curl user survey analysis 2023

“Words can't express the gratitude many people have, on how beautiful the impact of curl is to the world. It's just awesome.”

summary and analysis by Daniel Stenberg

Jun 17, 2023

About curl

Curl is an established and mature open source project that produces the curl tool and the libcurl library. While it is a small project, with just a few maintainers, its products run in several billions of Internet connected devices, applications, tools, games and services. curl is no doubt one of the world's most widely used software components. This year, curl celebrates its 25th birthday.

I believe a key to remaining relevant and to maintain a relevant place in people's toolboxes, curl as a project must keep up. Keep up with what users want, with how internet transfers are done, with Internet standards and with protocol development.

Survey Background

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 25 to and including June 7**. This was the 10th annual survey as the first one ran in 2014 and has run every year since.

The survey was announced on the curl-users and curl-library mailing lists (with two reminders), numerous times on Daniel's twitter feed (@bagder), on 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.

Survey 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.

Survey 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 last year. It might imply that the responses are indicative for a wider population.

Hosted by Google

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 not found a compelling and competitive alternative provider for the survey.

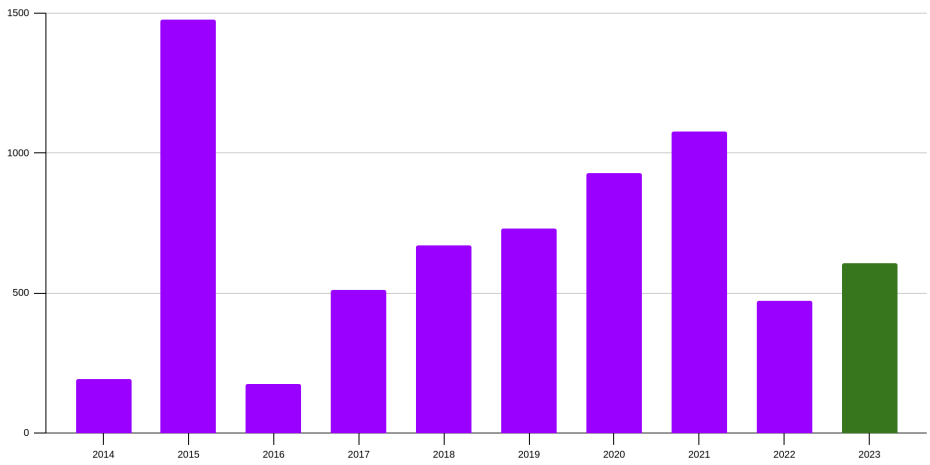
Five quick things to take away

If you are in too much of a hurry to read it all, here are five key facts this year's survey revealed:

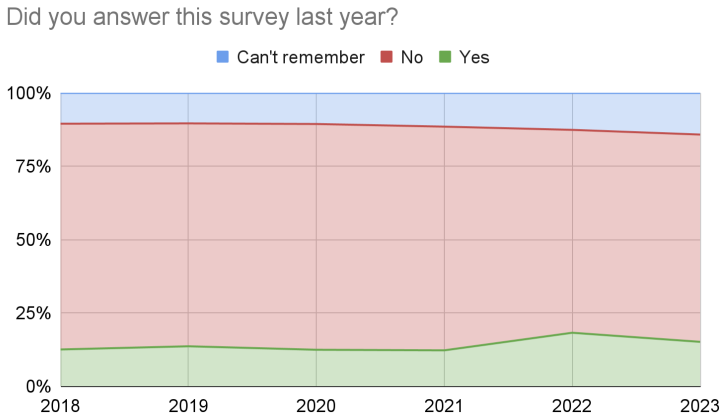
1. curl users leave Twitter and join Mastodon in quite notable amounts.
2. Windows 11 is growing quickly as a platform curl users are on
3. HTTP/3 is used by a quarter of all curl users
4. WebSocket reached the top-10 of most used protocols before its firth birthday
5. The positive comments in section 21 are heart-warming

1. Responses

Last year was a trend break as much fewer people responded to the survey than in the past, but this year we seem to have improved a little again. Up 28% from 473 to 606. Far away from the top years but we are grateful for the feedback we get. It is not easy to understand nor affect these fluctuations.

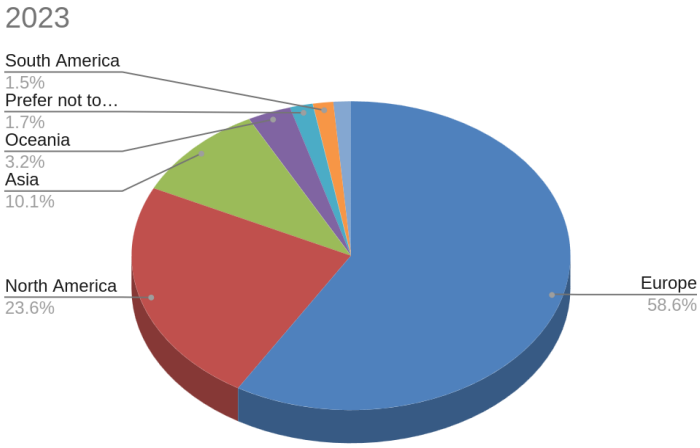


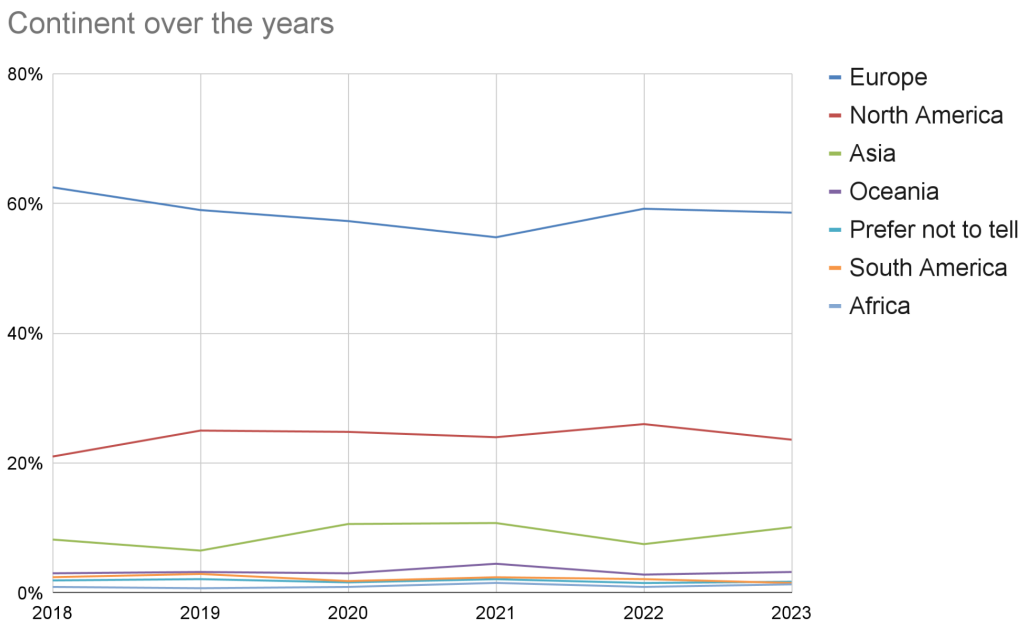
Only 15.1% (the median rate from the last 6 years is at 13.05%) say they answered the survey last year. A rate that is remarkably stable over the years, even as the participation numbers have gone up and down a lot.



2. Continents

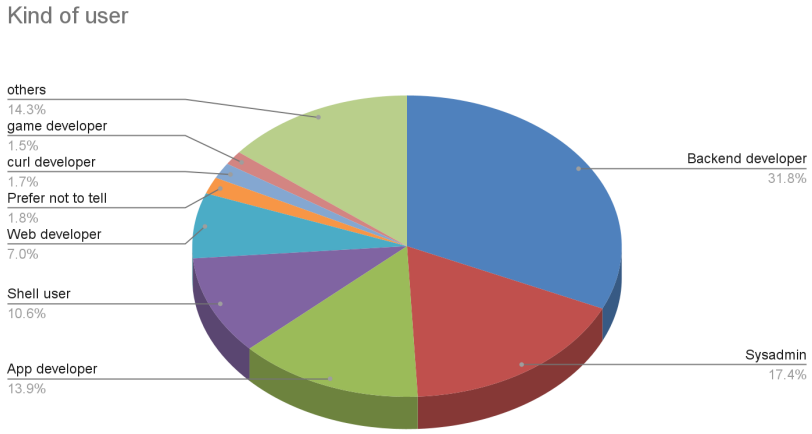
The users who answer the survey remain European to a large degree (58.6% in 2023) as this question appears for the 6th year. North America is 2nd as usual, at 23.6% this year.





3. Kind of users

The label most commonly applicable for the users who filled in the survey remains “backend developer”. This is the 5th year for this question and “backend developer” has never been below 30%. Labels are tricky, and that is presumably why so many select “other”.



