

Azure para Arquitetos

Segunda Edição

Implementando o design de nuvem, DevOps, contêineres, IoT e soluções sem servidor em sua nuvem pública



Ritesh Modi

Packt

www.packt.com

Azure para Arquitetos

Segunda Edição

Implementando o design de nuvem, DevOps,
contêineres, IoT e soluções sem servidor em sua
nuvem pública

Ritesh Modi



BIRMINGHAM – MUMBAI

Azure para Arquitetos

Segunda Edição

Copyright © 2019 Packt Publishing

Todos os direitos reservados. Nenhuma parte deste livro pode ser reproduzida, armazenada em um sistema de recuperação ou transmitida sob qualquer forma ou por qualquer meio sem a permissão prévia por escrito da editora, exceto no caso de breves citações incorporadas em artigos críticos ou comentários.

Não se poupou esforços na preparação deste livro para garantir a precisão das informações apresentadas. No entanto, as informações contidas neste livro são vendidas sem garantia, expressa ou implícita. O autor, a Packt Publishing e seus revendedores e distribuidores não serão responsabilizados por quaisquer danos causados ou supostamente causados de forma direta ou indireta por este livro.

A Packt Publishing empenha-se em fornecer informações de marca registrada sobre todas as empresas e produtos mencionados neste livro pelo uso adequado de capitais. No entanto, a Packt Publishing não garante a precisão dessas informações.

Editor de comissionamento: Vijn Boricha

Editor de aquisições: Shrilekha Inani

Editores de desenvolvimento de conteúdo: Abhishek Jadhav

Editor técnico: Aditya Khadye

Editora: Safis Editing

Coordenador de projetos: Jagdish Prabhu

Revisor: Safis Editing

Indexadores: Priyanka dhadke

Gráficos: Tom Scaria

Coordenador de produção: Shraddha Falebhai

Publicado pela primeira vez: outubro de 2017

Segunda edição: janeiro de 2019

Referência de produção: 1310119

Publicado pela Packt Publishing Ltd.

Livery Place

35 Livery Street

Birmingham B3 2PB, Reino Unido.

ISBN 978-1-78961-450-3

www.packtpub.com



mapt.io

O Mapt é uma biblioteca digital online que lhe oferece acesso total a mais de 5.000 livros e cursos, bem como a ferramentas líderes da indústria para ajudá-lo a planejar seu desenvolvimento pessoal e avançar em sua carreira. Para mais informação, visite nosso site.

Por que se inscrever?

- Gaste menos tempo aprendendo e mais tempo codificando com e-books e vídeos práticos de mais de 4.000 profissionais da indústria
- Aprenda melhor com planos de habilidade criados especialmente para você
- Receba um e-book ou vídeo gratuito todo mês
- O Mapt é totalmente pesquisável
- Copie e cole, imprima e marque conteúdos

Packt.com

Você sabia que a Packt oferece versões e-book de todos os livros publicados, com arquivos disponíveis em ePub e PDF? Você pode fazer um upgrade para a versão e-book em www.Packt.com e, como cliente de livro impresso, você tem direito a um desconto na cópia do e-book. Entre em contato conosco pelo e-mail customercare@packtpub.com para obter mais detalhes.

Em www.Packt.com, você também pode ler uma coleção de artigos técnicos gratuitamente, inscrever-se em vários boletins informativos grátis e receber descontos exclusivos e ofertas de livros e e-books da Packt.

Colaboradores

Sobre o autor

Ritesh Modi foi evangelista tecnológico sênior da Microsoft. Ele é diretor regional da Microsoft e lead regional de Treinadores Certificados pela Microsoft.

Ele é arquiteto, evangelista sênior, arquiteto da nuvem, autor publicado, palestrante e um líder reconhecido por suas contribuições para blockchain, Ethereum, data centers, Azure, bots, serviços cognitivos, DevOps, inteligência artificial e automação. Ele é o autor de cinco livros.

Ele fez palestras em mais de 15 conferências, incluindo a TechEd e a PowerShell Asia, e é um autor publicado da MSDN Magazine. Ele tem mais de uma década de experiência na criação e implantação de soluções empresariais para clientes. Ele tem mais de 25 certificações técnicas.

Durante a redação deste livro, cresci pessoalmente como uma pessoa que agora tem mais paciência, perseverança e persistência. Devo agradecer às pessoas que significam o mundo para mim. Estou falando de minha mãe, Bimla Modi, minha esposa, Sangeeta Modi e minha filha, Avni Modi. Agradeço também a equipe da Packt por seu apoio.

Sobre os revisores

Kasam Shaikh, entusiasta do Microsoft Azure, é um profissional experiente com uma atitude "Pode fazer", com 10 anos de experiência na indústria trabalhando como arquiteto de nuvem com uma das principais empresas de TI em Mumbai, Índia. Ele é um arquiteto certificado para o Azure, foi reconhecido como um MVP por uma comunidade online líder, além de ser palestrante global de IA. Ele foi autor de livros sobre o Azure Cognitive, Azure Bots e Microsoft Bot Frameworks. Ele lidera a comunidade do Azure India (az-INDIA), a comunidade online de crescimento mais rápido de aprendizado do Azure.

Alexey Bokov é um experiente arquiteto de nuvem e trabalhou para a Microsoft como especialista técnico e engenheiro sênior do Azure desde 2011, onde ajudou desenvolvedores de software em todo o mundo a desenvolver aplicativos baseados na plataforma do Azure. Sua principal área de interesse é a segurança na nuvem, e principalmente a segurança e proteção de dados para aplicativos organizados em contêineres

A Packt está à procura de autores como você

Se você está interessado em se tornar um autor para a Packt, visite authors.packtpub.com e candidate-se hoje. Temos trabalhado com milhares de desenvolvedores e profissionais de tecnologia, assim como você, para ajudá-los a compartilhar sua visão com a comunidade de tecnologia global. Você pode fazer uma candidatura geral, candidatar-se para um assunto específico para o qual estamos recrutando um autor ou enviar a sua própria ideia.

Sumário

Prefácio	xiii
Capítulo 1: Introdução	1
Computação na nuvem	2
Vantagens da computação na nuvem	3
Padrões de implantação no Azure	3
IaaS	4
PaaS	4
SaaS	4
Compreender o Azure	4
O Azure como nuvem inteligente	6
ARM	6
Arquitetura do ARM	7
Limitações do Gerenciador de Serviços do Azure (ASM)	7
Vantagens do ARM	8
Conceitos do ARM	8
Provedores de recursos	9
Tipos de recursos	9
Grupos de recursos	9
Recursos e instâncias de recursos	10
Recursos do ARM	10
Virtualização	12
Contêineres	12
Docker	14
Interação com a nuvem inteligente	14
O portal do Azure	15
PowerShell	15
Azure CLI	16
API REST do Azure	16

Table of Contents

Modelos do ARM	16
Implantações	17
Resumo	17
Capítulo 2: Disponibilidade e escalabilidade da solução do Azure	19
Alta disponibilidade	20
SLA	21
Fatores que afetam a alta disponibilidade	21
Manutenção planejada	21
Manutenção não planejada	21
Arquitetura de implantação de aplicativo	22
Alta disponibilidade x escalabilidade	22
Alta disponibilidade z Disaster Recovery	22
Alta disponibilidade do Azure	23
Conceitos	23
Conjuntos de disponibilidade	23
O domínio de falha	24
O domínio de atualização	24
Zonas de disponibilidade	24
Balanceamento de carga	25
Alta disponibilidade de VMs	26
Alta disponibilidade de computação	26
Alta disponibilidade de armazenamento	28
Alta disponibilidade de PaaS	28
Plataformas de alta disponibilidade	29
Alta disponibilidade da dados	30
Azure Cosmos DB	30
Replicação do SQL do Azure	31
Armazenamento de tabelas do Azure	31
Alta disponibilidade de aplicativos	31
Balanceadores de carga no Azure	32
Balanceadores de carga do Azure	33
Balanceamento de carga público	33
Balanceamento de carga interno	34
Encaminhamento de porta	36
Gateway de Aplicativo do Azure	36
Gerenciador de Tráfego do Azure	37
Considerações sobre arquitetura para alta disponibilidade	38
Alta disponibilidade em regiões do Azure	39
Alta disponibilidade entre regiões do Azure	40
Práticas recomendadas	41
Alta disponibilidade de aplicativos	41
Implantação	41
Gerenciamento de dados	42
Monitoramento	42

Table of Contents

Escalabilidade	42
Escalabilidade versus performance	44
Escalabilidade do Azure	44
Conceitos	44
Escalabilidade da PaaS	46
PaaS – Expansão e redução verticais	48
PaaS – Expansão e redução horizontais	49
Escalabilidade da IaaS	50
VMSS	51
Arquitetura VMSS	52
Escala VMSS	52
Atualizações e manutenção	55
Atualizações de aplicativos	56
Atualizações de convidados	56
Atualizações de imagens	56
Práticas recomendadas de dimensionamento fornecido pelo VMSS	57
A preferência pela expansão horizontal	57
Instâncias bare metal versus inativas	57
Configuração apropriada do número máximo e mínimo de instâncias	57
Simultaneidade	57
Sem estado	58
Armazenamento em cache e a Rede de Distribuição de Conteúdo (CDN)	58
Design N+1	58
Resumo	58
Capítulo 3: Segurança e monitoramento	59
Segurança	60
Ciclo de vida de segurança	61
Segurança do Azure	63
Segurança de IaaS	64
Grupos de segurança de rede	64
Design de NSG	66
Firewalls	66
Redução da área de superfície de ataque	68
Implementação de servidores de salto	69
Segurança de PaaS	69
Log Analytics	70
Armazenamento	71
SQL do Azure	75
Azure Key Vault	78
Monitoramento e auditoria de segurança	78
Azure Monitor	79
Central de Segurança do Azure	80
Monitoramento	81

Table of Contents

Monitoramento do Azure	82
Logs de atividade do Azure	82
Logs de diagnóstico do Azure	83
Logs de aplicativos do Azure	83
Logs do sistema operacional convidado e host	83
Azure Monitor	84
Azure Application Insights	84
Azure Log Analytics	84
Application Insights	84
Provisionamento	85
Log Analytics	87
Provisionamento	88
Agentes do Log Analytics	90
Pesquisar	92
Soluções	93
Alertas	94
Execução de runbooks em alertas	97
Integração ao Power BI	101
Resumo	104
Capítulo 4: Implantações entre assinaturas usando modelos do ARM	105
Modelos do ARM	106
Implantar grupos de recursos com modelos do ARM	109
Implantar recursos em assinaturas e grupos de recursos	112
Outro exemplo de implantações entre assinaturas e grupos de recursos	113
Implantar entre assinaturas e grupos de recursos usando modelos vinculados	116
Resumo	120
Capítulo 5: Design modular e implementação de modelos do ARM	121
Problemas com um único modelo	122
Flexibilidade reduzida na alteração de modelos	122
Solucionar problemas com modelos grandes	122
Abuso de dependência	122
Agilidade reduzida	122
Sem reutilização	123
Entender o princípio da responsabilidade única	123
Solução de problemas e depuração mais rápidas	123
Modelos modulares	124
Recursos de implantações	124

Table of Contents

Modelos vinculados	125
Modelos aninhados	126
Configurações de fluxo livre	128
Configurações conhecidas	129
Resumo	139
Capítulo 6: Design e implementação de soluções Sem servidor	141
Sem servidor	142
A evolução da função sem servidor	142
Princípios da tecnologia sem servidor	145
As vantagens do Azure Functions	145
FaaS	147
Tempo de execução do Azure Functions	147
Associação e gatilhos do Azure Functions	147
Monitoramento	150
Autenticação e autorização	151
Configuração do Azure Functions	152
Configuração de plataforma	152
Configurações da função de Serviço de Aplicativo	154
Planos de custo do Azure Functions	154
Casos de uso do Azure Functions	155
Tipos de Azure Functions	156
Como criar seu primeiro Azure Functions	156
Como criar uma função orientada a eventos	160
Proxies do Azure Functions	163
Noções básicas sobre fluxos de trabalho	164
Durable Functions	165
Etapas para criar um Durable Functions	166
Como criar uma arquitetura conectada com funções	172
Resumo	176
Capítulo 7: Soluções de integração do Azure	177
Grade de Eventos do Azure	177
A arquitetura da Grade de Eventos	178
Eventos de recursos	181
Eventos personalizados	185
Aplicativos Lógicos do Azure	187
Atividade	188
Conectores	188
Como trabalhar em um aplicativo lógico	188

Table of Contents

Criação de uma solução de ponta a ponta usando tecnologias sem servidor	197
A declaração do problema	197
Visão	197
Solução	198
Arquitetura	198
Automação do Azure	199
Um tópico personalizado	
de Grade de Eventos do Azure	200
Aplicativos Lógicos do Azure	200
Azure Functions	200
Pré-requisitos	200
Implementação	200
Etapa 1	201
Etapa 2	201
Etapa 3	203
Etapa 4	205
Etapa 5	206
Etapa 6	212
Etapa 7	214
Etapa 8	219
Etapa 9	226
Etapa 10	227
Etapa 11	236
Testes	244
Resumo	245
Capítulo 8: Gerenciamento de custos	247
Noções básicas de cobrança	248
Faturamento	252
Clientes do Enterprise Agreement	253
Uso e cotas	254
Provedores de recursos	254
As APIs de uso e cobrança	255
Modelos de preços do Azure	255
Benefício Híbrido do Azure	256
Instâncias de máquina virtual reservadas do Azure	256
Contas pré-pagas	256
Enterprise Agreements	257
O modelo de Provedor de Soluções na Nuvem	257
A calculadora de preços do Azure	257
Práticas recomendadas	260
Práticas recomendadas de computação	260
Práticas recomendadas de armazenamento	261

Table of Contents

Práticas recomendadas de PaaS (Plataforma como Serviço)	262
Práticas recomendadas gerais	263
Resumo	263
Capítulo 9: Criação de políticas, bloqueios e marcas	265
Marcas do Azure	266
Marcas com PowerShell	268
Marcas com modelos do Azure Resource Manager	268
Grupos de recursos versus recursos	269
Políticas do Azure	269
Políticas internas	270
Linguagem de políticas	270
Campos permitidos	272
Bloqueios do Azure	273
Azure RBAC	274
Funções Personalizadas	277
Como os bloqueios são diferentes do RBAC?	277
Um exemplo de implementação de recursos de governança do Azure	277
Contexto	277
RBAC para Company Inc	278
Políticas do Azure	278
Implantações em determinados locais	278
Marcas de recursos e grupos de recursos	278
Logs de diagnóstico	
e Application insights para todos os recursos	279
Bloqueios do Azure	279
Resumo	279
Capítulo 10: Soluções do Azure que usam Serviços de Contêiner do Azure	281
Registro de Contêiner do Azure	282
Instâncias de Contêiner Azure	293
Serviço Azure Kubernetes	297
Arquitetura do Kubernetes	299
nós mestres	299
Pods	300
servidor de API	300
Kubelets	300
Proxy Kube	300
Controlador de replicação/gerenciador de controlador	300
Arquitetura do Azure Kubernetes	301
Como provisionar o Serviço de Kubernetes do Azure	301
Contêineres do Serviço de App	306

Table of Contents

Comparação de todas as opções de contêiner	311
Contêineres em máquinas virtuais	311
Os contêineres em máquinas virtuais com	
Kubernetes como o orquestrador	312
Serviço Azure Kubernetes	312
Contêineres no Serviço de Aplicativo do Azure	313
Contêineres em Instâncias de Contêiner do Azure	313
Contêineres no Azure Functions	314
Contêineres no Service Fabric	314
Resumo	314
Capítulo 11: Azure DevOps	315
DevOps	316
Práticas de DevOps	319
Gerenciamento de configuração	320
Configuração de Estado Desejado	321
Chef, Puppet e Ansible	322
Modelos do ARM	322
Integração contínua	322
Automação de build	324
Automação de testes	324
Empacotamento	324
Implantação contínua	325
Implantação do ambiente de teste	326
Automação de testes	326
Implantação do ambiente de preparo	327
Testes de aceitação	327
Implantação na produção	327
Entrega contínua	327
Aprendizado contínuo	327
Azure DevOps	328
TFVC	330
Git	331
Preparação para DevOps	331
Provisionamento de organização do Azure DevOps	333
Provisionamento do Azure Key Vault	333
Provisionamento de um servidor/serviço	
de gerenciamento de configuração	333
Provisionamento do Log Analytics	334
Conta de Armazenamento do Azure	334
Imagens de origem	334
Ferramentas de monitoramento	334
Ferramentas de gerenciamento	335

DevOps para soluções de PaaS	335
Serviços de Aplicativos do Azure	336
Slots de implantação	337
SQL do Azure	337
O pipeline de build e lançamento	337
DevOps para soluções baseadas em máquinas virtuais (IaaS)	346
Máquinas Virtuais do Azure	347
Balanceadores de carga públicos do Azure	347
O pipeline de build	348
O pipeline de lançamento	349
DevOps para soluções baseadas em contêineres (IaaS)	350
Contêineres	350
Docker	351
Dockerfile	351
O pipeline de build	351
O pipeline de lançamento	352
Azure DevOps e Jenkins	353
Automação do Azure	355
Provisionamento de conta da Automação do Azure	356
Criação de configuração de DSC	357
Importação da configuração de DSC	358
Compilação da configuração de DSC	359
Atribuição de configurações a nós	360
Navegação no servidor	360
Azure para DevOps	361
Resumo	363
Capítulo 12: Soluções OLTP do Azure que usam fragmentação, pools e híbrido de SQL do Azure	365
Serviços de nuvem do Azure	366
Aplicativos de OLTP	367
Bancos de dados relacionais	367
Modelos de implantação	368
Bancos de dados em máquinas virtuais do Azure	368
Bancos de dados hospedados como serviços gerenciados	369
Banco de dados SQL do Azure	369
Recursos do aplicativo	370
Instância única	370
Alta disponibilidade	370
Backups	372

Table of Contents

Replicação geográfica	373
Escalabilidade	375
Segurança	375
Firewall	376
Azure SQL Server em redes dedicadas	377
Bancos de dados criptografados em repouso	379
Máscara de dados dinâmicos	380
Integração do Azure Active Directory	381
Pools elásticos	381
Instância Gerenciada	383
Preços do banco de dados SQL	385
Preços baseados em DTU	385
Preços baseados na vCPU	387
Como escolher o modelo de preços apropriado	388
Resumo	389
Capítulo 13: Soluções de Big Data do Azure com Azure Data Lake Storage e Data Factory	391
Integração de dados	391
ETL	392
Uma cartilha sobre o Data Factory	393
Uma cartilha sobre o Data Lake Storage	394
Entendendo o processamento de big data	395
Ingestão de dados	395
Processamento de dados	395
Armazenamento de dados	396
Apresentação de dados	396
Migração de dados do Armazenamento do Azure para o Data Lake Gen2 Storage	396
Preparação da conta de armazenamento de origem	396
Provisionamento de um novo grupo de recursos	396
Provisionamento de uma conta de Armazenamento	397
Criação de um novo serviço do Data Lake Gen2	398
Provisionamento de pipeline do Azure Data Factory	399
Configurações do repositório	401
Criação do primeiro conjunto de dados	405
Criação do segundo conjunto de dados	408
Criação de um terceiro conjunto de dados	409
Criação de um pipeline	411
Adicionar mais uma atividade de cópia de dados	412
Publicação	413
Resultado final	417
Resumo	417

Table of Contents

Capítulo 14: Azure Stream Analytics e Hubs de Eventos	419
Introdução a eventos	420
Eventos	420
Streaming de eventos	420
Hubs de Eventos	422
Arquitetura dos Hubs de Eventos	423
Grupos de consumidores	428
Taxa de transferência	429
Uma cartilha sobre o Stream Analytics	430
Ambiente de hospedagem	433
Unidades de streaming	433
Exemplo de aplicativo usando Hubs de Eventos e Stream Analytics	434
Provisionamento de um novo grupo de recursos	434
Criação de um namespace de Hubs de Eventos	435
Criação de um hub de eventos	436
Provisionamento de um aplicativo lógico	436
Provisionamento na conta de armazenamento	439
Criação de um contêiner de armazenamento	439
Criação de trabalhos do Stream Analytics	440
Execução do aplicativo	442
Resumo	443
Capítulo 15: Como projetar soluções IoT	445
IoT	445
Arquitetura da IoT	447
Conectividade	448
Identidade	449
Captura	450
Ingestão	450
Armazenamento	450
Transformação	450
Análise	451
Apresentação	452
IoT do Azure	452
Identidade	452
Captura	453
Ingestão	453
Armazenamento	453
Transformação e análise	454
Apresentação	455

Table of Contents

Hubs IoT	455
Protocolos	456
Registro do dispositivo	456
Gerenciamento de mensagens	458
Serviço de mensagens do dispositivo para a nuvem	458
Serviço de mensagens da nuvem para o dispositivo	459
Segurança	460
Segurança na IoT	460
Escalabilidade	461
Edição de SKU	461
Unidades	463
Alta disponibilidade	463
Resumo	464
Outro livro que você pode gostar	465

Prefácio

Ao longo dos anos, os serviços de nuvem do Azure cresceram rapidamente e o número de organizações que adotam o Azure para seus serviços de nuvem também tem aumentado. Os principais gigantes da indústria estão descobrindo que o Azure preenche seus extensos requisitos de nuvem.

Este livro começa com uma introdução abrangente a todas as categorias de designs disponíveis com o Azure. Esses padrões de design concentram-se em diferentes aspectos da nuvem, incluindo alta disponibilidade e gerenciamento de dados. Gradualmente, passaremos por diversos outros aspectos, como a construção de sua arquitetura e implantação de nuvem. Todos os arquitetos devem ter uma boa compreensão de algumas das preocupações importantes de arquitetura relacionadas a qualquer aplicativo. Essas preocupações estão relacionadas à alta disponibilidade, segurança, escalabilidade e monitoramento. Todas elas são muito importantes porque toda a premissa da nuvem depende dessas preocupações importantes. Este livro fornecerá aos arquitetos todas as opções importantes relacionadas à escalabilidade, disponibilidade, segurança e monitoramento das implantações da **Infraestrutura como um Serviço (IaaS)** e da **Plataforma como um Serviço (PaaS)**. Os dados se tornaram um dos aspectos mais importantes dos aplicativos de nuvem. Este livro aborda as considerações de arquitetura e design para implantação de aplicativos de **Processamento de Transações Online (OLTP)** no Azure. Atividades de big data e de dados relacionados, incluindo limpeza de dados, filtragem, formatação e o uso de serviços **Extrair-Transformar-Carregar (ETL)**, são fornecidos pelo serviço Azure Data Factory. Finalmente, as tecnologias sem servidor estão sendo muito acessadas devido a sua orquestração que usa Aplicativos Lógicos do Azure. Isso também será abordado de forma abrangente neste livro.

No fim deste livro, você estará apto para desenvolver uma instância de nuvem completa do Azure.

A quem este livro se destina

Se você for um arquiteto de nuvem, engenheiro de DevOps ou desenvolvedor que deseja aprender sobre os principais aspectos arquitetônicos da plataforma de nuvem do Azure, este livro é destinado a você.

O ideal é ter um conhecimento prévio de nível básico da plataforma de nuvem do Azure.

Tópicos abordados

O Capítulo 1, *Introdução*, apresenta a plataforma de nuvem do Azure. Ele fornece detalhes sobre IaaS e PaaS e uma introdução a alguns dos recursos importantes que ajudam no projeto das soluções.

O Capítulo 2, *Disponibilidade e escalabilidade da solução do Azure*, aborda a perspectiva de um arquiteto para implantar aplicativos altamente disponíveis e escalonáveis no Azure.

O Capítulo 3, *Segurança e monitoramento*, ajuda você a entender como a segurança é, sem dúvida, o requisito não funcional mais importante para os arquitetos implementarem.

O Capítulo 4, *Implantações entre assinaturas que usam modelos ARM*, explica como os modelos ARM são o mecanismo preferencial para provisionamento de recursos.

O Capítulo 5, *Modelos ARM – Design e implementação modulares*, se concentra em escrever modelos Azure Resource Manager (ARM) modulares, sustentáveis e extensíveis.

O Capítulo 6, *Design e implementação de soluções sem servidor*, se concentra em fornecer uma explicação do paradigma sem servidor, do Azure Functions e de seus recursos.

O Capítulo 7, *Soluções de integração do Azure*, é uma continuação do capítulo anterior, continuando a discussão sobre tecnologias sem servidor e abordando a Grade de Eventos do Azure como parte de eventos sem servidor e os Aplicativos Lógicos do Azure como parte de fluxos de trabalho sem servidor.

O Capítulo 8, *Gerenciamento de custos*, se concentra no cálculo do custo de implantação no Azure usando a calculadora de custo do Azure. Ele também demonstra como a alteração do local, do tamanho e do tipo de recursos afeta o custo das soluções e fornece as práticas recomendadas para reduzir o custo geral de implantações do Azure.

O Capítulo 9, *Criação de políticas, bloqueios e marcas*, ajuda você a entender as práticas recomendadas para implementar políticas e bloqueios e como ambos podem trabalhar juntos para fornecer controle completo sobre os recursos do Azure.

O Capítulo 10, *Soluções do Azure que usam Serviços de Contêiner do Azure*, explica sobre os inúmeros serviços, incluindo Serviços de Contêiner do Azure, Registro de Contêiner do Azure e Instâncias de Contêiner do Azure para hospedagem e gerenciamento de contêineres usando serviços de orquestração, como Kubernetes.

O Capítulo 11, *Azure DevOps*, é sobre a adoção e a implementação de práticas que reduzem esse risco consideravelmente e garantem que um software de alta qualidade possa ser entregue ao cliente.

O Capítulo 12, *Soluções OLTP do Azure que usam fragmentação, pools e híbrido de SQL do Azure*, se concentra em vários aspectos do uso do armazenamento de dados de transação, como o Azure SQL, e outros bancos de dados open source normalmente usados em aplicativos OLTP.

O Capítulo 13, *Soluções de Big Data do Azure com Azure Data Lake Storage e Data Factory*, se concentra em soluções de big data no Azure. Estudaremos o Data Lake Storage, o Data Lake Analytics e o Data Factory.

O Capítulo 14, *Azure Stream Analytics e Hubs de Eventos*, diz respeito à criação de soluções para esses eventos. Ele se concentra em ler esses eventos, armazená-los e processá-los e, em seguida, entendê-los.

O Capítulo 15, *Como projetar soluções de IoT*, abrange tópicos relacionados a Hub IoT, Stream Analytics, Hubs de Eventos, registro de dispositivos, conversão de dispositivo em plataforma e registro em log e roteamento de dados para os destinos adequados.

Aproveite ao máximo este livro

Este livro pressupõe que você tenha um nível básico de conhecimento sobre a computação na nuvem e o Azure. Para usar esse livro, tudo o que você precisa é de uma assinatura válida do Azure e conectividade com a Internet. Um sistema operacional Windows 10 com 4 GB de RAM é suficiente para usar o PowerShell e executar os modelos ARM. Clique no link a seguir para criar sua conta do Azure: <https://azure.microsoft.com/enus/free/>.

Faça download dos arquivos EPUB/mobi e exemplo de código

Uma versão EPUB e mobi deste livro está disponível gratuitamente no Github. Você pode fazer download deles e do pacote de código em <https://github.com/PacktPublishing/Azure-for-Architect-Second-Edition>.

Você pode fazer download dos arquivos de exemplo de código para este livro em sua conta em <http://www.packt.com>. Se comprou este livro em outro lugar, você pode visitar <http://www.packt.com/support> e registrar-se para ter os arquivos enviados por email diretamente para você.

Você pode fazer o download dos arquivos de código seguindo estas 3 etapas:

1. Faça logon ou registre-se em <http://www.packt.com>.
2. Selecione a aba **SUPPORT**.
3. Clique em **Downloads de código e Errata**.
4. Digite o nome do livro na caixa de **Pesquisa** e siga as instruções na tela.

Assim que o arquivo é baixado, certifique-se de descompactar ou extrair a pasta usando a versão mais recente do:

- WinRAR / 7-Zip para Windows
- Zipeg / iZip / UnRarX para Mac
- 7-zip / PeaZip para Linux

O pacote de código para o livro também está hospedado no GitHub em <https://github.com/PacktPublishing/Azure-for-Architect-Second-Edition>. Caso haja uma atualização para o código, ele será atualizado no repositório do GitHub existente.

Também temos outros pacotes de código em nosso rico catálogo de livros e vídeos disponíveis em <https://github.com/PacktPublishing/>. Dê uma olhada neles!

Convenções usadas

Existem várias convenções de texto usadas ao longo de todo este livro.

CodeInText: indica palavras de código em texto, nomes de tabela de banco de dados, nomes de pastas, arquivos, extensões de arquivo, nomes de caminhos, URLs fictícios, entrada do usuário e usuários do Twitter. Por exemplo; "Procure o arquivo * .ova extraído para o Kali Linux e clique em **Abrir**".

Um bloco de código é definido da seguinte forma:

```
html, body, #map {  
    height: 100%;  
    margin: 0;  
    padding: 0  
}
```

Quando desejamos chamar a atenção para uma parte específica de um bloco de código, as linhas ou itens relevantes são definidos em negrito:

```
[default]
exten => s,1,Dial(Zap/1|30)
exten => s,2,Voicemail(u100)
exten => s,102,Voicemail(b100)
exten => i,1,Voicemail(s0)
```

Any command-line input or output is written as follows:

```
$ mkdir css
$ cd css
```

Negrito: indica um novo termo, uma palavra importante ou palavras que você vê na tela, por exemplo, em menus ou caixas de diálogo, também aparece em texto como este. Por exemplo: "Selecione **Informações do sistema** no painel **Administração**".



Avisos ou notas importantes aparecem desta forma.



Dicas e truques aparecem desta forma.

Entre em contato

Os comentários dos nossos leitores são sempre bem-vindos.

Comentários gerais: se você tiver dúvidas sobre qualquer aspecto deste livro, mencione o título do livro no assunto da sua mensagem e envie-nos um email para customercare@packtpub.com.

Errata: embora tomemos o máximo de cuidado para garantir a precisão de nosso conteúdo, erros acontecem. Se você encontrou um erro neste livro, ficaríamos gratos se você nos relatasse. Visite <http://www.packtpub.com/submit-errata>, selecione seu livro, clique no link Formulário de envio de errata e insira os detalhes.

Pirataria: se você se deparar com cópias ilegais de nossas obras em qualquer forma na Internet, ficaríamos gratos se você nos enviasse o endereço do local ou nome do site. Entre em contato conosco pelo e-mail copyright@packt.com com um link para o material.

Se você está interessado em se tornar um autor: se há um tópico no qual você tem experiência e está interessado em escrever ou contribuir para um livro, visite <http://authors.packtpub.com>.

Comentários

Por favor, deixe um comentário. Depois de ler e usar este livro, que tal deixar um comentário sobre o site no qual você o comprou? Potenciais leitores poderão ver e usar sua opinião imparcial para fazer decisões de compra, a Packt pode entender o que você acha sobre nossos produtos e nossos autores podem ver seus comentários sobre seu livro. Obrigado!

Para obter mais informações sobre o Packt, acesse packt.com.

1

Introdução

Todos os anos, surgem inovações tecnológicas que mudam todo o cenário e o ecossistema ao redor delas. Se voltarmos no tempo, você lembrará que as décadas de 1970 e 1980 foram o tempo dos mainframes. Esses mainframes eram enormes, ocupando praticamente salas amplas e quase todo o trabalho de computação era realizado por eles. Eram difíceis de adquirir e também era demorado para usar. Muitas empresas costumavam encomendar meses antes de poderem ter um mainframe operacional configurado.

Depois, a primeira parte da década de 1990 foi a era da computação pessoal e da internet. Os computadores ficaram muito menores e eram comparativamente mais fáceis de adquirir. A contínua inovação da computação pessoal e das frentes da internet mudaram toda a indústria de computadores. Muitas pessoas tinham um desktop no qual eram capazes de executar vários programas e se conectar à internet. O surgimento da internet também propagou o surgimento das implantações de cliente-servidor. Agora, poderia haver servidores centralizados hospedando aplicativos e serviços acessados por qualquer pessoa que tivesse uma conexão com a internet em qualquer lugar do mundo. Este foi também um momento em que a tecnologia do servidor ganhou muita proeminência; o Windows NT foi lançado nesta época e logo foi seguido pelo Windows 2000 e Windows 2003, na virada do século.

A inovação mais notável dos anos 2000 foi o surgimento e a adoção de dispositivos portáteis, especialmente os smartphones, e com eles vieram uma infinidade de aplicativos. Os aplicativos podiam se conectar a servidores centralizados na internet e conduzir os negócios normalmente. Os usuários não eram mais dependentes de navegadores para fazer este trabalho; todos os servidores eram auto-hospedados ou hospedados usando um provedor de serviços, como um **Provedor de Serviços de Internet (ISP)**.

Os usuários não tinham muito controle sobre seus servidores. Vários clientes e suas implantações faziam parte do mesmo servidor, mesmo sem os clientes saberem disso.

No entanto, algo mais estava acontecendo a partir de meados da primeira década dos anos 2000. Era o surgimento da computação na nuvem e, novamente, isso transformou todo o cenário da indústria de TI. Inicialmente, a adoção foi lenta e as pessoas a abordaram com cautela, seja porque a nuvem estava em fase inicial e ainda tinha muito a evoluir ou porque as pessoas tinham noções negativas variadas a respeito.

Neste capítulo, vamos abordar os seguintes tópicos:

- Computação na nuvem
- **Infraestrutura como um Serviço (IaaS), Plataforma como um Serviço (PaaS) e Software como um Serviço (SaaS)**
- Compreender o Azure
- **Azure Resource Manager (ARM)**
- Virtualização, contêineres e Docker
- Interação com a nuvem inteligente

Computação na nuvem

Hoje, a computação na nuvem é uma das tecnologias em evolução mais promissoras, e as empresas, não importa seu porte, a estão adotando como parte de sua estratégia de TI. Nos dias de hoje, é difícil ter uma conversa significativa sobre uma estratégia de TI sem incluir a computação na nuvem nas discussões sobre soluções em geral.

A computação na nuvem, ou simplesmente a nuvem em termos gerais, refere-se à disponibilidade de recursos na internet. Esses recursos são disponibilizados para os usuários na internet como serviços. Por exemplo, o armazenamento é disponibilizado sob demanda pela internet para que os usuários possam usá-los para armazenar seus arquivos, documentos e assim por diante. Nesse caso, o armazenamento é um serviço oferecido por um provedor de nuvem.

O provedor de nuvem é uma empresa ou um consórcio de empresas que fornecem serviços de nuvem para outras empresas e consumidores. Eles hospedam e gerenciam esses serviços em nome do usuário. Eles são responsáveis para viabilizar e manter a integridade dos serviços. Normalmente, existem grandes data centers no mundo que foram abertos por provedores de nuvem para atender às demandas de TI dos usuários.

Os recursos de nuvem consistem em hospedar serviços na infraestrutura sob demanda, como computação, redes e instalações de armazenamento. Essa variante da nuvem é conhecida como IaaS.

Vantagens da computação na nuvem

A adoção da nuvem está em alta de todos os tempos e está crescendo por causa de várias vantagens, como estas:

- **Modelo de pagamento conforme o uso:** os clientes não precisam comprar hardware e software para recursos de nuvem. Não há despesa de capital para usar um recurso de nuvem; os clientes pagam apenas pelo tempo que usam um recurso.
- **Acesso global:** os recursos de nuvem estão disponíveis globalmente pela Internet. Os clientes podem acessar seus recursos sob demanda de qualquer lugar.
- **Recursos ilimitados:** a escala da nuvem é ilimitada; os clientes podem provisionar o maior número de recursos que quiserem, sem restrições. Isso também é conhecido como escalabilidade ilimitada.
- **Serviços gerenciados:** o provedor de nuvem fornece inúmeros serviços que são gerenciados por eles para os clientes. Isso elimina todas as responsabilidades técnicas e financeiras do cliente.

Padrões de implantação no Azure

Existem três padrões de implantação diferentes que estão disponíveis no Azure. São eles:

- IaaS
- PaaS
- SaaS

A diferença entre esses três padrões de implantação é o nível de controle que é exercido pelos clientes por meio do Azure. O diagrama a seguir exibe os diferentes níveis de controle dentro de cada um desses padrões de implantação:



Serviços de nuvem – IaaS, PaaS e SaaS

Está claro no diagrama anterior que os clientes têm mais controle ao usar implantações de IaaS e esse nível de controle diminui continuamente à medida que você progride das implantações de PaaS para SaaS.

IaaS

IaaS é um tipo de implantação que permite aos clientes provisionar sua própria infraestrutura no Azure. O Azure fornece vários recursos de infraestrutura e os clientes podem provisioná-los sob demanda. Os clientes são responsáveis por manter e governar sua própria infraestrutura. O Azure garantirá a manutenção da infraestrutura física em que esses recursos de infraestrutura virtual estão hospedados. Com base nessa abordagem, os clientes exigem gerenciamento ativo e operações no ambiente do Azure.

PaaS

A PaaS usa a implantação de infraestrutura e o controle do cliente. Esta é uma abstração de nível mais alto em comparação com a IaaS. Nesta abordagem, os clientes trazem o próprio aplicativo, código e dados e os implantam na plataforma fornecida pelo Azure. Essas plataformas são gerenciadas e controladas pelo Azure e os clientes são os únicos responsáveis por seus aplicativos. Os clientes executam apenas atividades relacionadas à implantação de seus aplicativos. Este modelo fornece opções mais rápidas e fáceis para a implantação de aplicativos em comparação com a IaaS.

SaaS

A SaaS é uma abstração de nível mais alto em comparação com a PaaS. Nesta abordagem, o software e seus serviços estão disponíveis para o consumo do usuário final. Os clientes apenas apresentam seus dados nesses serviços e não têm nenhum controle sobre esses serviços.

Compreender o Azure

O Azure fornece todos os benefícios da nuvem enquanto permanece aberto e flexível. O Azure é compatível com uma ampla variedade de sistemas operacionais, linguagens, ferramentas, plataformas, utilitários e estruturas. Por exemplo, ele é compatível com Linux e Windows, SQL Server, MySQL e PostgreSQL; é compatível com linguagens como C#, Python, Java, Node.js e Bash; é compatível com bancos de dados NoSQL, como MongoDB e Cosmos DB; e também é compatível com ferramentas de integração contínua, com Jenkins e **Visual Studio Team Services (VSTS)**.

A ideia por trás desse ecossistema é permitir que os usuários tenham liberdade de escolha em termos de linguagem, plataforma e sistema operacional, banco de dados, armazenamento, ferramentas e utilitários. Os usuários não devem ficar limitados pela perspectiva da tecnologia, mas devem sim ser capazes de criar e focar em sua solução de negócios, e o Azure fornece a eles um acervo tecnológico de primeira linha para uso.

O Azure é compatível com a escolha de acervo tecnológico do usuário. Por exemplo, o Azure é compatível com todos os ambientes de banco de dados populares (open source e comerciais). O Azure fornece os serviços PaaS pelo Azure SQL, MySQL e Postgres. Fornece um ecossistema Hadoop e oferece o HDInsight, um serviço PaaS 100% baseado no Apache Hadoop. Também fornece o Hadoop na implementação de **máquinas virtuais (VM)** Linux para clientes que preferem a abordagem IaaS. O Azure também fornece um serviço de Cache Redis e oferece suporte a outros ambientes de banco de dados populares, como Cassandra, Couchbase e Oracle como uma implementação IaaS.

O número de serviços está aumentando a cada dia no Azure e a lista mais atualizada de serviços pode ser encontrada em <https://azure.microsoft.com/en-in/services/>.

O Azure também fornece um paradigma de computação na nuvem original – conhecido como nuvem híbrida. A nuvem híbrida refere-se a uma estratégia de implantação em que um subconjunto de serviços é implantado em uma nuvem pública, enquanto outros serviços são implantados em uma nuvem privada ou em um data center na infraestrutura local. Há uma conexão de **Rede Virtual Privada (VPN)** entre a nuvem pública e a privada. O Azure oferece aos usuários a flexibilidade de dividir e implantar seu workload na nuvem pública e em um data center na infraestrutura local.

O Azure tem data centers em todo o mundo, combinados em regiões. Cada região tem vários data centers para garantir que a recuperação de desastres seja rápida e eficiente. Até o momento da redação deste documento, há 38 regiões espalhadas pelo mundo. Isso fornece aos usuários a flexibilidade para implantar serviços no local e na região que eles desejarem. Eles também podem combinar essas regiões para implantar uma solução resistente a desastres e próxima de sua base de clientes.



Na China e na Alemanha, a nuvem do Azure é separada para uso geral e para uso governamental. Isso significa que os serviços de nuvem são mantidos em data centers separados.

O Azure como nuvem inteligente

O Azure fornece infraestrutura e serviços para ingerir milhões e bilhões de transações usando o processamento de hiperescala. Ele fornece vários petabytes de armazenamento para dados, além de uma série de serviços interconectados que podem transferir dados entre si. Com esses recursos implantados, os dados podem ser processados para gerar conhecimento e insights significativos. Há vários tipos de insights que podem ser gerados por meio da análise de dados, que são os insights:

- **Descritiva:** esse tipo de análise fornece detalhes sobre o que está acontecendo ou aconteceu no passado.
- **Preditiva:** esse tipo de análise fornece detalhes sobre o que vai acontecer no futuro próximo ou no futuro.
- **Prescritiva:** esse tipo de análise fornece detalhes sobre o que deve ser feito para melhorar ou evitar eventos atuais ou futuros.
- **Cognitiva:** esse tipo de análise atualmente executa as ações determinadas pela análise prescritiva de forma automatizada.

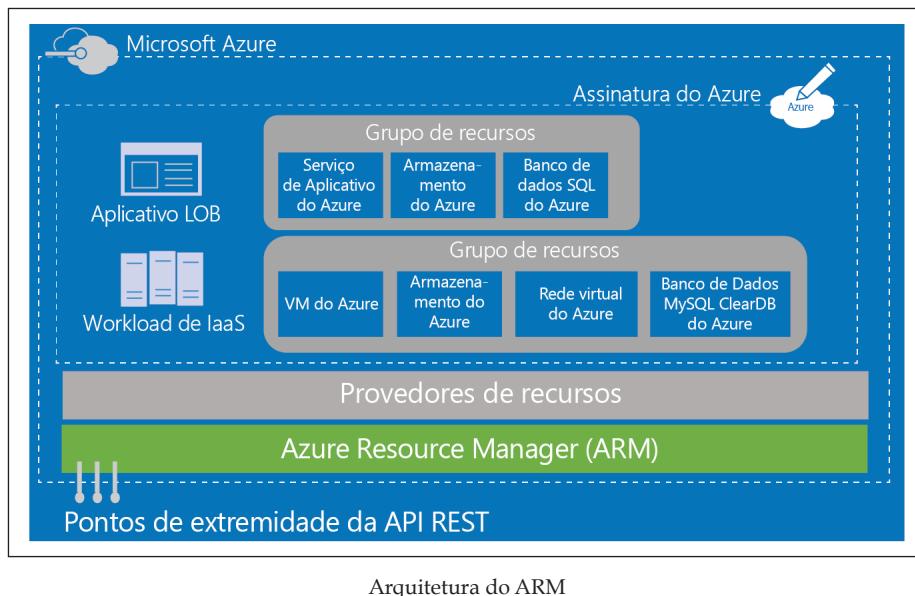
Ao derivar insights de dados é bom, é igualmente importante agir sobre eles. O Azure fornece uma plataforma avançada para ingerir grandes volumes de dados, processá-los e transformá-los, armazenar e gerar insights a partir deles e exibi-los em painéis em tempo real. Também é possível tomar medidas sobre esses insights automaticamente. Esses serviços estão disponíveis para todos os usuários do Azure e fornecem um amplo ecossistema no qual criar soluções. As empresas estão criando inúmeros aplicativos e serviços que estão transformando completamente as indústrias devido à fácil disponibilidade desses serviços inteligentes do Azure, que são combinados para agregar um valor significativo aos clientes finais. O Azure assegura que serviços que eram comercialmente inviáveis para se implementar por pequenas e médias empresas agora podem ser consumidos e implantados prontamente em poucos minutos.

ARM

ARM é o serviço de orquestração e plataforma tecnológica da Microsoft que engloba todos os componentes abordados anteriormente. Ele reúne os provedores de recursos, os recursos e os grupos de recursos do Azure para formar uma plataforma de nuvem coesa. Ele torna os Serviços do Azure disponíveis para assinaturas, disponibiliza os tipos de recursos para os grupos de recursos, torna o recurso e as APIs de recursos acessíveis para o portal e outros clientes, além de autenticar o acesso a esses recursos. Também habilita recursos como identificação, autenticação, **controle de acesso baseado em função (RBAC)**, bloqueio de recursos e aplicação de políticas para as assinaturas e seus grupos de recursos. Ele também fornece recursos de implantação e gerenciamento usando o portal do Azure, o Azure PowerShell e as ferramentas de interface de linha de comando.

Arquitetura do ARM

A arquitetura do ARM e seus componentes são mostrados no diagrama a seguir. Como você pode ver no diagrama a seguir, a **Assinatura do Azure** é composta por vários grupos de recursos. Cada grupo de recursos contém instâncias de recursos que são criadas a partir dos tipos de recursos disponíveis no provedor de recursos:



Arquitetura do ARM

Limitações do Gerenciador de Serviços do Azure (ASM)

O ASM tem restrições inerentes. Por exemplo, as implantações do ASM são lentas e causam bloqueio. As operações serão bloqueadas se uma operação anterior já estiver em andamento. Algumas das limitações do ASM são as seguintes:

- **Paralelismo**: o paralelismo é um desafio no ASM. Não é possível executar várias transações em paralelo com êxito. As operações no ASM são lineares e executadas uma após a outra. Se várias transações forem executadas ao mesmo tempo, haverá erros de operações paralelas ou as transações serão bloqueadas.
- **Recursos**: os recursos no ASM são provisionados e gerenciados isoladamente uns dos outros. Não existe relação entre os recursos do ASM. Não é possível agrupar serviços e recursos nem os configurar em conjunto.
- **Serviços de nuvem**: os serviços de nuvem são as unidades de implantação no ASM. Eles dependem de grupos de afinidade e não são escalonáveis devido ao seu design e arquitetura.

Funções e permissões granulares e discretas não podem ser atribuídas aos recursos no ASM. Os usuários são administradores de serviços ou coadministradores na assinatura. Eles têm controle total sobre os recursos ou não têm acesso algum a eles. O ASM não fornece suporte para implantação. As implantações são manuais ou você precisará recorrer à escrita de scripts de procedimentos no PowerShell ou .NET. As APIs do ASM não são consistentes entre os recursos.

Vantagens do ARM

O ARM fornece as seguintes vantagens e benefícios distintos em relação ao ASM:

- **Agrupamento:** o ARM permite o agrupamento de recursos em um contêiner lógico. Esses recursos podem ser gerenciados em conjunto e passar por um ciclo de vida comum, como um grupo. Isso facilita a identificação de recursos relacionados e dependentes.
- **Ciclos de vida comuns:** os recursos de um grupo têm o mesmo ciclo de vida. Esses recursos podem evoluir e ser gerenciados em conjunto, como uma unidade.
- **RBAC:** funções e permissões granulares podem ser atribuídas a recursos, proporcionando acesso discreto para os usuários. Os usuários também podem ter somente os direitos que lhe são atribuídos.
- **Suporte para implantação:** o ARM fornece suporte para implantação em termos de modelos que viabilizam DevOps e IaC (**Infraestrutura como Código**). Essas implantações são mais rápidas, consistentes e previsíveis.
- **Tecnologia superior:** o custo e a cobrança dos recursos podem ser gerenciados como uma unidade. Cada grupo de recursos pode fornecer suas informações de uso e custo.
- **Capacidade de gerenciamento:** o ARM fornece recursos avançados, como segurança, monitoramento, auditoria e marcação, para melhor capacidade de gerenciamento de recursos. Os recursos podem ser consultados com base em marcas. As marcas também fornecem informações de custo e cobrança para recursos marcados da mesma forma.
- **Migração:** migração mais fácil e atualização de recursos internos e entre grupos de recursos.

Conceitos do ARM

Com o ARM, tudo no Azure é um recurso. Exemplos de recursos são VMs, interfaces de rede, endereços IP públicos, contas de armazenamento e redes virtuais. O ARM é baseado em conceitos relacionados a provedores de recursos e a consumidores de recursos. O Azure fornece recursos e serviços por meio de vários provedores de recursos são consumidos e implantados em grupos.

Provedores de recursos

Esses são os serviços responsáveis por fornecer tipos de recursos por meio do ARM. O conceito de nível superior do ARM engloba o provedor de recursos. Esses provedores são contêineres de tipos de recursos. Os tipos de recursos são agrupados em provedores de recursos. Eles são responsáveis por implantar e gerenciar os recursos. Por exemplo, um tipo de recurso de VM é fornecido por um provedor de recursos chamado **Namespace Microsoft.Compute**. As operações de APIs **Representational State Transfer (REST)** são versionadas para distinguir entre elas. A nomenclatura das versões é baseada nas datas em que elas são lançadas pela Microsoft. É necessário que um provedor de recursos relacionado esteja disponível para uma assinatura para implantar um recurso. Nem todos os provedores de recursos estão disponíveis para uma assinatura de imediato. Se um recurso não estiver disponível na assinatura, você precisará verificar se o provedor de recursos necessário está disponível em cada região. Se estiver disponível, o usuário poderá registrar-se explicitamente na assinatura.

Tipos de recursos

Os tipos de recursos são uma especificação real de recursos que define sua interface e implementação de API pública. Eles implementam o funcionamento e as operações com suporte do recurso. Semelhantes aos provedores de recursos, os tipos de recursos também evoluem ao longo do tempo, no que se refere à sua implementação interna, e têm várias versões de seus esquemas e das interfaces de API pública. Os nomes das versões são baseados nas datas em que elas são lançadas pela Microsoft como uma versão prévia ou **Disponibilidade Geral (GA)**. Os tipos de recursos tornam-se disponíveis como uma assinatura depois que um provedor de recursos é registrado nela. Além disso, nem todos os tipos de recursos estão disponíveis em todas as regiões do Azure. A disponibilidade de um recurso depende da disponibilidade e do registro de um provedor de recursos em uma região do Azure e deve dar suporte à versão da API necessária para o provisionamento dele.

Grupos de recursos

Os grupos de recursos são unidades de implantação no ARM. Eles são contêineres que agrupam várias instâncias de recursos em um limite de gerenciamento e segurança. Um grupo de recursos recebe um nome exclusivo em uma assinatura. Os recursos podem ser provisionados em diferentes regiões do Azure e ainda pertencerem ao mesmo grupo de recursos. Os grupos de recursos fornecem serviços adicionais a todos os recursos do grupo. Os grupos de recursos fornecem serviços de metadados, como marcação, o que permite a categorização dos recursos, o gerenciamento de recursos com base em políticas, o RBAC, a proteção de recursos contra exclusão ou atualizações acidentais, entre outros. Como mencionado anteriormente, eles têm um limite de segurança, e os usuários que não têm acesso a um grupo de recursos não podem acessar os recursos contidos nele. Cada instância de recurso precisa fazer parte de um grupo de recursos; caso contrário, ela não poderá ser implantada.

Recursos e instâncias de recursos

Os recursos são criados a partir de tipos de recursos e devem ser exclusivos dentro de um grupo de recursos. A exclusividade é definida conjuntamente pelo nome do recurso e seu tipo. Se compararmos isso com construções de programação orientadas a objetos, as instâncias de recurso podem ser vistas como objetos e os tipos de recursos podem ser vistos como classes. Os serviços são consumidos por meio das operações com suporte e são implementados por instâncias de recursos. Eles definem as propriedades que devem ser configuradas antes do uso. Algumas propriedades são obrigatórias, enquanto outras são opcionais. Elas herdam a configuração de segurança e acesso de seu grupo de recursos pai. Essas atribuições de funções e permissões herdadas podem ser substituídas para cada recurso. Um recurso pode ser bloqueado de forma que algumas de suas operações possam ser bloqueadas e disponibilizadas para funções, usuários e grupos, mesmo que eles tenham acesso ao recurso. Os recursos podem ser marcados para facilitar a detecção e o gerenciamento.

Recursos do ARM

Aqui estão alguns dos principais recursos fornecidos pelo ARM:

- **RBAC:** o Azure Active Directory (Azure AD) autentica os usuários para fornecer acesso a assinaturas, grupos de recursos e recursos. O ARM implementa OAuth e RBAC na plataforma, permitindo a autorização e o controle de acesso a recursos, grupos de recursos e assinaturas com base nas funções atribuídas a um usuário ou grupo. Uma permissão define o acesso às operações em um recurso. Essas permissões podem permitir ou negar acesso ao recurso. Uma definição de função é uma coleção dessas permissões. As funções mapeiam usuários e grupos do Azure AD para permissões específicas. As funções são atribuídas posteriormente a um escopo, que pode ser um indivíduo, uma coleção de recursos, um grupo de recursos ou a assinatura. As identidades do Azure AD (usuários, grupos e princípios de serviço) adicionadas a uma função ganham acesso ao recurso de acordo com as permissões definidas na função. O ARM fornece várias funções prontas para uso. Ele fornece funções de sistema, como o proprietário, o colaborador e o leitor. Também fornece funções baseadas em recursos, como colaborador de BD SQL e colaborador de VM. O ARM também permite a criação de funções personalizadas.

- **Marcas:** as marcas são pares de nome-valor que adicionam informações e metadados aos recursos. Recursos e grupos de recursos podem ter várias marcas. As marcas ajudam na categorização dos recursos para melhorar a capacidade de descoberta e gerenciamento. Os recursos podem ser pesquisados rapidamente e identificados de forma fácil. Informações de cobrança e custos também podem ser buscadas para os recursos que têm as mesmas marcas. Embora esse recurso seja fornecido pelo ARM, um administrador de TI define seu uso e taxonomia em matéria de recursos e grupos de recursos. A taxonomia e as marcas, por exemplo, podem estar relacionadas a departamentos, uso de recursos, localização, projetos, ou quaisquer outros critérios considerados adequados do ponto de vista de custo, uso, cobrança ou pesquisa. Essas marcas podem, então, ser aplicadas aos recursos. As marcas definidas no nível de grupo de recursos não são herdadas por seus recursos.
- **Políticas:** outro recurso de segurança fornecido pelo ARM são as políticas personalizadas. As políticas personalizadas podem ser criadas para controlar o acesso aos recursos. As políticas são definidas como regras e convenções e devem ser respeitadas durante a interação com os recursos e grupos de recursos. A definição de política contém uma negação explícita de ações nos recursos ou acesso aos recursos. Por padrão, todo acesso é permitido quando não é mencionado na definição de política. Essas definições de política são atribuídas ao escopo de recursos, grupos de recursos e assinaturas. É importante notar que essas políticas não substituem o RBAC (controle de acesso baseado em função). Na verdade, elas complementam e trabalham em conjunto com o RBAC. As políticas são avaliadas depois que um usuário é autenticado pelo Azure AD e autorizado pelo serviço RBAC. O ARM fornece uma linguagem de definição de política baseada em JSON para definir políticas. Alguns exemplos de definição de política são que uma política deve marcar todos os recursos provisionados, e os recursos podem ser provisionados somente para regiões específicas do Azure.
- **Bloqueios:** assinaturas, grupos de recursos e recursos podem ser bloqueados para evitar exclusão ou atualizações acidentais por um usuário autenticado. Os bloqueios aplicados em níveis mais altos se propagam aos recursos filho. Como alternativa, os bloqueios aplicados no nível de assinatura bloqueiam todos os grupos de recursos e os recursos contidos neles.
- **Várias regiões:** o Azure fornece várias regiões para o provisionamento e a hospedagem de recursos. O ARM permite que recursos sejam provisionados em locais diferentes enquanto ainda residam no mesmo grupo de recursos. Um grupo pode conter recursos de regiões diferentes.
- **Idempotente:** esse recurso assegura a previsibilidade, padronização e consistência na implantação de recursos, garantindo que cada implantação resulte no mesmo estado de recursos e sua configuração, não importa quantas vezes o processo seja executado.
- **Extensível:** o ARM fornece uma arquitetura extensível para permitir a criação e conexão de novos provedores de recursos e tipos de recursos na plataforma.

Virtualização

A virtualização foi uma inovação revolucionária que mudou completamente a maneira como os servidores físicos eram vistos. Refere-se à abstração de um objeto físico em um objeto lógico.

A virtualização de servidores físicos levou a servidores virtuais conhecidos como VMs. Essas VMs consomem e compartilham a mesma CPU física, memória, armazenamento e outros componentes de hardware com o servidor físico em que são hospedadas. Isso permitiu um provisionamento mais rápido e fácil de ambientes de aplicativos sob demanda, fornecendo alta disponibilidade e escalabilidade com custo reduzido. Um servidor físico era suficiente para hospedar várias VMs, com cada VM contendo o próprio sistema operacional e serviços de hospedagem.

Não havia mais necessidade de comprar servidores físicos adicionais para implantar novos aplicativos e serviços. Os servidores físicos existentes eram suficientes para hospedar mais VMs. Além disso, como parte da racionalização, o número de servidores físicos foi reduzido com a ajuda da virtualização.

Cada VM contém todo o sistema operacional e é completamente isolada de outras VMs, incluindo os hosts físicos. Embora uma VM use o hardware fornecido pelo servidor host físico, ela tem controle total sobre seus recursos atribuídos e seu ambiente. Essas VMs podem ser hospedadas em uma rede, como um servidor físico com sua própria identidade.

O Azure pode criar VMs do Linux e do Windows em alguns minutos. A Microsoft fornece suas próprias imagens, juntamente com imagens de seus parceiros e da comunidade; os usuários também podem fornecer suas próprias imagens. As VMs são criadas usando essas imagens.

Contêineres

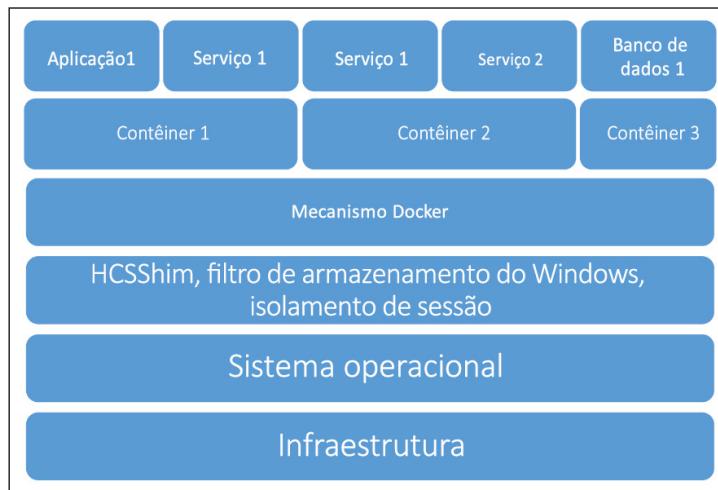
Contêineres também são uma tecnologia de virtualização. No entanto, eles não virtualizam um servidor. Na verdade, um contêiner é uma virtualização no nível do sistema operacional. Isso significa que os contêineres compartilham o kernel do sistema operacional (que é fornecido pelo host) entre si, juntamente com o host. Vários contêineres em execução em um host (físico ou virtual) compartilham o kernel do sistema operacional do host. Os contêineres garantem que eles reutilizem o kernel do host em vez de cada um ter um kernel dedicado a eles mesmos.

Os contêineres são completamente isolados de seu host ou de outros contêineres em execução no host. Os contêineres do Windows usam drivers de filtro de armazenamento e isolamento de sessão do Windows para isolar serviços do sistema operacional, como sistema de arquivos, registro, processos e redes. O mesmo acontece para contêineres do Linux em execução em hosts Linux. Os contêineres do Linux usam o namespace do Linux, os grupos de controle e o sistema de arquivos da união para virtualizar o sistema operacional do host.

O contêiner aparece como se tivesse um sistema operacional e recursos completamente novos e intocados. Esse arranjo fornece muitos benefícios. São eles:

- Os contêineres são rápidos de provisionar e levam alguns segundos para serem criados. A maioria dos serviços em um contêiner é fornecida pelo sistema operacional do host.
- Os contêineres são leves e exigem menos recursos de computação do que as VMs. A sobrecarga de recursos do sistema operacional não é mais necessária com os contêineres.
- Os contêineres são muito menores do que as VMs.
- Os contêineres podem ajudar a resolver problemas relacionados ao gerenciamento de várias dependências de aplicativos de maneira intuitiva, automatizada e simples.
- Os contêineres fornecem a infraestrutura para definir todas as dependências de aplicativos em um único lugar.

Os contêineres são um recurso inerente do Windows Server 2016 e do Windows 10. No entanto, eles são gerenciados e acessados por meio de um cliente do Docker e um daemon do Docker. Os contêineres podem ser criados no Azure com uma SKU do Windows Server 2016 como imagem. Cada contêiner tem um único processo principal que deve estar em execução para o contêiner existir. Um contêiner será interrompido quando esse processo terminar. Além disso, um contêiner pode funcionar no modo interativo ou no modo desconectado como serviço:



Arquitetura de contêineres

O diagrama anterior mostra todas as camadas técnicas que habilitam os contêineres. A camada inferior fornece a infraestrutura básica em termos de rede, armazenamento,平衡adores de carga e placas de rede. Acima da infraestrutura está a camada de computação, que consiste em um servidor físico ou em servidores físicos e virtuais acima de um servidor físico. Essa camada contém o sistema operacional com a capacidade de hospedar contêineres. O sistema operacional fornece o driver de execução que as camadas acima usam para chamar o código do kernel e os objetos para executar os contêineres. A Microsoft criou o **Host Container System Shim (HCSShim)** para gerenciar e criar contêineres e usa drivers de filtro de armazenamento do Windows para gerenciamento de imagens e arquivos.

A capacidade de isolamento do ambiente de contêineres é fornecida para a sessão do Windows. O Windows Server 2016 e o Nano Server fornecem o sistema operacional, habilitam os recursos do contêiner e executam o cliente do Docker e o mecanismo do Docker no nível do usuário. O mecanismo do Docker usa os serviços do HCSShim, os drivers de filtro de armazenamento e as sessões para gerar vários contêineres no servidor, com cada um contendo um serviço, aplicativo ou banco de dados.

Docker

O Docker fornece recursos de gerenciamento a contêineres do Windows. Ele é composto pelos dois executáveis a seguir:

- Daemon do Docker
- Cliente do Docker

O daemon do Docker é útil para o gerenciamento de contêineres. É um serviço do Windows responsável por gerenciar todas as atividades no host relacionadas a contêineres. O cliente do Docker interage com o daemon do Docker e é responsável pela captura de entradas e pelo seu envio ao daemon do Docker. O daemon do Docker fornece tempo de execução, bibliotecas, drivers de gráficos e o mecanismo para criar, gerenciar e monitorar contêineres e imagens no servidor host. Ele também tem a capacidade de criar imagens personalizadas que são usadas para criar e enviar aplicativos para vários ambientes.

Interação com a nuvem inteligente

O Azure fornece várias maneiras de conectar, automatizar e interagir com a nuvem inteligente. Todos esses métodos exigem que usuários sejam autenticados com credenciais válidas para que possam ser usados. As diferentes maneiras de se conectar ao Azure são:

- O portal do Azure
- PowerShell

- **Interface de linha de comando (CLI)** do Azure
- API REST do Azure

O portal do Azure

O portal do Azure é um ótimo lugar para começar. Com o portal do Azure, os usuários podem fazer logon e começar a criar e gerenciar recursos do Azure manualmente. O portal fornece uma interface de usuário intuitiva e fácil de usar por meio do navegador. O Portal do Azure fornece uma maneira fácil de navegar nos recursos usando **folhas**. As folhas exibem todas as propriedades de um recurso, incluindo seus logs, custo, sua relação com outros recursos, marcas, opções de segurança e muito mais. Toda a implantação da nuvem pode ser gerenciada por meio do Portal.

PowerShell

O PowerShell é um shell de linha de comando baseado em objeto e linguagem de script usado para a administração, a configuração e o gerenciamento de infraestrutura e ambientes. Ele é baseado no .NET Framework e fornece recursos de automação. O PowerShell realmente se tornou um cidadão de primeira classe entre administradores de TI e desenvolvedores de automação para gerenciar e controlar o ambiente Windows. Hoje, quase todos os ambientes Windows e muitos ambientes Linux podem ser gerenciados pelo PowerShell. Na verdade, quase todos os aspectos do Azure também podem ser gerenciados pelo PowerShell. O Azure fornece suporte avançado para o PowerShell. Ele fornece um módulo do PowerShell para cada provedor de recursos contendo centenas de cmdlets. Os usuários podem usar esses cmdlets em seus scripts para automatizar a interação com o Azure. O módulo do Azure PowerShell está disponível por meio do instalador da plataforma Web e pela **Galeria do PowerShell**. O Windows Server 2016 e o Windows 10 fornecem gerenciamento de pacotes e módulos PowerShellGet para downloads e instalações rápidos e fáceis de módulos do PowerShell da Galeria do PowerShell. O módulo PowerShellGet fornece o cmdlet `Install-Module` para fazer download e instalar módulos do sistema.

A instalação de um módulo é o simples ato de copiar os arquivos do módulo em locais de módulos bem definidos, o que pode ser feito da seguinte forma:

```
Import-Module PowerShellGet  
Install-Module -Name AzureRM -verbose
```

Azure CLI

O Azure também fornece o Azure CLI 2.0, que pode ser implantado em sistemas operacionais Linux, Windows e macOS. O Azure CLI 2.0 é o novo utilitário de linha de comando do Azure para gerenciar os recursos do Azure. O Azure CLI 2.0 é otimizado para gerenciar e administrar recursos do Azure a partir da linha de comando e para criar scripts de automação que funcionam com o ARM. A CLI pode ser usada para executar comandos usando o shell do Bash ou a linha de comando do Windows. A CLI do Azure é muito famosa entre usuários não Windows, pois nos permite interagir com o Azure no Linux e no macOS. As etapas para instalação do Azure CLI 2 estão disponíveis em <https://docs.microsoft.com/cli/azure/install-azure-cli?view=azure-cli-latest>.

A API REST do Azure

Todos os recursos do Azure são expostos aos usuários por meio de pontos de extremidade REST. As APIs REST são pontos de extremidade de serviço que implementam operações (ou métodos) HTTP, fornecendo **acesso de criação, recuperação, atualização ou exclusão (CRUD)** aos recursos do serviço. Os usuários podem consumir estas APIs para criar e gerenciar recursos. Na verdade, a CLI e o mecanismo PowerShell usam essas APIs REST internamente para interagir com recursos no Azure.

Modelos do ARM

Em uma seção anterior, analisamos os recursos de implantação, como vários serviços, várias regiões, extensível e idempotente, fornecidos pelo ARM. Os modelos do ARM são os principais meios de provisionamento de recursos no ARM. Os modelos do ARM fornecem suporte de implementação para os recursos de implantação do ARM.

Eles fornecem um modelo declarativo por meio do qual recursos, suas configurações, scripts e extensões são especificados. Os modelos do ARM se baseiam no formato **JavaScript Object Notation (JSON)**. Eles usam as convenções e a sintaxe JSON para declarar e configurar recursos. Arquivos JSON são baseados em texto, fáceis de usar e de ler.

Eles podem ser armazenados em um repositório de código-fonte e submetidos ao controle de versão. Eles também servem para representar a IaC que pode ser usada para provisionar recursos em um grupo de recursos do Azure de forma constante, previsível e uniforme. Um modelo precisa de um grupo de recursos para implantação. Ele só pode ser implantado em um grupo de recursos, que deve existir antes de executar a implantação de um modelo. Um modelo não é capaz de criar um grupo de recursos.

Os modelos fornecem a flexibilidade de serem genéricos e modulares no seu design e na sua implementação. Eles fornecem a capacidade de aceitar parâmetros de usuários, declarar variáveis internas, definir dependências entre recursos, vincular recursos dentro do mesmo grupo de recursos ou em grupos de recursos diferentes, além de executar outros modelos. Eles também fornecem expressões e funções do tipo de linguagem de script que os tornam dinâmicos e personalizáveis no tempo de execução.

Implantações

O PowerShell permite os dois modos a seguir para a implantação de modelos:

- **Incremental:** a implantação incremental adiciona recursos declarados no modelo que não existem em um grupo de recursos, deixa os recursos inalterados em um grupo que não faz parte de uma definição de modelo e deixa os recursos inalterados em um grupo existente no modelo e no grupo com o mesmo estado de configuração.
- **Completa:** a implantação completa, por outro lado, adiciona recursos declarados em um modelo ao grupo de recursos, exclui recursos que não existem no modelo do grupo e deixa inalterados os recursos que existem no grupo e no modelo com o mesmo estado de configuração.

Resumo

A nuvem não tem mais de 10 anos. É um novo paradigma e ainda está em fase incipiente. Haverá muita inovação e recursos adicionados ao longo do tempo. O Azure é um dos principais provedores de nuvem do momento e fornece recursos avançados por meio de implantações de IaaS, PaaS, SaaS e híbridas. Na verdade, o Azure Stack, que é uma implementação da nuvem privada da Microsoft, será lançado em breve. Ele terá os mesmos recursos que estão disponíveis em uma nuvem privada na nuvem pública. Ambas poderão, de fato, se conectar e funcionar em conjunto de forma transparente e integrada.

É muito fácil começar a usar o Azure, mas os desenvolvedores e arquitetos também podem cair em uma armadilha se não projetarem e arquitetarem suas soluções adequadamente. Este livro é uma tentativa de fornecer orientações e instruções para arquitetar soluções da maneira certa, usando os serviços e recursos adequados. Cada serviço no Azure é um recurso. É importante compreender como esses recursos são organizados e gerenciados no Azure.

Introdução

Este capítulo forneceu contexto em torno do ARM e de grupos, que são as estruturas básicas que fornecem os blocos de construção para os recursos. O ARM oferece um conjunto de serviços para recursos que ajudam a fornecer uniformidade, padronização e consistência no gerenciamento deles. Os serviços, como o RBAC, marcas, políticas e bloqueios, estão disponíveis para todos os recursos e provedores de recursos. O Azure também fornece funcionalidades avançadas de automação para automatizar e interagir com recursos. Ferramentas como o PowerShell, modelos do ARM e CLI do Azure podem ser incorporadas como parte dos pipelines de lançamento, da implantação contínua e da entrega. Os usuários podem se conectar ao Azure a partir de ambientes heterogêneos usando essas ferramentas de automação.

O próximo capítulo abordará algumas das importantes preocupações de arquitetura que ajudam a resolver problemas comuns da implantação baseada na nuvem e a garantir que o aplicativo seja seguro, disponível, escalonável e de fácil manutenção no longo prazo.

2

Disponibilidade e escalabilidade da solução do Azure

As preocupações com arquitetura, como alta disponibilidade e escalabilidade, são alguns dos itens de maior prioridade para qualquer arquiteto. Isso é comum em muitos projetos e soluções. No entanto, isso se torna ainda mais importante ao implantar aplicativos na nuvem devido à complexidade envolvida. Na maioria das vezes, a complexidade não vem do aplicativo, mas das opções disponíveis em termos de recursos semelhantes na nuvem. O outro problema complexo que surge da nuvem é a disponibilidade constante de recursos mais recentes. Esses novos recursos podem tornar as decisões de um arquiteto completamente redundantes em retrospectiva.

Neste capítulo, analisaremos a perspectiva de um arquiteto para implantar aplicativos altamente disponíveis e escalonáveis no Azure.

O Azure é uma plataforma madura que fornece várias opções para implementar alta disponibilidade e escalabilidade em vários níveis. Para um arquiteto, é essencial conhecê-las, incluindo as diferenças entre elas e os custos envolvidos, e escolher uma solução adequada que atenda aos melhores requisitos. Não há uma única solução, e sim uma boa solução para cada projeto.

A execução de aplicativos e sistemas disponíveis para os usuários para consumo sempre que eles precisam é uma das principais prioridades das organizações. Eles querem que seus aplicativos sejam operacionais, funcionais e continuem disponíveis para os clientes mesmo que algum evento desagradável aconteça. A alta disponibilidade é o tema principal deste capítulo. *Manter as luzes acesas* é a metáfora comum usada para a alta disponibilidade. Alcançar a alta disponibilidade para aplicativos não é uma tarefa fácil, e as organizações precisam gastar tempo, energia, recursos e dinheiro consideráveis para isso. Além disso, ainda há o risco de a implementação de uma organização não produzir os resultados desejados.

O Azure fornece muitos recursos de alta disponibilidade para **máquinas virtuais (VMs)** e o serviço de **Plataforma como Serviço (PaaS)**. Neste capítulo, veremos os recursos de arquitetura e design fornecidos pelo Azure para garantir a alta disponibilidade de aplicativos e serviços.

Neste capítulo, abordaremos os seguintes tópicos:

- Alta disponibilidade
- Alta disponibilidade do Azure
- Considerações sobre arquitetura para alta disponibilidade
- Escalabilidade
- Atualizações e manutenção

Alta disponibilidade

A alta disponibilidade é uma das principais preocupações de qualquer arquiteto. Ela é um dos requisitos técnicos não funcionais essenciais para qualquer serviço sério e sua implantação. A alta disponibilidade refere-se ao recurso de um serviço ou aplicativo que os mantém operacionais continuamente. Para fazer isso, ela atende ou supera o **acordo de nível de serviço (SLA)** prometido. Os usuários recebem a promessa de determinado SLA com base no tipo de serviço. O serviço deve estar disponível para utilização com base em seu respectivo SLA. Por exemplo, um SLA pode definir 99% de disponibilidade de um aplicativo para o ano inteiro. Isso significa que ele deve estar disponível para os usuários 361,35 dias por ano. Se a disponibilidade for menor do que essa, haverá uma violação do SLA. A maioria dos aplicativos de missão crítica define seu SLA de alta disponibilidade com 99,999% de disponibilidade por um ano. Isso significa que o aplicativo deve estar ativo, em execução e disponível o ano todo, mas só pode ficar inativo e indisponível por 5,2 horas.

É importante observar que a alta disponibilidade é definida em termos de tempo (anual, mensal, semanal ou uma combinação desses períodos).

Um aplicativo ou serviço possui vários componentes, que são implantados em camadas e níveis separados. Além disso, um aplicativo ou serviço é implantado em um **sistema operacional (SO)** e hospedado em um computador físico ou VM. Ele consome serviços de rede e armazenamento para diversas finalidades. Tal aplicativo ou serviço pode até mesmo depender de sistemas externos. Para que esses serviços ou aplicativos sejam altamente disponíveis, é importante que as redes, o armazenamento, os sistemas operacionais, os computadores físicos ou as VMs e todos os componentes do aplicativo sejam desenvolvidos considerando o SLA e a alta disponibilidade. Um processo de ciclo de vida do aplicativo definido usado para garantir que a alta disponibilidade tenha respaldo desde o início do planejamento do aplicativo até sua introdução às operações. Isso também envolve a introdução da redundância. Os recursos redundantes devem ser incluídos na arquitetura geral da implantação e do aplicativo para garantir que, se um recurso ficar inativo, o outro assumirá e atenderá às solicitações do cliente.

SLA

O SLA é um acordo entre duas ou mais partes em que uma é o cliente e as outras são provedores de serviços. Aspectos específicos – qualidade, disponibilidade e responsabilidades – são acordados entre o provedor e o usuário do serviço. O componente mais comum do SLA é que os serviços devem ser fornecidos ao cliente conforme acordado no contrato.

Fatores que afetam a alta disponibilidade

Manutenção planejada, manutenção não planejada e arquitetura de implantação de aplicativo são os principais fatores que afetam a alta disponibilidade de um aplicativo. Veremos cada um desses fatores nas seções a seguir.

Manutenção planejada

A manutenção planejada refere-se ao processo de manter atualizados o aplicativo e seu ecossistema ao redor, composto por plataformas, estruturas, software, sistema operacional e drivers de host e convidado, com as versões estáveis mais recentes. É importante corrigir os softwares, drivers e sistemas operacionais com as atualizações mais recentes, pois isso ajuda a manter o ambiente saudável de uma perspectiva de segurança, performance e preparação para o futuro. Não atualizar um ambiente não é uma opção, e isso se tornou uma realidade. Os aplicativos devem até mesmo ser atualizados com hotfixes, bugs e funcionalidade aprimorada. Todas as organizações planejam atualizações de ambiente e aplicativo. Normalmente elas envolvem encerrar e reiniciar o aplicativo e o sistema operacional. Elas também podem envolver a inicialização do sistema operacional do host físico que, por sua vez, reinicia todas as máquinas virtuais convidadas em execução nele. No Microsoft Azure, você pode gerenciar, receber notificações e visualizar as janelas de manutenção planejada para VMs. Você pode encontrar informações mais detalhadas em <https://docs.microsoft.com/azure/virtual-machines/windows/maintenance-notifications>.

Manutenção não planejada

Como o nome sugere, a manutenção não planejada refere-se à manutenção que não pode ser planejada e é ad hoc por natureza. Ela se refere a falhas de hardware e software, como armazenamento corrompido, falha de rede ou roteador, queda de energia e uma série de outras falhas. Bugs na plataforma subjacente que causam a inatividade do aplicativo também fazem parte da manutenção não planejada.

Arquitetura de implantação de aplicativo

A arquitetura de aplicativo desempenha um papel crucial na garantia da alta disponibilidade de um aplicativo. Um aplicativo cujos componentes são implantados em uma única máquina não é altamente disponível. Quando a máquina é reiniciada, o aplicativo não fica disponível para seus usuários. Em outras palavras, um aplicativo pode apresentar tempo de inatividade se qualquer um dos componentes de sua arquitetura não tiver implantações redundantes. Cada componente de um aplicativo deve ser projetado de modo que possa ser implantado em várias máquinas, e a redundância não deve ser um gargalo. Alguns softwares podem fornecer recursos que estão relacionados à alta disponibilidade e não dependem dos sistemas operacionais do host ou de outras ferramentas de terceiros. Os grupos de disponibilidade do SQL Server são um exemplo de tal recurso.

Alta disponibilidade x escalabilidade

Alta disponibilidade é diferente de escalabilidade, embora as duas sejam sérias preocupações de arquitetura. Escalabilidade refere-se à flexibilidade e à elasticidade necessárias para adicionar mais recursos ou reduzi-los para uma implantação existente com o intuito de acomodar mais usuários do que o normal sem comprometer a performance do aplicativo. A escalabilidade ajuda indiretamente a tornar um aplicativo altamente disponível. No entanto, isso não significa que a escalabilidade pode levar à alta disponibilidade. A alta disponibilidade é uma preocupação de arquitetura que não depende do número de usuários. Já as regras de escalabilidade são determinadas pelo número de usuários que consomem o serviço. A alta disponibilidade pode ser um requisito mesmo que haja pouquíssimos usuários. Essencialmente, a alta disponibilidade ocorre quando os serviços estão presentes e operacionais conforme e quando os usuários solicitarem seu consumo. Portanto, ela é uma função de consumo com base no SLA.

Alta disponibilidade x Disaster Recovery

A alta disponibilidade é, novamente, diferente de Disaster Recovery. No entanto, a diferença pode ser muito sutil. A alta disponibilidade é uma função do aplicativo em um estado consumível como e quando o usuário solicitar. Portanto, ela é projetada para operações anteriores a um desastre, enquanto o Disaster Recovery é uma função que aparece depois de um desastre. Disaster Recovery refere-se à implementação da arquitetura por meio da qual os serviços funcionam após um desastre, enquanto a alta disponibilidade cuida da disponibilidade antes de um desastre. O Disaster Recovery inclui backup de dados e servidores arquivados e inativos em todos os continentes, enquanto a alta disponibilidade consiste em平衡adores de carga, distribuição de carga e redundância ativo-passivo e ativo-ativo.

Alta disponibilidade do Azure

É difícil obter alta disponibilidade e atender aos altos requisitos do SLA. O Azure fornece muitos recursos que permitem alta disponibilidade de aplicativos, do sistema operacional host e convidado para aplicativos que usam PaaS. Os arquitetos podem usar esses recursos para obter alta disponibilidade em seus aplicativos usando a configuração, em vez de criar esses recursos do zero ou depender de ferramentas de terceiros.

Nesta seção, veremos as funcionalidades e os recursos fornecidos pelo Azure para tornar os aplicativos altamente disponíveis. Antes de entrarmos em detalhes de arquitetura e configuração, é importante compreender os conceitos relacionados à alta disponibilidade do Azure.

Conceitos

Os conceitos fundamentais fornecidos pelo Azure para obter alta disponibilidade são:

- Conjuntos de disponibilidade
- O domínio de falha
- O domínio de atualização
- Zonas de disponibilidade

Conjuntos de disponibilidade

No Azure, a alta disponibilidade é obtida principalmente por meio de **redundância**. Redundância significa que há mais de uma instância do recurso do mesmo tipo que assume o controle em caso de falha do recurso principal. No entanto, só o fato de ter mais recursos semelhantes não os torna altamente disponíveis. Por exemplo, pode haver várias VMs provisionadas dentro de uma assinatura, mas esse fato por si só não as torna altamente disponíveis. O Azure fornece um recurso conhecido como conjunto de disponibilidade, e ter várias VMs associadas a ele o torna altamente disponível. Para que o conjunto de disponibilidade se torne altamente disponível, é necessário hospedar no mínimo duas VMs nele. Todas as VMs no conjunto de disponibilidade se tornam altamente disponíveis porque são colocadas em racks físicos separados no datacenter do Azure. Durante as atualizações, essas VMs são atualizadas uma por vez, e não todas ao mesmo tempo. Para isso, os conjuntos de disponibilidade fornecem domínios de falha e de atualização. Falaremos mais sobre isso na próxima seção. Resumindo, os conjuntos de disponibilidade fornecem redundância no nível do data center, semelhante ao armazenamento localmente redundante.

É importante observar que os conjuntos de disponibilidade fornecem alta disponibilidade dentro de um data center. Se o datacenter inteiro estiver indisponível, a disponibilidade do aplicativo será afetada. Para garantir que os aplicativos continuem disponíveis mesmo quando um data center ficar indisponível, o Azure introduziu um novo recurso conhecido como **zonas de disponibilidade**, que conheceremos em breve.

O domínio de falha

Quando uma VM é provisionada e atribuída a um conjunto de disponibilidade, ela é hospedada em um domínio de falha. Cada conjunto de disponibilidade tem dois ou três domínios de falha por padrão, dependendo das regiões do Azure. Algumas regiões fornecem dois, enquanto outras fornecem três domínios de falha em um conjunto de disponibilidade. Os domínios de falha não podem ser configurados pelos usuários. Quando várias VMs são criadas, elas são colocadas em domínios de falha separados. Se o número de VMs for maior do que a quantidade de domínios de falha, as VMs adicionais serão colocadas em domínios de falha existentes. Por exemplo, se houver cinco VMs, haverá domínios de falha hospedados em mais de uma VM. Os domínios de falha estão relacionados aos racks físicos no data center do Azure. Os domínios de falha fornecem alta disponibilidade em caso de tempo de inatividade não planejado devido a falhas de hardware, energia e rede. Como cada VM é colocada em um rack diferente com hardware, fonte de alimentação e rede diferentes, outras VMs continuarão sendo executadas se esse rack for desligado.

O domínio de atualização

O domínio de falha cuida do tempo de inatividade não planejado. Já o domínio de atualização gerencia o tempo de inatividade da manutenção planejada. Cada VM também recebe um domínio de atualização. Pode haver até 20 domínios de atualização em um único conjunto de disponibilidade. Os domínios de atualização não podem ser configurados pelos usuários. Quando várias VMs são criadas, elas são colocadas em domínios de atualização separados. Se mais de 20 VMs forem provisionadas em um conjunto de disponibilidade, elas serão colocadas em um modo de distribuição alternada nesses domínios de atualização. Os domínios de atualização cuidam da manutenção planejada.

Zonas de disponibilidade

Esse é um conceito relativamente novo introduzido pelo Azure. Ele é muito semelhante à redundância de zonas de contas de armazenamento. As zonas de disponibilidade colocam instâncias de VMs em data centers separados em uma região para proporcionar alta disponibilidade nela. As zonas de disponibilidade são aplicáveis a muitos recursos no Azure, incluindo VMs, discos gerenciados, conjuntos de escala de VM e平衡adores de carga. Veja a lista completa de recursos compatíveis com a zona de disponibilidade em <https://docs.microsoft.com/azure/availability-zones/az-overview#services-that-support-availability-zones>. A impossibilidade de configurar a disponibilidade entre zonas foi uma falha no Azure por muito tempo. Isso foi corrigido com a introdução das zonas de disponibilidade.

Cada região do Azure é composta por vários data centers. Algumas regiões têm mais data centers, enquanto outras têm menos. Esses data centers dentro da região são conhecidos como zonas. A implantação de VMs em uma zona de disponibilidade garante que elas fiquem em data centers, racks e redes diferentes. Esses data centers em uma região se relacionam com redes de alta velocidade e não há atraso na comunicação entre essas VMs.



Você pode encontrar mais informações sobre as zonas de disponibilidade em <https://docs.microsoft.com/pt-br/azure/availability-zones/az-overview>.



Se um aplicativo precisar de maior disponibilidade e quiser garantir a disponibilidade mesmo que uma região inteira do Azure esteja inativa, o próximo degrau da escada para a disponibilidade será o recurso Gerenciador de tráfego, que será discutido posteriormente neste capítulo.

Balanceamento de carga

O balanceamento de carga, como o nome sugere, refere-se ao processo de balanceamento de uma carga entre VMs e aplicativos. Com uma VM, não há necessidade de balanceador porque a carga inteira está em uma única VM, e não há outra para compartilhá-la. No entanto, com várias VMs contendo o mesmo aplicativo e serviço, é possível distribuir a carga entre elas por meio do balanceamento de carga. O Azure fornece alguns recursos para habilitar o balanceamento de carga:

- **Balanceadores de carga:** o balanceador de carga do Azure ajuda a criar soluções com alta disponibilidade. Na pilha do **Protocolo de Controle de Transmissão (TCP)** há um balanceador de carga no nível de transporte da camada 4. Ele é um balanceador de carga da camada 4 que distribui o tráfego de entrada entre instâncias íntegras dos serviços definidos em um conjunto de balanceamento de carga. Os平衡adores de carga de nível 4 funcionam no nível de transporte e têm informações de rede, como endereço IP e porta, para decidir o destino da solicitação de entrada. Os balanceadores de carga são explicados em mais detalhes posteriormente neste capítulo.
- **Gateways de aplicativo:** o gateway de aplicativo do Azure proporciona alta disponibilidade para seus aplicativos. Ele é um balanceador de carga de camada 7 que distribui o tráfego de entrada entre instâncias íntegras de serviços. Os balanceadores de carga de nível 7 podem funcionar no nível do aplicativo e possuem informações sobre ele, como cookies, HTTP, HTTPS e sessões da solicitação de entrada. Os gateways de aplicativo são explicados em mais detalhes posteriormente neste capítulo. Eles também são usados ao implantar os serviços do Kubernetes do Azure especificamente para cenários em que o tráfego de ingresso da Internet deve ser roteado para os serviços do Kubernetes no cluster.

Alta disponibilidade de VMs

As VMs fornecem recursos de computação. Elas fornecem poder de processamento e hospedagem para aplicativos e serviços. Se um aplicativo for implantado em uma única VM e ela estiver inativa, o aplicativo não ficará disponível. Se um aplicativo for composto por várias camadas e cada uma delas for implantada em sua própria instância única de uma VM, até mesmo um tempo de inatividade em uma única VM poderá tornar o aplicativo todo indisponível. O Azure tenta tornar até mesmo as instâncias de VM única altamente disponíveis 99,9% do tempo, especialmente quando essas VMs usam armazenamento Premium para seus discos.

O Azure fornece um SLA superior para as VMs agrupadas em um conjunto de disponibilidade. Ele fornece um SLA de 99,95% para a disponibilidade de VMs que fazem parte de um conjunto de disponibilidade com duas ou mais VMs. O SLA será de 99,99% se as VMs forem colocadas em zonas de disponibilidade.

Alta disponibilidade de computação

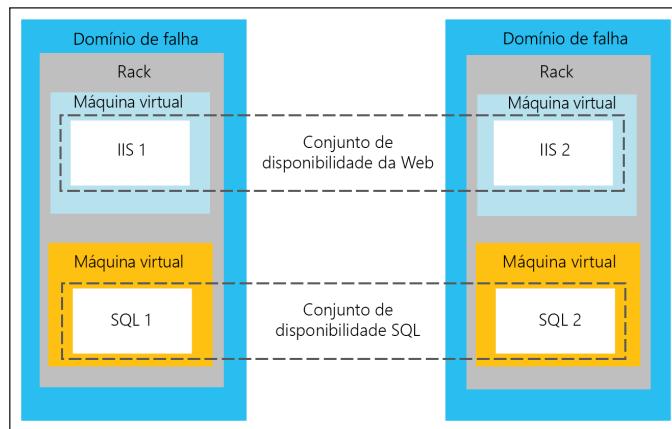
Os aplicativos que exigem alta disponibilidade devem ser implantados em várias VMs no mesmo conjunto de disponibilidade. Se os aplicativos forem compostos por várias camadas, cada camada deverá ter um grupo de VMs em seu conjunto de disponibilidade dedicado. Resumindo, se houver três camadas de um aplicativo, deverá haver três conjuntos de disponibilidade e no mínimo seis VMs (duas em cada conjunto de disponibilidade) para tornar todo o aplicativo altamente disponível.

Então, como o Azure fornece um SLA e alta disponibilidade para VMs em um conjunto de disponibilidade com várias VMs em cada conjunto de disponibilidade? Essa é a pergunta que pode vir à mente.

Aqui, o uso dos conceitos que consideramos antes entra em ação – isto é, os domínios de falha e atualização. Quando o Azure vê várias VMs em um conjunto de disponibilidade, ele as coloca em um domínio de falha separado. Em outras palavras, essas VMs são colocadas em racks físicos separados, e não no mesmo rack. Isso garante que pelo menos uma VM continue disponível mesmo que haja uma falha de energia, hardware ou rack. Há dois ou três domínios de falha em um conjunto de disponibilidade e, dependendo do número de VMs em um conjunto de disponibilidade, as VMs são colocadas em domínios de falha separados ou repetidas em um modo de distribuição alternada. Isso garante que a alta disponibilidade não seja afetada pela falha do rack.

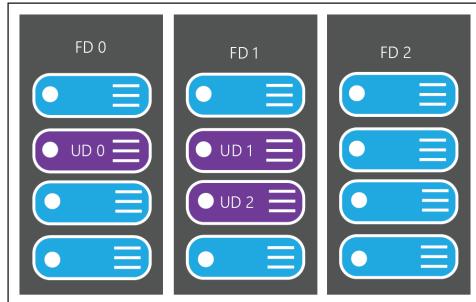
O Azure também coloca essas VMs em um domínio de atualização separado. Em outras palavras, o Azure marca essas VMs internamente de modo que elas sejam corrigidas e atualizadas uma após a outra e que qualquer reinicialização em um domínio de atualização não afete a disponibilidade do aplicativo. Isso garante que a alta disponibilidade não seja afetada pela manutenção da VM e do host.

Com a colocação de VMs em domínios de falha e atualização separados, o Azure garante que nem todas elas fiquem inativas ao mesmo tempo, garantindo que pelo menos algumas permaneçam ativas e disponíveis para atender às solicitações, mesmo que estejam em manutenção ou enfrentando desafios de inatividade física:



O diagrama anterior mostra quatro VMs (duas têm **Serviços de Informações da Internet (IIS)** e as outras duas têm o SQL Server instalado). As VMs IIS e SQL fazem parte de um conjunto de disponibilidade. As VMs IIS e SQL estão em domínios de falha separados e em racks diferentes no data center. Elas também estão em domínios de atualização separados.

O diagrama a seguir mostra a relação entre os domínios de falha e atualização:



Alta disponibilidade de armazenamento

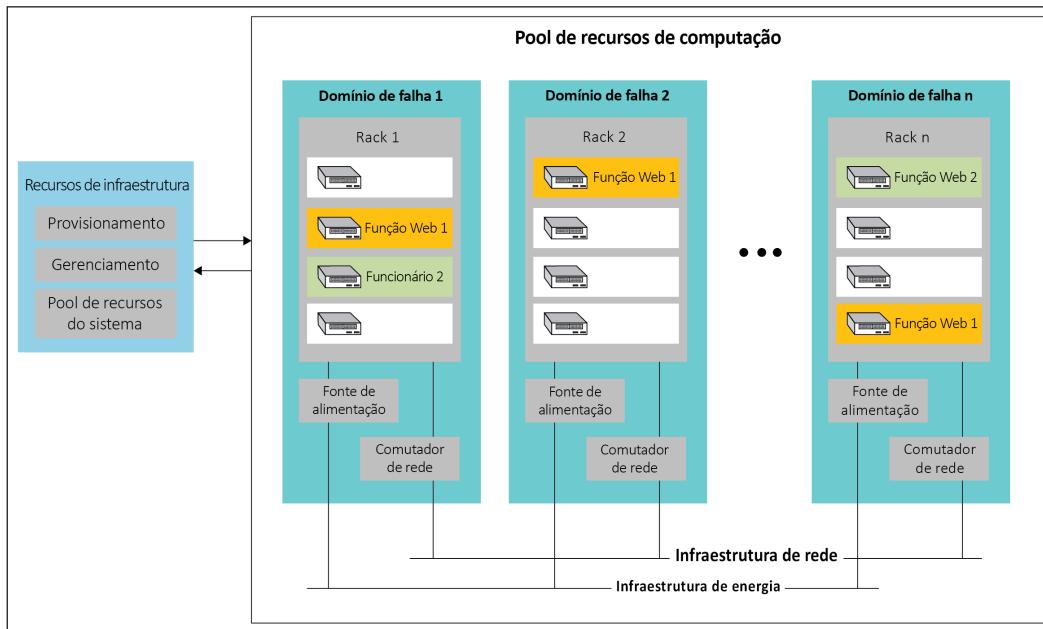
Para fazer backup das VMs, é necessário copiar os arquivos do **disco rígido virtual (VHD)** delas. Embora os conjuntos de disponibilidade ofereçam alta disponibilidade para instâncias de computação, eles não garantem a alta disponibilidade de arquivos VHD para VMs armazenadas em contas de armazenamento. Os arquivos VHD de todas as VMs podem ser colocados no mesmo cluster de armazenamento, e qualquer falha no cluster pode tornar todas as VMs indisponíveis ou menos disponíveis do que o necessário. Resumindo, não são apenas os serviços computados que precisam estar altamente disponíveis, mas até mesmo as contas que armazenam os arquivos VHD devem ser colocadas em clusters separados de modo que, em caso de falha, pelo menos uma ou algumas VMs continuem disponíveis da perspectiva do computador e do armazenamento.

O Azure fornece discos gerenciados e instalações de gerenciamento de discos. Os discos gerenciados fornecem melhor confiabilidade para os conjuntos de disponibilidade, garantindo que os discos das VMs em um conjunto de disponibilidade sejam suficientemente isolados uns dos outros para evitar pontos únicos de falha. O Azure faz isso colocando automaticamente os discos em clusters de armazenamento diferentes. Se um cluster de armazenamento falhar devido à falha de hardware ou software, somente as instâncias de VM com discos nesses carimbos falharão. Cada VHD de VM em um conjunto de disponibilidade deve ser colocado em uma conta de armazenamento separada, embora as VMs de diferentes conjuntos de disponibilidade possam ser colocadas na mesma conta de armazenamento. O Azure fornece discos gerenciados como uma alternativa às contas de armazenamento padrão. Esses discos gerenciados colocam os arquivos VHD automaticamente nas contas de armazenamento apropriadas internamente, e os usuários não precisam criar uma conta de armazenamento. Ao criar um disco gerenciado em vez de contas de armazenamento, os usuários podem delegar o gerenciamento do armazenamento ao Azure.

Alta disponibilidade de PaaS

O Azure fornece serviços de aplicativo e de nuvem para hospedar plataformas gerenciadas. Serviços e aplicações podem ser implantados nessas plataformas. Eles fornecem flexibilidade, elasticidade e economia para criar e implantar aplicativos. Essas plataformas são gerenciadas pelo Azure e os usuários não interagem com a infraestrutura de base na qual elas estão implantadas. Elas trazem um nível mais alto de abstração em comparação com a **Infraestrutura como serviço (IaaS)**, pois permitem que os desenvolvedores se concentrem em seus problemas de negócios e usam a plataforma para acelerar o processo de desenvolvimento e implantação. Isso permite que elas gerenciem, operem e monitorem a infraestrutura de base. Quando um aplicativo é implantado em serviços de aplicativo ou de nuvem, o Azure provisiona VMs não visíveis para os usuários. Os aplicativos são implantados nessas VMs, e os recursos de infraestrutura do Azure são responsáveis por provisioná-los, gerenciá-los e monitorá-los. Os recursos de infraestrutura monitoram o status do hardware e software das instâncias de máquinas host e convidadas. Ao detectarem uma falha, eles mantêm os SLAs realocando automaticamente as instâncias de VM.

Quando várias instâncias da função de serviço de nuvem são implantadas, o Azure as implanta em domínios de falha e atualização diferentes:



O diagrama anterior mostra os serviços de PaaS com várias instâncias de VM implantando essas funções da Web e de trabalho em domínios de falha separados. Implantá-las em domínios de falha separados significa implantá-las em racks separados em um data center. Isso também significa que esses serviços têm comutadores de rede e fontes de alimentação separados. Isso garante que, mesmo que um dos racks passe por manutenção ou haja interrupção da alimentação do rack ou falha do comutador de rede, haverá outras instâncias disponíveis para atender às solicitações do cliente.

Plataformas de alta disponibilidade

O Azure introduziu muitos novos recursos nos últimos tempos em relação à alta disponibilidade para PaaS. Um deles está relacionado a contêineres e ao ecossistema que os rodeia. O Azure introduziu os seguintes serviços:

- Contêineres em serviços de aplicativo
- Grupos de instâncias de contêiner do Azure
- Serviços do Kubernetes do Azure
- Outros orquestradores de contêiner, como DC/OS e Swarm

A outra plataforma importante que traz alta disponibilidade é o **Service Fabric**. Os orquestradores do Service Fabric e do contêiner que incluem o Kubernetes garantem que o número desejado de instâncias do aplicativo estejam sempre em execução em um ambiente. Portanto, mesmo que uma das instâncias fique inativa no ambiente, o orquestrador saberá disso por meio de monitoramento ativo e irá gerar uma nova instância em outro nó, mantendo assim o número ideal de instâncias. Ele faz isso sem qualquer interferência manual ou automatizada do administrador.

Enquanto o Service Fabric permite que qualquer tipo de aplicativo se torne altamente disponível, os orquestradores, como o Kubernetes, DC/OS e Swarm, são específicos dos contêineres. Além disso, é importante entender que essas plataformas fornecem recursos que ajudam a lançar atualizações, em vez de uma grande atualização de banco de dados que pode afetar a disponibilidade do aplicativo.

Alta disponibilidade de dados

Embora as VMs, os serviços de aplicativo e os contêineres forneçam alta disponibilidade para computação e os discos gerenciados forneçam alta disponibilidade para armazenamento, também precisamos garantir que nossos dados e plataformas de dados estejam altamente disponíveis.

O Azure fornece os seguintes recursos que tornam os dados altamente disponíveis.

Azure Cosmos DB

O Azure Cosmos DB é um armazenamento de dados NoSQL georeplicado verdadeiramente global e altamente disponível. O Cosmos DB está disponível em quase todas as regiões do Azure, e também é possível configurar a georeplicação entre todas essas regiões. O Cosmos DB permite criar coleções que são replicadas em várias regiões de maneira assíncrona. Ele também fornece flexibilidade na determinação do nível de consistência enquanto configura a estratégia de replicação para alta disponibilidade. Esses níveis de consistência podem ajudar um arquiteto a determinar a natureza crítica da disponibilidade de dados em outras regiões. Esses níveis de consistência são:

- **Forte:** garante que cada região replicada obtenha seus dados antes de retornar ao usuário.
- **Desatualização limitada:** garante que os dados não fiquem desatualizados em regiões de leitura além de um determinado ponto – ou seja, um número fixo de gravações ou um intervalo de tempo.
- **Sessões:** garante que os dados sejam consistentes em uma sessão.

- **Prefixos ordenados:** ocorrem quando as gravações vêm para regiões replicadas em uma ordem semelhante à medida que são escritas na região de gravação.
- **Eventual:** é possível ter leituras sujas aqui, e não há SLA para determinar a disponibilidade de dados. Ele segue o princípio da consistência eventual.

Replicação do SQL do Azure

O SQL do Azure fornece a replicação do banco de dados para outras regiões para torná-lo altamente disponível. A replicação pode ser feita em qualquer região. No entanto, um arquiteto deve escolher uma região de mesmo nível para replicação. Essas regiões de mesmo nível estão há pelo menos 483 km de distância e são conectadas com redes de alta velocidade. Elas também são corrigidas uma de cada vez e, portanto, não há risco de que a correção aconteça paralelamente a essas regiões.

Os dados no site replicado podem ser disponibilizados no modo leitura para os aplicativos.

Armazenamento de tabelas do Azure

O Azure também fornece armazenamento de tabela, que são os dados de chave-valor armazenados em uma conta de armazenamento do Azure. O Azure mantém três cópias dos dados e as disponibiliza em tempos de necessidade. Os dados são armazenados em partições. Cada partição é identificada usando uma chave de partição, e cada linha recebe um ID de linha. O ID de linha e de partição fazem parte da carga de dados. Eles fornecem armazenamento para dados sem um esquema, de maneira semelhante aos armazenamentos de dados NoSQL. Na verdade, os dados NoSQL podem ser armazenados em tabelas do Azure facilmente.

Uma conta de armazenamento pode ter várias tabelas, e cada tabela armazena entidades identificadas usando identificadores de partição e linha.

Alta disponibilidade de aplicativos

A alta disponibilidade pode ser um recurso interno do software usado para aplicativos. Caso contrário, ela é criada do zero nos aplicativos. Um exemplo do recurso de alta disponibilidade fornecido pelo software são os grupos de disponibilidade AlwaysOn do SQL Server. Eles ajudam a manter os bancos de dados altamente disponíveis.

Os Serviços do Azure também têm um mecanismo de alta disponibilidade interno. No SQL do Azure, os dados são replicados de maneira síncrona na região. A replicação geográfica ativa permite até quatro cópias de banco de dados adicionais na mesma região ou em regiões diferentes. O armazenamento do Azure tem seu próprio mecanismo para disponibilizar dados replicando-os para vários data centers e regiões.

Balanceadores de carga no Azure

O Azure fornece dois recursos que têm a funcionalidade de um balanceador de carga. Ele fornece um balanceador de carga de nível 4 que funciona na camada de transporte dentro da pilha TCP OSI e outro de nível 7 (gateway de aplicativo) que funciona na camada do aplicativo e da sessão.

Embora os gateways de aplicativo e os平衡adores de carga fornecem os recursos básicos para balancear a carga, eles têm finalidades diferentes. Há diversos casos de uso em que faz mais sentido implantar um gateway de aplicativo, em vez de um balanceador de carga.

O gateway de aplicativo fornece os seguintes recursos que não estão disponíveis com os balanceadores de carga do Azure:

- **Firewall do Aplicativo Web:** é um firewall adicional, além do firewall do sistema operacional, e tem a capacidade de inspecionar as mensagens recebidas. Isso ajuda a identificar e evitar ataques comuns na Web, como injeção de SQL, ataques de script entre sites e sequestros de sessão.
- **Afinidade de sessão baseada em cookies:** os balanceadores de carga distribuem o tráfego de entrada para instâncias de serviço que estão íntegras e relativamente livres. Uma solicitação pode ser atendida por qualquer instância de serviço. No entanto, há aplicativos que precisam de recursos avançados em que todas as solicitações subsequentes à primeira devem ser processadas pela mesma instância de serviço. Isso é conhecido como afinidade de sessão baseada em cookies. O gateway de aplicativo fornece afinidade de sessão baseada em cookies para manter uma sessão de usuário na mesma instância de serviço usando cookies.
- **Descarga SSL (Secure Sockets Layer):** a criptografia e a descriptografia de dados de solicitação e resposta são executadas pelo SSL, e essa operação geralmente é cara. O ideal é que os servidores Web gastem recursos no processamento e atendimento das solicitações, e não na criptografia e descriptografia do tráfego. A descarga SSL ajuda a transferir esse processo de criptografia do servidor Web para o balanceador de carga, fornecendo assim mais recursos aos servidores Web que atendem aos usuários. A solicitação do usuário é criptografada, mas é descriptografada no gateway de aplicativo, e não no servidor Web. A solicitação do gateway de aplicativo para o servidor Web não é criptografada.
- **SSL de ponta a ponta:** embora a descarga SSL seja um bom recurso para um determinado aplicativo, há certos aplicativos seguros de missão crítica que precisam de criptografia e descriptografia SSL completas, mesmo que o tráfego passe por平衡adores de carga. O gateway de aplicativo também pode ser configurado para uma criptografia SSL de ponta a ponta.

- **Roteamento de conteúdo baseado em URL:** os gateways de aplicativo também são úteis para redirecionar o tráfego para servidores diferentes, com base no conteúdo do URL das solicitações de entrada. Isso ajuda na hospedagem de vários serviços junto com outros aplicativos.

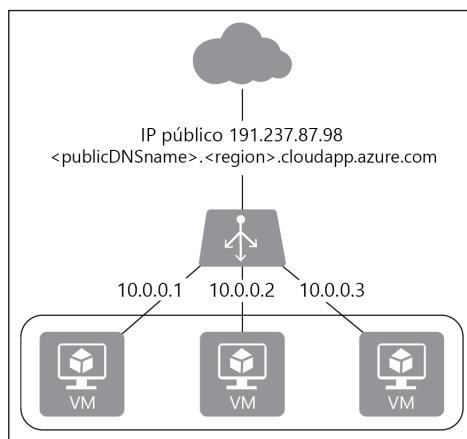
Balanceadores de carga do Azure

Um balanceador de carga do Azure distribui o tráfego de entrada com base nas informações de nível de transporte disponíveis para isso. Esse processo depende dos seguintes recursos:

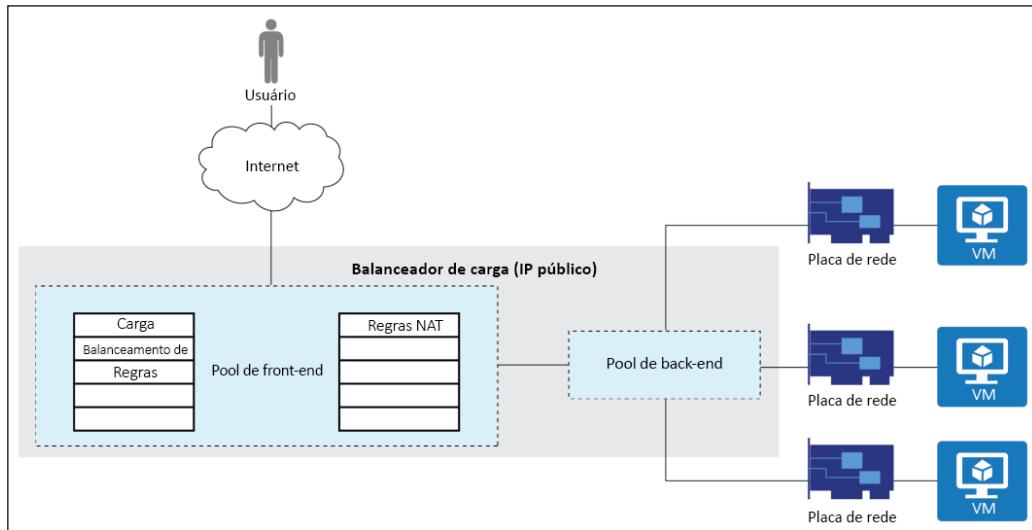
- Um endereço IP de origem
- Um endereço IP de destino
- Um número da porta de origem
- Um número da porta de destino
- Um tipo de protocolo – TCP ou HTTP

Balanceamento de carga público

Em uma configuração de balanceamento de carga público, os平衡adores de carga recebem um endereço IP público. A atribuição de um endereço IP público garante que o balanceador de carga possa aceitar solicitações da Internet. Sem um endereço IP público, não é possível acessar um recurso da Internet. O balanceador de carga pode ser configurado com regras de balanceamento de carga. As regras de balanceamento de carga funcionam no nível da porta. Com isso, as portas de origem e destino podem ser mapeadas de modo que, sempre que um balanceador de carga receber uma solicitação para a porta de origem, a solicitação seja encaminhada a uma VM de um grupo de VMs conectadas ao balanceador de carga na porta de destino. Isso é mostrado no diagrama a seguir:



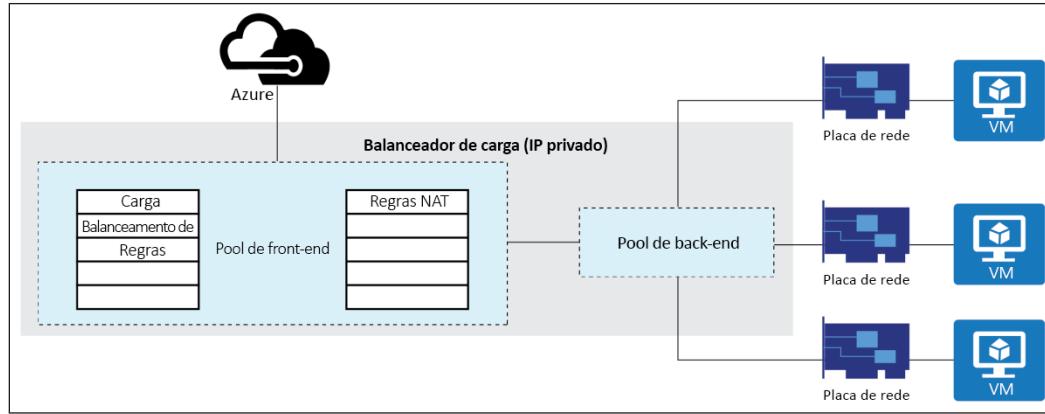
Mas como tudo isso funciona? Como um endereço IP público é atribuído a um balanceador de carga? O que o balanceador de carga contém? Como ele é configurado com as regras do balanceador de carga? Como o balanceador de carga envia solicitações para as VMs? Como a VM sabe que está conectada ao balanceador de carga? As respostas para todas essas perguntas podem ser vistas no diagrama a seguir:



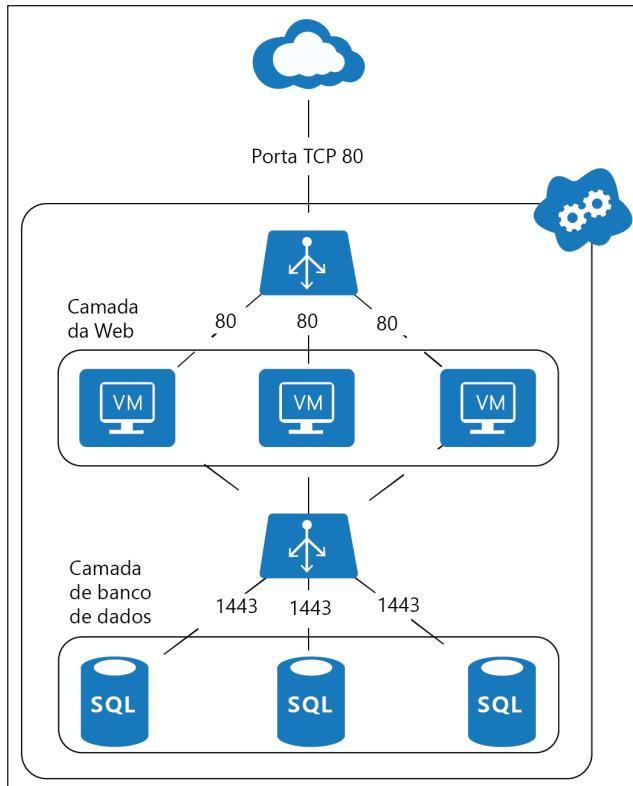
Nessa configuração, o balanceador de carga recebe um endereço IP público. O balanceador de carga é acessível pela Internet e pode aceitar solicitações de clientes. O balanceador de carga pode ser configurado com regras de平衡amento de carga e Conversão de endereço de rede (NAT). As regras de平衡amento de carga e NAT fazem parte da configuração de front-end. A configuração de front-end envia as solicitações de clientes para um dos endereços IP disponíveis no pool de back-end. Esses endereços IP são atribuídos à interface de rede que está associada às VMs.

Balanceamento de carga interno

O diagrama a seguir mostra o funcionamento de um balanceador de carga interno. Você pode ver que a solicitação vem de recursos do próprio Azure, pois não é acessível na Internet. Nessa configuração, o balanceador de carga recebe um endereço IP privado. O balanceador de carga só é acessível na rede virtual à qual está conectado. Ele não pode ser acessado pela Internet. O restante da configuração é semelhante à de um balanceador de carga público. O balanceador de carga pode ser configurado com regras de平衡amento de carga e NAT:



O diagrama a seguir mostra como vários平衡adores de carga podem ser implantados para criar soluções. Desse modo, há um平衡ador de carga público que aceita solicitações de clientes e um平衡ador de carga interno para a camada de banco de dados. As VMs da camada de banco de dados não são acessíveis pela Internet, apenas pelo平衡ador de carga na porta 1433:



Encaminhamento de porta

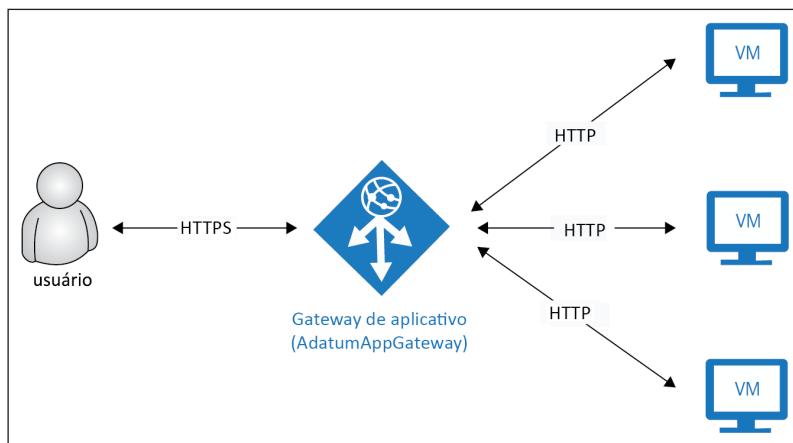
Às vezes, é necessário que uma solicitação seja sempre redirecionada para uma VM. O balanceador de carga do Azure nos ajuda a fazer isso com as regras de NAT. As regras de NAT são avaliadas depois que as regras de平衡amento de carga são avaliadas e não são consideradas satisfatórias. As regras de NAT são avaliadas para cada solicitação de entrada e, depois de encontrá-las, a solicitação é encaminhada para essa VM por meio de um pool de back-end. Lembre-se de que uma VM não pode registrar a mesma porta para o encaminhamento de porta usando regras de NAT e de平衡amento de carga.

Gateway de Aplicativo do Azure

O balanceador de carga do Azure nos ajuda a habilitar soluções no nível da infraestrutura. No entanto, às vezes serviços e recursos avançados são obrigatórios com um balanceador de carga. Esses serviços avançados incluem terminação SSL, sessões fixas e segurança avançada, entre outros. Um Gateway de Aplicativo do Azure fornece esses recursos adicionais. Um Gateway de Aplicativo do Azure é um balanceador de carga de nível 7 que funciona com o aplicativo e a carga de sessão em uma pilha TCP OSI.

Os gateways de aplicativo têm mais informações do que os平衡adores de carga do Azure para tomar decisões sobre o roteamento de solicitações e o平衡amento de carga entre servidores. Os gateways de aplicativo são gerenciados pelo Azure e altamente disponíveis.

Um gateway de aplicativo situa-se entre os usuários e as VMs, como mostrado no diagrama a seguir:



Os gateways de aplicativo são um serviço gerenciado. Eles usam o **Application Request Routing (ARR)** para rotear solicitações para diferentes serviços e pontos de extremidade. A criação de um gateway de aplicativo requer um endereço IP privado ou público. Então, o gateway de aplicativo roteia o tráfego HTTP/HTTPS para pontos de extremidade configurados.

Um gateway de aplicativo é semelhante a um balanceador de carga do Azure de uma perspectiva de configuração, com componentes e recursos adicionais. O gateway de aplicativo pode ser configurado com um endereço IP de front-end, certificado, configuração de porta, pool de back-end, afinidade de sessão e informações de protocolo.

Gerenciador de Tráfego do Azure

Depois de entender bem o balanceador de carga do Azure e o gateway de aplicativo, é hora de entrar nos detalhes do Gerenciador de Tráfego. Os平衡adores de carga e os gateways de aplicativo do Azure são recursos muito necessários para a alta disponibilidade em um data center e uma região. No entanto, para obter a alta disponibilidade entre regiões e data centers, há a necessidade de outro recurso e o Gerenciador de Tráfego nos ajuda nisso.

O Gerenciador de Tráfego nos ajuda a criar soluções altamente disponíveis que abrangem várias localidades geográficas, regiões e data centers. O Gerenciador de Tráfego não é semelhante aos balanceadores de carga. Ele usa o **Serviço de Nomes de Domínio (DNS)** para redirecionar solicitações a um ponto de extremidade apropriado, determinado por sua integridade e configuração. O Gerenciador de Tráfego não é um proxy nem um gateway e não vê o tráfego que passa entre o cliente e o serviço. Ele simplesmente redireciona as solicitações com base nos pontos de extremidade mais adequados.

O Gerenciador de Tráfego do Azure permite controlar a distribuição de tráfego pelos pontos de extremidade do aplicativo. Um ponto de extremidade é qualquer serviço voltada para a Internet hospedado dentro ou fora do Azure.

Pontos de extremidade são URLs públicos que podem ser acessados pela Internet. Os aplicativos são provisionados em várias localidades geográficas e regiões do Azure. Os aplicativos implantados em cada região têm um ponto de extremidade exclusivo chamado **DNS CNAME**. Esses pontos de extremidade são mapeados para o ponto de extremidade do Gerenciador de Tráfego. Quando um Gerenciador de Tráfego é provisionado, ele obtém um ponto de extremidade por padrão com uma extensão de URL `.trafficmanager.net`.

Quando uma solicitação chega ao URL do Gerenciador de Tráfego, ele encontra o ponto de extremidade mais apropriado na lista e redireciona a solicitação para ele. Resumindo, o Gerenciador de Tráfego do Azure age como um DNS global para identificar a região que atenderá à solicitação.

Mas, como o Gerenciador de Tráfego sabe quais pontos de extremidade usar e para qual deles a solicitação do cliente deve ser redirecionada? Há dois aspectos que o Gerenciador de Tráfego implementa para determinar o ponto de extremidade e a região mais apropriados.

Primeiro, o Gerenciador de Tráfego monitora ativamente a integridade de todos os pontos de extremidade. Ele pode monitorar a integridade de VMs, serviços de nuvem e de aplicativo. Se ele determinar que a integridade de um aplicativo implantado em uma região não é adequada para redirecionar o tráfego, redirecionará as solicitações para um ponto de extremidade íntegro.

Em segundo lugar, o Gerenciador de Tráfego pode ser configurado com informações de roteamento. Há seis métodos de roteamento de tráfego disponíveis no Gerenciador de Tráfego:

- **Prioridade:** deve ser usado quando todo o tráfego deve ir para um ponto de extremidade padrão, com backups disponíveis caso os pontos de extremidade principais não estejam disponíveis.
- **Ponderado:** deve ser usado para distribuir uniformemente o tráfego entre os pontos de extremidade ou de acordo com os pesos definidos.
- **Performance:** deve ser usado para pontos de extremidade em diferentes regiões, e os usuários devem ser redirecionados ao ponto de extremidade mais próximo com base em sua localização. Isso causa um impacto direto na latência da rede.
- **Geográfico:** deve ser usado para redirecionar os usuários de uma localização geográfica específica para um ponto de extremidade (isto é, Azure, externo ou aninhado) disponível nessa localização geográfica ou o mais próximo possível dela. Os exemplos incluem o cumprimento de requisitos de soberania de dados, localização de conteúdo e experiência do usuário e medição do tráfego de diferentes regiões.
- **Sub-rede:** é um novo método de roteamento adicionado. Ele ajuda a fornecer aos clientes pontos de extremidade diferentes com base em seus endereços IP. Nesse método, um intervalo de endereços IP é atribuído a cada ponto de extremidade. Esses intervalos de endereços IP são mapeados para o endereço IP do cliente para determinar um ponto de extremidade de retorno apropriado. Com esse método de roteamento, é possível fornecer conteúdo diferente para pessoas diferentes com base no endereço IP de origem delas.
- **Multivalor:** é também um novo método adicionado ao Azure. Nele, vários pontos de extremidade são retornados para o cliente, e qualquer um deles pode ser usado. Isso garante que, se um ponto de extremidade não estiver íntegro, outros poderão ser usados. Isso ajuda a aumentar a disponibilidade geral da solução.

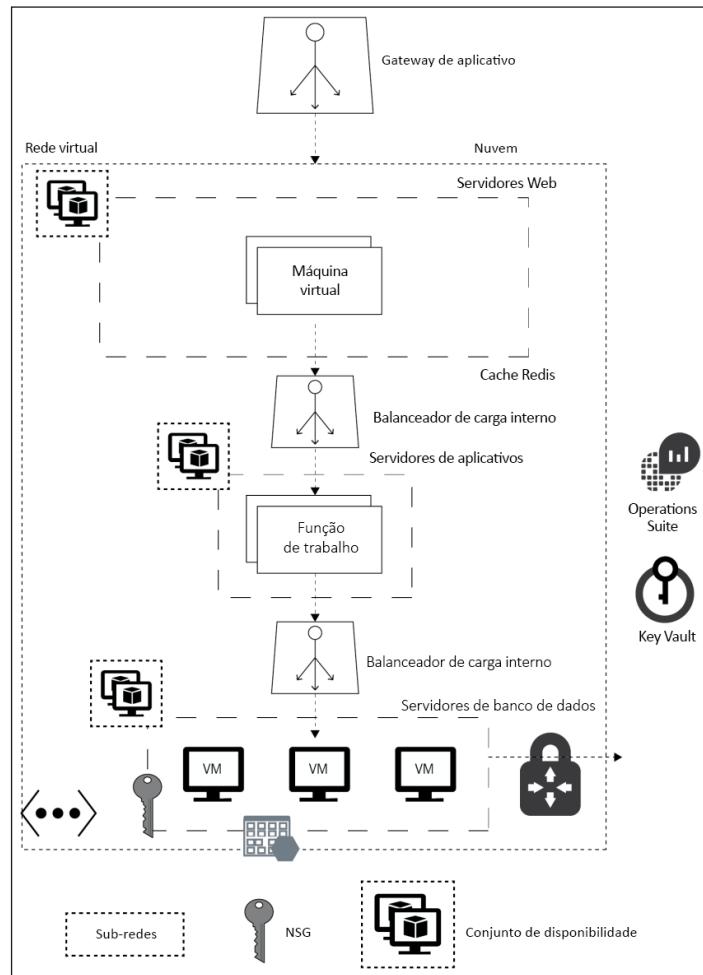
É importante observar que, depois que o Gerenciador de Tráfego determina um ponto de extremidade íntegro válido, os clientes se conectam diretamente ao aplicativo.

Considerações sobre arquitetura para alta disponibilidade

O Azure fornece alta disponibilidade para vários meios e em vários níveis. A alta disponibilidade pode estar no nível do data center, da região ou de todo o Azure. Nesta seção, explicaremos algumas das arquiteturas de alta disponibilidade.

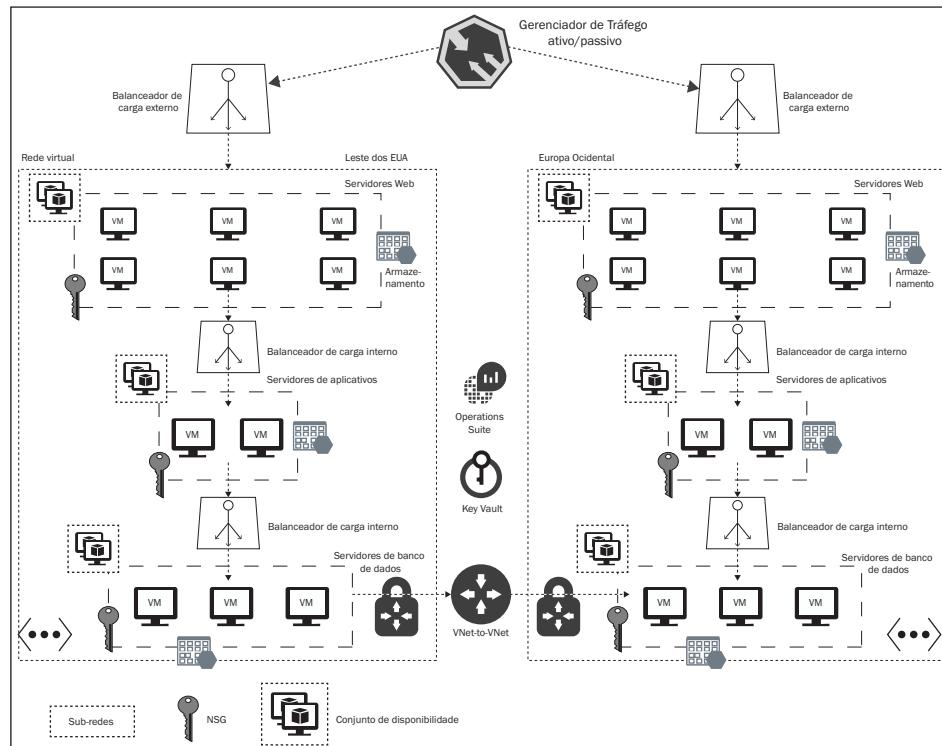
Alta disponibilidade em regiões do Azure

A arquitetura mostrada no diagrama a seguir mostra uma implantação de alta disponibilidade em uma única região do Azure. A alta disponibilidade é desenvolvida no nível de recurso individual. Nessa arquitetura, há várias VMs em cada camada conectada por meio de um gateway de aplicativo ou balanceador de carga e elas fazem parte de um conjunto de disponibilidade. Cada nível é associado a um conjunto de disponibilidade. Essas VMs são colocadas em domínios de falha e atualização separados. Enquanto os servidores Web são conectados a gateways de aplicativo, as camadas restantes, como aplicativo e banco de dados, têm平衡adores de carga internos:



Alta disponibilidade entre regiões do Azure

Essa arquitetura mostra implantações semelhantes em duas regiões diferentes do Azure. Conforme mostrado no diagrama a seguir, ambas as regiões têm os mesmos recursos implantados. A alta disponibilidade é desenvolvida no nível de recurso individual nessas regiões. Há várias VMs em cada camada, conectadas por meio do balanceador de carga, e elas fazem parte do conjunto de disponibilidade. Essas VMs são colocadas em domínios de falha e atualização separados. Enquanto os servidores Web são conectados a平衡adores de carga externos, as camadas restantes, como aplicativo e banco de dados, têm平衡adores de carga internos. Lembre-se de que os平衡adores de carga de aplicativo podem ser usados para camadas de servidores Web e de aplicativo, em vez de平衡adores de carga do Azure, se houver necessidade de serviços avançados, como afinidade de sessão, terminação SSL, segurança avançada usando um **firewall do aplicativo Web (WAF)** e roteamento baseado em caminho. Os bancos de dados em ambas as regiões são conectados entre si usando emparelhamento de redes virtuais e gateways. Isso é útil na configuração de envio de log, do SQL Server AlwaysOn e de outras técnicas de sincronização de dados. Os pontos de extremidade de平衡adores de carga de ambas as regiões são usados para configurar pontos de extremidade do Gerenciador de Tráfego; o tráfego é roteado com base no método de balanceamento de carga prioritário. O Gerenciador de Tráfego ajuda a rotear todas as solicitações para a região do leste dos EUA e, após o failover, para a Europa Ocidental em caso de indisponibilidade da primeira região:



Práticas recomendadas

Esta seção descreve as práticas recomendadas de alta disponibilidade. Elas foram categorizadas em aplicativo, implantação, gerenciamento de dados e monitoramento.

