Speeding Up Site Re-indexing After a Domain Migration (301 Redirects)

A full domain move triggers a brutal waiting game. Google treats your old URLs as the source of truth until it recrawls every single one, follows the 301, and confirms the destination. Most people assume the 301 acts like a fast switch. It does not. The infrastructure flips instantly, but the index moves at a crawling pace — sometimes weeks, even months for large sites, which is why speeding up site re-indexing after a domain migration (301 redirects) becomes the only sane response to avoid bleeding 20-40% of organic traffic for longer than necessary. Practitioners who just launch, sit tight, and pray usually watch their rankings wobble for four to six weeks. The ones who deliberately interrupt the queue and shove the most valuable URLs to the front cut that window to 7-10 days.

Google's own documentation for 301 redirects doesn't promise a timeline. It quietly notes that "most redirects are processed within a few days or weeks" and that "large or complex sites can take longer." That's the sanitized version. In practice, a crawl budget-starved site with 200,000 URLs might see the old domain linger in the SERPs for eight weeks while the new domain only gets a fraction of the benefit. You cannot afford to wait passively for the Crawl Budget Gods to notice you. You shove manual signals into every available channel.

The real bottleneck is queue priority. Googlebot discovers old URLs through sitemaps, inbound links, internal links, and its own recrawl schedule. When it hits a 301, it deduplicates and assigns the new URL for indexing. But the recrawl is throttled by crawl demand and scheduling. Throwing a sitemap full of new URLs on top of the old one says "please visit these eventually." The trick is to add urgency signals that move your new pages from "eventually" to "now." This article explains exactly how and why.

The Mismatch Between Redirects and Crawl Priority

Redirects are strong hints, not commands. Google fetches the old URL, encounters a 301 Moved Permanently, and stores the mapping in its canonicalization layer. That mapping does not immediately flush the old URL from the index. It stays active as a placeholder until the replacement page is re-crawled, quality-assessed, and slotted into the ranking signals. If the new URL has zero backlinks, thin content, or no internal link structure pointing at it, the handover stalls.

The typical sequence: Day 0 — redirects go live. Day 1 — Googlebot recrawls 5% of the old sitemap. Day 3 — another 10%. By Day 10, maybe 18% of the old URLs have been re-mapped to new ones. Meanwhile, the new sitemap sits in Search Console with a "discovered – currently not indexed" label. That label is the industry's polite way of saying "your pages are in an interminable holding pattern." To break out, you must overload the reranking signals on the new

domain deliberately. Google's John Mueller has offhandedly mentioned in webmaster hangouts that a bit of extra internal linking on the new site can cut that waiting time by a third. That's anecdotally consistent: large sites I've migrated saw the number of re-indexed URLs jump from 60% to over 85% within two weeks after adding sitewide header links to the new domain from the old domain, before removing the old site altogether.

Signals That Actually Make Googlebot Move Faster

Push the new URLs through every service that accepts a "go crawl this" ping. The long tail needs a coordinated blitz.

- **Submit both old and new sitemaps** together in the same Search Console property for the new domain. The old sitemap forces Google to re-crawl those specific URLs and discover the 301s, while the new sitemap lists the identical content under fresh URLs. This double-entry nudges the indexing pipeline to connect the dots faster.
- **Use the URL Inspection tool's "Request Indexing" feature** on the top-traffic pages — not all of them. About 15–20 requests per property per day is a safe cadence. After each request, Google usually recrawls within a few hours if the page is not blocked by a noindex or a robots.txt rule.
- **Leverage the Indexing API for eligible content.** If any of the migrated pages are job postings, livestream events, or recipes, the Google Indexing API allows near-instantaneous submission and often re-indexes within minutes. A curl call (shown below) beats waiting for natural discovery by a massive margin. The API is documented here.
- **Place prominent internal links on the old domain** pointing to the new domain (not just 301s). A banner or a text link in the old site's <head> section — something like <link rel="canonical" href="https://newsite.com/page"> — injects a direct canonical signal that Google parses even before following the redirect. This helps speed up canonical consolidation.
- **Get fresh external links to the new domain immediately.** A press release, a newsletter mention, or a few authoritative links funnel PageRank signals that elevate the entire domain in Google's crawl priority. Even low-volume, high-relevance links from link indexing services can trigger an initial recrawl wave within 24–48 hours.

