Dispatch
Internet-Draft
Intended status: Standards Track
Expires: January 1, 2018

J. Yasskin
Google

June 30, 2017

Web Packaging
draft-yasskin-dispatch-web-packaging-00

Abstract

Web Packages provide a way to bundle up groups of web resources to transmit them together. These bundles can then be signed to establish their authenticity.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 1, 2018.

Copyright Notice

Copyright (c) 2017 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction

People would like to use content offline and in other situations where there isn't a direct connection to the server where the content originates. However, it's difficult to distribute and verify the authenticity of applications and content without a connection to the network. The W3C has addressed running applications offline with Service Workers ([ServiceWorkers]), but not the problem of distribution.

We've started work on this problem in <https://github.com/WICG/webpackage>, but we suspect that the IETF may be the right place to standardize the overall format. More details can be found in that repository.

1.1. Use Cases

1.1.1. Offline Installation

People with expensive or intermittent internet connections are used to sharing files via P2P links and shared SD cards. They should be able to install web applications they received this way. Installing a web application requires a TLS-type guarantee that it came from and can use data owned by a particular origin.

1.1.2. Snapshot packages

Verification of the origin of the content isn't always necessary. For example, users currently share screenshots and MHTML documents with their peers, with no guarantee that the shared content is authentic. However, these formats have low fidelity (screenshots) and/or aren't interoperable (MHTML). We'd like an interoperable format that lets both publishers and readers package such content for use in an untrusted mode.

1.1.3. CDNs

CDNs want to re-publish other origins' content so readers can access it more quickly or more privately. Currently, to attribute that content to the original origin, they need the full ability to publish arbitrary content under that origin's name. There should be a way to let them attribute only the exact content that the original origin published.

1.1.4. ...

1.2. Why not ZIP?

WICG/webpackage#45 [1]

1.3. The Need for Standardization

Publishers and readers should be able to generate a package once, and have it usable by all browsers.

1.4. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Format

2.1. Mode of specification

This specification defines how conformant web package parsers convert a sequence of bytes into the semantics of a web package. It does not constrain how web package encoders produce such a package: although there are some guidelines in Section 3, encoders MAY produce any sequence of bytes that a conformant parser would parse into the intended semantics.

In places, this specification says the parser "MAY return" some data. This indicates that the described data is complete enough that later parsing failures do not need to discard it.

In places, this specification says the parser "MUST fail". The parser MAY report these failures to its caller in any way, but MUST NOT return any data it has parsed so far that wasn't mentioned in a "MAY return" statement.

This specification creates local variables with the phrase "Let _variable-name_ be ...". Use of a variable before it's created is a defect in this specification.

2.2. Top-level structure

The package is roughly a CBOR item with the following CDDL schema, but package parsers are required to successfully parse some byte strings that aren't valid CBOR. For example, sections may have padding between them, or even overlap, as long as the embedded relative offsets cause the parsing algorithm in this specification to return data.

```
webpackage = [
  magic1: h'F0 9F 8C 90 F0 9F 93 A6',  ; &#127760;&#128230; in UTF-8.
  section-offsets: { * (($section-name .within tstr) => uint) },
  sections: [ *$section ],
  length: uint,                        ; Total number of bytes in the package.
  magic2: h'F0 9F 8C 90 F0 9F 93 A6',  ; &#127760;&#128230; in UTF-8.
]
```

The parser MAY begin parsing at either the beginning (Section 2.2.2) or end (Section 2.2.1) of the byte string representing the package. Parsing from the end is useful when the package is embedded in another format such as a self-extracting executable, while parsing from the beginning is useful when loading from a stream.

2.2.1. From the end

To parse from the end, the parser MUST load the last 18 bytes as the following [CDDL] group in array context: [ednote-loading-cddl]

```
tail = (
  length: uint,                       ; Total number of bytes in the package.
  magic2: h'F0 9F 8C 90 F0 9F 93 A6',  ; &#127760;&#128230; in UTF-8.
)
```

If the bytes don't match this group or these two CBOR items don't occupy exactly 18 bytes, parsing MUST fail.

Otherwise, continue as if the byte "length" bytes before the end of the string were the beginning of the package, and the parser were a from the beginning (Section 2.2.2) parser.

2.2.2. From the beginning

If the first 10 bytes of the package are not "85 48 F0 9F 8C 90 F0 9F 93 A6" (the CBOR encoding of the 5-item array header and 8-byte bytestring header, followed by 🌐📦 in UTF-8), parsing MUST fail.

