1: Setting Up Your Python Environment

What is Kali Linux (2015(2))

- Kali is a penetration testing operating system.
- It is based on Debian Linux.
- It is designed by Offensive Security.
- It comes with a number of hacking tools preinstalled.

What is a virtual environment?

- A Virtual Environment, put simply, is an isolated working copy of Python which allows you to work on a specific project without worry of affecting other project
- It enables multiple side-by-side installations of Python, one for each project.
- It doesn't actually install separate copies of Python, but it does provide a clever way to keep different project environments isolated.

What is PyCharm and give some characteristics of PyCharm (2016)

- Pycharm is an python IDE (Integrated developer environment)
- It has a python interpreter
- Debugger
- Bash shell
- Git integration

When setting up a virtual machine in VirtualBox, explain in brief as many system components as you can, that should be defined for the machine (2016)

- Define which os is running on the vm
- Define how big the RAM should be
- Define how much CPU the vm should have
- Define how big the video memory is
- Define hard drive space on vm
- Define the type of network adapter
- Define the shared folder between the host and guest os

2. The Network: Basics

Name all necessary components for making a simple TCP client in Python (2015):

- Import socket
- Define target host (URL/IP) and port number
- client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
  Parameters: AF_INET -> Standard ipv4 or hostname, SOCK_STREAM -> TCP
- client.connect(host, port)
- client.send("http-request")
- client.recv(bytes)

Name all necessary components for making a simple TCP server in Python:

- Import socket, threading
- Define server ip and port
- Create a server socket object
- Start listening - server.bind(ip, port)

- server.listen(n) - Maximum size n of connection backlogs
- Accept incoming requests in a loop:
  - client_socket, addr = server.accept()
  - client_handler = threading.thread(target, args), client_handler.start():
    - client_socket.recv(bytes)
    - client_socket.send("message")
    - client_socket.close()

What is "sys" module of Python used for? Name at least 3 methods (functionalities) that we used in our hack scripts in the lab (2015)

- Can be used to create command line interface and accessing data streams
- sys.argv - Receive command-line arguments
- sys.stdout.write() - write data to terminal
- sys.stdin.read() - read data from terminal
- sys.stderr - read/write error stream
- sys.exit() - exit python

What is "crypto" module of Python used for? Name at least 3 methods (functionalities) that we used in our in the lab

- It is a Python Cryptography Toolkit
- Offers a collection of both secure hash functions and various encryption algorithms.
- Crypto.Cipher.AES.new(key, mode, iv)
- hash = Crypto.Hash.SHA.new(text).digest
- random_generator = Crypto.Random.new().read
- key = Crypto.PublicKey.RSA.generate(len, random_generator)
- key.public_key.encrypt(text, bits)
- key.decrypt(cipher_text)
- key.sign(hash, '')
- key.verify(hash, signature)

Explain how RSA-keys are generated, why is it important to generate the keys using new prime numbers for each key generated?

- p, q = prime_numbers
- n = pq (the modulus)
- $\lambda(n) = \text{lcm}(\lambda(p), \lambda(q)) = \text{lcm}(p-1, q-1)$
- e = integer; st. 1 < e < λ(n) and gcd(e, λ(n)) = 1 (e and λ(n) are coprime)
- $d \equiv e^{-1} \pmod{\lambda(n)}$
  The *public key* consists of the modulus *n* and the public (or encryption) exponent *e*. The *private key* consists of the modulus *n* and the private (or decryption) exponent *d*, which must be kept secret. *p*, *q*, and *λ*(*n*) must also be kept secret because they can be used to calculate *d*.
- If Alice and Bob have some common prime numbers p_a = p_b , then it is easy to compute all the factors:
  - p_a = p_b = GCD(n_a, n_b)
  - q_x = n_x / p_x
  - λ_x = (x_p - 1) * (x_q - 1)
  - d = modinv(x_e, x_d)

What is Netcat and how can it be replaced?

