curl user survey analysis 2025

Version: 1.0

Daniel Stenberg, July 3, 2025

About curl

curl is an established and mature open source project that produces the curl tool and the libcurl library among other things. While a small project, with just a few maintainers, its products run in many billions of Internet connected devices, applications, tools, games and services. curl is one of the world's most widely used software components.

curl was first released in March 1998, building on its predecessors originating back to November 1996.

To remain relevant and to maintain a place in people's toolboxes, curl must keep up. Keep up with what users want, with how internet transfers are done, with Internet standards and with protocol development.

About the survey

Background

curl features no telemetry, and the curl website has no logs and no tracking. Most users even download and install curl from elsewhere and not directly from us. In order to get proper feedback and get a feel for what users think, asking questions like this is the only viable way.

We run a curl user survey annually in an attempt to catch trends, views and long term changes in the project, its users, its surrounding and in how curl fits into the wider ecosystem. This year, the survey was up 14 days from May 19 to and including June 1. This was the 12th annual survey.

The survey was announced on the curl-users and curl-library mailing lists (with reminders), numerous times on Daniel's Mastodon (@bagder@mastodon.social) on LinkedIn and on Daniel's blog (https://daniel.haxx.se/blog). The survey was also announced on the curl web site at the top of most pages on the site that made it hard to miss for visitors.

Bias

We only reach and get responses from a small subset of users who voluntarily decide to fill in the questionnaire while the vast majority of users and curl developers never get to hear about it and never get an opportunity to respond. Self-selected respondents to a survey makes the results hard to interpret and judge. This should make us ask ourselves: is this what our users think, or is it just the opinions of the tiny subset of users that we happened to reach this year. We simply have to work with what we have.

Many statements listed in this survey analysis are verbatim quotes from respondents. We in the curl project do not necessarily agree with those or think the same way. Some might even be downright offensive. Beware.

Stability

For several years we have witnessed how the responses to the surveys are strikingly similar year-to-year even while the majority of the people who respond say they did not answer the survey the previous year. It might imply that the responses are indicative for a wider population.

Hosting

We use a service run by Google to perform the survey, which leads to us losing the share of users who refuse to use services hosted by them. We feel compelled to go with simplicity, no cost and convenience of the service rather than trying to please everyone. We have simply not put in the effort to switch to an alternative provider for the survey.

Analysis

This analysis is now for the first time done entirely using markdown, perl scripts and gnuplot, with conversions to HTML and pdf using pandoc. This should hopefully make it easier and quicker to repeat this work in the coming years. We can also easily do updates, fix typos etc since everything is in git.

You may also notice that this year I have included the year 2025 in many image titles and a semi-transparent curl logo symbol in the graphs. This, to make sure that the images don't lose that context if they are copied and shared elsewhere.

10 take-aways

1. Linux is the primary curl platform
2. HTTPS and HTTP remain the most used protocols

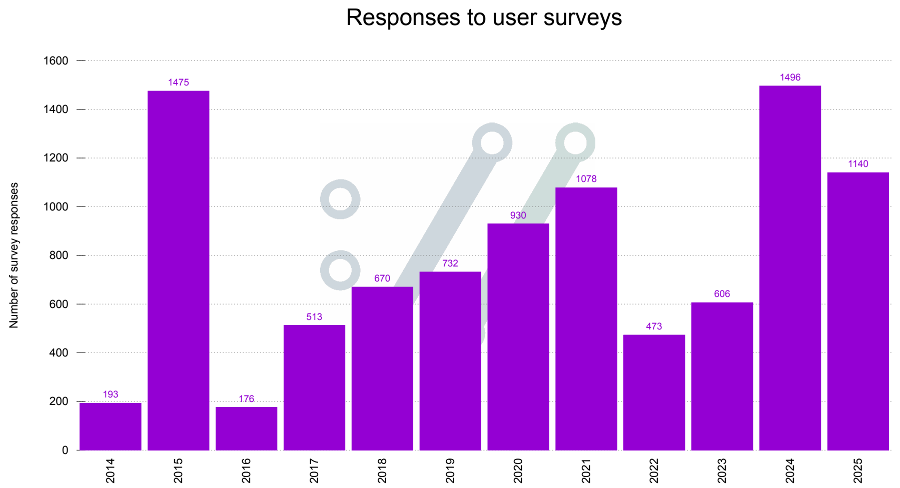
3. Windows 11 is the most used Windows version people run curl on
4. 32 bit x86 is used by just 7% of the users running curl on Windows
5. all supported protocols are used by at least some users
6. OpenSSL remain the most used TLS backend
7. libssh2 is the most used SSH backend
8. 85% of respondents scored curl 5 out of 5 for “security handling”
9. Mastodon is a popular communication channel, and is wanted more
10. The median used curl version is just one version away from the latest

Responses

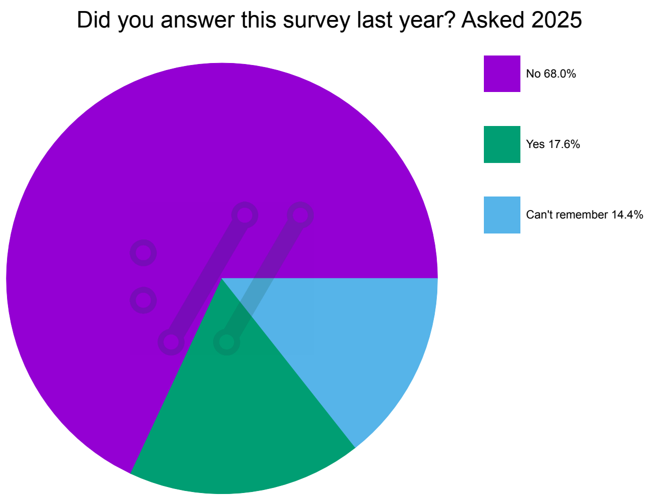
This year we managed to get answers from 1,140 individuals. Less than last year, but still decent. We did everything we could to make users respond to the survey.

Since respondents are free to skip individual questions, each particular question section below will list how many that actually responded.

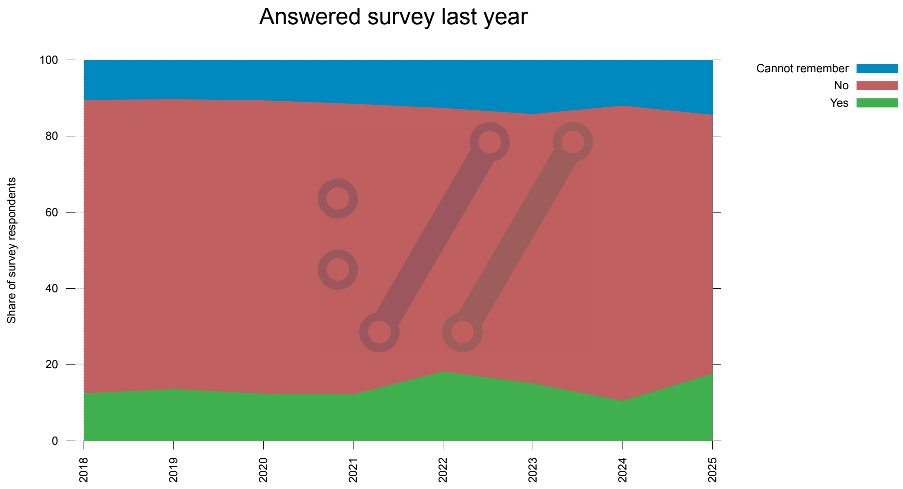
A big thanks of course go to all you who donated valuable time to give us this feedback that then allows me to do this analysis and draw conclusions. It truly helps.



As per usual, most respondents say they did not answer the survey last year, which is partly good because it makes sure we get a lot of fresh answers but it is also a failure that we cannot manage to reach out to the same people again.



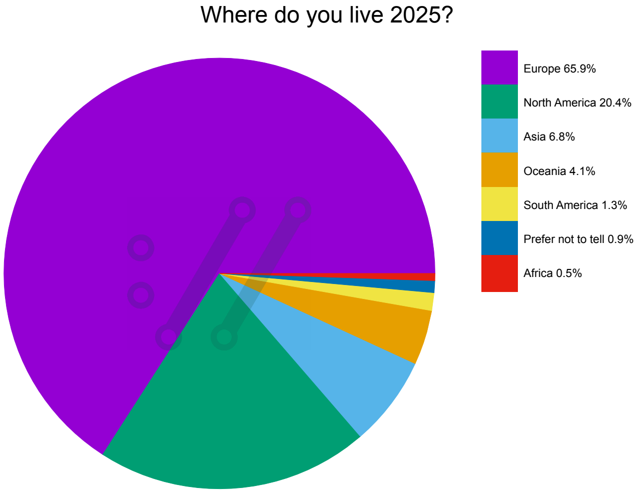
As can be seen in the trend graph, this year was quite similar to past years:



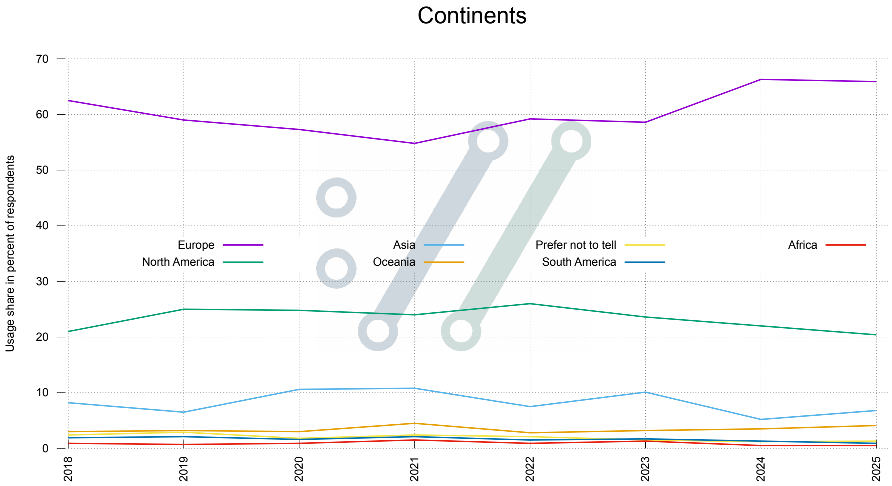
Continents

1,120 responses

The respondents come primarily from Europe, almost two thirds. It could be noticed that among the top contributors to curl, Europeans are in a majority as well.



The distribution over specific continents has remained oddly stable over past surveys as well.



Protocols

1,130 responses

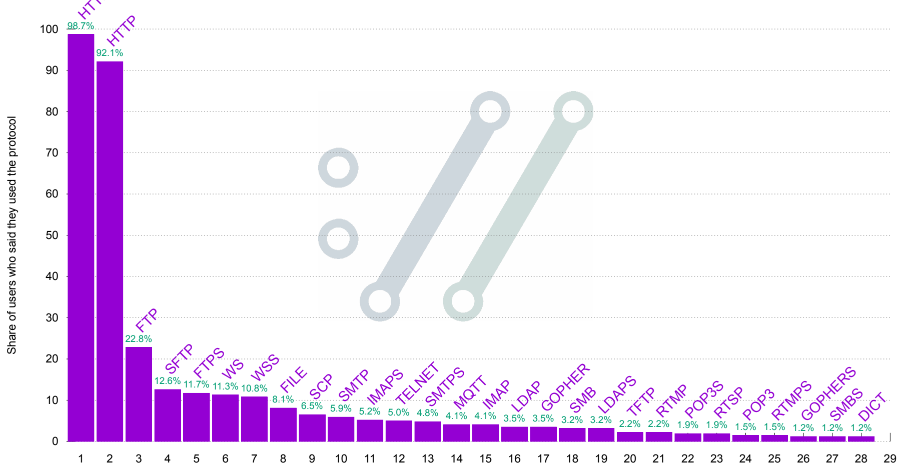
In this survey, we consider URL schemes to be protocols for simplicity. Which of the twenty-eight supported ones did the respondents use? Or at least say they used, because in the curl team we often doubt some of the numbers in here.

HTTPS and HTTP are the undisputed protocol kings in the curl universe, and the ones following are fairly stable as well. The steep usage difference between number two and number three is really visible in a bar graph like this.

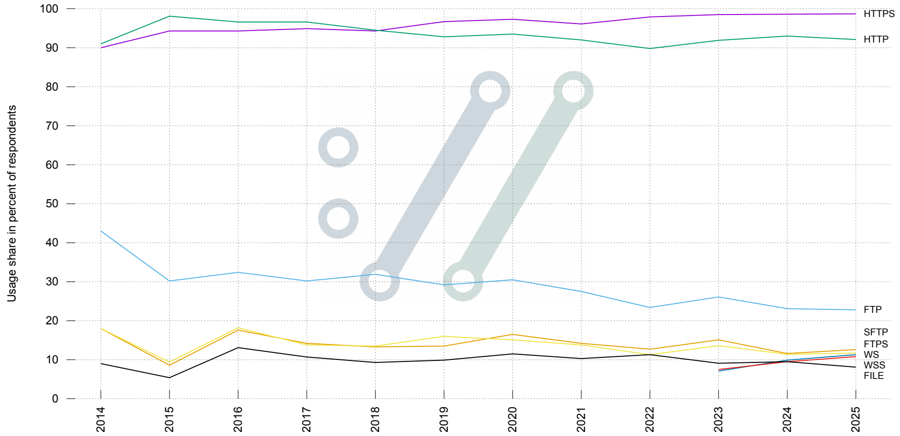
FTP use has gradually been shrinking for many years. The WebSocket protocols that were newcomers in the curl family in 2023 are now the 6th and 7th most used.

No less than *seven* protocols are below two percent usage and in fact only seven are used more than by ten percent of the respondents.

