Microsoft

a

Best Practices for Running Applications on Azure Stack

By Mahesh Kshirsagar
Azure Global Customer Engineering (AGCE)

December 2019

Contents

Authored by Mahesh Kshirsagar. Edited by Nanette Ray. Reviewed by AGCE.

Introduction

Azure Stack is an extension of Microsoft Azure. It offers a consistent cloud management experience, deployment model, and Azure compute, storage, and networking services. However, unlike the public cloud Azure platform, Azure Stack offers all these services from your existing on-premises datacenters. Azure Stack provides a great choice when you must comply with data residency requirements and run your applications in an on-premises environment, or if you just want to make the most of your existing datacenter investment. Azure Stack gives you the advantage of the familiar Azure operating model.

Running any solution in a hybrid environment can pose certain challenges for application designers. Azure Global Customer Engineering (AGCE) is an advisory team that works with customers who are starting to use Azure Stack, and we have noticed common concerns about availability, resiliency, scalability, security, and management. With some upfront planning, you can design, develop, and deploy Azure Stack solutions successfully and make the most of a hybrid cloud platform.

This article is a collection of best practices to help you run applications successfully on Azure Stack and avoid the common pitfalls that challenge application design. These best practices are organized according to the two key phases of a typical Azure Stack deployment:

- Before the original equipment manufacturer (OEM) delivers the Azure Stack integrated system to your datacenter.
- After delivery, when a solution provider typically works with you to deploy the integrated system.

Best practices for each phase are discussed in this article. These practices apply specifically to Azure Stack in a connected scenario—that is, connected to the internet. However, most of the practices also apply when using Azure Stack in a disconnected scenario.

NOTE This article links to many related resources. These links are up-to-date as of the publishing date, but links can change. Some links go to products and services from other companies. These are not endorsements, just intended to be helpful on your cloud journey.

Best Practices Before Deployment

Azure Stack integrated systems are designed to help you get up and running quickly, but a little preparation will ensure a smooth integration with your environment. Consider the following practices before starting the Azure Stack deployment process.

Gather Benchmarks and Metrics

- Benchmark key metrics for existing applications in the current environment, whether that's a public cloud or an on-premises datacenter. Develop a set of metrics to provide a relative comparison between Azure Stack and the current hosting environment. For example, these key metrics can include:
  - Service-level agreements (SLAs).
  - Average, peak, and maximum scalability measured in requests per second (RPS) and transactions per second (TPS).
  - Request times.
  - Query-execution times.
  - Disk input/output operations per second (IOPS) and throughput.
  - Network throughput and latencies.
  - Recovery point objective (RPO) and recovery time objective (RTO).
- Profile each application's performance when under performance and during stress testing. Plan to use these test scripts for the application after deploying it to Azure Stack, and then compare the performance metrics.
- Consider your Azure Stack expansion options up front, choosing between a gradual capacity increase and a high-fixed capacity.
- Use guidance on building apps for Azure Stack to understand the high-level differences between Azure Stack and Azure.

Assess Compute, Storage, and Networking Needs

- Familiarize yourself with the service-level virtual machine (VM) differences, such as the various series, disk performance, and other considerations.
- Familiarize yourself with the service-level storage differences and the available features, such as file storage, the replications options, and storage access.
- Familiarize yourself with the service-level networking differences, such as IPV6 support, service endpoints, network watcher, and other options.
- Understand datacenter integration considerations to drive deployment activity with your OEM partner.
- Plan to avoid IP-address conflicts—a potential issue if you are deploying multiple Azure Stack instances or integrating with an existing on-premises site or another public cloud site (such as Azure or AWS).
- Understand the identity integration model to ensure that you get a one-time decision right the first time.

- Use Azure Stack Capacity Planner to determine the appropriate capacity and configuration of your Azure Stack hardware solutions. Ask for an OEM-specific capacity planner worksheet and use it to identify any deviation later.
- Learn about the Azure Stack resiliency resources, which are excluded from available capacity for running applications.

