param $option

The only accepted option is "debug", which is not case sensitive

Setup environment

Setup the $Path, the SWP main method to call, and the job list. Set the working directory

while $KEEP_RUNNING

Start-Sleep 300 seconds

foreach $Job in $JobList

Get-Job

is $null -ne $myJob

YES

is $null -eq $myJob(StartJob)

YES

$KEEP_RUNNING = $false

EXCEPTION!

Language method pulls the process name from the kernel, returns output to dictionary if found, otherwise returns $null

ellseif $option -eq "DEBUG"

YES

write-out debug data

is $myJob.State -eq "Completed"

ellseif $option -eq "DEBUG"

YES

write-out debug data

Remove-Job

ellseif $option -eq "DEBUG"

EXCEPTION $KEEP_RUNNING = $false

StartJob $myJob, $ProgramToRun

is $Test-Path $ProgramTo Run?

YES

return(Start-Thread -Name $myJob)

-ThrottleLimit hard coded to 4 threads, requires VM CP size of 8 procs if run on same host as Nagios Core
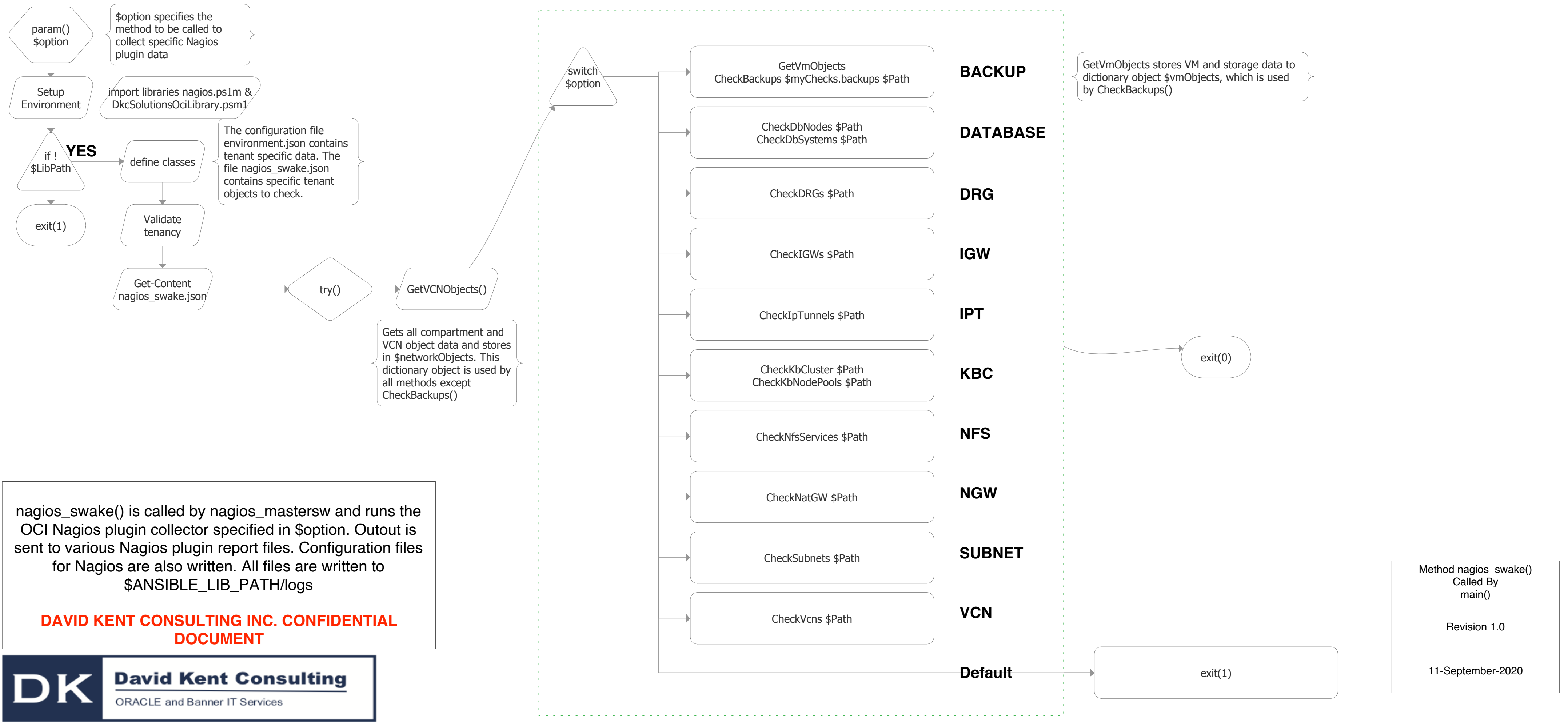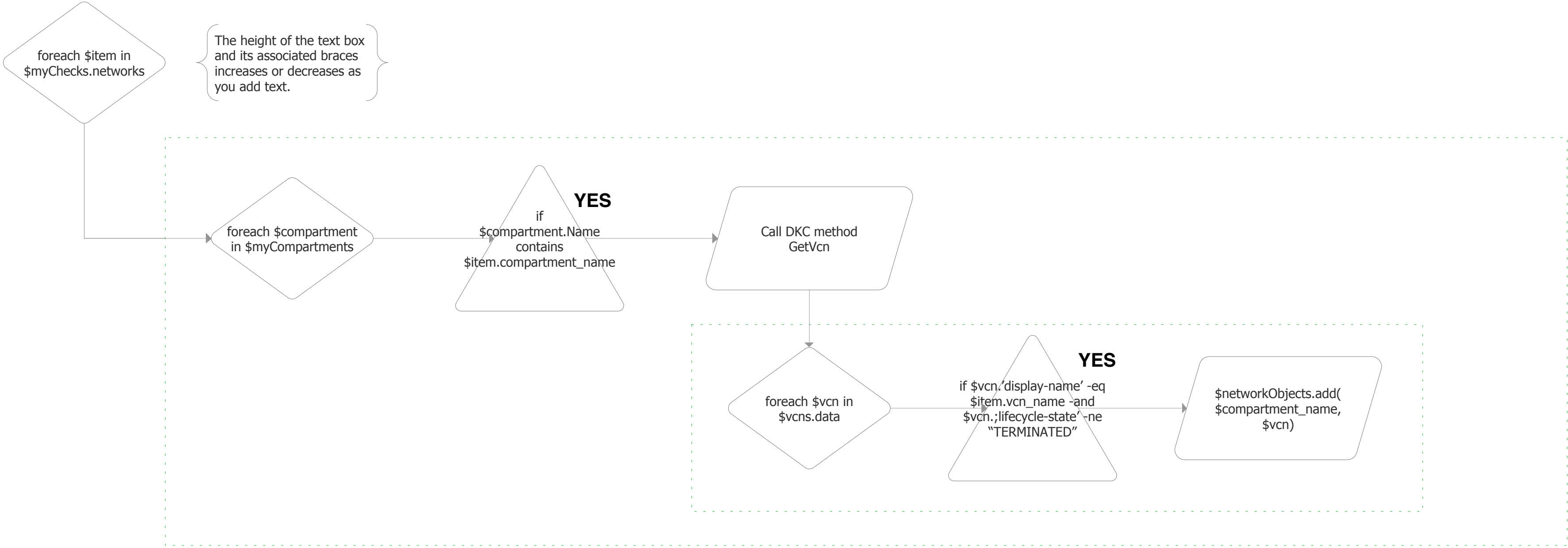
return($null)

