SoDA: Enabling CDN-ISP Collaboration with Software Defined Anycast

Matthias Wichtlhuber, Jan Kessler, Sebastian Bücker
Technische Universität Darmstadt, Darmstadt, Germany
Peer-to-Peer Systems Engineering
{mwichtlh,jkessler,sbuecker}@ps.tu-darmstadt.de

Ingmar Poese
BENOCS GmbH, Berlin, Germany
ingmar@benocs.com

Jeremias Blendin, Christian Koch, David Hausheer
Technische Universität Darmstadt, Darmstadt, Germany
Peer-to-Peer Systems Engineering
{jblendin,ckoch,hausheer}@ps.tu-darmstadt.de

***Abstract*—Content Delivery Networks (CDNs) carry an increasing traffic share due to the performance benefits enabled by a global deployment of proxy servers and a request-routing mechanism mapping clients to the closest CDN proxy. The standard approach to map clients/proxies is based on Over the Top (OTT) information such as delay measurements. As this approach is challenging with respect to finding the best network path, more recent Anycast CDN (A-CDN) architectures adopt Internet Protocol (IP) anycast routing. IP anycast gives the control of proxy selection completely to the network, while the standard approach gives the full control to the application. There is no middle ground enabling the use of both sets of information. Consequently, this work proposes Software Defined Anycast (SoDA) to fill the gap. SoDA enables anycast routing based on a thin layer of Software Defined Networking (SDN) functionality. Our approach allows CDN providers to take influence on anycast routing and enables collaborative load balancing at a constant SDN rule count. SoDA is shown to offer more fair resource usage at a better performance for Internet Service Providers (ISPs) and CDN providers by using a number of fine grained data sets from a large European tier-1 ISP for evaluation.**

I. Introduction

CDNs are a key technology for ecommerce and multimedia services. By caching content close to the requesting client, CDNs are able to deliver bulky content like video streams as well as high-frequency-low-volume content like Hyper Text Transfer Protocol (HTTP) requests at a high throughput and low delay. Recent studies foresee a major growth of traffic delivered by CDN infrastructures [4]. In fact, the ISP traces used for this work's evaluation indicate that already today the six largest CDN infrastructures transmit at least 34% of the overall traffic in the network[1].

A core task of all CDN architectures is the mapping of clients to the closest CDN proxy cluster. For that purpose, large CDN providers apply unicasted load balancing mechanisms based on frequently updated Domain Name System (DNS) entries or HTTP redirects. The distance of clients and CDN proxy servers is usually estimated by performing large-scale, complex delay measurements. While being able to provide load balancing among CDN proxies, these approaches cannot utilize information on the physical structure of the network. Thus, client and CDN proxy mismatches [18] occur frequently, a problem currently subject to research and standardization [10], [16].

More recent Anycast CDN (A-CDN) architectures adopt IP anycast routing as a request-routing mechanism [12], [2]. IP anycast works by announcing the same network prefix from multiple locations. Clients are routed to the closest proxy cluster as determined by the routing protocol. While IP anycast routing offers the benefit of low overhead and simple management, it sacrifices flexibility, as it is indifferent to load for the ISP as well as for the A-CDN provider.

Anycast's shortcomings with respect to load balancing are difficult to eliminate due to the inflexible architecture of today's ISP networks. However, a minimal deployment of Software Defined Networking (SDN) hardware offers means to design an optimized IP anycast routing service combining the advantages of unicasted load balancing mechanisms and standard IP anycast routing.

Consequently, this work proposes Software Defined Anycast (SoDA). As depicted in Figure 1, SoDA is designed with the following design goals in mind:

1) SoDA enables collaborative load balancing of A-CDN proxy clusters and ISP networks.
2) SoDA requires comparable management overhead to IP anycast routing for the A-CDN provider.

We evaluate our approach using several fine-grained data sets from a large European tier-1 ISP network focusing on the traffic of a large, globally operating CDN provider.

II. Background

In the following, existing CDN architectures are discussed with a focus on *request-routing mechanisms*. The request-routing mechanism solves the problem of locating clients and mapping their request to the closest CDN proxy. The prevalent approaches are *DNS request-routing*, *HTTP request-routing* and *anycast request-routing*.

[1]Due to the properties of the provided data sets, this is a lower boundary.

ISBN 978-3-901882-94-4

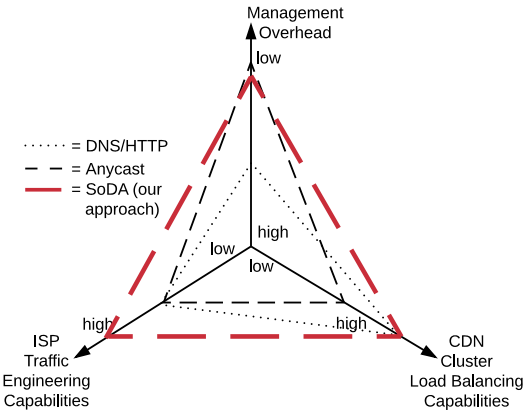

Fig. 1: SoDA's design goals in the context of existing CDN request-routing schemes.

A. DNS Request-Routing

Many CDN providers follow the DNS request-routing approach pioneered by Akamai [7]. For that purpose, the CDN provider maintains authoritative name servers. When a DNS request is to be resolved, the returned IP is chosen according to the origin of the request. More precisely, the name server assumes the client to reside near the first DNS resolver (LDNS) and delegates the client to a close proxy based on historical delay measurements and load balancing information.

The DNS request-routing approach leads to a frequent mislocation of clients due to the implicit assumption that the LDNS is close to the clients [18], [7]. This is not always true due to the use of public resolvers, company networks with own DNS resolvers, and the size of large ISP networks. There has been some recent progress with Extended DNS (EDNS) [17], [7], which is used by Google and OpenDNS. For a more elaborate discussion of EDNS and related approaches aiming at increasing DNS request-routing precision, see Section V.

B. HTTP Request-Routing

HTTP request-routing works by requesting content from a presumably close proxy, e.g., based on LDNS information or IP/geolocation databases. The proxy server is not guaranteed to possess or transparently fetch the content, but it has knowledge where the content can be found and answers with an HTTP redirect to be re-resolved by the client to find the content. HTTP requests can be performed in multiple rounds and can also incorporate DNS knowledge when the client resolves the domain names in the HTTP redirect [5].

