NTFS File System
Folder Structure Analysis

Track	Digital Forensic
Category	Tech_01
Nick Name	L3ad0xFF

목 차

1. 개발 목적 및 목표

- 활성상태의 NTFS File System에서 폴더명을 입력 받아 그 하위 구조를 탐색한다.
- 탐색한 하위 구조의 Meta Data를 출력한다.
- 각각 순번, 파일명, 크기, 경로, Modified Time, Access Time, Create Time, Entry Changed Time의 형태의 표를 같는 HTML 파일을 저장한다.
- 파일명은 상대경로로 하이퍼링크를 통해 열람이 가능하도록 한다.
- 파일명, 경로, 시간 등의 정/역순 정렬기능을 제공한다.
- Index Buffer(#i30) 정보 및 MFT Entry 정보를 포함하여 출력한다.

2. 개발 진행 시 사용 특이 Module 설명

2.1 Pytsk3

- SleuthKit에서 제공하는 파일 시스템 분석 도구이며, python에서 동작하도록 도움을 준다.
- SleuthKit에서 제공하는 libtsk등의 라이브러리를 import하여 파일 시스템 분석 시, 각 오프셋과 영역에 접근할 수 있는 함수를 제공한다.

2.2 wmi (Windows Management Instrumentation)

- WMI는 계측 구성요소 정보를 제공하는 운영체제 인터페이스를 제공하는 Windows 드라이버 모델의 확장 집합
- Python에서 해당 모듈을 사용하여 사용자 system에 접근이 가능하도록 한다.

2.3 time

- UnixTimeStamp 형식을 일반적인 날짜 및 시간 형식으로 변환 표기가 가능하다.
- Localtime 내장 함수를 사용하여 시스템에 설정된 UTC 시간으로 변환이 가능하다.
- Strftime 내장 함수로 출력할 날짜 및 시간의 형식을 사용자가 임의로 설정할 수 있다.

2.4 dateTime

- localtime 내장 함수를 이용하여 time 모듈과 마찬가지로 시스템에 설정된 시간대역으로 시간 값을 설정한다.
- Time 모듈과 차이점은 time 모듈은 unixtimestamp 변환이 가능하다는 점이다.

3. 개발 Script – PyQt5 GUI 프로그램

3.1 Function maplogicphysic

- 현재 활성중인 system에서 기본 사용 및 외부연결 저장 Disk를 탐색한다.
- 탐색의 목적은 system에 등록 중인 물리 디스크 번호와 논리 디스크 이름의 상관관계를 파악하기 위한 목적이다.
- Table 형태로 output을 제공하며, 사용자에게 탐색하고자 하는 Volume명에 관련된 물리 디스크번호를 정확하게 선택하여 오류없이 입력하도록 하고자 한다.

```
def maplogicphysic(self) :
    global phylogset
    phylogset = {}
    w = wmi.WMI()
    drivenum = 0
    for drive in w.Win32_LogicalDisk() :
        # print ("\t"+ drive.Caption + " = PhysicalDrive" + str(drivenum))
        phylogset['PhysicalDrive'+str(drivenum)] = drive.Caption
        drivenum += 1

    line_num = len(phylogset)
    self.tableWidget.setRowCount(line_num)
    self.tableWidget.setColumnCount(2)

    self.tableWidget.setHorizontalHeaderLabels(["Drive_Name", "PhysicalDrive"])

    value_list = list()

    for value in phylogset.values() :
        value_list.append(value)

    for row in range(0, len(phylogset)) :
        nd_column = value_list[row]
        item = QTableWidgetItem(nd_column)
        self.tableWidget.setItem(row, 0, item)

    for row in range(0, len(phylogset)) :
        st_column = "PhysicalDrive" + str(row)
        item2 = QTableWidgetItem(st_column)
        self.tableWidget.setItem(row, 1, item2)


    self.tableWidget.resizeRowsToContents()
    self.tableWidget.resizeColumnsToContents()
```

  - 윈도우에서 관리정보에 접근 하기위한 WMI(Windows Management Instrumentation) 구조체 module을 import하여 현재 system의 활성화 된 disk에 접근하여 Volume명을 출력한다.

3.2 Function start

- 사용자에게 탐색 대상의 PhysicalDrive정보를 입력 받고, 해당 물리 디스크에 접근하여 파티션 정보를 얻는다.
- 먼저 모든 파일을 탐색하고 그들의 크기를 측정한다.

  1. pytsk3.Img_Info를 이용하여, 사용자가 입력 대상의 디스크의 정보를 확인한다.
  2. pytsk3.Volume_Info를 통해, Partition 정보에 접근하고, 해당 디스크에 구성된 Partition을 탐색한다.
  3. NTFS File System으로 구성된 Partition에 한하여 정보를 추출하고, 각 파티션의 시작위치를 저장하여, 디스크 내 정보 탐색에서 NTFS File System 탐색 시, 모두 탐색 하도록 한다.

3.2.1 Partition Information Search & Size Calculate

- Start 함수에서 pytks3.Volume_info로 구성한 MBR 영역에 접근하여, Partition 정보를 읽어 온다.
- Partition size는 default 단위가 Bytes이기 때문에, 직접 생성한 partition 함수에서 계산하여 단위를 사용자가 인식하기 쉽게 계산한다.
- Partition 정보는 GUI환경에서 참조할 수 있는 정도의 비중으로 text 창에 출력한다.

```
def partitionsize (self, num) :
    size_list = ['B', 'MB', 'KB', 'GB']
    size = num * 512.

    for div_count in range (0, 4) :
        if int(size) > 0 :
            size = size / 1024
        else :
            break

    return '%.2f %s' % (size *1024, size_list[div_count])

def searchPartition (self, PartitionTables) :
    ntfs_list = list()
    for partition in PartitionTables:
        if partition.desc == b'NTFS / exFAT (0x07)' :
            par_type = "[+] Type : " +  partition.desc.decode('utf-8')
            par_number = "\n\t[-] Number : " + str(partition.start)
            par_start_sec = "\n\t[-] Start Sector : " + str(partition.start)
            par_sec_cnt = "\n\t[-] Sector Count : " + str(partition.len)
            par_size = "\n\t[-] Size : " + self.partitionsize(partition.len)
            ntfs_list.append(partition.start)
    self.textBrowser.setText(par_type + par_number + par_start_sec + par_sec_cnt
    return ntfs_list
```