Program nagios_mastersw.ps1multi-threaded sleep-wake daemon launches multiple concurrent jobs to available PROCS, then sleeps for 5 minutes. It then awakens and repeats infinitely unless an exception is detected, in which case it gracefully exits.
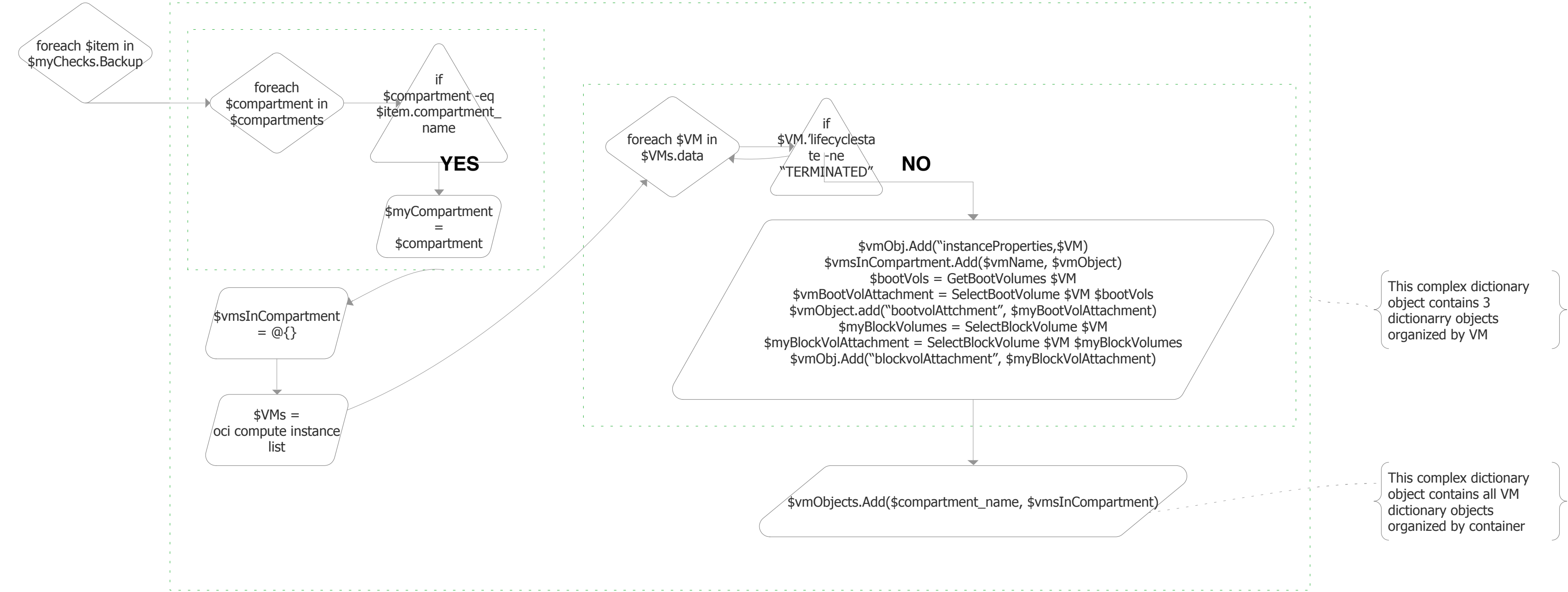
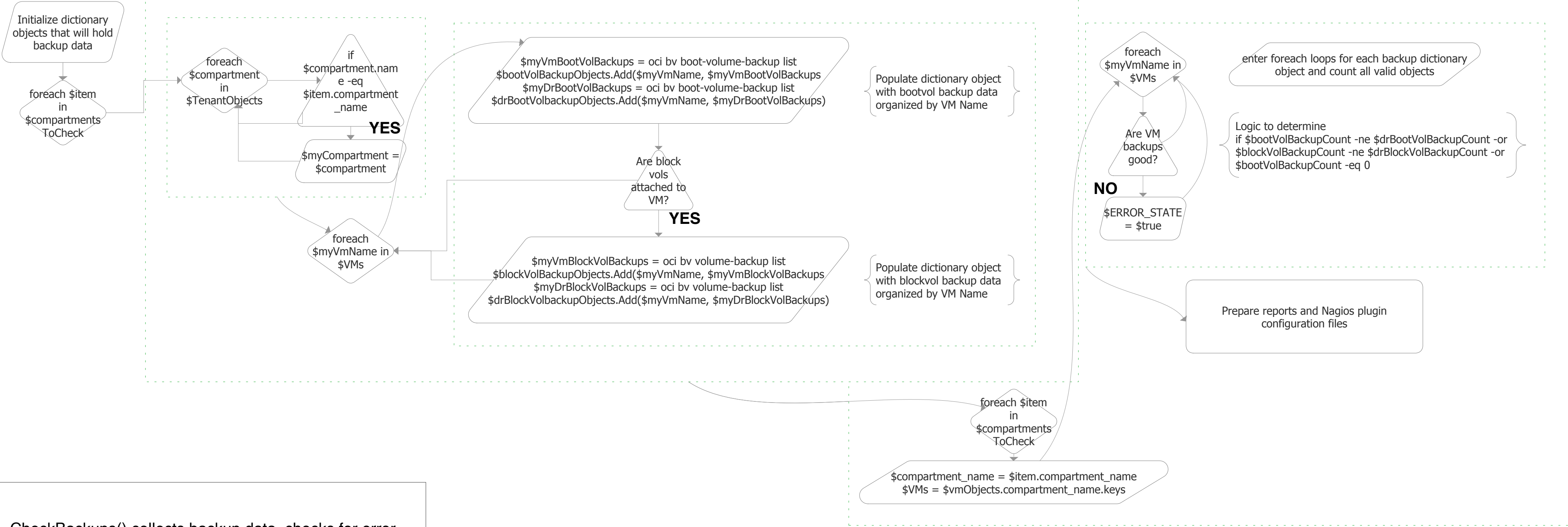**DAVID KENT CONSULTING INC. CONFIDENTIAL DOCUMENT**

DK **David Kent Consulting**
ORACLE and Banner IT Services

main()
Revision 1.0
11-September-2020

param()
$option

$option specifies the method to be called to collect specific Nagios plugin data

Setup Environment

import libraries nagios.ps1m & DkcSolutionsOciLibrary.psm1

if ! $LibPath — **YES** → define classes

The configuration file environment.json contains tenant specific data. The file nagios_swake.json contains specific tenant objects to check.

exit(1)

