[Installing and configuring PuppetDB on CentOS 7](https://geekdudes.wordpress.com/2018/11/23/installing-and-configuring-puppetdb-on-centos7/" \o "Permanent link to Installing and configuring PuppetDB on CentOS 7)

Posted: November 23, 2018 in [**Linux**](https://geekdudes.wordpress.com/category/linux/), [**puppet**](https://geekdudes.wordpress.com/category/puppet/)

[**0**](https://geekdudes.wordpress.com/2018/11/23/installing-and-configuring-puppetdb-on-centos7/#respond)

PuppetDB is open source storage service for Puppet nodes

**Installation**

In this example postgresql 11 is installed

rpm -Uvh <https://download.postgresql.org/pub/repos/yum/11/redhat/rhel-7-x86_64/pgdg-centos11-11-2.noarch.rpm>  
yum install postgresql11-server postgresql11-contrib

Initialize postgresql

/usr/pgsql-11/bin/postgresql-11-setup initdb

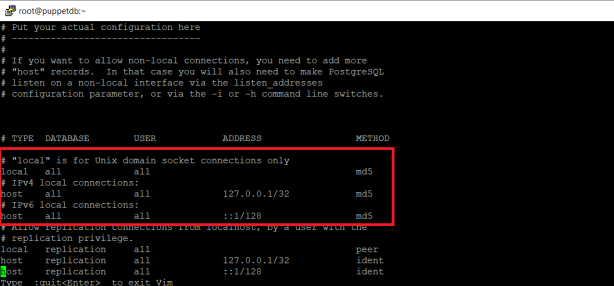
Start PostrgeSQL service

systemctl enable postgresql-11.service  
systemctl start postgresql-11.service

Switch to postgress user, create user puppetdb and puppetdb database

sudo -iu postgres  
createuser -DRSP puppetdb  
createdb -E UTF8 -O puppetdb puppetdb  
psql puppetdb -c 'create extension pg\_trgm'

Edit  /var/lib/pgsql/11/data/pg\_hba.conf



Install puppetdb

rpm -Uvh <https://yum.puppetlabs.com/puppet5/puppet5-release-el-7.noarch.rpm>  
yum install puppetdb

Edit /etc/puppetlabs/puppetdb/conf.d/database.ini, specify puppetdb username/password

[database]

classname = org.postgresql.Driver

subprotocol = postgresql

# The database address, i.e. //HOST:PORT/DATABASE\_NAME

subname = //localhost:5432/puppetdb

# Connect as a specific user

username = puppetdb

# Use a specific password

password = puppetdb

Edit /etc/puppetlabs/puppetdb/conf.d/jetty.ini

Uncomment host = 0.0.0.0

REPORT THIS AD

Edit /etc/sysconfig/puppetdb and re-map memory needed for puppetdb  
  
JAVA\_ARGS="-Xmx192m

Start puppetdb

systemctl start puppetdb && systemctl enable puppetdb

**Setting Puppet server**

make sure puppetb DNS name is resolveable (/etc/hosts)

Edit  /etc/puppetlabs/puppet/puppet.conf,add following lines

[master]

storeconfigs = true

storeconfigs\_backend = puppetdb

Create /etc/puppetlabs/puppet/puppetdb.conf

[main]  
server\_urls = <https://puppetdb.example.com:8081/>

Create /etc/puppetlabs/puppet/routes.yaml

---

master:

facts:

terminus: puppetdb

cache: yaml

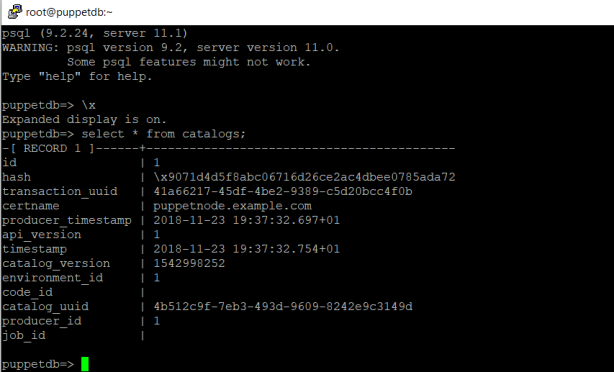
install puppetdb-termini and restart puppet server

yum install puppetdb-termini  
systemctl restart puppetserver

On puppet node run puppet -t

Login to puppetdb and verify data from node are transfered to puppetdb

psql -h localhost puppetdb puppetdb  
puppetdb=>\x  
puppetdb=>select \* from catalogs;



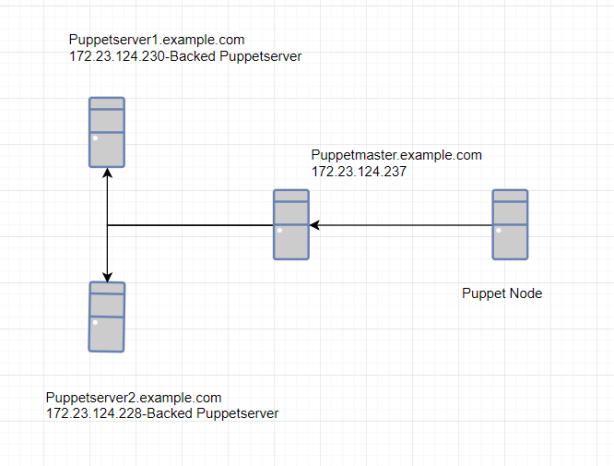
Advertisements

REPORT THIS AD

[Puppet Load Balancing](https://geekdudes.wordpress.com/2018/11/22/puppet-load-balancing/)

Posted: November 22, 2018 in [**Linux**](https://geekdudes.wordpress.com/category/linux/), [**puppet**](https://geekdudes.wordpress.com/category/puppet/)

[**0**](https://geekdudes.wordpress.com/2018/11/22/puppet-load-balancing/#respond)



Puppetmaster is Load Balancer,SSL termination happens there, Puppet client communicates only with puppetmaster, and puppetmaster sends requests to puppetservers1/2

**Settings puppetmaster,puppetservers 1 and 2**

rpm -Uvh <https://yum.puppetlabs.com/puppet5/puppet5-release-el-7.noarch.rpm> yum -y install puppetserver

export PATH=/opt/puppetlabs/bin:$PATH

source ~/.bash\_profile

edit /etc/puppetlabs/puppetserver/conf.d/webserver.conf.This will configure puppetservers to listen on port 8141 for TLS encrypted traffic and port 18140 for unencrypted traffic.

webserver: {

access-log-config: /etc/puppetlabs/puppetserver/request-logging.xml

client-auth: want

ssl-host: 0.0.0.0

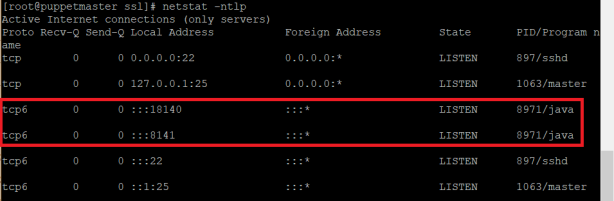
ssl-port: 8141

host: 0.0.0.0

port: 18140

}

On all machines run netstat -nltp to make sure ports 18140/8141 are opened



Edit /etc/puppetlabs/puppet/puppet.conf

[main]

certname=puppetmaster.example.com

[master]

dns\_alt\_names = puppet,puppetmaster,puppetserver

REPORT THIS AD

**Settings on puppetserver 1 and 2**

When a Puppet agent connects to a Puppet master, the communication is authenticated with SSL certificates.On these backend servers we need to configure them to access certificate information passed in SSL certificates headers. **Setting allow-header-cert-info to ‘true’ puts Puppet Server in vulnerable state**. Ensure puppeservers1/1 are not reachable by an untrusted network.With allow-header-cert-info set to ‘true’, authorization code will use only the client HTTP header values—not an SSL-layer client certificate—to determine the client subject name, authentication status, and trusted facts.

