

Azure para arquitectos

Implementación de diseño de cloud, DevOps, IoT y soluciones sin servidor en el cloud público



Packt

www.packt.com

Ritesh Modi

Azure para arquitectos

Implementación de diseño de cloud, DevOps,
IoT y soluciones sin servidor en el cloud público

Ritesh Modi



BIRMINGHAM - MUMBAI

Azure para arquitectos

Copyright © 2017 Packt Publishing

Todos los derechos reservados. Ninguna parte de este libro puede reproducirse, almacenarse en un sistema de recuperación o transmitirse en cualquier formato o por cualquier medio, sin el permiso previo por escrito del editor, excepto en el caso de citas breves incluidas en revisiones o artículos críticos.

En aras de asegurar exactitud de la información presentada, se han realizado todos los esfuerzos posibles en la preparación de este libro. No obstante, la información contenida en él se proporciona sin garantía, ya sea expresa o implícita. Ni el autor ni Packt Publishing, o sus filiales y distribuidores, serán responsables de cualquier daño causado o presuntamente causado por este libro ya sea de manera directa o indirecta.

Si bien Packt Publishing ha procurado suministrar información sobre las marcas comerciales de todas las empresas y productos mencionados en este libro mediante el uso correspondiente de mayúsculas, no puede garantizar la exactitud de esta información.

Primera publicación: octubre de 2017

Referencia de producción: 1181017

Publicado por Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, Reino Unido.

ISBN 978-1-78839-739-1

www.packtpub.com

Créditos

Autor

Ritesh Modi

Revisores

Paul Glavich

Vikram Pendse

Rubén Oliva Ramos

Responsable de publicación

Gebin George

Responsable de adquisición

Shrilekha Inani

**Responsable de desarrollo
de contenido**

Abhishek Jadhav

Responsable técnico

Aditya Khadye

Correctores

Safis Editing

Juliana Nair

Coordinador de proyecto

Judie Jose

Corrector de textos

Safis Editing

Indexador

Tejal Daruwale Soni

Gráficos

Kirk D'Penha

Coordinador de producción

Arvindkumar Gupta

Acerca del autor

Ritesh Modi fue uno de los principales defensores de la tecnología de Microsoft, aunque actualmente trabaja como asesor principal en Infront Consulting Group. Es arquitecto, promotor senior, arquitecto de clouds, autor publicado, portavoz y reconocido líder por sus contribuciones a soluciones tales como centros de datos, Azure, bots, cadenas de bloques, servicios cognitivos y DevOps, además de soluciones de inteligencia artificial y automatización. Ha publicado numerosos libros. *Developing Bots using Bot Framework* y *DevOps with Windows Server 2016* son algunas de sus publicaciones más recientes. Ha colaborado también con el equipo de Windows Server en otro libro, titulado *Introducing Windows Server 2016 Technical Preview*.

Ha participado en más de 15 conferencias, incluidas TechEd y la Conferencia de Asia de PowerShell, además de realizar algunas colaboraciones con la revista MSDN. Cuenta con más de una década de experiencia en el diseño y la implementación de soluciones empresariales para consumidores, así como con más de 25 certificaciones técnicas a sus espaldas. Entre sus intereses y aficiones se incluyen escribir, jugar con su hija, ver películas y continuar aprendiendo nuevas tecnologías. Su alias de Twitter es @automationnext.

Ritesh vive actualmente en Hyderabad (La India).

Agradecimientos

Escribir este libro ha sido una experiencia fantástica. Personalmente he crecido como persona y he ganado en paciencia, perseverancia y tenacidad. Le debo mucho a la gente que me animó con su apoyo y motivación. Me gustaría dar las gracias a muchas personas por hacer que este libro haya sido posible.

Debo empezar por las personas que significan todo para mí, que me inspiran para continuar y que, a fin de cuentas, hacen que todo merezca la pena. Me refiero a mi madre, Bimla Modi, mi mujer, Sangeeta Modi, y mi hija, Avni Modi: las tres mujeres más maravillosas de mi vida. También me gustaría dar las gracias a mi padre por su apoyo continuo para asegurarse de que no abandonaba este proyecto.

Y, por supuesto, debo dar las gracias al equipo de Packt. Me gustaría agradecer a Abhishek Jadhav, responsable de desarrollo de contenido, por embarcarme en este proyecto y ayudarme a través de él. Me gustaría agradecer a Shrilekha Inani, responsable de adquisición, por encontrarme para hacer este libro. Y también me gustaría dar las gracias al responsable técnico, Aditya Khadye, que leyó varias veces este libro y me proporcionó información increíblemente útil.

Por último, me gustaría pedir disculpas a mi familia, a mis amigos y a todas las personas que me aprecien por no poder pasar mucho tiempo con ellos durante el último año. Les voy a compensar.

Acerca de los revisores

Paul Glavich ha sido un MVP de ASP.NET durante 13 años y actualmente trabaja como asesor principal para Readify. Anteriormente, fue director de tecnología (CTO) en Saasu, arquitecto de soluciones en Datacom, asesor senior en Readify y, antes que todo eso, ejerció como arquitecto técnico para EDS Australia. Tiene una experiencia de más de 20 años en el sector que abarca desde PICK, C, C++, Delphi, and Visual Basic 3/4/5/6 hasta su especialidad actual en .NET con C#, ASP.NET, Azure, Cloud y DevOps.

Paul ha estado desarrollando tecnologías .NET desde que .NET se encontraba en cierres en su versión beta. También fue el arquitecto técnico de una de las primeras soluciones por Internet que utilizaban tecnología .NET del mundo.

Paul ha formado parte de diferentes grupos informativos relacionados con la tecnología .NET y del grupo de usuarios de .NET de Sidney y TechEd, además de ser miembro de ASPInsiders. Ha escrito, también, algunos artículos técnicos, que pueden verse en sitios de la comunidad tales como ASP Alliance. Paul ha publicado un total de tres libros, *Beginning AJAX in ASP.NET*, *Beginning Microsoft ASP.NET AJAX* y su última obra *.NET Performance Testing and Optimisation*. Actualmente, su trabajo se centra en la arquitectura general, el diseño de soluciones y las soluciones Cloud de Microsoft.

En un plano más personal, Paul está casado y tiene tres hijos y tres nietos. Es cinturón negro de 5º grado en Budo Jitsu y también practica Wing Chun Kung Fu.

Hay muchas personas que me han ayudado a llegar a donde estoy hoy, pero se necesitaría otro libro. Así que, para ser breves, me gustaría dar las gracias a mis tres hijos, Kristy, Marc y Elizabeth por ser increíbles, a mis padres por comprarme aquel primer ordenador, a mis amigos "friquis" por compartir mis aficiones, pero, sobre todo, me gustaría dar las gracias a mi mujer, Michele, por apoyarme en mi vida y carrera profesional y por aguantar mis interminables charlas tecnológicas y chistes malos.

Vikram Pendse es un MVP de Microsoft para Azure y ha sido conferencista distinguido en diversos eventos de Microsoft en los últimos 10 años. Es miembro muy activo de varias comunidades de Microsoft en La India. Trabaja como arquitecto de soluciones de cloud y actualmente colabora con uno de los principales partners de Microsoft en Pune, donde es responsable de diseñar la estrategia para traspasar las cargas de trabajo de Amazon AWS a Azure, proporcionar soluciones centradas en el cloud, diseñar arquitecturas, ofrecer asistencia para RFP y llevar a cabo entregas a nivel mundial. Lo puedes seguir en Twitter en @VikramPendse.

Ha sido revisor técnico en Packt para los siguientes libros:

- *Microsoft SharePoint 2010 Enterprise Applications on Windows Phone 7*
- *Microsoft Silverlight 4 Data and Services Cookbook*

Me gustaría dar las gracias a mi mujer, Aarti, y a mi hijo, Aditya, por su aliento y apoyo durante todo el proceso.

Rubén Oliva Ramos es ingeniero en Sistemas computacionales por el Instituto Tecnológico de León, además de contar con un máster en Ingeniería de sistemas electrónicos y computacionales y ser Especialista en teleinformática y redes por la Universidad de la Salle Bajío en León (Guanajuato, México). Cuenta con más de 5 años de experiencia en el desarrollo de aplicaciones web para dispositivos de supervisión y control conectados a Arduino y Raspberry Pi que utilizan marcos de trabajo web y servicios en el cloud para crear aplicaciones del Internet de las cosas.

Es profesor de Mecatrónica en la Universidad de la Salle Bajío y ejerce como docente en el Máster en Diseño e ingeniería de sistemas mecatrónicos. También trabaja en el Centro de Bachillerato Tecnológico Industrial 225 en León (Guanajuato, México), es profesor de materias como electrónica, robótica y control, así como automatización y microcontroladores en la especialidad de Mecatrónica. Ejerce como asesor y desarrollador de diversos proyectos, entre los que se incluyen la supervisión de sistemas y datos de registradores de datos que utilizan tecnologías como Android, iOS, Windows Phone, HTML5, PHP, CSS, Ajax, JavaScript, Angular y bases de datos de ASP .NET (SQLite, MongoDB y MySQL), servidores WEB (Node.js e IIS), programación de hardware (Arduino, Raspberry Pi, Ethernet Shield, GPS y GSM/GPRS y ESP8266), y sistemas de control y supervisión de programación y adquisición de datos.

Ha escrito un libro para Packt titulado *Internet of Things Programming with JavaScript*. También es autor de los libros *Monitoring, Controlling, and Acquisition of Data with Arduino* y *Visual Basic .NET for Alfaomega*.

Me gustaría dar las gracias a Jesucristo, mi Señor y Salvador, por darme la fuerza y el coraje para perseverar en este proyecto. A mi querida esposa, Mayte, a nuestros dos encantadores hijos, Rubén y Darío, a mi padre, Rubén, a mi querida mamá, Rosalía, a mi hermano, Juan Tomás, y a mi hermana, Rosalía, a quien adoro, por todo su apoyo mientras revisaba este libro, por permitirme alcanzar mi sueño y por tolerar mis ausencias después de mis largos días de trabajo.

Estoy muy agradecido a Packt por darme la oportunidad de colaborar como autor y revisor, así como por pertenecer a este honrado y profesional equipo.

www.PacktPub.com

Para obtener descargas y archivos complementarios relacionados con el libro, visita www.PacktPub.com.

¿Sabías que Packt ofrece versiones en e-book de cada libro publicado en formato PDF e ePub? Puedes actualizar a la versión e-book en www.PacktPub.com y, como cliente de libros impresos, tienes derecho a un descuento en la copia del e-book. Ponte en contacto con nosotros en la dirección de correo electrónico customercare@packtpub.com para obtener más detalles.

En www.PacktPub.com, también podrás leer una colección de artículos técnicos gratuitos, inscribirte en una gran variedad de boletines gratuitos y recibir descuentos y ofertas exclusivas en libros e e-books de Packt.



<https://www.packtpub.com/mapt>

Disfruta de las habilidades de software más demandadas con Mapt. Mapt te permite acceder a todos los cursos en vídeo y libros de Packt, así como a las herramientas líderes del sector, para ayudarte a planificar tu desarrollo personal y avanzar en tu carrera profesional.

¿Por qué suscribirte?

- Acceso total a cualquier libro publicado por Packt
- Capacidad para copiar, pegar, imprimir y marcar contenidos
- Búsquedas a demanda y accesibles mediante un explorador web

Comentarios de clientes

Gracias por adquirir este libro de Packt. En Packt, la calidad es la base de nuestro proceso editorial. Para ayudarnos a mejorar, puedes dejarnos tu sincera opinión en la página de este libro de Amazon en <https://www.amazon.com/dp/1788397398>.

Si te gustaría unirte a nuestro equipo de revisores habituales, puedes enviarnos un correo electrónico a customerreviews@packtpub.com. Premiamos a nuestros revisores habituales con vídeos e e-books gratuitos a cambio de sus valiosos comentarios. Ayúdanos a ser implacables en la mejora de nuestros productos.

Contenido

Prefacio	xi
Capítulo 1: Introducción	1
Infraestructura como servicio	3
Plataforma como servicio	4
Software como servicio	4
¿Qué es Azure?	5
Azure como un cloud inteligente	7
Azure Resource Manager	8
Arquitectura de Azure Resource Manager	8
ARM y ASM	9
Ventajas de ARM	10
Conceptos de ARM	10
Proveedores de recursos	11
Tipos de recursos	11
Grupos de recursos	11
Recursos e instancias de recursos	12
Características de Azure Resource Manager	12
Virtualización	14
Contenedores	15
Docker	17
Interacción con el cloud inteligente	17
Azure Portal	17
PowerShell	18
Interfaz de la línea de comandos (CLI) de Azure	18
API de REST de Azure	19
Plantillas de Azure Resource Manager	19
Implementaciones	20
Resumen	20

Capítulo 2: Patrones de diseño de Azure	21
Zonas y regiones de Azure	21
Disponibilidad de recursos	22
Cumplimiento normativo relativo a datos y privacidad	22
Rendimiento de las aplicaciones	22
Coste de ejecución de aplicaciones	23
Red virtual	23
Consideraciones de arquitectura para redes virtuales	24
Beneficios de las redes virtuales	27
Diseño de red virtual	27
Conexión con los recursos dentro de la misma región y suscripción	28
Conexión con los recursos dentro de la misma región y en otra suscripción	28
Conexión con los recursos en diferentes regiones y en otra suscripción	29
Conexión con centros de datos on-premises	29
Almacenamiento	32
Categorías de almacenamiento	33
Tipo de almacenamiento	33
Características de almacenamiento	34
Consideraciones de arquitectura para cuentas de almacenamiento	35
Patrones de diseño	36
Patrones de mensajería	37
Consumidores en competencia	38
Cola de prioridad	39
Patrón de nivelación de carga basada en cola	40
Patrones de rendimiento y escalabilidad	41
Patrón CQRS (Command and Query Responsibility Segregation)	41
Patrón de limitación	42
Otros patrones	43
Patrón de reintento	43
Patrón de interruptor de circuito	45
Resumen	47
Capítulo 3: Diseño de alta disponibilidad	49
Alta disponibilidad	50
SLA	50
Factores que afectan a la alta disponibilidad	51
Mantenimiento planificado	51
Mantenimiento no planificado	51
Arquitectura de implementación de aplicaciones	51
Alta disponibilidad y escalabilidad	52
Alta disponibilidad y recuperación ante desastres	52
Alta disponibilidad de Azure	52
Conceptos	53
Conjuntos de disponibilidad	53

Dominio de error	53
Dominio de actualización	54
Zonas de disponibilidad	54
Equilibrio de carga	54
Alta disponibilidad de las máquinas virtuales	55
Alta disponibilidad informática	55
Alta disponibilidad del almacenamiento	57
Alta disponibilidad de PaaS	58
Alta disponibilidad de aplicaciones	59
Equilibrio de carga	59
Equilibradores de carga de Azure	60
Equilibrio de carga público	61
Equilibrio de carga interno	62
Reenvío de puertos	64
Puertas de enlace de aplicaciones de Azure	64
Azure Traffic Manager	65
Consideraciones sobre la arquitectura de alta disponibilidad	66
Alta disponibilidad en las regiones de Azure	67
Alta disponibilidad en las regiones de Azure	68
Prácticas recomendadas	69
Alta disponibilidad de aplicaciones	69
Implementación	69
Administración de datos	70
Supervisión	70
Resumen	71
 Capítulo 4: Implementación de la escalabilidad	 73
Escalabilidad	74
Escalabilidad y rendimiento	75
Escalabilidad de Azure	75
Conceptos	75
Escalado	75
Escalado vertical	76
Reducción vertical	76
Escalado horizontal	76
Reducción horizontal	77
Escalado automático	77
Escalabilidad PaaS	77
Escalado o reducción vertical de PaaS	79
Escalado o reducción horizontal de PaaS	79
Escalabilidad IaaS	81
Conjuntos de escalado de máquinas virtuales	81
Arquitectura de VMSS	83

Contenido

Escalado de VMSS	83
Escalado horizontal frente a vertical	84
Capacidad	84
Escalado automático	84
Actualizaciones	87
Actualizaciones de la aplicación	89
Actualizaciones de máquinas invitadas	89
Actualizaciones de imagen	89
Prácticas de escalado recomendadas	90
Preferencia del escalado horizontal	90
Reconstrucción completa frente a instancias latentes	90
Configuración adecuada del número máximo y mínimo de instancias	90
Simultaneidad	91
Sin estado	91
Almacenamiento en caché y CDN	91
Diseño N+1	91
Resumen	92
Capítulo 5: Seguridad en el cloud	93
Seguridad	94
Ciclo de vida de seguridad	95
Seguridad de Azure	97
Seguridad de IaaS	98
Grupos de seguridad de red	98
Firewalls	100
Reducción de la superficie de ataque	102
Implementación de servidores de acceso	103
Seguridad de PaaS	103
Operations Management Suite (OMS)	103
Almacenamiento	105
Azure SQL	108
Azure Key Vaults	111
Auditoría y supervisión de seguridad	111
Azure Monitor	111
Azure Security Center	112
Resumen	114
Capítulo 6: Diseño de soluciones de IoT	115
IoT	115
Arquitectura de IoT	117
Conectividad	118
Identidad	120
Captura	120
Ingesta	121
Almacenamiento	121

Transformación	121
Análisis	122
Presentación	123
Azure IoT	123
Identidad	124
Captura	124
Ingesta	125
Almacenamiento	125
Transformación y análisis	126
Presentación	127
Centros de IoT	127
Protocolos	128
Registro de dispositivos	128
Administración de mensajes	130
Mensajería de dispositivo a cloud	130
Mensajería de cloud a dispositivo	131
Seguridad	132
Seguridad en IoT	132
Escalabilidad	133
Edición SKU	133
Unidades	135
Alta disponibilidad	135
Resumen	136
Capítulo 7: Diseño e implementación de soluciones de datos	137
Azure SQL	138
Disponibilidad de Azure SQL	140
Seguridad de Azure SQL	141
Grupos elásticos	142
Escalabilidad horizontal de Azure SQL	142
Stream Analytics	144
Orígenes de datos	146
Integración de datos	146
Transformación de datos	146
Motor Stream Analytics	146
Almacenamiento y presentación	147
Arquitectura	147
Azure Data Factory	148
Orígenes de datos	149
Transformación de datos	151
Publicación y presentación	151
Uso de Data Factory	151

Azure Data Lake	162
Almacén de Azure Data Lake	162
Data Lake Security	164
Rendimiento de Data Lakes	164
Azure Data Lake Analytics	165
Azure SQL Data Warehouse	165
Table Storage	169
Resumen	170
Capítulo 8: Diseño e implementación de soluciones sin servidor	171
Una breve historia sobre el tema "sin servidor"	172
Sin servidor	174
Principios de la tecnología sin servidor	175
Menor coste	175
Controlada por eventos	175
Responsabilidad individual	175
Rápida ejecución	175
Azure Functions o funciones como un servicio - FaaS	175
Azure Functions Runtime, enlaces y desencadenadores	176
Tiempo de ejecución de la función de Azure	176
Enlaces y desencadenadores de funciones de Azure	176
Proxies de la función de Azure	179
Supervisión	180
Autenticación y autorización	180
Configuración de la función de Azure	182
Configuración de la plataforma	183
Configuración de la función de servicios de aplicaciones	184
Planes de coste de la función de Azure	184
Ventajas de Azure Functions	185
Casos de uso de Azure Functions	188
Implementación de microservicios	189
Integración entre varios puntos de conexión	189
Procesamiento de datos	189
Integración de aplicaciones heredadas	189
Trabajos programados	190
Gateways de comunicación	190
Tipos de funciones en Azure	190
Cómo crear tu primera función de Azure	190
Cómo crear una función impulsada por eventos	196
Creación de una arquitectura conectada con funciones	200
Resumen	204

Capítulo 9: Diseño de políticas, bloqueos y etiquetas	205
Etiquetas de Azure	206
Etiquetas con PowerShell	208
Etiquetas con la plantilla de ARM	208
Grupos de recursos y recursos	209
Políticas de Azure	209
Políticas integradas	210
Lenguaje de la política	210
Campos permitidos	212
Bloqueos de Azure	213
Azure RBAC	215
Roles personalizados	217
¿En qué se diferencia de RBAC?	218
Ejemplos de la implementación de funciones de gestión de Azure	218
Contexto	218
Control de acceso basado en roles	219
Resumen	219
Capítulo 10: DevOps en Azure	221
¿Qué es DevOps?	222
Prácticas de DevOps	225
Administración de configuración	226
Desired State Configuration	228
Chef, Puppet y Ansible	228
Plantillas de Azure Resource Manager	228
Integración continua	229
Automatización de compilación	230
Automatización de pruebas	231
Empaquetado	231
Implementación continua	231
Implementación del entorno de prueba	233
Automatización de pruebas	233
Implementación del entorno de ensayo	233
Pruebas de aceptación	233
Implementación en producción	233
Entrega continua	234
Aprendizaje continuo	234
Visual Studio Team Services	235
Control de versiones de Team Foundation	237
GIT	238
Preparación de DevOps	238
Aprovisionamiento de cuenta de VSTS	240
Aprovisionamiento de Azure Key Vault	241

Aprovisionamiento de un servidor de administración de configuración	241
Aprovisionamiento de Log Analytics	241
Cuenta de Azure Storage	242
Imágenes	242
Herramientas de supervisión	242
Herramientas de administración	242
DevOps para soluciones de PaaS	243
Azure App Services	244
Ranuras de implementación	245
Azure SQL	245
Canalización de versión y compilación	245
DevOps para soluciones basadas en máquinas virtuales (IaaS)	246
Máquina virtual de Azure	246
Equilibrador de carga público de Azure	247
Canalización de compilación	247
Canalización de versión	248
DevOps para soluciones basadas en contenedor (IaaS)	249
Contenedores	250
Docker	250
DockerFile	250
Canalización de compilación	251
Canalización de versión	251
Azure Automation	252
Aprovisionamiento de una cuenta de Azure Automation	253
Creación de configuración DSC	254
Importación de la configuración DSC	255
Compilación de la configuración DSC	256
Asignación de la configuración a los nodos	256
Navegar por el servidor	257
Azure para DevOps	258
Resumen	260
Capítulo 11: Cost Management	261
Descripción de la facturación	262
Facturación	266
Consumidores con un contrato EA (Enterprise Agreement)	267
Uso y cuotas	267
Proveedores de recursos	268
API de uso y facturación	269
Modelos de precios de Azure	269
Ventajas híbridas de Azure	270
Instancias de máquina virtual reservadas de Azure	270

Cuentas de pago por uso	270
Contratos EA (Enterprise Agreement)	271
Proveedor de soluciones en el cloud	271
Calculadora de precios de Azure	271
Prácticas recomendadas	273
Prácticas recomendadas de computación	274
Prácticas recomendadas de almacenamiento	275
Prácticas recomendadas de PaaS	275
Prácticas recomendadas generales	276
Resumen	276
Capítulo 12: Supervisión y auditoría	277
Supervisión	278
Supervisión de Azure	279
Registros de actividad de Azure	279
Registros de diagnóstico de Azure	279
Registros de aplicación de Azure	280
Registros de sistema operativo invitado y host	280
Azure Monitor	281
Azure Application Insights	281
Azure Log Analytics	281
Application Insights	281
Aprovisionamiento	282
Log Analytics	286
Aprovisionamiento	286
Agentes de OMS	289
Búsqueda	290
Soluciones	291
Alertas	292
Ejecución de runbooks en alertas	296
Integración de Power BI	303
Resumen	305

Prefacio

Azure, el cloud de Microsoft, es una plataforma de cloud madura y en continuo crecimiento. Está ganando muchísimo impulso, tracción y popularidad y sigue siendo la plataforma de cloud preferida de muchas empresas. Azure es una plataforma muy grande, pero, detrás de ella, hay cientos de recursos y servicios que hacen que esa magia se convierta en realidad. Todos estos recursos y servicios se proporcionan a los usuarios de manera uniforme a través de Azure Resource Manager. Una plataforma de cloud debe respetar a los usuarios y las reglas soberanas de cada país relativas a la seguridad y los datos. Azure cuenta con más de 35 centros de datos en todo el mundo y este número sigue creciendo año tras año. Azure dispone de la mayoría de las certificaciones de seguridad que existen actualmente en el sector. Azure proporciona diferentes niveles de control sobre la implementación utilizando diversos modelos, como el de Infraestructura como servicio, Plataforma como servicio y Software como servicio. También ofrece recursos y características muy valiosos para implementar el cloud híbrido. De hecho, con el lanzamiento de Azure Stack, Azure se ha convertido en una de las plataformas más maduras y más completas para realizar implementaciones híbridas. Azure es un cloud abierto, que permite ejecutar en él cualquier sistema operativo, cualquier lenguaje de programación y cualquier tiempo de ejecución. Azure es flexible y ofrece múltiples recursos y opciones para implementar funcionalidades similares, aunque tengan algunas diferencias. Azure proporciona numerosos modelos de coste y uso y cubre prácticamente cualquier tipo de cliente, ya sea en modo pago por uso, con contratos EA (Enterprise Agreement) o con un modelo de proveedor de soluciones en el cloud. Asimismo, tiene diversas ofertas, como instancias de VM reservadas y ventajas híbridas de Azure para reducir los costes generales de la implementación. Azure dispone de valiosas herramientas para garantizar que los consumidores puedan automatizar sus implementaciones y también iniciar su viaje de DevOps. DevOps es un paradigma emergente y Azure proporciona todas las características para poder implementarlo.

Con tantas opciones, recursos y modelos de implementación diferentes, es importante que los usuarios de Azure entiendan el propósito, la importancia y la utilidad de cada recurso al nivel de la arquitectura y en qué se diferencian de otros recursos similares. En función de los requisitos, se deberían implementar los recursos adecuados. Una arquitectura para una solución en el cloud consta de varios recursos. La elección de recursos, su configuración y la interacción se deben diseñar meticulosamente y de la manera apropiada. Azure ofrece plataformas avanzadas, como IoT, sin servidor y de big data. Se trata de tecnologías emergentes y este libro las abarca todas.

Azure ofrece casi todo tipo de servicios para satisfacer las necesidades informáticas de cualquier organización y es importante introducirnos en ellas utilizando la estrategia y la arquitectura adecuadas. Este libro es un intento de ir en esa dirección, para proporcionar a sus lectores suficiente munición para diseñar y construir sus soluciones y, para ello, se abarcarán temas como los patrones de diseño, la alta disponibilidad, la seguridad, la escalabilidad, la administración de costes, la supervisión y la auditoría. Cada uno de los temas de todos los capítulos de este libro podrían ser objeto de un libro completo. Ha resultado extremadamente difícil resumir los aspectos de la arquitectura, las prácticas recomendadas y el uso de las características de Azure en un solo capítulo. Insto a todos los lectores a repasar cada capítulo y leer la documentación online de Microsoft relacionada con cada uno de ellos para obtener más información.

Temas tratados en este libro

El capítulo 1, *Introducción*, presenta la computación en el cloud como una nueva estrategia y paradigma. El tema central de este libro es Azure y empieza en la misma introducción. Proporciona detalles sobre IaaS, PaaS y una introducción a algunas de las características importantes que ayudan a la hora de diseñar soluciones. Presenta Azure Resource Manager y los grupos de recursos. También describe importantes recursos de Azure, como la computación, la red, el almacenamiento, las funciones, IoT, los servicios de datos y las herramientas de automatización y lenguajes.

El capítulo 2, *Patrones de diseño de Azure*, trata sobre los patrones de cloud de Azure relacionados con las redes virtuales, las cuentas de almacenamiento, las regiones y los conjuntos de disponibilidad. También analiza brevemente patrones de cloud que ayudan a implementar la escalabilidad y el rendimiento. Los patrones de mensajería ayudan a crear soluciones fiables. Estos patrones también se explican en este capítulo.

El capítulo 3, *Diseño de alta disponibilidad*, se centra en describir las características de alta disponibilidad que tiene Azure. Las empresas necesitan tener alta disponibilidad para sus implementaciones. Este capítulo ofrece una base sólida sobre los conceptos de alta disponibilidad y ayuda al usuario tomar decisiones informadas relacionadas con las estrategias de implementación de IaaS y PaaS.

El *capítulo 4, Implementación de la escalabilidad*, se centra en el diseño de soluciones que pueden aumentar y reducir automáticamente los recursos disponibles en función de su consumo actual para mantener sus niveles de rendimiento. Azure proporciona conjuntos de escalado de máquinas virtuales (VMSS) para implementar soluciones altamente escalables. Este capítulo se centra en la implementación y la arquitectura basados en VMSS. También se describe la escalabilidad basada en PaaS y sus estrategias.

El *capítulo 5, Seguridad del cloud*, introduce conceptos importantes desde el punto de vista de la seguridad. La seguridad es muy importante en cualquier implementación de cloud. Azure ofrece grupos de seguridad de red, firewalls, NAT, un centro de seguridad y características de almacenes de claves para implementar aplicaciones de ciberseguridad. En este capítulo, se proporciona información acerca de estas características y se crea una solución que las utiliza.

El *capítulo 6, Diseño de soluciones de IoT*, ofrece información detallada sobre la implementación de una solución de IoT con el cloud de Azure. El cloud de Azure proporciona una plataforma de IoT completa para el desarrollo de soluciones basadas en dispositivos. Este capítulo explica cómo crear soluciones basadas en IoT utilizando el cloud de Azure. También se describen los aspectos de la arquitectura que todos los arquitectos deben tener en cuenta al crear una solución. En este capítulo, se explican temas relacionados con centros IoT, centros de eventos, registro de dispositivos, una conversación entre un dispositivo y la plataforma y su registro y enrutamiento a los destinos apropiados.

El *capítulo 7, Diseño e implementación de soluciones de datos*, está dedicado al almacenamiento de datos y los servicios. Azure cuenta con numerosas características relacionadas con los servicios de datos. Este capítulo se centra en proporcionar ideas acerca de qué características y recursos deben utilizarse para diferentes tipos de soluciones y sus pros, contras y ventajas. Parte de este capítulo trata de una arquitectura completa para la ingestión, limpieza y filtrado de datos y su almacenamiento en almacenes de datos apropiados, tales como Data Lake y Cosmos DB, para luego enviar esos datos a Power BI para visualizarlos.

El *capítulo 8, Diseño e implementación de soluciones sin servidor*, se centra en la computación sin servidor. Las funciones de Azure conforman una plataforma versátil para albergar funcionalidades de pequeñas empresas como funciones y ayudar a crear soluciones juntas. En este capítulo, se explica el paradigma sin servidor, las funciones de Azure, sus capacidades, la creación de soluciones mediante la combinación de múltiples funciones, los disparadores y los parámetros, además de las diferentes fuentes de entradas y salidas.

El capítulo 9, *Diseño de políticas, bloqueos y etiquetas*, se centra en el uso de las características de administración que proporciona Azure para realizar y administrar mejor las implementaciones. Las etiquetas ayudan al añadir información de metadatos adicional a los recursos de Azure. También ayudan proporcionando información de la arquitectura relativa a los recursos de Azure. En este capítulo, se proporcionan directrices de diseño para definir etiquetas para las implementaciones. También se proporcionan detalles sobre las políticas y los bloqueos para restringir y controlar los recursos de Azure en lo referente a su ubicación, uso, tamaño, accesibilidad y permisos, entre otras cosas. Proporcionar control de administración sobre los recursos de Azure es un concepto importante.

El capítulo 10, *DevOps en Azure*, está dedicado a DevOps. El cloud de Azure dispone de herramientas, utilidades y soporte de script muy valiosos para utilizar la automatización en DevOps. Azure permite el uso de plantillas de Azure Resource Manager, la configuración de estado deseado, PowerShell, la API de Rest y tecnologías de código abierto, como Chef, Python y Linux, para crear la automatización completa de la integración, la entrega y la implementación continuos. La Infraestructura como código y la administración de la configuración también se admiten de forma inherente mediante características de Azure, como la automatización. Este capítulo se centra en la creación de procesos CI/CD y la administración de la configuración para recursos de Azure utilizando VSTS.

El capítulo 11, *Cost Management*, se centra en un ángulo un poco diferente en comparación con los otros capítulos de este libro. No es un capítulo técnico, sino que analiza diversas formas de reducir el coste de las implementaciones en Azure. Este capítulo explica cómo se calcula el coste de implementación en Azure usando su calculadora de costes. También demuestra cómo el cambio de ubicación, tamaño y tipo de recursos puede afectar al coste de la solución y también proporciona las prácticas recomendadas para reducir el coste general de las implementaciones de Azure.

El capítulo 12, *Supervisión y auditoría*, explica cómo los servicios de Azure, tales como Operational Insights y Application Insights, proporcionan capacidades de supervisión y auditoría. En este capítulo, aprenderemos a configurarlos y a utilizarlos para supervisar los recursos de Azure y realizar acciones basadas en ellos. Este capítulo también se centra en la creación de soluciones de supervisión para las implementaciones de cloud de Azure.

Qué necesitas para este libro

En este libro, se supone que el lector tiene un nivel de conocimiento básico sobre computación en el cloud y Azure. Todo lo que necesitas para usar este libro es una suscripción válida a Azure y conexión a Internet. Un sistema operativo Windows 10 con 4 GB de RAM es suficiente para usar PowerShell y ejecutar plantillas de ARM.

A quién está destinado este libro

Para aprovechar el contenido de este libro, es deseable tener un nivel de conocimiento básico del cloud y Azure. Si crees que no posees ese nivel, siempre es posible ponerse al día con los requisitos básicos leyendo rápidamente la documentación de Azure sobre los componentes más importantes en <https://docs.microsoft.com/azure/>. En esencia, este libro está destinado los arquitectos de cloud, desarrolladores, consultores e ingenieros de DevOps que utilizan Azure para prestar sus servicios a clientes finales y empresas. Si también estás dispuesto a crear soluciones completas en Azure, este libro es ideal para ti. Si ya tienes alguna experiencia con la arquitectura en Azure, este libro puede ayudarte a profundizar en ella de una forma acelerada.

Convenciones

En este libro, encontrarás diversos estilos de texto que sirven para distinguir los distintos tipos de información. Estos son algunos ejemplos de estos estilos y la explicación de su significado.

Las palabras de código en el texto, los nombres de tablas de base de datos, los nombres de las carpetas, los nombres de archivo, las extensiones de archivo, las rutas de acceso, las URL ficticias, las entradas del usuario y los identificadores de Twitter se muestran de la siguiente forma: "Podemos incluir otros contextos mediante el uso de la directiva `include`."

Un bloque de código aparece de la siguiente forma:

```
Import-DscResource -ModuleName 'PSDesiredStateConfiguration'  
Node WebServer {  
    WindowsFeature IIS  
    {  
        Name = "Web-Server"  
        Ensure = "Present"  
    }  
}
```

Los términos nuevos y **las palabras importantes** se muestran en negrita. Las palabras que ves en la pantalla, por ejemplo, en menús o cuadros de diálogo, aparecen en texto como este: "El primer paso es crear un recurso de fábrica de datos. Una vez creado, haz clic en el botón **Copiar datos**."



Las advertencias o las notas importantes aparecen en un cuadro como este.



Los consejos y trucos aparecen de esta forma.

Comentarios del lector

Los comentarios de nuestros lectores siempre son bienvenidos. Cuéntanos lo que opinas acerca de este libro, lo que te ha gustado y lo que no. Los comentarios de los lectores son importantes para nosotros, ya que nos ayudan a crear títulos que puedas aprovechar mejor.

Para enviarnos comentarios generales, simplemente escríbenos un correo electrónico a feedback@packtpub.com mencionando el título del libro en el asunto de tu mensaje.

Si hay un tema sobre el que eres experto y estás interesado en escribir o colaborar en un libro, consulta nuestra guía de autores en www.packtpub.com/authors.

Atención al cliente

Ahora que eres el orgulloso propietario de un libro de Packt, tenemos varias cosas que pueden ayudarte a aprovecharlo al máximo.

Descarga las imágenes en color de este libro

También te ofrecemos un archivo PDF que tiene imágenes en color de las capturas de pantalla/diagramas que se utilizan en este libro. Las imágenes en color te ayudarán a entender mejor los cambios en la salida. Puedes descargar este archivo desde https://www.packtpub.com/sites/default/files/downloads/AzureforArchitects_ColorImages.pdf.

Fe de erratas

Aunque hemos tenido muchísimo cuidado para asegurarnos de que nuestro contenido sea correcto, siempre se producen errores. Si encuentras un error en uno de nuestros libros (como un error en el texto o en el código), te agradeceríamos que nos avisaras. De esta forma, puedes evitar la frustración de otros lectores y nos ayuda a mejorar las versiones posteriores de este libro. Si encuentras alguna errata, comunícanoslo en <http://www.packtpub.com/submit-errata>. Para ello, selecciona el libro, haz clic en el enlace del **formulario de envío de erratas** e introduce los detalles de la errata. Una vez que se hayamos verificado la errata, la aceptaremos y la subiremos a nuestro sitio web o la añadiremos a una lista de erratas existentes en la sección Fe de erratas de dicho título.

Para ver las erratas ya publicadas, ve a <https://www.packtpub.com/books/content/support> y escribe el nombre del libro en el campo de búsqueda. La información requerida aparecerá en la sección **Fe de erratas**.

Piratería

La piratería de material con copyright en Internet es un problema continuo en todos los medios. En Packt, nos tomamos muy en serio la protección de nuestros derechos de copyright y licencias. Si te encuentras con copias ilegales de nuestros trabajos en cualquier formato en Internet, te rogamos que nos indiques la dirección o el nombre del sitio web inmediatamente para poder remediarlo.

Ponte en contacto con nosotros en copyright@packtpub.com indicando el enlace al presunto material pirateado.

Te agradecemos tu ayuda en la protección de nuestros autores y nuestra capacidad para ofrecerte contenido valioso.

Preguntas

Si tienes algún problema con cualquier aspecto de este libro, ponte en contacto con nosotros en questions@packtpub.com y haremos todo lo que podamos para solucionar el problema.

1

Introducción

Cada varios años, surgen innovaciones tecnológicas que cambian todo el panorama y el ecosistema que las rodea. Si nos remontamos en el tiempo, las décadas de 1970 y 1980 fueron la época de los grandes sistemas centrales. Estos sistemas eran enormes; ocupaban prácticamente la totalidad de grandes salas y se encargaban de llevar a cabo casi todo el trabajo informático. Resultaba difícil adquirir uno y también había que dedicarle mucho tiempo. Las empresas solían pedirlos meses antes de que pudieran contar con una instalación de sistema central operativa.

La primera mitad de la década de 1990 fue la era de los ordenadores personales y de Internet. Los ordenadores redujeron considerablemente sus dimensiones y pasaron a ser más fáciles de adquirir que antes. Tanto los ordenadores personales como la innovación que supuso Internet cambiaron por completo el sector informático. La gente disponía de un escritorio a través del cual podían ejecutar varios programas y conectarse a Internet. El auge de Internet también derivó en el aumento de las implementaciones de soluciones cliente-servidor. Ahora, hay servidores centralizados que alojan aplicaciones y servicios a los que puede acceder cualquier persona que tenga una conexión a Internet a nivel mundial. Este fue también un momento en que la tecnología de los servidores adquirió una gran importancia. Windows NT, Windows 2000 y Windows 2003 se lanzaron durante este tiempo.

La innovación más notable de la década del 2000 fue el surgimiento y la adopción de dispositivos, especialmente de smartphones, y, con ellos, de un sinfín de aplicaciones. Las aplicaciones podían conectarse con servidores centralizados en Internet y operar sin problemas. Los usuarios ya no dependían de navegadores para que funcionaran. Todos los servidores solían estar autoalojados o se encontraban alojados con un proveedor de servicios, como un **proveedor de acceso a Internet (ISP)**.

Los usuarios no gozaban de mucho control sobre sus servidores. Numerosos consumidores y sus implementaciones formaban parte del mismo servidor sin que los propios consumidores lo supieran.

Sin embargo, algo más estaba sucediendo hacia mitad de la década del 2000: el auge de la computación en el cloud, tecnología que, nuevamente, transformó por completo el sector de las TI. No obstante, al principio la adopción fue lenta y el público se acercó a esta solución con precaución; ya sea porque la tecnología estaba aún en ciernes y todavía tenía que madurar o porque el público contaba con múltiples y variadas opiniones acerca de ella.

A pesar de ello, actualmente la computación en el cloud es una de las tecnologías más prometedoras y con un gran potencial: independientemente del tamaño, todas las empresas y organizaciones la han adoptado como parte de su estrategia de TI. Hoy en día, resulta difícil mantener una conversación significativa sin incluir la computación en el cloud en los debates sobre soluciones generales.

La computación en el cloud, o simplemente el cloud, hace referencia a la disponibilidad de recursos en Internet. Estos recursos se ponen a disposición de los usuarios en Internet en forma de servicios. Por ejemplo, el almacenamiento está disponible para los usuarios a demanda a través de Internet para que estos puedan almacenar sus archivos o documentos, entre otros. En este ejemplo, el almacenamiento es un servicio prestado por un proveedor de cloud.

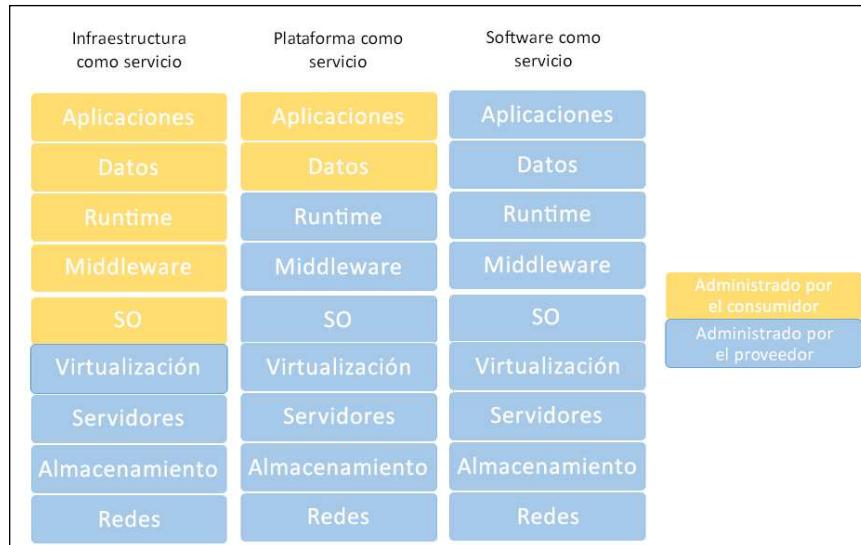
Un proveedor de cloud es una empresa o consorcio de empresas que brindan servicios en el cloud a otras empresas y a los consumidores. Organizan y gestionan los servicios en nombre del usuario. Son responsables de permitir y mantener el buen estado de los servicios. Por lo general, se dispone de centros de datos de gran tamaño distribuidos por todo el mundo abiertos por los proveedores de cloud, los cuales se encargan de atender las demandas de las TI de los usuarios.

Entre los recursos de cloud podrían incluirse la prestación de servicios de hosting o el suministro de infraestructuras a demanda, tales como equipos, redes e instalaciones de almacenamiento para su explotación por parte de los usuarios. Este tipo de servicio de cloud también suele denominarse **Infraestructura como servicio**.

Hay tres tipos de servicios prestados por el cloud en función de su nivel de abstracción y grado de control sobre estos servicios por parte de los usuarios y los proveedores de cloud:

- Infraestructura como servicio (popularmente conocida como IaaS)
- Plataforma como servicio (popularmente conocida como PaaS)
- Software como servicio (popularmente conocido como SaaS)

Los servicios de IaaS, PaaS y SaaS difieren según el nivel de control entre los consumidores y los proveedores de cloud. Con el SaaS, el proveedor de cloud tiene un control casi total sobre los servicios, mientras que el consumidor controla únicamente sus datos y la aplicación. De igual forma, el proveedor de cloud tiene mayor control con un servicio de tipo IaaS en comparación con el consumidor.



Servicios cloud: IaaS, PaaS, SaaS

La figura de arriba muestra las tres categorías de servicios disponibles a través de los proveedores de cloud y las capas que conforman cada servicio. Estas capas se apilan verticalmente unas sobre otras y cada una de ellas presenta un color diferente en función de la persona que la gestiona: el consumidor o el proveedor. En la figura podemos ver que para IaaS, el proveedor es responsable de proporcionar, controlar y administrar las capas que abarcan desde la capa de la red hasta la capa de virtualización. Del mismo modo, para la PaaS, es el proveedor quien controla y administra desde la capa de hardware hasta la capa de tiempo de ejecución, mientras que el consumidor controla únicamente las capas de aplicación y datos.

Infraestructura como servicio

Como su propio nombre sugiere, el servicio de tipo IaaS se compone de servicios de infraestructuras prestados por un proveedor de cloud. Este servicio incluye el hardware físico y su configuración, el hardware de red y su configuración, el hardware de almacenamiento y su configuración, los equilibradores de carga, los procesos y la virtualización. Cualquier capa por encima de la virtualización es responsabilidad del consumidor. El consumidor puede decidir utilizar la infraestructura subyacente proporcionada de la manera que mejor se adapte a sus requisitos. Por ejemplo, los consumidores pueden consumir el almacenamiento, la red y la virtualización para aprovisionar máquinas virtuales sobre estos. En este sentido, es responsabilidad del consumidor administrar y controlar las máquinas virtuales y el software que se implemente en estas.

Plataforma como servicio

El servicio tipo PaaS permite a los consumidores implementar sus aplicaciones y servicios en la plataforma proporcionada, lo que se traduce en la explotación del tiempo de ejecución subyacente, del software intermedio y de los servicios. El proveedor de cloud presta desde los servicios de infraestructura hasta los de tiempo de ejecución. Los consumidores no pueden aprovisionar máquinas virtuales, ya que no pueden acceder a ellas ni controlarlas. En cambio, solo pueden controlar y administrar sus aplicaciones. Se trata de un método de desarrollo e implementación comparativamente más rápido, dado que ahora el consumidor puede centrarse en la implementación y en el desarrollo de aplicaciones. Entre algunos ejemplos de Plataforma como servicio se incluyen Azure Automation, Azure SQL y Azure App Services.

Software como servicio

Software como servicio ofrece un control total del servicio al proveedor de cloud. El proveedor de cloud aprovisiona, configura y administra todo: desde la infraestructura hasta la aplicación. Incluye el aprovisionamiento de infraestructura, la implementación y la configuración de aplicaciones, al tiempo que brinda acceso a las aplicaciones al consumidor. El consumidor no controla ni administra la aplicación y puede utilizar y configurar únicamente algunas partes de la aplicación. Lo único que puede controlar son sus datos y la configuración. Por norma general, las aplicaciones multiinquilino son utilizadas por varios consumidores, tales como Office 365 y Visual Studio Team Services, ambos ejemplos de SaaS.

En los últimos años, se viene observando un crecimiento exponencial en la adopción de soluciones de cloud. Mientras que la mayor parte del crecimiento inicial procedía de pequeñas y medianas empresas, actualmente son las grandes corporaciones las que están adoptando esta tecnología. Esto se debe principalmente a los factores clave que se mencionan a continuación:

- **Rentabilidad:** cloud ayuda en la eliminación de gastos de capital y, en su lugar, solo conlleva costes operativos. Los usuarios pueden dejar de comprar hardware físico, costosas licencias de software e instalar centros de datos de gran tamaño. Todas estas soluciones están disponibles en el cloud sin que ello suponga un gasto para el usuario.
- **Capacidad y escalabilidad ilimitadas:** el cloud brinda una disponibilidad ilimitada de recursos. Esto insta a las organizaciones a implementar sus cargas de trabajo en este tipo de solución, ya que no se ven restringidas por limitaciones de la disponibilidad del hardware.
- **Elasticidad:** la computación en el cloud es flexible por naturaleza. Los consumidores pueden reducir o aumentar su presencia en el cloud cómodamente según sus necesidades y con ayuda de una interfaz de usuario fácil de usar. No hay gastos por adelantado, limitaciones de disponibilidad de recursos ni retrasos en caso de que desee realizar cualquier reducción o ampliación.

- **Pago por uso:** con el cloud, se eliminan los gastos de capital y las organizaciones solo pagan por lo que usan, lo que se traduce en una máxima rentabilidad de la inversión. Las organizaciones no tienen que crear infraestructuras adicionales para alojar sus aplicaciones en momentos de máxima demanda.
- **Mayor velocidad y optimización:** el cloud ofrece aplicaciones listas para usar, así como un aprovisionamiento y una implementación de entornos más rápidos. Además, las organizaciones reciben servicios mejor gestionados por parte de sus proveedores de cloud con unos acuerdos de nivel de servicio superiores.

¿Qué es Azure?

Según la Wikipedia:

"Azure es un servicio en la nube ofrecido como servicio y alojado en los Data Centers de Microsoft. Windows Azure es una plataforma general que tiene diferentes servicios para aplicaciones, desde servicios que alojan aplicaciones en alguno de los centros de procesamiento de datos de Microsoft para que se ejecute sobre su infraestructura [...] hasta servicios de comunicación segura y federación entre aplicaciones".

Evidentemente, Azure ofrece todos los beneficios del cloud, pero también es un cloud abierto y flexible. El cloud de Azure es compatible con una gran variedad de sistemas operativos, lenguajes, herramientas, plataformas, utilidades y marcos. Es compatible con Linux y Windows, SQL Server, MySQL, Postgres y más, C#, Python, Java, Node.js, Bash y más lenguajes, las bases de datos MongoDB y DocumentDB NoSQL, así como desde Jenkins hasta VSTS como herramientas de integración continua. La idea que subyace tras este ecosistema es brindar a los usuarios la posibilidad de elegir con libertad el lenguaje, la plataforma, el sistema operativo y las bases de datos, además de la opción de elegir el almacenamiento, las herramientas y las utilidades. Los usuarios no deben verse limitados desde el punto de vista tecnológico; por el contrario, deben ser capaces de construir y centrarse en su solución de negocios y Azure les proporciona toda una oferta tecnológica de primer nivel. Azure es compatible con la oferta tecnológica que escoja el usuario.

Por ejemplo, Azure proporciona disponibilidad para todos los entornos de bases de datos populares (ya sean de código abierto o comerciales). Azure ofrece el servicio PaaS Postgres, MySQL y Azure SQL. Ofrece, además, un ecosistema de Hadoop y HDInsight, un servicio PaaS basado íntegramente en Apache Hadoop. También permite la implementación de Hadoop on Linux VM para consumidores que opten por un tipo de servicio de IaaS. Asimismo, Azure presta un servicio de caché de Redis y admite otros entornos de bases de datos populares, como MongoDB, Couchbase, Oracle y muchos otros, en forma de implementación de IaaS.

Introducción

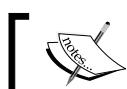
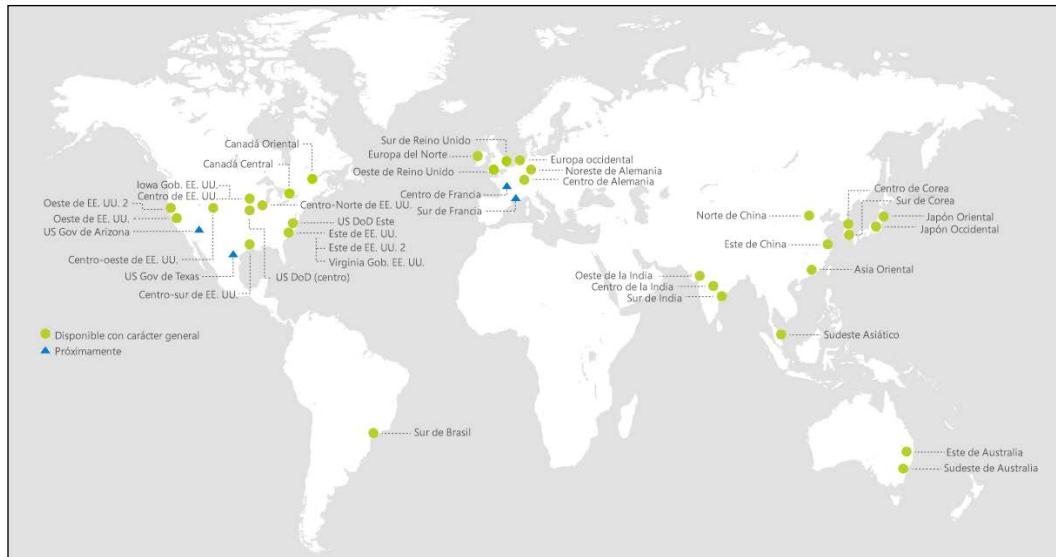
El número de servicios aumenta por días y en la siguiente imagen solo figura el completo conjunto de servicios prestados por Azure. No todos los servicios aparecen en esta imagen y su número sigue aumentando.



Servicios de Azure

Azure también ofrece un concepto único de computación en el cloud: el cloud híbrido. Este concepto engloba una estrategia de implementación en la que un subconjunto de los servicios se implementa en un cloud público; mientras que los demás servicios se implementan en un centro de datos o cloud privado on-premise. Hay una conexión de **red privada Virtual (VPN)** entre el cloud público y el privado. Azure brinda a los usuarios la flexibilidad de dividir e implementar su carga de trabajo tanto en clouds públicos como en centros de datos on-premise.

Azure cuenta con centros de datos en todo el mundo y combina dichos centros de datos para formar regiones. Cada región dispone de varios centros de datos para asegurar una recuperación antes desastres rápida y eficiente. En el momento en que hemos redactado este documento, hay 38 regiones en todo el mundo. Esto otorga a los usuarios flexibilidad para implementar sus servicios en la ubicación y las regiones que prefieran. Además, pueden combinar estas regiones para implementar una solución resistente a desastres e implementarla cerca de su base de consumidores.



Azure también tiene clouds independientes para Alemania, China y diferentes gobiernos.



Azure como un cloud inteligente

Azul no es solo un cloud; es un cloud inteligente. Ahora, tal vez se pregunte qué es un cloud inteligente. El público consume potencia de computación principalmente debido a dos razones: buscan algo y, después de encontrar lo que estaban buscando, actúan sobre ello. Toda la potencia de computación está relacionada con estos dos propósitos. Azure proporciona infraestructura y servicios para invertir millones y miles de millones de registros con un procesamiento a hiperescala. Proporciona múltiples petabytes de almacenamiento de datos. Proporciona un host de servicios interconectados que pueden pasarse los datos entre sí. Al contar con tales capacidades, se pueden procesar datos para generar información y conocimiento significativos. Hay varios tipos de información que pueden generarse a través del análisis de los datos:

- **Descriptivo:** este tipo de análisis proporciona información sobre lo que está sucediendo o sucedió en el pasado.
- **Predictivo:** este tipo de análisis proporciona información sobre lo que va a suceder en el futuro cercano o el futuro.

- **Preceptivo:** este tipo de análisis proporciona información sobre qué se debe hacer para mejorar o prevenir el suceso actual o futuro.
- **Cognitivo:** este análisis realmente ejecuta las acciones determinadas por el análisis prescriptivo de forma automatizada.

Si bien la información es valiosa, también es importante actuar sobre esta de forma reactiva o proactiva. Azure ofrece una valiosa plataforma para introducir grandes volúmenes de datos, procesar y aumentar los datos a través de sus completos servicios, almacenar datos en sus sistemas de almacenamiento de datos de gran tamaño, realizar análisis sobre estos, generar ideas y paneles, y, posteriormente, ejecutar acciones basadas en estos. Estos servicios están disponibles para todos los usuarios de Azure y proporcionan un ecosistema comprehensivo para crear soluciones adicionales. Debido a la gran disponibilidad de estos servicios inteligentes de Azure, los cuales se pueden combinar para crear valor significativo para los consumidores finales, las empresas están creando aplicaciones y servicios que están transformando por completo los mercados. Azure ha afirmado que los servicios cuya implementación en pequeñas y medianas empresas era comercialmente inviable ahora pueden adquirirse e implementarse fácilmente en pocos minutos.

Azure Resource Manager

Azure Resource Manager es el servicio de orquestación y la plataforma tecnológica de Microsoft que fusiona todos los componentes descritos anteriormente. Reúne grupos de recursos, recursos y proveedores de recursos de Azure para formar una plataforma de cloud cohesionada. Ayuda en el registro de los proveedores de recursos a suscripciones y regiones, pone a disposición los diferentes tipos de recursos para los grupos de recursos, permite el acceso a los recursos y las API de recursos a otros clientes y al portal, además de autenticar el acceso a los recursos. También habilita características, tales como el etiquetado, la autenticación, el **control de acceso basado en roles (RBAC)**, el bloqueo de recursos, y la aplicación de políticas para las suscripciones y sus grupos de recursos. Ofrece la misma experiencia de implementación y administración, ya sea a través de herramientas basadas en cliente o un portal, como PowerShell o una interfaz de línea de comandos.

Arquitectura de Azure Resource Manager

La arquitectura de Azure Resource Manager y sus componentes se muestra en la figura siguiente. Como podemos ver la **suscripción de Azure** se compone de varios grupos de recursos. Cada uno de estos contiene instancias de recursos que se crean a partir de tipos de recursos disponibles en el proveedor de recursos.



Arquitectura de Azure Resource Manager

ARM y ASM

ASM presenta algunas limitaciones y algunas de las principales se analizan este apartado: las implementaciones de ASM son lentas y obstructivas. Las operaciones se bloquean si ya está en curso una operación anterior:

- **Paralelismo:** el paralelismo constituye todo un desafío en ASM. No es posible ejecutar con éxito varias transacciones a la vez. Las operaciones en ASM son lineales y se ejecutan una tras otra. O bien se producen errores de funcionamiento en paralelo o bien se bloquean estas operaciones.
- **Recursos:** los recursos en ASM son aprovisionados y administrados de manera aislada unos de otros, esto es, no hay ninguna relación entre los recursos de ASM. La agrupación de servicios y recursos o su configuración conjunta no es factible.
- **Servicios cloud:** los servicios cloud constituyen el mecanismo de implementación en ASM. Dependen de grupos de afinidad y, dado su diseño y arquitectura, no pueden ampliarse.

Los permisos y roles específicos e individuales no pueden asignarse a los recursos en ASM. Los usuarios son administradores de servicios o coadministradores en la suscripción. Gozan de un control total sobre los recursos o no tienen ningún acceso a ellos. ASM no proporciona ningún tipo de soporte en la implementación. Las implementaciones son manuales o deben basarse en scripts de procesos en PowerShell o .NET.

Las API de ASM no son uniformes entre los diversos recursos.

Ventajas de ARM

ARM ofrece unas ventajas y unos beneficios diferenciados en comparación con ASM.

- **Agrupación:** ARM permite agrupar recursos en un contenedor lógico. Estos recursos pueden administrarse juntos y experimentar un ciclo de vida común como grupo. Esto facilita la identificación de recursos relacionados y dependientes.
- **Ciclo de vida común:** los recursos de un grupo tienen el mismo ciclo de vida. Estos recursos pueden evolucionar y administrarse juntos como una unidad.
- **Control de acceso basado en roles:** pueden asignarse roles y permisos específicos a recursos para ofrecer acceso individual a usuarios. Los usuarios pueden tener únicamente los derechos que se le han asignado.
- **Soporte en implementación:** ARM ofrece soporte en la implementación en términos de plantillas que permite el empleo de DevOps y de **Infraestructura como código (IAC)**. Las implementaciones son más rápidas, coherentes y predecibles.
- **Tecnología superior:** el coste y la facturación de recursos pueden administrarse como una unidad. Cada grupo de recursos puede proporcionar información sobre su uso y costes.
- **Capacidad de gestión:** ARM ofrece características avanzadas, tales como seguridad, supervisión, auditoría y etiquetado para una mejor administración de los recursos. Pueden consultar recursos basados en etiquetas. Las etiquetas también incluyen información de costes y facturación para los recursos con etiquetado similar.
- **Migración:** la migración y la actualización de los recursos resulta más fácil tanto dentro de estos como a través de grupos de recursos.

Conceptos de ARM

Con ARM, todo en Azure es un recurso. Entre algunos ejemplos de recursos se incluyen máquinas virtuales, interfaces de red, direcciones IP públicas, cuentas de almacenamiento y redes virtuales, entre muchos otros. ARM se basa en conceptos relacionados con los proveedores de recursos y los consumidores de recursos. Azure proporciona recursos y servicios a través de varios proveedores de recursos que son utilizados e implementados en grupos.

Proveedores de recursos

Estos proveedores son los responsables de los servicios que proporcionan tipos de recursos a través de Azure Resource Manager. El proveedor de recursos ocupa el nivel superior en ARM. Estos proveedores son contenedores de los tipos de recursos. Los tipos de recursos se agrupan en proveedores de recursos. Son responsables de la implementación y administración de los recursos. Por ejemplo, un proveedor de recursos denominado **Microsoft.Compute Namespace** proporciona un tipo de recurso de máquina virtual. Las operaciones de la API de REST ofrecen diferentes versiones para distinguirlas entre sí. La denominación de la versión se basa en las fechas de lanzamiento de Microsoft. Es necesario que un proveedor de recursos esté disponible en una suscripción para implementar un recurso. No todos los proveedores de recursos están disponibles en una suscripción “out of the box”. Si un recurso no está disponible en la suscripción, se debe verificar si el proveedor de recursos requerido está disponible en las regiones correspondientes. Si está disponible, el usuario puede registrarse de forma explícita en la suscripción.

Tipos de recursos

Se trata de una especificación de recurso real que define la implementación y la interfaz de la API pública. Implementa el funcionamiento y las operaciones admitidas por el recurso. Al igual que los proveedores de recursos, los tipos de recursos también evolucionan con el tiempo en términos de implementación interna y presentan varias versiones de su diseño e interfaz de API pública. Los nombres de las versiones se basan en las fechas de lanzamiento de Microsoft en forma de preview o de **disponibilidad general (GA)**. Los tipos de recursos están disponibles para una suscripción después de que un proveedor de recursos lo haya registrado. Además, no todos los tipos de recursos están disponibles en todas las regiones de Azure. La disponibilidad de un recurso depende de la disponibilidad y del registro de un proveedor de recursos en una región de Azure y debe ser compatible con la versión de API necesaria para su aprovisionamiento.

Grupos de recursos

Los grupos de recursos son un mecanismo de implementación en ARM. Son contenedores que agrupan varias instancias de recursos dentro de un límite de seguridad y administración. Un grupo de recursos se denomina de manera única en una suscripción. Los recursos pueden aprovisionarse en diferentes regiones de Azure, pero siguen perteneciendo al mismo grupo de recursos. Presta servicios adicionales a todos los recursos dentro de este. Los grupos de recursos ofrecen servicios de metadatos, tales como el etiquetado, que permiten la categorización de los recursos, la gestión de los recursos basada en políticas, el RBAC, la protección de los recursos frente a borrados accidentales o actualizaciones, y muchos servicios más. Como se ha mencionado anteriormente, cuentan con un límite de seguridad y los usuarios que no tienen acceso a un grupo de recursos determinado no pueden acceder a los recursos que este contiene. Cada instancia de recurso debe formar parte de un grupo de recursos o, de lo contrario, no se puede implementar.

Recursos e instancias de recursos

Los recursos se crean a partir de tipos de recursos y debe ser únicos dentro de un grupo de recursos. Esta exclusividad se define mediante la combinación del nombre del recurso y de su tipo. En lenguaje OOP, las instancias de recursos pueden ser objetos, mientras que los tipos de recursos pueden ser una clase. Los servicios se consumen a través de las operaciones admitidas e implementadas por las instancias de recursos. Estas definen las propiedades que deben configurarse antes de su uso. Algunas son propiedades obligatorias, mientras que otras son opcionales. Heredan la seguridad y configuración de acceso del grupo de recursos al que pertenecen. Estas asignaciones de roles y estos permisos heredados se pueden omitir en recursos individuales. Un recurso puede bloquearse de tal manera que algunas de sus operaciones puedan estar bloqueadas o no disponibles para determinados roles, usuarios y grupos a pesar de que tengan acceso a él. Pueden etiquetarse para facilitar su localización y administración.

Características de Azure Resource Manager

A continuación, se describen algunas de las principales características que ofrece Azure Resource Manager:

- **Control de acceso basado en roles:** Azure Active Directory (AAD) autentica a los usuarios para otorgar acceso a las suscripciones, los grupos de recursos y los recursos. ARM implementa OAuth y RBAC dentro de la plataforma, lo que permite la autorización y el control de acceso a los recursos, los grupos de recursos y las suscripciones según los roles asignados a un usuario o grupo. Un permiso define el acceso a las operaciones en un recurso. Estos permisos pueden permitir o denegar el acceso a un recurso. Un rol puede definirse como un conjunto de estos permisos. Los roles asignan usuarios y grupos de ADD a los permisos. Posteriormente, los roles se asignan a un ámbito, que puede ser un ámbito individual, una colección de recursos, un grupo de recursos o una suscripción. Las identidades de AAD (usuarios, grupos y principios de servicio) añadidas a un rol obtienen acceso al recurso de acuerdo con los permisos definidos en el rol. ARM ofrece varios roles de manera inmediata (“out of the box”). Proporciona roles de sistema, tales como **propietario, colaborador o lector**, entre muchos otros más. También ofrece funciones basadas en roles, como colaborador de la base de datos de SQL o colaborador de la máquina virtual, entre muchas otras. ARM permite la creación de roles personalizados.

- **Etiquetas:** las etiquetas son parejas de nombre y valor que añaden información adicional y metadatos a los recursos. Tanto los recursos como los grupos de recursos pueden marcarse con varias etiquetas. Las etiquetas ayudan en la categorización de los recursos para mejorar su visibilidad y capacidad de administración. Los recursos pueden buscarse rápidamente e identificarse fácilmente. La información de facturación y de costes pueden recopilarla los recursos que tienen las mismas etiquetas aplicadas. Si bien esta función la proporciona ARM, un administrador de TI define su uso y su taxonomía con respecto a los recursos y los grupos de recursos. Por ejemplo, la taxonomía y las etiquetas pueden definirse basándose en departamento, uso de recursos, ubicación, proyecto o cualquier otro criterio que se considere pertinente desde el punto de vista de los costes, la utilización, la facturación y la búsqueda. Posteriormente, estas etiquetas podrán aplicarse a los recursos. Sus recursos no heredan las etiquetas definidas en el nivel de grupo de recursos.
- **Políticas:** otra característica de seguridad proporcionada por ARM son las políticas. Pueden crearse políticas personalizadas para controlar el acceso a los recursos. Las políticas consisten en convenciones y reglas definidas que deben respetarse mientras se interactúa con los recursos y los grupos de recursos. La definición de políticas comprende la negación explícita de acciones en recursos o de acceso a recursos. De forma predeterminada, los accesos que no se mencionen en la definición de una política están permitidos. Estas definiciones de políticas se asignan en el nivel de recursos, grupos de recursos y suscripciones. Es importante señalar que estas políticas no sustituyen ni reemplazan el RBAC. De hecho, se complementan y trabajan en combinación con el RBAC. Las políticas se evalúan después de que un usuario es autenticado por AAD y autorizado por el servicio RBAC. Para definir las políticas, ARM proporciona un lenguaje de definición de políticas basada en JSON. Algunos de los ejemplos de definición de políticas pueden comprender el etiquetado de cada recurso aprovisionado o el aprovisionamiento exclusivo de recursos a regiones específicas de Azure.
- **Bloqueos:** las suscripciones, los grupos de recursos y los recursos se pueden bloquear para evitar el borrado accidental y actualizaciones por parte de un usuario autenticado. Los bloqueos aplicados en los niveles superiores se aplican de manera descendente en los niveles inferiores. Los bloqueos aplicados en el nivel de suscripción bloquean todos los grupos de recursos y los recursos que la conforman.
- **Multirregionalidad:** Azure proporciona varias regiones para el aprovisionamiento y el hosting de recursos. ARM permite el aprovisionamiento de los recursos en diferentes localizaciones, aunque estos pertenezcan al mismo grupo de recursos. Un grupo de recursos puede contener recursos de diferentes regiones.

- **Idempotencia:** esta función asegura la previsibilidad, estandarización y coherencia en la implementación de recursos, de forma que se garantiza que cada implementación resultará en el mismo estado de los recursos y de su configuración con independencia del número de veces que se lleve a cabo.
- **Extensibilidad:** la arquitectura de ARM proporciona una arquitectura extensible que permite la creación y conexión de nuevos proveedores de recursos y tipos de recursos en la plataforma.

Virtualización

La virtualización supuso toda una revolución que transformó por completo la manera de consultar los servidores físicos. Esta tecnología se entiende como la abstracción de un objeto físico en un objeto lógico.

La virtualización de los servidores físicos permitió la creación de varios servidores virtuales, más conocidos como máquinas virtuales. Estas máquinas virtuales consumen y comparten la misma CPU física, memoria, almacenamiento y otras soluciones de hardware con el servidor físico en el que están alojadas. Esto facilitó un aprovisionamiento más rápido y sencillo de los entornos de aplicaciones a demanda, lo que derivó en una alta escalabilidad y disponibilidad a un precio reducido. Un servidor físico podía alojar varias máquinas virtuales, cada una de las cuales contenía su propio sistema operativo y alojaba servicios en ella.

Se eliminó la necesidad de comprar servidores físicos adicionales para implementar nuevas aplicaciones y servicios. Los servidores físicos existentes eran suficientes para alojar más máquinas virtuales. Además, como parte del proceso de racionalización y gracias a la virtualización, muchos servidores físicos se unificaron reduciendo así su número.

Cada máquina virtual contiene todo el sistema operativo y se encuentra completamente aislada de otras máquinas virtuales, incluidos los hosts físicos. Aunque una máquina virtual utiliza el hardware proporcionado por el servidor de host físico, tiene control total sobre el entorno y los recursos que se le han asignado. Estas máquinas virtuales se pueden alojar en una red como un servidor físico con su propia identidad.

Azure ayuda en la creación de máquinas virtuales Linux y Windows en unos pocos minutos. Microsoft proporciona sus propias imágenes junto con imágenes de partners y la comunidad. Los usuarios pueden aportar sus propias imágenes. Las máquinas virtuales se crean utilizando estas imágenes.

Contenedores

Los contenedores son una solución perteneciente a la tecnología de la virtualización; sin embargo, estos no virtualizan un servidor físico. Por el contrario, un contenedor es una virtualización en el nivel del sistema operativo. Esto quiere decir que los contenedores comparten el kernel del sistema operativo proporcionado por el host entre sí junto con el host. Varios contenedores que se ejecuten en un host (físico o virtual) comparten el kernel del sistema operativo del host. Los contenedores garantizan que se vuelve a usar el kernel del host en lugar de tener que contar con un kernel específico para cada uno.

Los contenedores también se encuentran totalmente aislados del host y otros contenedores, como una máquina virtual. Los contenedores utilizan controladores de filtro de almacenamiento de Windows y el aislamiento de la sesión para proporcionar aislamiento de servicios del sistema operativo como el sistema de archivos, el registro, los procesos y las redes. Cada contenedor obtiene su propia copia de recursos del sistema operativo.

El contenedor tiene la percepción de que tiene recursos y un sistema operativo totalmente nuevos e intactos. Esta disposición ofrece numerosos beneficios, entre los que se incluyen los siguientes:

- Los contenedores son más rápidos de aprovisionar. No necesitan proporcionar el sistema operativo y sus servicios del kernel. Están disponibles en el sistema operativo del host.
- Los contenedores son ligeros y requieren menos recursos informáticos en comparación con las máquinas virtuales. Los contenedores ya no requieren una sobrecarga de recursos del sistema operativo.
- Los contenedores presentan unas dimensiones mucho reducidas en comparación con las máquinas virtuales.
- Los contenedores ayudan a resolver los problemas relacionados con la administración de múltiples dependencias de aplicaciones de una manera intuitiva, automatizada y sencilla.
- Los contenedores proporcionan infraestructura para definir todas las dependencias de aplicaciones en un solo lugar.

Los contenedores constituyen una parte inherente y una característica de Windows Server 2016 y Windows 10; sin embargo, se administran y se accede a ellos con un cliente de Docker y un demonio de Docker. Los contenedores pueden crearse en Azure con Windows Server 2016 por SKU como una imagen.

Introducción

Cada contenedor cuenta con un único proceso principal que debe ejecutarse para que el contenedor exista. Un contenedor se detendrá cuando termine este proceso. Además, un contenedor puede ejecutarse en modo interactivo o en modo individual como un servicio.



Arquitectura de contenedores

La figura muestra todas las capas técnicas que permiten el uso de contenedores. La capa inferior proporciona la infraestructura básica en términos de red, almacenamiento, equilibradores de carga y tarjetas de red. En la parte superior de la infraestructura se encuentra la capa informática, que consiste en un servidor físico o en servidores tanto físicos como virtuales sobre un servidor físico. Esta capa contiene el sistema operativo con capacidad para alojar los contenedores. El sistema operativo proporciona el controlador de ejecución que las capas superiores utilizan para llamar al código del kernel y a los objetos para ejecutar los contenedores. Microsoft ha creado **Host Container System Shim (HCSShim)** para la administración y creación de contenedores y utiliza los controladores de filtro de almacenamiento de Windows para la administración de las imágenes y los archivos.

Para la sesión de Windows, se aísla el entorno del contenedor. Windows Server 2016 y Windows Nano Server suministran el sistema operativo y habilitan las funciones del contenedor, además de ejecutar el motor de Docker y el cliente de Docker en el nivel de usuario. El motor de Docker utiliza los servicios de HCSShim, los controladores de filtro de almacenamiento y las sesiones para generar varios contenedores en el servidor, cada uno con un servicio, una aplicación o una base de datos.

Docker

Docker proporciona funciones de administración para los contenedores de Windows. Se compone de dos archivos ejecutables:

- Demónio de Docker
- Cliente de Docker

El demonio de Docker es el caballo de batalla para la administración de los contenedores. Se trata de un servicio de Windows encargado de administrar todas las actividades en el host relacionadas con los contenedores. El cliente de Docker interactúa con el demonio de Docker y es responsable de la captura de entradas, así como de su envío a través del demonio de Docker. El demonio de Docker proporciona el tiempo de ejecución, bibliotecas, controladores de gráficos y el motor para crear, administrar y supervisar los contenedores y las imágenes en el servidor host. También ofrece capacidades para crear imágenes personalizadas que se utilizan en la creación y el envío de aplicaciones a múltiples entornos.

Interacción con el cloud inteligente

Azure proporciona numerosas formas de conectar, automatizar e interactuar con él. Todos los métodos requieren la autenticación de los usuarios y códigos con credenciales válidas antes de que se pueden utilizar.

- Azure Portal
- PowerShell
- **Interfaz de la línea de comandos (CLI) de Azure**
- API de REST de Azure

Azure Portal

Azure Portal es el lugar ideal para comenzar. Con Azure Portal, los usuarios pueden iniciar sesión y empezar a crear y administrar recursos de Azure manualmente. El portal te proporciona una interfaz de usuario intuitiva y fácil de usar a través del explorador, además de una manera cómoda de explorar los recursos con ayuda de las **hojas**. Las hojas muestran todas las propiedades de un recurso, los registros, los costes, su relación con otros recursos, las etiquetas y las opciones de seguridad, entre muchas otras. La implementación del cloud puede administrarse por completo a través del portal.

PowerShell

PowerShell es un shell de línea de comandos basado en objetos y un lenguaje de scripting utilizado para la administración, configuración y gestión de infraestructuras y entornos. Se basa en .NET framework y proporciona capacidades de automatización. PowerShell se ha convertido en uno de los grandes protagonistas en el sector de los desarrolladores de automatización y administradores de TI a la hora de administrar y controlar el entorno de Windows. Actualmente, casi todos los entornos de Windows y numerosos entornos de Linux pueden administrarse mediante PowerShell. De hecho, casi todos los elementos que conforman Azure también pueden administrarse con PowerShell. Azure ofrece una amplia compatibilidad con PowerShell. Ofrece un módulo de PowerShell para cada proveedor de recursos que contiene cientos de cmdlets. Los usuarios pueden utilizar estos cmdlets en sus scripts para automatizar la interacción con Azure. El módulo de PowerShell de Azure está disponible a través del instalador de plataforma web, así como a través de la **Galería de PowerShell**. Windows Server 2016 y Windows 10 proporcionan los módulos PowerShellGet y de administración de paquetes para unas descargas e instalaciones rápidas y sencillas de los módulos PowerShell de la Galería de PowerShell. El módulo PowerShellGet ofrece el cmdlet `Install-Module` para descargar e instalar módulos en el sistema. La instalación de un módulo consiste simplemente en copiar archivos de módulos en las ubicaciones de módulos bien establecidas:

```
Import-Module PowerShellGet  
Install-Module -Name AzureRM (detallado)
```

Interfaz de la línea de comandos (CLI) de Azure

Azure también incluye la versión 2.0 de la CLI de Azure (Azure CLI 2.0), que se puede implementar en sistemas operativos Linux, Windows y Mac. Azure CLI 2.0 es nueva utilidad de línea de comandos de Azure para la administración de recursos de Azure. Azul CLI 2.0 se ha optimizado para la gestión y administración de los recursos de Azure desde la línea de comandos, además de para generar scripts de automatización que funcionen sobre Azure Resource Manager. La interfaz de la línea de comandos puede utilizarse para ejecutar comandos usando Bash Shell o la línea de comandos de Windows. Azure CLI es un usuario no perteneciente a Windows que goza de gran popularidad y que nos permite interactuar con Azure en sistemas Linux y Mac. Podrás encontrar los pasos para instalar Azure CLI 2 en <https://docs.microsoft.com/cli/azure/install-azure-cli?view=azure-cli-latest>.

API de REST de Azure

Todos los recursos de Azure están expuestos a los usuarios a través de los puntos de conexión de REST. Las API de la **transferencia de estado representacional (REST)** son puntos de conexión del servicio que implementan las operaciones HTTP (métodos), que permiten **crear, obtener, actualizar o eliminar (CRUD)** acceso a los recursos del servicio. Los usuarios pueden consumir estas API para crear y administrar recursos. De hecho, el mecanismo de CLI y PowerShell utiliza estas API de REST de manera interna para interactuar con recursos en Azure.

Plantillas de Azure Resource Manager

En una sección anterior, vimos características de implementación, tales como multiservicios, multirregionalidad, extensibilidad e idempotencia, proporcionadas por ARM. Las plantillas de ARM son los principales medios de aprovisionamiento de recursos en ARM. Las plantillas de ARM permiten la implementación de las características de implementación de ARM.

Las plantillas de ARM ofrecen un modelo declarativo a través del cual se especifican los recursos, su configuración, los scripts y las extensiones. Estas se basan en el formato **JavaScript Object Notation (JSON)**. Utilizan las convenciones y la sintaxis JSON para declarar y configurar los recursos. Los archivos JSON son archivos basados en texto, comprensibles para el usuario y fácilmente legibles.

Pueden almacenarse en un repositorio de códigos fuente y tienen control sobre las versiones. También son un medio de representación de la IAC que puede utilizarse para aprovisionar recursos en un grupo de recursos de Azure de manera reiterada, predecible, coherente y uniforme. Una plantilla precisa de un grupo de recursos para su implementación. Solo se puede implementar en un grupo de recursos y dicho grupo debe existir antes de ejecutarse la implementación de la plantilla. Una plantilla no puede crear un grupo de recursos.

Las plantillas se caracterizan por su flexibilidad debido a su diseño e implementación genéricos y modulares. Las plantillas brindan la posibilidad de aceptar los parámetros de los usuarios, declarar variables internas, ayudar a definir dependencias entre recursos, vincular los recursos dentro de un grupo de recursos o entre diferentes grupos de recursos y ejecutar otras plantillas. Ofrecen, además, funciones y expresiones de tipo de lenguaje de scripting que las convierten en herramientas dinámicas y personalizables en el tiempo de ejecución.

Implementaciones

PowerShell permite dos modos de implementación de plantillas:

- Incremental
- Completa

La implementación incremental añade los recursos declarados en la plantilla que no existen en un grupo de recursos, no modifica los recursos de un grupo de recursos que no forme parte de la definición de una plantilla ni tampoco modifica los recursos de un grupo de recursos que exista tanto en la plantilla como en el grupo de recursos con el mismo estado de configuración.

Por su parte, la implementación completa añade recursos declarados en una plantilla al grupo de recursos, elimina recursos que no existen en la plantilla del grupo de recursos y no modifica los recursos que existen en el grupo de recursos y en la plantilla con el mismo estado de configuración.

Resumen

Las soluciones de cloud no tienen más de 10 años. Se trata de un nuevo paradigma que todavía está en cíernes. Con el tiempo se irán agregando numerosas innovaciones y capacidades. Actualmente, Azure es uno de los principales proveedores de cloud y ofrece unas completas capacidades a través de las implementaciones híbridas, IaaS, PaaS y SaaS. De hecho, Azure Stack, una implementación del cloud privado de Microsoft, se lanzará pronto al mercado. Contará con las mismas características disponibles tanto del cloud privado como del público. En realidad, ambos se conectarán y trabajarán juntos de forma transparente y sin problemas. Resulta muy fácil comenzar a trabajar con Azure; no obstante, los desarrolladores y los arquitectos pueden caer en la trampa de no crear un diseño y una arquitectura adecuados para sus soluciones. El objetivo de este libro es servir de orientación y guía para el correcto diseño de soluciones con los recursos y servicios adecuados. Todos los servicios de Azure son un recurso. Es importante entender cómo se organizan y administran estos recursos en Azure. Este capítulo te ha ofrecido un contexto sobre Azure Resource Manager y los grupos, que constituyen el marco de trabajo básico para los bloques de construcción de recursos. Proporciona un conjunto de servicios a los recursos que ayudan a proporcionar uniformidad, estandarización y coherencia en su administración. Los servicios, como el RBAC, las etiquetas, las políticas y los bloqueos, entre muchos otros, están disponibles para cada proveedor de recursos y recurso. Azure también te brinda unas completas características de automatización para automatizar los recursos e interactuar con ellos. Herramientas como PowerShell, las plantillas de ARM y Azure CLI puede incorporarse como parte de las canalizaciones de lanzamiento, además de como herramientas de implementación y entrega continuas. Los usuarios pueden conectarse a Azure desde una gran variedad de entornos gracias a estas herramientas de automatización.

En el siguiente capítulo analizaremos algunos de los patrones más importantes que ayudan a solucionar problemas frecuentes relacionados con la implementación basada en cloud, al tiempo que garantizan la seguridad, la disponibilidad, la capacidad de ampliación y el mantenimiento a largo plazo de la aplicación.

2

Patrones de diseño de Azure

En el capítulo anterior, describimos a grandes rasgos el cloud de Azure y explicamos algunos conceptos importantes relacionados con esta solución. Este capítulo trata sobre los patrones de cloud de Azure relacionados con las redes virtuales, las cuentas de almacenamiento, las regiones y los conjuntos de disponibilidad. Estos son componentes importantes que afectan la arquitectura final que se entrega a los consumidores en términos de coste, eficiencia y productividad general. En este capítulo también se describirán brevemente patrones de cloud que ayuden en la implementación de escalabilidad y rendimiento dentro de la estructura general.

En este capítulo, abordaremos los siguientes temas:

- Diseño de red virtual de Azure
- Diseño de almacenamiento de Azure
- Zonas, regiones y conjuntos de disponibilidad de Azure
- Patrones de diseño de Azure relacionados con las siguientes prestaciones:
 - Mensajería
 - Rendimiento
 - Escalabilidad

Zonas y regiones de Azure

El cloud de Azure cuenta con grandes centros de datos interconectados en una única red de gran tamaño. Los centros de datos se agrupan según su proximidad física a las regiones de Azure. Por ejemplo, dos centros de datos en Europa occidental se encuentran disponibles para los usuarios de Azure de Europa occidental. Los usuarios no pueden controlar el centro de datos que utilizan, es decir, el centro de datos exacto de su implementación. Pueden informar a Azure sobre la región y Azure elegirá un centro de datos adecuado.

La elección de la región correcta es una decisión arquitectónica importante ya que afecta a los siguientes aspectos:

- Disponibilidad de recursos
- Cumplimiento normativo relativo a datos y privacidad
- Rendimiento de las aplicaciones
- Coste de ejecución de aplicaciones

Disponibilidad de recursos

No todos los recursos están disponibles en todas las regiones de Azure. Si la arquitectura de la aplicación exige un recurso y este no está disponible en una región, no resultará útil elegir dicha región. Por el contrario, una región debe elegirse basándose en la disponibilidad de todos los recursos que requiera la aplicación. Puede darse el caso de que el recurso no esté disponible en la fase de desarrollo de la arquitectura de la aplicación y que pueda figurar en la hoja de ruta de Azure para que esté disponible posteriormente.

Por ejemplo, Log Analytics no está disponible en todas las regiones. La transferencia de datos a Log Analytics implicaría costes de red de salida si la aplicación se implementara en una región que no tiene Log Analytics (anteriormente conocido como **Operational Management Suite**). Del mismo modo, Azure Key Vault proporciona servicios únicamente a los recursos que se encuentran en la misma región. Otro ejemplo es que una red virtual de Azure puede alojar máquinas virtuales y equilibradores de carga de las mismas regiones. Las máquinas virtuales de diferentes regiones no pueden formar parte de la misma red virtual.

Cumplimiento normativo relativo a datos y privacidad

Cada país cuenta con una regulación propia en términos de conformidad normativa sobre datos y privacidad. Algunos países son muy restrictivos a la hora de almacenar sus datos y los datos de sus ciudadanos en otro país. Estos son requisitos legales y deben tomarse en consideración durante el diseño de la arquitectura de una aplicación.

Rendimiento de las aplicaciones

El rendimiento de una aplicación depende de la ruta de red tomada por las solicitudes y de la respuesta de los usuarios. Si una región de Azure se encuentra cerca de los usuarios de la aplicación, estos obtendrán un mejor rendimiento en comparación con los usuarios que se encuentren lejos de la región de Azure. Una aplicación implementada en Europa occidental para usuarios en el sureste de Asia no ofrecerá el mismo nivel de rendimiento que una aplicación implementada en la región de Asia oriental para los usuarios en el sudeste de Asia.

Coste de ejecución de aplicaciones

El coste de los servicios de Azure puede variar entre una región y otra. Debe elegirse una región con un coste total más reducido. En este libro, hay un capítulo completo sobre la administración de costes y debe consultarse para obtener más información sobre este extremo.

Red virtual

Una **red virtual** debe concebirse como una red LAN física en la oficina o domicilio. Conceptualmente, ambas son iguales, aunque la red virtual de Azure se implementa como una red definida por software y respaldada por una infraestructura de red física gigante.

Es necesario contar con una red virtual para alojar una máquina virtual. Ofrecen un medio de comunicación seguro entre los recursos de Azure para que se conecten entre sí. Además, proporcionan una dirección IP interna para estos, así como acceso y conectividad a otros recursos, incluyendo máquinas virtuales de la misma red virtual, enrutamiento de solicitudes y conectividad con otras redes.

Una red virtual se encuentra dentro de un grupo de recursos y está alojada dentro de una región, por ejemplo, Europa occidental. La red virtual no puede abarcar varias regiones, pero puede abarcar todos los centros de datos en una región. Para lograr la conectividad entre todas las regiones, las redes virtuales pueden conectarse mediante una conectividad entre redes virtuales (VNET a VNET). La red virtual también ayuda en la conexión con un centro de datos on-premise compatible con un cloud híbrido. Hay numerosos tipos de tecnologías VPN disponibles para conectarse a centros de datos on-premises como VPN de sitio a sitio y VPN de punto a sitio. Existe también una conexión dedicada entre la red virtual de Azure y la red on-premise a través de **ExpressRoute**.

Las redes virtuales son gratuitas. Cada suscripción puede crear hasta 50 redes virtuales en todas las regiones. No obstante, esta capacidad puede aumentarse si se pone en contacto con el servicio de asistencia de Azure. No habrá ningún cargo por la transferencia de datos dentro de una VNET.



Puede obtener más información sobre los límites de las conexiones de red en <https://docs.microsoft.com/azure/azure-subscription-service-limits#networking-limits>.

Consideraciones de arquitectura para redes virtuales

Como cualquier otro recurso, las redes virtuales pueden aprovisionarse con plantillas de ARM, API de REST, PowerShell y CLI. Es muy importante planificar la topología de red en la etapa inicial para evitar problemas más adelante en el ciclo de vida de desarrollo. Esto se debe a que, una vez que la red se aprovisiona y se empiezan a utilizar los recursos, es difícil realizar modificaciones sin tiempos de inactividad. Por ejemplo, para mover una máquina virtual de una red a otra será necesario apagar dicha máquina virtual.

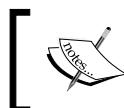
- **Regiones:** la red virtual es un recurso de Azure y recibe aprovisionamiento dentro de una región como Europa occidental. Las aplicaciones que abarcan varias regiones necesitarán redes virtuales separadas, una por cada región, además de estar conectadas utilizando la conectividad VNET a VNET. Hay un coste asociado a esta conectividad, tanto para el tráfico de entrada como el de salida. No obstante, para una red sin conectividad VNET a VNET, no hay cargos para los datos de entrada (ingreso): únicamente los cargos asociados a los datos de salida. Además, no hay cargos asociados a la transferencia de datos dentro de la red virtual.
- **DNS dedicado:** de manera predeterminada, las redes virtuales utilizan el DNS proporcionado por Azure para resolver los nombres dentro de una red virtual, además de permitir la resolución de nombres en Internet. Si la aplicación desea un servicio de resolución de nombres dedicado o desea conectarse a los centros de datos on-premises, debe proporcionar su propio servidor DNS y deben configurarse dentro de la red virtual para una resolución correcta de los nombres.
- **Rendimiento de la red:** cada red virtual tiene un nivel de rendimiento definido, y se recomienda crear una red virtual independiente para ofrecer un rendimiento constante en vez de ubicar todos los recursos relacionados con la red en una única red. Esto último podría obstruir la red.
- **Número de redes virtuales:** el número de redes virtuales se ve afectado por el número de regiones, el uso del ancho de banda por parte de los servicios, la conectividad entre regiones y la seguridad.
- **Número de subredes en cada red virtual:** las subredes ofrecen aislamiento dentro de una red virtual. También se trata de un límite de seguridad. Los grupos de seguridad de la red pueden asociarse a subredes de forma que restrinjan o permitan el acceso específico a puertos y direcciones IP. Los componentes de aplicaciones con requisitos de accesibilidad y seguridad separadas deben ubicarse dentro de subredes separadas.

- **Intervalos IP para redes y subredes:** cada subred tiene un intervalo IP. El intervalo IP no debe tener unas dimensiones que deriven en su infrautilización mientras que otras subredes estén asfixiándose por una falta de direcciones IP. Debe disponer de suficientes direcciones IP para satisfacer las necesidades actuales y las futuras. Esto debe tenerse en cuenta después de la comprensión de las necesidades de direcciones IP futuras de la implementación.
 - La planificación debe girar en torno al direccionamiento y los intervalos IP para los centros de datos on-premises, las subredes y las redes de Azure. No deben producirse solapamientos en aras de garantizar una conectividad y una accesibilidad perfectas.
- **Supervisión:** la supervisión es un aspecto importante de la arquitectura, así como un elemento fundamental que debe incluirse en la arquitectura de implementación general. **Azure Network Watcher** ofrece funcionalidades de registro y diagnóstico con información sobre el estado y el rendimiento de la red. Permite ver los paquetes de entrada o salida hacia y desde las máquinas virtuales, si han sido permitidos o rechazados, el siguiente salto del flujo para la validación de la configuración de las rutas definidas por el usuario, una auditoría de la seguridad de red y la captura de paquetes junto con límites de suscripción. También proporciona registros de diagnóstico para todos los recursos de red en un grupo de recursos.
El rendimiento de la red puede supervisarse a través de Log Analytics. La solución de administración de supervisión del rendimiento de la red proporciona la capacidad de supervisión de la red. Supervisa el estado, la disponibilidad y la accesibilidad de las redes. También se utiliza para supervisar la conectividad entre cloud público y centros de datos on-premises y subredes que alojan varios niveles de una aplicación de varios niveles.
- **Consideraciones sobre seguridad:** las redes virtuales son uno de los primeros componentes a los que tiene acceso cualquier recurso en Azure. La seguridad desempeña un papel importante a la hora de permitir o denegar el acceso al recurso. Los **grupos de seguridad de red (NSG)** son el principal medio de habilitación de seguridad de las redes virtuales. Pueden integrarse en subredes de redes virtuales y cada flujo de entrada y salida se puede restringir, filtrar y permitir. El **enrutamiento definido por el usuario (UDR)** y el reenvío IP también ayudan en el filtrado y enrutamiento de las solicitudes a los recursos de Azure.



Puedes obtener más información sobre el UDR y la tunelación forzada en <https://docs.microsoft.com/azure/virtual-network/virtual-networks-udr-overview>.

