Restart the xinetd service to start NRPE:
sudo service xinetd restart

Configure Nagios

Open the Nagios configuration file:

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

#cfg_dir=/usr/local/nagios/etc/servers

#cfg_dir=/usr/local/nagios/etc/printers

#cfg_dir=/usr/local/nagios/etc/switches

#cfg_dir=/usr/local/nagios/etc/routers

Uncomment all the devices you will be monitoring, and save the file.

Now create the directories for all the devices you will be monitoring in each of those paths above: for example type sudo mkdir /usr/local/nagios/etc/servers for the servers.

Configure Nagios Contacts

In order to receive notifications from Nagios you need to configure the contacts. open the contact configuration file, and add the email address you want to receive the emails from Nagios.

Find the email directive, and replace its value nagios@localhost with our own email address nagiosadmin@gmail.com (Taken Randomly)

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

email nagios@localhost ; <<***** CHANGE THIS TO YOUR EMAIL ADDRESS ******

```
##############################################################################

# Just one contact defined by default - the Nagios admin (that's you)
# This contact definition inherits a lot of default values from the 'generic-contact'
# template which is defined elsewhere.

define contact{
        contact_name                    nagiosadmin             ; Short name of user
        use                             generic-contact         ; Inherit default val
        alias                           Nagios Admin            ; Full name of user

        email                           nagiosadmin@gmail.com   ; <<***** CHANGE THIS
        }



##############################################################################
##############################################################################
#
# CONTACT GROUPS
#
##############################################################################
##############################################################################

# We only have one contact in this simple configuration file, so there is
# no need to create more than one contact group.

define contactgroup{
        contactgroup_name       admins
        alias                   Nagios Administrators
        members                 nagiosadmin
        }
```

In the screenshot, we also seen a contact group created to whom nagios should send notifications. The contact “nagiosadmin” is added to the contact group named “admins”

Configure the NRPE command

Open the command configuration file sudo nano /usr/local/nagios/etc/objects/commands.cfg and add the following command at the end of the file:

define command {

command_name check_nrpe

command_line $USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$

}

```
# 'process-host-perfdata' command definition
define command{
        command_name    process-host-perfdata
        command_line    /usr/bin/printf "%b" "$LASTHOSTCHECK$\t$HOSTNAME$\t$HOSTSTATE$\t$HOSTATTEMPT$
        }


# 'process-service-perfdata' command definition
define command{
        command_name    process-service-perfdata
        command_line    /usr/bin/printf "%b" "$LASTSERVICECHECK$\t$HOSTNAME$\t$SERVICEDESC$\t$SERVICE
        }

define command{
        command_name check_nrpe
        command_line $USER1$/check_nrpe -H $HOSTADDRESS$ -c $ARG1$
}
```

Configure Apache

Enable mod_rewrite

sudo a2enmod rewrite

enable the CGI module

sudo a2enmod cgi

Create the default Nagios account to login to the Nagios using this command,

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

enter the password when prompted:

```
nagios@coreserve :~$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
```

Create a symbolic link from the nagios.conf file to the apache enabled sites:

sudo ln -s /etc/apache2/sites-available/nagios.conf /etc/apache2/sites-enabled/

Start Nagios and restart Apache

sudo service nagios start

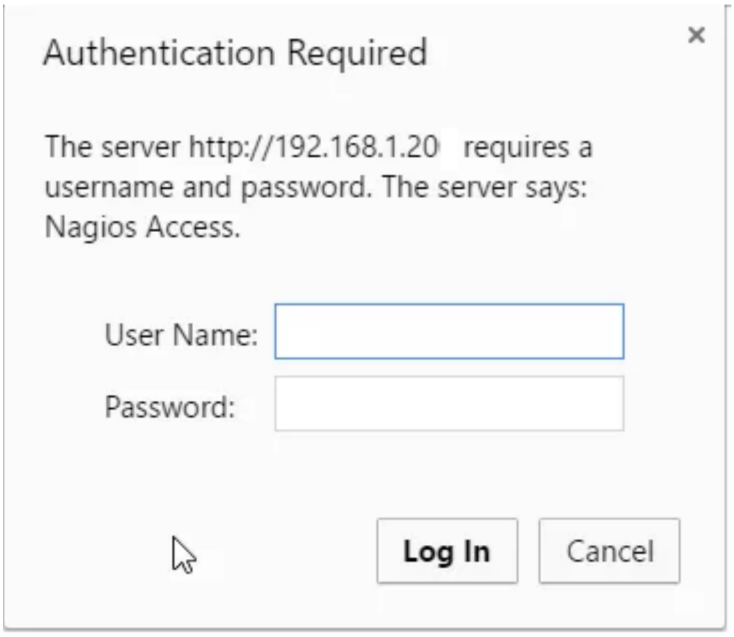
sudo service apache2 restart

To start Nagios at every server reboot, create this symbolic link

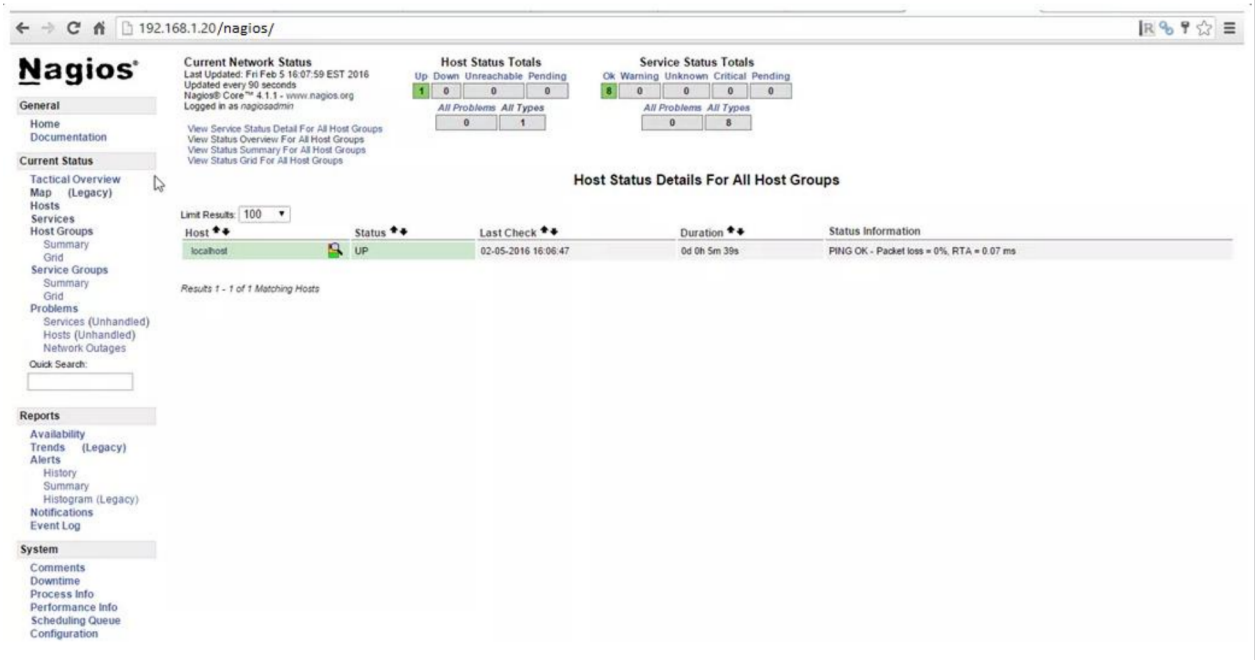
sudo ln -s /etc/init.d/nagios /etc/rcS.d/S99nagios

7. Login to Nagios

Kudos! we have successfully installed, and configured Nagios. type the IP address of your Nagios server on your browser with the /nagios sub-directory like this http://ipaddress/nagios and you should get the login prompt:



Type the user name we created above “nagiosadmin” and the password you chose.and you should land in your newly built Nagios web page.



8. Monitor a Linux Host with NRPE

In this section, we'll show you how to add a new host to Nagios, so it will be monitored. Repeat this section for each server you wish to monitor.

On a server that you want to monitor,

sudo apt-get update

Now install Nagios Plugins and NRPE,

sudo apt-get install nagios-plugins nagios-nrpe-server

Configure Allowed Hosts

Now, let's update the NRPE configuration file. Open it in your favourite editor (we're using vi):

sudo vi /etc/nagios/nrpe.cfg

Find the allowed_hosts directive, and add the private IP address of your Nagios server to the comma-delimited list (substitute it in place of the highlighted example):

allowed_hosts=127.0.0.1,192.168.1.20

Save and exit. This configures NRPE to accept requests from your Nagios server, via its private IP address.

Configure Allowed NRPE Commands

Look up the name of your root filesystem (because it is one of the items we want to monitor):

df -h /

We will be using the filesystem name in the NRPE configuration to monitor your disk usage (it is probably /dev/vda). Now open nrpe.cfg for editing:

sudo vi /etc/nagios/nrpe.cfg

The NRPE configuration file is very long and full of comments. There are a few lines that you will need to find and modify:

server_address: Set to the private IP address of this host
allowed_hosts: Set to the private IP address of your Nagios server
command[check_hda1]: Change /dev/hda1 to whatever your root filesystem is called

The three aforementioned lines should look like this (substitute the appropriate values):

server_address=client_private_IP
allowed_hosts=nagios_server_private_IP
command[check_hda1]=/usr/lib/nagios/plugins/check_disk -w 20% -c 10% -p /dev/vda

Note that there are several other "commands" defined in this file that will run if the Nagios server is configured to use them. Also note that NRPE will be listening on port 5666 because server_port=5666 is set. If you have any firewalls blocking that port, be sure to open it to your Nagios server.

Save and quit.

Restart NRPE

Restart NRPE to put the change into effect:

sudo service nagios-nrpe-server restart

Once you are done installing and configuring NRPE on the hosts that you want to monitor, you will have to add these hosts to your Nagios server configuration before it will start monitoring them.

Add Host to Nagios Configuration

On your Nagios server, create a new configuration file for each of the remote hosts that you want to monitor in /usr/local/nagios/etc/servers/. Replace the highlighted word, "yourhost", with the name of your host:

sudo vi /usr/local/nagios/etc/servers/testlinux.com.cfg

```
define host {
use                     linux-server
host_name                       testlinux.com
alias                   Linux server for this forevergeeks.com tutorial
address                 192.168.1.100
contact_groups          admins
}
define service {
use                     generic-service
host_name                       testllinux.com
service_description     PING
check_command           check_ping!100.0,20%!500.0,60%
}
define service {
use                     generic-service
host_name               testlinux.com
service_description     PING
check_command           check_ping!100.0,20%!500.0,60%
}
define service {
use                             generic-service
host_name                       testlinux.com
service_description             SSH
check_command                   check_ssh
notifications_enabled           0
}
define service{
use                     generic-service
host_name               testlinux.com
service_description     Total Processes
```

```
check_command       check_nrpe!check_total_procs
}
define service{
use                         generic-service
host_name                   testlinux.com
service_description         Root Partition
contact_groups              admins
check_command               check_nrpe!check_disk
}
define service{
use                 generic-service
host_name           testlinux.com
service_description Load
check_command       ccheck_nrpe!check_load!5!10
}
```

In the configuration file above we setup to monitor a Ping, Load, Number of processes, SSH, Disk space:

9. Directives

host_name: This directive is used to define a short name used to identify the host. It is used in host group and service definitions to reference this particular host. Hosts can have multiple services (which are monitored) associated with them. When used properly, the $HOSTNAME$ macro will contain this short name.

