K Types, Under the Hood

For various reasons it is sometimes important to understand the underlying representation of K data. This article's purpose is to explain the most basic cases of how K data is represented under the hood.

Note that this article describes the output/input to _db/_bd as well as the way data is put on the wire by 3: and 4:. It does not describe the way data is represented in memory/on disk.

All K data starts with an 8 byte header. The first byte indicates endianness, the second and third are ignored, the fourth byte is only used during IPC, and indicates if the message is a 3:, 4: or response to a 4:. The fifth through eighth bytes represent the number of bytes remaining in the data structure. This header only exists at the beginning of the data, and is not repeated for each element in lists.

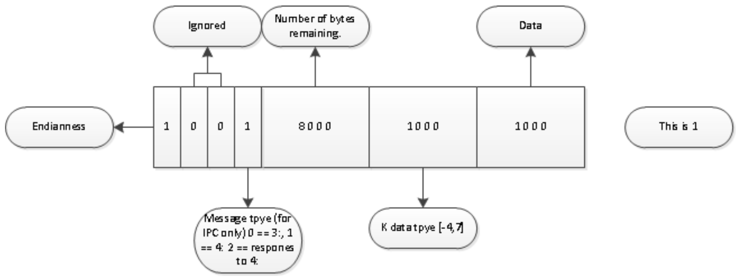
After the header, the next four bytes represent the type of the K data, -4 through 7.

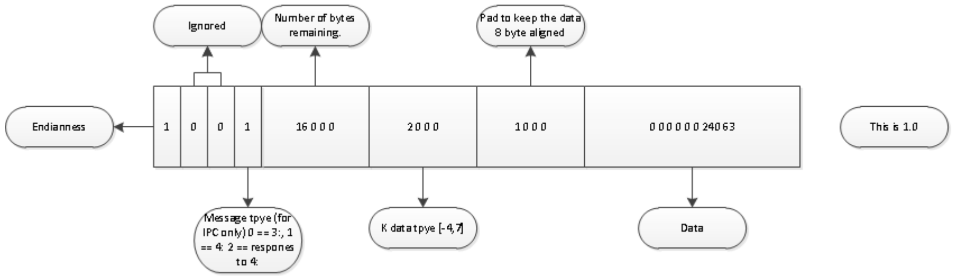
If we are dealing with an array (k types -4,-3,-2,-1,0,5), the next four bytes represent the length of the array.

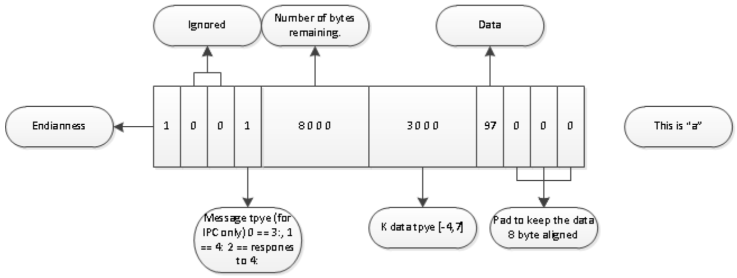
With the exception of floats, next comes the data. K types -3 and 4 always include a null terminating byte. For arrays the data members appear one after the other. For mixed type lists the "data" is a sequence of k type, (array length,) data, k type, (array length,) data.

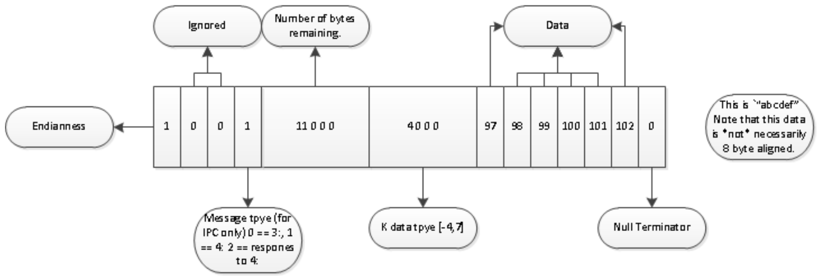
K types -2,0,1,2,3,5,6 are always 8 byte aligned. To facilitate this 8 byte alignment, after the data a pad is applied to round out the data structure. For k type 0 lists these pads are applied between each element so each element is 8 byte aligned. The exception, as alluded to earlier, is that for floats, the pad is applied before the data, after the k type.