HTTP redirects incur a high overhead in terms of delay. Consequently, this method is mainly used for the distribution of content that is bandwidth critical and not delay critical, e.g., video on demand streams.

C. Anycast Request-Routing

Recently emerging A-CDN architectures used by Microsoft (FastRoute [12]), Cloudflare [2] and others [8] utilize IP anycast routing as a mapping mechanism. For that purpose, the CDN provider announces the same anycast prefix from multiple locations. From there, the prefix is propagated into other Autonomous Systems (ASs). A router receiving conflicting announcements can decide based on a number of criteria which route should be preferred. The decision is usually based on AS hop count and the Interior Gateway Protocol (IGP) weight. Thus, proxy clusters with a direct connection to an ISP's network will usually be preferred due to a shorter AS hop count.

Anycast request-routing is a mechanism with a low overhead in terms of network engineering, as no delay measurements and no large DNS infrastructure is necessary. At the same time, anycast request-routing finds a close to optimum network path in terms of the involved routing protocols' distance metrics.

However, anycast request-routing sacrifices flexibility to reach this goal. First, IP anycast routing is agnostic to network load. If an anycast prefix is announced from multiple locations, an announcing AS has to handle the traffic from all routers deciding to route content requests to the respective AS. Consequently, A-CDNs must either adapt their peering strategy or add content delivery capacity to proxy clusters where necessary to balance load, which leads to less dense deployments. Second, ISPs cannot easily distinguish A-CDN traffic from other traffic. Thus, ISPs can hardly apply dedicated traffic engineering to anycast traffic and cannot take advantage of the knowledge that the content is available from multiple proxy clusters.

III. System Design

As an SDN enabled ISP network is assumed as a base of this work, a typical ISP network architecture and the assumptions taken for an SDN deployment are discussed. Moreover, we present SoDA's architecture and its SDN approach.

A. Assumptions

ISP networks are usually composed of an access network connecting customers and a core network routing internal traffic and connecting the AS to neighboring ASs [11]. Both are connected by a Broadband Network Gateway (BNG) performing admission control and generating billing information. Moreover, the BNG routes traffic between both networks.

Similar to recent industrial projects [3] aiming at virtualizing the edge between core and access network, we assume an SDN layer at the BNGs. This approach allows managing traffic before core routing is performed. Consequently, such a deployment is indifferent to the core routing mechanism, regardless whether Multiprotocol Label Switching (MPLS) or a native Internet Protocol version 6 (IPv6) routing core is used. As MPLS routing is widely deployed, we assume an MPLS core in the following and use the MPLS terminology, i.e., BNG and Label Edge Router (LER) are used interchangeably, as the LER functionality is usually implemented in the BNG.

B. Idea and Architecture

Figure 2 illustrates how SoDA integrates into an existing A-CDN architecture. The most relevant change is the

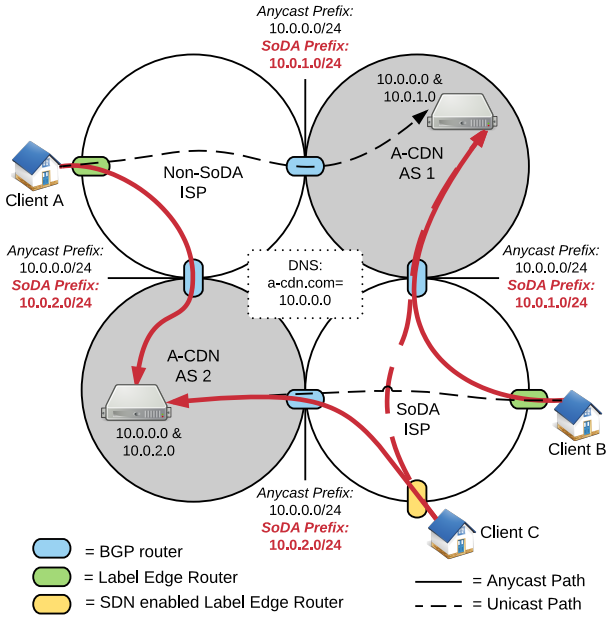

Fig. 2: Inter-AS level view of SoDA and compatibility to IP anycast, even when an SDN data plane is only partially deployed in a SoDA AS.

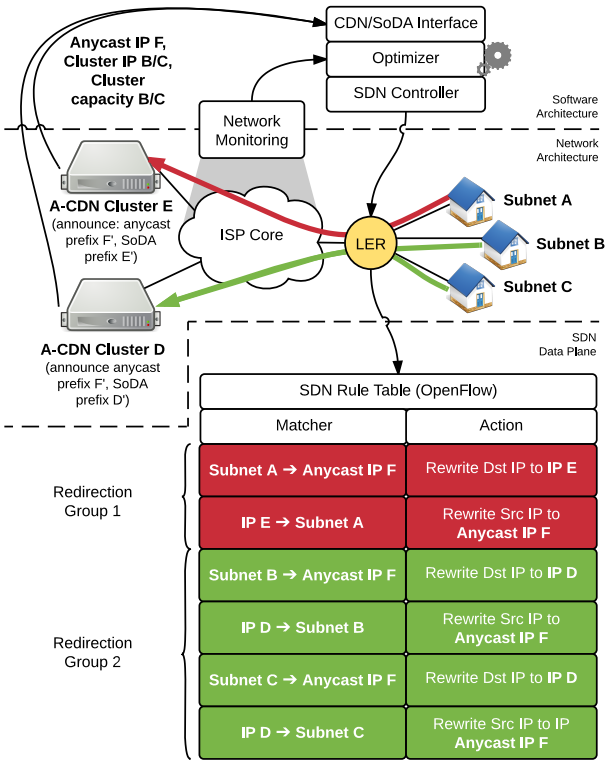

Fig. 3: Overview of SoDA system design.

announcement of two network prefixes per A-CDN cluster instead of one. The *anycast prefix* is intended to maintain compatibility with existing IP anycast routing and is equal for each cluster. As opposed to the anycast prefix, the *SoDA prefix* is a unique prefix per A-CDN cluster. It establishes unicast paths from any LER to any A-CDN cluster, thus establishing the possibility to route to a specific proxy cluster by selecting a destination IP address from the cluster's SoDA prefix.

