EasyChair Preprint

№ 6123

Live Monitoring 4chan Discussion Threads

Ylli Prifti, Iacopo Pozzana and Alessandro Provetti

July 20, 2021

Live monitoring 4chan Discussion Threads

Y. Prifti, I.Pozzana, A. Provetti

Department of Computer Science and Information Systems
Birkbeck University of London
[ylli, i.pozzana, ale]@dcs.bbk.ac.uk

Keywords: 4chan; online conspiracy theories; social web; data retrieval; distributed systems; web communities

Abstract

The 4chan portal has been known for several years as a “fringe” internet service for sharing and commenting pictures. Thanks to the possibility to post anonymously, guaranteed by the total lack of a registration/identification mechanism, the portal has somewhat evolved to a global, if mostly US-centred, locus for the posting of extreme views, including racism and all sorts of hate speech. A pivotal role in the emergence of the website as a bastion of “free speech” has been played by the /pol/ board[1] , which declares its commitment to host “politically incorrect” discussions.

Several research groups have intensively studied 4chan structure, dynamics and contents. Thanks to works such as [4, 12], we now have a fairly clear description of how 4chan works and what type of discussion dynamics the site supports.

In particular, the latter work shed light on the extremely ephemeral nature of discussions, with threads lasting on the website for a few hours at most, and often just for minutes - depending on the traffic they generate - before being removed to make room for new discussion.

Given the fast-paced nature of the evolution of the content of the boards, and especially given how such ephemerality shapes the tone and the content of the discussion itself [4, 14], it is of extreme importance for researchers to be able to capture the content of the threads at various points over the course of their short lives.

To the best of our knowledge, the existing 4chan literature has relied either on autoptic exploration by the scholars [14], or on large scale data collection campaigns that drew their content from the archived versions of the threads [12], i.e. on copies of the threads as they appeared at the time of their closure, and at that time only.

In order to observe at a more fine-grained level the content on the website, we devised a “scraping” architecture, summarised in Figure 2, which based on the OXPath platform [9]. It enables the retrieval of the threads posted on a board at various points while they are still live.

[1] `https://boards.4chan.org/pol/`

1 Introduction

With an ever-growing number of registered domains, active sites and the overall size of the world wide web[2], combined with the more recent paradigm where the consumer is also content producer (as for example in online social networks and internet communities), the web presents unprecedented opportunities for data mining and information extraction. This opportunity is clearly reflected in the amount of research and scientific papers based on data retrieved from the web.
The web remains highly unstructured and attempts to apply a machine readable structure [3] have failed to become large scale standards. The human-centric nature of the data exchanged on the web means we are a long way from Tim Berners Lee's concept of semantic web. The need for machine readable content and necessity for content to be read by machines has evolved in different directions, adopting to fast paced web development trends. Whilst after two decades the semantic web has evolved [13], two other themes for data retrieval are as important and have evolved with similar pace: "web scraping" and "API exposure". These two methods are not mutually exclusive and somehow orthogonal to each other [10]. Large and well established Online Social Networks (and not only) have mature APIs that allow accessing samples of posts or tweets (for example Facebook and Twitter). Whilst in most cases querying APIs is a non-trivial way for data retrieval, this method rely on what is being provided by the API developer and might not always be a true representation of what end users experience [17, 19]. Web Scraping is more challenging but yet widely used both in the business world and for scientific purposes. De S Sirisuriya [20] categorised the different web scraping techniques in nine different groups. In a more recent analysis on web-scraping, Sarr et al. [7] apply a different categorisation based on "approach" with the following different approaches discussed.

1. Mimicry Approach
2. Weight Measurement Approach
3. Differential Approach
4. Machine Learning Approach

The considerations above need also be seen from another dimension. The web tends to be divided in three categories based on its reachability. Most commonly, when we talk about the web we are referring to the "Surface web" that tends to be reachable from traditional mainstream web engines. However, an even bigger and more information qualitative[2] part of the web is the "Deep Web". The Deep Web is usually hidden behind passwords, not linked to or many links deep that is difficult to reach with the traditional approaches of web crawling. Dedicated approaches are found in literature that address the Deep Web data extraction. Of particular interest for our research is the work of Gottlob et. al [8] on OXpath - an XPath extension for web crawling and scraping that is particularly successfully on extracting data from the deep web.
Another section of the web is the so called "Dark Web" that is usually hidden behind private networks and accessible via VPNs. The dark web is infamously known for the number of illegal activities happening in its realm.