Validate tenancy

Get-Content nagios_swake.json → try() → GetVCNObjects()

Gets all compartment and VCN object data and stores in $networkObjects. This dictionary object is used by all methods except CheckBackups()

switch $option

GetVmObjects
CheckBackups $myChecks.backups $Path — **BACKUP**

GetVmObjects stores VM and storage data to dictionary object $vmObjects, which is used by CheckBackups()

CheckDbNodes $Path
CheckDbSystems $Path — **DATABASE**

CheckDRGs $Path — **DRG**

CheckIGWs $Path — **IGW**

CheckIpTunnels $Path — **IPT**

exit(0)

CheckKbCluster $Path
CheckKbNodePools $Path — **KBC**

CheckNfsServices $Path — **NFS**

CheckNatGW $Path — **NGW**

CheckSubnets $Path — **SUBNET**

CheckVcns $Path — **VCN**

Default → exit(1)

nagios_swake() is called by nagios_mastersw and runs the OCI Nagios plugin collector specified in $option. Outout is sent to various Nagios plugin report files. Configuration files for Nagios are also written. All files are written to $ANSIBLE_LIB_PATH/logs

DAVID KENT CONSULTING INC. CONFIDENTIAL DOCUMENT

DK David Kent Consulting
ORACLE and Banner IT Services

Method nagios_swake()
Called By
main()

Revision 1.0

11-September-2020

foreach $item in $myChecks.networks

The height of the text box and its associated braces increases or decreases as you add text.

foreach $compartment in $myCompartments

YES

if $compartment.Name contains $item.compartment_name

Call DKC method GetVcn

foreach $vcn in $vcns.data

YES

if $vcn.'display-name' -eq $item.vcn_name -and $vcn.;lifecycle-state' -ne "TERMINATED"

$networkObjects.add($compartment_name, $vcn)

GetVcnObjects creates dictionary objects for all VCNs specified within the .JSON file input to nagios_swake()
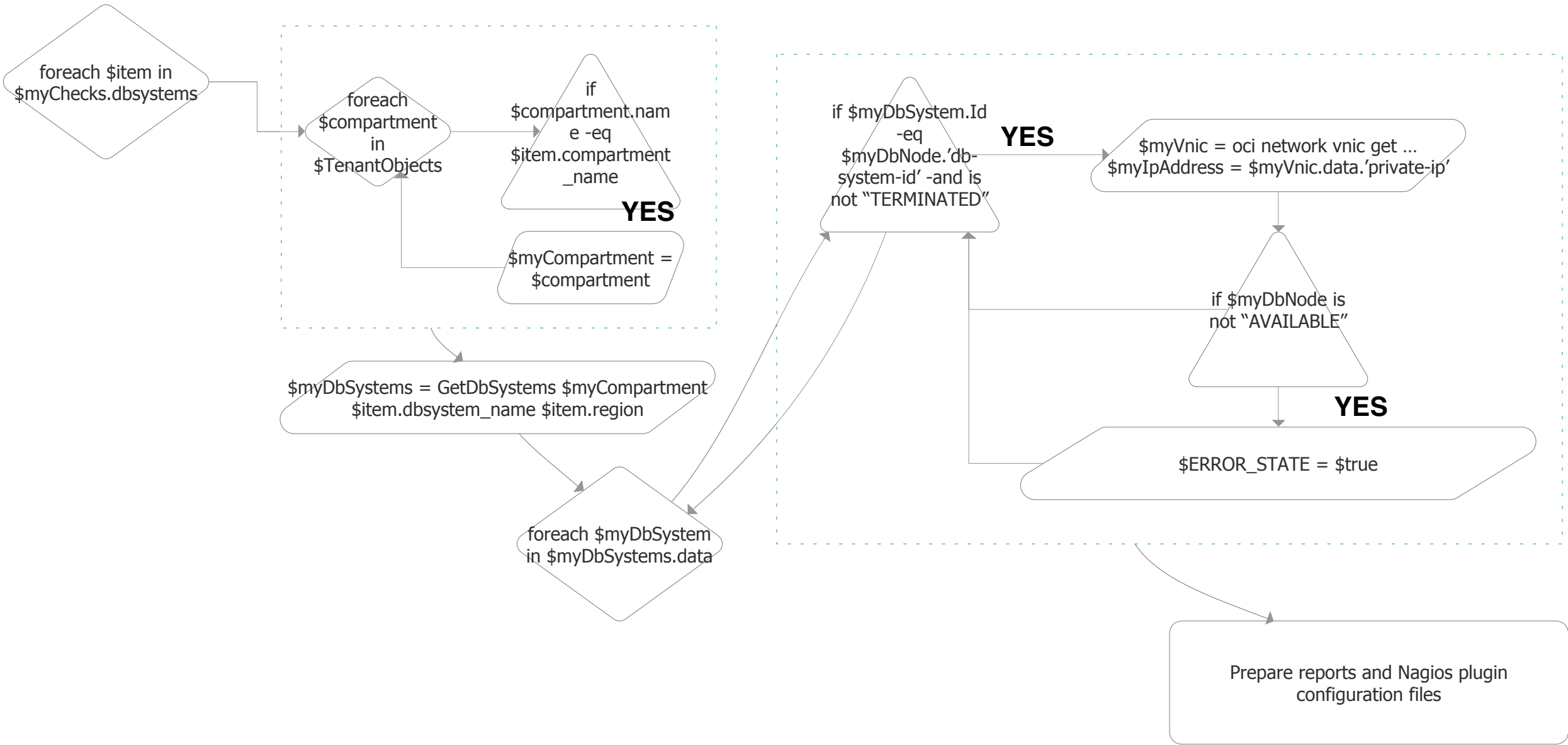
DK **David Kent Consulting**
ORACLE and Banner IT Services

foreach $item in
$myChecks.Backup

foreach
$compartment in
$compartments

if
$compartment -eq
$item.compartment_
name

YES

$myCompartment
=
$compartment

foreach $VM in
$VMs.data

if
$VM.'lifecyclesta
te -ne
"TERMINATED"

NO

$vmsInCompartment
= @{}

$vmObj.Add("instanceProperties,$VM)
$vmsInCompartment.Add($vmName, $vmObject)
$bootVols = GetBootVolumes $VM
$vmBootVolAttachment = SelectBootVolume $VM $bootVols
$vmObject.add("bootvolAttchment", $myBootVolAttachment)
$myBlockVolumes = SelectBlockVolume $VM
$myBlockVolAttachment = SelectBlockVolume $VM $myBlockVolumes
$vmObj.Add("blockvolAttachment", $myBlockVolAttachment)

This complex dictionary
object contains 3
dictionarry objects
organized by VM

$VMs =
oci compute instance
list

$vmObjects.Add($compartment_name, $vmsInCompartment)

This complex dictionary
object contains all VM
dictionary objects
organized by container

GetVmObjects method builds a complex dictionary object
that contains all reusable VM instance and disk attachment
data for all compartments specified in the .JSON file passed
to nagios_swake()

DK **David Kent Consulting**
ORACLE and Banner IT Services

| Method GetVmObjects()
Called By
nagios_swake() |
| Revision 1.0 |
| 11-September-2020 |

Initialize dictionary objects that will hold backup data

foreach $item in $compartments ToCheck

foreach $compartment in $TenantObjects

if $compartment.name -eq $item.compartment _name

YES

$myCompartment = $compartment

$myVmBootVolBackups = oci bv boot-volume-backup list
$bootVolBackupObjects.Add($myVmName, $myVmBootVolBackups
$myDrBootVolBackups = oci bv boot-volume-backup list
$drBootVolbackupObjects.Add($myVmName, $myDrBootVolBackups)

Populate dictionary object with bootvol backup data organized by VM Name

Are block vols attached to VM?