Alta disponibilidade de aplicativos

Um aplicativo deve ser criado com alta disponibilidade como uma das preocupações mais importantes de arquitetura. Algumas das práticas de alta disponibilidade importantes relacionadas ao aplicativo são descritas aqui:

- Um aplicativo deve implementar o tratamento de exceções apropriado para recuperar e informar adequadamente os interessados sobre a exceção.
- Um aplicativo deve tentar executar a mesma operação novamente em um intervalo fixo e um determinado número de vezes antes de sair em caso de erro ou exceção.
- Um aplicativo deve ter o recurso interno de tempo limite para decidir se uma exceção não pode ser recuperada.
- A manutenção de logs e a gravação de logs para todos os erros, exceções e execuções devem ser adotadas no aplicativo.
- Os aplicativos devem ter um perfil criado para que encontrem suas necessidades de recursos reais em termos de computação, memória e largura de banda de rede para um número diferente de usuários.

Consulte <https://docs.microsoft.com/en-us/azure/architecture/checklist/availability> para saber mais sobre aplicativos e outras práticas recomendadas de alta disponibilidade.

Implantação

Uma estratégia de implantação, em grande escala, afeta a disponibilidade do aplicativo e o ambiente geral. Veja a seguir várias estratégias que você pode implantar:

- Implantar várias instâncias de recursos do Azure, incluindo várias instâncias de VMs, serviços de nuvem e outros recursos.
- Implantar VMs em conjuntos de disponibilidade ou zonas de disponibilidade (lembre-se de que elas não podem ser usadas juntas).
- Implantar várias instâncias de VMs em várias regiões.
- Criar vários ambientes e mantenha pelo menos um deles no modo de espera.

Gerenciamento de dados

Algumas das práticas recomendadas importantes relacionadas a dados para alta disponibilidade incluem:

- Se possível, armazenar dados em serviços fornecidos pelo Azure, como Azure SQL, Cosmos DB e armazenamento de tabelas.
- Usar contas de armazenamento que são baseadas no tipo de redundância geográfica.
- Garantir que os dados sejam replicados para várias regiões, e não apenas na zona ou no data center.
- Fazer backups periódicos e testes de restauração com frequência.
- Ao armazenar dados em VMs, garantir que haja várias VMs e que elas estejam em conjuntos de disponibilidade ou em zonas de disponibilidade.
- Usar chaves e segredos para dados armazenados no Azure Key Vault.

Monitoramento

Algumas das práticas recomendadas importantes relacionadas a monitoramento para alta disponibilidade são:

- Usar o serviço de análise de log para capturar logs do ambiente e habilitar a auditoria.
- Usar insights de aplicativo para capturar informações de telemetria do aplicativo personalizado e do ambiente relacionadas a computação, armazenamento e redes, além de outras informações de log.
- Garantir que os alertas sejam configurados no OMS (análise de log) para problemas relacionados à disponibilidade do ambiente e do aplicativo.

Escalabilidade

A execução de aplicativos e sistemas que estão disponíveis para os usuários para consumo é importante para arquitetos de qualquer aplicativo sério. No entanto, há outro recurso de aplicativo igualmente importante que é uma das principais prioridades para os arquitetos: a escalabilidade do aplicativo.

Imagine uma situação em que um aplicativo é implantado e obtém excelente performance e disponibilidade com poucos usuários, mas tanto a disponibilidade quanto a performance diminuem à medida que o número de usuários começa a aumentar. Há momentos em que um aplicativo sob carga normal tem boa performance, mas essa performance diminui com o aumento do número de usuários. Isso pode acontecer quando há um aumento repentino no número de usuários e o ambiente não foi criado para um número tão grande de usuários.

Para acomodar esses picos no número de usuários, você pode provisionar o hardware e a largura de banda para gerenciar esses picos. O desafio com isso é que a capacidade adicional não é usada na maior parte do ano e, portanto, não fornece retorno sobre o investimento. Ela é provisionada para uso apenas durante a temporada de férias ou vendas. Espero que até o momento você esteja se familiarizando com os problemas que os arquitetos estão tentando resolver. Todos esses problemas estão relacionados ao dimensionamento da capacidade e à escalabilidade de um aplicativo. O foco deste capítulo é entender a escalabilidade como uma preocupação de arquitetura e conferir os serviços fornecidos pelo Azure para implementar a escalabilidade.

Planejamento e dimensionamento de capacidade são algumas das principais prioridades dos arquitetos para seus aplicativos e serviços. Os arquitetos devem encontrar um equilíbrio entre a compra e o provisionamento de recursos a mais ou a menos. Se tiver menos recursos, você não conseguirá atender a todos os usuários, e eles recorrerão aos concorrentes. Por outro lado, ter mais recursos pode prejudicar seu orçamento e retorno sobre o investimento, pois a maioria dos recursos permanece não utilizada na maior parte do tempo. Além disso, o problema aumenta com um nível variado de demanda em diferentes momentos. É quase impossível prever o número de usuários para o aplicativo para todos os momentos. No entanto, é possível encontrar um número aproximado usando informações anteriores e monitoramento contínuo.

A escalabilidade pode ser definida da seguinte maneira:

"Escalabilidade é a capacidade de um sistema, rede ou processo de lidar com uma quantidade crescente de trabalho, ou seu potencial de ser ampliado para acomodar esse crescimento. Por exemplo, um sistema é considerado escalável quando é capaz de aumentar seu rendimento total sob uma carga maior quando recursos (normalmente hardware) são adicionados."

A escalabilidade refere-se à capacidade de gerenciar um número crescente de usuários e de fornecer a eles o mesmo nível de performance quando há menos usuários em implantação de aplicativos, processos e tecnologia. A escalabilidade pode se referir ao atendimento de mais solicitações sem reduzir a performance ou pode se referir à capacidade de lidar com um trabalho maior e mais demorado sem qualquer perda de performance nos dois casos.

Os exercícios de planejamento e dimensionamento de capacidade devem ser realizados pelos arquitetos no início do projeto e durante a fase de planejamento para oferecer escalabilidade para aplicativos.

Alguns aplicativos têm padrões de demanda estáveis, embora seja difícil prever outros. Requisitos de escalabilidade são conhecidos para aplicativos de demanda estável, embora esse seja um processo mais envolvido para aplicativos de demanda variável. O dimensionamento automático, um conceito que veremos na próxima seção, deve ser usado para os aplicativos cujas demandas não podem ser previstas.

Escalabilidade x performance

É muito fácil ficar confuso entre as preocupações de arquitetura de escalabilidade e performance, pois a escalabilidade tem a ver com garantir que, independentemente do número de usuários que consomem o aplicativo, todos recebam o mesmo nível predeterminado de performance.

A performance está relacionada aos recursos do aplicativo que garantem que ela atenda a taxas de transferência e tempos de resposta e predefinidos. A escalabilidade refere-se a ter provisões para mais recursos quando necessário com a finalidade de acomodar mais usuários sem sacrificar a performance.

É melhor entender isso usando uma analogia: a velocidade de um trem determina a performance de um sistema ferroviário. No entanto, permitir que mais trens funcionem em paralelo a uma velocidade igual ou maior demonstra a escalabilidade do sistema ferroviário.

Escalabilidade do Azure

Nesta seção, veremos as funcionalidades e os recursos fornecidos pelo Azure para tornar os aplicativos altamente escaláveis. Antes de entrarmos em detalhes de arquitetura e configuração, é importante compreender os conceitos relacionados à alta disponibilidade do Azure.

Conceitos

Os conceitos fundamentais fornecidos pelo Azure para obter alta disponibilidade são:

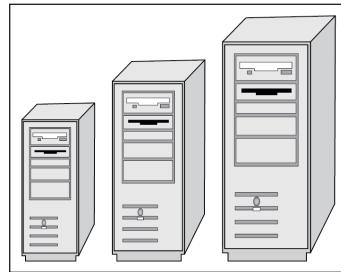
- Escala
- Expansão e redução verticais
- Expansão e redução horizontais
- Dimensionamento automático
- Atualizações ininterruptas

Escala

Escala refere-se a qualquer transformação que aumenta ou diminui as unidades de recursos usados para atender às solicitações dos usuários. A escala pode ser manual ou automática. A escala manual exige que um administrador inicie manualmente o processo de escalonamento, enquanto o dimensionamento automático refere-se a um aumento ou diminuição automática de recursos com base nos eventos disponíveis no ambiente e no ecossistema, como disponibilidade da memória e da CPU. A escala pode ser expansão ou redução vertical ou horizontal, o que será explicado a seguir nesta seção.

Escala vertical

A escala vertical de uma VM ou de um serviço refere-se à adição de recursos para servidores existentes, como CPU, memória e discos. O objetivo é aumentar a capacidade do hardware e dos recursos físicos existentes:

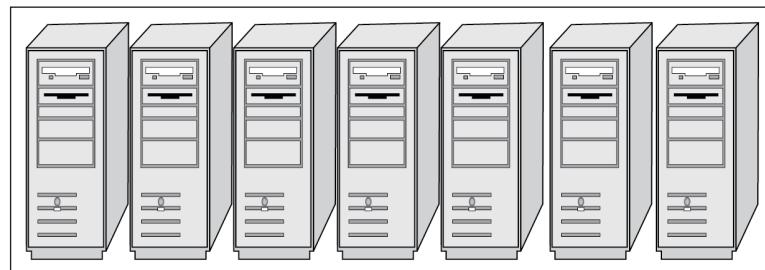


Redução vertical

A redução vertical de uma VM ou de um serviço refere-se à remoção de recursos existentes de servidores existentes, como CPU, memória e discos. O objetivo é reduzir a capacidade do hardware e dos recursos físicos e virtuais existentes.

Escala horizontal

A escala horizontal refere-se ao processo de adição de hardware como servidores e capacidade adicionais. Isso normalmente envolve adicionar novos servidores, atribuir endereços IP a eles, implantar aplicativos neles e torná-los parte dos平衡adores de carga existentes, de modo que o tráfego possa ser encaminhado para eles. A escala horizontal também pode ser manual ou automática. No entanto, para melhores resultados, a automação deve ser usada:



Redução horizontal

A redução horizontal refere-se ao processo de remoção do hardware existente em termos de capacidade e servidores existentes. Isso normalmente envolve remover servidores existentes, desalocar endereços IP e removê-los da configuração existente de平衡adores de carga de modo que o tráfego não possa ser roteado para eles. Assim como a escala horizontal, a redução horizontal pode ser automática ou manual.

Dimensionamento automático

O dimensionamento automático refere-se ao processo de expandir/reduzir vertical ou horizontalmente de forma dinâmica com base na demanda do aplicativo e ocorre por meio da automação. O dimensionamento automático é útil porque garante que a implantação sempre consista em um número correto e ideal de instâncias de servidor. O dimensionamento automático ajuda a criar aplicativos tolerantes a falhas. Isso não só ajuda na escalabilidade, mas também torna as aplicações altamente disponíveis. Por fim, ela proporciona o melhor gerenciamento de custos. O dimensionamento automático ajuda a ter a configuração ideal para instâncias de servidor com base na demanda. Ele ajuda a não provisionar servidores subutilizados e remove servidores que não são mais necessários após a escala horizontal.

Escalabilidade da PaaS

O Azure fornece serviços de aplicativo para hospedar aplicativos gerenciados. Os serviços de aplicativo são uma oferta de PaaS do Azure. Eles fornecem serviços para a Web e plataformas móveis. Por trás das plataformas Web e móvel está uma infraestrutura gerenciada pelo Azure em nome de seus usuários. Os usuários não veem nem gerenciam a infraestrutura. No entanto, eles podem estender a plataforma e implantar seus aplicativos nela. Com isso, os arquitetos e desenvolvedores podem se concentrar em seus problemas de negócios em vez de se preocupar com a plataforma básica e o provisionamento, a configuração e a solução de problemas da infraestrutura. Os desenvolvedores têm a flexibilidade de escolher qualquer linguagem, sistema operacional e estrutura para desenvolver seus aplicativos.

Os serviços de aplicativo fornecem vários planos e, com base nos planos escolhidos, vários recursos de escalabilidade são disponibilizados. Os serviços de aplicativo fornecem os cinco planos a seguir:

- **Gratuito:** usa uma infraestrutura compartilhada. Isso significa que vários aplicativos serão implantados na mesma infraestrutura do mesmo ou de vários locatários. Ele fornece 1 GB de armazenamento gratuito. No entanto, não há capacidade de escala disponível nesse plano.
- **Compartilhado:** também usa uma infraestrutura compartilhada e fornece 1 GB de armazenamento gratuito. Além disso, domínios personalizados também são fornecidos como um recurso extra. No entanto, não há capacidade de escala disponível nesse plano.
- **Básico:** tem três **unidades de manutenção (SKUs)** – B1, B2 e B3. Elas têm um número maior de unidades de recursos disponíveis em termos de CPU e memória. Resumindo, essas unidades fornecem uma configuração melhor das VMs com apoio para esses serviços. Além disso, fornece armazenamento, domínios personalizados e suporte para SSL. O plano básico oferece recursos básicos para escala manual. Não há dimensionamento automático disponível nesse plano. No máximo três instâncias podem ser usadas para a escala horizontal do aplicativo.
- **Padrão:** também tem três SKUs diferentes: S1, S2 e S3. Elas têm um número maior de unidades de recursos disponíveis em termos de CPU e memória. Resumindo, essas unidades fornecem uma configuração melhor das VMs com apoio para esses serviços. Além disso, elas fornecem armazenamento, domínios personalizados e suporte para SSL semelhantes ao plano básico. Esse plano também fornece um gerenciador de tráfego, slots de preparo e um backup diário como recurso adicional acima do plano básico. O plano padrão oferece recursos para dimensionamento automático. No máximo 10 instâncias podem ser usadas para a escala horizontal do aplicativo.
- **Premium:** também tem três SKUs diferentes: P1, P2 e P3. Elas têm um número maior de unidades de recursos disponíveis em termos de CPU e memória. Resumindo, essas unidades fornecem uma configuração melhor das VMs com apoio para esses serviços. Além disso, elas fornecem armazenamento, domínios personalizados e suporte para SSL semelhantes ao plano básico. Esse plano também fornece um gerenciador de tráfego, slots de preparo e 50 backups diários como recurso adicional acima do plano básico. O plano padrão oferece recursos para dimensionamento automático. No máximo 20 instâncias podem ser usadas para a escala horizontal do aplicativo.

PaaS – Expansão e redução verticais

A expansão e a redução verticais de serviços hospedados em serviços de aplicativo é bastante simples. O aplicativo Azure oferece itens de menu para escala vertical, o que abre uma nova folha com todos os planos e suas SKUs listadas. A escolha de um plano e um SKU expandirá ou reduzirá o serviço, conforme mostrado na captura de tela a seguir:

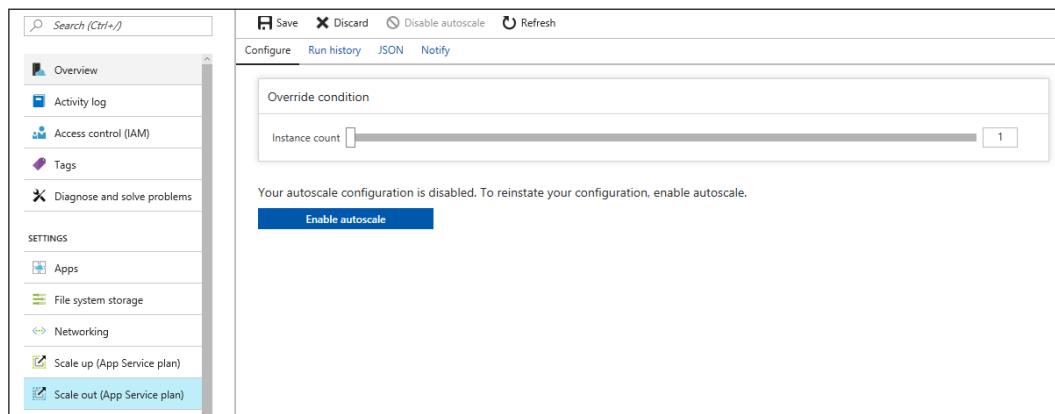
P1 Premium		P2 Premium		P3 Premium	
1	Core	2	Core	4	Core
1.75	GB RAM	3.5	GB RAM	7	GB RAM
250 GB Storage	Custom domains / SSL SNI Incl & IP SSL Support	250 GB Storage	Custom domains / SSL SNI Incl & IP SSL Support	250 GB Storage	Custom domains / SSL SNI Incl & IP SSL Support
Up to 20 instance(s) * Subject to availability	Up to 20 instance(s) * Subject to availability	Up to 20 instance(s) * Subject to availability	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					
50 times daily Backup					
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	Custom domains / SSL SNI Incl & IP SSL Support	50 GB Storage	Custom domains / SSL SNI Incl & IP SSL Support	50 GB Storage	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	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	Daily Backup	Daily Backup	Daily Backup
5 slots Web app staging	5 slots Web app staging	5 slots Web app staging	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	Traffic Manager Geo availability	Traffic Manager Geo availability	Traffic Manager Geo availability
Select					

PaaS – Expansão e redução horizontais

A expansão e a redução horizontais de serviços hospedados em serviços de aplicativo também é bastante simples. O aplicativo Azure oferece itens de menu para expansão, o que abre uma nova folha com as opções de configuração de escala.

Por padrão, o dimensionamento automático é desativado para nos planos Premium e padrão. Ela pode ser ativada usando o item de menu **Scale Out** e clicando no botão **Enable autoscale**, conforme mostrado na captura de tela a seguir:

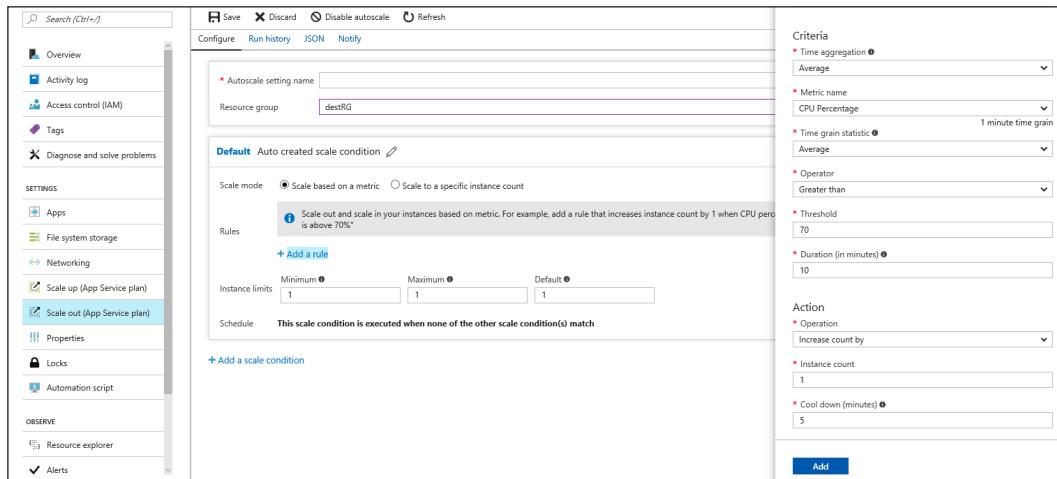


O dimensionamento manual não precisa de configuração, mas o dimensionamento automático ajuda na configuração por meio das seguintes propriedades de escala:

- **Modo de dimensionamento:** baseia-se em uma métrica de performance, como CPU ou uso de memória. Os usuários também podem especificar a contagem de instâncias para dimensionamento.
- **Quando escalar:** várias regras podem ser adicionadas que determinar quando expandir ou reduzir horizontalmente. Cada regra pode determinar critérios como consumo de CPU ou memória, se é preciso aumentar ou diminuir instâncias, quantas instâncias aumentar ou diminuir por vez. É preciso configurar pelo menos uma regra para a escala horizontal e uma regra para a redução horizontal. As definições de limites ajudam a definir os limites superior e inferior que devem acionar o dimensionamento automático – aumentando ou diminuindo o número de instâncias.

Disponibilidade e escalabilidade da solução do Azure

- **Como escalar:** especifica quantas instâncias criar ou remover em cada expansão ou redução horizontal:



É um recurso muito bom para ser habilitado em qualquer implantação. No entanto, os leitores devem habilitar as escalas vertical e horizontal em conjunto para garantir que o ambiente volte à capacidade normal após a expansão.

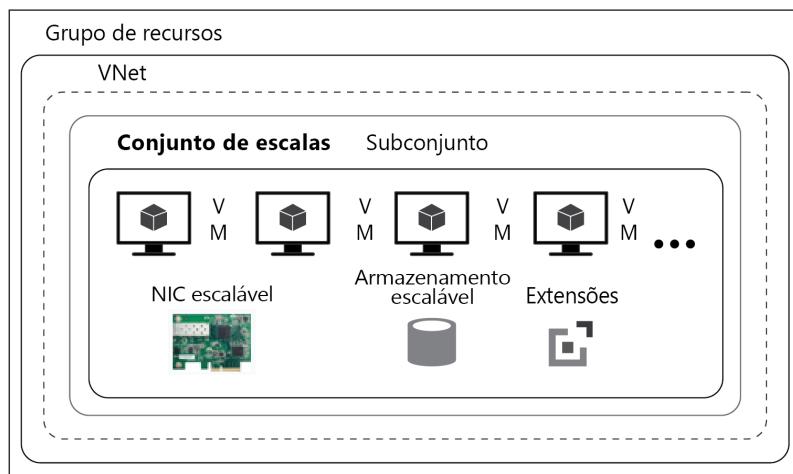
Escalabilidade da IaaS

Há usuários que querem ter controle total sobre a infraestrutura de base, a plataforma e o aplicativo. Eles preferem consumir soluções de IaaS, em vez de soluções de PaaS. Para esses clientes, quando eles criam VMs, eles também são responsáveis por dimensionar e escalar a capacidade. Não há nenhuma configuração pronta para o uso para dimensionamento manual ou automático de VMs. Esses clientes precisarão escrever os próprios scripts de automação, gatilhos e regras para obter o dimensionamento automático. Com as VMs também vem a responsabilidade de mantê-las. A aplicação de patches, atualização e upgrade de VMs é da responsabilidade dos proprietários. Os arquitetos devem pensar tanto na manutenção planejada quanto na não planejada. Como essas VMs devem receber patches, a ordem, o agrupamento e outros fatores devem ser ponderados para garantir que a escalabilidade e a disponibilidade de um aplicativo não sejam comprometidas. Para ajudar a aliviar esses problemas, o Azure fornece **conjuntos de escalas de VM (VMSS)** como uma solução.

VMSS

O VMSS é um recurso de computação do Azure que você pode usar para implantar e gerenciar um conjunto de VMs idênticas. Com todas as VMs configuradas da mesma maneira, os conjuntos de dimensionamentos são projetados para oferecer suporte ao dimensionamento automático real, e nenhum pré-provisionamento de VMs é necessário. Isso ajuda no provisionamento de várias VMs idênticas conectadas umas às outras por meio de uma rede virtual e sub-rede.

O VMSS consiste em várias VMs, mas é gerenciado no nível dos VMSS. Todas as VMs fazem parte dessa unidade e todas as alterações feitas são aplicadas à unidade que, por sua vez, as aplica a VMs que usam um algoritmo predeterminado:



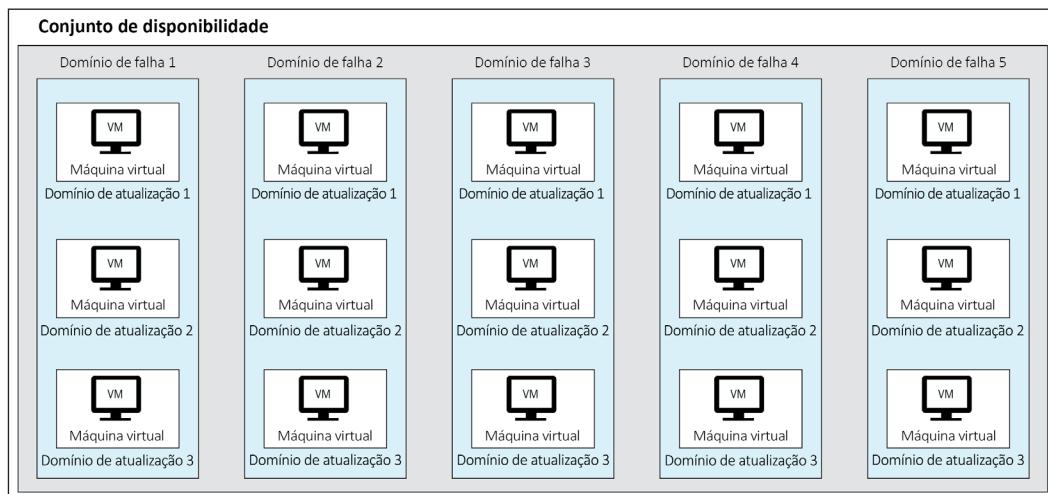
Isso permite que essas VMs tenham a carga balanceada usando o平衡ador de carga do Azure ou os gateways de aplicativo. Todas as VMs podem ser sistemas operacionais Windows ou Linux. Elas podem executar scripts automatizados usando uma extensão do PowerShell e podem ser gerenciadas centralmente usando a configuração de estado desejada. Elas podem ser monitoradas como uma unidade ou individualmente usando a análise de log.

O VMSS pode ser provisionado por meio do portal do Azure, da **interface de linha de comando** do Azure (CLI), dos modelos do Azure Resource Manager, das APIs REST e de cmdlets do PowerShell. É possível invocar as APIs REST e a CLI do Azure de qualquer plataforma, ambiente e sistema operacional e em qualquer linguagem.

Muitos serviços do Azure já usam VMSS como arquitetura subjacente. Entre eles estão o Lote do Azure, Azure Service Fabric e Serviços de Contêiner do Azure. Os Serviços de Contêiner do Azure, por sua vez, provisionam o Kubernetes e o DC/OS no VMSS.

Arquitetura VMSS

O VMSS permite a criação de até 1.000 VMs em um conjunto de dimensionamento ao usar uma plataforma, imagens e 100 VMs, se estiver usando uma imagem personalizada. Se a quantidade de VMs for inferior a 100 em um conjunto de dimensionamento, elas serão colocadas em um único conjunto de disponibilidade. No entanto, se o número for superior a 100, vários conjuntos de disponibilidade serão criados (conhecidos como grupos de posicionamento), e as VMs serão distribuídas entre esses conjuntos de disponibilidade. Sabemos desde o Capítulo 1, *Introdução*, que as VMs em um conjunto de disponibilidade são colocadas em domínios de falha e atualização separados. Os conjuntos de disponibilidade relacionados ao VMSS têm cinco domínios de falha e atualização por padrão. O VMSS fornece um modelo que contém informações de metadados para todo o conjunto. Alterar esse modelo e aplicar alterações afeta todas as instâncias de VM. Essas informações de metadados incluem instâncias de VM máximas e mínimas, a SKU e a versão do sistema operacional, o número atual de VMs, domínios de falha e atualização e muito mais. Isso é demonstrado no diagrama a seguir:



Escala VMSS

A escala refere-se a um aumento ou diminuição de recursos de computação e armazenamento. O VMSS é um recurso avançado que torna a escala fácil e eficiente. Ele oferece o dimensionamento automático, que ajuda a expandir ou reduzir verticalmente com base em eventos e dados externos, como uso de CPU e memória. Alguns recursos de escala VMSS são explicados nesta seção.

Escala horizontal x vertical

A escala pode ser horizontal, vertical ou uma combinação de ambas. Escala horizontal é outro nome para a expansão e redução horizontais, enquanto a escala vertical refere-se à expansão e redução verticais.

Capacidade

O VMSS tem uma propriedade de capacidade que determina o número de VMs em um conjunto de dimensionamento. O VMSS pode ser implantado com o zero como o valor dessa propriedade. Isso não criará uma única VM. No entanto, se você provisionar o VMSS fornecendo um número para a propriedade de capacidade, esse número de VMs será criado.

Dimensionamento automático

O dimensionamento automático de VMs no VMSS refere-se à adição ou remoção de instâncias de VM com base nos ambientes configurados para atender às demandas de performance e escalabilidade de um aplicativo. Geralmente, na ausência do VMSS, isso é conseguido por meio de scripts de automação e runbooks.

O VMSS ajuda nesse processo de automação, com o auxílio da configuração. Em vez de escrever scripts, o VMSS pode ser configurado para expansão e redução verticais automatizadas.

O dimensionamento automático consiste em vários componentes integrados para atingir seu objetivo final. O dimensionamento automático monitora continuamente as VMs e coleta dados de telemetria sobre elas. Ela armazena esses dados e os combina e avalia com base em um conjunto de regras para determinar se é preciso acionar o dimensionamento automático. O gatilho poderia ser para uma expansão/redução horizontal ou vertical.

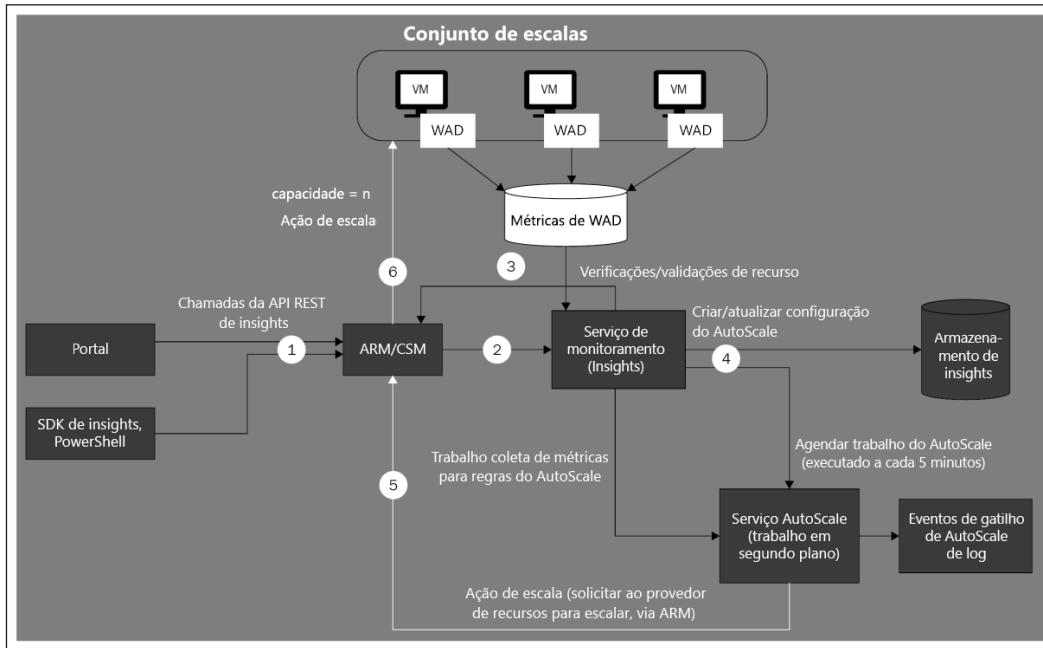
O dimensionamento automático usa logs de diagnóstico para coletar dados de telemetria de VMs. Esses logs são armazenados em contas de armazenamento como métricas de diagnóstico. O dimensionamento automático também usa o serviço de monitoramento de insights, que lê essas métricas e as combina e armazena em sua própria conta de armazenamento.

Os trabalhos em segundo plano com dimensionamento automático são executados continuamente para ler os dados de armazenamento de insights, avaliá-los com base em todas as regras configuradas para dimensionamento automático e executar o processo de dimensionamento automático, se alguma das regras ou uma combinação de regras retornar um resultado positivo. As regras podem levar em consideração as métricas das VMs convidadas e do servidor host.



As regras definidas usando as descrições de propriedade estão disponíveis em <https://docs.microsoft.com/pt-br/azure/virtual-machine-scale-sets/virtual-machine-scale-sets-autoscale-overview>.

A arquitetura de dimensionamento automático é mostrada no diagrama a seguir:



O dimensionamento automático pode ser configurado para cenários mais complexos do que as métricas gerais disponíveis nos ambientes. Por exemplo, o dimensionamento pode ser baseado em qualquer um dos eventos a seguir:

- Dimensionamento em um dia específico
- Dimensionamento em uma agenda recorrente, como fins de semana
- Dimensionamento diferente em dias úteis e fins de semana
- Dimensionamento durante feriados, ou seja, um dos eventos
- Dimensionamento em várias métricas de recursos

Elas podem ser configuradas usando a propriedade de agendamento de recursos de insights que ajudam a registrar regras.

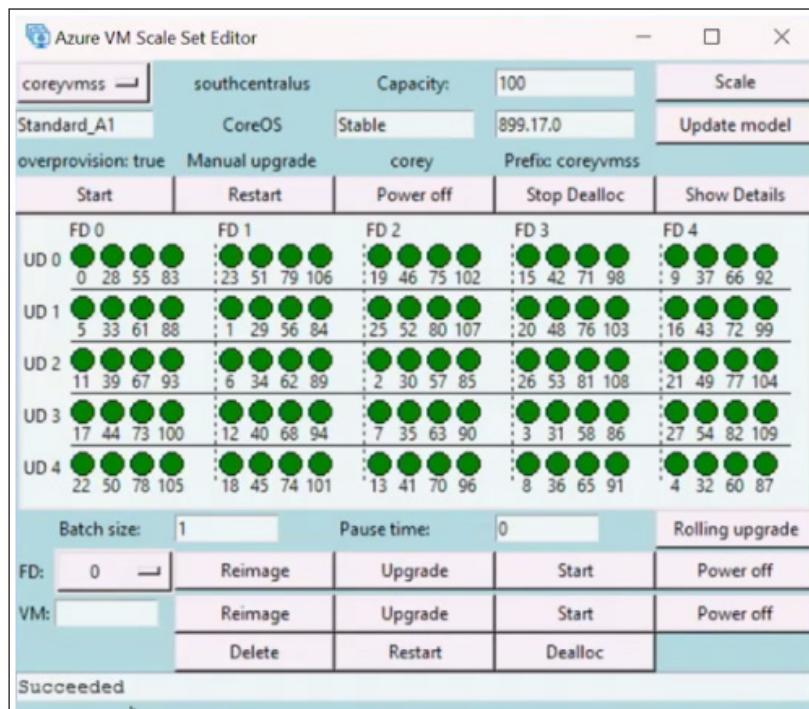
Os arquitetos devem assegurar que pelo menos duas ações, expansão e redução horizontais, sejam configuradas em conjunto. A configuração de expansão e redução horizontais não ajudará a obter os benefícios de escala fornecidos pelo VMSS.

Atualizações e manutenção

Depois que o VMSS e os aplicativos são implantados, eles precisam ser mantidos ativamente. A manutenção planejada deve ser realizada periodicamente para garantir que tanto o ambiente quanto o aplicativo sejam atualizados com os recursos mais recentes e que o ambiente seja atualizado do ponto de vista de segurança e resiliência.

Atualizações podem ser associadas a aplicativos, à instância de VM convidada ou à própria imagem. Os upgrades podem ser bastante complexos, pois devem acontecer sem afetar a disponibilidade, escalabilidade e performance de ambientes e aplicativos. Para garantir que as atualizações possam ocorrer em uma instância de cada vez usando métodos de upgrade sem interrupção, é importante que o VMSS ofereça suporte e recursos para esses cenários avançados.

A equipe do Azure fornece um utilitário para gerenciar atualizações do VMSS. É um utilitário baseado em Python cujo download pode ser feito em <https://github.com/gbowerman/vmssdashboard>. Ele faz chamadas da API REST para o Azure para gerenciar conjuntos de escalas. Esse utilitário pode ser usado para iniciar, parar, atualizar e recriar VMs em um domínio de falha ou grupo de VMs, conforme mostrado na captura de tela a seguir:



Atualizações de aplicativos

As atualizações de aplicativo no VMSS não devem ser executadas manualmente. Elas devem ser executadas como parte do gerenciamento e pipelines de versões que usam a automação. Além disso, a atualização deve ocorrer em uma instância de aplicativo por vez e não deve afetar a disponibilidade geral e a escalabilidade do aplicativo. Ferramentas de gerenciamento de configuração, como a configuração de estado desejada, devem ser implantadas para gerenciar as atualizações de aplicativos. O servidor de pull do DSC pode ser configurado com a versão mais recente da configuração de aplicativos, e eles devem ser aplicados sem interrupção a cada instância.

Atualizações de convidados

As atualizações da VM são de responsabilidade do administrador. O Azure não é responsável por corrigir VMs convidadas. As atualizações de convidados estão no modo de visualização e os usuários devem controlar o aplicativo de patches manualmente ou usar a automação personalizada, como runbooks e scripts. No entanto, os upgrades de patch sem interrupção estão no modo de visualização e podem ser configurados no modelo Azure Resource Manager usando uma política de atualização, conforme mostrado aqui:

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

Atualizações de imagens

O VMSS pode atualizar a versão do sistema operacional sem qualquer tempo de inatividade. As atualizações do sistema operacional envolvem a alteração da versão ou da SKU do sistema operacional ou a alteração do URI de uma imagem personalizada. Atualizar sem tempo de inatividade significa atualizar as VMs uma de cada vez ou em grupos (como um domínio de falha por vez) em vez de todas de uma vez. Ao fazer isso, as VMs que não estiverem sendo atualizadas poderão continuar funcionando.

Práticas recomendadas de dimensionamento fornecido pelo VMSS

Nesta seção, explicaremos algumas das práticas recomendadas que os aplicativos devem implementar para aproveitar as vantagens do recurso de escalabilidade fornecido pelo VMSS.

A preferência pela expansão horizontal

Expandir horizontalmente é a melhor solução de escala comparado à expansão vertical. Expandir ou reduzir verticalmente significa redimensionar instâncias de VM. Quando uma VM é redimensionada, ela geralmente precisa ser reiniciada, o que tem suas próprias desvantagens. Primeiro, há um tempo de inatividade para a máquina. Segundo, se houver usuários ativos conectados ao aplicativo nessa instância, poderá haver indisponibilidade do aplicativo ou até mesmo perda de transações. Expandir horizontalmente não afeta as VMs existentes, mas provisiona máquinas mais recentes e as adiciona ao grupo.

Instâncias bare metal x inativas

O dimensionamento de novas instâncias pode ter duas abordagens amplas: criar a nova instância do zero, ou seja, instalar aplicativos, configurá-los e testá-los ou iniciar as instâncias inativas quando elas forem necessárias devido à pressão de escalabilidade em outros servidores.

Configuração apropriada do número máximo e mínimo de instâncias

Definir dois como o valor mínimo e máximo de contagem de instâncias com a atual contagem de instâncias sendo dois, nenhuma ação de escala pode ocorrer. Deve haver uma margem adequada entre as contagens máxima e mínima de instâncias, que são inclusivas. O dimensionamento automático sempre escala entre esses limites.

Simultaneidade

Os aplicativos são projetados para escalabilidade para focar na simultaneidade. Os aplicativos devem usar padrões assíncronos para garantir que as solicitações do cliente não esperem indefinidamente pela aquisição de recursos, caso eles estejam ocupados atendendo a outras solicitações. A implementação de padrões assíncronos no código garante que os threads não aguardem pelos recursos e que os sistemas esgotem todos os threads disponíveis. Os aplicativos devem implementar o conceito de tempo limite se houver falhas intermitentes esperadas.

Sem estado

Aplicativos e serviços devem ser projetados para não terem estado. A escalabilidade pode se tornar um desafio de resolver com serviços com estado e é muito fácil escalar serviços sem estado. Com o estado vem o requisito de componentes adicionais e implementações, como replicação, repositório centralizado ou descentralizado, manutenção e sessões fixas. Todos eles são obstáculos no caminho para a escalabilidade. Imagine um serviço de manutenção de estado ativo em um servidor local. Não importa o número de solicitações no aplicativo geral ou no servidor individual, as solicitações subsequentes devem ser atendidas pelo mesmo servidor. As solicitações subsequentes não podem ser processadas por outros servidores. Isso torna a implementação de escalabilidade um desafio.

Armazenamento em cache e a Rede de Distribuição de Conteúdo (CDN)

Aplicativos e serviços devem tirar proveito do armazenamento em cache. O armazenamento em cache ajuda a eliminar várias chamadas subsequentes para qualquer banco de dados do sistema de arquivos. Isso ajuda a disponibilizar os recursos para mais solicitações. A CDN é outro mecanismo usado para armazenar em cache arquivos estáticos, como imagens e bibliotecas de JavaScript. Eles estão disponíveis em servidores no mundo todo. Eles também disponibilizam recursos para solicitações de clientes adicionais — isso torna os aplicativos altamente escaláveis.

Design N+1

O **design N + 1** refere-se à criação de redundância na implantação geral para cada componente. Significa planejar alguma redundância mesmo quando ela não é necessária. Isso pode significar VMs adicionais, armazenamento e interfaces de rede.

Resumo

A alta disponibilidade e a escalabilidade são preocupações de arquitetura muito importantes. Quase todos os arquitetos e aplicativos tentam implementar a alta disponibilidade. O Azure é uma plataforma madura que comprehende a necessidade dessas preocupações de arquitetura em aplicativos e fornece recursos para implementá-las em vários níveis. Essas preocupações de arquitetura não são um conceito tardio e devem fazer parte do desenvolvimento do ciclo de vida do aplicativo, começando pela própria fase de planejamento. No próximo capítulo, nós nos aprofundaremos nos conceitos relacionados à segurança e ao monitoramento no Azure.

3

Segurança e monitoramento

A segurança é, sem dúvida, o requisito não funcional mais importante para que os arquitetos implementem. As empresas enfatizam muito a implementação correta de sua estratégia de segurança. Na verdade, a segurança é uma das principais preocupações para quase todos os stakeholders no desenvolvimento, implantação e gerenciamento de um aplicativo. Ela se torna até mais importante quando o mesmo aplicativo é criado para implantação na nuvem.

Os tópicos a seguir serão abordados neste capítulo:

- Considerações
- Monitoramento
- Monitoramento do Azure
- Application Insights
- Log Analytics
- Execução de runbooks em alertas
- Integração ao Power BI

Segurança

A segurança é um elemento importante para qualquer software ou serviço. É necessário criar uma segurança adequada para que o aplicativo só possa ser usado por pessoas que têm permissão para acessá-lo, e os usuários não podem executar operações para as quais não têm permissão. Da maneira semelhante, todo o mecanismo de solicitação-resposta deve ser criado usando métodos que garantam que somente as partes pretendidas entendam as mensagens e que seja fácil detectar se elas foram adulteradas ou não. Pelos motivos a seguir, a segurança no Azure é ainda mais importante. Em primeiro lugar, as organizações que implantam seus aplicativos não estão no controle total do hardware e das redes subjacentes. Em segundo lugar, a segurança precisa ser incorporada a todas as camadas, incluindo hardware, redes, sistemas operacionais, plataformas e aplicativos. Quaisquer omissões ou configurações incorretas podem tornar o aplicativo vulnerável a intrusos.

Proteger um aplicativo significa que entidades desconhecidas e não autorizadas não podem acessá-lo. Também significa que a comunicação com o aplicativo é segura e não adulterada. Isso inclui as seguintes medidas de segurança:

- **Autenticação:** a autenticação verifica a identidade de um usuário e garante que a identidade determinada possa acessar o aplicativo ou serviço. A autenticação é realizada no Azure usando OpenID Connect.
- **Autorização:** a autorização permite e estabelece permissões que uma identidade pode realizar no aplicativo ou serviço. A autorização é realizada no Azure usando OAuth.
- **Confidencialidade:** a confidencialidade garante que a comunicação entre o usuário e o aplicativo permaneça segura. A troca de carga entre entidades é criptografada de modo que faça sentido apenas para o remetente e o receptor, mas não o contrário. A confidencialidade de mensagens é executada usando a criptografia simétrica e assimétrica. Os certificados são usados para implementar a criptografia, isto é, a criptografia e descriptografia de mensagens.
- **Integridade:** a integridade garante que a troca de cargas e mensagens entre remetente e receptor não seja adulterada. O receptor recebe a mesma mensagem como enviada pelo remetente. Os hashes e as assinaturas digitais são os mecanismos de implementação para verificar a integridade de mensagens recebidas.

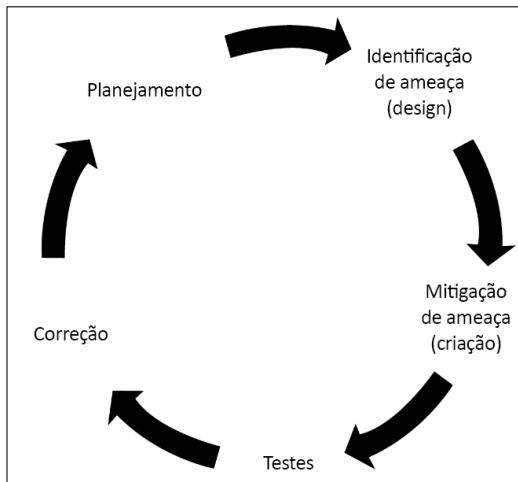
A segurança é uma parceria entre o provedor e o consumidor do serviço. Ambas as partes têm diferentes níveis de controle em pilhas de implantação inteiras e cada um deve implementar as práticas recomendadas de segurança para garantir que todas as ameaças sejam identificadas e mitigadas. Já vimos no *Capítulo 1, Introdução*, que a nuvem fornece amplamente três paradigmas – IaaS, PaaS e SaaS – e cada um com diferentes níveis de controle colaborativo sobre a pilha de implantação. Cada parte deve implementar práticas de segurança para componentes sob seu controle e âmbito. A falha ao implementar a segurança em qualquer camada da pilha ou por qualquer das partes tornaria a implantação e o aplicativo vulneráveis a ataques.

Ciclo de vida de segurança

A segurança é geralmente considerada como um requisito não funcional para uma solução. No entanto, com os crescentes ataques cibernéticos, ela é considerada como um requisito funcional nos dias de hoje.

Cada organização segue algum tipo de gerenciamento de ciclo de vida do aplicativo para seus aplicativos. Quando a segurança é tratada como um requisito funcional, ela deve seguir o mesmo processo de desenvolvimento de aplicativos. A segurança não deve ser deixada para um segundo momento. Ela deve fazer parte do aplicativo desde o início. No âmbito da fase de planejamento global para um aplicativo, a segurança também deve ser planejada. Dependendo da natureza do aplicativo, diferentes tipos e categorias de ameaças devem ser identificados e, com base nessas identificações, eles devem ser documentados em termos de abordagem e escopo para atenuá-los. Um exercício de modelagem de ameaças deve ser realizado para ilustrar a ameaça a que cada componente pode estar sujeito. Isso levará à criação de padrões e políticas de segurança para o aplicativo. Isso normalmente é a fase de design de segurança. A próxima fase é a chamada de **mitigação de segurança** ou fase de **compilação**. Nela, a implementação de segurança em termos de código e configuração é executada para atenuar as ameaças e os riscos de segurança.

Um sistema não pode estar seguro até que seja testado. Os testes de penetração adequados e outros testes de segurança devem ser realizados para identificar possíveis mitigações de ameaças que não foram implementadas ou foram ignoradas. Os bugs de teste são corrigidos, e o ciclo continua até o final da vida útil do aplicativo. Esse processo de gerenciamento de ciclo de vida do aplicativo, conforme mostrado no diagrama a seguir, deve ser seguido para a segurança:



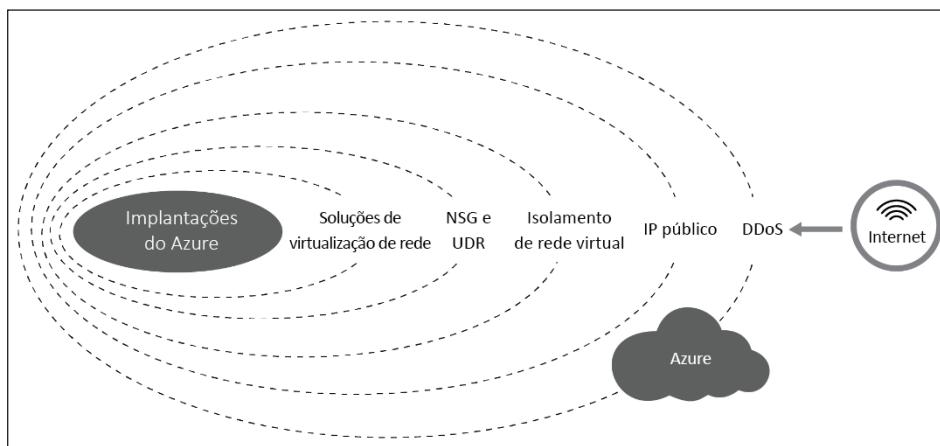
Planejamento, identificação, mitigação, testes e correção são processos iterativos que continuam mesmo quando um aplicativo ou serviço está em funcionamento. Deve haver monitoramento ativo de ambientes e aplicações inteiros para atenuá-los e identificar ameaças proativamente. O monitoramento também deve habilitar alertas e logs de auditoria para ajudar no diagnóstico reativo, na solução de problemas e na eliminação de ameaças e vulnerabilidades.

O ciclo de vida de segurança para qualquer aplicativo começa com a fase de planejamento, que leva à fase de design. Na fase de design, a arquitetura do aplicativo é decomposta em componentes granulares com comunicação discreta e limites de hospedagem. As ameaças são identificadas com base na sua interação com outros componentes dentro e entre os limites de hospedagem. As ameaças identificadas são mitigadas com a implementação de recursos de segurança adequados na arquitetura geral e os testes para identificar se tal vulnerabilidade ainda existe. Depois que o aplicativo é implantado na produção e se torna operacional, ele é monitorado para verificar se possui quaisquer falhas de segurança e vulnerabilidades, e uma correção proativa ou reativa é conduzida.

A Microsoft fornece informações e orientações completas sobre o ciclo de vida da segurança. Elas estão disponíveis em <https://www.microsoft.com/en-us/securityengineering/sdl/practices>.

Segurança do Azure

O Azure fornece todos os seus serviços por meio de data centers em várias regiões. Estes data centers também estão interconectados nas regiões, bem como entre elas. O Azure entende que ele hospeda dados, serviços e aplicativos importantes e de missão crítica para seus clientes. Ele deve garantir que a segurança é de extrema importância para seus data centers e regiões. Os clientes implantam aplicativos na nuvem com base na confiança de que o Azure protegerá seus aplicativos e dados contra vulnerabilidades e violação. Os clientes não migrarão para a nuvem se essa confiança for quebrada e, portanto, o Azure implementa a segurança em todas as camadas, conforme visto no próximo diagrama, do perímetro físico dos data centers para os componentes de software lógicos. Cada camada é protegida e até mesmo a equipe do data center do Azure não tem acesso a elas:



A segurança é de suma importância para a Microsoft e para o Azure. O Azure é uma plataforma de nuvem hospedada pela Microsoft. A Microsoft garante que a confiança seja construída com seus clientes e faz isso garantindo que a implantação, as soluções e os dados do cliente estejam completamente seguros, física e virtualmente. As pessoas não usarão nenhuma plataforma de nuvem se ela não for física e digitalmente segura. Para garantir que os clientes tenham confiança no Azure, cada atividade em desenvolvimento do Azure é planejada, documentada, auditada e monitorada de uma perspectiva de segurança. Os data centers do Azure físicos são protegidos por qualquer intrusão e acesso não autorizado. Na verdade, nem a equipe de operações e o pessoal da Microsoft têm acesso à solução e aos dados do cliente. Alguns dos recursos de segurança prontos para uso fornecidos pelo Azure estão listados aqui:

- **Proteger o acesso do usuário:** a implantação, a solução e os dados de um cliente só podem ser acessados pelo cliente. Até mesmo o pessoal do data center do Azure não tem acesso a nenhum artefato do cliente. Os clientes podem permitir o acesso de outras pessoas; no entanto, isso fica a critério deles.

- **Criptografia em repouso:** o Azure criptografa todo o gerenciamento de dados de modo que eles não possam ser lidos por qualquer pessoa. Ele também fornece essa funcionalidade para seus clientes, bem como aqueles que podem criptografar seus dados em repouso.
- **Criptografia em trânsito:** o Azure criptografa todos os dados que fluem de sua rede. Ele também garante que seu backbone de rede esteja protegido contra acessos não autorizados.
- **Monitoramento e auditoria contínuos:** o Azure monitora todos os seus data centers ativamente e de maneira contínua. Ele identifica ativamente qualquer violação, ameaça ou risco e os mitiga.

O Azure atende aos padrões de conformidade internacionais e específicos da indústria, do país e do local. Eles podem ser encontrados em <https://www.microsoft.com/en-us/trustcenter/compliance/complianceofferings>.

Segurança de IaaS

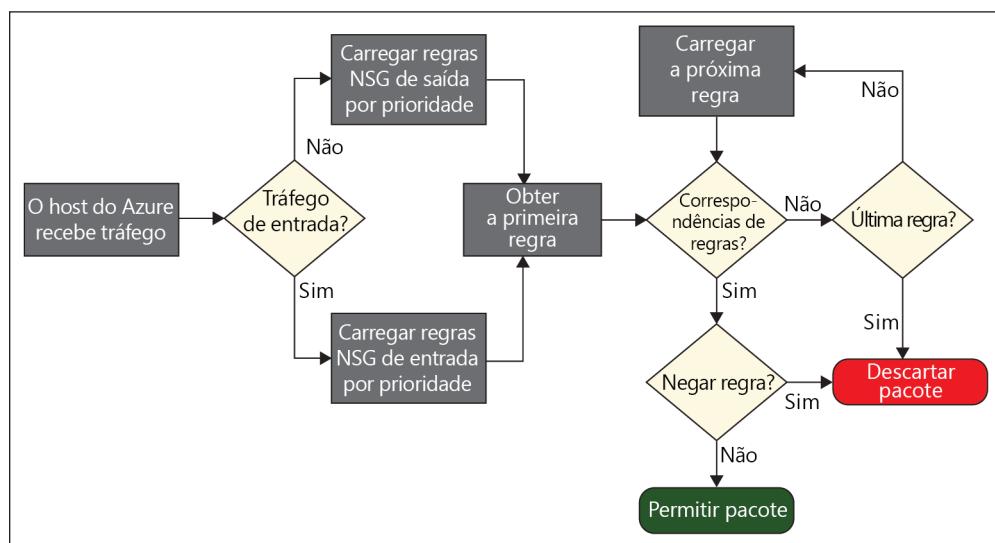
O Azure é uma plataforma madura para a implantação de soluções de IaaS. Há muitos usuários do Azure que querem o controle completo sobre suas implantações, e normalmente eles usam IaaS para suas soluções. É importante que essas implantações e soluções sejam seguras por padrão e design. O Azure fornece valiosos recursos de segurança para proteger soluções IaaS. Nesta seção, algumas das principais características serão cobertas.

Grupos de segurança de rede

O mínimo necessário para a implantação de IaaS consiste em máquinas e redes virtuais. As máquinas virtuais podem ser expostas à Internet por meio da aplicação de um IP público à sua interface de rede ou podem estar disponíveis somente para os recursos internos. Alguns desses recursos internos, por sua vez, podem ser expostos à Internet. De qualquer maneira, as máquinas virtuais devem ser protegidas de modo que as solicitações não autorizadas nem mesmo as alcancem. As máquinas virtuais devem ser protegidas usando instalações que podem filtrar solicitações na própria rede, em vez de deixá-las chegar a uma máquina virtual para que a VM tome providências em relação a elas. Delimitação é um mecanismo que as máquinas virtuais usam como um de seus mecanismos de segurança. Essa delimitação pode permitir ou negar solicitações com base em seu protocolo, IP de origem, IP de destino, porta de origem e porta de destino. Esse recurso é implantado com o recurso Grupos de Segurança de Rede (NSGs) do Azure. Os NSGs são compostos por regras que são avaliadas para solicitações de entrada e de saída. Dependendo da execução e da avaliação dessas regras, será determinado se as solicitações deverão ter o acesso permitido ou negado.

Os NSGs são flexíveis e podem ser aplicados a uma sub-rede da rede virtual ou interfaces de rede individuais. Ao aplicá-los a uma sub-rede, as regras de segurança são aplicadas a todas as máquinas virtuais hospedadas nessa sub-rede. Por outro lado, aplicá-lo a uma interface de rede afeta as solicitações apenas para uma determinada máquina virtual associada a essa interface de rede. Também é possível aplicar NSGs a interfaces de rede e de sub-rede da rede simultaneamente. Normalmente, esse design deve ser usado para a aplicação de regras comuns de segurança no nível de sub-rede da rede e as regras de segurança diferentes exclusivas no nível da interface de rede. Ele ajuda no design de regras de segurança modulares e sua aplicabilidade.

O fluxo para avaliar os NSGs é mostrado no diagrama a seguir:



Quando uma solicitação atinge um host do Azure (com base no tipo de solicitação: de entrada ou saída), as regras apropriadas são carregadas, e cada uma é executada em relação à solicitação/resposta. Se a regra corresponder à solicitação/resposta, a solicitação/resposta será permitida ou negada. A correspondência de regras consiste em informações importantes de solicitação/resposta, como endereço IP de origem, endereço IP de destino, porta de origem, porta de destino e protocolo usado.

Design de NSG

O primeiro passo no design é determinar os requisitos de segurança do recurso.

O seguinte deve ser determinado ou considerado:

- O recurso só pode ser acessado da Internet?
- O recurso pode ser acessado dos recursos internos e da Internet?
- O recurso só pode ser acessado dos recursos internos?
- Determinar os recursos, o balanceador de carga, as máquinas virtuais e os gateways usados.
- Configuração de uma rede virtual e sua sub-rede.

Com os resultados dessas investigações, um design NSG adequado deve ser criado. Idealmente, deve haver várias sub-redes da rede para cada tipo de recurso e workload. Não é recomendável implantar平衡adores de carga e máquinas virtuais na mesma sub-rede.

Considerando seus requisitos, devem ser determinadas as regras que são comuns para sub-redes e workloads diferentes de máquina virtual. Por exemplo, para uma implantação do SharePoint, o aplicativo de front-end e os SQL Servers são implantados em sub-redes separadas. Portanto, devem ser determinadas regras para cada sub-rede.

Após a identificação das regras comuns no nível da sub-rede, as regras para recursos individuais devem ser identificadas e ser aplicadas ao nível da interface de rede.

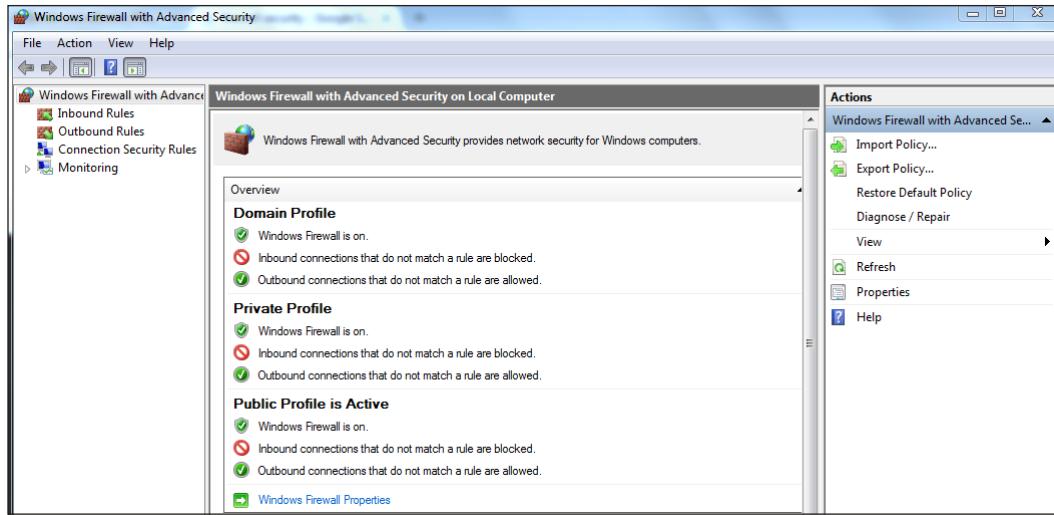
É importante entender que, se uma regra permite uma solicitação de entrada em uma porta, essa porta também poderá ser usada para solicitações de saída sem qualquer configuração.

Se os recursos puderem ser acessados na Internet, as regras deverão ser criadas com portas e intervalos de IP específicos onde for possível. Testes funcionais e de segurança cuidadosos devem ser executados para garantir que regras NSG adequadas e ideais sejam abertas e fechadas.

Firewalls

Os NSGs fornecem perímetros de segurança externa para solicitações. No entanto, isso não significa que as máquinas virtuais não devem implementar medidas de segurança adicionais. É sempre melhor implementar a segurança interna e externamente. As máquinas virtuais no Linux ou no Windows fornecem um mecanismo para filtrar solicitações no nível do sistema operacional. Isso é conhecido como firewall no Windows e no Linux.

É aconselhável implementar firewalls para sistemas operacionais. Eles ajudam na criação de um muro de segurança virtual que permite somente as solicitações consideradas confiáveis. Quaisquer pedidos não confiáveis terão o acesso negado. Existem até mesmo dispositivos de firewall físicos mas, no software da nuvem, os firewalls do sistema operacional são usados. A captura de tela a seguir mostra a configuração do firewall no sistema operacional Windows:



Os firewalls filtram pacotes de rede e identificam portas de entrada e endereços IP. Com as informações desses pacotes, o firewall avalia as regras e decide se deve permitir ou negar o acesso.

Design de firewall

Como uma prática recomendada, os firewalls devem ser avaliados para sistemas operacionais individuais. Cada máquina virtual tem uma responsabilidade distinta na implantação e a solução globais. As regras para essas responsabilidades individuais devem ser identificadas e os firewalls devem ser abertos e fechados em conformidade.

Ao avaliar as regras de firewall, é importante considerar as regras de NSG no nível da interface de rede individual e de sub-rede. Se não for feito corretamente, é possível que as regras sejam negadas no nível do NSG, mas deixadas em aberto no nível de firewall e vice-versa. Se uma solicitação for permitida no nível do NSG e negada no nível de firewall, o aplicativo não funcionará conforme o esperado, já que os riscos de segurança aumentarão se uma solicitação for negada no nível da regra NSG e permitida no nível do firewall.

Um firewall ajuda você a criar várias redes isoladas por suas regras de segurança. Testes funcionais e de segurança cuidadosos devem ser executados para garantir que regras de firewall adequadas e ideais sejam abertas e fechadas.

Redução da área de superfície de ataque

Os NSGs e os firewalls ajudam no gerenciamento de solicitações autorizadas para o ambiente. No entanto, o ambiente não deve ser abertamente exposto a ataques de segurança. A área de superfície do sistema deve ser idealmente habilitada para atingir sua funcionalidade, mas desabilitada o suficiente para que os invasores não possam encontrar brechas e acessar áreas abertas sem qualquer uso pretendido ou abertas, mas não protegidas adequadamente. A segurança deve ser adequadamente fortalecida, tornando difícil para qualquer invasor entrar no sistema.

Algumas das configurações que devem ser feitas incluem:

- Remover todos os grupos e usuários desnecessários do sistema operacional.
- Identificar a associação de grupo para todos os usuários.
- Implementar políticas de grupo usando os serviços de diretório.
- Bloquear a execução de scripts, a menos que ele seja assinado por autoridades confiáveis.
- Registrar em log e auditar todas as atividades.
- Instalar software antivírus e malware, agendar varreduras e atualizar as definições frequentemente.
- Desabilitar ou desligar serviços que não são necessários.
- Bloquear o sistema de arquivos para permitir somente acessos autorizados.
- Bloquear as alterações no registro.
- Um firewall deve ser configurado de acordo com os requisitos.
- A execução de script do PowerShell deve ser definido como restricted ou RemoteSigned.
- Habilitar a proteção avançada por meio do Internet Explorer.
- Restringir a capacidade de criar novos usuários e grupos.
- Remover o acesso à Internet e implementar servidores de salto para RDP.
- Proibir o registro em log de servidores que usam o RDP por meio da Internet. Em vez de usar a VPN de site a site, a VPN de ponto a site ou o express route para RDP em máquinas remotas de dentro da rede.
- Implantar regularmente todas as atualizações de segurança.
- Executar a ferramenta de gerenciador de conformidade de segurança no ambiente e implementar todas as suas recomendações.
- Monitoreativamente o ambiente usando a Central de Segurança e o Operations Management Suite.

- Implantar dispositivos virtuais de rede para rotear tráfego para proxies internos e reversos.
- Todos os dados confidenciais, como configuração, cadeias de conexão e credenciais, devem ser criptografados.

Implementação de servidores de salto

Convém remover o acesso à Internet de máquinas virtuais. Também é uma prática recomendada limitar a acessibilidade de serviços de área de trabalho remoto da Internet, mas então como você acessa as máquinas virtuais? Uma boa maneira é permitir apenas os recursos internos para RDP em máquinas virtuais usando opções da VPN do Azure. No entanto, também há uma outra maneira – por meio do uso de **servidores de salto**.

Os servidores de salto são servidores implantados na **zona desmilitarizada (DMZ)**. Isso significa que eles não estão na rede que hospeda as soluções e os aplicativos principais. Em vez disso, estão numa rede ou sub-rede separada. A finalidade principal do servidor de salto é aceitar solicitações RDP de usuários e ajudá-los a iniciar sessão nele. Desse servidor de salto, os usuários ainda podem navegar para outras máquinas virtuais usando o RDP. Eles têm acesso a duas ou mais redes: uma que tenha conectividade com o mundo externo e outro interno para a solução. O servidor de salto implementa todas as restrições de segurança e fornece um cliente seguro para conexão com outros servidores. Normalmente, o acesso a emails e à Internet está desabilitado em servidores de salto.

Um exemplo de implantação de um servidor de salto com os VMSS está disponível em <https://azure.microsoft.com/en-in/resources/templates/201-vmss-windows-jumpbox/> usando modelos do Azure Resource Manager.

Segurança de PaaS

O Azure fornece inúmeros serviços de PaaS, cada um com seus próprios recursos de segurança. Em geral, os serviços PaaS podem ser acessados usando credenciais, certificados e tokens. Os serviços PaaS permitem a geração de tokens de acesso de segurança de curta duração. Os aplicativos cliente podem enviar esse token de acesso de segurança para representar usuários confiáveis. Nesta seção, vamos cobrir alguns dos mais importantes serviços PaaS que são usados em quase todas as soluções.

Log Analytics

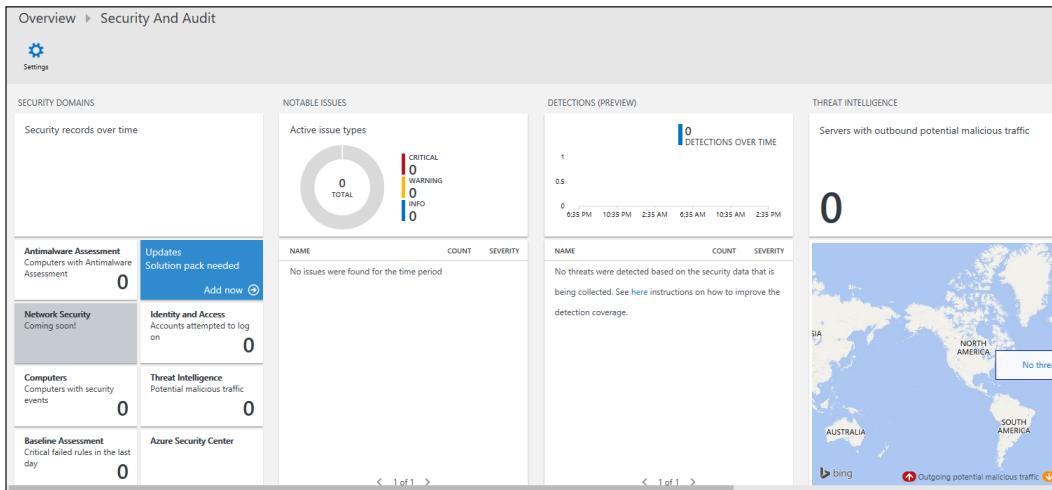
O Log Analytics é uma nova plataforma de análise para o gerenciamento de implantações de nuvem, data centers na infraestrutura local e soluções híbridas.

Ele fornece várias soluções modulares – funcionalidade específica que ajuda a implementar um recurso. Por exemplo, as soluções de segurança e auditoria ajudam a determinar uma visão completa de segurança para a implantação de uma organização. Da mesma forma, existem muitas outras soluções, como automação e controle de alterações, que devem ser implementadas de uma perspectiva de segurança.

A segurança e auditoria do Log Analytics fornece informações nas cinco categorias a seguir:

- **Domínios de segurança:** eles permitem exibir registros de segurança, avaliações de malware, avaliações de atualização, segurança de rede, informações de identidade e de acesso e computadores com eventos de segurança. O acesso também é fornecido para o painel da Central de Segurança do Azure.
- **Avaliação de antimalware:** ajuda na identificação de servidores que não são protegidos contra malware e têm problemas de segurança. Ela fornece uma exposição global para problemas de segurança potenciais e avalia seu nível de importância. Os usuários podem tomar ações proativas com base nessas recomendações. As subcategorias da Central de Segurança do Azure fornecem informações coletadas pela Central de Segurança do Azure.
- **Principais problemas:** identificam rapidamente problemas ativos e classificam a gravidade deles.
- **Detecções:** esta categoria está no modo de versão prévia. Permite a identificação de padrões de ataque por meio da visualização de alertas de segurança.
- **Inteligência contra ameaças:** ajuda na identificação de padrões de ataque visualizando o número total de servidores com tráfego IP de saída mal-intencionado, o tipo de ameaça mal-intencionada e um mapa que mostra de onde esses IPs estão vindo.

Os detalhes anteriores, quando visualizados no portal, são mostrados na seguinte captura de tela:



Armazenamento

As contas de armazenamento desempenham um papel importante na arquitetura geral da solução. As contas de armazenamento podem armazenar informações importantes, como dados PII do usuário, transações comerciais e dados. É de extrema importância que as contas de armazenamento sejam seguras e permitam apenas o acesso de usuários autorizados. Os dados armazenados são criptografados e transmitidos por meio de canais seguros. O armazenamento e os usuários e aplicativos cliente que utilizam a conta de armazenamento e seus dados desempenham um papel crucial na segurança geral dos dados. Eles também devem manter dados criptografados em todos os momentos. Isto também inclui credenciais e cadeias de conexão a armazenamentos de dados.

O Azure fornece **controle de acesso com base em função (RBAC)** para administrar quem pode gerenciar contas de Armazenamento do Azure. Essas permissões RBAC são concedidas para usuários e grupos no Azure **do Active Directory (AD)**. No entanto, quando um aplicativo a ser implantado no Azure for criado, terá usuários e clientes que não estão disponíveis no Azure AD. Para permitir que os usuários acessem a conta de armazenamento, o Armazenamento do Azure fornece armazenamento de chaves de acesso. Existem dois tipos de chaves de acesso no nível da conta de armazenamento – primário e secundário. Os usuários com essas chaves podem se conectar à conta de armazenamento. Essas chaves de acesso de armazenamento são usadas na etapa de autenticação ao acessar a conta de armazenamento. Os aplicativos podem acessar contas de armazenamento usando chaves primárias ou secundárias. Duas chaves são fornecidas de modo que, se a chave primária for comprometida, os aplicativos poderão ser atualizados para usar a chave secundária, enquanto a primária é gerada novamente. Isso ajuda a minimizar o tempo de inatividade do aplicativo. Além disso, ela aumenta a segurança ao remover a chave comprometida sem afetar os aplicativos. Os detalhes da chave de armazenamento, como visto no portal do Azure, são mostrados na captura de tela a seguir:

The screenshot shows the 'Storage account name' input field filled with a placeholder and a 'Regenerate key' button. Below it, the 'Default keys' section displays two rows of keys:

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

Each key row has a 'Regenerate key' button and a 'Download connection string' button.

O Armazenamento do Azure fornece quatro serviços – blob, arquivos, filas e tabelas em uma conta. Cada um desses serviços também fornece a infraestrutura para protegerem-se usando tokens de acesso seguro. Uma **assinatura de acesso compartilhado (SAS)** é um URI que concede direitos de acesso restrito a serviços de Armazenamento do Azure -- blobs, arquivos, filas e tabelas. Esses tokens SAS podem ser compartilhados com clientes que não são confiáveis a ponto de receber a chave inteira da conta de armazenamento para restringir o acesso deles a determinados recursos da conta. Ao distribuir um URI de SAS a esses clientes, o acesso a recursos é concedido por um período especificado.

Os tokens SAS existem na conta de armazenamento e nos níveis individuais de blob, arquivo, tabela e fila. A assinatura no nível da conta de armazenamento é mais poderosa e tem os direitos para permitir e negar permissões no nível do serviço individual. Ela também pode ser usada em vez de níveis do serviço de recurso individual:

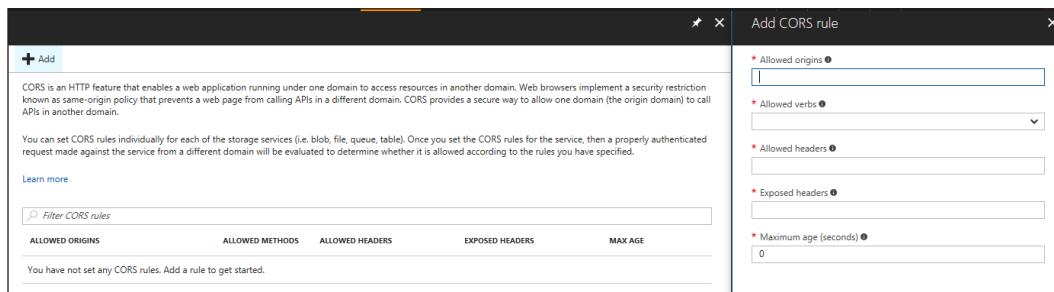
The screenshot shows the configuration interface for generating a Shared Access Signature (SAS) in the Azure Storage portal. The configuration includes:

- 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:** (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

É preferível gerar e compartilhar tokens SAS do que compartilhar chaves da conta de armazenamento. Os tokens SAS fornecem acesso granular a recursos e também podem ser combinados. Esses tokens incluem **ler, gravar, excluir, listar, adicionar, criar, atualizar e processar**. Além disso, até mesmo o acesso a recursos pode ser determinado durante a geração de tokens SAS. Poderia ser para blobs, tabelas, filas e arquivos individualmente ou uma combinação deles. As chaves de conta de armazenamento servem para a conta inteira e não podem ser restritas a serviços individuais. Elas também não podem ser restritas do ponto de vista de permissões. É muito mais fácil criar e revogar tokens SAS do que para as chaves de acesso de armazenamento. Os tokens SAS podem ser criados para uso por um determinado período, após o qual eles se tornam inválidos automaticamente.

Lembre-se de que, se as chaves de conta de armazenamento forem regeneradas, o token SAS baseado nelas será invalidado, e um token SAS mais novo deverá ser criado e compartilhado com os clientes.

Roubo de cookies, injeção de scripts e ataques de negação de serviço são meios comuns usados por invasores para violar um ambiente e roubar dados. Os navegadores e o protocolo HTTP implementam um mecanismo embutido que garante que essas atividades mal-intencionadas não possam ser executadas. Geralmente, qualquer coisa entre domínios não é permitida por HTTP e=ou navegadores. Um script em execução em um domínio não pode solicitar recursos de outro domínio. No entanto, há casos de uso válidos onde tais solicitações devem ser autorizadas. O protocolo HTTP implementa o **Compartilhamento de Recursos entre Origens (CORS)**. Com a ajuda de CORS, é possível acessar recursos entre domínios e fazê-los funcionar. O Armazenamento do Azure configura as regras CORS para blobs, arquivo, fila e recursos de tabela. O Armazenamento do Azure permite a criação de regras avaliadas para cada solicitação autenticada. Se as regras forem atendidas, a solicitação terá permissão para acessar o recurso:



Os dados não devem apenas ser protegidos em trânsito, eles também devem ser protegidos em repouso. Se os dados em repouso não forem criptografados, qualquer um que tiver acesso à unidade física no data center poderá lê-los. Embora a possibilidade seja insignificante, os clientes ainda devem criptografar seus dados. A criptografia do serviço de armazenamento também ajuda a proteger dados em repouso. Esse serviço funciona de forma transparente e injeta a si mesmo sem que os usuários tome conhecimento. Ele criptografa dados quando eles são salvos em uma conta de armazenamento e os descriptografa automaticamente quando eles são lidos. Todo esse processo acontece sem que os usuários executem qualquer atividade adicional.

As chaves de conta do Azure devem ser giradas periodicamente. Isso irá garantir que um invasor não seja capaz de romper o acesso a contas de armazenamento.

Também convém gerar as chaves novamente. No entanto, isso deve ser avaliado em relação ao seu uso em aplicativos existentes. Se isso interromper o aplicativo existente, esses aplicativos deverão ser priorizados para o gerenciamento de alterações, que deverão ser aplicadas gradualmente.

Os tokens SAS no nível de serviço individual com um período de tempo limitado devem ser gerados e fornecidos aos usuários que devem acessar os recursos tanto quanto possível. As permissões devem ser avaliadas e permissões ideais devem ser fornecidas.

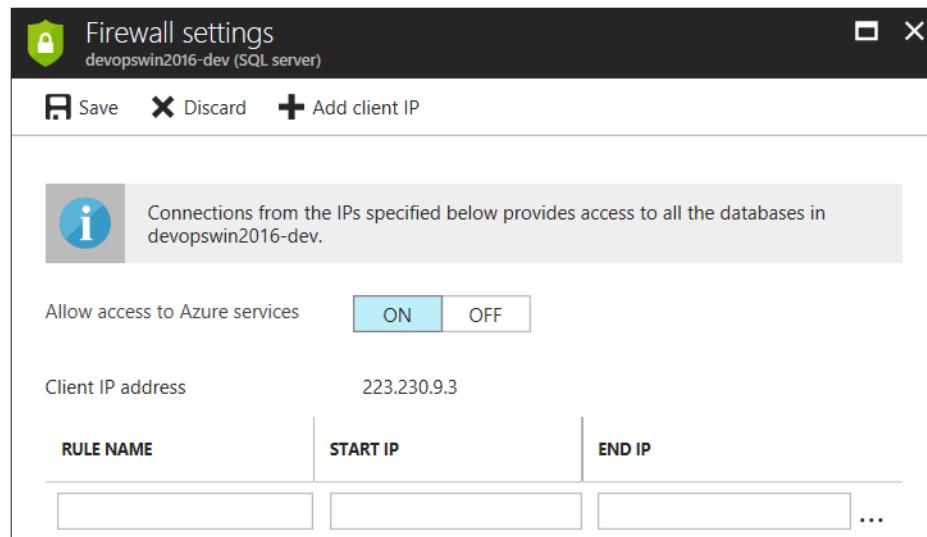
As chaves SAS e as chaves de conta de armazenamento devem ser armazenadas em um Azure Key Vault. Ele fornece armazenamento de segurança e acesso a elas. Essas chaves podem ser lidas em tempo de execução por aplicativos do cofre de chaves, em vez de armazená-las em arquivos de configuração.

SQL do Azure

O SQL Server armazena dados relacionais no Azure. Ele é um serviço SaaS que fornece uma plataforma altamente disponível, escalável, centrada na performance e segura para armazenar dados. Ele é acessível de qualquer lugar, com qualquer programação linguagem e plataforma. Os clientes precisam de uma cadeia de conexão composta por servidor, banco de dados e informações de segurança para se conectar a ele.

O SQL Server fornece configurações de firewall que impedem o acesso de qualquer pessoa por padrão. Os endereços IP e os intervalos devem ser incluídos na lista de permissões para acessar o SQL Server. Apenas os endereços IP que arquitetos têm certeza de que pertencem aos clientes ou a parceiros devem ser incluídos na lista de permissões. Existem implantações no Azure para as quais há um monte de endereços IP ou os endereços IP não são conhecidos, como aplicativos implantados no Azure Functions ou em Aplicativos Lógicos. Para que tais aplicativos acessem o SQL do Azure, ele permite uma lista de permissões de todos os endereços IP para serviços Azure em assinaturas.

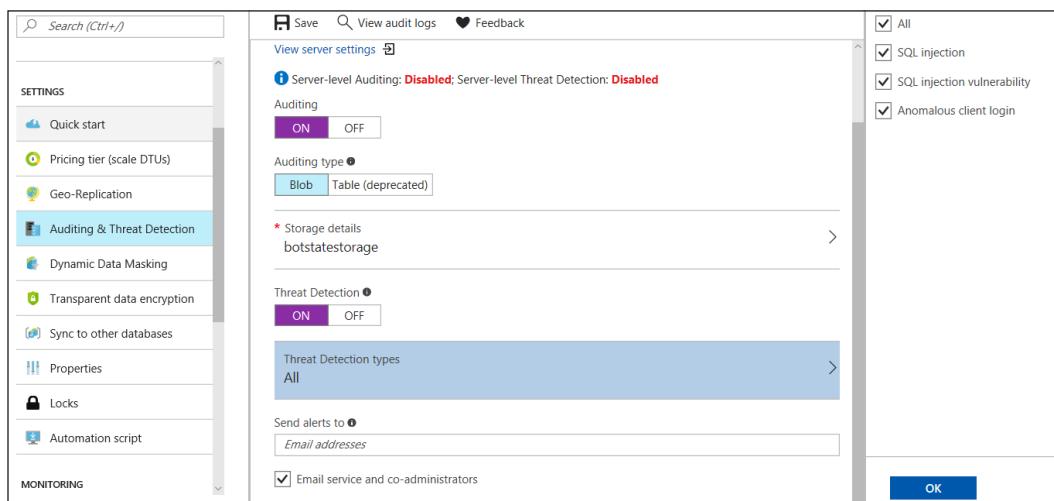
Lembre-se de que a configuração do firewall está no nível do servidor, e não no nível do banco de dados. Isso significa que quaisquer alterações aqui afetam todos os bancos de dados dentro de um servidor:



O SQL do Azure também fornece segurança avançada ao criptografar os dados em repouso. Isso garante que ninguém, incluindo os administradores de data center do Azure, possa exibir os dados armazenados no SQL Server. A tecnologia usada pelo SQL Server para criptografar dados em repouso é conhecida como **Criptografia de Dados Transparente (TDE)**. Não há nenhuma alteração necessária no nível do aplicativo para implementar a TDE. O SQL Server criptografa e descriptografa os dados de modo transparente quando o usuário salva e lê os dados. Esse recurso está disponível no nível do banco de dados.

O SQL Server também fornece **Máscara de Dados Dinâmicos (DDM)**, que é especialmente útil para mascarar certos tipos de dados, como cartões de crédito ou dados PII de usuários. O mascaramento não é igual à criptografia. Ele não criptografa dados, mas apenas os mascara, o que garante que os dados não estejam em formato legível. Os usuários devem mascarar e criptografar dados confidenciais no SQL Server do Azure.

O SQL Server também fornece um serviço de **auditoria e detecção de ameaças** para todos os servidores. Existem serviços avançados de coleta de dados e inteligência em execução nesses bancos de dados para descobrir ameaças e vulnerabilidades e alertar os usuários com base neles. Os logs de auditoria são mantidos pelo Azure em contas de armazenamento e podem ser exibidos por administradores para ação. Os threads como a injeção de SQL e logons de cliente anônimo podem gerar alertas para que os administradores possam ser informados por email:



Os dados podem ser mascarados no SQL do Azure. Isso ajuda no armazenamento de dados em um formato que não faça sentido:

Save Discard Add mask Feedback

Masking rules

MASK NAME

MASK FUNCTION

You haven't created any masking rules.

SQL users excluded from masking (administrators are always excluded) [?](#)

SQL users excluded from masking (administrators are always excluded)

Recommended fields to mask

SCHEMA **TABLE** **COLUMN**

There are no recommended fields to mask

Add Discard Delete

Mask name

Select what to mask

Schema

Table

Column

Select how to mask

Masking field format

Default value (0, xxxx, 01-01-1900)

O SQL do Azure também fornece **criptografia de dados transparente** para criptografar dados em repouso, conforme mostrado na captura de tela a seguir:

Search (Ctrl+ /)

Save Discard Feedback

SETTINGS

Quick start

Pricing tier (scale DTUs)

Geo-Replication

Auditing & Threat Detection

Dynamic Data Masking

Transparent data encryption

Encrypts your databases, backups enable TDE, go to each database.

Learn more [↗](#)

Data encryption

ON OFF

Encryption status

Encrypted

Azure Key Vault

A proteção de recursos usando senhas, chaves, credenciais, certificados e identificadores exclusivos é um elemento importante para qualquer ambiente e aplicativo. Eles são elementos importantes da perspectiva da segurança. Eles precisam ser protegidos e garantir que esses recursos permaneçam seguros e não sejam comprometidos é um importante pilar da arquitetura de segurança. O gerenciamento e as operações que mantêm os segredos e as chaves seguras enquanto os mantém disponíveis quando necessário são aspectos importantes que não podem ser ignorados. Normalmente, esses segredos são usados por todo o lado – dentro do código-fonte, dos arquivos de configuração, em pedaços de papel e em outros formatos digitais. Para superar esses desafios e armazenar todos os segredos uniformemente em um armazenamento centralizado e seguro, o Azure Key Vault deve ser criado.

O Azure Key Vault é bem integrado a outros serviços do Azure. Por exemplo, seria fácil usar um certificado armazenado no Azure Key Vault e implantá-lo no armazenamento de certificados de máquinas virtuais do Azure. Todos os tipos de chaves, incluindo as chaves de armazenamento, as chaves IoT e de evento e as cadeias de conexão podem ser armazenados como segredos no Azure Key Vault. Eles podem ser recuperados e usados de forma transparente sem ninguém os exibindo ou armazenando temporariamente em qualquer lugar. As credenciais do SQL Server e outros serviços também podem ser armazenadas no Azure Key Vault.

O Azure Key Vault funciona por região. Isso significa que um recurso do Azure Key Vault deve ser provisionado na mesma região onde o aplicativo e o serviço são implantados. Se uma implantação consistir em mais de uma região e precisar dos serviços do Azure Key Vault, várias instâncias dele deverão ser provisionadas.

Um recurso importante do Azure Key Vault é que os segredos, chaves e certificados não são mantidos no armazenamento geral. Esses dados confidenciais são copiados pelo módulo de segurança de hardware. Isso significa que esses dados são armazenados em hardware separado no Azure que só podem ser desbloqueados por chaves pertencentes a usuários.

Monitoramento e auditoria de segurança

O Azure fornece os dois recursos de segurança importantes a seguir para gerenciar todos os aspectos da assinatura, grupos de recursos e recursos do Azure:

- Azure Monitor
- Central de Segurança do Azure

Azure Monitor

No Azure Monitor, você monitora os recursos do Azure em um único lugar. Ele fornece informações sobre recursos do Azure e o estado deles. Ele fornece uma interface de consulta avançada usando as informações que podem ser divididas usando dados nos níveis da assinatura, grupo de recursos, recurso individual e tipo de recurso.

O Azure Monitor pode ser usado por meio do portal do Azure, do PowerShell, da CLI e da API REST:

The screenshot shows the Azure Monitor interface for the Activity log. On the left, there's a sidebar with 'EXPLORE' (Activity log, Operation log (classic), Metrics, Diagnostics settings, Log search) and 'MANAGE' (Alerts, Action groups). The main area has a search bar at the top. Below it are several filter dropdowns: 'Select query ...', 'Subscription', 'Resource group', 'Resource', 'Resource type', 'Operation', 'Event category', 'Event severity', 'Timestep', 'All resource groups', 'All resources', 'All resource types', 'Event initiated by', 'Last 6 hours', 'All categories', '4 selected', and 'Email or name or service principal /'. There are 'Apply' and 'Reset' buttons. To the right, a message says 'Insights (Last 24 hours): 0 failed deployments | 0 role assignments | 0 errors | 0 alerts fired | 0 outage notifications'. Below this is a table with columns: OPERATION NAME, STATUS, TIME, TIME STAMP, SUBSCRIPTION, and EVENT INITIATED BY. A note says 'No results to display'.

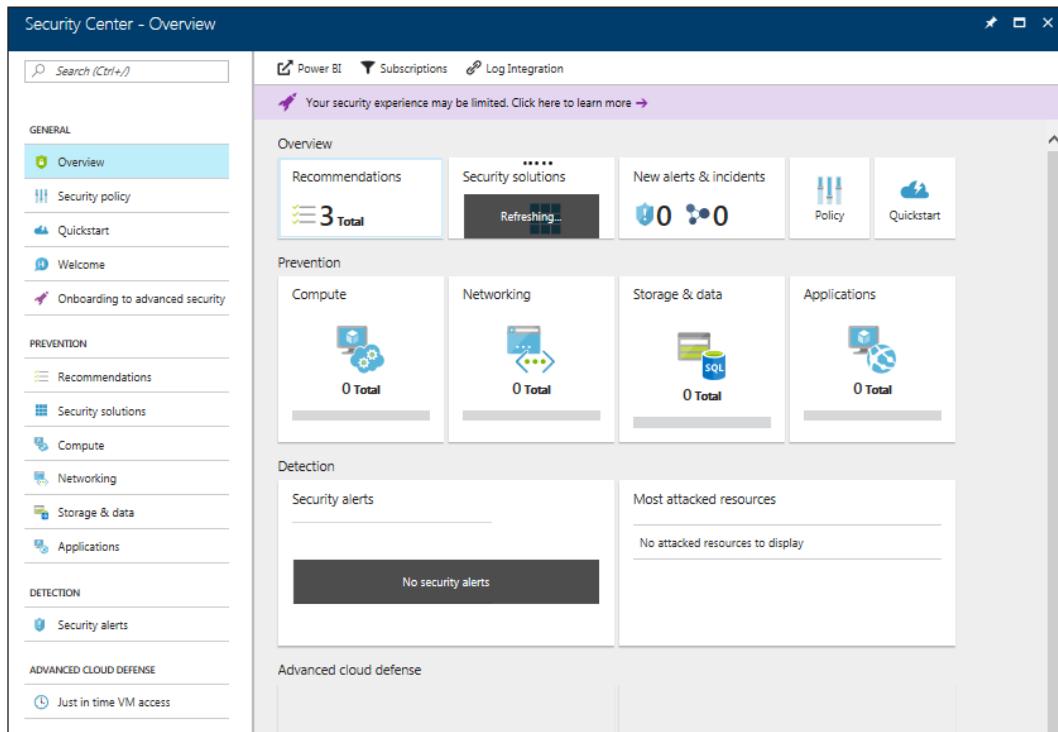
Os logs a seguir são aqueles fornecidos pelo Azure Monitor:

- **Log de atividades:** fornece todas as operações no nível de gerenciamento realizadas nos recursos. Ele fornece detalhes sobre o momento da criação, o criador, o tipo de recurso e o status dos recursos.
- **Log de operação (clássico):** fornece detalhes de todas as operações realizadas nos recursos dentro de um grupo de recursos e de uma assinatura.
- **Métricas:** ajudam na obtenção de informações sobre o nível de performance para recursos individuais e define alertas neles.
- **Configurações de diagnóstico:** ajudam na configuração dos logs de efeito por meio da configuração do Armazenamento do Azure para armazenar logs, transmiti-los em tempo real para hubs de eventos do Azure e enviá-los ao Log Analytics.
- **Pesquisa de log:** ajuda na integração do Log Analytics com o Azure Monitor.

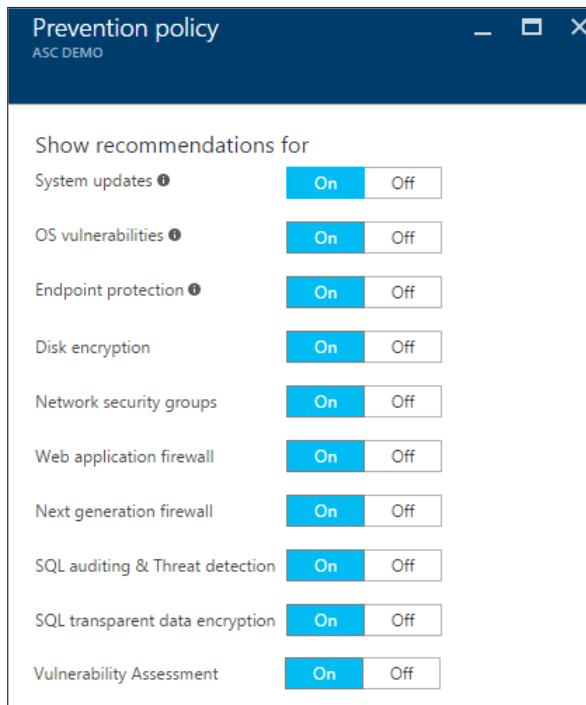
O Azure Monitor pode ajudar a identificar incidentes relacionados à segurança e a tomar ações apropriadas. É importante que somente pessoas autorizadas possam acessar o Azure Monitor, já que ele pode conter informações confidenciais.

Central de Segurança do Azure

A Central de Segurança do Azure, como o nome sugere, é um único lugar para todas as necessidades de segurança. Geralmente há duas atividades relacionadas à segurança – implementação de segurança e monitoramento para quaisquer ameaças e violações. A Central de Segurança foi criada principalmente para ajudar com essas duas atividades. A Central de Segurança do Azure permite que os usuários definam suas políticas de segurança e as implementem nos recursos do Azure. Com base no estado atual dos recursos Azure, a Central de Segurança Azure fornece recomendações de segurança para fortalecer a solução e os recursos individuais do Azure. As recomendações incluem quase todas as práticas recomendadas de segurança do Azure, incluindo a criptografia de dados e discos, proteção de rede, discos, proteção de ponto de extremidade, listas de controle de acesso, listas de permissões de solicitações de entrada e bloqueio de solicitações não autorizadas. A gama de recursos de componentes de infraestrutura, como平衡adores de carga, grupos de segurança de rede e recursos de rede virtual para PaaS, como o SQL e o Armazenamento do Azure:



A Central de Segurança do Azure é uma plataforma avançada e fornece recomendações para vários serviços, conforme mostrado na captura de tela a seguir:



Monitoramento

O monitoramento é uma importante questão de arquitetura que deve ser parte de qualquer solução, seja ela grande ou pequena, de missão crítica ou não, na nuvem ou não. Ele não deve ser negligenciado.

O monitoramento refere-se ao ato de manter o controle de soluções e capturar várias informações de telemetria, processá-las, identificar as informações que se qualificam para alertas com base em regras e criá-los. Geralmente, um agente é implantado dentro do ambiente e o monitora, enviando as informações de telemetria para um servidor centralizado, onde ocorrerá o restante do processamento de geração de alertas e notificação de stakeholders.

O monitoramento realiza ações proativas e reativas e avalia a solução. Ele também é o primeiro passo em direção à capacidade de auditoria da solução. Sem a disponibilidade de monitoramento de registros de log, é difícil auditar o sistema em relação a vários aspectos, como segurança, performance e disponibilidade.

O monitoramento ajuda a identificar problemas de disponibilidade, performance e escalabilidade antes que eles aconteçam. Falhas de hardware, configuração incorreta de software e desafios de atualização de patches podem ser descobertos bem antes de impactar os usuários usando o monitoramento e a degradação de performance pode ser corrigidos antes de acontecer.

O monitoramento de logs reativamente indica áreas e locais que estão causando problemas, identifica os problemas e permite reparos mais rápidos e melhores.

As equipes podem identificar padrões de problemas usando o monitoramento de informações de telemetria e os eliminam ao inovar com novas soluções e recursos.

O Azure é um ambiente de nuvem sofisticado que fornece várias funcionalidades e recursos sofisticados de monitoramento para monitorar não apenas a implantação baseada em nuvem, mas também a implantação na infraestrutura local.

Monitoramento do Azure

A primeira pergunta que deve ser respondida é: *o que devemos monitorar?* Essa questão se torna mais importante para soluções que são implantadas na nuvem devido ao seu controle restrito.

Há alguns componentes importantes que devem ser monitorados. Eles incluem:

- Aplicativos personalizados
- Recursos do Azure
- Sistemas operacionais convidados (máquinas virtuais)
- Sistemas operacionais de host (servidores físicos do Azure)
- Infraestrutura do Azure

Há logs e monitoramentos do Azure diferentes para esses componentes.

Logs de atividade do Azure

Anteriormente conhecidos como logs de auditoria e logs operacionais, eles são eventos do plano de controle na plataforma do Azure. Eles fornecem informações gerais e informações de telemetria no nível da assinatura, em vez do nível do recurso individual. Eles rastreiam informações sobre todas as alterações que ocorrem no nível da assinatura, como criação, exclusão e atualização dos recursos usando o **Azure Resource Manager (ARM)**. Eles ajudam a descobrir a identidade (como entidade de serviço, usuários ou grupos) e realizar uma ação (como gravação ou atualização) nos recursos (por exemplo, armazenamento, máquinas virtuais ou SQL) a qualquer momento. Eles fornecem informações sobre os recursos que são modificados em sua configuração, mas não seu funcionamento e execução.

Logs de diagnóstico do Azure

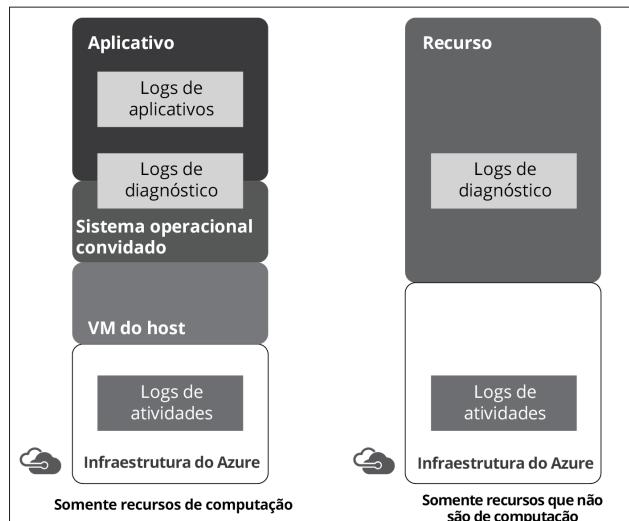
As informações originadas do funcionamento interno dos recursos do Azure são capturadas nos **logs de diagnóstico**. Eles fornecem informações de telemetria sobre as operações de recursos que são inerentes ao recurso. Nem todo recurso fornece logs de diagnóstico, e os recursos que fornecem logs em seu próprio conteúdo são completamente diferentes de outros recursos. Os logs de diagnóstico são configurados para cada recurso individualmente. Exemplos de logs de diagnóstico incluem armazenar um arquivo em um contêiner em um serviço Blob em uma conta de armazenamento.

Logs de aplicativos do Azure

Os logs de aplicativos podem ser capturados pelos recursos do Application Insights e podem ser gerenciados de maneira central. Eles obtêm informações sobre o funcionamento interno de aplicativos personalizados, como as métricas de performance disponibilidade, e os usuários podem obter insights deles para gerenciá-los melhor.

Logs do sistema operacional convidado e host

Os logs de sistema operacional convidado e do host são exibidos para usuários que usam o recurso Azure Monitor. Eles fornecem informações sobre o status dos sistemas operacionais convidado e do host:



Veja a seguir os recursos importantes do Azure relacionados ao monitoramento:

- Azure Monitor
- Azure Application Insights
- Log Analytics, anteriormente conhecido como **Operational Insights**

Há outras ferramentas, como o **System Center Operations Manager (SCOM)**, que não fazem parte do recurso de nuvem, mas que podem ser implantados em máquinas virtuais baseadas em IaaS para monitorar qualquer workload no Azure ou no data center na infraestrutura local.

Azure Monitor

O Azure Monitor é um recurso e ferramenta central que fornece funcionalidades completas de gerenciamento que permitem monitorar uma assinatura do Azure. Ele fornece recursos de gerenciamento para logs de atividades, logs de diagnóstico, métricas, Application Insights e Log Analytics. Ele deve ser tratado como um recurso de gerenciamento e painel para todos os outros recursos de monitoramento.

Azure Application Insights

O Azure Application Insights fornece recursos centralizados de métricas, logs e monitoramento na escala do Azure para aplicativos personalizados. Os aplicativos personalizados podem começar a enviar métricas, logs e outras informações de telemetria para o Azure Application Insights. Ele também fornece recursos de análise, painéis e relatórios para obter insights de dados de entrada e tomar medidas em relação a eles.

Azure Log Analytics

O Azure Log Analytics fornece processamento de logs centralizado e geração de insights e alertas deles. Logs de atividades, logs de diagnóstico, logs de aplicativos, logs de eventos e até mesmo logs personalizados podem enviar informações para o Log Analytics, que pode fornecer ainda mais recursos sofisticados de relatórios, painéis e análise para obter insights de dados de entrada e tomar medidas sobre eles.

Application Insights

Como o nome sugere, o Azure Application Insights fornece insights sobre o funcionamento de um aplicativo. Os insights relevantes para um aplicativo Web incluem o número de solicitações de entrada por segundo, solicitações com falha por segundo, uso de hardware em termos de utilização da CPU e disponibilidade de memória. O Application Insights fornece um painel, relatórios e gráficos para exibir várias métricas relacionadas ao aplicativo. Isso permite visualizar e entender melhor as tendências em relação ao uso do aplicativo, sua disponibilidade, o número de solicitações e muito mais, para realizar ações tanto preventivas quanto reativas sobre o aplicativo. As informações de tendências podem ser usadas para descobrir o que não está funcionando a favor do aplicativo e o que está funcionando corretamente durante um período específico.

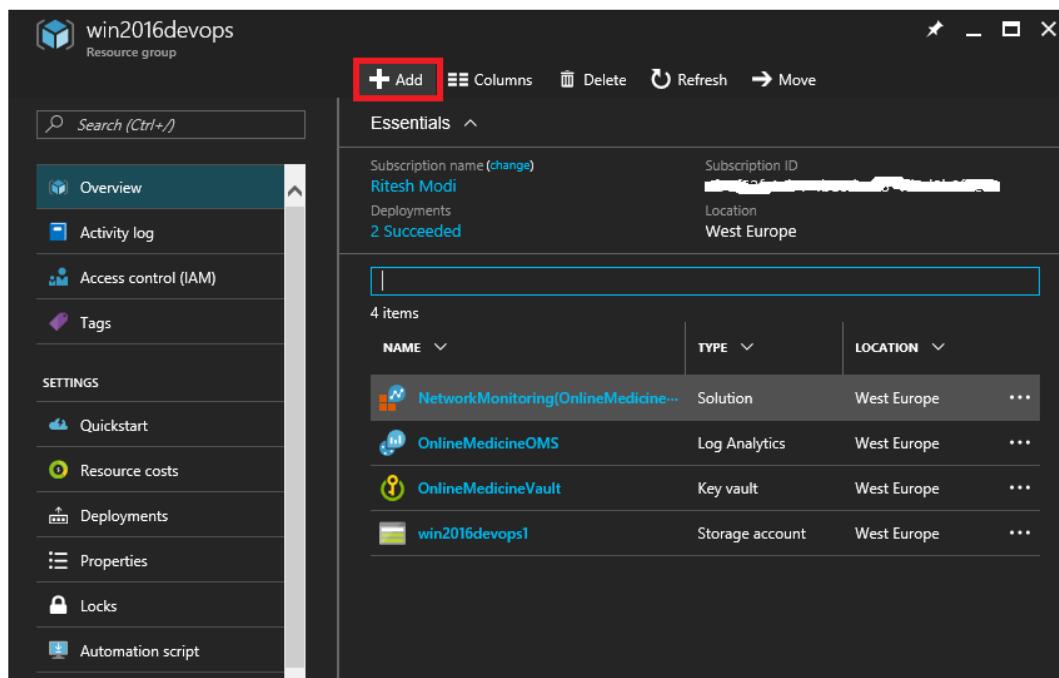
O primeiro passo para trabalhar com o Application Insights é provisionar esse serviço no Azure em um grupo de recursos que tenha todos os serviços gerais e de gerenciamento consumidos por todos os aplicativos. Se você se lembra, havíamos criado um grupo de recursos semelhante chamado Win2016devOps, que abriga todos os serviços comuns, como o Azure Key Vault, o Application Insights, o Operational Insights e o Armazenamento, para manter os scripts e modelos usados em todo ambiente e aplicativos.

Provisionamento

Como mencionado anteriormente, o primeiro passo no consumo de serviços do Application Insights é provisioná-lo no Azure.

O Application Insights pode ser provisionado manualmente usando o portal do Azure, as APIs REST do Azure, o PowerShell e os modelos do ARM. Aqui usaremos o portal do Azure:

1. Faça logon no portal do Azure usando as credenciais e navegue até um grupo de recursos existente ou crie um novo grupo de recursos. Clique no botão **Adicionar**:

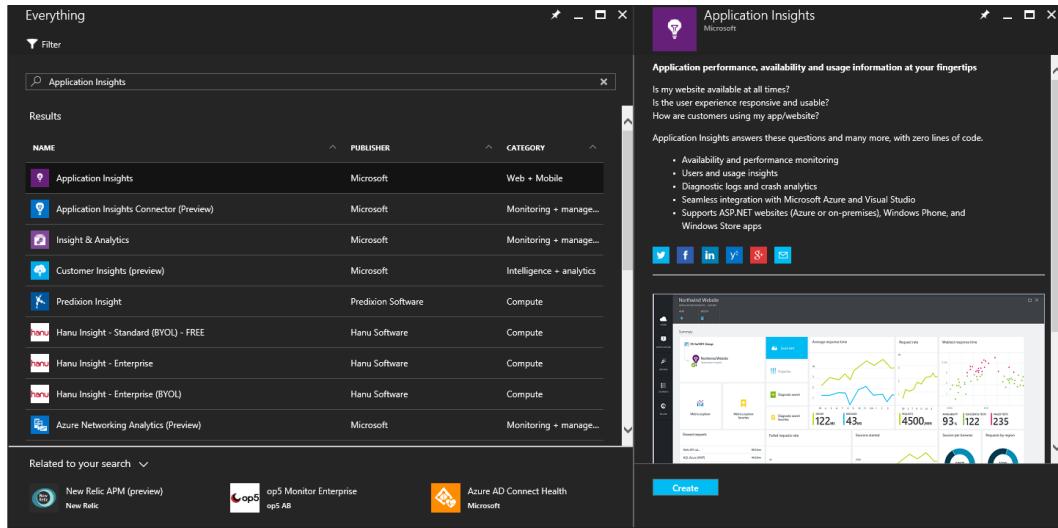


The screenshot shows the Azure Resource Group 'win2016devops' interface. On the left, there's a sidebar with navigation links: Overview, Activity log, Access control (IAM), Tags, SETTINGS, Quickstart, Resource costs, Deployments, Properties, Locks, and Automation script. The main area is titled 'Essentials'. It displays subscription information: Subscription name (Ritesh Modi), Subscription ID (redacted), and 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 redboxed, is located at the top of the main content area.

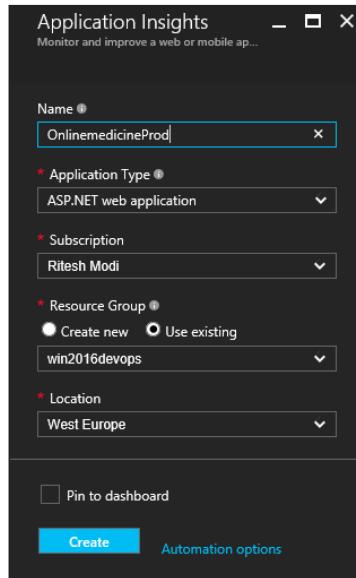
NAME	TYPE	LOCATION	...
NetworkMonitoring(OnlineMedicine...)	Solution	West Europe	...
OnlineMedicineOMS	Log Analytics	West Europe	...
OnlineMedicineVault	Key vault	West Europe	...
win2016devops1	Storage account	West Europe	...

Segurança e monitoramento

2. Digite Application Insights na caixa de pesquisa da próxima folha. O primeiro link deve se referir ao Application Insights. Clique nele para criar uma nova instância de serviço do Application Insights. Clique no botão Criar para começar:



3. A próxima folha solicitará o nome da instância de serviço do Application Insights, o tipo de aplicativo, a assinatura, o grupo de recursos e o local do serviço. Insira os detalhes adequados e clique no botão Criar. Isso provisionará o serviço:



Name

* Application Type

* Subscription

* Resource Group Create new Use existing

* Location

Pin to dashboard

4. Agora, navegue até o serviço que mostra as propriedades essenciais, como sua **chave de instrumentação**, conforme destacado na captura de tela a seguir. A chave será diferente em cada instância e, geralmente, copiada e usada no Visual Studio. Algumas das informações foram ocultas em preto por motivos de segurança:

The screenshot shows the Azure Application Insights interface 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 displays the 'Essentials' section with details about the resource group ('win2016devops'), location ('West Europe'), subscription name ('Ritesh Modi'), and type ('ASP.NET'). The 'Instrumentation Key' field is explicitly highlighted with a red rectangular box. Below this, there are sections for Alerts (0), Live Stream (Click to configure), Smart Detection (ONLINEMEDICINEPROD, 0 detections last 24h), Web tests (0), and App map. The 'Health' section includes 'Overview timeline' for 'ONLINEMEDICINEPROD' and two charts for 'SERVER RESPONSE TIME' and 'PAGE VIEW LOAD TIME', both showing data from 0ms to 100ms.

Log Analytics

O serviço do Log Analytics ajuda a fornecer informações sobre esses ambientes usando seus recursos avançados de análise e armazenamento de log. O Log Analytics é usado para capturar logs de várias origens, incluindo aplicativos. No entanto, é igualmente importante monitorar o ambiente no qual um aplicativo está hospedado e em execução. Isso envolve a infraestrutura, como as máquinas virtuais, os contêineres do Docker e seus componentes relacionados.

Provisionamento

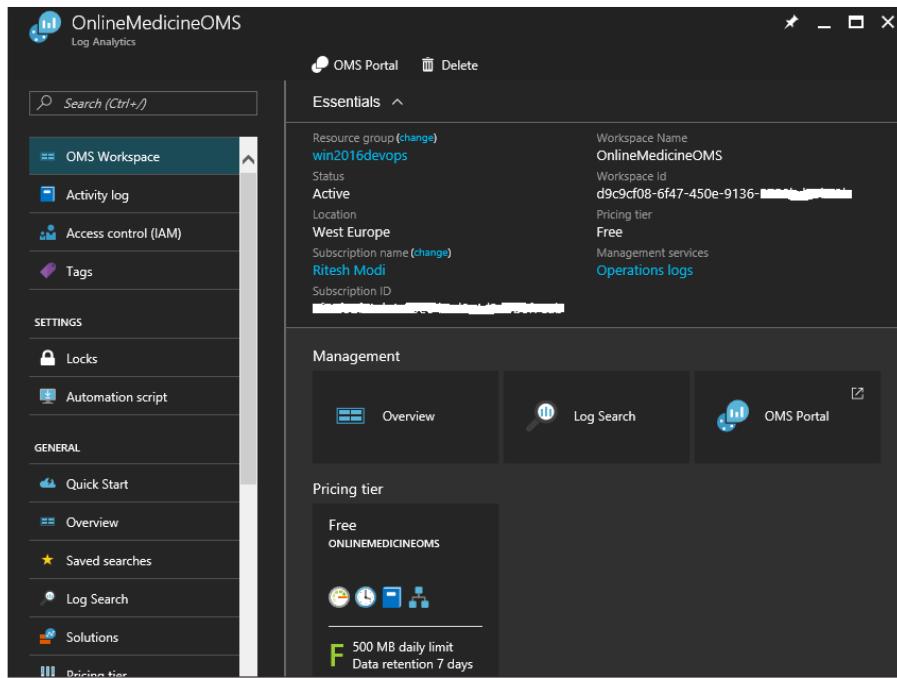
O Log Analytics deve ser provisionado no Azure antes de poder ser usado para monitorar máquinas virtuais e contêineres. Novamente, de maneira semelhante ao Application Insights, o Log Analytics pode ser provisionado por meio do portal do Azure, do PowerShell, das APIs REST e dos modelo do gerenciador de grupo de recursos. Um espaço de trabalho do Log Analytics é um limite de segurança que pode ser acessado por determinados usuários. Vários espaços de trabalho devem ser criados para isolar usuários e seu acesso correspondente aos dados de telemetria do ambiente.

O script JSON usado para provisionar um espaço de trabalho do Log Analytics é mostrado aqui:

```
{  
    "apiVersion": "2015-11-01-preview",  
    "type": "Microsoft.OperationalInsights/workspaces",  
    "name": "[parameters('workspaceName')]",  
    "location": "[parameters('deployLocation')]",  
    "properties": {  
        "sku": {  
            "Name": "[parameters('serviceTier')]"  
        }  
    }  
}
```

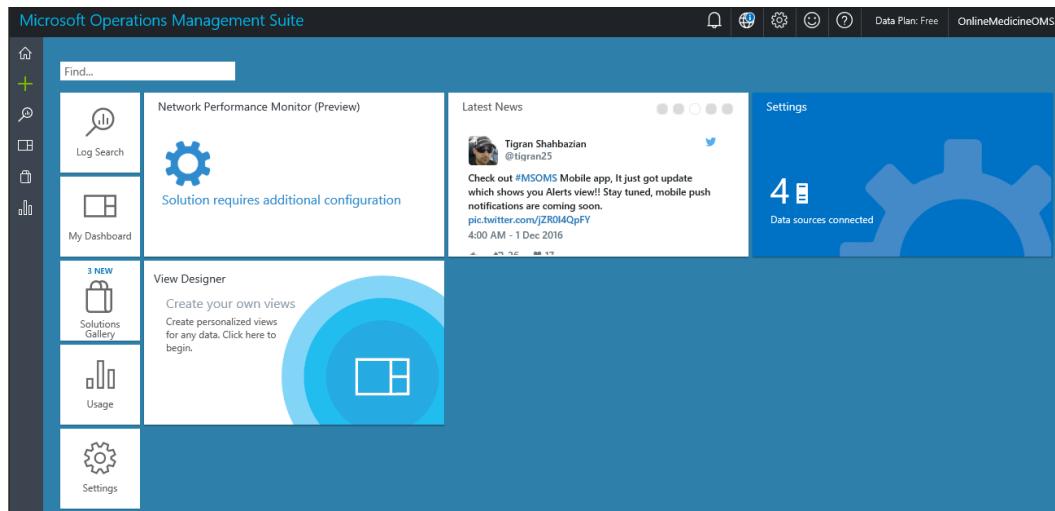
As informações de nome, local e SKU são necessárias para configurar um espaço de trabalho, e os valores para eles são fornecidos com o uso de parâmetros.

Veja na captura de tela a seguir o espaço de trabalho após o provisionamento:



Clique na seção Portal do Log Analytics para abrir o portal do espaço de trabalho. Esse portal é usado para exibir todas as informações de telemetria capturadas pelo Operational Insights, configurar o Operational Insights e fornecer funcionalidades e recursos de painel.

A tela inicial do Log Analytics é mostrada aqui:



A seção **Soluções** mostra que duas soluções foram implantadas. Pode ser adotada uma estratégia diferente de ter um espaço separado para cada ambiente, e cabe aos leitores decidir o melhor para seus aplicativos e soluções. O Log Analytics pode ser configurado usando o bloco de **Configurações**:

A screenshot of the Microsoft Operations Management Suite interface. The top navigation bar shows 'Microsoft Operations Management Suite'. Below it, the 'Overview' and 'Settings' tabs are visible. On the left, there's a sidebar with icons for Home, Add, Search, Connected Sources, Data, Computer Groups, Accounts, Alerts, and Preview Features. The main content area is titled 'Installed Solutions: 2'. It lists 'Log Search' and 'Network Performance Monitor (Preview)'. Each solution has a brief description and a 'Remove' button.

Agentes do Log Analytics

Você pode ter notado que não há nenhuma alteração de código necessária no nível do aplicativo para utilizar o Log Analytics. O Log Analytics depende da instalação de um agente em máquinas virtuais. Esses agentes coletam dados de telemetria e os enviam para o espaço de trabalho do Log Analytics, onde são armazenados por um período de tempo especificado, dependendo da SKU escolhida. Esses agentes podem ser instalados manualmente em máquinas virtuais ou automaticamente usando os modelos ARM imediatamente após o provisionamento das máquinas virtuais. O código JSON para o provisionamento de um agente em uma máquina virtual é:

```
{  
    "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]"  
    }  
}  
}
```

Os elementos `workspaceId` e `accountid` estão disponíveis no bloco de **Configurações** do espaço de trabalho do Log Analytics, e o elemento de cópia é usado para implantá-lo em várias máquinas virtuais. Esse é um recurso filho do recurso de máquina virtual, garantindo que essa extensão seja executada sempre que uma máquina virtual for provisionada ou atualizada.

Segurança e monitoramento

Veja a seguir a configuração relacionada ao ID do espaço de trabalho e da conta. A chave primária é usada como um ID da conta para configurar os agentes usando modelos do ARM:

The screenshot shows the Microsoft Operations Management Suite interface. On the left, there's a navigation sidebar with options like Overview, Settings, Solutions, Connected Sources (which is highlighted), Data, Computer Groups, Accounts, Alerts, and Preview Features. The main content area is titled 'Windows Servers' and shows a summary: 'Windows Servers' (Attach any Windows server or client.), '4 WINDOWS COMPUTERS CONNECTED', and download links for 'Windows Agent (64 bit)' and 'Windows Agent (32 bit)'. It also includes fields for 'WORKSPACE ID' and 'PRIMARY KEY' with a 'Regenerate' button, and a 'SECONDARY KEY' field with a 'Regenerate' button. At the bottom, there's information about the 'OMS Gateway' and a link to 'Download OMS Gateway'.

Pesquisar

O espaço de trabalho do Log Analytics fornece recursos de pesquisa para pesquisar entradas de log específicas, exportar todos os dados de telemetria para o Excel e/ou o Power BI e linguagem de consulta de pesquisa semelhante ao SQL.

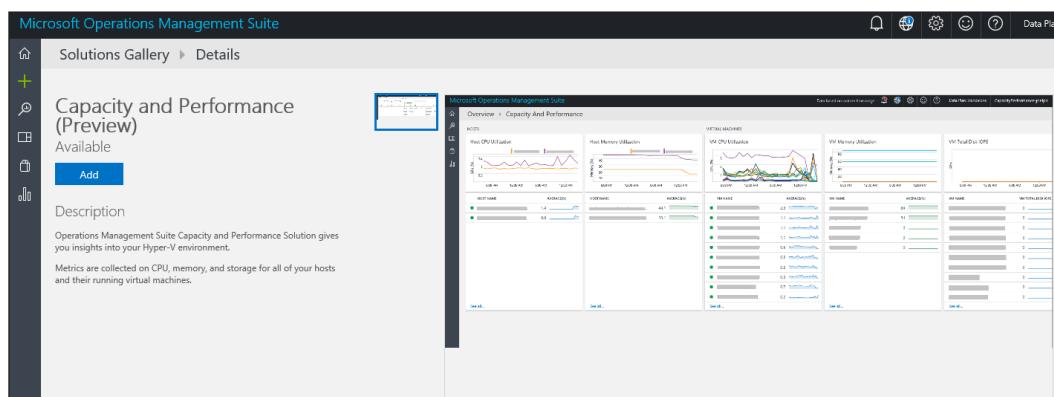
Veja a seguir a tela Pesquisa de logs:

The screenshot shows the Microsoft Operations Management Suite Log Search interface. The top bar includes a search bar with placeholder 'Begin searching here...', a magnifying glass icon, and a user profile 'OnlineMedicineOMS'. Below the search bar, there's a section titled 'Let's start searching!' with a query editor containing the text: 'Type=Perf CounterName=% Processor Time' | measure avg(CounterValue) by Computer interval 30MINUTE'. This query is broken down into four parts: 'Pull in all performance data for processor utilization' (Required Data Collection: Performance Counters), 'Perform a function on returned results', 'Get the average processor utilization grouped by individual computer', and 'Average over 30 minute periods'. To the left of the search bar, there are 'Favorites' and 'History' buttons. Below the search bar, there's a section titled 'A few more queries to try' with various pre-defined queries like 'All collected data', 'Count of all data collected grouped by Type', 'All data collected grouped by 1HOUR interval', and several security-related queries such as 'All Configuration Changes', 'All computers with missing critical or security updates', 'Computers with guest account logons', 'Computers with Available memory more than 2Gb', 'SQL Assessment recommendations by Computer', and 'All syslog data grouped by Facility'. To the right of 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'.

Soluções

As soluções no Log Analytics são recursos adicionais que podem ser adicionados ao espaço de trabalho com a captura de dados de telemetria adicionais que não são capturados por padrão. Quando essas soluções são adicionadas ao espaço de trabalho, Management Packs adequados são enviados para todos os agentes conectados ao espaço de trabalho no contexto de configurarem-se para a captura de dados específicos da solução de máquinas virtuais e contêineres e, em seguida, começar enviá-los para o espaço de trabalho do Log Analytics.

A captura de tela a seguir mostra a galeria da soluções e a solução de capacidade e performance no espaço de trabalho do Log Analytics. Ao clicar em qualquer solução e, em seguida, clicar no botão **Adicionar**, a solução será adicionada ao espaço de trabalho da seguinte maneira:

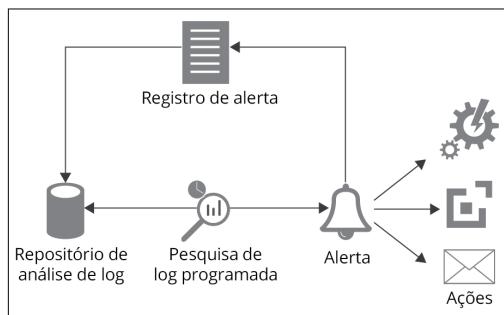


O Azure fornece muitas soluções do Log Analytics para controle e monitoramento de diferentes aspectos de ambientes e aplicativos. No mínimo, um conjunto de soluções que são genéricas e aplicáveis a praticamente qualquer ambiente deve ser adicionado ao espaço de trabalho:

- Capacidade e performance
- Integridade do agente
- Controle de alterações
- Contêineres
- Segurança e auditoria
- Gerenciamento de atualizações
- Monitoramento de performance de rede

Alertas

O Log Analytics permite gerar alertas sobre os dados ingeridos. Ele faz isso por meio da execução de uma consulta predefinida composta de condições sobre os dados de entrada. Se for encontrado um registro que se encaixe no âmbito dos resultados da consulta, será gerado um alerta. A Log Analytics fornece um ambiente altamente configurável para determinar as condições para a geração de alertas, janelas de tempo em que a consulta deve retornar os registros, janelas de tempo em que a consulta deve ser executada e a ação a ser realizada quando a consulta retorna resultados como alertas:



O primeiro passo para configurar um alerta é criar uma pesquisa salva. Uma pesquisa salva é simplesmente uma consulta de pesquisa em relação ao Log Analytics:

A captura de tela mostra a interface de usuário do Log Search no Log Analytics. No topo, há uma barra com ícones para Export, PowerBI, Alert, Save, Favorites, History e Analytics. Abaixo, uma barra de filtro mostra "Data based on last 1 day" e uma escala de tempo de 1 hora. O resultado da consulta é exibido na central, mostrando uma única linha de resultado com detalhes como TimeGenerated, Computer e EventLevelName. À direita, uma caixa de diálogo "Save Search" está aberta, permitindo ao usuário definir o nome ("AllEvents"), categoria e outras opções. Botões para "Save" e "Cancel" estão na base da caixa.

Salve a consulta atribuindo um nome a ela. Após salvar a consulta, clique no botão **Alerta** no menu **Pesquisa de logs**. Ele fornece ao usuário uma interface para definir e adicionar uma nova regra de alerta:

Log Search > Add Alert Rule

General

Alert information

Name: eventsalert

Description: this alert is raised whenever events arrives

Severity: Critical

Search query: Use current search query

search * | where (Type == "Event")

Time window: 15 Minutes

This search returned:

Schedule

Alert frequency: Check for this alert every 15 Minutes

Generate alert based on: Number of results (Metric measurement)

Number of results: Greater than 2

Suppress alerts: When checked, allows you to set an amount of time to wait before alerting again to reduce alert noise

Actions

- Email notification: Yes
- Webhook: No
- Runbook: Yes
- Automation account: datacenterautomation
- Select a runbook: TestRunbook
- Run on: Azure Hybrid worker
- ITSM Actions: Yes

Save Cancel

Nessa tela única, todas as configurações relacionadas a uma regra de alerta podem ser executadas. Você precisa fornecer o **Nome**, a **Descrição**, a **Gravidade** e a consulta a ser executada como parte da avaliação de regra em seus respectivos campos.

A janela de tempo especifica o intervalo de dados no qual a consulta deve ser executada. Na captura de tela, é possível ver que sempre que a regra é executada, ela processa dados dos últimos 15 minutos.

A seção de programação ajuda você a configurar a frequência de execução da regra. Ela responde à pergunta "*Com que frequência a consulta deve ser realizada?*" Na captura de tela anterior, a regra é executada a cada 15 minutos. A janela de tempo deve ser maior do que a **frequência do alerta**. Os alertas podem ser configurados ainda mais com base no número de resultados encontrados. Não é necessário que um alerta seja gerado para cada instância de dados encontrados com base na consulta. Esses dados podem ser acumulados, e os cálculos realizados nesses grupos de dados podem ser usados para gerar alertas. Os alertas também podem ser gerados com base em medições métricas de recursos. As métricas geralmente são associadas aos dados de performance fornecidos por um recurso, como utilização da CPU, uso da rede e disponibilidade de memória. Você também pode criar um intervalo de tempo que deve decorrer antes da execução da ação. Nesse caso, são gerados alertas, mas a ação é executada somente depois de decorrido o intervalo configurado.

A seção **Ações** nos permite configurar coisas que devem seguir um alerta. Em geral, deve haver uma ação corretiva e/ou de notificação. O Log Analytics fornece quatro maneiras diferentes de criar uma nova ação. Você pode combiná-las como quiser. Um alerta executará toda e qualquer uma das seguintes ações configuradas:

- **Notificação por e-mail:** envia um e-mail para os destinatários configurados:

The dialog box is titled "Actions". It contains a section for "Email notification" with a checked checkbox. Below it are fields for "Subject" (labeled "Email subject") and "Recipients (semi-colon separated)" (containing "callritz@hotmail.com"). At the bottom are "Yes" and "No" buttons.

