HashiCorp

WHITEPAPER

Platform Teams: Best Practices

Contents

Executive Summary

Adopting the cloud demands a new operating model that transforms your cloud infrastructure and tools away from a diverse collection of ad hoc deployments to a unified central platform that provides consistency across all four layers of your cloud estate: infrastructure, security, networking, and applications. To lower costs, reduce risk, and increase development and deployment speed, enterprises turn to centralized platform engineering teams to build and enable shared services.

Depending on the organization, these platform teams may be called cloud teams, DevOps teams, SRE teams, or some other term. Regardless of the name, their mandate is to support the organization's different application teams — their customers — with standardized shared services in the cloud. This helps enterprises meet their larger digital transformation goal of speeding delivery of new business and customer value at scale. The platform team is tasked with building and deploying standardized cloud tools and systems across the organization — much the same way as a software platform is developed and managed. For each cloud layer, the goal is to build a consistent set of standardized shared workflows and services that become the system of engagement for development teams.

The platform team is typically composed of cloud engineers responsible for implementing the platform following the blueprint of a cloud operating model. Successful adoption of a cloud operating model requires that the team and the platform it creates are properly formed and defined from the start. The teams need a clear mandate on the goals for the platform they arebuilding and its ongoing role once the platform is deployed.

This paper focuses on understanding the phases of cloud adoption, the role a platform team plays, and how to set up and ensure the success of these teams and the platforms they create.

The 3 phases of cloud adoption

What does the typical cloud adoption journey look like? In most cases, cloud adoption follows a multi-phased approach, initially driven by individual development teams, but quickly requiring a more centralized organization in order to scale, control costs, and promote proper security implementations.

Phase 1: Adopting — An ad hoc approach to cloud

Typically, an organization starts by empowering its development teams to begin experimenting and transitioning to the cloud. Executives provide high-level goals and budgets and then let application developers figure out what works best. These application teams all build cloud applications, but each a little differently. They may use different cloud providers, tools, services, etc.

As organizations progress through Phase 1 of cloud adoption, this process becomes very ad hoc. While necessary to jump-start a cloud adoption program, it can also lead to chaotic implementations across the larger organization. Each team picks its own tooling, approaching in its own way, building its own pipelines, and establishing unique processes. Almost inevitably, within 12 to 18 months, this approach results in several critical problems:

- **Cost overruns:** As every team does things its own way, the result is often oversized and orphaned infrastructure. Additionally, without standardization, economies of scale aren't built into contracts and cloud costs are not optimized.
- **Security vulnerabilities:** Without proper security processes and oversight, vulnerabilities begin to appear as teams focus on their own applications and ignore larger security implications.
- **Compliance inconsistencies:** Every team sets up its own systems for compliance, creating inconsistencies and breaking compliance from one group to another as services expand across an organization.
- **Replication of work:** As teams build their own isolated systems, they often reinvent the wheel instead of reusing existing tooling and workflows.

Phase 2: Standardizing — Platform as a service

As organizations work to move beyond ad hoc adoption and enter the Standardization phase of cloud adoption, platform teams come into play. In this phase, organizations no longer find it feasible for each team to have totally unique cloud implementations. Instead, those implementations must be dictated by a central platform team supporting all the application teams with a standardized approach to cloud.

This platform team defines the systems, patterns, and workflows developers will use for all layers of the cloud. This results in a shared service model for all aspects of the cloud. The platform team also becomes a centralized resource with which other cross-organizational groups can interface. For example, the security and compliance team can now work with a single group to more easily standardize security and compliance across all the application teams.

This is a critically important phase of the cloud maturity model, as the platform team formally establishes the structure of cloud consumption across the organization.

Phase 3: Scaling — Hybrid and multi-cloud platforms

Once a functioning cloud platform is in place, organizations are ready to enter the third and final phase of cloud adoption. In the Scaling phase, platform teams take what they have learned and begin applying it across the organization's broader digital estate.

First, they extend these services from the cloud back to the organization's datacenter and SaaS applications. Starting with infrastructure tools, platform teams can apply infrastructure as code, secrets management, networking, and application staging and deployment workflows to this larger private estate. Further, they expand functionality with SaaS tools from a library of providers and integrations. Eventually, platform teams can apply cloud structures to the organization's private datacenters and other clouds. The platform becomes a consistent shared service for all developers, not just in the cloud, but also private datacenters and multi-cloud implementations. Platform teams bring order to chaos

Platform teams bring order to chaos

What's in scope for platform teams?

