AJAX and Network-Side Scripting

Lori MacVittie, 2009-16-09

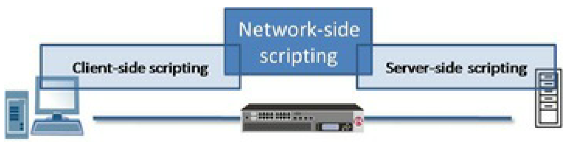
AJAX enables the use of network-side scripting enabled application delivery solutions to offload client-side functionality and improve capacity and performance of dynamic (Web 2.0/AJAX) applications.

In the last couple of weeks I've embarked on a home project to rewrite – from scratch – a couple of web applications that Don and I and friends use on a regular basis. Consider it a very restricted (in terms of users) social networking application, because that's basically what it is. I made heavy use of AJAX for one component in the past version but have been really leveraging it a lot more in the new version because it just makes life so much easier – and the app is much more interesting because of it.

Because we also leverage the application delivery controller (ADC) we have for development and testing it's natural that eventually I'd figure out that if you're using AJAX-enabled applications and have a network-side scripting capable ADC that you could put the two together to introduce some interesting capabilities.

CLIENT-SIDE versus SERVER-SIDE

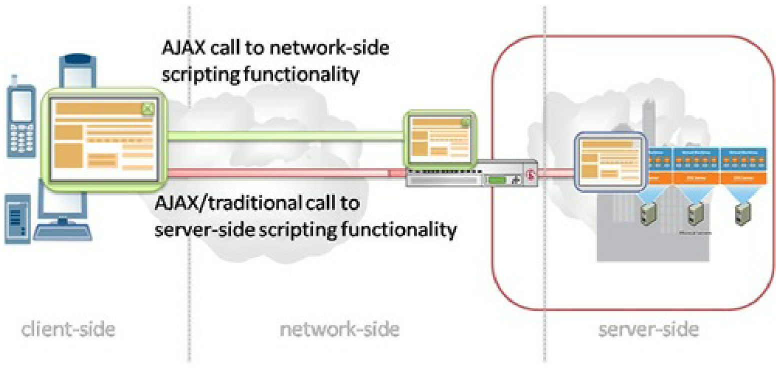
The reason this is coolness is that while not impossible it was never really practical before AJAX to leverage the programmatic capabilities of an ADC from the *client* side. Introducing it into the flow of responses and using it to manipulate response data seemed like the best way to add value to applications. Manipulating the response either to rewrite URIs or content or add DLP (data leak protection) is a pretty common way to take advantage of any scripting capable ADC. But it doesn't really lend itself to offload or service-oriented / RESTful applications because the request was generally always sent to the server and only after it returned a response did the ADC get down to business.
The reason is that in most web applications if you were going to offload application layer functionality from the server to the ADC you'd have to basically turn the ADC into a mini-web server. That was inefficient and honestly, network-side scripting languages are far more restrictive than a full programmatic environment like PHP or ASP. They have to be, they're focused on provided specific snippets of functionality and while I've seen some amazing network-side scripting rules that are basically full-fledged applications, an ADC wasn't designed to ***serve*** applications, it was designed to ***deliver*** applications. It can do the former, but it really excels when doing the latter.



But AJAX enables developers to break down applications into discrete chunks of application functionality some of which are perfectly suited to running in the network-side scripting environment instead of the server. Additionally there are some functions available in network-side scripting environments that provide information about users and requests that is very hard and often compute intense to retrieve on the server-side. Because of the position in the network and the application flow, the ADC appears to the client (the browser) as the same domain as the application (indeed in most cases where an ADC is deployed it *is* the domain for all intents and purposes as it virtualized the domain and mediates for all servers making up the domain). This means that it is completely feasible to make an AJAX-based call *to the ADC* instead of the "application", making it possible to take advantage of functionality and information it was not easy or possible to retrieve in the past.

THE ADC as a SERVICE

What we're really talking about doing is turning the ADC via its network-side scripting capabilities into a service; into an endpoint capable of receiving, processing, and responding to AJAX-based queries for information.


AJAX call to network-side scripting functionality
AJAX/traditional call to server-side scripting functionality
client-side
network-side
server-side

Obviously the functionality available to you on the network-side scripting platform is determined by the ADC, so YMMV depending on what ADC you are using / is available to you.

For example, I implemented a fairly simple encryption "service" using iRules. Both the encryption and decryption functionality is executed in the network-side scripting environment on the ADC – in this case a BIG-IP.

You can play with it here – unfortunately including AJAX in this blog would violate security rules regarding cross-site requests. The actual example is not AJAX, either, but just calls the URI of the network-side scripting service to illustrate the use of a network-side scripting service is the same as any other URI. Thus, turning this into an AJAX call in client-side code would be simplicity. Another example would be the die rolling functionality we built in this blog post. Such functionality is easily incorporated into an application, if you're for some reason in need of generating numbers by rolling polyhedral dice.

NOTE: *The exception to this would be if you're using toolkits like XAJAX that generate the client-side code for you. In most cases the network-side scripting environment can't execute the code (you can't deploy them on the ADC) and thus it isn't possible to use them easily with this scenario. Client-side (JavaScript) libraries that simplify AJAX, however, should be able to make use of network-side services with the same measure of alacrity with which they can leverage server-side services.*

What might be really interesting is the breadth and depth of functionality available on a network-side scripting platform that isn't available – or easily obtained – on the server-side. The ADC, acting as an intermediary or broker between the client and server(s), has available information regarding *both* sides of the network and application layer stacks. This makes services hosted on the ADC platform ideal for debugging, for providing back-channel authentication services, and for exchanging a variety of TCP and IP-based information. The possibilities are really limited by the developer's creativity and driven by necessity. The platform is not the limiting factor in most cases.

THE BENEFITS

The biggest benefit of using network-side scripting with AJAX in application development is that it opens up new functionality and possibilities. There is a lot of information available on an ADC you won't get elsewhere in the environment. But it's really hard to justify an investment based on "enabling innovation", I'm sure.
Tangible benefits would include offloading of functionality from servers, which in turn increases capacity of servers and can improve performance. Fiscal benefits are more likely realized in a cloud environment where the result of offloading is a need for fewer instances, which directly translates into cost savings. Even in a private cloud environment, in which chargeback is used as a means to spread costs across multiple applications, projects, and departments, this can result in a reduction in operating expenses for the application in question.

There's also benefits both in reducing time to deploy through reuse of services, although SOA and other reuse-enabling architectures have failed to really produce these benefits mostly because, well, developers are stubborn and often fail to use existing services because (a) they don't know about them or (b) they don't trust them.

But the benefits in terms of improving capacity and performance are hopefully enough to at least investigate the option of leveraging the network-side scripting capabilities inherent in most ADCs today. Go ahead and try it, just for fun. You might be surprised at the different solutions you can come up with.

You can start your investigation by perusing the examples in DevCentral's iRules CodeShare. These are F5 specific, but should give you a generalized idea of the kinds of functionality you can implement using network-side scripting in general.

- Your load balancer wants to take a level of fighter and wizard
- Understanding ***network-side*** scripting
- If Your Users See an HTTP Error Code You’re Doing It Wrong
- I Can Has UR .htaccess File
- Automatically Removing Cookies
- Clickjacking Protection Using X-FRAME-OPTIONS Available for Firefox
- What is server offload and why do I need it?

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