YES

foreach $myVmName in $VMs

$myVmBlockVolBackups = oci bv volume-backup list
$blockVolBackupObjects.Add($myVmName, $myVmBlockVolBackups
$myDrBlockVolBackups = oci bv volume-backup list
$drBlockVolbackupObjects.Add($myVmName, $myDrBlockVolBackups)

Populate dictionary object with blockvol backup data organized by VM Name

foreach $myVmName in $VMs

enter foreach loops for each backup dictionary object and count all valid objects

Logic to determine
if $bootVolBackupCount -ne $drBootVolBackupCount -or
$blockVolBackupCount -ne $drBlockVolBackupCount -or
$bootVolBackupCount -eq 0

Are VM backups good?

NO

$ERROR_STATE = $true

Prepare reports and Nagios plugin configuration files

foreach $item in $compartments ToCheck

$compartment_name = $item.compartment_name
$VMs = $vmObjects.compartment_name.keys

CheckBackups() collects backup data, checks for error conditions, and then produces a nagios plugin report with nagios configuration files.
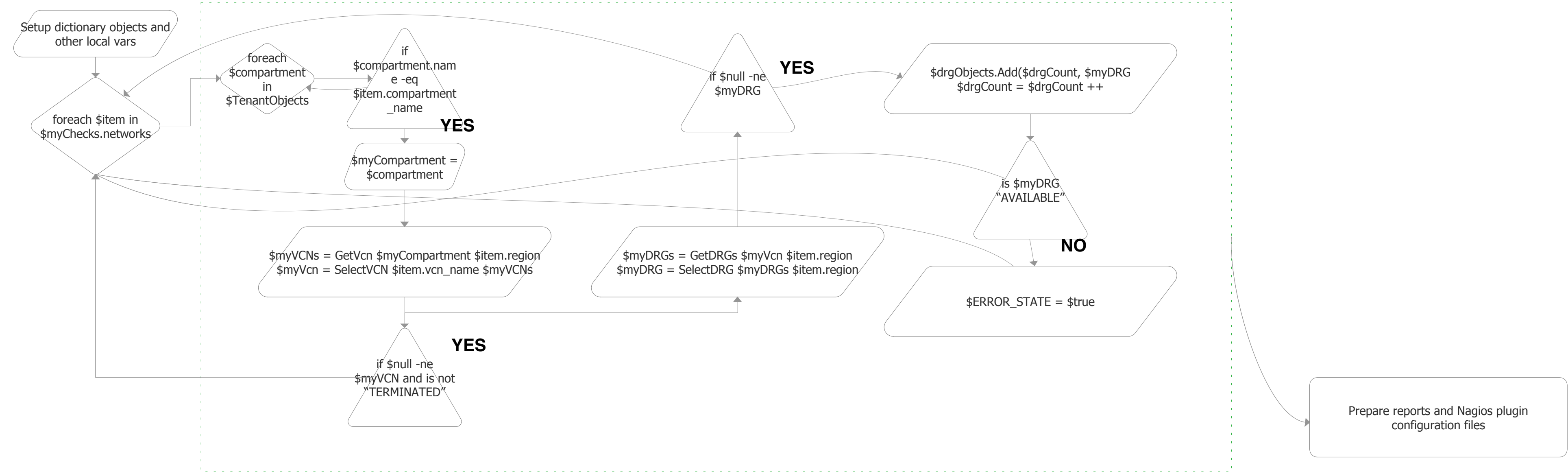
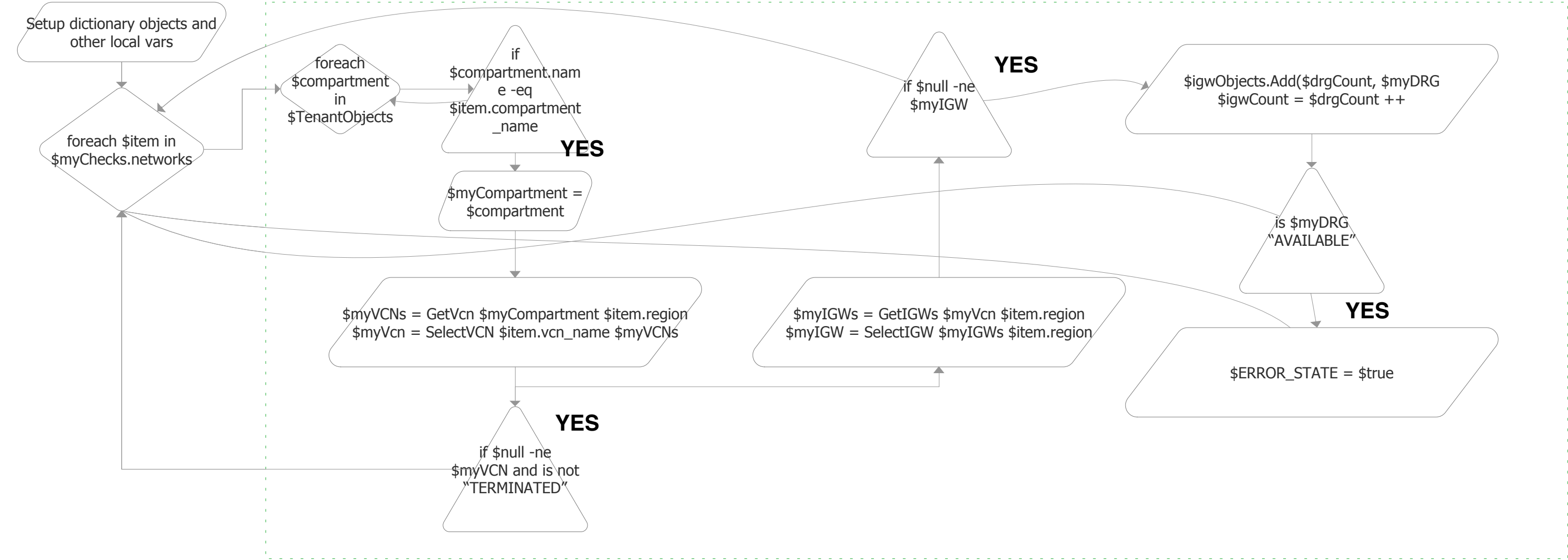**DAVID KENT CONSULTING INC. CONFIDENTIAL DOCUMENT**

