

Network Working Group
Internet-Draft
Expires: August 16, 2004

L. Daigle
T. Hardie
Editor
Internet Architecture Board
IAB
February 16, 2004

Considerations on Increasing Character Repertoires for Protocol
Actionable Elements
draft-iab-char-rep-01

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of [Section 10 of RFC2026](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on August 16, 2004.

Copyright Notice

Copyright (C) The Internet Society (2004). All Rights Reserved.

Abstract

This document describes a set of considerations and strategies to use in increasing the character repertoire available in a protocol actionable element or suite of protocol actionable elements. This document is not meant to provide normative instruction to protocol designers, but does hope to provide guidance on common issues arising from this task. Feedback should be sent to the editors or the IAB.

Internet-Draft

[draft-iab-char-rep-01](#)

February 2004

Table of Contents

1.	Definitions	3
2.	Introduction	4
3.	Avoidance mechanisms	5
3.1	Choosing a large initial character repertoire	5
3.2	Choosing opaque protocol tokens	5
3.3	Expansion mechanisms	6
3.4	Replace	6
3.5	Subsume	7
3.6	Map	7
4.	Layering a presentation element on a new protocol element . .	9
5.	Selecting a strategy	10
6.	Case Studies	11
6.1	Uniform and Internationalized Resource Identifiers	11
7.	Security Considerations	12
8.	IANA Considerations	13
9.	Acknowledgements	14
	References	15
	Authors' Addresses	15
	Full Copyright Statement	16

Internet-Draft

[draft-iab-char-rep-01](#)

February 2004

1. Definitions

Protocol (actionable) element: A protocol actionable element, or protocol element is any portion of a message which affects processing of that message by the protocol in question. In general, protocol elements are bound to specific processing choices by membership in a set of predetermined tokens or by explicit structure. Protocol elements are context dependent in that the processing for a token is specific to a protocol. To IP, for example, a TCP port number is payload; to TCP it is a protocol element. Similarly, to TCP a Content-encoding: header is payload; to HTTP, it is a protocol element.

Character repertoire: A character repertoire is the set of all characters in all permitted encodings which may be used in a protocol element. Each element in a character repertoire is a tuple of a code point and an encoding. Thus the glyph "a" would appear three times in a character repertoire that permitted ASCII, iso-8859-1, and iso-8859-7.

Character set: As it says, "a set of characters", but more particularly a set of characters as represented by code points in a particular encoding.

[2.](#) Introduction

After a protocol's initial deployment, changes in the use of the protocol sometimes necessitate revisiting the character repertoire originally chosen for one or more of the elements which make up the protocol. On rare occasions, this occurs because the protocol designers need to increase the number of tokens available in a fixed-length field and choose to do so by increasing the number of characters which may be used. More commonly, the motive for the increase of a character repertoire is the exposure of a protocol element to a user community. Once this leakage occurs, there is often pressure to expand the permitted character repertoire of the protocol element to match the character repertoire in use in that community.

Though increasing a character repertoire may appear to be a relatively simple matter, there are a number of protocol processing functions which may be affected. First among these is matching. Many encodings have very specific matching rules or equivalence tables; increasing a character repertoire to include a new encoding implies that the protocol must specify how matching works in that encoding. Like matching, sorting works in different ways in different encoding schemes, and including a new encoding means specifying sorting algorithms for use with it. Transformation presents some unique issues, as it may be possible for some systems to map only unidirectionally from one encoding to another. Any of these, and more, can present problems to a protocol designer who must post-facto retrofit an increased character repertoire into a deployed

protocol.

[3.](#) Avoidance mechanisms

To avoid the need to increase character repertoires at some later date, protocol designers can either start with a character repertoire which is large enough to encompass that in use in the target user community or use protocol elements that are sufficiently opaque to a human user that their leakage is unlikely to present later pressure. Both strategies, unfortunately, have been notoriously difficult to get right.

[3.1](#) Choosing a large initial character repertoire

In this avoidance strategy, the protocol designers presume that their protocol elements will leak in the future and provide a character repertoire which is sufficiently rich to match the user community's needs. Increasing use of a protocol, however, often changes the target user community beyond the initial designers' projections. A character repertoire which looks large to one user community may be completely wrong or very limited to another. When protocol designers attempt to avoid the issue by using a character repertoire with a very large number of code points in a very large number of encodings,

they incur real costs in parser complexity, processing overhead, and bloat. They also risk that misconfiguration of these complex parsers will result in incorrect protocol processing.

[3.2](#) Choosing opaque protocol tokens

In the second case, designers who choose to use tokens or structure which are not human-readable can resist later pressure to increase the character repertoire available. As those who have used encodings like ASN.1 can attest, there is, however, an increased development cost, as those working with the protocol must develop an understanding of the use of the tokens or structure without the aid of readability. This avenue may also be blocked or narrowed to protocol designers who will need to pass the new elements among different protocols; in those cases, the new protocol is either constrained by the previous choices or must provide a normative mapping to them.

