setup.py py2exe

```
-*- coding: UTF-8 -*-
#
#

import wx
import os
import sys
import time
import runpy
import socket
import shutil
import gettext
import sqlite3
import threading
import subprocess

import wx.lib.agw.pyprogress as PP
from BeautifulSoup import BeautifulSoup

# Set script PATH
sys.path.append("./usbmuxd")
sys.path.append("./scripts")
sys.path.append("./")

from util.ramdiskclient import RamdiskToolClient
from keystore.keybag import Keybag

# set i18n
_ = wx.GetTranslation

# Hook debug window
class LogWindow(wx.PyOnDemandOutputWindow):
   def __init__(self):
      wx.PyOnDemandOutputWindow.__init__(self, title = "Message Window")

   def write(self, text):
      if self.frame is None:

         # Bug Fix with Init self.frame fast... XD
         self.frame = "____"
         if not wx.Thread_IsMain():
            wx.CallAfter(self.CreateOutputWindow, text)
         else:
            self.CreateOutputWindow(text)
      else:
         if not wx.Thread_IsMain():
            wx.CallAfter(self.__write, text)
```

```
            else:
                self.__write(text)

    def __write(self, text):
        try:
            self.text.AppendText(text)
        except AttributeError:
            pass

        for item in self.triggers:
            if item in text:
                self.frame.Raise()
                break

# hook print statement
class HookStdOut(object):
    def __init__(self, *argv, **argd):
        object.__init__(self)

    def write(self, s):
        try:
            sys.__stdout__.write(s)
        except:
            pass

class MyApp(wx.App):
    outputWindowClass = LogWindow
    def __init__(self, redirect=True):
        wx.App.__init__(self, redirect)

class JB_Dialog(wx.Dialog):
    def __init__(self, parent, ID, title, size=wx.DefaultSize, pos=wx.DefaultPosition,
style=wx.DEFAULT_DIALOG_STYLE,):
        pre = wx.PreDialog()
        pre.Create(parent, ID, title, pos, size, style)
        self.PostCreate(pre)

        # set description str
        label = wx.StaticText(self, -1, _("\nPlease select your decide type."))
        sizer = wx.BoxSizer(wx.VERTICAL)
        sizer.Add(label, 0, wx.ALIGN_CENTRE|wx.ALL, 5)

        # set select bar
        self.l = wx.Choice(self,
                    -1,
                    (100, 50),
                    choices=["Iphone 4 (GSM)", "Iphone 4 (CDMA)", "Iphone 3GS ", "Ipod Touch 3G", "Ipod
Touch 4G", "Ipad 1"]
                    )
        self.l.SetSelection(0)
```

```
        sizer.Add(self.l, 2, wx.EXPAND|wx.ALL, 10)
        box = wx.BoxSizer(wx.HORIZONTAL)
        btnsizer = wx.StdDialogButtonSizer()

        # set OK button
        btn = wx.Button(self, wx.ID_OK)
        btn.SetDefault()
        btnsizer.AddButton(btn)

        # set CANCEL button
        btn = wx.Button(self, wx.ID_CANCEL)
        btnsizer.AddButton(btn)
        btnsizer.Realize()

        sizer.Add(btnsizer, 0, wx.ALIGN_CENTER_VERTICAL|wx.ALL, 5)
        self.SetSizer(sizer)
        sizer.Fit(self)

class LangChoice(wx.Dialog):
    def __init__(self, parent, ID, title, size=wx.DefaultSize, pos=wx.DefaultPosition,
style=wx.DEFAULT_DIALOG_STYLE,):
        pre = wx.PreDialog()
        pre.Create(parent, ID, title, pos, size, style)
        self.PostCreate(pre)
        sizer = wx.BoxSizer(wx.VERTICAL)

        # set select bar
        self.l = wx.Choice(self,
                           -1,
                           (100, 50),
                           choices=["Chinese", "English", "Japanese"]
                          )
        self.l.SetSelection(0)
        sizer.Add(self.l, 2, wx.EXPAND|wx.ALL, 10)
        box = wx.BoxSizer(wx.HORIZONTAL)
