Code execution manual

In terms of overall design and implementation, the project is divided into three parts as a whole, namely：

Part1 Artifacts Collector and Parser

Part2 Artifacts Audit and Analysis Platform

Part3 Malicious and Improve Behavior Analysis Model

The following is the specific process of how to run each part of the code, the main reference code has been marked in the report or code source file, and will not be repeated in this article.

Part1 Artifacts Collector and Parser:

Notice: This part should run in Python 3.9 and above.

In order to better package Part1 into a portable application, there is only one script file in Part1, named acpir Py, the script needs to install the library called by the program first, as shown in Figure 1.

For example, users can use the PIP command to install pymysql, and the command is :

pip install pymysql

Before running the script correctly, users should ensure that all library files have been correctly installed.

At the same time, in Visual Studio Code, after clicking the **Run Python File** button, as shown in Figure 2, users can also supplement the required library files according to the error information provided by the program. According to the error information, this example should install the win32evtlog library.

ACPIR.py > ...

```
import pymysql
import win32evtlog  # requires pywin32 pre-installed
import os
import sys
import ctypes
import winreg
import datetime
import string
import binascii
import struct
import subprocess
import collections
import uuid
import msvcrt
import time
import wmi
import socket
import json
from traceback import print_tb
import olefile
from sqlalchemy import false, true
from pathlib import Path
import win32security
from sqlalchemy import create_engine
import pandas as pd
import io
import hashlib
import ntpath
import tempfile
import re
```

Figure 1 Library files required for program operation

```
PS C:\Users\Zhu_Y> & c:/Users/Zhu_Y/AppData/Local/Programs/Python/Python37/python.exe c:/Users/Zhu_Y/ACPIR.py
Traceback (most recent call last):
  File "c:/Users/Zhu_Y/ACPIR.py", line 2, in <module>
    import win32evtlog  # requires pywin32 pre-installed
ModuleNotFoundError: No module named 'win32evtlog'
PS C:\Users\Zhu_Y>
```

Figure 2 Missing win32evtlog Library

The configuration information of the connection data of the program is as shown in Figure 3, which means that the user should configure the user name and password of the database, create the corresponding library file, and adjust the corresponding IP address information according to figure 3.

```
try:
    conn = pymysql.connect(host='127.0.0.1',port=3306,user='root',password='msp21074a.',db='artifacts_v2.9')
    #conn = pymysql.connect(host='192.168.43.128',user='root',password='07cc24',db='cysun31')
    curs=conn.cursor()
    print()
    print("[Remote Server Connected Successfuly!]")
```

Figure 3 Database Configuration

After the above steps are completed in the correct configuration, run the script in Visual Studio code, and the program box as shown in Figure 4 will pop up, which requests to obtain administrator permission.

Figure 4 UAC

After clicking the "yes" button, the program will show the results shown in Figure 5, which indicates that the program runs successfully.

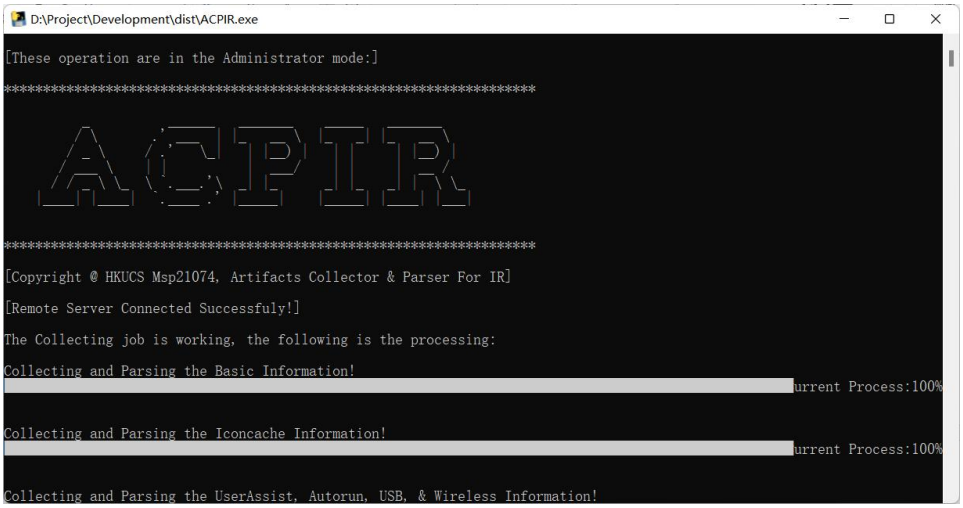


Figure 5 Program running

At the same time, the script can be packaged into an executable file by using pyinstaller program. The example command of program packaging is:

pyinstaller.exe -F D:\Project\Development\ACPIR_local.py

The packaged program does not need complex operations, and can be executed by double clicking, but the configuration information of its connection database is consistent with the information in its corresponding source code.

Notice! In part1, the script file is ACPIR.py, and the packaged program is ACPIR.exe. These ACPIR.py file is in the uploaded attachment.

