RSS Podcast Feed Inefficiency: Apple, Amazon, Spotify et al.

Damon Hart-Davis
d.hart-davis@surrey.ac.uk
Centre for Environment and Sustainability, University of Surrey
Guildford, Surrey, UK

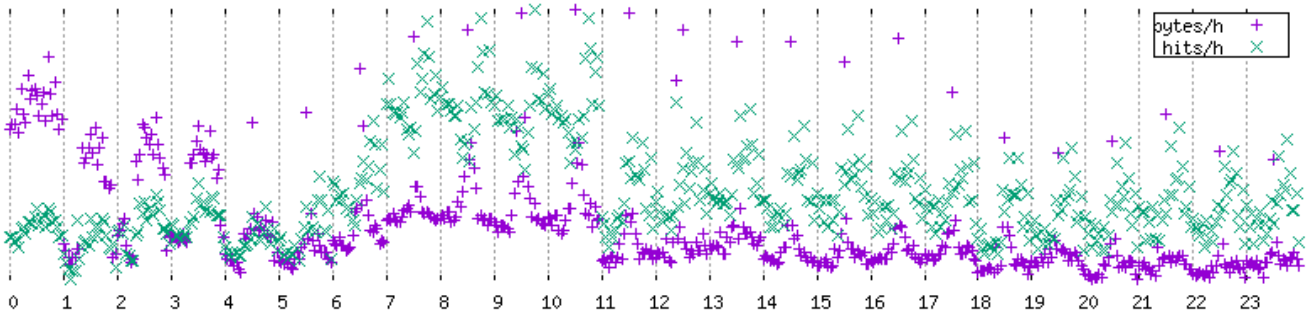

Figure 1: Visualisation of RSS feed load over time by hour of day UTC.

ABSTRACT

Centralised social media systems are somewhat out of favour in 2024 for reasons from fake news and privacy to the actions of billionaire owners. Federated and more decentralised systems such as Mastodon and the Fediverse, plain old email, and RSS feeds including podcasts, are cool again. With much of the workings being out of sight for ordinary users, and in a system designed before intermittent renewable power generation was a thing, podcasting and RSS in particular are unnecessarily wasting an appreciable portion of their bandwidth and CPU time, and adding to climate change. Indeed, key players are consuming orders of magnitude more resources across their systems and others than necessary. There are already several simple and widely-used technical mechanisms that could help, but many participants are ignoring them. This paper suggests sustainability improvements for elements of the ecosystem that should be largely transparent to end users, including Cache-Control, conditional GET and skipHours, saving likely much more than 100kWh per day of electricity globally.

1 INTRODUCTION

In a world fighting climate change, one of our most powerful weapons is efficiency. There is no cheaper nor lower-carbon kWh of electricity for a participating client or server than the kWh not used at all. RSS feeds [7] are regaining popularity as a robust decentralised distribution for blogs and podcasts [8], with a comparatively ancient (in Internet timescales) and stable protocol, but with a vibrant ecosystem and useful extensions regularly added and used. Some of the biggest central participants such as directories and aggregators are consuming bandwidth and CPU resources (their own, and that of creators) at two or three orders of magnitude more than is useful. This at the very least is money wasted and unnecessary carbon pollution. A large part of the author's life has been invested in developing low-power electronics and software, optimising resource use (CPU, time, heat, etc), and the author runs the Earth.Org.UK (EOU) Web, DNS, email, and other servers off-grid on a tiny power budget, and thus finds this to be unnecessary and outrageous profligacy. A huge amount of electricity, not nearly all zero-carbon, is being consumed in data centres that host RSS servers and clients, such as in Ireland where those data centres use more than urban homes [1]. As of 2019 data centres accounted for approximately 3% of the electricity consumption and almost 4% greenhouse gas emissions globally [9] and in the UK alone will likely grow six-fold over the next decade [6]; the waste described here is one small piece of the puzzle to fix. A few days of engineering time across the top perpetrators would easily recover much of this, but they have indicated to the author that they see no problem in what they do and are unwilling to change their behaviour. They are ignoring multiple signals both RSS-specific and so generic that all mainline Web browsers already observe them. Indeed the Internet would be unworkably slow without browsers doing so. Better news is that many of the niche and end-user client developers are very willing to engage and improve, and so almost vanish from usage logs, supporting the notion that this is easily fixable. This paper both describes what is needed by existing participants to reduce the waste, and some defensive measures, stateless and stateful, that feed servers such as the author's can take to mitigate bad actor load.

2 METHOD

This paper documents the continuing interaction of one hosted podcast feed (and a lighter-weight variant) with a roughly monthly update cadence and a very small number of listeners, the statistics of its use, deployed tactics to cut down the unreasonable load imposed by clients, and a developed set of priorities and suggestions for client implementors (and feed servers to some extent). The feed in question is the EOU (Earth.Org.UK) podcast feed [4], plus a

LOCO 2024 - submitted extended abstract, 3 December 2024, Glasgow, Scotland 2024.

very lightweight variant published to only one major aggregator (Spotify) as a sub-experiment.

3 RESULTS

An analysis of feed traffic by 'hits' (number of HTTP accesses) and 'bytes' (nominally HTTP response bytes, excluding some overheads) is presented. This is approximately week-long slices, exact boundaries determined by when the Web server logs happen to be rolled, shown normalised to per-day and percentage-of-full-site values.

