

# Unlocking the Cloud Operating Model on Microsoft Azure for Financial Services

Achieving the fastest path to value in a modern, hybrid-cloud datacenter



Authored by HashiCorp

# Executive Summary

For most enterprises, digital transformation efforts mean delivering new business and customer value more quickly, and at very large scale. The financial service industry is no different. However, as with other regulated industries, there are additional considerations to account for throughout a digital transformation. **Security is critical to financial institutions' success – customers rely on industry-leading security practices, regulatory bodies expect visibility and auditability, and modern datacenter teams require consistency and predictability.**

For the financial service industry, digital technologies impact how customers interact with financial service organizations. For example, in banking there has been a shift from using traditional physical branches to instead completing tasks, such as cashing a check, using a Bank's online and mobile applications. This means a bank's customers can now complete banking tasks in minutes and from almost anywhere rather than having to take time out of their day to visit a physical branch. Financial service organizations differentiate themselves through engagements that provide speed and convenience while not sacrificing the trust, security, and integrity required to be a top global financial organization.

The cloud is an inevitable part of this shift as it presents the opportunity to rapidly deploy on-demand services with limitless scale.

To unlock the fastest path to value of the cloud, Financial Service organizations must consider how to industrialize the application delivery process across each layer of the cloud: embracing the cloud operating model, and tuning people, process, and tools to it. This whitepaper looks at four specific areas of the Cloud Operating Model across infrastructure, security, networking, and application delivery using Microsoft Azure.

## The Shift from Static to Dynamic

The transition to cloud, and heterogeneous environments, is a generational transition for IT. This transition means shifting from largely dedicated servers in a private datacenter to a pool of compute capacity available on demand. The cloud presents an opportunity for speed and scale optimization for new “systems of engagement” – the applications built to engage customers and users. These new apps are the primary interface for the customer to engage with a business, and are ideally suited for delivery in the cloud as they tend to:

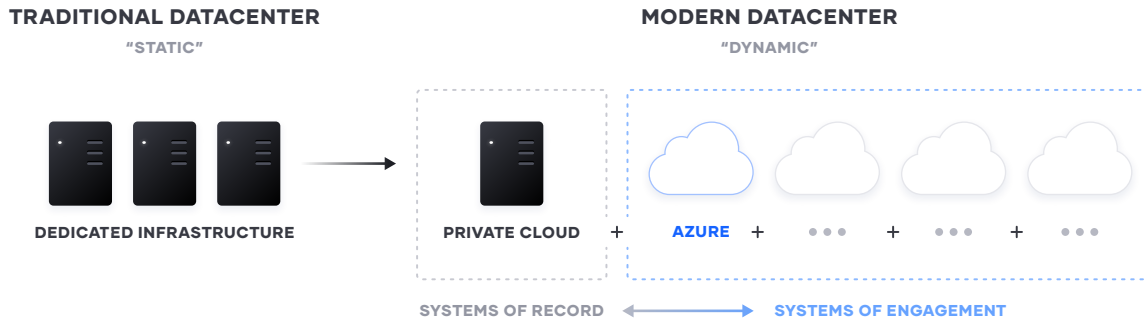
- **Scale.** Have dynamic usage characteristics, needing to scale loads up and down by orders of magnitude during short time periods.
- **Ephemerality.** Be under pressure to quickly build and iterate. Many of these new systems may be ephemeral in nature, delivering a specific user experience around an event or campaign.
- **Heterogeneity.** Organizations are composing infrastructure from cloud service providers as well as other value-add services. This means any topology of infrastructure has multiple providers to be provisioned when infrastructure is provisioned.
- **Security and Compliance.** Maintaining security and compliance without compromise while still moving swiftly.

For most enterprises and financial service organizations, these systems of engagement must connect to existing “systems of record” – the core business databases and internal applications, which often continue to reside on infrastructure in existing data centers. As a result, enterprises end up with a hybrid – a mix of multiple public and private cloud environments.

One challenge that large customers in financial services face is the compounded years of applications and technical debt. Most financial services organizations are not “born in the cloud” – they have existing applications that have evolved over the years and run their business. They feel threats by “born in the cloud” upstarts pushing customer experience in financial services and providing Uber-like disruption in core business areas such as payment processing in banking.

Adopting new technologies throughout the decades has allowed the financial services sector to automate manual interaction and let our financial markets flourish and grow. With the shift from a static to dynamic environment, replacing these manual steps with more and more automation has gotten us to more volume than was previously thought possible, along with larger revenues and increased efficiency.