3.3 Function find_directory(self, vbr_offset)

3.3.1 탐색 대상의 하위 폴더 및 파일 정보

- GUI환경에서 사용자가 탐색할 경로를 입력한다.
- 입력된 경로에 접근하여, 해당 경로의 하위 파일 및 폴더를 모두 탐색한다.
- 탐색된 파일 및 폴더를 구별하여 표기하고, 모든 파일 및 폴더에 관한 과제에서 요구한 Meta Data(파일 및 폴더 명, 경로, mtime, atime, ctime, etime. 파일 크기)를 읽어 온다.
  파일 및 폴더 명 : directory.info.name.naem.decode('utf-8') – 한글의 경우 utf-8로 변환
  파일 및 폴더 변경 시간 : directory.info.mtime
  파일 및 폴더 접근 시간 : directory.info.atime
  파일 및 폴더 생성 시간 : directory.info.crtime
  파일 및 폴더 엔트리 시간 : directory.info.ctime
- 경로 및 분류 타입은 직접 설정한다. 파일과 폴더의 분류는 pytsk3.TSK_FS_META_TYPE_DIR 를 이용하며, 폴더일 경우 DIR, 파일일 경우 REG 값이 반환되는 값이다.

- 경로 : 사용자가 입력한 물리디스크 값에 매칭되는 논리 디스크 이름 + 사용자가 탐색을 위해 입력한 경로 + 파일 및 폴더 명
- pytsk3.TSK_FS_META_TYPE_DIR = DIR : Directory 문자열 삽입. / REG : File 문자열 삽입

```
def find_directory(self, vbr_offset) :
    global in_path_list, set_path, fs
    fs = pytsk3.FS_Info(Volume_img, offset = vbr_offset)
    set_path = self.lineEdit_2.text()
    in_path_list = list()

    directorys = fs.open_dir(path = set_path)
    # count = 0
    for directory in directorys :
        if ((directory.info.name.name).decode('utf-8') == '.') or ((directory.info.name.name).decode('utf-8') == '..') :
            continue

        try :
            if directory.info.meta.type == pytsk3.TSK_FS_META_TYPE_DIR :
                dir_list = list()
                dir_name = (directory.info.name.name).decode('utf-8')
                dir_type = "Directory"
                #dir_EA = directory.info.meta.addr
                dir_path = root_path + set_path + ((directory.info.name.name).decode('utf-8'))
                dir_mtime = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(directory.info.meta.mtime))
                dir_atime = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(directory.info.meta.atime))
                dir_ctime = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(directory.info.meta.crtime))
                dir_ectime = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(directory.info.meta.ctime))
                dir_size = str(directory.info.meta.size) + " Bytes"
                #count += 1
                dir_list = [dir_type, dir_name, dir_size, dir_path, dir_mtime, dir_atime, dir_ctime, dir_ectime]
                in_path_list.append(dir_list)
            else:
                file_list = list()
                file_name = (directory.info.name.name).decode('utf-8')
                file_type = "File"
                #file_EA = directory.info.meta.addr
                file_path = root_path + set_path + ((directory.info.name.name).decode('utf-8'))
                file_mtime = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(directory.info.meta.mtime))
                file_atime = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(directory.info.meta.atime))
                file_ctime = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(directory.info.meta.crtime))
                file_ectime = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(directory.info.meta.ctime))
                file_size = str(directory.info.meta.size) + " Bytes"
                #count += 1
                file_list = [file_type, file_name, file_size, file_path, file_mtime, file_atime, file_ctime, file_ectime]
                in_path_list.append(file_list)
        except :
            pass
```

- 첫번째 Red Box 부분은 GUI에서 사용자가 원하는 탐색 경로 입력을 받는 부분
- 두번째 Red Box : pytsk3.TST_FS_META_TYPE_DIR = DIR
- 세번째 Red Box : pytsk3.TST_FS_META_TYPE_DIR = REG

3.3.2 Table에 탐색 한 정보 출력 설정

- 앞서 탐색한 폴더 및 파일의 Meta Data를 GUI 환경에 출력하기 위한 부분이다.
- 탐색한 폴더 및 파일에 관한 Meta Data를 각각 하나의 리스트로 구성하고, 그 리스트들을 모두 다시 하나의 리스트로 구성하였다. 전체 리스트의 길이가 하위 폴더 및 파일의 개수가 된다.
- Table 출력의 row 개수는 탐색한 경로에 존재하는 모든 폴더 및 파일의 개수로 설정하며, 전체 리스트의 길이로 설정한다. (setRowCount)
- Table 출력의 Column 개수는 추출한 정보 및 정보를 조합하여 사용자가 양식에 맞게 조합한 요구하는 항목들의 개수로서 총 8개로 구성한다. (setColumnCount)
- Column이 의미하는 값을 첫 행에 설명한다. (setHorizontalHeaderLabels를 이용하여 사용자 입력 값이 각 column의 이름으로 구성된다.)
- 출력 시 자동 정렬이 아닌 사용자의 선택에 선택된 column명을 기준으로 자동 정렬을 수행하기 위해 default값으로는 우선 정렬 기능을 제거한다. (setSortingEnabled(False))

