It begins with understanding the event logging capabilities of the infrastructure in use in the multicloud environment. If you are starting from scratch, this is easy because you can ensure that all systems or vendors being deployed meet strict logging requirements that you set. Those requirements should consist of things like:

- Consistent event logging with clear explanation of events
- A way to access logs remotely, either through Syslog, an API, or another standard logging interface
- A well-defined log structure so that it is easy to develop new collectors

Compatibility with Log Aggregate Tools Used by the Organization

If a vendor cannot meet these requirements and others that might be specific to your organization, choose another one whenever possible.

Of course, even new multicloud deployments might involve the use of legacy systems. These systems might not meet the requirements you outline. In these cases, you need to ensure that the compensating controls from other security systems can provide your team with the same types of information.

Remember that it's important to standardize on a log aggregate tool. This will help you avoid the prospect of having to manage the security of dozens or hundreds of systems in your multicloud architecture one console at a time. Instead, you want to make sure that you are sending those logs to a centralized location so that they can be aggregated and correlated.

Of course, you might already have a log aggregate tool in place. You might use a SIEM system or an MSSP to handle your monitoring

and alerting. Bringing logs from the multicloud environment into an existing log collection system makes the most sense because your team is already comfortable with the tool.

The log aggregate tool makes your life a lot easier and makes your security team much more effective because it takes the disparate log sources, normalizes them, and presents them to the security team in a manner that is standard across all systems and providers. This normalization and standardization allows security teams to better monitor and prioritize alerts. Instead of running around trying to fix every alert that comes through, the team can focus on the highest priority alerts first and work its way down the list.

To truly enhance the effectiveness of your security monitoring solution, consider sending asset and vulnerability scan results to your log aggregate system (assuming it supports that type of data). This will allow for an even more effective correlation. For example, if your SIEM system alerts you to a potential XSS attack against your web application, you can quickly pivot to asset information to see whether the target system is running any applications that are vulnerable to that type of attack. The answers to these questions enable teams to assign the most accurate priority level to the alert and respond accordingly.

Your chosen logging platform should use data analytics technology to process incoming logs and parse those based on the priority of the event, adding additional weight to the event if it can be correlated across multiple systems and across multiple cloud providers. Using those same analytics, the log aggregate platform can help reduce the number of high-priority alerts from hundreds per day to dozens.

Just as containers help organizations seamlessly deploy complex systems across multiple cloud providers, an effective log aggregation tool provides the same picture of a security alert across all systems and cloud providers. Allowing your incident response (IR) team to deal with an incident in a more efficient fashion, hopefully stopping the attack before it can do any damage.

Conclusion

Multicloud environments come with their share of security challenges. That's why security should be factored in to the architecture process from the start. Treating multicloud infrastructure as the new edge requires teams to implement a layered security approach that includes vulnerability scanning, API gateway deployment, and WAFs as a final check against web application exploitation.

Of course, all of this security infrastructure is pointless if it is not properly monitored from both a networking and a security perspective. Using the right tools to aggregate monoitoring from all systems and providers in a single location will help ensure that the process of managing security for the multicloud environment will be as seamless as possible.

In Chapter 4, we delve into specific use cases for web application security.

CHAPTER 4

Multicloud Security Use Cases

In Chapters 2 and 3, we covered multicloud security at the core and the edge, but there are other use cases for multicloud security that we need to explore in more depth. Some of the use cases described in this chapter are not applicable to every multicloud installation. But it is important to be aware of their existence in the event that they become relevant.

Each use case described in this chapter was designed to improve availability, reduce the threat of targeted application attacks, or mitigate the threat of DDoS attacks.

DNS Resiliency and Traffic Steering

When we talk about securing the edge of your multicloud architecture, it's important to begin with DNS. Surprisingly, using DNS to enhance the security of a multicloud environment is often an afterthought, or not considered at all. But there are a lot of security enhancements that high-quality DNS infrastructure can provide.

Selection of a DNS provider is important because without reliable DNS your application will be effectively unreachable. The right DNS provider can enhance the security of web applications by improving resilience and optimizing traffic management.