- **Webhook:** executa um processo externo arbitrário usando um mecanismo de HTTP POST. Por exemplo, uma API REST pode ser executada ou o Service Manager/APIs ServiceNow podem ser invocados para criar um tíquete:

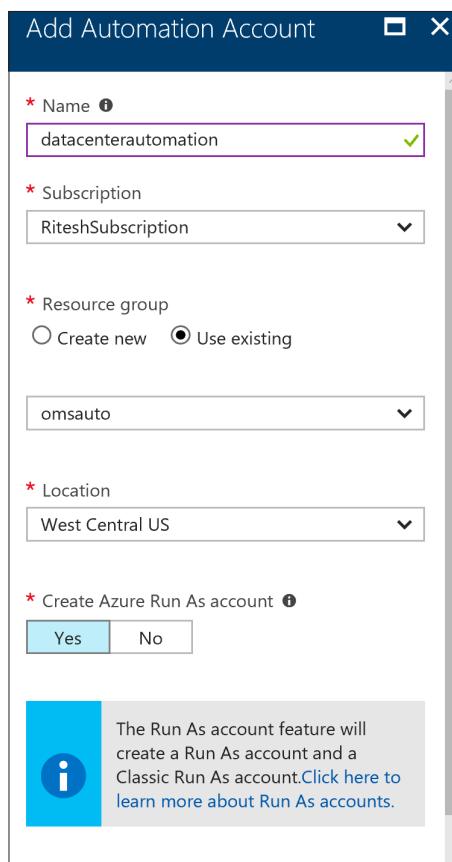
The dialog box is titled "Webhook". It has a "Yes" button. Below it is a "Webhook URL" field containing "Invoke webhook when alert fires". A checked checkbox "Include custom JSON payload" is followed by a text area for "Enter your custom JSON payload" containing sample code. A red message "Url is invalid" is displayed above a "Test webhook" button.

- **Runbooks:** essa ação executa runbooks de Automação do Azure. Na próxima seção, veremos todo o processo de execução de um runbook de Automação do Azure.
- **Ações ITSM:** soluções ITSM devem ser provisionadas antes do uso dessa opção. Elas ajudam a conectar e enviar informações para sistemas ITSM.

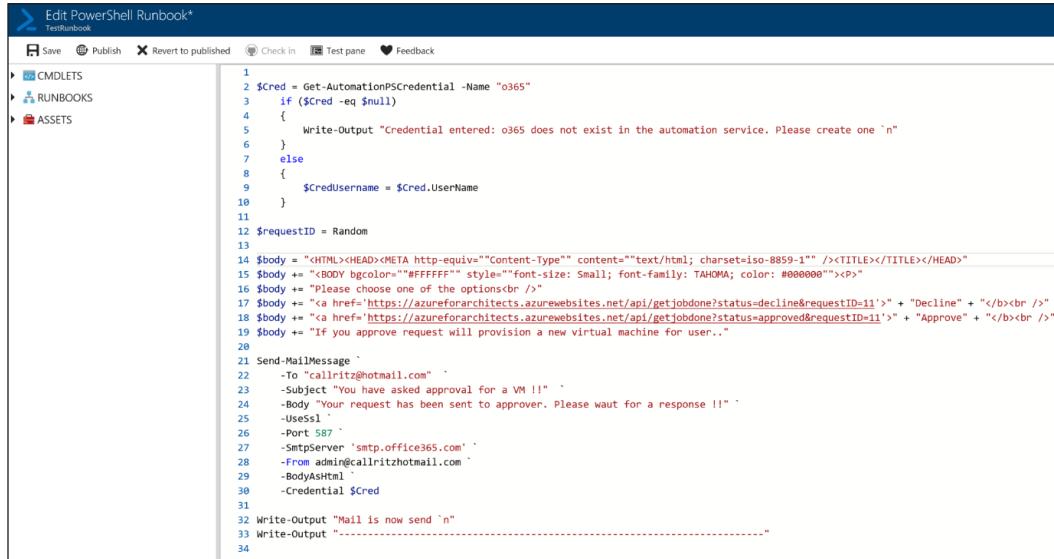
Execução de runbooks em alertas

Uma das ações proporcionadas por um alerta do Log Analytics é a execução de um runbook de Automação do Azure. Essa facilidade de executar runbooks após um alerta permite tomar medidas em relação ao alerta para fazer correções e informar os stakeholders relevantes usando notificações.

1. A primeira etapa na execução de um runbook em resposta a um alerta é criar uma conta de Automação do Azure:

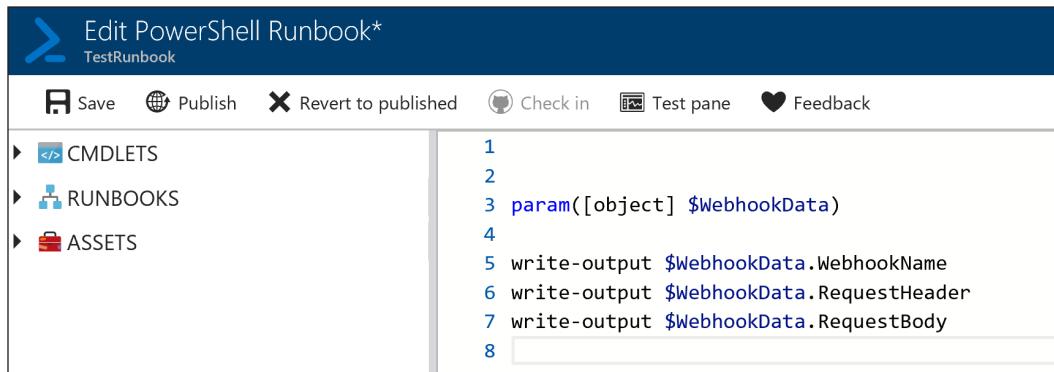


- Depois que a conta tiver sido criada, crie um runbook, conforme mostrado na captura de tela a seguir, para provar que ele pode ser executado como parte da geração de alertas. Nesse caso, o runbook envia um e-mail como parte da notificação. Ele usa as credenciais da Automação do Azure para enviar um e-mail usando o servidor SMTP O365. Os usuários devem ter uma conta válida do O365 antes de enviar um e-mail usando a Automação do Azure:



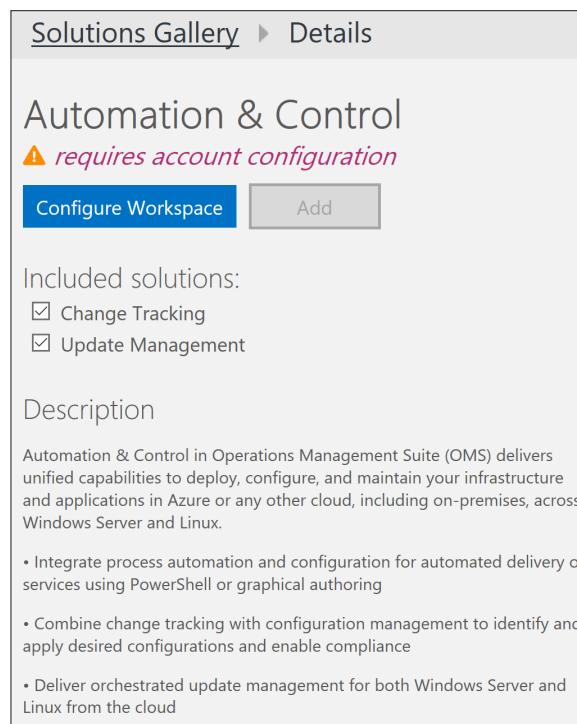
```
1 $Cred = Get-AutomationPSCredential -Name "O365"
2 if ($Cred -eq $null)
3 {
4     Write-Output "Credential entered: O365 does not exist in the automation service. Please create one `n"
5 }
6 else
7 {
8     $CredUsername = $Cred.UserName
9 }
10
11 $requestID = Random
12
13 $body = "<HTML><HEAD><META http-equiv=""Content-Type"" content=""text/html; charset=iso-8859-1"" /><TITLE></TITLE></HEAD>
14 <BODY bgcolor=""#FFFFFF"" style=""font-size: Small; font-family: TAHOMA; color: #000000""><p>
15 $body += "Please choose one of the options<br />
16 $body += "<a href=""https://azureforarchitects.azurewebsites.net/api/getjobdone?status=decline&requestID=11"">" + "Decline" + "</a><br />
17 $body += "<a href=""https://azureforarchitects.azurewebsites.net/api/getjobdone?status=approved&requestID=11"">" + "Approve" + "</a><br />
18 $body += "If you approve request will provision a new virtual machine for user.."
19
20 Send-MailMessage `
21     -To "callritz@hotmail.com" `
22     -Subject "You have asked approval for a VM !!" `
23     -Body "Your request has been sent to approver. Please wait for a response !!!" `
24     -UseSsl `
25     -Port 587 `
26     -SmtpServer 'smtp.office365.com' `
27     -From admin@callritzhotmail.com `
28     -BodyAsHtml `
29     -Credential $Cred
30
31 Write-Output "Mail is now send `n"
32 Write-Output "-----"
33 Write-Output "-----"
34
```

- Lembre-se de que isso é apenas uma demonstração. O runbook também pode aceitar parâmetros e alertas do Log Analytics e enviar um único parâmetro do tipo objeto. Esse parâmetro contém todos os dados referentes a origem do alerta, detalhes sobre o alerta e informações que estão disponíveis com o Log Analytics:

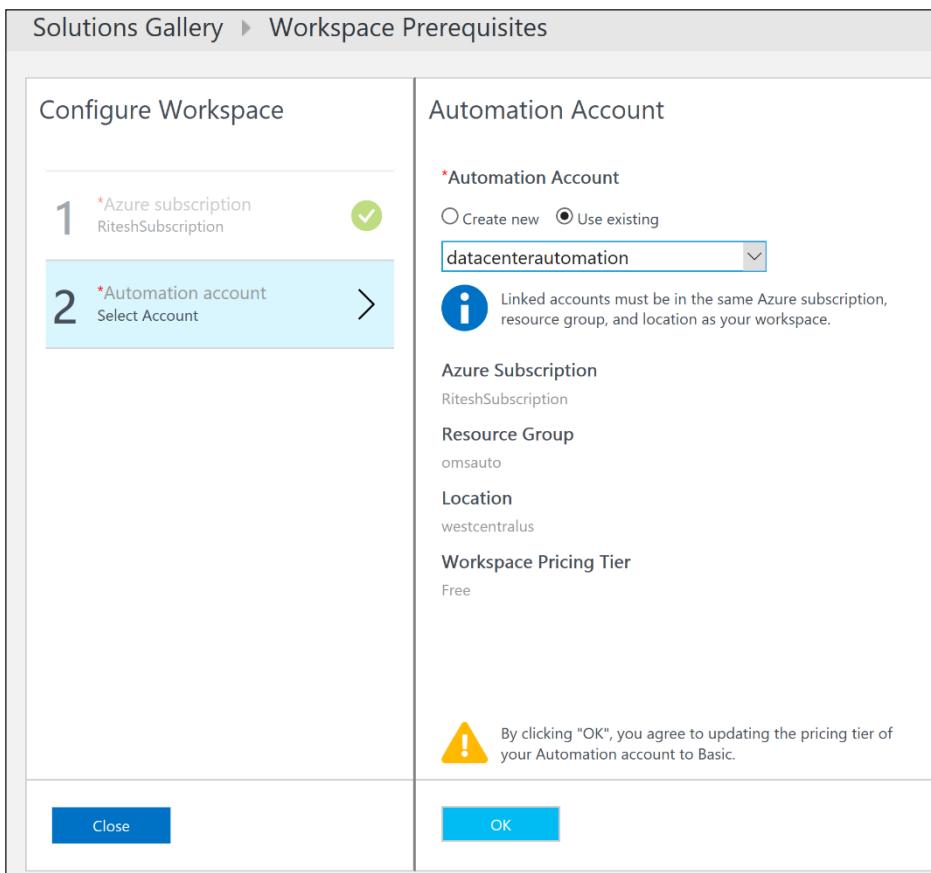


```
1
2
3 param([object] $WebhookData)
4
5 write-output $WebhookData.WebhookName
6 write-output $WebhookData.RequestHeader
7 write-output $WebhookData.RequestBody
8
```

4. Os dados estão em um formato JSON, e um cmdlet `ConvertFrom-JSON` pode ser usado para criar objetos do PowerShell.
5. O próximo passo é definir uma configuração do Log Analytics para que ele possa se conectar à conta de Automação do Azure. Para isso, uma solução de **Automação e controle** precisa ser habilitada e implantada.
6. Se você clicar nesse bloco, acessará a janela de configuração **Galeria de soluções**. Clique em **Configurar espaço de trabalho** para implantá-la:



7. Selecione a **conta da Automação** do Azure recém-criada como parte da implantação da solução:



8. Após a implantação da solução, navegue até a janela **Configurações** no espaço de trabalho do Log Analytics e certifique-se de que as configurações da Automação do Azure mostrem detalhes sobre a **conta de Automação** do Azure, conforme mostrado na captura de tela a seguir. Isso garante que o espaço de trabalho do Log Analytics esteja conectado à **conta da Automação** do Azure:

The screenshot shows the Azure Portal's 'Settings' page under 'Overview'. On the left, there is a sidebar with various 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 sections for 'Workspace Information', 'Manage Users', 'Azure Subscription & Data Plan', and 'Automation Account' (also highlighted in blue). At the top, a message says 'You can now view and edit settings in Azure Portal.' Below the sidebar, there are details about the subscription: 'Subscription ID: 9755ffce-e94b-4332-9be8-1ade15e78909', 'Resource Group Name: omsauto', and 'Automation Account Name: datacenterautomation'.

9. Agora, o runbook deve estar disponível ao configurar o runbook de ação de alerta, conforme mostrado na captura de tela a seguir:

The screenshot shows a modal dialog titled 'Runbook'. It has two buttons at the top: 'Yes' (highlighted in blue) and 'No'. Below the buttons, it says 'Automation account' followed by the name 'datacenterautomation'. Underneath that, it says 'Select a runbook' and shows a dropdown menu with 'TestRunbook' selected. At the bottom, it says 'Run on' with two options: 'Azure' (highlighted in blue) and 'Hybrid worker'.

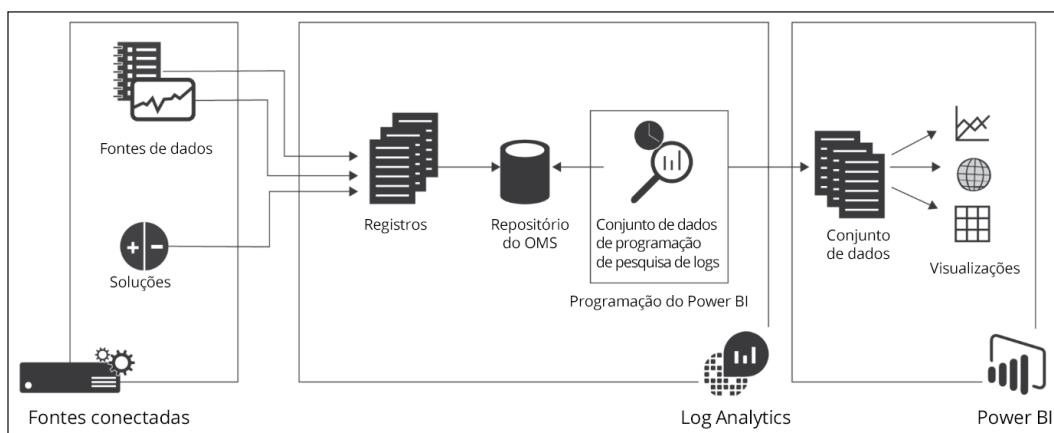
Integração ao Power BI

Coletar de dados e armazená-los em um repositório central é um aspecto importante. No entanto, deve haver ferramentas e utilitários para processar os dados e gerar insights deles. O Power BI é um serviço fornecido pela Microsoft criado especificamente para visualizar e gerar insights a partir de dados brutos.

O Power BI pode ser habilitado usando o menu **Configurações**, assim como a configuração para a Automação do Azure. A conexão ao Power BI deve ser feita no menu **Configurações**. Quando a conexão for estabelecida, ela poderá ser usado para enviar dados do Log Analytics para o Power BI.

O Log Analytics proporciona duas maneiras diferentes de interagir com o Power BI. Primeiro, ele precisa ser habilitado no menu **Configurações**.

Assim como os alertas, a opção de menu do Power BI está disponível no menu de nível superior da pesquisa de logs. Quando você clica nela, nós podemos configurar a conectividade do Power BI. Um agendador é executado periodicamente para realizar consultas de pesquisa e enviar os dados resultantes para o Power BI. Os dados são armazenados como conjuntos de dados no Power BI e podem ser usados para gerar gráficos, relatórios e painéis, conforme mostrado no diagrama a seguir:



A outra maneira de enviar dados ao Power BI a partir do Log Analytics é usar a linguagem **Power Query** no Power BI.

A **linguagem de fórmulas do Power Query (Linguagem M)** pode ser usada com o Power Query no Microsoft Excel e no Power BI Desktop.

Para usar o Power BI Desktop, siga as próximas instruções para fazer download e gerar relatórios usando consultas:

1. Faça download do Power BI Desktop em <https://powerbi.microsoft.com/en-us/desktop/>.
2. No Power BI Desktop, selecione **Obter dados | Consulta em branco | Editor de consulta avançada**.

3. Cole o seguinte script de Linguagem M no **Editor de consulta avançada** e selecione **Concluído**. Isso resultará na execução da consulta e trará os dados do Log Analytics para o Power BI. Forneça um nome e adicione um novo conjunto de dados clicando no botão **Aplicar e Fechar** no Editor de Consultas do 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[columns]),
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
```

Resumo

A segurança é sempre um aspecto importante de qualquer implantação e solução. Ela se tornou muito mais importante e relevante devido à implantação na nuvem. Além disso, há uma ameaça crescente de ataques cibernéticos. Em tais circunstâncias, a segurança tornou-se um ponto focal para as organizações. Não importa o tipo de implantação ou de solução -- seja IaaS, PaaS ou SaaS, a segurança é necessária em todas elas. Os data centers do Azure são completamente seguros e têm diversas certificações internacionais de segurança. Elas são seguras por padrão. Elas fornecem recursos de segurança IaaS como NSGs, tradução de endereço de rede, pontos de extremidade seguros, certificados, cofres de chaves, criptografia de armazenamento e de máquina virtual e recursos de segurança de PaaS para recursos individuais de PaaS. A segurança tem seu próprio ciclo de vida completo, e ele deve ser adequadamente planejado, projetado, implementado e testado, assim como qualquer outra funcionalidade de aplicativo.

O monitoramento é um aspecto de arquitetura importante de qualquer solução. Ele também é o primeiro passo em direção à capacidade de auditoria. Ele permite que as operações gerenciem uma solução, tanto reativamente quanto proativamente. Ele oferece os registros necessários para solução e correção de problemas que podem surgir em plataformas e aplicativos. Há muitos recursos no Azure que são específicos para implementar o monitoramento no Azure, outras nuvens e data centers na infraestrutura local. O Application Insights e o Log Analytics são alguns dos recursos importantes para isso. Evidentemente, é uma obrigatoriedade melhorar suas soluções e produtos ao inovar com base em insights de dados de monitoramento.

No próximo capítulo, veremos um aspecto completamente separado do Azure relacionado ao provisionamento de recursos usando modelos do Azure Resource Manager.

4

Implantações entre assinaturas usando modelos do ARM

Os **modelos do Azure Resource Manager (ARM)** são o mecanismo preferencial para provisionar recursos e configurá-los no Azure.

Os modelos do ARM ajudam a implementar um paradigma relativamente novo conhecido como **infraestrutura como código** (IaC). Os modelos do ARM convertem a infraestrutura e sua configuração em código. A conversão da infraestrutura em código tem inúmeras vantagens. A IaC traz um alto nível de consistência e previsibilidade em implantações entre ambientes. Ela também garante que os ambientes possam ser testados antes de ir para a produção e, por fim, proporciona um alto nível de confiança no processo de implantação, manutenção e governança.

Os tópicos a seguir serão abordados neste capítulo:

- Modelos do ARM
- Implantar grupos de recursos com modelos do ARM
- Implantar recursos em assinaturas e grupos de recursos
- Implantar entre assinaturas e grupos de recursos usando modelos vinculados

Modelos do ARM

Uma vantagem proeminente da IaC é que ela pode ter controle de versão. Ela pode ser reutilizada entre ambientes, o que proporciona um alto nível de consistência e previsibilidade em implantações. Ela também garante que o impacto e o resultado da implantação de um modelo do ARM sejam os mesmos, independentemente do número de vezes que o modelo é implantado. Esse recurso é conhecido como um **modelo idempotente**.

Os modelos do ARM foram apresentados com a introdução da especificação do ARM. Desde então, receberam mais recursos e ficaram mais maduros. É importante compreender que geralmente há uma lacuna que dura de algumas semanas a alguns meses entre a configuração de recurso real e a disponibilidade dessa configuração em modelos do ARM.

O recurso tem sua própria configuração. Essa configuração pode ser afetada de diversas maneiras, inclusive usando o Azure PowerShell, a CLI do Azure, os SDKs do Azure, a API REST e os modelos do ARM.

Cada uma dessas maneiras tem seu próprio ciclo de vida de desenvolvimento e liberação, que é diferente do desenvolvimento real de recursos. Vamos tentar entender isso com a ajuda de um exemplo.

O recurso do Azure Databricks tem seu próprio ciclo de vida de cadência e desenvolvimento. Os consumidores desse recurso têm seu próprio ciclo de vida de desenvolvimento, que é diferente do desenvolvimento real dos recursos. Se o Databricks receber sua primeira versão em 31 de dezembro, os cmdlets do Azure PowerShell para ele podem não estar disponíveis na mesma data e podem ser lançados em 31 de janeiro do próximo ano. De maneira semelhante, a disponibilidade desses recursos nos modelos do ARM e da API REST pode ocorrer em torno de 15 de janeiro.

Os modelos do ARM são documentos baseados em JSON que, quando executados, invocam a API REST no plano de gerenciamento do Azure e enviam o documento inteiro para ele. A API REST tem seu próprio ciclo de vida de desenvolvimento, e o esquema JSON para o recurso tem seu próprio ciclo de vida.

Isso significa que o desenvolvimento do Azure de uma função dentro de um recurso deve acontecer em pelo menos três componentes diferentes antes que eles possam ser consumidos a partir de modelos do ARM. Entre eles estão:

- O próprio recurso
- A API REST do recurso
- O esquema de recursos do modelo do ARM

Cada recurso no modelo do ARM tem a propriedade `apiVersion`. Essa propriedade ajuda a decidir qual versão da API REST deve ser usada para provisionar e implantar o recurso. O próximo diagrama mostra o fluxo de solicitações do modelo do ARM para APIs de recursos que são responsáveis pela criação, atualização e exclusão de recursos:



Uma configuração de recurso, como uma conta de armazenamento no modelo do ARM, tem a seguinte aparência:

```

{
  "type": "Microsoft.Storage/storageAccounts",
  "apiVersion": "2017-06-01",
  "name": "[variables('storage2')]",
  "location": "[resourceGroup().location]",
  "kind": "Storage",
  "sku": {
    "name": "Standard_LRS"
  }
}
  
```

Nesta listagem de código, a disponibilidade desse esquema para definir o SKU se baseia no desenvolvimento do esquema de modelo do ARM. A disponibilidade da API REST e seu número de versão é determinada pela `apiVersion`, que é 2017-06-01. O recurso real é determinado pela propriedade `Type`, que tem as duas partes a seguir:

- **Namespace do provedor de recursos:** os recursos no Azure são hospedados em namespaces, e os recursos relacionados são hospedados no mesmo namespace.
- **Tipo de recurso:** os recursos são mencionados usando o nome do tipo.

Nesse caso, o recurso é identificado pelo nome e tipo de provedor, que é `Microsoft.Storage/storageaccounts`.

Até recentemente, os modelos do ARM esperavam que os grupos de recursos estivessem disponíveis antes da implantação. Eles também eram limitados à implantação em um único grupo de recursos em uma única assinatura.

Isso significa que, até recentemente, um modelo do ARM poderia implantar todos os recursos em um único grupo de recursos. Essa foi a prática por anos, até que o Azure anunciou recentemente que um único modelo do ARM poderia ser usado para implantar recursos em vários grupos na mesma assinatura de várias assinaturas simultaneamente. Agora é possível criar grupos de recursos como parte dos modelos do ARM. Portanto, agora é possível implantar recursos de várias regiões em grupos de recursos diferentes.

Por que precisamos criar grupos de recursos a partir de modelos do ARM e ter implantações entre assinaturas e grupos de recursos simultaneamente?

Para apreciar o valor da criação de uma implantação entre assinaturas e grupos de recursos, precisamos entender como as implantações foram realizadas antes que esses recursos fossem disponibilizados.

Para implantar um modelo do ARM, um grupo de recursos é um pré-requisito. Os grupos de recursos devem ser criados antes da implantação de um modelo. Os desenvolvedores usam o PowerShell, a CLI do Azure ou a API REST para criar grupos de recursos e, em seguida, iniciar a implantação de modelos do ARM. Isso significa que qualquer implantação de ponta a ponta consiste em várias etapas. A primeira etapa é a provisão do grupo de recursos, e a próxima é a implantação do modelo do ARM no grupo de recursos recém-criado. Essas etapas podem ser executadas usando um único script do PowerShell ou etapas individuais da linha de comando do PowerShell. O script do PowerShell deve implementar o tratamento de exceções, o código de compensação, a reversão para um valor mínimo até que esteja pronto para a empresa. É importante observar que os grupos de recursos podem ser excluídos do Azure e, na próxima vez que o script for executado, a disponibilidade deles pode ser esperada. Haverá falha porque ele poderá assumir que o grupo de recursos existe. Resumindo, a implantação do modelo do ARM em um grupo de recursos deve ser uma etapa atômica, e não várias etapas.

Compare isso com a capacidade de criar grupos de recursos e seus recursos constituintes juntos nos mesmos modelos do ARM. Sempre que você implanta o modelo, ele cria grupos de recursos, se eles ainda não existirem, e continua a implantar recursos para eles após a criação.

Veremos também como esses novos recursos podem ajudar a remover algumas das restrições técnicas relacionadas a sites de recuperação de desastres.

Antes desses recursos, se você tivesse que implantar uma solução que foi criada pensando na recuperação de desastres, havia duas implantações separadas: uma para a região primária e outra para a secundária. Por exemplo, se estivesse implantando um aplicativo MVC ASP.NET usando os serviços de aplicativo, você criaria um serviço de aplicativo e o configuraria para a região primária. Em seguida, você conduziria outra implantação com o mesmo modelo para outra região com a mesma configuração.

Com a disponibilidade de implantações entre assinaturas e grupos de recursos, é possível criar o site de recuperação de desastres ao mesmo tempo que o site primário. Isso elimina duas implantações e garante que a mesma configuração possa ser usada em vários sites.

Implantar grupos de recursos com modelos do ARM

Nesta seção, um modelo do ARM será criado e implantado, o que criará alguns grupos de recursos na mesma assinatura.

Para usar o PowerShell para implantar modelos que contenham grupos de recursos e recursos entre assinaturas, é necessário usar a versão mais recente do PowerShell. No momento da gravação, a versão 6.6.0 do módulo do Azure foi usada:

PS C:\Users\rites> Get-Module azurerm.resources			
ModuleType	Version	Name	ExportedCommands
script	6.6.0	AzureRM.Resources	{Add-AzureRmADGroupMember,

Se o módulo mais recente do Azure não estiver instalado, faça isso usando o seguinte comando:

```
install-module -Name Azurerm -Force
```

Agora, é hora de criar um modelo do ARM que criará vários grupos de recursos na mesma assinatura. O código do modelo do ARM é mostrado a seguir:

```
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/
deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "resourceGroupInfo": {
      "type": "array"
    },
    "multiLocation": {
      "type": "array"
    }
  },
  "resources": [
    {
      "type": "Microsoft.Resources/resourceGroups",
```

```
    "location": "[parameters('multiLocation') [copyIndex()]]",
    "name": "[parameters('resourceGroupInfo') [copyIndex()]]",
    "apiVersion": "2018-05-01",
    "copy": [
        {
            "name": "allResourceGroups",
            "count": "[length(parameters('resourceGroupInfo'))]"
        },
        "properties": {}
    ],
    "outputs": {}
}
```

A primeira seção da listagem de código fala sobre os parâmetros que os modelos do ARM esperam. Eles são parâmetros obrigatórios, e qualquer pessoa que implantar esses modelos deverá fornecer valores para eles. É necessário fornecer valores de matriz para ambos os parâmetros.

A segunda seção principal é a matriz JSON de recursos, que pode conter vários deles. Neste exemplo, estamos criando grupos de recursos. Portanto, isso é declarado na seção Recursos. Os grupos de recursos estão sendo provisionados em um loop devido ao uso do elemento de cópia. O elemento de cópia garante que o recurso seja executado por um número especificado de vezes e crie um novo recurso em todas as iterações. Se enviarmos dois valores para o parâmetro de matriz resourceGroupInfo, o comprimento da matriz será "dois", e o elemento de cópia garantirá que o recurso resourceGroup seja executado duas vezes.

Todos os nomes de recursos em um modelo devem ser exclusivos de um tipo de recurso. Portanto, a função copyIndex é usada para fornecer um número de iteração atual e atribuída ao nome para torná-lo exclusivo. Além disso, queremos que os grupos de recursos sejam criados em regiões diferentes usando nomes de regiões distintos enviados como um parâmetro. A atribuição de um nome e local para cada grupo de recursos é feita usando a função copyIndex.

O código do arquivo de parâmetros é mostrado a seguir. Esse código é bastante simples e fornece valores de matriz para os dois parâmetros esperados pelo modelo anterior. Os valores nesse arquivo devem ser alterados para todos os parâmetros de acordo com o ambiente do leitor:

```
{
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/
deploymentParameters.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "resourceGroupInfo": {
```

```
        "value": [ "firstResourceGroup", "SeocndResourceGroup" ]
    },
    "multiLocation": {
        "value": [
            "West Europe",
            "East US"
        ]
    }
}
```

Para implantar esse modelo com o PowerShell, faça logon no Azure com credenciais válidas usando o seguinte comando:

```
Login-AzureRmAccount
```

As credenciais válidas podem ser uma conta de usuário ou uma entidade de serviço. Então, use um cmdlet New-AzureRmDeployment recém-lançado para implantar o modelo. O script de implantação está disponível no arquivo multipleResourceGroups.ps1:

```
New-AzureRmDeployment -Location "West Europe" -TemplateFile "c:\users\rites\source\repos\CrossSubscription\CrossSubscription\multipleResourceGroups.json" -TemplateParameterFile "c:\users\rites\source\repos\CrossSubscription\CrossSubscription\multipleResourceGroups.parameters.json" -Verbose
```

É importante entender que o cmdlet New-AzureRMResourceGroupDeployment não pode ser usado aqui. Ele não pode ser usado porque o escopo do cmdlet New-AzureRMResourceGroupDeployment é um grupo de recursos. Além disso, ele espera que um grupo de recursos esteja disponível como um pré-requisito. Para implantar recursos no nível da assinatura, o Azure lançou um novo cmdlet que pode funcionar acima do escopo do grupo de recursos. O novo cmdlet, new-AzureRmDeployment, funciona no nível da assinatura.

O mesmo modelo também pode ser implantado usando a CLI do Azure. Veja a seguir as etapas para implantá-lo usando a CLI do Azure:

1. Use a versão mais recente da CLI do Azure para criar grupos de recursos com o modelo do ARM. No momento da elaboração deste e-book, a versão 2.0.43 foi usada para a implantação, como mostrado aqui:

```
C:\Users\rites>az --version
azure-cli (2.0.43)
```

2. Faça logon no Azure usando o seguinte comando e selecione a assinatura certa a ser usada:

```
C:\Users\rites>az login
```

3. Se o logon tiver acesso a várias assinaturas, selecione uma assinatura apropriada usando o seguinte comando:

```
C:\Users\rites>az account set --subscription xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

4. Execute a implantação usando o comando a seguir. O script de implantação está disponível no arquivo `multipleResourceGroupsCLI.txt`:

```
C:\Users\rites>az deployment create --location WestUS --template-file "c:\users\rites\source\repos\CrossSubscription\CrossSubscription\azuredeploy.json" --parameters @"c:\users\rites\source\repos\CrossSubscription\CrossSubscription\azuredeploy.parameters.json" --verbose
```

Implantar recursos em assinaturas e grupos de recursos

Na última seção, os grupos de recursos foram criados como parte dos modelos do ARM, que é um recurso relativamente novo do Azure. Outro novo recurso do Azure é a provisão de recursos em várias assinaturas simultaneamente a partir de uma única implantação usando um único modelo do ARM. Nesta seção, forneceremos uma nova conta de armazenamento em duas assinaturas e grupos de recursos diferentes. A pessoa que está implantando o modelo do ARM seleciona uma das assinaturas como a assinatura base. Com ela, essa pessoa inicia a implantação e provisiona a conta de armazenamento na assinatura atual e em outra. O pré-requisito para implantar esse modelo é que a pessoa que está fazendo a implantação deve ter acesso a pelo menos duas assinaturas e direitos de colaborador nelas. A listagem de código é mostrada aqui e está disponível no arquivo `CrossSubscriptionStorageAccount.json` no código que o acompanha:

```
{
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "storagePrefix1": {
```

```

    "type": "string",
    "defaultValue": "st01"
  ...
    "type": "string",
    "defaultValue": "rg01"
  },
  "remoteSub": {
    "type": "string",
    "defaultValue": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
  }
}
...
}
],
"outputs": {}
}
}
]
,
"outputs": {}
}

```

É importante observar que os nomes do grupo de recursos usados no código já devem estar disponíveis nas respectivas assinaturas. O código lançará um erro se os grupos de recursos não estiverem disponíveis. Além disso, os nomes do grupo de recursos devem corresponder exatamente no modelo do ARM.

O código para implantar esse modelo é mostrado a seguir. Nesse caso, usamos New-AzureRmResourceGroupDeployment porque o escopo da implantação é um grupo de recursos. O script de implantação está disponível no arquivo CrossSubscriptionStorageAccount.ps1:

```

New-AzureRmResourceGroupDeployment -TemplateFile "<< path to your
CrossSubscriptionStorageAccount.json file >>" -ResourceGroupName
"<<provide your base subscription resource group name>>"
-storagePrefix1 <<provide prefix for first storage account>>
-storagePrefix2 <<provide prefix for first storage account>> -verbose

```

Outro exemplo de implantações entre assinaturas e grupos de recursos

Nesta seção, criamos duas contas de armazenamento em duas assinaturas, grupos de recursos e regiões diferentes de um modelo do ARM e uma única implantação. Ele usa a abordagem de modelos aninhados junto com o elemento de cópia para fornecer nomes e locais diferentes para esses grupos de recursos em assinaturas distintas.

No entanto, antes de podermos executar o próximo conjunto de modelos do ARM, uma instância do Azure Key Vault deve ser provisionada como um pré-requisito, e um segredo deve ser adicionado a ela. Isso acontece porque os nomes das contas de armazenamento são recuperados do Azure Key Vault e transmitidos como parâmetros a modelos do ARM para provisionamento da conta de armazenamento.

Para provisionar o Azure Key Vault usando o Azure PowerShell, o próximo conjunto de comandos pode ser executado. O código para os seguintes comandos está disponível no arquivo CreateKeyVaultAndSetSecret.ps1 com o código que o acompanha:

```
New-AzureRmResourceGroup -Location <>replace with location of your  
key vault>> -Name <>replace with name of your resource group for key  
vault>> -verbose  
New-AzureRmKeyVault -Name <>replace with name of your key vault>>  
-ResourceGroupName <>replace with name of your resource group  
for key vault>> -Location <>replace with location of your key  
vault>> -EnabledForDeployment -EnabledForTemplateDeployment  
-EnabledForDiskEncryption -EnableSoftDelete -EnablePurgeProtection  
-Sku Standard -Verbose
```

Os leitores devem ter em mente que o valor ResourceID deve ser observado a partir do resultado do cmdlet New-AzureRmKeyVault. Esse valor precisará ser substituído no arquivo de parâmetros. Veja detalhes na captura de tela a seguir:

```
VERBOSE: Performing the operation "Create key vault" on target "testkeyvaultbook".  
Vault Name : testkeyvaultbook  
Resource Group Name : rg01  
Location : West Europe  
Resource ID : /subscriptions/.../resourceGroups/rg01/providers/Microsoft.KeyVault/vaults/testkeyvaultbook  
Vault URL : https://testkeyvaultbook.vault.azure.net/  
Tenant ID :  
SKU : Standard  
Enabled For Deployment? : True  
Enabled For Template Deployment? : True  
Enabled For Disk Encryption? : True  
Soft Delete Enabled? : True  
Access Policies :  
    Tenant ID :  
    Object ID :  
    Application ID :  
    Display Name : Ritesh Modi (callritz@hotmail.com#EXT#@callritzhotmail.onmicrosoft.com)  
    Permissions to Keys : get, create, delete, list, update, import, backup, restore, recover  
    Permissions to Secrets : get, list, set, delete, backup, restore, recover  
    Permissions to Certificates : get, delete, list, create, import, update, deleteissuers, getissuers, listissuers, managecontacts, manageissuers, setissuers, recover, backup, restore  
    Permissions to (Key Vault Managed) Storage : delete, deletesas, get, getsas, list, listsas, regeneratekey, set, setosas, update, recover, backup, restore  
Network Rule Set :  
    Default Action : Allow  
    Bypass : Azureservices  
    IP Rules :  
    Virtual Network Rules :
```

Execute o seguinte comando para adicionar um novo segredo à instância recém-criada do Azure Key Vault:

```
Set-AzureKeyVaultSecret -VaultName <>replace with name of your  
key vault>> -Name <>replace with name of yoursecret>> -SecretValue  
$(ConvertTo-SecureString -String <>replace with value of your secret>>  
-AsPlainText -Force ) -Verbose
```

A listagem de código está disponível no arquivo

`CrossSubscriptionNestedStorageAccount.json` no código que o acompanha:

```
{  
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/  
deploymentTemplate.json#",  
    "contentVersion": "1.0.0.0",  
    "parameters": {  
        "hostingPlanNames": {  
            "type": "array",  
            "minLength": 1  
        },  
        ...  
        "type": "Microsoft.Resources/deployments",  
        "name": "deployment01",  
        "apiVersion": "2017-05-10",  
        "subscriptionId": "[parameters('subscriptions')[copyIndex()]]",  
        "resourceGroup": "[parameters('resourceGroups')[copyIndex()]]",  
        "copy": {  
            "count": "[length(parameters('hostingPlanNames'))]",  
            "name": "mywebsites", "mode": "Parallel"  
        },  
        ...  
        "kind": "Storage",  
        "properties": {  
        }  
    }  
}
```

Veja a seguir o código do arquivo de parâmetros. Ele está disponível no arquivo `CrossSubscriptionNestedStorageAccount.parameters.json`:

```
{  
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/  
deploymentParameters.json#",  
    "contentVersion": "1.0.0.0",  
    "parameters": {  
        "hostingPlanNames": {  
            ...  
        "storageKey": {  
            "reference": {  
                "keyVault": { "id": "<>replace it with the value of Key vault  
ResourceId noted before>>" }  
            }  
        }  
    }  
}
```

```
        "secretName": "<<replace with the name of the secret available  
in Key vault>>"  
    }  
}  
}  
}  
}
```

Veja a seguir o código do PowerShell para implantar o modelo anterior. O script de implantação está disponível no arquivo `CrossSubscriptionNestedStorageAccount.ps1`:

```
New-AzureRmResourceGroupDeployment -TemplateFile "c:\users\rites\source\repos\CrossSubscription\CrossSubscription\CrossSubscriptionNestedStorageAccount.json" -ResourceGroupName rg01 -TemplateParameterFile "c:\users\rites\source\repos\CrossSubscription\CrossSubscription\CrossSubscriptionNestedStorageAccount.parameters.json" -Verbose
```

Implantar entre assinaturas e grupos de recursos usando modelos vinculados

O exemplo anterior usava modelos aninhados para implantação em várias assinaturas e grupos de recursos. No próximo exemplo, implantaremos vários planos de serviço de aplicativo em assinaturas e grupos de recursos separados usando modelos vinculados. Os modelos vinculados são armazenados no Armazenamento de Blobs do Azure, que é protegido com políticas. Isso significa que somente o titular da chave da conta de armazenamento ou uma assinatura de acesso compartilhado válida pode acessar esse modelo. A chave de acesso é armazenada no Azure Key Vault e acessada a partir do arquivo de parâmetros usando referências do elemento `storageKey`. Os leitores devem carregar o arquivo `website.json` em um contêiner do Armazenamento de Blobs do Azure. O arquivo `website.json` é um modelo vinculado responsável por provisionar um plano de Serviço de Aplicativo e um Serviço de Aplicativo. O arquivo é protegido usando a política **Privada (sem acesso anônimo)**, como mostrado na próxima captura de tela. Uma política de privacidade garante que o acesso anônimo não seja permitido. Criei um contêiner chamado `armtemplates` e o defini com uma política privada:

IDENTIFIER	START TIME	EXPIRY TIME	PERMISSIONS
armtemplates			<input checked="" type="checkbox"/> Read <input checked="" type="checkbox"/> Write <input checked="" type="checkbox"/> Delete <input checked="" type="checkbox"/> List <input checked="" type="checkbox"/> Add <input checked="" type="checkbox"/> Create <input checked="" type="checkbox"/> Update <input checked="" type="checkbox"/> Process

Esse arquivo só pode ser acessado usando as chaves SAS. As chaves SAS podem ser geradas a partir do portal do Azure para uma conta de armazenamento usando o **item de assinatura de acesso compartilhado** no menu à esquerda, conforme mostrado a seguir. Os leitores devem clicar no botão **Gerar a cadeia de conexão e de SAS** para gerar o token SAS. Lembre-se de que um token SAS é exibido uma vez e não é armazenado com o Azure. Então, copie e armazene-o em algum lugar para que ele possa ser carregado no Azure Key Vault. A próxima captura de tela mostra a geração do token SAS:

Usaremos o mesmo Key Vault que foi criado na seção anterior. Só precisamos garantir que haja dois segredos disponíveis no Key Vault. O primeiro segredo é `StorageName`, e o outro é `StorageKey`. Os comandos para criar esses segredos no Key Vault são:

```
Set-AzureKeyVaultSecret -VaultName "testkeyvaultbook" -Name  
"storageName" -SecretValue $(ConvertTo-SecureString -String  
"uniqueusername" -AsPlainText -Force ) -Verbose  
  
Set-AzureKeyVaultSecret -VaultName "testkeyvaultbook" -Name  
"storageKey" -SecretValue $(ConvertTo-SecureString -String "?sv=2018-  
03-28&ss=bfqt&srt=sco&sp=rwdlacup&se=2019-03-29T21:51:03Z&st=2019-  
01-30T14:51:03Z&spr=https&sig=gTynGhj20er6pDl7Ab%2Bpc29WO3%2BJhvi%2  
BfF%2F6rHYWp4g%3D" -AsPlainText -Force ) -Verbose
```

Os leitores são aconselhados a alterar os nomes do Key Vault e o valor da chave secreta com base na respectiva conta de armazenamento.

Depois de garantir que o Key Vault tenha os segredos necessários, o código do arquivo do modelo do ARM poderá ser usado para implantar o modelo aninhado entre assinaturas e grupos de recursos.

O código do modelo do ARM está disponível no arquivo `CrossSubscriptionLinkedStorageAccount.json` e também é mostrado aqui. Os leitores são aconselhados a alterar o valor da variável `templateUrl` nesse arquivo. Ele deve ser atualizado com um local de arquivo válido do Armazenamento de Blobs do Azure:

```
{  
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/  
deploymentTemplate.json#",  
  "contentVersion": "1.0.0.0",  
  "parameters": {  
    "hostingPlanNames": {  
      "type": "array",  
      "minLength": 1  
    ...  
    "type": "Microsoft.Resources/deployments",  
    "name": "fsdfsdf",  
    "apiVersion": "2017-05-10",  
    "subscriptionId": "[parameters('subscriptions') [copyIndex()]]",  
    "resourceGroup": "[parameters('resourceGroups') [copyIndex()]]",  
    "copy": {  
      "count": "[length(parameters('hostingPlanNames'))]"  
    }  
  }  
}
```

```
        "name": "mywebsites",
        "mode": "Parallel"
    ...
]
}
```

O código do arquivo de parâmetros é mostrado a seguir. Os leitores são aconselhados a alterar os valores dos parâmetros, incluindo o resourceid do Key Vault e o nome do segredo. Os nomes dos Serviços de Aplicativo devem ser exclusivos. Caso contrário, o modelo não será implantado. O código do arquivo de parâmetros está disponível no arquivo de código `CrossSubscriptionLinkedStorageAccount.parameters.json`:

```
{
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/
deploymentParameters.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "hostingPlanNames": {
            "value": [ "firstappservice", "secondappservice" ]
        ...
        "storageKey": {
            "reference": {
                "keyVault": { "id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxx/resourceGroups/keyvaluedemo/providers/Microsoft.KeyVault/
vaults/forsqlvault1" },
                "secretName": "storageKey"
            }
        }
    }
}
```

Veja a seguir o comando para implantar o modelo. O script de implantação está disponível no arquivo `CrossSubscriptionLinkedStorageAccount.ps1`:

```
New-AzureRmResourceGroupDeployment -TemplateFile "c:\users\rites\source\repos\CrossSubscription\CrossSubscription\CrossSubscriptionLinkedStorageAccount.json" -ResourceGroupName <>replace with the base subscription resource group name >> -TemplateParameterFile "c:\users\rites\source\repos\CrossSubscription\CrossSubscription\CrossSubscriptionLinkedStorageAccount.parameters.json" -Verbose
```

Resumo

A capacidade de implantar recursos usando uma única implantação para várias assinaturas, grupos de recursos e regiões proporciona melhor capacidade de implantar, reduzir bugs na implantação e oferecer benefícios avançados, como criar sites de recuperação de desastres e alta disponibilidade.

No próximo capítulo, vamos nos concentrar na criação de modelos do ARM modulares, uma habilidade essencial para os arquitetos que realmente querem levar seus modelos do ARM para o próximo nível. O próximo capítulo mostrará várias maneiras de desenvolver modelos do ARM e criar modelos do ARM reutilizáveis e modulares.

5

Design modular e implementação de modelos do ARM

Sabemos que há várias maneiras de criar um modelo do **Azure Resource Manager (ARM)**. É muito fácil criar um que provisone todos os recursos necessários no Azure usando o Visual Studio e o Visual Studio Code. Um único modelo do ARM pode consistir em todos os recursos necessários para uma solução no Azure. Esse modelo pode ter apenas alguns recursos ou ser maior e conter muitos recursos.

Enquanto a criação de um único modelo que consiste em todos os recursos é muito tentadora, é aconselhável planejar com antecedência uma implementação do modelo do ARM dividida em vários modelos do ARM menores para que problemas futuros relacionados a eles possam ser evitados.

Neste capítulo, veremos como escrever modelos do ARM de maneira modular para que eles evoluam ao longo de um período de tempo com o mínimo de envolvimento em termos de alterações e esforço em testes e implantação.

No entanto, antes de escrever modelos modulares, é melhor entender os problemas resolvidos escrevendo-os de forma modular.

Os tópicos a seguir serão abordados neste capítulo:

- Problemas com um único modelo
- Entender a implantação aninhada e vinculada
- Modelos vinculados
- Modelos aninhados
- Configurações de fluxo livre
- Configurações conhecidas

Problemas com um único modelo

Na superfície, pode não parecer que um único modelo grande que consiste em todos os recursos terá problemas, mas eles poderão surgir no futuro. Vamos entender os problemas que podem surgir com modelos grandes únicos.

Flexibilidade reduzida na alteração de modelos

O uso de um único modelo grande com todos os recursos dificulta sua alteração no futuro. Com todas as dependências, parâmetros e variáveis em um único modelo, a alteração dele pode levar um tempo considerável em comparação com modelos menores. A alteração pode ter um impacto em outras seções do modelo, que podem passar despercebidas, bem como introduzir bugs.

Solucionar problemas com modelos grandes

É difícil solucionar problemas com modelos grandes. Esse é um fato conhecido. Quanto maior o número de recursos em um modelo, mais difícil será solucionar os problemas dele. Um modelo implanta todos os recursos nele, e encontrar um bug envolve a implantação do modelo com bastante frequência. Os desenvolvedores reduzem a produtividade enquanto aguardam a conclusão da implantação do modelo.

Além disso, a implantação de um único modelo é mais demorada do que modelos menores. Os desenvolvedores precisam aguardar a implantação de recursos que contenham erros antes de tomar qualquer medida.

Abuso de dependência

As dependências entre recursos também tendem a se tornar mais complexas em modelos maiores. É muito fácil abusar do uso do recurso `dependsOn` em modelos do ARM devido à maneira como eles funcionam. Cada recurso em um modelo pode se referir a todos os seus recursos anteriores, em vez de criar uma árvore de dependências. Os modelos do ARM não reclamarão se um único recurso depender de todos os outros no modelo do ARM, mesmo que esses outros recursos possam ter interdependências. Isso torna a mudança de modelos do ARM propensa a bugs e, às vezes, nem sequer é possível alterá-los.

Agilidade reduzida

Geralmente, há várias equipes de recursos em um projeto, cada uma com seus próprios recursos no Azure. Essas equipes acharão difícil trabalhar com um único modelo do ARM porque um único desenvolvedor deve atualizá-lo.

A atualização de um único modelo com várias equipes pode causar conflitos e mesclagens difíceis de resolver. Com vários modelos menores, cada equipe pode criar seu próprio modelo do ARM.

Sem reutilização

Não há problema se você tem um único modelo, e o uso dele acarreta a implantação de todos os recursos. Não há nenhuma possibilidade de selecionar recursos individuais sem algumas manobras, como a adição de recursos condicionais. Um único modelo grande perde a reutilização, pois ou você implanta todos os recursos, ou não implanta nenhum.

Sabendo que modelos grandes únicos têm tantos problemas, convém criar modelos modulares para ter os seguintes benefícios:

- Várias equipes podem trabalhar em seus próprios modelos isoladamente
- Os modelos podem ser reutilizados em projetos e soluções
- É fácil depurar e solucionar problemas dos modelos

Entender o princípio da responsabilidade única

O princípio da responsabilidade única é um dos princípios fundamentais dos princípios de design sólidos. Ele informa que um segmento de classe ou código deve ser responsável por uma única função e deve possuir essa funcionalidade completamente. O código deve ser alterado ou evoluído somente se houver uma alteração funcional ou um bug na funcionalidade atual, e não o contrário. Esse código não deve ser alterado devido a mudanças em algum outro componente ou código que não faz parte do componente atual.

Aplicar o mesmo princípio aos modelos do ARM ajuda a criar modelos que têm a única responsabilidade de implantar um único recurso ou funcionalidade, em vez de implantar todos os recursos e uma solução completa.

O uso desse princípio ajudará você a criar vários modelos, cada um responsável por um único recurso ou um grupo menor de recursos, e não por todos os recursos.

Solução de problemas e depuração mais rápidas

Cada implantação de modelo é uma atividade distinta no Azure e é uma instância separada que consiste em entradas, saídas e logs. Quando vários modelos são implantados para uma solução, cada implantação de modelo tem entradas de log separadas, além de suas descrições de entrada e saída. É muito mais fácil isolar bugs e solucionar problemas usando esses logs independentes de várias implantações, em comparação com um único modelo grande.

Modelos modulares

Quando um único modelo grande é decomposto em vários modelos em que cada modelo menor cuida de seus recursos, e esses recursos são exclusivamente mantidos, de propriedade e responsabilidade do modelo que o contém, podemos dizer que temos modelos modulares. Cada modelo dentro desses modelos segue o princípio da responsabilidade única.

Antes de entender como dividir um modelo grande em vários modelos reutilizáveis menores, é importante entender a tecnologia por trás da criação de modelos menores e como criá-los para implantar soluções completas.

Recursos de implantações

O ARM facilita a vinculação de modelos. Embora já tenhamos visto os modelos vinculados em detalhes, vou mencioná-los aqui para ajudar você a entender como os modelos de vinculação ajudam a alcançar a modularidade, a composição e a reutilização.

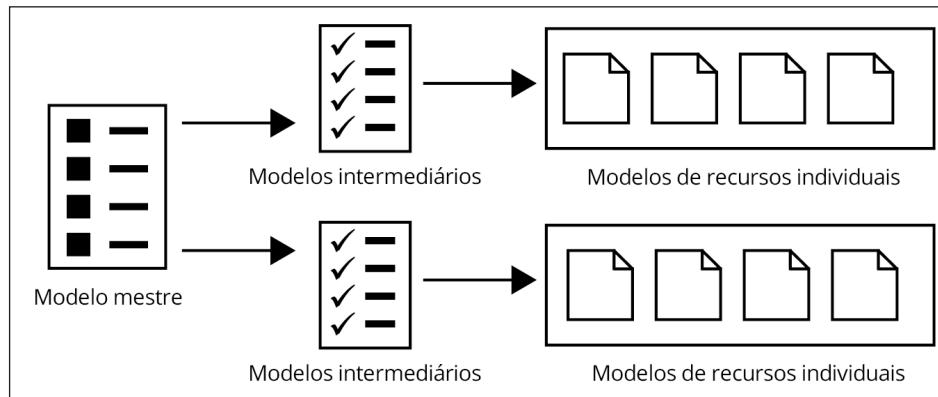
Os modelos do ARM fornecem recursos especializados conhecidos como **implantações**, que estão disponíveis no namespace Microsoft.Resources. Um recurso de implantações em um modelo do ARM tem aparência muito semelhante à do segmento de código a seguir:

```
"resources": [
  {
    "apiVersion": "2017-05-10",
    "name": "linkedTemplate",
    "type": "Microsoft.Resources/deployments",
    "properties": {
      "mode": "Incremental",
      <nested-template-or-external-template>
    }
  }
]
```

Esse modelo é auto-explicativo, e as duas configurações mais importantes no recurso do modelo são o tipo e as propriedades. O tipo aqui refere-se ao recurso de implantações, e não a qualquer recurso específico do Azure (armazenamento, VM e assim por diante), e as propriedades especificam a configuração de implantação, incluindo uma implantação de modelo vinculado ou uma implantação de modelo aninhada.

No entanto, o que o recurso de implantações faz? A função de um recurso de implantações é implantar outro modelo. Outro modelo pode ser um externo em um arquivo de modelo do ARM separado ou um modelo aninhado. Portanto, é possível invocar outros modelos de um modelo, assim como uma chamada de função.

Pode haver níveis aninhados de implantações em modelos do ARM. Isso significa que um único modelo pode chamar outro, e o modelo chamado pode chamar outro, e isso pode continuar por cinco níveis de chamadas aninhadas. No próximo diagrama, há um modelo mestre que chama internamente dois outros modelos usando dois recursos distintos de implantações, e esses dois recursos chamam modelos adicionais que implantam outros recursos. Essa relação pai-filho pode ir até cinco níveis de profundidade:



Modelos vinculados

Os modelos vinculados são modelos que invocam modelos externos. Os modelos externos são armazenados em arquivos de modelo do ARM diferentes. Um exemplo de modelos vinculados é:

```

"resources": [
  {
    "apiVersion": "2017-05-10",
    "name": "linkedTemplate",
    "type": "Microsoft.Resources/deployments",
    "properties": {
      "mode": "Incremental",
      "templateLink": {
        "uri": "https://mystorageaccount.blob.core.windows.net/
AzureTemplates/newStorageAccount.json",
        "contentVersion": "1.0.0.0"
      },
      "parametersLink": {
        "uri": "https://mystorageaccount.blob.core.windows.net/
AzureTemplates/newStorageAccount.parameters.json",
        "contentVersion": "1.0.0.0"
      }
    }
  }
]
  
```

```
        }
    }
]
```

Propriedades adicionais importantes neste modelo em comparação com o modelo anterior são `templateLink` e `parametersLink`. Agora, `templateLink` refere-se ao URL real do local do arquivo de modelo externo e `parametersLink` é o local do URL do arquivo de parâmetros correspondente. É importante lembrar que o modelo do chamador deve ter direitos de acesso para o local do modelo chamado. Por exemplo, se os modelos externos são armazenados no Armazenamento de Blobs do Azure, que é protegido por chaves, as chaves SAS apropriadas devem estar disponíveis para que o modelo do chamador possa acessar os modelos vinculados.

Também é possível fornecer parâmetros em linha explícitos em vez do valor `parametersLink`, conforme mostrado aqui:

```
"resources": [
{
    "apiVersion": "2017-05-10",
    "name": "linkedTemplate",
    "type": "Microsoft.Resources/deployments",
    "properties": {
        "mode": "Incremental",
        "templateLink": {
            "uri": "https://mystorageaccount.blob.core.windows.net/AzureTemplates/newStorageAccount.json",
            "contentVersion": "1.0.0.0"
        },
        "parameters": {
            "StorageAccountName": {
                "value": "[parameters('StorageAccountName')]"
            }
        }
    }
]
```

Modelos aninhados

Os modelos aninhados são um recurso relativamente novo em modelos do ARM em comparação com modelos vinculados externos.

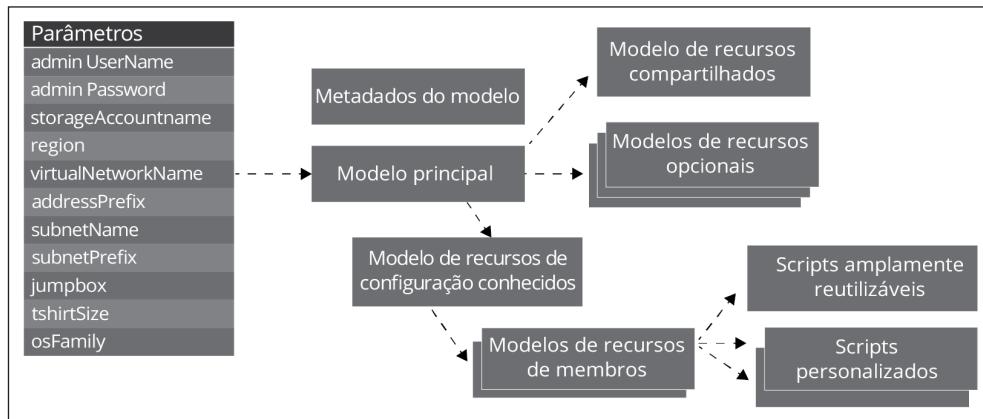
Os modelos aninhados não definem recursos em arquivos externos. Os recursos são definidos no próprio modelo do chamador e no recurso de implantações, como mostrado aqui:

```
"resources": [
  {
    "apiVersion": "2017-05-10",
    "name": "nestedTemplate",
    "type": "Microsoft.Resources/deployments",
    "properties": {
      "mode": "Incremental",
      "template": {
        "$schema": "https://schema.management.azure.com/schemas/2015-
          01-01/deploymentTemplate.json#",
        "contentVersion": "1.0.0.0",
        "resources": [
          {
            "type": "Microsoft.Storage/storageAccounts",
            "name": "[variables('storageName')]",
            "apiVersion": "2015-06-15",
            "location": "West US",
            "properties": {
              "accountType": "Standard_LRS"
            }
          }
        ]
      }
    }
  ]
]
```

Neste segmento de código, podemos ver que o recurso de conta de armazenamento está aninhado dentro do modelo original como parte do recurso de implementações. Em vez de usar os atributos `templateLink` e `parametersLink`, uma matriz de recursos é usada para criar vários recursos como parte de uma única implantação. A vantagem de usar uma implantação aninhada é que os recursos dentro de um pai podem ser usados para reconfigurá-los usando seus nomes. Normalmente, um recurso com um nome pode existir apenas uma vez dentro de um modelo. Os modelos aninhados nos permitem usá-los dentro do mesmo modelo e garantir que todos os modelos sejam autossuficientes, em vez de serem armazenados separadamente, e eles podem ou não estar acessíveis para esses arquivos externos.

Agora que entendemos a tecnologia por trás dos modelos do ARM modulares, como devemos dividir um modelo grande em modelos menores?

Há várias maneiras pelas quais um modelo grande pode ser decomposto em modelos menores. A Microsoft recomenda o seguinte padrão para a decomposição de modelos do ARM:



Ao decompor um modelo grande em modelos menores, há sempre o modelo principal, que é usado para implantar a solução. Esse modelo principal ou mestre invoca internamente outros modelos aninhados ou vinculados e eles, por sua vez, invocam outros modelos. Por fim, os modelos que contêm recursos do Azure são implantados.

O modelo principal pode invocar um modelo de recurso de configuração conhecido que, por sua vez, invocará os modelos que compõem os recursos do Azure. O modelo de recurso de configuração conhecido é específico de um projeto ou solução e não tem muitos fatores reutilizáveis associados a ele. Os modelos de recurso de membro são modelos reutilizáveis invocados pelo modelo de recurso de configuração conhecido.

Opcionalmente, o modelo mestre pode invocar modelos de recursos compartilhados e outros modelos de recursos, se existirem.

É importante entender as configurações conhecidas. Os modelos podem ser criados como configurações conhecidas ou de fluxo livre.

Configurações de fluxo livre

Os modelos do ARM podem ser criados como modelos genéricos em que a maioria (se não todos) dos valores atribuídos às variáveis é obtida como parâmetros. Isso permite que a pessoa que usa o modelo transmita qualquer valor que considere necessário para implantar recursos no Azure. Por exemplo, a pessoa que está implantando o modelo pode escolher uma máquina virtual de qualquer tamanho, qualquer número de máquinas virtuais e qualquer configuração para seu armazenamento e redes. Ela é conhecida como configuração de fluxo livre, onde a maior parte da configuração é permitida e os modelos são provenientes do usuário, e não declarados no modelo.

Há desafios nesse tipo de configuração. O maior deles é que não há suporte para todas as configurações em todos os data centers e regiões do Azure. Os modelos falharão ao criar recursos se eles não tiverem permissão para serem criados em regiões ou locais específicos. Outro problema com a configuração de fluxo livre é que os usuários podem fornecer qualquer valor que considerem necessário e um modelo os atenderá, aumentando assim o custo e a pegada de implantação, mesmo que eles não sejam completamente necessários.

Configurações conhecidas

Configurações conhecidas, por outro lado, são configurações pré-determinadas específicas para implantar um ambiente usando modelos do ARM. Essas configurações pré-determinadas são conhecidas como **configurações de dimensionamento t-shirt**. Semelhante ao modo como um t-shirt está disponível em uma configuração pré-determinada, como pequenas, médias e grandes, os modelos do ARM podem ser pré-configurados para implantar um ambiente pequeno, médio ou grande, dependendo dos requisitos. Isso significa que os usuários não podem determinar qualquer tamanho personalizado aleatório para o ambiente, mas podem escolher entre várias opções fornecidas, e os modelos do ARM executados durante o tempo de execução garantirão que uma configuração apropriada do ambiente seja fornecida.

Assim, a primeira etapa da criação de um modelo do ARM modular é decidir quais serão as configurações conhecidas de um ambiente.

Como exemplo, veja a seguir a configuração de uma implantação de data center no Azure:

Tamanho t-shirt	Configuração do modelo do ARM
Pequena	4 máquinas virtuais com 7 GB de memória, além de 4 núcleos de CPU
Média	8 máquinas virtuais com 14 GB de memória, além de 8 núcleos de CPU
Grande	16 máquinas virtuais com 28 GB de memória, além de 8 núcleos de CPU

Agora que conhecemos as configurações, podemos criar modelos do ARM modulares.

Há duas maneiras de escrever modelos do ARM modulares:

- **Modelos compostos:** modelos compostos vinculados a outros modelos. Exemplos de modelos compostos são os modelos mestre e intermediário.
- **Modelos de nível folha:** são modelos que contêm um único recurso do Azure.

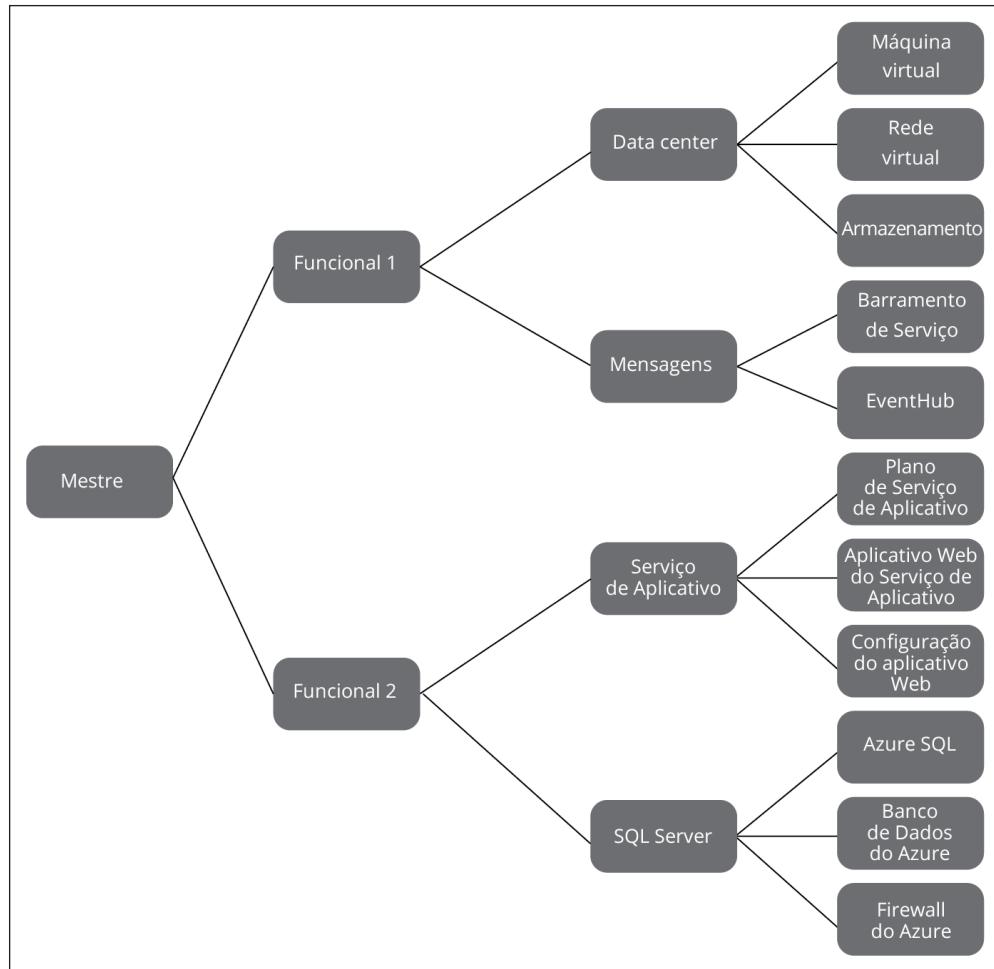
Os modelos do ARM podem ser divididos em modelos modulares com base em:

- Tecnologia
- Funcionalidade

Uma maneira ideal de decidir qual método modular será usado para criar um modelo do ARM é:

- Defina modelos de nível folha ou recursos que consistem em recursos únicos. No diagrama a seguir, os modelos de extrema direita são modelos de nível folha. No diagrama, as máquinas virtuais, a rede virtual, o armazenamento e outros na mesma coluna representam modelos de nível folha.
- Componha modelos específicos do ambiente usando os modelos de nível folha. Esses modelos específicos do ambiente fornecem um ambiente do Azure, como um ambiente do SQL Server, de Serviço de Aplicativo ou de data center. Vamos nos aprofundar um pouco mais neste tópico. Vamos ver o exemplo de um ambiente de SQL do Azure. Para criar um ambiente de SQL do Azure, são necessários vários recursos. No mínimo, um SQL Server lógico, um banco de dados SQL e alguns recursos de firewall do SQL devem ser provisionados. Todos esses recursos são definidos em modelos individuais ao nível folha. Esses recursos podem ser compostos juntos em um único modelo que tem a capacidade de criar um ambiente SQL do Azure. Qualquer pessoa que queira criar um ambiente de SQL pode usar esse modelo composto. O diagrama mostrado a seguir tem data center, mensagens e Serviço de Aplicativo como modelos específicos do ambiente.
- Crie modelos com maior abstração compondo vários modelos específicos do ambiente em soluções. Esses modelos são compostos por modelos específicos do ambiente que foram criados na etapa anterior. Por exemplo, para criar uma solução de inventário de comércio eletrônico que precise de um ambiente de Serviço de Aplicativo e um ambiente de SQL, dois modelos de ambiente, Serviço de Aplicativo e SQL Server podem ser compostos juntos. O diagrama mostrado a seguir tem os modelos **Funcional 1** e **Funcional 2**, que são compostos por modelos filhos.
- Por fim, um modelo mestre deve ser criado e deve ser composto por vários modelos em que cada um deles pode implantar uma solução.

As etapas anteriores para criar um modelo projetado modular podem ser facilmente entendidas por meio de um diagrama:



Agora, vamos implementar uma parte da funcionalidade mostrada no diagrama anterior. Nesta implementação, forneceremos uma máquina virtual com uma extensão de script usando uma abordagem modular. A extensão de script personalizada implanta binários do Docker e prepara um ambiente de contêiner em uma máquina virtual do Windows Server 2016.

Agora, vamos criar uma solução usando modelos do ARM com uma abordagem modular. Como mencionado anteriormente, a primeira etapa é criar modelos de recursos individuais. Tais modelos serão usados para compor modelos adicionais capazes de criar um ambiente. Esses modelos serão necessários para criar uma máquina virtual. Todos os modelos do ARM mostrados aqui estão disponíveis no código do capítulo que os acompanha. Os nomes e o código desses modelos são:

- Storage.json
- virtualNetwork.json
- PublicIPAddress.json
- NIC.json
- VirtualMachine.json
- CustomScriptExtension.json

Primeiro, vejamos o código do modelo `Storage.json`. Esse modelo fornece uma conta de armazenamento de que todas as máquinas virtuais precisam para armazenar seus arquivos de sistema operacional e disco de dados:

```
{
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "storageAccountName": {
            "type": "string",
            "minLength": 1
        },
        "storageType": {
            "type": "string",
            "minLength": 1
        },
        ...
    },
    "outputs": {
        "resourceDetails": {
            "type": "object",
            "value": "[reference(parameters('storageAccountName'))]"
        }
    }
}
```

Em seguida, veremos o código do modelo de endereço IP público. Uma máquina virtual que deve ser acessível pela Internet precisa de um recurso de endereço IP público atribuído à sua placa de interface de rede. Embora a exposição de uma máquina virtual à Internet seja opcional, esse recurso pode ser usado para criar uma máquina virtual. O código a seguir está disponível no arquivo `PublicIPAddress.json`:

```
{  
    "$schema": "https://schema.management.azure.com/schemas/2015-01-  
01/deploymentTemplate.json#",  
    "contentVersion": "1.0.0.0",  
    "parameters": {  
        "publicIPAddressName": {  
            "type": "string",  
            "minLength": 1  
        },  
        "publicIPAddressType": {  
            "type": "string",  
            "minLength": 1  
        }  
    },  
    "...  
    "outputs": {  
        "resourceDetails": {  
            "type": "object",  
            "value": "[reference(parameters('publicIPAddressName'))]"  
        }  
    }  
}
```

Em seguida, veremos o código da rede virtual. As máquinas virtuais no Azure precisam de uma rede virtual para comunicação. Esse modelo será usado para criar uma rede virtual no Azure com um intervalo de endereços pré-definidos e sub-redes. O código a seguir está disponível no arquivo `virtualNetwork.json`:

```
{  
    "$schema": "https://schema.management.azure.com/schemas/2015-01-  
01/deploymentTemplate.json#",  
    "contentVersion": "1.0.0.0",  
    "parameters": {  
        "virtualNetworkName": {  
            "type": "string",  
            "minLength": 1  
        }  
    },  
    "...  
    "subnetPrefix": {  
        "type": "string",  
        "minLength": 1  
    },  
    "resourceLocation": {  
        "type": "string",  
        "minLength": 1  
    }  
}
```

```
        "minLength": 1
    }
    ...
    "subnets": [
        {
            "name": "[parameters('subnetName')]",
            "properties": {
                "addressPrefix": "[parameters('subnetPrefix')]"
            }
        }
    ]
},
"outputs": {
    "resourceDetails": {
        "type": "object",
        "value": "[reference(parameters('virtualNetworkName'))]"
    }
}
}
```

Em seguida, veremos o código da placa da interface de rede. Uma placa de rede virtual é necessária para que uma máquina virtual se conecte a uma rede virtual, além de aceitar e enviar solicitações da Internet e para ela. O código a seguir está disponível no arquivo NIC.json:

```
{
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "nicName": {
            "type": "string",
            "minLength": 1
        },
        "publicIpReference": {
            "type": "string",
            "minLength": 1
        },
        ...
    },
    "resources": [
        {
            "type": "Microsoft.Network/networkInterfaces",
            "name": "[resourceId(subscription().subscriptionId, resourceGroup().name, 'Microsoft.Network/networkInterfaces', parameters('nicName'))]",
            "properties": {
                "ipConfigurations": [
                    {
                        "name": "ipConfig1",
                        "properties": {
                            "privateIPAllocationMethod": "Dynamic",
                            "subnet": "[resourceId(subscription().subscriptionId, resourceGroup().name, 'Microsoft.Network/virtualNetworks', parameters('vnetRef')).subnets[0].id]"
                        }
                    }
                ],
                "networkSecurityGroup": "[resourceId(subscription().subscriptionId, resourceGroup().name, 'Microsoft.Network/networkSecurityGroups', parameters('nsgName'))]"
            }
        }
    ]
}
```

```
virtualNetworks', parameters('virtualNetworkReference'))]",
    "subnet1Ref": "[concat(variables('vnetRef'), '/subnets/', parameters('subnetReference'))]"
},
...
        "id": "[variables('subnet1Ref')]"
    }
}
]
}
],
"outputs": {
    "resourceDetails": {
        "type": "object",
        "value": "[reference(parameters('nicName'))]"
    }
}
}
```

Em seguida, veremos o código para criar uma máquina virtual. Cada máquina virtual é um recurso no Azure. Além disso, lembre-se de que esse modelo não tem nenhuma referência ao armazenamento, à rede, aos endereços IP públicos ou a outros recursos criados anteriormente. Essa referência e composição acontecerá posteriormente nesta seção usando outro modelo. O código a seguir está disponível no arquivo `VirtualMachine.json`:

```
{
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "vmName": {
            "type": "string",
            "minLength": 1
        },
        ...
    },
    "imageOffer": {
        "type": "string",
        "minLength": 1
    },
    "windowsOSVersion": {
        "type": "string",
        "minLength": 1
    },
    ...
}
```

```
"outputs": {
    "resourceDetails": {
        "type": "object",
        "value": "[reference(parameters('vmName'))]"
    }
}
```

Em seguida, veremos o código para criar uma extensão de script personalizado. Esse recurso executa um script do PowerShell em uma máquina virtual depois que é provisionado. Esse recurso permite executar tarefas pós-provisionamento em máquinas virtuais do Azure. O código a seguir está disponível no arquivo `CustomScriptExtension.json`:

```
{
    "$schema": "http://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "VMName": {
            "type": "string",
            "defaultValue": "sqldock",
            "metadata": {
                ...
                "commandToExecute": "[concat('powershell -ExecutionPolicy Unrestricted -file docker.ps1')]"
            },
            "protectedSettings": {
            }
        }
    },
    "outputs": {
    }
}
```

Em seguida, veremos o código do PowerShell de extensão de script personalizado que prepara o ambiente do Docker. Uma reinicialização de máquina virtual pode ocorrer durante a execução do script do PowerShell, dependendo de o recurso de contêineres do Windows já estar instalado ou não. O script a seguir instala o pacote NuGet, o provedor `DockerMsftProvider` e o arquivo executável do Docker. O arquivo `docker.ps1` está disponível com o código de capítulo que o acompanha:

```
#  
# docker.ps1  
#
```

```
Install-PackageProvider -Name Nuget -Force -ForceBootstrap  
-Confirm:$false  
Install-Module -Name DockerMsftProvider -Repository PSGallery -Force  
-Confirm:$false -verbose  
Install-Package -Name docker -ProviderName DockerMsftProvider -Force  
-ForceBootstrap -Confirm:$false
```

Todos os modelos vinculados anteriormente vistos devem ser carregados em um contêiner em uma conta de Armazenamento de Blobs do Azure. Esse contêiner pode ter uma política de acesso privado aplicada, como vimos no capítulo anterior. No entanto, para este exemplo, definiremos a política de acesso como contêiner. Isso significa que esses modelos vinculados podem ser acessados sem a necessidade de um token SAS.

Por fim, vamos nos concentrar em escrever o modelo mestre. No modelo mestre, todos os modelos vinculados são compostos juntos para criar uma solução – para implantar uma máquina virtual e executar um script nela. A mesma abordagem pode ser usada para criar outras soluções, como fornecer um data center que consiste em várias máquinas virtuais interconectadas. O código a seguir está disponível no arquivo Master.json:

```
{  
    "$schema": "https://schema.management.azure.com/  
schemas/2015-01-01/deploymentTemplate.json#",  
    "contentVersion": "1.0.0.0",  
    "parameters": {  
        "storageAccountName": {  
            "type": "string",  
            "minLength": 1  
        },  
        "subnetName": {  
            "type": "string",  
            "minLength": 1  
        },  
        "subnetPrefix": {  
            "type": "string",  
            "minLength": 1  
        },  
        "windowsOSVersion": {  
            "type": "string",  
            "minLength": 1  
        },  
        "vhdStorageName": {  
            "type": "string",  
            "minLength": 1  
        }  
    },  
    "variables": {},  
    "resources": [],  
    "outputs": {}  
}
```

```
        },
        "vhdStorageContainerName": {
            "type": "string",
            "minLength": 1
            ...[concat('https://',parameters('storageAccountName'),'armtfiles.
blob.core.windows.net/',variables('containerName'),'/Storage.json')]",
            "contentVersion": "1.0.0.0"
        },
        "parameters": {
            "storageAccountName": {
                "value": "[parameters('storageAccountName')]"
            },
            "storageType": {
                "value": "[parameters('storageType')]"
            },
            "resourceLocation": {
                "value": "[resourceGroup().location]"
            ...
        },
        "outputs": {
            "resourceDetails": {
                "type": "object",
                "value": "[reference('GetVM').outputs.resourceDetails.value]"
            }
        }
    }
}
```

Os modelos mestres invocam os modelos externos e também coordenam interdependências entre eles.

Os modelos externos devem estar disponíveis em um local bem conhecido para que o modelo mestre possa acessá-los e invocá-los. Neste exemplo, os modelos externos são armazenados no contêiner de Armazenamento de Blobs do Azure, e essas informações foram transmitidas para o modelo do ARM por meio de parâmetros.

Os modelos externos no Armazenamento de Blobs do Azure podem ter acesso protegido por meio da configuração de políticas de acesso. O comando usado para implantar o modelo mestre é mostrado a seguir. Pode parecer um comando complexo, mas a maioria dos valores é usada como parâmetros. Os leitores são aconselhados a alterar o valor desses parâmetros antes da execução. Os modelos vinculados foram carregados em uma conta de armazenamento denominada `st02gvwldcxm5suwe` no contêiner `armtemplates`. O grupo de recursos deverá ser criado se não existir no momento. O primeiro comando é usado para criar um novo grupo de recursos na região **Europa Ocidental**:

```
New-AzureRmResourceGroup -Name "testvmrg" -Location "West Europe"
-Verbose
```

O restante dos valores de parâmetro é necessário para configurar cada recurso. O nome da conta de armazenamento e o valor dnsNameForPublicIP devem ser únicos no Azure:

```
New-AzureRmResourceGroupDeployment -Name "testdeploy1"
-ResourceGroupName testvmrg -Mode Incremental -TemplateFile "C:\chapter 05\Master.json" -storageAccountName "st02gvwldc xm5suwe"
-storageType "Standard_LRS" -publicIPAddressName "uniipaddname"
-publicIPAddressType "Dynamic" -dnsNameForPublicIP
"azureforarchitectsbook" -virtualNetworkName vnetwork01
-addressPrefix "10.0.1.0/16" -subnetName "subnet01" -subnetPrefix
"10.0.1.0/24" -nicName nic02 -vmSize "Standard_DS1" -adminUsername
"sysadmin" -adminPassword $(ConvertTo-SecureString -String
sysadmin@123 -AsPlainText -Force) -vhdStorageName oddnewuniqueacc
-vhdStorageContainerName vhds -OSDiskName mynewvm -vmName
vm10 -windowsOSVersion 2012-R2-Datacenter -imagePublisher
MicrosoftWindowsServer -imageOffer WindowsServer -containerName
armtemplates -Verbose
```

Resumo

Os modelos do ARM são os meios preferidos de provisionamento de recursos no Azure. Eles são idempotentes por natureza, trazendo consistência, previsibilidade e reutilização para a criação do ambiente. Neste capítulo, vimos como criar um modelo do ARM modular. É importante que as equipes passem um tempo criando modelos do ARM de modo adequado para que várias equipes possam trabalhar neles juntas. Eles são altamente reutilizáveis e exigem mudanças mínimas para evoluir.

O próximo capítulo abordará uma vertente diferente e muito popular da tecnologia conhecida como sem servidor no Azure. O Azure Functions é um dos principais recursos sem servidor no Azure. Ele será abordado de maneira detalhada, incluindo o Durable Functions.

6

Design e implementação de soluções sem servidor

Sem servidor é um dos chavões mais quentes em tecnologia hoje em dia, e todo mundo quer pegar uma carona. Sem servidor traz muitas vantagens em computação em geral, processos de desenvolvimento de software, infraestrutura e implementação técnica. Há muita coisa acontecendo na indústria: em uma extremidade do espectro está a **Infraestrutura como Serviço (IaaS)**, e na outra a função sem servidor. Entre elas estão a **Plataforma como Serviço (PaaS)** e os contêineres. Conheci muitos desenvolvedores, e parece-me que existe um certo nível de confusão entre eles sobre IaaS, PaaS, contêineres e computação sem servidor. Além disso, há muita confusão sobre casos de uso, aplicabilidade, arquitetura e implementação para o paradigma sem servidor. Sem servidor é um novo paradigma que está mudando não apenas a tecnologia, mas também a cultura e os processos dentro das organizações.

Neste capítulo, exploraremos a função sem servidor cobrindo os seguintes tópicos:

- Sem servidor
- Azure Functions
- Eventos sem servidor
- Grade de Eventos do Azure
- Fluxos de trabalho sem servidor
- Aplicativos Lógicos
- Criar uma solução completa usando a integração sem servidor

Sem servidor

Sem servidor refere-se a um modelo de implantação na qual os usuários são responsáveis somente pelo código e pela configuração do seu aplicativo. Os clientes do sem servidor não precisam se preocupar com a plataforma subjacente e a infraestrutura e podem se concentrar na resolução dos problemas de negócios escrevendo códigos.

Sem servidor não significa que não existem servidores. O código e a configuração sempre precisam de algum servidor para execução. No entanto, da perspectiva do cliente, um aplicativo pode ser praticamente sem servidor. Sem servidor significa que o cliente não vê o servidor. Eles não se importam com a plataforma e a infraestrutura subjacentes. Eles não precisam gerenciar nem monitorar nada. Sem servidor fornece um ambiente que pode ser escalado e reduzido verticalmente, de forma automática, sem os clientes ficarem cientes disso. Todas as operações relacionadas a plataformas e infraestruturas acontecem nos bastidores. Os clientes recebem **contratos de nível de serviço (SLAs)** relacionados à performance, e o Azure garante que eles atendam aos SLAs, independentemente das demandas de servidor.

Os clientes só precisam trazer seu código; é responsabilidade do provedor de nuvem fornecer a infraestrutura e a plataforma necessárias para executar o código.

A evolução da função sem servidor

Antes de entendermos a arquitetura e a implementação da função sem servidor, é importante entender sua história e como ela evoluiu. No início, havia servidores físicos. Embora os usuários tivessem total controle sobre os servidores físicos, havia muitas desvantagens:

- Longos períodos de gestação entre a encomenda e a implantação real do servidor
- Capital intensivo por natureza
- Desperdício de recursos
- Retorno menor sobre o investimento
- Difícil de escalar horizontal e verticalmente

Uma evolução natural dos servidores físicos foi a virtualização. A virtualização refere-se à criação de máquinas virtuais em servidores físicos e à implantação de aplicativos dentro delas. As máquinas virtuais oferecem várias vantagens:

- Não há necessidade de adquirir hardware físico
- Comparativamente mais fácil de criar máquinas virtuais mais novas

- Isolamento completo de ambientes
- Custos mais baixos em comparação a servidores físicos

No entanto, a virtualização tinha seu próprio conjunto de desvantagens:

- Ainda dependente de contratos físicos de um servidor para a escala horizontal após um número de instâncias de máquina virtual.
- Ainda é caro por causa da dependência de humanos e de hardware.
- Desperdício de recursos de computação – cada máquina virtual executa um sistema operacional completo.
- Altos custos de manutenção.

A próxima evolução foi a IaaS de provedores de nuvem. Em vez da aquisição e gerenciamento de data centers e infraestruturas, a estratégia foi criar máquinas virtuais, armazenamento e redes na nuvem. A nuvem fornece serviços de infraestrutura definida por software e esconde todos os detalhes relacionados a armazenamento, redes e servidores físicos. Isso tinha algumas vantagens:

- Sem despesa de capital, apenas despesas operacionais. As funções são cobradas com base no modelo de consumo, em vez de custo fixo (embora haja um modelo de Serviço de Aplicativo baseado em custo fixo).
- Sem tempo de gestação para criar novas máquinas virtuais – novas máquinas virtuais podem ser provisionadas em minutos, em vez de horas.
- Tamanho flexível de máquinas virtuais.
- Mais fácil escalar acima e fora de máquinas virtuais.
- Completamente seguro.

As máquinas virtuais na nuvem têm algumas desvantagens:

- Requer monitoramento ativo e auditoria.
- Requer manutenção ativa de máquinas virtuais.
- Escalabilidade, alta disponibilidade e performance de máquinas virtuais devem ser gerenciadas pelos usuários – qualquer degradação e melhoria subsequente é responsabilidade do usuário.
- Uma opção cara porque os usuários pagam pela máquina toda, seja ela usada ou não.

A nuvem também fornece outro padrão para implantar aplicativos, popularmente conhecidos como PaaS. PaaS fornece abstração da infra-estrutura subjacente em termos de máquinas virtuais, redes virtuais e armazenamento na nuvem. Ele fornece uma plataforma e um ecossistema onde os usuários não precisam saber qualquer coisa sobre a infraestrutura. Eles podem simplesmente implantar seu aplicativo nessas plataformas. Há vantagens no uso de PaaS em comparação com IaaS, mas ainda há opções melhores. As principais desvantagens de PaaS são os seguintes:

- Os aplicativos PaaS são implantados em máquinas virtuais nos bastidores, e o modelo de pagamento não é granular; ele ainda está no nível da implantação.
- A PaaS ainda exige monitoramento para escala e redução horizontais.
- Os usuários ainda precisam identificar os requisitos para sua plataforma. Há opções limitadas disponíveis para diferentes tipos de plataforma. O Azure forneceu exclusivamente um ambiente baseado no Windows até recentemente, quando começou a oferecer o Linux também. Além disso, a instalação de software, utilitários e pacotes é de responsabilidade de seus usuários.

Surgiu um novo paradigma, conhecido como contêineres, que é popularizado principalmente pelo Docker. Os contêineres fornecem um ambiente leve, isolado e seguro que tem todos os benefícios de máquinas virtuais sem suas desvantagens. Eles não têm um sistema operacional dedicado e, em vez disso, contam com um sistema operacional de servidor de base. Os contêineres vêm nos padrões IaaS e PaaS. Os contêineres fornecem muitas vantagens:

- Provisionamento mais rápido de ambientes
- Criação de ambientes consistente e previsível
- Facilita a criação de arquiteturas de microsserviços
- Um ecossistema rico com serviços avançados do Kubernetes, Swarm e DC/OS

Os contêineres têm algumas desvantagens:

- Eles exigem monitoramento ativo e auditoria.
- Eles exigem manutenção ativa.
- A escalabilidade, alta disponibilidade e performance de contêineres devem ser gerenciadas por ferramentas de orquestração, como o Kubernetes. Essas ferramentas de orquestração precisam de implantações de extensão e habilidades para serem gerenciadas.

A função sem servidor, por definição, é um paradigma de implantação. Ela pode ser implantada em máquinas virtuais, bem como em contêineres. Para tirar o melhor proveito da função sem servidor, ela deve ser implantada em contêineres para utilizar seus recursos de destruição e criação mais rápida. Isso terá um impacto direto na escalabilidade e alta disponibilidade da plataforma sem servidor, e também será muito mais rápido em comparação com uma máquina virtual.

Princípios da tecnologia sem servidor

A tecnologia sem servidor se baseia nos seguintes princípios:

- **Custo mais baixo:** o custo se baseia no consumo real de recursos de computação e energia. Não haverá custos se não houver nenhum consumo.
- **Escalabilidade ilimitada:** com a função sem servidor, não há necessidade de realizar operações de dimensionamento manual ou automático. A expansão e a redução verticais são gerenciadas diretamente pela plataforma do Azure. A plataforma garante que servidores suficientes sejam fornecidos para oferecer suporte à implantação sem servidor, seja para algumas centenas de usuários ou para milhões deles. Esse dimensionamento acontece de maneira transitória, sem qualquer interferência ou conhecimento da organização que está implantando os componentes sem servidor.
- **Controlada por eventos:** funções sem servidor devem ser executadas com base em determinados eventos que estão acontecendo. O evento deve acionar a execução da função. Em outras palavras, sem servidor deve permitir que funções a dissociar-se de outras funções e em vez disso dependem o fuzilamento de determinados eventos em que estão interessados.
- **Responsabilidade única:** as funções sem servidor devem implementar funcionalidade e responsabilidade únicas e devem fazer isso bem. Várias responsabilidades não devem ser codificadas ou implementadas em uma única função.
- **Execução rápida:** as funções sem servidor não devem demorar muito tempo para concluir um trabalho. Elas devem ser capazes de executar rapidamente e voltar.

As vantagens do Azure Functions

A computação sem servidor é um paradigma relativamente novo que ajuda as organizações a converter grandes funcionalidades em funções menores, discretas, sob demanda que podem ser invocadas e executadas por meio de gatilhos automatizados e trabalhos agendados. Eles também são conhecidos como **Funções como Serviço (FaaS)**, nos quais as organizações podem se concentrar nos desafios do domínio em vez de na plataforma e na infraestrutura subjacentes. As FaaS também ajudam na transferência de arquiteturas de solução para funções reutilizáveis menores, aumentando assim o retorno sobre os investimentos.

Há uma infinidade de plataformas de computação sem servidor disponíveis. Algumas das mais importantes são:

- Azure Functions
- AWS Lambda
- IBM OpenWhisk
- Iron.io
- Funções do Google Cloud

Na verdade, todos os dias parece que há uma nova estrutura sendo apresentada e está se tornando cada vez maior a dificuldade para que as empresas decidam e escolham a estrutura que funciona melhor para elas. O Azure fornece um valioso ambiente sem servidor conhecido como Azure Functions, e eu gostaria de salientar alguns recursos para os quais ele oferece suporte:

- Diversas maneiras de invocar uma função – manual, agendada ou com base em um evento
- Diversos tipos de suporte vinculativo
- Capacidade de executar funções de forma síncrona e assíncrona
- Executar funções com base em vários tipos de gatilhos
- Capacidade de executar funções de longa e curta duração
- Capacidade de usar recursos de proxy para utilizar arquiteturas de função diferentes
- Vários modelos de uso, incluindo o consumo, bem como o modelo de Serviço de Aplicativo
- Capacidade de criar funções usando várias linguagens, como JavaScript, Python e C#
- Autorização baseada em OAuth
- Várias opções de autenticação, incluindo o Azure AD, Facebook, Twitter e outros provedores de identidade
- Configure facilmente parâmetros de entrada e saída
- Integração do Visual Studio para criação do Azure Functions
- Paralelismo maciço

FaaS

O Azure fornece FaaS. Elas são implementações sem servidor do Azure. Com o Azure Functions, o código pode ser escrito em qualquer linguagem em que o usuário se sinta confortável, e o Azure Functions fornecerá um tempo de execução para executá-lo. Com base na linguagem escolhida, uma plataforma adequada é fornecida aos usuários por trazer o seu próprio código. As funções são uma unidade de implantação e podem ser automaticamente ativadas e desativadas. Ao lidar com as funções, os usuários não podem exibir as máquinas virtuais e as plataformas subjacentes, mas o Azure Functions fornece uma pequena janela para vê-las por meio do console do **Kudu**.

Há dois componentes principais do Azure Functions:

- Tempo de execução do Azure Functions
- Associação e gatilhos do Azure Functions

Tempo de execução do Azure Functions

O núcleo do Azure Functions está em seu tempo de execução do Azure. O precursor do Azure Functions foi o Azure WebJobs. O código do Azure WebJobs também forma o núcleo do Azure Functions. Existem recursos e extensões adicionais adicionados aos Azure WebJobs para criar o Azure Functions. O tempo de execução das funções é a mágica por trás do funcionamento das funções. O Azure Functions é hospedado nos Serviços de Aplicativo do Azure. O Serviço de Aplicativo do Azure carrega o tempo de execução do Azure e aguarda um evento externo ocorrer ou alguma solicitação HTTP. Na chegada de uma solicitação ou a ocorrência de um gatilho, o Serviço de Aplicativo carrega a carga de entrada, lê o arquivo `function.json` da função para encontrar ligações e o gatilho da função, mapeia os dados de entrada para parâmetros de entrada e invoca a função com valores de parâmetro. Quando a função concluir sua execução, o valor é passado novamente para o tempo de execução do Azure Functions por meio de um parâmetro de saída definido como associação no arquivo `function.json`. O tempo de execução da função retorna os valores para o chamador. O tempo de execução do Azure Functions age como a cola que habilita toda a performance das funções.

Associação e gatilhos do Azure Functions

Se o tempo de execução for o cérebro do Azure Functions, as associações e os gatilhos são o coração. O do Azure Functions promove o acoplamento e a alta coesão entre serviços usando gatilhos e associações. O aplicativo implementa o código usando sintaxe imperativa para parâmetros de entrada e de saída e valores de retorno. Isso geralmente resulta em codificar os parâmetros de entrada. Como o Azure Functions deve ser capaz de invocar qualquer função definida, elas implementam um mecanismo genérico para invocar funções por meio de gatilhos e associação.

Vinculação refere-se ao processo de criação de uma conexão entre os dados de entrada e o Azure Functions, mapeando os tipos de dados. A conexão pode ser um único sentido do tempo de execução ao Azure Functions, do Azure Functions para o tempo de execução para valores de retorno ou vários sentidos – a mesma associação pode transmitir dados entre o tempo de execução do Azure e o Azure Functions. O Azure Functions usa uma maneira declarativa para definir associações.

Os gatilhos são um tipo especial de associação por meio da qual as funções podem ser invocadas com base em eventos externos. Além de invocar a função, os gatilhos também transmitem os dados de entrada, a carga e os metadados para a função.

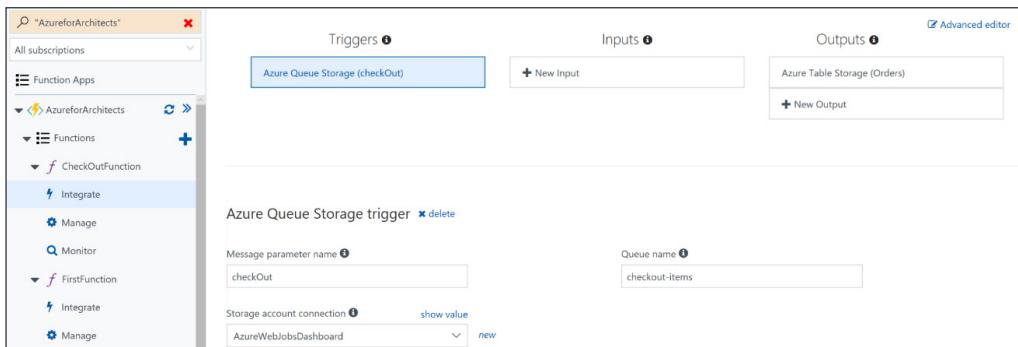
As associações são definidas no arquivo `function.json`:

```
{  
    "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  
}
```

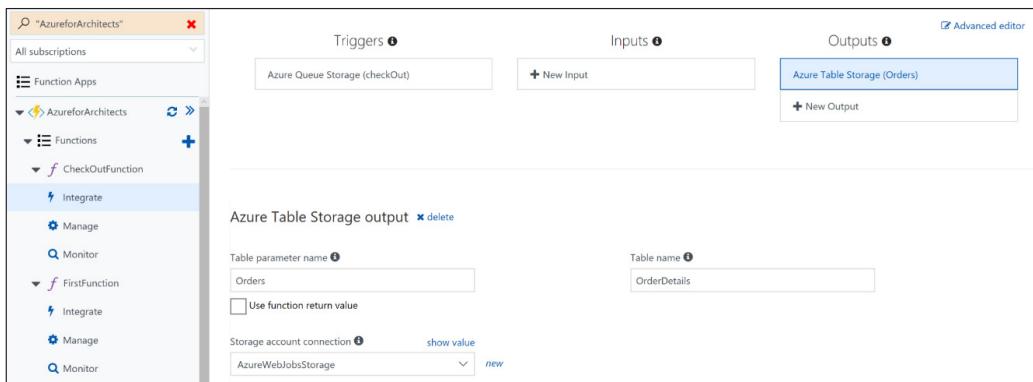
Neste exemplo, um gatilho é declarado e invoca a função sempre que há um novo item na fila de armazenamento. O tipo é `queueTrigger`, o sentido é entrada, `queueName` é `checkout-items` e os detalhes sobre a conexão da conta de armazenamento de destino e o nome da tabela também são mostrados. Todos esses valores são importantes para o funcionamento dessa associação. O nome `checkOut` pode ser usado no código da função como uma variável.

De maneira semelhante, uma associação para valores de retorno é declarada. Aqui, o valor de retorno é chamado de `Pedidos`, e os dados são a saída do Azure Functions. A associação grava os dados de retorno no armazenamento de tabelas do Azure usando a cadeia de conexão fornecida.

Tanto as associações e os gatilhos podem ser modificados e criados usando a guia Integrar no Azure Functions. Nos bastidores, o arquivo function.json é atualizado. O gatilho checkOut é declarado, como mostrado aqui:



A saída **Pedidos** é mostrada a seguir:



Os autores do Azure Functions não precisam gravar nenhum código de conexão para obter dados de várias fontes. Eles simplesmente decidem o tipo de dados esperado do tempo de execução do Azure. Isso é mostrado no próximo segmento de código. O checkout está disponível como uma cadeia de caracteres para a função. As funções fornecem vários tipos diferentes para ser capaz de enviar para uma função. Por exemplo, uma associação de fila pode fornecer o seguinte:

- **Objeto antigo e simples (POCO)**
- Cadeia de caracteres
- Byte[]
- CloudQueueMessage

O autor da função pode usar qualquer um desses tipos de dados, e o tempo de execução do Azure Functions garantirá que um objeto adequado como um parâmetro seja enviado para a função:

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

Também é importante saber que, nas capturas de tela anteriores, os nomes de conta de armazenamento são `AzureWebJobsStorage` e `AzureWebJobsDashboard`. Estas são as chaves definidas na configuração `appSettings` do Azure Functions.



Para obter mais informações sobre associações e gatilhos do Azure, consulte <https://docs.microsoft.com/azure/azure-functions/functions-bindings-storage-queue>.



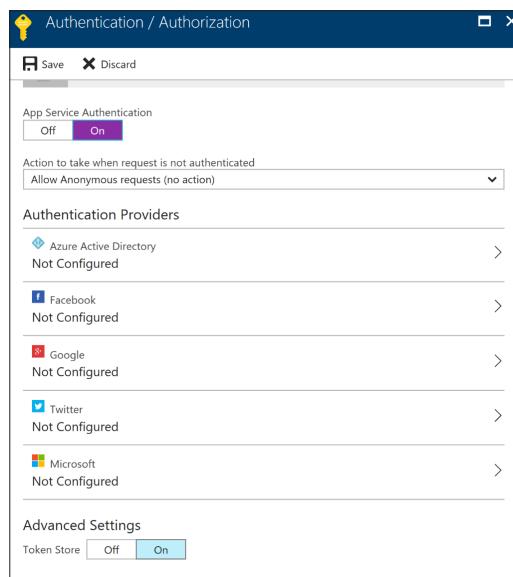
Monitoramento

As informações completas de log são fornecidas para cada solicitação ou gatilho na guia **Monitor** do Azure Functions. Isso ajuda na identificação de problemas e na auditoria de qualquer risco, bug ou exceção no Azure Functions.

A screenshot of the Azure Functions monitor interface. The left sidebar shows a navigation tree with 'AzureforArchitects' selected, under which 'Function Apps' is expanded to show 'AzureforArchitects' and 'FirstFunction'. 'FirstFunction' is further expanded to show 'Integrate', 'Manage', and 'Monitor' options, with 'Monitor' currently selected. Other collapsed items include 'QueueTriggerCSharp1', 'Proxies (preview)', and 'Slots (preview)'. The main content area has a header 'AzureforArchitects - FirstFunction' and a search bar with the placeholder 'AzureforArchitects'. A yellow banner at the top right says 'For a richer monitoring experience, including live metrics and custom queries, we recommend using Azure Application Insights.' Below this are two summary boxes: 'Success count since Sep 1st' (1) and 'Error count since Sep 1st' (0). The 'Invocation log' section shows a single entry: 'FirstFunction (Method: GET, Uri: ...)' with a green status indicator, 'Details: Last ran (duration)' showing '2 hours ago (1,572 ms)', and a 'live event stream' button. To the right, the 'Invocation details' section shows parameters: 'req' (Method: GET, Uri: https://azureforarchitects.azurewebsites.net/api/FirstFunction), 'log', '_binder', '_context' (ac42bcdd-ae49-40a8-a3cb-3cf8d615d356), '_logger', and '\$return' (response). At the bottom right is a 'Logs' section with a large empty box.

Autenticação e autorização

O Azure Functions se baseia no Serviço de Aplicativo do Azure para suas necessidades de autenticação. O Serviço de Aplicativo tem recursos de autenticação avançados em que os clientes podem usar OpenConnectID para autenticação e OAuth para autenticação. Os usuários podem ser autenticados usando o Azure AD, contas Microsoft, o Google, o Twitter ou o Facebook:



O Azure Functions que se baseia em HTTP também pode incorporar o uso de chaves, que devem ser enviadas junto com as solicitações HTTP:

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

As chaves a seguir estão disponíveis na guia **Gerenciar** no Azure Functions:

- **Teclas de função:** permitem autorização para funções individuais. Essas chaves devem ser enviadas como parte do cabeçalho da solicitação HTTP.
- **Teclas host:** permitem a autorização para todas as funções em um aplicativo de função. Essas chaves devem ser enviadas como parte do cabeçalho da solicitação HTTP.
- **Chaves padrão:** usadas quando você usa as teclas de função e host. Há uma tecla de host adicional chamada `_master` que ajuda no acesso administrativo para a API de tempo de execução do Azure Functions.

Configuração do Azure Functions

O Azure Functions fornece opções de configuração em vários níveis. Ele fornece a configuração para:

- A própria plataforma
- Os serviços de aplicativo de função

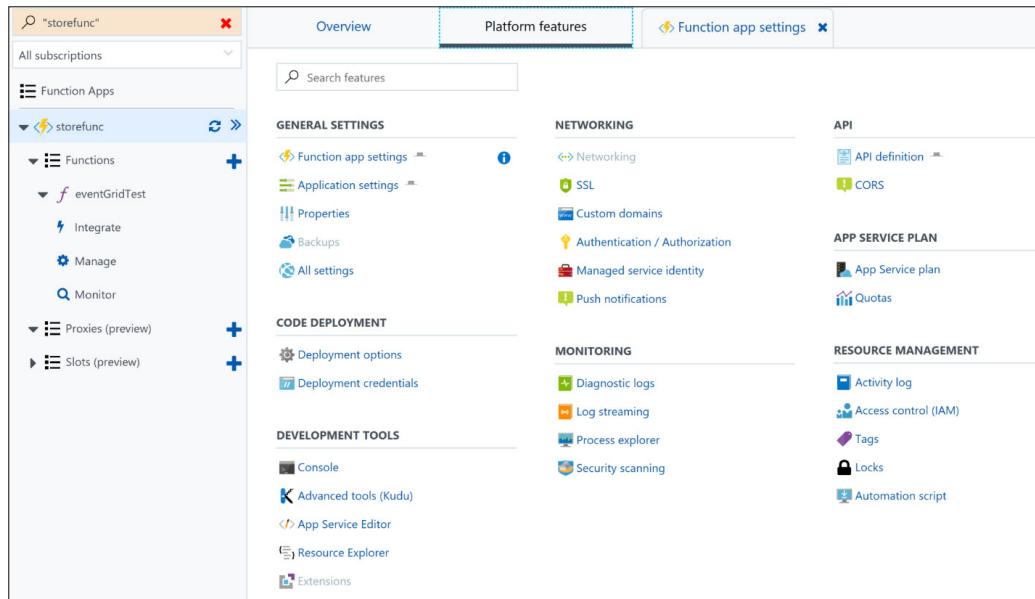
Essas configurações afetam cada função contida por elas. Mais informações sobre essas configurações estão disponíveis em <https://docs.microsoft.com/pt-br/azure/azure-functions/functions-how-to-use-azure-function-app-settings>.

Configuração de plataforma

O Azure Functions é hospedado no Serviço de Aplicativo do Azure e, portanto, obtém todos os recursos. Os logs de diagnóstico e monitoramento podem ser configurados com facilidade usando recursos de plataforma. Além disso, o Serviço de Aplicativo fornece opções para atribuir certificados SSL usando autenticação, autorização e um domínio personalizado como parte de seus recursos de rede.

Embora os clientes não estejam preocupados com a infraestrutura, o sistema operacional, o sistema de arquivos e a plataforma na qual as funções realmente são executadas, o Azure Functions fornece as ferramentas necessárias para ter uma prévia do sistema subjacente e fazer alterações. O console e o console do Kudu são as ferramentas usadas para essa finalidade. Eles fornecem um editor avançado para criar o Azure Functions e editar sua configuração.

O Azure Functions, assim como o Serviço de Aplicativo, permite o armazenamento de informações de configuração na seção de configuração do aplicativo web .config, que pode ser lido sob demanda:



Configurações da função de Serviço de Aplicativo

Essas configurações afetam todas as funções. As configurações do aplicativo podem ser gerenciadas aqui. Os proxies no Azure Functions podem ser habilitados ou desabilitados. Nós discutiremos os proxies mais à frente neste capítulo. Eles também ajudam na alteração do modo de edição de um aplicativo de função e a implantação em slots:

The screenshot shows the Azure Functions 'Function app settings' blade for the 'storefunc' app. It includes sections for Daily Usage Quota (GB-Sec), Application settings (Manage application settings), Proxies (preview) (Enable Azure Functions Proxies (preview)), Function app edit mode (Change the edit mode of your function app), and Host Keys (All functions). The 'Host Keys' table lists two entries: '_master' and 'default', each with a 'Click to show' button and actions like Copy, Renew, and Revoke.

Planos de custo do Azure Functions

O Azure Functions baseia-se no Serviço de Aplicativo do Azure e fornece um modelo de custos melhor para os usuários. Existem dois modelos de custo:

- **Plano de consumo:** isso se baseia no consumo real e na execução de funções. Esse plano calcula o custo com base no uso de computação durante o consumo real e a execução da função. Se uma função não for executada, não há nenhum custo associado a ela. No entanto, isso não significa que a performance esteja comprometida nesse plano. O Azure Function será expandido de forma automática com base na demanda para garantir que os níveis mínimos básicos de performance sejam mantidos. Uma execução de função é permitida 10 minutos para a sua conclusão.

- **Plano de serviço de aplicativo:** esse plano fornece funções com máquinas virtuais dedicadas completas nos bastidores e, por isso, o custo é diretamente proporcional ao custo da máquina virtual e seu tamanho. Há um custo associado a esse plano, mesmo que as funções não sejam executadas. O código de função pode ser executado durante o tempo necessário. Não há nenhum limite de tempo. No âmbito do plano de Serviço de Aplicativo, o tempo de execução da função fica ocioso se não utilizado dentro de alguns minutos e pode ser despertado somente usando um gatilho de HTTP. Há uma configuração **Always On** que pode ser usada para impedir que o tempo de execução da função fique ocioso. A escala é manual ou baseada em configurações de escala automática.

Casos de uso do Azure Functions

Existem muitos casos de uso válidos para utilização e implementação do Azure Functions:

- **Implementação de microserviços:** o Azure Functions ajudam na divisão de aplicativos grandes em unidades de código funcional discretas e menores. Cada unidade é tratada de forma independente do outra e evolui em seu próprio ciclo de vida. Cada unidade de código tem sua própria computação, hardware e requisitos de monitoramento. Cada função pode ser conectada a todas as outras funções. Essas unidades são tecidas juntas por orquestradores para criar funcionalidade completa. Por exemplo, em um aplicativo de comércio eletrônico, pode haver funções individuais (unidades de código), cada uma responsável por listagem de catálogos, recomendações, categorias, subcategorias, compras, carrinhos, check-outs, tipos de pagamento, gateways de pagamento, endereços de entrega, endereços de faturamento, impostos, frete, cancelamentos, devoluções, emails, SMS e assim por diante. Algumas dessas funções são reunidas para criar casos de uso para aplicativos de comércio eletrônico, como navegação de produtos e fluxo de check-out.
- **Integração entre vários pontos de extremidade:** o Azure Functions pode criar funcionalidades gerais de aplicativos ao integrarem várias funções. A integração pode se basear no disparo de eventos ou pode ser feita por push. Ela ajuda na decomposição de grandes aplicativos monolíticos em pequenos componentes.
- **Processamento de dados:** o Azure Functions pode ser usado para o processamento de dados de entrada em lotes. Ele pode ajudar no processamento de dados vem em vários formatos, como XML, CSV, JSON e TXT. Ele também pode executar algoritmos de conversão, enriquecimento, limpeza e filtragem. Na verdade, várias funções podem ser usadas, cada uma fazendo conversão ou enriquecimento, limpeza ou filtragem. O Azure Functions também pode ser usado para incorporar serviços cognitivos avançados, como **reconhecimento óptico de caracteres** (OCR), visão computacional e manipulação e conversão de imagem.

- **Integração de aplicativos herdados:** o Azure Functions pode ajudar na integração de aplicativos herdados com novos protocolos e aplicativos modernos. É possível que os aplicativos herdados não estejam usando protocolos e formatos padrão da indústria. O Azure Functions pode atuar como um proxy para esses aplicativos herdados, aceitar solicitações de usuários ou outros aplicativos, converter os dados em um formato compreendido por um aplicativo herdado e conversar com ele sobre protocolos que ele entenda. Isso abre muitas oportunidades para integrar e trazer aplicativos antigos e herdados para o portfólio principal.
- **Trabalhos agendados:** o Azure Functions pode ser usado para ser executado de modo contínuo ou periódico para determinadas funções do aplicativo. Essas funções do aplicativo podem realizar tarefas como fazer backups periódicos, restaurar, executar trabalhos de lote, exportar e importar dados e enviar emails em massa.
- **Gateways de comunicação:** o Azure Functions pode ser usado em gateways de comunicação ao utilizar hubs de notificação, SMS e email, por exemplo.

Tipos de Azure Functions

O Azure Functions pode ser categorizado em três tipos diferentes:

- **Funções sob demanda:** são funções executadas quando explicitamente chamadas ou invocadas. Os exemplos de tais funções incluem funções e webhooks baseados em HTTP.
- **Funções agendadas:** essas funções são como trabalhos de timer e executam as funções em intervalos fixos.
- **Funções baseadas em evento:** essas funções são executadas com base em eventos externos. Por exemplo, carregar um novo arquivo para Armazenamento de Blobs do Azure gera um evento que pode iniciar a execução do Azure Functions.

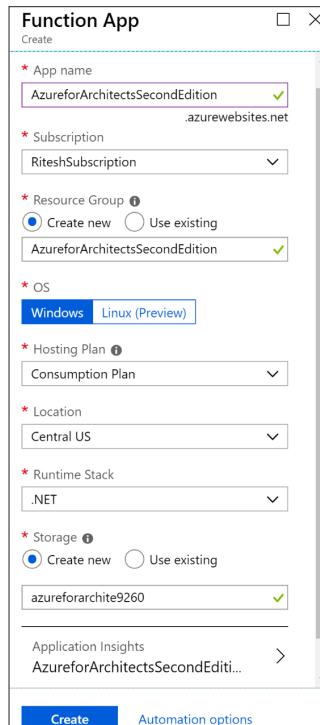
Como criar seu primeiro Azure Functions

O Azure Functions pode ser criado usando o portal do Azure, PowerShell, CLI do Azure e APIs REST. As etapas para criar uma função usando o modelo do ARM já foram detalhadas em <https://docs.microsoft.com/azure/azure-functions/functions-infrastructure-as-code>. Nesta seção, o Azure Functions será provisionado usando o portal.

O Azure Functions é hospedado no Serviços de Aplicativo do Azure. Os usuários criam um novo aplicativo de função que, por sua vez, cria um plano de Serviço de Aplicativo e um Serviço de Aplicativo. O plano de Serviço de Aplicativo é configurado com base no seguinte:

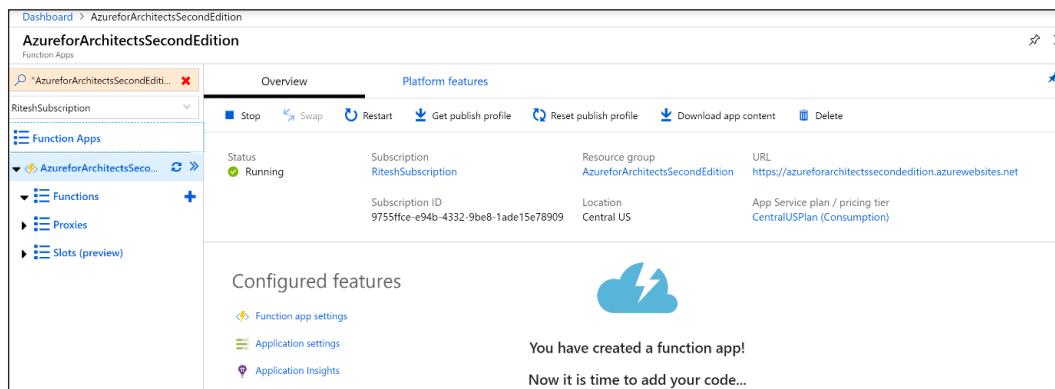
1. Depois de configurar o plano de Serviço de Aplicativo, preencha os detalhes da seguinte maneira:
 - **Nome:** o nome do Serviço de Aplicativo. O nome deve ser exclusivo em `.azurewebsites.net`.
 - **Local:** o local para hospedar o Serviço de Aplicativo do Azure Functions.
 - **Plano de hospedagem:** isso também é conhecido como o plano de preços. Aqui, duas opções estão disponíveis, conforme discutido anteriormente – planos de consumo e de Serviço de Aplicativo.
 - **Nome do grupo de recursos:** o nome do grupo de recursos que contém o plano de Serviço de Aplicativo e o Serviço de Aplicativo.
 - **Conta de armazenamento:** o Azure Functions precisa de uma conta de armazenamento do Azure para armazenar seus dados e logs internos.
 - **Habilitar insights de aplicativo:** habilite essa opção para capturar informações de telemetria do Azure Functions.

Depois de preencher os detalhes, clique em **Criar**:



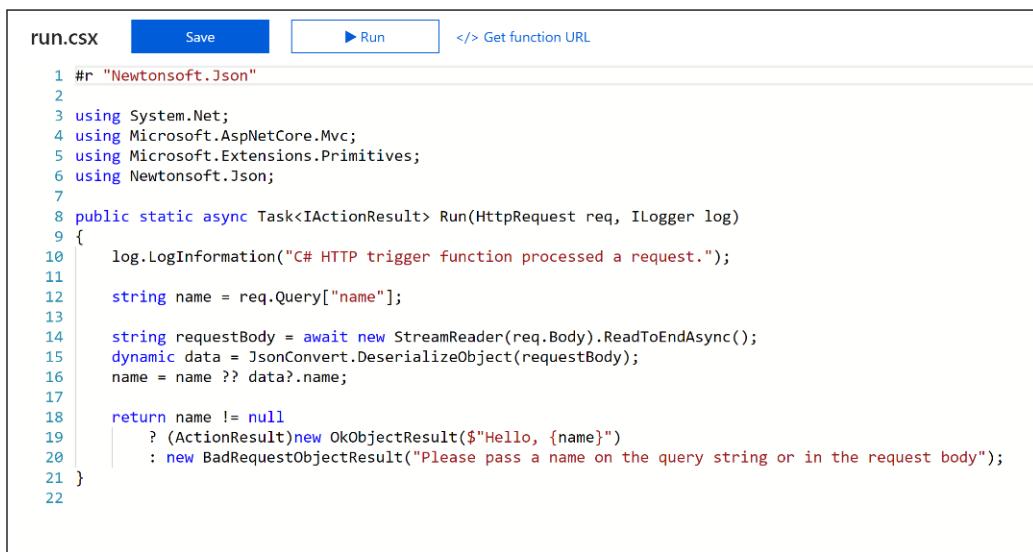
Design e implementação de soluções sem servidor

Ao criar um aplicativo Azure Functions, você acessará o painel a seguir após o provisionamento:



The screenshot shows the Azure Functions Overview page for the 'AzureforArchitectsSecondEdition' function app. The app is running in the RiteshSubscription under the AzureforArchitectsSecondEdition resource group. The URL is https://azureforarchitectssecondedition.azurewebsites.net. The 'Configured features' section includes links to Function app settings, Application settings, and Application Insights. A message says 'You have created a function app! Now it is time to add your code...'.

2. Clicar no botão + ao lado de **Functions** mostra um assistente para a criação de uma nova função. Esse assistente mostra as ferramentas que podem ser usadas para criar o Azure Functions. Selecione **No portal** como uma opção e clique no botão **Continuar** para ir para a próxima tela, que exibe vários tipos de modelo, incluindo **WebHook + API**, **Timer** e **Mais modelos...**. Selecione **WebHook + API** e clique no botão **Criar**. Isso criará a função com código scaffolding e a estrutura para começar. Esse código de scaffolding também é gerado para associações e gatilhos padrão:



The screenshot shows the code editor for a C# HTTP trigger function named 'run.csx'. The code uses the NewtonSoft.Json library to handle JSON requests. It logs an information message, reads the request body, deserializes it into a dynamic object, and checks if the 'name' parameter is present. If not, it returns a bad request error; otherwise, it returns a success response with the name.

```
run.csx
Save Run </> Get function URL

1 #r "Newtonsoft.Json"
2
3 using System.Net;
4 using Microsoft.AspNetCore.Mvc;
5 using Microsoft.Extensions.Primitives;
6 using Newtonsoft.Json;
7
8 public static async Task<IActionResult> Run(HttpContext req, ILogger log)
9 {
10     log.LogInformation("C# HTTP trigger function processed a request.");
11
12     string name = req.Query["name"];
13
14     string requestBody = await new StreamReader(req.Body).ReadToEndAsync();
15     dynamic data = JsonConvert.DeserializeObject(requestBody);
16     name = name ?? data?.name;
17
18     return name != null
19         ? (ActionResult)new OkObjectResult($"Hello, {name}")
20         : new BadRequestObjectResult("Please pass a name on the query string or in the request body");
21 }
22
```

3. Criar essa função fornece uma função completa de criação de ambiente integrado junto com algum código. Esse código obtém o conteúdo bruto do parâmetro req de entrada, que é preenchido pelo tempo de execução do Azure com dados de entrada (cadeia de caracteres de consulta, valores de formulário e assim por diante). Pode haver vários tipos de dados dentro deste parâmetro de entrada e, nessa função, um único valor é extraído. O valor é convertido do objeto JSON para .NET e, com base na presença ou ausência do parâmetro Name, a resposta apropriada é retornada para o usuário.
4. Essa função pode ser invocada usando uma solicitação HTTP do navegador. O URL dessa função está disponível desde o ambiente e é composta do nome do aplicativo da função junto com o nome da função. O formato é `https://<<function app name>>.azurewebsites.net/api/<<function name>>`. Nesse caso, o URL será `https://azureforarchitects.azurewebsites.net/api/FirstFunction`.
5. Para enviar parâmetros para essa função, os parâmetros de cadeia de caracteres de consulta adicionais podem ser acrescentados ao final do URL. Por exemplo, para enviar parâmetros name para essa função, o `https://azureforarchitects.azurewebsites.net/api/FirstFunction?name=ritesh` URL pode ser usado. A saída da função é mostrada na captura de tela a seguir:



6. Para funções com base em HTTP, o Azure Functions já fornece gatilhos e associações no arquivo `function.json`, como mostrado aqui. Esse arquivo é usado para definir todos os gatilhos e associações no nível da função e há um associado a cada função:

```

function.json Save Run </> Get function URL
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 }

```

O modelo HTTP cria um gatilho para todas as solicitações recebidas. O gatilho invoca o Azure Functions e transmite todos os dados de entrada e cargas como um parâmetro denominado `req`. Esse parâmetro está disponível no Azure Functions. A resposta da função é uma associação que leva a saída da variável `res` do Azure Functions e a envia de volta para o canal HTTP como resposta.

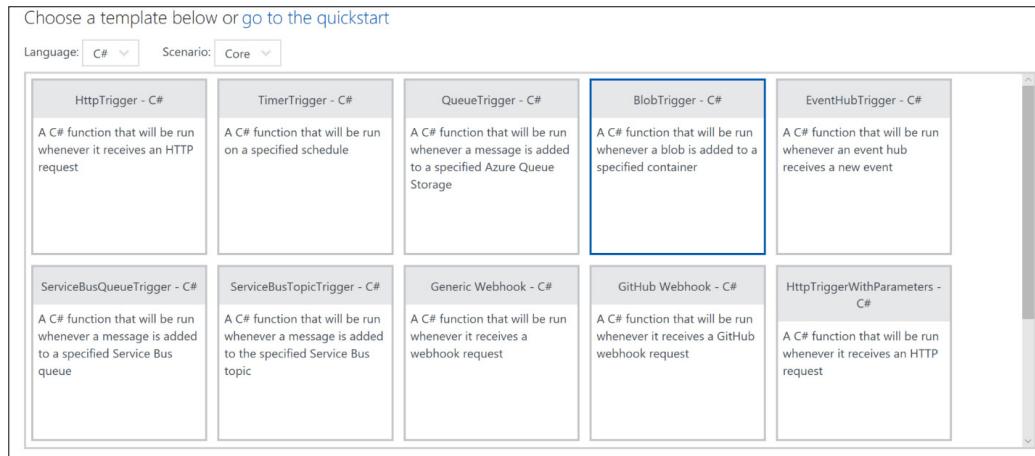
Como criar uma função orientada a eventos

Neste exemplo, um Azure Functions será criado e conectado à conta do armazenamento do Azure. A conta de armazenamento tem um contêiner para armazenar todos os arquivos de blob. O nome da conta de armazenamento é `incomingfiles` e o contêiner é `orders`, conforme mostrado na captura de tela a seguir:

The screenshot shows the Azure Blob service interface for the 'incomingfiles' storage account. At the top, there's a header with the account name and a 'Container' button. Below the header, detailed information about the 'orders' container is displayed, including its status as 'Available' and location as 'West Central US'. A search bar at the bottom allows for searching by container prefix. A table below lists the single container named 'orders'.

NAME
orders

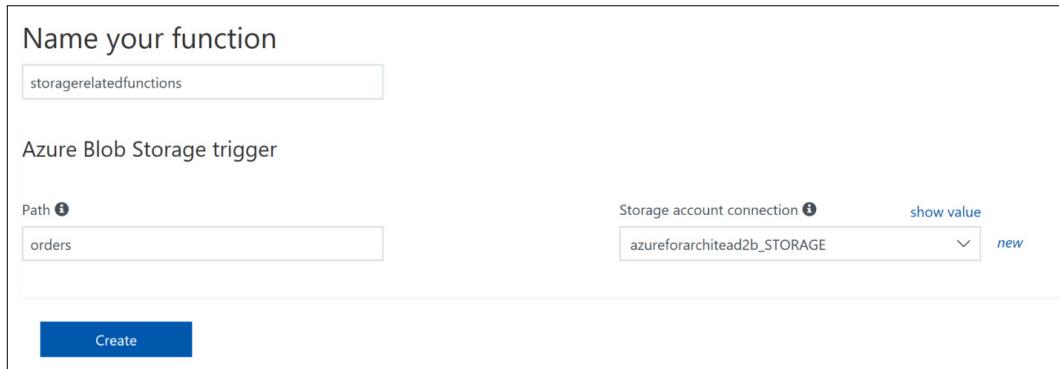
Criar um novo Azure Functions no portal do Azure.



Agora, esse Azure Functions não tem conectividade com a conta de armazenamento. O Azure Functions precisa de informações de conexão para a conta de armazenamento e está disponível na guia **Teclas de acesso** na conta de armazenamento. A mesma informação pode ser obtida usando o ambiente do editor do Azure Functions. Na verdade, esse ambiente permite a criação de uma nova conta de armazenamento do mesmo ambiente de editor.

Isso pode ser adicionado usando o novo botão ao lado do tipo de entrada de **Conexão de conta de armazenamento**. Ele permite selecionar uma conta de armazenamento existente ou criar uma nova conta de armazenamento. Como eu já tenho algumas contas de armazenamento, estou reutilizando-as. Você deve criar uma conta de armazenamento do Azure separada. Selecionar uma conta de armazenamento atualizará as configurações na seção **appsettings** com a cadeia de conexão adicionada a ela.

