|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **공개 SW 실무**  보고서 및 논문 윤리 서약   1. 나는 보고서 및 논문의 내용을 조작하지 않겠습니다. 2. 나는 다른 사람의 보고서 미 논문의 내용을 내 것처럼 무단으로 복사하지 않겠습니다. 3. 나는 다른 사람의 보고서 및 논문의 내용을 참고하거나 인용할 시 참고 및 인용 형식을갖추고 출처를 반드시 밝히겠습니다. 4. 나는 보고서 및 논문을 대신하여 작성하도록 청탁하지도 청탁받지도 않겠습니다.   나는 보고서 및 논문 작성 시 위법 행위를 하지 않고, 명지인으로서 또한 공학인으로서 나의 양심EMB0000240825a1과 명예를 지킬 것을 약속합니다.  보고서명 : 공개SW 기능 분석  학 과 : 컴퓨터공학과  담당교수 : 류연승  7조  학 번 : 60152235 이 름 : 장 현 호 (서명)  학 번 : 60172549 이 름 : 홍 지 은 (서명)  학 번 : 60142289 이 름 : 설 다 훈 (서명)  학 번 : 60152196 이 름 : 배 상 우 (서명)  목차   1. 소스의 전체 구조 2. 디렉토리(폴더)의 구조 3. 각 디렉토리(폴더) 내의 파일명 과 각 파일의 내용 4. 기능 설계 5. 기능별 설명 6. 기능별 관련 소스파일, 헤더파일, 클래스명, 함수명 7. 자주 사용되는 모듈 8. 주요 사용자의 인터페이스 (UI)   1) add targets  2) send to webSession  3) send to Bruter   1. 참고 문헌 2. 소스의 전체 구조 3. 디렉토리(폴더)의 구조      1. 각 디렉토리(폴더) 내의 파일명과 각 파일의 내용 2. **assets**     **Images**  android-icon.png closed.gif  filtered.gif freebsd-icon.png  icon.png ios-icon.png  linux-icon.png love.png  macosx-icon.png open.gif  openbsd-icon.png solaris-icon.png  unix-icon.png unknown-icon.png  wall.png windows-icon.png    **ui**  #hostview.glade# logger.glade  add.glade main.glade  bruter.glade servicesview.glade  hostlist.glade websession.glade  hostview.glade. widgets.glade     1. **conf**     bruter.conf  metasploit.conf  shell.conf  shodan.conf  webssession.conf   1. **core**     \_\_init\_\_.py  addtargets.py  database.py  extensions.py  file\_filters.py  icons.py  main.py  widgets.py  workspace.py   1. **extensions**     -**importers**    \_\_init.\_\_.py  geoplugin.py  masscan.py  nmap.py  shodan.py  -**workspace**    \_\_init\_\_.py  browser.py  bruter.py  metasploit.py  screenshot.py  shell.py  web\_session.py   1. **scripts**     finger-user-enum.pl smap.py  ftp-uer-enum.pl sploitus.py  http\_screenshot.js vnc\_screenshot.py  rdp\_screenshot.sh vulners.nse  rtsp\_screenshot.sh x11\_screenshot.sh   1. **wordlists**     **http**  -Top1000-RobotsDisallowed.txt  -Top10000-RobotsDisallowed.txt  pass.txt  snmap-default.txt  subdomains.txt  users.txt  **.gitignore**  **extension**  **license.md**  **resdme.md**  **requirements.txt**   1. 기능 설계 2. 기능별 설명  |  |  |  | | --- | --- | --- | | 기능명 | 서브 기능 | 설명 | | Add targets | nmap | 네트워크상의 컴퓨터를 탐지하고 대상 컴퓨터에 열려있는 포트를 나열한다. | | masscan | Nmap과 유사하며 현재 가장 빠른  인터넷 포트 스캐너이다. | | shodan | 라우터, 스위치, FTP, 특정 웹 서버  (Apache, IIS등)에 대한 정보를 수집하고 그 결과를 보여주는 기능 | | Send to webSession | XSS | 해당 페이지에 스크립트를 집어넣어 실제 코드가 적용되는지 확인하는 기능 | | SQLI | SQL Injection으로 로그인과 같은 인증을 할 경우 Injection공격으로 인증 우회 등 디비를 비정상적으로 조각이 되는지 확인하는 기능 | | LFI | 대상 서버의 URL에 상대경로를 주고 내부파일에 접근이 되는지 확인 기능 | | RCE | 원격으로 쉘코드가 삽인되는 지 확인하는 기능. 부가적으로 현재 user가 무엇인지 확인하는 기능도 있다. | | Send to Bruter | Bruter force attack | 무차별 대입 공격으로 ssh가 열린 포트에 txt파일에 있는 목록들을 대상 서버에 원격으로 접속시도를 하는 기능 |  1. 기능별 관련 소스파일, 헤더파일, 클래스명, 함수명   smap.py   |  |  | | --- | --- | | 기능 | Shodan API가 있다면 있을 경우 shodan을 이용하여 해당 ip에 정보를 수집하여 출력. | | import | import argparse  import sys  import configparser  import ipaddress  import os  import json  import threading | | 변수  및  함수 | **shodanThread :** threading.Thread를 매개변수로 가지는 클래스이다. 