Los recursos también pueden estar asegurados y protegidos mediante la implementación de dispositivos de red como Barracuda, F5 y otros componentes de terceros.



Puedes obtener más información sobre estos dispositivos en
<https://azure.microsoft.com/solutions/network-appliances/>.



- **Implementación:** las redes virtuales deben implementarse en sus propios grupos de recursos dedicados. Los administradores de red deben gozar de permisos de propietario en este grupo de recursos, mientras que los desarrolladores o los miembros del equipo deben tener permisos de colaborador para que se les permita crear otro recurso de Azure en otros grupos de recursos que consumen servicios de la red virtual.

También se recomienda implementar recursos con direcciones IP estáticas en una subred dedicada, mientras que los recursos relacionados con direcciones IP dinámicas pueden ubicarse en otra subred.

Las políticas deben crearse para que solo los administradores de red puedan eliminar la red virtual, y también deben etiquetarse para facilitar la facturación.

- **Conectividad:** los recursos de una región de una red virtual pueden comunicarse perfectamente. Incluso los recursos en otras subredes dentro de una misma red virtual pueden interactuar entre sí sin necesidad de una configuración específica. Los recursos en varias regiones no pueden utilizar la misma red virtual. La red virtual no pueden sobrepasar los límites de una región. Para que los recursos estén disponibles a través de varias regiones, deben comunicarse a través de cada puerta de enlace dedicada, necesaria en cada extremo para facilitar la conversación. Las redes en cada extremo están conectadas a otras redes a través de estas puertas de enlace.

Sin embargo, a veces los recursos de Azure también tienen que conectarse con centros de datos on-premises. Las redes virtuales de Azules pueden conectarse con centros de datos on-premises con ayuda de la tecnología VPN y ExpressRoute. De hecho, una red virtual puede conectarse a varios centros de datos on-premises y otras regiones de Azure a la vez. Es recomendable que cada una de estas conexiones se encuentre en sus subredes dedicadas dentro de una red virtual.

Beneficios de las redes virtuales

Las redes virtuales son un elemento fundamental en la implementación de cualquier solución de IaaS significativa. Las máquinas virtuales no se pueden aprovisionar sin redes virtuales. Además de ser un componente prácticamente obligatorio en las soluciones, ofrece grandes ventajas en la arquitectura, entre las que destacan las siguientes:

- **Aislamiento:** la mayoría de los componentes de las aplicaciones presenta diferentes requisitos de seguridad y ancho de banda y tiene una administración del ciclo de vida diferente. Las redes virtuales ayudan en la creación de bolsas aisladas para que estos componentes puedan administrarse de manera independiente de otros componentes con la ayuda de las redes virtuales y las subredes.
- **Seguridad:** el filtrado y el seguimiento de los usuarios que acceden a los recursos son dos características importantes que ofrecen las redes virtuales. Pueden detener el acceso a un puerto y dirección IP maliciosos.
- **Extensibilidad:** las redes virtuales actúan como una red LAN privada en el cloud. También pueden ampliarse hasta convertirse en una WAM mediante la conexión con otras redes virtuales distribuidas por todo el mundo, además de ampliarse como una extensión a los centros de datos on-premises.

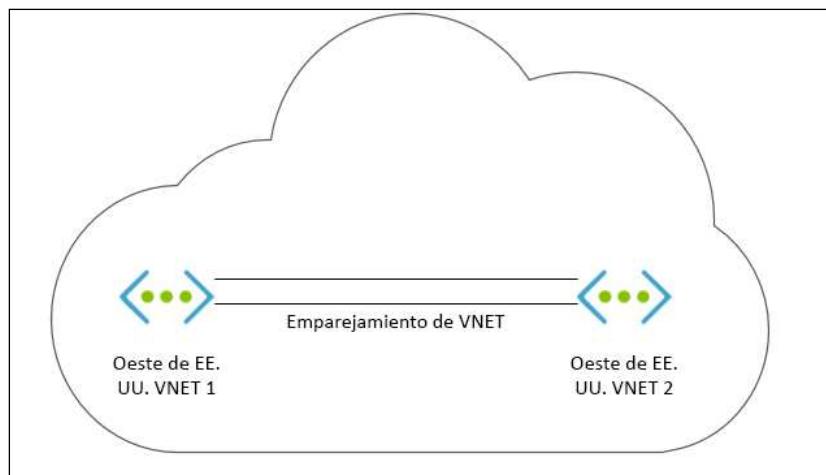
Diseño de red virtual

En esta sección, tendremos en cuenta algunos diseños y usos populares de las redes virtuales.

Puede haber varios usos de las redes virtuales. Una puerta de enlace puede implementarse en cada punto de conexión de la red virtual para habilitar la seguridad y transmitir paquetes con integridad y confidencialidad. Una puerta de enlace es un elemento fundamental en la conexión con redes on-premises; no obstante, es un componente opcional al emparejar las redes virtuales.

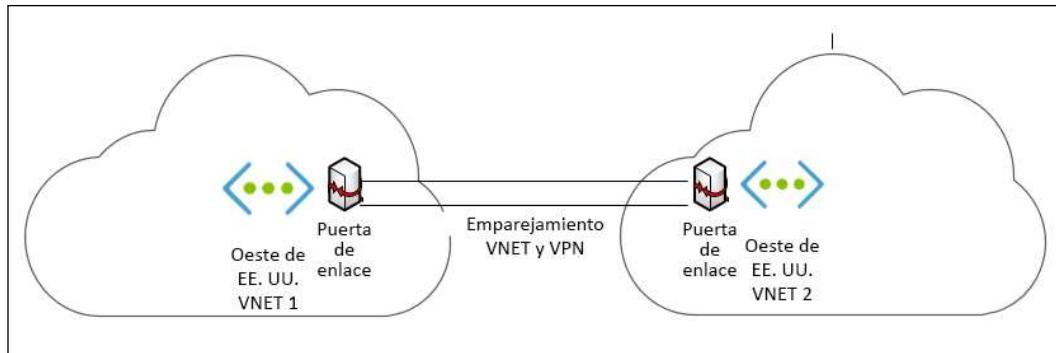
Conexión con los recursos dentro de la misma región y suscripción

Pueden conectarse entre sí varias redes virtuales dentro de la misma región y suscripción. Con ayuda del emparejamiento de las redes virtuales, ambas redes pueden conectarse y utilizar la red troncal de la red privada de Azure para la transmisión de paquetes entre sí. Las máquinas virtuales y los servicios en estas redes pueden comunicarse entre sí, aunque están sujetos a las limitaciones del tráfico de la red:



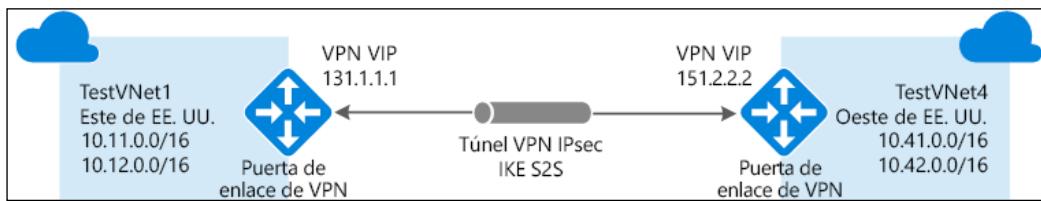
Conexión con los recursos dentro de la misma región y en otra suscripción

Si bien este uso es muy parecido al anterior, hay que añadir la complejidad de que otra red se encuentra en otra suscripción. Si ambas suscripciones pertenecen al mismo inquilino, pueden conectarse usando el emparejamiento de redes virtuales como se ha indicado anteriormente. En este caso, también se usaría la red troncal de la red privada de Azure para la transmisión de paquetes entre sí:



Conexión con los recursos en diferentes regiones y en otra suscripción

En este caso, el tráfico pasará a través de la red pública y el emparejamiento de redes virtuales debe implementarse con puertas de enlace para cifrar el tráfico con VPN:

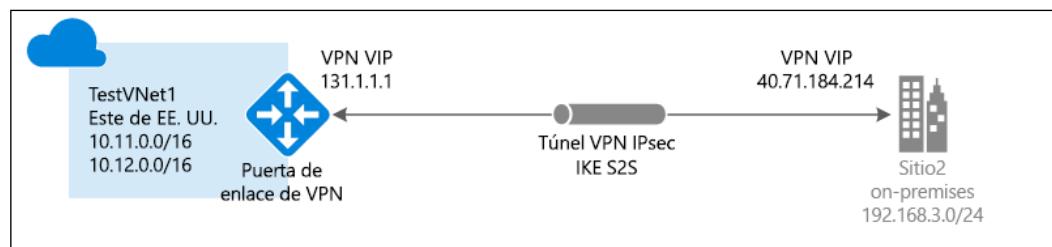


Conexión con centros de datos on-premises

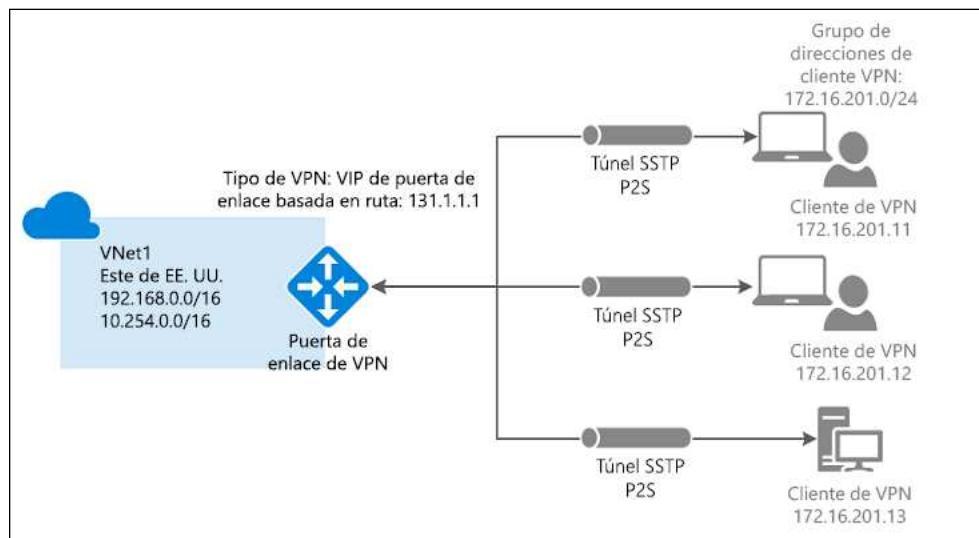
Las redes virtuales pueden conectarse con centros de datos on-premises, de manera que un centro de datos on-premise y Azure se conviertan en una única **red de área extensa (WAN)**. La conexión con la red on-premise necesita la implementación de puertas de enlace y de redes privadas virtuales en ambos extremos de la red.

Hay tres tecnologías diferentes disponibles para este propósito:

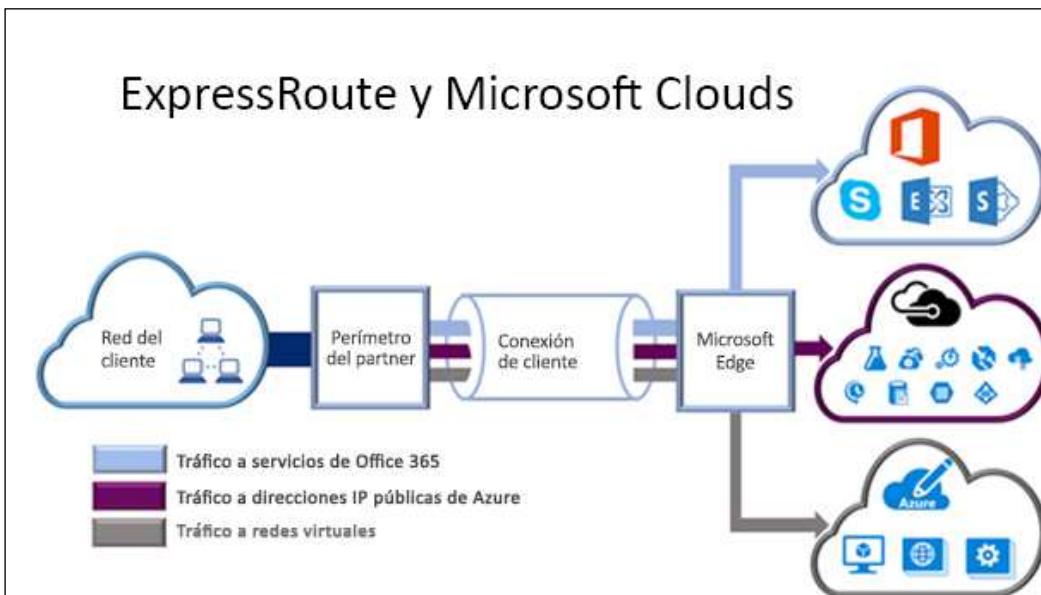
- **VPN de sitio a sitio:** esta tecnología debe usarse cuando la red de Azure y el centro de datos on-premise deban conectarse para formar una WAN en la que cualquier recurso en ambas redes pueda acceder a cualquier otro recurso independientemente de si se trata de un centro de datos on-premise o una red de Azure. Las puertas de enlace de VPN deben estar disponibles en ambos extremos de las redes por motivos de seguridad. Además, se deben implementar puertas de enlace de Azure en las subredes de las redes virtuales para conectarse con los centros de datos on-premises. Las direcciones IP públicas deben asignarse a puertas de enlace de centros on-premises para que Azure pueda conectarse a través de la red pública.



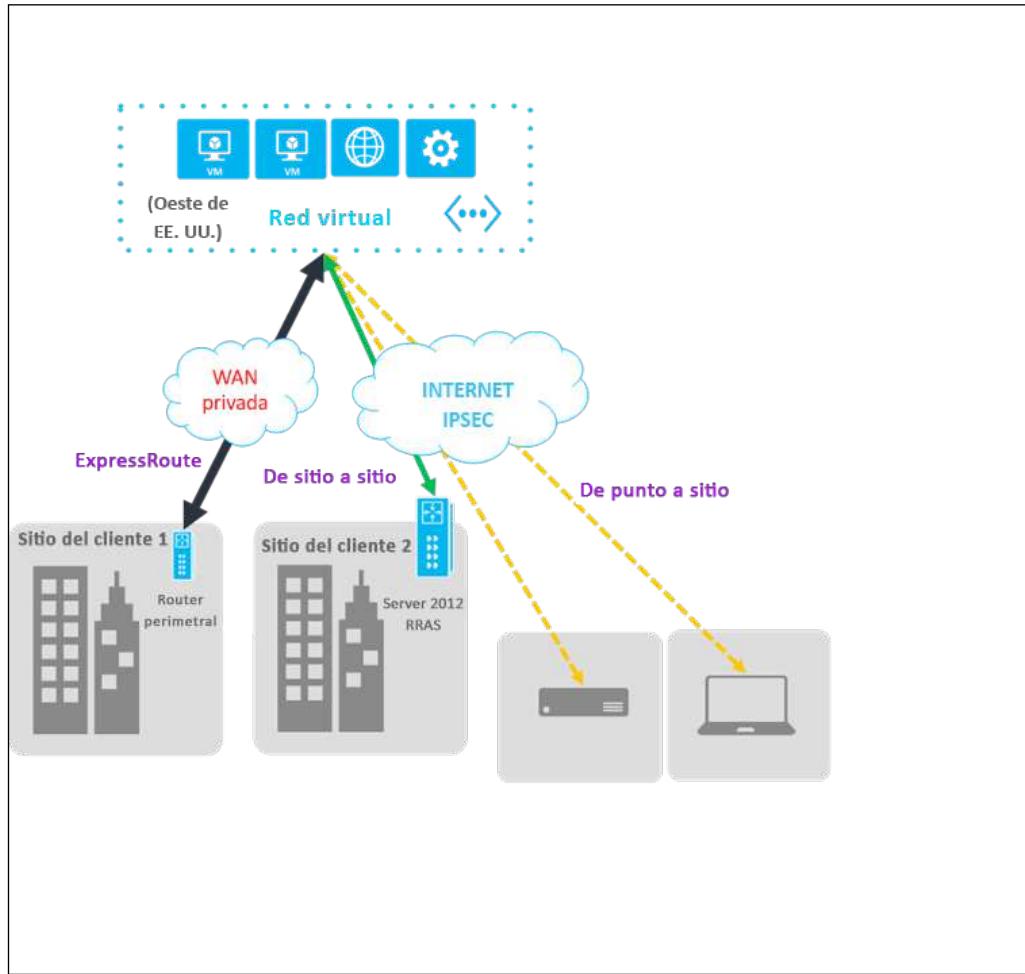
- **VPN de punto a sitio:** esta tecnología es similar a la conectividad VPN de sitio a sitio; sin embargo, hay un solo servidor u ordenador conectado al centro de datos on-premise. Se debe usar cuando haya algunos usuarios o clientes que deseen conectarse a Azure de forma segura desde ubicaciones remotas. Además, en este caso no es necesario utilizar direcciones IP públicas ni puertas de enlace en el centro de datos on-premise:



- **ExpressRoute:** tanto la tecnología VPN de sitio a sitio como la de punto a sitio funcionan a través de Internet público. Cifra el tráfico entre la red con la tecnología VPN y la tecnología de certificados. Sin embargo, hay aplicaciones que se implementan en modo híbrido. Algunos de sus recursos están alojados en la red de Azure y otros en el centro de datos on-premises. A pesar de que los recursos están alojados en Azure, estos recursos no deben usar Internet público para conectarse a los centros de datos on-premises. Azure ExpressRoute es la mejor solución para estos casos, aunque implica un precio elevado en comparación con la conexión tipo VPN de sitio a sitio y de punto a sitio. Se trata de una conectividad altamente segura y fiable que ofrece una velocidad mucho mayor, además de una menor latencia en comparación con otras tecnologías VPN. Esto se debe a que el tráfico no usa nunca Internet público, sino más bien conexiones dedicadas con proveedores de servicios. Azure ExpressRoute te ayuda a ampliar las redes on-premises e integrarlas en Azure a través de una conexión privada dedicada facilitada por un proveedor de conectividad:



La siguiente figura muestra los tres tipos de redes híbridas:



Por motivos de seguridad y aislamiento, se recomienda que las redes virtuales tengan subredes independientes para cada componente lógico con implementaciones separadas.

Almacenamiento

Azure proporciona soluciones de almacenamiento duraderas, de alta disponibilidad y escalables a través de sus servicios de almacenamiento.

El almacenamiento se utiliza para almacenar datos a largo plazo. Azure Storage está disponible a través de Internet en casi todos los lenguajes de programación.

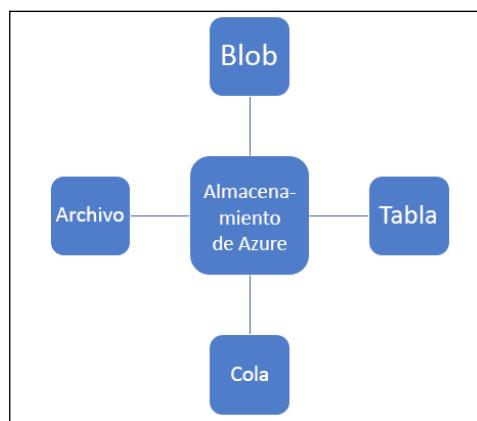
Categorías de almacenamiento

La solución de almacenamiento se compone de dos categorías de cuentas de almacenamiento:

- Un nivel de rendimiento de almacenamiento estándar que permite almacenar tablas, colas, archivos, blobs y discos de máquinas virtuales de Azure.
- Un nivel de rendimiento de almacenamiento premium con capacidad para almacenar discos de máquinas virtuales de Azure en el momento de su escritura. El tipo de almacenamiento premium ofrece un mayor rendimiento y operaciones de entrada/salida por segundo (IOPS) en comparación con el almacenamiento general estándar. El nivel premium está disponible actualmente en forma de discos de datos para máquinas virtuales que cuentan con SSD.

Tipo de almacenamiento

Azure proporciona cuatro tipos de servicios de almacenamiento general tal y como se muestra en el siguiente diagrama:



- **Almacenamiento de blobs:** este tipo de almacenamiento es el más adecuado para datos no estructurados como documentos, imágenes y otros tipos de archivos. El almacenamiento de los blobs puede ser en caliente o frío. El almacenamiento en caliente significa que al contenido del almacenamiento del blob se accede con frecuencia en comparación con el almacenamiento en frío.
- **Almacenamiento de tablas:** se trata de un almacén de datos de atributos clave NoSQL. Debe utilizarse para datos estructurados. Los datos se almacenan en forma de entidades.
- **Almacenamiento de colas:** ofrece un almacenamiento de mensajes fiable.
- **Almacenamiento de archivos:** se trata de un almacenamiento compartido basado en el protocolo SMB. Se utiliza normalmente para almacenar y compartir archivos. Con este tipo también se almacenan datos no estructurados, pero la principal diferencia estriba en que puede compartirse en el protocolo SMB.

Estos cuatro tipos de almacenamiento satisfacen diferentes necesidades de diseño y cubren casi todos los tipos de instalaciones de almacenamiento de datos.

Características de almacenamiento

La solución Azure Storage es flexible. Es decir, puedes almacenar desde tan solo unos pocos megabytes hasta petabytes de datos. No es necesario bloquear previamente la capacidad, ya que la solución aumentará y se reducirá automáticamente. Los consumidores solo tienen que pagar por el uso real de almacenamiento.

Azure Storage es una solución segura. Solo se puede acceder a ella con el protocolo SSL. Además, dicho acceso debe autenticarse previamente. Azure Storage permite generar un token **Secure Access Signature (SAS)** de nivel de cuenta que puede ser utilizado por los clientes de almacenamiento para su autenticación. También es posible generar tokens SAS de nivel de servicio individuales para blobs, colas, tablas y archivos. Los datos almacenados en Azure Storage pueden cifrarse. Esto se conoce como asegurar datos en reposo. El cifrado de disco de Azure se utiliza para cifrar el SO y los datos de los discos máquinas virtuales de IaaS. El **cifrado en el cliente (CSE)** y el **cifrado del servicio Storage (SSE)** se utilizan para cifrar los datos en Azure Storage. SSE es un ajuste de almacenamiento que abarca toda la cuenta y que garantiza que los datos se cifran al escribir los datos en el almacenamiento y que se descifran cuando el motor de almacenamiento los lee. Por consiguiente, no es necesario realizar ningún cambio en la aplicación para activar los SSE. En el modo CSE, las aplicaciones cliente pueden utilizar Storage SDK para cifrar los datos antes de enviar y escribir dichos datos en Azure Storage. Más adelante, la aplicación cliente puede descifrarlos durante la lectura. Esto proporciona seguridad tanto a los datos en tránsito y como a los datos en reposo. CSE depende de los secretos de Azure Key Vault. Se trata de otro servicio de cifrado de disco de Azure que se utiliza para cifrar el SO y los datos de los discos máquinas virtuales de IaaS.

Azure Storage ofrece una alta disponibilidad, así como durabilidad. Esto significa que Azure siempre conserva varias copias de las cuentas de Azure. La ubicación y el número de copias dependen de la configuración de replicación. Azure ofrece cuatro configuraciones de replicación. Estas configuraciones afectan tanto a los costes como a la disponibilidad en caso de desastres. El almacenamiento con redundancia local es la opción más económica y proporciona el menor nivel de disponibilidad en comparación con otras soluciones, mientras que el almacenamiento con redundancia geográfica con acceso de lectura es la opción más costosa y ofrece una alta disponibilidad.

- **Almacenamiento con redundancia local:** el **almacenamiento con redundancia local (LRS)** replica y conserva tres copias del almacenamiento dentro del mismo centro de datos de una región. Significa que los datos ofrecen una alta disponibilidad, pero solo dentro de un centro de datos. El almacenamiento se perderá o no estará disponible si este centro de datos se desactiva por alguna razón. Sigue el patrón sincrónico de las operaciones de escritura, esto es, para que el procedimiento se considere correcto se debe escribir en todas las réplicas. Una solicitud de escritura solo se considera correcta después de que se escriba en las tres réplicas. No obstante, Azure asegura la tolerancia a errores en el nivel del soporte y el disco de almacenamiento. Para ello, ubica las tres réplicas en

diferentes dominios de actualización y errores. Estos dominios se describen con mayor detalle en el siguiente capítulo.

- **Almacenamiento con redundancia de zona:** el **almacenamiento con redundancia de zona (ZRS)** es una solución más costosa que el almacenamiento con redundancia local, ya que, junto con las tres copias en el mismo centro de datos, también almacena los datos en otro centro de datos, así como dentro de una región. Puesto que este proceso implica a varios centros de datos, las escrituras en los almacenamientos se producen de manera asincrónica con objeto de mantener el rendimiento y la latencia del SLA.
- **Almacenamiento con redundancia geográfica:** el **almacenamiento con redundancia geográfica (GRS)** ofrece una mayor durabilidad y disponibilidad en comparación con la redundancia de zona mediante la replicación del almacenamiento en otra región, además de la redundancia local. Si bien se trata de una de las opciones más costosa, ofrece capacidad de recuperación ante desastres en el nivel de la región.
- **Almacenamiento con redundancia geográfica con acceso de lectura:** el **almacenamiento con redundancia geográfica con acceso de lectura (RA-GRS)** es similar al almacenamiento con redundancia geográfica, pero además proporciona acceso de solo lectura a las réplicas.

Consideraciones de arquitectura para cuentas de almacenamiento

La cuenta de almacenamiento debe aprovisionarse dentro de la misma región que la de otros componentes de la aplicación. De esta forma, se contribuye a la utilización de la misma red troncal de la red del centro de datos sin incurrir en cargos por utilización de red.

Cada uno de los servicios de Azure Storage tiene objetivos de escalabilidad determinados en términos de capacidad (GB), velocidad de transacciones y ancho de banda. Una cuenta de almacenamiento general permite un almacenamiento de hasta 500 TB de datos. Si es necesario almacenar más de 500 TB de datos, se deben crear varias cuentas de almacenamiento o utilizar un tipo de almacenamiento premium. El rendimiento general del almacenamiento es, como máximo, de 20 000 IOPS o 60 MB de datos por segundo. Se limitará cualquier requisito de un nivel de IOPS o datos administrados por segundo superior. Si, en términos de rendimiento, esto no es suficiente para las aplicaciones, se deben utilizar bien un almacenamiento premium o bien varias cuentas de almacenamiento.

El tamaño de una máquina virtual determina el tamaño y la capacidad de los discos de datos disponibles. Si bien unas máquinas virtuales con un mayor tamaño ofrecen discos de datos con mayor capacidad para IOPS, la capacidad máxima seguirá estando limitada a 20 000 IOPS y 60 MB por segundo. Debe recordar que estos son los niveles máximos y, por tanto, deberá tenerse en cuenta el uso de niveles inferiores al finalizar la arquitectura de almacenamiento.

Las cuentas de almacenamiento de Azure deben estar habilitadas para incluir la función de autenticación con tokens SAS. No se debe permitir el acceso anónimo. Por otra parte, para el almacenamiento de blobs, deben crearse diferentes contenedores con tokens SAS separados y basados en los diferentes tipos y las diferentes categorías de clientes que acceden a dichos contenedores. Estos tokens SAS se deben regenerar periódicamente para garantizar que estas claves no puedan ser adivinadas ni traspasadas por nadie.

Normalmente, los blobs recuperados para las cuentas de almacenamiento de blobs deben almacenarse en caché y puede determinarse la obsolescencia de los datos en una caché mediante la comparación de la propiedad de última modificación del blob para volver a recuperar el último blob.

La cuenta de almacenamiento de Azure ofrece características de simultaneidad para asegurar que el mismo archivo y los mismos datos no son modificados simultáneamente por diferentes usuarios. Esta solución ofrece las siguientes prestaciones:

- **Simultaneidad optimista:** permite que varios usuarios modifiquen los datos a la vez, pero realiza comprobaciones de escritura si el archivo o los datos se han modificado. En tal caso, se informa a los usuarios que deben volver a recuperar los datos y que deben volver a actualizar. Se trata de la función de simultaneidad predeterminada para los servicios de tablas.
- **Simultaneidad pesimista:** cuando una aplicación intenta actualizar un archivo, emite un bloqueo que niega explícitamente cualquier actualización por parte de otros usuarios. Se trata de la función de simultaneidad predeterminada para servicios de archivos cuando se accede a estos con el protocolo SMB.
- **“El último en escribir gana”:** en esta opción las actualizaciones no están limitadas y el último usuario es el que actualiza el archivo independientemente de lo que se haya leído al principio. Se trata de la función de simultaneidad predeterminada para servicios de colas, blobs y archivos (cuando se accede a estos con REST).

Patrones de diseño

Los patrones de diseño son soluciones probadas para abordar problemas de diseño conocidos. Son soluciones reutilizables que se pueden aplicar a diversos problemas. No son diseños ni códigos reutilizables que pueden incorporarse como parte de una solución. Consisten en guías de orientación y descripciones documentadas para resolver un problema. El problema puede manifestarse en un contexto diferente y los patrones de diseño pueden ayudar a resolverlos. Azure brinda numerosos servicios y cada uno ofrece una característica y capacidad específicas. La utilización de estos servicios es sencilla, pero la creación de las soluciones que conforman estos múltiples servicios puede resultar todo un desafío. Además, lograr ofrecer prestaciones como una alta disponibilidad, una elevada escalabilidad, fiabilidad, rendimiento y seguridad para una solución no es una tarea de poca importancia. Los patrones de diseño de Azure ofrecen soluciones inmediatas para estos problemas, al tiempo que proporcionan soluciones que se pueden adaptar para resoluciones individuales. Ayudan a ofrecer soluciones altamente disponibles, escalables, fiables, seguras y centradas en

el rendimiento en Azure. Aunque hay muchos patrones y algunos de ellos se describen con mayor detalle en capítulos posteriores, en este capítulo abordaremos algunos de los patrones de mensajería, rendimiento y escalabilidad. Incluiremos, además, enlaces a una descripción más detallada de dichos patrones. Estos patrones de diseño merecen un libro completo por sí mismos. Se han mencionado en este capítulo para que lectores conozcan su existencia y se proporcionan enlaces para obtener información más detallada.

Patrones de mensajería

Los patrones de mensajería ayudan a conectar servicios de forma flexible. Esto quiere decir que los servicios nunca se comunican directamente entre sí. Por el contrario, un servicio genera y envía un mensaje a un agente (generalmente una cola) y cualquier otro servicio que esté interesado en dicho mensaje puede recogerlo y procesarlo. No se establece ninguna comunicación directa entre el servicio emisor y el receptor. Este desacoplamiento no solo hace que los servicios y la aplicación general sean más fiables, sino que también los convierte en una solución más sólida y tolerante a errores. Los receptores pueden recibir y leer mensajes a su propia velocidad.

La mensajería ayuda en la creación de patrones asincrónicos. Esta implica enviar mensajes de una entidad a otra. Estos mensajes son creados y enviados por un remitente, almacenados en un almacenamiento duradero y, finalmente, son consumidos por los destinatarios.

Los principales aspectos arquitectónicos que aborda la mensajería son los siguientes:

- **Durabilidad:** los mensajes se almacenan en un almacenamiento duradero y la aplicación puede leerlos más adelante después de aparezcan en caso de fallo previo.
- **Fiabilidad:** los mensajes ayudan en la introducción de fiabilidad a través del diseño, ya que estos no se pierden al permanecer en el disco.
- **Disponibilidad de los mensajes:** los mensajes están disponibles para su consumo por parte de las aplicaciones después de la restauración de la conectividad y de un periodo de inactividad previo.

Azure proporciona temas y colas del bus de servicio para implementar patrones de mensajería dentro de las aplicaciones. La cola de almacenamiento también puede utilizarse para el mismo propósito.

Al elegir entre la cola de Azure Service Bus y la cola de almacenamiento se determina la duración de almacenamiento del mensaje, el tamaño del mensaje, la latencia y el coste.

Azure Service Bus permite unos tamaños de mensaje de 256 KB, mientras que la cola del almacenamiento permite unos tamaños de 64 KB. Azure Service Bus puede almacenar mensajes durante un periodo de tiempo ilimitado y la cola de almacenamiento puede almacenar mensajes durante siete días.

El coste y la latencia son más altos en el caso de las colas de Azure Bus Service.

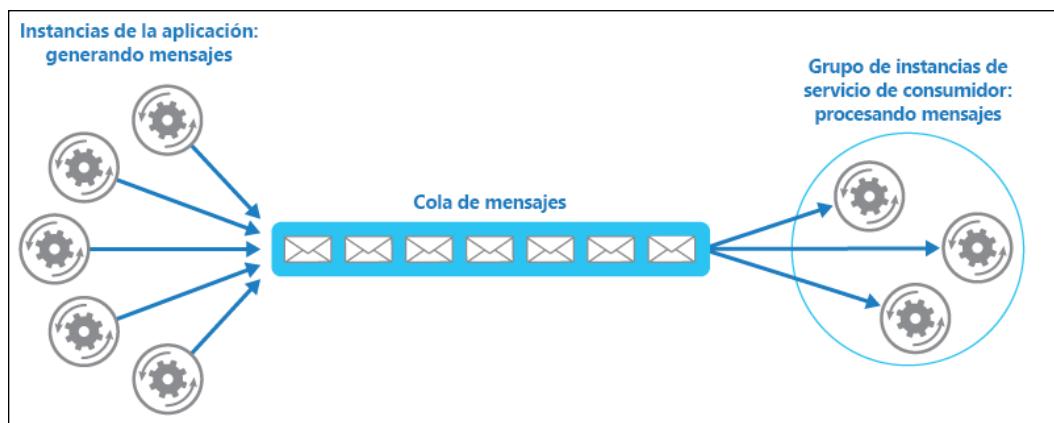
En función de las necesidades y los requisitos de las aplicaciones, deben considerarse los factores anteriores antes de decidir la cola más adecuada.

Consumidores en competencia

Un solo consumidor de mensajes trabaja de manera sincrónica, a menos que la propia aplicación implemente la lógica de lectura de mensajes de forma asíncrona. El patrón de los consumidores en competencia ayuda a implementar una solución cuando varios consumidores están listos para procesar el mensaje entrante y compiten por ello. Esto ayuda a diseñar soluciones altamente disponibles y escalables. Te ofrece escalabilidad porque, con varios consumidores, se puede procesar un mayor número de mensajes en un menor periodo de tiempo. Y ofrece una alta disponibilidad porque puede haber uno o más consumidores para procesar los mensajes, aun cuando se bloqueen algunos de ellos.

Este patrón debe usarse cuando los mensajes sean independientes unos de otros. Los mensajes por sí mismos contienen la información completa para que consumidor pueda llevar a cabo una tarea. Este patrón no debe utilizarse si no hay ninguna dependencia entre mensajes. Los consumidores deben ser capaces de realizar las tareas de manera aislada. Además, este patrón puede aplicarse en casos de servicios con demanda variable. Según la demanda, pueden añadirse o quitarse consumidores adicionales.

Es necesaria una cola de mensajes para la implementación del patrón de consumidores en competencia. En este patrón, los patrones procedentes de varias fuentes pasan por una única cola que está conectada a varios consumidores en el otro extremo. Estos consumidores borran el mensaje después de leerlo para que no se vuelva a procesar.



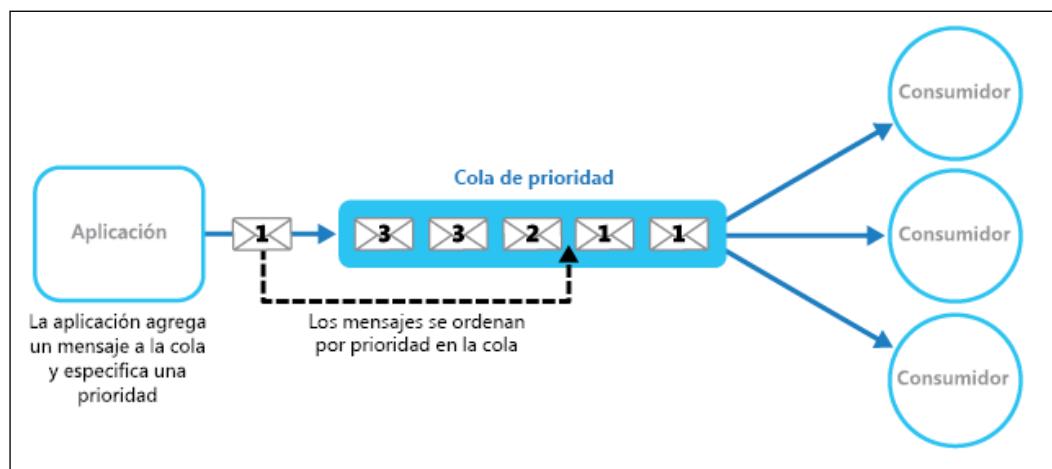
Se puede obtener más información sobre este patrón en
<https://docs.microsoft.com/azure/architecture/patterns/competing-consumers>.

Cola de prioridad

A menudo, durante el procesamiento, es necesario otorgar prioridad a los mensajes con mayor importancia en comparación con mensajes generales de menor prioridad. Este patrón es importante para aplicaciones que ofrecen diferentes **acuerdos de nivel de servicio (SLA)** a los consumidores, los cuales prestan servicios basados en diferentes planes y suscripciones.

Las colas siguen el patrón “primero en entrar, primero en salir”. Los mensajes se procesan de manera secuencial. Sin embargo, con la ayuda de este patrón, es posible el procesamiento acelerado de algunos mensajes debido a su mayor prioridad. Existen varias maneras de implementar este patrón. Si la cola proporciona la capacidad de asignar prioridad y reorganizar mensajes según su prioridad, puede que una única cola sea suficiente para implementar este patrón.

No obstante, si la cola no tiene la capacidad de reorganizar mensajes, puede crear colas separadas para las diferentes prioridades y cada cola puede tener asociados consumidores separados:



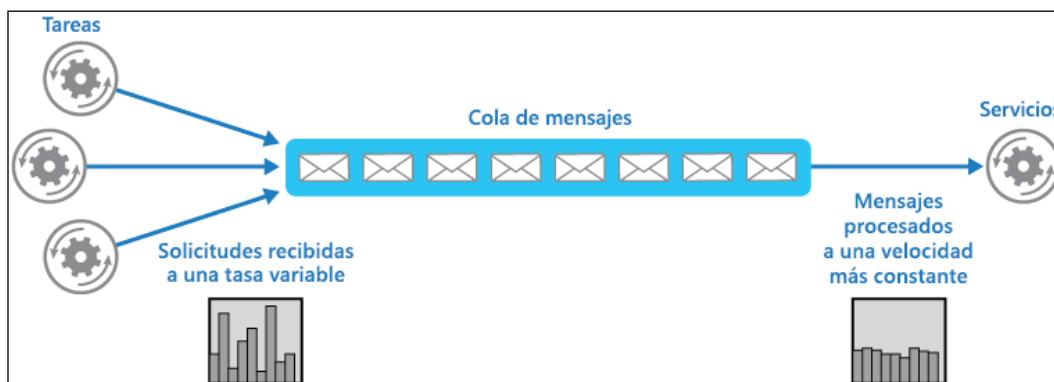
De hecho, si es necesario, este patrón puede volver a utilizar patrones de consumidores en competencia para un procesamiento acelerado de mensajes de cada cola que



Se puede obtener más información sobre este patrón en
<https://docs.microsoft.com/azure/architecture/patterns/priority-queue>.

Patrón de nivelación de carga basada en cola

En ocasiones la carga en una aplicación no se puede determinar en todo momento. Aunque la demanda de una aplicación suele ser coherente y predecible, hay veces en que esta carga puede aumentar de manera considerable y derivar en un fallo del servicio, en la prestación de un menor rendimiento o en la falta de disponibilidad. El patrón de nivelación de carga basado en cola puede ayudar en estas situaciones. En este patrón, se mantiene una cola y todas las solicitudes del servicio se almacenan como mensajes en esta cola. La cola actúa como un almacenamiento temporal altamente disponible y duradero que envía mensajes al servicio a una velocidad controlada, con la consiguiente reducción de interrupciones del servicio. En la siguiente figura se muestra un ejemplo de este patrón. Hay varias tareas que envían mensajes a la cola de mensajes. La cola almacena los mensajes y garantiza que el servicio recibe estos mensajes a una velocidad coherente con los recursos disponibles en dicho servicio:



Este patrón asegura que no se consumen todos los recursos ni se producen escalados innecesarios al tener que aprovisionar más instancias para satisfacer una mayor demanda del servicio. Tiene un efecto directo sobre los costes, además de ofrecer previsión en términos de utilización e instancias de recursos.

Una mayor escalabilidad y una alta disponibilidad son otras ventajas derivadas de la implementación de este patrón.

Patrones de rendimiento y escalabilidad

Los conceptos de rendimiento y escalabilidad van de la mano. El rendimiento es un indicativo de la capacidad de respuesta de un sistema para ejecutar cualquier acción en un intervalo de tiempo determinado; mientras que la escalabilidad es la capacidad de un sistema para afrontar los incrementos de carga sin que se vea afectado el rendimiento. En esta sección, se describen un par de patrones de diseño relacionados con el rendimiento y la escalabilidad.

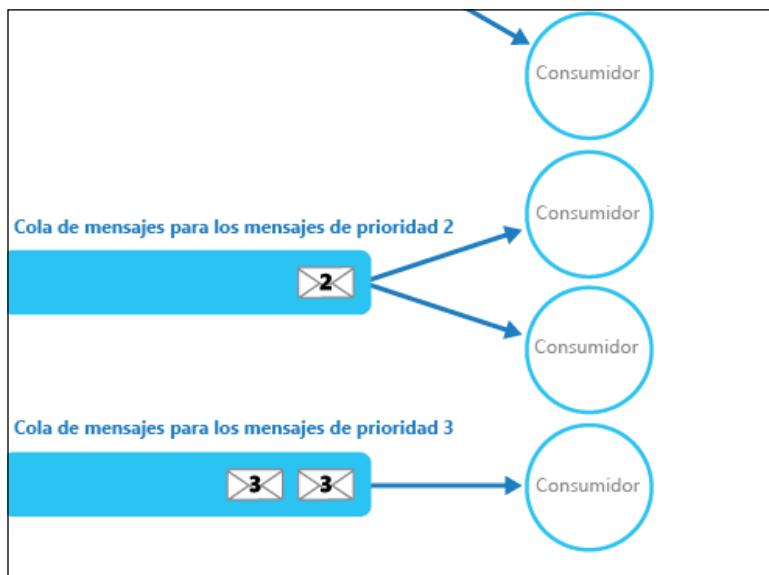
Patrón CQRS (Command and Query Responsibility Segregation)

CQRS es un patrón genérico que puede aplicarse en cualquier situación que comprenda unos datos almacenados en un almacén de datos y a los que debe accederse de manera que aumente la capacidad de respuesta y el rendimiento general de la aplicación.

Las operaciones de los datos pueden clasificarse a grandes rasgos en operaciones de lectura y de escritura. Hay múltiples maneras de leer y escribir en el almacén de datos y suele haber una capa de acceso a los datos y un componente encargado de realizar estas operaciones. Este componente de acceso a los datos contiene información acerca de la conexión con el almacén de datos y lleva a cabo operaciones tanto de lectura como de escritura. Realizar ambas operaciones dentro una única interfaz puede resultar todo un desafío en términos de rendimiento, sobre todo si los datos son de gran tamaño y la relación de lectura y escritura está sesgada.

Command and Query Responsibility Segregation (CQRS) es un patrón que ayuda en la implementación de operaciones de lectura y escritura utilizando para ello diferentes interfaces. Significa que los componentes que implementan las operaciones de lectura están separadas de las operaciones de escritura y que, por tanto, pueden utilizarse individualmente en una instancia separada. Esto ayuda en el aprovisionamiento de capacidad de recursos dedicados a dichas operaciones. También ayuda en situaciones en las que el tiempo de ejecución para las operaciones de lectura y escritura es significativamente amplio e implica un mayor consumo de recursos.

El patrón CQRS no solo ayuda a mejorar el rendimiento de la aplicación, sino que también ayuda en el diseño y la implementación entre varios equipos. Dada su naturaleza de utilización de modelos separados, el patrón CQRS no es una solución adecuada si utiliza herramientas de generación:



Se puede obtener más información sobre este patrón en
<https://docs.microsoft.com/azure/architecture/patterns/cqrs>.



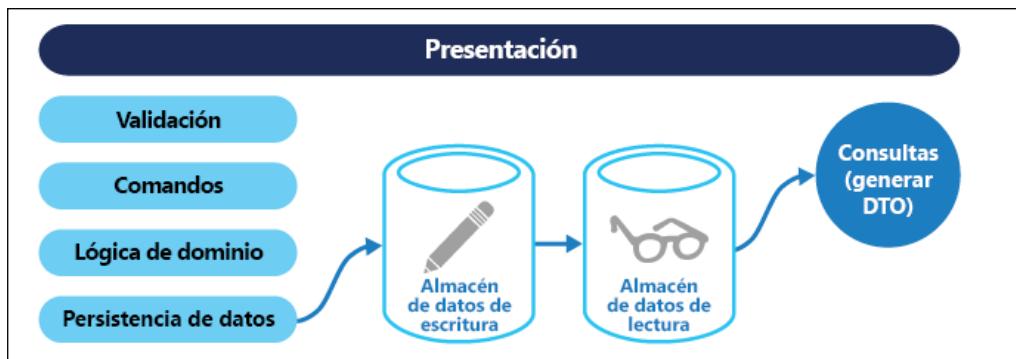
Patrón de limitación

A veces, hay aplicaciones que tienen requisitos de SLA muy estrictos en cuanto a rendimiento y escalabilidad, independientemente del número de usuarios que consuman el servicio. En tales circunstancias, es importante implementar patrones de limitación porque ayudan a limitar el número de solicitudes que se permite ejecutar. La carga en las aplicaciones no se puede predecir con precisión en todas las circunstancias. Cuando la carga en la aplicación alcanza su nivel máximo, la limitación ayuda a reducir la presión sobre los servidores y los servicios mediante el control del consumo de recursos.

Este patrón debe utilizarse cuando el cumplimiento del SLA sea una prioridad para las aplicaciones y para evitar que algunos usuarios consuman más recursos de los que tienen asignados, además de para optimizar los picos de demanda y optimizar los costes en relación con el consumo de recursos. Estos son escenarios plausibles para aplicaciones diseñadas para su implementación en el cloud.

Puede haber múltiples estrategias que se utilicen para la limitación de una aplicación. La estrategia de limitación puede rechazar nuevas solicitudes una vez que se supere un umbral o bien puede comunicar al usuario que su solicitud está en cola y que tendrá la oportunidad de ejecutarse cuando se reduzca el número de solicitudes.

El siguiente diagrama ilustra la implementación del patrón de limitación en un sistema multiinquilino donde a cada inquilino se le asigna un límite de utilización de recursos fijo. Cuando se exceda este límite, se restringirá cualquier demanda adicional de recursos, con lo que se logrará mantener unos recursos suficiente:



[Se puede obtener más información sobre este patrón en <https://docs.microsoft.com/azure/architecture/patterns/throttling>.]

Otros patrones

En esta sección se describen otros patrones importantes.

Patrón de reintento

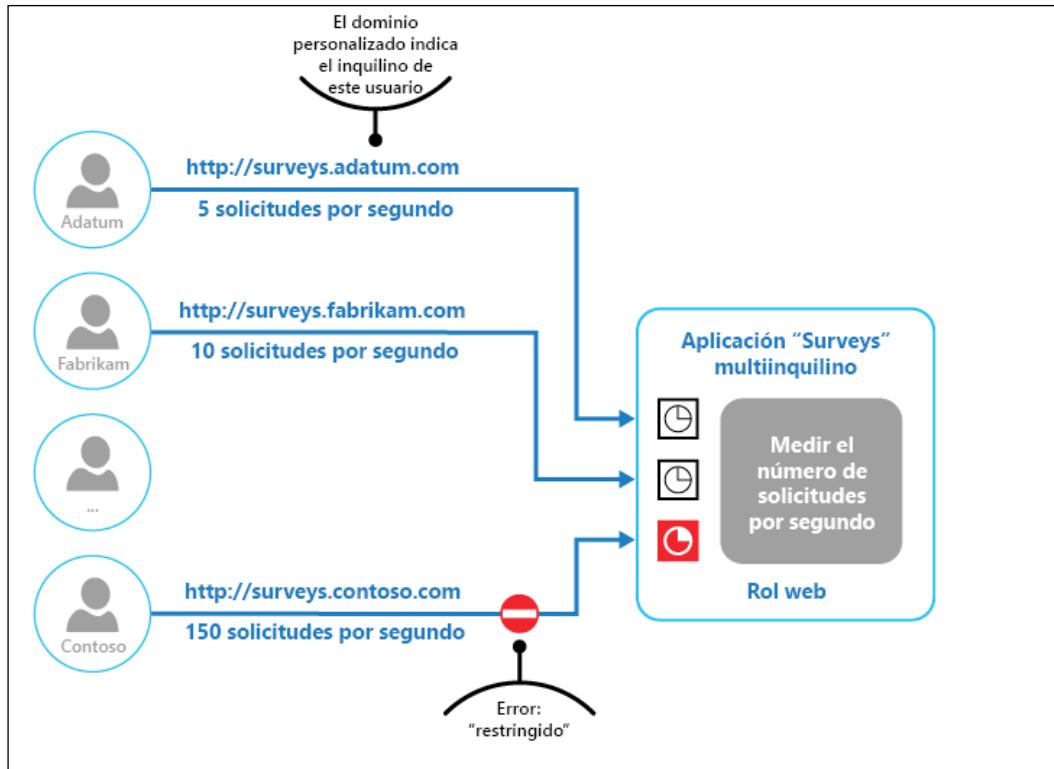
El patrón de reintento es un patrón extremadamente importante que permite que las aplicaciones y los servicios ofrezcan una mayor resistencia a fallos transitorios. Piensa en una situación en la que intentas conectar y utilizar el servicio, y el servicio no está disponible por cualquier motivo. Si el servicio va a restablecerse en un corto periodo de tiempo, tiene sentido seguir intentando lograr establecer una conexión. Esto hace que la aplicación sea más sólida y tolerante a errores, por lo que mejora la estabilidad. En Azure, la mayoría de los componentes se ejecuta en Internet y dicha conexión a Internet puede experimentar errores transitorios de manera intermitente. Dado que estos fallos se corrigen en cuestión de segundos, no se debería permitir que una aplicación quedase bloqueada. La aplicación debe diseñarse de manera que se pueda volver a intentar utilizar el servicio repetidamente durante estos fallos. También debe diseñarse para que deje de intentarlo si el servicio sigue sin estar disponible después de un determinado número de reintentos.

Este patrón debe implementarse cuando una aplicación pueda experimentar errores transitorios al interactuar con un servidor remoto o acceder a un recurso remoto. Se espera que estos errores sean de corta duración y que la repetición de una solicitud que previamente ha fallado pueda llevarse a cabo correctamente en un intento posterior.

El patrón de reintento puede adoptar diferentes estrategias de reintento en función de la naturaleza de los errores y de la aplicación:

- **Número fijo de intentos:** indica que la aplicación intentará comunicarse con el servicio un número concreto de veces antes de que pueda determinar un fallo y emitir una excepción. Por ejemplo, puede intentar conectarse a otro servicio tres veces. Si se establece una conexión correctamente en unos de estos tres intentos, se considerará que la operación se ha llevado a cabo correctamente; no obstante, después de que hayan transcurrido estos tres intentos, se emitirá una excepción.
- **Reintento basado en horario:** indica que la aplicación intentará comunicarse con el servicio repetidamente durante un número determinado de segundos o minutos y esperará a que haya transcurrido un número determinado de segundos o minutos antes de volver a intentarlo. Por ejemplo, puede volver a intentar conectarse a otro servicio cada tres segundos durante 60 segundos. Si se establece una conexión correctamente en unos de estos intentos, se considerará que la operación se ha llevado a cabo correctamente; no obstante, después de que hayan transcurrido estos 60 segundos, se emitirá una excepción.
- **Aplazamiento y retardo del reintento:** indica que la aplicación intentará comunicarse con el servicio repetidamente basándose en un periodo de tiempo y que seguirá añadiendo un retardo incremental en los intentos posteriores. Por ejemplo, lo intentará durante un total de 60 segundos periodo durante el cual el primer reintento se producirá después de un segundo; el segundo intento, dos segundos después del último reintento; el tercero, cuatro segundos después del último intento; y así sucesivamente. De esta forma, se logra reducir el número total de reintentos.

Un ejemplo de patrón de reintento puede ser una situación en la que la primera solicitud obtiene HTTP 500 como mensaje de respuesta, el segundo reintento vuelve a obtener HTTP 500 como respuesta, y, por último, la solicitud se realiza correctamente y obtiene HTTP 200:



Se puede obtener más información sobre este patrón en
<https://docs.microsoft.com/azure/architecture/patterns/retry>.

Patrón de interruptor de circuito

Se trata de un patrón muy útil. Piensa de nuevo en una situación en la que intentas conectar con un servicio y utilizarlo, y el servicio no está disponible por cualquier motivo. Si el servicio no va a restablecerse en breve, no tiene sentido seguir intentando lograr establecer una conexión. Por otra parte, mantener ocupados otros recursos mientras realiza reintentos supone un desperdicio de muchos recursos que potencialmente podrían estar siendo utilizados para otros fines.

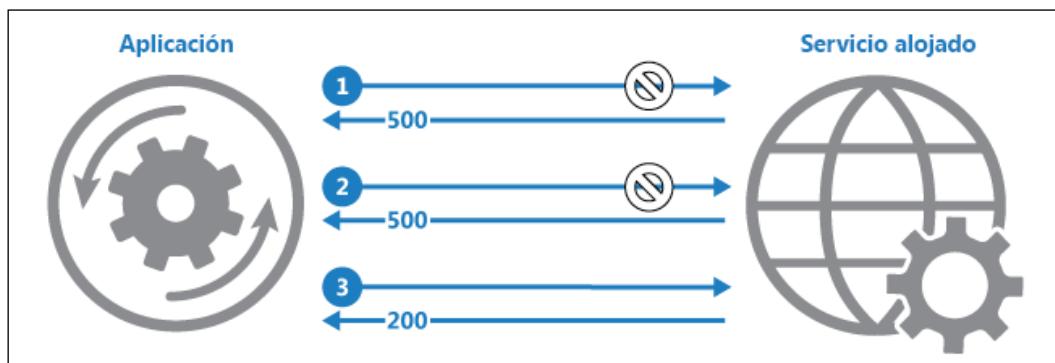
El modelo de interruptor de circuito ayuda a eliminar tal despilfarro de recursos. Se puede evitar que las aplicaciones intenten reiteradamente conectarse y utilizar un servicio que no está disponible. También ayuda a las aplicaciones a detectar si el servicio vuelve a estar activo y en funcionamiento y, finalmente, permite que las aplicaciones se conecten a él.

Para implementar el patrón de interruptor de circuito, todas las solicitudes al servicio deben pasar a través de otro servicio. Este servicio actúa como un proxy en relación con el servicio original. El objetivo de este servicio de proxy es mantener una máquina de estados y actuar como una puerta de enlace al servicio original. Hay tres estados que mantiene. Puede incorporar más estados que podrían incluirse según los requisitos de la aplicación.

Los estados mínimos necesarios para implementar este patrón son los siguientes:

- **Open:** indica que el servicio está inactivo y la aplicación muestra una excepción inmediatamente en lugar de permitir un reintento o esperar a que transcurra el tiempo de espera. Cuando el servicio vuelve a estar activo, el estado pasa a estado Half-Open.
- **Closed:** este estado indica que el servicio funciona correctamente y que la aplicación puede seguir adelante y conectarse a él. Por lo general, se mantiene un control del número de fallos antes de que pasar al estado Open.
- **Half-Open:** en algún momento, cuando el servicio esté activo y funcionando, este estado permitirá que un número limitado de solicitudes pasen a través de él. Este estado es una prueba decisiva, ya que comprueba si la solicitud que pasa a través de este se lleva a cabo correctamente o no. En tal caso, el estado pasa del estado Half-Open al estado Closed. Este estado también puede implementar un contador para permitir que solo se lleven a cabo un número determinado de solicitudes antes de que se pueda pasar al estado Closed.

En la siguiente figura se muestran los estados mencionados:





Se puede obtener más información sobre este patrón en
<https://docs.microsoft.com/azure/architecture/patterns/circuit-breaker>.



Resumen

Hay numerosos servicios disponibles en Azure y la mayoría puede combinarse para crear soluciones reales. Es difícil abordar todos los patrones de diseño en solo 20 páginas, ya que hay cientos de servicios y numerosas formas de conectarlos. En este capítulo se han explicado los tres servicios más importantes de Azure: las regiones, el almacenamiento y las redes. Estos conforman casi la totalidad de la base de todas las soluciones implementadas en todos los clouds. Este capítulo ha proporcionado detalles acerca de estos servicios y acerca de cómo su configuración y aprovisionamiento pueden afectar a las decisiones de diseño. Las consideraciones importantes a tener en cuenta en términos de almacenamiento y redes se han descrito en este capítulo. Tanto las redes como los almacenamientos ofrecen un gran número de opciones y, por tanto, es esencial elegir una configuración apropiada según los requisitos correspondientes. Finalmente, se han descrito algunos de los patrones de diseño más importantes relacionados con la mensajería, tales como los consumidores en competencia, las colas de prioridad y la nivelación de carga. Los patrones relacionados con CQRS y la limitación también se han explicado, junto con otros patrones como los patrones de reintento y de interruptor de circuito.

El próximo capítulo será un capítulo muy interesante ya que analizaremos las opciones de alta disponibilidad en Azure en relación con la implementación de las cargas de trabajo.

3

Diseño de alta disponibilidad

Ejecutar aplicaciones y sistemas que estén disponibles para su consumo por parte de los usuarios siempre que lo necesiten es una de las principales prioridades de la mayoría de los directores de sistemas de información. Estos quieren que sus aplicaciones sean operativas, funcionales y que sigan estando disponibles para sus consumidores, incluso cuando sucede algún evento desafortunado; este es precisamente el tema de este capítulo: la alta disponibilidad. Mantener las luces encendidas es la metáfora común utilizada para la alta disponibilidad. Lograr una alta disponibilidad para las aplicaciones no es una tarea sencilla y las organizaciones deben invertir una cantidad considerable de tiempo, energía, recursos y dinero para lograrlo. Incluso cuando se utilizan, sigue habiendo posibilidades y riesgos de que su aplicación no produzca los resultados deseados. Azure proporciona un montón de características de alta disponibilidad para **máquinas virtuales (VM)** y servicios de PaaS. En este capítulo, repasaremos las características arquitectónicas y de diseño proporcionadas por Azure para asegurar la alta disponibilidad para aplicaciones y servicios.

En este capítulo, abordaremos los siguientes temas:

- Conjuntos de disponibilidad de Azure:
 - Dominio de error
 - Actualizar dominio
- Equilibrador de carga de Azure
- Puertas de enlace de aplicaciones de Azure

Alta disponibilidad

La alta disponibilidad es una de las principales preocupaciones arquitectónicas para cualquier arquitecto. Constituye uno de los principales requerimientos técnicos no funcionales para cualquier servicio serio y su implementación. La alta disponibilidad hace referencia a la característica de un servicio o aplicación que lo mantiene operativo de manera continua, satisfaciendo o superando su **acuerdo de nivel de servicio (SLA)** definido prometido. A los usuarios se les promete cierto SLA sobre la disponibilidad de un servicio. El servicio debe estar disponible para el consumo basándose en su SLA. Por ejemplo, un SLA puede tener un 99 % de disponibilidad para una aplicación durante todo el año (suponiendo 365 días). Esto significa que debe estar disponible para el consumo por los usuarios durante 361,35 días. Si ofrece menos de esto, se produce una infracción del SLA. La mayoría de las aplicaciones críticas define su SLA de alta disponibilidad con cinco nueves durante un año. Significa que la aplicación debe estar activa, operativa y disponible durante todo el año y solo puede estar inactiva y no disponible durante 5,2 horas.

Es importante señalar aquí que la alta disponibilidad se define en términos de tiempo, es decir, anualmente, mensualmente, semanalmente y así sucesivamente, e incluso podría ser una combinación de estos.

Un servicio o aplicación está formado por múltiples componentes que se implementan en diferentes niveles y capas. Además, se implementa en un sistema operativo y se aloja en una máquina física o virtual. Consume servicios de red y almacenamiento para diversos propósitos. Podría ser incluso dependiente de sistemas externos. Para que estos servicios o aplicaciones estén altamente disponibles, es importante que las redes, el almacenamiento, el sistema operativo, las máquinas físicas o virtuales, cada componente de la aplicación esté creado y diseñado con el SLA y la alta disponibilidad en mente. Debe haber un proceso de ciclo de vida de la aplicación definido que debe utilizarse para asegurar que la alta disponibilidad se integra desde el principio de la planificación de la aplicación hasta su introducción en las operaciones. También implica el diseño para la redundancia. Deben incluirse recursos redundantes en la arquitectura de implementación y de la aplicación en general para asegurarse de que, si uno se inactiva, el otro lo sustituya para cubrir las necesidades del consumidor.

SLA

SLA se define en la Wikipedia como:

"Un acuerdo de nivel de servicio es un acuerdo entre dos o más partes, en el que uno es el consumidor y los demás son proveedores de servicios. Los aspectos particulares del servicio (calidad, disponibilidad, responsabilidad) se acuerdan entre el proveedor y el usuario del servicio. El componente más común del SLA es que los servicios se deben prestar al consumidor según lo acordado en el contrato".

Factores que afectan a la alta disponibilidad

El mantenimiento planificado, el mantenimiento no planificado y la arquitectura de implementación de la aplicación son los principales factores que afectan a la alta disponibilidad de una aplicación.

Mantenimiento planificado

El mantenimiento planificado se refiere al proceso de mantener la aplicación y el ecosistema circundante que consta de plataformas, estructuras, software, sistema operativo y controladores de host e invitado actualizados con las versiones estables más recientes. Es importante parchear el software, los controladores y los sistemas operativos con las últimas actualizaciones, ya que ello ayuda a mantener la salud del entorno desde una perspectiva de seguridad, rendimiento y preparación para el futuro. No actualizar un entorno no es una opción y es una realidad. Incluso las aplicaciones deben actualizarse con revisiones, errores y una funcionalidad mejorada. Todas las organizaciones planifican las actualizaciones de las aplicaciones y el entorno y, por lo general, estas implican apagar y reiniciar la aplicación y el sistema operativo. También podría conllevar el inicio del sistema operativo host físico que, a su vez, reiniciará todas las máquinas virtuales invitadas que se ejecuten sobre él.

Mantenimiento no planificado

Como su nombre indica, el mantenimiento no planificado se refiere al mantenimiento que no se puede planificar y es ad hoc por naturaleza. Hace referencia a fallos de hardware tales como corrupción de almacenamiento, error de red o del router, corte de alimentación y un montón de otros errores de hardware y software. Los errores en la plataforma subyacente que desconectan la aplicación también son parte del mantenimiento no planificado.

Arquitectura de implementación de aplicaciones

La arquitectura de aplicaciones desempeña un papel crucial para garantizar la alta disponibilidad de una aplicación. Una aplicación cuyos componentes se implementan en una sola máquina no tiene una alta disponibilidad. Cuando se reinicia la máquina, la aplicación no está disponible para sus usuarios. Del mismo modo, depender de una sola instancia de un recurso puede convertirse en un punto único de error desde el punto de vista de la alta disponibilidad. Cada componente de aplicación debe estar diseñado de manera que se puede implementar en varias máquinas y la redundancia no debe ser un cuello de botella. Algunos softwares proporcionan características relacionadas con la alta disponibilidad y no dependen de sistemas operativos de host u otras herramientas de terceros. Los grupos de disponibilidad de SQL server son un ejemplo de tales características.

Alta disponibilidad y escalabilidad

La alta disponibilidad es diferente de la escalabilidad, aunque ambas son preocupaciones arquitectónicas serias. La escalabilidad se refiere a la flexibilidad y elasticidad para agregar o reducir recursos a la implementación existente a fin de dar cabida a más usuarios de lo normal sin poner en peligro el rendimiento de la aplicación. La escalabilidad ayuda indirectamente a hacer una aplicación altamente disponible. Sin embargo, esto no significa que la escalabilidad conduzca finalmente a la alta disponibilidad. La alta disponibilidad es una preocupación arquitectónica que no depende del número de usuarios, mientras que las reglas de la escalabilidad están determinadas por un número de usuarios que consumen el servicio. La alta disponibilidad podría ser un requisito, incluso aunque haya muy pocos usuarios. La alta disponibilidad hace referencia a servicios presentes y operativos cómo y cuándo los usuarios demandan su consumo. Es una función de consumo basada en un SLA. La escalabilidad es el tema del próximo capítulo, donde se comentará con más detalle.

Alta disponibilidad y recuperación ante desastres

La alta disponibilidad es también diferente de la recuperación ante desastres; sin embargo, la diferencia puede ser muy sutil. La alta disponibilidad es una función de la aplicación lista para usar en el momento en el que el usuario lo solicite y de la manera que lo precise. Por lo tanto, está diseñada para las operaciones anteriores a un desastre, mientras que la recuperación ante desastres es una función que entra en escena después de un desastre. La recuperación ante desastres se refiere a la implementación de la arquitectura a través de la cual los servicios están activos y funcionando después de un desastre, mientras que la alta disponibilidad se encarga de la disponibilidad antes de un desastre. La recuperación ante desastres incluye la copia de seguridad de datos, servidores archivados y latentes en diferentes continentes, mientras que la alta disponibilidad consta de equilibradores de carga, distribución de carga, redundancia activa-pasiva y activa-activa.

Alta disponibilidad de Azure

Resulta complicado lograr una alta disponibilidad que cumpla con los elevados requisitos del SLA. Azure proporciona un montón de características que permiten la alta disponibilidad para aplicaciones, desde el sistema operativo host e invitado hasta las aplicaciones que utilizan su PaaS. Los arquitectos pueden utilizar estas características para obtener alta disponibilidad en sus aplicaciones mediante la configuración en lugar de crear estas características desde cero o depender de herramientas de terceros.

En esta sección, vamos a ver funciones y capacidades proporcionadas por Azure para hacer que las aplicaciones sean altamente disponibles. Antes de adentrarnos en los detalles arquitectónicos y la configuración, es importante entender conceptos relacionados con la alta disponibilidad de Azure.

Conceptos

Las construcciones fundamentales proporcionadas por Azure para lograr la alta disponibilidad son:

- Conjuntos de disponibilidad
- Dominio de error
- Dominio de actualización
- Zonas de disponibilidad

Conjuntos de disponibilidad

La alta disponibilidad en Azure se logra principalmente a través de la **redundancia**.

La redundancia significa que hay más de una instancia de un recurso, de manera que, en caso de fallo de un recurso, el otro asume el control. Sin embargo, el simple hecho de tener más recursos similares no los hace altamente disponibles. Por ejemplo, tener más de una máquina virtual no hace que estas máquinas virtuales sean altamente disponibles. Azure proporciona conjuntos de recursos de disponibilidad para etiquetar y reunir estos recursos para formar un grupo. Todas las máquinas virtuales del conjunto de disponibilidad se convierten en altamente disponibles porque se colocan en soportes físicos independientes en el centro de datos de Azure y las máquinas virtuales se actualizan de una en una en lugar de todas al mismo tiempo. Los conjuntos de disponibilidad proporcionan un dominio de error y un dominio de actualización para lograrlo, algo que se comenta en la siguiente sección. En resumen, los conjuntos de disponibilidad proporcionan una redundancia en el nivel de centro de datos similar al almacenamiento con redundancia local.

Dominio de error

Cuando se aprovisiona y se asigna una máquina virtual a un conjunto de disponibilidad, se le asigna un dominio de error. Con **Azure Resource Manager (ARM)**, cada conjunto de disponibilidad tiene dos o tres dominios de errores predeterminados en función de las regiones de Azure. Algunos proporcionan dos, mientras que otros proporcionan tres dominios de errores en un conjunto de disponibilidad. Los usuarios no pueden configurar los dominios de errores. Cuando se crean varias máquinas virtuales, se colocan en dominios de errores diferentes. Si se aprovisionan más de cinco máquinas virtuales en un conjunto de disponibilidad, se colocan en forma round-robin en cinco dominios de errores. Los dominios de errores están relacionados con soportes físicos del centro de datos de Azure. Los dominios de errores proporcionan una alta disponibilidad del mantenimiento no planificado debido a fallos de hardware, alimentación y red. Puesto que una sola máquina virtual solo se coloca en un soporte, otras máquinas virtuales siguen funcionando en caso de que el soporte en cuestión se rompa.

Dominio de actualización

El dominio de error se encarga del mantenimiento no planificado y el dominio de actualización gestiona el mantenimiento planificado. Cada máquina virtual se asigna también a un dominio de actualización. Puede haber hasta 20 dominios de actualización en un único conjunto de disponibilidad. Los usuarios no pueden configurar los dominios de actualización. Cuando se crean varias máquinas virtuales, se colocan en dominios de actualización diferentes. Si se aprovisionan más de 20 máquinas virtuales en un conjunto de disponibilidad, se colocan en forma round-robin en estos dominios de actualización. Los dominios de actualización se encargan del mantenimiento planificado.

Zonas de disponibilidad

Este es un concepto relativamente nuevo introducido en Azure y muy similar a la redundancia de zona que vimos para las cuentas de almacenamiento. Las zonas de disponibilidad proporcionan alta disponibilidad dentro de una región mediante la colocación de instancias de máquinas virtuales en centros de datos diferentes dentro de una región. Las zonas de disponibilidad son aplicables a máquinas virtuales, discos gestionados, conjuntos de escalado de máquinas virtuales y equilibradores de carga. Esta ha sido una carencia de Azure durante mucho tiempo, que se ha eliminado recientemente desde una perspectiva de alta disponibilidad informática. Más información sobre las zonas de disponibilidad en <https://docs.microsoft.com/azure/availability-zones/az-overview>.

Equilibrio de carga

El equilibrio de carga, como su nombre indica, se refiere al proceso de equilibrar la carga entre máquinas virtuales y aplicaciones. Con una máquina virtual, no hay necesidad de equilibrador de carga porque toda la carga está en una sola máquina virtual y no hay ninguna otra máquina virtual para compartirla. Sin embargo, con varias máquinas virtuales con la misma aplicación y servicio, es posible distribuir la carga entre ellas a través del equilibrio de carga. Azure proporciona un par de recursos para habilitar el equilibrio de carga:

- **Equilibradores de carga:** el equilibrador de carga de Azure ayuda a diseñar soluciones con alta disponibilidad. Dentro de la pila TCP, hay un equilibrador de carga de nivel de transporte de capa 4. Es un equilibrador de carga de capa 4 que distribuye el tráfico entrante entre instancias en buen estado de servicios definidos en un conjunto con equilibrio de carga. Los equilibradores de carga de nivel 4 trabajan en el nivel de transporte y tienen información de nivel de red, como la dirección IP y el puerto, para decidir el destino de la solicitud entrante. Los equilibrios de carga se comentan en detalle más adelante en este capítulo.

- **Puertas de enlace de aplicaciones:** las puertas de enlace de aplicaciones de Azure ofrecen una alta disponibilidad a tus aplicaciones. Es un equilibrador de carga de capa 7 que distribuye el tráfico entrante entre instancias de servicios en buen estado. Los equilibradores de carga de nivel 7 pueden trabajar en el nivel de aplicación y tienen la información de nivel de aplicación como cookies, HTTP, HTTPS y sesiones para la solicitud entrante. Las puertas de enlace de aplicaciones se comentan en detalle más adelante en este capítulo.

Alta disponibilidad de las máquinas virtuales

Las máquinas virtuales proporcionan capacidades de cálculo. Proporcionan potencia de procesamiento y hosting para aplicaciones y servicios. Si la aplicación se implementa en una sola máquina virtual y la máquina deja de funcionar, entonces incluso la aplicación deja de estar disponible. Si la aplicación se compone de varios niveles y cada nivel se implementa en su propia instancia individual de una máquina virtual, incluso un tiempo de inactividad en una sola máquina virtual puede hacer que la aplicación no esté disponible. Aunque Azure intenta hacer que hasta las implementaciones en una única máquina virtual estén altamente disponibles mediante la colocación en soportes diferentes tan pronto como lo descubre, Azure no asegura ni garantiza ningún SLA sobre ellos. Azure proporciona SLA para las máquinas virtuales que están agrupadas en un conjunto de disponibilidad. Proporciona SLA con cinco nueves (99,999 %) para la disponibilidad de las máquinas virtuales si forman parte de un conjunto de disponibilidad y en dicho conjunto de disponibilidad hay más de dos máquinas virtuales.

Alta disponibilidad informática

Las aplicaciones que exigen alta disponibilidad deben implementarse en múltiples máquinas virtuales en el mismo conjunto de disponibilidad. Si las aplicaciones se componen de múltiples niveles, entonces cada nivel debe tener un grupo de máquinas virtuales en su conjunto de disponibilidad dedicado. En definitiva, si hay tres niveles de una aplicación, debe haber tres conjuntos de disponibilidad y un mínimo de seis máquinas virtuales (dos en cada conjunto de disponibilidad) para que toda la aplicación sea altamente disponible.

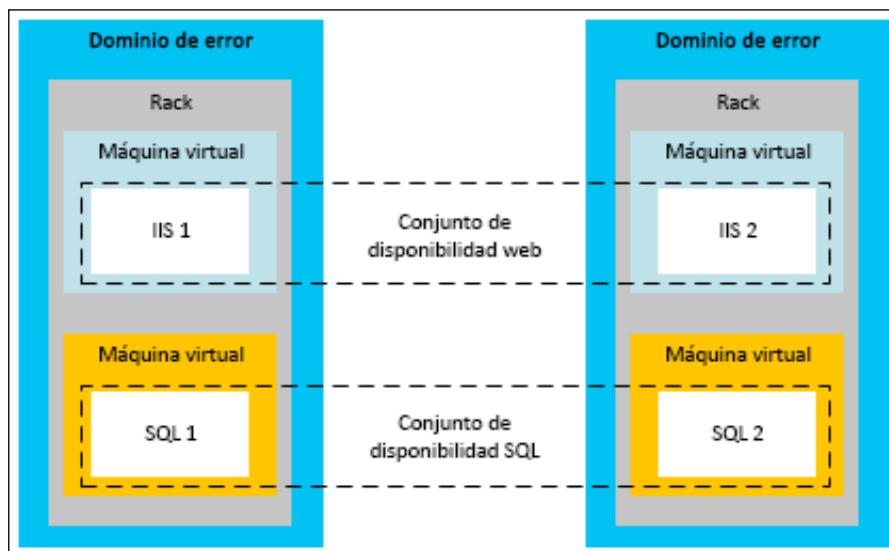
¿Cómo proporciona Azure SLA y alta disponibilidad a las máquinas virtuales de un conjunto de disponibilidad con múltiples máquinas virtuales en cada conjunto de disponibilidad? Esta es la pregunta que podría venir a la mente.

Aquí entra el uso de los conceptos que hemos planteado antes: dominio de error y de actualización.

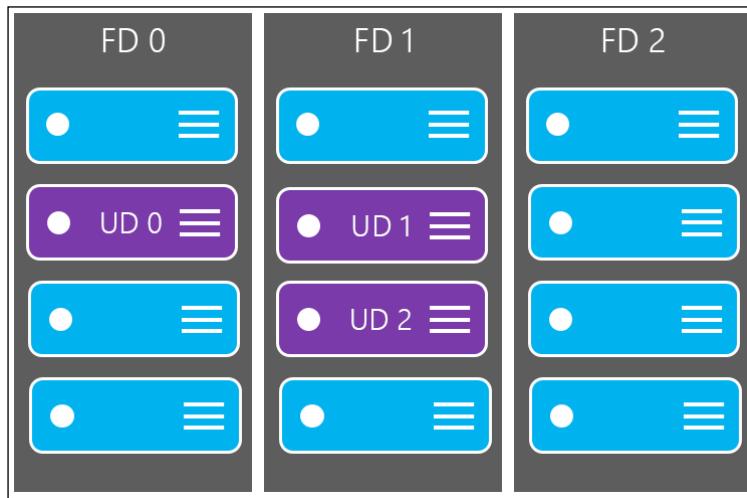
Cuando Azure ve varias máquinas virtuales en un conjunto de disponibilidad, coloca esas máquinas virtuales en un dominio de error diferente. En otras palabras, estas máquinas virtuales se colocan en soportes físicos separados en lugar de en el mismo soporte. Esto garantiza que al menos una máquina virtual siga disponible incluso si se produce un error de alimentación, hardware o soporte. Hay dos o tres dominios de error en un conjunto de disponibilidad y, dependiendo del número de máquinas virtuales en un conjunto de disponibilidad, las VM se colocan en dominios de error separados o se repiten en forma round-robin. Esto garantiza que la alta disponibilidad no se vea afectada debido a un error del soporte.

Azure también coloca estas VM en un dominio de actualización independiente. En otras palabras, Azure etiqueta estas VM internamente de tal forma que dichas máquinas virtuales se revisan y se actualizan una tras otra, de manera que ningún reinicio de un dominio de actualización afecte a la disponibilidad de la aplicación. Esto asegura que la alta disponibilidad no se vea afectada debido al mantenimiento de la máquina virtual y el host.

Con la colocación de máquinas virtuales en dominios de error y de actualización independientes, Azure garantiza que no dejen de funcionar todas al mismo tiempo y que estén activas y disponibles para dar servicios a las solicitudes, incluso aunque se estén sometiendo a tareas de mantenimiento o afrontando problemas de inactividad física.



La imagen anterior muestra cuatro máquinas virtuales (dos relacionadas con IIS y dos con SQL). Tanto las máquinas virtuales IIS como las SQL forman parte de su conjunto de disponibilidad. Las máquinas virtuales IIS y SQL se encuentran en dominios de errores independientes y en soportes diferentes del centro de datos. También deben estar en dominios de actualización independientes.



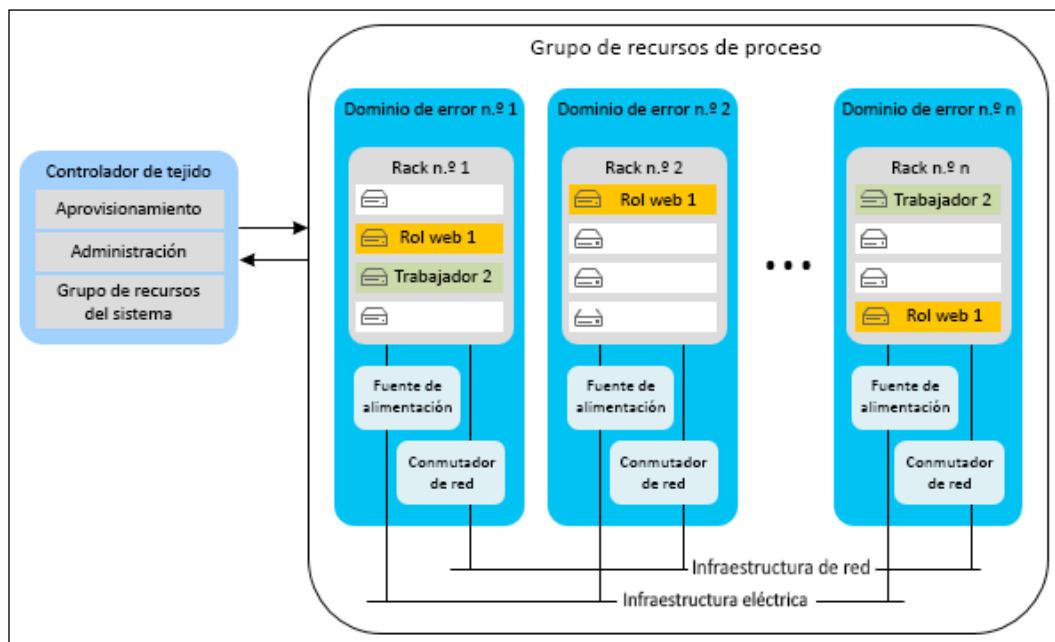
Alta disponibilidad del almacenamiento

Las máquinas virtuales están respaldadas por cuentas de almacenamiento, donde se almacenan sus archivos VHD. Aunque los conjuntos de disponibilidad proporcionan alta disponibilidad para instancias de proceso, no garantizan la alta disponibilidad de los archivos VHD para máquinas virtuales almacenadas en cuentas de almacenamiento. Los archivos VHD para todas las VM podrían colocarse en el mismo clúster de almacenamiento y cualquier error del clúster puede hacer que todas las máquinas virtuales dejen de estar disponibles o estén menos disponibles de lo necesario. En definitiva, no solo servicios informáticos deben ser altamente disponibles, sino que incluso las cuentas de almacenamiento de archivos VHD deben colocarse en clústeres separados de manera que, en caso de error, al menos una o varias máquinas virtuales sigan estando disponibles tanto desde el punto de vista de procesamiento como de almacenamiento.

Azure proporciona discos administrados como un concepto que proporciona instalaciones de administración de discos. Los discos administrados proporcionan una mayor fiabilidad para los conjuntos de disponibilidad al garantizar que los discos de las VM de un conjunto de disponibilidad estén suficientemente aislados entre sí para evitar puntos únicos de error. Esto se logra colocando automáticamente los discos en diferentes clústeres de almacenamiento. Si un clúster de almacenamiento falla debido a un error de hardware o software, solo fallan las instancias de VM con discos en esos sellos. Cada máquina virtual VHD de un conjunto de disponibilidad debe colocarse en una cuenta de almacenamiento diferente. No obstante, las máquinas virtuales de conjuntos de disponibilidad diferentes pueden colocarse en una cuenta de almacenamiento.

Alta disponibilidad de PaaS

Azure proporciona servicios de aplicaciones y servicios en el cloud para plataformas de hosting administradas. Los servicios y las aplicaciones se pueden implementar sobre ellos. Proporcionan flexibilidad, elasticidad y ahorro para crear e implementar aplicaciones. Estas plataformas están gestionadas por Azure y los usuarios no interactúan con la infraestructura de base sobre la que están implementadas. Aportan un mayor nivel de abstracción en comparación con la IaaS al permitir que los desarrolladores se concentren en su problema empresarial y utilizar la plataforma para acelerar su proceso de desarrollo e implementación. Esto permite aligerar su carga para administrar, operar y supervisar la infraestructura de base. Cuando una aplicación se implementa en servicios en la aplicación o en servicios en el cloud, Azure aprovisiona máquinas virtuales que no son visibles para los usuarios. Las aplicaciones se implementan en estas máquinas virtuales y el controlador de tejido de Azure es responsable de aprovisionarlas, administrarlas y controlarlas. El controlador de tejido controla el estado del hardware y el software de las instancias de las máquinas que funcionan como host e invitado. Cuando detecta un error, mantiene los SLA recolocando automáticamente las instancias de VM. Cuando se implementan múltiples instancias de rol de servicio en el cloud, Azure implementa estas instancias en diferentes dominios de error y actualización.



El diagrama anterior muestra que los servicios de PaaS con varias instancias de máquinas virtuales implementan estos roles web y de trabajo en dominios de error independientes. La implementación en dominios de error independientes supone una implementación en soportes independientes dentro de un centro de datos. También significa que estos servicios tienen unos commutadores de red y una fuente de alimentación independientes, lo que garantiza que, incluso si uno de los soportes se somete a mantenimiento, si se interrumpe el suministro de energía al soporte o si se produce un fallo del commutador de red, existen otras instancias disponibles para dar servicio a las solicitudes del consumidor.

Alta disponibilidad de aplicaciones

La alta disponibilidad puede incorporarse en el software utilizado para las aplicaciones o puede crearse desde cero en las aplicaciones. Un ejemplo de la función de alta disponibilidad que proporciona el software es SQL Server, siempre en grupos de disponibilidad. Así, se ayuda a mantener la alta disponibilidad de las bases de datos.

Los servicios Azure también tienen un mecanismo incorporado de alta disponibilidad. En Azure SQL, los datos se replican sincrónicamente dentro de la región. La replicación geográfica activa permite hasta cuatro copias adicionales de la base de datos en la misma región o en regiones diferentes. El almacenamiento de Azure tiene su propio mecanismo para hacer que los datos estén disponibles mediante su replicación en diversos centros de datos y regiones.

Equilibrio de carga

Azure proporciona dos diseños para aprovisionar equilibradores de carga. Proporciona un equilibrador de carga de nivel 4 que funciona en la capa de transporte en la pila TCP OSI, así como un equilibrador de carga de nivel 7 que funciona en las aplicaciones y las sesiones.

Aunque tanto las puertas de enlace de aplicaciones como el equilibrador de carga ofrecen características básicas de equilibrado de la carga, cumplen funciones distintas. Existen casos en los que tiene más sentido implementar la puerta de enlace de aplicaciones que el equilibrador de carga.

La puerta de enlace de aplicaciones proporciona las siguientes funciones que no están disponibles en los equilibradores de carga de Azure:

- **Firewall de aplicaciones web:** es un firewall adicional sobre el firewall del sistema operativo que tiene la capacidad de comprobar los mensajes entrantes. Esto ayuda a detectar y protegerse frente a los frecuentes ataques basados en web, como inyección SQL, ataques de ejecución de secuencias de comandos en sitios cruzados y secuestros de sesión.

- **Afinidad de sesión basada en cookies:** los equilibradores de carga distribuyen el tráfico entrante hacia las instancias de servicios que están en buen estado y relativamente libres. Una solicitud puede ser atendida por cualquier instancia de servicio. Hay aplicaciones que necesitan funciones avanzadas en las que todas las solicitudes posteriores a la primera solicitud deben ser procesadas por la misma instancia de servicio. Esto se conoce como afinidad de sesión basada en cookies. La puerta de enlace de aplicaciones proporciona afinidad de sesión basada en cookies para mantener una sesión de usuario en la misma instancia de servicio mediante el uso de cookies.
- **Descarga de capa de sockets seguros (SSL):** el cifrado y el descifrado de los datos de solicitud y respuesta se realizan mediante SSL y, por lo general, se trata de una operación costosa. Lo ideal es que los servidores web inviertan sus recursos en procesar y dar servicio a solicitudes en lugar de en cifrar y descifrar el tráfico. La descarga de SSL ayuda a transferir este proceso de cifrado desde el servidor web hasta el equilibrador de carga, lo que proporciona más recursos a los servidores web que prestan servicio a los usuarios. La solicitud del usuario se cifra, pero se descifra en la puerta de enlace de aplicaciones en lugar de en el servidor web. La solicitud de la puerta de enlace de aplicaciones al servidor web no está cifrada.
- **SSL integral:** aunque la descarga de SSL es una buena función para una aplicación determinada, hay ciertas aplicaciones de seguridad críticas que requieren un cifrado y descifrado SSL completo, incluso aunque el tráfico pase a través de equilibradores de carga. La puerta de enlace de aplicaciones también puede configurarse para un cifrado SSL integral.
- **Enrutamiento de contenido basado en URL:** la puerta de enlace de aplicaciones también es útil para redirigir el tráfico hacia diferentes servidores basándose en el contenido de la URL de las solicitudes entrantes. Esto ayuda a hospedar varios servicios junto con otras aplicaciones.

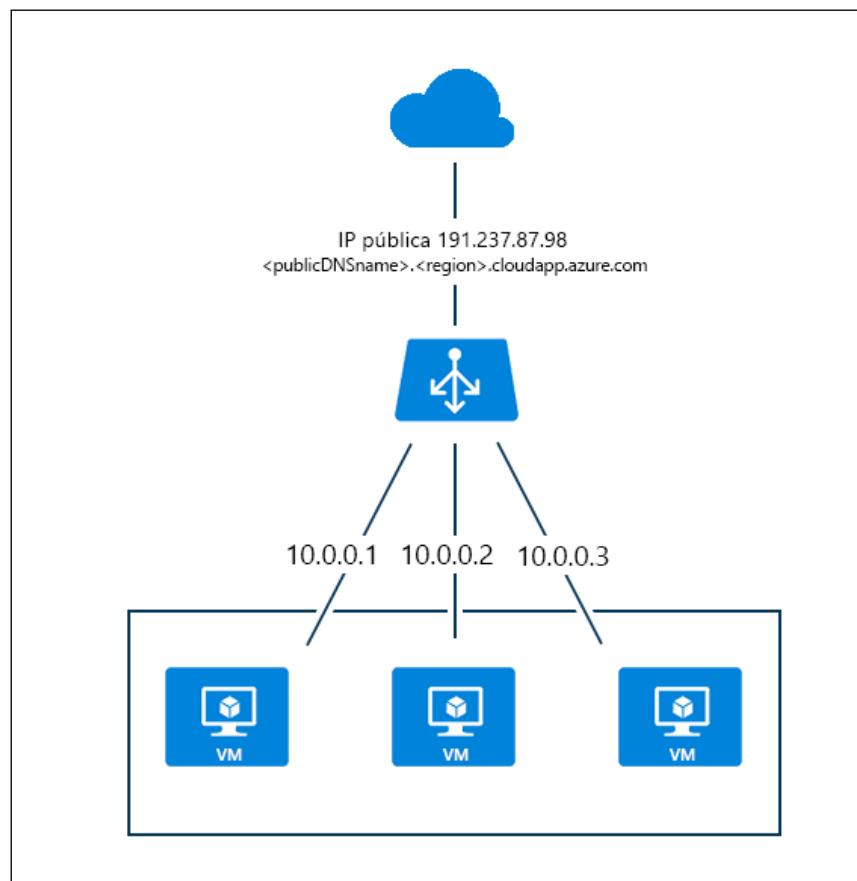
Equilibradores de carga de Azure

El equilibrador de carga de Azure distribuye el tráfico entrante basándose en la información de nivel de transporte a su disposición. Depende de lo siguiente:

- Dirección IP de origen
- Dirección IP de destino
- Número de puerto de origen
- Número de puerto de destino
- Tipo de protocolo: TCP o HTTP

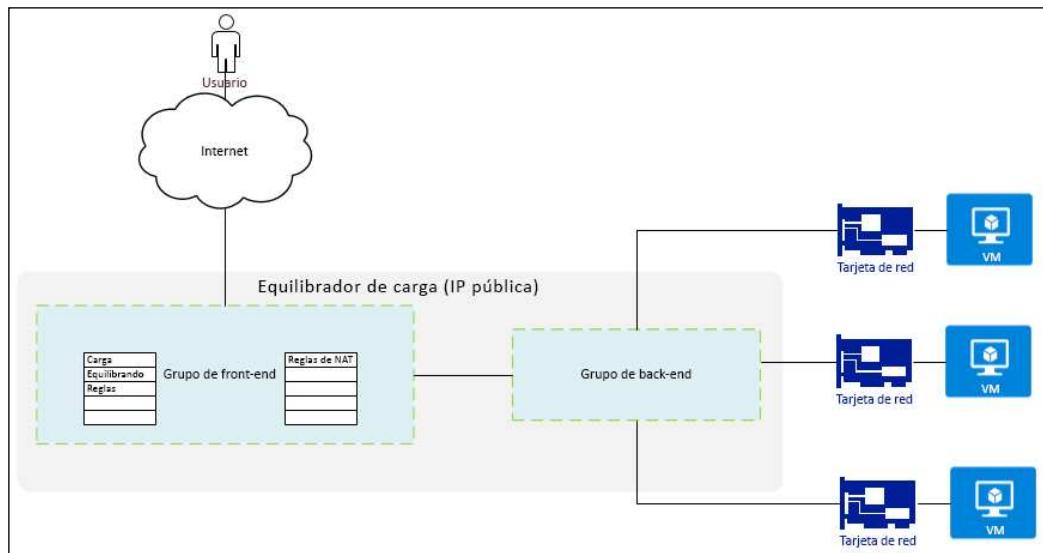
Equilibrio de carga público

En esta configuración, los equilibradores de carga se asignan a una dirección IP pública. La asignación de una dirección IP pública garantiza que el equilibrador de carga pueda aceptar las solicitudes procedentes de Internet. Sin una dirección IP pública no es posible acceder al recurso de Internet. El equilibrador de carga puede configurarse con las reglas de equilibrio de carga. Las reglas de equilibrio de carga funcionan en el nivel de puerto. Acepta que los puertos de origen y destino se asignen de forma conjunta de manera que siempre que un equilibrador de carga reciba una solicitud para el puerto de origen, la solicitud se reenvíe a una máquina virtual de un grupo de máquinas virtuales conectado al equilibrador de carga del puerto de destino. Esto se muestra en el siguiente diagrama:



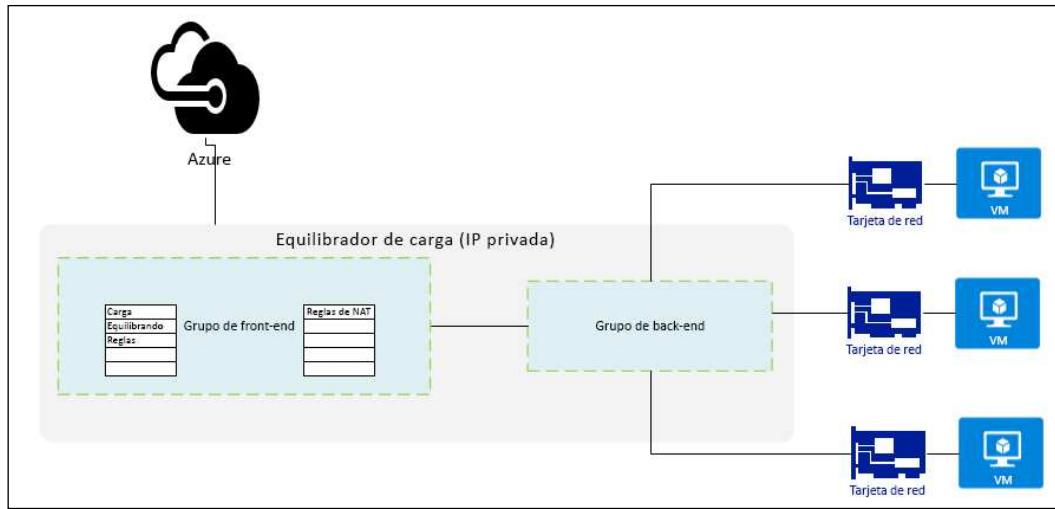
¿Cómo funciona todo esto? ¿Cómo se asigna una dirección IP pública a un equilibrador de carga? ¿Qué contiene el equilibrador de carga? ¿Cómo se configura con las reglas del equilibrador de carga? ¿Cómo envía el equilibrador de carga solicitudes a las máquinas virtuales? ¿Cómo sabe la máquina virtual que está conectada al equilibrador de carga, entre otras cosas? La respuesta a todas estas preguntas puede verse en el siguiente diagrama.