4. Protocols

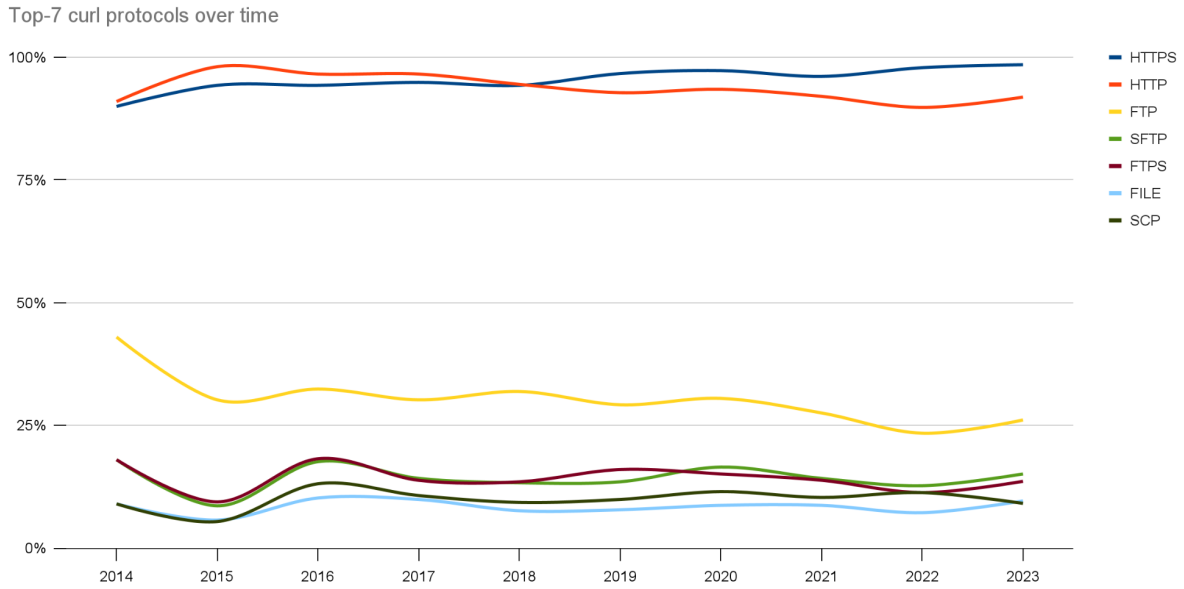
Asking users what protocol they have used with curl is of course a blunt tool. The people who respond to the survey are self-selected and we cannot tell if they answer factually. Maybe they confuse two protocols. Maybe they forget how long ago they actually used one of the protocols.

In this aspect we speak of the protocols as in different URL schemes curl supports. In spring of 2023, curl supports 28 different protocols where two of them (WS and WSS) are still labeled experimental, meaning that they need to be enabled explicitly at build time and therefore are not as accessible to all users as the others are.

This is the complete table 2023.

1	**HTTPS**	98.50%
2	**HTTP**	91.90%
3	**FTP**	26.10%
4	**SFTP**	15.10%
5	**FTPS**	13.60%
6	**SCP**	9.60%
7	**FILE**	9.10%
8	**IMAPS**	7.60%
9	**WSS**	7.50%
10	**SMTP**	7.30%
11	**WS**	7.10%
12	**SMTPS**	6.80%
13	**IMAP**	6.00%
14	**LDAP**	5.10%
15	**LDAPS**	5.00%
16	**TELNET**	4.80%
17	**MQTT**	4.50%
18	**TFTP**	3.80%
19	**SMB**	3.80%
20	**POP3S**	3.30%
21	**GOPHER**	3.20%
22	**POP3**	3.20%
23	**SMBS**	2.50%
24	**RTSP**	2.20%
25	**RTMP**	1.80%
26	**RTMPS**	1.50%
27	**DICT**	1.50%
28	**GOPHERS**	1.50%

In general the positions among the protocols in the table are solid. The top-7 protocols remain the same. Long term FTP has been shrinking but it seems to have stabilized somewhat. The two newcomers in the 2023 survey (WSS and WS) entered the table at position 9 and 11, well above many of the protocols that have been supported for decades.



If we look at this year's protocol popularity compared to five years ago, only two protocols show a noticeable decrease: HTTP is down 2.6 percent points and FTP is down 5.8 percent points.

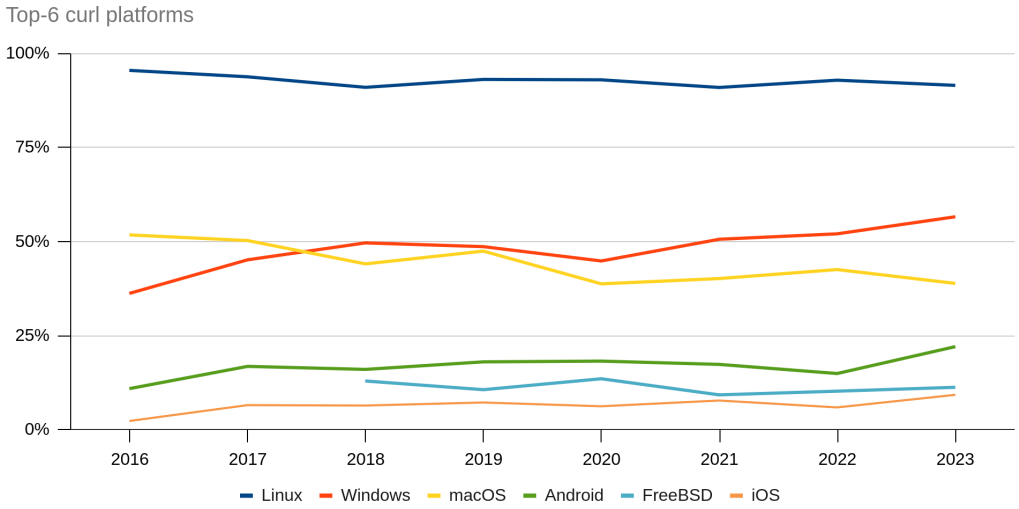
On average, each respondent selected **3.53** protocols, up from 3.2 last year. The median remained at **two**. Three users selected all 28 protocols.

4.95% use 10 or more protocols (which is up from 2.9% last year). 20.0% use 5 or more (up from 16.2% last year). **55.3%** of the entire population only used one or two protocols, and virtually all of them selected HTTPS and HTTP.

5. Platforms

We have records of curl running on **92** different operating systems over the years. Several of those were probably custom modified with changes we never got

upstreamed, and many of the users on niche systems most likely did not respond to this survey.



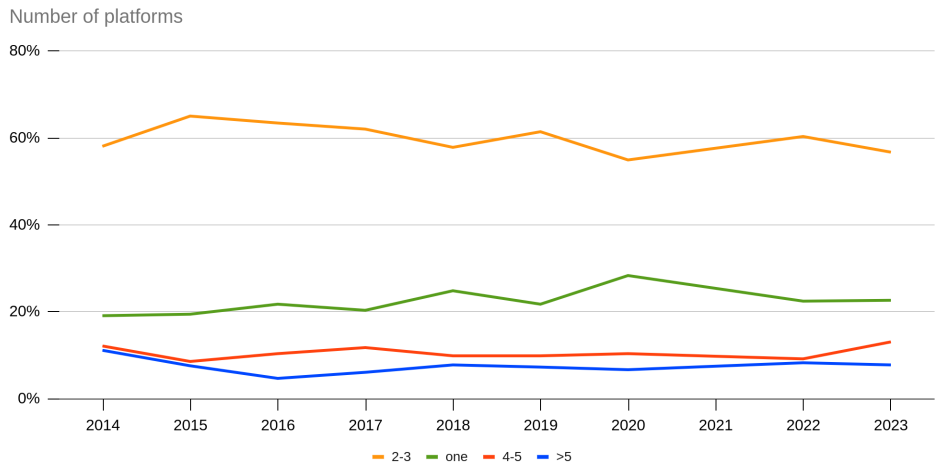
The top-6 platforms seem to remain the same over time. The Windows share might display a slow growth over time (at 56.6% it is up 4.6 points to its highest level so far) while the macOS line might be shrinking at the same time (down 3.6 points from last year to 38.9%). Linux remains the strong and undefeated leader. But also: Android's 22.1% is up 7.2 points from last year and is higher than ever, and so is iOS that is up 3.4 points to 9.3%).

On average, users selected 2.5 platforms each in the answer.

The complete platform distribution for 2023

Platform	2023
Linux	91.50%
Windows	56.60%
macOS	38.90%
Android	22.10%
FreeBSD	11.30%
iOS	9.30%
OpenBSD	6.00%
NetBSD	3.20%
Game console	3.00%
Solaris	1.80%
Another unix	1.70%
MS-DOS	1.50%
OpenIndiana	1.30%
RTOS	1.30%
AIX	1.20%
HPUX	0.80%
VMS	0.70%
AmigaOS	0.70%
IBM I	0.50%
IRIX	0.30%

Almost 60% of users remain using 2-3 platforms and the distribution of single and multi platform users remains astonishingly fixed over the years. Maybe a few have left the 2-3 share and are now instead in the 4-5 camp.

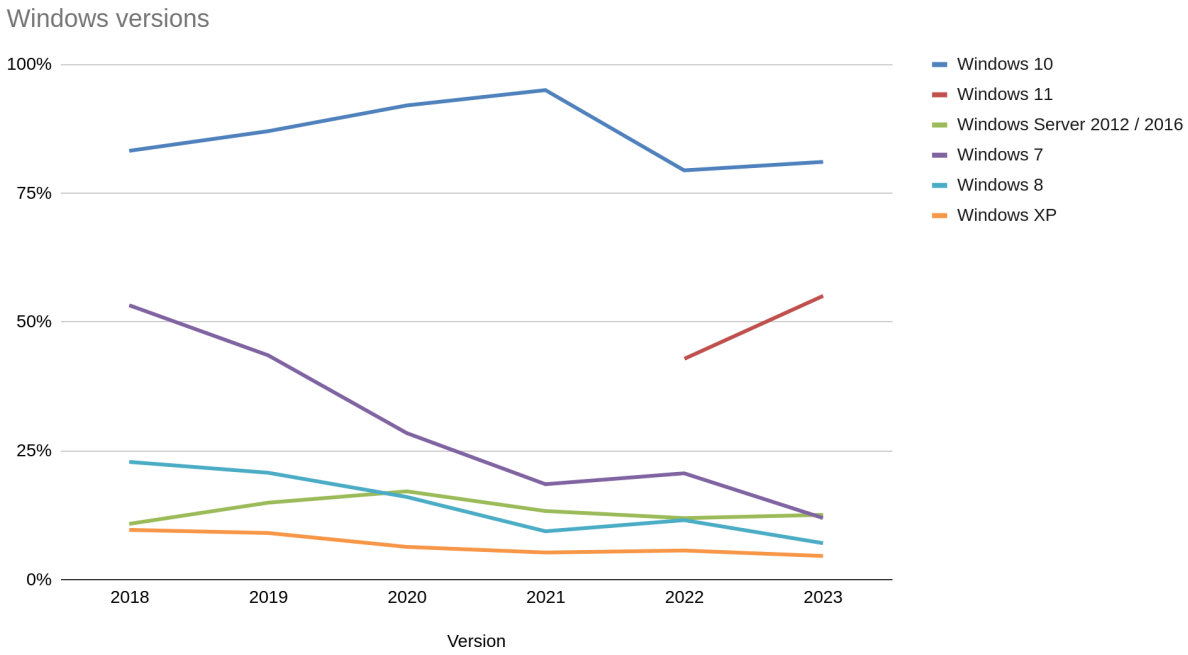


6. Windows versions

(The percentages shown for this question are the shares among the users who say they use curl on Windows.)

