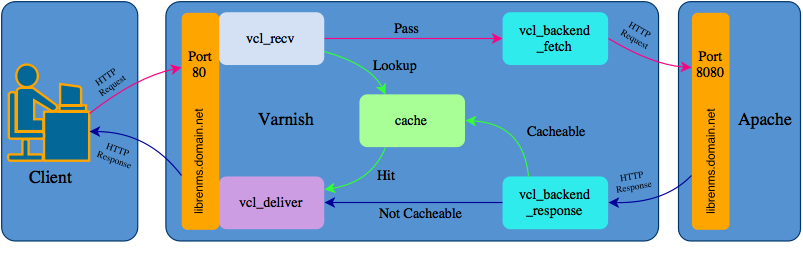
# **Varnish**

Varnish is a software package that works like a reverse proxy caching server/http accelerator. It gets placed in front of the web server which handles the incoming request from client first and serves the cached content without sending the request to main server. It results into reducing the load time on server and increases the number of requests web server can handle simultaneously.



Above diagram shows the [proxy server](http://en.wikipedia.org/wiki/Reverse_proxy) placement in web architecture, which explains that proxy server/Varnish server is a separate service so it doesn’t matter whether actual application is hosted on Apache/NGINX/IIS based server.

Few Highlights:

* Varnish caching is also for anonymous user only, so it will be effective for not logged in user.
* Drupal native page caching i.e. cache pages for anonymous user should be enabled to work Varnish.
* Varnish is in-memory/file based caching, means it can be configured to cache content either in memory or file system. So, it you have enough memory space then enable the memory based caching.

# **Modify Apache to run on different Port**

Varnish works on port 80, so we will have to configure the Apache on different port so that Varnish can handle the incoming request on server.

1. open /etc/apache2/ports.conf file

2. Inside this file change the ‘NameVirtualHost’ and ‘Listen’ port from 80 to 8181 (in this example we are configuring apache to run on port 8181)

NameVirtualHost \*:8181   
 Listen 8181

3. Save this file

4. In case there are many host defined on the server, we will have to change the port in host file also. E.g. if you have host defined as ‘drupal7.local’ then go to /etc/apache2/sites-enabled and open ‘drupal7.local’ or ‘default’ host config file.

5. Update the port under <VirtualHost \*:8181> tag.

6. Restart the apache with this command: sudo service apache2 restart

7. You have successfully configured the apache on different port i.e. 8181

# **Install Varnish**

Varnish can be installed using apt command. Below are the steps to install the Varnish on ubuntu.

1. apt-get install apt-transport-https

2. curl https://repo.varnish-cache.org/debian/GPG-key.txt | apt-key add -

3. Add the package URL to apt-get repository sources list:

echo "deb http://repo.varnish-cache.org/ubuntu/ precise varnish-3.0" | sudo tee -a /etc/apt/sources.list

apt-get update

apt-get install varnish

Varnish has been installed. Now. now next task is to configure varnish to accept incoming request.

# **Enabling Varnish to accept incoming request**

Varnish has two configuration files which needs to be modified:

1. /etc/default/varnish - Configuration file where port, memory allocation, secret key path and default vcl file will be defined.
2. /etc/varnish/default.vcl - Varnish uses this file containing instructions written in VCL Language in order to run its program. This is used to define how Varnish should handle the requests and how the document *caching* system should work

## **Modify /etc/default/varnish** **file** to change port and allocate memory

1. Open this file /etc/default/varnish
2. You will see a block of text defining the Varnish *daemon* options starting with the text DAEMON\_OPTS, similar to:

DAEMON\_OPTS="-a :6081   
 -T localhost:6082   
 -f /etc/varnish/default.vcl  
 -S /etc/varnish/secret  
 -s malloc,256m"

Let's modify it to change the port from 6081 to 80:

DAEMON\_OPTS="-a :80   
 -T localhost:6082   
 -f /etc/varnish/default.vcl   
 -S /etc/varnish/secret   
 -s malloc,256m"