En esta configuración, el equilibrador de carga se asigna a una dirección IP pública. El equilibrador de carga es accesible desde Internet y puede aceptar solicitudes de cliente. El equilibrador de carga puede configurarse con las reglas de equilibrio de carga y NAT. Tanto las reglas NAT como las de equilibrio de carga forman parte de la configuración del front-end. La configuración del front-end envía las solicitudes de cliente a una de las direcciones IP disponibles en el grupo de back-end. Estas direcciones IP se asocian con la tarjeta de interfaz de red, que a su vez se conecta a máquinas virtuales.

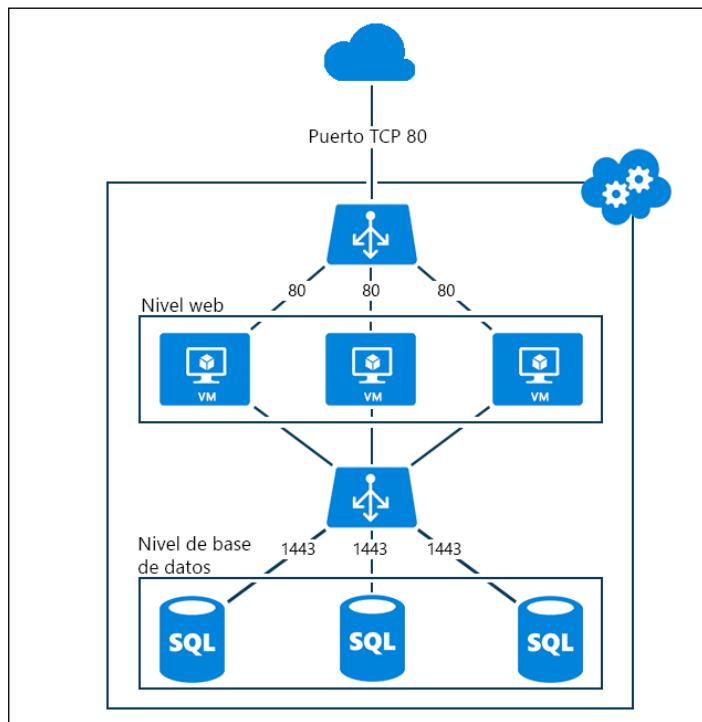


Equilibrio de carga interno

El siguiente diagrama muestra el funcionamiento de un equilibrador de carga interno. Se puede ver que la solicitud proviene de recursos del propio Azure, ya que no es accesible en Internet. En esta configuración, el equilibrador de carga se asigna a una dirección IP privada. Solo se puede acceder al equilibrador de carga dentro de la red virtual a la que está conectado. No puede accederse a él a través de Internet. El resto de su configuración es como la de un equilibrador de carga público. El equilibrador de carga puede configurarse con las reglas de equilibrio de carga y NAT.



El siguiente diagrama muestra cómo varios equilibradores de carga pueden implementarse para crear soluciones. En este sentido, hay un equilibrador de carga público que acepta solicitudes de clientes y un equilibrador de carga interno para las bases de datos. No se puede acceder en Internet a las máquinas virtuales de bases de datos, sino solo a través del equilibrador de carga en el puerto 1433.



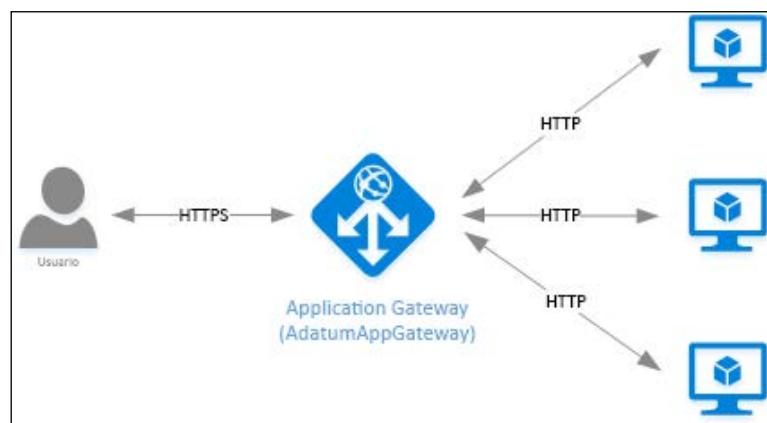
Reenvío de puertos

A veces, existe una necesidad en la que una solicitud siempre se debe redirigir a una máquina virtual. El equilibrador de carga de Azure nos ayuda a lograrlo con las reglas de NAT. Las reglas de NAT se evalúan después de haber evaluado las reglas de equilibrio de carga y no haberse cumplido ninguna de ellas. Las reglas de NAT se evalúan para cada solicitud entrante y, una vez que se encuentran, se remite la solicitud a esa máquina virtual a través de un grupo de back-end. Cabe señalar que una máquina virtual no puede registrar el mismo puerto para el reenvío de puertos mediante reglas de NAT y reglas de equilibrio de carga.

Puertas de enlace de aplicaciones de Azure

El equilibrador de carga de Azure nos ayuda a habilitar soluciones en la infraestructura. Sin embargo, hay ocasiones en las que, desde el propio equilibrador de carga, se requieren servicios y características avanzados. Estos servicios avanzados incluyen terminación SSL, sesiones adhesivas, seguridad avanzada, etc. La puerta de enlace de aplicaciones de Azure se basa en los equilibradores de carga de Azure para proporcionar estas funciones adicionales. La puerta de enlace de aplicaciones de Azure es un equilibrador de carga de nivel 7 que funciona con la carga de la aplicación y la sesión en una pila OSI TCP. Las puertas de enlace de aplicaciones tienen más información que el equilibrador de carga de Azure para tomar decisiones sobre el enruteamiento de las solicitudes y el equilibrio de carga entre servidores. Azure gestiona las puertas de enlace de aplicaciones, que son altamente disponibles.

Una puerta de enlace de aplicaciones se encuentra entre los usuarios y las máquinas virtuales, como se muestra en la figura siguiente:



Las puertas enlace de aplicaciones se implementan internamente mediante máquinas virtuales. **Internet Information Services (IIS)** está instalado y configurado con **Enrutamiento de solicitud de aplicaciones (ARR)** en estas máquinas virtuales. Estas puertas de enlace pueden instalarse en varias máquinas virtuales, lo que proporciona alta disponibilidad para las propias puertas de enlace. Aunque no son visibles, los equilibradores de carga de Azure distribuyen las cargas entre varios servidores de puertas de enlace de aplicaciones. La creación de una puerta de enlace de aplicaciones necesita una dirección IP interna o pública, que los usuarios utilizan para enviarle solicitudes. Esta IP pública o IP interna la proporciona el equilibrador de carga de Azure, que trabaja en el nivel de transporte (TCP/UDP), y cuya carga de tráfico de red entrante se equilibra hacia instancias de trabajo de la puerta de enlace de aplicaciones. La puerta de enlace de aplicaciones enruta entonces el tráfico HTTP/HTTPS en función de su configuración, ya sea una máquina virtual, un servicio en el cloud, o una dirección IP interna o externa.

Una puerta de enlace de aplicaciones es similar al equilibrador de carga de Azure en cuanto a la configuración, aunque posee características y diseños adicionales. Proporciona IP de front-end, protocolos, configuración de certificados y puertos, grupo de back-end, puerto, afinidad de sesión y configuración de protocolos.

Azure Traffic Manager

Después de tener un buen conocimiento del equilibrador de carga de Azure y de la puerta de enlace de aplicaciones, es el momento de adentrarse en los detalles de Traffic Manager. Los equilibradores de carga y las puertas de enlace de aplicaciones de Azure son recursos indispensables para la alta disponibilidad en un centro de datos y una región; sin embargo, para lograr la alta disponibilidad en todas las regiones y centros de datos, se necesita otro recurso, que es Traffic Manager. Traffic Manager nos ayuda a crear soluciones de alta disponibilidad que abarcan múltiples ubicaciones geográficas, regiones y centros de datos. Traffic Manager no es similar a los equilibradores de carga. Utiliza DNS para redirigir las solicitudes a un punto de conexión apropiado determinado por su estado y su configuración. Traffic Manager no es un proxy ni una puerta de enlace. Traffic Manager no ve el tráfico que pasa entre el cliente y el servicio. Simplemente redirige la solicitud basándose en los puntos de conexión más apropiados.

Azure Traffic Manager te permite controlar la distribución del tráfico entre los puntos de conexión de tu aplicación. Un punto de conexión es cualquier servicio de conexión a Internet alojado dentro o fuera de Azure.

Los puntos de conexión son URL públicas accesibles expuestas a Internet. Las aplicaciones se aprovisionan en múltiples ubicaciones geográficas y regiones de Azure. Las aplicaciones implementadas en cada región tienen un punto de conexión único que recibe el nombre de **DNS CNAME**. Estos puntos de conexión se asignan al punto de conexión Traffic Manager. Cuando se aprovisiona un Traffic Manager, obtiene un punto de conexión predeterminado con una extensión URL `.trafficmanager.net`.

Cuando llega una solicitud a la URL de Traffic Manager, encuentra el punto de conexión más adecuado de su lista y le redirige la solicitud. En resumen, Traffic Manager actúa como un DNS global para identificar la región que atenderá la solicitud.

Sin embargo, ¿cómo sabe Traffic Manager qué puntos de conexión utilizar y hacia cuáles redirigir la solicitud de cliente? Hay dos aspectos que Traffic Manager implementa para determinar la región y el punto de conexión más apropiados.

En primer lugar, Traffic Manager supervisa activamente el estado de todos los puntos de conexión. Puede supervisar el estado de máquinas virtuales, servicios en el cloud y servicios de aplicación. Si determina que el estado de una aplicación implementada en una región no es adecuado para redireccionar el tráfico hacia ella, redirige las solicitudes a un punto de conexión en buen estado.

En segundo lugar, Traffic Manager puede configurarse con información de enrutamiento. Existen cuatro métodos de enrutamiento de tráfico en Traffic Manager:

- **Prioridad:** debe utilizarse cuando todo el tráfico debe ir a un punto de conexión predeterminado y existen copias de seguridad en caso de que los puntos de conexión principales no estén disponibles.
- **Ponderado:** debe utilizarse para distribuir el tráfico entre todos los puntos de conexión de manera uniforme o según pesos definidos.
- **Rendimiento:** debe usarse para puntos de conexión en diferentes regiones y los usuarios deben redirigirse hasta el punto de conexión más cercano según su ubicación. Esto repercute de forma directa en la latencia de red.
- **Geográfico:** debe usarse para redirigir a los usuarios de una ubicación geográfica específica a un punto de conexión (Azure, externo, o anidado) disponible en esa ubicación geográfica o más cercano a esa ubicación geográfica. Los ejemplos incluyen el cumplimiento de mandatos de soberanía de los datos, localización de la experiencia de usuario y el contenido, así como medición del tráfico de diferentes regiones.

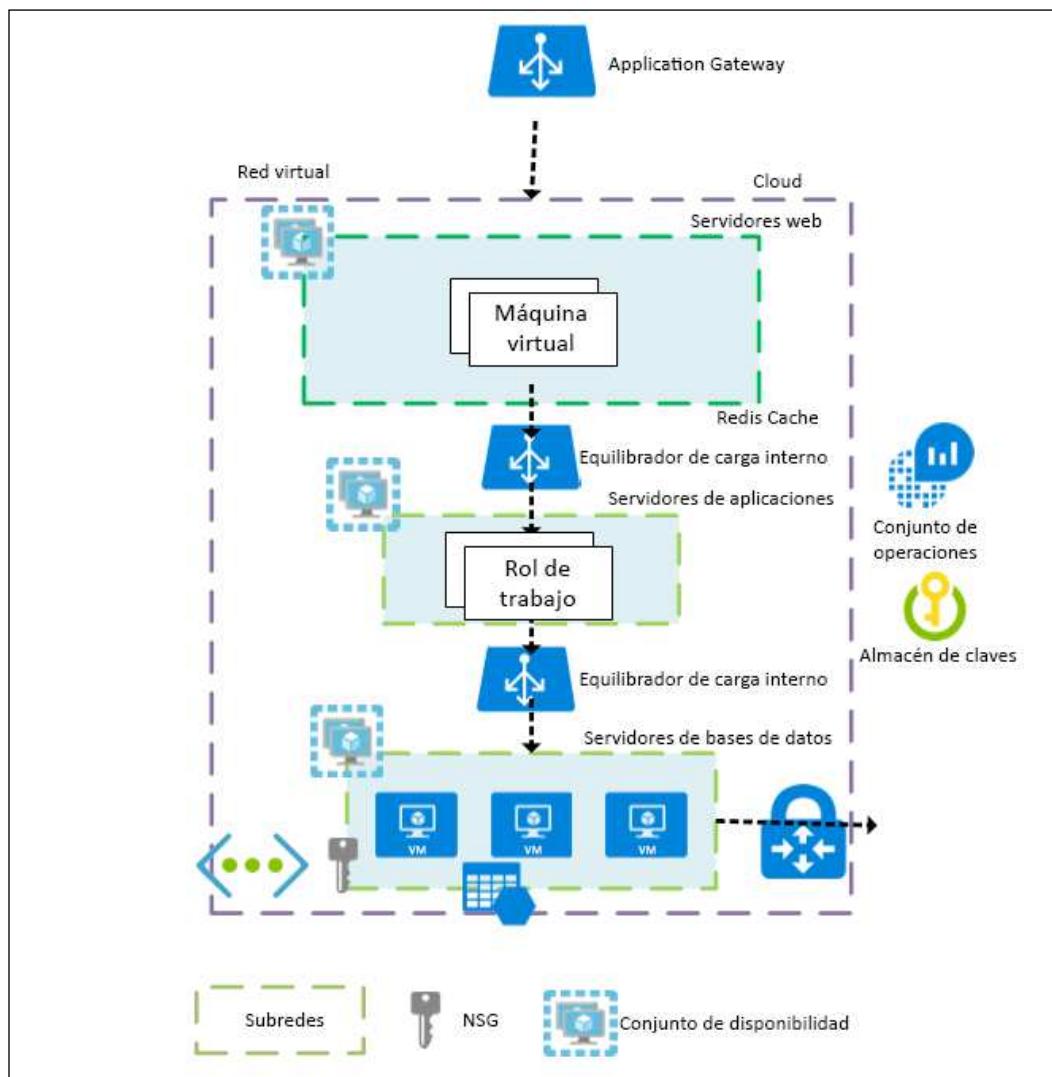
Cabe indicar que después de que Traffic Manager determine un punto de conexión en buen estado válido, los clientes se conectan directamente a la aplicación.

Consideraciones sobre la arquitectura de alta disponibilidad

En esta sección, estudiaremos algunas de las arquitecturas de alta disponibilidad.

Alta disponibilidad en las regiones de Azure

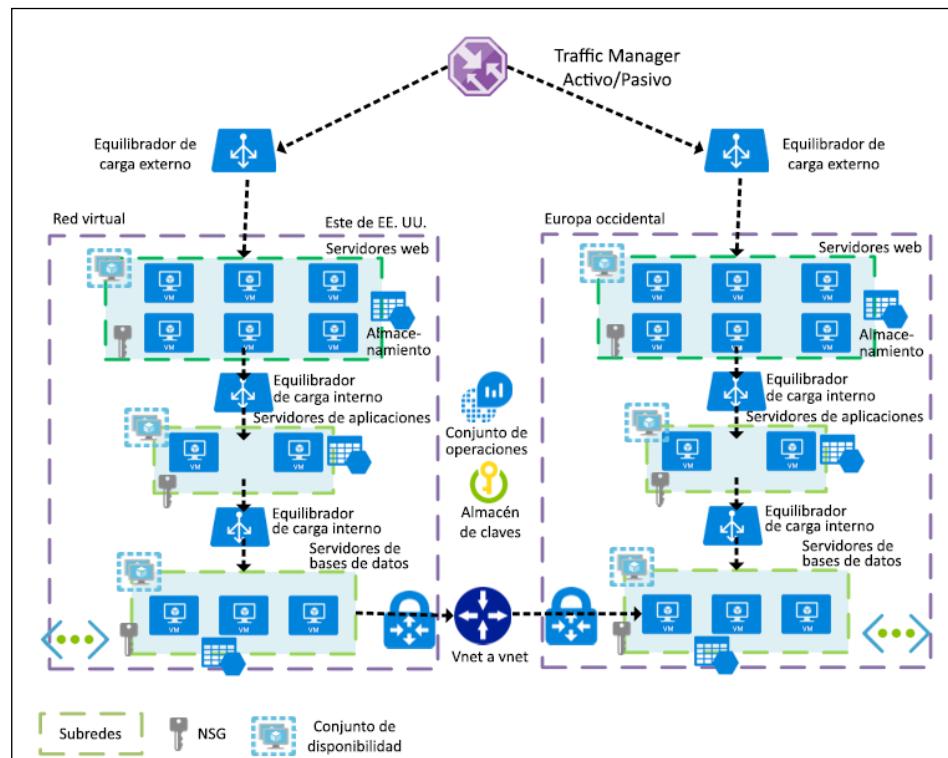
La arquitectura que se presenta a continuación muestra la implementación de alta disponibilidad en una única región de Azure. La alta disponibilidad está diseñada en el nivel de recursos individuales. En esta arquitectura, hay múltiples máquinas virtuales en cada nivel conectadas a través de una puerta de enlace de aplicaciones o un equilibrador de carga, que forman parte de un conjunto de disponibilidad. Cada nivel está asociado con un conjunto de disponibilidad. Estas máquinas virtuales se colocan en dominios de error y actualización diferentes. Mientras los servidores web se conectan a las puertas de enlace de aplicaciones, el resto de los niveles, como los de aplicación y base de datos, tienen equilibradores de carga interno.



Alta disponibilidad en las regiones de Azure

La arquitectura que se presenta a continuación muestra implementaciones similares en dos regiones diferentes de Azure. Ambas regiones tienen los mismos recursos implementados. La alta disponibilidad está diseñada en el nivel de recursos individuales dentro de estas regiones. Hay varias máquinas virtuales en cada nivel conectadas a través del equilibrador de carga, que forman parte del conjunto de disponibilidad. Estas máquinas virtuales se colocan en dominios de error y actualización diferentes. Mientras los servidores web se conectan a equilibradores de carga externos, el resto de los niveles, como los de aplicación y base de datos, tiene equilibradores de carga interna. Cabe señalar que los equilibradores de carga de la aplicación se podrían haber usado para los niveles de servidores web y aplicaciones en lugar del equilibrador de carga de Azure si existiera una necesidad de servicios avanzados, como afinidad de sesión, terminación SSL, seguridad avanzada con WAF y enruteamiento basado en ruta. Las bases de datos de ambas regiones están conectadas entre sí mediante emparejamiento de VNET y puertas de enlace. Esto es útil para la configuración del trasvase de registros, SQL Server AlwaysOn y otras técnicas de sincronización de datos.

Los puntos de conexión de los equilibradores de carga de ambas regiones se utilizan para configurar puntos de conexión de Traffic Manager y el tráfico se enruta basándose en el método de equilibrio de carga de prioridad. Traffic Manager ayuda al enruteamiento de todas las solicitudes de la región del Este de EE. UU. y comuta a Europa Occidental en caso de que la anterior región no esté disponible.



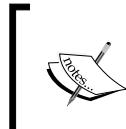
Prácticas recomendadas

Esta sección describe las prácticas recomendadas de alta disponibilidad. Se han clasificado en aplicación, implementación, administración de datos y supervisión.

Alta disponibilidad de aplicaciones

Al diseñarse una aplicación, hay que considerar la alta disponibilidad como una de las preocupaciones arquitectónicas importantes. A continuación, se mencionan algunas aplicaciones importantes relacionadas con los procedimientos de alta disponibilidad:

- Una aplicación debe implementar un manejo de excepciones adecuado para recuperar e informar correctamente a las partes interesadas del problema
- Una aplicación debe intentar volver a realizar la misma operación en el intervalo fijado un determinado número de veces antes de cerrarse en caso error o excepción
- Una aplicación debe tener una funcionalidad integrada de tiempo de espera para decidir que no se puede recuperar de una excepción
- El mantenimiento de registros y registros escritos de todos los errores, excepciones y ejecución debe adoptarse dentro de la aplicación
- Se deben crear perfiles de las aplicaciones para encontrar sus requisitos de recursos reales en términos de cálculo, me



Consulta <https://docs.microsoft.com/azure/architecture/checklist/availability> para obtener más información sobre la aplicación y el resto de las prácticas recomendadas de alta disponibilidad.

Implementación

La estrategia de implementación afecta en gran medida a la disponibilidad de las aplicaciones y al entorno general. Algunas de las cosas importantes que deben tenerse en cuenta son:

- Implementar múltiples instancias de recursos de Azure, incluidas varias instancias de máquinas virtuales, servicios en el cloud y otros recursos
- Implementar máquinas virtuales en conjuntos de disponibilidad o zonas de disponibilidad. No se pueden usar juntos

- Implementar múltiples instancias de máquinas virtuales en múltiples regiones
- Crear múltiples entornos y mantener al menos uno de ellos en modo de espera

Administración de datos

Algunos datos importantes relacionados con las prácticas recomendadas de alta disponibilidad son:

- Si es posible, almacene datos en los servicios proporcionados por Azure, como Azure SQL, Cosmos DB y table storage
- Utilice cuentas de almacenamiento basadas en las de redundancia geográfica
- Asegúrese de que los datos se replican a varias regiones y no solamente dentro de la zona o centro de datos
- Realice copias de seguridad periódicas y efectúe pruebas de restauración con frecuencia
- Si se almacenan datos en máquinas virtuales, asegúrate de que haya varias máquinas virtuales y de que estén en conjuntos o en zonas de disponibilidad
- Utiliza claves y secretos para los almacenes de datos de Azure Key Vault

Supervisión

Algunas de las prácticas recomendadas relacionadas con la supervisión para la alta disponibilidad son:

- Usar OMS (Log Analytics) para supervisar el entorno y habilitar la auditoría de registros
- Usar Application Insights para capturar la información de telemetría de la aplicación personalizada y el entorno relacionados con el cálculo, almacenamiento, red y otra información de registro
- Asegurarse de que alertas están configuradas en OMS para temas relacionados con la disponibilidad del entorno y la aplicación
- Visitar Azure Monitor con frecuencia para obtener recomendaciones relacionadas con la alta disponibilidad

Resumen

La alta disponibilidad es una preocupación arquitectónica importante y crucial. Casi todas las aplicaciones y arquitectos intentan implementar la alta disponibilidad. Tradicionalmente, se implementaba la alta disponibilidad desde cero sin ninguna ayuda de las plataformas. Azure es una plataforma madura que entiende la importancia de la alta disponibilidad para las aplicaciones y, en este ámbito, proporciona recursos para implementar la alta disponibilidad desde el nivel pormenorizado hasta el nivel de centro de datos. La alta disponibilidad no es una idea adicional y debe formar parte del desarrollo del ciclo de vida de las aplicaciones desde la misma fase de la planificación. Azure provee conjuntos de disponibilidad que garantizan que las máquinas virtuales se colocan en dominios de error y de actualización independientes. Los dominios de error aseguran que las perturbaciones imprevistas en el soporte no cambien la disponibilidad de una aplicación, mientras que los dominios de actualización se encargan del mantenimiento planificado. Incluso los servicios de Azure PaaS utilizan dominios de error y actualización entre bastidores para mantener la aplicación activa y en funcionamiento. El equilibrador de carga interno y externo de Azure son construcciones de transporte de nivel 4 proporcionadas por Azure para distribuir la carga entre varias instancias de la aplicación. Asimismo, las puertas de enlace de aplicaciones son equilibradores de carga de nivel 7 y de sesión que proporcionan funcionalidades avanzadas tales como terminación SSL y sesiones adhesivas. Traffic Managers son construcciones de alta disponibilidad de DNS globales para que aplicaciones de todas las ubicaciones geográficas sean altamente disponibles. El siguiente capítulo se centrará en la escalabilidad, que es otra faceta importante para la arquitectura global en Azure.

4

Implementación de la escalabilidad

Ejecutar aplicaciones y sistemas que están disponibles para su consumo por parte de los usuarios es un aspecto importante para los arquitectos en cualquier aplicación seria. Sin embargo, hay otra característica de las aplicaciones igualmente importante que es una de las principales prioridades de los arquitectos: la escalabilidad de las aplicaciones. Imagina situaciones en las que se implementan aplicaciones y obtienes un gran rendimiento y disponibilidad con unos pocos usuarios, pero tanto la disponibilidad como el rendimiento se deterioran a medida que aumenta el número de usuarios. En ocasiones, una aplicación con una carga normal funciona bien, pero su rendimiento se deteriora con el aumento del número de usuarios. Esto sucede sobre todo si se produce un aumento repentino en el número de usuarios y el entorno no está diseñado para una cantidad tan grande de usuarios. Para dar cabida a estos picos en el número de usuarios, puedes haber aprovisionado el hardware y el ancho de banda necesarios para gestionarlos. El problema de esto es que la capacidad adicional no se usa durante la mayor parte del año y no proporciona ningún retorno de la inversión. Su uso solo se apropria durante algunos festivales u ofertas. Espero que entiendas los problemas que los arquitectos tratan de resolver. Todos estos problemas están relacionados con el ajuste de tamaño de la capacidad y la escalabilidad de una aplicación. La prioridad de este capítulo es comprender la escalabilidad como una preocupación arquitectónica y comprobar los servicios que presta Azure para implementar la escalabilidad.

En este capítulo, abordaremos los siguientes temas:

- Escalabilidad
- Escalabilidad en las soluciones de infraestructura como servicio (IaaS) y plataforma como servicio (PaaS)
- Conceptos básicos de los conjuntos de escalado de máquinas virtuales
- Arquitectura de VMSS
- Escalado automático en VMSS

- Reglas de escalado automático
- Escalado o reducción vertical
- Escalado o reducción horizontal
- Automatización relacionada con VMSS

Escalabilidad

La planificación y el ajuste de tamaño de la capacidad son una de las principales prioridades que tienen los arquitectos para sus aplicaciones y servicios. Los arquitectos deben encontrar un equilibrio entre comprar y aprovisionar demasiados recursos o menos recursos. Tener menos recursos puede conllevar el no poder dar servicio a todos los usuarios, lo que hará que se pasen a la competencia, mientras que tener más recursos puede dañar los presupuestos y el retorno de las inversiones porque la mayoría de estos recursos permanece sin utilizar la mayor parte del tiempo. Por otra parte, el problema aumenta con la variación en el nivel de demanda durante las diferentes épocas del año. Es casi imposible predecir el número de usuarios de la aplicación a lo largo de cada día y durante todo el año. Sin embargo, es posible calcular un número aproximado utilizando información de períodos anteriores y una supervisión continua.

Según la Wikipedia (<https://en.wikipedia.org/wiki/Scalability>), esta es la definición de escalabilidad:

“La escalabilidad es la capacidad de un sistema, red o proceso de gestionar una cantidad creciente de trabajo o su potencial para ser ampliado y dar cabida a ese incremento. Por ejemplo, un sistema se considera escalable si es capaz de aumentar sus resultados totales bajo una carga mayor cuando se añaden recursos (normalmente, hardware)”.

La escalabilidad se refiere a la capacidad en la implementación, el proceso y la tecnología de la aplicación de gestionar el aumento del número de usuarios y de proporcionar el mismo nivel de rendimiento cuando hay menos usuarios. La escalabilidad puede hacer referencia a atender más solicitudes sin que se deteriore el rendimiento o puede implicar la gestión de un trabajo más grande que consume más tiempo sin que se produzca una pérdida de rendimiento en ninguno de los casos.

Los arquitectos deben llevar a cabo ejercicios de ajuste de tamaño y planificación de la capacidad al inicio del proyecto, durante la fase de planificación, para proporcionar escalabilidad a las aplicaciones.

Algunas aplicaciones tienen patrones de demanda estables, mientras que otros son difíciles de prever. Los requisitos de escalabilidad son conocidos en el caso de las aplicaciones con una demanda estable, mientras que es un proceso más complejo en las aplicaciones con una demanda variable. El escalado automático, un concepto que estudiaremos en la siguiente sección, debe utilizarse para las aplicaciones cuyas demandas no se pueden predecir.

Escalabilidad y rendimiento

Es bastante fácil confundirse entre escalabilidad y las preocupaciones arquitectónicas de rendimiento, porque la escalabilidad consiste en asegurar que, con independencia del número de usuarios que consuman la aplicación, todos obtengan el mismo nivel predeterminado de rendimiento.

El rendimiento está relacionado con las funciones de la aplicación que aseguran que esta satisface el tiempo de respuesta y el rendimiento predefinidos, la escalabilidad se refiere a la capacidad de suministrar más recursos según sea necesario para dar cabida a más usuarios sin sacrificar el rendimiento.

Se entiende mejor mediante una analogía. La velocidad de un tren hace referencia al rendimiento de los sistemas de ferrocarril, sin embargo, dar cabida a más trenes para que marchen en paralelo con la misma o mayor la velocidad se denomina escalabilidad de la red ferroviaria.

Escalabilidad de Azure

En esta sección, vamos a ver funciones y capacidades proporcionadas por Azure para hacer que las aplicaciones sean altamente disponibles. Antes de adentrarnos en los detalles arquitectónicos y de configuración, es importante entender conceptos relacionados con la alta disponibilidad de Azure.

Conceptos

Las construcciones fundamentales proporcionadas por Azure para lograr la alta disponibilidad son:

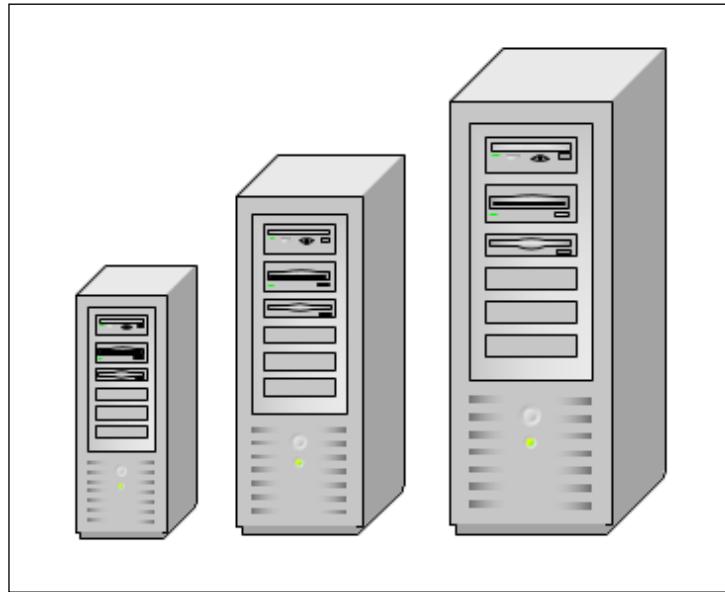
- Escalado
- Escalado o reducción vertical
- Escalado o reducción horizontal
- Escalado automático
- Actualizaciones graduales

Escalado

El escalado se refiere a la transformación que aumenta o disminuye las unidades de recursos utilizadas para atender las solicitudes de los usuarios. El escalado puede ser automático o manual. El escalado manual requiere que un administrador inicie manualmente el proceso de escalado, mientras que el escalado automático se refiere a un aumento o disminución automáticos de los recursos en función de los eventos disponibles en el entorno y el ecosistema, como la memoria y la disponibilidad de la CPU. El escalado y la reducción pueden ser verticales u horizontales, algo que se explicará más adelante en esta sección.

Escalado vertical

El escalado vertical de una máquina virtual o de un servicio hace referencia a la adición de recursos adicionales a los servidores existentes, como CPU, memoria y discos. Supone aumentar la capacidad del hardware y los recursos físicos existentes.



Reducción vertical

La reducción vertical de una máquina virtual o de un servicio se refiere a eliminar recursos existentes de los servidores existentes, como CPU, memoria y discos. Supone reducir la capacidad del hardware y los recursos físicos existentes.

Escalado horizontal

El escalado horizontal se refiere al proceso de adición de hardware adicional en términos de servidores y capacidad adicionales. Por lo general, se trata de agregar nuevos servidores, asignándoles direcciones IP, implementando aplicaciones en ellos y haciendo que formen parte de los equilibradores de carga existentes, de manera que el tráfico se pueda encaminar hacia ellos. El escalado horizontal también puede ser automático o manual. Sin embargo, para obtener mejores resultados, debe utilizar la automatización.



Reducción horizontal

La reducción horizontal se refiere al proceso de eliminación del hardware existente en términos de servidores y capacidad existentes. Normalmente esto implica la eliminación de los servidores existentes, la anulación de la asignación de sus direcciones IP, así como la eliminación de la configuración del equilibrador de carga existente, de manera que el tráfico no pueda encaminarse hacia ellos. Como el escalado horizontal, la reducción horizontal puede ser automática o manual.

Escalado automático

El escalado automático se refiere al proceso de escalado/reducción vertical o escalado/reducción horizontal de forma dinámica y según la demanda de la aplicación, que se produce mediante automatización. El escalado automático es útil porque garantiza que la implementación siempre conste de un número de instancias de servidor correcto e ideal. El escalado ayuda a crear aplicaciones con tolerancia a los errores. Esto no solo ayuda a la escalabilidad, sino que también hace que las aplicaciones estén altamente disponibles. Por último, ofrece la mejor administración de costes. El escalado automático ayuda a tener la configuración óptima para las instancias de servidor basándose en la demanda. Ayuda a no hacer un aprovisionamiento excesivo de servidores que estén infráutilizados o elimina servidores que ya no son necesarios después del escalado horizontal.

Escalabilidad PaaS

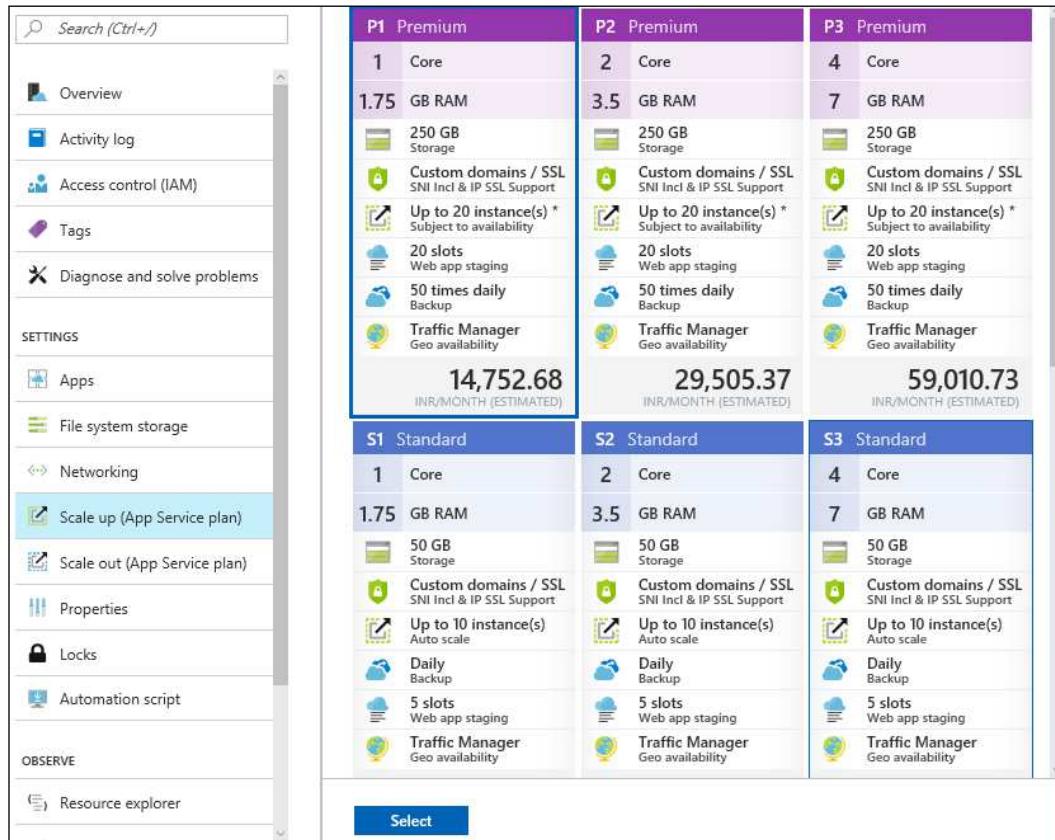
Azure proporciona servicios de aplicación para alojar aplicaciones administradas. Los servicios de aplicación son una oferta de PaaS de Azure. Proporciona la web y la plataforma móvil. Detrás de esta web y esta plataforma móvil hay una infraestructura administrada que gestiona Azure en nombre de sus usuarios. Los usuarios no ven ni administran la infraestructura; sin embargo, tienen la capacidad de ampliar la plataforma e implementar sus aplicaciones sobre ella. Con esto, los desarrolladores y arquitectos pueden concentrarse en sus problemas empresariales en lugar de preocuparse por la plataforma base y el aprovisionamiento de la infraestructura, la configuración y la resolución de problemas.

Los desarrolladores tienen la flexibilidad de elegir cualquier idioma, sistema operativo y marcos para desarrollar sus aplicaciones. Los servicios de aplicaciones ofrecen múltiples planes y, basándose en los planes seleccionados, existen capacidades de escalabilidad. Los servicios de aplicaciones ofrecen cinco planes. Son los siguientes:

- **Libre:** utiliza infraestructura compartida. Significa que se implementarán varias aplicaciones en la misma infraestructura de un mismo inquilino o de varios inquilinos. Consigue 1 GB de almacenamiento libre de coste. Este plan no dispone de instalación de escalado.
- **Compartido:** también utiliza infraestructura compartida y obtiene 1 GB de almacenamiento libre de coste. Además, también se proporcionan dominios personalizados como una característica extra. Este plan no dispone de instalación de escalado.
- **Básico:** tiene tres referencias **de almacén diferentes (SKU):** B1, B2 y B3. Cuenta con un mayor número de unidades de recursos disponibles para él en términos de CPU y memoria. En definitiva, proporciona una mayor configuración de máquinas virtuales que prestan soporte a estos servicios. Además, proporciona almacenamiento, dominios personalizados y soporte SSL. El plan básico ofrece funciones básicas para el escalado manual. Este plan no dispone de escalado automático. Se pueden utilizar como máximo tres instancias para el escalado horizontal de la aplicación.
- **Estándar:** también tiene tres SKU diferentes: S1, S2 y S3. Cuenta con un mayor número de unidades de recursos disponibles para él en términos de CPU y memoria. En definitiva, proporciona una mayor configuración de máquinas virtuales que prestan soporte a estos servicios. Además, proporciona almacenamiento, dominios personalizados y soporte SSL, de manera similar al plan básico. También ofrece Traffic Manager, ranuras de ensayo y una copia de seguridad diaria como función adicional al plan básico. El plan estándar ofrece funciones para el escalado manual. Se pueden utilizar como máximo diez instancias para el escalado horizontal de la aplicación.
- **Premium:** también tiene tres SKU diferentes: P1, P2 y P3. Cuenta con un mayor número de unidades de recursos disponibles para él en términos de CPU y memoria. En definitiva, proporciona una mayor configuración de máquinas virtuales que prestan soporte a estos servicios. Además, proporciona almacenamiento, dominios personalizados y soporte SSL, de manera similar al plan básico. También ofrece Traffic Manager, ranuras de ensayo y 50 copias de seguridad diarias como función adicional al plan básico. El plan estándar ofrece funciones para el escalado manual. Se pueden utilizar como máximo 20 instancias para el escalado horizontal de la aplicación.

Escalado o reducción vertical de PaaS

El escalado y la reducción verticales de los servicios alojados en la aplicación son bastante sencillos. Los elementos del menú de Azure App Services se deben escalar verticalmente, lo que abre una nueva hoja con todos los planes y sus SKU enumerados. Al elegir un plan y un SKU se escala o se reduce verticalmente el servicio.



The screenshot shows the Azure App Service Pricing page. On the left, there's a sidebar with navigation links: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, SETTINGS (with Apps, File system storage, Networking, Scale up (App Service plan) selected, Scale out (App Service plan), Properties, Locks, Automation script), OBSERVE, and Resource explorer. The main area displays three columns of pricing plans:

P1 Premium	P2 Premium	P3 Premium
1 Core	2 Core	4 Core
1.75 GB RAM	3.5 GB RAM	7 GB RAM
250 GB Storage	250 GB Storage	250 GB Storage
Custom domains / SSL SNI Incl & IP SSL Support	Custom domains / SSL SNI Incl & IP SSL Support	Custom domains / SSL SNI Incl & IP SSL Support
Up to 20 instance(s) * Subject to availability	Up to 20 instance(s) * Subject to availability	Up to 20 instance(s) * Subject to availability
20 slots Web app staging	20 slots Web app staging	20 slots Web app staging
50 times daily Backup	50 times daily Backup	50 times daily Backup
Traffic Manager Geo availability	Traffic Manager Geo availability	Traffic Manager Geo availability
14,752.68 INR/MONTH (ESTIMATED)	29,505.37 INR/MONTH (ESTIMATED)	59,010.73 INR/MONTH (ESTIMATED)

S1 Standard	S2 Standard	S3 Standard
1 Core	2 Core	4 Core
1.75 GB RAM	3.5 GB RAM	7 GB RAM
50 GB Storage	50 GB Storage	50 GB Storage
Custom domains / SSL SNI Incl & IP SSL Support	Custom domains / SSL SNI Incl & IP SSL Support	Custom domains / SSL SNI Incl & IP SSL Support
Up to 10 instance(s) Auto scale	Up to 10 instance(s) Auto scale	Up to 10 instance(s) Auto scale
Daily Backup	Daily Backup	Daily Backup
5 slots Web app staging	5 slots Web app staging	5 slots Web app staging
Traffic Manager Geo availability	Traffic Manager Geo availability	Traffic Manager Geo availability

At the bottom center is a blue "Select" button.

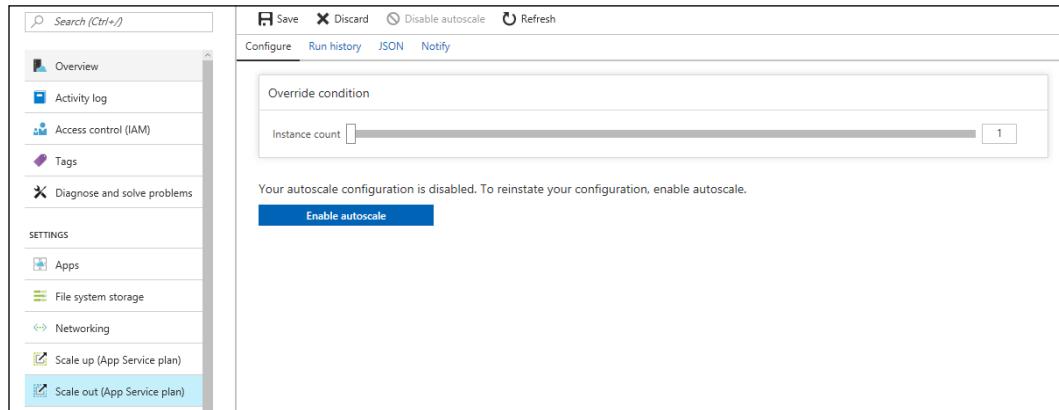
Escalado o reducción horizontal de PaaS

El escalado y la reducción horizontales de los servicios alojados en la aplicación también son bastante sencillos. Los elementos del menú de Azure App Services se deben escalar horizontalmente, lo que abre una nueva hoja con todas las opciones de configuración del escalado.

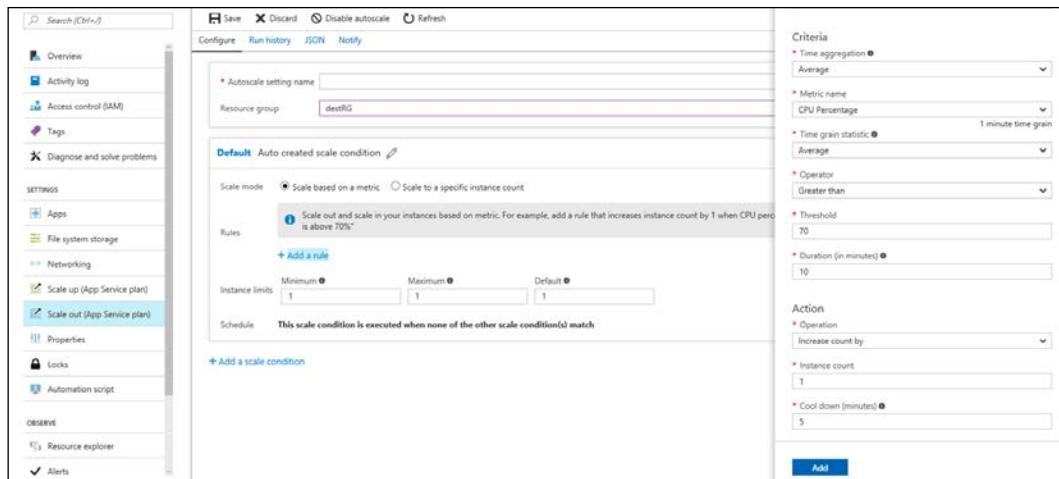
Implementación de la escalabilidad

De forma predeterminada, el escalado automático está deshabilitado para los planes premium y estándar. Puede activarse utilizando el elemento de menú **Escalar horizontalmente** haciendo clic en el botón **Habilitar escalado automático**.

El escalado manual no necesita configuración, pero el escalado automático contribuye a la configuración con la ayuda de las siguientes propiedades.



- **Modo de escalado:** basado en algunas métricas, como el uso de la CPU o la memoria, o solo en la escala para especificar el número de instancias.
- **Cuándo escalar:** pueden agregarse varias reglas que determinen cuándo realizar el escalado y la reducción verticales. Cada regla puede determinar criterios como el consumo de CPU o memoria, si aumentar o disminuir las instancias, cuántas instancias aumentar o reducir al mismo tiempo. Debe configurarse al menos una regla para el escalado horizontal y otra para la reducción horizontal. La definición de un umbral ayuda a definir los límites superior e inferior que, si se traspasan, deben activar el escalado automático, ya sea para aumentar o para disminuir el número de instancias.
- **Cómo escalar:** especifica cuántas instancias crear o desaprovisionar en cada escalado o reducción horizontal.



Escalabilidad IaaS

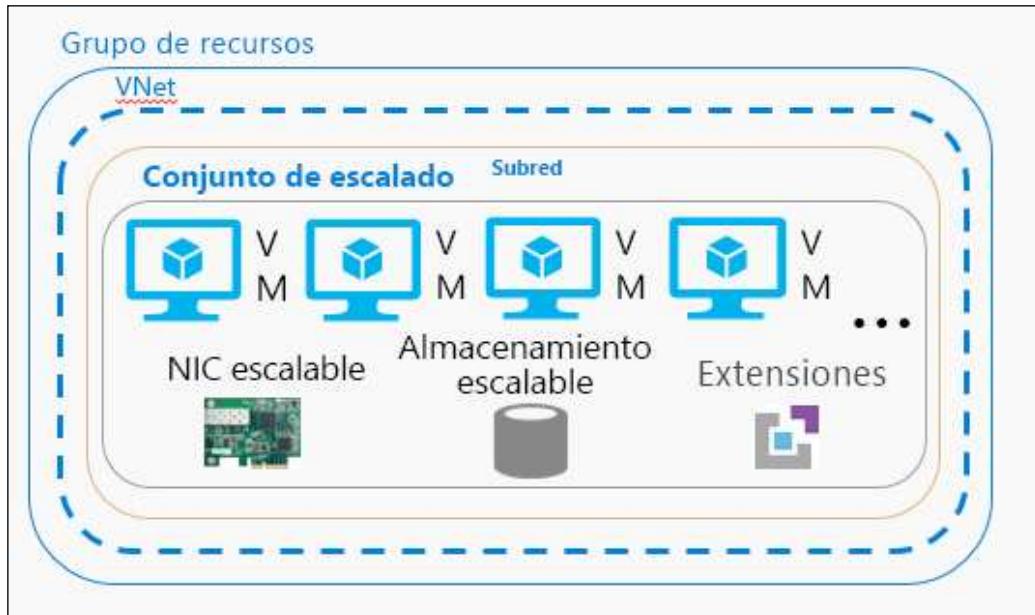
Hay usuarios que quieren tener un control total sobre la infraestructura base, la plataforma y la aplicación. Prefieren consumir soluciones de IaaS que soluciones de PaaS. En el caso de dichos consumidores, cuando crean máquinas virtuales también son responsables del ajuste del tamaño de la capacidad y del escalado. No existe la configuración “out of the box” para el escalado manual o automático de las máquinas virtuales. Estos consumidores tendrán que escribir sus propios scripts de automatización, desencadenadores y reglas para lograr el escalado automático. El uso de máquinas virtuales incluye la responsabilidad de mantenerlas. La aplicación de revisiones y las actualizaciones de las máquinas virtuales son responsabilidad de los propietarios. Los arquitectos deben pensar tanto en el mantenimiento planificado como en el no planificado. Debe pensarse bien cómo se deben aplicar las revisiones, el orden, su agrupamiento y otros factores de estas máquinas virtuales para asegurarse de que no se pone en riesgo la escalabilidad y la disponibilidad de una aplicación. Para paliar estos problemas, Azure proporciona conjuntos de escalado de máquinas virtuales como una solución.

Conjuntos de escalado de máquinas virtuales

Los conjuntos de escalado de máquinas virtuales (VMSS) son un recurso informático de Azure que puedes utilizar para implementar y administrar un conjunto de máquinas virtuales idénticas. Con todas las VM configuradas de la misma manera, los conjuntos de escalado están diseñados para admitir el escalado automático real y no hay que realizar ningún aprovisionamiento previo de máquinas virtuales. Esto ayuda al aprovisionamiento de varias máquinas virtuales idénticas conectadas entre sí mediante una red virtual y una subred.

Implementación de la escalabilidad

VMSS crea un conjunto que se puede crear, configurar y administrar como una unidad. Todas las máquinas virtuales forman parte de esta unidad y cualquier cambio se aplica a la unidad, que a su vez lo aplica a las máquinas virtuales mediante un algoritmo predeterminado.



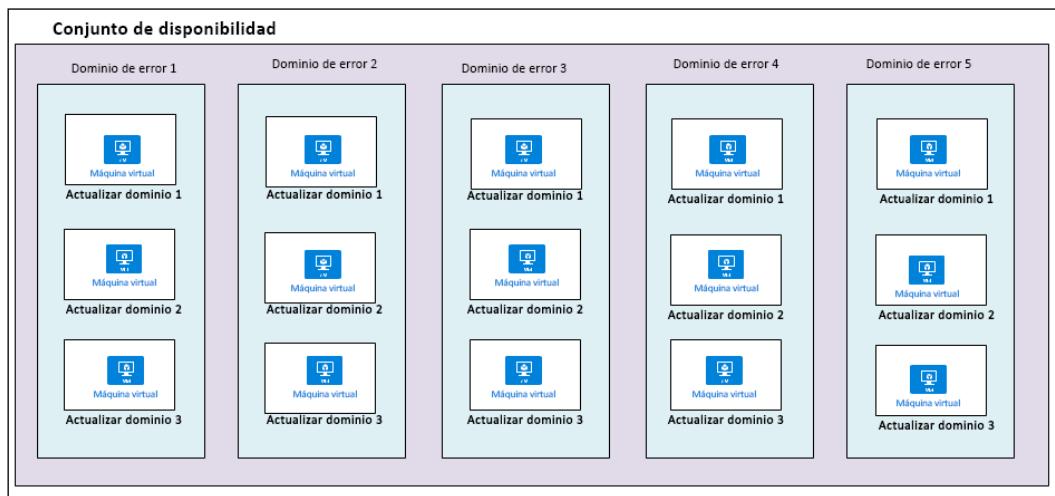
Esto permite que estas máquinas virtuales se equilibren mediante el equilibrador de carga o las puertas de enlace de aplicaciones de Azure. Todas las máquinas virtuales pueden tener un sistema operativo Windows o Linux. Pueden ejecutar scripts automatizados con una extensión PowerShell y pueden administrarse de forma centralizada mediante la configuración del estado deseado. Pueden controlarse como una unidad, y también individualmente con Log Analytics.

VMSS puede aprovisionarse desde el portal de Azure, la interfaz de línea de comandos de Azure, las plantillas de Azure Resource Manager, la API REST y los cmdlets de PowerShell. Es posible invocar la API REST y la CLI de Azure desde cualquier plataforma, entorno, sistema operativo e idioma.

Ya muchos servicios de Azure usan los VMSS como su arquitectura subyacente. Los principales de entre ellos son Azure Batch, Azure Service Fabric y servicios de contenedores de Azure. Los servicios del contenedores de Azure a su vez aprovisionan Kubernetes y DC/OS en estos conjuntos de escalado de máquinas virtuales.

Arquitectura de VMSS

VMSS permite la creación de hasta 1000 máquinas virtuales en un conjunto de escalado si se utiliza una imagen de plataforma y 100 máquinas virtuales si se usa una imagen personalizada. Si el número de máquinas virtuales es inferior a 100 en un conjunto de escalado, se colocan en un único conjunto de disponibilidad; sin embargo, si son más de 100, se crean varios conjuntos de disponibilidad, denominados grupos de selección de ubicación, y las máquinas virtuales se distribuyen entre estos conjuntos de disponibilidad. Sabemos por el último capítulo que las máquinas virtuales de un conjunto de disponibilidad se colocan en dominios de error y actualización diferentes. Los conjuntos de disponibilidad relacionados con VMSS tienen cinco dominios de error y actualización predeterminados. VMSS proporciona un modelo que contiene información de los metadatos de todo el conjunto. Cambiar este modelo y aplicar los cambios afecta a todas las instancias de máquinas virtuales. Esta información incluye las instancias máxima y mínima de máquinas virtuales, las SKU y la versión del sistema operativo, el número actual de máquinas virtuales, el dominio de error y de actualización y mucho más.



Escalado de VMSS

El escalado hace referencia al aumento o la disminución en recursos de procesos y de almacenamiento. VMSS es un recurso rico en funciones que hace que el escalado resulte sencillo y eficaz. Ofrece un escalado automático que ayuda a escalar o reducir verticalmente basándose en eventos y datos externos, como el uso de la CPU y la memoria.

Escalado horizontal frente a vertical

El escalado puede ser horizontal, vertical o ambos. El escalado o reducción horizontal es como se denomina el scaling out e in, mientras que el escalado y la reducción vertical hacen referencia al scaling up y down.

Capacidad

VMSS tiene una propiedad de capacidad que determina el número de máquinas virtuales en un sistema de escalado. VMSS puede implementarse con cero como valor de esta propiedad. No creará ni una sola máquina virtual; sin embargo, si aprovisionas VMSS proporcionando un número para la propiedad de capacidad, se crea ese número de máquinas virtuales.

Escalado automático

El escalado automático de máquinas virtuales en VMSS se refiere a la adición o eliminación de instancias de máquinas virtuales basándose en los entornos configurados para satisfacer las demandas de rendimiento y escalabilidad de una aplicación. En general, en ausencia de VMSS, esto se logra utilizando scripts de automatización y runbooks.

VMSS contribuye a este proceso de automatización con la ayuda de la configuración. En lugar de escribir scripts, VMSS puede configurarse para el escalado y la reducción automáticas verticales.

El escalado automático consta de varios componentes integrados para lograr su objetivo final. El escalado automático supervisa continuamente las máquinas virtuales y recoge datos de telemetría de estas. Almacena estos datos, los combina y los evalúa conforme a un conjunto de reglas para determinar si debe activar el escalado automático. El desencadenador podría ser un escalado o una reducción horizontal. También puede ser un escalado o una reducción vertical.

El escalado automático utiliza registros de diagnóstico para la recopilación de datos de telemetría de las máquinas virtuales. Estos registros se almacenan en las cuentas de almacenamiento como métricas de diagnóstico. El escalado automático también utiliza el servicio de supervisión de información que lee estas métricas, las combina y las almacena en su propia cuenta de almacenamiento.

Los trabajos de escalado automático en segundo plano se ejecutan continuamente para leer los datos de almacenamiento de información, evaluarlos basándose en todas las reglas configuradas para el escalado automático y ejecutar el proceso de escalado automático si alguna de las reglas o la combinación de reglas da resultados positivos. Las reglas pueden tener en cuenta las métricas de la máquina virtual invitada así como del servidor host.

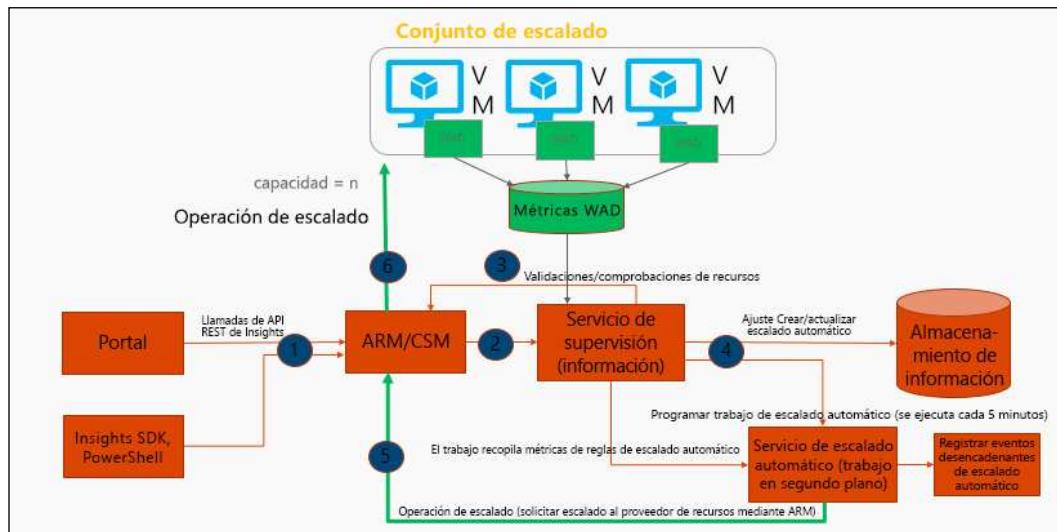
Las reglas se definen mediante las siguientes propiedades. Las descripciones de estas propiedades están disponibles en <https://docs.microsoft.com/azure/virtual-machine-scale-sets/virtual-machine-scale-sets-autoscale-overview>.

Regla	Descripción
metricName	Este valor es el mismo que el contador de rendimiento que se ha definido en la variable <code>wadperfcounter</code> para la extensión del diagnóstico. En el ejemplo anterior, se utiliza el contador de número de subprocessos.
metricResourceUri	Este valor es el identificador de recursos de VMSS. Este identificador contiene el nombre del grupo de recursos, el nombre del proveedor de los recursos y el nombre del conjunto de escalado que se va a escalar.
timeGrain	Este valor es la granularidad de las métricas que se recogen. En el ejemplo anterior, los datos se recogen en un intervalo de un minuto. Este valor se utiliza con <code>timeWindow</code> .
statistic	Este valor determina cómo se combinan las métricas para dar cabida a la acción de escalado automático. Los valores posibles son: media, mín y máx.
timeWindow	Este valor es el intervalo de tiempo en el que se recogen los datos de la instancia. Debe estar entre 5 minutos y 12 horas.
timeAggregation	Este valor determina cómo se deben combinar los datos que se recogen con el tiempo. El valor predeterminado es una media. Los valores posibles son: media, mínimo, máximo, último, total y recuento.
operator	Este valor es el operador que se utiliza para comparar los datos de las métricas y el umbral. Los valores posibles son: Igual, No igual, Mayor que, Mayor o igual que, Menor que y Menor o igual que.
threshold	Este valor es el valor que desencadena la acción de escalado. Asegúrate de proporcionar una diferencia suficiente entre los valores umbral para las acciones de escalado horizontal y reducción horizontal. Si estableces los mismos valores para ambas acciones, el sistema prevé el cambio constante, lo que impide que implemente una acción de escalado. Por ejemplo, fijar ambos subprocessos en 600 en el ejemplo anterior no funciona.
direction	Este valor determina la acción que se toma cuando se alcanza el valor umbral. Los valores posibles son aumento o disminución.

Implementación de la escalabilidad

Regla	Descripción
type	Este valor es el tipo de acción que se debería producir y debe establecerse en ChangeCount.
value	Este valor es el número de máquinas virtuales que se añaden o se eliminan del conjunto de escalado. Este valor debe ser 1 o superior.
cooldown	Este valor es la cantidad de tiempo que hay que esperar desde la última acción de escalado antes de que produzca la siguiente acción. Este valor debe estar entre un minuto y una semana.

La arquitectura de escalado automático se muestra en el siguiente diagrama:



El escalado automático se puede configurar para escenarios que son más complejos que las métricas generales disponibles desde los entornos. Por ejemplo, el escalado podría basarse en cualquiera de los siguientes eventos:

- Escalado en un día específico
- Escalado con programación recurrente, como los fines de semana
- Escalado diferente en días laborables y fines de semana
- Escalado durante los días de fiesta, es decir, uno de los eventos
- Escalado en múltiples métricas de recursos

Estas pueden configurarse con la propiedad de programación de recursos de información que sirven para registrar las reglas.

Los arquitectos deben garantizar que al menos dos acciones, el escalado horizontal y la reducción horizontal, se configuren conjuntamente. El escalado o la reducción horizontales no ayudarán a lograr los beneficios de escalado proporcionados por VMSS.

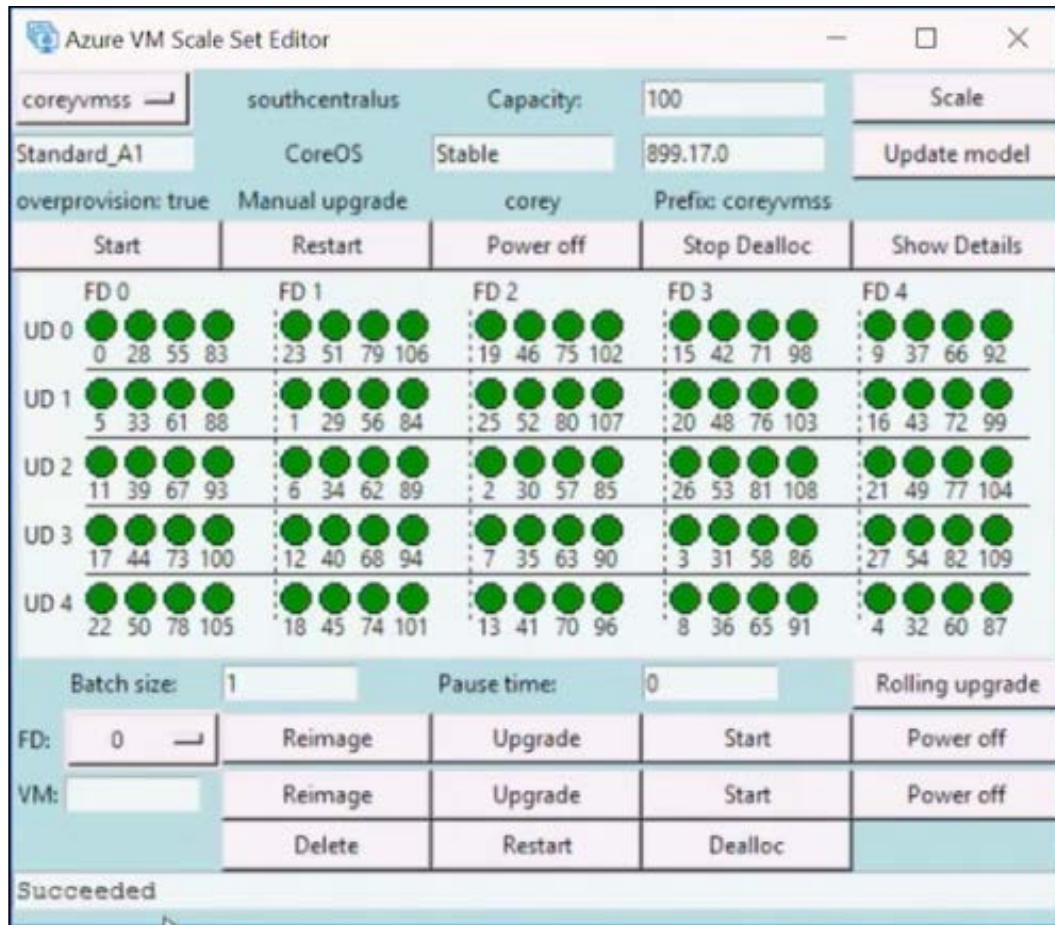
Actualizaciones

Después de implementar VMSS y las aplicaciones, ambos deben mantenerse de forma activa. El mantenimiento planificado debe realizarse periódicamente para asegurar que tanto el entorno como la aplicación están actualizados con los últimos bits y que el entorno está vigente desde el punto de vista de la seguridad y la resiliencia.

Las actualizaciones pueden estar asociadas con las aplicaciones, la instancia de máquina virtual invitada o con la propia imagen. Las actualizaciones pueden ser bastante complejas, porque deben producirse sin afectar a la disponibilidad, la escalabilidad y el rendimiento de los entornos y las aplicaciones. Para asegurar que las actualizaciones pueden tener lugar en una instancia cada vez utilizando métodos de actualización gradual, es importante que VMSS apoye y proporcione las capacidades para estos escenarios avanzados.

Implementación de la escalabilidad

Existe una utilidad proporcionada por el equipo de Azure para administrar las actualizaciones de VMSS. Es una utilidad basada en Python que se puede descargar desde <https://github.com/gbowerman/vmssdashboard>. Hace llamadas de API REST a Azure para administrar los conjuntos de escalado. Esta utilidad puede utilizarse para comenzar, detener, actualizar y restablecer la imagen inicial de las máquinas virtuales en un dominio de error o grupo de máquinas virtuales.



Actualizaciones de la aplicación

Las actualizaciones de la aplicación en VMSS no deben ejecutarse manualmente. Deben ejecutarse como parte de la administración de versiones y la cartera de productos en desarrollo con la automatización. Por otra parte, la actualización debe producirse en una instancia de la aplicación cada vez, sin afectar a la disponibilidad y la escalabilidad globales de la aplicación. Las herramientas de administración de la configuración, como la configuración del estado deseado, deben implementarse para administrar las actualizaciones de la aplicación. El servidor de extracción DSC se puede configurar con la última versión de bits, que deben aplicarse de forma gradual en cada instancia.

Actualizaciones de máquinas invitadas

Las actualizaciones de la máquina virtual son responsabilidad del administrador. Azure no es responsable de aplicar las revisiones en las máquinas virtuales invitadas. Las actualizaciones de las máquinas invitadas están en el modo preview y los usuarios deben controlar la aplicación de las revisiones manualmente o utilizando la automatización personalizada, como runbooks y scripts. Sin embargo, las revisiones graduales están en preview y se pueden configurar en la plantilla de ARM con la política de actualización, como se muestra aquí:

```
"upgradePolicy": {  
    "mode": "Rolling",  
    "automaticOSUpgrade": "true" or "false",  
    "rollingUpgradePolicy": {  
        "batchInstancePercent": 20,  
        "maxUnhealthyUpgradedInstanceCount": 0,  
        "pauseTimeBetweenBatches": "PT0S"  
    }  
}ssssssssssssss
```

Actualizaciones de imagen

VMSS puede actualizar la versión del SO sin ningún tiempo de inactividad. La actualización del sistema operativo consiste en cambiar la versión o las SKU del sistema operativo o cambiar el URI de una imagen personalizada. La actualización sin tiempo de inactividad significa actualizar las máquinas virtuales de una en una o en grupos (por ejemplo, un dominio de error cada vez) en lugar de todas a la vez. Al hacerlo así, cualquier máquina virtual que no se esté actualizando puede seguir funcionando.

Prácticas de escalado recomendadas

En esta sección, repasaremos algunas de las prácticas recomendadas que deben implementar las aplicaciones para aprovechar las ventajas de la capacidad de escalabilidad proporcionada por VMSS.

Preferencia del escalado horizontal

El escalado horizontal es una solución de escalado mejor que el escalado vertical. El escalado o la reducción vertical significa el redimensionamiento de las instancias de máquinas virtuales. Cuando se cambia el tamaño de una máquina virtual, por lo general debe reiniciarse, lo que tiene sus propias desventajas. En primer lugar, hay un tiempo de inactividad para la máquina. En segundo lugar, si hay usuarios activos conectados a la aplicación en esa instancia, podrían enfrentarse a la falta de disponibilidad de la aplicación o podrían incluso perder transacciones. El escalado horizontal no afecta a las máquinas virtuales existentes. Aprovisiona máquinas más nuevas y las añade al grupo.

Reconstrucción completa frente a instancias latentes

El escalado de nuevas instancias puede adoptar dos enfoques amplios. Crear la nueva instancia desde cero, que significa instalar aplicaciones, configurarlas y probarlas, o por otra parte, puede haber instancias latentes en espera que se pueden iniciar cuando sea necesario gracias a la presión de la escalabilidad en otros servidores.

Configuración adecuada del número máximo y mínimo de instancias

Si se establece un valor de dos para el recuento de instancias mínimo y máximo, y siendo el recuento actual de instancias dos, no puede producirse ninguna acción de escalado. Debe haber un margen suficiente entre los recuentos de instancias máximo y mínimo, que son inclusivos. El escalado automático siempre escala entre estos límites.

Simultaneidad

Las aplicaciones están diseñadas para que la escalabilidad se centre en la simultaneidad. La aplicación debe usar patrones asincrónicos para asegurar que las solicitudes de cliente no esperan indefinidamente para adquirir recursos si los recursos están ocupados atendiendo otras solicitudes. La implementación de patrones asincrónicos en código garantiza que los subprocesos no esperen los recursos y que los sistemas se agoten por todos los subprocesos disponibles. Las aplicaciones deben implementar el concepto de tiempo de espera si se esperan errores intermitentes.

Sin estado

Las aplicaciones y los servicios deben diseñarse para carecer de estado. La escalabilidad puede ser difícil de conseguir en el caso de servicios con estado, mientras que resulta muy fácil escalar servicios sin estado. Con el estado aparece el requisito de componentes adicionales y de implementación, como replicación, repositorio centralizado o descentralizado, mantenimiento y sesiones adhesivas. Todos estos son obstáculos en el camino hacia la escalabilidad. Imagina un estado activo de mantenimiento de servicio en un servidor local. No importa el número de solicitudes en una aplicación general o en un servidor en concreto, las solicitudes posteriores deben ser atendidas por el mismo servidor. Las solicitudes posteriores no pueden ser procesadas por otros servidores. Esto hace que la implantación de la escalabilidad sea un problema.

Almacenamiento en caché y CDN

Las aplicaciones y los servicios deben aprovechar el almacenamiento en caché. El almacenamiento en caché ayuda a eliminar varias llamadas posteriores a cualquiera de las bases de datos del sistema de ficheros. Esto contribuye a la disponibilidad de los recursos y los libera para más solicitudes. La **Red de distribución de contenido (CDN)** es otro mecanismo para almacenar en caché archivos estáticos, como imágenes y bibliotecas JavaScript. Está disponible en los servidores de todo el mundo. También hace que los recursos estén disponibles y libres para las solicitudes de cliente adicionales. Esto hace que aplicaciones sean altamente escalables.

Diseño N+1

El **diseño N + 1** se refiere a la creación de redundancia dentro de la implementación general para cada componente. Significa planificar cierta redundancia, incluso cuando no se requiere. Podría significar máquinas virtuales, almacenamiento y tarjeta de interfaz de red adicionales.

Resumen

La escalabilidad es una preocupación arquitectónica importante y crucial. Casi cada aplicación y cada arquitecto intentan aplicar la escalabilidad junto con la disponibilidad y el rendimiento. Azure es una plataforma madura que entiende la necesidad de escalabilidad de las aplicaciones y ofrece opciones de escalabilidad para PaaS, así como para soluciones de IaaS. Los servicios de aplicación de PaaS pueden configurarse para el escalado automático y las máquinas virtuales pueden implementarse en un conjunto de escalado para aprovechar el escalado. El escalado puede ser escalado/reducción vertical o escalado/reducción horizontal. Similar a la alta disponibilidad, la escalabilidad no es una ocurrencia tardía y debe formar parte del desarrollo del ciclo de vida de las aplicaciones desde la misma fase de la planificación. El escalado puede ser vertical, que supone un aumento el tamaño de las instancias de máquina virtual, o puede ser horizontal, que implica la adición de servidores adicionales al conjunto existente. VMSS proporciona un modelo que contiene información de los metadatos de todo el conjunto. Cambiar este modelo y aplicar los cambios afecta a todas las instancias de máquinas virtuales. Esto permite actualizar, cambiar el tamaño, detener e iniciar máquinas virtuales de forma gradual. Por último, este capítulo ha descrito algunas de las prácticas recomendadas importantes en relación con la escalabilidad.

El siguiente capítulo trata la seguridad, que es la preocupación arquitectónica más importante para las implementaciones en el cloud.

5

Seguridad en el cloud

La seguridad es, sin duda, el requisito no funcional más importante que deben implementar los arquitectos. Las empresas priorizan y brindan una atención extrema para implementar correctamente su estrategia de seguridad. De hecho, la seguridad es una de las principales preocupaciones para casi todas las partes interesadas en el desarrollo, la implementación y la administración de cualquier aplicación. Se vuelve aún más importante cuando la misma aplicación está diseñada para su implementación en el cloud.

Ejecutar aplicaciones y sistemas que están disponibles para su consumo por parte de los usuarios es un aspecto importante para los arquitectos en cualquier aplicación seria. Sin embargo, hay otra característica de la aplicación igualmente importante que es una de las principales prioridades de los arquitectos: la escalabilidad de las aplicaciones. Imagina situaciones en las que se implementan aplicaciones y obtienes un gran rendimiento y disponibilidad con unos pocos usuarios, pero tanto la disponibilidad como el rendimiento se ven afectados a medida que aumenta el número de usuarios. También puede suceder que, aunque la aplicación sea eficaz y esté disponible con un gran número de usuarios, haya un momento concreto de un día o una semana, o se produzcan acontecimientos especiales durante los que el número de usuarios aumente y resulte imposible medir o predecir el número de usuarios. Además de la situación anterior, es posible que hayas aprovisionado el hardware y el ancho de banda para el manejo de los usuarios en estas ocasiones y cuando haya picos; sin embargo, la mayoría de las veces, el hardware adicional no se utiliza y no proporciona ningún retorno de la inversión. Su uso solo se aprovisiona durante algunos festivales u ofertas. Espero que entiendas los problemas que los arquitectos tratan de resolver. Todos estos problemas están relacionados con el ajuste de tamaño de la capacidad y la escalabilidad de una aplicación. El enfoque de este capítulo es comprender la escalabilidad como una preocupación arquitectónica y detallar las características proporcionadas por Azure para abordar estas inquietudes.

En este capítulo, abordaremos los siguientes temas:

- Principios de seguridad
- Seguridad de Azure
- Cumplimiento y certificación
- Directorio:
 - Identidad--autenticación
 - Autorización
 - oAuth y open connect
- Seguridad de IaaS:
 - Seguridad de red
 - Seguridad de Compute
 - Seguridad de almacenamiento
- Seguridad de PaaS:
 - SQL Server
 - Almacén de claves
- Servicios de seguridad
- Azure Security Center
- OMS: supervisión y auditoría
- Centro de confianza de Azure

Seguridad

Proteger una aplicación significa no permitir que entidades desconocidas y no autorizadas accedan a ella. También significa que la comunicación con la aplicación es segura y no se ve reducida.

Eso incluye lo siguiente:

- **Autenticación:** la autenticación hace referencia al establecimiento de la identidad de un usuario y al hecho de garantizar que dicha identidad puede acceder a la aplicación o el servicio. En Azure, la autenticación se lleva a cabo mediante open connect, conocido también como **ConnectID**.
- **Autorización:** la autorización se refiere a permitir y establecer permisos que una identidad puede realizar dentro de la aplicación o servicio. La autorización se lleva a cabo en Azure mediante la tecnología oAuth.

- **Confidencialidad:** se refiere a que la comunicación entre el usuario y la aplicación es segura. El intercambio de carga entre entidades está cifrado, por lo que solo tiene sentido para el remitente y el receptor, y en ningún otro caso. La confidencialidad de los mensajes se realiza mediante cifrado simétrico y asimétrico. Los certificados se utilizan para implementar criptografía: cifrado y descifrado de mensajes.
- **Integridad:** la integridad garantiza que el intercambio de carga y de mensajes entre el remitente y el receptor no se reduzca. El receptor recibe el mismo mensaje que envió el remitente. Las firmas digitales y los hashes son el mecanismo de implementación para verificar la integridad de los mensajes entrantes.

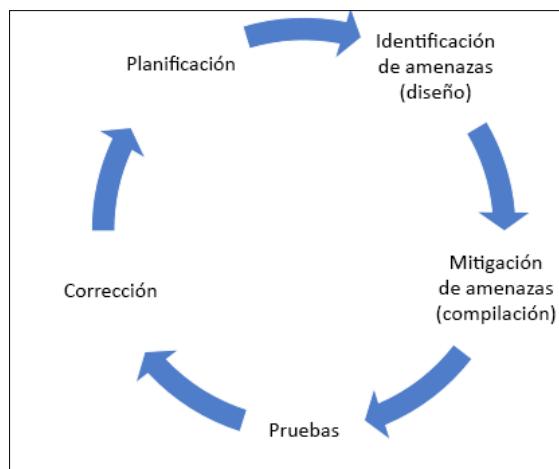
La seguridad es una asociación entre el proveedor de servicios y el consumidor del servicio. Ambas partes tienen diferentes niveles de control en pilas de implementación completas y cada una debe implementar las prácticas recomendadas de seguridad para garantizar la identificación y mitigación de todas las amenazas. En el *capítulo 1, Introducción* hemos visto que el cloud ofrece tres paradigmas: IaaS, PaaS y SaaS, cada uno con diferentes niveles de control colaborativo sobre la pila de implementación. Cada parte debe implementar prácticas de seguridad para los componentes que se encuentran bajo su control y ámbito. La falta de implementación de seguridad en cualquier capa de la pila o por parte de cualquiera haría que la implementación completa y la aplicación fueran vulnerables a los ataques.

Ciclo de vida de seguridad

Normalmente, la seguridad se considera un requisito no funcional para una solución. Sin embargo, actualmente y debido al aumento de los ciberataques, se considera un requisito funcional.

Para sus aplicaciones, cada organización sigue algún tipo de administración del ciclo de vida de la aplicación. Cuando la seguridad se trata como un requisito funcional, debe seguir el mismo proceso de desarrollo de aplicaciones. La seguridad no debe ser una ocurrencia tardía, sino que debe ser parte de la aplicación desde el principio. En la fase de planificación general de una aplicación, también se debe planificar la seguridad. En función de la naturaleza de la aplicación, se deben identificar diferentes tipos y categorías de amenazas y, en función de estas identificaciones, se deben documentar en términos de enfoque y alcance para mitigarlas. Se debe realizar un ejercicio de simulación de amenazas para ilustrar la amenaza de la que puede ser objeto cada componente. Esto llevará al diseño de estándares y políticas de seguridad para la aplicación. Normalmente, esta es la fase de diseño de seguridad. La fase siguiente se denomina **Mitigación** de amenazas o fase de **compilación**. En esta fase, la implementación de la seguridad en términos de código y configuración se ejecuta para mitigar los riesgos y las amenazas de seguridad.

Un sistema no puede ser seguro hasta que se prueba. Deben realizarse pruebas de penetración apropiadas y otras pruebas de seguridad para identificar posibles mitigaciones que no están implementadas, se han olvidado o se han pasado por alto. Los errores de las pruebas se corrigen y el ciclo continúa hasta la vida de la aplicación. Por motivos de seguridad, debe seguirse este proceso de administración del ciclo de vida de la aplicación.

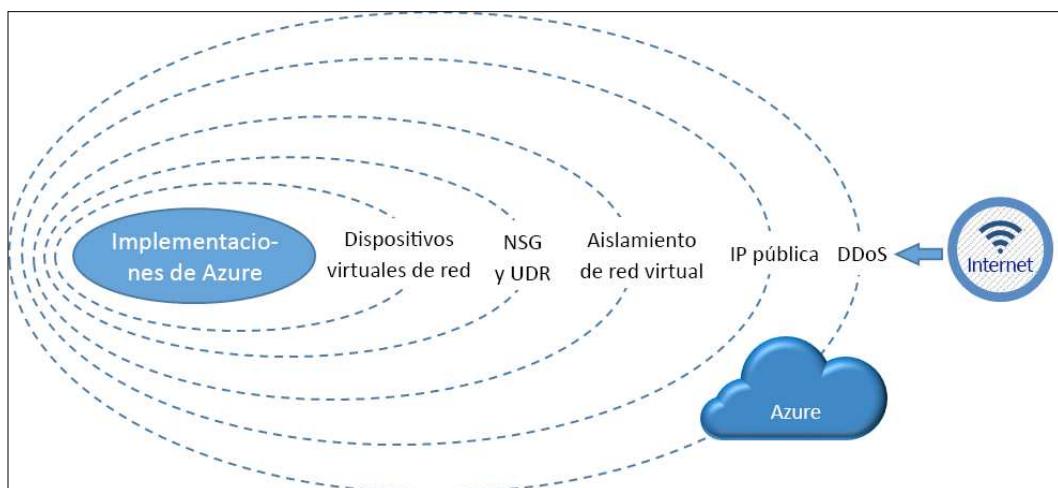


La simulación de amenazas, la identificación, la mitigación, las pruebas y la corrección son procesos iterativos que continúan incluso cuando una aplicación o un servicio están operativos. Debería haber una supervisión activa de entornos y aplicaciones completos para identificar proactivamente las amenazas y mitigarlas. La supervisión también debe habilitar alertas y registros de auditoría para ayudar en el diagnóstico reactivo, la solución de problemas y la eliminación de amenazas y vulnerabilidades.

El ciclo de vida de la seguridad de cualquier aplicación comienza con la fase de planificación, que finalmente conduce a la fase de diseño. En la fase de **diseño**, la arquitectura de la aplicación se descompone en componentes granulares con comunicación discreta y límites de hosting. Las amenazas se identifican en función de su interacción con otros componentes dentro y fuera de los límites de hosting. Las amenazas identificadas se mitigan mediante la implementación de características de seguridad adecuadas dentro de la arquitectura general y se prueban para identificar si aún existe dicha vulnerabilidad. Una vez que la aplicación se implementa en producción y se pone en funcionamiento, se supervisa para detectar cualquier brecha de seguridad y vulnerabilidad y se lleva a cabo una corrección proactiva o de reacción.

Seguridad de Azure

Azure proporciona todos sus servicios a través de centros de datos que se encuentran en varias regiones. Estos centros de datos están interconectados dentro de las regiones y a través de las regiones. Azure entiende que aloja aplicaciones, servicios y datos de misión crítica e importantes para sus consumidores. Debe garantizar que la seguridad es de suma importancia para sus centros de datos y regiones. Los consumidores implementan aplicaciones en el cloud basándose en esta confianza de que Azure protegerá sus aplicaciones y datos de vulnerabilidades y vulneraciones. Los consumidores no se trasladarán al cloud si esta confianza se rompe y, por lo tanto, Azure implementa la seguridad en todas las capas, desde el perímetro del centro de datos físico hasta los componentes lógicos del software. Todas las capas están protegidas, e incluso el equipo del centro de datos de Azure no tiene acceso a ellas.



La seguridad es de suma importancia tanto para Microsoft como para Azure. Azure es una plataforma de cloud, una plataforma alojada por Microsoft. Microsoft garantiza que la confianza se construye con sus consumidores y lo hace al garantizar que la implementación, las soluciones y los datos de sus consumidores sean completamente seguros, física y virtualmente. La gente no utilizará ninguna plataforma de cloud si no es segura física y digitalmente. Para garantizar que los consumidores confíen en Azure, se planifica, documenta, audita y supervisa cada actividad en el desarrollo de Azure desde una perspectiva de seguridad. Los centros de datos físicos de Azure están protegidos contra cualquier intrusión y acceso no autorizado. De hecho, incluso el personal de Microsoft y el equipo de operaciones no tienen acceso a la solución y los datos del consumidor.

- **Acceso seguro de usuario:** solo el consumidor puede acceder a sus datos, solución e implementación. Incluso las personas del centro de datos de Azure no tienen acceso a ningún artefacto del consumidor. Los consumidores pueden permitir el acceso a más personas, según considere el consumidor.

- **Cifrado en reposo:** Azure cifra todos sus datos de administración para que nadie pueda leerlos. También proporciona esta funcionalidad a sus consumidores, así como a aquellos que pueden cifrar sus datos en reposo.
- **Cifrado en tránsito:** Azure cifra todos los datos que fluyen desde su red. También garantiza que la red troncal de su red esté protegida contra cualquier acceso no autorizado.
- **Supervisión activa y auditoría:** Azure supervisa activamente todos sus centros de datos de forma continua. Identifica activamente cualquier vulneración, amenaza y riesgo y los mitiga.

Azure cumple con las normas de cumplimiento locales, internacionales y sectoriales específicas de cada país. De nuevo, se pueden encontrar en <https://www.microsoft.com/trustcenter/compliance/complianceofferings>.

Seguridad de IaaS

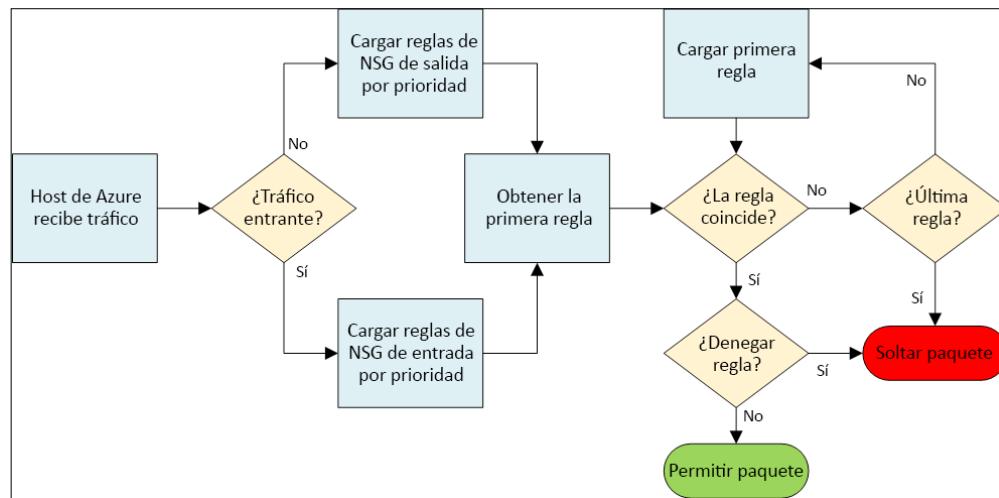
Azure es una plataforma madura para la implementación de soluciones de IaaS. Hay muchos usuarios de Azure que desean un control completo de sus implementaciones y generalmente usan IaaS para sus soluciones. Es importante que estas implementaciones y soluciones sean seguras de forma predeterminada y en cuanto al diseño. Azure proporciona funciones de seguridad Enriquecidas para proteger soluciones de IaaS. En esta sección, se tratarán algunas de las características principales.

Grupos de seguridad de red

El mínimo indispensable de implementación de IaaS consiste en máquinas y redes virtuales. Las máquinas virtuales pueden estar expuestas a Internet al aplicar una IP pública a su interfaz de red o pueden estar disponibles solo para recursos internos. A su vez, los recursos internos podrían estar expuestos a Internet. En cualquier caso, es necesario proteger las máquinas virtuales para que ni siquiera las alcancen las solicitudes no autorizadas. Las máquinas virtuales deben protegerse mediante instalaciones que puedan filtrar solicitudes en la propia red, en lugar de que lleguen a la máquina virtual y que actúen en ellas. Esto es como crear una cerca alrededor de las máquinas virtuales. Esta cerca puede permitir o denegar solicitudes según su protocolo, IP de origen, IP de destino, puerto de origen y puerto de destino. Esta característica se implementa con el recurso Azure Network Security Groups (NSG). NSG se compone de reglas que se evalúan para las solicitudes de entrada y de salida. Según la ejecución y evaluación de estas reglas, se determina si se debe permitir o denegar el acceso de las solicitudes.

Los NSG son flexibles y pueden aplicarse a una subred de red virtual o interfaces de red individuales. Cuando se aplica a una subred, las reglas de seguridad se aplican a cualquier recurso, es decir, a las máquinas virtuales o equilibradores de carga en esta subred, mientras que la aplicación a una interfaz de red afecta las solicitudes solo para esa interfaz de red. También es posible aplicar NSG a la subred de red y las interfaces de red simultáneamente. Normalmente, este diseño debe usarse para aplicar reglas de seguridad comunes en el nivel de subred de red y reglas de seguridad diferentes únicas en el nivel de interfaz de red. Es útil en el diseño de reglas y aplicaciones de seguridad modular.

El flujo para evaluar NSG se muestra en la siguiente figura:



Azure proporciona algunas reglas predeterminadas “out of the box”. Son muy importantes y son útiles cuando las implementaciones desean usar reglas relacionadas con la solicitud desde/hacia Internet, redes virtuales y equilibradores de carga. En general, las direcciones IP son cambiantes para estos recursos y al utilizar estas reglas se proporciona abstracción para usar estas direcciones IP directamente.

Diseño del grupo de seguridad de red

En el diseño, el primer paso es determinar los requisitos de seguridad del recurso. Es necesario responder las preguntas siguientes:

- ¿Se puede acceder al recurso solo desde Internet?
- ¿Se puede acceder al recurso tanto desde los recursos internos como desde Internet?
- ¿Se puede acceder al recurso solo desde el recurso interno?
- Determinar el equilibrador de carga de los recursos, las puertas de enlace y las máquinas virtuales que se han utilizado
- Configuración de una red virtual y su subred

En función de las respuestas, se debe crear un diseño válido de NSG.

Idealmente, debería haber varias subredes de red para cada carga de trabajo y tipo de recurso. No se recomienda implementar equilibradores de carga y máquinas virtuales en la misma subred.

En función de los requisitos, es necesario determinar las reglas que son comunes para diferentes subredes y cargas de trabajo de máquinas virtuales. Por ejemplo, para una implementación de SharePoint, la aplicación front-end y los SQL Servers se implementan en subredes separadas. Se deben determinar las reglas de cada subred.

Después de identificar las reglas comunes de nivel de subred, se deben identificar las reglas de los recursos individuales y estas deben aplicarse en el nivel de la interfaz de red.

Es importante comprender que si una regla permite una solicitud entrante en un puerto, ese puerto también se puede utilizar para solicitudes salientes sin ninguna configuración.

