

WIZ - EXHIBIT 1016 (Part 2)
Wiz, Inc. v. Orca Security Ltd.

- **Relays** Provide a bidirectional (two-way) communication pattern. Instead of storing messages like queues and topics, a relay simply proxies calls from the client/producer to the server/receiver. Service Bus Relays is also one of the older services in Azure. It was publicly announced in May 2006 as Live Labs Relay (since incorporated into the Service Bus family).
- **Event Hubs** Provide a highly scalable event and telemetry ingestion service for supporting scenarios requiring low latency and high reliability. The section below discusses Event Hubs in more detail.

Azure Event Hubs

Azure Event Hubs is a highly scalable managed service capable of ingesting millions of events per second, enabling you to capture, process, and analyze massive amounts of data originating from connected devices (often IoT scenarios) and applications. You can think of Event Hubs as a gateway, or entry point, for an event processing pipeline. Data is collected into an Event Hub, then transformed and stored. You have control over what data transformations and storage are needed.

The programmatic interface for Event Hubs is AMQP (Advanced Message Queuing Protocol) or HTTP(S), making it very easy for a wide range of clients to publish event data to Event Hubs. To support the need for massive scale, Event Hubs uses a partitioning pattern to scale the load internally. Receiving messages from an Event Hub is handled via consumer groups. Consumer groups are responsible for knowing from which partition to read and maintaining a view (state, position in the stream, etc.) of the Event Hub.

You will often see Azure Event Hubs used to ingest data in a big data or IoT scenario. A characteristic of both scenarios is the generation and processing of large volumes of (often relatively small in size) data. To process and analyze the data, another Azure service, Azure Stream Analytics, is often paired with Event Hubs.

It is important not to confuse Event Hubs with Azure Service Bus queues or topics. While the two are similar in that they are both messaging systems, Event Hubs is designed specifically for handling message events at high scale. It does not implement some of the messaging capabilities of Service Bus queues and topics, such as dead lettering, filters (property based routing), and various message retrieval, delivery, and scale semantics. Service Bus is better suited for per-message needs, while Event Hubs is better suited for event streaming needs.

Azure Notification Hubs

While Event Hubs allow you to take in millions of events per second, Azure Notification Hubs send data in the other direction—they enable you to send push notifications to mobile devices from any backend, whether in the cloud or on-premises. With a single API call, you can target individual users or entire audience segments of millions of users across all of their devices.

Push notifications are challenging. In general, the app developer still has to do much of the work to implement even common push notification scenarios, like sending notifications to a specific group of customers. To make them work, you have to build infrastructure that is complicated and, in most cases, unrelated to the business logic for the app.

Notification Hubs remove that complexity, eliminating the need for you to manage the challenges of push notifications. Notification Hubs are cross-platform—they can be used to support Windows, iOS, Android, and Windows Phone apps; they reduce the amount of push-specific code you have to put in your backend. They are fully scalable, allowing you to send notifications to millions of devices with a single API call.

All of the functionality of a push infrastructure is implemented in Notification Hubs for you. The devices only have to register their PNS handles, and the backend can send messages to customers without worrying about the platform the customers are using.

Azure Media Services

Azure Media Services enables you to provide audio or video content that can be consumed on-demand or via live streaming. For example, NBC used Azure Media Services to stream the 2014 Olympics (<http://blogs.microsoft.com/blog/2014/02/06/going-for-gold-windows-azure-media-services-provide-live-and-on-demand-streaming-of-2014-olympic-winter-games-on-nbc/#sm.00001fhr9yr2zfcwlu2fkqhgu8kp>).

To use Media Services, you can call the .NET or REST APIs, which allow you to securely upload, store, encode, and package your content. You can build workflows that handle the process from start to finish and even include third-party components as needed. For example, you may use a third-party encoder and do the rest (upload, package, deliver) using Media Services.