---



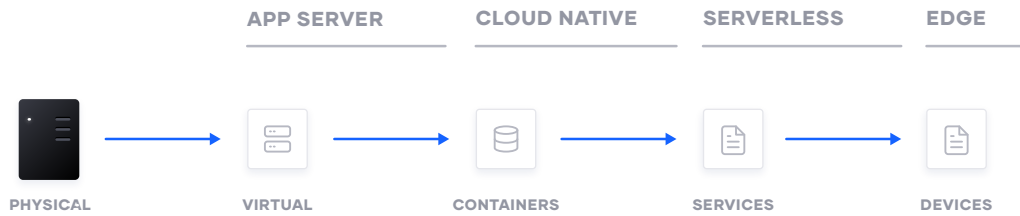
## Unlocking the Cloud Operating Model on Azure for Financial Services

As the implications of the cloud operating model impact teams across infrastructure, security, networking, and applications, we see a repeating pattern amongst finserv customers of establishing central shared services – centers of excellence – to deliver the dynamic infrastructure necessary at each layer for successful application delivery. This is especially crucial for financial services because of the importance of delivering a reliable customer experience.

When working with financial services customers on Microsoft Azure, cloud implementation is an iterative process for migrating and modernizing the digital estate, aligned with targeted business outcomes and change management controls. During each iteration, workloads are migrated or modernized in alignment with the strategy and plan. Decisions regarding IaaS, PaaS, or hybrid are made during the assess phase of the Migrate methodology to optimize control and execution. Those decisions will drive the tools used during each iteration of the migration phase within the same methodology.

The challenge for most enterprises then is how to deliver these applications to the cloud with consistency while also ensuring the least possible friction across the various development teams.

Compounding this challenge, the underlying primitives have changed from manipulating Virtual Machines in a self-contained environment, to manipulating cloud ‘resources’ in a shared environment. Enterprises then have competing operational models to maintain their existing estate, while developing the new cloud infrastructure.











For cloud computing to work, there needs to be consistent workflows that can be reused at scale across multiple cloud providers. This requires:

- Consistent instruction sets for provisioning
- Identity for security and for network connections
- Privileges and rights so they can be deployed and run

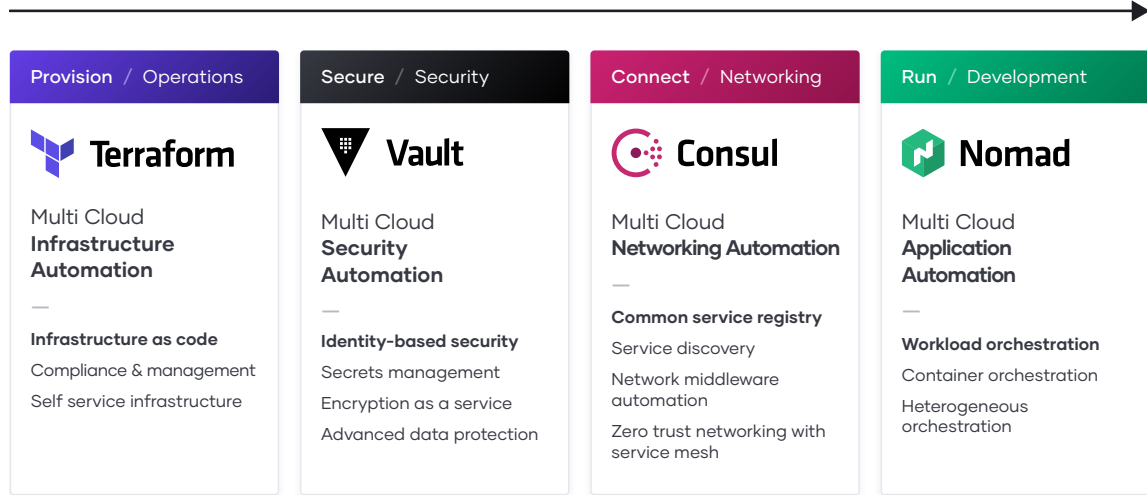
## Implications of the Cloud Operating Model

The essential implication of the transition to the cloud is the shift from “static” infrastructure to “dynamic” infrastructure: from a focus on configuration, and management of a static fleet of IT resources, to provisioning, securing, connecting, and running dynamic resources on demand. Not only is our infrastructure dynamic, our secret information, the methods used to encrypt, protect, and access can also be dynamic, giving us a higher level of confidence, and a much smaller attack surface.