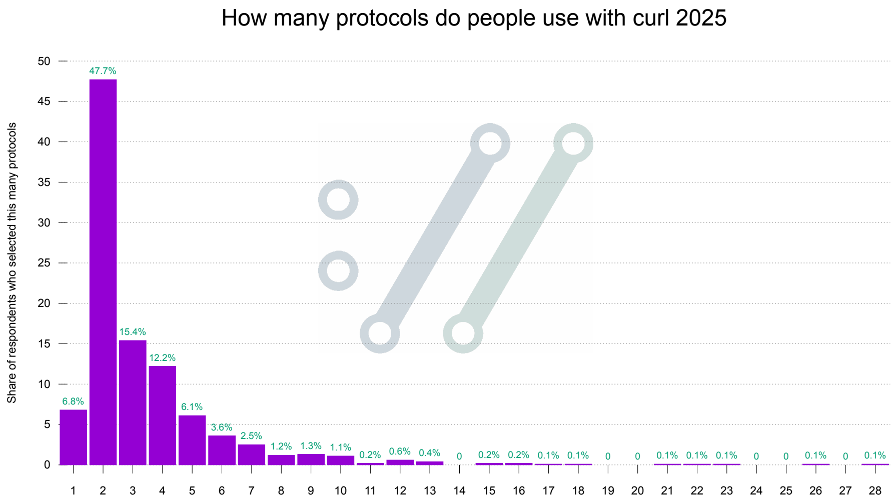
Protocol use 2025



Top-8 used protocols development



The median number of protocols people use curl for is two, which certainly means that for most users it only does HTTPS and HTTP. It also means that 45% of users use *other* protocols as well in addition to those two.



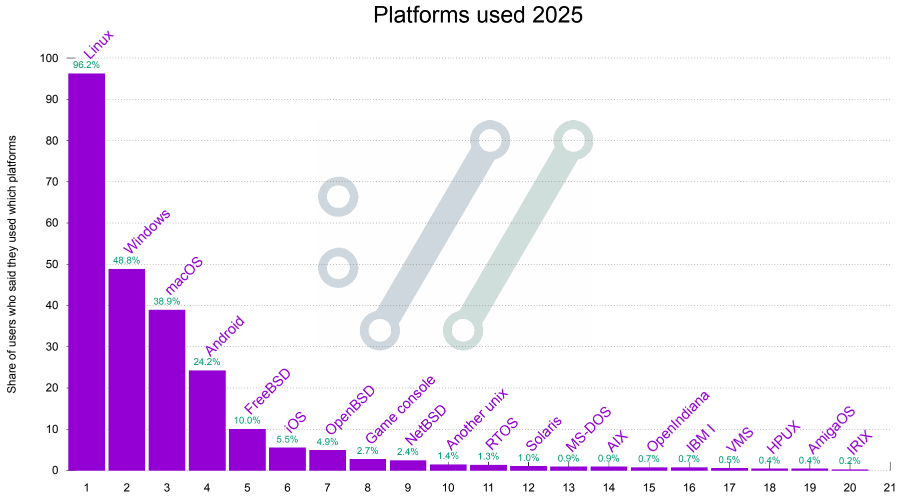
Platforms

1,132 responses

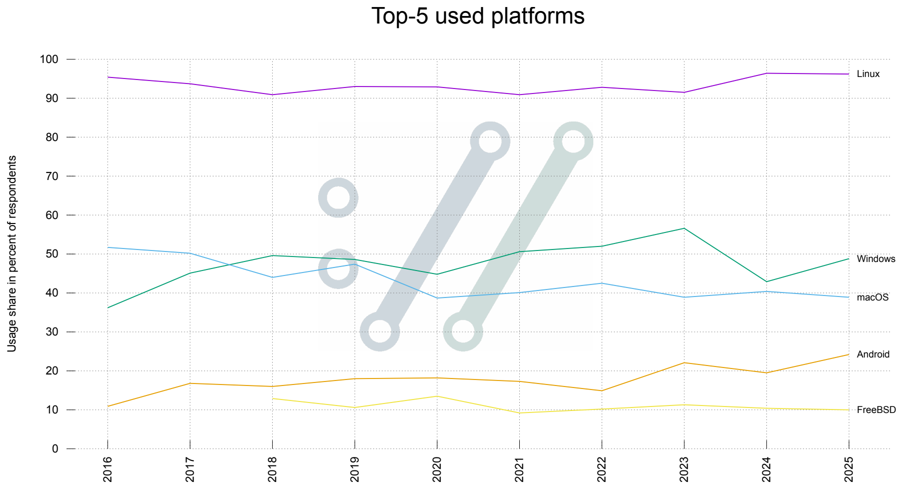
At 96.2% share, almost every survey respondent uses curl on Linux as it remains the top platform curl is used on. The top-5 platforms remain the same, in the same order as they have been for the last several years. I cannot say that I see any reason this is likely to change in coming years either…

Windows bumped back a little from their quite drastic “fall” last year.

As always, it is interesting to see and perhaps questionable, if the ones selecting the most ancient operating systems in this question are actually telling the truth.



Platforms used 2025

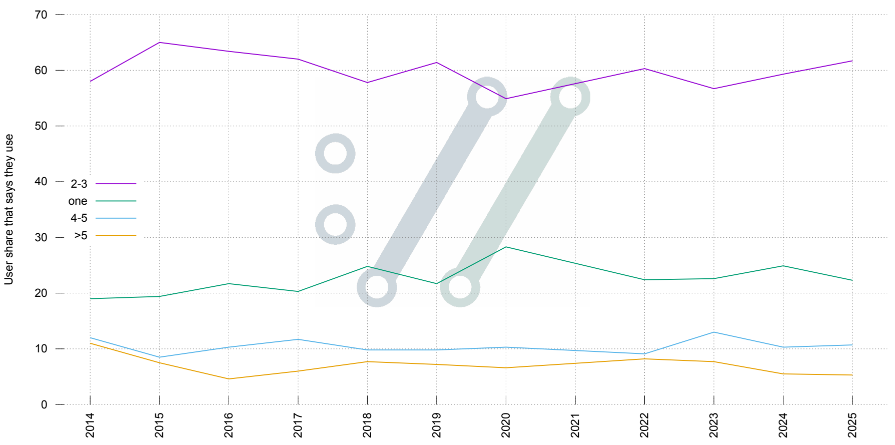


Top-5 used platforms

Number of platforms

1,128 responses

How many platforms do you use curl on? Less than a quarter of the respondents use it only on a single platform.

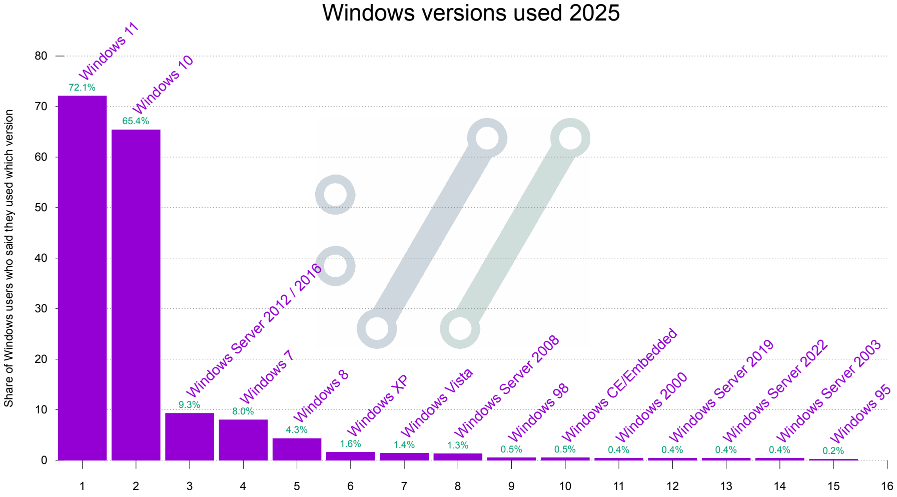


Number of used platforms 2025

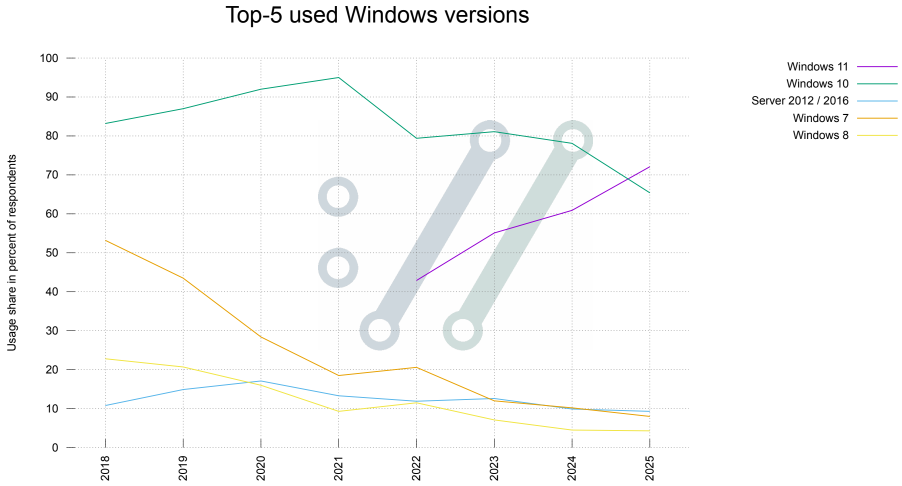
Windows versions

560 responses

This year Windows 11 finally overtook Windows 10 as the most common Windows version used when curl runs on Windows. Should we believe that there are still users running curl on Windows 95?

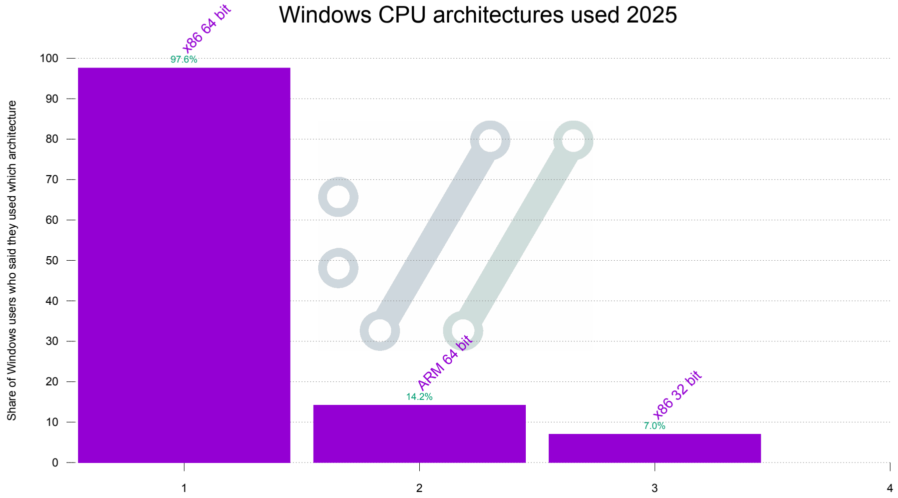


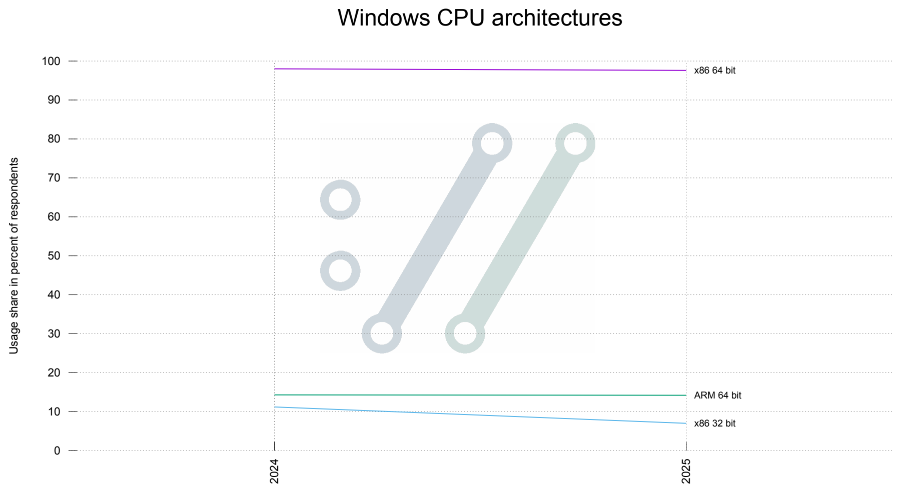
Windows versions used 2025



588 responses

Which architecture users are using curl on Windows on was a new question in 2024 so there is not too much history here, but we can still see how the 32 bit x86 usage shared dropped a little to a mere 7% now.



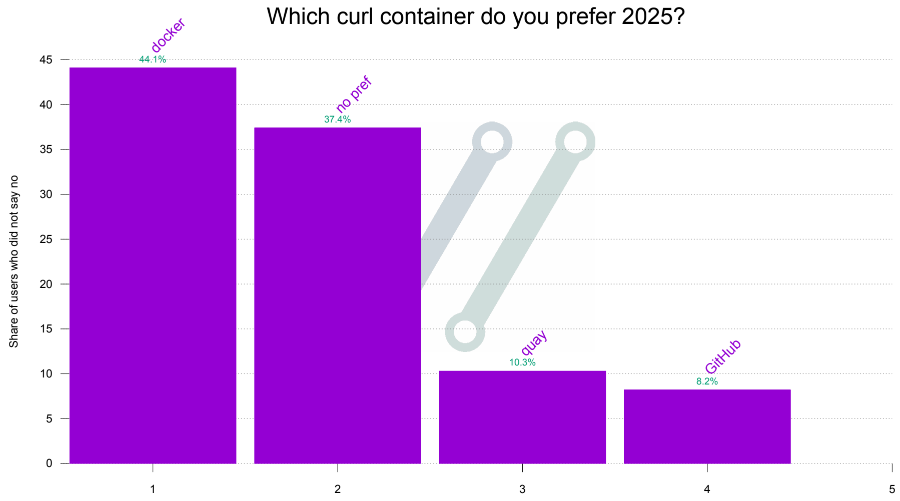


Container

1,089 responses

We have been providing container versions of curl for several years by now. Is it used and which flavor do people prefer?

80.5% of respondents had not used it, but that means that almost out of five (19.5%) has! Many do not have a preference for which to use, but among those that do, docker is a clear winner.