Media Services is easy to scale. You can set the number of Streaming Reserved Units and Encoding Reserved Units for your account. Also, although the storage account data limit is 500 TB, if you need more storage, you can add more storage accounts to your Media Services account to increase the amount of available storage to the total of the combined storage accounts. And last but not least, you can use the Azure CDN with Media Services for the fastest content delivery possible.

Azure Backup

Azure Backup is a backup as a service offering that provides protection for physical or virtual machines no matter where they reside—on-premises or in the cloud. Azure Backup encompasses several components (Azure Backup agent, System Center Data Protection Manager [DPM], Azure Backup Server, and Azure Backup [VM extension]) that work together to protect a wide range of servers and workloads.

Azure Backup uses a Recovery Services vault for storing the backup data. A vault is backed by Azure Storage (block) blobs, making it a very efficient and economical long-term storage medium. With the vault in place, you can select the machines to back up and define a backup policy (when snapshots are taken and for how long they're stored).

Azure Backup can be used for a wide range of data backup scenarios, such as the following:

- Files and folders on Windows OS machines (physical or virtual)
- Application-aware snapshots (VSS—Volume Shadow Copy Service)
- Popular Microsoft server workloads such as Microsoft SQL Server, Microsoft SharePoint, and Microsoft Exchange (via System Center DPM or Azure Backup Server)
- Linux support (if hosted on Hyper-V)
- Native support for Azure Virtual Machines, both Windows and Linux
- Windows 10 client machines

Even though Azure Backup and Azure Site Recovery share the same Azure portal experience, they are different services and have different value propositions. Azure Backup is for the backup and restore of data on-premises and in the cloud—it keeps your data safe and recoverable. Azure Site Recovery is about replication of virtual or physical machines—it keeps your workloads available in an outage.

Azure Site Recovery

Azure Site Recovery (ASR) provides a disaster recovery as a service solution for Hyper-V, VMware, and physical servers, using either Azure or your secondary datacenter as the recovery site. ASR can be a key part of your organization's business continuity and disaster recovery (BCDR) strategy by

orchestrating the replication, failover, and recovery of workloads and applications if the primary location fails.

While there are many attractive technical features to ASR, there are at least two significant business advantages:

- ASR enables the use of Azure as a destination for recovery, thus eliminating the cost and complexity of maintaining a secondary physical datacenter.
- ASR makes it incredibly simple to test failovers for recovery drills without impacting production environments. This makes it easy to test your planned or unplanned failovers. After all, you don't really have a good disaster recovery plan if you've never tried to fail over.

The recovery plans you create with ASR can be as simple or as complex as your scenario requires. They can include custom PowerShell scripts, Azure Automation runbooks, or manual intervention steps. You can leverage the recovery plans to replicate workloads to Azure, easily enabling new opportunities for migration, temporary bursts during surge periods, or development and testing of new applications.

Azure Key Vault

Azure Key Vault is used to safeguard cryptographic keys and secrets in hardware security modules (HSMs) and allows Azure applications and services to use them. For example, you might use Key Vault to store storage account keys, data encryption keys, authentication keys, .PFX files, or passwords.

You can use Azure Active Directory (Azure AD) to control access to a Key Vault, which means you can control access to your keys and secrets using Azure AD. We talked about one example in Chapter 4, "Azure Storage," where you can store your storage account keys that are used by a service principal (an identity representing an application) into an Azure Key Vault and give access only to that service principal, thus protecting your storage account keys.

You can generate keys using Key Vault, but you can also store keys you have generated outside Azure. For security purposes, Microsoft cannot see or extract your keys. There is also logging capability that allows you to monitor the use of your keys in Key Vault.

More Azure services

The list of Azure services in the preceding pages is a sampling of the many services available in the Azure platform. Azure moves at a rapid pace, and new services and features are offered frequently. The rapid pace of innovation is one of the many fun aspects of working with a dynamic platform like Azure.

