Puppet Load Balancing

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

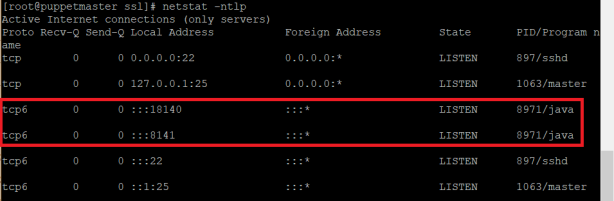
ssl-port: 8141

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

**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

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.