        btnsizer = wx.StdDialogButtonSizer()

        # set OK button
        btn = wx.Button(self, wx.ID_OK)
        btn.SetDefault()
        btnsizer.AddButton(btn)

        # set CANCEL button
        btn = wx.Button(self, wx.ID_CANCEL)
        btnsizer.AddButton(btn)
        btnsizer.Realize()

        sizer.Add(btnsizer, 0, wx.ALIGN_CENTER_VERTICAL|wx.ALL, 5)
        self.SetSizer(sizer)
        sizer.Fit(self)
```

```
class File_Dialog(wx.Dialog):
   def __init__(self, parent, ID, title, size=wx.DefaultSize, pos=wx.DefaultPosition,
style=wx.DEFAULT_DIALOG_STYLE,):
      pre = wx.PreDialog()
      pre.Create(parent, ID, title, pos, size, style)
      self.PostCreate(pre)
      sizer = wx.BoxSizer(wx.VERTICAL)

      # set select bar
      global c
      global cx
      c.execute("SELECT * FROM projects")

      sampleList = []
      for i in c.fetchall():
         sampleList.append( "%d. %s #%s" % (i[0], i[3], i[4]) )

      self.l = wx.ListBox(self, -1, size=(400, 150), choices=sampleList, style=wx.LB_SINGLE)
      if len(sampleList) > 0:
         self.l.SetSelection(0)
      sizer.Add(self.l, 2, wx.EXPAND|wx.ALL, 10)

      box = wx.BoxSizer(wx.HORIZONTAL)

      btn1 = wx.Button(self, -1, _("Open"))
      btn2 = wx.Button(self, -1, _("Delete"))
      btn3 = wx.Button(self, -1, _("Import"))

      box.Add(btn1)
      box.Add(btn2)
      box.Add(btn3)

      btn1.Bind(wx.EVT_BUTTON, self.open)
      btn2.Bind(wx.EVT_BUTTON, self.delete)
      btn3.Bind(wx.EVT_BUTTON, self.imp)

      sizer.Add(box, 0, wx.ALIGN_CENTER_VERTICAL|wx.ALL, 5)
      self.SetSizer(sizer)
      sizer.Fit(self)

   def open(self, evt):
      s = self.l.GetStringSelection()
      if s:
         s = s.split(".", 1)[0]
         s = int(s)

         s = c.execute("SELECT * FROM projects WHERE id=%d" % s).fetchone()
         global udid, dataVolumeUUID
         udid = s[1]
```

```
        dataVolumeUUID = s[2]
        name = s[3]
        frame.statusbar.SetStatusText("Project name is %s" % name)

    self.Destroy()

def delete(self, evt):
    s = self.l.GetStringSelection()
    if s:
        s = s.split(".", 1)[0]
        s = int(s)

        s = c.execute("DELETE FROM projects where id=%d" % s)
        cx.commit()

    self.Destroy()

def imp(self, evt):
    dlg = wx.DirDialog(self, "Choose import directory:",
                       style=wx.DD_DEFAULT_STYLE
                       | wx.DD_DIR_MUST_EXIST
                       #| wx.DD_CHANGE_DIR
                       )

    if dlg.ShowModal() == wx.ID_OK:
        directory = dlg.GetPath()
    else:
        directory = None
    dlg.Destroy()


    if directory:
        udid = directory.split("\\")[-1]
        __ = os.listdir(directory)

        vid = ""
        for i in __:
            if i.endswith(".plist"):
                vid = i.split(".")[0]
                break

        if vid and udid:
            value = wx.GetTextFromUser(_("\nPlease enter case name."), _("Message!"), "")
            if not value:
                name = ""
            else:
                name = value
        else:
            wx.MessageBox("File not completed", _("Error!"), style=wx.ICON_HAND)
```

```
        if vid and udid and name:
            _time = " ".join(time.asctime().split(" ")[2:])
            c.execute( "INSERT INTO projects(udid, vid, name, time) values(?, ?, ?, ?)", (udid,
dataVolumeUUID, name, _time) )
            cx.commit()

    self.Destroy()


class main(wx.Frame):
  def __init__(self, parent, id, title, size):
    wx.Frame.__init__(self, parent, id, title, size = size)
    self._main()

  def CreateElement(self):
