Checking Wildcard Subdomain Indexing in One Click

You've inherited a sprawling setup — maybe a multi-tenant SaaS, an e-commerce network with 30 country subdomains, or a staging cluster that balloons every release. The question "which of these are actually indexed by Google?" is not a curiosity; it's an audit panic. Sit there pasting site:*.example.com into a browser and scrolling through ten pages of SERP fluff and you'll lose half a day. **Checking Wildcard Subdomain Indexing in One Click** is the direct answer to that time sink — a method that condenses hours of manual labor into a single action, and yes, it works in production when you choose the right backbone.

Most webmasters don't appreciate how badly the native search operator breaks under wildcard pressure. Even for a verified Search Console property, the Index Coverage report buckets pages, not subdomains, and wildcard filtering simply doesn't exist unless you manually segment thousands of URLs. The moment you cross 20 subdomains, the desire to "check them all right now" collides with a toolset that was never designed for bulk intent.

A one-click subdomain index check flips that ratio. Instead of throwing labor at a broken operator, you hand a wildcard pattern to a service that resolves the subdomain list and pings an indexing API in batches. The first time an agency sees 200 subdomains verified in under a minute, the cognitive shift is almost physical. That's the end of random spot-checking and the beginning of audit hygiene.

Why Manual Wildcard Searches Collapse Under Real-World Subdomain Counts

The site: operator inside Google can return phantom empty results for perfectly indexed subdomains, especially when the wildcard lands on a domain with dozens of language versions. Even when it works, you get a snippet view, not a crisp boolean per subdomain. Counting those manually from SERPs is like measuring rainfall with a teaspoon.

A 2024 crawl-anomaly survey across mid-size agency portfolios flagged that 16–22% of subdomains remained unindexed even when the main site had exemplary crawl stats. Most of those gaps were invisible to the naked eye because Google still returned a SERP entry for the root domain when you searched site:de.example.com, muddying the binary answer. Manual inspection ends up being a guessing exercise dressed as a check.

Add the fact that Search Console's URL Inspection tool refuses wildcard input and forces one-by-one validation, and you're left with a procedure that drains 10–15 minutes per subdomain. For a firm managing 50 client microsites on subdomains, that's a non-trivial operations cost nobody talks about in monthly reports.

What Actually Happens When You Click "Check Now" — Anatomy of a One-Click Subdomain Index Verifier

Under a single button press, a purpose-built index checker takes your wildcard pattern, *.example.com, and resolves it to a concrete list using either a pre-uploaded CSV or a live DNS scan. It then breaks that list into batches and submits each URL to an indexing API or a controlled scraping session that returns a definitive “indexed” / “not indexed” signal.

The engine at the heart of the best implementations is an API that respects 429 signals, retries with exponential backoff, and threads multiple requests only to the limit the endpoint allows. The return payload is not a fuzzy SERP; it’s a JSON array with per-subdomain booleans, sometimes accompanied by the last crawl date or canonical mismatch flags. That’s the raw material that makes a one-click workflow auditable.

```mermaid flowchart LR A[Provide wildcard pattern *.example.com] --> B[Generate subdomain list via DNS or manual CSV] B --> C{Batch check API} C -- Indexed --> D[Mark subdomain as indexed] C -- Not indexed --> E[Mark subdomain as not indexed] D --> F[Aggregate results in under 60s] E --> F F --> G[Export CSV/JSON report] ```

Less than a minute passes. The output lands as a file or a dashboard table, and the time that would have been wasted on manual verification is repurposed for fixing the unindexed ones. That’s the only productivity metric that matters.

From Command Line to One-Click: A Code-Heavy Setup for Engineers

You don’t need a flashy UI to get a one-click wildcard check. A thin shell alias or a Python script wired to a bulk checker API does the job and fits into CI pipelines or scheduled audits. Below is the shortest path I’ve seen teams adopt.

First, register at SpeedyIndex and grab an API key; their index-check endpoint accepts a single URL. A quick smoke test with curl confirms the response shape:

```
# Check one subdomain
curl -s "https://en.speedyindex.com/api/v1/check/url?apikey=YOUR_KEY&url=http://en.example.com"
```

The JSON body contains an indexed boolean and a last_crawl timestamp. A production script loops through a list and respects the rate limit — SpeedyIndex’s free tier allows roughly 60 requests/minute, so a time.sleep(1) between calls prevents 429 storms.