DK **David Kent Consulting**
ORACLE and Banner IT Services

Method GetBackups()
Called By
nagios_swake()

Revision 1.0

11-September-2020

foreach $item in
$myChecks.dbsystems

foreach
$compartment
in
$TenantObjects

if
$compartment.nam
e -eq
$item.compartment
_name

YES

$myCompartment =
$compartment

$myDbSystems = GetDbSystems $myCompartment
$item.dbsystem_name $item.region

foreach $myDbSystem
in $myDbSystems.data

if $myDbSystem.Id
-eq
$myDbNode.'db-
system-id' -and is
not "TERMINATED"

YES

$myVnic = oci network vnic get ...
$myIpAddress = $myVnic.data.'private-ip'

if $myDbNode is
not "AVAILABLE"

YES

$ERROR_STATE = $true

Prepare reports and Nagios plugin
configuration files

CheckDbNodes() collects the state of DBaaS service nodes,
checks for error conditions, and then produces a nagios
plugin report with nagios configuration files.

DK **David Kent Consulting**
ORACLE and Banner IT Services

Method CheckDbNodes()
Called By
nagios_swake()

Revision 1.0

11-September-2020

foreach $item in $myChecks.dbsystems

foreach $compartment in $TenantObjects

if $compartment.name -eq $item.compartment_name

YES

$myCompartment = $compartment

$myDbSystems = GetDbSystems $myCompartment $item.dbsystem_name $item.region

foreach $myDbSystem in $myDbSystems.data

$record = SelectDbSystem $myDbSystem.'display-name' $item.region
$dbSystemObjects.add($cntr, $record)

if $record -ne "AVAILABLE"

YES

$ERROR_STATE = $true

Prepare reports and Nagios plugin configuration files

CheckDbSystems() collects the state of DBaaS service by DBaaS, checks for error conditions, and then produces a nagios plugin report with nagios configuration files.

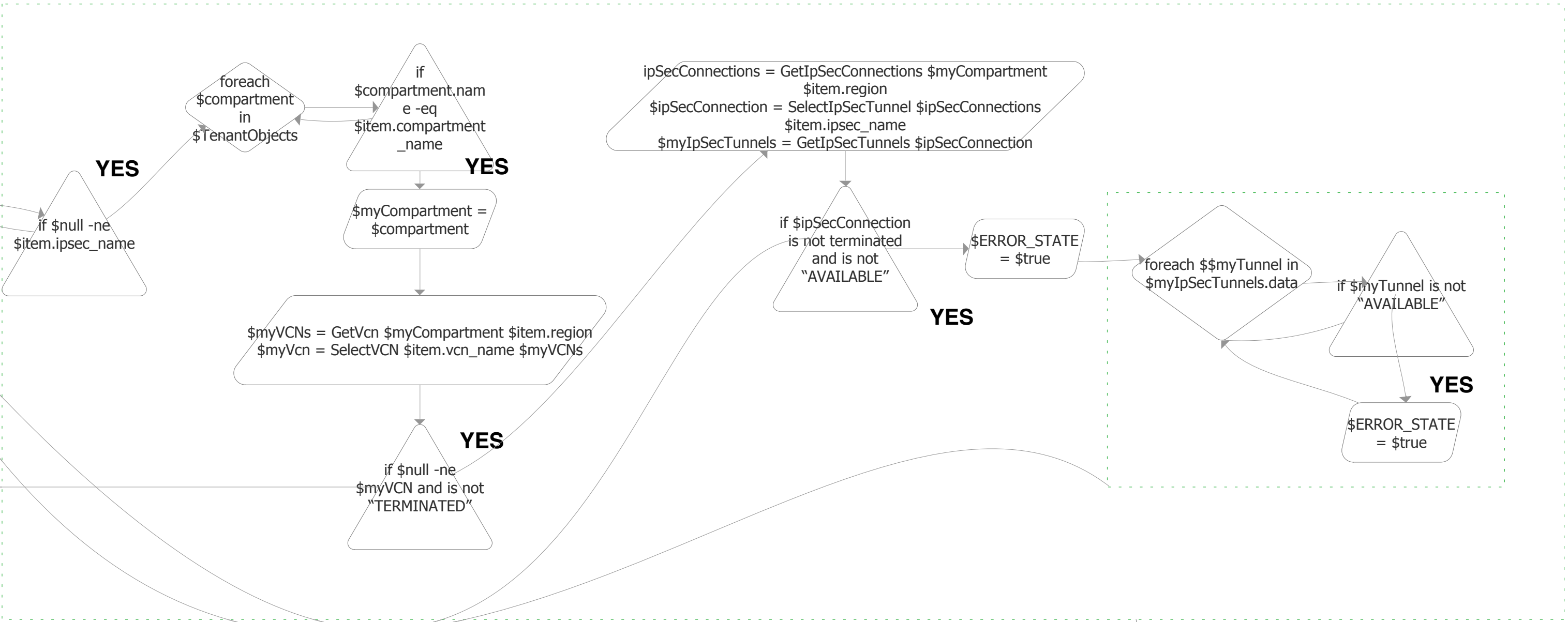**DAVID KENT CONSULTING INC. CONFIDENTIAL DOCUMENT**

Method CheckDbSystems()
Called By
nagios_swake()
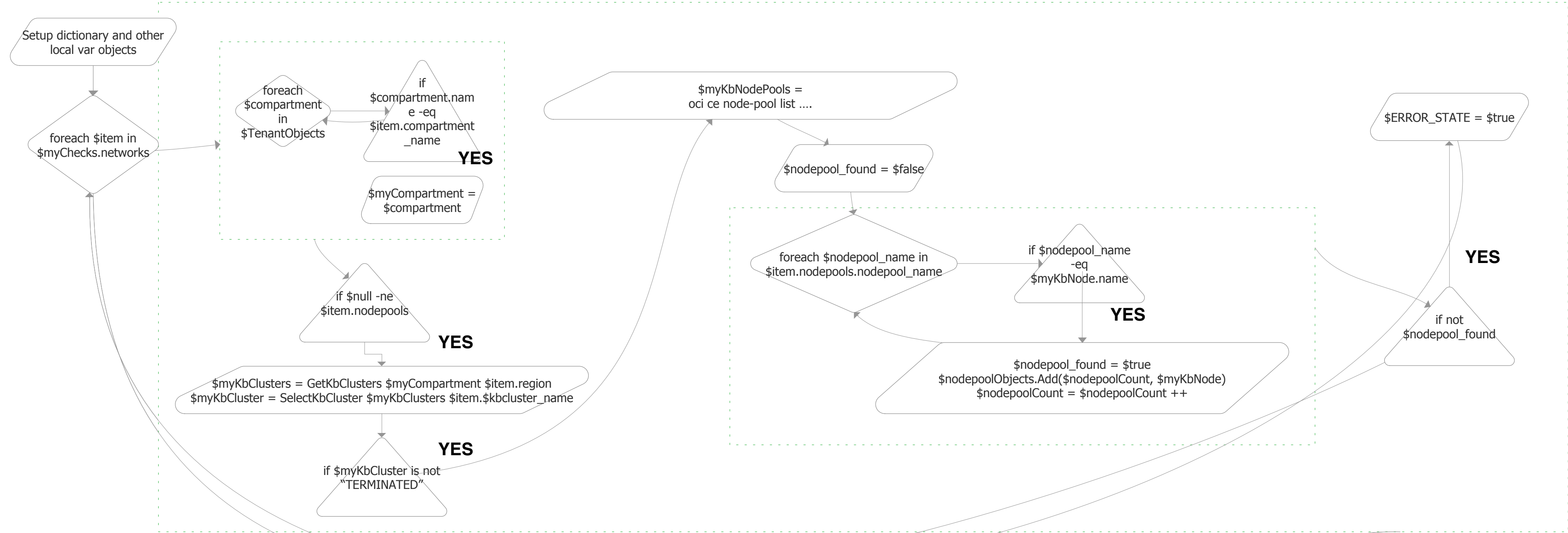
Revision 1.0