Consider Deployment and Support

- Understand the Azure Stack hardware drive types and deployment options. Choose between the all-flash deployment possibilities and hybrid deployment possibilities (the default) to match your disk performance requirements.
- Understand how support responsibilities are coordinated between your OEM, Microsoft, and you. Make sure you know how to plan for and respond to the two types of update packages that occur on a regular basis.
- Connect with Microsoft to get support on planning and buying Azure Stack.
- Explore the option to use the lab and test environments provided by Microsoft or your OEM and validate that your workload can run on Azure Stack as expected.
- Keep track of the Azure Stack roadmap to understand product innovations and how they can drive business innovation.

Best Practices After Deployment

After you deploy Azure Stack using the best practices cited earlier, consider your application environment. Start planning for availability, resiliency, scalability, and other key considerations that affect the design and operation of your applications.

Prepare for the Hand-Over from the OEM to the Azure Stack Operator

Use the following best practices to ensure a smooth transition from the OEM to the Azure Stack operator.

Prepare Environment for Users

- Use the Azure Stack validation tool to verify the integrity of your Azure Stack deployment.
- Configure your Azure Stack diagnostics and frequently offload the logs to an external storage space, such as a storage account in Azure or an additional on-premises storage device. Keep them there for one to three months, depending on your requirements.
- Register Azure Stack with Azure by using an Azure subscription that is identified for raising support incidents.
- Validate Azure Stack connectivity either to Azure or your existing hosting environment, if applicable. Set up ExpressRoute if Azure Stack needs to be connected to Azure.
- Ensure that the Azure Stack CA root certificate and the VM aliases endpoint are available so that developers can use Azure Stack CLI for configuration and use the command line and Azure Resource Manager templates for deployment.

- Install PowerShell for Azure Stack.
- Validate the opening and closing of the privileged endpoint (PEP) session with an Azure Stack support engineer.
- Verify and deploy any Azure Stack updates, if applicable. Make sure you follow the Azure Stack update activity checklist before, during, and after the updates.
- Make sure you know how to handle hardware component and physical disk replacement.

Integrate Systems

- Set up integration with existing datacenter monitoring, reporting, and ticketing solutions with Azure Stack.
- Set up role-based access control (RBAC) to control access to different operating and developer groups.
- Understand Azure Stack health and alert monitoring.
- Enable multi-tenancy in Azure Stack, if applicable.
- Populate the Azure Stack Marketplace.
- Add custom VM images in Marketplace that comply with your organization's policy.
- Deploy SQL Server, MySQL, and App Service resource providers, if applicable.
- Create an Azure Stack plan, offer, quota, and subscription, as required.

Manage and Monitor

- Understand the Azure Stack infrastructure security posture, and use the Infrastructure Backup Service reference.
- Validate your Azure Stack usage reporting.
- Understand the Azure Stack servicing policy. Plan an upgrade schedule in advance, considering any Azure Stack and OEM upgrades that are required to keep Azure Stack on the N-2 servicing policy. Azure Stack typically releases a new version every month. Check with your OEM on the release schedule.
- Configure and validate Azure Stack backup to ensure the backup process works to recover Azure Stack from a catastrophic failure.
- Use the template validation tool in Azure Stack if you are planning to move VMs from Azure to Azure Stack.
- Use QuickStart Azure Resource Manager templates to validate your deployment on Microsoft Azure Stack.

Maximize Availability

Use the following practices for high availability (HA). An application's availability is often expressed in your SLAs.

Use HA Strategies

- Use virtual machine scale sets or an availability set for high availability and scalability.
- Consider using Azure Site Recovery to make Azure Stack VMs readily available in the event of a disaster.
- Consider running a disaster recovery drill using Azure Site Recovery or an existing preferred tool so you can test backup and restoration.
- Use Azure Backup Server on Azure Stack to back up files and applications, such as SharePoint 2010, SharePoint 2013, and SharePoint 2016 and SQL Server on Azure Stack, to ensure high availability.
- Consider configuring hybrid cloud connectivity between Azure Stack and a secondary site to support highly available applications. For the secondary site, use Azure, another Azure Stack instance, your existing datacenter, or any other public cloud with the appropriate connectivity.
- Consider creating a geo-distributed app solution that spans Azure Stack and a secondary site. This allows you to deal with an outage in any of the sites and to maintain high application availability.
- Use Azure Traffic Manager to profile and configure cross-cloud routing, which ensures the traffic is routed between multiple application instances.
- Consider automating your DevOps processes and creating a hybrid continuous integration (CI) and continuous deployment (CD) pipeline. Manual deployment practices can lead to application unavailability. Use a CI tool such as Jenkins or TeamCity, or consider Azure DevOps, a PaaS service that provides a 99.9-percent SLA and multiple compliance certifications.
- Use guidance on deployment strategies for rolling upgrades, blue/green deployment, canary deployment, and versioned deployment for highly available applications.
- Consider using native database high-availability options, such as highly available MySQL databases.

