Web Packaging Design

WPACK BoF, IETF 106, November 2019

Jeffrey Yasskin

Preliminary!

Outline

1. Package format
2. Origin Trust

Semantics of a package

https://server.example

/path/package1

https://foo.ex/bar.html

en-US

Content-Type: text/html
Content-Language: en-US

<html>
Hello
...

es-419

Content-Type: text/html
Content-Language: es-419

<html>
Hola
...

https://foo.ex/cat

image/jpeg

Content-Type: image/jpeg

text/*

Content-Type: text/plain

```
 |\__/,|   (`\
 |_ _  |.--.) )
 ( T   )     /
(((^_(((/(((_/
```

https://baz.ex/quux.js

Content-Type: text/javascript

export const name = 'quux';
...

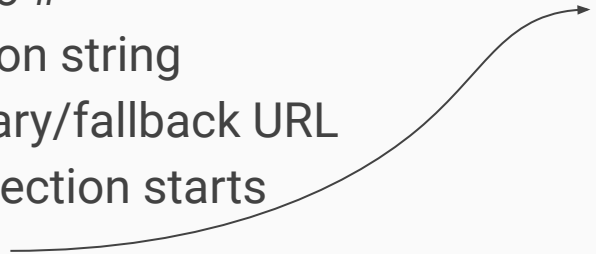
Overall format

CBOR subset

- Invariant block:
  - Magic #
  - Version string
  - Primary/fallback URL
- Table of section starts
- Sections
- Total length

Sections

- Index
- Manifest
- Signatures
- Critical section names
- Responses



Described in draft-yasskin-wpack-bundled-exchanges.

Index section

A map

- URL ⇒
  - Content negotiation info ⇒
    - A range within the responses section.

Responses section

Index points to individual HTTP responses within this section, which are parsed individually.

An HTTP response is represented as a map from header names to values, followed by a body.

Does not represent trailers.

Currently assumes repeated header fields have been combined:
can't represent Set-Cookie.

Manifest section

URL of an App Manifest or other package metadata.

Critical sections section

Which other sections MUST the client understand?

Signatures section

- List of "authorities".
  - Could be X.509 certificates, raw public keys, or something else.
  - An X.509 certificate could represent a domain owner or something else.
  - Includes any certificates needed to build a chain.
- Each signature:
  - Identifies the signing authority.
  - Includes a validity window in time.
  - Identifies a URL that contains newer versions of itself.
  - Covers a particular subset of resources to allow multi-origin packages.
    - Resource *hashes* are signed, currently using MT's MICE.
- No design yet for counter-signatures.

Semantics of untrusted content

- Resource has a package URL and a claimed origin.
- Resource is cross-origin with:
  - TLS resources from the claimed origin.
  - Resources with a different claimed origin in the same package.
  - Unpackaged resources that are same-origin with the package.
  - Packaged resources in different packages served by same origin?

Origin Trust

Origin Trust

Sign the content with a certificate issued in the same way as a server's TLS certificate.

Dangers

Intrinsic

- Off-path attackers
- Vulnerabilities last until signature expiration

Avoidable

- Personalized data
- Mismatched content versions
- User ID transfer (Tracking)

- Transport->object security?

Mitigating the intrinsic dangers

Servers opt into the danger.

- Requires the "CanSignHttpExchanges" X.509 extension.
- Enabled by a CAA record.

Origin-trusted signatures are limited to 7 days.

Avoiding personalized data

Servers: Don't Do That. (With some advice for how)

Supported by a client-enforced block on stateful headers.

Making versions match

Signatures cover a full set of resources.

If the attacker removes one resource from a signed group, it doesn't fall back to the network.

Fetching a resource not mentioned by the package goes to the network.

Blocking user ID transfer

Origin-trusted packages must be requested without credentials.

distributor URL = https://distributor.origin/<function(primary URL)>

Transport → Object security?

Are there more dangers introduced by this shift to object security?

The WG's charter includes looking into this.

Signed Exchanges?

Clarification and Discussion