밑에 2개 함수를 멤버 함수로 가짐.  **\_\_init\_\_ :** self, threadID, name, targets, verbose=False를 매개변수로 갖는 함수 (초기화 함수)  **run :** self를 매개 변수로 갖는 함수, self.targets 의 target인 동안 실행된다. target 의 ip, os, org, city, region code, ISP, country code, latitude, longtitude, hostnames를 차례대로 출력한다. 만약 self.verbose이면 target과 not found라는 구문을 출력한다.  **chunkify :** lst, n을 매개변수로 가지고 lst[i::n] for i in range(n)을 반환한다.  **config :** configparser.ConfigParser()로 초기화한다.  **parser :** argparse.ArgumentParser()로 초기화한다.  **args :** parser.parse\_args()로 대입한다.  **got\_api :** False를 나타내는 변수  **pt :** chunkify( )함수에 todo,t\_num넣어 나오는 값을 나타내는 변수 |   sploitus.py   |  |  | | --- | --- | | 기능 | Sploitus.com 에 공격 툴을 검색하고 다운받을 수 있는 기능. | | import | import requests  import json  import sys | | 변수  및  함수 | **q :** q 에 sys.argv[1] 를 저장한다  **p :** 'offset': 0, 'query': q, 'sort': 'default', 'title': 'false', 'type': 'exploits' 를 주어 만들어진 객체  **r :** requests.post 에 'https://sploitus.com/search', json=p를 받아 text 로 전달하여 받은 값  **j :** json.loads()에 위의 변수 r을 대입한 값  그리고 '[!] Sploitus made by @i\_bo0om | Full result: https://sploitus.com/?query=' + q + '#exploits' 과 '[!] Found: ' + str(j['exploits\_total']) + ' results'의 내용을 출력한다.  j['exploits'] 의 값을 넣어 e['title'] + e['href'] + tPublished: ' + e['published']의 내용을 출력한다. |   vnc\_screenshot.py   |  |  | | --- | --- | | 기능 | 이미지를 저장하고 블랙 스크린 샷을 제거하는 역할을 한다. | | import | import base64  import gi  import time  import threading  import sys  from gi.repository import GtkVnc  from gi.repository import Gtk | | 변수  및  함수 | **screenshot(dpy)** : dpy라는 매개변수를 가지는 함수, 이 함수는 이미지를 받는 함수이다. 우선 함수에 진입하면 이미지를 받기 위해 time.sleep()을 사용하여 몇 초간 기다린다. 그 후 dpy.get\_pixbuf() 메소드를 사용해서 pix에 받아온다. 그 다음에 pix.savev()를 이용해 "/tmp/badkarma-screenshot"+sys.argv[1]+"-"+sys.argv[2], "png", "", "" 를 넣어 이미지를 저장한다. 그리고 ("/tmp/badkarma-screenshot"+sys.argv[1]+"-"+sys.argv[2], "rb" )를 열고 image변수에 저장한다. base64.b64encode(image.read()).decode()에 값을 출력한다. 마지막으로 Gtk.main\_quit(),quit() 를 사용해서 함수를 마친다.  **vnc\_connected(dpy):** dpy라는 매개변수를 가지는 함수, 검은 스크린샷을 제거하기 위한 함수이다. threading.Thread()에 target=screenshot,args=[dpy]를 넣어서 thread에 저장한다. 그리고 thread.start()을 이용해 함수를 마친다. |   Badkarma.py   |  |  | | --- | --- | | 기능 | 새로운 메뉴, 메타플루트 확장을 하고 버그를 수정하는 기능을 한다. | | import | import signal  import argparse  import gi  from gi.repository import Gtk  from core.main import Handler  from core.database import DB | | 변수  및  함수 | **parser :** argparse.ArgumentParser() 에 값을 저장하는 변수이다.  **args :** parser.parse\_args()에 값을 저장하는 변수이다.  **Database :** args.session이면 DB에 db\_loc=args.session를 넣어주고 아니면 DB에 아무 것도 넣어주지 않은 값을 저장하는 변수이다. |   main.py   |  |  | | --- | --- | | import | import os  import sys  import shutil  import gi  import random  import json  import string  import core.file\_filters as file\_filters | | 변수  및  함수 | **Handler :** 매개변수를 가지지 않는 클래스이다. 총 22개의 함수를 멤버 함수로 가짐.  **\_\_init\_\_ :** self, database를 매개변수로 갖는 함수 (초기화 함수)  **\_sync:** self, reset=False, history=False를 매개 변수로 가지는 함수  **\_clear\_workspace:** self를 매개변수로 가지는 함수  **\_filter\_service :** self, service를 매개변수로 가지는 함수. 