Windows 11 entered the survey last year and it continued its climb this year. Up 12.2 points to 55.1% Interestingly, the Windows 10 share still grew a little since 2022. Up 1.7 points to 81.1% and remains in the top.

Windows 7 and 8 keep plummeting, and even the seemingly never-dying Windows XP lost another point and is down to 4.6%.



The full Windows version distribution 2023. The average response selected 1.85 versions.

Version	2023
Windows 10	81.1%
Windows 11	55.1%
Windows Server 2012 / 2016	12.6%
Windows 7	12.0%
Windows 8	7.1%

Windows XP	4.6%
Windows Server 2019	2.9%
Windows Server 2008	2.6%
Windows Vista	2.3%
Windows 2000	1.4%
Windows Server 2003	1.1%
Windows Server 2022	0.9%
Windows CE/Embedded	0.6%
Windows 98	0.6%
Windows 95	0.3%

7. Building curl

This data also remains the same from previous years. Over **60%** of the users answering the survey do not build curl themselves. I think the fact that almost 40% do it tells something about the developer focus among the surveyed population.

Out of those who do build curl themselves, the configure/autotools build method remains the most popular one. The median share over the seven years we have asked this question is 60% of those users, which is also exactly the share of people who answered “configure” this year.

8. Features

What do people do with curl? This is one of the questions that shows changes year-to-year.

HTTP/2 flatlines at 65.4% this year, up just 0.2 points from last year but instead HTTP/3 continued its growth up to 26.9% (from 21.3% last year). A significant number as the feature is still **experimental** in curl. Hopefully it will leave that state for next year’s survey and then I assume we can see it grow further.

The largest changes this year compared to last year is the somewhat unexpected SOCKS proxy at 22.2% (up 5.5 points) and HSTS at 15.9% (up 4.6 points). libcurl multithreaded use keeps climbing, at 25.3% this year (up from 22.1%).

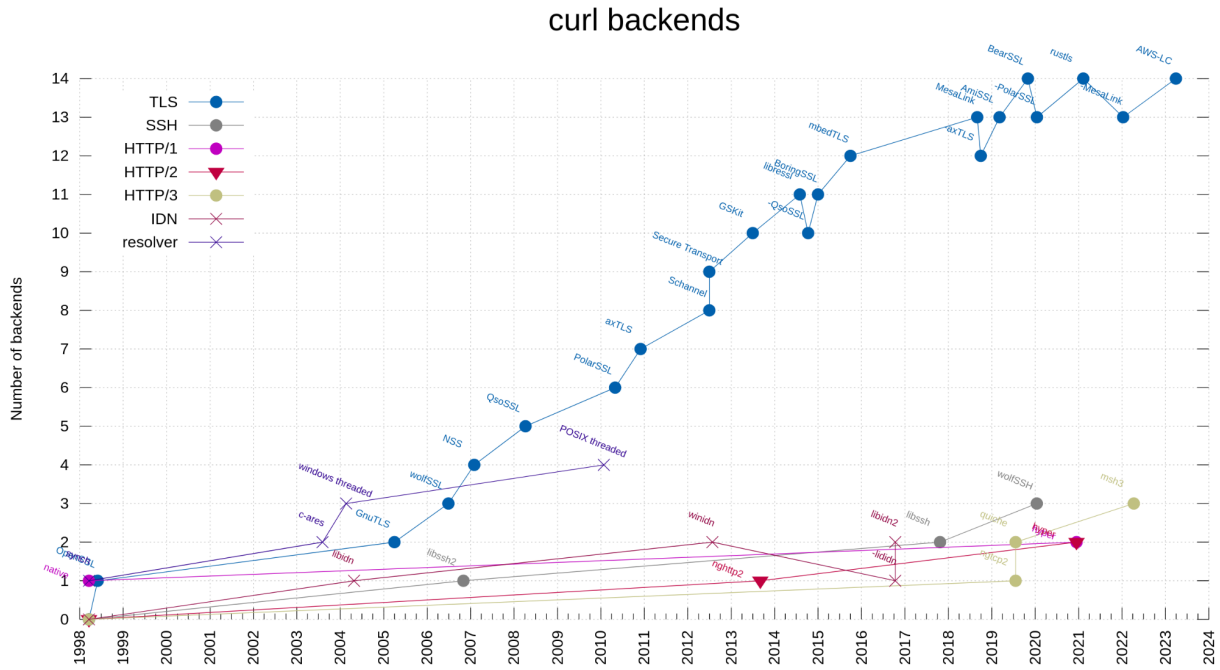
The feature that has changed the most in 2023 compared to the median usage share, is TLS client certificates that is down 15.6 points from the median to this year’s 20.3%.

On average, users selected 3.7 features each.

Feature	2023
HTTP/2	65.4%
HTTP proxy	33.1%
HTTP automatic decompression	28.2%
HTTP/3	26.9%
using libcurl multi-threaded	25.3%
SOCKS proxy	22.2%
HTTPS proxy	20.7%
TCP keepalive	20.5%
TLS client certificates	20.3%
HSTS	15.9%
UNIX domain sockets	12.6%
DNS-over-HTTPS (DoH)	11.6%
curl_multi_socket API	11.6%
.netrc	11.4%
Bandwidth rate limiting	10.1%
NTLM auth	8.7%
HTTP/0.9	8.5%
CURLOPT_FAILONERROR	8.1%
Alt-svc	5.6%
the share interface	5.2%

9. TLS backends

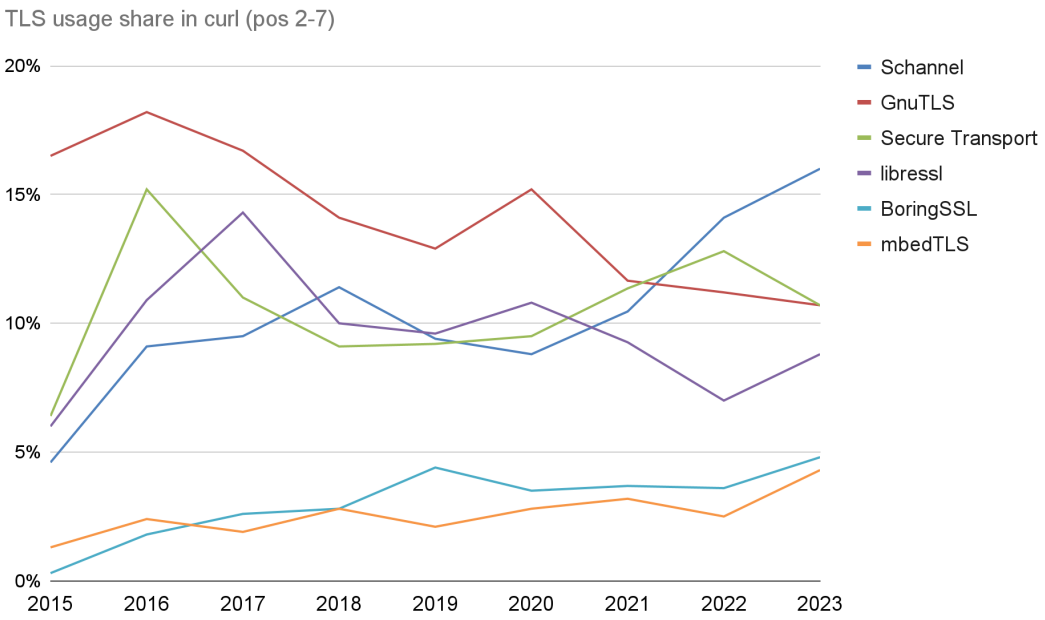
At the time of this year's survey, curl supported 14 different TLS backends, up one from last year. AWS-LC being the newcomer, another OpenSSL fork.



This year 26.2% said they did not know their curl's TLS backend. It might be a good thing that not all users know this, as it is not supposed to be an important factor for users. This is the fourth consecutive year with roughly one out of four not knowing.

OpenSSL remains the undisputed king of TLS backends for curl users at 71.6%. Which converts to 97% of the ones who did not select *I don't know*. Of course we cannot tell if some users still say "OpenSSL" even if they in reality might be using one of the many OpenSSL forks.

In the popularity contest position 2-7, there are signs of changes. Schannel climbs to its highest score yet (16.0%) while GnuTLS keeps going in the other direction (10.7%). Secure Transport and libressl keep fluctuating - presumably at least partly because of how macOS has been going back and forth. Both BoringSSL and mbedTLS climb to their respective highest usage share ever at 4.8% and 4.3% respectively.