When designers use tokens or structures which are not human readable, it is common to create a presentation format or layer which is mapped to the tokens or structures. One of the advantage to this approach is that new mappings can be defined as new user communities express the need for them. It is important, however, that these are always retained as mappings to the protocol elements, and are not treated as protocol elements themselves.

[3.3](#) Expansion mechanisms

For designers who must increase the character repertoire for a particular protocol element, there are three basic strategies available: they may replace the existing protocol element with a new one; they may subsume the character repertoire of the existing protocol element in a new one; they may map the new character repertoire into the existing repertoire.

For each of the following strategies, consider the following example: a protocol element called "POSTAL" used to name the U.S. zip code in which the network element is placed cannot handle postal codes containing characters outside (0,1,2,3,4,5,6,7,8,9) encoded in a subset of US-ASCII. We will refer to this character set as (NUM-

ASCII). The original character repertoire for this protocol element has NUM-ASCII as its single member character set.

[3.4](#) Replace

Replacing an existing protocol element with an entirely new protocol element with a different character repertoire is by far the cleanest solution from a design perspective. A new protocol element may have its own matching and sorting rules, without regard to any previous deployment. This means that the new element will have as little baggage as is possible when updating parsers and setting forth how it fits into the protocol's semantics.

Unfortunately, this method presents a raft of deployment problems. Since existing protocol implementations will know nothing about it, they cannot be interoperable with any entirely new protocol element. At best, they can ignore it gracefully; at worst, they will fail. A protocol designer can react to this by changing the revision number on a protocol, by using some form of feature negotiation, or by using heuristics (including failure!) to determine whether or not a new protocol element may be used. All of these are difficult to get right, especially in hop-by-hop protocols, in which it may not be possible to determine whether all hops support specific features or versions.

A protocol designer tackling this problem for the protocol element naming the postal code in which a network element is placed might replace "POSTAL" with "NEW_POSTAL" and create a new character repertoire for "NEW_POSTAL" which contained the single entry (ISO-8859-1). [This is merely an example; the choice of which character set or sets to use would be made in this instance by reference to the relevant international postal standards.] Obviously, any system which did not understand "NEW_POSTAL" would need to be upgraded to handle the new character set. Depending on the transition mechanism,

systems communicating postal codes which were numeric-only might well include both "POSTAL" and "NEW_POSTAL" protocol elements.

[3.5](#) Subsume

Rather than completely replacing an existing protocol element, another strategy is to create a protocol element which subsumes the

character repertoire of the existing protocol element. When this option is chosen, the new protocol element retains all the character sets and the related matching and sorting rules which were originally present. These become a strict subset of the new character repertoire.

This strategy limits the functionality of the new protocol element both by forcing it to include specific character sets and by requiring that the semantics of the new protocol element exactly match the existing protocol element. This strategy also retains many of the deployment problems of the replacement strategy, though it offers some opportunities to mitigate the issues. Like the replacement strategy, there may need to be negotiation mechanisms capable of handling both protocol elements, though new implementations can sometimes treat the old protocol element as a degenerate case of new protocol element.

If our "POSTAL" protocol design team took this strategy, they might replace the (NUM-ASCII) character repertoire of "POSTAL" with a new protocol element "BIG_POSTAL" for which the character repertoire is (NUM-ASCII, US-ASCII). Because NUM-ASCII is a strict subset of US-ASCII, the protocol can treat all "POSTAL" protocol elements as if they were "BIG_POSTAL" protocol elements. Note that this is the simplest possible example of this particular strategy, as there is no need to mark which character set from the character repertoire is in use. More complex examples may require much more complex processing to achieve the same results.

[3.6](#) Map

In some instances it may be possible and desirable to map an expanded character repertoire onto the existing code points specified by a protocol. In this case, the code points are themselves retained but the character encoding portion of the tuple is changed to create an expanded character repertoire. This strategy can only work when some marker is used to indicate which character encoding applies to a specific instance of the protocol. This marker must be something which is non-operative in the original protocol processing, or the strategy will incur the negotiation costs mentioned above. This strategy will tend to increase the size of protocol elements unless the original code points were radically under-used. It also carries

the near-certainty that there will be occasions in which protocol elements encoded with the new character encoding are mis-identified as being encoded with the original character encoding.

This strategy has somewhat unique deployment consequences, in that it is both easier to get initial deployment and harder to get complete penetration. Because the same code points are used throughout, there is no requirement that all systems upgrade for the increased character repertoire to be available to a subset of users. There is also, however, almost no incentive for upgrade of systems which do not themselves require the increased repertoire. This is particularly true in hop-by-hop and commonly proxied protocols, because the on-path intermediate systems will pass the elements of the expanded repertoire by virtue of their being legitimate code points in the original repertoire; they do not need to upgrade and they probably never will.

