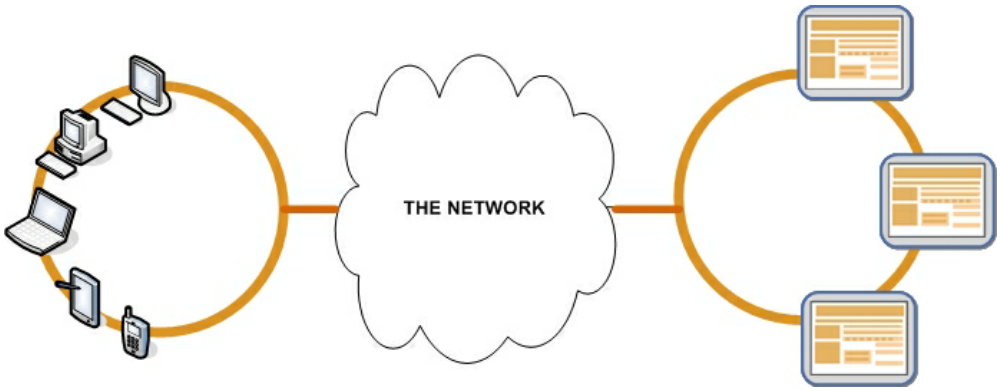
The Shortest Distance Between Two Points

Lori MacVittie, 2008-05-05

The shortest distance between two points is, according to geometry, a straight line. Unfortunately for everyone, there's no way we can hope to physically have a straight line between us and any server - unless we're in the data center troubleshooting a problem. Whether it's because of physical distance limitations, location, or the device we're using, there's bound to be many points along the path between our client and any given server.



But when we diagram network architectures we often obscure in a cloud the actual representation of the network - either because there's just too many devices (the Internet) or we consider the actual architecture irrelevant. But an accurate representation of "the network" can be important in unraveling application performance issues.

Most of us understand that there is a cost associated with every "hop" in the network, where every "hop" includes every device that data flows through on its way from the server to the client. In college we learned about Dijkstra's algorithm, which taught us how to find the shortest path through an interconnected set of points (hops). From that, we learned the basics of routing algorithms - whether we knew it or not at the time.

Every hop adds latency (cost), period. It may be so minute that it's nearly unmeasurable, but it's there if for no other reason than the requirement that data must travel a distance to get to the device, and traveling costs ***something***, no matter how small a time interval that ***something*** may be. It stands to reason then that the more hops in a path the higher the cost of traversing that path.

The goal of application acceleration, then, is to reduce the cost associated with traversing a particular path such that data is delivered as fast as possible to the client. We want "the network" to act as close to a "straight line" in terms of performance as one can get, and the best way to do that is to reduce the amount of data traversing the path. Less data means mean time to travel and less time to process, making the total time to deliver data from application to client, well, less.

There are several ways in which application acceleration solutions can reduce the amount of data being exchanged:

> **1. Don't deliver it at all**
>
> In many cases, data already exists on the client, particularly web clients (browsers). By making better use of client-side caching, application acceleration reduces the amount of data traversing the network.
>
> **2. Compress It**
>
> Most applications exchange data in text formats - HTML, JavaScript, XML, JSON - and can therefore benefit from compression. Compression reduces the total amount of data that needs to traverse the network. But be careful! Compression benefits begin to reduce dramatically as the amount of data being transfered reduces in size.
>
> **3. Deduplication**
>
> Another way to reduce the amount of data being exchanged is to only send data that is new; that is, remove data that has already been transfered and replace it with a tiny marker instead. This reduction is typically how symmetric acceleration products work.

Another way that application acceleration can improve performance is to reduce the latency introduced by individual hops along the path. An application acceleration solution accomplishes this by performing what we often refer to as "offloading" tasks from other devices or, more typically, the servers. It is almost unilaterally true that the web or application server will introduce the largest amount of latency into the path and therefore solutions attempt to reduce that latency by taking on some of the functions normally associated with web/application servers. Some of these functions are:

> **1. SSL processing**
>
> By offloading SSL processing and bulk encryption/decryption to a hardware-acceleration solution, the total time required to process SSL is reduced.
>
> **2. Compression**
>
> Compression sits in both camps (reduction of data and offloading) because compression can be - and often is - a function of the web/application server. But it is often the case that an application acceleration solution can perform the task faster and much more efficiently than the server.
>
> **3. TCP Session Management**
>
> Managing TCP connections takes time, and application acceleration solutions offload much of that time by reducing the total number of connections required to the server and reusing them as often as possible. This makes the process of exchanging data between the server and the application acceleration solution much more efficient and reduces the total time required to deliver data to the client.

We are unlikely to ever be in a situation where there is a straight line between the client and the server, which means multiple hops must be traversed in order to deliver data (i.e. applications) to clients. Because we know that every hop has a cost, one of the best ways to reduce the cost associated with those hops is to reduce the amount of data being exchanged and let the most efficient "hop" in the network perform specific tasks.

And while it won't be the ***shortest*** path through the "network", it may well be the "fastest". And when it comes to your users you can be sure they aren't counting hops, they're counting time.

Imbibing: Coffee

Technorati tags: MacVittie, F5, application delivery, application acceleration, acceleration, web acceleration

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc.	F5 Networks	F5 Networks Ltd.	F5 Networks
Corporate Headquarters	Asia-Pacific	Europe/Middle-East/Africa	Japan K.K.
info@f5.com	apacinfo@f5.com	emeainfo@f5.com	f5j-info@f5.com