



GIRLSCRIPT
INDIA SUMMIT' 2020

Investigating Digital Crimes using Python

Gajendra Deshpande,
KLS Gogte Institute of Technology, India

<https://gcdeshpande.github.io>

Outline of the Talk

- Introduction to digital crimes, digital forensics, the process of investigation, and collection of evidence.
- Investigation of acquisition media using the pyscreenshot module
- Investigation of emails
- Investigation of embedded metadata
- Conclusion and Questions

Cyber Crime Statistics

- The Internet Crime Report for 2019, released by USA's Internet Crime Complaint Centre (IC3) of the Federal Bureau of Investigation, has revealed that India stands third in the world among top 20 countries that are victims of internet crimes.
 - USA-4,67,361; UK-93,796; India-27,248 (2018 data)
- According to RSA report (2015), mobile transactions are rapidly growing and cyber criminals are migrating to less protected 'soft' channels.
- According to report by Norton (2015), an estimated 113 million Indians lost about Rs. 16,558 on an average to cybercrime
- According to an article published in Indian Express on 19th November 2016, *Over 55% Millennials in India Hit by Cybercrime.*

Digital Forensics

- Forensic science is the use of scientific methods or expertise to investigate crimes or examine evidence that might be presented in a court of law.
- Cyber Forensics is investigation of various crimes happening in the cyber space.
- Examples of cyber-attacks include phishing, ransomware, fake news, fake medicine, extortion, and insider frauds.
- According to DFRWS, Digital Forensics can be defined as “the use of scientifically derived and proven method toward the preservation, collection, validation, identification, analysis, interpretation, documentation, and presentation of digital evidence derived from digital sources for the purpose of facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations.”

Digital Forensics Investigation Process



- Identification

- Collection

- Validation

- Examination

- Preservation

- Presentation

Python Packages for Digital Forensics

- ▣ Pyscreenshot — Screenshot of a screen
- ▣ quopri - Encode and decode MIME quoted-printable data
- ▣ Mutagen - Python module to handle audio metadata.
- ▣ PyPDF2 - Pure-Python library built as a PDF toolkit.
- ▣ pefile - a multi-platform Python module to parse and work with Portable Executable (PE) files.

Pyscreenshot

- Pyscreenshot tries to allow to take screenshots without installing 3rd party libraries. It is cross-platform but mainly useful for Linux based distributions.
- Features: Cross-platform wrapper; Capturing the whole desktop or an area; pure Python library; Supported Python versions: 2.7, 3.6, 3.7, 3.8; Interactivity is not supported; Mouse pointer is not visible.
- Performance is not a target for this library, but you can benchmark the possible settings and choose the fastest one.

Installation

```
$ python3 -m pip install Pillow pyscreenshot
```

Source: <https://github.com/ponty/pyscreenshot>

Pyscreenshot – Full Screen

```
# pyscreenshot/examples/grabfullscreen.py
```

```
"Grab the whole screen"
```

```
import pyscreenshot as ImageGrab
```

```
# grab fullscreen
```

```
im = ImageGrab.grab()
```

```
# save image file
```

```
im.save("fullscreen.png")
```


Pyscreenshot – Part of Screen

```
# pyscreenshot/examples/grabbox.py

"Grab the part of the screen"
import pyscreenshot as ImageGrab

# part of the screen
im = ImageGrab.grab(bbox=(10, 10, 510, 510)) # X1,Y1,X2,Y2

# save image file
im.save("box.png")
```

Pyscreenshot - Performance

```
$ python3 -m pyscreenshot.check.speedtest
```

```
n=10
```

```
-----  
default          1    sec      ( 102 ms per call)  
pil  
mss              2.1  sec      ( 214 ms per call)  
scrot            1    sec      ( 101 ms per call)  
maim             1.5  sec      ( 147 ms per call)  
imagemagick      2.5  sec      ( 247 ms per call)  
pyqt5            4.4  sec      ( 442 ms per call)  
pyqt             3.5  sec      ( 352 ms per call)  
pyside2          5    sec      ( 495 ms per call)  
pyside           3.5  sec      ( 350 ms per call)  
wx               3.3  sec      ( 329 ms per call)  
pygtk3           2.3  sec      ( 225 ms per call)  
mac_screencapture  
mac_quartz  
gnome_dbus        1.7  sec      ( 166 ms per call)  
gnome-screenshot  2.3  sec      ( 231 ms per call)  
kwin_dbus
```