DNS Resiliency

There are a couple of ways that the appropriate DNS architecture can improve the resiliency of your multicloud architecture. The first is by hosting primary and secondary DNS name servers on separate networks and different provider platforms. When registering a new domain, the registrar asks for a list of name servers. Most people default to the servers the registrar provides to them. Or, if they have a separate DNS provider, they use the name servers given by that provider.

But the DNS protocol allows for primary and secondary name servers. A primary name server is the authoritative name server that hosts the zone file, which contains the relevant information for a domain name or subdomain. The secondary name server receives automatic updates from the primary and does not need to reside on the same network. In fact, hosting the secondary name service on a different network is highly recommended. This increases the resiliency of the DNS setup and means that even if there were a complete outage within the primary provider, the secondary DNS service would continue to serve at least some traffic.

The second way that DNS providers promote resiliency is with the implementation of *anycast protocols* on their authoritative name servers. Anycast is a routing protocol commonly (but not always, so check with your provider) implemented by DNS providers as a way to improve availability and speed up responses to DNS queries. The anycast protocol allows multiple, geographically diverse servers to share the same IP address. For example, the "hints" file that sits on every recursive server points to 13 root servers, but those 13 servers actually mask more than 600 servers dispersed around the world. The IP address for each root server is an anycast address for multiple servers.

When choosing or changing DNS providers, it is important to find out whether the authoritative DNS servers used for your domain sit behind anycast IP addresses. Anycast doesn't just act as a force multiplier, it actually helps speed up response by using existing routing protocols to ensure that the request for the IP address is fulfilled by the closest network server. DNS providers using anycast don't just increase resiliency by having multiple DNS servers behind an anycast address. They also increase performance by ensuring the closest server fulfills each request.

DNS Traffic Steering

The DNS resiliency features discussed in the previous section are built in to DNS and help DNS work effectively in large-scale deployments. But there are features that some DNS providers include as add-ons that aren't built in to the DNS protocol. This doesn't mean that these features are any less valuable or important to security. It just means that fewer DNS providers offer them.

Most of these enhanced features revolve around traffic steering, the process of redirecting requests between different sites in a multicloud architecture based on a previously-defined set of rules. DNS sits in a unique position because it is the first step in the process for requesting content from a web application. As a result, DNS is an effective place to implement traffic steering rules.

For example, organizations in the process of adding a new site to a multicloud solution might want to ease it into the rotation to ensure that the new site is as resilient as the others. In a case like this, the organization might choose to direct the traffic-steering capable DNS server to send only 20% of the traffic to the new site.

A more common approach is to use a DNS provider to perform health checks on your services prior to responding to a DNS request with an answer. In this type of traffic steering set up, the DNS provider continuously monitors the health of all web properties, regardless of where they are hosted. If a cloud provider stops responding to requests, the DNS server will no longer direct DNS requests to that cloud provider until it is responding again.

Using DNS in this manner is not quite as reliable as using a load balancer, but it can be much more cost effective. Customers or end users will be able to access the web applications without interruption, even during a major outage at a cloud provider's datacenter.

Finally, some DNS providers are able to use traffic steering as a way to maximize the performance of web applications. Performance optimization can take many forms. A common type of performance optimization is redirection of traffic based on geolocation information. In a multicloud environment, organizations might have the same web application running in dozens or even hundreds of locations. DNS providers can optimize incoming DNS requests and send visitors to the closest cloud providers. For example, suppose that the multicloud architecture is set up in datacenters in Miami,

San Francisco, London, and Tokyo, and a DNS request comes in from a location in Atlanta. A DNS provider that has traffic steering capabilities can direct that request to the datacenter in Miami.

You can use these same techniques to identify potential new locations in which to expand the cloud architecture. For example, if the DNS provider begins recording an increasing number of DNS requests from Morocco, and this is confirmed as a growing trend, it might be time to consider expanding to a cloud provider with a datacenter in Africa.