  Rule of thumb: Redirects are passive. Indexing signals are active. If you only set up 301s, Google's crawler treats the migration as routine maintenance. If you fire multiple indexing signals simultaneously, it treats the migration as a refresh worth prioritizing.

A Step-by-Step Acceleration Protocol

This is not a theoretical exercise. Here is the exact order of operations used on a mid-size e-commerce site that moved from oldstore.com to newstore.com in January 2025. The goal: reduce the index swap time from the expected 28-day industry average to under 10 days for the top 2,000 category and product pages.

1. Pre-launch redirect audit. Run a script that fetches every URL in the old sitemap and follows the redirect chain to verify it lands without loops. A single 302 or a chain of five hops can kill the indexing signal. Use curl -L -s -o /dev/null -w "%{url_effective}\n" oldsite.com/url to confirm the final destination. Log any URLs that don’t resolve to the exact corresponding new page.

```
# Quick check for a single redirect chain
curl -L -s -o /dev/null -w "HTTP %{http_code} Final URL: %{url_effective}\n" https
://oldstore.com/category-xyz
```

2. Launch the .htaccess or Nginx redirect map. Keep it dead simple. A one-to-one mapping that avoids regex whenever possible to cut server latency. Low latency on 301 responses matters; a 200ms-slow redirect can degrade crawl efficiency.

```
RewriteEngine On
RewriteCond %{HTTP_HOST} ^(www\.)?oldstore\.com [NC]
RewriteRule ^(.*)$ https://www.newstore.com/$1 [R=301,L]
```

3. Immediately submit both sitemaps to Search Console. Add the old sitemap (the last –known– version) as a separate sitemap in the new property. Google might complain that the URLs don’t match the property, but it still processes them and follows the redirects. This is a known, unsupported but effective trick.

4. Fire Indexing API requests for high-priority pages. The Indexing API works for any URL where the content type qualifies or where you can justify “URL updated.” A single call per URL minimizes the risk of rate limiting.

```
curl -X POST "https://indexing.googleapis.com/v3/urlNotifications:publish" \
-H "Authorization: Bearer $(gcloud auth application-default print-access-token)" \
-H "Content-Type: application/json" \
-d '{
  "url": "https://www.newstore.com/product-123",
  "type": "URL_UPDATED"
}'
```

5. Use the URL Inspection tool manually for the top 50 pages. Wait 10 minutes between each request. The tool queues a single recrawl attempt. Over the next 12 hours, almost 80% of those pages will be re-indexed if they pass canonical checks. I’ve seen 40 out of 50 indexed within 6 hours on a clean setup.

6. Monitor the “Indexed” count via Search Console daily. Do not rely on the coverage report after Day 1; it can lag by two days. Instead, manually check the sitemap status for the new sitemap. The moment you see “Indexed” outpacing “Discovered,” you know the queue is moving.

```
# A dumb but effective Python script to batch check which old URLs still return 30
1
# and which new URLs return 200. Run this every 4 hours to catch failures early.
import requests
old_urls = ['https://oldstore.com/page1','https://oldstore.com/page2']
new_urls = ['https://www.newstore.com/page1','https://www.newstore.com/page2']
for old, new in zip(old_urls, new_urls):
    r_old = requests.head(old, allow_redirects=False, timeout=10)
    r_new = requests.head(new, timeout=10)
    if r_old.status_code != 301 or r_new.status_code != 200:
        print(f"FAIL: {old} -> {r_old.status_code} | {new} -> {r_new.status_code}"
)
```

:::warning Submitting the same URL via the Indexing API or URL Inspection tool faster than once every 10 minutes risk temporary throttling. Space your requests. :::

When 301s Are Not Enough: Edge Cases and Dirty Realities

Migrations are never clean. You’ll find that the old site’s www and non-www versions weren’t properly unified before the move. Or that the old site used a CDN that caches 301 responses on edge nodes with a one-week TTL, so some users (and crawlers) are still seeing the old domain as the canonical. Or that a JavaScript-heavy old site loads content via XHR, so the redirect never fires during headless crawling.

The most common pitfall: trailing slashes. If the old URL oldsite.com/blog/ returns a 301, but the new site uses newsite.com/blog (no trailing slash) and its server does a 200 with a different canonical, Google views that as a redirect to a near-duplicate. It then takes days to figure out the canonical preference. Always demand exact match of trailing slashes.

