https://bit.ly/wpack-draft-charter

You should be able to suggest changes to this document just by typing into it. If you want folks to know who suggested the changes, please log in at the top right. I want feedback and proposed changes from anyone interested in Web Packaging at the IETF.

This document falls under the IETF's Note Well. Please read it.

Problem

There are large populations of users who have trouble making direct connections to HTTPS origin servers over the public internet. For example:

- 55% of internet users live in countries where political, social, or religious content was blocked online. (https://freedomhouse.org/sites/default/files/FOTN_2018_Final%20Booklet_11_1_2018.pdf).
- Others use satellite connections with high latency and packet loss. Anecdotally, see https://meyerweb.com/eric/thoughts/2018/08/07/securing-sites-made-them-less-accessible/.
- Users can't fetch websites if they've run out of paid-for data in their mobile plan part-way through a month or if they've aggressively disabled mobile data to make sure other apps don't waste it in the background.

These users currently have to, at best, tell their browsers to pre-fetch sites when they have a cheap real-time connection available or wait until they find such a connection, and at worst, can't browse the content at all.

Even users with highly-available internet connections want to be able to read and interact with web pages as quickly as possible after clicking a link. Needing to make extra connections in the critical path and having multiple connections competing for bandwidth without the ability to prioritize interfere with that goal. Existing attempts to solve this problem achieve the speed but are suboptimal because they allow the referrer to modify the content, require publishers to write content in a new format, and execute the content outside the publisher's usual web origin.

All of these users deserve to know whether the content they're reading or application they're running actually comes from the origin it claims, and the publishers of those origins value their users being able to verify that the content is authentic. Both groups also value the users' ability to store data within websites and web applications (e.g. preferences and content they've created) and have it available the next time they load the same URL.

Scope

1. Define a way to package the public portion of one or more websites or web apps into one or more files that can be distributed and used without a direct connection to the origin server.
2. Define a way to sign these packages such that the result gives the recipient confidence in the integrity and authenticity of what they received. This work will ensure that a client loading a Web Package signed using the initial specification has security and privacy properties that are at least as good as a transferring the contained resources over HTTPS+TLS1.3 from the publishers' origin server(s) except that:
    a. Packages do not guarantee confidentiality, but if the channel that transfers the package provides confidentiality guarantees, packages at most compromise that by revealing information to the original origin server of the resources in the package and by making a TLS1.3-or-later private connection to that origin server.
    b. Packages only guarantee their contents were vouched by the origin servers' keys within the life of the signature, not during the life of the connection(s) that transferred them.
    c. Packages do not provide any of the security "guarantees" of the DNS system.
    d. A package generated for one client can be shared to other clients.
    e. TBD: Are the above sufficient? Can/should we give the WG permission to change this list?
3. Optionally define a way for clients to check that an origin server still vouches for its package in real time.
4. Define a way for an end-user to package their view of a site in order to show a peer what they saw. This could just be an update of RFC2557 (MHTML) but could instead be an un-signed variant of the same format as above. This kind of package will not guarantee integrity or authenticity from the origin server, but might from the end-user who packaged the site.

5. Define the resulting security and privacy model and compare it to the status quo security and privacy model of HTTPS. Search for places the existing web platform depends on the fact that HTTPS provides transport security and describe how those have to change or not given Packaging's addition of object security. This may involve working with W3C groups like WebAppSec and PING.
6. Formats that this group defines should be usable efficiently:
    a. When individual contained resources are either small (~10s of bytes) or large (~GB),
    b. When the total number of contained resources is either small (1) or large (thousands for the initial format with enough extensibility for an update to expand it to millions or billions),
    c. When the total size of all contained resources is either small (~KB) or large (~TB (matching El Paquete Semanal in Cuba) initially, with enough extensibility for an update to expand it to EB), and
    d. When the format is either streamed to the client or loaded from a slow but random-access medium like an SD card.
7. Ensure that any formats or protocols this group defines can be migrated to better cryptography when their original cryptography is broken.
8. As long as it doesn't compromise or delay the above goals, try to:
    a. Support signed statements about the content beyond just assertions that it's the representation of a particular URL. For example, that it appears on a transparency log, that it passed a certain kind of static analysis, that a particular real-world entity vouches for it.
    b. Address the threat model of a website whose frontend might be compromised after a user first uses the site.
    c. Support books being published in the format.
    d. Support long-lived archival storage.
    e. Optimize transport of large numbers of small same-origin resources.
    f. Allow the format to be used in self-extracting executables.
    g. Allow publishers to efficiently combine sub-packages from other publishers.