	STATIC	DYNAMIC
 <b>Run</b>	Dedicated Infrastructure	 Scheduled across the fleet
 <b>Connect</b>	Host-based Static IP	 Service-based Dynamic IP
 <b>Secure</b>	High trust IP-based	 Low trust Identity-based
 <b>Provision</b>	Dedicated servers Homogeneous	 Capacity on-demand Heterogeneous

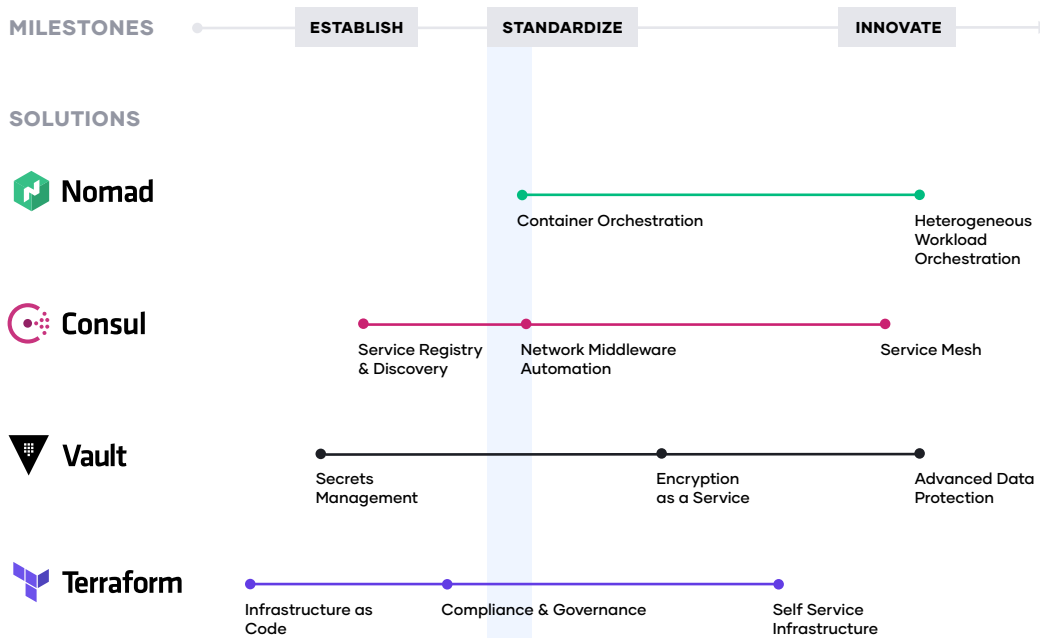
Together, Microsoft and HashiCorp tools come together to help finserv customers align on a clear strategy and plan to guide cloud adoption, implementation, and usage. As teams adopt shared services across the cloud operating model, IT velocity increases; which in turn accelerates the cloud maturity.

## EXPANDING USE OF THE HASHICORP STACK INCREASES MATURITY AND VELOCITY FOR OUR CUSTOMERS



The typical journey we have seen financial services customers adopt, as they unlock the cloud operating model, involves three major milestones;

- 1. Establish the cloud essentials** - As you begin your journey to the cloud, the immediate requirements are provisioning the cloud infrastructure typically by adopting infrastructure as code and ensuring it is secure with a secrets management solution. These are the bare necessities that will allow you to build a scalable and truly dynamic cloud architecture that is futureproof.
- 2. Standardize on a set of shared services** - As cloud consumption starts to pick up, you will need to implement and standardize on a set of shared services so as to take full advantage of what the cloud has to offer. This also introduces challenges around governance and compliance as the need for setting access control rules and tracking requirements become increasingly important.
- 3. Innovate using a common logical architecture** - As you fully embrace the cloud and depend on cloud services and applications as the primary systems of engagement, there will be a need to create a common logical architecture. This requires a control plane that connects with the extended ecosystem of cloud solutions and inherently provides advanced security and orchestration across services and multiple clouds.



## Customer Example: Capital One

**Capital One** wanted to modernize its customer engagement through mobile channels. With over a dozen business groups, change was difficult. A new shared technology group was created and tightly aligned to the credit card business unit. This “sapling” hired the people, defined the process, and adopted the tools needed for a modern delivery platform. That platform has expanded to support many internal customers.