구문을 실행시킨 뒤 service를 반환한다.  **\_controller\_switch :**self, widget, test, newpage를 매개변수로 가지는 함수  **\_scope :** self, widget, add, targets를 매개변수로 가지는 함수  **\_showhide\_scope :** self, widget을 매개변수로 가지는 함수  **\_showhide\_logs:** self, widget을 매개변수로 가지는 함수  **\_sensitive\_true**: self, widget, add를 매개변수로 가지는 함수. 매개변수 add가 존재 할 경우 target에 self.add\_window.target\_input.get\_text()를 대입한다.  **add\_target**: self, widget을 매개변수로 가지는 함수. 하나 혹은 여러가지 target을 add한다.  **open\_file**: self, widget을 매개변수로 가지는 함수. project file을 실행시킨다.  **dialog**: Gtk.FileChooserDialog를 대입한다.  **response**: dialog.run()을 대입한다.  **save\_file\_as**: self, widget을 매개변수로 가지는 함수  **import\_file**: self, widget을 매개변수로 가지는 함수. nmap xml을 import한다.  **\_delete\_host**: self, widget, hosts를 매개변수로 가지는 함수. database에서 host를 제거한다.  **services\_row**: self, listbox, cell, listboxrow를 매개변수로 가지는 함수. dictionary를 generate한다.  **\_history\_activated**: self, listbox, cell, listboxrow를 매개변수로 가지는 함수.  **tree\_iter**: model.get\_iter(path)를 대입한다.  **log\_id**: model.get\_value(tree\_iter,0)를 대입한다.  **on\_row\_activated**: self, listbox, cell, listboxrow를 매개변수로 가지는 함수.  **host\_id**: model.get\_value(tree\_iter,5)를 대입한다.  **run\_multi\_extra**: self, widget, targets, ext, service, sub\_item을 매개변수로 가지는 함수  **host\_click**: self, tv event를 매개변수로 가지는 함수  **extra**: self.extensions.get\_extra(extra\_name)를 대입한다.  **mouse\_click**: self, tv, event, alltargets=False를 매개변수로 가지는 함수. Menu와 submenu objects를 생성하고 에러가 없을 시 True를 반환한다.  **end\_task**: self, caller, out, id를 매개변수로 가지는 함수. extension의 task를 끝낼 때 실행된다. 에러가 있을 경우 ‘e’를 출력  **run\_extra**: self, widget, ext, service, sub\_item을 매개변수로 가지는 함수. python extension을 실행한다. 에러가 발생할 경우 구문을 실행하지 않는다.  **port\_string**: str(self.\_selected\_opt["port"])를 대입한다.  **host\_string**: self.\_selected\_opt["host"] 를 대입한다.  **banner\_string**: self.\_selected\_opt["banner"] 를 대입한다.  **ext:** self.extensions.get\_new(ext.name) 를 대입한다.  **out:** ext.task( ext\_conf ) 를 대입한다.  **close\_task\_tab:** self, btn, widget을 매개변수로 가지는 함수. close button을 click할 때 발생하는 이벤트  **on\_window\_delete\_event:** self, \*args를 매개변수로 가지는 함수. 종료 이벤트 |   icons.py   |  |  | | --- | --- | | 기능 | 각 아이콘을 클릭했을 때 이벤트를 커리하는 기능 | | import | Import gi  Import os | | 변수  및  함수 | **Icon:** typed를 매개변수로 가지는 함수.  GdkPixbuf.Pixbuf.new\_from\_file\_at\_scale(filename=os.path.dirname(os.path.abspath(\_\_file\_\_)) + "/../assets/images/"+typed+"-icon.png",  width=24, height=24, preserve\_aspect\_ratio=True)를 반환한다.  **Icon\_lg:** typed를 매개변수로 가지는 함수.  GdkPixbuf.Pixbuf.new\_from\_file\_at\_scale(filename=os.path.dirname(os.path.abspath(\_\_file\_\_)) + "/../assets/images/"+typed+"-icon.png",  width=64, height=64, preserve\_aspect\_ratio=True)를 반환한다.  **port\_open\_icon:** GdkPixbuf.Pixbuf.new\_from\_file\_at\_scale(filename=os.path.dirname(os.path.abspath(\_\_file\_\_)) + "/../assets/images/open.gif",width=14, height=14, preserve\_aspect\_ratio=True)를 반환한다.  **port\_open\_icon:** GdkPixbuf.Pixbuf.new\_from\_file\_at\_scale(filename=os.path.dirname(os.path.abspath(\_\_file\_\_)) + "/../assets/images/open.gif",width=14, height=14, preserve\_aspect\_ratio=True)를 반환한다.  **port\_filtered\_icon:**  GdkPixbuf.Pixbuf.new\_from\_file\_at\_scale(filename=os.path.dirname(os.path.abspath(\_\_file\_\_)) + "/../assets/images/filtered.gif",width=14, height=14, preserve\_aspect\_ratio=True)를 반환한다.  **port\_closed\_icon:** GdkPixbuf.Pixbuf.new\_from\_file\_at\_scale(filename=os.path.dirname(os.path.abspath(\_\_file\_\_)) + "/../assets/images/closed.gif",width=14, height=14, preserve\_aspect\_ratio=True)를 반환한다.  **gtk\_exit\_fullscreen:** image를 매개변수로 가지는 함수.  image.set\_from\_stock(Gtk.STOCK\_LEAVE\_FULLSCREEN, Gtk.IconSize.LARGE\_TOOLBAR)를 반환한다.  **gtk\_fullscreen:** image를 매개변수로 가지는 함수.  **get\_icon:** os, lg=False를 매개변수로 가지는 함수. os배열에 android, ios, linux, windows, solaris, freebsd, openbsd, macos, unix가 존재할 경우 해당 문자열을 icon\_func에 매개변수로 받아 반환시키고 그렇지 않으면 unknown을 반환한다. |   extensions.py   |  |  | | --- | --- | | 기능 | Module 추가 버튼을 눌렀을 때 해당 모듈을 디비에 추가하여 쓸 수 있게 처리하는 기능 | | import | Import os  Import importlib | | 변수  및  함수 | **Importers:** 매개변수를 가지지 않는 클래스. 이하 두개의 함수를 가진다.  **\_\_init\_\_:** self, database를 매개변수로 가지는 함수.  **\_indicize\_modules:** self, database를 매개변수로 가지는 함수. 에러가 발생 할 경우 구문을 실행하지 않는다.  **Extensions:** 매개변수를 가지지 않는 클래스. 이하 5개의 함수를 가진다.  \_\_init\_\_:self를 매개변수로 가지는 함수. extensions를 intialize한다.  **get\_extra:** self, service, all=True를 매개변수로 가지는 함수. returndict를 반환한다.  **get\_extra\_by\_name:** self, name을 매개변수로 가지는 함수  **get\_new:** self, extension\_name을 매개변수로 가지는 함수. Extension\_name == serv일 경우 module을 초기화하고 module.karma\_ext()를 반환한다.  **\_indicize\_modules:** self를 매개변수로 가지는 함수. 에러가 발생하면 구문을 실행하지 않고 True를 반환 |   file\_filters.py   |  |  | | --- | --- | | 기능 | Gi의 GUI를 구성하는 기능. | | import | Import gi | | 변수  및  함수 | **add\_filter\_database:** dialog를 매개변수로 가지는 함. filter\_text, filter\_any의 name을 설정  **add\_filter\_nmap:** dialog를 매개변수로 가지는 함수. filter\_text의 name을 설정한다.  **add\_filter\_txt:** dialog를 매개변수로 가지는 함수. filter\_text, filter\_any의 name을 설정  **filter\_text:** Gtk.FileFilter()를 대입  **filter\_any:** Gtk.FileFilter()를 대입  **filter\_text:** GtkFileFilter()를 대입 |   Addtargets.py   |  |  | | --- | --- | | 기능 | Target ip를 입력받는 기능 | | Import | Import gi  Gi.require\_version(‘Gtk’, ‘3.0’)  From gi.repository import Gtk  Import ipaddress  Import os | | 함수 및 변수 | Class Targetadd  \_\_init\_\_ : 클래스 초기화 함수  \_check\_nmap : nmap checkbox에 체크를 해제할경우를 처리한다.  Add\_host : host의 ip를 입력받는다.\ |   Database.py   |  |  | | --- | --- | | 기능 | 입력받은 ip를 DB에 추가하는 기능 | | Import | from sqlalchemy import Column, ForeignKey, Integer, String, Text, Boolean, Float  from sqlalchemy.ext.declarative import declarative\_base  from sqlalchemy.orm import relationship, sessionmaker  from sqlalchemy import create\_engine  from sqlalchemy import update   |  | | --- | |  | |  | | | 함수 및 변수 | Class targets : targets 설정  Class services : services 설정  Class activity\_log : activity\_log 설정  Class notes : notes 설정  Class DB  \_\_init\_\_ : 클래스 초기화 함수  \_find\_nmap\_service : nmap내에서 service file을 검색한다.  Switch\_scope : host의 scope 상태를 switch한다.  Add\_note : note를 추가한다.  Add\_host : scan하지 않고 host를 추가한다.  