***\*\*\* NAGIOS AND RUNDECK \*\*\****

Pre-Request : A LAMP stack is also required.

**%mInstall Nagios 4.**

<https://www.digitalocean.com/community/tutorials/how-to-install-nagios-4-and-monitor-your-servers-on-centos-7>

**Install Build Dependencies:**

sudo yum install gcc glibc glibc-common gd gd-devel make net-snmp openssl-devel xinetd unzip -y

**Create Nagios User and Group:**

sudo useradd nagios

sudo groupadd nagcmd

sudo usermod -a -G nagcmd nagios

**Install Nagios Core:**

cd /opt

curl -L -O https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.6.tar.gz

tar xvf nagios-\*.tar.gz

cd nagios-\*

**Before building Nagios, we must configure it with this command:**

./configure --with-command-group=nagcmd

Now compile Nagios with this command:

make all

**Now we can run these make commands to install Nagios, init scripts, and sample configuration files:**

sudo make install

sudo make install-commandmode

sudo make install-init

sudo make install-config

sudo make install-webconf

**In order to issue external commands via the web interface to Nagios, we must add the web server user, apache, to the nagcmd group:**

sudo usermod -G nagcmd apache

**Install Nagios Plugins:**

cd /opt; curl -L -O http://nagios-plugins.org/download/nagios-plugins-2.1.1.tar.gz

tar xvf nagios-plugins-\*.tar.gz

cd nagios-plugins-\*

**Before building Nagios Plugins, we must configure it. Use this command:**

./configure --with-nagios-user=nagios --with-nagios-group=nagios --with-openssl

**Now compile Nagios Plugins with this command:**

Make; sudo make install

**Install NRPE**

cd /opt;

curl -L -O http://downloads.sourceforge.net/project/nagios/nrpe-2.x/nrpe-2.15/nrpe-2.15.tar.gz

**Extract the NRPE archive with this comman**

tar xvf nrpe-\*.tar.gz

**Then change to the extracted directory:**

cd nrpe-\*

**Configure NRPE with these commands:**

./configure --enable-command-args --with-nagios-user=nagios --with-nagios-group=nagios --with-ssl=/usr/bin/openssl --with-ssl-lib=/usr/lib/x86\_64-linux-gnu

**Now build and install NRPE and its xinetd startup script with these commands:**

make all

sudo make install

sudo make install-xinetd

sudo make install-daemon-config

**Open the xinetd startup script in an editor:**

sudo vi /etc/xinetd.d/nrpe

only\_from = 127.0.0.1, 192.168.1.20(server Name)

**Restart the xinetd service to start NRPE:**

sudo service xinetd restart

**Configure Nagios:**

Organize Nagios Configuration

sudo vi /usr/local/nagios/etc/nagios.cfg

Now find an uncomment this line by deleting the #:

cfg\_dir=/usr/local/nagios/etc/servers

sudo mkdir /usr/local/nagios/etc/servers

**Configure Nagios Contacts**

sudo vi /usr/local/nagios/etc/objects/contacts.cfg

email silambarasancv.rajendran@gmail.com ; << \*\*CHANGE THIS TO YOUR EMAIL ADDRESS \*\*\*

**Configure check\_nrpe Command**

sudo vi /usr/local/nagios/etc/objects/commands.cfg

define command{

command\_name check\_nrpe

command\_line $USER1$/check\_nrpe -H $HOSTADDRESS$ -c $ARG1$

}

**Configure Apache**

sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin

Note: If you create a user that is not named “nagiosadmin”, you will need to edit /usr/local/nagios/etc/cgi.cfg and change all the “nagiosadmin” references to the user you created.

sudo systemctl daemon-reload

sudo systemctl start nagios.service

sudo systemctl restart httpd.service

sudo chkconfig nagios on

**Optional: Restrict Access by IP Address**

sudo vi /etc/httpd/conf.d/nagios.conf

Find and comment the following two lines by adding # symbols in front of them:

Order allow,deny

Allow from all

Then uncomment the following lines, by deleting the # symbols, and add the IP addresses or ranges (space delimited) that you want to allow to in the Allow from line:

# Order deny,allow

# Deny from all

# Allow from 127.0.0.1

**Now start Nagios and restart Apache to put the change into effect:**

sudo systemctl restart nagios.service

sudo systemctl restart httpd.service

**Accessing the Nagios Web Interface:**

**http://192.168.1.20/nagios**

**Monitor a CentOS 7 Host with NRPE**

sudo yum install epel-release

sudo yum install nrpe nagios-plugins-all

sudo vi /etc/nagios/nrpe.cfg

allowed\_hosts=127.0.0.1,192.168.1.20

**Restart NRPE to put the change into effect:**

sudo systemctl start nrpe.service

sudo systemctl enable nrpe.service

**Add Host to Nagios Configuration**

sudo vi /usr/local/nagios/etc/servers/jiraExampleCom.cfg

# Host Definetion

define host {

use linux-server

host\_name jira.example.com

alias My first Apache server

address 192.168.1.20

max\_check\_attempts 5

check\_period 24x7

notification\_interval 30

notification\_period 24x7

}

# Ping:

define service {

use generic-service

host\_name jira.example.com

service\_description PING

check\_command check\_ping!100.0,20%!500.0,60%

}

# SSH

define service {

use generic-service

host\_name jira.example.com

service\_description SSH

check\_command check\_ssh

notifications\_enabled 0

}

**Restart Nagios server**

sudo systemctl reload nagios.service

***Custom Plugin for Nagios server***

***\*\*\* ON LOCALHOST \*\*\****

<https://www.howtoforge.com/tutorial/write-a-custom-nagios-check-plugin/>

***Step 1*** - Install RPMForge Repository and NRPE on client VPS

rpm -ivh http://pkgs.repoforge.org/rpmforge-release/rpmforge-release-0.5.3-1.el6.rf.x86\_64.rpm

yum -y install nagios-nrpe

useradd nrpe && chkconfig nrpe on

(OR)

yum install nagios\* -y; useradd nrpe

***Step 2*** - Create your Bash Script

cd /usr/local/nagios/libexec/

[root@pocnagios libexec]# cat servicestatus.sh

#!/bin/bash

status=dead

for services in httpd sshd

do

systemctl status $services | grep -i 'running\|dead' | awk '{print $3}' | sed 's/[()]//g' | while read output;

do

if [ "$output" == "$status" ]; then

echo "CRITICAL - $services Service is NOT Running"

curl -D - \

-X "POST" -H "Accept: application/json" \

-H "Content-Type: application/json" \

-H "X-Rundeck-Auth-Token: NSUt9PpUTTJ6INH7xcjt09VyMcr8cexX" \

-d "{\"argString\":\"-servicename $services \"}" \

http://192.168.5.116:4440/api/16/job/769de6f3-c068-45de-90ce-c636a9297488/executions

exit 2

else

echo "OK - All Services are Running"

exit 0

fi

done

done

chmod +x servicestatus.sh

vi /usr/local/nagios/etc/objects/commands.cfg

# Custom plugins commands...

define command {

command\_name check\_servicestatus\_ps1

command\_line $USER1$/check\_nrpe -H $HOSTADDRESS$ -c check\_servicestatus\_ps1

}

vi /usr/local/nagios/etc/objects/localhost.cfg

define service {

use generic-service

host\_name pocnagios.novalocal.com

service\_description Service Status Nagios

check\_command check\_servicestatus

}

sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

systemctl restart nagios.service

***\*\*\*\*\*\*\*\*\*\*\*\*\* CENTOS NODE AND UBUNTU NODE CONFIGURATION \*\*\*\*\*\*\*\*\*\*\****

**How To Create Nagios Plugins With Bash On CentOS:**

<https://www.digitalocean.com/community/tutorials/how-to-create-nagios-plugins-with-bash-on-ubuntu-12-10>

<https://www.digitalocean.com/community/tutorials/how-to-create-nagios-plugins-with-bash-on-centos-6>

**Step :1** - Install RPMForge Repository and NRPE on client VPS

rpm -ivh http://pkgs.repoforge.org/rpmforge-release/rpmforge-release-0.5.3-1.el6.rf.x86\_64.rpm

yum -y install nagios-nrpe

useradd nrpe && chkconfig nrpe on ***(OR)*** yum install nagios\* -y; useradd nrpe