Capital One is another great case study. Their CEO famously announced their commitment to move to a public cloud, but their initial attempt at adoption floundered because of the balance of priorities between transformation and business continuity. After 18 months of little success, they created a new dedicated business group, which was to be the shared technology group, or “sapling”.

This new group was aligned to the credit card group, as one of the more progressive groups trying to get to cloud. Ultimately they helped deliver the new mobile banking platform running in the cloud using the HashiCorp tools. This shared technology group then broadened their focus to onboard new applications and business groups<sup>1</sup>.

<sup>1</sup> Running a Terraform Environment at Scale: <https://www.youtube.com/watch?v=3JVGSq7QIS0>

# Step 1: Infrastructure Automation with Terraform

The foundation for adopting the cloud in any financial service institution is infrastructure provisioning. HashiCorp Terraform is the world's most widely used cloud provisioning product and provides infrastructure automation to many of the top global financial organizations such as [Capital One](#), [JPMorgan Chase](#), [Deutsche Börse Group](#), and [TMX Group](#).

Terraform uses infrastructure as code to provide infrastructure automation for provisioning, compliance, and management of cloud infrastructure, including Microsoft Azure. HashiCorp and Microsoft have partnered to develop the Microsoft Azure provider for Terraform. The Azure provider for Terraform provides lifecycle management of Microsoft Azure using the Azure Resource Manager APIs maintained by the Azure team at Microsoft and the Terraform team at HashiCorp. The Azure provider has over 30 million downloads and has become the standard for how the majority of financial service organizations provision their infrastructures in Azure.

To achieve shared services for infrastructure provisioning, financial service organizations should start by implementing collaborative infrastructure as code practices, and then layering compliance and governance workflows to ensure appropriate controls for reproducible infrastructure.

## Collaborative Infrastructure as Code

The first goal of a shared service for infrastructure provisioning is to enable the delivery of infrastructure as code, providing Cloud Operations teams a way to plan and provision resources inside infrastructure automation workflows.

Cloud operations teams can create Terraform templates that express the configuration of services from Microsoft Azure and many other ISV providers to include monitoring agents, application performance monitoring (APM) systems, security tooling, DNS, Content Delivery Networks, and more. Once defined, the templates can be provisioned as required in an automated way. In doing so, Terraform becomes the lingua franca and common workflow for teams provisioning resources.

[Terraform Cloud](#) and [Enterprise](#) provide collaboration capabilities including remote operations, connecting with version control, private module registries to publish reusable modules, and more.

## HashiCorp and GitHub. Better Together

Integrating with GitHub helps realize the value of version-controlled infrastructure with Terraform. In addition to providing a single, familiar view where Terraform users can see the status and impact of their changes, the integration also brings about continuous integration and testing for infrastructure

---



changes. The consistent GitHub workflow pairs well with HashiCorp's goals of providing a technology-agnostic workflow for provisioning, securing, and running any infrastructure for any application.

### **Enterprise-Ready**

Streamline operations and provision any infrastructure more securely and efficiently with Terraform Enterprise. Centralize infrastructure deployment within one workflow and provision, govern, and audit any environment.

### **Compliance and Management**

For financial service organizations, there is also a need to enforce policies for security, compliance, and operation best practices: what type of infrastructure is created, how it is used, and which teams get to use it, does it follow compliance policies, does it expose security concerns?

[Terraform Cloud](#) and [Enterprise](#) provide capabilities to standardize provisioning practices across teams and eventually an entire organization. Capabilities include access controls, policy as code enforcement, and audit.

HashiCorp's Sentinel policy as code framework provides compliance and governance without requiring a shift in the overall team workflow. Because Sentinel defines policies as code, it enables automation, collaboration, and comprehension for DevSecOps. Without policy as code, organizations resort to using a ticket-based review process to approve changes. This results in developers waiting weeks or longer to provision infrastructure and becomes a bottleneck. Policy as code allows us to solve this by splitting the definition of the policy from the execution of the policy.

Centralized teams codify policies enforcing security, compliance, and operational best practices across all cloud provisioning. Automated enforcement of policies ensures changes are in compliance without creating a manual review bottleneck.

### **Benefits of Terraform with Azure**

Leveraging Terraform on Azure empowers you to increase productivity, reduce risk, and accelerate cloud adoption.

