# Politely Downloading Millions of WARC Files Without Burning the Servers Down

Pedro Ortiz Suarez, Thom Vaughan, Greg Lindahl

## Motivation

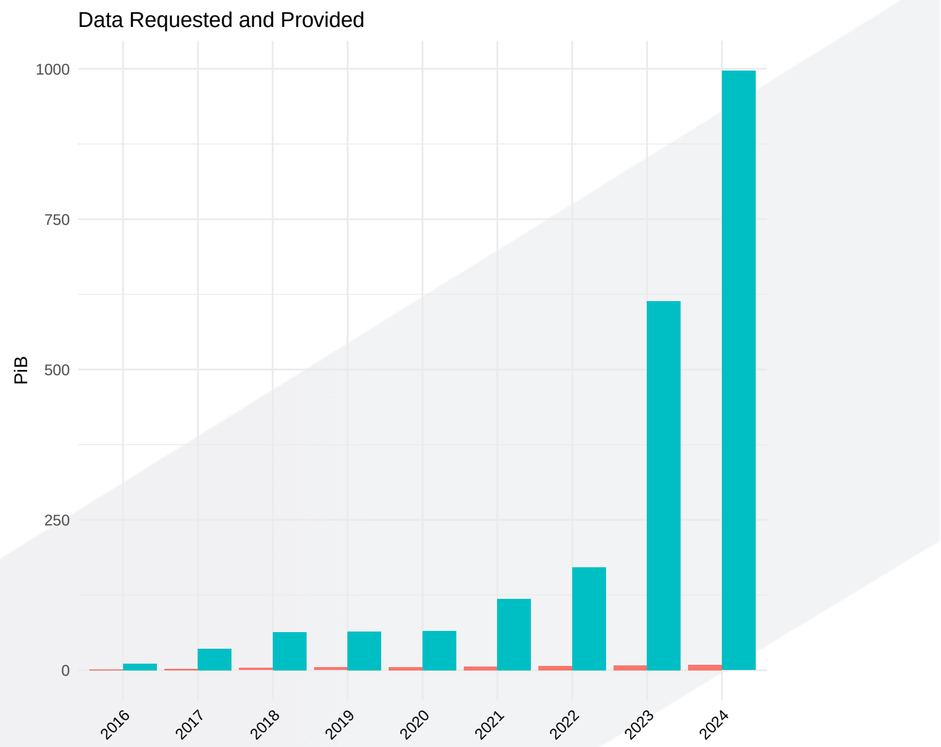


Figure 1. Amount of data requested and amount of data ”provided” or released by year.

- The advent of LLMs, AI and RAG has drastically increased the interest in web archives and web data.
- For Common Crawl this has translated into a significant increase in data requested in the recent years, cf. Figure 1.
- Common Crawl data is distributed via Amazon CloudFront [1], a content delivery network (CDN) on Amazon Web Services.
- Users generally use traditional clients like `curl` or `wget` without implementing any retry delay. They also sometimes implement custom concurrent clients with greedy retry strategies.
- The sudden increase in interest in Common Crawl's data, plus the aggressive retry strategies implemented by some downloaders has managed to overload the CloudFront as shown in Figure 2, making it extremely difficult to access our dataset for all users.
- Moreover, when the servers are overloaded, clients implementing greedy retry strategies send even more requests, further increasing the traffic and making our data virtually inaccessible.
- We then need to provide our users with a polite download client.

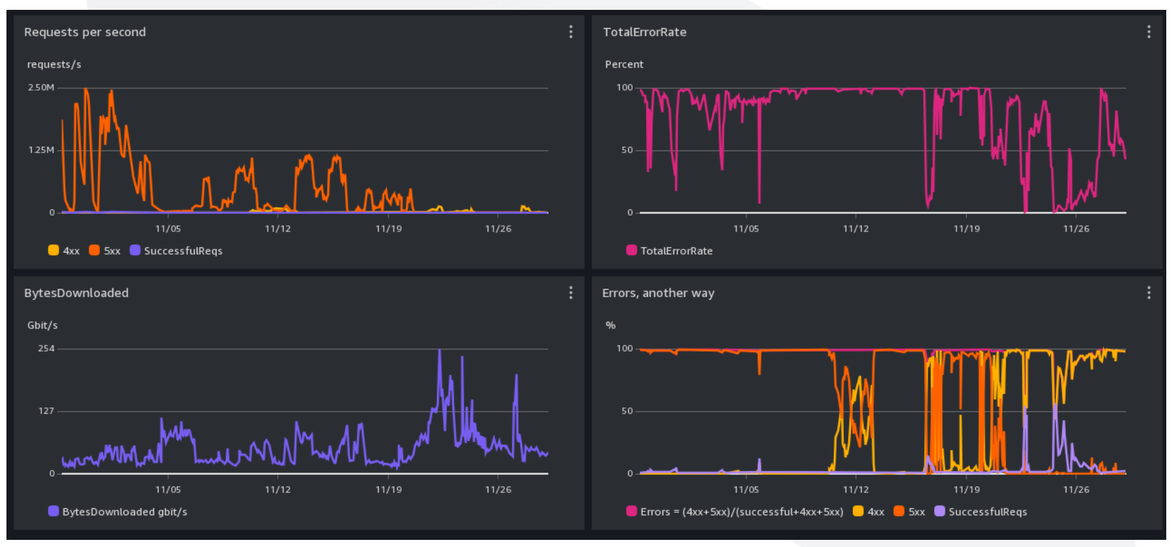


Figure 2. CloudFront (https) Performance Screenshot for November 2023

## About Common Crawl

Common Crawl is a non-profit organization which regularly crawls a significant sample of the web and makes the data accessible free of charge to everyone interested in running machine-scale analysis on web data.