**Step :2** - Create your Bash Script

Centos (and) Ubuntu

cd /usr/lib64/nagios/plugins/ (and) /usr/lib/nagios/plugins/

[root@centosnode plugins]# cat servicestatus.sh

#!/bin/bash

status=dead

for services in httpd sshd

do

systemctl status $services | grep -i 'running\|dead' | awk '{print $3}' | sed 's/[()]//g' | while read output;

do

if [ "$output" == "$status" ]; then

echo "CRITICAL - $services Service is NOT Running"

curl -D - \

-X "POST" -H "Accept: application/json" \

-H "Content-Type: application/json" \

-H "X-Rundeck-Auth-Token: NSUt9PpUTTJ6INH7xcjt09VyMcr8cexX" \

-d "{\"argString\":\"-servicename $services \"}" \

http://192.168.5.116:4440/api/16/job/769de6f3-c068-45de-90ce-c636a9297488/executions

exit 2

else

echo "OK - All Services are Running"

exit 0

fi

done

done

chmod +x servicestatus.sh

***Step :3***

vi /etc/nagios/nrpe.cfg

allowed\_hosts=198.168.4.53

command[servicestatus \_bash]=/usr/lib64/nagios/plugins/ servicestatus.sh

service nrpe restart (and) service nagios-nrpe-server restart

**vi /usr/local/nagios/etc/objects/commands.cfg**

**Note: If name already exits Don’t add this**

# Custom plugins commands...

define command{

command\_name check\_warnings

command\_line $USER1$/check\_warnings.sh

}

**vi /usr/local/nagios/etc/server/centos.cfg**

define service {

use generic-service

host\_name centosnode

service\_description Service Status CentOS

check\_command servicestatus \_bash

} **(if Ubuntu)**

**vi /usr/local/nagios/etc/server/ubuntu.cfg**

define service {

use generic-service

host\_name centosnode

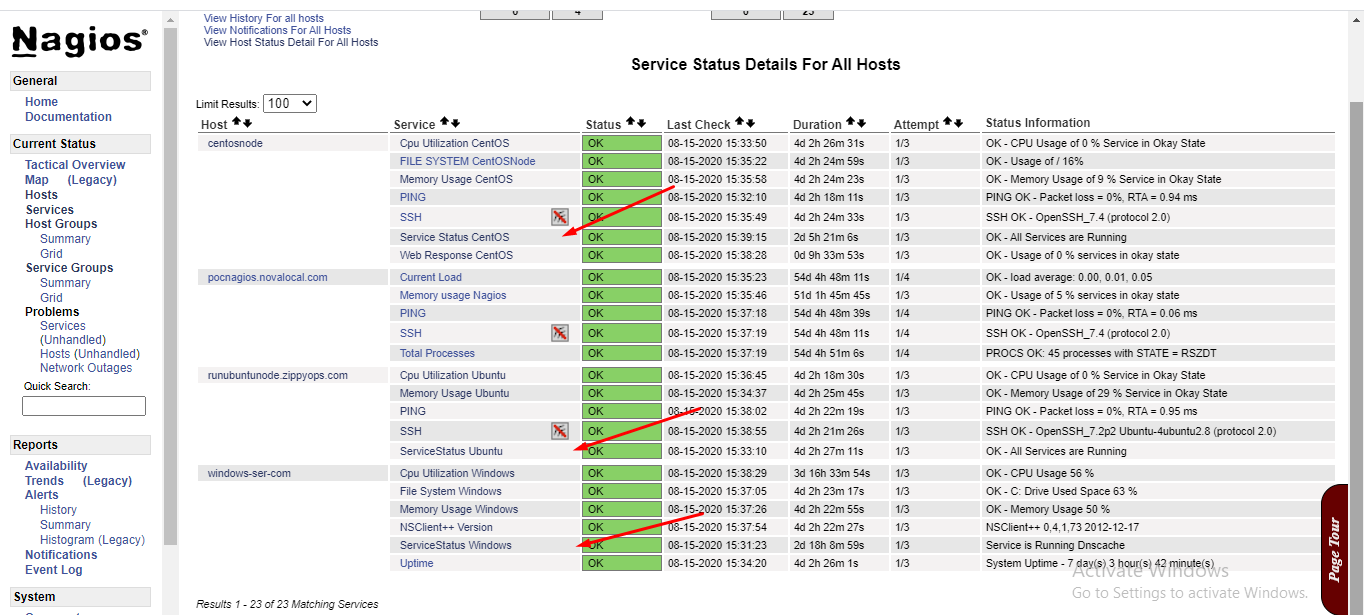
service\_description Service Status Ubuntu