You are encouraged to review the main Azure site at <http://azure.microsoft.com> to learn more about the many services available.

Also, there is a web application that shows the many services of Azure and allows you to drill down to learn more. See <http://aka.ms/azposterapp>.

Business cases

There are many business cases for using Microsoft Azure: from spinning up temporary development and test environments to extending your on-premises infrastructure into the cloud or developing new applications that take advantage of the features available in Azure. In this chapter, we discuss a few common scenarios to give you some ideas for how you can use Azure.

Development and test scenarios

One of the common workloads in Azure is development and test (dev/test). In most cases, you can replicate all or part of your production infrastructure in Azure, whether it be on-premises or already running in Azure, and use the replica for development, staging, or testing.

If you have an on-premises datacenter and you want to set up a dev/test environment, you have to procure hardware, install the operating system and the rest of the software, set up networking, configure the firewall, and so on. This can take a substantial amount of time. Once the testing is over, you have to either leave the hardware idle or repurpose it until you need it for other testing.

With Azure, you can provision what you need (virtual machines [VMs], web apps, databases, storage, and so on) and proceed with the testing within minutes. When you are finished testing, you can tear down all of the services and stop paying for them. In fact, using Azure you can automate the deployment and teardown of your dev/test environment by using PowerShell, the command-line interface (CLI), and/or Azure Resource Manager (ARM) templates.

Best of all, as your infrastructure grows, you can easily scale your dev/test environment to fit current needs. With an on-premises dev/test infrastructure, you have to go through the procurement and configuration process again.

If everything you have is on-premises, you can still use Azure for dev/test. You can set up a virtual network and extend your on-premises network into Azure. For example, you might want to test your application against a new version of SQL Server; you can have a web application running in your local datacenter that accesses SQL Server hosted in Azure.

If you have an MSDN subscription, you get a monthly credit to use for your dev/test infrastructure in Azure. In addition, several of the services are discounted. For example, Windows VMs are billed at the

equivalent Linux rate (effectively removing the Windows license cost). This can significantly lower the overall cost of setting up and using a dev/test infrastructure.

Here are some other business cases that you can cover by using Azure to quickly replicate the parts of your infrastructure.

- **The flexibility of trying something small really fast** Let's say you only want to test one thing. For example, you want to change the way something is displayed in your website, but you're not sure if it will work or how it will work. You can make the modifications to the website and then deploy it as a new website with the configuration pointing to the production backend. Then, you can check the workflow and visual layout and decide if it is worth setting up a complete dev/test environment to proceed.
- **Load testing** You can create an entire copy of your production environment and then do load testing on the copy. This can include different VMs, websites, storage, virtual networks, and so on. This gives you the isolation to do load testing without affecting any of the production services, and it can help you pinpoint potential bottlenecks in your workflow so you can handle them before they affect the customer.

You can use load testing to figure out the scope of the resources you need to handle different loads, such as the size or number of VMs or App Service instances. You can then change these services to autoscale when needed. For example, you might discover that as the percentage of CPU utilization exceeds 60 percent, your website becomes unacceptably slow, so you decide to implement autoscaling to increase the instance count when the CPU utilization hits the target value.

Load testing leads to a better overall experience for your customer.

- **Software upgrades** If you are using software from an external company, you will want to test your software with the company's software for compatibility before upgrading your production services. For example, if you use SQL Server 2012, and SQL Server 2014 is released, you will want to test your application against the new version before upgrading. You might need to modify your software to work with the new version of SQL Server and go through the whole cycle of testing, staging, and implementation.

In an on-premises environment, you probably can get a prerelease copy or a free short-term trial of the new version. However, you need to have infrastructure on which to run it, so as with the previous examples, you might need to procure hardware and so on. With Azure, the software might be available in a preconfigured VM, as is the case with Windows, SQL Server, Oracle, and Linux, among others. In that case, you can just provision a new VM with the new version in a dev/test scenario and run your software against it.

