Research Statement

Balakrishnan Chandrasekaran (balac@mpi-inf.mpg.de)

My research focusses on the performance and security aspects of networked systems. The Internet at its inception, circa 1984, was a simple system: The network of networks that constituted it, as well as interactions between applications across the networks, could be traced virtually on the back of an envelope. Over time, however, the Internet has evolved to become an extremely complex, engineered, distributed system. Tracing even simple interactions over the Internet such as fetching of a web page from a browser is a non-trivial exercise: The mental model it conjures up in this case comprising a browser requesting the page, a server responding, and the interaction facilitated by a set of routers and switches connecting the two is far too simplistic. Numerous other entities (e.g., firewalls screening the request, caches attempting to reduce the response time, and clusters of servers sharing the load from similar requests) play a vital role in the interaction.

As the example demonstrates, determining the contribution of the different entities (e.g., browsers, firewalls, caches, and servers) towards the overall performance of applications (e.g., the time it takes to load the page on the browser) is extremely hard. It is, however, a crucial first step in designing better systems. To this end, I pursue an empirical, data-driven approach for analyzing the design, deployment, and operations of networked systems. My research subsequently exploits the insights derived from these empirical analyses to improve the performance and security of such systems. Below, I explain some of my strongest contributions leveraging real, empirical network measurements, highlighting in each a few open questions to hint at the short-term goals. I conclude with an outlook on my long-term research agenda.

Characterizing Server-to-Server Web Traffic

It is infeasible to handle requests for content (e.g., web pages and video streams) from a large number of end users, if that content is served from a small number of servers. Deploying a large fleet of servers in geographically diverse locations (close to end users), keeping them up to date with latest content, redirecting end-users' requests to the nearest server, balancing demand surges, and handling failures are, however, extremely hard problems. Unsurprisingly, content providers typically delegate the design, deployment, and operation of this complex serving infrastructure to a third-party called a *content delivery network (CDN)*. CDNs operate a massively distributed fleet of servers that act as a logically centralized intermediary between the end users and the content providers. CDNs provide fast, reliable, and secure delivery of content to end users. If a request can be served, for instance, from the cache of a CDN server close to the user, instead of a far away content provider's server, the response time of that request can be drastically reduced.

CDNs are one of the crucial components of the Internet's infrastructure, since the majority of the Internet traffic today is delivered via content delivery networks (CDNs). The ever increasing end-user demands for bandwidth and latency seem to assure that the volume of content delivered through CDNs will only increase going forward. Typically, the CDN hauls data from origins (i.e., content providers) to its *back-end* servers, moves this data (over a sophisticated overlay network) to its *front-end* servers, and serves the data from there to the end users. If we focus on the path traversed by content from origins to end users via a CDN a simple fact becomes apparent: a significant fraction of this path traverses the CDN infrastructure, between the back-end and front-end servers, and the performance of this server-to-server path has implications for end users.

As a first step towards understanding the importance of the server-to-server path, I collaborated with researchers from Akamai Technologies (one of the largest CDNs in the World) and Technische Universität Berlin on estimating the volume of web traffic traversing this path. While prior work had analyzed the traffic between end users and (front-end) web servers, we were the first to provide a characterization of the back-office web traffic, i.e., web traffic between the front-end servers and back-end as well as the origin servers [7]. The work revealed that delivering content to the end users involves a rich and complex interaction between many servers. Advertisement objects, for instance, result from an online bidding process involving several servers before these objects are delivered to the end users. Web objects served by CDNs similarly may involve interactions across a hierarchy of servers before being delivered to end users. Such server-to-server interactions, our study revealed, amount to a significant fraction of the core Internet traffic. The

study also revealed that the percentage of back-office web traffic varies significantly across different locations and that a non-trivial fraction of this traffic is invisible to the public Internet (i.e., infeasible to measure for a third party without access to the CDNs).

The above observations highlight the need for a more systematic approach for measuring and characterizing the performance of content delivery in the Internet. Techniques for reducing latency in delivering content, for instance, would be very remiss without an accurate analysis of the server-to-server interactions. Understanding what affects govern a performance metric (say, the loading time of a web page on a browser), and evaluating an optimization to improve that metric require a deeper understanding of CDNs, or, more generally, the server-to-server interactions. Furthermore, generating a traffic matrix that succinctly captures the dynamics of the server-to-server traffic over time and across different geographies will be immensely useful to the networking community. It could, for instance, inform the design and evaluation of new class of network protocols tailored for the server-to-server path.