Verifique se um contêiner já existe no serviço de blob da conta de armazenamento do Azure de destino. A entrada do caminho se refere ao caminho para o contêiner. Neste caso, o contêiner de **pedidos** já existe na conta de armazenamento. O botão **Criar** mostrado aqui provisionará a nova função que monitora o contêiner de conta de armazenamento:



O código do Azure Functions é:

```
public static void Run(Stream myBlob, TraceWriter log)
{
    log.Info($"C# Blob trigger function Processed blob\n \n Size
{myBlob.Length} Bytes");
}
```

As associações são mostradas aqui:

```
{
  "bindings": [
    {
      "name": "myBlob",
      "type": "blobTrigger",
      "direction": "in",
      "path": "orders",
      "connection": "azureforarchitead2b_STORAGE"
    }
  ],
  "disabled": false
}
```

Agora, fazer upload de qualquer arquivo blob no contêiner de pedidos deve disparar a função:

The screenshot shows the Azure Functions developer portal interface. At the top, there is a file named "run.csx" with a "Save" button and a "Run" button. Below the file name, the code for the function is displayed:

```
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 "Logs" section. It contains the following log entries:

```
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
```

Proxies do Azure Functions

Os proxies do Azure Functions são uma das adições mais recentes ao Azure Functions. Depois de começar a usar o Azure Functions, haverá um momento quando haverá muita implementação de função e será difícil integrar as funções em um fluxo de trabalho. Em vez de deixar clientes tecer essas funções juntas, os proxies do Azure podem ser usados. Os proxies ajudam ao fornecer aos clientes um único URL de função e, então, invocar diversos Azure Functions nos bastidores para concluir fluxos de trabalho.

É importante compreender que proxies sejam aplicáveis nos casos onde as funções aceitem solicitações sob demanda em vez de orientado por eventos. Essas funções internas, conectadas a proxies podem ser ficar em um único aplicativo de função ou em vários aplicativos separados. Os proxies obtêm solicitações de clientes, convertem, substituem e aumentam a carga e as enviam para as funções internas de back-end. Assim que receberem uma resposta dessas funções, eles podem novamente converter, substituir e aumentar a resposta e devolvê-la ao cliente.



Mais informações sobre proxies do Azure Functions podem ser encontradas em <https://docs.microsoft.com/pt-br/azure/azure-functions/functions-proxies>.



Noções básicas sobre fluxos de trabalho

Um fluxo de trabalho é uma série de etapas ou atividades que são executadas em paralelo ou em sequência ou em uma combinação dessas duas opções. Como as atividades podem ser executadas em paralelo, elas podem realizar trabalhos em vários serviços ao mesmo tempo sem serem bloqueadas.

Veja a seguir os recursos dos fluxos de trabalho:

- **Capacidade de recuperação de falhas:** um fluxo de trabalho pode ser hidratado, o que significa que seu estado pode ser salvo em pontos bem definidos dentro do fluxo de trabalho. Se o fluxo de trabalho falhar, ele será iniciado novamente a partir do último estado salvo, em vez do início. Esse recurso também é conhecido como pontos de verificação.
- **Longa duração:** os fluxos de trabalho geralmente são de longa duração por natureza. Eles podem ser executados por minutos a horas ou dias. Eles novamente salvam o estado quando estão aguardando uma ação externa para serem concluídos e podem ser iniciados novamente a partir do último estado salvo depois que a atividade externa for concluída.
- **Siga etapas em paralelo:** já discutimos esse ponto, mas este é um grande benefício de um fluxo de trabalho em comparação com a programação de sequência.
- **Estado de manutenção:** os fluxos de trabalho são normalmente "com estado" por natureza. Eles mantêm o estado para que, em caso de falha ou reinicialização de um fluxo de trabalho, ele não comece do início, mas possa continuar a partir desses pontos de verificação.

O Azure Functions precisa terminar a execução em cinco minutos. Isso significa que eles são de curta duração, e compor fluxos de trabalho com eles pode realmente ser difícil. Existe a possibilidade de implementar várias funções e garantir que elas sejam coordenadas para serem executadas uma após a outra. Até mesmo esse procedimento tem suas limitações.

Ele é mais como um corte e precisa de esforço, design e operações consideráveis. Todas as funções precisarão ter entradas vinculadas às saídas de sua função anterior, e cada função ainda será executada por alguns minutos. Outra desvantagem de implementar fluxos de trabalho dessa maneira é que é extremamente difícil entender sua conectividade, dependência e sequência de execução com uma visão rápida. Em suma, o Azure Functions é adequado para implementar uma única responsabilidade de curta duração. Ele não é adequado para a implementação de fluxos de trabalho.

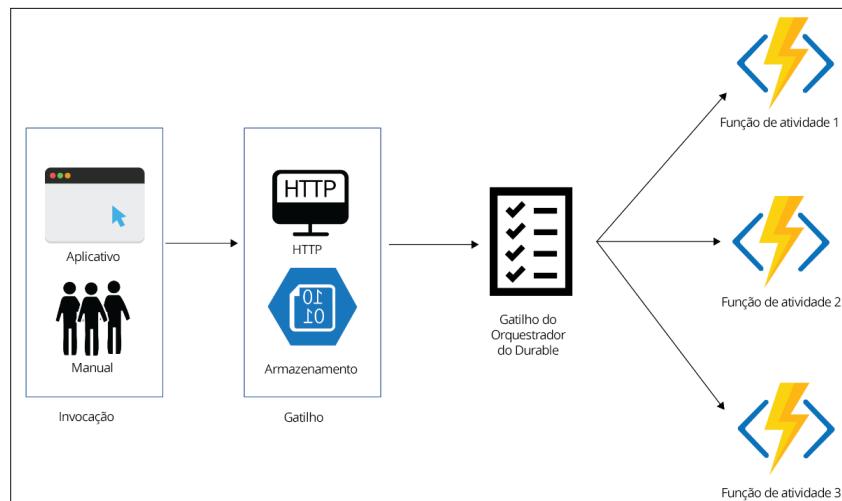
Outro recurso importante do Azure Functions é que ele não tem estado. Isso significa que as funções são passageiras e só ficam disponíveis no momento em que estão sendo executadas. Uma execução de instância de função não tem relação com a instância anterior ou consequente em execução. Isso significa que não há possibilidade de armazenar o estado com o Azure Functions.

O Azure Functions pode ser executado em qualquer servidor nos bastidores, e esse servidor pode não ser o mesmo utilizado nas execuções subsequentes. Portanto, as funções não podem armazenar seu estado no servidor em que são executadas.

A única maneira de salvar o estado no Azure functions é armazenar o estado externamente em armazenamentos de dados, como o Cosmos DB, bancos de dados SQL ou Armazenamento do Azure. Entretanto, esse design terá problemas significativos de performance, pois cada vez que a função for executada, ele precisará se conectar a um armazenamento de dados, recuperar e gravar nele. Agora que nós entendemos os fluxos de trabalho, prosseguiremos com o Durable Functions.

Durable Functions

O Durable Functions é uma das adições mais recentes ao Azure Functions. Ele preenche um espaço que existiu por algum tempo:



O Azure Durable Functions pode ser invocado por qualquer gatilho fornecido pelo Azure Functions. Esses gatilhos incluem HTTP, Armazenamento de Blobs, armazenamento de tabelas, filas do Barramento de Serviço e muito mais. Eles podem ser acionados manualmente por alguém com acesso a eles ou por um aplicativo. O diagrama anterior mostra alguns gatilhos como exemplo. Eles também são conhecidos como Durable Functions iniciais. O Durable Functions inicial invoca o **gatilho do orquestrador do Durable**, que contém a lógica principal para a orquestração e orquestra a invocação das funções da atividade. Essas funções de atividade podem ser chamadas com ou sem um mecanismo de repetição. O Durable Functions pode ajudar a resolver muitos desafios e a fornecer recursos para escrever funções que podem:

- Executar funções de execução longa
- Manter o estado
- Executar funções filhas em paralelo ou em sequência
- Recuperar-se da falha facilmente
- Orquestrar a execução de funções em um fluxo de trabalho

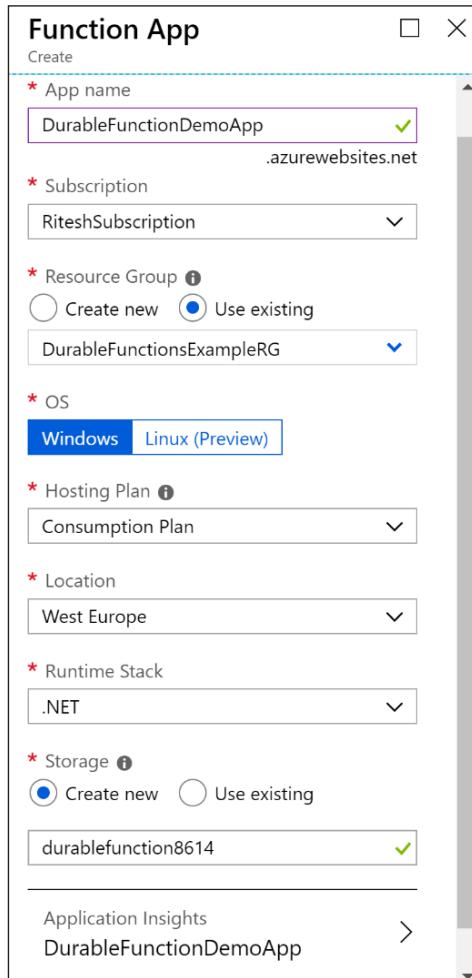
Etapas para criar um Durable Functions

Veja a seguir as etapas para criar um Durable Functions:

1. Navegue até o portal do Azure e clique em **Grupos de recursos** no menu à esquerda.
2. Clique no botão **+Adicionar** no menu superior para criar um novo grupo de recursos.
3. Forneça as informações do grupo de recursos no formulário resultante e clique no botão **Criar**, como mostrado aqui:

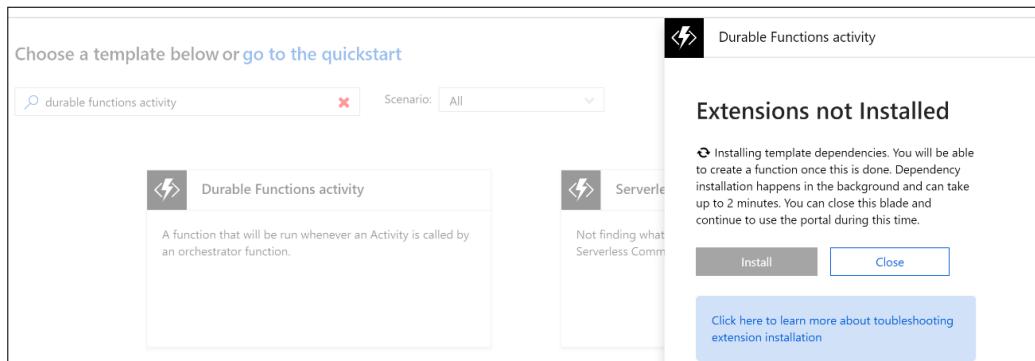
The screenshot shows the 'Create a resource group' wizard. At the top, there are three tabs: 'Basics' (which is selected), 'Tags', and 'Review + Create'. Below the tabs, a descriptive text explains what a Resource group is: 'Resource group - A container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization.' A 'Learn more' link is also present. The 'PROJECT DETAILS' section contains two fields: 'Subscription' (set to 'RiteshSubscription') and 'Resource group' (set to 'DurableFunctionsExampleRG'). The 'RESOURCE DETAILS' section contains one field: 'Region' (set to 'West Europe').

4. Navegue até o grupo de recursos recém-criado e adicione um novo aplicativo de função. Para isso, clique no botão **+Adicionar** no menu superior e procure pelo aplicativo de função na caixa de pesquisa resultante.
5. Selecione **Aplicativo de função** e clique no botão **Criar**. Preencha o formulário de aplicativo de função resultante e clique no botão **Criar**, como mostrado aqui:

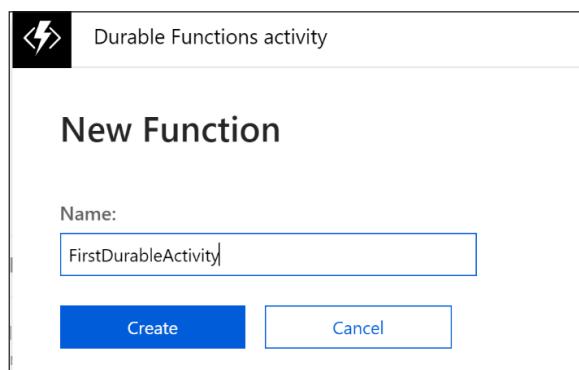


6. Após a criação do aplicativo de função, clique no botão **Functions +** no menu à esquerda para criar uma nova função. Selecione **No portal** | **Continuar** | **Mais modelos...** | **Conclua e visualize os modelos**.

7. Na caixa de pesquisa resultante, pesquise Atividade do Durable Functions e selecione o modelo de **Atividade do Durable Functions**. Se a extensão Microsoft.Azure.WebJobs.Extensions.DurableTask ainda não estiver instalada no aplicativo de funções, ele pedirá que você a instale.
8. Clique no botão **Instalar** para instalar a extensão:



9. As atividades do Durable Functions são funções invocadas pela função do orquestrador principal. Geralmente há uma função de orquestrador principal e várias atividades do Durable Functions. Após a instalação da extensão, forneça um nome para a função e escreva um código que faça algo útil, como enviar um email ou SMS, conectar-se a sistemas externos e executar lógica ou executar serviços usando seus pontos de extremidade, como os Serviços Cognitivos. Deve haver uma atividade do Durable Functions para cada trabalho no fluxo de trabalho:



10. O código gerado deve ser modificado para refletir o código, conforme mostrado na captura de tela a seguir:

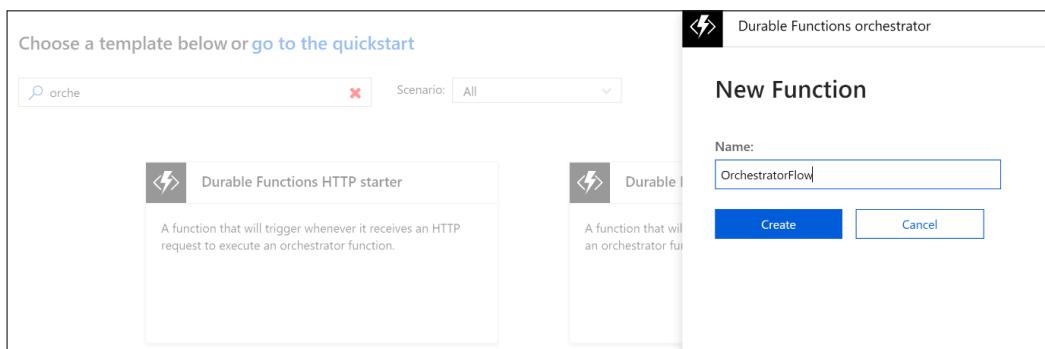
```
#r "Microsoft.Azure.WebJobs.Extensions.DurableTask"

public static async Task< string> Run(string name)
{
    await Task.Delay(10000);
    return $"Hello {name}!";
}
```

A única alteração é a adição de uma única linha de código que faz com que a função aguarde por 10 segundos.

De maneira semelhante, crie outra função de atividade com o mesmo código e atribua outro nome a ela: por exemplo, SecondDurableActivity.

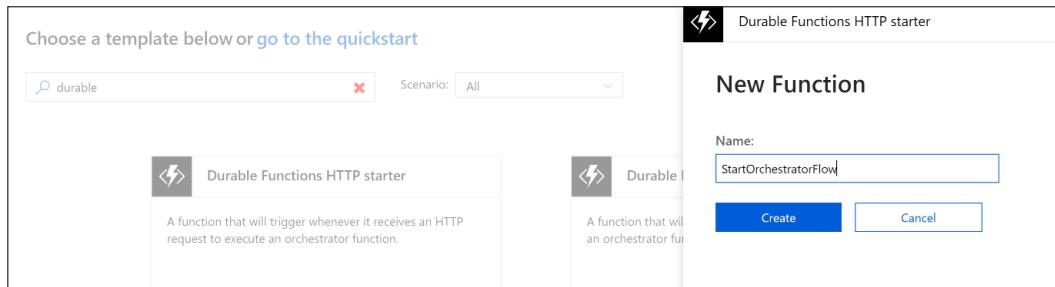
Crie uma nova função de orquestrador chamada OrchestratorFlow, conforme mostrado na captura de tela a seguir:



Altere o código gerado para o código mostrado aqui:

```
#r "Microsoft.Azure.WebJobs.Extensions.DurableTask"
public static async Task<List<string>> Run(DurableOrchestrationContext context)
{
    var outputs = new List<string>();
    // Replace "Hello" with the name of your Durable Activity Function.
    outputs.Add(await context.CallActivityAsync<string>("FirstDurableActivity", "Tokyo"));
    outputs.Add(await context.CallActivityAsync<string>("SecondDurableActivity", "Seattle"));
    // returns ["Hello Tokyo!", "Hello Seattle!", "Hello London!"]
    return outputs;
}
```

Em seguida, crie uma função de gatilho do orquestrador. Há um Durable Functions inicial HTTP fornecido pronto para uso. Entretanto, é possível criar um Durable Functions que se baseie em gatilhos externos, como um arquivo adicionado ao contêiner de Armazenamento de Blobs do Azure. Aqui, estamos criando uma função simples que inicia o fluxo de trabalho do orquestrador chamando o Durable Functions do orquestrador OrchestratorFlow. Podemos iniciar o fluxo de trabalho invocando-o usando um navegador ou ferramentas como o Postman:



O código para isso está listado aqui:

```
#r "Microsoft.Azure.WebJobs.Extensions.DurableTask"
#r "Newtonsoft.Json"
using System.Net;
public static async Task<HttpResponseMessage> Run(
    HttpRequestMessage req,
    DurableOrchestrationClient starter,
    ILogger log)
{
    // Function input comes from the request content.
    dynamic eventData = await req.Content.ReadAsAsync<object>();
    // Pass the function name as part of the route
    string instanceId = await starter.StartNewAsync("OrchestratorFlow", eventData);
    log.LogInformation($"Started orchestration with ID = '{instanceId}'.");
    return starter.CreateCheckStatusResponse(req, instanceId);
}
```

O arquivo `function.json` que contém a declaração de gatilhos e associações é modificado, e o código resultante é mostrado aqui:

```
{
  "bindings": [
    {
      "authLevel": "function",
```

```

    "name": "req",
    "type": "httpTrigger",
    "direction": "in",
    "route": "orchestrators",
    "methods": [
        "post",
        "get"
    ],
},
{
    "name": "$return",
    "direction": "out"
},
{
    "name": "starter",
    "type": "orchestrationClient",
    "direction": "in"
}
],
"disabled": false
}

```

Clique no link **Obter URL da função** na interface do usuário e copie o URL resultante.

Vamos invocar esse URL usando uma ferramenta conhecida como **Postman** (é possível fazer download dela em <https://www.getpostman.com/>). Essa atividade é mostrada na captura de tela a seguir:

The screenshot shows the Postman interface with the following details:

- Method:** POST
- URL:** https://durablefunctiondemopapp.azurewebsites.net/api/orchestrators?code=q0lW1/KoWhx27ykdB7BA2cxF7ECpZOlkCx1xElaOWb9KKCqcH0Lvw==
- Params:** A table with one row: **code** (Value: q0lW1/KoWhx27ykdB7BA2cxF7ECpZOlkCx1xElaOWb9KKCqcH0Lvw==)
- Body:** A JSON object with the following structure:


```

1 ~ {
2   "id": "19701537d174d0b65e662f787f90b7",
3   "status": "Running", "url": "https://durablefunctiondemopapp.azurewebsites.net/runtime/webhooks/durabletask/instances/19701537d174d0b65e662f787f90b7/taskHub=DurableFunctionsHub&connection=Storage&code=Evd932qljLjRf6axFkR3625MOixa3f/rdvnZcLMkEj1Hhrqkw=",
4   "sendEventPostUri": "https://durablefunctiondemopapp.azurewebsites.net/runtime/webhooks/durabletask/instances/19701537d174d0b65e662f787f90b7/raiseEvent?eventName=taskHub",
5   "terminatePostUri": "https://durablefunctiondemopapp.azurewebsites.net/runtime/webhooks/durabletask/instances/19701537d174d0b65e662f787f90b7/terminate?reason=(text)&taskHub",
6   "rewindPostUri": "https://durablefunctiondemopapp.azurewebsites.net/runtime/webhooks/durabletask/instances/19701537d174d0b65e662f787f90b7/rewind?reason=(text)&taskHub"
}
      
```
- Status:** 202 Accepted
- Time:** 5206 ms
- Size:** 1.51 KB

Design e implementação de soluções sem servidor

Lembre-se de que quatro URLs são gerados quando você inicia um orquestrador:

- O URL `statusQueryGetUri` é usado para localizar o status atual do orquestrador. Clicar nesse URL no Postman abre uma nova guia e mostra o status do fluxo de trabalho:

```
1 > [
  2   "instanceId": "19701537d17d4db0b65e662f787f90b7",
  3   "runStatus": "Completed",
  4   "input": null,
  5   "output": [
  6     {
  7       "text": "Hello Tokyo!",
  8       "text": "Hello Seattle!"
  9     }
  10  ],
  11  "createdTime": "2019-01-31T13:23:52Z",
  12  "lastUpdatedTime": "2019-01-31T13:24:14Z"
]
```

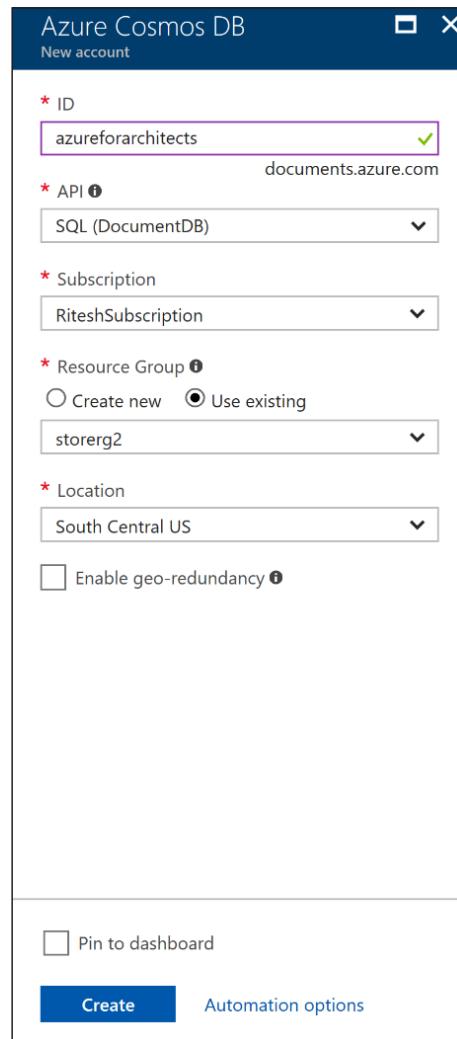
- O URL `terminatePostUri` é usado para interromper uma função do orquestrador já em execução.
- O URL `sendEventPostUri` é usado para postar um evento em um Durable Functions suspenso. O Durable Functions pode ser suspenso se estiver aguardando um evento externo. Esse URL é usado nesses casos.
- O URL `rewindPostUri` é usado para postar uma mensagem para retroceder uma função do orquestrador.

Como criar uma arquitetura conectada com funções

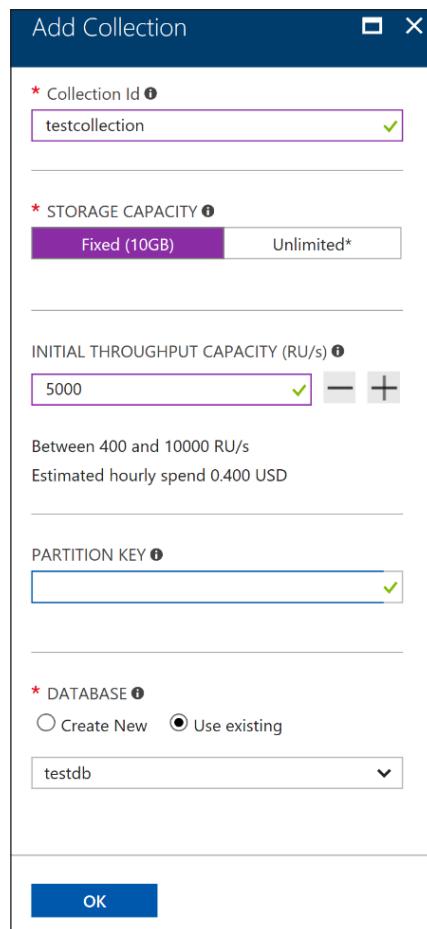
Uma arquitetura conectada com funções refere-se à criação de múltiplas funções, segundo a qual a saída de uma função aciona outra função e fornece dados para a próxima função executar sua lógica. Nesta seção, vamos continuar com o cenário anterior da conta de armazenamento. Nesse caso, a saída da função disparada usando arquivos de blob do Armazenamento do Azure gravarão o tamanho do arquivo no Azure Cosmos DB.

A configuração do Cosmos DB é mostrada a seguir. Por padrão, não há nenhuma coleção criada no Cosmos DB.

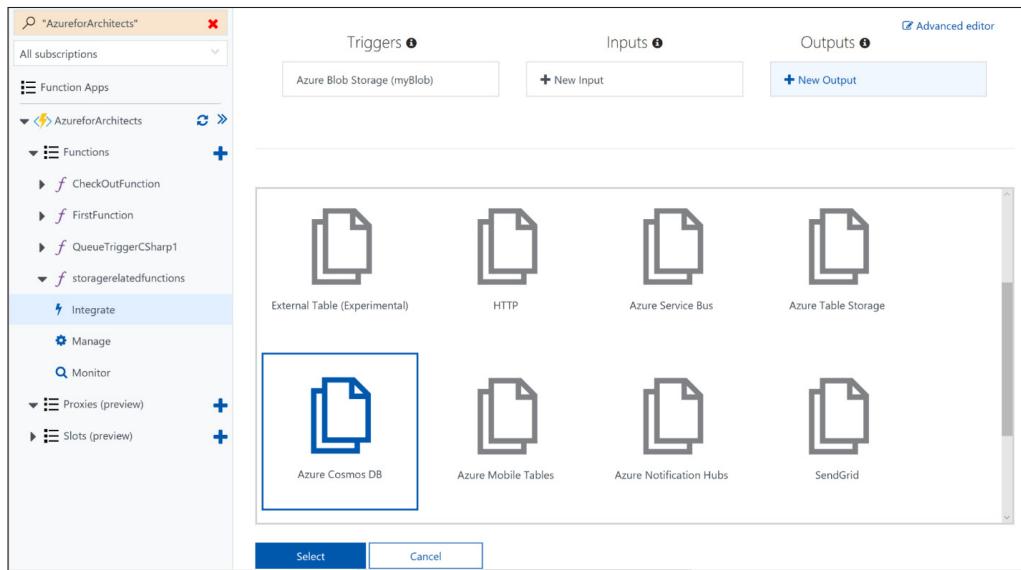
Uma coleção será criada automaticamente ao criar uma função que será acionada quando o Cosmos DB obtiver dados:



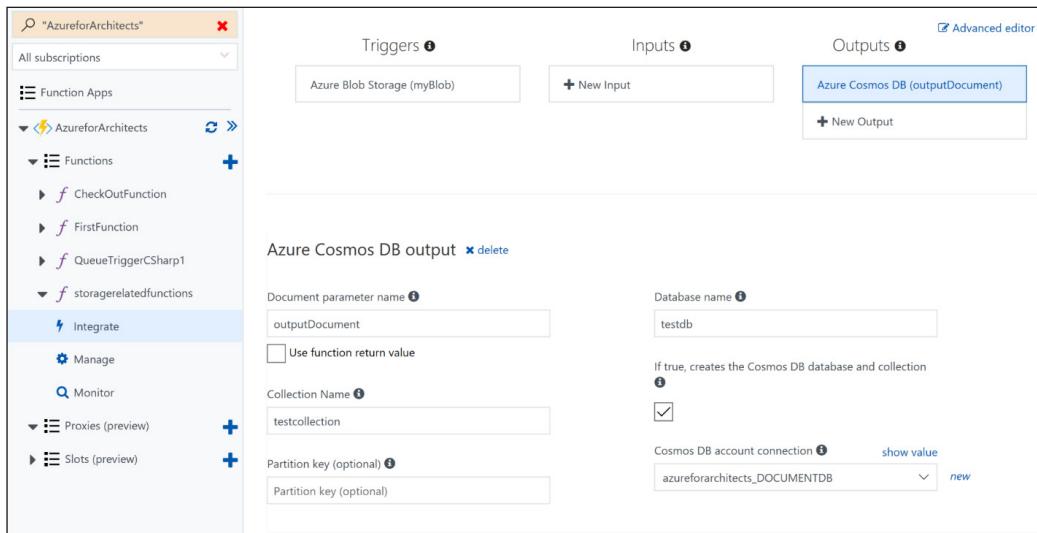
Crie um novo banco de dados, `testdb`, no Cosmos DB e uma nova coleção denominada `testcollection` dentro dele. Você precisa do nome do banco de dados e da coleção ao configurar o Azure Functions:



É hora de revisitar a função `storagerelatedfunctions` e alterar sua associação para retornar o tamanho dos dados do arquivo carregado. Esse valor retornado será gravado no Cosmos DB. Isso exigirá uma mudança para as associações, bem como com um responsável adicional por capturar os valores de saída. Essa associação vai eventualmente gravar na coleção do Cosmos DB. Navegue até a guia **Integrar**, clique no botão **Nova Saída** abaixo do rótulo **Saídas** e selecione **Azure Cosmos DB**:



Forneça os nomes apropriados para o banco de dados e a coleção (marque a caixa de seleção para criar a coleção, se ela não existir), clique no botão **Novo** para selecionar nosso recém-criado Azure Cosmos DB e deixe o nome do parâmetro como `outputDocument`:



Modifique a função conforme mostrado na captura de tela a seguir:

The screenshot shows the Azure Functions developer portal interface. On the left, there's a sidebar with a search bar, a dropdown for 'All subscriptions', and a tree view showing 'Function Apps' under 'AzureforArchitects'. Under 'Functions', two functions are listed: 'CheckOutFunction' and 'FirstFunction'. The main area is a code editor titled 'RUN.CSX' with the following C# code:

```
1 public static void Run(string myBlob, out object outputDocument, TraceWriter log)
2 {
3     log.Info($"C# blob trigger function processed: {myBlob.Length}");
4     log.Info($"C# blob trigger function processed: {myBlob}");
5     outputDocument = new {
6         id = myBlob.Length.ToString(),
7         len = myBlob.Length,
8         data = myBlob
9     };
10 }
11 }
```

Agora, o upload de um arquivo novo para a coleção de pedidos na conta de armazenamento do Azure executará uma função que gravará na coleção do Azure Cosmos DB. Outra função pode ser escrita com a recém-criada conta do Azure Cosmos DB como uma associação de gatilho. Ele fornecerá o tamanho dos arquivos e a função poderá agir neles. Isso é mostrado aqui:

The screenshot shows the Azure Cosmos DB portal. On the left, there's a navigation bar with 'Create', 'Upload', 'More', 'Save', 'Discard', 'Delete', 'Refresh', and 'Properties'. Below it, a dropdown shows 'testcollection'. A search bar says 'Search by Document id or Partition Key'. A table lists documents with columns 'ID' and 'data'. The 'data' column shows the JSON representation of a document with fields 'id', 'len', and 'data'. The 'data' field contains XML data.

ID	data
137	{ "id": "137", "len": 137, "data": "<?xml version=\\"1.0\\" encoding=\\"utf-8\\"?>\r\n<configuration>\r\n <Ritesh id=\\"1\\">\r\n aaaa</Ritesh>\r\n <Ritesh id=\\"2\\">\r\n bbbb</Ritesh>\r\n</configuration>" }
1580	
603	

Resumo

A evolução das funções dos métodos tradicionais levou à criação da arquitetura de baixo acoplamento, de evolução independente, autossuficiente sem servidor que era apenas um conceito antigamente. O Functions é uma unidade de implantação. Ele fornece um ambiente que não precisa ser gerenciado pelo usuário. Eles só têm que se preocupar com o código escrito para a funcionalidade. O Azure fornece uma plataforma madura para hospedagem de funções e as integra perfeitamente, com base em eventos ou sob demanda. Quase todos os recursos no Azure podem participar de uma arquitetura composta por Azure Functions. O futuro é das funções, à medida que mais e mais organizações quiserem se manter afastadas do gerenciamento de infraestruturas e plataformas. Elas desejam descarregar isso nos provedores de nuvem. O Azure Functions é um recurso essencial a ser dominado por todos os arquitetos que lidam com o Azure.

O próximo capítulo se baseará no que aprendemos neste, pois explorará outras tecnologias sem servidor, como Aplicativos Lógicos do Azure e Grades de Eventos. Ele também mostrará uma solução completa de ponta a ponta usando todas essas tecnologias.

7

Soluções de integração do Azure

Este capítulo é uma continuação do capítulo anterior. No Capítulo 6, *Design e implementação de soluções sem servidor*, discutimos Azure Functions de computação sem servidor e, neste capítulo, continuaremos nossa discussão sobre tecnologias sem servidor e abordaremos as Grades de Eventos do Azure como parte de eventos sem servidor e também os Aplicativos Lógicos do Azure como parte de fluxos de trabalho sem servidor. Uma solução completa de ponta a ponta também será criada usando vários serviços do Azure, como Automação, Aplicativos Lógicos, Functions, Grade de Eventos, SendGrid, Twilio, PowerShell, Azure AD e Cofres de Chaves.

Os tópicos a seguir serão abordados neste capítulo:

- Grade de Eventos do Azure
- Aplicativos Lógicos do Azure
- Criação de uma solução de ponta a ponta usando tecnologias sem servidor

Grade de Eventos do Azure

A Grade de Eventos do Azure é um serviço relativamente novo. Também foi chamado de uma plataforma de eventos sem servidor. Ele ajuda na criação de aplicativos baseados em eventos (também conhecido como **design controlado por eventos**). É importante entender o que são eventos e como lidamos com eles antes da Grade de Eventos. Um evento é *algo que aconteceu* – ou seja, uma atividade que mudou o estado de um assunto. Quando um assunto passa por uma alteração em seu estado, geralmente gera um evento.

Os eventos normalmente seguem o padrão de publicação/assinatura (também conhecido popularmente **como o padrão pub/sub**), no qual um assunto gera um evento devido à sua alteração de estado, e esse evento pode então ser inscrito por várias partes interessadas, também conhecidas como **assinantes**. O trabalho do evento é notificar os assinantes de tais mudanças e também fornecê-los com dados como parte de seu contexto. Os assinantes podem tomar qualquer ação que considerem necessárias, o que varia de assinante para assinante.

Antes da Grade de Eventos, não havia nenhum serviço que poderia ser descrito como uma plataforma de eventos em tempo real. Havia serviços separados, e cada um forneceu seu próprio mecanismo para manipular eventos.

Por exemplo, o Log Analytics, também conhecido como **Operations Management Suite (OMS)**, fornece uma infraestrutura para capturar logs de ambiente e telemetria em que os alertas podem ser gerados. Esses alertas podem ser usados para executar um runbook, um webhook ou uma função. Isso é próximo ao tempo real, mas eles não são completamente tempo real. Além disso, foi bastante complicado capturar logs individuais e agir sobre eles. Da mesma forma, há o Application Insights, que fornece recursos semelhantes para o Log Analytics, mas para aplicativos.

Há outros logs, como os logs de atividades e logs de diagnóstico, mas, novamente, eles dependem de princípios semelhantes aos de outros recursos relacionados ao log. As soluções são implantadas em vários grupos de recursos em várias regiões, e os eventos gerados de qualquer um deles devem estar disponíveis para os recursos implantados em outro lugar.

A Grade de Eventos remove todas as barreiras e, como resultado, os eventos podem ser gerados pela maioria dos recursos (cada vez mais estão se tornando disponíveis) e até mesmo eventos personalizados podem ser gerados. Esses eventos podem então ser assinados por qualquer recurso, em qualquer região e em qualquer grupo de recursos dentro da assinatura.

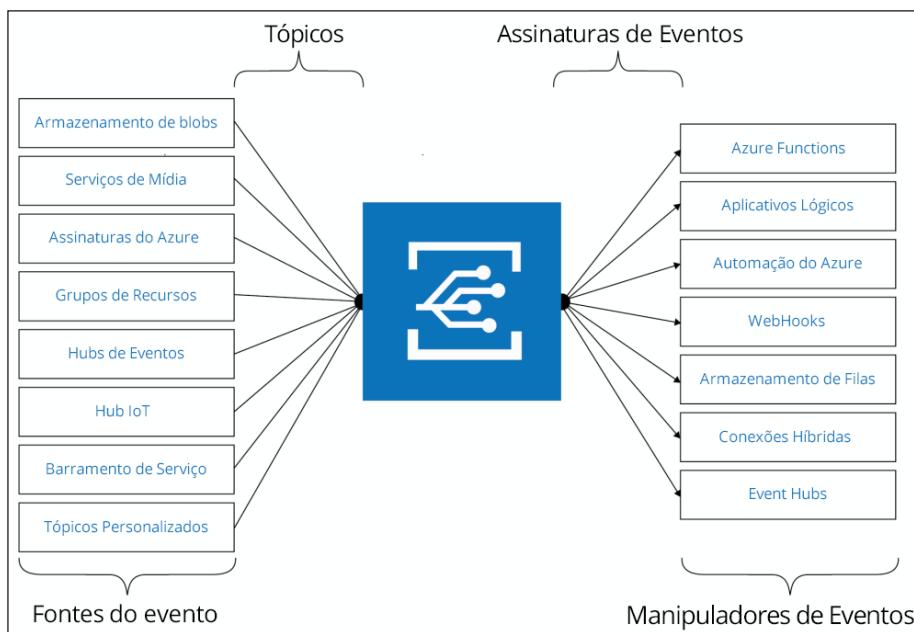
A Grade de Eventos já está estabelecida como parte da infraestrutura do Azure, junto com data centers e redes. Os eventos gerados em uma região podem ser facilmente assinados por recursos em outras regiões, e uma vez que essas redes estejam conectadas, será extremamente eficiente para a entrega de eventos aos assinantes.

A arquitetura da Grade de Eventos

A arquitetura da Grade de Eventos se baseia em tópicos do barramento de serviço. Os tópicos do barramento de serviço, como já sabemos, baseiam-se no mecanismo de publicação/assinatura. Há editores de eventos e há consumidores de eventos; no entanto, pode haver vários assinantes para o mesmo evento.

O editor de um evento pode ser um recurso do Azure, como armazenamento de Blobs, hubs da **Internet das Coisas (IoT)** e muitos outros. Esses editores também são conhecidos como origens de eventos. Esses editores usam tópicos do Azure prontos para uso para enviar seus eventos para a Grade de Eventos. Não é necessário configurar o recurso ou o tópico. Os eventos gerados por recursos do Azure já estão usando internamente tópicos para enviar seus eventos para a Grade de Eventos. Assim que o evento atingir a grade, poderá ser consumido pelos assinantes.

Os assinantes, ou consumidores, são recursos interessados em eventos e desejam executar uma ação com base nesses eventos. Esses assinantes fornecem um manipulador de eventos quando eles assinam o tópico. Os manipuladores de eventos podem ser funções do Azure, web hooks personalizados, aplicativos lógicos ou outros recursos. As origens de eventos e os assinantes que executam manipuladores de eventos são mostrados no diagrama a seguir:



Quando um evento atinge um tópico, vários manipuladores de eventos podem ser executados simultaneamente, cada um executando sua própria ação.

Também é possível criar um evento personalizado e enviar um tópico personalizado para a Grade de Eventos. A Grade de Eventos fornece recursos para a criação de tópicos personalizados, e esses tópicos são anexados automaticamente à Grade de Eventos. Esses tópicos conhecem o armazenamento da Grade de Eventos e enviam automaticamente suas mensagens para ele. Os tópicos personalizados têm duas propriedades importantes, da seguinte maneira:

- **Um ponto de extremidade:** é o ponto de extremidade do tópico. Os editores e as origens de eventos usam esse ponto de extremidade para enviar e publicar seus eventos na Grade de Eventos. Em outras palavras, os tópicos são reconhecidos usando seus pontos de extremidade.
- **Chaves:** os tópicos personalizados fornecem um par de chaves. Essas chaves permitem a segurança para o consumo do ponto de extremidade. Somente os editores com essas chaves podem enviar e publicar suas mensagens na Grade de Eventos.

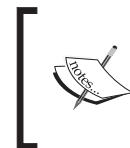
Cada evento tem um tipo de evento e é reconhecido por ele. Por exemplo, o armazenamento de blobs fornece tipos de eventos, como blobAdded e blobDeleted. Os tópicos personalizados podem ser usados para enviar um evento personalizado definido, como um evento personalizado do tipo KeyVaultSecretExpired.

Por outro lado, os assinantes têm a capacidade de aceitar todas as mensagens ou apenas obter eventos com base em filtros. Esses filtros podem ser baseados no tipo de evento ou em outras propriedades dentro da carga do evento.

Cada evento tem pelo menos estas cinco propriedades:

- `id`: é o identificador exclusivo para o evento.
- `eventType`: é o tipo de evento.
- `eventTime`: é a data e hora em que o evento foi gerado.
- `subject`: é uma breve descrição do evento.
- `data`: é um objeto de dicionário e contém dados específicos de recursos ou quaisquer dados personalizados (para tópicos personalizados).

Atualmente, as funcionalidades da Grade de Eventos não estão disponíveis com todos os recursos; no entanto, o Azure está continuamente adicionando mais e mais recursos com a funcionalidade de Grade de Eventos.



Para saber mais sobre os recursos que podem gerar eventos relacionados à grade de evento e manipuladores que podem manipular esses eventos, acesse a <https://docs.microsoft.com/azure/event-grade/visão-geral>.

Eventos de recursos

Nesta seção, as etapas a seguir são fornecidas para criar uma solução na qual os eventos que são gerados pelo armazenamento de Blobs são publicados na Grade de Eventos e, em última análise, roteados para uma função do Azure:

1. Faça logon no portal do Azure e em sua assinatura usando as credenciais adequadas e crie uma nova conta de armazenamento em um grupo de recursos existente ou novo. A conta de armazenamento deve ser StorageV2 ou armazenamento de Blobs. Conforme demonstrado na captura de tela a seguir, a Grade de Eventos não funcionará com StorageV1:

Create storage account

[Basics](#) [Advanced](#) [Tags](#) [Review + create](#)

Azure Storage is a Microsoft-managed service providing cloud storage that is highly available, secure, durable, scalable, and redundant. Azure Storage includes Azure Blobs (objects), Azure Data Lake Storage Gen2, Azure Files, Azure Queues, and Azure Tables. The cost of your storage account depends on the usage and the options you choose below. [Learn more](#)

PROJECT DETAILS

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

* Subscription: RiteshSubscription

* Resource group: azureclitest [Create new](#)

INSTANCE DETAILS

The default deployment model is Resource Manager, which supports the latest Azure features. You may choose to deploy using the classic deployment model instead. [Choose classic deployment model](#)

* Storage account name: storageforeventgrid

* Location: West Europe

Performance: Standard Premium

Account kind: StorageV2 (general purpose v2)

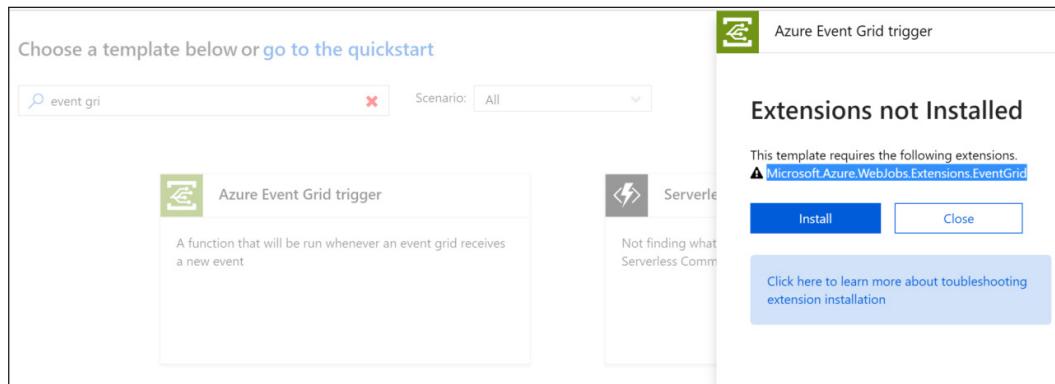
Replication: Read-access geo-redundant storage (RA-GRS)

Access tier (default): Cool Hot

[Review + create](#) [Previous](#) [Next : Advanced >](#)

Soluções de integração do Azure

2. Crie um novo aplicativo de função ou reutilize um aplicativo de função existente para criar uma função do Azure. A função do Azure será hospedada no aplicativo de função.
3. Crie uma nova função usando o modelo de gatilho da Grade de Eventos do Azure. Instale a extensão Microsoft.Azure.WebJobs.Extensions.EventGrid se ainda não estiver instalada, como mostrado na seguinte captura de tela:



4. Nomeie a função StorageEvent Handler e crie-a. O seguinte código padrão gerado será usado como o manipulador de eventos:

run.csx Save Run Add Event Grid subscription

```
1 #r "Microsoft.Azure.EventGrid"
2 using Microsoft.Azure.EventGrid.Models;
3
4 public static void Run(EventGridEvent eventGridEvent, ILogger log)
5 {
6     log.LogInformation(eventGridEvent.Data.ToString());
7 }
8
```

A assinatura de eventos de armazenamento pode ser configurada da **interface do usuário** do Azure Functions (UI), clicando em Adicionar assinatura da Grade de Eventos ou da própria conta de armazenamento propriamente dita.

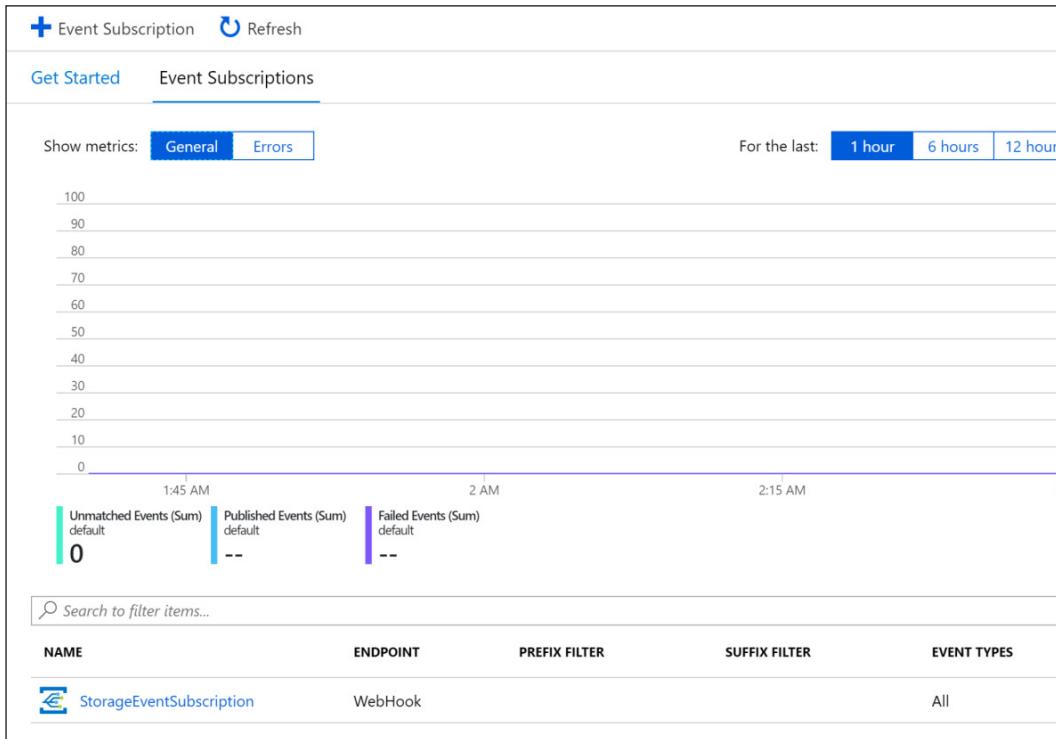
5. Clique no link **Adicionar assinatura de Grade de Eventos** na interface do usuário do Azure Functions para adicionar uma assinatura aos eventos gerados pela conta de armazenamento criada na etapa anterior. Forneça um nome para a assinatura e escolha **Esquema de Eventos** seguido por **Esquema da Grade de Eventos**. Defina **Tipos de Tópico** como **Contas de Armazenamento**, defina uma **Assinatura** apropriada e o grupo de recursos que contém a conta de armazenamento:

The screenshot shows the 'Create Event Subscription' page in the Azure Functions UI. The page has a header 'Home > DurableFunctionDemoApp - StorageEventHandler > Create Event Subscription'. Below the header, there's a section titled 'Create Event Subscription' with a 'Event Grid' icon. There are three tabs: 'Basic' (selected), 'Filters', and 'Additional Features'. A note below the tabs says: 'Event Subscriptions listen for events emitted by the topic resource and send them to the endpoint resource. [Learn more](#)'. The 'EVENT SUBSCRIPTION DETAILS' section contains fields for 'Name' (StorageEventSubscription) and 'Event Schema' (Event Grid Schema). The 'TOPIC DETAILS' section contains fields for 'Topic Types' (Storage Accounts), 'Subscription' (RiteshSubscription), and 'Resource Group' (azureclitest). Below these, there's a list of 'EVENT TYPES' with options for 'someoddstoreacc1', 'storageforeventgrid', and 'someoddstoreacc'. A checkbox at the bottom left is checked, labeled 'Subscribe to all event types'.

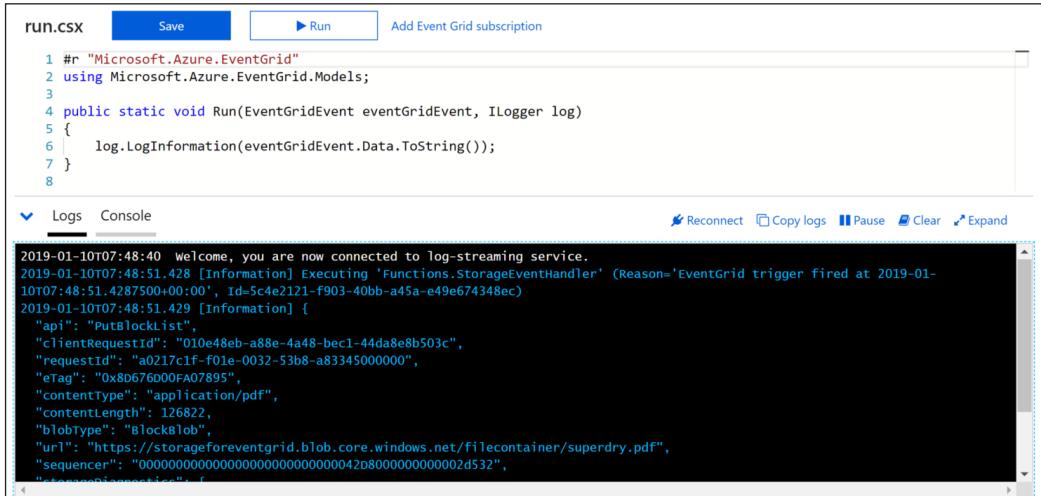
Soluções de integração do Azure

Verifique se a caixa de seleção Assinar todos os tipos de evento está marcada e clique no botão Criar (ele deverá ser habilitado assim que uma conta de armazenamento for selecionada).

6. Se agora navegamos para a conta de armazenamento no portal do Azure e clicamos no link eventos no menu à esquerda, a assinatura da conta de armazenamento deverá estar visível:



- Faça upload de um arquivo para o armazenamento de Blobs depois de criar um contêiner, e a função do Azure deverá ser executada. A ação de upload acionará um novo evento do tipo blobAdded e o enviará para o tópico da Grade de Eventos para contas de armazenamento. Conforme mostrado na captura de tela a seguir, a assinatura já está definida para obter todos os eventos deste tópico e a função é executada como parte do manipulador de eventos:



```

run.csx
Save ▶ Run Add Event Grid subscription

1 #r "Microsoft.Azure.EventGrid"
2 using Microsoft.Azure.EventGrid.Models;
3
4 public static void Run(EventGridEvent eventGridEvent, ILogger log)
5 {
6     log.LogInformation(eventGridEvent.Data.ToString());
7 }
8

Logs Console
Reconnect Copy logs Pause Clear Expand

2019-01-10T07:48:40 Welcome, you are now connected to log-streaming service.
2019-01-10T07:48:51.428 [Information] Executing 'Functions.StorageEventHandler' (Reason='EventGrid trigger fired at 2019-01-10T07:48:51.428Z500+00:00', id=5c4e2121-f903-40bb-a45a-e49e674348ec)
2019-01-10T07:48:51.429 [Information] {
    "api": "PutBlockList",
    "clientRequestId": "010e48eb-a88e-4a48-becl-44da8e8b503c",
    "requestId": "a0217c1f-f01e-0032-53b8-a83345000000",
    "eTag": "0xb6e76d000fa07895",
    "contentType": "application/pdf",
    "contentLength": 126822,
    "blobType": "BlockBlob",
    "url": "https://storageforeventgrid.blob.core.windows.net/filecontainer/superdry.pdf",
    "sequencer": "00000000000000000000000000000042d80000000000002d532",
    "storageDiagnostic": "f"
}

```

Eventos personalizados

Neste exemplo, em vez de usar recursos prontos para uso para gerar eventos, serão usados eventos personalizados. Usaremos o PowerShell para criar essa solução e reutilizar a mesma função do Azure que foi criada no último exercício como o manipulador:

- Faça logon e conecte-se à assinatura do Azure à sua escolha usando o cmdlet `Login-AzureRMAccount` e `Set-AzureRmContext`.
- A próxima etapa é criar um novo tópico de Grade de Eventos no Azure em um grupo de recursos. O cmdlet `New-AzureRmEventGridTopic` é usado para criar um novo tópico:

```
New-AzureRmEventGridTopic -ResourceGroupName CustomEventGridDemo
-Name "KeyVaultAssetsExpiry" -Location "West Europe"
```

3. Depois que o tópico for criado, sua URL de ponto de extremidade e a chave deverão ser recuperadas, já que são necessárias para enviar e publicar o evento para ele. Os cmdlets `Get-AzureRmEventGridTopic` e `Get-AzureRmEventGridTopicKey` são usados para recuperar esses valores. Observe que `Key1` é recuperada para se conectar ao ponto de extremidade:

```
$topicEndpoint = (Get-AzureRmEventGridTopic -ResourceGroupName containers -Name KeyVaultAssetsExpiry).Endpoint
```

```
$keys = (Get-AzureRmEventGridTopicKey -ResourceGroupName containers -Name KeyVaultAssetsExpiry).Key1
```

4. Uma nova tabela de hash é criada com todas as cinco propriedades de evento da Grade de Eventos importantes. Uma nova propriedade `guid` é gerada para a ID, a propriedade `subject` é definida como `Expiração de Evento de cofre de chaves`, `eventType` é definido como `Expiração de Certificado`, `eventTime` é definido com a hora atual e data contém informações sobre o certificado:

```
$eventgridDataMessage = @{  
    id = [System.guid]::NewGuid()  
    subject = "Key Vault Asset Expiry"  
    eventType = "Certificate Expiry"  
    eventTime = [System.DateTime]::UtcNow  
    data = @{  
        CertificateThumbprint = "sdfervdserwetsgfhgdg"  
        ExpiryDate = "1/1/2019"  
        Createdon = "1/1/2018"  
    }  
}
```

5. Como os dados da Grade de Eventos devem ser publicados na forma de uma matriz JSON, a carga é convertida na matriz JSON. Os colchetes `"[", "`"]" representam uma matriz JSON:

```
$finalBody = "[" + $(ConvertTo-Json $eventgridDataMessage) + "]"
```

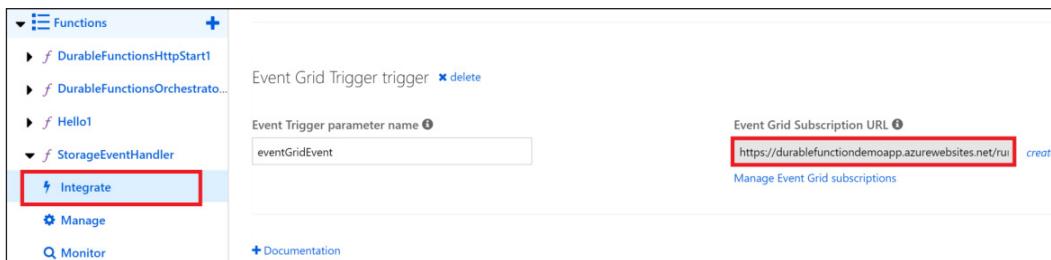
6. O evento será publicado usando o protocolo HTTP e as informações de cabeçalho apropriadas devem ser adicionadas à solicitação. A solicitação é enviada usando o tipo de conteúdo `Application/JSON` e a chave pertencente ao tópico é atribuída ao cabeçalho `aeg-sas-key`. É obrigatório nomear o cabeçalho e o conjunto de chaves como `aeg-sas-key`:

```
$header = @{  
    "contentType" = "application/json"  
    "aeg-sas-key" = $keys}
```

- Uma nova assinatura é criada para o tópico personalizado com um nome, o grupo de recursos que contém o tópico, o nome do tópico, o ponto de extremidade do webhook e o ponto de extremidade real que atua como o manipulador de eventos. Nesse caso, o manipulador de eventos é a função do Azure:

```
New-AzureRmEventGridSubscription -TopicName KeyVaultAssetsExpiry
-EventSubscriptionName "customtopicsubscriptionautocar"
-ResourceGroupName CustomEventGridDemo -EndpointType webhook
-Endpoint "https://durablefunctiondemoapp.azurewebsites.net/runtime/webhooks/
EventGrid?functionName=StorageEventHandler&code=0aSw6sxvtFmafXHvt7iOw/
Dsb8o1M9RKKagzVchTUkwe9Elkzl4mCg=="
-Verbose
```

A URL da função do Azure está disponível na guia Integrar, conforme mostrado na captura de tela a seguir:



- Até agora, o assinante (manipulador de eventos) e o editor foram configurados. A próxima etapa é enviar e publicar um evento para o tópico personalizado. Os dados do evento já foram criados na etapa anterior e, usando o cmdlet `Invoke-WebRequest`, a solicitação é enviada ao ponto de extremidade junto com o corpo e o cabeçalho:

```
Invoke-WebRequest -Uri $topicEndpoint -Body $finalBody -Headers
$header -Method Post
```

Aplicativos Lógicos do Azure

Aplicativos Lógicos é a oferta de fluxo de trabalho sem servidor do Azure. Ele tem todos os recursos de tecnologias sem servidor, como custeio baseado no consumo e escalabilidade ilimitada. O Aplicativos Lógicos nos ajuda a criar uma solução de fluxo de trabalho com facilidade usando o portal do Azure. Ele fornece uma interface do usuário de arrastar e soltar para criar e configurar o fluxo de trabalho.

O uso de Aplicativos Lógicos é a maneira preferencial de integrar serviços e dados, criar projetos de negócios e criar um fluxo completo de lógica. Há uma série de conceitos importantes que devem ser entendidos antes de criar um aplicativo lógico.

Atividade

Atividade refere-se à execução de uma única unidade de trabalho. Exemplos de atividades incluem a conversão de XML em JSON, a leitura de blobs do armazenamento do Azure e a gravação em uma coleção de documentos do Cosmos DB. O Aplicativos Lógicos é um fluxo de trabalho que consiste em várias atividades relacionadas em uma sequência. Há dois tipos de atividade no Aplicativos Lógicos, da seguinte maneira:

- **Gatilho:** os gatilhos referem-se à *iniciação de uma atividade*. Todos os aplicativos lógicos têm um único gatilho que forma a primeira atividade. É o gatilho que cria uma instância do aplicativo lógico e inicia a execução. Exemplos de gatilhos são a chegada de mensagens da Grade de Eventos, a chegada de um email, uma solicitação HTTP e um agendamento.
- **Ações:** qualquer atividade que não seja um gatilho é uma atividade de etapa e cada uma delas executa uma responsabilidade. As etapas estão conectadas entre si em um fluxo de trabalho.

Conectores

Os **conectores** são recursos do Azure que ajudam a conectar um aplicativo lógico a serviços externos. Esses serviços podem estar na nuvem ou na infraestrutura local. Por exemplo, há um conector para conectar Aplicativos Lógicos à Grade de Eventos. Da mesma forma, há outro conector para se conectar ao Office 365 Exchange. Quase todos os tipos de conectores estão disponíveis em Aplicativos Lógicos, e eles podem ser usados para se conectar a serviços. Os conectores contêm informações de conexão e também lógica para conectar a serviços externos usando essas informações de conexão.



Toda a lista de conectores está disponível em <https://docs.microsoft.com/connectors/>.



Como trabalhar em um aplicativo lógico

Vamos criar um fluxo de trabalho de Aplicativos Lógicos que seja acionado quando uma das contas de email recebe um email. Ele responde ao remetente com um email padrão e executa a análise de sentimento sobre o conteúdo do email. Para análise de sentimento, o recurso de Análise de Texto de Serviços Cognitivos deve ser provisionado antes da criação do aplicativo lógico:

1. Navegue até o portal do Azure, faça logon e crie um recurso de **Análise de Texto** em um grupo de recursos, conforme mostrado na captura de tela a seguir:

The screenshot shows the Microsoft Azure Marketplace interface. On the left, there's a search bar with 'Text Analytics' typed in. Below it are filters for 'Pricing' (set to 'All'), 'Operating System' (set to 'All'), and 'Publisher' (set to 'All'). The results section shows a single item: 'Text Analytics' by Microsoft, which falls under the 'AI + Machine Learning' category. To the right of the main content is a sidebar titled 'Text Analytics' under the 'Microsoft' publisher. The sidebar includes sections for 'Sentiment analysis' (described as finding user opinions), 'Key phrase extraction' (automatically extracting key phrases), and 'Language detection' (determining language). There are also 'Save for later' and 'Create' buttons.

2. Forneça um **Nome**, **Local** e nome de **Assinatura**, um nome de **Grupo de recursos** e **Tipo de preço**, da seguinte maneira:

The screenshot shows the 'Create' dialog box for 'Text Analytics'. It contains the following fields:

- Name:** emailanalysis
- Subscription:** RiteshSubscription
- Location:** West Europe
- Pricing tier:** F0 (5K Transactions per 30 days)
- Resource group:** EmailAnalysis

At the bottom of the dialog, there is a 'Create new' link.

Soluções de integração do Azure

- Depois que o recurso for provisionado, navegue até a página **Visão geral** e copie a URL do ponto de extremidade. Guarde-o em um local temporário. Esse valor será necessário na configuração do aplicativo lógico:

The screenshot shows the 'Overview' tab of a Resource Group named 'EmailAnalysis'. Key details include:

- Resource group: EmailAnalysis
- Status: Active
- Location: West Europe
- Subscription ID: 9755fce-e94b-4332-9be8-1ade15e78909
- Tags: Click here to add tags
- API type: Text Analytics
- Pricing tier: Free
- Endpoint: https://westeurope.api.cognitive.microsoft.com/text/analytics/v2.0 (highlighted with a red box)
- Manage keys
- Show access keys ...

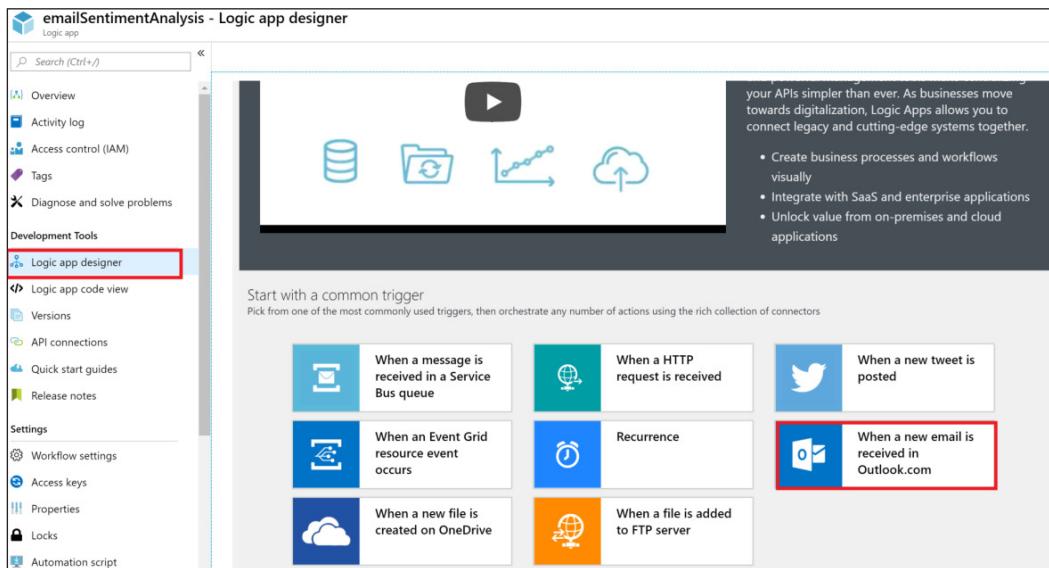
- Navegue até a página **Chaves** e copie o valor de **Chave 1** e armazene-o em um local temporário. Esse valor será necessário na configuração do aplicativo lógico.
- A próxima etapa será criar um aplicativo lógico; para criar um aplicativo lógico, navegue até o grupo de recursos no portal do Azure no qual o aplicativo lógico deve ser criado. Procure o **Aplicativo Lógico** e crie-o fornecendo um **Nome**, **Local**, nome de **Grupo de recursos** e nome de **Assinatura**:

The 'Logic App' creation dialog is shown with the following configuration:

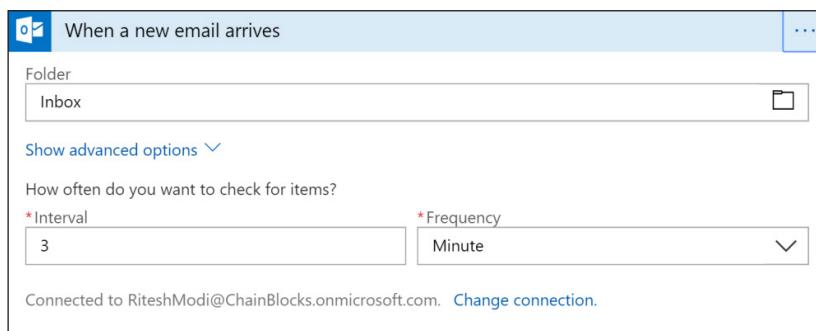
- Name: emailSentimentAnalysis
- Subscription: RiteshSubscription
- Resource group: EmailAnalysis (radio button selected for 'Create new')
- Location: West Europe
- Log Analytics: On

Note: You can add triggers and actions to your Logic App after creation.

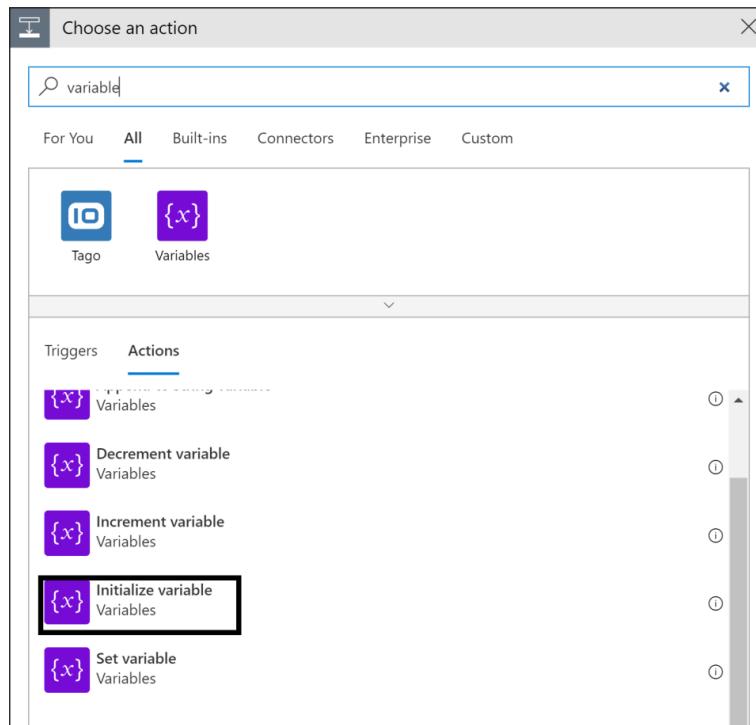
- Depois que o aplicativo lógico tiver sido criado, navegue até o recurso, clique em Designer de aplicativo lógico no menu à esquerda e, em seguida, selecione o modelo **Quando um novo email é recebido em Outlook.com** para criar um novo fluxo de trabalho. O modelo fornece um jumpstart, adicionando cópias de gatilhos e atividades. Isso adicionará um gatilho do Office 365 Outlook automaticamente ao fluxo de trabalho, conforme mostrado na captura de tela a seguir:



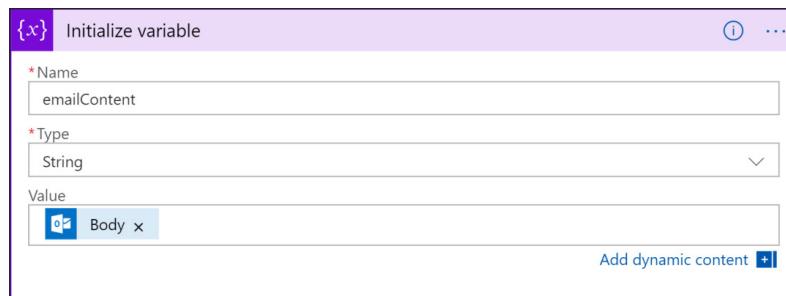
- Clique no botão Entrar no gatilho; ele abrirá uma nova janela do Internet Explorer. Em seguida, entre em sua conta. Após a conexão bem-sucedida, um novo conector de email do Office 365 será criado com as informações de conexão para a conta.
- Clique no botão **Continuar** e configure o gatilho com uma frequência de sondagem de 3 minutos, conforme mostrado na captura de tela a seguir:



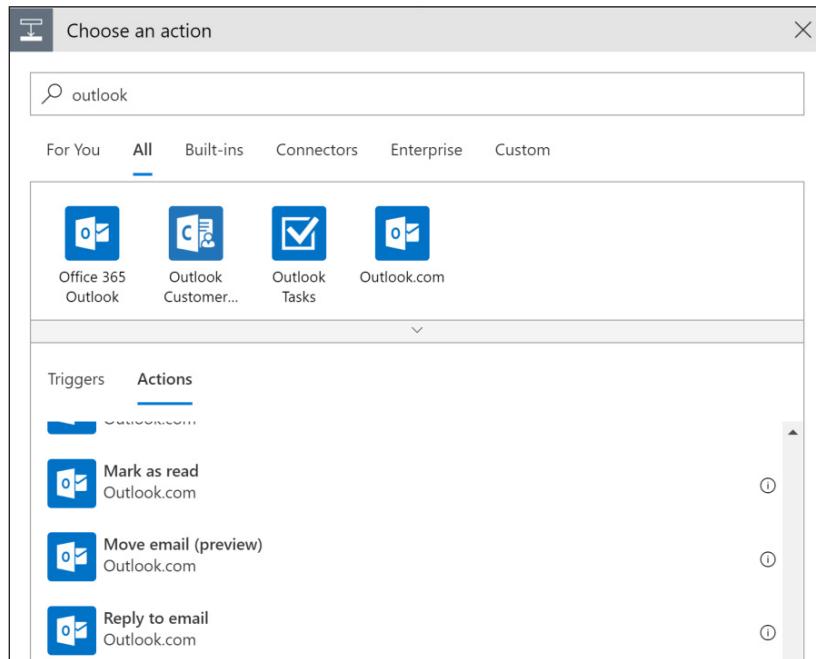
9. Clique na próxima etapa para adicionar outra ação e digite variáveis. Em seguida, selecione a ação Inicializar variável, conforme demonstrado na captura de tela a seguir:



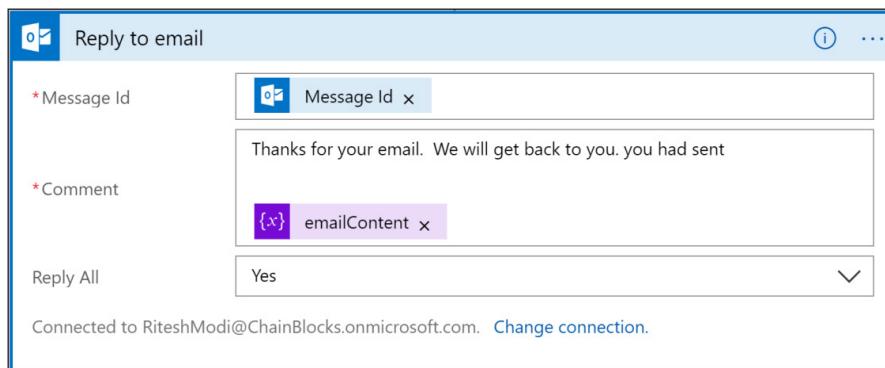
10. Depois, configure a ação de variável. Quando a caixa de texto **Valor** for clicada, aparecerá uma janela pop-up mostrando Conteúdo dinâmico e Expressão. O Conteúdo dinâmico refere-se a propriedades que estão disponíveis para a ação atual, que são preenchidas com valores de tempo de execução de ações anteriores e gatilhos. As variáveis ajudam a manter os fluxos de trabalho genéricos. Nessa janela, selecione **Corpo** em Conteúdo dinâmico:



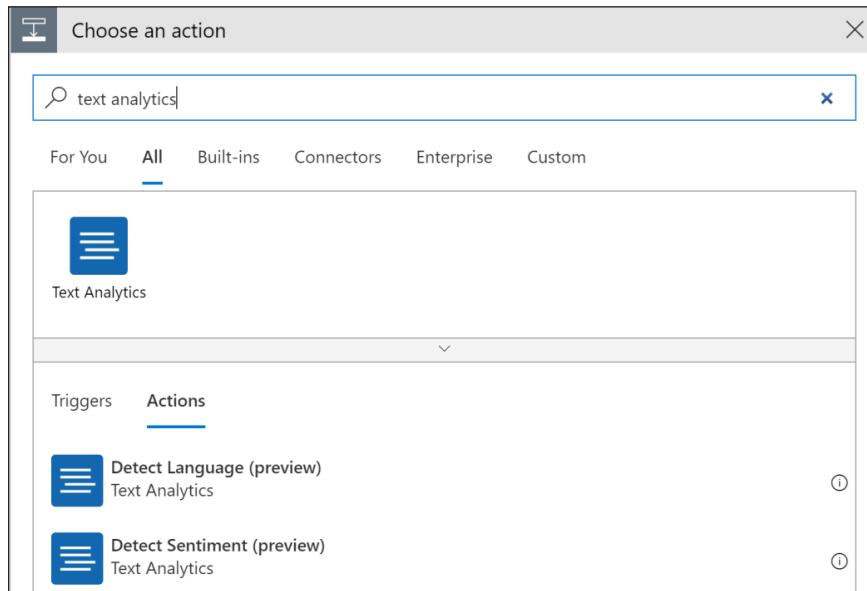
11. Adicione outra ação clicando na etapa **Adicionar**, digitando outlook e, em seguida, selecionando a ação **Responder a email**:



12. Configure a nova ação; verifique se **Id da Mensagem** está definida com o conteúdo dinâmico, **Id da Mensagem** e, em seguida, digite a resposta que você gostaria de enviar na caixa **Comentário**:



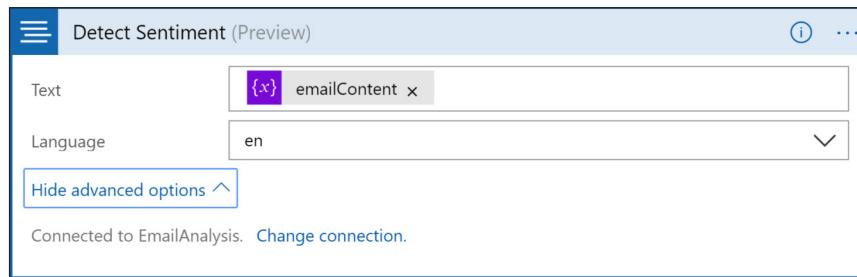
13. Adicione outra ação, digite análise de texto e, em seguida, selecione **Detectar Sentimento (visualização)**:



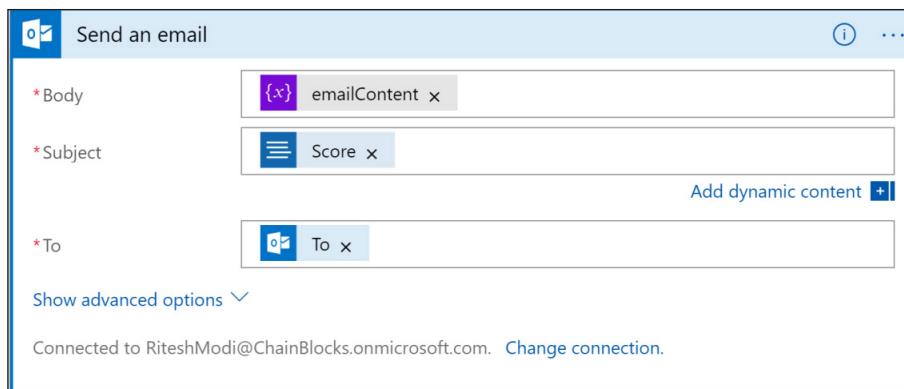
14. Configure a ação de sentimento, conforme mostrado na captura de tela a seguir – os valores de ponto de extremidade e chave devem ser usados aqui. Agora clique no botão **Criar**, como demonstrado na seguinte captura de tela:

* Connection Name	EmailAnalysis
* Account Key
Site URL	https://westeurope.api.cognitive.microsoft.com/text/analytics/v2.0

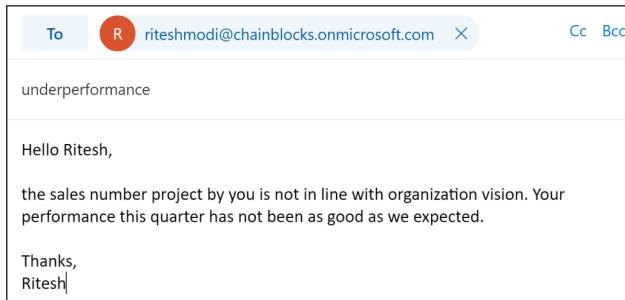
15. Forneça o texto à ação adicionando conteúdo dinâmico e selecionando a variável criada anteriormente, emailContent. Em seguida, clique em **Mostrar opções avançadas** e selecione **pt** para **Idioma**:



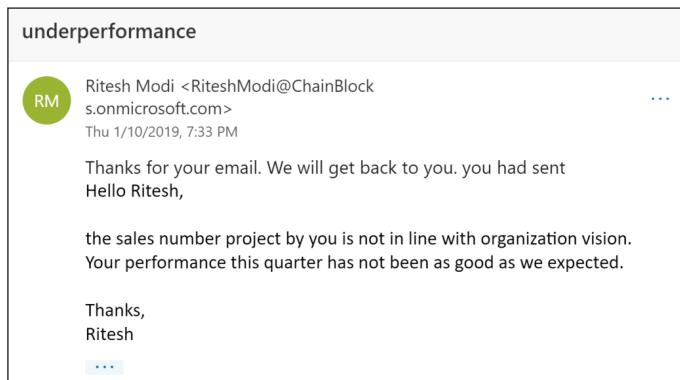
16. Em seguida, adicione uma nova ação selecionando **Outlook** e, em seguida, selecione **Enviar um email**. Essa ação envia ao destinatário original o conteúdo do email com a pontuação de sentimento em seu assunto. Ele deve ser configurado como mostrado na captura de tela a seguir. Se a pontuação não estiver visível na janela de conteúdo dinâmico, clique no link **Ver mais** ao lado dele:



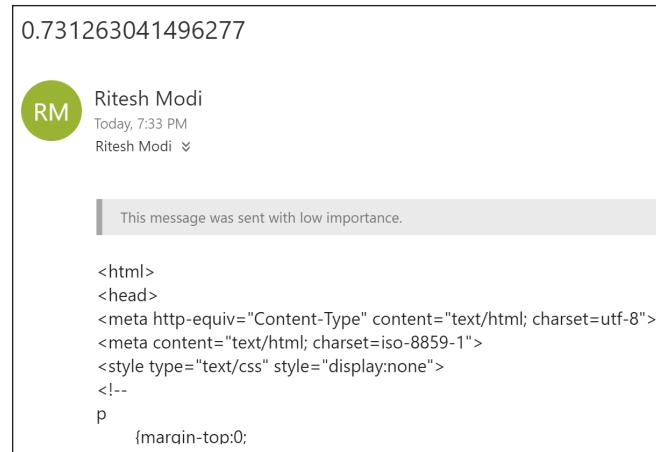
17. Salve o aplicativo lógico, navegue de volta para a página de visão geral e clique no gatilho **Executar**. O gatilho verificará novos emails a cada três minutos, responderá aos remetentes, executará a análise de sentimento e enviará um email para o destinatário original. Um email de exemplo com conotações negativas é enviado para a ID de email fornecida:



18. Depois de alguns segundos, o aplicativo lógico é executado e o remetente recebe a seguinte resposta:



19. O destinatário original recebe um email com a pontuação de sentimento e o texto de email original, conforme mostrado na captura de tela a seguir:



Criação de uma solução de ponta a ponta usando tecnologias sem servidor

Nesta seção, criaremos uma solução de ponta a ponta compreendendo tecnologias sem servidor que discutimos nas seções anteriores.

A declaração do problema

O problema que vamos resolver aqui é que os usuários e organizações não são notificados sobre a expiração de qualquer segredo em seu cofre de chaves, e os aplicativos param de funcionar quando expiram. Os usuários estão reclamando que o Azure não fornece a infraestrutura para monitorar segredos, chaves e certificados do cofre de chaves.

Visão

O Azure Key Vault é um serviço do Azure que fornece armazenamento seguro e acesso a credenciais, chaves, segredos e certificados. Ele fornece um cofre que tem o backup feito por dispositivos de hardware conhecidos como **Módulos de Segurança de Hardware (HSM)**, e que compõe a mais alta forma de armazenamento seguro para segredos e chaves.

O Azure Key Vault permite o armazenamento dessas chaves, segredos e credenciais, e eles têm uma data de expiração. Por exemplo, um certificado carregado para o Azure Key Vault expirará em um ano, ou um segredo expirará em dois anos. Embora seja uma das práticas recomendadas de segurança do Azure para armazenar informações confidenciais no Key Vault, o Key Vault não fornece nenhuma infraestrutura para monitorar esses segredos e notificar os usuários de que esses segredos estão expirando antecipadamente. Nessas situações, se o usuário não estiver monitorando seus próprios segredos do Key Vault ativamente, os aplicativos que dependem desses segredos (como cadeias de conexão e nomes de usuário) pararão de funcionar, e as medidas reativas precisarão ser empreendidas para corrigir o aplicativo por renovar o segredo no Key Vault.

A visão é criar uma solução completa de ponta a ponta usando os serviços do Azure para que os usuários sejam notificados com bastante antecedência de que os segredos do Key Vault expirarão em breve e que precisarão agir para renová-los.

Solução

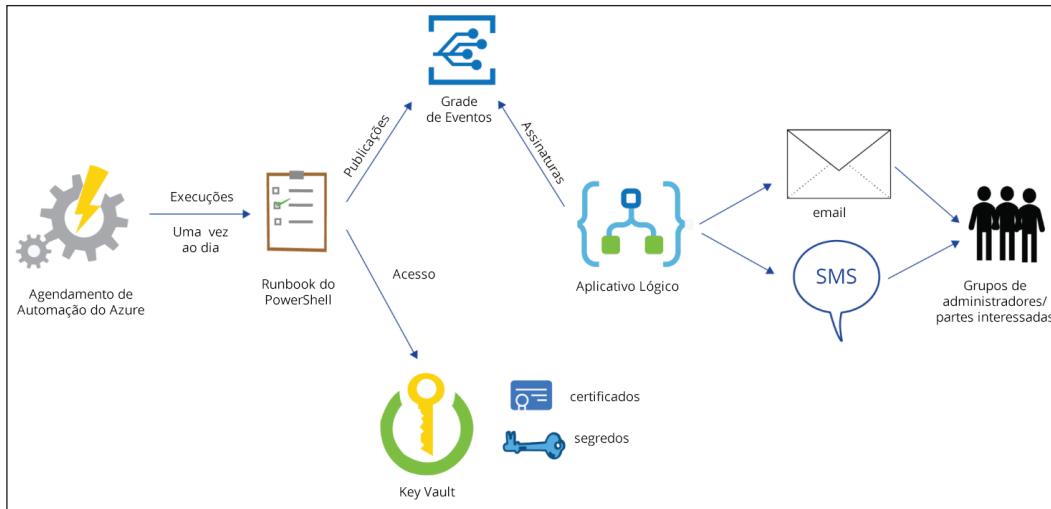
A solução para esse problema é combinar vários serviços do Azure e integrá-los para que os usuários possam ser notificados proativamente sobre a expiração de segredos. A solução enviará notificações usando dois canais – email e SMS.

Os serviços do Azure usados para criar essa solução incluem o seguinte:

- Azure Key Vault
- Azure AD
- Grade de Eventos do Azure
- Automação do Azure
- Aplicativos Lógicos
- Azure Functions
- SendGrid
- SMS do Twilio

Arquitetura

A arquitetura da solução compreende vários serviços, conforme mostrado no diagrama a seguir:



Vamos passar por cada um desses serviços e entender seus papéis e a funcionalidade que eles fornecem na solução geral.

Automação do Azure

A Automação do Azure fornece runbooks e estes runbooks podem ser usados para executar a lógica utilizando o PowerShell, Python e outras linguagens de script. O runbook pode ser executado localmente ou na nuvem e fornece infraestrutura e instalações ricas para criar scripts. Esses scripts são conhecidos como **runbooks**. Normalmente, os runbooks implementam um cenário como interromper ou iniciar uma máquina virtual, ou criar e configurar contas de armazenamento. É muito fácil conectar-se ao ambiente do Azure desde runbooks com a ajuda de ativos, como variáveis, certificados e conexões.

Na solução atual, queremos conectar-se ao Azure Key Vault, ler todos os segredos e chaves armazenados nele e buscar suas datas de expiração. Essas datas de expiração devem ser comparadas com a data de hoje e, se a data de expiração estiver dentro de um mês, o runbook deve disparar um evento personalizado na Grade de Eventos usando um tópico personalizado de Grade de Eventos.

Um runbook da Automação do Azure utilizando um script do PowerShell será implementado para conseguir isso. Junto com o runbook, um agendador também será criado que executará o runbook uma vez por dia à meia-noite.

Um tópico personalizado de Grade de Eventos do Azure

Uma vez que o runbook identifique que um segredo ou chave vai expirar dentro de um mês, ele gerará um novo evento personalizado e o publicará no tópico personalizado criado especificamente para essa finalidade. Novamente, veremos os detalhes da implementação na próxima seção.

Aplicativos Lógicos do Azure

Um aplicativo lógico é um serviço sem servidor que fornece recursos de fluxo de trabalho. Nosso aplicativo lógico será configurado para ser disparado quando um evento for publicado no tópico de Grade de Eventos personalizado. Depois de disparado, ele invocará o fluxo de trabalho e executará todas as atividades contidas nele uma após a outra. Geralmente, há várias atividades, mas para a finalidade deste exemplo, invocaremos uma função do Azure que enviará mensagens de email e SMS. Em uma implementação completa, essas funções de notificação devem ser implementadas separadamente em funções separadas do Azure.

Azure Functions

O Azure Functions é usado para notificar os usuários e as partes interessadas sobre a expiração de segredos e chaves usando email e SMS. SendGrid é usado para enviar emails, enquanto o Twilio é usado para enviar mensagens SMS do Azure Functions.

Pré-requisitos

No mínimo, você precisará de uma assinatura do Azure com direitos de colaborador.

Implementação

Um Azure Key Vault já deve existir. Caso contrário, deverá ser criado.

Esta etapa deve ser executada se um novo Azure Key Vault precisar ser provisionado. O Azure fornece várias maneiras de provisionar recursos no Azure; proeminente entre eles é o Azure PowerShell e a CLI do Azure. A CLI do Azure é uma interface de linha de comando que funciona em várias plataformas. A primeira tarefa será provisionar um cofre de chaves no Azure. O Azure PowerShell é usado para provisionar o Azure Key Vault.

Antes que o Azure PowerShell possa ser usado para criar um cofre de chaves, é importante fazer logon no Azure para que os comandos subsequentes possam ser executados com êxito e criar um cofre de chaves.

Etapa 1

A primeira etapa é preparar o ambiente para a amostra. Isso envolve o logon no portal do Azure, selecionando uma assinatura apropriada e, em seguida, criando um novo grupo de recursos do Azure e um novo recurso do Azure Key Vault:

1. Execute o comando `Login-AzureRmAccount` para fazer logon no Azure. Ele solicitará credenciais em uma nova janela.
2. Após um logon bem-sucedido, se houver várias assinaturas disponíveis para a ID de logon fornecida, elas serão todas listadas. É importante selecionar uma assinatura apropriada; isso pode ser feito executando o cmdlet `Set-AzureRmContext`:

```
Set-AzureRmContext -Subscription xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxx
```

3. Crie um novo grupo de recursos no seu local preferencial. Nesse caso, o nome do grupo de recursos é `IntegrationDemo` e é criado na região Europa Ocidental:

```
New-AzureRmResourceGroup -Name IntegrationDemo -Location "West Europe" -Verbose
```

4. Crie um novo Azure Key Vault; o nome do cofre, nesse caso, é `keyvaultbook`, e ele está habilitado para implantação, implantação de modelo, criptografia de disco, exclusão suave e proteção contra remoção:

```
New-AzureRmKeyVault -Name keyvaultbook -ResourceGroupName IntegrationDemo -Location "West Europe" -EnabledForDeployment -EnabledForTemplateDeployment -EnabledForDiskEncryption -EnableSoftDelete -EnablePurgeProtection -Sku Standard -Verbose
```

O comando anterior, quando executado com êxito, criará um novo Azure Key Vault. A próxima etapa é fornecer acesso a uma entidade de serviço no Key Vault.

Etapa 2

Em vez de usar uma conta individual para se conectar ao Azure, o Azure fornece entidades de serviço, que são, em essência, contas de serviço que podem ser usadas para se conectar ao Azure Resource Manager e executar atividades. Adicionar um usuário ao locatário do Azure os torna disponíveis em todos os lugares, inclusive em todos os grupos de recursos e recursos, devido à natureza da herança de segurança no Azure. O acesso deve ser explicitamente revogado de grupos de recursos para usuários se eles não tiverem permissão para acessá-lo. As entidades de serviço ajudam a atribuir acesso e controle granular a grupos de recursos, recursos e, se necessário, podem receber acesso ao escopo da assinatura. Eles também podem ser atribuídos permissões granulares, como leitor, colaborador ou permissões de proprietário.

Em suma, as entidades de serviço devem ser o mecanismo preferencial para consumir serviços do Azure. Eles podem ser configurados com uma senha ou com uma chave de certificado. O processo de provisionamento de uma entidade de serviço com uma senha usando o PowerShell é o seguinte:

1. Crie um aplicativo de serviço usando `New-AzurermAApplication` passando valores para o nome de exibição, URIs identificados, home page e senha. A senha deve ser uma cadeia de caracteres segura.

2. Uma cadeia de caracteres segura pode ser gerada usando o cmdlet `ConvertTo-SecureString`, conforme mostrado no código a seguir:

```
ConvertTo-SecureString -String sysadminpassword -AsPlainText  
-Force
```

3. O aplicativo de serviço pode ser criado em seguida. O valor de retorno do comando é armazenado em uma variável para que ele possa ser usado no seguinte comando:

```
$app = New-AzureRmADApplication -DisplayName "https://keyvault.  
book.com" -IdentifierUris "https://keyvault.book.com" -HomePage  
"https://keyvault.book.com" -Password (ConvertTo-SecureString  
-String sysadminpassword -AsPlainText -Force) -Verbose
```

4. Depois que o aplicativo de serviço tiver sido criado, uma entidade de serviço com base no aplicativo de serviço deve ser criada. O aplicativo atua como um blueprint, e o principal atua como uma instância do aplicativo. Podemos criar vários principais usando o mesmo aplicativo em diferentes locatários. O comando para criar uma entidade de serviço é o seguinte:

```
New-AzureRmADServicePrincipal -ApplicationId $app.ApplicationId  
-DisplayName "https://keyvault.book.com" -Password (ConvertTo-  
SecureString -String sysadminpassword -AsPlainText -Force) -Scope  
"/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx" -Role  
Owner -StartDate ([datetime]::Now) -EndDate $([datetime]::now.  
AddYears(1)) -Verbose
```

5. A `ApplicationId` do comando anterior é usada neste comando por meio da variável `$app`. `StartDate` e `EndDate` são adicionados à entidade de serviço.

6. Os valores de configuração importantes são o escopo e a função. O escopo determina a área de acesso para o aplicativo de serviço; atualmente é mostrado no nível da assinatura. Os valores válidos para o escopo são os seguintes:

```
/subscriptions/{subscriptionId}  
/subscriptions/{subscriptionId}/resourceGroups/{resourceGroupName}  
/subscriptions/{subscriptionId}/resourcegroups/  
{resourceGroupName}/providers/{resourceProviderNamespace}/  
{resourceType}/{resourceName}  
/subscriptions/{subscriptionId}/resourcegroups/  
{resourceGroupName}/providers/{resourceProviderNamespace}/  
{parentResourcePath}/{resourceType}/{resourceName}
```

A função fornece permissões no escopo atribuído. Os valores válidos são os seguintes:

- Proprietário
- Colaborador
- Leitor
- Permissões específicas de recursos

No comando anterior, as permissões de proprietário foram fornecidas para a entidade de serviço recém-criada.

Etapa 3

Para criar uma entidade de serviço usando certificados, as seguintes etapas devem ser executadas:

1. Criar um certificado autoassinado ou comprar um certificado: um certificado autoassinado é usado para criar este exemplo de aplicativo de ponta a ponta. Para implantações da vida real, um certificado válido deve ser comprado de uma autoridade de certificação.

Para criar um certificado autoassinado, o comando a seguir pode ser executado. O certificado autoassinado é exportável e armazenado em uma pasta pessoal no computador local; ele também tem uma data de expiração:

```
$currentDate = Get-Date  
$expiryDate = $currentDate.AddYears(1)  
$finalDate = $expiryDate.AddYears(1)  
$servicePrincipalName = "https://automation.book.com"  
$automationCertificate = New-SelfSignedCertificate -DnsName  
$servicePrincipalName -KeyExportPolicy Exportable -Provider  
"Microsoft Enhanced RSA and AES Cryptographic Provider" -NotAfter  
$finalDate -CertStoreLocation "Cert:\LocalMachine\My"
```

2. **Exportar o certificado recém-criado:** o novo certificado deve ser exportado para o sistema de arquivos para que, posteriormente, ele possa ser carregado para outros destinos, como o Azure AD, para criar uma entidade de serviço.

Os comandos usados para exportar o certificado para o sistema de arquivos local são mostrados a seguir. Note que este certificado tem chaves públicas e privadas e, por isso, enquanto é exportado, tem de ser protegido utilizando uma senha e a senha tem de ser uma cadeia de caracteres segura:

```
$securepfxpwd = ConvertTo-SecureString -String 'password'  
-AsPlainText -Force # Password for the private key PFX certificate  
$cert1 = Get-Item -Path Cert:\LocalMachine\  
My\$($automationCertificate.Thumbprint)  
Export-PfxCertificate -Password $securepfxpwd -FilePath "C:\book\  
azureautomation.pfx" -Cert $cert1
```

O cmdlet Get-Item lê o certificado do repositório de certificados e o armazena na variável \$cert1. O cmdlet Export-PfxCertificate realmente exporta o certificado no repositório de certificados para o sistema de arquivos. Nesse caso, ele está na pasta c:\book.

3. **Ler o conteúdo do arquivo PFX recém-gerado:** um objeto X509Certificate é criado para armazenar o certificado em memória, e os dados são convertidos em uma cadeia de caracteres de Base64 usando a função System.Convert:

```
$newCert = New-Object System.Security.Cryptography.  
X509Certificates.X509Certificate -ArgumentList "C:\book\  
azureautomation.pfx", $securepfxpwd  
$newcertdata = [System.Convert]::ToBase64String($newCert.  
GetRawCertData())
```

Crie uma instância da classe PSADKeyCredential e forneça valores para suas propriedades importantes. Ela é usada durante a criação de uma nova entidade de serviço. Essa classe está disponível no namespace Microsoft.Azure.Graph.RBAC.Version1_6.ActiveDirectory. Um novo valor Guid também é gerado para a propriedade keyid de PSADKeyCredential. O valor de certificado codificado em Base64 é atribuído à propriedade CertValue, e os valores para StartDate e EndDate também são fornecidos. Quando uma entidade de serviço for criada usando este objeto, ela será configurada de acordo com os valores fornecidos aqui:

```
$keyid = [Guid]::NewGuid()  
$keyCredential = New-Object -TypeName Microsoft.Azure.Graph.RBAC.  
Version1_6.ActiveDirectory.PSADKeyCredential  
$keyCredential.StartDate = [datetime]::Now  
$keyCredential.KeyId = $keyid  
$keyCredential.CertValue = $newcertdata  
$keyCredential.EndDate = [datetime]::Now.AddYears(1)
```

4. **Criar o aplicativo de serviço e a entidade de serviço no Azure:** nós já executamos esses comandos uma vez na criação da entidade de serviço com uma senha. Dessa vez, a diferença principal é que, em vez de uma senha, a propriedade `keyCredential` é usada. Por fim, uma entidade de serviço é criada com direitos de proprietário.

Nós usaremos essa mesma entidade na conexão ao Azure da conta da Automação do Azure. É importante que a ID do aplicativo, a ID do locatário, a ID da assinatura e os valores de impressão digital do certificado sejam armazenados em um local temporário para que possam ser usados para configurar recursos subsequentes:

```
$adAppName = http://automationcertcred2
$adApp =New-AzureRmADApplication -DisplayName $adAppName
-HomePage $adAppName -IdentifierUris $adAppName -KeyCredentials
$keyCredential -Verbose
New-AzureRmADServicePrincipal -ApplicationId $adApp.ApplicationId.
Guid -Role owner
```

Etapa 4

Nesta fase, criamos a entidade de serviço e o cofre de chaves. No entanto, a entidade de serviço ainda não tem acesso ao cofre de chaves. Essa entidade de serviço será usada para consultar e listar todos os segredos, chaves e certificados do cofre de chaves, e ele deve ter as permissões necessárias para fazer isso.

Para fornecer a permissão de entidade de serviço recém-criada para acessar o cofre de chaves, voltaremos ao console do Azure PowerShell e executaremos o seguinte comando:

```
Set-AzureRmKeyVaultAccessPolicy -VaultName keyvaultbook
-ResourceGroupName IntegrationDemo -ObjectId "ea36bc00-6eff-
4236-8c43-65c0c2e7e4cb" -PermissionsToKeys get,list,create
-PermissionsToCertificates get,list,import -PermissionsToSecrets
get,list -Verbose
```

Referindo-se ao bloco de comando anterior, dê uma olhada nos seguintes pontos:

- `Set-AzureRmKeyVaultAccessPolicy` fornece permissões de acesso a usuários, grupos e entidades de serviço. Ele aceita o nome do cofre de chaves e a ID do objeto de entidade de serviço. Esse objeto é diferente da ID do aplicativo. A saída de `New-AzureRmAdServicePrincipal` contém uma propriedade `Id`; enquanto o valor de `ObjectId` é este valor:

<code>ServicePrincipalNames</code>	: { "values": ["urn:oid:2e84c2f3-0000-0000-0000-000000000003"] }
<code>ApplicationId</code>	: f000-0000-0000-000000000000
<code>DisplayName</code>	: f000-0000-0000-000000000000 Id
<code>Id</code>	: 2e84c2f3-0000-0000-0000-000000000003
<code>ObjectId</code>	: f000-0000-0000-000000000000
<code>Type</code>	: servicePrincipal

- PermissionsToKeys fornece acesso a chaves no cofre de chaves e as permissões get, list e create são fornecidas a essa entidade de serviço. Não há nenhuma permissão de gravação ou atualização fornecida a essa entidade.
- PermissionsToKeys fornece acesso a segredos no cofre de chaves e as permissões get e list são fornecidas a essa entidade de serviço. Não há nenhuma permissão de gravação ou atualização fornecida a essa entidade.
- PermissionsToKeys fornece acesso a chaves no cofre de chaves e as permissões get, list e create são fornecidas a essa entidade de serviço. Não há nenhuma permissão de gravação ou atualização fornecida a essa entidade.

Etapa 5

Assim como antes, nós usaremos o Azure PowerShell para criar uma nova conta da Automação do Azure em um grupo de recursos. Antes de criar um grupo de recursos e uma conta de automação, uma conexão com o Azure deve ser estabelecida. No entanto, desta vez, utilize as credenciais para que Entidade de Serviço se conecte ao Azure. As etapas para conectar-se ao Azure usando Entidade de Serviço são mostradas a seguir:

1. O comando para se conectar ao Azure usando o aplicativo de serviço é o seguinte:

```
Login-AzureRmAccount -ServicePrincipal -CertificateThumbprint  
"003B0D26705C792DB60823DA5804A0897160C306" -ApplicationId  
"xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx" -Tenant "xxxxxxxx-xxxx-  
xxxx-xxxx-xxxxxxxxxxxx"
```

2. Aqui, applicationId está disponível após a execução do cmdlet New-AzureRmADApplication, e a ID do locatário e a ID da assinatura podem ser recuperadas usando o comando a seguir. A ID de assinatura será necessária em comandos subsequentes:

```
Get-AzureRmContext
```

3. Depois de se conectar ao Azure, um novo recurso com os recursos para a solução e uma nova conta da Automação do Azure devem ser criados. Estamos chamando o grupo de recursos como VaultMonitoring e criando-o na região Europa Ocidental. Nós criaremos o restante dos recursos neste grupo de recursos também:

```
$IntegrationResourceGroup = "VaultMonitoring"  
$rgLocation = "West Europe"  
$automationAccountName = "MonitoringKeyVault"
```

```
New-AzureRmResourceGroup -name $IntegrationResourceGroup -Location  
$rgLocation  
New-AzureRmAutomationAccount -Name $automationAccountName  
-ResourceGroupName $IntegrationResourceGroup -Location $rgLocation  
-Plan Free
```

4. Em seguida, crie três variáveis de automação. Os valores para eles, ou seja, a ID de subscrição, a ID de locatário e ID de aplicativo, já devem estar disponíveis utilizando as etapas anteriores:

```
New-AzureRmAutomationVariable -Name "azuresubscriptionid"  
-AutomationAccountName $automationAccountName -ResourceGroupName  
$IntegrationResourceGroup -Value "xxxxxxxx-xxxx-xxxx-xxxx-  
xxxxxxxxxxxx" -Encrypted $true
```

```
New-AzureRmAutomationVariable -Name "azuretenantid"  
-AutomationAccountName $automationAccountName -ResourceGroupName  
$IntegrationResourceGroup -Value "xxxxxxxx-xxxx-xxxx-xxxx-  
xxxxxxxxxxxx" -Encrypted $true  
New-AzureRmAutomationVariable -Name "azureappid"  
-AutomationAccountName $automationAccountName -ResourceGroupName  
$IntegrationResourceGroup -Value "xxxxxxxx-xxxx-xxxx-xxxx-  
xxxxxxxxxxxx" -Encrypted $true
```

5. Agora é hora de fazer upload de um certificado que será usado na conexão ao Azure desde a Automação do Azure:

```
$securepfxpwd = ConvertTo-SecureString -String 'password'  
-AsPlainText -Force # Password for the private key PFX certificate  
New-AzureRmAutomationCertificate -Name  
"RitestSubscriptionCertificate" -Path "C:\book\  
azureautomation.pfx" -Password $securepfxpwd  
-AutomationAccountName $automationAccountName -ResourceGroupName  
$IntegrationResourceGroup
```

6. A próxima etapa é instalar os módulos do PowerShell relacionados ao Key Vault e à Grade de Eventos na conta da Automação do Azure, pois esses módulos não são instalados por padrão.
7. No portal do Azure, navegue até o grupo de recursos **VaultMonitoring** já criado clicando no ícone **Grupos de Recursos** no menu à esquerda.

Soluções de integração do Azure

8. Clique na conta da Automação do Azure já provisionada, **MonitoringKeyVault** e então clique em **Módulos** no menu à esquerda, conforme mostrado na seguinte captura de tela:

NAME	LAST MODIFIED	STATUS	VERSION
AuditPolicyDsc	12/13/2018, 3:10 PM	Available	1.1.0.0
Azure	12/13/2018, 3:02 PM	Available	1.0.3
Azure.Storage	12/13/2018, 3:07 PM	Available	1.0.3
AzureRM.Automation	12/13/2018, 3:05 PM	Available	1.0.3
AzureRM.Compute	12/13/2018, 3:04 PM	Available	1.2.1
AzureRM.Profile	12/13/2018, 3:05 PM	Available	1.0.3
AzureRM.Resources	12/13/2018, 3:06 PM	Available	1.0.3
AzureRM.Sql	12/13/2018, 3:06 PM	Available	1.0.3
AzureRM.Storage	12/13/2018, 3:07 PM	Available	1.0.3
ComputerManagementDsc	12/13/2018, 3:08 PM	Available	5.0.0.0
GPRRegistryPolicyParser	12/13/2018, 3:10 PM	Available	0.2
Microsoft.PowerShell.Core	12/13/2018, 3:02 PM	Available	0.0
Microsoft.PowerShell.Diagnostics	12/13/2018, 3:01 PM	Available	
Microsoft.PowerShell.Management	12/13/2018, 3:01 PM	Available	

O módulo Grade de Eventos depende do módulo `AzureRM.profile` e, portanto, temos que instalá-lo antes do módulo Grade de Eventos.

9. Clique em **Procurar na Galeria** no menu superior e digite `Azurerm.profile` na caixa de pesquisa, como mostrado na seguinte captura de tela:

Home > VaultMonitoring > MonitoringKeyVault - Modules > Browse Gallery

Browse Gallery

Search: Azurerm.profile

AzureRM.profile

Microsoft Azure PowerShell - Profile credential management cmdlets for Azure Resource Manager

Tags: Azure ResourceManager ARM Profile Authentication Environment Subscription PSModule

Created by: azure-sdk
23499238 downloads
Last updated: 11/21/2018

10. Nos resultados da pesquisa, selecione `AzureRM.profile` e clique no botão **Importar** no menu superior. Por fim, clique no botão **OK**. Esta etapa leva alguns segundos para ser concluída; após alguns segundos, o módulo deve ser instalado, como mostrado na seguinte captura de tela:

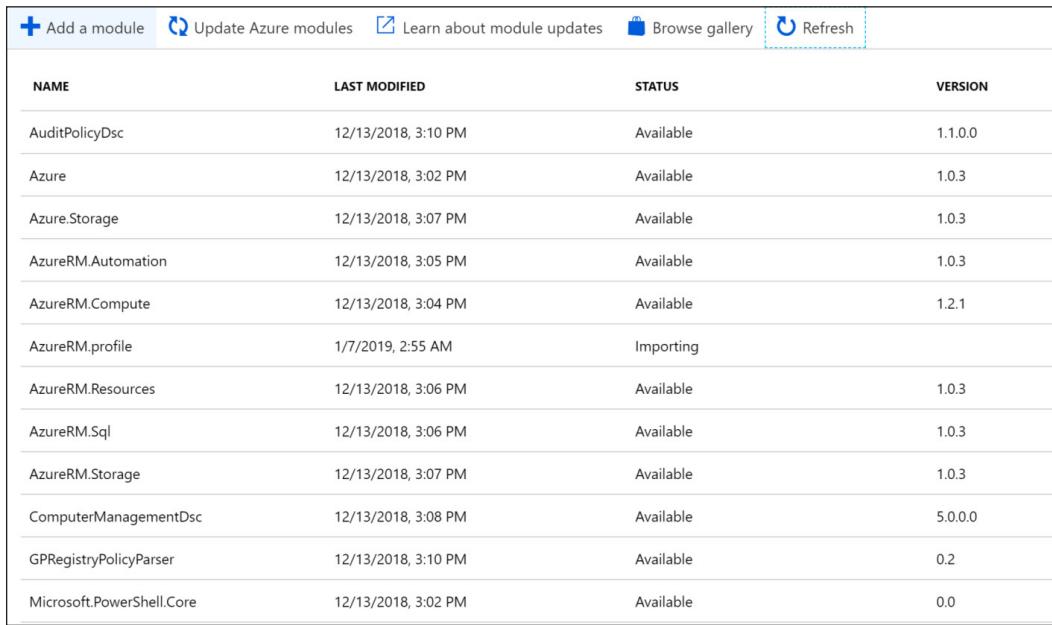
The screenshot shows the Azure PowerShell Gallery page for the `AzureRM.profile` module. At the top, it says "AzureRM.profile" and "PowerShell Module". Below that is a "Import" button. The main content area includes the following information:

- Created by:** azure-sdk
- Tags:** Azure ResourceManager ARM Profile Authentication Environment Subscription PSModule
- View Source Project**
- Learn more**: [View in PowerShell Gallery](#), [Documentation](#), [Licensing information](#)
- Content**: A table with a search bar at the top. The columns are **TYPE** and **NAME**. The data is as follows:

TYPE	NAME
Cmdlet	Disable-AzureRmDataCollection
Cmdlet	Disable-AzureRmContextAutosave
Cmdlet	Enable-AzureRmDataCollection
Cmdlet	Enable-AzureRmContextAutosave
Cmdlet	Remove-AzureRmEnvironment
Cmdlet	Get-AzureRmEnvironment

Soluções de integração do Azure

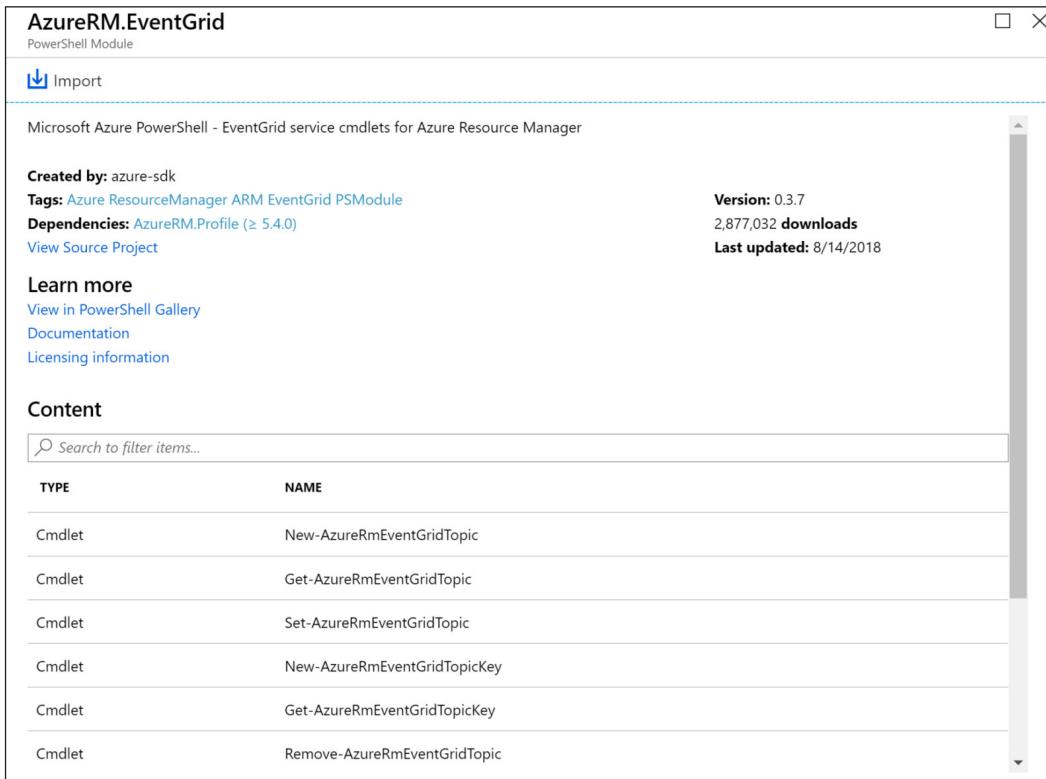
11. O status da instalação pode ser verificado no item de menu **Módulo**. A captura de tela a seguir demonstra como podemos importar um módulo:



The screenshot shows a table of Azure PowerShell modules. The columns are NAME, LAST MODIFIED, STATUS, and VERSION. One row, 'AzureRM.profile', is highlighted with a blue border and has a status of 'Importing'.

NAME	LAST MODIFIED	STATUS	VERSION
AuditPolicyDsc	12/13/2018, 3:10 PM	Available	1.1.0.0
Azure	12/13/2018, 3:02 PM	Available	1.0.3
Azure.Storage	12/13/2018, 3:07 PM	Available	1.0.3
AzureRM.Automation	12/13/2018, 3:05 PM	Available	1.0.3
AzureRM.Compute	12/13/2018, 3:04 PM	Available	1.2.1
AzureRM.profile	1/7/2019, 2:55 AM	Importing	
AzureRM.Resources	12/13/2018, 3:06 PM	Available	1.0.3
AzureRM.Sql	12/13/2018, 3:06 PM	Available	1.0.3
AzureRM.Storage	12/13/2018, 3:07 PM	Available	1.0.3
ComputerManagementDsc	12/13/2018, 3:08 PM	Available	5.0.0.0
GPRegistryPolicyParser	12/13/2018, 3:10 PM	Available	0.2
Microsoft.PowerShell.Core	12/13/2018, 3:02 PM	Available	0.0

12. Execute as etapas 9, 10 e 11 novamente para importar e instalar o módulo AzureRM.EventGrid:



The screenshot shows the PowerShell Gallery interface for the AzureRM.EventGrid module. At the top, it says "AzureRM.EventGrid" and "PowerShell Module". Below that is a large blue button labeled "Import". A sub-header reads "Microsoft Azure PowerShell - EventGrid service cmdlets for Azure Resource Manager". To the left, there's a sidebar with "Learn more" links: "View in PowerShell Gallery", "Documentation", and "Licensing information". To the right, there are details about the module: "Created by: azure-sdk", "Tags: Azure ResourceManager ARM EventGrid PSModule", "Dependencies: AzureRM.Profile (≥ 5.4.0)", "Version: 0.3.7", "2,877,032 downloads", and "Last updated: 8/14/2018". The main content area is titled "Content" and contains a table with a single column header "NAME" and seven rows of cmdlet names:

TYPE	NAME
Cmdlet	New-AzureRmEventGridTopic
Cmdlet	Get-AzureRmEventGridTopic
Cmdlet	Set-AzureRmEventGridTopic
Cmdlet	New-AzureRmEventGridTopicKey
Cmdlet	Get-AzureRmEventGridTopicKey
Cmdlet	Remove-AzureRmEventGridTopic

Soluções de integração do Azure

13. Execute as etapas 9, 10 e 11 novamente para importar e instalar o módulo `AzureRM.KeyVault`:

The screenshot shows the PowerShell Gallery interface. A search bar at the top contains the text "keyvault". Below it, a list of modules is displayed:

- AzureRM.KeyVault**: Microsoft Azure PowerShell - KeyVault service cmdlets for Azure Resource Manager. Created by: azure-sdk, 8578130 downloads, Last updated: 8/29/2018.
- Az.KeyVault**: Microsoft Azure PowerShell - Key Vault service cmdlets for Azure Resource Manager in Windows PowerShell and PowerShell Core. Created by: azure-sdk, 75510 downloads, Last updated: 12/18/2018.
- Az.KeyVault.Admin**: KeyVault Admin Client. Created by: azur-sdk, 12592 downloads, Last updated: 9/17/2018.
- CredentialStore.AzureKeyVault**: Import and Export functionality to sync CredentialStore with Azure KeyVault. Created by: fodonnel, 203 downloads, Last updated: 7/8/2018.
- cAzureKeyVault**: Community Azure Key Vault DSC resource for retrieving Key Vault secrets. Created by: nshenoy, 374 downloads, Last updated: 11/16/2017.
- PSBuildSecrets**: This module contains helper functions for build automation variable setup using azure kev vault. Created by: synax, 463 downloads.

To the right of the search results, a detailed view of the **AzureRM.KeyVault** module is shown:

- Import** button.
- Microsoft Azure PowerShell - KeyVault service cmdlets for Azure Resource Manager**.
- Created by: azure-sdk**, **Tags: Azure ResourceManager ARM KeyVault PSMODULE**, **Dependencies: AzureRM.Profile (≥ 5.5.1)**.
- View Source Project**.
- Learn more**: [View in PowerShell Gallery](#), [Documentation](#), [Licensing information](#).
- Content**: A table listing cmdlets:

TYPE	NAME
Cmdlet	Add-AzureKeyVaultCertificate
Cmdlet	Update-AzureKeyVaultCertificate
Cmdlet	Stop-AzureKeyVaultCertificateOperation
Cmdlet	Get-AzureKeyVaultCertificateOperation
Cmdlet	Import-AzureKeyVaultCertificate
Cmdlet	Add-AzureKeyVaultCertificateContent

Etapa 6

O comando usado para criar um tópico de Grade de Eventos usando o PowerShell é o seguinte:

```
New-AzureRmEventGridTopic
```

O processo de criação de um tópico de Grade de Eventos usando o portal do Azure é o seguinte:

1. No portal do Azure, navegue até o grupo de recursos **Vaultmonitoring** já criado clicando no ícone **Grupos de Recursos** no menu à esquerda.
2. Em seguida, clique em **+Adicionar** e procure **Tópico de Grade de Eventos** na caixa de pesquisa; selecione-o e clique no botão **Criar**:

Marketplace

Event Grid Topic

Pricing: All | Operating System: All | Publisher: All

Results

NAME	PUBLISHER	CATEGORY
Event Grid Topic	Microsoft	
Event Grid Domain	Microsoft	
Azure Blockchain Workbench	Microsoft	Compute

Event Grid Topic
Microsoft

Send custom events anywhere by using application topics to connect to Azure. Application topics provide a SAS authenticated DNS entry point for custom events to leverage Azure Event Grid's deep integration with other services. You can use your Application Topic just as you would any native Azure resource.

Use Application Topics to:

- Send custom events to Azure Event Grid.
- Leverage Azure Event Grid to route and filter custom events.
- Subscribe to events in your application topic - private to you.
- Reliably deliver custom events to Azure or 3rd party services.

Save for later

+ Event Subscription

NAME	ENDPOINT URL	Subscription
subName	https://requestbin.net/1nqz0t1	Azure Event Grid - Dev/Test
kvtest11	https://requestbin.net/va0fgos	
subName2	https://requestbin.net/va0fgos	
subName	https://requestbin.net/va0fgos	
mysecretfeed	https://requestbin.net/va0fgos	
requestbin	https://requestbin.net/7dfqjut	
adfunction	https://requestbin.net/8gkqfut	
nodejs	http://1178.192.22.1137/	
subName	https://requestbin.net/va0fgos	

Create

3. Preencha os valores apropriados no formulário resultante fornecendo um nome, selecionando uma assinatura, selecionando o grupo de recursos recém-criado, o local e o esquema do evento:

Create Topic
Event Grid

* Name
ExpiredAssetsKeyVaultEvents

* Subscription
RiteshSubscription

* Resource group
VaultMonitoring
[Create new](#)

* Location
West Europe

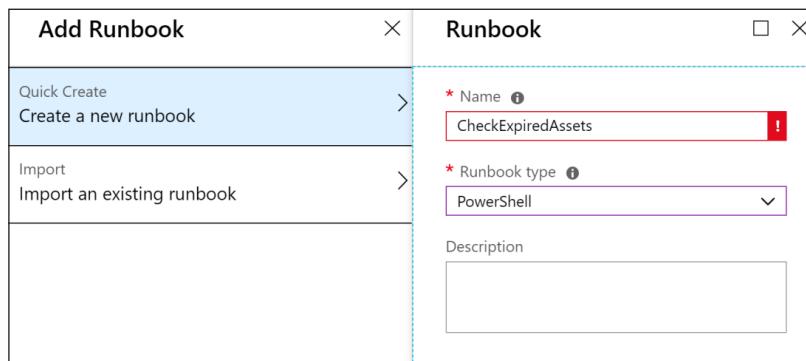
Event Schema
Event Grid Schema

Create

Etapa 7

Esta etapa se concentrará na criação de uma conta da Automação do Azure e Runbooks do PowerShell que conterá a lógica principal de ler Azure Key Vaults e recuperar segredos armazenados dentro dele. As etapas necessárias para configurar a Automação do Azure são mencionadas a seguir:

1. **Criar o runbook da Automação do Azure:** no portal do Azure, navegue até o grupo de recursos Vaultmonitoring já criado clicando no ícone **Grupos de Recursos** no menu à esquerda:
 1. Clique na conta da Automação do Azure já provisionada, MonitoringKeyVault. Em seguida, clique em **Runbooks** no menu à esquerda e clique em **+Adicionar um Runbook** no menu superior.
 2. Clique em **Criar um novo Runbook** e forneça um nome. Vamos chamar este runbook de `CheckExpiredAssets` e então definir o **Tipo de runbook** como **PowerShell**:



2. **Codificar o runbook:** declare algumas variáveis para manter a ID da assinatura, ID do locatário, ID do aplicativo e as informações de impressão digital do certificado. Esses valores devem ser armazenados em variáveis da Automação do Azure e o upload do certificado deve ser feito nos certificados da Automação. A chave usada para o certificado obtido por upload é \$RiteshSubscriptionCertificate. Os valores são recuperados desses armazenamentos e são atribuídos às variáveis, conforme mostrado a seguir:

```
$subscriptionID = get-AutomationVariable "azuresubscriptionid"  
$tenantID = get-AutomationVariable "azuretenantid"  
$applicationId = get-AutomationVariable "azureappid"  
$cert = get-AutomationCertificate "RitestSubscriptionCertificate"  
$certThumbprint = ($cert.Thumbprint).ToString()
```

O próximo código no runbook ajuda a fazer logon no Azure usando a entidade de serviço com valores de variáveis declaradas anteriormente. Além disso, o código seleciona uma assinatura apropriada. O código é mostrado a seguir.