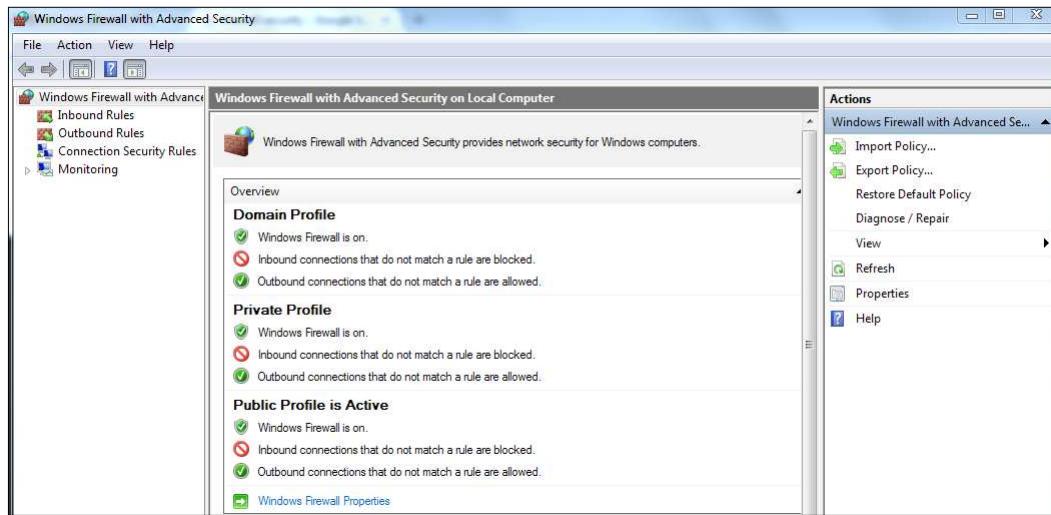
En la medida de lo posible, si se puede acceder a los recursos desde Internet, se deben crear reglas con rangos de IP y puertos específicos.

Se deben realizar pruebas funcionales y de seguridad de forma cuidadosa para garantizar que se abran y cierren las reglas de NSG válidas y óptimas.

Firewalls

NSG proporciona perímetros de seguridad externos para las solicitudes. Sin embargo, esto no significa que las máquinas virtuales no deban implementar medidas de seguridad adicionales. Siempre es mejor implementar la seguridad tanto interna como externamente. Ya sea en Linux o Windows, las máquinas virtuales proporcionan un mecanismo para filtrar solicitudes en el nivel del sistema operativo. Esto se conoce como firewall tanto en Windows como en Linux.

Es recomendable implementar firewalls para sistemas operativos. Ayudan en la generación de un muro de seguridad virtual para permitir solo aquellas solicitudes que se consideran de confianza. Se deniega el acceso de cualquier solicitud que no sea de confianza. Incluso hay dispositivos de firewall físicos, pero en el software de cloud se utilizan firewalls de sistema operativo.



Los firewalls ayudan a filtrar paquetes de red, identificar puertos entrantes y direcciones IP. A partir de la información de estos paquetes, evalúa las reglas y determina si debe permitir el acceso o denegarlo.

Diseño de firewall

Como buena práctica, los firewalls deben evaluarse para sistemas operativos individuales. Cada máquina virtual tiene una responsabilidad distinta en la implementación y la solución generales. Las reglas para estas responsabilidades individuales deben identificarse y los firewalls deben abrirse y cerrarse en consecuencia.

Al evaluar las reglas del firewall, es importante tener en cuenta las reglas del grupo de seguridad de red para la subred y el nivel de la interfaz de red individual. Si no se realiza correctamente, es posible que las reglas se denieguen en el nivel de NSG, pero que se dejen abiertas en el nivel de firewall y viceversa. Si se permite una solicitud en el nivel de regla de NSG y se deniega en el nivel de firewall, la aplicación no funcionará como se esperaba, mientras que los riesgos de seguridad aumentan si la solicitud se deniega en el nivel de regla de NSG y se permite en el nivel de firewall.

Un firewall ayuda a generar múltiples redes aisladas por sus reglas de seguridad.

Se deben realizar pruebas funcionales y de seguridad de forma cuidadosa para garantizar que se abran y cierren las reglas de firewall válidas y óptimas.

Reducción de la superficie de ataque

Los NSG y el firewall ayudan a administrar solicitudes autorizadas en el entorno. Sin embargo, el entorno no debe exponerse abiertamente a ataques de seguridad. La superficie del sistema debe estar habilitada de manera óptima para poder lograr su funcionalidad, pero debe estar lo suficientemente deshabilitada para que los atacantes no puedan encontrar lagunas y áreas de acceso que se abran sin ningún uso previsto o abiertas sin estar protegidas adecuadamente. La seguridad debe reforzarse adecuadamente para dificultar el acceso al sistema de cualquier atacante.

Algunas de las áreas que deben configurarse incluyen:

- Eliminar todos los usuarios y grupos innecesarios del sistema operativo.
- Identificar la pertenencia al grupo de todos los usuarios.
- Implementar políticas de grupo utilizando servicios de directorio.
- Bloquear la ejecución del script a menos que esté firmado por autoridades de confianza.
- Registrar y auditar todas las actividades.
- Instalar software antivirus y antimalware, programar análisis y actualizar las definiciones con frecuencia.
- Deshabilitar o apagar servicios que no son necesarios.
- Bloquear el sistema de archivos solo para el acceso autorizado.
- Bloquear cambios en el registro.
- El firewall debe configurarse de acuerdo con las necesidades de la solución.
- La ejecución del script de PowerShell se debe establecer en restringida o en RemoteSigned.
- Protección mejorada habilitada a través de Internet Explorer.
- Restringir la capacidad de crear nuevos usuarios y grupos.
- Quitar el acceso a Internet e implementar servidores de acceso para RDP.
- No permitir el inicio de sesión en servidores utilizando RDP a través de Internet. En su lugar, utilice una VPN sitio a sitio o una VPN punto a sitio o una ruta rápida a RDP en máquinas remotas desde la red.
- Implementar regularmente todas las actualizaciones de seguridad.
- Ejecutar la herramienta del administrador de cumplimiento de seguridad en el entorno e implementar todas sus recomendaciones.
- Supervisar el entorno activamente utilizando el centro de seguridad y el conjunto de administración de operaciones.
- Implementar dispositivos virtuales de red para enrutar el tráfico al proxy interno y al proxy inverso.
- Toda la información confidencial, como la configuración, las cadenas de conexión, las credenciales, etc., debe estar cifrada.

Implementación de servidores de acceso

Es una buena idea quitar el acceso a Internet de las máquinas virtuales. También es una buena práctica eliminar la accesibilidad de los servicios de escritorio remoto de Internet, pero luego, ¿cómo se puede acceder a las máquinas virtuales? Una buena manera es permitir que solo lo hagan los recursos internos para RDP en máquinas virtuales que utilicen opciones de VPN de Azure. Sin embargo, también hay otra forma: mediante servidores de acceso.

Los servidores de acceso son servidores que se implementan en una red perimetral (DMZ). Esto significa que se encuentran en una red diferente y no en la red que aloja las soluciones y aplicaciones principales. En su lugar, están en una red o subred separada. El objetivo principal del servidor de acceso es aceptar solicitudes RDP de los usuarios y ayudarles a iniciar sesión. Desde este servidor de acceso, los usuarios pueden navegar a otras máquinas virtuales usando RDP. Tiene acceso a dos o más redes, una que tiene conectividad con el mundo exterior y otra interna con la solución. El servidor de acceso implementa todas las restricciones de seguridad y proporciona un cliente seguro para conectarse a otros servidores. Normalmente, el acceso a correos electrónicos e Internet está deshabilitado en los servidores de acceso.

Hay un ejemplo de implementación de un servidor de acceso con conjuntos de escalado de máquinas virtuales (VMSS) en <https://azure.microsoft.com/resources/templates/201-vmss-windows-jumpbox/> usando plantillas del administrador de recursos de Azure.

Seguridad de PaaS

Azure proporciona varios servicios PaaS, cada uno con sus propias características de seguridad. En general, se puede acceder a los servicios PaaS con credenciales, certificados y tokens. Los servicios PaaS permiten la generación de tokens de acceso de seguridad de corta duración. Las aplicaciones cliente pueden enviar este token de acceso de seguridad para representar a usuarios de confianza. En esta sección, cubriremos algunos de los servicios PaaS más importantes que se utilizan en prácticamente todas las soluciones.

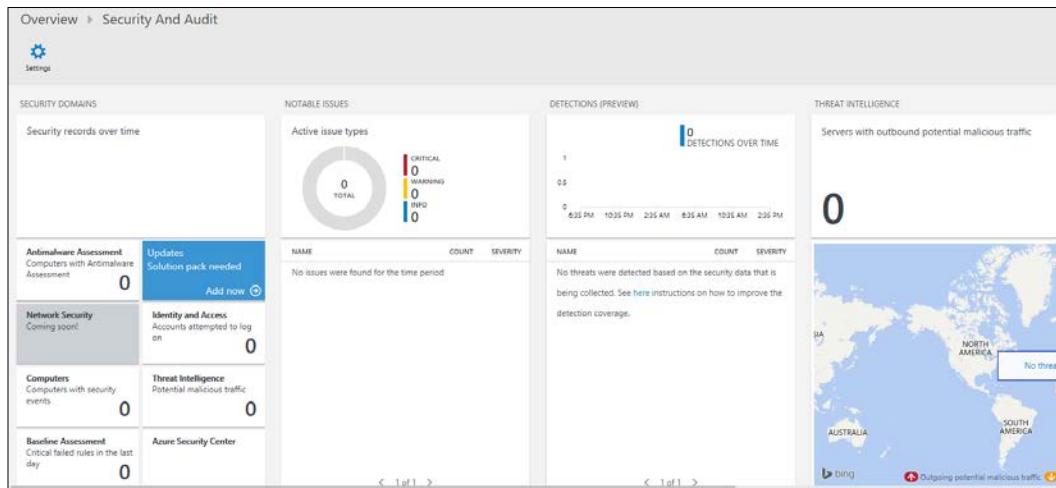
Operations Management Suite (OMS)

Microsoft Management Suite, también conocida como Log Analytics, es una nueva plataforma para administrar implementaciones de cloud, centros de datos on-premise y soluciones híbridas.

OMS proporciona múltiples soluciones modulares, una funcionalidad específica que ayuda a implementar una función. Por ejemplo, las soluciones de seguridad y auditoría ayudan a determinar una visión completa de la seguridad para la implementación de las organizaciones. Del mismo modo, hay muchas más soluciones, como la automatización, el seguimiento de cambios, etc., que deben implementarse desde una perspectiva de seguridad.

La seguridad y auditoría de OMS proporcionan información en cuatro categorías:

- **Dominios de seguridad:** proporciona funcionalidad para ver registros de seguridad, evaluar el malware, evaluar las actualizaciones, la seguridad de la red y la información de identidad y acceso, equipos con eventos de seguridad y proporciona acceso al panel de Azure Security Center.
- **Evaluación antimalware:** ayuda a identificar los servidores que no están protegidos contra malware y que tienen problemas de seguridad. Ayuda a proporcionar una exposición general a posibles problemas de seguridad y su criticidad. Los usuarios pueden tomar medidas proactivas basadas en estas recomendaciones. La subcategoría Azure Security Center proporciona información recopilada por Azure Security Center.
- **Problemas notables:** ayuda a identificar rápidamente los problemas activos y su gravedad.
- **Detecciones:** esta categoría está en el modo preview. Permite identificar patrones de ataque mediante la visualización de alertas de seguridad.
- **Inteligencia sobre amenazas:** ayuda a identificar los patrones de ataque al visualizar la cantidad total de servidores con tráfico IP malintencionado saliente, el tipo de amenaza malintencionada y un mapa que muestra de dónde provienen estas IP.



Almacenamiento

La cuenta de almacenamiento es un componente importante en la arquitectura general de la solución. Las cuentas de almacenamiento pueden almacenar información importante, como datos de identificación personal de usuario, transacciones comerciales, datos, etc. Es de suma importancia que las cuentas de almacenamiento sean seguras y que solo permitan el acceso restringido a usuarios autorizados. Los datos almacenados están cifrados y se transmiten mediante canales seguros. No solo el almacenamiento, sino también los usuarios y las aplicaciones cliente que consumen la cuenta de almacenamiento y sus datos deben desempeñar un papel crucial en la seguridad general de los datos. También deben mantener los datos cifrados en todo momento. Esto también incluye credenciales y cadenas de conexión que se conectan a almacenes de datos.

Azure proporciona control RBAC sobre quién puede administrar las cuentas de almacenamiento de Azure. Estos permisos RBAC están permitidos para los usuarios y grupos de Azure **Active Directory (AD)**. Sin embargo, cuando se crea una aplicación que se implementará en Azure, tendrá usuarios y consumidores que no están disponibles en Azure AD. Para permitir que los usuarios accedan a la cuenta de almacenamiento, el almacenamiento de Azure proporciona claves de acceso de almacenamiento. Hay dos tipos de claves de acceso en el nivel de cuenta de almacenamiento: principal y secundario. Los usuarios que poseen estas claves pueden conectarse a la cuenta de almacenamiento. Estas claves de acceso de almacenamiento se utilizan en la autenticación para acceder a la cuenta de almacenamiento. Las aplicaciones pueden acceder a las cuentas de almacenamiento usando claves principales o secundarias. Se proporcionan dos claves, de modo que si la clave principal está incluida, las aplicaciones se pueden actualizar para usar la clave secundaria, mientras que la clave principal se regenera. Esto ayuda a minimizar el tiempo de inactividad de la aplicación. Además, proporciona y mejora la seguridad al eliminar la clave incluida sin afectar a las aplicaciones.

The screenshot shows the 'Default keys' section of the Azure Storage Account Keys page. It includes a table with columns for NAME, KEY, and CONNECTION STRING. Key1 and Key2 are listed, each with a copy icon and a regenerate icon.

NAME	KEY	CONNECTION STRING
key1	[REDACTED]	DefaultEndpointsProtocol=https;AccountName= [REDACTED]
key2	[REDACTED]	DefaultEndpointsProtocol=https;AccountName= [REDACTED]

El almacenamiento de Azure proporciona cuatro servicios: blob, archivos, colas y tablas en una cuenta. Cada uno de estos servicios también proporciona infraestructura para protegerse a sí mismos usando tokens de acceso seguro. Una **firma de acceso compartido (SAS)** es un URI que otorga derechos de acceso restringido a los servicios de almacenamiento de Azure: blob, archivos, colas y tablas. Estas firmas de acceso compartido pueden compartirse con clientes en los que no se debe confiar con la clave de la cuenta de almacenamiento completa, solo para restringir el acceso a ciertos recursos de la cuenta de almacenamiento. Al distribuir un URI de firma de acceso compartido a estos clientes, se otorga acceso a los recursos por un período determinado.

La firma de acceso compartido existe tanto en la cuenta de almacenamiento como en los niveles individuales de blob, archivo, tabla y cola. La firma de nivel de cuenta de almacenamiento es más potente y tiene derechos para permitir y denegar permisos en el nivel de servicio individual. También se puede utilizar en lugar de niveles de servicio de recursos individuales.

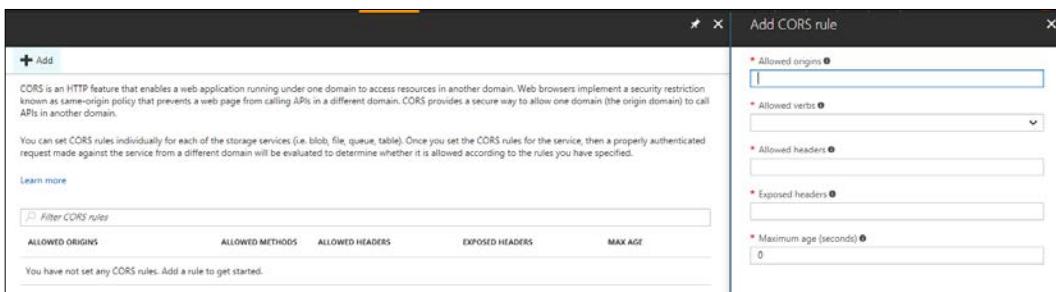
The screenshot shows the configuration options for generating a SAS token:

- Allowed services:** Blob, File, Queue, Table (all checked)
- Allowed resource types:** Service, Container, Object (all checked)
- Allowed permissions:** Read, Write, Delete, List, Add, Create, Update, Process (all checked)
- Start and expiry date/time:**
 - Start: 2017-07-30, 10:39:25 AM
 - End: 2017-07-30, 6:39:25 PM
 - Timezone: UTC - Coordinated Universal Time
- Allowed IP addresses:** for example, 168.1.5.65 or 168.1.5.65-168.1.5.70
- Allowed protocols:** HTTPS only (selected)
- Signing key:** key1
- Generate SAS** button

Se prefiere generar y compartir firmas de acceso compartido a compartir las claves de la cuenta de almacenamiento. Las firmas de acceso compartido brindan acceso granular a los recursos y pueden combinarse también entre sí. Incluyen leer, escribir, eliminar, enumerar, agregar, crear, actualizar y procesar. Además, incluso el acceso a los recursos se puede determinar al generar firmas de acceso compartido. Podría ser para blobs, tablas, colas y archivos individualmente o una combinación de ellos. Las claves de la cuenta de almacenamiento son para toda la cuenta y no se pueden limitar para servicios individuales. Tampoco se puede limitar desde la perspectiva de los permisos. Es mucho más fácil crear y revocar firmas de acceso compartido en comparación con las claves de acceso de almacenamiento. Las firmas de acceso compartido se pueden crear para su uso durante un determinado período de tiempo después del cual se vuelven no válidas automáticamente.

Se debe tener en cuenta que si las claves de la cuenta de almacenamiento se regeneran, la firma de acceso compartido basada en ellas se invalidará y se debe crear y compartir con los clientes las nuevas firmas de acceso compartido.

El robo de cookies, la inserción de scripts y los ataques por denegación de servicio son medios comunes utilizados por los atacantes para interrumpir o alterar un entorno y robar datos. Los navegadores y el protocolo HTTP implementan un mecanismo incorporado que garantiza que no se pueden realizar estas actividades malintencionadas. En términos generales, cualquier cosa entre dominios no está permitida tanto por HTTP como por navegadores. Un script que se ejecuta en un dominio no puede solicitar recursos de otro dominio. Sin embargo, hay casos de uso válidos en los que se deben permitir estas solicitudes. El protocolo HTTP implementa **Cross Origin Resource Sharing (CORS)**. Con la ayuda de CORS, es posible acceder a los recursos a través de dominios y hacer que funcionen. El almacenamiento de Azure ayuda a configurar las reglas de CORS para los recursos de blobs, archivos, colas y tablas. El almacenamiento de Azure permite la creación de reglas que se evalúan para cada solicitud autenticada. Si se cumplen las reglas, se permite que la solicitud acceda al recurso.



Los datos no solo deben protegerse mientras están en tránsito, sino que también deben protegerse mientras están en reposo. Si los datos en reposo no están cifrados, cualquier persona con acceso a la unidad física en el centro de datos puede leer los datos. Aunque la posibilidad es casi insignificante, los consumidores deben seguir cifrando sus datos. El cifrado del servicio de almacenamiento también ayuda a proteger los datos en reposo. Este servicio funciona de forma transparente y se inserta a sí mismo sin que los usuarios lo sepan. Cifra los datos cuando se guardan en una cuenta de almacenamiento y se descifra automáticamente cuando se lee. Todo este proceso ocurre sin que los usuarios realicen ninguna actividad adicional.

Las claves de la cuenta de Azure deben cambiar periódicamente. Esto garantizará que un atacante no pueda vulnerar el acceso a las cuentas de almacenamiento.

También es una buena idea volver a generar las claves; sin embargo, esto debe evaluarse con respecto a su uso en aplicaciones existentes. Si se rompe la aplicación existente, es necesario priorizar estas aplicaciones para la administración de cambios y los cambios deben aplicarse gradualmente.

En la medida de lo posible, los tokens SAS de nivel de servicio individual con un marco de tiempo limitado deben generarse y proporcionarse a los usuarios que deben acceder a ellos. Se deben evaluar los permisos y proporcionar los permisos óptimos.

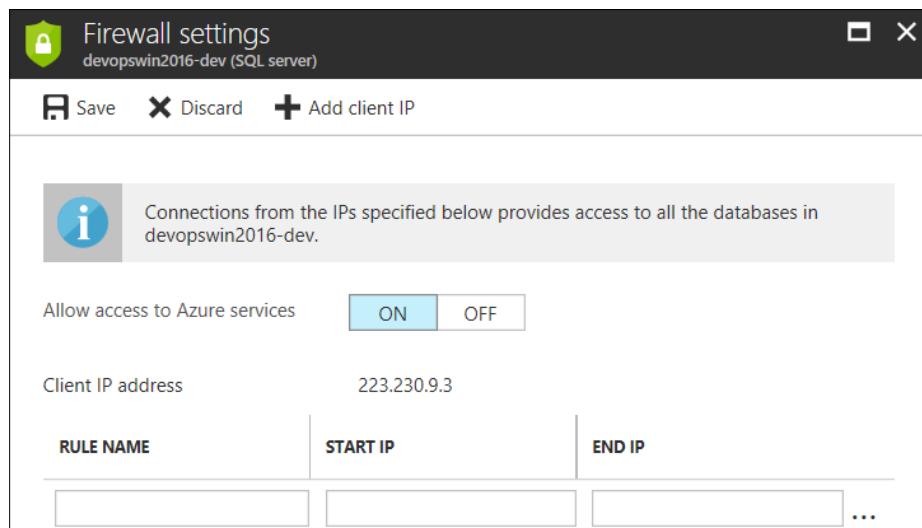
Las claves SAS y las claves de la cuenta de almacenamiento deben almacenarse en Azure Key Vault. Proporciona almacenamiento de seguridad y acceso a ellas. Las aplicaciones pueden leer estas claves en tiempo de ejecución desde el almacén de claves en lugar de almacenarlas en archivos de configuración.

Azure SQL

SQL Server ayuda a almacenar datos relacionales en Azure. Es un servicio SaaS que proporciona una plataforma de alta disponibilidad, escalable, centrada en el rendimiento y segura para almacenar datos. Es accesible desde cualquier lugar, cualquier lenguaje de programación y plataforma. Los clientes necesitan una cadena de conexión que comprenda información de servidor, base de datos y seguridad para conectarse a ella.

SQL server proporciona configuración de firewall que no permite el acceso a nadie de forma predeterminada. Los rangos y las direcciones IP deben estar en la lista blanca para acceder a SQL Server. Solo se deben incluir en la lista blanca las direcciones IP en las que los arquitectos confían por pertenecer al consumidor o asociados. Hay implementaciones en Azure para las que hay muchas direcciones IP o las direcciones IP no son conocidas. Por ejemplo, las aplicaciones implementadas en funciones de Azure o aplicaciones lógicas. Para que dichas aplicaciones accedan a Azure SQL, Azure SQL proporciona listas blancas de todas las direcciones IP a los servicios de Azure en todas las suscripciones.

Se debe tener en cuenta que la configuración del firewall está en el nivel del servidor y no en la base de datos. Significa que cualquier cambio aquí afecta a todas las bases de datos de un servidor.

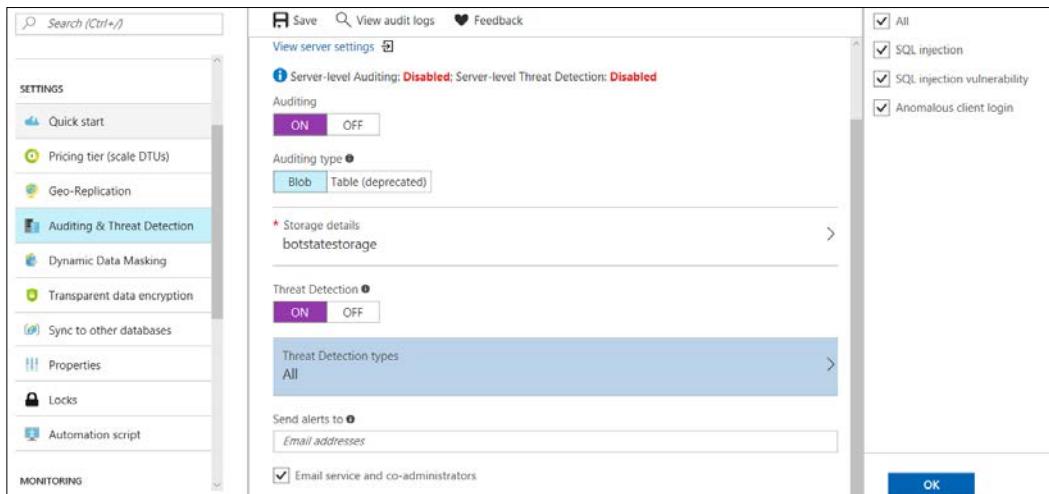


Azure SQL también proporciona seguridad mejorada al cifrar los datos en reposo. Esto garantiza que nadie, incluidos los administradores del centro de datos de Azure, pueda ver los datos almacenados en SQL Server. La tecnología utilizada por SQL Server para cifrar datos en reposo se conoce como **Cifrado de datos transparente (TDE)**. No se requieren cambios en el nivel de la aplicación para implementar TDE. SQL Server cifra y descifra los datos de forma transparente cuando el usuario los guarda y los lee. Esta característica está disponible en el nivel de base de datos.

SQL Server también proporciona **Enmascaramiento dinámico de datos (DDM)**, que es especialmente útil para enmascarar ciertos tipos de datos, como tarjetas de crédito o datos de identificación personal del usuario. El enmascaramiento no es lo mismo que el cifrado. El enmascaramiento no cifra los datos, solo enmascara, hecho que garantiza que los datos no se encuentren en un formato legible.

Los usuarios deben enmascarar y cifrar información confidencial en SQL Server de Azure.

SQL Server también proporciona el servicio de detección de amenazas y auditoría para todos los servidores. Hay servicios avanzados de recopilación de datos y de inteligencia que se ejecutan en la parte superior de estas bases de datos para encontrar amenazas y vulnerabilidades y alerta a los usuarios. Azure actualiza los registros de auditoría en las cuentas de almacenamiento y los administradores pueden verlos en acción. Los subprocesos, como la inserción de SQL y los inicios de sesión anónimos de los clientes, pueden generar alertas sobre las que se puede informar a los administradores por correo electrónico.



Seguridad en el cloud

Los datos se pueden enmascarar en Azure SQL. Esto ayuda a almacenar los datos en un formato que no tiene sentido fácilmente.

The screenshot shows the 'Masking rules' section of the Azure portal. It includes fields for 'MASK NAME' and 'MASK FUNCTION'. A note says 'You haven't created any masking rules.' Below this is a section for 'SQL users excluded from masking (administrators are always excluded)'. On the right, there's a configuration panel for creating a new mask, with fields for 'Mask name', 'Select what to mask' (Schema, Table, Column), and 'Select how to mask' (Masking field format).

Azure SQL también proporciona cifrado de datos transparente para cifrar datos en reposo.

The screenshot shows the 'SETTINGS' page in the Azure portal. Under 'Dynamic Data Masking', the 'Transparent data encryption' option is selected. On the right, there's a summary of encryption status: 'Encrypts your databases, backups enable TDE, go to each database.', a 'Learn more' link, a toggle switch for 'Data encryption' set to 'ON', and an 'Encryption status' section showing 'Encrypted' with a green checkmark.

Azure Key Vaults

Proteger recursos mediante contraseñas, claves, credenciales, certificados e identificadores únicos es un elemento importante para cualquier entorno y aplicación. Son elementos importantes desde el punto de vista de la seguridad. Deben estar protegidos y para garantizar que estos recursos permanezcan seguros y no se comprendan, es un pilar importante de la arquitectura de seguridad. La administración y las operaciones que mantienen seguros los secretos y las claves al tiempo que están disponibles cuando es necesario es un aspecto importante que se debe tener en cuenta.

Por lo general, estos secretos se utilizan en todo el sitio: dentro del código fuente, el archivo de configuración, pedazos de papel y en otros formatos digitales. Para superar estos desafíos y almacenar todos los secretos de manera uniforme en un almacenamiento seguro centralizado, se deben crear Azure Key Vaults.

Azure Key Vault está bien integrado con otros servicios de Azure. Por ejemplo, se puede realizar fácilmente mediante el uso de un certificado almacenado en Azure Key Vault y su implementación en el almacén de certificados de máquinas virtuales de Azure. Todo tipo de claves, incluidas las claves de almacenamiento, las claves de IoT y eventos y las cadenas de conexión se pueden almacenar como secretos en Azure Key Vault. Se pueden recuperar y utilizar de forma transparente sin que nadie las vea o las almacene temporalmente en cualquier lugar. Las credenciales de SQL Server y otros servicios también se pueden almacenar en Azure Key Vault.

Azure Key Vault funciona por región. Lo que esto significa es que un recurso de Azure Key Vault debe aprovisionarse en la misma región en la que se implementan la aplicación y el servicio. Si una implementación consta de más de una región y necesita servicios Azure Key Vault, se deben aprovisionar múltiples instancias de Azure Key Vault.

Auditoría y supervisión de seguridad

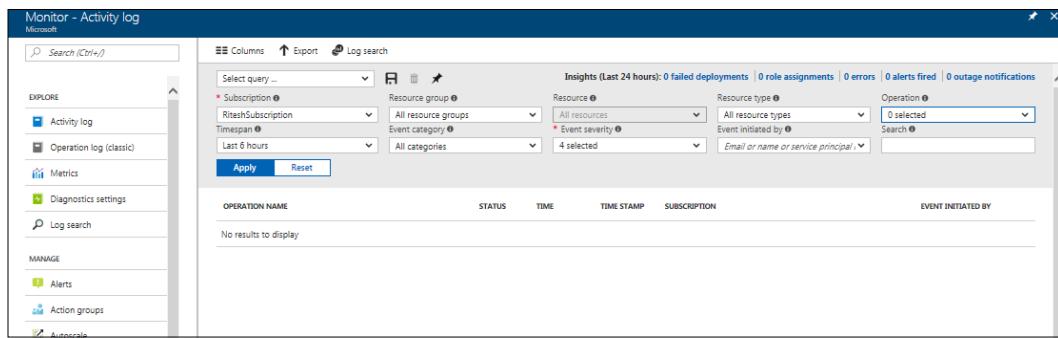
Azure proporciona dos recursos de seguridad importantes para administrar todos los aspectos de seguridad de la suscripción de Azure, los grupos de recursos y los recursos:

- Azure Monitor
- Azure Security Center

Azure Monitor

Azure Monitor es un punto de parada para supervisar recursos de Azure. Proporciona información acerca de los recursos de Azure y su estado. Proporciona una interfaz de consulta enriquecida mediante la cual la información se puede dividir y separar en segmentos utilizando los datos de suscripción, el grupo de recursos, el recurso individual, el tipo de recurso y el nivel de intervalo de tiempo.

Azure Monitor se puede utilizar a través del portal de Azure, PowerShell, CLI y la API de REST.

The screenshot shows the Azure Monitor - Activity log interface. On the left, there's a sidebar with options like 'Activity log', 'Operation log (classic)', 'Metrics', 'Diagnostics settings', 'Log search', 'Alerts', 'Action groups', and 'Autoscale'. The main area has a search bar at the top with placeholder text 'Search (Ctrl+Shift+F)'. Below it are several filter dropdowns: 'Subscription' (set to 'Subscription'), 'Resource group' (set to 'All resource groups'), 'Resource type' (set to 'All resources'), 'Operation' (set to '0 selected'), 'Event category' (set to 'All categories'), 'Event severity' (set to '4 selected'), and 'Last 6 hours'. There are also 'Apply' and 'Reset' buttons. To the right of the filters, a message says 'Insights (Last 24 hours): 0 failed deployments | 0 role assignments | 0 errors | 0 alerts fired | 0 outage notifications'. Below the filters is a table with columns: 'OPERATION NAME', 'STATUS', 'TIME', 'TIME STAMP', 'SUBSCRIPTION', and 'EVENT INITIATED BY'. A note below the table says 'No results to display'.

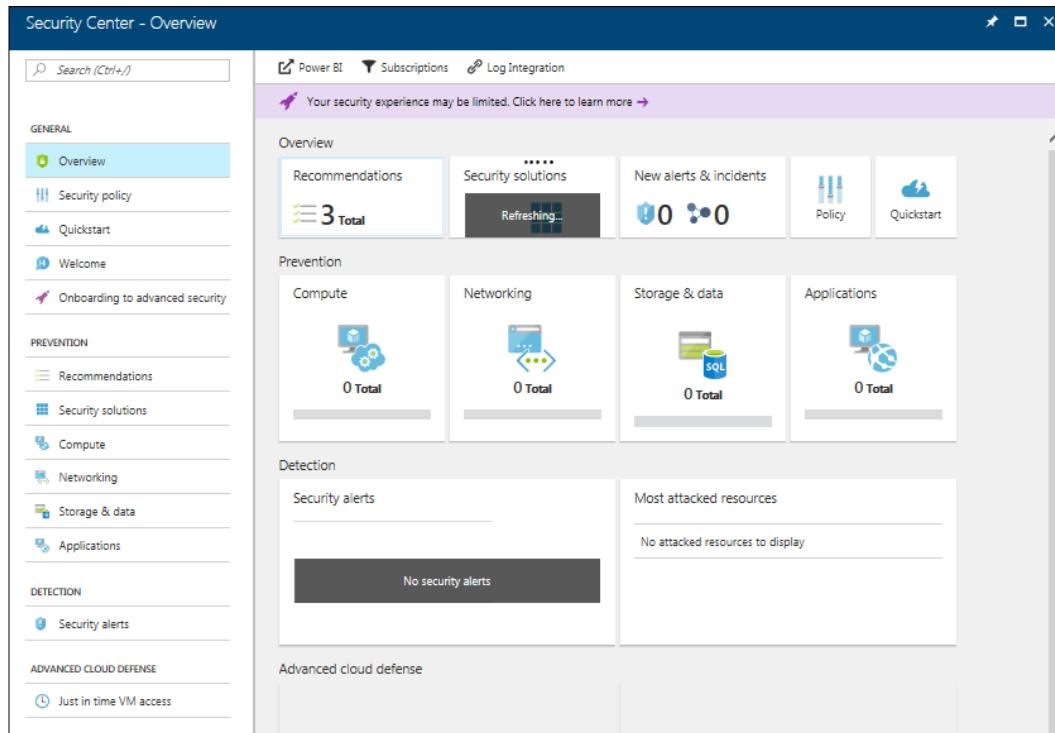
- El registro de **actividades** proporciona todas las operaciones de nivel de administración realizadas en los recursos. Proporciona detalles sobre el tiempo de creación, el creador, el tipo de recurso y el estado.
- El registro de **operaciones (clásico)** proporciona detalles de todas las operaciones realizadas en los recursos en un grupo de recursos y una suscripción.
- Las **métricas** ayudan a obtener información sobre el nivel de rendimiento de los recursos individuales y establecen alertas sobre ellos.
- La **configuración de diagnóstico** ayuda a configurar los registros de efectos al configurar el almacenamiento de Azure para almacenar registros, transmitir los registros en tiempo real a los concentradores de eventos de Azure y enviarlos a Log Analytics (anteriormente conocido como Operational Management Suite).
- La **búsqueda de registros** ayuda a integrar el análisis de registros con el Azure Monitor.

Azure Monitor puede ayudar a identificar los incidentes relacionados con la seguridad y tomar las medidas adecuadas en función de ellos. Es importante que solo las personas autorizadas puedan acceder a Azure Monitor, ya que puede contener información confidencial.

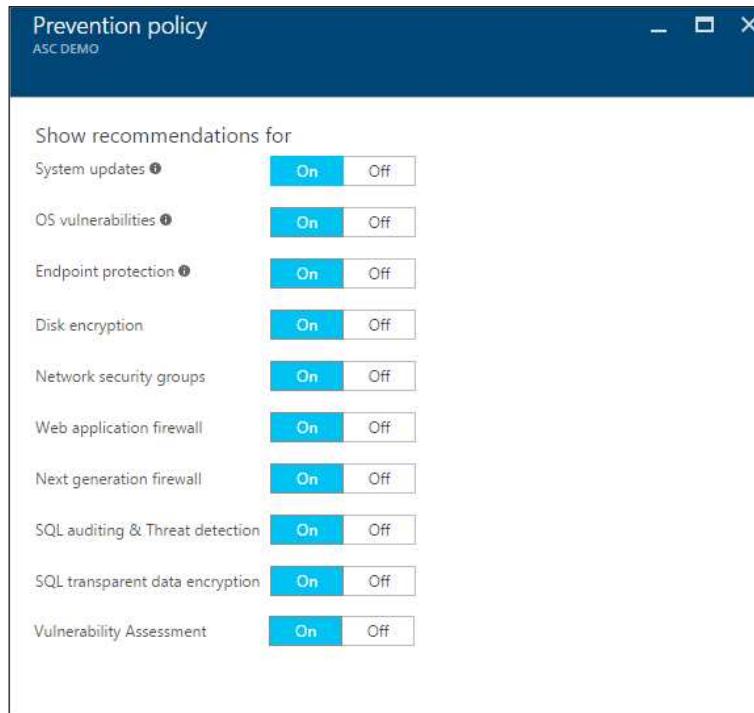
Azure Security Center

Como su nombre indica, Azure Security es un punto de parada para todas las necesidades de seguridad. En general, existen dos actividades relacionadas con la seguridad: la implementación de la seguridad y la supervisión de amenazas y vulneraciones. Centro de seguridad se ha creado principalmente para ayudar en ambas actividades. Azure Security Center permite a los usuarios definir sus políticas de seguridad y hacer que se implementen en los recursos de Azure. En función del estado actual de los recursos de Azure, Azure Security Center brinda recomendaciones de seguridad para fortalecer la solución y los recursos individuales de Azure.

Las recomendaciones incluyen casi todas las prácticas recomendadas de seguridad de recursos de Azure, incluido el cifrado de datos, discos, protección de red, protección de puntos finales, listas de control de acceso, listas blancas de solicitudes entrantes, bloqueo de solicitudes no autorizadas, etc. Los recursos van desde componentes de infraestructura como equilibradores de carga, grupos de seguridad de red y red virtual hasta recursos de PaaS como Azure SQL, almacenamiento, etc.



Azure Security Center es una plataforma enriquecida y puede proporcionar recomendaciones para varios servicios.



Resumen

La seguridad siempre fue un aspecto importante para cualquier implementación y solución. Se ha vuelto mucho más importante y relevante debido a las implementaciones en el cloud. Además, cada vez hay más eventos y amenazas de ataques de ciberseguridad. En tales circunstancias, la seguridad se ha convertido en el punto central de las organizaciones. Independientemente del tipo de implementación y solución, ya sea IaaS, PaaS o SaaS, la seguridad es necesaria en todas ellas. Los centros de datos de Azure son completamente seguros y cuentan con una docena de certificaciones de seguridad internacionales. Son seguros de forma predeterminada. Proporcionan recursos de seguridad de IaaS, como NSG, traducción de direcciones de red, puntos de conexión seguros, certificados, almacenes de claves, almacenamiento y cifrado de máquinas virtuales y características de seguridad de PaaS para recursos individuales de PaaS. La seguridad tiene un ciclo de vida completo y se debe planificar, diseñar, implementar y probar adecuadamente, al igual que cualquier otra funcionalidad de la aplicación.

En el próximo capítulo, algunas de las soluciones y arquitecturas específicas de la tecnología serán el tema central del libro. En el siguiente capítulo hablaremos sobre el **Internet de las cosas (IoT)** en Azure.

6

Diseño de soluciones de IoT

Hasta el momento, hemos tratado las preocupaciones arquitectónicas y sus soluciones en Azure en general. Este capítulo no se basa en arquitectura generalizada. Este capítulo trata sobre una de las tecnologías más revolucionarias de este siglo. En este capítulo se describirá con detalle la arquitectura de IoT en Azure.

En este capítulo se explicarán los siguientes temas:

- Azure e IoT
- Información general de Azure IoT
- Administración de dispositivos
 - Registro de dispositivos
 - Comunicación entre el dispositivo y el centro de IoT
- Escalado de soluciones de IoT
- Alta disponibilidad de soluciones de IoT
- Protocolos de IoT
- Uso de las propiedades de los mensajes para el enrutamiento de mensajes

IoT

IoT es una abreviatura de **Internet of Things (el Internet de las cosas)**. Hay dos palabras clave importantes en esta frase: Internet y cosas. Para entenderlo mejor, vamos a retroceder algunos siglos en la historia para establecer una relación con el surgimiento actual de esta tecnología.

Se inventó en el Internet de los noventa y estuvo disponible de manera general para todo el público aunque su entrada tuvo un impacto menor. Durante este tiempo, casi todo el mundo tenía a tener una presencia en Internet y empezó a crear páginas web estáticas. Estas páginas web estáticas mostraban detalles sobre diversos aspectos. Con el tiempo, el contenido estático se convirtió en dinámico y ahora se generaba sobre la marcha en función del contexto. En casi todos los casos, se necesitaba un explorador para tener acceso a Internet. Había una gran cantidad de exploradores, pero sin ellos el uso de Internet era todo un desafío.

Durante la primera década de este siglo, se produjo un desarrollo interesante: el surgimiento de dispositivos portátiles en la forma de teléfonos móviles y tabletas. Los teléfonos móviles empezaron a ser cada vez más asequibles y a estar disponibles por doquier. Las capacidades de hardware y software de estos dispositivos portátiles mejoraron considerablemente con el tiempo. Tanto es así que la gente empezó a utilizar los exploradores en los dispositivos móviles en lugar de hacerlo en sus equipos de escritorio. Sin embargo, se produjo un cambio sutil que sí resultó evidente: el surgimiento de las aplicaciones móviles. Las aplicaciones móviles se descargaban de alguna tienda y se conectaban a Internet para comunicarse con sistemas backend. Hacia el final de la década pasada, había millones de aplicaciones disponibles con casi cualquier funcionalidad concebible integrada. El sistema backend para estas aplicaciones se construyó en el cloud para que se pudieran escalar con rapidez. Fue la época de conectar aplicaciones y servidores.

Pero, ¿se había alcanzado la cima de la innovación? ¿Cuál sería la próxima evolución que ofrecería una opción mejor para todos en general? Un examen más minucioso desvela que había otro paradigma acaparando protagonismo. Se trataba del IoT. Cuando las aplicaciones se conectaban a un sistema backend, este consistía básicamente de un dispositivo que tenía capacidades de cálculo y almacenamiento y que estaba conectado a Internet para realizar solicitudes. Pero, ¿qué ocurría si conectábamos cada dispositivo conectado a Internet? En lugar de solo móviles y tabletas conectados a Internet, puede haber otros dispositivos con capacidad de conexión a Internet. Estos dispositivos estaban disponibles en mercados selectos, eran caros, no estaban disponibles para el público general y tenían unas capacidades limitadas de hardware y software. Sin embargo, durante la primera parte de la década actual empezó la comercialización de estos dispositivos a gran escala. Estos dispositivos tenían un tamaño cada vez más reducido, contaban con mayor capacidad desde una perspectiva de hardware y software, tenían una mayor potencia de cálculo y almacenamiento, podían conectarse a Internet según diversos protocolos y se podían acoplar a casi cualquier objeto. Estamos en la época de conectar dispositivos a servidores, aplicaciones y otros dispositivos.

Esto condujo a la formulación de la idea y las aplicaciones que podían cambiar el modo en que los sectores estaban operando. Soluciones innovadoras que en el pasado fueron desatendidas ahora empezaban a convertirse en realidad. Ahora estos dispositivos podían acoplarse a cualquier objeto y obtener información para enviarla a un sistema backend que podía asimilar esa información de todos los dispositivos y adoptar una acción proactiva o notificar lo que sucede.

Algunos ejemplos de IoT son los sistemas de seguimiento de los vehículos que controlan todos los parámetros esenciales de un vehículo y envían estos datos a un almacén de datos centralizado para su análisis, servicios urbanos inteligentes (como seguimiento de niveles de contaminación, temperatura, atascos en calles, etc.) y actividades agrícolas relacionadas con la fertilidad del suelo, la humedad, etc.

Arquitectura de IoT

Antes de empezar con Azure y sus características y servicios relacionados con IoT, es importante comprender varios componentes necesarios para crear soluciones de IoT integrales.

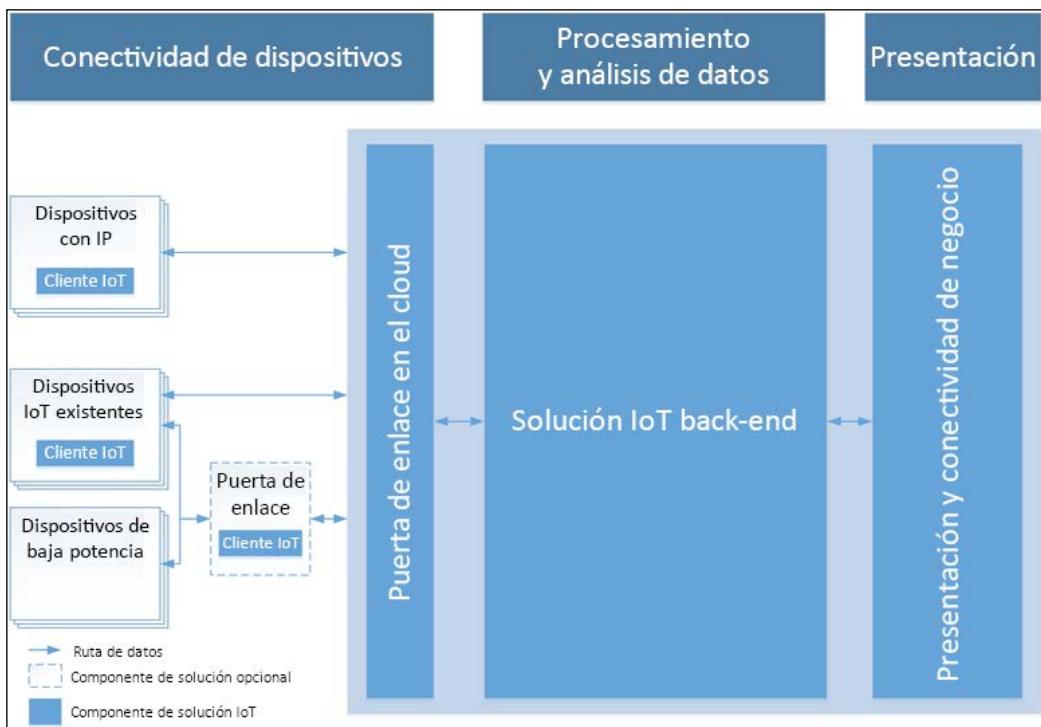
Imagina dispositivos de IoT de todo el mundo enviando millones de mensajes cada segundo a una base de datos centralizada. ¿Por qué se recopilan estos datos? Estos datos se recopilan para extraer información enriquecida acerca de eventos, anomalías, valores atípicos y acontecimientos dentro de esos dispositivos y de los objetos a los que están conectados.

Vamos a describir esto con mayor detalle.

La arquitectura de IoT se puede dividir en fases distintas:

- Conectividad
- Identidad
- Captura
- Ingesta
- Almacenamiento
- Transformación
- Análisis
- Presentación

Las imágenes siguientes muestran una arquitectura basada en IoT genérica. Los dispositivos generan o recopilan datos y los envían a puertas de enlace en el cloud. A su vez, la puerta de enlace en el cloud envía los datos a varios servicios de backend para su procesamiento. Las puertas de enlace en el cloud son un componente opcional. Se recomienda su uso cuando los propios dispositivos no son capaces de enviar solicitudes a servicios de backend por limitaciones de recursos o ausencia de una red fiable. Estas puertas de enlace en el cloud pueden recopilar datos de diversos dispositivos y enviarlos a servicios de backend. A continuación, los datos procesados por los servicios de backend se presentan como conocimientos o paneles analíticos a los usuarios.



Conectividad

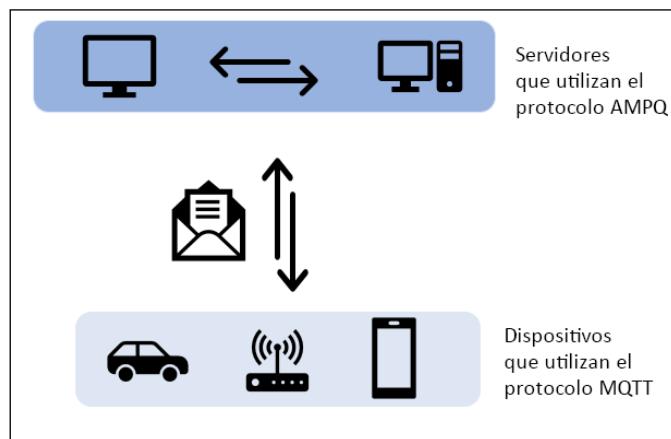
Los dispositivos de IoT necesitan comunicarse. Existen varios tipos de conectividad. Podría ser entre dispositivos de una región, entre dispositivos y una puerta de enlace centralizado, de dispositivos con una plataforma de IoT y mucho más.

En todos los casos mencionados anteriormente, los dispositivos de IoT necesitan la capacidad de conectarse. Esta capacidad podría ser una conectividad a Internet, Bluetooth, infrarrojos o cualquier otro tipo de comunicación con dispositivos cercanos.

Puede que algunos dispositivos no tengan la capacidad de conectarse a Internet. En esos casos, se pueden conectar a través de otros medios a una puerta de enlace que, por otra parte, sí tiene conectividad a Internet.

Los dispositivos de IoT utilizan protocolos para enviar mensajes. Los protocolos más importantes de este tipo son **Advanced Message Queuing Protocol (AMQP)** y **Message Queue Telemetry Transport (MQTT)**.

Los datos de los dispositivos se deben enviar a la infraestructura de TI. El protocolo MQTT es un protocolo de dispositivo a servidor que pueden usar los dispositivos para enviar telemetría y otro tipo de información a los servidores. Una vez el servidor recibe el mensaje a través del protocolo MQTT necesita transportar los mensajes a otros servidores mediante una tecnología fiable basada en mensajes y colas. AMQP es el protocolo preferido para mover mensajes entre servidores en la infraestructura de TI de una manera fiable y predecible.



Los servidores que reciben los mensajes iniciales de los dispositivos de IoT deben enviar los mensajes a otros servidores para su procesamiento, como almacenamiento en registros, análisis y fines de presentación.

Algunos dispositivos no tienen capacidades de conexión a Internet o no admiten los protocolos que entienden los protocolos de IoT. Como se ha mencionado anteriormente, los dispositivos IoT tienen diversas capacidades y, a continuación, se mencionan algunas de las características que afectan a sus capacidades. Para que estos dispositivos funcionen con la plataforma de IoT y el cloud, las puertas de enlace intermedias permiten adaptarlos para enviar información al cloud. Las puertas de enlace ayudan en la incorporación de dispositivos con conectividad y redes que son lentas y no uniformes y dispositivos que utilizan protocolos que no son estándar; además, sus capacidades están limitadas en términos de recursos y potencia.

En estas circunstancias en las que los dispositivos necesitan una infraestructura adicional para participar y conectarse a servicios de backend, se pueden implementar puertas de enlace en el cliente. Estas puertas de enlace reciben mensajes de dispositivos cercanos y los reenvían y transfieren a la infraestructura de TI y la plataforma de IoT para su posterior consumo. Estas puertas de enlace son capaces de realizar una traducción de protocolos de ser necesario.

Identidad

Los dispositivos de IoT se deben registrar en la plataforma de cloud. No todos los dispositivos deben contar con permiso para conectarse a la plataforma de cloud. Los dispositivos se deben registrar y deben recibir la asignación de una identidad. El dispositivo debe enviar la información de identidad mientras se conecta y envía información al cloud. Si el dispositivo no consigue enviar esta información de identidad, la conectividad no se producirá. Más adelante veremos en este capítulo cómo generar una identidad para un dispositivo mediante una aplicación de simulación.

Captura

Los dispositivos de IoT deben poder capturar la información de sí mismos y del ecosistema que los rodea. Por ejemplo, deben tener la capacidad de leer el contenido de humedad en el aire o en el suelo. La información se puede capturar según la frecuencia, que puede llegar a ser de un segundo. Una vez capturada la información, los dispositivos deben poder enviarla a la plataforma de IoT para su procesamiento. Si un dispositivo no tiene la capacidad de conectarse directamente a la plataforma de IoT, se puede conectar a puertas de enlace intermedias y en el cloud para transferir la información capturada. El tamaño de los datos capturados y la frecuencia son los aspectos más importantes del dispositivo. Si el dispositivo cuenta o no con almacenamiento local y almacena temporalmente los datos capturados es otro aspecto importante que debe tenerse en cuenta. El dispositivo puede funcionar en un modo sin conexión si hay disponible almacenamiento local. Incluso los dispositivos móviles actúan, en ocasiones, como dispositivos de IoT conectados a varios instrumentos y cuentan con la capacidad para almacenar datos.

Ingesta

Los datos capturados y generados por los dispositivos se deben enviar a una plataforma de IoT que sea capaz de ingerir y consumir esos datos para extraer de ellos información y conocimientos significativos. El servicio de ingestión es un servicio importante y esencial debido a que su disponibilidad y escalabilidad afectan al rendimiento de los datos entrantes. Si los datos empiezan a obstruirse debido a problemas de escalabilidad o no se pueden ingerir debido a problemas de disponibilidad, esos datos pueden perderse y podrían quedar sesgados o ser parciales.

Almacenamiento

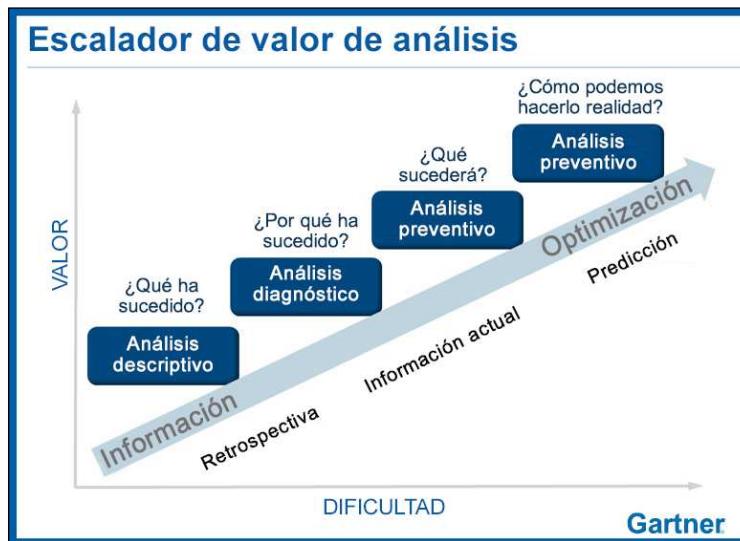
Las soluciones de IoT lidian normalmente con millones y miles de millones de registros que abarcan terabytes y petabytes de datos. Se trata de datos valiosos que pueden proporcionar conocimientos sobre las operaciones y su estado. Estos datos tienen que almacenarse para que se puedan realizar análisis sobre ellos. El almacenamiento debe estar disponible con facilidad para aplicaciones de análisis y los servicios que lo consumen. Debe proporcionar una velocidad y latencia suficientes desde una perspectiva de rendimiento, ofrecer alta disponibilidad y escalabilidad y ser seguro por naturaleza.

Transformación

Las soluciones de IoT se basan normalmente en los datos y tienen un volumen alto considerable. Imagina que cada coche cuenta con un dispositivo y cada uno envía mensajes cada cinco segundos. Si hay un millón de coches enviando mensajes, obtendremos 288 millones de mensajes al día y 8 mil millones de mensajes al mes. Todos estos datos combinados tienen gran cantidad de información y conocimientos ocultos; sin embargo, entender este tipo de datos mediante la simple observación es bastante complicado. Los datos capturados y almacenados se pueden consumir para resolver problemas empresariales. En función de la naturaleza del problema, no todos los datos capturados tienen importancia. Puede que solo haya un subconjunto de datos que se debe utilizar para resolver un problema. Los datos capturados y almacenados podrían además no ser uniformes. Para garantizar que los datos son uniformes, sin ningún sesgo, se debe ejecutar la transformación adecuada sobre ellos para que se puedan consumir con facilidad. Este proceso se conoce normalmente como transformación. Aquí, los datos se filtran, ordenan, eliminan, enriquecen y transforman en una estructura; de esta manera, componentes y aplicaciones posteriores podrán consumirlos con facilidad.

Análisis

Los datos transformados en el paso anterior se convierten en la entrada y la fuente de datos para análisis. Según la naturaleza del negocio y el problema en cuestión existen varios tipos diferentes de análisis que se pueden realizar en los datos transformados.



La figura anterior se extrae de *Gartner* y representa muy bien los diferentes tipos de análisis que pueden realizarse:

- **Análisis descriptivo:** este tipo de análisis ayuda a detectar patrones y detalles sobre estados de dispositivos de IoT y del estado general. Es normalmente la primera fase del análisis, en la que se identifican y resumen los datos para su posterior consumo por análisis más avanzados. Ayudará a encontrar información como el resumen, estadísticas relacionadas con probabilidad, la desviación y otros conceptos estadísticos básicos.
- **Análisis diagnóstico:** este tipo de análisis es más avanzado que el análisis descriptivo. Se construye sobre el análisis descriptivo e intenta responder a consultas acerca de por qué sucedieron las cosas. Trata de encontrar la causa raíz de los eventos que se han producido. Intenta encontrar respuestas mediante conceptos avanzados, como hipótesis y correlación.

- **Análisis predictivo:** este tipo de análisis intenta predecir cosas que tienen una alta probabilidad de producirse en el futuro. Consiste en la predicción basada en datos anteriores. La regresión es uno de los ejemplos basados en datos anteriores y podría, por ejemplo, predecir el precio de un coche, acciones en el mercado de valores, cuándo reventará el próximo neumático, etc.
- **Análisis prescriptivo/cognitivo:** este análisis se encuentra en los niveles más altos de madurez y ayudar a adoptar acciones automáticamente. Este análisis ayuda a identificar la acción más adecuada que debería aplicarse para garantizar que el estado de los dispositivos y soluciones no se degrade y que se puedan adoptar medidas proactivas. Esto ayuda a evitar y eliminar los problemas de raíz.

Presentación

El análisis ayuda a identificar respuestas, patrones y conocimientos basados en datos disponibles del pasado. Estos conocimientos también tienen que estar disponibles para todas las partes interesadas en diferentes formas y en un formato que puedan entender. Deben poder consumirse con facilidad y estar disponibles en varios formatos. Los paneles e informes correspondientes se pueden generar, de manera estadística o dinámica, y se pueden presentar, a continuación, a las partes interesadas. Las partes interesadas pueden consumir estos informes para adoptar acciones adicionales y mejorar continuamente su solución.

Azure IoT

El conocimiento de los detalles sobre las distintas etapas ayuda a crear soluciones de IoT integrales. Cada una de estas etapas es esencial y su implementación es imprescindible para su éxito. Azure proporciona una gran cantidad de servicios para cada una de estas etapas. Además de estos servicios, Azure proporciona centros de IoT, el núcleo del servicio y la plataforma de IoT que es capaz de hospedar soluciones de IoT complejas, altamente disponibles y escalables. Trataremos con mayor detalle los centros de IoT tras describir el resto de servicios.

En esta sección se detallará brevemente cada uno de estos servicios.

Dispositivos	Conectividad de dispositivos	Almacenamiento	Análisis	Presentación y acción
Event Hubs	Base de datos SQL	Machine learning		App Service
Bus de servicio	Almacenamiento de tabla/blob	Stream Analytics		Power BI
Fuentes de datos externas	DocumentDB	HDInsight		Centros de notificaciones
	Fuentes de datos externas	Data Factory		Servicios móviles
			Servicios de BizTalk	

Identidad

Los centros de IoT de Azure también proporcionan servicios para la autenticación de dispositivos. Los centros de IoT proporcionan una interfaz para generar un hash de identidad único para cada dispositivo. Cuando los dispositivos envían mensajes que contienen este hash, el centro de IoT puede identificarlos tras la verificación de su propia base de datos para comprobar la existencia de estos hashes.

Captura

Azure proporciona puertas de enlace de IoT que permiten adaptar los dispositivos no compatibles con centros de IoT y habilitar la transferencia de datos. Existen puertas de enlace locales o intermedias implementadas cerca de los dispositivos, de manera que varios dispositivos se pueden conectar a una sola puerta de enlace para enviar su información. Del mismo modo, se pueden implementar varios clústeres de dispositivos con una puerta de enlace local. Puede haber una puerta de enlace de cloud implementada en el propio cloud, capaz de aceptar datos de varias fuentes e ingerirlos para los centros de IoT.

Ingesta

Azure proporciona un centro de IoT que se convierte en el único punto de contacto de dispositivos y otras aplicaciones para enviar datos. En otras palabras, la ingestión de los mensajes de IoT es responsabilidad del servicio del centro de IoT. Existen otros servicios, como los centros de eventos y la infraestructura de mensajería del bus de servicio que pueden proporcionar la ingestión de mensajes entrantes; sin embargo, los beneficios y ventajas de los centros de IoT para la ingestión de datos superan con creces a los de los centros de eventos y la mensajería del bus de servicio. De hecho, los centros de IoT se han creado específicamente para la ingestión de mensajes de IoT dentro del ecosistema de Azure, de manera que el resto de servicios y componentes pueda actuar sobre ellos.

Almacenamiento

Azure proporciona varios tipos de almacenamiento para almacenar los mensajes procedentes de los dispositivos de IoT. Estas cuentas de almacenamiento incluyen el almacenamiento de datos relacionales, datos NoSQL sin esquema y blobs:

- **Base de datos SQL:** la base de datos SQL proporciona almacenamiento de datos relacionales, documentos JSON y documentos XML. Proporciona un lenguaje de consulta SQL enriquecido y utiliza un auténtico SQL server como servicio. Los datos de los dispositivos se pueden almacenar en bases de datos SQL si los datos están bien definidos y se espera que el esquema no sufra cambios con frecuencia.
- **Almacenamiento de Azure:** el almacenamiento de Azure proporciona almacenamiento de tablas y blobs. El almacenamiento de tablas ayuda a almacenar datos como entidades en las que el esquema no es importante. Se trata de una implementación de bases de datos NoSQL. Los blobs ayudan a almacenar archivos en contenedores como blobs.
- **CosmosDB/DocumentDB:** DocumentDB es una base de datos NoSQL auténtica y de escalabilidad empresarial, disponible como servicio capaz de almacenar datos sin esquema. Se trata de una verdadera base de datos distribuida que puede abarcar continentes con alta disponibilidad y escalabilidad de los datos.
- **Fuentes de datos externas:** además de los servicios de Azure, los consumidores pueden aportar sus propios almacenes de datos, como un SQL server en máquinas virtuales de Azure, y pueden usarlos para almacenar datos en un formato relacional.

Transformación y análisis

- **Data Factory:** Azure Data Factory es un servicio de integración de datos basado en el cloud que nos permite crear flujos de trabajo basados en datos en el cloud para organizar y automatizar el movimiento de datos y su transformación. Azure Data Factory ayuda a crear y programar flujos de trabajo basados en datos (denominados procesos) que pueden ingerir datos de diferentes almacenes de datos, procesar/transformar los datos mediante el uso de servicios de cálculo como **Azure HDInsight Hadoop, Spark, Azure Data Lake Analytics** y **Azure Machine Learning**, y publicar los datos de salida en almacenes de datos, como un almacén de datos de SQL de Azure, para que los consuman aplicaciones de **business intelligence (BI)**. Se parece más a una plataforma que primero **extrae y carga** (EL, Extract-and-Load) y, a continuación, **transforma y carga** (TL, Transform-and-Load) en lugar de una plataforma tradicional que **extrae, transforma y carga** (ETL, Extract-Transform-and-Load).
- **Azure HDInsight:** Microsoft y Hortonworks se han unido para ayudar a las empresas al ofrecer la plataforma de análisis de big data en el servicio cloud de Azure. HDInsight es un entorno de servicio en el cloud completamente administrado y de gran potencia impulsado por Apache Hadoop y Apache Spark con Microsoft Azure HDInsight. Ayuda en la aceleración de cargas de trabajo sin problemas con el servicio cloud de big data líder del sector de Hortonworks y Microsoft.
- **Azure stream analytics:** se trata de un servicio de análisis de datos en tiempo real completamente administrado que ayuda en la realización de cálculos y transformaciones en datos de streaming. Stream Analytics puede examinar grandes volúmenes de datos que fluyen de dispositivos o procesos, extraer información del flujo de datos y buscar patrones, tendencias y relaciones.
- **Machine learning:** Machine learning es una técnica de ciencia de datos que permite a los equipos utilizar los datos existentes para prever comportamientos, resultados y tendencias futuros. Mediante Machine learning, los equipos aprenden sin haberse programado explícitamente para ello. Azure Machine Learning es un servicio de análisis predictivo en el cloud que permite crear e implementar rápidamente modelos predictivos como soluciones de análisis.

Proporciona una biblioteca lista para usar de algoritmos para crear modelos en un PC conectado a Internet e implementar con rapidez soluciones predictivas.

Presentación

Después de haber realizado los análisis pertinentes en los datos, los datos se deben presentar a las partes interesadas en un formato que puedan consumir. Hay numerosas formas en que se pueden presentar los conocimientos obtenidos de los datos. Entre ellas se incluyen la presentación de datos mediante aplicaciones web implementadas con Azure App Services, el envío de datos a centros de notificación que pueden, a continuación, enviar notificaciones a aplicaciones móviles y mucho más. Sin embargo, el método ideal para presentar y consumir los conocimientos es mediante el uso de informes y paneles de **Power BI**. Power BI es una herramienta de visualización de Microsoft para representar paneles e informes dinámicos en Internet de forma que sean accesibles desde cualquier lugar y en cualquier red.

Centros de IoT

Normalmente los proyectos de IoT presentan una naturaleza compleja. La complejidad surge debido a la gran cantidad de dispositivos y datos, la integración de los dispositivos por todo el mundo, la supervisión y la auditoría de los dispositivos, el almacenamiento de datos, la transformación y análisis en petabytes de datos y, finalmente, la adopción de acciones basada en los conocimientos obtenidos. Además, estos proyectos son de larga duración, tienen períodos de gestación largos y sus requisitos continúan cambiando debido a las escalas de tiempo.

Si una empresa quiere emprender el viaje de realizar proyectos de IoT, descubrirá antes de lo que piensa que estos problemas no son fáciles de resolver. Estos proyectos necesitan hardware de gran tamaño en términos de cálculo y almacenamiento junto con servicios que puedan trabajar con este tipo de volúmenes elevados de datos.

El centro de IoT es una plataforma que se ha creado para facilitar y habilitar proyectos de IoT con el objetivo de que la entrega sea más rápida, mejor y más sencilla. Ofrece todas las características y servicios necesarios para lo siguiente:

- Registro de dispositivos
- Conectividad de dispositivos
- Puertas de enlace de campo
- Puertas de enlace en el cloud

- Implementación de protocolos del sector como AMQP y MQTT
- Centro para almacenar mensajes entrantes
- Enrutamiento de mensajes basado en el contenido y las propiedades del mensaje
- Varios puntos de conexión para diferentes tipos de procesamiento
- Conectividad a otros servicios en Azure para análisis normal y en tiempo real y mucho más

Protocolos

El centro de IoT de Azure admite de manera nativa la comunicación a través de los protocolos MQTT, AMQP y HTTP. En algunos casos, es posible que los dispositivos o puertas de enlace de campo no sean capaces de usar uno de estos protocolos estándar y necesitarán una adaptación de protocolos. En estos casos, se puede implementar una puerta de enlace personalizada. Una puerta de enlace personalizada permite la adaptación de protocolos para puntos de conexión del centro de IoT al tender un puente entre el tráfico entrante y saliente del centro de IoT.

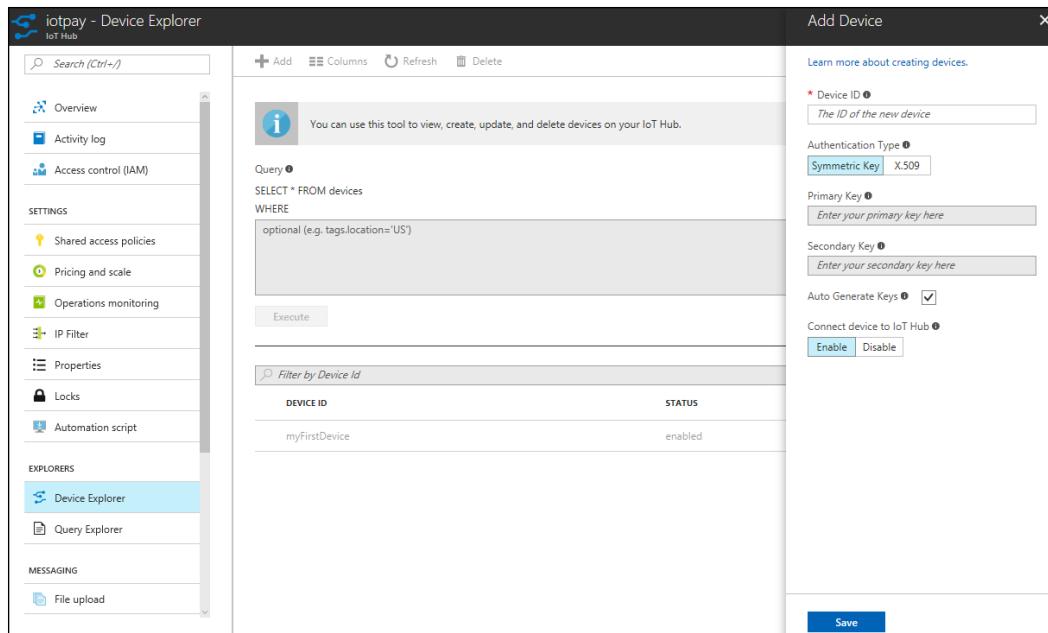
Registro de dispositivos

Los dispositivos se deben registrar antes de poder enviar mensajes al centro de IoT. El registro de dispositivos se puede realizar manualmente mediante el portal de Azure o se puede automatizar mediante el SDK del centro de IoT. Azure proporciona aplicaciones de simulación de muestra, con la ayuda de las cuales se facilita el registro de dispositivos virtuales con fines de desarrollo y pruebas. También hay un simulador online de Raspberry Pi que se puede utilizar como dispositivo virtual y, por lo tanto, existen otros dispositivos físicos que se pueden configurar para conectarse al centro de IoT.

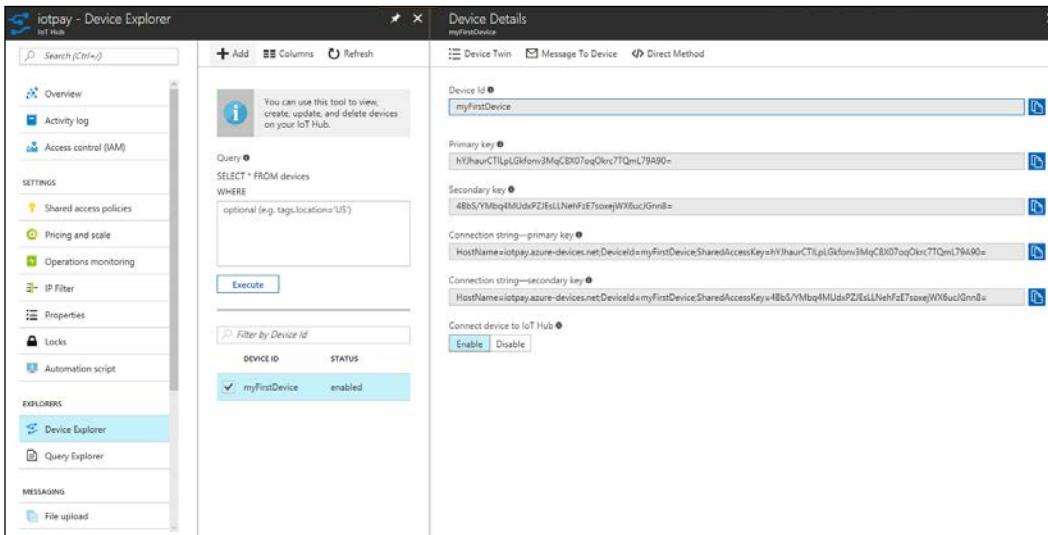
Para simular un dispositivo desde un PC local que se utiliza normalmente con fines de desarrollo y pruebas, hay tutoriales disponibles en los documentos de Azure en varios idiomas. Están disponibles en <https://docs.microsoft.com/azure/iot-hub/iot-hub-get-started-simulated>.

El simulador online de Raspberry Pi está disponible en <https://docs.microsoft.com/azure/iot-hub/iot-hub-raspberry-pi-web-simulator-get-started> y para usar dispositivos físicos para su registro en el centro de IoT, se deben usar los pasos mencionados en <https://docs.microsoft.com/azure/iot-hub/iot-hub-get-started-physical>.

Para agregar manualmente un dispositivo mediante el portal de Azure, el centro de IoT ofrece el menú **Explorador de dispositivos**, que puede utilizarse para configurar un nuevo dispositivo.



Tras crear la identidad del dispositivo, se debe usar una cadena de conexión de clave principal para el centro de IoT en cada dispositivo que se va a conectar con este.



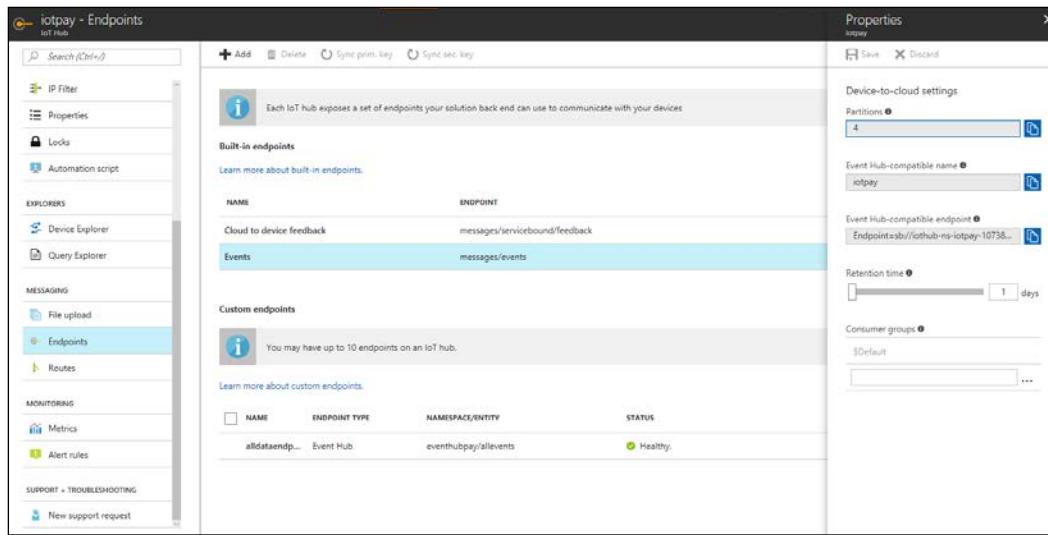
Administración de mensajes

Una vez se han registrado los dispositivos con el centro de IoT, ya pueden empezar a interactuar con él. La interacción puede ser de dispositivo a cloud o de cloud a dispositivo.

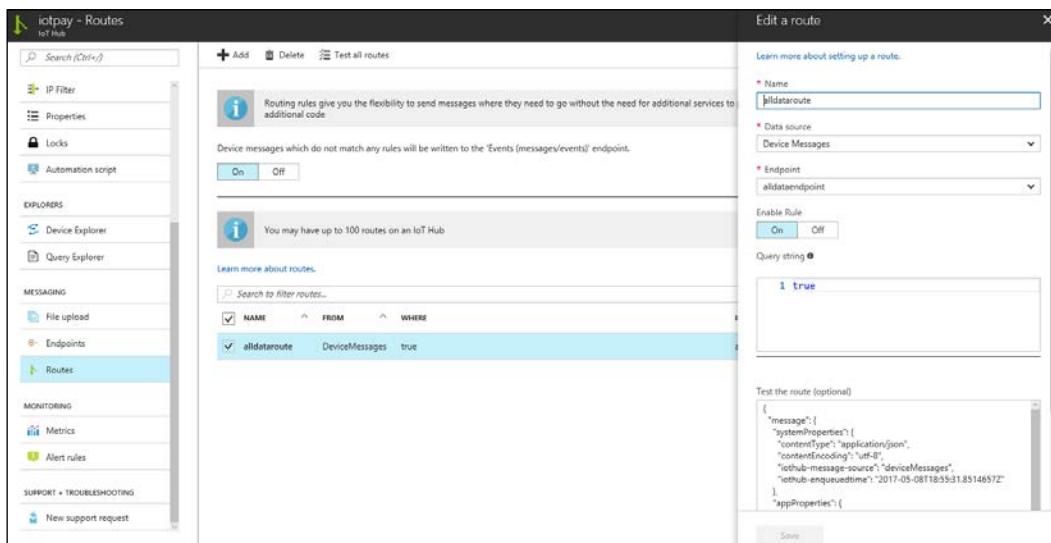
Mensajería de dispositivo a cloud

Una de las prácticas recomendadas que se debe seguir en esta comunicación es que aunque el dispositivo pueda estar capturando una gran cantidad de información, solo se deben transmitir al cloud aquellos datos que realmente sean importantes. El tamaño del mensaje es muy importante en las soluciones de IoT debido a la naturaleza intrínseca de estas soluciones de tener normalmente volúmenes muy altos. Hasta 1 KB de datos adicionales puede provocar que se desperdicie un gigabyte de almacenamiento y procesamiento. Cada mensaje tiene propiedades y una carga real. Las propiedades definen los metadatos del mensaje. Estos metadatos contienen datos sobre el dispositivo, la identificación, etiquetas y otras propiedades que resultan útiles en el enrutamiento y la identificación de mensajes.

Los dispositivos o puertas de enlace en el cloud se deben conectar a los centros de IoT para transferir datos. Los centros de IoT proporcionan puntos de conexión públicos que pueden utilizar los dispositivos para conectarse y enviar los datos. El centro de IoT se debe considerar como el primer punto de contacto para el procesamiento de backend, por lo que debe ser capaz de seguir transmitiendo y enrutiando estos mensajes a varios servicios. De manera predeterminada, los mensajes se almacenan en el centro de eventos. Se pueden crear varios centros de eventos para diferentes tipos de mensajes.



Los mensajes se pueden enrutar a diferentes puntos de conexión según las propiedades de cuerpo y encabezado del mensaje y esto mismo se muestra en la siguiente imagen.



El mensaje en el centro de IoT permanece durante 7 días de manera predeterminada. Su tamaño puede aumentar hasta 256 KB.

Hay una acción de muestra que actúa como simulador para el envío de mensajes al cloud. Está disponible en varios lenguajes y la versión C# se encuentra en <https://docs.microsoft.com/azure/iot-hub/iot-hub-csharp-csharp-c2d>.

Mensajería de cloud a dispositivo

El centro de IoT de Azure es un servicio administrado que proporciona una infraestructura de mensajería bidireccional. Los mensajes se pueden enviar desde el cloud a los dispositivos y, en función del mensaje, los dispositivos pueden actuar sobre ellos.

Hay tres tipos de patrones de mensajería de cloud a dispositivo:

- Los métodos directos requieren una confirmación inmediata del resultado. Los métodos directos se utilizan a menudo para el control interactivo de dispositivos, como la apertura y el cierre de puertas de garaje. Sigue el patrón de solicitud-respuesta.

- Las propiedades deseadas del gemelo para comandos de larga duración pretenden colocar el dispositivo en un estado deseado determinado. Por ejemplo, establecer el intervalo de envío de telemetría en 30 minutos. Los dispositivos gemelos son documentos JSON que almacenan información de estado del dispositivo (metadatos, configuraciones y condiciones). El centro de IoT conserva un dispositivo gemelo para cada dispositivo del centro de IoT.
- Mensajes del cloud al dispositivo para notificaciones unidireccionales en la aplicación del dispositivo. Aquí se sigue el patrón de enviar y olvidar (fire and forget).

Seguridad

La seguridad es un aspecto importante en aplicaciones basadas en IoT. Las aplicaciones basadas en IoT constan de dispositivos que utilizan una conexión a Internet pública para la conectividad con aplicaciones de backend. La protección de dispositivos, aplicaciones de backend y conectividad frente a usuarios malintencionados y hackers se debe considerar como una prioridad máxima para el éxito de estas aplicaciones.

Seguridad en IoT

Las aplicaciones de IoT se crean principalmente en torno a Internet y la seguridad debe desempeñar una función importante para garantizar que la solución no se vea comprometida en cuanto a identidad, confidencialidad e integridad. Algunas de las decisiones importantes de seguridad que afectan a la arquitectura de IoT son las siguientes:

- Dispositivos que utilizan puntos de conexión REST HTTPS frente a HTTP: los puntos de conexión REST protegidos por certificados garantizan que los mensajes transferidos del dispositivo al cloud y viceversa están cifrados y firmados correctamente. Los mensajes no deben tener sentido para un intruso y deben ser muy difíciles de descifrar.
- Si los dispositivos están conectados a una puerta de enlace local, esta se debe conectar al cloud mediante un protocolo HTTP seguro.
- Los dispositivos se deben registrar en los centros de IoT del cloud antes de que puedan enviar mensajes.
- La información transferida al cloud se debe conservar en un almacenamiento que esté bien protegido en cuanto a confidencialidad, integridad e identidad. Para la conexión se deben usar los tokens SAS o las cadenas de conexión correspondientes que se almacenan en Azure Key Vault .
- Se debe utilizar Azure Key Vault para almacenar todos los secretos, contraseñas y credenciales, incluidos los certificados.

Escalabilidad

La escalabilidad del centro de IoT es algo diferente a la de otros servicios. En el centro de IoT, hay dos tipos de mensajería:

- **Entrante:** mensajes del dispositivo al cloud
- **Saliente:** mensajes del cloud al dispositivo

Y ambos tienen que evaluarse en términos de escalabilidad.

El centro de IoT ofrece un par de opciones de configuración durante el tiempo de aprovisionamiento para configurar la escalabilidad. Estas opciones también están disponibles después del aprovisionamiento y se pueden actualizar para adaptarse mejor a los requisitos de la solución en términos de escalabilidad.

Las opciones de escalabilidad para el centro de IoT son:

- La edición Sku que se corresponde con el tamaño del centro de IoT
- Número de unidades

Edición SKU

El Sku en el centro de IoT determina el número de mensajes que puede manejar por unidad al día y esto incluye tanto mensajes entrantes como salientes. Existen cuatro Sku definidos. Son los siguientes:

- **Gratis:** permite 8000 mensajes por unidad al día y permite mensajes entrantes y salientes. Se puede aprovisionar 1 unidad como máximo. Esta edición es apta para familiarizarse con el sistema y probar las capacidades del servicio de centro de IoT.
- **Estándar (S1):** permite 400.000 mensajes por unidad al día y permite mensajes entrantes y salientes. Se pueden aprovisionar 200 unidades como máximo. Esta edición es apta para un número pequeño de mensajes.
- **Estándar (S2):** permite seis millones de mensajes por unidad al día y permite mensajes entrantes y salientes. Se pueden aprovisionar 200 unidades como máximo. Esta edición es apta para un número grande de mensajes.

- **Estándar (S3):** permite 300 millones de mensajes por unidad al día y permite mensajes entrantes y salientes. Se puede aprovisionar 10 unidades como máximo. Esta edición es apta para un número muy grande de mensajes.

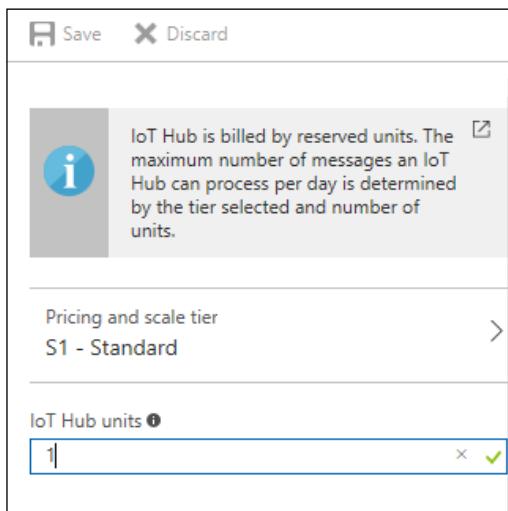
Choose your pricing and scale tier		
S1 Standard	S2 Standard	S3 Standard
400k messages/unit/day	6M messages/unit/day	300M messages/unit/day
 Device-to-cloud telemetry  Cloud-to-device messaging 200 units maximum	 Device-to-cloud telemetry  Cloud-to-device messaging 200 units maximum	 Device-to-cloud telemetry  Cloud-to-device messaging 10 units maximum
3,304.81 INR PER IOT HUB UNIT	33,048.13 INR PER IOT HUB UNIT	330,481.25 INR PER IOT HUB UNIT
F1 Free		
8k messages/unit/day  Device-to-cloud telemetry  Cloud-to-device messaging 1 unit IoT hub cannot transition between free and paid tiers.		
Select		

Un lector inteligente habrá notado que el Sku Estándar S3 permite solo un máximo de 10 unidades en comparación con el resto de unidades estándar que permiten un máximo de 200 unidades. Esto está directamente relacionado con el tamaño de las máquinas en que se ejecutan. El tamaño y la capacidad de las máquinas virtuales para Estándar S3 son considerablemente superiores a otros Sku donde el tamaño sigue siendo el mismo.

Unidades

Las unidades definen el número de instancias de cada SKU en que se ejecuta el servicio. Por ejemplo, 2 unidades de Sku Estándar S1 significa que el centro de IoT es capaz de manejar $400.000 * 2 = 800.000$ mensajes al día.

Las unidades aumentan la escalabilidad de la aplicación.



Alta disponibilidad

El centro de IoT es un servicio PaaS de Azure. Los consumidores y usuarios no interactúan directamente con el número y el tamaño subyacentes de las máquinas virtuales en las que se ejecuta el servicio de centro de IoT. Los usuarios deciden la región, el Sku del centro de IoT y el número de unidades para su aplicación. Azure se encarga de determinar y ejecutar el resto de la configuración. Azure se asegura de que cada servicio PaaS ofrece un alto nivel de disponibilidad de manera predeterminada. Para ello, se asegura de que varias máquinas virtuales aprovisionadas que ejecutan el servicio estén ubicadas en soportes separados en el centro de datos. Para lograrlo, coloca esas máquinas virtuales en un conjunto de disponibilidad y en un dominio de error y actualización diferente. Esto ayuda a ofrecer una alta disponibilidad para el mantenimiento planificado y no planificado. Los conjuntos de disponibilidad se ocupan de la alta disponibilidad en el nivel de centro de datos.

Para lograr una alta disponibilidad entre varios centros de datos de una región o entre regiones, los consumidores deben, de manera adicional, aprovisionar servicios de centro de IoT adicionales en regiones diferentes y asegurarse de que los dispositivos estén registrados en todos los servicios de centro de IoT, cada uno con el mismo identificador. Cree una aplicación de supervisión independiente en cada región, como una aplicación web, todas asociadas mediante el uso de un administrador de tráfico y supervisando continuamente el punto de conexión del administrador de tráfico en cuanto a la no disponibilidad de la ubicación actual para redirigir el tráfico a otros centros de IoT, y escriba una lógica para enrutar los mensajes de dispositivos al nuevo centro de IoT.

Resumen

IoT es una de las tecnologías emergentes más importantes de esta década y ya está revolucionando los sectores con el tipo de soluciones que pueden crear. Cosas que parecían imposibles ahora permiten el seguimiento, la supervisión y la aplicación de acciones de forma remota. El centro de IoT actúa como una plataforma que facilita la creación y entrega de soluciones de IoT al consumidor de una forma más rápida, mejor y más asequible. Proporciona implementación de todos los protocolos del sector, como MQTT y AMQP, junto con puertas de enlace de campo que pueden adaptar los dispositivos que no son estándar. Ofrece características de alta disponibilidad, escalabilidad y seguridad tanto para mensajes como para la solución general. Proporciona conectividad a un gran número de servicios de Azure y ayuda en el enrutamiento de mensajes a múltiples puntos de conexión, cada uno capaz de procesar mensajes. IoT puede acelerar todo el ciclo de vida de desarrollo y ayuda a que las estrategias de comercialización de las empresas sean más rápidas.

Azure proporciona numerosos servicios y opciones de almacenamiento de datos. El siguiente capítulo será una introducción sobre el almacenamiento de datos y los servicios en Azure.

7

Diseño e implementación de soluciones de datos

En la actualidad, la idea de que “los datos son la nueva moneda” está adquiriendo más sentido que nunca. Toda organización que tenga una cantidad considerable de datos está en posesión de información y conocimientos capaces de transformar el futuro. Durante la última década, se han producido innovaciones que han contribuido enormemente a la explosión de datos. La primera de ellas es el auge de las plataformas de redes sociales. Hay plataformas que tienen millones, si no miles de millones, de usuarios en todo el mundo, como Facebook, LinkedIn y Twitter; por lo general, esto supone millones de contenidos cada día. Este contenido derivado de las interacciones en las redes sociales entraña una gran cantidad de sabiduría y conocimientos. Otro paradigma transformador de los big data es el IoT. Los dispositivos IoT generan miles de millones de mensajes a diario para diversos sectores. El hardware se está abaratando, y esto contribuye aún más a su uso en contextos de IoT. También resulta bastante evidente que los métodos y herramientas convencionales no son adecuados para cargar un volumen de datos tan grande y generar conocimiento a partir de él. No están previstos para el análisis de big data. La disponibilidad de estos grandes volúmenes de datos ha dado lugar a innovaciones en cuanto al almacenamiento de datos de gran tamaño. El uso de sistemas de big data, como Hadoop, se ha generalizado bastante para el almacenamiento de grandes volúmenes de datos, y a partir de entonces, el análisis de progresos y la transmisión de datos en tiempo real se han convertido en puntos focales de innovación. El machine learning, la inteligencia artificial y el aprendizaje profundo han adquirido protagonismo a la hora de buscar conocimiento derivado de big data.

Los datos se caracterizan por volumen, velocidad y diversidad de formato. Se necesitan servicios diversos que puedan unirse entre sí para crear soluciones completas para datos que sean de gran volumen, de alta velocidad y que estén disponibles en varios formatos. Estos servicios deben ser de clase empresarial y escalables, con una alta disponibilidad, centrados en el rendimiento y seguros. Azure es una plataforma madura para gestionar todo el ciclo de vida de los datos, desde el almacenamiento hasta el análisis y la visualización: todos están disponibles con múltiples opciones para cada etapa del ciclo de vida.

En este capítulo, veremos en detalle lo siguiente:

- Azure SQL
- Extensión y particionamiento de Azure SQL
- Azure NoSQL Cosmos DB
- Azure SQL frente a Cosmos DB
- Azure Data Factory
- Azure Stream Analytics
- Data Lake
- Table Storage
- Almacenamiento de datos SQL

Azure SQL

Azure SQL es una base de datos cloud como servicio disponible en Azure. La tecnología de base de datos SQL Server insignia de Microsoft ha estado disponible durante más de 20 años, y Microsoft la proporciona también en el cloud y ofrece instancias de base de datos completas a sus usuarios. Azure SQL es un servicio de base de datos relacional completamente administrado que ofrece mayor productividad, aprovisionamiento más rápido y sencillo, y alternativas más económicas. Proporciona casi todas las características de un SQL Server completo sin necesidad de gestionar infraestructura y licencias.

Azure SQL se basa en un motor de base de datos SQL Server y proporciona una base de datos relacional como característica central. Los datos se almacenan en formato de tabla, organizados en filas y columnas. Una combinación de columnas forma el esquema de la tabla y las filas se almacenan en forma de tabla. Azure SQL es uno de los destinos de datos más populares de Azure. Casi todas las aplicaciones necesitan datos para persistir en el tiempo, y Azure SQL proporciona almacenes de datos para estas aplicaciones.