- column명은 horizontalHeader의 모듈 내장함수로 접근할 수 있으며, 해당 위치가 사용자에 의해 선택되었을 경우 signal을 발생시켜서, sorting이 가능한 함수로 연결한다. (column명 선택 시, 정렬 기능 일시적 활성화, table_column_sort 함수로 이동)
- Table을 구성하는 요소 중 행과 열에 해당하는 cell이 존재하며, 해당 cell을 선택 시에도 signal이 발생하여, 선택한 cell의 type이 폴더일 경우 선택된 cell의 폴더를 탐색한다. -> cell_was_sort 함수 호출
- setColumnWidth를 이용하여 Table의 Column 넓이를 사용자 설정으로 조정한다. **조정의 이유는 시간 정보의 일부가 default 크기의 경우 가려져서 설정하였다.**

```
line_num = len(in_path_list)
self.tableWidget_2.setRowCount(line_num)
self.tableWidget_2.setColumnCount(8)
self.tableWidget_2.setHorizontalHeaderLabels(["Type", "Name", "Size", "Path",
    "Modified time", "Access time", "Create time", "EntryChange time"])

for p_row in range(0, len(in_path_list)) :
    for p_col in range(0, 8) :
        item = QTableWidgetItem(in_path_list[p_row][p_col])
        self.tableWidget_2.setItem(p_row, p_col, item)

self.tableWidget_2.setSortingEnabled(False)
t_column = self.tableWidget_2.horizontalHeader()
t_column.sectionClicked.connect(self.table_column_sort)

self.tableWidget_2.cellClicked.connect(self.cell_was_clicked)

# self.tableWidget_2.resizeRowsToContents()
self.tableWidget_2.setColumnWidth(0, 100)
self.tableWidget_2.setColumnWidth(1, 170)
#self.tableWidget_2.setColumnWidth(2, 150)
self.tableWidget_2.setColumnWidth(3, 250)
self.tableWidget_2.setColumnWidth(4, 150)
self.tableWidget_2.setColumnWidth(5, 150)
self.tableWidget_2.setColumnWidth(6, 150)
self.tableWidget_2.setColumnWidth(7, 150)
```

3.4 Function table_column_sort(self, p_row)

- 출력된 Table의 폴더 및 파일의 정보들을 Column 명을 선택하여 정렬하는 기능을 제공한다.
- Click시 현재 함수로 이동하는 signal이 발생하면, 선택한 Column의 정보를 이용하여, 오름차순 및 내림차순이 가능하도록 setSortingEnabled(True)로 설정하면서 정렬하고, 무분별한 정렬 방지를 위해, 다시 False 옵션으로 비활성화를 한다.

```
def table_column_sort(self, p_row) :
    self.tableWidget_2.setSortingEnabled(True)
    self.tableWidget_2.setSortingEnabled(False)
```

3.5 Function cell_was_cliced(self, row, column)

- 최초 사용자가 입력한 경로를 탐색하여 출력된 하위 폴더 및 파일의 결과가 출력된 table에서 cell을 선택하는 signal이 발생하면 실행된다. (최초 table 출력 시 동작하는 함수 내에 cell 선택 할 경우 동작하는 signal이 수행된다)

- 선택 된 cell의 행과 열을 탐지하고, 선택된 행에서 path 경로가 저장된 4번 column의 cell item을 읽어서 next_path라는 변수명에 저장하여 다음 탐색 경로로 지정한다.
- 마지막으로 clickeddir함수를 호출하고 next_path를 인자로 한다.

```
def cell_was_clicked(self, row, column):
    global next_path
    next_path = ""
    for com_row in range (0, len(in_path_list)) :
        for com_col in range (0, 8) :
            if (com_row == row) and (com_col == column) :
                if self.tableWidget_2.item(row, 0).text() == "Directory" :
                    next_path_str = self.tableWidget_2.item(row, 3).text()
                    next_path = next_path_str[2:] +'/'
                    print ("sel : " + next_path_str)
                    print (next_path)
                else :
                    print ("Not Directory")

    self.clickeddir()
    return "clicked"
```

3.6 Function clickeddir

- 함수의 기능은 2.3항의 find_dirctory와 동일하게 수행하여 선택된 경로를 탐색하여 하위 폴더 및 파일을 출력한다. 그러나 find_dirctory는 GUI에서 사용자의 입력을 탐색 경로로 한다면, clickeddir의 함수의 경우 앞서 cell을 선택하여 얻은 다음 탐색 경로(next_path)가 새로운 탐색 경로 설정하여 table에 출력한다.

```
def clickeddir(self) :
    directorys = fs.open_dir(path = next_path)
    # count = 0
    in_path_list = list()
    for directory in directorys :
        if ((directory.info.name.name).decode('utf-8') == '.') or ((directory.info.name.name).decode('utf-8') == '..') :
            continue

        try :
            if directory.info.meta.type == pytsk3.TSK_FS_META_TYPE_DIR :
                dir_list = list()
                dir_name = (directory.info.name.name).decode('utf-8')
                dir_type = "Directory"
                #dir_EA = directory.info.meta.addr
                dir_path = root_path + next_path + ((directory.info.name.name).decode('utf-8'))
                dir_mtime = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(directory.info.meta.mtime))
                dir_atime = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(directory.info.meta.atime))
                dir_ctime = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(directory.info.meta.crtime))
                dir_ectime = time.strftime("%Y-%m-%d %H:%M:%S", time.localtime(directory.info.meta.ctime))
                dir_size = str(directory.info.meta.size) + " Bytes"
                #count += 1
                dir_list = [dir_type, dir_name, dir_size, dir_path, dir_mtime, dir_atime, dir_ctime, dir_ectime]
                in_path_list.append(dir_list)
```

3.7 Function makehtml

- 획득한 하위 폴더 및 파일들의 정보들을 html로 저장하기 위한 함수이며, GUI에서 'Save to HTML' button을 선택할 경우 실행 된다.
- 추출할 때의 현재시간으로 파일명을 생성하기 위해 time module을 이용하여, 시간 값을 획득한다. Html의 저장경로는 GUI를 실행하기 위해 python code가 실행되는 위치로 설정한다.

