F5 Friday: How to Create Your Own URL Shortener

Lori MacVittie, 2010-13-08

Network-side scripting and really big, really fast tables let you implement your own (controllable) URL shortening service

We all use URL shorteners to share links, especially via Twitter and other space-constrained communications channels. At the same time, we're leery of clicking on a short URL that comes from someone we don't know well enough to trust implicitly. And unless the service you're using to exchange thoughts automatically applies a URL shortening service to any links contained within your message, you're likely creating those short URLs by hand.

We love to hate them and we hate to love them. But it is what it is, and what it is is both useful and somewhat risky. Basically there's three core issues with leveraging URL shortening services:

1. Unless you've got a developer on hand (and even sometimes if you do) external URL shortening services require manual creation
2. Most services don't allow "custom" domains, i.e. allow you to use your domain and simply shorten the URI. Those that do (bit.ly for example) require changes to your infrastructure (specifically DNS entries)
3. Shortened URLs shared via traditional services are often suspect because these services have been used to "hide" the destination. The malicious use of short URLs engenders suspicion with many and a refusal to investigate on the off-chance the destination is a malware laden site or something NSFW (Not Safe For Work).

And yet sharing URLs becomes increasingly tedious the longer the URL is. Really, just because you can use several thousand characters doesn't mean you should. Thus URL shorteners, despite their shortcomings, have become the method du jour for turning long URLs into easily consumed, sharable tidbits. We hate to love them, we love to hate them. We're addicted to short URLs.

To address the shortcomings, wouldn't it be nice if you could maintain your own domain and still shorten those URLs? And wouldn't it be even nicer if that meant you could actually gather usage statistics about that URL? While bit.ly's "pro" service allows the former, it's still amazingly naive immature in the reporting department, and it's nigh-unto-impossible to extract that data any way but manually.

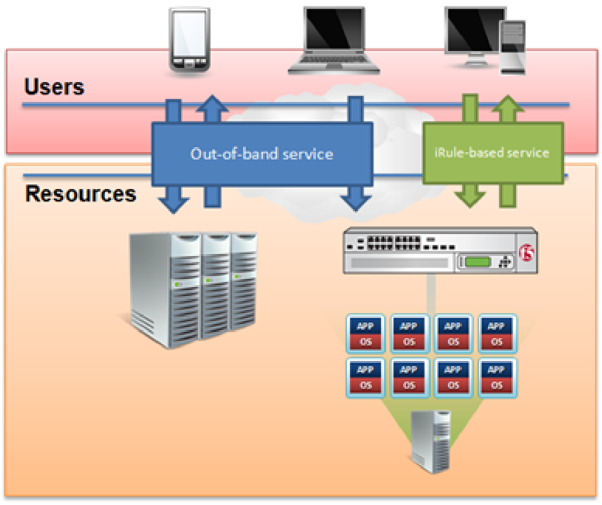
Finally, wouldn't it be nice if you could integrate the shortening process in a dynamic way rather than always creating them manually?

Have I got a deal for you…

iRULE CUSTOM URL SHORTENER

I talk a lot about network-side scripting as an agile method of well, manipulating application requests and data on-demand. From inbound inspection to outbound rewriting, network-side scripting is the realization of one of the foundational dynamic datacenter components: dynamic infrastructure. Providing real-time interaction with requests and responses traversing an intelligent intermediary means devops, infosec, developers, and network teams have the tools with which they can address a variety of obstacles and pain-points.

In this case, it's adding business value and increasing visibility; maintaining control and ensuring the integrity of links shared for whatever the reason. It also allows the ability to better discern from where and whom links are being picked up. It's real-time campaign tracking.



The core value here though is two-fold: (1) you maintain control and (2) you use your own domain to provide some measure of integrity assurance to those you're sharing the links with. The secondary and tertiary benefits are in having a way to track business and marketing campaigns.

An immediate question should be (it was for me) "what about performance?" Just how large can a table containing a mapping of short URIs to long URIs get before it starts to impede performance? This is essentially a proxy solution, so every microsecond it takes to look up the short URI and replace it with a long URI adds to the response time of a request. Well, the bonus if you're using BIG-IP LTM and an iRule is that the functionality is taking advantage of the core platform session table which, if you know a thing or two about networking, absolutely must be high-speed, high-performance in its ability to perform lookups because it can grow to billions of entries in high-traffic situations. So the answer from the experts to my question was, "Giant. Huge. Ginormous."

The second bonus is that you don't necessarily have to do a redirect, which adds to the overall response time. With out-of-band URL shortening services the request goes to a third-party proxy, is translated, and a redirect to the original is returned to the user. Then the user's browser automatically makes a second request and gets the content they wanted. With an integrated, full-proxy iRule-based solution the redirect isn't strictly necessary. While you can still use that same method, it would be much more efficient to simply look up the short URI, grab the full URI, and then simply replace the requested URI with the real one and send it on to the server. You're eliminating time on the wire between the third-party service and the user completely, and the associated TCP-session setup/teardown time which we know is rather expensive in terms of time and resources. You can still do a redirect if you want to, but it's completely unnecessary unless, of course, you're planning on offering the capability as an out-of-band service to your customers.

So by using an iRule you can improve performance, increase visibility, and provide some measure of integrity assurance while you're out there sharing links with whomever you're sharing them with. Additionally, it's just a darn cool use of iRules that has a lot of potential to be modified and used for other situations in which URI mapping might be useful.

And of course it happens to be the case that DevCentral's newest cohort, George Watkins, has written up an iRule to handle URI shortening. iRule wizard Colin Walker helped optimize the rule, so it ought to be a very efficient little iRule. Go ahead and give George's URI shortening iRule a look-see and try it out. If you don't have a BIG-IP yourself, then go ahead and get one – iRules are a part of the core TMOS platform upon which BIG-IP products and modules are based, so the VE (Virtual Edition) of BIG-IP LTM has everything you need to deploy the iRule and take it for a spin.

NOTE: George's version of the iRule is based on an out-of-band service model. Using HTTP::uri instead of HTTP::redirect for the URL will change the behavior and eliminate the overhead of the redirect, but don't forget to assign the iRule to the appropriate VIP. It is also a manual create, but there's no reason you could not integrate the iRule functionality into the response processing and rewrite all URIs in a page to be small URLs automatically – or just any URL with a length greater than .

Happy coding!

Related Posts

- All F5 Friday Entries on DevCentral
- All About iRules

from tag iRules

- F5 Friday: Eavesdropping on Availability
- Defeating Attacks Easier Than Detecting Them
- F5 Friday: An On-Demand Turing Test
- Out, Damn'd Bot! Out, I Say!
- F5 Friday: A Network Heatwave That's Good For Operations
- No Shirt, No Shoes, No HTTP Service

- No Shirt, No Shoes, No HTTP Service
- Is Vendor Lock-In Really a Bad Thing?
- AJAX and Network-Side Scripting
- Automatically Removing Cookies

from tag performance

- The Great Client-Server Architecture Myth
- IE8: Robbing Peter to pay Paul
- Your Network is Not My Network

(more..)

del.icio.us Tags: MacVittie,F5,F5 Friday,George Watkins,Colin Walker,iRules,URL shortener,bit.ly,performance

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