#### **Increase Productivity**

Terraform's template-based configuration files enable you to define, provision, and configure Azure resources in an automated, repeatable, and predictable manner reducing the time to provisioning infrastructure from weeks down to minutes:

---

- Deploys the same template multiple times to create identical development, test, and production environments
- Reduces the cost of development and test environments by creating them on-demand

### **Reduce Risk**

As a resource topology becomes complex, understanding the meaning and impact of infrastructure changes can be difficult. Terraform enables users to validate and preview infrastructure changes before being applied. This can minimize the risk of errors during provisioning that require debugging, security exposures, non-compliance, and costly over-provisioning.

- Team members can collaborate more effectively by quickly understanding proposed changes and their impact
- Unintended changes can be caught early in the development process
- Lowers the potential for human errors while deploying and managing infrastructure

### **Accelerate Azure, Adoption, and Agility**

Deploy infrastructure to Azure and other ISV providers with a consistent workflow and infrastructure as code approach.

- Remove the need to learn and use multiple tools to provision
- Provide a common catalog of validated and approved infrastructure topologies to make it simple for application developers to provision their infrastructure as needed

### **Customer Example: Barclays**

Barclays achieved major gains for teams using Terraform. Once they rolled it out to the organization and required all deployments to go through a pull request process controlled by Global Risk & Compliance their agility went back to what it was prior. The tool didn't change, but their workflow did - enter policy as code<sup>2</sup>.

<sup>2</sup> Using Terraform Enterprise and Chef to enable Continuous Deployment at Barclays: <https://www.youtube.com/watch?v=MU6cW2mgAQA>

Sure, there are great tools that can help you get a jump start. But, adopting a piece of technology isn't going to change the systemic problems that your organization faces when charting your modernization strategy.

It's about modernizing behaviors, not applications. Let's imagine you're having to answer questions from your executive leadership, and the local/federal government is asking around how security postures were compromised, costs are going unchecked, and data is flowing in areas with little to no governance... You're not talking about applications "modernization".

Instead, you're talking about a fundamental shift in how you're addressing security in a hybrid environment, mitigating costs structures through financial operations, and implementing new practices for handling any and all data.

This isn't about a tool. It's about how your teams are beginning to think about re-shaping operations to manage the complexity of your infrastructure & applications. Some are sitting in a cloud, some on-premises, oftentimes bouncing between the two.

## **Step 2: Building a Zero Trust Security Model with HashiCorp Vault and Azure**

The challenge for the financial services industry is both delivering a positive customer experience while meeting all of the strict security requirements of the industry. HashiCorp Vault and Azure can help financial services customers build a secure cloud environment.

### **Secrets, Encryption, Protection.**

Most organizations today have the issue of secret sprawl. These secrets include database passwords, certificates, and private keys that should be constantly protected. However, this should not impact the speed and reliability with which code is shipped. Working with Microsoft, HashiCorp launched Vault with a number of features to make secrets management easier to automate in Azure cloud.

Vault offers a wide array of Secrets Engines that go far beyond just basic K/V management. Vault Secrets Engines can manage dynamic secrets on certain technologies like Azure Service Principles, Databases, and Datastores. These secrets are both time and access bound, which often eliminates the need to rotate secrets. Dynamic secrets help reduce the blast damage of any leaked secrets or compromised systems because every authenticated entity will have a unique set of credentials.

This section of the cloud operating model will cover five key areas;

- Hybrid Security
- Encryption as a service
- Secrets Management
- Advance data protection
- Vault and Azure integrations

### **Hybrid Security**

Dynamic cloud infrastructure means a shift from host-based identity to application-based identity, with low- or zero-trust networks across multiple clouds without a clear network perimeter. In the traditional security world, we assumed high trust internal networks, which resulted in a hard shell and soft interior. With the modern “zero trust” approach, we work to harden the inside as well. This requires that applications be explicitly authenticated, authorized to fetch secrets and perform sensitive operations, and tightly audited. HashiCorp Vault enables teams to securely store and tightly control access to tokens, passwords, certificates, and encryption keys for protecting machines and applications. This provides a comprehensive secrets management solution. Beyond that, Vault helps protect data at rest and data in transit. Vault exposes a high-level API for cryptography for developers to secure sensitive data without exposing encryption keys. Vault also can act as a certificate authority to provide dynamic, short-lived certificates to secure communications with SSL/TLS. Lastly, Vault enables a brokering of identity between different platforms, such as Active Directory on premises and other IAM services to allow applications to work across platform boundaries.

