EPLEXITY

# Leveraging App Patterns for AWS Success

CXOS

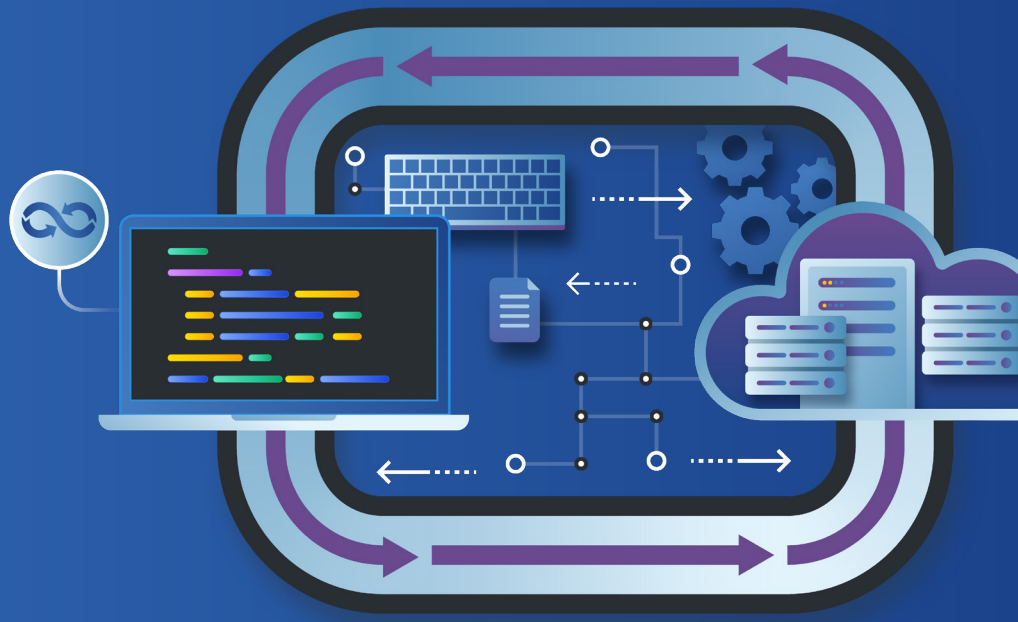
# Table of Contents



The cloud has made it easier than ever to deploy applications. With just a few clicks or a handful of keystrokes, you can provision an environment and start hosting an application in the cloud of your choice. At the same time, however, cloud computing has made it more challenging in many respects to deploy applications in an efficient, reliable and secure way. A typical cloud workload consists of many moving parts, and relies on multiple cloud services. Integrating all of these components effectively is complicated – much more so than it would be if you were simply hosting applications on on-premise servers.

The answer to addressing these challenges lies in choosing the right cloud application patterns. By application patterns, we mean cloud deployment strategies, integrations and best practices that maximize efficiency, reliability and security. Only by identifying and adhering to the application patterns that work best for your workloads can your business thrive in the complex, fast-changing world of cloud computing.

With this need in mind, we've prepared this eBook, which explains how to make the most of app patterns in the cloud. It focuses, in particular, on Amazon Web Services, or AWS, the most widely used public cloud computing platform. However, the lessons below apply in general to any modern public cloud.



*We hope the following pages empower your organization to get more out of its cloud computing strategy, and to leverage the ease and flexibility of the cloud without compromising efficiency, reliability or security.*

# What are AWS Application Patterns?

The cloud-native approach is not just about compute power and cloud services. A "set it and forget it" attitude won't work with cloud-native; there are far too many moving pieces and there is just way too much complexity. Instead, this approach requires constant testing, optimization, and improvement. This is epitomized in the two key phrases of the DevOps methodology: continuous integration and continuous delivery.

To make this a reality, however, it takes a strategic approach to defining application patterns that will govern the way in which cloud-native applications are built and shipped, rather than a one-off effort. Since AWS is the leading cloud provider, we should discuss which application patterns are relevant to the AWS platform. There are many aspects to consider when it comes to AWS application patterns, and we will cover them in this chapter.

## Operational excellence

The best starting point for application patterns is a commitment to operational excellence. Typically, the term 'Ops' is used to identify the IT team, but in this case, operational excellence rests on the shoulders of Dev, QA, and Ops teams. In cross-functional DevOps teams, every member should strive for operational excellence at every step - even in the smallest of tasks.

In their white paper on operational excellence, AWS encourages its users to manage operations as code. This helps avoid human error, enables automation of complex processes, and allows for quick and easy rollback when errors occur. Failure, according to AWS, is an opportunity for learning. They even advise performing 'pre-mortems' that try to anticipate potential points of failure before a deployment.

AWS Trusted Advisor is a useful tool that enables this type of accountability and shared responsibility within a team. It checks and reviews your new and existing AWS workloads looking for opportunities to save costs, improve performance, tighten security, or follow best practices for scaling operations.

## Security

Security is a pattern that is non-negotiable. The term 'DevOps' has inspired another variation called DevSecOps that signifies this fact. Security operations take center stage when talking about DevOps.

The interesting thing about DevSecOps is that it actually results in more freedom, flexibility, and trust across teams when done right. This is unlike traditional approaches to security which created silos, stifled creativity, and slowed down innovation. Security is now a key piece of the puzzle, not an afterthought. AWS has many machine-learning-based security solutions such as Macie which automatically identifies and secures personally identifiable information (PII).

Along with this, configuring role-based access controls using AWS IAM is essential to have greater control over who or which applications access data and for how long. Access Control Lists govern access to data stored in AWS S3 buckets.