- 저장할 폴더 및 파일의 meta data가 저장되는 위치 전까지 html의 구조를 저장하기 위해 open한 html에 기록한다. Meta data들은 html 내의 table 요소들로 저장 한다.
- 저장이 완료되면 MessageBox를 출력한다.

```
def makehtml(self) :
    now_date = time.strftime("%Y-%m-%d", localtime())
    now_time = time.strftime("%H_%M_%S", localtime())
    set_loctime = now_date + " " + now_time
    time_info = "Local time_" + set_loctime
    filename = time_info + ".html"
    print (filename)
    htmlfile = open(filename, 'w')
    head_string = '<!DOCTYPE html><html><head><meta charset="UTF-8"></meta><title>NTFS Printer</title><link href="http://fonts.googleapis.com/css?family=Open+Sans:40
    body_string = '<body><div class="window_header"><h2>NTFS Analysis</h2><p>Best Of the Best 6th Digital Forensic</p></div><div class="clickclick">'
    body_t_string = '<table style="width:100%"><tr><th>Number</th><th>Type</th><th>Name</th><th>Size</th><th>Path</th><th>Modified Time</th><th>Access Time</th><th>C
    init_string = head_string + body_string + body_t_string
    htmlfile.write(init_string)
    body_tb_string = " "
    for h_row in range(0, Len(in_path_list)) :
        body_tb_string += "\n<tr>\n<td>" + str(h_row+1) + "</td>\n"
        for h_col in range(0,8) :
            if h_col == 1 or h_col == 3 :
                body_tb_string += "\t<td>" + in_path_list[h_row][h_col] + "</td>\n"
                #print (in_path_list[h_row][h_col])
            else :
                body_tb_string += "\t<td>" + in_path_list[h_row][h_col] + "</td>\n"
        body_tb_string += "</tr>\n"
    body_tb_string += "</table>"

    footer_string = '<div class="window_footer" style="padding-top: 10px;height: 53.2px;"><a><span>@Author : L3ad0xFF</span> (Moonwon LEE)<br></a><p style="margin-to
    htmlfile.write(body_tb_string)
    htmlfile.write(footer_string)
    htmlfile.close()

    self.msgboxprint()
```

3.8 Function msgboxprint

- Makehtml 함수에서 생성할 html에 설정한 data를 기록 후 저장하고 나면, 저장된 경로를 알려주는 Messagebox를 출력한다.
- 출력되는 Message box에는 상세 정보를 열람할 수 있는 option을 추가하였으며, 상세정보 확인 button을 선택하면 저장한 로컬시간이 표출된다.

```
def msgboxprint(self) :
    save_path = os.getcwd()
    now_date = time.strftime("%Y-%m-%d", localtime())
    now_time = time.strftime("%H:%M:%S", localtime())
    set_loctime = now_date + " " + now_time
    time_info = "Local time = " + set_loctime
    infoBox = QMessageBox()
    infoBox.setIcon(QMessageBox.Information)
    infoBox.setText("Extract HTML File Success!")
    infoBox.setInformativeText("Save Path = " + save_path)
    infoBox.setWindowTitle("Window Title")
    infoBox.setDetailedText(time_info)
    infoBox.setStandardButtons(QMessageBox.Ok | QMessageBox.Cancel)
    infoBox.setEscapeButton(QMessageBox.Close)
    infoBox.exec_()
```

3.9 Imported Module state

```
from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtWidgets import *
from PyQt5.QtCore import *
import pytsk3
import sys
import os
from struct import *
import datetime
import time
from time import gmtime, localtime, strftime
import wmi
```

3.10 기본 Layout 설정 및 signal 설정

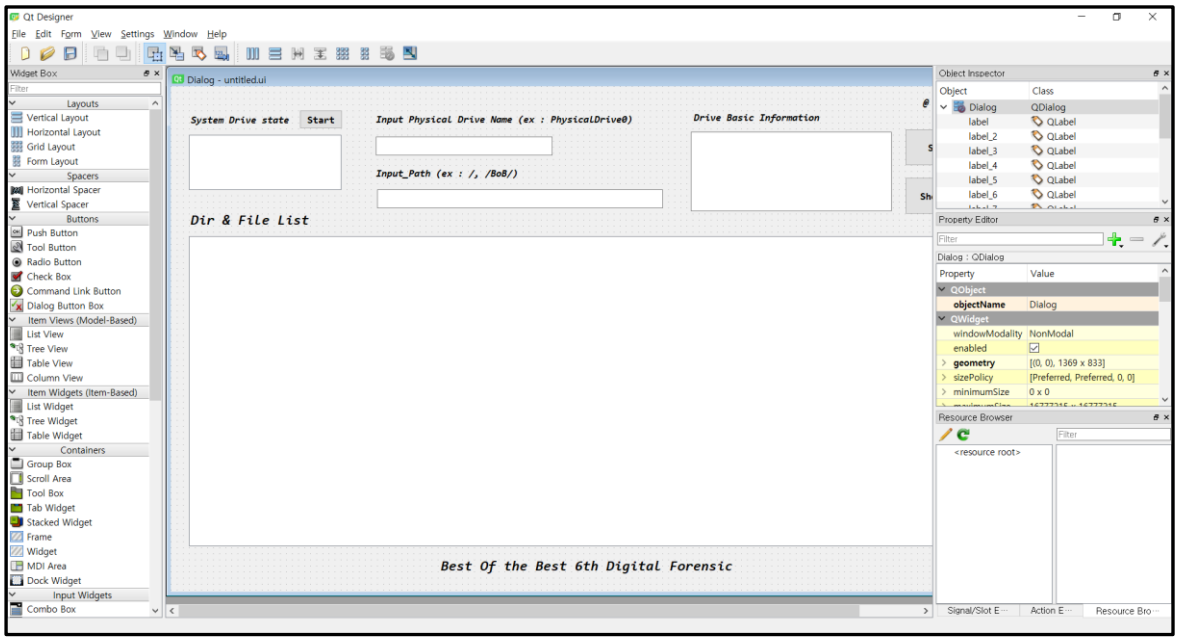
- 첫 Red Box 부분에 Button 선택 시 수행할 함수를 지정하여 signal을 지정한다.
- 두번째 Red Box 부분은 기본 Layout에 대한 사용자 문자열 설정이다.

```
    self.retranslateUi(Dialog)
    QtCore.QMetaObject.connectSlotsByName(Dialog)
    self.pushbutton_3.clicked.connect(self.maplogicphysic)
    self.pushbutton.clicked.connect(self.start)
    self.pushbutton_2.clicked.connect(self.makehtml)

def retranslateUi(self, Dialog):
    _translate = QtCore.QCoreApplication.translate
    Dialog.setWindowTitle(_translate("Dialog", "NTFS_Printer"))
    self.label.setText(_translate("Dialog", "System Drive state"))
    self.label_3.setText(_translate("Dialog", "Input_Path (ex : /, /BoB/)"))
    self.label_2.setText(_translate("Dialog", "Input Physical Drive Name (ex : PhysicalDrive0)"))
    self.pushbutton.setText(_translate("Dialog", "Search"))
    self.label_4.setText(_translate("Dialog", "Dir & File List"))
    self.pushbutton_2.setText(_translate("Dialog", "Save to html"))
    self.label_5.setText(_translate("Dialog", "Best Of the Best 6th Digital Forensic"))
    self.label_6.setText(_translate("Dialog", "@ L3ad0xFF"))
    self.pushbutton_3.setText(_translate("Dialog", "Start"))
    self.label_7.setText(_translate("Dialog", "Drive Basic Information"))
```