```
Login-AzureRmAccount -ServicePrincipal -CertificateThumbprint
$certThumbprint -ApplicationId $applicationId -Tenant $tenantID
Set-AzureRmContext -SubscriptionId $subscriptionID
```

Como a Grade de Eventos do Azure foi provisionada na *etapa 6* desta seção, seu ponto de extremidade e suas chaves são recuperados usando os cmdlets `Get-AzureRmEventGridTopic` e `Get-AzureRmEventGridTopicKey`.

Cada Grade de Eventos do Azure gera duas chaves – primária e secundária; a primeira referência chave é tomada como segue:

```
$eventGridName = "ExpiredAssetsKeyVaultEvents"
$eventGridResourceGroup = "VaultMonitoring"
$topicEndpoint = (Get-AzureRmEventGridTopic -ResourceGroupName
$eventGridResourceGroup -Name $eventGridName).Endpoint
$keys = (Get-AzureRmEventGridTopicKey -ResourceGroupName
$eventGridResourceGroup -Name $eventGridName).Key1
```

Em seguida, todos os cofres de chaves que foram provisionados na assinatura são recuperados usando a iteração. Durante o loop, todos os segredos são recuperados usando o cmdlet `Get-AzureKeyVaultSecret`.

A data de expiração de cada segredo é comparada à data atual e, se a diferença for inferior a um mês, gerará um evento de Grade de Eventos e o publicará usando o comando `Invoke-WebRequest`.

As mesmas etapas são executadas para certificados armazenados no cofre de chaves. O cmdlet usado para recuperar todos os certificados é `Get-AzureKeyVaultCertificate`.

O evento que é publicado na Grade de Eventos deve estar na matriz JSON. A mensagem gerada é convertida em JSON usando o cmdlet `ConvertTo-Json` e, em seguida, convertida em uma matriz por meio da adição de [e] como um prefixo e sufixo.

Para se conectar à Grade de Eventos do Azure e publicar o evento, o remetente deverá fornecer a chave em seu cabeçalho. A solicitação falhará se esses dados estiverem ausentes na carga da solicitação:

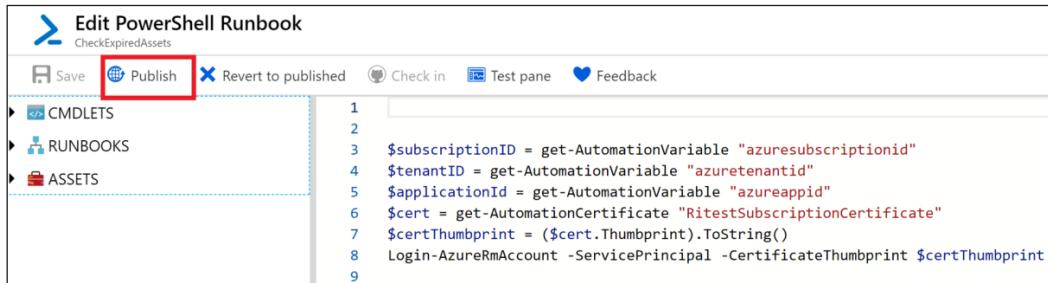
```
$keyvaults = Get-AzureRmKeyVault
foreach($vault in $keyvaults) {
    $secrets = Get-AzureKeyVaultSecret -VaultName $vault.VaultName
    foreach($secret in $secrets) {
        if( ! [string]::IsNullOrEmpty($secret.Expires) ) {
            if($secret.Expires.AddMonths(-1) -lt [datetime]::Now)
            {
                $secretDataMessage = @{
                    "secretName" = $secret.Name
                    "secretValue" = $secret.Value
                    "secretType" = $secret.SecretType
                    "secretExpires" = $secret.Expires
                }
                $secretDataMessage | ConvertTo-Json | Add-Content -Path $secretPath
            }
        }
    }
}
```

Soluções de integração do Azure

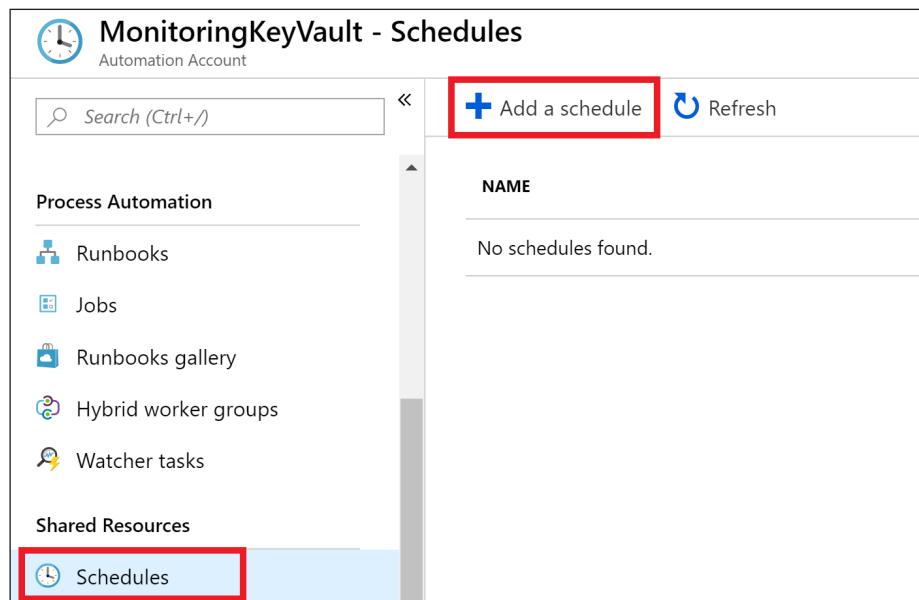
```
id = [System.guid]::NewGuid()
subject = "Secret Expiry happening soon !!"
eventType = "Secret Expiry"
eventTime = [System.DateTime]::UtcNow
data = @{
    "ExpiryDate" = $secret.Expires
    "SecretName" = $secret.Name.ToString()
    "VaultName" = $secret.VaultName.ToString()
    "SecretCreationDate" = $secret.Created.ToString()
    "IsSecretEnabled" = $secret.Enabled.ToString()
    "SecretId" = $secret.Id.ToString()
}
}

...
Invoke-WebRequest -Uri $topicEndpoint -Body $finalBody -Headers
$header -Method Post -UseBasicParsing
}
}
}
Start-Sleep -Seconds 5
}
}
```

Publique o runbook clicando no botão **Publicar**, conforme mostrado na captura de tela a seguir:



3. **Agendador** crie um ativo de agendador da Automação do Azure para executar esse runbook uma vez por dia à meia-noite. As etapas para criar um Agendamento da Automação do Azure são mencionadas a seguir:
 1. Clique em **Agendamentos** no menu à esquerda da Automação do Azure e clique no botão **+Adicionar um agendamento** no menu superior, conforme mostrado na captura de tela a seguir:



2. Forneça informações de agendamento no formulário resultante:

New Schedule □ X

* Name
ExecuteMonitoringRunbook ✓

Description
This schedule execute the Azure key vault monitoring runbook every day one at 12:00 AM to check for expired assets ✓

* Starts i
2019-01-11 12:00 AM

UTC

Recurrence
Once Recurring

* Recur every
1 Day ▼

Set expiration
Yes No

Expires
Never

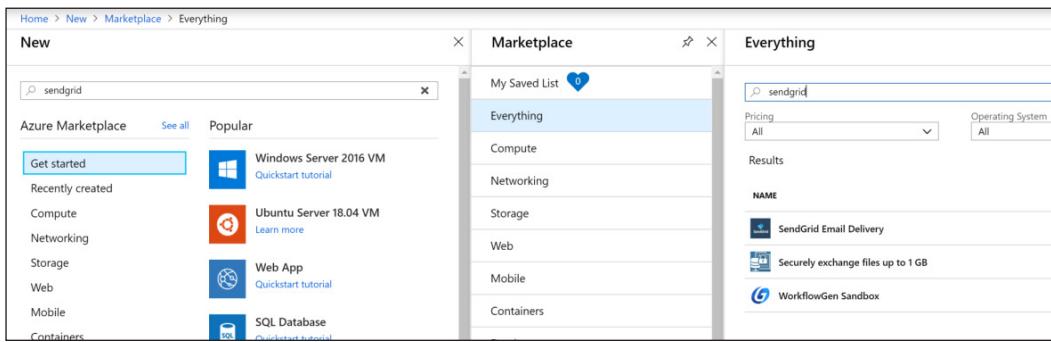
Create

Isso deve concluir a configuração da conta da Automação do Azure.

Etapa 8

Nesta etapa, criaremos um novo recurso SendGrid. O recurso SendGrid é usado para enviar emails do aplicativo sem a necessidade de instalar um servidor **Simple Mail Transfer Protocol (SMTP)**. Ele fornece uma API REST e um **Kit de Desenvolvimento de Software (SDK)** do C#, por meio do qual é muito fácil enviar emails em massa. Na solução atual, o Azure Functions será usado para invocar as APIs do SendGrid para enviar emails e, portanto, esse recurso precisa ser provisionado. Esse recurso tem custos separados e não é abordado como parte do custo do Azure; há um nível gratuito disponível e pode ser usado para enviar emails:

1. Um recurso SendGrid é criado como qualquer outro recurso do Azure. Procure `sendgrid`, e nós obteremos **Entrega de Email de SendGrid** nos resultados:



Soluções de integração do Azure

2. Selecione o recurso e clique no botão Criar para abrir seu formulário de configuração:

The screenshot shows the SendGrid Email Delivery configuration page. At the top, there's a header with the title "SendGrid Email Delivery" and a "SendGrid" logo. Below the header, a paragraph of text describes SendGrid's service, mentioning it's the world's largest cloud-based service for delivering email. It highlights that SendGrid's proven platform successfully delivers over 18B transactional and marketing related emails each month for Internet and mobile-based customers like Airbnb, Pandora, Hubspot, Spotify, Uber and FourSquare as well as more traditional enterprises like Walmart, Intuit and Costco. A "Save for later" button is visible at the bottom left of this section.

Azure customers receive up to 25,000 emails per month for free with paid packages starting at only \$9.95 per month. For customers requiring ability to send larger email volumes, SendGrid also offers Silver, Gold, Platinum and Premier packages, which include a dedicated IP as well as additional IP and user management features.

The main content area displays a "Statistics Overview" dashboard. It includes a summary card with "REQUESTS: 483,418", "UNIQUE OPENS: 17.61%", and "UNIQUE CLICKS: 3.81%". Below this is a line chart showing various metrics over time, with a legend listing items like REQUESTS, DELIVERED, BOUNCE, SPAM, UNSUBSCRIBE, CLICK, UNSUBSCRIBE, SPAM REPORTS, BLOCKED, and UNKNOWN BOUNCE. A sidebar on the left provides links to other statistics and geographical reports. At the bottom, there's a section for selecting a software plan, with "Free" selected (25,000 emails per month) and a "Create" button.

3. Selecione um tipo de preço apropriado, conforme mostrado na captura de tela a seguir:

Choose your pricing tier			<input type="checkbox"/>	
For customers requesting the Premier Volume Plan (5 million or more emails/month), a promotional code is required to activate the discount rate. To get started, email your request to accountsuccess@sendgrid.com			Recommended View all	
S2 Silver	S1 Bronze	F1 Free		
100,000 emails/month	40,000 emails/month	25,000 emails/month		
Advanced Reporting Open, Click, Bounce, Unsu...	Advanced Reporting Open, Click, Bounce, Unsu...	Advanced Reporting Open, Click, Bounce, Unsu...		
Custom Integration A... SMTP API, Web API, Event ...	Custom Integration A... SMTP API, Web API, Event ...	Custom Integration A... SMTP API, Web API, Event ...		
Advanced Delivery Fe... DKIM, SPF, Reputation Mo...	Advanced Delivery Fe... DKIM, SPF, Reputation Mo...	Advanced Delivery Fe... DKIM, SPF, Reputation Mo...		
Dedicated IP Address IP Whitelabeling, Automat...	Support 24x7 Phone, Chat, Email S...	Support 24x7 Phone, Chat, Email S...		
Sub-User Management Create and manage sub-us...				
Support 24x7 Phone, Chat, Email S...				
79.95 USD/MONTH (ESTIMATED)	9.95 USD/MONTH	0.00 USD/MONTH		

4. Forneça os detalhes de contato apropriados, conforme listado na seguinte captura de tela:

Contact Information

Enter your contact details.

The provided information will be used as contact info for support agents / technical contacts.

* First Name
Ritesh

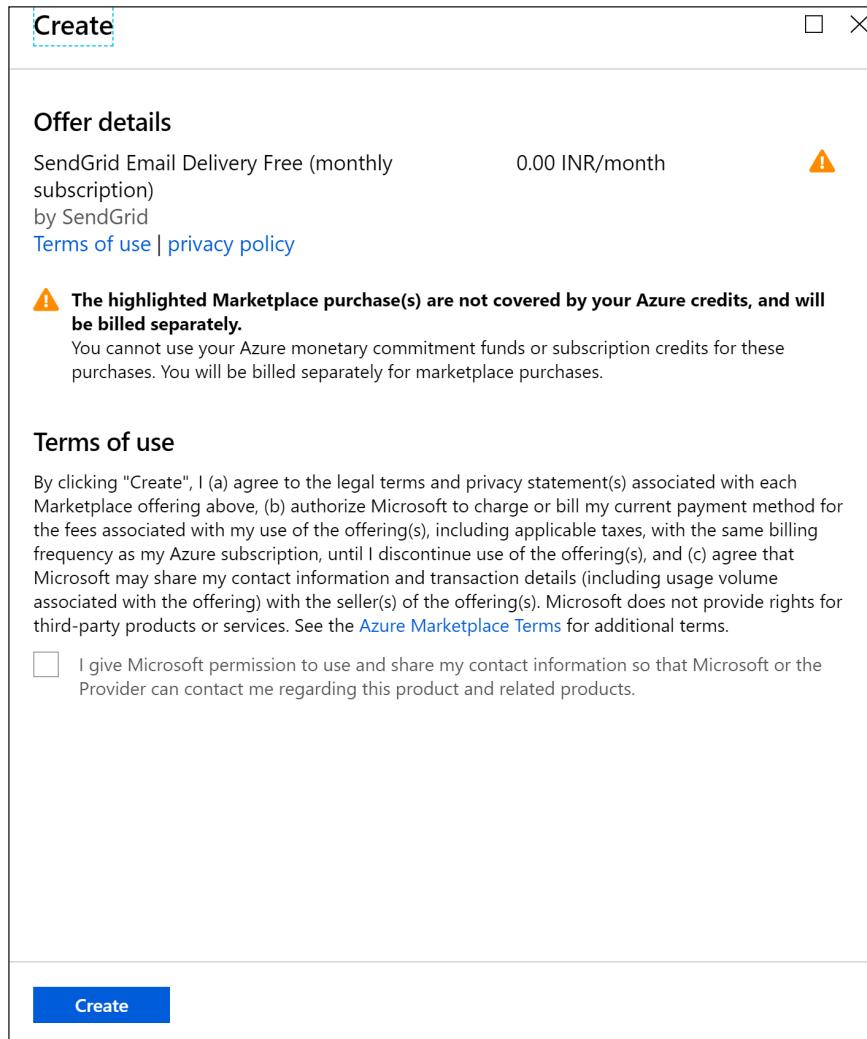
* Last Name
Modi

* Email
callritz@hotmail.com

Company
self

Website
automationnext.wordpress.com

5. Aceite a caixa de seleção **Termos de uso**, conforme mostrado na captura de tela a seguir:



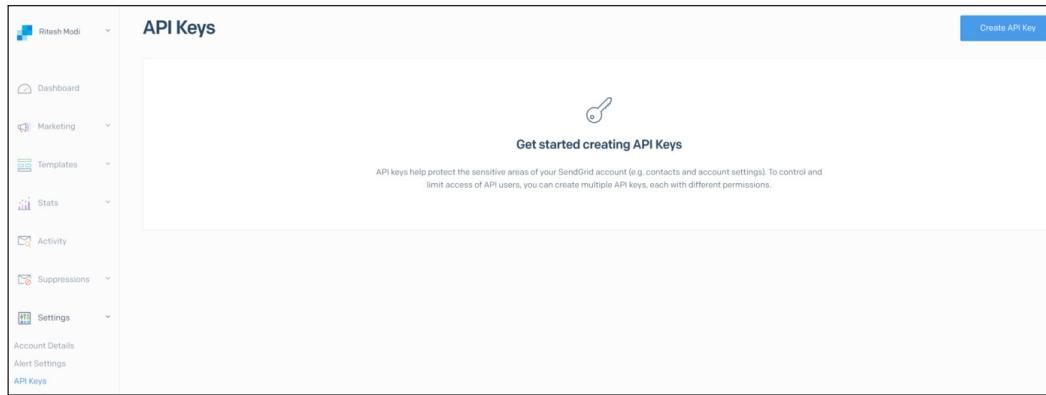
6. Preencha o formulário e então clique no botão **Criar**, como mostrado na captura de tela a seguir:

The screenshot shows the 'Create a New SendGrid A...' dialog box. It includes the following fields:

- Name: keyvaultmonitoring
- Password: (redacted)
- Confirm Password: (redacted)
- Subscription: RiteshSubscription
- Resource group:
 - Create new
 - Use existingVaultMonitoring
- Pricing tier: free
- Promotion Code: (empty)
- Contact Information: Completed.
- Legal terms: Legal terms accepted

At the bottom are two buttons: 'Create' (highlighted in blue) and 'Automation options'.

7. Depois que o recurso for provisionado, clique no botão **Gerenciar** no menu superior – isso abrirá o site do SendGrid. O site pode solicitar configuração de email. Em seguida, selecione **Chaves de API** na seção **Configurações** e clique no botão **Criar Chave de API**:



8. Na janela resultante, selecione **Acesso Completo** e clique no botão **Criar e Exibir**. Isso criará as chaves para o recurso SendGrid; anote essa chave, pois ela será usada com a configuração do Azure Function para SendGrid:

Create API Key

API Key Name *

API Key Permissions* ⓘ

 **Full Access**
Allows the API key to access GET, PATCH, PUT, DELETE, and POST endpoints for all parts of your account, excluding billing.

 **Restricted Access**
Customize levels of access for all parts of your account, excluding billing.

 **Billing Access**
Allows the API key to access billing endpoints for the account. (This is especially useful for Enterprise or Partner customers looking for more advanced account management.)

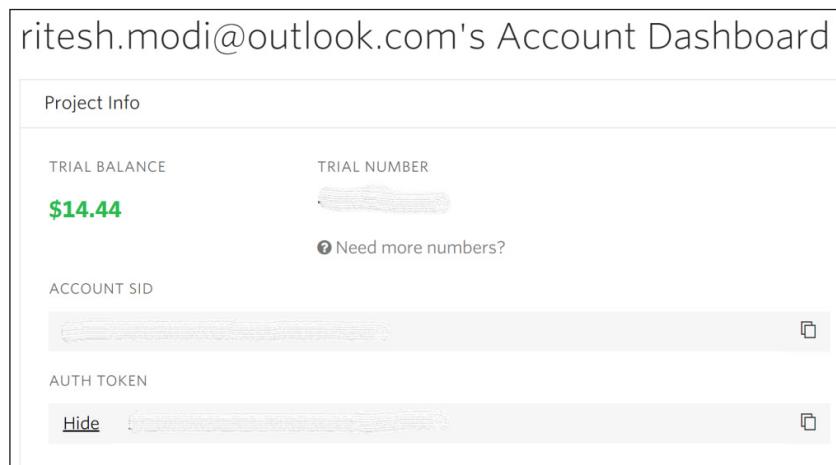
Create & View

Etapa 9

Nesta etapa, criaremos uma nova conta do Twilio. O Twilio é usado para enviar mensagens SMS em massa. Para criar uma conta com o Twilio, navegue até [twilio.com](#) e crie uma nova conta. Depois de criar uma conta com êxito, um número de celular será gerado e poderá ser usado para enviar mensagens SMS para receptores:



A conta do Twilio fornece chaves de produção e teste. Copie a chave de teste e o token para um local temporário, como o Bloco de Notas, pois eles serão necessários posteriormente no Azure Functions:



Etapa 10

Nesta etapa, criaremos uma nova função do Azure responsável pelo envio de emails e notificações por SMS. A finalidade da função do Azure na solução é enviar mensagens de notificação aos usuários sobre a expiração de segredos no cofre de chaves. Uma única função será responsável pelo envio de emails e mensagens SMS - note que isso poderia ter sido dividido em duas funções separadas. A primeira etapa é criar um novo aplicativo de função e hospedar uma função dentro dele:

1. Como fizemos antes, navegue até o seu grupo de recursos, clique no botão **+Adicionar** no menu superior e procure o recurso de aplicativo de função resource. Em seguida, clique no botão **Criar** para obter o formulário **Aplicativo de Função**:

The screenshot shows the Azure Marketplace interface. On the left, a search bar contains 'function app'. Below it are filters for Pricing (All), Operating System (All), and Publisher (All). The results list includes 'Function App' by Microsoft, which is highlighted with a blue selection bar. Other items listed include 'Functions Bot', 'F5 Per-App VE – BIG-IP LTM (PAYG, 200Mbps)', 'F5 Per-App VE – BIG-IP LTM (PAYG, 25Mbps)', 'F5 Per-App VE – Advanced WAF (PAYG, 25Mbps)', 'F5 Per-App VE – Advanced WAF (PAYG, 200Mbps)', 'Fusio open source API', 'Logic Apps Custom Connector', and 'F5 Per-App VE – Advanced WAF (BYOL)'. At the bottom, there are related items: 'CakePHP' by Cake Software Foundation, 'SlashDB Cloud Edition' by vt.enterprise, and 'Web App + MySQL' by Microsoft. On the right, the 'Function App' blade is displayed, showing its publisher as Microsoft, useful links to Documentation, Solution Overview, and Pricing Details, and a large 'Create' button at the bottom.

2. Preencha o formulário **Aplicativo de Função** e clique no botão **Criar**.
O nome do aplicativo de função deve ser exclusivo em todo o Azure:

The screenshot shows the 'Function App' creation dialog. The 'App name' field is set to 'NotificationFunctionAppBook'. The 'Subscription' is 'RiteshSubscription'. The 'Resource Group' is 'VaultMonitoring'. The 'OS' is 'Windows'. The 'Hosting Plan' is 'Consumption Plan'. The 'Location' is 'West Europe'. The 'Runtime Stack' is '.NET'. The 'Storage' option is 'Create new', and the storage account name is 'notificationfuna2a9'. At the bottom, there is an 'Application Insights' section with 'NotificationFunctionAppBook' selected, and 'Create' and 'Automation options' buttons.

Function App

Create

* App name
NotificationFunctionAppBook .azurewebsites.net

* Subscription
RiteshSubscription

* Resource Group
VaultMonitoring

* OS
Windows Linux (Preview)

* Hosting Plan
Consumption Plan

* Location
West Europe

* Runtime Stack
.NET

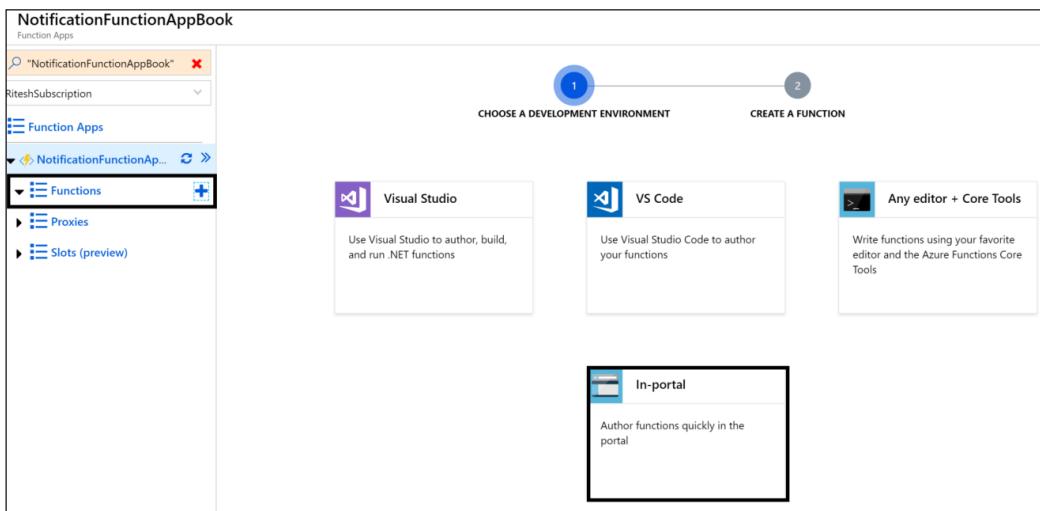
* Storage
Create new Use existing

notificationfuna2a9

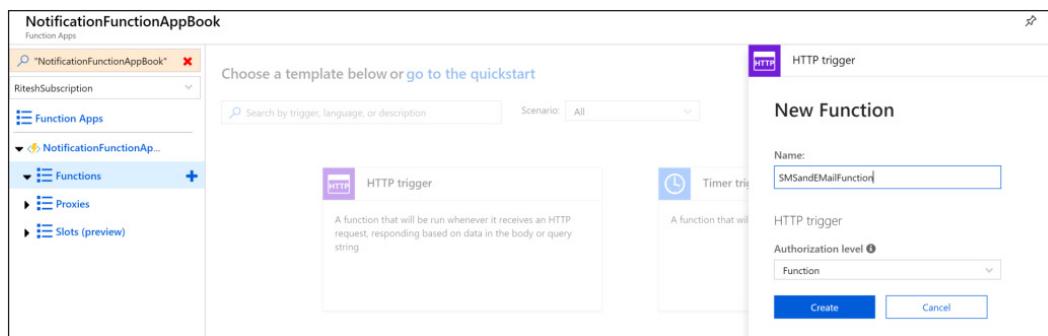
Application Insights >
NotificationFunctionAppBook

Create Automation options

3. Depois que o aplicativo de função for provisionado, crie uma nova função chamada **SMSandEMailFunction** clicando no botão **+** ao lado do item **Funções** no menu à esquerda. Em seguida, selecione **In-portal** no painel central. Isso é mostrado a seguir:



4. Selecione **Gatilho HTTP** e chame-o de **SMSandEMailFunction**. Em seguida, clique no botão **Criar**; a opção **Nível de autorização** pode ter qualquer valor:



5. Remova o código padrão, substitua-o pelo código mostrado na lista a seguir e clique no botão **Salvar** no menu superior:

```
#r "SendGrid"
#r "Newtonsoft.Json"
#r "Twilio.Api"
using System.Net;
using System;
using SendGrid.Helpers.Mail;
using Microsoft.Azure.WebJobs.Host;
using Newtonsoft.Json;
using Twilio;
using System.Configuration;
public static HttpResponseMessage Run(HttpRequestMessage req,
TraceWriter log, out Mail message,out SMSMessage sms)
{
    log.Info("C# HTTP trigger function processed a request.");
    string alldata = req.Content.ReadAsStringAsync().GetAwaiter().
    GetResult();
    message = new Mail();
    var personalization = new Personalization();
    personalization.AddBcc(new Email(ConfigurationManager.AppSettings[
    "bccStakeholdersEmail"]));
    personalization.AddTo(new Email(ConfigurationManager.AppSettings[
    "toStakeholdersEmail"]));
    var messageContent = new Content("text/html", alldata);
    message.AddContent(messageContent);
    message.AddPersonalization(personalization);
    message.Subject = "Key Vault assets Expiring soon..";
    message.From = new Email(ConfigurationManager.
    AppSettings["serviceEmail"]);
    string msg = alldata;
    sms = new SMSMessage();
    sms.Body = msg;
    sms.To = ConfigurationManager.AppSettings["adminPhone"];
    sms.From = ConfigurationManager.AppSettings["servicePhone"];
    return req.CreateResponse(HttpStatusCode.OK, "Hello ");
}
```

6. Clique no nome do aplicativo de função no menu à esquerda e clique novamente no link **Configurações do aplicativo** na janela principal:

7. Navegue até a seção **Configurações do aplicativo**, conforme mostrado na captura de tela anterior, e adicione algumas entradas clicando em **+ Adicionar nova configuração** para cada entrada.

APP SETTING NAME	VALUE
APPINSIGHTS_INSTRUMENTATIONKEY	Hidden value. Click to edit.
AzureWebJobsStorage	Hidden value. Click to edit.
FUNCTIONS_EXTENSION_VERSION	Hidden value. Click to edit.
FUNCTIONS_WORKER_RUNTIME	Hidden value. Click to edit.
WEBSITE_CONTENTAZUREFILECONNECTIO...	Hidden value. Click to edit.
WEBSITE_CONTENTSHARE	Hidden value. Click to edit.
WEBSITE_NODE_DEFAULT_VERSION	Hidden value. Click to edit.
adminPhone	Hidden value. Click to edit.
servicePhone	Hidden value. Click to edit.
serviceEmail	Hidden value. Click to edit.
bccStakeholdersEmail	Hidden value. Click to edit.
toStakeholdersEmail	Hidden value. Click to edit.
SendGridAPIKeyAsAppSetting	Hidden value. Click to edit.
TwilioAccountSid	Hidden value. Click to edit.
TwilioAuthToken	Hidden value. Click to edit.

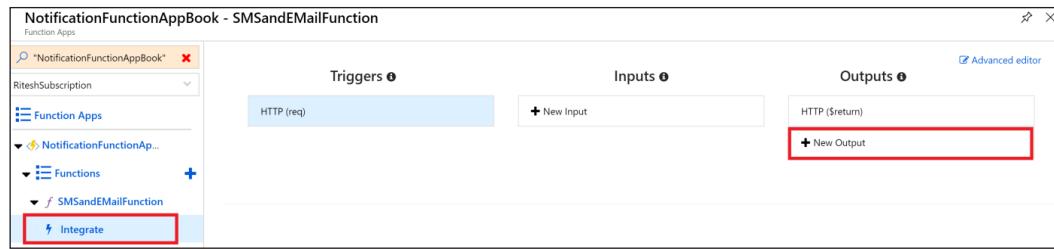
Observe que as entradas estão no formato de pares de chave-valor e os valores devem ser valores reais em tempo real. `adminPhone` e `servicePhone` já devem estar configurados no site do Twilio. `servicePhone` é o número de telefone gerado pelo Twilio que é usado para enviar mensagens SMS, e `adminPhone` é o número de telefone do administrador a quem o SMS deve ser enviado.

Observe também que o Twilio espera que o número de telefone de destino esteja em um formato específico, dependendo do país (para a Índia, o formato é +91 xxxxx xxxx). Observe os espaços e o código do país no número.

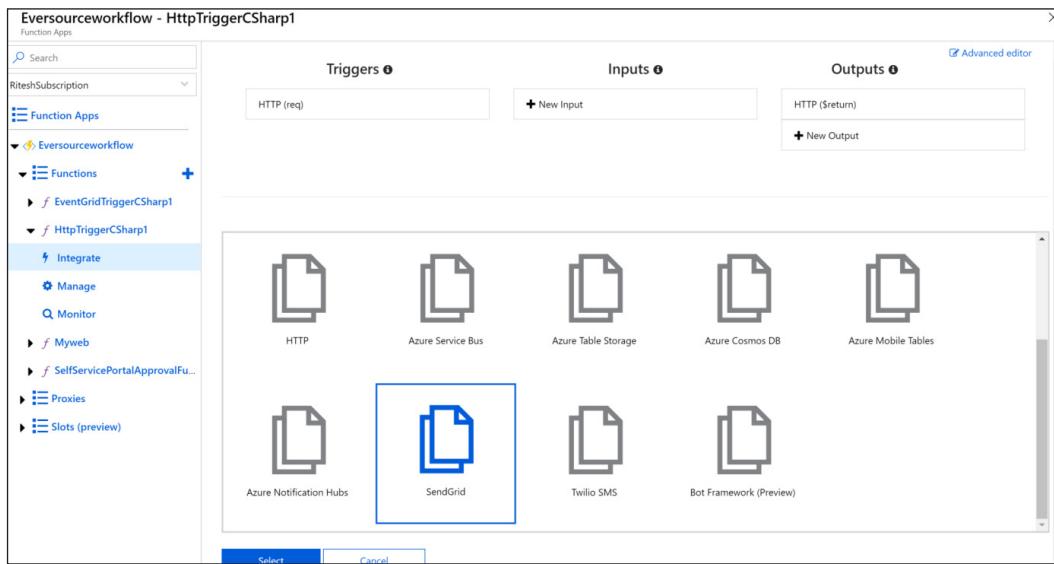
Também precisamos adicionar as chaves ao SendGrid e Twilio nas configurações do aplicativo. Essas configurações são mencionadas na lista a seguir. Os leitores já podem ter esses valores úteis devido a atividades executadas nas etapas anteriores:

- O valor de `SendGridAPIKeyAsAppSetting` é a chave para SendGrid.
- `TwilioAccountSid` é o identificador do sistema para a conta do Twilio. Esse valor já foi copiado e armazenado em um local transitório em uma etapa anterior.
- `TwilioAuthToken` é o token para a conta do Twilio. Esse valor já foi copiado e armazenado em um local temporário em uma etapa anterior.

8. Salve as configurações clicando no botão **Salvar** no menu superior;
9. Clique no link **Integrar** no menu à esquerda logo abaixo do nome da função e clique em **+ Nova Saída**. Isso é para adicionar uma saída para o serviço SendGrid:



10. Em seguida, selecione **SendGrid** – ele pode solicitar a instalação da extensão do SendGrid. Instale a extensão, o que levará alguns minutos:



11. Depois de instalar a extensão, o formulário de configuração de saída será exibido. Os itens de configuração importantes neste formulário são **Nome do parâmetro de mensagem** e **SendGridAPIKeyAppSetting**. Deixe o valor padrão para **Nome do parâmetro de mensagem** e clique na lista suspensa para selecionar **SendGridAPIKeyAsAppSetting** como a chave de configuração do aplicativo de API. Isso já foi configurado em uma etapa anterior na configuração de definições do aplicativo. O formulário deve ser configurado, como mostrado na captura de tela a seguir, então clique no botão **Salvar**:

The screenshot shows the 'SendGrid output' configuration dialog. It includes fields for 'Message parameter name' (set to 'message'), 'SendGrid API Key App Setting' (set to 'SendGridAPIKeyAsAppSetting'), 'From address', 'To address', 'Message subject', and 'Message Text'. There are also checkboxes for 'Use function return value' and 'Save' or 'Cancel' buttons. A 'Documentation' link is at the bottom.

SendGrid output

Message parameter name ⓘ
message

SendGrid API Key App Setting ⓘ show value
SendGridAPIKeyAsAppSetting new

From address ⓘ
From address

To address ⓘ
To address

Message subject ⓘ
Message subject

Message Text ⓘ
Message Text

Use function return value

Save Cancel

+ Documentation

12. Clique em **+ Nova Saída** novamente; isso serve para adicionar uma saída para o serviço Twilio.
13. Em seguida, selecione **SMS do Twilio**. Ele pode solicitar que você instale a extensão de SMS do Twilio. Instale a extensão, o que levará alguns minutos.

14. Depois de instalar a extensão, o formulário de configuração de saída será exibido. Os itens de configuração importantes neste formulário são **Nome do parâmetro de mensagem**, **Configuração de SID da conta** e **Configuração do Token de Autenticação**. Altere o valor padrão para o **Nome do parâmetro de mensagem** para sms. Isso é feito porque o parâmetro mensagem já é usado para o parâmetro de serviço SendGrid. Certifique-se de que o valor de **Configuração de SID da conta** seja TwilioAccountSid e que o valor da **Configuração do Token de Autenticação** seja TwilioAuthToken. Esses valores já foram configurados em uma etapa anterior na configuração de definições do aplicativo. O formulário deve ser configurado, como mostrado na captura de tela a seguir, então clique em **Salvar**:

The screenshot shows a configuration dialog for 'Twilio SMS output'. At the top, a message says 'Extension Installation Succeeded' with a green checkmark icon. Below this, there are several input fields and settings:

- Message parameter name:** A text input field containing 'sms'.
- Use function return value:** A checked checkbox.
- Auth Token setting:** A text input field containing 'TwilioAuthToken'.
- Account SID setting:** A text input field containing 'TwilioAccountSid'.
- From number:** A text input field containing 'From number'.
- Message text:** A text input field containing 'Message text'.

At the bottom of the dialog are two buttons: a blue 'Save' button and a white 'Cancel' button.

Etapa 11

Nesta etapa, criaremos um novo fluxo de trabalho de Aplicativo Lógico. Nós temos o autor de um runbook da Automação do Azure que consulta todos os segredos em todos os cofres de chaves e publica um evento caso ele encontre algum deles expirando em um mês. O fluxo de trabalho de Aplicativos Lógicos atua como um assinante para estes eventos:

1. A primeira etapa no menu **do aplicativo lógico** é criar um fluxo de trabalho de aplicativos lógicos:

The screenshot shows the Azure Marketplace interface. In the search bar, 'logic app' has been typed. The results table displays several items under the 'NAME' column, including 'Logic App', 'Logic Apps B2B', 'Logic Apps Custom Connector', 'Logic Apps Management (Preview)', 'Sumo Logic for Azure Web Apps', 'Sumo Logic for Azure Audit', 'RadiantOne on Linux', 'RadiantOne on Windows Server', and 'ASP.NET Starter Web App'. Each item includes a small icon, the publisher name (Microsoft or Sumo Logic), and its category (Management Tools, Compute, or Web). On the right side of the screen, there is a detailed description of 'Logic App' with sections like 'Easy to use design tools', 'Compose SaaS easily', 'Extensibility baked in', and 'Real integration horsepower'. Below the description is a 'Save for later' button and a preview window titled 'Logic App Designer' showing a simple workflow diagram.

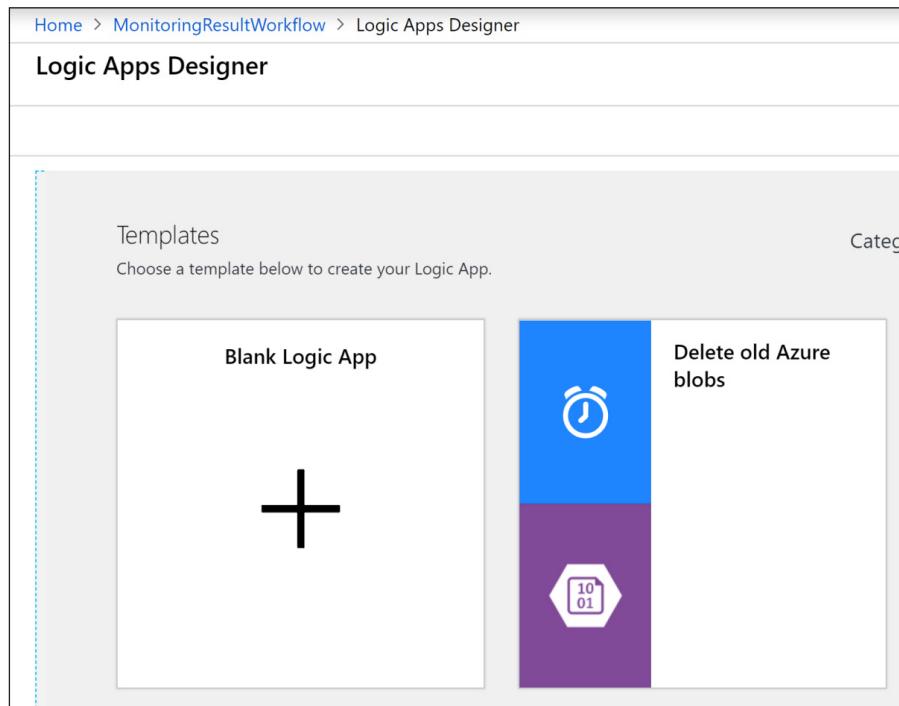
2. Preencha o formulário resultante depois de clicar no botão **Criar**. Estamos provisionando o aplicativo lógico no mesmo grupo de recursos do que os outros recursos para esta solução:

The screenshot shows the 'Logic App' creation wizard in the Azure portal. The steps are as follows:

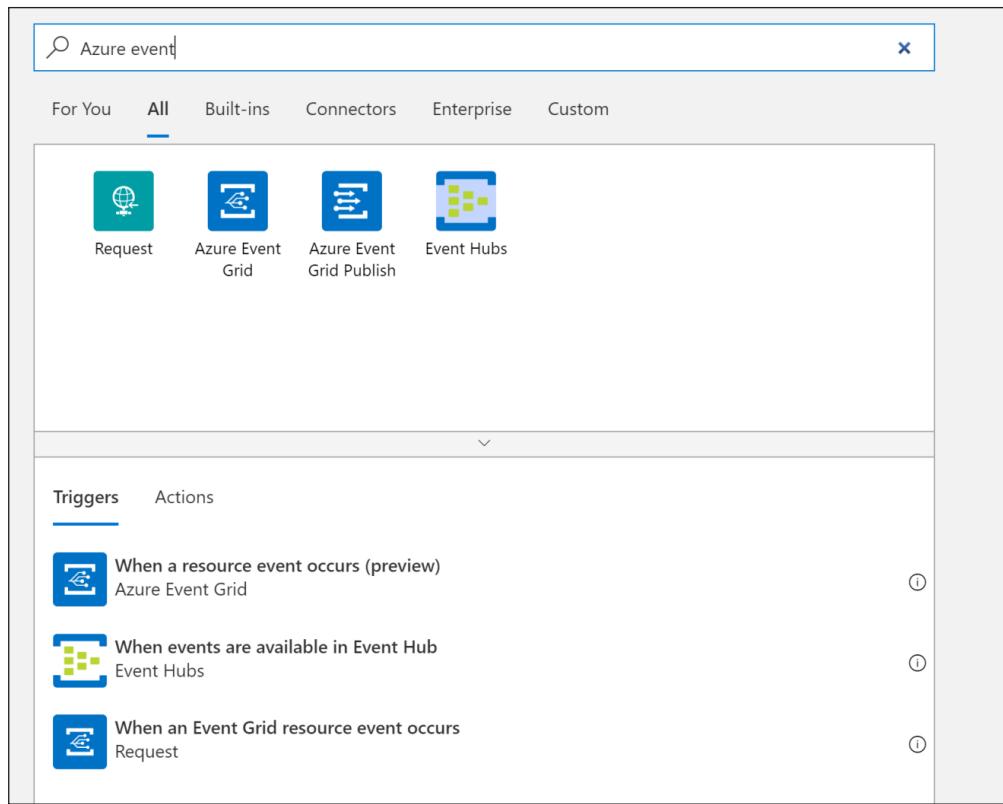
- Name:** MonitoringResultWorkflow
- Subscription:** RiteshSubscription
- Resource group:** Use existing (VaultMonitoring)
- Location:** West Europe
- Log Analytics:** On

A note at the bottom states: "You can add triggers and actions to your Logic App after creation." At the bottom right are the **Create** and **Automation options** buttons.

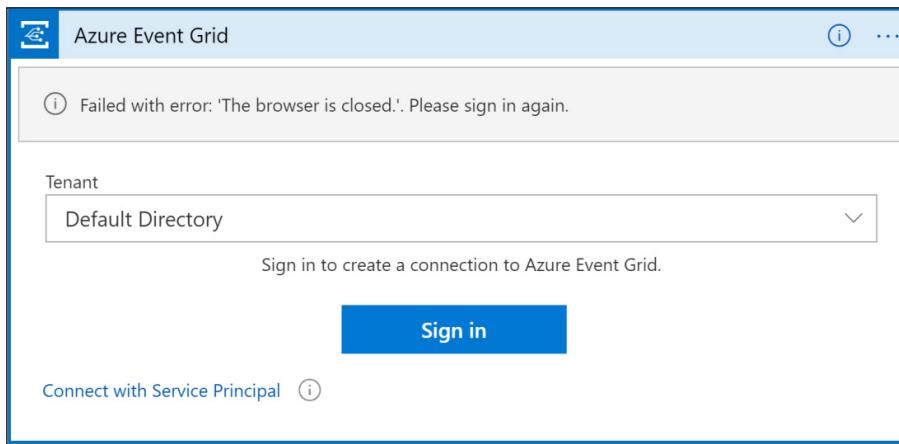
3. Depois que o aplicativo lógico for provisionado, ele abrirá a janela do designer. Selecione **Aplicativo Lógico em Branco** na seção **Modelos**, conforme demonstrado na captura de tela a seguir:



4. Na janela resultante, adicione um gatilho que pode assinar eventos da Grade de Evento. Os Aplicativos Lógicos fornecem um gatilho para a Grade de Eventos, e você pode procurar por isso para ver se ele está disponível:

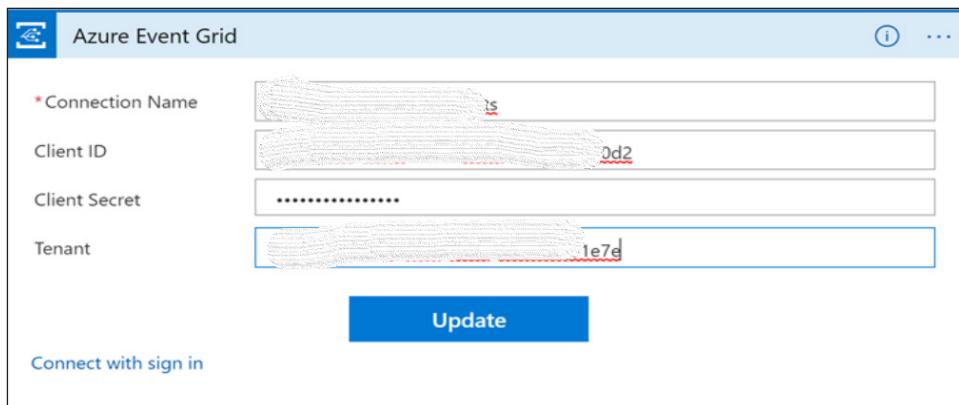


5. Em seguida, selecione o gatilho **Quando ocorrer um evento de recurso (visualização)**:

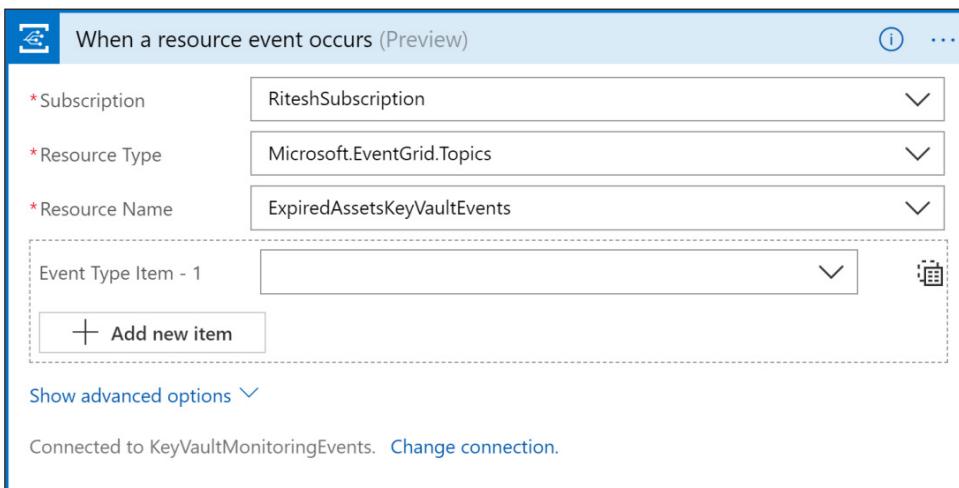


6. Na janela resultante, selecione **Conectar-se com Entidade de Serviço**.

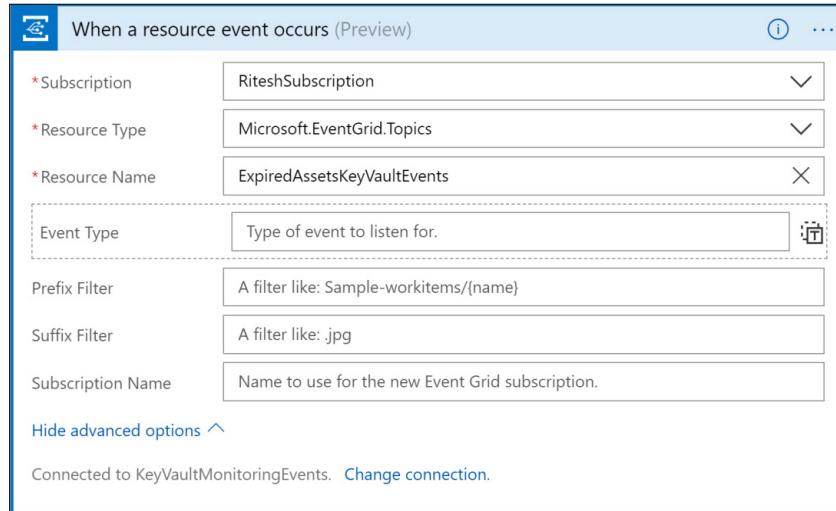
Forneça os detalhes da entidade de serviço, incluindo a ID de aplicativo (**ID do Cliente**), a ID do locatário e a senha. Este gatilho não aceita uma entidade de serviço que autentique com o certificado – ele aceita uma entidade de serviço somente com uma senha. Crie uma nova entidade de serviço neste estágio que autentique com uma senha (as etapas para criar uma entidade de serviço com base na autenticação de senha foram abordadas anteriormente na *Etapa 2* na etapa 2) e use os detalhes da entidade do aplicativo recém-criado para a configuração da Grade de Eventos do Azure, da seguinte maneira:



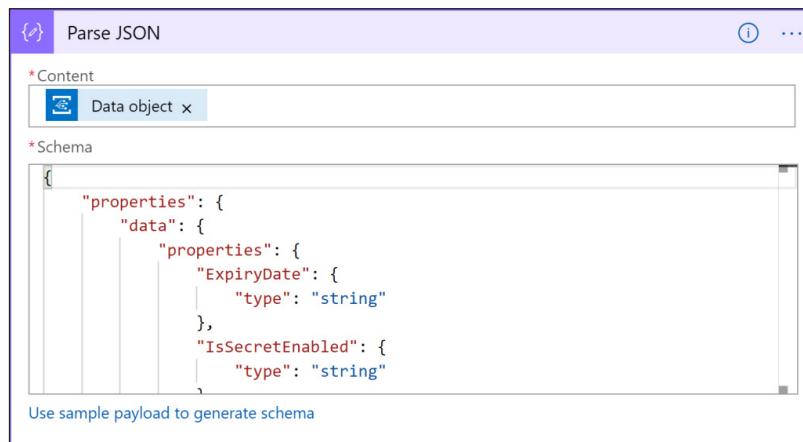
7. Selecione a assinatura. Com base no escopo da entidade de serviço, isso será preenchido automaticamente. Selecione `Microsoft.EventGrid.Topics` como o **Tipo de Recurso** e defina o nome do tópico personalizado como `ExpiredAssetsKeyVaultEvents`:



8. A etapa anterior criará um conector e as informações de conexão poderão ser alteradas clicando em **Alterar conexão**.
9. A configuração final do gatilho da Grade de Eventos deve ser semelhante à seguinte captura de tela:



10. Adicione uma nova atividade **Analisar JSON** depois do gatilho da Grade Eventos; essa atividade precisa do esquema JSON. Geralmente, o esquema não está disponível, mas essa atividade ajudará a gerar o esquema se um JSON válido for fornecido a ele:



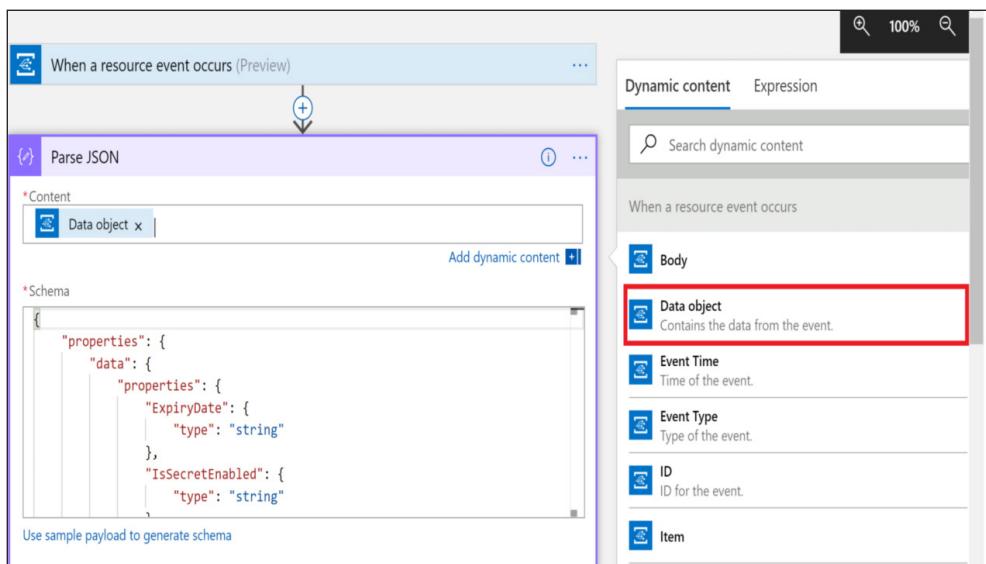
11. Clique em **Usar o conteúdo de amostra para gerar o esquema** e forneça os seguintes dados:

```
{  
    "ExpiryDate": "",  
    "SecretName": "",  
    "VaultName": "",  
    "SecretCreationDate": "",  
    "IsSecretEnabled": "",  
    "SecretId": ""  
}
```

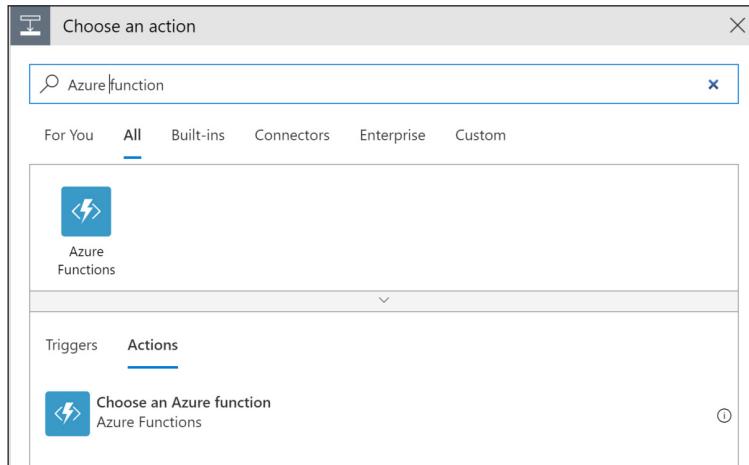
Pode surgir aqui uma pergunta a respeito do conteúdo de amostra. Como você sabe nessa fase qual é a carga gerada pelo editor da Grade de Eventos? A resposta para isso reside no fato de que esse conteúdo de amostra é exatamente o mesmo que é usado no elemento de dados no runbook de automação do Azure; você pode dar uma olhada no trecho de código novamente:

```
data = @{  
    "ExpiryDate" = $certificate.Expires  
    "CertificateName" = $certificate.Name.ToString()  
    "VaultName" = $certificate.VaultName.ToString()  
    "CertificateCreationDate" = $certificate.Created.ToString()  
    "IsCertificateEnabled" = $certificate.Enabled.ToString()  
    "CertificateId" = $certificate.Id.ToString()  
}
```

12. A caixa de texto **Conteúdo** deve conter conteúdo dinâmico vindo do gatilho anterior, conforme demonstrado na captura de tela a seguir:



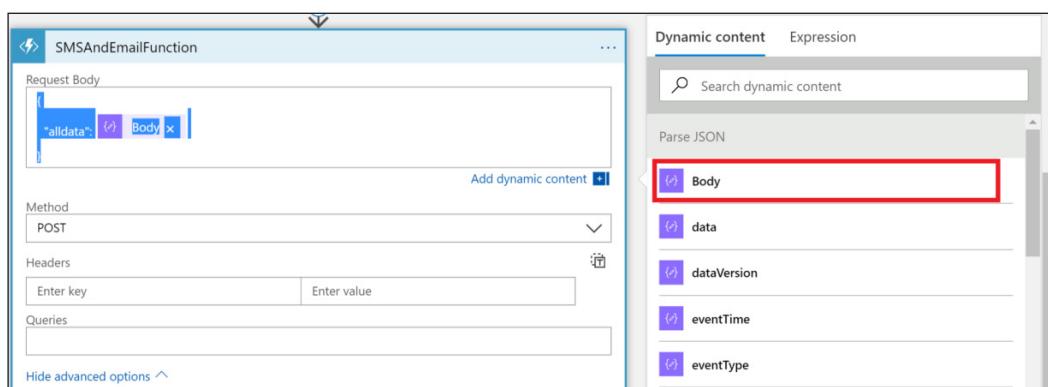
13. Adicione outra ação de **Azure Functions** depois de **Analisar JSON** e então selecione **Escolher uma função do Azure**. Selecione os aplicativos de função do Azure chamados **NotificationFunctionAppBook** e **SMSAndEmailFunction**, que foram criados anteriormente:



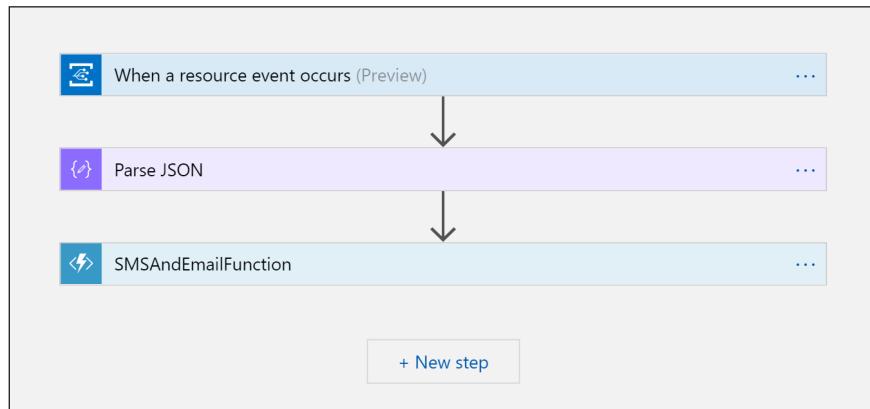
14. Clique na área de texto **Corpo da Solicitação** e preencha-a com a seguinte listagem de código. Isso é feito para converter os dados em JSON antes de enviá-los para a função do Azure:

```
{
  "alldata" :
}
```

15. Coloque o cursor depois de : no código anterior e clique em **Adicionar conteúdo dinâmico | Corpo da atividade anterior**:



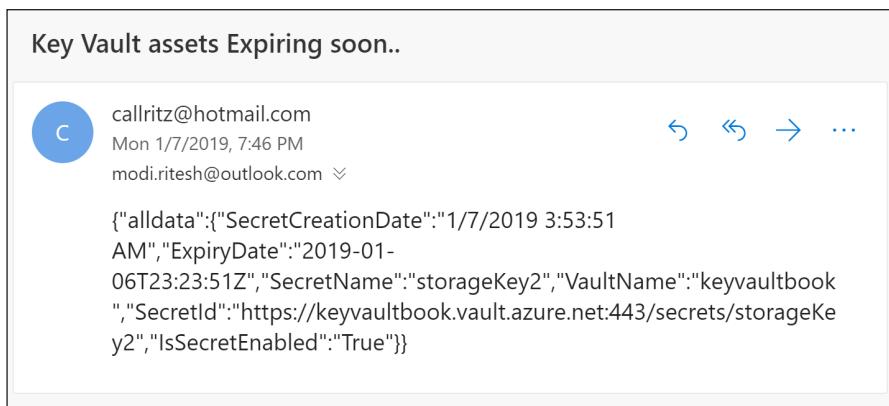
16. Salve o aplicativo lógico inteiro; o aplicativo lógico inteiro deve ter a seguinte aparência:



Testes

Faça upload de alguns segredos e certificados que tenham datas de expiração para o Azure Key Vault e execute o runbook da Automação do Azure. O runbook está agendado para ser executado por agendamento. Além disso, o runbook publicará eventos na Grade de Eventos. O aplicativo lógico deve ser habilitado e ele escolherá o evento e, por fim, invocará a função do Azure para enviar notificações por email e SMS.

O email deve ter a seguinte aparência:



Resumo

Este foi um capítulo grande – ele apresentou a Grade de Eventos ao fornecer algumas soluções de ponta a ponta usando recursos de armazenamento e tópicos personalizados. Ele também apresentou os Aplicativos Lógicos do Azure como fluxos de trabalho sem servidor automatizados. O capítulo concentrou-se principalmente na criação de uma arquitetura que integrou vários serviços do Azure para criar uma solução de ponta a ponta. Os serviços usados na solução foram Automação do Azure, Aplicativos Lógicos do Azure, Grades de Eventos do Azure, Azure Functions, SendGrid e Twilio. Esses serviços foram implementados por meio do portal do Azure e do PowerShell usando entidades de serviço como contas de serviço. Ele também mostrou uma série de maneiras para criar entidades de serviço com autenticação de senha e certificado.

O próximo capítulo é um capítulo importante da perspectiva de gerenciamento de custos do Azure. É muito fácil usar novos recursos no Azure sem saber o custo real de executá-los. No próximo capítulo, forneceremos detalhes sobre o gerenciamento de custos.

8

Gerenciamento de custos

O principal motivo pelo qual as corporações estão migrando para a nuvem é a economia de custos. Não há um custo inicial para ter uma assinatura do Azure. O Azure oferece um mecanismo de pagamento **conforme o uso**, o que significa que o pagamento se baseia no consumo; o Azure mede o uso e fornece faturas mensais com base em seu consumo. Não há limite superior para o quanto você pode consumir – o Azure fornece recursos ilimitados e qualquer pessoa com acesso ao Azure pode criar quantos recursos quiser. É claro, nessas situações, é importante que as empresas mantenham um monitoramento sobre o consumo do Azure. Embora as empresas possam criar políticas para definir convenções e padrões organizacionais, também existe a necessidade de ter as informações de cobrança e consumo do Azure prontamente disponíveis. Além disso, as empresas devem procurar aplicar as práticas recomendadas de consumo dos recursos do Azure para que os retornos sejam maximizados. Para isso, os arquitetos precisam estar plenamente cientes dos recursos do Azure, seus custos correspondentes e realizar comparações de custo-benefício.

Neste capítulo, abordaremos os seguintes tópicos:

- Noções básicas de cobrança do Azure
- Faturamento
- Clientes do Enterprise Agreement
- Uso e cotas
- Provedores de recursos
- APIs de uso e cobrança
- Os modelos de preços e a calculadora do Azure
- Práticas recomendadas

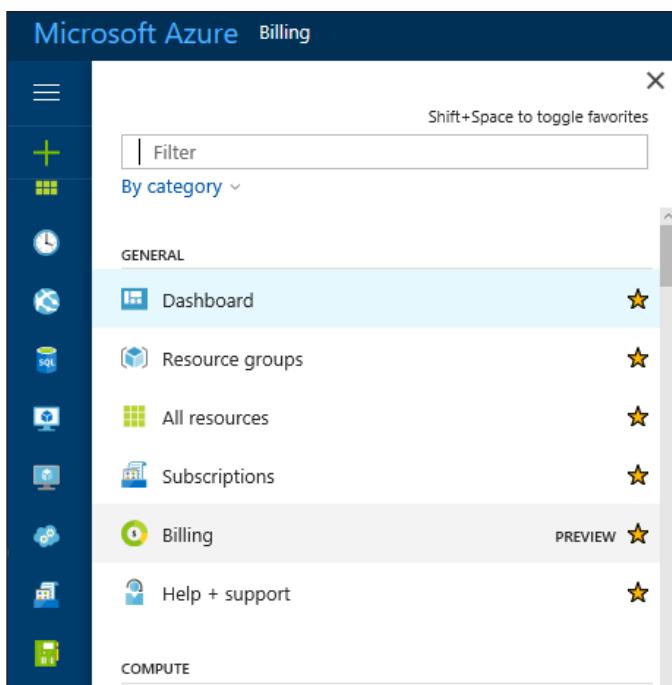
Noções básicas de cobrança

O Azure é um utilitário de serviço que oferece as seguintes vantagens:

- Sem custos iniciais
- Sem taxas de cancelamento
- Cobrança por minuto
- Pagamento com base no consumo

Nessas circunstâncias, é muito difícil estimar o custo inicial de consumo dos recursos do Azure. Cada recurso do Azure tem seu próprio modelo de custo e cobrança baseado em armazenamento, uso e intervalo de tempo. É muito importante que a gerência, a administração e o departamento financeiro de uma empresa acompanhem o uso e os custos. O Azure fornece os relatórios de cobrança e uso necessários para que a gerência e os administradores de uma organização possam gerar um relatório de custo e uso com base em todos os tipos de critério.

O portal do Azure fornece informações detalhadas sobre a cobrança e o uso por meio do recurso **Cobrança**, que pode ser acessado por meio da folha de navegação mestre:



Ele fornece um submenu para a geração de relatórios sobre custos e cobranças:

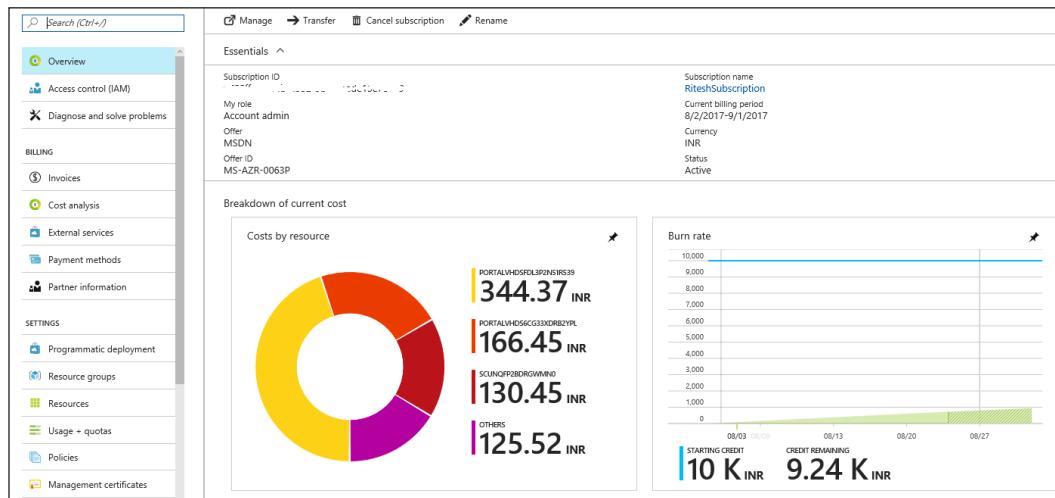
The screenshot shows the 'Billing' interface. On the left, a sidebar lists navigation options: Overview (selected), Diagnose and solve problems, Subscriptions (selected), Invoices, Contact info, Billing address, and Payment methods. Below these are sections for SUPPORT + TROUBLESHOOTING and New support request. The main content area has tabs for 'New subscription' and 'Manage'. Under 'Essentials', it shows 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 section titled 'Recent billing history' is present, and a note at the bottom states 'Your bill amount was 0 INR.'

Ao clicar no menu **Assinaturas** nesta folha, você verá uma lista de todas as assinaturas a que o usuário tem acesso para gerar relatórios:

The screenshot shows the 'Billing - Subscriptions' page. The sidebar is identical to the previous one. The main content area has tabs for 'New subscription' and 'Manage'. Under 'My subscriptions', there is a table with columns NAME, SUBSCRIPTION ID, OFFER, and STATUS. One row is shown: RiteshSubscription, with a long subscription ID, MSDN as the offer, and Active status. Under 'Other subscriptions', it says 'You don't have any subscriptions with billing read access'.

Gerenciamento de custos

Ao clicar no nome da assinatura nessa lista, você verá um painel no qual poderá encontrar informações completas sobre cobrança, fatura, uso, consumo, políticas, grupos de recursos e recursos. O gráfico nesta seção **Visão geral** fornece uma percentagem de custo por recurso e também uma taxa de gravação:



Ao clicar no gráfico, você verá os detalhes de custos com base em cada recurso. Aqui, há várias contas de Armazenamento provisionadas no Azure, e é mostrado o custo para cada uma delas. Usando essa tela, muitos tipos de relatório podem ser gerados ao fornecer diferentes critérios, como qualquer combinação do seguinte:

- Tipos de recursos
- Grupo de recursos
- Intervalo de tempo
- Marcas

As marcas são particularmente interessantes. As consultas com base em marcas, como departamento, projeto, proprietário, centro de custos ou qualquer outro par nome-valor, podem ser usadas para exibir as informações sobre custos. Você também pode fazer download do relatório de custos como um arquivo CSV ao usar o botão **Download**:

NAME	TYPE	RESOURCE GROUP	COST (INR)	TAGS
portalvhdsfdl3p2n51\$539	Storage account (classic)	Default-Storage-WestUS	344.37	...
portalvhds6cg33xdrh2ypl	Storage account (classic)	Default-Storage-EastUS	166.45	...
scunqlp2bdrgwmn0	Storage account	abra	130.45	...
scunqlp2bdrgwtx	Storage account	abra	64.74	...
portalvhdsn0bxtwdlj38yn	Storage account (classic)	Default-Storage-EastAsia	60.79	...

Ao clicar em um recurso individual, você verá os dados do consumo de custo diário do recurso:

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

Faturamento

O recurso Cobrança do Azure também fornece informações sobre faturas mensais.

Ao clicar no menu **Faturas**, você verá uma lista de todas as faturas geradas:

The screenshot shows the Azure Billing Invoices page. On the left, there's a sidebar with links for Overview, Access control (IAM), Diagnose and solve problems, and Billing (Invoices, Cost analysis, External services, Payment methods, Partner information). 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 "RiteshSubscription" selected. Under "Grid" view, there's a search bar and a table with the following data:

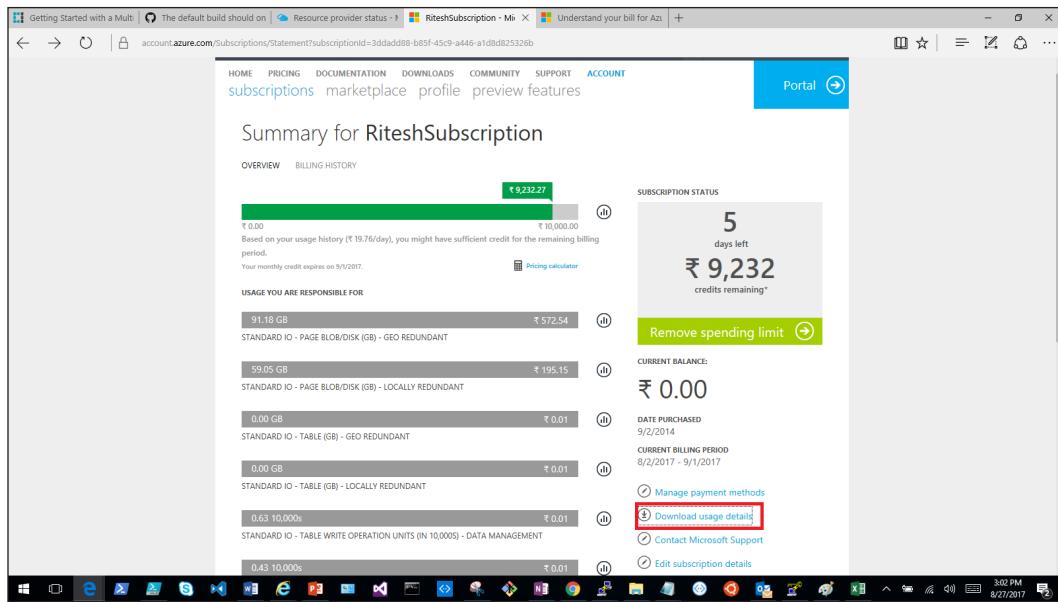
BILLING PERIOD	CHARGE DATE	AMOUNT (INR)	INVOICE
7/2/2017-8/1/2017	8/2/2017	₹ 0.00	Not available
6/2/2017-7/1/2017	7/2/2017	₹ 0.00	Not available
5/2/2017-6/1/2017	6/2/2017	₹ 0.00	Not available
4/2/2017-5/1/2017	5/2/2017	₹ 0.00	Not available

Ao clicar em qualquer uma das faturas, você verá os detalhes sobre essa fatura:

The screenshot shows the Azure Costs by service page. It includes a note about excluding non-Microsoft services and a search bar. The main table has columns for Name, Resource GUID, Consumed Units, Billable Units, and Pre-Tax Cost (INR). The data is as follows:

NAME	RESOURCE GUID	CONSUMED UNITS	BILLABLE UNITS	PRE-TAX COST (INR)
Standard IO - Page Blob/Disk (GB) - Geo Redundant	0e9d03b-abbd-4312-9c7e-3794e22af9c4	107,966	107,966	0.00
Standard IO - Page Blob/Disk (GB) - Locally Redundant	d23a5753-f8f5-4df-df28-8cc5cf3d3882	82,764	82,764	0.00
Standard IO - Table (GB) - Geo Redundant	92567832-8613-47c4-9e8a-b2b9461a4875	0.001	0.001	0.00
Standard IO - Table (GB) - Locally Redundant	3f2b1e1c- c886-4ec6-ad6f-d00ef38819c9	0.004	0.004	0.00
Standard IO - Table Write Operation Units (in 10,000s) - Data Management	9cb0bde8-bc0d-46bc-8423-a25fe06779d3	0.878	0.878	0.00
Standard IO - Table Write Operation Units (in 10,000s) - Geo Redundant	ad22fac8-9da5-4577-8683-56ae94d39e42	0.512	0.512	0.00

Há também uma maneira alternativa de fazer download de detalhes da fatura. Os detalhes da fatura estarão disponíveis quando você fizer logon em <https://account.azure.com> e fizer download deles:



Clientes do Enterprise Agreement

Os clientes corporativos com um Enterprise Agreement podem utilizar <https://ea.azure.com> para fazer download dos relatórios de uso e cobrança. Além disso, também foi lançado recentemente um novo pacote de conteúdo do Power BI lançado recentemente que pode ser utilizado para exibir o uso e os custos do Azure por meio de relatórios e de um painel no Power BI.

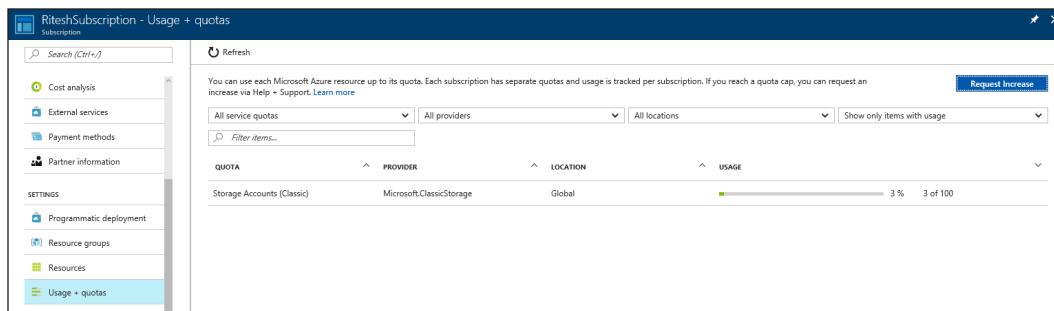


Mais informações sobre o uso e os custos dos contratos corporativos estão disponíveis em <https://azure.microsoft.com/pt-br/blog/new-power-bi-content-pack-for-azure-enterprise-users/>.

Uso e cotas

Cada assinatura possui um limite de cota para cada tipo de recurso. Por exemplo, pode haver no máximo 60 endereços IP públicos provisionados com uma conta Microsoft MSDN. Da mesma forma, todos os recursos têm um limite máximo padrão para cada tipo de recurso. Estes números de tipos de recursos para uma assinatura podem ser aumentados ao contatar o suporte do Azure ou ao clicar no botão **Solicitar aumento**.

As informações sobre o uso e as cotas estão disponíveis no submenu **Uso + cotas** do menu **Assinatura**:



The screenshot shows the 'Usage + quotas' page in the Azure portal. The left sidebar has a 'Subscription' section with 'Cost analysis', 'External services', 'Payment methods', 'Partner information', 'SETTINGS', 'Programmatic deployment', 'Resource groups', 'Resources', and 'Usage + quotas' selected. The main area has a 'Refresh' button and a note about quota caps. It includes filters for 'All service quotas', 'All providers', 'All locations', and 'Show only items with usage'. Below is a table with columns: QUOTA, PROVIDER, LOCATION, and USAGE. One row is shown: 'Storage Accounts (Classic)' under 'Microsoft.ClassicStorage' with a 'Global' location and 3% usage, 3 of 100 total.

Esta folha mostra todos os tipos de recursos provisionados em uma assinatura juntamente com as localizações e contagens deles. Aqui, a cota para contas de Armazenamento clássicas é 100 e, no momento, foram consumidas três contas de Armazenamento clássicas.

Você pode filtrar por local, provedor, uso e cota. Podem ser gerados relatórios personalizados com base nesses critérios de filtragem.

Provedores de recursos

Os recursos são baseados em tipos de recurso e estão disponíveis por meio de provedores de recursos. Há inúmeros provedores disponíveis no Azure, que fornecem os tipos de recursos necessários para que os usuários criem suas instâncias. Por exemplo, o provedor de recursos **Microsoft.Compute** fornece tipos de recurso de máquinas virtuais. Com os tipos de recurso de máquinas virtuais, podem ser criadas instâncias de máquina virtual.

É necessário que os provedores de recursos sejam registrados com assinaturas do Azure. Os tipos de recursos não estarão disponíveis em uma assinatura se os provedores de recursos não estiverem registrados. Para obter uma lista dos provedores disponíveis, ou para ver aqueles que estão registrados e os que não estão, ou até mesmo para registrar os provedores não registrados, use este painel:

PROVIDER	STATUS	
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

As APIs de uso e cobrança

Embora o portal seja uma excelente maneira de encontrar informações sobre o uso, a cobrança e a fatura manualmente, o Azure também fornece o seguinte:

- **A API de download de faturas:** use essa API para fazer download de faturas.
- **A API de uso de recursos:** use essa API para obter a estimativa de dados de consumo do Azure.
- **A API RateCard:** use essa API para obter uma lista de recursos disponíveis do Azure e informações sobre estimativas de preços para cada um deles.

Essas APIs podem ser usadas para recuperar detalhes de forma programática e criar relatórios e painéis personalizados. Qualquer linguagem de programação ou script pode usar essas APIs e criar uma solução completa de cobrança.

Modelos de preços do Azure

O Azure tem vários modelos de preços. Ele tem um modelo para todos os tipos de cliente. Isso inclui desde contas gratuitas para estudantes e contas pré-pagas para desenvolvedores até modelos de contratos Enterprise Agreement e do parceiro provedor de soluções na nuvem. Além desses tipos de contas, há preços e descontos complementares, como instâncias de VM reservadas e o Benefício Híbrido do Azure.

Benefício Híbrido do Azure

Quando uma máquina virtual é provisionada no Azure, há dois tipos de custo envolvidos. Esses dois custos são o custo do recurso para execução da máquina virtual e o custo de licença do sistema operacional. Embora os clientes do Enterprise Agreement obtenham alguns descontos em comparação a outras contas em relação aos preços, o Azure fornece outra oferta para eles: o Benefício Híbrido do Azure. Neste esquema, os clientes existentes do Enterprise Agreement podem usar suas licenças na infraestrutura local para o sistema operacional a fim de criar suas máquinas virtuais no Azure, e o Azure não cobrará pelo custo da licença. As economias de custos podem chegar a 40% do custo original ao usar esse esquema. Os clientes do Enterprise Agreement também devem ter o Software Assurance para aproveitar esse benefício, que é aplicável às edições Windows Standard e Datacenter. Cada licença de dois processadores ou cada conjunto de licenças de 16 núcleos tem direito a duas instâncias de até 8 núcleos ou uma instância de até 16 núcleos. O Benefício Híbrido do Azure para as licenças da edição Standard só pode ser usado em uma instância, na infraestrutura local ou no Azure. A edição Datacenter permite a utilização simultânea na infraestrutura local e no Azure.

Instâncias de máquina virtual reservadas do Azure

Os clientes podem reservar um número fixo de máquinas virtuais antecipadamente por um a três anos para os sistemas operacionais Windows e Linux. O Azure fornece até 72% de desconto nessas máquinas virtuais com base em um modelo de preços pré-pago. Embora haja um compromisso inicial, não há nenhuma obrigação de usar as instâncias. Essas instâncias reservadas podem ser canceladas a qualquer momento. Essa oferta pode até ser combinada com o esquema do Benefício Híbrido do Azure para reduzir ainda mais o custo de licença dessas máquinas virtuais.

Contas pré-pagas

Estas são contas gerais do Azure e são cobradas mensalmente aos clientes. Os clientes não se comprometem com nenhum uso e podem usar qualquer recurso com base em suas necessidades. Os custos dos recursos são calculados com base no uso e no tempo de atividade. Entretanto, cada recurso tem seu próprio modelo de custo. Também não há custos iniciais associados a essas contas. Geralmente, não há descontos disponíveis nesse esquema.

Enterprise Agreements

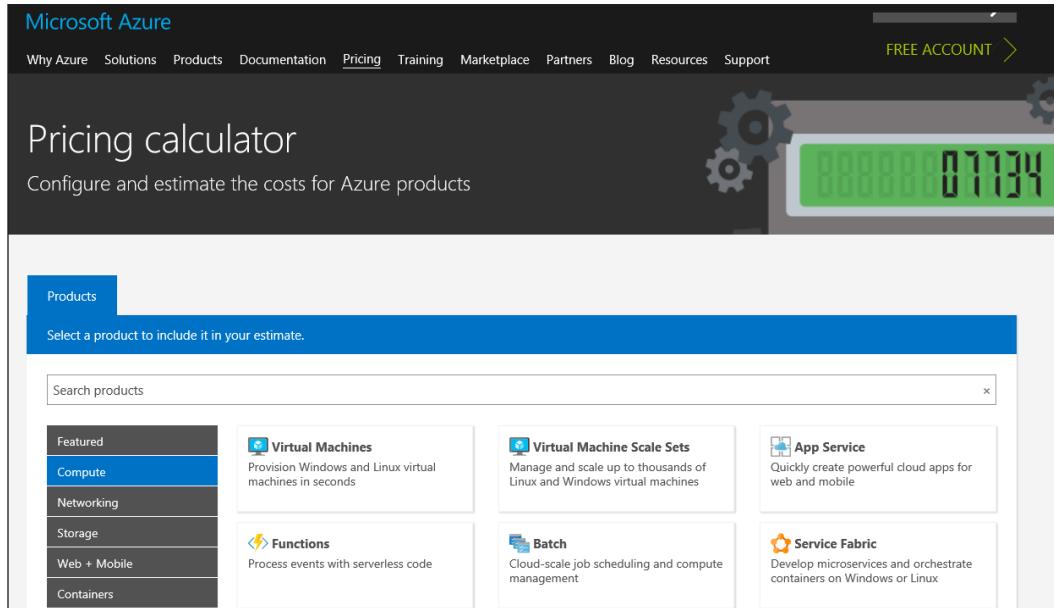
Os clientes que já têm contratos com a Microsoft podem adicionar seus locatários do Azure como parte do Enterprise Agreement. Os clientes poderão aproveitar excelentes descontos se fizerem parte de Enterprise Agreements. Eles só precisam criar antecipadamente o compromisso monetário anual e poderão ser adicionados a esse esquema. Os clientes são livres para consumir o que quiserem. Consulte <https://azure.microsoft.com/en-in/pricing/enterprise-agreement/> para obter mais informações.

O modelo de Provedor de Soluções na Nuvem

O **Provedor de Soluções na Nuvem** (CSP) é um modelo para parceiros da Microsoft. O CSP permite que os parceiros tenham total propriedade do ciclo de vida e do relacionamento com o cliente para o Microsoft Azure. Os parceiros podem implantar suas soluções na nuvem e cobrar os clientes usando esse esquema. Consulte <https://azure.microsoft.com/en-in/offers/ms-azr-0145p/> para obter mais informações.

A calculadora de preços do Azure

O Azure oferece uma calculadora de custos para que os usuários e clientes estimem seu custo e uso. Essa calculadora está disponível em <https://azure.microsoft.com/en-in/pricing/calculator/>:



Gerenciamento de custos

Os usuários podem selecionar vários recursos no menu à esquerda, os quais serão adicionados à calculadora. No exemplo a seguir, uma máquina virtual é adicionada. Uma configuração adicional em relação a região, sistema operacional, tipo, camada, tamanho de instância, número de horas e número de máquinas virtuais pode ser feita:

Your Estimate

Virtual Machines [] [] 1: D1: 1 cores, 3.5 GB RAM, 50 GB disk

Virtual Machines

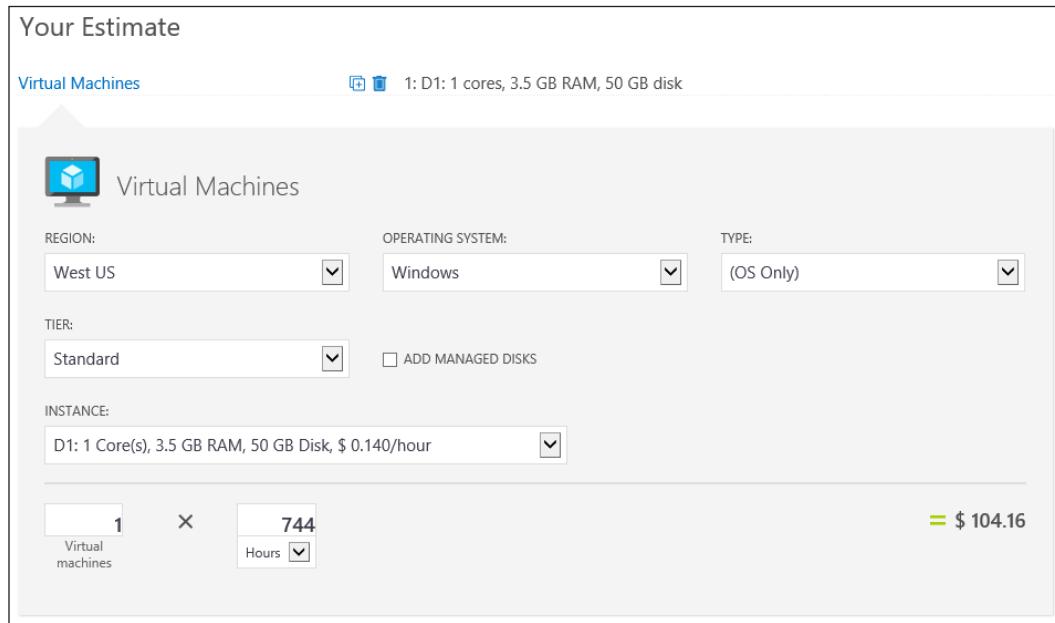
REGION: West US OPERATING SYSTEM: Windows TYPE: (OS Only)

TIER: Standard ADD MANAGED DISKS

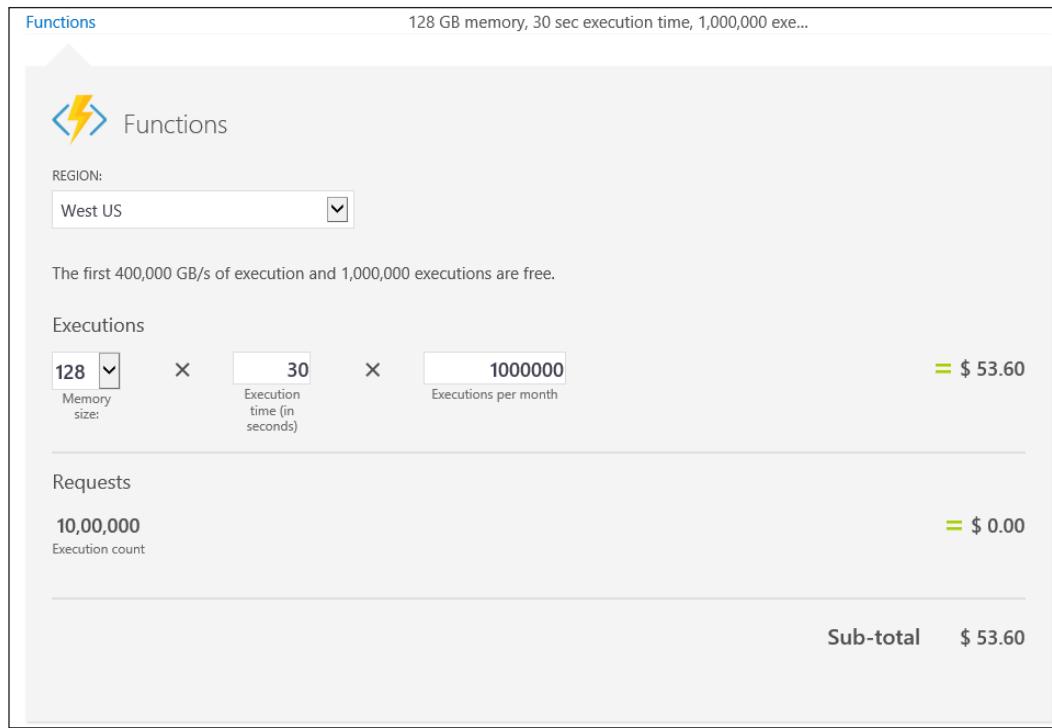
INSTANCE: D1: 1 Core(s), 3.5 GB RAM, 50 GB Disk, \$ 0.140/hour

1 **744** **\$ 104.16**

Virtual machines Hours



Da mesma forma, o custo de funções do Azure para tamanho de memória, tempo de execução e execução por segundo das máquinas virtuais é mostrado a seguir:



O Azure fornece diferentes níveis e planos de suporte, que são os seguintes:

- **Suporte padrão:** gratuito
- **Suporte ao desenvolvedor:** US\$ 29 por mês
- **Suporte padrão:** US\$ 300 por mês
- **Professional direct:** US\$ 1.000 por mês

Por fim, o custo total estimado é exibido:

The screenshot shows the Azure Cost Management interface. At the top, there's a section for 'Support' with a dropdown menu set to 'Developer' which costs '\$ 29.00'. Below that is a section for 'Programs and Offers' with a dropdown menu set to 'Microsoft Online Services Program (MOSP)'. There's also a checkbox for 'SHOW DEV/TEST PRICING'. Underneath these sections, the 'Estimated monthly cost' is displayed as '\$ 186.76' in US Dollar (\$). A blue 'Export' button is located at the bottom left of this section.

É importante que os arquitetos compreendam como cada recurso do Azure é usado na arquitetura e solução em geral. A precisão da calculadora do Azure depende de quais recursos foram selecionados e da sua configuração. Qualquer informação incorreta levaria a estimativas tendenciosas e erradas, que seriam diferentes da cobrança real.

Práticas recomendadas

Os arquitetos precisam se esforçar ainda mais para compreender sua arquitetura e os componentes do Azure utilizados. Com base no monitoramento ativo, auditorias e uso, eles devem estar plenamente cientes do SKU, do tamanho e dos recursos da arquitetura. Esta seção detalhará algumas das práticas recomendadas a serem adotadas de uma perspectiva de otimização de custos.

Práticas recomendadas de computação

A computação refere-se a serviços que ajudam na execução de serviços. Algumas das práticas recomendadas em relação à computação são as seguintes:

- Escolha o melhor local para seus serviços de computação, como as máquinas virtuais. Escolha um local onde todos os recursos e funcionalidades do Azure estejam disponíveis juntos, na mesma região. Isso evitará o tráfego de saída.
- Escolha o tamanho ideal para suas máquinas virtuais. Uma máquina virtual maior tem um custo mais alto do que uma de um tamanho menor, e talvez uma máquina virtual de tamanho maior não seja realmente necessária.

- Redimensione máquinas virtuais de acordo com a demanda. O Azure libera novos tamanhos de máquina virtual com frequência. Se um novo tamanho mais adequado às suas necessidades for disponibilizado, ele deverá ser usado.
- Desligue os serviços de computação quando não forem necessários. Isso aplica-se particularmente aos ambientes de não produção.
- Desaloque máquinas virtuais em vez de desligá-las. Isso liberará todos os recursos e o consumo será interrompido.
- Use laboratórios de desenvolvimento/testes para fins de desenvolvimento e testes. Eles fornecem políticas e recursos de desligamento automático e de inicialização automática.
- Com conjuntos de dimensionamento de máquina virtual, provisione algumas máquinas virtuais e aumente sua contagem com base na demanda.
- Escolha o tamanho correto (pequeno, médio ou grande) para os gateways de aplicativos. Eles têm o suporte de máquinas virtuais e podem ajudar a reduzir custos se forem dimensionadas de forma ideal. Além disso, escolha o gateway de aplicação da camada básica se um firewall do aplicativo Web não for necessário.
- Escolha as camadas corretas para gateways de rede privada virtual (incluindo uma rede virtual privada básica, padrão, de alto desempenho e ultra desempenho).
- Minimize o tráfego de rede entre regiões do Azure com a colocação de recursos na mesma região.
- Use um平衡ador de carga com um IP público para acessar várias máquinas virtuais em vez de atribuir um IP público a cada máquina virtual.
- Monitore as máquinas virtuais e suas métricas de uso e performance. Com base nessas métricas, determine se deseja escalar a máquina virtual de forma horizontal ou vertical. A consulta das métricas também pode resultar em downsizing das máquinas virtuais.

Práticas recomendadas de armazenamento

Aqui estão algumas práticas recomendadas para otimizar o armazenamento para custo:

- Escolha o tipo de redundância de armazenamento adequado (GRS, LRS ou RA-GRS). O GRS é mais caro do que o LRS, por exemplo.
- Arquive dados de armazenamento para acessar com menos frequência ou arquivar a camada de acesso. Mantenha os dados acessados com frequência na camada de acesso frequente.
- Remova blobs que não são necessários.
- Exclua explicitamente os discos do sistema operacional da máquina virtual após excluir a máquina virtual, caso eles não sejam necessários.

- As contas de armazenamento são medidas com base em suas operações de tamanho, gravação, leitura, listagem e contêiner.
- Prefira os discos standard em vez de discos premium. Use discos premium somente se forem uma exigência da empresa.
- Use a **Rede de Distribuição de Conteúdo (CDN)** e o armazenamento em cache para arquivos estáticos em vez de obtê-los toda vez no armazenamento.

Práticas recomendadas de PaaS (Plataforma como Serviço)

Veja a seguir algumas das práticas recomendadas se PaaS for o modelo de implantação de preferência:

- Escolha a camada adequada (básica, standard, premium RS ou premium) e os níveis de performance adequados do SQL do Azure.
- Escolha adequadamente entre os bancos de dados individuais e elásticos. Se houver muitos bancos de dados, será econômico usar bancos de dados elásticos em comparação com bancos de dados individuais.
- Garanta a segurança do SQL do Azure – criptografar dados em repouso e em movimento, mascaramento de dados, proteção contra ameaças estão habilitados.
- Verifique se a estratégia de backup e a replicação de dados estão configuradas de acordo com as demandas comerciais.
- Verifique se há redundância para aplicativos Web com disponibilidade de várias regiões usando o gerenciador de tráfego.
- Use o cache Redis e a CDN para uma entrega mais rápida de dados e páginas.
- Rearquitetando suas soluções para usar soluções de PaaS (como soluções sem servidor e microsserviços em contêineres) em vez de soluções de **Infraestrutura como Serviço (IaaS)**. Essas soluções PaaS removem os custos de manutenção e estão disponíveis com base no consumo por minuto. Se você não consumir esses serviços, não haverá nenhum custo, mesmo se seu código e seus serviços sempre estiverem disponíveis.

Práticas recomendadas gerais

Algumas das outras práticas recomendadas gerais estão listadas aqui:

- Os custos de recursos diferem entre regiões. Tente usar uma região com custos menores.
- Os Enterprise Agreements oferecem os melhores descontos. Se você não estiver em um Enterprise Agreement, tente usar um para ter os benefícios de custo.
- Se os custos do Azure puderem ser pré-pagos, serão oferecidos descontos para todos os tipos de assinatura.
- Exclua ou remova recursos não utilizados. Descubra os recursos que são subutilizados e reduza seu SKU ou tamanho. Se eles forem desnecessários, exclua-os.
- Use o Assistente do Azure e leve suas recomendações a sério.

Resumo

A administração e o gerenciamento de custos são atividades importantes ao lidar com a nuvem. Isso se deve principalmente ao fato de que a despesa mensal pode ser muito baixa, mas pode ser muito alta sem a devida atenção. Os arquitetos devem projetar seus aplicativos de forma que o custo seja minimizado o máximo possível. Eles devem usar os recursos do Azure, o SKU, a camada e o tamanho adequados e devem saber quando iniciar, parar, escalar verticalmente e horizontalmente, transferir dados e muito mais. O gerenciamento de custos adequado garantirá que as despesas reais correspondam às despesas orçamentárias.

O próximo e último capítulo deste livro aborda os recursos de monitoramento e auditoria do Azure.

9

Criação de políticas, bloqueios e marcas

O Azure é uma plataforma de nuvem versátil. Os clientes podem não apenas criar e implantar seus aplicativos, mas também gerenciar e governar seus ambientes ativamente. As nuvens geralmente seguem um paradigma pré-pago no qual um cliente assina e pode implantar praticamente qualquer coisa nelas. Desde uma máquina virtual básica até milhares de máquinas virtuais com SKUs maiores. O Azure não impedirá que nenhum cliente provisione os recursos que deseja provisionar. Em uma organização, pode haver um grande número de pessoas com acesso à assinatura do Azure da organização. Há uma necessidade de haver um modelo de governança em vigor para que somente os recursos necessários sejam provisionados por pessoas com o direito de criá-los. O Azure fornece funções de gerenciamento de recursos, como o **Controle de Acesso Baseado em Função (RBAC)** do Azure, além de políticas e bloqueios para gerenciar e fornecer governança para os recursos.

Outro aspecto importante da governança é o gerenciamento de custos, uso e informações. A gerência de uma organização sempre quer se manter atualizada sobre seu consumo e custos de nuvem. Ela quer identificar a porcentagem de uso do custo total por parte de equipes, departamentos e unidades. Em suma, ela quer ter relatórios com base em várias dimensões sobre consumo e custo. O Azure fornece um recurso de marcação que pode ajudar a fornecer esse tipo de informação em tempo real.

Neste capítulo, abordaremos os seguintes tópicos:

- Marcas do Azure
- Políticas do Azure
- Bloqueios do Azure
- Azure RBAC
- Implementação de recursos de governança do Azure

Marcas do Azure

Uma marca é definida pelo Dicionário Oxford (<https://en.oxforddictionaries.com/definition/tag>) desta forma:

"um rótulo anexado a algo ou alguém para fins de identificação ou para fornecer outras informações".

O Azure permite a marcação de recursos e grupos de recursos com pares nome-valor. A marcação ajuda na categorização e organização lógica dos recursos. O Azure também permite a marcação de 15 pares nome-valor para um grupo de recursos e seus recursos. Embora um grupo de recursos seja um contêiner de recursos, sua marcação não é sinônimo da marcação dos recursos contidos nele. Os recursos e os grupos de recursos devem ser marcados com base no seu uso, o que será explicado posteriormente nesta seção. As marcas funcionam no nível da assinatura. O Azure aceita quaisquer pares nome-valor e, por isso, é importante para uma organização definir os nomes e seus possíveis valores.

Mas por que a marcação é importante? Em outras palavras, quais problemas podem ser resolvidos usando a marcação? A marcação tem os seguintes benefícios:

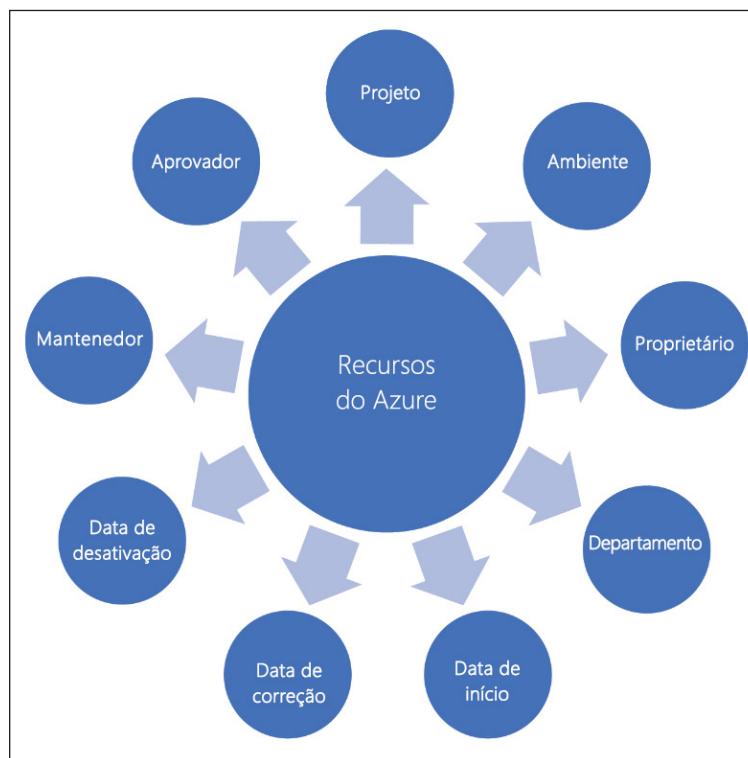
- **Categorização de recursos:** uma assinatura do Azure pode ser usada por vários departamentos e funções dentro de uma organização. É importante para a equipe de gerência identificar os proprietários de quaisquer recursos. A marcação ajuda na atribuição de identificadores a recursos que possam ser usados representar departamentos ou funções.
- **Gerenciamento de informações para recursos do Azure:** novamente, os recursos do Azure em uma assinatura podem ser provisionados por qualquer pessoa que tenha acesso a ela. As organizações desejam ter uma categorização adequada dos recursos em vigor para atender às políticas de gerenciamento de informações. Tais políticas podem ser baseadas no gerenciamento do ciclo de vida do aplicativo, como o gerenciamento dos ambientes de desenvolvimento, teste e produção. Essas políticas podem basear-se no uso ou em outras prioridades. Cada organização tem sua própria maneira de definir as categorias de informações e o Azure supre isso com as marcas.
- **Gerenciamento de custos:** a marcação no Azure pode ajudar a identificar recursos com base na sua categorização. As consultas podem ser executadas no Azure para identificar o custo por categoria, por exemplo. Por exemplo, o custo dos recursos no Azure para o desenvolvimento de um ambiente para o departamento financeiro e o departamento de marketing pode ser facilmente verificado. Além disso, o Azure também fornece informações de cobrança com base em marcas. Isso ajuda a identificar as taxas de consumo de equipes, departamentos ou grupos.

As marcas no Azure têm certas limitações, no entanto:

- O Azure permite que um máximo de 15 pares nome-valor de marca sejam associados a grupos de recursos.
- Marcas não são herdáveis. As marcas aplicadas a um grupo de recursos não se aplicam aos recursos individuais dentro dele. No entanto, é muito fácil esquecer de marcar os recursos ao provisioná-los. As políticas do Azure são o mecanismo para garantir o uso de marcas com o valor apropriado durante o tempo de provisionamento. Consideraremos os detalhes de tais políticas mais adiante neste capítulo.

As marcas podem ser atribuídas a recursos e grupos de recursos usando PowerShell, CLI do Azure 2.0, modelos do Azure Resource Manager, o portal do Azure e as APIs REST do Azure Resource Manager.

Um exemplo de categorização de gerenciamento de informações usando marcas do Azure é mostrado aqui. Neste exemplo, os pares nome-valor de **Departamento**, **Projeto**, **Ambiente**, **Proprietário**, **Aprovador**, **Mantenedor**, **Data de Início**, **Data de Desativação** e **Data de Correção** são usados para marcar recursos. É extremamente fácil encontrar todos os recursos para uma marca específica ou uma combinação de marcas usando o PowerShell, a CLI do Azure ou as APIs REST:



Marcas com PowerShell

As marcas podem ser gerenciadas usando o PowerShell, os modelos do Azure Resource Manager, o portal do Azure e as APIs REST. Nesta seção, o PowerShell será usado para criar e aplicar marcas. O PowerShell fornece um cmdlet para recuperar e anexar marcas a recursos e grupos de recursos:

- Para recuperar marcas associadas a um recurso usando o PowerShell, o cmdlet `Find-AzureRMResource` pode ser usado:

```
(Find-AzureRmResource -TagName Dept -TagValue Finance).Name
```

- Para recuperar marcas associadas a um grupo de recursos usando o PowerShell, o seguinte comando pode ser usado:

```
(Find-AzureRmResourceGroup -Tag @{ Dept="Finance" }).Name
```

- Para definir marcas para um grupo de recursos, o cmdlet `Set-AzureRmResourceGroup` pode ser usado:

```
Set-AzureRmResourceGroup -Name examplegroup -Tag @{ Dept="IT"; Environment="Test" }
```

- Para definir marcas para um recurso, o cmdlet `Set-AzureRmResource` pode ser usado:

```
Set-AzureRmResource -Tag @{ Dept="IT"; Environment="Test" }  
-ResourceName examplevnet -ResourceGroupName examplegroup
```

Marcas com modelos do Azure Resource Manager

Os modelos do Azure Resource Manager também fornece ajuda na definição de marcas para cada recurso. Eles podem ser usados para atribuir várias marcas a cada recurso, desta forma:

```
{
    "$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": {
            }
        }
    ]
}
```

No exemplo anterior, duas marcas, Dept e Environment, são adicionados a um recurso de conta de Armazenamento usando modelos do Azure Resource Manager.

Grupos de recursos x recursos

É imprescindível que arquitetos decidam a arquitetura de informações e taxonomia para recursos e grupos de recursos do Azure. Eles devem identificar as categorias pelas quais os recursos serão classificados com base nos requisitos da consulta. No entanto, eles também devem identificar se as marcas devem ser anexadas a recursos individuais ou grupos de recursos.

Se todos os recursos dentro de um grupo de recursos precisarem da mesma marca, então será melhor marcar o grupo de recursos, em vez de marcar cada recurso. É importante considerar as consultas nas marcas antes de decidir se as marcas devem ser aplicadas no nível do recurso ou do grupo de recursos. Se as consultas se relacionarem a tipos de recursos individuais em uma assinatura e em grupos de recursos, a atribuição de marcas a recursos individuais fará mais sentido. No entanto, se a identificação de grupos de recursos for suficiente para que suas consultas sejam efetivas, as marcas deverão ser aplicadas somente aos grupos de recursos.

Políticas do Azure

Na seção anterior, falamos sobre a aplicação de marcas para implantações do Azure. As marcas são ótimas para organizar recursos; no entanto, há algo mais que não foi discutido: como as organizações garantem que as marcas sejam aplicadas para cada implantação? Deve haver uma imposição automática das marcas do Azure a recursos e grupos de recursos. Não há nenhuma verificação do Azure para garantir que marcas apropriadas sejam aplicadas a recursos e grupos de recursos. Agora, isso não é apenas específico das marcas – se aplica à configuração de qualquer recurso no Azure. Por exemplo, você pode querer restringir onde seus recursos podem ser provisionados geograficamente (somente para a região leste dos EUA, por exemplo).

Talvez você já tenha adivinhado que esta seção é sobre a formulação de um modelo de governança no Azure. A governança é um elemento importante no Azure porque ajuda a manter os custos sob controle. Ela também garante que todos que acessarem o ambiente do Azure estejam cientes dos processos e das prioridades organizacionais. Ela ajuda na definição das convenções organizacionais para o gerenciamento de recursos.

Cada política pode ser criada usando várias regras, e várias políticas podem ser aplicadas a uma assinatura ou a um grupo de recursos. Se as regras forem satisfeitas, as políticas poderão executar várias ações. Uma ação poderia ser negar uma transação em andamento, auditar (uma transação o que significa gravar em logs e permitir sua conclusão) ou acrescentar metadados a uma transação se eles estiverem ausentes.

As políticas podem estar relacionadas à convenção de nomenclatura de recursos, marcação de recursos, tipos de recursos que podem ser provisionados, localização de recursos ou qualquer combinação dessas regras.

Políticas internas

O Azure fornece infraestrutura para criar políticas personalizadas; no entanto, ele também fornece algumas políticas prontas para uso que podem ser usadas para a governança. Essas políticas estão relacionadas a localizações permitidas, tipos de recursos permitidos e marcas. Mais informações sobre essas políticas internas podem ser encontradas em <https://docs.microsoft.com/en-us/azure/azure-resource-manager/resource-manager-policy>.

Linguagem de políticas

As políticas do Azure usam a linguagem JSON para definir e descrever as políticas.

Há duas etapas na adoção de políticas. A política deve ser definida e, em seguida, deve ser aplicada e atribuída. As políticas têm escopo e podem ser aplicadas no nível da assinatura e do grupo de recursos.

As políticas são definidas usando blocos `if...then`, que são semelhantes a qualquer linguagem de programação popular. O bloco `if` é executado para avaliar as condições e, com base no resultado dessas condições, o bloco `then` é executado:

```
{
  "if": {
    <condition> | <logical operator>
  },
  "then": {
    "effect": "deny | audit | append"
  }
}
```

As políticas do Azure não só permitem condições `if` simples, mas várias condições `if` podem ser associadas logicamente para criar regras complexas. Essas condições podem ser associadas usando operadores **AND**, **OR** e **NOT**:

- A sintaxe **AND** exige que todas as condições sejam verdadeiras.
- A sintaxe **OR** exige que uma das condições seja verdadeira.
- A sintaxe **NOT** inverte o resultado da condição.

A sintaxe AND é mostrada a seguir. Ela é representada pela palavra-chave `allOf`:

```
"if": {
  "allOf": [
    {
      "field": "tags",
      "containsKey": "application"
    },
    {
      "field": "type",
      "equals": "Microsoft.Storage/storageAccounts"
    }
  ]
},
```

A sintaxe OR é mostrada em seguida. Ela é representada pela palavra-chave `anyOf`:

```
"if": {
  "anyOf": [
    {
      "field": "tags",
      "containsKey": "application"
    },
    {
      "field": "type",
      "equals": "Microsoft.Storage/storageAccounts"
    }
  ]
},
```

A próxima é a sintaxe NOT. Ela é representada pela palavra-chave `not`:

```
"if": {
  "not": [
    {
      "field": "tags",
      "containsKey": "application"
    },
    {
      "field": "type",
      "equals": "Microsoft.Storage/storageAccounts"
    }
  ]
},
```

Na verdade, esses operadores lógicos podem ser combinados da seguinte maneira:

```
"if": {
  "allOf": [
    {
      "not": {
        "field": "tags",
        "containsKey": "application"
      }
    },
    {
      "field": "type",
      "equals": "Microsoft.Storage/storageAccounts"
    }
  ]
},
```

Isso é muito semelhante ao uso das condições `if` em linguagens de programação populares como C# e Node.js

```
If ("type" == "Microsoft.Storage/storageAccounts") {
  Deny
}
```

É importante observar que não há uma ação `allow`, embora haja uma ação `Deny`. Isto significa que as regras políticas devem ser escritas com a possibilidade de negação na mente. As regras devem avaliar as condições e retornar `Deny` se elas forem verdadeiras.

Campos permitidos

Os campos permitidos nas políticas são os seguintes:

- Nome
- Variante
- Tipo
- Local
- Marcas
- Marcas.*
- Aliases de propriedade

Bloqueios do Azure

Os bloqueios são mecanismos para interromper determinadas atividades nos recursos. O RBAC fornece direitos a usuários, grupos e aplicativos em um determinado escopo. Há funções RBAC prontas para uso, como proprietário, colaborador e leitor. Com a função de colaborador, é possível excluir ou modificar um recurso. Como tais atividades podem ser impedidas apesar do usuário ter a função de colaborador? Insira bloqueios do Azure.

Os bloqueios do Azure podem ajudar de duas maneiras:

- Eles podem bloquear recursos de modo que eles não possam ser excluídos, mesmo que você tenha acesso de proprietário.
- Eles podem bloquear recursos de forma que não possam ser excluídos nem ter sua configuração modificada.

Os bloqueios normalmente são muito úteis para os recursos em ambientes de produção que não devem ser modificados nem excluídos acidentalmente.

Os bloqueios podem ser aplicados nos níveis da assinatura, do grupo de recursos e do recurso individual. Os bloqueios podem ser herdados entre assinaturas, grupos de recursos e recursos. A aplicação de um bloqueio no nível pai garantirá que aqueles no nível filho também o herdarão. Mesmo os recursos que você adicionar mais tarde herdarão o bloqueio do pai. O bloqueio mais restritivo na herança tem precedência. A aplicação de um bloqueio no nível do recurso também não permitirá a exclusão do grupo de recursos que contenha o recurso.

Os bloqueios são aplicados somente a operações que ajudam no gerenciamento do recurso, em vez daquelas que são internas a ele. Os usuários precisam das permissões de RBAC `Microsoft.Authorization/*` ou `Microsoft.Authorization/locks/*` para criar e modificar bloqueios.

Os bloqueios podem ser criados e aplicados usando o portal do Azure, o Azure PowerShell, a CLI do Azure, os modelos do Azure Resource Manager e APIs REST.

A criação de um bloqueio usando um modelo do Azure Resource Manager é feita da seguinte maneira:

```
{  
    "$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"  
    }  
}  
]  
}
```

A criação e a aplicação de um bloqueio a um recurso usando o PowerShell são feitas como a seguir:

```
New-AzureRmResourceLock -LockLevel CanNotDelete -LockName LockSite '  
-ResourceName examplesite -ResourceType Microsoft.Web/sites '  
-ResourceGroupName exempleresourcegroup
```

A criação e a aplicação de um bloqueio a um grupo de recursos usando o PowerShell são feitas como a seguir:

```
New-AzureRmResourceLock -LockName LockGroup -LockLevel CanNotDelete '  
-ResourceGroupName exempleresourcegroup
```

A criação e a aplicação de um bloqueio a um recurso usando a CLI do Azure são feitas como a seguir:

```
az lock create --name LockSite --lock-type CanNotDelete \  
--resource-group exempleresourcegroup --resource-name examplesite \  
--resource-type Microsoft.Web/sites
```

A criação e a aplicação de um bloqueio a um grupo de recursos usando a CLI do Azure são feitas como a seguir:

```
az lock create --name LockGroup --lock-type CanNotDelete \  
--resource-group exempleresourcegroup
```

Azure RBAC

O Azure fornece autenticação para seus recursos por meio do Azure Active Directory. Uma vez que uma identidade tenha sido autenticada, os recursos a que a identidade deve ter acesso deverão ser decididos. Isso é conhecido como autorização. A autorização avalia as permissões que foram oferecidas a uma identidade. Qualquer pessoa que tenha acesso a uma assinatura do Azure deve receber apenas as permissões necessárias para que seu trabalho específico possa ser realizado, e nada mais.

A autorização também é popularmente conhecida como RBAC. O RBAC no Azure se refere à atribuição de permissões a identidades em um escopo. O escopo pode ser uma assinatura, um grupo de recursos ou recursos individuais.

O RBAC ajuda na criação e na atribuição de permissões diferentes a identidades diferentes. Isso ajuda a segregar as tarefas dentro das equipes, em vez de todos terem todas as permissões. O RBAC ajuda a tornar as pessoas responsáveis pelo seu trabalho, pois outras talvez nem tenham o acesso necessário para realizá-lo. Deve-se observar que fornecer permissões a um escopo superior automaticamente garante que os recursos filho herdem essas permissões. Por exemplo, fornecer uma identidade com acesso de leitura para um grupo de recursos significa que a identidade terá acesso de leitura a todos os recursos dentro desse grupo também.

O Azure fornece três funções internas de uso geral. São eles:

- A função de proprietário, que tem acesso total a todos os recursos
- A função de colaborador, que tem acesso para leitura/gravação de recursos
- A função de leitor, que tem somente permissões de leitura sobre recursos

Há mais funções fornecidas pelo Azure, mas elas são específicas de recursos, como as funções de colaborador de rede e Gerenciador de segurança.

Para obter todas as funções fornecidas pelo Azure para todos os recursos, execute o comando `Get-AzureRmRoleDefinition` no console do PowerShell.

Cada definição de função tem determinadas ações permitidas e não permitidas. Por exemplo, a função de proprietário tem todas as ações permitidas e nenhuma delas é proibida:

```
PS C:\Users\rimodi> Get-AzureRmRoleDefinition -Name "owner"

Name          : Owner
Id            : 8e3af657-a8ff-443c-a75c-2fe8c4bcb635
IsCustom      : False
Description   : Lets you manage everything, including access to
resources.
Actions       : {*}
NotActions    : {}
AssignableScopes : {/}
```

Cada função comprehende várias permissões. Cada recurso fornece uma lista de operações. A operação com o suporte de um recurso pode ser obtida usando o cmdlet `Get-AzureRmProviderOperation`. Esse cmdlet obtém o nome do provedor e do recurso para recuperar as operações:

```
Get-AzureRmProviderOperation -OperationSearchString "Microsoft.
Insights/*"
```

Isso resultará na seguinte saída:

```
PS C:\Users\rimodi> get-AzureRmProviderOperation  
-OperationSearchString "Microsoft.Insights/*" | select operation  
  
Operation  
-----  
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
```

Funções Personalizadas

As funções personalizadas são criadas por meio da combinação de várias permissões. Por exemplo, uma função personalizada pode consistir em operações de vários recursos, da seguinte forma:

```
$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
```

Como os bloqueios são diferentes do RBAC?

Os bloqueios não são iguais ao RBAC. O RBAC ajuda a permitir ou negar permissões para recursos. Essas permissões estão relacionadas à realização de operações como leitura, gravação e atualização nos recursos. Os bloqueios, por outro lado, estão relacionados à rejeição de permissões para configurar ou excluir recursos.

Um exemplo de implementação de recursos de governança do Azure

Nesta seção, vamos analisar um exemplo de implementação de arquitetura em uma organização fictícia que deseja implementar os recursos de governança e gerenciamento de custos do Azure.

Contexto

Company Inc é uma empresa internacional que está implementando uma solução de mídias sociais em uma plataforma IaaS do Azure. Ela usa servidores Web e servidores de aplicativos implantados em redes e máquinas virtuais do Azure. O Azure SQL Server age como o banco de dados de back-end.

RBAC para Company Inc

A primeira tarefa é garantir que os proprietários de aplicativos e as equipes apropriadas possam acessar seus recursos. Reconhece-se que cada equipe tem requisitos diferentes. Para fins de clareza, o SQL do Azure é implantado em um grupo de recursos separado em comparação aos artefatos da IaaS do Azure.

O administrador atribui as seguintes funções na assinatura:

Função	Atribuída a	Descrição
Proprietário	Administrador	Gerencia todos os grupos de recursos e a assinatura.
Gerente de segurança	Administradores de segurança	Esta função permite que os usuários acessem a Central de Segurança do Azure e o status dos recursos.
Colaborador	Gerenciamento de infraestrutura	Gerencia máquinas virtuais e outros recursos.
Leitor	Desenvolvedores	Pode exibir os recursos, mas não pode modificá-los. Espera-se que os desenvolvedores trabalhem em seus ambientes de desenvolvimento/teste.

Políticas do Azure

A empresa deve implementar políticas do Azure para garantir que seus usuários sempre provisionem recursos de acordo com as diretrizes da empresa.

As políticas do Azure regem vários aspectos relacionados à implantação de recursos. As políticas também governarão as atualizações após a implantação inicial.

Algumas das políticas que devem ser implementadas são fornecidas na seção a seguir.

Implantações em determinados locais

Os recursos e implantações do Azure só podem ser executados para determinados locais escolhidos. Não seria possível implantar recursos em regiões fora da política. Por exemplo, as regiões permitidas são a Europa Ocidental e o Leste dos EUA. Não deve ser possível implantar recursos em qualquer outra região.

Marcas de recursos e grupos de recursos

Todos os recursos do Azure, incluindo os grupos de recursos, obrigatoriamente terão marcas atribuídas a ele. As marcas incluirão, no mínimo, detalhes sobre o departamento, o ambiente, os dados de criação e o nome do projeto.

Logs de diagnóstico e Application insights para todos os recursos

Todos os recursos implantados no Azure devem ter logs de diagnóstico e logs de aplicativos habilitados sempre que possível.

Bloqueios do Azure

A empresa deve implementar bloqueios do Azure para garantir que os recursos importantes e cruciais não sejam apagados acidentalmente.

Cada recurso crucial para o funcionamento de uma solução precisa ser bloqueado. Isso significa que até mesmo os administradores dos serviços em execução no Azure não têm a capacidade de excluir esses recursos. A única maneira de excluir o recurso é removê-lo bloqueio primeiro.

Todos os ambientes de produção e pré-produção, além do ambiente de desenvolvimento e teste, seriam bloqueados para exclusão.

Todos os ambientes de desenvolvimento e teste que têm instâncias únicas também seriam bloqueados para exclusão.

Todos os recursos relacionados a <https://www.eversource.com/content/> seriam bloqueados para exclusão para todos os ambientes de desenvolvimento à produção.

Todos os recursos compartilhados serão bloqueados para exclusão, independentemente do ambiente.

Resumo

A governança e o gerenciamento de custos estão entre algumas das principais prioridades para empresas que estão migrando para a nuvem. Ter uma assinatura pré-paga do Azure com um esquema pré-pago pode prejudicar o orçamento da empresa, pois qualquer um que tenha acesso à assinatura poderá provisionar quantos recursos quiser. Alguns recursos são gratuitos, mas outros são caros. É importante para as organizações manter o controle sobre seus custos de nuvem. As marcas ajudam na geração de relatórios de cobrança. Esses relatórios podem ser baseados em departamentos, projetos, proprietários ou qualquer outro critério. O custo e a governança são igualmente importantes. O Azure fornece bloqueios, políticas e RBAC para implementar a governança adequada. As políticas garantem que as operações de recursos possam ser negadas ou auditadas, os bloqueios garantem que os recursos não possam ser modificados nem excluídos, e o RBAC garante que os funcionários tenham as permissões corretas para a realização de seus trabalhos. Com todos esses quatro recursos, as empresas podem ter governança sólida e controle de custos para suas implantações do Azure.

O DevOps no Azure será o foco do próximo capítulo, no qual alguns de seus conceitos mais importantes e técnicas de implementação serão discutidos.

10

Soluções do Azure que usam Serviços de Contêiner do Azure

Os contêineres são as tecnologias mais comentadas nos dias de hoje. As corporações estão falando sobre contêineres em vez de máquinas virtuais e queremvê-los em quase todas as suas soluções. As soluções podem ser microserviços, funções sem servidor, aplicativos Web, APIs Web ou apenas sobre qualquer coisa.

Neste capítulo, mergulharemos na maioria dos recursos do Azure relacionados a contêineres, incluindo o seguinte:

- Serviço Azure Kubernetes
- Contêineres em máquinas virtuais
- Registro de Contêiner do Azure
- Instância de Contêiner do Azure
- Aplicativos Web para contêineres
- Hospedagem de imagens no Docker Hub

Não entraremos nos detalhes e nos fundamentos dos contêineres e nos concentraremos nos serviços de contêiner específicos do Azure.

Cada contêiner precisa de uma imagem como ponto de partida. Uma imagem pode ser criada usando um Dockerfile ou pode ser baixada de um registro de contêiner. Nós usaremos uma única imagem para todas as nossas amostras neste capítulo, e essa imagem será criada de um Dockerfile.

O conteúdo do Dockerfile usado em todos os exemplos neste capítulo é listado aqui. Todo o código e a solução samplewebapp estão disponíveis com o código-fonte acompanhante:

```
FROM microsoft/dotnet:2.1-aspnetcore-runtime-nanoserver-sac2016 AS
base
WORKDIR /app
EXPOSE 80

FROM microsoft/dotnet:2.1-sdk-nanoserver-sac2016 AS build
WORKDIR /src
COPY ["samplewebapp/samplewebapp.csproj", "samplewebapp/"]
RUN dotnet restore "samplewebapp/samplewebapp.csproj"
COPY . .
WORKDIR "/src/samplewebapp"
RUN dotnet build "samplewebapp.csproj" -c Release -o /app

FROM build AS publish
RUN dotnet publish "samplewebapp.csproj" -c Release -o /app

FROM base AS final
WORKDIR /app
COPY --from=publish /app .
ENTRYPOINT ["dotnet", "samplewebapp.dll"]
```

Registro de Contêiner do Azure

Antes do Registro de Contêiner do Azure, o Docker Hub era o serviço de Registro mais conhecido para imagens do Docker.

O Registro de contêiner do Azure é um repositório alternativo para o Docker Hub. Um registro é um local na Internet que fornece listagens de imagens, juntamente com as instalações para carregar e baixar as imagens sob demanda. Há dois tipos de registros:

- Público
- Privado

Um repositório público, como o nome sugere, é de natureza pública e as imagens dele podem ser baixadas e usadas por qualquer pessoa. No entanto, o upload de imagens para um repositório público é discricionário e, dependendo do provedor, pode ou não permitir o upload de imagens.

Por outro lado, os repositórios privados destinam-se apenas a pessoas que tenham acesso ao repositório. Eles precisam se autenticar antes que possam fazer upload ou download de imagens.

O Docker Hub fornece a capacidade de criar contas de usuário, e essas contas podem ter repositórios públicos, bem como privados.

Na verdade, a Microsoft também tem contas no Docker Hub, onde todas as imagens conhecidas do Windows estão disponíveis. Você pode encontrar todas as imagens da Microsoft executando o comando `docker search Microsoft`, conforme mostrado na captura de tela a seguir:

NAME	DESCRIPTION	STARS
microsoft/dotnet	Official images for .NET Core and ASP.NET Co...	1395
microsoft/mssql-server-linux	Official images for Microsoft SQL Server on ...	1078
microsoft/aspnet	Microsoft IIS images	819
microsoft/windowsservercore	The official Windows Server Core base image	644
microsoft/aspnetcore	Official images for running compiled ASP.NET...	582
microsoft/nanoserver	The official Nano Server base image	473
microsoft/iis	Microsoft IIS images	356
microsoft/mssql-server-windows-developer	Official Microsoft SQL Server Developer Edit...	289
microsoft/mssql-server-windows-express	Official Microsoft SQL Server Express Editio...	281
microsoft/aspnetcore-build	Official images for building ASP.NET Core ap...	270
microsoft/azure-cli	Official images for Microsoft Azure CLI	156
microsoft/powershell	PowerShell for every system!	145
microsoft/vsts-agent	Official images for the Visual Studio Team S...	116
microsoft/dynamics-nav	Official images for Microsoft Dynamics NAV o...	108
microsoft/dotnet-samples	.NET Core Docker Samples	70
microsoft/bcsandbox	Business Central Sandbox	53
microsoft/mssql-tools	Official images for Microsoft SQL Server Com...	51
microsoft/oms	Monitor your containers using the Operations...	41
microsoft/cntk	CNTK images from github.com/Microsoft/CNTK-d...	38
microsoft/wcf	Microsoft WCF images	28
microsoft/dotnet-nightly	Preview images for .NET Core and ASP.NET Cor...	23
microsoft/dotnet-framework-build	The .NET Framework build images have moved t...	17
microsoft/mmlspark	Microsoft Machine Learning for Apache Spark	7
microsoft/aspnetcore-build-nightly	Images to build preview versions of ASP.NET ...	4
microsoft/cntk-nightly	CNTK nightly image from github.com/Microsoft...	2

Como mencionado antes, a Microsoft também fornece um registro alternativo como um serviço do Azure, conhecido como **Registro de Contêiner do Azure**. Ele fornece funcionalidade semelhante à do Docker Hub.

Uma vantagem de usar o Registro de Contêiner do Azure sobre o Docker Hub é que, se você estiver usando uma imagem do Registro de Contêiner do Azure que está no mesmo local do Azure que hospeda contêineres com base na imagem, a imagem pode ser baixada mais rapidamente usando a rede de backbone da Microsoft em vez de passar pela Internet. Por exemplo, hospedar um contêiner no Serviço de Kubernetes do Azure cuja imagem de origem esteja dentro do Registro de Contêiner do Azure será muito mais rápida do que o Docker Hub.

O Registro de Contêiner do Azure é um serviço gerenciado que cuidará de todas as necessidades operacionais relacionadas a imagens, como armazenamento e disponibilidade. As imagens e suas camadas são armazenadas no registro e são enviadas por download para o host de destino para a criação de contêineres sob demanda. É responsabilidade do Azure cuidar dessas imagens e de suas camadas.

A beleza do Registro de Contêiner do Azure é que ele funciona com a CLI do Azure, o PowerShell e o portal do Azure. E não para por aí: ele também tem alta fidelidade com a linha de comando do Docker. Você pode usar comandos do Docker para fazer upload e download de imagens do Registro de Contêiner do Azure em vez do Docker Hub.

Agora, vamos nos concentrar em como trabalhar com o Registro de Contêiner do Azure usando a CLI do Azure:

1. Obviamente, a primeira tarefa é fazer logon no Azure usando o comando `az login`:

```
C:\Users\citynextadmin>az login
```

2. Se você tiver várias assinaturas associadas ao mesmo logon, selecione uma assinatura apropriada para trabalhar com o Registro de Contêiner do Azure:

```
C:\Users\citynextadmin>az account set --subscription [REDACTED]
```

3. Crie um novo grupo de recursos para hospedar uma nova instância do Registro de Contêiner do Azure:

```
C:\Users\citynextadmin>az group create --name rg03 --location "west europe" --verbose
{
  "id": "/subscriptions/[REDACTED]/resourceGroups/rg03",
  "location": "westeurope",
  "managedBy": null,
  "name": "rg03",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null
}
```

4. A CLI do Azure fornece comandos `acr`. Crie uma nova instância do Registro de Contêiner do Azure usando o seguinte comando:

```
az acr create
```

5. Você precisa fornecer o SKU do Registro de Contêiner do Azure, um nome e o nome de um grupo de recursos. Ele também pode opcionalmente habilitar privilégios de administrador.

6. Os SKUs disponíveis para o Registro de Contêiner do Azure são os seguintes:
 - **Básico:** isso não é recomendado para cenários de produção e é mais adequado para o trabalho de desenvolvimento/teste. Isso é porque a quantidade de recursos disponíveis para armazenamento e largura de banda é restrita em comparação com SKUs mais altos.
 - **Padrão:** tem todos os recursos do Básico, juntamente com maior configuração e disponibilidade de recursos. Isso é adequado para implantações de produção.
 - **Premium:** novamente, tem todos os recursos do SKU Padrão junto com maior disponibilidade de recursos. Tem recursos adicionais, como replicação geográfica.
 - **Clássico:** não é um serviço gerenciado e as imagens armazenadas usando esse SKU são armazenadas em contas de armazenamento fornecidas pelo usuário. Os usuários precisam gerenciar ativamente essas contas de armazenamento em termos de segurança, administração e governança.
7. A saída a seguir contém informações importantes usadas com bastante frequência para fazer upload e download de imagens. Eles incluem as propriedades `loginServer` e `name`. O nome de usuário do Registro de Contêiner do Azure é igual ao seu nome:

```
C:\Users\citynextadmin>az acr create --sku basic --resource-group rg03 --name sampleappbookacr --verbose --admin-enabled
{
  "adminUserEnabled": true,
  "creationDate": "2018-12-31T07:15:48.719352+00:00",
  "id": "/subscriptions/[REDACTED]/resourceGroups/rg03/providers/Microsoft.ContainerRegistry/registries/sampleappbookacr",
  "location": "westeurope",
  "loginServer": "sampleappbookacr.azurecr.io",
  "name": "sampleappbookacr",
  "provisioningState": "Succeeded",
  "resourceGroup": "rg03",
  "sku": {
    "name": "Basic",
    "tier": "Basic"
  },
  "status": null,
  "storageAccount": null,
  "tags": {},
  "type": "Microsoft.ContainerRegistry/registries"
}
```

8. O serviço no portal é configurado como mostrado na seguinte captura de tela:

The screenshot shows the 'Access keys' section of the Azure Container Registry 'samplewebappbookacr'. The left sidebar lists 'Overview', 'Activity log', 'Access control (IAM)', 'Tags', 'Quick start', 'Events', 'Settings' (with 'Access keys' selected), 'Locks', and 'Automation script'. The main area displays the following configuration:

- Registry name:** samplewebappbookacr
- Login server:** samplewebappbookacr.azurecr.io
- Admin user:** A button labeled 'Enable' is highlighted.
- Username:** samplewebappbookacr
- password:** TcU1V3G8vwNI=f6JKxBCQgFD05=kijfb
- password2:** d5Jkcu08phgoVgFw/oAhShYLGzWD02qX

9. Se o administrador estiver habilitado, as credenciais para o administrador poderão ser buscadas usando um comando, conforme mostrado na captura de tela a seguir. Há duas senhas geradas, que podem ser trocadas quando necessário. O logon é necessário para enviar por push imagens e não é necessário para obter imagens do Registro de Contêiner do Azure:

```
C:\Users\citynextadmin>az acr credential show --name sampleappbookacr --resource-group rg03 --verbose
{
  "passwords": [
    {
      "name": "password",
      "value": "igzgtUxlehYn2/92KvNk=+X4UtbtUQim"
    },
    {
      "name": "password2",
      "value": "XMbIrrHpl0H9DQmLu/ReXmYBietU8ht3"
    }
  ],
  "username": "sampleappbookacr"
}
```

10. Usando os dados de nome de usuário e senha (que temos até agora), é possível fazer logon no Registro de Contêiner do Azure:

```
C:\Users\citynextadmin>az acr login --name sampleappbookacr --password "igzgtUxlehYn2/92KvNk=x4Utb1UQim" --resource-group rg03 --username sampleappb  
okacr  
Login Succeeded  
WARNING! Your password will be stored unencrypted in C:\Users\citynextadmin\.docker\config.json.  
Configure a credential helper to remove this warning. See  
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
```

11. Você precisará iniciar sessão no Registo de Contêiner do Azure se estiver protegido por senha. A próxima tarefa é preparar imagens locais para que seja possível fazer upload delas para o Registro de Contêiner do Azure. Essa preparação precisa marcar as imagens locais com o nome do servidor do registro.
12. A captura de tela a seguir mostra um comando no qual uma imagem chamada samplewebapp é marcada usando o comando tag para sampleappbooacr.azurecr.io/samplewebapp:latest. Aqui, sampleappbooacr.azurecr.io refere-se ao nome do servidor que foi criado anteriormente como parte da criação do registro. Observe que os comandos do Docker são usados para marcar as imagens:

```
C:\Users\citynextadmin>docker tag samplewebapp:latest sampleappbookacr.azurecr.io/samplewebapp:latest
```

13. Depois de marcar usando o comando docker push, a imagem pode ser enviada por push para o Registro de Contêiner do Azure:

```
C:\Users\citynextadmin>docker push sampleappbookacr.azurecr.io/samplewebapp:latest  
The push refers to repository [sampleappbookacr.azurecr.io/samplewebapp]  
036d27f25d59: Pushed  
f6ac38202aab: Pushed  
7605499be4ca: Pushed  
1bd33b5316f6: Pushed  
f3eff1ccb92a: Pushed  
cbe35d8d94e0: Pushed  
e36c5fabaebe: Pushed  
8b28210b2d46: Pushed  
cc80cb0b7044: Pushed  
e4a9564a3ea3: Pushed  
f5c673c58dcc: Skipped foreign layer  
6c357baed9f5: Skipped foreign layer  
latest: digest: sha256:da53d5ab40890824b24712e37129819f2d4b65d8eb1df5c5df8b2bc65786d1fd size: 3026
```

14. A partir de agora, qualquer pessoa que tenha conhecimento sobre o registro sampleappbookacr, poderá consumir essa imagem.

O registro no portal é mostrado na seguinte captura de tela:

The screenshot shows the Azure Container Registry interface. On the left, there's a sidebar with options like Overview, Activity log, Access control (IAM), Tags, Quick start, Events, Settings (Access keys, Locks, Automation script), and Services (Repositories). The main area is titled "samplewebappbookacr - Repositories" and shows a list of repositories. One repository, "samplewebapp", is selected and highlighted in blue. To the right of the repository list is a "samplewebapp" card with a "Repository" section containing a "Refresh" button and a "Delete" button, and an "Essentials" section with a search bar for tags.

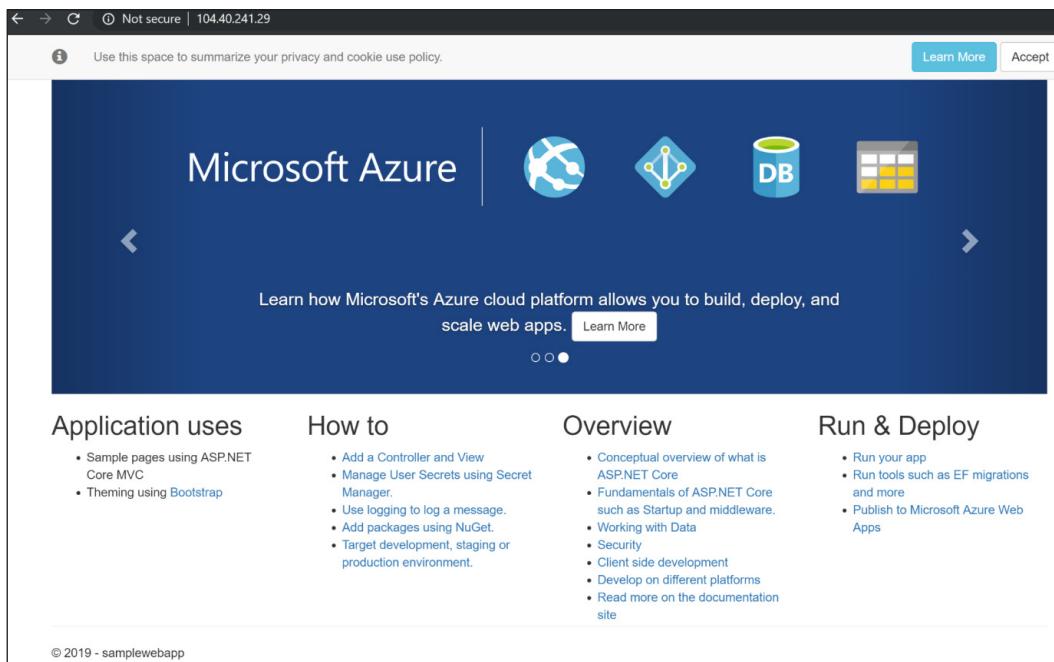
15. Crie um contêiner da imagem que foi carregada apenas no Registro de Contêiner do Azure. Para este exercício, crie uma nova máquina virtual com base no Windows Server 2016 ou Windows 10 que tenha os contêineres habilitados e execute o comando, conforme mostrado na captura de tela a seguir, para criar um novo contêiner usando a imagem samplewebapp do registro samplewebappbookacr:

```
C:\Users\citynextadmin>docker run -it -p 80:80 samplewebappbookacr.azurecr.io/samplewebapp:latest powershell
```

16. É importante notar que, mesmo se a imagem não está disponível localmente, o comando irá baixar as imagens durante a execução do comando e, em seguida, criar um contêiner fora dele. A saída da execução do contêiner é mostrada na captura de tela a seguir:

```
Administrator: Command Prompt - docker run -it -p 80:80 samplewebapp:latest -d
Hosting environment: Production
Content root path: C:\app
Now listening on: http://[::]:80
Application started. Press Ctrl+C to shut down.
```

17. O resultado mostrado na captura de tela a seguir exibe que o aplicativo Web está em execução no momento:



O Dockerfile e o arquivo de solução usados para criar a imagem local estão disponíveis como parte do código-fonte. Também é possível fazer upload da mesma imagem para o Docker Hub.

18. A primeira etapa é ter uma conta no Docker Hub. Para isso, uma nova conta ou uma conta existente pode ser usada.
19. Depois de criar a conta, faça logon na sua conta.

20. Crie um novo repositório selecionando **Criar um repositório**, conforme mostrado na captura de tela a seguir:



21. Forneça um nome e selecione um repositório Público:

The screenshot shows the 'Create Repository' form on the Docker Hub website. At the top, it says 'Repositories / Create'. On the right, it indicates 'Using 0 of 1 private repositories. [Get more](#)'. The main area is titled 'Create Repository' and has a dropdown for 'Owner' set to 'azureforarchitects' and a repository name 'samplewebapp'. A note below says 'this repos contains image for samplewebapp for book'. To the right is a 'Pro tip' section with a command box containing 'docker tag local-image:tagname new-repo:tagname' and 'docker push new-repo:tagname'. Below that is a note: 'Make sure to change *tagname* with your desired image repository tag.' Under 'Visibility', the 'Public' radio button is selected, with the note 'Public repositories appear in Docker Hub search results'. The 'Private' radio button is also shown. In the 'Build Settings (optional)' section, there's a note about autobuild and a link to 'Learn More'. Below that, there's a warning about GitHub and Bitbucket account linking. At the bottom, there are three buttons: 'Cancel' (red border), 'Create' (blue), and 'Create & Build' (blue).

22. A próxima captura de tela mostra o comando a ser executado para enviar por push imagens para este repositório:

Repositories / [azureforarchitects / samplewebapp](#) Using 0 of 1 private repositories. [Get more](#)

[General](#) Tags Builds Timeline Collaborators Webhooks Settings

azureforarchitects / samplewebapp

this repos contains image for samplewebapp for book

⌚ Last pushed: a few seconds ago

Docker commands

To push a new tag to this repository,

```
docker push  
azureforarchitects/samplewebapp:tagname
```

Tags

This repository contains 1 tag(s).

latest		⌚ a few seconds ago
--------	--	---------------------

[See all](#)

Full Description

Repository description is empty. Click [here](#) to edit.

23. Da sua máquina virtual que contém a imagem local para samplewebapp, crie uma nova marca. Este é um exercício semelhante ao que fizemos para o Registro de Contêiner do Azure; no entanto, desta vez, é para o Docker Hub:

```
C:\Users\citynextadmin>docker tag samplewebapp:latest azureforarchitects/samplewebapp:latest
```

24. Usando a CLI do Docker, faça logon no Docker Hub:

```
C:\Users\citynextadmin>docker login -u azureforarchitects -p Dsc123..  
WARNING! Using --password via the CLI is insecure. Use --password-stdin.  
WARNING! Your password will be stored unencrypted in C:\Users\citynextadmin\.docker\config.json.  
Configure a credential helper to remove this warning. See  
https://docs.docker.com/engine/reference/commandline/login/#credentials-store  
Login Succeeded
```

25. Por fim, envie por push a imagem recém-marcada para o Hub Docker usando o comando docker push:

```
C:\Users\citynextadmin>docker push azureforarchitects/samplewebapp:latest
The push refers to repository [docker.io/azureforarchitects/samplewebapp]
036d27f25d59: Pushed
f6ac38202aab: Pushed
7605499be4ca: Pushed
1bd33b5316f6: Pushed
f3eff1ccb92a: Pushed
cbe35d8d94e0: Pushed
e36c5fabaebb: Pushed
8b28210b2d46: Pushed
cc80cb0b7044: Pushed
e4a9564a3ea3: Pushed
f5c673c58dcc: Skipped foreign layer
6c357baed9f5: Skipped foreign layer
latest: digest: sha256:da53d5ab40890824b24712e37129819f2d4b65d8eb1df5c5df8b2bc65786d1fd size: 3026
```

Agora, essa nova imagem pode ser usada em hosts diferentes, incluindo os Serviços de Aplicativo do Azure.

Instâncias de Contêiner Azure

Na última seção, nós criamos um contêiner com base em uma imagem do Registro de Contêiner do Azure em uma máquina virtual. Nesta seção, investigaremos as Instâncias de Contêiner do Azure. As Instâncias de Contêiner do Azure são um serviço gerenciado fornecido pelo Azure para hospedar contêineres. É um serviço que ajuda na execução de contêineres, fornecendo uma plataforma totalmente gerenciada pelo Azure.

O uso de Instâncias de Contêiner do Azure elimina completamente a necessidade de qualquer componente **Infraestrutura como Serviço (IaaS)**, como a criação de máquinas virtuais para hospedar contêineres. Usaremos novamente a CLI do Azure para criar Instâncias de Contêiner do Azure também.

Prossiga com as etapas da seguinte maneira:

1. Se você ainda não tiver entrado no Azure usando a CLI, faça logon no Azure usando o comando az login e selecione uma assinatura apropriada.
2. O comando para fazer logon no Azure é mostrado a seguir. Ele abrirá uma janela do navegador solicitando credenciais para começar com o processo de logon:

```
C:\Users\citynextadmin>az login
```

3. Depois de se conectar ao Azure, o comando para selecionar uma assinatura adequada deve ser executado como mostrado a seguir:

```
C:\Users\citynextadmin>az account set --subscription
```

4. Crie um novo grupo de recursos para hospedar uma Instância de Contêiner do Azure usando o seguinte comando:

```
C:\Users\citynextadmin>az group create --name rg04 --location "west europe" --verbose
```

5. Crie uma Instância do Registro de Contêiner do Azure usando o seguinte comando:

```
az container create
```

Passe as informações necessárias descritas nos seguintes pontos de marcador:

- O nome DNS por meio do qual o contêiner pode ser acessado.
- O número de CPUs alocadas para o contêiner.
- O nome de imagem totalmente qualificado – é o mesmo nome que criamos na seção anterior (`samplewebappbookacr.azurecr.io/samplewebapp:latest`).
- O tipo de endereço IP, seja público ou privado – um endereço IP público garantirá que o contêiner esteja disponível para ser consumido na Internet.
- A localização do contêiner – neste caso, é igual à localização do registro de contêiner. Isso ajuda na criação mais rápida de contêineres.
- A memória em GB a ser alocada para o contêiner.
- O nome do contêiner.
- O sistema operacional do contêiner usando a opção `-os-type`. Neste caso, é o Windows.
- O protocolo e as portas a serem abertas. Abrimos a porta 80, já que nosso aplicativo será hospedado nessa porta.
- A política de reinicialização é definida como sempre. Isso garantirá que, mesmo se o host que estiver executando o contêiner for reiniciado, o contêiner será executado automaticamente.
- As informações de Registro de Contêiner do Azure de onde a imagem será obtida, incluindo o servidor, a senha e o nome de usuário de logon do registro:

```
C:\Users\citynextadmin>az container create --resource-group rg03 --cpu 4 --dns-name-label mysamplewebappcontainer --image sampleappbookacr.azurecr.io/samplewebapp:latest --ip-address Public --location "West Europe" --memory 7 --name mysamplewebappcontainer --os-type Windows --protocol tcp --ports 8 --restart-policy Always --registry-login-server sampleappbookacr.azurecr.io --registry-password "igzgtUxLehYn2/92KvNk+=X4UtbiUQim" --registry-username
```

O comando deve ser executado com êxito para criar um novo contêiner. Isso criaria um novo contêiner na Instância de Contêiner do Azure. A configuração do portal é a seguinte:

The screenshot shows the Azure Container Instances blade for the resource group rg04. The container instance named mysamplewebappcontainer is selected. The main pane displays the following details:

- Resource group (change):** rg04
- Status:** Running
- Location:** West Europe
- Subscription (change):** Visual Studio Ultimate with MSDN
- Subscription ID:** 364c7779-65a1-40ed-8847-c7a86661c1c3
- FQDN:** mysamplewebappcontainer.westeuropewest.azurecontainer.io
- Container count:** 1
- Tags:** Click here to add tags

A configuração do contêiner é a seguinte:

The screenshot shows the Azure Container Instances - Containers blade for the resource group rg04. It lists one container named mysamplewebappcontainer. The blade also includes a log viewer at the bottom:

NAME	IMAGE	STATE	START TIME	RESTART COUNT
mysamplewebappcontainer	samplewebappbookacr.azurecr.io...	Running	2019-01-24T09:07:20Z	0

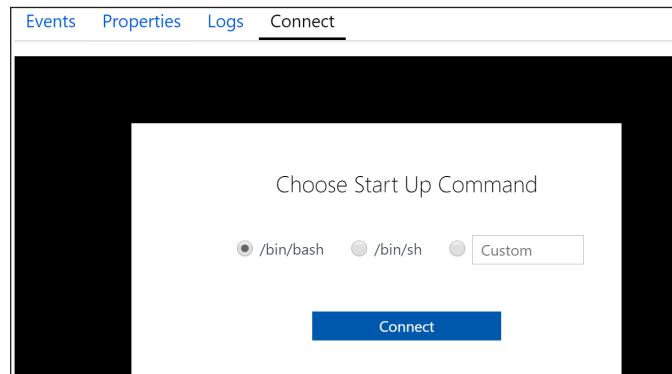
Events

NAME	TYPE	FIRST TIMESTAMP	LAST TIMESTAMP	MESSAGE	COUNT
Started	Normal	1/24/2019, 4:07 AM EST	1/24/2019, 4:07 AM EST	Started container with dock...	1
Pulled	Normal	1/24/2019, 4:07 AM EST	1/24/2019, 4:07 AM EST	Successfully pulled image "s...	1
Created	Normal	1/24/2019, 4:07 AM EST	1/24/2019, 4:07 AM EST	Created container with dock...	1
Pulling	Normal	1/24/2019, 4:06 AM EST	1/24/2019, 4:06 AM EST	pulling image "sampleweba..."	1

As propriedades são as mesmas que fornecemos ao criar a instância de contêiner:

Events	Properties	Logs	Connect
Name	mysamplewebappcontainer		
Image	samplewebappbookacr.azurecr.io/samplewebapp:latest		
Ports	80		
Memory	7 GB		
Cores	4		
Commands	(no commands)		
Environment variables			

Também é possível conectar-se ao contêiner usando a guia **Conectar**:



A saída da página final é mostrada na seguinte captura de tela:

The screenshot shows a web browser window with the URL mysamplewebappcontainer.westeurope.azurecontainer.io. At the top, there's a privacy and cookie use policy summary area with 'Learn More' and 'Accept' buttons. Below this is a purple header bar with the Visual Studio logo, HTML5, CSS3, and JS icons. A message says: "There are powerful new features in Visual Studio for building modern web apps." Below the header are four main sections:

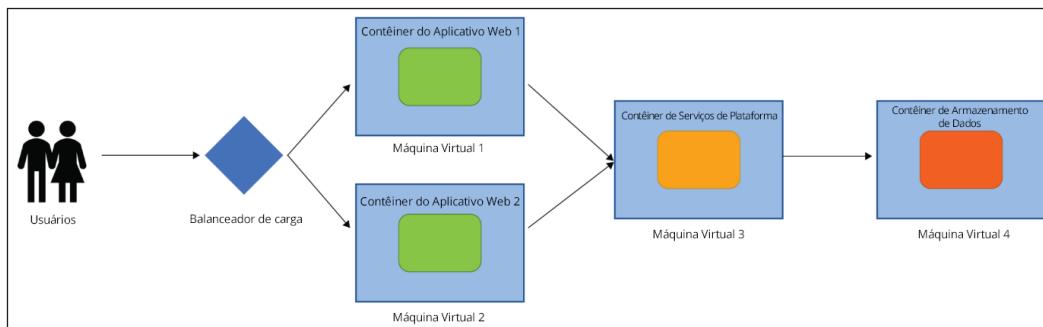
- Application uses**: Includes bullet points: "Sample pages using ASP.NET Core MVC", "Theming using Bootstrap".
- How to**: Includes bullet points: "Add a Controller and View", "Manage User Secrets using Secret Manager", "Use logging to log a message", "Add packages using NuGet", "Target development, staging or production environment".
- Overview**: Includes bullet points: "Conceptual overview of what is ASP.NET Core", "Fundamentals of ASP.NET Core such as Startup and middleware", "Working with Data", "Security", "Client side development", "Develop on different platforms", "Read more on the documentation site".
- Run & Deploy**: Includes bullet points: "Run your app", "Run tools such as EF migrations and more", "Publish to Microsoft Azure Web Apps".

At the bottom left, it says "© 2019 - samplewebapp".

Serviço Azure Kubernetes

O Serviço de Kubernetes do Azure é um dos principais ambientes de contêiner gerenciados no mundo. Antes de entrar no Serviço de Kubernetes do Azure, vamos entender o que é o kubernetes e por que ele é necessário.

Kubernetes é um mecanismo e serviço de orquestração de contêiner. A orquestração de contêineres é uma obrigação para qualquer solução empresarial implantada em cima de contêineres. Imagine que haja quatro contêineres para uma solução, que são hospedados em quatro máquinas virtuais diferentes, conforme mostrado no diagrama a seguir:



Os contêineres, por sua própria natureza, são bastante voláteis. Podem ser criados com facilidade e com bastante rapidez, e também podem ser removidos com a mesma facilidade. Por exemplo, se a máquina virtual trés for reiniciada, o contêiner em execução também será destruído. Se qualquer contêiner, ou qualquer host que hospede um contêiner, ficar inativo por qualquer motivo, o aplicativo e a solução ficarão inativos. Isso ocorre porque não há nenhum monitoramento ativo desses contêineres e hosts.

Embora os sistemas de monitoramento tradicionais possam ser usados para monitorar os hosts, não há uma única solução que gerencie e monitore os hosts e contêineres juntos. Além disso, o problema não está relacionado apenas à disponibilidade. A integridade dos hosts e contêineres também pode ter um impacto significativo para o funcionamento do aplicativo. Mesmo se um contêiner não for capaz de executar conforme o esperado, a solução inteira terá um desempenho reduzido.

Os orquestradores do contêiner resolvem estes problemas acima mencionados com contêineres. Os orquestradores eliminam a carga de monitoração e de gerência ativos dos usuários e automatizam o conjunto inteiro das atividades e de seu impacto. Um orquestrador pode identificar se qualquer host ou qualquer contêiner está sofrendo performance degradada. Ele pode acelerar imediatamente novos contêineres em hosts diferentes. Não só isso, mas mesmo se um dos contêineres ficar inativo, uma nova instância de contêiner poderá ser criada em outro host. Isso não para por aqui. Se um host completo ficar inativo, todos os contêineres em execução nesse host poderão ser recriados em um host diferente. Isso é responsabilidade de orquestradores de contêiner.

Um orquestrador de contêiner deve ser informado sobre a quantidade desejada e esperada de cada tipo de contêiner. Para o nosso exemplo, a quantidade desejada de contêineres de aplicativos Web é dois, para serviços de plataforma é um e para o armazenamento de dados é um. Se essas informações forem alimentadas em um orquestrador, ele sempre tentará manter o ambiente para ter a quantidade desejada de contêineres. Se os contêineres ficarem inativos em qualquer host, o orquestrador observará isso e criará um novo contêiner em outro host para compensar.

O Kubernetes é um orquestrador de contêiner e fornece todos esses recursos.

O Kubernetes pode ser implantado em máquinas virtuais diretamente por meio da implantação de IaaS e, ao mesmo tempo, pode ser provisionado como um Serviço de Kubernetes do Azure. A diferença entre os dois paradigmas de implantação diferentes é que a implantação de IaaS é controlada e gerenciada pelo usuário, enquanto o Serviço de Kubernetes do Azure é um serviço gerenciado e todos os hosts e contêineres são gerenciados pelo Azure sem qualquer esforço do usuário que o implanta. O gerenciamento de uma implantação de Kubernetes baseada em IaaS não é uma tarefa fácil. Somente as pessoas com conhecimento profundo do Kubernetes são capazes de gerenciar esse ambiente, e não é fácil encontrar esses profissionais. Além disso, a implantação geral do Kubernetes e seu gerenciamento é um exercício complexo e complicado que poucas organizações estão dispostas a empreender. Por esses motivos, o Serviço de Kubernetes do Azure é popular entre as organizações que terceirizam todo o gerenciamento do Kubernetes para o Azure.

O Kubernetes, sendo baseado em mecanismos de orquestrador, funciona perfeitamente com o tempo de execução do Docker. Ele pode, no entanto, até mesmo trabalhar com outros tempos de execução de contêiner e provedores de rede.

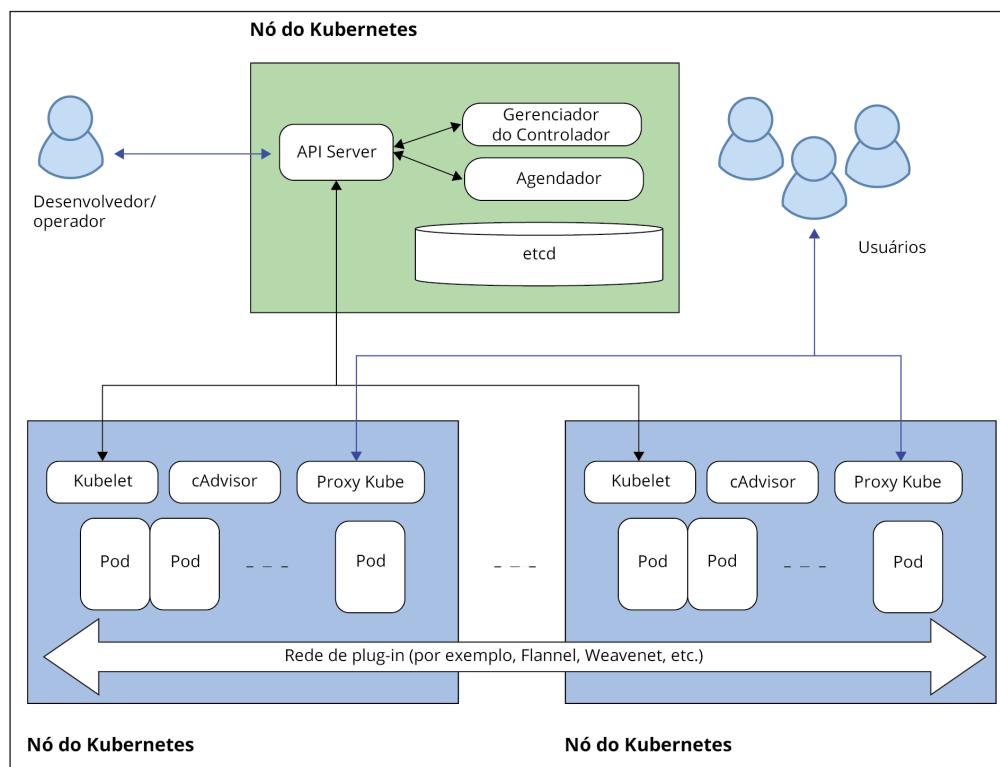
Arquitetura do Kubernetes

O Kubernetes compreende um conjunto de servidores, e esses servidores juntos são chamados de clusters. O Kubernetes tem dois tipos de nós:

- nós mestres
- Nós de pod

Nós mestres

Os nós mestres contêm todos os componentes de gerenciamento do Kubernetes, como o **servidor de API**, o **agendador**, os serviços de configuração distribuídos, como etcd e o controlador de replicação, enquanto os nós de pod contêm os componentes **Kube-Proxy** e **Kubelet**. Os contêineres são implantados em nós de pod:



Pods

O primeiro componente a ser entendido no Kubernetes é o conceito de pods. Os pods são coleções de contêineres implantados, criados e gerenciados como uma única unidade. Os contêineres não são a menor unidade de gerenciamento no Kubernetes, mas sim os pods.

Os pods são criados usando ferramentas da CLI fornecidas pelo Kubernetes, como Kubectl, ou usando arquivos YAML. Esses arquivos YAML contêm os metadados do pod e as informações de configuração. Quando esses arquivos YAML forem enviados para o Kubernetes, o mecanismo principal do kubernetes agenda a criação de pods em nós de pod.

servidor de API

O servidor de API é o cérebro de todo o serviço Kubernetes. Todas as solicitações de usuários realmente são enviadas para o servidor de API. O servidor de API é um ponto de extremidade baseado em REST que aceita solicitações e atua neles. Por exemplo, as solicitações para criar pods são submetidas ao servidor de API. O servidor de API armazena essas informações e informa outro componente, conhecido como agendador, para provisionar pods com base nos arquivos YAML. As informações de integridade dos pods e nós também são enviadas para o servidor de API por Kubelets de nós de pod.

Kubelets

Os Kubelets são componentes provisionados em nós de pod. Seu trabalho é monitorar os nós de pod e os pods nesses nós e informar o servidor de API sobre sua integridade. Os Kubelets também são responsáveis por configurar o componente Kube-Proxy, que desempenha um papel crucial no fluxo de solicitações entre pods e nós que surgem de fontes externas e internas.

Proxy Kube

O Kube-Proxy é responsável por manter rotas nos nós de pod. Também é responsável por alterar os cabeçalhos dos pacotes de dados com as informações de **Conversão de Endereços de Rede de Origem (SNAT)** e de **Conversão de Endereços de Rede de Origem (DNAT)** adequadas e permite o fluxo de tráfego entre pods e nós no cluster.

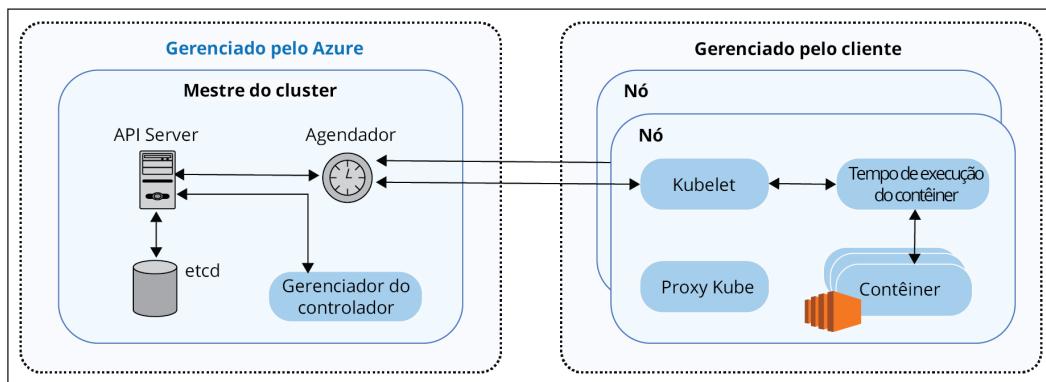
Controlador de replicação/gerenciador de controlador

O controlador de replicação é responsável por manter o número desejado de instâncias de pod. Se pods ou nós ficarem inativos, o Kubelet informará o servidor de API e o servidor de API informará o controlador de replicação. O controlador agendará imediatamente a criação de novos pods nos mesmos nós ou em diferentes, com base na integridade atual do nó.

Arquitetura do Azure Kubernetes

Quando provisionamos o Serviço de Kubernetes do Azure, temos um ambiente que consiste em um cluster do Kubernetes inteiro composto de nós mestres e de pod. Esses nós de pod já têm o tempo de execução do Docker e os plugins de rede instalados e configurados.

O mestre tem todos os componentes instalados e configurados. O mestre é gerenciado pelo Serviço de Kubernetes do Azure, enquanto os nós são gerenciados pelos clientes. Isso é mostrado no diagrama a seguir:



Como provisionar o Serviço de Kubernetes do Azure

Nesta seção, provisionaremos o Serviço de Kubernetes do Azure usando a CLI do Azure. O Serviço de Kubernetes do Azure também pode ser provisionado do portal do Azure e do Azure PowerShell:

1. Faça logon no Azure usando o seguinte comando:
`az login`
2. Selecione uma assinatura apropriada usando o seguinte comando:
`az account set -subscription <<xxxxxxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxx>>`
3. Os recursos do Serviço de Kubernetes do Azure estão disponíveis no comando `az aks`.
4. Crie um novo grupo de recursos para hospedar o Serviço de Kubernetes do Azure usando o seguinte comando:
`az group create --name "akdemo" --location "west Europe"`

5. Depois que o grupo de recursos for criado, podemos criar o cluster do Kubernetes usando o seguinte comando:

```
az aks create --resource-group akdemo --name AKSPortalCluster  
--node-count 2 --enable-addons monitoring --generate-ssh-keys
```

6. Este comando demora um pouco para criar o cluster do Kubernetes.
7. Depois que o cluster for provisionado, será possível baixar a ferramenta de CLI Kubectl usando a CLI do Azure e configurá-la para funcionar sem o cluster.
8. O comando para fazer download da ferramenta Kubectl é az aks install-cli:

```
C:\Users\citynextadmin>az aks install-cli  
Downloading client to "C:\Users\citynextadmin\.azure-kubectl\kubectl.exe" from "https://storage.googleapis.com/kubernetes-release/release/v1.13.2/bin/w  
indows/amd64/kubectl.exe"  
Please add "C:\Users\citynextadmin\.azure-kubectl" to your search PATH so the 'kubectl.exe' can be found. 2 options:  
  1. Run "set PATH=%PATH%;C:\Users\citynextadmin\.azure-kubectl" or "$env:path += 'C:\Users\citynextadmin\.azure-kubectl'" for PowerShell. This is go  
d for the current command session.  
  2. Update system PATH environment variable by following "Control Panel->System->Advanced->Environment Variables", and re-open the command window. Y  
ou only need to do it once
```

É importante notar que a ferramenta Kubectl deve estar no caminho de pesquisa para executá-la na linha de comando pelo seu nome. Se não estiver no caminho, então todo o caminho da ferramenta Kubectl deverá ser fornecido sempre que for necessário executá-la. Uma abordagem ideal é adicioná-la ao caminho de pesquisa usando o seguinte comando:

```
set PATH=%PATH%;C:\Users\citynextadmin\.azure-kubectl
```

Depois de executar este comando, é possível executar o comando kubectl:

```
C:\Users\citynextadmin>kubectl  
kubectl controls the Kubernetes cluster manager.  
  
Find more information at: https://kubernetes.io/docs/reference/kubectl/overview/  
  
Basic Commands (Beginner):  
  create      Create a resource from a file or from stdin.  
  expose      Take a replication controller, service, deployment or pod and expose it as a new Kubernetes Service  
  run         Run a particular image on the cluster  
  set         Set specific features on objects  
  
Basic Commands (Intermediate):  
  explain     Documentation of resources  
  get         Display one or many resources  
  edit        Edit a resource on the server  
  delete      Delete resources by filenames, stdin, resources and names, or by resources and label selector
```

Nesta fase, a ferramenta Kubectl pode ser executada, mas ainda não está conectada ao nosso cluster provisionado recentemente. O kubectl precisa de credenciais configuradas para poder ser autenticado pelo cluster. Note que não fornecemos nenhuma informação de credencial ao criar o cluster e as credenciais foram geradas pelo Serviço de Kubernetes do Azure ao provisionar o cluster.

Essas credenciais podem ser obtidas por download e o Kubectl pode ser configurado usando o comando fornecido pelo Serviço de Kubernetes do Azure na CLI do Azure:

```
C:\Users\citynextadmin>az aks get-credentials --resource-group akdemo --name AKSPortalCluster
Merged "AKSPortalCluster" as current context in C:\Users\citynextadmin\.kube\config
```

A próxima etapa é criar pods no cluster. Para isso, criaremos um novo arquivo YAML e o submeteremos ao servidor de API usando os comandos disponíveis na ferramenta CLI do Kubectl.

O arquivo YAML deve ser salvo em uma máquina local para que ele possa ser referenciado da linha de comando. Não é possível explicar o arquivo YAML neste livro, mas há outros livros que cobrem as configurações de YAML e kubernetes em detalhes. O exemplo de YAML está listado aqui:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: azure-vote-back
spec:
  replicas: 1
  selector:
    matchLabels:
      app: azure-vote-back
  template:
    metadata:
      labels:
        app: azure-vote-back
    spec:
      containers:
        - name: azure-vote-back
          image: redis
      resources:
        requests:
          cpu: 100m
          memory: 128Mi
        limits:
          cpu: 250m
          memory: 256Mi
      ports:
        - containerPort: 6379
          name: redis
---
apiVersion: v1
```

```
kind: Service
metadata:
  name: azure-vote-back
spec:
  ports:
    - port: 6379
  selector:
    app: azure-vote-back
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: azure-vote-front
spec:
  replicas: 1
  selector:
    matchLabels:
      app: azure-vote-front
  template:
    metadata:
      labels:
        app: azure-vote-front
    spec:
      containers:
        - name: azure-vote-front
          image: microsoft/azure-vote-front:v1
          resources:
            requests:
              cpu: 100m
              memory: 128Mi
            limits:
              cpu: 250m
              memory: 256Mi
          ports:
            - containerPort: 80
          env:
            - name: REDIS
              value: "azure-vote-back"
---
apiVersion: v1
kind: Service
metadata:
  name: azure-vote-front
spec:
```

```
type: LoadBalancer
ports:
- port: 80
selector:
app: azure-vote-front
```

O arquivo foi salvo como `aksdemo.yaml`. Este arquivo YAML é responsável pela criação de dois tipos de pod. Cada um desses tipos deve ser usado para criar uma única instância. Em resumo, deve haver dois pods criados, um para cada tipo. Ambas as especificações do pod estão disponíveis no arquivo YAML na seção `template`.

O arquivo YAML pode ser enviado usando o comando `apply`:

```
C:\Users\citynextadmin>kubectl apply -f aksdemo.yaml
deployment.apps/azure-vote-back created
service/azure-vote-back created
deployment.apps/azure-vote-front created
service/azure-vote-front created
```

Nesta fase, os nós podem ser consultados da seguinte forma:

```
C:\Users\citynextadmin>kubectl get nodes
NAME           STATUS   ROLES   AGE   VERSION
aks-nodepool1-39448715-0   Ready    agent   20m   v1.9.11
aks-nodepool1-39448715-1   Ready    agent   20m   v1.9.11
```

Os pods podem ser consultados como mostrado na seguinte captura de tela:

```
C:\Users\citynextadmin>kubectl get pods
NAME          READY   STATUS      RESTARTS   AGE
azure-vote-back-f7c88f6f5-2zxp7   1/1     Running    0          67s
azure-vote-front-7c79dddc4c-f99hl  0/1     ContainerCreating   0          67s
```

Assim que os pods forem criados, eles deverão mostrar um status `Em Execução`. No entanto, não são apenas os pods que precisavam ser criados. Há mais recursos para criar no cluster, como mostrado na próxima captura de tela.

Isso inclui dois serviços. Um serviço, do tipo `ClusterIP`, é usado para comunicação interna entre pods, e o outro, do tipo `LoadBalancer`, é usado para habilitar solicitações externas para alcançar o pod. A configuração dos serviços está disponível no arquivo YAML como `kind: Service`. Há duas implantações também; as implantações são recursos que gerenciam o processo de implantação. A configuração dos serviços está disponível no arquivo YAML como `kind: Deployment`.

Dois conjuntos de réplicas também são criados, um para cada tipo de pod. Os conjuntos de réplicas eram anteriormente conhecidos como controladores de replicação e garantem que o número desejado de instâncias de pod seja sempre mantido no cluster. A configuração dos serviços está disponível no arquivo YAML como o elemento `spec`:

```
C:\Users\citynextadmin>kubectl get all
NAME                                         READY   STATUS    RESTARTS   AGE
pod/azure-vote-back-f7c88f6f5-2zxp7        1/1     Running   0          4m11s
pod/azure-vote-front-7c79dddc4c-f99hl      1/1     Running   0          4m11s

NAME                TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)         AGE
service/azure-vote-back  ClusterIP      10.0.180.11    <none>          6379/TCP       4m11s
service/azure-vote-front LoadBalancer  10.0.56.53     52.236.179.79  80:32110/TCP   4m11s
service/kubernetes       ClusterIP      10.0.0.1       <none>          443/TCP        27m

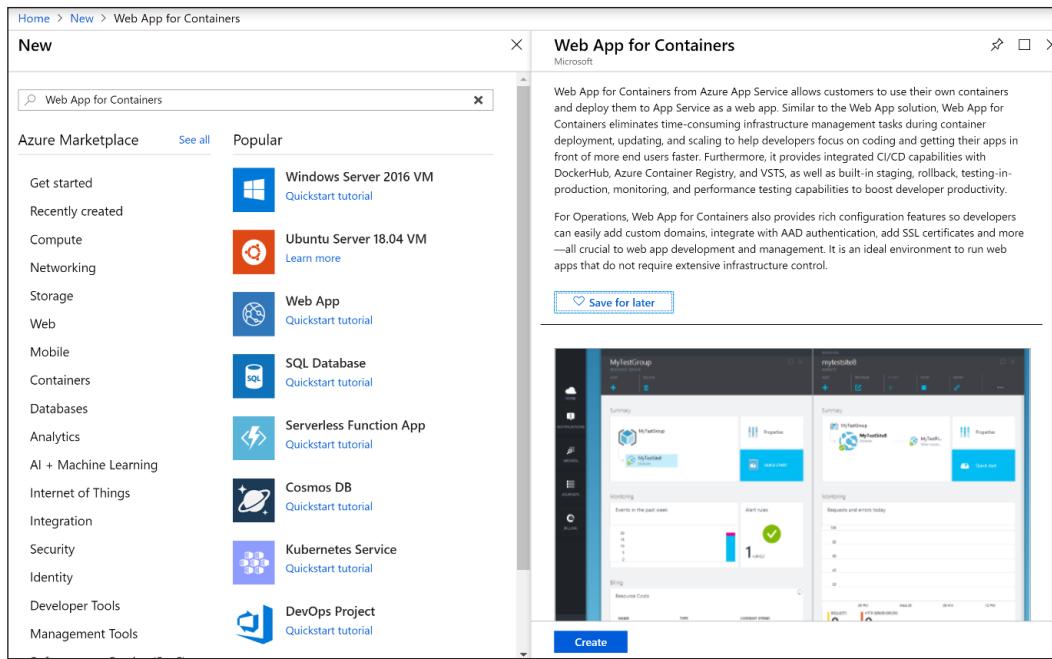
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.extensions/azure-vote-back  1/1     1           1           4m11s
deployment.extensions/azure-vote-front 1/1     1           1           4m11s

NAME                  DESIRED   CURRENT   READY   AGE
replicaset.extensions/azure-vote-back-f7c88f6f5  1         1         1         4m11s
replicaset.extensions/azure-vote-front-7c79dddc4c 1         1         1         4m11s
```

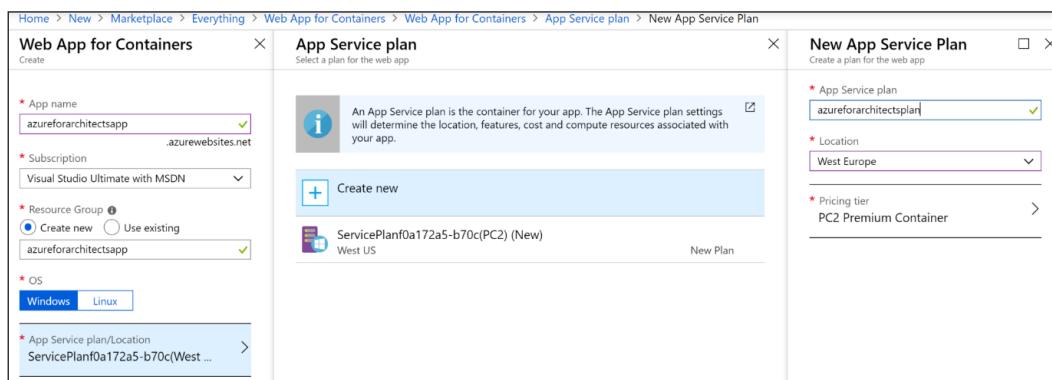
Contêineres do Serviço de App

Anteriormente, não era possível usar contêineres junto com o Serviço de Aplicativo do Azure. O Serviço de Aplicativo ofereceu a capacidade de hospedar aplicativos em máquinas virtuais. Agora, há um novo recurso de visualização que foi adicionado recentemente ao Serviço de Aplicativo do Azure. Esse recurso permite que você hospede contêineres no Serviço de Aplicativo e faça com que os contêineres contenham os binários e as dependências do aplicativo:

1. Para implantar um contêiner no Serviço de Aplicativo, vá para o portal do Azure e selecione **Aplicativo Web para Contêineres** e crie um da seguinte maneira:

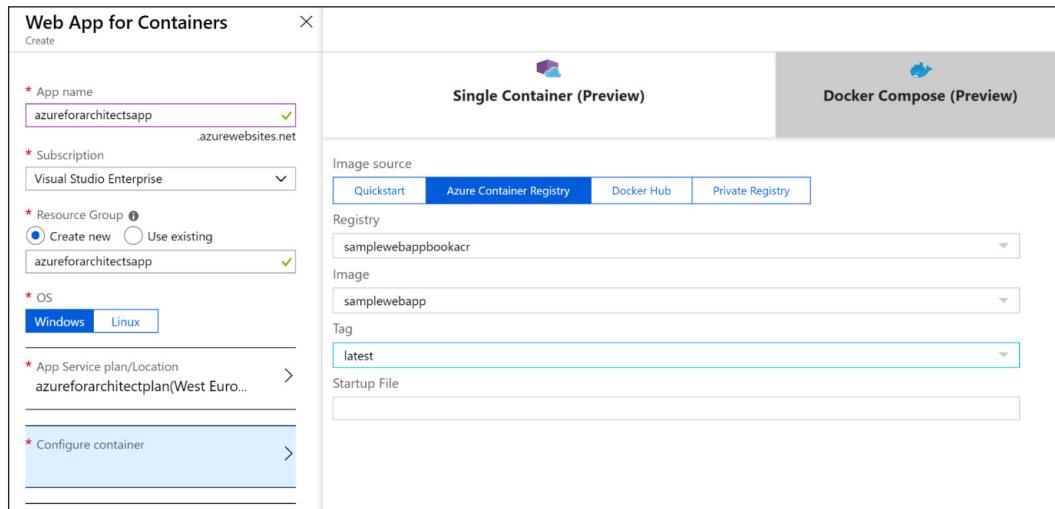


2. Forneça um nome para ele, junto com as informações de assinatura e grupo de recursos. Selecione um sistema operacional apropriado. O Linux e o Windows estão disponíveis como opções.
3. Crie um novo **plano de Serviço de Aplicativo**, conforme mostrado na captura de tela a seguir, ou use um plano existente:

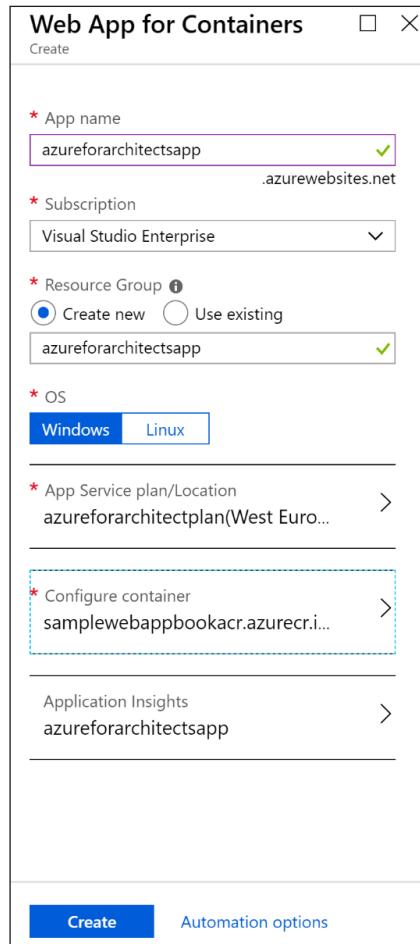


Soluções do Azure que usam Serviços de Contêiner do Azure

4. Forneça as informações de configuração **Configurar contêiner**. Aqui, já temos nossa imagem no Registro de Contêiner do Azure, que foi escolhida para este exemplo. As outras opções são o Docker Hub e os registros privados. Forneça as informações sobre o registro que hospeda a imagem. Como já fizemos upload de uma imagem para o Docker Hub, será possível usar a imagem do Docker Hub em vez do Registro de Contêiner do Azure:

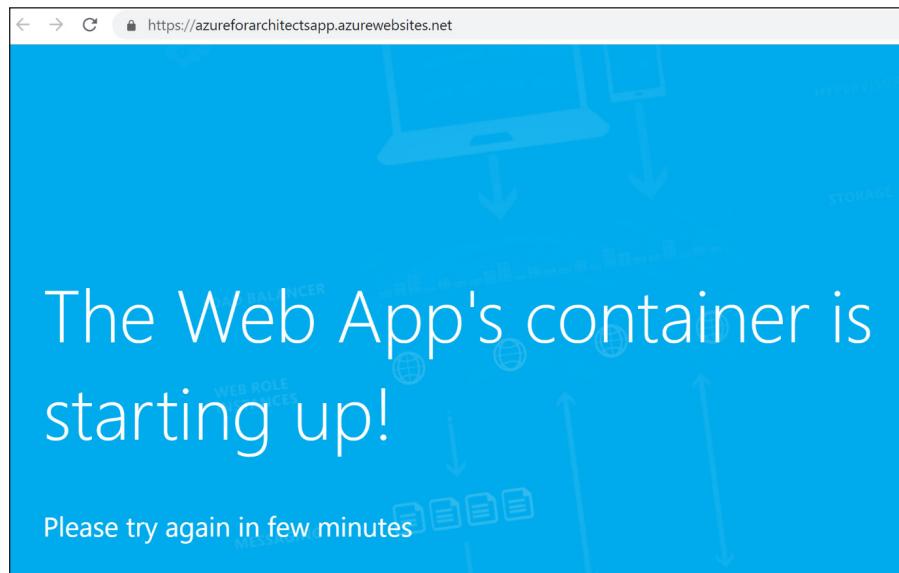


5. A configuração completa para **Aplicativo Web para Contêineres** deve ser semelhante à mostrada na captura de tela a seguir. A criação do Serviço de Aplicativo fará download da imagem do registro de contêiner e criará um novo contêiner com ela:



Soluções do Azure que usam Serviços de Contêiner do Azure

6. Demora um pouco para o contêiner entrar em funcionamento. Enquanto isso, se você navegar até a URL gerada para o Serviço de Aplicativo, ela mostrará a seguinte página:



7. Se você navegar até o menu **Configurações do contêiner** à esquerda, ele mostrará os logs de diagnóstico para o contêiner:

A screenshot of the Azure portal showing the "Container settings" page for an app service named "azureforarchitectsapp". The left sidebar has a "Settings" section with "Container settings" selected. The main area shows "Container settings" configuration options: "Image and optional tag (eg 'image:tag')" set to "azureforarchitects/samplewebapp:latest", "Startup File" empty, "Continuous Deployment" set to "Off", and a "Webhook URL" input field with a "Copy" button. Below these is a "Logs" section containing two log entries:

```
31/12/2018 10:09:57.075 INFO - Site: azureforarchitectsapp - Image: azureforarchitects/samplewebapp:latest
Custom Registry: https://index.docker.io
Status: 6f2071ddc729 Extracting
[=====] 106.4MB/182.7MB
31/12/2018 10:09:57.420 INFO - Site: azureforarchitectsapp - Image: azureforarchitects/samplewebapp:latest
Custom Registry: https://index.docker.io
Status: 6f2071ddc729 Extracting
[=====] 107MB/182.7MB
```

At the bottom of the logs are "Download" and "Refresh" buttons, and at the very bottom are "Save" and "Discard" buttons.

Comparação de todas as opções de contêiner

É hora de comparar todas as opções de hospedagem de contêiner no Azure. Esta é uma das habilidades mais cruciais que os arquitetos devem possuir para decidir o recurso ideal para suas necessidades. As opções para hospedar contêineres são as seguintes:

- Contêineres em máquinas virtuais
- Os contêineres em máquinas virtuais com Kubernetes como o orquestrador
- Serviço Azure Kubernetes
- Contêineres no Serviço de Aplicativo do Azure
- Contêineres em Instâncias de Contêiner do Azure
- Contêineres no Azure Functions
- Contêineres no Service Fabric

Dado o grande número de opções para hospedar contêineres, é importante entender sua natureza, incluindo seus custos e benefícios, antes de finalizar um recurso ou grupo de recursos. Nesta seção, vamos avaliar e comparar recursos com base no seguinte:

- Custo
- Complexidade
- Controlar
- Agilidade e flexibilidade
- Escalabilidade e disponibilidade

Contêineres em máquinas virtuais

Ao implantar contêineres em máquinas virtuais sem um orquestrador, todo o fardo de garantir que os contêineres estejam em execução fica no cliente. O cliente pode ter que comprar serviços de monitoramento adicionais que podem monitorar hosts, bem como contêineres. Isso, no entanto, também não é uma situação ideal. Isso porque, embora o monitoramento possa gerar alertas e informar os administradores, o cliente deverá gastar mais para criar automação em cima desses alertas.

Essa opção também é dispendiosa em comparação com outras, pois o custo envolverá o custo das máquinas virtuais, juntamente com o custo de outros recursos do Azure, como o Armazenamento do Azure e o Balanceador de Carga.

Para ambientes de produção e aplicação de missão crítica, esta opção não é aconselhável.

Os contêineres em máquinas virtuais com Kubernetes como o orquestrador

Ao implantar contêineres em máquinas virtuais com um orquestrador, o custo será maior em comparação à implantação sem um orquestrador devido à necessidade de máquinas virtuais adicionais para atuar como servidores de gerenciamento que hospedam o servidor de API, agendadores e controladores. Se o mestre estiver altamente disponível, então serão necessárias máquinas virtuais adicionais. O benefício dessa solução em comparação com as soluções anteriores é que os aplicativos estarão continuamente disponíveis sem tempo de inatividade, pois o Kubernetes garantirá que, se um host ou pod ficar inativo, os pods serão criados em outros hosts.

Essa opção é ideal quando uma organização quer controle total sobre o ambiente e a solução e tem pessoas qualificadas para gerenciar o ambiente. A opção também ajuda na distribuição de upgrades para o aplicativo. O dimensionamento de hosts será uma atividade manual, que pode tornar as operações complexas e demoradas, a menos que os conjuntos de dimensionamento de máquinas virtuais com regras de escalabilidade sejam usados para dimensionamento.

Serviço Azure Kubernetes

O Serviço de Kubernetes do Azure é uma **Plataforma como Serviço (PaaS)**. Isso significa que ao implantar o Serviço de Kubernetes do Azure, os clientes geralmente não acessam os nós mestres e de agente diretamente; no entanto, se necessário, eles podem fazer logon em nós de agente para fins de solução de problemas. O único acesso necessário aos clientes é para a ferramenta da CLI Kubectl, que interage com o servidor de API. Os clientes do Serviço de Kubernetes do Azure não precisam gerenciar os nós e o Azure garantirá que todas as atualizações, bem como a manutenção planejada e não planejada, serão automaticamente cuidadas. Os clientes podem implantar seus pods no cluster e podem ter certeza de que o estado desejado do ambiente de contêiner será mantido pelo Serviço de Kubernetes do Azure.

Esta solução não é complexa em comparação com as duas soluções anteriores e não requer operadores de Kubernetes altamente qualificados. A solução é altamente escalável, pois o serviço de kubernetes do Azure fornece os meios pelos quais você pode aumentar o número de nós sob demanda. Também é altamente disponível, já que o Serviço de Kubernetes do Azure garante que, se um nó ficar inativo, outro nó será disponibilizado para o cluster.

A desvantagem dessa solução é que os clientes perdem algum controle sobre a infraestrutura subjacente. Os clientes têm controle sobre a rede do Azure, a pilha de rede do Kubernetes e os nós de agente; no entanto, eles não têm acesso a nós mestres. Os nós mestres estão ocultos dos clientes. Algo importante a ser notado sobre esta solução é que ela funciona bem com imagens baseadas em Linux; no entanto, há tarefas adicionais que precisam ser executadas no caso do Windows. Além disso, os nós mestres são sempre baseados em Linux. Estas são limitações que definitivamente mudarão no futuro pois a Microsoft torna o Windows completamente nativo aos contêineres.

Esta é uma das opções e soluções mais altamente recomendadas de todas as opções disponíveis.

Contêineres no Serviço de Aplicativo do Azure

Este é um recurso relativamente novo fornecido pelo Azure e está em visualização no momento da escrita. O Serviço de Aplicativo do Azure é outro PaaS que fornece um ambiente gerenciado para implantar aplicativos. O Serviço de Aplicativo pode ser escalado automaticamente de forma horizontal e vertical em regras e condições, e os contêineres podem ser criados nesses novos nós. Os Aplicativos Web para Contêineres fornecem todos os benefícios do Serviço de Aplicativo do Azure com um benefício adicional. Ao provisionar um Serviço de Aplicativo, os clientes são vinculados pela plataforma fornecida em termos de versão do .NET e sistema operacional. Os clientes não podem controlar a versão do sistema operacional ou a versão do .NET. No entanto, ao implantar contêineres na parte superior do Serviço de Aplicativo, as imagens de contêiner podem consistir em qualquer tempo de execução, estrutura e SDK do .NET. Além disso, a imagem do contêiner pode basear-se em qualquer versão compatível do sistema operacional base. Em suma, os contêineres no Serviço de Aplicativo fornecem muito mais flexibilidade para provisionar um ambiente. Também é rentável em comparação com as opções de contêiner de IaaS; no entanto, há um controle reduzido de máquinas virtuais. Essa opção, no entanto, fornece acesso mínimo aos serviços e servidores subjacentes.

Essa opção requer que as imagens sejam armazenadas em algum registro. Pode ser o Registro de Contêiner do Azure, o Docker Hub ou qualquer registro privado. Não é possível fazer upload de um Dockerfile segmentando imagens do Windows para o Serviço de Aplicativo; no entanto, essa opção de usar o Docker Compose e o Kubernetes está disponível para contêineres do Linux.

A complexidade de criar e manter aplicativos com aplicativos Web para contêineres é consideravelmente menor em comparação a outras opções. É uma ótima solução para ambientes de demonstração e desenvolvimento/teste.

Contêineres em Instâncias de Contêiner do Azure

Isso, novamente, é um recurso relativamente novo no Azure e ajuda na hospedagem de contêineres. Ele ajuda a criar rapidamente um contêiner e hospedá-lo sem a necessidade de quaisquer máquinas virtuais. Uma das principais complexidades dessa solução é que as soluções de vários contêineres precisam de mais atenção, pois esse serviço trata cada contêiner como uma solução autônoma. Ele pode fazer download de imagens do Registro de Contêiner do Azure.

Contêineres no Azure Functions

Esse recurso é específico para executar Azure Functions dentro de contêineres. Se os clientes estiverem escrevendo e implantando funções e quiserem hospedar em qualquer ambiente que seja diferente do ambiente pronto para uso fornecido pelo Azure, essa é uma boa solução. A Microsoft já forneceu algumas imagens hospedadas no Docker Hub relacionadas a funções. Essas imagens já contêm o tempo de execução e a estrutura do Azure Functions. É importante observar que as imagens que não contêm o tempo de execução do Azure Functions não devem ser usadas para hospedar contêineres com funções.

Os benefícios e as restrições para essa solução são semelhantes aos de contêineres com aplicativos Web.

Contêineres no Service Fabric

O Service Fabric é outro recurso do Azure no qual os contêineres podem ser hospedados. O Service Fabric pode ser hospedado localmente, em máquinas virtuais do Azure e como um serviço gerenciado, fornecendo uma plataforma gerenciada.

As vantagens e desvantagens são semelhantes às das soluções IaaS e PaaS mencionadas anteriormente neste capítulo.

Resumo

Este capítulo concentrou-se em contêineres no Azure. O Azure fornece vários serviços nativos ou compatíveis com contêineres. Esses serviços são recursos relativamente novos e recursos mais recentes estão sendo continuamente adicionados a eles. Alguns dos serviços importantes do Azure relacionados a contêineres são o Registro de Contêiner do Azure, as Instâncias de Contêiner do Azure, o Serviço de Kubernetes do Azure e o Serviço de Aplicativo do Azure. Há outros serviços, como o Azure Service Fabric, que oferece suporte a hospedagem de contêineres e contêineres de aplicativo de função, que dão suporte a funções sendo executadas em contêineres. Os contêineres podem ser hospedados como IaaS ou PaaS. Ambos têm suas vantagens e desvantagens, e as organizações devem avaliar seus requisitos e escolher um serviço apropriado para seus contêineres. É importante saber qual serviço de contêiner usar em diferentes cenários, pois isso seria um fator chave no sucesso de um arquiteto no Azure.

11

Azure DevOps

O desenvolvimento de software é uma tarefa complexa composta de vários processos e ferramentas, envolvendo pessoas de diferentes departamentos. Todos precisam se unir e trabalhar de maneira coesa. Com tantas variáveis, os riscos são altos quando você entrega aos clientes finais. Uma pequena omissão ou configuração incorreta pode arruinar o aplicativo. Este capítulo é sobre a adoção e a implementação de práticas que reduzem esse risco consideravelmente e garantem que um software de alta qualidade possa ser entregue ao cliente constantemente.

Antes de analisar os detalhes do DevOps, vamos listar os problemas enfrentados pelas empresas de software que são resolvidos pelo DevOps:

- Organizações rígidas e que não aceitam mudanças
- Processos rígidos e demorados
- Equipes isoladas trabalhando em silos
- Design monolítico e implantações repentinhas
- Execução manual
- Falta de inovação

Neste capítulo, abordaremos os seguintes tópicos:

- DevOps
- Práticas de DevOps
- Azure DevOps
- Preparação de DevOps
- DevOps para soluções de PaaS
- DevOps para soluções baseadas em máquinas virtuais (IaaS)
- DevOps para soluções baseadas em contêineres (IaaS)

- Azure DevOps e Jenkins
- Automação do Azure
- Ferramentas do Azure para DevOps

DevOps

No momento, não há um consenso na indústria quanto à definição de DevOps. As organizações têm formulado sua própria definição de DevOps e tentado implementá-la. Elas têm sua própria perspectiva e acreditam que terão implementado o DevOps se implementarem o gerenciamento de automação e configuração e usarem processos ágeis.

O DevOps consiste no mecanismo de entrega de sistemas de software. Ele aproxima as pessoas, fazendo com que elas colaborem e se comuniquem, trabalhando juntas para alcançar um objetivo e uma visão comuns. Ele incentiva a assunção conjunta de responsabilidade e propriedade. Ele permite a implementação de processos que promovem a colaboração e uma mentalidade de serviço. Ele habilita os mecanismos de entrega que levam agilidade e flexibilidade à organização. Ao contrário da crença popular, o DevOps não é sobre ferramentas, tecnologia, automação. Esses são facilitadores que ajudam na colaboração, na implementação de processos ágeis e na entrega melhor e mais rápida ao cliente.

Há várias definições disponíveis na Internet para o DevOps e elas não estão erradas. O DevOps não fornece uma estrutura nem uma metodologia. É um conjunto de princípios e práticas que, quando utilizado dentro de uma organização, engajamento ou projeto, atinge o objetivo e a visão do DevOps e da organização. Esses princípios e práticas não exigem processos, ferramentas, tecnologias ou ambientes específicos. O DevOps fornece as orientações que podem ser implementadas por meio de qualquer ferramenta, tecnologia ou processo, embora parte da tecnologia e dos processos possa ser mais aplicável do que outras para alcançar a visão de princípios e práticas de DevOps.

Embora as práticas de DevOps possam ser implementadas em qualquer organização que forneça serviços e produtos para clientes, neste livro, examinaremos o DevOps da perspectiva do desenvolvimento de software e do departamento de operações de qualquer organização.

Então, o que é o DevOps? O DevOps é definido como um conjunto de princípios e práticas que reúne todas as equipes, incluindo desenvolvedores e operações desde o início do sistema de software para obter uma entrega de ponta a ponta mais rápida e eficiente desse sistema ao cliente final, de forma consistente e previsível, reduzindo o tempo de chegada ao mercado e, consequentemente, obtendo uma vantagem competitiva.

Leia em voz alta a definição anterior do DevOps. Se você observar com atenção, verá que ela não indica nem se refere a processos, ferramentas ou tecnologia específicos. Ela não prescreve qualquer metodologia ou ambiente.

O objetivo da implementação de princípios e práticas de DevOps em qualquer organização é garantir que as demandas das partes interessadas (incluindo os clientes) e as expectativas sejam atendidas com eficiência e eficácia.

As demandas e expectativas do cliente são atendidas quando o seguinte acontece:

- O cliente obtém os recursos desejados.
- O cliente obtém os recursos quando desejado.
- O cliente recebe atualizações mais rápidas sobre os recursos.
- A qualidade da entrega é alta.

Quando uma organização consegue atender a essas expectativas, os clientes ficam satisfeitos e permanecem fiéis a ela. Por sua vez, isso aumenta a competitividade de mercado da organização, que resulta em um maior valor de marca e de mercado. Isso tem um impacto direto sobre a receita e o lucro líquido da organização. Ela pode investir mais em inovação e comentários dos clientes, provocando mudanças contínuas em seus sistemas e serviços para permanecer relevante.

A implementação de princípios e práticas de DevOps em qualquer organização é guiada pelo ecossistema ao seu redor. Esse ecossistema é composto pela indústria e pelos domínios aos quais a organização pertence.

O DevOps se baseia em um conjunto de princípios e práticas. Vamos analisar os detalhes desses princípios e práticas mais adiante neste capítulo. Os princípios fundamentais do DevOps são:

- Agilidade
- Automação
- Colaboração
- Comentários

As práticas fundamentais do DevOps são:

- Integração contínua
- Gerenciamento de configuração
- Implantação contínua
- Entrega contínua
- Aprendizado contínuo

O DevOps não é um paradigma novo; no entanto, tem ganhado muita popularidade e trânsito recentemente. Sua adoção está no auge e cada vez mais empresas estão iniciando essa jornada. Eu me referi intencionalmente ao DevOps como uma jornada porque há diferentes níveis de maturidade dentro dele. Enquanto a implementação bem-sucedida de implantação e entrega contínuas é considerada como o nível mais alto de maturidade nessa jornada, a adoção do desenvolvimento de software Agile e controle de código-fonte é considerada como a primeira etapa na jornada de DevOps.

Um dos primeiros tópicos abordados pelo DevOps é a remoção de barreiras entre as equipes de desenvolvimento e de operações. Isso resulta na estreita colaboração entre várias equipes. Ela muda a mentalidade de que o desenvolvedor é responsável apenas por escrever o código e passá-lo adiante às operações para que seja implantado após passar por testes. Ela também muda a mentalidade de que as operações não desempenham nenhum papel nas atividades de desenvolvimento. As operações devem influenciar o planejamento do produto e estar cientes dos recursos que serão lançados em breve. Elas também devem continuamente fornecer comentários aos desenvolvedores sobre os problemas operacionais, de modo que eles possam ser corrigidos em versões posteriores. Elas devem influenciar o design do sistema para melhorar o funcionamento operacional desse sistema. Da mesma forma, os desenvolvedores devem ajudar a equipe de operações a implantar o sistema e resolver os incidentes quando surgirem.

A definição de DevOps fala sobre uma entrega de ponta a ponta dos sistemas às partes interessadas de forma mais rápida e eficiente. Ela não fala sobre o nível de rapidez ou eficiência que a entrega deve ter. Seu nível de rapidez depende do domínio da organização, da indústria, da segmentação de clientes e necessidades. Para algumas organizações, as versões trimestrais são suficientes, enquanto para outras elas poderiam ser semanais. Ambos são válidos de um ponto de vista de DevOps, e essas organizações podem implantar processos e tecnologias relevantes para atingir seus prazos de lançamento de destino. O DevOps não impõe qualquer período de tempo específico para CI/CD. As organizações devem identificar a melhor implementação de princípios e práticas de DevOps com base no seu projeto como um todo, no engajamento e na visão organizacional.

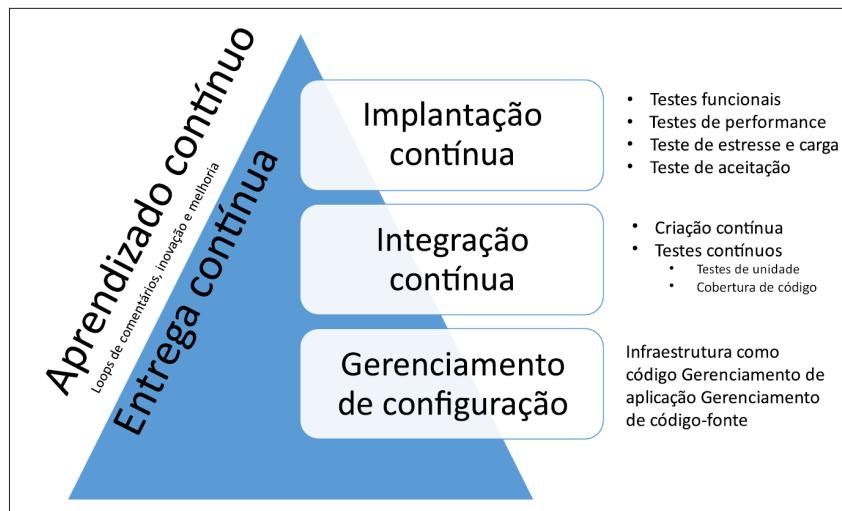
A definição também fala sobre entrega de ponta a ponta. Isso significa que tudo do planejamento e a entrega do sistema aos serviços e às operações devem fazer parte da adoção de DevOps. Os processos devem permitir maior flexibilidade, modularidade e agilidade no ciclo de vida de desenvolvimento do aplicativo. Embora as organizações sejam livres para usar o melhor processo de ajuste – Waterfall, Agile, Scrum e muito mais. Normalmente, as organizações tendem a favorecer processos ágeis com a entrega baseada em iterações. Isso permite uma entrega mais rápida em unidades menores que são muito mais testáveis e gerenciáveis em comparação a uma grande entrega.

O DevOps fala repetidamente sobre clientes finais de forma consistente e previsível. Isso significa que as organizações devem continuamente entregar aos clientes recursos novos e atualizados usando a automação. Não podemos alcançar consistência e previsibilidade sem o uso da automação. O trabalho manual não deve existir para garantir um alto nível de consistência e previsibilidade. A automação também deve ser de ponta a ponta para evitar falhas. Isso também indica que o design do sistema deve ser modular, permitindo entregas mais rápidas em sistemas confiáveis, disponíveis e escaláveis. Os testes desempenham um papel importante em entregas consistentes e previsíveis.

O resultado final da implementação dessas práticas e princípios mencionados anteriormente é que a organização é capaz de atender às expectativas e demandas dos clientes. Ela é capaz de crescer mais rápido do que a concorrência e aumentar ainda mais a qualidade e a capacidade de seus produtos e serviços por meio de inovação e melhoria contínuas.

Práticas de DevOps

O DevOps consiste em várias práticas, cada uma fornecendo funcionalidades distintas ao processo como um todo. O diagrama a seguir mostra a relação entre elas. O gerenciamento de configuração, a integração contínua e a implantação contínua formam as práticas fundamentais que habilitam o DevOps. Quando fornecemos serviços de software que combinam esses três serviços, alcançamos a entrega contínua. A entrega contínua é uma capacidade e nível de maturidade de uma organização dependente da maturidade de gerenciamento de configuração, integração contínua e implantação contínua. Comentários contínuos em todos os estágios constituem o loop de comentários que ajuda a fornecer serviços melhores aos clientes. Ele é executado em todas as práticas de DevOps. Vamos nos aprofundar em cada um desses recursos e práticas de DevOps:



Gerenciamento de configuração

Serviços e aplicativos de negócios precisam de um ambiente no qual podem ser implantados. Normalmente, o ambiente é uma infraestrutura composta de vários servidores, computadores, rede, armazenamento, contêineres e muitos outros serviços funcionando juntos de modo que os aplicativos de negócios possam ser implantados sobre eles. Aplicativos de negócios são decompostos em vários serviços executados em vários servidores, na infraestrutura local ou na nuvem, e cada serviço tem sua própria configuração juntamente com requisitos relacionados à configuração da infraestrutura. Em suma, o aplicativo e a infraestrutura são necessários para fornecer sistemas aos clientes e cada um tem sua própria configuração. Se houver desvio na configuração, o aplicativo pode não funcionar conforme o esperado, resultando em tempo de inatividade e falhas. Além disso, como os processos de ALM determinam o uso de vários estágios e ambientes, um aplicativo seria implantado em vários ambientes com diferentes configurações. O aplicativo será implantado no ambiente de desenvolvimento para que os desenvolvedores vejam o resultado do seu trabalho. Ele será implantado em vários ambientes de teste com diferentes configurações para testes funcionais, de carga e estresse, de performance, de integração e muito mais. Também será implantado no ambiente de pré-produção para realizar testes de aceitação do usuário e, por fim, no ambiente de produção. É importante que um aplicativo possa ser implantado em vários ambientes sem alterações manuais na sua configuração.

O gerenciamento de configuração fornece um conjunto de processos e ferramentas para garantir que cada ambiente e aplicativo tenha sua própria configuração. Ele acompanha itens de configuração, e qualquer coisa que mude de um ambiente para outro deve ser tratada como um item de configuração. Ele também define as relações entre os itens de configuração e como as alterações em um desses itens afetará o outro.

O gerenciamento de configuração ajuda nos seguintes locais:

- **Infraestrutura como Código:** quando o processo de provisionamento da infraestrutura e sua configuração são representados por meio de código, e o mesmo código passa pelo processo de ciclo de vida do aplicativo, isso é conhecido como **Infraestrutura como Código (IaC)**. O IaC ajuda a automatizar o provisionamento e a configuração de infraestrutura. Ela também representa toda a infraestrutura em código que pode ser armazenado em um repositório e submetido ao controle de versão. Isso permite que os usuários empreguem as configurações do ambiente anterior quando necessário. Também permite o provisionamento de um ambiente várias vezes de forma consistente e previsível. Todos os ambientes provisionados dessa forma são consistentes e iguais em todos os estágios de ALM.

- **Implantação e configuração do aplicativo:** a implantação de um aplicativo e sua configuração são a etapa seguinte ao provisionamento da infraestrutura. A implantação de um pacote webdeploy em um servidor, a implantação de um esquema e dados de um SQL Server (bacpac) em outro servidor, e a alteração da cadeia de conexão do SQL no servidor Web para representar o SQL Server apropriado são exemplos de implantação e configuração do aplicativo. O gerenciamento de configuração armazena valores da configuração do aplicativo para cada ambiente no qual ele é implantado.

A configuração aplicada também deve ser monitorada. A configuração esperada e desejada deve ser mantida consistentemente. Qualquer desvio dessa configuração esperada e desejada tornará o aplicativo indisponível. O gerenciamento de configuração também é capaz de encontrar o desvio e reconfigurar o aplicativo e o ambiente para o seu estado desejado.

Com o gerenciamento de configuração automatizado em vigor, ninguém na equipe precisa implantar e configurar os ambientes e aplicativos em produção. A equipe de operações não depende da equipe de desenvolvimento ou de uma longa documentação de implantação.

Outro aspecto do gerenciamento de configuração é o controle do código-fonte. Serviços e aplicativos de negócios englobam o código e outros artefatos. Vários membros da equipe trabalham nos mesmos arquivos. O código-fonte deve estar sempre atualizado e ser acessado somente por membros autenticados da equipe. O código e outros artefatos isolados são itens de configuração. O controle do código-fonte ajuda na colaboração e comunicação dentro da equipe, pois todos estão cientes do que todos estão fazendo e os conflitos são resolvidos em um estágio inicial.

O gerenciamento de configuração pode ser amplamente dividido em duas categorias:

- Dentro da máquina virtual
- Fora da máquina virtual

As ferramentas disponíveis para o gerenciamento de configuração dentro da máquina virtual são discutidas a seguir.

Configuração de Estado Desejado

A **Configuração de Estado Desejado** (DSC) é uma nova plataforma de gerenciamento de configuração da Microsoft criada como uma extensão do PowerShell. O DSC foi originalmente lançado como parte do **Windows Management Framework** (WMF) 4.0. Ela está disponível como parte do WMF 4.0 e 5.0 para todos os sistemas operacionais Windows Server anteriores ao Windows 2008 R2. O WMF 5.1 está disponível pronto para uso no Windows Server 2016 e no Windows 10.

Chef, Puppet e Ansible

Além da DSC, há uma série de ferramentas de gerenciamento de configuração, como Chef, Puppet e Ansible com suporte do Azure. Detalhes sobre essas ferramentas não são abordados neste livro.

As ferramentas disponíveis para gerenciamento de configuração fora de uma máquina virtual são:

Modelos do ARM

Os modelos do ARM são os principais meios de provisionamento de recursos no ARM. Eles fornecem um modelo declarativo por meio do qual recursos, suas configurações, scripts e extensões são especificados. Os modelos do ARM se baseiam no formato **JavaScript Object Notation (JSON)**. Eles usam as convenções e a sintaxe JSON para declarar e configurar recursos. Os arquivos JSON são baseados em texto, fáceis de usar e de ler. Eles podem ser armazenados em um repositório de código-fonte e submetidos ao controle de versão. Eles também servem para representar a infraestrutura como código que pode ser usada para provisionar recursos em grupos de recursos do Azure de forma constante, previsível, consistente e uniforme. Um modelo precisa de um grupo de recursos para implantação. Ele só pode ser implantado em um grupo de recursos, que deve existir antes de executar a implantação do modelo. Um modelo não é capaz de criar um grupo de recursos.

Os modelos fornecem a flexibilidade de serem genéricos e modulares no seu design e na sua implementação. Os modelos fornecem a capacidade de aceitar parâmetros de usuários, declarar variáveis internas, ajudar na definição de dependências entre recursos, vincular recursos dentro de grupos de recursos iguais ou diferentes, além de executar outros modelos. Eles também fornecem expressões e funções do tipo de linguagem de script que os tornam dinâmicos e personalizáveis no tempo de execução.

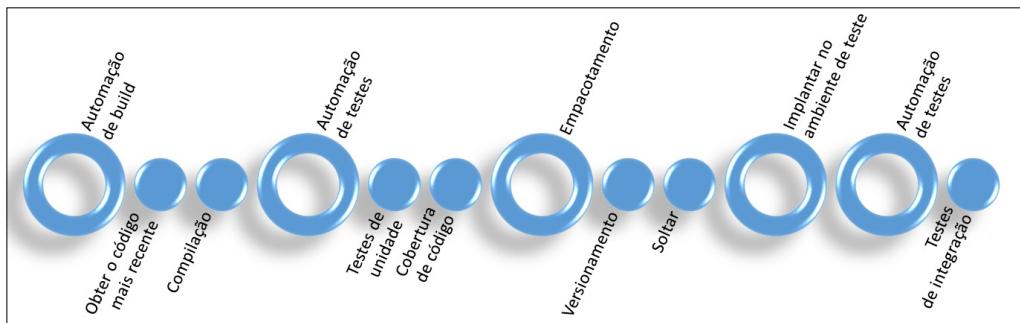
Integração contínua

Vários desenvolvedores escrevem código que eventualmente é armazenado em um repositório comum. O código normalmente é verificado ou enviado ao repositório quando o desenvolvedor termina de desenvolver seu recurso. Isso pode ocorrer em um dia, ou pode levar dias ou semanas. Alguns dos desenvolvedores talvez estejam trabalhando no mesmo recurso e também seguindo as mesmas práticas de envio/verificação de código em dias ou semanas. Isso pode criar problemas para a qualidade do código. Um dos princípios do DevOps é falhar rápido. Os desenvolvedores devem verificar/enviar o código ao repositório frequentemente e compilá-lo para verificar se não introduziram bugs e se ele é compatível com o código escrito por seus colegas. Se o desenvolvedor não seguir essa prática, o código em sua máquina ficará muito grande e será difícil integrá-lo ao código de outro colega. Além disso, se o build falhar, será difícil e demorado corrigir os problemas decorrentes disso.

A integração contínua resolve esses tipos de desafios. Ela ajuda no build e validação do código enviado/verificado por um desenvolvedor por meio de uma série de etapas de validação. A integração contínua cria um fluxo de processo que consiste em várias etapas. A integração contínua é composta por compilações automatizadas contínuas e testes automatizados contínuos. Normalmente, a primeira etapa é o build do código. Após o build bem-sucedida, cada etapa é responsável por validar o código de uma perspectiva específica. Por exemplo, os testes de unidade podem ser executados no código compilado e a cobertura de código pode ser executada para verificar quais caminhos de código são executados pelos testes de unidade. Eles podem revelar se testes de unidade abrangentes estão escritos ou se há espaço para adicionar mais testes de unidade. O resultado final da integração contínua são pacotes de implantação que podem ser usados pela implantação contínua para implantá-los em vários ambientes.

Os desenvolvedores são incentivados a verificar seu código várias vezes por dia, em vez de fazer isso após dias ou semanas. A integração contínua inicia a execução do pipeline inteiro logo após a verificação ou o envio do código. Se o build for bem-sucedido, os testes de código e outras atividades que fazem parte do pipeline serão executados sem erros, o código será implantado em um ambiente de teste, e testes de integração serão executados nele. Embora cada sistema exija sua própria configuração de integração contínua, um pequeno exemplo dela é mostrado no diagrama a seguir.

A integração contínua aumenta a produtividade dos desenvolvedores. Eles não precisam compilar seu código manualmente, executar vários tipos de testes, um após o outro, e criar pacotes a partir deles. Ela também reduz o risco de introdução de bugs no código, o qual não fica obsoleto. Ela fornece comentários precoces aos desenvolvedores sobre a qualidade do seu código. Em geral, a qualidade dos produtos finais é alta e eles são entregues com mais rapidez por meio da adoção de práticas de integração contínua. Um pipeline de integração contínua de exemplo é mostrado aqui:



Automação de build

A automação de build consiste em várias tarefas executadas em sequência. Geralmente, a primeira tarefa é responsável pela obtenção do código-fonte mais recente do repositório. O código-fonte pode incluir vários projetos e arquivos. Eles são compilados para gerar artefatos, como executáveis, bibliotecas de vínculo dinâmico e assemblies. A automação de build bem-sucedida revela que não há erros de tempo de build no código.

Pode haver mais etapas na automação de build dependendo da natureza e do tipo de projeto.

Automação de testes

A automação de testes consiste em tarefas que são responsáveis por validar diferentes aspectos do código. Essas tarefas têm como objetivo testar o código de uma perspectiva diferente e são executadas em sequência. Geralmente, a primeira etapa é a execução de uma série de testes de unidade no código. Os testes de unidade se referem ao processo de testes da menor denominação de um recurso, validando seu comportamento isolado de outros recursos. Eles podem ser manuais ou automatizados, no entanto, a preferência é por testes de unidade automatizados.

A cobertura de código é outro tipo de teste automatizado que pode ser realizado no código para descobrir que porção dele é executada durante os testes de unidade. Ela geralmente é representada como uma porcentagem e se refere à porção de código testável por meio de testes de unidade. Se a cobertura de código não for próxima a 100%, é porque o desenvolvedor não criou testes de unidade para esse comportamento ou o código não coberto não é necessário.

A execução bem-sucedida da automação de testes que não resultar em nenhuma falha de código significativa deverá iniciar a execução das tarefas de empacotamento. Pode haver mais etapas na automação de testes dependendo da natureza e do tipo de projeto.

Empacotamento

O empacotamento se refere ao processo de geração de artefatos implantáveis, como pacotes MSI, NuGet e webdeploy, pacotes de bancos de dados, seu versionamento e armazenamento em um local onde possam ser consumidos por outros pipelines e processos.

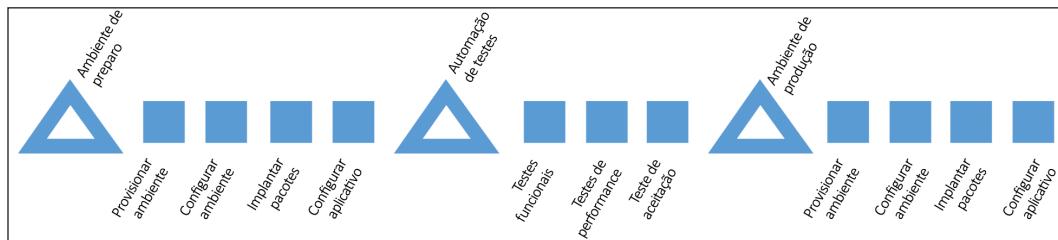
Implantação contínua

Quando o processo chegar na implantação contínua, a integração contínua terá garantido que tenhamos bits totalmente funcionais de um aplicativo que agora poderá passar por diferentes atividades de implantação contínua. A implantação contínua se refere à capacidade de implantação de serviços e aplicativos de negócios a ambientes de pré-produção e produção por meio da automação. Por exemplo, a implantação contínua pode provisionar e configurar o ambiente de pré-produção, implantar aplicativos nele e configurá-los. Após a realização de várias validações, como testes funcionais e de performance no ambiente de pré-produção, o ambiente de produção é provisionado, configurado e o aplicativo é implantado em ambientes de produção por meio da automação. Não há etapas manuais no processo de implantação. Todas as tarefas de implantação são automatizadas. A implantação contínua pode provisionar o ambiente e implantar o aplicativo a partir do zero, enquanto ele pode simplesmente implantar as alterações delta no ambiente existente se o ambiente já existir.

Todos os ambientes são provisionados por meio de automação usando a IaC. Isso garante que todos os ambientes, sejam eles de desenvolvimento, teste, pré-produção ou produção, sejam iguais. Da mesma forma, o aplicativo é implantado por meio da automação, garantindo que também seja implantado uniformemente em todos os ambientes. A configuração nesses ambientes pode ser diferente para o aplicativo.

A implantação contínua geralmente está assimilada à integração contínua. Quando a integração contínua conclui seu trabalho gerando os pacotes implantáveis finais, a implantação contínua entra em ação e inicia seu próprio pipeline. Esse pipeline é chamado de pipeline de lançamento. O pipeline de lançamento consiste em vários ambientes e cada um deles consiste em tarefas responsáveis por provisionar e configurar o ambiente, implantar e configurar aplicativos, executar validação operacional em ambientes e testar o aplicativo em vários ambientes.

Empregar a implantação contínua fornece benefícios imensos. Há um alto nível de confiança no processo de implantação como um todo que ajuda em lançamentos mais rápidos e livres de riscos na produção. As chances de algo dar errado são drasticamente reduzidas. A equipe terá níveis de estresse mais baixos e a reversão para o ambiente de trabalho anterior é possível se houver problemas na versão atual:



Embora cada sistema exija sua própria configuração de pipeline de lançamento, um pequeno exemplo da implantação contínua é mostrado no diagrama anterior. É importante observar que geralmente o provisionamento e a configuração de vários ambientes fazem parte do pipeline de lançamento e aprovações devem ser obtidas antes de passar para o próximo ambiente. O processo de aprovação pode ser manual ou automatizado dependendo da maturidade da organização.

Implantação do ambiente de teste

O pipeline de lançamento é iniciado quando o depósito da integração contínua fica disponível e o primeiro passo é obter todos os artefatos do depósito. Depois disso, ele pode criar um ambiente de teste bare-metal completamente novo ou reutilizar um já existente. Novamente, isso depende do tipo de projeto e da natureza dos testes planejados para esse ambiente. O ambiente é provisionado e configurado. Os artefatos do aplicativo são implantados e configurados.

Automação de testes

Após a implantação de um aplicativo, uma série de testes pode ser realizada no ambiente. Um dos testes realizados aqui é um teste funcional. Os testes funcionais são destinados principalmente a validar a plenitude do recurso e a funcionalidade do aplicativo. Esses testes são criados a partir de requisitos do cliente. Outro conjunto de testes que pode ser realizado está relacionado a escalabilidade e disponibilidade do aplicativo. Isso normalmente inclui testes de carga, de estresse e de performance. Também deve incluir uma validação operacional do ambiente de infraestrutura.

Implantação do ambiente de preparo

Ela é muito semelhante à implantação do ambiente de teste, a única diferença é que os valores de configuração para o ambiente e o aplicativo são diferentes.

Testes de aceitação

Os testes de aceitação geralmente são conduzidos pelas partes interessadas do aplicativo e podem ser manuais ou automatizados. Essa etapa é uma validação do ponto de vista do cliente sobre a exatidão e a plenitude da funcionalidade do aplicativo.

Implantação na produção

Depois que o cliente fornece sua aprovação, as mesmas etapas de implantação dos ambientes de teste e preparo são executadas, a única diferença é que os valores de configuração para o ambiente e o aplicativo são específicos ao ambiente de produção. Uma validação é realizada após a implantação para garantir que o aplicativo está funcionando conforme o esperado.

Entrega contínua

A entrega contínua e a implantação contínua podem parecer semelhantes para muitos leitores, no entanto, não são a mesma coisa. Enquanto a implantação contínua fala sobre a implantação em vários ambientes e, por fim, no ambiente de produção por meio da automação, as práticas de entrega contínua consistem na capacidade de gerar pacotes de aplicativos de forma que sejam prontamente implantáveis em qualquer ambiente. Para gerar artefatos prontamente implantáveis, a integração contínua deve ser usada para gerar os artefatos do aplicativo e um ambiente novo ou existente deve ser usado para implantar esses artefatos e conduzir testes funcionais, testes de performance e testes de aceitação do usuário por meio da automação. Depois que essas atividades forem executadas com êxito e sem erros, o pacote do aplicativo será considerado prontamente implantável. A entrega contínua inclui a integração e implantação contínua em um ambiente para validações finais. Ele ajuda a obter feedback mais rapidamente de ambas as operações e o usuário final. Esses comentários podem ser usados para implementar iterações subsequentes.

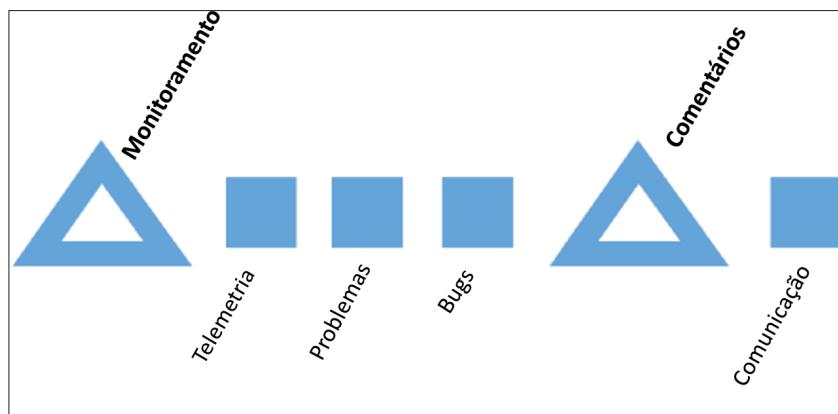
Aprendizado contínuo

Com todas as práticas de DevOps mencionadas anteriormente, é possível criar ótimos aplicativos de negócios e implantá-los automaticamente no ambiente de produção, no entanto, os benefícios do DevOps não serão duradouros se os princípios de comentários e melhoria contínuos não estiverem em vigor. É extremamente importante que comentários em tempo real sobre o comportamento do aplicativo, provenientes dos usuários finais e da equipe de operações, sejam repassados à equipe de desenvolvimento.

Os comentários devem ser repassados às equipes, fornecendo informações relevantes sobre o que está dando certo e, mais importante, o que não está dando certo.

A arquitetura e o design de um aplicativo devem ser criados com o monitoramento, a auditoria e a telemetria em mente. A equipe de operações deve coletar as informações de telemetria do ambiente de produção, capturar os bugs e problemas, e repassá-los à equipe de desenvolvimento para que possam ser corrigidos em versões subsequentes.

O aprendizado contínuo ajuda a tornar o aplicativo robusto e resistente a falha. Ele ajuda a garantir que o aplicativo atenda aos requisitos dos consumidores. O diagrama a seguir mostra o loop de feedback que deve ser implementado entre diferentes equipes:



Azure DevOps

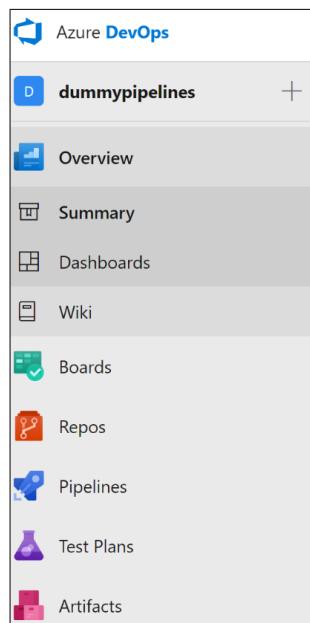
Agora, vamos nos concentrar em outro serviço online revolucionário, o Visual Studio Team Services (VSTS), que permite a integração, a implantação e a entrega contínuas com perfeição: Azure DevOps. Na verdade, seria mais apropriado chamá-lo de um conjunto de serviços disponíveis sob um único nome. O Azure DevOps é uma PaaS fornecida pela Microsoft e hospedada na nuvem. O mesmo serviço está disponível como **Team Foundation Server (TFS)** na infraestrutura local. Todos os exemplos mostrados neste livro usam o Azure DevOps.

De acordo com a Microsoft, o Azure DevOps é uma plataforma de colaboração baseada na nuvem que ajuda as equipes a compartilhar código, acompanhar o trabalho e fornecer software. O Azure DevOps é um novo nome; anteriormente, era conhecido como **Visual Studio Team Services (VSTS)**. O Azure DevOps é um serviço e uma ferramenta de desenvolvimento de software empresarial que possibilita que as organizações fornecam recursos de automação ao seu processo de gerenciamento do ciclo de vida do aplicativo de ponta a ponta, desde o planejamento até a implantação de aplicativos e o recebimento de comentários de sistemas de software em tempo real. Isso aumenta a maturidade e a capacidade de uma organização fornecer sistemas de software de alta qualidade aos seus clientes.

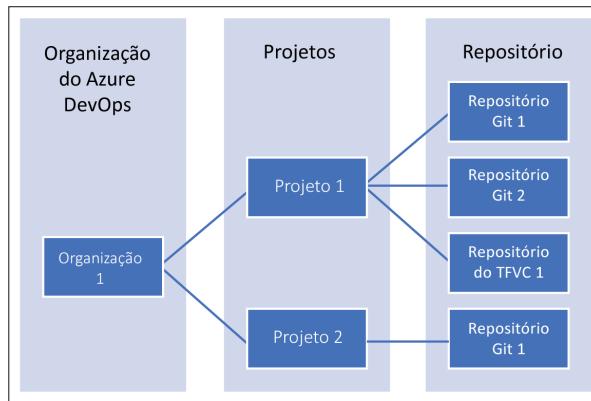
O fornecimento bem-sucedido de software envolve a união eficiente de muitos processos e atividades. Entre eles estão a execução e a implementação de vários processos Agile, o aumento da colaboração entre as equipes, a transição perfeita e automática de artefatos de uma fase de ALM para outra, e implantações em vários ambientes. É importante acompanhar e relatar essas atividades para medir, agir e aprimorar os processos de entrega. O Azure DevOps torna isso simples e fácil. Ele fornece todo um conjunto de serviços que possibilita o seguinte:

- Colaboração entre todos os membros da equipe, fornecendo uma única interface para o gerenciamento de todo o ciclo de vida do aplicativo.
- Colaboração entre equipes de desenvolvimento usando serviços de gerenciamento de código-fonte.
- Colaboração entre equipes de teste usando serviços de gerenciamento de testes.
- Validação automática de código e empacotamento por meio da integração contínua, usando serviços de gerenciamento de builds.
- Validação automática de funcionalidade do aplicativo, implantação e configuração de vários ambientes por meio de implantação e entrega contínuas usando serviços de gerenciamento de versões.
- Acompanhamento e gerenciamento de itens de trabalho usando serviços de gerenciamento de trabalhos.

A captura de tela a seguir mostra todos os serviços disponíveis para um projeto da barra de navegação esquerda do **Azure DevOps**:



Uma organização no Azure DevOps é um limite de segurança e um contentor lógico que fornece todos os serviços necessários para implementar uma estratégia de DevOps. O Azure DevOps permite a criação de vários projetos em uma única organização. Por padrão, um repositório é criado juntamente com o projeto, no entanto, o Azure DevOps permite a criação de repositórios adicionais dentro de um mesmo projeto. A relação entre a **Organização do Azure DevOps**, **Projetos** e **repositório** é mostrada no diagrama a seguir:



O Azure DevOps fornece dois tipos de repositório:

- Git
- **Controle de Versão do Team Foundation (TFVC)**

Ele também fornece a flexibilidade de escolher o repositório de controle de código-fonte do Git ou do TFVC. Pode haver uma combinação de repositórios do TFS e do TFVC disponíveis em um único projeto.

TFVC

O TFVC é a maneira tradicional e centralizada de implementar o controle de versão na qual há um repositório central e os desenvolvedores trabalham nele diretamente no modo conectado para verificar suas alterações. Se o repositório central estiver offline ou não disponível, os desenvolvedores não poderão verificar seu código e terão que esperar que ele fique online e disponível. Outros desenvolvedores poderão ver somente o código verificado. Os desenvolvedores podem agrupar várias alterações em um único conjunto a fim de verificar alterações de código que são logicamente agrupadas para formar uma única alteração. O TFVC bloqueia os arquivos de código que estão passando por edições. Outros desenvolvedores poderão ler o arquivo bloqueado, mas não poderão editá-lo. Eles deverão aguardar a conclusão da edição anterior e liberar o bloqueio para poder editá-lo. O histórico de verificações e alterações é mantido no repositório central, enquanto os desenvolvedores possuem a cópia funcional dos arquivos, mas não o histórico.

O TFVC funciona muito bem com equipes grandes que estão trabalhando nos mesmos projetos. Isso permite o controle do código-fonte em um local central. Ele também funciona melhor quando projetos têm uma longa duração, pois o histórico é gerenciado em um local central. O TFVC não tem problemas para trabalhar com arquivos grandes e binários.

Git

O Git por outro lado é uma maneira moderna e distribuída de implementar o controle de versão na qual os desenvolvedores podem trabalhar em suas próprias cópias locais de código e histórico no modo offline. Os desenvolvedores podem trabalhar offline no seu clone de código local. Cada desenvolvedor tem uma cópia local do código e de todo o histórico, e eles trabalham nas suas alterações com esse repositório local. Eles podem confirmar o código no repositório local. Podem se conectar ao repositório central para a sincronização do repositório local conforme necessário. Isso permite que todos os desenvolvedores trabalhem em qualquer arquivo, pois eles trabalhariam na sua cópia local. A ramificação no Git não cria outra cópia do código original e sua criação é extremamente rápida.

O Git funciona bem com equipes pequenas e grandes. A ramificação e a mesclagem são tranquilas com opções avançadas do Git.

O Git é a maneira recomendada de usar o controle de código-fonte devido à funcionalidade avançada que oferece. Usaremos o Git como o repositório para nosso aplicativo de exemplo neste livro.

Preparação para DevOps

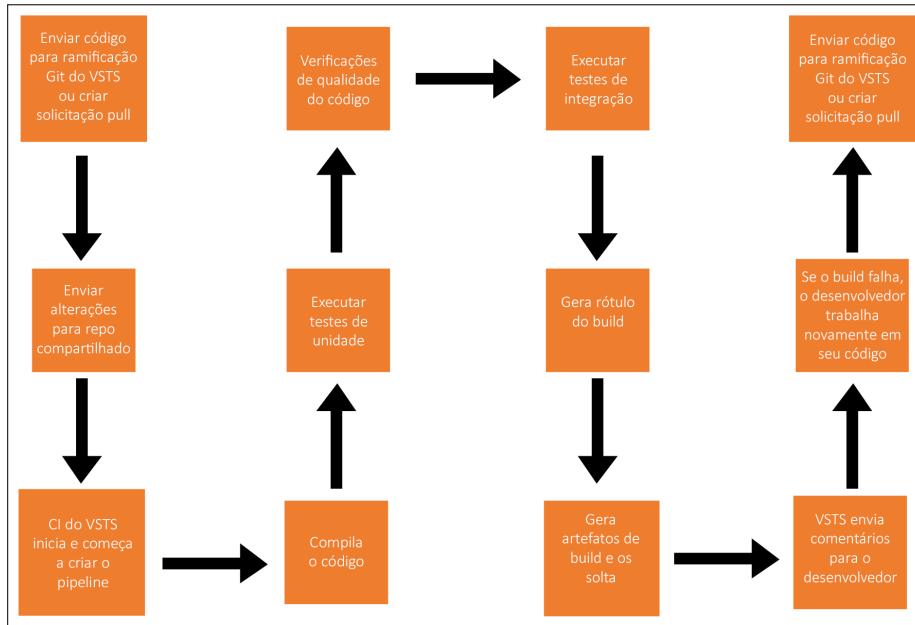
Seguindo adiante, nosso foco será na automação de implantações e processos usando diferentes padrões no Azure. Entre eles estão:

- DevOps para soluções de IaaS
- DevOps para soluções de PaaS
- DevOps para soluções baseadas em contêiner

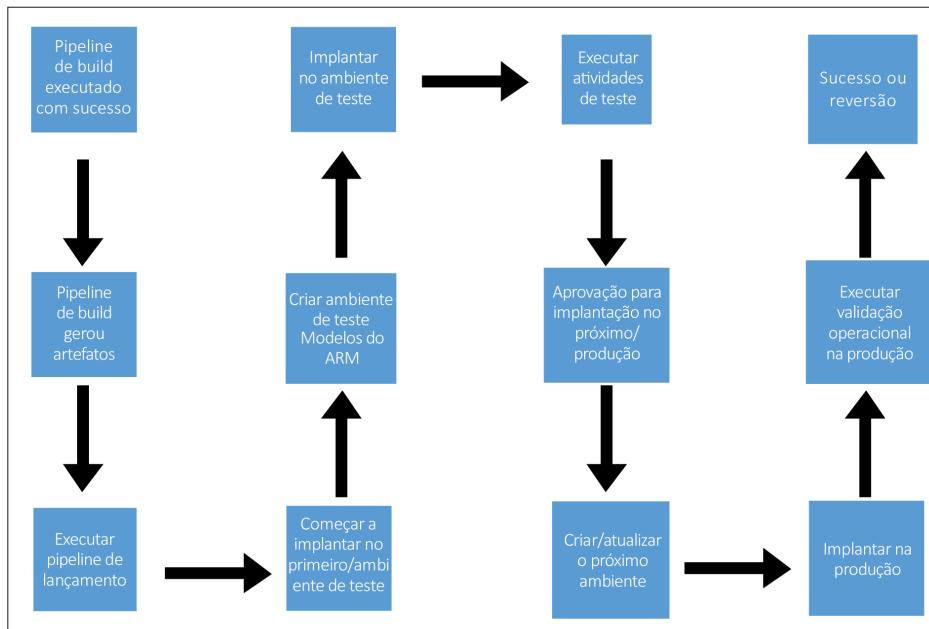
Geralmente, há serviços comuns compartilhados não exclusivos a nenhum aplicativo. Seus serviços são consumidos por vários aplicativos de diferentes ambientes, como desenvolvimento, teste e produção. O ciclo de vida desses serviços comuns compartilhados é diferente para cada aplicativo. Portanto, eles têm diferentes repositórios de controle de versão, uma base de código diferente e gerenciamento de lançamento e build. Eles têm seu próprio ciclo de planejamento, design, criação, teste e lançamento.

Os recursos que fazem parte desse grupo são provisionados usando modelos do ARM, o PowerShell e configurações de DSC.

O fluxo geral para a criação desses componentes comuns é mostrado aqui:



O processo de lançamento é mostrado no diagrama a seguir:



Na jornada de DevOps, é importante compreender e provisionar os serviços e componentes comuns antes de iniciar qualquer engajamento de software, produto ou serviço.

Provisionamento de organização do Azure DevOps

Um sistema de controle de versão é necessário para colaborar no nível do código. O Azure DevOps fornece versões centralizadas e descentralizadas de sistemas de controle. O Azure DevOps também fornece serviços de orquestração para a criação e a execução de pipelines de build e de lançamento. É uma plataforma madura que organiza todo controle de versão relacionado ao DevOps e cria e lança artefatos relacionados a itens de trabalho. Depois que uma organização for provisionada no Azure DevOps, um projeto do Azure DevOps deverá ser criado para armazenar todos os artefatos relacionados ao projeto.

Uma organização do Azure DevOps pode ser provisionada visitando <https://dev.azure.com>.

Provisionamento do Azure Key Vault

Não é aconselhável armazenar segredos, certificados, credenciais ou outras informações confidenciais em arquivos de configuração de código, bancos de dados, ou qualquer outro sistema de armazenamento geral. É aconselhável armazenar esses dados importantes em um cofre projetado especificamente para armazenar segredos e credenciais. O Azure Key Vault fornece esse serviço. O Azure Key Vault está disponível como um recurso e serviço do Azure.

Provisionamento de um servidor/serviço de gerenciamento de configuração

Um servidor/serviço de gerenciamento de configuração que fornece armazenamento a elas e aplica essas configurações a diferentes ambientes é sempre uma boa estratégia para automatizar implantações. DSC em máquinas virtuais personalizadas, DSC da Automação do Azure, Chef, Puppet e Ansible são algumas das opções e podem ser usados no Azure perfeitamente para ambientes Windows e Linux. Este livro usa o DSC como uma ferramenta de gerenciamento de configuração para todos os fins e fornece um servidor de pull que mantém todos os documentos de configuração (arquivos MOF) para o aplicativo de exemplo. Ele também mantém o banco de dados de todos os contêineres e máquinas virtuais que estão configurados e registrados no servidor de pull para obter documentos de configuração dele. O gerenciador de configurações local nesses contêineres e máquinas virtuais de destino verifica periodicamente a disponibilidade de novas configurações, bem como desvios na configuração atual e os relata de volta ao servidor de pull. Ele também possui recursos internos de relatório que fornecem informações sobre nós que estão em conformidade, bem como aqueles que não estão em conformidade em uma máquina virtual. Um servidor de pull é um aplicativo Web geral que hospeda o ponto de extremidade do servidor de pull do DSC.

Provisionamento do Log Analytics

O Log Analytics é um serviço de auditoria e monitoramento fornecido pelo Azure para obter informações em tempo real sobre todas as alterações, os desvios e eventos que ocorrem dentro de máquinas virtuais e contêineres. Ele fornece um painel e espaço de trabalho centralizado para que administradores de TI possam exibir, pesquisar e realizar pesquisas detalhadas em todas as alterações, os desvios e eventos que ocorrerem nessas máquinas virtuais. Ele também fornece agentes que são implantados em máquinas virtuais e contêineres de destino. Depois de implantados, esses agentes começam a enviar todas as alterações, os eventos e desvios para o espaço de trabalho centralizado.

Conta de Armazenamento do Azure

O Armazenamento do Azure é um serviço fornecido pelo Azure para armazenar arquivos como blobs. Todos os scripts e códigos para automatizar o provisionamento, a implantação e a configuração da infraestrutura e do aplicativo de exemplo são armazenados no repositório Git do Azure DevOps e são empacotados e implantados em uma conta de Armazenamento do Azure. O Azure fornece recursos de extensão de scripts do PowerShell que podem automaticamente fazer download de scripts do DSC e do PowerShell, e executá-los em máquinas virtuais durante a execução de modelos do Azure Resource Manager. Esse armazenamento funciona como um armazenamento comum em todas as implantações para vários aplicativos.

Imagens de origem

As imagens de máquinas virtuais e de contêineres devem ser criadas como parte do pipeline de lançamento e de build de serviços comuns. Ferramentas como Packer e Build do Docker podem ser usadas para gerar essas imagens.

Ferramentas de monitoramento

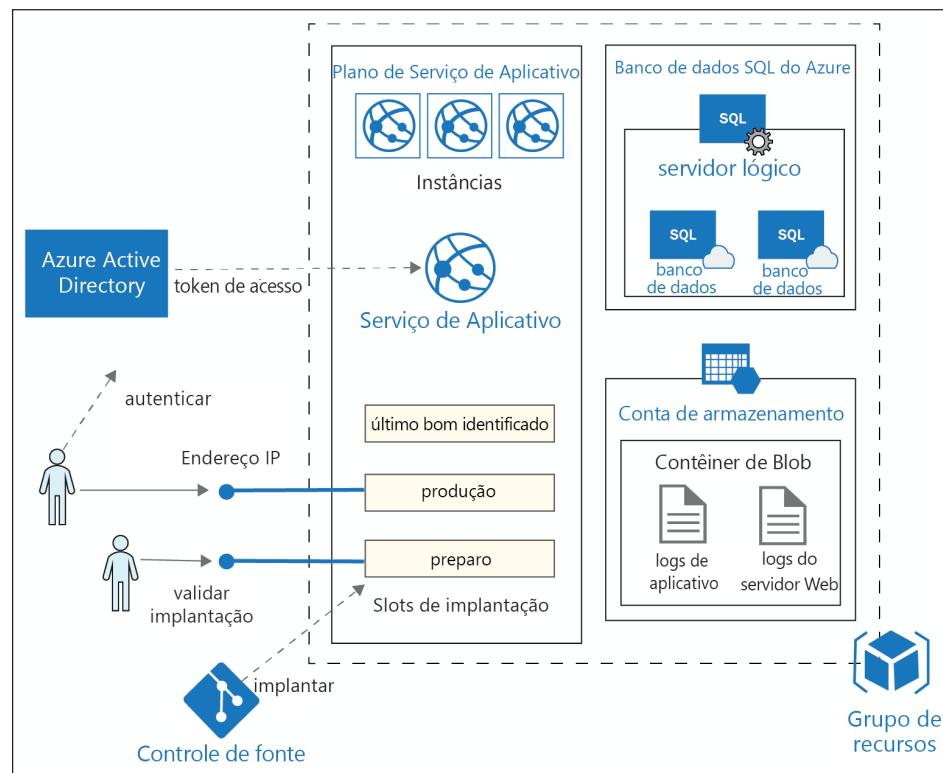
Todas as ferramentas de monitoramento, como Azure Monitor, Application Insights, Log Analytics, OMS e System Center Operations Manager devem ser provisionadas e configuradas durante o pipeline de lançamento de serviços comuns.

Ferramentas de gerenciamento

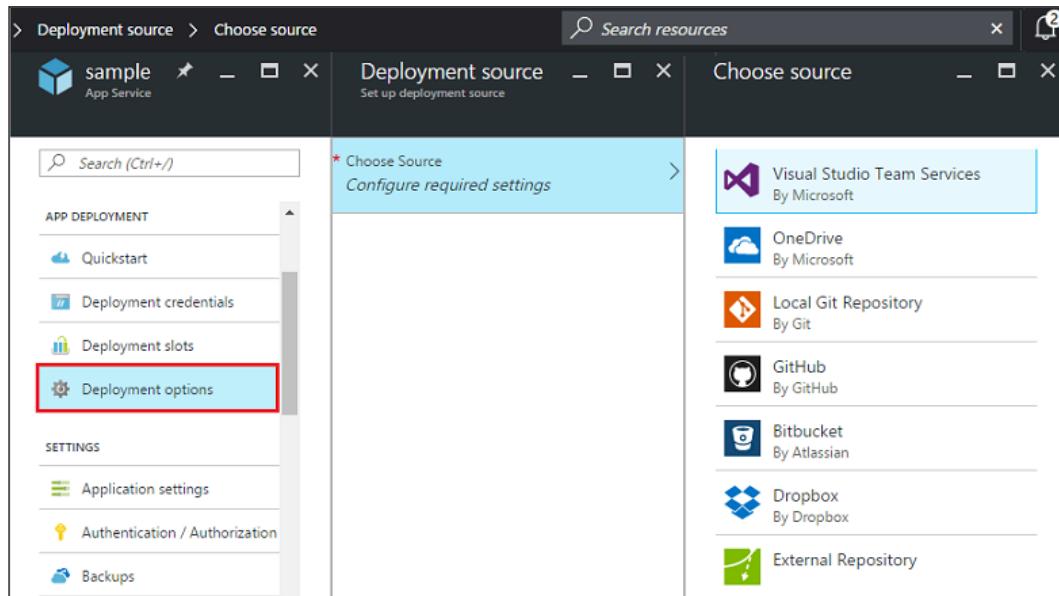
Todas as ferramentas de gerenciamento, como Kubernetes, DC/OS, Docker Swarm e ferramentas ITIL devem ser provisionadas nesta etapa.

DevOps para soluções de PaaS

A arquitetura típica para serviços de aplicativos de PaaS do Azure se baseia no diagrama mostrado a seguir:



A arquitetura mostra alguns dos componentes importantes, como SQL do Azure, contas de armazenamento, o sistema de controle de versão e muitos outros que participam da arquitetura de solução de nuvem baseada em Serviços de Aplicativo do Azure. Esses artefatos devem ser criados usando modelos ARM. Esses modelos do ARM devem fazer parte da estratégia global de gerenciamento de configuração. Ele pode ter seus próprios pipelines de gerenciamento de lançamento e build muito semelhantes àquele mostrado na captura de tela a seguir:



O modelo também deve configurar a implantação contínua por meio da configuração de **Opções de implantação**.

Serviços de Aplicativos do Azure

Os Serviços de Aplicativo do Azure fornece serviços de hospedagem gerenciados para soluções de nuvem. É uma plataforma totalmente gerenciada que provisiona e implanta soluções de nuvem. Os Serviços de Aplicativo do Azure eliminam o fardo de criar e gerenciar a infraestrutura e fornecem **contratos de nível de serviço (SLA)** mínimos para hospedar suas soluções de nuvem.

Eles são abertos, permitindo que os usuários escolham a linguagem que desejam usar para criar suas soluções de nuvem, e flexíveis suficientes para hospedar a solução de nuvem em sistemas operacionais Windows ou Linux. A criação de um serviço de aplicativo provisiona internamente máquinas virtuais nos bastidores que são completamente gerenciadas pelo Azure sem que os usuários as vejam. Vários tipos diferentes de soluções de nuvem, como aplicativos Web, APIs de back-end móvel, pontos de extremidade de API e contêineres podem ser hospedados perfeitamente em Serviços de Aplicativo do Azure.

Slots de implantação

Os Serviços de Aplicativo do Azure fornece slots de implantação que tornam a implantação neles tranquila e fácil. Existem vários slots e a troca entre slots é feita no nível de DNS. Isso significa que qualquer coisa no slot de produção pode ser trocado com um slot de preparo simplesmente trocando as entradas de DNS. Isso ajuda na implantação inicial da solução de nuvem personalizada no preparo e, após todas as verificações e testes, eles podem ser trocados para produção, se considerados satisfatórios. No entanto, caso ocorram problemas na produção após a troca, os bons valores anteriores do ambiente de produção podem ser restabelecidos por meio de uma nova troca.

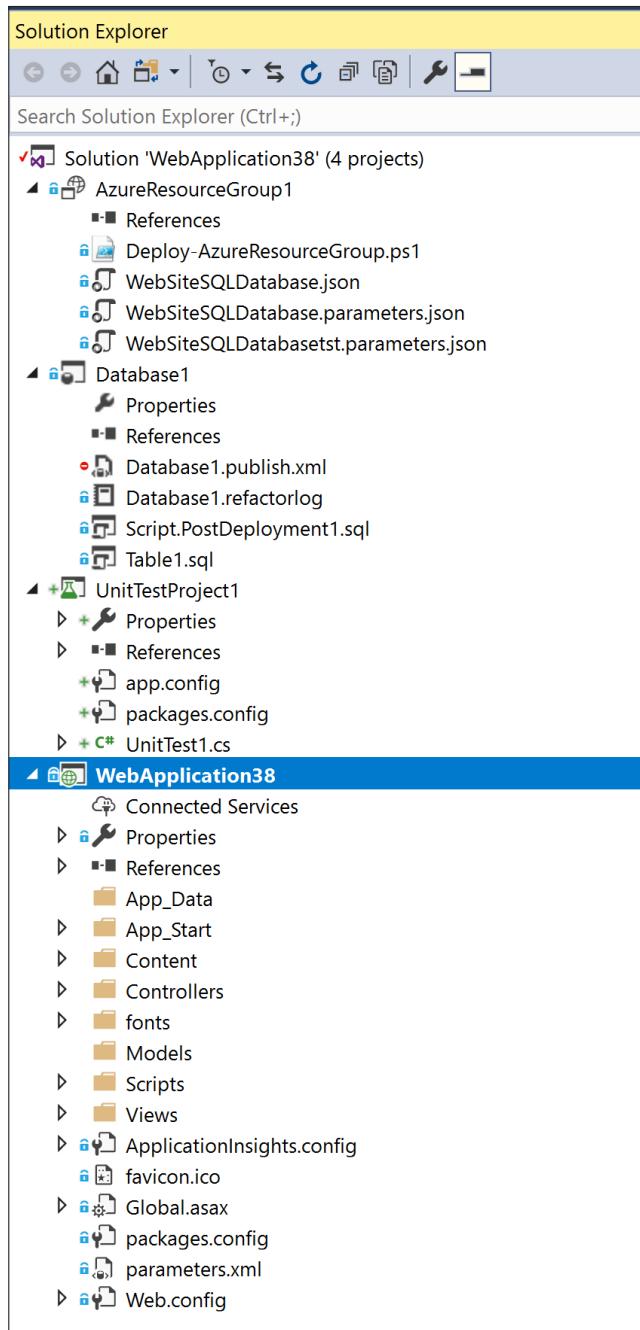
SQL do Azure

O SQL do Azure é um serviço de PaaS do SQL fornecido pelo Azure para hospedar bancos de dados. O Azure fornece uma plataforma segura para hospedar bancos de dados e assume total propriedade para gerenciar a disponibilidade, a confiabilidade e a escalabilidade do serviço. Com o SQL do Azure, não é necessário provisionar máquinas virtuais personalizadas, implantar um SQL Server e configurá-lo. Em vez disso, a equipe do Azure realiza isso nos bastidores e as gerencia em nosso nome. Ela também fornece um serviço de firewall que habilita a segurança, e somente um endereço IP permitido pelo firewall pode se conectar ao servidor e acessar o banco de dados. As máquinas virtuais provisionadas para hospedar aplicativos Web têm endereços IP públicos distintos atribuídos a elas e são adicionadas dinamicamente às regras de firewall do SQL do Azure. O Servidor SQL do Azure e seu banco de dados são criados ao executar o modelo ARM.

O pipeline de build e lançamento

Nesta seção, um novo pipeline de build é criado que compila e valida um aplicativo ASP.NET MVC e, em seguida, gera pacotes para implantação. Após a geração do pacote, uma definição de versão garante que a implantação no primeiro ambiente ocorra em um serviço de aplicativo e no SQL do Azure como parte da implantação contínua.

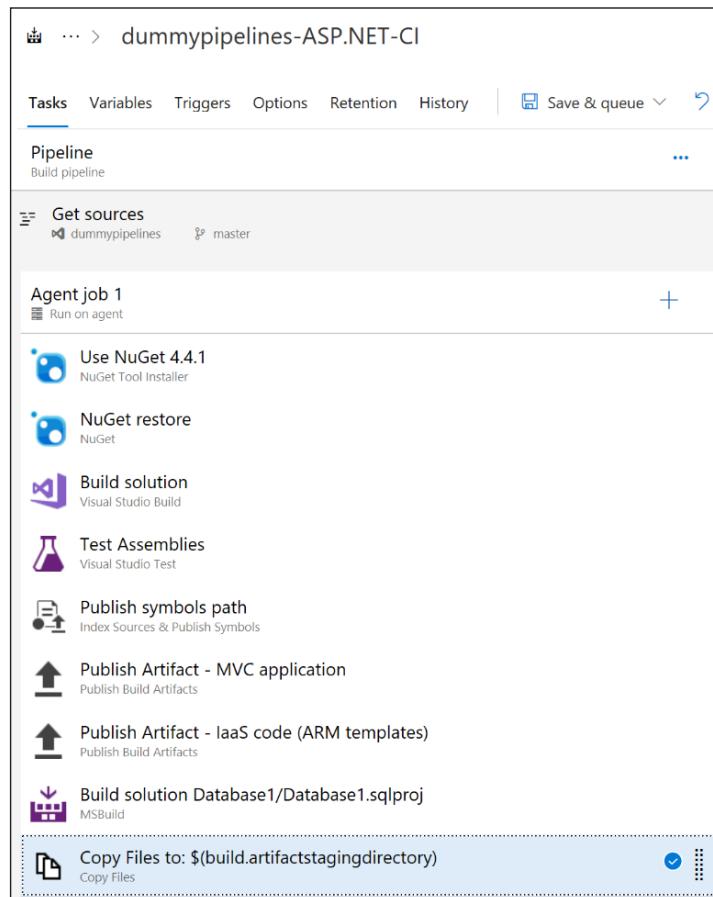
A estrutura do projeto do aplicativo de exemplo é mostrada na seguinte captura de tela:



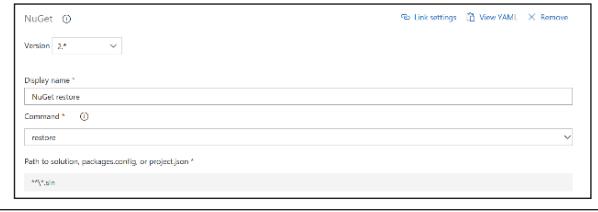
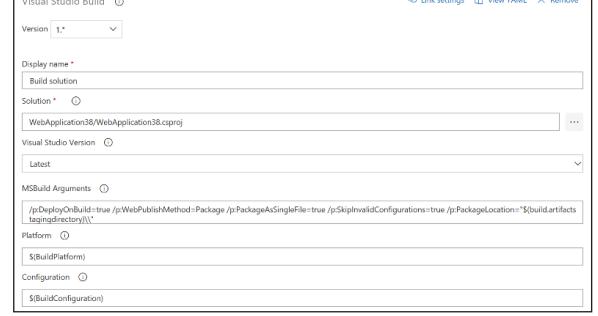
Neste projeto, há um aplicativo ASP.NET MVC – o aplicativo principal, que consiste em páginas de aplicativo. Os pacotes WebDeploy serão gerados fora deste projeto de pipelines de build e eles eventualmente estarão em aplicativos Web do Azure.

- **Projeto de teste de unidade:** código para teste de unidade do aplicativo ASP.NET MVC. Os assemblies deste projeto serão gerados e executados na execução de build.
- **Projeto do Banco de Dados SQL:** código relacionado ao esquema, estrutura e dados mestre do banco de dados SQL. Os arquivos DacPac serão gerados fora deste projeto usando a definição de build.
- **Projeto de Grupo de Recursos do Azure:** modelos ARM e código de parâmetros para provisionar todo o ambiente do Azure no qual o aplicativo ASP.NET MVC e as tabelas SQL são criadas.

O pipeline de build é mostrado na captura de tela a seguir:



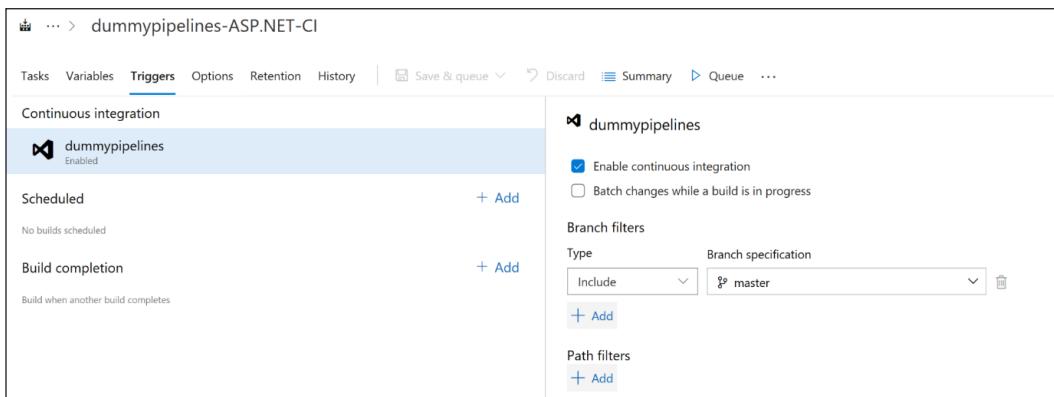
A configuração de cada tarefa é mostrada na tabela a seguir:

Nome da tarefa	Configuração da tarefa
Usar o NuGet 4.4.1	 <p>NuGet Tool Installer</p> <p>Version: 4.4.1</p> <p>Display name: Use NuGet 4.4.1</p> <p>Version of NuGet.exe to install: 4.4.1</p> <p>Always download the latest matching version</p> <p>Control Options</p> <p>Output Variables</p>
Restauração do NuGet	 <p>NuGet</p> <p>Version: 2.*</p> <p>Display name: NuGet restore</p> <p>Command: restore</p> <p>Path to solution, packages.config, or project.json: ***.sln</p>
Solução de build	 <p>Visual Studio Build</p> <p>Version: 1.*</p> <p>Display name: Build solution</p> <p>Solution: WebApplication38\WebApplication38.csproj</p> <p>Visual Studio Version: Latest</p> <p>MSBuild Arguments: /p:DeployOnBuild=true /p:WebPublishMethod=Package /p:PackageAsSingleFile=true /p:SkipInvalidConfigurations=true /p:PackageLocation="\$(build.artifactsDirectory)\\"</p> <p>Platform: \$(BuildPlatform)</p> <p>Configuration: \$(BuildConfiguration)</p>
Assemblies de teste	 <p>Visual Studio Test</p> <p>Version: 2.*</p> <p>Display name: Test Assemblies</p> <p>Test selection</p> <p>Select tests using: Test assemblies</p> <p>Test files: **\\$(BuildConfiguration)**test*.dll !**\obj**</p> <p>Search folder: \$(System.DefaultWorkingDirectory)</p>

Nome da tarefa	Configuração da tarefa
Publicar caminho de símbolos	<p>Index Sources & Publish Symbols</p> <p>Version: 2.*</p> <p>Display name *</p> <p>Publish symbols path</p> <p>Path to symbols folder</p> <p>\$(build.SourcesDirectory)</p> <p>Search pattern *</p> <p>**\bin*.pdb</p> <p><input checked="" type="checkbox"/> Index sources</p> <p><input type="checkbox"/> Publish symbols</p>
Publicar Artefato - Aplicativo MVC	<p>Publish Build Artifacts</p> <p>Version: 1.*</p> <p>Display name *</p> <p>Publish Artifact - MVC application</p> <p>Path to publish</p> <p>\$(build.artifactstagingdirectory)</p> <p>Artifact name *</p> <p>drop</p>
Publicar Artefato - código IaaS (modelos ARM)	<p>Publish Build Artifacts</p> <p>Version: 1.*</p> <p>Display name *</p> <p>Publish Artifact - IaaS code (ARM templates)</p> <p>Path to publish</p> <p>AzureResourceGroup1</p> <p>Artifact name *</p> <p>drop1</p> <p>Artifact publish location *</p> <p>Azure Pipelines/TFS</p>
Solução de build Database1/Database1.sqlproj	<p>MSBuild</p> <p>Version: 1.*</p> <p>Display name *</p> <p>Build solution Database1\Database1.sqlproj</p> <p>Project *</p> <p>Database1\Database1.sqlproj</p> <p>MSBuild</p> <p><input checked="" type="radio"/> Version</p> <p>MSBuild Version</p> <p>Latest</p> <p>MSBuild Architecture</p> <p>MSBuild x64</p> <p>Platform</p> <p>Configuration</p> <p>MSBuild Arguments</p> <p>/t:build /p:CmdLineInMemoryStorage=True</p>

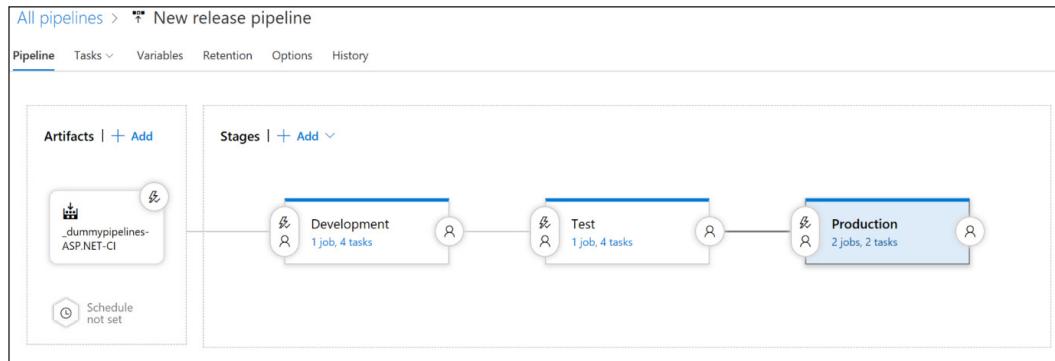
Nome da tarefa	Configuração da tarefa
Copiar Arquivos para: \$(build.artifactstagingdirectory)	

O pipeline de build é configurado para executar automaticamente como parte da integração contínua, conforme mostrado na captura de tela a seguir:

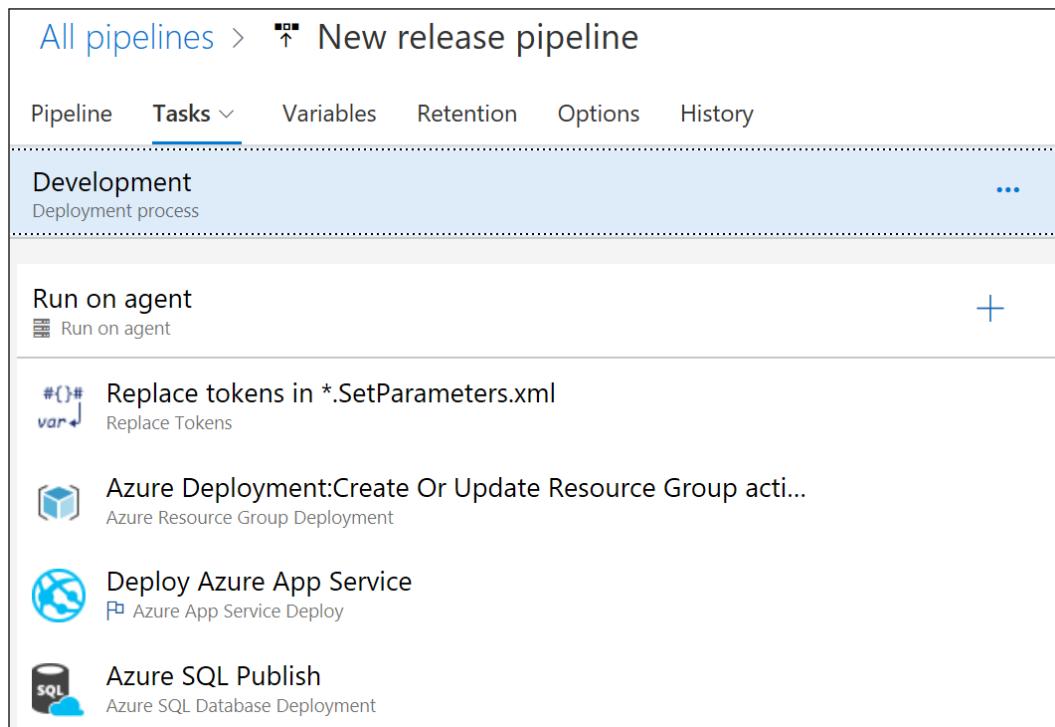


A definição de versão consiste em vários ambientes, como desenvolvimento, teste, SIT, UAT, pré-produção e produção. As tarefas são bastante semelhantes em cada ambiente, com a adição de tarefas específicas para esse ambiente. Por exemplo, um ambiente de teste tem tarefas adicionais relacionadas à interface do usuário e testes funcionais e de integração, em comparação a um ambiente de desenvolvimento.