Azure SQL es una base de datos basada en transacciones. Esto significa que proporciona soporte para transacciones. Existen cuatro postulados básicos aplicables a las transacciones, también conocidos como **propiedades ACID**. Son los siguientes:

- Atomicidad
- Coherencia
- Aislamiento
- Durabilidad

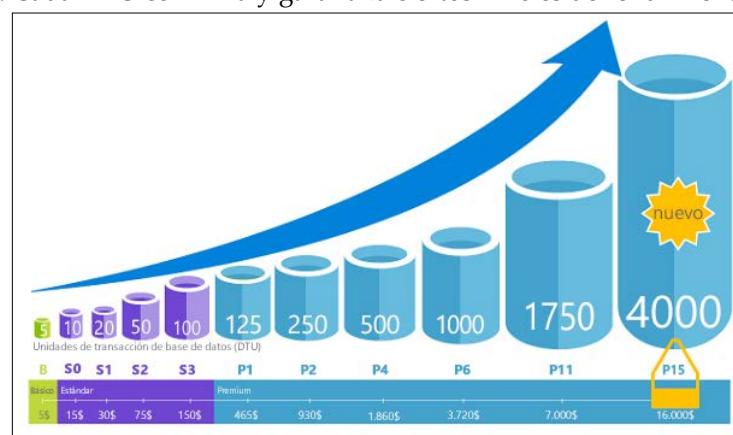
Azure SQL consta de dos componentes importantes:

- **Servidor lógico de Azure SQL:** equivale a una instancia de SQL Server en el centro de datos on-premises o a máquinas virtuales de Azure. Un servidor es un límite físico y lógico que consta de varias bases de datos. Proporciona que un límite de seguridad con su propio conjunto de usuarios, grupos y permisos. Los usuarios deben autenticarse en el servidor para poder acceder a las bases de datos que este contiene. El servidor proporciona todas las funciones de administración y las características de administración y mantenimiento de Azure SQL.
- **Bases de datos de Azure SQL:** es el componente central de Azure SQL, pues en él es donde se almacenan los datos. Cada base de datos se compone de varias tablas, y cada tabla consta a su vez de columnas que, juntas, forman filas.

Las columnas de Azure SQL proporcionan diversos tipos de datos que permiten la compatibilidad con datos de distintos tipos. Esto incluye tipos de datos generales, tales como char, varchar, enteros, etc., y otros más avanzados como JSON, XML y espaciales. Las tablas pueden normalizarse de forma que tanto la escritura como la lectura de datos puedan ser extremadamente rápidas, aunque esto añade datos redundantes. También permite crear varios tipos de índices en las columnas (agrupados, no agrupados, de cobertura, únicos, etc.) para mejorar el rendimiento de las consultas.

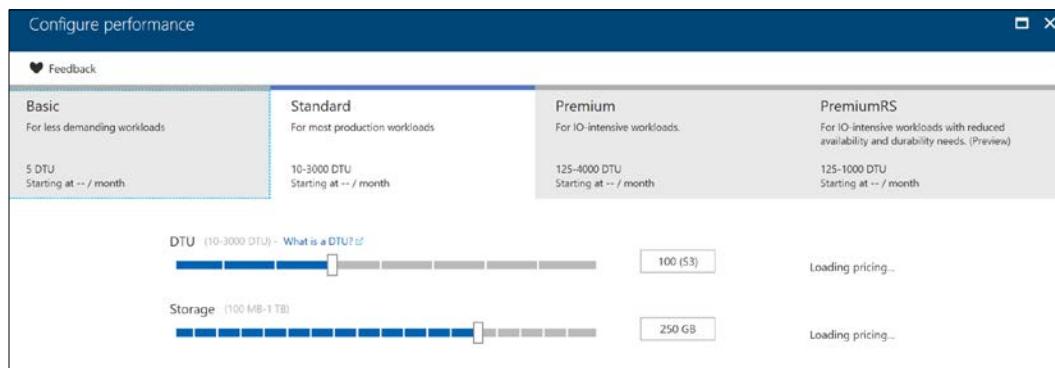
Proporciona **Transact-SQL** como lenguaje para emitir declaraciones de manipulación y de definición de datos. El motor de datos SQL asume la pesada carga que suponen el análisis, la conversión en tokens, el almacenamiento, la búsqueda de un plan de ejecución óptimo y la ejecución propiamente dicha de las consultas. También interactúa con el motor de almacenamiento para recuperar datos desde almacenamiento a la memoria y viceversa. Proporciona e implementa capacidades transaccionales y ofrece características de administración tales como registros, tareas, agentes, copia de seguridad, restauración y disponibilidad ininterrumpida.

Rendimiento de Azure SQL se mide en **unidades de transacción de base de datos** o **DTU** (del inglés Database Transaction Units). Para que Azure pueda garantizar un rendimiento mínimo **del contrato de nivel de servicio (SLA)**, se calcula de forma conjunta una combinación de recursos de proceso, memoria y almacenamiento para formar DTU. Cada DTU confirma y garantiza ciertos niveles de rendimiento.



Azure SQL proporciona cuatro niveles de rendimiento. Son los siguientes:

- **Básico:** es para las cargas de trabajo menos exigentes y proporciona un rendimiento de 5 DTU.
- **Estándar:** es para cargas de trabajo generales de producción y proporciona cualquier rendimiento comprendido entre 10 y 3.000 DTU.
- **Premium:** es para cargas de trabajo con alta densidad de ES y proporciona entre 125 y 4.000 DTU.
- **PremiumRS:** también es para cargas de trabajo con alta densidad de ES, pero con disponibilidad y durabilidad reducidas, y proporciona entre 125 y 1.000 DTU.



Los usuarios deben decidir las necesidades de DTU óptimas en función de sus requisitos de aplicaciones. Pueden comenzar con pocas DTU e ir aumentando progresivamente según aumente la demanda de su aplicación. La ampliación de DTU añade más recursos de proceso, memoria y ESPS (entrada/salir por segundo) sin detener la base de datos. La ampliación o reducción se produce de forma transparente, sin afectar a la disponibilidad de la base de datos. El almacenamiento también puede ampliarse o reducirse en función de las necesidades.

Disponibilidad de Azure SQL

Azure garantiza un alto grado de disponibilidad de los datos. Los datos siguen estando disponibles incluso en caso de desastre o fallo de hardware, y proporciona diversas capacidades para garantizar un alto grado de disponibilidad.

Azure SQL realiza una copia de seguridad automática de la base de datos del usuario. Las copias de seguridad de estas bases de datos se realizan de tal forma que se minimice el tiempo de recuperación en caso de desastre. Se realiza una copia de seguridad semanal completa, una copia de seguridad diferencial cada hora, y una copia de seguridad de los registros de transacciones cada cinco minutos. Estas copias de seguridad se almacenan entonces en un almacenamiento con redundancia geográfica que conserva varias copias en distintas ubicaciones geográficas. En caso de desastre, el soporte técnico de Azure puede ayudar a obtener estas copias de seguridad y a restaurarlas.

Azure SQL también ofrece replicación geográfica de bases de datos, con hasta cuatro bases de datos secundarias legibles en distintas regiones. Todas estas bases de datos principales y secundarias se sincronizan en régimen continuo mediante replicación asincrónica. También ofrece la capacidad de comutar automáticamente a otra base de datos ubicada en otra región en caso de error.

Azure SQL permite la recuperación a un momento dado y también restaura las bases de datos.

Seguridad de Azure SQL

Azure proporciona características de seguridad avanzadas que garantizan que los datos jamás se vean comprometidos.

Azure SQL cifra y descifra de forma transparente todos los datos, copias de seguridad y archivos de registro mediante **cifrado de datos transparente (TDE)** automático. Las aplicaciones quedan ajenas a este proceso de cifrado y descifrado en segundo plano. Utilizan los datos de la forma en que lo hacen normalmente, pero los datos en reposo en un servidor SQL Server no pueden ser descifrados por cualquiera que pueda acceder a ellos con herramientas de base de datos. Las claves de cifrado se pueden administrar y almacenar en Azure Key Vault, el almacén de claves de Azure.

Azure SQL también se encarga de proteger los datos en tránsito que se estén transmitiendo a través de una red. Ofrece una característica **Always Encrypted** (siempre claves cifradas) que mantiene los datos cifrados incluso cuando se almacenan en reposo, durante el procesamiento de consultas y en la red.

Otra de las excelentes características de seguridad que proporciona Azure SQL es el **enmascaramiento dinámico de datos**. Esta característica ayuda a exponer datos a usuarios sin privilegios, ocultando los datos confidenciales en los resultados de consulta. También funciona de forma transparente sin cambiar la aplicación.

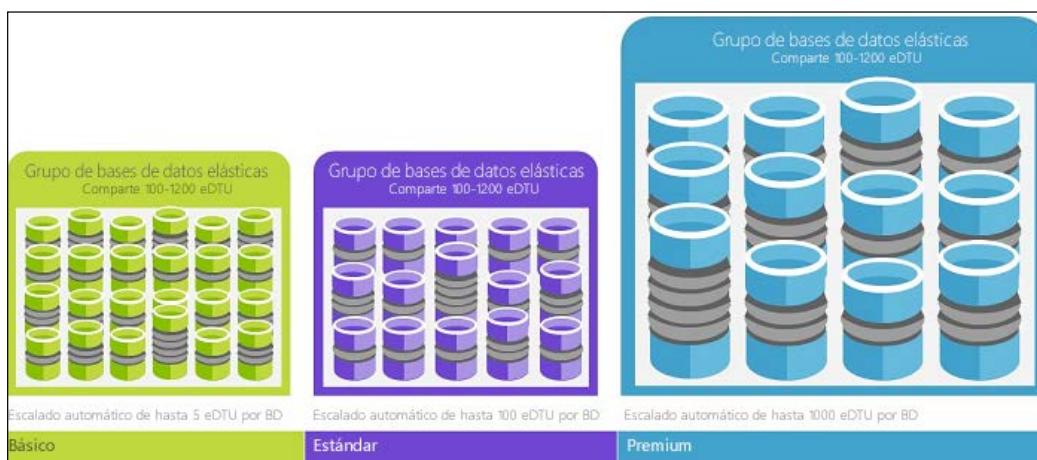
Los usuarios y grupos de Azure SQL pueden administrarse mediante Azure **Active Directory (AD)**. Desde Azure Active Directory, los usuarios pueden proporcionar acceso a servidores y bases de datos, y esto simplifica la administración general de identidades mediante su centralización y externalización.

La seguridad de Azure SQL va desde el servidor hasta las filas de las tablas. Proporciona seguridad de nivel de fila por la cual los usuarios pueden acceder solo a aquellas filas para las que están autorizados.

También permite acceder a aquellas direcciones IP expresamente permitidas por el administrador. Además, proporciona capacidades de auditoría avanzadas en cuanto a quién accede al servidor, y registros de todas las actividades de lectura y escritura de la base de datos.

Grupos elásticos

Las DTU de Azure son magníficas para garantizar un nivel mínimo de rendimiento y consumo de recursos; sin embargo, Azure SQL dedica los recursos de una determinada DTU a una base de datos SQL y sigue haciéndolo incluso si los recursos están infrautilizados. Cuando existe un gran número de bases de datos, cada base de datos tendrá su propia DTU dedicada, si bien algunas de ellas podrían no utilizarse en absoluto o permanecer infrautilizadas. Azure ofrece una alternativa mejor para tales situaciones en las que existe un gran número de bases de datos. Azure proporciona grupos elásticos que ayudan a alojar varias bases de datos en un mismo servidor SQL Server y las coloca en el mismo grupo. Cada grupo tiene asignada una serie de DTU dedicadas que se denominan eDTU. Este grupo podría contener bases de datos que consuman más recursos de lo previsto, y otras que consuman escasos recursos. Un grupo elástico ayuda a agregar las DTU utilizadas por cada una de estas bases de datos del grupo, asegurándose de que las bases de datos sobreutilizadas no se congestionen y de que las bases de datos infrautilizadas no se sobrecarguen. El coste de las eDTU se amortiza con varias bases de datos a pesar de que algunas bases de datos experimenten un incremento repentino en la demanda. Los grupos elásticos contribuyen a reducir el coste global de varias bases de datos, el tiempo que se aseguran de que las bases de datos más exigentes obtengan más recursos en momentos de necesidad.

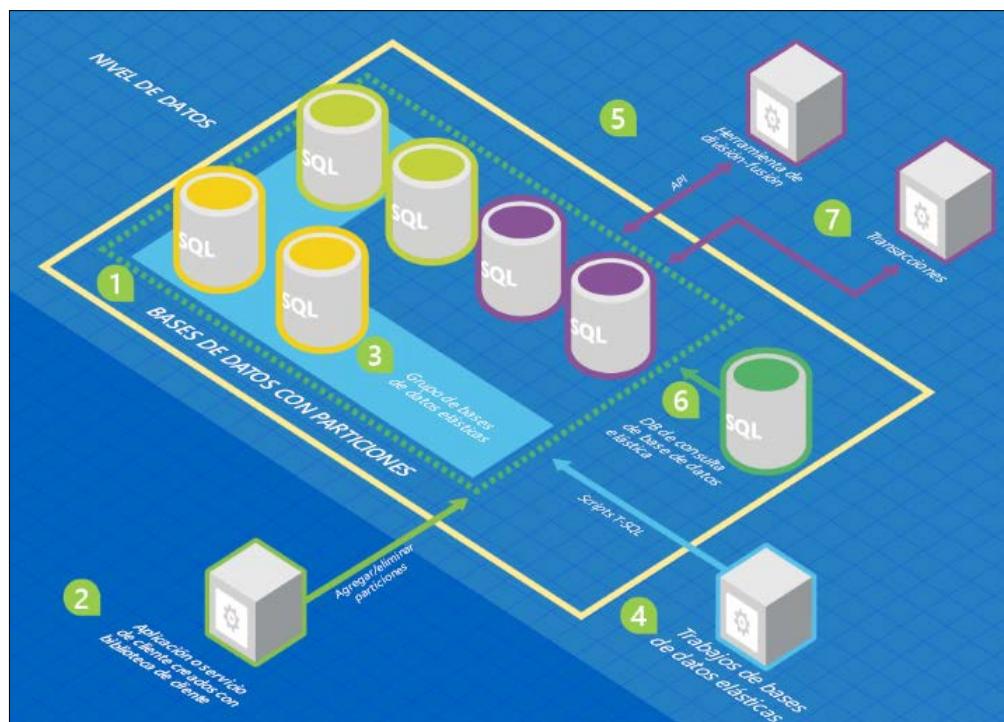


Escalabilidad horizontal de Azure SQL

Azure SQL es una plataforma versátil que proporciona características para atender a diferentes necesidades. Una de estas características es la escalabilidad horizontal de Azure, también denominada **particionamiento**. Imagina una situación en la que existe una gran base de datos con millones de filas. El rendimiento de las consultas está disminuyendo debido a que un gran número de operaciones de escritura y lectura tienen que recorrer grandes conjuntos de registros. El diseño mejora al descomponer esta gran base de datos en varias bases de datos más pequeñas y distribuir los datos entre ellas.

Sin embargo, escribir como un controlador o administrador resulta una tarea difícil y hercúlea. Azure proporciona características de escalabilidad para atender esta necesidad de partición horizontal de datos entre varias bases de datos. Las aplicaciones cliente se conectan a la base de datos principal de Azure, que se encarga de consultar datos de las bases de datos apropiadas, cotejar la información y volver a enviarla a la aplicación cliente. Las operaciones de escritura siguen este mismo proceso. Azure SQL también proporciona los medios para volver a fusionar estas particiones en una base de datos.

La tabla debe haber efectuado las particiones horizontales según una serie de claves predeterminadas, y la base de datos principal debe mantener información detallada sobre el intervalo de claves de cada base de datos. Esta base de datos principal también lleva a cabo una serie de tareas de administración adicionales. La imagen que se muestra a continuación ilustra una estrategia de particionamiento típica. Cada color muestra una base de datos con particiones. Mientras que algunas son grupos elásticos, el resto consume sus DTU dedicadas.



Azure Portal no proporciona la interfaz de usuario necesaria para el particionamiento de bases de datos. La escalabilidad horizontal y el particionamiento se llevan a cabo utilizando la API de REST, PowerShell o cualquier lenguaje de programación que consuma SDK de Azure SQL y la biblioteca cliente de bases de datos elásticas.



Las herramientas para administrar el particionamiento están disponibles en <https://docs.microsoft.com/azure/sql-database/sql-database-elastic-scale-get-started>.



Estas herramientas consumen SDK de Azure SQL y las bibliotecas cliente de bases de datos elásticas para ofrecer distintas alternativas para administrar el proceso de particionamiento y fusión.

Proporciona opciones para dividir y fusionar bases de datos, crear una nueva partición, añadir bases de datos a particiones, ejecutar consultas en particiones y eliminar una partición. La imagen anterior ilustra la interacción de estas aplicaciones cliente con Azure SQL.

Stream Analytics

Stream Analytics es un motor de procesamiento de datos y eventos completamente administrado que ayuda a proporcionar análisis en tiempo real sobre los datos transmitidos. Mientras que Cosmos DB y Azure SQL proporcionan datos ya disponibles en su almacenamiento, Stream Analytics proporciona procesos de análisis sobre datos que se están transmitiendo en directo a medida que se obtienen. No se trata de consultas a los datos almacenados en el almacenamiento permanente. Es una capacidad extremadamente potente para obtener información en tiempo real, en lugar de buscar información mucho después de que tenga lugar el evento.

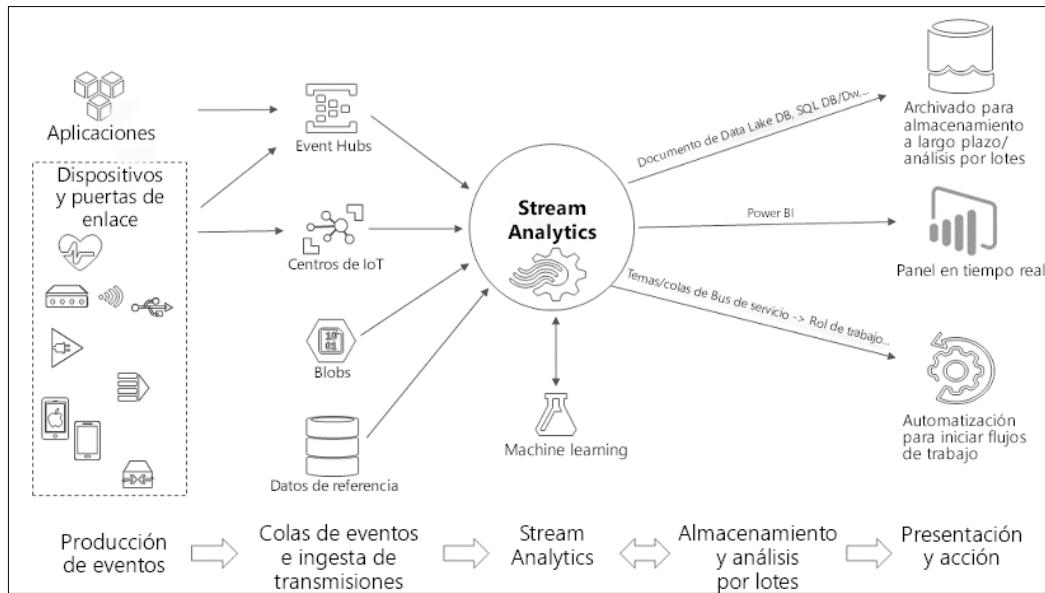
La virtud de Stream Analytics reside en el hecho de que los datos pueden proceder de cualquier lugar: aplicaciones, dispositivos IoT, otros servicios de Azure, tales como centros de eventos, o cualquier otro sistema.

Otro principio central esencial de Stream Analytics es el procesamiento de un gran volumen de datos. Se ha creado para procesar una enorme cantidad de datos y extraer información de ellos en tiempo real.

También permite la integración necesaria con otros sistemas y servicios para emprender acciones relacionadas con la información extraída, como, por ejemplo, enviar notificaciones, emitir alertas, iniciar sesión en varios servicios de análisis de registros y generar informes con herramientas de visualización.

El aspecto crucial de Stream Analytics radica en el hecho de que se pueden procesar datos de gran volumen, de alta velocidad y de diversos formatos mediante un único servicio en tiempo y emprender acciones sobre dichos datos.

El siguiente diagrama muestra la arquitectura de Stream Analytics:



Stream Analytics proporciona la funcionalidad de **extracción, transformación y carga (ETL)** para procesos en tiempo real. Define un trabajo configurado con datos de entrada junto con el origen de datos, las transformaciones y el registro de datos de salida con los destinos.

Existen numerosas situaciones en las que se puede implementar Stream Analytics. Son los siguientes:

- Son muchos los sectores que precisan soluciones analíticas en tiempo real, como los sistemas de estacionamiento, los sistemas de cabinas de peaje en las autopistas, las ventas de entradas de cine, etc.
- Análisis de opinión en las redes sociales
- Análisis de tráfico web
- Log Analytics para infracciones de seguridad y de otro tipo, entre otras
- Supervisión de infraestructura

Son varios los componentes que forman parte del servicio Stream Analytics de Azure.

Orígenes de datos

Stream Analytics actúa en los datos. Los datos se extraen de la ubicación de origen y, tras su transformación, se cargan en la ubicación de destino. En el lenguaje analítico de Azure, tanto la ubicación de origen como la de destino se denominan orígenes de datos. Existen orígenes de datos, tales como los centros de eventos y centros IoT de Azure, que pueden proporcionar datos a Azure Analytics. Puede haber otros orígenes de datos, tales como dispositivos IoT, que puedan enviar datos directamente a Stream Analytics, y otros que residan on-premises o en distintas plataformas en el cloud. Los orígenes de datos podrían ser un lugar en el que se generan los datos o el evento, o podrían ser ubicaciones de almacenamiento, tales como el almacenamiento de Azure, los centros IoT, etc.

Integración de datos

Los orígenes de datos son ubicaciones de provisión y recepción de datos. Sin embargo, el nexo que conecta a Azure Stream Analytics con ellas es la integración de datos. Azure Stream Analytics proporciona integración tanto para orígenes de datos de origen como de destino. Esta integración consulta datos comprendidos en el intervalo de tiempo especificado y los proporciona al motor de Stream Analytics para su procesamiento.

Transformación de datos

Los datos de entrada se procesan para recabar información. Este proceso de transformación implica el filtrado, aumento y enriquecimiento de los datos. Puede que no todos los datos de entrada sean de interés. Podría tratarse de datos incorrectos o ausentes. Para garantizar que la información no sea sesgada, los datos deben incorporarse en una etapa que proporcione resultados precisos. Esto constituye el trabajo de transformación. Azure Stream Analytics proporciona un lenguaje de consulta de tipo SQL que contribuye a agrupar, agregar, unir y filtrar los datos en función de las necesidades empresariales.

Motor Stream Analytics

Este es el componente central de Stream Analytics. Aquí es donde se ejecutan los trabajos de Stream Analytics y donde interactúan todos los componentes para realizar la extracción, transformación y carga (ETL) de los datos. Este motor se mantiene en funcionamiento y ejecuta continuamente trabajos en intervalos de series temporales para asegurarse de poder recopilar información en tiempo real.

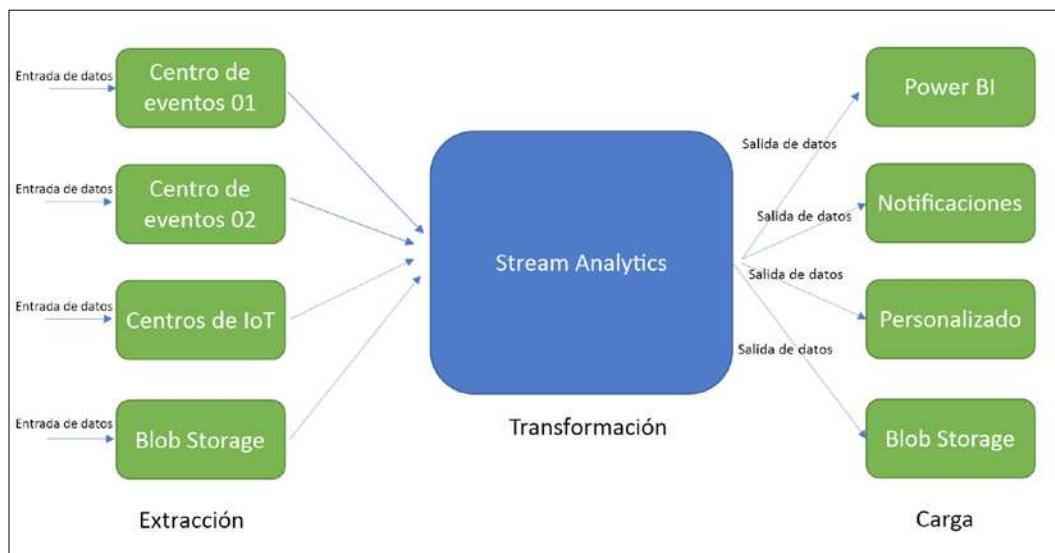
Almacenamiento y presentación

El motor de análisis también puede ayudar a emprender acciones en la información recopilada. Puede enviar datos al almacenamiento durable de Azure Blob Storage o presentar datos mediante herramientas de visualización, tales como Power BI, o usarlos para enviar notificaciones.

Arquitectura

Azure Stream Analytics es una plataforma madura y un servicio de análisis en tiempo real. Ofrece características para diversas entradas y salidas. En la imagen siguiente, aparecen varios **centros de eventos**, **centros IoT** y **almacenamientos de blob** como orígenes de datos de entrada. Estos pueden unirse entre sí, agruparse y agregarse en un mismo trabajo e información generada por **Stream Analytics**.

Azure está sujeto al coste de **salida de datos**, pero este coste no es aplicable a una región. Si la transferencia de datos tiene lugar en una región, el coste de salida no es aplicable. Es aplicable cuando la transferencia de datos tiene lugar entre regiones. Por este motivo, a modo de buenas prácticas, los orígenes de datos de Azure deben estar ubicados idealmente en la misma región que Azure Stream Analytics.



En Azure Portal, el panel de Azure Stream Analytics proporciona el número de entradas y salidas.



Azure Data Factory

Por lo general, los datos se almacenan en archivos planos, bases de datos relacionales, bases de datos NoSQL y otras ubicaciones en diversos formatos, como JSON, XML, CSV, archivos de texto y binarios. Además, cada almacén de datos define su propio modelo de almacenamiento de datos. Un ID de empleado de un almacén de datos se llama EMPIDID en otro almacén de datos y EID en el tercer almacén de datos de una misma organización. Existen varias facetas de los mismos datos en distintos almacenes de datos. Azure Data Factory es un servicio de integración de datos que ayuda a crear escenarios de orquestación y flujo de trabajo de ETL de datos. Azure Data Factory trabaja con datos procedentes de cualquier ubicación en el cloud, on-premises, y funciona en el nivel de cloud. Maneja grandes volúmenes de datos en diversos formatos.

Los flujos de trabajo de Azure Data Factory se conocen como canales de datos, y Azure Data Factory ayuda a crear, implementar, controlar y programar estos canales de datos fácilmente. Azure Data Factory es un servicio de PaaS administrado que proporciona un alto grado de disponibilidad, escalabilidad y tolerancia a errores a los canales de datos.

Azure Data Factory puede ingerir datos de varias ubicaciones y cargar los datos finales transformados a diversas ubicaciones y tipos de destino. En un principio, Data Factory parece otro servicio ETL; sin embargo, existe una gran diferencia. La principal diferencia entre Data Factory y otras herramientas ETL reside en la **transformación**. La transformación ejecutada por Data Factory se basa en servicios de proceso, como HDInsight Hadoop, Azure Data Lake Analytics, Spark y Azure Machine Learning, mientras que la transformación ejecutada por otras herramientas ETL tiene que ver con la transformación de filas y columnas, el filtrado y la agregación de filas y columnas, y el enriquecimiento de datos. Azure Data Factory prepara los datos mediante filtrado, agregación, agrupación, unión y enriquecimiento, y, a continuación, aplica la transformación utilizando servicios de proceso. El resultado de esta transformación es información útil; es decir, información que puede utilizarse posteriormente con herramientas de visualización tales como Power BI y otras aplicaciones personalizadas.

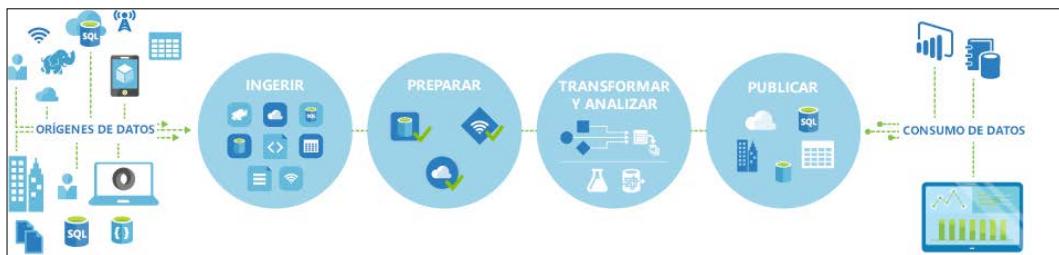
Azure Data Factory proporciona conectores para prácticamente todos los grandes almacenes de datos, incluidos SAP, Oracle, MongoDB y Cassandra, y admite varios tipos de protocolos y controladores para la conectividad con estos almacenes de datos, incluidos ODBC, OLEDB, ODATA y HTTP.

Los canales de Data Factory se pueden programar para ejecutarse con carácter diario, horario y semanal.

La principal diferencia entre Azure Data Factory y Azure Stream Analytics es que Azure Data Factory trabaja con una instantánea de datos que ya está disponible en un almacenamiento duradero, mientras que Azure Stream Analytics trabaja con datos transitorios en tiempo real que se procesan a medida que se van generando.

Existen dos versiones de Azure Data Factory: V1 y V2. La V2 se encuentra actualmente en fase preview y solo está disponible en las regiones Este de EE. UU. y Este de EE. UU. 2. La V1 es de dominio público y está disponible en las siguientes regiones: Este de EE. UU., Oeste de EE. UU., Centro-oeste de EE. UU. y Norte de Europa. A modo de buenas prácticas, los orígenes de datos de Azure deben estar ubicados idealmente en la misma región que Azure Data Factory.

La siguiente imagen ilustra la arquitectura de Azure Data Factory:



El servicio Data Factory consta de varios componentes.

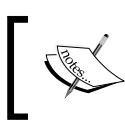
Orígenes de datos

En primer lugar, los datos deben introducirse en Data Factory. Puede tratarse de datos en el cloud u on-premises. Los datos deben moverse dentro de Azure Data Factory. Azure Data Factory proporciona servicios vinculados y construcciones de conjuntos de datos para conectar y recuperar datos en Data Factory.

Los conjuntos de datos se refieren a datos reales almacenados en el origen de datos o en la ubicación de datos. Puede tratarse de un archivos de valores separados por comas ubicado en el servidor on-premises o de un archivo ubicado en un contenedor de la cuenta de almacenamiento de Azure como blob o en Azure SQL. Las posibilidades son infinitas. Los servicios vinculados se refieren al proceso de conectar con el origen de datos mediante una cadena de conexión específica del origen de datos y utilizar esta última para recuperar datos en Data Factory. Por ejemplo, para copiar datos desde el archivo de blob de Azure a Azure SQL, se necesitan dos conjuntos de datos y dos servicios vinculados. Un servicio vinculado consistiría en la cadena de conexión a la cuenta de almacenamiento de Azure, mientras que el otro contendrá la cadena de conexión con Azure SQL. El conjunto de datos relativo al blob de Azure estaría compuesto por los archivos que contienen los datos, mientras que el conjunto de datos relativo a Azure SQL contendría las tablas en las que se escribirían los datos. El canal de datos utiliza actividades (por ejemplo, la actividad de copia) para conectar con los orígenes de datos mediante servicios vinculados y copiar los conjuntos de datos desde la ubicación de origen hasta Data Factory.

Data Factory consta de diversos canales, y cada canal se compone de actividades. Los canales se componen de diversas actividades, y cada actividad se encarga de ejecutar una determinada responsabilidad. Por ejemplo, la actividad de copia se encarga de copiar datos de una ubicación de datos (utilizando servicios vinculados y conjuntos de datos) a otra (utilizando servicios vinculados y conjuntos de datos de destino).

A continuación, se muestra la relación existente entre actividades, canales, servicios vinculados y conjuntos de datos.



Los tipos de conjuntos de datos disponibles en Data Factory están disponibles en <https://docs.microsoft.com/azure/data-factory/v1/data-factory-create-datasets#Type>.



Los canales de Azure Data Factory pueden ejecutarse manualmente o programarse para su ejecución periódica.

Transformación de datos

Una vez que los datos están disponibles en el almacén de datos, pueden procesarse utilizando servicios de proceso, tales como Azure HDInsight Hadoop, Spark, Data Lake Analytics y Azure Machine Learning. Los datos de salida resultantes de las transformaciones se pueden publicar para su consumo posterior.

Publicación y presentación

La salida resultante de la transformación puede almacenarse en un almacenamiento durable, como Cosmos DB, Azure SQL, el almacenamiento de datos de Azure o cualquier herramienta de análisis personalizada. Además, Power BI puede conectarse al almacenamiento durable para crear informes, paneles y vistas.

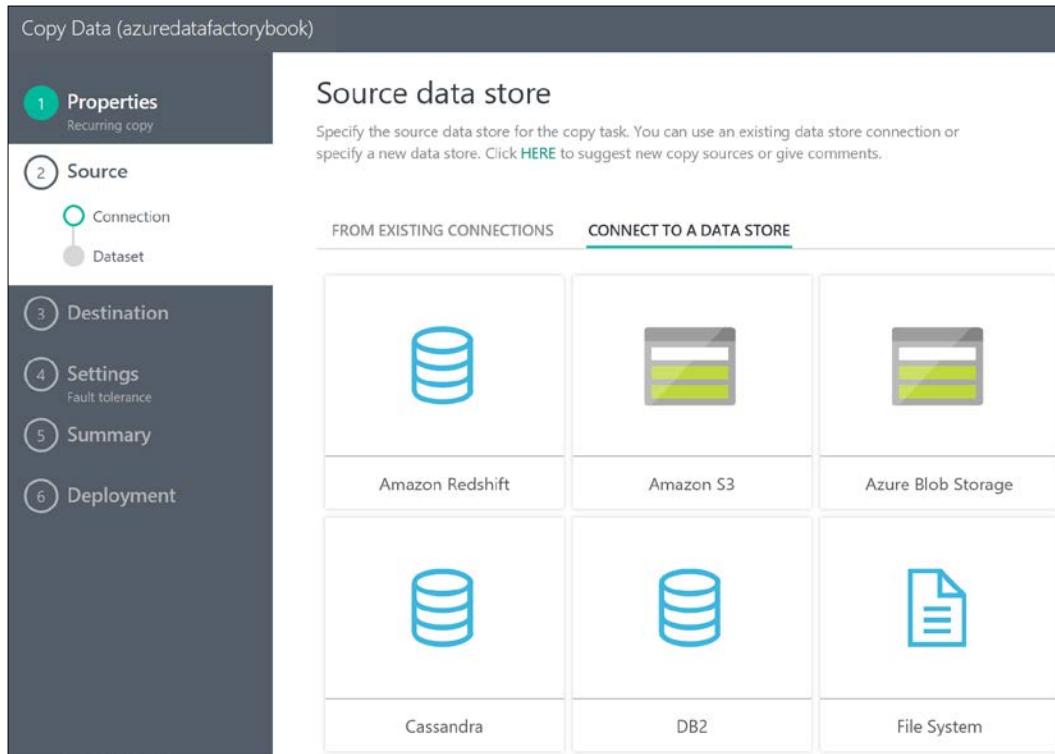
Uso de Data Factory

Esta sección proporciona instrucciones detalladas sobre cómo usar Azure Data Factory para crear conjuntos de datos, servicios vinculados y canales, así como para transferir datos de una cuenta de almacenamiento a otra.

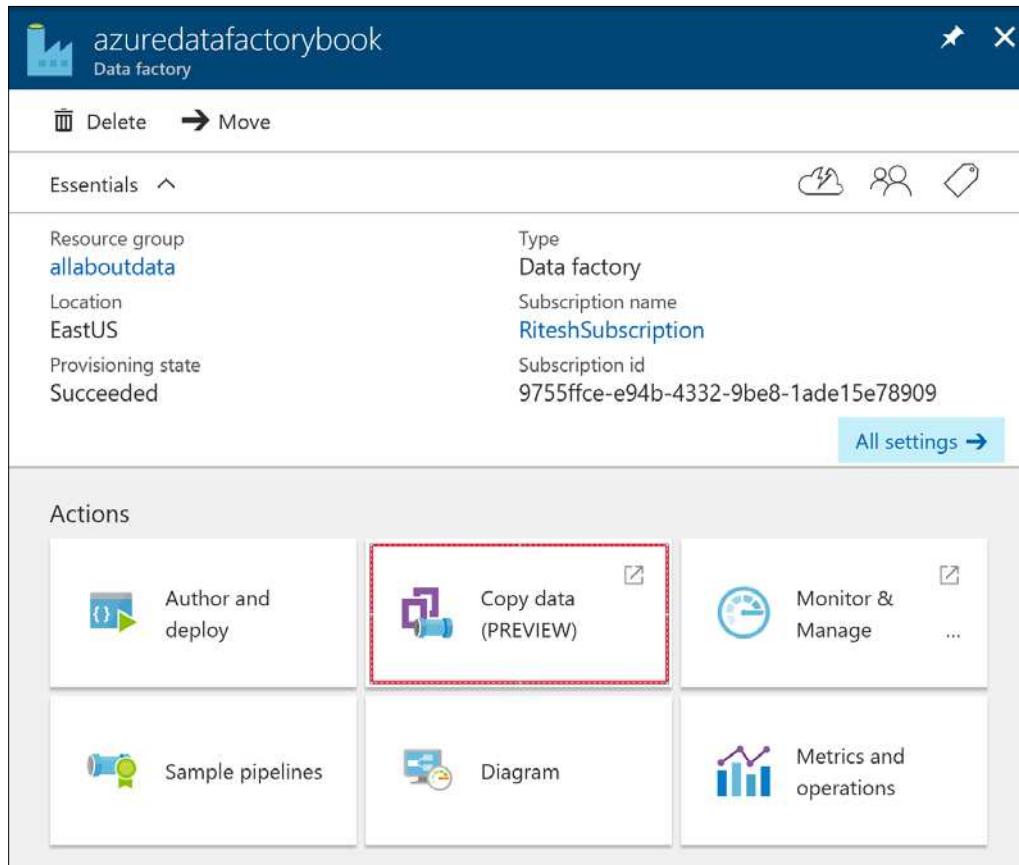
Data Factory proporciona numerosos almacenes a partir de los cuales se pueden leer y transferir datos. Data Factory ofrece una gran flexibilidad en cuanto a la elección de un almacén de datos de origen y de destino. La siguiente captura de pantalla muestra el número y tipo de almacenes de datos disponibles en Data Factory. Puede elegir cualquiera de ellos como origen y destino.

Amazon Redshift	Amazon S3	Azure Blob Storage	Azure Data Lake	Azure DocumentDB	Azure SQL Database	Azure SQL Data Warehouse	Azure Table Storage	Gestiones
DB2	Sistema de archivo	FTP	HDFS	HTTP	MongoDB	MySQL	Oracle	ODBC
Oracle	PostgreSQL	Salesforce	SAP BW	SAP HANA	SFTP	SQL Server	Sybase	Teradata

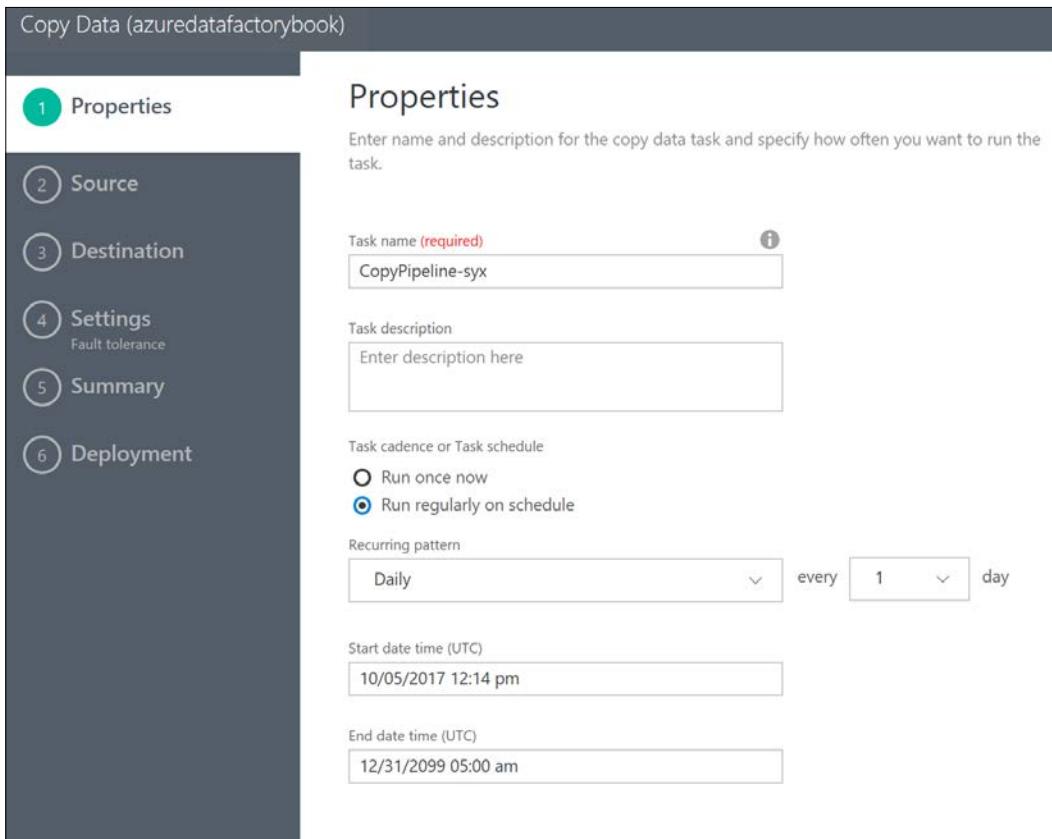
Azure Data Factory también proporciona una interfaz de asistente para identificar un almacén de datos de origen y sus datos, así como un almacén de datos de destino y sus datos. La siguiente captura de pantalla muestra lo mismo:



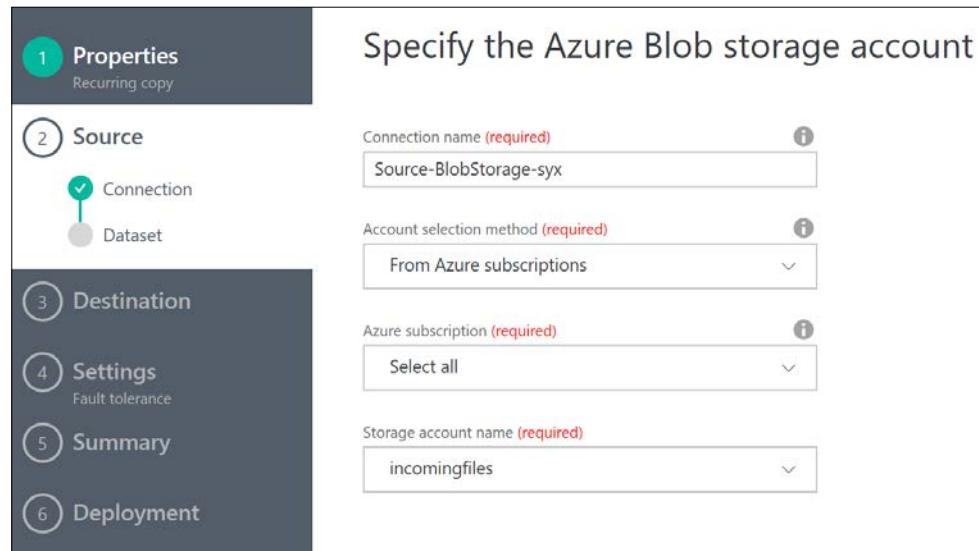
1. El primer paso consiste en crear un recurso de Data Factory. Una vez creado, haz clic en el botón **Copiar datos**:



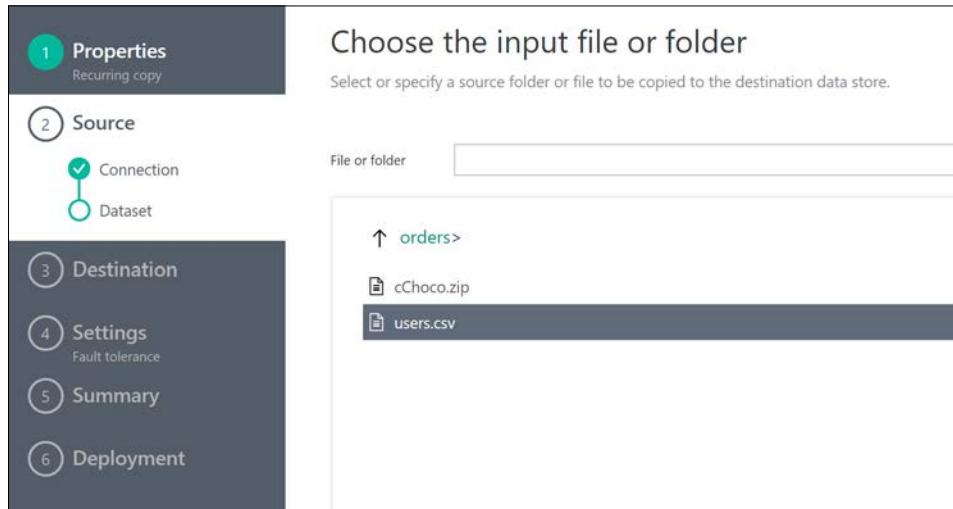
2. Esto abrirá una ventana del navegador desde la que deberás acceder a <https://datafactory.azure.com/>. Esta URL proporciona una interfaz de usuario basada en asistente que permite configurar Data Factory:



3. Rellena todos los datos, como el nombre de la tarea, la información de programación, el patrón de repetición, la fecha de inicio y la fecha de fin. Haz clic en **Siguiente**.
4. Selecciona **Azure Blob Storage** como almacén de datos de origen. Podría tratarse de cualquier origen de datos. En el siguiente paso, el asistente solicitará el valor para construir la cadena de conexión y los servicios vinculados para la conexión. Esto se muestra en la siguiente captura de pantalla. Haz clic en el botón **Siguiente**:

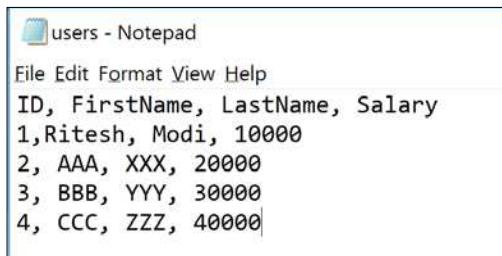


- En el siguiente paso se pide al usuario que seleccione el almacén de datos de origen. Esta actividad crea un conjunto de datos para Data Factory:



- El archivo `users.csv` es un archivo simple que contiene una serie de datos representativos; no obstante, en una situación real, podría haber disponibles millones, o miles de millones, de registros con terabytes de datos.

7. El archivo users.csv tiene el siguiente aspecto. Incluye un encabezado en la primera fila, mientras que las filas restantes contienen datos separados por comas. Elija el tipo de compresión si el archivo ya está comprimido.

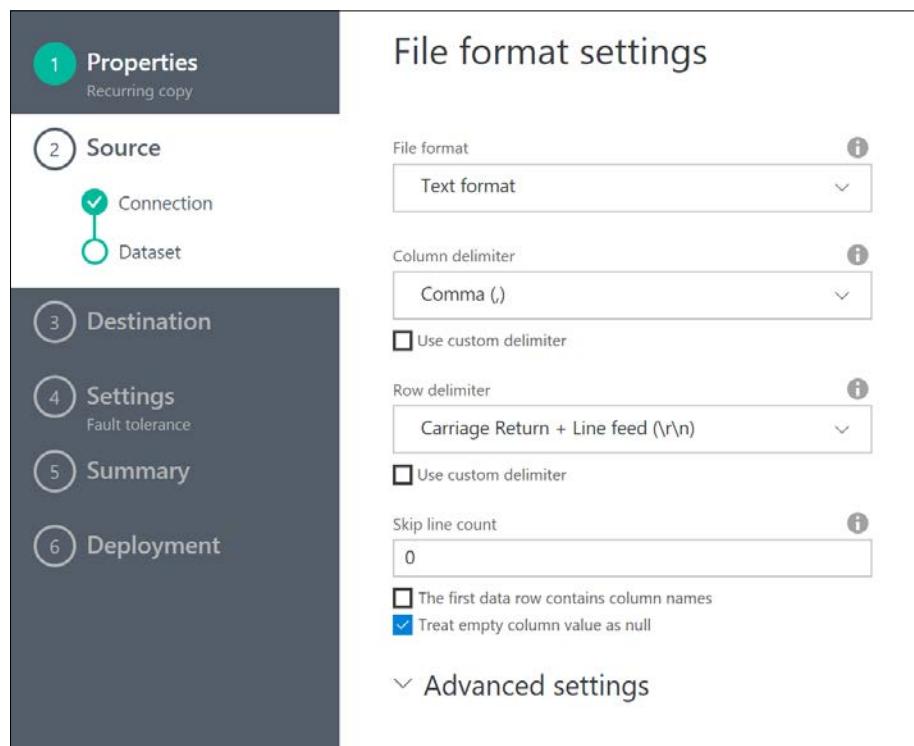


users - Notepad

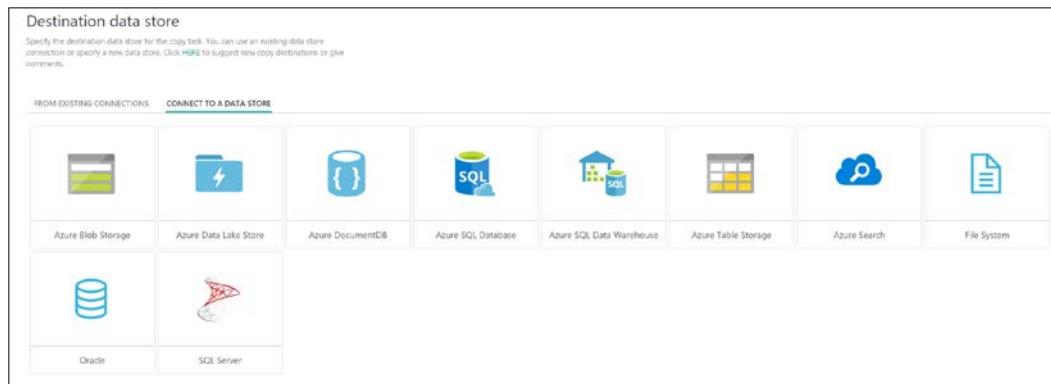
File Edit Format View Help

ID	FirstName	LastName	Salary
1	Ritesh	Modi	10000
2	AAA	XXX	20000
3	BBB	YYY	30000
4	CCC	ZZZ	40000

8. Al hacer clic en **Siguiente** se identifica el formato de archivo y se muestra al usuario. Aquí es donde el usuario puede cambiar la configuración relativa al formato de archivo. Esto se muestra en la siguiente captura de pantalla:



9. Al hacer clic en **Siguiente**, se solicitará la configuración del **Almacén de datos de destino**. Este es similar al almacén de datos de origen. Selecciona un almacén de datos de destino adecuado tal como se muestra a continuación:



10. En nuestro ejemplo, estamos seleccionando **Azure Blob Storage**, pero a los usuarios se les recomienda utilizar todo tipo de almacenes de datos, tales como **Almacén de Azure Data Lake**, **Azure DocumentDB**, **Azure SQL Data Warehouse**, **Azure Table Storage**, **Búsqueda de Azure**, **Oracle** y **SQL Server**.
11. Especifica los datos de la cadena de conexión del almacén de datos de destino tal como se muestra a continuación, y luego haz clic en **Siguiente**.

Specify the Azure Blob storage account

Connection name **(required)**
Destination-BlobStorage-syx

Account selection method **(required)**
From Azure subscriptions

Azure subscription **(required)**
Select all

Storage account name **(required)**
test2451835d

12. Indica los datos correspondientes al conjunto de datos de destino (como el nombre del contenedor y el nombre de archivo, tal como se muestra en la siguiente captura de pantalla) y haz clic en **Siguiente**:

Choose the output file or folder

Specify a folder that will contain output files or a specific output file in the destination data store.

Folder path

Filename

You can use variables in the folder path to copy data from a folder that is determined at runtime. Make sure that you select a folder with that structure using the Browse button first. The supported variables are: {year}, {month}, {day}, {hour}, {minute} and {custom}. See [Data Movement Activities](#) article for details about these variables.
Example: inputfolder/{year}/{month}/{day}.

Compression type i

Copy behavior i

13. Configura el ajuste **Formato de archivo** correspondiente al almacén de datos de destino, tal como se muestra en la siguiente captura de pantalla, y haz clic en **Siguiente:**

File format settings

File format i
Text format ▼

Column delimiter i
Comma (,) ▼

Use custom delimiter

Row delimiter i
Carriage Return + Line feed (\r\n) ▼

Use custom delimiter
 Add header to file i

^ Advanced settings

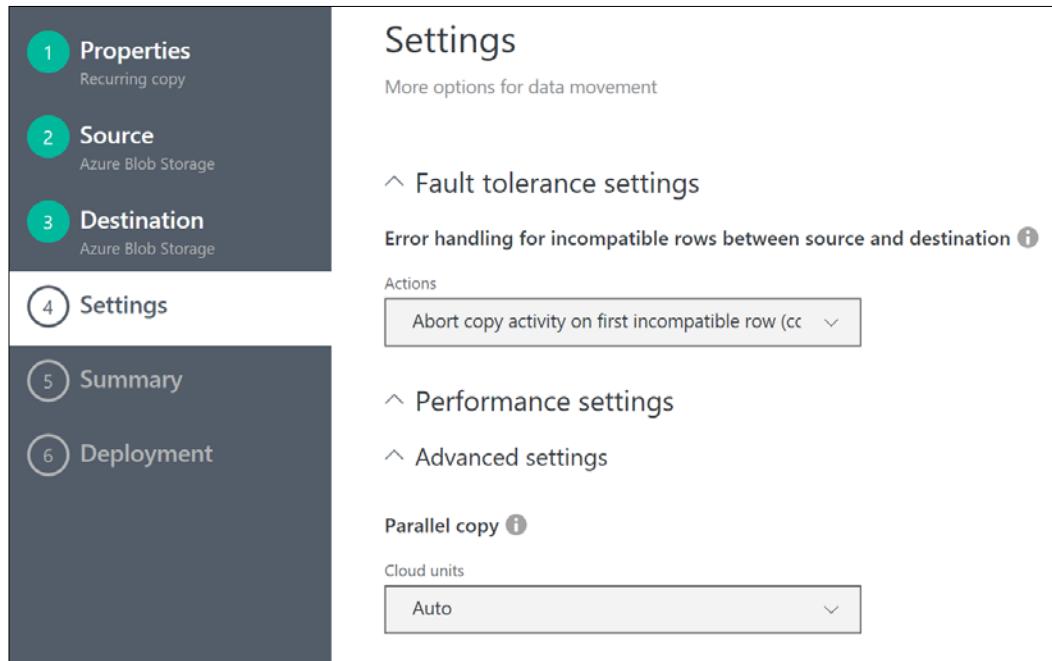
Escape character i

Quote character i

Null value i
\N

Encoding name i

14. Configura los ajustes de tolerancia y rendimiento (con escrituras paralelas) tal como se muestra a continuación y haz clic en **Siguiente**:



15. Valida todos los datos que se muestran en la ventana **Resumen**. Esto se muestra en la siguiente captura de pantalla. A continuación, haz clic en **Siguiente**:

Summary

You are running scheduled pipeline to copy data from Azure Blob Storage to Azure Blob Storage.

Azure Blob Storage
orders
Region: West Central US Copy Run Time Region: South Central US Azure Blob Storage
orders
Region: South Central US

Properties

Task name	CopyPipeline-syx	Edit
Task description	<no description has been provided>	
Task cadence	Daily, every 1 day between Thu, 05 Oct 2017 12:14:16 GMT and Thu, 31 Dec 2099 05:00:00 GMT	

Source

Connection	Account: incomingfiles	Edit
Connection name	Source-BlobStorage-syx	
Dataset name	InputDataset-syx	
Blob path	orders	
Region	West Central US	
Minimum size in MB		
Incremental updates	Not enabled	
Data delay	00:00:00	
Maximum retry		
Retry interval	00:00:00	
Retry timeout	00:00:00	

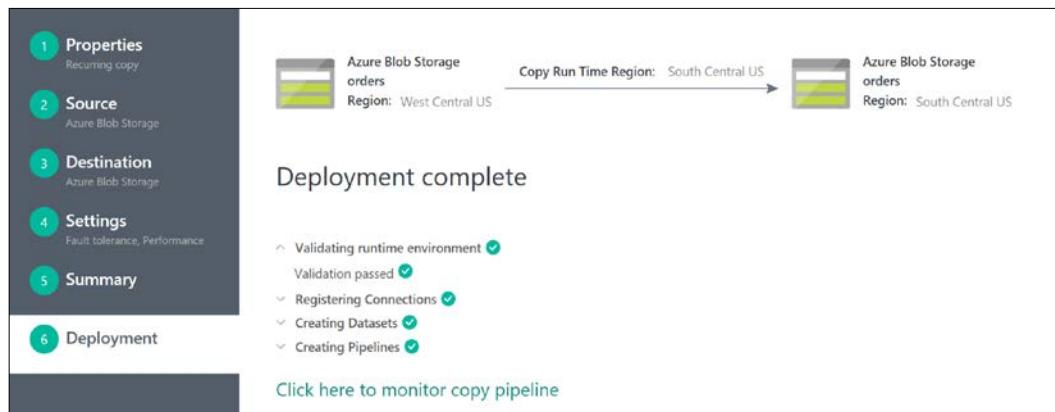
Destination

Connection	Account: test2451835d	Edit
Connection name	Destination-BlobStorage-syx	

[Previous](#) Next

16. Esto iniciará el proceso de implementación. La siguiente captura de pantalla muestra que la implementación se ha realizado con éxito y que los datos se han transferido desde Centro-oeste de EE. UU. a **Centro-sur de EE. UU.** desde un **almacenamiento de blob de Azure** a otro.

17. Si se ha configurado la programación previamente, esta será una actividad programada.



Azure Data Lake

Azure Data Lake es un servicio alojado de Azure que proporciona servicios de extremo a extremo para almacenar big data, transformarlos y realizar análisis en ellos. Se compone de varios servicios:

- Almacén de Azure Data Lake
- Azure Data Lake Analytics

Proporciona escalabilidad, rendimiento y seguridad de categoría empresarial para el almacenamiento y procesamiento de datos. Interviene en la ejecución masiva de cargas de trabajo paralelas y es un servicio muy solicitado para científicos de datos y analistas.

Almacén de Azure Data Lake

El almacén de Azure Data Lake proporciona almacenamiento para big data. Proporciona petabytes de almacenamiento y cuenta con un alto grado de optimización para almacenar datos de gran volumen, alta velocidad y diversos tipos y formatos. Se pueden almacenar conjuntamente datos de diversos formatos y tamaños en un mismo Data Lake.

Los almacenamientos convencionales, como sistemas de archivos, bases de datos relacionales y otros, son cuentas de almacenamiento de uso genérico no aptas para almacenar los big data a los que se accede con fines de análisis. El repositorio debe optimizarse para mejorar la rapidez y la eficiencia de las consultas, y debe poder almacenar tanto datos brutos como procesados. Azure Data Lake se ha creado específicamente para este propósito.

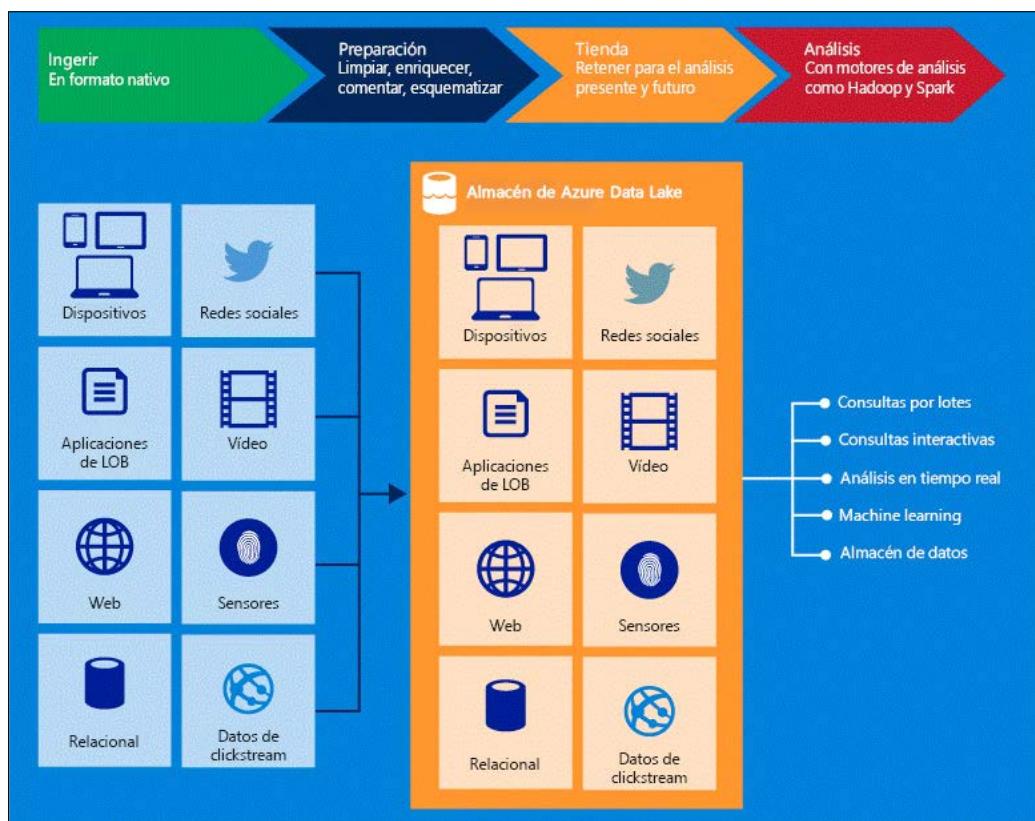
Se puede acceder a Azure Data Lake a través de diversas herramientas y servicios, incluidos HDInsight Hadoop, Azure Machine Learning, Stream Analytics, Data Factory, Event Hubs, etc.

Los datos proceden de diversos orígenes y dispositivos, tales como redes sociales, dispositivos IoT, aplicaciones de línea de negocio, bases de datos relacionales y no relacionales, extracción web, y muchos otros.

Actualmente solo está disponible en tres ubicaciones de Azure: Este de EE. UU. 2, Norte de Europa y Centro de EE. UU.

Al final, Azure Data Lake es un almacenamiento jerárquico basado en archivos, como un sistema de archivos de Windows, Linux o Hadoop, que se ha optimizado para dotar de mayor eficiencia a las consultas y el almacenamiento de cargas de trabajo de big data.

Los datos de Azure Data Lake ofrecen un alto grado de disponibilidad, ya que están replicados en varias instancias y almacenados en distintos centros de datos.



Data Lake Security

Los aspectos relativos a la seguridad son extremadamente importantes a la hora de manejar datos. Azure Data Lake es un repositorio seguro cuyo acceso administra Azure AD. Los usuarios, grupos y aplicaciones de servicio proporcionados y habilitados en Azure AD pueden acceder al almacén de Azure Data Lake. Asimismo, tras la autenticación, la identidad deberá disponer de los permisos necesarios para llevar a cabo actividades en Azure Data Lake Store. El **control de acceso basado en roles (RBAC)** de **Azure Resource Manager (ARM)** se encarga de aplicar la comprobación y autorización de permisos de una determinada identidad. Existe una seguridad adicional en las autorizaciones en la forma de **listas de control de acceso (ACL)**. Las ACL pueden aplicarse a permisos de lectura, escritura y ejecución de carpetas, subcarpetas y archivos.

Como sucede con Azure SQL, solo pueden acceder a Azure Data Lake Store aquellas direcciones IP explícitamente permitidas mediante su inclusión en listas blancas en la configuración del almacén de Azure Data Lake. Puesto que a Azure Data Lake suelen acceder con frecuencia otros servicios de Azure, existen opciones de configuración adicionales que permiten proporcionar acceso a estos servicios sin necesidad de conocer sus direcciones IP.

Los datos almacenados en el almacén de Azure Data Lake pueden cifrarse. Los datos en reposo pueden estar cifrados, de forma que nadie con acceso a ellos pueda leerlos. Como sucede con SQL Azure TDE, los datos del almacén de Azure Data Lake pueden cifrarse antes de su almacenamiento, y descifrarse de forma transparente tras su recuperación, sin que la aplicación tenga conocimiento de ello. Se considera una buena práctica conservar los datos en reposo en formato cifrado. Las claves de cifrado y descifrado se pueden almacenar en Azure Key Vault, el almacén de claves de Azure.

Rendimiento de Data Lakes

El rendimiento es otro aspecto extremadamente importante de las soluciones basadas en Data Lake Store. Puesto que Data Lake Store es un repositorio, debe proporcionar un almacenamiento de alta densidad y alto rendimiento en términos de mayor índice de ESPS para lecturas y escritura. Los almacenamientos convencionales no consumen la totalidad del rendimiento disponible en el almacenamiento y el hardware. En su lugar, el almacén de Azure Data Lake proporciona un mayor rendimiento establecimiento accesos paralelos a los datos y consumiendo el ancho de banda disponible del hardware.

Ten en cuenta que, en el caso de las soluciones de extremo a extremo, no solo es importante el rendimiento del almacén de Azure Data Lake; de hecho, también es importante el rendimiento de todos los orígenes de datos a partir de los cuales de ingieren los datos en el almacén de Azure Data Lake, así como de los servicios que acceden a los datos con fines analíticos.

Azure Data Lake Analytics

Azure Data Lake Analytics es uno de los servicios insignia de Azure para la ejecución de trabajos de análisis. El almacén de Azure Data Lake se encarga de almacenar big data de una forma optimizada. Sin embargo, en un contexto de extremo a extremo, los datos almacenados resultan útiles cuando los lee un servicio y dicho servicio puede realizar análisis en ellos y adquirir información significativa. El servicio de análisis entraña un procesamiento intensivo debido al procesamiento en profundidad de grandes cantidades de datos. Configurar un entorno para un servicio tal pasa por aprovisionar y configurar máquinas virtuales, ajustar su rendimiento y realizar su mantenimiento. Azure Data Lake Analytics proporciona capacidades de plataforma como servicio (PaaS) para la ejecución de trabajos de análisis. Esto significa que los usuarios ya no tienen que preocuparse por aprovisionar y configurar la infraestructura y el hardware y que, en su lugar, pueden concentrarse en escribir sus trabajos de análisis y centrarse en la problemática empresarial.

Azure Data Lake Analytics proporciona su propio lenguaje de tipo SQL, U-SQL, que tiene la capacidad de integrar código generalmente escrito con lenguajes de programación tales como C#. Se trata de un lenguaje versátil capaz de realizar consultas en diversos tipos de bases de datos, tales como Azure SQL, IaaS SQL Server y Azure SQL Data Warehouse.

Otra ventaja importante, aparte de su escaso mantenimiento, es que ofrece el modelo de pago por uso desde una perspectiva de coste. Los usuarios pagan solo cuando los trabajos de análisis están en ejecución, y en ningún otro caso. No se aplican cargos por el aprovisionamiento de Azure Data Lake Analytics si no hay análisis en ejecución.

Azure Data Lake Analytics es un servicio escalable. Significa que, cuando se ejecuta un trabajo en Azure Data Lake Analytics, este proporciona recursos de proceso y de memoria en segundo plano y continúa ampliándose horizontalmente de forma dinámica a medida que aumenta la demanda de recursos. Una vez finalizado el trabajo, los recursos se escalan en sentido descendente de forma automática. Todo esto sucede de forma transparente, sin que el usuario esté al tanto de ello.

Azure Data Lake Analytics, como Azure Date Lake Store, se integra con Azure AD con fines de autenticación. Los usuarios, grupos y aplicaciones de servicios disponibles en Azure AD pueden acceder a Azure Date Lake Analytics. El control de acceso basado en roles (RBAC) de ARM también se aplica en términos de autorización para determinar los permisos que se aplican a los usuarios y a Azure Date Lake Analytics.

Azure SQL Data Warehouse

Azure SQL es un servicio altamente escalable, durable, centrado en el rendimiento y de alta disponibilidad para datos transaccionales. Tradicionalmente, esto se ha denominado aplicaciones de **procesamiento de transacciones en línea (OLTP)**. Existe otro conjunto de aplicaciones popularmente conocidas como de **procesamiento de análisis en línea (OLAP)**. Por lo general, las aplicaciones OLAP se basan en un almacén de datos que almacena cantidades masivas de datos en un formato directamente utilizable por un motor de análisis, como **SQL Server Analysis Services (SSAS)**. El almacén de datos almacena los datos en forma no normalizada para mejorar la eficiencia de las consultas.

Los almacenes de datos se clasifican en grandes almacenes de datos de lectura. La escritura en estos almacenes de datos tiene lugar de forma periódica, al contrario de lo que sucede con las aplicaciones OLTP, donde las escrituras y lecturas son comparables. Por lo general, las aplicaciones que proporcionan servicios de informes, visualización y análisis se conectan a un almacén de datos para cargar y generar información a partir de este.

Los almacenes de datos también se caracterizan por su enorme necesidad de almacenamiento y potencia de proceso. Generalmente consisten en terabytes y petabytes de datos, y llevan a cabo la agregación y agrupación de millones y millones de registros.

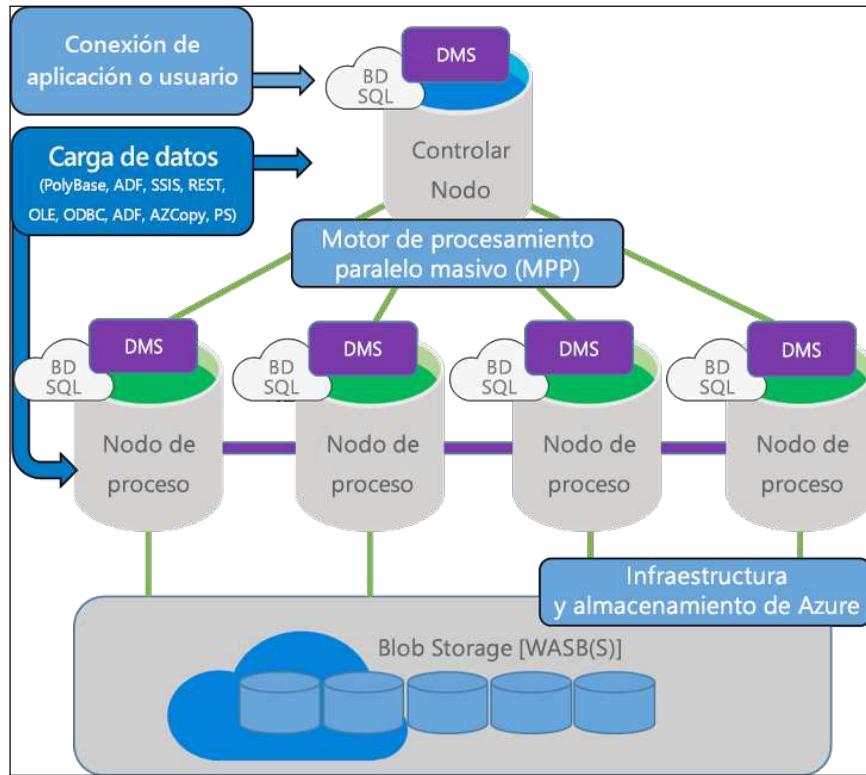
Azure proporciona SQL Data Warehouse como un servicio en el cloud increíblemente escalable, con un alto rendimiento ajustable, de alta disponibilidad y seguro. Está basado en Azure SQL Server y en sus bases de datos, por lo que se beneficia de todas sus ventajas y características. También almacena datos en los servicios de Azure Blob Storage, lo que contribuye a almacenar en ellos una cantidad prácticamente ilimitada de datos.

A continuación, se muestra la arquitectura de Azure SQL Data Warehouse. Azure SQL Data Warehouse consta de dos tipos de nodos:

- **Nodo de control:** este nodo es responsable de todas las interacciones con los usuarios y aplicaciones que interactúan con Azure SQL Data Warehouse. Es el cerebro principal y el motor que subyace a la ejecución de Azure SQL Data Warehouse. Su misión es coordinarse con todos los nodos de trabajadores o de proceso en cuanto a las necesidades de almacenamiento y de proceso. También ayuda a distribuir la carga entre ellos.
- **Nodos de proceso:** estos nodos son los verdaderos protagonistas de Azure SQL Data Warehouse. Son los que se encargan de almacenar y recuperar datos de Azure Blob Storage, y proporcionan potencia de proceso mediante la ejecución de instrucciones SQL relativas a los datos. Implementan una tecnología denominada **PolyBase** que ayuda a ejecutar instrucciones SQL relativas a datos almacenados en almacenes no relacionales, tales como el servicio Azure Blob Storage.

Tanto el nodo de control como el de proceso se basan en la base de datos de Azure SQL y se alojan en el servidor lógico SQL Server de Azure.

Existe otro componente importante denominado **Data Movement Service (DMS)** disponible en todos los nodos de Azure SQL Data Warehouse, con independencia de los nodos de control y de proceso. En realidad, se trata del aglutinante que permite el movimiento de datos, combinando y agregando datos de diversos nodos de proceso, y presentando datos uniformes a quien los solicita.



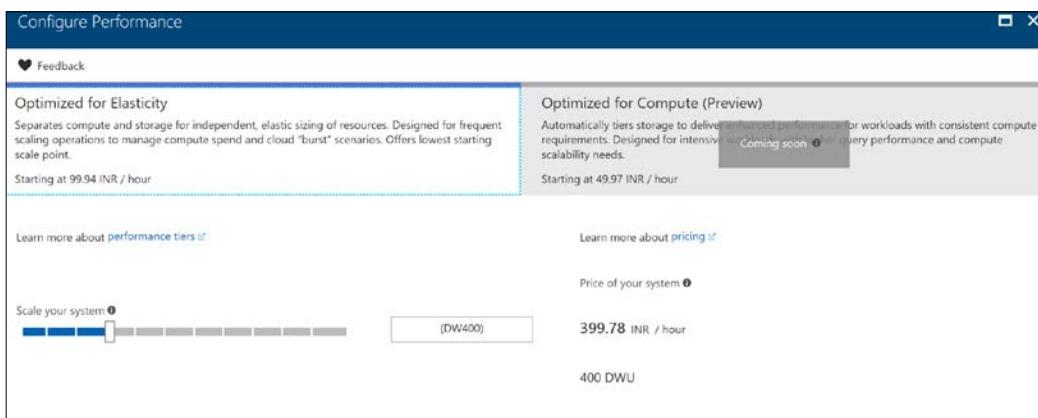
A continuación, se enumeran algunas de las características importantes de Azure SQL Data Warehouse:

- **Sistema de procesamiento paralelo masivo (MPP)** que ofrece características para implementar almacenes de datos paralelos.
- La interacción se basa en Transact-SQL. Capacidad de conectar mediante herramientas y controladores SQL.
- Disponible prácticamente en todas las ubicaciones de Azure para contribuir a la soberanía y gestión de los datos.
- Completamente elástico desde el punto de vista del almacenamiento. Almacenamiento prácticamente ilimitado disponible.
- La capa de proceso es distinta de la capa de almacenamiento. Esto contribuye a su escalabilidad por separado.
- Proporciona la capacidad y la potencia de consultas de un servidor SQL Server.

- Proporciona todas las ventajas de Azure SQL:
 - Lista de control de acceso de firewall basada en FIP
 - Escalabilidad y disponibilidad ininterrumpidas
 - Particionamiento horizontal
 - Agrupación con fines de optimización de costes
 - Seguridad

Un aspecto importante de Azure SQL Data Warehouse es la configuración de rendimiento ajustable. El rendimiento se define utilizando una **unidad de almacenamiento de datos (DWU)**. Cada DWU proporciona SLA en términos de proceso, ESPS y memoria. El coste de Azure SQL Data Warehouse es mayor cuanto mayor es la configuración de DWU. Esto es similar a las DTU que proporciona Azure SQL. Los usuarios pueden decidir las necesidades de rendimiento óptimas para su aplicación y ajustar el rendimiento en consecuencia. El rendimiento puede ajustarse de dos formas distintas. Son los siguientes:

- **Optimizado para la elasticidad:** según Azure, esto separa el proceso y el almacenamiento para poder ajustar el tamaño de los recursos de manera flexible e independiente. Está pensado para las operaciones de escalabilidad frecuentes, para administrar escenarios de consumo de proceso y ráfagas en el cloud. Ofrece el punto de partida de escalabilidad más bajo.
- **Optimizado para el proceso (preview):** próximamente.



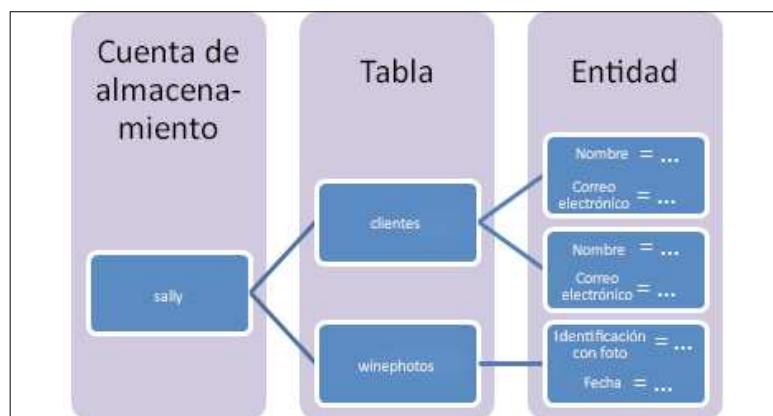
Azure SQL Data Warehouse puede integrarse en un host de servicios de Azure como:

- Machine Learning
- Power BI
- Stream Analytics
- Data Factory

Table Storage

El servicio Azure Table Storage está disponible como parte de la cuenta de almacenamiento de Azure. Azure Storage proporciona diversos tipos de almacenamiento, y Table Service es uno de ellos. Azure Table Service es un servicio de almacenamiento a escala de petabytes seguro, escalable, de alta disponibilidad y totalmente administrado que ayuda a almacenar datos en formato NoSQL. Como sabemos, NoSQL ayuda a crear diseños no esquematizados, y Azure Table Service también tiene un diseño no esquematizado. Ayuda a almacenar documentos utilizando pares de clave-atributo. Esto significa que Azure Table Service no impone ningún esquema a las entidades. Las entidades pueden tener distintas propiedades por entidad de forma simultánea sin modificación.

Los datos contenidos en las tablas de Azure se almacenan como entidades. Los datos se almacenan en tablas, donde las entidades constituyen las filas y las propiedades constituyen columnas individuales. Una entidad es muy similar a una fila de una base de datos relacional; cada servicio Table Service consta de varias entidades; y cada entidad consiste en un par de claves que ayudan a identificar únicamente a la entidad. A continuación, se muestra la relación existente entre Table Service, las entidades y el almacén de pares de clave-valor. Las referencias a las entidades se realizan mediante claves de partición y fila. Los valores de las claves de partición y fila deben proceder de los propios datos.



Las particiones deben concebirse como elementos que dividen una tabla en una estructura lógica para mejorar la velocidad y la sencillez de las consultas. Una partición se identifica de forma única mediante el nombre de la cuenta de almacenamiento, el nombre de la tabla y el nombre de la partición. Es muy importante que las claves de partición y fila se decidan correctamente. Pueden tener un tamaño máximo de 1 KB cada una. Deben combinarse de forma única en la tabla de forma que cualquier entidad pueda identificarse de forma unívoca, y también están indexadas. Las particiones ayudan a escalar las tablas en terabytes sin renunciar al rendimiento de las consultas. Es muy similar al particionamiento en Azure SQL.



Puedes obtener información detallada sobre el diseño de tablas en Azure en <https://docs.microsoft.com/azure/cosmos-db/table-storage-design-guide>.



Ten en cuenta que Table Service forma parte de la cuenta de almacenamiento. Cada cuenta de almacenamiento general tiene un límite máximo de 20.000 operaciones por segundo. Según los requisitos de las operaciones, esto podría suponer un uso muy inferior o muy superior al provisto. Si estos números de operaciones no son suficientes, se deberá considerar el uso de Cosmos DB. Otro aspecto es el límite de almacenamiento de Table Service, que se sitúa en 500 TB. En almacenamientos superiores a 500 TB, se debe considerar el uso de Cosmos DB.

Otra limitación de Table Service es que permite indexar únicamente las claves de partición y fila. Esto los hace menos eficientes a la hora de realizar consultas basadas en otras columnas de Table Service. Con Table Service, resulta difícil escribir consultas complejas con combinaciones.

Table Services almacena los datos como propiedades en formato de tabla, mientras que Cosmos DB almacena los datos como objetos de documento JSON, de forma que se puede indexar el objeto en su totalidad, en lugar de solo unas columnas.

Resumen

Los datos son uno de los elementos más importantes de una solución. Los datos tienen un valor y deben estar protegidos y seguros. La naturaleza de los datos ha cambiado en los últimos años. Los datos han aumentado exponencialmente en términos de volumen, y la velocidad de generación de datos también ha aumentado en gran medida. Los datos también se almacenan on-premises y en el cloud, en servicios de PaaS y de IaaS, y existe una gran cantidad de controladores y protocolos para acceder a ellos. Los datos pueden tener cualquier formato, por lo que el análisis y comprensión de datos de todos los formatos constituye un desafío en sí mismo. Azure proporciona servicios diversos para lidiar con todas estas nuevas facetas de los datos. Azure proporciona Azure SQL, SQL en IaaS, grupos elásticos, particionamiento, Data Factory, Stream Analytics, Data Lake Store y Analytics, Data Warehouse, Machine Learning y Power BI para crear soluciones de datos de extremo a extremo, desde la ingesta de los datos hasta su procesamiento, transformación, carga y análisis. En este capítulo se ha ofrecido una introducción a estos servicios de Azure, y se deberá profundizar en ellos utilizando la documentación de Azure.

En el siguiente capítulo hablaremos de uno de los patrones de implementación más emergentes en el cloud, denominado “sin servidor”.

8

Diseño e implementación de soluciones sin servidor

Actualmente, “sin servidor” es uno de los términos de moda más candentes en el ámbito de la tecnología y todos quieren subirse al carro. Sin servidor ofrece muchas ventajas en la informática general, procesos de desarrollo de software, infraestructura e implementación técnica. En el sector están sucediendo muchas: en un extremo del espectro está la infraestructura como servicio y en el otro extremo, sin servidor. Entre uno y otro, encontramos los contenedores y PaaS. He conocido a muchos desarrolladores y me parece que existe cierto nivel de confusión entre ellos sobre IaaS, PaaS, contenedores e informática sin servidor. Además, hay mucha confusión acerca de los casos de uso, la aplicación, la arquitectura y la implementación del paradigma sin servidor. Sin servidor es un nuevo paradigma que está cambiando no solo la tecnología, sino también la cultura y los procesos de las organizaciones.

Este capítulo se centrará en los siguientes temas relacionados con el aspecto “sin servidor”:

- ¿Qué es la arquitectura sin servidor?
- ¿Qué es Azure Functions?
- Cómo crear tu primera función de Azure
- Tipos de funciones en Azure
- Cómo crear una función conectada
- Cómo crear una función impulsada por eventos
- Cómo crear una solución que comprende varias funciones

En este capítulo, veremos en detalle lo siguiente:

- Facturación de Azure
- Facturación
- Uso y cuotas
- API de uso y facturación
- Calculadora de precios de Azure
- Prácticas recomendadas

Una breve historia sobre el tema "sin servidor"

Antes de entender de qué trata “sin servidor”, su arquitectura e implementación, es importante comprender su historia y cómo evolucionó.

Al principio había servidores físicos. Aunque los usuarios tenían control total sobre los servidores físicos, había muchas desventajas:

- Largo período de gestación entre el pedido y la implementación real del servidor
- Por naturaleza, capital intensivo
- Desperdicio de recursos
- Menor retorno de la inversión
- Difícil de escalar horizontal y verticalmente

Una evolución natural a los servidores físicos fue la virtualización. La virtualización hace referencia a la creación de máquinas virtuales sobre servidores físicos y la implementación de aplicaciones dentro de ellos. Las máquinas virtuales proporcionan ventajas en términos de lo siguiente:

- No es necesario adquirir hardware físico
- En comparación, es más fácil crear nuevas máquinas virtuales
- Aislamiento completo de los entornos
- Costes inferiores en comparación con los servidores físicos

Sin embargo, la virtualización tenía su propio conjunto de desventajas. Son las siguientes:

- Aún dependía de la adquisición física del servidor para el escalado después de varias instancias de máquinas virtuales.
- Seguía siendo costosa debido a la dependencia humana y de hardware.
- Despilfarro de recursos informáticos. Cada máquina virtual ejecuta un sistema operativo completo en ella.
- Alta dependencia y altos costes de mantenimiento.

La siguiente evolución fue IaaS de los proveedores de cloud. En lugar de procurar y administrar centros de datos e infraestructuras, la estrategia era crear máquinas virtuales, almacenamientos y redes en el cloud. El cloud proporciona servicios de infraestructura definidos por software y oculta todos los detalles relacionados con los servidores físicos, las redes y el almacenamiento. Esto tenía algunas ventajas:

- Sin gastos de capital. Solo gastos operativos.
- Sin tiempo de gestación para crear nuevas máquinas virtuales. Las nuevas máquinas virtuales se pueden aprovisionar en minutos en lugar de horas.
- Tamaño flexible de las máquinas virtuales.
- Escalado más fácil de máquinas virtuales.
- Completamente segura.

Las desventajas de las máquinas virtuales en el cloud son las siguientes:

- Se requiere una supervisión y auditoría activas.
- Es necesario un mantenimiento activo de las máquinas virtuales.
- Los usuarios deben administrar la escalabilidad, la alta disponibilidad y el rendimiento de las máquinas virtuales. Cualquier degradación y mejora posterior es responsabilidad del usuario.
- Se trata de una opción costosa porque los usuarios pagan por cada la máquina, tanto si se utiliza como si no.

El cloud también proporciona otro patrón para implementar aplicaciones, conocido popularmente como PaaS. PaaS proporciona abstracción de la infraestructura subyacente en términos de máquinas virtuales, redes virtuales y almacenamiento en el cloud. Proporciona una plataforma y un ecosistema donde los usuarios no necesitan saber nada acerca de la infraestructura; pueden traer e implementar su aplicación en estas plataformas. Existen ventajas precisas al utilizar PaaS en comparación con IaaS, pero todavía hay algunas opciones mejores. Las principales desventajas de PaaS son las siguientes:

- Las aplicaciones PaaS se implementan en máquinas virtuales tras bambalinas y el modelo de pago no es granular. Esto aún se encuentra en el nivel de implementación.
- Todavía exige supervisión para el escalado horizontal e interno.
- Los usuarios aún deben identificar los requisitos para su plataforma. Hay opciones limitadas disponibles para diferentes tipos de plataforma. Azure proporcionó el entorno de Windows durante mucho tiempo y desde hace poco Linux también está disponible. Además, la instalación de paquetes, utilidades y software es responsabilidad de sus usuarios.

También surgió un nuevo paradigma conocido como contenedores, que principalmente popularizó Docker. Los contenedores proporcionan un entorno ligero, aislado y seguro que tiene todos los beneficios de las máquinas virtuales, menos sus desventajas. No tienen un sistema operativo dedicado y, en cambio, dependen de un sistema operativo de servidor base. Los contenedores están disponibles en patrones de IaaS y PaaS. Los contenedores proporcionan una gran cantidad de ventajas:

- Aprovisionamiento más rápido de entornos
- Creación coherente y previsible de entornos
- Ayudan en la creación de arquitecturas de microservicios
- Ecosistema enriquecido con servicios avanzados de Kubernetes, Swarm y DC/OS

Las desventajas de los contenedores son las siguientes:

- Requieren una supervisión y auditoría activas.
- Requieren mantenimiento activo.
- Los usuarios o herramientas avanzadas, como Kubernetes deben administrar la escalabilidad, la alta disponibilidad y el rendimiento de las máquinas virtuales. Cualquier degradación y mejora posterior es responsabilidad del usuario.

Y esto lleva a “sin servidor” como el estado actual de la evolución para la implementación y el consumo de aplicaciones.

Sin servidor

Sin servidor se refiere a un modelo de implementación en el que los usuarios solo son responsables del código de su aplicación y la configuración. Los consumidores de sin servidor no tienen que preocuparse por la plataforma y la infraestructura subyacentes y pueden centrarse en resolver sus problemas empresariales escribiendo el código.

Sin servidor no significa que no haya servidores. El código y la configuración necesitan algún servidor para su ejecución. Sin servidor es desde la perspectiva del consumidor. Desde la perspectiva del consumidor, sin servidor significa que no ven el servidor en absoluto. No se preocupan de la plataforma ni de la infraestructura subyacentes. No deben administrarlas ni supervisarlas. Sin servidor proporciona un entorno que puede escalar en todos los sentidos, automáticamente, sin que los consumidores lo sepan. Todas las operaciones relacionadas con las plataformas y las infraestructuras suceden entre bambalinas, de manera transparente, sin que el usuario lo sepa. A los consumidores se les proporcionan **acuerdos de nivel de servicio (SLA)** de rendimiento y Azure se asegura de que cumpla con ese SLA independientemente del número o tamaño de los servidores requeridos.

Los consumidores solo tienen que introducir su código y el resto de los artefactos necesarios para ejecutar el código es responsabilidad del proveedor de cloud.

Principios de la tecnología sin servidor

La tecnología sin servidor se basa en los principios siguientes.

Menor coste

El coste se basa en el consumo real de los recursos de proceso y la potencia. Si no hay consumo, no hay coste asociado.

Controlada por eventos

Las funciones sin servidor deben poder ejecutarse en función de ciertos eventos que ocurren. El evento debe desencadenar la ejecución de la función. En otras palabras, sin servidor debería permitir que las funciones se desacoplen de otras funciones y, en cambio, confíen en la activación de ciertos eventos en los que están interesados.

Responsabilidad individual

Sin servidor debe implementar una única funcionalidad y responsabilidad y debería hacerlo bien. Las múltiples responsabilidades no deben codificarse ni implementarse en una sola función.

Rápida ejecución

Las funciones sin servidor no deben tardar mucho tiempo para completar un trabajo. Deben poder ejecutarse rápidamente y volver atrás.

Azure Functions o funciones como un servicio - FaaS

Azure proporciona funciones como un recurso. Son implementaciones sin servidor de Azure. Con Azure Functions, el código puede escribirse en cualquier idioma con el que el usuario se sienta cómodo y una función de Azure proporcionará un tiempo de ejecución para ejecutarlas. Según el idioma elegido, se proporciona una plataforma adecuada a los usuarios para que traigan su propio código. Son una unidad de implementación y pueden salir y entrar automáticamente. Cuando se trata de funciones, los usuarios no pueden ver la plataforma y las máquinas virtuales, pero Azure Functions proporciona una pequeña ventana para verlas en términos de la consola **Kudu**.

Azure Functions Runtime, enlaces y desencadenadores

Existen dos componentes principales de Azure Functions.

Tiempo de ejecución de la función de Azure

La base y el cerebro de Azure Functions es el tiempo de ejecución de Azure. Los trabajos web de Azure son el cursor previo a Azure Functions. El código para Azure WebJobs también constituye el núcleo de Azure Functions. Existen funciones y extensiones adicionales agregadas a Azure WebJobs para crear funciones de Azure. El tiempo de ejecución de la función es la magia que encontramos detrás de hacer que la función funcione. Las funciones de Azure están alojadas en Azure App Services. Azure App Services carga el tiempo de ejecución de Azure y espera que ocurra un evento externo o que se produzcan solicitudes HTTP. Al llegar una solicitud o la aparición de un desencadenador, se carga la carga útil entrante, se lee el archivo `function.json` de la función para encontrar los enlaces y el desencadenador de la función asigna los datos entrantes a los parámetros entrantes e invoca la función con valores de parámetros. Una vez que la función completa su ejecución, el valor se pasa nuevamente al tiempo de ejecución de la función de Azure mediante un parámetro saliente definido como vinculante en el archivo `function.json`. El tiempo de ejecución de la función devuelve los valores al autor de la llamada. El tiempo de ejecución de la función de Azure actúa como un pegamento que permite el funcionamiento completo de las funciones.

