Teil 1: Einführung und Nutzen von Azure DevTest Labs

Ein DevOps-Testtraum

Wäre es nicht ein Traum, seine Entwicklungsumgebungen selbst hoch- und runterzufahren, und dabei selbst zu bestimmen, was auf der Maschine läuft? Müsste es nicht herrlich sein, wenn die ganzen Maschinen individuell nach einer standardisierten Formel erstellt werden würden? Noch viel besser wäre es doch, wenn der Manager dies alles mit voller Kostenkontrolle den Entwicklern frei zur Verfügung stellen könnte, oder? Ein Microsoft Azure Service hat die Antwort.

von Sebastian Schütze und Tyrone Guiamo

All diese Szenarien und noch viel mehr sind möglich. Dafür hat Microsoft einen Azure Service aufgefahren, der sich Azure DevTest Labs [1] nennt. Azure DevTest Labs existiert bereits seit über zwei Jahren und wurde öffentlich im Mai 2016 angekündigt.

Dieser Service ist Microsofts Art für Entwickler, das Leben leichter zu machen. Nebenbei gewinnen Firmen dringend benötigte Agilität und Automatisierung in einer standardisierten Lösung, die auch unkompliziert zu managen ist und deren Kosten unter Kontrolle gehalten werden können. Kontrolle wird hier durch Benachrichtigungen und Alarme sichergestellt. Die DevTest Labs stellen eine Win-win-Situation für drei verschiedene Stakeholder in einem Unternehmen dar:

- **IT-Manager und/oder Projektmanager**: Pro Projekt oder Team kann ein Limit gesetzt werden, wie viel Kosten pro Monat oder für eine festgesetzte Zeit maximal aufkommen dürfen. Grenzwerte können erstellt werden, deren Überschreiten einen Alarm auslösen.
- **Entwickler/Test/Trainer**: Sie haben eine Testumgebung, in der sie VMs in Self-Service-Manier an und ausschalten können. Sie können sich ihre Maschinen

Artikelserie

Teil 1: Einführung und Nutzen von Azure DevTest Labs
Teil 2: Das eigene DevTest-Labor im Detail

inklusive Software in Baukastenformat zusammenstellen, wobei vom Unternehmen die Rahmenbedingungen festgelegt werden.
- **IT-Administratoren bzw. Ops:** Kurz: weniger Arbeit! Sie können sich auf Policies konzentrieren und sich anderen Aufgaben widmen, als neue Entwicklungs- oder Testmaschinen hochfahren oder installieren zu müssen. Zusätzlich kann die Anzahl der Maschinen pro Team oder pro Person festgelegt werden, die erstellt werden dürfen. Sie können dedizierten Zugriff zum Lab gewähren, ohne dabei andere Ressourcen in Azure freigeben zu müssen.

Da jedoch die Grenze zwischen Entwickler und IT-Ops immer durchlässiger wird, könnten die letzten beiden Stakeholder auch einfach DevOps genannt werden. Denn sie arbeiten heute enger zusammen und bauen gemeinsam ihr DevTest Lab auf.

Sicherheit in Azure DevTest Labs

Von Hause aus werden drei verschiedene Rollentypen angeboten, denen Benutzer zugeordnet werden können: Owner (Besitzer), Contributer (Mitwirkende) und DevTest-Labs-User. Zusätzlich gibt es aber auch benutzerdefinierte Rollen. Der Unterschied zwischen Owner und Contributor ist minimal. Dem Contributor ist es lediglich verwehrt, Berechtigungen an andere Benutzer zu vergeben. Ansonsten sind sie identisch. Tabelle 1 gibt einen Überblick über die Rollenunterschiede.

Wir haben eine Rolle verschwiegen: Reader (Lesende). Sie kann jedoch vernachlässigt werden, da damit lediglich nur betrachtet werden kann, was in dem DevTest Lab existiert. Ein wichtiger Punkt ist zudem zu beachten: DevTest-Labs-User werden automatisch als Owner der VMs festgelegt. Dies bezieht sich jedoch nur auf die VM und nicht das Lab selbst.

Benutzerdefinierte Rollen können nur mit PowerShell-Skripten erstellt werden. Diese sind zum Beispiel nützlich, um externen Benutzern Zugang zu ihrem DevTest Lab zu geben. Benutzer müssen bereits ihrem Azure Active Directory hinzugefügt sein (Azure B2B), damit sie diesem Benutzer eine Rolle zuweisen können.