Again, these capabilities are not inherently part of DNS and not all DNS providers support them. If your DNS provider does support them, these features can help to enhance the security and availability of your multicloud architecture.

Secure Interaction with the DNS Provider

DNS providers can offer web applications an extra layer of security. But you need to be aware of the threat of DNS hijacking. It's such a big problem that ICANN issued a warning about the threat. It's important to use DNS to add a layer of security to cloud infrastructure. But organizations should also be sure to secure communication with DNS providers. Take the following precautions immediately for every domain currently registered by your organization:

- Catalog all of the domains that the organization has registered.
- Add two-factor authentication with registrars.
- Lock domains so that they can't be transferred or updated.
- Enable Domain Name System Security Extensions (DNSSEC) for all of your domains.
- Test regularly to ensure security procedures are followed.

Domain registrars should be considered part of your potential attack surface, so be sure to take steps to keep domains safe.

Bot Management

Organizations running public-facing web applications have undoubtedly run into problems with bots. Bots are programs that automate activity across the internet. Some bots are useful, such as Google's search crawler, whereas other bots engage in malicious activity. Malicious bot traffic accounts for almost 44% of all website traffic, according to Distil Networks.

Malicious bots engage in a wide range of activity from information stealing and automated hacking attempts to DDoS attacks. We discuss DDoS prevention in a later section. For now, let's focus on preventing bots from stealing information or hijacking services.

Companies often use bots to target competitors' sites and gather pricing information to ensure that their prices are lower. Although these simple bots are mostly an annoyance, more advanced bots can be used to cause serious damage.

For example, bots often target airline and hotel reservation sites. These bots will make reservations for flights or hotels, keeping legitimate clients from being able to make reservations and forcing those clients to go to a competitor. The bots then cancel the reservations, or simply let their carts expire (depending on the site and service), costing the airline or hotel site millions in lost revenue.

Other bots engage in even more malicious activity. These bots scan hundreds of millions of sites looking for specific flaws to exploit. It might be an XSS attack, or they might be scanning for sites with vulnerable versions of common applications, such as WordPress or Drupal. Upon finding a vulnerable site, they will exploit it and steal sensitive information or install malware or a form-jacking script to intercept credit card data.

Unfortunately, no matter how secure your site is, it is still possible to be attacked by these bots. With hundreds of thousands of bots scanning the internet around the clock, your security needs to be perfect. But the bad guys need to get lucky only once.

A multicloud installation of any size requires a cloud-based bot mitigation solution. Stopping malicious bots before they have a chance to interact with web applications keeps web services available for clients and end users. Outsourced bot protection services also offer several advantages over the do-it-yourself approach.

For starters, because these services have seen thousands of bots, they have the ability to detect bot traffic earlier than if you were trying to do it yourself. They effectively identify patterns because they are monitoring for signs of bot traffic across all of their customers, not just you. They can even detect bot traffic that is operating in "low and slow" mode, avoiding detection by accessing the target web application infrequently and from a range of IP addresses designed to look innocuous.

These services also have ways of challenging potentially suspicious traffic, while not disrupting service if the traffic is legitimate. One way that sites manage this type of behavior is through the use of CAPTCHAs, which are little challenges that are designed to distinguish human from bot. If you have ever seen the question, "How many of these pictures have traffic lights?" or "How many images contain cars?" you have experienced a CAPTCHA challenge.

Unfortunately, bots are getting very good at solving CAPTCHAs—some bots are better at it than a lot of people. Rather than relying on faulty CAPTCHAs to distinguish humans from bots, bot management services will try JavaScript challenges and other methods of querying the browser to make that distinction. Because bots don't have full browsers behind them, they almost always fail these types of challenges.

Bot management services can significantly reduce the amount of malicious bot traffic that reaches your web application. Cloud-based bot management services can be quickly deployed across a multicloud architecture, and you can easily add or remove them as you scale up or scale down services within the multicloud environment.

