# Step 3. Create Auto pool with hardcoded authentication

- task: AzureCLI@2

displayName: 'Create auto Pool'

inputs:

azureSubscription: 'CCDC-DEV-UAT-SPN'

scriptType: 'ps'

scriptLocation: 'inlineScript'

addSpnToEnvironment: false # We'll use hardcoded values

inlineScript: |

Write-Host "Setting up authentication with hardcoded values..."

# Clear any existing Azure CLI context

az account clear

# Get the service principal details from the service connection

# Since we can't access the secret directly, we'll use the environment variables

# that Azure DevOps sets when using AzureCLI@2 task

# The AzureCLI@2 task should already be authenticated, but let's ensure we're in the right context

$currentAccount = az account show --query name -o tsv

Write-Host "Current account: $currentAccount"

# Login to the batch account directly (the AzureCLI task should have already authenticated)

Write-Host "Logging into Batch account..."

az batch account login `

--name lvbatchdev `

--resource-group lv-batch-dev

if ($LASTEXITCODE -ne 0) {

Write-Error "Failed to login to batch account"

# Try alternate approach - get the subscription ID and set it explicitly

$subscriptionId = "8d9342e2-96c8-436c-854a-1ad509c7356c"

az account set --subscription $subscriptionId

# Try login again

az batch account login `

--name lvbatchdev `

--resource-group lv-batch-dev

if ($LASTEXITCODE -ne 0) {

Write-Error "Still failed to login to batch account"

exit 1

}

}

Write-Host "Successfully logged in to batch account"

# Job configuration with auto-pool

$timestamp = Get-Date -Format "yyyyMMddHHmmss"

$jobId = "pythonApp-$timestamp"

Write-Host "Creating job with ID: $jobId"

# Create job with auto-pool specification

$jobJson = @"

{

"id": "$jobId",

"poolInfo": {

"autoPoolSpecification": {

"autoPoolIdPrefix": "autopool",

"poolLifetimeOption": "job",

"keepAlive": false,

"pool": {

"vmSize": "Standard\_D2\_v2",

"virtualMachineConfiguration": {

"imageReference": {

"publisher": "canonical",

"offer": "0001-com-ubuntu-server-focal",

"sku": "20\_04-lts",

"version": "latest"

},

"nodeAgentSkuId": "batch.node.ubuntu 20.04"

},

"targetDedicatedNodes": 1

}

}

}

}

"@

# Save JSON to temporary file

$tempFile = [System.IO.Path]::GetTempFileName()

$jobJson | Out-File -FilePath $tempFile -Encoding UTF8

Write-Host "JSON file created at: $tempFile"

Write-Host "JSON content:"

Get-Content $tempFile

# Create job with auto-pool

Write-Host "Creating batch job..."

$output = az batch job create --json-file $tempFile 2>&1

$result = $LASTEXITCODE

Write-Host "Command output: $output"

# Clean up temp file

Remove-Item $tempFile -Force

if ($result -ne 0) {

Write-Error "Job creation with auto-pool failed with exit code: $result"

Write-Error "Output: $output"

exit 1

}

Write-Host "Job $jobId created successfully with auto-pool”

Method 2:

# Step 3. Create Auto pool using regular script task

- task: PowerShell@2

displayName: 'Create auto Pool'

inputs:

targetType: 'inline'

script: |

# Install Azure CLI if needed

$azCmd = Get-Command az -ErrorAction SilentlyContinue

if (-not $azCmd) {

Write-Host "Installing Azure CLI..."

Invoke-WebRequest -Uri https://aka.ms/installazurecliwindows -OutFile .\AzureCLI.msi

Start-Process msiexec.exe -Wait -ArgumentList '/I AzureCLI.msi /quiet'

$env:Path += ";C:\Program Files (x86)\Microsoft SDKs\Azure\CLI2\wbin"

}

# Since we're in a PowerShell task, we need to authenticate manually

# The service connection ID from your screenshot

$serviceConnectionId = "2219ee47-f666-4c64-be85-cdad385c9661"

# Get the service endpoint details from Azure DevOps

$serviceEndpoint = Get-VstsEndpoint -Name "CCDC-DEV-UAT-SPN" -Require

if ($serviceEndpoint) {

$tenantId = $serviceEndpoint.Auth.Parameters.TenantId

$clientId = $serviceEndpoint.Auth.Parameters.ServicePrincipalId

$clientSecret = $serviceEndpoint.Auth.Parameters.ServicePrincipalKey

# Login to Azure

az login --service-principal -u $clientId -p $clientSecret --tenant $tenantId

} else {

# Fallback: Use environment variables if available

if ($env:servicePrincipalId -and $env:servicePrincipalKey -and $env:tenantId) {

az login --service-principal -u $env:servicePrincipalId -p $env:servicePrincipalKey --tenant $env:tenantId

} else {

Write-Error "Cannot authenticate - no service principal credentials available"

exit 1

}

}

# Set subscription

az account set --subscription "8d9342e2-96c8-436c-854a-1ad509c7356c"

# Login to batch account

az batch account login --name lvbatchdev --resource-group lv-batch-dev

# Create job

$jobId = "pythonApp-" + (Get-Date -Format "yyyyMMddHHmmss")

$jobJson = @"

{

"id": "$jobId",

"poolInfo": {

"autoPoolSpecification": {

"autoPoolIdPrefix": "autopool",

"poolLifetimeOption": "job",

"keepAlive": false,

"pool": {

"vmSize": "Standard\_D2\_v2",

"virtualMachineConfiguration": {

"imageReference": {

"publisher": "canonical",

"offer": "0001-com-ubuntu-server-focal",

"sku": "20\_04-lts",

"version": "latest"

},

"nodeAgentSkuId": "batch.node.ubuntu 20.04"

},

"targetDedicatedNodes": 1

}

}

}

}

"@

$jobJson | Out-File -FilePath "job.json" -Encoding UTF8

az batch job create --json-file job.json

Remove-Item "job.json"

Write-Host "Job created successfully"

pwsh: true