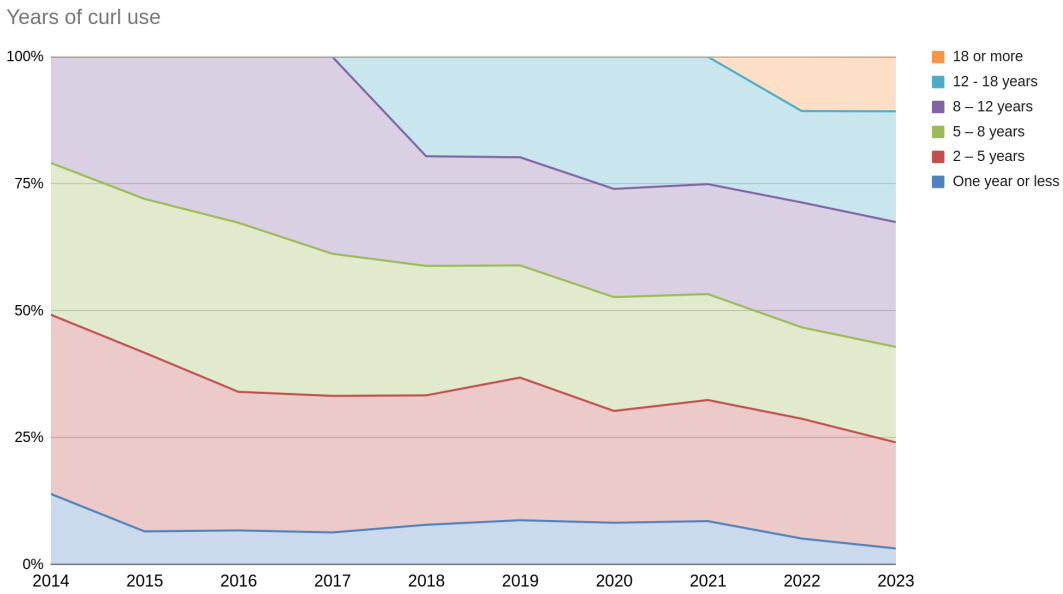
The complete distribution in 2023 looks like this:

TLS	2023
OpenSSL	71.60%
I don't know	26.20%
Schannel	16.00%
GnuTLS	10.70%
Secure Transport	10.70%
libressl	8.80%
BoringSSL	4.80%
mbedTLS	4.30%
wolfSSL	2.90%
rustls	2.90%
NSS	2.60%
BearSSL	1.70%
AmiSSL	0.50%
gskit	0.50%
AWS-LC	0.50%

10. Years of curl use

This is the curl 25 year anniversary year. The general sense is that we have quite happy and loyal users. curl mostly delivers on its promises. It is a solid and trusted tool. Many of our users have used curl during their entire computer lives.

Last year we introduced an answer alternative for *18 years or more* as previously *12 years or more* was the top alternative. There might be a lower share of *one year or less* respondents. It could be a sign that newbies do not use curl to the same extent as they did in the past?



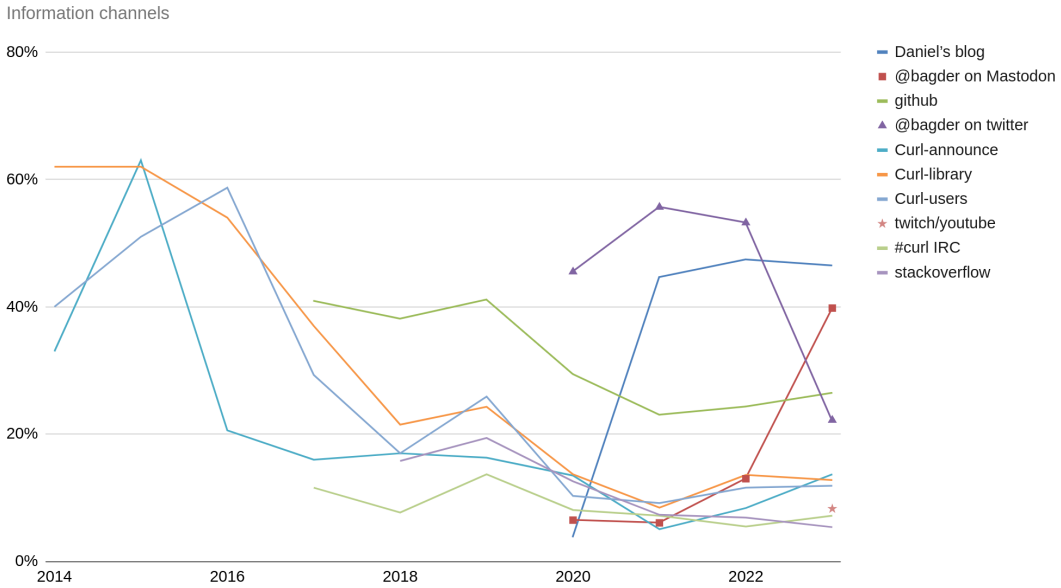
11. Participating channels

Where do users learn and talk about curl? This of course depends partly on where information is shared and discussions are held in the first place, but also on where users prefer to hang out.

This year shows a few significant changes compared to previous years: the share of Twitter users have plummeted, from 53.2% last year to 22.2% this year. The opposite development was instead seen in followers of @bagder on Mastodon, which bounced up to 39.8%, up from 13% last year! Explanations for this might include the general Twitter demise but also that Daniel has shifted a lot of curl related talk from Twitter to Mastodon.

Interestingly, this development has now made *Daniel's blog* the resource most people say they follow for curl information, with 46.5% of users selecting this option.

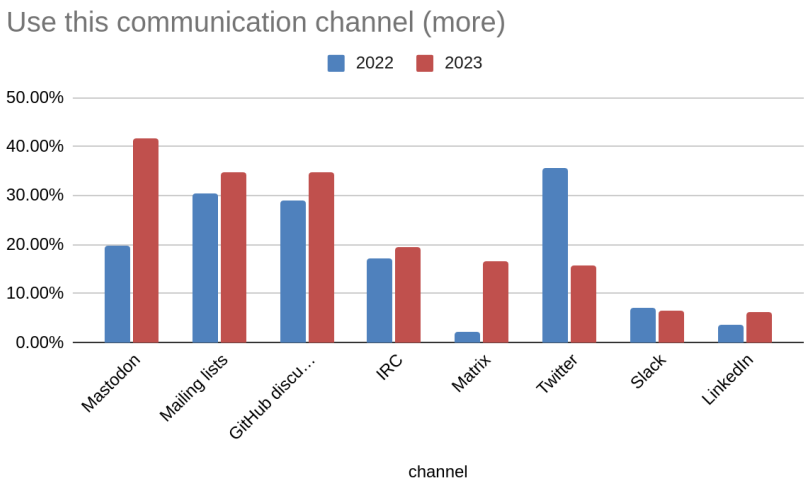
The new answer option for the year, *Twitch/YouTube*, lands at 8.3%. A significant share.



Channel	2023
Daniel's blog	46.5%
@bagder on Mastodon	39.8%
github	26.5%
@bagder on twitter	22.2%
Curl-announce	13.7%
Curl-library	12.8%
Curl-users	11.9%
twitch/youtube	8.3%
#curl IRC	7.2%
stackoverflow	5.4%

In 2022 we added a question to the survey about what communication channels users think we should use or use more. To talk about curl matters and subjects presumably.

In 2022, the top requested communication channel was Twitter. In 2023, Twitter fell down to the 6th place (to 15.7% from 35.7% last year). Instead Mastodon (41.7% up from 19.7%) took the first place while mailing lists and GitHub discussions remain popular options. This graph below shows the choices with 5% or more this year, compared to last year. It is only Slack and Matrix on the list that we do not already "exist on".

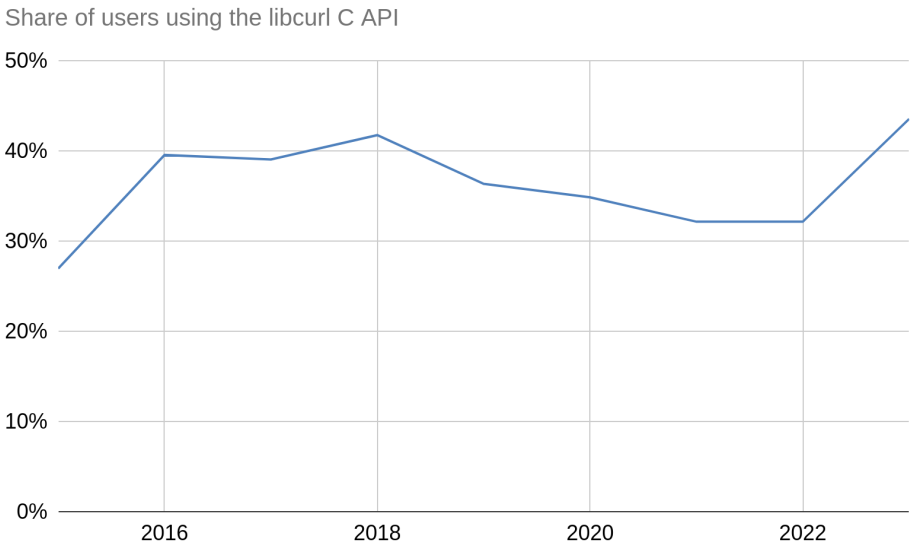


12. Accessing libcurl

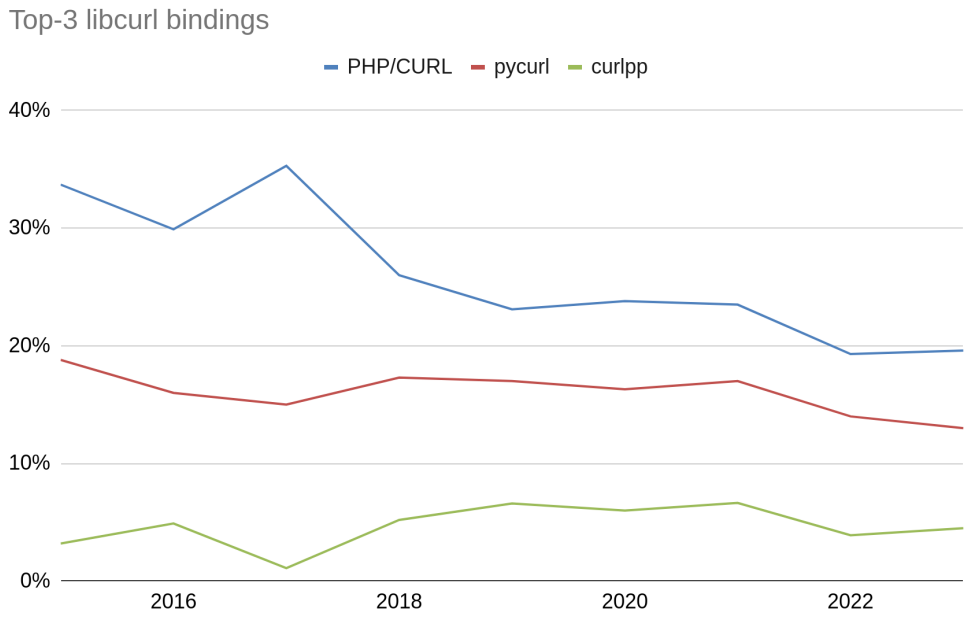
libcurl is the network transfer engine of the command line tool and is commonly accessed by users via bindings. The bindings are what makes libcurl truly accessible to almost all developers everywhere. Which ones do people use?