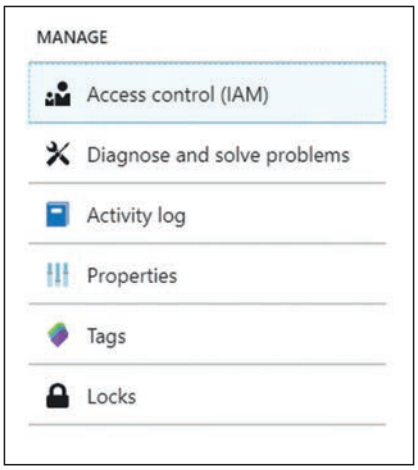

Abb. 1: Verwaltungseinstellungen des DevTest Labs

Benutzer und Rollen können im Bereich „Konfiguration und Richtlinien“ (engl. Configuration and Policies) und dann weiter im Bereich „Access Control (IAM)“ in der Kategorie „Verwalten“ zugewiesen und verwaltet werden (**Abb. 1**).

Azure-DevTest-Labs-Features

Mit Azure DevTest wird dem Benutzer eine agile Umgebung an die Hand gegeben, um Azure-Ressourcen in Self-Service-Prozessen zu verteilen und dabei gleichzeitig volle Kontrolle über Kosten zu haben. Zudem existieren viele Images, die über den Marketplace verwendet werden können und jeden Geschmack abdecken. Dies geht von Linux bis hin zu jeder erdenklichen Form von Windows mit extra vorinstallierter Software oder ohne. Jedoch möchte man als Unternehmen nicht unbedingt jede Software zulassen, sondern das Ganze organisiert

Aktionen	DevTest-Labs-Benutzer	Besitzer	Mitwirkender
Lab-Aufgaben			
Benutzer zu einem Lab hinzufügen	x	o	x
Kosteneinstellungen aktualisieren	x	o	o
Grundlegende Aufgaben für virtuelle Computer			
Benutzerdefinierte Images hinzufügen und entfernen	x	o	o
Formeln hinzufügen, aktualisieren und löschen	o	o	o
Azure Marketplace Images in eine Positivliste aufnehmen	x	o	o
Aufgaben für virtuelle Computer			
Virtuelle Computer erstellen	o	o	o
Virtuelle Computer starten, beenden und löschen	c	o	o
VM-Richtlinien aktualisieren	x	o	o
Datenträger zu virtuellen Computern hinzufügen bzw. davon entfernen	c	o	o
Artefaktaufgaben			
Artefakt-Repositories hinzufügen und entfernen	x	o	o
Artefakte anwenden	o	o	o
Legende: o = ja, x = nein, c = nur vom Benutzer erstellte VMs			