4. GUI 실행

4.1 QT Design을 이용하여 ui 설정

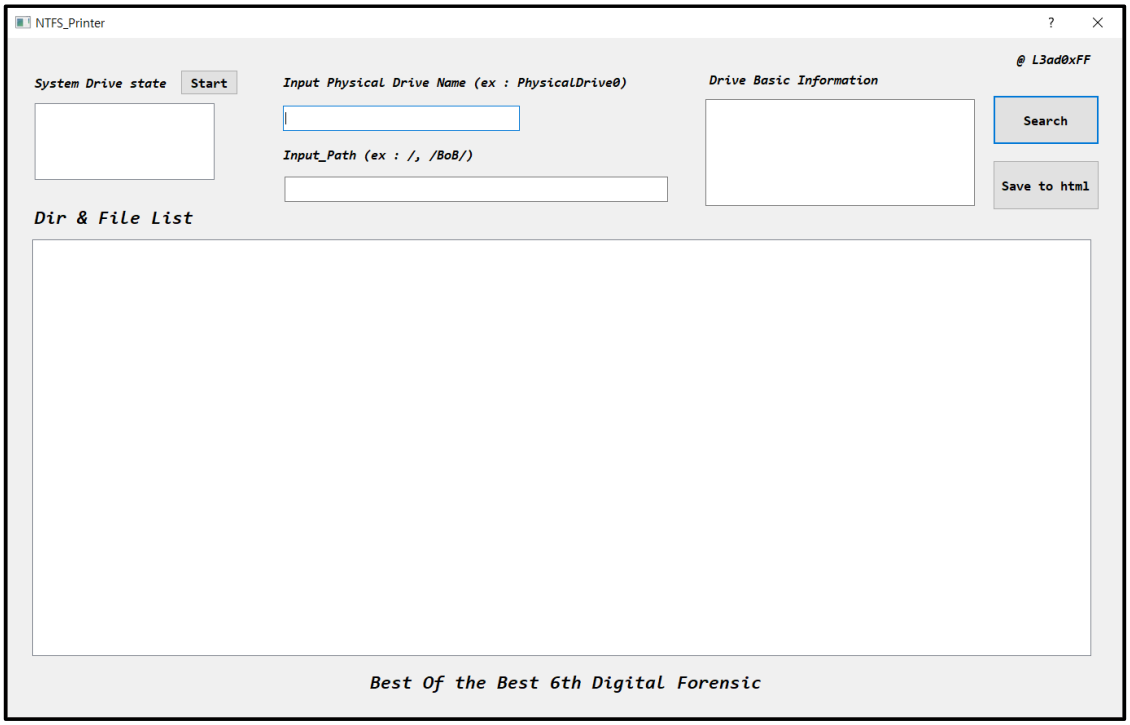


➢ 기본 Layout을 구성 후 ui 확장자로 저장한다.

4.2 ui -> py 변환 후 실행

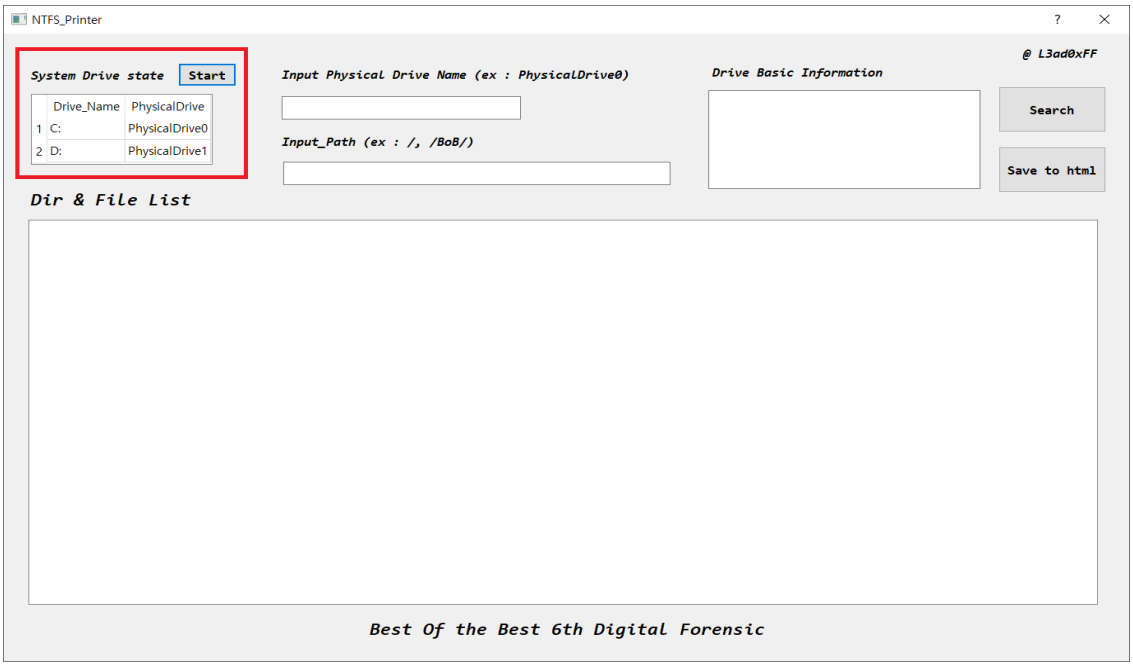
➢ PyQT5에서 ui 파일을 py로 변환하는 코드가 있는 경로에서 python -m PyQt5.uic.pyuic -o <ui filename.ui> -x <pythonfilename.py> 명령으로 변환한다.