    panel = wx.Panel(self, -1, style=wx.RAISED_BORDER)
    self.option0 = wx.Button(panel, -1, _("Start"), style=wx.NO_BORDER)
    self.option1 = wx.Button(panel, -1, _("Change language"), style=wx.NO_BORDER)
    self.option2 = wx.Button(panel, -1, _("Using SSH over USB, Setup Usbmux. (Optional)"),
style=wx.NO_BORDER)
    self.option3 = wx.Button(panel, -1, _("Using JB Exploit to Load Forensic Ramdisk."),
style=wx.NO_BORDER)
    self.option4 = wx.Button(panel, -1, _("Brute Force the Passcode."), style=wx.NO_BORDER)
    self.option5 = wx.Button(panel, -1, _("Export the Important Password Report from Keychain."),
style=wx.NO_BORDER)
    self.option6 = wx.Button(panel, -1, _("Mirror IOS Data Partition."), style=wx.NO_BORDER)
    self.option7 = wx.Button(panel, -1, _("Decrypt IOS Data Pratition Image."), style=wx.NO_BORDER)
    self.option8 = wx.Button(panel, -1, _("Recover Deleted Files from Decrypted Image."),
style=wx.NO_BORDER)
    self.option9 = wx.Button(panel, -1, _("Deeply Revocer Deleted Files from Decrypted Image."),
style=wx.NO_BORDER)
    self.option10 = wx.Button(panel, -1, _("About"), style=wx.NO_BORDER)

    box = wx.BoxSizer(wx.VERTICAL)

    # dirty way to eval button
    for i in range(0, 11):
      eval("box.Add(self.option%d, 5, wx.EXPAND | wx.ALL, 4)" % i)

    panel.SetSizer(box)

  def OnShow(self, evt):
    value = wx.GetTextFromUser(_("\nPlease enter case number (12 digits)."), _("Message!"), "")
    if value:
      try:
        value = int(value)
      except:
        wx.MessageBox("Case number must be digit.", _("Error!"), style=wx.ICON_HAND)
        value = None
```

```
    if value:
        # do_lookup(value)
        pass

def ChgLang(self, evt):
    dlg = LangChoice(self, -1, _("Change Language"), size=(350, 200))
    dlg.CenterOnScreen()
    val = dlg.ShowModal()
    if val == wx.ID_OK:
        choice = dlg.l.CurrentSelection
        if choice == 0:
            mylocale.AddCatalog('cht')
        elif choice == 1:
            mylocale.AddCatalog('en')
        elif choice == 2:
            mylocale.AddCatalog('jp')


    dlg.Destroy()

def JB(self, evt):
    dlg = JB_Dialog(self, -1, _("Using JB Exploit to Load Forensic Ramdisk"), size=(350, 200))
    dlg.CenterOnScreen()
    val = dlg.ShowModal()

    if val == wx.ID_OK:
        choice = dlg.l.CurrentSelection
        if choice == 0:     # Iphone 4 (GSM)
            i = "iPhone3,1_5.0_9A334_Restore.ipsw"
            r = "iPhone3,1_5.0_9A334_Restore.dmg"
            k = "iPhone3,1_5.0_9A334_Restore.patched"
        elif choice == 1:   # Iphone 4 (CDMA)
            i = "iPhone3,3_5.0_9A334_Restore.ipsw"
            r = "iPhone3,3_5.0_9A334_Restore.dmg"
            k = "iPhone3,3_5.0_9A334_Restore.patched"
        elif choice == 2:   # Iphone 3GS
            i = "iPhone2,1_5.0_9A334_Restore.ipsw"
            r = "iPhone2,1_5.0_9A334_Restore.dmg"
            k = "iPhone2,1_5.0_9A334_Restore.patched"
        elif choice == 3:   # Ipod Touch 3G
            i = "iPod3,1_5.0_9A334_Restore.ipsw"
            r = "iPod3,1_5.0_9A334_Restore.dmg"
            k = "iPod3,1_5.0_9A334_Restore.patched"
        elif choice == 4:   # Ipod Touch 4G
            i = "iPod4,1_5.0_9A334_Restore.ipsw"
            r = "iPod4,1_5.0_9A334_Restore.dmg"
            k = "iPod4,1_5.0_9A334_Restore.patched"
        elif choice == 5:   # Ipad 1
            i = "iPad1,1_5.0_9A334_Restore.ipsw"
            r = "iPad1,1_5.0_9A334_Restore.dmg"
```

```
            k = "iPad1,1_5.0_9A334_Restore.patched"

        # run redsn0w in order to JailBreak
        wx.MessageBox(_("Please click CANCEL button to reboot after JailBreak finish."), _("Message!"))