If there is no preconfigured VM in the Azure Marketplace, you can provision a Windows or Linux VM, install the new version of software on it, and use that for your dev/test scenario.

- **A/B testing** Let's say you want to perform A/B testing on your website without repeatedly redeploying the different versions of the website. Azure Web Apps allows multiple deployment slots. You can publish version A to one slot and version B to another and then swap them in and out of production as needed to perform the testing and metrics collection.

Another option is to use the DNS query and routing policies available with Azure Traffic Manager. Traffic Manager gives you the ability to balance incoming traffic among multiple services using the following methods: performance, weighted, and priority. With the weighted method, you can distribute load to various endpoints based upon an assigned weight for each endpoint. This means you can divert a small percentage of traffic to a separate deployment to perform A/B testing.

For more information about using Traffic Manager load balancing, check out <https://azure.microsoft.com/documentation/articles/traffic-manager-overview/>.

Hybrid scenarios

The number of companies running solutions in the cloud is increasing at an incredible rate. Their success encourages other organizations to take the same step. Some organizations will not be able to move all of their workloads into the cloud, either because of regulatory issues or because some workloads cannot run in a virtualized environment. In these cases, hybrid computing, in which a company runs part of its infrastructure in the cloud and part on-premises, will be an important strategy.

The Microsoft Azure platform provides a great hybrid computing story. There are multiple ways to connect an on-premises datacenter to one or more Azure regions. As discussed in Chapter 5, “Azure Virtual Networks,” Azure provides both site-to-site and point-to-site virtual network connectivity. Either option provides a secure VPN connection between on-premises assets and resources hosted in Azure. An additional hybrid connectivity option is Azure ExpressRoute, which enables a private connection between Azure and your on-premises infrastructure or colocation facility, all without going over the public Internet.

Network connectivity

Regardless of the chosen option—site-to-site, point-to-site, or ExpressRoute—hybrid connectivity is a key scenario for the Azure platform. Creating a hybrid connection opens a wide range of possibilities to extend an on-premises infrastructure to the cloud. Two common scenarios for network-enabled hybrid connectivity are the following:

- **Hosting a website in Azure but keeping the database on premises** In an organization’s journey to the cloud, migrating the on-premises data to Azure can be one of the more difficult tasks. The difficulty usually comes in one of two forms: a technical issue or a compliance requirement. On the technical front, as an example, the application in question is designed to use a database that is not supported in Azure. On the compliance front, perhaps there is a regulatory requirement that cannot be met with Azure SQL Database or by running a database (SQL Server, MongoDB, and so on) on Azure Virtual Machines. In these cases, an organization might choose to host the website in Azure using Azure Web Apps or Azure Virtual Machines, with the database remaining on premises. Connectivity between the website and the database could then be established using one of the aforementioned technologies: a site-to-site connection, a point-to-site connection, or ExpressRoute.
- **Accessing an on-premises service** Sometimes, a website has a dependency on a particular service that cannot be moved to the cloud. Perhaps the website depends on an API that performs a crucial business calculation, and that API cannot be moved due to security because other on-premises services also depend on the service or because it is legacy technology that is not supported in Azure. In such a scenario, a hybrid connection is established between Azure and the on-premises infrastructure to allow the Azure-hosted website to freely communicate with the necessary API that continues to reside on-premises.

Besides using a network connection in this scenario, an Azure Service Bus Relay could be used to access an on-premises service. For information on how to use the Azure Service Bus Relay service, please refer to <http://azure.microsoft.com/documentation/articles/service-bus-dotnet-how-to-use-relay/>.