11-September-2020

Setup dictionary objects and other local vars

foreach $compartment in $TenantObjects

if $compartment.name -eq $item.compartment_name **YES**

foreach $item in $myChecks.networks

if $null -ne $myDRG **YES**

$drgObjects.Add($drgCount, $myDRG
$drgCount = $drgCount ++

$myCompartment = $compartment

is $myDRG "AVAILABLE" **NO**

$myVCNs = GetVcn $myCompartment $item.region
$myVcn = SelectVCN $item.vcn_name $myVCNs

$myDRGs = GetDRGs $myVcn $item.region
$myDRG = SelectDRG $myDRGs $item.region

$ERROR_STATE = $true

if $null -ne $myVCN and is not "TERMINATED" **YES**

Prepare reports and Nagios plugin configuration files

CheckDRGs() collects the state of dynamic cloud routers within the VCN, checks for error conditions, and then produces a nagios plugin report with nagios configuration files.

Method CheckDRGs()
Called By
nagios_swake()

Revision 1.0

11-September-2020

Setup dictionary objects and other local vars

foreach $compartment in $TenantObjects

if $compartment.name -eq $item.compartment_name **YES**

foreach $item in $myChecks.networks

$myCompartment = $compartment

if $null -ne $myIGW

YES

$igwObjects.Add($drgCount, $myDRG
$igwCount = $drgCount ++

$myVCNs = GetVcn $myCompartment $item.region
$myVcn = SelectVCN $item.vcn_name $myVCNs

$myIGWs = GetIGWs $myVcn $item.region
$myIGW = SelectIGW $myIGWs $item.region

is $myDRG "AVAILABLE"

YES

$ERROR_STATE = $true

if $null -ne $myVCN and is not "TERMINATED" **YES**

Prepare reports and Nagios plugin configuration files

CheckIGWs() collects the state of internet cloud routers within the VCN, checks for error conditions, and then produces a nagios plugin report with nagios configuration files.

DK **David Kent Consulting**
ORACLE and Banner IT Services

Method CheckIGWs()
Called By
nagios_swake()

Revision 1.0

11-September-2020

Setup dictionary objects and other local vars

foreach $item in $myChecks.networks

if $null -ne $item.ipsec_name

YES

foreach $compartment in $TenantObjects

if $compartment.name -eq $item.compartment_name

YES

$myCompartment = $compartment

$myVCNs = GetVcn $myCompartment $item.region
$myVcn = SelectVCN $item.vcn_name $myVCNs

YES

if $null -ne $myVCN and is not "TERMINATED"

ipSecConnections = GetIpSecConnections $myCompartment $item.region
$ipSecConnection = SelectIpSecTunnel $ipSecConnections $item.ipsec_name
$myIpSecTunnels = GetIpSecTunnels $ipSecConnection

if $ipSecConnection is not terminated and is not "AVAILABLE"

YES

$ERROR_STATE = $true

foreach $$myTunnel in $myIpSecTunnels.data

if $myTunnel is not "AVAILABLE"

YES

$ERROR_STATE = $true

Prepare reports and Nagios plugin configuration files

CheckIpTunnels() collects the state of the IPSEC connections and IPSEC tunnels within the VCN, checks for error conditions, and then produces a nagios plugin report with nagios configuration files.

DAVID KENT CONSULTING INC. CONFIDENTIAL DOCUMENT

DK **David Kent Consulting**
ORACLE and Banner IT Services

Method CheckIpTunnels()
Called By
nagios_swake()

Revision 1.0

11-September-2020

Setup dictionary and other local var objects

foreach $item in $myChecks.networks

foreach $compartment in $TenantObjects

if $compartment.name -eq $item.compartment_name
YES

$myCompartment = $compartment

if $null -ne $item.nodepools
YES

$myKbClusters = GetKbClusters $myCompartment $item.region
$myKbCluster = SelectKbCluster $myKbClusters $item.$kbcluster_name

if $myKbCluster is not "TERMINATED"
YES

$myKbNodePools = oci ce node-pool list ....

$nodepool_found = $false

foreach $nodepool_name in $item.nodepools.nodepool_name

if $nodepool_name -eq $myKbNode.name
YES

$nodepool_found = $true
$nodepoolObjects.Add($nodepoolCount, $myKbNode)
$nodepoolCount = $nodepoolCount ++

$ERROR_STATE = $true

YES

if not $nodepool_found

Prepare reports and Nagios plugin configuration files

CheckKbNodePools() collects the state of the Kubernetes Nodepools within the compartment, checks for error conditions, and then produces a nagios plugin report with nagios configuration files.

DAVID KENT CONSULTING INC. CONFIDENTIAL DOCUMENT

DK **David Kent Consulting**
ORACLE and Banner IT Services

Method CheckKbNodePools()
Called By
nagios_swake()

Revision 1.0

11-September-2020

Initialize local dictionary and var objects

foreach $item in $myChecks.Networks

foreach $compartment in $TenantObjects

if $compartment.name -eq $item.compartment_name

YES

$myCompartment = $compartment

if $null -ne $item.natgw_name

YES

$myNatGWs = oci network nat-gateway list ....

if $null -ne $myNatGWs

YES

$myNatGW = SelectNGW $item.natgw_name $myNatGWs

if $null -ne $myNatGW

$natGwObjects.Add($natGwCount, $myNatGW $natGwCount ++

if $myNatGW not healthy

Logic: $myNatGW.'lifecycle-state' -ne 'TERMINATED' -and $myNatGW.'lifecycle-state' -ne 'AVAILABLE'

$ERROR_STATE = $true

NO

$myVCNs = GetVCN $myCompartment $item.dr_region

We have to account for checking for NAT GWs in the DR region

Any VCNs found that are not 'TERMINATED'

All conditions equal YES

$myNatGW = SelectNGW $item.natgw_name $myNatGWs

if $null -ne $myNatGW

if $null -ne $myNatGW

All conditions equal YES

$natGwObjects.Add($natGwCount, $myNatGW $natGwCount ++

if $myNatGW not healthy

$ERROR_STATE = $true

Prepare reports and Nagios plugin configuration files

CheckNetGW() collects the state of cloud network address translation gateways within the VCN, checks for error conditions, and then produces a nagios plugin report with nagios configuration files.

