Laboratory no 5

Developing programs for injection and exploitation using Python

Basically these are custom programs for a specific well-defined context. Unfortunately, their development requires knowledge from all branches of computer science, which makes many low-level or medium-level hackers use ready-made programs when possible. This solution has a low degree of success but also an increased risk of both detection and infection of the person who uses it. Some mid-level hackers go on analyzing and modifying existing programs of this type.

Obviously, this convenience involves the same disadvantages but to a lesser extent.

As we would discuss in the course a real security professional uses after the situation a combination of existing tools as well as many applications developed by him.

This lab will give you some examples of rapid development for beginners of some applications such as payload or exploit but including the framework presented has all educational role i.e. no value in the tactical field.

However, these are a first step in the primary understanding of the way in which such tools are developed for real cases. Although the examples are on Python due to the ease in development of this language, professionals do not really use it for this purpose.

Since we asked you to be clear references to vulnerabilities if you develop something you will also need

https://pypi.org/project/cvesearch/

so

pip3 install cvesearch

or in pycharm add directly to the current project specific environment

First of all do not forget that the development of load/exploits malware/ransomware is carried out only inside a virtual machine well insulated and in no case on the basic operating system.

Or you can also go on the external HDD approach that I explained to you (but turn off the automount of the additional hdds detected. For windows sons you can test for the post-operating phase and the tool at

https://securityonline.info/crackmapexec-v4-0-pentesting-networks/

Obviously penguins can play by testing/modifying its modules whose codes are in the links on the same page Where we find vulnerable applications or servers to perform testing. In addition to the solutions discussed at the laboratory like metasploitable2/3 or the Red/Blue approach, you can also use the following sources:

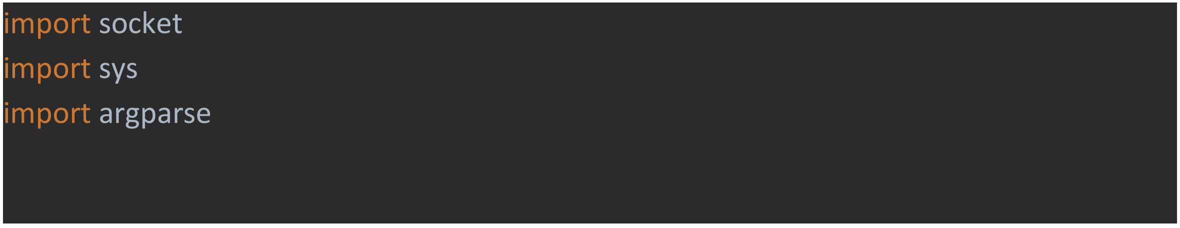
https://www.exploit-db.com/google-hacking-database - for sites

https://www.exploit-db.com/ sites - for applications

If you want to see what concerns a student interested in pentesting looks like maybe you should look at Orange's blog page -

https://blog.orange.tw/.