Use HA Application Patterns

- Use Kubernetes for Azure Stack for running containerized workloads with high availability.
- Use Bitnami Kafka Cluster on Azure Stack for brokered communication between the frontend and backend, or use publish-subscribe (pub-sub) messaging.
- Use additional brokered communication solutions available on Azure Stack Marketplace.
- Consider using messaging patterns to address idempotency.
- Consider using a Throttling pattern to control a load-hitting application. When there is no more capacity for an application to scale, this pattern keeps it available for existing users.
- Consider using the Retry pattern to handle a transient network or other errors that can reduce availability.
- Consider using the Circuit Breaker pattern to avoid accessing services that are unavailable or recovering.

- Consider using the Command and Query Responsibility Segregation (CQRS) pattern to decouple the read and write operations so they don't access the same resource (such as a database) at the same time.

Monitor Availability

- Consider using guidance on integrating an external monitoring solution with Azure Stack to detect failed components that are affecting the availability.
- Consider using Azure Monitor on Azure Stack, installing Azure Log Analytics agents on Azure Stack VMs, and performing log analytics. These tools report in detail on errors and failures, enabling your operations staff to determine the cause, mitigate the situation, and ensure availability.
- Build a customized Azure Stack dashboard to give your operations staff quick insights into cloud resources, daily tasks, and other key details.

Increase Resiliency

Failures are inevitable, but you can be prepared. To avoid surprises, make sure you know your system's limits. Review your Azure Stack plan, offer, quota, and subscription limits. Make sure you understand the Azure Stack servicing policy so there's no delay when you need to engage the support process.

Use the following practices to improve the resilience of your applications in the event of a failure.

Design Resilient Architectures

- Consider using guidance on defining your resiliency requirements. Document your resiliency needs.
- Consider using the Azure Stack guidance on business continuity and disaster recovery.
- Consider configuring hybrid cloud connectivity between Azure Stack and a secondary site to serve user requests when the primary site is affected.
- Consider creating a geo-distributed app solution that spans Azure Stack and a secondary site. Distribute the application load across multiple sites to make it resilient to failures.
- Consider using business continuity and disaster recovery solutions from certified partners.
- Consider using guidance on Azure Stack firewall integration, or use your preferred solution to prevent distributed denial of service (DDoS) attacks and to provide a web app firewall (WAF). Options in Azure Stack Marketplace can help you isolate malicious requests from legitimate user requests.

Design Resilient Applications

- Test applications using test-for-resiliency strategies.
- Use multiple storage accounts for applications that exceed the scalability targets of a single storage account.
- Consider using data transfer tools for Azure Stack storage for storage replication.

- Use asynchronous programming in your preferred programming language with the async and await pattern.
- Consider using RBAC for Azure Stack and the Azure Stack infrastructure security posture to prevent someone from purposely or accidentally deleting resources and making an application unavailable.
- Use Apache JMeter load tests with Azure DevOps or perform load testing with Azure DevOps or with your preferred solution. Identify application behavior under load to remove any uncertainty about its behavior under stress.
- Use Azure DevOps Release Management and audit releases using the Azure DevOps History field, or use your preferred release auditing tool to prevent operators from deploying a bad update or improperly configured settings for an application and associated rollback.
- Use the blue/green or canary release deployment techniques to allow users to be redirected to production code in the event of a failure.
- Use guidance on deployment strategies for rolling upgrades, blue/green deployment, and Azure DevOps conditional rollback, or use your preferred CI/CD tool for rollback and to minimize downtime in case of a wrong deployment.
- Consider performing usage analysis with Application Insights for an instantaneous view into the health of your application and to reveal any issues in the services.