For our protocol design team to tackle "POSTAL" using this strategy they must develop or discover an encoding which allows them to represent all the needed characters using just (NUM-ASCII). If, for example, the character repertoire needed to add a character set which included (A-Z), but no others, the team could use US-ASCII's three digit decimal encoding for each included character. A postal code like "KLHSW1" would then be encoded as "075076071083087049". Provided that the original POSTAL protocol element had a field length sufficient to handle the new encoding, it could carry the new values without any difficulty. The difficulty would be determining whether the new encoding or the old should be assumed; in this limited case, length alone could be made a marker by padding any short alphabetic postal codes with the ASCII null character, "000", until they reached a length sufficient to trigger treatment as non-ZIP code postal codes. In other cases more complex triggers would be required.

Internet-Draft[draft-iab-char-rep-01](#)

February 2004

[4.](#) Layering a presentation element on a new protocol element

It is noted above that designers using non-human readable tokens may provide a mapping to a presentation element which can be used by humans working with the protocol. In employing any of the strategies above, it is useful for protocol designers to consider introducing a presentation element at the same time. This is almost a required part of the mapping strategy, as using an encoding based on the original set of code points does not help the user community unless it can also be mapped to an encoding in common use for presentation. It may be used with any of them, though, and given the potential for the introduction of new character encodings, it must be considered carefully as a method of ensuring that the same problem does not face the protocol in a few years time.

[5.](#) Selecting a strategy

The first step in selecting a strategy is identifying the protocol processing choices which depend on the protocol element. If a protocol element is passed among different protocols, this set of choices must be identified for each of the protocols which depend on the element. After those have been identified, the available methods for passing the protocol elements from one protocol to another must be considered.

If at all possible, a single strategy should be selected for use with a specific protocol element, even when that protocol element will be passed among different protocols. Since protocol processing is context-specific, it is technically possible to use different methods in different contexts, but this increase in complexity rarely has a corresponding gain.

Whether the protocol element will be used in one protocol or several, the core question to consider is how best to maintain interoperability while increasing the character repertoire. For example, if creating a new protocol element as a fully fledged replacement, are there available mechanisms to handle the negotiation and/or versioning? Alternatively, are there methods which would allow both protocol elements to coexist?

The second question to consider is the cost of implementation. If, for example, a choice is made to introduce a protocol element which subsumes the original character repertoire in a larger character repertoire, how expensive will the increase in parsing complexity be?

The third question to consider is likely deployment patterns. For a client/server protocol, will it be feasible to update both client and server? For a hop-by-hop protocol, will there be any pressure for intermediate servers to upgrade?

A related question is whether this change will be tied to other

changes which will drive adoption, or whether this change will be unrelated to other updates to the protocol.

[6.](#) Case Studies

[6.1](#) Uniform and Internationalized Resource Identifiers

Uniform Resource Names ([\[1\]](#)) make use of the 7-bit US-ASCII character repertoire. The syntax of the URI permits other encodings to be mapped into that repertoire, by defining a hex-encoding framework.

Increasingly, new URI schemes are using UTF-8 to for characters beyond US-ASCII. In recognition of this, and to provide a means to handle such identifiers in a more straightforward manner, the "Internationalized Resource Identifier" (IRI) has been introduced.

From [\[2\]](#):

"This document defines a new protocol element, the Internationalized Resource Identifier (IRI), as a complement to the URI [RFCYYYY]. An IRI is a sequence of characters from the Universal Character Set [ISO10646]. A mapping from IRIs to URIs is defined, which means that IRIs can be used instead of URIs where appropriate to identify resources."

The IRI specification applies the "replace", "map" and "subsume" strategies for expansion outlined above. As noted in the quoted text from the IRI document, IRIs are defined as a new protocol element ("replace"). Therefore, any protocol or message format defined in the future may use an IRI protocol element and not a URI protocol element. However, as URIs are ubiquitous and IRIs would face steep deployment challenges without the possibility of relating to URIs.

Therefore, [2] defines a mapping strategy to ensure IRIs can be mapped onto URIs and vice versa.

The IRI document also goes on to note that there are specifications already designated to handle IRIs -- "anyURI" in XML Schema. This is an example of subsumption.

While the IRI document is clear that conversions between IRI and URI formats must be made when transitioning from systems that understand IRIs to ones that do not, it is unclear how message parsers that detect and interpret "http://" as a URI will recognize IRIs as distinct from (malformed) URIs.

[7](#). Security Considerations

Any protocol processing which depends on a specific set of tokens or structure is at risk when the matching and sorting rules for the set is indeterminate. In some cases, this can result in a denial of service, as legitimate tokens are not recognized; in other cases, inappropriate access may be granted by matching incorrectly.

[8.](#) IANA Considerations

There are no IANA considerations defined in this memo.

[9.](#) Acknowledgements

The authors would like to thank Martin Duerst for his attention and expertise.

- [1] Berners-Lee, T., Fielding, R. and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", [RFC 2396](#), August 1998.
- [2] Duerst, M. and M. Suignard, "Internationalized Resource Identifiers (IRIs)", [draft-duerst-iri-05.txt](#) (work in progress), October 2003.

Authors' Addresses

Leslie Daigle
Editor

Ted Hardie
Editor

Internet Architecture Board
IAB

EMail: iab@iab.org

Full Copyright Statement

Copyright (C) The Internet Society (2004). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.