Out of scope:

1. DRM
2. A way to distribute the private portions of a website. For example, WPACK might define a way to distribute Gmail's application but wouldn't define a way to distribute individual emails without a direct connection to Gmail's origin server.
3. Defining the details of how web browsers load the formats and interact with any protocols we define here. The W3C and/or WHATWG are more appropriate fora for that effort.
4. A way to automatically discover the URL for an accessible package that includes content for a blocked or expensive-to-access URL that the user wants to browse.

Risks of Unwanted Side Effects

This is now in https://tools.ietf.org/html/draft-yasskin-wpack-ecosystem-effects. Comments are welcome in https://github.com/jyasskin/wpack-ecosystem-effects.

The ESCAPE conference was chartered to (among other things) look for any increase in consolidation that might result from standardizing Web Packaging. The known possible effects on centralization and power imbalances, arranged by the type of service provider, and not filtered by benefit, harm, or likelihood are:

General

- The implementation of any new technology is a smaller fraction of a large organization's budget, which pushes toward centralization.

Aggregators

- Aggregators' primary power comes from ranking: telling people that they probably want to visit particular URLs. That's not affected by packaging.
- Aggregators already rank based on sites' content and technology choices. e.g. Google's promotion of HTTPS sites. Packaging can give the aggregator more certainty about the user's experience, which might lead to more intrusive requirements.
  For example, the Google Search Carousel might be able to insist on particular Javascript that could handle swipe gestures where they might not be willing to rely on just having seen such JS on the last crawl. However, packaged Javascript must also be able to reload the page from the origin to deal with retracted content, and that could break reliance on knowing the exact content in the same way.
- Prefetch improves navigation speeds from aggregators that can predict which links users will click. The ability to make that prediction is an economy of scale which encourages centralization. This is likely to have more effect for some kinds of aggregators (search engines?) than others (news streams?).

Browsers

- Packages add pressure to have just one or a few versions of a site's content, which might reduce publishers' willingness to support lots of different browser engines with different features. They'll either settle on the lowest common denominator with some progressive enhancement or target the most popular couple engines, which is likely to be Chromium (Google Chrome, Edge, Brave, Samsung Internet, Opera, UC Browser, etc.) and WebKit (Safari), disadvantaging Gecko (Firefox).
  However, "we only tested in Chrome" is enough of a problem on the online web

that it's not clear how big an additional impact packaging can have.
-

CDNs

- Adding a new kind of distribution might transfer traffic from CDNs to large aggregators, which would reduce CDNs' revenue.
- However, CDNs will still be needed to serve URLs that users type in, bookmark, or navigate to via same-site links, so there's disagreement, even among employees of CDNs, about the likely size of this effect.
- CDNs might be able to acquire even more traffic by offering package caches to let smaller sites take advantage of prefetching.
-

Content Producers

- If Web Packages become an additional format publishers need to produce (https://xkcd.com/927/), that will advantage the larger publishers who can afford the engineering to maintain lots of formats. If instead they replace at least 2 of the existing formats (e.g. AMP, Apple News, Facebook Instant Articles), that'll reduce that advantage of larger publishers and contribute to decentralization.
- If aggregators use packaging to serve a significant fraction of content producers' bytes for free, this reduces the amount the producers need to pay CDNs, which would allow more marginal content producers to stay profitable, increasing diversity.

