```powershell
##############################################
### Invoke Commands on Guest from vCenter
### ECI.EMI.Automation.OS.Invoke.ps1
##############################################

Param(
    [Parameter(Mandatory = $True)] [string]$ServerID,
    [Parameter(Mandatory = $True)] [string]$ConfigurationMode,
    [Parameter(Mandatory = $True)] [string]$HostName,
    [Parameter(Mandatory = $True)] [string]$ServerRole,
    [Parameter(Mandatory = $True)] [string]$IPv4Address,
    [Parameter(Mandatory = $True)] [string]$SubnetMask,
    [Parameter(Mandatory = $True)] [string]$DefaultGateway,
    [Parameter(Mandatory = $True)] [string]$PrimaryDNS,
    [Parameter(Mandatory = $True)] [string]$SecondaryDNS,
    [Parameter(Mandatory = $True)] [string]$ClientDomain,
    [Parameter(Mandatory = $True)] [string]$AdministrativeUserName,
    [Parameter(Mandatory = $True)] [string]$AdministrativePassword
)

function Rename-ECI.EMI.Configure.OS.Invoke.LocalAdmin
{
    Param(
    [Parameter(Mandatory = $True)] [string]$NewName,
    [Parameter(Mandatory = $True)] [string]$LocalAdminName,
    [Parameter(Mandatory = $True)] [string]$LocalAdminPassword
    )

    $FunctionName = $((Get-PSCallStack)[0].Command);Write-Host `r`n('-' * 75)`r`n
    "EXECUTING FUNCTION: " $FunctionName `r`n('-' * 75) -ForegroundColor Gray

    $ECILocalAdminName = $NewName

    $Params = @{
        "#ECILocalAdminName#" = $ECILocalAdminName
    }

    Write-Host `r`n("=" * 20)`r`n"RENAMING LOCAL ADMIN:"`r`n("=" * 20)`r`n
    -ForegroundColor Yellow
    Write-Host "Old LocalAdminName   : $LocalAdminName"     -ForegroundColor Cyan
    Write-Host "New LocalAdminName   : $ECILocalAdminName"  -ForegroundColor Cyan

    $RenameLocalAdmin ={
        ### Use .NET to Find the Current Local Administrator Account
        Add-Type -AssemblyName System.DirectoryServices.AccountManagement
        $ComputerName          = [System.Net.Dns]::GetHostName()
        $PrincipalContext      = New-Object
        System.DirectoryServices.AccountManagement.PrincipalContext([System.DirectoryServ
        ices.AccountManagement.ContextType]::Machine, $ComputerName)
        $UserPrincipal         = New-Object
        System.DirectoryServices.AccountManagement.UserPrincipal($PrincipalContext)
        $Searcher              = New-Object
        System.DirectoryServices.AccountManagement.PrincipalSearcher
        $Searcher.QueryFilter = $UserPrincipal

        ### The Administrator account is the only account that has a SID that ends with
        "-500"
        $Account = $Searcher.FindAll() | Where-Object {$_.Sid -Like "*-500"}
        $CurrentAdminName = $Account.Name

        $ECILocalAdminName = "#ECILocalAdminName#"

        Rename-LocalUser -Name $CurrentAdminName -NewName $ECILocalAdminName
        -ErrorAction SilentlyContinue | Out-Null
    }


    ### Inject Variables into ScriptText Block
    ### ---------------------------------------
```

```
62        foreach ($Param in $Params.GetEnumerator())
63        {
64            $RenameLocalAdmin =  $RenameLocalAdmin -replace $Param.Key,$Param.Value
65        }
66
67        try
68        {
69            ### IMPORTANT! Rename-Admin will genererate an excpected error, this is
              expected. Use "-ErrorAction SilentlyContinue | Out-Null".