Design Resilient Compute

- Consider using guidance on protecting VMs deployed on Azure Stack for enhanced resiliency.
- Use VM scale sets for horizontal scaling to meet resiliency requirements appropriate for the incoming load.
- Consider using Hybrid Runbook Worker to vertically autoscale VM scale sets to maintain resiliency during a varying traffic profile.
- Consider using the Azure Stack load balancer and health probe, or use your preferred load balancing health probe solution, to prevent requests going to a faulty VM.
- Consider integrating an external monitoring solution with Azure Stack to detect and isolate faulty VMs.
- Consider using Application Insights and analyze log data to predict reliability issues and to monitor third-party SLAs.
- Use certified Azure Stack Marketplace images to create new VMs. Ensure that Linux images conform to the recommended guidance.
- Understand VM sizes supported in Azure Stack. Use Azure DevOps load testing, or your preferred tool, to perform regular load testing and to identify and fix capacity hotspots.

Design Resilient Recovery

- Consider running a disaster recovery drill using Azure Site Recovery or your preferred disaster recovery solution. Verify successful failover and failback for improved resiliency.
- Orchestrate database restore operations and test the validity regularly.

- Use guidance on integrating an external monitoring solution with Azure Stack, Azure Monitor on Azure Stack, or your preferred alerting and monitoring tool, and send alerts in case of failures.

Improve Scalability

Use the following best practices to plan your scalability targets and make your applications scalable. Some of these practices differ from those you may have used on Azure, because the capacity of Azure Stack systems can vary.

Scale Compute

- Use VM scale sets for horizontal scaling.
- Consider using Hybrid Runbook Worker to vertically autoscale with VM scale sets.
- Use the database optimization guidance for Azure Stack to help enhance performance of database VMs.
- Use the disk optimization guidance for Azure Stack to improve disk input/output operations per second (IOPS) and throughput.

Scale Applications

- Use multiple storage accounts for applications that exceed the scalability targets of a single storage account.
- Use topic partitioning for Kafka, or your preferred queuing solution, for improved scalability.
- Consider adopting a microservices architecture to distribute application components and maximize the use of each compute unit.
- Use Bitnami Kafka Cluster on Azure Stack, or your preferred brokered messaging solution from Azure Stack Marketplace, to queue requests and to balance your application load.
- Use the background jobs guidance to prevent an unresponsive application if it is initiating synchronous jobs.
- Use appropriate cache control headers, when applicable, for web apps and APIs, to reduce server load for static content for each request.
- Use an asynchronous programming pattern available in your programming language to prevent locking the thread while accessing resources with higher latency, limited I/O, or limited network bandwidth.
- Use gzip compression and bundling and minification for web apps and APIs to reduce load on the network and to help them scale.
- Use Azure DevOps load testing, or your preferred load-testing tool, for regular stress testing. Make sure the application scales as expected.

Scale Databases

- Cache frequently used data to prevent direct database access. Use one of the many cache solutions from Azure Stack Marketplace.
- Use the partitioning guidance to meet scalability requirements for databases.

- Use batching over multiple and frequent database queries to reduce chatty operations and improve scalability.
- Use an appropriate query tuning configuration for your databases for improved query performance and scalability.
- Use your database provider's options for managing future growth and preventing poor database throughput—two issues that affect scalability.
- Use an appropriate level of consistency and isolation while making a database connection to prevent poor performance from services with high latency.
- Use database connection pooling appropriately to limit the connection resources.

Boost Security

Security is a critical aspect of running any application. Use the following practices to address various security concerns.

