Read PGN/SPN csv file

Hello verybody, finally we will dive into the package analysis. This is your first shot, so we decided to go with python due to simpliness. Please find the PGN/SPN csv list in your repository. First we will just read it into a dataframe, which we will use later for decoding the stream. The module pandas does actually all the work for us, we just need to pass the proper parameters.

In [1]:
```
import pandas as pd
from pathlib import Path
```

In [2]:
```
# set file path and read into pandas dataframe
# sorry guys, but currently we are not sure if we are allowed to publish you
r pgn spn list
pgnSpnPath = Path('/home/akarner/Downloads/JohnFearData/pgnSpn/pgn_spn.csv')
# make sure to pass the pipe as delimiter and remove all NaN fields
psDf = pd.read_csv(pgnSpnPath, sep='|', na_filter=False)
```

In [3]:
```
psDf
```
Out[3]:

	PGN#	PGNLabel	Acronym	PGNDescription	Multipacket	PGNLength	Priority	PGNReference
0	0	Torque/Speed Control 1	TSC1	NOTE - Retarder may be disabled by commanding ...	No	8	3	
1	0	Torque/Speed Control 1	TSC1	NOTE - Retarder may be disabled by commanding ...	No	8	3	
2	0	Torque/Speed Control 1	TSC1	NOTE - Retarder may be disabled by commanding ...	No	8	3	
3	0	Torque/Speed Control 1	TSC1	NOTE - Retarder may be disabled by commanding ...	No	8	3	
4	0	Torque/Speed Control 1	TSC1	NOTE - Retarder may be disabled by commanding ...	No	8	3	
...	...	...	...	...	...	...	...	...
5997	131067	Proprietary B - Page 1	PropB1_FB		Yes			
5998	131068	Proprietary B - Page 1	PropB1_FC		Yes			
5999	131069	Proprietary B - Page 1	PropB1_FD		Yes			
6000	131070	Proprietary B - Page 1	PropB1_FE		Yes			
6001	131071	Proprietary B - Page 1 (last entry)	PropB1_FF		Yes			

6002 rows × 18 columns

Read Wireshark (pcapng) file

We are going to use scapy module for package analysis in python

In [4]:
```
from scapy.all import *
```

In [5]:
```
# read your capture file
capturePath = '/home/akarner/tulocal/JohnFear/captures/13Feb2020.pcapng'
capture = rdpcap(capturePath)
```

In [6]:
```python
# get some information from a random package
#   -> this will be 2712 package in wireshark
pack = capture[2711]

# general about the package
print(pack)

# package fields
print(pack.fields)

# value from a field
print(pack.fields['src'])

# get raw payload
print(pack.payload)
```

```
b'\x00\x01\x01\x18\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0c\x91\xfe\xf
f\x8c\x08\x00\x00\x00\x83\xfc\xf4\xf0\xf0\xff\xff\xff'
{'pkttype': 1, 'lladdrtype': 280, 'lladdrlen': 0, 'src': b'\x00\x00\x00\x00\x
00\x00\x00\x00', 'proto': 12}
b'\x00\x00\x00\x00\x00\x00\x00\x00'
b'\x91\xfe\xff\x8c\x08\x00\x00\x00\x83\xfc\xf4\xf0\xf0\xff\xff\xff'
```

In [7]:
```python
# custom poc j1939 class, which will dissect { canId, pgn and data }
class j1939():
    def __init__(self, bdata):
        self.bdata = bdata
        self.canId = None
        self.pgn = None
        self.data = None

        self._readCanId()
        self._readPgn()
        self._readData()

    def _readCanId(self):
        canId = bytearray(4)
        # pack canId bytes
        for idx in range(0, 4):
            struct.pack_into('!B', canId, 4 - (idx+1), self.bdata[idx])

        # remove first 3 bits from msb -> 0x1f and mask
        canId[0] &= 0x1f
        self.canId = bytes(canId)

    def _readPgn(self):
        # remove last byte
        pgn = bytearray(self.canId[:-1])
        # reserve just first two bits from first byte
        pgn[0] &= 0x03
        self.pgn = int.from_bytes(pgn, byteorder='big')

    def _readData(self):
        self.data = self.bdata[8:]

    @staticmethod
    def getHexString(bytearr):
        return '0x' + ''.join('%02x' % b for b in bytearr)

    def __str__(self):
        return 'j1939[canId: %s, pgn: %i, data: %s]' % (j1939.getHexString(s
elf.canId), self.pgn, j1939.getHexString(self.data))
```

Finally we have prepared your poc j1939 class which extracts all the information out of the raw payload. Further all the pgn and spns are loaded to your pandas dataframe, so everything is prepared for linking them together.

Because pandas supports joining dataframes, we will transform the captured data into a dataframe and join by pgn the information to it.