The A-CDN's DNS entry is not adapted and points to an IP from the anycast prefix. Thus, clients attached to a non-SDN enabled LERs use the anycast route. However, at SDN enabled LERs, the SDN functionality can be used to rewrite the IP address to an address from the SoDA prefix. Consequently, an SDN enabled LER can chose a route different from the anycast route to load balance traffic between CDN proxy clusters. This approach is compatible with a partial roll-out of SDN hardware, thus offering added value from the very beginning of a migration to an SDN-enabled ISP network.

The general system design of SoDA as implemented by the ISP is depicted in Figure 3. The A-CDN provider announces the anycast IP used for DNS resolution and a tuple (IP_n, C_n) to the ISP, where IP_n is the unicast IP per proxy cluster n, and C_n is the maximum traffic volume capacity that can be handled by the respective proxy cluster.

The information on clusters is passed to an optimizer. The optimizer has access to the ISP's network monitoring information. Based on the view of the A-CDN and ISP network, the optimizer assigns each of the ISP's customer facing IP subnets to an A-CDN cluster IP. The mapping decision is forwarded to the SDN controller instance updating the SDN data plane.

C. SDN-based Redirection

The SDN rules used for redirection are depicted in the lower part of Figure 3. There are two rules per subnet and LER, one for traffic flowing from the client's subnet to the proxy cluster, the other for the backward direction. The two rules establish a virtual anycast IP F at the LER.

We explain the purpose of the two rules using subnet A as an example. The traffic from the client's side is matched to come from the respective subnet A and for having F as a destination address. If both criteria are met, the destination IP address is rewritten to the CDN's IP E, which is the proxy cluster handling the request. For the opposite direction, a second rule matches the traffic from proxy to the client by matching the destination IP to be in subnet A and the source IP address to be equal to the proxy IP E. If the packet matches, the source IP of the traffic is rewritten to anycast IP F.

Augmenting a virtual anycast IP at the LER has two advantages. First, just like IP anycast, the approach is completely transparent to the client. However, the ISP can identify and manage the traffic. Second, both flows of an HTTP session traverse the same LER, which is not guaranteed for other routers. Consequently, the whole path of the A-CDN traffic of a subnet can be determined by updating two SDN rules in one LER, while updating multiple SDN devices at the same time can have a number of non-trivial consistency issues [20].

As depicted in Figure 3, LERs usually handle multiple subnets. We define the following terminology: A *redirection rule* consists of two rules redirecting a single subnet to an

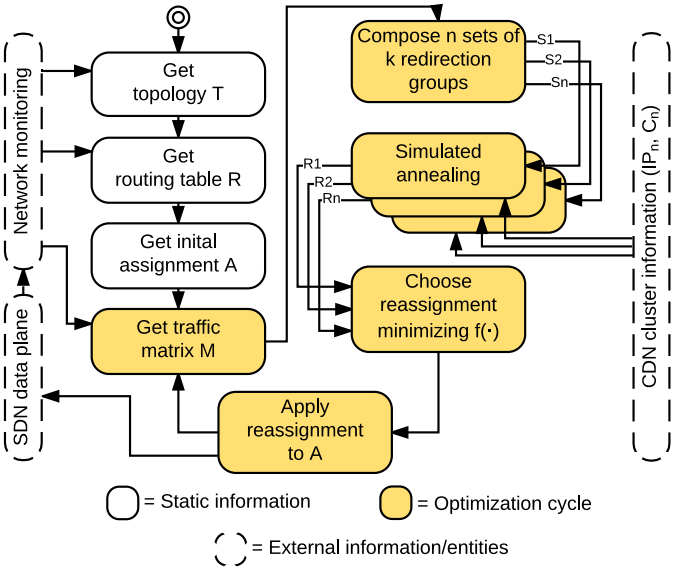

Fig. 4: Flow of SoDA's optimization cycle.

A-CDN proxy IP. A *redirection group* is a set of redirection rules redirecting multiple subnets to the same IP. Multiple redirection rule groups may exist in an LER.

D. Optimization

The joint optimization of the ISP network and the CDN network is a multi criteria optimization problem. The goal of the assignment optimization process is to find an LER proxy cluster assignment A assigning all redirection rule sets to a proxy cluster given a topology T, the ISP's routing matrix R, and the ISP's traffic matrix M. The target function to be minimized is defined as:

$$\begin{aligned} f(A,T,R,M) = {} & w_{\text{wpd}} * \text{wpd}(A,T,R,M) + \\ & w_{\text{wah}} * \text{wah}(A,T,R,M) + \\ & w_{\text{et}} * \text{et}(A,T,R,M) + \\ & w_{\text{llf}} * (1 - \text{llf}(A,T,R,M)) + \\ & w_{\text{clf}} * (1 - \text{clf}(A,T,R,M)) + \\ & C * (\text{llv}(A,T,R,M) + \text{clv}(A,T,R,M)), \end{aligned} \tag{1}$$

where $\text{wpd}(\cdot)$ is the average two way path delay over all assignments weighted by traffic volume, $\text{wah}(\cdot)$ is the average one way hop count over all assignments weighted by traffic volume, $\text{et}(\cdot)$ is the traffic volume exchanged over links with an AS distance larger than one, $\text{llf}(\cdot)$ is Raj Jain's fairness index [13] over the link utilizations and $\text{clf}(\cdot)$ is Raj Jain's fairness index over all server utilizations. Link load violations ($\text{llv}(\cdot)$) and server capacity violations ($\text{clv}(\cdot)$) are punished by a cost factor C.

The problem is a discrete optimization problem with a large solution space. With the given ISP data, searching the whole solution space is not tractable. Moreover, $f(\cdot)$ lacks useful mathematical properties for optimization. Consequently, we use a heuristics based algorithm for approximating the global optimum, as depicted in Figure 4.

