Automating Fuzzers

The target is **VLC media player**

VLC is an and open source cross-platform multimedia player and framework that plays most multimedia files

Attack surface on transcode files

The focus will be directed towards transcode files and we sending unexpected inputs and see what the vlc will do

Generating Test Case by Radamsa and go fuzzing by script python

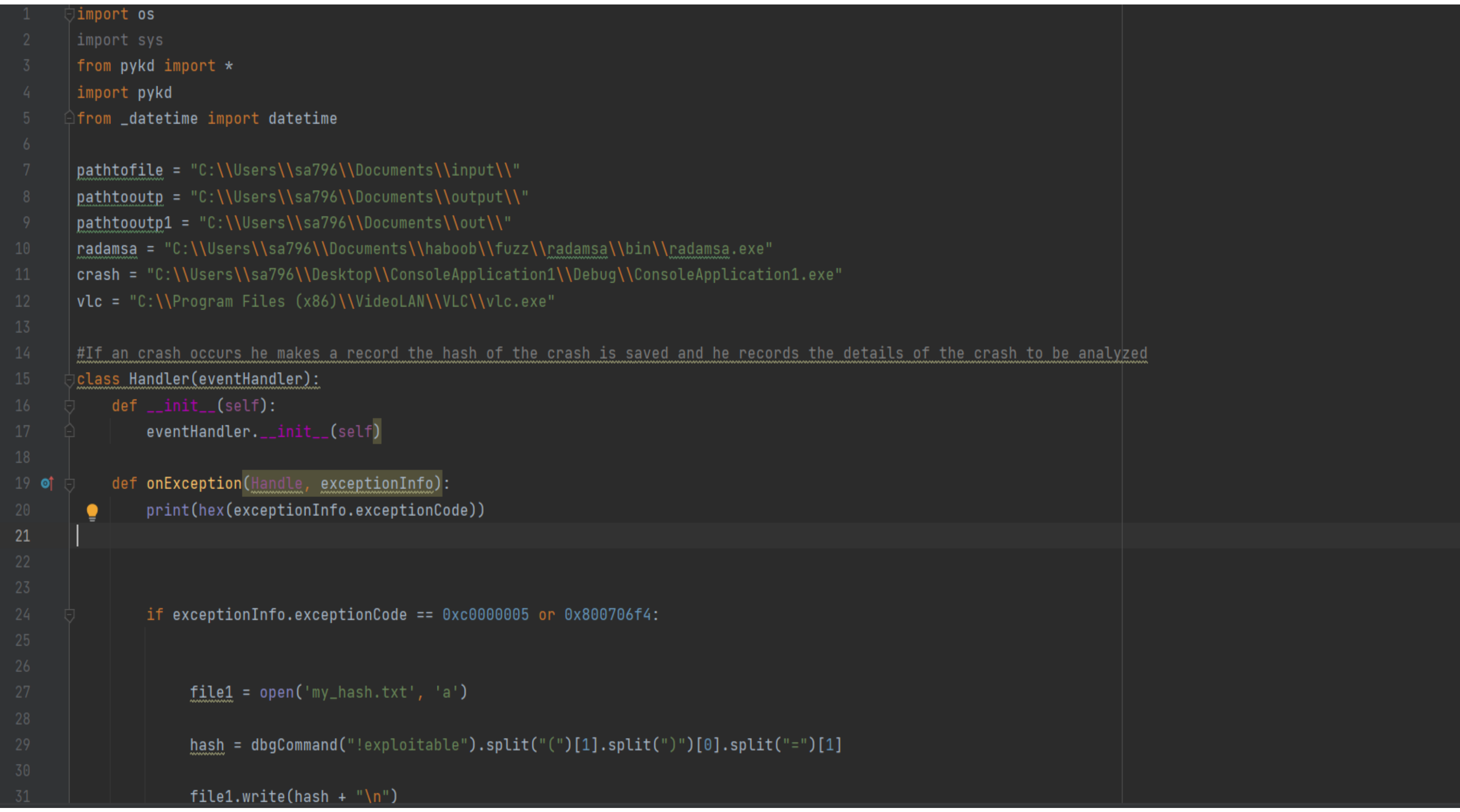
file types uses like (avi,mp4,avo. atc)