DAVID KENT CONSULTING INC. CONFIDENTIAL DOCUMENT

DK David Kent Consulting
ORACLE and Banner IT Services

Method CheckNetGW() Called By nagios_swake()

Revision 1.0

11-September-2020

Initialize local dictionary and var objects

foreach $item in $myChecks.fileStorage

foreach $compartment in $TenantObjects

if $compartment.name -eq $item.compartment_name

YES

$myCompartment = $compartment

$myAvailabilityDomains = GetAvaiabilityDomains $item.region

foreach $AD in $myAvailabilityDomains.data

$myFileSystems = GetFileSystems $myCompartment $AD.name $item.region
$myFileSystem = GetFileSystem $myFileSystems $item.filesystem_name

if $null -ne $myFileSystem

YES

$nfsObjects.Add($nfsCount, $myFileSystem)
$nfsCount ++

NO

if $myFileSystem not healthy

$ERROR_STATE = $true

Logic:
$myMountTarget.'lifecycle-state' -ne 'ACTIVE'

$myMountTargets = GetMountTargets $myCompartment $AD.name $item.region
$myMountTarget = GetMountTarget $myMountTargets $item.mount_target

if $null -ne $myMountTarget

YES

$mtObjects.Add($mtCount, $myMountTarget)
$mtCount ++

NO

if $myMountTarget not healthy

$ERROR_STATE = $true

Prepare reports and Nagios plugin configuration files

CheckNfsServices() collects the state of cloud NFS service and the Mount Point Service within the VCN, checks for error conditions, and then produces a nagios plugin report with nagios configuration files.

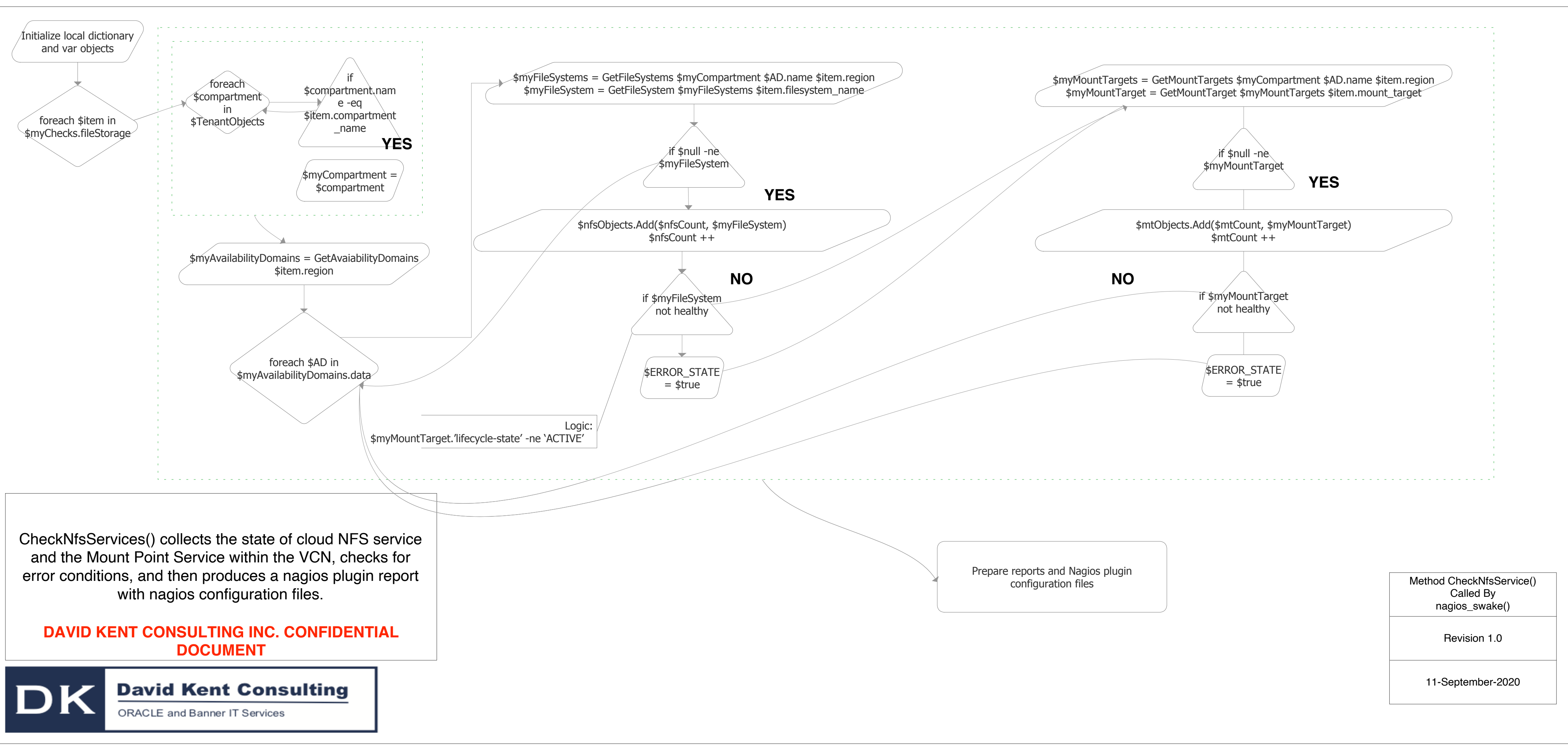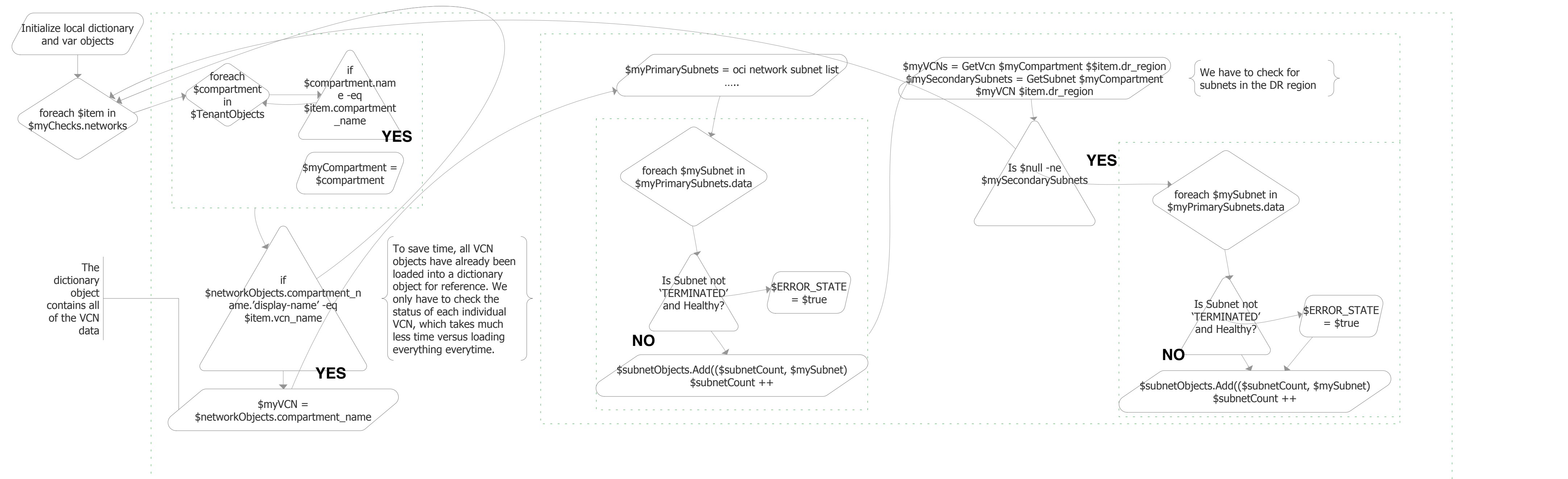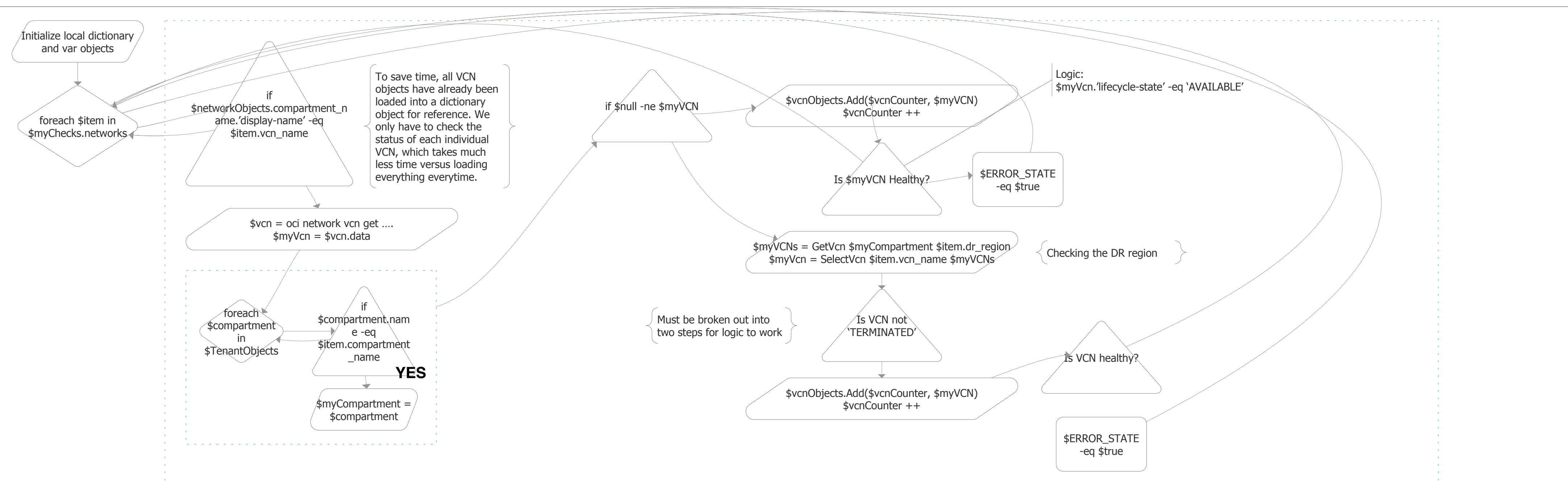**DAVID KENT CONSULTING INC. CONFIDENTIAL DOCUMENT**

