SCULPTOR Has Your Back(up Path): Resilient, Efficient Interdomain Routing for Large Service Providers

Paper #559, 12 pages body, 19 pages total

Abstract

Large cloud and content (service) providers help serve an ever-expanding suite of applications that are increasingly integrated with our lives, but have to contend with an unreliable, unpredictable public Internet to route user requests. To enhance reliability to dynamic scenarios such as failure and DDoS attacks, large service providers overprovision sites to accommodate peak loads and build emergency systems for shifting excess traffic. We take a different approach with SCULPTOR, which uses a service provider's global resources to handle dynamic traffic conditions. SCULPTOR ensures users have many interdomain routes to deployments, and moves excess traffic to backup paths to avoid congestion and minimize latency. As predicting interdomain routes is challenging, SCULPTOR builds models of Internet routing to solve a large integer optimization problem at scale using gradient descent. We prototyped SCULPTOR on a global public cloud and tested it in real Internet conditions, demonstrating that SCULPTOR uses service provider infrastructure up to **TBD: 28**% more efficiently than other solutions, reduces congestion on links during site failures by **TBD: 17**% on average, and enables service providers to achieve low-latency objectives for up to **TBD: 17**% more traffic.

1 Introduction

Cloud and content providers (hereafter Service Providers) enable Internet applications used daily by billions of users. The applications have increasingly stringent performance and reliability demands, as the Internet is increasingly used for mission-critical applications (*e.g.,* enterprise services [**?**]) and as performance requirements become tighter (*e.g.,* virtual reality requires 10 ms round trip latency [**?**] and 3 ms jitter [**?**]). To meet these stricter requirements with the same fundamental Internet protocols, Service Providers have deployed interconnected sites and connected with thousands of networks in hundreds of locations, offering rich connectivity and distributed capacity.

The deployment's physical resources (*i.e.,* peering links, servers) are provisioned to accommodate demands from (ingress) and to (egress) user networks. Service Providers balance egress demands across links to minimize congestion and latency [**? ? ?**]. However, it is difficult to control exactly how much traffic reaches each site and how much *ingresses* each link into the Service Provider's network since (a) ingress traffic can only be controlled indirectly, by directing users to different *prefixes*, and (b) it is difficult to advertise prefixes in a way that balances load in dynamic network conditions.

Given this lack of control, resource overload occurs due to link failures [**?**], DDoS attacks [**? ? ? ?**], flash crowds [**? ?**], and route changes that cause significant traffic shifts [**? ? ? ? ? ?**]. This overload leads to degraded service for user traffic [**? ? ?**]. Service Providers may respond to resource congestion by (a) overprovisioning resources [**? ?**] or (b) draining congested links and moving some of that traffic to other resources with free capacity [**?**]. We demonstrate in Section 2.3 that overprovisioning resources can require overprovisioning rates as 70% more, significantly increasing expenses.

Recent work drains traffic from overloaded links/sites by *withdrawing* prefix announcements that are also advertised via other (healthy) links/sites [**? ? ?**], so that traffic destined to those prefixes is forced to arrive on other links/sites after BGP converges (within tens of seconds [**?**]). However, this solution to overload is post-hoc, unpredictable, and ineffective. Other solutions use traffic engineering systems in user networks to optimize latency and dynamically shift traffic [**?**], but do not provide guarantees that good backup options exist (see **??** for what can go wrong).

To solve these problems, we propose a system — SCULPTOR (**S**calable **C**onfigurations for **U**tilization, **L**oss, and **P**erformance-aware **T**raffic **O**ptimization & **R**outing) — that proactively advertises multiple prefixes to peers to expose diverse routing options, then assigns user traffic to paths towards these prefixes to optimize latency in a way that balances load across links/sites. Our key insight is to frame this problem as finding a way of advertising reachability that maximizes latency subject to capacity constraints both with and

without failures as failure requires load balancing traffic on backup paths. Our solution uses the rich connectivity of Service Providers, balancing traffic over backup paths through many sites and links as conditions change.

We make two contributions. First, we present an optimization framework that can be used to minimize performance metrics (*e.g.,* maximum link utilization, latency, latency under single link failures) over different sets of prefix advertisements. Part of our framework is a model for predicting performance in unknown scenarios—we compute probability distributions of performance metrics in advertisement configurations we have not measured, which is important as changing routing configurations and then issuing measurements is slow. This model enables SCULPTOR to efficiently optimize over a large space without measuring every possible configuration. We demonstrate that this model quickly learns how to accurately predict performance metrics using few measurements compared to alternate approaches. We then optimize these (modeled) performance metrics using gradient descent which is appropriate in our setting due to the high dimension of the problem and degree of parallelism gradient descent admits.

