A Feasibility Study of Cache in Smart Edge Router for Web-Access Accelerator

Krittin Intharawijitr*
Autonomous Networking Research and Innovation Department
Rakuten Mobile, Inc., Tokyo, Japan
ar-intharawi.krittin@rakuten.com

Paul Harvey
Autonomous Networking Research and Innovation Department
Rakuten Mobile, Inc., Tokyo, Japan
paul.harvey@rakuten.com

Pierre Imai
Autonomous Networking Research and Innovation Department
Rakuten Mobile, Inc., Tokyo, Japan
pierre.imai@rakuten.com

***Abstract*—Regardless of the setting, edge computing has drawn much attention from both the academic and industrial communities. For edge computing, content delivery networks are both a concrete and production deployable use case. While viable at the WAN or telco edge scale, it is unclear if this extends to others, such as in home WiFi routers, as has been assumed by some.**

In this work-in-progress, we present an initial study on the viability of using smart edge WiFi routers as a caching location. We describe the simulator we created to test this, as well as the analysis of the results obtained. We use 1 day of e-commerce web log traffic from a public data set, as well as a sampled subset of our own site - part of an ecosystem of over 111 million users. We show that in the best case scenario, smart edge routers are inappropriate for e-commerce web caching.

***Index Terms*—edge computing, CDN, simulation, study, smart edge router**

I. Introduction

Over 60% of the traffic on the internet is streaming video content. To reduce latency and provide high quality of service to users without linearly increasing network capacity, this content if often provided at geographically distributed locations around the world. This reduces the latency of content access, as well as the load on the backend (origin) servers. Such *edge* infrastructure and service is know as content delivery networks (CDNs).

CDNs are one of the initial use cases being targeted by telecommunication operators who are deploying edge computing infrastructure and services within their networks. Unlike cloud computing, which consists of relatively centralised resource-rich data centers, telco edges are more distributed and (comparatively) resource-constrained. A core concept of edge computing is that the closer to the user the better,

However, the telco edge does not always represent the closest infrastructure point to the user. Base stations are often cited as a potential site, however, operators do not agree, preferring simpler base stations to reduce maintenance [1], [2]. Going further, internet and/or telco providers often offer in-home, -school, -office, -school, or -transport hardware such as set-top boxes to access media or WiFi routers to share internet access. we define such devices as smart edge routers (SERs): a WiFi router with a CPU, memory and storage and provides Internet access via a mobile network, Figure 1. These devices represent a much closer end-point to the user, however, the validity of using such devices as caching locations is unclear.

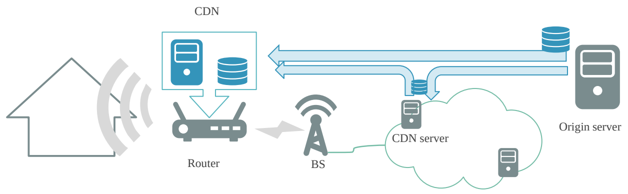


Fig. 1: Smart edge router with CDN software

In this work-in-progress, we investigate the question of what (if any) is the appropriate point between an end-user and origin server to provide caching functionality. To the best of our knowledge, this is the first work to pragmatically attempt to answer the question of *are SERs viable for an e-commerce CDN*.

To answer this question, we created an event-based simulator to explore potential impact of having a CDN cache at different points along the user-origin spectrum, as well as carried out an initial investigation based on both a public data set and a sampled subset of our own web traffic to our e-commerce platform - part of an ecosystem of over 111 million members. We also describe a SER platform that we have created to perform real world testing in a future field trail.

The paper is arranged as follows: Section II describes the background for this work, Section III discusses the motivation to push caches closer to users. Section IV describes the simulator used in this study, then Section V discusses the results. Section VI introduces our real platform design, and lastly, the future work and conclusions are in Section VIII.

II. Background

Previous work has explored efficient ways to place content [3], [4] or route traffic [5], [6] for a CDN in an edge network, however, these make assumptions that are not necessarily realistic. For example, Weng Hu, et al. [7] explore the content placement problem using simulation of different wifi hotspots and mobile phone users, which subsequently extended [8] to video streaming applications in a peer-to-peer CDN [9]. In this work, it is assumed that all home "hotspots" can be accessed by all users. However, such systems offer

*Krittin Intharawijitr was with Rakuten Mobile, Inc. until July, 2020.

limited coverage due to being placed inside homes which absorb signals, are often slow as the owner's connection is prioritised, are operator-specific (if even present), and are subject to security and privacy concerns limiting user adoption.