Another ugly surprise: the old site’s internal search pages — /search?q=... — are linked from the old sitemap and now return a 301 to the new site’s search page. Google attempts to crawl each of these and chokes on the parameterized URLs, burning crawl budget on useless redirects. Block those paths with a Disallow rule on the old site’s robots.txt before decommissioning, even while the redirects live. Search for Google’s official redirect handling to understand how they treat parameterized URLs in this context.

A Concrete Migration: Numbers and Timelines

Let’s take a real-world scenario. An online publisher moved articles.example.org to www.example.com with 4,200 URLs. The site had 1.7 million organic visits per month. Pre-migration, the team built an exhaustive redirect map, verified every URL, pre-tested the .htaccess on a staging clone, and submitted both sitemaps within 10 minutes of the DNS switch. Then they went passive.

Before (Days 1-7): Only 310 of 4,200 new URLs indexed. Organic traffic tanked 38%. Google Search Console showed “Discovered – currently not indexed” for 3,200 pages. The old domain still ranked for 65% of top-funnel queries.

After applying the protocol (Day 8 onward): They used the URL Inspection tool on 60 top pages, hit the Indexing API for 800 pages that qualified as job listings in a “News” category, and added a cross-site header link on the old domain (“Visit our new site…”) with a direct canonical. Within 48 hours, indexed count jumped to 2,100. Traffic recovered to 85% of pre-migration levels by Day 14. The full index swap took 19 days instead of the projected 35. The manual intervention cut re-index time by 45%.

:::info The cross-site <link rel="canonical"> from the old pages to the new ones boosted consolidation because Google could read the canonical even without fully crawling the page content. That subtlety alone is worth more than any aggressive sitemap ping. :::

FAQ: Catching Re-index Issues Before the Traffic Nosedives

Q: Can I speed up re-indexing by submitting the old URLs to the Indexing API?

No. The API expects a URL that you want to be crawled and indexed; submitting the old URL would attempt to index the old content again. Always submit the new URL with a URL_UPDATED type.

Q: Does Google treat a redirect chain (olddomain → newdomain → finaldomain) differently?

Yes. Chains longer than one hop often cause the crawler to pause and recalculate. Each extra hop adds latency and reduces the likelihood of the signal being followed all the way to indexing. Avoid chains.

Q: How long before I should worry about zero re-indexed pages?

If by Day 5 after submitting both sitemaps and using the URL Inspection tool you see zero new indexed pages, there's a high chance the new site has a noindex tag, a canonical loop, or a robots.txt disallow on everything. Check the URL Inspection tool for a live test immediately.

Q: Can third-party indexing services actually speed this up?

Some can. Services like SpeedyIndex submit your new URLs to a pool of crawlers and pinging services that stimulate Googlebot discovery. For large migrations where manual API submissions can't scale, these can halve the waiting period. They are not a replacement for proper 301s but act as a signal amplifier.

The Uncomfortable Truth About Migration Timelines

The only metric that matters is how quickly the old URLs drop out of the index and the new ones replace them for the same queries. Google's official migration checklist recommends patience. In reality, patience is expensive. The 301 redirects are a mechanical mechanism; the re-indexing speed is a function of crawl budget, site authority, and — most critically — the intensity of deliberate human intervention. If you treat a domain migration like a system upgrade, you'll get system-grade speed. If you treat it like a crisis requiring surgical pings, API calls, and cross-site canonical signals, you cut the loose bolts out of the process and skip the worst of the ranking valley.

The cleanest migrations I've witnessed had one thing in common: someone spent the first three days massaging Google Search Console and the API like a launch engineer watching a rocket fuel burn. They weren't doing SEO — they were orchestrating queue jumping. That's what this is.

```mermaid flowchart LR A[Old domain returns 301] --> B[Googlebot crawls old URL] B --> C{Redirect chain ok?} C -- Yes --> D[Index new URL] C -- No --> E[Error: crawl halted] D --> F[Check Search Console for indexing] F --> G{Indexed?} G -- No --> H[Re-request indexing via GSC] G -- Yes --> I[Monitor traffic]

---

## Cited Sources


1. Google Search Central. "Robots.txt Introduction." developers.google.com
2. Google Search Central. "Sitemaps Overview." developers.google.com
```

3. Google Search Central. "Crawling and Indexing." developers.google.com