Investigating an odd RDI file

Dan Kelley

2023-03-23

Introduction

A user tried reading what I'm told is a Sentinel RDI file, but found that it failed. I have the (91 Mb) file locally, and this document is the result of my investigation of its contents, after I confirmed that, indeed, `read.adp.rdi()` cannot read the file.

How does the file start?

According to RDI documentation in my possession (which includes documents from 2006), the first two bytes of an RDI file must be 0x7f 0x7f. Using `od` to inspect the file,

```
od -x ~/Downloads/wetransfer_dplsp000-000_2023-03-20_0847/DPLSP000.000 | head -1
```

yields

```
0000000      797f    0056    0100    0008    0103    2832    41cc    6029
```

so, clearly, this file does not match expectation.

Note that the above is as shown on a little-endian machine, so the byte order is 0x7f followed by 0x79.

It is convenient to use R to examine the file. The first 16 bytes are as follows (as expected, by transposing bytes in the `od` output).

```
f <- "~/Downloads/wetransfer_dplsp000-000_2023-03-20_0847/DPLSP000.000"
buf <- readBin(f, "raw", 1e6) # read a million bytes
buf[1:16]
```

```
##  [1] 7f 79 56 00 00 01 08 00 03 01 32 28 cc 41 29 60
```

Is the starting byte pair repeated?

Using

```
l1 <- which(buf == as.raw(0x7f))
l2 <- which(buf == as.raw(0x79))
n <- min(length(l1), length(l2))
df <- data.frame(l1[1:n], l2[1:n])
colnames(df) <- c("0x7f", "0x79")
df[1:10, ]
```

```
##    0x7f 0x79
## 1     1    2
## 2    89   73
## 3   179   90
## 4   269  180
```

```
## 5   359  270
## 6   449  360
## 7   529  450
## 8   539  540
## 9   629  630
## 10  719  720
```

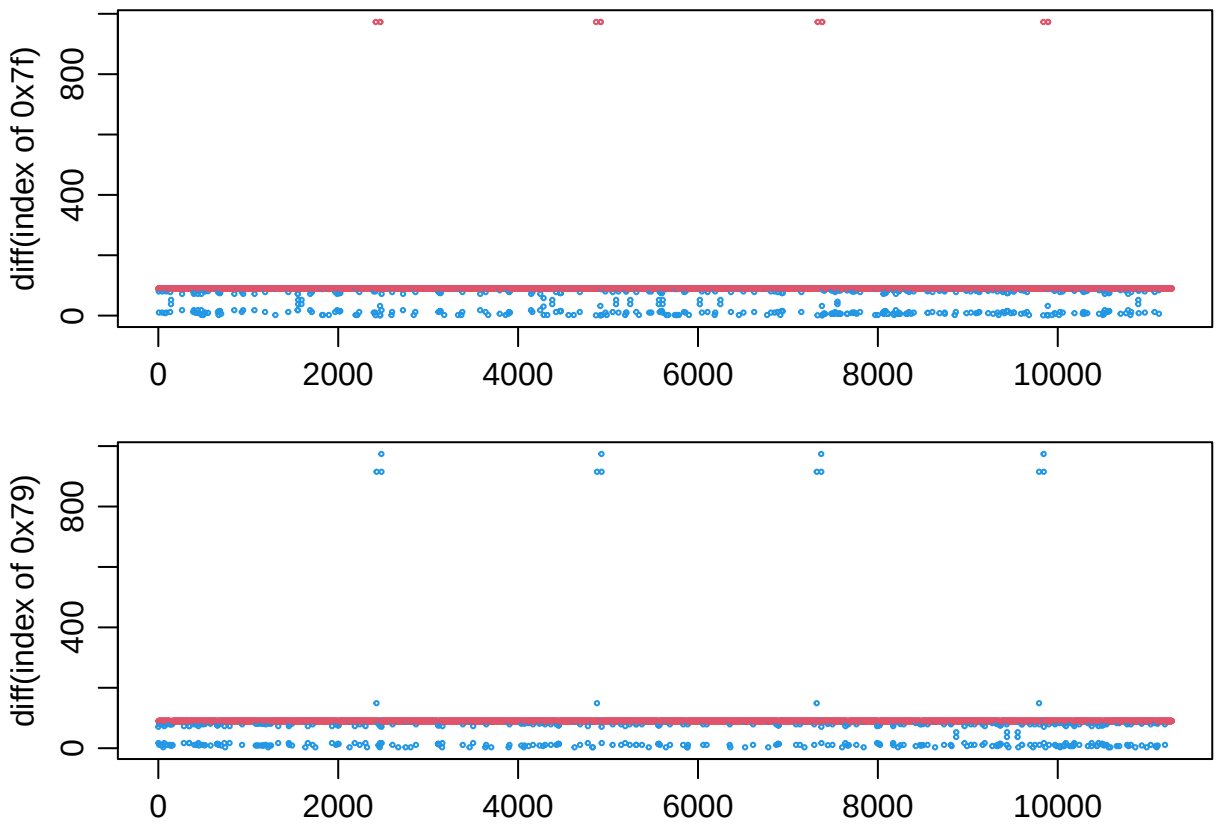
reveals a pairing pattern, with interruptions. (The interruptions are not unexpected, since these values might occur within data, not just at the starts of data chunks.) If we are willing to skip some values, we can see pairs here, at 1-2, then (skipping 73) at 89-90, then at 179-180, 269-270, 359-360, 449-450, and (skipping 529) 539-540, then at 629/630 and 719/720.

There is also a pattern to pair separation, with values 88, 90, 90, 90, 90, and 90 bytes. Perhaps the first chunk is of one type, and the others are of another type.

Is this byte pattern repeated later in the file?

We may examine byte distribution (over the first Mb, say) with

```
d1 <- diff(df[, 1])
d2 <- diff(df[, 2])
knownDiff <- c(90, 973) # by inspection
par(mfrow=c(2, 1), mar=c(2,3,1,1), mgp=c(2, 0.7, 0))
plot(d1, col=ifelse(d1 %in% knownDiff, 2, 4),
    xlab="", ylab="diff(index of 0x7f)", type="p", cex=0.3)
plot(d2, col=ifelse(d2 %in% knownDiff, 2, 4),
    xlab="", ylab="diff(index of 0x79)", type="p", cex=0.3)
```



In this graph, some values that seemed (by eye and histogram) to be common are shown in red. Again, a pattern is discernible, but it is not simple. Both byte values repeat frequently at an index separation of 90,

but the longer separation of 973 only occurs for 0x7f. And, more oddly, the high-separation 0x79 values occur at several separations.

Can oce read the file if the byte-pair is changed?

I have tried altering the existing oce code to recognize the start-pair 0x7f 0x79 but the values that get read do not make much sense. Frankly, this is not especially surprising. If I were an RDI engineer, and we had a chance to make a whole new format, I might change things. Perhaps new things need to be stored now. Perhaps the bit-packing has run into limitations, requiring an extra byte here and there. Perhaps the checksum algorithm would be more error-tolerant if it were changed. Changes along these lines cannot be inferred by looking at bytes and guessing. We need a manual that documents the format.

So, for now, oce cannot read these files.