The algorithm starts by collecting the necessary static information, i.e., the topology, the routing table, and the initial assignment of LERs and proxy clusters as given by IP anycast routing. Afterwards, the optimization cycle starts by updating the traffic matrix with current monitoring information from the network. Subsequently, n sets $S_1, \ldots, S_n$ of k redirection groups are randomly composed from all possible redirection groups. The Simulated Annealing (SA) [14] optimization algorithm is applied on each set. Intuitively, the SA algorithm reassigns each of the k redirection groups iteratively to other proxy clusters considering the given state of the network while trying to minimize $f(\cdot)$. The result of the process are n reassignments $R_1, \ldots, R_n$. The reassignment minimizing $f(\cdot)$ is accepted as a solution. It is applied to the network by updating the redirection groups on the respective LERs.

This approach allows taking influence on the convergence speed and quality of the solution: if k is chosen to be large, more routes will be adapted in each optimization cycle inducing a higher cost in terms of route stability, but constraint violations may be resolved faster as well. Moreover, if n is chosen to be large, the solution space can be investigated more thoroughly by evaluating more reassignments in each optimization cycle at the cost of additional CPU cycles.

E. Overhead, Guarantees, and Resilience

The system design and optimization approach allows formulating hard guarantees on the resource usage of the data plane. First, the number of total SDN rules in the network is constant, as it is determined by the number of subnets configured by the ISP. This is a highly desirable property, as it allows dimensioning the data plane independently of the traffic volume. Consequently, the maximum SDN rules updates that may be triggered during an optimization cycle can be bound as well: the maximum number of SDN rule updates is triggered, if the optimizer chooses to reassign the k largest redirection groups. Likewise, the controller will communicate with a maximum of k LERs at the same time, which circumvents known problems when communication with many SDN devices [22]. A more rigorous definition of these guarantees can be found in [24].

As SoDA is designed to superimpose standard IP anycast routing at SDN enabled LERs, IP anycast routing can be used as a fallback mechanism in case the controller fails. For that purpose, SDN devices usually implement a fallback to the built-in logic if the SDN controller is unreachable.

IV. Evaluation

SoDA is evaluated using a two-step approach. First, relevant parts of the system are implemented using mininet [15] and a Ryu[2] SDN controller (emulation model). Insights from the emulation model are used to design a valid simulation model with respect to data plane behavior. The simulation model is written in Python and relies heavily on the Fast Network

[2]https://osrg.github.io/ryu/, last visited 04/27/2017.

Simulation Setup framework [21]. This approach was chosen as the high amount of entities and the high traffic volume found in the ISP's data sets can hardly be emulated at full scale. If not stated otherwise, simulation results are presented in the following while details on the emulation experiments may be found in [24].

A. Data sets

In total, there are four data sets: *sampled NetFlow* traces, a *packet trace*, the ISP's *topology*, and the ISP's *routing matrix*. The sets span a whole day. The NetFlow traces are sampled by all several hundred LERs of the ISP's network with a constant sampling rate and a resolution of five minutes. The LERs cover the whole edge of the network including customer and external facing links, i.e., the measurements contain a complete view of the ISP's traffic. Additionally, the NetFlow traces include information on whether the traffic is *external* traffic routed via multiple AS hops, or whether the traffic is *on net*, i.e., coming from a directly neighboring AS and therefore from a point of presence of the CDN provider within the ISP's network. As the NetFlow traces are sampled, one unsampled packet trace was obtained from one top-talking LER, i.e., an LER routing a comparably high traffic volume. This trace has full resolution and allows synthesizing realistic CDN request traffic patterns.

The ISP topology data set describes the topology of the network including the locations of routers. We assume the delay between routers to scale with the geographic distance using a factor of $\frac{2}{3} * c$, where c is the speed of light. This is an estimate taking the slowing of propagation delay of light in fiber into account [1]. In order to provide for correct routing information, the topology is complemented by the routing matrix defining an MPLS path between any two routers in the network. As the MPLS paths are set according to the IGP weights, the routing matrix accurately reflects the traffic engineering efforts of the ISP.

B. Session Loss Affected Volume Model

Adapting a redirection group leads to a disruption of ongoing traffic, as all packets traversing the LER are immediately sent to the new destination. This raises the question whether a route stabilization mechanism is necessary to uphold ongoing sessions or whether a hard redirect is sufficient. In order to evaluate the impact of session loss, a number of emulation experiments are performed. Due to space limitations, the setup and results are summarized, only.

In a mininet emulation environment, the following setup is instantiated: a star topology with one client and two HTTP servers, which are connected using a single SDN switch emulating an LER in the middle. The client requests HTTP content from the servers with the same request length and volume distributions as found in the packet trace. Using the SDN based mapping approach, the delivering HTTP server is swapped with differing update intervals.

The main conclusion of this experiment is that the loss of HTTP sessions mainly depends on the update interval of the route and asymptotically approaches zero quickly with an increasing update interval. This insight is used to define the Session Loss Affected Volume (SLAV) model. The SLAV is a predictor for the fraction of traffic affected by session loss caused by SoDA's route adaptations. More details on the experiment setup and the SLAV model can be found in a technical report backing this work [24].

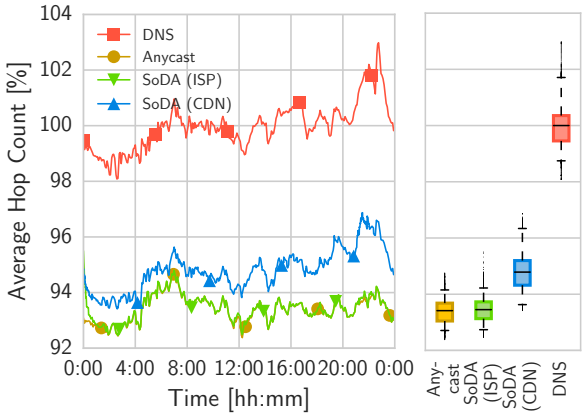

Fig. 5: Comparison of average hop count relative to CDN peak traffic volume normalized to the performance of the DNS scenario. Whiskers mark the 5th/95th percentile.

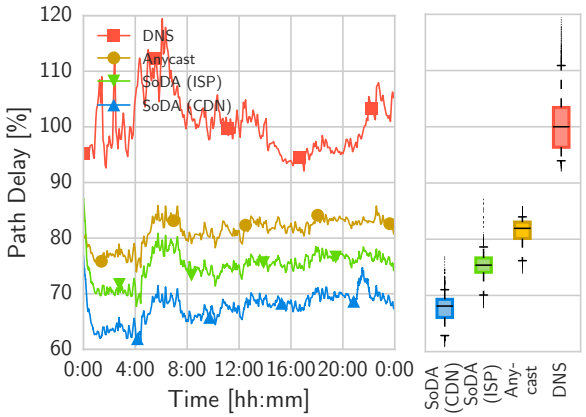

Fig. 6: Comparison of path delay relative to CDN peak traffic volume normalized to the performance of the DNS scenario. Whiskers mark the 5th/95th percentile.

