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 expanding suite of applications that are increasingly integrated with our lives, but have to contend with an unreliable, unpredictable public Internet to route user traffic. To enhance reliability to dynamic scenarios such as failure and DDoS attacks, large service providers overprovision 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 28% more efficiently than other solutions, reduces congestion on links during site failures by 17% on average, and enables service providers to achieve low latency objectives for up to 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 [35]) and as performance requirements become tighter (*e.g.,* virtual reality requires 10 ms round trip latency [51] and 3 ms jitter [71]). 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 (*e.g.,* 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 [65, 81, 26]. However, it is difficult to do the same on the ingress 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. Current approaches to routing try to achieve low steady state latency, but do not consider whether low latency paths can support dynamic loads (§2.3).

Given this lack of control, resource overload occurs due to link failures [49], DDoS attacks [54, 79, 61], flash crowds [30, 40, 42], and route changes that cause significant traffic shifts [52, 47, 25, 48, 56, 49]. This overload can lead to degraded service for user traffic [53, 59, 36].

Service Providers may respond to resource congestion by (a) proactively overprovisioning resources [72, 1, 44] or (b) reactively draining congested links and moving some of that traffic to other resources with free capacity [49]. However, we demonstrate in Section 2.3 that overprovisioning resources can require overprovisioning rates as high as 70%, significantly increasing expenses.

Service Providers can also drain traffic from overloaded links/sites by *withdrawing* prefix announcements that are also advertised via other healthy links/sites [54, 49, 62], so that traffic destined to those prefixes is forced to arrive on other links/sites after BGP converges (within tens of seconds [86]). However, this solution to overload is error-prone [44], post-hoc, unpredictable, and ineffective (§2.3).

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 minimizes latency subject to capacity constraints both with and without failures as failure requires load balancing traffic on backup