Whilst presented with the challenge of retrieving a large amount of data from a large number of largely heterogeneous online social communities with ever more challenging characteristics

[2] https://news.netcraft.com/archives/category/web-server-survey/

of anonymity, ephemerality, lack of structure and machine readable data it became apparent that the literature and the tooling available to researchers lacked a holistic approach to addressing the problem. In fact we believe that the discussion of the following challenges should be addressed an a single body of work rather than in isolation:

- **Queryable**: We believe that an architecture that can reuse upon existing technologies for querying web pages and extend it to harness data across pages and sites, would be appealing to a wider audience, have better chances of universal applicability, sustainability and lower learning curve. When looking at the different techniques suggested by [20], apart from the semantic web, all techniques are either not repeatable (1, 6), require ad-hoc coding that mostly isn't transferable (2,3,4,5,9) or require you to learn and use proprietary and costly platforms (6,7). The semantic web on the other hand is not universal and often is not supported.

- **Scalable**: The OSNs are very large with Facebook estimated to have 2.5B active users[6] (i.e. 2.5B or more active profiles). Given enough resources, there needs to be a solution able to scale horizontally to harness OSNs the size of Facebook. Efficiency and cost should be embedded in the architecture. Similarly vertical scaling to potentially use the full extent of available resources, should also be a characteristic.

- **Distributed**: Web Crawling architectures[15] are often described as a dual process of discovery (breadth) and data extraction (depth) where architectural choices are made around priorities (for example breadth-first-search) and ordering of discovered URLs. As the system scales up, it is understood that multiple agents will be running at the same time. Distributed characteristics for synchronising, achieving common goal, avoiding effort duplication and conflict and finally a distributed storage to support storing of semi structured data are part of the fundamental architectural characteristics of the system.

- **Open Source and Extendable**: Open sources and extendability is a gap that has often been observed when approaching the issue of collecting data from the web. Often the tools are either not free and commercial, or they rarely incorporate more modern technologies, scale and are extendable

- **User Centric**: As discussed in [16] and [20] both APIs and different mechanisms for data scraping might have downsides and be inaccurate in terms of the data as seen by end users versus data retrieved. We believe in a WYSIWYG[3] mechanism where the data collection is as seen by the end user and not what is presented to a crawler agent, or what is provided via APIs.

- **Security, Privacy and Policy Compliant**: Online social networks represent user interactions. We believe in a system that makes the data security and privacy of each subject as a core characteristic. Data encryption, aggregations and anonymisation mechanisms must be at the core of the systems. Furthermore, access to (public facing) web pages needs comply with usage policy of the individual websites[4]. We understand that a more pragmatic approach must be taken when dealing with individual policies and the full extent of privacy and security. Part of the compliance cannot be incorporated in the system design and must be taken in consideration by the specific implementation and/or system use.

[3] Whilst WYSIWYG (i.e. the acronym "what you see is what you get") is often applied to web development IDEs, in this context is a good description of the meaning "user centric" being close to "as seen by the end user"

[4] These cannot be more restrictive that the legislation on the country where such web pages are begin consumed

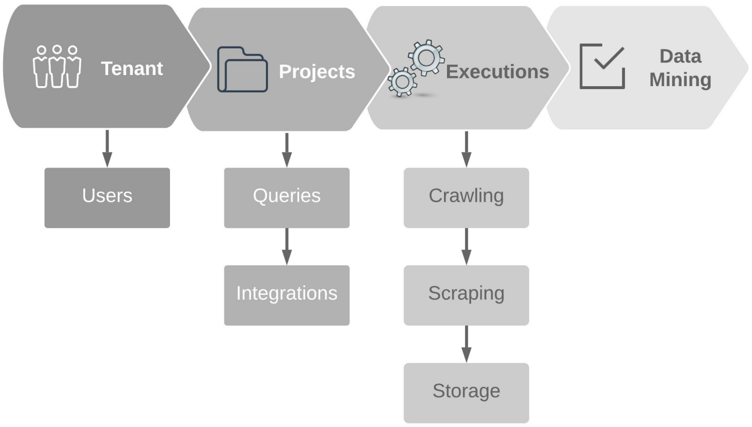

Figure 1: Tenant View

We addressed the problem with the above characteristics in mind and in the next sections will be presenting architectural design, implementation details and as a special use-case, a campaign of running the system for data retrieval from 4chan /pol board.