The news is full of instances of data breaches due to weak security policies and access controls. Capital One is a recent example of a data breach that affected 100 million of its users. While security in the cloud is more complex than before, AWS has the necessary tools and systems in place. It's up to organizations to implement a security strategy that prevents misuse of their data.

# What are AWS Application Patterns? (cont.)

## Reliability

Today, there is a low tolerance for downtime because of the potential loss of revenue. With the stakes so high, reliability is a pattern that any application custodian would have high on their list of priorities. AWS has multiple reporting and health monitoring tools to keep you constantly on the pulse of your applications.

Architectural patterns like microservices enable better resilience by decomposing monolithic applications into more manageable chunks called services. This 'divide and conquer' approach leads to higher availability, which is critical for SaaS applications.

Patterns like chaos engineering seek to intentionally kill random instances and services and force the DevOps team to figure out workarounds and backups for every scenario.

## Application Performance

With the explosion of data, applications are always dealing with lots of data to transfer, store, and process. An application's performance depends on three factors: the backend data storage layer, the middleware components, and the frontend.

The backend data layer is the most critical component when it comes to performance. Choosing the right storage formats, database types, storage devices, and organization patterns all contribute to the speed of data access. AWS has an amazing array of storage services, including S3, RedShift, Aurora, and Glacier. AWS gives its users every type of storage solution, all of which should be considered to improve data management.

AWS is devising new ways to query large datasets without bearing the load on your existing systems.

RedShift Spectrum is one example of this. It is a great way to "minimize the need to scale Redshift with a new node, which can be expensive." If you'd still rather not have to manage any RedShift clusters on your own AWS Athena is a great serverless alternative. That gives you all the power you need to query large datasets in S3 without having to maintain any clusters of your own.

This pay-per-use model is making its ways across the entire AWS stack with services like ECS and Fargate offering completely serverless experiences when operating containers at scale. What started out with AWS Lambda is catching on fast, and many AWS services like the ones mentioned above enable serverless experiences with a focus on application performance.

# What are AWS Application Patterns? (cont.)

## Cost optimization

When moving to the cloud, pricing is completely different. Rather than paying for perpetual licenses, you now pay a small fee for every small resource that you use. Though small on its own, you'll soon realize that cloud costs can shoot up easily if you're not paying close attention to your spending.

Cost optimization is necessary as you scale operations in the cloud. Cost optimization includes looking for more efficient services, whether that's within the same vendor platform or a different platform. At a minimum it may require a lift-and-shift approach to transition from expensive old infrastructure to inexpensive new alternatives. For example, shifting workloads from VMs to containers to serverless within the same platform can show great cost savings, since containers and serverless are generally more cost-effective than VMs.

It pays to leverage the expertise of experts in this area. They should ideally be people who have spent years using AWS, understand Well Architected best practices, and have found ways to cut costs and derive more value from the same budget.

## Application patterns enable extensibility

Microservice applications adopt a pluggable architecture. This enables them to be easily integrated with external applications. Typically, API-based communication patterns are most common. AWS API Gateway is a service that manages API design for AWS applications. In some cases, an SDK is the better choice - for example, when you need more control over the developer experience and integrations. It's important to know when to choose either option.

## Application patterns enable portability

Portability is the ability to move workloads across instances without interrupting their functioning. As instances change, the workloads which they support shouldn't be affected. This is only possible if there is a clear separation layer between the workloads and the infrastructure that runs them.

# What are AWS Application Patterns? (cont.)

## Application patterns enable shift-left

The DevOps methodology is known to encourage a shift-left approach to software delivery. This means that not only Dev, but also QA and Ops contribute to the design and development stage of an application. This fosters deeper collaboration across teams and breaks down silos.

A shift-right approach holds true as well. Shift-right means that developers are responsible for the code that they write. They are equal owners with Ops teams for the resilience of the apps they build and also the bugs that they introduce into the system. This practice of putting ownership and responsibility where it belongs is very healthy for a DevOps team.

AWS facilitates pipeline management using tools like CodePipeline, CodeCommit, and CodeDeploy. They each perform similar but complementary roles and help build a CI/CD pipeline. They make it easy to get started with CI/CD within the AWS platform.

For organizations that are further down the line in their cloud journey, AWS' pipeline management tools may seem too simplistic for something like GitOps. Instead, they'd want to look outside AWS for tools like Jenkins X or Weave Flux which are designed with GitOps automation in mind.

**Whether it's the simple or advanced route, a pipeline brings great benefits over a manual development process.**

## Application patterns summed up

Though tools enable the implementation of helpful application patterns, on their own, they are ineffective. Application patterns need a deliberate strategy and an approach for a start. Only after a strategy is in place, can you move to deciding which tools to use. With a strong strategy in place, any of the tools mentioned here can be swapped out for other better tools and the results will be the same. It's important to not be distracted by the next shiny tool, but keep a keen focus on the strategy for implementing AWS application patterns.

As we've seen here, application patterns are diverse, but they all build upon each other. Without even one of these pillars, organizations would not be able to realize the goal of cloud-native applications. On the other hand, if they give careful thought to each of these application patterns, organizations would deliver cutting-edge applications that can only get better with the passing of time.

# How Organizations Benefit from AWS App Patterns

As a startup, an organization thrives by breaking rules, challenging the status quo, and standing for all that is against the norm. The same rules don't apply once the same organization grows, matures, becomes popular, gains a long list of clients, and expands to a global footprint. With scale and growth comes the need for reliability, predictability, and consistency, not just with the technology, but with the customer experience as well. At this stage, application patterns become indispensable to an organization's day-to-day functioning.