1. Here “-s malloc, 256m” defines the storage system as in-memory with 256 MB allocated to varnish.

**Modify /etc/varnish/default.vcl file**

There are many settings in this file which needs to be configured intelligently but below are the few main settings:

1. Backend default section:

backend default {  
 .host = "127.0.0.1"; // or it can be host defined in host config file e.g. drupal7.local  
 .port = "8181";  
 .max\_connections = 250;  
 .connect\_timeout = 600s;  
 .first\_byte\_timeout = 600s;  
 .between\_bytes\_timeout = 600s;  
}

Note: Underneath, you will see a large block of commented out default VCL configuration code. You are free to keep it as long as you do not uncomment them by mistake, in which case compiling it would generate errors.

1. In order to define the allowed addresses which can access the cron.php or install.php, append the following:

acl internal {  
 "127.0.0.1”;  
 # For remote access, add your IP address here.  
 # Ex: 162.xxx.xx.xx  
}

You can simply update this file with below given code snippet which shall work for the most of the website. Please update backend default based on your set up.

# This is a basic VCL configuration file for varnish. See the vcl(7) main page for details on VCL syntax and semantics.

# Default backend definition. Set this to point to your content server.

backend default {

.host = "127.0.0.1";

.port = "8181";

.connect\_timeout = 600s;

.first\_byte\_timeout = 600s;

.between\_bytes\_timeout = 600s;

}

acl internal {

"127.0.0.1";

# For remote access, add your IP address here.

# Ex: 162.xxx.xx.xx

}

# Respond to incoming requests.

sub vcl\_recv {

if (req.request == "GET" && req.url ~ "^/varnishcheck$") {

error 200 "Varnish is Ready";

}

# Allow the backend to serve up stale content if it is responding slowly.

if (!req.backend.healthy) {

# Use anonymous, cached pages if all backends are down.

unset req.http.Cookie;

if (req.http.X-Forwarded-Proto == "https") {

set req.http.X-Forwarded-Proto = "http";

}

set req.grace = 30m;

} else {

set req.grace = 15s;

}

# Get ride of progress.js query params

if (req.url ~ "^/misc/progress\.js\?[0-9]+$") {

set req.url = "/misc/progress.js";

}

# Do not cache these paths.

if (req.url ~ "^/status\.php$" ||

req.url ~ "^/update\.php$" ||

req.url ~ "^/ooyala/ping$" ||

req.url ~ "^/admin" ||

req.url ~ "^/admin/.\*$" ||

req.url ~ "^/user" ||

req.url ~ "^/user/.\*$" ||

req.url ~ "^/users/.\*$" ||

req.url ~ "^/info/.\*$" ||

req.url ~ "^/flag/.\*$" ||

req.url ~ "^.\*/ajax/.\*$" ||

req.url ~ "^.\*/ahah/.\*$") {

return (pass);

}

# Pipe these paths directly to Apache for streaming.

if (req.url ~ "^/admin/content/backup\_migrate/export") {

return (pipe);

}

# Do not allow outside access to cron.php or install.php.

if (req.url ~ "^/(cron|install)\.php$" && !client.ip ~ internal) {

# Have Varnish throw the error directly.

error 404 "Page not found.";

# Use a custom error page that you've defined in Drupal at the path "404".

# set req.url = "/404";

}

# Handle compression correctly. Different browsers send different

# "Accept-Encoding" headers, even though they mostly all support the same

# compression mechanisms. By consolidating these compression headers into

# a consistent format, we can reduce the size of the cache and get more hits.= # @see: http:// varnish.projects.linpro.no/wiki/FAQ/Compression

if (req.http.Accept-Encoding) {

if (req.http.Accept-Encoding ~ "gzip") {

# If the browser supports it, we'll use gzip.

set req.http.Accept-Encoding = "gzip";