On Puppetmaster1/2 edit /etc/puppetlabs/puppetserver/conf.d/auth.conf  and add following line at the beginning of file:allow-header-cert-info: true

authorization: {

version: 1

 allow-header-cert-info: true

 rules: [

{

…………………………..

**Create SSL Certificate on puppetmaster (Load Balancer)**

Initialize CA certificate

puppet cert list -a

Create certificate request  
puppet certificate generate --dns-alt-names puppet,puppetmaster,puppetserver puppetmaster.example.com --ca-location local

**Issue certificate**

puppet cert sign puppetmaster.example.com --allow-dns-alt-names  
puppet certificate find puppetmaster.example.com --ca-location local

On Puppetserver1/2 remove contents of /etc/puppetlabs/puppet/ssl folder

Now, from Puppetmaster copy content of /etc/puppetlabs/puppet/ssl/ to the same location of puppetserver1/2

cd /etc/puppetlabs/puppet/ssl  
scp -r \* root@172.23.124.228:/etc/puppetlabs/puppet/ssl/  
scp -r \* root@172.23.124.230:/etc/puppetlabs/puppet/ssl/

Once certificates are copied, on puppetserver1/2 execute puppet cert list -a

all copied certificates should be recognized by both puppet backend servers.

On puppetmaster (Load Balancer) install apache and mod\_ssl

yum install httpd mod\_ssl

Create /etc/httpd/conf.d/puppetlb.conf

Listen 8140

ServerName puppetmaster.example.com

SSLEngine on

SSLProtocol -ALL +TLSv1 +TLSv1.1 +TLSv1.2

SSLCipherSuite ALL:!ADH:RC4+RSA:+HIGH:+MEDIUM:-LOW:-SSLv2:-EXP

SSLCertificateFile /etc/puppetlabs/puppet/ssl/certs/puppetmaster.example.com.pem

SSLCertificateKeyFile /etc/puppetlabs/puppet/ssl/private\_keys/puppetmaster.example.com.pem

SSLCertificateChainFile /etc/puppetlabs/puppet/ssl/ca/ca\_crt.pem

SSLCACertificateFile /etc/puppetlabs/puppet/ssl/ca/ca\_crt.pem

SSLCARevocationFile /etc/puppetlabs/puppet/ssl/ca/ca\_crl.pem

SSLVerifyClient optional

SSLVerifyDepth 1

SSLOptions +StdEnvVars +ExportCertData

RequestHeader unset X-Forwarded-For

RequestHeader set X-SSL-Subject %{SSL\_CLIENT\_S\_DN}e

RequestHeader set X-Client-DN %{SSL\_CLIENT\_S\_DN}e

RequestHeader set X-Client-Verify %{SSL\_CLIENT\_VERIFY}e

ProxyPassMatch ^/(puppet-ca/v[123]/.\*)$ balancer://puppetca/$1

ProxyPass / balancer://puppetworker/

ProxyPassReverse / balancer://puppetworker

BalancerMember http://172.23.124.237:18140

BalancerMember http://172.23.124.228:18140

BalancerMember http://172.23.124.230:18140

REPORT THIS AD

This configuration creates an Apache VirtualHost that will listen for connections on port 8140 and redirect traffic to one of the three puppetserver instances.Communication between the puppetserver machine and the puppetser1/2 will be unencrypted.

To redirect all certificate related traffic to a specific machine, the following ProxyPassMatch directive can be used:  
ProxyPassMatch ^/([^/]+/certificate.\*)$ balancer://puppetca/$1

On puppetserver start httpd and puppet

systemctl start httpd && systemctl enable httpd && systemctl start puppetserver && systemctl enable puppetserver

On backend puppetserver1/2 start puppeserver service

systemctl start puppetserver && systemctl enable puppetserver

Now on puppet node run puppet agent -t, sign certificate and puppet agent should work fine.

ADVERTISEMENT

[Puppet – Join machine to the Windows AD Domain](https://geekdudes.wordpress.com/2018/11/15/puppet-join-machine-to-the-windows-ad-domain/)

Posted: November 15, 2018 in [**puppet**](https://geekdudes.wordpress.com/category/puppet/), [**Windows Server**](https://geekdudes.wordpress.com/category/windows-server/)

[**1**](https://geekdudes.wordpress.com/2018/11/15/puppet-join-machine-to-the-windows-ad-domain/#comments)

install powershell module

puppet module install puppetlabs-powershell

Under modules/module name/manifests folder create manifest file,password [is encrypted with Hiera](https://geekdudes.wordpress.com/2018/11/01/installing-domain-controller-using-puppet/),after machine is joined to domain, it will be rebooted.

class domain\_membership (

$domain = 'ad.contoso.com',

$username = 'administrator',

$password = lookup('password'),

$secure\_password = false,

$machine\_ou = 'OU=test,DC=ad,DC=contoso,DC=com',

){

$code = " \

\$secStr=ConvertTo-SecureString '${password}' -AsPlainText -Force; \

if (-not \$?) { \

write-error 'Error: Unable to convert password string to a secure string'; \

exit 10; \

} \

\$creds=New-Object System.Management.Automation.PSCredential( '${username}', \$secStr ); \

if (-not \$?) { \

write-error 'Error: Unable to create PSCredential object'; \

exit 20; \

} \

Add-Computer -DomainName ${domain} -OUPath $\_machine\_ou -Restart -Force -Cred \$creds; \

if (-not \$?) { \

write-error 'Error: Unable to join domain'; \

exit 30; \

} \

exit 0"

#

# Use the Josh Cooper PowerShell provider

#

exec { 'join\_domain':

command => $code,

provider => powershell,

logoutput => true,

unless => "if ((Get-WMIObject Win32\_ComputerSystem).Domain -ne '${domain}') { exit 1 }",

}

}

[Configuring replica Domain controller using Puppet](https://geekdudes.wordpress.com/2018/11/14/configuring-replica-domain-controller-using-puppet/)

Posted: November 14, 2018 in [**puppet**](https://geekdudes.wordpress.com/category/puppet/), [**Windows Server**](https://geekdudes.wordpress.com/category/windows-server/)