A definição de lançamento para esse aplicativo é mostrada na seguinte captura de tela:



As tarefas de lançamento para um único ambiente são mostradas na seguinte captura de tela:



A configuração para cada uma das tarefas é listada aqui:

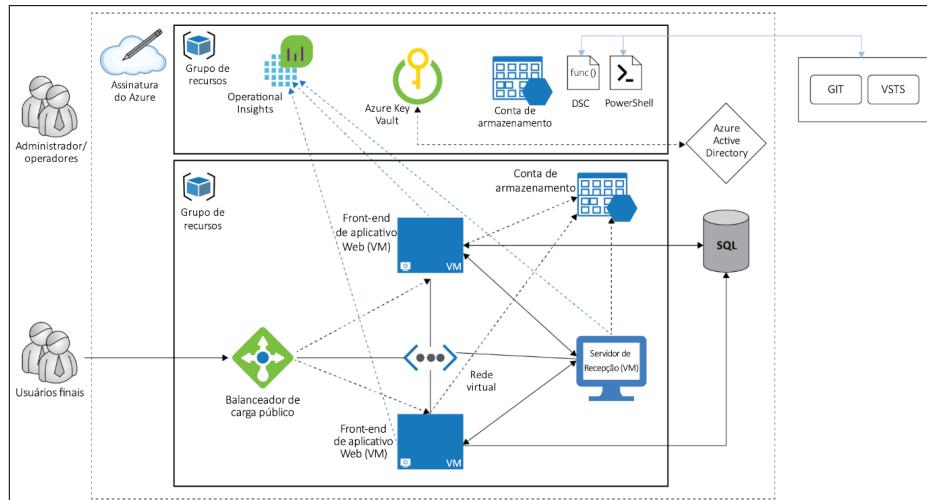
Nome da tarefa	Configuração da tarefa
Substitua tokens em *.SetParameters.xml (Essa é uma tarefa instalada desde o MarketPlace).	

Nome da tarefa	Configuração da tarefa
Implantação do Azure: criar ou Atualizar ação do Grupo de Recursos no devRG	<p>Azure Resource Group Deployment</p> <p>Version 2*</p> <p>Display name * Azure Deployment:Create Or Update Resource Group action on devRG</p> <p>Azure Details ^</p> <p>Azure subscription * myconnection</p> <p>Action * Create or update resource group</p> <p>Resource group * devRG</p> <p>Location * West Europe</p> <p>Template ^</p> <p>Template location * Linked artifact</p> <p>Template * \${System.DefaultWorkingDirectory}/_dummypipelines-ASP.NET-C/drop1/WebSiteSQLDatabase.json</p> <p>Template parameters * \${System.DefaultWorkingDirectory}/_dummypipelines-ASP.NET-C/drop1/WebSiteSQLDatabase.parameters.json</p> <p>Override template parameters * -sqlserverName \${SQLServerName} -hostingPlanName \${AppServiceName}</p> <p>Deployment mode * Incremental</p>
Implantar o Serviço de Aplicativo do Azure	<p>Azure App Service Deploy</p> <p>Version 3*</p> <p>Display name * Deploy Azure App Service</p> <p>Azure subscription * myconnection</p> <p>App type * webApp</p> <p>App Service name * \$(AppServiceName)</p> <p><input type="checkbox"/> Deploy to slot</p> <p>Virtual application</p> <p>Package or folder * \${System.DefaultWorkingDirectory}/_dummypipelines-ASP.NET-C/drop/WebApplication38.zip</p>

Nome da tarefa	Configuração da tarefa
Publicação do SQL do Azure	

DevOps para soluções baseadas em máquinas virtuais (IaaS)

A arquitetura típica para uma solução de IaaS baseada em máquinas virtuais é mostrada aqui:



Máquinas Virtuais do Azure

Máquinas Virtuais do Azure que hospedam aplicativos Web, servidores de aplicativos, bancos de dados e outros serviços são provisionadas usando modelos do ARM. Cada máquina virtual tem uma única placa de rede com um IP público atribuído a ela. Elas estão conectadas a uma rede virtual e têm um endereço IP privado da mesma rede. O IP público para máquinas virtuais é opcional, pois elas estão conectadas a um balanceador de carga público. Essas máquinas virtuais são baseadas em uma imagem de servidor do Windows 2016. Os agentes de insight operacional são instalados em máquinas virtuais para monitorá-las. Scripts do PowerShell também são executados nessas máquinas virtuais cujo download foi feito de uma conta de armazenamento disponível em outro grupo de recursos para abrir portas de firewall relevantes, fazer download de pacotes apropriados e instalar certificados locais para garantir o acesso por meio do PowerShell. O aplicativo Web está configurado para ser executado na porta fornecida nessas máquinas virtuais. O número da porta para o aplicativo Web e toda a sua configuração é obtido do servidor de pull do DSC e atribuído dinamicamente.

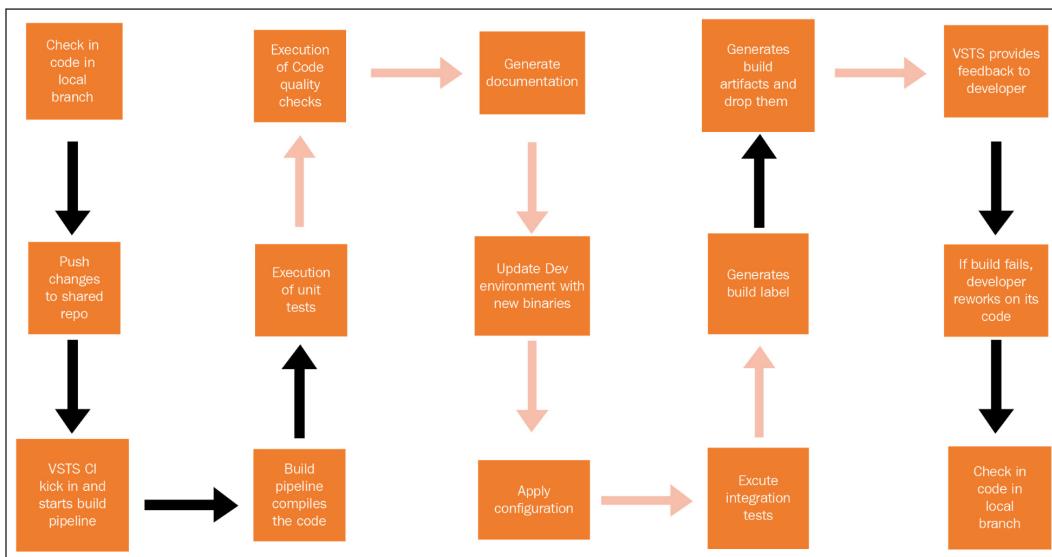
Balanceadores de carga públicos do Azure

Um balanceador de carga público é conectado a algumas das máquinas virtuais para enviar solicitações a elas de maneira alternada. Isso geralmente é necessário para APIs e aplicativos Web front-end. Um endereço IP público e um nome DNS podem ser atribuídos a um balanceador de carga de modo que ele possa atender a solicitações da Internet. Ele aceita solicitações HTTP da Web em portas diferentes e as encaminha para as máquinas virtuais. Ele também explora determinadas portas em protocolos HTTP com alguns caminhos de aplicativos fornecidos. Regras de **Conversão de Endereços de Rede (NAT)** também podem ser aplicadas de modo que possam ser usadas para fazer logon nas máquinas virtuais usando áreas de trabalho remotas.

Um recurso alternativo ao balanceador de carga pública do Azure é o gateway de aplicativo do Azure. Os gateways de aplicativo são平衡adores de carga de camada 7 e fornecem recursos como terminação SSL, afinidade de sessão e roteamento baseado em URL.

O pipeline de build

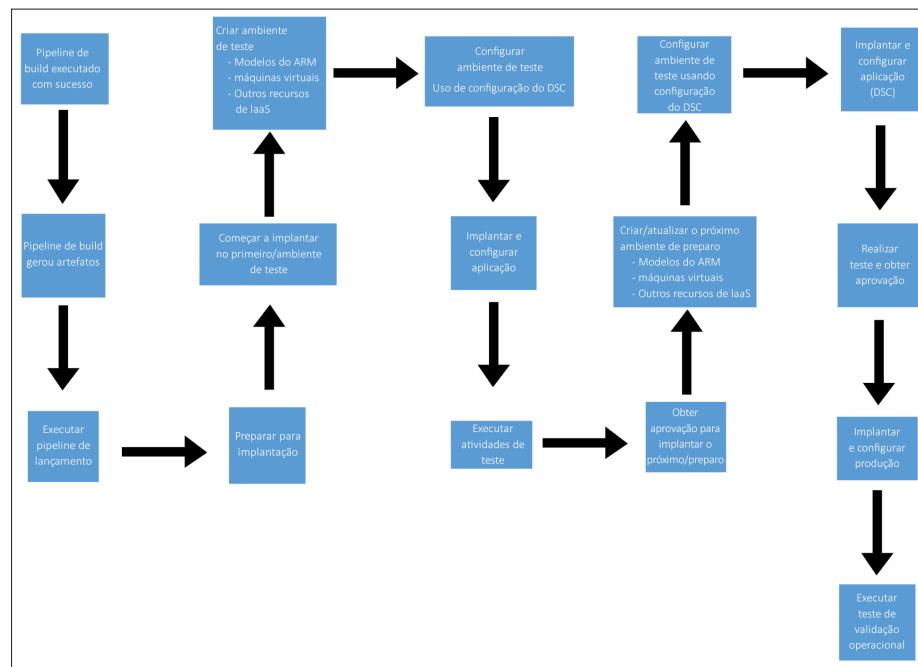
Um pipeline de build típico para uma solução de IaaS baseada em máquinas virtuais é mostrado a seguir. Um pipeline de lançamento é iniciado quando um desenvolvedor envia seu código ao repositório. O pipeline de build é iniciado automaticamente como parte da integração contínua. Ele compila e cria o código, realiza testes de unidade e verificações de qualidade nele, verifica a qualidade do código e gera a documentação desde os comentários do código. Ele implanta os novos binários no ambiente de desenvolvimento (observe que esse ambiente não é recém-criado), altera configurações, realiza testes de integração e gera rótulos de build para fácil identificação. Em seguida, ele deposita os artefatos gerados em um local acessível pelo pipeline de lançamento. Caso haja algum problema durante a execução de qualquer etapa desse pipeline, isso será comunicado ao desenvolvedor como parte dos comentários do pipeline de build para que possa retrabalhar e reenviar suas alterações. O pipeline de build falhará ou será aprovado com base na gravidade dos problemas encontrados e isso varia de acordo com a organização. Um pipeline de build típico é mostrado no seguinte diagrama:



O pipeline de lançamento

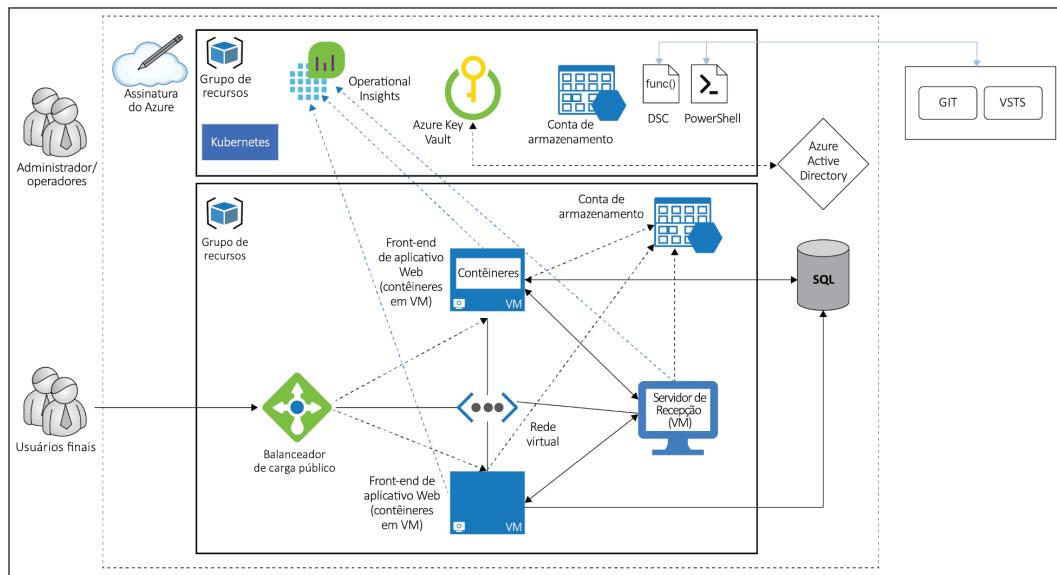
Um pipeline de lançamento típico para uma implantação de IaaS baseada em máquinas virtuais é mostrado a seguir. Um pipeline de lançamento é iniciado após a conclusão do pipeline de build. A primeira etapa do pipeline de lançamento é a reunião dos artefatos gerados pelo pipeline de build. Eles geralmente são documentos de configuração, binários e assemblies implantáveis. O pipeline de lançamento executa e cria ou atualiza o primeiro ambiente que geralmente é um ambiente de teste. Ele usa modelos do ARM para provisionar todos os recursos e serviços de IaaS e PaaS no Azure, além de configurá-los. Ele também ajuda na execução de scripts e na configuração de DSC após a criação de máquinas virtuais como etapas de pós-criação. Isso ajuda a configurar o ambiente dentro da máquina virtual e do sistema operacional. Nesse estágio, os binários de aplicativos do pipeline de build são implantados e configurados. Diferentes testes automatizados são realizados para verificar o funcionamento na solução e, se considerado satisfatório, o pipeline passa para a implantação no próximo ambiente após obter as aprovações necessárias. As mesmas etapas são executadas novamente no próximo ambiente, incluindo o ambiente de produção. Por fim, os testes de validação operacional são realizados na produção para garantir que o aplicativo esteja funcionando conforme o esperado e que não haja desvios.

Nesse estágio, se houver problemas ou bugs, eles devem ser corrigidos e o ciclo inteiro deve ser repetido; no entanto, se isso não acontecer dentro de um período de tempo estipulado, o último instantâneo conhecido deve ser restaurado no ambiente de produção para minimizar o tempo de inatividade. Um pipeline de lançamento típico é mostrado no seguinte diagrama:



DevOps para soluções baseadas em contêineres (IaaS)

A arquitetura típica para soluções baseadas em contêineres IaaS é mostrada aqui:



Na arquitetura mostrada anteriormente, os tempos de execução do contêiner são implantados em máquinas virtuais e os contêineres são executados dentro deles. Esses contêineres são gerenciados por orquestradores de contêiner, como Kubernetes. Os serviços de monitoramento são fornecidos pelo Log Analytics e todos os segredos e chaves são armazenados no Azure Key Vault. Há também um servidor de pull, que pode estar em uma máquina virtual ou na Automação do Azure, fornecendo informações de configuração para as máquinas virtuais.

Contêineres

Os contêineres são uma tecnologia de virtualização; no entanto, eles não virtualizam servidores físicos. Em vez disso, contêineres são uma virtualização no nível do sistema operacional. Isso significa que os contêineres compartilham o kernel do sistema operacional fornecido pelo host entre eles e com o host. A execução de vários contêineres em um host (físico ou virtual) compartilha o kernel do sistema operacional do host. Há um único kernel de sistema operacional fornecido pelo host e usado por todos os contêineres em execução sobre ele.

Os contêineres também são completamente isolados de seu host e de outros contêineres, assim como uma máquina virtual. Os contêineres usam namespaces do sistema operacional, grupos de controles no Linux, para fornecer a percepção de um novo ambiente de sistema operacional e usar técnicas específicas de virtualização do sistema operacional no Windows. Cada contêiner tem sua própria cópia dos recursos do sistema operacional.

Docker

O Docker fornece recursos de gerenciamento a contêineres. Ele é composto por dois executáveis:

- Daemon do Docker
- Cliente do Docker

O daemon do Docker é útil para o gerenciamento de contêineres. É um serviço de gerenciamento responsável por gerenciar todas as atividades no host relacionadas a contêineres. O cliente do Docker interage com o daemon do Docker e é responsável pela captura de entrada e pelo seu envio ao daemon do Docker. O daemon do Docker fornece tempo de execução, bibliotecas, drivers de gráficos, os mecanismos para criar, gerenciar e monitorar contêineres e imagens no servidor host. Ele também pode criar imagens personalizadas que são usadas para criar e enviar aplicações para vários ambientes.

Dockerfile

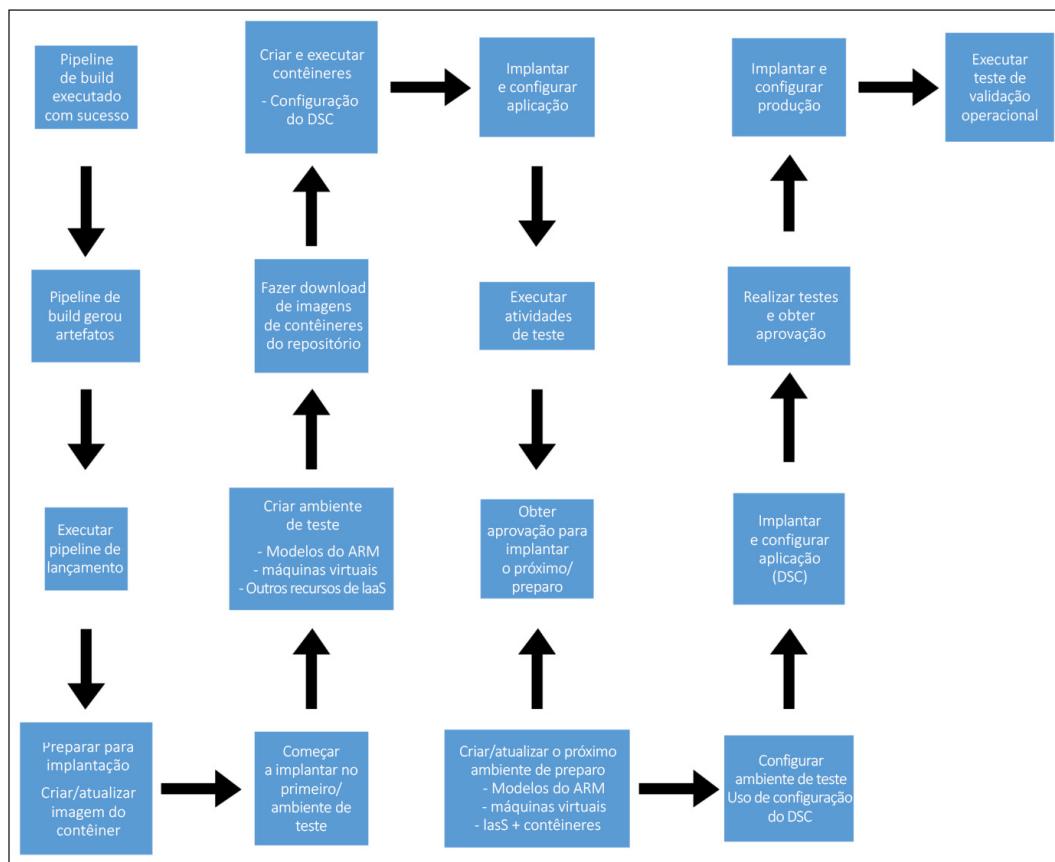
O Dockerfile é o bloco de construção fundamental na criação de imagens de contêiner do Windows. É um arquivo simples baseado em texto e legível por humanos sem nenhuma extensão, e é até chamado de Dockerfile. Embora haja um mecanismo para nomeá-lo de forma diferente, geralmente é chamado de Dockerfile. O Dockerfile contém instruções para criar uma imagem personalizada usando uma imagem base. Essas instruções são executadas sequencialmente de cima para baixo pelo daemon do Docker. As instruções referem-se ao comando e seus parâmetros, como COPY, ADD, RUN e ENTRYPOINT. O Dockerfile possibilita as práticas de IaC, convertendo a configuração e a implantação do aplicativo em instruções que podem ter a versão controlada e ser armazenadas em um repositório de código-fonte.

O pipeline de build

Não há nenhuma diferença, da perspectiva do build, entre o contêiner e uma solução baseada em máquinas virtuais. A etapa de build permanece a mesma. Consulte a seção de soluções baseadas em *DevOps para a máquina virtual (IaaS)* para obter detalhes de pipeline de build.

O pipeline de lançamento

Um pipeline de lançamento típico para uma implantação de IaaS baseada em contêineres é mostrado a seguir. A única diferença entre ele e o pipeline de lançamento é o gerenciamento de imagens de contêiner e a criação de contêineres usando o Dockerfile e o Docker Compose. Os utilitários avançados de gerenciamento de contêineres avançados, como Docker Swarm, DC/OS e Kubernetes, também podem ser implantados e configurados como parte do gerenciamento de versões. No entanto, note que essas ferramentas de gerenciamento de contêineres devem fazer parte do pipeline de lançamento de serviços compartilhados conforme discutido anteriormente. O diagrama a seguir mostra um pipeline de lançamento típica para uma solução baseada em contêiner:

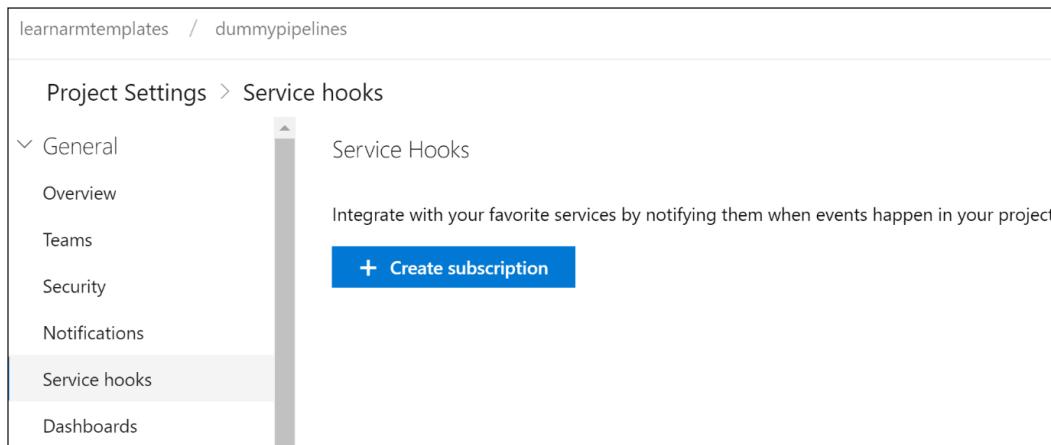


Azure DevOps e Jenkins

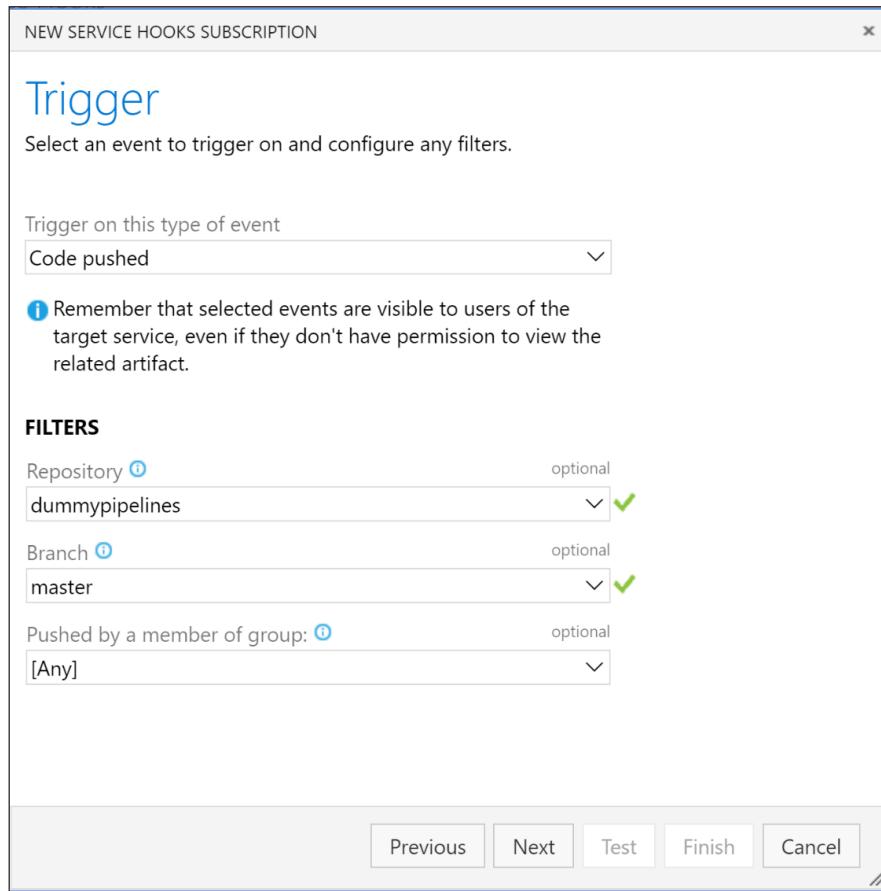
O Azure DevOps é um orquestrador de plataforma aberta que se integra com outras ferramentas de orquestrador sem problemas. Ele fornece toda a infraestrutura necessária e recursos que se integram bem com Jenkins, também. As organizações com pipelines de CI/CD bem estabelecidos criados no Jenkins podem reutilizá-las com os recursos avançados mas simples do Azure DevOps para orquestrar-los.

O Jenkins pode ser usado como um repositório e pode executar pipelines de CI/CD no Azure DevOps, enquanto também é possível ter um repositório no Azure DevOps e executar pipelines de CI/CD no Jenkins.

A configuração do Jenkins pode ser adicionada ao Azure DevOps como ganchos de serviço e sempre que qualquer alteração de código for confirmada no repositório do Azure DevOps, ele poderá acionar pipelines no Jenkins. A próxima captura de tela mostra a configuração do Jenkins da seção de configuração de gancho de serviço do Azure DevOps:



Há vários gatilhos que executam os pipelines no Jenkins; um deles é o **Código enviado por push**, como mostrado na seguinte captura de tela:



Também é possível implantar na VM do Azure e executar pipelines de lançamento do Azure DevOps, conforme explicado aqui: <https://docs.microsoft.com/azure/virtual-machines/linux/tutorial-build-deploy-jenkins>:

O Jenkins já deve ter sido implantado antes de usá-lo em qualquer cenário. O processo de implantação no Linux pode ser encontrado em <https://docs.microsoft.com/azure/virtual-machines/linux/tutorial-jenkins-github-docker-cicd>.

Automação do Azure

A Automação do Azure é a plataforma da Microsoft para todas as implementações de automação referentes a implantações na nuvem, na infraestrutura local e híbridas. A Automação do Azure é uma plataforma de automação madura que fornece recursos avançados para:

- Definição de ativos, como variáveis, conexões, credenciais, certificados e módulos
- Implementação de runbooks usando Python, scripts do PowerShell e fluxos de trabalho do PowerShell
- Fornecimento de IUs para criar runbooks
- Gerenciamento do ciclo de vida completo do runbook, incluindo criação, testes e publicação
- Agendamento de runbooks
- A capacidade de executar runbooks em qualquer lugar — na nuvem ou na infraestrutura local
- DSC como uma plataforma de gerenciamento de configuração
- Gerenciamento e configuração de ambientes — Windows e Linux, aplicativos e implantação
- A capacidade de estender a Automação do Azure importando módulos personalizados

A Automação do Azure fornece um servidor de pull do DSC que ajuda na criação de um servidor de gerenciamento de configuração centralizado que consiste em configurações para nós/máquinas virtuais e seus componentes.

Ele implementa o padrão hub e spoke no qual os nós podem se conectar ao servidor de pull do DSC, fazer download das configurações atribuídas a eles e se reconfigurar para refletir o estado desejado. Quaisquer alterações ou desvios dentro desses nós serão autocorrigidos por agentes de DSC na próxima vez que forem executados. Isso garante que os administradores não precisem monitorarativamente o ambiente para encontrar desvios.

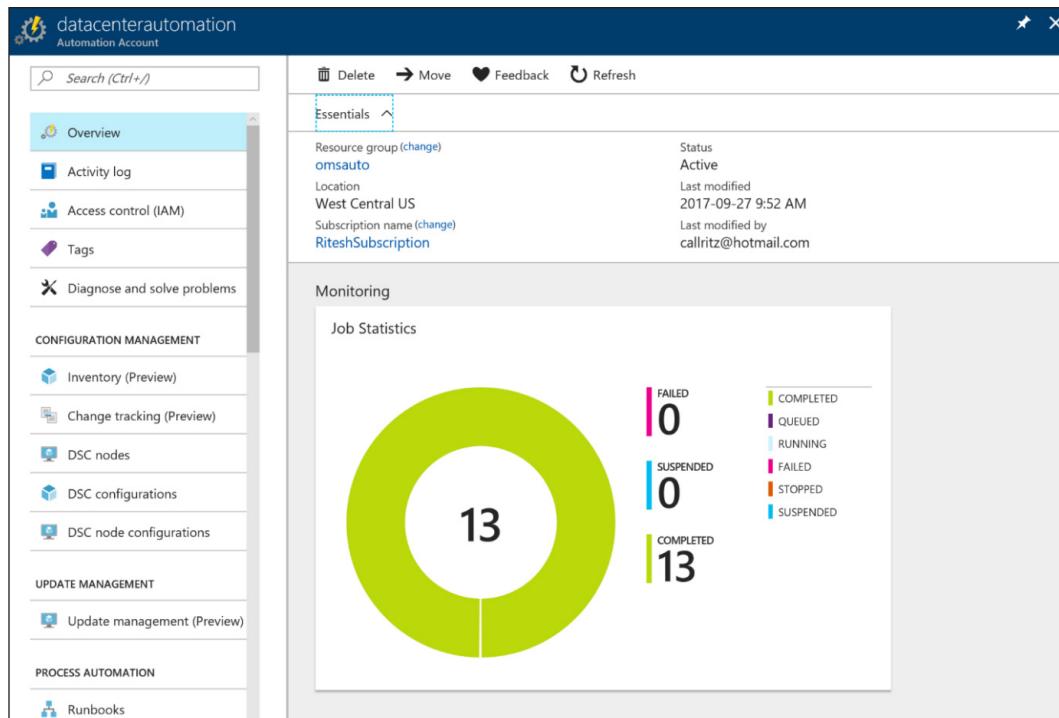
O DSC fornece uma linguagem declarativa, na qual você define a intenção e a configuração, mas não como executar e aplicar essas configurações. Essas configurações são baseadas na linguagem do PowerShell e facilitam o processo de gerenciamento de configuração.

Nesta seção, analisaremos uma implementação simples de uso do DSC da Automação do Azure para configurar uma máquina virtual com o objetivo de instalar e configurar o servidor Web (IIS) e criar um arquivo index.htm que informe aos usuários que o site está em manutenção.

Provisionamento de conta da Automação do Azure

Crie uma nova conta da Automação do Azure no portal do Azure ou no PowerShell dentro de um grupo de recursos novo ou existente. Leitores astutos verão no próximo diagrama que a Automação do Azure fornece itens de menu para DSC. Ela oferece o seguinte:

- **Nós de DSC:** listam todos os contêineres e máquinas virtuais que estão registrados no atual servidor de pull do DSC da Automação do Azure. Esses contêineres e máquinas virtuais são gerenciados usando as configurações do atual servidor de pull do DSC atual.
 - **Configurações de DSC:** listam todas as configurações brutas do PowerShell importadas e carregadas no servidor de pull do DSC. Elas estão em formato legível por humanos e não estão em um estado compilado.
 - **Configurações de nós de DSC:** listam todas as compilações de configurações de DSC disponíveis no servidor de pull a serem atribuídas a nós — máquinas virtuais e contêineres. Uma configuração de DSC produz arquivos MOF após compilações e eles eventualmente são usados para configurar nós.
- A próxima captura de tela mostra uma conta da Automação do Azure depois de provisionada:



Criação de configuração de DSC

A próxima etapa é a criação de uma configuração de DSC usando qualquer editor do PowerShell para refletir a intenção da configuração. Para este exemplo, uma única configuração, `ConfigureSiteOnIIS`, é criada. Ela importa o módulo de base do DSC, `PSDesiredStateConfiguration`, que consiste em recursos usados dentro da configuração. Ela também declara um servidor Web de nó. Quando essa configuração for enviada por upload e compilada, ela gerará uma Configuração de DSC chamada `ConfigureSiteOnIISwebserver`. Essa configuração poderá então ser aplicada aos nós.

A configuração consiste em alguns recursos. Esses recursos configuram o nó de destino. Os recursos instalam um servidor Web, o ASP.NET e a estrutura, e criam um arquivo `index.htm` no diretório `inetpub\wwwroot` com conteúdo para mostrar que o site está em manutenção. Para obter mais informações sobre a criação de configuração de DSC, consulte <https://docs.microsoft.com/en-us/PowerShell/dsc/configurations>.

A próxima listagem de código mostra toda a configuração descrita no parágrafo anterior. O upload dessa configuração feito na conta da Automação do Azure:

```
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
```

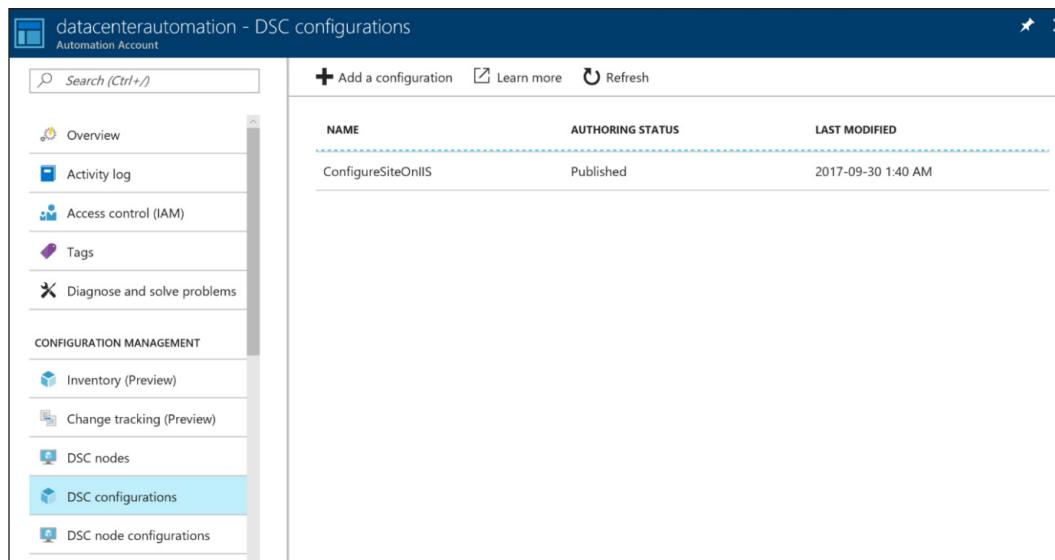
```
Contents = "<HTML><HEAD><Title> Website under
construction.</Title></HEAD><BODY>
`  
    <h1>If you are seeing this page, it means the website is
under maintenance and DSC Rocks !!!!!</h1></BODY></HTML>"  
}  
}  
}
```

Importação da configuração de DSC

A configuração de DSC ainda não é conhecida pela Automação do Azure. Ela está disponível em alguns computadores locais. Ela deve ser carregada nas Configurações de DSC da Automação do Azure. A Automação do Azure fornece o cmdlet `Import-AzureRmAutomationDscConfiguration` para importar a configuração para ela:

```
Import-AzureRmAutomationDscConfiguration -SourcePath "C:\DSC\AA\
DSCfiles\ConfigSiteOnIIS.ps1" -ResourceGroupName "omsauto"
-AutomationAccountName "datacenterautomation" -Published -Verbose
```

A configuração do DSC no Azure após a aplicação da configuração ao nó deve ter a seguinte aparência:



The screenshot shows the Azure DevOps Automation Account interface. The left sidebar has a 'DSC configurations' item selected. The main area displays a table of configurations:

NAME	AUTHORING STATUS	LAST MODIFIED
ConfigureSiteOnIIS	Published	2017-09-30 1:40 AM

Compilação da configuração de DSC

Depois que a configuração de DSC ficar disponível na Automação do Azure, seu build pode ser compilada. A Automação do Azure fornece outro cmdlet para isso. Use o cmdlet `Start-AzureRmAutomationDscCompilationJob` para compilar a configuração importada. O nome da configuração deve corresponder exatamente ao nome da configuração enviada no upload. O build cria um arquivo MOF cujo nome é a combinação do nome da configuração e do nó, que, nesse caso, é o servidor Web `ConfigureSiteOnIIS`. A execução do comando é mostrada aqui:

```
Start-AzureRmAutomationDscCompilationJob -ConfigurationName
ConfigureSiteOnIIS -ResourceGroupName "omsauto" -AutomationAccountName
"datacenterautomation" -Verbose
```

A configuração do nó DSC no Azure após a aplicação da configuração deve ter a seguinte aparência:

NAME	CREATED	LAST MODIFIED
ConfigureSiteOnIIS.WebServer	2017-09-30 1:42 AM	2017-09-30 1:42 AM

Atribuição de configurações a nós

As configurações de DSC compiladas podem ser aplicadas aos nós. Use Register-AzureRmAutomationDscNode para atribuir a configuração a um nó. O parâmetro NodeConfigurationName identifica o nome de configuração que deve ser aplicado ao nó. Esse é um cmdlet poderoso que também pode configurar o agente de DSC localconfigurationmanager em nós antes que possam fazer download das configurações e aplicá-las. Há vários parâmetros localconfigurationmanager que podem ser configurados — veja detalhes em <https://docs.microsoft.com/en-us/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
```

Após a aplicação da configuração, os nós DSC no Azure deverão ter a seguinte aparência:

NAME	STATUS	NODE CONFIGURATION	LAST SEEN	VM DSC EXT
testtwo	✓ Compliant	ConfigureSiteOnIIS.WebS...	2017-09-30 2:34 AM	Yes

Navegação no servidor

Se apropriado, os grupos de segurança de rede e firewalls são abertos e habilitados para a porta 80 e um IP público é atribuído à máquina virtual, possibilitando a navegação no site padrão usando o endereço IP. Caso contrário, faça logon na máquina virtual que é usada para aplicar a configuração de DSC e navegue até <http://localhost>.

Isso deve mostrar a seguinte página:



Esse é o poder do gerenciamento de configuração: sem escrever nenhum código significativo, a criação de uma única configuração pode ser aplicada várias vezes ao mesmo servidor e a vários outros, e você poderá ter a certeza de que funcionarão no estado desejado sem nenhuma intervenção manual.

Azure para DevOps

Como mencionado anteriormente, o Azure é uma plataforma sofisticada e madura que fornece o seguinte:

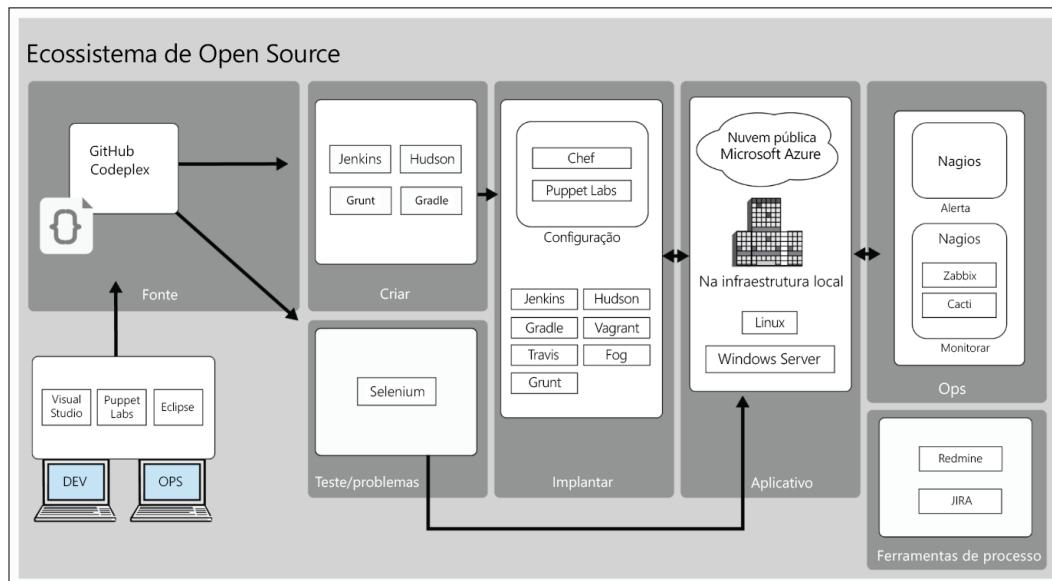
- Várias opções de linguagens
- Várias opções de sistemas operacionais
- Várias opções de ferramentas e utilitários
- Vários padrões para implantação de soluções (como máquinas virtuais, serviços de aplicativos, contêineres e microsserviços)

Com tantas opções e escolhas, o Azure oferece o seguinte:

- **Nuvem aberta:** aberta para serviços, ferramentas e produtos de software livre, da Microsoft ou não.
- **Nuvem flexível:** é fácil o suficiente para os usuários de negócios e desenvolvedores para usá-lo com suas habilidades e conhecimentos existentes.
- **Gerenciamento unificado:** fornece recursos de gerenciamento e monitoramento ininterruptos.

Todos os recursos mencionados aqui são importantes para a implementação bem-sucedida de DevOps. O próximo diagrama mostra os utilitários e ferramentas de Open Source que podem ser usados para diferentes fases do gerenciamento do ciclo de vida do aplicativo e do DevOps em geral. Essa é apenas uma pequena representação de todos os utilitários e ferramentas – há muito mais opções disponíveis, como a seguir:

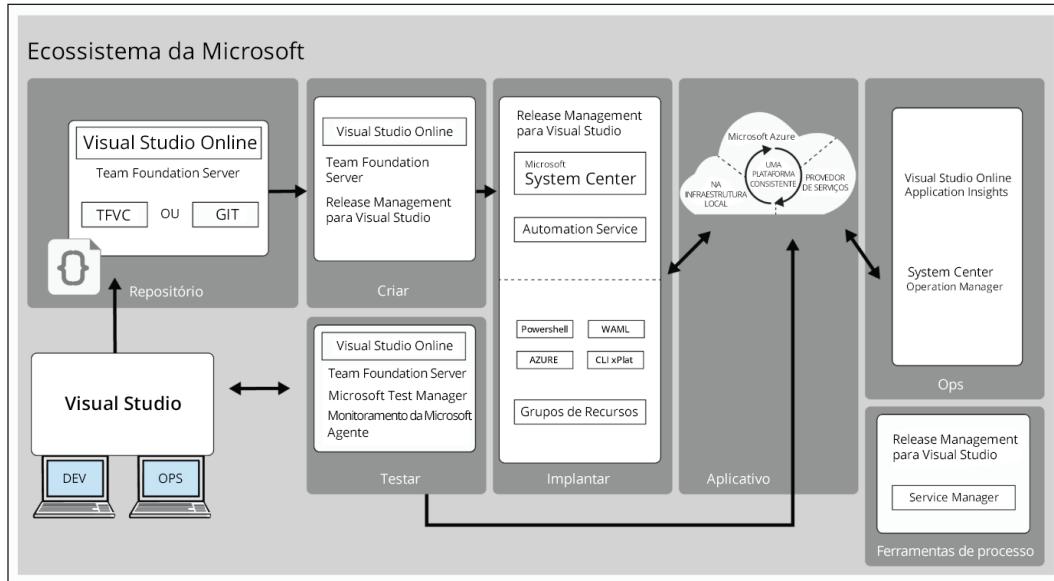
- As ferramentas de Jenkins, Hudson, Grunt e Gradle para construção do pipeline de build
- Selenium para testes
- Chef, Puppet, Jenkins, Hudson, Grunt, Gradle e muitos outros para gerenciamento de configurações ou implantações
- Nagios para alertas e monitoramento
- Jira e Redmine para gerenciamento de processos



O diagrama a seguir mostra os utilitários e ferramentas da Microsoft que podem ser usados para diferentes fases do gerenciamento do ciclo de vida do aplicativo e do DevOps em geral. Novamente, essa é apenas uma pequena representação de todos os utilitários e ferramentas – há muito mais opções disponíveis, como a seguir:

- Orquestração de build do Azure DevOps para construção de um pipeline de build
- Microsoft Test Manager e Pester para testes
- Modelos DSC, PowerShell e ARM para implantação ou gerenciamento de configuração

- Log Analytics, Application Insights e **System Center Operations Manager (SCOM)** para alertas e monitoramento
- Azure DevOps e System Center Service Manager para gerenciamento de processos:



Resumo

O DevOps está obtendo muita tração e impulso na indústria. A maioria das organizações percebeu seus benefícios e está procurando implementá-lo. Isso está acontecendo enquanto a maioria delas está migrando para a nuvem. O Azure, como um modelo de nuvem, fornece serviços de DevOps sofisticados e maduros, facilitando a implementação de DevOps para essas organizações. Neste capítulo, discutimos o DevOps e suas práticas fundamentais, como gerenciamento de configuração, integração contínua, entrega contínua e implantação. Também discutimos diferentes soluções de nuvem baseadas em PaaS, uma IaaS de máquina virtual e uma IaaS de contêiner, juntamente com seus respectivos recursos do Azure, os pipelines de build e lançamento. O gerenciamento de configuração foi a parte principal deste capítulo e discutimos os serviços de DSC da Automação do Azure e o uso de servidores de pull para configurar máquinas virtuais automaticamente. Por fim, abordamos a abertura e a flexibilidade do Azure em relação à opção de linguagens, ferramentas e sistemas operacionais.

12

Soluções OLTP do Azure que usam fragmentação, pools e híbrido de SQL do Azure

O Azure fornece os serviços **Infraestrutura como um Serviço (IaaS)** e **Plataforma como um Serviço (PaaS)**. Esses dois tipos de serviços fornecem diferentes níveis e controles sobre armazenamento, computação e redes para as organizações. Armazenamento é o recurso usado ao trabalhar com o armazenamento e a transmissão de dados. O Azure fornece muitas opções para armazenar dados, como blobs de armazenamento do Azure, tabelas, Cosmos DB, Azure SQL, Azure Data Lake, e muito mais. Embora alguns deles sejam destinados ao armazenamento, análise e apresentação de big data, há outros que são destinados a aplicativos que processam transações. SQL do Azure é o recurso principal no Azure que funciona com dados de transação.

Este capítulo se concentrará em vários aspectos do uso de armazenamentos de dados de transação, como o SQL do Azure e outros bancos de dados open source geralmente usados em sistemas de **Processamento de Transações Online (OLTP)** e abrangerá os seguintes tópicos:

- Aplicativos de OLTP
- Bancos de dados relacionais
- Modelos de implantação
- Banco de dados SQL do Azure
- Pools elásticos
- Instâncias gerenciadas
- Preços do banco de dados SQL

Serviços de nuvem do Azure

Uma pesquisa por `sql` no portal do Azure fornece vários resultados. Eu marquei alguns deles para mostrar os recursos que podem ser usados diretamente para aplicativos OLTP:

The screenshot shows the Microsoft Azure portal interface. On the left is the navigation sidebar with options like 'Create a resource', 'Home', 'Dashboard', 'All services', 'FAVORITES', 'All resources', 'Resource groups', 'App Services', and 'Function Apps'. The main area has a search bar at the top with the placeholder 'Search resources, services, and docs' and a search term 'sql' entered. Below the search bar, there's a button labeled 'All services' and a small dropdown menu showing 'sql'. The results list contains several items, each with a small icon, a name, a 'PREVIEW' or star icon, and a 'Keywords' field:

- Elastic Job agents (Keywords: SQL)
- Managed databases (Keywords: SQL)
- SQL databases (Keywords: SQL)
- SQL managed instances (Keywords: SQL)
- SQL servers (Keywords: SQL)
- HDInsight clusters (Keywords: SQL)
- SQL data warehouses (Keywords: SQL)
- SQL elastic pools (Keywords: SQL)
- SQL Server stretch databases (Keywords: SQL Server)
- Virtual machines (Keywords: SQL Server)

The 'SQL Server stretch databases' item is highlighted with a blue background.

A captura de tela anterior mostra os diversos recursos e opções disponíveis para a criação de bancos de dados baseados no SQL Server no Azure.

Novamente, uma pesquisa rápida por banco de dados no portal do Azure fornece vários recursos e aqueles marcados podem ser usados para aplicativos OLTP:

The screenshot shows the Microsoft Azure portal interface, similar to the previous one but with a different search term. The navigation sidebar is identical. The main area has a search bar with the placeholder 'Search resources, services, and docs' and a search term 'database' entered. Below the search bar, there's a button labeled 'All services' and a small dropdown menu showing 'database'. The results list contains several items, each with a small icon, a name, a star icon, and a 'Keywords' field:

- Azure Cosmos DB (Keywords: Database)
- Azure Database for MySQL servers (Keywords: database)
- Azure Database Migration Services (Keywords: database)
- Free services (Keywords: free database)
- SQL databases (Keywords: Stretch Database)
- Azure Database for MariaDB servers (Keywords: database)
- Azure Database for PostgreSQL servers (Keywords: database)
- Elastic Job agents (Keywords: database) (PREVIEW)
- Managed databases (Keywords: database)
- SQL Server stretch databases (Keywords: Stretch Database)

The 'SQL databases' item is highlighted with a blue background.

A captura de tela anterior mostra os recursos fornecidos pelo Azure que podem hospedar dados em vários bancos de dados, incluindo o seguinte:

- Bancos de dados MySQL
- Bancos de dados MariaDB
- Bancos de dados PostgreSQL
- Cosmos DB

Aplicativos de OLTP

Aplicativos de OLTP são aplicativos que ajudam no processamento e gerenciamento de transações. Esses aplicativos executam captura de dados, processamento de dados, recuperação, modificação e armazenamento. No entanto, ele não para por aqui. Os aplicativos OLTP tratam estas tarefas de dados como transações. As transações têm algumas propriedades importantes e os aplicativos OLTP aderem a essas propriedades. Essas propriedades são agrupadas com o acrônimo **ACID**. Vamos discutir estas propriedades:

- **Atomicidade:** essa propriedade informa que uma transação deve consistir em instruções e que todas as instruções devem ser concluídas com êxito ou nenhuma instrução deve ser executada. Se várias instruções forem agrupadas juntas, elas formarão a transação. Atomicidade significa que cada transação é tratada como a menor unidade única de execução que é concluída com êxito ou falha.
- **Consistência:** essa propriedade se concentra no estado dos dados em um banco de dados. Ela determina que qualquer alteração no estado deve ser completa e baseada nas regras e restrições do banco de dados, e que atualizações parciais não devem ser permitidas.
- **Isolamento:** essa propriedade informa que pode haver várias transações simultâneas executadas no sistema e cada transação deve ser tratada isoladamente. Uma transação não deve saber sobre a outra ou interferir em nenhuma outra transação. Se as transações tiverem que ser executadas em sequência, no final, o estado dos dados deverá ser o mesmo que antes.
- **Durabilidade:** essa propriedade informa que os dados devem ser persistentes e estar disponíveis, mesmo após uma falha, depois que estão comprometidos com o banco de dados. Uma transação confirmada torna-se um fato.

Bancos de dados relacionais

Os aplicativos OLTP geralmente dependem de bancos de dados relacionais para gerenciamento e processamento de suas transações. Os bancos de dados relacionais geralmente são fornecidos em um formato tabular, que consiste em linhas e colunas. O modelo de dados é convertido em várias tabelas, em que cada tabela está conectada a outra tabela (com base em regras) usando relacionamentos. Este processo também é conhecido como normalização.

Como mencionado anteriormente, o Azure fornece vários bancos de dados relacionais, como SQL Server, MySQL e PostgreSQL.

Modelos de implantação

Há dois modelos de implantação para implementar bancos de dados no Azure:

- Bancos de dados em máquinas virtuais do Azure (IaaS)
- Bancos de dados hospedados como serviços gerenciados (PaaS)

Bancos de dados em máquinas virtuais do Azure

O Azure fornece várias SKUs para máquinas virtuais. Há máquinas de alta computação e alta taxa de transferência (IOPS) que também estão disponíveis junto com máquinas virtuais de uso geral. Em vez de hospedar um SQL Server, MySQL ou qualquer outro banco de dados em servidores na infraestrutura local, é possível implantar esses bancos de dados nessas máquinas virtuais. A implantação e a configuração desses bancos de dados não são diferentes das implantações na infraestrutura local. A única diferença é que o banco de dados é hospedado na nuvem, não em servidores na infraestrutura local. Os administradores devem executar as mesmas atividades e etapas que normalmente executariam para a implantação na infraestrutura local. Embora esta opção seja ideal quando os clientes desejam ter controle total sobre sua implantação, há modelos que podem ser mais econômicos, escalonáveis e altamente disponíveis em comparação a essa opção.

As etapas para implantar qualquer banco de dados em máquinas virtuais do Azure são as seguintes:

1. Crie uma máquina virtual com um tamanho que atenda aos requisitos de performance do aplicativo.
2. Implante um banco de dados sobre ela.
3. Defina a configuração da máquina virtual e do banco de dados.

Esta opção não fornece nenhuma alta disponibilidade pronta para uso, a menos que vários servidores sejam provisionados e o SQL Server esteja definido com uma configuração **AlwaysOn**. Ela também não fornece recursos para escala automática, a menos que a automação personalizada dê suporte a ela.

A Disaster Recovery também é responsabilidade do cliente e deve implantar servidores em várias regiões conectadas usando o emparelhamento global ou os gateways de rede VPN. É possível que essas máquinas virtuais sejam conectadas a um data center na infraestrutura local por meio de VPNs site a site ou ExpressRoute sem ter nenhuma exposição ao mundo externo.

Esses bancos de dados também são conhecidos como **bancos de dados não gerenciados**.

Bancos de dados hospedados como serviços gerenciados

Os bancos de dados hospedados com o Azure, diferentes das máquinas virtuais, são gerenciados pelo Azure e são conhecidos como **serviços gerenciados**. Serviços gerenciados significa que o Azure fornece serviços de gerenciamento para os bancos de dados. Esses serviços gerenciados incluem a hospedagem do banco de dados, garantindo que o host esteja altamente disponível, garantindo que os dados sejam replicados internamente para disponibilidade durante Disaster Recovery, garantindo a escalabilidade dentro da restrição de um SKU escolhido, monitorando os hosts e bancos de dados e gerando alertas para notificações ou executando ações, fornecendo serviços de log e auditoria para solução de problemas e cuidando do gerenciamento de desempenho e dos alertas de segurança.

Para resumir, existem muitos serviços que os clientes disponibilizam quando usam serviços gerenciados do Azure e não precisam executar o gerenciamento ativo nesses bancos de dados. Neste capítulo, analisaremos o SQL do Azure em detalhes e forneceremos informações sobre outros bancos de dados, como Cosmos DB, MySQL e Postgre.

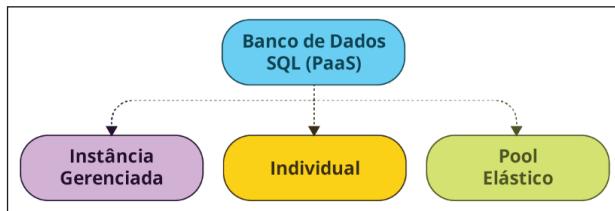
Banco de dados SQL do Azure

O servidor SQL do Azure fornece um banco de dados relacional hospedado como uma PaaS. Os clientes podem provisionar esse serviço, trazer seus próprios dados e esquemas de banco de dados e conectar seus aplicativos a ele. Ele fornece todos os recursos do SQL Server que você recebe ao implantar em uma máquina virtual. Esses serviços não fornecem uma interface de usuário para criar tabelas e esquema correspondente, nem fornecem recursos de consulta diretamente. Você deve usar o SQL Server Management Studio e as ferramentas de CLI do SQL para se conectar a esses serviços e trabalhar diretamente com eles.

O Banco de Dados SQL do Azure é fornecido com três modelos de implantação distintos:

- **Instância Única:** neste modelo de implantação, um único banco de dados é implantado em um servidor lógico. Isso envolve a criação de dois recursos no Azure:
 - Servidor lógico SQL
 - Banco de dados SQL
- **Pool elástico:** nesse modo de implantação, vários bancos de dados são implantados em um servidor lógico. Novamente, Isso envolve a criação de dois recursos no Azure:
 - Servidor lógico SQL
 - Pool de banco de dados elástico SQL – consiste em todos os bancos de dados

- **Instância Gerenciada:** um modelo de implantação relativamente novo da equipe de SQL do Azure. Essa implantação reflete uma coleção de bancos de dados em um servidor lógico, fornecendo controle completo sobre os recursos no que diz respeito aos bancos de dados do sistema. Geralmente, os bancos de dados do sistema não são visíveis em outros modelos de implantação, mas estão disponíveis no modelo. Esse modelo é muito parecido com a implantação do SQL Server na infraestrutura local:



Recursos do aplicativo

O Banco de Dados SQL do Azure fornece vários recursos específicos do aplicativo que atendem aos diferentes requisitos dos sistemas OLTP:

- **Armazenamento colunar:** esse recurso permite que o armazenamento de dados seja em um formato colunar, em vez de em um formato de linha.
- **OLTP in-memory:** geralmente, os dados são armazenados em arquivos de back-end em SQL e são obtidos dos arquivos sempre que for necessário pelo aplicativo. Ao contrário disso, o OLTP in-memory coloca todos os dados na memória e não há latência na leitura do armazenamento de dados. O armazenamento de dados OLTP in-memory no SSD proporciona a melhor performance possível para o SQL do Azure.
- Todos os recursos do SQL Server na infraestrutura local.

Instância única

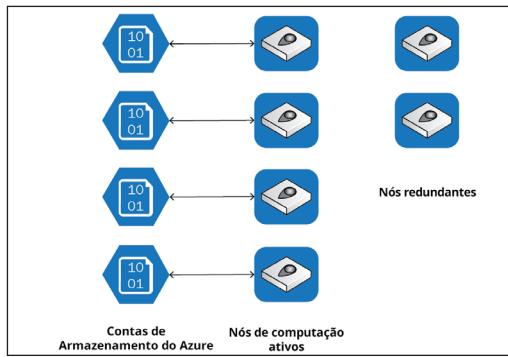
Os bancos de dados de Instância Única são hospedados como um banco de dados único em um servidor lógico único. Esses bancos de dados não têm acesso aos recursos completos fornecidos pelo SQL Server.

Alta disponibilidade

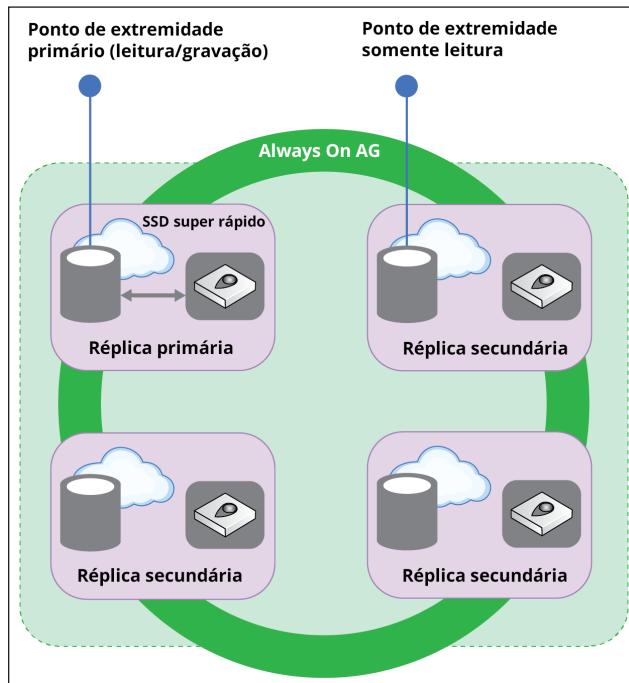
Por padrão, o SQL do Azure é 99,99% altamente disponível. Ele tem duas arquiteturas diferentes para manter a alta disponibilidade com base nas SKUs. Para os SKUs Básico, Padrão e Geral, toda a arquitetura é dividida nas duas camadas a seguir.

Há redundância incorporada para essas duas camadas fornecerem alta disponibilidade:

- Camada de computação
- Camada de armazenamento:



Para os SKUs Premium e críticos para os negócios, a computação e o armazenamento estão na mesma camada. A alta disponibilidade é alcançada pela replicação de computação e armazenamento implantada em um cluster de quatro nós, usando tecnologia semelhante aos Grupos de Disponibilidade AlwaysOn do SQL Server:



Backups

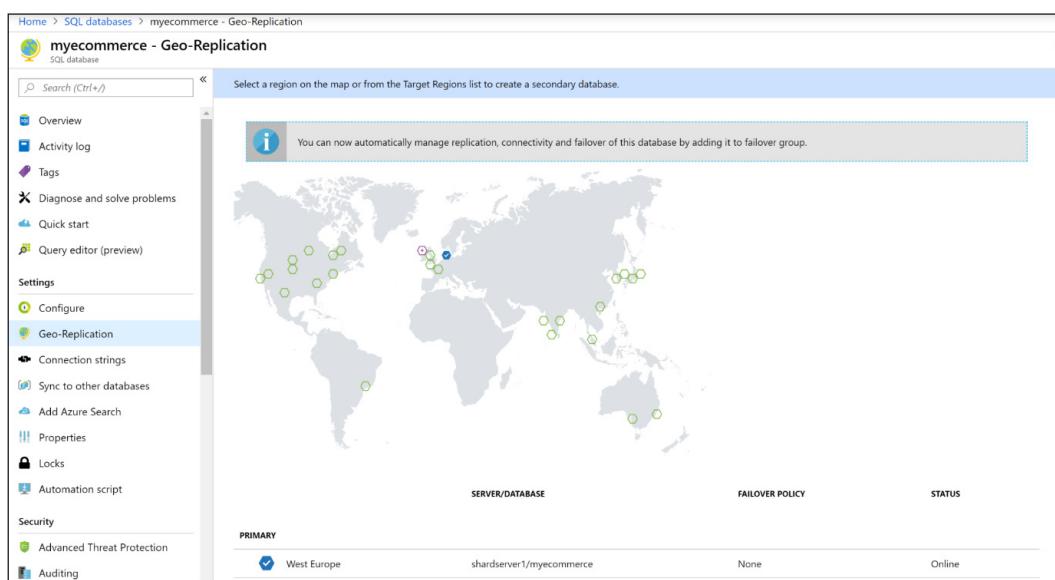
O SQL Azure também fornece recursos para fazer backup automático de bancos de dados e armazená-los em contas de armazenamento. Esse recurso é importante principalmente nos casos em que um banco de dados se torna corrompido ou um usuário exclui acidentalmente uma tabela. Esse recurso está disponível no nível do servidor, conforme mostrado na captura de tela a seguir:

The screenshot shows the Azure portal interface for managing backups. On the left, there's a sidebar with various options like Overview, Activity log, Access control (IAM), Tags, and Diagnose and solve problems. The 'Manage Backups' option is selected. The main area shows 'shardserver1 - Manage Backups' with a search bar and navigation buttons for Configure policies and Available backups. A table lists databases with their backup retention settings. One row for 'myecommerce' is highlighted, showing 'PITR BACKUPS' set to '35 days'. To the right, a modal dialog titled 'Configure policies' is open, specifically for 'Point In Time Restore Configuration'. It includes fields for 'Configure PiTR backup retention' (set to 'Days') and three sections for Long-term Retention Configurations: 'Weekly LTR Backups', 'Monthly LTR Backups', and 'Yearly LTR Backups'. Each section has a dropdown for 'How long would you like [] backups to be kept?' and a dropdown for 'Which weekly backup of the year would you like to retain?'. At the bottom of the modal are 'Apply' and 'Cancel' buttons.

Os arquitetos devem preparar uma estratégia de backup para que os backups possam ser usados em momentos de necessidade. Eles não devem ser muito antigos, nem devem ser configurados para backups muito frequentes. Com base nas necessidades do negócio, um backup semanal ou até mesmo um backup diário deve ser configurado, ou ainda mais frequentemente do que isso, se necessário. Esses backups podem ser usados para fins de restauração.

Replicação geográfica

O SQL do Azure também fornece o benefício de ser capaz de replicar o banco de dados para uma região diferente, também conhecido como uma região secundária. O banco de dados na região secundária pode ser lido por aplicativos. Ele permite bancos de dados secundários legíveis. Esta é uma ótima solução de continuidade dos negócios, pois o banco de dados legível está disponível em qualquer momento. Com a replicação geográfica, é possível ter até quatro regiões secundárias de um banco de dados em diferentes regiões ou na mesma região. Estes são bancos de dados legíveis. Com a replicação geográfica, também é possível realizar o failover para um banco de dados secundário no caso de um desastre. A replicação geográfica é configurada no nível do banco de dados, conforme mostrado na captura de tela a seguir:



Soluções OLTP do Azure que usam fragmentação, pools e híbrido de SQL do Azure

Se você rolar para baixo nessa tela, as regiões que podem agir como secundárias são listadas, conforme mostrado na seguinte captura de tela:

The screenshot shows the Azure portal interface for managing a SQL database named "myecommerce". The left sidebar contains navigation links for Overview, Activity log, Tags, Diagnose and solve problems, Quick start, Query editor (preview), Settings (Configure, Geo-Replication, Connection strings, Sync to other databases, Add Azure Search, Properties, Locks, Automation script), Security (Advanced Threat Protection, Auditing), and a search bar.

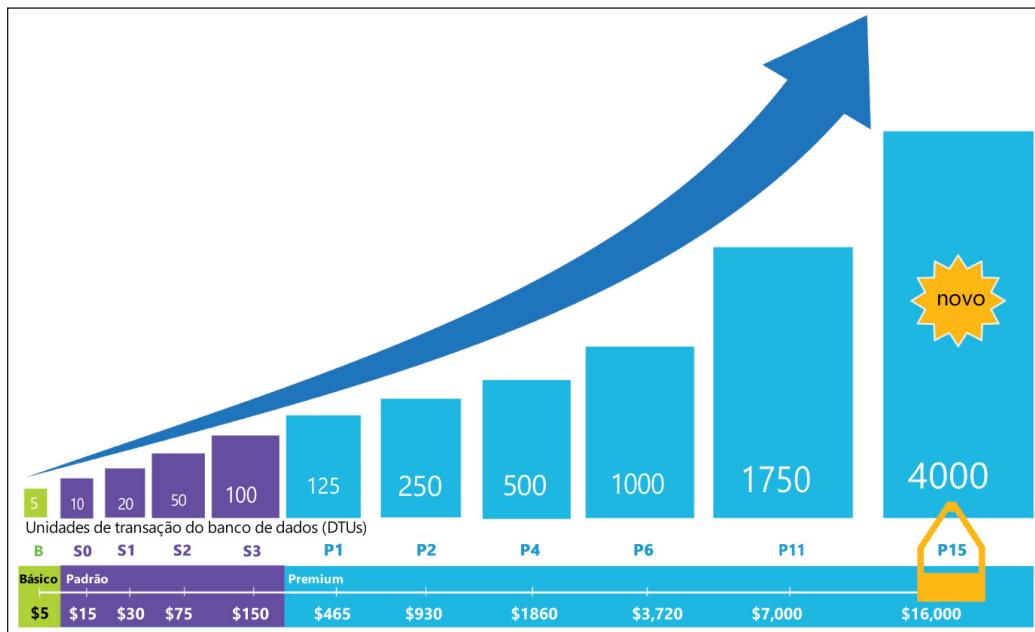
The main content area displays the "Geo-Replication" blade. At the top, it says "Select a region on the map or from the Target Regions list to create a secondary database." Below this, the "PRIMARY" section shows "West Europe" as the selected region, associated with the server "shardserver1/myecommerce", with a "None" failover policy and an "Online" status. The "SECONDARIES" section indicates "Geo-Replication is not configured".

The "TARGET REGIONS" section lists the following regions, each represented by a green hexagonal icon:

- West US
- West US 2
- Central US
- West Central US
- South Central US
- North Central US
- Canada Central
- East US
- Canada East

Escalabilidade

O SQL Azure fornece escalabilidade vertical adicionando mais recursos (como computação, memória e IOPS). Isso pode ser feito aumentando o número de DTUs provisionadas para ele. O SQL de instância única fornece recursos para escala manual apenas:



Segurança

A segurança é um fator importante para qualquer solução de banco de dados e serviço. O SQL do Azure fornece segurança de nível empresarial para o SQL do Azure, e esta seção listará alguns dos recursos de segurança importantes no SQL do Azure.

Firewall

Por padrão, o SQL do Azure não fornece acesso a nenhuma solicitação. Os endereços IP de origem devem ser explicitamente incluídos na lista de permissões para acessar o SQL Server. Há uma opção para permitir que todos os serviços baseados no Azure também acessem o banco de dados SQL. Essa opção inclui máquinas virtuais hospedadas no Azure.

O firewall pode ser configurado no nível do servidor, em vez do nível do banco de dados. A opção **Permitir o acesso aos serviços do Azure** permite que todos os serviços, incluindo máquinas virtuais, acessem o banco de dados hospedado no servidor lógico:

shardserver1 - Firewalls and virtual networks

SQL databases

SQL elastic pools

Deleted databases

Import/Export history

DTU quota

Properties

Locks

Automation script

Security

Save Discard Add client IP

Connections from the IPs specified below provides access to all the databases in shardserver1.

Allow access to Azure services

ON OFF

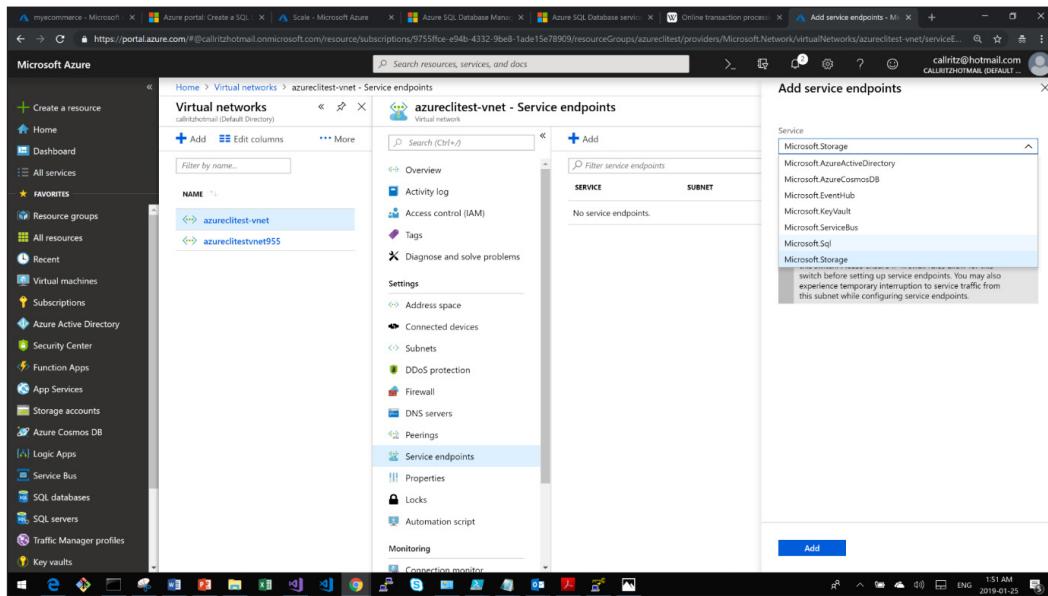
Client IP address 223.230.50.55

RULE NAME	START IP	END IP	...
ClientIPAddress_2019-1-25...	223.230.50.55	223.230.50.55	...
ClientIPAddress_2019-1-16...	182.66.149.67	182.66.149.67	...

Azure SQL Server em redes dedicadas

Embora o acesso ao SQL Server esteja geralmente disponível pela Internet, é possível que o acesso ao SQL Server possa ser limitado a solicitações resultantes de redes virtuais. Este é um recurso relativamente novo no Azure. Isso ajuda no acesso a dados no SQL Server do aplicativo em outro servidor da rede virtual sem a solicitação passar pela Internet.

Para isso, um ponto de extremidade de serviço do tipo `Microsoft.Sql` deve ser adicionado dentro da rede virtual e a rede virtual deve estar na mesma região do SQL do Azure:

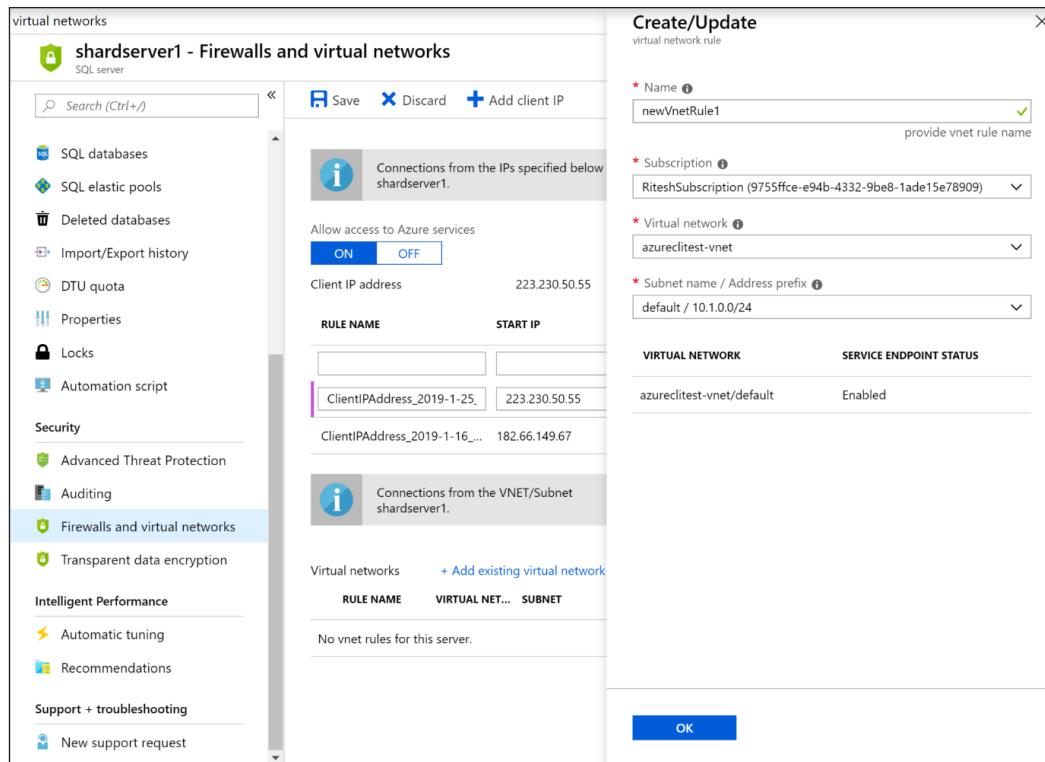


Soluções OLTP do Azure que usam fragmentação, pools e híbrido de SQL do Azure

Uma sub-rede apropriada dentro das redes virtuais deve ser escolhida:

The screenshot shows the 'Service endpoints' blade for the 'azureclitest-vnet' virtual network. The left sidebar lists various settings like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Address space, Connected devices, Subnets, DDoS protection, Firewall, DNS servers, Peerings, Service endpoints (which is selected), Properties, Locks, and Automation script. The main area shows a table with columns 'SERVICE' and 'SUBNET'. A note at the bottom right of the blade states: "rules using Azure public IP addresses will stop working with this switch. Please ensure IP firewall rules allow for this switch before setting up service endpoints. You may also experience temporary interruption to service traffic from this subnet while configuring service endpoints."

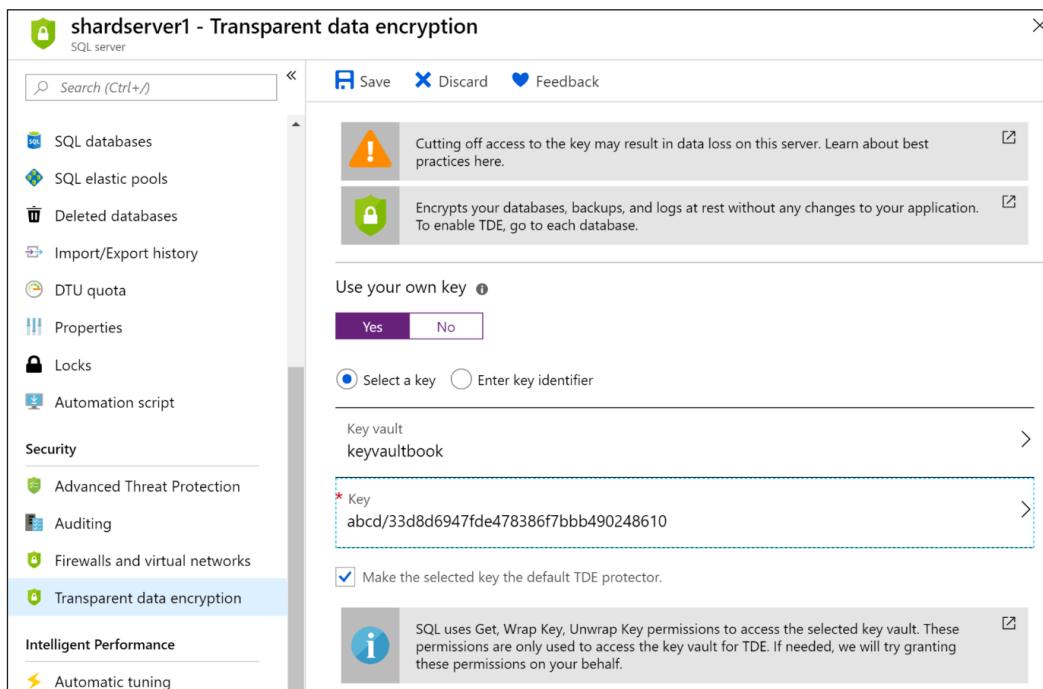
Por fim, da folha de configuração do Azure SQL Server, deve ser adicionada uma rede virtual existente que tenha um ponto de extremidade **Microsoft.Sql** habilitado:



Bancos de dados criptografados em repouso

Os bancos de dados devem estar em um formato criptografado quando estão em repouso. **Repouso** aqui significa o local de armazenamento do banco de dados. Embora você possa não ter acesso ao SQL Server e seu banco de dados, é preferível criptografar o armazenamento do banco de dados.

Os bancos de dados no sistema de arquivos podem ser criptografados usando chaves. Essas chaves devem ser armazenadas no Azure Key Vault e o cofre deve estar disponível na mesma região do Azure SQL Server. O sistema de arquivos pode ser criptografado usando o item de menu **Transparent data encryption** da folha de configuração do SQL Server e selecionando **Sim para Usar sua própria chave**. A chave é uma chave RSA 2048 e deve existir no cofre. O SQL Server descriptografará os dados no nível da página quando quiser lê-los e enviá-los para o chamador e, em seguida, irá criptografá-los depois de gravar no banco de dados. Nenhuma alteração nos aplicativos é necessária e isso é completamente transparente para eles:



Máscara de dados dinâmicos

O SQL Server também fornece um recurso que mascara colunas individuais que contêm dados confidenciais para que ninguém, além dos usuários privilegiados, possa exibir os dados reais consultando-os no SQL Server Management Studio. Os dados permanecerão mascarados e só serão desmascarados quando um aplicativo ou usuário autorizado consultar a tabela. Os arquitetos devem garantir que dados confidenciais, como detalhes de cartão de crédito, números de segurança social, números de telefone, endereços de e-mail e outros detalhes financeiros, devem ser mascarados.

Integração do Azure Active Directory

Outro importante recurso de segurança do SQL do Azure é que ele pode ser integrado ao **Azure Active Directory (AD)** para fins de autenticação. Sem integração com o Azure AD, o único mecanismo de autenticação disponível para o SQL Server é via autenticação de nome de usuário e senha; ou seja, a autenticação SQL. Não é possível usar a autenticação integrada do Windows. A cadeia de conexão para autenticação SQL consiste em nome de usuário e senha em texto não criptografado, o que não é seguro. A integração com o Azure AD habilitou a autenticação do aplicativo com a autenticação do Windows, o nome da entidade de serviço ou baseada em token. É recomendável usar o SQL do Azure integrado ao Azure AD.

Há outros recursos de segurança, como proteção avançada contra ameaças, auditoria do ambiente e monitoramento, que devem ser habilitados em todas as implantações do Azure SQL de nível empresarial:

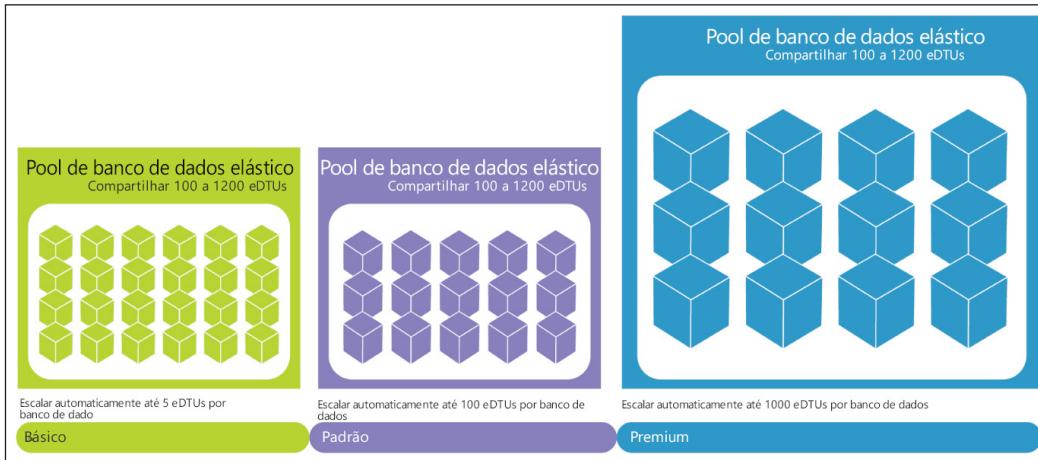
Pools elásticos

Um pool elástico é um contêiner lógico que pode hospedar vários bancos de dados em um único servidor lógico. Os SKUs disponíveis para pools elásticos são os seguintes:

- Básico
- Padrão
- Premium

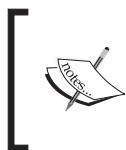
Soluções OLTP do Azure que usam fragmentação, pools e híbrido de SQL do Azure

A captura de tela a seguir mostra a quantidade máxima de DTUs que podem ser provisionadas para cada SKU:



Todos os recursos discutidos para Instâncias Únicas do SQL Azure também estão disponíveis para pools elásticos. No entanto, a escalabilidade horizontal é um recurso adicional que ele fornece com a ajuda do sombreamento. Sombreamento refere-se ao particionamento vertical ou horizontal de dados e ao armazenamento deles em bancos de dados separados. Também é possível ter a escala automática de bancos de dados individuais em um pool elástico consumindo mais DTUs do que na verdade são alocadas para esse banco de dados.

Os pools elásticos também proporcionam outra vantagem em termos de custo. Veremos em uma seção posterior que o preço do SQL do Azure é fixado usando o conceito de DTUs, e as DTUs são provisionadas assim que o serviço do SQL Server é provisionado. As DTUs são cobradas independentemente de serem consumidas ou não. Se houver vários bancos de dados, será possível colocar esses bancos de dados em pools elásticos e compartilhar as DTUs entre eles.

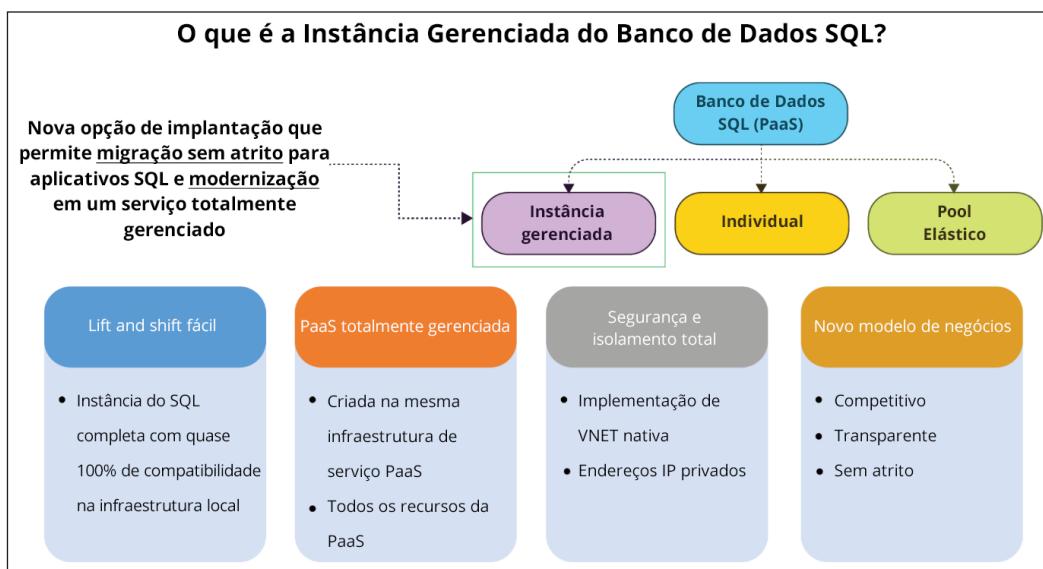


Todas as informações para implementar a fragmentação com pools elásticos do SQL do Azure foram fornecidas em <https://docs.microsoft.com/en-us/azure/sql-database/sql-database-elastic-scale-introduction>.

Instância Gerenciada

Instância Gerenciada é um serviço exclusivo que fornece um servidor SQL gerenciado semelhante ao que está disponível nos servidores na infraestrutura local. Os usuários têm acesso ao mestre, modelo e outros bancos de dados do sistema. O uso da Instância Gerenciada é adequado quando há vários bancos de dados e clientes que migram suas instâncias para o Azure. A Instância Gerenciada consiste em vários bancos de dados.

O Banco de Dados SQL do Azure fornece um novo modelo de implantação conhecido como Instância Gerenciada do Banco de Dados SQL do Azure que fornece quase 100% de compatibilidade com o Mecanismo de Banco de Dados da Edição Enterprise do SQL Server. Esse modelo fornece uma implementação de VNet nativa que aborda os problemas de segurança habituais e é um modelo de negócios altamente recomendado para clientes do SQL Server na infraestrutura local. A Instância Gerenciada permite que os clientes existentes do SQL Server façam o "lift and shift" de seus aplicativos na infraestrutura local para a nuvem com alterações mínimas de aplicativo e banco de dados, preservando todos os recursos de PaaS ao mesmo tempo. Esses recursos de PaaS reduzem drasticamente a sobrecarga de gerenciamento e o custo total de propriedade, conforme mostrado na captura de tela a seguir:



Soluções OLTP do Azure que usam fragmentação, pools e híbrido de SQL do Azure

Os principais recursos de Instância Gerenciada são mostrados na captura de tela a seguir:

Esta tabela mostra os principais recursos da Instância Gerenciada:	
Recurso	Descrição
Versão/build do SQL Server	Mecanismo BD SQL Server (recém-estável)
Backups automatiz. gerenc.	Sim
Inst. inc. e monit. e métr. bd	Sim
Patch de software automático	Sim
Recursos recentes de Mec. BD	Sim
Nº arq. dados (LINHAS) por bd	Vários
Nº arq. log (LOG) por bd	1
VNet - implant. Azure Resource Manager	Sim
VNet - Modelo clássico implant.	Não
Supporte ao portal	Sim
Serviço de integr. incorp. (SSIS)	Não - SSIS faz parte do Azure Data Factory PaaS
Serviço de análise incorp. (SSAS)	Não - SSAS separado do PaaS
Serviço de relatório incorp. (SSRS)	Não - use PowerBI ou SSRS IaaS

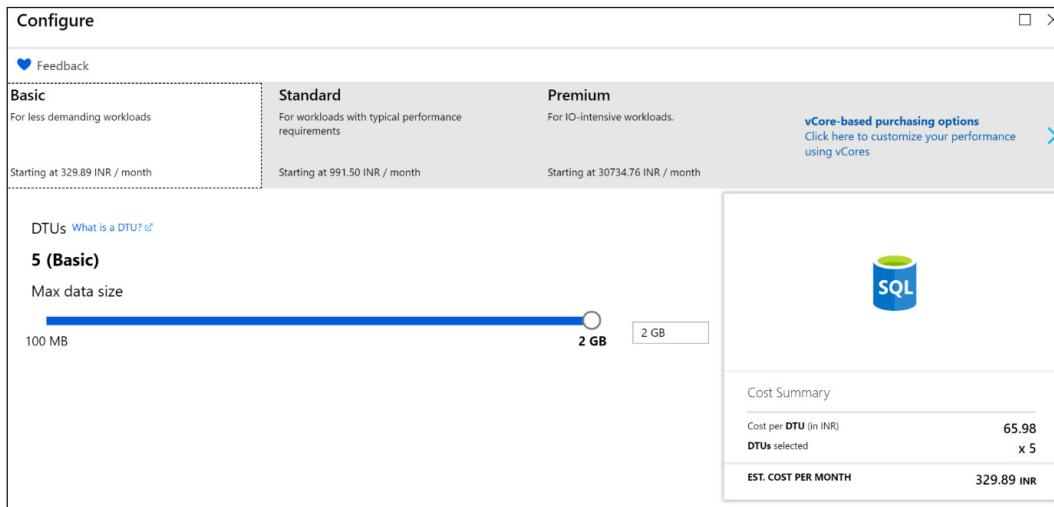
Preços do banco de dados SQL

O SQL do Azure anteriormente tinha apenas um modelo de preços – um modelo baseado em **Unidades de Taxa de Transferência de Banco de Dados (DTUs)** e um modelo de preços alternativo mais recente baseado em vCPUs também foram lançados.

Preços baseados em DTU

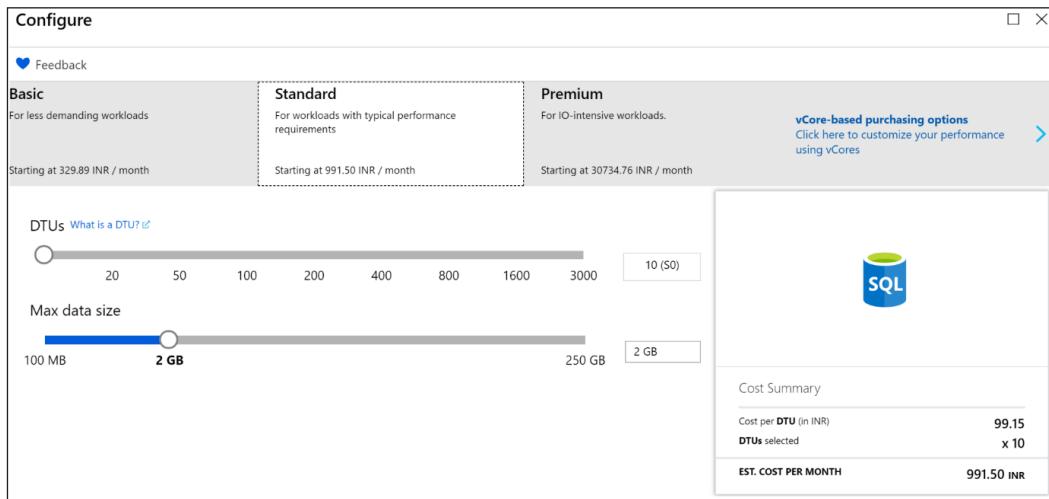
A DTU é a menor unidade de medida de performance para o SQL do Azure. Cada DTU corresponde a uma certa quantidade de recursos. Esses recursos incluem armazenamento, ciclos de CPU, IOPS e largura de banda da rede. Por exemplo, uma DTU única pode fornecer três IOPS, alguns ciclos de CPU e uma latência de E/S de 5 ms para operações de leitura e 10 ms para operações de gravação.

O SQL do Azure fornece vários SKUs para criar bancos de dados, e cada um deles definiu restrições para a quantidade máxima de DTUs. Por exemplo, o SKU Básico fornece apenas 5 DTUs com um máximo de 2 GB de dados, como mostrado na seguinte captura de tela:



Soluções OLTP do Azure que usam fragmentação, pools e híbrido de SQL do Azure

Por outro lado, o SKU padrão fornece qualquer coisa entre 10 DTUs e 300 DTUs com um máximo de 250 GB de dados. Como você pode ver aqui, cada DTU custa cerca de 999 rupias ou em torno de US\$ 1,40:



Uma comparação entre esses SKUs quanto a performance e recursos é fornecida pela Microsoft, conforme mostrado na seguinte captura de tela:

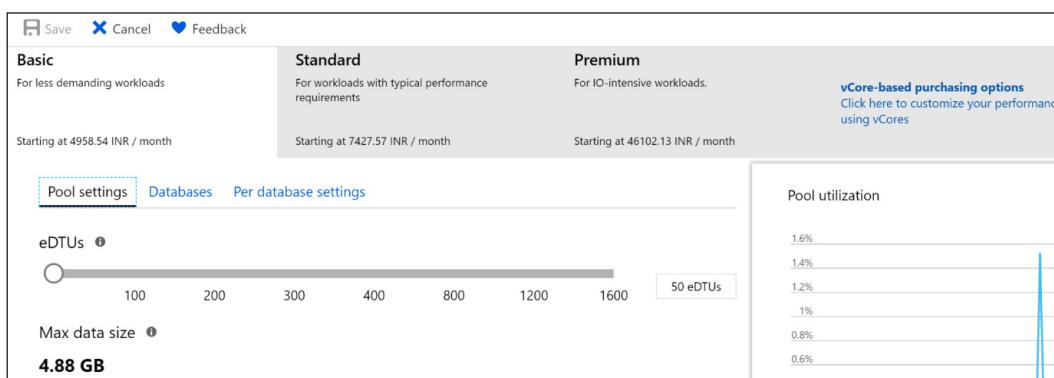
	Basic	Standard	Premium
Workload de destino	Desenv. e produção	Desenv. e produção	Desenv. e produção
SLA de tempo de ativ.	99,99%	99,99%	99,99%
Retenção de backup	7 dias	35 dias	35 dias
CPU	Baixo	Baixo, Médio, Alto	Médio, Alto
Taxa transf. E/S (aprox.)	2.5 IOPS por DTU	2.5 IOPS por DTU	48 IOPS por DTU
Latência E/S (aprox.)	5 ms (leitura), 10 ms (grav.)	5 ms (leitura), 10 ms (grav.)	2 ms (leitura, gravação)
Indexação columnstore	N/A	53 e acima	Com suporte
OLTP in-memory	N/A	N/A	Com suporte

Depois de provisionar um determinado número de DTUs, os recursos de back-end (CPU, IOPS e memória) são alocados e cobrados, quer sejam consumidos ou não. Se forem adquiridas mais DTUs do que são realmente necessárias, isso levará a um desperdício, resultando em gargalos de performance se não forem provisionadas DTUs suficientes.

O Azure fornece pools elásticos para essa finalidade também. Como sabemos, há vários bancos de dados em um pool elástico e as DTUs são atribuídas a pools elásticos, em vez de bancos de dados individuais. É possível que todos os bancos de dados no pool compartilhem as DTUs. Isso significa que, se um banco de dados tiver baixa utilização e estiver consumindo 5 DTUs, haverá outro banco de dados consumindo 25 DTUs para compensar.

É importante observar que, coletivamente, o consumo de DTUs não pode exceder a quantidade de DTUs provisionadas para o pool elástico. Além disso, há uma quantidade mínima de DTUs que devem ser atribuídas a cada banco de dados no pool elástico, e essa contagem mínima de DTUs é pré-alocada para o banco de dados.

Um pool elástico é fornecido com seus próprios SKUs:



Além disso, há uma limitação na quantidade máxima de bancos de dados que podem ser criados em um único pool elástico.

Preços baseados na vCPU

Este é o novo modelo de preços para o SQL do Azure. Nesse modelo de preços, em vez de identificar a quantidade de DTUs necessárias para um aplicativo, ele fornece opções para adquirir o número de **CPUs virtuais (vCPUs)** alocadas para o servidor. Uma vCPU é uma CPU lógica com hardware conectado, como armazenamento, memória e núcleos da CPU.

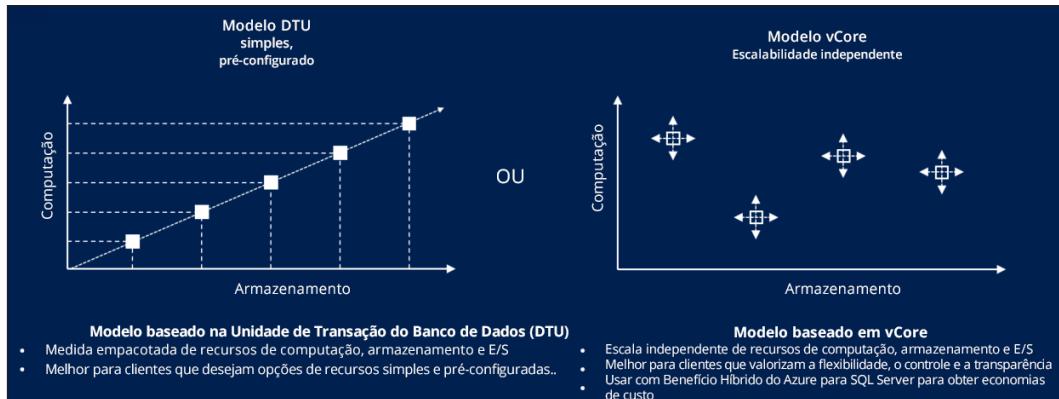
Neste modelo, existem três SKUs: **Uso Geral**, **Hiperescala** e **Comercialmente Crítico**, com um número variado de vCPUs e recursos disponíveis. Este preço está disponível para todos os modelos de implantação de SQL:

The screenshot shows the Azure portal's configuration interface for a new SQL database. In the top navigation bar, there are icons for 'Feedback' and a search bar containing 'Looking for basic, standard, premium?'. Below this, there are three categories: 'General Purpose' (scalable compute and storage options), 'Hyperscale' (on-demand scalable storage), and 'Business Critical' (high transaction rate and high resiliency). The 'General Purpose' section is selected. Under 'Compute Generation', 'Gen5' is chosen over 'Gen4'. Configuration sliders are shown for 'vCores' (set to 2) and 'Max data size' (set to 32 GB). On the right, a summary box displays the 'Cost Summary' for the selected configuration: 'Gen5 - General Purpose (GP_Gen5_2)', 'Cost per vCore (in INR) 8233.91', 'vCores selected x 2', 'Cost per GB (in INR) 9.05', 'Max storage selected (in GB) x 41.6', and 'EST. COST PER MONTH 16844.11 INR'.

Como escolher o modelo de preços apropriado

Os arquitetos devem ser capazes de escolher um modelo de preços apropriado para o SQL do Azure. As DTUs são um grande mecanismo para fixação de preços em que há um padrão de uso aplicável e disponível para o banco de dados. Como a disponibilidade de recurso no esquema da DTU das coisas é linear, como mostrado no diagrama a seguir, é bem possível que o uso seja mais intensivo da memória do que intensivo da CPU. Nesses casos, é possível escolher diferentes níveis de CPU, memória e armazenamento para um banco de dados. Nas DTUs, os recursos vêm empacotados, e não é possível configurar esses recursos em um nível granular. Com um modelo vCPU, é possível escolher diferentes níveis de memória e CPU para diferentes bancos de dados. Se o padrão de uso para um aplicativo for conhecido, usar o modelo de preços da vCPU poderá ser uma opção melhor em comparação com o modelo de DTU. Na verdade, o modelo vCPU também fornece o benefício de licenças híbridas se uma organização já tem licenças do SQL Server na infraestrutura local. Um desconto de até 30% é fornecido para essas instâncias do SQL Server.

Na captura de tela a seguir, você pode ver no gráfico à esquerda que, à medida que a quantidade de DTUs aumenta, a disponibilidade de recursos também cresce linearmente; no entanto, com o preço da vCPU (no gráfico à direita), é possível escolher configurações independentes para cada banco de dados:



Resumo

O SQL do Azure é um dos principais serviços do Azure. Milhões de clientes estão usando esse serviço atualmente e ele oferece todos os recursos empresariais necessários para um sistema de gerenciamento de banco de dados de missão crítica. Há vários tipos de implantação para o SQL do Azure, como Instância Única, Instância Gerenciada e pools elásticos. Os arquitetos devem fazer uma avaliação completa de seus requisitos e escolher o modelo de implantação apropriado. Depois de escolher um modelo de implantação, os arquitetos devem escolher uma estratégia de preços entre DTUs e vCPUs. Eles também devem configurar todos os requisitos de segurança, disponibilidade, Disaster Recovery, monitoramento, performance e escalabilidade no SQL do Azure em relação aos dados.

13

Soluções de Big Data do Azure com Azure Data Lake Storage e Data Factory

O big data vem ganhando uma força significativa ao longo dos últimos anos. Ferramentas especializadas, software e armazenamento são necessários para lidar com ele. Curiosamente, essas ferramentas, plataformas e opções de armazenamento não estavam disponíveis como serviços há alguns anos. No entanto, com a nova tecnologia de nuvem, o Azure fornece inúmeras ferramentas, plataformas e recursos para criar soluções de Big Data com facilidade.

Os tópicos a seguir serão abordados neste capítulo:

- Integração de dados
- **Extrair-Transformar-Carregar (ETL)**
- Data Factory
- Data Lake Storage
- Migração de dados do Armazenamento do Azure para o Data Lake Storage

Integração de dados

Estamos cientes de como os padrões de integração são usados para aplicativos. Os aplicativos compostos de vários serviços são integrados usando vários padrões. No entanto, há outro paradigma que é um requisito fundamental para muitas organizações, conhecido como integração de dados. Este aumento na integração de dados aconteceu principalmente durante a última década, quando a geração e a disponibilidade de dados foram incrivelmente altas. A velocidade, a variedade e o volume de dados que estão sendo gerados aumentaram drasticamente, e há dados quase em todos os lugares.

Cada organização tem muitos tipos diferentes de aplicativos, e todos eles geram dados em seus próprios formatos. Muitas vezes, os dados também são adquiridos do mercado. Mesmo durante fusões e integrações de organizações, os dados precisam ser migrados e combinados.

A integração de dados refere-se ao processo de trazer dados de várias fontes e gerar uma nova saída que tenha mais significado e usabilidade.

Há uma necessidade absoluta de integração de dados para os seguintes cenários:

- Para migrar dados de uma origem ou de um grupo de origens para um destino-alvo. Isso é necessário para disponibilizar dados em diferentes formatos para diferentes partes interessadas e consumidores.
- Com a rápida disponibilidade de dados, as organizações desejam obter insights deles. Elas desejam criar soluções que fornecem insights; dados de várias fontes devem ser mesclados, limpos, aumentados e armazenados em um data warehouse.
- Para gerar painéis e relatórios em tempo real.
- Para criar soluções de análise.

A integração de aplicativos apresenta um comportamento de tempo de execução quando os usuários estão consumindo o aplicativo; por exemplo, no caso de validação e integração de cartão de crédito. Por outro lado, a integração de dados acontece como um exercício de back-end e não está diretamente vinculada à atividade do usuário.

ETL

Um processo muito popular, conhecido como ETL, ajuda na criação de uma fonte de dados de destino para dados internos que são consumíveis por aplicativos. Geralmente, os dados estão em um formato bruto e, para que se tornem consumíveis, devem passar pelas três fases distintas a seguir:

- **Extrair:** durante essa fase, os dados são extraídos de vários lugares. Por exemplo, pode haver várias fontes e todas precisam estar conectadas para recuperar os dados. As fases de extração normalmente usam conectores de dados que consistem em informações de conexão relacionadas à fonte de dados de destino. Elas também podem ter armazenamento temporário para trazer os dados da fonte e armazená-los para uma recuperação mais rápida. Essa fase é responsável pela ingestão de dados.
- **Transformar:** os dados que ficam disponíveis após a fase de extração podem não ser diretamente consumíveis por aplicativos. Isso pode ocorrer por vários motivos; por exemplo, os dados podem ter irregularidades, pode haver dados ausentes ou pode haver dados errados. Ou pode até mesmo haver dados que não são necessários.

Alternativamente, o formato dos dados pode não ser apropriado para consumo pelos aplicativos de destino. Em todos esses casos, a transformação deve ser aplicada aos dados de forma que possa ser consumida de forma eficiente pelos aplicativos.

- **Carregar:** após a transformação, os dados devem ser carregados para a fonte de dados de destino em um formato e esquema que permitam uma disponibilidade mais rápida, fácil e centrada na performance para aplicativos. Mais uma vez, isso geralmente consiste em conectores de dados para fontes de dados de destino e carregamento de dados para elas.

Uma cartilha sobre o Data Factory

O Data Factory é uma ferramenta totalmente gerenciada, altamente disponível e escalável e fácil de usar para criar soluções de integração e implementar fases de ETL. O Data Factory ajuda você a criar novos pipelines em uma forma de arrastar e soltar usando uma interface de usuário, sem a necessidade escrever código; no entanto, ele ainda fornece recursos para permitir que você escreva código em seu idioma preferido.

Existem alguns conceitos importantes para aprender antes de usar o serviço Data Factory, os quais exploraremos em mais detalhes nas seções a seguir:

- **Atividades:** tarefas individuais que permitem a execução e o processamento da lógica em um pipeline de Data Factory. Existem vários tipos de atividades. Há atividades relacionadas a movimentação de dados, transformação de dados e atividades de controle. Cada atividade tem uma política por meio da qual é possível decidir o mecanismo e o intervalo de repetição.
- **Pipelines:** no Data Factory, eles são compostos por grupos de atividades e são responsáveis por reunir as atividades. Pipelines são os fluxos de trabalho e os orquestradores que permitem a execução das fases de ETL. Os pipelines permitem a união de atividades e a declaração de dependências entre elas. Usando dependências, é possível executar algumas tarefas em paralelo e outras em sequência.
- **Conjuntos de dados:** são as origens e os destinos dos dados. Eles podem ser as contas de Armazenamento do Azure, o Data Lake Storage ou um host de outras origens.
- **Serviços vinculados:** serviços que contêm as informações de conexão e conectividade para os conjuntos de dados e são utilizados por tarefas individuais para se conectar a elas.

- **Tempo de execução de integração:** o mecanismo principal responsável pela execução do Data Factory é chamado de tempo de execução de integração. O tempo de execução de integração está disponível nas três configurações a seguir:
 - **Azure:** nesta configuração, o Data Factory é executado nos recursos de computação fornecidos pelo Azure.
 - **Auto-hospedado:** nesta configuração, o Data Factory é executado quando você traz seus próprios recursos de computação. Isso pode acontecer por meio de servidores de máquinas virtuais baseados na nuvem ou na infraestrutura local.
 - **Azure SQL Server Integration Services (SSIS):** esta configuração permite a execução de pacotes SSIS tradicionais gravados usando o SQL Server.
- **Versões:** o Data Factory é fornecido com duas versões diferentes. É importante entender que todos os novos desenvolvimentos ocorrerão na V2 e que a V1 permanecerá como está, ou desaparecerá em algum momento. V2 é a versão preferencial devido aos seguintes motivos:
 - Fornece a capacidade para executar os pacotes de integração do SQL Server.
 - Tem funcionalidades melhoradas em comparação com a V1.
 - É fornecida com monitoramento aprimorado, o que está faltando na V1.

Uma cartilha sobre o Data Lake Storage

O Azure Data Lake Storage fornece armazenamento para soluções de big data. Ele foi especialmente projetado para armazenar as grandes quantidades de dados que geralmente são necessários em soluções de big data. É um serviço gerenciado fornecido pelo Azure e, portanto, é completamente gerenciado pelo Azure. Os clientes precisam apenas apresentar seus dados e armazená-los em um Data Lake.

Há duas versões do Azure Data Lake Storage: versão 1 (Gen1) e a versão atual, versão 2 (Gen2). A Gen2 tem todas as funcionalidades da Gen1, mas com a diferença de que é baseada no Armazenamento de Blob do Azure.

Como o Armazenamento de Blob do Azure está altamente disponível, pode ser replicado várias vezes, está preparado para desastre e tem baixo custo, esses benefícios são transferidos para o Gen2 Data Lake. O Data Lake pode armazenar qualquer tipo de dados, incluindo dados relacionais, não relacionais, baseados em sistema de arquivos e hierárquicos.

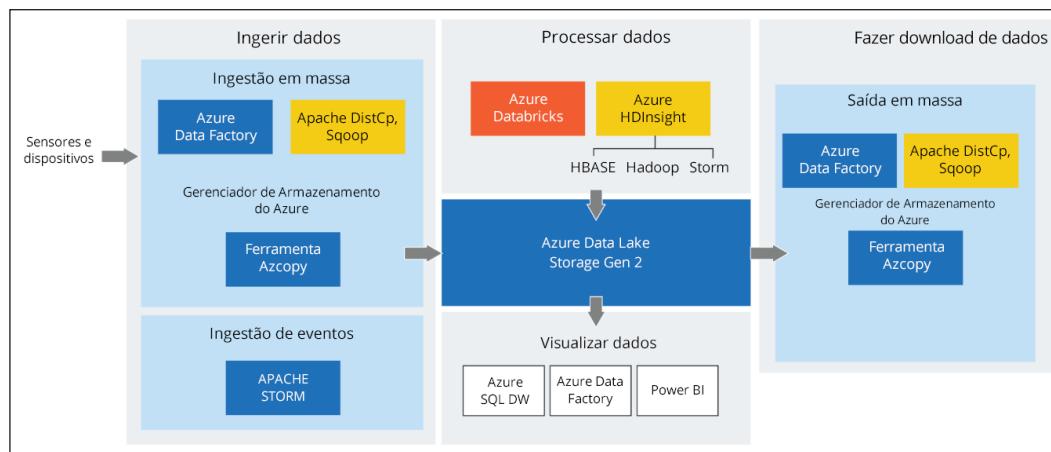
Criar uma instância do Data Lake Gen 2 é tão simples quanto criar uma nova conta de Armazenamento. A única alteração que precisa ser feita é habilitar o namespace hierárquico na guia Avançado da sua conta de Armazenamento.

Entendendo o processamento de big data

Há quatro fases distintas no processamento de big data, como descrito a seguir:

- A ingestão de big data
- O processamento de big data
- O armazenamento dos dados resultantes em um armazenamento de dados
- A apresentação de insights e dados

Essas quatro fases estão resumidas no diagrama a seguir. Estaremos analisando cada uma dessas fases nas próximas seções:



Ingestão de dados

A fase de ingestão de dados traz dados para o Data Lake. Esses dados podem ser transmitidos em tempo real usando ferramentas, como o Azure HDInsight Storm.

Os dados na infraestrutura local também podem ser migrados para o Data Lake.

Processamento de dados

Os dados no Data Lake podem ser processados e transformados usando ferramentas, como o Azure Databricks e o HDInsight, e qualquer saída que normalmente é exigida pelos aplicativos pode ser gerada.

Armazenamento de dados

Nesta fase, os dados transformados são movidos para armazenamentos de dados, como data warehouses, Cosmos DB e Azure SQL, de forma que possam ser consumidos diretamente pelos aplicativos.

Apresentação de dados

Os insights e a sabedoria de dados gerados podem ser usados para visualização, painéis em tempo real, relatórios e notificações.

Migração de dados do Armazenamento do Azure para o Data Lake Gen2 Storage

Nesta seção, migraremos dados do Armazenamento de Blob do Azure para outro contêiner do Azure da mesma instância do Armazenamento de Blob do Azure e também migraremos dados para uma instância do Azure Gen2 Data Lake usando um pipeline do Azure Data Factory. As seções a seguir detalham as etapas necessárias para criar essa solução completa.

Preparação da conta de armazenamento de origem

Para podermos criar pipelines do Azure Data Factory e usá-los para migração, precisamos criar uma nova conta de Armazenamento, que consiste em vários contêineres, e carregar os arquivos de dados. No mundo real, esses arquivos e a conexão de armazenamento já estariam preparados. A primeira etapa para criar uma nova conta de Armazenamento do Azure é criar um novo grupo de recursos ou escolher um grupo de recursos existente em uma Assinatura do Azure.

Provisionamento de um novo grupo de recursos

Cada recurso no Azure está associado a um grupo de recursos. Antes de provisionar uma conta do Armazenamento do Azure, devemos criar um grupo de recursos que hospedará a conta de Armazenamento. As etapas para a criação de um grupo de recursos são mencionadas aqui. Observe que um novo grupo de recursos pode estar sendo criado durante o provisionamento de uma conta de Armazenamento do Azure ou um grupo de recursos existente pode ser usado:

1. Navegue até o portal do Azure, faça login e clique em **+ Criar um recurso**; em seguida, procure Grupo de recursos.
2. Selecione **Grupo de recursos** nos resultados da pesquisa e crie um novo grupo de recursos. Forneça um nome e escolha um local apropriado. Observe que todos os recursos devem ser hospedados no mesmo grupo de recursos e local para que seja fácil excluí-los.

Provisionamento de uma conta de Armazenamento

Nesta seção, percorreremos as etapas de criação de uma nova conta de Armazenamento do Azure. Essa Conta de Armazenamento será a fonte dos dados de origem da qual os dados serão migrados.

1. Clique em **+ Criar um recurso** e procure Conta de Armazenamento. Selecione **Conta de Armazenamento** nos resultados da pesquisa e crie uma nova conta de armazenamento.
2. Forneça um nome e local e, em seguida, selecione uma assinatura com base no grupo de recursos que foi criado anteriormente.
3. Selecione **StorageV2 (V2 de uso geral)** para **Tipo conta, Padrão para Performance e Armazenamento com redundância local (LRS)** para **Replicação**, conforme demonstrado na captura de tela a seguir:

Create storage account

Azure Storage is a Microsoft-managed service providing cloud storage that is highly available, secure, durable, scalable, and redundant. Azure Storage includes Azure Blobs (objects), Azure Data Lake Storage Gen2, Azure Files, Azure Queues, and Azure Tables. The cost of your storage account depends on the usage and the options you choose below. [Learn more](#)

PROJECT DETAILS

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

* Subscription	RiteshSubscription
└─ * Resource group	BigDataSolutions
	Create new

INSTANCE DETAILS

The default deployment model is Resource Manager, which supports the latest Azure features. You may choose to deploy using the classic deployment model instead. [Choose classic deployment model](#)

* Storage account name	adfsamplesourcedata
* Location	East US
Performance	<input checked="" type="radio"/> Standard <input type="radio"/> Premium
Account kind	StorageV2 (general purpose v2)
Replication	Locally-redundant storage (LRS)
Access tier (default)	<input type="radio"/> Cool <input checked="" type="radio"/> Hot

4. Agora, crie alguns contêineres na conta de armazenamento. **rawdata** contém os arquivos que serão extraídos pelo pipeline do Data Factory e atuará como o conjunto de dados de origem, enquanto **finaldata** conterá arquivos para os quais os pipelines do Data Factory gravarão dados e atuará como o conjunto de dados de destino:

The screenshot shows the Azure Storage Blobs interface for the 'adfsamplesourcedata' storage account. On the left, there's a sidebar with links: Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, and Events. The main area displays a list of containers under the heading 'NAME'. Two containers are listed: 'finaldata' and 'rawdata'. The 'finaldata' container is highlighted with a dashed blue border around its row.

5. Carregue um arquivo de dados (esse arquivo está disponível com o código-fonte) para o contêiner rawdata, como mostrado a seguir:

The screenshot shows the Azure Storage Blobs upload interface for the 'rawdata' container. At the top, there are buttons for Upload, Refresh, Delete, and Acquire lease. Below that, it says 'Location: rawdata'. There's a search bar labeled 'Search blobs by prefix (case-sensitive)'. The main area is titled 'NAME' and lists a single file: 'products.csv'.

Criação de um novo serviço do Data Lake Gen2

Como já sabemos, o serviço Data Lake Gen3 é baseado na conta de Armazenamento do Azure. Por esse motivo, criaremos uma nova conta de Armazenamento da mesma forma que fizemos anteriormente - com a única diferença sendo a seleção de **Habilitado para Namespace hierárquico** na guia **Avançado** da nova conta de Armazenamento do Azure. Isso criará o novo serviço Data Lake Gen2:

Create storage account

Basics **Advanced** Tags Review + create

SECURITY

Secure transfer required i Disabled Enabled

VIRTUAL NETWORKS

Allow access from All networks Selected network
i All networks will be able to access this storage account. [Learn more](#)

DATA LAKE STORAGE GEN2 (PREVIEW)

Hierarchical namespace i Disabled Enabled

Provisionamento de pipeline do Azure Data Factory

Agora que já provisionamos o grupo de recursos e a Conta de Armazenamento do Azure, é hora de criar um novo Pipeline do Data Factory:

1. Crie um novo pipeline do Data Factory selecionando V2 e fornecendo um nome e local junto com uma seleção de grupo de recursos e assinatura, conforme mostrado na captura de tela a seguir:

New data factory

* Name i
 ✓

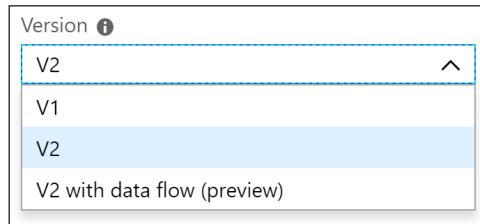
* Subscription
 ▼

* Resource Group i
 Create new Use existing
 ✓

Version i
 ▼

* Location i
 ▼

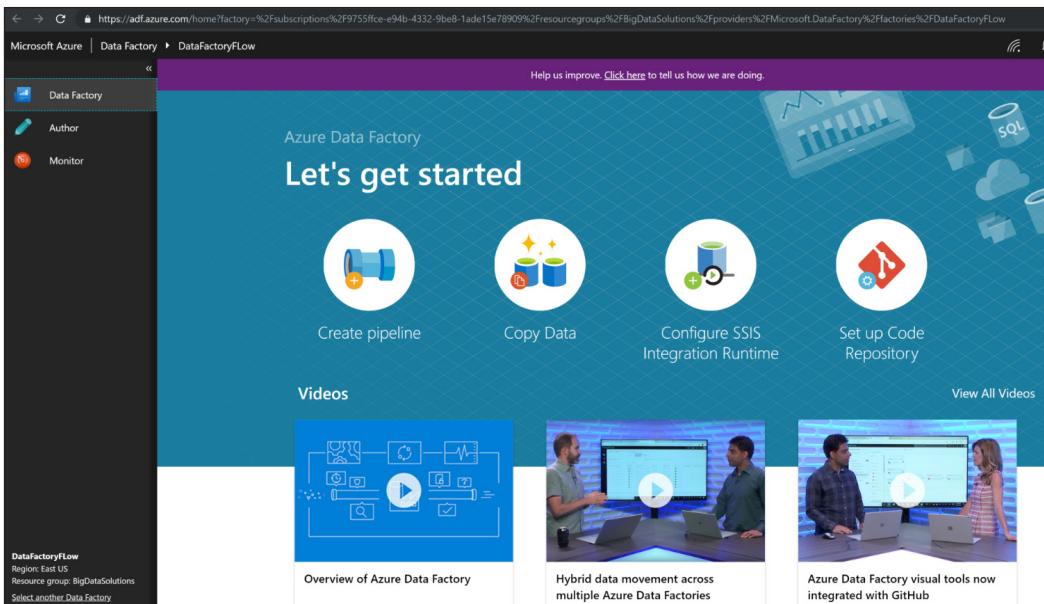
2. O Data Factory tem três versões diferentes, conforme mostrado na captura de tela a seguir. Já discutimos **V1** e **V2**. A **V2 com fluxo de dados** está em versão prévia; portanto, você precisará obter sua assinatura na lista de permissões antes de usá-la:



3. Depois que o recurso Data Factory for criado, clique no link **Autor e Monitor** na página central:

A screenshot of the Azure portal showing the properties of a Data Factory resource. The resource group is "BigDataSolutions", status is "Succeeded", location is "East US", subscription is "RiteshSubscription", and ID is "9755ffce-e94b-4332-9be8-1ade15e78909". Below the properties, there are two links: "Documentation" and "Author & Monitor". The "Author & Monitor" link is highlighted with a red box.

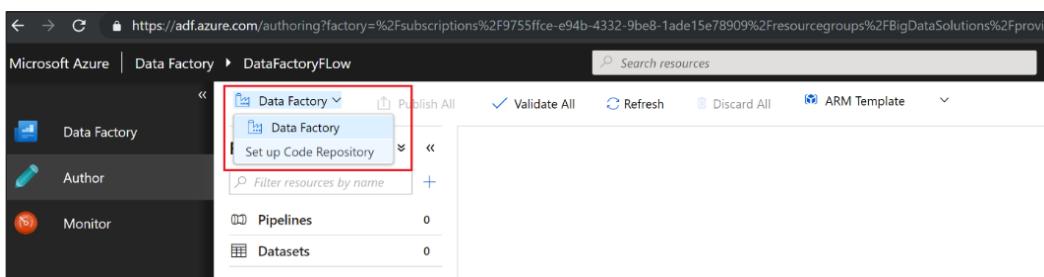
4. Com isso, outra janela será aberta, exibindo o designer do Data Factory para os pipelines:



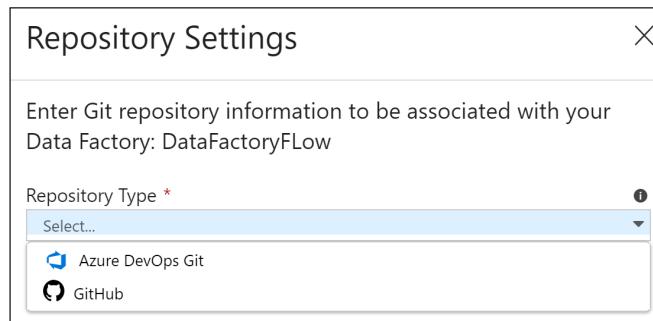
Configurações do repositório

Antes de criar qualquer artefato do Data Factory, como conjuntos de dados e pipelines, é recomendável configurar o repositório de código para hospedar arquivos relacionados ao Data Factory:

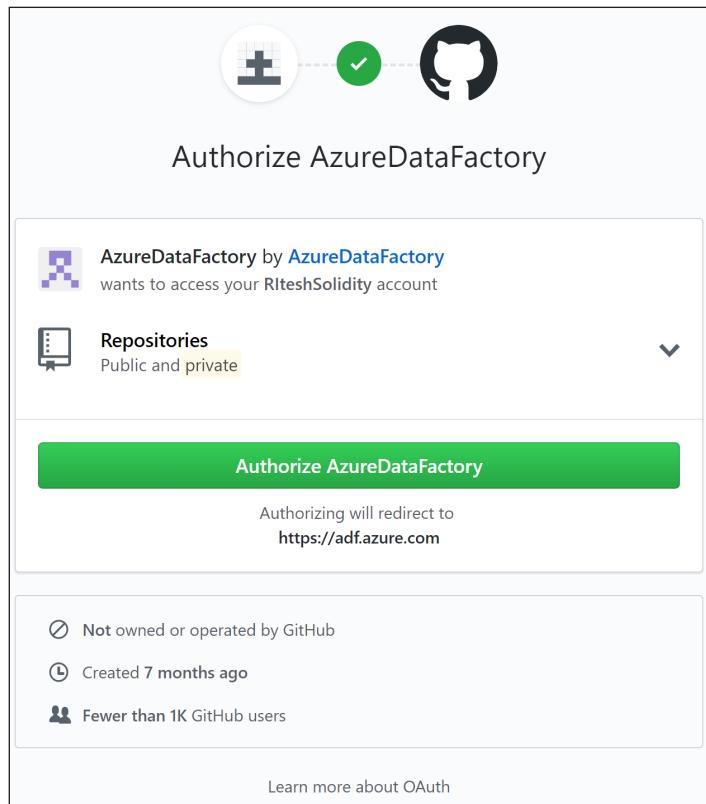
1. Clique na lista suspensa Data Factory no menu superior:



2. Na folha resultante, selecione qualquer um dos repositórios em que você deseja armazenar os arquivos de código do Data Factory. Neste caso, vamos selecionar o **GitHub**, conforme demonstrado na captura de tela a seguir:



3. Será solicitado que você forneça autorização para a conta do GitHub.
4. Faça login no GitHub com suas credenciais e forneça permissões para o serviço do Azure Data Factory:



5. Crie ou reutilize um repositório existente do GitHub. No nosso caso, estamos criando um novo repositório chamado ADF. Se você estiver criando um novo repositório, assegure-se de que ele já esteja inicializado, caso contrário, a configuração do repositório Data Factory reclamará:

Create a new repository
A repository contains all the files for your project, including the revision history.

Owner Repository name *

 RiteshSolidity / ADF ✓

Great repository names are short and memorable. Need inspiration? How about [verbose-winner](#).

Description (optional)
Repository for ADF demo

 **Public**
Anyone can see this repository. You choose who can commit.

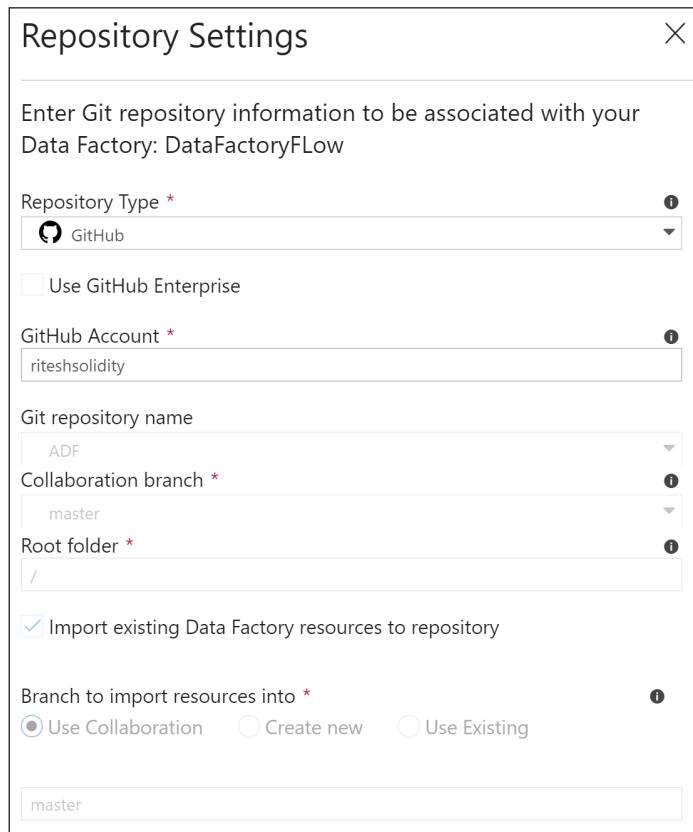
 **Private**
You choose who can see and commit to this repository.

Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

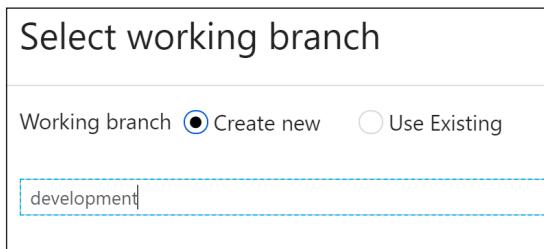
Add .gitignore: None ▾ | Add a license: None ▾ ⓘ

Create repository

6. Agora, podemos voltar para a janela de pipeline do Data Factory e garantir que as configurações do repositório estejam corretas, incluindo o **Nome do repositório Git**, a **Ramificação de colaboração**, a **Pasta raiz** e a **Ramificação**, que atuará como a ramificação principal:



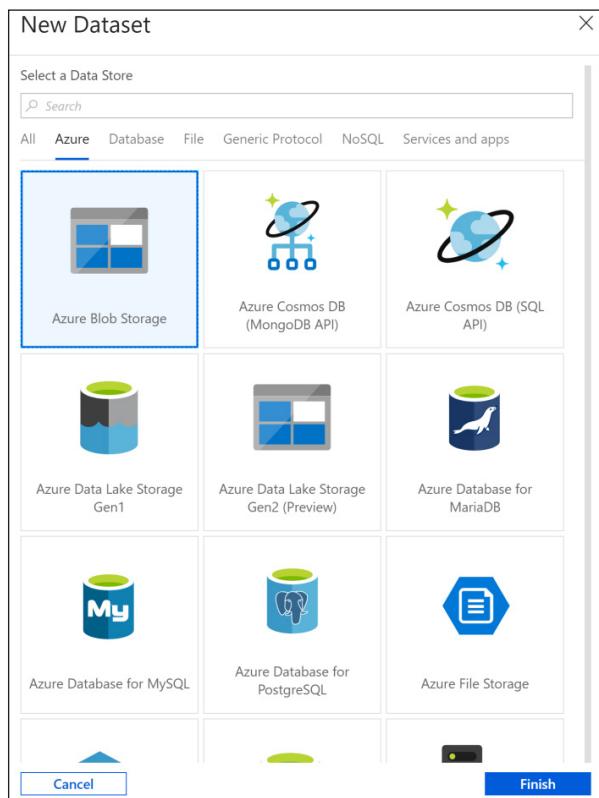
7. Na próxima tela, crie uma nova ramificação na qual os desenvolvedores estarão trabalhando. No nosso caso, uma nova ramificação de desenvolvimento é criada, conforme demonstrado na seguinte captura de tela:



Criação do primeiro conjunto de dados

Agora podemos voltar ao pipeline do Data Factory. Primeiro, crie um novo conjunto de dados que atuará como o conjunto de dados de origem. Será a primeira conta de armazenamento para a qual criamos e carregamos o arquivo de amostra `products.csv`:

1. Clique em **+ Conjunto de Dados** no menu superior e selecione **Armazenamento de Blob do Azure**:



2. Em seguida, no painel inferior resultante, na guia **Geral**, forneça um nome para o conjunto de dados:

General	Connection ¹	Schema	Parameters
Name *	InputBlobStorageData		
Description			
Annotations	+ New		

3. Na guia **Conexão**, crie um novo serviço vinculado de acordo com a configuração da seguinte captura de tela:

New Linked Service X

Name *
AzureBlobStorage1

Description

Connect via integration runtime *
AutoResolveIntegrationRuntime

Authentication method
 Use account key

Account selection method
 From Azure subscription Enter manually

Azure subscription
RiteshSubscription (9755ffce-e94b-4332-9be8-1ade15e78909)

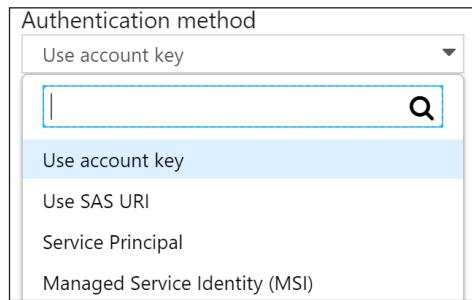
Storage account name *
adfsamplesourcedata

Additional connection properties
+ New

Annotations
+ New

► Advanced i

4. Os serviços vinculados fornecem vários métodos de autenticação, e nós estamos usando o método **uniform resource locator (URL) de assinatura de acesso compartilhado (SAS)**. Também é possível usar uma chave de conta, uma entidade de serviço e uma identidade gerenciada como métodos de autenticação:



5. A guia **Conexão** após a configuração deve ser semelhante à captura de tela a seguir. Observe que o caminho inclui o nome do contêiner e o nome do arquivo:

General	Connection	Schema	Parameters
	Linked service * <input type="button" value="AzureBlobStorage1"/> <input type="button" value="Test connection"/> <input type="button" value="Edit"/> <input type="button" value="New"/> File path <input type="text" value="rawdata"/> / <input type="text" value="products.csv"/> <input type="button" value="Browse"/> <input type="button" value="Preview data"/> Compression Type <input type="button" value="None"/> Filter by last modified <input type="text"/> <input type="text"/> Binary Copy <input type="checkbox"/> File Format Settings File format <input type="button" value="Text format"/> <input type="button" value="Detect Text Format"/> Column delimiter <input type="button" value="Comma (,)"/> <input type="checkbox" value="Use custom delimiter"/> Row delimiter <input type="button" value="Carriage Return + Line feed (\r\n)"/> <input type="checkbox" value="Use custom delimiter"/>		

6. Neste ponto, se você clicar no botão **Dados de visualização**, ele mostrará os dados de visualização do arquivo products.csv. Na guia Esquema, adicione duas colunas e nomeie-as como ProductID e ProductPrice. O esquema ajuda a fornecer um identificador para as colunas e também a mapear as colunas de origem no conjunto de dados de origem para as colunas de destino no conjunto de dados de destino quando os nomes não são os mesmos:

The screenshot shows the Azure Data Factory schema editor interface. On the left, there's a navigation bar with 'Save', 'General', 'Connection', 'Schema' (which is selected), and 'Parameters'. Below this is a toolbar with 'Import Schema', 'Preview data' (which is highlighted in blue), and 'New column'. A table below lists columns: 'ProductID' (String type) and 'ProductPrice' (Decimal type). Both columns have a checked checkbox next to them. To the right, a 'Data Preview' window is open, showing a table with 10 rows of data. The columns in the preview are 'ProductID' and 'ProductPrice'. The data values are as follows:

ProductID	ProductPrice
90269	9.989999771118164
101285	9.989999771118164
102676	9.989999771118164
84745	9.989999771118164
119920	9.989999771118164
123586	9.989999771118164
156029	15.989999771118164
130727	13.989999771118164
130727	9.989999771118164

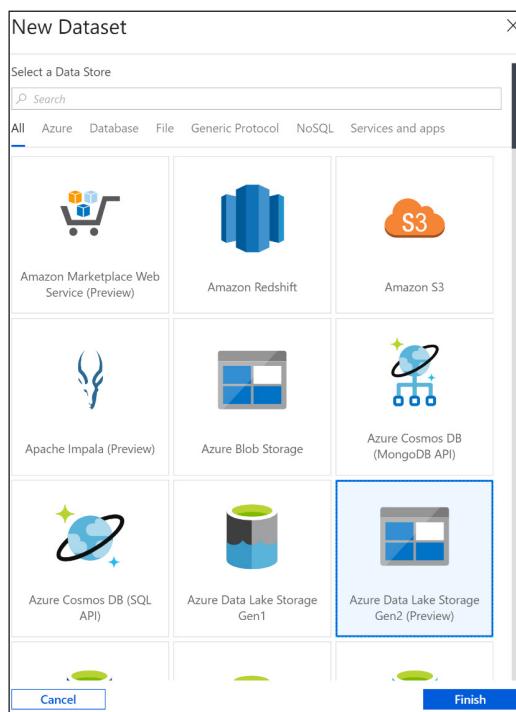
Criação do segundo conjunto de dados

Crie um novo conjunto de dados e serviço vinculado para a conta de Armazenamento de Blob de destino da mesma maneira que você fez antes. Observe que a conta de Armazenamento é a mesma que a origem, mas o contêiner é diferente. Certifique-se de que os dados recebidos tenham informações de esquema associadas a eles, como mostra a captura de tela a seguir:

The screenshot shows the 'Connection' tab of the Azure Data Factory dataset configuration. It includes fields for 'Linked service' (set to 'AzureBlobStorage1'), 'File path' ('finaldata/processedproducts.csv'), 'Compression Type' ('None'), and various filtering and delimiter settings. Buttons for 'Test connection', 'Edit', 'New', 'Browse', and 'Preview data' are visible.

Criação de um terceiro conjunto de dados

Crie um novo conjunto de dados para a instância de armazenamento do Data Lake Gen2 como o conjunto de dados de destino. Para fazer isso, selecione o novo conjunto de dados e selecione **Azure Data Lake Storage Gen2 (Versão Prévia)**, conforme demonstrado na seguinte captura de tela:



Dê um nome ao novo conjunto de dados e crie um novo serviço vinculado na guia **Conexão**. Escolha **Usar chave de conta** como o método de autenticação e o restante da configuração será preenchida automaticamente depois de selecionar o nome da Conta de armazenamento. Em seguida, teste a conexão clicando no botão **Testar conexão**. Mantenha a configuração padrão para o restante das guias, conforme mostrado na captura de tela a seguir:

New Linked Service (Azure Data Lake Storage Gen2 ... ×)

Name *
AzureDataLakeStorage1

Description

Connect via integration runtime *
AutoResolveIntegrationRuntime

Authentication method
Use account key

Account selection method
 From Azure subscription Enter manually

Azure subscription
Select all

Storage account name *
datalakegen2productstore

[Sign up to the public preview of Azure Data Lake Storage Gen2.](#)

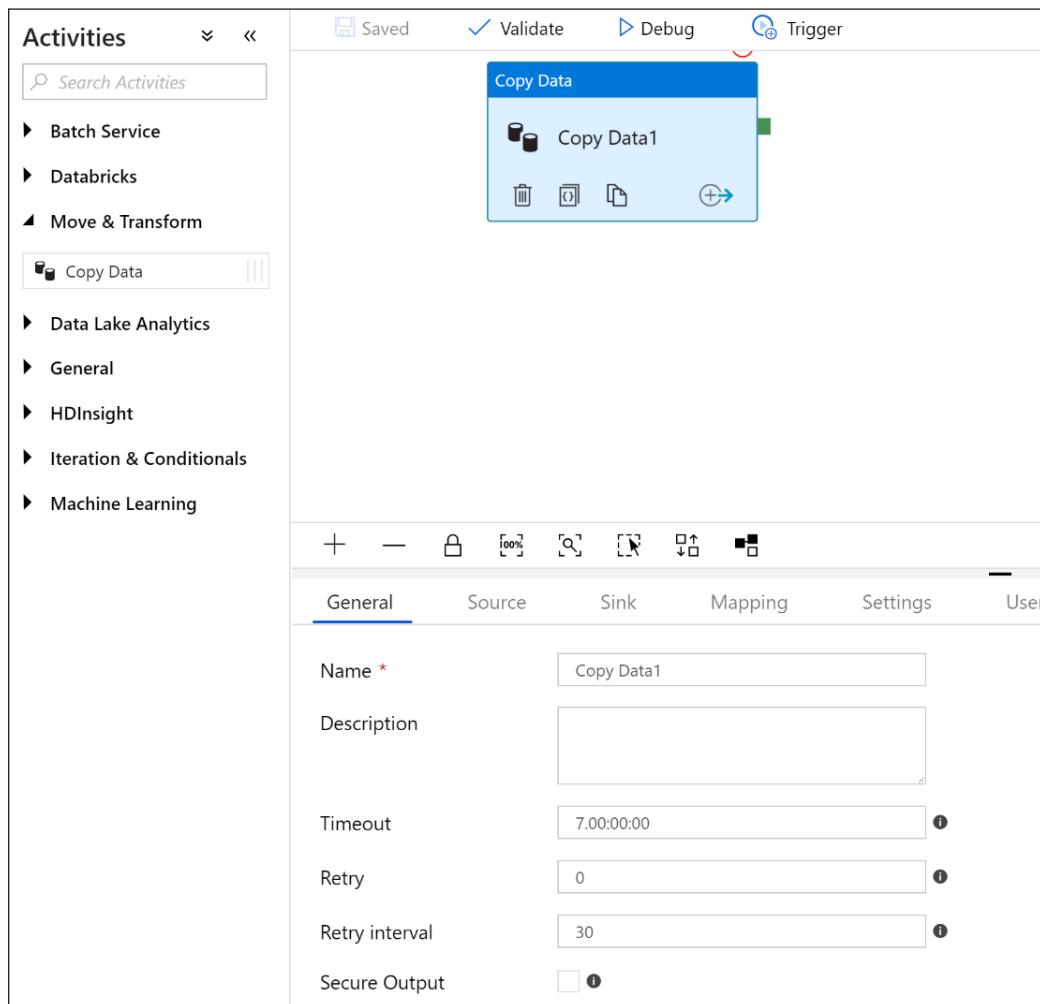
Annotations
+ New
► Advanced ⓘ

[Cancel](#) [Test connection](#) [Finish](#)

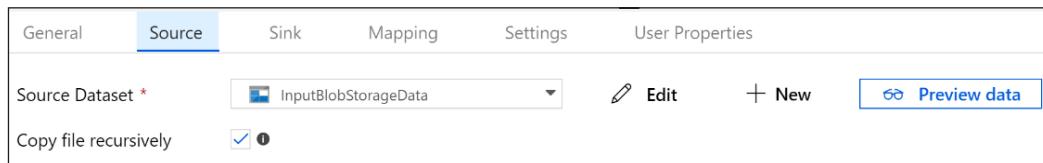
Criação de um pipeline

Depois que todos os conjuntos de dados são criados, podemos criar um pipeline que consumirá esses conjuntos de dados. As etapas para criar um pipeline são mencionadas a seguir:

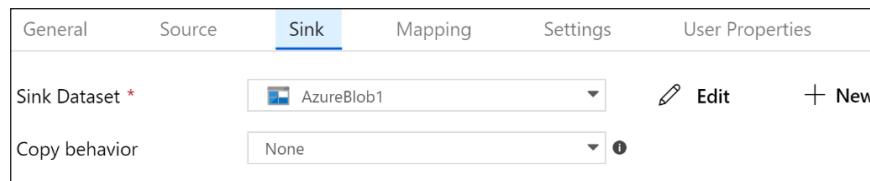
1. Clique no menu **+ Pipeline** do menu superior para criar um novo pipeline. Em seguida, arraste e solte a atividade **Copiar Dados** do menu **Mover e Transformar**, conforme demonstrado na captura de tela a seguir:



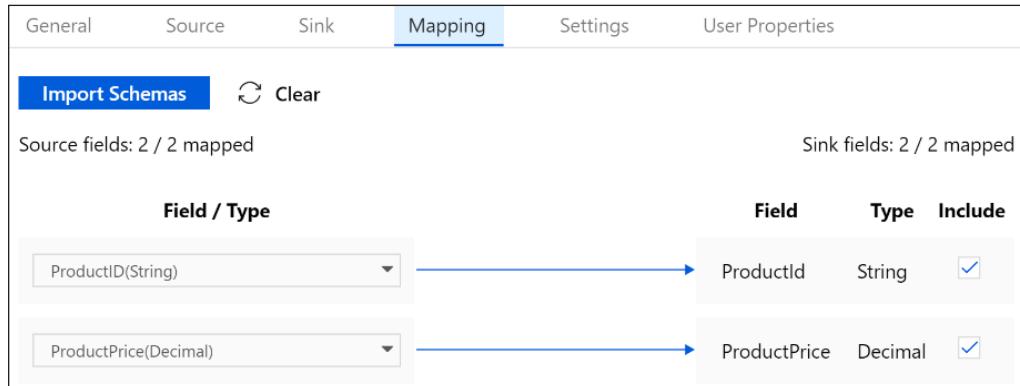
2. A guia **Geral** resultante pode ser deixada como está, mas a guia **Origem** deve ser configurada para usar o conjunto de dados de origem que configuramos anteriormente:



3. A guia **Coletor** é usada para configurar o armazenamento de dados de destino e o conjunto de dados e deve ser configurada para usar o conjunto de dados de destino que configuramos anteriormente:



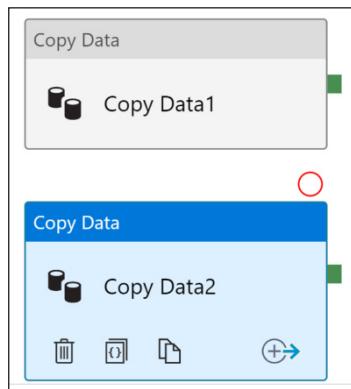
4. Na guia **Mapeamento**, mapeie as colunas da origem para as colunas do conjunto de dados de destino, conforme mostrado na captura de tela a seguir:



Adicionar mais uma atividade de cópia de dados

No Pipeline, podemos adicionar várias atividades, cada uma responsável por uma tarefa de transformação específica. A tarefa mencionada nesta seção é responsável por copiar dados da Conta de Armazenamento do Azure para o Azure Data Lake Storage:

1. Adicione outra atividade **Copiar Dados** no menu de atividades à esquerda para migrar dados para o Data Lake Storage. As duas tarefas de cópia serão executadas em paralelo:

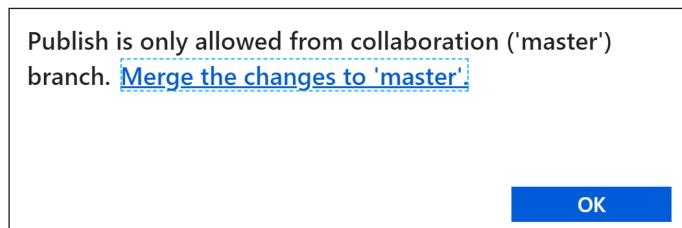


2. A configuração para a origem é a conta de Armazenamento de Blob do Azure que contém arquivo products.csv.
3. A configuração do coletor escolherá a conta de armazenamento do Data Lake Gen2.
4. O restante da configuração pode ser deixado nas respectivas configurações padrão para a segunda atividade de cópia de dados.

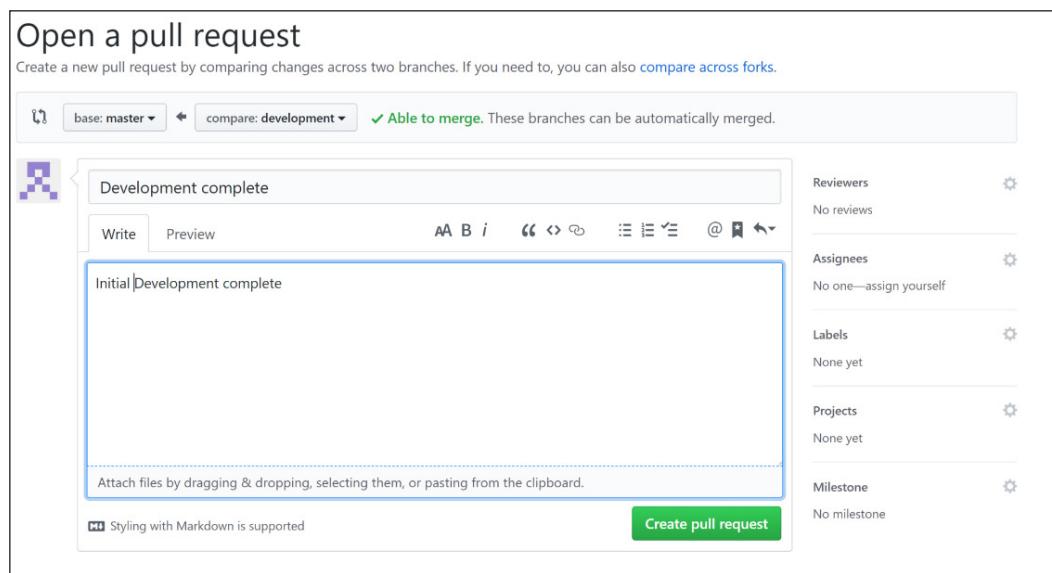
Publicação

Depois de criar e configurar o pipeline, é importante que eles sejam publicados de forma que possam ser consumidos e agendados para uso da produção. As etapas para publicar um pipeline são mencionadas aqui:

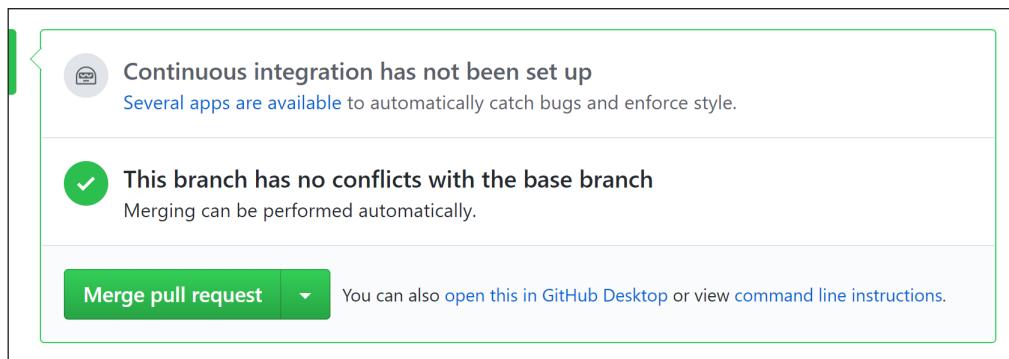
1. Podemos executar o pipeline já configurado no modo de depuração. Após a depuração, podemos publicar o pipeline. No entanto, para que possamos publicar o pipeline, precisamos mesclar o código da ramificação de desenvolvimento para a ramificação mestre e, opcionalmente, excluir a ramificação de desenvolvimento:



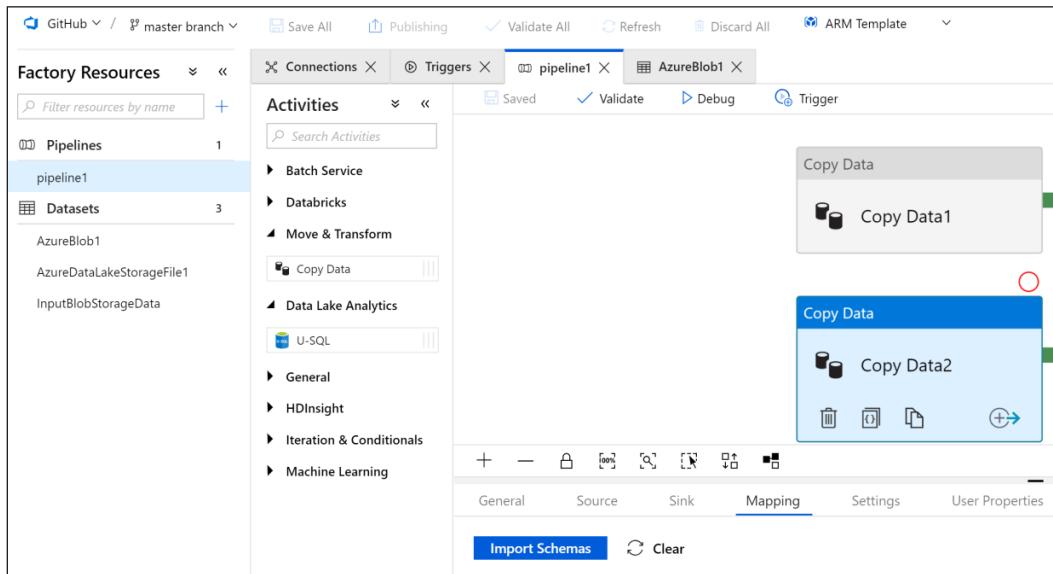
- Vá para a conta do GitHub e crie uma nova solicitação pull para mesclar o código da ramificação de desenvolvimento para a ramificação mestre. Isso ocorre porque o código inteiro para os pipelines, conjuntos de dados, serviços vinculados e modelos do **Azure Resource Manager (ARM)** está na ramificação de desenvolvimento:



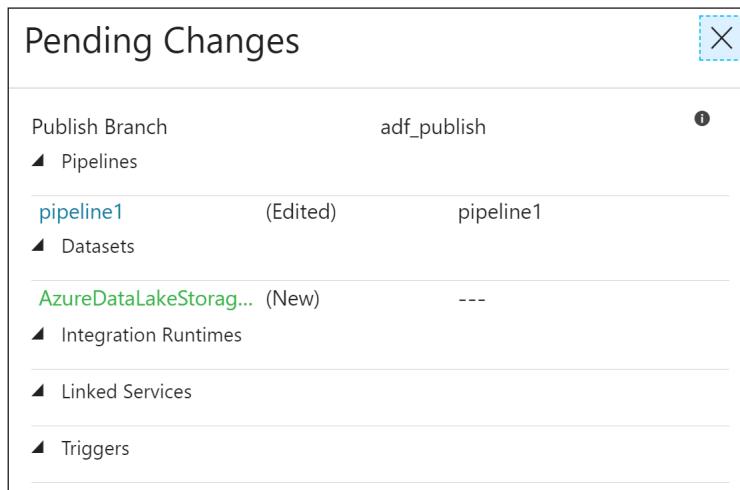
- Depois que a solicitação pull for gerada, aceite-a e mescle o código da ramificação de desenvolvimento para a ramificação mestre:



4. De volta ao designer do Data Factory, selecione **ramificação mestre** no menu superior e publique o código:



5. Clique na folha resultante, como demonstrado na seguinte captura de tela:



Soluções de Big Data do Azure com Azure Data Lake Storage e Data Factory

6. O repositório GitHub deve parecer com a seguinte captura de tela:

The screenshot shows a GitHub repository page for 'RiteshSolidity / ADF'. The repository name is 'Repository for ADF demo'. It has 6 commits, 3 branches, 0 releases, and 1 contributor. The most recent commit is 'adf_publish' (less than a minute ago). Below the commit list, there are buttons for 'Create new file', 'Upload files', 'Find file', and 'Clone or download'. The commit list includes:

- RiteshSolidity Merge pull request #1 from RiteshSolidity/development ... (Latest commit 220155e 3 minutes ago)
- dataset Adding dataset: AzureBlob1 (9 minutes ago)
- linkedService Adding linkedService: AzureBlobStorage1 (16 minutes ago)
- pipeline Adding pipeline: pipeline1 (7 minutes ago)
- README.md commit (an hour ago)

7. Acione o pipeline clicando no botão **Disparar Agora** no menu superior quando a publicação estiver concluída:

The screenshot shows the Azure Data Factory pipeline editor. On the left, the 'Factory Resources' sidebar lists 'Pipelines' (1) and 'Datasets' (3). The 'pipelinel' pipeline is selected. In the main area, the 'Activities' pane shows a 'Copy Data' activity under 'Move & Transform'. A context menu is open over this activity, with the 'Trigger Now' option highlighted. The pipeline interface includes tabs for 'General', 'Source', 'Sink', 'Mapping', 'Settings', and 'User Properties'. There are also buttons for 'Import Schemas' and 'Clear'.

Disparar um data factory forçará o código para o repositório, gerará os modelos do ARM e executará o pipeline.

Resultado final

Se navegarmos para a conta de Armazenamento que contém os contêineres de origem e destino, poderemos ver o nosso novo arquivo `processedproducts.csv` da seguinte forma:

The screenshot shows the Azure Storage Explorer interface. On the left, there's a sidebar with 'finaldata' selected as the container. Under 'Settings', 'Access policy', 'Properties', and 'Metadata' are listed. The main area shows a table with one row:

NAME	MODIFIED	ACCESS TIER	BLOB TYPE	SIZE	LEASE STATE
<code>processedproducts.csv</code>	1/27/2019, 5:47:22 AM	Hot (Inferred)	Block blob	9.12 MB	Available

Não podemos ver os arquivos do Data Lake Storage; portanto, temos que fazer download da ferramenta Gerenciador de Armazenamento para visualizar os arquivos na conta do Data Lake Storage:

The screenshot shows the Azure Storage Explorer interface. On the left, the 'EXPLORER' pane shows a tree view with 'datalakegen2productstore (ADLS Gen2)' expanded, showing 'Blob Containers', 'products' (selected), 'File Shares', 'Queues', and 'Tables'. The main pane shows the contents of the 'products' container. The toolbar includes 'Upload', 'Download', 'New Folder', 'Select All', 'Rename', 'Manage Access', 'Properties', 'Delete', and 'Refresh'. A search bar at the top of the main pane contains the text 'products'. Below it is a table with one row:

Name	Last Modified	Content Type	Size
<code>information.csv</code>	1/27/2019, 2:20:37 AM		9.1 MB

A ferramenta Gerenciador de Armazenamento não deve ser anterior à versão 1.6.2 e o download pode ser feito em <https://aka.ms/portalfx/downloadstorageexplorer>.

Resumo

Este foi outro capítulo sobre como lidar com big data. Este capítulo tratou do serviço Azure Data Factory, que é responsável por fornecer serviços ETL no Azure. Como é uma **Plataforma como Serviço (PaaS)**, ela fornece escalabilidade ilimitada, alta disponibilidade e pipelines fáceis de configurar. Sua integração com o Azure DevOps e o GitHub também é perfeita. Também exploramos os recursos e benefícios do uso do armazenamento do Azure Data Lake Gen2 para armazenar qualquer tipo de big data. É um armazenamento de dados hierárquico econômico e altamente escalonável para lidar com big data e é compatível com o Azure HDInsight, Databricks e o ecossistema Hadoop.

O próximo capítulo terá como foco a criação de soluções baseadas em eventos. O Azure nos fornece recursos que ajudam na geração e no consumo de eventos. O próximo capítulo tratará em mais detalhes os recursos, incluindo os Hubs de Eventos e o Stream Analytics.

14

Azure Stream Analytics e Hubs de Eventos

Os eventos estão em toda parte! Qualquer atividade ou tarefa que altera o estado atual gera um evento. Os eventos não eram tão populares há alguns anos; não havia nenhuma nuvem, e não tinha muita tração para a **Internet das Coisas (IoT)** e dispositivos ou sensores. Durante esse período, as organizações usavam ambientes hospedados de **provedores de serviços de internet (ISPs)** que apenas tinham sistemas de monitoramento em cima deles. Esses sistemas de monitoramento geraram eventos que eram poucos e distantes.

No entanto, com o surgimento da nuvem, as coisas estão mudando rapidamente. Com o aumento das implantações na nuvem, especialmente dos serviços **Plataforma como um Serviço (PaaS)**, as organizações não precisam mais de muito controle sobre o hardware e a plataforma, e agora, toda vez que ocorre uma alteração em um ambiente, é gerado um evento. Com o surgimento de eventos na nuvem, a IoT ganhou muita proeminência e os eventos começaram a roubar os holofotes.

Outro fenômeno durante esse período foi a rápida explosão de crescimento na disponibilidade de dados. A velocidade, a variedade e o volume de dados aumentaram, assim como a necessidade de soluções para armazenamento e processamento de dados. Múltiplas soluções e plataformas surgiram, como o Hadoop, data lakes para armazenamento, data lakes para análise e serviços de aprendizado de máquina para análise.

Além do armazenamento e da análise, há também a necessidade de serviços capazes de ingerir milhões e milhões de eventos e mensagens de várias fontes. Há também a necessidade de serviços que possam trabalhar em dados temporais, em vez de trabalhar em todo o instantâneo de dados.

Neste capítulo, percorreremos alguns serviços pertinentes no Azure, da seguinte forma:

- Hubs de Eventos do Azure
- Azure Stream Analytics

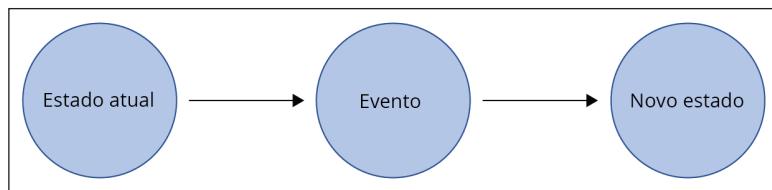
Introdução a eventos

Eventos são componentes importantes na arquitetura do Azure e do Aplicativo Azure. Os eventos estão em toda parte dentro do ecossistema de software. Geralmente, qualquer ação realizada resulta em um evento que pode ser retido e outras ações podem ser tomadas. Para levar adiante esta discussão, é importante primeiro entender os fundamentos dos eventos.

Eventos

Os eventos ajudam a alterar o estado atual para o estado de destino. Os eventos são diferentes das mensagens. As mensagens estão relacionadas à funcionalidade do negócio, como enviar detalhes do pedido para outro sistema. Em comparação, os eventos são diferentes; por exemplo, uma máquina virtual que está sendo interrompida é um evento.

Um evento, junto com o estado atual, ajuda na transição do estado atual para o estado de destino, e essa transição é demonstrada no diagrama a seguir:



Streaming de eventos

Se houver apenas alguns eventos, não haverá muita necessidade de streaming de evento, pois ele fornece valor real com processos de big data, em que os dados vêm em grandes volumes e em muitos formatos diferentes.

O streaming de eventos refere-se a serviços que podem aceitar dados como e quando surgem, em vez de aceitá-los periodicamente. Por exemplo, os fluxos de eventos devem ser capazes de aceitar informações de temperatura de dispositivos como e quando enviá-las, em vez de fazer com que os dados esperem em uma fila ou um ambiente de preparo.

O streaming de eventos também tem a capacidade de consultar dados em trânsito. Estes são dados temporais que são armazenados por um tempo, e as consultas ocorrem nos dados em movimento; portanto, os dados não são estacionários. Esse recurso não está disponível em outras plataformas de dados, que podem consultar apenas dados armazenados e não dados temporais que acabaram de ser ingeridos.

Os serviços de streaming de eventos devem ser capazes de escalar facilmente para aceitar milhões ou até bilhões de eventos. Eles devem estar altamente disponíveis para que as fontes possam enviar eventos e dados para eles a qualquer momento. A ingestão de dados em tempo real e a capacidade de trabalhar com esses dados, em vez de dados armazenados em um local diferente, é a chave para o streaming de eventos.

Mas quando já temos tantas plataformas de dados com recursos avançados de execução de consulta, por que precisamos de streaming de eventos? Deixe-me fornecer alguns cenários em que trabalhar com dados recebidos é muito importante. Esses cenários não podem ser resolvidos de forma eficaz e eficiente por plataformas de dados existentes:

- **Detecção de fraude de cartão de crédito:** isso deve acontecer quando uma transação fraudulenta estiver acontecendo.
- **Informações de telemetria dos sensores:** no caso de dispositivos de IoT que enviam informações vitais sobre seus ambientes, o usuário deve ser notificado como e quando uma anomalia for detectada.
- **Painéis ativos:** o streaming de eventos é necessário para criar painéis que mostram informações ao vivo.
- **Telemetria do ambiente do data center:** permitirá que o usuário saiba sobre invasões, violações de segurança, falhas de componentes, e muito mais.

Há muitas possibilidades de aplicação de streaming de eventos dentro de uma empresa, e sua importância não pode ser enfatizada o suficiente.

Hubs de Eventos

Hubs de Eventos é o serviço no Azure que fornece recursos relacionados à ingestão e ao armazenamento de eventos necessários para o streaming.

Ele pode ingerir dados de uma variedade de fontes; essas fontes podem ser sensores de IoT ou qualquer aplicativo que use o **Kit de Desenvolvimento de Software (SDK)** dos Hubs de Eventos para enviar dados. Esse serviço é compatível com vários protocolos para ingerir e receber dados. Esses protocolos são padrão da indústria e incluem o seguinte:

- **HTTP:** uma opção sem estado e não requer uma sessão ativa.
- **Advanced Messaging Queuing Protocol (AMQP):** requer uma sessão ativa (ou seja, uma conexão estabelecida usando soquetes) e funciona com os protocolos **TLS** e **SSL**.
- **Apache Kafka:** uma plataforma de streaming distribuída semelhante ao Stream Analytics.

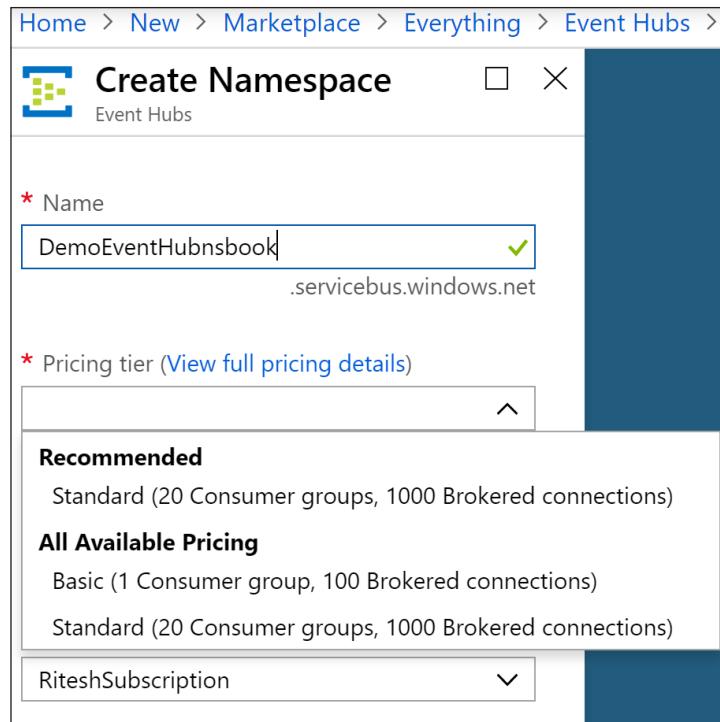
Hubs de Eventos é um serviço de ingestão de eventos. Não tem a capacidade de consultar e enviar resultados da consulta para outro local. Esta é a responsabilidade do Stream Analytics, que é abordado na próxima seção.

Os Hubs de Eventos, sendo uma PaaS no Azure, são altamente distribuídos, disponíveis e escalonáveis.

Os Hubs de Eventos vêm com os dois SKUs ou as camadas de preço a seguir:

- **Básico:** é fornecido com um grupo de consumidores e pode reter mensagens por 1 dia. Ele pode ter um máximo de 100 conexões intermediadas.
- **Padrão:** é fornecido com um máximo de 20 grupos de consumidores e pode reter mensagens por 1 dia com armazenamento adicional por 7 dias. Ele pode ter um máximo de 1.000 conexões intermediadas. Também é possível definir políticas neste SKU.

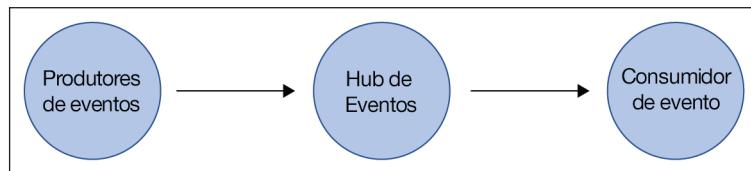
A captura de tela a seguir mostra o formulário para criar um novo namespace de Hubs de Eventos. Ele fornece uma opção para escolher uma camada de preços apropriada, juntamente com outros detalhes importantes:



É importante observar que o SKU não pode ser alterado após o provisionamento de um namespace de Hubs de Eventos. A devida consideração e planejamento devem ser adotados antes de selecionar um SKU. O processo de planejamento deve incluir o planejamento do número de grupos de consumidores necessários e o número de aplicativos interessados em ler eventos do hub.

Arquitetura dos Hubs de Eventos

Há três componentes principais da arquitetura de Hubs de Eventos: **Produtores de Eventos**, **Hub de Eventos** e **Consumidor de Eventos**, como mostra o diagrama a seguir:



Os **Produtores de Eventos** geram eventos e os enviam ao Hub de Eventos. O **Hub de Eventos** armazena os eventos ingeridos e fornece esses dados ao **Consumidor de Eventos**. O **Consumidor de Eventos** é o que está interessado nesses eventos e se conecta ao **Hub de Eventos** para buscar os dados.

Não é possível criar hubs de eventos sem um namespace de Hubs de Eventos. O namespace de Hubs de Eventos funciona como um contêiner e pode hospedar vários hubs de eventos. Cada namespace de Hubs de Eventos fornece um ponto de extremidade exclusivo baseado em REST que é consumido pelos clientes para enviar dados aos Hubs de Eventos. Esse namespace é o mesmo namespace necessário para artefatos do Barramento de Serviço, como tópicos e filas.

A cadeia de conexão de um namespace de Hubs de Eventos é composta pela respectiva URL, nome da política e chave. Uma cadeia de conexão de amostra é mostrada no seguinte bloco de código:

```
Endpoint=sb://demoeventhubnsbook.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=M/E4eeBsr7DA1Xcvw6ziFq1SDNbFX6E49Jfti8CRkbA=
```

Essa cadeia de conexão pode ser encontrada no item de menu **Assinatura de Acesso Compartilhado (SAS)** do namespace. Pode haver várias políticas definidas para um namespace, cada uma com diferentes níveis de acesso ao namespace. Os três níveis de acesso são os seguintes:

- **Gerenciar:** pode gerenciar o hub de eventos de uma perspectiva administrativa. Também tem direitos para enviar e ouvir eventos.
- **Enviar:** pode gravar eventos em Hubs de Eventos.
- **Ouvir:** pode ler eventos de Hubs de Eventos.

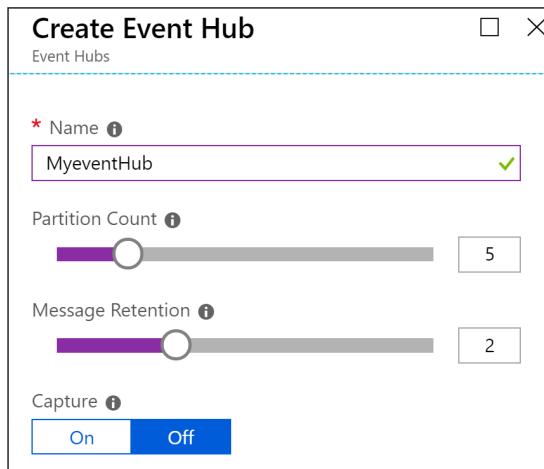
Por padrão, a política `RootManageSharedAccessKey` é criada ao criar um hub de eventos, conforme mostrado na captura de tela a seguir. As políticas ajudam na criação de controle de acesso granular em Hubs de Eventos. A chave associada a cada política é usada pelos consumidores para determinar sua identidade; políticas adicionais também podem ser criadas:

The screenshot shows the Azure portal interface for managing shared access policies. On the left, there's a sidebar with navigation links like Home, DemoEventHubnsbook - Overview, DemoEventHubnsbook - Shared access policies, Overview, Access control (IAM), Tags, Diagnose and solve problems, Events, Settings, Shared access policies (which is selected and highlighted in blue), Geo-Recovery, Firewalls and virtual networks, and Properties. The main content area is titled "DemoEventHubnsbook - Shared access policies". It shows a table with one row for the "RootManageSharedAccessKey" policy. The table has two columns: "POLICY" and "CLAIMS". The "POLICY" column contains "RootManageSharedAccessKey". The "CLAIMS" column contains "Manage, Send, Listen". To the right of the table, there's a "SAS Policy: RootManageSharedAccessKey" panel with options to Save, Discard, Delete, and More. It also lists "Manage", "Send", and "Listen" checkboxes, all of which are checked. Below these are fields for "Primary key" (containing the value M/E4eeBsr7DA1Xcvw6ziFq1SDNbFX6E49Jfti8CRkbA=) and "Secondary key" (containing the value t\$ar0nreoyK7D/92R27d3l5bu4QuX00sRzzVqxG+GQ=). At the bottom, there are fields for "Connection string-primary key" (containing the value Endpoint=sb://demoeventhubnsbook.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=M/E4eeBsr7DA1Xcvw6ziFq1SDNbFX6E49Jfti8CRkbA=) and "Connection string-secondary key" (containing the value Endpoint=sb://demoeventhubnsbook.servicebus.windows.net/;SharedAccessKeyName=RootManageSharedAccessKey;SharedAccessKey=t\$ar0nreoyK7D/92R27d3l5bu4QuX00sRzzVqxG+GQ=).

Os hubs de eventos podem ser criados a partir do serviço de namespace de Hubs de Eventos, clicando em **Hubs de Eventos** no menu à esquerda e em **+ Hub de Eventos** na tela resultante:

The screenshot shows the Azure portal interface for a namespace named "DemoEventHubsbook". The left sidebar contains links for Overview, Access control (IAM), Tags, Diagnose and solve problems, Events, Settings (Shared access policies, Geo-Recovery, Firewalls and virtual networks, Properties, Locks, Automation script), and Entities (Event Hubs). The main content area is titled "DemoEventHubsbook - Event Hubs" and displays a search bar, a red box around the "+ Event Hub" button, and a refresh button. Below these are sections for NAME and Events, with the message "no Event Hubs yet.".

Em seguida, forneça informações sobre os valores de **Contagem de Partições** e **Retenção de Mensagem**, junto com o nome com o qual criará o hub de eventos. Em seguida, selecione Desativada para Capturar, conforme demonstrado na captura de tela a seguir:



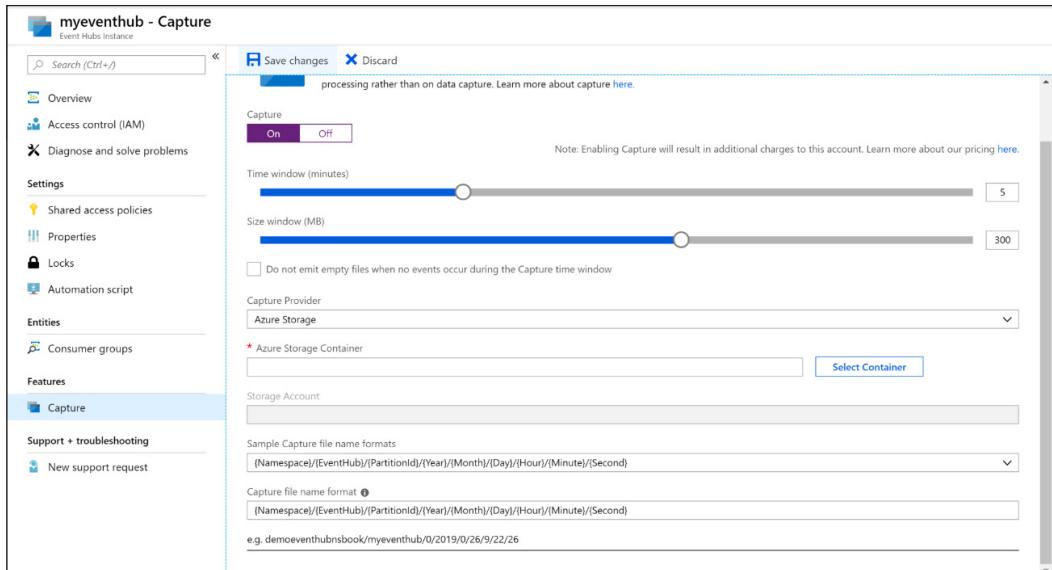
Políticas separadas podem ser atribuídas a cada hub de eventos adicionando uma nova política no nível do hub de eventos.

Depois de criar a política, a cadeia de conexão está disponível no item de menu esquerdo **Assinatura de Acesso Seguro** no portal do Azure.

Como um namespace pode consistir em vários hubs de eventos, a cadeia de conexão para um hub de eventos individual será semelhante ao bloco de códigos a seguir. A diferença aqui é o valor da chave e a adição de EntityPath com o nome do hub de eventos:

```
Endpoint=sb://azurertittereventdata.servicebus.windows.net/;SharedAccessKeyName=EventhubPolicy;SharedAccessKey=rxEu5K4Y2qsi5wEeOKuOvRnhtgW8xW35UBex4VlIKqg=;EntityPath=myeventhub
```

Tivemos que manter a opção **Capturar** definida como **Desativada** enquanto criamos o hub de eventos e ela pode ser ativada após a criação do hub de eventos. Isso ajuda a salvar eventos no Armazenamento de Blobs do Azure ou em uma conta do Azure Data Lake Storage automaticamente. A configuração para o intervalo de tamanho e tempo é mostrada na captura de tela a seguir:

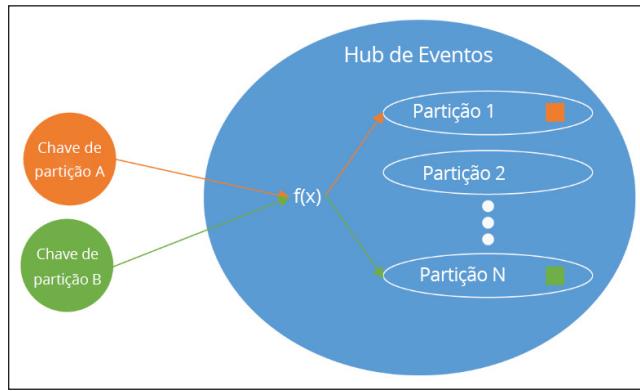


Nós não abrangemos os conceitos de partições e opções de retenção de mensagens ao criar este hub de eventos.

A partição é um conceito importante relacionado à escalabilidade de qualquer armazenamento de dados. Os eventos são retidos nos hubs de eventos por um período de tempo específico. Se todos os eventos forem guardados no mesmo armazenamento de dados, será extremamente difícil escalar esse armazenamento de dados. Todo produtor de evento se conectará ao mesmo armazenamento de dados e enviará seus eventos a ele. Compare isso com um armazenamento de dados que pode particionar os mesmos dados em vários armazenamentos de dados menores, cada um identificado exclusivamente com um valor. O armazenamento de dados menor é chamado de **partição** e o valor que define a partição é conhecido como **chave de partição**. Essa chave de partição é parte dos dados do evento.

Agora, os produtores de eventos podem se conectar ao hub de eventos e, com base no valor da chave de partição, o hub de eventos armazenará os dados em uma partição apropriada. Isso permitirá que o hub de eventos ingira vários eventos ao mesmo tempo em paralelo.

Decidir sobre o número de partições é um aspecto crucial para a escalabilidade de um hub de eventos. O diagrama a seguir mostra que os dados ingeridos são armazenados na partição apropriada internamente pelos Hubs de Eventos usando a chave de partição:



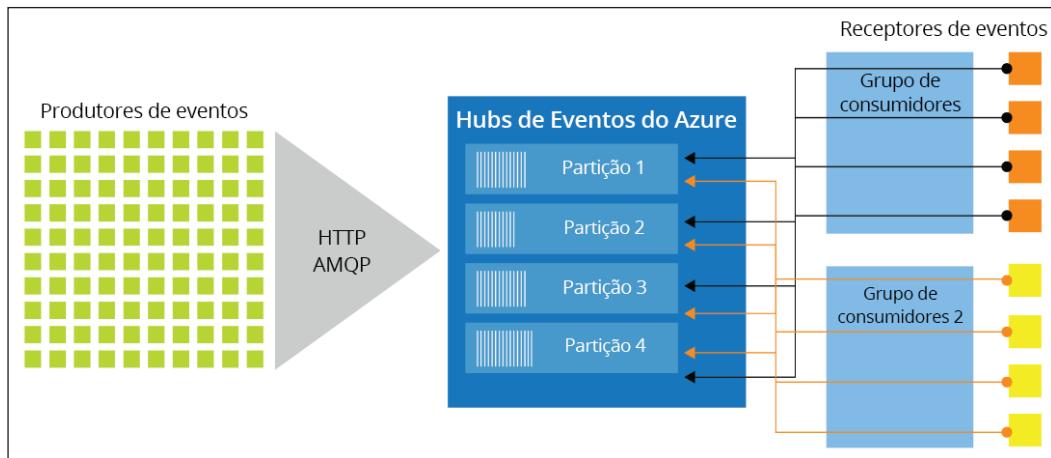
É importante entender que a mesma partição pode ter várias chaves. O usuário decide quantas partições são necessárias e o hub de eventos decide internamente a melhor maneira de alocar as chaves de partição entre eles. Cada partição armazena dados de forma ordenada usando um carimbo de data/hora e eventos mais recentes são anexados ao final da partição.

É importante observar que não é possível alterar o número de partições depois que o hub de eventos é criado.

Também é importante lembrar que as partições ajudam a trazer paralelismo e simultaneidade para aplicativos que leem os eventos. Por exemplo, se houver 10 partições, 10 leitores paralelos poderão ler os eventos sem qualquer degradação na performance.

Grupos de consumidores

Os consumidores são aplicativos que leem eventos de um hub de eventos. Os grupos de consumidores são criados para que os consumidores se conectem a ele a fim de ler os eventos. Pode haver vários grupos de consumidores para um hub de eventos e cada um deles tem acesso a todas as partições de um hub de eventos. Cada grupo de consumidores faz uma consulta sobre os eventos nos hubs de eventos. Os aplicativos podem usar grupos de consumidores e cada aplicativo receberá uma exibição diferente dos eventos dos hubs de eventos. Um grupo padrão de consumidores `$default` é criado ao criar um hub de eventos. É recomendável para um consumidor estar associado a um grupo de consumidores para obter excelente performance. No entanto, é possível ter cinco leitores em cada partição em um grupo de consumidores:



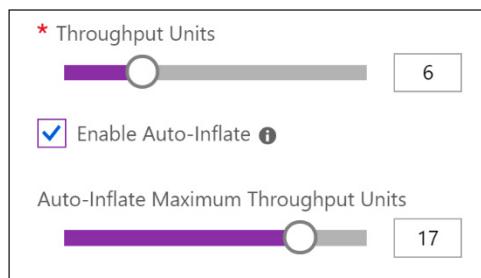
Taxa de transferência

As partições ajudam na escalabilidade, enquanto a taxa de transferência ajuda na capacidade por segundo. Então, qual é a capacidade em termos de Hubs de Eventos? É a quantidade de dados que podem ser manipulados por segundo.

Nos Hubs de Eventos, uma única unidade de taxa de transferência permite o seguinte:

- 1 MB de dados de ingestão por segundo ou 1.000 eventos por segundo (o que acontecer primeiro)
- 2 MB de dados de saída por segundo ou 4.096 eventos por segundo (o que acontecer primeiro)

A inflação automática ajuda a aumentar a taxa de transferência automaticamente se o número de eventos de entrada/saída ou o tamanho total de entrada/saída ultrapassar um limite. Em vez de limitar, a taxa de transferência aumentará e diminuirá. A configuração da taxa de transferência no momento da criação do namespace é mostrada na captura de tela a seguir. Novamente, deve-se pensar cuidadosamente em decidir as unidades de taxa de transferência:

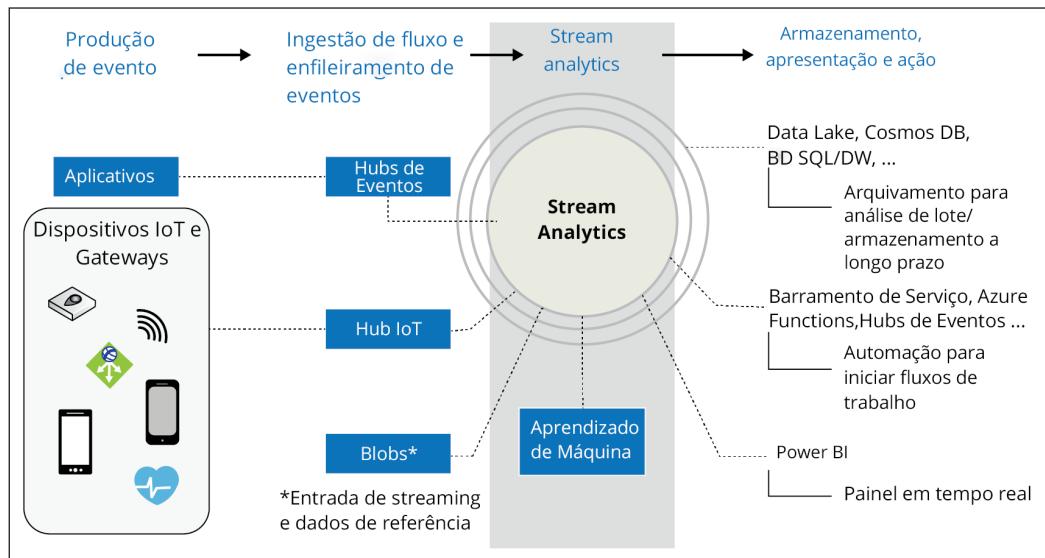


Uma cartilha sobre o Stream Analytics

Os Hubs de Eventos são apenas uma plataforma de ingestão de eventos; portanto, precisamos de outro serviço que possa processar esses eventos como um fluxo e não apenas como dados armazenados. O Stream Analytics ajuda no processamento e no exame de um fluxo do big data, e os trabalhos do Stream Analytics ajudam a executar o processamento de eventos.

O Stream Analytics pode processar milhões de eventos por segundo e é muito fácil começar com ele. O Azure Stream Analytics é uma PaaS totalmente gerenciada pelo Azure. Os clientes do Stream Analytics não precisam gerenciar o hardware e a plataforma subjacentes.

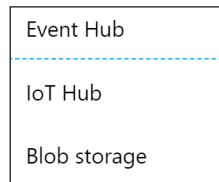
Cada trabalho comprehende várias entradas, saídas e uma consulta, que transforma os dados recebidos em novos resultados. Toda a arquitetura do Stream Analytics é mostrada no diagrama a seguir:



No diagrama anterior, as fontes de eventos são exibidas na extrema esquerda. Estas são as fontes que produzem os eventos. Podem ser dispositivos IoT, aplicativos personalizados escritos em qualquer linguagem de programação ou eventos provenientes de outras plataformas do Azure, como Log Analytics ou Application Insights.

Esses eventos devem primeiro ser ingeridos no sistema e existem vários serviços do Azure que podem ajudar a ingerir esses dados. Já vimos os Hubs de Eventos e como eles ajudam na ingestão de dados. Há outros serviços, como IoT do Hub, que também ajudam na ingestão de dados específicos do dispositivo e do sensor. A IoT do Hub e a ingestão são abordadas em detalhes no *Capítulo 15, Como projetar soluções IoT*. Esses dados ingeridos passam por processamento à medida que chegam em um fluxo, e esse processamento é feito usando o Stream Analytics. A saída do Stream Analytics pode ser uma plataforma de apresentação, como o Power BI, mostrando dados em tempo real para os stakeholders, ou uma plataforma de armazenamento, como Cosmos DB, Data Lake Storage ou Armazenamento do Azure, a partir das quais os dados podem ser lidos e executados posteriormente pelas filas do Azure Functions e do Barramento de Serviço.

O Stream Analytics é capaz de ingerir milhões de eventos por segundo e tem a capacidade de executar consultas sobre eles. No momento em que este artigo foi escrito, o Stream Analytics oferece suporte às três origens de eventos listadas na captura de tela a seguir:



Os dados de entrada são compatíveis com qualquer um dos três formatos a seguir:

- **JavaScript Object Notation (JSON)**: um formato leve e baseado em texto simples, legível para humanos. Consiste em pares nome-valor; um exemplo de um evento JSON é o seguinte:

```
{
  "SensorId" : 2,
  "humidity" : 60,
  "temperature" : 26C
}
```

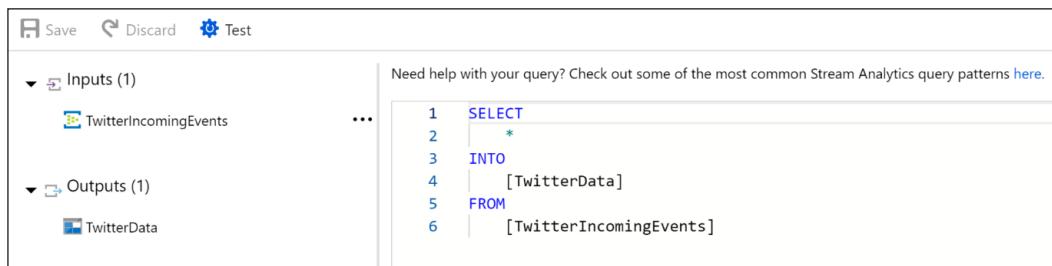
- **Valores Separados por Vírgulas (CSV)**: também são valores de texto simples, separados por vírgulas. Exemplos de CSV são mostrados na próxima imagem. A primeira linha é o cabeçalho contendo três campos seguidos por duas linhas de dados.

SensorId, humidity, temperature
2,60,26C
3,65,31C

- **Avro**: esse formato é semelhante ao JSON; no entanto, ele é armazenado em um formato binário, em vez de um formato de texto.

O Stream Analytics não só pode receber eventos, mas também fornece recursos avançados de consulta para os dados que recebe. As consultas podem extrair insights importantes dos fluxos de dados temporais e gerá-los.

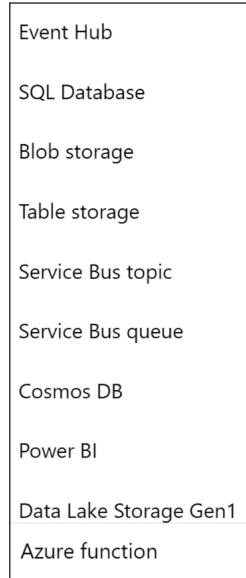
Conforme mostrado na captura de tela a seguir, há uma entrada e uma saída; a consulta move os eventos da entrada para a saída. A cláusula `INTO` refere-se ao local de saída e a cláusula `FROM` refere-se ao local de entrada. As consultas são muito semelhantes às consultas SQL; portanto, a curva de aprendizado não é muito alta para programadores SQL:



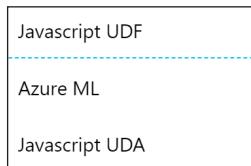
The screenshot shows the Azure Stream Analytics query editor interface. At the top, there are buttons for Save, Discard, and Test. Below that, there are sections for Inputs and Outputs. The Inputs section contains one item: TwitterIncomingEvents. The Outputs section contains one item: TwitterData. To the right of these sections is a code editor window displaying a Stream Analytics query:

```
1  SELECT
2    *
3  INTO
4    [TwitterData]
5  FROM
6    [TwitterIncomingEvents]
```

Os Hubs de Eventos também fornecem mecanismos para enviar saídas de consultas para destinos. No momento da escrita, o Stream Analytics oferece suporte a vários destinos para eventos e saídas de consulta. Esses destinos são mostrados na captura de tela a seguir:



Também é possível definir funções personalizadas que podem ser reutilizadas nas consultas. Há três opções fornecidas para definir funções personalizadas, da seguinte forma:



Ambiente de hospedagem

Os trabalhos do Stream Analytics podem ser executados em hosts que estão em execução na nuvem ou podem ser executados em dispositivos do IoT Edge. Os dispositivos do IoT Edge são dispositivos que estão perto de sensores de IoT, em vez de na nuvem. A captura de tela a seguir mostra a folha **Novo trabalho do Stream Analytics**:

The screenshot shows the 'New Stream Analytics job' dialog box. It contains the following fields:

- * Job name: Enter job name
- * Subscription: RiteshSubscription
- * Resource group: Select existing... (with 'Create new' link)
- * Location: East US
- Hosting environment: Cloud (selected) / Edge
- Streaming units (1 to 120): 3

Unidades de streaming

Na captura de tela anterior, podemos ver que a única configuração exclusiva para o Stream Analytics é **Unidades de Streaming**. As unidades de streaming referem-se aos recursos (isto é, CPU e memória) que são atribuídos para executar um trabalho do Stream Analytics. As unidades de streaming mínimas e máximas são 1 e 120, respectivamente.

Dependendo da quantidade de dados e do número de consultas executadas nesses dados, as unidades de streaming devem ser pré-alocadas; caso contrário, o trabalho falhará.

É possível aumentar e diminuir as unidades de streaming do portal do Azure.

Exemplo de aplicativo usando Hubs de Eventos e Stream Analytics

Nesta seção, estaremos criando um exemplo de aplicativo que comprehende vários serviços do Azure, incluindo os Aplicativos Lógicos do Azure, os Hubs de Eventos do Azure, o Armazenamento do Azure e o Azure Stream Analytics.

Neste exemplo de aplicativo, estaremos lendo todos os tweets que contenham a palavra "Azure" e armazenando-os em uma conta de Armazenamento do Azure.

Para criar essa solução, primeiro precisamos provisionar todos os recursos necessários.

Provisionamento de um novo grupo de recursos

Navegue até o portal do Azure, faça login usando credenciais válidas, clique em **+ Criar um recurso** e pesquise por **Grupo de recursos**. Selecione **Grupo de recursos** nos resultados da pesquisa e crie um novo grupo de recursos. Em seguida, forneça um nome e escolha um local apropriado. Observe que todos os recursos devem ser hospedados no mesmo grupo de recursos e local para que seja fácil excluí-los:

The screenshot shows the 'Create a resource group' wizard in the Azure portal. The top navigation bar includes 'Home > New > Marketplace > Everything > Resource group > Create a resource group'. The main title is 'Create a resource group'. Below it, there are three tabs: 'Basics' (selected), 'Tags', and 'Review + Create'. A descriptive text explains what a resource group is: 'Resource group - A container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization.' A 'Learn more' link is provided. The 'PROJECT DETAILS' section contains fields for 'Subscription' (set to 'RiteshSubscription') and 'Resource group' (set to 'TwitterAnalysis'). The 'RESOURCE DETAILS' section contains a field for 'Region' (set to 'West Europe').

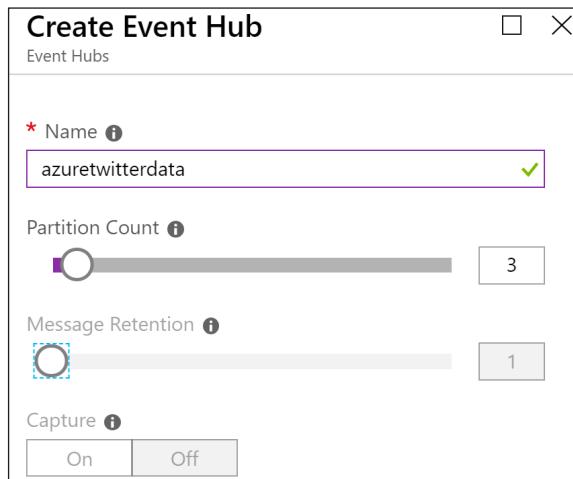
Criação de um namespace de Hubs de Eventos

Clique em + Criar um recurso e procure por Hubs de eventos. Selecione **Hubs de eventos** nos resultados da pesquisa e crie um novo hub de eventos. Em seguida, forneça um nome e local e selecione uma assinatura com base no grupo de recursos que foi criado anteriormente. Selecione **Padrão** como a camada de preço e também selecione **Habilitar Inflação Automática**, como mostrado na captura de tela a seguir:

The screenshot shows the 'Create Namespace' dialog box for Azure Event Hubs. The form includes fields for Name (set to 'AzureTwitterEventData'), Pricing tier (set to 'Standard (20 Consumer groups, 1000 B...)'), and various configuration options like Kafka and redundancy. It also includes dropdowns for Subscription ('Visual Studio Enterprise') and Resource group ('TwitterAnalysis'), and a Location ('West Europe'). The 'Throughput Units' section features a slider set at 1, and the 'Enable Auto-Inflate' checkbox is checked. A note at the bottom states: 'Namespace already exists. Enter a different name.'

Criação de um hub de eventos

No serviço de namespace Hubs de Eventos, clique em **Hubs de eventos** no menu à esquerda e, em seguida, clique em **+ Hubs de eventos** para criar um novo hub de eventos. Nomeie-o como `azuretwitterdata` e forneça um número ideal de partições e um valor de **Retenção de Mensagem**:



Provisionamento de um aplicativo lógico

Depois que o grupo de recursos for provisionado, clique em **+ Criar um recurso** e procure por **Aplicativos Lógicos**. Selecione **Aplicativos Lógicos** nos resultados da pesquisa e crie um novo aplicativo lógico. Em seguida, forneça um nome e local e selecione uma assinatura com base no grupo de recursos criado anteriormente. É recomendável habilitar **Análise do Log**. Os Aplicativos Lógicos são abordados em mais detalhes no *Capítulo 7, Soluções de integração do Azure*. O aplicativo lógico é responsável por conectar-se ao Twitter usando uma conta e buscar todos os tweets com Azure neles:

Logic App

Create

* Name: GetAzureTwitterData

* Subscription: Visual Studio Enterprise

* Resource group: TwitterAnalysis (Create new)

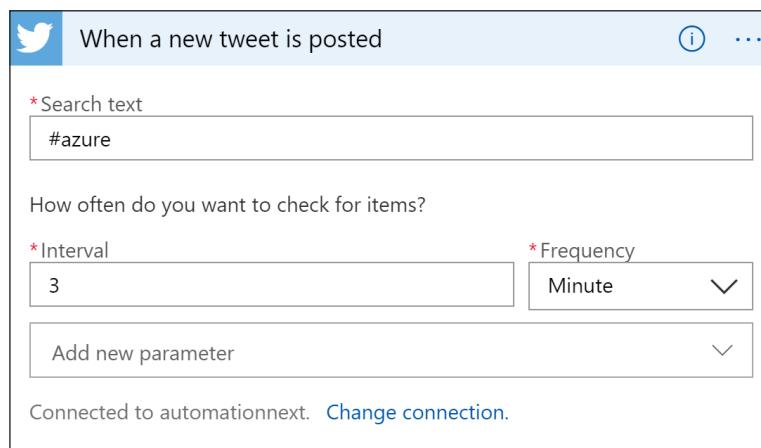
* Location: West Europe

Log Analytics: On

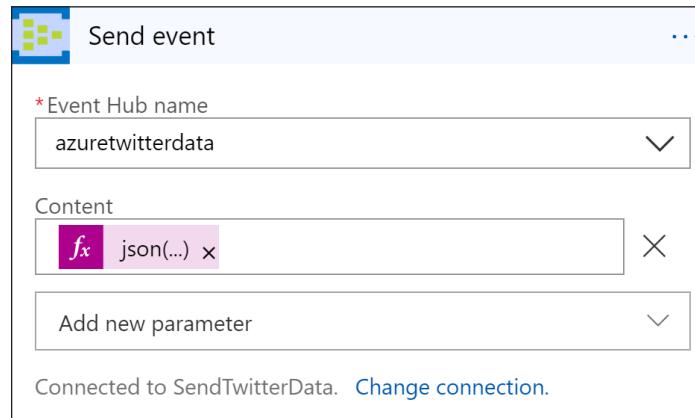
* Log Analytics workspace: No Log Analytics workspace found

You can add triggers and actions to your Logic App after creation.

Depois que o aplicativo lógico for criado, selecione **Quando um novo tweet for postado** ação na superfície de design, faça login e configure-o conforme mostrado na captura de tela a seguir. Você precisará de uma conta no Twitter válida antes de configurar este gatilho:

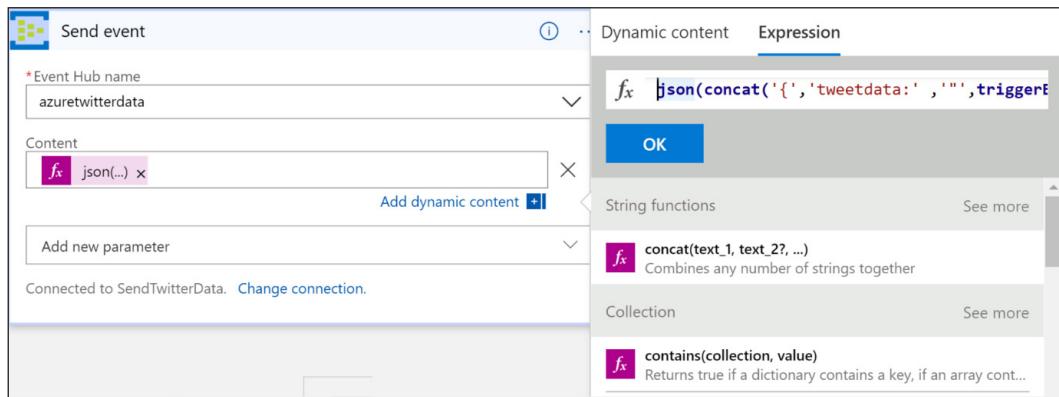


Em seguida, solte uma ação de **Enviar evento** na superfície do designer; essa ação é responsável pelo envio de tweets para o hub de eventos:



Selecione o nome do hub de eventos que foi criado em uma etapa anterior.

O valor especificado na caixa de texto do conteúdo é uma expressão que foi redigida dinamicamente usando funções fornecidas pelo Aplicativo lógico e dados do Twitter. Ao clicar em **Adicionar conteúdo dinâmico**, uma caixa de diálogo é fornecida, por meio da qual é possível redigir a expressão:



O valor da expressão é o seguinte:

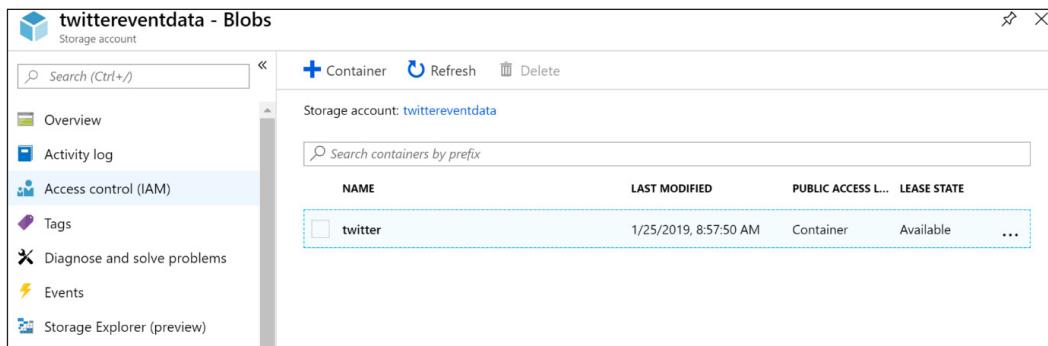
```
json(concat('{' , 'tweetdata:' , "'", triggerBody()?['TweetText'] , "'",
'}'))
```

Provisionamento na conta de armazenamento

Clique em **+ Criar um recurso** e procure Conta de Armazenamento. Selecione **Conta de Armazenamento** nos resultados da pesquisa e crie uma nova conta de armazenamento. Em seguida, forneça um nome e local e selecione uma assinatura com base no grupo de recursos que foi criado anteriormente. Finalmente, selecione **StorageV2** para **Tipo de Conta**, **Padrão para Performance** e **Armazenamento com redundância local (LRS)** para o campo **Replicação**.

Criação de um contêiner de armazenamento

O Stream Analytics emitirá os dados como arquivos que serão armazenados em um contêiner de Armazenamento de Blobs. Um contêiner chamado **twitter** é criado no Armazenamento de Blobs, como mostrado captura de tela a seguir:



The screenshot shows the Azure Storage Explorer interface for the 'twittereventdata' storage account. On the left, there's a sidebar with options like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Events, and Storage Explorer (preview). The main area is titled 'Blobs' and shows a list of containers. A single container named 'twitter' is listed, with details: LAST MODIFIED (1/25/2019, 8:57:50 AM), PUBLIC ACCESS L... (Container), and LEASE STATE (Available). There are also 'Container', Refresh, and Delete buttons at the top of the list.

Criação de trabalhos do Stream Analytics

Vamos criar um novo trabalho do Stream Analytics com um ambiente de hospedagem na nuvem e definir as unidades de streaming para as configurações padrão. A entrada para este trabalho do Stream Analytics vem do hub de eventos e, portanto, precisamos configurá-lo no menu Entradas:

A saída para o trabalho do Stream Analytics é uma conta de Armazenamento de Blobs, portanto, precisamos configurar a saída de acordo. Forneça um padrão de caminho adequado para este exercício; por exemplo, {datetime:ss} é o padrão de caminho que estamos usando para este exercício:

StreamAzureTwitterJob - Outputs

NAME SINK

TwitterData	Blob storage
-------------	--------------

TwitterData

* Output alias TwitterData

Provide Blob storage settings manually
 Select Blob storage from your subscriptions

Subscription RiteshSubscription

* Storage account twittereventdata

Storage account key *****

* Container Create new Use existing
twitter

Path pattern (datetime:ss)

Date format YYYY/MM/DD

Time format HH

Save

A consulta é bem simples; estamos apenas copiando os dados da entrada para a saída:

StreamAzureTwitterJob - Query

Inputs (1) TwitterIncomingEvents

Outputs (1) TwitterData

```

1 SELECT
2 *
3 INTO
4     [TwitterData]
5 FROM
6     [TwitterIncomingEvents]

```

Execução do aplicativo

O aplicativo lógico deve estar habilitado e o Stream Analytics deve estar em execução. Agora, execute o aplicativo lógico; ele deve ter trabalhos criados, conforme mostrado na captura de tela a seguir:

GetAzureTwitterData

Resource group (change) : TwitterAnalysis

Location : West Europe

Subscription (change) : RiteshSubscription

Subscription ID : 9755ffce-e94b-4332-9be8-1ade15e78909

Status : Enabled

Runs last 24 hours : 217 successful, 1275 failed

Integration Account : ---

Summary

Trigger

TWITTER
When a new tweet is posted

FREQUENCY
Runs every 3 minutes

EVALUATION
Evaluated 2357 times, fired 1494 times in the last 24 hours
[See trigger history](#)

Runs history

STATUS	START TIME	IDENTIFIER	DURATION
Succeeded	1/26/2019, 10:16 AM	08586530910864161272447961552CU24	299 Milliseconds
Succeeded	1/26/2019, 10:15 AM	08586530911478685688296257377CU29	447 Milliseconds

O contêiner **Conta de Armazenamento** deve obter dados, conforme mostrado na captura de tela a seguir:

twitter

Container

Location: twitter / 24

Properties

URL : https://twittereventdata.blob.core.windows.net/twitter/24/0_48c599701d414248959070c92c937b3a_1.json

LAST MODIFIED : 1/25/2019, 9:06:25 AM

CREATION TIME : 1/25/2019, 9:06:24 AM

TYPE : Block blob

SIZE : 10.8 KiB

SERVER ENCRYPTED : true

ETAG : 0x8D682CE4AF8642B

CONTENT-MDS : -

LEASE STATUS : Unlocked

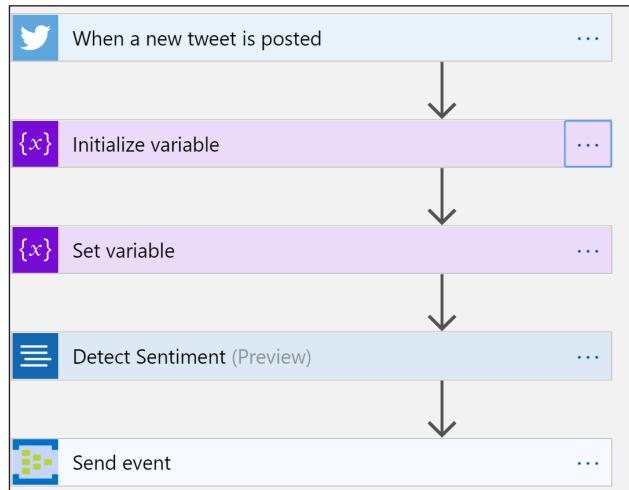
LEASE STATE : Available

LEASE DURATION : -

COPY STATUS : -

COPY COMPLETION TIME : -

Como exercício, você pode estender essa solução de amostra e avaliar o sentimento dos tweets a cada três minutos. O fluxo de trabalho de Aplicativos Lógicos para esse exercício será o seguinte:



Para o exercício de detecção de sentimento, você precisará usar a API de Análise de Texto, que deve ser configurada antes de ser usada em Aplicativos Lógicos.

Resumo

Este capítulo abordou tópicos relacionados a eventos de streaming e eventos temporais. Os eventos são gerados de várias origens e, para obter insights em tempo real sobre atividades e seus eventos relacionados, serviços como os Hubs de Eventos e o Stream Analytics desempenham um papel significativo. Hubs de Eventos é um serviço que ajuda a ingerir milhões de eventos por segundo e armazená-los temporariamente. Esses dados podem ser enviados ao Stream Analytics para processamento. Durante o processamento, insights adicionais podem ser obtidos usando consultas personalizadas e podem ser enviados como saídas para vários destinos. Esses serviços funcionam em dados em tempo real que não são armazenados em nenhum armazenamento permanente.

No próximo capítulo, passaremos pelo processo de criação de uma solução de IoT usando o Hub IoT do Azure e exploraremos seus vários recursos.

15

Como projetar soluções IoT

Até agora, nós temos lidado com preocupações relacionadas à arquitetura e suas soluções no Azure em geral. No entanto, este capítulo não se baseia na arquitetura generalizada. De fato, ele explora uma das tecnologias mais revolucionárias deste século. Este capítulo discutirá os detalhes da **Internet das Coisas (IoT)** e do Azure.

Este capítulo abordará especificamente os seguintes tópicos:

- Azure e IoT
- Visão geral do Azure IoT
- Gerenciamento de dispositivos:
 - Registro de dispositivos
 - Comunicação do dispositivo para o hub IoT
- Escala de soluções IoT
- Alta disponibilidade de soluções IoT
- Protocolos IoT
- Como usar as propriedades da mensagem para roteamento de mensagens

IoT

Para entender o que é IoT, vamos voltar alguns anos.

A Internet foi inventada durante a segunda metade do último século e tornou-se amplamente disponível entre nós. Durante esse período, quase todo mundo avançou em direção a ter uma presença na internet e começou a criar suas próprias páginas da Web estáticas. Finalmente, o conteúdo estático tornou-se dinâmico e mais conteúdo era gerado dinamicamente com base no contexto. Em quase todos os casos, um navegador era necessário para acessar a internet. Havia uma infinidade de navegadores disponíveis e, sem eles, o uso da internet era um desafio.

Durante a primeira década deste século, houve um acontecimento interessante no desenvolvimento: o surgimento de dispositivos portáteis, como telefones celulares e tablets. Os celulares começaram a ficar cada dia mais baratos e disponíveis de forma onipresente. Os recursos de hardware e software desses dispositivos portáteis estavam melhorando consideravelmente; tanto que as pessoas começaram a usar navegadores em seus dispositivos móveis, em vez de em desktops. Mas uma mudança particularmente distinta foi o surgimento de aplicativos móveis. Esses aplicativos móveis eram obtidos de alguma loja por download e eram conectados à internet para comunicar-se com sistemas de back-end. Até o final da década passada, havia milhões de aplicativos disponíveis com quase todas as funcionalidades imagináveis dentro deles. O sistema de back-end para esses aplicativos foi construído na nuvem para que eles pudessem ser escalados rapidamente. Essa foi a era da conexão entre aplicativos e servidores.

Mas esse foi o auge da inovação? Qual foi a próxima evolução da Internet? Bem, outro paradigma agora está tomando o centro do palco: IoT. Em vez de apenas celulares e tablets conectados à internet, por que não pode haver outros dispositivos conectados à internet? Anteriormente, esses dispositivos estavam disponíveis apenas em mercados selecionados; eles eram caros, não estavam disponíveis para as massas e tinham recursos limitados de hardware e software. No entanto, desde a primeira parte desta década, a comercialização desses dispositivos tem crescido em grande escala. Esses dispositivos estão se tornando menores e menores, têm mais capacidade em termos de hardware e software, têm mais poder de computação e armazenamento, podem se conectar à internet em vários protocolos e podem ser conectados a quase tudo. Esta é a era da conexão de dispositivos a servidores, aplicativos e outros dispositivos.

Isso levou à formulação da ideia de que os aplicativos de IoT podem, de fato, mudar a forma como as indústrias estão operando. As soluções mais recentes que eram inéditas começaram a se tornar realidade. Agora, esses dispositivos podem ser conectados a qualquer coisa; podem obter informações e enviá-las para um sistema de back-end que pode assimilar informações de todos os dispositivos e executar ações ou relatar incidentes.

Exemplos de aplicativos de IoT incluem sistemas de rastreamento de veículos, que podem rastrear todos os parâmetros vitais de um veículo e enviar detalhes para um armazenamento de dados centralizado para análise; serviços públicos inteligentes, como o controle de níveis de poluição, temperatura e congestionamentos; além de atividades relacionadas à agricultura, como medição da fertilidade do solo, a umidade e muito mais.

Arquitetura da IoT

Antes de entrar no Azure e em seus recursos e serviços relacionados à IoT, é importante compreender os vários componentes necessários para criar soluções completas de IoT.

Imagine que dispositivos IoT em todo o mundo estão enviando milhões de mensagens a cada segundo para um banco de dados centralizado. Por que esses dados estão sendo coletados? Bem, a resposta é extrair informações valiosas sobre eventos, anomalias e exceções que estão relacionadas a tudo quando esses dispositivos estão monitorando.

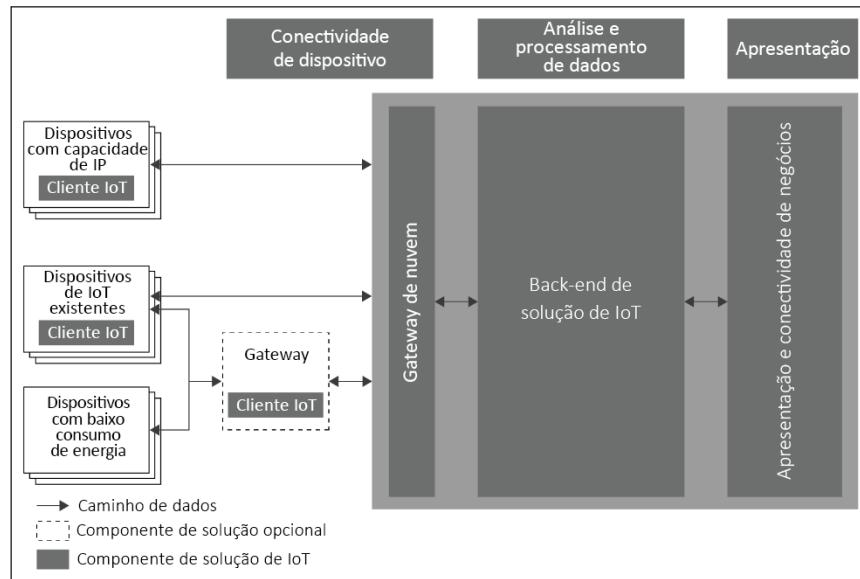
Vamos entender isso mais detalhadamente.

A arquitetura da IoT pode ser dividida em fases distintas, como a seguir:

- Conectividade
- Identidade
- Captura
- Ingestão
- Armazenamento
- Transformação
- Análise
- Apresentação

O diagrama a seguir mostra uma arquitetura genérica baseada em IoT. Os dados são gerados ou coletados por dispositivos e enviados por meio do gateway de nuvem. O gateway de nuvem, por sua vez, envia os dados para vários serviços de back-end para processamento.

Os gateways de nuvem são componentes opcionais. Eles devem ser usados quando os próprios dispositivos não são capazes de enviar solicitações para serviços de back-end, seja por causa das limitações de recursos ou pela falta de uma rede confiável. Esses gateways de nuvem podem agrupar dados de vários dispositivos e enviá-los para os serviços de back-end. Em seguida, os dados podem ser processados pelos serviços de back-end e mostrados como insights ou painéis aos usuários:



Conectividade

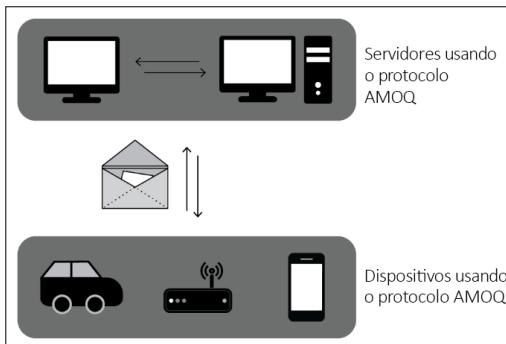
Os dispositivos de IoT precisam se comunicar para se conectarem a outros dispositivos. Existem vários tipos de conectividade; por exemplo, a conectividade pode existir entre dispositivos em uma região, entre dispositivos e um gateway centralizado e entre dispositivos e uma plataforma IoT.

Em todos esses casos, os dispositivos IoT precisam de capacidade de conectividade. Essa capacidade poderia ser na forma de conectividade com a Internet, Bluetooth, infravermelho ou qualquer outra comunicação com dispositivos próximos.

No entanto, alguns dispositivos IoT podem não ter a capacidade de se conectar à internet. Nesses casos, eles podem se conectar por um gateway que, por sua vez, possui conectividade com a internet.

Os dispositivos IoT usam protocolos para enviar mensagens. Os principais protocolos são o **Advanced Message Queuing Protocol (AMQP)** e o **Message Queue Telemetry Transport (MQTT)**.

Os dados de dispositivos devem ser enviados para uma infraestrutura de TI. O protocolo MQTT é um protocolo de dispositivo para servidor que os dispositivos podem usar para enviar dados de telemetria e outras informações para servidores. Uma vez que o servidor recebe uma mensagem por meio do protocolo MQTT, precisa transportar a mensagem para outros servidores usando uma tecnologia confiável baseada em mensagens e filas. AMQP é o protocolo preferencial para mover mensagens de forma confiável e previsível entre os servidores na infraestrutura de TI:



Os servidores que recebem as mensagens iniciais de dispositivos IoT devem enviar essas mensagens para outros servidores para qualquer processamento que seja necessário, como salvamento em logs, avaliação, análise e apresentação.

Alguns dispositivos não têm a capacidade de se conectar à Internet ou não oferecem suporte a protocolos que são compatíveis com outras tecnologias de servidor. Para permitir que esses dispositivos funcionem com uma plataforma IoT e na nuvem, gateways intermediários podem ser usados. Os gateways ajudam na integração de dispositivos cuja conectividade e capacidade de rede sejam lentas e não consistentes; esses dispositivos podem usar protocolos não padronizados ou suas capacidades podem ser limitadas em termos de recursos e potência.

Nas circunstâncias em que os dispositivos precisam de infraestrutura adicional para se conectar a serviços back-end, os gateways cliente podem ser implantados. Esses gateways recebem mensagens de dispositivos próximos e os encaminham (ou enviam por push) para a infraestrutura de TI e para a plataforma de IoT para mais consumo. Esses gateways podem traduzir protocolos, se necessário.

Identidade

Os dispositivos IoT devem ser registrados na plataforma de nuvem. Os dispositivos que não estão registrados não devem ter permissão para conectar a uma plataforma de nuvem. Os dispositivos devem ser registrados e obter uma identidade. Um dispositivo deve enviar suas informações de identidade durante a conexão com a nuvem. Se o dispositivo não conseguir enviar essas informações de identidade, a conectividade deverá falhar. Veremos mais adiante neste capítulo como gerar uma identidade para um dispositivo usando um aplicativo de simulação.

Captura

Os dispositivos IoT devem ser capazes de capturar informações. Eles devem ser capazes, por exemplo, de ler ou monitorar o conteúdo sobre a umidade no ar ou no solo. Essas informações podem ser capturadas com base na frequência – talvez até uma vez por segundo. Assim que as informações são capturadas, o dispositivo deve ser capaz de enviá-las para a plataforma de IoT para processamento. Se um dispositivo não tiver a capacidade de se conectar diretamente à plataforma de IoT, poderá se conectar a um gateway de nuvem intermediário e ter que enviar por push as informações capturadas. O tamanho dos dados capturados e a frequência de captura são os aspectos mais importantes para o dispositivo. O fato de um dispositivo ter armazenamento local para conseguir armazenar temporariamente os dados capturados é outro aspecto importante que deve ser considerado. Por exemplo, um dispositivo poderá funcionar em modo offline se houver armazenamento local disponível suficiente. Até mesmo os dispositivos móveis às vezes atuam como dispositivos IoT conectados a diversos instrumentos e têm a capacidade de armazenar dados.

Ingestão

Os dados capturados e gerados por dispositivos devem ser enviados para uma plataforma IoT capaz de ingerir e consumir esses dados para extrair deles informações significativas e insights. O serviço de ingestão é essencial porque a disponibilidade e a escalabilidade do serviço afetam a taxa de transferência dos dados de entrada. Se os dados começarem a ficar limitados por problemas de escalabilidade ou se a ingestão não for possível devido a problemas de disponibilidade, os dados serão perdidos e o conjunto de dados poderá ser tornar tendencioso ou distorcido.

Armazenamento

As soluções IoT geralmente lidam com milhões ou, até mesmo, bilhões de registros, com terabytes ou petabytes de dados. Esses são dados valiosos que podem fornecer insights sobre as operações e sua integridade. Esses dados precisam ser armazenados de tal forma que possam ser analisados. O armazenamento deve estar prontamente disponível para análise, aplicativos e serviços para consumi-lo. As soluções de armazenamento devem fornecer taxa de transferência e latência adequadas de uma perspectiva de performance, bem como ser altamente disponíveis, escalonáveis e seguras.

Transformação

Em geral, as soluções IoT são orientadas a dados e têm um volume de dados consideravelmente alto para lidar. Imagine que cada carro tenha um dispositivo e esteja enviando mensagens a cada cinco segundos. Se existirem um milhão de carros enviando mensagens, isso será igual a 288 milhões de mensagens por dia e 8 bilhões de mensagens por mês.

Juntos, esses dados têm muitas informações e insights ocultos; no entanto, extrair informações desse tipo de dados por mera análise é bastante difícil. Os dados capturados e armazenados por dispositivos IoT podem ser consumidos para resolver problemas de negócios, mas nem todos os dados capturados são importantes. Apenas um subconjunto de dados pode ser necessário para resolver um problema. Além disso, os dados que os dispositivos IoT coletam podem não ser consistentes. Para garantir que os dados sejam consistentes e não tendenciosos ou distorcidos, as transformações adequadas devem ser executadas neles para torná-los prontos para análise. Durante essa transformação, os dados são filtrados, classificados, removidos, enriquecidos e transformados em uma estrutura, de forma que os dados possam ser consumidos por componentes e aplicativos downstream.

Análise

Os dados transformados na etapa anterior tornam-se a entrada para a etapa de análise. Dependendo do problema em questão, existem diferentes tipos de análise que podem ser executados nos dados transformados.

A seguir estão os diferentes tipos de análise que podem ser executados:

- **Análise descritiva:** esse tipo de análise ajuda na procura de padrões e detalhes sobre o status dos dispositivos IoT e de sua integridade geral. Essa etapa identifica e resume os dados para consumo adicional por meio de etapas de análise mais avançadas. Ela ajudará no resumo, na descoberta de estatísticas relacionadas à probabilidade, na identificação do desvio e em outras tarefas estatísticas.
- **Análise de diagnóstico:** esse tipo de análise é mais avançado do que a análise descritiva. Ela se baseia na análise descritiva e tenta responder às consultas sobre o porquê de certas coisas acontecerem. Isto é, ela tenta encontrar as causas raiz dos eventos. Ela tenta encontrar respostas usando conceitos avançados, como hipótese e correlação.
- **Análise preditiva:** esse tipo de análise tenta prever acontecimentos com uma alta probabilidade de acontecer no futuro. Ela gera previsões que são baseadas em dados antigos; a regressão é um dos exemplos que se baseia em dados antigos. A análise preditiva poderia, por exemplo, prever o preço de um carro, o comportamento do estoque no mercado de ações, quando um pneu de carro vai estourar, e muito mais.
- **Análise prescritiva:** esse tipo de análise é o mais avançado. Essa etapa ajuda a identificar a ação que deve ser executada para garantir que a integridade dos dispositivos e das soluções não seja prejudicada e a identificar que medidas proativas sejam realizadas. Os resultados desta etapa de análise podem ajudar a evitar problemas futuros e eliminar os problemas em suas causas raiz.

Apresentação

A Análise ajuda na identificação de respostas, padrões e insights com base nos dados. Esses insights também precisam estar disponíveis para todos os stakeholders em formatos que eles consigam compreender. Para essa finalidade, painéis e relatórios podem ser gerados, de forma estatística ou dinâmica, e então podem ser apresentados aos stakeholders. Os stakeholders podem consumir esses relatórios para executar outras ações e melhorar continuamente suas soluções.

IoT do Azure

Agora, aprendemos sobre os vários estágios de soluções de IoT de ponta a ponta; cada um desses estágios é fundamental e sua implementação é imprescindível para o sucesso de qualquer solução. O Azure fornece muitos serviços para cada um desses estágios. Além desses serviços, o Azure fornece o Hub IoT, que é o principal serviço e plataforma de IoT do Azure. Ele é capaz de hospedar soluções de IoT complexas, altamente disponíveis e escalonáveis. Nós detalharemos bastante o Hub IoT depois de passarmos por alguns outros serviços:

Dispositivos	Conectividade de dispositivo	Armazenamento	Análise	Apresentação e ação
	Eventos	Banco de dados SQL	Aprendizado de Máquina	Serviços de Aplicativos
	Barramento de Serviço	Armazenamento de tabela/Blob	Stream Analytics	PowerBI
	Fontes de dados externas	Cosmos DB	HDInsights	Hubs de Notificação
		Fontes de dados externas	Data Factory	Serviços Móveis
		Time Series Insights	Databricks	Aplicativos Lógicos

Identidade

O Hub IoT do Azure também fornece serviços para autenticar dispositivos. Ele fornece uma interface para a geração de hashes de identidade exclusiva para cada dispositivo. Quando os dispositivos enviarem mensagens contendo esse hash, o Hub IoT poderá autenticá-las após a verificação do seu próprio banco de dados em busca da existência de tais hashes.

Captura

O Azure fornece gateways IoT que habilitam dispositivos que não estão em conformidade com o Hub IoT para se adaptar e enviar dados por push. Gateways locais ou intermediários podem ser implantados perto de dispositivos de forma que vários dispositivos possam se conectar a um único gateway para enviar suas informações. Da mesma forma, vários desses clusters de dispositivos com gateways locais também podem ser implantados. Pode haver um gateway de nuvem implantado na própria nuvem, o qual é capaz de aceitar dados de várias fontes e ingeri-los para Hubs IoT.

Ingestão

Um Hub IoT pode ser um único ponto de contato para dispositivos e outros aplicativos. Em outras palavras, a ingestão de mensagens IoT é de responsabilidade do serviço do Hub IoT. Existem outros serviços, tais como Hubs de Eventos e a infraestrutura de serviço de mensagens de Barramento de Serviço, que podem ingerir as mensagens recebidas; no entanto, os benefícios e as vantagens de usar Hub IoT para ingerir dados da IoT superam muito o uso de Hubs de Eventos e serviço de mensagens de Barramento de Serviço. Na verdade, os Hubs IoT foram feitos especificamente com a finalidade de ingerir mensagens IoT dentro do ecossistema do Azure, de forma que os outros serviços e componentes possam agir sobre eles.

Armazenamento

O Azure fornece várias maneiras de armazenar mensagens de dispositivos IoT. Essas contas de armazenamento incluem o armazenamento de dados relacionais, dados NoSQL sem esquemas e blobs:

- **Banco de dados SQL:** o banco de dados SQL fornece armazenamento de dados relacionais, JSON e documentos XML. Ele fornece uma linguagem de consulta SQL rica e usa um SQL Server completo como um serviço. Os dados dos dispositivos poderão ser armazenados em bancos de dados SQL caso estejam bem definidos e o esquema não precisará passar por mudanças frequentes.
- **Armazenamento do Azure:** o Armazenamento do Azure fornece armazenamento de tabelas e de blobs. O armazenamento de tabelas ajuda no armazenamento de dados como entidades, onde o esquema não é importante. Os blobs ajudam no armazenamento de arquivos em contêineres como blobs.
- **Cosmos DB:** o Cosmos DB é um banco de dados NoSQL de escala empresarial completo. Ele está disponível como um serviço que é capaz de armazenar dados sem esquema. É um banco de dados verdadeiramente distribuído que pode se estender por continentes, fornecendo alta disponibilidade e escalabilidade de dados.
- **Fontes de dados externas:** além dos serviços do Azure, os clientes podem trazer ou usar seus próprios armazenamentos de dados, como um SQL Server em máquinas virtuais do Azure, e podem usá-los para armazenar dados no formato relacional.

Transformação e análise

O Azure fornece vários recursos para executar tarefas e atividades em dados ingeridos. Alguns deles são mencionados como segue:

- **Data Factory:** o Azure Data Factory é um serviço de integração de dados baseados em nuvem que nos permite criar fluxos de trabalho orientados por dados na nuvem para orquestração e automatização da movimentação de dados e transformação de dados. O Azure Data Factory ajuda a criar e a agendar fluxos de trabalho orientados a dados (chamados pipelines) que podem ingerir dados de armazenamentos de dados diferentes; processar e transformar os dados usando serviços de computação, como **Azure HDInsight, Hadoop, Spark, Azure Data Lake Analytics** e **Azure Machine Learning**; além de publicar dados de saída para um data warehouse para aplicativos de **Business Intelligence (BI)**, em vez de uma plataforma tradicional **Extrair, Transformar e Carregar (ETL)**.
- **Azure HDInsight:** a Microsoft e a Hortonworks se uniram para ajudar as empresas oferecendo uma solução analítica de big data com o Azure. O HDInsight é um ambiente de serviço de nuvem totalmente gerenciado, de alto desempenho, powered by Apache Hadoop e Apache Spark usando Microsoft Azure HDInsight. Ajuda na aceleração de workloads perfeitamente usando o serviço de nuvem de big data líder da indústria da Microsoft e da Hortonworks.
- **Azure Stream Analytics:** é um serviço de análise de dados em tempo real totalmente gerenciado que ajuda na realização de cálculo e transformação no streaming de dados. O Stream Analytics pode examinar grandes volumes de dados fluindo de dispositivos ou processos, extrair informações do fluxo de dados e procurar padrões, tendências e relacionamentos.
- **Machine Learning:** é uma técnica de ciência de dados que permite que os computadores usem os dados existentes para previsão de comportamentos futuros, resultados e tendências. Usando o Machine Learning, os computadores aprendem sem serem explicitamente programados. O Azure Machine Learning é um serviço de análise preditiva baseado em nuvem que possibilita a rápida criação e implantação de modelos preditivos como soluções de análise. Ele fornece uma biblioteca de algoritmos pronta para uso para criar modelos em um PC conectado à internet e implantar soluções preditivas rapidamente.

Apresentação

Após a realização de análises adequadas nos dados, os dados deverão ser apresentados aos stakeholders em um formato consumível por eles. Existem inúmeras maneiras de apresentar insights de dados. Isso inclui a apresentação de dados por meio de aplicativos web implantados usando o Serviço de Aplicativo do Azure, enviando dados para hubs de notificação que então podem notificar aplicativos móveis e muito mais. No entanto, a abordagem ideal para apresentar e consumir insights é por meio de relatórios e painéis do Power BI. O Power BI é uma ferramenta de visualização da Microsoft que é usada para a renderização de relatórios dinâmicos e painéis na internet.

Hubs IoT

Os projetos de IoT são geralmente complexos por natureza. A complexidade surge devido ao alto volume de dispositivos e dados. Os dispositivos são incorporados em todo o mundo; por exemplo, monitorando e auditando dispositivos, que são usados para armazenar dados, transformar e analisar petabytes de dados e, finalmente, tomar ações com base em insights. Além disso, esses projetos têm longos períodos de gestação e seus requisitos continuam a mudar por causa dos prazos.

Se uma empresa quer embarcar em uma viagem com um projeto de IoT, acabará percebendo que os problemas que mencionamos não são facilmente resolvidos. Esses projetos precisam de hardware suficiente em termos de computação e armazenamento para lidar, além de serviços que possam trabalhar com grandes volumes de dados.

O Hub IoT é uma plataforma criada para ajudar a facilitar e a viabilizar projetos de IoT para entrega mais rápida, melhor e mais fácil. Ele fornece todos os recursos e os serviços necessários, incluindo o seguinte:

- Registro do dispositivo
- Conectividade de dispositivo
- Gateways de campo
- Gateways de nuvem
- Implementação de protocolos de indústria, como AMQP e MQTT
- Um hub para armazenar as mensagens recebidas
- O roteamento de mensagens baseadas em propriedades e conteúdo de mensagem
- Vários pontos de extremidade para diferentes tipos de processamento
- Conectividade com outros serviços no Azure para análise em tempo real e muito mais

Protocolos

O Hub IoT do Azure dá suporte nativo à comunicação sobre os protocolos MQTT, AMQP e HTTP. Em alguns casos, dispositivos ou gateways de campo podem não ser capazes de usar um desses protocolos padrão e exigirão adaptação do protocolo. Em tais casos, gateways personalizados podem ser implantados. Um gateway personalizado pode habilitar adaptação do protocolo para pontos de extremidade de Hub IoT ao criar uma ponte para o tráfego de e para o Hub IoT.

Registro do dispositivo

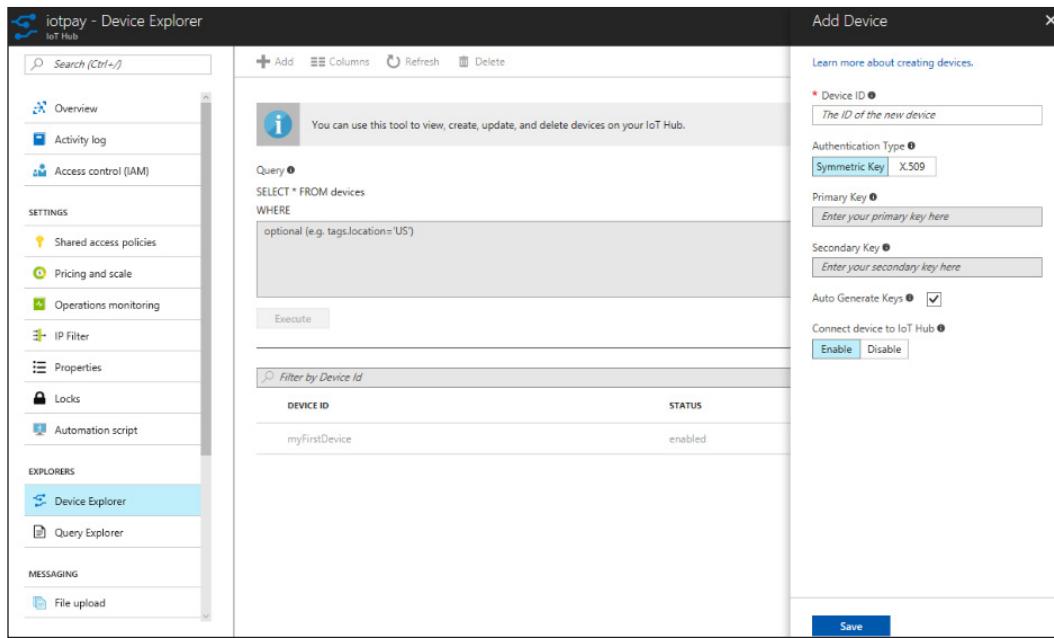
Os dispositivos devem ser registrados antes que eles possam enviar mensagens para um Hub IoT. O registro de dispositivos pode ser feito manualmente usando o portal do Azure ou pode ser automatizado usando o SDK do Hub IoT. O Azure também fornece aplicativos de simulação de amostra, por meio dos quais fica mais fácil registrar dispositivos virtuais para fins de desenvolvimento e de testes. Há também um simulador online de Raspberry Pi que pode ser usado como um dispositivo virtual, e depois, obviamente, haverá outros dispositivos físicos que poderão ser configurados para se conectar ao Hub IoT.

Se os leitores desejarem simular um dispositivo de um computador local que geralmente é usado para fins de testes e de desenvolvimento, há tutoriais disponíveis na documentação do Azure em vários idiomas. Esses tutoriais estão disponíveis em <https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-get-started-simulated>.

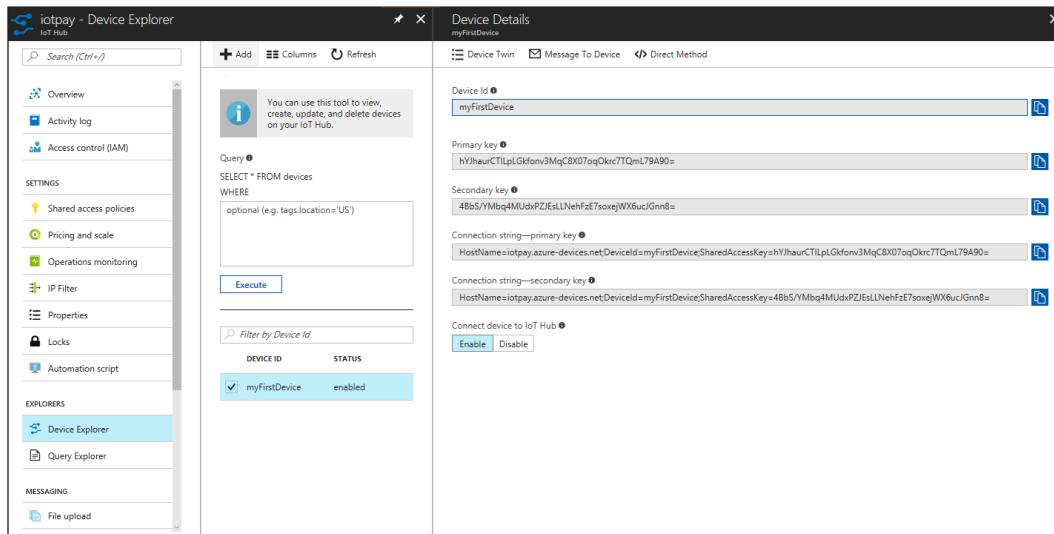


O simulador online do Raspberry Pi está disponível em <https://docs.microsoft.com/azure/iot-hub/iot-hub-raspberry-pi-web-simulator-get-started>, e para dispositivos físicos que precisam ser registrados com o Hub IoT, as etapas mencionadas em <https://docs.microsoft.com/pt-br/azure/iot-hub/iot-hub-get-started-physical> devem ser usadas.

Para adicionar manualmente um dispositivo usando o portal do Azure, o Hub IoT fornece um menu **Gerenciador de Dispositivos**, que pode ser usado para configurar um novo dispositivo.



Após a criação da identidade do dispositivo, uma cadeia de conexão da chave primária para o Hub IoT deve ser usada em cada dispositivo para se conectar a ele:



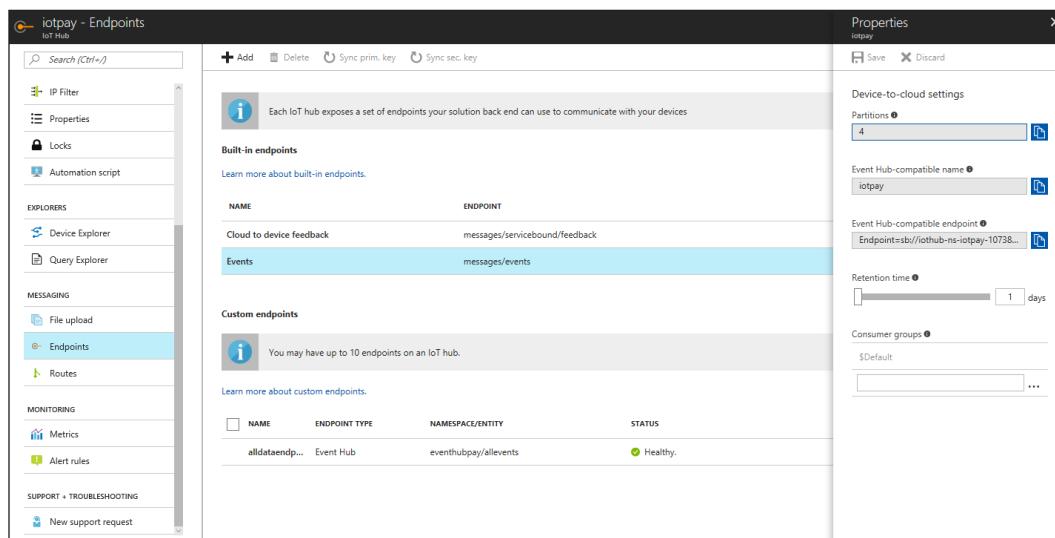
Gerenciamento de mensagens

Depois que os dispositivos forem registrados no Hub IoT, eles poderão começar a interagir com ele. Essa interação pode ser do dispositivo para a nuvem ou da nuvem para o dispositivo.

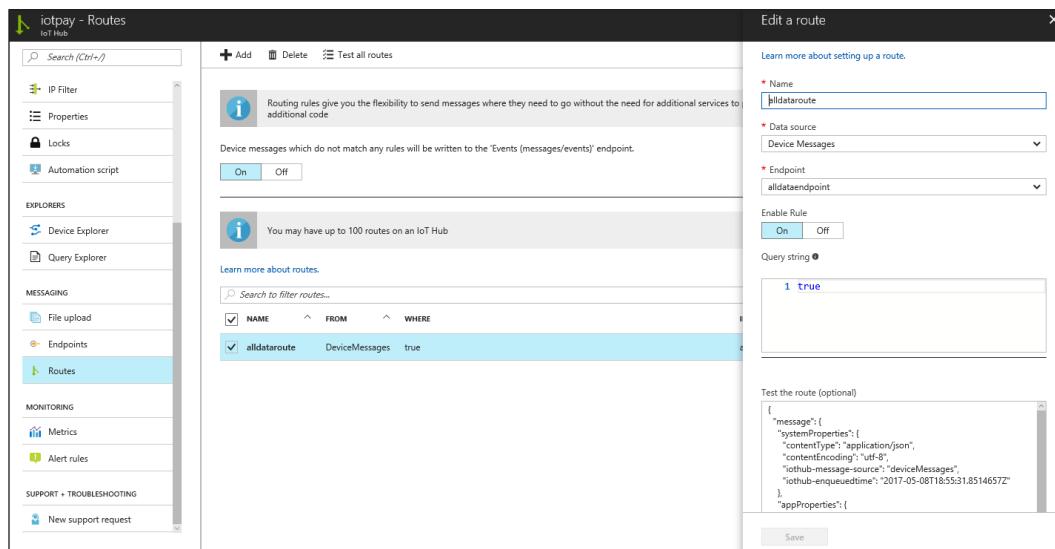
Serviço de mensagens do dispositivo para a nuvem

Uma das melhores práticas que devem ser seguidas nesta comunicação é que, embora o dispositivo possa estar capturando muitas informações, somente os dados de alguma relevância devem ser transmitidos para a nuvem. O tamanho da mensagem é muito importante em soluções IoT devido ao fato que as soluções IoT em geral têm volumes muito elevados de dados. Até mesmo 1 KB de dados extras pode resultar em um gigabyte de armazenamento e processamento desperdiçados. Cada mensagem tem propriedades e cargas; as propriedades definem os metadados para a mensagem. Esses metadados contêm dados sobre o dispositivo, identificação, marcas e outras propriedades que são úteis no roteamento e na identificação das mensagens.

Os dispositivos ou os gateways de nuvem devem se conectar ao Hub IoT para transferir dados. O Hub IoT fornece pontos de extremidade públicos que podem ser utilizados por dispositivos para a conexão e o envio de dados. O Hub IoT deve ser considerado como o primeiro ponto de contato para processamento de back-end. O Hub IoT é capaz de transmitir e rotear essas mensagens para vários serviços. Por padrão, as mensagens são armazenadas em hubs de eventos. Vários hubs de eventos podem ser criados para diferentes tipos de mensagens:



As mensagens podem ser roteadas para diferentes pontos de extremidade com base em propriedades de cabeçalho e de corpo da mensagem, como mostrado na seguinte captura de tela:



As mensagens em um Hub IoT permanecem lá por sete dias por padrão, e seu tamanho pode ir até 256 KB.

Há um simulador de exemplo fornecido pela Microsoft para simular o envio de mensagens para a nuvem. Ele está disponível em vários idiomas; e a versão em C# pode ser visualizada em <https://docs.microsoft.com/pt-br/azure/iot-hub/iot-hub-csharp-csharp-c2d>.

Serviço de mensagens da nuvem para o dispositivo

O Hub IoT do Azure é um serviço gerenciado que fornece uma infraestrutura de serviço de mensagens bidirecional. As mensagens podem ser enviadas da nuvem para dispositivos e com base na mensagem sobre a qual os dispositivos podem agir.

Existem três tipos de padrões de serviço de mensagens da nuvem para o dispositivo:

- Os métodos diretos exigem confirmação imediata de resultados. Com frequência, os métodos diretos são usados para o controle interativo de dispositivos, como a abertura e o fechamento de portões de garagem. Eles seguem o padrão de solicitação-resposta.
- A configuração de propriedades do dispositivo usando a IoT do Azure fornece propriedades do **dispositivo gêmeo**. Por exemplo, você pode definir o intervalo de envio de telemetria para 30 minutos. Os dispositivos gêmeos são documentos JSON que armazenam informações do estado do dispositivo (como metadados, configurações e condições). Um Hub IoT persiste um dispositivo gêmeo para cada dispositivo no Hub IoT.
- Mensagens da nuvem para o dispositivo são usadas para notificações unidirecionais para o aplicativo do dispositivo. Isso segue o padrão disparar e esquecer.

Segurança

A segurança é um aspecto importante dos aplicativos baseados em IoT. Os aplicativos baseados em IoT são compostas por dispositivos que usam a internet pública para conectividade com aplicativos de back-end. A proteção de dispositivos, de aplicações de back-end e de conectividade de usuários mal-intencionados e hackers deve ser considerada uma prioridade para o sucesso dessas aplicações.

Segurança na IoT

Os aplicativos de IoT são construídos principalmente em torno da Internet, e a segurança deve desempenhar um papel importante para garantir que a solução não seja comprometida. Algumas das decisões de segurança mais importantes que afetam a arquitetura da IoT são as seguintes:

- Os dispositivos que usam HTTP versus pontos de extremidade REST HTTPS – os pontos de extremidade REST protegidos por certificados garantem que as mensagens transferidas de um dispositivo para a nuvem e vice-versa sejam bem criptografadas e assinadas. As mensagens não devem fazer sentido para um intruso e devem ser extremamente difíceis de decifrar.
- Se os dispositivos estiverem conectados a um gateway local, o gateway local deverá se conectar à nuvem usando um protocolo HTTP seguro.
- Os dispositivos devem ser registrados no Hub IoT para que possam enviar mensagens.
- As informações passadas para a nuvem devem ser mantidas no armazenamento, que está bem protegido e seguro. Os tokens SAS apropriados ou as cadeias de conexão armazenados no Azure Key Vault devem ser usados para a conexão.
- O Azure Key Vault deve ser usado para armazenar todos os segredos, as senhas e as credenciais, incluindo os certificados.

Escalabilidade

A escalabilidade para o Hub IoT é um pouco diferente de outros serviços. No Hub IoT, existem dois tipos de mensagens:

- **Entrada:** mensagens do dispositivo para a nuvem
- **Saída:** mensagens da nuvem para o dispositivo

Os dois tipos precisam ser contabilizados em termos de escalabilidade.

O Hub IoT fornece um par de opções de configuração durante o tempo de provisionamento para configurar a escalabilidade. Essas opções também estarão disponíveis após o provisionamento e podem ser atualizadas para se ajustarem melhor aos requisitos da solução em termos de escalabilidade.

As opções de escalabilidade disponíveis para o Hub IoT são:

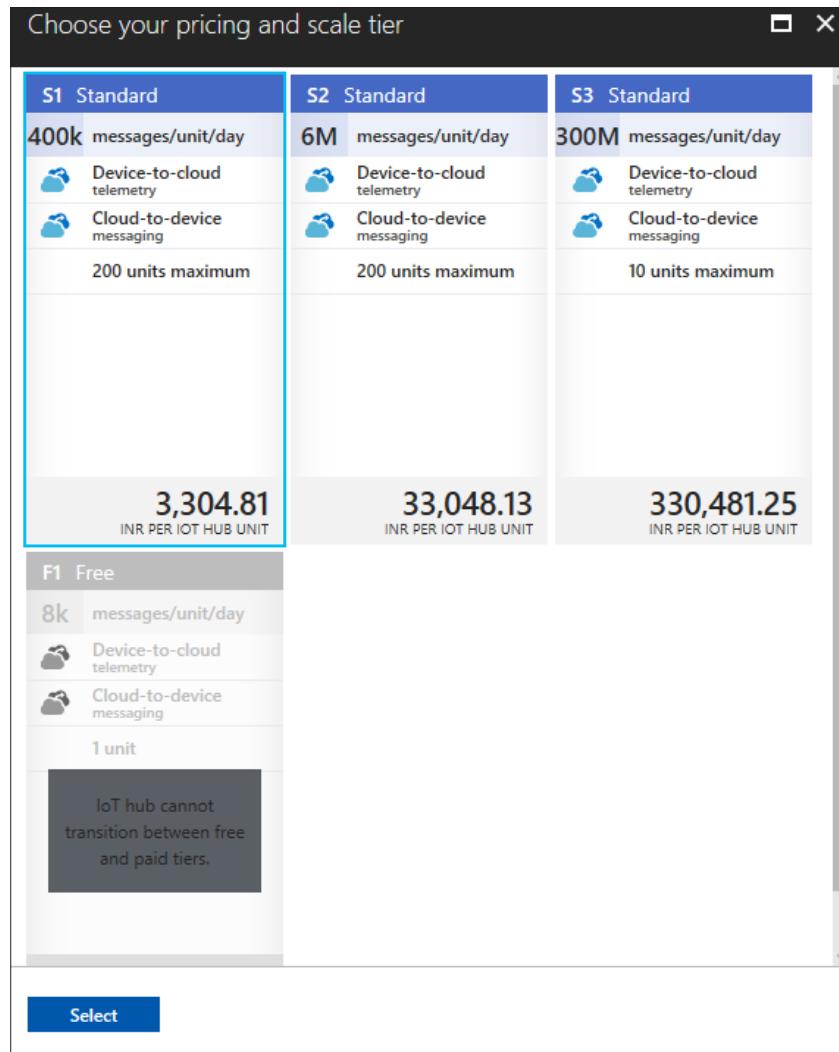
- A edição de **Unidade de Manutenção de Estoque (SKU)**, que é o tamanho do Hub IoT
- Número de unidades

Edição de SKU

O SKU no Hub IoT determina o número de mensagens que um hub pode manipular por unidade por dia, e isso inclui as mensagens de entrada e de saída. Há quatro níveis, como segue:

- **Gratuito:** permite 8000 mensagens por unidade por dia e permite mensagens de entrada e de saída. No máximo, uma unidade pode ser provisionada. Essa edição é apropriada para ganhar familiaridade e testar os recursos do serviço do Hub IoT.
- **Standard (S1):** permite 400.000 mensagens por unidade por dia e permite mensagens de entrada e de saída. No máximo, 200 unidades podem ser provisionadas. Essa edição é apropriada para um número pequeno de mensagens.
- **Standard (S2):** permite seis milhões de mensagens por unidade por dia e permite mensagens de entrada e de saída. No máximo, 200 unidades podem ser provisionadas. Essa edição é apropriada para um número grande de mensagens.

- **Standard (S3):** permite 300 milhões de mensagens por unidade por dia e permite mensagens de entrada e de saída. No máximo, 10 unidades podem ser provisionadas. Essa edição é apropriada para um número muito grande de mensagens:

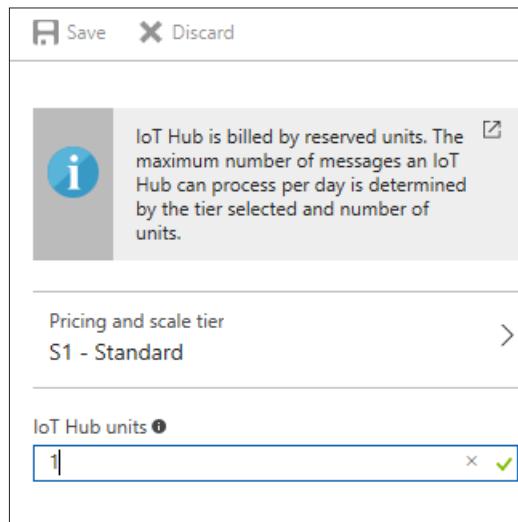


Um leitor astuto teria notado que o nível Standard S3 permite um máximo de 10 unidades só comparado a outras unidades padrão que permitem 200 unidades. Isso está diretamente relacionado ao tamanho dos recursos de computação provisionados para executar serviços de IoT. O tamanho e a capacidade de máquinas virtuais para o Standard S3 são significativamente maior em comparação a outros níveis onde o tamanho permanece o mesmo.

Unidades

As unidades definem o número de instâncias de cada SKU em execução por trás do serviço. Por exemplo, duas unidades do nível de SKU Standard S1 significarão que o Hub IoT é capaz de lidar com $400\text{ K} * 2 = 800\text{ K}$ de mensagens por dia.

Mais unidades aumentarão a escalabilidade do aplicativo:



Alta disponibilidade

O Hub IoT é uma **Plataforma como um Serviço (PaaS)** do Azure. Os clientes e os usuários não interagem diretamente com o número e o tamanho subjacentes de máquinas virtuais nas quais o serviço do Hub IoT está em execução. Os usuários decidem a região, o SKU do Hub IoT e o número de unidades para seu aplicativo. O restante da configuração é determinado e executado pelo Azure nos bastidores. O Azure garante que cada serviço PaaS seja altamente disponível por padrão. Ele faz isso ao garantir que várias máquinas virtuais provisionadas por trás do serviço estejam em racks separados no data center. Ele faz isso colocando as máquinas virtuais em um conjunto de disponibilidade e em domínios de falha e de atualização separados. Isso ajuda na alta disponibilidade para a manutenção planejada e não planejada. Os conjuntos de disponibilidade cuidam da alta disponibilidade no nível do data center.

Resumo

O IoT é uma das maiores tecnologias futuras desta década e já está transformando as indústrias. Coisas que pareciam impossíveis antes agora são repentinamente possíveis. O Hub IoT é uma plataforma que facilita a criação e a entrega de soluções IoT para o cliente de uma maneira mais rápida, melhor e mais barata. Ele fornece implementações de todos os protocolos de indústria, como MQTT e AMQP, juntamente com gateways de campo que podem adaptar dispositivos não padrão. Ele fornece recursos de alta disponibilidade, escalabilidade e segurança para mensagens e soluções em geral. Fornece conectividade para um grande número de serviços do Azure e ajuda no roteamento de mensagens para vários pontos de extremidade diferentes, cada um com capacidade para processamento de mensagens. A IoT pode rastrear o ciclo de vida de desenvolvimento inteiro e ajudar a agilizar o tempo de lançamento no mercado para as empresas.

O Azure oferece inúmeras opções de armazenamento de dados e serviços. Este é o último capítulo do livro. Espero que você tenha realmente gostado de ler todos os capítulos e esteja pronto para arquitetar soluções no Azure.

Obrigado!

Outros livros de que você pode gostar

Se você gostou deste livro, você pode se interessar por estes outros livros da Packt:



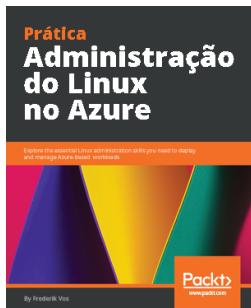
Hands-On Azure for Developers (Treinamento prático do Azure para Desenvolvedores)

KamilMrzygłód

ISBN: 978-1-78934-062-4

- ▶ Implementar componentes sem servidor, como funções e aplicativos lógicos do Azure
- ▶ Integrar aplicativos com armazenamentos e contêineres disponíveis
- ▶ Entender componentes de mensagens, incluindo Hubs de Eventos do Azure e Armazenamento de Filas do Azure
- ▶ Obter uma compreensão do Application Insights e de outras soluções de monitoramento adequadas
- ▶ Armazenar seus dados com serviços como o Azure SQL e o Azure Data Lake Storage
- ▶ Desenvolver aplicativos de nuvem rápidos e escalonáveis

Outro livro de que você pode gostar —————



Hands-On Linux Administration on Azure (Administração prática do Linux no Azure)

Frederik Vos

ISBN: 978-1-78913-096-6

- ▶ Entender por que o Azure é a solução ideal para seus workloads de open source
- ▶ Dominar habilidades essenciais do Linux e aprender a descobrir o seu caminho no ambiente Linux
- ▶ Implantar o Linux em um ambiente do Azure
- ▶ Usar o gerenciamento de configuração para gerenciar o Linux no Azure
- ▶ Gerenciar contêineres em um ambiente do Azure
- ▶ Melhorar a segurança do Linux e usar os sistemas de gerenciamento de identidade do Azure
- ▶ Automatizar a implantação com o Azure Resource Manager (ARM) e o PowerShell
- ▶ Usar o Ansible para gerenciar instâncias do Linux em um ambiente de nuvem do Azure

Deixe um comentário – deixe outros leitores saberem o que você pensa

Compartilhe suas opiniões sobre este livro com outras pessoas deixando um comentário no site que você comprou. Se você comprou o livro na Amazon, deixe um comentário honesto na página do Amazon sobre este livro. Isso é essencial para que outros possíveis leitores possam ler e usar sua opinião imparcial para tomar decisões de compra, para que possamos entender o que nossos clientes pensam sobre nossos produtos e para que nossos autores possam ver seus comentários sobre o livro no qual eles têm trabalhado junto com a Packt. Levará apenas alguns minutos do seu tempo, mas será valioso para outros possíveis clientes, para nossos autores e para a Packt. Obrigado!