Parse one CBOR item starting at the 11th byte of the package. If this does not match the CDDL

```
section-offsets = { * tstr => uint },
```

or it is not a Canonical CBOR item (Section 3.9 of [CBOR]), parsing MUST fail.

Let _sections-start_ be the offset of the byte after the "section-offsets" item. For example, if "section-offsets" were 52 bytes long, _sections-start_ would be 63.

This specification defines two section names: "indexed-content" and "manifest".

If "section-offsets"["indexed-content"] is not present, parsing MUST fail.

The parser MUST ignore unknown keys in the "section-offsets" map because new sections may be defined in future specifications. [ednote-critical-sections]

Let _index_ be the result of parsing the bytes starting at offset _sections-start_ + "section-offsets"["indexed-content"] using the instructions in Section 2.3.

If "section-offsets"["manifest"] is present, let _manifest_ be the result of parsing the bytes starting at offset _sections-start_ + "section-offsets"["manifest"] using the instructions in Section 2.4.

The parser MAY return a semantic package consisting of _index_, and, if initialized, _manifest_.

To parse each resource described within _index_, the parser MUST follow the instructions in Section 2.5.

2.3. Parsing the index

The main content of a package is an index of HTTP requests pointing to HTTP responses. These request/response pairs hold the manifests of sub-packages and the resources in the package and all of its sub-packages. Both the requests and responses can appear in any order, usually chosen to optimize loading while the package is streamed.

To parse the index, starting at offset _index-start_, the parser MUST do the following:

If the byte at _index-start_ is not 0x82 (the [CBOR] header for a 2-element array), the parser MUST fail.

Load a CBOR item starting at _index-start_ + 1 as the "index" array in the following CDDL:

```
$section-name /= "indexed-content"
$section /= index

index = [* [resource-key: http-headers,
            offset: uint,
            ? length: uint] ]

; http-headers is a byte string in HPACK format (RFC7541).
; The dynamic table begins empty for each instance of
; http-headers.
http-headers = bstr
```

If the item doesn't match this CDDL, or it is not a Canonical CBOR item (Section 3.9 of [CBOR]), the parser MUST fail.

Let _resources-start_ be the offset immediately after the "index" item. For example, if _index-start_ were 75 and the "index" item

were 105 bytes long, _resources-start_ would be 75+1+105=181. (1 for the 0x82 array header.)

Decode all of the "resource-key"s using [HPACK], with an initially-empty dynamic table for each one. [ednote-compression] The decoded "resource-key"s are header lists ([HPACK], Section 1.3), ordered lists of name-value pairs.

The parser MUST fail if any of the following is true:

1. HPACK decoding encountered an error.

2. Any "resource-key"'s first three headers are not named ":scheme", ":authority", and ":path", in that order. Note that ":method" is intentionally omitted because only the GET method is meaningful.

3. Any of the pseudo-headers' values violates a requirement in Section 8.1.2.3 of [HTTP2].

4. Any "resource-key" has a non-pseudo-header name that includes the ":" character or is not lower-case ascii ([HTTP2], Section 8.1.2).

5. Any two decoded "resource-key"s are the same. Note that header lists with the same header fields in a different order are not the same.

Increment all "offset"s by _resources-start_.

Return the resulting "index", an array of decoded-resource-key, adjusted-offset, and optional-length triples.

The optional "length" field in the index entries is redundant with the length prefixes on the "response-headers" and "body" in the content, but it can be used to issue Range requests [RFC7233] for responses that appear late in the content.

2.4. Parsing the manifest

A package's manifest contains some metadata for the package; hashes, used in Section 2.5, Paragraph 9, for all resources included in that package; and validity information for any sub-packages (Section 2.4.1) the package depends on. The manifest is signed, so that UAs can trust that it comes from its claimed origin. [ednote-manifest-name]

To parse a manifest starting at _manifest-start_, a parser MUST do the following:

Load one CBOR item starting at _manifest-start_ as a "signed-manifest" from the following CDDL:

```
$section-name /= "manifest"
$section /= signed-manifest

signed-manifest = {
  manifest: manifest,
  certificates: [+ certificate],
  signatures: [+ signature]
}

manifest = {
  metadata: manifest-metadata,
  resource-hashes: {* hash-algorithm => [hash-value]},
  ? subpackages: [* subpackage],
}

manifest-metadata = {
  date: time,
  origin: uri,
  * tstr => any,
}

; From https://www.w3.org/TR/CSP3/#grammardef-hash-algorithm.
hash-algorithm /= "sha256" / "sha384" / "sha512"
; Note that a hash value is not base64-encoded, unlike in CSP.
hash-value = bstr

; X.509 format; see https://tools.ietf.org/html/rfc5280
certificate = bstr

signature = {
  ; This is the index of the certificate within the certificates array to use
  ; to validate the signature.
  keyIndex: uint,
  signature: bstr,
}
```

If the item doesn't match the CDDL or it's not a Canonical CBOR item (Section 3.9 of [CBOR]), parsing MUST fail.