API Protection

In Chapter 3, we discussed the importance of APIs in a multicloud architecture. APIs are used to connect all of the disparate services running in a multicloud environment and are critical for getting information from one source to another and presenting it in a unified manner to an end user or client.

This is why API protection is so important. Attackers have become wise to the fact that APIs can provide them with a treasure trove of sensitive information. As a result, these bad actors are constantly looking for ways to exploit APIs, including the use of bots.

API protection technology encompasses a number of different areas:

- Limiting who or what can access APIs
- Limiting how much data can be retrieved at any one time
- Ensuring that all data transmitted via API is properly encrypted

Many of these protections can be put in place using a combination of an API gateway and a WAF. This means that APIs don't necessarily require an investment in new technology. Organizations simply need to take advantage of the right API gateway and WAF features. Let's take a closer look at the process of ensuring API protection.

For the most part, a human should never access an API directly. An API is designed to programmatically share information from one system to another. The second system will render the information and present it in a way that humans can understand and use. Clients and end users should automatically be blocked from making API calls, and those calls should be coming only from authorized systems that are part of the multicloud architecture or from trusted partners that might also be querying your backend systems.

Users are restricted from directly making API calls, but they can still find ways to manipulate authorized systems to make malicious API calls. This is why it is not enough to restrict which systems can make calls. The calls themselves should also be limited by the amount and type of data returned.

For example, an API call from a public-facing system should never reveal multiple usernames and passwords. (In fact, it is probably a bad idea to have the ability to reveal a password at all.) Additionally, there shouldn't be a way to pull all customer names and email addresses from a public-facing system. Using the API gateway to control access and creating signatures on the WAF to block these types of queries can prevent data leaks from occurring, even if an API is unintentionally left exposed.

You also can use API gateways and WAFs to enforce encryption policies across API communication. Most API communication should be conducted using Transport Layer Security (TLS) encryption. If for some reason an API is not encrypted natively, the call can be forced to run across a VPN to provide the communication with some level of encryption.

APIs are a critical component of multicloud architecture. It takes a great deal of planning to deploy APIs securely and ensure that the data shared between different systems via API calls remains protected.

Application-Layer DDoS Protection

DDoS attacks are a reality that any organization hosting a public-facing web application must face. DDoS attacks can be launched at any time against any organization for any reason, or no reason. Protecting against these attacks should be part of your design plan.

There are two types of DDoS attacks that we focus on in this book: *application-layer* DDoS attacks and *network-layer* DDoS attacks.

As the name suggests, application-layer DDoS attacks target a specific web application or API, using up the Layer 7 (L7) resources while preventing legitimate users from accessing the services. Network-layer DDoS attacks flood the entire network with traffic, making all resources at your cloud provider unavailable, not just a specific web application or service.

Application-layer DDoS attacks are often more difficult to detect and stop because the attackers are making legitimate HTTP requests. As a result, sorting through the traffic and separating the attacker's requests from legitimate requests can be a challenge.

There are a number of ways to implement effective application-layer DDoS protection. The most common method is to use a WAF to block malicious requests before they can reach the web application itself. This requires implementing a WAF that can process a high volume of traffic without slowing down legitimate requests.

It also often involves understanding the nature of each attack. The very nature of a DDoS attack means that the attack is distributed—originating from thousands or hundreds of thousands of IP addresses. The attacks can also blend in with legitimate traffic, at least at first glance. Fortunately, there are usually distinguishing features to application-layer DDoS attacks. These features allow security teams to build signatures that can be deployed to the WAF and stop that traffic without impeding the flow of legitimate traffic.

A cloud-based WAF enables organizations to quickly deploy these protections across the entire multicloud infrastructure. If your orga-

nization does not possess the skill sets required to identify these patterns and implement protections, it might be advantageous to take a look at managed WAF services, which are available from providers who can monitor and deploy protections on your behalf.

Another way to protect against some types of application-layer DDoS attacks is to use different types of challenges. We discussed this type of protection earlier when we took a look at bot protection. Some types of application layer DDoS attacks behave similarly to bots. In fact, they often use the same underlying technology. This means that they can be stopped by using many of the same techniques.