Enlaces y desencadenadores de funciones de Azure

Si el tiempo de ejecución de la función de Azure es el cerebro de Azure Functions, los enlaces y los desencadenadores de la función son el corazón de Azure Functions. Azure Functions promueve el acoplamiento flexible y la alta cohesión entre los servicios que utilizan desencadenadores y enlaces. En general, la aplicación implementa código utilizando la sintaxis imperativa para los parámetros entrantes y salientes y los valores de retorno. Esto generalmente resulta en la codificación de los parámetros entrantes. Dado que Azure Functions debería ser capaz de invocar cualquier función definida, implementa un mecanismo genérico para invocar funciones mediante desencadenadores y enlaces.

“Enlazar” se refiere al proceso de crear una conexión entre los datos entrantes y la función de Azure, asignando los tipos de datos. La conexión podría ser de una única dirección desde el tiempo de ejecución hasta la función de Azure, la función de Azure hasta el tiempo de ejecución para valores de retorno, o de varias direcciones: el mismo enlace puede transmitir datos entre el tiempo de ejecución de Azure y Azure Functions. Azure Functions utiliza una forma declarativa para definir enlaces.

Los desencadenadores son un tipo especial de enlace a través del cual se pueden invocar funciones basadas en eventos externos. Además de invocar la función, también se pasan los datos entrantes, la carga útil y los metadatos a la función.

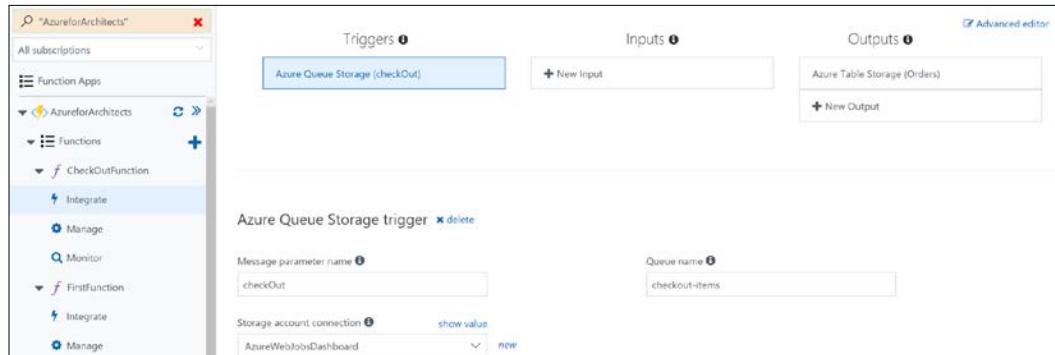
Los enlaces se definen en el archivo `function.json`:

```
{  
  "bindings": [  
    {  
      "name": "checkOut",  
      "type": "queueTrigger",  
      "direction": "in",  
      "queueName": "checkout-items",  
      "connection": "AzureWebJobsDashboard"  
    },  
    {  
      "name": "Orders",  
      "type": "table",  
      "direction": "out",  
      "tableName": "OrderDetails",  
      "connection": "<<Connection to table storage account>>"  
    }  
  "disabled": false  
}
```

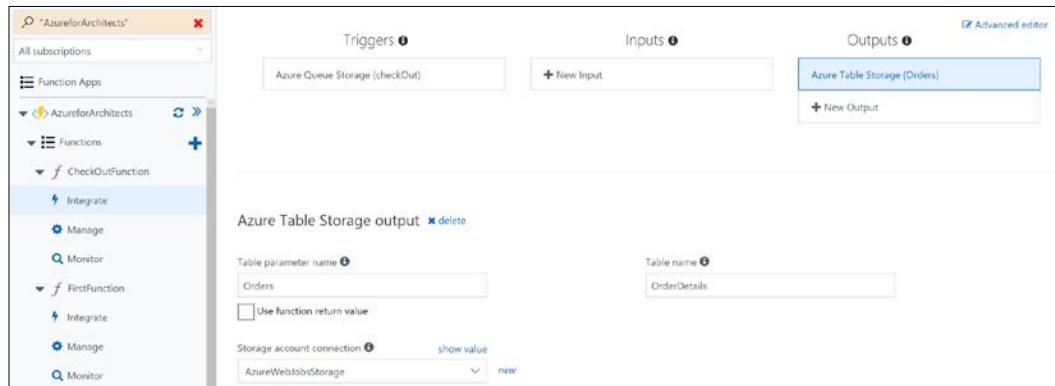
En este ejemplo, se declara un desencadenador que invoca la función cada vez que hay un elemento nuevo en la cola de almacenamiento. Type es `queueTrigger`, direction es `in`, queueName es `checkout-items` y también se muestran los detalles sobre la conexión de la cuenta de almacenamiento de destino y el nombre de la tabla. Todos estos valores son importantes para el funcionamiento de este enlace. El nombre `checkOut` se puede utilizar en el código de las funciones como una variable.

Del mismo modo, se declara un enlace para valores de retorno. Aquí el valor de retorno se denomina `Orders` y los datos son el resultado de la función de Azure. Escribe los datos de retorno en el almacenamiento de la tabla de Azure utilizando la cadena de conexión provista.

Tanto los enlaces como los desencadenadores se pueden modificar y crear mediante la pestaña **Integrar** en Azure Functions. El archivo `function.json` se actualiza entre bambalinas. Se declara el desencadenador `checkOut`, como se muestra a continuación:



A continuación, se muestran las órdenes de salida:



Los autores de Azure Functions no necesitan escribir código base para obtener datos de múltiples fuentes. Solo deciden el tipo de datos esperados del tiempo de ejecución de Azure. Esto se muestra en el siguiente segmento de código. Observe que la verificación está comprobación como una cadena para la función. Las funciones proporcionan varios tipos diferentes para poder enviar a una función. Por ejemplo, un enlace de cola puede proporcionar lo siguiente:

- **Objeto simple convencional (POCO)**
- Cadena
- Byte[]
- CloudQueueMessage

El autor de la función puede usar cualquiera de estos tipos de datos y el tiempo de ejecución de la función de Azure garantizará que se envíe un objeto adecuado como parámetro a la función:

```
using System;
public static void Run(string checkOut, TraceWriter log)
{
    log.Info($"C# Queue trigger function processed: { checkOut }");
}
```

También es importante saber que en las imágenes anteriores, los nombres de las cuentas de almacenamiento son `AzureWebJobsStorage` y `AzureWebJobsDashboard`. Son teclas que se definen en la configuración `appSettings` de Azure Functions.

Para obtener más información sobre los enlaces y desencadenador de Azure, consulte el siguiente enlace: <https://docs.microsoft.com/azure/azure-functions/functions-bindings-storage-queue>.

Proxies de la función de Azure

Los proxies de la función de Azure son una de las últimas funciones adicionales de Azure Functions. Después de comenzar a usar Azure Functions, habrá un momento en el que puede haber mucha implementación de funciones y será difícil integrarlas en un flujo de trabajo por parte de los clientes. En lugar de permitir que los clientes apliquen estas funciones, se pueden usar proxies de Azure. Los proxies ayudan a proporcionar a los clientes una única URL de función y luego, a su vez, invocan varias funciones de Azure entre bambalinas para completar los flujos de trabajo. Es importante comprender que los servidores proxy son aplicables en aquellos casos en que las funciones aceptan solicitudes a petición en lugar de ser controlados por eventos. Estas funciones internas conectadas a un proxy pueden estar dentro de una aplicación de función única o en varias aplicaciones separadas. Los proxies obtienen solicitudes de los clientes, convierten, anulan, aumentan la carga útil y las envían a las funciones internas del servidor. Una vez que obtienen una respuesta de estas funciones, pueden volver a convertir, anular y aumentar la respuesta y enviarla de vuelta al cliente.

Encontrará más información sobre Azure Functions en <https://docs.microsoft.com/azure/azure-functions/functions-proxies>.

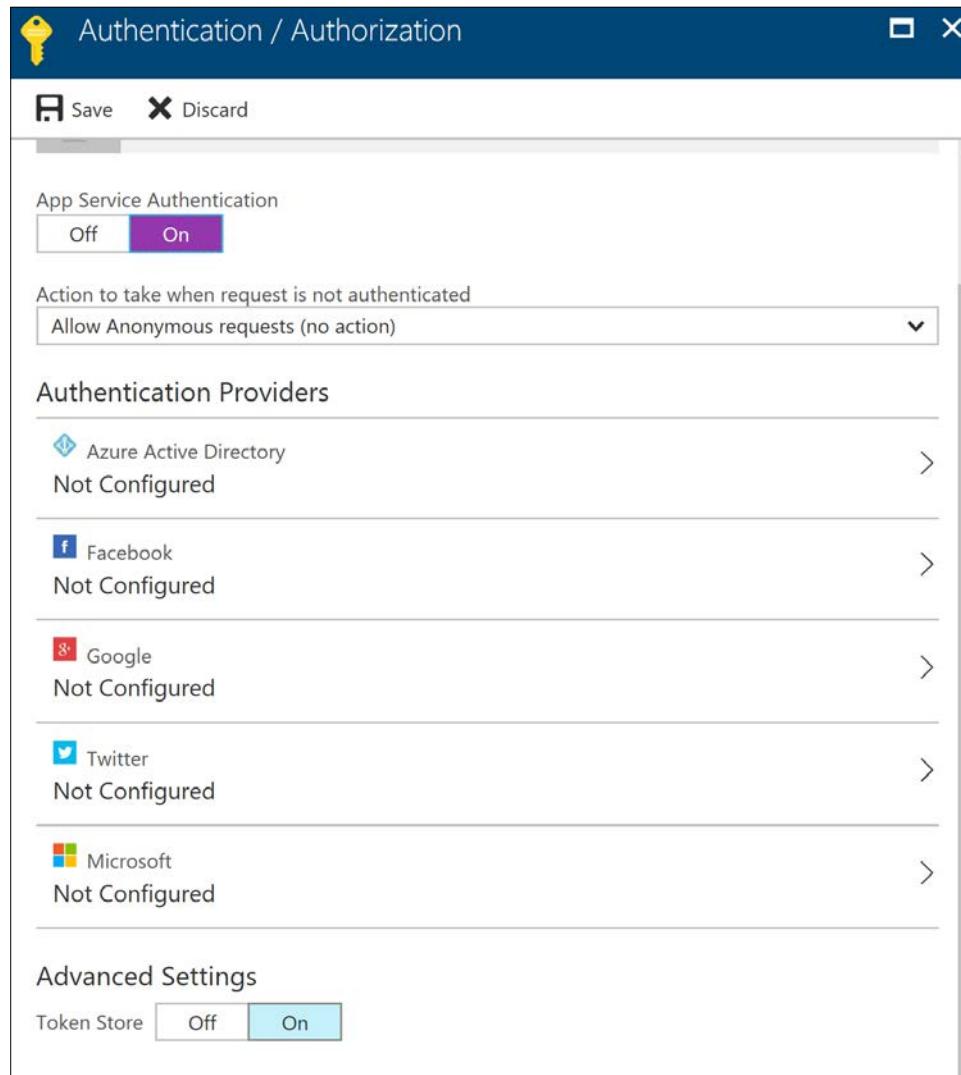
Supervisión

Se proporciona información de registro completa para cada solicitud o desencadenador en la pestaña **Supervisar** de Azure Functions. Esto ayuda a identificar problemas y auditar cualquier riesgo, error o excepción en Azure Functions.

The screenshot shows the Azure Functions monitoring interface for the 'FirstFunction' in the 'AzureforArchitects' app. On the left, there's a sidebar with navigation links for 'Integrate', 'Manage', and 'Monitor'. The 'Monitor' link is currently selected. The main area displays two summary boxes: 'Success count since Sep 1st' (1) and 'Error count since Sep 1st' (0). Below these are sections for 'Invocation log' and 'Invocation details'. The 'Invocation log' table shows one entry: 'FirstFunction (Method: GET, Uri: ...)' with a status of green checkmark, 'Details: Last ran (duration)' of '2 hours ago (1,572 ms)', and a 'live event stream' button. The 'Invocation details' section shows parameters: 'req' (Method: GET, Uri: https://azureforarchitects.azurewebsites.net/api/FirstFunction), '_log', '_binder', '_context' (ac42bcdd-ae89-40a8-a3cb-3cfbd615d356), '_logger', and '_return' (response). At the bottom right is a 'Logs' section with a large empty box.

Autenticación y autorización

Azure Functions se basa en Azure App Services para sus necesidades de autenticación. Los servicios de aplicaciones tienen funciones de autenticación enriquecidas en las que los clientes pueden usar OpenConnectID para la autenticación y OAuth para la autorización. Los usuarios pueden autenticarse utilizando las cuentas de Azure AD, Facebook, Google, Twitter o Microsoft.



Las funciones de Azure basadas en HTTP también pueden incorporar el uso de teclas que deben enviarse junto con las solicitudes HTTP. Estas teclas están disponibles en la pestaña **Administrar** de las funciones.

NAME	VALUE	ACTIONS
default	Click to show	Copy Renew Revoke

NAME	VALUE	ACTIONS
_master	Click to show	Copy Renew
default	Click to show	Copy Renew Revoke

Las teclas de las funciones permiten la autorización de funciones individuales. Estas teclas deben enviarse como parte del encabezado de solicitud HTTP.

Las teclas del host también permiten la autorización de todas las funciones dentro de una aplicación de funciones. Estas teclas deben enviarse como parte del encabezado de solicitud HTTP.

Las teclas predeterminadas se utilizan cuando se utilizan las teclas de función y del host. Hay una tecla del host adicional llamada `_master` que ayuda en el acceso administrativo a la API de tiempo de ejecución de la función de Azure.

Configuración de la función de Azure

Azure Functions proporciona opciones de configuración en varios niveles. Proporcionan configuración para:

- La propia plataforma
- Los servicios de la aplicación de funciones

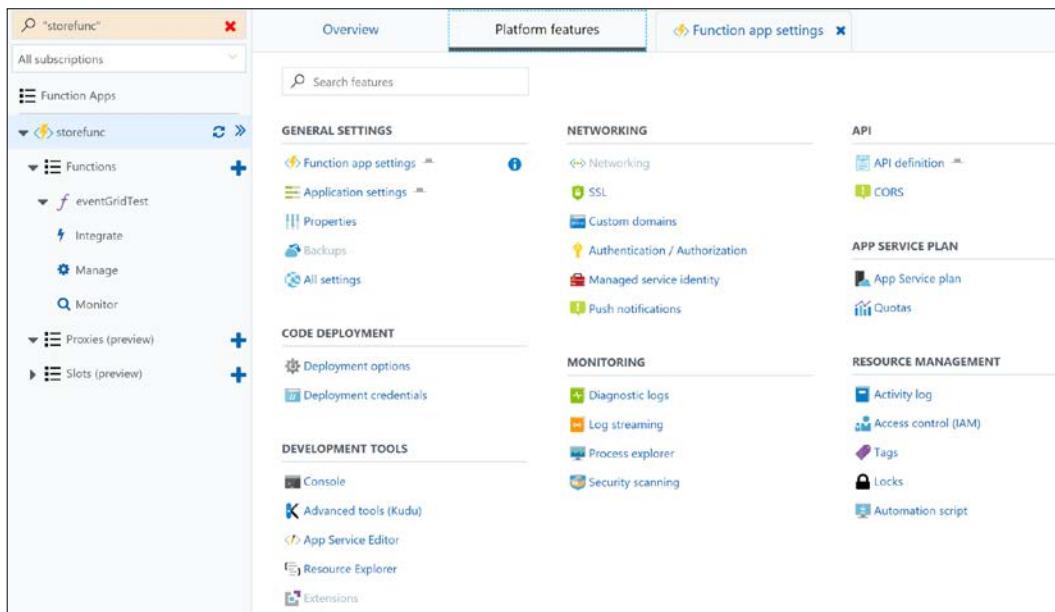
Estos ajustes afectan a todas las funciones que contienen. Encontrará más información sobre estos ajustes en <https://docs.microsoft.com/azure/azure-functions/functions-how-to-use-azure-function-app-settings>.

Configuración de la plataforma

Las funciones de Azure están alojadas en Azure App Services, de modo que obtienen todas sus características. Los registros de diagnóstico y supervisión se pueden configurar fácilmente con las características de la plataforma. Además, proporciona opciones para asignar certificados SSL, mediante un dominio, autenticación y autorización personalizados (mediante el uso de Azure AD, Facebook y otros protocolos OAuth y OpenConnectID) como parte de las funciones de red.

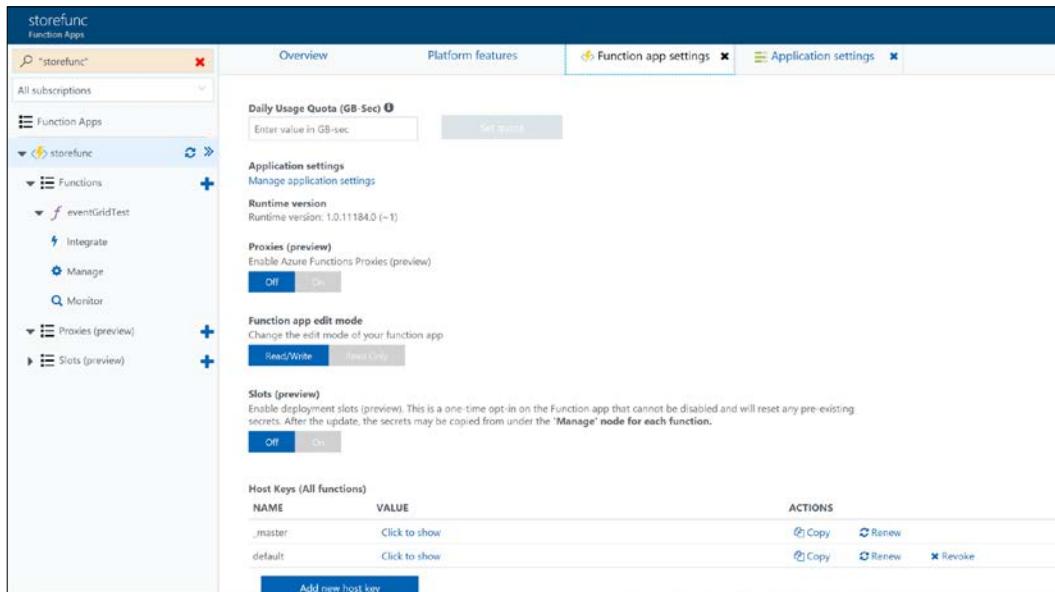
Aunque a los consumidores no les preocupa la infraestructura, el sistema operativo, el sistema de archivos y la plataforma en la que se ejecutan las funciones, la función de Azure proporciona las herramientas necesarias para mirar dentro del sistema subyacente para cambiarlas y modificarlas. La consola y la consola Kudu son herramientas que se utilizan con este objetivo. Proporcionan un editor Enriquecido para crear funciones de Azure y editar su configuración.

Azure Functions, al igual que los servicios de aplicaciones, permite almacenar información de configuración dentro de la sección de configuración de aplicación `web.config`, que se puede leer a petición.



Configuración de la función de servicios de aplicaciones

Esta configuración afecta a todas las funciones. La configuración de la aplicación se puede administrar aquí. Los proxies de la función de Azure se pueden habilitar y deshabilitar. Hablaremos sobre los proxies más adelante en este capítulo. También ayudan a cambiar el modo de edición de una aplicación de función y la implementación en ranuras.



Planes de coste de la función de Azure

Las funciones de Azure se basan en servicios de aplicación de Azure y proporcionan un modelo de costes mejor a los usuarios. Existen dos modelos de costes:

- **Plan de consumo:** se basa en el consumo real y la ejecución de funciones. Este plan calcula el coste en función del uso de proceso durante el consumo real y la ejecución de la función. Si una función no se ejecuta, no tiene ningún coste asociado. Sin embargo, esto no significa que el rendimiento se vea comprometido en este plan. Las funciones de Azure se escalarán automáticamente según la demanda para garantizar que se mantengan los niveles de rendimiento mínimos básicos. Se permite una ejecución de la función de 10 minutos para su finalización.

- **Plan del Servicio de aplicaciones:** este plan proporciona, entre bambalinas, máquinas virtuales dedicadas completas dedicadas a las funciones, por lo que el coste es directamente proporcional al coste de la máquina virtual y su tamaño. Hay un coste asociado con este plan, incluso si las funciones no se ejecutan. El código de la función puede ejecutarse durante el tiempo que sea necesario para la finalización. No hay límite de tiempo. Dentro del plan del Servicio de aplicaciones, el tiempo de ejecución de la función queda inactivo si no se utiliza en unos pocos minutos y se puede activar solo mediante un desencadenador HTTP. El ajuste Always On que se puede usar para no permitir que el tiempo de ejecución de la función quede inactivo. La escala es manual o se basa en la configuración de escala automática.

Ventajas de Azure Functions

La informática sin servidor es un paradigma relativamente nuevo en la tecnología de servidores y ayuda a las organizaciones a convertir grandes funcionalidades en pequeñas funciones discretas a demanda que pueden invocarse y ejecutarse a través de desencadenadores automáticos y trabajos programados. También se denominan FaaS, en el que las organizaciones pueden centrarse en los desafíos de su dominio en lugar de la infraestructura y plataforma subyacentes. También ayuda a democratizar la arquitectura de soluciones en funciones reutilizables más pequeñas que obtienen un mayor valor para sus inversiones.

Hay una gran cantidad de plataformas de proceso sin servidor disponibles en Internet. Algunas de las más importantes son:

- Azure Functions
- AWS Lambda
- IBM OpenWhisk
- Iron.io
- Funciones de Google Cloud

De hecho, cada pocos días se introduce un nuevo marco y cada vez es más difícil para las empresas decidir y elegir el marco que mejor se adapte a ellos.

Azure proporciona un entorno Enriquecido sin servidor conocido como Azure Functions y me gustaría señalar algunas características que admite para que estén listas para la empresa:

- **Gran variedad de desencadenadores:** los desencadenadores actúan como eventos y ejecutan funciones de proceso sin servidor automáticamente. Estos desencadenadores pasan los datos sin procesar, la acción y el evento importantes para las funciones. La plataforma de proceso sin servidor debe proporcionar diferentes tipos de desencadenadores que permitan a las empresas varias opciones para la arquitectura de su solución. También debe proporcionar facilidades para ejecutar las funciones periódicamente como una tarea programada.

- **Webhooks:** un webhook ayuda en la integración avanzada con servicios de terceros. Coloca enlaces para que cualquier cambio en un servicio de terceros comience a ejecutar las funciones de proceso sin servidor. Proporciona una infraestructura enriquecida que puede colaborar con servicios importantes como los sistemas de control de versiones y las plataformas de redes sociales.
- **Ejecución a petición:** las funciones no solo deben ejecutarse en función de los desencadenadores, sino que también deben tener un mecanismo para ejecutarse manualmente. Esto aporta flexibilidad durante el desarrollo y la prueba de las funciones.
- **Escalabilidad ilimitada:** una plataforma de procesos sin servidor debe escalar funciones de forma dinámica según su demanda. Normalmente, las funciones de proceso sin servidor son accesibles a través de Internet y es difícil predecir la cantidad de usuarios que podrían conectarse a ellas. Es muy importante que se amplíen automáticamente con más instancias durante una alta demanda y que disminuyan cuando la demanda se reduce. Esto ayudaría a las empresas a reducir los gastos de capital en una infraestructura que permanece inactiva durante gran parte del año.
- **Alta disponibilidad:** resulta evidente. Las funciones sin servidor deben tener una alta disponibilidad. A las empresas no les gusta el tiempo de inactividad no planificado y es extremadamente importante que las funciones sin servidor estén siempre disponibles (24x7x365). Si una plataforma de proceso sin servidor no garantiza más de cinco nueves en términos de disponibilidad (99,999 %), las empresas deben buscar otra plataforma.
- **Seguridad:** la plataforma debe ofrecer una infraestructura adecuada para proporcionar seguridad a las funciones sin servidor. Debe haber provisión para autenticar, autorizar y mantener la confidencialidad y la identidad de los usuarios. Además, la plataforma debe proporcionar aprovisionamiento para autenticar utilizando los protocolos de autorización basados en OAuth de esta nueva era. La plataforma también debe permitir el **uso compartido de recursos entre orígenes (CORS)**. CORS permite que el código JavaScript que se ejecuta en un navegador en un host externo interactúe con funciones sin servidor para funciones accesibles a través de Internet.
- **Estándares de la industria:** la plataforma debe utilizar estándares de la industria en lugar de tecnología patentada. JSON se está convirtiendo en el estándar de facto para el intercambio de carga útil de mensajes en puntos de conexión REST. Debe admitir los protocolos HTTP y HTTPS. Esto permite a las aplicaciones multiplataforma invocar y consumir estos servicios.

- **Supervisión, registros y auditoría:** cualquier plataforma sin servidor debe proporcionar infraestructura para supervisar funciones en el nivel operativo, generar alertas e informar a los equipos para que tomen medidas correctivas sobre ellas. La plataforma debe proporcionar un mecanismo de registro Enriquecido para capturar actividades en funciones y proporcionar métricas de telemetría a su propietario. Esto debe incluir detalles de disponibilidad, escalabilidad y rendimiento, entre otros. Los registros deben ser fácilmente accesibles en forma de paneles, lo que facilita el ejercicio de auditoría para los administradores. Las empresas desean optimizar el tiempo, el esfuerzo y el dinero gastado para mantener operativas las funciones sin servidor. Deben elegir una plataforma sin servidor que proporcione información operativa Enriquecida a través de las capacidades de supervisión, registro y auditoría.
- **Apto para DevOps:** las empresas han alcanzado un gran momento con DevOps. Las grandes empresas ya han adoptado DevOps; las pequeñas y medianas empresas no se quedan atrás. Todas las empresas desean implementar tecnologías en su entorno y ecosistema que sean automatizables a partir de procesos continuos de integración, implementación y entrega. La plataforma no solo debe proporcionar los puntos de conexión y las utilidades de automatización, sino que también debería ser fácil para los desarrolladores configurar las prácticas relacionadas con DevOps para el desarrollo de funciones.
- **Compatibilidad lingüística:** una plataforma sin servidor debe proporcionar varias opciones de idioma para el desarrollo de funciones. Se acabaron los días en que las empresas utilizaban un solo idioma popular para todas sus actividades de desarrollo. Hoy en día, las empresas implementan idiomas bien conocidos por sus desarrolladores. También utilizan idiomas heterogéneos en su entorno para una mayor productividad y afinidad con el tipo de aplicación que se desarrolla. Puede ser C#, JavaScript, Bash, PowerShell, o una combinación de estas tecnologías.
- **Herramientas avanzadas para la creación, las pruebas y la implementación:** las herramientas deberían facilitar el desarrollo, las pruebas y la depuración para el desarrollador de funciones. Además de proporcionar asistentes de inicio rápido, las herramientas deben aumentar la productividad del desarrollador al proporcionar un entorno integrado para desarrollar y depurar, ejecutar pruebas de unidad e integración, la capacidad de simular objetos y ayudar en implementaciones automatizadas. El editor de creación debe proporcionar plantillas y código de Scaffold para acelerar el desarrollo. También debe proporcionar emuladores que ayuden a probar la función sin alojar en el entorno real.
- **Tolerancia a fallos:** las funciones sin servidor no deben fallar o morir debido a errores o excepciones en el código de usuario o en la infraestructura del servidor. Deben registrar las excepciones, informar al usuario y continuar ejecutando elegantemente la siguiente solicitud.

- **Actividades de ejecución de larga duración:** una plataforma sin servidor debe proporcionar flexibilidad para ejecutar actividades de larga duración. Después de que las actividades de ejecución de larga duración completen su ejecución, la plataforma sin servidor debe recoger la ejecución y devolverla al usuario.
- **Posibilidad de configurar:** si bien un cálculo sin servidor significa que no hay acceso a la plataforma subyacente, debe haber puntos de conexión adecuados a través de los cuales los administradores puedan mirar y cambiar la configuración de la plataforma subyacente. Esto ayudará a las empresas a solucionar problemas, realizar implementaciones avanzadas y ajustar la plataforma según sus necesidades.
- **Establecer cuotas:** la plataforma de proceso sin servidor debe tener flexibilidad para determinar las cuotas de ejecución. Esto ayudaría a restringir el uso sin sentido de un proceso sin servidor.
- **Varios planes de hosting:** la plataforma debe proporcionar opciones para varios planes de uso dependiendo del consumo de recursos y las ejecuciones. Las empresas pueden elegir el plan que mejor se adapte a sus necesidades y optimizar sus gastos al mismo tiempo que garantizan beneficios en los costes.
- **Integraciones enriquecidas:** la plataforma de proceso sin servidor debe poder integrarse con otros servicios, tales como servicios cognitivos, servicios de análisis, servicios de comunicación y otros servicios basados en transacciones, para proporcionar servicios contextuales y enriquecidos a los consumidores. Esto ayudará a las empresas a crear soluciones que sean innovadoras y relevantes para la industria.
- **Paralelismo masivo:** la plataforma de servidor debe ser capaz de ejecutar peticiones de forma sincrónica y asincrónica. Finalmente, esto ayuda a escribir patrones que pueden ayudar en la escalabilidad masiva de funciones al utilizar de manera óptima los recursos de la plataforma.
- **Configuraciones de varios niveles:** la plataforma debe proporcionar opciones para configurar en el nivel de función individual, nivel de agrupación de funciones y nivel de plataforma. Esto ayudará a las empresas a cambiar el comportamiento general de sus funciones, incluido el comportamiento de la plataforma, con el mínimo esfuerzo.

Casos de uso de Azure Functions

Hay muchos casos de uso válidos para usar e implementar Azure Functions.

Implementación de microservicios

Azure Functions ayuda a dividir las aplicaciones grandes en unidades de código funcional más pequeñas y discretas. Cada unidad se trata de forma independiente de la otra y evoluciona en su propio ciclo de vida. Cada una de estas unidades de código tiene sus propios requisitos de proceso, hardware y supervisión. Cada función se puede conectar a todas las demás funciones. Estas unidades están entrelazadas por orquestadores para construir una funcionalidad completa. Por ejemplo, en una aplicación de comercio electrónico, puede haber funciones individuales (unidades de código) para cada responsable para enumerar catálogos, recomendaciones, categorías, subcategorías, carritos de compra, pagos, tipos de pago, pasarelas de pago, direcciones de envío, direcciones de facturación, impuestos, gastos de envío, cancelaciones, devoluciones, correos electrónicos, SMS, etc. Algunas de estas funciones se unen para crear casos de uso para aplicaciones de comercio electrónico, como la navegación de productos, el flujo de pago, etc.

Integración entre varios puntos de conexión

Azure Functions puede crear una funcionalidad general de la aplicación mediante la integración de varias funciones. La integración puede basarse en la activación de eventos o podría ser sobre una base push. Ayuda a descomponer grandes aplicaciones monolíticas en pequeños componentes.

Procesamiento de datos

Azure Functions se puede usar para procesar datos entrantes en lotes. Es útil para procesar datos que vienen en varios formatos, por ejemplo XML, CSV, JSON, texto, etc. También pueden ejecutar algoritmos de conversión, enriquecimiento, limpieza y filtrado. De hecho, se pueden usar varias funciones, cada una de las cuales realiza conversión o enriquecimiento, limpieza o filtrado. Azure Functions también se puede usar para incorporar servicios cognitivos avanzados como el reconocimiento óptico de caracteres (OCR), la visión de proceso, la manipulación de imágenes y la conversión.

Integración de aplicaciones heredadas

Azure Functions puede ayudar a integrar aplicaciones heredadas con protocolos más nuevos y aplicaciones modernas. Las aplicaciones heredadas pueden no estar utilizando protocolos y formatos estándar de la industria. Azure Functions puede actuar como un proxy para estas aplicaciones heredadas, aceptar solicitudes de los usuarios u otras aplicaciones, convertir los datos en un formato comprendido por una aplicación heredada y hablar sobre los protocolos que comprende. Esto abre un mundo de oportunidades para integrar y traer aplicaciones antiguas y heredadas a la cartera principal.

Trabajos programados

Azure Functions se puede utilizar para ejecutarse de forma continua y periódicamente para algunas funciones de la aplicación. Estas funciones de la aplicación pueden ir desde la realización periódica de copias de seguridad, restauración, ejecución de trabajos por lotes, exportación e importación de datos, correos electrónicos masivos, etc.

Gateways de comunicación

Azure Functions se puede utilizar en gateways de comunicación, como el uso de hubs de notificaciones, servicio de mensajería (SMS) y correo electrónico.

Tipos de funciones en Azure

Azure Functions se puede clasificar en tres tipos diferentes:

- **Funciones a petición:** son funciones que se ejecutan cuando se llaman o invocan explícitamente. Ejemplos de tales funciones incluyen funciones basadas en HTTP y webhooks.
- **Funciones programadas:** estas funciones son como trabajos de temporizador y ejecutan funciones en intervalos fijos.
- **Funciones basadas en eventos:** estas funciones se ejecutan en base a eventos externos. Por ejemplo, cargar un archivo nuevo en el almacenamiento de blobs de Azure genera un evento que puede iniciar la ejecución de la función de Azure.

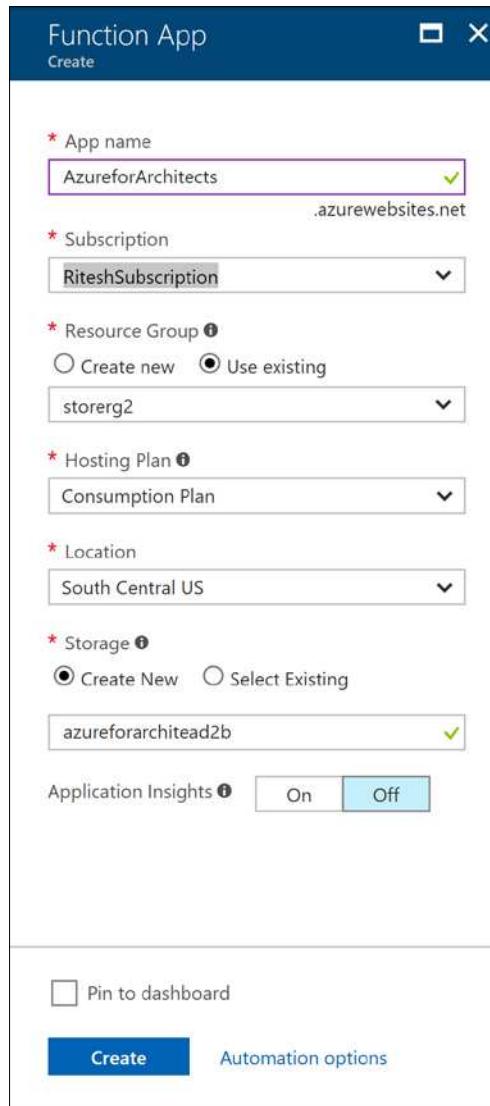
Cómo crear tu primera función de Azure

Las funciones de Azure se pueden crear utilizando el portal de Azure, PowerShell, Azure CLI y las API de REST. Los pasos para crear una función utilizando la plantilla de ARM ya están detallados en <https://docs.microsoft.com/azure/azure-functions/functions-infrastructure-as-code>. En esta sección, las funciones de Azure se aprovisionarán utilizando el portal.

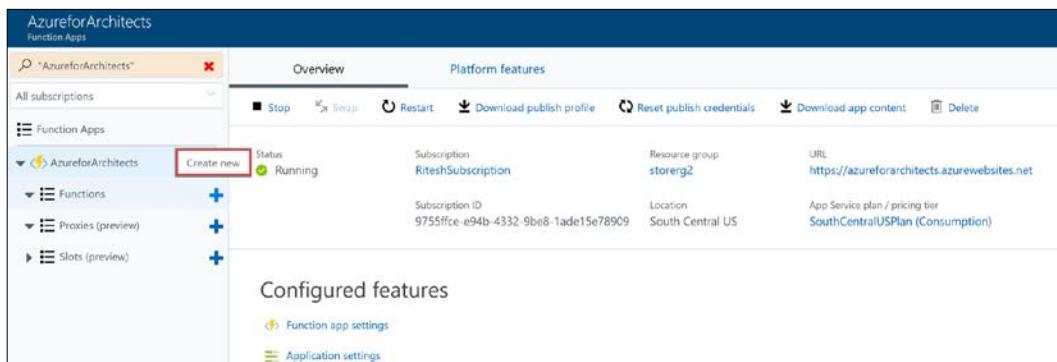
Las funciones de Azure están alojadas en Azure App Services. Los usuarios crean una nueva aplicación de funciones que a su vez crea un plan de servicio de aplicaciones y un servicio de aplicaciones. El plan de servicio de aplicaciones se configura según lo siguiente:

- **Nombre:** nombre del servicio de aplicaciones. El nombre debe ser único en el dominio .azurewebsites.net.
- **Ubicación:** ubicación para alojar el servicio de aplicaciones de la función de Azure.

- **Plan de hosting:** también conocido como plan de precios. Aquí, dos opciones, como se mencionó anteriormente, están disponibles: plan de servicios de aplicaciones y consumo.
- **Nombre del grupo de recursos:** el nombre del grupo de recursos que contiene tanto el plan de servicios de aplicaciones como el servicio de aplicaciones.
- **Cuenta de almacenamiento:** Azure Functions necesita una cuenta de Azure Storage para almacenar sus datos y registros internos.
- **Habilitar información de aplicación:** para capturar información de telemetría de Azure Functions.



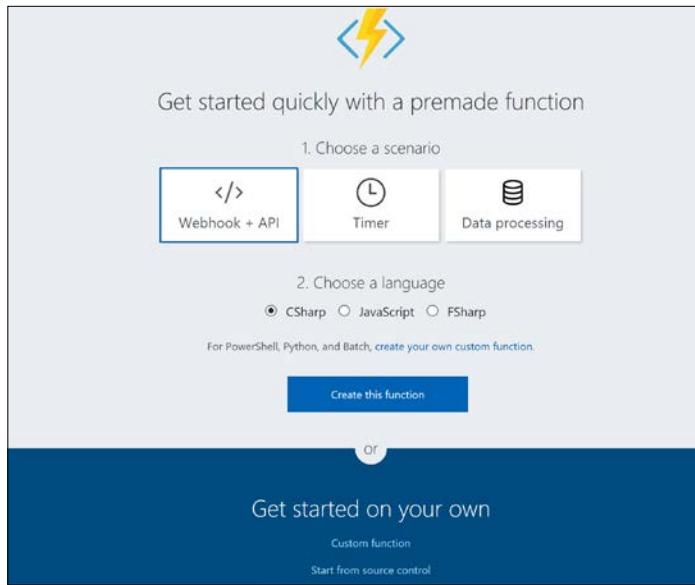
Crear una aplicación de funciones de Azure proporcionará un panel después del aprovisionamiento.



Al hacer clic en el botón más junto a las funciones, se mostrará un asistente para crear una nueva función. Este asistente muestra plantillas rápidas para crear las funciones **Webhook + API**, **Temporizador** y **Procesamiento de datos**. Eso creará el código y la estructura de Scaffold necesarios para comenzar. Este código de Scaffold también se genera para los enlaces y desencadenadores predeterminados. También permite la selección de un lenguaje para la implementación de la función. Azure Functions permiten lo siguiente:

- C#
- JavaScript
- F#
- PowerShell
- Bash
- Lenguajes Python

Este asistente también permite la creación de funciones personalizadas desde cero.



Se proporcionan muchas más opciones para crear una función personalizada.

The screenshot shows a grid of ten Azure Function templates. At the top, there are filters for "Language: All" and "Scenario: Core". The templates are arranged in two rows of five:

HttpTrigger - C# A C# function that will be run whenever it receives an HTTP request	HttpTrigger - F# An F# function that will be run whenever it receives an HTTP request	HttpTrigger - JavaScript A JavaScript function that will be run whenever it receives an HTTP request	TimerTrigger - C# A C# function that will be run on a specified schedule	TimerTrigger - F# An F# function that will be run on a specified schedule
TimerTrigger - JavaScript A JavaScript function that will be run on a specified schedule	QueueTrigger - C# A C# function that will be run whenever a message is added to a specified Azure Queue Storage	QueueTrigger - F# An F# function that will be run whenever a message is added to a specified Azure Queue Storage	QueueTrigger - JavaScript A JavaScript function that will be run whenever a message is added to a specified Azure Queue Storage	BlobTrigger - C# A C# function that will be run whenever a blob is added to a specified container

En esta sección, se creará una función personalizada de Azure utilizando el lenguaje PowerShell, que se puede ejecutar cada vez que recibe una solicitud HTTP. Proporciona un nombre adecuado, en este caso `FirstFunction` y selecciona **Anónimo** como nivel de autorización. Hablaremos sobre los niveles de autorización más adelante en este capítulo.

The screenshot shows the Azure Functions quickstart template selection interface. At the top, there are dropdown menus for 'Language' (set to 'PowerShell') and 'Scenario' (set to 'Experimental'). Below these are three template cards:

- HttpTrigger - PowerShell**: (Preview) A PowerShell function that will be run whenever it receives an HTTP request.
- TimerTrigger - PowerShell**: (Preview) A PowerShell function that will be run on a specified schedule.
- QueueTrigger - PowerShell**: (Preview) A PowerShell function that will be run whenever a message is added to a specified Azure Queue Storage.

A note below the templates states: "This template is experimental and does not yet have full support. If you run into issues, please file a bug on our [GitHub repository](#)".

The 'Name your function' field contains 'FirstFunction'. Under 'HTTP trigger', the 'Authorization level' dropdown is set to 'Anonymous'.

La creación de esta función proporciona un entorno integrado de creación de funciones completo junto con algunos códigos. Este código obtiene el contenido sin procesar del parámetro `req` entrante, que se completa con el tiempo de ejecución de Azure con los datos entrantes (cadena de consulta, valores de formulario, etc.) y lo convierte en el objeto de PowerShell. Podría haber varios tipos de datos dentro de este parámetro entrante y se extrae un solo valor de `name`. Azure Functions Runtime también crea una nueva variable para cada `querystring` entrante con una solicitud HTTP y la agrega con `$req_query_`. Por lo tanto, el envío del nombre como `querystring` proporcionará una variable lista para usar, `$req_query_name`, que contiene el valor para `querystring` del nombre y se consumirá dentro de la función. La respuesta para esta función se escribe en un archivo que el tiempo de ejecución de la función de Azure lee y envía al navegador.



```

1 # POST method: $req
2 $requestBody = Get-Content $req -Raw | ConvertFrom-Json
3 $name = $requestBody.name
4
5 # GET method: each querystring parameter is its own variable
6 if ($req_query_name)
7 {
8     $name = $req_query_name
9 }
10
11 Out-File -Encoding Ascii -FilePath $res -inputObject "Hello $name"
12

```

Esta función se puede invocar mediante una solicitud HTTP desde el navegador. La URL para esta función está disponible en el entorno y se compone del nombre de la aplicación de la función junto con el nombre de la función. El formato es `https://<<function app name>>.azurewebsites.net/api/<<function name>>`. En este caso, la URL será `https://azurerefarchitects.azurewebsites.net/api/FirstFunction`.

Para enviar parámetros a esta función, se pueden agregar parámetros de cadena de consulta adicionales al final de la URL. Por ejemplo, para enviar parámetros de nombre a esta función, se puede usar la URL `https://azurerefarchitects.azurewebsites.net/api/FirstFunction?name=ritesh`. La salida de la función se muestra en la siguiente figura:



Para las funciones basadas en HTTP, la función de Azure ya proporciona desencadenadores y enlaces dentro del archivo `function.json`. Este archivo se utiliza para definir todos los desencadenadores y enlaces de nivel de función y hay uno asociado con cada función.

The screenshot shows the Azure Functions configuration interface. At the top, there are buttons for 'Save' and 'Run'. To the right of 'Run' is a link 'Get function URL'. The main area contains the `function.json` file content:

```
1 {
2   "bindings": [
3     {
4       "name": "req",
5       "type": "httpTrigger",
6       "direction": "in",
7       "authLevel": "anonymous"
8     },
9     {
10      "name": "res",
11      "type": "http",
12      "direction": "out"
13    }
14  ],
15  "disabled": false
16 }
```

La plantilla HTTP crea un desencadenador para todas las solicitudes entrantes. El desencadenador invoca la función Azure y transfiere todos los datos entrantes y la carga útil como un parámetro denominado `req`. Este parámetro está disponible dentro de la función de Azure. La respuesta de la función es un enlace que toma la salida de la variable `res` de la función de Azure y la envía de vuelta al canal HTTP como respuesta.

Cómo crear una función impulsada por eventos

En este ejemplo, se creará una función de Azure conectada a la cuenta de Azure Storage. La cuenta de almacenamiento tiene un contenedor para guardar todos los archivos blob. El nombre de la cuenta de almacenamiento es `incomingfiles` y el del contenedor es `orders`:

Blob service
incomingfiles

+ Container Refresh

Storage account
[incomingfiles](#)

Status
Primary: Available

Location
West Central US

Subscription (change)
[RiteshSubscription](#)

Subscription ID
9755ffce-e94b-4332-9be8-1ade15e78909

Search containers by prefix

NAME
orders

Crea una nueva función de Azure desde el portal.

Choose a template below or go to the quickstart

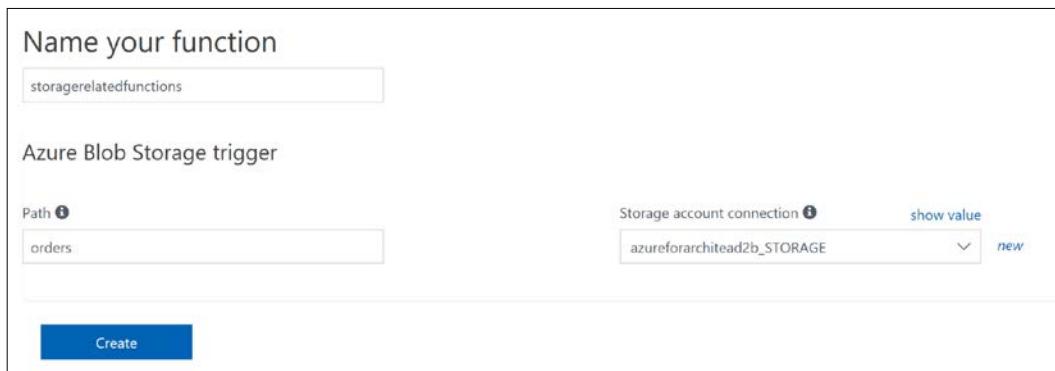
Language: C# Scenario: Core

HttpTrigger - C# A C# function that will be run whenever it receives an HTTP request	TimerTrigger - C# A C# function that will be run on a specified schedule	QueueTrigger - C# A C# function that will be run whenever a message is added to a specified Azure Queue Storage	BlobTrigger - C# A C# function that will be run whenever a blob is added to a specified container	EventHubTrigger - C# A C# function that will be run whenever an event hub receives a new event
ServiceBusQueueTrigger - C# A C# function that will be run whenever a message is added to a specified Service Bus queue	ServiceBusTopicTrigger - C# A C# function that will be run whenever a message is added to the specified Service Bus topic	Generic Webhook - C# A C# function that will be run whenever it receives a webhook request	GitHub Webhook - C# A C# function that will be run whenever it receives a GitHub webhook request	HttpTriggerWithParameters - C# A C# function that will be run whenever it receives an HTTP request

A partir de ahora, esta función de Azure no tiene información de conectividad con la cuenta de almacenamiento. Azure Functions necesita información de conexión para la cuenta de almacenamiento y que sea visible desde la pestaña **Claves de acceso** en la cuenta de almacenamiento. Se puede obtener la misma información mediante el entorno del editor de funciones de Azure. De hecho, permite la creación de una nueva cuenta de almacenamiento desde el mismo entorno de edición.

Esto se puede agregar usando el nuevo botón al lado del tipo de **entrada de conexión de la cuenta de almacenamiento**. Permite seleccionar una cuenta de almacenamiento existente o crear una nueva cuenta de almacenamiento. Como ya tengo un par de cuentas de almacenamiento, las estoy reutilizando. Los lectores deben crear una cuenta de Azure Storage separada. La selección de una cuenta de almacenamiento actualizará la configuración en la sección **appsettings** con la cadena de conexión agregada.

Asegúrate de que ya exista un contenedor dentro del servicio blob de la cuenta de Azure Storage de destino. La entrada de ruta se refiere a la ruta al contenedor. En este caso, el contenedor de órdenes ya existe dentro de la cuenta de almacenamiento. El botón **Crear** aprovisionará la nueva función que supervisa el contenedor de la cuenta de almacenamiento.



El código de la función de Azure es el siguiente:

```
public static void Run(Stream myBlob, TraceWriter log)
{
    log.Info($"C# Blob trigger function Processed blob\n \n Size
{myBlob.Length} Bytes");
}
```

Los enlaces se muestran a continuación:

```
{
  "bindings": [
    {
      "name": "myBlob",
      "type": "blobTrigger",
      "path": "orders"
    }
  ]
}
```

```
        "direction": "in",
        "path": "orders",
        "connection": "azureforarchitead2b_STORAGE"
    }
],
"disabled": false
}
```

Ahora, cargar cualquier archivo blob en el contenedor de órdenes debería activar la función:

The screenshot shows the Azure Functions developer tools interface. At the top, there is a toolbar with a 'Save' button and a 'Run' button. Below the toolbar is the code editor window containing the following C# code:

```
1 public static void Run(Stream myBlob, TraceWriter log)
2 {
3     log.Info($"C# Blob trigger function Processed blob\n \n Size: {myBlob.Length}");
4 }
5
```

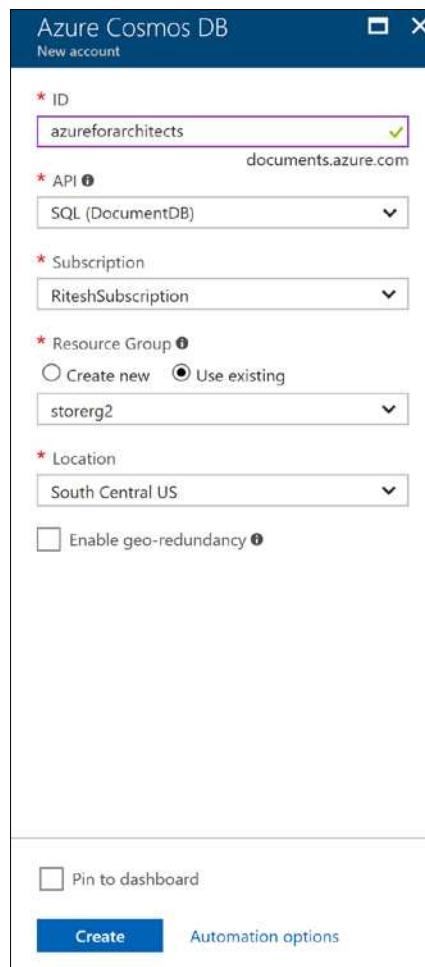
Below the code editor is a large empty space, likely a preview or preview logs area. At the bottom of the interface is the 'Logs' panel. The logs panel has a header with buttons for 'Pause', 'Clear', 'Copy logs', 'Expand', and a dropdown menu. The log entries are as follows:

```
2017-09-20T10:24:12.504 Function started (Id=ac68a8f8-712f-4df8-975e-238b0ecf5b05)
2017-09-20T10:24:12.536 C# Blob trigger function Processed blob
    Size: 2480471 Bytes
2017-09-20T10:24:12.536 Function completed (Success, Id=ac68a8f8-712f-4df8-975e-238b0ecf5b05, Dur
```

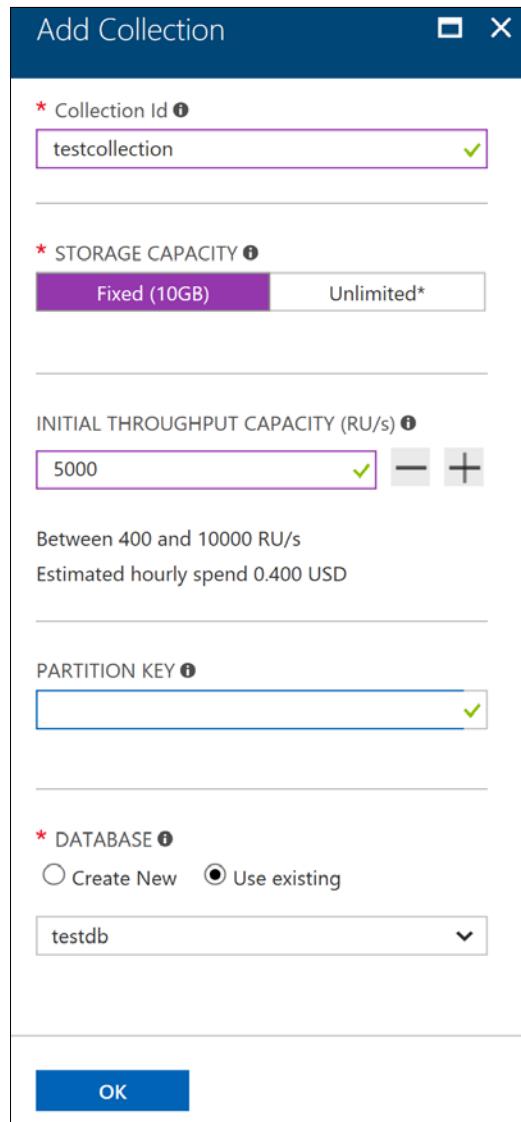
Creación de una arquitectura conectada con funciones

Una arquitectura conectada con funciones se refiere a la creación de múltiples funciones, por lo que la salida de una función activa otra función y proporciona datos para que la siguiente función ejecute su lógica. En esta sección, continuaremos con el escenario anterior de la cuenta de almacenamiento. En este caso, la salida de la función que se desencadenará mediante archivos de Azure Storage Blob escribirán el tamaño del archivo a Azure Cosmos DB.

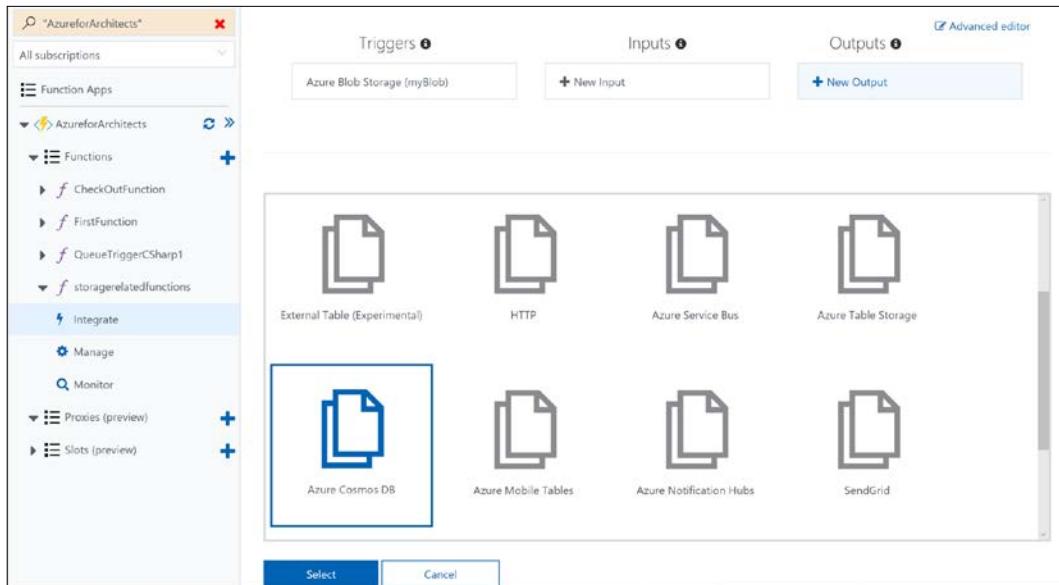
La configuración de Cosmos DB se muestra a continuación. De forma predeterminada, no hay colecciones creadas en Cosmos DB. Una colección se creará automáticamente mientras se crea una función que se activará cuando Cosmos DB obtenga cualquier información.



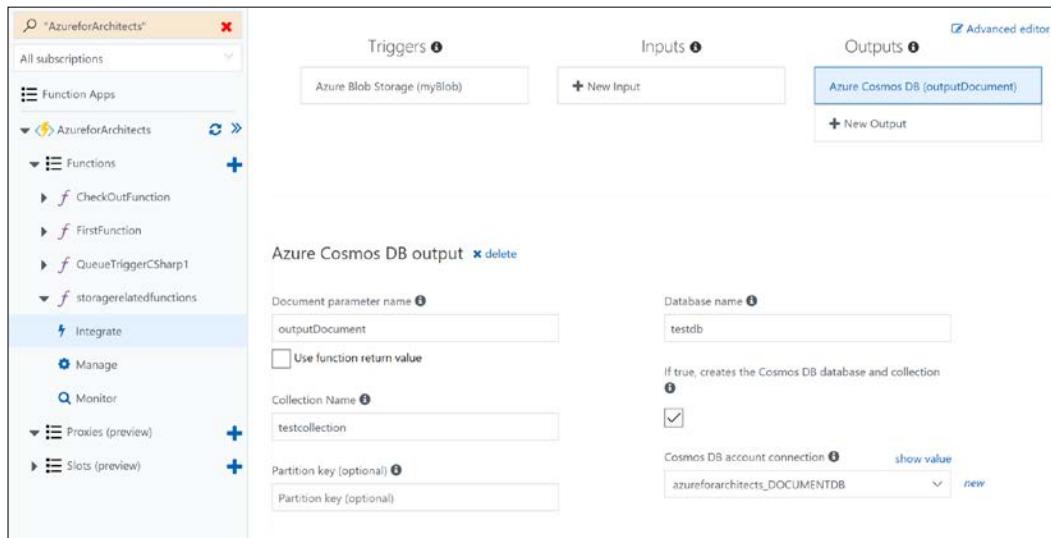
Crea una base de datos nueva `testdb` en DB y una colección nueva llamada `testcollection` en ella. Necesitas tanto la base de datos como el nombre de la colección al configurar Azure Functions.



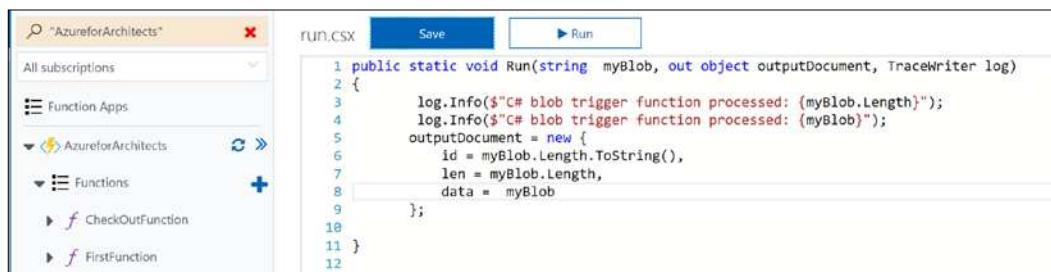
Es hora de volver a visitar la función `storagerelatedfunctions` y cambiar su enlace para volver al tamaño de los datos del archivo cargado. Este valor devuelto se escribirá en Cosmos DB. Esto exigirá un cambio en los enlaces, así como un de adicional responsable de capturar los valores de salida. Finalmente, este enlace escribirá en la colección Cosmos DB. En la pestaña **Integrar**, haz clic en el botón **Nueva salida** y en la etiqueta **Salidas** y selecciona **Azure Cosmos DB**.



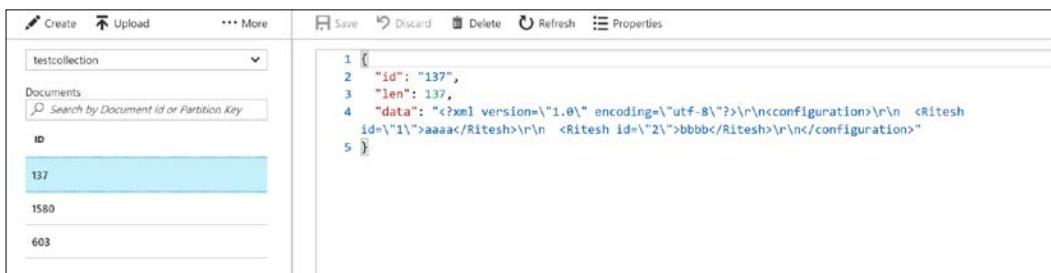
Proporciona los nombres apropiados para la base de datos y la colección (marca la casilla de verificación para crear la colección si no existe), haz clic en el botón **Nuevo** para seleccionar nuestro recién creado Azure Cosmos DB y deja el nombre del parámetro como `outputDocument`.



Modifica la función como se muestra a continuación.



Ahora, al cargar un nuevo archivo a la colección de órdenes en la cuenta Azure Storage se ejecutará una función que se escribirá en la colección de Azure Cosmos DB. Ahora se puede escribir otra función con el recién creado Azure Cosmos DB como un enlace de activación. Proporcionará el tamaño de los archivos y la función puede actuar sobre él. Esto se muestra a continuación:



Resumen

Sin servidor es una nueva unidad de implementación en el mundo de hoy. La evolución de las funciones de los métodos tradicionales ha llevado al diseño de una arquitectura de acoplamiento flexible, de evolución independiente y autosuficiente, que anteriormente era solo un concepto. Las funciones son una unidad de implementación y proporcionan un entorno donde los usuarios no necesitan administrar el entorno. Todo lo que tienen que importar es el código escrito para la funcionalidad. Azure proporciona una plataforma madura para alojar funciones e integrarlas sin problemas en base a eventos o a petición. Casi todos los recursos de Azure pueden participar en una arquitectura compuesta por funciones de Azure. El futuro está en las funciones, ya que cada vez más organizaciones quieren mantenerse alejadas de la gestión de infraestructuras y plataformas. Quieren descargar esto a los proveedores de cloud. Es una característica esencial que debe dominar todo arquitecto que trabaje con Azure.

El siguiente capítulo es muy interesante, ya que trata los conceptos de Azure Resource Manager, como las políticas, los bloqueos y etiquetas de los detalles.

9

Diseño de políticas, bloqueos y etiquetas

Azure es una versátil plataforma en el cloud. Los consumidores pueden no solo crear e implementar su aplicación, también pueden administrar y gestionar sus entornos. Por lo general, el cloud sigue un paradigma de pago por uso en el que un consumidor contrata una suscripción de Azure en la que puede implementar prácticamente cualquier cosa. Puede ser tan pequeño como una pequeña máquina virtual básica o constar de miles de máquinas virtuales con unas SKU mayores. Azure no impedirá que ningún consumidor aprovisione ningún recurso ni lo limitará en términos de números.

Dentro de una organización, podría haber un gran número de personas con acceso a la suscripción de Azure. Es necesario tener implantado un modelo de gestión, de manera que solo se aprovisionen los recursos necesarios y suficientes por parte de personas que posean autorización para crearlos. Azure proporciona funciones de administración de los recursos, como el **control de accesos basado en roles** de Azure (**RBAC**) y las políticas y bloqueos para administrar y proporcionar una gestión de recursos.

Otro aspecto importante de la gestión es la administración de los costes, el uso y la información. La gerencia de la organización siempre quiere mantenerse actualizada sobre el consumo y el coste de su cloud. Le gustaría identificar qué equipo, departamento o unidad está utilizando qué porcentaje de su coste total. En resumen, quieren tener informes basados en varias dimensiones sobre el consumo y el coste. Azure proporciona una función de etiquetado que puede ayudar a proporcionar este tipo de información a la gerencia sobre la marcha.

En este capítulo, abordaremos los siguientes temas:

- Azure RBAC
- Políticas de Azure
- Bloqueos de Azure
- Etiquetas de Azure

Etiquetas de Azure

Las etiquetas en un diccionario se definen como *una etiqueta que se asocia a alguien o algo con fines de identificación o para dar otra información*. Azure permite el etiquetado de los grupos de recursos y recursos con pares nombre-valor. El etiquetado ayuda a la organización lógica y la clasificación de los recursos. Azure también permite el etiquetado de 15 pares de nombre-valor por grupos de recursos y sus recursos. Aunque el grupo de recursos es un contenedor de recursos, el etiquetado del grupo de recursos no supone el etiquetado de sus recursos constituyentes. Los grupos de recursos y los recursos deben etiquetarse basándose en su uso, lo que se explicará más adelante en esta sección. Las etiquetas funcionan en un nivel de suscripción. Azure acepta cualquier par de nombre-valor y por ello es importante para una organización definir los nombres y sus posibles valores.

Pero, ¿por qué es importante el etiquetado? En otras palabras, qué problemas pueden resolverse mediante el etiquetado. El etiquetado tiene las siguientes ventajas:

- **Clasificación de recursos:** una suscripción a Azure la pueden emplear varios departamentos y roles dentro de una organización. Es importante para la gerencia identificar a los propietarios de estos recursos. El etiquetado ayuda a asignar identificadores a los recursos que pueden representar a departamentos o roles.
- **Administración de información para recursos de Azure:** de nuevo, cualquiera que tenga acceso a una suscripción puede aprovisionar recursos de Azure de dicha suscripción. A las organizaciones les gustaría tener una clasificación adecuada de los recursos en términos de las políticas de administración de la información. Las políticas se pueden basar en la administración del ciclo de vida de la aplicación, como los entornos de desarrollo, pruebas y producción. Pueden basarse en el uso: orientación interna o externa. Pueden basarse en sus prioridades y mucho más. Cada organización tiene su propia forma de definir las categorías de la información y, para ellas, Azure no es diferente. Las etiquetas ayudan a organizar los recursos de manera lógica.
- **Administración de costes:** el etiquetado en Azure puede ayudar a identificar los recursos basándose en su clasificación. Se pueden ejecutar consultas en Azure para identificar el coste por categoría según las categorías de administración de la información definidas. Por ejemplo, el coste de los recursos en Azure para el desarrollo de un entorno para el departamento de finanzas y el departamento de marketing puede determinarse con facilidad. Por otra parte, Azure también proporciona información de facturación basándose en las etiquetas. Esto ayuda a identificar qué equipos, departamentos o grupos están consumiendo.

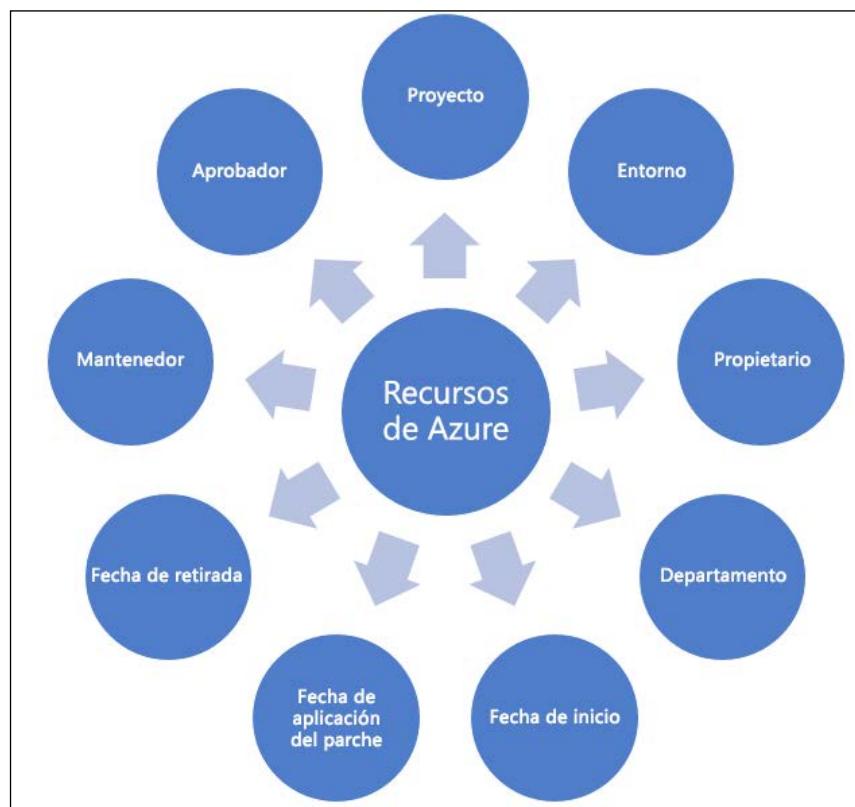
Las etiquetas tienen ciertas limitaciones en Azure.

Azure permite asociar un máximo de 15 pares de etiquetas nombre-valor a recursos y grupos de recursos.

Las etiquetas no son heredables. Las etiquetas aplicadas a un grupo de recursos no se aplican a los recursos individuales que hay dentro de él. Sin embargo, es muy fácil olvidar etiquetar los recursos mientras se aprovisionan. Las políticas de Azure son el mecanismo para asegurar que las etiquetas se etiquetan con el valor apropiado durante el aprovisionamiento. Tendremos en cuenta los detalles de las políticas más adelante en este capítulo. Se dedica una sección completa a las políticas.

Las etiquetas pueden asignarse a recursos y grupos de recursos con PowerShell, Azure CLI 2.0, las plantillas de administración de recursos de Azure, el portal de Azure y la API REST de Azure Resource Manager.

Aquí se muestra un ejemplo de clasificación de la administración de la información con las etiquetas de Azure que pueden reutilizarse. En este ejemplo, se utilizan las etiquetas de departamento, proyecto, entorno, propietario, aprobador, encargado, así como los pares de nombre-valor de fecha-inicio, fecha-retirada y corrección-fecha para etiquetar un grupo de recursos o recursos. Es muy fácil encontrar todos los recursos para una etiqueta determinada o una combinación de etiquetas con PowerShell, Azure CLI o API de REST.



Etiquetas con PowerShell

Las etiquetas pueden gestionarse con PowerShell, las plantillas de Azure Resource Manager, el portal y API de REST. En esta sección, se utilizará PowerShell para crear y aplicar etiquetas. PowerShell proporciona un cmdlet para recuperar y asociar etiquetas a los recursos y los grupos de recursos.

- Para recuperar etiquetas asociadas a un recurso con PowerShell, puede utilizarse el cmdlet `Find-AzureRMResource`:

```
(Find-AzureRmResource -TagName Dept -TagValue Finance).Name
```

- Para recuperar etiquetas asociadas a un grupo de recursos con PowerShell, puede utilizarse el siguiente comando:

```
(Find-AzureRmResourceGroup -Tag @{ Dept="Finance" }).Name
```

- Para asignar etiquetas a un grupo de recursos, puede utilizarse el cmdlet `Set-AzureRmResourceGroup`:

```
Set-AzureRmResourceGroup -Name examplegroup -Tag @{ Dept="IT"; Environment="Test" }
```

- Para asignar etiquetas a un recurso, puede utilizarse el cmdlet `Set-AzureRmResource`:

```
Set-AzureRmResource -Tag @{ Dept="IT"; Environment="Test" } -ResourceName examplevnet -ResourceGroupName examplegroup
```

Etiquetas con la plantilla de ARM

La plantilla de Azure Resource Manager también proporciona ayuda para definir las etiquetas para cada recurso. Puede utilizarse para asignar varias etiquetas a cada recurso. Esto se muestra como sigue:

```
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "resources": [
    {
      "apiVersion": "2016-01-01",
      "type": "Microsoft.Storage/storageAccounts",
      "name": "[concat('storage', uniqueString(resourceGroup().id))]",
      "location": "[resourceGroup().location]",
      "tags": {
        "Dept": "Finance",
        "Environment": "Production"
      }
    }
  ]
}
```

```
        "sku": {  
            "name": "Standard_LRS"  
        },  
        "kind": "Storage",  
        "properties": {}  
    }  
]  
}
```

En el ejemplo anterior, se añade un par de etiquetas (Dpto y Entorno) a un recurso de cuenta de almacenamiento con las plantillas de Azure Resource Manager.

Grupos de recursos y recursos

Es necesario que los arquitectos decidan la taxonomía y la arquitectura de la información para los recursos y grupos de recursos de Azure. Deben identificar las categorías en las que se clasificarán los recursos según los requisitos de consulta. Sin embargo, también deben identificar si las etiquetas deben asociarse a recursos individuales o grupos de recursos.

Si todos los recursos de un grupo de recursos necesitan las mismas etiquetas, es mejor etiquetar el grupo de recursos en lugar de etiquetar cada recurso. Es importante tener en cuenta las consultas sobre etiquetas antes de determinar si estas etiquetas deben aplicarse en el nivel de los recursos o en el nivel del grupo de recursos. Si las consultas se refieren a tipos de recursos individuales en toda la suscripción y grupos de recursos, entonces, tiene más sentido asignar las etiquetas a los recursos; sin embargo, si se identifican bastantes grupos de recursos en las consultas, entonces las etiquetas deben aplicarse a grupos de recursos.

Políticas de Azure

La sección anterior describía la aplicación de etiquetas para implementaciones de Azure. Las etiquetas son ideales para la organización de los recursos; sin embargo, hay un factor más y un asunto que no se ha mencionado. ¿Cómo imponen las organizaciones la aplicación de las etiquetas para cada implementación? Debe haber una aplicación automatizada de las etiquetas de Azure a los recursos y los grupos de recursos. No hay ninguna comprobación de Azure que garantice la aplicación de las etiquetas apropiadas a los recursos y a los grupos de recursos. Esto no está solo relacionado con las etiquetas; puede estar relacionado con cualquier configuración de cualquier recurso de Azure. Por ejemplo, todos los recursos deben implementarse únicamente en el Este de Estados Unidos y no debe implementarse ningún recurso en ningún otro lugar.

Puede que a estas alturas ya hayas adivinado que esta sección trata acerca de la formulación de un modelo de gestión en Azure. La gestión es un elemento importante para Azure porque ayuda a mantener la disciplina y el coste bajo control. También garantiza que todo aquel que acceda al entorno de Azure sea consciente de las prioridades y los procesos de la organización. Esto contribuye a definir las convenciones de la organización que ayudan a administrar los recursos.

Cada política puede crearse con varias reglas y pueden aplicarse varias políticas a grupos de suscripciones y de recursos. Después de que las reglas se hayan aplicado y cuando se hayan satisfecho, se ejecutará una acción. La acción podría ser denegar la transacción en curso, auditar la transacción, lo que significa escribirla en registros y permitir que finalice, así como anexar metadatos a la transacción si carece de ellos.

Algunos ejemplos de políticas podrían estar relacionados con la convención de nomenclatura de recursos, el etiquetado de recursos, los tipos de recursos que se pueden aprovisionar, la ubicación de los recursos o cualquier combinación de estas reglas.

Políticas integradas

Azure proporciona infraestructura para crear políticas personalizadas, aunque también proporciona algunas políticas "out of the box" que con frecuencia son necesarias y utilizadas para la gestión. Estas políticas se refieren a la ubicación permitida, los tipos de recursos permitidos y las etiquetas. Se puede encontrar más información sobre estas políticas integradas en <https://docs.microsoft.com/azure/azure-resource-manager/resource-manager-policy>.

Lenguaje de la política

Las políticas de Azure utilizan el lenguaje JSON para definir y describir las políticas.

Hay dos pasos en la adopción de una política. La política debe definirse y luego debe aplicarse y asignarse. Las políticas tienen un alcance y pueden aplicarse en el nivel de grupo de recursos y de suscripción.

Las políticas se definen mediante bloques `si...` `entonces`, lo que es similar a cualquier lenguaje de programación popular. El bloque `si` se ejecuta para evaluar las condiciones y se basa en los resultados de esas condiciones; a continuación, se ejecuta el bloque `entonces`:

```
{
  "if": {
    <condition> | <logical operator>
  },
  "then": {
    "effect": "deny | audit | append"
  }
}
```

Las políticas de Azure no solo permiten condiciones **si** sencillas, sino que varias condiciones **si** pueden combinarse de forma lógica para crear reglas complejas. Estas condiciones pueden unirse con operadores **Y**, **O** y **NO**.

- La sintaxis **Y** requiere que todas las condiciones sean verdaderas
- La sintaxis **O** requiere que una de las condiciones sea verdadera
- La sintaxis **NO** invierte el resultado de la condición

A continuación, se muestra la sintaxis **Y**. Está representada por la palabra clave **allOf**:

```
"if": {
  "allOf": [
    {
      "field": "tags",
      "containsKey": "application"
    },
    {
      "field": "type",
      "equals": "Microsoft.Storage/storageAccounts"
    }
  ]
},
```

A continuación, se muestra la sintaxis **O**. Está representada por la palabra clave **anyOf**:

```
"if": {
  "anyOf": [
    {
      "field": "tags",
      "containsKey": "application"
    },
    {
      "field": "type",
      "equals": "Microsoft.Storage/storageAccounts"
    }
  ]
},
```

A continuación, se muestra la sintaxis **NO**. Está representada por la palabra clave **not**:

```
"if": {
  "not": [
    {
      "field": "tags",
      "containsKey": "application"
    }
  ]
},
```

```
{  
    "field": "type",  
    "equals": "Microsoft.Storage/storageAccounts"  
}  
]  
,
```

De hecho, estos operadores lógicos pueden combinarse como sigue:

```
"if": {  
    "allOf": [  
        {  
            "not": {  
                "field": "tags",  
                "containsKey": "application"  
            }  
        },  
        {  
            "field": "type",  
            "equals": "Microsoft.Storage/storageAccounts"  
        }  
    ]  
},
```

Esto es muy similar a las condiciones `si` en lenguajes de programación populares como C# y Node.js

```
If ("type" == "Microsoft.Storage/storageAccounts") {  
    Deny  
}
```

Cabe indicar que no hay acción `allow` (permitir), aunque hay una acción `Deny` (denegar). Esto significa que las reglas de la política se deben escribir con la denegación en perspectiva. Las reglas deben evaluar las condiciones y devolver `Deny` si devuelven "true".

Campos permitidos

Los campos que se permiten en las políticas son los siguientes:

- Name
- Kind
- Type
- Location

- Tags
- Tags.*
- Property aliases

Bloqueos de Azure

Los bloqueos son un mecanismo para detener determinadas actividades sobre los recursos. RBAC proporciona derechos a usuarios/grupos/aplicación en un determinado ámbito. Hay roles RBAC "out of the box", como propietario, colaborador y lector. Con el rol de colaborador, es posible eliminar o modificar un recurso. ¿Cómo pueden evitarse este tipo de actividades a pesar de que el usuario tenga un rol de colaborador? Aquí entran los bloqueos de Azure.

Los bloqueos de Azure pueden ayudar de dos maneras:

- Pueden bloquear recursos de manera que no puedan borrarse, incluso si se tiene acceso de propietario
- Pueden bloquear recursos de tal manera que no puedan borrarse y no pueda modificarse su configuración

Por lo general, esto resulta muy útil para recursos en el entorno de producción, en el que no deben modificarse ni eliminarse accidentalmente.

Los bloqueos se pueden aplicar en un nivel de suscripción, de grupo de recursos y de recursos individuales. Los bloqueos siguen la herencia entre suscripciones, grupo de recursos y recursos. Aplicar un bloqueo en el nivel primario garantizará que aquellos en el nivel secundario también lo hereden. Incluso los recursos que añadas más tarde heredarán el bloqueo primario. El bloqueo más restrictivo en la herencia tiene prioridad. Aplicar un bloqueo en el nivel de recursos tampoco permitirá la eliminación de un grupo de recursos que contenga el recurso.

Los bloqueos se aplican solo a las operaciones que ayudan a la administración del recurso en lugar de a las operaciones que son internas al recurso. Los usuarios necesitan permisos RBAC `Microsoft.Authorization/*` o `Microsoft.Authorization/locks/*` para crear y modificar bloqueos.

Los bloqueos pueden crearse y aplicarse a través del portal de Azure, Azure PowerShell, Azure CLI, las plantillas de ARM y API REST.

