Web Packaging and CDNs

Mark Nottingham

There's been a considerable amount of discussion about the Web Packaging proposal and its relationship to Content Delivery Networks. This paper is an attempt to contextualise that from the viewpoint of someone who has worked on and with CDNs for over two decades, without commenting on other aspects of Web Packaging (such as its potential effects on publishers and advertising).

What is a CDN?

Content Delivery Networks (CDNs) are used to serve much of the Internet's traffic; some say they have become a critical part of the infrastructure. Large Web sites and other services (particular video) use them to scale traffic out to large audiences with greater end-user perceived performance and higher reliability.

Typically, a CDN has hundreds to thousands of servers running across the globe, mostly serving HTTP traffic that is directed to the appropriate server (as judged by the CDN) using some combination of DNS and anycast.

The scalability and performance gains achieved by CDNs relies heavily on HTTP caching, along with protocol optimisations on the path back to the backend origin server. They often offer additional services such as DDoS protection, bot detection, and request and response rewriting.

Modern CDNs typically serve an entire Web site, including cached "static" content as well as uncacheable "dynamic" content (whose requests are generally written through to the origin server). While some people refer to services like jsDeliver as CDNs, these are just third-party content services that happen to be delivered by CDNs, and thus aren't considered further here.

Is Web Packaging Competitive with CDNs?

One of the primary use cases for Web Packaging appears to be Google's use of it for AMP -- that is, when a user clicks on a search result, a Google-hosted cache serves it, rather than the clicked URL's origin server.

Some have commented that this appears to be competitive with the business model of CDNs; the thinking goes that since search and social interactions generate a significant portion of the Web's traffic, having search engines and social networks host content on behalf of popular content owners will take business away from CDNs.

I don't believe this is the case, for a few reasons.

A CDN acts as an agent of the origin server; it assures that the cache hit rate is as high as possible, for example, and that the service is highly available. It reports back about the traffic it sees and interposes new services on behalf of the origin server. A CDN is providing a service to its customers -- the site owners -- usually based upon a business relationship.

In contrast, a social network or search engine treats the origin's content as the product that they are selling to advertisers; their goal is to procure that product at the smallest possible price. While some CDN customers are advertising-driven and thus at least partially aligned with that goal, many others are not.

As such, I believe that while some CDN customers might produce Web Packages of their content to gain advantages on some platforms, they'll still want to use a CDN, to assure that their content is served efficiently and reliably when the Web Package isn't used, and to maintain as much control of that content as possible.

In some ways, this is similar to the dynamic between "forward" proxies and CDNs[1]. While networks deployed forward proxies extensively in the 1990s, origin servers never fully trusted them; since network operators used them primarily to serve their own interests -- managing their networks -- they could over-cache things, expire things too early (due to being under-sized), or introduce unacceptable latency. This mismatch of incentives spurred the development and rapid success of CDNs, whose interests are more naturally aligned with the origin servers'. Similarly, Web Packaging allows "any old person" to host your content -- whether or not their interests are aligned with yours.

It's true that the total traffic served by CDNs might drop by some small amount as a result of Web Packaging being widely available. However, the same could be said of browser caching, which is seen as something that works in concert with CDNs, rather than a competing technology. As a result, I don't see Web Packaging as directly competitive with CDNs.

[1] Before the wide deployment of HTTPS in the last five years, making forward proxies largely irrelevant.

Does Web Packaging Help CDNs?

Another claim sometimes heard is that CDNs are “very interested” in Web Packaging because they can introduce new services based upon it.

The most apparent such service is creating Web Packages on behalf of customers. However, there is very little unique value being added by a CDN in this case.

Any party that has access to the key material for the site’s TLS certificate can create a Web package; for example, this could easily be done as as-a-service by a startup, using a relatively small amount of infrastructure. The only part of this that is relevant to a CDN is that CDNs typically need access to the key material to serve traffic, so it’s a natural place to perform such a function.