Secure Identities and Access

- Use Azure Active Directory (Azure AD) multi-factor authentication, or your preferred multi-factor solution, to secure access to data and applications and to prevent risk of credential theft attack, which may lead to data compromise.
- Consider using RBAC for Azure Stack and grant users the least privilege to get their jobs done. Allow only certain actions at a particular scope to reduce the risk of data compromise associated with unrestricted access.
- Use Azure AD, or your preferred identity management solution, to centralize identity management, a tactic that reduces administrative overhead and decreases the likelihood of mistakes and security breaches.
- Use Azure AD, or your preferred identity management solution, to enable single sign-on (SSO), a feature that helps prevent the likelihood of users reusing passwords or using weak passwords.
- Use Azure AD with Azure Resource Manager to control resource creation using security policies that prevent users from abusing the service and creating more resources than they need.

Secure Against Threats

- Consider using solutions from Azure Stack Marketplace for endpoint protection to prevent attackers from compromising workstations and leveraging user credentials to gain access to your organization's data.
- Use any of the solutions from Check Point CloudGuard, F5 BIG-IP VE, Azure Stack firewall integration, or other services from Azure Stack Marketplace for DDoS and WAF protection.
- Use virtual private network (VPN) gateway solutions from Arista, F5, Palo Alto Networks, or your preferred network traffic control solution to control traffic and IP address translation.
- Use solutions from Check Point CloudGuard, F5 BIG-IP VE, or other vendors in the Azure Stack Marketplace that offer an intrusion detection system (IDS) and intrusion prevention system

(IPS), provide antivirus protection, deliver vulnerability management, protect from botnets, and address other security concerns.

- Consider using Azure Stack infrastructure security practices and integrating Azure Stack with Qualys or your preferred threat detection solution. Scan for vulnerabilities and policy compliance so you can prevent, detect, and respond to threats.
- Use Azure site-to-site VPN and application-level SSL/TLS for protection against man-in-the-middle and other attacks, eavesdropping, and session hijacking.

Protect Workloads and Data

- Consider using Azure Stack infrastructure security practices to reduce the threat of malicious activity, avoid data integrity issues, prevent malicious or rogue users from stealing data and compromising accounts, and gaining unauthorized access to data.
- Use F5 BIG-IP Virtual Edition, Azure Traffic Manager, or your preferred global load-balancing solution that offers policy-based, centralized management. Make sure applications and services remain available even if datacenters become unavailable.
- Learn how Azure Security Center helps protect hybrid cloud workloads by providing Microsoft Monitoring Agent, syslog forwarding, and physical device auditing to build a deeper understanding of network traffic patterns.
- Store secrets on a hardware secure module (HSM) to prevent attackers from gaining access to your secret keys and decrypting potentially confidential data. For example, use Thales CipherTrust Cloud Key Manager. Verify if your existing security solution is providing a similar HSM solution on Azure Stack Marketplace.
- Use managed disks in Azure Stack with BitLocker 128-bit AES encryption to enforce file-level data encryption.
- Use Azure Stack Key Vault to store secrets, credentials, certificates, and keys.
- Enable native auditing for databases, native database services for built-in security, automatic monitoring, and threat detection to identify suspicious database activities, potential vulnerabilities, SQL injection attacks, and anomalous database access patterns.
- Use secure database authentication schemes.
- Consider using Azure Stack diagnostics tools to trace requests, analyze usage trends, and diagnose issues.
- Use Secure DevOps Kit for Azure to build and deploy applications on Azure Stack with security integrated at every step.

Strengthen Security Management

- Consider using devices from Azure Stack Marketplace that help you understand, diagnose, and gain insights to your network in Azure Stack.
- Use a virtual network in Azure Stack to establish a network boundary that isolates VMs and keeps the communication between them private.
- Use a virtual network subnet in Azure Stack to define access policies based on the workload types and to control traffic flows between them.

- Use the WAF and load balancing solutions from Azure Stack Marketplace, Check Point, F5 Networks, Palo Alto Networks, or Positive Technologies to define communication paths within network boundaries.
- Specify user-defined routes (UDRs) within Azure Stack virtual networks to control the next traffic hop.
- Define network security groups (NSGs) to allow or deny traffic to and from a single IP address, to and from multiple IP addresses, or even to and from entire subnets.
- Use NSGs or add firewall devices from Azure Stack Marketplace to control traffic to database.
- Use forced tunneling to route internet connections through your datacenter's network security devices, thus preventing direct outbound connections from any VM and reducing the attack surface area leveraged by attackers.
- Disable RDP/SSH access to Azure Stack VMs. Use a VPN to access these VMs for remote management and to prevent attackers from using brute force techniques to gain access and launch other attacks.
- Consider using a malware protection solution from Azure Stack Marketplace.

