**Steps:**

1. Create a Function App (Create a Resource -> Compute -> Function App)
2. Go to Functions in the left navigation
3. Add a Timer Trigger and choose how often you want this function to run

*\*Steps 4-6 will allow the function to run without the user having to log in every time*

1. Go to Identity (under Settings in the navigation on the left side)
2. Turn on Status and click Save
3. Under Permissions, add an Azure Role Assignment (Contributor to the Storage Account the file share you want to autoincrease is located in)

1. Go back to the Function App
2. Click on Functions
3. Select the function you previously created
4. Click on Code+Test in the left navigation
5. Copy and paste the code below:

#Input bindings are passed in via param block.

param($Timer)

#Get the current universal time in the default string format.

$currentUTCtime = (Get-Date).ToUniversalTime()

#The 'IsPastDue' property is 'true' when the current function invocation is later than scheduled.

if ($Timer.IsPastDue) {

Write-Host "PowerShell timer is running late!"

}

#Variable Definitions

$subscription\_id = “subscription ID”

$resource\_group = “resource group name”

$storage\_account\_name = “storage account name”

$file\_share\_name = “file share name”

#Define how many requests the workload can tolerate being throttled. 0 = No tolerance.

$ThrottleTolerance = 0

#Autogrow Percentage, default is 20%.

$AutoGrowRate = 1.2

#Connect to Azure and Import Az Module

Connect-AzAccount -Identity

Import-Module -Name Az.Accounts

Import-Module -Name Az.Storage

Set-AzContext -SubscriptionId $subscription\_id

#Get ResourceID

$resourceID = Get-AzResource -Name $storage\_account\_name

$resourceID = $resourceID.ResourceId +"/fileServices/default"

#Get file share

$PFS = Get-AzRmStorageShare -ResourceGroupName $resource\_group -StorageAccountName $storage\_account\_name -Name $file\_share\_name -GetShareUsage

#Determine if Success With Share Egress Throttling existed over 1 hour

$dimFilter = [String](New-AzMetricFilter -Dimension responsetype -Operator eq -Value "SuccessWithShareEgressThrottling")

$SuccessShareEgressThrottling = Get-AzMetric -ResourceId $ResourceID -MetricName Transactions -TimeGrain 01:00:00 -MetricFilter $dimFilter -AggregationType "Total"

#Determine if Success With Share Ingress Throttling existed over 1 hour

$dimFilter = [String](New-AzMetricFilter -Dimension responsetype -Operator eq -Value "SuccessWithShareIngressThrottling")

$SuccessShareIngressThrottling = Get-AzMetric -ResourceId $ResourceID -MetricName Transactions -TimeGrain 01:00:00 -MetricFilter $dimFilter -AggregationType "Total"

#Determine if Success With Share Iops Throttling existed over 1 hour

$dimFilter = [String](New-AzMetricFilter -Dimension responsetype -Operator eq -Value "SuccessWithShareIopsThrottling")

$SuccessShareIopsThrottling = Get-AzMetric -ResourceId $ResourceID -MetricName Transactions -TimeGrain 01:00:00 -MetricFilter $dimFilter -AggregationType "Total"

#Determine if Client Share Egress Throttling Error existed over 1 hour

$dimFilter = [String](New-AzMetricFilter -Dimension responsetype -Operator eq -Value "ClientShareEgressThrottlingError")

$ClientShareEgressThrottlingError = Get-AzMetric -ResourceId $ResourceID -MetricName Transactions -TimeGrain 01:00:00 -MetricFilter $dimFilter -AggregationType "Total"

#Determine if Client Share Ingress Throttling Error existed over 1 hour

$dimFilter = [String](New-AzMetricFilter -Dimension responsetype -Operator eq -Value "ClientShareIngressThrottlingError")

$ClientShareIngressThrottlingError = Get-AzMetric -ResourceId $ResourceID -MetricName Transactions -TimeGrain 01:00:00 -MetricFilter $dimFilter -AggregationType "Total"