- Netcat is a computer networking utility for reading from and writing to network connections using TCP or UDP. It can be replaced by a custom “black hat python network tool”
- Define parameters i.e: listen, command, upload, execute, target, destination, port
- opts, args = getopt.getopt(sys.argv[1:], regex) - Read command line arguments
- Can send execution file to stdin or use interactively with shell
- Create client socket and connect to server and send data from stdin
- Create server socket, trim payload on arrival to reveal command
- subprocesses.check_output(command, stderr, shell=True) - execute command

Name all necessary component for making simple TCP Proxy server

- A TCP proxy can be used for traffic forwarding, or access network-based software
- $ ./proxy.py 127.0.0.1 21 speedtest.tele2.net 21 True
- $ ftp 127.0.0.1 21
- The server can be runned in a loop:
- server = socket.socket(AF_INET, socket.SOCK_STREAM)
- server.bind((local_host, local_port)
- server.listen(5); while True:
    - client_socket, addr = server.accept()
    - proxy_tread = threading.thread(target, args)
    - proxy_tread.start():
        - remote_socket = socket.socket(...)
        - connection = remote_socket.connect((host, port))
        - response = connection.recv(size) - Receive first data if necessary
        - local_buffer = client_socket.recv(size) - Command from client
        - client_socket.send(response) - Response to client
        - remote_socket.send(local_buffer) - Request to remote server

How can you build a simple SSH client in Python? (2015(2))

- pip install paramiko - We need the python module “Paramiko”
- import paramiko
- ssh_command(ip, user, password, command)
    i. ip - ip address of ssh server
    ii. user - username
    iii. password - password
    iv. command - command that should be executed on successful connection
- client = paramiko.SSHClient()
- client.load_host_keys(key_file) - Supports SSH keys
- client.set_missing_host_key_policy(paramiko.AutoAddPolicy()) - Trust server
- client.connect(ip, username, password)
- ssh_session = client.gettransport().open_session()
- ssh_session.exec_command(command)

Explain how hash functions works, what is it used for and what is a good hash function?

- Hash functions can be used in password management and storage.
- Web sites usually store the hash of a password and not the password
- The hash is generated by the password and compared to stored hash
- A good hash function maps the values uniformly to avoid collisions
- hash = Crypto.Hash.SHA.new(text).digest

Explain encryption algorithms
Encryption algorithms

- Encryption algorithms take some text as input and produce ciphertext using a key.
- You have 2 types of ciphers: block and stream.
- Block ciphers work on blocks of a fixed size (8 or 16 bytes).
- Stream ciphers work byte-by-byte.
- Knowing the key, you can decrypt the ciphertext

Explain public key algorithms: Key generation, encrypt/decrypt and signatures.

- With public-key algorithms, there are two different keys: one to encrypt (the public one) and one to decrypt (the private one).
- You only need to share the public key and only you can decrypt the message with your private decryption key.
- You can also produce digital signatures with your private key

What is Angr, explain its functionalities.

- Multi-architecture binary analysis platform
- Can perform dynamic symbolic execution - (system functions)
- p = angr.Project(binary) - load binary
- cfg = p.analyses.CFG() - Construct control flow graph
- r = p.analyses.ROP() - Generate ROP-Chain
- r.find_gadgets()
- angr.angrize(project) - Generate values to pass input checks on gadgets

In order to speed up the hacking that a function "do_some_hack" is doing we want to run 10 instances of that function in parallel. How can we achieve that in Python? (2015(2))

- import threading
- define number of threads
- define function
- run loop in range(threads):
  - t = threading.thread(target=do_some_hack)
  - t.start()

3. The Network: Raw sockets and Sniffing

What is “os” module of Python used for? Name at least 3 methods (functionalities) that we used in our hack scripts in the lab. (2016)

- The OS module in Python provides a way of using operating system dependent functionality.
- os.environ() - get the user environment
- os.system() - execute shell command
- os.walk(“.”) - walk through files in directory
- os.listdir(“.”) - returns all the files of a directory
- os.name - returns name of os