        subprocess.call( "tools\\redsn0w.exe -i dmg\\%s -r dmg\\%s -k dmg\\%s" % (i, r, k) )
        #os.system("tools\\redsn0w.exe -i dmg\\%s -r dmg\\%s -k dmg\\%s" % (i, r, k))


    dlg.Destroy()

def MountUSB(self, evt):
    if not CheckPort(2222):
        threading.Thread(target=self._MountUSB).start()

def _MountUSB(self):
    # Run script
    sys.argv = []
    sys.argv.append( "" )
    sys.argv.append( "-t" )
    sys.argv.append( "22:2222" )
    sys.argv.append( "1999:1999" )
    #hookGlobal["sys.stdout"] = open( "a.txt", "w+" )
    runpy.run_module("tcprelay", run_name="__main__", alter_sys=True, init_globals=hookGlobal)


def BruteForce(self, evt):
    # Check tcp relay work
    if not CheckPort(2222):
        self.MountUSB(None)

    threading.Thread(target=self._BruteForce).start()

def _BruteForce(self):
    os.chdir("./projects")

    # Run script
    runpy.run_module("demo_bruteforce", run_name="__main__", alter_sys=True)

    filename = "%s.plist" % GetDi("dataVolumeUUID")
    if os.path.exists(filename):
        with open(filename, "a+") as fp:
            flag = False
            for i in fp.readlines():
                if flag:
                    pw = i
                    flag = False
                    break

                if i.find("<key>passcode</key>") != -1:
                    flag = True
```

```
            if pw:
                pw = BeautifulSoup(pw)
                pw = pw.findAll("string")[0].string
                if pw == None:
                    wx.MessageBox(_("There is no password") , _("Message!"))
                else:
                    wx.MessageBox(_("The password is %s" % pw), ("Message!"))
        else:
            wx.MessageBox(_("Can not find password file"), _("Error!"))

        # Back to work direcytory
        os.chdir("../../")

    def KeychainDeCrypt(self, evt):
        # Check tcp relay work
        # QQ strange bug QQ
        if not CheckPort(2222):
            self.MountUSB(None)

        threading.Thread(target=self._KeychainDeCrypt).start()

    def _KeychainDeCrypt(self):
        os.chdir("./projects/%s" % GetDi("udid"))

        # Check plist and keychain
        if os.path.exists( "%s.plist" % GetDi("dataVolumeUUID") ) and os.path.exists( "keychain-2.db" ):
            # Run script
            sys.argv = []
            sys.argv.append( "" )
            sys.argv.append( "-d")
            sys.argv.append( "keychain-2.db" )
            sys.argv.append( "%s.plist" % GetDi("dataVolumeUUID") )
            #hookGlobal["sys.stdout"] = open( ".\\%s\\password.txt" % GetDi("udid"), "w+" )
            runpy.run_module("keychain_tool", run_name="__main__", alter_sys=True,
init_globals=hookGlobal)

            dlg = wx.TextEntryDialog(
                    self, 'The password report has been saved. Please enter project name.',
                    _('Message'))

            dlg.SetValue("Project 1")

            if dlg.ShowModal() == wx.ID_OK:
                global udid
                global dataVolumeUUID
                udid = GetDi("udid")
                dataVolumeUUID = GetDi("dataVolumeUUID")
                _time = " ".join(time.asctime().split(" ")[2:])
                name = dlg.GetValue()
```

```
            c.execute( "INSERT INTO projects(udid, vid, name, time) values(?, ?, ?, ?)", (udid,
dataVolumeUUID, name, _time) )
            cx.commit()

            self.statusbar.SetStatusText("Project name is %s" % name)

         dlg.Destroy()
         wx.MessageBox(_("The password report has been saved."), _("Message!"))
      else:
         wx.MessageBox(_("Please brute force the passcode first."), _("Message!"))