DK **David Kent Consulting**
ORACLE and Banner IT Services

Initialize local dictionary and var objects

foreach $item in $myChecks.networks

foreach $compartment in $TenantObjects

if $compartment.name -eq $item.compartment_name

YES

$myCompartment = $compartment

The dictionary object contains all of the VCN data

if $networkObjects.compartment_name.'display-name' -eq $item.vcn_name

YES

$myVCN = $networkObjects.compartment_name

To save time, all VCN objects have already been loaded into a dictionary object for reference. We only have to check the status of each individual VCN, which takes much less time versus loading everything everytime.

$myPrimarySubnets = oci network subnet list .....

foreach $mySubnet in $myPrimarySubnets.data

Is Subnet not 'TERMINATED' and Healthy?

$ERROR_STATE = $true

NO

$subnetObjects.Add(($subnetCount, $mySubnet) $subnetCount ++

$myVCNs = GetVcn $myCompartment $$item.dr_region
$mySecondarySubnets = GetSubnet $myCompartment $myVCN $item.dr_region

We have to check for subnets in the DR region

Is $null -ne $mySecondarySubnets

YES

foreach $mySubnet in $myPrimarySubnets.data

Is Subnet not 'TERMINATED' and Healthy?

$ERROR_STATE = $true

NO

$subnetObjects.Add(($subnetCount, $mySubnet) $subnetCount ++

Prepare reports and Nagios plugin configuration files

CheckSubnets() collects the state of cloud subnet within the VCN, checks for error conditions, and then produces a nagios plugin report with nagios configuration files.

DK **David Kent Consulting**
ORACLE and Banner IT Services

Method CheckSubnets()
Called By
nagios_swake()

Revision 1.0

11-September-2020

Initialize local dictionary and var objects

foreach $item in $myChecks.networks

if $networkObjects.compartment_name.'display-name' -eq $item.vcn_name

To save time, all VCN objects have already been loaded into a dictionary object for reference. We only have to check the status of each individual VCN, which takes much less time versus loading everything everytime.

$vcn = oci network vcn get ....
$myVcn = $vcn.data

foreach $compartment in $TenantObjects

if $compartment.name -eq $item.compartment_name

YES

$myCompartment = $compartment

if $null -ne $myVCN

$vcnObjects.Add($vcnCounter, $myVCN)
$vcnCounter ++

Is $myVCN Healthy?

$ERROR_STATE -eq $true

Logic:
$myVcn.'lifecycle-state' -eq 'AVAILABLE'

$myVCNs = GetVcn $myCompartment $item.dr_region
$myVcn = SelectVcn $item.vcn_name $myVCNs

Checking the DR region

Must be broken out into two steps for logic to work

Is VCN not 'TERMINATED'

$vcnObjects.Add($vcnCounter, $myVCN)
$vcnCounter ++

Is VCN healthy?

$ERROR_STATE -eq $true

Prepare reports and Nagios plugin configuration files

CheckVCNs() collects the state of cloud VCN within the compartment, checks for error conditions, and then produces a nagios plugin report with nagios configuration files.

Method CheckVCN()
Called By
nagios_swake()

Revision 1.0

11-September-2020