C. Evaluation Scenarios

We define four evaluation scenarios: a *DNS scenario*, an *anycast* scenario, and two scenarios using different optimization weight vectors for SoDA (*ISP optimized scenario* and *CDN optimized scenario*).

In a first step, we filter the NetFlow traces for the traffic of the top-talking CDN provider, a world-wide operating CDN with multiple on net presences within the ISP's network. As the CDN provider is using a DNS based mapping mechanism, we define the *DNS scenario* as a pure playback of the CDN traffic as gathered from the data. The *anycast scenario* is derived from the DNS scenario by routing the traffic at each LER to the proxy cluster with the lowest IGP weight.

The *ISP optimized scenario* applies SoDA with a weighting vector leaning towards the ISP's interest. The optimizer optimizes for a low hop count (w_{wah}) and a high link load fairness (w_{llf}). Moreover, the optimization avoids external downstream traffic (w_{et}), i.e., traffic served from non-neighboring ASs. The *CDN optimized scenario* uses SoDA with a weighting vector in favor of the CDN provider's performance indicators. The main weight is put on a low path delay (w_{wpd}) and a high cluster load fairness (w_{clf}). The CDN optimized scenario avoids external traffic (w_{et}), as external traffic traverses multiple ASs and may therefore be routed over under provisioned peered links incurring unnecessary path delay.

D. SoDA vs. IP Anycast/DNS

Figures 5 and 6 show the performance in terms of path delay and hop count. The hop count statistic represents the number of ISP core hops from proxy server to client. The path delay denotes the delay from client to proxy server and back only considering the time a packet spends in fiber in the core network. Both metrics are weighted by the transmitted volume and normalized to the performance of the DNS scenario.

IP anycast delivers the traffic at the lowest hop count outperforming the DNS scenario by about 6%. Nevertheless, the same is not true for the path delay, where IP anycast as well as DNS is outperformed by both SoDA configurations. As the ISP configuration of SoDA optimizes for a high fairness of the link utilizations, while the CDN configuration optimizes for path delay, the CDN configuration has a small advantage with respect to the latter metric. However, when comparing the CDN configuration of SoDA and the DNS scenario, the path delay can be lowered by 33%.

Figure 7 investigates the fairness of proxy cluster utilization using Raj Jain's fairness index [13]. The SoDA scenarios constantly outperform the other scenarios at a near perfect cluster load fairness, where anycast is clearly performing bad, as the cluster load capacities in the CDN provider's deployment are not optimized for anycast request-routing. This observation aligns with the measurement of the cluster load excess, as illustrated in Figure 9. Cluster load excess is defined as the sum of traffic that cannot be handled by the CDN provider as it is requested from overloaded proxy clusters. It is expressed as a fraction of the peak traffic volume delivered by the CDN provider. Due to the missing load balancing capabilities of anycast request-routing, up to 16.8% of the volume would have to be dropped. This demonstrates the shortcomings of anycast request-routing with respect to arbitrary capacity deployments.

Figure 7 also shows an interesting observation for the DNS scenario: while the three other scenarios show a more or less constant performance, the DNS scenario's performance varies over the day. We cannot explain this effect with absolute confidence. A likely explanation is, that the CDN provider sacrifices fairness for lower delay when demand is low.

The plot depicted in Figure 8 compares the external link traffic for the four scenarios. It is expressed as the fraction of the peak traffic volume delivered by the CDN provider that is routed via more than one AS hop. Anycast as well as the two SoDA configurations avoid external traffic, while the DNS scenario triggers up to 12% of the peak traffic volume. This is likely to be a symptom of the LDNS problem (Section II-A).

Notably, SoDA does not trigger any external traffic and

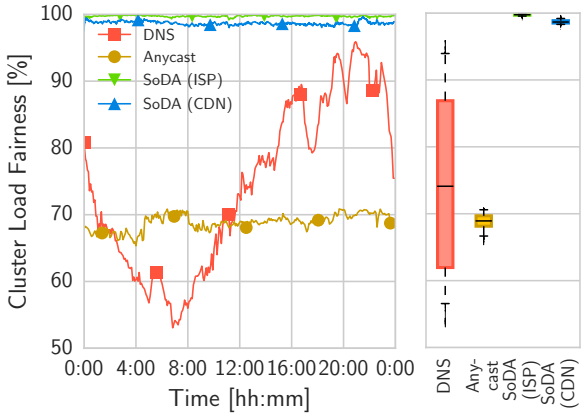

Fig. 7: Comparison of cluster load fairness. Whiskers mark the 5th/95th percentile.

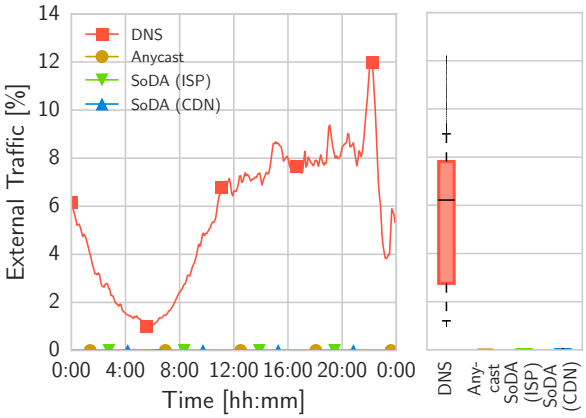

Fig. 8: External traffic volume as fraction of the CDN's peak traffic volume. 12% of the traffic is delivered via multiple AS hops with DNS. Whiskers mark the 5th/95th percentile.