Add\_log : database에 activity log entry를 추가한다.  Save\_note : note를 저장한다.  Host\_exist : 포트가 database에 이미 있는지 확인하는 함수  Get\_note : note를 리턴함  Get\_notes : note를 전부 리턴함  Get\_host : host를 리턴함  Get\_hosts : host를 전부 리턴함  Get\_service : service를 리턴함  Get\_service\_uniq : service를 전부 리턴함  Get\_ports\_by\_service : service의 port를 리턴한다.  Get\_ports\_by\_host : host의 port를 리턴한다.  Get\_logs : log를 리턴한다.  Get\_history : history를 리턴한다.  Get\_log\_id : log id를 리턴한다.  Remove\_log : log를 지운다.  Remove\_note : note를 지운다  Remove\_host : host를 지운다  Rename\_note : note이름을 바꾼다 |   Workspace.py   |  |  | | --- | --- | | 기능 | Main GUI 각각의 기능을 하게 하는 파일 | | Import | Import os  Import signal  Import datetime  Import gi  Import copy  From gi.repository import Gtk  from gi.repository import GObject  from gi.repository import Gdk  from gi.repository import GdkPixbuf  from core.extensions import Extensions  from core.widgets import \*  import core.file\_filters as file\_filters  import core.icons as iconslib | | 변수 및 함수 | Class Logger  \_\_init\_\_ : 초기화 함수  \_scroll : auto scroll을 만든다  Refresh : log tree를 새 database로 새로고침한다.  Delete\_log : database에서 log를 지운다.  Response에 ok가 리턴되면 닫고, cancel이 리턴되면 닫지 않는다.  Add\_log : log를 추가한다.  새로운 로그의 id에 -1을 해주고, 리스트의 맨 뒤에 추가한 뒤 로그id를 리턴 한다.  Complete\_log : progress bar를 100%로 설정한다.  \_pulse\_progressbars : progress bar에 running animation이 나오도록 한다.  On\_row\_activated : log list를 만든다.  Open\_log : 로그 화면을 띄운다.  Close\_log\_tab : log notebook tab을 닫는 button의 이벤트를 처리한다.  Mouse\_click : log를 마우스 우클릭 했을때의 이벤트를 처리한다.  Kill\_task : task를 kill 한다  Export\_log : 로그를 txt파일로 만든다.  Class Serviceslist  \_\_init\_\_ : 초기화 함수  Refresh : log tree를 새로고침한다.  Toggle\_scope : scope를 스위칭하고 새로고침한다.  \_search\_service : hostlist에서 host를 찾는다.  Class Hostlist  \_\_init\_\_ : 초기화 함수  Toggle\_scope : scope를 스위칭하고 새로고침한다.  \_search\_host : hostlist에서 host를 찾는다.  Class Serviceview  \_\_init\_\_ : 초기화 함수  Refresh : 새 database로 새로고침한다.  Class Hostview  \_\_init\_\_ : 초기화 함수  \_size\_changed : host view의 scroll size를 바꾼다.  Refresh : 새로고침한다.  Tab\_click\_max : 전체화면으로 구성한다.  Class Main  \_\_init\_\_ : 초기화 함수  \_donate\_url : 후원계좌  \_quit\_menu : popover 메뉴를 숨긴다.  Close\_about : about 메뉴를 닫는다  Show\_about : about 메뉴를 보인다. |   widgets.py   |  |  | | --- | --- | | 기능 | Widget을 생성하는 파일 | | Import | Import sys  Import os  Import gi  Import core.icons as iconslib  from gi.repository import Gtk  from gi.repository import Gdk  from gi.repository import GdkPixbuf  from gi.repository import GtkSource  from gi.repository import OsmGpsMap as osmgpsmap  from gi.repository import Vte  from gi.repository import GtkSource  from gi.repository import Glib  from gi.repository import WebKit2 | | 함수 및 변수 | Class Webview  \_\_init\_\_ : 초기화 함수  \_load\_uri : 새로운 uri를 load한다.  \_load\_changed : address bar를 update한다.  \_reload : 다시 불러온다.  Class SourceView  \_\_init\_\_ : 초기화 함수  Class Termianl  \_\_init\_\_ : 초기화 함수  \_key\_press\_event : Vte terminal의 key press event를 처리한다. (Copy&Paste)  Class host\_information  \_\_init\_\_ : 초기화 함수  Refresh : 새로고침한다.  Class OSM  \_\_init\_\_ : 초기화 함수  Refresh : history를 새로고침한다.  