Engineer

Crawler

🕷 a program that fetches pages to index them

What it is

There is no master list of the web's pages, so a search engine has to go and find them, and the program it sends is a crawler. It starts from addresses it already knows, fetches each one over HTTP, reads the links in what comes back and adds the new addresses to its list. Google's guide to how Search works gives the other names in use, "also known as a crawler, robot, bot, or spider", and RFC 9309, the robots exclusion standard, describes crawlers as automated clients that search engines run "to recursively traverse links for indexing".

Links are the main road in. Google says Googlebot finds new addresses "primarily from links embedded in previously crawled pages", and its guide gives the familiar case of a category page that links to a new post. A sitemap is the second road, a list the site owner hands over directly. A page with neither gives a crawler no way to learn that it exists.

Fetching is the first of three stages Google describes. Crawling downloads the text, images and video of a page; indexing analyses what was downloaded and stores it; serving answers a search from that index. During the crawl Google also renders the page, running its JavaScript in a recent version of Chrome, since without that step it "might not see that content". None of the stages is promised: Google "doesn't guarantee that it will crawl, index, or serve your page", and it "doesn't accept payment to crawl a site more frequently".

A crawler is also expected to say who it is. RFC 9309 has each crawler set its own name, a product token made only of letters, underscores and hyphens, and says the token should appear in the identification string the crawler sends, which over HTTP is the User-Agent header, and that the string should describe the crawler's

purpose. The RFC's example header carries `ExampleBot/0.1` and a link to a page about the crawler, and the matching robots.txt line reads `user-agent: ExampleBot`. The name is a claim, not a credential, since any program can send it.

Not every automated request from Google is a crawl. Its overview sorts its clients into three kinds: common crawlers such as Googlebot, which "always respect robots.txt rules for automatic crawls"; special-case crawlers, which work under an agreement with the site and may or may not follow those rules; and user-triggered fetchers, which usually make a single request because a person asked, and ignore robots.txt.

Use the concept in a decision

Put "What is a crawler?" to work in a real decision. We can help relate the concept to the system you are building or buying. A site or an app that the AI assistants your customers use can find, read and act on, checked on every release rather than promised once.

Explore our services Get in touch →

Test the assumptions behind the term

Use the explanation and linked work to frame questions for your project. We can examine the assumptions with you before a tool is chosen. Built a multilingual Hugo site in 3 languages — 108 template files serving three languages and four formats

Built a multilingual Hugo site in 3 languages Get in touch →

Why it matters

A crawler decides for itself when to come back, and its manners set part of the cost of being found. Google says Googlebot "shouldn't access your site more than once every few seconds on average" on most sites, and its crawlers back off when a site struggles; in the Search guide's words, HTTP 500 errors mean "slow down". Google's crawl budget guide describes the whole mechanism, but it is written for sites of roughly a million pages changing weekly or ten thousand changing daily, and Google says a site whose pages are crawled the day they are published does not need it.

A crawler also reads only so much of each file. Googlebot crawling for Search takes the first 2 MB of any file whose type it supports and the first 64 MB of a PDF, measured before compression, and ignores the rest.

The weak point is the name. Google warns that the User-Agent header Googlebot sends "is often spoofed by other crawlers", so a log line that says Googlebot proves nothing by itself. Google documents two checks. For a single

address, run a reverse DNS lookup, confirm that the host name ends in `googlebot.com`, `google.com` or `googleusercontent.com`, then look that name up forwards and confirm it returns the same address. Google's own example:

```
host 66.249.66.1
1.66.249.66.in-addr.arpa domain name pointer crawl-66-249-66-1.googlebot.com.
host crawl-66-249-66-1.googlebot.com
crawl-66-249-66-1.googlebot.com has address 66.249.66.1
```

For many addresses at once, match each one against the IP ranges Google publishes as JSON files in CIDR notation, one file per kind of client. Both checks rest on the network a request came from, which the sender cannot simply type into a header.