2 A distributed system for web data retrieval

We envision the resulting system being used by different groups of people that collaborate on the same web data retrieval projects. We call these tenants and diagram 1 shows how the usage of the system flows from the tenant prospective where:

1. Tenants have one or more users collaborating
2. Tenants create projects and for each project define one or more crawling queries and one (and only one) scraping query. Queries can be generated at run-time and retrieved as part of an API integration
3. The system will schedule projects for execution and will execute asynchronously the crawling queries. Results will be ordered and will be input to the scraping query execution. Structured and semi-structured results will be stored in persistent storage.
4. Data is made available to the tenant and its users for data analysis

To address the queraibility challenge the system was designed so that multiple query engines could be used. We have integrated two of them:

- OXPath and
- DR Web Engine

2.1 Using OXPath as query engine

Quoting the authors: *"OXPath is a web data extraction tool."* The source code of OXPath is openly available[5]. OXPath provides an effective [11] way to scrap websites and web pages. It

[5] https://github.com/oxpath/oxpath

is an extension of XPath language used to query XML documents. A typical OXPath query looks like the following listing:

Listing 1: OXPath based query

```
doc("https://www.google.com/search?q=web+data+extraction")
   /(//a[@id=’pnnext’][1]/ {click /})*
   //div[@id=’search’][1]//div[@class=’g’]:<links> [
      .//div[@class=’rc’]/div[@class=’r’]/a/@href:<link=
         string(.)>
     [? //h3/text():<title=string(.)> ]
    ]
```

The query structure is very close to the structure of the input (i.e. the pages you are trying to extract data from) and this helps with building the queries. OXPath is widely discussed in literature and has reached certain maturity that provides similar benefits [11] to the systems overall.

2.2 A JSON based query engine

OXPath is build in Java and uses Selenium and WebDriver [1] as underlying system to interact with the web browser and data for data extraction. We found it to be a non-trivial task to upgrade the underlying system and rely on more modern browsers[6]. On some occasions, this limited our ability to use OXPath on websites that only supported more recent browser versions.
The storage subsystem in our design is a Distributed DocumentDB, specifically MongoDB. The OXPath CLI[7] allows to specify the output format and support XML or JSON as output. Using JSON output highlights the discrepancy between the query structure and the structure of the output. At the cost of loosing the associated benefit of having close links between the query and the input, we designed a query engine, based on the same OXPath research, but that could address the the two challenges above: easier distribution and extendibility, and JSON based query structure (an extension over JSON). The data retrieval web engine is distributed as a python package[8] and is also provided open sources on GitHub[9].

Listing 2: JSON based query

```
{
    "_doc":"https://www.google.com/search?q=web+data+
       extraction",
    "links":[{
        "_base_path": "//div[@id=’search’][1]//div[@class=’g’
          ]",
        "_follow": "//a[@id=’pnnext’][1]/@href",
        "link": "//div[@class=’rc’]/div[@class=’r’]/a/@href",
        "title": "//h3/text()"
    }]
}
```

[6] whilst you can specify a different browser, depending on the version of selenium and WebDriver, there are limits on the version of browsers supported.