To achieve shared services for security, IT teams should enable centralized secrets management services, and then use that service to deliver more sophisticated encryption-as-a-service use cases such as certificate and key rotations, and encryption of data in transit and at rest.

### **Encryption as a Service**

Using Vault as a basis for encryption-as-a-service solves difficult problems faced by security teams such as certificate and key rotation. Vault enables centralized key management to simplify encrypting data in transit and at rest across clouds and data centers. This helps reduce costs around expensive Hardware Security Modules (HSM) and increases productivity with consistent security workflows and cryptographic standards across the organization.

Enterprises need to encrypt application data at rest and in transit. Vault can provide encryption-as-a-

---

service to provide a consistent API for key management and cryptography. This allows developers to perform a single integration and then protect data across multiple environments.

While many organizations provide a mandate for developers to encrypt data, they don't often provide the "how" which leaves developers to build custom solutions without an adequate understanding of cryptography. Vault provides developers a simple API that can be easily used while giving central security teams the policy controls and lifecycle management APIs they need.

### Secrets Management: Secure Dynamic Infrastructure Across Clouds and Environments

The shift from static, on-premise infrastructure to dynamic, multi-provider infrastructure changes the approach to security. Security in static infrastructure relies on dedicated servers, static IP addresses, and a clear network perimeter. A dynamic infrastructure is defined by ephemeral applications and servers, trusted sources of user and application identity, and software-based encryption.

Static Infrastructure	Dynamic Infrastructure
Datacenters with inherently high-trust networks with clear network perimeters.	Multiple clouds and private data centers without a clear network perimeter.
Traditional Approach	Vault Approach
<ul style="list-style-type: none"><li>• High trust networks</li><li>• A clear network perimeter</li><li>• Security enforced by IP Address</li></ul>	<ul style="list-style-type: none"><li>• Low-trust networks in public clouds</li><li>• Unknown network perimeter across clouds</li><li>• Security enforced by Identity</li></ul>

The first step in cloud security is typically secrets management: the central storage, access control, and distribution of dynamic secrets. Instead of depending on static IP addresses, integrating with identity-based access systems such as Azure AAD to authenticate and access services and resources is crucial.

Vault uses policies to codify how applications authenticate, which credentials they are authorized to use, and how auditing should be performed. It can integrate with an array of trusted identity providers such as cloud identity and access management (IAM) platforms, Kubernetes, Active Directory, and other SAML-based systems for authentication.

Vault then centrally manages and enforces access to secrets and systems based on trusted sources of application and user identity.

Enterprise IT teams should build a shared service that enables the request of secrets for any system through a consistent, audited, and secured workflow.

### **Advanced Data Protection**

Organizations moving to the cloud or spanning hybrid environments still maintain and support on-premise services and applications that need to perform cryptographic operations, such as data encryption for storage at rest. These services do not necessarily want to implement the logic around managing these cryptographic keys, and thus seek to delegate the task of key management to external providers. Advanced Data Protection allows organizations to securely connect, control, and integrate advanced encryption keys, operations, and management between infrastructure and Vault Enterprise, including automatically protecting data in MySQL, MongoDB, PostgreSQL, and other databases using transparent data encryption (TDE).

For organizations that have high security requirements for data compliance (PCI DSS, HIPAA, etc), protecting data, and cryptographically-protecting anonymity for personally identifiable information (or PII), Advanced Data Protection provides organizations with functionality for data tokenization, such as data masking, to protect sensitive data, such as credit cards, sensitive personal information, bank numbers, etc.

### **Vault and Azure-Specific Integrations**

Azure users can leverage all of these HashiCorp Vault features to automate their secrets management and retrieval through Azure specific integrations. First and foremost Vault can be automatically unsealed using KMS keys from Azure Key Vault. Next, MSI credentials can be used to authenticate systems and applications preventing the need to distribute initial access credentials. Lastly, Vault can dynamically generate Azure Service Principals and role assignments. This allows users and applications off-cloud an easy method for generating flexible time and permission bound access into Azure APIs.

If you would like a quick way of testing out Vault in Azure, [this GitHub repo](#) contains all the code to create a Vault environment in Azure including all instructions on how to obtain Terraform, run it, connect to your Azure instance and run the Vault commands. This is a great way to learn the concepts covered here with a low barrier to entry.