Measuring Performance of the Internet's Core

Server-to-server interactions play a vital role in the delivery of content to end users. Since the front-end servers of a CDN, at any geographic location, may cater to hundreds of thousands of users or more, performance (e.g., latency and congestion) of the server-to-server paths presumably has huge implications for end users. To evaluate the factors that affect the end-to-end latency of server-to-server paths, I conducted a large-scale measurement study in collaboration with researchers from Akamai, MIT, and the Center for Applied Internet Data Analysis (CAIDA). We gathered more than a billion measurements (i.e., traceroutes and pings) over both IPv4 and IPv6 protocols between hundreds of CDN servers, located in diverse geographic locations and networks, across a period of 16 months. Additionally, by carefully choosing the servers from a CDN infrastructure, to represent a diverse mix from servers from different networks and geographies, we used the server-to-server paths as a proxy to measure the performance of the Internet's core (i.e., end-to-end latency of paths across the Internet's core).

Our measurement study revealed that, for the most part, server-to-server paths experienced only a few routing changes and these changes, typically, did not significantly increase the end-to-end round-trip times (RTTs) [4]. Network congestion in the Internet's core (for which the server-to-server paths serve as a proxy) will increase the end-to-end latencies and adversely impact the end-user experiences. We discovered, however, that *consistent congestion*—defined as recurring daily oscillations in round-trip times—are *not* the norm in the core. Between congestion and (network-level or AS-level) routing changes, our empirical analyses suggested that the latter typically increases the end-to-end latencies more than the former.

Although our (consistent) congestion inferences are based on well-studied, widely used delay-based methods, the caveats of such delay-based inferences are also well-known. We could exploit, nevertheless, the widespread adoption of explicit signals (i.e., explicit congestion notification or ECN), and design a large-scale measurement study to generate an Internet-wide congestion "heat map." The heat map will inform policies regarding how networks peer and how we perform traffic engineering in the different networks, and the "hot spots" will highlight areas that need further optimization. Identifying whether congestion (or, more generally, any performance metric) over IPv4 and IPv6 paths are correlated will also provide key insights into the Internet's infrastructure. We can estimate, for instance, the extent to which network paths over the IPv4 and IPv6 protocols share the underlying infrastructure, and the insights will immensely benefit both researchers and network operators.

Rethinking Transport for Video Streaming

Video traffic constitutes the majority of the Internet's traffic. While the Internet's infrastructure, the communication protocols, and software stacks have undergone significant changes, the way videos are streamed over the Internet—using reliable transport protocols, such as TCP—remains virtually unchanged. Along with researchers at the University of Massachusetts Amherst and Akamai Technologies, I am questioning the status quo in video streaming.

Today, TCP is the dominant transport protocol for video streaming, due to the widespread use of dynamic adaptive streaming over HTTP (DASH). The rich body of prior work on optimizing TCP, adaptive bitrate selection (ABR) algorithms, or TCP variants, however, highlights TCP's shortcomings. In our preliminary investigation, we found that, when streaming using TCP, even at a loss rate of 0.16%—lower than that typically observed in the Internet—the video player spends 20% of the total video time in simply waiting (i.e., *stalls*) for new content to arrive and resume playback [6].

Studies indicate that stalls significantly degrade end-users' video-streaming experiences.

A simple observation highlights that reliable transports are ill-suited for video streaming: Video data consists of different types of frames, some types of which do *not* require reliable delivery. The loss of some types of frames has minimal or no impact (since such losses can be recovered) on the end-user's quality of experience (QoE). Our approach is, therefore, to design a transport protocol that offers partial or selective reliability. Rather than designing such a transport from scratch, however, we are working on extending QUIC, a recent transport protocol from Google that is easier to extend or modify compared to TCP. Popular web browsers, content-provider servers, and CDNs also support QUIC. We intend to develop a transport protocol that builds atop QUIC and and involves a backward compatible, incrementally deployable extension to support partial reliability [6].

The design of a transport protocol to support partial reliability opens up interesting avenues of research. We are examining how the applications could indicate to the underlying transport which pieces of the data require reliability. Whether the interface to facilitate this interaction between the application layer and the transport layer benefits other applications beyond video streaming remains unexplored. Our proposal also hints at the need for a cross-layer optimization, rather than focussing on improving transport or the application layers independently. With increasingly more events (e.g., Felix Baumgartner's space jump) being streamed "live" over the Internet to millions of users, we plan on investigating the benefits of our novel transport for low-latency live streaming.