70            Invoke-VMScript -ScriptText $RenameLocalAdmin -VM $VMName -ScriptType
              Powershell -GuestUser $Creds.LocalAdminName -GuestPassword
              $Creds.LocalAdminPassword -ErrorAction SilentlyContinue | Out-Null
71        }
72        catch
73        {
74            Write-ECI.ErrorStack
75        }
76        Write-Host `r`n('-' * 50)`r`n "END FUNCTION:" $FunctionName -ForegroundColor DarkGray
77    }
78
79    ########################################
80    ### ORIGINAL: Execute Invoke Process
81    ########################################
82    [ScriptBlock]$BootStrapModuleLoader =
83    {
84        ipconfig /all
85        ### BEGIN: Import BootStrap Module Loader
86        Set-ExecutionPolicy ByPass -Scope CurrentUser
87        $AcctKey=ConvertTo-SecureString -String
          "VSRMGJZNI4vn0nf47J4bqVd5peNiYQ/8+ozlgzbuA1FUnn9hAoGRM9Ib4HrkxOyRJkd4PHE8j36+pfnCUw3o
          8Q==" -AsPlainText -Force
88        $Credentials=$Null
89        $Credentials=New-Object System.Management.Automation.PSCredential -ArgumentList
          "Azure\eciscripts", $AcctKey
90        $RootPath="\\eciscripts.file.core.windows.net\clientimplementation"
91        New-PSDrive -Name V -PSProvider FileSystem -Root $RootPath -Credential $Credentials
          -Persist -Scope Global
92        #.
          "\\eciscripts.file.core.windows.net\clientimplementation\Root\ECI.Root.ModuleLoader.p
          s1" -Env dev
93        ### END: Import BootStrap Module Loader
94    }
95    function Invoke-ECI.ScriptText-ORIGINAL
96    {
97
98        ### Replace Parameters with #LiteralValues#
99        ### ------------------------------------
100       $Params = @{
101       "#Env#"                           = $Env
102       "#Environment#"                   = $Environment
103       "#Step#"                          = $Step
104       "#VMName#"                        = $VMName
105       "#ServerID#"                      = $ServerID
106       "#ServerRole#"                    = $ServerRole
107       "#HostName#"                      = $HostName
108       "#ClientDomain#"                  = $ClientDomain
109       "#IPv4Addres#"                    = $IPv4Address
110       "#SubnetMask#"                    = $SubnetMask
111       "#DefaultGateway#"                = $DefaultGateway
112       "#PrimaryDNS#"                    = $PrimaryDNS
113       "#SecondaryDNS#"                  = $SecondaryDNS
114       "#BuildVersion#"                  = $BuildVersion
115       "#CDROMLetter#"                   = $CDROMLetter
116       "#InternetExplorerESCPreference#" = $InternetExplorerESCPreference
117       "#IPv6Preference#"                = $IPv6Preference
118       "#NetworkInterfaceName#"          = $NetworkInterfaceName
119       "#SMBv1#"                         = $SMBv1
120       "#RDPResetrictionsPreference#"    = $RDPResetrictionsPreference
121       "#RemoteDesktopPreference#"       = $RemoteDesktopPreference
```

```
122         "#PageFileLocation#"              = $PageFileLocation
123         "#PageFileMultiplier#"            = $PageFileMultiplier
124         "#WindowsFirewallPreference#"     = $WindowsFirewallPreference
125         "#AdministrativeUserName#"        = $AdministrativeUserName
126         "#AdministrativePassword#"        = $AdministrativePassword
127         "#AutomationLogPath#"             = $AutomationLogPath
128         }
129
130         ### Inject Variables into ScriptText Block
131         ### ------------------------------------
132         foreach ($Param in $Params.GetEnumerator())
133         {
134             $ScriptText =  $ScriptText -replace $Param.Key,$Param.Value
135         }
136
137         ### Inject BootStrap Module Loader into VM
            Host                                          # <----- not using bootstrap ????