[7] Command Line Interface

[8] https://pypi.org/project/dr-web-engine/

[9] https://github.com/yllipriftí/dr-web-engine

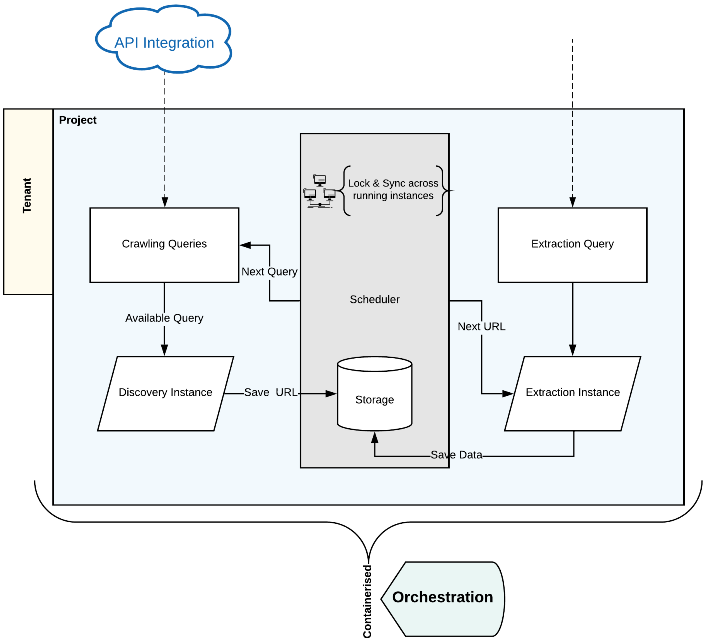

Figure 2: Dataflow of our Oxpath extraction engine

The immediate advantage of a JSON based query engine is that the structure of the output is represented in the query. In fact, if you remove the keywords used by the query engine, that start with an underscore, the remainder is exactly the structure of the output. This is unlike OXPath where the strucutre of the output isn't immediately visible.

2.3 System design

Leaving the full discussion of the system design outside of the scope of this article, we have highlighted in diagram 2 how the various core components interact together. In our implementation of the designed system, we used Kubernetes for orchestration, Docker as container engine and MongoDB for storage. The system can scale horizontally indefinitely, both as more nodes added to existing Kubernetes clusters and as more clusters running on the same project (i.e. crawling and extraction queries) - for as long as the storage distributed systems keeps at pace (i.e. there are enough MongoDB shards, replica sets and nodes to support storage needs).

2.4 Conclusion

Our scraping architecture has been carefully designed to be as general as possible without compromising on some what we believe to be core features. Such flexibility highlights an engine that through configuration and convention can behave and be used for multiple purposes. For example, whilst crawling and extraction queries suggest a breadth-first crawling engine [5], the crawling query can be written to both crawl and extract data and completely change the engine's behaviour.
Such generality, at the cost of some lack of operational optimisation, makes our architecture readily adaptable to changes, usable against multiple scenarios, as it is highlighted in the selected use case scenario.

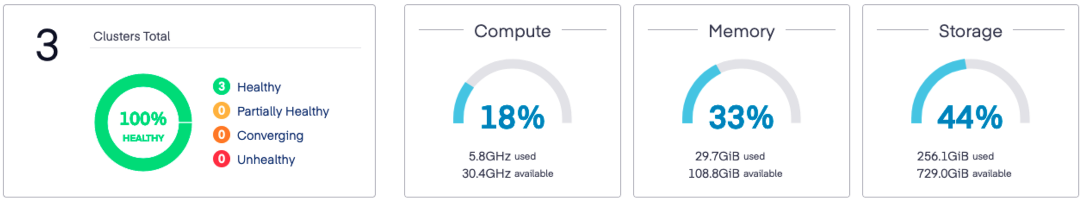

Figure 3: Cluster Capacity

3 Use Case: 4chan data retrieval

Because of its dynamic characteristics, ephemerality, anonymity and structure - 4chan is an interesting use case with many questions remaining still open. 1. *Is the data provides by the 4chan API*[10] *a close reflection of what the users experience?* 2. *Is there moderation in 4chan, what drives it and what impacts it has on the overal discussion?* 3. *At what level does the moderation happen? Is it thread only or does it go at post level?* To be able to answer these question a closer look to the data is needed. The magnifying glass effect, that is possible by continuously crawling and extracting data from browser end user like interactions, allows us to answer many of these questions.