78.4% answered the command line tool curl, almost identically to the 78.5% from last year. There is no surprise a large share of users use the command line. This year users selected 1.92 answers on average.

The plain C API reached an all-time high selection rate with 43.6% (up from 32.2% last year).



The top-3 most used bindings below the C API remain the same as they have been since 2015 when we began asking this question: PHP/CURL, pycurl and curlpp. The graph below shows with clarity that PHP/CURL usage share has been shrinking gradually for many years, pycurl has also decreased a little while curlpp remains at roughly the same year over year, albeit at a notably lower share than the other two.



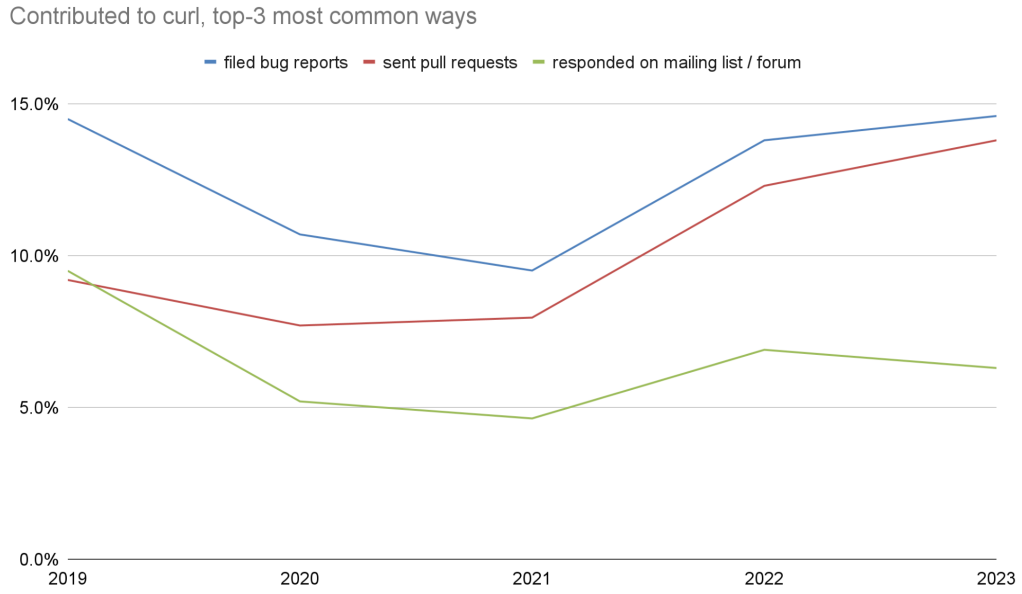
Here's the complete binding distribution for 2023:

Binding	2023
curl	78.4%
plain C	43.6%
PHP/CURL	19.6%
pycurl	13%
curlpp	4.5%
.NET core	4.3%
Rust-curl	4.3%
www::curl (perl)	4.3%
Node-libcurl	4.1%
Go-curl	3.9%
Ruby	2.4%
Java	2.2%
Common Lisp	2%
Lua	1.5%

cpr	1.3%
Tclcurl	0.9%
R curl	0.7%
FreeBasic	0.6%
ocurl	0.6%

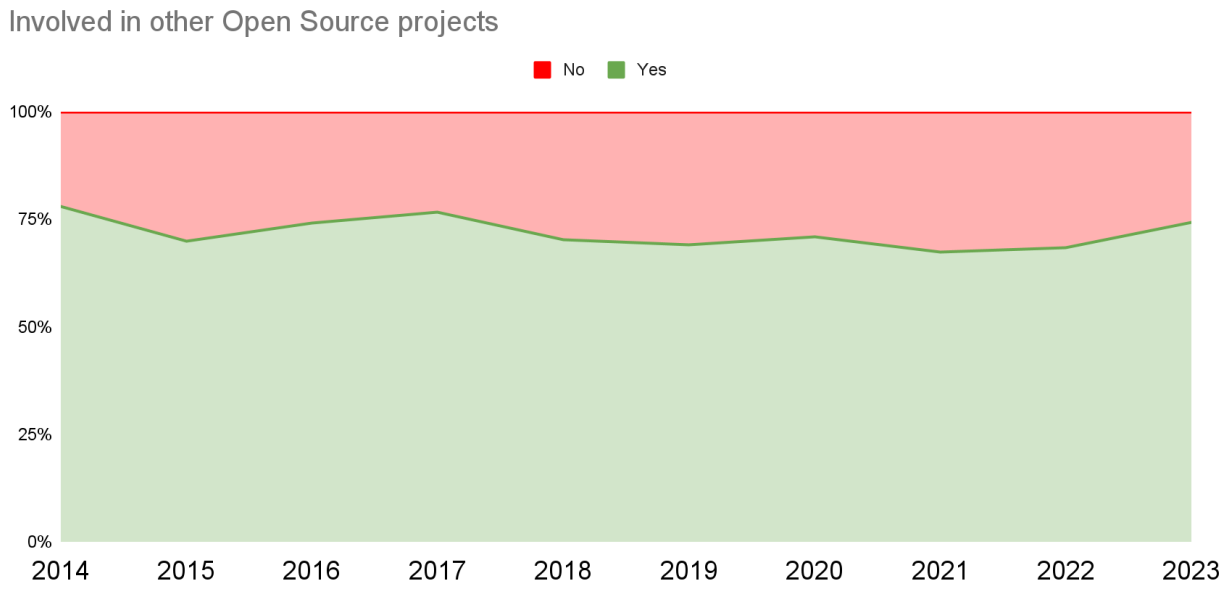
13. Contribution

Users still 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. About one in seven (14.6%) say they submitted a bug report and that just does not scale for the entire user population. 13.8% submitted a pull request!



Help	**2023**
I haven't contributed yet	71.0%
filed bug reports	14.6%
sent pull requests	13.8%
curl stickers on prominent places	6.4%
responded on mailing list / forum	6.3%
helped out in other ways	4.7%
donated money	3.8%
can't remember	3.6%
spend time in the IRC channel	3.4%
run tests or provide infrastructure	2.5%
write documentation	2.1%

curl users continue to be involved in other Open Source projects to a very high degree. We are but one cog in a huge Open Source machinery ecosystem. 74.3% said yes in 2023 to the question *Are you involved in other open source projects?*



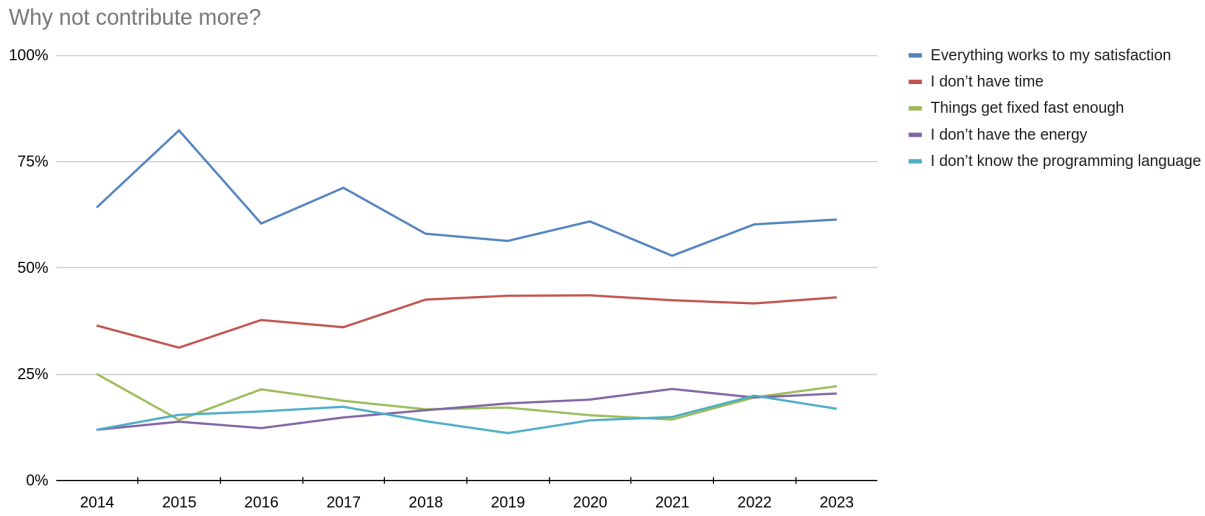
When asked for the reason why respondents have not contributed or not contributed more, the reasons they state are:

Why not contribute?	**2023**
Everything works to my satisfaction	61.4%
I don't have time	43.1%
Things get fixed fast enough	22.2%
I don't have the energy	20.5%
I don't know the programming language	16.9%
Too hard to get started	12.4%
I don't like or approve of github	5.3%
my work/legal reasons prohibit me	3.8%
I can't deal with the tools	2.2%
the project doesn't want my changes	1.7%
I don't like or use email	1.6%
I find it hard to work with the curl developers	1.0%

Presumably, this is what an old and mature project should get. Most things work, people are busy and the issues that do appear are typically fixed rather swiftly. curl is written in C, a language that is not the most hip one and probably is not even taught to younglings these days which might explain the 16.9% who do not know (or like) the language.