Other effects not necessarily related to centralization

- When an aggregator prefetches a web package, the static content will load instantly, but ads and other dynamic content will have a visible delay. Personalization might have a delay or might be loaded from local storage effectively instantly. It's not clear what ecosystem effects the changes in loading speed are likely to have.
-

Charter

Background

There is a long history of webpages grouping multiple subresources into a single

combined resource because this allows cross-resource compression and because HTTP/1 made requests expensive. This encouraged the W3C TAG (Technical Architecture Group) to propose a web packaging format based on multipart/* to give web browsers visibility into the substructure of these combined resources. HTTP/2 was expected to make these bundles unnecessary, but that hasn't panned out.

In countries with expensive and/or unreliable mobile data, there is an established practice of sharing content and native applications peer-to-peer using SD cards, WiFi Direct, and similar transmission channels. With the web's move to HTTPS, it became impossible to share web apps over these channels, so a team within Google Chrome began adding an index and origin-trusted signatures to the TAG's packaging format.

The AMP project realized that this new format could solve the problem that pages served by the AMP cache got the wrong URLs. This eventually led to Google Chrome shipping an evolution of the format and Google Search using it in search results.

WPACK

The WPACK working group will develop a specification for a web packaging format that efficiently bundles multiple HTTP resources. It will also specify a way to optionally sign these resources such that a user agent can trust that they came from their claimed web origins.

Key goals for WPACK are:

* Allowing web apps to be safely installed after having been retrieved from a peer.

* Minimizing the latency to load a subresource from a package, whether the package is signed or unsigned, and whether the package is streamed or loaded from random-access storage.

* Being able to smoothly migrate to support future requirements such as random access within subresources, larger numbers of subresources, or avoidance of obsolete cryptography.

* Losing as little security and privacy as practical relative to TLS 1.3 and documenting exactly what is lost and how content authors can compensate. Part of this is analyzing how the shift from transport security to object security changes the security properties of the web's existing features.

* Minimizing the likelihood that the new format increases centralization or power imbalances on the web.

The packaging format will also achieve the following secondary goals as long as they don't compromise or delay the above properties.

* Support more-efficient signing of a single, possibly same-origin HTTP resource.

* Support signed statements about subresources beyond just assertions that they're accurate representations of particular URLs. For example, that they appear on a transparency log, that they have passed a certain kind of static analysis, that a particular real-world entity vouches for them, etc.

* Address the threat model of a website whose frontend might be compromised after a user first uses the site.

* Support books being published in the format.

* Support long-lived archival storage.

* Optimize transport of large numbers of small same-origin resources.

* Allow the format to be used in self-extracting executables.

* Allow publishers to efficiently combine sub-packages from other publishers.

The following potential goals are out of scope under this charter:

* DRM

* A way to distribute the private portions of a website. For example, WPACK might define a way to distribute Gmail's application but wouldn't define a way to distribute individual emails without a direct connection to Gmail's origin server.

* Defining the details of how web browsers load the formats and interact with any protocols we define here. Other standards bodies are more appropriate fora for this.

* A way to automatically discover the URL for an accessible package that includes content for a blocked or expensive-to-access URL that the user wants to browse.

Note that consensus is required both for changes to the current protocol mechanisms and retention of current mechanisms. In particular, because something is in the initial document set (consisting of draft-yasskin-webpackage-use-cases, draft-yasskin-wpack-bundled-exchanges, and draft-yasskin-http-origin-signed-responses) does not imply that there is consensus around the feature or around how it is specified.

Relationship to Other WGs and SDOs

WPACK will work with the W3C and WHATWG to identify the existing security and privacy

models for the web, and to ensure those SDOs can define how this format is used by web browsers.

Milestones

* Chartering + 3 Months: Working group adoption of use cases document
* Chartering + 3 Months: Working group adoption of bundling document
* Chartering + 3 Months: Working group adoption of security analysis document
* Chartering + 3 Months: Working group adoption of privacy analysis document
* Chartering + 3 Months: Working group adoption of signing document
* Chartering + 18 Months: Use cases document to IESG
* Chartering + 18 Months: Bundling document to IESG
* Chartering + 24 Months: Security analysis document to IESG
* Chartering + 24 Months: Privacy analysis document to IESG
* Chartering + 24 Months: Signing document to IESG