Building curl

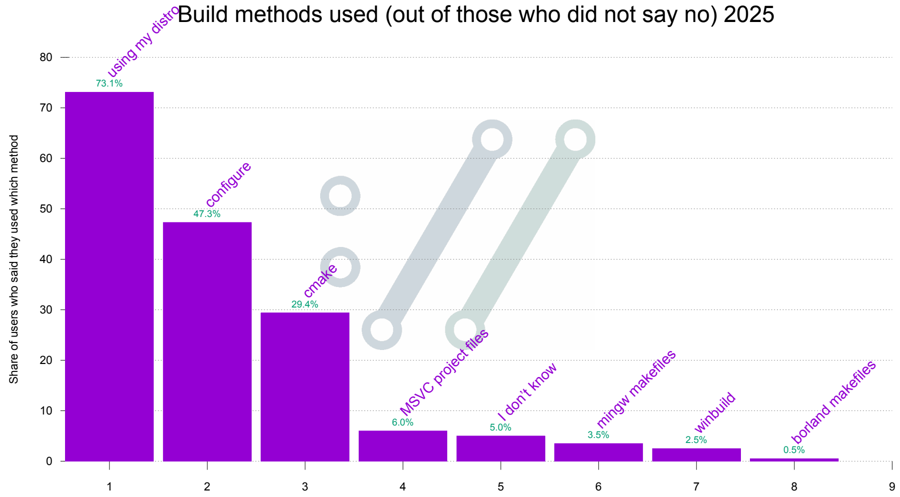
1,127 responses

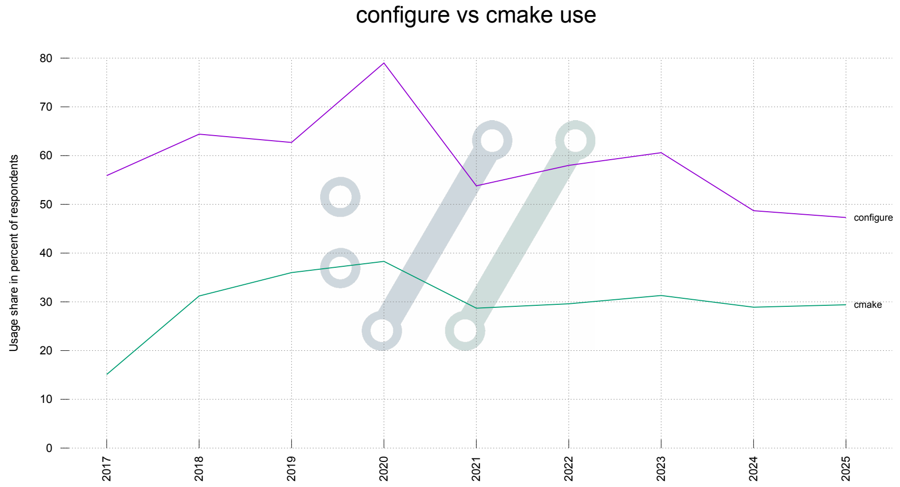
79.9% answered they do not build curl themselves. I have excluded those from the selection out of which I made the graphs below for.

configure remains the top choice of build flavor if we exclude the even larger group of people that build curl using their distro - who then presumably just run with what they use.

There *might* be a longer trend that makes the configure share slowly go down, so this is something to keep an eye on in the coming years.

I must say that it is curious that %5 don’t know.



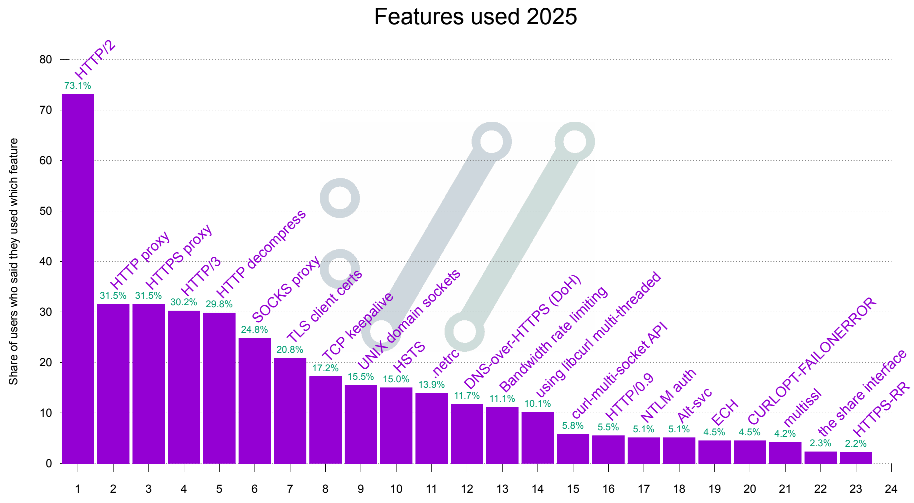


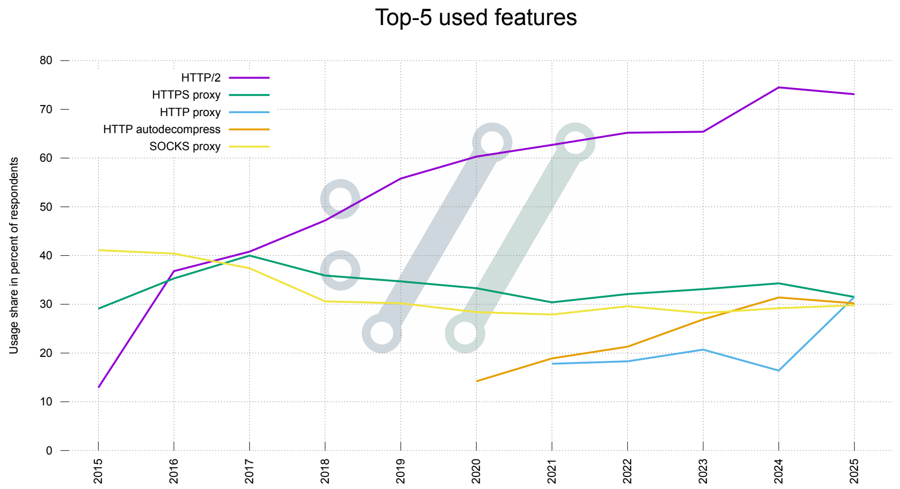
Features

908 responses

What do people do with curl? The answers to this question often raises some eyebrows in the style of "huh, that many use X?" and "huh, only Y use feature F!".

This year we see HTTPS proxy make a fair climb (to 31.5%) as HTTP/2 remains much used (73%) and HTTP/3 stays at 30%.





TLS backends

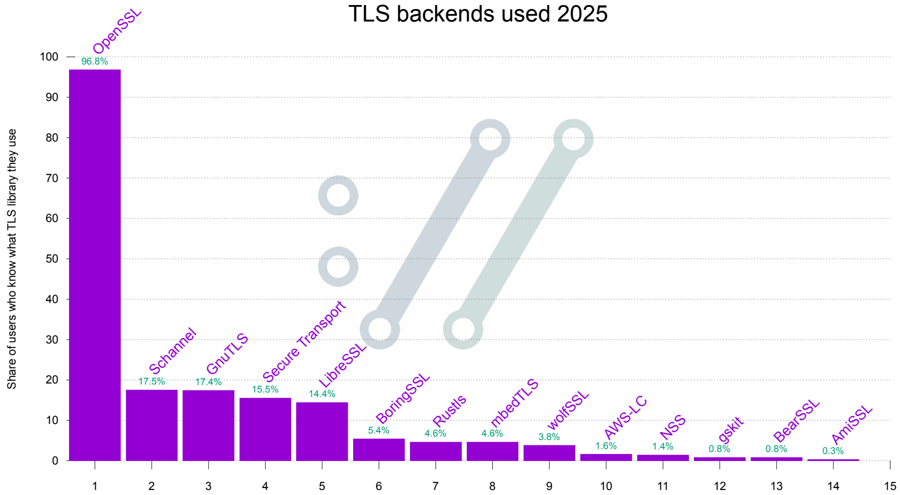
1,096 responses

At the time of this year's survey, curl supported 13 different TLS backends in its latest release. But of course, respondents might have used older curl releases that still had support for now removed TLS libraries.

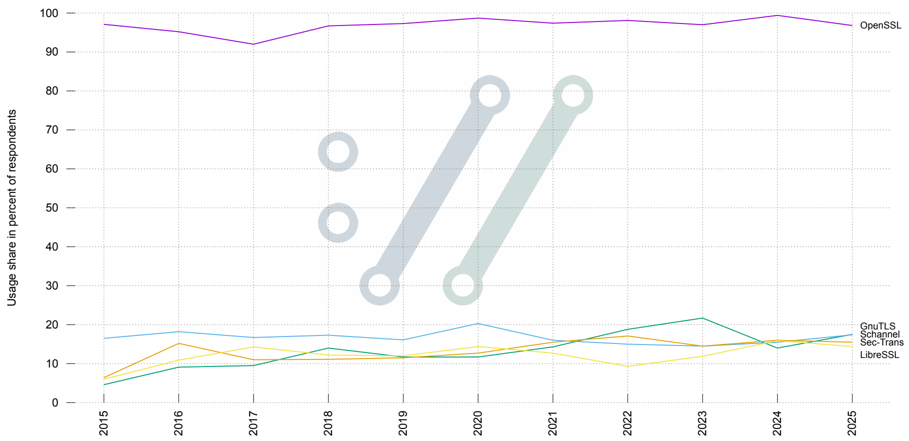
Interesting this year is perhaps that as I write this analysis, we have dropped support for Secure Transport and BearSSL. Secure Transport is the 4th most selected backend this year at 15.5%.

As always, OpenSSL leads this race far ahead of everyone else. This however with a minor caveat: many users of *OpenSSL forks* may also wrongly select this, thus artificially inflating this a bit.

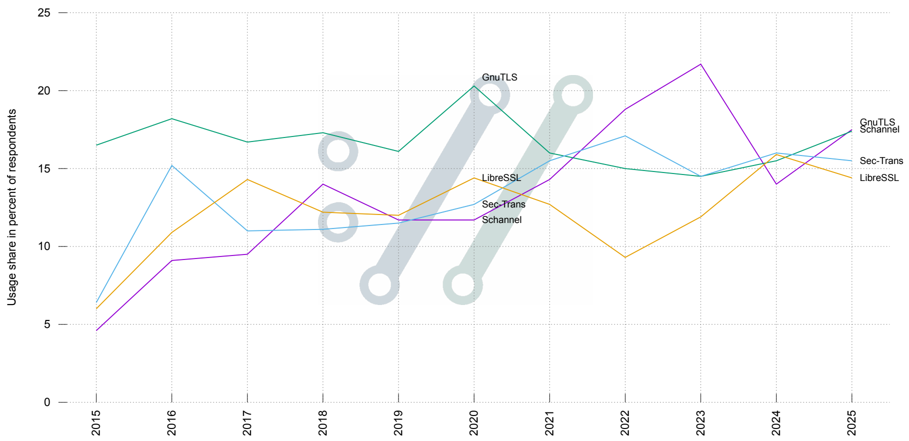
Because OpenSSL is in a league of its own here, I also produced a graph that compares the other most used libraries when OpenSSL is removed from the graph to make them a little more visible.



Top-5 used TLS libraries



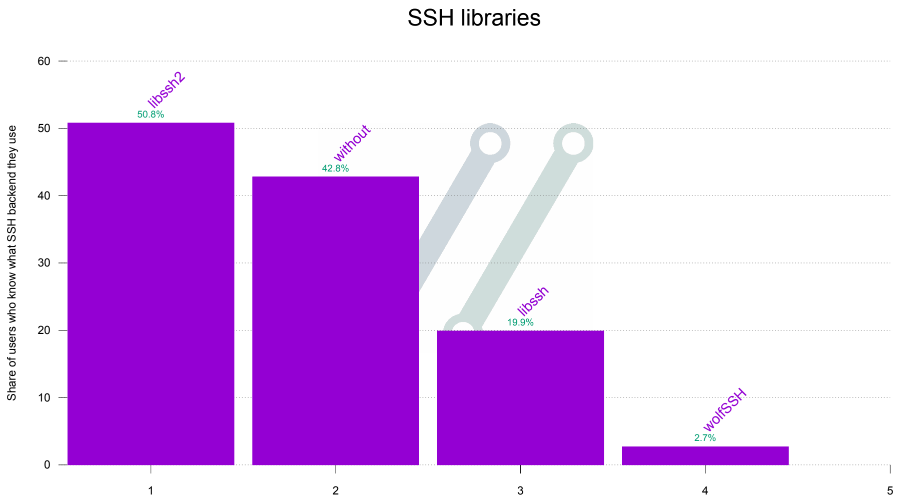
top-2 to top-5 used TLS libraries (wo OpenSSL)



SSH backends

930 responses

curl supports three SSH backends and this is the first year we ask about which SSH backend they use. libssh2 is the winner.

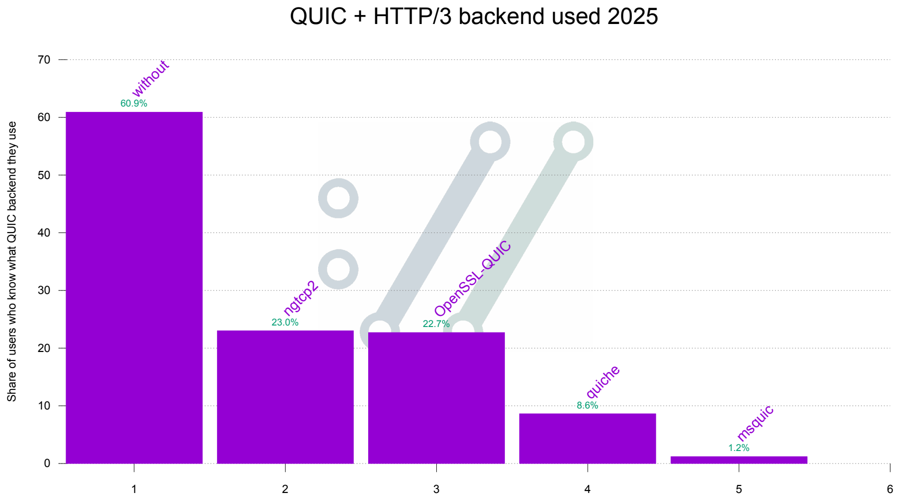


QUIC and HTTP/3 backends

925 responses

curl supported four different QUIC backends at the time the question was asked.

The ngtcp2 one is the only non-experimental backend among the bunch. I suspect a certain number of respondents selected OpenSSL-QUIC erroneously, not being able to tell the difference. The msquic backend does not really work so the 1.2% who selected that option must be special people.