check\_command servicestatus \_bash

}

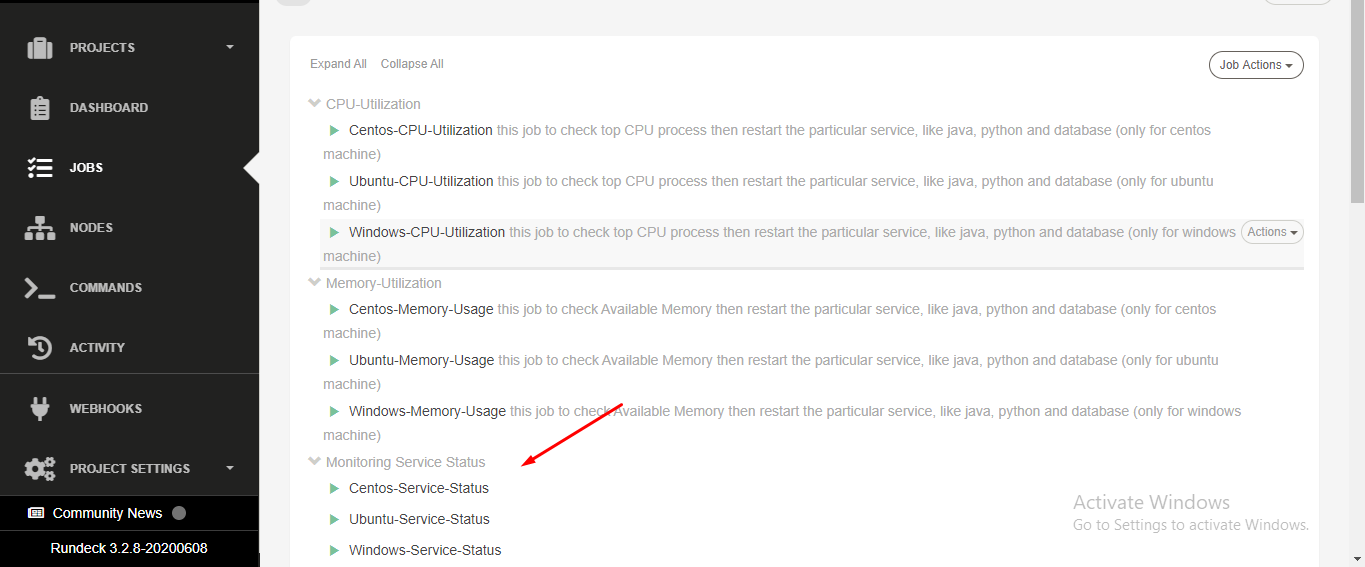
sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