Future Directions

The prior empirically grounded studies reveal several paths for further exploration towards improving the performance as well as security of the Internet. Below, I highlight a few initial explorations.

Towards a Speed-of-Light Internet

Reducing latency across the Internet is of immense value to both content providers and end users. Microsoft's Bing, for instance, found that a two second slowdown translated to a drop in revenue per user of 4.3%. Latency is also crucial to facilitate the idea of running software in the cloud. A low-latency network allows more computations to be offloaded to the cloud, while giving the end users an illusion that they are running their computations locally (on their own machines). The Internet, however, is shockingly slow today!

In a joint effort with researchers at Duke University, University of Illinois at Urbana-Champaign, Yale University, University of California, Santa Cruz, and ETH Zürich I quantified the latency "inflation" in the Internet, defined as the ratio of the observed RTT to the theoretical limit—the speed of light in free space [3]. The rich and diverse data sets we used in this work and shared with the community was awarded the "Best Dataset Award" at the 2017 Passive and Active Measurements (PAM) conference held in Sydney, Australia. Our study showed that the median time to fetch just the HTML documents of popular Web sites was 37-times slower than the round-trip speed-of-light latency (between the corresponding clients and servers). We showed that infrastructural improvements alone could reduce latency by a factor of three or higher.

Our preliminary design and evaluation of a speed-of-light (or *cSpeed*) network, using microwaves for communication, confirms that it is economically feasible to build and operate a cSpeed network [2]. While the reliability of a microwave network under adverse weather conditions and varying network traffic demands requires a thorough evaluation, the possibility of a cSpeed network, operating in parallel to the existing (mostly fiber-based) Internet infrastructure, opens up several new avenues of research. One straightforward application is to deliver latency sensitive content over the cSpeed network (which is bandwidth contrained compared to, for instance, an optical fiber network), but it opens up new challenges: How can we reliably determine the latency-sensitive content among the overall traffic? Will accelerating only a subset of the overall traffic improve end-users' experiences? While some of our initial investigations show, for instance, that speeding up the client-server path alone could improve the end-user's browsing experiences (estimated by measuring the page-load times) [2], it is clear if the benefits are generalizable to other applications. The cSpeed network could also be used to carry only control information or even only data concerning failures or issues observed in the regular Internet. Delivering "bad news" fast could improve the recovery time and resilience of the Internet. More generally, the availability of a low-latency network in parallel to the existing Internet could fundamentally affect the design of several distributed applications.

Network-assisted Congestion Control

While the prior work showed that congestion (in the core) does not typically increase the end-to-end path latencies, the few atypical cases result in significant increases in RTTs. Effectively controlling the congestion is, nevertheless, becoming increasingly difficult due to several factors, e.g., availability of very-high-speed links, increases in traffic diversity and burstiness, and decreases in buffer sizes (relative to link speed). Fair sharing of network resources is no longer the only goal of a congestion control mechanism; objectives include, for instance, lowering end-to-end delays, maximizing throughput, and solving incast challenges. Any congestion control design would be remiss not to exploit the recent advancements as well as the adoption of programmable data planes. The design of a *network-assisted* congestion feedback signal and a congestion control mechanism to exploit that signal have far-reaching implications for networked systems.

As the first step in this space, I collaborated with researchers at the Max Planck Institute for Software Systems to sketch the design of a novel network-assisted congestion feedback mechanism [5]. While the idea of eliciting support from the network to improve end-to-end CC schemes is not new, the scope of prior work in this space, however, has been rather narrow: They either restrict themselves to using only a few bits for signaling or to setting explicit rates for senders. Our design, in contrast, elicits hardware support in switches for a few queues and the increasing adoption of programmable data planes (e.g., P4). We also exploit the programmability of today's switches to identify elephant flows (i.e., long flows, which contribute most to traffic volume) and send explicit congestion notifications (NACKs) to the sender of these flows for throttling them. The NACKs, sent directly to the sender, are expected to arrive at the sender faster than prior work on network-assisted feedback (e.g., ECN and NDP) and should significantly reduce the sender's reaction time. The evaluation and realization of the feedback signal constitutes a major part of our future work. Understanding to what extent the identification of mice and elephant flows help in avoiding congestion is also crucial to scope the applicability of the design. The availability of a rich congestion-control feedback signal from the network also has implications for the congestion control algorithsm at the sender: How can we redesign, for instance, the congestion control mechanism to fully exploit the rich congestion signal from the network?