138         ### ------------------------------------
139         #$ScriptText =  $ScriptText -replace '#BootStrapModuleLoader#',$BootStrapModuleLoader
140
141         ### Debugging: Write ScriptText Block to Screen
142         ### ------------------------------------
143         #Write-Host "ScriptText:`r`n" $ScriptText -ForegroundColor Gray
144
145         ###############################
146         ### Inovke VMScript
147         ###############################
148         #-------------------------------------------------------
149         #   Invoke-VMScript
150         #     -Verbose
151         #     -Debug
152         #     | Select -ExpandProperty ScriptOutput
153         #     | Select -ExpandProperty ExitCode
154         #-------------------------------------------------------
155
156         Write-Host `r`n('*' * 55)`r`n`r`n "      ~~~~~~~~ INVOKING OS CONFIGURATION
            ~~~~~~~~  " `r`n`r`n(' ' * 18) "STEP:" $Step `r`n`r`n " --------- THIS PROCESS MAY
            TAKE SEVERAL MINUTES ---------  " `r`n`r`n('*' * 55)`r`n -ForegroundColor Cyan
157
158         ### ----------------------------------------
159         ### Production: Run Invoke as Variable
160         ### ----------------------------------------
161
162         ### Test Guest
163         ###---------------------------------
164         Start-ECI.EMI.Automation.Sleep -t $WaitTime_StartSleep -Message "Test Guest
            State"          #<---- COMBINE
165         Wait-ECI.EMI.Automation.VM.VMTools -VMName $VMName                #<---- COMBINE
166         Test-ECI.EMI.VM.GuestState -VMName $VMName                        #<---- COMBINE
167         #Test-ECI.EMI.Automation.VM.InvokeVMScript -VMName $VMName           #<---- COMBINE
168
169         ### Invoke Script Block
170         ### ------------------------------------
171         Write-Host "Invoke: Please Wait. This may take a while ..." -ForegroundColor Yellow
172         Write-Host "ScriptText: " `r`n $ScriptText -ForegroundColor DarkMagenta
173
174         function Try-InvokeScriptext
175         {
176             $script:Invoke             = $False
177             $script:InvokeRetryCounter = 0
178             $script:RetryCount         = 3
179
180             Invoke-VMScript -ScriptText $ScriptText -VM $VMName -ScriptType Powershell
                -GuestUser $Creds.LocalAdminName -GuestPassword $Creds.LocalAdminPassword
                -ErrorAction Stop -ErrorVariable +ECIError
181         }
182
183         function Retry-InvokeScriptext
184         {
```

```
185             Param([Parameter(Mandatory = $True)][int]$script:InvokeRetryCounter)
186
187             for ($i=1; $i -le $RetryCount; $i++)
188             {
189                 Start-ECI.EMI.Automation.Sleep -t 30 -Message "Retry"
190                 Write-Warning "Re-trying Invoke....  Count: " <# $InvokeRetryCounter #>
                    -WarningAction Continue
191                 Write-Host    "ERROR:" `r`n $Error[0] -ForegroundColor Red
192
193                 Try-InvokeScriptext
194             }
195
196             Write-Host "ABORT ERROR!" -ForegroundColor Red
197             #Write-ECI.ErrorStack
198         }
199
200         try
201         {
202             Try-InvokeScriptext
203         }
204
205         catch [VMware.VimAutomation.ViCore.Types.V1.ErrorHandling.GuestOperationsUnavailable]
206         {
207             ### The guest operations agent could not be contacted.
208
209             $script:InvokeRetryCounter ++
210             Write-Host "ERROR     :" `r`n $Error[0] -ForegroundColor Red
211             Write-Host "CAUGHT    :
                VMware.VimAutomation.ViCore.Types.V1.ErrorHandling.GuestOperationsUnavailable"
                -ForegroundColor DarkGray
212             Write-Host "EXCEPTION :" $Error[0].Exception.GetType().Fullname
                -ForegroundColor Red
213
214             Write-Host "Restarting VMTools."  -ForegroundColor Yellow
215             Restart-ECI.EMI.VM.VMTools
216
217             Write-Host "Re-Try Invoke."  -ForegroundColor Yellow
218             Retry-InvokeScriptext -InvokeRetryCounter $InvokeRetryCounter
219         }
220
221         catch [VMware.VimAutomation.ViCore.Types.V1.ErrorHandling.SystemError]
222         {
223             $script:InvokeRetryCounter ++
224             Write-Host "ERROR     :" `r`n $Error[0] -ForegroundColor Red
225             Write-Host "CAUGHT    :  System.Management.Automation.ErrorRecord"
                -ForegroundColor DarkGray
226             Write-Host "EXCEPTION :" $Error[0].Exception.GetType().Fullname
                -ForegroundColor Red
227
228             Write-Host "Restarting VMTools."  -ForegroundColor DarkGray
229             Restart-ECI.EMI.VM.VMTools
230
231             Write-Host "Re-Try Invoke."  -ForegroundColor DarkGray
232             Retry-InvokeScriptext -InvokeRetryCounter $InvokeRetryCounter
233         }
234
235         catch
236         {
237             $script:InvokeRetryCounter ++
