Setting up LiteSpeed Cache Plugin

PagePipe.com

Setting up LiteSpeed Cache Plugin

1. Cache

Verify caching is enabled.

CACHE LOGGED-IN USERS - OFF
We don't recommend activating this option. Disable it. Why would you want to cache logged-in or logged-out users?

CACHE COMMENTS - OFF
We don’t “do” comments on our blogs. So this is unneeded.

CACHE REST API - ON
Should be safe to leave on. Turn off if anything breaks.

CACHE LOGIN PAGE - OFF
Don't bother caching it. You will have issues if you use recaptcha or verification tokens. We dont recommend recaptcha by the way for speed.

CACHE FAVICON.ICO - ON
The favicon ICO is always requested, especially if there is no icon present on the page.

CACHE PHP RESOURCES - ON
Caching those resources reduce the server impact.

CACHE MOBILE - OFF
This is necessary if you're using AMP or a mobile-specific theme. Don’t use these. They slow down your site.

List of Mobile User Agents - OFF
Leave off unless Cache Mobile (above) is on.

Private Cached URLs - OFF
Don’t use. It’s for pages that need to be cached separately for each visitor.

Force Cache URLs - OFF
Should not be needed.

Choose to drop query strings when caching - Defaults are good
The defaults are Facebook click, Google click, Google Analytics, those are good suggestions. Pages served for Facebook then receive a cached URL.

2. TTL

We leave the default values for these settings. Would only change if you very rarely update your website.

DNS TTL (time to live)
This setting tells the DNS resolver how long to cache a query before requesting a new one. The information gathered is then stored in the cache of the recursive or local resolver for the **TTL** before it collects new, updated details. DNS TTL (time to live) is not an instant change or an expiry date like far-futures expiry settings in the .htaccess file. TTL is how long a page or item is cached. The private cache default is 1800 seconds, so 30 minutes. Caching the pages for one week is fine. Your homepage page cache is a week.

If you publish frequently, make sure to lower the amount of time. If publishing once a day; set TTL to 24 hours in units of seconds. If you're using the rest API for a mobile app, you may want to lower this. If you set this lower than 30 seconds, it prevents TTL caching.

Default HTTP status code page
This specifies an HTTP SAS code and the number of seconds to cache these pages.

404 page
Your 404 pages will be cached for 3600 seconds acceptable lifespan. If your 404 page is hammered a lot, this severely reduces the impact on your server.

404 pages are typically bypassed on cache using W3 Total Cache plugin to avoid serving a 200 as a status code. This is no longer needed.

3. Purge

We leave the default values for these settings.

Purge All on Upgrade - ON
You can choose to “purge all” on upgrade. So anytime a plugin theme or WordPress core is updated, the cache gets purged (cleared). We recommend this setting.

Auto purging for publication and updates of post
The default settings are good for most sites. If you have shortcodes or widgets on lots of pages, you can select “All Pages” to ensure that all caches are cleared and no old cached content remains displayed

Serve Stale - OFF
The stale copy of a cached page will be shown to visitors until a new cache copy is available.

Scheduled Purge URLs and Time - NONE
You can set specific pages to purge at a specific time. For example a 1 am scheduled purge may give the optimal load time. Off-peak purges save server resources.

4. Exclusions

Exclusions allow you to exclude a variety of posts, pages, cookies, and user agents from the cache. You can also choose to not cache for roles.

We suggest excluding any pages with forms and e-commerce functionality. Here is what we exclude on a typical site with Easy Digital Downloads:

/checkout/
purchase-confirmation
^/downloads/
/wordpress-speed-tuning/   <<< transaction pages

Woocommerce checkout is excluded automatically.

5. ESI Settings

ESI -- edge side includes.
This allows designating parts of dynamic pages as separate fragments. This is called fragment caching. It can reduce the load time and the server load. It's complicated to set up. There is documentation on how to use this. It is a developer feature, and not user-friendly.

Enable ESL - OFF

Cache Admin Bar - ON

Cache Comment Form - OFF
We turn comments off but you may wish to leave this ON if you allow comments.

ESI Nonce - Default

Vary Group - Default

6. Object Cache

Object cache is not enabled unless your webhost offers it or you maintain your own server and you have enable it.

Object Cache - OFF for most people

Method - Default

Host - Default

Port - Default

Default Object Lifetime - Default

Username - None unless you have SASL installed

Password - None unless you have SASL installed

Redis Database ID - Default

Global Groups - Default

Do Not Cache Groups - Default

Persistent Connection - ON

Cache Wp-Admin - OFF

Store Transients - ON

7. Browser Cache

Browser Cache - ON

Browser Cache TTL - 31557600

Notes
Browser cache controls far-futures expiration of CSS, JS, images and makes sure that they're stored in the browser.