[**9**](https://geekdudes.wordpress.com/2018/11/14/configuring-replica-domain-controller-using-puppet/#comments)

In [previous post](https://geekdudes.wordpress.com/2018/11/01/installing-domain-controller-using-puppet/) we created very first domain controller, in this one we’ll add Domain Controller to existing forest

node 'windows.example.com' {

file {['c:/NTDS']:

ensure => directory

}

dsc\_windowsfeature {'dns':

dsc\_ensure => 'Present',

dsc\_name => 'DNS',

}

dsc\_windowsfeature { 'addsinstall':

dsc\_ensure => 'Present',

dsc\_name => 'AD-Domain-Services',

}

dsc\_windowsfeature {'addstools':

dsc\_ensure => 'Present',

dsc\_name => 'RSAT-ADDS',

}

dsc\_windowsfeature {'addnstools':

dsc\_ensure => 'Present',

dsc\_name => 'RSAT-DNS-Server',

}

dsc\_xwaitforaddomain {'DscForestWait':

dsc\_domainname => 'ad.contoso.com',

dsc\_domainusercredential=> {

'user' => 'Administrator@ad.contoso.com',

'password' => Sensitive(lookup('password'))

},

dsc\_retrycount => 55,

dsc\_retryintervalsec => 10,

subscribe => Dsc\_windowsfeature['addsinstall'],

}

dsc\_xaddomaincontroller {'ReplicaDC':

dsc\_domainname => 'ad.contoso.com',

dsc\_domainadministratorcredential => {

'user' => 'Administrator@ad.contoso.com',

'password' => Sensitive(lookup('password'))

},

dsc\_safemodeadministratorpassword => {

'user' => 'admin',

'password' => 'Passw0rd01'

},

dsc\_databasepath => 'C:\NTDS',

dsc\_logpath => 'C:\NTDS',

dsc\_sysvolpath => 'C:\SYSVOL',

subscribe => Dsc\_xwaitforaddomain['DscForestWait'],

}

#this applies to AWS/Azure machines only

exec { 'Check if DNS is set automatically':

command => 'Set-DnsClientServerAddress -InterfaceAlias "Ethernet" -ResetServerAddresses',

unless => 'if (!((netsh interface ipv4 show dns | select-string "DNS servers configured through DHCP:") -match "DNS servers configured through DHCP:")) {exit 1}',

provider => powershell,

logoutput => true,

}

reboot {'dsc\_reboot':

subscribe => Dsc\_xaddomaincontroller['ReplicaDC'],

message => 'DSC has requested a reboot',

}

}

[Configuring AD Sites and Subnets using Puppet](https://geekdudes.wordpress.com/2018/11/13/configuring-ad-sites-and-subnets-using-puppet/)

Posted: November 13, 2018 in [**puppet**](https://geekdudes.wordpress.com/category/puppet/), [**Windows Server**](https://geekdudes.wordpress.com/category/windows-server/)

[**0**](https://geekdudes.wordpress.com/2018/11/13/configuring-ad-sites-and-subnets-using-puppet/#respond)

I modified [this](http://activedirectoryfaq.com/2016/02/create-active-directory-sites-with-powershell/) script:

Add-Type -Assembly Microsoft.VisualBasic

$dc = ($env:logonserver).SubString(2)

$sitesPath = "CN=Sites,CN=Configuration," + (Get-ADDomain).DistinguishedName

$logPath = "C:\Script\logs\"

#$fileDate = get-filedate

#$fileNameBig = "import-sites\_" + $env:username + "\_" + $fileDate + ".log"

$fileNameBig = "CreateADSIte" + ".log"

$fileNameSmall = "ADSites" + ".JSON"

$logFileBig = $logPath + $fileNameBig

$logFileSmall = $logPath + $fileNameSmall

function init-log {

if((Test-Path $logPath) -eq $false) {

Write-Host ("Creating log folder " + $logpath + "...") -ForegroundColor "Yellow"

try {

New-Item -ItemType Directory -Path $logPath -ErrorAction Stop | Out-Null

}

catch {

Write-Warning ("Log path (" + $logPath + ") could not be created! Please change variable!")

Exit

}

Write-Host ("Log folder successfully created.") -ForegroundColor "Green"

}

if((Test-Path $logFileBig) -eq $false) {

$date = Get-Date

Write-Host ("Creating log file " + $logFileBig + "...") -ForegroundColor "Yellow"

try {

"Date: " + $date | Out-File -Append -Encoding UTF8 -FilePath $logFileBig -ErrorAction Stop

}

catch {

Write-Warning "Could not write to log file (" + $logFileBig + ") Please check!"

Exit

}

Write-Host ("Log file successfully created.") -ForegroundColor "Green"

#("UserName: " + $env:username) | Out-File -Append -Encoding UTF8 -FilePath $logFileBig

Out-File -Append -Encoding UTF8 -FilePath $logFileBig

("Log file: " + $logFileBig) | Out-File -Append -Encoding UTF8 -FilePath $logFileBig

}

}

function get-subnetAD {

param($subnet)

$ldapFilterSubnet = "(&(objectCategory=subnet)(objectClass=subnet)(name=" + $subnet + "))"

$subnetAD = Get-ADObject -LDAPFilter $ldapFilterSubnet -SearchBase $sitesPath -Server $dc -Properties siteObject, location

return $subnetAD

}

function remove-newlines {

param($string)

$string = $string = ($string.Replace("`n"," ")).Replace("`r","")

return $string

}

function get-siteAD {

param($siteName)

$ldapFilterSite = "(&(objectClass=site)(objectCategory=site)(name=" + $siteName + "))"

$siteAD = Get-ADObject -LDAPFilter $ldapFilterSite -SearchBase $sitesPath -Server $dc -Properties location

return $siteAD

}

function get-filedate {

$date = Get-Date

$year = (($date.Year).ToString()).SubString(2,2)

if($date.Month -lt 10) {

$month = "0" + $date.Month

}

else {

$month = ($date.Month).ToString()

}

if($date.Day -lt 10) {

$day = "0" + ($date.Day).ToString()

}

else {

$day = $date.Day

}

$fileDate = $year + $month + $day

return $fileDate

}

function create-site {

param($siteName,

$location)

$ldapFilterSite = "(&(objectClass=site)(objectCategory=site)(name=" + $siteName + "))"

$siteAD = Get-ADObject -LDAPFilter $ldapFilterSite -SearchBase $sitesPath -Server $dc

if($siteAD -eq $null) {

try {

$siteAD = New-ADReplicationSite -Name $siteName -Server $dc -PassThru -ErrorAction Stop

}

catch {

log-write ("ADReplicationSite " + $siteName + " could not be created. Reason: " + $\_.Exception.Message) -foregroundColor "warn"

return $null

}

try {

#$siteAD = Set-ADReplicationSite -Identity $siteAD -Add @{location=$location} -Server $dc -ErrorAction Stop -PassThru

$siteAD = Set-ADReplicationSite -Identity $siteAD -Server $dc -ErrorAction Stop -PassThru

