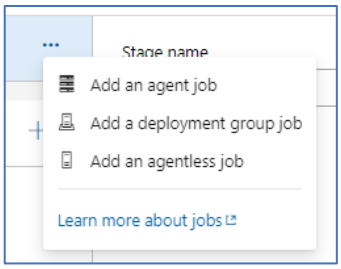
Dit is een interventie

Met een beetje creativiteit het onmogelijke mogelijk maken binnen een Azure DevOps pipeline

Het is voorgekomen dat een outbound IP adres van een Azure appservice is aangepast zonder dat we daar erg in hadden. Daardoor had de appservice geen toegang tot een andere resource die met IP restricties afgeschermd was.

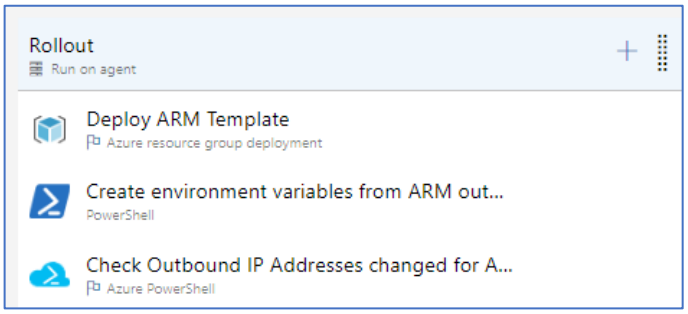


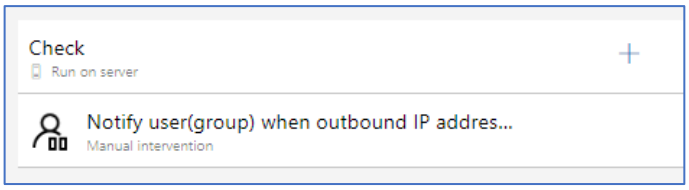
Om daar mogelijk iets eerder van op de hoogte te zijn, controleren we op aanpassingen in de outbound IP adressen tijdens het uitrollen van de applicatie.

De implementatie

Een release pipeline, bestaande uit twee fasen:

1. Een **Agent job** die de uitrol doet (onderstaand plaatje bevat alleen de stappen die relevant zijn voor dit artikel)
2. Een **Agentless job** die een notificatie verstuurd, wanneer de IP adressen aangepast zijn.

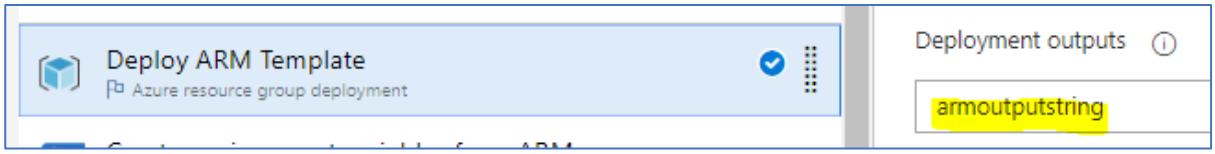




Deploy ARM template

Uit deze stap komen de outbound IP adressen van de appservice voort.

```
"outboundIpAddresses": {
      "type": "array",
      "value": "[split(reference(resourceId('Microsoft.Web/sites', variables('webApplicationSettings').name), '2018-11-01').possibleOutboundIpAddresses, ',')]" }
```



Create environment variable from ARM output

Deze PowerShell task vormt de ARM output om tot bruikbare variabelen.

```
$json = $(armoutputstring) | convertfrom-json
$ips= ($json.outboundIpAddresses.value | ConvertTo-Json -Compress)
Write-Host "##vso[task.setvariable variable=armoutboundips;]$ips"
```