Second, we prototype and evaluate our framework in a system, SCULPTOR, at Internet scale using the PEERING testbed [**?**], which is now deployed at 32 Vultr cloud locations [**?**]. Vultr is a global public cloud that allows us to issue BGP advertisements via more than 10,000 peerings. We demonstrate that SCULPTOR effectively computes prefix advertisements that give users low latency both during normal operation and variable network conditions, and allows Service Providers use their deployments more efficiently, reducing costs.

Through thorough evaluations, we found SCULPTOR reduces latency during both steady state and failure: SCULPTOR improves the amount of traffic within 10 ms of optimal by **TBD: 3**% in steady state, **TBD: 11**% during link failure, and **TBD: 17**% during site failure. Hence, SCULPTOR's benefits translate to real differences in user performance for emerging applications such as VR that have a tight 10 ms latency target. SCULPTOR also reduces congestion on links during site failures by **TBD: 17**% on average, giving Service Providers more security that services will still be available during partial failure

Providing good backup paths to handle dynamic conditions improves more than just latency — we find that by load balancing traffic on backup paths during peak times, we can satisfy very high peak demands with the same infrastructure. SCULPTOR can handle flash crowds (DDoS attacks, for example) at more than **TBD: 3**× expected traffic volume, drastically reducing the amount of overprovisioning Service Providers need, thus reducing costs. SCULPTOR can also handle large diurnal swings of almost **TBD: 2**× expected load.

Hence, SCULPTOR improves interdomain routing for users to for Service Providers today, uses Service Provider resources more effectively, and acts as a tool for more efficient capacity planning preparing them to provide the increasingly reliable, performant service that our applications need.

2 Motivation and Key Challenges

2.1 Tighter Reliability Requirements, Variable Conditions

Service Providers offer their services from tens or hundreds of geo-distributed sites. Each site offers identical service so can serve any user, but users benefit from reaching a low-latency site for performance. Sites consist of sets of servers which have an aggregate capacity. Service Providers also connect to other networks (often thousands) at sites via dedicated links or shared IXP fabrics. Each such link also has a capacity. When utilization of a site or link (collectively, a resource) nears/exceeds 100%, performance suffers, so Service Providers strive to keep utilization reasonably low. Resources can also fail completely due to, for example, physical failure and misconfiguration.

Ensuring users can reliably route over healthy paths to Service Providers even during partial system failure is increasingly important as shown by the considerable attention that related problems have seen in the news and research community. Microsoft recently stressed the importance of resolving congestion on ingress links [**?**], developed a system to identify performance problems on paths [**?**], and is considering deployment deep within user networks to further enhance ingress routing performance and reliability [**?**]. Google similarly developed a system to identify performance problems on paths [**?**]. Peering disputes regularly make their way into the news, and these disputes often lead to long-standing congestion on interdomain links **TBD: cite**. DDoS attacks are an ever-present problem, as shown by recent events **TBD: cite recent events or work** and by several publications presenting new methods of mitigating DDoS attack effects **TBD: cite**. Research from Facebook, Google, and Microsoft all demonstrates considerable recent focus on providing service even under partial system failure (as partial system failure is a constant in a large networked system) [**? ? ? ?**].

Moreover, new trends in service offerings and application usage make operation under dynamic conditions both more challenging and important. Service Providers increasingly offer mission-critical services such as enterprise solutions [**? ? ? ? ?**], so ensuring performant, reliable operation is more important for these services now than ever. Recent work shows that user traffic demands are highly variable due to flash crowds, congestion, and path changes and so are much harder to plan/optimize for than inter-datacenter demands [**?**]. Operators regularly report such events on social media and blog posts **TBD: cite**. Such trends could become more salient as 5G/next-generation applications drive massive amounts of traffic from users to services [**?**], making user demands even more variable and more challenging to satisfy.

