```
#!/usr/bin/env python
import sys
try:
        import os, glob, sys, getopt
        import tempfile
        import tarfile
        from datetime import datetime
        from SrbRegisterUtil import execMe,execSMe,PyIsSinit,CheckPyVersion
except:
        raise "Check you local Python Version. A minimum Python version required"

""" A script that takes a tar bundle (tar.gz) file as argument, read the file listing, put said tar bundle file into SRB and register metadata """


SCHEME_NAME = "Name_Value"

def usage():
        print """SrbTarManifest.py
        usage: SrbTarManifest.py <option> <file or group of files>
                -h: This help
                -m: Include metadata attribute to my file
                -f: Force flag. Overwrite older version in SRB.
                --version: Print the script version

        Example
                SrbTarManifest.py 310_archv_1234_440.tar.gz 310_archm_1234_440.tar.gz
                SrbTarManifest.py -f 310_arch*.tar.gz
                SrbTarManifest.py -m -f 310_arch*.tar.gz
        """
# end function


def version():
        print """$Header: /cvs_repository/customers/HPCMP/testbed/NavyPilot/SrbTarManifest.py,v 1.6 2013/06/03 20:44:28 martin Exp $"""
# end function


def parseAndRegister(tarfile):
        """
        A function to take a well formatted tarfile and register parts of
        the name as metadata in SRB MCAT database

        example: 310_archv_2012062618_2012062700.tar.gz

        parsed:
                ExperimentNum = 310
                OutputType = archv
                DateRun = 2012062618 (parsed to ISO 8601)
                DateValid = 2012062700 (parsed to ISO 8601)
        """
        rc = 0
        strTarFile = tarfile.replace('.tar.gz','') # chop extension
        lisComp = strTarFile.split('_')
```

```
54
55          experimentNum = lisComp[0]
56          outputType = lisComp[1]
57          dateRun = lisComp[2]
58          dateValid = lisComp[3]
59
60          if (rc == 0):
61                  (rc,out) = execSMe("Sscheme -w -scheme {0} -val
    \"Name[0]::ExperimentNum, Value[0]::\"{1}\"\" {2}".format(SCHEME_NAME,
    experimentNum, tarfile))
62
63
64          if (rc == 0):
65                  (rc,out) = execSMe("Sscheme -w -scheme {0} -val
    \"Name[1]::OutputType, Value[1]::\"{1}\"\" {2}".format(SCHEME_NAME, outputType,
    tarfile))
66
67
68          if (rc == 0):
69                  (rc,out) = execSMe("Sscheme -w -scheme {0} -val
    \"Name[2]::DateRun, Value[2]::\"{1}\"\" {2}".format(SCHEME_NAME, dateRun,
    tarfile))
70
71
72          if (rc == 0):
73                  (rc,out) = execSMe("Sscheme -w -scheme {0} -val
    \"Name[3]::DateValid, Value[3]::\"{1}\"\" {2}".format(SCHEME_NAME, dateValid,
    tarfile))
74
75
76          if (rc != 0):
77                  print "Warning: One or more metadata from tar filename failed to
    register"
78
79
80  # end function
81
82  def main():
83
84          if not CheckPyVersion():
85                  sys.exit(1)
86
87          forceFlag = 0
88          metadataFlag = 0
89
90          try:
91                  opts,args = getopt.getopt(sys.argv[1:], "hmfv", ['version'])
92          except getopt.GetoptError, err:
93                  print str(err)
94                  usage()
95                  sys.exit(2)
96
97          for arg,val in opts:
98                  if arg == "-h":
99                          usage()
100                         sys.exit()
101                 elif arg == "-f":
```

```
102                         forceFlag = 1
103                 elif arg == "-m":
104                         metadataFlag = 1
105                 elif arg == "--version" or arg == "-v":
106                         version()
107                         sys.exit()
108         #end for
109 
110 
111         if (PyIsSinit()):
112                 print "Sinit is OK..."
113                 status = 0
114         else:
115                 print "No SRB session detected. Have you run Sinit yet?"
116                 sys.exit(2)
117 
118 
119         if len(args) > 0:
120 
121                 print "Stage 1: Sput {0} to SRB current collection".format('
    '.join(args))
122 
123                 # Construct command string cmd
124                 # with force flag
125                 #   Sput -f abel.tar.gz .
126                 # without force flag
127                 #   Sput abel.tar.gz .
128 
129                 cmd = "Sput "
130                 if forceFlag == 1:
131                         cmd = cmd + "-f "
132                 cmd = cmd + "{0} .".format(' '.join(args))
133 
134 
135                 (rc,out) = execSMe(cmd)
136                 if (rc != 0):
137                         print "Error putting file to SRB!"
138                         rc = -1
139                 else:
140                         print "Put file to SRB successful"
141 
142                 if (rc == 0):
143                         print "Stage 2: Reading tar file manifest"
144                         # for each input file do
145                         for tarfilenames in args:
146 
147                                 lrealfile = glob.glob(tarfilenames) #extract
    real file from wildcard filename
148                                 for realfile in lrealfile:
149 
150                                         tar = tarfile.open(realfile,'r')
151                                         rows = 0
152                                         #create temporary file
153                                         fd, path = tempfile.mkstemp()
154                                         manifest =
    "unix_mode,unix_uid,unix_gid,file_size,date_modify,file_name\n"
155                                         for tarinfo in tar:
```

```
156                                                     if ((rows % 5000) == 0):
157                                                             #bug 1944 memory error
    if writing too many rows to string manifest
158                                                             os.write(fd, manifest)
159                                                             manifest = ''
160                                                     #end if
161
162                                                     manifest = manifest +
    oct(tarinfo.mode)[0:8] + ',' + str(tarinfo.uid)[0:8] + ',' + \
163                                                             str(tarinfo.gid)[0:8] +
    ',' +  str(tarinfo.size)[0:14] + ','
    +(datetime.fromtimestamp(tarinfo.mtime)).__str__() + '.0000,' + \
164                                                             tarinfo.name[0:256] +
    '\n'
165                                                     rows = rows + 1
166                                             #end for loop
167
168                                             os.write(fd, manifest)
169                                             os.close(fd)
170
171
172                                             (rc,out) = execMe("Sscheme -w -scheme
    TOC -file {0} {1}".format(path, realfile))
173                                             if (rc != 0):
174                                                     print "Error writing manifest
    for {0}".format(realfile)
175
176
177                                             #os.remove(path) #clean up
178                                             if (rc == 0 ) and (metadataFlag == 1):
179                                                     rc = parseAndRegister(realfile)
180                                             #end if
181                                     #end-for inside loop
182
183                             #end-for loop
184                             if rc == 0:
185                                     print "Success. Tar.gz manifest has been written
    to MCAT for all input files"
186                     #end if passed second gate if (rc == 0)
187             #end if
188
189
190 #end of function
191
192 if __name__=="__main__":
193             main()
194
195
```