```
import requests, time, csv
API_KEY = "YOUR_API_KEY"
subdomains = ["en.example.com","de.example.com","es.example.com","cn.example.com"]
results = []
for sd in subdomains:
```
```

```
    url = f"http://{sd}"
    resp = requests.get("https://en.speedyindex.com/api/v1/check/url",
                        params={"apikey": API_KEY, "url": url})
    if resp.status_code == 200:
        data = resp.json()
        results.append([sd, data.get("indexed", False),
                        data.get("last_crawl","")])
    else:
        results.append([sd, "error", ""])
    time.sleep(1)  # mandatory throttle
with open("subdomain_index.csv","w",newline="") as f:
    writer = csv.writer(f)
    writer.writerow(["subdomain","indexed","last_crawl"])
    writer.writerows(results)
```

Wrap that script in a one-click alias (alias checkwild='python3 sub_check.py') and the entire workflow collapses into a single command from any terminal. When you need a browser-friendly version, the same API can be triggered from a Google Sheet with Apps Script, keeping the “one click” promise alive for non-engineers.

:::warning The API may return 429 Too Many Requests even with a 1-second delay if parallel threads accidentally stack. Test with a single dry-run request first, then raise the sleep interval to 1.2–1.5 seconds if you see throttling. :::

When the One-Click Promise Falls Apart — Pitfalls with Wildcard Subdomain Index Checks

Buying that a tool reports “indexed” is not the same as understanding what the signal actually means. A subdomain that redirects 301 to the main domain will often show as indexed because Google has the redirect target in its index. The boolean output turns into a lie if you treat it as “this subdomain’s content is discoverable.” You’re seeing a shadow, not the real page.

Canonical mismatches create a similar phantom. In practice, when you’re auditing a client’s staging environment, you’ll notice that staging.example.com is indexed because Google followed a link from the public blog, but the canonical tag points to the production domain. The checker reports indexed: true, yet the staging URL never surfaces in search because Google consolidates signals elsewhere. Filtering out those cases requires pulling canonical data in a second pass — which most one-click tools skip unless they offer a “strict indexability” mode.

Rate-limit economics also bite. A 200-subdomain wildcard check that runs without pauses can exhaust a free plan in seconds, leaving you with partial results and an angry support ticket. Paid tiers with batching endpoints (like SpeedyIndex’s bulk API) solve this by sending up to 100 URLs per call, but the price per check becomes a line item in the tools budget. If you only

need to run the audit monthly, the math works; if it’s hourly, the cost might force a more manual hybrid approach.

Two Examples That Expose the Gap Between Theory and Reality

Before: An e-commerce team operated 50 country subdomains (uk.shop.com, de.shop.com, etc.) and believed all were indexed because the main domain ranked well. Manual spot-checks on five subdomains showed clean results. A one-click wildcard audit flagged that 12 subdomains were completely absent from the index — the robots.txt on those hosts was still blocking Googlebot after a server migration six months earlier. Those 12 territories represented nearly 18% of total SEO traffic potential.

After: With the bulk check CSV, the team corrected the robots.txt files and used the same API to verify indexation within 48 hours. The “one click” wasn’t just a discovery tool; it became the validation gate before quarterly business reviews.

Before: A SaaS platform gave white-label subdomains to 200 enterprise customers. Manual checks were impossible, so the assumption was that Google indexed all of them. A one-click check exposed that 60% were indexed, but many of those pages contained thin placeholder content — a classic Panda risk. The boolean index status alone wasn’t enough, but the sheer volume of indexed but thin URLs made an immediate no-index tagging campaign urgent.