}

catch {

log-write ("ADReplicationSite " + $siteAD.Name + " could not be set. Reason: " + $\_.Exception.Message) -foregroundColor "warn"

}

}

return $siteAD

}

function create-subnet {

param($subnet,

$siteAD)

#$location)

try {

$subnetAD = New-ADReplicationSubnet -Name $subnet -Site $siteAD -Location $location -Server $dc -ErrorAction Stop -PassThru

#$subnetAD = New-ADReplicationSubnet -Name $subnet -Site $siteAD -Server $dc -ErrorAction Stop -PassThru

}

catch {

log-write ("ADReplicationSubnet " + $subnet + " could not be created. Reason: " + $\_.Exception.Message) -foregroundColor "warn"

return $null

}

return $subnetAD

}

function log-write {

param([String]$output,

$foregroundColor,

$backgroundColor)

#$timeStamp = get-timestamp

$output = $output

if($foregroundColor -ieq "warn") {

Write-Warning $output

}

else {

if(($backgroundColor -eq $null) -and ($foregroundColor -eq $null)) {

Write-Host $output

}

if(($backgroundColor -eq $null) -and ($foregroundColor -ne $null)) {

Write-Host $output -ForegroundColor $foregroundColor

}

if(($backgroundColor -ne $null) -and ($foregroundColor -eq $null)) {

Write-Host $output -BackgroundColor $backgroundColor

}

if(($backgroundColor -ne $null) -and ($foregroundColor -ne $null)) {

Write-Host $output -ForegroundColor $foregroundColor -BackgroundColor $backgroundColor

}

}

try {

$output | Out-File -Append -Encoding UTF8 -FilePath $logFileBig -ErrorAction Stop

}

catch {

Write-Warning "Error writing to log file!!"

Write-Host $\_.Exception.Message

}

}

function get-subnetMask {

param($subnet)

if(($subnet -eq $null) -or ($subnet -eq [String]::Empty)) {

return $null

}

$subnetMask = 0

$array = $subnet.Split(".")

foreach($octet in $array) {

if([Microsoft.VisualBasic.Information]::IsNumeric($octet)) {

$charArray = ([Convert]::ToString($octet,2)).ToCharArray()

foreach($bit in $charArray) {

$subnetMask += $bit.ToString()

}

}

else {

return $null

}

}

return $subnetMask

}

init-log

$CONFIGDATA = Get-Content -Path "C:\AD\Sites.json" | ConvertFrom-JSON

foreach($network in $CONFIGDATA)

{

$network | Add-Member NoteProperty "ADSite" $null

$subnetMask = get-subnetMask $network.Subnet

$subnet = $network.NetworkAddress + "/" + $subnetMask

$adsite=$network.SiteName

if(($network.SiteName -ne $null) -and ($network.SiteName -ne [String]::Empty)) {

$network.SiteName = remove-newlines $network.SiteName

}

$siteAD = get-siteAD $adsite

if($siteAD -eq $null) {

log-write ("Site " + $adsite + " could not be found. Creating site...") -foregroundColor "Yellow"

$siteAD = create-site $adsite

if($siteAD -eq $null) {

log-write ("Site " + $adsite + " could not be created!") -foregroundColor "warn"

#$network.SiteStatus = "Error"

export $network

continue

}

else {

log-write ("Site " + $adsite + " was successfully created") -foregroundColor "Green"

#$network.SiteStatus = "OK"

}

}

$subnetAD = get-subnetAD $subnet

if($subnetAD -eq $null) {

log-write ("Subnet could not be found in AD. Attempting to create it...") -foregroundColor "Yellow"

$subnetAD = create-subnet $subnet $siteAD

if($subnetAD -ne $null) {

log-write ("Subnet successfully created.") -foregroundColor "Green"

#$network.SubnetStatus = "OK"

}

else {

log-write ("Subnet could not be created") -foregroundColor "warn"

#$network.SubnetStatus = "Error"

#export $network

continue

}

}

}

REPORT THIS AD

Script above will read content of ADSites.JSON file

[

{

"NetworkAddress": "10.140.40.0",

"Subnet": "255.255.252.0",

"SiteName": "CTA001"

},

{

"NetworkAddress": "10.83.0.0",

"Subnet": "255.255.0.0",

"SiteName": "CTA001"

},

{

"NetworkAddress": "10.196.112.0",

"Subnet": "255.255.252.0",

"SiteName": "CTA001"

},

{

"NetworkAddress": "10.196.136.0",

"Subnet": "255.255.252.0",

"SiteName": "CTA002"

}

]

Script will create subnets specified in JSON file and associate it to corresponding AD site.  
Puppet manifest file will create C:\AD if it doesn’t exist, will copy JSON and ps1 file from /opt/puppetfiles/  in Puppet server to C:\AD on windows node and will execute createsite.ps1 script (Script will be executed only if C:\AD\logs doesn’t exist or if log file has Subnet successfully created string.

#Create C:\Script\ directory

file {

['c:/Script/']:

ensure => directory,

}

# copy JSON file with AD Site names and subnets

file { 'c:\Script\ADSites.json':

ensure => present,

content => file('/opt/puppetfiles/ADSites.json'),

}

# copy Script

file { 'c:\Script\createsite.ps1':

ensure => present,

content => file('/opt/puppetfiles/createsite.ps1'),

}

#Execute powershell script on Windows node (run it only if log files doesn't contain "Subnet successfully created"

#string

exec { 'Create sites':

command => 'C:\Script\createsite.ps1',

unless => 'if (!(Test-Path C:\Script\logs\CreateADSite.log) -or !(Select-String -Path C:\Script\logs\CreateADSite.log -Pattern "Subnet successfully created")) {exit 1}',

provider => powershell,

logoutput => true,

}

REPORT THIS AD

[Auto Signing Certificates for Puppet agent](https://geekdudes.wordpress.com/2018/11/12/auto-signing-certificates-for-puppet-agent/)

Posted: November 12, 2018 in [**Linux**](https://geekdudes.wordpress.com/category/linux/), [**puppet**](https://geekdudes.wordpress.com/category/puppet/)

[**1**](https://geekdudes.wordpress.com/2018/11/12/auto-signing-certificates-for-puppet-agent/#comments)

Puppet agent and server comunicate via SSL connection.By default, Puppet agent create Certificate Signing Request (CSR)  after we run puppet agent -t on client, and we need to manually sign this request on Puppet server puppet cert sign

We can automate this process in 2 ways:

* whitelisting domain name
* by script which reads log file and approves CSR automatically

**Whitelisting Domain Name**

On Puppet server run puppet config print --section master autosign

Create file /etc/puppetlabs/puppet/autosign.conf and whitelist domain name

Content of /etc/puppetlabs/puppet/autosign.conf:

\*.mshome.net

Restart Puppet master service:systemctl restart puppetserver

If we now run puppet agent -t on Puppet node, CSR will be signed automatically.