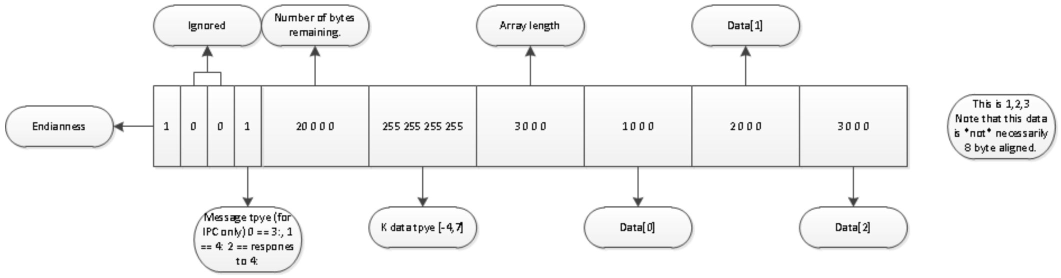
Below is a graphic illustrating the above. Note, where it says "number of bytes remaining" this indicates the number of bytes left in that data structure, not to be misconstrued with number of bytes remaining on the wire during an IPC transfer.

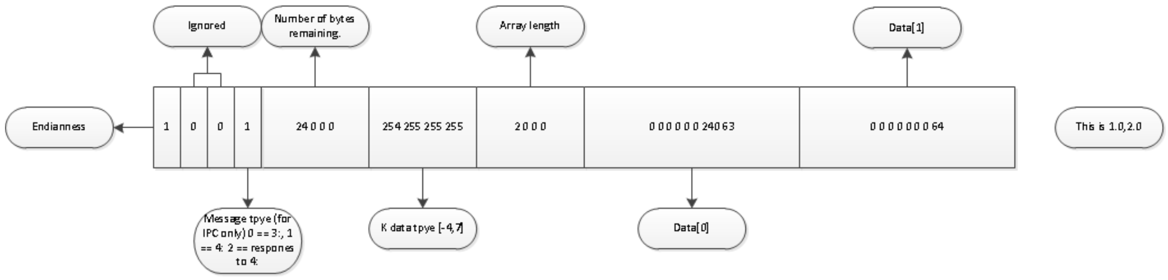

Ignored
Number of bytes remaining.
Data
Endianness
1
0
0
1
8 0 0 0
1 0 0 0
1 0 0 0
This is 1
Message tpye (for IPC only) 0 == 3:, 1 == 4: 2 == respones to 4:
K data tpye [-4,7]

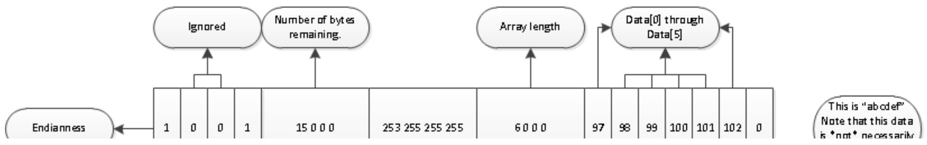

Ignored
Number of bytes remaining.
Pad to keep the data 8 byte aligned
Endianness
1
0
0
1
16 0 0 0
2 0 0 0
1 0 0 0
0 0 0 0 0 0 240 63
This is 1.0
Message tpye (for IPC only) 0 == 3:, 1 == 4: 2 == respones to 4:
K data tpye [-4,7]
Data