8. Advanced

Login Cookie - Default
There's a default login cookie. Don’t mess with this unless you have multiple sites sharing the same domain name (one in a sub directory).

Improve HTTP/HTTPS Compatibility - OFF
Best to sort out any mixed content warnings at the root of the problem.

Instant Click - Usually OFF
Hovering over a link begins to pre-load the page. This tells the browser to begin downloading before the user clicks the link.

The downside is it increases server load. And may cause performance issues. If you have content organized in a 4x4 grid of post featured images, and on hover the title is revealed, hovering to read the titles may trigger downloading three posts in the background. That consumes server resources and slows down the browsing speed. Now the browser is fetching three pages worth of assets, and it didn't need to visit them. This is a tricky feature; test it.

CDN

ALL OFF
We don't use CDN band-aids on speed sites. So we ignore the CDN settings and Cloudflare APIs.

Image Optimization

A QUIC cloud feature, allows image compression. It requests optimization via scheduled optimization (a chron job).

We don't use LiteSpeed's image optimization functionality.

Auto pull cron
Disabling this stops the cron job responsible for sending optimizing images offsite to a cloud image server. Your server offloads the images to QUCI-cloud paid services. They do the compressing and resizing on their server. It's good when you have low CPU power. We don't use it.

If you don't have this option activated when using this feature, you can't retrieve those images. It won't prevent them from working. They'll remain on the remote server.

Best practice is uploading the original image and backing up. Then remove the original. You need a way of restoring images in case the image quality degrades. A safeguard is restoring from your off-site backup.

WebP images
We don’t recommend them. Here’s why:

WebP is wasted effort. Some guys chase this lossy format with fervor. From our real-world testing, it makes little difference -- immeasurable.

Why? Because a 10-percent improvement in image weight is not even a 1-percent gain in speed. Images load in parallel. We've never observe the big gains claimed -- even from page weight. But we suspect Google advocates are not comparing apples to apples -- or even to fruit. You can do it if you want. No harm. But WebP isn’t the secret ingredient saving your speed bacon.

There is one rarely mentioned benefit from WebP: It's hard to steal your images. WebP images require special processing to get them back into JPEG format again. Now that tediousness is worth bragging about. But if you want people to steal your images because they're watermarked with your URL, then it's not goodness.

Two of our clients use this steal-my-image strategy. Both sell interior-decoration services. One in the UK and one in Australia. It's a strange world. They want people to share. Put it on Pinterest.

Also, have you ever tried uploading a WebP image to the WordPress media library? We've never seen one appear in the images. WordPress does not support the uploading of WebP images directly to your media library. ”Sorry, this file type is not permitted for security reasons.” That means you are now addicted to the WebP service and remote server. Will that come back to bite you?

Here's a plugin workaround:
https://wordpress.org/plugins/allow-webp-image/

We’re amazed at omitted and neglected details advocates of WebP fail to mention.

WordPress image quality control
By default, WordPress compresses JPEG images to a quality of 82. If you want to disable it, set it to 100.

Page Optimization

CSS Settings

99% of the time we leave everything OFF. Minification and concatenation often result in slower load times and broken pages. So make sure you test if you turn anything here on.

CSS Minify - OFF

CSS Combine - OFF

CSS HTTP/2 PUSH - OFF
You can enable HTTP-2 push if you have a server that has the HTTP/2 protocol enabled. This starts downloading these resource right away, instead of waiting. Most of our clients don't have HTTP/2.

Load CSS Asynchronously - OFF
Hurts user experience by causing a flash of unstyled content (FOUC) while your page loads.

Generate Critical CSS - OFF

Generate Critical CSS in Background - OFF

Separate CCSS Cache Post Types - NONE

Separate CCSS Cache URLs - NONE

Inline CSS Async Lib - Usually OFF
If on, make sure you test for a flash of unstyled content.

Font Display Optimization
LiteSpeed Plugin offers two choices, Default or Swap.

The Default setting will use your browser's "Block" method and briefly hide the text until the font has downloaded.

The Swap method will initially show a fallback font until your chosen font is downloaded.

Test to see which works better for your site.
Reference: https://css-tricks.com/almanac/properties/f/font-display/

JS Settings

JS minify - OFF

JS Combine - OFF

JS Combine External and Inline - OFF

JS HTTP/2 Push - OFF

Load JS Deferred - OFF

Load Inline JS - Default OFF

Optimization

CSS/JS Cache TTL - Default
During site development, it can be nice to decrease this number so you don't have to purge the cache manually all the time. Otherwise, we leave it as is.

HTML Minify - OFF

DNS Prefetch - Make sure you test
Tricky business to preload DNS requests for external domains. So, for instance, Google fonts typically load fonts.googleapis.com as their DNS prefetch. You can also use fonts.gstatic.com.

How to find them? Open your site in a private window. Right Click > Inspect > Sources - Now add all the external domain names you see to the DNS prefetch list.