Internet connectivity

There are many scenarios in which all that is needed is an Internet connection rather than a special hybrid connectivity solution. After all, the ability to connect to Internet-accessible services is one of the attractive features of cloud computing. A few common scenarios include these:

- **Storage of archival data** Large amounts of data, especially archival data that is rarely accessed, can be very expensive to store on-premises. The cost in terms of infrastructure, people, software licenses, and physical space can quickly put a tremendous financial burden on an organization. As discussed in Chapter 4, “Azure Storage,” Azure provides virtually limitless storage capacity at an incredibly low price. An organization might wish to use the scalable storage provided by Azure Blob storage as a data archival store. When the data is needed, the on-premises service(s) download the data from Azure Blob storage and perform the necessary processing. A basic Internet connection will often suffice, but an ExpressRoute connection could also be used for improved speed and security.

Another option for storage of archival data is Microsoft Azure StorSimple. StorSimple includes a hardware appliance that is installed on-premises. The appliance keeps frequently accessed data local (on the device). As data ages (is accessed less frequently), it is automatically moved to Azure Blob storage. For more information on StorSimple, please refer to <http://azure.microsoft.com/documentation/services/storsimple/>.

- **Azure Active Directory** As discussed in Chapter 7, “Azure Active Directory,” organizations can choose to synchronize their Azure AD users and groups with user and group information from their on-premises Active Directory. In doing so, they can use Azure Active Directory Connect to synchronize the user data and a password hash, making Azure AD the authority for user authentication. Alternatively, an organization might wish to synchronize the user data but require users to authenticate via an Active Directory Federation Services (AD FS) endpoint residing on-premises, effectively redirecting the user to an on-premises AD FS site for authentication before redirecting to the desired location.
- **Burst to the cloud** Sometimes, an organization’s on-premises infrastructure is not able to handle the required load. Maybe there is a holiday season rush or a government-mandated period to sign up for an important service. Instead of building the on-premises infrastructure to handle the temporary surge in demand, an organization might choose to leverage the elastic nature of the cloud to burst to the cloud when needed and scale back to only on-premises services when the load returns to normal. In this scenario, an organization could use Azure Web Apps or Azure Virtual Machines to host the service and could implement autoscale rules to ensure capacity keeps up with user demand.

Application and infrastructure modernization and migration

There comes a time in every application’s life when it is time to upgrade. It could be a user interface redesign or a hardware refresh. The Azure platform cannot help create an appealing, modern user interface, but it can modernize the supporting infrastructure.

Many organizations will go through a periodic hardware refresh cycle; typically, this happens about every three years. When it is time for a hardware refresh, organizations today have a new question to ask: Should we buy new on-premises hardware, or should we leverage our infrastructure and services to the cloud?

Besides a required hardware refresh, an organization might choose to migrate to the cloud because it has reached physical capacity limits in its existing on-premises datacenter or because it is going to in

the very near future. Perhaps the current datacenter does not have enough physical space for more servers or cannot supply the necessary power or cooling. Maybe there is a desire to eliminate or reduce the management of hardware infrastructure going forward. Moving to the cloud might enable the organization to get out of the datacenter business completely, or at least partially (see the section “Hybrid scenarios” earlier in this chapter). In this case, Microsoft is responsible for the hardware and related infrastructure components of the datacenter, and the organization can focus on providing great business solutions.

Some organizations will choose to migrate to the cloud to get capacity in new geographies they can’t currently support because they have no presence in that area or because it would be cost-prohibitive. There are Azure datacenters in over 22 regions around the world from Melbourne to Amsterdam and from Sao Paulo to Singapore. Additionally, Microsoft has an arrangement with 21Vianet, making Azure available in two regions in China. Microsoft has also announced the deployment of Azure to another eight regions. Instead of building and maintaining a global datacenter presence, an organization can elect to take advantage of Microsoft’s existing investments and deploy to multiple regions with ease.

See Also For the current list of Azure regions, please refer to <http://azure.microsoft.com/regions/>.