Fuzzing techniques and tools used

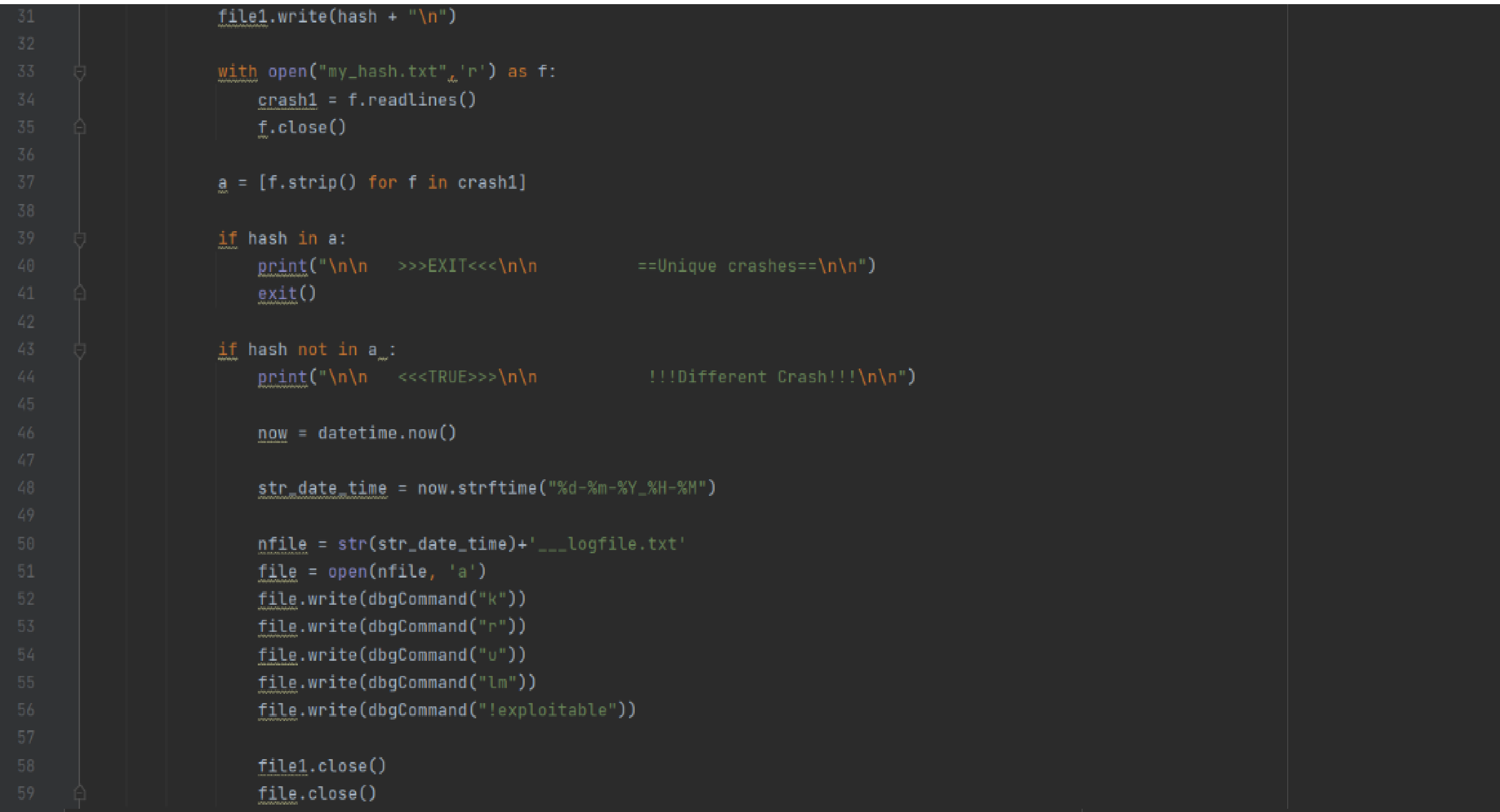
- Radamsa
- Script python