The overall story is that the defensive techniques deployed have reduced 'bytes' per day by about a factor of three, though 'hits' have risen a little because of the bad response of many clients to being asked to slow down with an HTTP error response (429 or 503) where they instead poll more and faster.

4 DISCUSSION

It is evident that there is a lot of waste in podcast feed polling, driven by sloppy engineering and ignoring most or all of the already-available signals that could improve that. It is also evident from my corresponance with them that the big players do not care to fix things, with those such as Apple and Spotify having a very unequal power relationship with the content creators; the type of imbalance and resulting casual indifference to others' costs that the EU Digital Markets Act (DMA) seeks to fix. There is also a chicken-and-egg issue of why clients would improve their game in the face of badly-configured servers and vice versa when there may be no immediate gain with at least some counterparties. For creatives providing feeds and paying for bandwidth there can be a very quick payback from relatively simple configuration fixes on their side. Podnews enabled `gzip` compression and more than halved its feed bandwidth bill instantly [2, 3]. The EOU podcast feed avoided the problematic implmentation of `ETag` on Apache HTTP servers, leaning more heavily on `Last-Modified`, and started returning more 304 'Not Modified' lightweight empty-body responses to many clients such as search engines, as well as to good RSS feed clients. The EOU podcast feed also substantially reduced bandwidth with some defensive adjustments without the cooperation of the likes of Apple and Spotify.

4.1 Suggestions to implementors

In Hart-Davis [5] practical suggestions are made both to implementors of existing feed clients such as Apple iTunes and Spotify, and new RSS and podcast readers, and also to implementers and servers of feeds. The EOU feed behaviour was improved with some of the server recommendations.

Feed efficiency suggestions for aggregators and other RSS clients include, in roughly this priority order, to save bandwidth, CPU, money and the climate:

(1) use `Cache-Control max-age` HTTP headers for a "do not poll again before" time: savings of 10x or much more are likely if the feed server is set up well: an unnecessary feed poll avoided entirely is the cheapest kind!
(2) use a local cache and conditional `GET` (eg send `If-Modified-Since` and/or `ETag` HTTP headers): savings of 10x or more are likely
(3) allow compression of the feed that you pull down (set `Accept-Encoding` HTTP headers) with at least `gzip`: savings of 2x to 10x are likely
(4) avoid fetching the feed on `skipHours` (and/or `skipDays`) in an RSS feed: savings of 2x are plausible, and can be especially renewables/climate friendly
(5) use error responses 429 (Too Many Requests) and 503 (Service Unavailable) `Retry-After` header for a "do not poll again before" time (like `Cache-Control max-age` above) when present: do *not* retry immediately/faster/repeatedly!

Meanwhile, servers of podcast feeds should make sure that:

(1) at least gzip compression is available
(2) conditional fetches work properly so most Apache-2.4-hosted RSS feeds should turn off `ETag`s for example
(3) set a feed expiry to match when new episodes are published.

5 CONCLUSIONS

This paper contributes to the debate on efficiency and frugality in network use to help remedy climate change, and illustrates some practical automated measures that podcast RSS feed site owners can take against the worst-behaved clients to manage load. First, this paper characterises the apparent inability and unwillingness of some of the biggest and most important participants in the podcast space to conserve resources: theirs and the creators. They are ignoring most of the existing support and basic good practice already available. For a podcast with approximately monthly updates it is seen that well over 99% of the resources consumed are unnecessary. Avoiding that would save money and climate-damaging carbon emissions. An estimate of the wasted energy an emissions is made. Second, this shows that creators serving RSS podcast feeds can often easily help themselves, eg simple configuration changes can immediately cut load [3]. Also creators can if they wish add defences in their configuration to reduce load from bad actors, though doing so requires some sophistication. These adjustments can be especially climate friendly, rejecting wasteful demands when servicing them would be carbon intensive. Third, future work to further reduce network demands, and help all parties trim resource consumption have been mooted, again neither requiring new technology nor standards to be invented.

REFERENCES

[1] 2024. National Energy and Climate Plan (NECP) 2021-2030. https://www.gov.ie/en/publication/a856a-national-energy-and-climate-plan-necp-2021-2030/
[2] James Cridland. [n. d.]. *Podnews: Our RSS Stats.* https://podnews.net/about/rss-stats
[3] James Cridland. 2024. Podnews ... A saving bandwidth special! https://podnews.net/update/podlp-cloud-phone
[4] Damon Hart-Davis. [n. d.]. Earth Notes Podcast. https://www.earth.org.uk/rss/podcast.rss
[5] Damon Hart-Davis. 2024. RSS Podcast Feed Inefficiency. https://www.earth.org.uk/RSS-efficiency.html
[6] John Pettigrew. 2024. Transforming the supergrid of the 1950s to a network built on an electrified future for generations to come. https://www.linkedin.com/pulse/transforming-supergrid-1950s-network-built-future-come-john-pettigrew-jpbcf/
[7] RSS Advisory Board. 2009. RSS 2.0 Specification. https://www.rssboard.org/rss-specification
[8] Matthew Sharpe. 2020. A review of metadata fields associated with podcast RSS feeds. https://doi.org/10.48550/ARXIV.2009.12298
[9] Xiaolei Yuan, Yumin Liang, Xinyi Hu, Yizhe Xu, Yongbao Chen, and Risto Kosonen. 2023. Waste heat recoveries in data centers: A review. *Renewable and Sustainable Energy Reviews* 188 (12 2023), 113777. https://doi.org/10.1016/j.rser.2023.113777