In the lab exercises we have built a simple packet sniffer in Python. Name some of the main commands used for that sniffer. (2016)

- Done differently on windows and linux. Linux forces you to specify protocol we did this in linux, so linux method is shown below.
- socket_protocol = socket.IPPROTO_IP - Windows ip protocol
- socket_protocol = socket.IPPROTO_ICMP - linux ip protocol
- host = **"10.22.70.158" -** define a host you want to listen to
- sniffer = socket.socket(address family, socket type, socket_protocol) - create new socket
- sniffer.bind((host, 0)) - bind socket to address
- sniffer.setsockopt(level, optname, value) - tell sniffer to include ip header in capt
- sniffer.recvfrom(buffer size) - receive data from the socket

Explain the following Python instruction: sniff(filter="", iface="any" ,prn=function , count=N) (2015)

- This is a function from the python module “Scapy”. It sniffs packets from the network and returns them in a packet list.
- filter = allows us to specify a filter for packets “” = Wireshark BPF style filter
- iface = which network interface to sniff on
- prn = a function that you can do with the data (callback function)
- count = how many packets to sniff 0 = inf

How can you build a simple scanner in python?

a. host = “0.0.0.0” - define a host
b. subnet = “0.0.0.0/24” - subnet we want to target
c. magic_message = “something” - what we want to check ICMP responses for
d. udp_sender(subnet, magic_message) - floods the subnet
    i. sender = socket.socket(addressfamily, socket type)
    ii. for ip in IPNetwork(subnet) - from netaddr import IPNetwork
        1. sender.sendto(magic_message, address)
e. create sniffer that listens to host (see how to create sniffer)
f. t = threading.Thread(target = udp_sender, args = (subnet, magic_message))
g. t.start - start sending packets
    i. raw_buffer = sniffer.recvfrom(address) - read responses
    ii. if ICMP header we want it

iii. icmp_header = ICMP(buffer) - create ICMP structure
iv. check header for type = 3 and code = 3 - indicates a host is up but no available port to talk to

4. Owning the Network with Scapy

How can scapy be used to create a mail sniffer?

a. from scapy.all import *
b. def packet_callback(packet)
    i. if packet[TCP].payload - check to make sure packets has payload
    ii. mail_packet = str(packet[TCP].payload)
    iii. if user or password in mail_packet
        1. print packet[IP].dst - print destination
        2. print packet[TCP].payload
c. sniff(filter = “common mail tcp ports”, prn = packet_callback, store = 0) - tell scapy to start sniffing

What are pcap files and how can one use scapy to analyse them?

- pcap (packet capture files) consists of an API for capturing network traffic.
- Unix-like systems implement pcap in the libpcap library, windows uses a port of this called winPcap
- defragment(pcap)
- p for p in pcap if Raw in p - remove packets that don't have any content

What is “Scapy” module of Python used for? Name at least 3 methods (functionalities) that we used in our hack scripts in the lab.

- Scapy is a packet manipulation tool. It can forge, decode packets for a wide number of protocols. Send them, capture them, match requests and replies and much more.
- Scapy can handle the most classical tasks like scanning, tracerouting, probing, attacks or network discovery.
- sniff(filter = “”, iface = “any”, count = N, prn = callbackfunction)
- get_mac(target_ip) - get mac address
- rdpcap(file.pcap) - read pcap file

How can we sniff 3 packets with scapy and Python? (2015(2))

- from scapy.all import * - import scapy
- def callback_function(packet) - define callback function to show packets
    - print packet.show()
- sniff(prn = callback_function, count = 3) - sniff 3 packets

5. Web Hackery

What is “Urllib2” module of Python used for? Name at least 3 methods (functionalities) that we used in our hack scripts in the lab

What is dojo?