238             Write-Host "ERROR     :" `r`n $Error[0] -ForegroundColor Red
239             Write-Host "CAUGHT    :  Error"  -ForegroundColor DarkGray
240             Write-Host "EXCEPTION :" $Error[0].Exception.GetType().Fullname
                -ForegroundColor Red
241
242             Write-Host "Restarting VMTools."  -ForegroundColor DarkGray
243             Restart-ECI.EMI.VM.VMTools -InvokeRetryCounter $InvokeRetryCounter
244
245             Write-Host "Re-Try Invoke."  -ForegroundColor DarkGray
246             Retry-InvokeScriptext
```

```
247         }
248
249         finally
250         {
251             if($Invoke)
252             {
253                 ### Check Exit Code for any Errors
254                 ### --------------------------------------
255                 if (($Invoke.ExitCode) -eq 0)
256                 {
257                     $ExitCodeStatus = "Success"
258                     $ExitCodeColor  = "Green"
259                 }
260                 elseif (($Invoke.ExitCode) -ne 0)
261                 {
262                     $ExitCodeStatus = "Failure"
263                     $ExitCodeColor  = "Red"
264                 }
265                 Write-Host "Invoke.ExitCode Status :"       $ExitCodeStatus
                    -ForegroundColor $ExitCodeColor
266                 Write-Host "Invoke.ExitCode        :"       $Invoke.ExitCode
                    -ForegroundColor $ExitCodeColor
267                 Write-Host "Invoke.ScriptOutput    : " `r`n $Invoke.ScriptOutput
                    -ForegroundColor $ExitCodeColor
268             }
269         }
270
271         Write-Host `r`n('-' * 50)`r`n "END FUNCTION:" $FunctionName -ForegroundColor DarkGray
272     }
273     function Invoke-ECI.EMI.Configure.OS.InGuest--ORIGINAL
274     {
275         Param([Parameter(Mandatory = $True)] [string]$Step)
276         $ScriptText = {
277         #BootStrapModuleLoader#
278         $global:Env                                 = "#Env#"
279         $global:Environment                         = "#Environment#"
280         $global:Step                                = "#Step#"
281         $global:VMName                              = "#VMName#"
282         $global:ServerID                            = "#ServerID#"
283         $global:ServerRole                          = "#ServerRole#"
284         $global:HostName                            = "#HostName#"
285         $global:ClientDomain                        = "#ClientDomain#"
286         $global:IPv4Address                         = "#IPv4Addres#"
287         $global:SubnetPrefixLength                  = "#SubnetPrefixLength#"
288         $global:DefaultGateway                      = "#DefaultGateway#"
289         $global:PrimaryDNS                          = "#PrimaryDNS#"
290         $global:SecondaryDNS                        = "#SecondaryDNS#"
291         $global:BuildVersion                        = "#BuildVersion#"
292         $global:CDROMLetter                         = "#CDROMLetter#"
293         $global:InternetExplorerESCPreference       = "#InternetExplorerESCPreference#"
294         $global:IPv6Preference                      = "#IPv6Preference#"
295         $global:NetworkInterfaceName                = "#NetworkInterfaceName#"
296         $global:SMBv1                               = "#SMBv1#"
297         $global:RDPResetrictionsPreference          = "#RDPResetrictionsPreference#"
298         $global:RemoteDesktopPreference             = "#RemoteDesktopPreference#"
299         $global:PageFileLocation                    = "#PageFileLocation#"
300         $global:PageFileMultiplier                  = "#PageFileMultiplier#"
301         $global:WindowsFirewallPreference           = "#WindowsFirewallPreference#"
302         $global:AdministrativeUserName              = "#AdministrativeUserName#"
303         $global:AdministrativePassword              = "#AdministrativePassword#"
304         $global:AutomationLogPath                   = "#AutomationLogPath#"
305         foreach($Module in (Get-Module -ListAvailable ECI.*)){Import-Module -Name
            $Module.Path -DisableNameChecking}
306         . "C:\Program
            Files\WindowsPowerShell\Modules\ECI.Modules.$Env\ECI.EMI.Configure.OS.$Env\ECI.EMI.Co
            nfigure.OS.InGuest.$Env.ps1"
307     } # END ScriptText
308
309
```

```
    ### Clean the Scripttext Block
    ###----------------------------------------------------------------------
    $CleanScriptText = $Null
    foreach( $Line in ((($Scripttext -replace("  ","") -replace("= ","=")) -replace(" =","=") ) -split("`r`n"))) | ? {$_.Trim() -ne ""} )
    {
        $Line = ($Line) + "`r`n"
        $CleanScriptText = $CleanScriptText + $Line
    }

    [int]$CharLimit = 2869
    [int]$CharCount = ($CleanScriptText | Measure-Object -Character).Characters
    if($CharCount -gt $CharLimit)
    {
        Write-Warning "The Scripttect block exceeds $CharLimit Chararter Limit."