Pyscreenshot – Performance

```
$ python3 -m pyscreenshot.check.speedtest --childprocess 0
```

```
n=10
```

```
-----  
default                0.16 sec      ( 16 ms per call)  
pil  
mss                    0.17 sec      ( 17 ms per call)  
scrot                  1      sec      ( 104 ms per call)  
maim                   1.5      sec      ( 145 ms per call)  
imagemagick            2.5      sec      ( 246 ms per call)  
pyqt5                  1.1      sec      ( 110 ms per call)  
pyqt                   1      sec      ( 104 ms per call)  
pyside2                1.2      sec      ( 121 ms per call)  
pyside                 1      sec      ( 104 ms per call)  
wx                     0.33      sec      ( 32 ms per call)  
pygdk3                 0.2      sec      ( 19 ms per call)  
mac_screencapture  
mac_quartz  
gnome_dbus              1.5      sec      ( 152 ms per call)  
gnome-screenshot       2.3      sec      ( 230 ms per call)  
kwin_dbus
```

Pyscreenshot – Force backend

```
import pyscreenshot as ImageGrab  
im = ImageGrab.grab(backend="scrot")
```

```
# best performance  
import pyscreenshot as ImageGrab  
im = ImageGrab.grab(backend="mss", childprocess=False)
```

E-Mail Investigation

Re: Reminder: [Update Report] [Statement of Promise] Statement Updated Payment account Netflix: Wednesday, September 16 2020. Order Number 56871497



Netflix <rahma-cakupyufja-pokangenpapsfw@bihvgh.com>

om>

Wed 9/16/2020 11:58 PM

To: live@microsoft.com



Update current billing information

Unfortunately, we are unable to approve your payment for your next subscription cycle. Because financial institutions have rejected monthly charges, Netflix cannot receive payment.


To resolve the issue, Please update your payment information by pressing the button below.

[Try Again Payment](#)

For more information. Please visit the Help Center for more info or contact us

E-Mail Investigation

STRICTLY AND CONFIDENTIAL.

 This message was identified as junk. We'll delete it after 6 days. [It's not junk](#)



Mr.Sal Kavar <salkavar2@gmail.com>
Mon 9/14/2020 4:27 PM



Dear friend.

I assume you and your family are in good health. I am the Foreign operations Manager at one of the leading generation bank here in West Africa.

This being a wide world in which it can be difficult to make new acquaintances and because it is virtually impossible to know who is trustworthy and who can be believed, i have decided to repose confidence in you after much fasting and prayer. It is only because of this that I have decided to confide in you and to share with you this confidential business.

In my bank; there resides an overdue and unclaimed sum of \$15.5m, (Fifteen Million Five Hundred Thousand Dollars Only) When the account holder suddenly passed on, he left no beneficiary who would be entitled to the receipt of this fund. For this reason, I have found it expedient to transfer this fund to a trustworthy individual with capacity to act as foreign business partner. Thus i humbly request your assistance to claim this fund.

Upon the transfer of this fund in your account, you will take 45% as your share from the total fund, 10% will be shared to Charity Organizations in both country and 45% will be for me. Please if you are really sure you can handle this project, contact me immediately.

Yours Faithful,
Mr.Sal Kavar.

E-Mail Investigation

An investigator has the following goals while performing email forensics

- To identify the main criminal
- To collect necessary evidences
- To presenting the findings
- To build the case

Challenges in Email Forensics

- Fake Emails
- Spoofing
- Anonymous Re-emailing

Source:

https://www.tutorialspoint.com/python_digital_forensics/python_digital_forensics_investigation_using_emails.htm

E-Mail Investigation

Techniques used in Email Investigation

- Header Analysis
- Server Investigation
- Network Device Investigation

E-Mail Investigation

```
def main(input_file):  
    emlfile = message_from_file(input_file)  
    for key, value in emlfile._headers:  
        print("{}: {}".format(key, value))  
print("\nBody\n")  
  
if emlfile.is_multipart():  
    for part in emlfile.get_payload():  
        process_payload(part)  
else:  
    process_payload(emlfile[1])
```

access the headers,
body content,
attachments and
other payload
information

E-Mail Investigation

```
def process_payload(payload):
    print(payload.get_content_type() + "\n" + "=" * len(payload.get_content_type()))
    body = quopri.decodestring(payload.get_payload())

    if payload.get_charset():
        body = body.decode(payload.get_charset())
    else:
        try:
            body = body.decode()
        except UnicodeDecodeError:
            body = body.decode('cp1252')

    if payload.get_content_type() == "text/html":
        outfile = os.path.basename(args.EML_FILE.name) + ".html"
        open(outfile, 'w').write(body)
    elif payload.get_content_type().startswith('application'):
        outfile = open(payload.get_filename(), 'wb')
        body = base64.b64decode(payload.get_payload())
        outfile.write(body)
        outfile.close()
        print("Exported: {}\n".format(outfile.name))
    else:
        print(body)
```