}

else if (req.http.Accept-Encoding ~ "deflate") {

# Next, try deflate if it is supported.

set req.http.Accept-Encoding = "deflate";

}

else {

# Unknown algorithm. Remove it and send unencoded.

unset req.http.Accept-Encoding;

}

}

# Always cache the following file types for all users.

if (req.url ~ "(?i)\.(png|gif|jpeg|jpg|ico|swf|css|js)(\?[a-z0-9]+)?$") {

unset req.http.Cookie;

}

set req.http.Cookie = regsuball(req.http.Cookie, "(^|;\s\*)(\_\_[a-z]+|has\_js)=[^;]\*", "");

# Remove all cookies that Drupal doesn't need to know about. ANY remaining

# cookie will cause the request to pass-through to a backend. For the most part

# we always set the NO\_CACHE cookie after any POST request, disabling the

# Varnish cache temporarily. The session cookie allows all authenticated users

# to pass through as long as they're logged in.

#

# 1. Append a semi-colon to the front of the cookie string.

# 2. Remove all spaces that appear after semi-colons.

# 3. Match the cookies we want to keep, adding the space we removed

# previously, back. (\1) is first matching group in the regsuball.

# 4. Remove all other cookies, identifying them by the fact that they have

# no space after the preceding semi-colon.

# 5. Remove all spaces and semi-colons from the beginning and end of the

# cookie string.

if (req.http.Cookie) {

set req.http.Cookie = ";" + req.http.Cookie;

set req.http.Cookie = regsuball(req.http.Cookie, "; +", ";");

set req.http.Cookie = regsuball(req.http.Cookie, ";(S{1,2}ESS[a-z0-9]+|NO\_CACHE)=", "; \1=");

set req.http.Cookie = regsuball(req.http.Cookie, ";[^ ][^;]\*", "");

set req.http.Cookie = regsuball(req.http.Cookie, "^[; ]+|[; ]+$", "");

if (req.http.Cookie == "") {

# If there are no remaining cookies, remove the cookie header. If there

# aren't any cookie headers, Varnish's default behavior will be to cache

# the page.

unset req.http.Cookie;

}

else {

# If there is any cookies left (a session or NO\_CACHE cookie), do not

# cache the page. Pass it on to Apache directly.

return (pass);

}

}

## From default below ##

if (req.restarts == 0) {

if (req.http.x-forwarded-for) {

set req.http.X-Forwarded-For =

req.http.X-Forwarded-For + ", " + client.ip;

} else {

set req.http.X-Forwarded-For = client.ip;

}

}

if (req.request != "GET" &&

req.request != "HEAD" &&

req.request != "PUT" &&

req.request != "POST" &&

req.request != "TRACE" &&

req.request != "OPTIONS" &&

req.request != "DELETE") {

/\* Non-RFC2616 or CONNECT which is weird. \*/

return (pipe);

}

if (req.request != "GET" && req.request != "HEAD") {

/\* We only deal with GET and HEAD by default \*/

return (pass);

}

## Unset Authorization header if it has the correct details...

#if (req.http.Authorization == "Basic ") {

# unset req.http.Authorization;

#}

if (req.http.Authorization || req.http.Cookie) {

/\* Not cacheable by default \*/

return (pass);

}

return (lookup);

}

# Code determining what to do when serving items from the Apache servers.

sub vcl\_fetch {

if (beresp.status == 301) {

set beresp.ttl = 1h;

return(deliver);

}

## Doesn't seem to work as expected

#if (beresp.status == 500) {

# set beresp.saintmode = 10s;

# return(restart);

#}

# Allow items to be stale if needed.

set beresp.grace = 1h;

}

# Set a header to track a cache HIT/MISS.

sub vcl\_deliver {

if (obj.hits > 0) {

set resp.http.X-Varnish-Cache = "HIT";

}

else {

set resp.http.X-Varnish-Cache = "MISS";

}

}

# In the event of an error, show friendlier messages.

sub vcl\_error {

set obj.http.Content-Type = "text/html; charset=utf-8";

set obj.http.Retry-After = "5";

synthetic {"