Caching locations in the cloud [10], dedicated CDN providers [11], and telco edge [12] have already been shown as effective locations to host CDN caches and are currently in operation. When considering alternate locations to cache content, base stations or cell towers are often cited as locations [2], [13], [14], however, as operators and vendors seek to simplify the hardware in these locations through virtualisation [15] to reduce the maintenance cost, this is increasingly less likely.

CDNSim [16] is a caching simulation platform for CDNs that replicates WANs down the to transport layer. While our goals are in the same area, CDNSim focus on relationship of client, CDN, and origin server mainly for wired infrastructure. In contrast, our goal is to modify the locations of cache nodes in a wireless network.

Industry has also been investigating the topic of smart edge routers:

a) CacheBox[1]*:* is a product for a network administrators which supports cache and networking functions for schools or small companies. It must be mounted in a server rack as a proxy server, so it is not friendly to all users. This is one of the intended use cases of our study.

b) NightShift[2]*:* was a collaboration between an internet service provider (ISP) and Netflix to retrofit home WiFi routers with storage capability to become a cache. The effort was terminated in 2017.

c) Smart Home Gateway: (i.e Xiaomi Mi Smart Home) is a router that connects and manages all smart devices in a house. Its operation is a router and Wi-Fi gateway with intelligent features of a smart home. While this is a platform that we target, it neither promotes or requires caching.

d) Mobile Routers: are Wi-Fi hotspots from Mobile Network Operator which connects to the internet via the mobile network. This product type is common from different operators: SoftBank Air[3], AU Speed Wi-Fi HOME[4], but does not support or promote caching nor have the capability to.

In summary, academic efforts have focused on optimising placement and caching strategies for smart edge environments based on unrealistic assumptions in either the target location, operational environment, or simulation granularity. Considering the industrial community, there have been efforts to use the SER for caching, however, none have been truly successful. One of the main reasons is that there must be a partnership between the CDN operator and the platform provider to ensure that data, content, and platform are available.

III. Study Motivation

We now describe the different scenarios in which a SER would be used as well as the specific questions that we want to answer via analysis in Section III-B.

A. Scenarios

We consider three usage scenarios:

a) Small Scale: A group of 3–5 users in the same room or home environment. Here, the distance to an origin server is greatest, however, there is likely to be regularity in the content accessed.

b) Medium Scale: A group of 10–200 users in the same building or area, such as a cafe, restaurant, school, or small to medium office. The content accessed is more diverse and short-lived than the small case given the greater number of users and the potentially transient relationship between person and geographical location.

c) Large Scale: A group of more than 500 users in the same large area or building, such as public parks, stadium, or amusement parks. The diversity and frequency of the content accessed is more so than the small or medium case. The smart edge router is least suited to this case given the large number of users and wide coverage area.

As noted above, there is much work on different aspects of CDN caching. The goal of this paper is to address fundamental viability, not potential optimisations.

B. Viability Questions

The main question we intend to answer is *How viable is a SER as a cache for real data?*. Given the lack of existing artefacts in this domain, we will initially use simulation to answer the question, see Section IV.

We refine the above question as follows:

- What is the appropriate level to place a SER cache?
- How much traffic can a SER cache reduce in the network?
- How much can a SER cache decrease the latency to access content?
- How much storage does a SER cache require?

In our study we focus on e-commerce data, however, these questions are generic enough toother content types.

IV. Simulation

As no smart edge router infrastructure exists to test upon, and before deploying our own, we created a simulator. Our goal is **not** to create an emulator or protocol simulator, but instead to have a tool that can serve as a preliminary indicator as to the appropriateness of using a smart edge router for a particular service before committing the time and effort to a deployment or proof of concept. In this section we focus on describing the simulator, with experimental results discussed in Section V.

A. Design & Implementation

The simulation process works in two parts: topology creation and event-based simulation.