Part2 Artifacts Audit and Analysis Platform

To run the web server, the first thing is to download the required third party module for the program. To do that, just type the command in command line shell. All the modules is listed in the requirements.txt file inside the project folder.

```shell
pip install -r requirements.txt
```

The second thing to be settle is the database configuration. In our project we use docker to build our MySQL container. The following command is to initialize the database.

```shell
docker run --name final_mysql -e MYSQL_ROOT_PASSWORD=msp21074a. -d -p 3306:3306 mysql
grant all privileges on *.* to root@'%' identified by 'msp21074a.' with grant option;
create database`artifacts_v2.9`;
FLUSH PRIVILEGES;
```

The third thing to do is the database migration, this is for connecting the database and the webserver.

```
python manage.py migrate
```

Last, to run the web server, just simply type the command in the shell.

```shell
python manage.py runserver
```

Part3 Malicious and Improper Behavior Analysis Model

There are three models in this section, each consisting of a training section and a loading section. Use the data collected by part1 to train the model locally, and then package the trained model and load function to the server side of part2 for use.

The import list shows below, as all the imports all python lib, they can be installed as

```
pip install xxxx
```

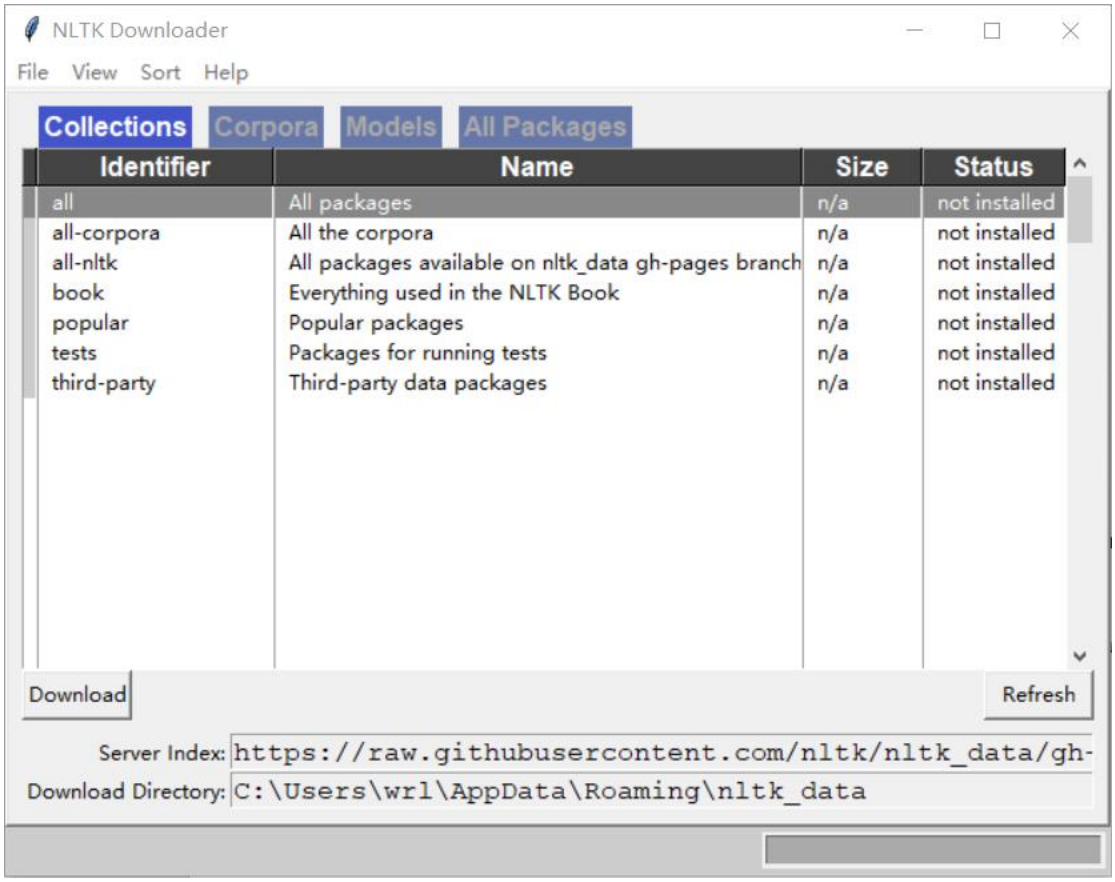
some lib's install name is not same as import name, if error occur, just search error and try.

```
#machine learning
import pandas as pd
import numpy as np
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from nltk.tokenize import RegexpTokenizer
from nltk.stem.snowball import SnowballStemmer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.pipeline import make_pipeline
from PIL import Image
from sklearn import preprocessing
#NLP
import nltk
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
from nltk.tokenize import word_tokenize
from nltk.tag import pos_tag
from stanfordcorenlp import StanfordCoreNLP
#workflow
import matplotlib.pyplot as plt
import time
import csv
import datetime
import os
import pickle
import torch
import joblib
#deep learning
import torch
import torch.nn as nn
import torch.nn.functional as F
import torchvision
from torchvision import transforms
```

```
from torch.utils.data import TensorDataset
from torchvision.utils import save_image
```

After install and import libs, there is still an operation need to be done for NLTK, the lib NLTK only contains code, but there is no pre-trained model, so run python code shows below and get a GUI download page. Click download to get model.

```
import nltk
nltk.download()
```



When testing models, put load_xxxx.py file with model and run. And xxxx is name for URL, event log and registry.

The invoke of load and pre functions are prepared in load.py, if wants to use functions in it, please model the call format in the load file.