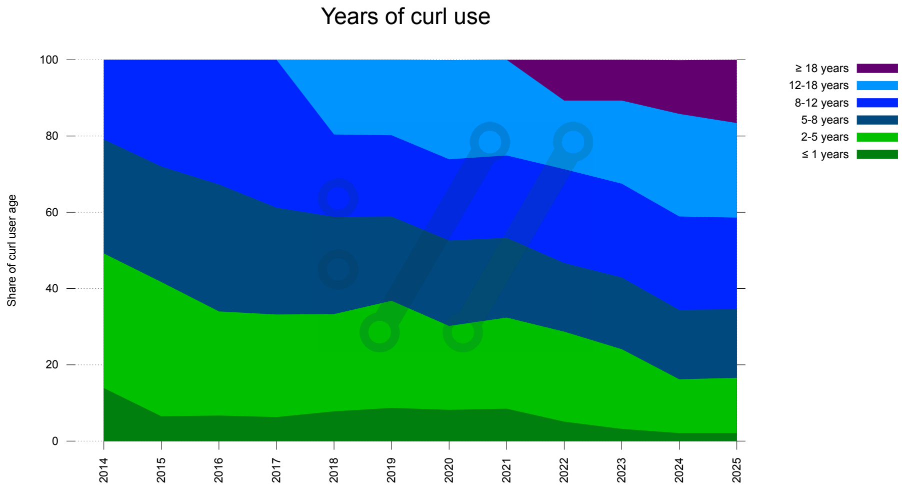
curl use

1,107 responses

This is the year curl turned 27 years old. (Even though sometimes we count the birth from the first date in 1996 when the precursor httpget was released.) The general sense is that we have quite satisfied and loyal users. curl mostly delivers on its promises and is a solid and trusted tool. Many of our users have used curl during their entire computer lives.

A few years back we introduced an answer alternative for 18 years or more as previously 12 years or more was the top alternative. It makes sense that the older alternatives get more answers over time and I think it is good news that we still get newcomers.

Every 6th respondent says they used curl for 18 or more years!



curl version

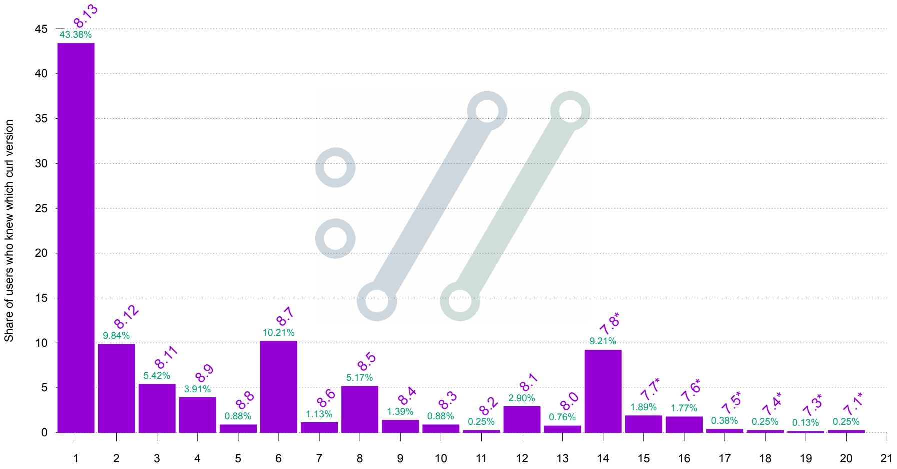
981 responses

When the survey went up, curl 8.13.0 was the latest published release, even if there was an rc build of 8.14.0 out (which many people mentioned to me they actually used but could select in the survey).

Almost two thirds, 63.4% use versions from the last twelve months period. The median release was 8.12, the second newest!

(There is a bug here: the “8.10” version got absorbed into “8.1” because of how Google forms plus Google sheet “helpfully” consider them the same number.)

Versions used in 2025

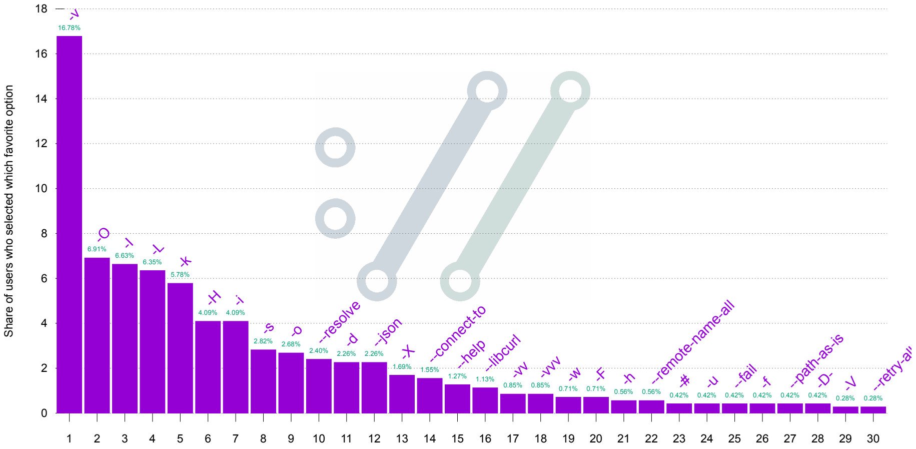


Favorite command line options

724 responses

This is of course a question mostly for fun and is hard to do anything scientific with. The free text nature of the answer form also adds to the problem of doing analysis on it. I did my test and have tried to accumulate mentions of options to the short version of them if they have one.

Top-30 favorite command line options 2025



At the end of this analysis you can also find many helpful answers to the question: Which curl command line option do you think needs improvement and how?

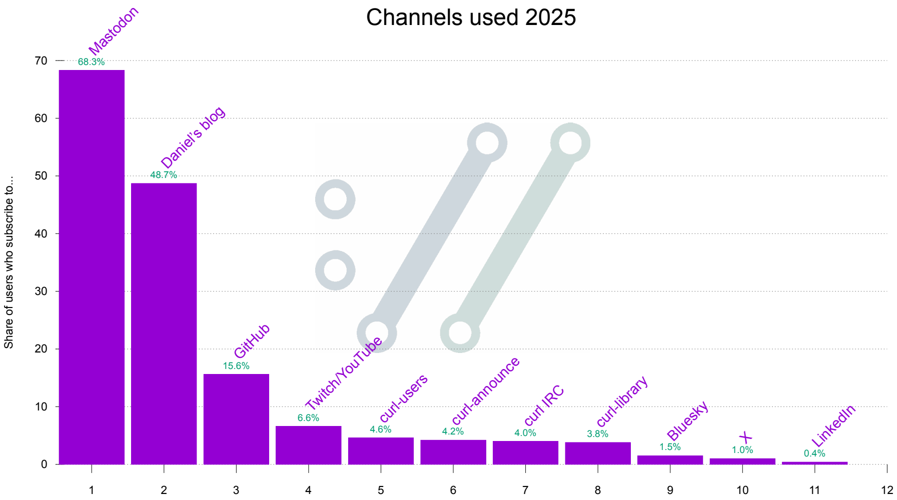
Used channels

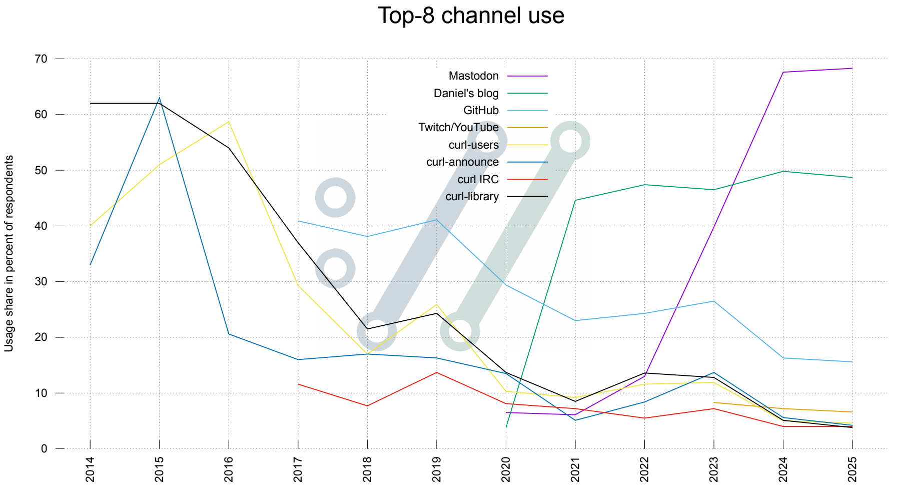
877 responses

Where do users learn and talk about curl? This question tends to be more where do the people who found the curl user survey and responded to it, also find information about curl related things?

This is a question with answers that have changed a lot of the years and they keep changing. Mastodon and Daniel's blog have climbed up far as the primary curl information channels.

GitHub as well as the mailing lists have fallen in popularity from their hay days.


Channels used 2025

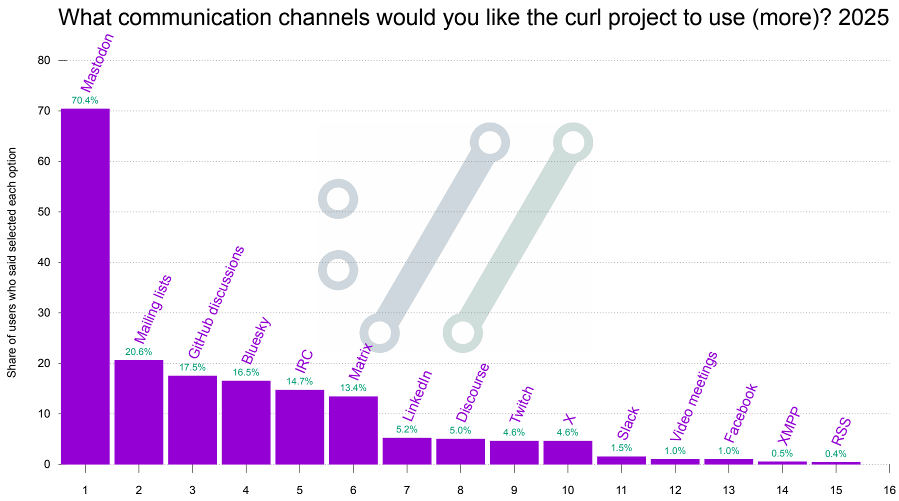

Top-8 channel use

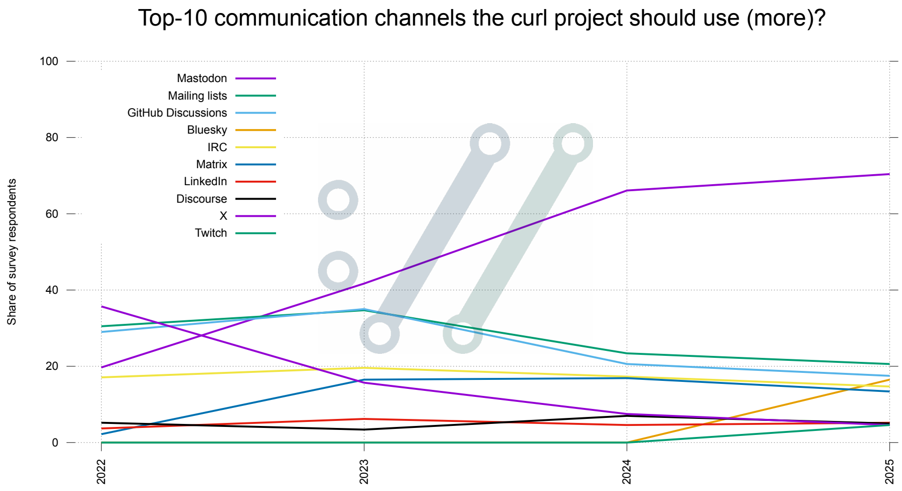
Channels to use

733 responses

Related is where people would like the curl project to be more visible and which communication channels to use more going forward.

Maybe the biggest discrepancy between these two questions is that 1.5% seem to be using Bluesky now, but 16.5% want us to use it more.





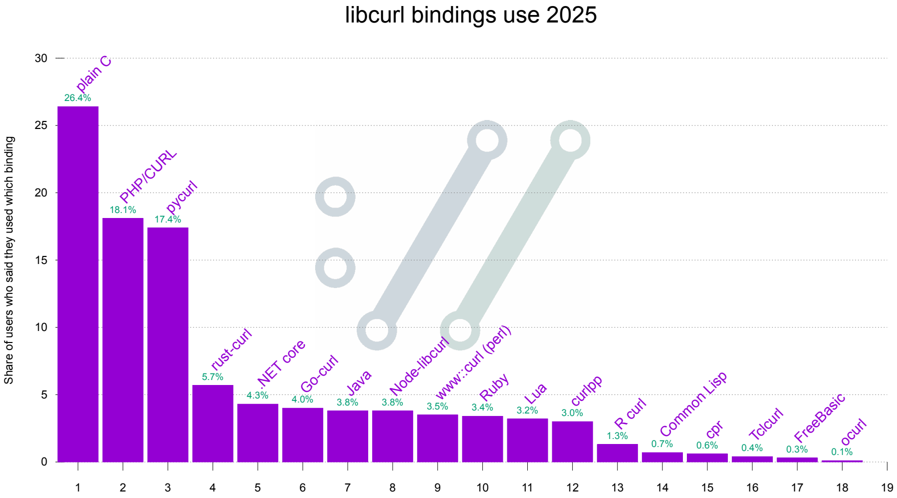
Accessing libcurl

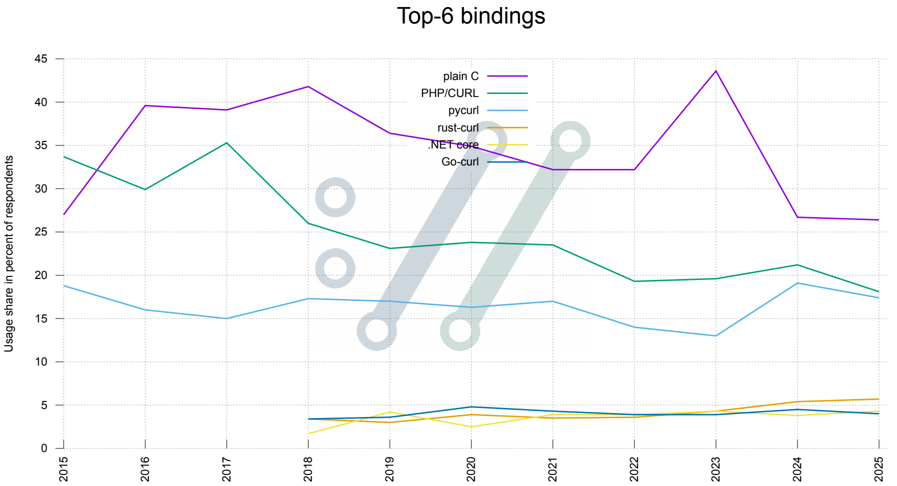
949 responses

libcurl is the network transfer engine of the command line tool and is commonly accessed by users via bindings. The bindings are what makes libcurl truly accessible to almost all developers everywhere. Which ones do people use?

85.7% of the respondents this year say they use curl, the command line tool, so presumably at least some of them only use libcurl (or neither).