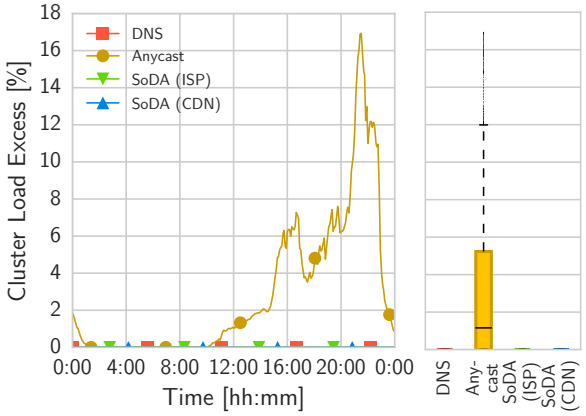

Fig. 9: Comparison of cluster load excess relative to CDN peak traffic volume. Whiskers mark the 5th/95th percentile.

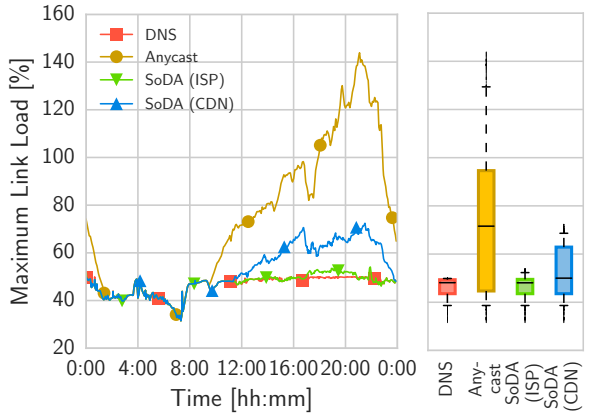

Fig. 10: Comparison of maximum link load. Whiskers mark the 5th/95th percentile.

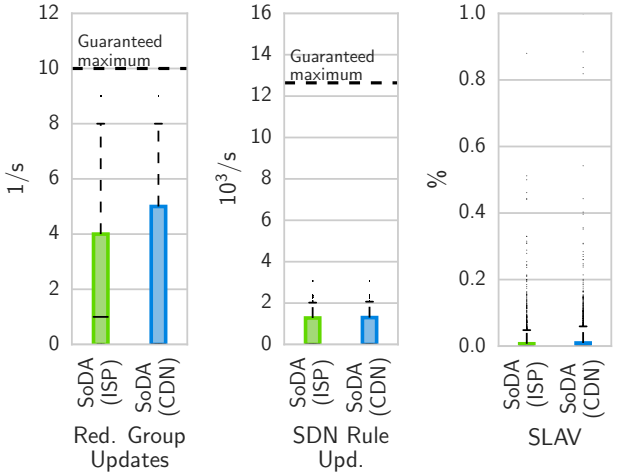

Fig. 11: Updated redirection rule sets (left), maximum SDN rule update rate (middle), and SLAV (right) per 20s optimization cycle. Whiskers mark the 5th/95th percentile.

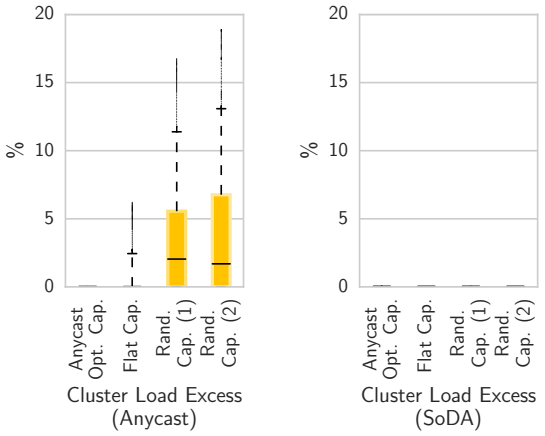

Fig. 12: Comparison of cluster load excess with IP anycast as fraction of the CDN's peak traffic volume comparing different capacity deployments. Whiskers mark the 5th/95th percentile.

no cluster load excess (Figure 9) at the same time. This demonstrates that the capacity installed by the CDN provider is enough to keep the complete traffic local within the ISP's AS. However, due to the LDNS problem and a non-optimal utilization of the clusters during the day, a non-negligible amount of traffic is served from distant ASs.

In Figure 10 the maximum link load is shown, i.e., the utilization of the most utilized link in the network. While IP anycast causes at least one congested link, the utilization of the ISP friendly configuration of SoDA is comparable to that of the DNS scenario, while the CDN friendly configuration increases the metric. This illustrates the generic trade-off between cluster load fairness (Figure 7) and maximum link load, i.e., cluster load fairness can be increased at the cost of the maximum link load.

E. Optimization Dynamics and Cost

SoDA's cost metrics are mainly influenced by the dynamics of the optimization process, i.e., how aggressive routes are adapted to meet the optimization goals. For all experiments, the maximum number of redirection groups to be adapted during an optimization cycle is limited to $k = 10$, while the length of the optimization cycle is 20s.

Figure 11 (left) shows how aggressively the maximum number of redirection group updates is utilized. In 95% of the cases, 8 or less updates are performed by SoDA, i.e., the budget of redirection group updates does not constrain the optimization algorithm. The SDN rule update rate (middle) measures the maximum observed update rate of SDN rules observed at a single LER. For both configurations, the 95th percentile is as high as 2022.0 rule updates per second and an observed maximum 3082.0 rule updates per second, which is far below the guaranteed maximum. The SLAV (right) expressing the amount of traffic affected by session loss due to redirections does not exceed 0.06% (95th percentile). Consequently, a session stabilization mechanism for HTTP sessions as discussed in Section IV-B is not necessary. This is a desirable result, as stabilizing existing sessions would require keeping per session state in the SDN hardware: OpenFlow does not have means to separate existing sessions from new sessions.

F. Capacity Management

In order to evaluate SoDA's capabilities to manage arbitrary capacity deployments, we introduce four new scenarios with varied capacity constraints. The *anycast optimized scenario* is adapted to the requirements of anycast, i.e., we deploy exactly the cluster and networking capacity needed at every CDN cluster such that anycast will not face any cluster load excess. The *flat capacities* scenario distributes the capacity uniformly to the clusters. Two random distributions of capacity to the CDN's clusters complement the range of scenarios. All scenarios have the same total amount of capacity available.