Parse the elements of "certificates" as X.509 certificates within the [RFC5280] profile. If any certificate fails to parse, parsing MUST fail.

Let _message_ be the concatenation of the following byte strings. This matches the [TLS1.3] format to avoid cross-protocol attacks when TLS certificates are used to sign manifests.

1. A string that consists of octet 32 (0x20) repeated 64 times.

2. A context string: the ASCII encoding of "Web Package Manifest".

3. A single 0 byte which serves as a separator.

4. The bytes of the "manifest" CBOR item.

Let _signing-certificates_ be an empty array.

For each element _signature_ of "signatures":

1. Let _certificate_ be "certificates"[_signature_["keyIndex"]].

2. The parser MUST define a partial function from public key types to signing algorithms, with the following map as a subset:

   RSA, 2048 bits: rsa_pss_sha256 as defined in Section 4.2.3 of [TLS1.3]

   EC, with the secp256r1 curve: ecdsa_secp256r1_sha256 as defined in Section 4.2.3 of [TLS1.3]

   EC, with the secp384r1 curve: ecdsa_secp384r1_sha384 as defined in Section 4.2.3 of [TLS1.3]

   Let _signing-alg_ be the result of applying this function to the key type in _certificate_'s Subject Public Key Info. If the function is undefined on this input, the parser MUST continue to the next _signature_.

3. Use _signing-alg_ to verify that _signature_["signature"] is _message_'s signature by _certificate_'s public key. If it's not, the parser MUST continue to the next _signature_.

4. Append _certificate_ to _signing-certificates_. Note that failed signatures simply cause their certificate to be ignored, so that packagers can give new signature types to parsers that understand them.

Let _origin_ be "manifest"["metadata"]["origin"].

Try to find a certificate in _signing-certificates_ that has an identity ([RFC2818], Section 3.1) matching _origin_'s hostname, and

that is trusted for serverAuth ([RFC5280], Section 4.2.1.12) using paths built from elements of "certificates" or any other certificates the parser is aware of. If no such certificate is found, and the package is not already trusted as received from _origin_'s hostname, for example because it was received over a TLS connection to that host, then parsing MUST fail.

TODO: Process the "subpackages" item by fetching those manifests via the index, and checking their signatures and dates/hashes, recursively.

The parsed manifest consists of the set of _signing-certificates_ and the "manifest" CBOR item. The items in "manifest"["metadata"] SHOULD be interpreted as described in the [appmanifest] specification.

2.4.1. Sub-packages

A sub-package is represented by a Section 2.4 file looked up as a Section 2.5 within the "indexed-content" section. The sub-package's resources are not otherwise distinguished from the rest of the resources in the package. Sub-packages can form an arbitrarily-deep tree.

There are three possible forms of dependencies on sub-packages, of which we allow two. Because a sub-package's manifest is protected by its own signature, if the main package trusts the sub-package's server, it could avoid specifying a version of the sub-package at all. However, this opens the main package up to downgrade attacks, where the sub-package is replaced by an older, vulnerable version, so we don't allow this option.

```
subpackage = [
  resource: resource-key,
  validation: {
    ? hash: {+ hash-algorithm => hash-value},
    ? notbefore: time,
  }
]
```

If the main package wants to load either the sub-package it was built with or any upgrade, it can specify the date of the original sub-package:

```
[32("https://example.com/loginsdk.package"), {"notbefore": 1(1486429554)}]
```

Constraining packages with their date makes it possible to link together sub-packages with common dependencies, even if the sub-packages were built at different times.

If the main package wants to be certain it's loading the exact version of a sub-package that it was built with, it can constrain sub-package with a hash of its manifest:

```
[32("https://example.com/loginsdk.package"),
 {"hash": {"sha256": b64'9qg0NGDuhsjeGwrcbaxMKZAvfzAHJ2d8L7NkDzXhgHk='}}]
```

Note that because the sub-package may include sub-sub-packages by date, the top package may need to explicitly list those sub-sub-packages' hashes in order to be completely constrained.