Understanding the optimal scope for a platform team can become a philosophical question: "How much do you want to standardize?" To find their optimal answer, organizations should ask: "What are the critical, non-functional requirements of an application?" From a pre-production pipeline perspective, every app has a common set of requirements. For example, every app needs:

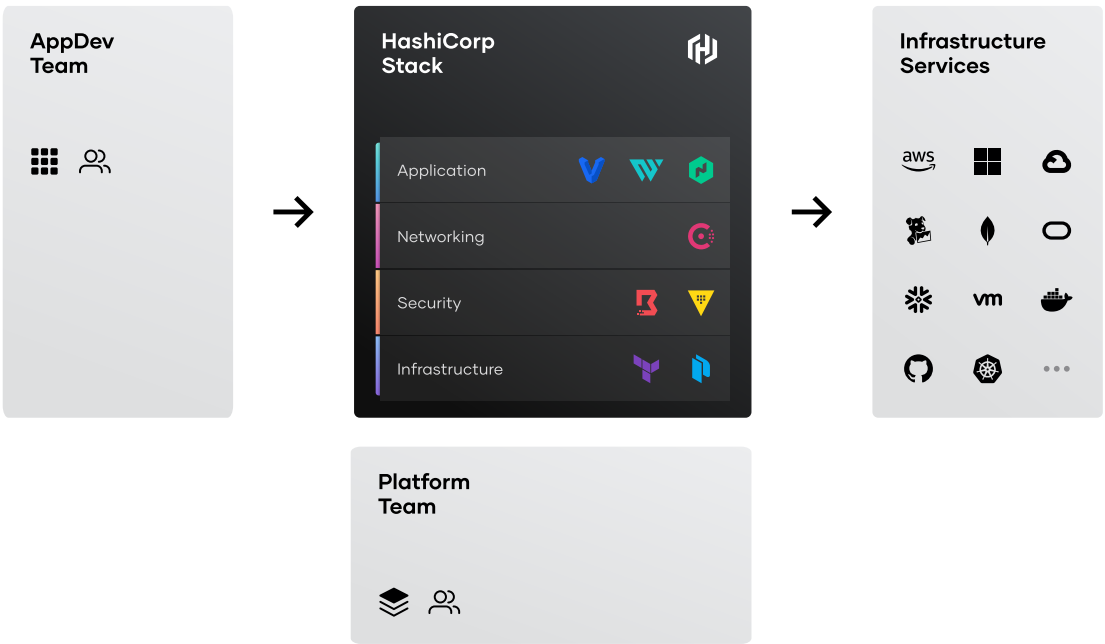
- A version control system (VCS)
- A continuous integration/continuous delivery (CI/CD) system
- Artifact management
- Static code analysis

The same is true from a production perspective. Organizations have a consistent set of requirements for every application developer, including:

- Infrastructure provisioning
- Secrets management
- Service networking
- Orchestration
- Observability

It doesn't matter what kind of app it is, these are all consistent requirements, and it's the responsibility of the platform team to deliver a consistent solution to application teams. The platform team must also determine how best to deliver the platform and its functionality to the masses. App teams want to focus on developing the application, not the underlying cloud infrastructure. Delivering as much as possible as a service lets development teams easily deploy the tools and functionality they need to support the application, without having to worry about everything else.

Platform teams accelerate cloud adoption



In addition to providing standardized shared services, platform teams:

- **Establish common sets of best practices for developers** to engage with cloud services. Then they codify these best practices into workflows and educational programs such as a Cloud Center of Excellence.
- **Serve as a single point of integration for other corporate teams** such as security, compliance, financial controls, etc. to ensure best practices are baked into relevant workflows.
- **Integrate platform as a product practices** to ensure ongoing development of the platform. Two-way communication helps iterate the continuing needs of the organization and its development teams into the larger platform.
- **Create a central system for tracking, reporting, and auditing** to provide visibility into the adoption and usage of a cloud operating model. This gives the organization insight into the platform's progress and can highlight risks around security, compliance, and spend.