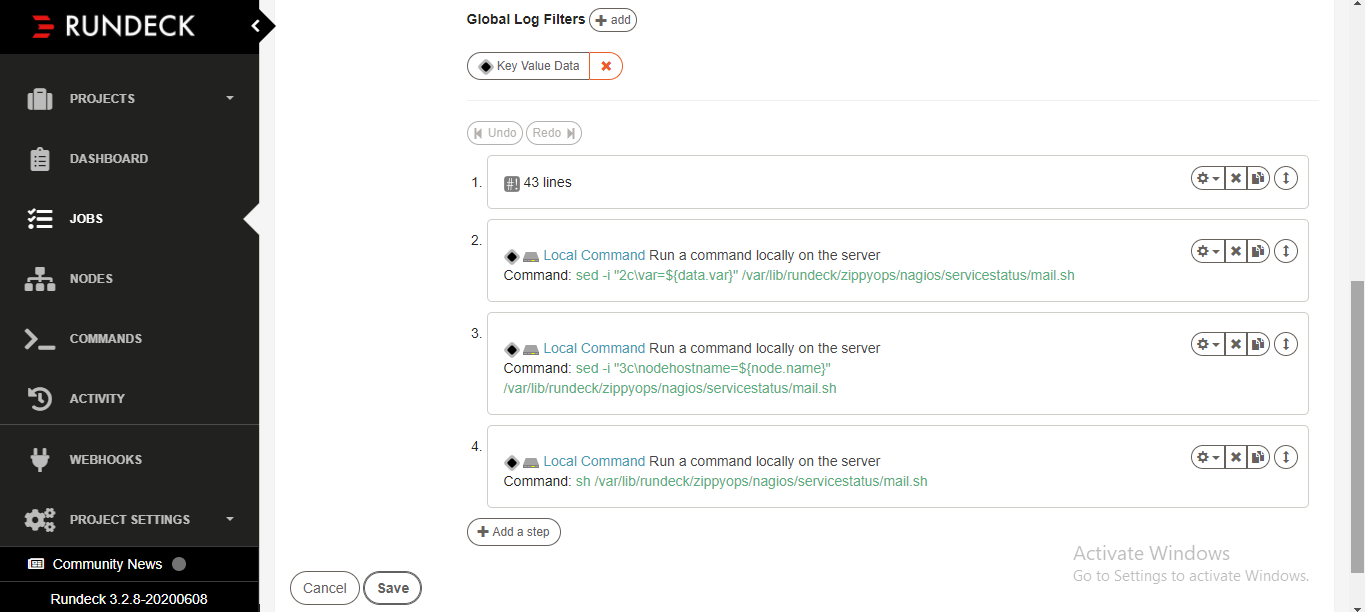
**systemctl restart nagios.service**

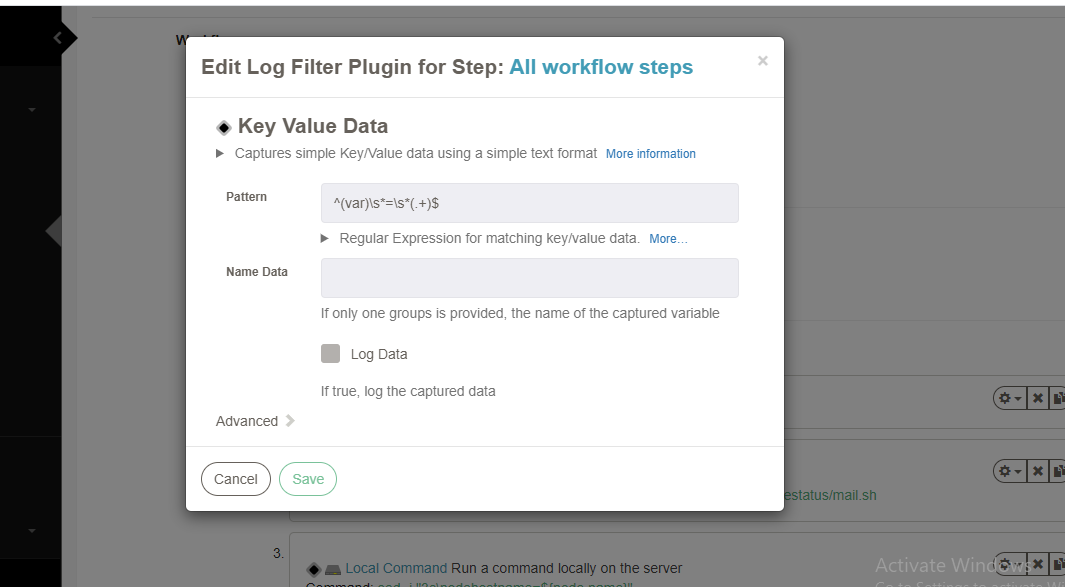


**\*\*\* Nagios and Rundeck jobs for linux \*\*\***

**------------------------------------------ TO Rundeck -------------------------------**

****

****

****

**\*\*\* In-line script for Linux \*\*\***

**-------------------------------------------------------------------------------------------------------------------------------------**

**nodeostype=@node.os-name@**

**SERVICE=@option.servicename@**

**echo "$nodeostype"**

**echo "$SERVICE"**

**if [ "$nodeostype" == "centos" ]; then**

**echo "This is linux machine 192.168.4.115"**

**if [ "$SERVICE" = "sshd" ]**

**then**

**systemctl restart sshd**

**var="$SERVICE"**

**echo "$var"**

**#sleep 10**

**echo "this is $SERVICE"**

**elif [ "$SERVICE" = "httpd" ]**

**then**

**systemctl restart httpd.service**

**var="$SERVICE"**

**echo "$var"**

**echo "this is $SERVICE"**

**elif [ "$SERVICE" = "mysqld" ]**

**then**

**systemctl restart mariadb**

**var="$SERVICE"**

**echo "$var"**

**echo "this is $SERVICE"**

**sleep 10**

**elif [ "$SERVICE" = "crond" ]**

**then**

**systemctl restart crond**

**var="$SERVICE"**

**echo "$var"**

**echo "this is $SERVICE"**

**sleep 10**

**else**

**echo "not for this"**

**var="$SERVICE"**

**echo "$var"**

**echo "this is $SERVICE"**

**fi**

**fi**

**echo "var=$var"**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Mail.sh common for windows and linux \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