Practice Good Management

Availability, resiliency, and scalability all depend on how well you maintain and operate the Azure Stack environment. Use the following best practices to keep Azure Stack in top condition.

Monitor the Environment

- Consider using Azure Stack diagnostics tools, forwarding syslogs, and integrating external monitoring solutions to keep critical conditions from going unnoticed.
- Consider importing Azure Log Analytics data into Power BI to look at monitoring data from multiple perspectives.
- Consider following the guidance on reporting Azure Stack usage data to Azure for usage telemetry and optimization.
- Consider using Hybrid Runbook Worker to unify management. Automate ITSM, DevOps, and monitoring using webhooks to provide a way to fulfill requests and ensure continuous delivery and operations.
- Consider using an Azure Log Analytics agent to collect log data, then consolidate views on an Azure Stack dashboard that can be shared with multiple stakeholders.
- Consider using Application Insights and analyze log data to form a complete picture of the operating environment.
- Consider using services from Azure Stack Marketplace or your preferred solution to detect clients that try to resolve malicious domain names, request loads on DNS servers, or cause dynamic DNS registration failures.

Manage Costs

- Use a billing partner, such as CloudAssert or Exivity, to help you extract Azure Stack usage data and provide consumption-based billing and cost management.

- Understand guidance on usage and billing in Azure Stack to organize resources for billing or management operations.
- Consider using Azure Hybrid Benefit for Windows Server for cost optimization.

Manage Access and Compliance

- Use Azure Resource Manager RBAC to manage access to Azure resources and control what users can do with those resources.
- Use Azure Cloud Shell integration with Azure Stack to manage Azure Stack resources without the overhead of installing, versioning, and maintaining a jumpbox.
- Consider using Hybrid Runbook Worker for automation to create, maintain, and invoke scheduled work for apps and to set multiple one-time and recurring schedules.
- Consider using guidance on Azure CLI in Azure Stack to provide secure and automatic authentication account access for management operations.
- Use Azure management and monitoring for process automation to support operations at scale.
- Consider using Hybrid Runbook Worker with update and change tracking solutions enabled via automation runbook to prevent configuration changes, detect health incidents, automatically receive configurations, conform to a desired state, and report on compliance.
- Consider using Azure Stack Policy Modules to keep resources compliant with corporate standards and SLAs.

Manage Resources

- Use Azure Resource Manager or HashiCorp Terraform Provider for Azure Stack to define the dependencies between resources so they're deployed in the correct order.
- Use resource tags to associate resources with the appropriate department, customer, and environment for easier management.
- Consider using compliance assessment reports for Azure Stack to enforce common and consistent tag usage, and to audit existing resources for appropriate configurations and settings.

Manage Deployments

- Use Azure Resource Manager to implement an infrastructure-as-code model for easier management and deployment.
- Use Azure Resource Manager templates and scripts to automate resource provisioning.
- Use guidance on deploying apps to Azure Stack using Azure DevOps.
- Use guidance on setting up hybrid CI/CD processes for Azure Stack.

Manage Backups and Recovery

- Consider using a solution for business continuity and disaster recovery or backup and data recovery and choose the data to protect.

- Use data protection and disaster recovery services that replace cumbersome, error-prone manual backup and file recovery tasks. Consider offerings from Microsoft partners such as Acronis, Actifio, Carbonite, Commvault, Dell EMC, Micro Focus, Quest, Rubrik, Veritas, and ZeroDown.

Learn More

Many Azure readiness practices also apply to Azure Stack, such as using good network security and naming and tagging conventions. For more information about these Azure practices, see the Best practices for Azure readiness.

Azure Architecture Center

Design considerations for hybrid applications

Designing reliable Azure applications

Scalability checklist

Azure Stack ISV solutions

DevOps checklist