Class ServicesTree  \_\_init\_\_ : 초기화 함수  Refresh : 새로고침한다.  Class PortsTree  \_\_init\_\_ : 초기화 함수  Refresh : 새로고침한다.  Class Notesview  \_\_init\_\_ : 초기화 함수  Refresh : 새로고침한다.  Mouse\_click : note를 우클릭하는 이벤트를 처리한다.  Add\_note : database에 note를 추가한다.  On\_row\_activated : note list를 만든다.  Saver\_note : 노트를 저장한다.  Rename\_note : 노트의 이름을 바꾼다.  \_destroy\_rename\_note : rename note 화면을 닫는다  \_save\_rename\_note : rename note한 것을 저장한다.  Delete\_note : 노트를 지운다. |   geoplugin.py   |  |  | | --- | --- | | 기능 | 해당 ip의 지역을 찾는 소스파일 | | import | import json  from core.database import \* | | 변수  및  함수 | \_\_init\_\_ : 클래스 초기화 해주는 함수 match : geoplugin의 request가 string header에 있을 경우 true 아닐경우 false를 retrun하는 함수  parse : geoplugin output의 값(json으로 되어 있음)을 decoding하는 함수.  decoding 이후 db에 commit |   masscan.py   |  |  | | --- | --- | | 기능 | Masscan의 소스파일 | | import | import json  from core.database import \*  from xml.etree import ElemenTree | | 변수  및  함수 | \_\_init\_\_ : 클래스를 초기화해주는 함수  match : masscan가 string header에 있을 경우 true 아닐경우 false를 retrun하는.함수  parse : masscan output의 값(xml로 되어있음) 을 decoding하는 함수  decoding 이후 db에 commit  parse\_json : masscan output의 값(json로 되어있음) 을 decoding하는 함수.  decoding 이후 db에 commit |   nmap.py   |  |  | | --- | --- | | 기능 | Nmap의 소스파일 | | import | import re  from core.database import \*  from libnmap.parser import NmapParser | | 변수  및  함수 | \_\_init\_\_ : 클래스 초기화 해주는 함수  match : nmap의 request가 string header에 있을 경우 true 아닐경우 false  retrun하는함수  parse : nmap output의 값(xml으로 되어 있음)을 decoding하는 함수.  decoding 이후 db에 commit |   shodan.py   |  |  | | --- | --- | | 기능 | Shodan의 소스파일 | | import | import json  from core.database import \* | | 변수  및  함수 | \_\_init\_\_ : 클래스 초기화 해주는 함수  match : shodan의 request가 string header에 있을 경우 true 아닐경우 false  retrun하는함수  parse : shodan output의 값(json으로 되어 있음)을 decoding하는 함수.  decoding 이후 db에 commit |   browser.py   |  |  | | --- | --- | | 기능 | Port에 따라Brower에 들어간 url 주소를 변경해주는 파일 | | import | import os | | 변수  및  함수 | \_\_init\_\_ : 클래스 초기화 해주는 함수  task : 포트에 따라 host 주소를 http 와 https 로 변형하여 url주소를 return한다. |   bruter.py   |  |  | | --- | --- | | 기능 | Bruter force attack을 기능을 가진 소스파일 | | import | import os  import gi  import signal  import configparser  from gi.repository import Gtk  from gi.repository import Gdk  from gi.repository import GtkSource  from gi.repository import GObject  from core import widgets, file\_filters | | 변수  및  함수 | \_\_init\_\_ : 클래스 초기화(GObject도 초기화) 해주는 함수  get\_log : GObject에 있는 gtk로 window창에 스크롤이 있는 윈도우 창을 생성해주는 함수  task : service, proxychains, autoexec, rhost, rport, outfile, path\_config, path\_wordlist, path\_script 등 선언하여 gtk의 builder에 각 영역을 분류하고 gtk의 liststore에 "afp", "asterisk", "cisco-enable", "cisco", "cvs", "firebird", "ftp", "http-proxy-urlenum", "http-proxy", "http-get","http-head","https-get", "https-head", "icq", "imap", "irc", "ldap", "mssql", "mysql", "ncp", "nntp", "oracle-listener", "oracle-sid", "oracle", "pcanywhere", "pcnfs", "pop3", "postgres", "rdp", "redis", "rexec", "rlogin", "rpcap", "rsh", "rtsp", "s7-300", "sapr3", "sip", "smb","smtp-enum", "smtp", "snmp", "socks5", "ssh", "ssh-key", "svn", "teamspeak", "telnet", "time", "vmauthd", "vnc", "xmpp" 을 할당한다. (Bruter wordlist를 만듬) gtk의 터미널을 생성한다. User\_wordlist와 pass\_wordlist에 path\_wordlist에 있는 값들을 무작위로  대입한다. bruter 공격을 할 때 디폴트 값을 지정한다.  read : output의 값을 그대로 return 한다.  task\_terminated : bruter\_box에 있는 것들을 false로 만들고 widget에 있는 값을  지운다.  bruter\_start : widget에 따라 bruter force process를 실행 시킨다. 