A demonstration of the final product

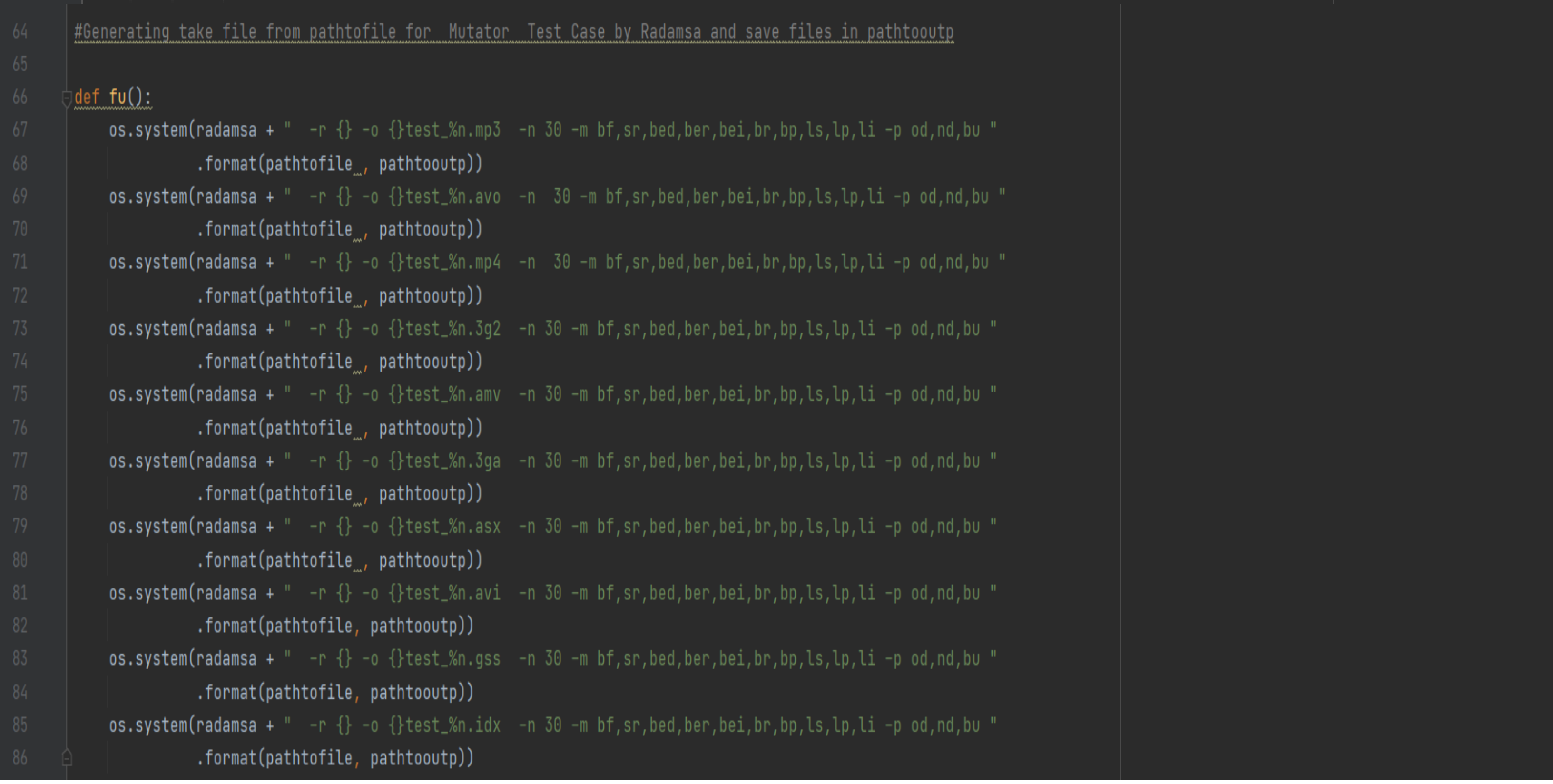
I will attach the python script used in the fuzzing