In [8]:
```
# initiate new dataframe, seems like this takes very very long :)
capDf = pd.DataFrame(None, columns=['ccanid', 'cpgn', 'cdata'])
for cap in capture[:100]:
    m = j1939(cap.payload.load)
    capDf = capDf.append({'ccanid':m.canId, 'cpgn': m.pgn, 'cdata': m.data},
ignore_index=True)
```

In [9]:
```
capDf
```

Out[9]:

	ccanid	cpgn	cdata
0	b'\x00\x00\x00\x0c'	0	b'\x00\x00\x04\x00\x00\x00\x00\x00'
1	b'\x00\x00\x00\x0c'	0	b'\x00\x00\x04\x00\x00\x00\x00\x00'
2	b'\x00\x00\x00\x04'	0	b'\x00\x00\x00\x00\x00\x00\x00\x00'
3	b'\x18\xee\xff\x00'	61183	b'\x81\x90&\x04\x00\x00\x02 '
4	b'\x18\xee\xff\x14'	61183	b'\x93\x13 \x04\x00\x11\x00 '
...	...	...	...
95	b'\x18\xea\x8c\x14'	60044	b'\t\xff\x00\x00\x00\x00\x00\x00'
96	b'\x18\xea2\x14'	59954	b'\t\xff\x00\x00\x00\x00\x00\x00'
97	b'\x18\xea\x06\x14'	59910	b'\t\xff\x00\x00\x00\x00\x00\x00'
98	b'\x18\xea1\x14'	59953	b'\t\xff\x00\x00\x00\x00\x00\x00'
99	b'\x18\xea\x05\x14'	59909	b'\t\xff\x00\x00\x00\x00\x00\x00'

100 rows × 3 columns

```
In [10]: # let's start the join fun
         result = pd.merge(capDf, psDf, left_on='cpgn', right_on='PGN#')

         print(result)

         # dump the result as csv
         result.to_csv('result.csv', sep='|')
```

```
                  ccanid   cpgn                                cdata   PGN#
\
0    b'\x00\x00\x00\x0c'      0  b'\x00\x00\x04\x00\x00\x00\x00\x00'      0
1    b'\x00\x00\x00\x0c'      0  b'\x00\x00\x04\x00\x00\x00\x00\x00'      0
2    b'\x00\x00\x00\x0c'      0  b'\x00\x00\x04\x00\x00\x00\x00\x00'      0
3    b'\x00\x00\x00\x0c'      0  b'\x00\x00\x04\x00\x00\x00\x00\x00'      0
4    b'\x00\x00\x00\x0c'      0  b'\x00\x00\x04\x00\x00\x00\x00\x00'      0
..                   ...    ...                                  ...    ...
363  b'\x18\xfe\xee\x00'  65262        b'>5\xff\xff\xff\xff\xff\xff'  65262
364  b'\x18\xfe\xee\x00'  65262        b'>5\xff\xff\xff\xff\xff\xff'  65262
365  b'\x18\xfe\xee\x00'  65262        b'>5\xff\xff\xff\xff\xff\xff'  65262
366  b'\x18\xfe\xee\x00'  65262        b'>5\xff\xff\xff\xff\xff\xff'  65262
367  b'\x18\xef\x00\x06'  61184     b'\xf21\xff\xff\xcd\xf0\xff\xff'  61184

                   PGNLabel Acronym  \
0    Torque/Speed Control 1    TSC1
1    Torque/Speed Control 1    TSC1
2    Torque/Speed Control 1    TSC1
3    Torque/Speed Control 1    TSC1
4    Torque/Speed Control 1    TSC1
..                      ...     ...
363    Engine Temperature 1     ET1
364    Engine Temperature 1     ET1
365    Engine Temperature 1     ET1
366    Engine Temperature 1     ET1
367           Proprietary A   PropA

                                        PGNDescription Multipacket PGNLength
\
0    NOTE - Retarder may be disabled by commanding ...          No         8
1    NOTE - Retarder may be disabled by commanding ...          No         8
2    NOTE - Retarder may be disabled by commanding ...          No         8
3    NOTE - Retarder may be disabled by commanding ...          No         8
4    NOTE - Retarder may be disabled by commanding ...          No         8
..                                                 ...         ...       ...
363                                                             No         8
364                                                             No         8
365                                                             No         8
366                                                             No         8
367  This proprietary PG uses the Destination Speci...         Yes

    Priority  ... SPNPos   SPN  \
0          3  ...    1.1   695
1          3  ...    1.3   696
2          3  ...    1.5   897
3          3  ...    2-3   898
4          3  ...      4   518
..       ...  ...    ...   ...
363        6  ...    3-4   175
364        6  ...    5-6   176
365        6  ...      7    52
366        6  ...      8  1134
367           ...    1-8  2550

                                            SPNName  \
0                      Engine Override Control Mode
1         Engine Requested Speed Control Conditions
2                    Override Control Mode Priority
3                Engine Requested Speed/Speed Limit
4              Engine Requested Torque/Torque Limit
..                                              ...
363                        Engine Oil Temperature 1
364             Engine Turbocharger Oil Temperature
365                  Engine Intercooler Temperature
366     Engine Charge Air Cooler Thermostat Opening
367  Manufacturer Specific Information (PropA_PDU1)

                                        SPNDescription          SPNLength  \
In [ ]:
```