[1]www.appliansys.com/cachebox/
[2]https://www.ocalasatellite.com/night-shift
[3]https://www.softbank.jp/ybb/air/
[4]https://www.au.com/english/mobile/product/data/hws33/

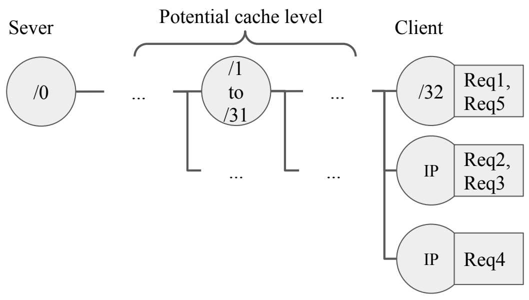


Fig. 2: A hierarchical group of IP address

1) Topology Creation: Before simulation, it is necessary to create a network topology which represents the location(s) of a service, users, and the caching locations. We create this topology based on the information contained in a web access log for a particular service. This log can be for an origin server, or a CDN. The log is then parsed to extract the source IP address, URL of the content, content size, status code, and accessing platform - e.g. android, windows, mac, etc.

This data is then compressed and filtered to remove noise and reduce storage and runtime space consumption. Namely, we encrypted the URL to normalise the length, removed requests without status 200 (ok) HTTP responses, and removed requests from bots which will not access content multiple times by design.

Next, the IP addresses are grouped hierarchically based on their prefix, where the top of the graph represents the origin server and the leaf nodes the clients, Figure 2. We can now specify different levels using different subnet masks. For example, /0 represents the root level as it contains all addresses in the subnet, whereas /32 represents the lowest level as it contains only a single IP address. Here /32 is a client, not a cache. Using this approach we specify the levels (or slashes) at which we choose to locate CDN caches in our system during the simulation phase.

Our assumption is that nodes within the same subnet will exhibit some geographical locality, and the smaller the number of participants in the group, the higher the locality. Of course this is not a perfect solution, but accurate geographical data is hard to obtain in the general case.

All of this is done automatically and is independent of the service represented by the weblog.

2) Event-Based Simulation: Once the topology has been created, the user can now place the CDN caches. The simulator allows the user to configure at which levels a CDN cache can be placed, the size of the cache available at each level, the bandwidth available between each level of the hierarchy, the content eviction algorithm used, and the transmission delay between each level. Content pre-fetch is not supported. All of these parameters can be customised per simulation element or pairing.

Metrics which can be collected are *cache hit*, *cache miss*, *cache occupancy*, *cache content eviction*, *accesses per sub-group*, and *latency to access content*.

	Harvard	HTTP	HTTPS
Total number of requests	1,885,865	6,346,384	4,442,230
Total number of IP address	54,170	959,055	439,476
Total size of access (GB)	21	92	136

TABLE I: Dataset Properties

The simulation itself is a “replay” of a weblog against the cache-enabled topology, where each entry in the weblog is a request to access the specified content. In the simulation, when the node containing a cache (cache node) receives a request for content it inspects its cache. If the content is hit, it will stop forwarding the request and reply with content. Otherwise, the request will be forwarded to a next node in the upper level. Eventually, if there is no cached content in any node, the request will be sent to an origin server which will reply. While we tested with real weblogs, there is no limitation in using developer-defined weblogs.

The response time is the time taken to receive content from a cache or origin node after a request is made from a /32 node. We calculate the response time of each access as follows:

- Response time = delay of sending a request until hit + delay of replying content to a source
- Delay of sending a request on one link = Size of packet / Bandwidth
- Size of packet = size of header + size of content

V. Experiment, Results, and Discussion

This section explains and discusses the simulation result to answer each question in Section III-B.

A. Experimental Setup

In this analysis, we used logging from two online e-commerce platforms for one day. One is a public data set from Harvard dataverse [17] which contains only HTTP records. The other is a sampled subset from our online e-commerce platform split into HTTP and HTTPS requests.

Table I describes properties for each dataset. In this study, we focused on the logs from our online platform to show the performance from the real data and used Harvard both to show the generality of our simulator as well as a reference.

Also, in our simulations we assume the best case scenario to understand the initial viability before investing time to refine the simulation setups. For example, there are no lost packets and re transmissions.

B. What is the Appropriate (Subnet) Level for the SER?

In considering the small and medium use cases, if the members of the subnet group are too few there will not be enough repeat access to content to justify the cache. Figure 3 shows the distribution of subnet member sizes across the /16 (existing CDN cache), /20 and /24 subnets respectively. These data sets and subnet levels are long tail distributions with large numbers of groups with few members. Given this, we chose the /24 level for the new cache level. This subnet accommodates up to 256 users per group and fits our use cases best.