The graphs below show libcurl binding usage among the respondents.


libcurl bindings use 2025

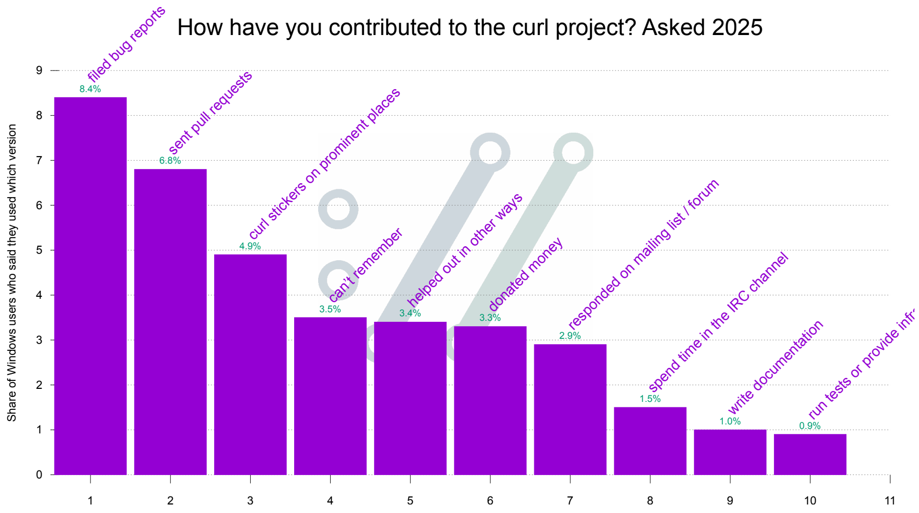

Top-6 bindings

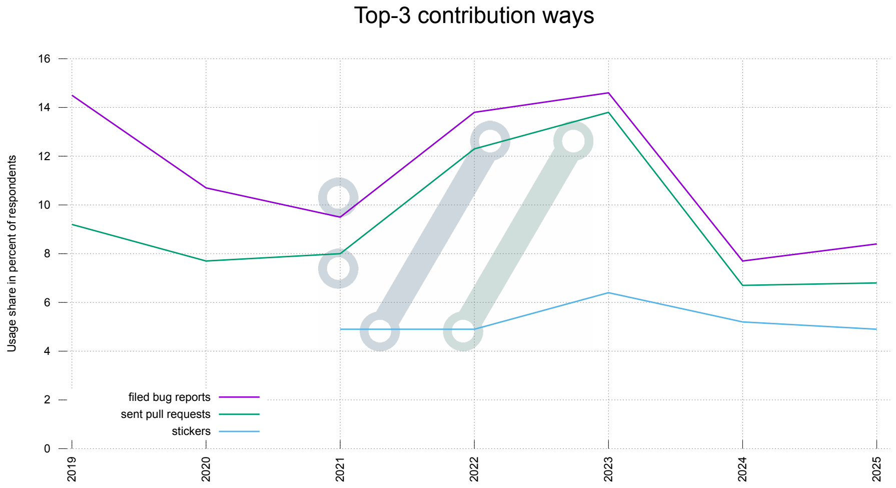
Contribution

989 responses

Users contribute to curl to a large degree. Maybe this question tells us more about the particular user population that answered the survey than what it says about curl users in general.

How did the respondents contribute to the project?





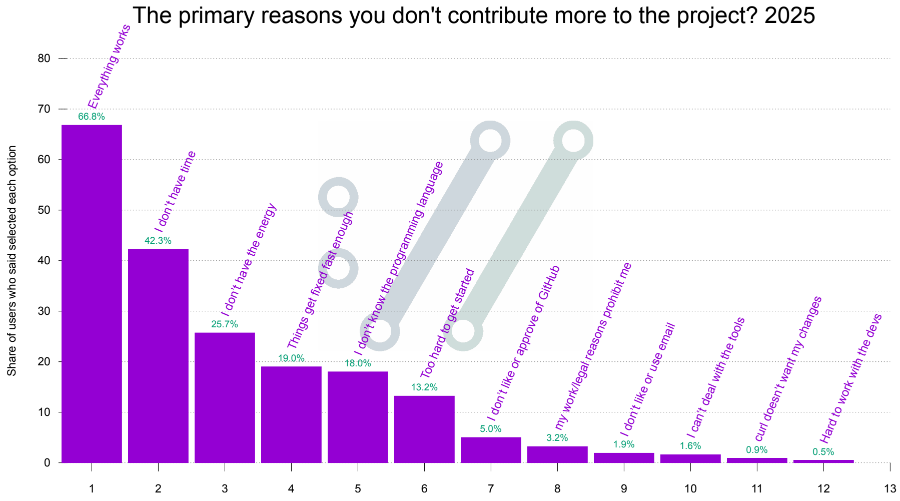
Why not contribute more?

1,072 responses

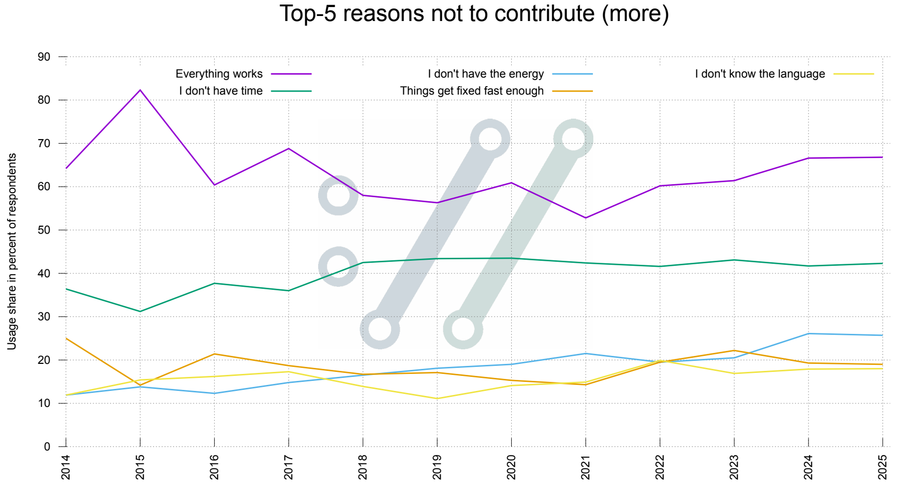
Why do the respondents not contribute (more) to the project we asked.

curl of course is in competition with every other Open Source project for contributors' time and attention, but also with everything else in life, like YouTube, games and TV shows.

By having an idea what the major obstacles are for more contributions, maybe there is something we can do?



The primary reasons you don't contribute more to the project? 2025

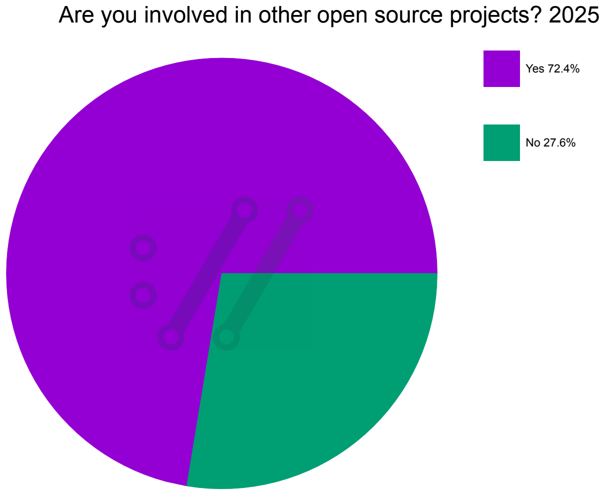


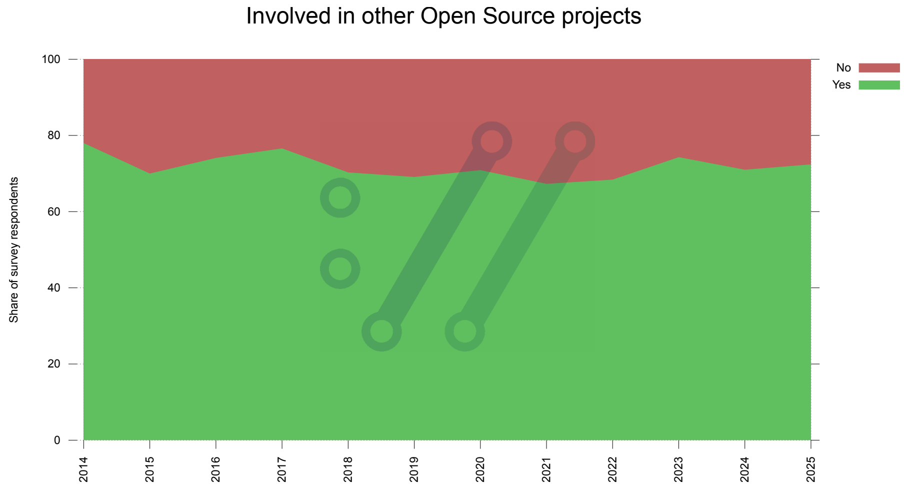
Top-5 reasons not to contribute (more)

Other Open Source

1,104 responses

The respondents are to a large degree involved in other Open Source projects than curl. Perhaps in addition to.





How good is the curl project

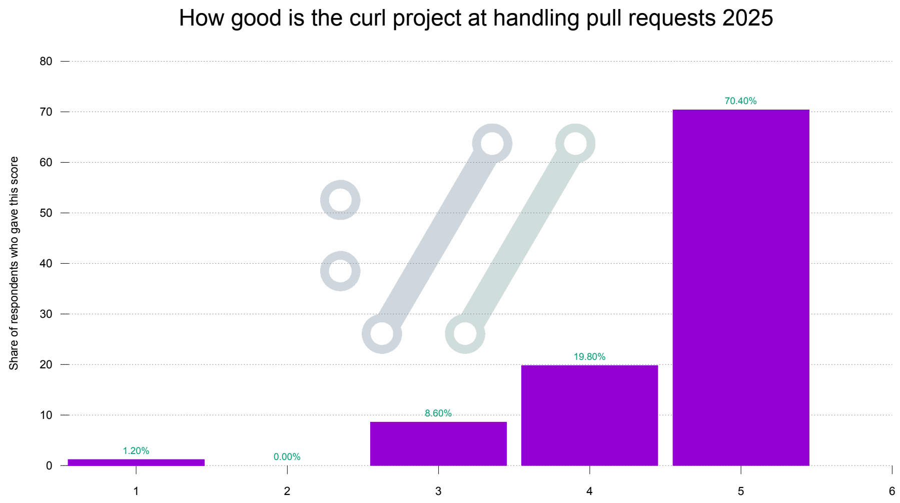
We asked the respondents to grade how good the curl project is to handle things in seven different areas. 1 is the worst, 5 is the best. To compensate for the fact that there are different amounts of answers to each section, we count selection by percent.

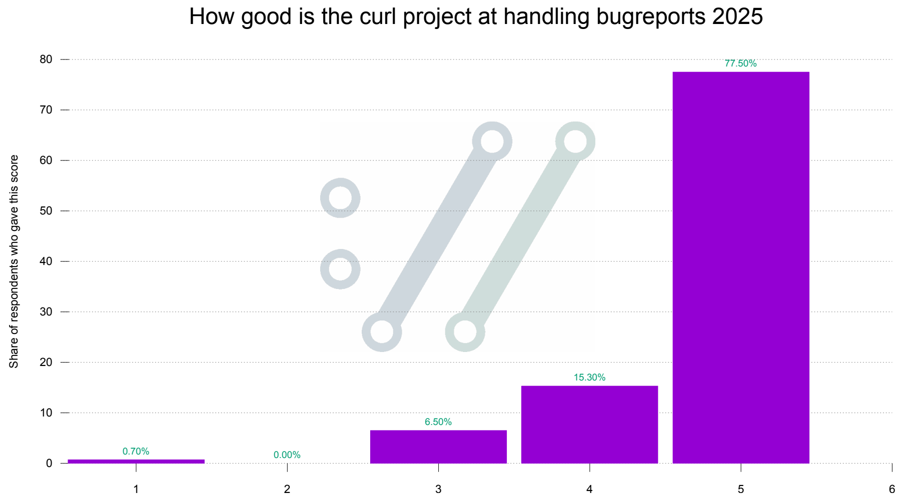
Participation:

section	number of responses
patches and pull-requests	243

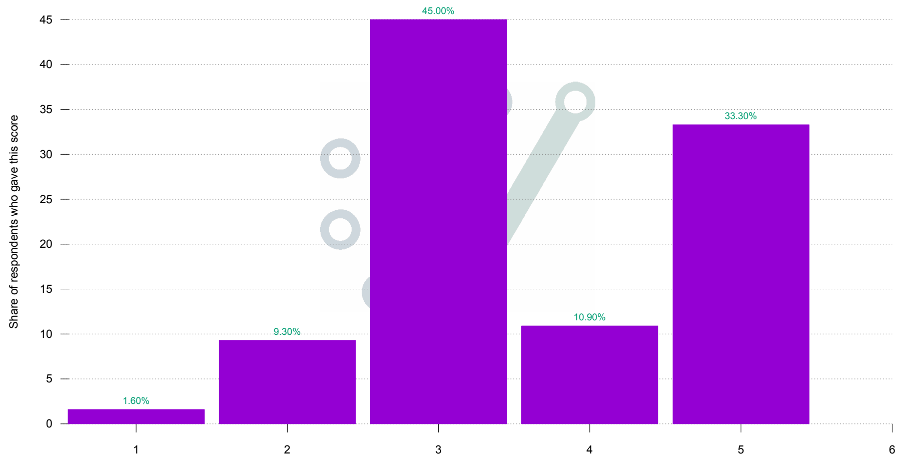
section	number of responses
bug reports	275
female contribs and minorities	129
attribution and giving credits	292
helping newcomers	198
information	362
security	412

