Internet Draft
draft-ietf-idn-sace-00.txt
Expires: 27 February 2001

Dan Oscarsson
Telia ProSoft
27 August 2000

Simple ASCII Compatible Encoding (SACE)

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/1id-abstracts.txt

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

Abstract

This document describes a way to encode non-ASCII characters in host names in a way that is completely compatible with the current ASCII only host names that are used in DNS. It can be used both with DNS to support software only handling ASCII host names and as a way to downgrade from 8-bit text to ASCII in protocols.

1. Introduction

This document defines an ASCII Compatible Encoding (ACE) of names that can be used when communicating with DNS. It is needed during a transition period when non-ASCII names are introduced in DNS to avoid breaking programs expecting ASCII only.

The Simple ASCII Compatible Encoding (SACE) defined here can be compared to [RACE]. The main differences are:

- RACE encodes by first compressing and the encoding the resulting bit stream into ASCII. SACE encodes each character directly in one

pass.
- SACE recognises that at lot of latin based names are mostly composed of ASCII characters and gives a higher compression for those. In the 63 byte limit of DNS RACE will allow 36 characters for ISO 8859-1 and less if characters from the additional Latin characters are needed. SACE will allow around 40 characters if about 10 % of a Latin name is non-ASCII (in the UCS [ISO10646] range 0-0x217). SACE is closer to the compression that UTF-8 have than RACE.
- Most ASCII characters will not be encoded so Latin based names composed of mostly ASCII characters will be somewhat readable.

1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Simple ASCII Compatible Encoding

The encoding encodes values using the available characters allowed in a ASCII host name (a-z0-9 and hyphen).

Values are encoded as follows:

Character - value mapping

value	character	value	character
0	a	18	s
1	b	19	t
2	c	20	u
3	d	21	v
4	e	22	w
5	f	23	x
6	g	24	y
7	h	25	z
8	i	26	1
9	j	27	2
10	k	28	3
11	l	29	4
12	m	30	7
13	n	31	9
14	o	32	0
15	p	33	8
16	q	34	5
17	r	35	6

```
In the following description the following syntax will be used:
   B => one value in the range 0-35 mapped to a character as above
   X => one value in the range 0-31 mapped to a character as above

Each UCS character is identified as follows:
   latin  => a character in the range 0-0x217
   10bit  => a character in the range 0x218-0x2FFF
   base36 => all other characters

During encoding/decoding a string a current mode is used. In each
mode characters are encoded like this:
   latin  => as themselves, 00 for 0, 88 for 8 or as 10 bit value
             encoded as 0XX (two 5 bit values)
   10bit  => as 15 bits represented by its current prefix of 5 bits
             followed by 10 bits encoded as XX
             (the value is the 15 bits of prefix and
             10 bits concatenated)
   base36 => as a base 36 value represented by its current base 36
             prefix followed by three base 36 digits encoded as BBB
             (the value is prefix*36*36*36*36+B*36*36+B*36+B)
             Before encoding the character value must first be
             reduced:
               if >= 0xd800 reduce by 8192 (private/surrogate start)
               then reduce by 0x2FFF.
             After decoding the character value need to be restored
as
               add 0x2FFF
               followed by adding 8192 if >= 0xd800
```

2.1 Decoding a string

```
During decode you start with:
   Mode: latin
   10bit prefix: 0
   base36 prefix: 0

Then the characters in an encoded string are interpreted as follows
depending on current mode:

 When in latin mode:
   00  => the character 0
   0XX => XX represents 10 bits which decodes to one character
   88  => the character 8
   85  => switch to 10bit mode with same prefix as last time
   8X5 => switch 10 10bit mode setting X as current 10bit prefix
   87  => switch to base36 mode with same prefix as last time
   8X7 => switch to base36 mode setting X as current base36 prefix
```

```
   other  => the characters represent itself

 When in 10bit mode
   - => the character -
   0 => switch to latin mode
   X5 => switch 10 10bit mode using X as current prefix
   7  => switch to base36 mode with same prefix as last time
   X7 => switch to base36 mode using X as current prefix
   XX => current 10bit prefix plus XX gives the character

 When in base36 mode
   -- => the character -
   -0 => switch to latin mode
   -5 => switch to 10bit mode with same prefix as last time
   -X5 => switch 10 10bit mode setting X as current prefix
   -X7 => switch to base36 mode setting X as current prefix
   XXX => current base36 prefix plus XXX as base 36 values gives
character
```

2.2 Encoding a string

To encode a string you start with the data as UCS characters and:

```
   Mode: latin
   10bit prefix: 0
   base36 prefix: 0
```

Then for each UCS character, the mode and/or prefix is switched if needed and then the character is encoded as defined above.

3. References

[RFC2119] Scott Bradner, "Key words for use in RFCs to Indicate Requirement Levels", March 1997, RFC 2119.

[RFC2279] F. Yergeau, "UTF-8, a transformation format of ISO 10646", RFC 2279, January 1998.

[ISO10646] ISO/IEC 10646-1:2000. International Standard -- Information technology -- Universal Multiple-Octet Coded Character Set (UCS)

[Unicode] The Unicode Consortium, "The Unicode Standard -- Version 3.0", ISBN 0-201-61633-5. Described at http://www.unicode.org/unicode/standard/versions/Unicode3.0.html

[IDNREQ] James Seng, "Requirements of Internationalized Domain Names", draft-ietf-idn-requirement.

[RACE] Paul Hoffman, "RACE: Row-based ASCII Compatible Encoding for IDN", draft-ietf-idn-race.

4. Acknowledgements

Paul Hoffman for many good ideas.

Author's Address

Dan Oscarsson
Telia ProSoft AB
Box 85
201 20 Malmo
Sweden

E-mail: Dan.Oscarsson@trab.se