Enterprise platform capabilities

An enterprise cloud platform should have consistent workflows and processes across six functional areas:

1. **Unified workflow management:** The platform must have central leadership and processes to unify common workflows across all cloud layers and teams.
2. **Reliability and scale:** Solutions created for the cloud platform must be dependable and perform consistently at scale across all levels of an organization.
3. **Policy and security:** The platform should incorporate tools to integrate policies and guardrails directly into workflows.
4. **Governance, risk, and compliance:** The platform must establish a consistent philosophy for the integration of security and compliance frameworks directly into all layers of the cloud estate.
5. **Visibility and optimization:** The platform team must build tools and dashboards to view and audit all aspects of the cloud platform to ensure consistent performance and drive optimization.
6. **Integration and APIs:** The platform team is responsible for creating tooling and integrations that give the platform the required functionality and can be easily adopted by the organization.

Platform team best practices

HashiCorp has observed a number of common best practices among organizations that have successfully leveraged a platform team to build and manage a holistic cloud platform.

Just as in cloud adoption, platform teams go through phases. Critical to the initial forming phase is a proper foundation for the platform team and the platform.

Next, platform teams build the platform and expand it across the organization. This involves potential organizational changes, as development teams move away from managing these systems and become consumers of the platform. Critically, that doesn't mean they no longer have input. Instead, development teams become one of the core customers for the platform team and their requirements continue to drive the evolution of the platform as a whole.

Finally, as the platform is deployed, platform teams drive adoption and usage while expanding the platform across an organization's entire digital estate.

For maximum impact, platform teams should focus on pragmatic approaches to each phase of platform creation.

Considerations when establishing a platform and platform team

Understanding the customers

There are many tools and services your organization can consume in the cloud. This wealth of options drives the need for a platform team and requires standardization of solutions and workflows across the larger cloud platform. That said, different teams or business requirements may require more than one solution for similar use cases. The platform team must understand and address the specific needs of the developers using the platform. The platform team uses that knowledge to lead evaluations of the products and tools that underpin the platform and maintain a pragmatic service catalog that developers can use to meet unique needs.

Platform teams can negotiate with business and technical stakeholders to establish a reasonable list of supported services balancing innovation and sustainability. The goal is to give teams the customization they need without allowing the cloud infrastructure to devolve into chaos.

Define and measure reliability at the start

From the beginning, the platform team must earn the trust of developers who depend on the platform. A reliable and predictable set of services gives developers — and executives — confidence that the platform will perform as expected during the organization's moments of truth.

To create that trust, platform teams must define reliability in terms of users' desired outcomes. Those outcomes may include increasing development team efficiency, speeding application deployments, meeting application uptime and performance targets, and ensuring consistent and secure services. Traditional service level agreements (SLAs) can demonstrate the benefits of the platform and over time, more advanced reliability engineering practices, such as service level objectives (SLOs) and error budgets can yield even greater benefits.

Build in security and compliance

Security, compliance, encryption, auditing, and other InfoSec concerns are crucial challenges. A core benefit of a cloud platform is consistency and compliance of these functions across the organization.

To deliver that result, these considerations must be deeply integrated into the platform via thoughtful implementation of automation and reference architectures. For example, platform teams can deploy standardized golden images with the latest security patches in response to critical vulnerabilities. Platform engineers can configure secrets management and credential-rotation workflows to be performed automatically on behalf of developers. Platform teams can build consistent policy guardrails into all cloud workflows, forming the foundation of self-service tools that empower developers without increasing risk. This lets individual teams focus their security and compliance efforts on their particular service rather than needing to think through the entire stack, freeing up critical resources and boosting developer productivity.

Use a maturity model to inform roadmap investments

Following a cloud maturity model helps platform teams better understand where they still need to go. A maturity model is a blueprint for an organization's larger cloud program that helps stakeholders understand where they stand in the cloud adoption journey, and also what implementation steps they need to take as they expand their cloud use cases.

A maturity model is a critical tool for platform teams building a larger cloud roadmap. For example, at the cloud security layer, organizations typically start with secrets management use cases, then move on to certificate and key rotation, and eventually graduate to more complex data protection use cases such as encryption. A maturity model helps platform teams work through each level as they build and

deploy services and tools. This makes platform evolution more sustainable and helps achieve desired business outcomes.