Doing so is certainly not a revolution in CDN services; it’s merely offering a service that can be performed by the origin server itself, if it wants to. This might be convenient for CDN customers (especially those with difficulty managing their serving requirements), but it’s neither a fundamentally new technique for CDNs, nor unique to them.

However, two potential uses of Web Packaging *are* in an area of interest to CDNs - cross-CDN serving, and serving content without keys.

The Cross-CDN Serving Use Case

Section 2.2.5 of the Use Cases and Requirements for Web Packaging explains this use case:

> When a web page has subresources from a different origin, retrieval of those subresources can be optimised if they’re transferred over the same connection as the main resource. If both origins are distributed by the same CDN, in-progress mechanisms like [I-D.ietf-httpbis-http2-secondary-certs] allow the server to use a single connection to send both resources, but if the resource and subresource don’t share a CDN or don’t use a CDN at all, existing mechanisms don’t help.
>
> If the subresource is signed by its publisher, the main resource’s server can forward it to the client.
>
> There are some yet-to-be-solved privacy problems if the client and server want to avoid transferring subresources that are already in the client’s cache: naively telling the server that a resource is already present is a privacy leak.

This use case refers to something exciting to CDNs and other performance-minded Web folks -- reducing the number of connections required for a page load.

While HTTP/2 (and soon, HTTP/3) have helped to reduce the number of connections used by browsers to load a page to one per origin server, the composition of modern Web pages often requires contacting many -- sometimes 50, 100 or more -- origin servers to load a Web page. When this happens, those connections must compete for resources on the path to the browser, and since they're uncoordinated, it often results in congestion events and other suboptimal behaviours.

If a page were to be able to be loaded over a single connection, it would likely result in significant performance improvements, especially on "bad" networks.

However, it's not at all clear that Web Packaging will be able to reduce the number of connections in a page load significantly. While the use case is correct in describing the limitations of Secondary Certificates, it fails to mention the limitations of using Web Packaging -- that the responses need to be cacheable, and the requests' Cookies are stripped.

On the surface, this makes it a poor choice for some of the most common kinds of third party content on the Web, such as advertising. However, it might be that serving the "static" portions of a page load via Web Packages while creating new connections for the "dynamic" parts might help performance incrementally.

Additionally, it's not at all clear that Web Packaging is the best mechanism for solving this problem. In addition to the approach outlined in Secondary Certificates, it's easy to imagine other techniques to reduce the need for creating new connections, or at least for sending so much traffic over them, depending on the specifics of the use case.

As a result, while Web Packaging has some potential for addressing this use case, it needs much more extensive evaluation before being considered a viable solution, and saying that this is a motivator for standardising Web Packaging is premature.

Serving Content Without Keys

Signed Exchanges has some similarity to a recurring wish-list item for CDNs: the ability to serve content, even when not holding the key material associated with the TLS certificate (see also previous discussions on LURK).

CDNs desire this because some customers are reluctant to make key material available to a third party, while still wanting the performance benefits of a CDN. If browsers considered HTTP responses with the Signature header field attached as coming from the associated origin, it would allow a CDN to cache them, addressing this use case.

However, in its current form Web Packaging does not enable this; if a browser visits a CDN that doesn't hold the appropriate TLS certificate and key material, it will simply fail to negotiate a secure connection.

Protocol work could address this shortcoming (e.g., a TLS handshake extension, with fallback to LURK for clients that don't support it). It's not clear if browsers would be interesting in making such changes, though.

Conclusions

In its current form, I believe that Web Packaging is neither directly competitive nor significantly helpful to CDNs; it does not offer significant new capabilities in their quest to improve performance, scalability and reliability to their customers.

That is not to say that it is not useful in other cases; in particular, some sites will use it to improve performance in specific situations (especially pre-loading content from search results and social networking pages).

However, the changes in the guarantees of the Web's security model that Web packaging requires to be useful (often referred to as the "origin substitution problem") are not trivial, and I share others' concerns about it -- even for use cases that might be useful to CDNs.