Cache level		Server	/8	/16	/20	/24	Client
HARVARD	Seen	28%	47%	64%	80%	100%	100%
	cached	0%	18%	16%	17%	20%	0%
HTTP	Seen	55%	74%	81%	86%	100%	100%
	cached	0%	19%	12%	5%	7%	0%
HTTPS	Seen	48%	57%	59%	61%	100%	100%
	cached	0%	9%	2%	2%	39%	0%

TABLE II: Percentage of Traffic Observed and Reduced per Subnet Level

The IP addresses from web access logs are not so clustered; even if some groups have many of members (see maximum in Figure 3). The result also shows that not many clients in the same subnet access the same server. Although we use a smaller subnet such as /16 and /20, the median (50% of data) of all data sets in Figure 3 is still lower than 10 hosts per group. In conclusion, most request were sent from different areas, with a few areas having intensive server interaction.

C. How Much Network Traffic can the SER Reduce?

Total traffic is the sum of all content accesses from the web logs, and traffic reduction is the total amount of content which does not reach the origin servers due to a cache hit. In the simulation, the first access of a content is miss. The following access of the same content will be hit, preventing the request being forwarded to the next node, thus reducing network traffic.

Table II shows the traffic reductions for caches with unlimited storage at /8, /16, /20 and /24 levels. Unlimited storage was chosen to identify the best case scenario, and means that eviction policies do not interfere with the result.

100% of all requests from /32 clients were sent to nodes at /24. The caching layer at the /24 level performs reasonably well given the small number of members per group and fluctuations across the different data sets. From an percentage data saving perspective, this argues for the use of a smart edge router.

D. How Much Latency Decrease can the SER Provide?

In the simulation, we assume the same bandwidth is available between subnet levels and the size of each content request is found in the web log. The header size of the packet is set to 8 Bytes as minimum of UDP header.

We present both the overall aggregated effect that the smart edge router has on response time as well as a per request distribution. Given space limitations we report on the Rakuten data set in both cases. Using the least frequently used eviction policy (LFU), Figure 4 shows response time for HTTP and HTTPS traffic in four different cases: infinite cache (cache at all levels), cache at CDN level (/16), cache at CDN and edge router level (/16 and /24) and no cache.

In all cases, caching has a positive effect on the overall response time, with higher cache capacity leading to better response times. However, given the long tail distribution of users per subnet group (Figure 3), it is necessary to look closer at the response time distributions. Figure 5 shows the distribution of response time for HTTP and HTTPS traffic when the cache size is fixed at 1GB. When considering the mean, adding an EDGE (/24) cache to the existing CDN (/16) cache did not show a significant improvement: 0.05 ms for HTTP and 0.3 ms for HTTPS. Given the context of e-commerce websites, these improvements are unlikely to be perceived by a user.

E. How Much Storage is Necessary for a SER

The size of a cache determines the maximum amount of data to be stored and impacts performance, as seen in Figure 4.

Figure 6 shows the distribution of cache occupancy for edge router (/24) nodes when the cache capacity was infinite. The result shows that more than 75% of all nodes had cache occupancy lower than 1 MB. The first deviation of results were 35.46 MB for HTTP and 79.37 MB for HTTPs. Therefore, a cache capacity of 100MB would suffice. This is a relatively small amount of RAM - battery powered mobile phones contains GBs of RAM - and would account for at least the first standard deviation of the results.

F. What is the Effect of Cache Replacement Policies on the SER

Here, we demonstrate that our simulation platform can support different content eviction algorithms. We compare LRU and LFU in Figure 7 by evaluating the total response time of each approach using 1 MB and 1 GB cache capacity.

For the 1 GB case there was no significant difference as the cache node capacity was sufficient to hold all content. For the smaller 1 MB case, not all content could be cached so a difference case be seen. LFU preforms better than LRU as our dataset was from an e-commerce site where a page's content is not frequently changing, popularity or frequency of access is a more relevant metric in this case. Practically, the difference is negligible.