More information on HashiCorp Vault and How Microsoft Azure works with the HashiCorp Product Suite can be found on the Integrations page - <https://www.hashicorp.com/integrations/microsoft?product=vault>

# Step 3: Meshing together. An Overview of Consul on Azure

## What is Consul?

HashiCorp Consul is a distributed, highly available service mesh system that can run atop your infrastructure in Azure, including in multiple regions around the world. In addition to allowing services to discover each other, Consul allows you to monitor cluster health with built-in health checks, as well as serves as a distributed Key-Value store that you can utilize for a number of purposes.

## Why Consul: Modern Applications Discovery Challenges

Continuing evolution towards widely distributed software systems and microservices-based architectures brings with it multiple interesting challenges, with one of the main ones being “how do you keep track of all your deployed services?” For example, how does Service A deployed into the cloud (web service) know how to find Service B (a database)? In the static infrastructure world, you would hard-code the address of Service B into Service A’s configuration, and would need to redeploy Service A if Service B moves to a different server or datacenter. However, the downtime associated with identifying and resolving issues associated with service relocation is not acceptable in today’s rapidly changing business landscape.

This is where Consul deployed onto a modern hybrid cloud platform like Microsoft Azure can help. In addition to providing a service discovery mechanism, Consul allows for configuration changes to services deployed in Azure (such as feature flag updates) to be propagated rapidly, avoiding the inconsistent state that could bring distributed systems down.

## The Building Blocks: Agents & Servers

The basic building blocks of Consul are agents and servers. Agents allow services to announce their existence, while Consul servers collect the information shared by the agents and serve as a central repository of information about the services. Agents run on the nodes providing services and are responsible for health checking the service, as well as the node itself. To discover services, you can query either the agent or the server, with agents forwarding queries to servers automatically. You use DNS or HTTP for discovery.

## Consul, Containers, and AKS

Consul can also successfully extend modern container management platforms such as Azure Kubernetes Services (AKS) and Azure Service Fabric. While both Kubernetes and Service Fabric provide their own service discovery and health checking mechanisms, Consul allows those platforms to integrate with services that reside outside of their management boundary. For example, a web service or a database running outside of the Kubernetes cluster, and even potentially in on-prem data center, can be configured to be discoverable by services deployed on Kubernetes via Consul.

## Empowering Customers Together

“The visibility, transparency, and control we have with Consul eliminates so many of the service discovery and connectivity obstacles that used to prevent us from working as quickly and efficiently as we wanted... Consul lets us spread more than 200 microservices over several AKS clusters. Each AKS cluster connects to a local Consul client, which feeds into a Consul cluster that forms a larger service discovery mesh that allows us to find and connect services in a matter of minutes with minimal effort.”

– Sriram Govindarajan, Principal Infrastructure Engineer, Mercedes-Benz Research & Development (MBRDNA)

### The Benefits of a Single Control Plane

- Consul provides the control plane for multi and hybrid-cloud networking.
- Centrally control the distributed data plane to provide a scalable and reliable service mesh
- Automate centralized network middleware configuration to avoid human intervention
- Provide a real-time directory of all running services to improve application inventory management
- Enable visibility into services and their health status to enhance health and performance monitoring
- Automate lifecycle management of certificates which can be issued by 3rd party Certificate Authority
- Provide unified support across a heterogeneous environment with different workload types and runtime platforms

### Leveraging HashiCorp Consul with Azure

- Enable services running in any Azure region or on-premise environment to discover one another quickly and efficiently.
- Reduce deployment time of applications using Consul’s dynamic load balancing features with existing middleware (like F5, NGINX, or HAProxy).
- Enhance the Kubernetes experience by leveraging AKS and Consul’s service mesh capabilities.



## Step 4: Hybrid-Cloud Application Delivery

Finally, at the application layer, new apps are increasingly distributed while legacy apps also need to be managed more flexibly. HashiCorp Nomad provides a flexible orchestrator to deploy and manage legacy and modern applications, for all types of workloads: from long-running services, to short-lived batch, to system agents. To achieve shared services for application delivery, IT teams should use Nomad in concert with Terraform, Vault, and Consul to enable the consistent delivery of applications on cloud infrastructure, incorporating necessary compliance, security, and networking requirements, as well as workload orchestration and scheduling.