In simple steps, we wrote two queries: one for crawling the 4chan live /pol board and extracting live threads, the second one for extracting structured content from the live threads. To highlight the simplicity of writing these queries, the crawling query is includes in listing 3

Listing 3: OXPath query for crawling 4chan /pol board

```
doc("http://boards.4chan.org/pol/catalog")
  //div[@class="thread"]:<links>[
     ./a:<link=qualify-url(@href)>
  ]
```

We run the data campaign for over 6 months at different capacity (and hence with different zoom-in effects). At it's highest capacity, there were 14 nodes, spread across 3 clusters running at any point in time on average 100 crawling or extraction instances per minute. The storage subsystem was made of a MongoDB cluster composed of one primary replica-set, three secondary and one arbiter. This meant that a total of about 30GHz of computing power and about 110GBit of volatile memory was primary dedicated at the data extraction campaign 3.
As a result, we have collected over 3M Threads, Over 31M Posts and more than 110GB of semi-structured data. The granularity of the scans is at second for large amounts of time. The partial[11] structure of the data collected is represented in diagram 4

The resulting data provides a picture which is not crystallised in time, but on the contrary depicts the evolution of the discussion as it takes place. Such an upgrade on previous data collection strategies provides scholars with the ability to study for 4chan under a new, till now unavailable "magnifying glass," capable of showing the temporal dynamic of the discussion as it takes place.

[10] For example, the 4chan API provides all live threads under the following endpoint: https://a.4cdn.org/po/threads.json

[11] The full structure of the data is far too large to be visualised in this article

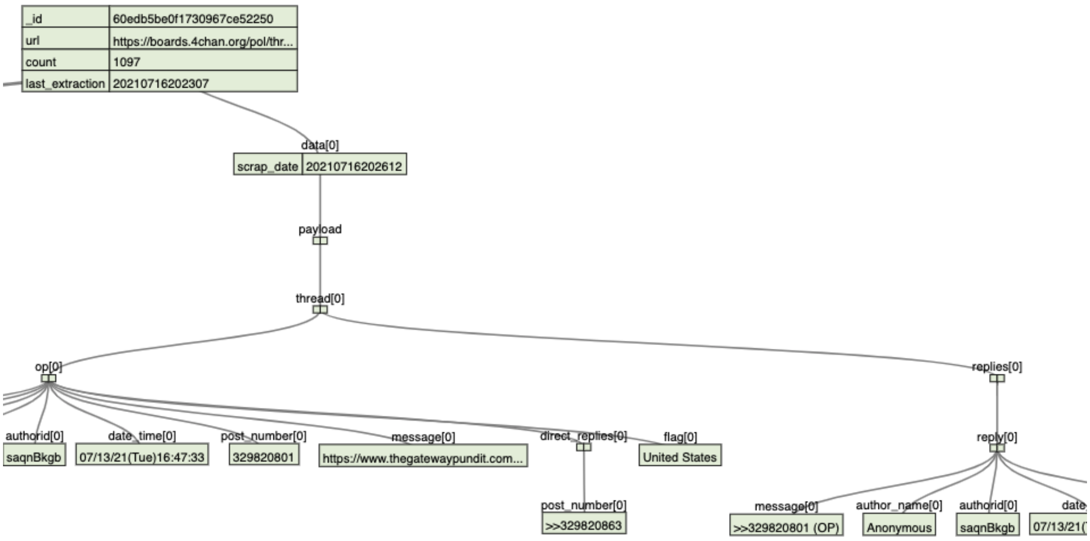

Figure 4: Partial 4chan data structure

3.1 Dataset availability

A full extraction of the dataset, covering a three month period from 1st of April 2021 to 1st of July 20201, has been published and made available to researches as detailed in [18]

4 Summary of findings

We believe to have provided researchers a powerful alternative to combine commodity hardware into large scale web scarping campaigns thanks do a distributed system we designed and implemented for this purpose. This system has characteristics that will allow for ease of use and adaptability to many scenarios. We showed the web data extraction framework in action for live monitoring and archiving of discussions from the 4chan online community. The main motivation behind our work is the study of the temporal evolution of 4chan threads, which would be impossible, given our on modest hardware, by existing crawlers. As of July 2021, our cyclic data collection operation has located and saved over 30 millions of unique posts. We made a portion of the collected dataset, and the cyclic updates, available to researchers. Thanks to the deployment of an advanced version of OxPath, we were able to track the evolution of the /pol/ bilboard with higher level of precision, which in turn revealed several unexpected results, e.g., in terms of actual moderation of the threads by 4chan janitors. Our current analytic work is focused on understanding the nature and dynamics of post deletion.

