Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 5, 2021

P. Faltstrom
Netnod
F. Ljunggren
Kirei
D. van Gulik
Webweaving

April 03, 2021

The Base45 Data Encoding
draft-faltstrom-base45-04

Abstract

This document describes the base 45 encoding scheme which is built upon the base 64, base 32 and base 16 encoding schemes.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

When using QR or Aztec codes a different encoding scheme is needed than the already established base 64, base 32 and base 16 encoding schemes that are described in RFC 4648 [RFC4648]. The difference from those and base 45 is the key table and that the padding with '=' is not required.

2. Conventions Used in This Document

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 RFC 2119 [RFC2119].

3. Interpretation of Encoded Data

Encoded data is to be interpreted as described in RFC 4648 [RFC4648] with the exception that a different alphabet is selected.

4. The Base 45 Encoding

A 45-character subset of US-ASCII is used, the 45 characters that can be used in a QR or Aztec code. If we look at Base 64, it encodes 3 bytes in 4 characters. Base 45 encodes 2 bytes in 3 characters.

The two bytes [A, B] are turned into [C, D, E] where (A*256) + B = C + (D*45) + (E*45*45). The values C, D and E are then looked up in Table 1 to produce a three character string and the reverse when decoding.

If the number of octets are not dividable by two, the last remaining byte is represented by two characters. [A] is turned into [C, D] where A = C + (D*45).

4.1. When to use Base45

If binary data is to be stored in a QR-Code one possible way is to use the Alphanumeric encoding that uses 11 bits for 2 characters as defined in section 7.3.4 in ISO/IEC 18004:2015 [ISO18004]. The ECI mode indicator for this encoding is 0010.

If the data is to use some other transport a transport encoding suitable for that transport should be used. It is not recommended to for example first encode data in Base45 and then encode the Base45 blob in for example Base64 if the data is to be sent via email. Instead the Base45 encoding should be removed, and the data itself should be encoded in Base64.

4.2. The alphabet used in Base45

The alphanumeric code is defined to use 45 characters as specified in this alphabet.

Table 1: The Base 45 Alphabet

Value	Encoding	Value	Encoding	Value	Encoding	Value	Encoding
00	0	12	C	24	O	36	Space
01	1	13	D	25	P	37	$
02	2	14	E	26	Q	38	%
03	3	15	F	27	R	39	*
04	4	16	G	28	S	40	+
05	5	17	H	29	T	41	-
06	6	18	I	30	U	42	.
07	7	19	J	31	V	43	/
08	8	20	K	32	W	44	:
09	9	21	L	33	X		
10	A	22	M	34	Y		
11	B	23	N	35	Z		

4.3. Encoding example

A series of bytes is turned into groups of two. Each such 16 bit value is turned into a series of three values calculated by doing successive calculations modulo 45. The values are in turned looked up in what is displayed in Table 1.

It should be noted that although the examples are all text, Base45 is an encoding for binary data where each octet can have any value 0-255.

Encoding example 1: The string "AB" is the byte sequence [65 66]. The 16 bit value is 65 * 256 + 66 = 16706. 16706 equals 11 + 45 * 11 + 45 * 45 * 8 so the sequence in base 45 is [11 11 8]. By looking up these values in the table we get the encoded string "BB8".

Encoding example 2: The string "Hello!!" is the byte sequence [72 101 108 108 111 33 33]. If we look at each 16 bit value, it is [18533 27756 28449 33]. Note the 33 for the last byte. When looking at the values modulo 45, we get [[38 6 9] [36 31 13] [9 2 14] [33 0]] where the last byte is represented by two. By looking up these values in the table we get the encoded string "%69 VD92EX0".

Encoding example 3: The string "base-45" is the byte sequence [98 97 115 101 45 52 53]. If we look at each 16 bit value, it is [25185 29541 11572 53]. Note the 53 for the last byte. When looking at the values modulo 45, we get [[30 19 12] [21 26 14] [7 32 5] [8 1]] where the last byte is represented by two. By looking up these values in the table we get the encoded string "UJCLQE7W581".

4.4. Decoding example

The series of characters are lookup up in Table 1, and the indices for the characters, three and three, are interpreted as a number in base 45. This number is then turned into two bytes in base 8.

Decoding example 1: The string "QED8WEX0" represents when lookup in Table 1 the values [26 14 13 8 32 14 33 0]. We look at the numbers in three number sequences (except last) and get [[26 14 13] [8 32 14] [33 0]]. In base 45 we get [26981 29798 33] where the bytes are [[105 101] [116 102] [33]]. If we look at the ascii values we get the string "ietf!".

5. IANA Considerations

There are no considerations for IANA in this document.

6. Security Considerations

When implementing encoding and decoding it is important to be very careful so that buffer overflow does not take place, or anything similar. This includes of course the calculations of modulo 45 and lookup in the table of characters. Decoder also must be robust regarding input, including proper handling of any byte value 0-255, including the NUL character (ASCII 0).

It should be noted that Base 64 (for example) pad the string so that the encoding has the correct number of characters. This is something that Base 45 does not do, i.e. Base 45 do not include padding. Because of this, special care is to be taken when odd number of octets are to be encoded which results not in N*3 characters, but (N-1)*3+2 characters in the encoded string and vice versa, when the number of encoded characters are not divisible by 3.

Further that a base45 encoded piece of data includes non-URL-safe characters so if base45 encoded data have to be URL safe, one have to use %-encoding.

7. Acknowledgements

The authors thank Alan Barrett, Tomas Harreveld, Christian Landgren, Anders Lowinger, Jakob Schlyter, Peter Teufl and Gaby Whitehead for the feedback. Also everyone that have been working with Base64 during the years that have proven the implementions are stable.

8. Normative References

[ISO18004]
ISO/IEC JTC 1/SC 31, "ISO/IEC 18004:2015 Information technology - Automatic identification and data capture techniques - QR Code bar code symbology specification", ISO/IEC 18004:2015 https://www.iso.org/standard/62021.html, February 2015.

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

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/info/rfc4648>.

Authors' Addresses

Patrik Faltstrom
Netnod

Email: paf@netnod.se

Fredrik Ljunggren
Kirei

Email: fredrik@kirei.se

Dirk-Willem van Gulik
Webweaving

Email: dirkx@webweaving.org