La creación de un bloqueo con la plantilla de ARM se ve como sigue:

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/  
  deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "parameters": {
```

```
    "lockedResource": {
        "type": "string"
    }
}
"resources": [
{
    "name": "[concat(parameters('lockedResource'), '/Microsoft.Authorization/myLock')]",
    "type": "Microsoft.Storage/storageAccounts/providers/locks",
    "apiVersion": "2015-01-01",
    "properties": {
        "level": "CannotDelete"
    }
}
]
```

La creación y la aplicación de un bloqueo al recurso con PowerShell se ve como sigue:

```
New-AzureRmResourceLock -LockLevel CannotDelete -LockName LockSite ^
-ResourceName examplesite -ResourceType Microsoft.Web/sites ^
-ResourceGroupName exempleresourcegroup
```

La creación y la aplicación de un bloqueo al grupo de recursos con PowerShell se ve como sigue:

```
New-AzureRmResourceLock -LockName LockGroup -LockLevel CannotDelete ^
-ResourceGroupName exempleresourcegroup
```

La creación y la aplicación de un bloqueo al recurso con Azure CLI se ve como sigue:

```
az lock create --name LockSite --lock-type CannotDelete \
--resource-group exempleresourcegroup --resource-name examplesite \
--resource-type Microsoft.Web/sites
```

La creación y la aplicación de un bloqueo al grupo de recursos con Azure CLI se ve como sigue:

```
az lock create --name LockGroup --lock-type CannotDelete \
--resource-group exempleresourcegroup
```

Azure RBAC

Azure proporciona autenticación con Azure AD para sus recursos. Una vez autenticado, se debe evaluar y decidir cuándo se debe permitir que la identidad acceda a cualquier recurso, a todos los recursos o solo al recurso seleccionado para ese usuario.

Tradicionalmente, esta actividad se ha denominado autorización. La autorización se evalúa si la identidad determinada tiene los permisos necesarios para acceder al recurso y si puede realizar la operación deseada. Cualquiera que tenga acceso a una suscripción de Azure debe tener solo los permisos suficientes para que el trabajo se pueda realizar.

No deben asignarse a las identidades más que los permisos necesarios para asegurar que la superficie de asociación siga siendo mínima.

La autorización también se conoce popularmente como Control de acceso basado en roles. RBAC en Azure se refiere a la asignación de permisos a las identidades (usuarios/grupos/aplicaciones) en un ámbito. Este ámbito puede ser una suscripción, un grupo de recursos o un recurso individual.

RBAC ayuda a la creación y la asignación de diferentes permisos a diferentes identidades. Esto ayuda a separar las funciones dentro de los equipos en lugar de que todo el mundo tenga todos los permisos. De esta manera, se ayuda a que la gente se haga responsable de su trabajo, porque puede que otros ni siquiera tengan acceso para hacerlo. Cabe señalar que proporcionar acceso a un ámbito superior garantiza automáticamente que los recursos secundarios hereden esos permisos. Por ejemplo, proporcionar acceso de lectura sobre un grupo de recursos garantiza que los recursos de su interior tengan permisos de lectura para la identidad en cuestión.

Azure proporciona tres roles incorporados con propósito general. Son los siguientes:

- El rol de propietario, que tiene acceso completo a todos los recursos
- El rol de colaborador, que tiene acceso a recursos de lectura y escritura
- El rol de lectores, que solo tiene permiso de lectura sobre los recursos

Hay más roles proporcionados por Azure, pero son específicos de cada recurso. Algunos ejemplos son el colaborador de red y el administrador de seguridad.

Para obtener todas las funciones proporcionadas por Azure para todos los recursos, ejecuta el comando `Get-AzureRmRoleDefinition` en la consola de PowerShell.

Cada definición de rol tiene unas determinadas acciones permitidas y no permitidas. Por ejemplo, el rol de propietario tiene todas las acciones permitidas y ninguna de las acciones está prohibida. Las acciones prohibidas prevalecen sobre todas las acciones:

```
PS C:\Users\rimodi> Get-AzureRmRoleDefinition -Nombre "propietario"
```

```
Name          : Propietario
Id           : 8e3af657-a8ff-443c-a75c-2fe8c4bcb635
IsCustom     : Falso
Description   : Te permite administrarlo todo, incluido el acceso a los recursos.
Actions       : {*}
NotActions    : {}
AssignableScopes : {/}
```

Cada rol consta de varios permisos. Cada recurso proporciona una lista de operaciones. La operación compatible con un recurso puede obtenerse con el cmdlet Get-AzureRmProviderOperation. Este cmdlet toma el nombre del proveedor y el recurso para recuperar las operaciones:

```
Get-AzureRmProviderOperation -OperationSearchString "Microsoft.Insights/*"
```

Esto se traducirá en el resultado siguiente:

```
PS C:\Users\rimodi> get-AzureRmProviderOperation -OperationSearchString "Microsoft.Insights/*" | select operation
```

```
Operación
-----
Microsoft.Insights/Register/Action
Microsoft.Insights/AlertRules/Write
Microsoft.Insights/AlertRules/Delete
Microsoft.Insights/AlertRules/Read
Microsoft.Insights/AlertRules/Activated/Action
Microsoft.Insights/AlertRules/Resolved/Action
Microsoft.Insights/AlertRules/Throttled/Action
Microsoft.Insights/AlertRules/Incidents/Read
Microsoft.Insights/MetricDefinitions/Read
Microsoft.Insights/eventtypes/values/Read
Microsoft.Insights/eventtypes/digestevents/Read
```

```
Microsoft.Insights/Metrics/Read
Microsoft.Insights/LogProfiles/Write
Microsoft.Insights/LogProfiles/Delete
Microsoft.Insights/LogProfiles/Read
Microsoft.Insights/Components/Write
Microsoft.Insights/Components/Delete
Microsoft.Insights/Components/Read
Microsoft.Insights/AutoscaleSettings/Write
Microsoft.Insights/AutoscaleSettings/Delete
Microsoft.Insights/AutoscaleSettings/Read
Microsoft.Insights/AutoscaleSettings/Scaleup/Action
Microsoft.Insights/AutoscaleSettings/Scaledown/Action
    Microsoft.Insights/AutoscaleSettings/providers/Microsoft.Insights/
MetricDefinitions/Read
Microsoft.Insights/ActivityLogAlerts/Activated/Action
Microsoft.Insights/DiagnosticSettings/Write
Microsoft.Insights/DiagnosticSettings/Delete
Microsoft.Insights/DiagnosticSettings/Read
Microsoft.Insights/LogDefinitions/Read
Microsoft.Insights/Webtests/Write
Microsoft.Insights/Webtests/Delete
Microsoft.Insights/Webtests/Read
Microsoft.Insights/ExtendedDiagnosticSettings/Write
Microsoft.Insights/ExtendedDiagnosticSettings/Delete
Microsoft.Insights/ExtendedDiagnosticSettings/Read
```

Roles personalizados

Los roles se crean mediante la combinación de varios permisos. Por ejemplo, un rol personalizado puede constar de las operaciones de múltiples recursos, como se muestra a continuación:

```
$role = Get-AzureRmRoleDefinition "Virtual Machine Contributor"
$role.Id = $null
$role.Name = "Virtual Machine Operator"
$role.Description = "Can monitor and restart virtual machines."
$role.Actions.Clear()
$role.Actions.Add("Microsoft.Storage/*/read")
```

```
$role.Actions.Add("Microsoft.Network/*/read")
$role.Actions.Add("Microsoft.Compute/*/read")
$role.Actions.Add("Microsoft.Compute/virtualMachines/start/action")
    $role.Actions.Add("Microsoft.Compute/virtualMachines/restart/action")
$role.Actions.Add("Microsoft.Authorization/*/read")
    $role.Actions.Add("Microsoft.Resources/subscriptions/resourceGroups/
read")
$role.Actions.Add("Microsoft.Insights/alertRules/*")
$role.Actions.Add("Microsoft.Support/*")
$roleAssignableScopes.Clear()
$role.AssignableScopes.Add("/subscriptions/c276fc76-9cd4-44c9-99a7-
4fd71546436e")
$role.AssignableScopes.Add("/subscriptions/e91d47c4-76f3-4271-a796-
21b4ecfe3624")
New-AzureRmRoleDefinition -Role $role
```

¿En qué se diferencia de RBAC?

Los bloqueos no son los mismos que en RBAC. RBAC ayuda a permitir o denegar permisos sobre los recursos que son intrínsecos al recurso. Estos permisos se refieren a realizar operaciones como leer, escribir y actualizar recursos. Los bloqueos, por otra parte, se refieren a la anulación de permisos para configurar o eliminar el recurso.

Ejemplos de la implementación de funciones de gestión de Azure

En esta sección, veremos un ejemplo de implementación de arquitectura para una organización ficticia que quiere implementar las funciones de gestión y administración de costes de Azure.

Contexto

Empresa Inc es una empresa mundial que está implementando una solución de redes sociales en la IaaS de Azure. Utiliza los servidores web y los servidores de aplicaciones implementados en las máquinas virtuales y las redes de Azure. Azure SQL Server actúa como la base de datos back-end.

Control de acceso basado en roles

La primera tarea es asegurar que los equipos apropiados y los propietarios de la aplicación pueden acceder a sus recursos. Se considera que cada equipo tiene requisitos diferentes. En aras de la comprensión, Azure SQL se implementa en un grupo de recursos independiente en comparación con los artefactos de la IaaS de Azure.

El administrador asigna los siguientes roles para la suscripción:

Rol	Asignado a	Descripción
Propietario	El administrador	Administra todos los grupos de recursos y suscripciones.
Administrador de seguridad	Administradores de seguridad	Este rol permite a los usuarios examinar el centro de seguridad de Azure y el estado de los recursos.
Colaborador	Administración de infraestructuras	Administración de máquinas virtuales y otros recursos.
Lector	Desarrolladores	Puede ver recursos, pero no los puede modificar. Se espera que los desarrolladores trabajen en sus entornos de desarrollo y pruebas.

Resumen

La gestión y la administración de costes es una de las principales prioridades para las empresas que migran al cloud. Tener una suscripción de Azure con pago por uso puede perjudicar al presupuesto de la empresa porque cualquiera que tenga acceso a la suscripción puede aprovisionar tantos recursos como le apetezca. Algunos recursos son gratuitos mientras que otros son caros. Es importante para las organizaciones mantener el control sobre sus costes en el cloud. Las etiquetas ayudan a generar informes de facturación. Estos informes pueden basarse en departamentos, proyectos o propietarios, o cualquier otro criterio que se considere adecuado. Aunque el coste es importante, la gestión es igualmente importante. Azure ofrece bloqueos, políticas y RBAC para formular la gestión. Las políticas garantizan que las operaciones con recursos pueden denegarse o auditarse, los bloqueos garantizan que los recursos no puedan modificarse ni eliminarse y RBAC garantiza que los empleados tengan los permisos óptimos para realizar sus trabajos. Con estas cuatro características, las empresas pueden aplicar una gestión coherente y un control de costes en sus implementaciones de Azure.

El próximo capítulo se centrará en DevOps en Azure. En él, se comentarán algunos de los conceptos y técnicas de implementación más importantes.

10

DevOps en Azure

El desarrollo de software es una tarea compleja que comprende múltiples procesos, herramientas e involucra a personas de diferentes departamentos. Todos ellos deben unirse y trabajar de manera cohesiva. Con tantas variables, los riesgos son altos en el momento de la entrega a los consumidores finales. Una pequeña omisión o mala configuración y la aplicación podría fallar. Este capítulo trata sobre la adopción e implementación de prácticas que reducen este riesgo considerablemente y aseguran que el software de alta calidad se pueda entregar al consumidor una y otra vez.

Antes de entrar en detalles sobre DevOps, debemos entendamos los problemas a los que se enfrentan las empresas de software y que DevOps aborda.

- Las organizaciones son rígidas y no aceptan cambios.
- Procesos rígidos y lentos
- Equipos aislados que trabajan en silos
- Diseño monolítico e implementaciones a modo de “big bang”
- Ejecución manual
- Falta de innovación

En este capítulo, veremos en detalle lo siguiente:

- DevOps
- Prácticas de DevOps
- Administración de configuración
- Integración continua
- Implementación continua
- Entrega continua
- Visual Studio Team Services

- Preparación de DevOps
- DevOps para solución PaaS
- DevOps para soluciones basadas en máquinas virtuales (IaaS)
- DevOps para soluciones basadas en contenedor (IaaS)
- Azure Automation
- Herramientas de Azure para DevOps

¿Qué es DevOps?

Actualmente no hay consenso en la industria con respecto a la definición de DevOps. Cada organización ha formulado su propia definición de DevOps y ha tratado de implementarla. Tienen su propia perspectiva y piensan que han implementado DevOps si tienen su automatización en su lugar, la administración de la configuración habilitada, mediante el uso de procesos ágiles o cualquier otra combinación.

DevOps trata sobre el mecanismo de entrega de sistemas de software. Se trata de unir a las personas, hacer que colaboren y se comuniquen, trabajar juntos hacia un objetivo y una visión comunes. Se trata de tomar responsabilidad conjunta, rendición de cuentas y propiedad. Se trata de implementar procesos que fomenten la colaboración y una mentalidad de servicio. Permite mecanismos de entrega que aportan agilidad y flexibilidad dentro de la organización. Contrariamente a la creencia popular, DevOps no se trata de herramientas, tecnología o automatización. Son habilitadores que ayudan en la colaboración, la implementación de procesos ágiles y la entrega más rápida y mejor al consumidor.

Hay varias definiciones disponibles en Internet para DevOps y no son correctas ni incorrectas. DevOps no proporciona un marco o metodología. Es un conjunto de principios y prácticas que, cuando se emplean dentro de una organización, interacción o proyecto, logran el objetivo y la visión tanto de DevOps como de la organización. Estos principios y prácticas no exigen ningún proceso, herramientas, tecnologías ni entorno específicos. DevOps proporciona la orientación que puede implementarse a través de cualquier herramienta, tecnología y proceso, aunque parte de la tecnología y los procesos podrían ser más aplicables para lograr la visión de los principios y prácticas de DevOps.

Aunque las prácticas de DevOps se pueden implementar en cualquier organización que ofrezca servicios y productos a los consumidores, en este libro veremos a DevOps desde la perspectiva del desarrollo de software y el departamento de operaciones de cualquier organización.

Entonces, ¿qué es DevOps? DevOps se define como:

Es un conjunto de principios y prácticas que reúne tanto a los desarrolladores como a los equipos de operaciones desde el inicio del sistema de software para una entrega de principio a fin más rápida, veloz y eficiente del sistema de software al consumidor final, una y otra vez de manera coherente y predecible reduciendo el tiempo de comercialización, obteniendo así una ventaja competitiva.

Lea en voz alta la definición anterior de DevOps y, si la observa detenidamente, verá que no indica ni se refiere a ningún proceso, herramienta o tecnología específicos. No está prescribiendo ninguna metodología o entorno particular.

El objetivo de implementar los principios y prácticas de DevOps en cualquier organización es garantizar que las demandas de las partes interesadas (incluidos los consumidores) y las expectativas se cumplan de manera eficaz y efectiva.

Las demandas y expectativas del consumidor se cumplen cuando:

- El consumidor obtiene las características que desea
- El consumidor las obtiene cuando lo desea
- El consumidor obtiene actualizaciones más rápidas de las características
- La calidad de entrega es alta

Cuando una organización puede cumplir con las expectativas anteriores, los consumidores están satisfechos y se mantienen leales a la organización. Esto, a su vez, aumenta la competitividad en el mercado de la organización, lo que se traduce en una mayor marca y valoración del mercado. Tiene un impacto directo en los resultados y las ventas de la organización. La organización puede invertir más en innovación y comentarios de los consumidores, provocando cambios continuos en su sistema y servicios para mantener su relevancia.

La implementación de los principios y prácticas de DevOps en cualquier organización está guiada por el ecosistema circundante. Este ecosistema está compuesto por la industria y los dominios a los que pertenece la organización.

DevOps se basa en un conjunto de principios y prácticas. Veremos detalles sobre estos principios y prácticas más adelante en este capítulo. Los principios básicos de DevOps son los siguientes:

- Agilidad
- Automatización
- Colaboración
- Comentarios

Las prácticas principales de DevOps son las siguientes:

- Integración continua
- Administración de configuración
- Implementación continua
- Entrega continua
- Aprendizaje continuo

DevOps no es un paradigma nuevo, pero recientemente está ganando mucha popularidad y relevancia. Su adopción se encuentra en su nivel más alto y cada vez más empresas están emprendiendo este proceso. Mencioné a propósito DevOps como un proceso porque hay diferentes niveles de madurez dentro de DevOps. Si bien el éxito de la implementación y la entrega continua se considera el nivel más alto de madurez en este proceso, al adoptar el control del código fuente, el desarrollo ágil de software se considera un comienzo.

Una de las primeras cosas de las que habla DevOps es *romper las barreras entre los desarrolladores y el equipo de operaciones*. Aporta el aspecto de colaboración estrecha entre varios equipos. Se trata de cambiar la mentalidad de la que el desarrollador es responsable de escribir solo el código y pasarlo a operaciones para la implementación una vez que se haya probado. También se trata de cambiar la mentalidad de que las operaciones no tienen ningún papel que desempeñar en las actividades de desarrollo. Las operaciones deben influir en la planificación del producto y deben ser conscientes de las características que se publicarán como lanzamiento. También deben proporcionar continuamente comentarios a los desarrolladores sobre los problemas operativos, de modo que puedan solucionarse en versiones posteriores. Deben influir en el diseño del sistema para un mejor funcionamiento operativo del sistema. De manera similar, los desarrolladores deben ayudar a las operaciones en la implementación del sistema y resolver incidentes cuando surjan.

La definición hace referencia a *la entrega más rápida, veloz y eficiente de los sistemas a los interesados*. No habla de lo rápida o eficiente que debería ser la entrega. Debería ser lo suficientemente rápida o veloz según el dominio de la organización, la industria, la segmentación de clientes, etc. Para algunas organizaciones, lo suficientemente rápido podría ser trimestral, mientras que para otras podría ser semanal. Ambos tipos son válidos para el punto de vista de DevOps y se pueden implementar procesos y tecnologías relevantes para lograr lo mismo. DevOps no lo ordena. Las organizaciones deben identificar la mejor implementación de los principios y las prácticas de DevOps en función de su visión general del proyecto, la interacción y la organización.

La definición también habla de *la entrega de un extremo a otro*. Esto significa que desde la planificación y la entrega del sistema a los servicios y operaciones debe ser parte de la implementación de DevOps. Los procesos deben ser los que permitan una mayor flexibilidad, modularidad y agilidad en el ciclo de vida del desarrollo de la aplicación. Mientras que las organizaciones son libres de usar el mejor proceso de ajuste: cascada, Agile, Kanban, etc., por lo general, las organizaciones tienden a favorecer los procesos ágiles con la entrega basada en iteraciones. Esto permite una entrega más rápida en unidades más pequeñas que son mucho más comprobables y manejables en comparación con las grandes entregas.

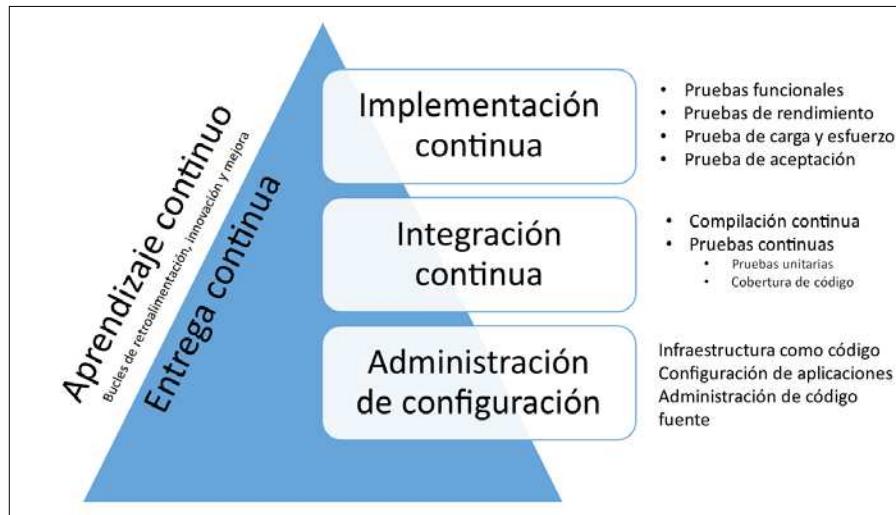
DevOps habla sobre *clientes finales una y otra vez de una manera coherente y predecible*. Esto significa que las organizaciones deben realizar entregas continuamente a los consumidores con funciones más nuevas y actualizadas utilizando la automatización. No podemos lograr consistencia y previsibilidad sin el uso de la automatización. El trabajo manual debe reducirse a ninguno para garantizar un alto nivel de consistencia y previsibilidad. La automatización también debe ser de extremo a extremo, para evitar fallos. Esto también indica que el diseño del sistema debe ser modular, lo que permite una entrega más rápida, ya que son fiables, disponibles y escalables. Las pruebas desempeñan un gran papel en la entrega coherente y predecible.

El resultado final de la implementación de las prácticas y los principios mencionados anteriormente es que la organización puede satisfacer las expectativas y demandas de los clientes. La organización puede crecer más rápido que la competencia y aumentar aún más la calidad y la capacidad de sus productos y servicios a través de la innovación y mejora continuas.

Prácticas de DevOps

DevOps consta de múltiples prácticas, cada una de las cuales proporciona una funcionalidad distinta al proceso general. La siguiente figura muestra la relación entre ellas. La administración de la configuración, la integración continua y la implementación continua forman las prácticas centrales que habilitan DevOps. Cuando entregamos servicios de software combinando estos tres servicios, logramos una entrega continua. La entrega continua es la capacidad y la madurez de una organización que depende de la madurez de la administración de la configuración, la integración continua y la implementación continua. Los comentarios continuos en todas las fases forman el ciclo de comentarios que ayuda a proporcionar servicios superiores a los clientes. Se ejecuta a través de todas las prácticas de DevOps.

Vamos a profundizar en cada una de estas capacidades y prácticas de DevOps:



Administración de configuración

Las aplicaciones y los servicios empresariales necesitan un entorno en el que se puedan implementar. Normalmente, el entorno es una infraestructura que comprende varios servidores, procesos, redes, almacenamiento, contenedores y muchos más servicios que trabajan conjuntamente de modo que las aplicaciones empresariales se puedan implementar sobre ellos. Las aplicaciones empresariales se descomponen en varios servicios que se ejecutan en diferentes servidores, ya sea on-premise o en los clouds; cada servicio tiene su propia configuración junto con los requisitos relacionados con la configuración de la infraestructura. En resumen, tanto la infraestructura como la aplicación son necesarias para entregar sistemas a los consumidores y las dos tienen su propia configuración. Si la configuración se desplaza, la aplicación podría no funcionar como se esperaba, lo que provocaría tiempos de inactividad y fallos. Además, como el proceso ALM dicta el uso de múltiples etapas y entornos, una aplicación se implementaría en varios entornos con diferentes configuraciones. La aplicación se implementará en el entorno de desarrollo para que los desarrolladores vean el resultado de su trabajo. La aplicación se implementará en varios entornos de prueba con diferentes configuraciones para pruebas funcionales, pruebas de carga y esfuerzo, pruebas de rendimiento, pruebas de integración y más, también se implementará en el entorno de preproducción para realizar pruebas de aceptación del usuario y, finalmente, en un entorno de producción. Es importante que una aplicación se pueda implementar en varios entornos sin realizar cambios manuales en su configuración.

La administración de la configuración proporciona un conjunto de procesos y herramientas y ayuda a garantizar que cada entorno y aplicación tengan su propia configuración. La administración de la configuración hace un seguimiento de los elementos de configuración y cualquier cosa que cambie de un entorno a otro debe tratarse como un elemento de configuración. La administración de la configuración también define las relaciones entre los elementos de configuración y cómo los cambios en un elemento de configuración afectarán al otro elemento de configuración.

La administración de la configuración ayuda en lo siguiente:

- **Infraestructura como código:** cuando el proceso de aprovisionamiento de infraestructura y su configuración se representa a través del código y el mismo código pasa por el proceso del ciclo de vida de la aplicación, se conoce como Infraestructura como código. La infraestructura como código ayuda a automatizar el aprovisionamiento y la configuración de la infraestructura. También representa toda la infraestructura en el código que se puede almacenar en un repositorio y una versión controlada. Esto permite a los usuarios utilizar las configuraciones de entorno anteriores cuando sea necesario. También permite el aprovisionamiento de un entorno varias veces de manera coherente y predecible. Todos los entornos aprovisionados de esta manera son coherentes e iguales en todas las etapas de ALM.
- **Implementación y configuración de la aplicación:** la implementación de una aplicación y su configuración es el siguiente paso después del aprovisionamiento de la infraestructura. Ejemplos de implementación y configuración de la aplicación es implementar un paquete webdeploy en un servidor, implementar esquemas y datos de un SQL Server (bacpac) en otro servidor, cambiar la cadena de conexión SQL en el servidor web para representar el SQL Server apropiado. La administración de la configuración almacena valores para la configuración de la aplicación para cada entorno en el que se implementa.

La configuración aplicada también debe ser supervisada. La configuración esperada y deseada debe mantenerse constantemente. Cualquier desplazamiento de esta configuración esperada y deseada haría que la aplicación no estuviera disponible. La administración de la configuración también es capaz de encontrar el desplazamiento y reconfigurar la aplicación y el entorno a su estado deseado.

Con la administración de la configuración automatizada implementada, nadie en el equipo tiene que implementar y configurar los entornos y las aplicaciones en producción. El equipo de operaciones no depende del equipo de desarrollo ni de la documentación de implementación larga.

Otro aspecto de la administración de la configuración es el control del código fuente. Las aplicaciones y los servicios empresariales constan de código y otros artefactos. Varios miembros del equipo trabajan en los mismos archivos. El código fuente debe estar actualizado en cualquier momento y solo debe ser accesible para los miembros del equipo autenticados. El código y otros artefactos por sí mismos son elementos de configuración. El control de código fuente ayuda a la colaboración y la comunicación dentro del equipo, ya que todos son conscientes de lo que está haciendo la otra persona y los conflictos se resuelven en una etapa temprana.

La administración de la configuración se puede dividir en dos categorías:

- Dentro de la máquina virtual
- Fuera de la máquina virtual

Las herramientas disponibles para la administración de la configuración dentro de la máquina virtual se enumeran a continuación.

Desired State Configuration

Desired State Configuration (DSC) es una nueva plataforma de administración de configuración de Microsoft creada como una extensión de PowerShell. DSC se lanzó originalmente como parte de WMF 4.0. Está disponible como parte de **Windows Management Framework (WMF)** 4.0 y 5.0 para todos los sistemas operativos de Windows Server anteriores a Windows 2008 R2. WMF 5.1 está disponible de forma inmediata en Windows Server 2016 y Windows 10.

Chef, Puppet y Ansible

Además de DSC, hay una gran cantidad de herramientas de administración de configuración, como Chef, Puppet y Ansible admitidas por Azure. Los detalles sobre estas herramientas no se incluyen en este libro.

Las herramientas disponibles para la administración de la configuración fuera de una máquina virtual son:

Plantillas de Azure Resource Manager

Las plantillas de ARM son el principal medio de aprovisionamiento de recursos en ARM. Las plantillas de ARM ofrecen un modelo declarativo a través del cual se especifican los recursos, su configuración, los scripts y las extensiones. Estas se basan en el formato **JavaScript Object Notation (JSON)**. Utiliza la sintaxis y convenciones JSON para declarar y configurar recursos. Los archivos JSON son archivos basados en texto, fáciles de usar y fáciles de leer. Se pueden almacenar en un repositorio de código fuente y tienen control de versiones. También son medios para representar la infraestructura como código que se puede usar para aprovisionar recursos en los grupos de recursos de Azure una y otra vez, de manera predecible, coherente y uniforme.

Una plantilla precisa de un grupo de recursos para su implementación. Solo se puede implementar en un grupo de recursos y dicho grupo debe existir antes de ejecutarse la implementación de la plantilla. Una plantilla no puede crear un grupo de recursos.

Las plantillas se caracterizan por su flexibilidad debido a su diseño e implementación genéricos y modulares. Las plantillas ofrecen la posibilidad de aceptar parámetros de los usuarios, declarar variables internas, ayudar a definir dependencias entre recursos, vincular recursos dentro de los mismos o diferentes grupos de recursos y también ejecutar otras plantillas. Ofrecen, además, funciones y expresiones de tipo de lenguaje de scripting que las convierten en herramientas dinámicas y personalizables en el tiempo de ejecución.

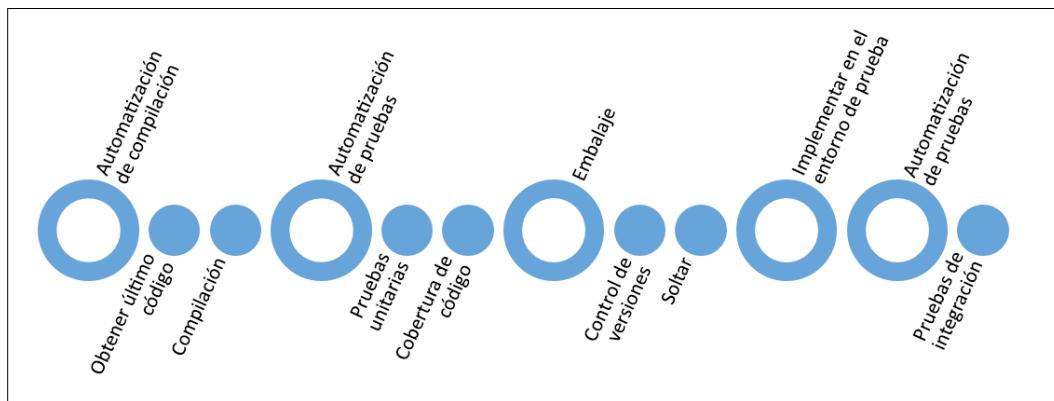
Integración continua

Varios desarrolladores escriben código que finalmente se almacena en un repositorio común. Normalmente, el código se registra o se envía al repositorio cuando el desarrollador ha terminado de desarrollar su función. Esto puede suceder en un día o puede tardar días o semanas. Algunos de los desarrolladores podrían estar trabajando en la misma función y también podrían seguir las mismas prácticas de empujar/registrar el código en días o semanas. Esto puede causar problemas con la calidad del código. Uno de los principios de DevOps es fallar rápido. Los desarrolladores deben registrar/enviar su código al repositorio a menudo y compilar el código para verificar si no ha introducido ningún error y si el código es compatible con el código escrito por su compañero. Si el desarrollador no sigue esta práctica, entonces el código en su máquina crecerá mucho y será difícil de integrar con el código de otros. Además, si la compilación falla, es difícil y lleva mucho tiempo solucionar los problemas que surgen de ella.

La integración continua resuelve este tipo de desafíos. La integración continua ayuda en la compilación y validación del código introducido/registrado por un desarrollador al llevarlo a cabo a través de una serie de pasos de validación. La integración continua crea un flujo de proceso que consta de varios pasos. La integración continua se compone de compilación automatizada continua y pruebas automatizadas continuas. Normalmente, el primer paso es la compilación del código. Después de la compilación exitosa, cada paso es responsable de validar el código desde una perspectiva específica. Por ejemplo, las pruebas unitarias pueden ejecutarse en el código compilado, la cobertura del código puede ejecutarse para verificar qué rutas de código se ejecutan mediante pruebas unitarias. Podrían revelar si se escriben pruebas unitarias exhaustivas o si hay margen para agregar más pruebas unitarias. El resultado final de la integración continua son los paquetes de implementación que pueden usarse mediante la implementación continua para implementarlos en múltiples entornos.

Se recomienda a los desarrolladores que revisen su código varias veces en un día en lugar de hacerlo después de días o semanas. La integración continua iniciaría la ejecución de toda la canalización inmediatamente tan pronto como se introduzca o se inserte el código. Si la compilación se realiza correctamente, las pruebas de código y otras actividades que forman parte de la canalización se ejecutan sin errores, el código se implementa en un entorno de prueba y se ejecutan pruebas de integración en él. Aunque cada sistema exige su propia configuración de integración continua, en la siguiente figura se muestra una integración continua de muestra mínima.

La integración continua aumenta la productividad de los desarrolladores. No tienen que compilar manualmente su código, ejecutar varios tipos de pruebas una tras otra y luego crear paquetes a partir de él. También reduce el riesgo de que se introduzcan errores en el código y el código no se vuelve obsoleto. Proporciona comentarios tempranos a los desarrolladores sobre la calidad de su código. En general, la calidad de los entregables es alta y los entregables se entregan más rápido al adoptar una práctica de integración continua:



Automatización de compilación

La automatización de compilación consta de varias tareas que se ejecutan secuencialmente. En general, la primera tarea es responsable de obtener el último código fuente del repositorio. El código fuente puede comprender múltiples proyectos y archivos. Se compilan para generar artefactos como ejecutables, bibliotecas de enlaces dinámicos, ensamblados, etc. La automatización de compilación exitosa refleja que no hay errores de tiempo de compilación en el código.

Podría haber más pasos en la automatización de compilación dependiendo de la naturaleza y el tipo de proyecto.

Automatización de pruebas

La automatización de pruebas consiste en tareas que son responsables de validar diferentes aspectos del código. Estas tareas están relacionadas con el código de prueba desde una perspectiva diferente y se ejecutan secuencialmente. Generalmente, el primer paso es ejecutar una serie de pruebas unitarias en el código. Las pruebas unitarias se refieren al proceso de probar la denominación más pequeña de una característica que valida su comportamiento de forma aislada de otras características. Puede ser automatizada o manual, sin embargo, la preferencia es hacia pruebas unitarias automatizadas.

La cobertura de código es otro tipo de prueba automatizada que se puede ejecutar en el código para averiguar cuánto código se ejecuta mientras se ejecutan las pruebas unitarias. Normalmente se representa como un porcentaje y se refiere a la cantidad de código que se puede probar a través de la prueba unitaria. Si la cobertura del código no es cercana al 100 %, es porque el desarrollador no ha escrito pruebas unitarias para ese comportamiento o el código no cubierto no es necesario en absoluto.

La ejecución exitosa de la automatización de pruebas que resulte en un fallo de código no significativo debe comenzar a ejecutar las tareas de empaquetado. Podría haber más pasos en la automatización de pruebas dependiendo de la naturaleza y el tipo de proyecto.

Empaquetado

El empaquetado se refiere al proceso de generación de artefactos desplegables, como los paquetes MSI, NuGet y webdeploy, los paquetes de base de datos, la versión y el almacenamiento en una ubicación para que puedan ser consumidos por otras canalizaciones y procesos.

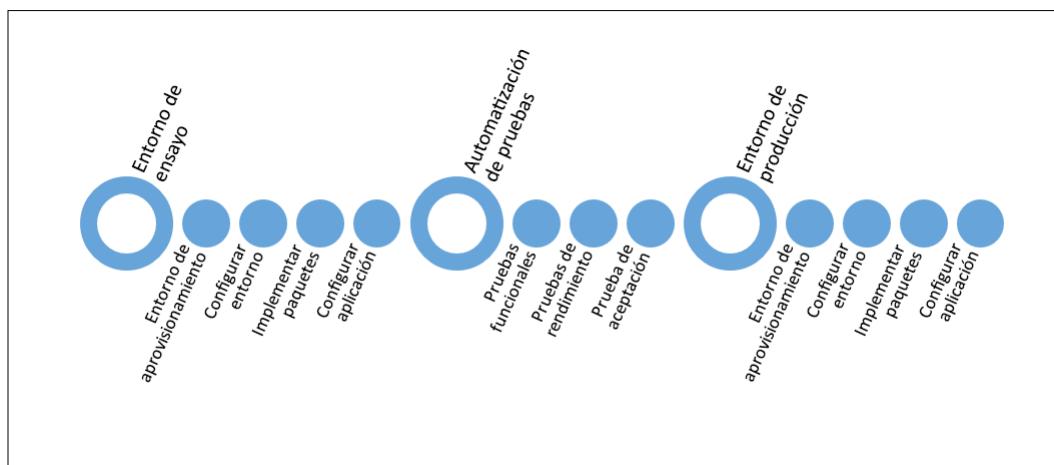
Implementación continua

En el momento en que el proceso alcanza la implementación continua, la integración continua ha asegurado que tengamos bits de trabajo completos de una aplicación que ahora puede tomarse a través de diferentes actividades de implementación continua. La implementación continua se refiere a la capacidad de implementar aplicaciones y servicios empresariales en entornos de preproducción y producción a través de la automatización. Por ejemplo, la implementación continua podría aprovisionar y configurar el entorno de preproducción, implementar la aplicación y configurarla. Después de realizar varias validaciones, como pruebas funcionales, pruebas de rendimiento en un entorno de preproducción, el entorno de producción se aprovisiona, configura y la aplicación se implementa en entornos de producción a través de la automatización. No hay pasos manuales en el proceso de implementación. Cada tarea de implementación es automatizada. La implementación continua puede aprovisionar el entorno e implementar la aplicación para una implementación completa, mientras que podría reutilizar el entorno existente y realizar solo la implementación de la aplicación si el entorno ya existe. Sin embargo, siempre es mejor llevar a cabo la implementación green field puramente metálica; la justificación empresarial puede hacer la demanda para implementaciones brown field.

Todos los entornos se aprovisionan a través de la automatización utilizando la infraestructura como código. Esto garantiza que todos los entornos, ya sean de desarrollo, prueba, preproducción, producción y cualquier otro entorno, sean iguales. De manera similar, la aplicación se implementa a través de la automatización, lo que garantiza que también se implementa de manera uniforme en todos los entornos. La configuración en estos entornos podría ser diferente para la aplicación.

La implementación continua generalmente se integra con la integración continua. Cuando la integración continua ha realizado su trabajo al generar los paquetes desplegables finales, la implementación continua se activa y comienza su propia canalización. La canalización se denomina **canalización de versiones**. La canalización de versiones consta de varios entornos; cada entorno consiste en tareas responsables de la provisión del entorno, la configuración del entorno, la implementación de aplicaciones, la configuración de aplicaciones, la ejecución de la validación operativa en entornos y la prueba de la aplicación en múltiples entornos. Examinaremos la canalización de versiones con mayor detalle en el siguiente capítulo y también el capítulo sobre la implementación continua.

Emplear una implementación continua proporciona inmensos beneficios. Existe un alto nivel de confianza en el proceso de implementación general que ayuda a las versiones de producción más rápidas y sin riesgos. Las posibilidades de que algo salga mal se reducen drásticamente. El equipo tendría niveles de estrés más bajos y la reversión al entorno de trabajo anterior es posible si hay problemas en la versión actual:



Aunque cada sistema exige su propia configuración de canalización de versiones, en la figura anterior se muestra un ejemplo mínimo de implementación continua. Es importante tener en cuenta que, en general, el aprovisionamiento y la configuración de múltiples entornos son parte del proceso de canalización y las aprobaciones deben buscarse antes de pasar al siguiente entorno. El proceso de aprobación puede ser manual o automatizado dependiendo de la madurez de la organización.

Implementación del entorno de prueba

La canalización de versiones comienza una vez que la colocación está disponible desde la integración continua y el primer paso que debe tomar es obtener todos los artefactos de la colocación. Después de esto, puede crear un entorno de prueba completo totalmente nuevo o reutilizar uno existente. De nuevo, esto depende del tipo de proyecto y la naturaleza de las pruebas planeadas para ejecutarse en este entorno. El entorno está aprovisionado y configurado. Los artefactos de la aplicación están implementados y configurados.

Automatización de pruebas

Después de implementar una aplicación, se pueden realizar una serie de pruebas en el entorno. Una de las pruebas que se ejecutan aquí son las pruebas funcionales. Las pruebas funcionales tienen como objetivo principal validar la integridad y funcionalidad de la aplicación. Estas pruebas están escritas desde los requisitos recopilados del consumidor. Otro conjunto de pruebas que se pueden ejecutar están relacionadas con la escalabilidad y la disponibilidad de la aplicación. Normalmente, esto incluye pruebas de carga, pruebas de estrés y pruebas de rendimiento. También debe incluir la validación operativa del entorno de infraestructura.

Implementación del entorno de ensayo

Esto es muy similar a la implementación del entorno de prueba, ya que la única diferencia es que los valores de configuración para el entorno y la aplicación serían diferentes.

Pruebas de aceptación

Normalmente, las pruebas de aceptación las realizan los interesados de la aplicación y pueden ser manuales o automatizadas. Este paso es una validación desde el punto de vista del consumidor sobre la corrección y la integridad de la funcionalidad de la aplicación.

Implementación en producción

Una vez que el cliente proporciona su aprobación, se ejecutan los mismos pasos que los de la implementación del entorno de prueba y ensayo, con la única diferencia de que los valores de configuración para el entorno y la aplicación son específicos para el entorno de producción. Se lleva a cabo una validación después de la implementación para garantizar que la aplicación se ejecuta de acuerdo con las expectativas.

Entrega continua

La entrega continua y la implementación continua pueden parecerles similares a muchos lectores; sin embargo, no son lo mismo. Si bien la implementación continua habla sobre la implementación en varios entornos y, finalmente, en el entorno de producción a través de la automatización, las prácticas de entrega continua son la capacidad de generar paquetes de aplicaciones de una manera que se puedan implementar fácilmente en cualquier entorno. Para generar artefactos que se pueden implementar fácilmente, se debe usar la integración continua para generar los artefactos de la aplicación, se debe usar un entorno nuevo o existente para implementar estos artefactos, realizar pruebas funcionales, pruebas de rendimiento y pruebas de aceptación del usuario mediante la automatización. Una vez que estas actividades se ejecutan con éxito y sin errores, el paquete de la aplicación se conoce como fácilmente implementable. La entrega continua comprende la integración continua junto con la implementación en un entorno para validaciones finales. Ayuda a obtener comentarios más rápidamente tanto de las operaciones como del usuario final. Estos comentarios se pueden utilizar para implementar iteraciones posteriores.

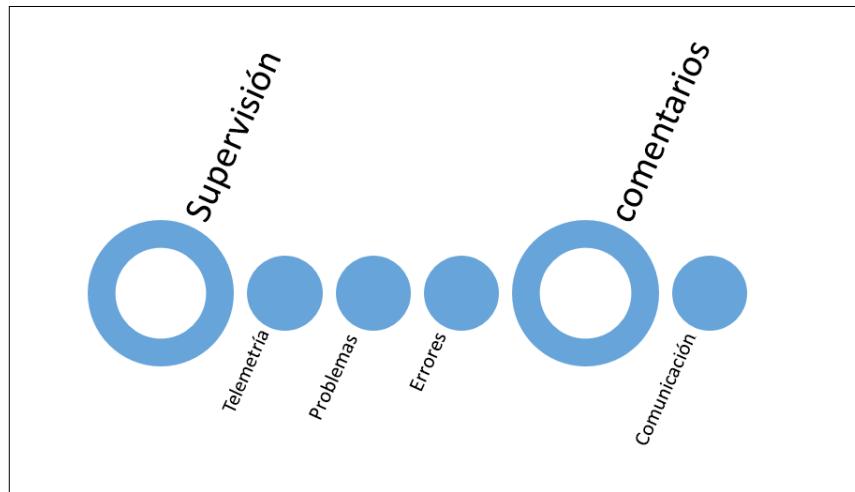
Aprendizaje continuo

Con todas las prácticas de DevOps mencionadas anteriormente, es posible crear aplicaciones empresariales excelentes e implementarlas automáticamente en el entorno de producción; sin embargo, los beneficios de DevOps no durarán mucho tiempo si no se implementan los principios de mejora continua y comentarios. Es de suma importancia que los comentarios en tiempo real sobre el comportamiento de la aplicación se transmitan como comentarios al equipo de desarrollo tanto de los usuarios finales como del equipo de operaciones.

Los comentarios se deben transmitir a los equipos que proporcionan información relevante sobre lo que está funcionando bien y lo que no está funcionando bien.

Las aplicaciones deben construirse teniendo en cuenta la supervisión, la auditoría y la telemetría. La arquitectura y el diseño deben admitir todo esto. El equipo de operaciones debe recopilar la información de telemetría del entorno de producción, capturar todos los errores y problemas y transmitirlo todo al equipo de desarrollo para que puedan subsanarse en las versiones posteriores. En la figura siguiente se muestra lo mismo.

El aprendizaje continuo ayuda a que la aplicación sea robusta y resistente a los fallos. Ayuda a garantizar que la aplicación cumpla con los requisitos de los consumidores:



Visual Studio Team Services

Ahora es el momento de centrarse en otro servicio online revolucionario, **Visual Studio Team Services** (VSTS), que permite la integración continua, la implementación continua y la entrega continua sin problemas. De hecho, sería más apropiado llamarlo un conjunto de servicios disponibles con un solo nombre. VSTS es un PaaS proporcionado por Microsoft que se aloja en el cloud. El mismo servicio está disponible como **Team Foundation Services** (TFS) on-premise. Todos los ejemplos utilizados en este libro utilizan VSTS.

Según Microsoft, VSTS es una plataforma de colaboración basada en el cloud que ayuda a los equipos a compartir código, realizar un seguimiento del trabajo y enviar software. VSTS es el nombre nuevo; anteriormente se denominaba **Visual Studio Online** (VSO). VSTS es un servicio y una herramienta de desarrollo de software empresarial que permite a las organizaciones proporcionar instalaciones de automatización a su proceso integral de gestión del ciclo de vida de la aplicación, desde la planificación hasta la implementación de la aplicación y obtener comentarios en tiempo real de los sistemas de software. Esto aumenta la madurez y la capacidad de una organización para entregar sistemas de software de alta calidad a sus clientes una y otra vez.

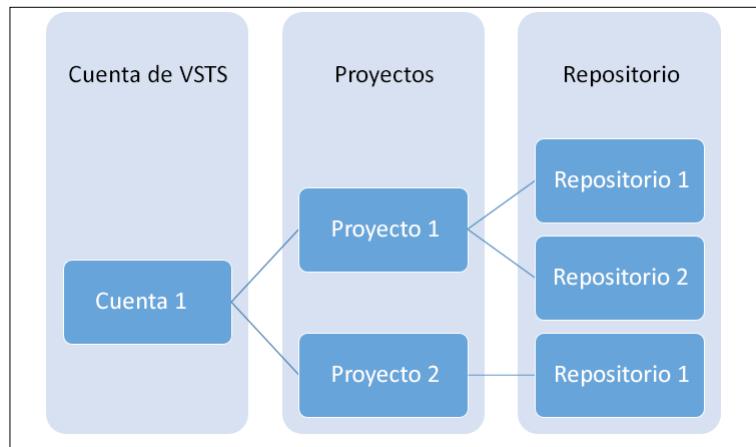
La entrega exitosa de software implica unir eficientemente numerosos procesos y actividades. Estos incluyen la ejecución e implementación de diversos procesos ágiles, el aumento de la colaboración entre equipos, la transición sin problemas y automática de los artefactos de una fase de ALM a otra fase y las implementaciones en múltiples entornos. Es importante hacer un seguimiento e informar sobre estas actividades para medir y tomar medidas y mejorar los procesos de entrega. VSTS lo facilita. Proporciona un conjunto completo de servicios que permite lo siguiente:

- Colaboración entre todos los miembros del equipo al proporcionar una interfaz única para la gestión completa del ciclo de vida de la aplicación.
- Colaboración entre los equipos de desarrollo utilizando servicios de gestión de código fuente.
- Colaboración entre equipos de pruebas utilizando servicios de administración de pruebas.
- Validación automatizada de código y empaquetado a través de la integración continua utilizando servicios de administración de compilación.
- Automatización de la validación de la funcionalidad de la aplicación, la implementación y la configuración de múltiples entornos a través de la implementación y entrega continua utilizando los servicios de administración de versiones.
- Seguimiento y administración de elementos de trabajo mediante servicios de administración de trabajo.

La siguiente figura muestra todos los servicios disponibles en la barra de navegación superior de VSTS:



Un proyecto en VSTS es un límite de seguridad y un contenedor lógico que proporciona todos los servicios que mencionamos en la sección anterior. VSTS permite la creación de varios proyectos dentro de una sola cuenta. De forma predeterminada, se crea un repositorio con la creación de un proyecto, sin embargo, VSTS permite la creación de repositorios adicionales dentro de un solo proyecto. La relación entre la cuenta, el proyecto y el repositorio de VSTS se muestra en la siguiente figura:



Relación entre cuenta, proyectos y repositorios de VSTS.

VSTS proporciona dos tipos de repositorios:

- GIT
- **Control de versiones de Team Foundation (TFVC)**

También proporciona la flexibilidad de elegir entre el repositorio de control de origen GIT o TFVC. Puede haber una combinación de repositorios TFS y TFVC disponibles dentro de un solo proyecto.

Control de versiones de Team Foundation

TFVC es la forma tradicional y centralizada de implementar el control de versiones en el que hay un repositorio central y los desarrolladores trabajan directamente en el modo conectado para registrar sus cambios. Si el repositorio central está sin conexión o no está disponible, los desarrolladores no pueden registrar su código y deben esperar a que esté online y disponible. Otros desarrolladores solo pueden ver el código introducido. Los desarrolladores pueden agrupar varios cambios en un único conjunto de cambios para registrar los cambios en el código que se agrupan lógicamente para formar un solo cambio. TFVC bloquea los archivos de código que están en proceso de edición. Otros desarrolladores pueden leer el archivo bloqueado pero no pueden editarlos. Deben esperar a que se complete la edición anterior y liberar el bloqueo antes de poder editar. El historial de registros y cambios se mantiene en el repositorio central, mientras que los desarrolladores tienen la copia de trabajo de los archivos pero no el historial.

TFVC funciona muy bien con equipos grandes que trabajan en los mismos proyectos. Esto permite el control sobre el código fuente en una ubicación central. También funciona mejor cuando el proyecto es de larga duración ya que el historial se administra en una ubicación central. TFVC no tiene problemas para trabajar con archivos grandes y binarios.

GIT

Por otro lado, GIT es una forma moderna y distribuida de implementar el control de versiones donde los desarrolladores pueden trabajar en sus propias copias locales de código e historial en modo sin conexión. Los desarrolladores pueden trabajar sin conexión en su clon local de código. Cada desarrollador tiene una copia local de código e historial completo y trabajan en sus cambios con este repositorio local. Pueden enviar su código al repositorio local. Se pueden conectar al repositorio central para la sincronización de su repositorio local según sea necesario. Esto permite que todos los desarrolladores trabajen en cualquier archivo, ya que estarían trabajando en su copia local. La ramificación en GIT no crea otra copia del código original y es de creación extremadamente rápida.

GIT funciona bien con un equipo más pequeño. Con equipos más grandes, hay una gran sobrecarga para administrar múltiples solicitudes de extracción para fusionar el código en un repositorio central. También funciona mejor para proyectos de menor duración, ya que el historial no sería demasiado grande para ser descargado y manejable en el repositorio local de cada desarrollador. La ramificación y la fusión es algo sencillo con opciones avanzadas.

GIT es la forma recomendada de usar el control de código fuente debido a la funcionalidad enriquecida que proporciona. Usaremos GIT como el repositorio de nuestra aplicación de muestra en este libro.

Preparación de DevOps

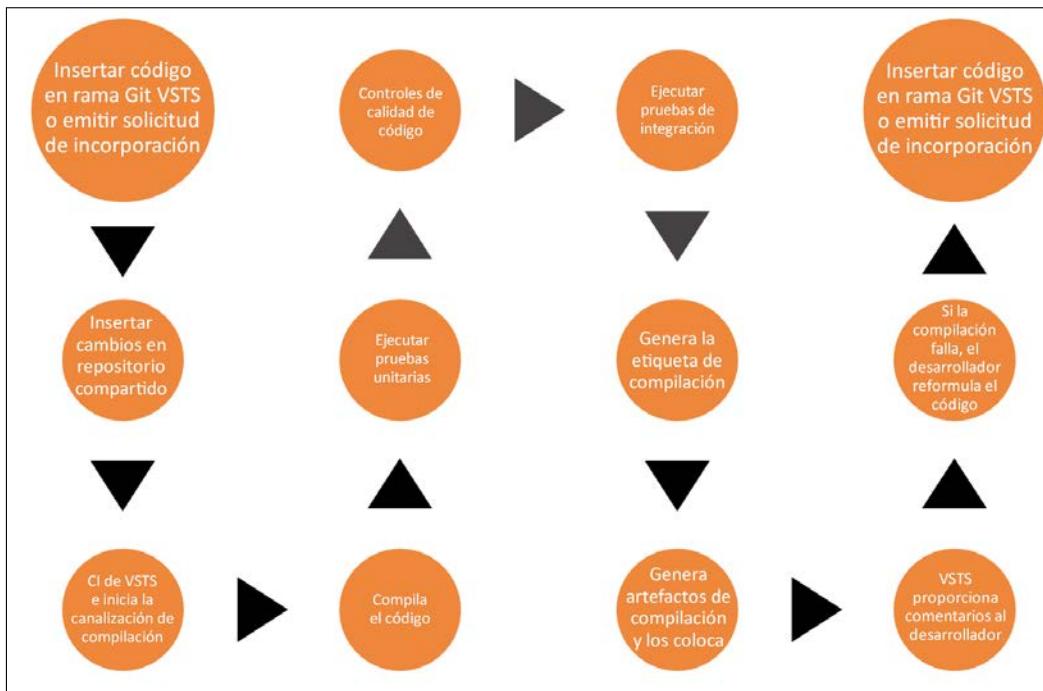
En este capítulo y de aquí en adelante, el enfoque estará en la automatización de procesos y la implementación utilizando diferentes patrones en Azure. Se incluyen los siguientes:

- DevOps para soluciones de IaaS
- DevOps para soluciones de PaaS
- DevOps para soluciones basadas en contenedor

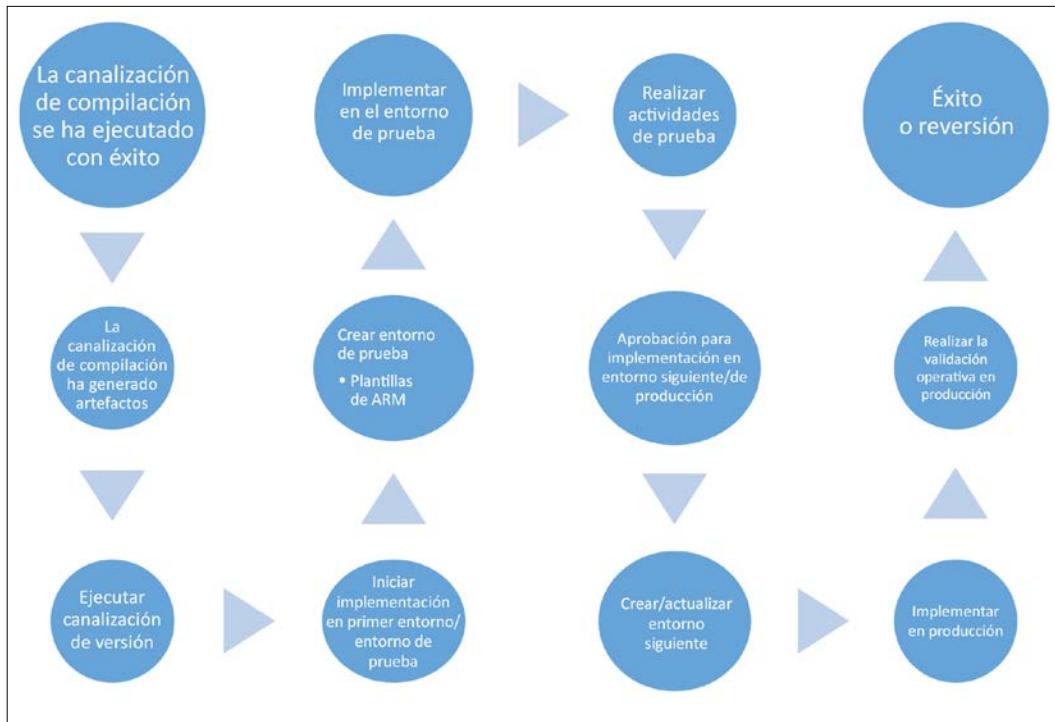
En general, hay servicios compartidos comunes que no son exclusivos de ninguna aplicación. Sus servicios son consumidos por múltiples aplicaciones de diferentes entornos como desarrollo, pruebas y producción. El ciclo de vida de estos servicios compartidos comunes es diferente de otras aplicaciones. Por lo tanto, tienen diferentes cuentas de control de versión, una base de código diferente, administración de compilación y lanzamiento. Tienen su propio ciclo de plan-diseño-construcción-prueba y lanzamiento.

Los recursos que forman parte de este grupo se aprovisionan utilizando plantillas de ARM, PowerShell y configuraciones DSC.

El flujo general para construir estos componentes comunes se muestra a continuación:



El proceso de liberación se muestra en el siguiente diagrama:



Para iniciar el viaje de DevOps, es importante comprender y proporcionar los componentes y servicios comunes antes de iniciar cualquier compromiso, producto o servicio de software.

Aprovisionamiento de cuenta de VSTS

Se necesita un sistema de control de versiones para colaborar en el nivel de código. VSTS ayuda a proporcionar versiones tanto centralizadas como descentralizadas de los sistemas de control. VSTS también proporciona servicios de orquestación para construir y ejecutar canalizaciones de compilación y versión. Es una plataforma madura para organizar todos los artefactos relacionados con el control de versiones, compilación y lanzamiento y elementos de trabajo relacionados con DevOps. Una vez que la cuenta está disponible, se debe crear un proyecto de VSTS para contener todos los artefactos relacionados con el proyecto.

Se puede aprovisionar una cuenta de VSTS visitando <https://www.visualstudio.com>.

Aprovisionamiento de Azure Key Vault

No es recomendable almacenar secretos, certificados, credenciales y otra información confidencial en archivos de configuración de código en archivos, bases de datos o cualquier otro sistema de almacenamiento general. Se recomienda almacenar estos datos importantes en un almacén diseñado específicamente para almacenar secretos y credenciales. Azure Key Vault proporciona este servicio. Azure Key Vault está disponible como un recurso y servicio de Azure.

Aprovisionamiento de un servidor de administración de configuración

Un servidor de administración de la configuración que proporciona almacenamiento para las configuraciones y que aplica esas configuraciones a diferentes entornos siempre es una buena estrategia para automatizar implementaciones. DSC en máquinas virtuales personalizadas, DSC de Azure Automation, Chef, Puppet y Ansible son algunas de las opciones y se pueden usar en Azure sin problemas tanto para entornos de Windows como de Linux. Este libro utiliza DSC como herramienta de administración de la configuración para todos los propósitos y proporciona un servidor de extracción que contiene todos los documentos de configuración (archivos MOF) para la aplicación de muestra. También mantiene la base de datos de todas las máquinas virtuales y contenedores que están configurados y registrados con el servidor de extracción para extraer los documentos de configuración. El administrador de configuración local en estos contenedores y máquinas virtuales de destino verifica periódicamente la disponibilidad de nuevas configuraciones, así como las desviaciones en la configuración actual e informa al servidor de extracción de los mismos. También tiene capacidades de generación de informes incorporadas que proporcionan información sobre los nodos que cumplen con los requisitos y los que no son compatibles con una máquina virtual. Un servidor de extracción es una aplicación web general que aloja el punto de conexión del servidor de extracción de DSC.

Aprovisionamiento de Log Analytics

Log Analytics es un servicio de auditoría y supervisión proporcionado por Azure para obtener información en tiempo real sobre todos los cambios, desviaciones, eventos que ocurren dentro de máquinas virtuales y contenedores. Proporciona un espacio de trabajo centralizado y un panel de control para los administradores de TI para ver, buscar y realizar búsquedas detalladas sobre todos los cambios, desviaciones y eventos que ocurren en estas máquinas virtuales. También proporciona agentes que se implementan en máquinas virtuales y contenedores de destino. Una vez implementados, estos agentes comienzan a enviar todos los cambios, eventos y desviaciones al espacio de trabajo centralizado.

Cuenta de Azure Storage

Azure Storage es un servicio proporcionado por Azure para almacenar archivos como blobs. Todos los scripts y el código para automatizar el aprovisionamiento, la implementación y la configuración de la infraestructura y la aplicación de muestra se almacenan en un repositorio git de VSTS y se empaquetan y se implementan en una cuenta de Azure Storage. Azure proporciona recursos de extensión de script de PowerShell que pueden descargar automáticamente los scripts DSC y PowerShell y ejecutarlos en máquinas virtuales durante la ejecución de las plantillas del administrador de recursos de Azure. Este almacenamiento actúa como un almacenamiento común en todas las implementaciones para múltiples aplicaciones.

Imágenes

Las imágenes de la máquina virtual y del contenedor deben construirse como parte de la canalización de compilación y versiones de servicios comunes. Se pueden usar herramientas como Packer, Docker para generar estas imágenes.

Herramientas de supervisión

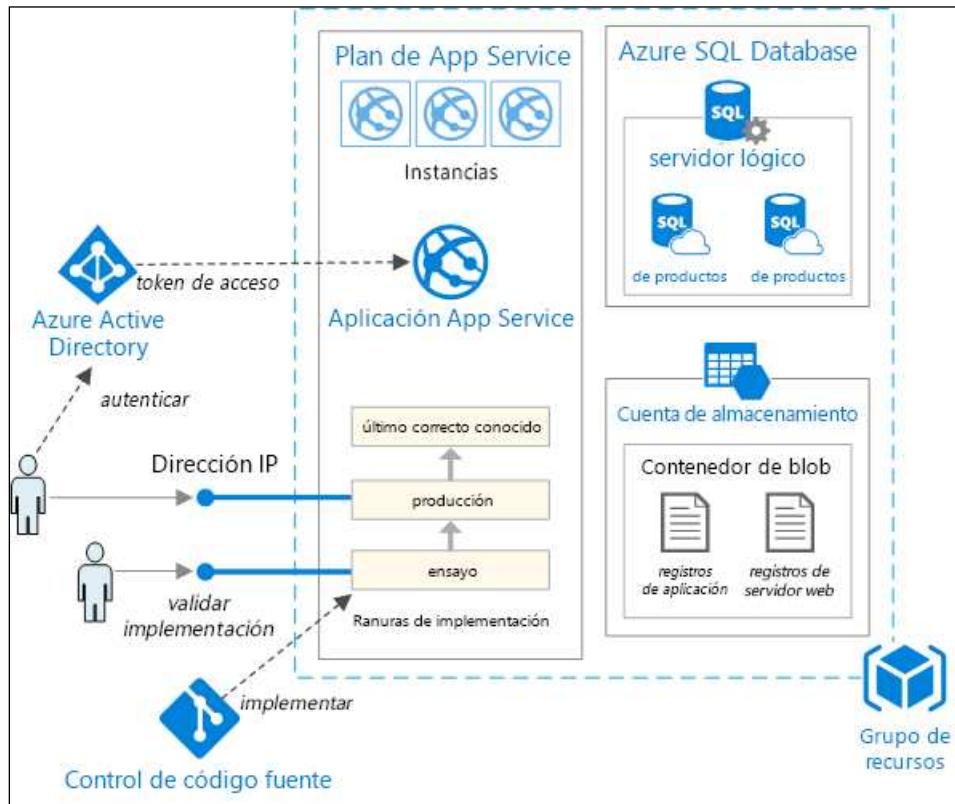
Todas las herramientas de supervisión, como el Azure Monitor, información de aplicaciones, Log Analytics, OMS, administrador de operaciones del centro del sistema deben aprovisionarse y configurarse durante el lanzamiento de la distribución de servicios comunes.

Herramientas de administración

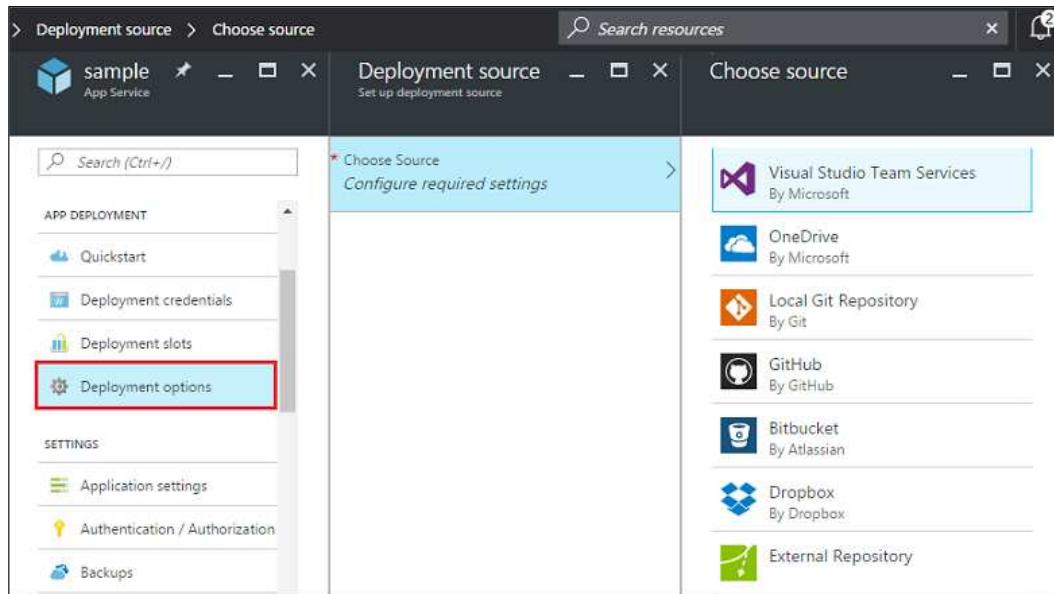
Todas las herramientas de administración, como Kubernetes, DC/OS, Docker Swarm, ITIL deben aprovisionarse en esta etapa.

DevOps para soluciones de PaaS

La arquitectura típica de los servicios de aplicación de PaaS de Azure se basa en la solución que se muestra a continuación:



La arquitectura muestra algunos de los componentes importantes, como Azure SQL, cuentas de almacenamiento, sistema de control de versiones y más que participan en la arquitectura de la solución de cloud basada en Azure App Services. Estos artefactos deben crearse utilizando plantillas de Azure Resource Manager. Estas plantillas de ARM deben formar parte de la estrategia general de administración de la configuración. Puede tener sus propias canalizaciones de administración de compilación y versiones muy similares a las que se muestran en la sección anterior:



La plantilla de ARM también debe configurar la implementación continua mediante la configuración de las **opciones de implementación**.

Azure App Services

Azure App Services proporciona servicios de alojamiento administrado para soluciones de cloud. Es una plataforma totalmente administrada para aprovisionar e implementar soluciones de cloud. App Services quitan la carga de crear y administrar infraestructura y proporcionan **acuerdos de nivel de servicio (SLA)** mínimos para alojar sus soluciones de cloud. Son abiertos y permiten que los usuarios decidan la elección del idioma que desean usar para generar sus soluciones de cloud y flexibles para alojar la solución de cloud en sistemas operativos Windows o Linux. La creación de un servicio de aplicaciones y el hosting de soluciones de cloud aprovisionan máquinas virtuales en segundo plano que están totalmente administradas por Azure y los usuarios no las ven. Múltiples tipos de soluciones de cloud, como aplicaciones web, API de back-end móvil, puntos de conexión de API y contenedores, pueden alojarse sin problemas en Azure App Services.

Ranuras de implementación

Azure App Services proporciona ranuras de implementación que facilitan y simplifican la implementación. Hay una ranura de producción y ensayo que permite a los usuarios intercambiarlas fácilmente. Esto ayuda a implementar primero la solución de cloud personalizada para el ensayo y después de todas las comprobaciones y pruebas, se pueden cambiar a producción si se encuentran satisfactorias. Sin embargo, en caso de cualquier problema en la producción después del intercambio, los buenos valores anteriores del entorno de producción se pueden restablecer mediante el intercambio.

Azure SQL

Azure SQL es el servicio PaaS SQL proporcionado por Azure para alojar bases de datos. Azure proporciona una plataforma segura para alojar bases de datos y toma toda la propiedad para administrar la disponibilidad, fiabilidad y escalabilidad del servicio. Con Azure SQL, no es necesario aprovisionar máquinas virtuales personalizadas, implementar un SQL Server y configurarlo. En su lugar, el equipo de Azure hace esto en segundo plano y las administra en nuestro nombre. También proporciona un servicio de firewall que permite la seguridad y solo una dirección IP permitida por el firewall puede conectar el servidor y acceder a la base de datos. Las máquinas virtuales aprovisionadas para alojar aplicaciones web tienen distintas direcciones IP públicas asignadas y se agregan dinámicamente a las reglas de firewall de Azure SQL. Azure SQL Server y su base de datos se crean al ejecutar la plantilla de Azure Resource Manager. Cabe señalar que, en general, las máquinas virtuales de administración deben ejecutar estas tareas.

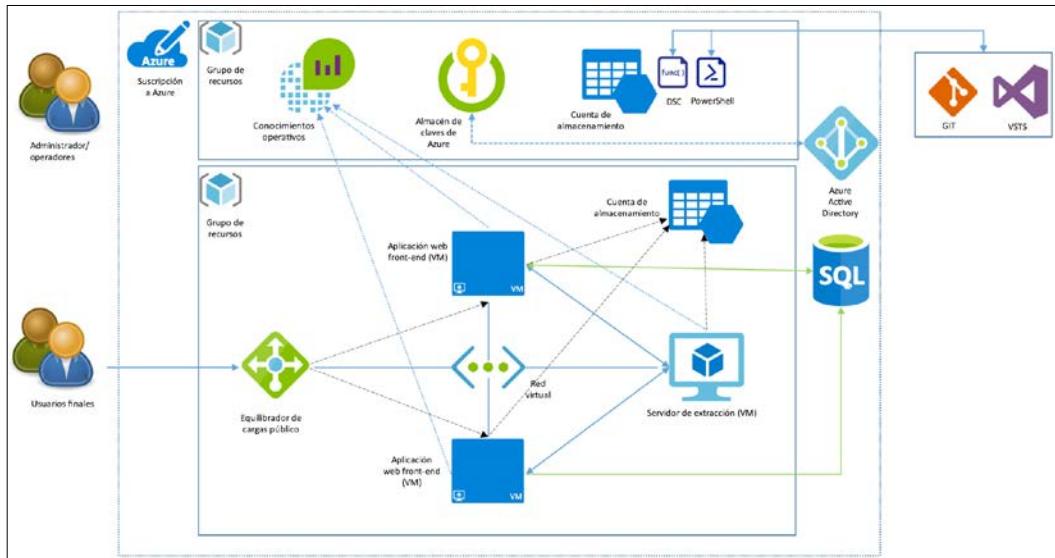
Canalización de versión y compilación

Azure App Services proporciona servicios listos para usar para la implementación continua y deben usarse para implementar soluciones de cloud personalizadas como se muestra a continuación. Se deben proporcionar detalles sobre la cuenta, el repositorio y la rama. Azure App Services se conectaría al repositorio y obtendría el último código fuente de la rama provista y ejecutaría los pasos de compilación para compilar y generar artefactos desplegables. También se implementará en las ranuras de implementación de ensayo automático.

Visual Studio en el extremo del desarrollador debe configurarse para conectarse al sistema de control de versiones. Azure App Services deben iniciar el proceso de creación siempre que un desarrollador registre o introduzca su código en la rama del repositorio.

DevOps para soluciones basadas en máquinas virtuales (IaaS)

La arquitectura típica para la solución basada en máquinas virtuales de IaaS se muestra a continuación:



Máquina virtual de Azure

Las máquinas virtuales de Azure que alojan aplicaciones web, servidores de aplicaciones, bases de datos y otros servicios se aprovisionan utilizando plantillas de ARM. Cada máquina virtual tiene una tarjeta de red individual con una IP pública asignada. Se adjuntan a una red virtual y tienen una dirección IP privada de la misma red. La IP pública para máquinas virtuales es opcional, ya que están conectadas a un equilibrador de carga público. Estas máquinas virtuales se basan en una imagen de servidor de Windows 2016. Los agentes de información operativa se instalan en máquinas virtuales para supervisar las máquinas virtuales. Los scripts de PowerShell también se ejecutan en estas máquinas virtuales descargadas de una cuenta de almacenamiento disponible en otro grupo de recursos para abrir puertos de firewall relevantes, descargar paquetes apropiados e instalar certificados locales para asegurar el acceso a través de PowerShell. La aplicación web está configurada para ejecutarse en el puerto provisto en estas máquinas virtuales. El número de puerto para la aplicación web y toda su configuración se obtiene del servidor de extracción DSC y se asigna dinámicamente.

Equilibrador de carga público de Azure

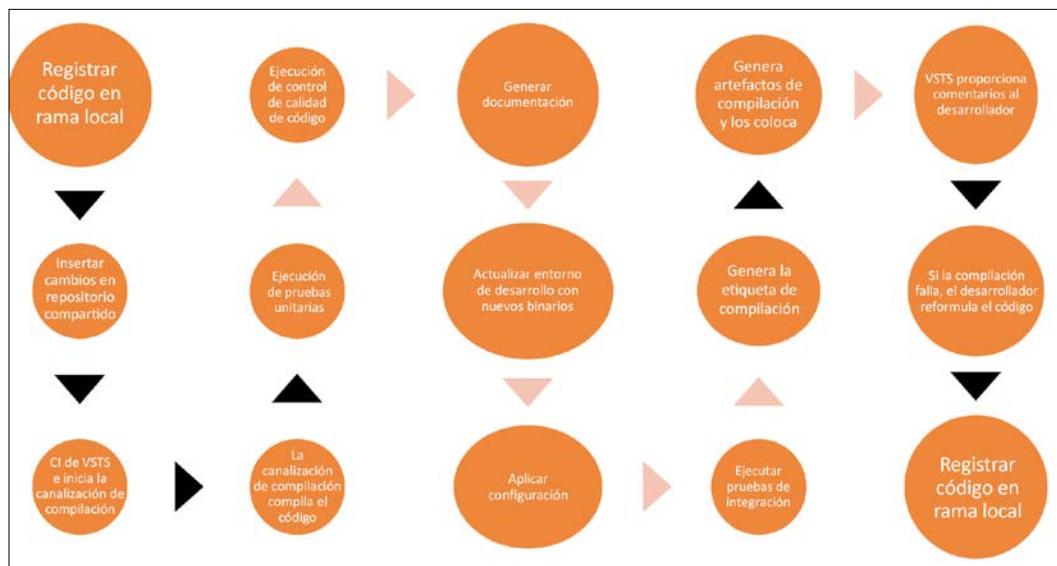
Se adjunta un equilibrador de carga público a algunas de las máquinas virtuales para enviarles solicitudes de forma round robin. Normalmente, esto es necesario para las aplicaciones web front-end y las API. Se puede asignar una dirección IP pública y un nombre DNS al equilibrador de carga para que pueda atender las solicitudes de Internet. Acepta solicitudes web HTTP en diferentes puertos y enruta los mismos a las máquinas virtuales. También investiga determinados puertos en protocolos HTTP con algunas rutas de aplicación proporcionadas. Las reglas de **traducción de direcciones de red (NAT)** también se pueden aplicar de manera que se puedan usar para iniciar sesión en las máquinas virtuales mediante escritorios remotos.

Un recurso alternativo al equilibrador de carga pública de Azure es la puerta de enlace de la aplicación de Azure y, según el escenario, se puede usar e implementar.

Canalización de compilación

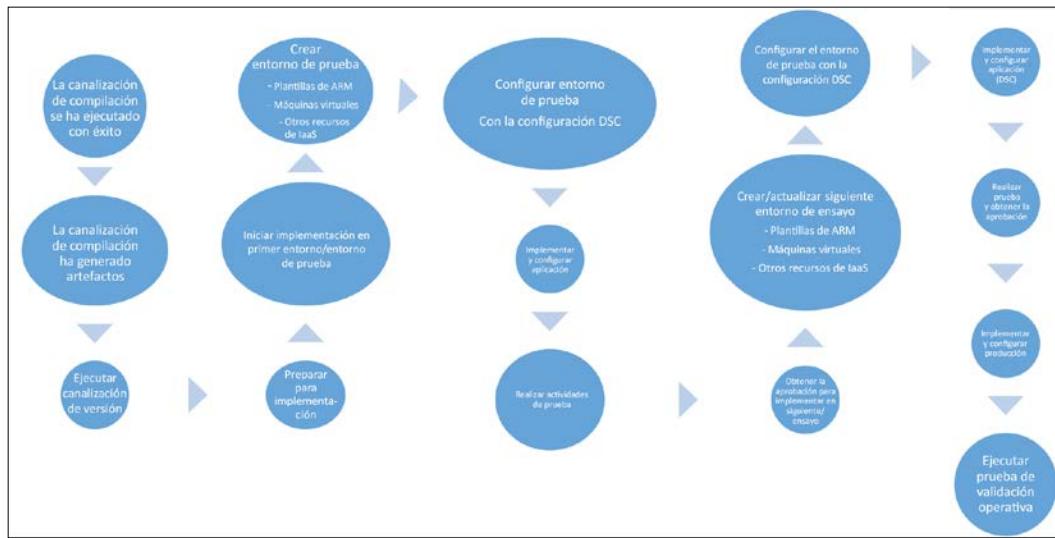
A continuación, se muestra una canalización de compilación para la solución basada en máquinas virtuales de IaaS. Una canalización de versión se inicia cuando un desarrollador inserta su código en el repositorio. La canalización de compilación se inicia automáticamente como parte de la integración continua. Compila y genera el código, ejecuta pruebas unitarias en él, verifica la calidad del código y genera documentación a partir de los comentarios del código. Implementa los nuevos binarios en el entorno dev (tenga en cuenta que el entorno de desarrollo no se ha creado recientemente), cambia la configuración, ejecuta pruebas de integración y genera etiquetas de compilación para una fácil identificación. A continuación, coloca los artefactos generados en una ubicación a la que puede acceder la canalización de versión. En caso de que haya problemas durante la ejecución de cualquier paso en esta canalización, se comunica lo mismo al desarrollador como parte de los comentarios de la canalización de compilación de manera que puedan volver a trabajar y enviar sus cambios.

La canalización de compilación debería fallar o pasar en función de la gravedad de los problemas encontrados y que depende de la organización:



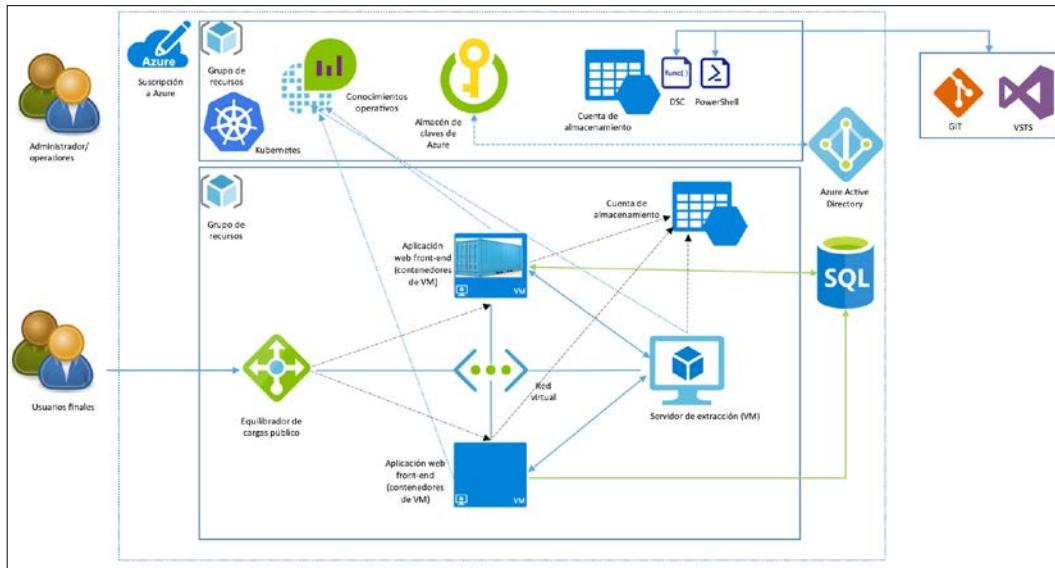
Canalización de versión

A continuación, se muestra una canalización de versión para la implementación basada en máquinas virtuales de IaaS. Una canalización de versión comienza después de la finalización de la canalización de compilación. El primer paso para la canalización de compilación es reunir los artefactos generados por la canalización de compilación. Generalmente son ensamblados desplegables, binarios y documentos de configuración. La canalización de versión ejecuta y crea o actualiza el primer entorno, que generalmente es un entorno de prueba. Utiliza las plantillas de Azure Resource Manager para aprovisionar todos los servicios y recursos de IaaS y PaaS en Azure y también los configura. También ayudan a ejecutar scripts y la configuración de DSC después de que las máquinas virtuales se crean como pasos posteriores a la creación. Esto ayuda a configurar el entorno dentro de la máquina virtual y el sistema operativo. En esta etapa, se implementan y configuran los binarios de la aplicación desde la compilación. Se realizan diferentes pruebas automatizadas para verificar el funcionamiento de la solución y, si resulta satisfactoria, la canalización se desplaza al siguiente entorno después de obtener las aprobaciones necesarias. Los mismos pasos se ejecutan de nuevo en el entorno siguiente, incluido el entorno de producción. Por último, las pruebas de validación operativa se ejecutan en producción para garantizar que la aplicación funciona como se esperaba y que no hay desviaciones. En esta etapa, si hay algún problema o error, se deben corregir y se debe repetir todo el ciclo; sin embargo, si no ocurre dentro de un período de tiempo estipulado, la instantánea conocida anterior debe restaurarse en el entorno de producción para minimizar el tiempo de inactividad:



DevOps para soluciones basadas en contenedor (IaaS)

La arquitectura típica para la solución basada en contenedor de IaaS se muestra a continuación:



Contenedores

Los contenedores son una solución perteneciente a la tecnología de la virtualización; sin embargo, estos no virtualizan un servidor físico. Por el contrario, un contenedor es una virtualización en el nivel de sistema operativo. Esto quiere decir que los contenedores comparten el kernel del sistema operativo proporcionado por el host entre sí junto con el host. Ejecutar varios contenedores en un host (físico o virtual) que comparten el kernel del sistema operativo del host. Existe un solo kernel de sistema operativo proporcionado por el host y utilizado por todos los contenedores que se ejecutan sobre él.

Los contenedores también se encuentran totalmente aislados del host y otros contenedores, como una máquina virtual. Los contenedores utilizan controladores de filtro de almacenamiento de Windows y el aislamiento de sesiones para proporcionar aislamiento de servicios del sistema operativo como el sistema de archivos, el registro, los procesos y las redes. Cada contenedor obtiene su propia copia de recursos del sistema operativo.

Docker

Docker proporciona funciones de administración para los contenedores de Windows. Se compone de dos archivos ejecutables:

- Demónio de Docker
- Cliente de Docker

El demonio de Docker es el caballo de batalla para la administración de los contenedores. Se trata de un servicio de Windows encargado de administrar todas las actividades en el host relacionadas con los contenedores. El cliente de Docker interactúa con el demonio de Docker y es responsable de la captura de entradas, así como de su envío a través del demonio de Docker. El demonio de Docker proporciona el tiempo de ejecución, bibliotecas, controladores de gráficos y motores para crear, administrar y supervisar los contenedores y las imágenes en el servidor host. También ofrece capacidades para crear imágenes personalizadas que se utilizan en la creación y el envío de aplicaciones a múltiples entornos.

DockerFile

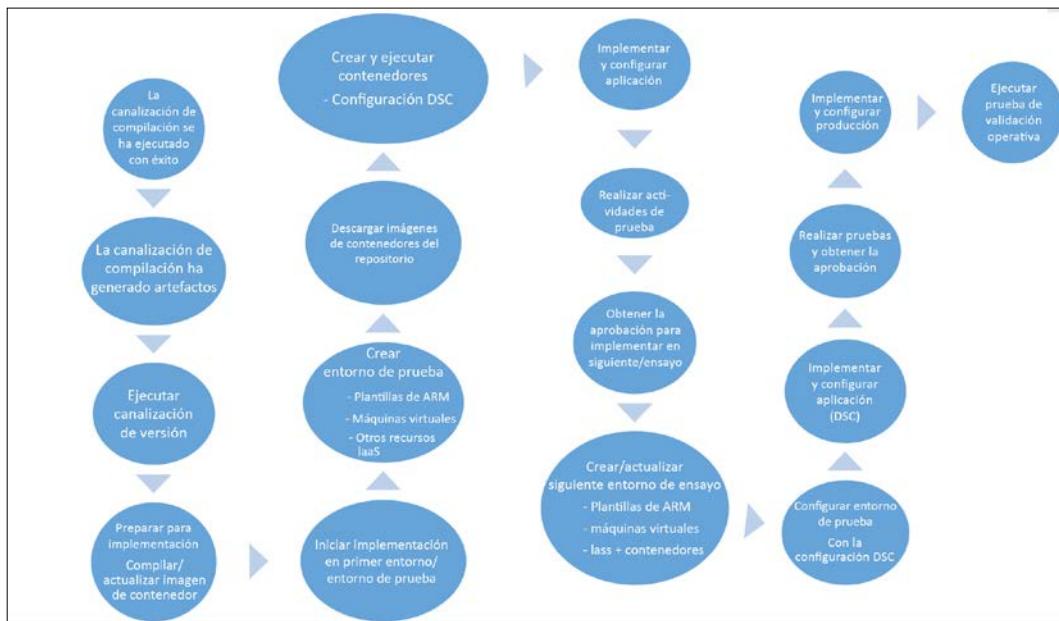
DockerFile es un módulo principal para crear imágenes de contenedor de Windows. Es un archivo legible por humanos basado en texto simple sin ninguna extensión y se llama DockerFile. Aunque existe un mecanismo para nombrarlo de manera diferente, generalmente se denomina DockerFile. DockerFile contiene instrucciones para crear una imagen de contenedor personalizado desde la imagen base. Estas instrucciones las ejecuta secuencialmente de arriba abajo el demonio de Docker, el motor detrás de todas las actividades relacionadas con los contenedores de Windows. Las instrucciones hacen referencia al comando y sus parámetros comprendidos por el demonio de Docker. DockerFile habilita la infraestructura como prácticas de código al convertir la implementación y la configuración de la aplicación en instrucciones que se pueden versionar y almacenar en un repositorio de código fuente.

Canalización de compilación

No hay diferencia desde la perspectiva de compilación entre el contenedor y una solución basada en máquina virtual. El paso de compilación no cambia. Consulte la sección anterior para obtener información sobre la canalización de compilación.

Canalización de versión

A continuación, se muestra una canalización de versión para la implementación basada en contenedores de IaaS. La única diferencia entre esta canalización y la canalización de versión es la administración de la imagen del contenedor y la creación de contenedores utilizando DockerFile y Docker Compose. Las utilidades avanzadas de administración de contenedores, como Docker Swarm, DC/OS y Kubernetes también se pueden implementar y configurar como parte de la administración de la versión. Sin embargo, se debe tener en cuenta que estas herramientas de administración de contenedores deben formar parte de la canalización de versión de servicios compartidos como se explicó anteriormente:



Azure Automation

Azure Automation es la plataforma de Microsoft para todas las implementaciones de automatización con respecto al cloud, en implementaciones on-premise e híbridas. Azure Automation es una plataforma de automatización madura que proporciona capacidades enriquecidas en los siguientes términos:

- Definir activos, como variables, conexiones, credenciales, certificados y módulos
- Implementar runbooks utilizando Python, scripts de PowerShell y flujos de trabajo de PowerShell
- Proporcionar interfaces de usuario para crear runbooks
- Administrar el ciclo de vida completo del runbook, incluyendo compilación, prueba y publicación
- Programar runbooks
- Capacidad para ejecutar runbooks en cualquier lugar: en el cloud u on-premise
- DSC como plataforma de administración de la configuración
- Administrar y configurar entornos, máquinas: Windows y Linux, aplicaciones e implementación
- Posibilidad de extender Azure Automation mediante la importación de módulos personalizados

Azure Automation proporciona un servidor de extracción DSC que ayuda a crear un servidor de administración de configuración centralizado que consiste en configuraciones para nodos/máquinas virtuales y sus componentes.

Implementa el patrón de tipo hub y spoke donde los nodos pueden conectarse al servidor de extracción DSC y descargar las configuraciones asignadas a ellos y reconfigurarse para reflejar su estado deseado. Los agentes DSC corregirán cualquier cambio y desviación dentro de estos nodos la próxima vez que se ejecutan. Esto garantiza que los administradores no tengan que seguir supervisando activamente el entorno para encontrar cualquier desviación.

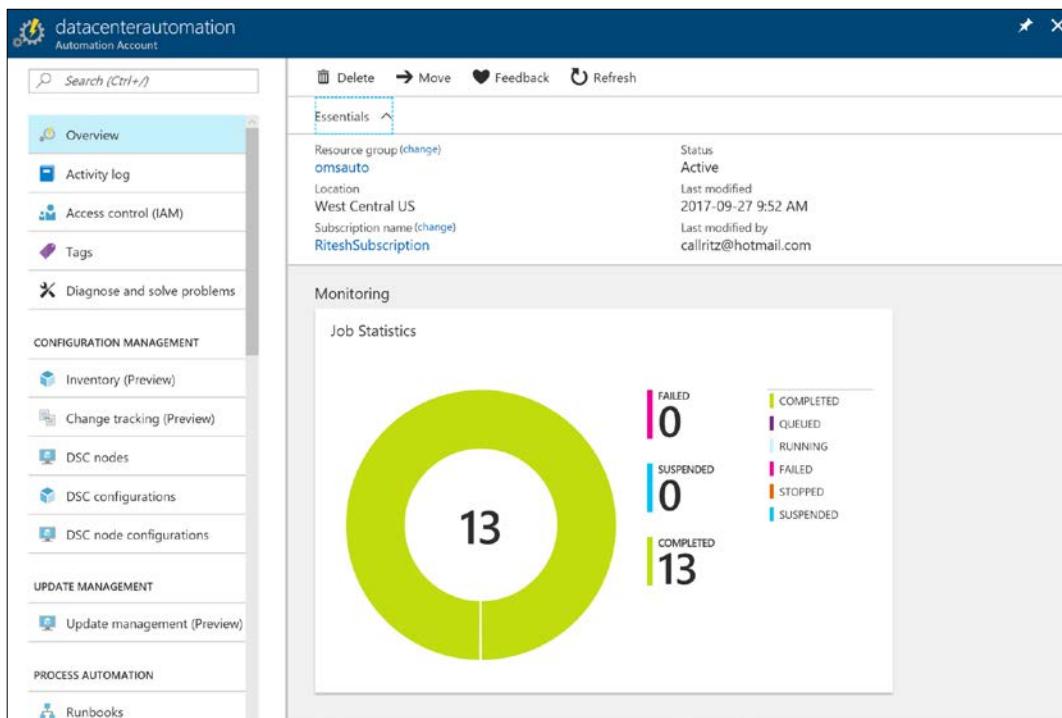
DSC proporciona un lenguaje declarativo en el cual se definen la intención y la configuración, pero no cómo ejecutar y aplicar esas configuraciones. Estas configuraciones se basan en el lenguaje de PowerShell y facilitan el proceso de administración de la configuración.

En esta sección, analizaremos una implementación simple del uso de DSC de Azure Automation para configurar una máquina virtual para instalar y configurar el servidor web (IIS) y crear un archivo `index.htm` que informe a los usuarios que el sitio web está en mantenimiento.

Aprovisionamiento de una cuenta de Azure Automation

Cree una nueva cuenta de Azure Automation desde el portal de Azure o PowerShell en un grupo de recursos nuevo o existente. Los lectores inteligentes encontrarán en la siguiente imagen que Azure Automation proporciona elementos de menú para DSC. Esta solución ofrece las siguientes prestaciones:

- **Nodos de DSC:** enumera todas las máquinas virtuales y contenedores que están alistados con el servidor de extracción DSC de Azure Automation actual. Estas máquinas virtuales y contenedores se administran utilizando configuraciones del servidor de extracción DSC actual.
- **Configuraciones de DSC:** enumera todas las configuraciones sin procesar de PowerShell importadas y cargadas en el servidor de extracción DSC. Tienen un formato legible por humanos y no tienen un estado compilado.
- **Configuraciones de nodos DSC:** enumera todas las compilaciones de configuraciones DSC disponibles en el servidor de extracción que se asignarán a los nodos: máquinas virtuales y contenedores. Una configuración DSC produce archivos MOF después de las compilaciones y, finalmente, se utilizan para configurar los nodos.



Creación de configuración DSC

El siguiente paso es escribir una configuración DSC utilizando cualquier editor de PowerShell para reflejar la intención de la configuración. Para este ejemplo, se crea una configuración `ConfigureSiteOnIIS` individual. Importa el módulo DSC base `PSDesiredStateConfiguration` que consta de recursos utilizados en la configuración. También declara un nodo `webserver`. Después de cargar y compilar esta configuración, generará un `DSCConfigurationNodes` denominado `ConfigureSiteOnIISwebserver`. Esta configuración puede aplicarse a los nodos.

La configuración consta de unos pocos recursos. Estos recursos configuran el nodo de destino. Los recursos instalan un servidor web, ASP.NET y framework y crean un archivo `index.htm` dentro del directorio `inetpub\wwwroot` con contenido para mostrar que el sitio está en mantenimiento. Para obtener más información sobre la escritura de la configuración DSC, consulte <https://docs.microsoft.com/PowerShell/dsc/configurations>.

```
Configuration ConfigureSiteOnIIS {
    Import-DscResource -ModuleName 'PSDesiredStateConfiguration'
    Node WebServer {
        WindowsFeature IIS
        {
            Name = "Web-Server"
            Ensure = "Present"
        }
        WindowsFeature AspDotNet
        {
            Name = "net-framework-45-Core"
            Ensure = "Present"
            DependsOn = "[WindowsFeature]IIS"
        }
        WindowsFeature AspNet45
        {
            Ensure = "Present"
            Name = "Web-Asp-Net45"
            DependsOn = "[WindowsFeature]AspNetDotNet"
        }
        File IndexFile
        {
            DestinationPath = "C:\inetpub\wwwroot\index.htm"
            Ensure = "Present"
            Type = "File"
            Force = $true
        }
    }
}
```

```
Contents = "<HTML><HEAD><Title> Website under
construction.</Title></HEAD><BODY>
<h1>If you are seeing this page, it means the website is
under maintenance and DSC Rocks !!!!!</h1></BODY></HTML>"
```

Importación de la configuración DSC

La configuración DSC aún es desconocida para Azure Automation. Está disponible en algunos equipos locales. Debe cargarse en las configuraciones DSC de Azure Automation. Azure Automation proporciona el cmdlet `Import-AzureRMAutomationDscConfiguration` para importar la configuración a Azure Automation.

```
Import-AzureRmAutomationDscConfiguration -SourcePath "C:\DSC\
AA\DSConfigfiles\ConfigSiteOnIIS.ps1" -ResourceGroupName "omsauto"
-AutomationAccountName "datacenterautomation" -Published -Verbose
```

La configuración DSC en Azure después de aplicar la configuración al nodo debe verse como se muestra a continuación:

NAME	AUTHORING STATUS	LAST MODIFIED
ConfigureSiteOnIIS	Published	2017-09-30 1:40 AM

Compilación de la configuración DSC

Una vez que la configuración DSC está disponible en Azure Automation, se puede solicitar que se compile. Azure Automation proporciona otro cmdlet para lo mismo. Use el cmdlet `Start-AzureRmAutomationDscCompilationJob` para compilar la configuración importada. El nombre de la configuración debe coincidir exactamente con el nombre de la configuración cargada. La compilación crea un archivo MOF con el nombre de la configuración y el nombre del nodo, que en este caso es `ConfigureSiteOnIIS.webserver`.

```
Start-AzureRmAutomationDscCompilationJob -ConfigurationName  
ConfigureSiteOnIIS -ResourceGroupName "omsauto" -AutomationAccountName  
"datacenterautomation" -Verbose
```

La configuración de nodos DSC en Azure después de aplicar la configuración al nodo debe verse como se muestra a continuación:

NAME	CREATED	LAST MODIFIED
ConfigureSiteOnIIS.WebServer	2017-09-30 1:42 AM	2017-09-30 1:42 AM

Asignación de la configuración a los nodos

Ahora, las configuraciones DSC compiladas se pueden aplicar a los nodos. Utilice `Register-AzureRmAutomationDscNode` para asignar la configuración a un nodo. El parámetro `NodeConfigurationName` identifica el nombre de configuración que debe aplicarse al nodo. Este es un cmdlet potente que también puede configurar el agente DSC que es `localconfigurationmanager` en nodos antes de que puedan descargar configuraciones y aplicar lo mismo.

Hay varios parámetros `localconfigurationmanager` que se pueden configurar y detalles sobre los mismos disponibles en <https://docs.microsoft.com/PowerShell/dsc/metaconfig>.