The disapproval rate of GitHub is at an all-time high at 5.3% up from 4.4% last year

Looking at the top-5 reasons over the years, they remain stable over time. This could be interpreted that the project keeps performing at a similar level.



This year, 122 respondents filled in something in the free-text form asking "*What could the curl project do/change to get (more) contributions from you?*".

It is hard to summarize free form replies, but a few patterns could be spotted:

1. Jokes about adding bugs, creating time machines and adding more hours to the day
2. Entries saying in many different ways they've found nothing to contribute because things work well already
3. "I don't know"
4. At least three mentioned "leave GitHub"
5. Variations of *list issues marked as "good for first time contributors* and other ways to make it easier to find where to contribute. Also a bunch of related entries asking for "better documentation"

That last category was also brought to our attention last year and is generally something we are aware of. We have improved documentation over the last few years, including “getting started” entry points, that is meant to help newcomers to the project find where to start. We also list TODO items and known bugs that are ready to get grabbed.

We do however not provide “good first issue” labels or similar on reported issues. The reason for this is simple: all our easy issues are fixed and closed almost instantly. We take great pride in our speed and agility in acting on reported mistakes. In order to provide such beginner labels on issues we would have to deliberately keep issues open (for how long?) and somehow restrict who would be allowed to fix them. I do not see us doing this in the near term.

Better documentation is of course something we want and we are constantly and always working on improving that. I think we need more specific requests and guidance to know what more and in what areas we should improve next.

14. How good is the project and its members

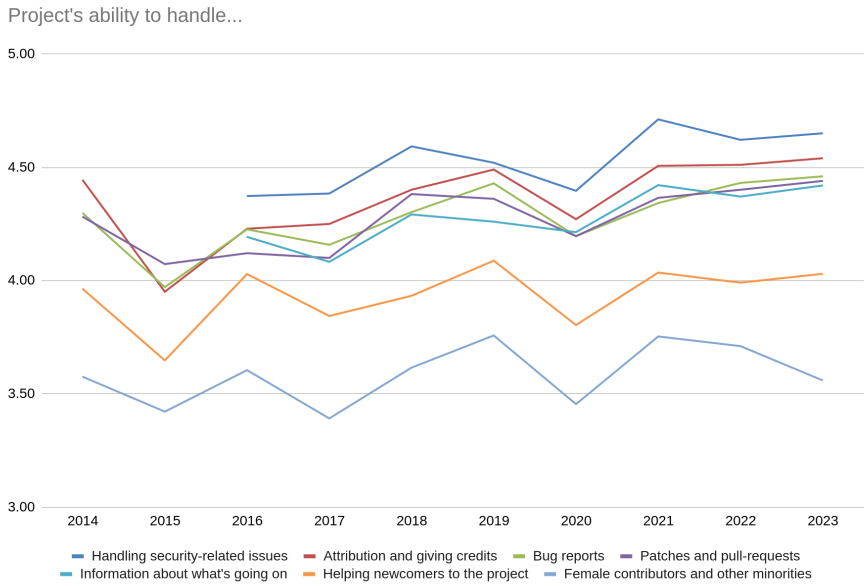
The respondents were asked to rate how good we are at handling things in these seven different areas on a scale from 1 to 5. From really bad to really good.

1. Handling security-related issues
2. Attribution and giving credits
3. Bug reports
4. Patches and pull-requests
5. Information about what's going on
6. Helping newcomers to the project
7. Female contributors and other minorities

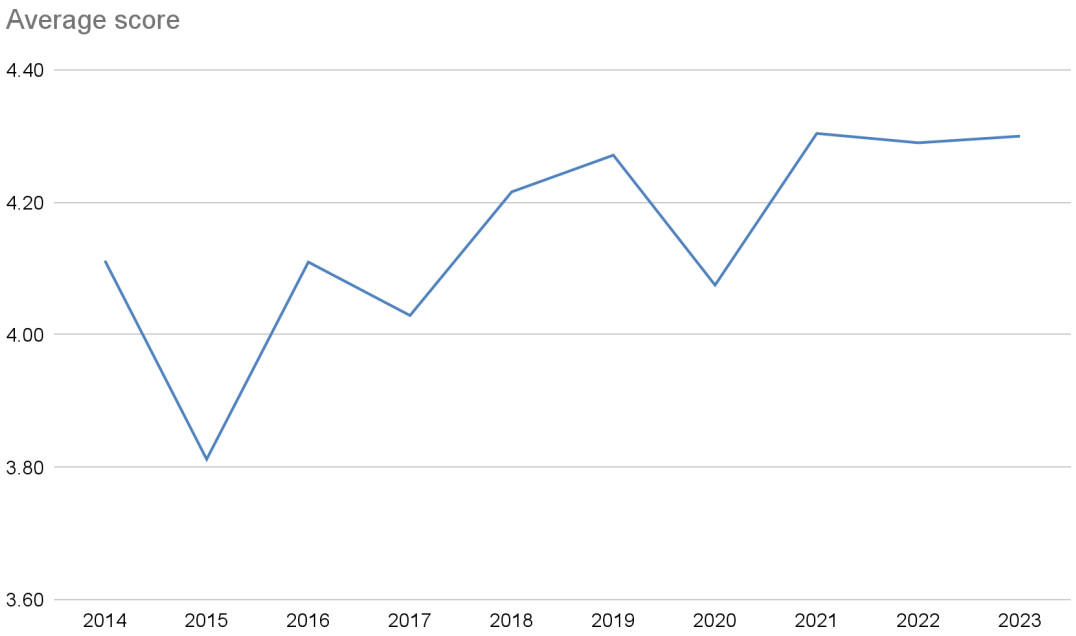
The order of the above areas are also the order of how good the users rank us. The order has remained remarkably similar over the years. I think the bottom two areas are what we should take away from this and dig deep to see how we can improve.

A note-worthy take-away from this year might be that compared to last year, we got higher scores in all areas except one. The increments were very small but still. The one area where we got a lower score than last year is the one we already scored the worst in: *Female contributors and other minorities*. We went down from 3.71 last year to 3.56

this year. I am interested in improving on this but I do not know how. I am probably too ignorant or too much of a problem myself.



Counting the average score on all 7 areas and plotting that in a graph shows that we remain at a high average score. It seems to imply we are not doing too bad as a project. This year’s average score is 4.30, up from 4.29 last year and identical to the score of 2021.



It is easy to suspect that the users filling in the survey generally have a positive attitude towards the project and therefore it is probably expected to get an above-3 average rate. However, reaching an above-4 rate consistently year over year (with the shifting audience every year) can probably be read as something more than just a skewed survey population.

15. Best/worst areas

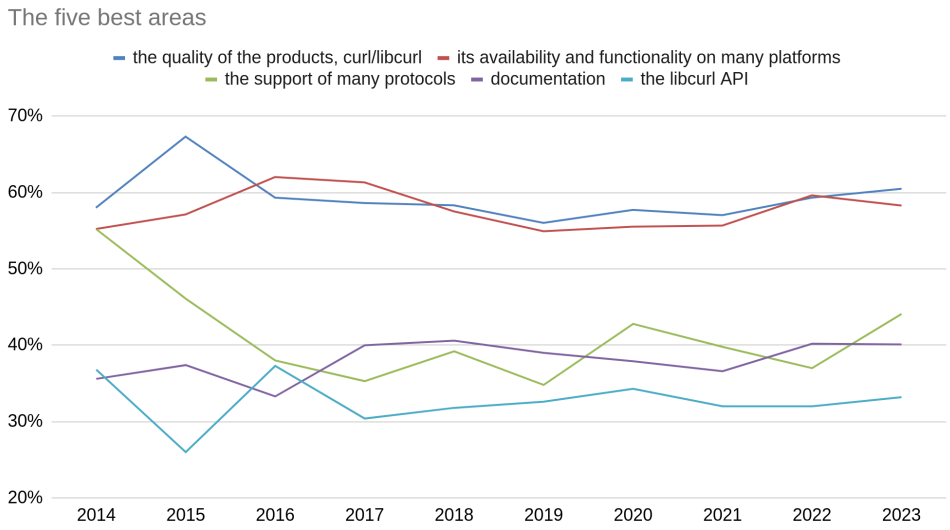
Users keep selecting roughly the same areas as the best ones in curl. Even the least selected area (build environment) was still selected best by 3.5% of users.

The three best areas of curl are: the **quality of the products**, **its availability and functionality on many platforms** and **the support of many protocols.**

Area	2023
the quality of the products, curl/libcurl	60.5%
its availability and functionality on many platforms	58.3%
the support of many protocols	44.1%
documentation	40.1%
the libcurl API	33.2%
standards compliance	27.6%
security	24.2%
the features of the protocol implementations	24.0%

support of multiple SSL backends	20.2%
project leadership	16.7%
footprint of the library/executable	15.0%
bug fix rate	13.6%
transfer speeds	10.9%
the user and developer community	10.2%
welcoming to new users and contributors	6.1%
test suite	5.0%
project web site and infrastructure	4.6%
its build environment/setup	3.5%

The top-5 best areas has shifted like this over the years since 2014:



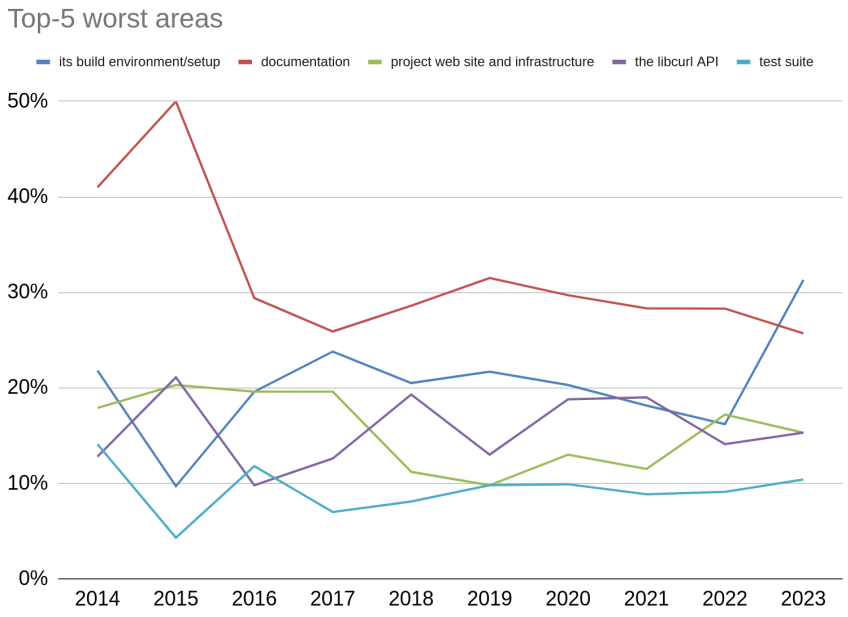
The question for best and worst areas feature the exact same set of areas to select from and while this year 521 persons answered the best areas question, only 144 answered the worst area.