➢ 변환된 python file명을 실행하면 GUI가 설정한 Layout으로 실행된다.



4.3 동작

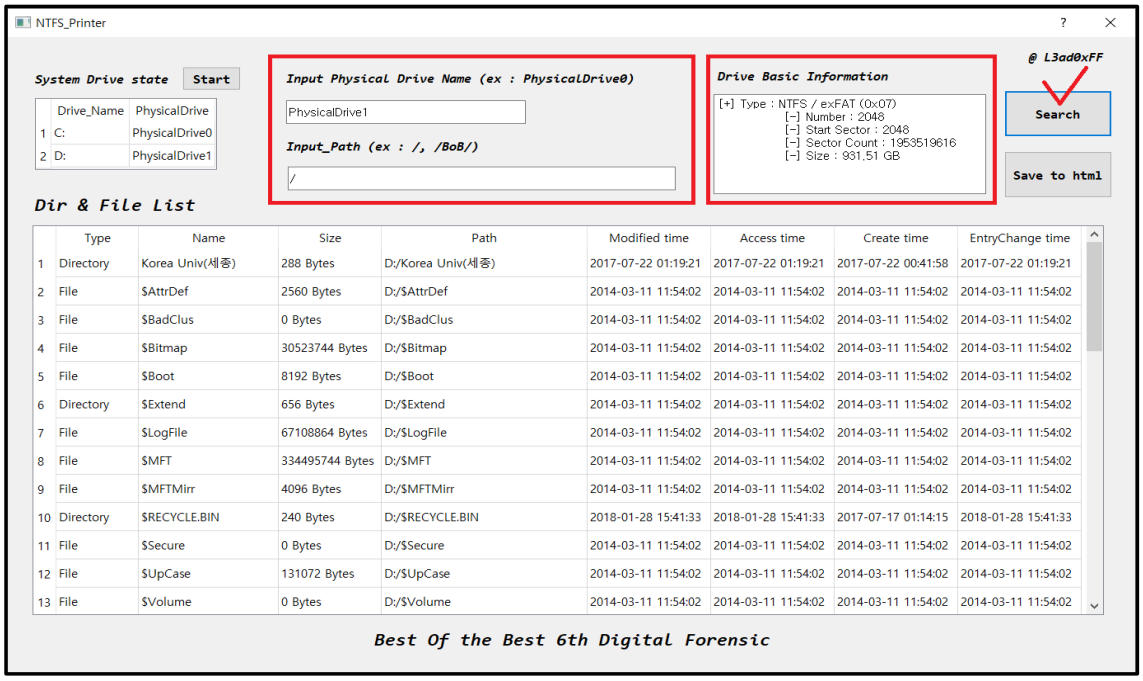
➢ Start Button을 눌러서 현재 system에 활성화 된 물리 디스크와 논리 디스크 명을 확인



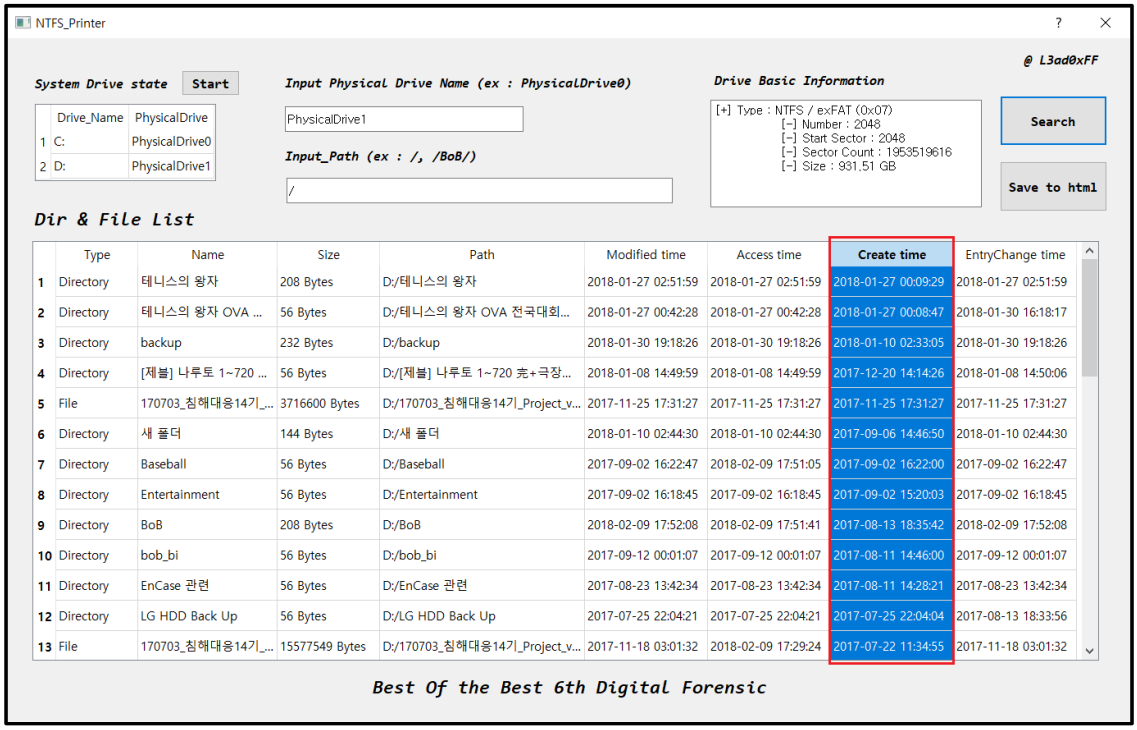
➢ Input Physical Drive 영역에 조회할 논리 디스크 명에 해당되는 물리 디스크를 작성한다. (대, 소문자 구별을 하기 때문에 정확하게 구분하여 입력해야 한다.)

