The third greatest (useful) hack in the history of the Web

Lori MacVittie, 2008-08-09

Developers have an almost supernatural ability to workaround restrictions, even though some of the restrictions on building applications delivered via the web have been akin to a kryptonite. Like Superman fighting through the debilitating effects of the imaginary mineral, they've gotten around those restrictions by coming up with ways to implement functionality and improve the behavior of browsers and thus web applications anyway.

The first greatest hack was giving HTTP state. The second? Cookie-based persistence. The third? The CNAME trick.

THE PROBLEM

The reason the "CNAME trick" came about was a limitation on browser connections to a single host imposed by

browsers, but particularly version of Internet Explorer previous to IE8. With only 2 connections per host name allowed and many times that number of objects on a page, the ability of IE in particular but really all browsers to quickly retrieve all those objects and render them was also hampered. This resulted in the appearance that the application performed poorly, when in reality it wasn't the application but the inherent delivery mechanisms that were slow due to limitations beyond the user's, the network admin's, and the developer's control.

Users, of course, don't care about any of this. All they know is that the application they are using is slow and they want it fast. And when some of those users are corporate business users, the developers are going to hear about it because the help desk is going to call *them* when they get barraged with complaints from users. This is the real reason developers develop nearly supernatural powers of hacking; they'll do anything to stop users from complaining.

THE HACK

Developers all over (including ours inside F5, working on building our application acceleration solution, WebAccelerator) figured that if the browser was going to limit the number of connections to a single host that the answer was simply to trick the browser into thinking it was talking to more than one host. Turns out doing this is rather trivial: simply add multiple CNAMEs for the same host to DNS, and then reference those as the host for some of the objects in the page. So *www.example.com* becomes *www1, www2, www3,* and so on.

This required changes to the application so that the additional host names were referenced, unless you made use of a proxy-based solution like WebAccelerator and BIG-IP Local Traffic Manager capable of rewriting outbound host names and virtualizing them to appear to the outside world as if they were a single host.

THE NEW PROBLEMS

This improved application performance, but at the cost of increasing the number of simultaneous connections to the server. This was "bad" in the sense that a web server, even well tuned, can only support X simultaneous connections at a time, and if each user was consuming Y connections per page, the number of concurrent users that could be supported on a single server was decreased by this hack. This made servers less efficient and required additional servers to ensure availability and scale.

Along comes AJAX, and the popularity of the "CNAME trick" rose rapidly. This is because AJAX became a way to provide near real-time updates to web browsers on a per-component basis. The result of this is a rich, interactive application that ends up maintaining a connection to the server on a nearly continual basis. So not only is a single application using *more* connections - and thus server resources - to load a page, it is also consuming more resources and connections throughout its execution.

Some web servers aren't good about dealing with virtual hosts. Rather than pretend that a single instance is all the virtual hosts, it will spawn more and more children instances, each one consuming resources until there are so many instances running that each one can handle fewer concurrent users than the parent.

The CNAME trick is also difficult to scale. Every time a new CNAME is added, it must be referenced within the application, which often means modifying applications. A costly proposition in terms of time and effort. If the CNAME trick is used as a reactive measure to address poor application performance, the entire application must be modified to reference the new host names.

THE SOLUTIONS

There are several solutions to the problems created by the CNAME trick, but they all take advantage of the same core principle: a proxy-based mediator. The proxy's job is to mediate for the client and aggregate connection requests to the server and make them more efficient, either by reusing connections to servers or by load-balancing them more efficiently, or by rewriting requests.

The advantage of implementing a proxy-based solution proactively is that applications don't necessarily need to be modified if the solution can rewrite host names in outbound responses. Basically, if the proxy is smart enough, it can change the host names in the page before it reaches the browser, and then aggregate and optimize them as the requests for each object come back in, essentially taking advantage of layer 7 switching to route all the requests to the appropriate web server.

The problem of additional connections can't be solved without a proxy of some kind. How efficient a web server is at handling the additional connections can be changed with tuning and optimization, but the additional connections and resulting consequences are going to hit that web server regardless without a mediator to deal with them.

THE FUTURE

This is a great hack, there is no doubt about it. It's one of the best ways to "workaround" the problems with browser limitations. Now that IE8 is increasing the connection limit from 2 to 6, the CNAME hack will be less necessary to combat the perception of poor application performance. But the problem of inefficiency and resource consumption on servers will not go away; in fact, it's likely to get worse as IE8 is adopted over the next year.

The increase in connections initiated from browsers will continue to strain the application infrastructure and, in fact, will get ***worse*** primarily because not everyone implemented the "CNAME trick" and thus not everyone's infrastructure felt the strain of those increased connections. With IE8 everyone will feel the impact of additional connections upon their application infrastructure - whether they're a small to medium business, a large enterprise, or a service provider.

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

F5 Networks, Inc.
Corporate Headquarters
info@f5.com

F5 Networks
Asia-Pacific
apacinfo@f5.com

F5 Networks Ltd.
Europe/Middle-East/Africa
emeainfo@f5.com

F5 Networks
Japan K.K.
f5j-info@f5.com

©2016 F5 Networks, Inc. All rights reserved. F5, F5 Networks, and the F5 logo are trademarks of F5 Networks, Inc. in the U.S. and in certain other countries. Other F5 trademarks are identified at f5.com. Any other products, services, or company names referenced herein may be trademarks of their respective owners with no endorsement or affiliation, express or implied, claimed by F5. CS04-00015 0113