### Mixed Workload Orchestration

Many new workloads are developed with container packaging with the intent to deploy to Kubernetes or other container management platforms, but many legacy workloads will not be moved onto those platforms, nor will future Serverless applications. Nomad provides a consistent process for deployment of all workloads from virtual machines, through standalone binaries, and containers, and provides core orchestration benefits across those workloads such as release automation, multiple upgrade strategies, bin packing, and resilience. Nomad provides the same consistent workflow at scale in any environment. Nomad is focused on simplicity and effectiveness at orchestration and scheduling, and avoids the complexity of platforms such as Kubernetes that require specialist skills to operate and solve only for container workloads.

### High Performance Compute

Nomad is designed to schedule applications with low latency across very large clusters. This is critical for customers with large batch jobs, as is common with High Performance Computing (HPC) workloads. In the million container challenge, Nomad was able to schedule one million instances of Redis across 5,000 machines in three data centers, in under 5 minutes. Several large Nomad deployments run at even larger scales.

### Multi-Datacenter Workload Orchestration

Nomad is multi-region and multi-cloud by design, with a consistent workflow to deploy any workload. As teams roll out global applications in multiple data centers, or across cloud boundaries, Nomad provides orchestrating and scheduling for those applications, supported by the infrastructure, security, and networking resources and policies to ensure the application is successfully deployed.

# In Conclusion

The transition to cloud and hybrid cloud environments is a generational transition for IT. For most enterprises, this transition means shifting from a static to dynamic environment and focusing on developing your Azure migration plan around people, process, and tools to drive the right outcomes for your organization.



## Assets

- Involve stakeholders
- Calculate your TCO
- Discover & evaluate apps



## Migrate

- Select migration strategy
- Apply migration strategy
- Find recommended tools



## Optimize

- Analyze your costs
- Save with offers
- Reinvest to do more



## Secure and Manage

- Security
- Data protection
- Monitoring

On-Premises



Azure

## Aligning People, Process, and Tools to the Azure migration Journey;

### 1: Create a cloud migration plan

By establishing your cloud migration priorities and objectives before you start planning, you can ensure a more successful migration. Automated cloud migration tools will also provide insights into your environment and dependencies to build out your cloud migration project plans.

### 2: Involve stakeholders

Reach out to key people throughout the organization - this should include representation from both IT and the involved business owners. Getting everyone's engagement and support before you migrate will lead to a smoother, faster cloud migration process that meets everyone's goals.

### 3: Calculate your TCO

Evaluate the potential cost savings of migrating to Azure by calculating and comparing your total cost of ownership (TCO) for Azure with that of a comparable on-premises deployment. Leverage the TCO documentation for leveraging HashiCorp tools to help accelerate Azure migration plans;

<https://www.terraform.io/docs/cloud/cost-estimation/azure.html>

#### **4: Discover and evaluate apps**

To start any migration, compile an inventory of the physical and virtual servers in your environment. While your current management tools may provide a good representation of the hundreds—maybe thousands—of applications your organization is running, you need an inventory mechanism that can feed data into subsequent steps.

##### **People, Process, and Tools:**

###### **People: Shifting to hybrid-cloud skills**

- Reuse skills from internal data center management and single cloud vendors and apply them consistently in any environment.
- Embrace DevSecOps and other agile practices to continuously deliver increasingly ephemeral and distributed systems.

###### **Process: Shifting to self-service IT**

- Position Central IT as an enabling shared service focused on application delivery velocity: shipping software every more rapidly with minimal risk.
- Establish centers of excellence across each layer of the cloud for self-service delivery of capabilities.

###### **Tools: Shifting to dynamic environments**

- Use tools that support the increasing ephemerality and distribution of infrastructure and applications and that support the critical workflows rather than being tied to specific technologies.
- Provide policy and governance tooling to match the speed of delivery with compliance to manage risk in a self-service environment.

## **About HashiCorp and Microsoft - Better together**

HashiCorp and Microsoft are longstanding partners in the cloud infrastructure community. In 2017, Microsoft committed to a multi-year partnership aimed at further integrating Azure services with HashiCorp products. As a result of this collaboration, organizations can rely on tools like Terraform to create and manage Azure infrastructure. The tight integration and support for Azure allows operators to easily deploy resources on Azure using Terraform and secure them via Vault. Additionally, Microsoft utilizes HashiCorp tools for internal use. Packer is trusted for the creation of new Linux images for Azure services. This collaboration enables Microsoft and HashiCorp to create new and innovative ways for their products to integrate further, easing the cloud adoption journey for enterprise organizations.



Authored by HashiCorp