## Support of the application now and long-term

Mid- to large-size organizations typically deal with numerous internal-only applications and external vendor applications. These applications are changed and updated regularly. To support the applications' needs, and effectively manage them, organizations need a change management system in place.

A service like AWS Config helps to audit AWS services that power applications so there's increased visibility. It tracks changes made by users and applications, and presents a clear picture of how these changes impact AWS resources.

## Enabling Shift-left without additional overhead

DevOps bring together Dev, QA, and Ops into smaller, cross-functional teams. The goal is to break down silos between teams, facilitate collaboration, and drive better results faster. Accordingly, shift-left is a tactic used to encourage QA and Ops teams to play a more pivotal role in the initial Dev stages of the pipeline.

The benefits are many. QA brings their experience of testing the application to its limits and informs about what's feasible to implement. Ops chimes in with concerns and things to watch for once deployed. While shift-left is great from a team standpoint, it can meet with hurdles and can fail if communication patterns are random, and ownership of each step is not clearly defined.

Application patterns encourage systematizing this collaboration between teams. Patterns help to define the type of infrastructure to be used, their configuration, and the kind of tools that fit in with the team's requirements. Application patterns guide decisions and help to make the right choices, and sometimes compromises, at every step.

# How Organizations Benefit from AWS App Patterns (cont.)

## Supporting the advancements in Open-Source without fear of tool attrition

There is an explosion of open source tools today. If DevOps teams are attracted to the next shiny new open source tool, they'll soon find themselves overwhelmed with tools they don't use and that slow them down.

Open source tools evolve quickly, and their popularity rises and falls with the times. One look at the open source monitoring space will show you that older tools like Nagios and Zabbix are now becoming less relevant, and modern container-aware monitoring tools like Prometheus are taking their place.

AWS supports the use of the top open source tools; for example, Elasticsearch. The AWS Elasticsearch service integrates seamlessly with Kibana for visualization and delivers the ELK stack without the maintenance.

Application patterns enable you to define a strategy for tool adoption. You may find the best open source tool in the world, but what's the point if you can't use it in your toolchain, or if it slows down the pace of operations? The popular DevOps mantra of people, processes, and then tools holds true here.

## Value Stream Management

Value stream mapping is about visualizing the flow of value from an organization to its customers. In simple words, it's about taking stock of what you have to offer to customers, and monitoring how your offerings change over time.

Value flows outward to customers, as well as internally between employees and systems. For example, in the context of DevOps, it helps to map the value of Ops to Dev and QA. As these teams are each others' internal customers, knowing what the value stream is can help remove bottlenecks and drive greater efficiency.

## Building a team and culture first

AWS is a great place to build team culture as it is expansive and includes services for pretty much anything you'll ever need. You can find tools like CodePipeline to quicken development pipelines, CloudWatch and SNS queues to make sure everyone on the team is informed of important changes , and Lambda to automate common tasks.

Rather than jumping feet first into AWS and then figuring out how it all works, it pays to first define application patterns and team culture to govern the way you use AWS. Patterns and culture are closely related. If culture is about the people, application patterns are about the infrastructure and technology stack being used by people.

Whether you've yet to move to AWS, or are already on board but not getting the most out of it, application patterns help evolve the way you manage applications in AWS. Application patterns help scale apps over the long term, and they enable shift-left for DevOps teams without the friction of overhead. Application patterns make it possible to adopt the right open source tools and find success with them. By using application patterns you can visualize the entire process and drive great efficiency, and finally, they support great team culture within DevOps teams.

DevOps teams that run their operations predominantly on AWS need not fly blind. Application patterns bring greater visibility, control, collaboration, and ultimately value.

# What It Really Takes to Build a Reliable App (Hint: Not Just Uptime)

What makes an app reliable? If you ask most IT professionals that question, their minds immediately go to uptime. That's not surprising; after all, it's fashionable for software and infrastructure vendors these days to boast about how many "9s" of uptime they guarantee.

To be sure, uptime is one component of delivering a reliable application experience for developers and end-users. But it's only one. In order to build reliability into your application patterns, you must think beyond uptime alone.

Keep reading for an overview of all the factors that go into achieving reliability.

### What is reliability?

A reliable application is one that meets the needs and expectations of everyone it serves. While those needs and expectations will vary from case to case, reliability is characterized by the ability of an app to:

- Be accessible when needed.
- Respond within the timeframe needed.
- Be updated or modified as needed.
- Provide security and privacy to the extent needed.
- Meet the needs and expectations not just of end-users, but of everyone else who helps create or support the app.

# What It Really Takes to Build a Reliable App (Hint: Not Just Uptime) (cont.)

## Maximizing reliability for cloud-native apps

Achieving these features for modern, cloud-native applications requires a multi-pronged approach. The following are the core considerations to address in order to build reliability into an app pattern:

### UPTIME

We'll start with uptime, which is the most obvious aspect of reliability. Uptime refers to the amount of time that your application is available, at least to some extent (as we'll discuss below, merely being available does not necessarily mean that the app satisfies other requirements of reliability).

Maximizing uptime typically involves solutions such as distributing workloads across different servers or data centers (so if part of your infrastructure fails, others remain available) and using automated failover to move a workload to new infrastructure if it fails in one location.

Culture and processes are part of the picture, too, since you need to make sure your team is prepared to respond quickly to any incidents that arise, in order to minimize downtime.

It's worth noting, by the way, that there can be such a thing as too much uptime. If you reach the point where adding more uptime doesn't deliver any additional value to your organization, then it's probably time to stop. Adding another 9 to your availability just because you can might be a poor use of resources. So, be sure to factor this part of the equation into your uptime calculations and strategy.