**Automating CSR using script**

On Puppet master run puppet config set --section master autosign /etc/puppetlabs/puppet/autosign.sh

Content of /etc/puppetlabs/pupppet/autosign.sh

#!/bin/bash

#

# a test script for policy based autosigning in puppet

#

# this script logs the certname and the CN from the CSR

# via syslog to local3.info. on centos7 this lands in

# /var/log/messages.

#

# $1 gets passed by the puppet master and is the certname of the agent

# the CSR is passed on STDIN

set -eof pipefail

export PATH=/bin:/sbin

readonly CERTNAME=$1

readonly CSR=$(cat -)

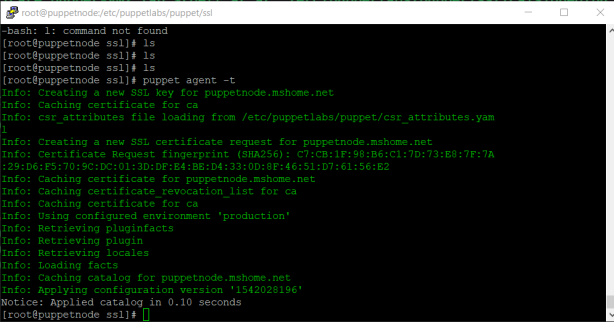
readonly CN=$(echo "${CSR}" | openssl req -noout -text | grep CN)

logger -p local3.info "received csr for host ${CERTNAME}"

logger -p local3.info "Common Name in CSR: ${CN}"

exit 0

Restart Puppet master service again and run puppet agent -t again



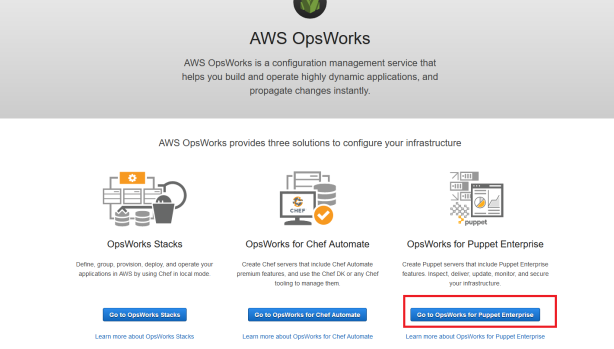
[AWS OpsWorks for Puppet Enterprise installation](https://geekdudes.wordpress.com/2018/11/02/opsworks-for-puppet-enterprise-installation/)

Posted: November 2, 2018 in [**Amazon Web Services (AWS)**](https://geekdudes.wordpress.com/category/amazon-web-services-aws/), [**puppet**](https://geekdudes.wordpress.com/category/puppet/)

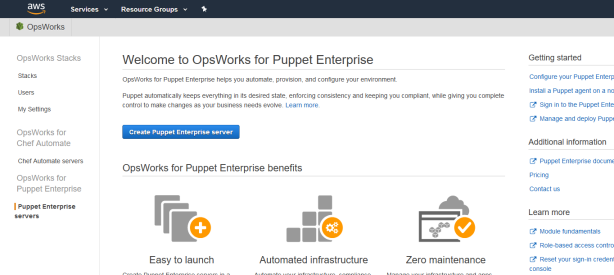
[**0**](https://geekdudes.wordpress.com/2018/11/02/opsworks-for-puppet-enterprise-installation/#respond)

OpsWorks for Puppet Enterprise runs Puppet enterprise server in AWS

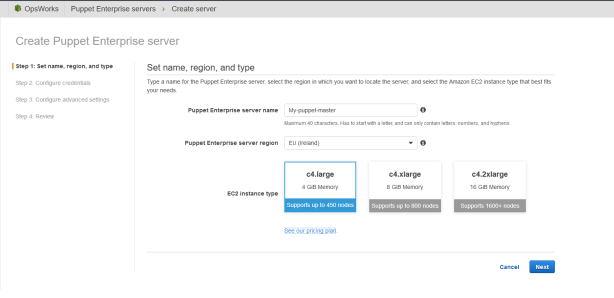
In AWS console select OpsWorks-Go to OpsWorks for Puppet Enterprise



Create Puppet Enterprise Server



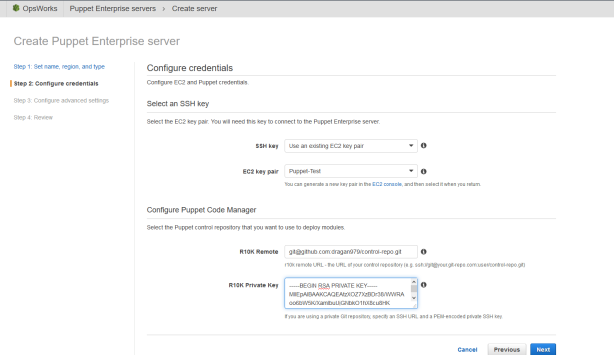
Specify server name,region and instance size



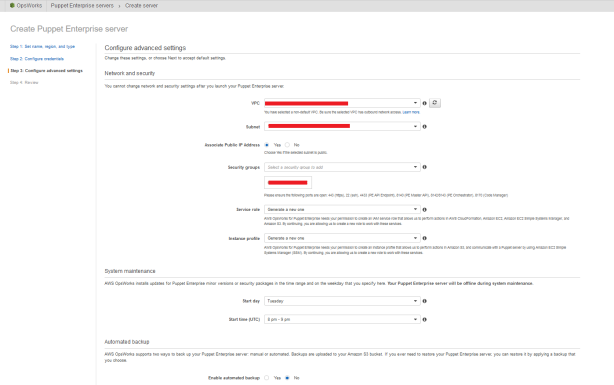
REPORT THIS AD

Specify EC2 Key pair,Puppet git control repository and private key.

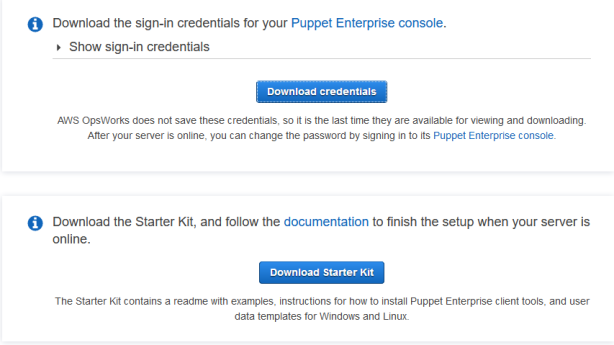
For direction how to create GitHub SSH connection see [this](https://geekdudes.wordpress.com/2018/04/30/committing-files-to-git-windows/) post.