DNS Prefetch Control - Usually OFF

Remove Comments - Usually OFF
This removes comments found inside CSS and JS files. May improve performance because it makes file sizes smaller.

Remove Query Strings - OFF during development, then ON
Google ReCaptcha is bypassed because it breaks with this option. Assets that have query strings added are css and js files. Removing Query Strings produces better test scores but not necessarily better speed.

Load Google Fonts Asynchronously - Usually OFF. Make sure you test if ON

Remove Google Fonts - ON

Remove Wordpress Emoji - ON

Remove Noscript Tag - Usually OFF
Noscript tags are used for compatibility with older browsers that don't support JavaScript. Probably safe to turn on but won't improve load times much. We don't use it.

Media Settings

Lazy Load Images - ON
Load images only when they enter the viewport. We like lazy load, especially if you add some CSS so that your images fade in nicely. WordPRess has lazy load built into core since August 2020.

Basic Image Placeholder
Use this transparent placeholder: data:image/gif;base64,R0lGODlhAQABAIAAAAAAAP///yH5BAEAAAAALAAAAAABAAEAAAIBRAA7

Responsive Placeholders - Usually OFF
Helps reduce page layout shuffling when images are loaded. We've found that this is unnecessary.

Responsive Placeholder SVG

<svg xmlns="http://www.w3.org/2000/svg" width="{width}" height="{height}" viewBox="0 0 {width} {height}"><rect width="100%" height="100%" fill="{color}"/></svg>

LQUIP Cloud Generator - Usually OFF but test
A low-quality image placeholder: unrecognizable on purpose. When the image loads, its progressive. It's better user experience because it feels more natural.

LQUIP Quality
Experiment with this setting.

Generate LQUIP in Background - Probably ON
Make sure to test this first.

Lazy Load Iframes - ON
Good for frames or video embeds.

Inline Lazy Load Images Library - OFF

Media Excludes

You can choose to exclude a variety of images and iframes based on classes, names, parent classes, URLs all in the media excludes for the lazy load option.

Localization

Gravatar Cache - Usually OFF
Turn on if you have tons of comments and use Gravatar. Or if you use Gravatar to identify authors on posts. Requesting Gravators from remote servers slow down your pages.

Gravatar Cache Cron - OFF unless you are caching Gravatar

Gravatar Cache TTL - Default is OK or increase

Localize Resources
Third party scripts can be “localized.” Worth trying if you have scripts from external sources.

Localization Files
Add the external URLs that you want to localize. Must be .js files.

Tuning

Since we don’t minify or combine CSS or JS, we leave all tuning settings on their default values.

Database

Manage
We run the “Clean All” function periodically to tidy up the database.

DB Optimization Settings
Defaults are fine.

Table Engine converter
This allows conversion to a known database for better performance.

Database Summary
Autoload data size should never be larger than one megabyte. If it is, you'll have performance issues. The autoloaded data is mostly plugin settings, and entries may remain even if you’ve deleted a given plugin. You can delete unused data by accessing your php myadmin tool via cPanel or similar.

Crawler

The crawler must be enabled at the server-level or virtual host level. So probably only useful if you run your own server. Overkill for most websites.

sitemap
Overkill. Sitemap features are built in to WordPress core since August 2020.

set the crawler cron delay
A reasonable interval is 500 microseconds for most hosting platforms. The rest of the defaults are acceptable. Crawler cron is most important for automatic preloading.

crawler simulation
Allows the crawler to fake being a logged-in user to preload pages. And then, you can include a sitemap. The crawler is complicated. We don't use it.

All you need in general settings is turn on the crawler cron, and then if you wanted to preload a logged-in user, enter the user id. Paste the URL of your sitemap. We don't use this.

Toolbox

We don't typically mess with any of these. Occasionally we'll purge a specific page from the Toolbox

Purge
Select a specific page or pages to purge.

Import/Export
Allows you to import or export your LSC settings.

Edit .htaccess
Proceed with caution. You can ruin your site.

Heartbeat
Use this is you see slow "admin-ajax.php" calls in your speed test waterfall.

Frontend Heartbeat Control
Off unless speed tests reveal slow admin-ajar.php call.

Frontend Heartbeat TTL

Set to 120 if heartbeat if causing problems. Otherwise, leave at 60.

Backend Heartbeat Control
Probably safe to turn on, but we don’t usually need to try it.

Backend Heartbeat TTL
Could increase the TTL or set to 0 to disable entirely. Proceed with caution.

Editor Heartbeat Control
Safest to leave this “Off” so your pages autosave frequently.

Editor Heartbeat TTL
If you have a bunch of admins writing on the site simultaneously, you might increase the TTL to reduce server load.

Setting up LiteSpeed Cache Plugin

PagePipe.com