Previous works also developed ad-how crawlers but their architecture prevented them from discovering the possible (and intermittent) gaps between what is posted by users and what becomes available on the billboard.

References

[1] S. Avasarala. *Selenium WebDriver practical guide*. Packt Publishing Ltd, 2014.

[2] M. K. Bergman. White paper: The deep web: Surfacing hidden value. *The Journal of Electronic Publishing*, 7(1), Aug. 2001.

[3] T. I. M. BERNERS-LEE, J. HENDLER, and O. R. A. LASSILA. THE SEMANTIC WEB. *Scientific American*, 284(5):34–43, 2001.

[4] M. S. Bernstein, A. Monroy-Hernández, D. Harry, P. André, K. Panovich, and G. G. Vargas. 4chan and /b/: An analysis of anonymity and ephemerality in a large online community. In L. A. Adamic, R. A. Baeza-Yates, and S. Counts, editors, *Proceedings of the Fifth International Conference on Weblogs and Social Media*. The AAAI Press, 2011.

[5] S. A. Catanese, P. De Meo, E. Ferrara, G. Fiumara, and A. Provetti. Crawling facebook for social network analysis purposes. In *Proceedings of the international conference on web intelligence, mining and semantics*, pages 1–8, 2011.

[6] J. Clement. Facebook users worldwide 2019, Jan 2020.

[7] R. Diouf, E. N. Sarr, O. Sall, B. Birregah, M. Bousso, and S. N. Mbaye. Web Scraping: State-of-the-Art and Areas of Application. In *2019 IEEE International Conference on Big Data (Big Data)*, 2019 IEEE International Conference on Big Data (Big Data), pages 6040–6042, Los Angeles, United States, 12 2019. IEEE.

[8] T. Furche, G. Gottlob, G. Grasso, C. Schallhart, and A. Sellers. Oxpath: A language for scalable data extraction, automation, and crawling on the deep web. *The VLDB Journal*, 22(1):47–72, 2013.

[9] T. Furche, G. Gottlob, G. Grasso, C. Schallhart, and A. J. Sellers. Oxpath: A language for scalable data extraction, automation, and crawling on the deep web. *VLDB J.*, 22(1):47–72, 2013.

[10] D. Glez-Peña, A. Lourenço, H. López-Fernández, M. Reboiro-Jato, and F. Fdez-Riverola. Web scraping technologies in an API world. *Briefings in Bioinformatics*, 15(5):788–797, 04 2013.

[11] G. Grasso, T. Furche, and C. Schallhart. Effective web scraping with oxpath. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 23–26, 2013.

[12] G. E. Hine, J. Onaolapo, E. D. Cristofaro, N. Kourtellis, I. Leontiadis, R. Samaras, G. Stringhini, and J. Blackburn. Kek, cucks, and god emperor trump: A measurement study of 4chan's politically incorrect forum and its effects on the web. In *Proceedings of the Eleventh International Conference on Web and Social Media, ICWSM 2017, Montréal, Québec, Canada, May 15-18, 2017.*, pages 92–101. AAAI Press, 2017.

[13] A. Hogan. The semantic web: Two decades on. *Semantic Web*, 11:169–185, 2020.

[14] L. Knuttila. User unknown: 4chan, anonymity and contingency. *First Monday*, 16(10), 2011.

[15] C. Olston, M. Najork, et al. Web crawling. *Foundations and Trends® in Information Retrieval*, 4(3):175–246, 2010.

[16] J. Pfeffer, K. Mayer, and F. Morstatter. Tampering with twitter's sample api. *EPJ Data Science*, 7(1):50, 2018.

[17] Pfeffer, Jürgen, Mayer, Katja, and Morstatter, Fred. Tampering with twitter´s sample api. *EPJ Data Sci.*, 7(1):50, 2018.

[18] Y. Prifti. 4chan data scraped form /pol board as a time evolution of threads and pots, July 2021.

[19] S. Silva and M. Kenney. Algorithms, platforms, and ethnic bias. *Communications of the ACM*, 62(11):37–39, 2019.

[20] D. S. Sirisuriya et al. A comparative study on web scraping. *Empty*, 2015.