<?xml version="1.0" encoding="utf-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html>

<head>

<title>"} + obj.status + " " + obj.response + {"</title>

</head>

<body>

<h1>Error "} + obj.status + " " + obj.response + {"</h1>

<p>"} + obj.response + {"</p>

<h3>Guru Meditation:</h3>

<p>XID: "} + req.xid + {"</p>

<hr>

<p>Varnish cache server</p>

</body>

</html>

"};

return (deliver);

}

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

3. Restart the varnish now

/etc/init.d/varnish restart

# Verify Apache and Varnish configured correctly

netstat -anp --tcp --udp | grep LISTEN

# **Configure Varnish to work with Drupal**

1. Install and enable https://www.drupal.org/project/varnish project.

2. Go to ‘Configuration’ > Development > Varnish

(http://<sitename>/admin/config/development/varnish)

3. Update below settings and save:

Flush pages cache on cron : Enabled (please make sure cron is set externally i.e. should not be invoked on every page request)

Varnish Version - 3.x

Varnish Control Terminal - 127.0.0.1:6082

Varnish control key - <copy the key from /etc/varnish/secret file and paste it here>

Varnish connection timeout - 100

Varnish caching clearing - Default (if expire module not used)

Varnish ban type – normal

4. Status should be shown as - ‘Varnish running’ now.

5. Go to http://<site\_admin>admin/config/development/performance page

6. Update below settings on the performance page:

Enable ‘cache pages for anonymous user’ //mandatory to work the varnish cache

Enable ‘cache blocks’

Minimum cache lifetime - 3 min //set as per requirement

Expiration of cache pages - 12 hours // set as per requirement

7. Update below settings in sites/default/settings.php file

8. Clear the cache and Varnish should be working now.

# **Verify Varnish is working or not**

One of the simplest way to ensure Varnish is working as expected is to query it using [isvarnishworking.com](http://www.isvarnishworking.com/), a quick and easy method of observing Varnish headers. This is a lazy way of either curling the site or inspecting headers using the browser which would produce the following results

[$ curl -I http://d](http://www.adammalone.net/)rupal7.local

|  |  |
| --- | --- |
| Accept-Ranges | bytes |
| Age | 31 |
| Cache-Control | public, max-age=1800 |
| Connection | keep-alive |
| Content-Encoding | gzip |
| Content-Language | en |
| Content-Type | text/html; charset=utf-8 |
| Date | Fri, 06 Jan 2017 06:37:05 GMT |
| Etag | "1483684593-1" |
| Expires | Sun, 19 Nov 1978 05:00:00 GMT |
| Last-Modified | Fri, 06 Jan 2017 06:36:33 GMT |
| Server | Apache/2.4.7 (Ubuntu) |
| Vary | Cookie,Accept-Encoding |
| Via | 1.1 varnish |
| X-Content-Type-Options | nosniff |
| X-Drupal-Cache | MISS |
| X-Frame-Options | SAMEORIGIN |
| X-Varnish | 1946530985 1946530981 |
| X-Varnish-Cache | HIT |
| X-generator | Drupal 7 ([http://drupal.org](http://drupal.org/)) |
|  |  |

If you are able to find the above highlighted header response, then your page is being served from varnish. Congrats!!!.

Reference:

<https://www.varnish-software.com/static/book/Tuning.html>

<https://www.varnish-cache.org/docs/3.0/tutorial/purging.html>

<http://openconcept.ca/blog/mmallett/apc-varnish-memcache-and-caching-beyond-drupal-core>

<https://www.lullabot.com/blog/article/configuring-varnish-high-availability-multiple-web-servers>

<https://fourkitchens.atlassian.net/wiki/display/TECH/Configure+Varnish+3+for+Drupal+7>

<http://www.rklawson.com/blog/2012/04/14/caching-drupal-7-varnish-apc-and-memcache>