Presenting suspicious traffic with a CAPTCHA or a JavaScript challenge before they can proceed to the targeted web application will allow you to quickly distinguish between malicious and legitimate traffic. These types of interrogating behaviors don't need to be applied universally. Instead, you can build rules that look for traffic patterns outside the norm and deploy the checks only when those patterns are identified. The advantage to this methodology is that you don't need to identify suspicious traffic, only traffic that lies outside of normal behavior. A disadvantage of this approach is that you run the risk of slowing down legitimate requests, which can cause clients to abandon the web application entirely.

Network-Layer DDoS Protection

The second type of DDoS attack is one that occurs at the network layer. These attacks use network protocols, such as DNS, Network Transfer Protocol (NTP), or Memcached to flood an entire network with so much traffic that all of the systems are overwhelmed. The largest network-layer DDoS attack ever reported generated 1.35 terabits per second of sustained traffic, which is more than all but the largest of networks can handle.

Network-layer DDoS protection involves different types of security measures that should be implemented at the network layer. Most cloud providers will not be able to stop these attacks at your edge.

The trick is to put DDoS protections in place that will intercept malicious network traffic and stop it before it has a chance to even reach the cloud provider. DDoS protection services monitor for malicious traffic and stop it even before it can reach the edge of your

architecture. Unlike application-layer DDoS attacks, network-layer attacks don't "blend in" with existing traffic in your network, so there is little chance of disrupting legitimate traffic to your web application while stopping the DDoS attack.

This is another advantage of running a multicloud architecture. By building a geographically diverse environment that is hosted across multiple networks, you are building in redundancy that makes the web application less susceptible to network-layer DDoS attacks. Even if one site is temporarily taken offline, the other sites can manage the load.

As with other types of security measures, it is important to plan your DDoS mitigation strategy and test that plan repeatedly to verify that the sites function as planned during the different types of DDoS attacks.

Deep Internet Monitoring: Data Intelligence

We've touched on monitoring a few times throughout this book, but it's worth a deeper examination. A multicloud infrastructure is complex by definition and requires a sophisticated monitoring solution. It is not simply a matter of monitoring the network infrastructure to ensure that it is up and running and monitoring the performance of the application. Organizations also need to understand how the web application performs from locations around the world.

Deep internet monitoring is about more than monitoring your architecture and how different sites connect to one another. It is about monitoring the performance of the internet itself. The internet is resilient, and a full outage is highly unlikely. But small, regional outages occur all the time. In 2018, there were 12,600 routing incidents worldwide. These types of disruptions can last for minutes, hours, or days and affect only part of the internet.

Organizations might be completely oblivious to these attacks as they occur. But they can affect the people trying to reach a web application no matter how much redundancy is in place. Your clients won't care why there is a disruption; they'll care only about the fact that they cannot reach your site. This is why it is so important to understand how your web application is performing from as many places as possible.

Collecting monitoring and performance data from a large number of sources and tracking performance over time allows organizations to better understand problems and react quickly to outages. These monitoring trendlines can help organizations pick the best cloud providers when they need to stand up additional infrastructure. They also help organizations determine which cloud providers are underperforming, ultimately saving money while organizations make the client experience better.

Combined Policy, Management, and Visibility

Everything described in this book works only if your organization combines policy, management, and visibility across all of your cloud providers. This is absolutely necessary, even though each cloud provider in a multicloud architecture has different capabilities and different provisioning and management platforms.

The only way for a multicloud solution to be efficient and effectively managed is if everything is viewed through the same platform on your end. Using a unified platform that is cloud-provider aware can make it easier to deploy, enforce policies, and manage and monitor all providers through the same "pane of glass."

This also allows organizations to take advantage of efficiencies of scale. Organizations can quickly deploy to new datacenters, patch systems across all cloud providers simultaneously, and enforce new security policies, such as new DDoS protection rules. A centralized management and monitoring framework provides end-to-end visibility and gives organizations an early warning when systems are beginning to fail.