\*\*\* Mail server configuration \*\*\*

<https://www.howtoforge.com/tutorial/write-a-custom-nagios-check-plugin/>

\*\*\* Mail server configuration \*\*\*

-------------------------------------------------------------------------------------------

yum install dnf -y;

dnf install postfix cyrus-sasl-plain mailx -y

sed -i 's/smtp\_tls\_security\_level = may/smtp\_tls\_security\_level = encrypt/' /etc/postfix/main.cf

echo "smtp\_tls\_security\_level = encrypt" >> /etc/postfix/main.cf

echo "smtp\_tls\_CAfile = /etc/pki/tls/certs/ca-bundle.crt" >> /etc/postfix/main.cf

cat >> /etc/postfix/main.cf << EOF

relayhost = [smtp.gmail.com]:587

smtp\_sasl\_auth\_enable = yes

smtp\_sasl\_password\_maps = hash:/etc/postfix/sasl\_passwd

smtp\_sasl\_security\_options = noanonymous

EOF

vi /etc/postfix/sasl\_passwd

[smtp.gmail.com]:587 alertsfromaiops@gmail.com:Zippyops@123

postmap /etc/postfix/sasl\_passwd

chown root:root /etc/postfix/sasl\_passwd\*

chmod 600 /etc/postfix/sasl\_passwd\*

systemctl enable postfix --now

echo "Test Postfix Gmail Relay" | mail -s "Postfix Gmail Relay" userid@gmail.com

tail -f /var/log/maillog

---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**Cat mail.sh**

**#!/bin/bash**

**var=**

**nodehostname=centosnode**

**mail=alerts@zippyops.in**

**if [ $nodehostname == centosnode ]; then**

**if [ $var == ssh ]; then**

**echo "$var service running on $nodehostname" | mail -s "$var Service Restarted on Centos node" $mail**

**elif [ $var == httpd ]; then**

**echo "$var service running on $nodehostname" | mail -s "$var Service Restarted on Centos node" $mail**

**elif [ $var == python ]; then**

**echo "$var service running on $nodehostname" | mail -s "$var Service Restarted on Centos node" $mail**

**elif [ $var != False ]; then**

**echo "$var service running on $nodehostname" | mail -s "$var Services Restarted on Centos node" $mail**

**else**

**echo "this is not $nodehostname"**

**exit 0**

**fi**

**elif [ $nodehostname == runubuntunode.zippyops.com ]; then**

**if [ $var == ssh ]; then**

**echo "$var service running on $nodehostname" | mail -s "$var Service Restarted on Ubuntu node" $mail**

**elif [ $var == java ]; then**

**echo "$var service running on $nodehostname" | mail -s "$var Service Restarted on Ubuntu node" $mail**

**elif [ $var == apache2 ]; then**

**echo "$var service running on $nodehostname" | mail -s "$var Service Restarted on Ubuntu node" $mail**

**elif [ $var != False ]; then**

**echo "$var service running on $nodehostname" | mail -s "$var Service Restarted on Ubuntu node" $mail**

**else**

**echo "this is not $nodehostname"**

**exit 0**

**fi**

**elif [ $nodehostname == windowsnode ]; then**

**if [ $var == System ]; then**

**echo "$var service running on $nodehostname" | mail -s "$var Service Restarted on Windows node" $mail**

**elif [ $var == httpd ]; then**

**echo "$var service running on $nodehostname" | mail -s "$var Service Restarted on Windows node" $mail**

**elif [ $var == python ]; then**

**echo "$var service running on $nodehostname" | mail -s "$var Service Restarted on Windows node" $mail**

**elif [ $var != False ]; then**

**echo "$var service running on $nodehostname" | mail -s "$var Service Restarted on Windows node" $mail**

**else**

**echo "this is not $nodehostname"**

**exit 0**

**fi**

**fi**

**--------------------------------- ####### &&&&&&&&&&& ##########-------------------------**

**On windows side configuration**

**\*\*\* NSClient++ \*\*\***

<https://www.linuxtechi.com/add-windows-linux-host-to-nagios-server/>

**\*\*\* Python \*\*\***

<https://www.youtube.com/watch?v=4Rx_JRkwAjY&list=LLxcM_ASJnlfXDjf4fkE_8QA&index=2&t=356s>

**After installed NSClient++ and Python**

**Step : 1**

**In addition to these settings, execute the follow commands on your windows server (in a command prompt) to ensure the External Scripts module is correctly loaded:**

cd "\Program Files\NSClient++\"

nscp settings --activate-module CheckExternalScripts --add-defaults

nscp settings --path "/settings/external scripts" --key "allow arguments" --set true