- An open source self-contained training environment based on kubuntu for web application security penetration testing. (Tools + target)
- Includes various web application security testing tools and vulnerable web applications - Such as insecure web app
- For learning and practicing web app security testing techniques.
- Does not need a network connection
- Remote attacks not possible

How can you disguise your browsing as “Googlebot” from Python? (2015 / 2016(2))

- Import urlib2
- url = "http://10.0.2.15" - define target
- headers[‘User-Agent] = “Googlebot” - Create dictionary headers
- request = urllib2.Request(url, headers=headers) - construct the url request
- response = utlib2.urlopen(request) - Contact the url

Explain how the web_app_mapper works

- A scanner that hunts all reachable files on a remote target
- Define threads, target, download directory and filters (i.e file extensions)
- os.chdir(directory)
- web_paths = Queue.Queue() - Where the files will be stored locally
- os.walk() - Searches through all files and directories, adding them to web_paths
- Traverse through web_paths and request the filtered files

Explain the content_bruter

- Use wordlists (from common brute forcers. This replaces the filters from the mapper)
- Define number of threads, target, wordlist_file
- Read the wordlist_file and put all the words in a Queue
- Run all the threads pairing the queue and the extensions:
- Iterate queue, getting all attempts and create requests with urlib2.
- Can for instance print response codes and url for each attempt

Explain the joomla_killer - HTML form authentication brute forcer

- Joomla is a free and open-source content management system for publishing web content.
- Retreive login page and accept cookies urlib2 + cookielib(cookieProcessor)
- Parse the form elements from the HTML
- Set username and password guess from word Queue
- Send HTTP POST to the login to the login processing script including all HTML form fields and the stored cookies
- Compare the response content to a success_check word, i.e “welcome”.

What is XSS, explain persistent/DOM based XSS attacks

- XSS is a vulnerability which when present in websites or web applications, allows malicious users to insert their client side code (normally JavaScript) in those web pages. When this malicious code along with the original webpage gets displayed in the web client, allows Hackers to gain greater access of that page.
- Non-persistent: When XSS code only gets displayed in the next page to the same user and not gets saved into persistent storage like database. This type of attack is less vulnerable, because Hacker can see only their own cookies and can make modifications in their own current opened pages. The risk with these kinds of XSS holes is that it opens way for Cross Site Request Forgery CSRF
- CSRF is a type of malicious exploit of a website whereby unauthorized commands are transmitted from a user that the website trusts. This can be done by placing some hidden links in some bad website.
- The Document Object Model (DOM) is a cross-platform and language-independent application programming interface that treats an HTML, XHTML, or XML document as a tree structure wherein each node is an object representing a part of the document.
- Dom based XSS attack where the attack payload is executed as a result of modifying the DOM “environment” in the victim’s browser used by the original client side script, so that the client side code runs in an “unexpected” manner. This is in contrast to other XSS attacks (stored or reflected), wherein the attack payload is placed in the response page (due to a server side flaw).
- As of 2015, the web servers and web browsers adopted a mandatory parsing of the escape characters

Explain the non-persistent XSS attack and its risks. (2016)

- XSS code gets saved into persistent storage like database with other data and then it is visible to other users also.
- One example of this kind of attacks is possible blog websites, where hacker can add their XSS code along with the comment text and if no validation or filtering is present on the server, XSS code can be successfully saved into the database. After this if anyone (other users) open the page into their browsers, XSS code can execute and can perform a variety of harmful actions. This type of attack is more vulnerable, because Hacker can steal cookies and can make modifications in the page.
- The risk with these kinds of attacks is any third party hacker can use this vulnerability to perform some actions on behalf of other users.

Everything you know about sql injection attacks (2016(2))

- SQL injection attack involves placing SQL statements in the user input in web pages
- Can use try to fire sql error revealing the input query i.e
  SELECT * FROM users WHERE username = “user” and password = “pass”
  Insert query injection to an input field: password = pass’ or ‘1’ = ‘1
- sqlmap is an open source penetration testing tool that automates the process of detecting and exploiting SQL injection flaw

- Preventive measures:
  - Escaping strings - \' or mysql_real_escape_string()
  - Validity of input syntax - valid string in a class/language, exclude problematic chars
  - Limit input length, because many SQL injections depends on entering long strings
  - Insert code scanning for SQL statements
  - Limit database permissions, i.e only read the database with user having read perm
  - segregate users, i.e never connect as database admin in web application

6. Extending the Burp Proxy

What is Burp and what functionality does it offer?