Specify VPC,Subnet,Security Group,System maintenance time and choose if you want to enable automatic backup of AWS Puppet instance



During installation download credentials and starter kit.Credentials will be used to authenticate to puppet when accessing via web console

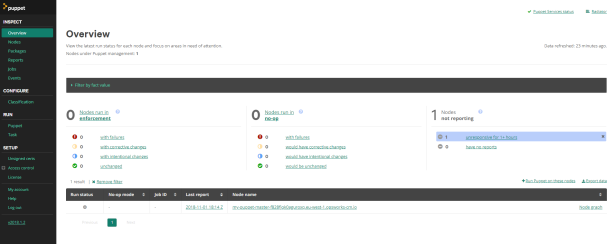


REPORT THIS AD

**Web access**:

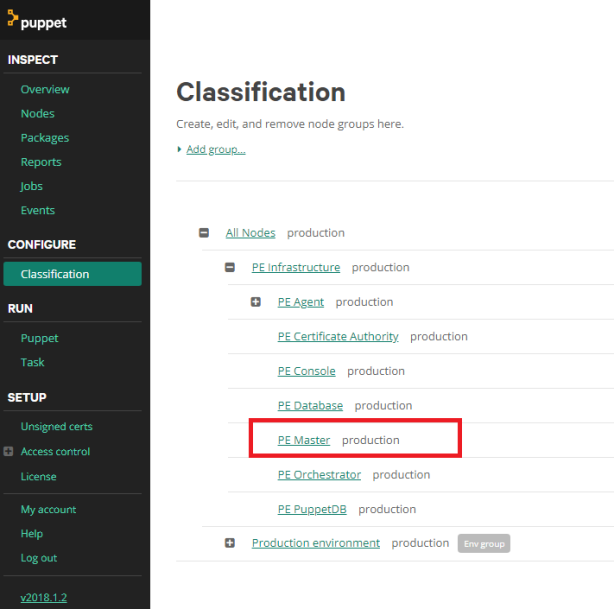
[https://puppet instance ip](https://18.202.96.126/#/inspect/overview)

Use credentials downloaded in previous step



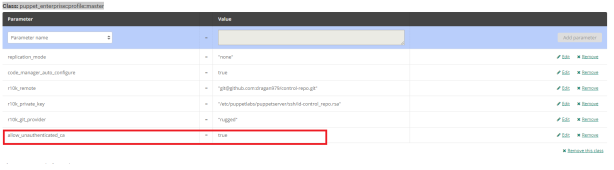
**Installing Windows agent**

First allow unauthenticated CA (to solve “access denied” issue when sending certification requests)  
In Puppet console click on Classification-Expand PE Infrastructure-PE Master



REPORT THIS AD

Click on configuration,under Class: puppet\_enterprise::profile::master add allow\_unauthenticated\_ca and set it to true



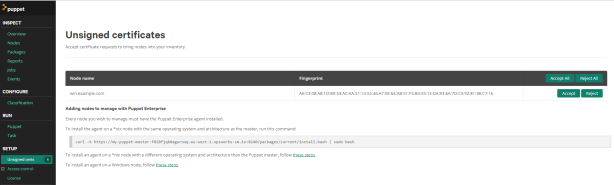
Windows agent is located at Puppet Enterprise server: /opt/puppetlabs/server/data/[ackages/public/<puppet version>/windows-x86\_64-<puppet version>/puppet-agent-x64/msi

Transfer that file to Windows node,

Open CMD as administrator and run

[puppet-agent-x64.msi](https://docs.corp.hentsu.com/display/ENG/puppet-agent-x64.msi) /qn PUPPET\_MASTER\_SERVER=[my-puppet.opsworks-cm.io](https://docs.corp.hentsu.com/display/ENG/my-puppet-f828fjqk0aguroxq.eu-west-1.opsworks-cm.io)PUPPET\_AGENT\_CERTNAME=[wind.example.com](https://docs.corp.hentsu.com/display/ENG/windows.example.com)

Go to Puppet Enterprise console-Unsigned certs and sign certificate



[Installing Domain Controller using Puppet](https://geekdudes.wordpress.com/2018/11/01/installing-domain-controller-using-puppet/)

Posted: November 1, 2018 in [**puppet**](https://geekdudes.wordpress.com/category/puppet/), [**Windows Server**](https://geekdudes.wordpress.com/category/windows-server/)

[**2**](https://geekdudes.wordpress.com/2018/11/01/installing-domain-controller-using-puppet/#comments)

Install DSC  and hiera-eyaml modules (for password encryption):

puppet module install puppetlabs-dsc

puppetserver gem install hiera-eyaml

Edit /etc/puppetlabs/puppet/hiera.yaml

---

version: 5

defaults:

datadir: data

data\_hash: yaml\_data

hierarchy:

- name: "Eyaml hierarchy"

lookup\_key: eyaml\_lookup\_key # eyaml backend

paths:

- "nodes/%{trusted.certname}.yaml"

- "windowspass.eyaml"

options:

pkcs7\_private\_key: "/etc/puppetlabs/puppet/keys/private\_key.pkcs7.pem"

pkcs7\_public\_key: "/etc/puppetlabs/puppet/keys/public\_key.pkcs7.pem"

Create keys (make sure key path reflects path from hiera.yaml file):

/opt/puppetlabs/puppet/bin/eyaml createkeys

Create password (-l is just label):

/opt/puppetlabs/puppet/bin/eyaml encrypt -l 'password' -s 'Pass' --pkcs7-public-key=/etc/puppetlabs/puppet/keys/[public\_key.pkcs7.pem](https://docs.corp.hentsu.com/display/ENG/public_key.pkcs7.pem) --pkcs7-private-key=/etc/puppetlabs/puppet/keys/[private\_key.pkcs7.pem](https://docs.corp.hentsu.com/display/ENG/private_key.pkcs7.pem)

Add this encrypted password to /etc/puppetlabs/puppet/data/windowspass.eyaml file:



/opt/puppetlabs/puppet/bin/eyaml edit [windowspass.eyaml](https://docs.corp.hentsu.com/display/ENG/windowspass.eyaml) --pkcs7-public-key=/etc/puppetlabs/puppet/keys/[public\_key.pkcs7.pem](https://docs.corp.hentsu.com/display/ENG/public_key.pkcs7.pem) --pkcs7-private-key=/etc/puppetlabs/puppet/keys/[private\_key.pkcs7.pem](https://docs.corp.hentsu.com/display/ENG/private_key.pkcs7.pem)

cat /etc/puppetlabs/puppet/data/[windowspass.eyaml](https://docs.corp.hentsu.com/display/ENG/windowspass.eyaml)

image2018-10-31_12-20-1.png

**Test decryption:**

/opt/puppetlabs/puppet/bin/eyaml decrypt -f [windowspass.eyaml](https://docs.corp.hentsu.com/display/ENG/windowspass.eyaml) --pkcs7-public-key=/etc/puppetlabs/puppet/keys/[public\_key.pkcs7.pem](https://docs.corp.hentsu.com/display/ENG/public_key.pkcs7.pem) --pkcs7-private-key=/etc/puppetlabs/puppet/keys/[private\_key.pkcs7.pem](https://docs.corp.hentsu.com/display/ENG/private_key.pkcs7.pem)

Secure keys:

chown -R puppet:puppet /etc/puppetlabs/puppet/keys

chmod 400 /etc/puppetlabs/puppet/keys/private\_key[.pkcs7.pem](https://docs.corp.hentsu.com/display/ENG/.pkcs7.pem)

chmod 400 /etc/puppetlabs/puppet/keys/public\_key[.pkcs7.pem](https://docs.corp.hentsu.com/display/ENG/.pkcs7.pem)