#Determine if Client Share Iops Throttling Error existed over 1 hour

$dimFilter = [String](New-AzMetricFilter -Dimension responsetype -Operator eq -Value "ClientShareIopsThrottlingError")

$ClientShareIopsThrottlingError = Get-AzMetric -ResourceId $ResourceID -MetricName Transactions -TimeGrain 01:00:00 -MetricFilter $dimFilter -AggregationType "Total"

#Determine if Client Share Iops Throttling Error existed over 1 hour

$dimFilter = [String](New-AzMetricFilter -Dimension responsetype -Operator eq -Value "ClientThrottlingError")

$ClientThrottlingError = Get-AzMetric -ResourceId $ResourceID -MetricName Transactions -TimeGrain 01:00:00 -MetricFilter $dimFilter -AggregationType "Total"

#Get provisioned capacity, used capacity, and Avg IOPS

$ProvisionedCapacity = $PFS.QuotaGiB

$UsedCapacity = $PFS.ShareUsageBytes

#Determine Throttling Total

$SuccessShareEgressThrottleTotal = $SuccessShareEgressThrottling.Data.Total

$SuccessShareIngressThrottleTotal = $SuccessShareIngressThrottling.Data.Total

$SuccessShareIopsThrottleTotal = $SuccessShareIopsThrottling.Data.Total

$ClientShareEgressThrottleErrorTotal = $ClientShareEgressThrottlingError.Data.Total

$ClientShareIngressThrottleErrorTotal = $ClientShareIngressThrottlingError.Data.Total

$ClientShareIopsThrottleErrorTotal = $ClientShareIopsThrottlingError.Data.Total

$ClientThrottleErrorTotal = $ClientThrottlingError.Data.Total

#Current Workload

Write-Host "Provisioned Capacity:" $ProvisionedCapacity

Write-Host "Share Usage GBytes:" ($UsedCapacity / ([Math]::Pow(2,30)))

Write-Host "Success Share Egress Throttle Total:" $SuccessShareEgressThrottleTotal

Write-Host "Success Share Ingress Throttle Total:" $SuccessShareIngressThrottleTotal

Write-Host "Success Share Iops Throttle Total:" $SuccessShareIopsThrottleTotal

Write-Host "Client Share Egress Throttle Error Total:" $ClientShareEgressThrottlingErrorTotal

Write-Host "Client Share Ingress Throttle Error Total:" $ClientShareIngressThrottleErrorTotal

Write-Host "Client Share Iops Throttle Error Total:" $ClientShareIopsThrottleErrorTotal

Write-Host "Client Throttle Error Total:" $ClientThrottleErrorTotal

#get storage account

$StorageAccount = Get-AzStorageAccount -ResourceGroupName $resource\_group -AccountName $storage\_account\_name

#if throttling requests exceeds the throttling tolerance, increase provisioned capacity by 20%

if (($ThrottleTolerance -lt $SuccessShareEgressThrottleTotal) -or

($ThrottleTolerance -lt $SuccessShareIngressThrottleTotal) -or

($ThrottleTolerance -lt $SuccessShareIopsThrottleTotal) -or

($ThrottleTolerance -lt $ClientShareEgressThrottlingErrorTotal) -or

($ThrottleTolerance -lt $ClientShareIngressThrottleErrorTotal) -or

($ThrottleTolerance -lt $ClientShareIopsThrottleErrorTotal) -or

($ThrottleTolerance -lt $ClientThrottleErrorTotal))

{

$Quota = $ProvisionedCapacity\*$AutoGrowRate

Update-AzRmStorageShare -StorageAccount $StorageAccount -Name $file\_share\_name -QuotaGiB $Quota

$ProvisionedCapacity = $Quota

Write-Host "New Provisioned Capacity:" $ProvisionedCapacity

}

#Write an information log with the current time.

Write-Host "PowerShell timer trigger function ran! TIME:" $currentUTCtime