alias: This directive is used to define a longer name or description used to identify the host. It is provided in order to allow you to more easily identify a particular host. When used properly, the $HOSTALIAS$ macro will contain this alias/description

address: This directive is used to define the address of the host. Normally, this is an IP address, although it could really be anything you want (so long as it can be used to check the status of the host). You can use a FQDN to identify the host instead of an IP address, but if DNS services are not available this could cause problems. When used properly, the $HOSTADDRESS$ macro will contain this address. Note: If you do not specify an address directive in a host definition, the name of the host will be used as its address. A word of caution about doing this, however - if DNS fails, most of your service checks will fail because the plugins will be unable to resolve the host name.

check_command: This directive is used to specify the *short name* of the command that should be used to check if the host is up or down. Typically, this command would try and ping the host to see if it is "alive". The command must return a status of OK (0) or Nagios will assume the host is down. If you leave this argument blank, the host will *not* be actively checked. Thus, Nagios will likely always assume the host is up (it may show up as being in a "PENDING" state in the web interface). This is useful if you are monitoring printers or other devices that are frequently turned off. The maximum amount of time that the notification command can run is controlled by the host_check_timeout option.

check_interval: This directive is used to define the number of "time units" between regularly scheduled checks of the host. Unless you've changed the interval_length directive from the default value of 60, this number will mean minutes.

contact_groups: This is a list of the *short names* of the contact groups that should be notified whenever there are problems (or recoveries) with this host. Multiple contact groups should be separated by commas. You must specify at least one contact or contact group in each host definition.

notification_period: This directive is used to specify the short name of the time period during which notifications of events for this host can be sent out to contacts. If a host goes down, becomes unreachable, or recoveries during a time which is not covered by the time period, no notifications will be sent out.

contacts: This is a list of the *short names* of the contacts that should be notified whenever there are problems (or recoveries) with this host. Multiple contacts should be separated by commas. Useful if you want notifications to go to just a few people and don't want to configure contact groups. You must specify at least one contact or contact

group in each host definition.

active_checks_enabled : This directive is used to determine whether or not active checks (either regularly scheduled or on-demand) of this host are enabled. Values: 0 = disable active host checks, 1 = enable active host checks (default).

passive_checks_enabled : This directive is used to determine whether or not passive checks are enabled for this host. Values: 0 = disable passive host checks, 1 = enable passive host checks (default).

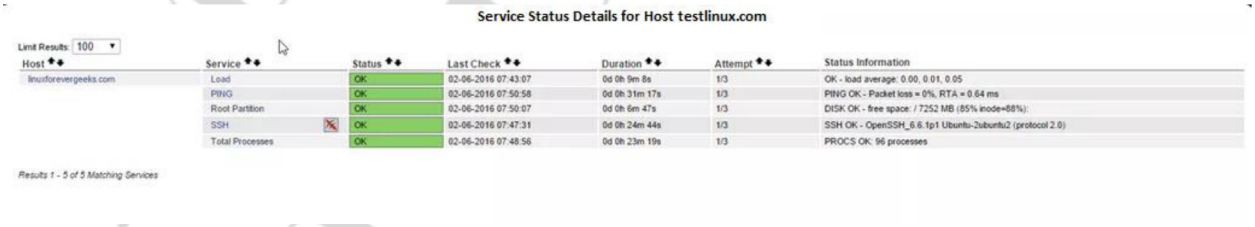
While defining the hots and services we have to use different kinds of parameters. To get a detailed information of them, go through https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/3/en/objectdefinitions.html

If you're not sure what use generic-service means, it is simply inheriting the values of a service template called "generic-service" that is defined by default.

Now save and quit. Reload your Nagios configuration to put any changes into effect:

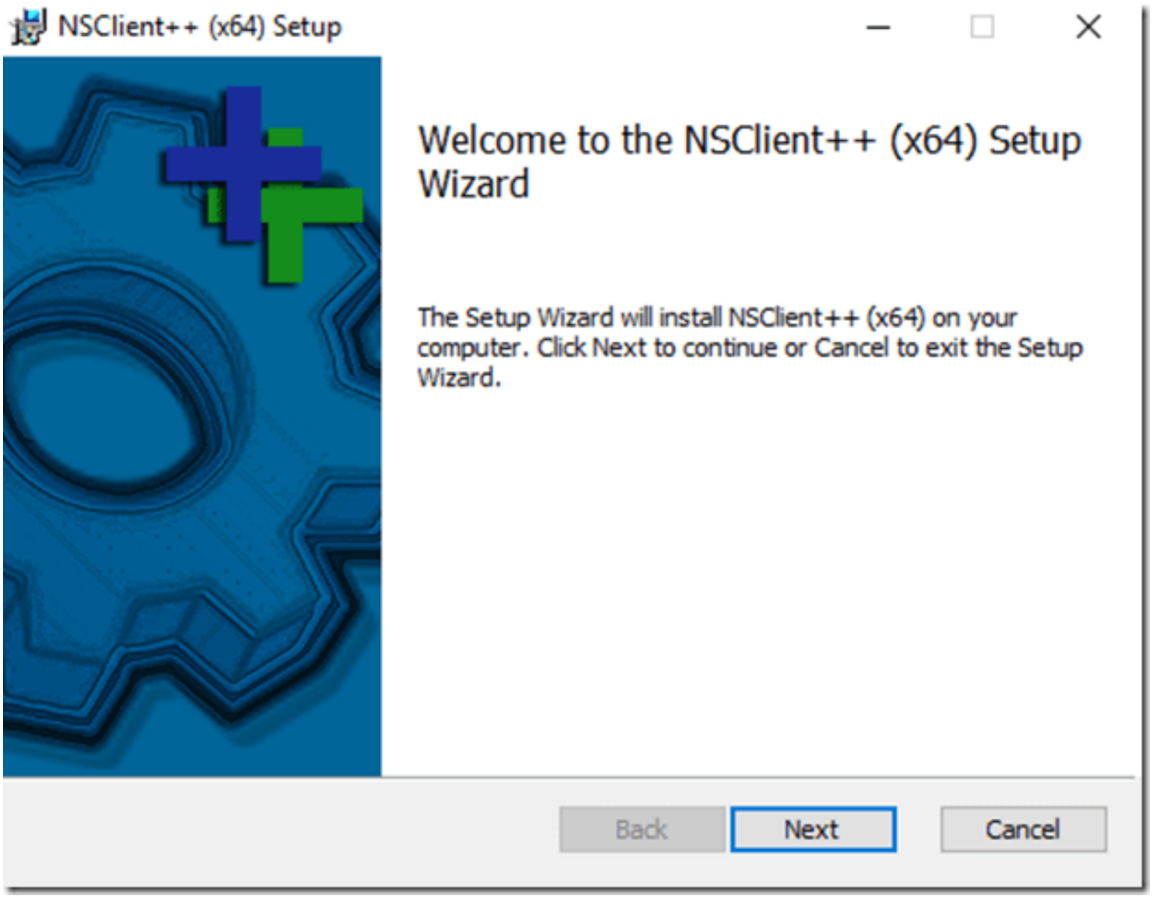
sudo service nagios reload

Once you are done configuring Nagios to monitor all of your remote hosts, you should be set. Be sure to access your Nagios web interface, and check out the Services page to see all of your monitored hosts and services:



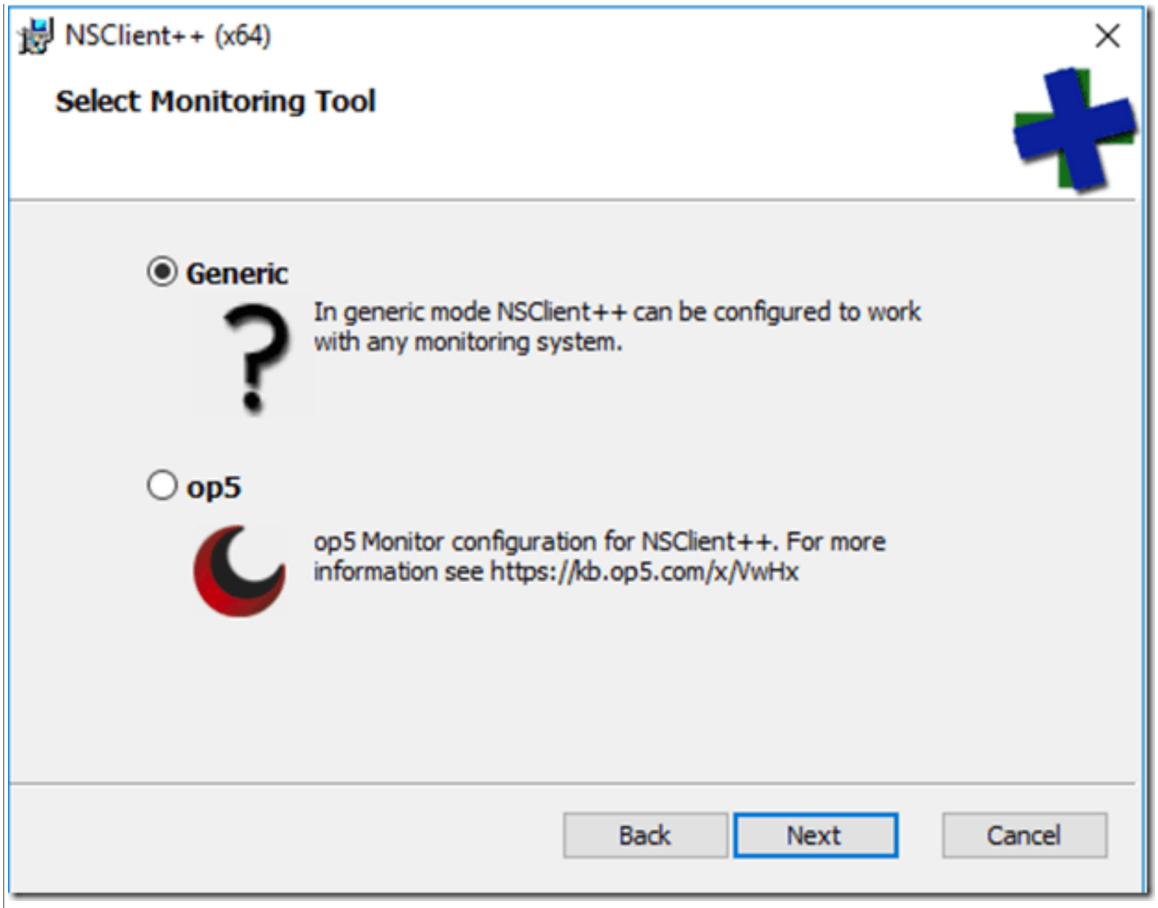
10. Monitoring a Windows Host

In order to monitor a Windows host either a server or desktop, we need to download and install NSclient++ in that computer. We can download the latest NSClient++ from this URL https://www.nsclient.org/download/ .download it, and execute it on the server you want to monitor. We are going to add a Windows 2016 server to our nagios server. after executing the NSClient+ we should get the installation wizard.