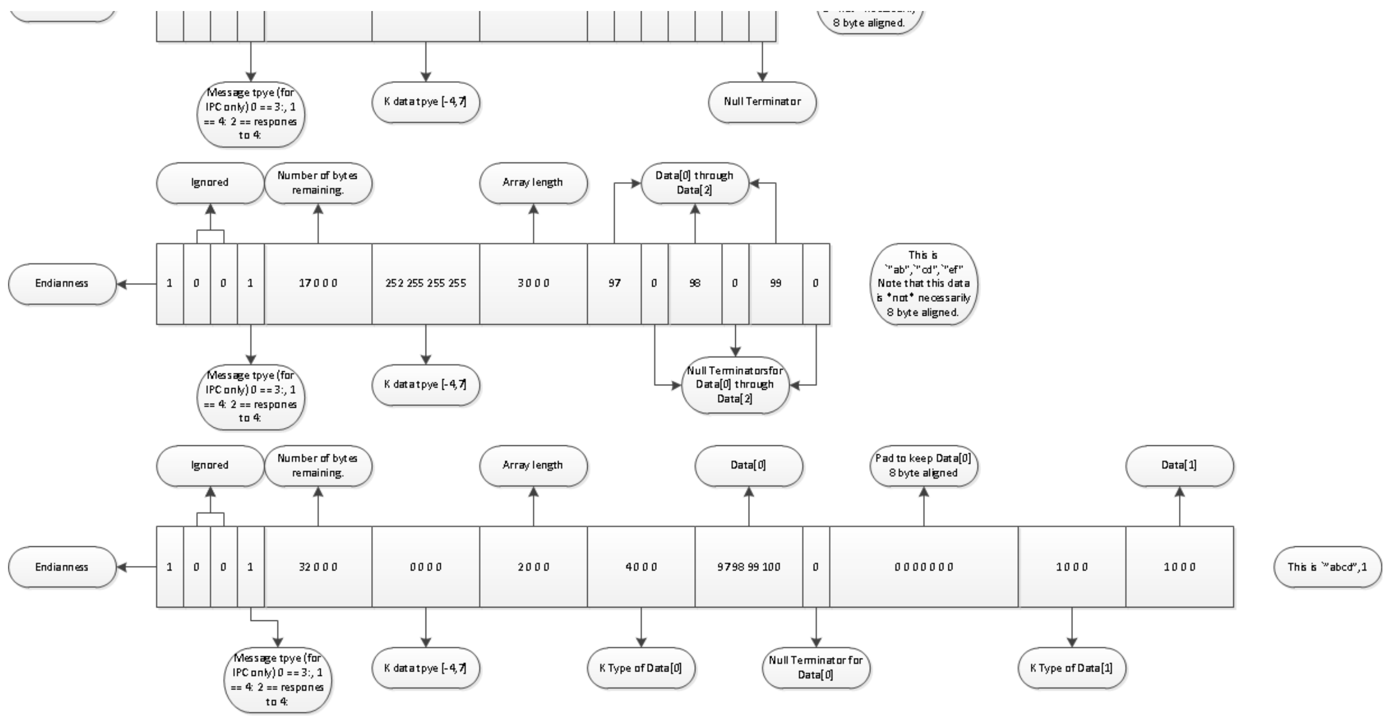

Ignored
Number of bytes remaining.
Data
Endianness
1
0
0
1
8 0 0 0
3 0 0 0
97
0
0
0
This is "a"
Message tpye (for IPC only) 0 == 3:, 1 == 4: 2 == respones to 4:
K data tpye [-4,7]
Pad to keep the data 8 byte aligned


Ignored
Number of bytes remaining.
Data
Endianness
1
0
0
1
11 0 0 0
4 0 0 0
97
98
99
100
101
102
0
This is "abcdef" Note that this data is *not* necessarily 8 byte aligned.
Message tpye (for IPC only) 0 == 3:, 1 == 4: 2 == respones to 4:
K data tpye [-4,7]
Null Terminator


Ignored
Number of bytes remaining.
Array length
Data[1]
Endianness
1
0
0
1
20 0 0 0
255 255 255 255
3 0 0 0
1 0 0 0
2 0 0 0
3 0 0 0
This is 1,2,3 Note that this data is *not* necessarily 8 byte aligned.
Message tpye (for IPC only) 0 == 3:, 1 == 4: 2 == respones to 4:
K data tpye [-4,7]
Data[0]
Data[2]


Ignored
Number of bytes remaining.
Array length
Data[1]
Endianness
1
0
0
1
24 0 0 0
254 255 255 255
2 0 0 0
0 0 0 0 0 0 240 63
0 0 0 0 0 0 0 64
This is 1.0,2.0
Message tpye (for IPC only) 0 == 3:, 1 == 4: 2 == respones to 4:
K data tpye [-4,7]
Data[0]


Ignored
Number of bytes remaining.
Array length
Data[0] through Data[5]
Endianness
1
0
0
1
15 0 0 0
253 255 255 255
6 0 0 0
97
98
99
100
101
102
0
This is "abcdef" Note that this data is *not* necessarily


8 byte aligned.
Message tpye (for IPC only) 0 == 3:, 1 == 4: 2 == respones to 4:
K data tpye [-4,7]
Null Terminator
Ignored
Number of bytes remaining.
Array length
Data[0] through Data[2]
Endianness
1
0
0
1
17 0 0 0
252 255 255 255
3 0 0 0
97
0
98
0
99
0
This is `"ab",`"cd",`"ef" Note that this data is *not* necessarily 8 byte aligned.
Message tpye (for IPC only) 0 == 3:, 1 == 4: 2 == respones to 4:
K data tpye [-4,7]
Null Terminators for Data[0] through Data[2]
Ignored
Number of bytes remaining.
Array length
Data[0]
Pad to keep Data[0] 8 byte aligned
Data[1]
Endianness
1
0
0
1
32 0 0 0
0 0 0 0
2 0 0 0
4 0 0 0
97 98 99 100
0
0 0 0 0 0 0 0
1 0 0 0
1 0 0 0
This is `"abcd",1
Message tpye (for IPC only) 0 == 3:, 1 == 4: 2 == respones to 4:
K data tpye [-4,7]
K Type of Data[0]
Null Terminator for Data[0]
K Type of Data[1]

NOTE: the next to last diagram is incorrect. it is the representation of `a `b `c, not `ab `cd `ef

Dictionary structure (k type 5)

Dictionaries seem to be represented almost identically to a list of lists of length 3 (key, value, attribute). In any case, the binary representation is exactly the same as this, except for the type code, where the 0 is replaced with a 5.

kdata.pdf

kdata.vsd