Plugging the Internet's security holes

The Internet was designed with an implicit notion of trust, and as such has been subjected to a broad spectrum of security threats and attacks during the last decade. The unprecedented growth of Internet-of-Things (IoT) devices and the emergence of a connected *Internet of Everything*, coupled with the fragility of the IoT ecosystem, provides a ripe platform for inimical parties to exploit the weaknesses and launch massive DDoS attacks. Although several point solutions have been proposed, they offer, if any, only a veneer of security, and we are in a dire need of a more distributed and concerted approach to secure the Internet [1].

The increasing sophistication of DDoS attacks (with some, for instance, intelligently regulating attack-traffic's volume to evade detection) hints at several feasible, but largely unexplored, approaches: Could we exploit the network and geographic diversity of a highly distributed CDN infrastructure, for instance, to detect botnet-based DDoS attacks well before the attack-traffic volume ramps up? While the exchange of attack signatures between the stakeholders, e.g., between Internet service providers and CDNs, and between CDNs and Internet exchange points, could immensely help in detecting and mitigating large-scale DDoS attacks, the design of an interface to facilitate such an exchange has remained largely unexplored. It is also unclear whether the detection of DDoS attacks could be performed in a distributed manner, rather than requiring traffic to be funneled to a centralized location for analyses, as is typically the case today.

Today, most DDoS mitigation solutions rely on anycast to ingest traffic from end users to the "nearest" datacenter where it is "scrubbed" to allow only legitimate traffic to proceed forward towards its intended destination. Several questions concerning such a design remain unanswered: How quickly can an anycast-based load balancing scheme react to a quick surge in traffic, as in the case of a typical DDoS attack? The scrubbing centers may, however, turn into "choke points" when attack volumes increase beyond a few hundred Gigabits per second. Although deploying more datacenters will delay the onset of such issues, the design of a more scalable system remains unexplored.

References

[1] T. Benson and B. Chandrasekaran. Sounding the Bell for Improving Internet (of Things) Security. In *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy*, IoTS&P '17, pages 77–82, New York, NY, USA, 2017. ACM.

[2] D. Bhattacherjee, S. A. Jyothi, I. N. Bozkurt, M. Tirmazi, W. Aqeel, A. Aguirre, B. Chandrasekaran, P. B. Godfrey, G. Laughlin, B. M. Maggs, and A. Singla. cISP: A Speed-of-Light Internet Service Provider. *CoRR*, abs/1809.10897, 2018.

[3] I. N. Bozkurt, A. Aguirre, B. Chandrasekaran, P. B. Godfrey, G. Laughlin, B. Maggs, and A. Singla. Why Is the Internet so Slow?! In M. A. Kaafar, S. Uhlig, and J. Amann, editors, *Passive and Active Measurement: 18th International Conference, PAM 2017, Sydney, NSW, Australia, March 30-31, 2017, Proceedings*, pages 173–187, Cham, 2017. Springer International Publishing.

[4] B. Chandrasekaran, G. Smaragdakis, A. Berger, M. Luckie, and K.-C. Ng. A Server-to-Server View of the Internet. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '15, pages 40:1–40:13, New York, NY, USA, December 2015. ACM.

[5] A. Feldmann, B. Chandrasekaran, S. Fathalli, and E. N. Weyulu. P4-enabled Network-assisted Congestion Feedback: A Case for NACKs. *Stanford Workshop on Buffer Sizing (BS)*, 2019.

[6] M. Palmer, T. Krüger, B. Chandrasekaran, and A. Feldmann. The QUIC Fix for Optimal Video Streaming. In *Proceedings of the Workshop on the Evolution, Performance, and Interoperability of QUIC*, EPIQ'18, pages 43–49, New York, NY, USA, 2018. ACM.

[7] E. Pujol, P. Richter, B. Chandrasekaran, G. Smaragdakis, A. Feldmann, B. M. Maggs, and K.-C. Ng. Back-Office Web Traffic on The Internet. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, IMC '14, pages 257–270, New York, NY, USA, 2014. ACM.