Extract message body
content by using
get_payload() method

Check the content
MIME type so that it
can handle the storage
of the email properly

E-Mail Investigation

Extract Attachments

```
def extract_attachments(msg, out_dir):  
    attachment_attrs = ['DisplayName', 'FileName', 'PathName', 'Position', 'Size']  
    i = 1 # Attachments start at 1  
  
    while True:  
        try:  
            attachment = msg.Attachments(i)  
        except pywin32.com_error:  
            break
```

E-Mail Investigation

Extract Message body

```
def extract_msg_body(msg, out_dir):  
    html_data = msg.HTMLBody.encode('cp1252')  
    outfile = os.path.join(out_dir, os.path.basename(args.MSG_FILE))  
  
    open(outfile + ".body.html", 'wb').write(html_data)  
    print("Exported: {}".format(outfile + ".body.html"))  
    body_data = msg.Body.encode('cp1252')  
  
    open(outfile + ".body.txt", 'wb').write(body_data)  
    print("Exported: {}".format(outfile + ".body.txt"))
```

Metadata Forensics

- ❑ Mutagen is a Python module to handle audio metadata.
- ❑ It supports ASF, FLAC, MP4, Monkey's Audio, MP3, Musepack, Ogg Opus, Ogg FLAC, Ogg Speex, Ogg Theora, Ogg Vorbis, True Audio, WavPack, OptimFROG, and AIFF audio files. All versions of ID3v2 are supported, and all standard ID3v2.4 frames are parsed. It can read Xing headers to accurately calculate the bitrate and length of MP3s. ID3 and APEv2 tags can be edited regardless of audio format. It can also manipulate Ogg streams on an individual packet/page level.
- ❑ Mutagen works with Python 3.6+ (CPython and PyPy) on Linux, Windows and macOS, and has no dependencies outside the Python standard library.

Source: <https://mutagen.readthedocs.io/en/latest/>

Metadata Forensics

Installation: `python3 -m pip install mutagen`

The File functions takes any audio file, guesses its type and returns a FileType instance or None

```
>>> import mutagen
>>> mutagen.File("11. The Way It Is.ogg")
{'album': [u'Always Outnumbered, Never Outgunned'],
 'title': [u'The Way It Is'], 'artist': [u'The Prodigy'],
 'tracktotal': [u'12'], 'albumartist': [u'The Prodigy'], 'date': [u'2004'],
 'tracknumber': [u'11'],
>>> _.info.pprint()
u'Ogg Vorbis, 346.43 seconds, 499821 bps'
>>>
```

Metadata Forensics

Installation: `python3 -m pip install mutagen`

The File functions takes any audio file, guesses its type and returns a FileType instance or None

```
from mutagen.flac import FLAC

audio = FLAC("example.flac")
audio["title"] = u"An example"
audio.pprint()
audio.save()
```

The following example gets the length and bitrate of an MP3 file.

```
from mutagen.mp3 import MP3

audio = MP3("example.mp3")
print(audio.info.length)
print(audio.info.bitrate)
```

Metadata Forensics

PyPDF2 - Pure-Python library built as a PDF toolkit. It is capable of:

- extracting document information (title, author, ...)
- splitting documents page by page
- merging documents page by page
- cropping pages
- merging multiple pages into a single page
- encrypting and decrypting PDF files
- It is a useful tool for websites that manage or manipulate PDFs.

Source: <https://pypi.org/project/PyPDF2/>

Metadata Forensics

- pefile is a multi-platform Python module to parse and work with Portable Executable (PE) files. Most of the information contained in the PE file headers is accessible, as well as all the sections' details and data.
- The structures defined in the Windows header files will be accessible as attributes in the PE instance. The naming of fields/attributes will try to adhere to the naming scheme in those headers. Only shortcuts added for convenience will depart from that convention.
- pefile requires some basic understanding of the layout of a PE file — with it, it's possible to explore nearly every single feature of the PE file format.

Source: <https://github.com/erocarrera/pefile>

Metadata Forensics

Some of the tasks that pefile makes possible are:

- Inspecting headers
- Analyzing of sections' data
- Retrieving embedded data
- Reading strings from the resources
- Warnings for suspicious and malformed values
- Basic butchering of PEs, like writing to some fields and other parts of the PE
- This functionality won't rearrange PE file structures to make room for new fields, so use it with care.
- Overwriting fields should mostly be safe.
- Packer detection with PEiD's signatures
- PEiD signature generation

Conclusion

- It is very important to follow the standard procedure laid by law enforcement agencies during investigation process.
- There are many open source as well as commercial tools for digital forensics. Learning to develop your own tool is advantageous.
- Many tools in written in Python are pure Python implementations

Thank You!