    }
    elseif($CharCount -lt $CharLimit)
    {
        Write-Host "The Scripttect block is under the $CharLimit Chararter Limit." -ForegroundColor DarkGreen
    }
    Write-Host "Scripttext Character Count: " $CharCount -ForegroundColor DarkGray
    $ScriptText = $CleanScriptText
    Invoke-ECI.ScriptText
}
########################################
########################################


########################
### Execute the Script
########################
&{
    BEGIN
    {
        #Start-Transcript -IncludeInvocationHeader -OutputDirectory (Set-TranscriptPath) #<--(Set Path)
        #Start-ECI.Transcript -TranscriptPath "C:\Scripts\ServerRequest\TranscriptLogs\" -TranscriptName "ECI.EMI.Configure.OS.Invoke.$Env.ps1"

        ### Write Header Information
        ###---------------------------------
        Write-Host `r`n`r`n('*' * 100)`r`n (' ' * 20)" --------- STARTING OS CONFIGURATION --------- " `r`n('*' * 100)  -ForegroundColor Cyan
        Write-Host `r`n('-' * 50)`r`n -ForegroundColor DarkCyan
        Write-Host "Env         : " $Env -ForegroundColor DarkCyan
        Write-Host "Environment : " $Environment -ForegroundColor DarkCyan
        Write-Host "Script      : " (Split-Path (Get-PSCallStack)[0].ScriptName -Leaf) -ForegroundColor DarkCyan
        Write-Host `r`n('-' * 50)`r`n -ForegroundColor DarkCyan

        ### Script Setup
        ###---------------------------------
        $script:OSStartTime = Get-Date
        $global:VerifyErrorCount = 0


#!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
### Waiting after OS Customization/Reboot
Write-Host `r`n`r`n "*** Waiting after OS Customization/Reboot *** " -ForegroundColor Magenta

Start-ECI.EMI.Automation.Sleep -t $WaitTime_StartSleep -Message "Start OS
```

```
Configuration"                                              #<--- COMBINE!!!!!
Wait-ECI.EMI.Automation.VM.VMTools -VMName $VMName                                    #<--- COMBINE!!!!!
#Wait-ECI.EMI.GuestState -VMName $VMName
#Interrogate-ECI.EMI.Automation.VM.GuestState -VMName $VMName -HostName $HostName
#!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

        Set-ECI.EMI.Automation.LocalAdminAccount -Template ###  Set Creds here for Execution Policy
        Configure-ECI.EMI.Automation.ExecutionPolicyonVMGuest -VMName $VMName
        Delete-ECI.EMI.Automation.ServerLogs -HostName $HostName

        ### Install ECI Modules on Guest
        ###---------------------------------
        $Params = @{
            Env         = $Env
            Environment = $Environment
        }
        Install-ECI.EMI.Automation.ECIModulesonVMGuest @Params
        Write-ECI.EMI.OS.ParameterstoGuest -ServerID $ServerID -VMName $VMName -HostName $HostName
    }

    PROCESS
    {
        ##########################################
        ### Invoke Configuration in VM Guest
        ##########################################

        ###----------------------------------------------
        ### STEP 1: Rename-ECI.LocalAdmin
        ###----------------------------------------------
        function Rename-ECI.EMI.Configure.OS.LocalAdmin
        {
            $FunctionName = $((Get-PSCallStack)[0].Command);Write-Host `r`n('-' * 75)`r`n "EXECUTING FUNCTION: " $FunctionName `r`n('-' * 75) -ForegroundColor Gray

            #Delete-ECI.EMI.Automation.ServerLogs -HostName $HostName


            ###---------------------------------------------------------------------------------------------
            Set-ECI.EMI.Automation.LocalAdminAccount -Template

            Rename-ECI.EMI.Configure.OS.Invoke.LocalAdmin -NewName $ECILocalAdminName -LocalAdminName $Creds.LocalAdminName -LocalAdminPassword $Creds.LocalAdminPassword
            Set-ECI.EMI.Automation.LocalAdminAccount -ECI

            ###---------------------------------------------------------------------------------------------


            ### WARNING:  Dont use these functions here! There are no log files from this operation. Will generate terminating errors.