터미널에  ‘hydra -t라는 옵션을 디폴트로 줘야 한다. (thread생성)  -e : 아래 3가지 옵션을 사용하기 위해서는 디폴트로 줘야한다.  (조건의 값이 존재(existence)  -r : reverse  -n : none  -s : login\_as  -S : ssl  -F : exit\_on\_first    -L : username을 모를 경우 사용  -l : username을 알 경우 사용  -P : wordlist에 있는 패스워드를 사용  -s : port를 맞출 때 사용  proxychains : proxy를 사용  bruter\_check\_user : widget에 선택한 항목에 따라 user\_wl의 path, open.sensitive값과 user\_box.sensitive값(Boolean)을 반대로 한다.  bruter\_open\_user\_file : file선택하는 창을 열고 user부분에 bruter공격을. 할. 파일을 선택한다.  bruter\_open\_pass\_file : file선택하는 창을 열고 password부분에 bruter  공격을 할 파일을 선택한다. |   metasploit.py   |  |  | | --- | --- | | 기능 | Metasploit의 소스파일 | | import | import os  import gi  import configparser  from gi.repository import Gtk  from gi.repository import Gdk  from gi.repository import GtkSource  from gi.repository import GObject  from core import widgets | | 변수  및  함수 | \_\_init\_\_ : 클래스 초기화 해주는 함수  service\_filter : service에 해당하는 값들을 약어로 바꾼다. (netbios-ssn-> samba, postgresql->postgres, domain->dns, rpcbind->rpc)  indicize : metasploit의 모듈을 색인하는 함수. 해당 디렉토리에 있는 파일들의. 리스트를 탐색하여 modules가 경로에 있다면 metaploit의 경로에 할당한다.  submenu : 전달받은 service가 generic과 hostlist일 경우 false를 리턴하고  아닐경우 현재의 module에 추가해서 리턴한다.  task : 설정값에 따라 설정값을 재할당한다.  task\_terminated : task가 끝났을 경우 해당 task에 할당된 모든 값들을 0으로. 만들고 삭제한다.  read : default로 지정된 shell에 대한 output을 그대로 retrun해준다.  get\_log : scrolledwindow, widget, textview를 생성하서 화면에 나타나게.  설정한 후 scrolledwindow를 return한다. |   screenshot.py   |  |  | | --- | --- | | 기능 | 스크린샷을 찍을 수 있게 하는 소스파일 | | import | import gi  import base64  import subprocess  import threading  from gi.repository import Gtk  from gi.repository.GdkPixbuf import Pixbuf  from gi.repository import GObject  from gi.repository import Gio  from gi repository import GLib | | 변수  및  함수 | \_\_init\_\_ : 클래스 초기화 해주는 함수  task : host, port, proxychains 를 설정값에 따라 할당한다. port에 따라 http, https를 구분하여 url을 재구성하고 service의 값에 따라 cmd에 입력해주는 값이 바뀌게 설정한다.  read : thread를 활성화 시켜 화면에 있는 image를 저장하여 화면에 띄워주는. scolledwindow를 return한다.  read\_screen : 해당 이미지의 값을 decoding한 후 Glib에 추가한다. |   shell.py   |  |  | | --- | --- | | 기능 | GUI의 화면을 실행시킬 수 있는 소스파일 | | import | import os  import gi  import configparser  from gi.repository import Gtk  from gi.repository import Gdk  from gi.repository import GObject  from core import widgets | | 변수  및  함수 | \_\_init\_\_ : 클래스 초기화 해주는 함수  submenu : 설정값을 return해주는 함수  task : 설정값들을 재할당한다. (ext->menu-sel, serv->service, proxychains  ->proxychains, auto\_exec, rhost->rhost 등등)  task\_terminated : task가 끝났을 경우 해당 task에 할당된 모든 값들을 0으로. 만들고 삭제한다.  read : default로 지정된 shell에 대한 output을 그대로 retrun해준다.  get\_log : scrolledwindow, widget, textview를 생성하서 화면에 나타나게 설정한. 