2.5. Parsing a resource

To parse the resource from a _package_ corresponding to a _header-list_, a parser MUST do the following:

Find the (_resource-key_, _offset_, _length_) triple in _package_'s index where _resource-key_ is the same as _header-list_. If no such triple exists, the parser MUST fail.

Parse one CBOR item starting at _offset_ as the following CDDL:

```
response = [response-headers: http-headers, body: bstr]
```

If the item doesn't match the CDDL or it's not a Canonical CBOR item (Section 3.9 of [CBOR]), parsing MUST fail.

Decode the "response-headers" field using [HPACK], with an initially-empty dynamic table. The decoded "response-headers" is a header list ([HPACK], Section 1.3), an ordered list of name-value pairs.

The parser MUST fail if any of the following is true:

1. HPACK decoding encountered an error.

2. The first header name within "response-headers" is not ":status", or this pseudo-header's value violates a requirement in Section 8.1.2.3 of [HTTP2].

3. Any other header name includes the ":" character or is not lower-case ascii ([HTTP2], Section 8.1.2).

4. The _header-list_ contains any header names other than ":scheme", ":authority", ":path", and either "response-headers" has no "vary" header (Section 7.1.4 of [RFC7231]) or these header names aren't listed in it.

Let _origin_ be the Web Origin [RFC6454] of _header-list_'s ":scheme" and ":authority" headers.

Let _resource-bytes_ be the result of encoding the array of [_header-list_, "response-headers", "body"] as Canonical CBOR in the following CDDL schema: [ednote-figure-in-list]

```
resource-bytes = [
  request: [
    *(header-name: bstr, header-value: bstr)
  ],
  response-headers: [
    *(header-name: bstr, header-value: bstr)
  ],
  response-body: bstr
]
```

Note that this uses the decoded header fields, not the bytes originally included in the package.

The hashed data differs from [SRI], which only hashes the body. Including the headers will usually prevent a package from relying on some of its contents being transferred as normal network responses, unless its author can guarantee the network won't change or reorder the headers.

If the _package_ contains a _manifest_:

1. *TODO:* Let _origin-manifest_ be the signed manifest for _origin_, found by searching through _manifest_'s subpackages for a matching origin.

2. Let _alg_ be one of the "hash-algorithm"s within _origin-manifest_. The parser SHOULD select the most collision-resistant hash algorithm. If the parser also implements [SRI], it SHOULD use the same order as its "getPrioritizedHashFunction()" implementation.

3. If the digest of _resource-bytes_ using _alg_ does not appear in the _origin-manifest_'s "resource-hashes"[_alg_] array, the parser MUST fail.

Return the (decoded "response-headers", "body") pair.

3. Guidelines for package authors

Packages SHOULD consist of a single Canonical CBOR item matching the "webpackage" CDDL rule in Section 2.2.

Every resource's hash SHOULD appear in every array within "resource-hashes": otherwise the set of valid resources will depend on the parser's choice of preferred hash algorithm.

4. Security Considerations

Signature validation is difficult.

Packages with a valid signature need to be invalidated when either

- the private key for any certificate in the signature's validation chain is leaked, or

- a vulnerability is discovered in the package's contents.

Because packages are intended to be used offline, it's impossible to inject a revocation check into the critical path of using the package, and even in online scenarios, such revocation checks don't actually work [2]. Instead, package consumers must check for a sufficiently recent set of validation files, consisting of OCSP responses [RFC6960] and signed package version constraints, for example within the last 7-30 days. *TODO:* These version constraints aren't designed yet.

Relaxing the requirement to consult DNS when determining authority for an origin means that an attacker who possesses a valid certificate no longer needs to be on-path to redirect traffic to them; instead of modifying DNS, they need only convince the user to visit another Web site, in order to serve packages signed as the target.

All subpackages that mention a particular origin need to be validated before loading resources from that origin. Otherwise, package A could include package B and an old, vulnerable version of package C that B also depends on. If B's dependency isn't checked before loading resources from C, A could compromise B.

5. IANA considerations

5.1. Internet Media Type Registration

IANA maintains the registry of Internet Media Types [RFC6838] at https://www.iana.org/assignments/media-types.