Here is the overview of how the scoring was done for the 2025 answers

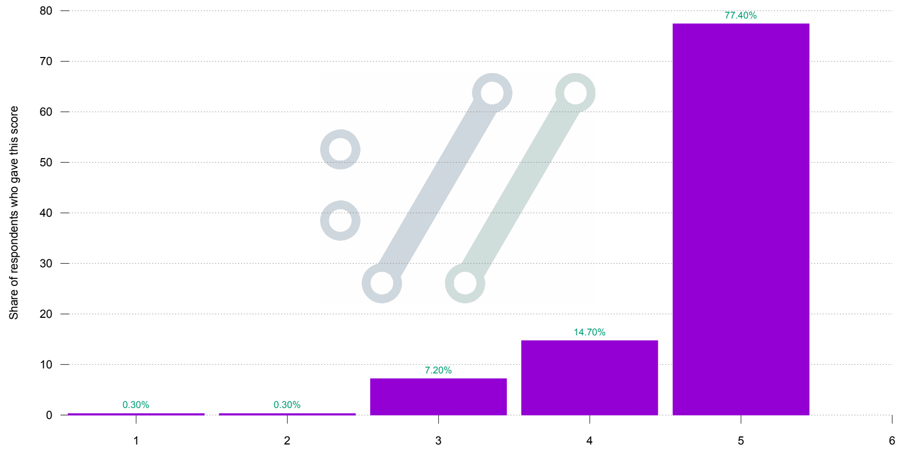




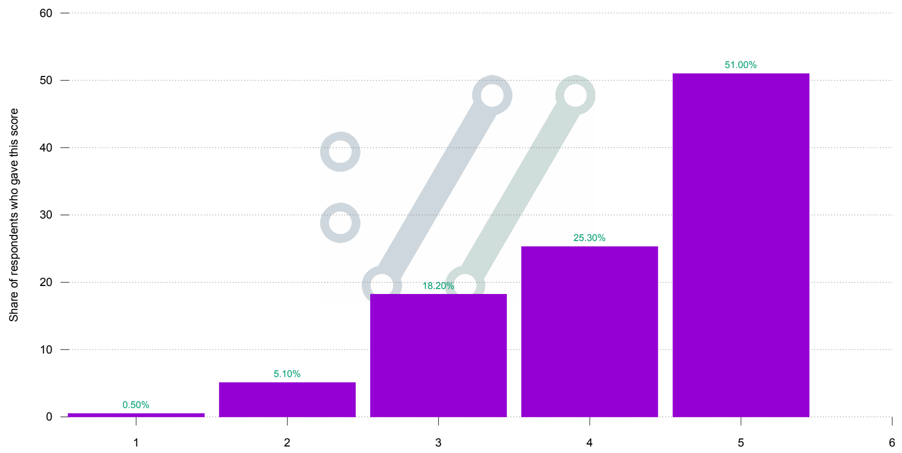
How good is the curl project at handling minorities 2025



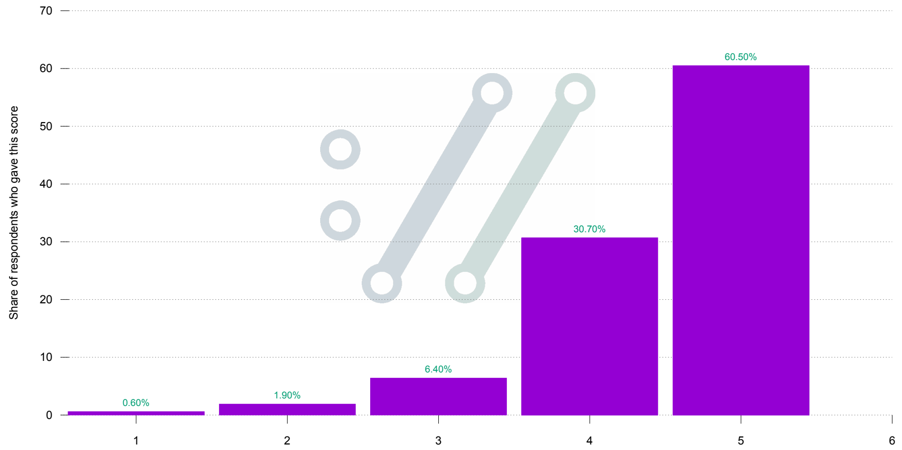
How good is the curl project at handling credits 2025

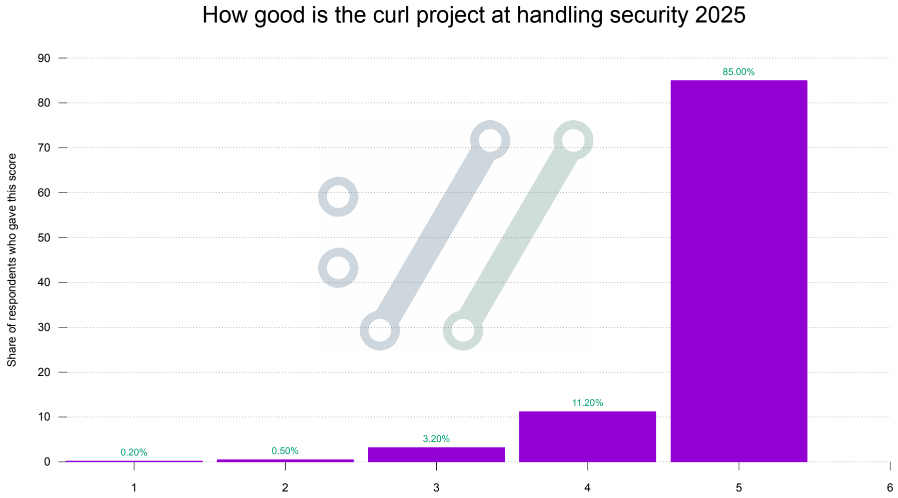


How good is the curl project at handling newcomers 2025

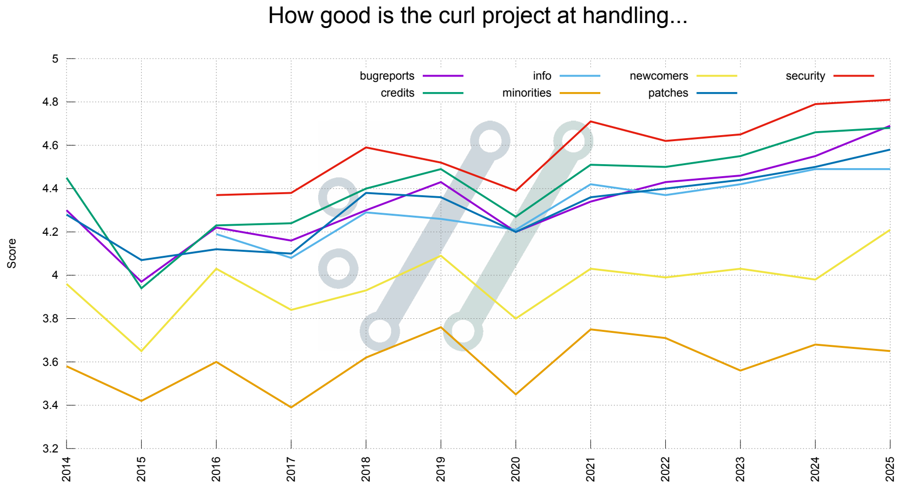


How good is the curl project at giving information 2025





Converting those into a single score per section, then comparing that single score with how people answered these questions in the past



The general pattern seems to remain: we are best at security. This year bug reporting surpassed giving credits, but both are still scored very high.

Our primary weak spot is "handling minorities", whatever that actually means. We are certainly bad at having a diverse representation among top contributors and project leadership.

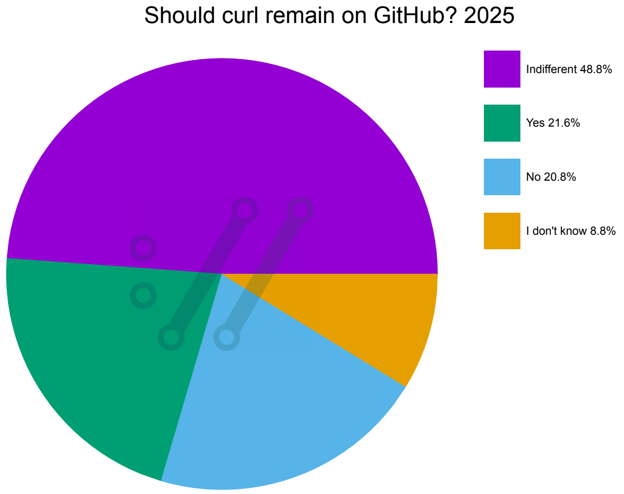
The "handling newcomers" is the second worst plot but at least it took a significant bump up this year, and with a score above 4.0 it cannot be *that* bad, can it?

Remain on GitHub

1,120 responses

This question was added to the survey because it has been brought up in the community once in a while recently so it was interesting to get a sense for what people in general think.

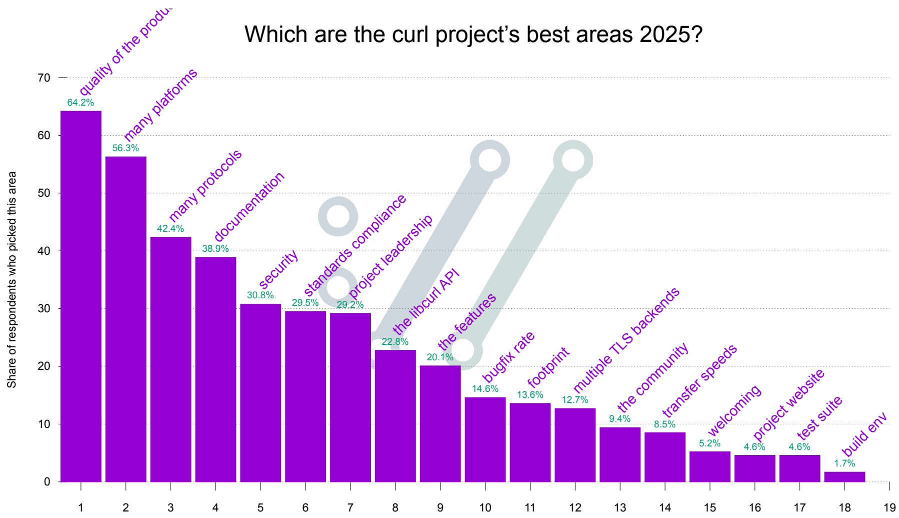
Clearly people don't have a lot of strong feelings for this. Only one in five said yes and roughly as many said no, and the rest were indifferent or did not know…

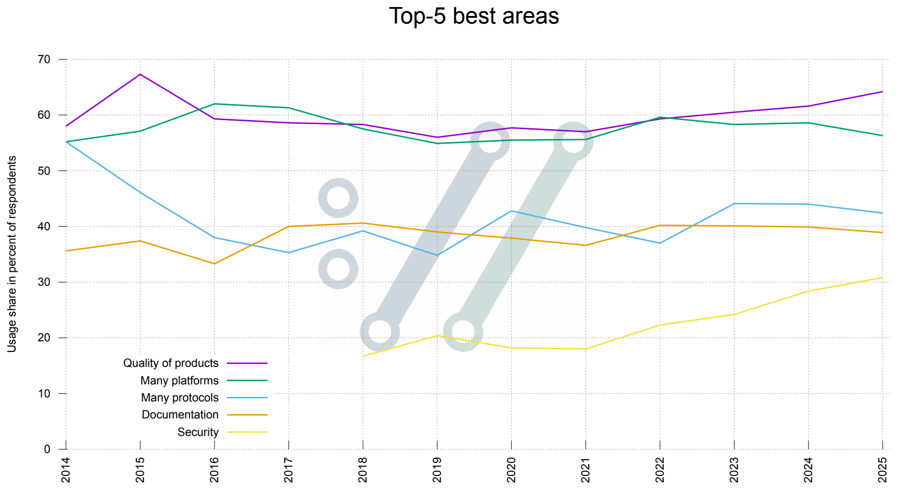


Best areas

938 responses

Asked to select up to four areas where curl is best, this is the distribution.



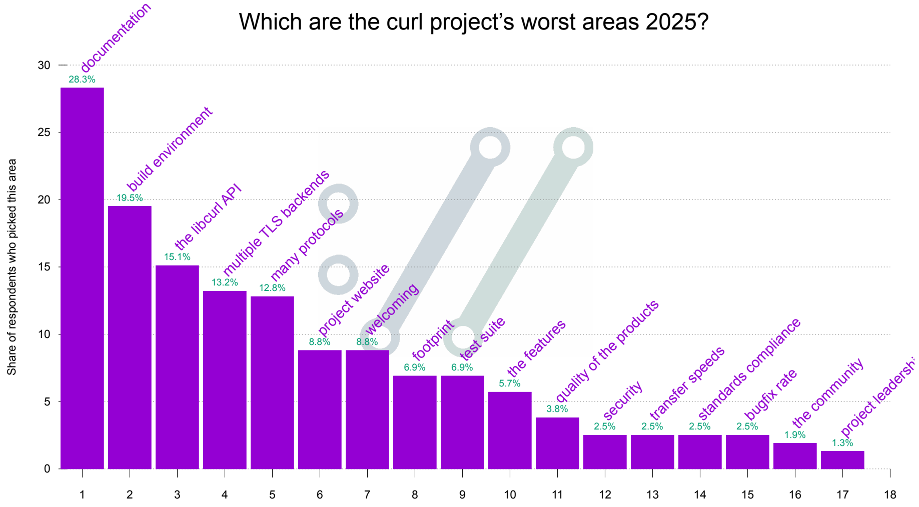


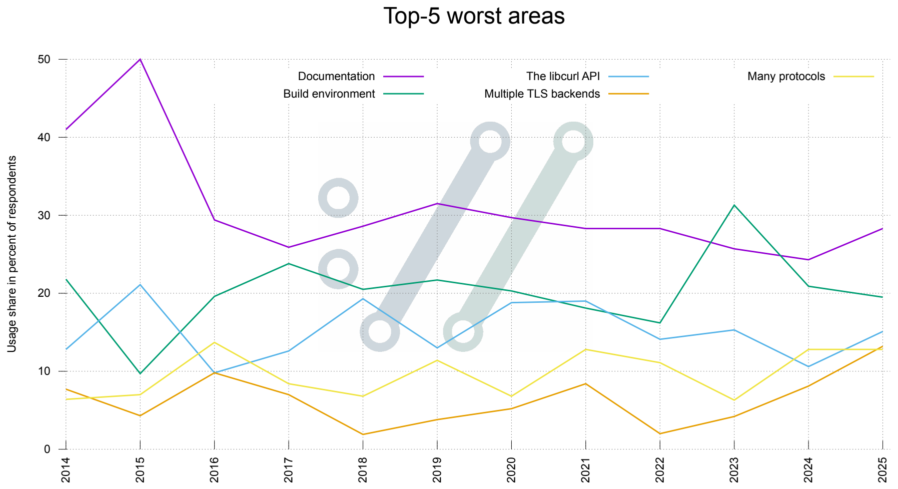
Worst areas

159 responses

Asked to select up to four areas where curl is worst, this is the distribution. The options are the exact same as in the "best areas" question above.