### PERFORMANCE AND EFFICIENCY

In order to meet stakeholder needs and expectations, an application must not only be available, but also able to respond within the timeframe required by those stakeholders. A website that is "up" 100 percent of the time but takes 20 seconds to load each page is not a very reliable website.

This is why performance testing and monitoring must be built into your application delivery chain. You should also ensure that you "rightsize" infrastructure (for example, by choosing the right types of cloud instances) in order to guarantee that your applications have enough resources available to perform adequately. (Of course, this priority must be balanced with cost-control.)

### ACCESSIBILITY AND USABILITY

An application that is available all of the time and responds quickly may still fail to meet stakeholder needs if it is difficult to work with due to poor design.

From an end-user's perspective, user-experience testing is the most obvious way to help mitigate the risk of poor accessibility or usability within your applications. But keep in mind that you also need to think about usability from the perspective of your own team. Make sure that access-control systems, infrastructure architectures, and documentation can be navigated easily. You don't want an application or a delivery chain that is too complex for your team to be able to support effectively.

### SECURITY AND DATA PRIVACY

A reliable application is a secure application. There are many obvious ways to help secure applications, such as using vulnerability monitoring tools. But remember that security is not just about finding vulnerabilities as they arise or deploying certain tools. You must also build security into the design of your infrastructure, processes, and culture.

# What It Really Takes to Build a Reliable App (Hint: Not Just Uptime) (cont.)

### BACKUP AND RECOVERY

You don't need to be a seasoned IT professional to know that backing up data is important for achieving reliability. When something goes wrong – which it occasionally will, despite your best plans and efforts – having a backup in place often means the difference between experiencing a blip in service and a major catastrophe.

But the thing that can be easier to overlook is the importance of having a disaster recovery plan in place as well as a backup routine. Simply backing up your data isn't enough; you must also know how you're going to restore it to new application instances quickly if disaster strikes. If, for example, all of your cloud-based virtual machines go down, what is the exact process you'll follow for standing them back up? Identifying the steps ahead of time – and, ideally, scripting them so that the recovery process can be as automated as possible – is crucial.

Also important is quantifying how often you need to back up data (which is determined by a metric called Recovery Point Objective, or RPO), and how quickly you need to be able to recover it to avoid serious business disruption (determined by Recovery Time Objective, or RTO). Along similar lines, error budgets, which define how much time you can tolerate your production systems being down, can help you define the available time monthly for making system improvements.

### MODIFIABILITY

Few apps are deployed and then never modified again. Instead, they must be updated or scaled constantly to meet changing needs and demands. That's why designing an application pattern that can be easily modified is critical for achieving reliability over the long term.

In practice, modifiability entails having the right tools (like automated deployment solutions) and the right processes (such as clear feedback loops between IT Ops and developers) in order to decide efficiently what modifications to implement. It might also involve taking advantage of infrastructure technologies (such as containers) that make it easy to update a running application without imposing downtime on users.

**Reliability is more complex than it may at first seem. No matter how many 9s are included in your SLAs, you are not guaranteeing true reliability unless you also address requirements such as performance, usability, and modifiability.**

# Operational Excellence and the Success of Software Deployments

When we talk about best practices for software reliability, the conversation tends to focus on optimizing the applications themselves and the infrastructure that hosts them. The driving idea is that reliability must be baked into system architectures and infrastructure from the beginning.

That's certainly true. But by focusing too much on the design and implementation of applications, you run the risk of overlooking another crucial element for achieving reliability (not to mention agility and extensibility): a culture of operational excellence.

It's only by baking excellence into your organization's operational culture that you can maximize reliability.

Let us explain...

### Defining operational excellence

Put simply, operational excellence is a mindset that is embraced across an organization to maximize outcomes and positive results.

Operational excellence can be reinforced by specific tools and processes, but it's ultimately a philosophy and a cultural principle. It must be integrated into your organizational culture in order to achieve its full effects.

### Operational excellence and software

Operational excellence is a buzzword that you hear occasionally in the context of business management. It only rarely appears in conversations about software. Nonetheless, the concept has an important role to play in several respects, including making software more reliable and extensible.

# Operational Excellence and the Success of Software Deployments (cont.)

## Operational excellence and software

### UNITING DISPARATE TEAMS

One of the most pervasive challenges in modern software deployment and management is the fact that every deployment depends on multiple teams. Developers, IT operations, infrastructure engineers, and business managers all have a role to play in ensuring that an application is a success.

Getting all of these teams to work together can be a challenge. The ideal solution, according to the DevOps mantra, is to "de-silo" your organization by creating a single DevOps team that oversees development, operations and everything in between. For some companies, that approach will work, but for others, single-team DevOps is not a realistic goal. There will always be some distinction between teams and specializations.

Yet whether you have a single DevOps team, or multiple teams working toward a common goal, operational excellence can help you achieve the cultural unification necessary to help everyone advance the common good. Operational excellence serves as cultural goal shared that is shared by all teams and team members during the software development and deployment process. By making excellence part and parcel of your culture, you gain a principle that can guide all of your teams, even if they otherwise share little in common culturally (which, generally speaking, is the case; developer culture is not the same as IT culture, and certainly not the same as business culture).

### OPERATIONALIZING BEST PRACTICES

Another key benefit of operational excellence is that it helps to ensure that best practices are actually followed across the software deployment process.

Most stakeholders in software development and deployment know what they should do when performing their jobs...but they don't always do it. Your developers might churn out an inefficient function because they are under pressure to meet a deadline and need to deliver something that works, even if they know that they are not following coding best practices. Your IT team may fail to configure granular IAM permissions for a virtual server because they lack the time, even though they know that is not a best practice from a security standpoint.