            ###---------------------------------------------------------------------------------------------------------------------------------
            #Copy-ECI.EMI.Automation.VMLogsfromGuest
            #Write-ECI.EMI.Automation.VMLogstoSQL
            #Delete-ECI.EMI.Automation.ServerLogs -HostName $HostName

            ###---------------------------------------------------------------------------------------------------------------------------------

            Write-Host `r`n('-' * 50)`r`n "END FUNCTION:" $FunctionName -ForegroundColor DarkGray
```

```
416         }
417
418         ###-------------------------------------------
419         ### STEP 2: Rename-ECI.GuestComputer
420         ###-------------------------------------------
421         function Rename-ECI.EMI.Configure.OS.GuestComputer
422         {
423             $FunctionName = $((Get-PSCallStack)[0].Command);Write-Host `r`n('-' *
                75)`r`n "EXECUTING FUNCTION: " $FunctionName `r`n('-' * 75)
                -ForegroundColor Gray
424
425
                ###-------------------------------------------------------------------
                ------------------
426             Wait-ECI.EMI.Automation.VM.VMTools -VMName $VMName
427             $Step = "Rename-ECI.EMI.Configure.OS.GuestComputer"
428             Invoke-ECI.EMI.Automation.ScriptTextInGuest -ScriptText
                (Process-ECI.EMI.Automation.ScriptText -Step $Step -Env $Env -Environment
                $Environment) -Step $Step
429
430
                ###-------------------------------------------------------------------
                ------------------
431             Write-Host "Guest OS was restarted after Renaming Computer:  `r`nWaiting
                for Guest OS to Resume . . ." -ForegroundColor Yellow
432             Wait-ECI.EMI.Automation.VM.VMTools -VMName
                $VMName                                          #<---- Consolidate
433             Start-ECI.EMI.Automation.Sleep -t $WaitTime_StartSleep -Message "Waiting
                after Rename Guest"                                    #<----
                Consolidate
434
435             ### Copy Log Files and Write to SQL
436             ###---------------------------------
437             Copy-ECI.EMI.Automation.VMLogsfromGuest -VMName $VMName -HostName
                $HostName                    #<---- Consolidate
438             Write-ECI.EMI.Automation.VMLogstoSQL                      #<---- Consolidate
439             Delete-ECI.EMI.Automation.ServerLogs -HostName $HostName #<---- Consolidate
440
441             Write-Host `r`n('-' * 50)`r`n "END FUNCTION:" $FunctionName
                -ForegroundColor DarkGray
442         }
443
444         ###-------------------------------------------
445         ### STEP 3: Configure OS
446         ###-------------------------------------------
447         function Configure-ECI.EMI.Configure.OS.GuestComputer
448         {
449             $FunctionName = $((Get-PSCallStack)[0].Command);Write-Host `r`n('-' *
                75)`r`n "EXECUTING FUNCTION: " $FunctionName `r`n('-' * 75)
                -ForegroundColor Gray
450
451             #Test-ECI.EMI.Automation.VM.InvokeVMScript -VMName
                $VMName                          #<---- Consolidate
452             #Test.ECI.EMI.VM.GuestReady  -VMName $VMName                          #<----
                Consolidate
453
454
455             ### Mount OS ISO
456             ###---------------------------------
457             $Params = @{
458                 VMName      = $VMName
459                 ISOName     = "2016Server"
460             }
461             Mount-ECI.EMI.Automation.VM.ISO -VMName $VMName -ISOName 2016Server
462             Start-ECI.EMI.Automation.Sleep -t $WaitTime_StartSleep -Message "Mount
                CD-ROM ISO"                                          #<---- Consolidate
463
464
                ###-------------------------------------------------------------------
```

```
            ------------------
            #Invoke-ECI.EMI.Configure.OS.InGuest -Step
            Configure-ECI.EMI.Configure.OS.GuestComputer
            Wait-ECI.EMI.Automation.VM.VMTools -VMName $VMName
