

Why do I keep seeing warmup cache request in my LiteSpeed/WP Rocket logs after every deploy, is it eating my server resources?

TechRounder Help Center PDF

Live article:

<https://www.techrounder.com/help/why-do-i-keep-seeing-warmup-cache-request-in-my-litespeed-wp-rocket-logs-after-every-deploy-is-it-eating-my-server-resources/>

By Vipin PG | Published August 14, 2026 | Updated August 15, 2026 | Topic: Solution | 4 min read

That log line isn't an error. It's your cache plugin doing exactly what it's supposed to do - the moment your deploy clears the cache, LiteSpeed's crawler or WP Rocket's Preload feature kicks in and starts re-visiting pages one by one to rebuild the cache before a real visitor hits an empty one. What you're reading in the logs is just that crawler announcing each fetch.

The real question isn't whether it's happening - it's whether it's happening more than it should, and whether it's actually costing you server resources or just looks like it is.

What's actually triggering it

Both plugins work off the same basic idea, but the mechanics differ:

- LiteSpeed Cache ships a built-in crawler that reads your XML sitemap and re-requests every URL on it once the cache is purged, so the pages are pre-rendered before anyone lands on them cold.
- WP Rocket's Preload Cache does something similar but through non-blocking background requests - it queues URLs from your sitemap into a database table and works through them in small batches, sized automatically based on how fast your server responded over the last few minutes.

Either way, a deploy = a purge = a full or partial re-crawl. That's by design, not a bug.

Is it actually eating resources, or does it just look like it is

Don't judge this from log volume alone - check actual load. Pull up your hosting panel's CPU graph or run 'top'/'htop' during a deploy and see if it correlates with a real spike, not just a burst of log lines.

WP Rocket's preload is designed to never push PHP memory past 90%, and CPU issues generally shouldn't occur with the default setup. That's the vendor's claim, and in practice it mostly holds - but "shouldn't" isn't "can't," especially on cheap shared hosting with tight CPU allocations.

For reference, an independent load test comparing warmup/preload across plugins during a 20-page crawl found LiteSpeed Cache peaking around 17% CPU and WP Rocket around 29%, while a more aggressive tool like RabbitLoader spiked to 85% in the same test. So the plugin you're using matters more than people assume - some are just heavier by design.

Where it actually becomes a problem

1. Frequent full purges. WooCommerce sites are the classic case - since product data can appear almost anywhere on a store, a single order or stock change often forces a near-complete cache purge, which restarts the crawl before the last one even finished. If your deploy pipeline also triggers a full flush every time, you're stacking two of these on top of each other.

2. Shared hosting throttling. Many hosting providers actively disable or rate-limit LiteSpeed's built-in crawler because of the load it generates, particularly on shared plans. If that's your setup, the crawler may be timing out mid-run rather than actually finishing - which explains repeated log entries without the cache ever fully warming.

3. Script timeouts. On shared hosting, the crawler frequently gets cut off by `max_execution_time` limits before it completes its full run, so it restarts on the next cycle and never gets ahead of itself.

Fixing it on LiteSpeed Cache

1. Go to LiteSpeed Cache -> Crawler in your WP admin.
2. Raise the Interval Between Runs so a new crawl can't start until the last one's realistically done.
3. Lower Crawler Threads if you're on shared hosting - fewer simultaneous requests means less load, even if it takes longer to finish.
4. Set a sensible Load Limit so the crawler pauses itself if server load crosses a threshold, instead of running blind.
5. Switch from full-cache purges to tag-based or URL-specific purging where possible, so a single product update doesn't force a site-wide re-crawl.

Fixing it on WP Rocket

You can't fully disable the automatic batch-sizing logic, but you can shrink what it has to crawl:

```
add_filter( 'rocket_preload_exclude_urls', function( $excluded_urls ) { // Skip deep pagination $excluded_urls[] =  
'/page/[5-9][0-9]*/'; // Skip filtered/faceted URLs that don't need caching $excluded_urls[] = '/shop/?filter(.*)'; return  
$excluded_urls; });
```

Drop that in a site-specific plugin (not your theme's `functions.php` - it'll survive theme changes that way). Trimming the sitemap itself - excluding tag archives, deep pagination, or post types nobody visits - has the same effect and is often the bigger win.

If your host has throttled or blocked the built-in crawler

You'll keep seeing purges in the logs with no actual warmup happening. In that case, run your own lightweight warmup via cron instead of relying on the plugin's crawler:

```
#!/bin/bash while read -r url; do wget -q -O /dev/null "$url" sleep 1 done < /home/youruser/urls.txt
```

Schedule it for a low-traffic window rather than immediately after deploy:

```
*/30 2-4 * * * /home/youruser/warmup.sh
```

The 'sleep 1' between requests is doing the real work here - it caps how fast you're hitting your own server, which is exactly what a throttled shared host is trying to protect against.

Quote: Never point a warmup script at logged-in-only pages - cart, checkout, account dashboards. They're not cacheable to begin with, and warming them can leak session-specific content into a shared cache.

One last thing worth checking: make sure your crawl interval is actually longer than the time a full crawl takes to complete. If it isn't, the crawler is permanently restarting before it finishes - which is the single most common reason people see this log line nonstop and assume something's broken.