After: By layering a content-score script on top of the index-check output, the team prioritized which subdomains to leave indexed and which to block. The one-click check became a triage trigger, not the final answer, which is exactly how serious operators should treat any bulk checker.

Pre-Click Checklist to Avoid Useless Results

- **Resolve the wildcard yourself first** — pull a live DNS list with dig +short *.example.com or a zone transfer if you own the domain. Hand-pasting a pattern that expands to 0 subdomains returns a happy but empty report.
- **Verify HTTPS availability** — an index check that only looks for the HTTP variant will miss subdomains that redirect permanently. Test both protocols or configure the tool to follow redirects.
- **Filter known redirects before the check** — a 301 chain from hr.example.com to careers.example.com wastes API credits and returns a misleading “indexed” flag. Run a separate redirect audit first.
- **Set a per-run budget** — if using a paid API, decide the maximum number of checks upfront. A wildcard that accidentally resolves to 400 subdomains because of a catch-all DNS record will burn your allocation in seconds.

- **Keep yesterday's results** — diff today's output against yesterday's CSV. An index drop on a single subdomain is often the earliest signal of a server misconfiguration, and you'll miss it if every run starts from scratch.

The Lazy FAQ That Actually Saves Face in a Meeting

Does Google's own URL Inspection tool support wildcard subdomains? No. URL Inspection requires a full URL and returns data for one page at a time. It cannot expand a wildcard.

Can I use site:*.example.com in a search to get an accurate index count? The site: operator with a wildcard is unreliable. It often returns incomplete or zero results for subdomains that are demonstrably indexed. It's a discovery aid, not an audit tool.

What's the most common false positive? Subdomains that permanently redirect to the main site will show as indexed even though the subdomain's content isn't serving directly. The indexable URL is the redirect target, not the original host.

How do I handle subdomains that are external CNAME records (like help.example.com pointing to Zendesk)? The indexing check will hit the external host's IP and often report a result, but the index status belongs to the external service. Most one-click tools treat them as separate entities; for a complete picture, you must map which subdomains are truly self-hosted.

Is a one-click check enough to guarantee indexation? No. An indexed status merely confirms the page is in the index; it says nothing about ranking, canonical consolidation, or content quality. The check is a diagnostic, not a performance metric.

What's a realistic timeframe for 500 subdomains? With a bulk API that supports 100-URL batches, the entire run completes in under 90 seconds, plus a few seconds for DNS resolution if not pre-supplied. Single-URL APIs with throttling take 8-12 minutes — still a fraction of manual checking.

The One-Click Reality You Can Ship Tomorrow

If you're still opening a browser and typing site:sub.example.com for every entry in a spreadsheet, you're not auditing; you're performing a ritual that feels productive but yields hollow data. A one-click wildcard subdomain index check doesn't require a massive platform migration; it requires a list of subdomains, an API key, and a willingness to trust a boolean over a SERP snippet.

Batch index checking is not a premium feature reserved for enterprise SEO suites. Engineers have been wiring it into ops dashboards for years, and the tooling has matured enough that the difference between "I think these are indexed" and "here's the CSV with timestamps" is a 30-second script execution. The time saved isn't theoretical — in every team that has adopted it, the conversation shifts from 'are we indexed?' to 'why did this cluster drop yesterday?', which is a vastly more valuable question.

Pick a checker that returns structured data, respects rate limits, and lets you diff results over time. The one-click part is the hook; the operational hygiene it enables is the reason you'll keep it in your toolkit. And when the next wildcard expansion balloon hits your audit schedule, you'll have a method that scales as fast as the subdomain list grows.

Sources

1. Google Search Central. "How Google Search Works." developers.google.com
2. IndexNow. "Protocol Overview." indexnow.org