```
Register-AzureRmAutomationDscNode -ResourceGroupName "omsauto"
-AutomationAccountName "datacenterautomation" -AzureVMName
testtwo -ConfigurationMode ApplyAndAutocorrect -ActionAfterReboot
ContinueConfiguration -AllowModuleOverwrite $true -AzureVMResourceGroup
testone -AzureVMLocation "West Central US" -NodeConfigurationName
"ConfigureSiteOnIIS.WebServer" -Verbose
```

Los nodos DSC en Azure después de aplicar la configuración al nodo deben verse como se muestra a continuación:

NAME	STATUS	NODE CONFIGURATION	LAST SEEN	VM DSC EXTENSION VERSION
testtwo	Compliant	ConfigureSiteOnIIS.WebS...	2017-09-30 2:34 AM	Yes

Navegar por el servidor

Si corresponde, los grupos de seguridad de la red y los firewalls se abren y habilitan para el puerto 80 y se asigna una IP pública a la máquina virtual, se puede navegar por el sitio web predeterminado usando la dirección IP. De lo contrario, inicie sesión en la máquina virtual que se utiliza para aplicar la configuración DSC y navegue hasta `http://localhost`.

Debería aparecer una página como se muestra a continuación:



Este es el poder de la administración de la configuración que, sin escribir ningún código significativo, la creación de una configuración una vez puede aplicarse varias veces a los mismos y múltiples servidores y tener la seguridad de que se ejecutarán en el estado deseado sin ninguna intervención manual.

Azure para DevOps

Como se mencionó anteriormente, Azure es una plataforma enriquecida y madura que ofrece lo siguiente:

- Varias opciones de idiomas
- Varias opciones de sistemas operativos
- Varias opciones de herramientas y utilidades
- Varios patrones para implementar soluciones (máquinas virtuales, servicios de aplicaciones, contenedores, microservicios)

Con tantas opciones y elecciones, Azure es:

- **Cloud abierto:** está abierto para productos, herramientas y servicios de código abierto, de Microsoft y de terceros
- **Cloud flexible:** brinda flexibilidad tanto a los usuarios como a los desarrolladores para que se sientan cómodos al utilizar sus habilidades y conocimientos en los que se basa
- **Administración unificada:** proporciona funciones de supervisión y administración sin problemas

Todas las características mencionadas anteriormente son importantes para la implementación exitosa de DevOps para cualquier proyecto o interacción. La siguiente imagen muestra las herramientas y utilidades de código abierto que se pueden usar para diferentes fases en la gestión del ciclo de vida de la aplicación y en las DevOps generales. Esta es solo una pequeña representación de todas las herramientas y utilidades y hay muchas más opciones disponibles.

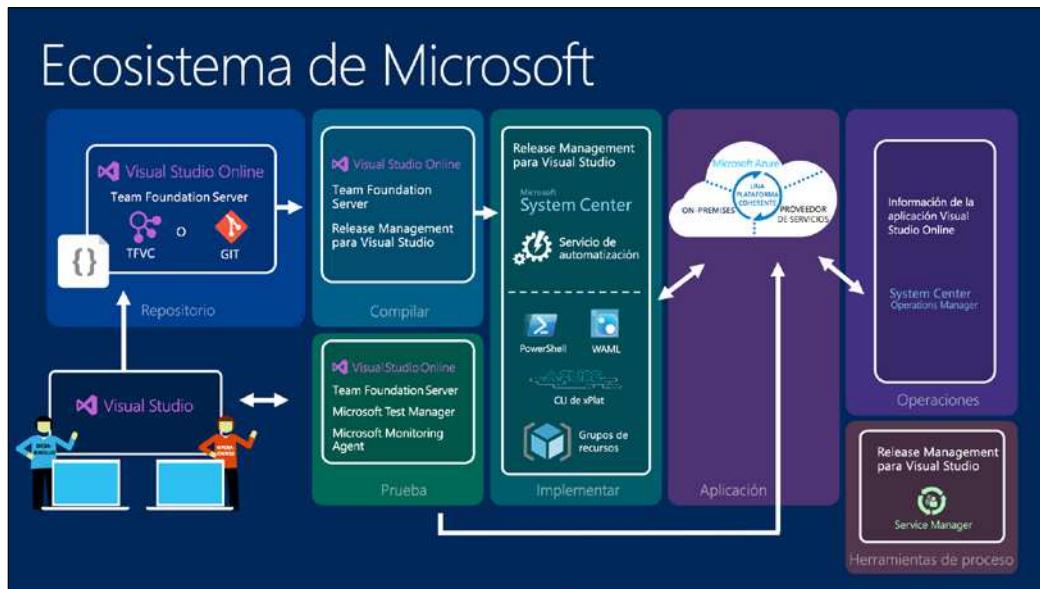
- Herramientas Jenkins, Hudson, Grunt y Gradle para generar canalización de compilación
- Selenium para la realización de pruebas
- Chef, Puppet, Jenkins, Hudson, Grunt, Gradle y más para la implementación o administración de configuración
- Nagios para alertas y supervisión
- Jira y Redmine para procesos de administración



La siguiente imagen muestra las herramientas y utilidades de Microsoft que se pueden usar para diferentes fases en la gestión del ciclo de vida de la aplicación y en las DevOps generales. De nuevo, esta es solo una pequeña representación de todas las herramientas y utilidades y hay muchas más opciones disponibles.

- Orquestación de compilación VSTS para construir una canalización de compilación
- Microsoft Test Manager, Pester para la realización de pruebas
- Desired State Configuration, PowerShell, plantillas de ARM, etc. para la implementación o la administración de configuración

- Log Analytics, Application Insights y System Center Operations Manager (SCOM) para alertas y supervisión
- VSTS y System Center Service Manager para procesos de administración



Resumen

DevOps está ganando mucha tracción e impulso en la industria. La mayoría de las organizaciones se ha dado cuenta de sus beneficios y espera implementar DevOps. Esto está sucediendo mientras la mayoría se están trasladando al cloud. Azure como modelo de cloud está enriquecido y es maduro en la prestación de servicios de DevOps, lo que facilita que dichas organizaciones implementen DevOps. En este capítulo, DevOps se ha discutido en detalle junto con sus prácticas básicas, como la administración de la configuración, la integración continua, la entrega continua y la implementación. También hemos hablado sobre diferentes máquinas virtuales basadas en PaaS, IaaS y una solución de cloud basada en IaaS de contenedores, junto con sus respectivos recursos de Azure, compilación y canalizaciones de versión. La administración de la configuración fue la parte central de este capítulo y discutimos los servicios DSC de Azure Automation y el uso de servidores de extracción para configurar máquinas virtuales automáticamente. Finalmente, se discutió la apertura y flexibilidad de Azure con respecto a la variedad de opciones de idiomas, herramientas y sistema operativo de código abierto.

El coste o es uno de los controladores más importantes para pasar al cloud y existen oportunidades para ahorrar y administrar los costes incluso en el cloud. El siguiente capítulo sobre administración de costes proporciona detalles sobre la administración de costes y la minimización en Azure.

11

Cost Management

La principal razón por la que las empresas se están trasladando al cloud se debe al ahorro de costes. Una suscripción a Azure no tiene ningún coste inicial. Azure ofrece un modelo de **pago por uso**, lo que significa que utiliza el consumo como base para medir el uso y proporciona facturas mensuales. No hay ningún límite máximo de consumo en Azure. Sus recursos son ilimitados, por lo que cualquier persona que tenga acceso a Azure puede crear todos los recursos que desee. En estas circunstancias, es importante que las empresas tengan un control estricto de su consumo y su uso en Azure. Aunque pueden crear políticas para establecer convenciones y estándares en la organización, también es necesario que puedan obtener fácilmente información sobre la facturación y el consumo en Azure. Por otra parte, quieren emplear prácticas recomendadas para consumir recursos de Azure con el fin de aumentar al máximo los beneficios. Para ello, los arquitectos necesitan conocer los recursos y las características de Azure, sus costes correspondientes y una comparación de los costes y beneficios entre la demanda de soluciones y las características.

En este capítulo, explicaremos lo siguiente:

- Facturación de Azure
- Facturación
- Uso y cuotas
- API de uso y facturación
- Calculadora de precios de Azure
- Prácticas recomendadas

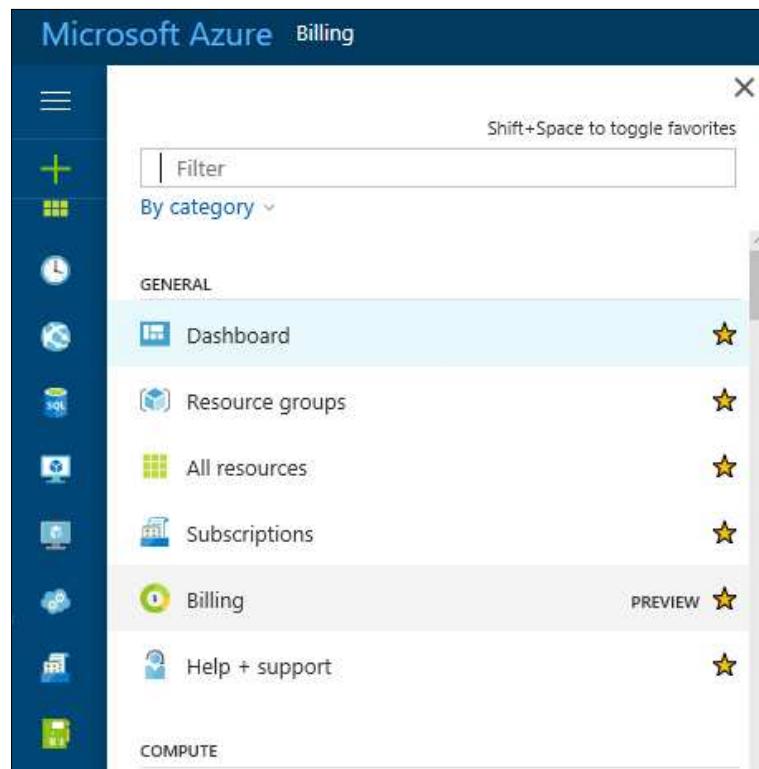
Descripción de la facturación

Azure es una utilidad de servicios que tiene las siguientes ventajas:

- Sin costes iniciales
- Sin tarifas de cancelación
- Facturación basada en el uso por minuto
- Pago basado en el consumo continuado

En estas circunstancias, resulta muy difícil calcular el coste inicial del consumo de recursos de Azure. Cada recurso de Azure tiene su propio modelo de costes y se cobra en función del almacenamiento, el uso y el intervalo de tiempo. Es muy importante para los departamentos de gestión, administración y finanzas controlar el uso y los costes. Azure proporciona los informes de uso y facturación necesarios para que los gestores y administradores de las organizaciones puedan generar un informe de uso y costes basado en diversos criterios.

El portal de Azure ofrece información detallada sobre la facturación y el uso a través de la característica de facturación, a la que se puede obtener acceso desde la hoja de navegación principal.



Dispone de un submenú para generar informes sobre los costes y la facturación:

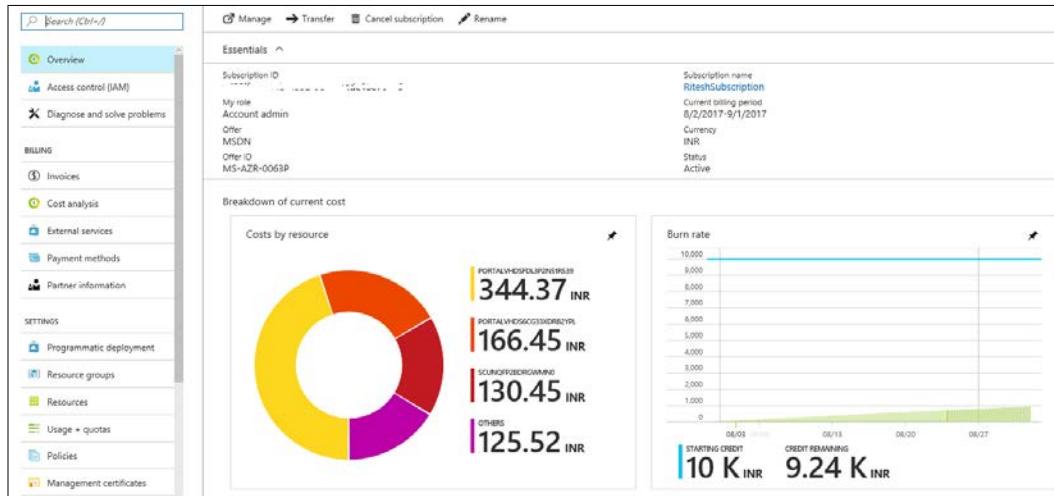
The screenshot shows the Microsoft Azure Billing interface. On the left, there's a sidebar with navigation links: Overview, Diagnose and solve problems, Subscriptions, Invoices, Contact info, Billing address, Payment methods, New support request, and Support + TROUBLESHOOTING. The 'Subscriptions' link is highlighted. The main content area has a header with 'New subscription' and 'Manage' buttons. Below this is a section titled 'Essentials' with account details: Account admin (callritz@hotmail.com), Next bill (9/1/2017), Next charge (9/11/2017), Currency (INR), Billing country/region (IN), and Total billed (last 12 mo) (0.00 INR). A 'Recent billing history' section follows, and at the bottom right, it says 'Your bill amount was 0 INR.'

Al hacer clic en el menú **Suscripciones** en esta hoja, aparece una lista de todas las suscripciones a las que tiene acceso el usuario para generar informes:

The screenshot shows the 'Billing - Subscriptions' page. The sidebar is identical to the previous one, with 'Subscriptions' highlighted. The main content area has a header with 'New subscription' and 'Manage' buttons. Below this is a section titled 'My subscriptions' which lists a single entry: 'RiteshSubscription' with 'SUBSCRIPTION ID' (not visible), 'OFFER' (MSDN), and 'STATUS' (Active). Below this is a section titled 'Other subscriptions' which states 'You don't have any subscriptions with billing read access'.

Cost Management

Al hacer clic en el nombre de la suscripción en esta lista, se abre un panel en el que pueden encontrarse los datos completos sobre facturación, facturas, uso, consumo, políticas, grupos de recursos y recursos. La tabla de esta sección de información general incluye el porcentaje de coste por recurso y también la tasa de uso:



Al hacer clic en la tabla, aparecen los detalles del coste basado en cada recurso. Aquí están las diversas cuentas de almacenamiento aprovisionadas en Azure, con el coste de cada una. Con esta pantalla, se pueden generar diversos tipos de informes utilizando diferentes criterios y diferentes combinaciones de estos criterios para lo siguiente:

- Tipos de recursos
- Grupo de recursos
- Intervalo de tiempo
- Etiqueta

Las etiquetas son una característica especialmente interesante. Se pueden utilizar consultas basadas en etiquetas, como Departamento, Proyecto, Propietario, Centro de costes o cualquier otro par de nombre-valor, para consultar el coste correspondiente. También se puede descargar el informe de costes como un archivo CSV utilizando el botón Descargar:

NAME	TYPE	RESOURCE GROUP	COST (INR)	TAGS
portalvhddfd3lpn51519	Storage account (classic)	Default-Storage-WestUS	344.37	---
portalvhdsfegj33xdrh2ypl	Storage account (classic)	Default-Storage-EastUS	166.45	---
scomqfp2ldrgwmr0	Storage account	abra	130.45	---
scomqfp2ldrgwtx	Storage account	abra	64.74	---
portalvhdsn0btwdlj38ym	Storage account (classic)	Default-Storage-EastAsia	60.79	---

Al hacer clic sobre cada recurso, se observa su consumo y coste diarios:

DATE	AMOUNT (INR)
8/2/2017	13.73
8/3/2017	13.73
8/4/2017	13.73
8/5/2017	13.73
8/6/2017	13.73
8/7/2017	13.73
8/8/2017	13.73
8/9/2017	13.73
8/10/2017	13.73
8/11/2017	13.73
8/12/2017	13.73
8/13/2017	13.73
8/14/2017	13.73
8/15/2017	13.73
8/16/2017	13.73

Facturación

La característica de facturación de Azure también proporciona información sobre las facturas que se generan cada mes.

Al hacer clic en el menú **Facturas**, se abre una lista de todas las facturas generadas:

The screenshot shows the Azure Billing Invoices page. On the left, there's a sidebar with links like Overview, Access control (IAM), Diagnose and solve problems, and Invoices (which is selected). The main area has tabs for Older invoices, Email invoice, Access to invoice, and External services. A note says "External services are invoiced separately." Below that is a "Subscription" section with a dropdown menu. The main table has columns for BILLING PERIOD, CHARGE DATE, AMOUNT (INR), and INVOICE. The data shows four entries for June 2017, all with a charge date of 6/2/2017 and an amount of ₹ 0.00. The INVOICE column says "Not available" for all rows.

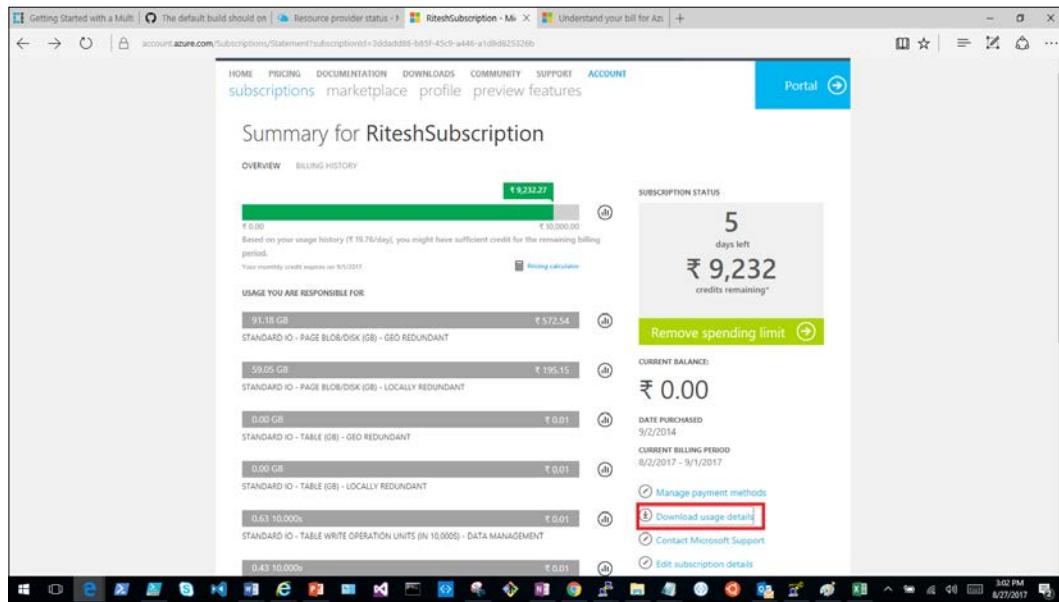
BILLING PERIOD	CHARGE DATE	AMOUNT (INR)	INVOICE
7/2/2017-8/1/2017	6/2/2017	₹ 0.00	Not available
6/2/2017-7/1/2017	7/2/2017	₹ 0.00	Not available
5/2/2017-6/1/2017	6/2/2017	₹ 0.00	Not available
4/2/2017-5/1/2017	5/2/2017	₹ 0.00	Not available

Al hacer clic en cualquiera de las facturas, se ofrecen los detalles de esa factura:

The screenshot shows the "Costs by service" details page for the month of June 2017. It includes a note about excluding non-Microsoft services. The main table has columns for NAME, RESOURCE ID, CONSUMED UNITS, BILLABLE UNITS, and PRE-TAX COST (INR). The data lists various Azure services and their usage statistics for the specified period.

NAME	RESOURCE ID	CONSUMED UNITS	BILLABLE UNITS	PRE-TAX COST (INR)
Standard IO - Page Blob/Disk (GB) - Geo Redundant	0e9d0c9b-ab5d-4312-9c7e-3794e22af9c4	107.966	107.966	0.00
Standard IO - Page Blob/Disk (GB) - Locally Redundant	d23a5753-#E5-4d9f-a028-8cc5cf0d3882	82.764	82.764	0.00
Standard IO - Table (GB) - Geo Redundant	92567832-#E3-47c4-9e0a-b2b9461a8d75	0.001	0.001	0.00
Standard IO - Table (GB) - Locally Redundant	3f2b1e1c-c888-4ec6-ad6f-dd0ef38819c9	0.004	0.004	0.00
Standard IO - Table Write Operation Units (in 10,000s) - Data Management	9cb07de8-bc0d-46bc-8423-a25fe06779e3	0.878	0.878	0.00
Standard IO - Table Write Operation Units (in 10,000s) - Geo Redundant	ad32ffsc8-9da5-4577-8683-56ae54d39e42	0.512	0.512	0.00

También existe una forma alternativa de descargar los detalles de la factura. Se pueden consultar iniciando sesión en <https://account.azure.com> y descargándolos:



Consumidores con un contrato EA (Enterprise Agreement)

Los consumidores empresariales que tengan un contrato EA (Enterprise Agreement) pueden usar <https://ea.azure.com> para descargar sus informes de uso y facturación. Asimismo, recientemente se ha publicado un nuevo paquete de contenido Power BI que se puede utilizar para consultar el uso y los costes de Azure a través de informes y un panel en Power BI.



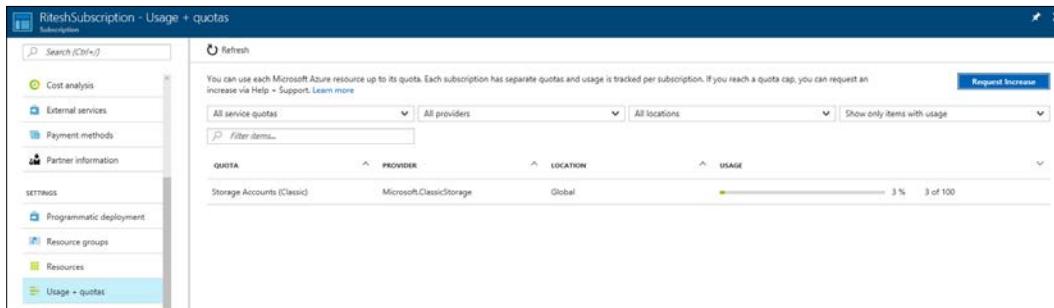
Hay disponible más información sobre este tema en <https://azure.microsoft.com/blog/new-power-bi-content-pack-for-azure-enterprise-users/>.

Uso y cuotas

Cada suscripción posee una cuota limitada para cada tipo de recurso. Por ejemplo, podría haber un máximo de 60 direcciones IP públicas aprovisionadas con una cuenta de Microsoft de MSDN. Del mismo modo, todos los recursos tienen un límite máximo predeterminado para cada tipo de recurso. Las cifras de los tipos de recursos de una suscripción se pueden aumentar poniéndose en contacto con el servicio de soporte técnico de Azure o haciendo clic en el botón **Solicitar aumento**.

Cost Management

La información sobre el uso y la cuota está disponible en el submenú **uso + cuotas** del menú **Suscripción**:



Esta hoja muestra todos los tipos de recursos aprovisionados en la suscripción junto con su ubicación y sus cifras. Aquí la cuota de las cuentas de almacenamiento clásico es 100 y, en la actualidad, se han consumido tres cuentas de almacenamiento clásico.

Hay disponibles criterios de filtrado en función de la ubicación, los proveedores, el uso y las cuotas. Se pueden generar informes personalizados basados en estos criterios de filtrado.

Proveedores de recursos

Los recursos se basan en tipos de recursos y están disponibles a través de proveedores de recursos. Hay disponibles numerosos proveedores en Azure que proporcionan los tipos de recursos necesarios que necesitan los usuarios para crear sus instancias. Por ejemplo, el proveedor de recursos **Microsoft.Compute** proporciona tipos de recursos de máquinas virtuales. Con estos tipos de recursos de máquinas virtuales, se pueden crear instancias de máquinas virtuales.

Es obligatorio que los proveedores de recursos se registren con suscripciones a Azure. Los tipos de recursos no están disponibles en una suscripción si los proveedores de recursos no están registrados. Para obtener una lista de los proveedores que están disponibles, los que están registrados y los que no, y para registrar a los proveedores no registrados o viceversa, se puede utilizar este panel:

PROVIDER	STATUS	Actions
Microsoft.AppService	Registered	Re-register Unregister
Microsoft.Automation	Registered	Re-register Unregister
Microsoft.Backup	Registered	Re-register Unregister
Microsoft.Batch	Unregistered	Register
Microsoft.Cache	Registered	Re-register Unregister
Microsoft.ClassicCompute	Registered	Re-register Unregister
Microsoft.ClassicNetwork	Registered	Re-register Unregister
Microsoft.ClassicStorage	Registered	Re-register Unregister
Microsoft.CognitiveServices	Registered	Re-register Unregister
Microsoft.Compute	Registered	Re-register Unregister
Microsoft.Devices	Registered	Re-register Unregister
Microsoft.DevTestLab	Registered	Re-register Unregister
Microsoft.DocumentDB	Registered	Re-register Unregister
microsoft.insights	Registered	Re-register Unregister
Microsoft.KeyVault	Registered	Re-register Unregister

API de uso y facturación

Aunque el portal es una forma muy práctica de consultar la información sobre el uso, la facturación y las facturas manualmente, Azure también proporciona lo siguiente:

- **API de descarga de facturas:** esta API sirve para descargar facturas
- **API de uso de recursos:** esta API de uso de recursos de Azure sirve para obtener una estimación del consumo en Azure
- **API de lista de tarifas:** la API de lista de tarifas de Azure sirve para obtener la lista de los recursos disponibles de Azure y la información sobre precios estimados de cada uno

Estas API pueden utilizarse para recuperar los datos mediante programación y crear informes y paneles personalizados. Estas API se pueden utilizar en cualquier lenguaje de programación o scripting para crear una solución de facturación completa.

Modelos de precios de Azure

Azul tiene varios modelos de precios. Prácticamente tiene un modelo para casi todos los tipos de consumidores: desde cuentas gratuitas para estudiantes y desarrolladores a modelos de pago por uso, además de contratos EA (Enterprise Agreement) y un modelo de partners de proveedores de soluciones de cloud. Además de estos tipos de cuentas, hay precios y descuentos adicionales, como instancias reservadas de VM y ventajas híbridas de Azure.

Ventajas híbridas de Azure

Cuando se aprovisiona una máquina virtual en Azure, hay dos tipos de costes: el coste de los recursos necesarios para ejecutar la máquina virtual y el coste de la licencia del sistema operativo. Aunque los consumidores con un contrato EA (Enterprise Agreement) obtienen algunos descuentos en los precios en comparación con otras cuentas, Azure tiene otra oferta para ellos. Es lo que se conoce como ventajas híbridas de Azure. Con este plan, los consumidores con un contrato EA (Enterprise Agreement) pueden traer las licencias on-premise de su sistema operativo para crear máquinas virtuales en Azure y Azure no les cobrará el coste de la licencia. El ahorro de costes puede ser llegar a ser hasta del 40 % con este plan. Los consumidores con un contrato EA (Enterprise Agreement) también deben tener Software Assurance (SA) para disfrutar de esta ventaja y su aplicación para Windows Standard y Datacenter Edition. Cada licencia de 2 procesadores o cada conjunto de licencias de 16 núcleos tiene derecho a dos instancias de hasta 8 núcleos o a una instancia de hasta 16 núcleos. La ventaja híbrida de Azure para licencias de Standard Edition solo puede utilizarse una vez, ya sea on-premises o en Azure. Data Center Edition permite el uso simultáneo tanto on-premises como en Azure.

Instancias de máquina virtual reservadas de Azure

Los clientes pueden reservar con antelación un número fijo de máquinas virtuales durante uno a tres años comprometiéndose a ese mismo número en Azure para el sistema operativo Windows y Linux. Azure proporciona hasta un 72 % de descuento en estas máquinas virtuales basado en el modelo de precios por uso. Aunque existe un compromiso inicial, no existe obligación de usarlas. Estas instancias reservadas se pueden cancelar en cualquier momento. Incluso se pueden asociar a ventajas híbridas de Azure que ofrecen una reducción adicional del coste de la licencia para estas máquinas virtuales.

Cuentas de pago por uso

Se trata de cuentas generales de Azure que se facturan mensualmente a los consumidores. Los consumidores no se comprometen a ningún uso y son libres de utilizar cualquier recurso en función de sus necesidades. Los costes de los recursos se calculan en función del uso de sus recursos y del tiempo de actividad. Esto, sin embargo, depende del tipo de recurso. Cada recurso tiene su propio modelo de costes. Estas cuentas tampoco tienen asociado ningún coste inicial. Por lo general, este plan no dispone de ningún descuento.

Contratos EA (Enterprise Agreement)

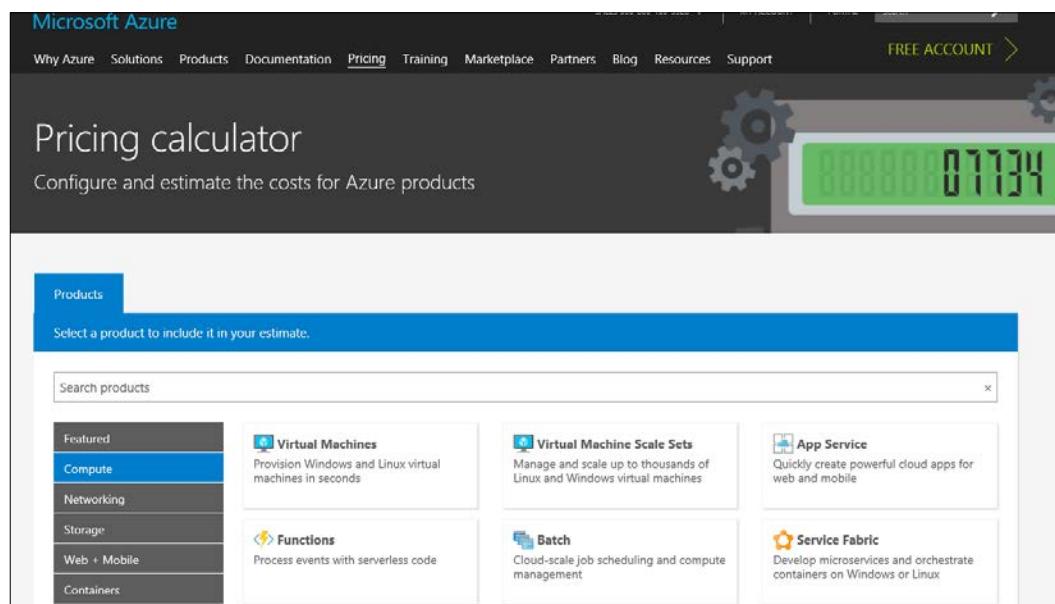
Los consumidores que ya tienen acuerdos con Microsoft pueden añadir su inquilino de Azure como parte de contratos EA (Enterprise Agreement). Los consumidores pueden disfrutar de grandes descuentos si poseen este tipo de contrato. Solo tienen que comprometerse inicialmente a una cantidad de dinero anual para ser incluidos en este plan. Pueden consumir los recursos acordados de la forma que mejor les convenga. Consulta <https://azure.microsoft.com/pricing/enterprise-agreement/> para obtener más información.

Proveedor de soluciones en el cloud

El proveedor de soluciones en el cloud (CSP) es un modelo para partners de Microsoft. CSP permite que los partners sean los propietarios absolutos de la relación y el ciclo de vida del consumidor para Microsoft Azure. Los partners pueden implementar sus soluciones en el cloud y cobrar a los consumidores utilizando este plan. Consulta <https://azure.microsoft.com/offers/ms-azr-0145p/> para obtener más información.

Calculadora de precios de Azure

Azure ofrece una calculadora de costes para que los usuarios y los consumidores calculen su coste y uso. Esta calculadora está disponible en <https://azure.microsoft.com/pricing/calculator/>:



Cost Management

Los usuarios pueden seleccionar varios recursos en el menú de la izquierda y estos se añadirán a la calculadora. En el ejemplo siguiente, se agrega una máquina virtual. Se proporciona una configuración adicional con respecto a la región de la máquina virtual, el sistema operativo, el nivel, el tamaño de instancia, el número de horas y el recuento para calcular su coste:

Your Estimate

Virtual Machines 1: D1: 1 cores, 3.5 GB RAM, 50 GB disk

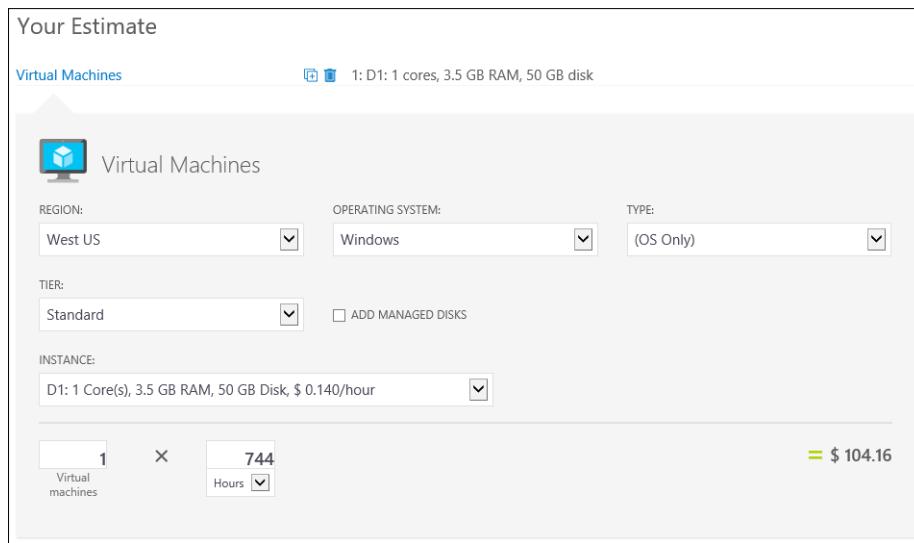
Virtual Machines

REGION: West US OPERATING SYSTEM: Windows TYPE: (OS Only)

TIER: Standard ADD MANAGED DISKS

INSTANCE: D1: 1 Core(s), 3.5 GB RAM, 50 GB Disk, \$ 0.140/hour

1 × 744 Hours = \$ 104.16



Del mismo modo, a continuación, se muestra el coste de las funciones de Azure con respecto al tamaño de la memoria de la máquina virtual, el tiempo de ejecución y la ejecución por segundos:

Functions 128 GB memory, 30 sec execution time, 1,000,000 exe...

Functions

REGION: West US

The first 400,000 GB/s of execution and 1,000,000 executions are free.

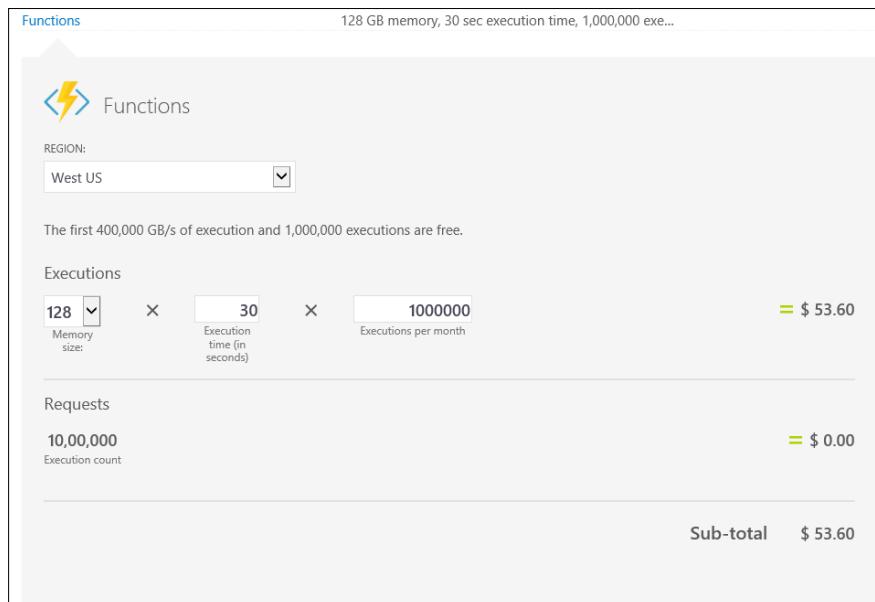
Executions

128 Memory size × 30 Execution time (in seconds) × 1000000 Executions per month = \$ 53.60

Requests

10,000,000 Execution count = \$ 0.00

Sub-total \$ 53.60



Azure ofrece diferentes niveles y planes de soporte, que son los siguientes:

- **Soporte predeterminado:** gratuito
- **Soporte para desarrolladores:** 29 dólares al mes
- **Soporte estándar:** 300 dólares al mes
- **Professional Direct:** 1000 dólares al mes

Por último, se muestra el coste total estimado :

The screenshot shows a web-based pricing calculator for Azure support. At the top, it says "Support ⓘ". Below that, under "SUPPORT:", there is a dropdown menu set to "Developer" which is highlighted with a blue background. To the right of the dropdown is the price "\$ 29.00". Under "Programs and Offers", it says "LICENSING PROGRAM:" followed by a dropdown menu set to "Microsoft Online Services Program (MOSP)". There is also a checkbox labeled "SHOW DEV/TEST PRICING" which is unchecked. In the bottom section, it says "Estimated monthly cost" followed by the price "\$ 186.76". To the right of this price is a dropdown menu set to "US Dollar (\$)". At the bottom left is a blue "Export" button.

Es importante que los arquitectos conozcan todas las características de Azure que se utilizan en la arquitectura general y en la solución. El éxito de la calculadora de Azure depende de qué recursos se seleccionen y su configuración. Cualquier falseamiento daría lugar a estimaciones sesgadas e incorrectas y no se correspondería con la facturación real.

Prácticas recomendadas

Los arquitectos tienen que realizar un esfuerzo adicional para entender su arquitectura y los componentes de Azure que se utilizan. Con la supervisión activa, las auditorías y el uso, deberían determinar la oferta de Microsoft en cuanto a SKU, tamaño y características. En esta sección, se detallan algunas de las prácticas recomendadas que deben adoptarse desde la perspectiva de optimización de costes.

Prácticas recomendadas de computación

La computación se refiere a servicios que ayudan en la ejecución de los servicios. Estas son algunas de las prácticas recomendadas relacionadas con la computación:

- Elige la mejor ubicación para tus servicios de computación, tales como máquinas virtuales. Elige un lugar donde todas las características y recursos de Azure estén disponibles juntos en la misma región. Esto evitara el tráfico de salida.
- Elige el tamaño óptimo de las máquinas virtuales. Una máquina virtual de mayor tamaño tiene unos costes más altos que otra de un tamaño más pequeño y es posible que no sea necesario tener una máquina virtual de un tamaño tan grande.
- Cambia el tamaño de las máquinas virtuales durante las épocas de más demanda y reduce su tamaño cuando la demanda disminuya. Azure publica nuevos tamaños de VM con frecuencia. Si hay un nuevo tamaño que sea más adecuado, se debe utilizar.
- Cierra los servicios de computación durante las horas de inactividad o cuando no son necesarios. Esto se aplica a entornos que no son de producción.
- Cancela la asignación de máquinas virtuales en lugar de cerrarlas. De esta forma, se liberarán todos los recursos y los medidores de consumo se detendrán.
- Utiliza laboratorios de desarrollo y prueba para realizar pruebas. Incluyen políticas y disponen de características parada e inicio automáticos.
- Con conjuntos de escalado de máquinas virtuales, comienza con un número reducido de máquinas virtuales y aumentalo cuando la demanda crezca.
- Elige el tamaño adecuado (pequeño, mediano o grande) para las puertas de enlace de las aplicaciones. Tienen el respaldo de máquinas virtuales y pueden ayudar a reducir los costes si tienen un tamaño óptimo. Además, elige la puerta de enlace de aplicaciones de nivel básico si no es necesario que haya un firewall de aplicaciones web.
- Elige los niveles correctos para las puertas de enlace VPN (VPN básica, estándar, de alto rendimiento o ultra rendimiento).
- Minimiza el tráfico de red entre regiones de Azure colocando los recursos en la misma región.
- Utiliza el equilibrador de carga con IP públicas para obtener acceso a múltiples máquinas virtuales en lugar de asignar una IP pública a cada máquina virtual.
- Supervisa las máquinas virtuales y busca sus métricas de rendimiento y uso. En función de ellas, determina si quieres mejorar la máquina virtual. También podría ser posible que hubiera que reducir las máquinas virtuales.

Prácticas recomendadas de almacenamiento

El almacenamiento también tiene un impacto importante en los costes generales:

- Elige el tipo de redundancia de **almacenamiento** adecuado (GRS, LRS, RA-GRS). GRS es más costoso que LRS.
- Archiva los datos de almacenamiento para liberar espacio o archiva el nivel de acceso. Mantén los datos en el nivel caliente al que se accede con más frecuencia.
- Quita los blobs que no son necesarios.
- Elimina explícitamente los discos del sistema operativo de la máquina virtual después de eliminar la máquina virtual si no son necesarios.
- Las cuentas de almacenamiento se miden en función de sus operaciones de dimensionamiento, escritura, lectura, lista y contenedor.
- Utiliza preferentemente discos estándar en lugar de discos premium. Utiliza discos premium solo si la empresa lo exige.
- Usa CDN y almacenamiento en caché de archivos estáticos en el lugar de obtenerlos del **almacenamiento** en cada ocasión.

Prácticas recomendadas de PaaS

Estas son algunas de las prácticas recomendadas cuando PaaS es el modelo de implementación preferido:

- Elige el nivel adecuado de SQL de Azure (básico, estándar, premium RS o premium) y los niveles de rendimiento adecuados.
- Elige adecuadamente entre bases de datos individuales y elásticas. Si hay una gran cantidad de bases de datos, lo más rentable es utilizar bases de datos elásticas en lugar de bases de datos individuales.
- Rediseña tus soluciones para que utilicen PaaS (sin servidor, microservicios en contenedores) en lugar de soluciones de IaaS. Estas soluciones PaaS no tienen costes de mantenimiento y su consumo se factura por minuto. Si no utilizas estos servicios, no hay ningún coste a pesar de que tu código y tus servicios están disponibles todo el día.
- Cada recurso posee optimizaciones de costes específicas y no es posible explicarlas todas en un solo capítulo. Se recomienda a los usuarios que lean la documentación relacionada con el coste de cada característica y uso.

Prácticas recomendadas generales

Estas son algunas de las prácticas recomendadas generales desde la perspectiva de los costes:

- El coste de los recursos es diferente en cada región. Prueba a utilizar una región que tenga un coste mínimo. Obviamente, esto depende de la justificación del negocio.
- Los contratos EA (Enterprise Agreement) ofrecen descuentos máximos. Si no tienes un EA, pruébalo para disfrutar de sus ventajas económicas.
- Si los costes de Azure se pueden pagar por adelantado, ofrece descuentos para todos los tipos de suscripciones.
- Elimina o retira los recursos que no se utilizan. Averigua qué recursos están infroutilizados y reduce su tamaño o SKU. Si no son necesarios, elimínalos.
- Utiliza el asesor de Azure y ten en cuenta muy en serio sus recomendaciones.

Resumen

La administración y gestión de costes es una actividad importante cuando se trabaja con el cloud. Esto se debe principalmente a que el gasto mensual podría ser muy bajo, pero también podría ser muy alto si no se le presta la debida atención. Los arquitectos deben diseñar su aplicación de manera que se minimicen los costes. Deben usar los recursos adecuados de Azure, el SKU, nivel y tamaño apropiados y deben saber cuándo iniciar, detener, escalar o reducir de forma vertical y horizontal, transferir datos y muchas otras cosas. Con una administración adecuada de los costes, los gastos reales se ajustarán a los gastos presupuestados.

El próximo y último capítulo de este libro trata sobre las funciones de supervisión y auditoría de Azure.

12

Supervisión y auditoría

“Si no puedes medirlo, no puedes mejorararlo”

- Lord Kelvin

Imagina que has diseñado una solución que se implementa en el cloud, pero también en un centro de datos on-premises, que consta de múltiples servicios, como los componentes de aplicaciones de IaaS y PaaS de varios niveles y de varias capas. Esta solución está basada en una arquitectura altamente optimizada en términos de disponibilidad, escalabilidad, seguridad y rendimiento, entre otras muchas cosas. Ahora está en producción. Ahora, imagina que no hay ninguna utilidad de supervisión integrada en la solución.

Aquí se plantean algunos problemas, que son los siguientes:

- Se desconoce la disponibilidad, la escalabilidad, la seguridad y la fiabilidad de la solución en cualquier punto del tiempo.
- El equipo de operaciones no tiene ni idea de si la solución funciona de la forma prevista o no hasta que los usuarios y los consumidores empiezan a llamarles.
- El rendimiento de la solución se degrada con el tiempo, pero el equipo no tiene las métricas de rendimiento actuales para compararlas con nada.
- Los equipos de operaciones y soluciones no tienen mucha información para diagnosticar y solucionar los problemas.
- No se pueden realizar comprobaciones de seguridad.
- Los equipos de operaciones y soluciones no tienen ni idea de qué cosas se pueden mejorar para el consumidor. Falta innovación.

Los problemas mencionados anteriormente son suficientes para empezar a pensar en incorporar la función de supervisión en las soluciones, no importa dónde y cómo se implemente y se diseñe.

En este capítulo, vamos a explicar el tema de la supervisión desde las siguientes perspectivas:

- Perspectiva de la arquitectura
- Recursos importantes de Azure para la supervisión
- Application Insights
- Log Analytics
- Alertas
- Automatización de alertas para ejecutar runbooks
- Integración de Power BI

Supervisión

La supervisión es un aspecto importante de la arquitectura que debe formar parte de cualquier solución, ya sea grande o pequeña, crítica o no, esté en el cloud o no. No debe evitarse de ninguna forma.

La supervisión se refiere al acto de realizar un seguimiento de las soluciones y obtener diversa información de telemetría, procesarlas, identificar las que necesitan alertas basadas en reglas y activarlas. Generalmente, en el entorno se implementa un agente que lo supervisa y envía la información de telemetría a un servidor centralizado donde se realiza el resto del proceso de generación de alertas y notificación a las partes interesadas.

La supervisión ayuda a tomar medidas y acciones tanto proactivas como reactivas en la solución. También es el primer paso hacia la capacidad de auditoría de la solución. Si no hubiera entradas de registro de supervisión, sería difícil auditar el sistema desde diversas perspectivas, como el de la seguridad, el rendimiento y la disponibilidad, entre muchos otros.

La supervisión ayuda a identificar problemas de escalabilidad, rendimiento y disponibilidad antes de que se produzcan. Gracias a la supervisión, las averías de hardware, una configuración incorrecta del software y los problemas actualización de parches se pueden conocer antes de que afecten a los usuarios. La degradación del rendimiento se puede corregir antes de que ocurra.

De forma reactiva, los registros ayudan a encontrar las áreas y los lugares que están causando problemas, ayudan a identificar los problemas y permiten solucionarlos de una forma más rápida y mejor.

Los equipos pueden identificar patrones de problemas utilizando la información de telemetría de la función de supervisión y ayudar a eliminarlos con soluciones y características nuevas e innovadoras.

Azure es un valioso entorno de cloud que ofrece múltiples características y recursos de supervisión para supervisar no solo la implementación en el cloud, sino también la implementación on-premises, junto con otros proveedores de cloud.

Supervisión de Azure

La primera pregunta que debe responderse es: ¿qué debemos supervisar? Esta pregunta es aún más importante para las soluciones que se implementan en el cloud, ya que el control es limitado.

Hay algunos componentes importantes que se deberían supervisar. Entre ellos se incluyen los siguientes:

- Aplicaciones personalizadas
- Recursos de Azure
- SO invitado (máquinas virtuales)
- SO de host (servidores físicos de Azure)
- Infraestructura de Azure

Hay diferentes registros y funciones de supervisión de Azure para los componentes mencionados.

Registros de actividad de Azure

Anteriormente conocidos como registros de auditoría y registros de funcionamiento, se trata de eventos del plano de control en la suscripción de Azure. Proporcionan información y datos de telemetría en el nivel de la suscripción en lugar del nivel del recurso individual. Realizan un seguimiento de la información sobre todos los cambios que se producen en el nivel de la suscripción, como la creación, eliminación y actualización de recursos con **Azure Resource Manager (ARM)**. Ayudan a conocer la identidad (entidad de servicio, usuarios, grupos) y realizar una acción (escribir, actualizar) en los recursos (por ejemplo, almacenamiento, máquinas virtuales, SQL) en un punto temporal específico. Proporcionan información sobre los recursos cuya configuración se modifica, pero no su funcionamiento y ejecución internos.

Registros de diagnóstico de Azure

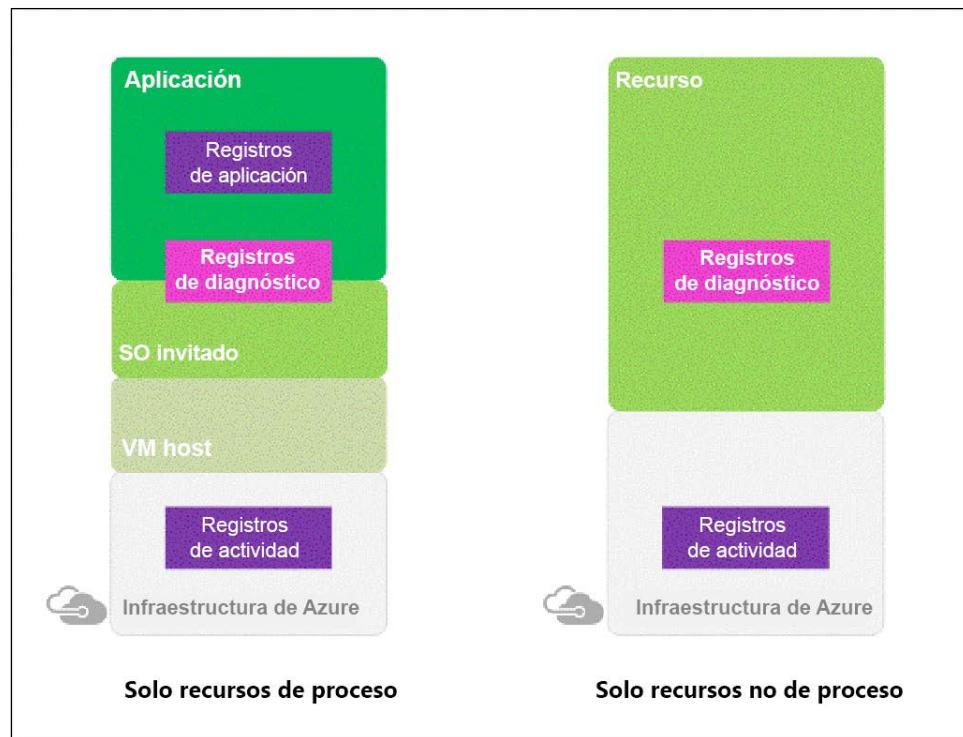
La información que se origina dentro del funcionamiento interno de los recursos de Azure que se obtiene se conoce como **registros de diagnóstico**. Estos registros proporcionan información de telemetría sobre las operaciones de los recursos que son inherentes al recurso. No todos los recursos proporcionan registros de diagnóstico y la persona que proporciona los registros de su propio contenido es completamente diferente en cada recurso. Los registros de diagnóstico se configuran individualmente para cada recurso. Un ejemplo de registro de diagnóstico es el almacenamiento de un archivo en un contenedor de un servicio de blob que se encuentra en una cuenta de almacenamiento.

Registros de aplicación de Azure

Los registros de aplicación pueden obtenerlos los recursos de Application Insights y se pueden administrar centralmente. Ayudan obtener más información sobre el funcionamiento interno de las aplicaciones personalizadas, como sus métricas de rendimiento y disponibilidad, entre otras, y obtener de ellas conocimientos para administrarlas mejor.

Registros de sistema operativo invitado y host

Los registros del sistema operativo invitado y host están a disposición de los usuarios que utilizan el recurso de supervisión de Azure. Proporcionan información sobre el funcionamiento, el estado del host y el sistema operativo invitado:



Los recursos importantes de Azure relacionados con la supervisión son los siguientes:

- Azure Monitor
- Azure Application Insights
- Log Analytics, anteriormente denominado **Operational Insights**

Hay otras herramientas como **System Center Operations Manager (SCOM)** que no forman parte de la característica del cloud, pero que se pueden implementar en las máquinas virtuales basadas en IaaS para supervisar cualquier carga de trabajo en Azure o en un centro de datos on-premises.

Azure Monitor

Azure Monitor es una herramienta y un recurso central que ofrece características de administración completas para configurar la función de supervisión en una suscripción de Azure. Proporciona características de administración de registros de actividad, registros de diagnóstico, métricas, Application Insights y Log Analytics. Se debe considerar un panel y un recurso de administración de todas las demás capacidades de supervisión.

Azure Application Insights

Azure Application Insights ayuda a ofrecer capacidades de supervisión, registros y métricas centralizadas en Azure a las aplicaciones personalizadas. Las aplicaciones personalizadas pueden empezar a enviar métricas, registros y otra información de telemetría a Azure Application Insights. También ofrece capacidades de informes, paneles y análisis completas para obtener conocimientos de los datos entrantes y realizar acciones en ellos.

Azure Log Analytics

Azure Log Analytics proporciona procesamiento centralizado de registros y generación de alertas e información a partir de ellos. Los registros de actividad, los registros de diagnóstico, los registros de aplicaciones, los registros de eventos e incluso los registros personalizados pueden enviar información a Log Analytics que, a su vez, puede proporcionar capacidades de informes, paneles y análisis completas para obtener conocimientos de los datos entrantes y realizar acciones en ellos.

Application Insights

Como su nombre sugiere, Azure Application Insights proporciona conocimientos dentro del funcionamiento de una aplicación. Las conocimientos que son importantes para una aplicación web incluyen el número de solicitudes entrantes por segundo, las solicitudes que fallan por segundo, el uso de hardware en lo que se refiere al uso de la CPU, la disponibilidad de la memoria y mucho más. Application Insights proporciona un panel, informes y gráficos para ver diversas métricas relacionadas con la aplicación. Esto ayuda a ver y comprender las tendencias en términos de uso de la aplicación, su disponibilidad, el número de solicitudes y mucho más para realizar acciones tanto preventivas como reactivas en la aplicación. La información de tendencias se puede utilizar para conocer qué cosas no funcionan bien para la aplicación y lo que funciona bien durante un período de tiempo.

El primer paso para trabajar con Application Insights consiste en aprovisionar este servicio en Azure en un grupo de recursos cuyos servicios generales y de administración consuman todas las aplicaciones. Si lo recuerdas, habíamos creado un grupo de recursos similar llamado Win2016devOps que alberga todos los servicios comunes, como Azure Key Vault, Application Insights, Operational insights y la cuenta de Azure Storage para contener los scripts y las plantillas que se utilizan en los entornos y las aplicaciones.

Aprovisionamiento

Como se mencionó anteriormente, el primer paso para consumir servicios de Application Insights es aprovisionarlos en Azure.

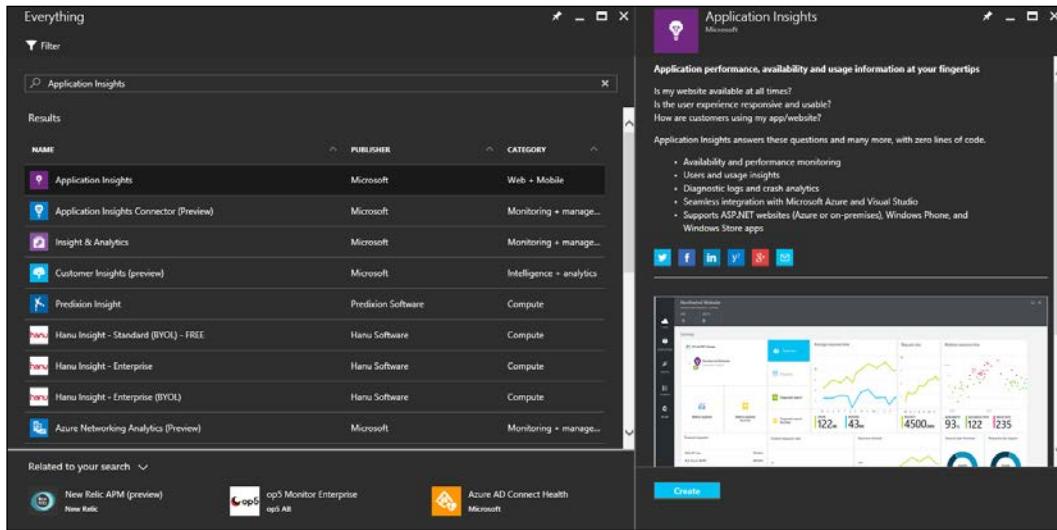
Application Insights se puede aprovisionar manualmente utilizando el portal de Azure, API de REST de Azure, PowerShell y plantillas de ARM:

1. Inicia sesión en el portal de Azure y suscríbete con las credenciales apropiadas. Luego, desplázate hasta un grupo de recursos existente o crea uno nuevo. Haz clic en el botón **Agregar**:

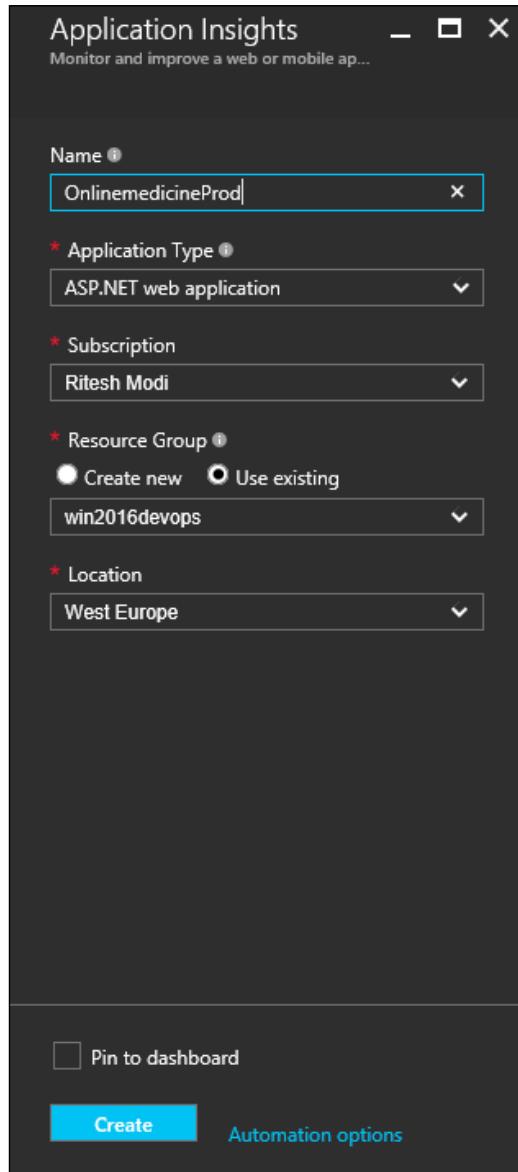
The screenshot shows the Azure portal interface for a resource group named 'win2016devops'. On the left, there's a sidebar with navigation links: Overview, Activity log, Access control (IAM), Tags, Quickstart, Resource costs, Deployments, Properties, Locks, and Automation script. The main area displays the 'Essentials' section with details about the subscription (Subscription name: Ritesh Modi, Subscription ID: [redacted], Location: West Europe). Below this is a table titled '4 items' showing four resources: NetworkMonitoring(OnlineMedicine...), OnlineMedicineOMS, OnlineMedicineVault, and win2016devops1. The 'Add' button, which is highlighted with a red box, is located at the top right of the main content area.

NAME	TYPE	LOCATION
NetworkMonitoring(OnlineMedicine...)	Solution	West Europe
OnlineMedicineOMS	Log Analytics	West Europe
OnlineMedicineVault	Key vault	West Europe
win2016devops1	Storage account	West Europe

2. Escribe Application Insights en el cuadro de búsqueda de la hoja resultante. El primer enlace debe referirse a Application Insights. Haz clic en él para crear una nueva instancia de servicio de Application Insights. Haz clic en el botón **Crear** para comenzar:



3. En la hoja resultante, se te pedirá que introduzcas el nombre de instancia del servicio de Application Insights, el tipo de aplicación, el nombre de la suscripción, el nombre del grupo de recursos y la ubicación del servicio. Proporciona la información apropiada y haz clic en el botón **Crear**. De esta forma, se aprovisionará el servicio:



4. Ahora, desplázate al servicio que muestra las propiedades esenciales, como su **clave de instrumentación**, resaltadas en la siguiente figura. La clave será diferente en cada instancia y, generalmente, se copia y se utiliza en Visual Studio. Ten en cuenta que parte de la información se ha ocultado por motivos de seguridad:

The screenshot shows the Microsoft Azure Application Insights Overview page for the 'OnlinemedicineProd' resource. The left sidebar contains navigation links for Overview, Activity log, Access control (IAM), Tags, INVESTIGATE (Application map, Smart Detection, Live Metrics Stream, Availability, Failures, Performance, Servers, Browser, Usage), and CONFIGURE. The main content area is titled 'Essentials' and displays the following information:

- Resource group: win2016devops
- Type: ASP.NET
- Instrumentation Key: bfc0dbe-c068-46f8-8443-... (partially obscured)
- Location: West Europe
- Subscription name: Ritesh Modi
- Subscription ID: [REDACTED]

Below this, there are four cards:

- Alerts: 0
- Live Stream: Click to configure
- Smart Detection: ONLINEMEDICINEPROD, 0 detections (last 24h)
- Web tests: 0

The 'Health' section features two charts:

- Overview timeline: SERVER RESPONSE TIME
- Page view load time: PAGE VIEW LOAD TIME

Both charts have a note: "Learn how to collect server response time data." and "Learn how to collect browser page load data."

Log Analytics

Application Insights se utiliza para supervisar la aplicación, sin embargo, es igualmente importante supervisar el entorno en el que se aloja y funciona la aplicación. Esto implica la infraestructura (por ejemplo, máquinas virtuales), un contenedor Docker y sus componentes relacionados.

Aprovisionamiento

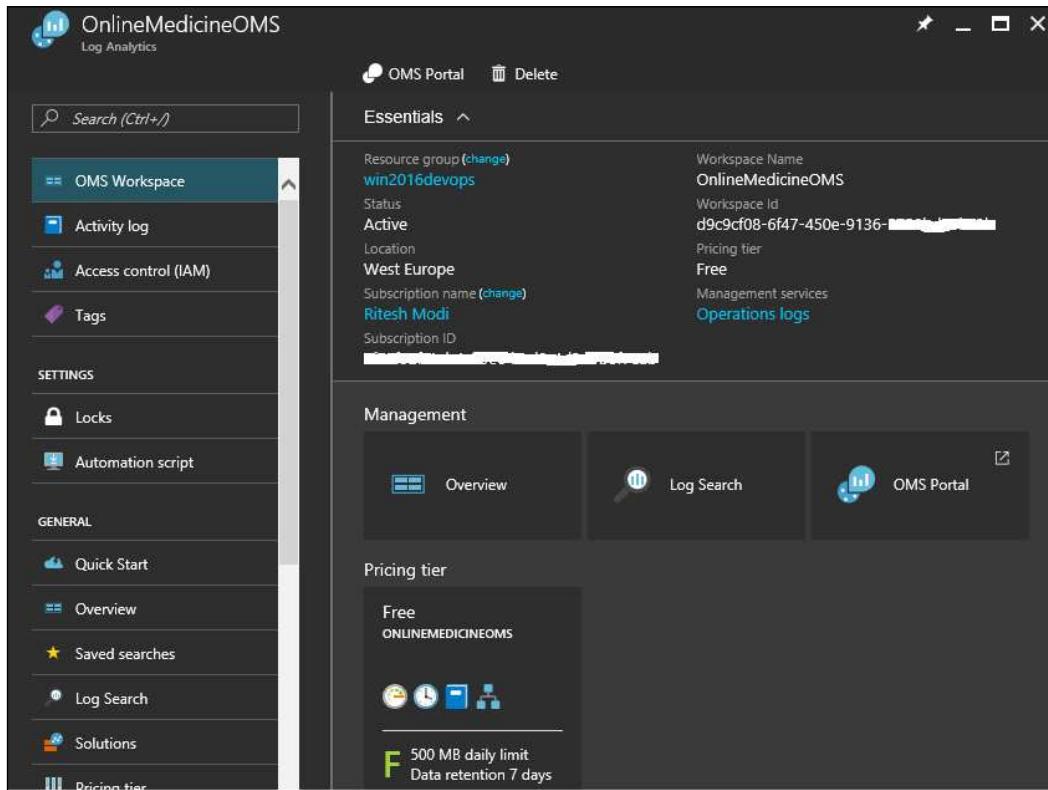
Log Analytics, que también se conoce como **Operational Management Suite (OMS)**, se debe aprovisionar en Azure para poder utilizarlo para supervisar las máquinas virtuales y los contenedores. Una vez más, de forma similar a Application Insights, Operational Insights se puede aprovisionar a través del portal de Azure, PowerShell, la API de REST y plantillas del administrador de grupos de recursos. Un espacio de trabajo de Operational Insights es un límite de seguridad cuyo acceso se permite a ciertos usuarios. Deben crearse varios espacios de trabajo para aislar a los usuarios y permitirles el correspondiente acceso a los datos de telemetría del entorno.

A continuación, se muestra el script JSON utilizado para el aprovisionamiento de un espacio de trabajo de Operational Insights:

```
{  
    "apiVersion": "2015-11-01-preview",  
    "type": "Microsoft.OperationalInsights/workspaces",  
    "name": "[parameters('workspaceName')]",  
    "location": "[parameters('deployLocation')]",  
    "properties": {  
        "sku": {  
            "Name": "[parameters('serviceTier')]"  
        }  
    }  
}
```

La información del nombre, la ubicación y sku es necesaria para aprovisionar un espacio de trabajo y sus valores se proporcionan mediante parámetros.

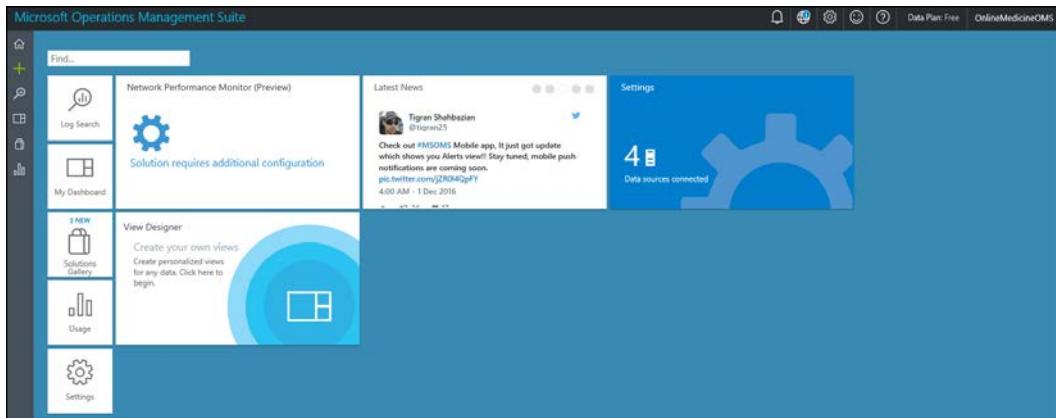
En la siguiente figura se muestra el espacio de trabajo después del aprovisionamiento:



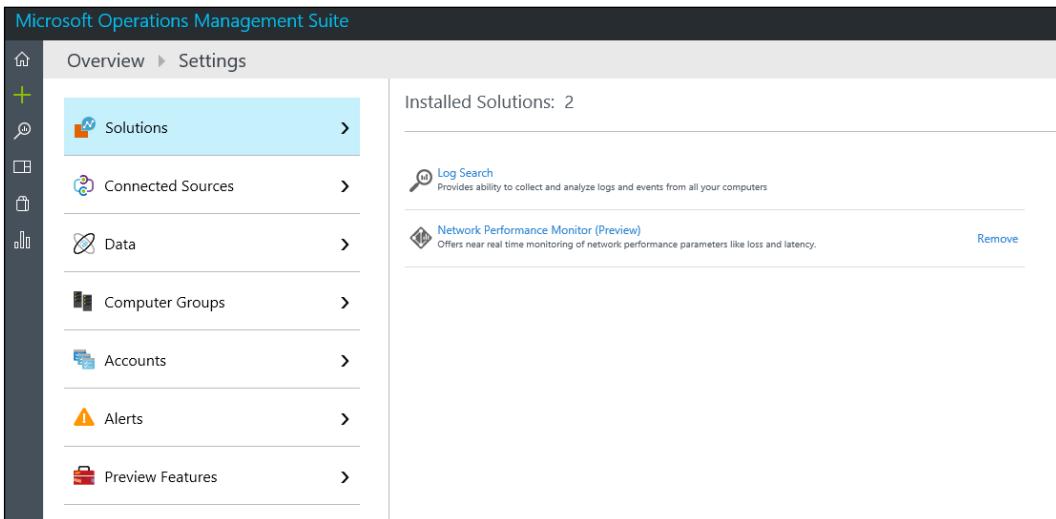
Haz clic en la sección **Portal de OMS** para abrir el portal del espacio de trabajo. Este portal se utiliza para ver toda la información de telemetría que ha obtenido Operational Insights, configurar Operational Insights y ofrecer funcionalidades y características del panel.

Supervisión y auditoría

Esta es la pantalla de inicio de Operational Insights:



En la sección **Configuración**, se indica que hay conectadas cuatro fuentes de datos. Se trata de cuatro máquinas virtuales conectadas al espacio de trabajo de OMS desde el entorno de prueba y de producción. Se puede adoptar una estrategia diferente: tener un espacio de trabajo distinto para cada entorno y dejar que sean los lectores los que decidan lo que es mejor para sus aplicaciones y soluciones. Operational Insights puede configurarse utilizando la ficha de configuración:



Agentes de OMS

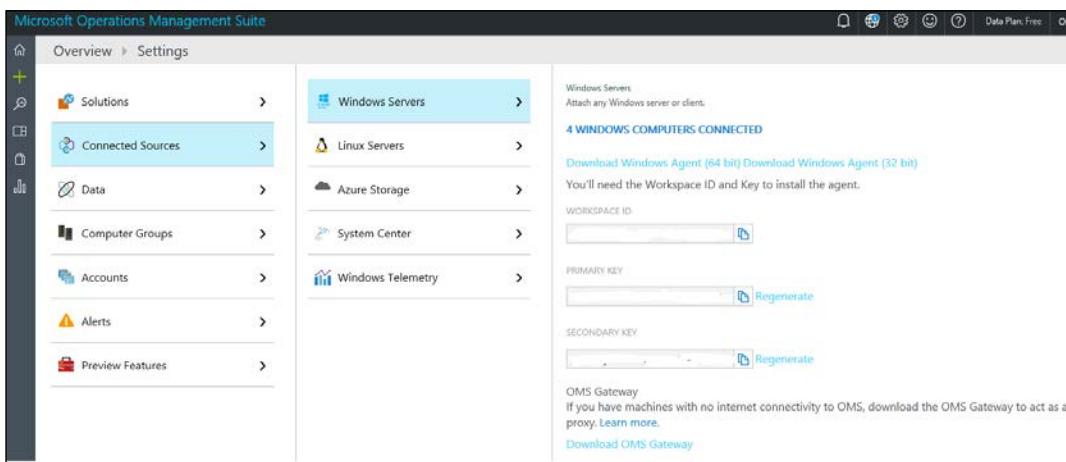
Es posible que te hayas dado cuenta de que no se realiza ningún cambio en el ensamblado ni el código de la aplicación para el consumo de Operational Insights. Operational Insights depende de la instalación de un agente en máquinas virtuales. Estos agentes recopilan datos de telemetría de estos hosts y se los envían al espacio de trabajo de Azure Operational Insights, donde se almacenan durante un período de tiempo determinado en función del SKU elegido. Estos agentes se pueden instalar manualmente en máquinas virtuales. Las extensiones de máquina virtual de Azure Resource Management instalan agentes de forma automática, inmediatamente después de aprovisionar las máquinas virtuales. El código JSON para el aprovisionamiento de un agente en una máquina virtual se muestra en el siguiente código:

```
{
    "apiVersion": "2015-06-15",
    "type": "Microsoft.Compute/virtualMachines/extensions",
    "name": "[concat(variables('vmName'), copyIndex(1), '/omsscript')]",
    "location": "[resourceGroup().location]",
    "dependsOn": [
        "[concat('Microsoft.Compute/virtualMachines/', variables('vmName'), copyIndex(1))]",
        "[resourceId('Microsoft.Compute/virtualMachines/extensions',
        concat(variables('vmName'), copyIndex(1)), 'powershellscript'))"
    ],
    "copy": {
        "count": "[parameters('countVMs')]",
        "name": "omsloop"
    },
    "properties": {
        "publisher": "Microsoft.EnterpriseCloud.Monitoring",
        "type": "MicrosoftMonitoringAgent",
        "typeHandlerVersion": "1.0",
        "settings": {
            "workspaceId": "[parameters('WorkspaceID')]"
        },
        "protectedSettings": {
            "workspaceKey": "[listKeys(variables('accountid'), '2015-11-01-preview').primarySharedKey]"
        }
    }
}
```

Supervisión y auditoría

El ID de espacio de trabajo y el ID de cuenta están disponibles en la ficha de configuración del espacio de trabajo de OMS y el elemento de copia se utiliza para implementarlo en múltiples máquinas virtuales. Este recurso es un recurso secundario del recurso de máquina virtual, lo que garantiza que esta extensión se ejecute siempre que se aprovisione o se actualice una máquina virtual.

La configuración relacionada con el **ID de espacio de trabajo** y el **ID de cuenta** se muestra a continuación. La clave principal se utiliza como **ID de cuenta** para configurar los agentes utilizando plantillas de ARM:



Búsqueda

El espacio de trabajo de la OMS proporciona capacidades de búsqueda para buscar entradas de registro específicas, exportar todos los datos de telemetría a Excel o Power BI y buscar el idioma específico de OMS.

A continuación, se muestra la pantalla **Búsqueda de registros**:

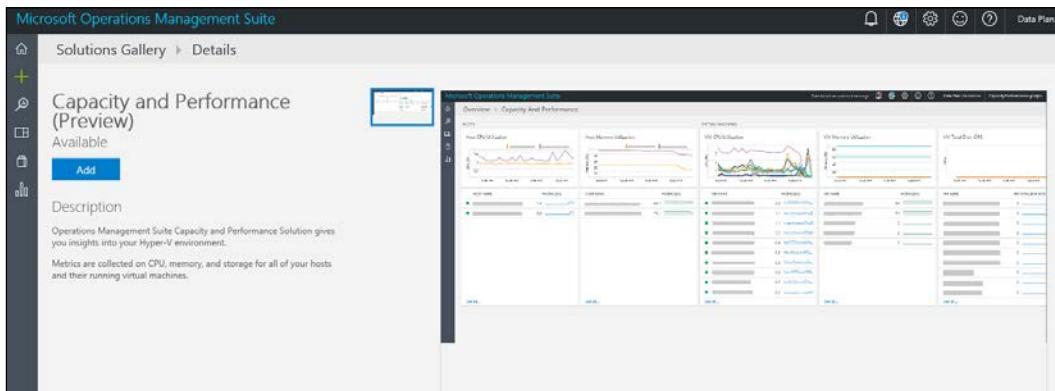
The screenshot shows the Microsoft Operations Management Suite (OMS) Log Search interface. At the top, there's a navigation bar with icons for Home, Log Search, Favorites, History, and a search icon. The main area has a search bar with the placeholder "Begin searching here..." and a search button. Below the search bar, there's a section titled "Let's start searching!" with a dropdown menu showing the query "Type=Perf CounterName = \"% Processor Time\"". To the right of the search bar, there's a detailed description of the query: "measure avg(CounterValue) by Computer interval 30MINUTE". This description includes the note "Get the average processor utilization grouped by individual computer" and "Average over 30 minute periods". Below the search bar, there's a section titled "For more information" with links to "Search Overview", "Query Language Help", "Create alerts on your searches", and "Export search data to Power BI". On the left side, there's a sidebar with various search suggestions like "All collected data", "Count of all data collected grouped by Type", and "All Configuration Changes". Each suggestion has a brief description and a link to "Required Data Collection".

Soluciones

Las soluciones de OMS son capacidades adicionales que pueden añadirse al espacio de trabajo para obtener de datos de telemetría adicionales que no se obtienen de manera predeterminada. Cuando se agregan estas soluciones al espacio de trabajo, los paquetes de administración correspondientes se envían a todos los agentes conectados al espacio de trabajo para que se configuren automáticamente para obtener datos específicos de la solución desde las máquinas virtuales y los contenedores y luego empiecen a enviarlos al espacio de trabajo de OMS.

Supervisión y auditoría

La captura de pantalla siguiente muestra la galería de soluciones y la solución de capacidad y rendimiento en el espacio de trabajo de OMS. Al hacer clic en cualquier solución y posteriormente en el botón **Agregar**, se agrega la solución en contexto en el espacio de trabajo:

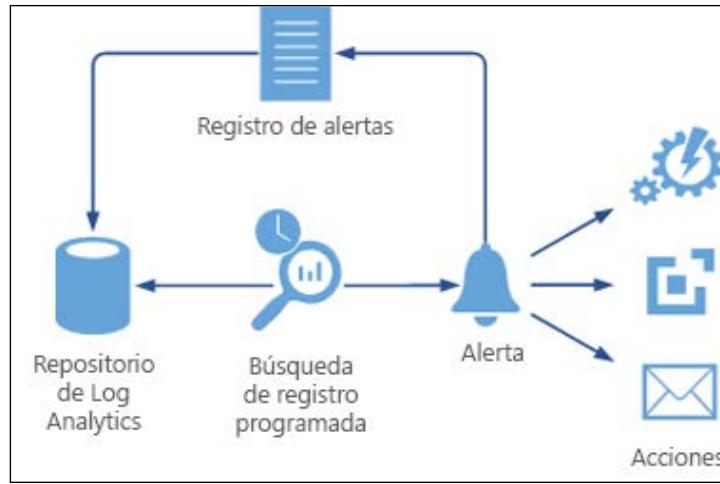


Azure dispone de muchas soluciones de OMS para el seguimiento y la supervisión de diferentes aspectos de los entornos y la aplicación. Como mínimo, se debería añadir al espacio de trabajo un conjunto de soluciones que sean genéricas y puedan aplicarse a casi cualquier entorno.

- Capacidad y rendimiento
- Estado del agente
- Seguimiento de cambios
- Contenedores
- Seguridad y auditoría
- Administración de actualizaciones
- Supervisión del rendimiento de la red

Alertas

Log Analytics proporciona disposiciones para generar alertas sobre los datos ingeridos. Para ello, ejecuta una consulta predefinida compuesta de condiciones en los datos entrantes. Si encuentra un grupo de registros que esté dentro del ámbito de la consulta especificada, genera una alerta. Log Analytics proporciona un entorno altamente configurable para determinar las condiciones para generar alertas, los intervalos de tiempo sobre los que la consulta debe devolver los registros, los intervalos de tiempo en los que debe ejecutarse la consulta y la acción que debe realizarse cuando la consulta devuelve los resultados como alertas:



El primer paso en la configuración de una alerta consiste en crear una búsqueda guardada. Una búsqueda guardada es simplemente una consulta de búsqueda en Log Analytics:

Log Search

Data based on last 1 day

search * | where (Type = "Event")

1 Results List Table

9/27/2017 9:52:55.870 AM | Event
... TimeGenerated : 9/27/2017 9:52:55.870 AM
... Computer : testtwo
... EventName : Information

TYPE (1)
COMPUTER (1)
EVENTLEVELNAME (1)

AllEvents

Save

Supervisión y auditoría

Para guardar la consulta , proporcionale un nombre. Después de guardar la consulta, haz clic en el botón **Alerta** en el menú **Búsqueda de registros**. Proporciona al usuario una interfaz para definir y agregar una nueva regla de alerta:

The screenshot shows the 'Log Search > Add Alert Rule' page. It is divided into three main sections: General, Schedule, and Actions.

- General:** Includes fields for Name (eventalert), Description (this alert is raised whenever events arrives), Severity (Critical), and a Search query dropdown set to 'Use current search query' with the query 'search * | where (Type == "Event")'. Below this is a Time window field set to 15 Minutes.
- Schedule:** Set to check for alerts every 15 Minutes. It also includes a 'Generate alert based on' section with 'Number of results' selected, showing 'Greater than 2'. There is also a 'Suppress alerts' checkbox with a note about reducing noise.
- Actions:** Options for Email notification (Yes), Webhook (Yes), Runbook (Yes), Automation account (datacenterautomation), Select a runbook (TestRunbook), Run on (Azure Hybrid worker), and ITSM Actions (Yes).

At the bottom are 'Save' and 'Cancel' buttons.

Dentro de esa única pantalla, se pueden realizar todas las configuraciones relacionadas con una regla de alerta. Proporciona un nombre, la descripción, la gravedad y la consulta que va a ejecutarse como parte de la evaluación de la regla dentro de sus respectivos campos.

El intervalo de tiempo ayuda a especificar el intervalo de datos en el que se debe ejecutar la consulta. Según la captura de pantalla del ejemplo, siempre que se ejecuta la regla, procesa datos de los últimos 15 minutos.

La sección de la programación ayuda a configurar la frecuencia de ejecución de la regla. Ayuda a contestar la pregunta: ¿con qué frecuencia se debe ejecutar la consulta? Según la captura de pantalla de ejemplo, la regla se ejecuta cada 15 minutos. El intervalo de tiempo debe ser superior a la frecuencia de alerta. Las alertas pueden configurarse adicionalmente en función del número de resultados encontrados. No es necesario que se genere una alerta por cada instancia de datos encontrados en función de la consulta.

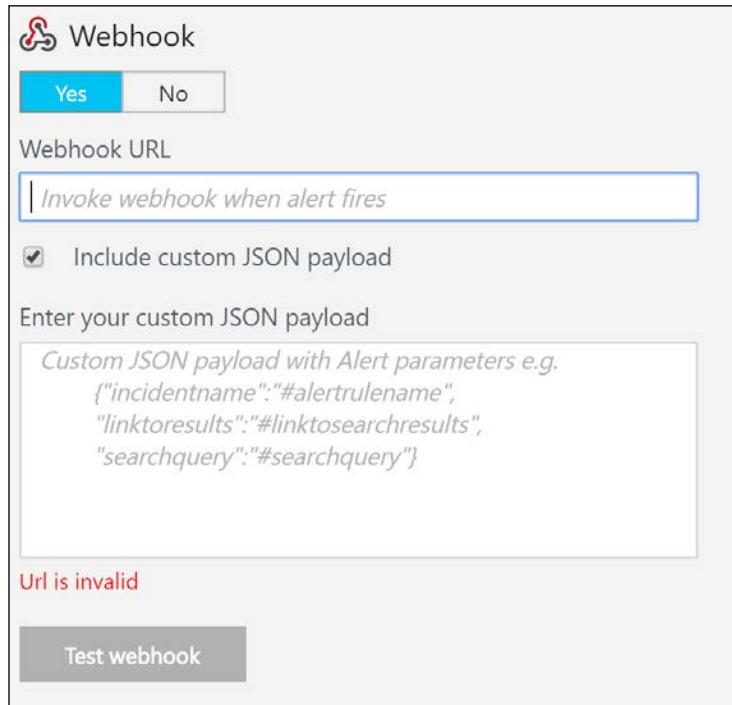
Puede cuantificarse aún más para que se acumule una determinada cantidad de resultados antes de enviar las solicitudes. También se pueden generar alertas basadas en las mediciones de las métricas. Se puede realizar una configuración adicional para suprimir alertas. Esto ayuda a crear un intervalo de tiempo que debe transcurrir antes de ejecutar la acción. En este caso, se generan alertas, pero solo se realiza una acción una vez transcurrido el intervalo de tiempo.

La sección de acciones sirve de ayuda para configurar lo que debe ocurrir después de una alerta. En general, debe producirse una acción correctiva o una notificación. Log Analytics ofrece cuatro maneras diferentes de crear una nueva acción que se pueden combinar de cualquier forma. Una alerta ejecuta todas las acciones configuradas:

- **Notificación por correo electrónico:** es la forma más sencilla, que envía un correo electrónico a los destinatarios configurados:

The screenshot shows a configuration window titled "Actions". Under the "Email notification" section, the "Yes" button is selected. The "Subject" field contains the placeholder text "Email subject". The "Recipients (semi-colon separated)" field contains the email address "callritz@hotmail.com".

- **Webhook:** Webhook ayuda a ejecutar cualquier proceso externo arbitrario utilizando un mecanismo POST de HTTP. Por ejemplo, se puede ejecutar una API de REST o se pueden invocar API de administradores de servicios o ServiceNow para crear un ticket:

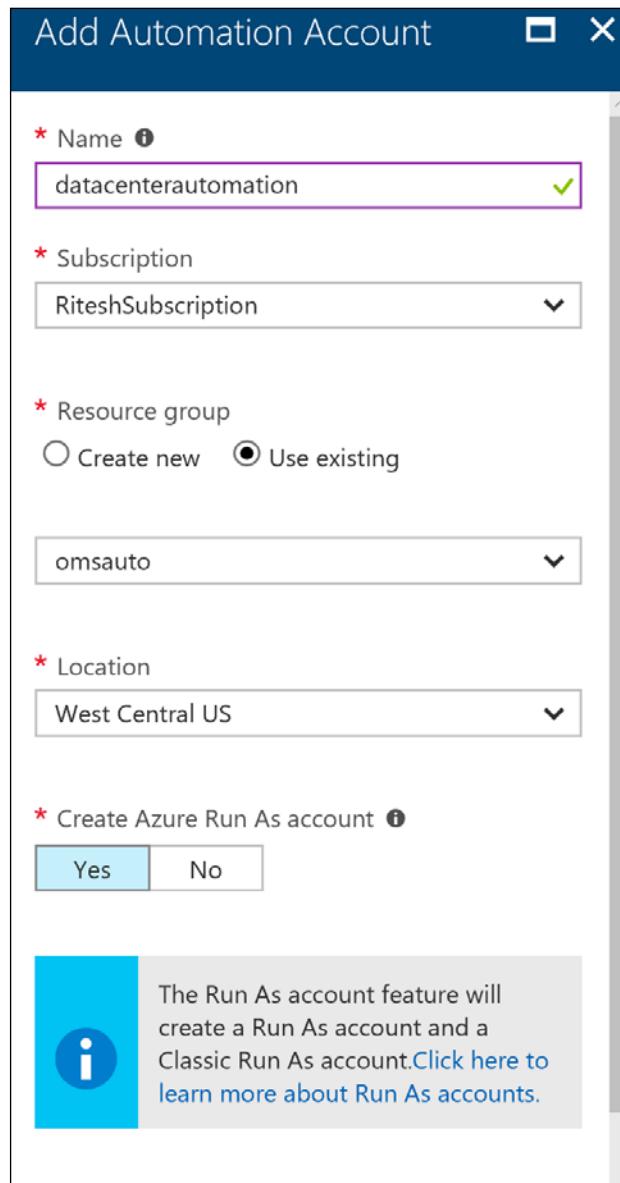


- **Runbooks:** esta acción ejecuta runbooks de automatización de Azure. En la siguiente sección, veremos todo el proceso de ejecución de un runbook de automatización de Azure.
- **Acciones de ITSM:** las soluciones de ITSM se deben aprovisionar antes de usar esta opción, que ayuda a conectar y enviar información a sistemas de ITSM.

Ejecución de runbooks en alertas

Una de las acciones que ofrece una alerta de Log Analytics consiste en ejecutar el runbook de automatización de Azure. Esta utilidad de ejecución de runbooks en una alerta proporciona inmenso poder para actuar sobre la alerta para remediarla, así como para informar a las partes interesadas mediante notificaciones.

1. El primer paso para ejecutar un runbook en respuesta a una alerta es crear una cuenta de automatización de Azure:



Supervisión y auditoría

- Después de aprovisionar la cuenta, crea un runbook solo para demostrar que se puede ejecutar como parte de la generación de alertas. En este caso, el runbook envía un correo electrónico como parte de la notificación. Utiliza credenciales de automatización de Azure para enviar un correo electrónico mediante el servidor SMTP de O365. Los usuarios deben tener una cuenta válida de O365 para poder enviar correos electrónicos utilizando la automatización de Azure:

```
1 $Cred = Get-AutomationPSCredential -Name "O365"
2 if ($Cred -eq $null)
3 {
4     Write-Output "Credential entered: O365 does not exist in the automation service. Please create one `n"
5 }
6 else
7 {
8     $CredUsername = $Cred.UserName
9 }
10
11
12 $requestID = Random
13
14 $body = "<HTML><HEAD><META http-equiv=""Content-Type"" content=""text/html; charset=iso-8859-1"" /><TITLE></TITLE></HEAD>
15 $body += "<BODY style=""background-color:#FFFFFF"" font-size: Small; font-family: TAHOMA; color: #000000""><p>
16 $body += "Please choose one of the options<br />
17 $body += "<a href=""https://azureforarchitects.azurewebsites.net/api/getjobdone?status=decline&requestID=$requestID">" + "Decline" + "</a><br />
18 $body += "<a href=""https://azureforarchitects.azurewebsites.net/api/getjobdone?status=approved&requestID=$requestID">" + "Approve" + "</a><br />
19 $body += "If you approve request will provision a new virtual machine for user..`n
20
21 Send-MailMessage `
22     -To "caliritz@hotmail.com" `
23     -Subject "You have asked approval for a VM !!" `
24     -Body "Your request has been sent to approver. Please wait for a response !!`n
25     -UseSsl `
26     -Port 587 `
27     -SmtpServer "smtp.office365.com" `
28     -From "admin@caliritz@hotmail.com" `
29     -BodyAsHtml `
30     -Credential $Cred
31
32 Write-Output "Mail is now send `n"
33 Write-Output "-----`n"
34
```

- Hay que en cuenta que esto es solo una demostración. El runbook también puede aceptar parámetros y alertas de Log Analytics y enviar un único parámetro de tipo objeto. Este parámetro contiene todos los datos relativos al origen de la alerta, detalles sobre la alerta y la información que está disponible con Log Analytics:

```
1
2
3 param([object] $WebhookData)
4
5 write-output $WebhookData.WebhookName
6 write-output $WebhookData.RequestHeader
7 write-output $WebhookData.RequestBody
8
```

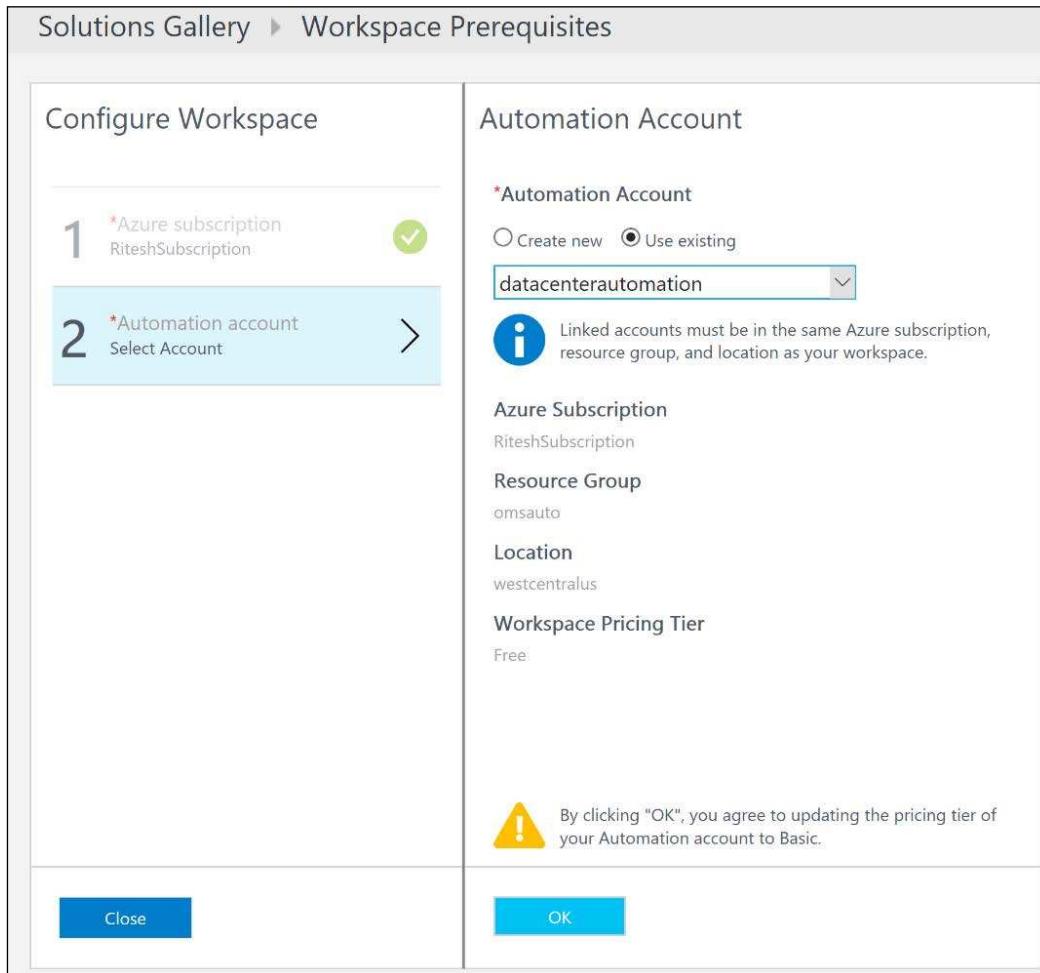
4. Los datos están en el formato JSON y se puede utilizar el cmdlet `ConvertFrom-JSON` para crear objetos de PowerShell.
5. El siguiente paso consiste en configurar Log Analytics para que pueda conectarse a la cuenta de automatización de Azure. Para ello, hay que habilitar e implementar la solución **Automation & Control**:



6. La ventana **Galería de soluciones**: al hacer clic en esta ficha, se abrirá su ventana de configuración. Haz clic en **Configurar el área de trabajo** para implementarla:

The screenshot shows the 'Automation & Control' solution details page. At the top, it says 'Solutions Gallery ▶ Details'. Below that, the title 'Automation & Control' is displayed, followed by a warning message: '⚠️ *requires account configuration*'. There are two buttons: 'Configure Workspace' (blue) and 'Add' (grey). The 'Included solutions:' section lists 'Change Tracking' and 'Update Management' with checked checkboxes. The 'Description' section contains a paragraph about OMS and a bulleted list of features: 'Integrate process automation and configuration for automated delivery of services using PowerShell or graphical authoring', 'Combine change tracking with configuration management to identify and apply desired configurations and enable compliance', and 'Deliver orchestrated update management for both Windows Server and Linux from the cloud'.

7. Selecciona la cuenta de automatización de Azure recién creada como parte de la implementación de la solución:



Supervisión y auditoría

8. Despues de implementar la solución, desplázate a la ventana de configuración dentro del espacio de trabajo de Log Analytics y asegúrate de que en la configuración de automatización de Azure aparezcan los detalles sobre la cuenta de automatización de Azure. De esta manera, garantizas que el espacio de trabajo de Log Analytics esté conectado a la cuenta de automatización de Azure:

The screenshot shows the 'Settings' section of the Azure Log Analytics workspace. On the left, there's a sidebar with options like 'Solutions', 'Connected Sources', 'Data', 'Computer Groups', 'Accounts' (which is highlighted in blue), 'Alerts', 'Preview Features', and 'Upgrade Summary'. On the right, there are three main sections: 'Workspace Information', 'Manage Users', and 'Automation Account'. The 'Automation Account' section is also highlighted in blue. At the top right, there are details about the subscription: 'Subscription ID: 9755ffce-e94b-4332-9be8-1ade15e78909', 'Resource Group Name: omsauto', and 'Automation Account Name: datacenterautomation'.

9. Ahora el runbook debe estar disponible durante la configuración del runbook de la acción de alerta:

The screenshot shows the 'Runbook' configuration dialog. It has a 'Yes' button highlighted in blue. Below it, it says 'Automation account datacenterautomation'. Under 'Select a runbook', 'TestRunbook' is selected. In the 'Run on' section, 'Azure' is selected. There's also a 'Hybrid worker' option.

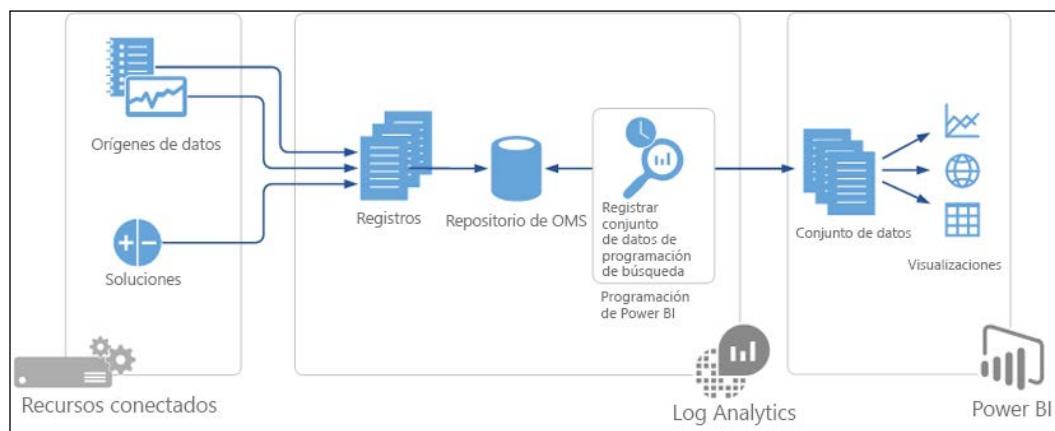
Integración de Power BI

Es importante recopilar datos y almacenarlos en un repositorio central. Sin embargo, debería haber herramientas y utilidades para procesar los datos y generar conocimientos a partir de ellos. Power BI es un servicio de Microsoft diseñado específicamente para la visualización y generación de conocimientos a partir de datos sin procesar.

Power BI puede habilitarse en el menú Configuración igual que la configuración de la automatización de Azure. La conexión a Power BI debe realizarse desde el menú **Configuración**. Una vez que se realice esta conexión, puede utilizarse para enviar datos de Log Analytics a Power BI.

Log Analytics tiene dos maneras diferentes de interactuar con Power BI. Primero debe habilitarse desde el menú **Configuración**.

Como en las alertas, la opción de menú Power BI está disponible en el menú de nivel superior de la búsqueda de registros. Esta opción ayuda a configurar la conectividad de Power BI. Un programador se ejecuta periódicamente para ejecutar consultas de búsqueda y enviar los datos resultantes a Power BI. Los datos se almacenan como conjuntos de datos en Power BI y pueden usarse para generar gráficos, informes y paneles:



Otra forma de obtener datos en Power BI desde Log Analytics consiste en usar el lenguaje de Power Query en Power BI. Aquí se muestra un ejemplo. Aquí se aplica la técnica de scaffolding en el código proporcionado por Log Analytics.

El lenguaje **Power Query Formula Language (Lenguaje M)** puede utilizarse con Power Query en Microsoft Excel y en Power BI Desktop.

Para Power BI Desktop, sigue las instrucciones:

1. Descarga Power BI Desktop de <https://powerbi.microsoft.com/desktop/>.
2. En Power BI Desktop, selecciona **Obtener datos | Consulta en blanco | Editor de consultas avanzado**.
3. Pega el script del lenguaje M en el **Editor de consultas avanzado** y selecciona **Listo**. Al hacerlo, se ejecutará la consulta y los datos de OMS se transferirán a Power BI. Proporciona un nombre y añade un nuevo conjunto de datos haciendo clic en los botones **Aplicar** y **Cerrar** en el editor de consultas de Power BI:

```
let AnalyticsQuery =
let Source = Json.Document(Web.Contents("https://management.
azure.com/subscriptions/9755ffce-e94b-4332-9be8-1ade15e78909/
resourceGroups/omsauto/providers/Microsoft.OperationalInsights/
workspaces/data_centermonitoring/api/query?api-version=2017-01-01-
preview",
[Query=[#"query"="search * | where ( Type == ""Event"" )
#"x-ms-app"="OmsAnalyticsPBI",#"timespan"="PT24H",#"prefer"="ai.
response-thinning=true"],Timeout=#duration(0,0,4,0)])),
TypeMap = #table(
{ "AnalyticsTypes", "Type" },
{
{ "string", Text.Type },
{ "int", Int32.Type },
{ "long", Int64.Type },
{ "real", Double.Type },
{ "timespan", Duration.Type },
{ "datetime", DateTimeZone.Type },
{ "bool", Logical.Type },
{ "guid", Text.Type }
}),
DataTable = Source[tables]{0},
Columns = Table.FromRecords(DataTable[columnns]),
ColumnsWithType = Table.Join(Columns, { "type" }, TypeMap ,
{ "AnalyticsTypes" }),
Rows = Table.FromRows(DataTable[rows], Columns[name]),
Table = Table.TransformColumnTypes(Rows, Table.
ToListAsync(ColumnsWithType, (c => { c{0}, c{3} })))
in
Table
in AnalyticsQuery
```

Resumen

La supervisión es un aspecto importante de la arquitectura de cualquier solución. También es el primer paso hacia la capacidad de auditoría. Permite que el departamento de operaciones pueda administrar la solución, tanto de forma reactiva como proactiva. Ayuda a proporcionar los registros necesarios para localizar y solucionar los problemas que puedan surgir en plataformas y aplicaciones. Hay muchos recursos en Azure que sirven para implementar la supervisión en Azure, otros clouds y centros de datos on-premises. Application Insights, OMS y Log Analytics son algunos recursos importantes a este respecto. Evidentemente, esto imprescindible para innovar con el fin de crear mejores soluciones y productos gracias a los conocimientos que se obtienen con los datos de supervisión.