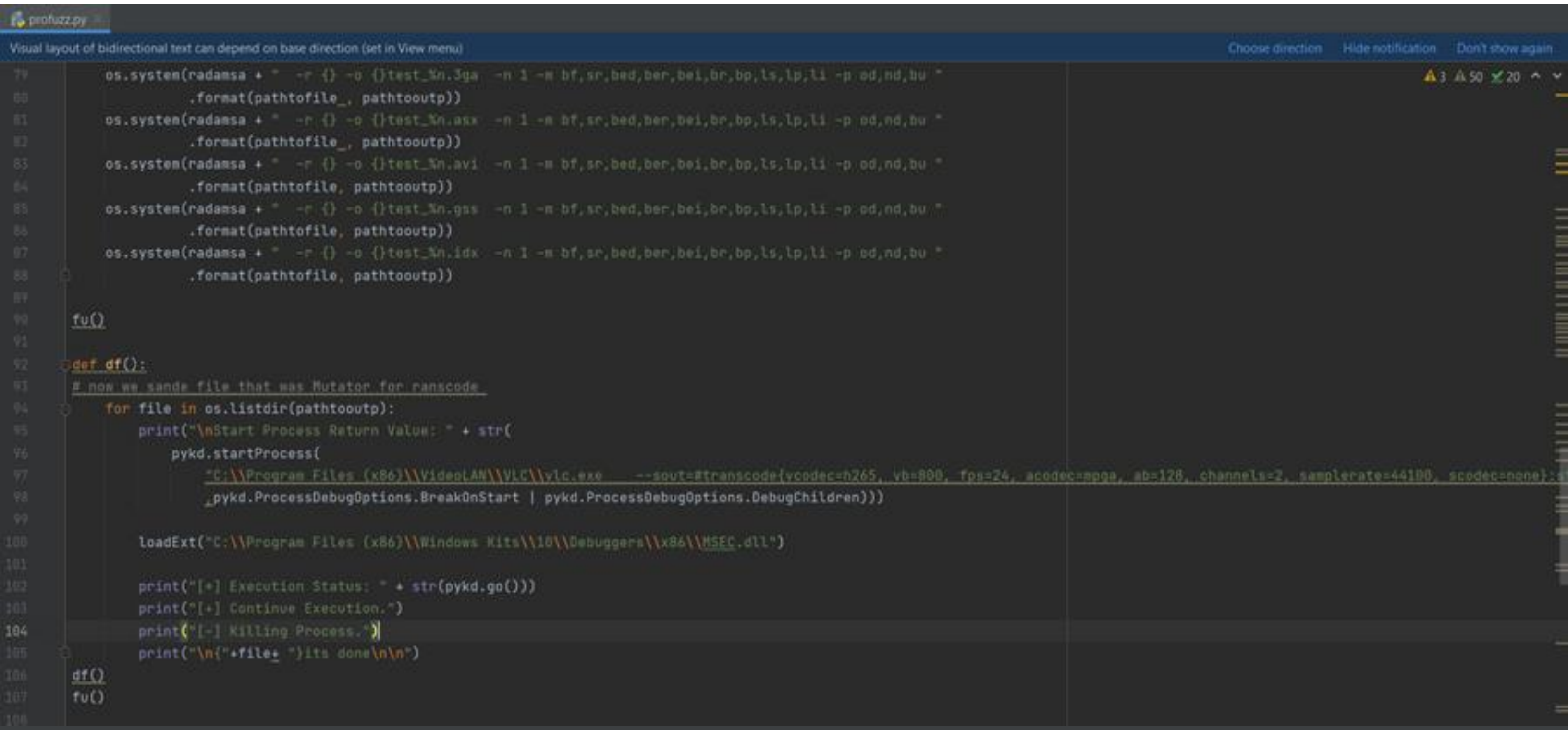
```python
import os
import sys
from pykd import *
import pykd
from _datetime import datetime

pathtofile = "C:\\Users\\sa796\\Documents\\input\\"
pathtooutp = "C:\\Users\\sa796\\Documents\\output\\"
pathtooutp1 = "C:\\Users\\sa796\\Documents\\out\\"
radamsa = "C:\\Users\\sa796\\Documents\\haboob\\fuzz\\radamsa\\bin\\radamsa.exe"
crash = "C:\\Users\\sa796\\Desktop\\ConsoleApplication1\\Debug\\ConsoleApplication1.exe"
vlc = "C:\\Program Files (x86)\\VideoLAN\\VLC\\vlc.exe"

#If an crash occurs he makes a record the hash of the crash is saved and he records the details of the crash to be analyzed
class Handler(eventHandler):
    def __init__(self):
        eventHandler.__init__(self)

    def onException(Handle, exceptionInfo):
        print(hex(exceptionInfo.exceptionCode))



        if exceptionInfo.exceptionCode == 0xc0000005 or 0x800706f4:


            file1 = open('my_hash.txt', 'a')

            hash = dbgCommand("!exploitable").split("(")[1].split(")")[0].split("=")[1]

            file1.write(hash + "\n")
```



```
            file1.write(hash + "\n")

            with open("my_hash.txt",'r') as f:
                crash1 = f.readlines()
                f.close()

            a = [f.strip() for f in crash1]

            if hash in a:
                print("\n\n   >>>EXIT<<<\n\n          ==Unique crashes==\n\n")
                exit()

            if hash not in a :
                print("\n\n   <<<TRUE>>>\n\n           !!!Different Crash!!!\n\n")

                now = datetime.now()

                str_date_time = now.strftime("%d-%m-%Y_%H-%M")

                nfile = str(str_date_time)+'___logfile.txt'
                file = open(nfile, 'a')
                file.write(dbgCommand("k"))
                file.write(dbgCommand("r"))
                file.write(dbgCommand("u"))
                file.write(dbgCommand("lm"))
                file.write(dbgCommand("!exploitable"))

                file1.close()
                file.close()
```