2.2 Approaches to Interdomain Routing

A challenge in offering low-latency, reliable services to users is routing traffic from user networks to Service Provider networks since Service Providers lack full control of which interdomain path traffic takes. BGP, the Internet's interdomain routing protocol, computes paths in a distributed fashion, giving each intermediate network a say in which paths are chosen and which are communicated to other networks. Service Providers can, however, advertise their reachability to peers/providers in different ways to increase the chance of there being good paths for users. Today, Service Providers either direct users to specific sites using `unicast` prefix advertisements [**? ? ?**], or use `anycast` prefix advertisements to provide relatively low latency and high availability at the expense of some control [**? ? ?**] **TBD: more**.

`anycast`, where Service Providers advertise a single prefix to all peers/providers at all sites, offers high availability following failures since BGP automatically ensures reachability to the deployment after tens of seconds for most networks [**?**]. Prior work shows that this availability comes with higher latency in some cases [**? ?**]. `unicast` advertises a unique prefix at each site, enabling user redirection to a particular site [**?**]. Hence, recent work proposes hybrids of `unicast` and `anycast` which achieve a happy medium between performance and availability [**? ? ? ?**]. This latter class of solutions advertises different prefixes to subsets of all peers/providers to offer users many low-latency options across different links/sites. We refer to these solutions as `selectivecast` solutions since they are *selective* about who they advertise prefix reachability to, and have traits of both `anycast` (advertising at many sites to many peers/providers) and `unicast` (many prefixes, selectivity).

2.3 Current Approaches Do Not Consider Dynamic Conditions

In `anycast`, `unicast`, and `selectivecast` scenarios, Service Providers set capacities based on steady state demand. For example, Service Providers might overprovision resources by a fixed percentage. Hence, none of these approaches to advertising reachability (`unicast`, `anycast`, or `selectivecast`) explicitly plan for dynamic traffic conditions. For example, `AnyOpt` and `PAINTER` (`selectivecast` solutions) find low-latency paths [**? ?**], but it is unclear whether these paths can satisfy user demands under a flash crowd, and it is unclear how to scale these approaches to account for such scenarios. None of these approaches *even take user demands into account* when computing how to advertise reachability.

Instead, the current state-of-practice to handle dynamic conditions is to drain traffic from overloaded links/sites by *withdrawing* prefix announcements that are also advertised via other (healthy) links/sites [**? ? ?**]. The traffic destined

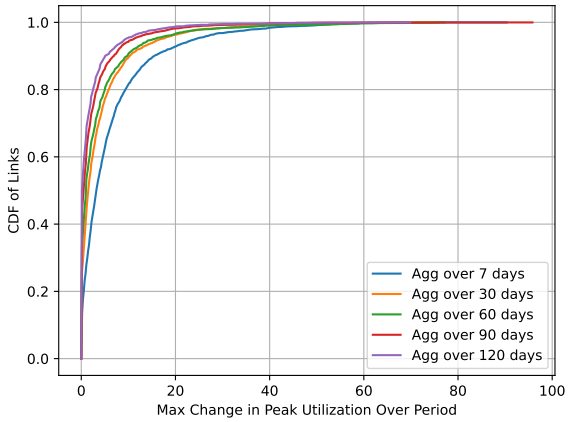

Figure 1: Planning for future peak loads requires excessive overprovisioning.

to those prefixes then arrives on other links/sites advertising those prefixes after BGP reconverges. However, this solution to handling dynamic conditions is post-hoc, unpredictable (as BGP is unpredictable), and ineffective. For example, TIPSY only achieves 70% prediction accuracy and (in a case study) required several prefix withdrawals to mitigate a single congested link [**?**].

Another common solution to tackling dynamic conditions is to overprovision resources to handle predicted peak loads in the future [**? ?**], but Figure 1 demonstrates that doing so can incur excessive costs. Figure 1 computes differences in peak utilization on links between different successive time periods, using longitudinal link utilization data from OVH cloud [**?**]. We split the dataset into successive, non-overlapping N-day periods and compute changes in 95^{th} percentile link utilization from one period to the next. We choose this percentile to reduce noise. Even when computing near-peak loads over successive 120-day time periods, near-peak loads can increase by 10% for 7% of links, 20% for 2% of links, and increase as high as 70%, which illustrates the amount Service Providers must overprovision to ensure they can meet resource demands. Using backup paths to handle such (rare) events can help keep costs low.

2.4 Key Problem

The key problem with current practices of routing users to Service Providers is that Service Providers have no guarantee that backup paths/sites can handle new traffic loads.