This year brought a significant trend shift as *the build environment* bumped up 15.1 points to 31.3% as the new leader of this table. It being the least selected among the best matches this pretty well.

The number two among the worst areas is harder to take, as that is *documentation*. This has previously always been rated worst while simultaneously being rated 3rd of 4th

best. Proving that documentation is hard and needs constant improving. There will always be details missing and there will be users who expected to find that missing information or who found outdated information and that then shows up in this survey question.

Areas	2023
its build environment/setup	31.3%
documentation	25.7%
project web site and infrastructure	15.3%
the libcurl API	15.3%
test suite	10.4%
welcoming to new users and contributors	9.7%
the features of the protocol implementations	6.9%
the support of many protocols	6.3%
footprint of the library/executable	6.3%
security	5.6%
transfer speeds	5.6%
bug fix rate	4.2%
support of multiple SSL backends	4.2%
the user and developer community	3.5%
its availability and functionality on many platforms	3.5%
the quality of the products, curl/libcurl	2.8%
standards compliance	2.1%
project leadership	1.4%



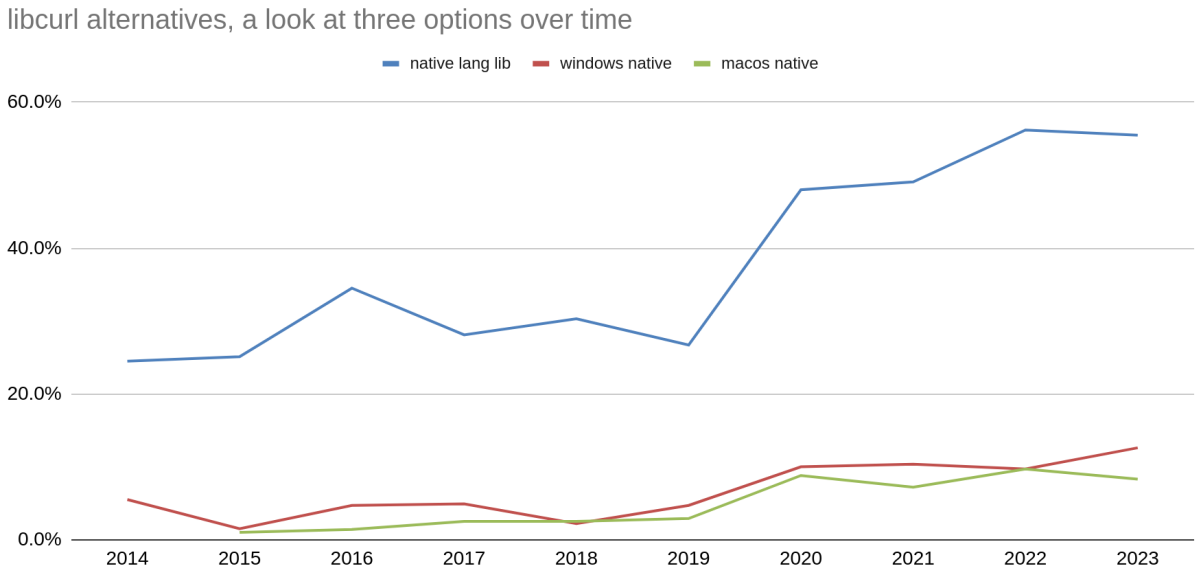
16. If you couldn't use libcurl, what would be your preferred transfer library alternatives?

Keeping an eye out for the competition and where to find inspiration for what libcurl is missing and how to further improve. The answers indicate that there is no competing portable “system-level” internet transfer library of significance.

The top option (61.1% up from 60.0% last year) is “wget (or code ripped out from it)” which is just a specific flavor of “homegrown” which is the 3rd rated answer (13.5% up from 11.6%). I think it either says that users have strong confidence in their own abilities or it says that the distance to proper alternatives is long. The second most chosen answer “native lang lib” at 55.5% (down from 56.2%) seems sensible as within different language ecosystems there are a huge variety of competent solutions. For Python, Perl, Java, JavaScript and the lot.

Alternatives	2023
wget	61.1%
native lang lib	55.5%
homegrown	13.5%
windows native	12.6%
macos native	8.3%
asio	6.0%
Qt	6.4%
libsoup	2.9%
poco	3.5%
Cpp-netlib	1.0%
neon	0.8%
serf	0.8%
libfetch	0.8%

The three alternatives that have changed the most compared to the median value, over the ten year span we asked this question, are *native language library* (up 23.1 points), *windows native* (up 7.4 points) and *macOS native* (up 5.4 points).

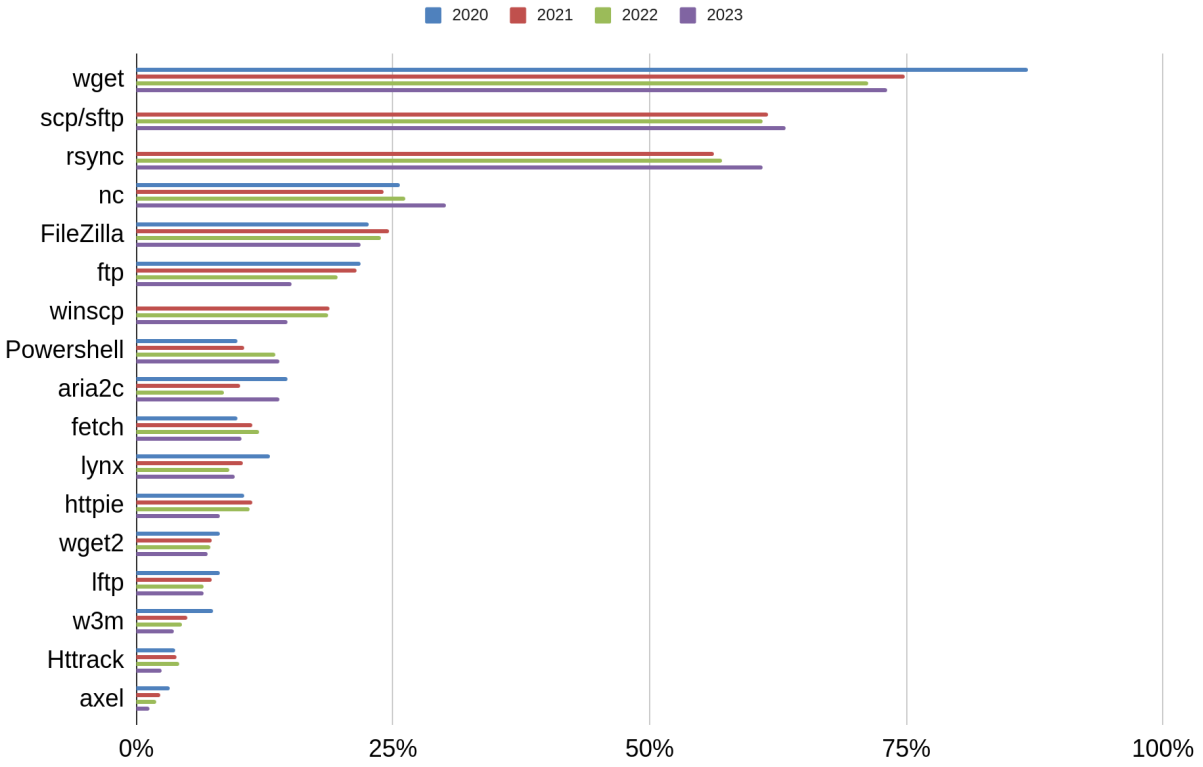


17. Which other download utilities do you normally use?

The top-3 download tool friends of curl are wget (73.1%), scp/sftp (63.2%) and rsync (61.0%). This is the fourth year we included this question and the response distribution remains very similar year to year.

The biggest differences compared to last year are **aria2c** up 5.4 points to 14.0%, **rsync** up 4 points to 61.0% and in the other direction **ftp** is down 4.5 points to 15.1% and **winscp** is down 4 points to 14.7%.

Other download utilities used



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

How do we know what features and protocols to add support for in curl? One important step in this process is to figure out what users would like to see, what we think is missing already and what cool new things that would be appreciated if added. With the risk of course that it is very easy for everyone to just say *sure I want that* but without any particular thoughts or plans for what to do with the feature if we were to provide it.

I added four new entries this year and removed a few - the new ones are highlighted with a yellow background in the table. WebSocket, which was always a feature selected very frequently in past surveys, was implemented late last year and is now on track to enrich the world of curl users so therefore no longer present in this table.