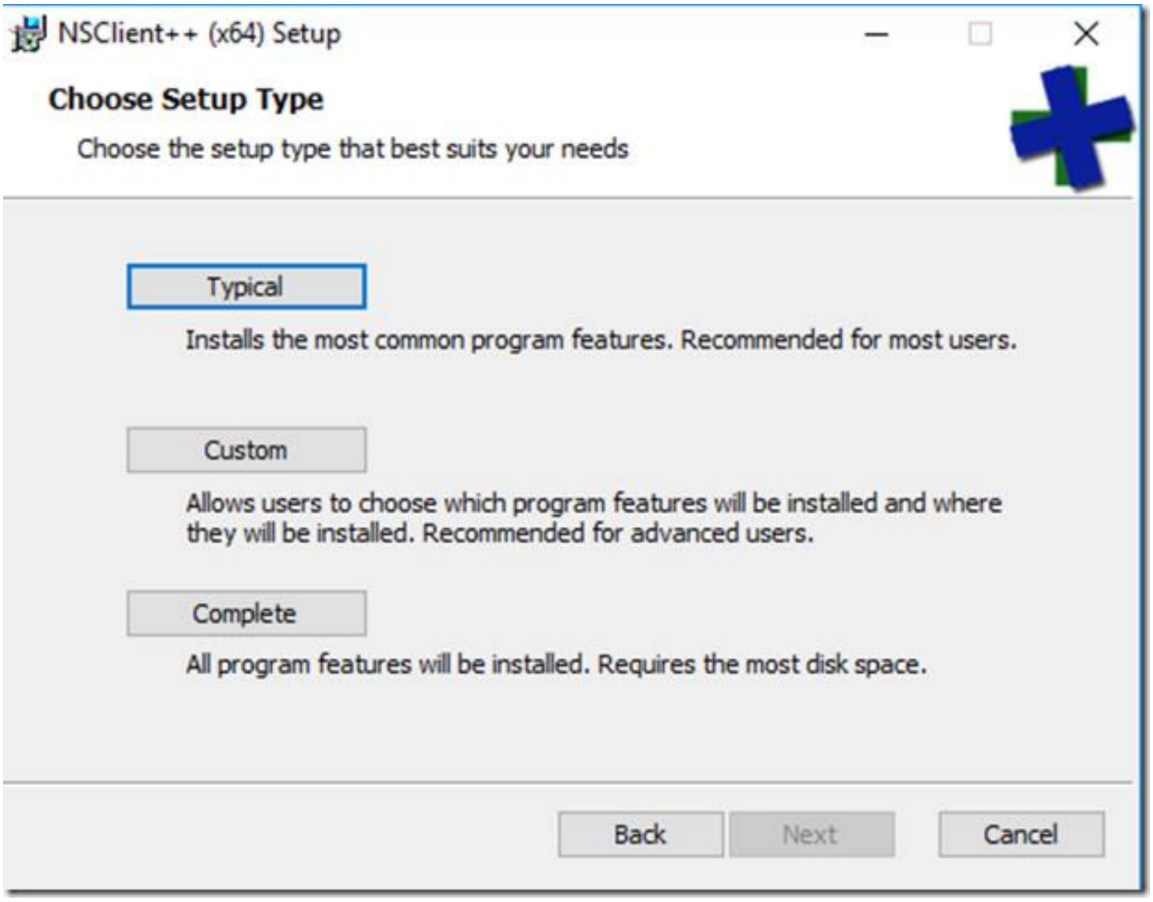


Click on Next.

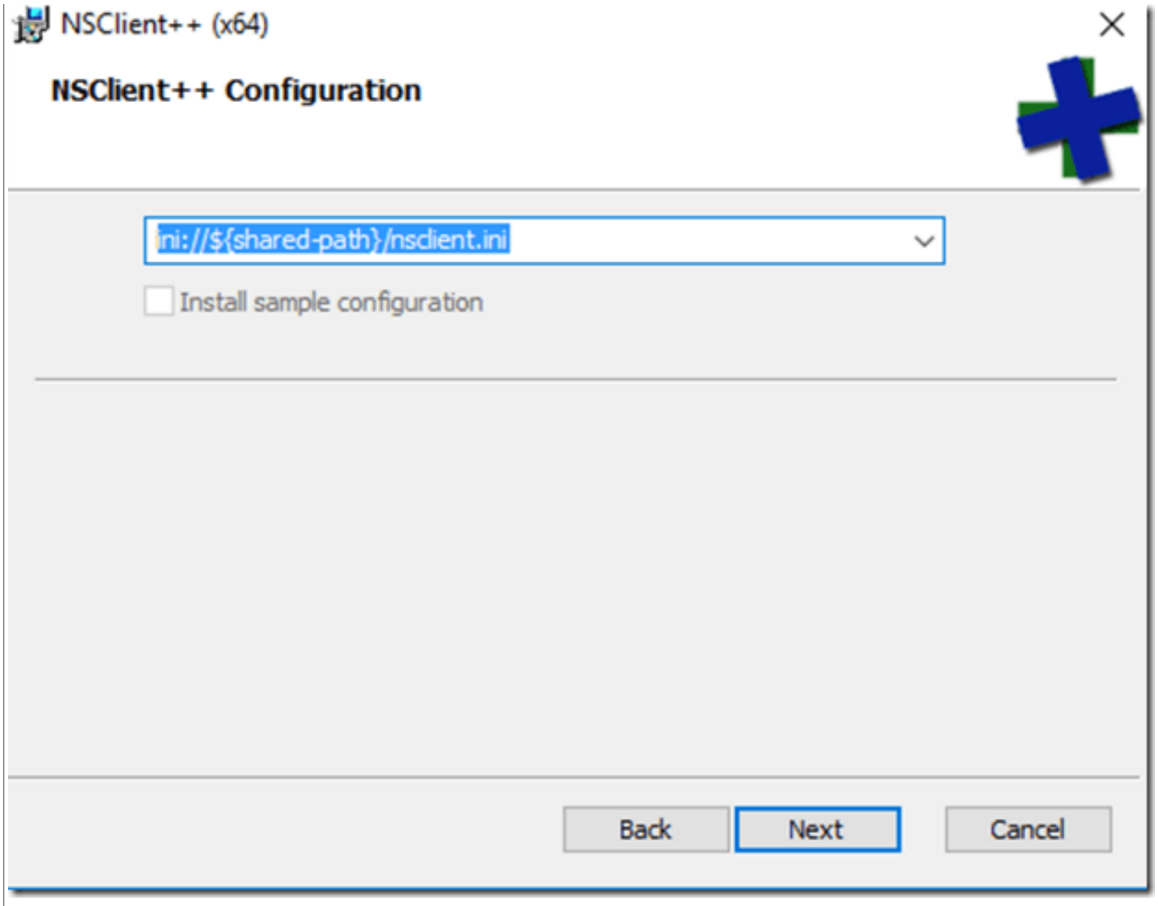
Choose Generic on the next screen:



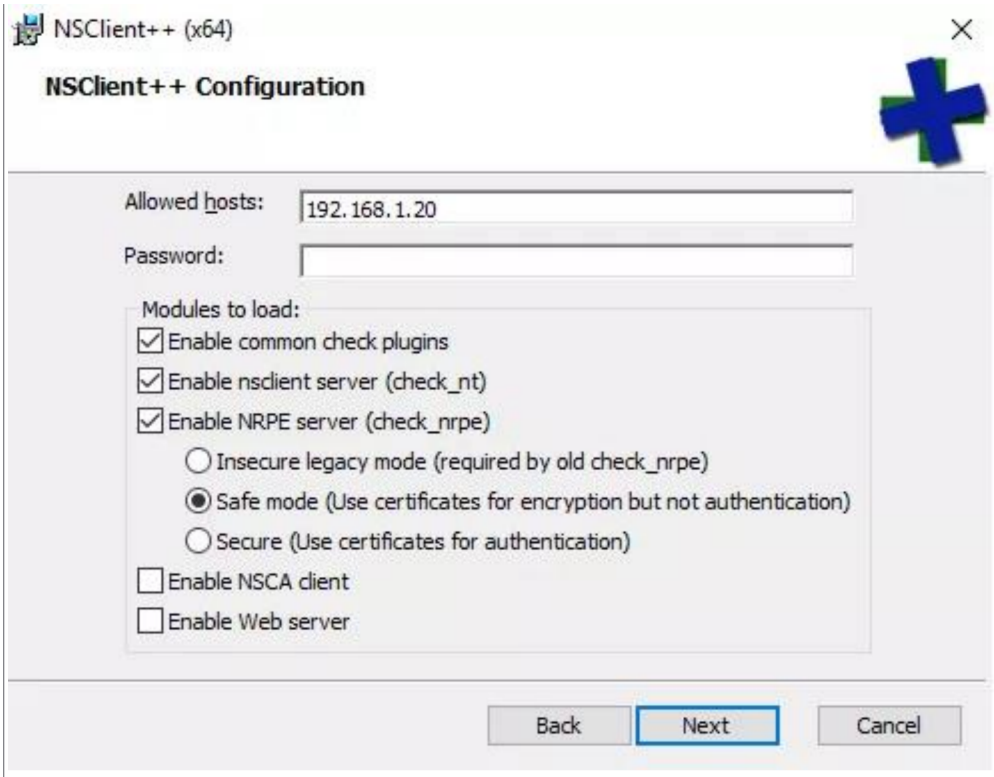
On the next screen click on “Complete” if you are concerned about security, and what to pick in choose what modules and scripts to install, feel free to choose the custom option.



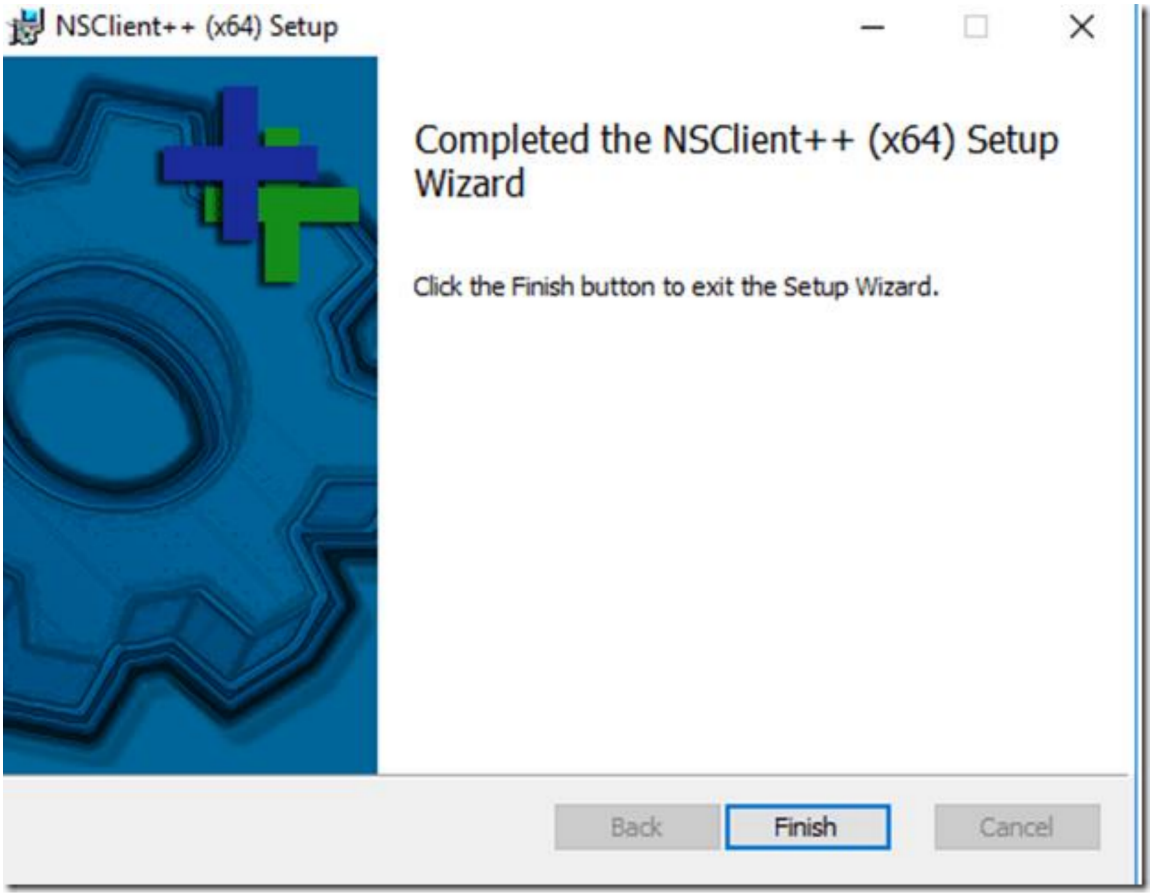
Click on Next in the following screen:



On the next screen enter the IP address of your Nagios server, and a password if you need to use a password:

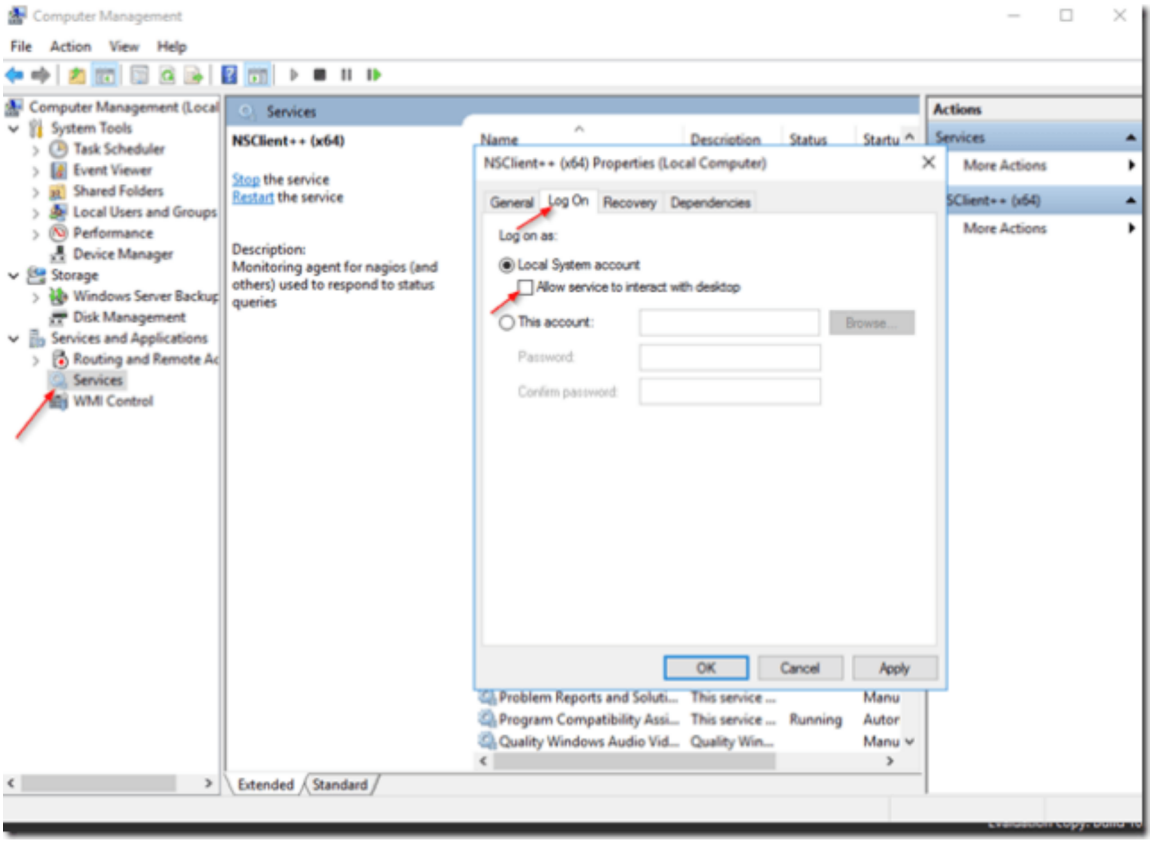


click on Next after you are done making your choices. then click on Install in the next screen.



then click on Finish.

After you finish installing the client, go to your server Services and right-click on NSClient++ service, and click on the Log On tab and check the Allow service to interact with the Desktop option:



Click on OK and make sure the service is running. you are done installing NSClient++ on the server.

Setting up the Windows config file

Let's create the configuration file the server. I will call my server "testwindows.com". create the file using nano with this command:

sudo nano /usr/local/nagios/etc/servers/testwindows.com.cfg

add this information to the file. (change it to your server information)