Complexity Requires Granularity of Policies

Although you can build out management and monitoring tools after the multicloud infrastructure is built, system policies should be decided prior to deployment. It is important to deploy compliant infrastructure from the start, whether these policies are related to security, information retention, or regulations such as the Payment Card Industry Data Security Standards (PCI DSS) or the Health Insurance Portability and Accountability Act (HIPAA).

Trying to retroactively make a non-policy compliant facility or system compliant after deployment is painful and can lead to gaps that violate the policy.

Policies also need to be granular to properly cover a multicloud deployment, especially if that multicloud deployment is spread across multiple countries. Different countries have different regulatory frameworks and might require various policies to remain compliant. For example, organizations using cloud providers in the European Union need to meet General Data Protection Regulation requirements. Understanding the policies for each location in which you are deployed and then enforcing those policies is a key part of the planning.

Beyond regulatory policies, organizations need to ensure that security is enforced in the same manner across all cloud installations. This is easier to do using containers over VMs. But you do need to worry about the hardware on which the containers operate. Understanding the patch and enforcement policies of each cloud provider and, if necessary, compensating for any deficiencies in their process should be part of your planning.

The complexity inherent in any multicloud solution can be somewhat mitigated by standardizing a management solution across all cloud providers that enforces policies and provides full visibility.

The Edge Allows for Simpler Managed Services Offerings

In the end, focusing security solutions at the edge rather than the core makes it easier to incorporate managed service offerings as part of your multicloud architecture. The organization will benefit from being able to quickly deploy these solutions across all cloud providers, and it will help keep costs down by using only the services it needs.

Edge deployment of managed services also makes it easier to deploy new managed services as needed. Organizations that are just starting the process of moving to the cloud might want to focus on deploying WAFs initially but hold off on other forms of protection. When you are ready to add additional services, it will be easier to find and deploy the right service across the entire multicloud architecture solution.

As web applications continue to grow, the managed services footprint can grow along with them. Organizations can add new providers and enable new services within existing providers. This type of deployment-on-demand also makes it easier to switch managed services offerings when a vendor isn't performing properly or the organization simply outgrows its capabilities.

Conclusion

A multicloud architecture requires rethinking how security is deployed across all cloud providers. By securing the installation at the edge, organizations have the flexibility to deliver security across the entire architecture while increasing visibility and improving the performance of web applications.

By working closely with security partners, organizations can find solutions that fit their specific needs. These solutions can grow along with the web application and the organization itself.

To take full advantage of these solutions, organizations must first understand what the needs are and ask the right questions. Failure to do so can result in your being boxed into a solution that is not a good fit. With proper training and research, your organization can effectively secure multicloud architectures at the network edge.

About the Authors

Laurent Gil runs product strategy for internet security at Oracle Cloud Infrastructure. A cofounder of Zenedge Inc., Laurent joined Oracle Dyn Global Business Unit in early 2018 with Oracle's acquisition of Zenedge. Prior to that, Laurent was CEO and cofounder of facial recognition software and machine learning company, Viewdle, which was acquired by Google in 2012.

Laurent holds degrees from the Cybernetic Institute of Ukraine (Doctorate Honoris Causa), the Wharton School of Business (MBA), Supélec (M.Sc., Computer Science and Signal processing), the Collège des Ingénieurs in Paris (postgraduate degree, Management), and graduated Summa Cum Laude from the University of Bordeaux (B.S. Mathematics).

Allan Liska is a solutions architect at Recorded Future. Allan has more than 15 years, experience in information security and has worked as both a blue teamer and a red teamer for the intelligence community and the private sector. Allan has helped countless organizations improve their security posture using more effective and integrated intelligence. He is the author of *The Practice of Network Security* (Prentice Hall), *Building an Intelligence-Led Security Program* (Syngress), and *NTP Security: A Quick-Start Guide* (Apress), and the coauthor of *DNS Security: Defending the Domain Name System* (Syngress) and *Ransomware: Defending Against Digital Extortion* (O'Reilly).