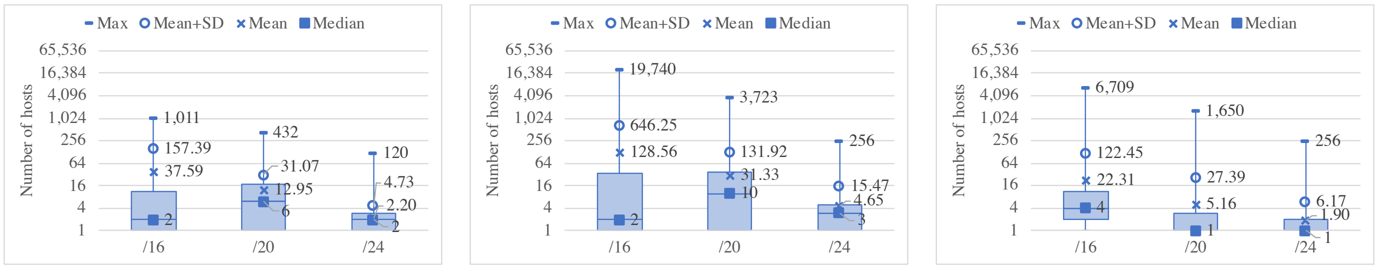
G. Summary

Assuming an oversimplified best case scenario for our simulation setup, we conclude that for e-commerce like content, the smart edge router is not a viable option in terms of the benefits that it provides in user response times or network load reduction. This raises questions about the assumptions that have been made in other works that state the opposite.

However, we believe that our relatively straight forward simulation tool has shown its worth and can easily be re-purposed to explore other potential caching scenarios.

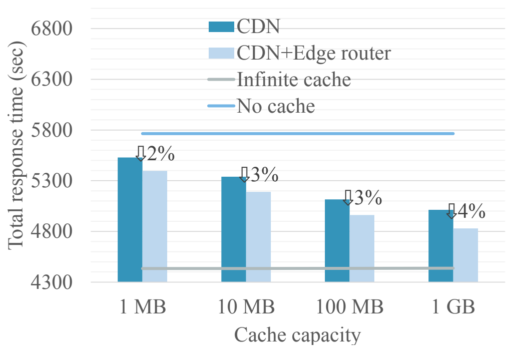
VI. Smart Edge Router Platform

Given the limitations of simulation, we have also created a real edge router platform to conduct field trails with real users in the near future. As well as confirming the simulation results, we also seek to investigate alternate approaches to smart edge router caching which prioritise the satisfaction of the *individual* customer, as opposed to the general satisfaction of the customer or the network. Finally, we will investigate the claims of some other works in the field.

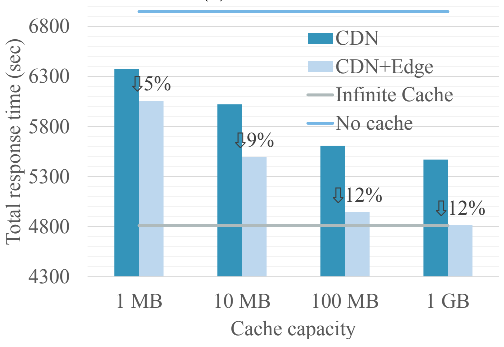


(a) Harvard (b) HTTP (c) HTTPS

Fig. 3: Distribution of host per subnet for each dataset



(a) HTTP



(b) HTTPS

Fig. 4: Total response time (sec) with cache capacity when cache policy is LFU

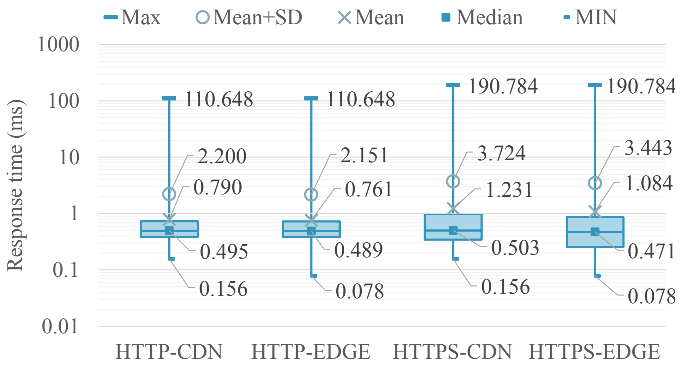


Fig. 5: Distribution of response time (ms) of all requests when cache capacity is 1 GB and policy is LFU

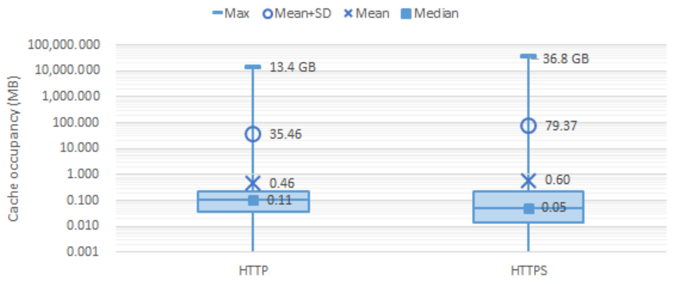


Fig. 6: Distribution of cache occupancy at edge routers when cache capacity is Infinite