Organizational considerations of platform teams

Create a formal platform team structure

Platform teams should be formed with a defined structure and roles. Typically, the platform team lead would manage the team as a whole, handle the relationships with executive leadership, relevant teams, and stakeholders, and work with development team leads. The person in this role must execute the platform strategy, and also evangelize and drive adoption of the platform. Platform team leads are often empowered by an executive sponsor, who typically supervises the organization's larger cloud program. This top-down structure formalizes the business and leadership needs of the platform to the platform engineers who embody the remainder of the platform team.

The engineering side of the platform team should be a mixture of different cloud engineers with general platform and system development expertise as well as specialized knowledge in key functional areas. These can include security and compliance, networking, infrastructure, application development and deployment, system reliability, and many others. With guidance from platform team leadership, platform engineers work together to define and create the tools and systems deployed to the organization as a whole.

Finally, platform teams need strong dotted-line connections to the teams the platform supports as well as other important stakeholders. These may include security and compliance teams, networking teams, finance, operations, and so on.

Provide a delightful developer experience

It is critical to focus on the customer of the platform services: the organization's development teams. Developers may initially need to be coaxed into deploying to the new platform instead of the existing systems they used to rely on. Mandating the use of a platform seldom works. To get developers to buy into the platform, give them what they crave: self-service rapid deployment options with minimal friction. The success of the platform team and the platform as a whole often comes down to how easy and attractive it is for developers to use.

Platform teams can lower barriers to adoption and create a great developer experience by incorporating sensible defaults, prescriptive guidance for common use cases, a paved path to production, detailed easy-to-follow tutorials, and best-practice code samples. In complex

organizations, relief from administrative hurdles and bottlenecks is also a powerful draw for developers to use approved platform services.

Build internal communities through advocacy

For platform teams to truly drive value, they need to be viewed as a value-added group to the development teams they serve. It isn't enough to build the platform and simply provide documentation in isolation. To drive cloud platform adoption, platform teams need to demonstrate and promote platform usage for the right workloads and use cases. Even with the most thoughtfully built platform, there is no value if developers don't know about its assorted capabilities and go about leveraging it.

Regular platform update meetings, internal town halls, and hackathons can help platform teams create a feedback loop where developers learn about platform functionality while platform teams get insight into the developers' goals and challenges.

The ultimate goal is that as developers see the platform's benefits for themselves, they'll advocate for its use across the organization. To make that happen, platform teams need to be as transparent as possible, document new capabilities, and share roadmap updates.

Of course, that's unlikely to happen unless the platform is something worth getting excited about. If developers don't have a positive experience, it will be that much harder to keep them engaged and onboard. Platform teams get only one chance at a first impression, so they must focus on delivering great products, great experiences, and great communities.

Ensuring platform adoption and success

Continuously reduce toil and increase automation

Reduced effort and complexity for developers is a core cloud platform benefit, which means platform teams must seek to automate as much as possible. Automation increases consistency, reduces toil, and improves developer productivity.

Successful automation requires full understanding of the steps in a given workflow. Discussions with engineering teams can yield the blueprint for a given process. From there, platform teams can automate task steps into a consistent, unified workflow. The result: reduced development costs, increased deployment frequency, and faster time to market.

Enable self-service and self-led education to encourage platform adoption

A successful platform must address all layers of the cloud estate, including infrastructure, security, networking, application deployments, observability, backing services, and much more. This can seem daunting to developers who have to learn new ways to manage all of these areas. Platform teams can help mitigate these concerns by creating a self-service portal with documentation of all aspects of the platform and its components.

This crucial step is often overlooked by platform teams focused on the platform instead of how development teams will actually use it. Platform engineers must put themselves in the shoes of their users and provide a thoughtful developer experience. This will win mindshare and encourage the use of the core platform standard patterns that will reap the core benefits of greater efficiency and more secure systems, and speed the delivery of new applications and services. And, in addition to the already discussed user groups and community, this will allow developers to learn and adopt at their own speed and in their own way.