- Type name: application
- Subtype name: package+cbor [ednote-mime-naming]
- Required parameters: N/A
- Optional parameters: N/A
- Encoding considerations: binary
- Security considerations: See Section 4 of this document.
- Interoperability considerations: N/A
- Published specification: This document
- Applications that use this media type: None yet, but it is expected that web browsers will use this format.
- Fragment identifier considerations: N/A
- Additional information:
  - Deprecated alias names for this type: N/A
  - Magic number(s): 85 48 F0 9F 8C 90 F0 9F 93 A6
  - File extension(s): .wpk
  - Macintosh file type code(s): N/A
- Person & email address to contact for further information: See the Author's Address section of this specification.
- Intended usage: COMMON
- Restrictions on usage: N/A
- Author: See the Author's Address section of this specification.
- Change controller: The IESG iesg@ietf.org [4]
- Provisional registration? (standards tree only): Not yet.

6. References

6.1. Normative References

[appmanifest] Caceres, M., Christiansen, K., Lamouri, M., and A. Kostiainen, "Web App Manifest", World Wide Web Consortium WD WD-appmanifest-20170608, June 2017, <https://www.w3.org/TR/2017/WD-appmanifest-20170608>.

[CBOR] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <http://www.rfc-editor.org/info/rfc7049>.

[CDDL] Birkholz, H., Vigano, C., and C. Bormann, "CBOR data definition language (CDDL): a notational convention to express CBOR data structures", draft-greevenbosch-appsawg-cbor-cddl-10 (work in progress), March 2017.

[HPACK] Peon, R. and H. Ruellan, "HPACK: Header Compression for HTTP/2", RFC 7541, DOI 10.17487/RFC7541, May 2015, <http://www.rfc-editor.org/info/rfc7541>.

[HTML] WHATWG, "HTML", n.d., <https://html.spec.whatwg.org/multipage/>.

[HTTP2] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <http://www.rfc-editor.org/info/rfc7540>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <http://www.rfc-editor.org/info/rfc2818>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <http://www.rfc-editor.org/info/rfc5280>.

[RFC6454] Barth, A., "The Web Origin Concept", RFC 6454, DOI 10.17487/RFC6454, December 2011, <http://www.rfc-editor.org/info/rfc6454>.

[RFC6960] Santesson, S., Myers, M., Ankney, R., Malpani, A., Galperin, S., and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 6960, DOI 10.17487/RFC6960, June 2013, <http://www.rfc-editor.org/info/rfc6960>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <http://www.rfc-editor.org/info/rfc7231>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <http://www.rfc-editor.org/info/rfc8174>.

[SRI] Akhawe, D., Braun, F., Marier, F., and J. Weinberger, "Subresource Integrity", World Wide Web Consortium Recommendation REC-SRI-20160623, June 2016, <http://www.w3.org/TR/2016/REC-SRI-20160623>.

[TLS1.3] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", draft-ietf-tls-tls13-20 (work in progress), April 2017.

6.2. Informative References

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <http://www.rfc-editor.org/info/rfc6838>.

[RFC7233] Fielding, R., Ed., Lafon, Y., Ed., and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Range Requests", RFC 7233, DOI 10.17487/RFC7233, June 2014, <http://www.rfc-editor.org/info/rfc7233>.

[ServiceWorkers] Russell, A., Song, J., Archibald, J., and M. Kruisselbrink, "Service Workers 1", World Wide Web Consortium WD WD-service-workers-1-20161011, October 2016, <https://www.w3.org/TR/2016/WD-service-workers-1-20161011>.

6.3. URIs

[1] https://github.com/WICG/webpackage/issues/45

[2] https://www.imperialviolet.org/2012/02/05/crlsets.html

[4] mailto:iesg@ietf.org

Appendix A. Acknowledgements

Thanks to Adam Langley and Ryan Sleevi for in-depth feedback about the security impact of this proposal.

Editorial Comments

[ednote-loading-cddl] jyasskin: CDDL doesn't actually define how to use it as a schema to load CBOR data.

[ednote-critical-sections] jyasskin: Do we need to mark critical section names?

[ednote-compression] jyasskin: This spec has different security constraints from the ones that drove HPACK, so we may be able to do better with another compression format.

[ednote-manifest-name] jyasskin: This section doesn't describe a manifest (https://www.merriam-webster.com/dictionary/manifest#h3), so consider renaming it to something like "authenticity".

[ednote-figure-in-list] jyasskin: This step would be inside the manifest-only block, but then the code block is rendered out-of-order.

[ednote-mime-naming] jyasskin: I suspect the mime type will need to be a bit longer: application/webpackage+cbor or similar.

Author's Address

Jeffrey Yasskin
Google

Email: jyasskin@chromium.org