# VARIABLES (update these for your environment)

$batchAccount = "lvbatchdev"

$resourceGroup = "lv-batch-dev"

$poolId = "batchpool-dev"

$appName = "ccdc-batch-jobs"

$identityName = "lv-batch-dev-UMI"

$identityResourceId = "/subscriptions/8d9342e2-96c8-436c-854a-1ad509c7356c/resourceGroups/lv-batch-dev/providers/Microsoft.ManagedIdentity/userAssignedIdentities/lv-batch-dev-UMI"

# Login to Azure

az login

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

# Login to Batch

az batch account login --name $batchAccount --resource-group $resourceGroup

# Delete existing pool if needed

$existingPool = az batch pool show --pool-id $poolId 2>$null

if ($existingPool) {

Write-Host "Deleting existing pool..."

az batch pool delete --pool-id $poolId --yes

Start-Sleep -Seconds 30

}

# Get latest active application package version

$latestVersion = az batch application package list `

--application-id $appName `

--query "[?state=='Active'] | sort\_by(@, &version) | [-1].version" -o tsv

if (-not $latestVersion) {

$latestVersion = "10979"

Write-Host "No active version found. Using: $latestVersion"

} else {

Write-Host "Using application version: $latestVersion"

}

# Construct pool JSON

$poolJson = @"

{

"id": "$poolId",

"vmSize": "standard\_d2s\_v3",

"targetDedicatedNodes": 1,

"virtualMachineConfiguration": {

"imageReference": {

"publisher": "MicrosoftWindowsServer",

"offer": "WindowsServer",

"sku": "2022-datacenter",

"version": "latest"

},

"nodeAgentSKUId": "batch.node.windows amd64"

},

"applicationPackageReferences": [

{

"applicationId": "$appName",

"version": "$latestVersion"

}

],

"identity": {

"type": "UserAssigned",

"userAssignedIdentities": {

"$identityResourceId": {}

}

},

"startTask": {

"commandLine": "cmd /c echo Pool bootstrap started && echo App package location: %AZ\_BATCH\_APP\_PACKAGE\_ccdc-batch-jobs% && dir %AZ\_BATCH\_APP\_PACKAGE\_ccdc-batch-jobs%\* && echo Bootstrap complete",

"userIdentity": {

"autoUser": {

"scope": "pool",

"elevationLevel": "admin"

}

},

"waitForSuccess": true,

"maxTaskRetryCount": 3

}

}

"@

# Save to file

$poolJson | Out-File -FilePath "pool.json" -Encoding utf8

Write-Host "Generated pool.json: "

Get-Content pool.json

# Create the pool

az batch pool create --json-file "pool.json"

if ($LASTEXITCODE -ne 0) {

Write-Error "Failed to create pool"

Remove-Item "pool.json" -Force

exit 1

}

Remove-Item "pool.json" -Force

Write-Host "Pool created successfully!"