Tabelle 1: Rollen, Benutzer und deren erlaubte Aktionen in DevTest Labs

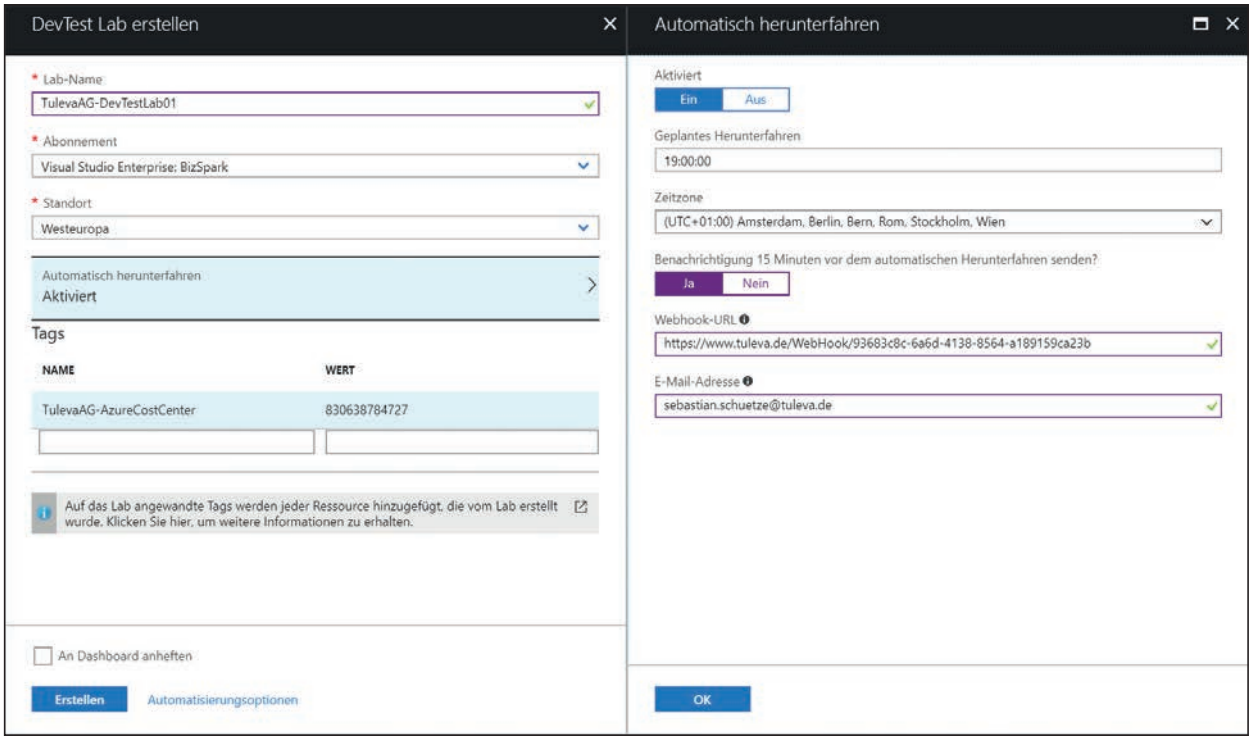

Abb. 2: Einzugebende Parameter bei der Erstellung eines DevTest Labs

ablaufen lassen. Dies ist durch „White Lists“ möglich, mit denen die Images festgelegt werden, die eingesetzt werden dürfen. Somit kann in einem Lab kontrolliert werden, ob z. B. nur Ubuntu-Server verwendet werden sollen statt beispielsweise Red Hat Enterprise.

Für die Kostenkontrolle kann ein fester Wert für eine Periode von einem Monat eingestellt werden. Für dieses Kostenziel können dann Schwellwerte eingestellt werden, die Benachrichtigungen auslösen. Zum Beispiel können Nachrichten versendet werden, wenn 50 Prozent, 100 Prozent oder sogar 125 Prozent erreicht wurden. Es ist möglich, sich die Kosten in verschiedenen Grafiken darstellen zu lassen.

Eine weitere Funktion stellt das automatische Herunterfahren und Hochfahren der Maschinen dar. Dadurch müssen einzelne Mitarbeiter nicht immer wieder daran erinnert werden, die Maschinen über Nacht oder über das Wochenende auszuschalten. Zusätzlich können 30 Minuten vor dem Herunterfahren Nachrichten an eine zentrale E-Mail-Adresse versendet werden. Diese warnt vor dem Herunterfahren und gibt dem Benutzer die Möglichkeit, dies um eine bis zwei Stunden zu verzögern. Es ist sogar möglich, das Herunterfahren einmalig zu stoppen, falls notwendig. Die Benachrichtigungs- und Ausschalteneinstellung ist entweder zentral für das gesamte Lab oder für einzelne Maschinen möglich.

Beachten Sie, dass die Nutzung des Labs selbst kostenlos ist. Es entstehen lediglich Kosten für die erstellten Ressourcen. Somit können auch mehrere Labs in einem Abonnement erstellt und für verschiedene Zwecke genutzt werden.

Der Spaß für Entwickler fängt jedoch bei der Möglichkeit an, ein eigenes Repository z. B. über VSTS oder GitHub einzubinden und damit eigene Ressourcen. Diese Ressourcen können dann als eigene Artefakte für die Zusammenstellung einer VM genutzt werden.

Aber was sind Artefakte?

Artefakte repräsentieren zusätzliche Softwarebausteine, die direkt installiert werden können, nachdem die VM provisioniert wurde. Das können Applikationen oder einfache Aktionen sein. Diese Artefakte sind in einem eigens definierten Repository gelagert und werden bei Bedarf verwendet. Es können eigene Artefakte für das Lab erstellt werden. Diese Artefakte bestehen im Prinzip aus Skripten und einem Manifest. Von daher sind den Möglichkeiten kaum Grenzen gesetzt. Je nach Betriebssystem werden außerdem auch Out-of-the-box-Artefakte wie z. B. 7-Zip, Notepad++, Visual Studio, VS Code usw. angeboten. Zudem können es auch einfach Skripte wie das Einbinden in das eigene Active Directory (AD) sein, das die Maschine direkt in die eigene Domain einbindet. Dieses Artefakt würde dann in einer definierten Maske lediglich einige Parameter wie Benutzer, Domäne und das in einer Azure Key Vault sicher gespeicherte Passwort abfragen. Bereits von Microsoft bereitgestellte und in Azure integriert Artefakte können auf der Entwicklungsplattform GitHub unter [2] eingesehen werden (weitere Informationen zum DevLab finden sich bei Microsoft unter [3] und [4]).