The idea behind this question is to help us figure out which areas perhaps need more attention than others. Where we should improve.





If you couldn’t use libcurl, what would be your preferred transfer library alternatives?

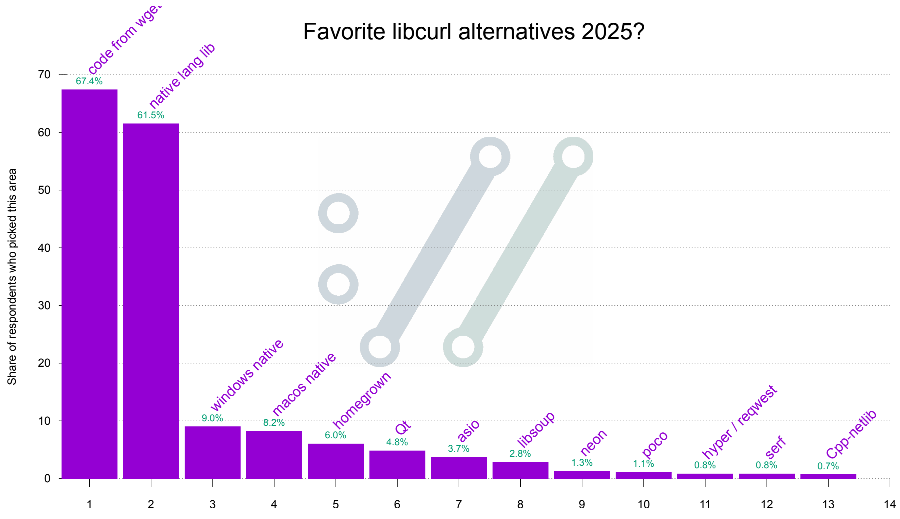
918 responses

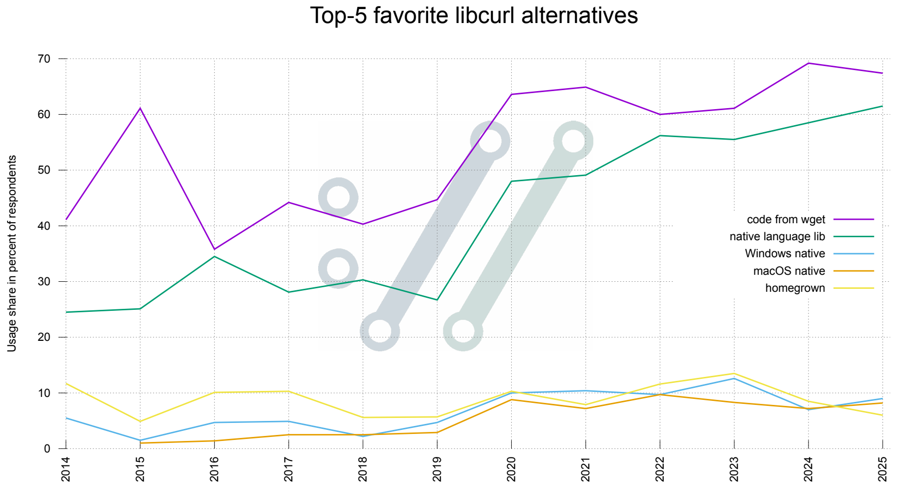
This question is meant to give us a picture of which libraries and options that are the biggest “competitors” always keep us puzzled because so many respondents actually say they would use code from wget , which in my book is close to “homegrown” but the latter only gets 6.0% of the responses.

This year we got a few write-ins for hyper/reqwest that made me add that to the graph. It clearly should have been a provided option from the start.

The longer name for “native language lib” is “native lib in Perl, Python, Java, Go, Rust, JavaScript etc”

The wget and native lib options have clearly increased in popularity as libcurl alternatives in recent years.





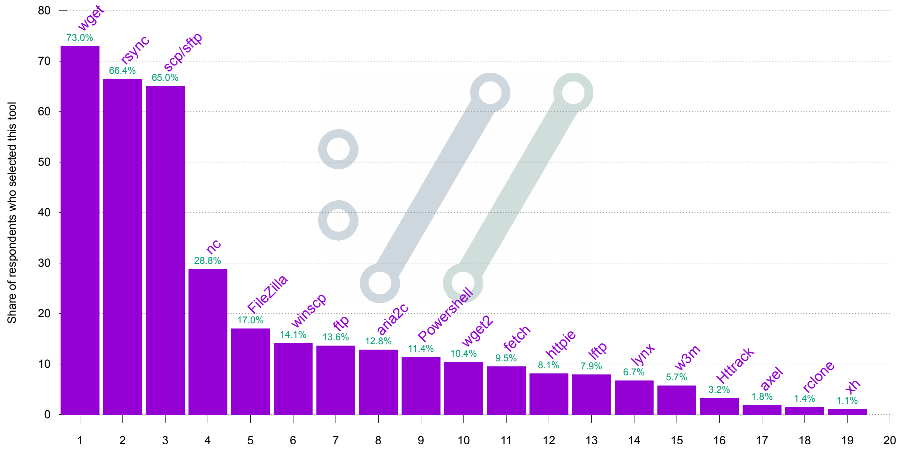
Which other download utilities do you normally use?

1,039 responses

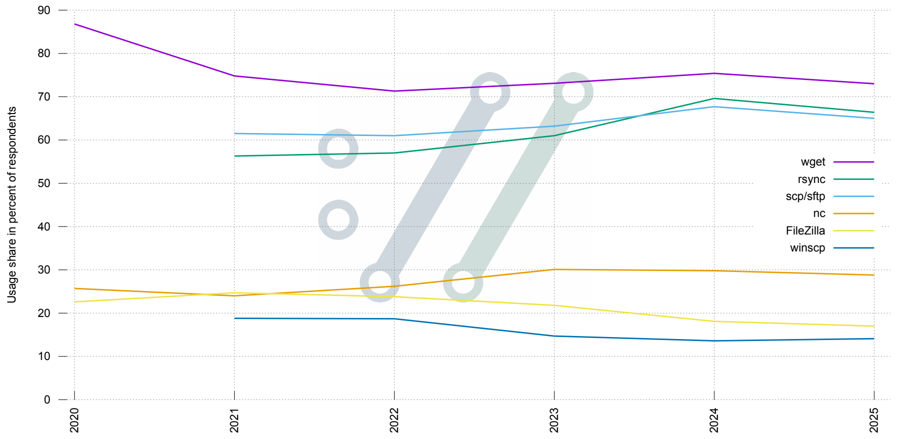
These other tools are partly competing alternatives to curl, but in many cases they of course instead offer tangental and additional functionality for the toolbox. Three out of four curl users also use wget.

This year xh and rclone were added as write-ins so often I decided to add them to the graph.

Which other download utilities do you normally use? 2025



Top-6 favorite download utilities

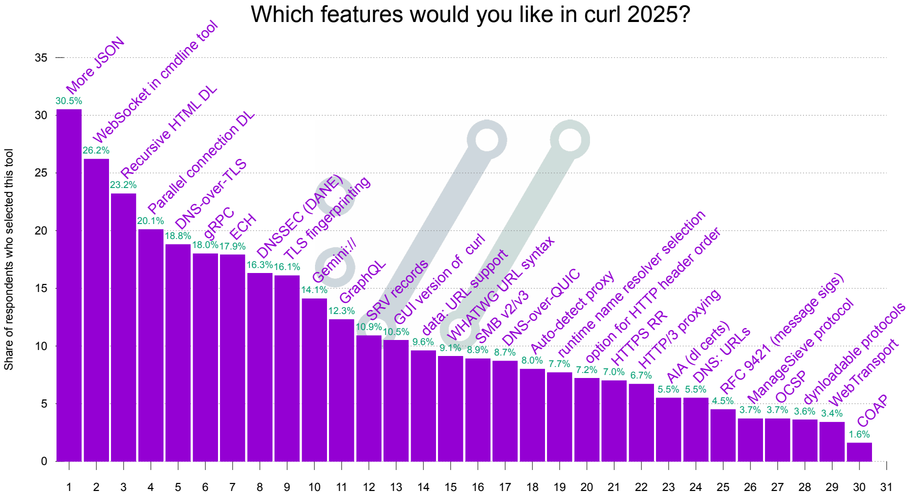


Which of these features would you like to see curl support?

861 responses

This is a question we repeat every year, but the answer alternatives tend to vary so much that a regular “development” graph for them makes less sense here.

The answers here help guide us in the years to come what to focus on and what maybe is less important.

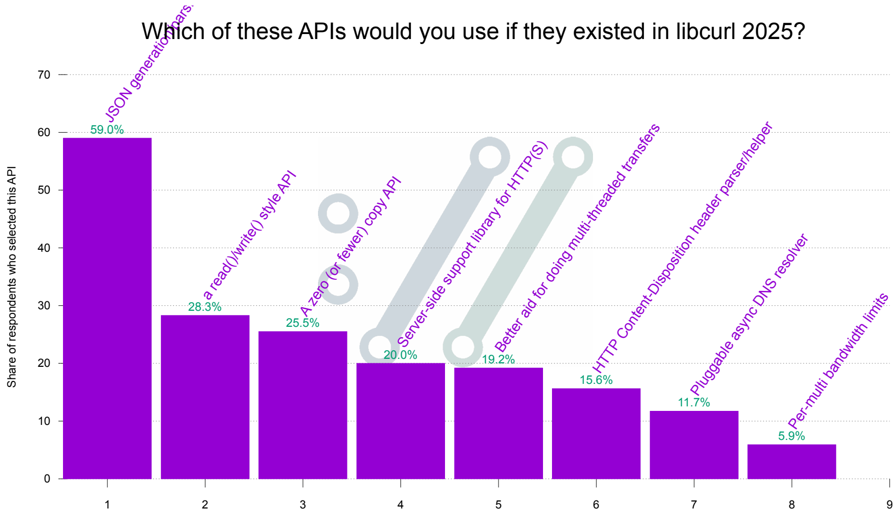


Which of these APIs would you use if they existed in libcurl?

505 responses

Maybe this list can serve as an indicator of what libcurl users want from libcurl. Although it is a little vague for some of them exactly what the options mean, and I think some of them are not likely to ever actually get implemented…

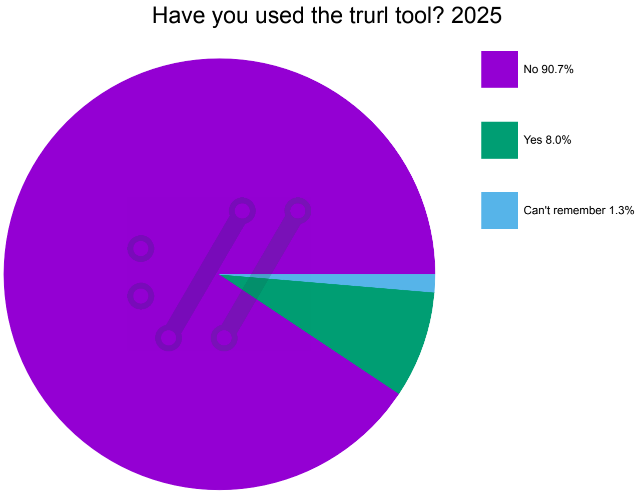
As with all questions, it is of course easy to select it in a survey without much extra consideration.



Have you used the trurl tool?

1,049 responses

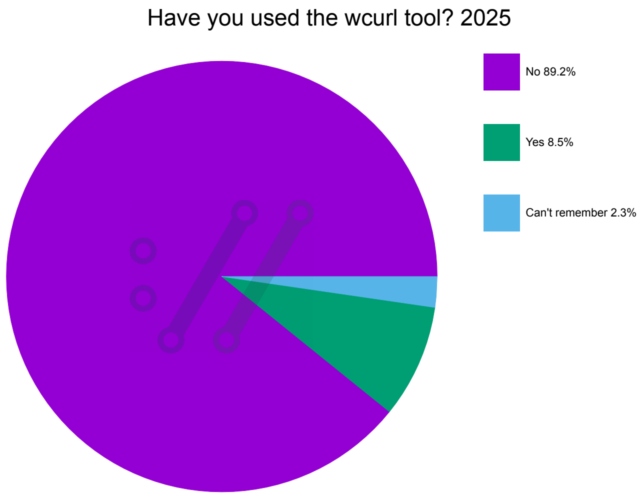
More of an information than a genuine question perhaps, but we asked last year as well and then 6.6% said yes. This year it moved up a tiny bit.



Have you used the wcurl tool?

1,046 responses

wcurl recently was made to get released as part of the curl tarball so it should now be more accessible and easy to use for many users. This was the premiere for this question and it shows that wcurl is not yet too widely used.



By your estimate, how many installations would you say curl and libcurl run in?

778 responses

I admit that this question was a little tongue-in-cheek, and since it was a free text field it is not really possible to do much with the responses here. If we keep this question in the survey next year, I think we should only allow numbers or maybe offer a wide range of pre-selected alternatives.

We got answers ranging from 1 (yeah, always the jokers) to 100 billion and eternity.

Which question would you like to see in this survey next year?

73 responses

- Thoughts about adopting a mascot!
- AI sentiment (assuming the bubble hasn't burst yet)
- something something… scripting usage (in shell scripts and stuff)
- Are you answering this questionnaire on behalf of a corporation or on behalf of yourself?

- Have you ever build open-source apps on top of curl? And how many
- Which program(s)/library(ies) you formerly used on a regular basis have you replaced with (lib)curl as first go-to?
- Question about age/gender
- What do you use curl for: large transfers, interactive debugging, automation, etc
- Probably some more differentiation between developers and sysadmins
- Are you or your company funding the project? Yes/No/How
- What is the oldest version of curl you used this year ? (Useful to know in professional environment where some really old software aren't updated that lacks some recent curl features and security)
- what alternative code hosting would you like to use (i.e gitlab, codeberg, self hosted, etc)
- Should we sell stickers?
- Do you use the libcurl URL API?
- Maybe some questions about what applications we use curl for?
- Ask if there were any obstacles in using curl. "What is stopping you from using curl?"
- What do you find most confusing about libcurl's API?
- What graphs do you think should be added to the dashboard?
- more protocol related questions
- What do you think could be dropped from curl?
- "Do you think curl is feature complete?" and "Do you use the musl, glibc, or busybox version of the curl utility?"
- Tell us about a time curl saved your day

If you miss support for something, tell us what!