Figure 12 compares IP anycast routing and SoDA with respect to the cluster load excess triggered by both mechanisms. While SoDA does not show any excess in dealing

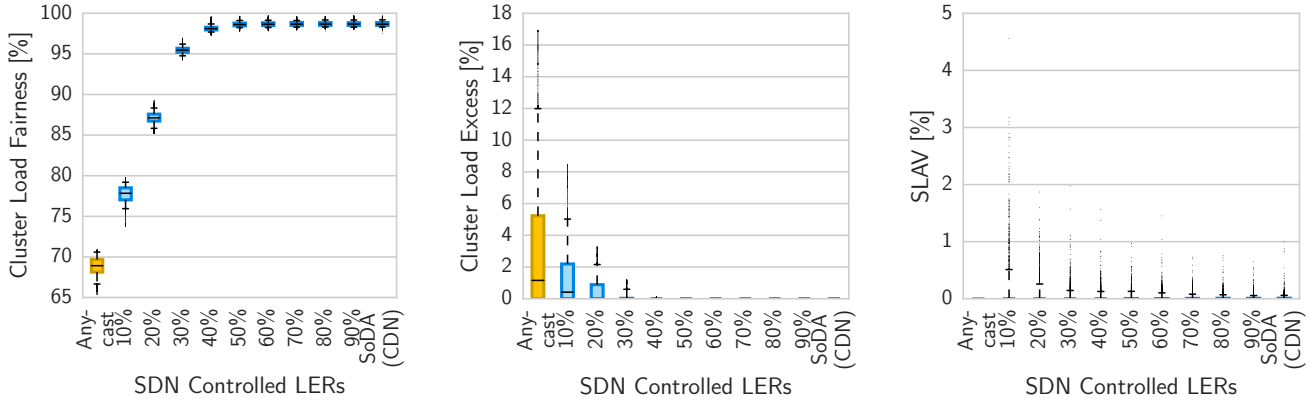

Fig. 13: Comparison of key metrics for different fractions of SDN enabled LERs. The whiskers mark the 5th and 95th percentile.

with varying capacities, anycast has considerable problems except for the anycast optimized deployment. Thus, SoDA allows A-CDN providers to adapt their deployment according to economic interest (e.g., co-location cost) instead of adapting their deployment to IP anycast routes.

G. How much SDN is needed?

SoDA is designed to work with a partially deployed SDN layer inside the ISP's network, i.e., only a fraction of the ISP's LERs may be configurable by the controller, while the remaining LERs run standard IP anycast routing. This is an important property allowing for an incremental deployment of SDN hardware and an early return on investment for ISPs.

Consequently, we determine the minimum fraction of SDN controlled LERs needed to provide added value. The fraction of SDN controlled LERs is increased incrementally from 0% (pure IP anycast routing) to 100% (pure SoDA routing). As the distribution of volume among LERs is highly skewed with 8.3% of the LERs routing 50% of the CDN traffic, we prefer top talking LERs, e.g., a fraction of 20% SDN enabled LERs represents the 20% LERs routing the highest traffic volumes.

The key metrics for an increasing fraction of SDN enabled LERs are depicted in Figure 13. The cluster load fairness quickly saturates. From a fraction of 50% on, no additional gains can be achieved. A similar trend can be observed for the cluster load excess. The constraint of not overwhelming single clusters with traffic can be met from a fraction of 40% SDN enabled LERs. At the same time, the SLAV is influenced positively with respect to the 95th percentile value and outliers. However, even with very low fractions of SDN control, the fraction of volume affected by session loss remains uncritical: the 95th percentile for the 10% case is as low as 0.52%.

V. Related Work

There are several related works to SoDA ranging from *CDN-ISP collaboration systems*, *A-CDN architectures* to *SDN based work* related to load balancing.

A number of works propose systems providing ISP's internal information to the application layer. Early approaches are motivated by the poor traffic locality achieved by Peer-to-Peer (P2P) content distribution systems, e.g., Provider Portal for Applications (P4P) [26]. Based on the same idea, the Internet Engineering Task Force (IETF) group on Application Layer Traffic Optimization (ALTO) started standardizing a protocol for the interaction of ISPs and P2P applications [6]. Recently, the ALTO group focuses on CDN proxy selection as well.

The Provider Aided Distance Information Service (PaDIS) [18] can act as a mechanism running the ALTO protocol for interaction with CDNs. Moreover, PaDIS can be integrated in ISPs' DNS resolvers. The proxy server IP as resolved by the CDN provider can be overridden with a more suitable IP from the ISP's perspective. The same authors proposed the Content-aware Traffic Engineering (CaTE) system [19]. Similar to SoDA, CaTE focuses on optimizing proxy selection. However, CaTE aims at DNS based request-routing instead of anycast routing.

The DNS client subnet extension [9] extends EDNS [10] requests to pass the requesting client's subnet along with the request during resolution to solve the LDNS problem. The client subnet extension is evaluated from the vantage point of the Akamai CDN in [7], showing performance benefits for the CDN provider. Obviously, this approach can increase the locality of traffic, but cannot utilize the hidden information of ISPs.

The Microsoft FastRoute architecture is presented in [12]. FastRoute utilizes multiple layers of anycast IPs and uses DNS to load balance between the layers. The system requires running DNS and CDN cache proxies on the same node. Cloudflare's architecture [2] utilizes IP anycast routing to route traffic to the closest proxy cluster and inside clusters. Both architectures are designed for fault tolerance by avoiding dependencies between CDN nodes to simplify management. Especially FastRoute [12] illustrates the need to compensate IP anycast's indifference to network load with additional mechanisms.

To this end, load-balancing with SDN has mainly been investigated for data centers [23], while the management of CDN traffic is a subject of active research. In [25], a route stabilization mechanism is investigated. As shown beforehand, route stability is not critical issue for our use case. To the

best of our knowledge, there is currently only one published proposal for SDN based anycast [27]. The authors investigate algorithms to optimize traffic but do not focus on CDNs.

VI. Conclusions

This work proposes SoDA, an SDN based anycast solution for load balancing A-CDN traffic. SoDA is motivated by the shortcomings of IP anycast routing with respect to load balancing and traffic engineering. These shortcomings are caused by the inflexibility of today's ISP architectures and a lack of cooperation between ISPs and CDN providers. SoDA is designed with the following design goals in mind.