            $Step = "Configure-ECI.EMI.Configure.OS.GuestComputer"
            Invoke-ECI.EMI.Automation.ScriptTextInGuest -ScriptText
            (Process-ECI.EMI.Automation.ScriptText -Step $Step -Env $Env -Environment
            $Environment) -Step $Step

            ###-------------------------------------------------------------------------
            ------------------

            DisMount-ECI.EMI.Automation.VM.ISO -VMName $VMName
            Start-ECI.EMI.Automation.Sleep -t 30 -Message "Dis-Mount CD-ROM
            ISO"                                        #<---- Consolidate


            ### Copy Log Files and Write to SQL
            ###--------------------------------
            Copy-ECI.EMI.Automation.VMLogsfromGuest -VMName $VMName -HostName
            $HostName                  #<---- Consolidate
            Write-ECI.EMI.Automation.VMLogstoSQL                      #<---- Consolidate
            Delete-ECI.EMI.Automation.ServerLogs -HostName $HostName #<---- Consolidate

            Write-Host `r`n('-' * 50)`r`n "END FUNCTION:" $FunctionName
            -ForegroundColor DarkGray
        }

        ###-----------------------------------------------
        ### STEP 4: RegisterDNS
        ###-----------------------------------------------
        function Configure-ECI.EMI.Configure.RegisterDNS
        {
            $FunctionName = $((Get-PSCallStack)[0].Command);Write-Host `r`n('-' *
            75)`r`n "EXECUTING FUNCTION: " $FunctionName `r`n('-' * 75)
            -ForegroundColor Gray

            #Invoke-ECI.EMI.Configure.OS.InGuest -Step
            Configure-ECI.EMI.Configure.OS.RegisterDNS

            $Step = "Configure-ECI.EMI.Configure.OS.RegisterDNS"
            Invoke-ECI.EMI.Automation.ScriptTextInGuest -ScriptText
            (Process-ECI.EMI.Automation.ScriptText -Step $Step -Env $Env -Environment
            $Environment) -Step $Step


            Write-Host `r`n('-' * 50)`r`n "END FUNCTION:" $FunctionName
            -ForegroundColor DarkGray
        }

        ###-----------------------------------------------
        ### STEP 5: Configure Roles
        ###-----------------------------------------------
        function Configure-ECI.EMI.Configure.Roles.GuestComputer
        {
            $FunctionName = $((Get-PSCallStack)[0].Command);Write-Host `r`n('-' *
            75)`r`n "EXECUTING FUNCTION: " $FunctionName `r`n('-' * 75)
            -ForegroundColor Gray

            Write-Host `r`n('=' * 75)`r`n "Configuring Server Role: " $ServerRole
            `r`n('=' * 75)`r`n -ForegroundColor Cyan
            Write-Host `r`n`r`n('*' * 100)`r`n (' ' * 20)" --------- STARTING ROLE
            CONFIGURATION --------- " `r`n('*' * 100)  -ForegroundColor Cyan

            ### Role Specific Configurerations
            ###--------------------------------
            switch ( $ServerRole )
            {
                "2016Server"
```

```
                {
                    $ConfigureRole = $False
                }
                "2016FS"
                {
                    $ConfigureRole = $False
                }
                "2016DC"
                {
                    $ConfigureRole = $False
                }
                "2016DCFS"
                {
                    $ConfigureRole = $False
                }
                "2016VDA"
                {
                    $ConfigureRole = $True
                    Mount-ECI.EMI.Automation.VM.ISO -VM $VMName -ISOName "XenApp"
                }
                "2016SQL"
                {
                    $ConfigureRole = $False
                }
                "2016SQLOMS"
                {
                    $ConfigureRole = $False
                }
            }

            ### Execute Role Configureration
            ###-------------------------------
            if($ConfigureRole -eq $True)
            {
                Write-Host "ConfigureRole:  " $ConfigureRole -ForegroundColor Cyan
                Write-Host "Configuring Server Role . . . " -ForegroundColor Cyan


                ###-------------------------------------------------------------------------
                ----------------------
                #Invoke-ECI.EMI.Configure.OS.InGuest -Step $ServerRole