**# \*\*\* Power Shell scripts \*\*\***

Create New file

C:\Program Files\NSClient++\scripts with the name check\_ servicestatus.ps1

#Content of the check\_servicestatus.ps1 is

# Invoke python file

C:\Python\Python38\python.exe C:\Users\ servicestatus.py

#!/bin/bash/python3

import os

import psutil

import json

import wmi, sys, requests

#DNS Client(Dnscache),IP Helper(iphlpsvc),Diagnostic Policy Service(DPS).

google=[ "Dnscache", "iphlpsvc","DPS", ]

for redhat in google:

s=psutil.win\_service\_get(redhat)

if s.status() == 'running':

print("Service is Running", s.name() )

else:

print("Service is NOT Running", s.name() )

service=s.name()

print(service)

headers = {

'Accept': 'application/json',

'Content-Type': 'application/json',

'X-Rundeck-Auth-Token': 'NSUt9PpUTTJ6INH7xcjt09VyMcr8cexX',

}

data = '{"argString":"-servername windows-ser-com -filesystem service "}'

response = requests.post('http://192.168.5.116:4440/api/16/job/f8321b99-0370-4013-b9b7-58d88202bf0c/executions', headers=headers, data=data)

**Open a command prompt as an \*administrator\* on your Windows machine and execute the following command:**

\*\*\* Trail running \*\*\*

powershell.exe Set-ExecutionPolicy Bypass

cd "\Program Files\NSClient++\scripts"

powershell.exe -File check\_servicestatus.ps1

\*\*\* Update nsclient.ini Configuration File \*\*\*

# Open the file C:\Program Files\NSClient++\nsclient.ini in Notepad. \*\*\*

[/settings/external scripts/scripts]

check\_servicestatus\_ps1 = cmd /c echo scripts\\check\_servicestatus.ps1 | powershell.exe -command -

\*\*\* From Nagios server side \*\*\*

Test PowerShell Script From Nagios

/usr/local/nagios/libexec/check\_nrpe -H 192.168.1.3 -c check\_servicestatus\_ps1

echo $?

------------------------------------------------------------------------------------------------------------------------------

\*\*\* Update Command.cfg On nagios server side \*\*\*

/usr/local/nagios/etc/objects/command.cfg

define command {

command\_name check\_servicestatus\_ps1

command\_line $USER1$/check\_nrpe -H $HOSTADDRESS$ -c check\_servicestatus\_ps1

}

\*\*\* Update windows.cfg On nagios server side \*\*\*

/usr/local/nagios/etc/objects/windows.cfg

define service {

use generic-service

host\_name windows-ser-com

service\_description Service Status Windows

check\_command check\_servicestatus\_ps1

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**\*\*\*\*\*\*\*\*Job for Windows.ps1 \*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Connecting windows machines to rundeck.**

**wget https://github.com/rundeck-plugins/rundeck-winrm-plugin/releases/download/v1.3.4/rundeck-winrm-plugin-1.3.4.jar -P**

**/var/lib/rundeck/libext/**

**chown -R rundeck. /var/lib/rundeck/libext/**

**/etc/init.d/rundeckd restart**

**Step 2./ Configure a Windows Server for WinRM**

**– On the remote host, open a PowerShell using the Run as Administrator option and execute the following commands to enable the WinRM :**