Instilling operational excellence into your teams helps to prevent these types of shortcuts. When a culture of excellence prevails, your team will follow best practices, even if it takes longer.

# Operational Excellence and the Success of Software Deployments (cont.)

## Operational excellence and software

### CONTINUOUS IMPROVEMENT

Human beings are naturally inclined to desire finality. Feeling like we will never reach the end of a task is discomforting (that's why poor Sisyphus was so unhappy). Yet, this discomfort needs to be managed if your organization is to achieve a culture of continuous improvement. Continuous improvement means that your teams constantly find ways to make things better, even if they're already quite good.

For developers and IT teams, this can be a tough pill to swallow. Developers or engineers who have achieved 99.99999 reliability for their application would probably rather feel proud of what they've accomplished than be told that they need to add another 9 to that figure.

Still, striving to become even better is what continuous improvement requires. A culture of operational excellence can help ensure that your teams welcome this type of challenge rather than resent it.

### COLLECTIVE EFFORT

When it comes to software development and deployment, personal "ownership" of any part of the code or process is a risk to the organization. If only one programmer knows how a certain microservice works, or only one IT engineer has the power to create new databases, your organization will lose agility, or it could even end up with a "hostage" employee situation.

Instead, software deployment should be as collaborative as possible. Everyone within a given team should understand and (in a pinch, at least) be able to perform everyone else's job.

A culture of operational excellence helps to ensure that your teams take a collaborative approach to the way in which they build and deploy software. Instead of letting each individual own part of the process and receive personal credit for its success, operational excellence encourages team members to think of their work as a collective process with collective responsibility for the results. That's because operational excellence focuses on the success of the organization as a whole rather than individual employees or teams.

To be sure, choosing the right architectures, tools, and people will do much to help maximize the reliability and effectiveness of your software...but those are only part of the solution. You also need a culture of operational excellence to unite your various teams behind a commitment to best practices, continuous improvement, and collective pride in the applications that they build and deploy.

# Why Integrated Infrastructure Is the Key to IT Success

At first glance, your IT infrastructure might seem like your house's basement...it supports everything, but you don't pay much attention to it or spend much time or energy trying to make it beautiful. Instead, you treat it as a generic component and lavish your attention on other parts of your IT landscape (or your house, as it were).

But the fact is that infrastructure is not so simple and generic. The way you design your infrastructure, as well as your ability to integrate it with the ever-changing set of tools that you need to keep workloads running efficiently, plays a critical role in defining overall business success.

To prove this point and encourage you to give your infrastructure some much-needed love, keep reading for an overview of why well-integrated infrastructure is so crucial, and why implementing an infrastructure that integrates seamlessly with the rest of your solution stack is an important  best practice.

### What is integrated infrastructure?

By "integrated infrastructure," we mean infrastructure that seamlessly connects with the entire ecosystem of tools that you use to deploy and manage applications.

To put that into context, consider some examples of how infrastructure interfaces with tools:

- Your CI/CD pipeline needs to be able to integrate effectively with the infrastructure that hosts CI servers, testing tools, and build tools.
- Release automation software must be able to deploy seamlessly to the infrastructure that will host live applications.
- Monitoring and performance optimization tools need to work easily with whichever infrastructure hosts your workloads, since monitoring the underlying infrastructure is one critical component of application monitoring.
- Security and access-control tools must be able to apply security policies and restrictions to host infrastructure to help keep applications secure.

**The list could go on, but you get the idea: an integrated infrastructure is one that can easily interface with all of the tools you use to create, deploy, manage, and secure the workloads that you host on that infrastructure.**

# Why Integrated Infrastructure Is the Key to IT Success (cont.)

## Agnostic infrastructure vs. integrated infrastructure

You may be thinking, "It's 2019, and most infrastructure is agnostic. My tools don't care which cloud I use to host my workloads."

You'd be right that most modern, cloud-native infrastructure is designed to be agnostic toward the tools that are used with it. Although there are certainly exceptions, the majority of infrastructure solutions available today – whether they are from a public cloud vendor or local infrastructures built using standard operating systems and hypervisors – can be configured to work with pretty much any mainstream tool you throw at them.

But agnostic infrastructure is not the same as integrated infrastructure. An integrated infrastructure is one in which the connections between your tools and your infrastructure are not only possible to create, but happen seamlessly. This difference is easiest to see if you compare public cloud vendors' native tools with tools from third-party vendors that work with those clouds, but don't always integrate seamlessly with them. For example, pretty much every modern APM tool can be configured to work with AWS. But setting them up takes time, and there is no guarantee that they will always be compatible. In contrast, there is CloudWatch, AWS's native monitoring tool. It falls far short of being a complete APM solution, but it is seamlessly integrated with any AWS infrastructure. You don't need to configure or install anything to use it, and it scales automatically with your AWS footprint.

## Achieving infrastructure integration

My point here is not that you need to choose between a feature-limited native toolset and a third-party one that offers more features but does not integrate as well. Instead, you should strive to build an infrastructure and a toolset that offers the best of both worlds: rich features and tight integration.

The best way to do that is to take a toolset-first approach. By that, we mean identifying which tools you need and designing your infrastructure around them. That is preferable to the strategy that most organizations use by default, which is to choose an infrastructure (usually a cloud provider) and then figure out how to reconcile their toolset with it. Not only does this approach limit your choice and inflate your costs (because you may end up having to use native tools from your cloud vendor that are less cost-effective than third-party alternatives), but it also makes integration between your tools and your infrastructure more difficult.