➢ Input_Path 영역에 탐색할 폴더 명을 입력한다. 이후 Search를 눌러 지정한 경로를 탐색한다. (root 경로는 = '/')

➢ Driver의 기본 Partition 정보와 Dir & File List에 탐색 위치 경로의 하위 폴더 및 파일의 meta data가 출력된다.



➢ 정렬 기준의 Column명을 선택하면 해당 Column의 정보로 순차 정렬을 수행한다.

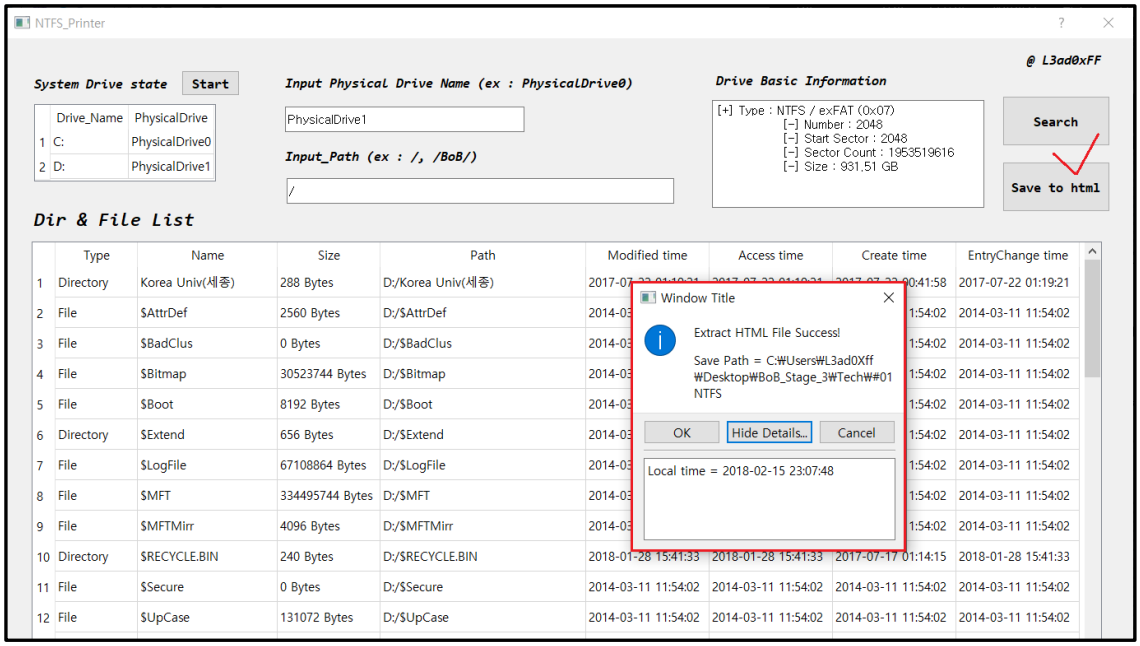


	Drive_Name	PhysicalDrive
1	C:	PhysicalDrive0
2	D:	PhysicalDrive1

	Type	Name	Size	Path	Modified time	Access time	Create time	EntryChange time
1	Directory	테니스의 왕자	208 Bytes	D:/테니스의 왕자	2018-01-27 02:51:59	2018-01-27 02:51:59	2018-01-27 00:09:29	2018-01-27 02:51:59
2	Directory	테니스의 왕자 OVA ...	56 Bytes	D:/테니스의 왕자 OVA 전국대회...	2018-01-27 00:42:28	2018-01-27 00:42:28	2018-01-27 00:08:47	2018-01-30 16:18:17
3	Directory	backup	232 Bytes	D:/backup	2018-01-30 19:18:26	2018-01-30 19:18:26	2018-01-10 02:33:05	2018-01-30 19:18:26
4	Directory	[제불] 나루토 1~720 ...	56 Bytes	D:/[제불] 나루토 1~720 完+극장...	2018-01-08 14:49:59	2018-01-08 14:49:59	2017-12-20 14:14:26	2018-01-08 14:50:06
5	File	170703_침해대응14기_...	3716600 Bytes	D:/170703_침해대응14기_Project_v...	2017-11-25 17:31:27	2017-11-25 17:31:27	2017-11-25 17:31:27	2017-11-25 17:31:27
6	Directory	새 폴더	144 Bytes	D:/새 폴더	2018-01-10 02:44:30	2018-01-10 02:44:30	2017-09-06 14:46:50	2018-01-10 02:44:30
7	Directory	Baseball	56 Bytes	D:/Baseball	2017-09-02 16:22:47	2018-02-09 17:51:05	2017-09-02 16:22:00	2017-09-02 16:22:47
8	Directory	Entertainment	56 Bytes	D:/Entertainment	2017-09-02 16:18:45	2017-09-02 16:18:45	2017-09-02 15:20:03	2017-09-02 16:18:45
9	Directory	BoB	208 Bytes	D:/BoB	2018-02-09 17:52:08	2018-02-09 17:51:41	2017-08-13 18:35:42	2018-02-09 17:52:08
10	Directory	bob_bi	56 Bytes	D:/bob_bi	2017-09-12 00:01:07	2017-09-12 00:01:07	2017-08-11 14:46:00	2017-09-12 00:01:07
11	Directory	EnCase 관련	56 Bytes	D:/EnCase 관련	2017-08-23 13:42:34	2017-08-23 13:42:34	2017-08-11 14:28:21	2017-08-23 13:42:34
12	Directory	LG HDD Back Up	56 Bytes	D:/LG HDD Back Up	2017-07-25 22:04:21	2017-07-25 22:04:21	2017-07-25 22:04:04	2017-08-13 18:33:56
13	File	170703_침해대응14기_...	15577549 Bytes	D:/170703_침해대응14기_Project_v...	2017-11-18 03:01:32	2018-02-09 17:29:24	2017-07-22 11:34:55	2017-11-18 03:01:32