후 scrolledwindow를 return한다. |   websession.py   |  |  | | --- | --- | | 기능 | Send to webSession을 실행시킬 수 있는 소스파일 | | import | import os  import gi  import configparser  import random  import string  import subprocess  import threading  import socket  from gi.repository import Gtk  from gi.repository import Gdk  from gi.repository import GObject  from core import widgets | | 변수  및  함수 | \_\_init\_\_ : 클래스 초기화 해주는 함수  task : 설정값들을 재할당한다.( ext->menu-sel, serv->service, proxychains->  proxychains, auto\_exec, rhost->rhost 등등 proxy port는 8080 builder의. 경로는 최상위의 assets/ui/websession.glade로 지정한다. websession-main과 payploads-bar 객체를 생성한다.  copy\_payload : clipboard 에 있는 payload 값을 copy한다.  task\_terminated : mitmdump를 이용.(옵션은 아래와 같다.)  -n : 프록시 포트에 바인딩하지 않는다.  -r : 뒤에 오는 파일에 모든 플로우를 읽는다. 해당 로그를. decoding하여 저장. 이후 dump file을 삭제한다.  read : reader를 default로 설정한다.  get\_log : logger view를. default로 설정한다. scrolledwindow, widget, textview를 생성해서 화면에 나타나게 설정한 후 scrolledwindow를 return한다. |  1. 자주 사용되는 모듈 2. os 모듈 : 운영체제에서 제공되는 기본적인 기능들을 제공한다. 3. string 모듈 : 상수를 정의하는 기능을 제공한다. 4. random 모듈 : 임의의 수는 뽑는 기능을 제공한다. 5. socket 모듈 : 소켓 프로그래밍에 필요한 시스템 콜을 래핑하는 기능을 제공한다. 6. threading 모듈 : thread를 생성하여 실행하는 기능을 제공한다. 7. configparser 모듈 : 원도우의 \*.ini 같은 환경 파일을 다룰 수 있도록 한다.   텍스트 방식의 설정 파일을 쉽게 파싱해 주는 모듈   1. subprocess 모듈 : 외부 명령어 실행, 파이썬 프로그램 내에서 새로운 프로세스 스폰, 입출력 파이프를 연결하여 리턴 코드를 획득할 수 있도록 하는 모듁이다. 다른 언어로 만들어진 프로그램을 통합, 제어 할 수 있게 하는 모듈. 2. json 모듈 : 파이썬 자료형을 json 문자열로 만드는 기능을 제공한다. 3. sys 모듈 : 파이썬 쉘 자체를 컨트롤 할 때 사용할 수 있는 시스템 함수들을   가지고 있다.   1. importlib 모듈 : 임포트 시스템과 상호 작용하기 위한 API를 제공한다. 2. time 모듈 : 시간을 표시하는 함수들을 보유하고 있따. 3. base64 모듈 : 바이너리 데이터를 인쇄 가능한 ASCII 문자로 인코딩하고 그러한 인코딩을 다시 바이너리 데이터로 디코딩하는 기능을 제공합니다. 4. 주요 사용자 인터페이스 5. add targets의 주요 기능     그림 1 메인화면   1. nmap     그림 2 스캔이후 화면  Nmap을 통해 대상 컴퓨터의 열린 포트가 나열된다. (masscan도 같은 결과가 나오며 속도 차이만 난다.)   1. Send to webSession (port의 유형이 http(80) or https(443) 일 경우에만 가능)   스크린샷이(가) 표시된 사진    자동 생성된 설명  그림 3 Send to webSession  스크린샷이(가) 표시된 사진    자동 생성된 설명  그림 4 Send to webSession-2   1. XSS(Cross Site Scripting) : 웹 페이지에 악성 스크립트를 삽입할 수 있는지를 테스트 할 수 있다. 다시 말해 웹페이지 관리자가 아닌 사람이 임의로 사용자의 정보(쿠키, 세션 등)를 탈취하는 기능을 수행 할 위험이 있는지 테스트 할 수 있다.   스크린샷이(가) 표시된 사진    자동 생성된 설명  그림 5 XSS   1. SQLI(SQL Injection) : 응용 프로그램 보안 상의 허점을 의도적으로 이용해, 악의적인 SQL문을 실행되게 함으로써 데이터 베이스를 비정상적으로 조작하는 코드 인젝션 공격 방법으로, 로그인과 같은 인증을 DB쿼리문을 조작하여 우회할 수 있는지를 테스트하는 기능이다.   스크린샷이(가) 표시된 사진    자동 생성된 설명  그림 6 SQLI   1. LFI(Local File Inclusion) : 로컬 파일을 포함시킨다는 것을 의미하며, LFI 취약점이 있다면, 서버에서 제공하는 디렉토리 외 로컬 영역의 디렉토리와 폴더를 열람할 수 있다. 해당 취약점이 있는지 확인하는 기능이다.   스크린샷이(가) 표시된 사진    자동 생성된 설명  그림 7 LFI   1. RCE(Remote code execution) : 다른 기기에서의 임의의 코드 실행을 하도록 만드는 것으로 공격자가 실행중인 프로세스를 통해 해당 기기를 완전히 제어할 수 있다. 해당 기능은 이런한 취약점이 있는지 확인하는 기능이다.   스크린샷이(가) 표시된 사진    자동 생성된 설명  그림 8 RCE   1. Send to Bruter : Bruter Force Attack(무차별 대입 공격) : 특정한 암호를 풀기 위해 무작위로 가능한 모든 값을 대입하는 공격을 의미한다.     그림 9 Send to Bruter 실행화면  Bruter force attack으로 대상 컴퓨터의 계정과 패스워드를 찾았을 때 아래와 같이 화면이 출력이 된다.    그림 10 Bruter force attack 성공화면   1. 참고 문헌 2. hithub repository 주소   -https://github.com/r3vn/badKarma |