Missing features	**2023**
More JSON	33.7%
DNS-over-TLS	24.0%
Parallel connection DL	22.8%
Recursive HTML DL	21.9%
gRPC	20.0%
DNSSEC (DANE)	18.1%
TLS fingerprint-counter	17.0%
GraphQL	14.9%
SRV records	14.0%
DNS-over-QUIC	13.5%
Auto-detect proxy	13.3%
SMB v2/v3	13.0%
ECH (ex ESNI)	12.8%
WHATWG URL syntax	12.3%
Gemini	11.4%
AIA (dl certs)	10.0%
OCSP	10.0%
0-RTT/early-data	9.8%
DNS: URLs	8.6%
dynloadable protocols	8.4%
ManageSieve protocol	2.3%
COAP	2.1%

Users also provided a lot of free form proposals. I have edited and filtered them somewhat and I show them below. It looks like we can keep ourselves busy for a while longer.

- Better use of cookies from browser profiles
- help deal with OAuth2
- better Windows support
- rsync
- Apple Network framework TLS/QUIC
- H3 Proxying
- DNS HTTPS records
- UDS for a WebSocket
- MQTT QoS 1&2
- MQTT Subscribe
- Expose libcurl's threaded resolver via a new API or the curl multi API
- Recursive WebDav downloads
- Oblivious HTTPS
- Colored output
- WebTransport
- Connection pool sharing (threadsafe)
- Onion URLs
- Improved DNS handling (c-ares does not resolve everything), a single lookup thread per request is bad when using multi mode
- RFC 9102 TLS DNSSEC Chain Extension
- WebSocket support in command line tool
- UUID generator
- Better AWS V4 signing support for curl (the cli)
- Custom per-request configurable DNS resolvers independent of the system set-up
- Adaptable Logging and Tracing in production
- Option to change HTTP header order
- S3:// endpoint compatibility
- Support RFC 3230 “Instance Digests in HTTP”
- data: url support
- Type-safe curl_easy_getinfo and curl_easy_setopt
- libcurl handling of multiple proxies, "proxy1:port1;proxy2:port2"
- Official Visual Studio builds for libcurl
- A Windows GUI
- Support for X-RateLimit header, i.e. pause automatically when reached rate limit indicated via received header

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

The distinction here between the previous question is that this is clearly a libcurl question. About APIs provided to applications using libcurl.

The answers to this question are difficult to compare to previous years, but the answer remains *JSON generation/parsing,* a top selection every year since 2016 and yet it is so vague that it is hard to understand what exactly such an API would do and what users are actually asking for when they select this answer. Clearly it shows there's a keen interest in (more) JSON.

The 2nd most selected answer is a trick one since we already provide the *fcurl*[1] project since several years back that offers exactly this functionality and yet it does not seem to be used much or attract much interest.

API	2023
JSON generation/parsing	56.5%
a read()/write() style API for downloading and uploading	29.1%
A zero (or fewer) copy API	25.2%
Server-side support library for HTTP(S)	23.5%
Better aid for doing multi-threaded transfer applications	21.2%
HTTP Content-Disposition header parser/helper for applications	16.3%
Pluggable async DNS resolver	12.7%
Per-multi bandwidth limitation settings	6.2%

[1] https://github.com/curl/fcurl

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

I am not sure any of these will actually make it into next year's survey:

- What curl problem is the most problematic to you?
- How often do you update cURL and how do you receive updates?
- How would you rate the vulnerability information provided by the curl project?
- Maybe more short answer questions?
- More on diversity
- Opinion on documentation depth and source organization
- Should curl move to have its own built-in HTTP/3 code? (eg. needing only ngtcp2 but not nghttp3, msquic but not msh3)
- Some questions about pycurl and maybe other binding.
- how many devices you have installed/can use curl
- To which collaboration platform should development migrate?
- What are some of your favorite projects that heavily use libcurl? NetSurf (a real web browser!) and Nheko are some of mine.
- What curl CLI flags do you use the most?
- What is a link to a URL/site that has provided you helpful information on integrating curl or libcurl into your projects?

21. Anything else you think we should know?

This is an open ended question that allows respondents to pass on whatever message they felt the people of the project should get or know about. We got 87 write-ins, where most of them were different variations of saying thanks. This is deeply heart-warming and encouraging so I have purposely included most of them unfiltered below for project members to take in and to use in coming days when things might feel less inspiring.

- A website (subdomain) specifically for learning and testing. For example: a page that is designed specifically for http methods and/or html form submissions for people learning how to really use curl.
- Absolutely primal simplicity. Extra. Just go on. We appreciate you enormously. THANKS.
- Although curl still has some issues, it is still very useful. Thank you for your work!
- An awesome project - many thanks!
- bagder is awesome <3
- Cool survey, I'll look for it in future years! It's neat to start to think of curl as an

active project and not just "a Unix utility I can take for granted" like many of the ones from the 80s.
- Curl and also sqlite let me believe that maybe not all computers are broken. Thank you!
- Curl is a great project with excellent leadership managing to look from different angles. Looking forward to new features and improvements!
- Curl is an indispensable program. It must be up there with git for things I use every day. Love it.
- curl is awesome!
- curl is great, thank you
- curl is great!
- Curl is possibly the best command line tool out there! (or at least in my top 5 :)
- cURL is really great. But because it supports so many protocols and stuff I wonder if a plugin system would make sense.
- cURL rocks!
- curl's great
- Don't use google forms :) Thank you for taking the time out of your day to ask about our opinions!
- Don't focus on diversity or other social nonsense, please just make curl the best!
- Even if you don't add any new features this year, curl is good and extremely handy.
- Everything is awesome (with the Lego soundtrack). Thanks again for all the work over the years for this amazing tool
- Good tool, I use it frequently, thanks!
- Good work, thanks
- How do I get these stickers that one of the questions mentioned? 😄
- I ♥ cURL and will use it regardless of what changes based on this survey
- I am greatly indebted to curl - the amount of times it has helped me to as a software engineer debug HTTP, HTTPS, TLS, nginx, apache, python, PHP, node, bash, authentication, uptime and generally anything else is incalculable. Thank you!!!!
- I appreciate all the work that goes into this product, even if I don't use half of the functionality. What I do use works quite well. Thank you!
- I don't need commercial support but if you'd offer paid training (in addition to official documentation) in the style of what Julius Volz (maintainer of Prometheus) does here https://training.promlabs.com/ I'd be interested to pay for that - both to support the project and to learn more. Just as an idea.
- I would love to help with curl, but I only know Fortran, no c at all.
- I do have one suggestion/request. I wish curl would be willing to do an ftp download without checking first for the size of the file.
- I know it's not libcurl's "fault" but the default packaging on ubuntu, etc includes all protocols which tends to bloat docker containers, etc. It would be awesome if a "Dynamic protocol loading" system would guide/help/steer downstream maintainers to split the packaging in optional packages. Otherwise, fantastic

work! Keep it up.
- I love the whole http://site.{one,two,three}.com formatting options for URLs.
- I primarily use libcurl for url grabbing, and the HTML response can be quite annoying to look through / parse. I think a HTML beautifier flag to make things a bit more manageable would be amazing.
- I really appreciate curl. I didn't realize it had so many features until I compiled libcurl
- I really enjoy your work and what the community has achieved with curl.
- I really love curl
- I semi-regularly run sysadmin and devops training to engineers starting out in the industry, and curl features part of the syllabus (APIs, shell scripting etc)
- I want to thank Daniel and all contributors for this awesome piece of software :)
- Keep doing this great work !
- Keep up the good work
- Keep up the good work!
- Keep up the good work. Thx!
- Keep up the great work and thanks for all!
- Keep up the great work, you are much appreciated!
- Keep up the great work!
- Libcurl is the best, we've done extensive comparisons of competing libs.
- Love curl, love that Daniel has been at it for so long and look forward to seeing the project continue to grow
- Love you all <3
- More detailed examples and documentation
- My interest in the project grew through the regular updates and videos, regarding a tool that I use and didn't think about much before. Really happy to see trurl added.
- My thanks to Daniel (and his fellow contributors) for libcurl.
- Our company has been using libCurl for many years. We may be not using it the most efficiently as the APIs have evolved so extra guidance on getting the most out of it would be helpful.
- Overall, you're doing a great job, and for such a long time already. Thank you!
- Please don’t use pie charts this time ;)
- Possibly worth adding Bittorrent to the downloader comparison chart -- not that I think curl should necessarily even ever attempt to support it, but it is a somewhat nice feature that Aria2 has over the others.
- Reducing CI flakiness and improving turnaround times might help contributing
- IR shows care that you now do these surveys. Really appreciate you setting the example!
- thank you
- Thank you for all your efforts :)
- Thank you for the amazing work and transparency of the project and the amazing blog posts
- Thank you for the great software!

- Thank you so much for your diligent work on a great project.
- Thanks a lot for this project
- Thanks for all the time and effort! :D
- Thanks for the hard work and good software. Curl works for me and is a great tool that keeps me happy.
- Thanks, for curl!
- The developers on this project do an excellent job and the libraries and utilities they come up with are very helpful.
- TLS and http2 fingerprinting is seriously affecting the free web, just merge the patches made by the curl-impersonate project.
- We love Daniel's help and response time. You are very friendly and kind to your users.
- Whenever people have a supposed API issue I force them or teach them to make a curl command that reproduces the issues. If they can, it is a bug with an easy test case, otherwise it is their client and their lack of understanding of HTTP.
- Where can I buy t-shirts and stickers?
- Words can't express the gratitude many people have, on how beautiful the impact of curl is to the world. It's just awesome.
- Yes, that i'm thankful for your great tool, that can do so much more than i can use
- You all are doing a great job and the world wouldn't be the same without cURL. Thanks a lot!
- you do so fine work
- You guys are amazing, thank you for the work you do :)
- You guys are doing great work!!!
- you're doing great work
- You're doing great work, please keep going!

22. Final words

Crunching the numbers, reading the comments, digesting the meaning, filtering the feedback and generating all these graphs in this report takes fricking forever, but I am happy to do it. This user survey is the only opportunity during the year to get wide and direct feedback from real-world users and I want to make the most out of it.

Enjoy this year’s analysis. I hope we do this dance again next year.

/ Daniel