Anyway, let's test a simple port scanner in Pycharm



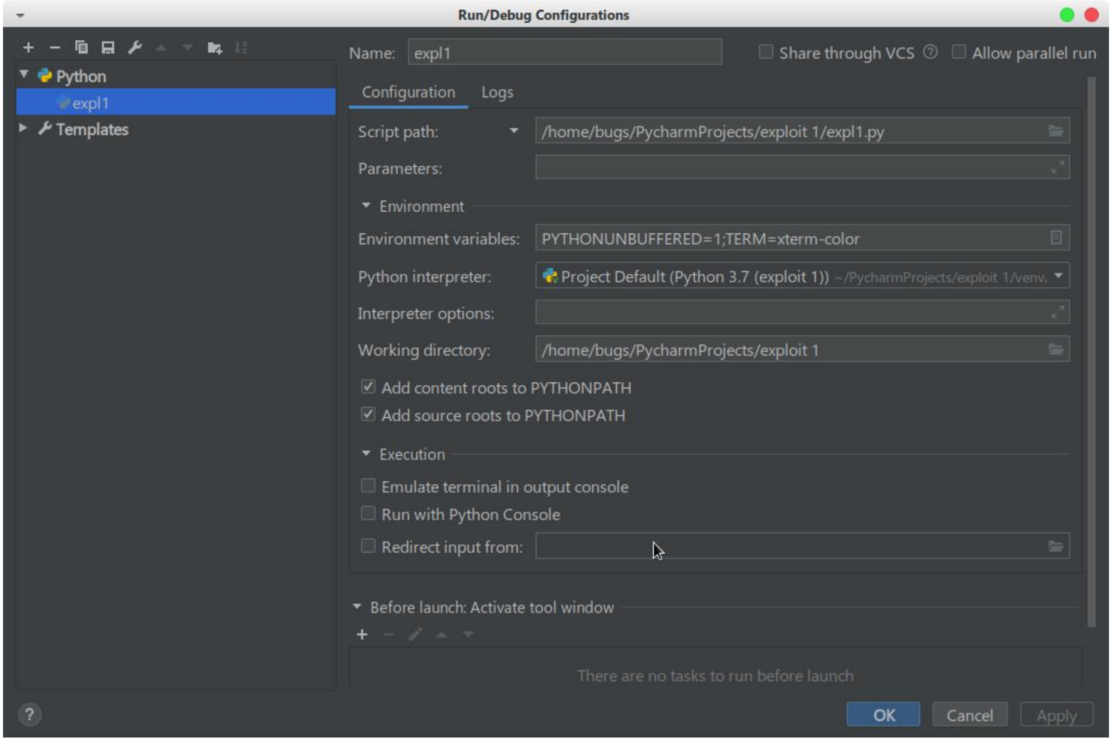
```
import socket
import sys
import argparse
```

```
class PortScan:
    @staticmethod
    def get_args():
        parser = argparse.ArgumentParser()
        parser.add_argument("address", help="addresă IP")
        parser.add_argument("-p", "--ports", nargs='*', help="Porturi pentru analiză " + "Acceptă valor 1,2 sau din gama 1-5")

        return parser.parse_args()

    def _scan(self, ip, ports):
        try:
            for port in ports:
                sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
                result = sock.connect_ex((ip, port))
                if result == 0:
                    print("Port {}: Deschis".format(port))

        except KeyboardInterrupt:
            sys.exit("Analiză teminată")

        except socket.error:
            sys.exit("Nu pot accesa serverul")

        finally:
            sock.close()

    def main(self):
        args = self.get_args()

        ports = list()
        if not args.ports:
            ports = range(1, 1025)
        else:
            for entry in args.ports:
                if '-' in entry:
                    values = [int(x) for x in entry.split('-')]
                    [ports.append(x) for x in range(values[0], values[1] + 1)]
                else:
                    ports.append(int(entry))
            self._scan(args.address, ports)


if __name__ == '__main__':
      port_scan = PortScan()
      port_scan.main()
```

To test the port analysis program you go to configurations and set the list of parameters with the ip of the station 127.0.0.1 -p1-100 (to analyze the first 100 ports) Exploits can also be found at
https://www.exploitalert.com/search-results.html?search=Python

https://www.securitynewspaper.com/2016/11/12/exploiting-python-code-injection-web-applications/

A scholar only example of a worm
For a small worm go to
https://github.com/keniel123/Worm.
The code is self-explanatory. However in testing the program you will encounter some small problems whose resolution is detailed below:
To generate another secret key
openssl rand -base64 16
I think it is obvious that after understanding the code you need to change the network addresses corresponding to the stations and change the activities of the worm to an absolutely harmless level For the error that we did not set the VARIABLE variable to test in Pycharm of the worm server add the variable as below



Name: expl1 ☐ Share through VCS ☐ Allow parallel run

Script path: /home/bugs/PycharmProjects/exploit 1/expl1.py
Parameters:
Environment
Environment variables: PYTHONUNBUFFERED=1;TERM=xterm-color
Python interpreter: Project Default (Python 3.7 (exploit 1)) ~/PycharmProjects/exploit 1/venv,
Interpreter options:
Working directory: /home/bugs/PycharmProjects/exploit 1
☑ Add content roots to PYTHONPATH
☑ Add source roots to PYTHONPATH
Execution
☐ Emulate terminal in output console
☐ Run with Python Console
☐ Redirect input from:
Before launch: Activate tool window
There are no tasks to run before launch

Don't forget pwgen for somewhat more secure passwords

Pwntools is a framework for script kiddies that allows the development of exploits. So they're only good for an educational role and in no case for more serious business. In the case of the RedBlue exercise you can use these resources because at your level of essential expertise is the skill of all the skills necessary for kill chain and not the level of complexity of the techniques and applications used.
At
https://github.com/Gallopsled/pwntools/tree/dev3/examples
you have a number of examples for which I ask you to create separate projects using pycharm (as you add to the current environment a library found in lab 14 from PP)
And the tutorial for pwntools
https://github.com/Gallopsled/pwntools-tutorial
After these examples you can also move on to those from the
http://thecyberrecce.net/
Also here and
https://www.fccomposites.com.mx/roypnirue/wtuds/pwntools-remote
You can also look at https://ocw.cs.pub.ro/courses/cns/labs/start

Homework
If you are from the red team then you have discovered a number of vulnerabilities using the educational tools presented so far. So you can each choose the most promising and try to create a payload/exploit according to the primary information in the CVE database plus additional study on the topic on the Internet.
If you are from the Blue Team and have discovered vulnerabilities on an attacker's station follow the same procedure. If you are not trying to develop using the lab framework a payload.