Benutzerdefinierte Images und Formeln

Bei der Bereitstellung von VMs gibt es grundsätzlich drei verschiedene Arten und Weisen, aus denen ausgewählt werden kann: Marketplace Images, benutzerdefinierte Images und Formeln.

Marketplace Images: Hier können per Standardeinstellung alle Images gewählt werden, die auch für normale VMs zur Verfügung stehen. Wurde eine Whitelist definiert, dann sind nur diese speziell ausgewählten Images verfügbar. Es gibt eine große Anzahl von Images zur Auswahl, die von Zeit zu Zeit aktualisiert werden.

Custom Images: Sie haben beispielsweise eine VM, die Sie in Azure oder On-Premises für Ihren Bedarf bereits fertiggestellt haben und wollen sie im Lab verwenden. Dieses Custom-Image kann als VHD-Datei gespeichert und dann zum Lab (z. B. mit PowerShell) hochgeladen werden. Es steht dann für zukünftige VMs zur Verfügung. Der große Vorteil der benutzerdefinierten Images ist, dass sie zwischen Labs und sogar Labs in anderen Abonnements geteilt werden können.

Formeln (wiederverwendbare Basen): Formeln stellen eine dynamische Art und Weise der Erstellung von VMs dar. Formeln und benutzerdefinierte Images haben einiges gemeinsam, aber beide ihre Vor- und Nachteile. Formeln erlauben genauso wie selbsterstellte Images die Erstellung mit einem Basis-Image aus einer VHD-Datei. Dieses Image kann dann wiederum verwendet werden, um eine neue VM zu erstellen. Ein benutzerdefiniertes Image hat jedoch den Nachteil, dass es, im Gegensatz zur Formel, statisch ist. Die Formel erlaubt in einem Schritt auch die Verwendung von Artefakten, die mit der Erstellung der Maschine mitinstalliert werden sollen. Zudem können in den Formeln auch gleich die Leistung der VM, das virtuelle Netzwerk (oder Subnet), in dem das Image erstellt wird, mit festgelegt werden. Somit sind Formeln eine Möglichkeit der Standardisierung. Parameter können im Bedarfsfall bei der konkreten Erstellung jedoch auch überschrieben und angepasst werden.

Und wie erstellt man ein DevTest Lab?

Zur Einführung wird das DevTest Lab über die Portalwebsite erstellt. Die Erstellung kann ebenfalls, wie alles in Azure, über PowerShell geschehen, was Entwicklern mehr Flexibilität an die Hand gibt. Zum Erstellen muss unter Ressource erstellen am besten nach DevTest Lab gesucht und dieses ausgewählt werden. Folgende Parameter werden bei der Erstellung benötigt (**Abb. 2**):

- **Lab-Name**: Der Name des Labs in alphanumerischen Zeichen.
- **Abonnement**: Das Abonnement, unter dem die Kosten verrechnet werden sollen.
- **Location**: In welchem Datenzentrum das Lab erstellt werden soll. In dieser Region werden auch die darunterliegenden VMs angelegt.

- **Automatisch herunterfahren**: Falls Sie wollen, dass alle VMs zur selben Zeit heruntergefahren werden sollen. Dies kann auch später festgelegt werden.
- **Tags**: Sie sind zur eigenen Einordnung gedacht, um Ressourcen leichter zu finden. Diese Tags werden jeder darunterliegenden Ressource hinzugefügt.

Wenn alle Optionen ausgefüllt sind, dann kann das Lab erstellt werden. Die Erstellung kann einige Minuten dauern.

Die folgenden Azure-Ressourcen werden mit einem Lab automatisch erstellt:

- **Ressourcengruppe** (**Resource Group**): Sie enthält alle Ressourcen, die beim Provisionieren des Labs erstellt werden. Der Name der Gruppe ist selbst vergeben und wird durch eine zufällig erstellte Zeichenkette ergänzt.
- Das **DevTest Lab** selbst mit dem am Anfang vergebenen Namen.
- **Speicherkonto** (**Storage Account**): Wird wieder mit einem automatisch generierten Namen erstellt.
- **Virtual Network**: Ebenfalls mit einem automatisch generierten Namen. Zu bemerken wäre, dass man jedes bereits existierende virtuelle Netzwerk nach dem Erstellen hinzufügen kann. Das erstellte Netzwerk muss nicht verwendet werden. Hier kann ein Subnet ausgewählt werden, in dem bestimmt wird, ob die VMs dort automatisch provisioniert werden, wie viel VMs je Benutzer darin erstellt werden dürfen und ob öffentliche IPs erlaubt sind.
- **Schlüsseltresor** (**Key Vault**): Besitzt ebenfalls einen automatisch generierten Namen. Dieser kann dazu verwendet werden, sicher verwahrte Zugangsdaten bei der Erstellung einer VM zuzuweisen, mit dem sich der Benutzer einloggen soll. Somit müssen Passwörter nicht weitergegeben werden, wenn die Ressourcen von einem Dritten provisioniert werden sollen (egal ob Benutzer oder Skript).

Über die Namen der Ressourcen besteht bei der Erstellung über das Portal keine Kontrolle. Wenn dies kontrolliert werden soll, dann muss es mit PowerShell durchgeführt werden.

Wofür ist DevTest Labs nicht da?

In diesem Fall erklärt der Name schon selbst, dass dieser Azure-Service nicht für produktive Umgebungen verwendet werden soll! Diese Umgebung ist für Entwickler und Tester, die eine höhere Flexibilität erhalten sollen, wenn es dazu kommt, Umgebungen zu verteilen, hoch- und runterzufahren und sie wieder zu beenden. Das Lab ist ebenfalls für temporäre Umgebungen gedacht. Das heißt, selbst wenn ein Build Agent oder Release Agent Teil der Entwicklungsplattform ist, sollte es bitte nicht im Lab erstellt werden. Das Lab kann ebenfalls zum Hochfahren von Trainingsumgebungen verwendet werden. Es kann ein Ablaufdatum für jede VM festgelegt werden bzw. es können diese in Reihe nacheinander hochgefahren werden. Dadurch werden VMs automatisch gelöscht, wenn z. B. das Training vorbei ist. Wie an den Möglichkeiten zu sehen, ist ein Lab also nicht für Qualitätssicherung (Acceptance) oder Produktion vorgesehen. In der produktiven Welt sollten Entwickler und Tester niemals Kontrolle über Umgebungen der Qualitätssicherung oder Produktion erhalten. Diese sind allein den Administratoren vorbehalten.

Zusammenfassung

Azure DevTest Labs bieten einen flexiblen und hoch anpassbaren Weg, um eine Reihe verschiedener Umgebungen anzulegen. Dazu zählen z. B. Softwareentwicklung, Test und Trainings. Zudem kann nicht nur ein, sondern es können beliebig viele DevTest Labs für verschiedene Anwendungsfälle und Teams erstellt werden. Außerdem können verschiedene Images und eigene Repositories für eigene Artefakte integriert werden, und Formeln bieten die Möglichkeit einer flexiblen Automatisierung und Standardisierung. Kosten können leicht verfolgt bzw. kontrolliert werden, und unterschiedlichste Rollen und Berechtigungen geben allen Beteiligten den passenden Zugriff, um das DevTest Lab korrekt verwenden zu können. Im nächsten Artikel wollen wir dann ans Eingemachte gehen und erklären, wie man eigene Artefakte erstellt, Formeln definiert und tiefer in die Einrichtung und Erstellung des Labs einsteigen kann.

Sebastian Schütze ist seit zehn Jahren Webentwickler, arbeitet seit fünf Jahren mit hybriden SharePoint-Plattformen und hat im Zuge der Entwicklung von Azure die Möglichkeiten der DevOps-Technologien für sich gefunden. Er arbeitet aktuell als Senior Azure/DevOps Consultant bei der Tuleva AG.

http://www.tuleva.de **@RazorSPoint**

sebastian.schuetze@tuleva.de

Tyrone Guiamo arbeitet als Microsoft Cloud Solution Architect bei Schindler und ist ebenfalls Koorganisator des monatlichen Azure Meetups in Berlin.

tyrone.guiamo@schindler.com

Links & Literatur

[1] https://azure.microsoft.com/de-de/services/devtest-lab/

[2] Öffentliches Azure DevTest Lab GitHub Repository: https://github.com/Azure/azure-devtestlab

[3] Neuigkeiten, Updates und Ankündigungen bezogen auf das DevTest Lab: https://blogs.msdn.microsoft.com/devtestlab

[4] Die offizielle Azure-DevTest-Lab-Dokumentation: https://docs.microsoft.com/en-us/azure/devtest-lab/