For Windows currently is not possible to hide passwords when running  agent in verbose output:

puppet agent -t -v

Map content of windowspass.eyaml to manifest file:

'password' => Sensitive(lookup('password'))

Complete code-**/etc/puppetlabs/code/environments/production/manifests/site.pp:**

node 'windows.example.com' {

file {

['c:/NTDS']:

ensure => directory

}

dsc\_windowsfeature {'dns':

dsc\_ensure => 'Present',

dsc\_name => 'DNS',

}

dsc\_windowsfeature { 'addsinstall':

dsc\_ensure => 'Present',

dsc\_name => 'AD-Domain-Services',

}

dsc\_windowsfeature {'addstools':

dsc\_ensure => 'Present',

dsc\_name => 'RSAT-ADDS',

}

dsc\_windowsfeature {'addnstools':

dsc\_ensure => 'Present',

dsc\_name => 'RSAT-DNS-Server',

}

dsc\_xaddomain { 'firstdc':

subscribe => Dsc\_windowsfeature['addsinstall'],

dsc\_domainname => 'ad.contoso.com',

dsc\_domainadministratorcredential => {

'user' => 'pagent',

'password' => Sensitive(lookup('password'))

},

dsc\_safemodeadministratorpassword => {

'user' => 'pagent',

'password' => 'password' => Sensitive(lookup('password'))

},

dsc\_databasepath => 'c:\NTDS',

dsc\_logpath => 'c:\NTDS',

}

reboot {'dsc\_reboot':

message => 'DSC has requested a reboot',

when => pending,

}

}

For debugging:

puppet master --debug --compile [windows.example.com](https://docs.corp.hentsu.com/display/ENG/windows.example.com) --environment=production

**Creating new AD user,create New Security group and add user to it:**

dsc\_xADUser {'FirstUser':

dsc\_ensure => 'present',

dsc\_domainname => '[ad.contoso.com',](https://docs.corp.hentsu.com/display/ENG/ad.contoso.com',)

dsc\_username   => 'tfl',

dsc\_userprincipalname => 'tfl@[ad.contoso.com',](https://docs.corp.hentsu.com/display/ENG/ad.contoso.com',)

dsc\_password   => {

'user' => 'tfl@[ad.contoso.com',](https://docs.corp.hentsu.com/display/ENG/ad.contoso.com',)

'password' => Sensitive(lookup('password'))

},

dsc\_passwordneverexpires => true,

dsc\_domainadministratorcredential => {

'user'  => 'Administrator@[ad.contoso.com',](https://docs.corp.hentsu.com/display/ENG/ad.contoso.com',)

'password' => Sensitive(lookup('password'))

},

}

dsc\_xgroup {'testgroup':

dsc\_ensure    => 'present',

dsc\_memberstoinclude => 'tfl@[ad.contoso.com',](https://docs.corp.hentsu.com/display/ENG/ad.contoso.com',)

dsc\_groupname   => 'test',

#dsc\_credential => {

#'user' => 'Administrator@[ad.contoso.com',](https://docs.corp.hentsu.com/display/ENG/ad.contoso.com',)

#'password' => 'Passw0rd01'

#},

}

[Install Puppet agent on Windows](https://geekdudes.wordpress.com/2018/11/01/install-puppet-agent-on-windows/)

Posted: November 1, 2018 in [**puppet**](https://geekdudes.wordpress.com/category/puppet/), [**Windows Server**](https://geekdudes.wordpress.com/category/windows-server/)

[**0**](https://geekdudes.wordpress.com/2018/11/01/install-puppet-agent-on-windows/#respond)

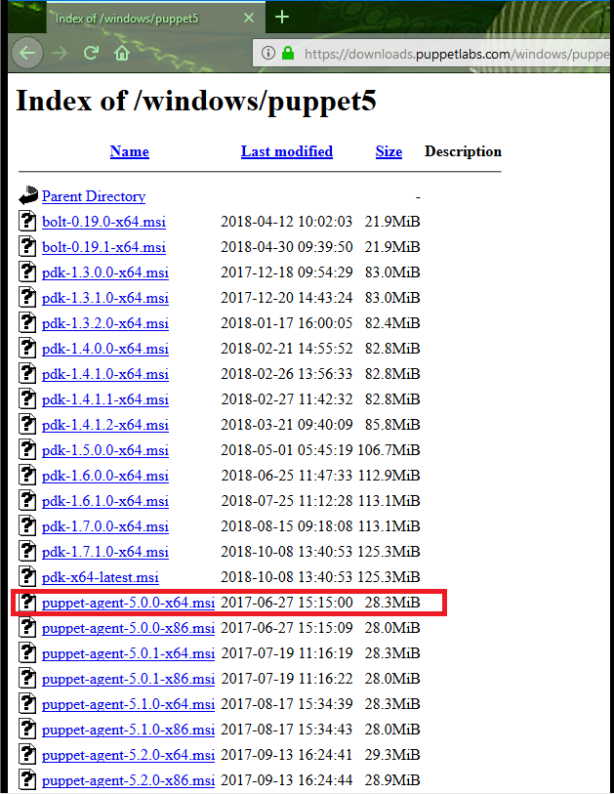
In [last post](https://geekdudes.wordpress.com/2018/11/01/puppet-master-installation-on-centos-7/) we installed Puppet server, in this one we’ll install Puppet agent on Windows server

Make sure Puppet server DNS name is resolvable from Windows host and vice-versa (skip this step if there is Host A DNS record for Puppet server):

C:\Windows\system32\drivers\etc\hosts

[192.168.1.97](https://docs.corp.hentsu.com/display/ENG/192.168.1.97) [puppetserver.example.com](https://docs.corp.hentsu.com/display/ENG/puppetserver.example.com)

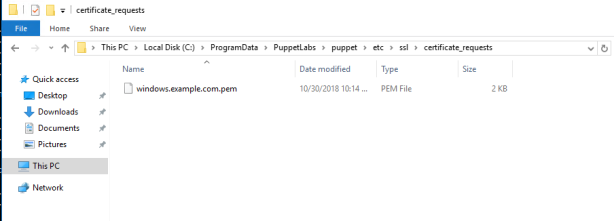
[Download](https://downloads.puppetlabs.com/windows/puppet5/) latest Puppet agent



Installation:

[puppet-agent-5.0.0-x64.msi](https://docs.corp.hentsu.com/display/ENG/puppet-agent-5.0.0-x64.msi) /qn PUPPET\_MASTER\_SERVER=[puppetserver.example.com](https://docs.corp.hentsu.com/display/ENG/puppetserver.example.com)PUPPET\_AGENT\_CERTNAME=[windows.example.com](https://docs.corp.hentsu.com/display/ENG/windows.example.com)

If all goes smooth, certificate request file should be seen in C:\ProgramData\PuppetLabs\puppet\etc\ssl\certificate\_requests