**winrm qc**

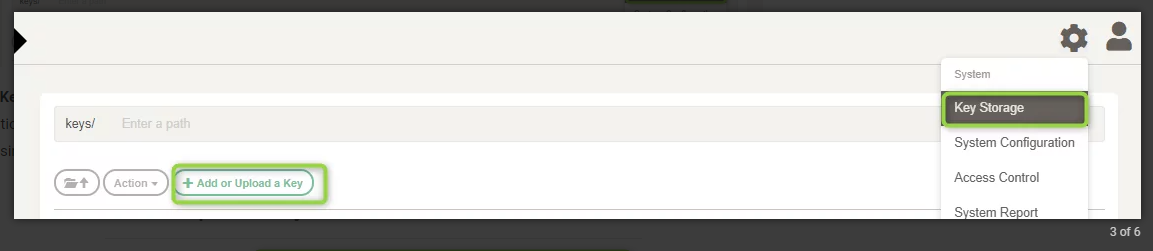
**Make these changes [y/n]? y**

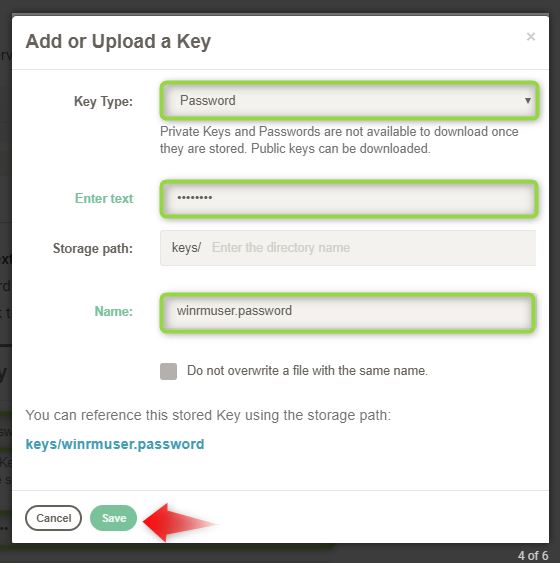
**winrm set winrm/config/service/Auth '@{Basic="true"}'**

**winrm set winrm/config/client/Auth '@{Basic="true"}'**

**winrm set winrm/config/service '@{AllowUnencrypted="true"}'**

**Get-Service -ComputerName windows.example.com -Name winrm | Select Status**

****

****

**NOTE: Edit this resource.xml file according to your server hostaddress and hostname.**

**[root@rundeck~]#vi /var/lib/rundeck/projects/nagios\_rundeck/etc/resource.xml**

**<?xml version="1.0" encoding="UTF-8"?>**

**<project>**

**<node name="192.168.43.10" description="Rundeck server node" tags="" hostname="192.168.43.10" osArch="amd64" osFamily="unix" osName="Linux" osVersion="3.10.0-1127.el7.x86\_64" username="rundeck"/>**

**<node name="Windows-Server-Machine" hostname="windows-ser-com" osName="Windows" username="Administrator" ps-authentication-type="CredSSP" winrm-protocol="http" winrm-cmd="powershell" winrm-password-storage-path="keys/windowspasswd" file-copier="WinRMcpPython" node-executor="WinRMPython"/>**

**</project>**

**-----------------------------------------------------------------------------------------------------------------------------------------**

**pip install requests && pip install urllib3**

**yum -y install python-pip**

**curl "https://bootstrap.pypa.io/get-pip.py" -o "get-pip.py"**

**python get-pip.py**

**pip -V**

**pip 1.3.1 from /usr/lib/python2.7/site-packages (python 2.7)**

**pip install "pywinrm>=0.3.0"**

**cd /var/lib/rundeck/libext/**

**unzip py-winrm-plugin-2.0.6.zip**

**chown rundeck:rundeck py-winrm-plugin-2.0.6**

**pip install pywinrm**

**vi /etc/hosts**

**192.168.4.53 pocnagios.example.com**

**-----------------------------------------------------------------------------------------------------------------------------------------**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* servicestatus.ps1 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**[root@tamilrundeck memoryusage]# cat windows.py**

**#!/bin/bash/python3**

**import os**

**import psutil**

**import subprocess**

**import wmi, sys**

**#DNS Client(Dnscache),IP Helper(iphlpsvc),Diagnostic Policy Service(DPS),Optimize drives(defragsvc),Phone Service(PhoneSvc).**

**google=[ "Dnscache", "iphlpsvc","DPS","defragsvc","PhoneSvc"]**

**for redhat in google:**

**service=psutil.win\_service\_get(redhat)**

**if (service.status() != 'running') and (service.name() == 'defragsvc'):**

**subprocess.Popen('powershell.exe Restart-Service -Name defragsvc')**

**print("Service is NOT Running", service.name() )**

**elif (service.status() != 'running') and (service.name() == 'DPS'):**

**subprocess.Popen('powershell.exe Restart-Service -Name dps')**

**print("Service is NOT Running", service.name())**

**elif (service.status() != 'running') and (service.name() == 'Dnscache'):**

**subprocess.Popen('powershell.exe Restart-Service -Name Dnscache')**

**print("Service is NOT Running", service.name())**

**elif (service.status() != 'running') and (service.name() == 'iphlpsvc'):**

**subprocess.Popen('powershell.exe Restart-Service -Name iphlpsvc')**

**print("Service is NOT Running", service.name())**

**else:**

**print("Service is Running", service.name()) }**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*THE END\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***