What it looks like in practice

This site's server keeps an access log, and the privacy page says how crawlers are told apart in it. Each line keeps the user agent, "which is how we separate people from search-engine crawlers", and a second step follows: "A crawler that gives a name is checked against the networks its owner publishes, using the blurred address below." That is Google's second method, and it suits what the log keeps. The last part of every IP address is replaced with zeroes before the line reaches the disk, so the full address a reverse lookup needs is never stored.

The summaries treat crawlers as an audience of their own: "the same for everything automated" beside the thirty days kept for people, a daily tally of "so many pages read, so many of them by crawlers, so many reaching the contact page", and a monthly ranking of "which crawlers and automated scanners called". A crawler that names itself honestly is counted under its own name, and a famous name sent from a network its owner does not publish fails the check.

The page on being ready for AI agents starts from the same reader: "An agent starts where a crawler does", at the robots.txt file, the sitemap and the search indexes. Its answer is to make each of them deliberate, down to an IndexNow announcement on every deploy, so the engines that take part "hear of a new page the day it ships rather than at their next visit".

Dates are one way a site talks to a crawler, and they can end up saying nothing useful. The write-up of a sitemap that gave 172 of its 176 pages one timestamp puts the stake plainly: “a modification date is a promise to a crawler that a page’s content changed”. A site that makes the promise for every page at once, it goes on, “is either telling the truth about a full rewrite or telling nobody anything useful”.

Where it goes wrong

Requests that call themselves Googlebot do not come from Google. The log shows heavy traffic under Googlebot’s name, and the owner either lets it through as a welcome guest or blocks the name outright. Since other crawlers often copy that name, neither reaction is safe: the first trusts an impostor, and the second shuts out the real crawler along with it. Check the source network, by reverse and forward DNS for one address or against Google’s published ranges for many, and decide on that rather than on the name.

A published page never shows up in a crawl. Nothing links to it, or it can only be reached after signing in. With no central registry, a crawler learns of an address from links and sitemaps, and Google’s guide names pages behind a login among those its crawler cannot reach. Link the page from somewhere already crawled, list it in the sitemap, and keep anything meant for search outside the login.

The crawl rate drops after a bad stretch for the server. Fewer pages are fetched each day, and changes take longer to be seen. Slow responses, 5xx errors and 429 answers all lower Google’s crawl capacity limit, which climbs back only as responses become stable and quick, and every Google crawler on the host shares what is left. Fix the errors and the latency first; on a site of a few thousand pages, Google’s crawl budget guide is not the place to start.

An unlinked page turns up in a crawler’s requests. The address was never published, perhaps a private draft or a test copy, and it still appears in the access log under a crawler’s name. Google calls it “almost impossible to keep a site secret by not publishing links to it”, since a click from that page to another site can leave its address in a referrer log that gets published. Put anything private behind password protection, which Google names as the way to keep both crawlers and people out.

Further reading

- RFC 9309: Robots Exclusion Protocol — the standard's view of crawlers: product tokens, the User-Agent string and the rules a crawler is asked to honour.
- In-depth guide to how Google Search works — discovery, crawling, rendering and indexing as Google describes them.
- Overview of Google crawlers and fetchers (user agents) — the three kinds of Google client, the file size limit, protocols and caching.
- Verify requests from Google crawlers and fetchers — the DNS lookups and the published IP ranges, with worked examples.
- Optimize your crawl budget — how Google sets its crawl rate, and which sites need to tune it.
- Googlebot — the smartphone and desktop crawlers, how often Googlebot visits, its file size cutoff and why an unlinked page is not secret.

Glossary

Choose for your actual constraints

Describe the workflow, the existing interfaces and the users who need the change. We can define a testable delivery and a handover that lets your team maintain it.

See what it costs Get in touch →

Make the terminology practical

Ask how "What is a crawler?" applies to your organisation. A concrete example from your current system is enough to start the discussion.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/glossary/crawler/