So, before you go and migrate all of your workloads to one public cloud or another, identify the various tools you depend on and evaluate how well each of them integrates with the clouds (or other infrastructure solutions) available to you. You may find that you should choose an infrastructure strategy built on multi-cloud or hybrid cloud to achieve better integration with your toolsets. This type of approach will make infrastructure architecture more complex, but it enables better infrastructure integration, which delivers more value in the long run.

## Integrate for today and the future

Keep in mind, too, that the tools you use today can and probably will change in the future. You thus want to ensure that your strategy enables tight infrastructure integration over the long term.

Here again, a multi-cloud or hybrid cloud strategy is probably the best approach for maximizing flexibility and integrability, although you'll need to assess your specific needs by looking at your tools first, and then plan an infrastructure strategy that will accommodate them as they evolve and grow.

It all boils down to this: although infrastructure might seem like a mundane and largely interchangeable part of your IT landscape, the extent to which it is well integrated with your tools can make or break the overall effectiveness of your IT strategy. Don't choose infrastructure first and then shoehorn your tools to fit it. Instead, design your infrastructure architecture around the tools you need, and make sure it can continue to accommodate them as your strategy evolves.

# Optimizing AWS Performance over the Long Term

Ask most folks how to optimize performance on AWS, and they'll talk about things like "autoscaling" and "rightsizing."

Those tools are certainly part of an effective AWS performance optimization strategy. On their own, however, they are not enough to guarantee performance and reliability over the long haul.

That's because tools and processes like autoscaling and rightsizing only help you optimize the performance of applications that are running currently, or are about to be deployed. They do little to ensure that performance remains optima – especially as cloud services, cost structures, and architectures evolve.

With that challenge in mind, let's discuss how to get the most out of AWS performance over the long term.



### What is AWS performance?

Before delving into long-term AWS performance strategies, let's make clear what we mean by "performance."

A high-performing application is one that:

- Meets or exceeds user expectations regarding responsiveness and speed.
- Meets or exceeds SLA requirements for availability.
- Is able to scale seamlessly as performance requirements change.
- Does all of the above in a cost-efficient manner.

These aspects of a high-performing AWS application are important to emphasize because they reflect a holistic approach to performance. Sometimes performance is defined narrowly, in terms of application responsiveness and/or availability. Those are part of the performance equation, but in order to maximize performance over the long term, you need to think about other dimensions of performance, like the role played by scalability and cloud spend.

# Optimizing AWS Performance over the Long Term (cont.)

## Long-term AWS performance considerations

To optimize AWS performance over the long term, you must take a variety of factors into consideration.

### HANDLING AWS SERVICE CHANGES AND UPGRADES

AWS is constantly evolving. A performance strategy that works well today may not be as effective tomorrow if AWS introduces a new service or changes the features of an existing service.

The challenge here lies not just in updating your app and your cloud configuration as AWS changes how a particular service works. It also requires constantly evaluating whether the service you are using is the best fit for your application.

For example, think back about five years to when AWS Lambda (AWS's serverless computing service) appeared. At the time, if you were running your apps inside virtual servers, you might have been able to improve overall performance by moving some of the workloads to serverless functions – not because anything in your virtual servers had become less efficient, but because better efficiency opportunities opened up with the addition of a new AWS service.

Thus, being constantly aware of the various services in AWS and the opportunities they offer is key to optimizing long-term performance.

### PLAN FOR CHANGING COST STRUCTURES

AWS not only changes its menu of service offerings frequently, it also updates pricing. Since cost optimization is one key component of performance optimization, you must ensure that your cloud workloads are running in the most cost-efficient way even as cost structures change.

Staying aware of new cloud services (such as lower-cost S3 storage tiers) when they appear is one way to do this. But you also need visibility into the tremendously complex world of AWS pricing in all of its region-specific detail.

### PLAN LONG-TERM FOR OSS DEPENDENCIES

Open-source software is something else that is constantly changing. If you rely on open source code to help run your AWS applications, you need to stay on top of changes in the projects you depend on.

If you are using the same version of an open-source application or module that you built into your application when you first deployed it, you may be missing out on performance optimization opportunities that have arisen since then.

Here again, constant awareness of and visibility into the complex open-source ecosystem is essential for ensuring that you are not missing out on optimization opportunities.

# Optimizing AWS Performance over the Long Term (cont.)

## Long-term AWS performance considerations

### APPLICATION-SPECIFIC PERFORMANCE MONITORING

Every application is a special snowflake -- really -- and you need to keep that in mind when it comes to the way you track and optimize application performance.

The metrics or KPIs that work for optimizing the performance of one app may not be ideal for another. The pre-deployment testing routines that you use for applications might vary too.

So, make sure to tailor your monitoring strategy to your applications. Doing so is the only way to ensure that you have the greatest, most relevant level of visibility into application performance over the long term.

### BACKUP AND DISASTER RECOVERY

No matter how well your AWS workloads perform, sooner or later something will go wrong and they will fail. Whether the failure leads to a critical availability disruption or a mere hiccup hinges in large part on whether you have a backup and disaster recovery plan in place beforehand.

For AWS apps, an effective backup and recovery plan entails more than just backing up data to storage buckets and downloading it to rebuild servers. It should also take advantage of opportunities like the ability to create image-level backups of virtual servers and restore them automatically to EC2 if disaster strikes.

And of course, as AWS changes, you'll need to keep your backup and recovery strategy in sync.

### CHANGING SLA REQUIREMENTS

Last but not least, keep your SLAs in mind. The SLAs you have to support today could change in the future -- and by extension, so will the levels of availability and responsiveness that your users expect.

For this reason, make sure that your AWS performance strategy is capable not only of meeting the SLAs you currently have in place, but also adapting and scaling as SLA requirements become more rigid.