Should the choice be to modernize or migrate to the cloud, there is certainly a wealth of Azure resources available. In choosing to adopt these resources, an organization could have many questions to answer, including these:

- Do we leverage platform as a service (PaaS), infrastructure as a service (IaaS), or both?
- Instead of maintaining a custom solution, should we leverage platform-provided services such as Azure Search or Azure Media Services?
- Should we move everything, or just some components? What hybrid model works best for our requirements?
- Which Azure region(s) should we use?
- How does using Azure affect our business and operations model?
- What is our service level agreement (SLA)? What is our disaster recovery story?

Azure Mobile Apps

In today’s world, mobile devices—from tablets to phones to watches to fitness bands—are everywhere you look. Having a mobile application can be a big plus for a company, whether it’s used externally, internally, or both.

Azure Mobile Apps, included as part of Azure App Service, is a backend as a service that provides multiple features to make it easier and quicker to create a mobile application. Mobile Apps is both flexible and scalable, so when your application becomes widely used, you can scale appropriately to handle your customers’ needs.

Another advantage of Azure Mobile Apps is that you only have to write one version of your backend. The backend can be used by devices running iOS, Android, and Windows, allowing you to reach every user on every platform without extra work.

The following are some of the features provided by Azure Mobile Apps. You can certainly program a service to implement these features from the ground up, but using Azure Mobile Apps saves you the time and money it would take to do that.

- **Data storage** You can choose for your data storage to be powered by SQL Database, which has an interface simple enough to use without being a DBA. You can also integrate with SQL Server, Azure Table Storage, MongoDB, DocumentDB, or via an API to software as a service (SaaS) providers such as Salesforce.com and Office 365.

You can write your application to work offline and synchronize the data when the application can go online again. This is helpful when the customer loses Internet connectivity—the customer can continue to work, knowing the work will be stored on the backend when connectivity is regained.

- **User authentication and data authorization are greatly simplified** You can easily implement single sign-on (SSO) with Azure AD, a Microsoft account, Facebook, Twitter, and Google.
- **Push notifications** You can send information for customer and enterprise applications to any customer's mobile device by using Microsoft Azure Notification Hubs. This can come from any backend, whether it runs in Azure or is on-premises. Notification Hubs automatically handles the server-side code to push messages to the push notification services for iOS, Android, and Windows devices.

Notification Hubs has a tagging feature that can be used to target audiences based on activity, interest, location, or preference. In addition, the templates feature of Notification Hubs enables you to send localized push notifications in the customer's own language.

- **Because Mobile Apps runs in Azure, you can easily scale in and out to meet customer demand** You can even set up autoscaling that will automatically scale out as demand increases, handling millions of devices.
- **You can use Microsoft Azure WebJobs to perform backend processing on the server at a scheduled time** For example, you might want to create a scheduled job that requests an update from your on-premises database and stores the new information in a table, waiting to be retrieved by your mobile application.
- **You can create a hybrid connection** This connection can be used to connect the mobile application to on-premises systems, Office 365, and SharePoint.

Machine learning

One of the most talked about emerging technologies in the last few years is machine learning. Machine learning isn't new by any stretch; however, the ability for a wide range of people to easily access the technology is relatively new—largely enabled by the ubiquitous nature of public cloud computing.

By using machine learning, we are able to leverage computers to analyze existing data to predict future behavior or outcomes. Maybe you want to predict when machines will break down instead of sending technicians to check on machines that may be working fine. Perhaps you want to analyze historical shopping data to predict what customers are likely to purchase in the future. These are just two of the many potential scenarios that are possible with machine learning.

It's becoming increasingly common to see machine learning mentioned along with Internet of Things (IoT). IoT solutions typically generate a lot of data from various sensors, such as temperature, vibrations, speed, and so on. The data itself is often largely useless—the true value comes in being able to analyze the data and determine what to do with the data to improve the overall solution. This is where machine learning comes into the picture. Combining the data from IoT solutions with machine learning can lead to interesting and useful insights about the data.