1) SoDA enables collaborative load balancing of A-CDN proxy clusters and ISP networks.
2) SoDA requires comparable management overhead to IP anycast routing for the A-CDN provider.

Thus, SoDA combines the load balancing performance of DNS based mapping and the low management overhead of IP anycast by moving load balancing functionality into the ISP network. At the same time, a CDN provider can influence the load balancing decisions by sharing a minimum of information. SoDA reaches a close to perfect load distribution among CDN proxy clusters and prevents external/non-local traffic at low cost in terms of SDN rule updates and session loss caused by route adaptation. The benefits of SoDA are measurable from a penetration of 10% SDN hardware inside the ISP's network.

As SoDA is evaluated using measurements from a large European tier-1 ISP, a number of interesting findings could be made. First, the collaboration of CDNs and ISPs can harness a considerably large optimization potential for anycast request-routing as well as DNS request-routing. A second interesting insight regards the incremental deployment of SDN hardware. As the volume distribution among routers is highly skewed, it is sufficient to equip a fraction of the largest routers with SDN capabilities to provide added value. This requires seamless integration with less flexible legacy services. SoDA is a good example for such a service, as it can be integrated with IP anycast routing services running on legacy routers.

Due to space limitations, only a part of the available results is shown in this work. For additional information, we refer to a technical report backing this work [24]. The report contains more details on the system design and additional evaluation results for an extended range of scenarios.

References

[1] "Design Best Practices for Latency Optimization," Cisco Systems Inc., Tech. Rep., 2007.

[2] "Load Balancing without Load Balancers," CloudFlare Inc., Tech. Rep., 2013. [Online]. Available: https://blog.cloudflare.com/cloudflares-architecture-eliminating-single-p/

[3] "Central Office Re-architected as Datacenter," AT&T, Tech. Rep., 2015.

[4] "Cisco Visual Networking Index: Forecast and Methodology, 2015-2020," Cisco Systems Inc., Tech. Rep., 2016.

[5] V. K. Adhikari, S. Jain *et al.*, "Vivisecting YouTube: An Active Measurement Study," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2012.

[6] R. Alimi, Y. Yang, and R. Penno, "RFC 7285: Application-Layer Traffic Optimization (ALTO) Protocol," Internet Engineering Task Force (IETF), Tech. Rep., 2014.

[7] F. Chen, R. K. Sitaraman, and M. Torres, "End-User Mapping: Next Generation Request Routing for Content Delivery," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 4, pp. 167–181, 2015.

[8] D. Cicalese, J. Augé *et al.*, "Characterizing IPv4 Anycast Adoption and Deployment," in *ACM Conference on Emerging Network Experiment and Technology (CoNEXT)*, 2015.

[9] C. Contavalli, W. Van Der Gaast *et al.*, "RFC7871: Client Subnet in DNS Queries," Internet Engineering Task Force (IETF), Tech. Rep., 2016.

[10] J. Damas, M. Graff, and P. Vixie, "RFC2671: Extension Mechanisms for DNS (EDNS(0))," Internet Engineering Task Force (IETF), Tech. Rep., 2013.

[11] R. D. Doverspike, K. K. Ramakrishnan, and C. Chase, *Guide to Reliable Internet Services and Applications*. Springer, 2010, ch. Structural Overview of ISP Networks, pp. 19–93.

[12] A. Flavel, P. Mani *et al.*, "Fastroute: A Scalable Load-Aware Anycast Routing Architecture for Modern CDNs," in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2015.

[13] R. Jain, D.-M. Chiu, and W. R. Hawe, "A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems," Digital Equipment Corporation, Tech. Rep., 1984.

[14] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[15] B. Lantz, B. Heller, and N. McKeown, "A Network in a Laptop: Rapid Prototyping for Software-defined Networks," *ACM SIGCOMM Workshop on Hot Topics in Networks (Hotnets)*, 2010.

[16] B. M. Maggs and R. K. Sitaraman, "Algorithmic Nuggets in Content Delivery," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 3, pp. 52–66, 2015.

[17] C. Partridge, T. Mendez, and W. Milliken, "RFC1546: Host Anycasting Service," Internet Engineering Task Force (IETF), Tech. Rep., 1993.

[18] I. Poese, B. Frank *et al.*, "Improving Content Delivery Using Provider-Aided Distance Information," in *ACM SIGCOMM Conference on Internet Measurement (IMC)*, 2010.

[19] ——, "Enabling Content-Aware Traffic Engineering," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 5, pp. 21–28, 2012.

[20] M. Reitblatt, N. Foster *et al.*, "Abstractions for Network Update," in *ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, 2012.

[21] L. Saino, C. Cocora, and G. Pavlou, "A Toolchain for Simplifying Network Simulation Setup," *ACM International Conference on Simulation Tools and Techniques (SimuTools)*, 2013.

[22] A. Tootoonchian, S. Gorbunov *et al.*, "On Controller Performance in Software-Defined Networks," *USENIX Workshop on Hot Topics in Management of the Internet, Cloud, and Enterprise Networks and Services (Hot-ICE)*, 2012.

[23] R. Wang, D. Butnariu, and J. Rexford, "OpenFlow-Based Server Load Balancing Gone Wild," in *USENIX Workshop on Hot Topics in Management of Internet, Cloud and Enterprise Networks and Services (Hot-ICE)*, 2011.

[24] M. Wichtlhuber, J. Kessler *et al.*, "A Deep Look at CDN-ISP Collaboration with Software Defined Anycast (SoDA)," Technische Universität Darmstadt, Tech. Rep., 2017. [Online]. Available: http://www.ps.tu-darmstadt.de/fileadmin/publications/PS-TR-2017-01.pdf

[25] M. Wichtlhuber, R. Reinecke, and D. Hausheer, "An SDN based CDN/ISP Collaboration Architecture for Managing High-Volume Flows," *IEEE TNSM Special Issue on Efficient Management of SDN and NFV-based Systems*, vol. 12, no. 1, pp. 406–409, 2015.

[26] H. Xie, Y. R. Yang *et al.*, "P4P: Provider Portal for Applications," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 4, pp. 351–362, 2008.

[27] H. Xu, X. Li *et al.*, "High-throughput Anycast Routing and Congestion-free Reconfiguration for SDNs," *ACM International Workshop on Quality of Service (IWQoS)*, 2016.