      os.chdir("../../")

   def MirrorIOS(self, evt):
      os.chdir("./projects/%s" % GetDi("udid"))

      wx.MessageBox(_("Please switch to the terminal window, and enter password <alpine>."), _("Notice"))

      tmp = {"flag": True}
      threading.Thread(target=self._MirrorIOS, args=(tmp,)).start()
      dlg = PP.PyProgress(None, -1, "Dialog", "Progress")
      while tmp["flag"]:
         wx.MilliSleep(250)
         try:
            mb = os.path.getsize( "encrypted.img" ) / (1000*1000)
         except:
            mb = 0
         mb = "%d MB" % mb
         dlg.UpdatePulse(mb)
      dlg.Destroy()

      if os.path.getsize( "encrypted.img" ) == 0:
         wx.MessageBox(_("Failed, May the password is incorrect."), _("Message"))
      else:
         wx.MessageBox(_("Mirroring finish, the file name is encrypted.img."), _("Message"))

      os.chdir("../../")

   def _MirrorIOS(self, arg):
      class __(object):
         def __init__(self, *argv, **argd):
            object.__init__(self)

         def write(self, s):
            pass
      sys.stdout = __()
      cmd = "test -e /dev/rdisk0s2s1 && a='rdisk0s2s1';test -e /dev/rdisk0s1s2 && a='rdisk0s1s2';dd
if=/dev/$a bs=8192;"
      os.system(".\\..\\..\\ssh\\ssh.exe -p 2222 root@localhost \"%s\" | .\\..\\..\\ssh\\dd.exe
```

```
of=encrypted.img" % cmd )

        sys.stdout = HookStdOut()
        arg["flag"] = False

    def MirrorDecrypt(self, evt):
        os.chdir("./projects/%s" % GetDi("udid"))

        # Check tcp relay work
        if not CheckPort(2222):
            self.MountUSB(None)

        # Check image.img exists
        if not os.path.exists("encrypted.img") and not os.path.exists("decrypted.img"):
            global _
            wx.MessageBox(_("Please Do Mirror Action First."), _("Message"))
            os.chdir("../../")
            return None

        if not os.path.exists("backup_encrypted.img"):
            def __(arg):
                shutil.copy("encrypted.img", "backup_encrypted.img")
                arg["flag"] = False

            # pass by reference so using a list
            tmp = {"flag": True}
            threading.Thread(target=__, args=(tmp,)).start()
            dlg = PP.PyProgress(None, -1, _("Dialog"), _("Please Watting Backup the Image File."))

            while tmp["flag"]:
                wx.MilliSleep(250)
                progress = "%d %%" % ( (os.path.getsize("backup_encrypted.img")*100.0) /
os.path.getsize("encrypted.img") )
                dlg.UpdatePulse(progress)
            dlg.Destroy()

        os.chdir("../../")
        threading.Thread(target=self._MirrorDecrypt).start()

    def _MirrorDecrypt(self):
        os.chdir("./projects/%s" % GetDi("udid"))

        if os.path.exists("encrypted.img"):
            os.rename( "encrypted.img", "decrypted.img" )

        # Run script
        sys.argv = []
        sys.argv.append( "" )
        sys.argv.append( "decrypted.img" )
        runpy.run_module("emf_decrypter", run_name="__main__", alter_sys=True, init_globals=hookGlobal)
```

```
        os.chdir("../../")
        wx.MessageBox(_("Finish IOS mirror decryption, original image file name is encrypted.img, decrypted file name is decryptde.img."), _("Message"))

    def MirrorUnDelete(self, evt):
        os.chdir("./projects/%s" % GetDi("udid"))

        # Check image.img exists
        if not os.path.exists( "decrypted.img" ):
            global _
            wx.MessageBox(_("Please Do Mirror Action First."), _("Message"))
            os.chdir("../../")
            return None

        os.chdir("../../")
        threading.Thread(target=self._MirrorUnDelete).start()

    def _MirrorUnDelete(self):
        os.chdir("./projects/%s" % GetDi("udid"))

        # Run script
        sys.argv = []
        sys.argv.append( "" )
        sys.argv.append( "decrypted.img" )
        runpy.run_module("emf_undelete", run_name="__main__", alter_sys=True, init_globals=hookGlobal)

        os.chdir("../../")
        wx.MessageBox(_("Recover files are under directory %s/%s ." % (os.getcwd(), GetDi("udid"))), _("Message"))