A. Platform design

We us a small Linux based PC[5] with 4GB RAM, 1 GHz quad core processor, 60GB storage, and an LTE module to connect directly to our mobile network enabling the device to be a stand-alone wifi router. We run Varnish-cache[6] within a docker container to configure caching rules and policies to accelerate access to specific web content.

B. System Design

To delivery a smart edge router for caching, we propose a prototype system that autonomously creates a custom cache scheme for individual routers based on logging data. Unlike other works in Section II, our approach is targeted towards the owners of the smart edge router. That said, it does not exclude sharing by design. The system will:

- Collect a log of home activities and storage status
- Create a model or a policy of how to cache contents at the home router
- Execute caching procedure under the model/policy

Figure 8 shows our architecture. The smart edge router itself is responsible for logging, reporting, and serving content. The controller server is resource rich and responsible for analysing and instructing which content should be cached in the router by creating ans transmitting caching policies.

[5]https://www.pcengines.ch/
[6]https://www.varnish-software.com/

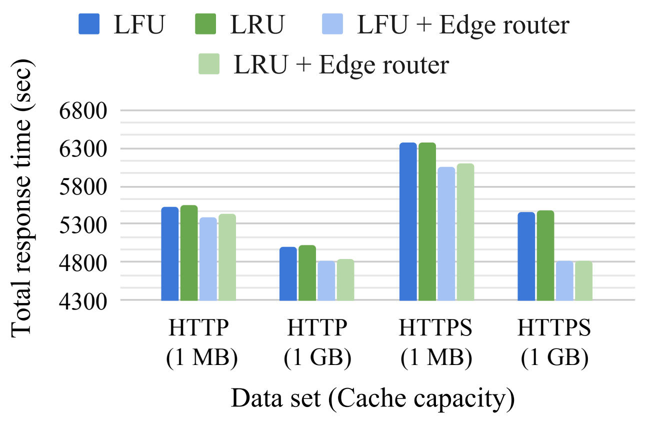


Fig. 7: Total response time (sec) when cache policy is LFU and LRU

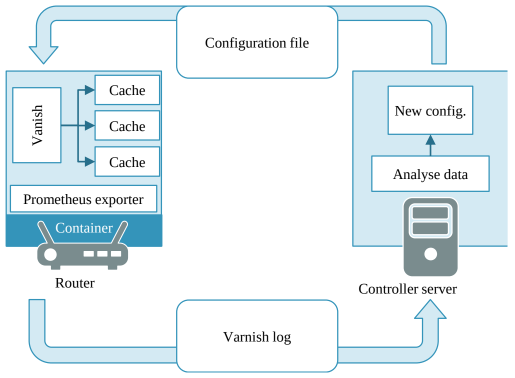


Fig. 8: Autonomous customization caching

VII. Future Work

To increase the veracity of the simulator, as well as have a more robust real world view, we intend to conduct a progressive and ongoing field trail using the platform discussed in Section VI. In addition to e-commerce, we plan to experiment with other services, such as streaming media, which is also part of our ecosystem.

As communication networks increasingly become the common denominator of modern life, so too does the pressure placed upon them. Equally, the decisions of where to place content, route traffic, or schedule CDN services on the edge must be made in software. As well as answering the question of where to host CDN services along the end-user origin spectrum, we will also investigate the topic of *autonomous* operation of the edge-router CDN service [18].

VIII. Conclusion

This work in progress answers the question of are smart edge routers appropriate platforms for CDN caches: in the case of e-commerce sites they are not. This was investigated by the creation of a high-level event-based CDN simulation tool that can automatically parse web access logs, generate a network topology, and replay user content requests in different scenarios. Using this tool on a public data set, as well as a sampled subset of our own e-commerce site logs, we have analysed the results to conclude that smart-edge routers are not appropriate for this use case, challenging the assumptions of other works.

We now plan to verify these results with field trials using our custom created platform, as well as investigate autonomous operation of CDN.