Optimize costs with showbacks and chargebacks

Platform teams are uniquely positioned to make cloud consumption more efficient. Shared resources, automation, and standardization, for example, help contain server sprawl. When building the platform, platform teams can bake cloud optimization policies into the workflows and services they deliver to developers.

Just as important, cost optimization comes when individual consumers understand how much they are actually spending. Platform teams can implement resource tagging to associate consumption services with a given team or business unit. This practice can provide cost transparency (showbacks) or be used to draw down internal budget allocations (chargebacks). Clear, transparent usage reports make budget allocation and cost-optimization discussions much easier, helping teams identify and eliminate cloud waste.

Evaluate platform success

Cloud platforms need the right tools to properly evaluate and report on outcome-oriented metrics, including:

- Platform team-related metrics:
  - Platform stability and reliability
  - Security workflow efficiency
  - Continued development and deployment of new features and functionality
  - Success of larger business priorities and goals
  - Platform usage metrics
  - Developer survey and satisfaction scores
- Developer-related metrics:
  - Developer onboarding efficiency
  - Release frequency and acceleration over time
  - Cost optimization
  - Increased application revenue
  - Reduction in security incidents

These types of metrics indicate whether the platform team is performing its job supporting the platform as well as whether developers are using it and reaping its benefits.

Align goals to business objectives and outcomes

At the end of the day, a platform team is an internal service provider. With this in mind, the cloud platform approach must align with the larger organization's desired business outcomes, not just the asks of the development teams. The platform team should regularly meet with executive leadership to ensure that the platform fits both current and future business priorities.

These meetings should track how the metrics defined above match the previously defined platform and development goals. Continuous evaluation of key metrics and business alignment makes it easier to evaluate the current state of the platform, see what is working and what could be better, and update the roadmap for future development.

The value of platform teams

Leveraging a platform team helps unlock the fastest path to value in a modern multi-cloud environment. That value includes development velocity, cloud and resource optimization, overcoming skills gaps, and hardening the organization's security posture at every level of the cloud. Organizations using a platform team to deploy a cloud operating model can expect multiple positive outcomes:

Increased efficiency to reduce costs

- Lower costs through both cloud and tool optimization policies that reduce unnecessary cloud usage and help avoid spending on redundant legacy products.
- Reused expertise and the leveraging of a service catalog with a set of common workflows for development teams to use and easily apply across all environments.

Reduced risk

- Integrated policy and governance tooling matches the speed of delivery with compliance to manage risk.
- Embedded security and compliance requirements within the platform itself accelerate product deployments.

Faster development

- Shared services focused on application delivery velocity allow software to ship more rapidly with minimal risk.
- Embracing DevSecOps and other agile practices eliminates legacy ticketing systems that slow deployment. Abstracting core functionality away from development teams through pre-approved templates and automation of core infrastructure workflows speeds application development.

Conclusion

Cloud computing is the most fundamental shift in computing over the last 20 years. But while the cloud promises dramatic advances in how organizations innovate, respond to market trends, and connect with their customers and employees, it also requires significant changes in how applications are built, deployed, and managed.

Creating a platform team to manage this change according to a cloud operating model can help guide this process. Platform teams encourage and centralize consistency, reliability, and efficiency by delivering shared services across each layer of the cloud estate. Leveraging the best practices outlined in this paper can help organizations standardize their cloud adoption journey, create a detailed roadmap for the future, and collect the knowledge they need to make the proper decisions at the correct time.

Through the use of a cloud operating model managed by a platform team, teams will succeed in building a cloud platform that not only meets the needs of the organization but is also universally adopted and deployed across the entire organization. This will enable product teams to deliver new business and customer value efficiently at a lower cost, with reduced risk and increased speed.

The HashiCorp suite of products provides solutions for each layer of cloud infrastructure, enabling platform teams to successfully lead the shift to a cloud operating model. For more than a decade, HashiCorp has partnered with thousands of organizations around the world to successfully implement a cloud operating model. Our portfolio of community and commercial products has been downloaded more than 450 million times in the last year alone, and the HashiCorp ecosystem includes hundreds of partners and more than 3,000 integrations.

Learn more about how HashiCorp can help your organization leverage platform teams to implement a cloud operating model at www.hashicorp.com

HashiCorp