                Wait-ECI.EMI.Automation.VM.VMTools -VMName $VMName
                $Step = "$ServerRole"
                Invoke-ECI.EMI.Automation.ScriptTextInGuest -ScriptText
                (Process-ECI.EMI.Automation.ScriptText -Step $Step -Env $Env
                -Environment $Environment)

                ###-------------------------------------------------------------------------
                ----------------------

                Copy-ECI.EMI.Automation.VMLogsfromGuest -VMName $VMName -HostName
                $HostName#<---- Consolidate
                Write-ECI.EMI.Automation.VMLogstoSQL    #<---- Consolidate
                Delete-ECI.EMI.Automation.ServerLogs  -HostName $HostName  #<----
                Consolidate
            }

            elseif($ConfigureRole -eq $False)
            {
                Write-Host "ConfigureRole:  " $ConfigureRole -ForegroundColor Cyan
                Write-Host "There are no specific Role based configurations needed for
                this build." -ForegroundColor DarkCyan
            }
            Write-Host `r`n('-' * 50)`r`n "END FUNCTION:" $FunctionName
            -ForegroundColor DarkGray
        }

        ###-------------------------------------------
        ### STEP 6: Restart Guest OS
```

```
574         ###-------------------------------------------
575         function Restart-ECI.EMI.Configure.OS.GuestComputer
576         {
577
578
579             ### Stop VM
580             ###---------------------------
581             Stop-ECI.EMI.Automation.VM                                                  #
                <--- need gracefule shutdown and restart!!!!!!!!!!!!!!!!!!
582
583             ### Stop VM
584             ###---------------------------
585             Start-ECI.EMI.Automation.VM
586
587             Write-Host `r`n('-' * 50)`r`n "END FUNCTION:" $FunctionName
                -ForegroundColor DarkGray
588         }
589
590         &{
591             PROCESS
592             {
593                 ###-----------
594                 ### Step: 1
595                 ###-----------
596                 Rename-ECI.EMI.Configure.OS.LocalAdmin
597
598                 ###-----------
599                 ### Step: 2
600                 ###-----------
601                 Rename-ECI.EMI.Configure.OS.GuestComputer
602
603                 ###-----------
604                 ### Step: 3
605                 ###-----------
606                 Configure-ECI.EMI.Configure.OS.GuestComputer
607
608                 ###-----------
609                 ### Step: 4
610                 ###-----------
611                 Configure-ECI.EMI.Configure.RegisterDNS
612
613                 ###-----------
614                 ### Step: 5
615                 ###-----------
616                 Configure-ECI.EMI.Configure.Roles.GuestComputer
617
618                 ###-----------
619                 ### Step: 6
620                 ###-----------
621                 Restart-ECI.EMI.Configure.OS.GuestComputer
622
623                 ###-----------
624                 ### Wait
625                 ###-----------
626                 Start-ECI.EMI.Automation.Sleep -t $WaitTime_StartSleep -message "Guest
                    Restarted"                          #<--- COMBINE !!!!!
627                 Wait-ECI.EMI.Automation.VM.VMTools -VMName
                    $VMName                              #<--- COMBINE !!!!!
628
629                 ###-----------
630                 ### Step: 7
631                 ###-----------
632                 #QA-ECI.EMI.Configure.OS.GuestComputer
633
634             }
635         }
636     }
637
638     END
```

```
639         {
640             #Delete-ECI.EMI.Automation.ECIModulesonVMGuest
641             $OSStopTime = Get-Date
642             $global:OSElapsedTime = ($OSStopTime-$OSStartTime)
643             Write-Host `r`n`r`n('=' * 75)`r`n "OS Configuration: Total Execution Time:`t"
                $OSElapsedTime `r`n('=' * 75)`r`n -ForegroundColor Gray
644             Write-Host "END OS onfiguration Script" -ForegroundColor Gray
645             #Stop-Transcript
646 
647             ### END CONFIGURE OS INVOKE SCRIPT
648         }
649 
650     }
651 
652 
```