- 'Burp Suite is multi-language tool used for web security testing.'[4] It is designed to be used by hands-on testers as a supporting tool.
- The Burp Suite offers several categories of tools that one can use to perform testing. Some tools are available in the free edition, while most of the functionality is reachable with a pro license.
- The Burp software offers a graphical user interface written in Java. Although one can make extensions with Python, Ruby or Java.
- HTTP Proxy,
- Scanner,
- Intruder,
- Spider,
- Repeater,
- Decoder,
- Comparer,
- Extender and
- Sequencer.

7. Github Command and Control

What is a Trojan Horse?

- Malware that appears to have a desirable function but performs undisclosed malicious functions.
- Running malicious scripts on remote systems are similar to trojan horses

Explain the functionality of BOTNETS

- A botnet is a collection of internet-connected programs communicating with other similar programs to perform tasks.
- could be to:
  - keep control of an IRC channel
  - send spam email
  - participate in DDOS attack

What is JSON?

- Code that is easy for humans to read and write
- easy for machines to parse and write

In the lab we used the git_trojan.py script to perform some simple Trojan actions on the GitHub.com servers. Name some of the basic actions that we performed working with this script. (2016)

- Create module directory in github
    - contains any modular code you want the trojan to pick up and execute
    - dirlister.py
        - exposes a run function that lists all of the files in the current directory
        - returns files as string
    - environment.py
        - retrieves any environment variables set on the infected remote machine
- Create config directory in github
    - holds configuration files that will be uniquely identified for each trojan
    - trojan looks in config directory for TROJANID.json, which will return a simple json doc that we can parse out and convert to python dictionary
    - abc.json
        - simple list of modules we want remote trojan to run
- Create Data directory in github
    - where the trojan checks in collected data (keystrokes, screenshots etc)
- git_trojan.py:
    - “local” trojan script that will initiate the remote scripts to run on the remote machines
    - Connects to repository
    - retrieve config file
    - pull in the two modules set in config file
    - run them

8. Common Trojaning Tasks on Windows

What is spyware, and what are symptoms?

- Spyware is a form of malware that spies on you and sends data back to the intruder.
- Consists of a core (public) functionality and a hidden functionality of information gathering
- Symptoms are high usage of memory and CPU

Explain the main components of creating a simple Python keylogger for linux.

- import os, struct, sys
- os.popen(file_recording_keystrokes)
- Define format and event_size=struct.calcsize(format)
- Define the keyboard layout
- Read from the file recording keystrokes to the log file
- sys.stdout.write(“pressed keys”)

What is a keylogger? (2015(2))

- Software keyloggers track system , collect keystoke data within the target operating system , store them on disk or in remote location , and send them to the attacker who installed the keyloggers. Can be installed by access to the computer, lure victim with virus/trojan or gaining access remotely
- Legal: Monitor for insider attacks (organization), monitor productivity of employees, software developing, backup, personal security
- Non-legal: Espionage, collection of private sensitive information
- Different types: Screen, mouse, keyboard, microphone, camera. Hardware/software
- Check background processes and detect keyloggers: “top” and “ps-aux”
- Use antivirus
- Scan local drive for log files
- Detect unauthorized protocol requests
- Scan incoming messages
- Detect encrypted files transmitted to questionable destinations
- Different kinds: PyKeylogger, simple-key-logger (SKeylogger), Linux KeyLogger (LKL)
- Check for wire extension (hardware). But could be installed inside or built in the device

How can we start a simple http server that listens the port 8080 with a single command from a terminal invoking the python interpreter? Explain the command (2016)

- python -m SimpleHttpServer 8080
    - -m = module
    - SimpleHttpServer = a python module
    - 8080 = portnumber