    def MirrorUnDeleteDeep(self, evt):
        os.chdir("./projects/%s" % GetDi("udid"))

        # Check image.img exists
        if not os.path.exists( "decrypted.img" ):
            global _
            wx.MessageBox(_("Please Do Mirror Action First."), _("Message"))
            os.chdir("../../")
            return None

        os.chdir("../../")
        threading.Thread(target=self._MirrorUnDeleteDeep).start()

    def _MirrorUnDeleteDeep(self):
        os.chdir("./projects/%s" % GetDi("udid"))

        # Run script
        sys.argv = []
        sys.argv.append( "" )
```

```
        sys.argv.append( "decrypted.img" )
        runpy.run_module("Undelete_deep", run_name="__main__", alter_sys=True)

        os.chdir("../../")

    def _main(self):
        # Create Buttons & Texts
        self.CreateElement()

        # Binding events
        self.option0.Bind(wx.EVT_BUTTON, self.File)
        self.option1.Bind(wx.EVT_BUTTON, self.ChgLang)
        self.option2.Bind(wx.EVT_BUTTON, self.MountUSB)
        self.option3.Bind(wx.EVT_BUTTON, self.JB)
        self.option4.Bind(wx.EVT_BUTTON, self.BruteForce)
        self.option5.Bind(wx.EVT_BUTTON, self.KeychainDeCrypt)
        self.option6.Bind(wx.EVT_BUTTON, self.MirrorIOS)
        self.option7.Bind(wx.EVT_BUTTON, self.MirrorDecrypt)
        self.option8.Bind(wx.EVT_BUTTON, self.MirrorUnDelete)
        self.option9.Bind(wx.EVT_BUTTON, self.MirrorUnDeleteDeep)
        self.option10.Bind(wx.EVT_BUTTON, self.About)
        #self.Bind(wx.EVT_SHOW, self.OnShow)

        self.statusbar = self.CreateStatusBar()
        self.statusbar.SetStatusText("initial completed.")
        self.Centre()
        self.Show()

    def File(self, evt):
        dlg = File_Dialog(self, -1, _("Choice a action."), size=(350, 200))
        dlg.CenterOnScreen()
        val = dlg.ShowModal()
        if val == wx.ID_OK:
            choice = dlg.l.CurrentSelection

    def About(self, evt):
        wx.MessageBox(_("Version 1.1 Writtern by OSSLab "), _("About"))


def CheckPort(port=2222):
    ret = False

    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    try:
        s.bind( ("127.0.0.1", port) )
        s.close()
    except Exception, e:
        ret = True

    return ret
```

```
def GetDi(key):
    global udid
    global dataVolumeUUID

    if udid and dataVolumeUUID:
        di = {}
        di["udid"] = udid
        di["dataVolumeUUID"] = dataVolumeUUID
    else:
        client = RamdiskToolClient()
        di = client.getDeviceInfos()

    #udid or dataVolumeUUID
    return di[key]

# Hook print statement
sys.stdout = HookStdOut()

# Hook raw_input()
def msg():
    wx.MessageBox(_("Plese enter OK to continue."), _("Message"))
raw_input = msg

hookGlobal = {"raw_input": msg, "sys.stdout": HookStdOut()}

if __name__ == "__main__":
    app = MyApp(redirect=True)

    # do i18n
    mylocale = wx.Locale()
    mylocale.AddCatalogLookupPathPrefix("./lang/")
    mylocale.AddCatalog('cht')

    # init database
    if not os.path.exists("./projects"):
        os.mkdir("./projects")

    if not os.path.exists("./projects/db.db"):
        cx = sqlite3.connect("./projects/db.db")
        cx.text_factory = lambda x: unicode(x, "utf-8", "ignore")
        c = cx.cursor()
        c.execute("CREATE TABLE projects (id integer primary key, udid nvarchar(255), vid nvarchar(255),
name nvarchar(255), time nvarchar(255))")
        c.execute( "INSERT INTO projects(udid, vid, name, time) values(?, ?, ?, ?)",
('a3472a95b42e4a40647a1182ac248373a5b5c8ad', 'e0ae60eaf0d19ea9', "test", "i am time") )
        cx.commit()
    else:
        cx = sqlite3.connect("./projects/db.db")
        cx.text_factory = lambda x: unicode(x, "utf-8", "ignore")
```

```
    c = cx.cursor()

  global udid, dataVolumeUUID
  udid = None
  dataVolumeUUID = None
  frame = main(None, -1, _("IOS  Forensic Tools for Windows"), (400, 400))
  app.MainLoop()
```

?menu

IOS dirty way

?????? ??????????

http://www.poedit.net/download.php

beautifulsoup4-4.1.0 ?????

setup.py install

progressbar????

progressbar-2.2>setup.py install