Prior work finds ways to prepare for intradomain failure even under dynamic traffic conditions [**? ? ? ? ?**], which implicitly requires allocating backup paths. Compared to this intradomain setting, two key differences in the interdomain setting are that (a) a Service Provider cannot precisely control the paths traffic takes since BGP relinquishes this control to other networks and (b) a Service Provider cannot easily deter-

mine the bottleneck capacity of paths before placing traffic on them. We explicitly address the former problem and leave the latter problem as future work. That is, we consider the problem of ensuring reliable (failure-resistant) interdomain routing when we know the available capacity of backup paths. We approximate the capacity of a path as the capacity of the corresponding peering link through which that path ingresses to the Service Providers' deployment. We consider traffic *ingressing* the deployment since, in the egress setting, the Service Provider has full control over traffic up until the egress point which is the only part of the egress path the Service Provider can control. Existing systems direct egress traffic to optimize performance and mitigate congestion [**? ? ?**].

Although we do not explicitly address problem (b) above, we implicitly address it by finding failure-resistant interdomain routing strategies (*i.e.,* having backup paths to route around congestion is a good thing), and there are natural extensions of our setting that consider such effects. For example, one could use congestion-detection mechanisms along paths to trigger traffic shifts to other, healthier (backup) paths.

Service Providers lack interdomain control for their ingress traffic, but there is hope that they can handle dynamic conditions. A Service Provider has enough global capacity to handle even large changes in traffic or sudden traffic shifts from one link to another [**? ?**]. It can use this global capacity to handle large, localized loads instead of provisioning for peak loads, but no current solution to interdomain routing makes use of this global capacity.

?? shows how a certain type of dynamic condition — failure — can lead to performance problems, even with `selectivecast` advertisements, and how Service Providers can avoid such overutilization with enough foresight. In normal operation, user traffic is split evenly across two prefixes. However when site B fails, all traffic ingresses through the same link since BGP chooses the route through Provider 1, causing overload (links have capacity 1). Advertising prefix 2 to provider 2 at site A *a priori* allows the Service Provider to split traffic between the two links during failure, avoiding overload.

?? encapsulates why current solutions to handling dynamic conditions are inefficient. `selectivecast` solutions that expose low-latency paths [**? ?**] do not plan for changing traffic conditions, only giving users low-latency *primary* paths. Withdrawing/advertising prefixes to shift load is another way to address this failure [**?**], but such strategies could inundate other, healthy links such as Internet routes are hard to predict. Moreover, such strategies still result in reduced performance as detecting, withdrawing, and advertising prefixes takes time. Finally, the Service Provider could overprovision to plan for failure, but doing so is a waste of resources as the deployment has enough global capacity to handle traffic congestion-free.

2.5 Key Challenges

Since Service Providers connect globally with thousands of networks, there is generally sufficient available capacity across paths/sites to satisfy user demand even under dynamic conditions. For example, an operator at a large Service Provider told us that they often have redundant connections to important customers at multiple sites. Placing traffic demand on paths to optimize performance objectives subject to capacity constraints would therefore be simple *if we could expose all the paths*.

However, exposing paths uses prefixes which are expensive [**?**]. IPv4 prefixes are monetarily expensive and pollute BGP routing tables. Prior work generally found that advertising O(50) prefixes was acceptable [**? ?**], and most Service Providers advertise fewer than 50. IPv6 is not a good alternative, as such entries take 8× the amount of memory to store in a router so would pollute global routing tables even more. Since we cannot expose all the paths by advertising a unique prefix to each connected network, we must find some subset of paths to expose.

Finding that right subset of paths to expose that satisfy performance objectives, however, is hard since there are exponentially many subsets to consider, and since measuring paths takes time. Therefore, we have to predict how different subsets of paths perform which is challenging since interdomain routing is difficult to model, and since there are too many possible dynamic traffic conditions (failures, attacks) over which to assess these predictions.

3 Methodology

3.1 Overview of Solution

- Key challenges
  - Be concrete about the size of the search space
  - Everything takes time
  - There's tons of failure scenarios
  - All the users are correlated
- Key insights
- Key approaches

3.2 Problem Setup and Definitions

3.2.1 Setting and Goals

We aim to advertise relatively few prefixes to connected networks to offer low-latency paths from user networks to the Service Provider subject to capacity constraints on links. We assess resilience based on single-link and single-site failures, but our methodology extends to the multi-link/site failure case and our evaluation shows that solutions that optimize for these metrics naturally offer other benefits (**??**). Adding capacity