58 responses

- Checking a download matches some predefined sha256sum, removing the file if it doesn't, and exiting with an error exit.
- Some way to signalize that I'd like to cache the curl-ed responses for a few minutes, so that I don't spam servers by frantically changing oneliners
- Better documentation on how to start using libcurl on currently-unsupported platforms (particularly ~bare-metal)

- Instead of directly supporting anti-fingerprinting, the case may be helped by supporting options to tweak low-level protocol / TLS-stack (where possible) behavior

- MQTT support in the curl CLI is rather bad. Binaries bundled with mosquitto are still way better. Things like error messages or the output format for messages in subscribed topics. I think a good benchmark of success would be MQTT tutorials and blog post not mentioning installing the mosquitto server on dev machines just to have access to these tools, but opt for curl instead.

- Comparing libcurl to platform native HTTP clients like NSURLSession on macOS, I want libcurl to have a easy callback to get notified at an exact point when all the redirection and response headers are ready, and we are about to receive response payload. I might have missed something, but currently I could not find a way but to manually predict based on headers received.

- Another thing that I hope libcurl to change is how it handles request methods and post fields settings. Currently one changed can affect another, which is a bit tricky as I need to handle combinations of methods and whether a request payload is set.

- NSS TLS backend for easier firefox/TB TLS fingerprinting countermeasures

- Free, separate opensource postman like Application -

- In some cases my clients do a blackbox/mitm with a trusted provider cloudflare (was the latest one). I would like to be able to have a warning/error if the certificate doesn't match the dns caa record.

- curl daemon mode

- A new –data-json command-line option, equivalent to –header 'Content-Type: application/json' and –data.

- More insights into how manpages and other documentation gets made (I'm too young for these but see their merits).

- Basic tls:// support (mentionned above). My alternative are openssl commands, which is not that bad

- I would like to better understand all the revocation options I have with curl.

- curl's websocket support is neat for debugging, but it would be neat if I could use it as a generic tty-like thing. same with telnet. note: i might have misunderstood the docs on this.

- configurable tcp mss clamping, because PMTUD isn't reliable and forcing a MSS only to a specifc target via the OS require high privileges

- anything related to multicast, less location-based downloading, anything useful for decentral (mesh) networks
- I like the idea of a lightweight GUI or some other way to have guidance around how to build you CLI commands without needing to read a few web pages each time I use it.
- LDAP add and modify features
- Color printing
- Ability to specify keytab or credential cache for –negotiate from the command line without having to use environment variables. It would be nice to have, but probably not useful for many people.
- Content-Type shorthand? I recall discussion of this in the past though.
- Choose TLS backend per request
- Meson build; I can't use CMake due to circular dependencies in my distribution and meson build support would enable meson projects to trivially use cURL as a subproject
- Ability to share cookies and other context with a browser
- rsync
- Remove curl_global_init(); handle it automatically in curl_easy_init() using a thread safe ref count
- Download over multiple TCP/UDP connections, like aria2c does, because I want to test how HTTP/3 works over multiple threads
- maybe bittorrent support would be cool like aria2 has, if that's not too out of scope
- better handling for resuming failed connections and when on e.g. cellular with high throughput but also high packet loss (TCP windowing basically makes most things impracticable)
- Simplified content negotiation similar to –json, but for XML.

Which curl command line option do you think needs improvement and how?

147 responses

This is a filtered and curated list of the free text answers we received.

- Can't think of any
- more –ca-native on Windows/macOS

- save SHA-256 with `--xattr`
- –connect-to and –resolve have mildly confusing syntax with all the colons, especially if IPv6 addresses are involved.
- –http2-prior-knowledge silently does nothing if https:// is used instead of http:// which makes it seem like it doesn't work if one just skims the man page.
- The –krb man entry could mention the existence of –negotiate.
- –oauth2-bearer includes "oauth2," but the Bearer authorization method is (per RFC 6750) also usable for non-OAuth tokens; an alias –bearer or –bearer-token might help.
- A shorthand for -H "Accept:" would be handy.
- maybe a command line flag option could be added to enable tor functionality (I don't use oniux yet, but I read daniel's blog post on the issue)
- -O behaves different from other programs (others take the filename to write to, curl explicitly uses it to not write to a specific filename separate from the provided URL)
- inline json
- Upload progress and multi upload progress
- curl -f file=@
- Difference between –data-binary and –upload-file is not clear.
- –cacert / –cerrt to make it easier to work with sites which use their own certificates (aka self-signed)
- The –data parameters, in particular –data-raw. Their specific functionality is quite confusing and I end up using –data-binary most of the time. It would be useful to have a short option for it.
- –method
- `-o /dev/null` i use during debugging a lot and I wish there were a shorter form
- a short option for ignoring any proxy settings and simply try directly.
- –hsts Should have better defaults on distros (preload list, automatically use a file)
- -o could use an alt-option that doesn't redirect -i output (to e.g. fetch a file to disk while printing the headers to stdout)
- I wish a feature that can auto change output to file or stdout by file type(e.g. curl example.com/aTextFile will output to stdout, and example.com/aPicture will will write to ./aPicture) n/a

- -L should be set by default IMHO
- It would be cool if the HTTP user-agent flag could just have a couple of common/recent strings built-in. So you could tell it "chrome" and it would just drop in a valid Chrome user-agent string.
- –data-urlencode would probably need a short alias.
- -i with -v prints everything twice
- `--help`: it'd be a bit more convenient if it had a few more of the most frequently used flags rather than needing to remember to do `--help all` or `man curl`.
- –progress could use more information, prediction and things like that
- –max-redirs . It should implement the -L option by default or have it's documentation update if it's already the case
- -H - some way to input multiple headers at once or with a shorter syntax would be welcome; currently repeating -H ".." takes up a lot of command-line space. (Also, process substitution with -H @<$(. . .) doesn't work (:")
- -u, –user, account for variable expansion or escape characters in username and password i.e. when used in a script file, or output error message when the following generalized condition is met: option is "-u, –user" AND no URL == true AND context of call is outside command line (I don't know if curl checks that, but - anyway): then, stderr return: (2) no URL specified ([0-9]+) username and password could not be read
- Websockets could be easier to use
- –retry: don't truncate when resuming download (https://github.com/curl/curl/issues/1084)
- -v, make it pretty and colorful!
- -b/-c - would be nice with a combination
- all of the long options, support –option= meaning with the = symbol
- Curl as a testing websocket client would be useful (websocket ideas acknowledges it I guess)
- –resolve (just give it an address and resolve every host to it)
- -z with yet-nonexistent file gives confusing warning
- –limit-rate, average too broad and it might not apply to upstream?
- curl smtps . . . -mail . . . attachment handling
- automated website crawling and link re-writing to have websites entirely offline (think wget -k)
- –user-agent - Read User-Agent from a file or EnvVar

- –include by providing an to optionally print the headers to a different stream, so that responses can be piped but the headers are shown on stderr, for example
- have a separate option for TLS client intermediate certificate chain (now one has to ‘cat’ client certificate and the chain)
- “–trace-ascii /dev/stdout” maybe need a shortcut option to print the output in the terminal, also also this flag is not shown with “–help” but it’s incredibly useful that I think should have more visibility.
- -b/-c: it would be useful to have a “session” style flag that would store session data in temporary storage (e.g. /tmp/curl/session-name.cookiejar). Oftentimes when I need -b and -c, I use them together for some chained script, and it’s a little annoying to have to mktemp + cleanup (but cleanup conditionally so I can analyze what went wrong if it did) etc.
- No progress bar by default please
- –config could probably have more features that’d allow users to run test scenarios and/or scripts from within a single config file. I unfortunately don’t have a proper description here.
- it is good but if it’s download interfce would a little bit modern and minimal it would be great. may be as a choice of an aulternative option would be enough.
- s/–happy-eyeballs-timeout-ms/–happy-eyeballs-timeout-ns/
- –write-out or similar nwe option to file or environment variable, allow several occurrences
- –data variants always confuse me, and I need to check the manpage everytime I need to post data
- Easy Content type + accept json
- -d and -F. I had cases where error messages originating from the use of them were confusing
- The –netrc-optional flag is very useful, but I wish there was a corresponding –authinfo-optional flag which could check for an optional ~/.authinfo.gpg file and an optional GPG agent socket and use the two together if they both exist and are valid.
- -i -I and -v, showing both http headers and downloading the file / redirecting the output on the command line is sometimes not intuitive for me
- –remote-header-name should work with –continue-at, I need files to have the correct name and be able to resume failed downloads. I’m currently looking at alternatives to curl because it doesn’t do this. I find curl kind of hostile when I just want files to have the “correct” name (i.e the name they would have if I used a web browser or any normal tool)

- -k is counter intuitive, because the character is none of "ignore certificate".
- A mode for silent success and sane error output
- –verbose: If I'm debugging an http issue I'm not interested in tls debug messages and vice versa.
- –time-cond, make it an option to use the time of the local file
- -g should be default
- –write-out {body_on_fail}
- –resolve : ditch the port number because it's unnecessary
- maybe a variant of silent that buffers –verbose output and shows it on errors?
- A bit silly, but it somehow bothers me that I have to quote headers I want to pass. I'd love it if I could type something like `curl -H accept=application/json https://google.com`. But that's not really a big deal, of course.
- –parallel - sometimes the DL% shows a value that's more than 100
- –cookie would be nice if it could more easily just say "use this browser profile's cookies" and handle it all seamlessly, so I could automatically download pages that I can access from my browser.
- It'd be nice for curl to be able to construct JSON data from key/value pairs without external tools such as jo. jo is seldom installed on machines, it is not that well known, and it's not mentioned in the man page either.
- maybe -w I think what I really want is an option to accept 2xx,3xx http status codes and exit 0 and 4xx,5xx status codes exit non-zero
- Would be nice to have some means to craft the requests body easier (TBH, I don't have a fully worked out idea how this could be done. Maybe some kind of templating…)
- -s should not silence error messages
- It would be nice to have a shortcut for commonly used headers like -H 'Content-Type: application/json' and -H 'Accept: application/json' when making API requests.
- I would really like an –cookie-insecure that ignores the secure flag on cookies.
- it's a bit confusing that I need to past `-I -X GET` to receive only the headers in non-verbose mode
- –json with GET, similar –xml would be good
- -J should be the default -O behavior

- mTLS on Windows can be a mess, but only if you use the build by MS + schannel - so… Probably not an "official" curl problem?
- –resolve The format is quite confusing
- –insecure, I wanted to take a parameter that will randomly fail a request some percentage of the time as a reminder that the insecure flag was used and needs to be removed when the certificates work again in temporary situations

Anything else you think we should know?

277 responses

This open-ended field turns out to be used primarily for people to say thank you and expressions grattitude. Truly ego-boosting to read, but since they do not actually contribute to the analysis I have filtered out most of that from this curation.

- If I remember right, you banned AI contributions for security tickets… But I'd suggest copying Gentoo's AI Policy for code as well. Copyright washing isn't great and likely could cause legal troubles depending on how existing lawsuits shake out (though it's likely corporations will get what they want, so maybe it'll be a non-issue)
- I would like to use libcurl instead of my home-grown URL dispatchers but haven't touched the relevant projects deeply enough to justify the work to change over yet (i.e. I have already been sold on libcurl).
- Things have improved tremendously since the last survey's period, and the very rough patch of continuous regressions around the time the version turned to 8. Last time, I was losing hope; now, I'm back to no longer fearing updating to a new stable release on day one. Whatever might have changed, it's been noticed, and much appreciated! Thank you! Even if it was down to a run of terrible bad luck back then, a return to normalcy is huge.
- The curl CLI could have better support for shell scripts. For example, an option like `--dump-info file.json` to write a JSON with all the info (headers, the fields from –write-out, etc) from the response, in a way that it can be easy to parse with the jq (or similar) tool.
- Love the work yall do! Dont feature creep yourselves into an entirely different scope just because you want growth. Maybe parsing graphql/Json belongs to tools like jq.
- I personally enjoy working with libcurl as the general functionality is intuitive, and the documentation is saturated with overall great examples.

- I would wish to see more template focused examples, like basic initalization templates to make individual components easier to grab, which would especially help during the occasional forgetfulness of needing to find the "CURLOPT_URL" or "CURLOPT_WRITEFUNCTION" from time to time, after being off from coding with libcurl for a minute.
- smaller and leaner is best, that is one main reason to prefer cURL over wget
- cURL is awesome. I have been using it since last century, longer than any other piece of software. After the feer has passed, only cURL will remain.
- curl does one thing and it does it good. don't feel like you need to add more features because you have time. it's a go-to tool because it's bug free and reliable.
- I use curl mostly for interactive debugging of / experimenting with services that accept http requests. So being able to see and control protocol version/headers/payload is more imoortant to me than transfer speed.
- Way too many and complicated questions. We are developing extensive systems, where Curl is an extremely small part. We have no ability to familiarize ourselves with all the functions and features that exist in the curl concept. Windows 11 and AI work perfectly to create curl code for Visual C++, and curl commands that can be invoked in Visual C++ programs.
- Keep on the great work! You mentioned stickers: If only I had some – maybe some merch online shop would be an option to collect (more) money from willing end users. At least I would buy some given reasonable shipping rates to Europe.
- curl is a flagshop FOSS project. I wish I could write that good code and documentation. Daniel and all the other contributors: Y'all are doing a great job.
- Idea for a new option, –cheat-sheet. When curl –cheat-sheet is passed text from the curl-cheat-sheet repo is output to the terminal. I'll probably work on this when I get time in the future, but if someone beats me to it, so be it.
- https://everything.curl.dev/ kinda sucks for searching for individual commandline switches. there are certain sections which don't list everything that's related, i've noticed myself doing `curl --help all 2>&1 | grep -C3   (...)` more often than i'd like to admit
- splitting off trurl basically kills it, as one heavy use is in simple scripts, but that wont happen if it's not available by default…
- curl is more than the hobby of some guy that definitely has business providing service to more than a billion people, you're awesome!

- I really appreciate the way Daniel actively works to bring people into the development process behind curl. I struggle to think of another open source project that makes a similar effort to keep people informed of what's going on in a manner that is similarly welcoming and easy to understand.
- I embed libcurl in our work-related projects. It works like charm using CMake and Conan! Thanks for making curl great, it's my go-to debugging tool for all sorts of network debugging, right after ping and before telnet. It's simply the best.
- wcurl is such a cool addition to the suite
- I would happily buy curl swag (stickers, T-shirts etc) if available.
- curl is a difficult project to criticize - i've never once needed to do something with it that it couldn't. the only time i came close was when it was out of date and i just had to download a new binary, which really is not a big deal and the fault of os package repos anyway