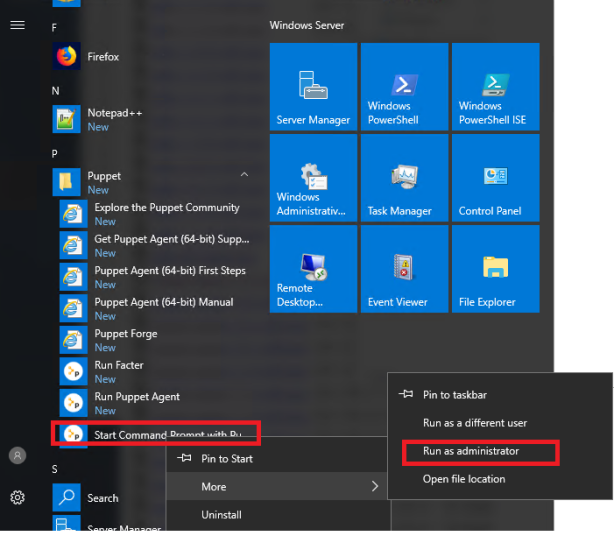


This certificate is waiting for signing, to do so, we need to move to Puppet server:

sudo puppet cert list

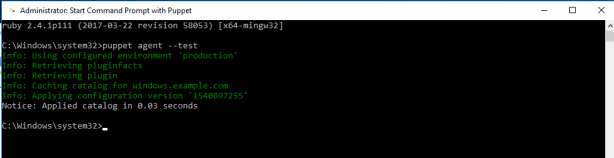
sudo puppet cert sign "[windows.example.com](https://docs.corp.hentsu.com/display/ENG/windows.example.com)"

On Windows server go to Puppet-Start Command Prompt with Puppet as Administrator



Test connection from Windows to Puppet server:

puppet agent --test

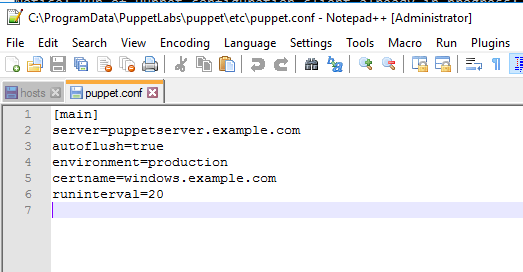


By default, Puppet agent polls Puppet server every 30 minutes.This behavior can be overridden by specifying custom polling interval by editing “C:\ProgramData\PuppetLabs\puppet\etc\puppet.conf” fileand adding line

runinterval=

One security concern is that catalog file stores credentials in plain text, to avoid this,prevent storing this file (which is in JSON format) to Puppet client

catalog\_cache\_terminus=""



**Creating  simple manifest file**

In this example we’ll make sure that IIS is installed

Install IIS module from Forge:

puppet module install puppetlabs-iis --version [4.5.0](https://docs.corp.hentsu.com/display/ENG/4.5.0)

Create manifest file and specify node and actions:

**/etc/puppetlabs/code/environments/production/manifests/site.pp**:

node '[windows.example.com'](https://docs.corp.hentsu.com/display/ENG/windows.example.com') {

$iis\_features = ['Web-WebServer','Web-Scripting-Tools','Web-Mgmt-Console']

iis\_feature { $iis\_features:

ensure => 'present',

include\_management\_tools => 'present'

}

}

Check for syntax errors:

puppet parser validate site.pp

**Run powershell scripts**

In order not to run script every time, when script runs if will create file with content “Done”

When Puppet tries to run this script again, it will check whether C:\log.txt or string “Done” exists in C:\log.txt, if both checks are true then script won’t be run (unless directive)

exec { 'configure\_gpo':

command => 'Set-ADDefaultDomainPasswordPolicy -Identity ad.contoso.com -ComplexityEnabled $true -MinPasswordLength 7 -MinPasswordAge 1 -MaxPasswordAge 30 -LockoutDuration 00:30:00 -LockoutObservationWindow 00:30:00 -LockoutThreshold 3;write-output "Done" | out-file C:\log.txt -Append',

unless => 'if (!(Test-Path C:\log.txt) -or !(Select-String -Path C:\log.txt -Pattern "Done")) {exit 1}',

provider => powershell,

logoutput => true,

}

[Puppet Master Installation on CentOS 7](https://geekdudes.wordpress.com/2018/11/01/puppet-master-installation-on-centos-7/)

Posted: November 1, 2018 in [**Linux**](https://geekdudes.wordpress.com/category/linux/), [**puppet**](https://geekdudes.wordpress.com/category/puppet/)

[**1**](https://geekdudes.wordpress.com/2018/11/01/puppet-master-installation-on-centos-7/#comments)

Make sure DNS resolution is in place.This is needed in case no central DNS server is available:

cat /etc/hosts

[192.168](https://docs.corp.hentsu.com/display/ENG/192.168).[1.97](https://docs.corp.hentsu.com/display/ENG/1.97) puppetmaster

[192.168](https://docs.corp.hentsu.com/display/ENG/192.168).[1.97](https://docs.corp.hentsu.com/display/ENG/1.97) [puppetserver.example.com](https://docs.corp.hentsu.com/display/ENG/puppetserver.example.com)

Add puppet repository:

rpm -Uvh https://yum[.puppetlabs.com](https://docs.corp.hentsu.com/display/ENG/.puppetlabs.com)/puppet5/puppet5-release-el-7[.noarch.rpm](https://docs.corp.hentsu.com/display/ENG/.noarch.rpm)

Install Puppet:

sudo yum -y install puppetserver

#check version:

/opt/puppetlabs/bin/puppet --version

#Add Puppet path to Environmental variable:

export PATH=/opt/puppetlabs/bin:$PATH

source ~/.bash\_profile

Alocate memory:

In this example 1GB of RAM is allocated to puppet:  
Edit /etc/sysconfig/puppetserver  
#Now change the line as below (Replace 1 with number of GBs)  
JAVA\_ARGS="-Xms1g -Xmx1g ...."  
#Save and exit.

Edit **/etc/puppetlabs/puppet/puppet.conf (make sure to put here puppet server name resolvable from DNS and from nodes)**

[master]

dns\_alt\_names=[puppetserver.example.com,puppetmaster,puppet](https://docs.corp.hentsu.com/display/ENG/puppetserver.example.com,puppetmaster,puppet)

[main]

certname = [puppetserver.example.com](https://docs.corp.hentsu.com/display/ENG/puppetserver.example.com)

server = [puppetserver.example.com](https://docs.corp.hentsu.com/display/ENG/puppetserver.example.com)

environment = production

runinterval = 5s

Enable/start puppet service

systemctl start puppetserver

systemctl enable puppetserver

Open port 8140 if firewalld is running

firewall-cmd --add-port=8140/tcp --permanent

firewall-cmd --reload

Add path to puppet command to VISUDO

sudo visudo

Defaults    secure\_path = /sbin:/bin:/usr/sbin:/usr/bin:/opt/puppetlabs/bin

**Disable agent (one time run)**

If we want “one time” run (execute some manifest and stop contacting Puppet master, we can configure agent to stop-at the end of manifest file)

service { 'puppet':

  ensure => 'stopped',

  enable => 'false',

}