Check outbound IP addresses changed

Deze Azure PowerShell task controleert of de outbound IP adressen -vanuit de ARM output- gelijk is aan de IP adressen in de SQL firewall rules*. Dit is het script in essentie, uiteraard zijn er betere vergelijkingsmethoden denkbaar.

```
$resourceGroupName = "$(rsrcg_name)"

$firewallrules = (Get-AzSqlServerFirewallRule  `
                    -ServerName $(ARMSQL)  `
                    -ResourceGroupName $resourceGroupName  `
                    -WarningAction SilentlyContinue `
                    | Where-
Object {$_.FirewallRuleName.StartsWith("$(firewallRulePrefix)")}).StartIpAddress -join ','

$outboundips =(Get-AzWebApp -ResourceGroup $resourceGroupName -
name $(ARMAPP)).PossibleOutboundIpAddresses

$different= $outboundips -ne $firewallrules
```

Op dit moment weten we of er een verschil is en of we dus wellicht nog andere acties moeten ondernemen nadat de uitrol is voltooid. Maar wat is de beste manier om dit te uiten?

We kunnen de release laten falen en een foutmelding tonen, maar dat is ietwat rigoureus. Het liefst willen een signaal ontvangen waar we naar eigen inzicht op kunnen reageren. We kiezen daarom om een **Manual Intervention** task aan het einde van de pipeline toe te voegen.

Een dergelijke task is alleen toe te voegen aan een Agentless Job. We moeten dus een extra phase toevoegen, van het type Agentless Job. Dat schept een uitdaging.

We willen alleen een notificatie wanneer de IP adressen verschillen. Onder **Control Options** (bij een task) of **Additional Options** (bij een phase) kun je een expressie definiëren waarmee je de betreffende task of phase conditioneel kunt laten uitvoeren. Echter zijn (custom en predefined) variabelen niet zonder meer overdraagbaar tussen phases (jobs)**.

Je zult opmerken dat deze expressie altijd False oplevert, omdat de Jobstatus variabele null is:

```
eq(variables['Agent.JobStatus'], 'Succeeded')
```

Terwijl de job status check functie gelukkig (en verrassend genoeg) wél het gewenste resultaat oplevert:

```
succeeded()
```

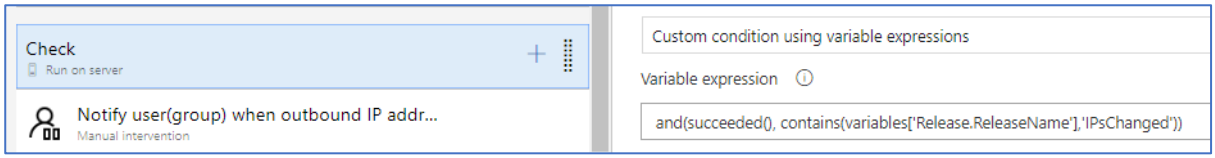
Ook de variabelen die we zelf definiëren in de ‘Check outbound IP addresses changed’ task, zijn niet beschikbaar. Ongeacht of je ze als output kenmerkt.

Wat wél beschikbaar is binnen elke job is release informatie. Dus de truc die bedacht is, is als volgt:

Wanneer de IP adressen blijken te verschillen voegen we een suffix toe aan de releasenaam.

```
Write-Host "##vso[release.updatereleasename]$(Release.ReleaseName)-IPsChanged"
```

In de conditie expressie van de Agentless Job phase kunnen we controleren of de releasenaam het suffix bevat. Indien dit het geval is, zal een notificatie worden uitgestuurd naar de aangewezen gebruiker(sgroep).



* Een dergelijke pipeline zou ook losstaand op een tijdschema kunnen draaien in plaats van tijdens een uitrol. In dat geval lees je de outbound IP adressen uit bij de appservice in plaats van dat je de ARM output variabele gebruikt.

** Het is wel mogelijk om variabelen door te geven naar een volgende job. Bijvoorbeeld door gebruik te maken van YAML. Of de waarde van de variabele op release niveau te beïnvloeden m.b.v. de Azure DevOps REST api in een PowerShell task. In dit geval gebruiken we geen YAML