At present, we crawl up to 3.0 billion web pages every month. The data is hosted in the Amazon cloud as part of the AWS Open Data program.

Contact:
`https://commoncrawl.org/`

`pedro@commoncrawl.org`
`thom@commoncrawl.org`
`greg@commoncrawl.org`

## Exponential Backoff and Jitter

- While greedy retry strategies significantly impact the performance of our services, they are also bad for users, as the more often they retry, the more they overload our servers and the harder it is to access our datasets.
- The problem of multiple users constantly requesting data over a network is not new, it has been studied since the 70s [2].
- The accepted solution to this problem today is to implement a retry strategy with *Exponential Backoff* [3] and *Jitter* [4], which today is part of the Ethernet Protocol [5] and has become *“ the pillar for how Amazon builds remote client libraries for resilient systems”* [6, 7]
- Exponential backoff is an algorithm that uses feedback (server errors) to multiplicatively decrease the rate of requests, in order to gradually find an acceptable rate. It can be modeled with a simple exponential function:

$$t = b^c \quad \text{or} \quad f = \frac{1}{b^c}$$

  where $t$ is the time delay applied between actions, $b$ is the multiplicative factor or base, $c$ is the number of server errors observed and $f$ is the frequency of the requests [3].
- Jitter is the deviation from true periodicity of a presumably periodic signal. In concurrent applications where multiple requests might come all at once, it is recommended to add some randomness or *Jitter* to the frequency of the retries, so that they don't happen all at once [7]. When used in addition to exponential backoff, one simply selects between $t = 0$ and $t = b^c$ at each retry.
- As exponential functions grow quickly, it is common to add an upper bound to the backoff function; this is also called *truncated exponential backoff* [3]. It is also common to add a lower bound to jitter so that retries won't happen immediately with $t = 0$.

## `cc-downloader`

- We use *Exponential Backoff* and *Jitter* to develop our own download client for Common Crawl, `cc-downloader` [8].
- We develop it in the `rust` programming language [9] and release pre-compiled binaries for Linux (ARM and x86-64), Mac (ARM and x86-64) and Windows (x86-64). We release `cc-downloader` under both Apache 2.0 and MIT licenses, so that other projects can reuse its codebase.
- `cc-downloader` uses a default configuration of 10 threads (10 concurrent downloads), a base of 2 s for the exponential backoff with an upper bound of 3600 s, a lower bound of 1 s for the jitter, and a maximum number of retries of 1000 per file requested.
- The default configuration can be tweaked but the user, and even the defaults, might change in the future as we study the usage of `cc-downloader`. For this we use

```
User-Agent: cc-downloader/X.X.X
```

  `cc-downloader` is distributed as a CLI tool, but we are planning on releasing it as a library and adding `python` bindings, to allow our users to add `cc-downloader` to their existing codebases.

## Links / Resources

Bibliography, source code and data:
`https://github.com/commoncrawl/cc-downloader`

*IIPC Web Archiving Conference 2025, 8 – 10 April 2025, Oslo, Norway*

COMMON CRAWL

# Politely Downloading Millions of WARC Files Without Burning the Servers Down

Pedro Ortiz Suarez, Thom Vaughan, Greg Lindahl

## References


[1] Wikipedia contributors. *Amazon CloudFront — Wikipedia, The Free Encyclopedia*. 2025. https://en.wikipedia.org/wiki/Amazon_CloudFront.

[2] Norman Abramson. “THE ALOHA SYSTEM: another alternative for computer communications”. In: *Proceedings of the November 17-19, 1970, Fall Joint Computer Conference*. AFIPS ’70 (Fall). Houston, Texas: Association for Computing Machinery, 1970, pp. 281–285. ISBN: 9781450379045. 10.1145/1478462.1478502. https://doi.org/10.1145/1478462.1478502.

[3] Wikipedia contributors. *Exponential backoff — Wikipedia, The Free Encyclopedia*. 2025. https://en.wikipedia.org/wiki/Exponential_backoff.

[4] Wikipedia contributors. *Jitter — Wikipedia, The Free Encyclopedia*. 2025. https://en.wikipedia.org/wiki/Jitter.

[5] Larry L. Peterson and Bruce S. Davie. *Computer Networks, Fifth Edition: A Systems Approach*. 5th. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011. ISBN: 0123850592.

[6] Marc Brooker. *Timeouts, retries, and backoff with jitter*. 2015. https://aws.amazon.com/builders-library/timeouts-retries-and-backoff-with-jitter/.

[7] Marc Brooker. *Exponential Backoff And Jitter*. 2015. https://aws.amazon.com/blogs/architecture/exponential-backoff-and-jitter/.

[8] Common Crawl Foundation and contributors. *cc-downloader*. 2025. https://github.com/commoncrawl/cc-downloader.

[9] The Rust Team. *The Rust Programming Language*. 2025. https://www.rust-lang.org.