The bottom line: Although we tend to think of performance within the context of the present, it's really a long-tail game. Sure, you can autoscale your workloads or use rightsizing tools to help choose the best type of EC2 instance for right now. But truly maximizing performance requires you to think longer-term and be prepared to react with agility as new performance challenges and opportunities arise.

# Why AWS Cost Optimization is about More than Saving Money

Why is it important to track and monitor cloud costs? The obvious answer is that it helps you save money – and everyone wants to save money.

Yet cost plays a much larger role in your cloud journey. If you think about cost only in terms of how much you are paying and what you are getting in return, then you're missing much of the value that careful cloud cost control can provide.

Let's take a look at all of the reasons why monitoring and optimizing costs is critical for a successful AWS strategy.



**What is cost optimization?**

In AWS and other clouds, cost optimization is the art and science of identifying and addressing inefficiencies in the way you are using resources.

Those inefficiencies come in many forms. They could be an EC2 instance that offers more resources than you need for a given workload, which could therefore be replaced by a different, less costly instance.

They could be a result of using one type of application architecture, like workloads running directly on virtual machines, when another, such as serverless or containers, would deliver the same performance for less overall cost.

They could be caused by failing to take advantage of the best opportunities available for hosting a given workload. For example, you might be using S3 Standard for storage when S3 Glacier would meet your needs just as well, at a lower cost.

**By identifying these sorts of problems, you can take steps to spend less while maintaining the same level of performance for your cloud workloads. A basic best practice is to use tools like AWS Trusted Advisor to help ensure that your infrastructure is set up in a cost-efficient way from the start.**

# Why AWS Cost Optimization is about More than Saving Money (cont.)

## Cost optimization insights

But these tools only go so far, and they are not the be-all, end-all of cost optimization. They are designed simply to help choose the right type of configuration for a given workload. However, cost monitoring as a whole goes further, and delivers several other important types of insight into your AWS workloads and strategy.

### ROGUE INFRASTRUCTURE

By its nature, rogue infrastructure – meaning use of infrastructure for purposes not officially authorized by your organization – is hard to detect. If it wasn't, it wouldn't be very good at being rogue.

If you monitor cloud costs carefully, however, rogue infrastructure will have a hard time hiding. If employees spin up a virtual server in the cloud for personal use, or create storage buckets for organizational data that is not supposed to exist in the cloud, you are likely to discover it when you perform cost optimization operations designed to help identify workloads that should not be running.

Having a strong IT governance policy in place that requires all cloud-based resources to be tagged is another way to help prevent the creation of rogue infrastructure, and to make unauthorized resources easy to detect. When possible, tagging should be automated to ensure the fastest and most consistent results.

### SECURITY RISKS

Although cost optimization is certainly not the only tool you should use to help identify security issues in your cloud workloads, it provides some useful insight that other tools might lack.

For example, if you notice that your bills for a certain application or service have increased due to heavier usage without a corresponding increase in legitimate requests, it could be a sign of brute-force attacks or other unauthorized access attempts.

### NEGLECTED PROJECTS

When was the last time you evaluated each of your applications to ensure that it was running as efficiently and securely as possible?

If you don't know because you don't regularly do this, cost optimization can help ensure that you do recognize workloads that could stand to be reconfigured. If you discover that you are paying more than you expect for a given workload, or that one workload's costs are rising more quickly than those of others without a clear reason, it's probably time to take a look at that workload and assess whether it should be scaled up or down, migrated to a different type of service, or otherwise adjusted to operate more efficiently.

### OPERATIONAL INEFFICIENCIES

Finally, cost optimization can help you identify inefficiencies in the way your organization is run. That is because fees charged by AWS that you could have avoided but didn't might reflect larger inefficiencies within the organization.

For instance, consider AWS data storage costs. If your AWS cost analysis shows that you are paying early-deletion fees for data stored in S3 Glacier, you should determine why that is happening. It could be simply because you should be using a different storage class, but it could also be because employees are deleting data sooner than intended.

Saving money by being cost-efficient should always be one of your goals. But running your overall business in an efficient, transparent manner is another. AWS cost optimization can help you achieve all of this.

# Practical Approaches to Long-Term Cloud-Native Security

There is no shortage of advice out there about how to secure modern, cloud-native workloads. By now, most developers and IT engineers who work with cloud-native deployments have heard all of the mantras about DevSecOps, shift-left security, multi-layer defenses, and dynamic baselining (to name just some of the key concepts that are driving IT security conversations these days).

But, it's one thing to talk about security best practices. It's another to design a cloud strategy that makes it not only possible, but also easy to implement them. Even more challenging is planning a strategy that facilitates security best practices over the long term.

**Keep reading for tips on meeting these challenges and devising a long-term security strategy.**

### Security is a long game

It's easy to think of security as something you have to do in real time. After all, threats and attacks usually happen suddenly, and reacting to them quickly is key to preventing serious damage.

But if your security strategy centers on finding and remediating threats as they appear, you end up stuck in what is essentially a break/fix mode. You're constantly reacting, rather than being proactive.

A much more effective security strategy is one that minimizes the threats you face in the first place. And that type of strategy requires long-term planning.

Sure, you will always need to be prepared to detect some threats that you didn't anticipate, and react quickly in the event a breach occurs. No security plan is perfect, and the unexpected will sometimes still happen. But by focusing as much as possible on long-term security solutions that stop most threats from materializing, you end up with a much safer and more reliable security posture.

# Practical Approaches to Long-Term Cloud-Native Security (cont.)

## Building a long-term security strategy

What does it actually take to implement a security strategy that protects you over the long term? It all boils down to four elements: data, infrastructure, processes, and culture.