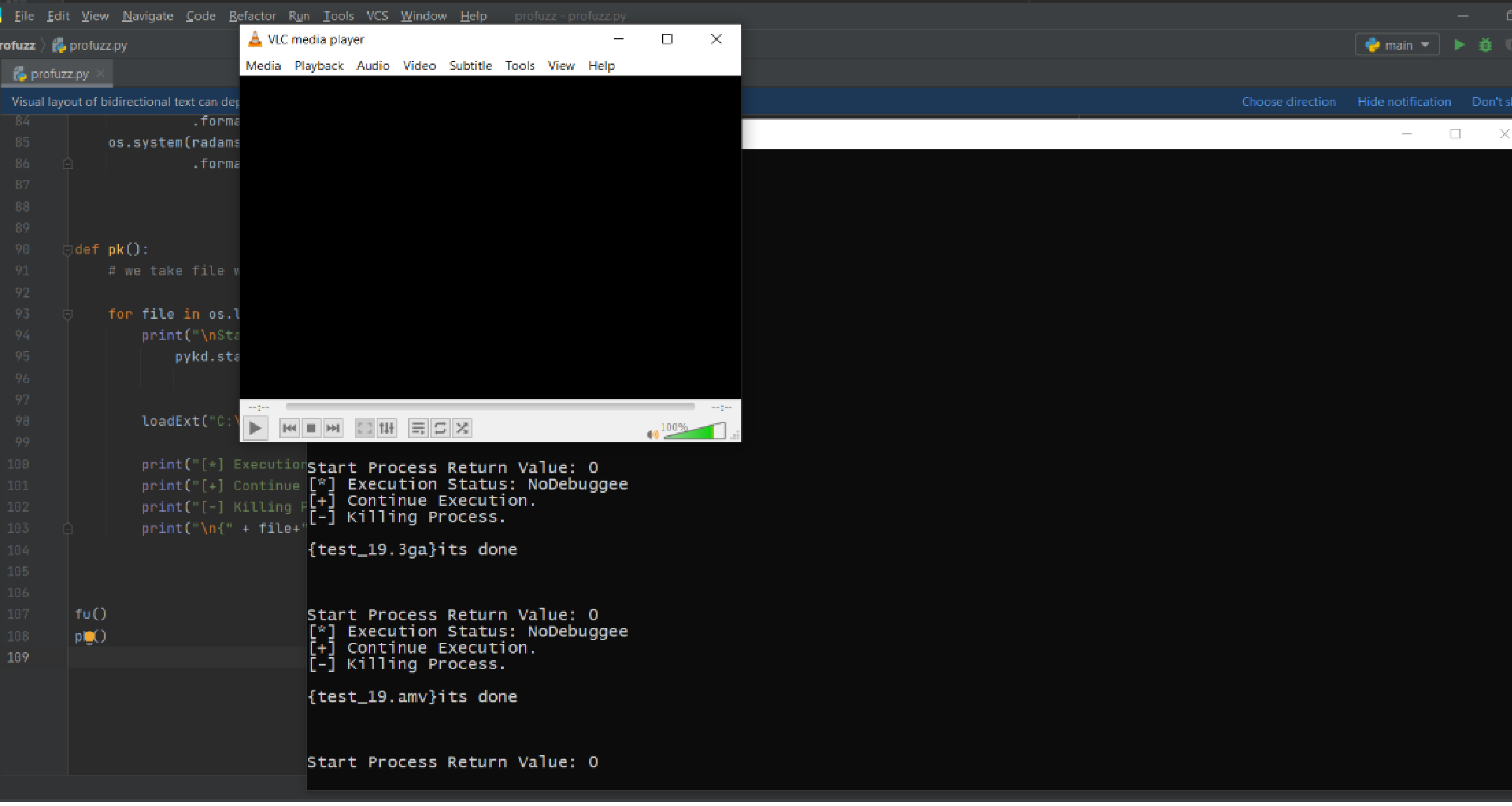


```
#Generating take file from pathtofile for  Mutator  Test Case by Radamsa and save files in pathtooutp

def fu():
    os.system(radamsa + "  -r {} -o {}test_%n.mp3  -n 30 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
              .format(pathtofile , pathtooutp))
    os.system(radamsa + "  -r {} -o {}test_%n.avo  -n  30 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
              .format(pathtofile , pathtooutp))
    os.system(radamsa + "  -r {} -o {}test_%n.mp4  -n  30 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
              .format(pathtofile , pathtooutp))
    os.system(radamsa + "  -r {} -o {}test_%n.3g2  -n 30 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
              .format(pathtofile , pathtooutp))
    os.system(radamsa + "  -r {} -o {}test_%n.amv  -n 30 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
              .format(pathtofile , pathtooutp))
    os.system(radamsa + "  -r {} -o {}test_%n.3ga  -n 30 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
              .format(pathtofile , pathtooutp))
    os.system(radamsa + "  -r {} -o {}test_%n.asx  -n 30 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
              .format(pathtofile , pathtooutp))
    os.system(radamsa + "  -r {} -o {}test_%n.avi  -n 30 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
              .format(pathtofile, pathtooutp))
    os.system(radamsa + "  -r {} -o {}test_%n.gss  -n 30 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
              .format(pathtofile, pathtooutp))
    os.system(radamsa + "  -r {} -o {}test_%n.idx  -n 30 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
              .format(pathtofile, pathtooutp))
```