At first this might seem daunting, but it isn't—especially when using a service such as Azure Machine Learning. With Azure Machine Learning, there is no hardware to purchase or virtual machines to

manage. In fact, you don't even need an Azure subscription to get started with the Free tier of Azure Machine Learning! You can learn more about the pricing options for Azure Machine Learning at <https://azure.microsoft.com/pricing/details/machine-learning/>.

See Also Azure Machine Learning is also a component of the Cortana Intelligence Suite. Learn more at <https://www.microsoft.com/server-cloud/cortana-intelligence-suite/>.

The basic workflow in Azure Machine Learning is relatively simple:

1. Build a model from existing data. The data can come from numerous data stores in Azure, such as Azure Storage tables or blobs, Azure HDInsight (Hadoop), Azure SQL Database, or Azure Data Lake.
2. Publish the model as a web service.
3. Optionally, consume that web service from any number of tools such as mobile applications, websites, or business intelligence tools.

With the data in place, you can create your predictive model in Azure Machine Learning Studio, a browser-based tool with drag-and-drop capabilities that make it easy to get started. If you're familiar with machine learning and understand how to use R (a programming language commonly used for data analysis) or Python, you can get started right away with Azure Machine Learning. If you're not familiar with machine learning, you can get started by using solution templates in the Cortana Intelligence Gallery or by leveraging existing solutions in the Azure Marketplace (Data Market).

Once the model is created and properly trained (a process for validating that the model works as expected), you can publish the model as a web service. This will allow you—or others, based on your usage needs—to send data to your service and receive the predictions!

Azure Machine Learning is the perfect complement to the voluminous amount of data generated by many of today's IoT solutions. It's never been easier to gather, store, analyze, and make decisions based on data.

About the authors



Robin E. Shahan has over 25 years of experience developing complex, business-critical applications for Fortune 100 companies. As President of Nightbird Consulting, she provided training and helped companies architect and develop scalable, efficient solutions on the Azure platform. She is a six-time Microsoft MVP, and now works for Microsoft as a Sr. Content Developer for Azure Storage.

Robin regularly speaks about Microsoft Azure at various .NET User Groups and Code Camps and runs the San Francisco Azure meetup. She can be found on Twitter as [@RobinDotNet](https://twitter.com/RobinDotNet), and you can read her articles about Microsoft Azure (and other subjects) at <http://robindotnet.wordpress.com>. You can reach her via e-mail at robin.shahan@microsoft.com.



Michael S. Collier is currently a Senior Software Development Engineer in the DX TED Commercial ISV team at Microsoft, and previously served as a Cloud Solution Architect. Prior to joining Microsoft in January 2015, Michael was a five-time Azure MVP and served as a Principal Cloud Architect with Aditi Technologies. He has over 15 years of experience with various consulting and technology firms where he was instrumental in leading and developing solutions for a wide range of clients. He has a vast amount of experience in helping companies determine the best strategy for adopting cloud computing, and providing the insight and hands-on experience to ensure they are successful. Michael is also a respected technology community leader, and is often found sharing his Microsoft Azure insights and experiences at regional and national conferences. Michael is also the co-founder of CloudDevelop Conference in Columbus, OH. You can follow Michael's experiences with Azure on his blog at <http://www.michaelscollier.com> and on Twitter at @MichaelCollier (<http://www.twitter.com/MichaelCollier>).

Michael lives in Marysville, Ohio with his wife and two sons. He is a 2003 graduate of The Ohio State University and is a passionate Buckeyes fan. Michael is also an avid golfer, although golf doesn't always like him.



From technical overviews to drilldowns on special topics, get *free* ebooks from Microsoft Press at:

www.microsoftvirtualacademy.com/ebooks

Download your free ebooks in PDF, EPUB, and/or Mobi for Kindle formats.

Look for other great resources at Microsoft Virtual Academy, where you can learn new skills and help advance your career with free Microsoft training delivered by experts.

Microsoft Press