### DATA

For many organizations, data stored in the cloud is the workload that poses the greatest risk. It's the reason why there is a seemingly never-ending stream of headlines about major security breaches that involve the theft of sensitive data stored in the cloud.

Therefore, mitigating threats to data in the cloud is a critical requirement for long-term security. Some best practices in this regard include:

- **Assess your data:** Data security starts with knowing what is in your data, and how sensitive the data is. Data cataloging can help with this process, but so can tools like Amazon Macie, which helps identify personally identifiable information with cloud-based data.
- **Access control:** Access control frameworks are another crucial line of defense against cloud-based data breaches. Just remember that simply enabling IAM is not enough; you must also audit IAM policies to avoid misconfigurations that could lead to a breach (which was the lesson learned by Capital One, among others).

### INFRASTRUCTURE

These days, we tend to treat infrastructure – meaning the cloud-based and/or on-premise data centers that host workloads – as a relatively generic and interchangeable part of the solution stack. But while it is true that, generally speaking, no one cloud or data center is inherently more secure than another, the way you design your infrastructure plays a key role in your ability to secure modern workloads over the long term.

Best practices on the infrastructure front, for long-term security include:

- **Immutable infrastructure:** Embrace infrastructure platforms, such as ECS or EKS, that make it easy to deploy and update applications in an immutable way. Immutable building-blocks may not work for all use cases, but where possible, they reduce the number of variables and handoffs you have to contend with in order to deploy or update an application (and the more variables and handoffs, the greater the chance of an oversight that will create a security vulnerability).
- **Access control:** While IAM frameworks can help prevent some types of unauthorized access, you can also increase the security of your infrastructure using Access Control Lists. A best practice is to adopt a whitelisting approach that blocks all traffic by default, and allows only explicitly trusted hosts.

# Practical Approaches to Long-Term Cloud-Native Security (cont.)

## Building a long-term security strategy

- **Avoiding bloat:** Cloud-native infrastructure makes it easy to spin up virtual machines, databases, or other infrastructure components that you don't truly need, and keep them running. Doing so increases your attack surface. For that reason, identifying and shutting down unused infrastructure is critical for ensuring long-term security. (It also helps cut cloud costs, but that is another topic.)

- **Parity:** In order for developers, IT Ops, and security teams to be able to communicate and coordinate effectively, they should all be working with the same types of infrastructure environments. Collaboration becomes much more difficult if, for example, you use an on-premise data center for development, then deploy production workloads to the cloud. Avoid this by striving to create parity across your infrastructure whenever possible.

### PROCESSES

Processes are the second key ingredient in creating a long-term security plan for cloud-native workloads. Obviously, the processes you use will reflect, in part, your particular workloads and tools. But no matter your situation, your processes should be designed with the following security goals in mind:

- **Consistency:** Your IT organization may consist of different teams, each with different responsibilities and agendas. Nonetheless, to the extent possible, all teams should follow the same procedures when developing, testing, staging, and deploying applications. Doing so reduces the chances of variables or oversights that can undermine security. You can help achieve this consistency by creating and enforcing IT governance procedures across the organization.

- **Shift-left:** Processes are the part of your IT operation where shift-left security can have the greatest effect. Shift-left security means making security a priority from the start of the application delivery pipeline. Importantly, however (and this is a point that some organizations overlook), doing shift-left security correctly also means bolstering security operations at later stages of the pipeline, too. Don't just shift security left and call it a day; aim to make security a primary consideration at all stages of your delivery chain.

# Practical Approaches to Long-Term Cloud-Native Security (cont.)

## Building a long-term security strategy

### CULTURE

Culture is something that can be difficult to formalize; indeed, if you try to stuff cultural values down your employees' throats, you risk compromising the whole point of having an organic culture in place. Instead, you want to encourage your team members to naturally embrace values that promote a culture of security. Strategies that can help achieve that include:

- **Bug bounties:** Incentivize your team to look for and address security issues by hosting bug bounties for your organization's workloads. For this purpose, "bug" should be defined broadly to include not just security bugs within code, but any kind of configuration issue in your infrastructure, tools, or applications that could lead to security vulnerabilities. That way, your bug bounties can include not just developers, but all members of the IT organization.

- **Encourage shared ownership:** A culture in which each engineer "owns" a specific application, infrastructure, or process is one that limits transparency and opportunities to address security risks. Strive instead to promote shared ownership of IT assets so that each team member knows his or her work may be reviewed by others, which helps to encourage best practices and reduce security issues. In essence, these are part of the values that DevOps and DevSecOps prioritize.

By proactively designing practices that promote security, you put your organization in a position to optimize security over the long term. Planning ahead for security is the only way to escape the break/fix cycle of responding to vulnerabilities as they are discovered, which leaves you always treading water and never pushing the needle.

**You won't be able to prevent all security issues, but you can greatly reduce the number that crop up by baking security into your infrastructure, processes, and culture.**

# Conclusion

The public cloud is a complex place. The ease with which the cloud lets organizations deploy applications can belie the challenges inherent in making sure that cloud-based workloads are reliable, secure and cost-efficient. But by following the right cloud service and integration patterns, businesses can master the complexity of cloud environments in order to get the most out of their cloud strategies, even as public cloud offerings continue to evolve and expand.

**Let us help you determine which solution is right for you.**

**GET STARTED TODAY**

info@eplexity.com
888.501.5979

EPLEXITY

CXOS

aws partner network

**Premier**

## Consulting Partner

- DevOps Competency
- Migration Competency
- Public Sector Partner
- Amazon EC2 for Microsoft Windows Server
- Well Architected