Visual layout of bidirectional text can depend on base direction (set in View menu)

```
        os.system(radamsa + "  -r {} -o {}test_%n.3ga  -n 1 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
                  .format(pathtofile_, pathtooutp))
        os.system(radamsa + "  -r {} -o {}test_%n.asx  -n 1 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
                  .format(pathtofile_, pathtooutp))
        os.system(radamsa + "  -r {} -o {}test_%n.avi  -n 1 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
                  .format(pathtofile, pathtooutp))
        os.system(radamsa + "  -r {} -o {}test_%n.gss  -n 1 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
                  .format(pathtofile, pathtooutp))
        os.system(radamsa + "  -r {} -o {}test_%n.idx  -n 1 -m bf,sr,bed,ber,bei,br,bp,ls,lp,li -p od,nd,bu "
                  .format(pathtofile, pathtooutp))

fu()

def df():
# now we sande file that was Mutator for ranscode
    for file in os.listdir(pathtooutp):
        print("\nStart Process Return Value: " + str(
            pykd.startProcess(
                "C:\\Program Files (x86)\\VideoLAN\\VLC\\vlc.exe    --sout=#transcode{vcodec=h265, vb=800, fps=24, acodec=mpga, ab=128, channels=2, samplerate=44100, scodec=none}:
                ,pykd.ProcessDebugOptions.BreakOnStart | pykd.ProcessDebugOptions.DebugChildren)))

        loadExt("C:\\Program Files (x86)\\Windows Kits\\10\\Debuggers\\x86\\MSEC.dll")

        print("[+] Execution Status: " + str(pykd.go()))
        print("[+] Continue Execution.")
        print("[-] Killing Process.")
        print("\n{"+file+ "}its done\n\n")
df()
fu()
```



File Edit View Navigate Code Refactor Run Tools VCS Window Help profuzz - profuzz.py
profuzz profuzz.py
main
VLC media player
Media Playback Audio Video Subtitle Tools View Help
100%

```
            .forma
    os.system(radams
            .forma



def pk():
    # we take file w

    for file in os.l
        print("\nSta
            pykd.sta


        loadExt("C:\

        print("[*] Execution
        print("[+] Continue
        print("[-] Killing P
        print("\n{" + file+



fu()
pk()
```

```
Start Process Return Value: 0
[*] Execution Status: NoDebuggee
[+] Continue Execution.
[-] Killing Process.

{test_19.3ga}its done


Start Process Return Value: 0
[*] Execution Status: NoDebuggee
[+] Continue Execution.
[-] Killing Process.

{test_19.amv}its done


Start Process Return Value: 0
```

Each file has its own number so that if this crash occurs, we will know the file that caused the crash