References

[1] M. Yang, Y. Li, D. Jin, L. Su, S. Ma, and L. Zeng, "OpenRAN: a software-defined ran architecture via virtualization," in *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM*, ser. SIGCOMM '13. New York, NY, USA: ACM, 2013, pp. 549–550.

[2] J. Yao, T. Han, and N. Ansari, "On mobile edge caching," *IEEE Commun. Surv. Tut.*, vol. 21, no. 3, pp. 2525–2553, 2019.

[3] J. Helt, G. Feng, S. Seshan, and V. Sekar, "Sandpaper: Mitigating performance interference in cdn edge proxies," in *Proc. 4th ACM/IEEE Symp. Edge Comput.*, ser. SEC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 30–46. [Online]. Available: https://doi.org/10.1145/3318216.3363313

[4] Z. Chang, L. Lei, Z. Zhou, S. Mao, and T. Ristaniemi, "Learn to cache: Machine learning for network edge caching in the big data era," *IEEE Wireless Commun.*, vol. 25, no. 3, pp. 28–35, 2018.

[5] F. Ahmed, Z. Shafiq, A. Khakpour, and A. X. Liu, "Optimizing internet transit routing for content delivery networks," in *2016 IEEE 24th Int. Conf. Netw. Protocols (ICNP)*, 2016, pp. 1–10.

[6] H. Hu, Y. Wen, T.-S. Chua, J. Huang, W. Zhu, and X. Li, "Joint content replication and request routing for social video distribution over cloud cdn: A community clustering method," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 7, pp. 1320–1333, 2015.

[7] W. Hu, Z. Wang, M. Ma, and L. Sun, "Edge video cdn: A wi-fi content hotspot solution," *J. Comput. Sci. and Technol.*, vol. 31, no. 6, pp. 1072–1086, 2016.

[8] G. Ma, Z. Wang, M. Zhang, J. Ye, M. Chen, and W. Zhu, "Understanding performance of edge content caching for mobile video streaming," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 5, pp. 1076–1089, 2017.

[9] A. Clouder. (2018, June) Content delivery acceleration and cost reduction with p2p cdn (pcdn). [Online]. Available: https://www.alibabacloud.com/blog/content-delivery-acceleration-and-cost-reduction-with-p2p-cdn-pcdn_593733

[10] Amazon, "AWS CDN," 2020. [Online]. Available: https://aws.amazon.com/cloudfront/#: :text=Amazon's CDN offers a simple,HTTPS Requests with Amazon CloudFront.

[11] Akamai, "Akamai CDN Hosting," 2020. [Online]. Available: https://www.akamai.com/uk/en/resources/cdn-hosting.jsp#: :text=CDN hosting with Akamai,is dedicated to their network.

[12] Qwilt, "Verizon CDN Hosting," 2020. [Online]. Available: https://qwilt.com/verizon-chooses-qwilt/

[13] W. Wang, R. Lan, J. Gu, A. Huang, H. Shan, and Z. Zhang, "Edge caching at base stations with device-to-device offloading," *IEEE Access*, vol. 5, pp. 6399–6410, 2017.

[14] J. Liu, B. Bai, J. Zhang, and K. B. Letaief, "Cache placement in fog-rans: From centralized to distributed algorithms," *IEEE Trans. Wireless Commun.*, vol. 16, no. 11, pp. 7039–7051, 2017.

[15] G. Karagiannis, A. Jamakovic, A. Edmonds, C. Parada, T. Metsch, D. Pichon, M. Corici, S. Ruffino, A. Gomes, P. S. Crosta *et al.*, "Mobile cloud networking: Virtualisation of cellular networks," in *2014 21st Int. Conf. Telecommun. (ICT)*. IEEE, 2014, pp. 410–415.

[16] K. Stamos, G. Pallis, A. Vakali, D. Katsaros, A. Sidiropoulos, and Y. Manolopoulos, "Cdnsim: A simulation tool for content distribution networks," *ACM Trans. Model. Comput. Simul.*, vol. 20, 01 2010.

[17] F. Zaker, "Online Shopping Store - Web Server Logs," 2019. [Online]. Available: https://doi.org/10.7910/DVN/3QBYB5

[18] P. Imai, P. Harvey, and T. Amin, "Towards a Truly Autonomus Network," Innovation Studio, Tokyo, Tech. Rep., 2020. [Online]. Available: https://corp.mobile.rakuten.co.jp/nw-lab/assets/pdf/towards_a_truly_autonomous_network.pdf