################ Windows Host Group #################

define hostgroup{

hostgroup_name windows-servers ; The name of the hostgroup

alias Windows Servers ; Long name of the group

}

################ Windows Server#########################

define host {

use windows-server

host_name testwindows.com

alias windows server for this forevergeeks.com tutorial

address 192.168.1.2

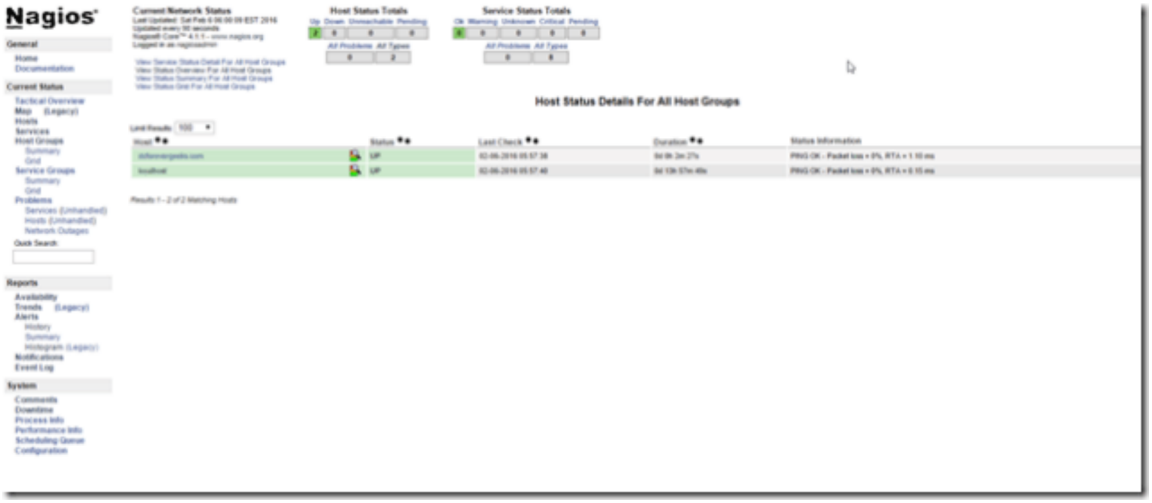
contact_groups admins

}

Reload the nagios service

```
sudo service nagios reload
```

and your server should show up under “Hosts” in Nagios:



Here Nagios is just pinging the server, but let’s monitor the server load, Uptime, memory usage, CPU load, and Hard drive space on that server as well. Open windows server configuration file and add this information.

define service{

use generic-service

host_name testwindows.com

service_description Uptime

check_command check_nt!UPTIME

contact_groups admins

}

define service{

use generic-service

host_name	testwindows.com
service_description	CPU Load
check_command	check_nt!CPULOAD!-l 5,80,90
contact_groups	admins

}

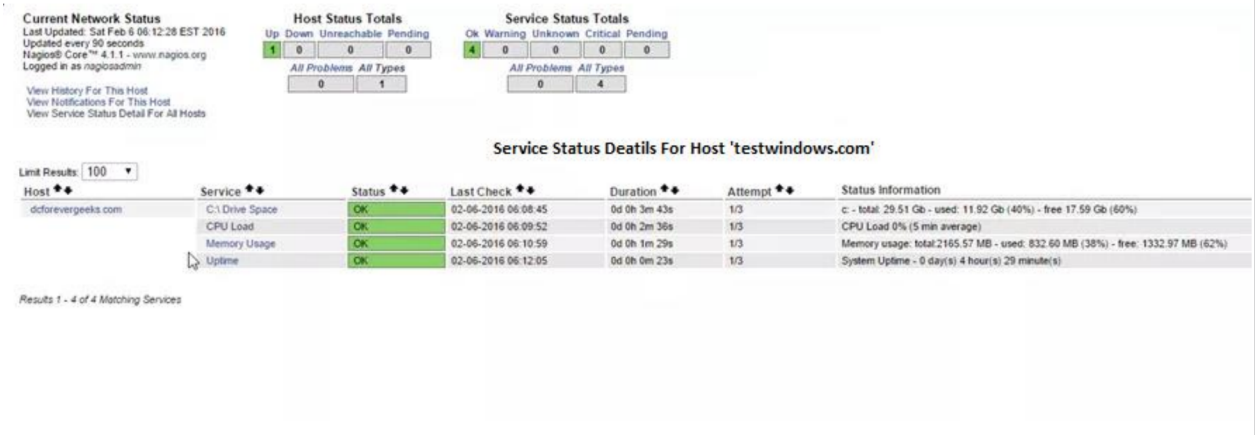
define service{	
use	generic-service
host_name	testwindows.com
service_description	Memory Usage
check_command	check_nt!MEMUSE!-w 80 -c 90
Contact_groups	admins

}

define service{	
use	generic-service
host_name	testwindows.com
service_description	C: Drive Space
check_command	check_nt!USEDDISKSPACE!-l c -w 80 -c 90
Contact_groups	admins

}

Reload the Nagios service sudo service nagios reload and those services should show up under your host now. if you get any error while reloading the Nagios service, check for typos.



Summary:

- Nagios is a powerful tool that provides you with instant awareness of your organization's mission-critical IT infrastructure. Nagios allows you to detect and repair problems and mitigate future issues before they affect end-users and customers.
- A LAMP stack is required to work on this monitoring tool. LAMP- Linux, Apache, Mysql and php.
- The complete process involves installation of the dependencies, Nagios core, Nagios plugins, NRPE (**Nagios** Remote Plugin Executor). Nagios core and Nagios plugins should be installed on the machine on which we want to see Nagios web interface. NRPE will be installed on remote machines which are going to be monitored by Nagios.
- Configure the Nagios to monitor different devices like servers, printers, routers and switches through nagios.cfg file and contacts also be defined for the notification purpose.
- Configure and allow the NRPE commands and add the host to the Nagios configuration.
- Nagios will be able to monitor the services on the remote machines after only defining them in the nagios configuration. Use Object Directives to define the services to be monitored. For detailed info for objectives go through, https://assets.nagios.com/downloads/nagioscore/docs/nagioscore/3/en/objectdefinitions.html
- Nagios can monitor the windows machines also. NSClient++ to be installed and configured on the windows machine. It functions similar to NRPE on linux machines.

Conclusion:

Every IT industry must need a monitoring tool to check the status of the services on the remote machines. This tool enables us to avoid any down time. A DevOps engineer must have knowledge on any monitoring tool. Nagios serve the same purpose with high efficiency.
Work on Nagios configuration files and Directives to dig deep into it.