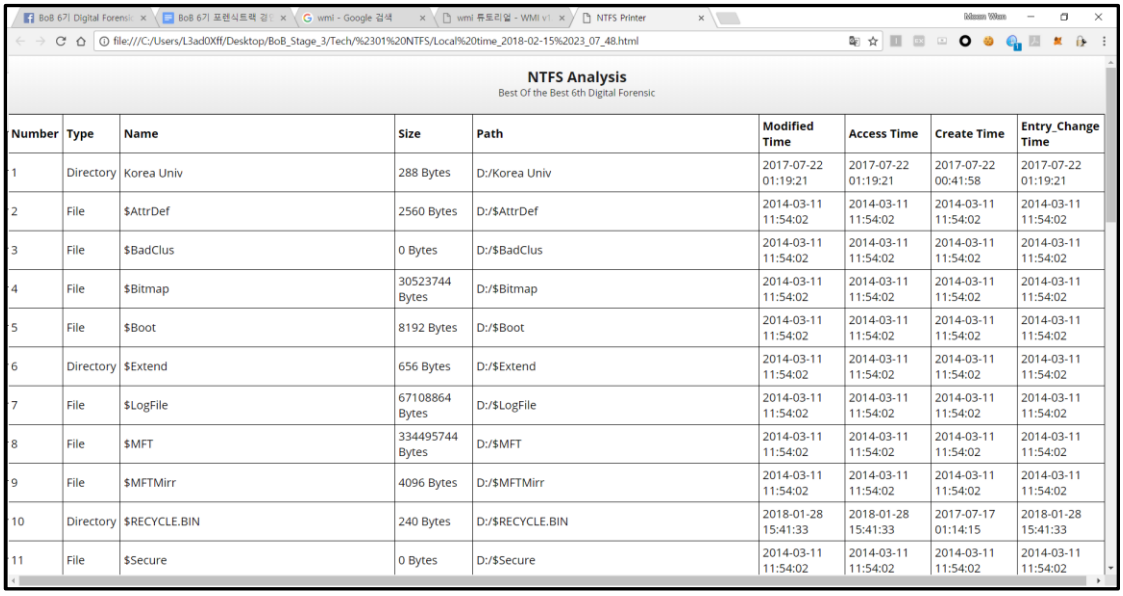
- Create Time을 기준으로 정렬하였으며, 최근 생성 시간 순으로 정렬을 하였다.

➢ Save to html을 이용하여 html 파일을 생성하며, 동작 시 MessageBox Window가 출력되며, 저장 경로 등의 정보를 담고 있다.



	Type	Name	Size	Path	Modified time	Access time	Create time	EntryChange time
1	Directory	Korea Univ(세종)	288 Bytes	D:/Korea Univ(세종)	2017-07-22 01:19:21	2017-07-22 01:19:21	2017-07-22 00:41:58	2017-07-22 01:19:21
2	File	$AttrDef	2560 Bytes	D:/$AttrDef	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
3	File	$BadClus	0 Bytes	D:/$BadClus	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
4	File	$Bitmap	30523744 Bytes	D:/$Bitmap	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
5	File	$Boot	8192 Bytes	D:/$Boot	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
6	Directory	$Extend	656 Bytes	D:/$Extend	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
7	File	$LogFile	67108864 Bytes	D:/$LogFile	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
8	File	$MFT	334495744 Bytes	D:/$MFT	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
9	File	$MFTMirr	4096 Bytes	D:/$MFTMirr	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
10	Directory	$RECYCLE.BIN	240 Bytes	D:/$RECYCLE.BIN	2018-01-28 15:41:33	2018-01-28 15:41:33	2017-07-17 01:14:15	2018-01-28 15:41:33
11	File	$Secure	0 Bytes	D:/$Secure	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
12	File	$UpCase	131072 Bytes	D:/$UpCase	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02

➢ 출력된 HTML 파일은 아래와 같다.



NTFS Analysis

Best Of the Best 6th Digital Forensic

Number	Type	Name	Size	Path	Modified Time	Access Time	Create Time	Entry_Change Time
1	Directory	Korea Univ	288 Bytes	D:/Korea Univ	2017-07-22 01:19:21	2017-07-22 01:19:21	2017-07-22 00:41:58	2017-07-22 01:19:21
2	File	$AttrDef	2560 Bytes	D:/$AttrDef	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
3	File	$BadClus	0 Bytes	D:/$BadClus	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
4	File	$Bitmap	30523744 Bytes	D:/$Bitmap	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
5	File	$Boot	8192 Bytes	D:/$Boot	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
6	Directory	$Extend	656 Bytes	D:/$Extend	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
7	File	$LogFile	67108864 Bytes	D:/$LogFile	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
8	File	$MFT	334495744 Bytes	D:/$MFT	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
9	File	$MFTMirr	4096 Bytes	D:/$MFTMirr	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02
10	Directory	$RECYCLE.BIN	240 Bytes	D:/$RECYCLE.BIN	2018-01-28 15:41:33	2018-01-28 15:41:33	2017-07-17 01:14:15	2018-01-28 15:41:33
11	File	$Secure	0 Bytes	D:/$Secure	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02	2014-03-11 11:54:02

5. 참고문헌

- Pytsk3 : https://github.com/py4n6/pytsk/wiki/Development
- wmi : https://stackoverflow.com/questions/9901792/wmi-win32-diskdrive-to-get-total-sector-on-the-physical-disk-drive/28709238
- NTFS Information : http://forensic-proof.com
- NTFS Info : http://kali-km.tistory.com/entry/NTFS-Python-MFT-Acqusition?category=532592